AUTOMATIZACIÓN

CONTROL ELECTROMECÁNICO

CONTROL NEUMÁTICO

CONTROL ELECTRONEUMÁTICO

CONTROL PROGRAMABLE PLC
GE FANUC

AUTORES:
MTRO. GUSTAVO GUTIÉRREZ CORONA
MTRO. ALBERTO DE LA MORA GÁLVEZ
MTRO. ENRIQUE GALVÁN MORALES
MTRO. ROBERTO CÁRDENAS RODRÍGUEZ

Autores de este libro

Gustavo Gutiérrez Corona
Egresado de la Universidad de Guadalajara en la carrera de Ingeniero en Comunicaciones y electrónica 1974-1979.
Estudios de postgrado en la alta escuela de Ingeniería (Hoch schoole Bremenhaven) en la Republica Federal de Alemania (1987-1989).
Estudios de maestría en el Centro Universitario de Ciencias Exactas e Ingenierías (1998- 2000).
Dir. Gral. y fundador de la de la empresa ASIP control y automatización desde 1990 a la fecha.
Secretario General del Colegio de Ingenieros en Comunicaciones y electrónica del Estado de Jalisco
Maestro del departamento de electrónica del C.U.C.E.I. desde 1983 a la fecha.

Alberto de la Mora Galvéz
Egresado de la Universidad de Guadalajara en la carrera de Ingeniero en Comunicaciones y electrónica 1976-1981
Estudios de maestría en el Centro Universitario de Ciencias Exactas e Ingenierías (1997-1999).
Secretario de la División Ciencias básicas del C.U.C.E.I
Director de la División de electrónica y Computación del C.U.C.E.I
Presidente del Colegio de Comunicaciones y Electrónica del Estado de Jalisco
Maestro del departamento de electrónica del C.U.C.E.I. desde 1981 a la fecha.

Enrique Galván Morales
Egresado de la Universidad de Guadalajara en la carrera de ingeniero en comunicaciones y electrónica 1971-1976
Estudios de maestría en el Centro Universitario de Ciencias Exactas e Ingenierías (1998- 2000).
Director General de la empresa CompuRight desde 1990
Maestro del departamento de electrónica del C.U.C.E.I. desde 1978 a la fecha.

Roberto Cárdenas Rodríguez
Egresado de la Universidad de Guadalajara en la carrera de Ingeniero en Comunicaciones y electrónica 1973-1978
Estudios de maestría en el Centro Universitario de Ciencias Exactas e Ingenierías (1999-2001).
Jefe del departamento de electrónica del C.U.C.E.I
Maestro del departamento de electrónica del C.U.C.E.I. desde 1980 a la fecha.

1° Edición 5 de Marzo del 2007
2° Edición 24 Marzo del 2008
3° Edición 22 de Septiembre del 2008
4° Edición 16 de Julio del 2009
5° Edición 28 de Enero del 2010
6° Edición 25 de Septiembre del 2010

Un libro que guía un aprendizaje más que libro es un amigo, y tal parece que este es el verdadero motivo de los autores de esta obra. Un libro que se antoja complicado y elevado en los conceptos que se tratan, pero la forma en que se exponen los contenidos y los ejemplos lo hacen un documento de mucha ayuda para todos los estudiantes que requieren repasar los temas que, no solo en el Centro Universitario de Ciencias Exactas e Ingenierías se abordan, sino en general para todos aquellos estudiosos de la automatización industrial.

Se requiere constancia y paciencia, pero sobre todo perseverancia, no solo para escribir una obra técnica, sino para exponerla en forma clara y concisa, es el espíritu con el que este libro se desarrolla, los estudiantes de estos temas serán los primeros beneficiados, al contar con un instrumento de aprendizaje eficaz.

La cantidad de ejemplos mostrados, las figuras que lo ilustran y las forma sencilla y practica en que son expuestos los temas lo hacen fácil de seguir y entender, el limite son las ideas de los lectores al concebir diferentes formas de aplicar lo estudiado y potenciar sus capacidades en el diseño de automatizaciones industriales.

El hombre conjuga con la automatización a dos de sus mejores dones: La visión e ingenio de su mente y la destreza creadora de sus manos, permitiéndose con ello, ser más productivo, más libre y más humano.

Mtro. Roberto Cárdenas Rodríguez

Automatización

AUTORES:
Mtro. GUSTAVO GUTIÉRREZ CORONA
Mtro. ALBERTO DE LA MORA GÁLVEZ
Mtro. ENRIQUE GALVÁN MORALES
Mtro. ROBERTO CÁRDENAS RODRÍGUEZ

Prologo

Un libro que guía el aprendizaje más que libro es un amigo, y este es el verdadero motivo de los autores de esta obra. Un libro que se antoja complicado y elevado en los conceptos que se tratan, pero la forma en que se exponen los contenidos y los ejemplos lo hacen un documento de apoyo para los estudiantes que requieren repasar los temas que, no solo en el Centro Universitario de Ciencias Exactas e Ingenierías se abordan, sino en general para todos aquellos estudiosos de la automatización industrial.

Se requiere constancia y paciencia, pero sobre todo perseverancia, no solo para escribir una obra técnica, sino para exponerla en forma clara y concisa, es el espíritu con el que este libro se desarrolla, los estudiantes de estos temas serán los primeros beneficiados, al contar con un instrumento de aprendizaje eficaz.

La cantidad de ejemplos mostrados, las figuras que lo ilustran y la forma sencilla y práctica en que son expuestos los temas lo hacen fácil de seguir y entender, el límite son las ideas de los lectores al concebir diferentes formas de aplicar lo estudiado y potenciar sus capacidades en el diseño de automatizaciones industriales.

Recomendamos no saltarse los primeros capítulos, en ellos se explica los fundamentos para comenzar a automatizar procesos sencillos con control electromecánico y paulatinamente en cuanto se van introduciendo nuevos elementos de control, éstos, se van utilizando para realizar automatizaciones más completas.

Si quiere convertirse en un experto en automatización recuerde que este es un buen principio y no olvidar que para automatizar solo se requieren dos cosas; intuición lógica y práctica. La intuición depende de cada uno de ustedes y para lograr esa intuición se requiere el conocimiento de los diferentes dispositivos que se tienen en el mercado para poder utilizarlos, esta es la función de este libro, y la práctica que se adquiere con el tiempo.

Obras bastante buenas y extensas existen en el mercado que nos enseñan a calcular, diseñar, interpretar e implementar circuitos de control; por lo tanto no espere encontrar en el presente trabajo, ni cálculos complicados, ni teoría extensa o profunda, sino tan solo, una breve descripción de los diferentes elementos que intervienen en una automatización y como se les aplica.

Mtro. Alberto de la Mora Gálvez.

Introducción

Esta sexta edición del libro se ha enriquecido con el capitulo 5 en donde se aborda el control programable de la marca Allen Bradley que es una de las marcas lideres en Automatización a nivel mundial y principalmente en el País por lo que consideramos será de gran utilidad a los estudiantes e interesados que aborden esta abra, el resto de la obra se ha enriquecido con ejercicios y la retroalimentación de alumnos y maestros de la academia, a los cuales les extendemos un sincero agradecimiento por sus aportaciones, se mantiene el objetivo primordial para servir de guía de aprendizaje y consulta del programa de la materia de automatización que se imparte en el Departamento de Electrónica del Centro Universitario de Ciencias Exactas e Ingenierías de la Universidad de Guadalajara.

Pretendemos que este sea su instrumento de aprendizaje y trabajo más preciado. Por ello, el primer requisito para su utilización eficaz es tenerlo al lado cuando trabaje, investigue o se divierta utilizando los conceptos que aquí se muestran para la automatización.

Incluye una gran cantidad de ejemplos prácticos y atractivos, el lector podrá encontrar por lo menos un ejercicio o más por cada uno de los elementos utilizados en la presente obra y que se utilizan en la automatización. A través de ellos le será mucho más fácil la comprensión de sus funciones y sus opciones.

En el capítulo 1 se dan las bases para el desarrollo de circuitos combinacionales utilizando control electromecánico y es el punto de partida para el objetivo principal de esta obra que es el control programable, considerando, además, que en la industria nacional se encuentran muchos equipos con control electromecánico a los cuales se les debe dar servicio, por otro lado, hay circuitos que por seguridad deben ser cableados y no programados, las normas y el correcto uso de las mismas para la elaboración de diagramas de escalera ya es un lenguaje común y mundialmente interpretable, las bases para el desarrollo de circuitos secuénciales con control electromecánico además se toma como punto de partida para lograr la introducción al control programable también se abordan en este capítulo.

El trabajo repetitivo en la industria lo proporciona la fuerza neumática que puede realizar muchas funciones mejor y más rápidamente que la destreza manual y la fuerza humana, de forma más regular y sobre todo durante más tiempo sin sufrir los efectos de la fatiga, aspectos que son tratados en el capítulo 2.

Dado que en la actualidad, la mayoría de las automatizaciones son realizadas por medio de control electro neumático, esto es, la parte de potencia es realizada por equipo neumático y la parte de control por medios eléctricos o electrónicos, en el capítulo 3 se expone las bases de este tipo de control.

En el capítulo 4 se establecen las bases del control programable mediante PLCs, dando una introducción muy clara de las características generales de los mismos e introduciendo al lector a la familia de PLC's de la marca GE Fanuc de la serie micro 90, 90-20 y 90-30**.**

Las formas como se programan las partes de la arquitectura del PLC Marca Ge Fanuc, con ejemplos prácticos y sencillos, desde circuitos combinacionales, circuitos donde intervienen las

memorias, circuitos donde interviene el tiempo, circuitos utilizando contadores, etc., hasta los circuitos con registros de corrimiento se explican ampliamente en el capítulo 6.

También encontrará algunos consejos que le harán explotar toda la potencia de su imaginación y le harán aceleran el proceso de diseño de automatizaciones industriales.

Es el libro para aprender el proceso de automatización industrial, le recomendamos no saltarse los primeros capítulos, en ellos se explica los fundamentos para comenzar a automatizar procesos sencillos con control electromecánico y paulatinamente en cuanto se van introduciendo nuevos elementos de control éstos, se van utilizando para realizar automatizaciones más completas.

No es pretensión de este trabajo sustituir a alguna otra obra en algún momento, sino complementarla simplemente. Es nuestro mayor deseo que el estudiante la encuentre útil, dado que se ha tratado de suprimir todo lo que se refiere a teoría ya explicada en otros libros para evitar ser una repetición o que por su complejidad, sería complicado enmarcarla en un trabajo de esta magnitud. Nos hemos propuesto de ésta, una guía de rápida consulta que nos proporcione la información necesaria para conocer y realizar pequeñas automatizaciones que nos sirvan de base para realizar automatizaciones más complejas.

Sirva entonces como complemento simplemente a tantas y tan buenas obras existentes.

Atentamente:
Los autores

CAPÍTULO 1 CONTROL ELECTROMECÁNICO

Introducción:

En este capítulo vamos a dar las bases para el desarrollo de ***circuitos combinacionales y circuitos secuenciales*** con control electromecánico. Además nos servirá como punto de partida para el objetivo principal de esta obra que es el control programable.

También es importante aclarar que en la industria nacional se encuentran muchos equipos con control electromecánico a los cuales se les debe dar servicio, y además de que hay circuitos que por seguridad deben ser cableados y no programados.

También en este capítulo veremos por que es importante el correcto uso de las normas para la elaboración de diagramas de escalera, de tal manera que sea un lenguaje común y mundialmente interpretable.

1.1 Estructura de un relevador

En la elaboración de diagramas de escalera se utilizan principalmente relevadores, temporizadores, contadores, finales de carrera, árboles de levas etc. sin embargo uno de los elementos fundamentales para la realización de los mismos son los relevadores, y estos sin importar el fabricante, modelo, cantidad de contactos o el voltaje aplicado a la bobina, están compuestos fundamentalmente por cinco partes como son: Bobina, núcleo, armadura, resorte y contactos.

Una representación de los mismos los podemos ver en las siguientes figuras 1.1.1, 1.1.2, 1.1.3 y 1.1.4, 1.1.4a, 1.1.4b.

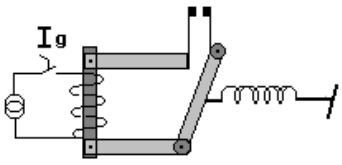


Figura 1.1.1

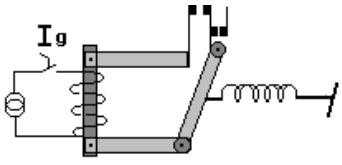


Figura 1.1.2

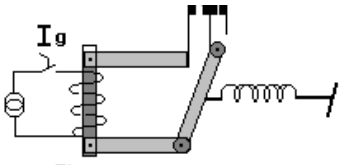


Figura 1.1.3

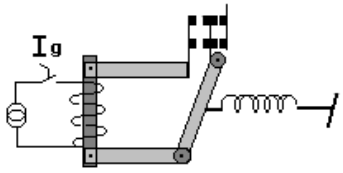


Figura 1.1.4

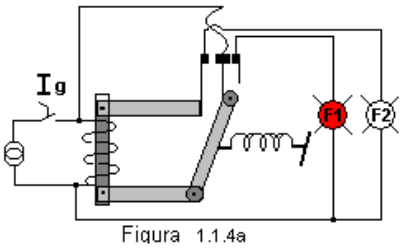


Figura 1.1.4a

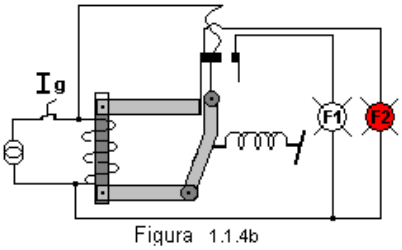


Figura 1.1.4b

Figuras 1.1.1, 1.1.2, 1.1.3, 1.1.4, 1.1.4a, 1.1.4b.

En la figura 1.1.1 podemos observar un relevador con un contacto normalmente abierto. En la figura 1.1.2 se tiene un relevador con un contacto abierto y uno cerrado (independientes), en la figura 1.1.3 se tiene un relevador con un común para un cerrado y un abierto y en la figura 1.1.4 se tiene un relevador con dos comunes, dos abiertos y dos cerrados, finalmente en la figura 1.1.4a se tiene conectado al relevador dos lámparas F1 y F2, donde F1 se encuentra encendida sin estar energizado la bobina del relevador mientras que F2 se encuentra apagada, en el momento que

cerramos el interruptor Ig se forma un campo magnético que provoca un cambio de posiciones de los contactos apagándose la lámpara F1 y encendiendo la lámpara F2, observe la figura 1.1.4b, finalmente cuando abrimos de nuevo el interruptor Ig regresa el relevador a su estado original tal como lo muestra la figura 1.1.4a.

Aún cuando se ha mencionado las partes de que está compuesto un relevador, hay tres puntos importantes que los diferencian, y son:

a.- El tipo de voltaje que se aplica a la bobina,
b.- La cantidad de contactos con que cuenta el mismo
c.- La capacidad de corriente que puede permitir al circular por sus contactos.

El voltaje aplicado a la bobina puede ser de corriente alterna o de corriente directa y los podemos encontrar en el mercado desde (3, 6, 12, 24, 48, 110, 220) Volts.

También podemos encontrar relevadores con un contacto normalmente abierto o con un contacto normalmente cerrado (también conocido como contactos secos), o con un contacto abierto y un contacto cerrado, o bien con un contacto común para el abierto y para el cerrado,

Los podemos encontrar con dos contactos abiertos y dos cerrados teniendo un común para cada juego de contactos y así sucesivamente hasta 90 juegos de contactos con sus comunes, aunque se hace la aclaración de que este tipo de relevadores fueron muy utilizados en circuitos de conmutación en sistemas telefónicos los cuales fueron sustituidos por sistemas electrónicos.

NOTA: La palabra normalmente abierto o normalmente cerrado se refiere al relevador sin energizar.

Cuando se mencionan los contactos es muy importante aclarar la cantidad de corriente que circulara por los contactos y de acuerdo a esa información, seleccionar el adecuado, por lo tanto lo podemos encontrar con capacidades desde miliamperes hasta miles de amperes.

Cuando se utilizan los relevadores para el arranque de un motor trifásico (con contactos secos para cada una de las fases) se les llaman ***contactores*** y si, a este, se le coloca el elemento de protección o termo magnético se le llama ***arrancador***.

1.1.1.- Funcionamiento del relevador:

Al cerrar el Interruptor Ig, (ver las figuras 1.1.1, 1.1.2, 1.1.3 1.1.4 1.1.4a 1.1.4b) se forma un campo magnético en la bobina, este campo magnético genera una fuerza magnetomotriz que a su vez trata de reducir al mínimo su reluctancia (oposición al campo magnético), y como podemos observar, el resorte se opone al campo magnético, de tal manera que si esta fuerza magnética es mayor a la oposición que esta ejerciendo el resorte, la armadura se desplazará de tal manera que el contacto que estaba abierto se cerrará y el contacto que estaba cerrado se abrirá y al abrir el interruptor se pierde la fuerza del campo magnético y el resorte lo regresará al estado original.

En las figuras 1.1.1, 1.1.2, 1.1.3 1.1.4 1.1.4a 1.1.4b se han presentado algunos relevadores sencillos pero los podemos encontrar con una serie de contactos que pueden alcanzar hasta 90 contactos abiertos y 90 cerrados.

1.2 Simbología

En la industria se pueden encontrar dispositivos de fabricación tanto europea como americana, es por ello importante describir los símbolos para ambos sistemas. En la figura 1.2.1 se observan los diferentes arreglos de contactos que podemos encontrar en un relevador.

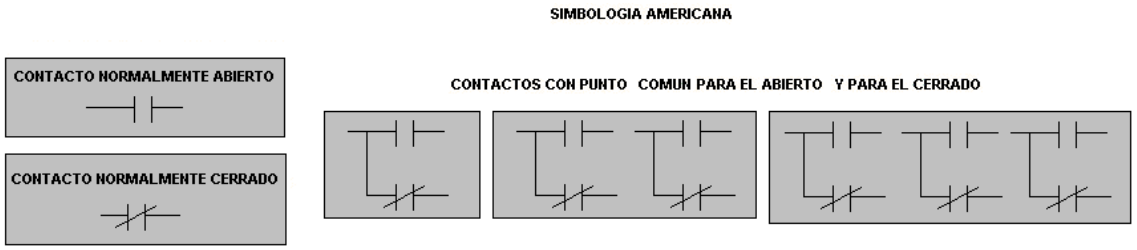
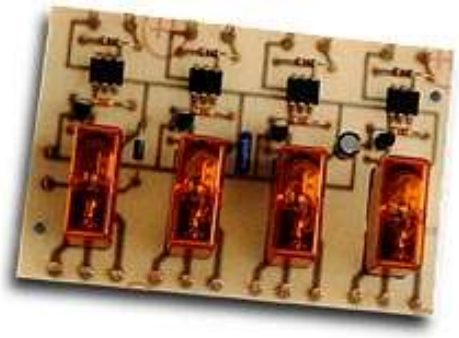


Figura 1.2.1

A simple vista podemos decir que este relevador se puede sustituir por un botón que tenga muchos juegos de contactos, si bien es cierto, la función del relevador es mucho mas importante ya que con el podemos generar memorias y con ellas secuencias automáticas con lo que ya el botón no nos podría dar dicha información.

Aquí se presenta una fotografía de cuatro relevadores montados en un circuito impreso



En la simbología americana la bobina se representa por un círculo y para diferenciar una bobina de otra se les ponen nombres por ejemplo R1 para el primer relevador y R2 para el segundo y así sucesivamente.

Para los contactos abiertos se ponen dos líneas paralelas y para los contactos cerrados se ponen dos líneas paralelas con una diagonal que lo cruza indicando que es cerrado.

En la simbología europea la bobina se representa por un rectángulo y para diferenciarla de otra bobina se le ponen iniciales por ejemplo R1 para el primer relevador y R2 para el segundo y así sucesivamente.

En ambas simbologías tanto la americana como la europea, para identificar las patitas del relevador se le ponen números, haciendo la aclaración que en la simbología europea para identificar las patitas de la bobina se ponen las letras (a, b).

En la figura 1.2.2 se muestra la simbología americana y europea.

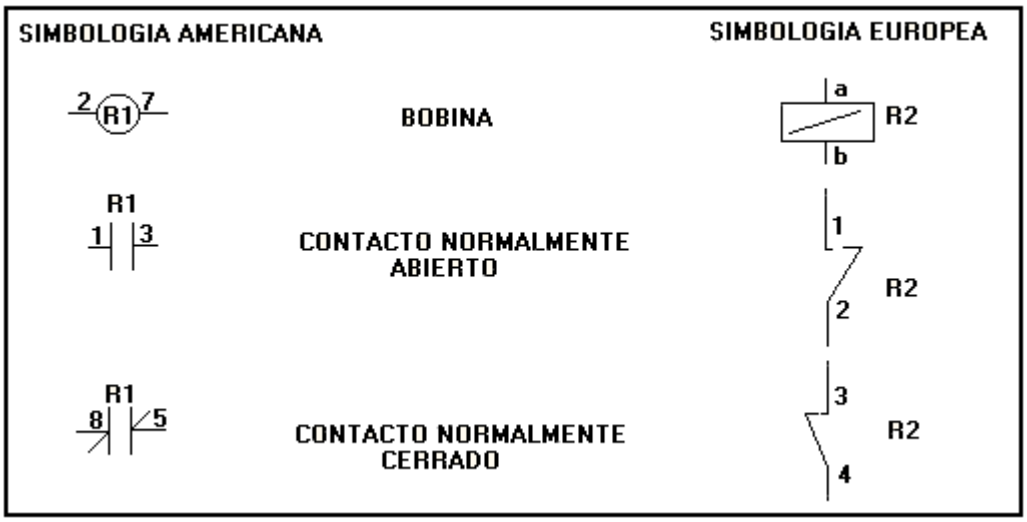


Figura 1.2.2

Cuando un contacto se encuentra solo o de forma independiente (sin conexión en común con otro contacto), se dice que es un contacto seco.

1.3 Normatividad

Normas que se deben seguir para la elaboración de un diagrama de escalera.

A continuación vamos a representar en un diagrama de escalera, el funcionamiento del relevador con simbología americana y europea. Ver figura 1.3.1.

Observe como los diagramas americanos se diseñan en forma vertical, mientras que en diagramas europeos se diseña en forma horizontal.

(Simbología americana) (Simbología europea)

Sin activar B1 Activando B1 Sin activar B1 Activando B1

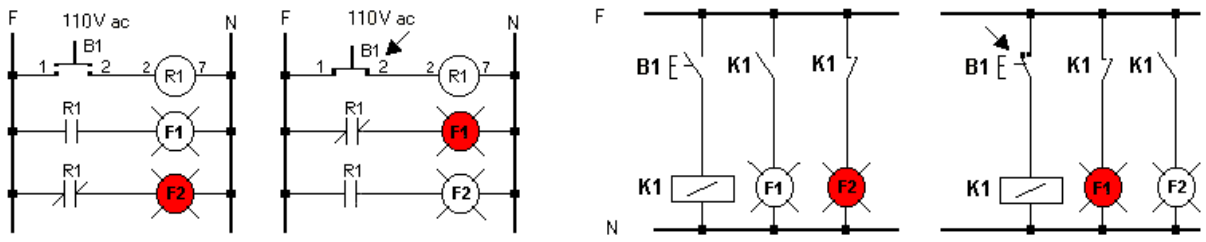


Figura 1.3.1

Como se puede observar en la figura 1.3.1 cuando se tiene sin activar el botón B1, el relevador R1 se encuentra desenergizado y el foco F2 se encuentra energizado; luego, al oprimir el botón B1 se energiza la bobina del relevador R1 y cambian de estado sus contactos y el foco F2 se apaga al abrirse el contacto cerrado y el foco F1 se enciende ya que el contacto abierto de R1 se cierra, y al soltar el botón B1 se desenergiza el relevador R1 regresando a su estado original, esto es F2 se enciende de nuevo y F1 se apaga.

Para que esto quede un poco más claro, vamos a poner un ejemplo: ver figura 1.3.2

Se tienen dos botones y se tienen cuatro focos, los cuales deberán prender en las siguientes condiciones:

1.- Si los dos botones se encuentran abiertos (sin activar) se enciende el foco número uno.
2.- Si se activa el botón uno se enciende el foco número dos y se apaga el uno.
3.- Si se activa el botón dos se apaga el foco dos y se enciende el foco número tres.
4.- Si se activan los dos botones se apaga el foco número tres y se enciende el foco número cuatro.

Como se puede observar nunca podrán encender más de un foco y siempre deberá estar encendido al menos uno.

NOTA:

Para este ejemplo nos están pidiendo que se resuelva el problema con botones que tengan un contacto abierto solamente, por lo tanto cuando nos dicen que si los dos botones están abiertos, deberá prender el foco uno inmediatamente podemos decir que esto seria imposible, pero vamos a suponer que se tienen botones con varios contactos abiertos y cerrados, con lo cual una solución seria como se describe paso a paso en la figura 1.3.2:

NOTA:

En el siguiente ejercicio y en los posteriores utilizaremos sólo simbología americana.

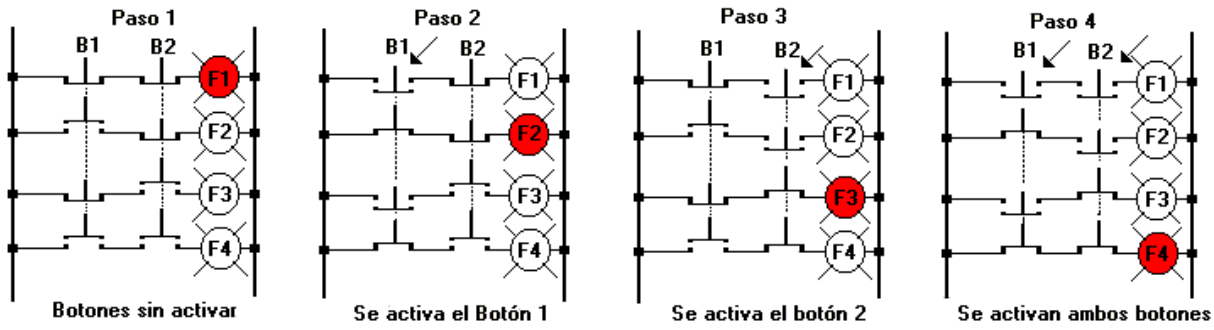


Figura 1.3.2

Aquí podemos observar que en sistemas donde no se requiere secuencia se pueden resolver algunos problemas de automatización con botones solamente, pero hay que tomar en cuenta que cuando son muchas las variables no es muy común que se tengan botones con gran cantidad de contactos.

Regresando al funcionamiento del relevador imaginemos que se tiene un problema como el anterior con botones con un solo contacto abierto, y que tengamos que utilizar relevadores, si para este ejemplo tuviéramos que dibujar el relevador tal como lo muestran las figuras 1.1.1, 1.1.2, 1.1.3, y 1.1.4 seria bastante engorroso, es por ello que se desarrollo cierta simbología para facilitar el trabajo.

1.4 Circuitos combinacionales.

Los circuitos combinacionales son aquellos en los que sus salidas dependen única y exclusivamente de las entradas sin importar la secuencia en que se introdujeron dichas señales y su respuesta es inmediata, esto es, son circuitos donde no interviene el tiempo.

Para la elaboración de circuitos electromecánicos de control, o control convencional o control cableado, como también suele ser conocido, se usan los diagramas de escalera, los cuales sirven para mostrar de forma clara y ordenada diagramas de control, un ejemplo del formato para un diagrama de escalera se muestra a continuación:

Primeramente se colocan dos rieles sobre los cuales se coloca el diferencial de potencial tal como lo muestra la figura 1.4.1 Este diferencial de potencial puede ser de corriente directa o de alterna, y al voltaje que se desee trabajar el cual puede ser de (220, 110, 48, 24, 12, 6 o 3 Volts de directa o de alterna). En esta figura 1.4.1 se muestran los rieles a los cuales se les ha aplicado el diferencial de 110 Volts de a.c. Luego se colocan los eslabones los cuales se representan en este caso como líneas punteadas tal como se observa en la fig. 1.4.2 y sobre estos eslabones se colocan los elementos de control. Como puede observarse parece una escalera y esta es la razón de que a estos circuitos se les llame diagramas de escalera. En la figura 1.4.3. se han colocado algunos elementos de control sobre los eslabones para formar un diagrama de escalera.

Funcionamiento del circuito 1.4.3. Al oprimir el botón 1 se energiza el relevador R1, éste a su vez cierra el contacto abierto energizando la bobina del relevador R3 el cual a su vez cierra su contacto abierto de R3 sosteniéndose, de tal manera que al soltar el botón B1 el relevador R3 continúa energizado. Luego al oprimir el botón B2 se energiza la bobina del relevador R2 abriendo el contacto cerrado con lo cual se desenergiza R3 y al soltar el botón B2 se desenergiza el relevador R2 y el contacto que estaba abierto regresa a su estado original que es normalmente cerrado.

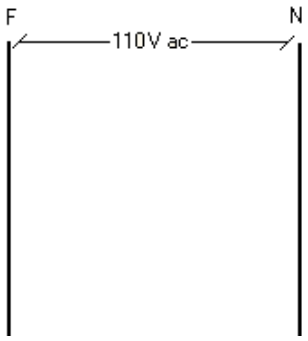


Figura 1.4.1

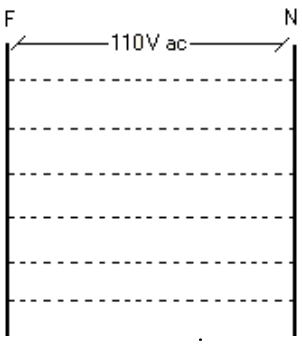


Figura 1.4.2

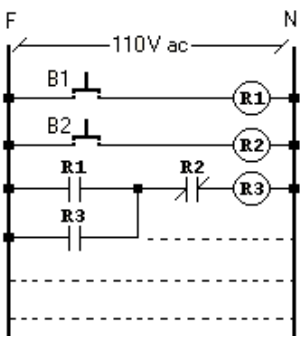


Figura 1.4.3

1.4.1 Ejemplos con circuitos combinacionales:

Los circuitos combinacionales son aquellos en los que sus salidas dependen única y exclusivamente de sus entradas sin importar la secuencia en que se introdujeron dichas señales y su respuesta es instantánea, esto es, es un circuito independiente del tiempo. Como no interviene el tiempo son los arreglos lógicos más sencillos. El uso de ecuaciones y métodos de simplificación suelen utilizarse solo en casos de la existencia de un número de variables reducidas.

EJEMPLO 1:

Se tienen dos botones y se tienen cuatro focos, los cuales deberán prender en las siguientes condiciones:

1.- Si los dos botones se encuentran abiertos se enciende el foco número uno.
2.- Si se activa el botón uno se enciende el foco número dos y se apaga el foco uno.
3.- Si se activa el botón dos se apaga el foco dos y se enciende el foco número tres.
4.- Si se activan los dos botones se apaga el foco número tres y se prende el foco número cuatro.

Como se puede observar nunca podrán encender más de un foco y siempre deberá estar encendido al menos uno. Para la solución del presente problema haga uso de los relevadores.

Como podemos observar en este ejercicio se tienen solo dos variables y los tiempos son arbitrarios por lo que es muy sencillo desarrollar las ecuaciones correspondientes. Ver figura 1.4.4

Tomando en cuenta que los botones son con un contacto abierto utilizamos relevadores.

$$F1 = \overline{B1} \cdot \overline{B2} = \overline{R1}\,\overline{R2}$$
$$F2 = B1 \cdot \overline{B2} = R1\,\overline{R2}$$
$$F3 = \overline{B1} \cdot B2 = \overline{R1}\,R2$$
$$F4 = B1 \cdot B2 = R1\,R2$$

Figura 1.4.4

Para leer estas ecuaciones sería de la siguiente manera:

1º.- Para que se energice el F1 se requiere que B1 y B2 se encuentren ausentes lo cual se representa con la negación indicada cor una línea en la parte superior.
2º .- Para que se energice el F2 se requiere que B1 se haga presente y que B2 este ausente.
3º .- Para que se energice el F3 se requiere que B1 este ausente y B2 se haga presente.
4º .-Para que se energice el F4 se requiere que B1 y B2 se hagan presentes.

SOLUCIÓN:

Una vez que se tienen las ecuaciones se procede a desarrollar el diagrama en formato de diagrama de escalera tal como se indica en la figura 1.4.5.

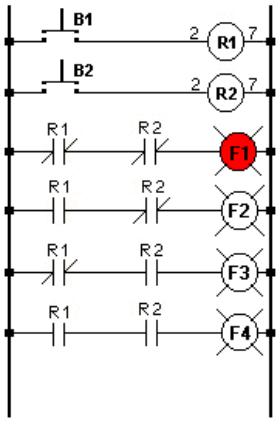


Figura 1.4.5

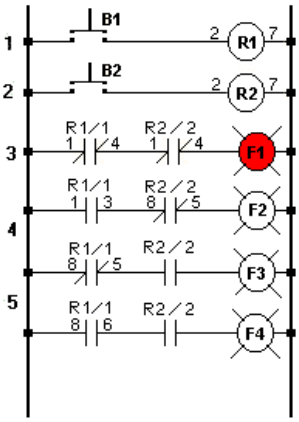


Figura 1.4.6

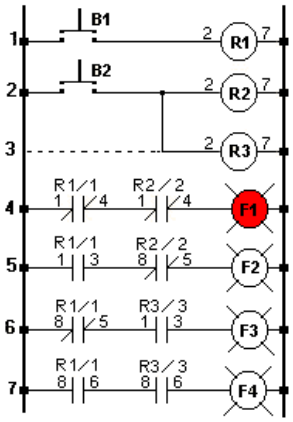


Figura 1.4.7

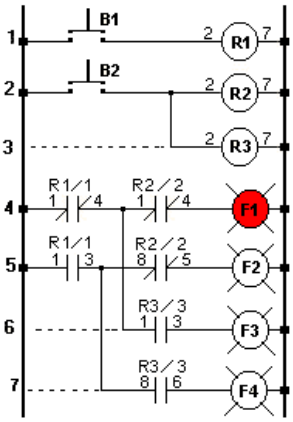


Figura 1.4.8

Si este circuito lo queremos llevar a la práctica tenemos que seleccionar los relevadores adecuados: Para este caso observamos de acuerdo a las ecuaciones que se requieren dos contactos abiertos y dos contactos cerrados para cada relevador, por lo que un relevador muy usado en la industria y que se encuentra en la mayoría de las firmas comerciales tal y como aparece en la figura 1.4.9

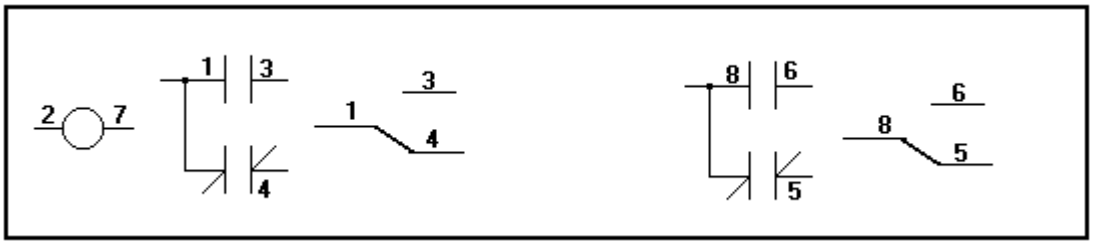


Figura 1.4.9

Una vez seleccionado el relevador procedemos a colocar los números de los contactos tal como se mostró en la figura 1.4.6 aquí, podemos observar que aún cuando el relevador seleccionado tiene dos juegos de contactos con dos abiertos y dos cerrados no podemos realizar el cableado ya que las patitas 1 y 4 y 8 y 5 del relevador R2 (ver figura 1.4.6 que no tienen números en sus patitas) no son comunes, por lo tanto tendremos que utilizar otro relevador en paralelo con R2 que le llamaremos R3 para que nos proporcione más contactos tal como lo observamos en la fig. 1.4.7

En la figura 1.4.7 observamos como los contactos de R1 en las patitas (1 y 4 con 8 y 5) son comunes igualmente las patitas (1 y 3 y 8 y 6) son comunes, considerando esto, lo podemos simplificar, eliminando contactos tal como aparece en la figura 1.4.8.

Al eliminar contactos de R1 se hacen comunes los contactos de R2 con los contactos de R3 con lo cual podemos sustituir de nuevo el relevador R3 por R2 tal como aparece en la figura 1.4.10

Una vez que se han realizado las simplificaciones correspondientes se procede a la reglamentación del diagrama siguiendo las siguientes bases:

- Numeración de los eslabones de la escalera en la parte izquierda del diagrama de escalera.

- Después se procede a indicar en la parte de la derecha de los relevadores la cantidad de contactos que se están utilizando por cada relevador y si estos son normalmente abiertos o normalmente cerrados indicando los cerrados con una negación en la parte superior del indicador del eslabón además de indicar en cual eslabón se encuentran dichos contactos.

- Cada contacto del diagrama, además del indicador de a que bobina pertenece, deberá indicarse en que eslabón se encuentra la bobina que lo controla y eso se representa poni éndole una diagonal y el numero del eslabón donde se encuentra dicha bobina.

- Se procede a colocar los números indicadores del cableado tomando como base que a cada transición de un elemento de control deberá incrementarse en la misma propor ción el número del cable, y éste número deberá estar dentro de un circulo.

- Para cada contacto se le deberá indicar los números de patitas del relevador para su correcta identificación.

- En la parte superior se indicara el voltaje de alimentación de los rieles.

- Por último en los focos indicadores se desplegará una leyenda indicando cual es su función.

- Finalmente en la figura 1.4.11 se presenta un dibujo con toda la normatividad de los diagrama de escalera.

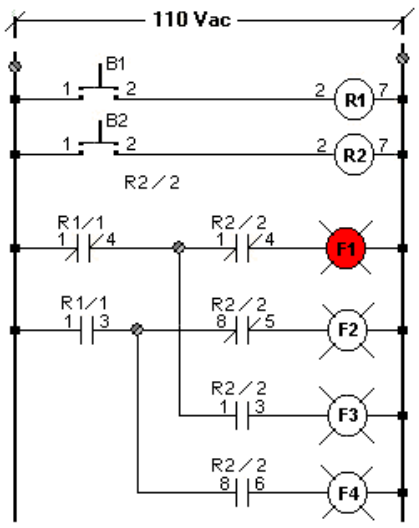


Figura 1.4.10

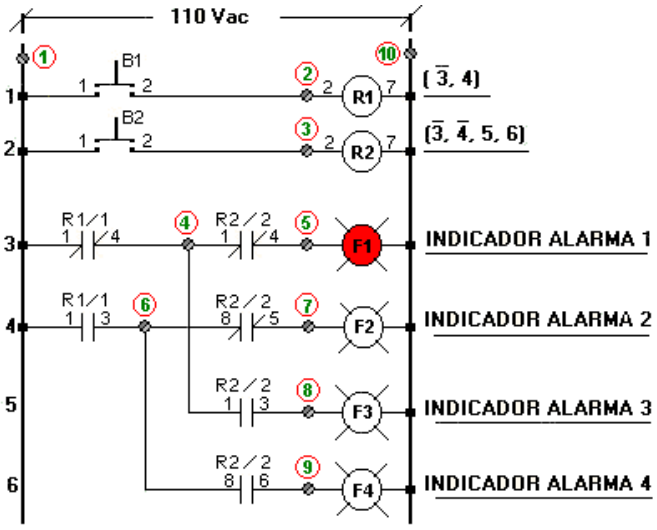


Figura 1.4.11

EJEMPLO 2:

El siguiente problema es similar al ejemplo número 1 solo que ahora tendremos tres botones y 8 focos, los cuales trabajaran bajo las siguientes condiciones.

1.- Si los tres botones se encuentran ausentes se prendera el foco 1.
2.- Si se oprime el botón 1 se apaga el foco uno y prende el foco 2.
3.- Si se oprime el botón 2 se apagará el foco 2 y prendera el foco 3.
4.- Si se oprime el botón 3 se apagará el foco 3 y prendera el foco 4.
5.- Si se oprimen los botones 1 y 2 se apagará el foco 4 y prendera el foco 5.
6.- Si se oprimen los botones 1 y 3 se apagará el foco 5 y prendera el foco 6.
7.- Si se oprimen los botones 2 y 3 se apagará el foco 6 y prendera el foco 7.
8.- Si se oprimen los botones 1 y 2 y 3 se apagara el foco 7 y prendera el foco 8.

Como se puede observar nunca podrán encender más de un foco y siempre deberá estar encendido al menos uno. Para la solución del problema haga uso de los relevadores.

Como podemos observar en este ejercicio se tienen sólo tres variables por lo que es muy sencillo desarrollar las ecuaciones correspondientes. Tomando en cuenta que los botones son con un contacto abierto utilizamos relevadores.

NOTA:

Sí no se sigue la secuencia indicada, el resultado es el mismo ya que pertenece a la categoría de los combinacionales.

SOLUCIÓN:

Como sólo tenemos tres variables y su aplicación pertenece a los circuitos combinacionales, se pueden obtener fácilmente las ecuaciones correspondientes.

$$F1 = \overline{B1}\cdot\overline{B2}\cdot\overline{B3} = \overline{B1}\ \overline{B2}\ \overline{B3} = \overline{R1}\ \overline{R2}\ \overline{R3}$$
$$F2 = B1\cdot\overline{B2}\cdot\overline{B3} = B1\ \overline{B2}\ \overline{B3} = R1\ \overline{R2}\ \overline{R3}$$
$$F3 = \overline{B1}\cdot B2\cdot\overline{B3} = \overline{B1}\ B2\ \overline{B3} = \overline{R1}\ R2\ \overline{R3}$$
$$F4 = \overline{B1}\cdot\overline{B2}\cdot B3 = \overline{B1}\ \overline{B2}\ B3 = \overline{R1}\ \overline{R2}\ R3$$
$$F5 = B1\cdot B2\cdot\overline{B3} = B1\ B2\ \overline{B3} = R1\ R2\ \overline{R3}$$
$$F6 = B1\cdot\overline{B2}\cdot B3 = B1\ \overline{B2}\ B3 = R1\ \overline{R2}\ R3$$
$$F7 = \overline{B1}\cdot B2\cdot B3 = \overline{B1}\ B2\ B3 = \overline{R1}\ R2\ R3$$
$$F8 = B1\cdot B2\cdot B3 = B1\ B2\ B3 = R1\ R2\ R3$$

Figura 1.4.12

Una vez que se han obtenido las ecuaciones correspondiente para cada objetivo (en este caso los focos figura 1.4.12) se procede a representarlos en diagrama de escalera como lo muestra la figura 1.4.13

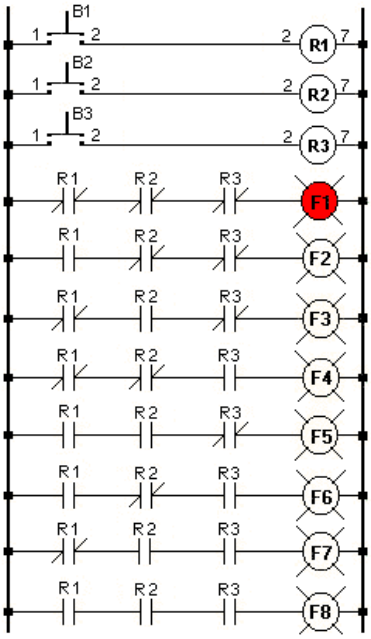


Figura 1.4.13

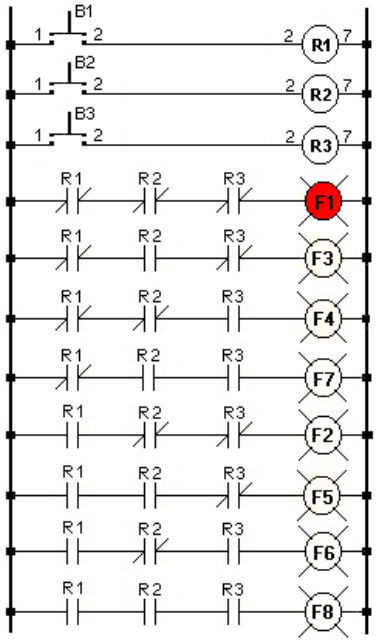


Figura 1.4.14

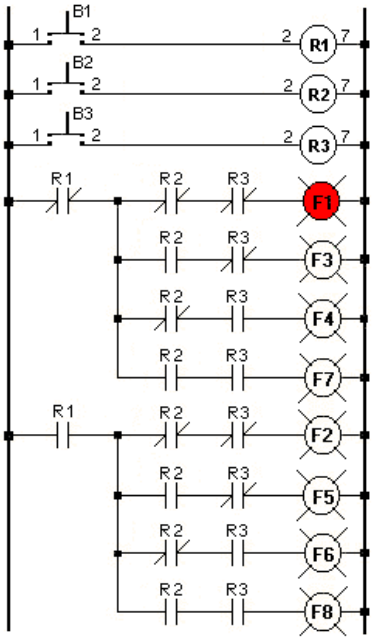


Figura 1.4.15

En la figura 1.4.14 se ha realizado un reacomodo de los contactos de R1 de tal manera que queden alineados los contactos abiertos y cerrados de R1, para que podamos utilizar los comunes.

En la figura 1.4.15 se han eliminado algunos contactos de R1 tanto abiertos como cerrados aprovechando que son comunes.

En la figura 1.4.16 se han reacomodado los contactos, primero los cerrados de R2 y luego los abiertos de tal manera que puedan ser comunes.

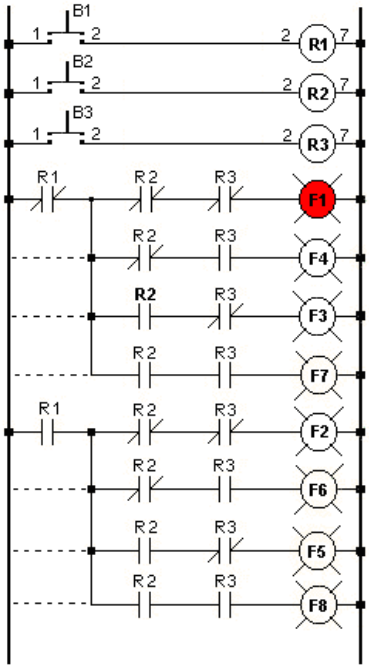


Figura 1.4.16

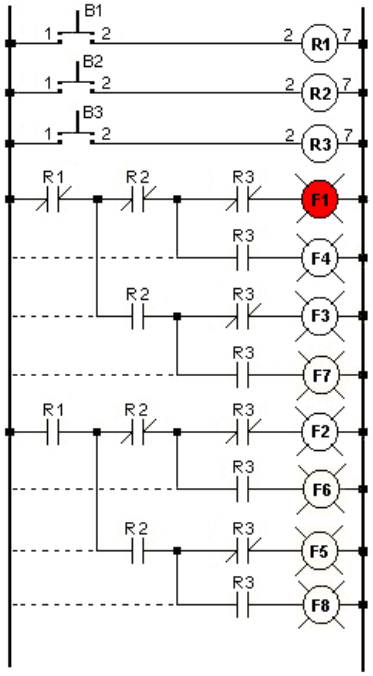


Figura 1.4.17

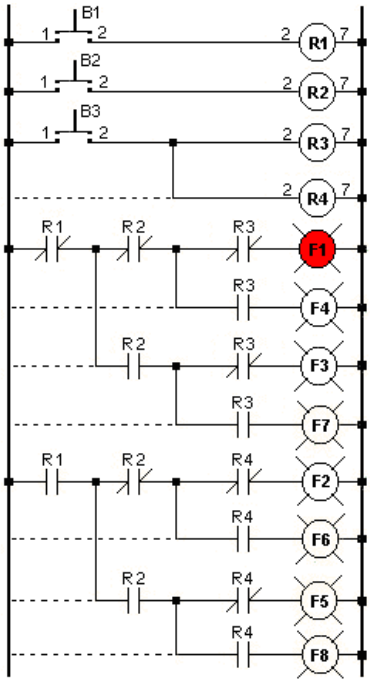


Figura 1.4.18

En la figura 1.4.17 se ha procedido a eliminar los contactos de R2 tanto los cerrados como los abiertos aprovechando que son comunes. Observe en la fig. 1.4.18 como se tienen cuatro juegos de contactos de R3 con cuatro comunes cuatro abiertos y cuatro cerrados.

En la figura 1.4.18 se ha colocado un nuevo relevador R4 en paralelo con R3 ya que los contactos de R3 no son suficientes para realizar todo el cableado, ahora con los contactos de R4 en paralelo con R3 podemos decir que tenemos el doble de contactos con lo cual podemos resolver el problema.

En la figura 1.4.19 se procedió primero a enumerar los eslabones de la escalera en la parte de la izquierda, luego en la parte de la derecha de cada relevador se indican los contactos que se están utilizando de cada relevador indicando en que eslabón se encuentran y si estos son abiertos o cerrados, si son cerrados se le coloca la negación (a esto se le llama referencia cruzada ó cross reference). Después en cada contacto de relevador se le puso una diagonal indicando donde se encuentra la bobina que lo controla (ejemplo R1/1 que indica que la bobina de R1 se encuentra en la línea 1), también en la parte de la derecha se le pone un enunciado a cada foco indicando cual es su función , en este caso se les llamo alarmas pero pueden tener el significado adecuado para cada automatización y por ultimo se procedió a indicar el numerado de los cables los cuales fueron encerrados en círculos rojos para mayor claridad, de esta manera tenemos un diagrama de escalera con toda la normatividad de los mismos.

NOTA:

Para la numeración del cableado, no se sigue ningún orden, solo se sigue la norma de que por cada elemento de control cambie su numeración. (Ver figura 1.4.19)

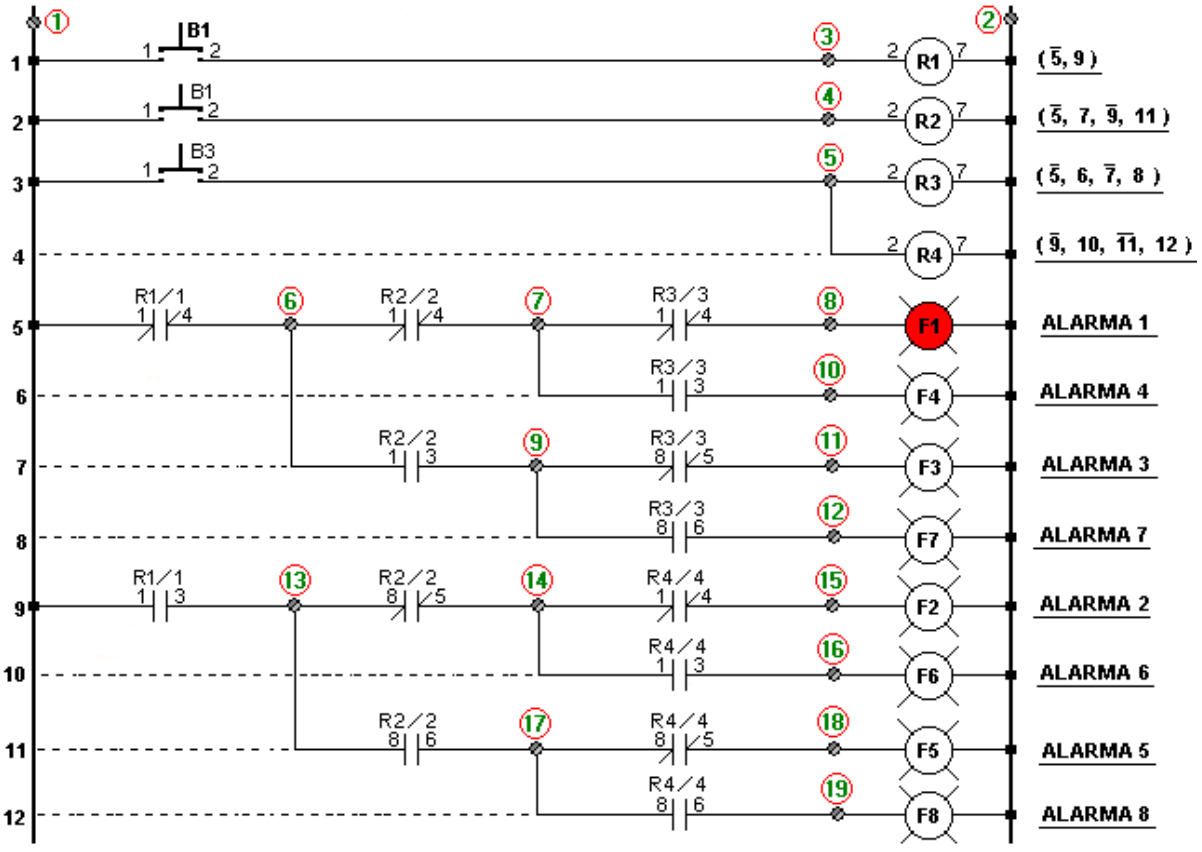


Figura 1.4.19

EJEMPLO 3:

Se desea automatizar una sala de cine, para ello se realizó el estudio de la generación de calor por persona obteniendo los siguientes puntos; cuando la sala esta al 33%, 66%, y 99% de su capacidad, así como el consumo de aire de acuerdo al área instalada y se llego a las siguientes conclusiones:

1.- Si el cine se encuentra con una concurrencia de menos del 33% de su capacidad, la temperatura deberá estar aproximadamente a 20º C, lo cual es una temperatura agradable y no hay necesidad de ningún control.

2.- Si la asistencia es arriba del 33%, la temperatura llega a 22º C y deberá entrar a trabajar un sistema de aire lavado con un motor de 5 H.P, que le vamos a llamar "motor pequeño", con este motor deberá bajar la temperatura de 22 º C y con ello parar el motor.

3.- Si la asistencia a la sala de cine es del 66 % la temperatura sube hasta 24 º C lo cual significa que la maquina pequeña no pudo con la carga térmica y deberá salir y entrar una maquina más grande de 10 H. P, le vamos a llamar "motor grande", la cual deberá bajar la temperatura.

4.- Si a la sala de cine entran el 99 % la temperatura sube hasta 26 º C esto significa que la maquina grande tampoco pudo con la carga térmica y deberá entrar a ayudar la maquina pequeña.

NOTA: Como condición se pide que para que trabajen las maquinas, deberá estar circulando agua en las maquinas como sistema de enfriamiento para evitar el sobrecalentamiento.

SOLUCIÓN: Para la solución del presente problema primero debemos identificar las variables a controlar:

A = Detector de flujo de agua en el sistema.
T1= Termostato para detectar la temperatura de 22 º C.
T2= Termostato para detectar la temperatura de 24 º C.
T3= Termostato para detectar la temperatura de 26 º C

Ahora podemos proceder a realizar las correspondientes ecuaciones tomando en cuenta que son pocas las variables.

$M1 = (AT1\overline{T2} + AT3)$ Si factorizamos $M1 = A (T1 \overline{T2} + T3)$
$M2 = AT2$

Es importante saber leer las ecuaciones considerando no las literales, si no, las variables lógicas que se están utilizando; una manera de leer estas ecuaciones sería:

Para que se energice la maquina 1 que es una maquina pequeña se requiere primero que haya agua circulando en el sistema, además que se haga presente la temperatura uno, esto es que la temperatura llegue a 22 º C y que no se haga presente la temperatura T2 esto es, que no alcance los 24 º C; También podrá entrar la maquina uno si hay agua circulando en el sistema y la temperatura T3 se hace presente o sea que llegue hasta 26 º C.

Para que entre la maquina dos se requiere que haya agua circulando en el sistema y que se haga presente la temperatura T2 esto es, que llegue hasta 24 º C.

Primero vamos a mostrar en Figura 1.4.20 la simbología para los sensores utilizados;

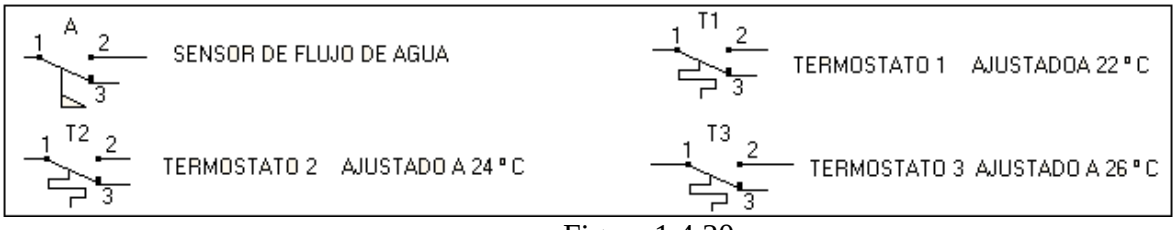


Figura 1.4.20

Una vez que se dio la lectura de las ecuaciones se procede a realizar el circuito directo de la ecuación como se ve en la figura 1.4.21

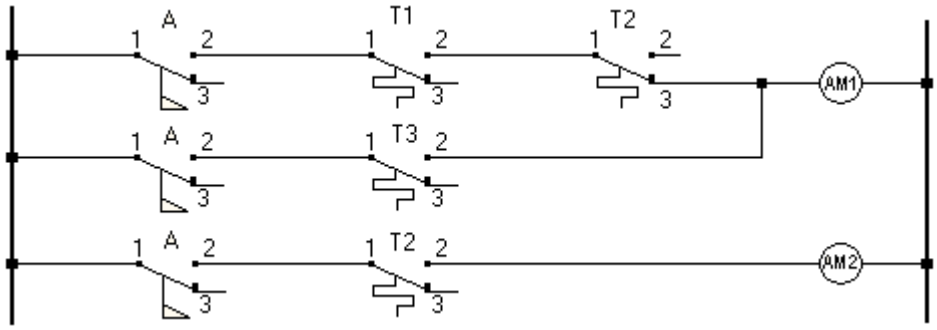


Figura 1.4.21

En la figura 1.4.22 aprovechando que el sensor de flujo tiene un común, se procedió a su arreglo indicado.

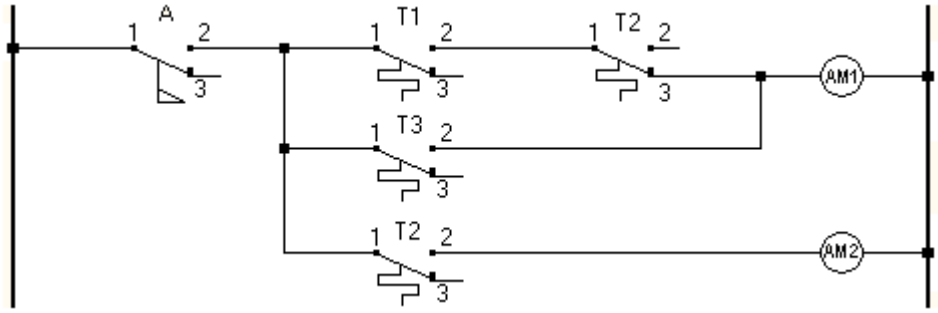


Figura 1.4.22

En la figura 1.4.23 se aplicó la propiedad distributiva del álgebra de Boole para hacer común al termostato T2.

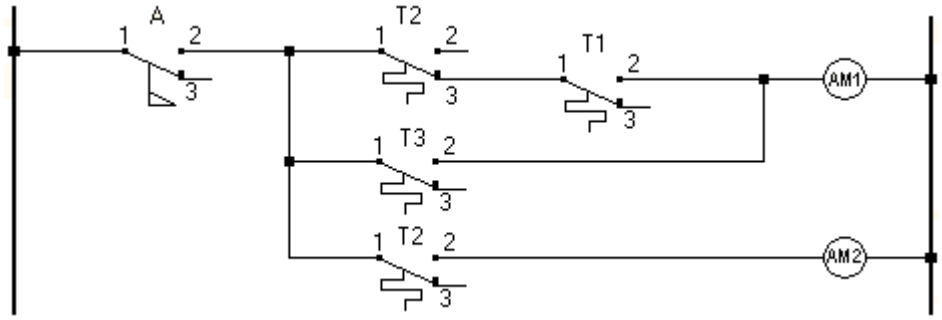


Figura 1.4.24

En la figura 1.4.24 se le dio un arreglo para su mejor interpretación.

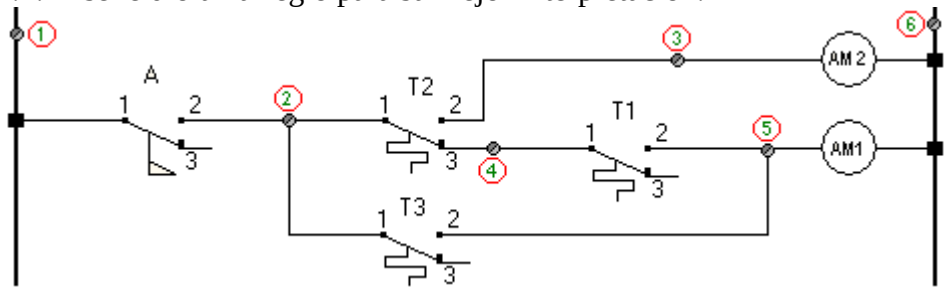


Figura 1.4.24

Automatización con circuitos secuenciales

Introducción:

En este capítulo vamos a dar las bases para el desarrollo de circuitos secuenciales con control electromecánico. Además nos servirá como punto de partida para nuestro objetivo principal que es el control programable.

Los circuitos secuenciales son aquellos en los que sus salidas dependen, además de sus señales de entrada, de la secuencia en que se introducen dichas señales de los sistemas de retroalimentación y de los sistemas de retardo existentes en el mismo sistema, esto es, es un circuito dependiente del tiempo

Los circuitos secuenciales se dividen en tres partes y se les llama de la siguiente forma:

1.- Circuitos BI-estable (Donde interviene la memoria): Son aquellos en los que como su nombre lo indica tienen dos estados estables, ejemplo: la memoria o flip-flop.

2.- Circuitos MONO-estables (Donde interviene el tiempo): Son aquellos en los que como su nombre lo indica tienen un estado estable y el otro casi estable, ejemplo el temporizador, relevador de tiempo ó timer.

3.- Circuitos AS-tables (Donde intervienen los osciladores). Son aquellos en los que como su nombre lo indica no tienen ningún estado estable, ejemplo el oscilador.

1.5.1 Circuitos BI-estables (Donde interviene la memoria)

La base para los circuitos secuenciales, es la función memoria, es decir, un arreglo capaz de "recordar" si el sistema ha recibido una señal de control, la función requiere una salida que posea una retroalimentación, esto significa que la salida debe de influir en la entrada, para que así mantenga la memoria de su estado.

Aquí podemos decir que la ecuación de la memoria es la ecuación más importante de los sistemas de control, ya sean estos sistemas eléctricos, electrónicos, neumáticos, hidráulicos, etc.

Existen dos maneras de llegar a la ecuación de la memoria, una es partiendo de la experiencia realizar el circuito y del circuito sacar la ecuación y la otra es de acuerdo al planteamiento: deducir la ecuación y de la ecuación realizar el circuito.

Vamos a inclinarnos por el segundo planteamiento para deducir primero la ecuación de la memoria y de esta realizar el circuito. Para ello vamos a plantear el siguiente problema;

Se tienen dos botones uno de arranque y uno de paro (ver figura 1.5.1) y deseamos arrancar un motor con un impulso del botón de arranque y que este nos recuerde o nos memorice que se ha activado y con el botón de paro queremos parar el motor, esto es tumbar la memoria.

Figura 1.5.1

Para resolver este planteamiento se requiere que el sistema tenga retroalimentación, por lo tanto se necesita la ayuda de un relevador para realizar dicha tarea y entonces la ecuación nos quedaría de la siguiente manera;

$$M1 = (Ba + m1)\,\overline{(Bp)}.$$ Ecuación Nº 1

Esta es la ecuación más importante de los sistemas de control.

Una manera de leer esta ecuación es de la siguiente manera;

Para que se haga presente M1 se requiere que se haga presente (Ba ó m1) y que no se haga presente Bp; luego como m1 pertenece a M1 sólo nos queda un camino que es el Ba y una vez que se hace presente Ba y como m1 esta en paralelo, se puede hacer ausente Ba con lo cual se sostiene M1. Es por ello que a m1 se le conoce también como contacto de sostén o contacto de memoria, y al hacerse presente Bp no se cumplen las condiciones y se pierde la memoria.

Una vez que se tiene la ecuación se procede a representarlo en un circuito de diagrama de escalera, el cual queda como se ve en la Figura 1.5.2

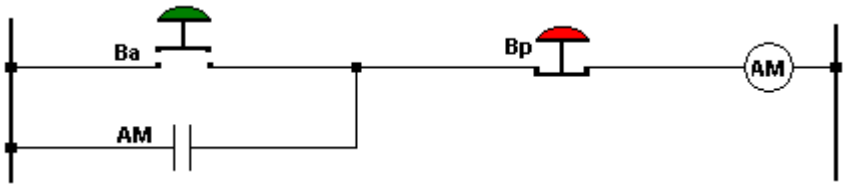


Figura 1.5.2

En electrónica a este circuito se le llama Flip flop RS y su equivalente a compuertas se muestra en la figura 1.5.3

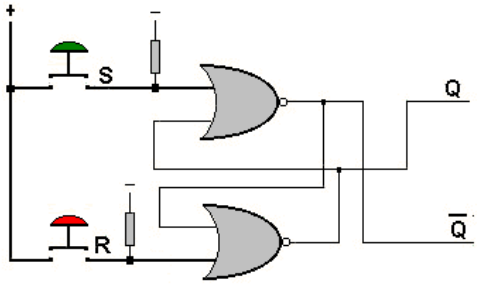


Figura 1.5.3

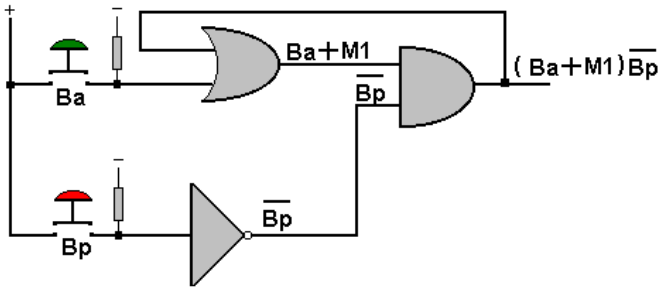


Figura 1.5.4

Este circuito es equivalente al circuito eléctrico y su demostración se realiza paso a paso en las figuras 1.5.4, 1.5.5, 1.5.6, 1.5.7 y 1.5.8.

En la Figura 1.5.4 podemos observar como directamente de la ecuación de la memoria se ha realizado el circuito de control con compuertas, en este circuito podemos observar que se requieren

tres circuitos integrados, uno de compuertas AND, uno de compuertas OR y uno más de compuertas NOT. Por supuesto que si esto lo llevamos a la práctica sale además de caro más laborioso que el circuito original que es con solo dos compuertas, por lo que se procede a convertirlo a compuertas NOR.

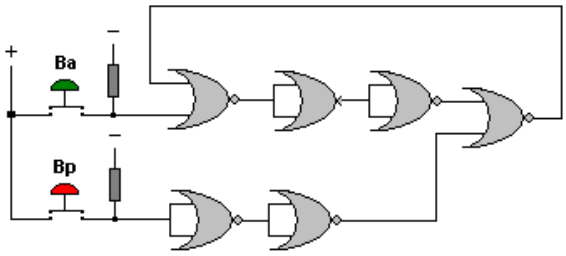


Figura 1.5.5

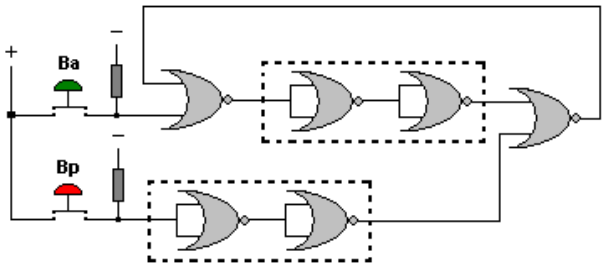


Figura 1.5.6

En la figura 1.5.5 podemos observar su circuito equivalente con compuertas NOR y ahora vemos que se requieren dos circuitos integrados, además de que son de un sólo tipo y por ser de compuertas NOR resulta más económico pero aún más caro que el circuito original.

En la figura 1.5.6 observamos como se aprovecha que con este tipo de compuertas se tienen estados redundantes, estos son los que están dentro de las líneas punteadas y procedemos a eliminarlos.

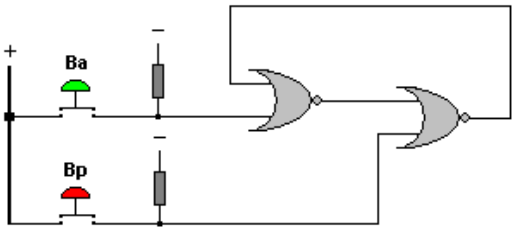


Figura 1.5.7

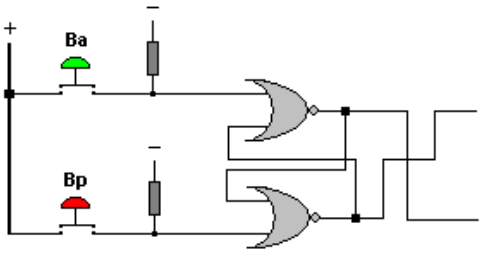


Figura 1.5.8

En la figura 1.5.7 podemos observar como se eliminaron los estados redundantes.

En la figura 1.5.8 se reacomodaron las compuertas con lo cual queda demostrado que la ecuación de la memoria corresponde al circuito Flip Flop y su equivalente a circuitos eléctricos en arreglo de diagramas de escalera.

Otra manera diferente de obtener el Flip Flop de la Fig. 1.5.8 a partir de la ecuación N° 1, es de la siguiente manera:

$M1 = (Ba + m1)\ \overline{(Bp)}$ si le aplicamos la doble negación a la ecuación no nos altera la ecuación original.

Se aplica la primera negación:

$\overline{\overline{M1 = (Ba + m1)\,(\overline{Bp})}}$ Procedemos primero a aplicar la primera negación.

$M1 = \overline{(\overline{Ba} \cdot \overline{m1}) + Bp}$ Por la ley de Morgan separamos los términos

Sustituimos $(\overline{Ba} \cdot \overline{m1})$ por X

Nos queda:

$M1 = \overline{x + Bp}$ Separamos

$$M1 = \overline{x} \cdot \overline{Bp}$$

Como podemos observar ésta es una compuerta NOR, con lo cual tenemos el Flip Flop de la Fig 1.5.8.

1.5.2 Ejemplos de circuitos bi-estables

EJEMPLO 1:

Se desea automatizar el suministro de agua de un aljibe hacia un depósito de agua por medio de una bomba. Dicha bomba deberá trabajar cuando el nivel del tanque de almacenamiento esté en el nivel bajo A y parar cuando el nivel este alto B. (ver Fig. 1.5.9)

SOLUCIÓN:

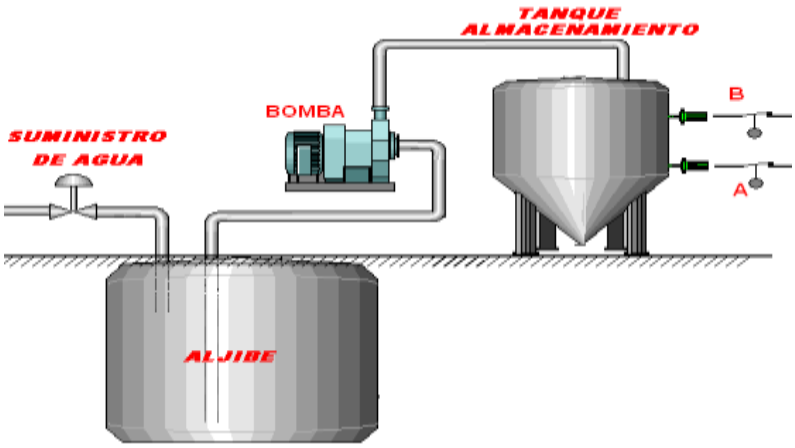


Figura 1.5.9

Para resolver este problema vamos a utilizar dos flotadores, de tal manera que cuando el nivel del tanque de almacenamiento este bajo los dos flotadores se encuentran cerrados, estando vacío el tanque y los contactos de los flotadores cerrados se deberá energizar la bomba, luego al estar energizada la bomba y suministrando agua empieza a subir el nivel y llega al punto A abriendo el flotador, sigue subiendo el nivel y abre el flotador B con lo cual deberá parar la bomba y cuando baje de nuevo el nivel deberá arrancar de nuevo.

Aquí podemos observar que este es un sistema de arranque y paro y que la ecuación de la memoria resuelve el problema. En este caso A es el arranque y B es el paro. (Ver figura 1.5.10).

$AB = (A + ab)\overline{B}$

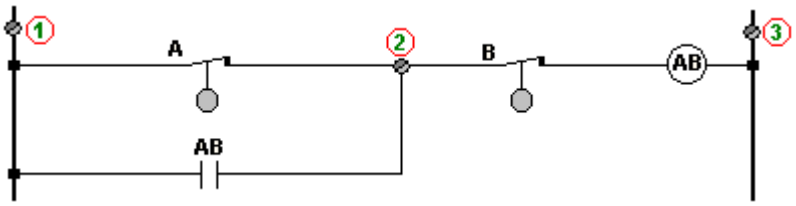


Figura 1.5.10

Una vez resuelto el problema, se nos pide que se controle también el nivel del aljibe ya que en ocasiones se queda este sin agua y la bomba, por carecer de sistema de enfriamiento que proporciona el agua, se quema: por lo tanto se requiere controlar que si el nivel del agua en el aljibe baja del punto C no deberá trabajar la bomba.

SOLUCIÓN:

Una solución muy sencilla es ponerle un flotador en el punto C al aljibe de tal manera que si baja de ese punto se pare la bomba. Observe la figura 1.5.11

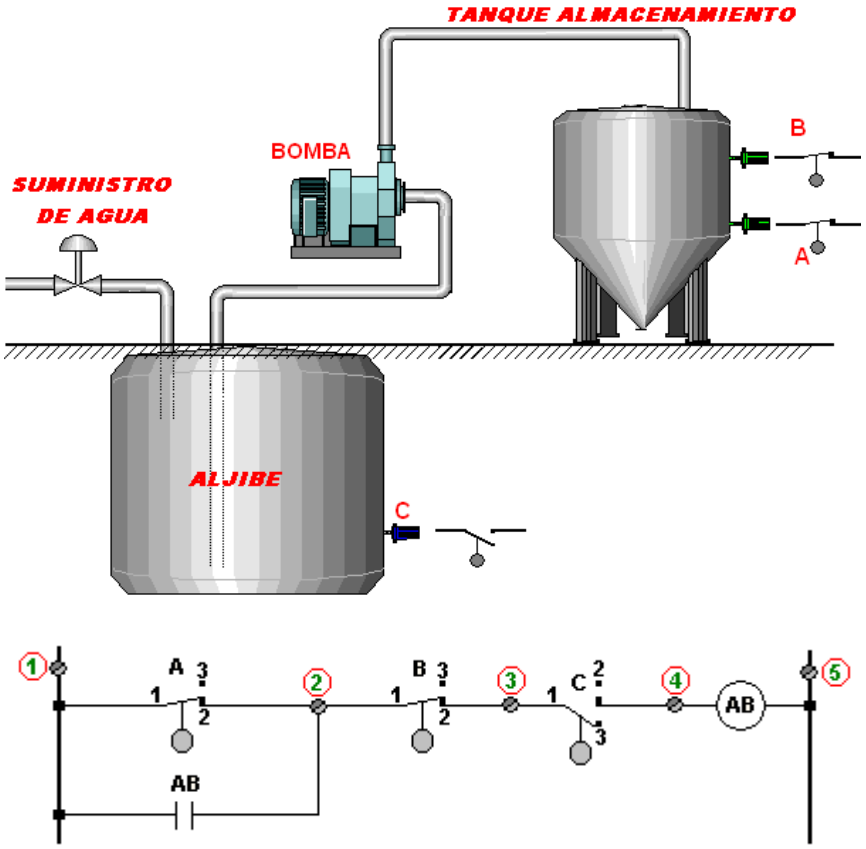


Figura 1.5.11

Una vez que se resolvió lo indicado se presento el siguiente problema; Al bajar el nivel de C en el aljibe para la bomba, pero al subir un poco el nivel se hace presente la variable C y la bomba

arranca de nuevo, ahora, considerando que el flujo que suministra agua al aljibe es menor al que succiona la bomba, está, arrancará y parará constantemente con el correspondiente gasto de energía eléctrica, por lo que ahora se requiere que se ponga un sensor de nivel D de tal manera que cuando baje el nivel hasta C no arranque la bomba hasta que llegue al nivel D, esto es para que nos garantice la cantidad suficiente de agua para llenar el tinaco. Ver figura 1.5.12

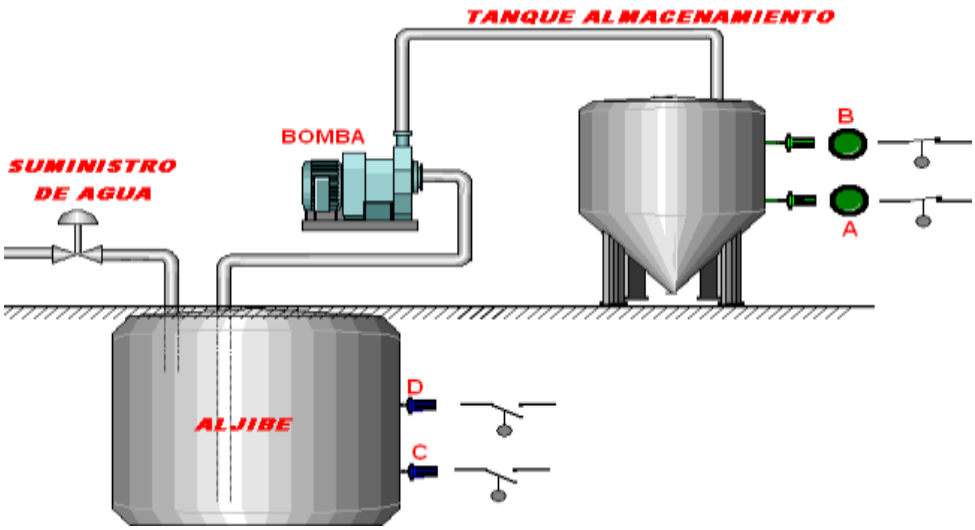


Figura 1.5.12

SOLUCIÓN:

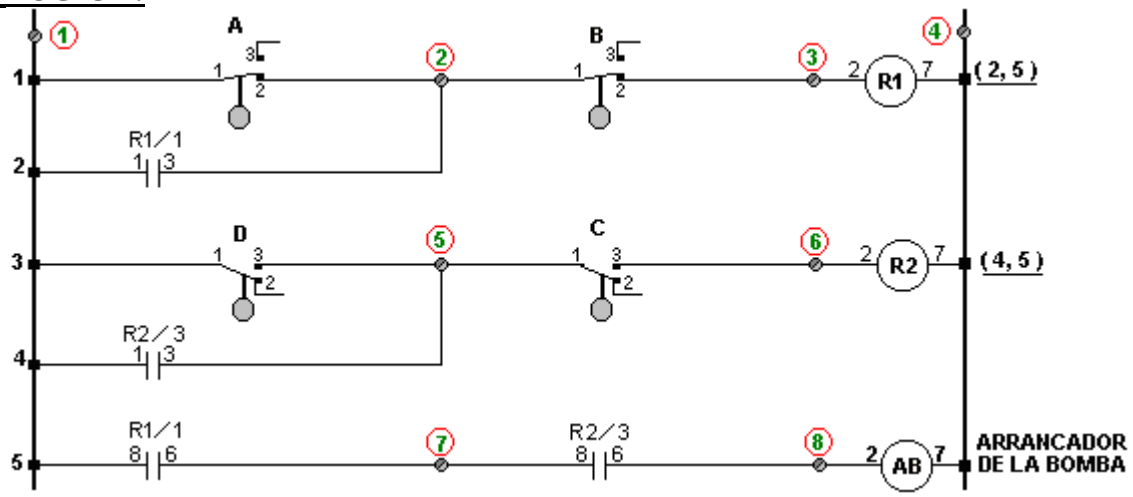


Figura 1.5.13

Considérese el problema anterior sólo que ahora queremos suministrar agua a un edificio de siete pisos, por lo que el tinaco se cambia por un deposito cerrado sometido a presión, el cual por norma deberá tener un manómetro para estar viendo la presión, un termómetro para estar viendo la temperatura, una válvula de seguridad para que en caso de falla de los sistemas eléctricos pueda abrir y desalojar la presión y un presóstato para estar controlando la presión, se requiere que el sistema suministre agua a presión a todos los pisos del edificio, bajo las siguientes condiciones;

Tomando en cuenta los controles anteriores ahora se desea que cuando el nivel del tanque de presión se encuentre en el punto B se cheque la presión en el tanque y si esta presión se encuentra por debajo de los 7 Kg/cm2 que se energice una electro válvula para inyectarle aire a presión al deposito y que cuando la presión sea igual o mayor a los 7 Kg/cms2 se des-energice dicha electro

válvula. Con un estudio previo se comprobó que con una presión de 7 Kg/cms2 se inyectará agua a todos los pisos del condominio.

En la figura 1.5.14 se muestra el sistema completo y en la figura 1.5.15 su control correspondiente.

La electro válvula utilizada, es una del tipo todo o nada, técnicamente conocida como 2 vías dos posiciones, normalmente cerrada, accionamiento eléctrico y retorno por muelle.

Este tipo de válvulas, así como su correcto funcionamiento se detallará en el capitulo 3.

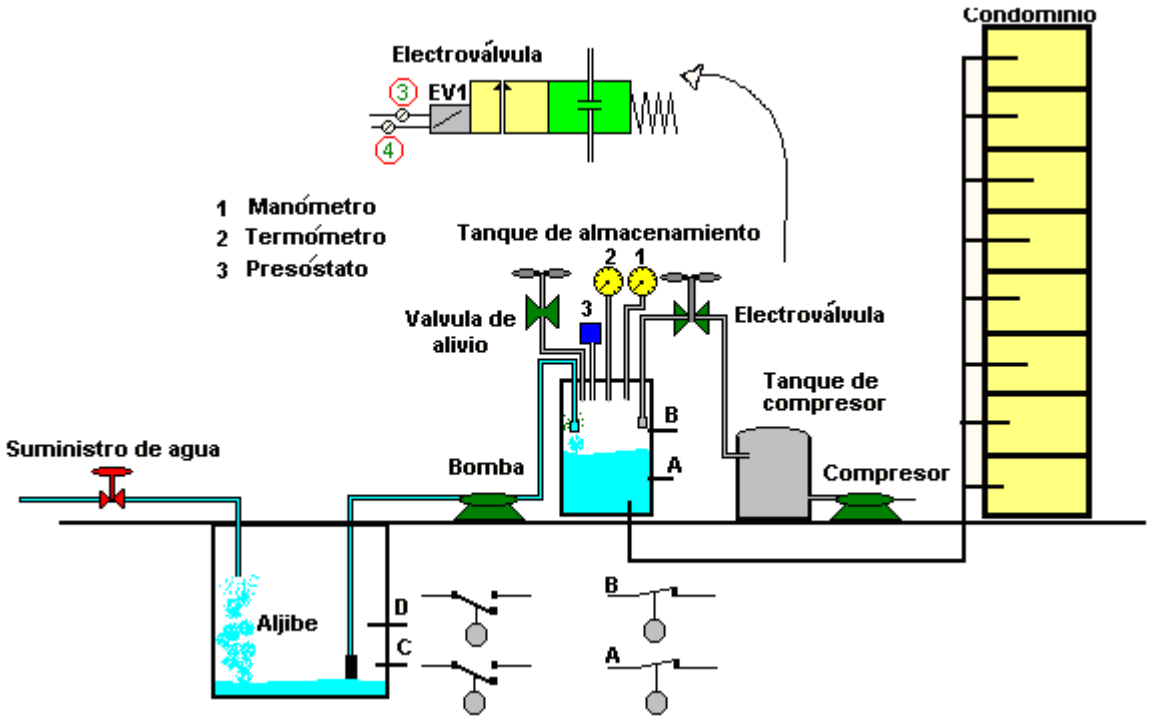


Figura 1.5.14

Obsérvese cómo en la figura 1.5.16 se reacomodo el flotador de B para que se pueda utilizar con el Sw. de presión P del presóstato.

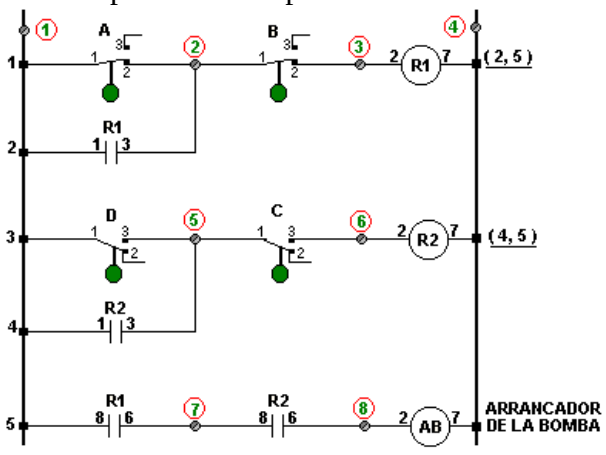


Figura 1.5.15

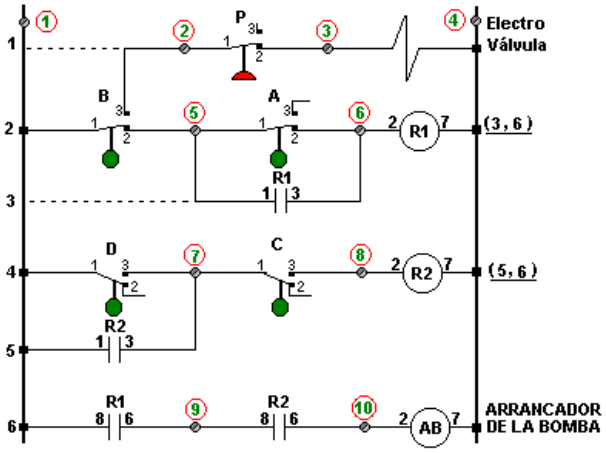


Figura 1.5.16

Por último se desea que el sistema cuente con un control de arranque y paro general, Ver figura 1.5.17

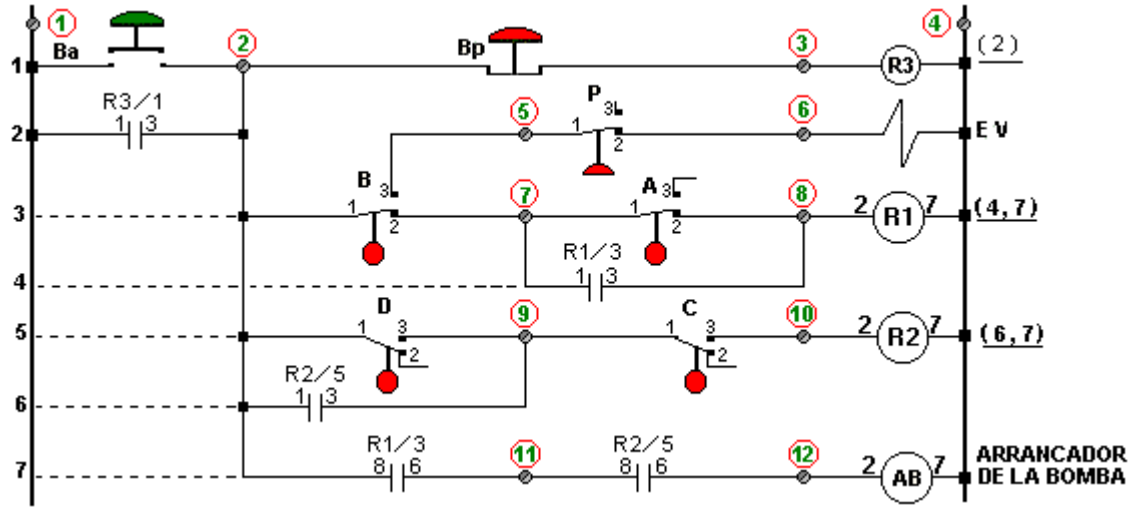


Figura 1.5.17

EJEMPLO 2:

El cuadro de la figura 1.5.18 representa una fábrica en la cual se tiene un sistema de aire acondicionado instalado en el taller de mantenimiento, los técnicos de aire acondicionado han realizado la instalación de ductos y de máquinas, ahora nos piden que realicemos el control eléctrico, el cual deberá contar con una estación de botones en cada oficina además de una estación de botones en el taller de mantenimiento para realizar las pruebas de arranque y paro cuando se les da mantenimiento, además nos piden que se tenga un interruptor general para que cuando el personal de mantenimiento este dando servicio al equipo queden fuera todas las estaciones de botones. La solución se presenta en la figura 1.5.19

Figura 1.5.18

NOTA: El sistema cuenta con una sola máquina de aire acondicionado. **SOLUCIÓN:**

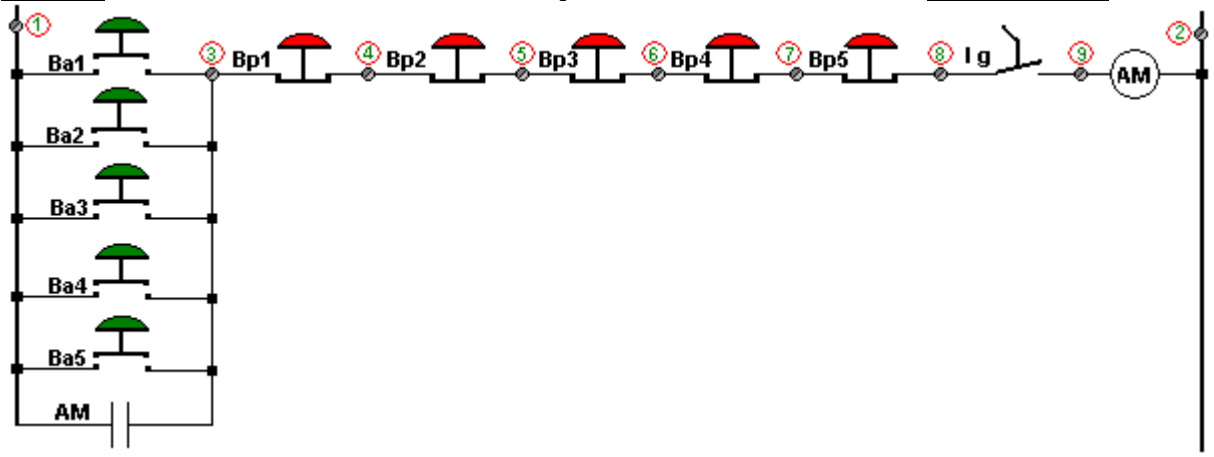


Figura 1.5.19

EJEMPLO 3:

Siguiendo la misma metodología realice el siguiente circuito.
Al oprimir el B1 se energice el Foco 1 y al soltarlo se quede prendido.
Luego al oprimir de nuevo el botón 1 se apague y al soltarlo que todo quede en condiciones para repetir el ciclo. Como podemos observar, éste es un Flip Flop tipo "T"

SOLUCIÓN:

Primeramente vamos a desarrollar un circuito como clock con R1 ya que el botón tiene un solo contacto y el relevador R1 tiene una serie de contactos abiertos y cerrados.

Luego al oprimir el botón 1 y energizar R1, con un contacto de R1 realizamos una memoria que nos recuerde que ya se activo el botón 1, además con esta memoria energizamos el Foco 1. Luego al soltar el botón 1 y como necesitamos que B1 se vuelva a activar, se requiere una memoria que nos recuerde que ya se desactivo el botón 1, por lo tanto realizamos una memoria que nos recuerde que ya se desactivo el botón 1 con R3.

Finalmente cuando activamos de nuevo el Botón 1 junto con la memoria que nos recuerda que ya lo habíamos soltado energizamos R4 y la memorizamos y con esta memoria tumbamos la señal del foco 1 y al soltar de nuevo el botón 1 nos energiza un nuevo relevador que nos desenergice todo el sistema para quedar en condiciones de repetir el ciclo.

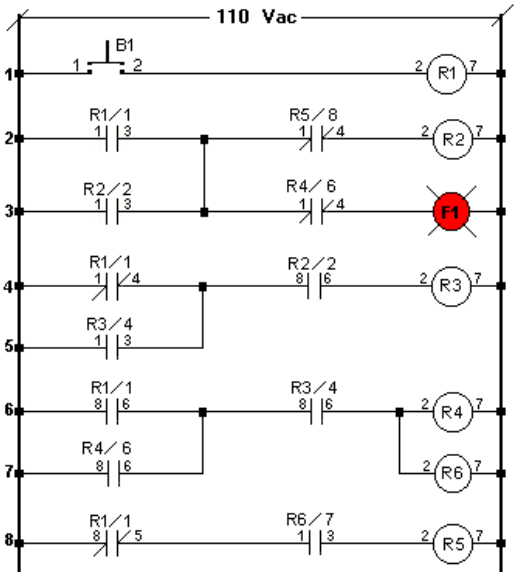


Figura 1.5.20

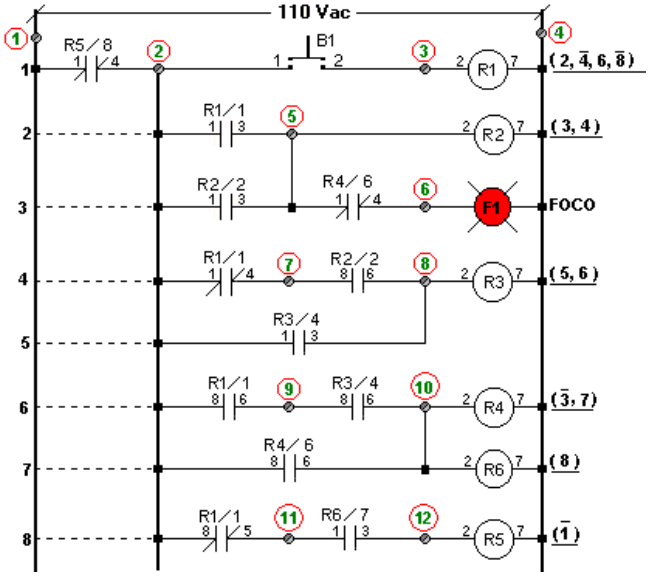


Figura 1.5.21

Explicación del ejercicio de la figura 1.5.20. En la línea 1, al oprimir el botón B1, se energiza el relevador R1, este hace un cambio de estado en sus contactos, pero debemos tomar en cuenta que primero se abren los contactos cerrados y un instante después se cierran los abiertos, considerando esto, podemos observar que en la línea 4 se abre el contacto de R1 primero y luego se cierra el contacto de R1 de la línea 2 con lo que pasa señal energizando R2, este a su vez cierra su contacto de la línea 3 con lo cual se sostiene R2 y prende el foco F1, también observe como en la línea 4 se

cierra el contacto de R2 solo que no hay ningún efecto ya que el contacto de R1 se encuentra en este instante abierto, luego al soltarlo el botón B1 se desergiza R1 y el contacto R1 abierto que estaba cerrado se abre, sin causar efecto sobre R2 ya que éste estaba sostenido. Pero el contacto de R1 de la línea 4 que es normalmente cerrado regresa a su estado original que es normalmente cerrado, con lo que se energiza R3 y este a su vez se sostiene y cierra un contacto en la línea 6 sin causar ningún efecto ya que el contacto de R1 en la línea 6 se encuentra abierto. Cuando se oprime por segunda ocasión el botón B1 se cierra de nuevo el contacto de R1 en la línea 6 y como el contacto de R3 en la misma línea ya esta cerrado se energiza R4 y R6 el cual sostiene energizada la bobina de R4 y R6, también con un contacto de R4 en la línea 3 desenergiza el F1 que es lo que se desea, finalmente al desactivar B1 se desenergiza R1 energizando R5 este a su vez desenergiza R2 el cual desenergiza R3 y este desenergiza a R4 y R6 con lo cual todo queda en condiciones de repetir el ciclo.

En la figura 1.5.20 tenemos un arreglo para la explicación anterior y en la figura 1.5.21 tenemos otro arreglo pero con un contacto permisivo de R5, ambos diagramas presentan la solución pero la alternativa 1.5.21 es más segura.

EJEMPLO 4:

Diseñe el circuito de control que realice las siguientes funciones:

1.- Que al oprimir un botón, se encienda el Foco 1.
2.- Al oprimir de nuevo el mismo botón que se encienda el foco 2 y que el primero continué energizado.
3.- Al oprimir de nuevo el botón se apaguen los dos focos y al soltarlo quede el sistema listo para reiniciar la operación.

SOLUCIÓN:

Aquí presentamos dos alternativas de solución:

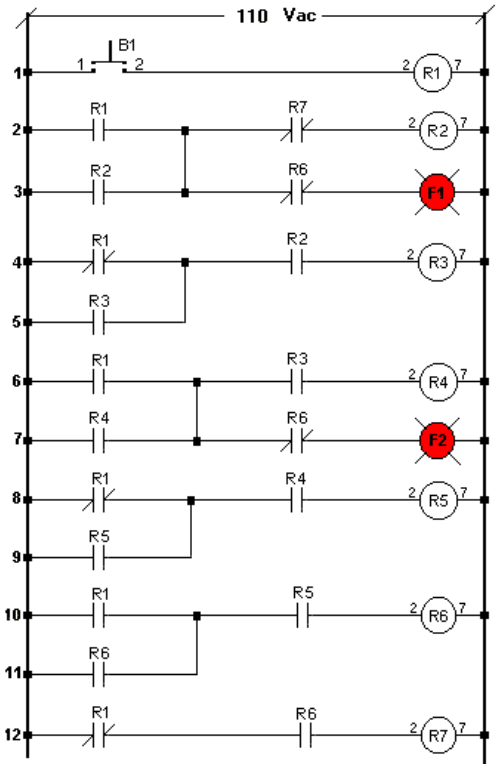


Figura 1.5.22

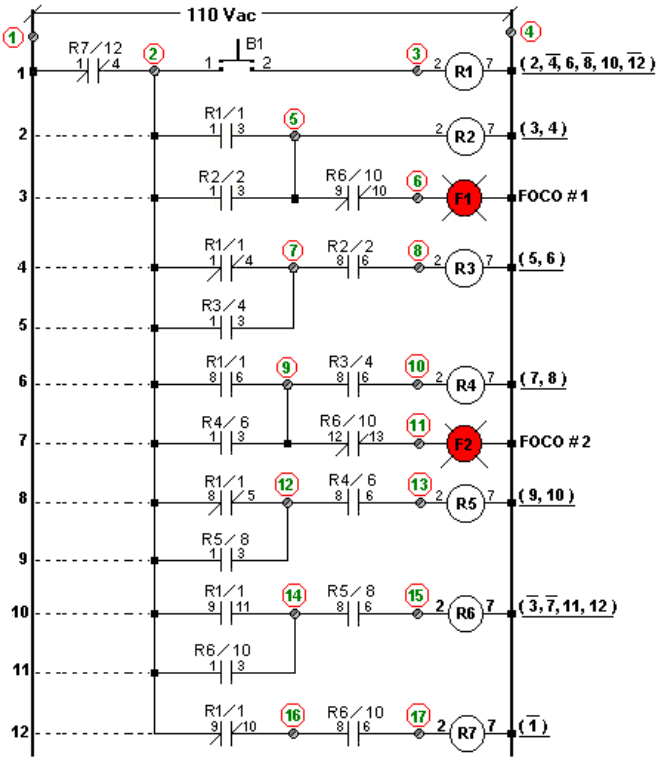


Figura 1.5.23

EJEMPLO 5:

Diseñe un circuito con cuatro focos y un botón, y que funcione de la siguiente manera;

1.- Que cuando se oprima el botón se energice el foco 1.
2.- Que al oprimir el botón por segunda vez, se apague el foco 1 y encienda el foco Nº 2.
3.- Que al oprimir el botón por tercera vez, se apague el foco 2 y encienda el foco 3 y el foco 1 continué apagado.
4.- Que al oprimir de nuevo el botón se apague el foco 3 y prenda el Foco 4 y los focos 1 y 2 permanezcan apagados.
5.- Que al oprimir el botón de nuevo se apague el foco 4 y prenda el foco 1 y así sucesivamente.

SOLUCIÓN:

Como se puede observar este es un contador de anillo.

NOTA: Se le llama contador de anillo porque si las salidas se colocan en círculo, y por cada pulso que se aplica, la salida va cambiando de posición, de tal manera que a estos pulsos se le de cierta frecuencia, tendrá la apariencia de un anillo.

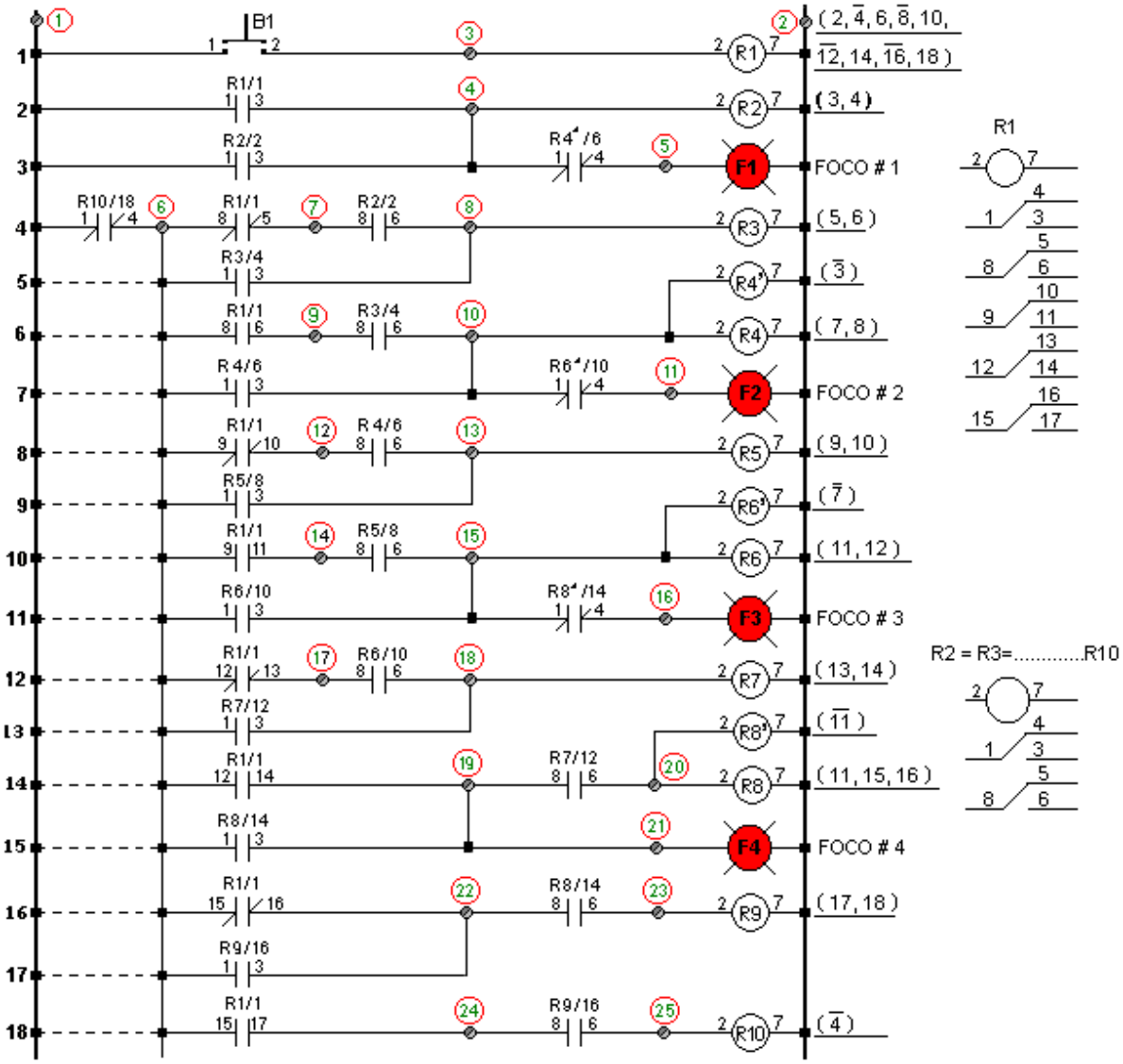


Figura 1.5.24

EJEMPLO 6:

Realice el circuito anterior, pero ahora coloque un botón de paro, de tal manera que si se oprime un instante el botón de paro en cualquier momento, cuando se apague el foco 4 ya no prenda el foco 1 indicando que terminó el ciclo. Si se le da un nuevo pulso, que inicie el contador de anillo.

SOLUCIÓN:

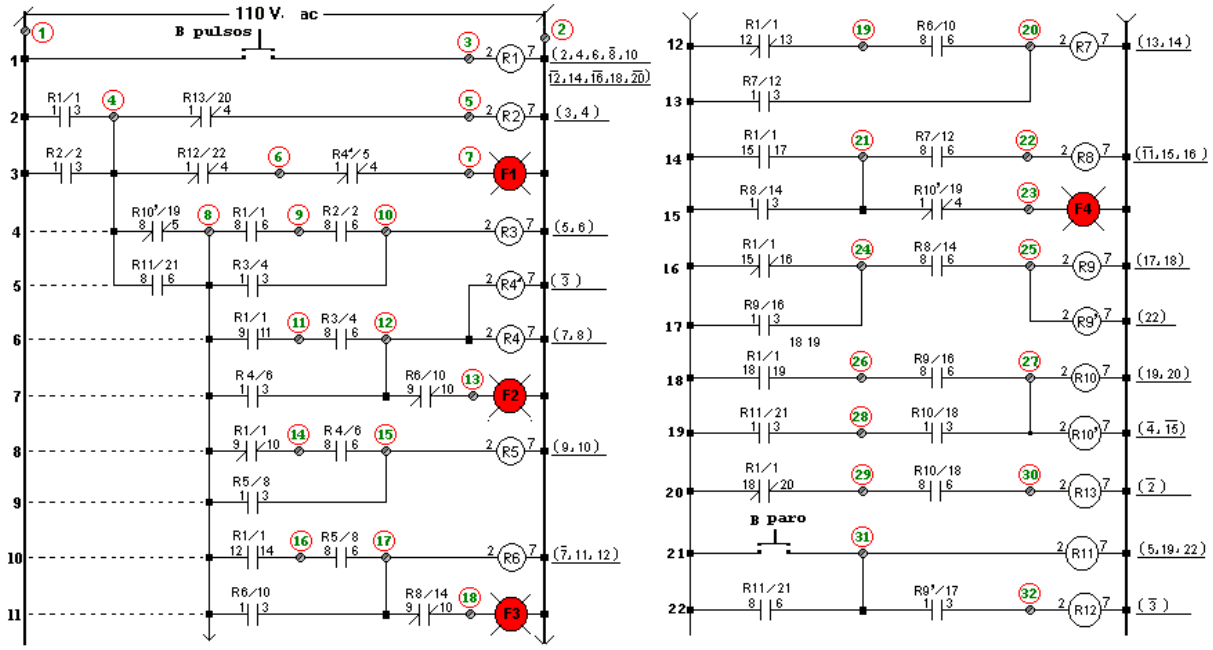


Figura 1.5.25

Observe como, cuando se da el botón de paro en cualquier instante, se memoriza R11, y un contacto de R11 se coloca en la línea 5 en paralelo con R10, de tal manera que cuando se energiza R10 se apaga el foco 4 pero no se cae todo el circuito además de que R10 se memoriza y cuando soltamos el botón de pulsos se energiza R13 el cual tiene un contacto en la línea 2 y desenergiza R2 este a su vez es el permisivo para que pueda trabajar todo el circuito.

EJEMPLO 7:

Diseñe un circuito con un botón de pulsos y cuatro focos que haga lo siguiente:

1.- Que al oprimir el botón se energice el foco 1.
2.- Que al oprimir de nuevo el botón de pulsos se deberá encender el foco 2 y el foco 1 deberá continuar encendido.
3.- Oprimiendo de nuevo el botón de pulsos deberá encender el foco 3 y el 1 y 2 deberán continuar encendidos.
4.- Al oprimir de nuevo el botón de pulsos se prende el foco 4 y el foco 1, 2 y 3 permanecen encendidos.
5.- Al oprimir de nuevo el botón de pulsos se deberá apagar el foco 1 y los focos 2 ,3 y 4 deberán continuar encendidos.
6.- Al oprimir de nuevo el botón de pulsos se deberá apagar el foco 2 y continuar apagado el foco 1 y prendidos el 3 y 4.

7.- Al oprimir de nuevo el botón de pulsos se deberá apagar el foco 3 y los focos 1 y 2 deberán permanecer apagados y solo el foco 4 permanece prendido.

8.- Si se oprime de nuevo el botón de pulsos, deberá apagarse el foco 4 y permanecer apagados los focos 1, 2, 3 y quedar todo en condiciones de repetir el ciclo.

SOLUCIÓN:

Como se podrá observar este es un contador de Johnson.

NOTA: Para explicar el contador de Johnson imaginemos que tenemos cuatro salidas (focos) y por cada pulso que se le aplique se va energizando una salida y permanecen energizadas, una vez que se energizaron las cuatro salidas al siguiente pulso se empiezan a desenergizar en sentido inverso.

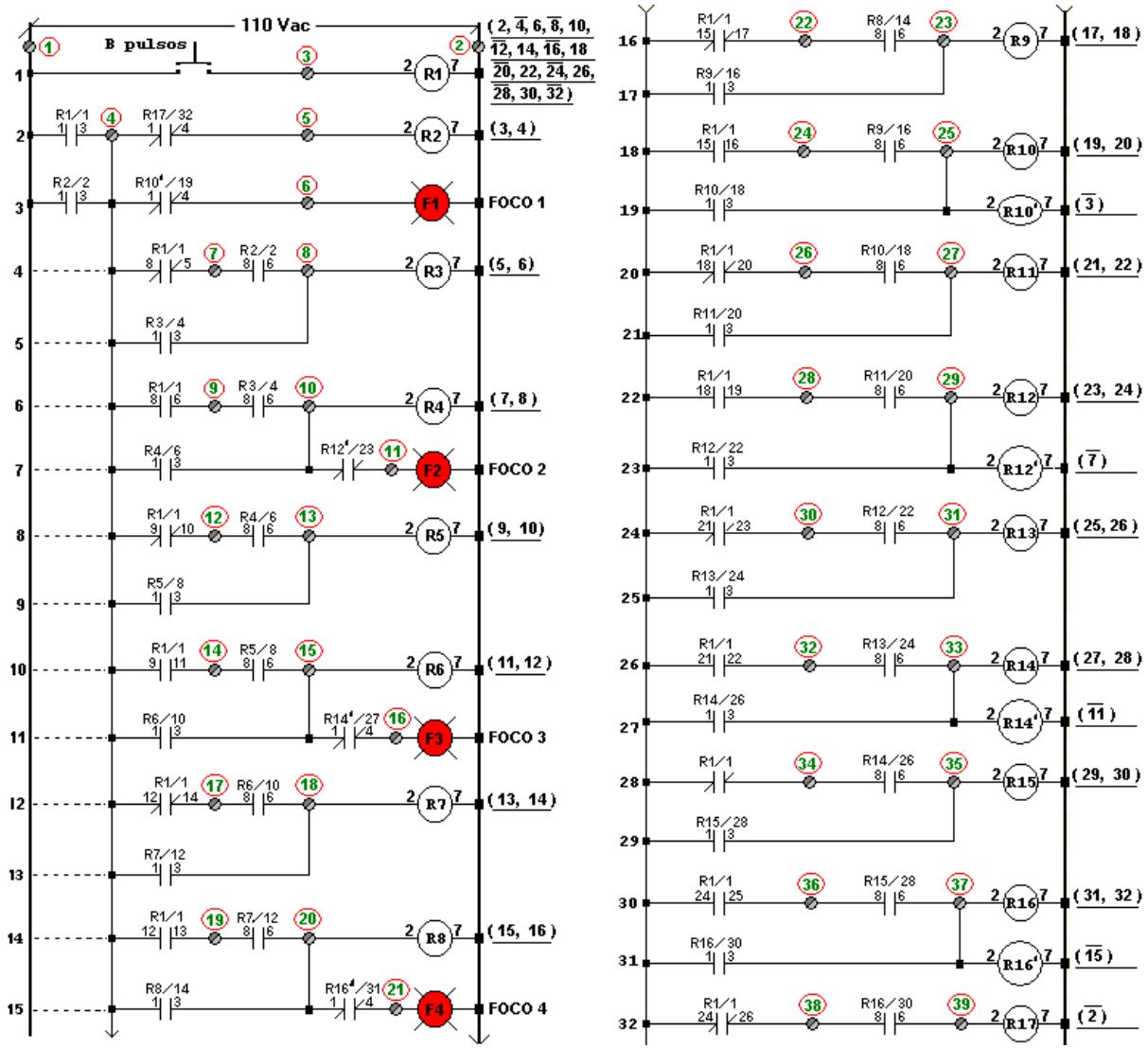
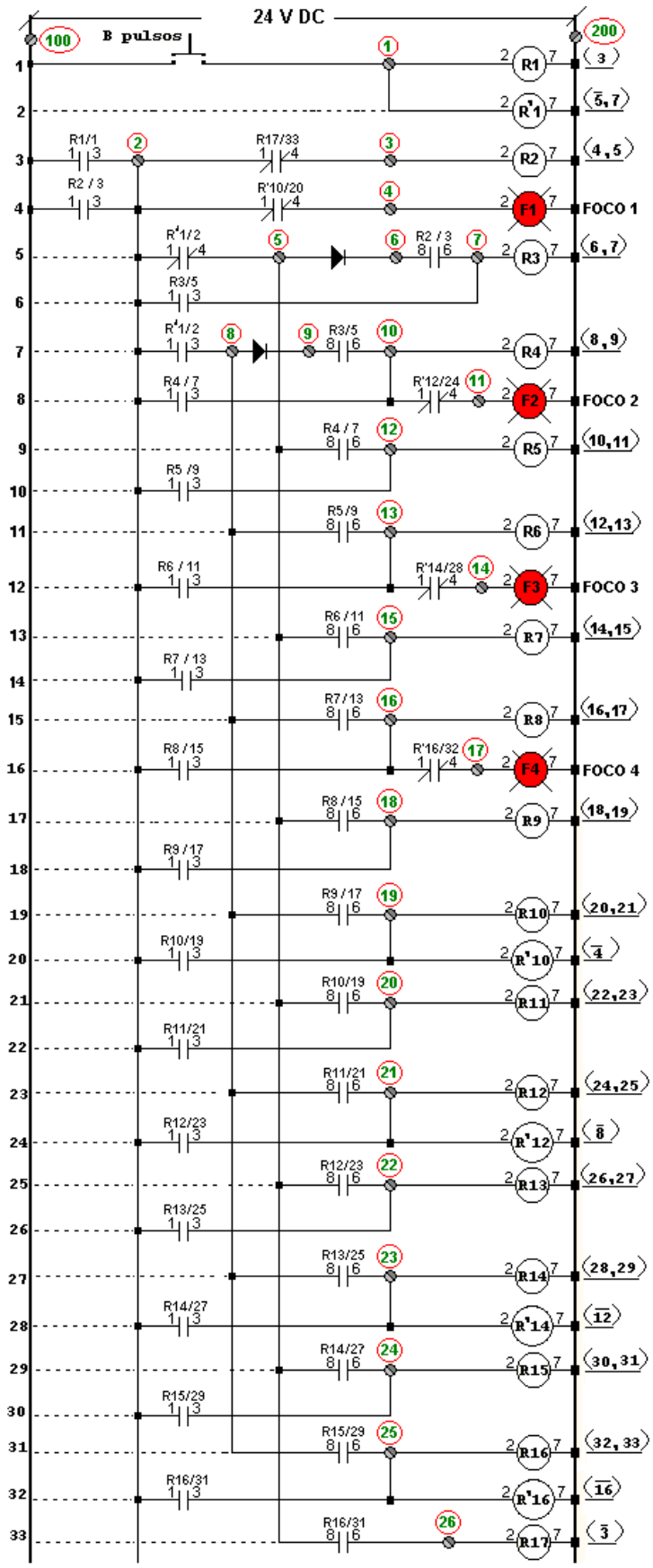


Figura 1.5.26



EJEMPLO: 8

Diseñe el circuito anterior pero ahora, se desea reducir al máximo el número de relevadores utilizados.

Para ello primero requerimos que nuestro control sea con alimentación de corriente directa y utilizamos la reducción del circuito por medio de diodos de control, tal como lo muestra la figura 1.5.27

Figura 1.5.27

EJEMPLO 9 Diseñe un contador binario con cuatro dígitos. **SOLUCIÓN:**

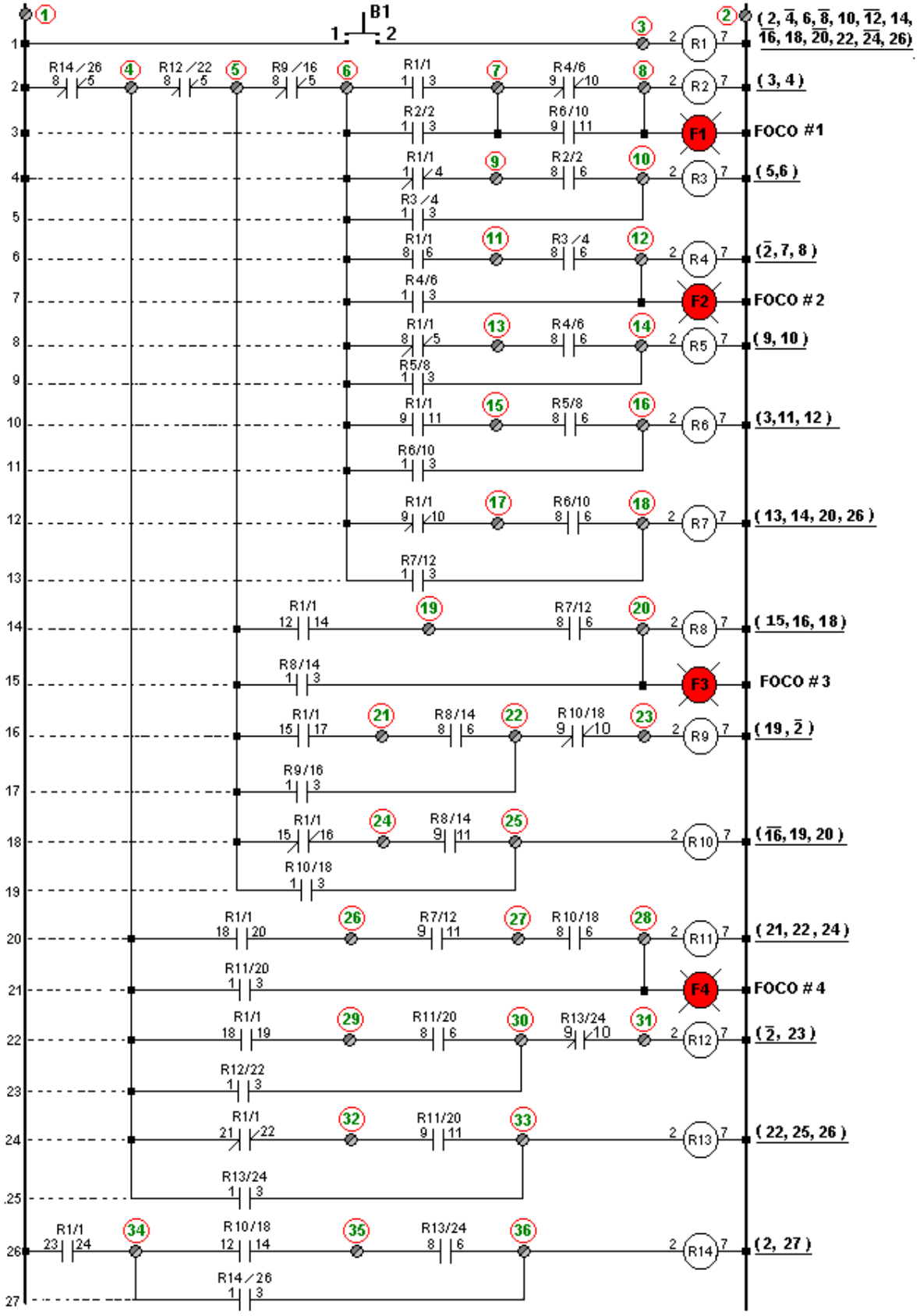
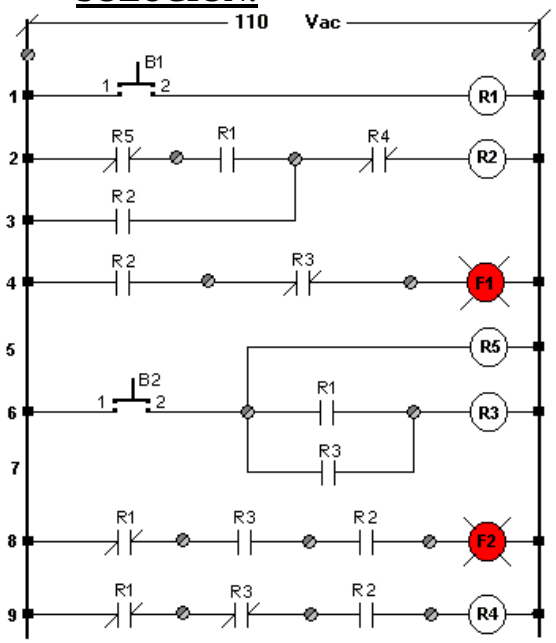


Figura 1.5.28

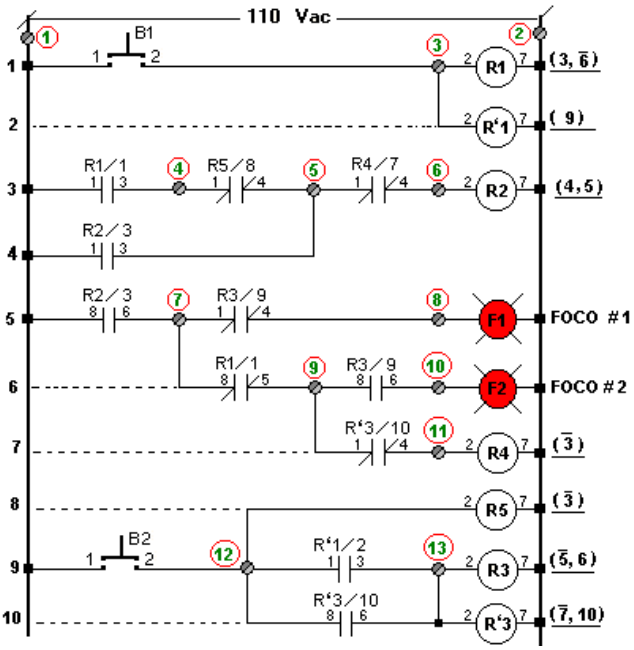
EJEMPLO 10

Se tienen dos botones B1 y B2 y se desea que al oprimir B1 se encienda el foco 1; luego al oprimir B2 y mantener oprimida a B1 se apague el foco 1; finalmente al abrir B1 se encienda el foco 2 y al soltar el botón B2 se apague. Tenga cuidado que se respete la secuencia deseada, esto es que si se oprime primero el botón B2 no pase nada y luego si se oprime el botón B1 habiendo oprimido primero el botón B2 no pase nada.

SOLUCIÓN:



1.5.29a



1.5.29b

1.6 Método intuitivo en la solución de secuenciales.

Como sé ha observado, los circuitos analizados corresponden a los sistemas Set Reset donde el Set nos recuerda que se hizo presente una señal y el Reset nos tumba dicha señal.

Ahora analizaremos circuitos en los que se requiere que nos recuerde que dicha señal se hizo ausente para que si queremos que la señal se haga presente de nuevo, tengamos una memoria que nos recuerde que se hizo ausente dicha señal, de tal manera, que si se hace presente de nuevo dicha señal, junto con la señal de que se hizo ausente podamos contar cuantas veces se ha hecho presente.

En electrónica se les llama Flip Flop maestro esclavo, esto es, se tienen dos Flip Flop donde uno corresponde al maestro, el cual cambia a la subida del clock y otro que es el esclavo el cual cambia a la caída del clock.

En electricidad simplemente se dice, "algo que nos recuerde que se hizo presente una señal y algo que nos recuerde que se hizo ausente".

Para el análisis de sistemas donde se tienen variables que se repiten en el tiempo, existe un método de diseño a partir de ciertas suposiciones intuitivas, pero hay que aclarar que este método no trabaja con eficiencia cuando los sistemas son complejos.

Debemos ser claros al definir los criterios a seguir en el diseño, ya que no existen fundamentos teóricos en que este basado el método, por lo tanto debe tomarse este método como un respaldo sin que se sustituya a la intuición lógica y a la práctica.

1.6.1 Ejemplo de circuitos con el método intuitivo

EJEMPLO 10:

Para la demostración del método intuitivo se utilizara el *"ejemplo clásico de la secuencia"* en el que se tienen dos botones B1 y B2 y se desea que al oprimir B1 se encienda el foco 1; luego al oprimir B2 y mantener oprimido B1 se apague el foco 1; finalmente al volver a abrir B2, se encienda el foco 2 y al soltar B1 se apague.

SOLUCIÓN:

La parte más importante del método consiste en desarrollar una tabla de tiempo, en la que se especifique la secuencia que siguen todas las variables a través del tiempo. La construcción de dicha tabla se efectúa poniendo el tiempo como abscisa y en la ordenada se colocan todas las variables, indicando los intervalos de tiempo donde se hacen presentes dichas variables.

En la figura 1.6.1 se observa el circuito de secuencia dentro de una caja negra a analizar.

Figura 1.6.1

En la figura 1.6.2 se observa que al estar presente B1 de t0 a t1 existe la salida F1; al hacerse presente B2 desaparece la salida F1 y cuando se hace ausente B2 se hace presente F2 de t2 a t3 y finalmente al hacerse ausente B1 sé desenergiza todo el sistema.

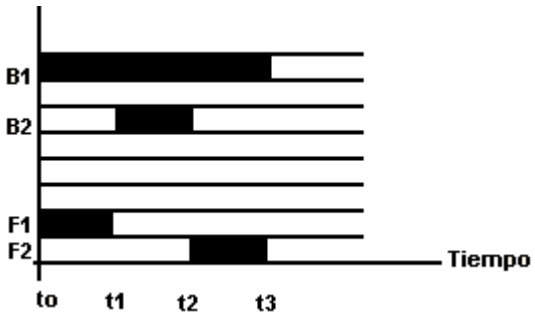


Figura 1.6.2

Construyendo la tabla de tiempo para el presente problema, se obtiene la figura 1.6.2, la cual indica que al estar presente B1, de t0 a t1 existe una salida F1; luego, al hacerse presente B2 y estar

presente B1 de t1 a t2 se hace ausente F1; luego cuando se hace ausente B2 de t2 a t3 se hace presente F2; finalmente cuando se hace ausente B1 sé desenergiza el sistema.

El problema como se puede observar en la figura 1.6.2 es que en los intervalos de tiempo de t0 a t1 y de t2 a t3 las entradas son idénticas, sin embargo, las salidas son diferentes. Para resolver esto, es necesaria la generación de señales secundarias en el sistema. Véase la figura 1.6.3.

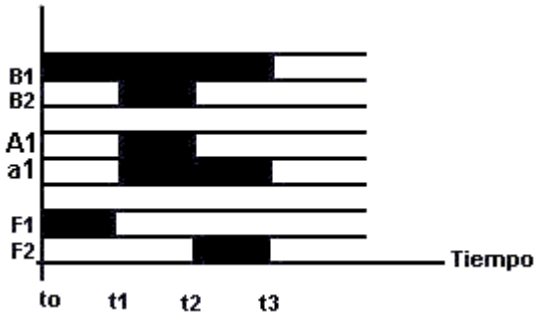


Figura 1.6.3

Una forma de lograr diferenciar un sistema de otro es generar una señal secundaria A1 con la señal de B2, que a su vez generara una señal a1; debido a que el tiempo está indefinido, este proceso debe ser a través de una memoria la cual deberá desaparecer cuando desaparezca la señal de B1. Si observamos la figura 1.6.3 podemos ver que los intervalos de tiempo de t0 a t1 y de t2 a t3 son completamente diferentes y por lo tanto no hay problema en que originen distintas señales.

A continuación se procede a generar las ecuaciones correspondientes a los intervalos de tiempo.

de t0 a t1	$F1 = B1\,\overline{B2}\,\overline{a1}$
de t1 a t2	$a1 = (\overline{B2} + a1)\,B1$
de t2 a t3	$F2 = B1\overline{B2}a1$

Una vez que se tienen las ecuaciones se procede a realizar el circuito.

Primero en la figura 1.6.4 se representa el circuito a partir de las ecuaciones correspondientes. En la figura 1.6.5 se aprovecho que R1 es común y se eliminaron los contactos de R1. Finalmente, en la figura 1.6.6 se observa como el contacto de R3 también es común, con lo cual se procedió a eliminarlo quedando el circuito final.

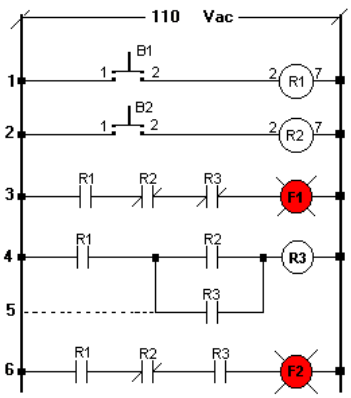


Figura 1.6.4

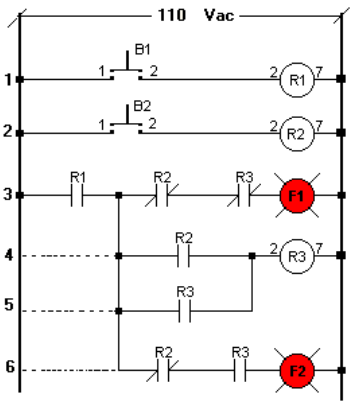


Figura 1.6.5

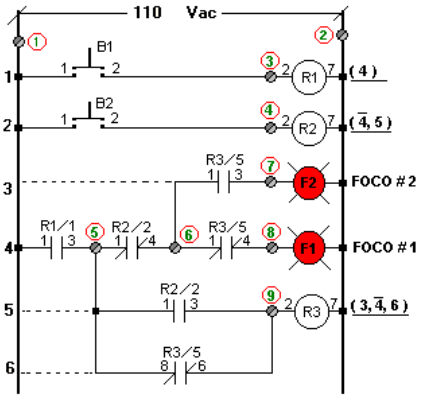


Figura 1.6.6

Este mismo circuito lo podemos resolver con un razonamiento lógico y algo de práctica, considerando los criterios de las memorias que nos recuerdan cuando algo este presente y cuando la misma señal se hace ausente.

Primero tenemos que cuando el B1 se hace presente que se prenda el foco 1, además B1 es el permisivo para que trabaje cualquier foco.

Luego procedemos a oprimir el botón 2 y que se apague el foco 1, aquí observamos que podemos poner un contacto cerrado de R3 en serie con el foco 1 de tal manera que cuando se active el botón 2 y se energice R2 se abra el contacto y se apague el foco1, pero al soltarlo, el foco 1 debe permanecer apagado por lo que necesitamos una memoria que nos recuerde que B2 se activo una vez y el circuito quedaría como lo muestra la figura 1.6.7

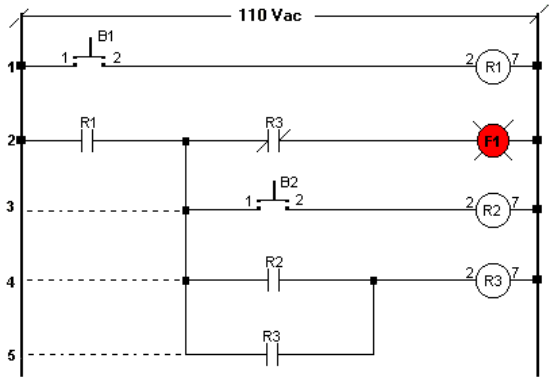


Figura 1.6.7

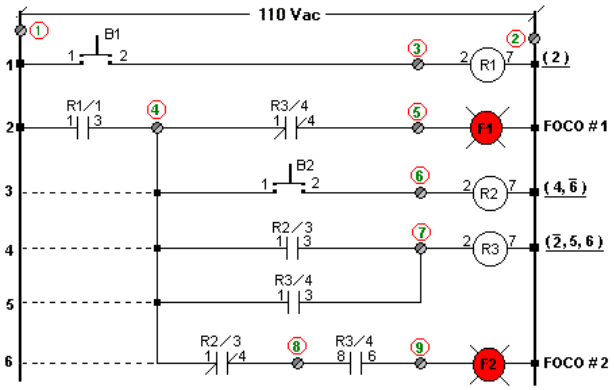


Figura 1.6.8

Ahora faltaría solo que cuando se desactive el B2 se prenda el Foco 2, por lo tanto necesitamos una señal que nos recuerde que se hizo presente B2 el cual esta recordado con el R3, luego R2 sé desenergiza cuando soltamos B2, por lo tanto podemos colocar un contacto cerrado de R2 en serie con un contacto de R3 para energizar el foco 2, tal como lo muestra la figura 1.6.8

Al soltar el B1, como es el permisivo para que trabaje el sistema, al soltarlo se desenergiza todo el sistema.

Ahora como corolario, podemos realizar los siguientes cambios y de esta manera podemos realizar modificaciones a los circuitos de acuerdo a las necesidades propias de una automatización.

Consideremos lo siguiente:

Imaginemos que se hace presente primero B1 y que se prenda el Foco #1, luego, si se hace presente B2 que se apague el Foco #1, hasta ahora podemos observar que el ejercicio es similar al anterior, observe el siguiente paso.

Si se hace ausente la señal de B1 que se prenda el Foco #2 y si se hace ausente el botón #2 que se apague el Foco #2.

Como podemos observar las condiciones son diferentes aun cuando el número de variables son las mismas, a esto se le llama el ejemplo clásico de la secuencia donde los resultados son diferentes en condiciones diferentes con las mismas variables de entrada en momentos diferentes.

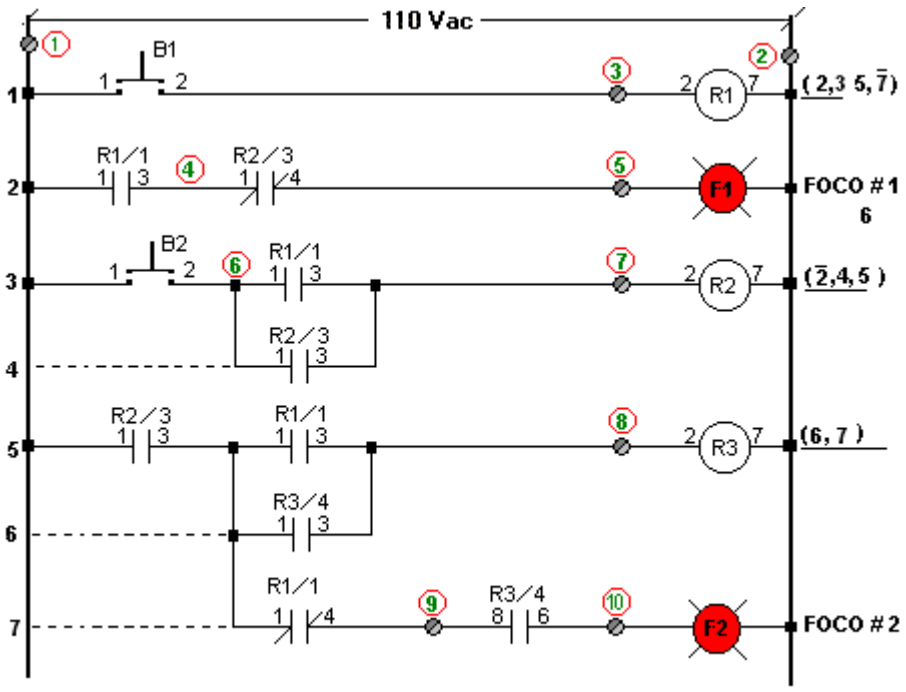


Figura 1.6.9

Este ejercicio es fundamental para que la persona que tome estos apuntes pueda realizar modificaciones a las maquinas de acuerdo a las necesidades propias del cliente y bajo los criterios que requiera el proceso.

1.7 Circuitos mono-estables. (Donde interviene el tiempo)

Como podemos observar hasta ahora, los circuitos analizados pertenecen a los secuenciales por que intervienen las memorias, pero el tiempo de respuesta son por tiempos indeterminados, ahora veremos circuitos donde el tiempo se hace determinado.

Cuando iniciamos el estudio de los circuitos secuenciales dijimos que los multivibradores Mono-estables eran aquellos que tenían un estado estable y el otro casi estable y dábamos por ejemplo el temporizador o relevador de tiempo o timer.

Existen dos tipos de temporizadores;

On delay o retardo a la conexión.
Off delay o retardo a la desconexión.

On delay: Son aquellos relevadores en los que al energizar la bobina empieza a transcurrir el tiempo, (éste tiempo es previamente ajustado, y se les llama con retardo a la conexión porque retarda la señal de entrada) cuando completa este tiempo cambian de estado sus contactos y permanecen en ese estado mientras esté energizada su bobina, cuando se desenergiza la bobina dichos contactos regresan a su estado original.

Dentro de este tipo de temporizadores los podemos encontrar en el mercado con contactos instantáneos que funcionan igual a los relevadores normales, en realidad lo que pasa es que dentro de los temporizadores traen un relevador normal.

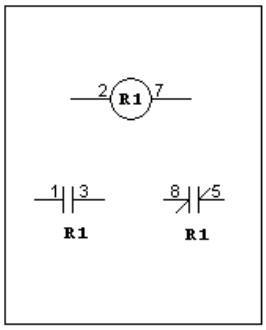
Para diferenciar un relevador normal de uno con tiempo se le pone las letras RT1, RT2, etc.
También al contacto abierto se le llama **N.A.T.C** que quiere decir normalmente abierto con tiempo para cerrar. Y al contacto cerrado se le llama **N.C.T.A** que quiere decir normalmente cerrado con tiempo para abrir.

Off delay: Son aquellos relevadores en los que al energizar la bobina sus contactos cambian de estado inmediatamente y permanecen en ese estado mientras este energizada la bobina y al desenergizar la bobina empieza a transcurrir el tiempo (este tiempo es previamente ajustado y se le llama con retardo a la desconexión porque el tiempo empieza a transcurrir al desenergizar la bobina) cuando completa su tiempo regresan sus contactos a su estado original.

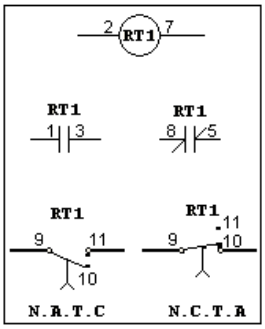
También al contacto abierto se le llama **N.A.T.A** que quiere decir normalmente abierto con tiempo para abrir. Y al contacto cerrado se le llama **N.C.T.C** que quiere decir normalmente cerrado con tiempo para cerrar.

NOTA: Siempre que sea posible se deberán utilizar temporizadores **ON DELAY** ya que son más económicos.

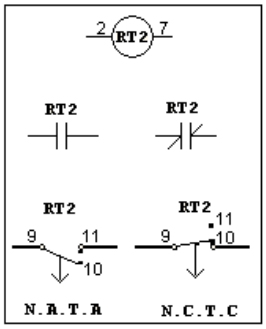
Simbología:



RELEY NORMAL



TEMPORIZADORES
ON DELAY
(Retardo a la conexión)



TEMPORIZADORES
OFF DELAY
(Retardo a la desconexión)

NOTA: Observe las flechas de los contactos

1.7.1 Ejemplos de circuitos Mono estables

EJEMPLO 1:

Se tiene un botón y un foco y se desea que al oprimir el botón se energice el foco y a los 5 segundos que se apague. Resuelva este circuito primero con **ON DELAY** (Figura 1.7.1)y luego con **OFF DELAY** (Figura 1.7.2)**.**

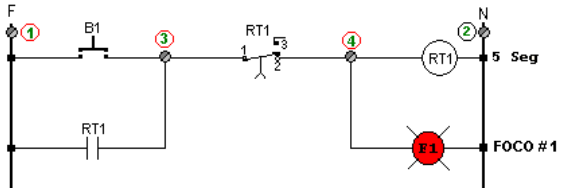


Figura 1.7.1

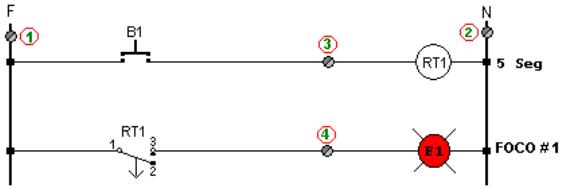


Figura 1.7.2

EJEMPLO 2:

Observe el circuito de la figura 1.7.1 y 1.7.2 se puede ver que si mantenemos oprimido el botón de inicio en la figura 1.7.1 cuando termina el tiempo se apaga el foco pero en un instante muy pequeño (tiempo de respuesta de los platinos en abrir y cerrar), entra de nuevo el foco y el temporizador, mientras que en la figura 1.7.2 el tiempo empieza a contar a partir de que soltamos en botón. Ahora realice las modificaciones para que funcione correctamente aun cuando los botones permanezcan activados, esto es, que no repita ciclo en la figura 1.7.1 y que el tiempo empiece a transcurrir aun cuando se mantenga oprimido el botón de inicio en la figura 1.7.2.

SOLUCION:

Para la solución de estos problemas se hace necesaria la implementación de un circuito denominado "one shot" (un disparo) y la solución se presenta en la figura 1.7.3 y en la figura 1.7.4

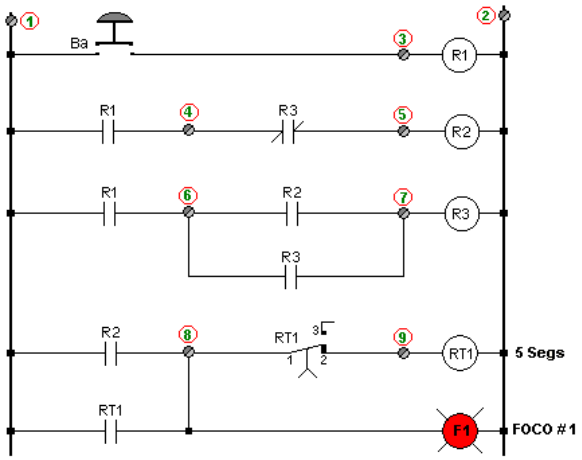


Figura 1.7.3

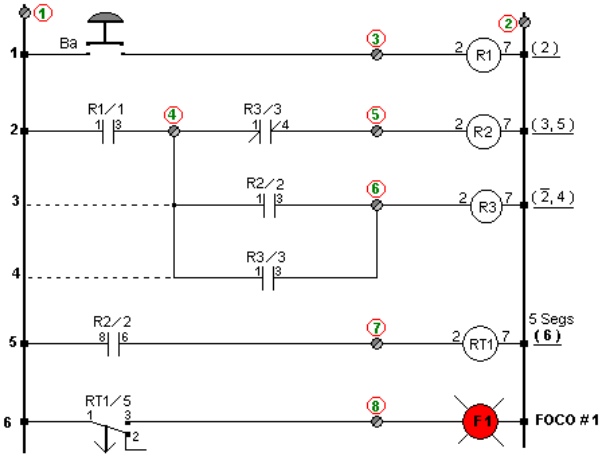


Figura 1.7.4

EJEMPLO 3:

Se tiene un botón 1 y se desea que al darle un pulso a los 5 segundos se prenda un foco 1 y con un botón 2 se apague. (Ver figura 1.7.5)

Observe como en este circuito se esta utilizando un temporizador que tienen contactos instantáneos y contactos con tiempo.

SOLUCIÓN:

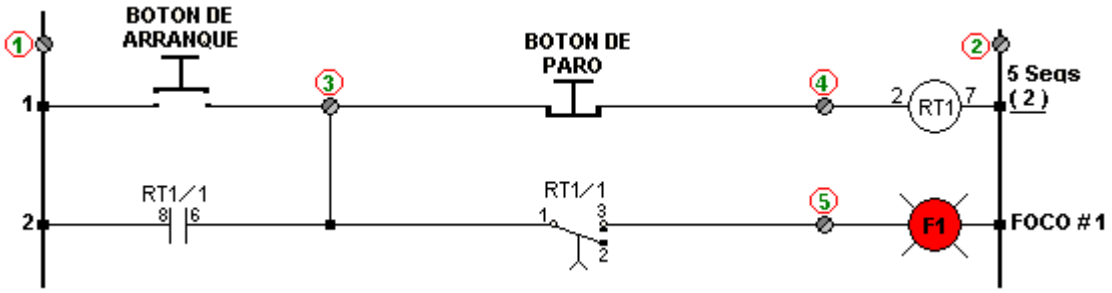


Figura 1.7.5

1.8 Circuitos as-tables. (Donde intervienen los osciladores).

Son aquellos circuitos en los que no se tiene ningún estado estable ejemplo los osciladores

1.8.1 Ejemplos de circuitos As-tables.

EJEMPLO 4:

Se tiene un botón y un foco y se desea que al oprimirlo a los 5 segundos se prenda el foco 1 y que dure prendido 3 segundos y que se apague.

SOLUCIÓN:

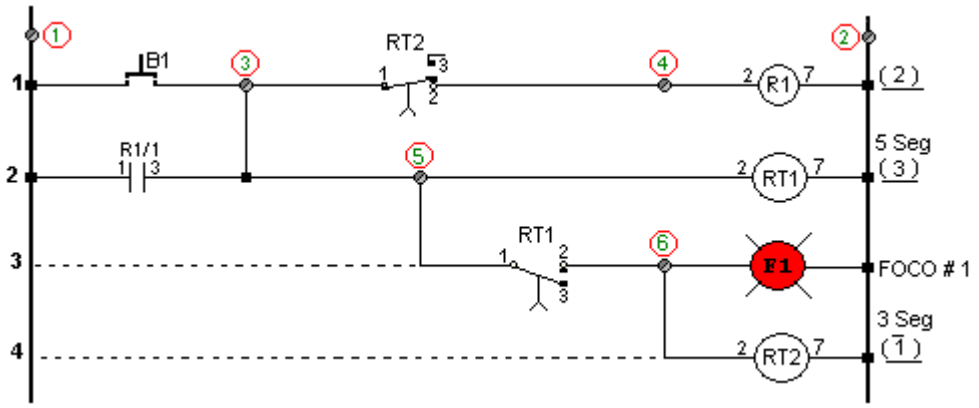


Figura 1.8.1

EJEMPLO 5:

Se tiene un botón y un foco y se requiere, que al oprimir el botón, a los 5 segundos se prenda un foco, que dure 3 segundos prendido y se apague y una vez que se apague, que a los 5 segundos se prenda de nuevo, dure tres segundos prendido y que se apague y así sucesivamente. Para pararlo utilice un botón 2. Como se podrá observar este es un oscilador.

SOLUCIÓN:

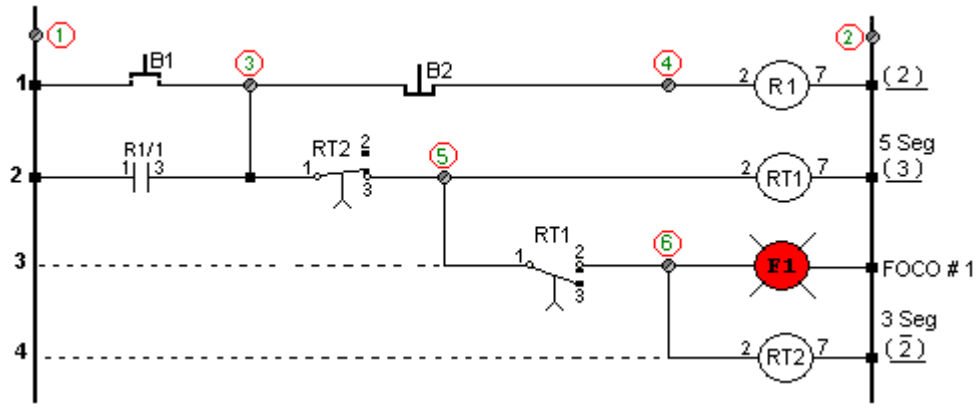


Figura 1.8.2

EJEMPLO 6:

Observe como en la fig. 1.8.2 al oprimir el botón 2 se apaga el foco inmediatamente sin completar su tiempo. Resuelva el circuito anterior pero ahora que al oprimir el botón de paro (B2) si el foco esta prendido que continué prendido y al apagarse que ya no encienda y si el foco esta apagado que ya no encienda.

SOLUCIÓN:

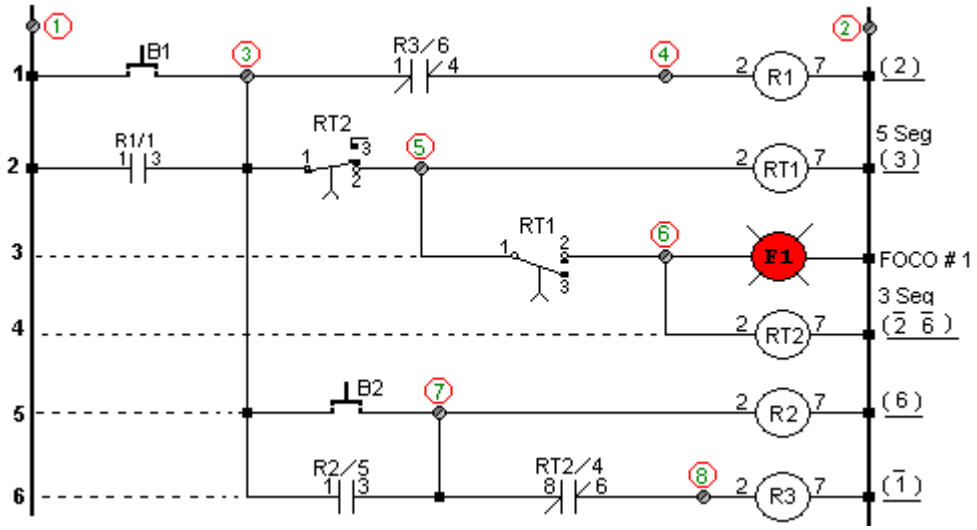


Figura 1.8.2

EJEMPLO 7:

Resuelva el circuito anterior solo que si no se da un pulso al botón de paro al completar 5 ciclos se pare automáticamente y si se da el botón de paro que continúe prendido sí estaba prendido, y si estaba apagado que ya no continúe su ciclo.

SOLUCIÓN:

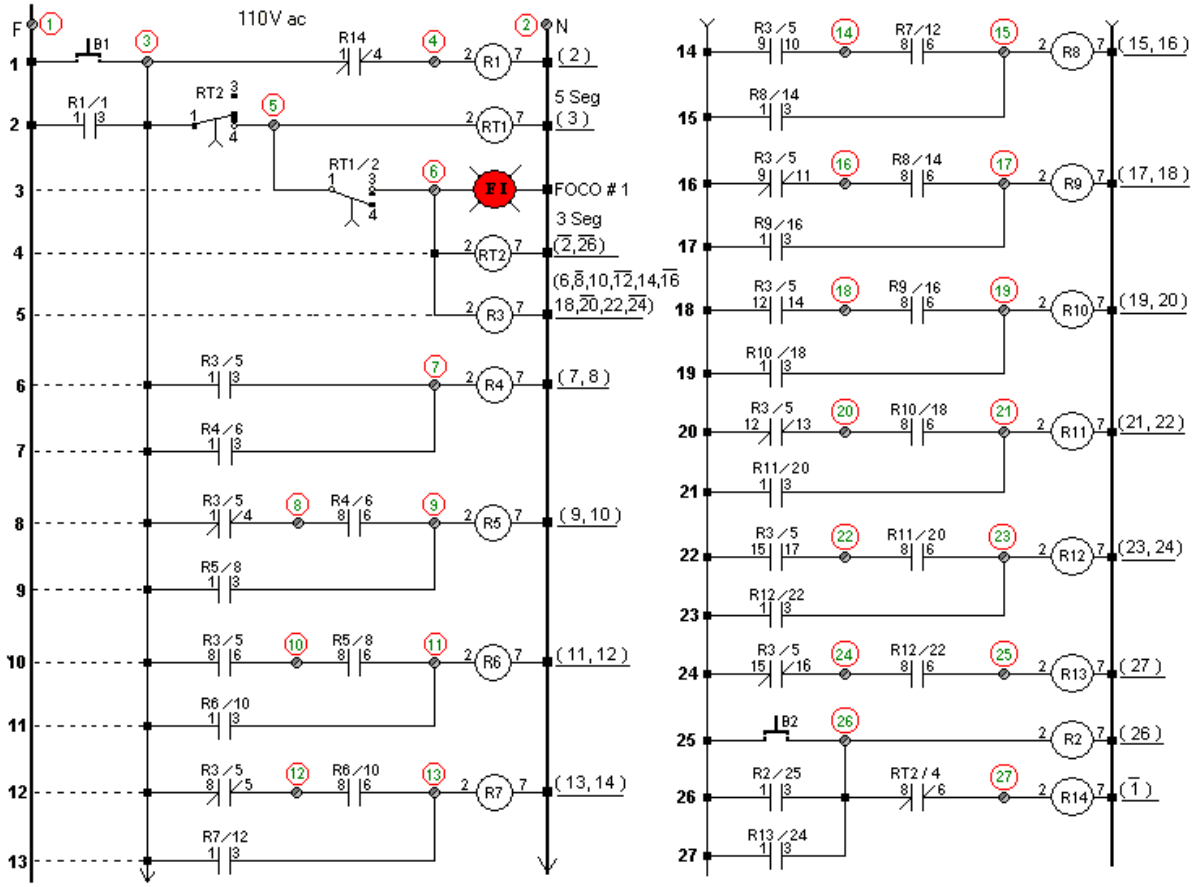


Figura 1.8.3

EJEMPLO 8:

Diseñe el circuito de control para automatizar una granja, donde se requiere que se prenda la iluminación a las 18:00 Hrs. y se apaguen a las 5:00 hrs. diariamente. Además deberá tener un botón de inicio, el cual se requiere que se de el impulso a las 11:00 Hrs. El sistema deberá tener un botón de paro que tumbe el sistema y que todo quede listo para iniciar de nuevo. (Ver Figura 1.8.4).

SOLUCIÓN:

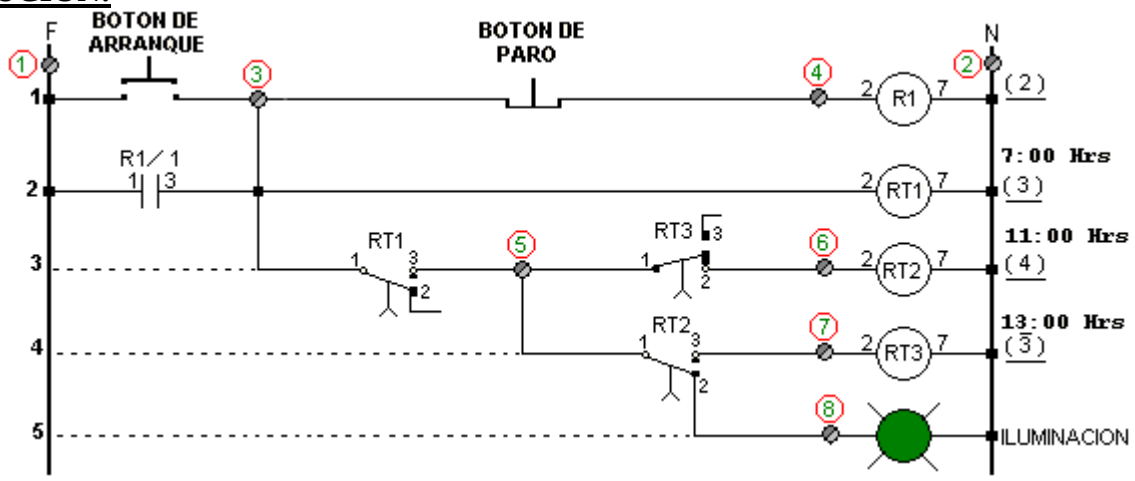


Figura 1.8.4

EJEMPLO 9:

Se tiene un botón y tres motores y se desea que al oprimir el botón de inicio arranque el motor 1 y después de que arranco el motor 1, a los 5 segundos arranque el motor dos y después de 2 segundos que arranco el motor 2 arranque el motor tres y después de 7 segundos que arranco el motor tres, se paren los motores y todo quede en condiciones de iniciar de nuevo (Ver Figura 1.8.5).

SOLUCIÓN:

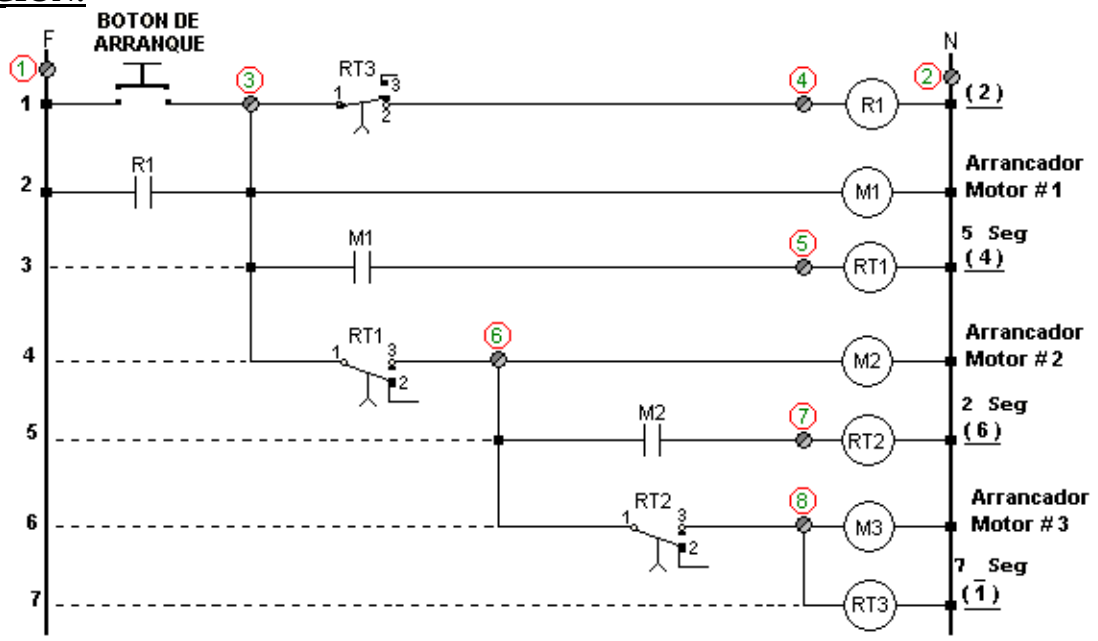


Figura 1.8.5

EJEMPLO 10:

Realice el circuito de control para un motor con tres botones: Botón 1 arranque hacia la derecha, botón 2 arranque hacia la izquierda y botón 3 de paro.

Condiciones de control:

1.- Si el motor esta girando hacia la derecha no podrá girar hacia la izquierda hasta que se de un pulso al botón de paro. Se aplica igualmente en sentido contrario.

2.- Si el motor esta girando hacia la derecha y se oprime el botón de paro no podrá girar el motor en sentido contrario hasta después de 10 segundos para perder la inercia de giro, pero si podrá girar en el mismo sentido si se para y se arranca. Esto aplica en ambos sentidos.

NOTA: Observe como para este circuito se están utilizando **TIMER OFF DELAY** (Ver figura 1.8.6)

SOLUCIÓN:

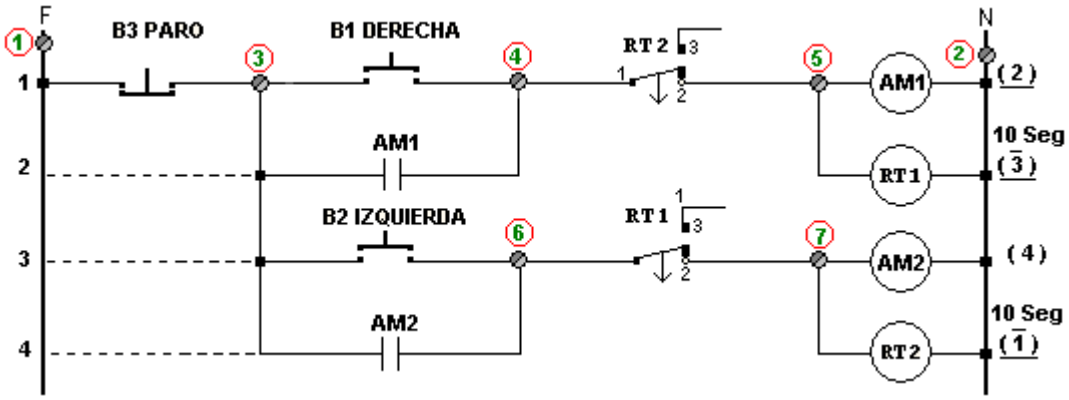


Figura 1.8.6

EJEMPLO 11:

Realice el circuito de control para un motor, en el que se tiene:

1.- Botón hacia la derecha.
2.- Botón hacia la izquierda.
3.- Botón de paro.
4.- Botón de pulsos hacia la derecha. (JOG).
5.- Botón de pulsos hacia la izquierda. (JOG).

NOTA:

1.- Para invertir la rotación se deberá primero oprimir el botón de paro.
2.- Si el motor esta girando en un sentido y se oprime el botón de jogeo en el mismo sentido, al soltarlo deberá pararse.

(Ver figura 1.8.7)

SOLUCIÓN:

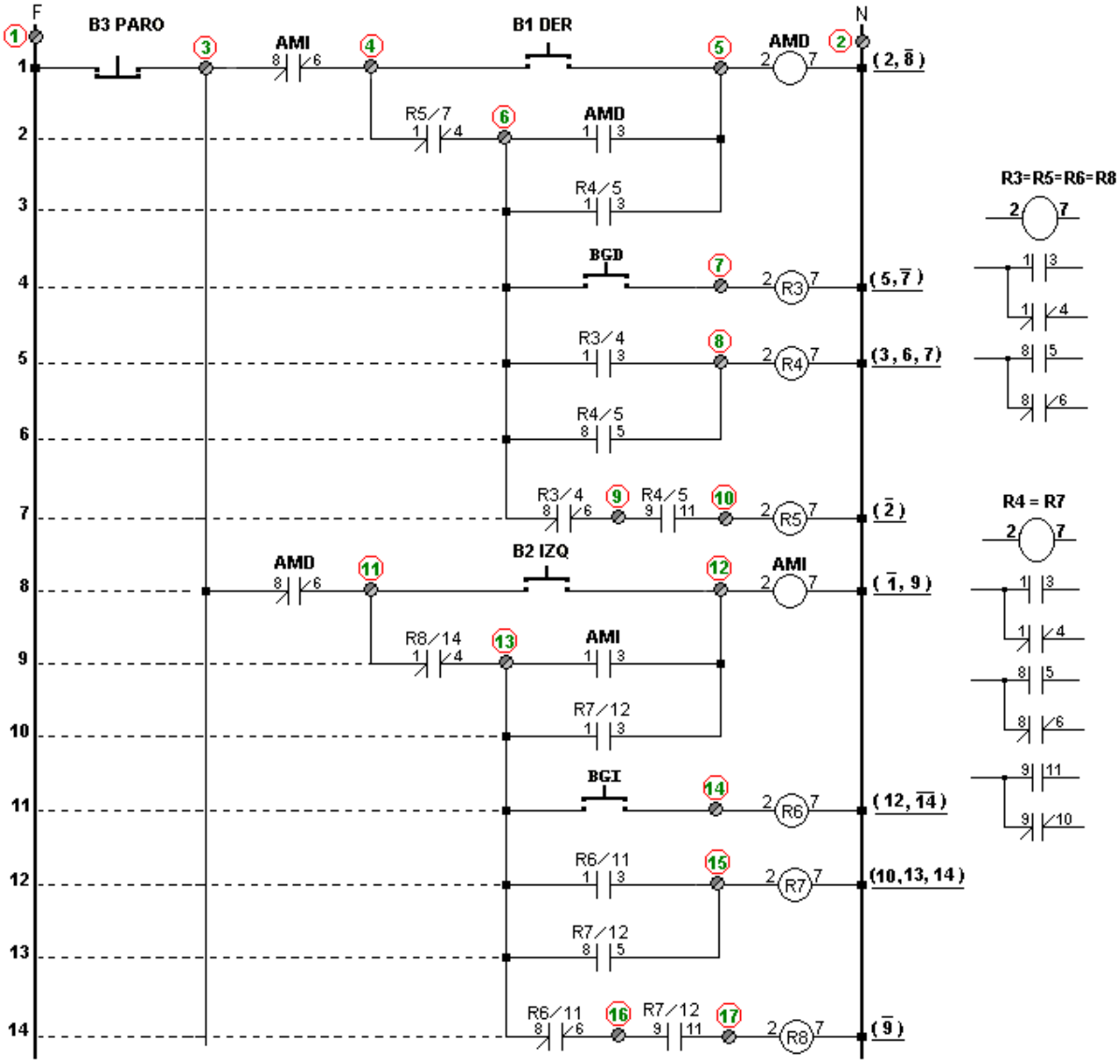


Figura 1.8.7

EJEMPLO 12:

Realice el circuito de control para un motor, en el que se tiene:

1.- Botón hacia la derecha.
2.- Botón hacia la izquierda.
3.- Botón de paro general.

Además se requiere que cuando se pare el motor y se necesite que arranque en sentido contrario, que no se pueda arrancar hasta pasar 5 segundos para que pierda la inercia de giro del motor. Dar la solución con temporizadores ON DELAY.

SOLUCIÓN: (Ver figura 1.8.8)

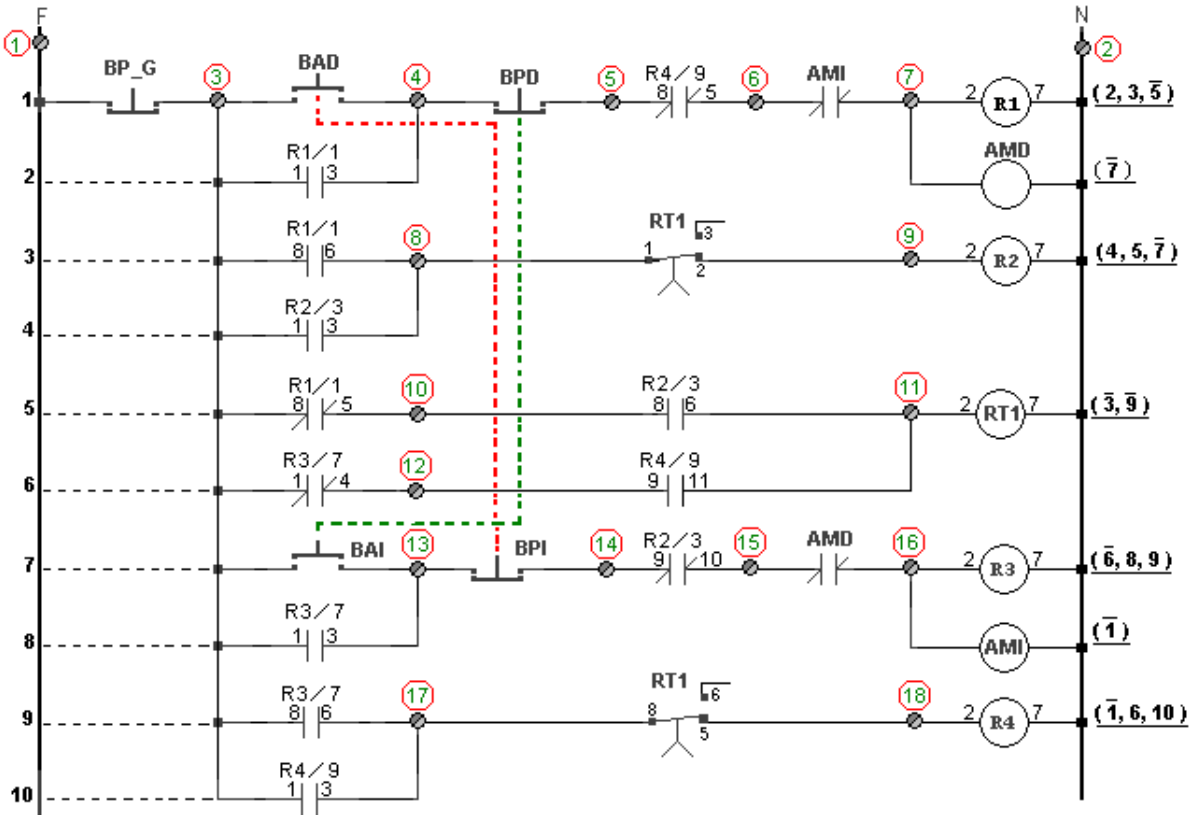


Figura 1.8.8

EJEMPLO 13:

Resolver el circuito del problema anterior pero con temporizadores OFF DELAY (Ver figura 1.8.9)

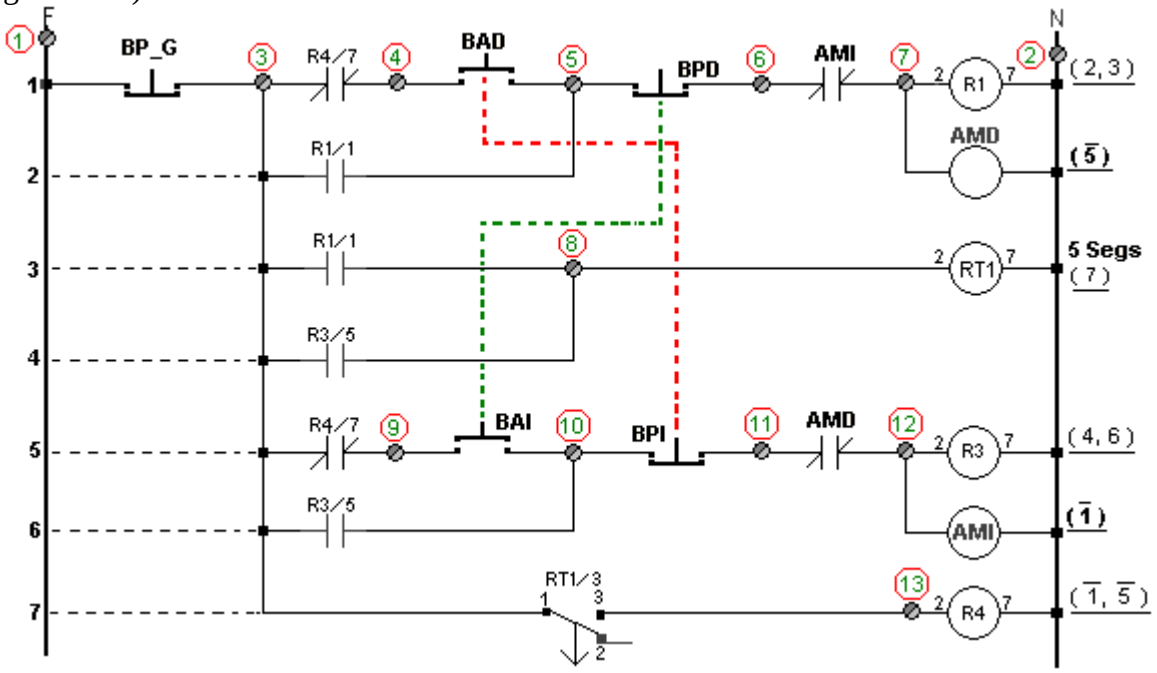


Figura 1.8.9

EJEMPLO 14:

Diseñe el circuito de control para prender la iluminación de un salón de clases de una escuela, en la cual se desea oprimir un botón en punto de las 7:00 hrs. La iluminación deberá prender a las siete de la mañana y apagarse a las 8:50, deberá prender de nuevo a las nueve y apagarse a las 10:45, deberá prender de nuevo a las 11:00 y apagarse 12:45, deberá prender de nuevo a las 13:00 y apagarse a las 14:45.

Las clases se suspenden a las 14:45 y se reinician a las 17:00 por lo cual se deberá prender de nuevo la iluminación a las 17:00 y apagarse a las 18:45, la iluminación se prenderá de nuevo a las 19:00 y apagarse a las 20:45 terminando las actividades y deberá apagarse el sistema en forma automática para quedar listo para el siguiente día laboral. Solución Figura 1.8.10

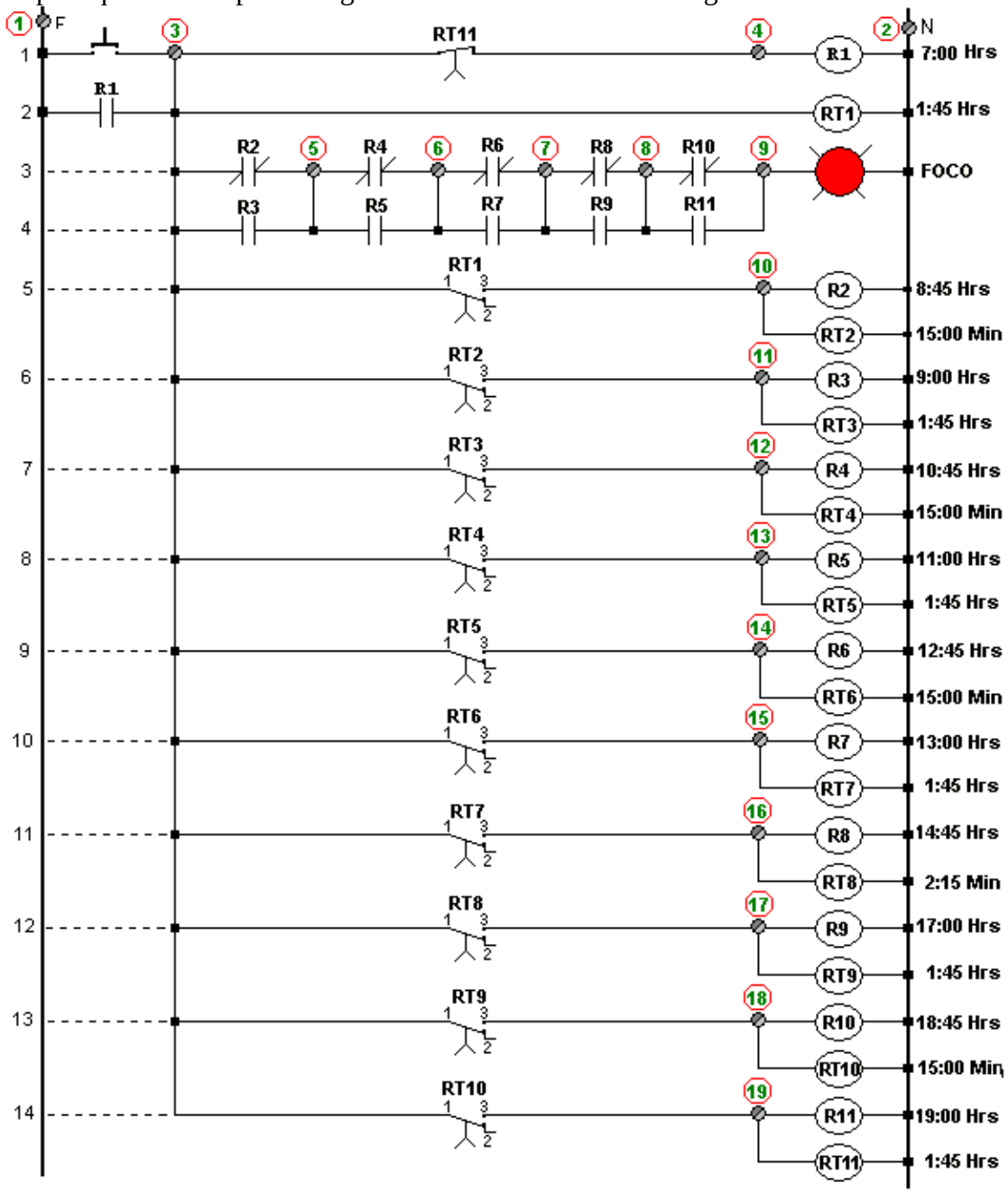


Figura 1.8.10

EJEMPLO 15:

Diseñe el circuito anterior pero ahora se desea que al terminar el turno del día, esto es, a las 20:45 se desea que se apaguen las luces y que en forma automática entre de nuevo a las 7:00 Hrs del día siguiente considerando que las actividades son de lunes a miércoles, esto significa que se oprime el botón el lunes de cada semana y deberá trabajar en forma automática de lunes a miércoles.

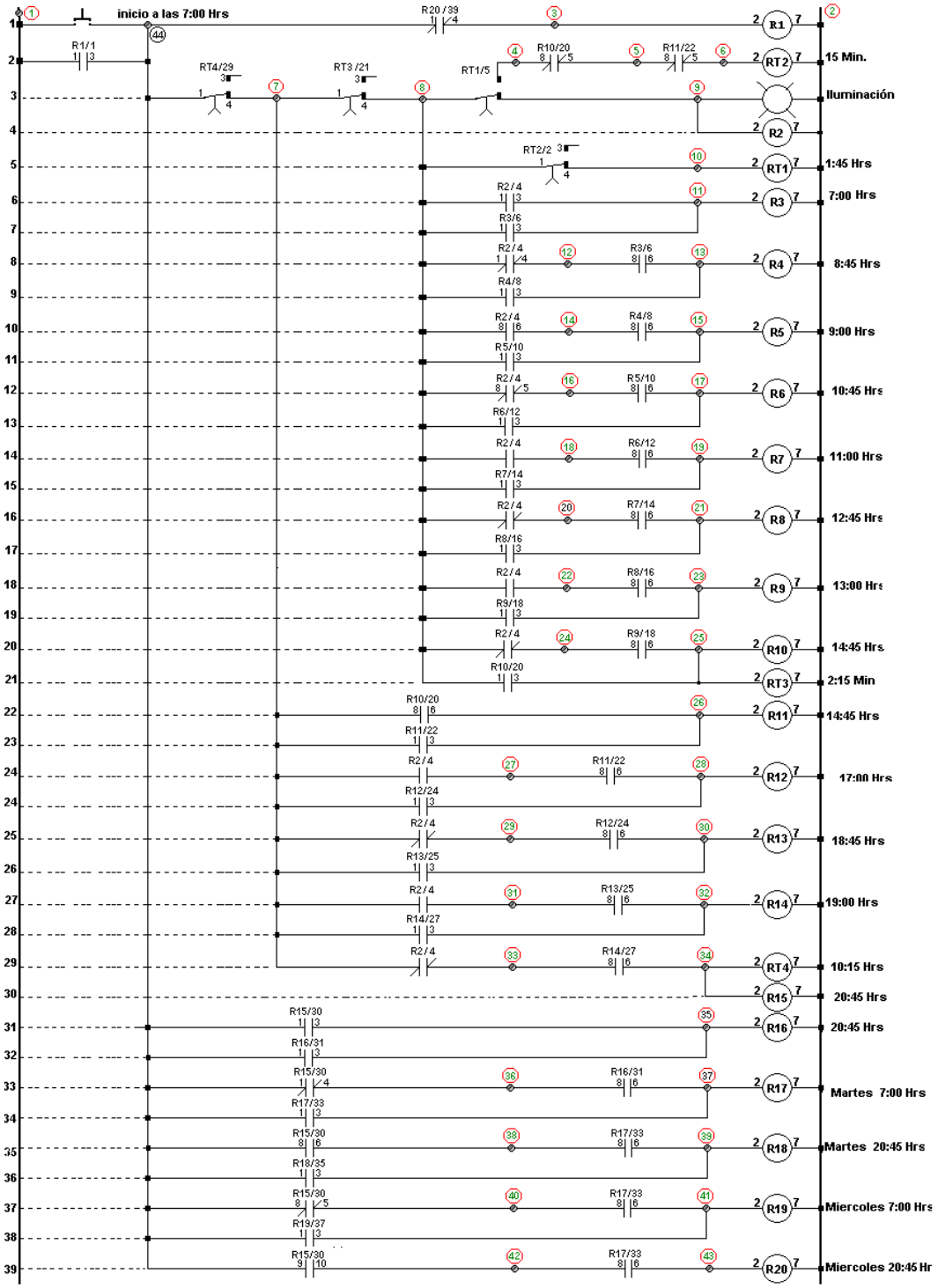


Figura 1.8.11

EJEMPLO 16:

Diseñe el circuito anterior de encendido de luces para una aula de clases pero ahora se desea que al terminar el turno vespertino, esto es, a las 20:45 se apaguen la luces y prendan de nuevo hasta el día siguiente a las 7:00 hrs., continuando el programa de lunes a viernes, esto es, el programa esta diseñado para que funcione una semana, para reiniciarlo deberá oprimirse el botón de inicio el lunes a las 7:00Hrs.

Solución Figura 1.8.12

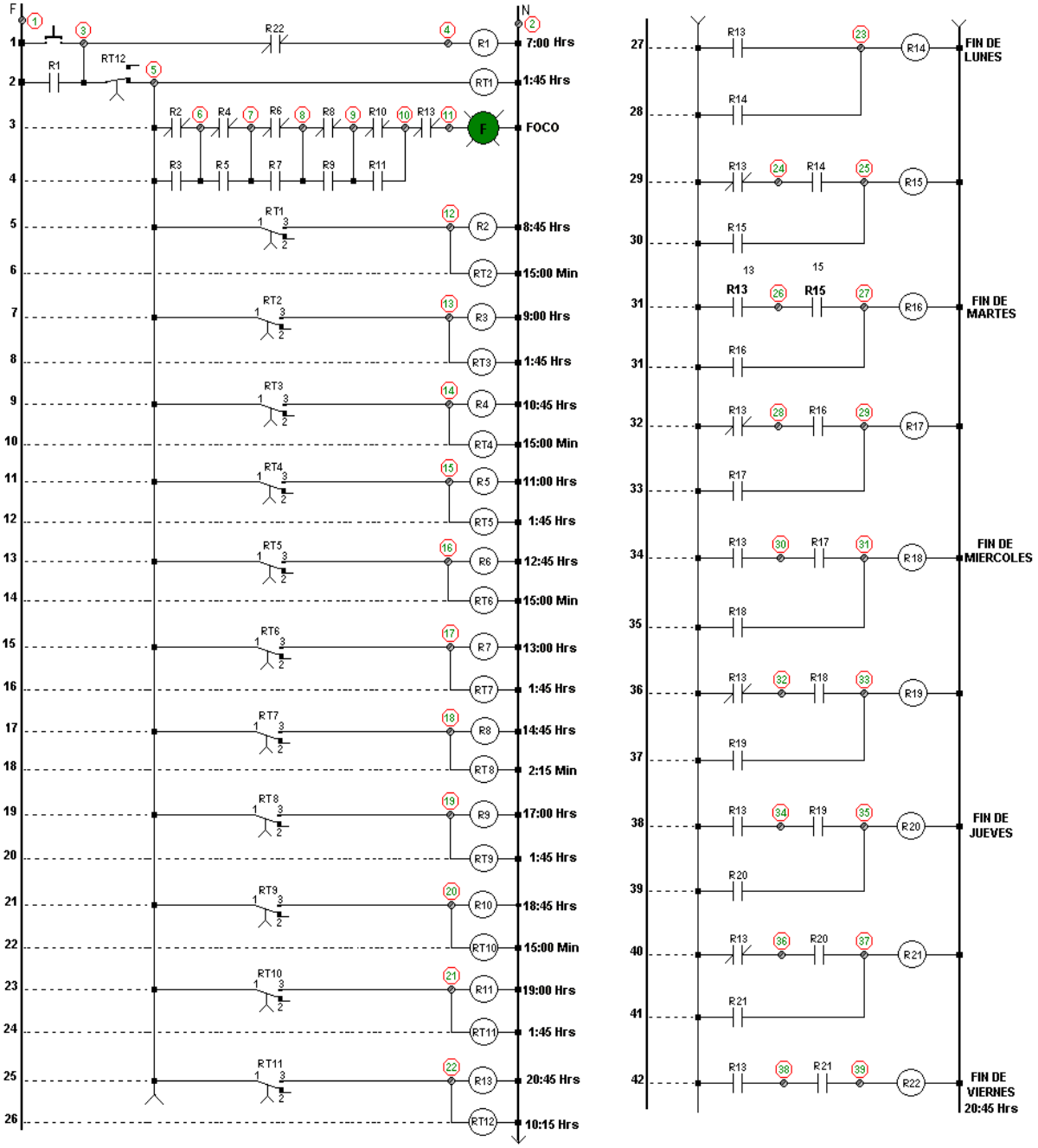


Figura 1.8.12

1.9 **Circuitos con finales de carrera** (Donde interviene el posicionamiento)

Es necesario en automatización que las máquinas no realicen operaciones a menos que estén seguras de que dicha acción no será dañina para el operador o para el proceso. Es aquí donde intervienen los dispositivos de final de carrera, estos interruptores colocados de forma apropiada indicaran a la máquina cuando alguna de sus partes o extensiones ha alcanzado su posición final, esta señal le indica a la máquina que puede continuar con el proceso, o por el contrario es una señal que indica al sistema de control que una pieza o actuador no ha tenido movimiento. A continuación se muestra su simbología de estos interruptores o sensores de final de carrera activados por levas.

Para detectar el posicionamiento se encuentran en el mercado una gran cantidad de sensores y existe una para cada aplicación en particular. En la figura 1.9.1 se muestran sensores mecánicos con activación por leva. También se hace una descripción de sus estados en cada posición.

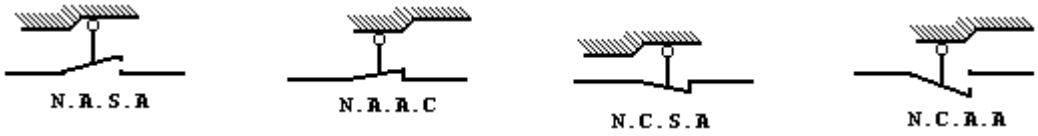


Figura 1.9.1

N.A.S.A = Significa normalmente abierto sin activar.

N.A.A.C = Significa normalmente abierto activado cerrado.

N.C.S.A = Significa normalmente cerrado sin activar.

N.C.A.A.= Significa normalmente cerrado activado abierto.

NOTA:

Aquí es importante aclarar, que los finales de carrera se dibujan en el diagrama tal como están colocados en su posición original, esto es, sin energizar el sistema y en la posición de reposo ya sean activados o desactivados.

1.9.1 Ejemplos de circuitos con finales de carrera.

EJEMPLO 1:

Se desea automatizar un taladro de banco, el cual cuenta con dos motores, uno que baja y sube y el otro que gira hacia la derecha y hacia la izquierda, (utilice motores de corriente directa), además se cuenta con un boton de inicio el cual deberá trabajar de la siguiente manera:

Al oprimir el boton de inicio deberán energizarse tanto el motor de bajar como el de giro hacia la derecha, cuando halla barrenado totalmente la pieza deberá parar ambos motores, durar un tiempo parados para vencer la inercia y cuando completen este tiempo deberán invertir la rotación (subir y girar hacia la izquierda) y cuando lleguen a la posición de reposo deberán parar.

Para resolver este problema se requieren dos finales de carrera, uno en reposo y otro al final tal como lo muestra la figura 1.9.2. La solución se encuentra en la figura 1.9.3 (circuito de control) y en la figura 1.9.4 el circuito de fuerza.

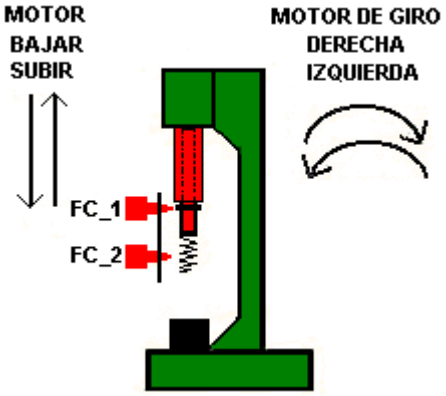


Figura 1.9.2

CIRCUITO DE CONTROL

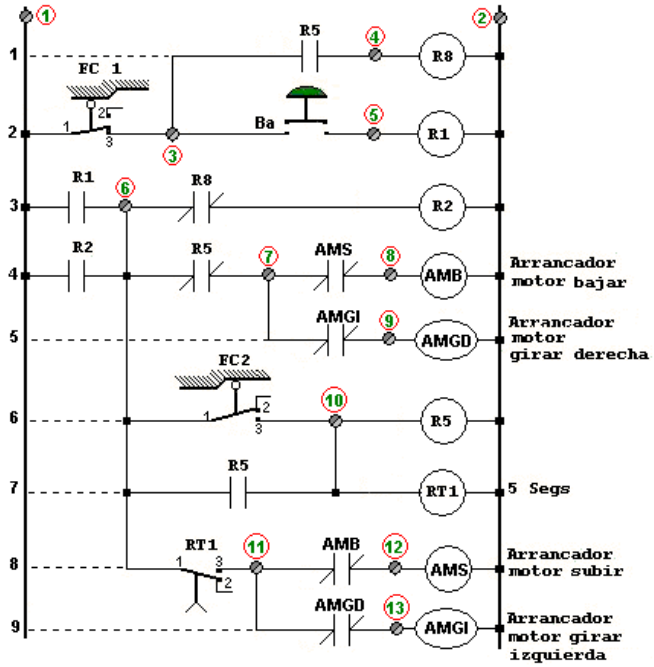


Figura 1.9.3

CIRCUITO DE FUERZA

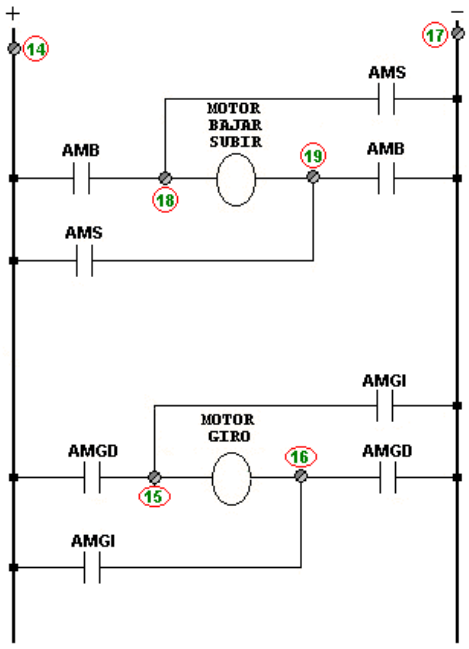
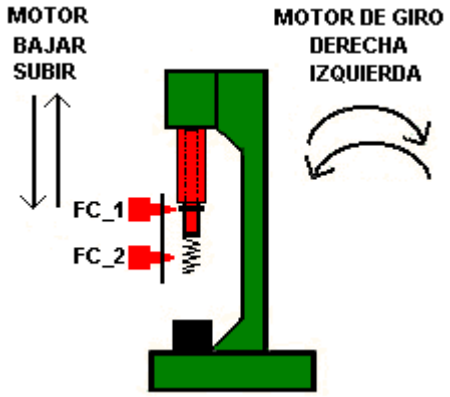


Figura 1.9.4

En el ejercicio de la página anterior se puede apreciar muy fácilmente que si el operador al dar inicio tiene la mano adentro del área de trabajo por cualquier circunstancia, el taladro bajará y puede causar un accidente, por lo que en el siguiente ejercicio se nos pide que realicemos una protección, de tal manera que, el operador tenga ocupada ambas manos, esto es deberá tener dos botones de inicio y solamente bajara el taladro si se oprimen ambos botones al mismo tiempo con una diferencia de un segundo uno del otro para asegurarnos de que el operador tenga las manos fuera del área de riesgo. Este tipo de ejercicios es obligatorio por normas donde se tienen talados pistones o prensas, además de que para que inicie el ciclo deberán activarse ambos botones al mismo tiempo de tal suerte de que si se activa un botón y después de un segundo se activa el segundo botón no baje el taladro, con ello aseguramos la integridad tanto del operador como de las herramientas de trabajo.



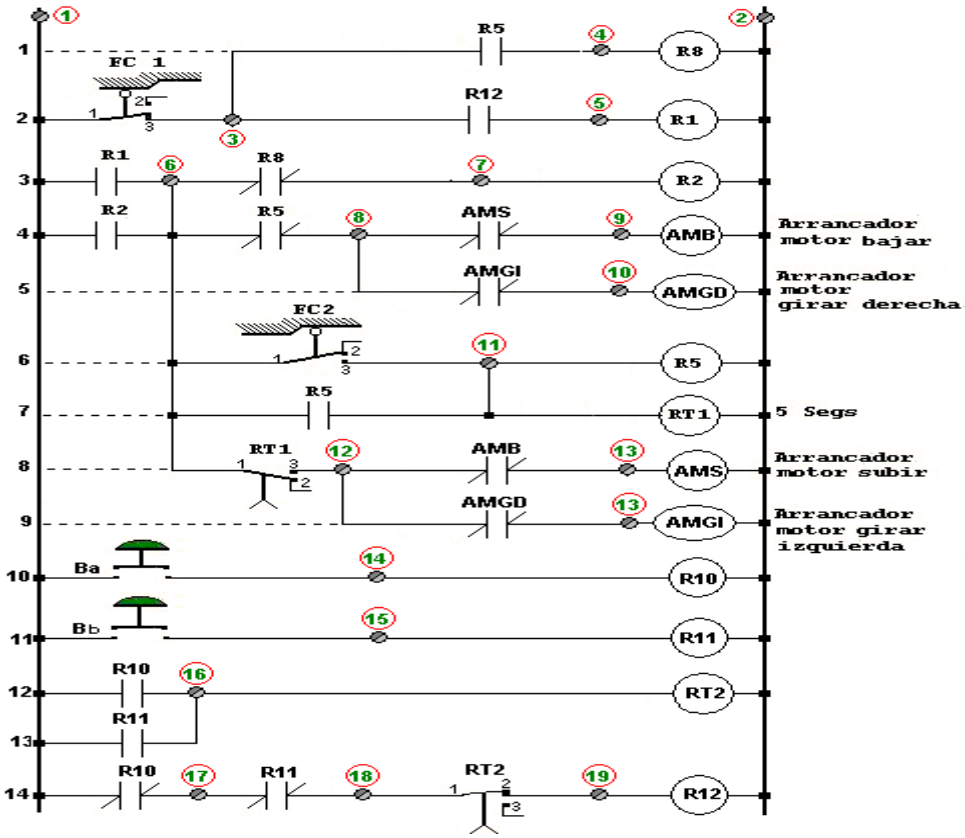


Figura 1.9.3a

EJEMPLO 2:

Automatizar el taladro del ejemplo anterior, sin embargo ahora queremos perforar una placa gruesa y se requiere que se haga en dos pasos para evitar que la broca se caliente demasiado.

Por lo tanto, se barrenará la mitad de la placa, se pararan los motores durante un tiempo, (para vencer la inercia), después invierten la rotación y al llegar a la posición de reposo se paran duran un tiempo parados y vuelven a bajar hasta perforar la placa totalmente, duran un tiempo parados y invierten la rotación hasta llegar a la posición de reposo, terminando el ciclo.

Para este sistema se requiere tan sólo un botón de inicio. (Ver figura 1.9.5)

SOLUCION:

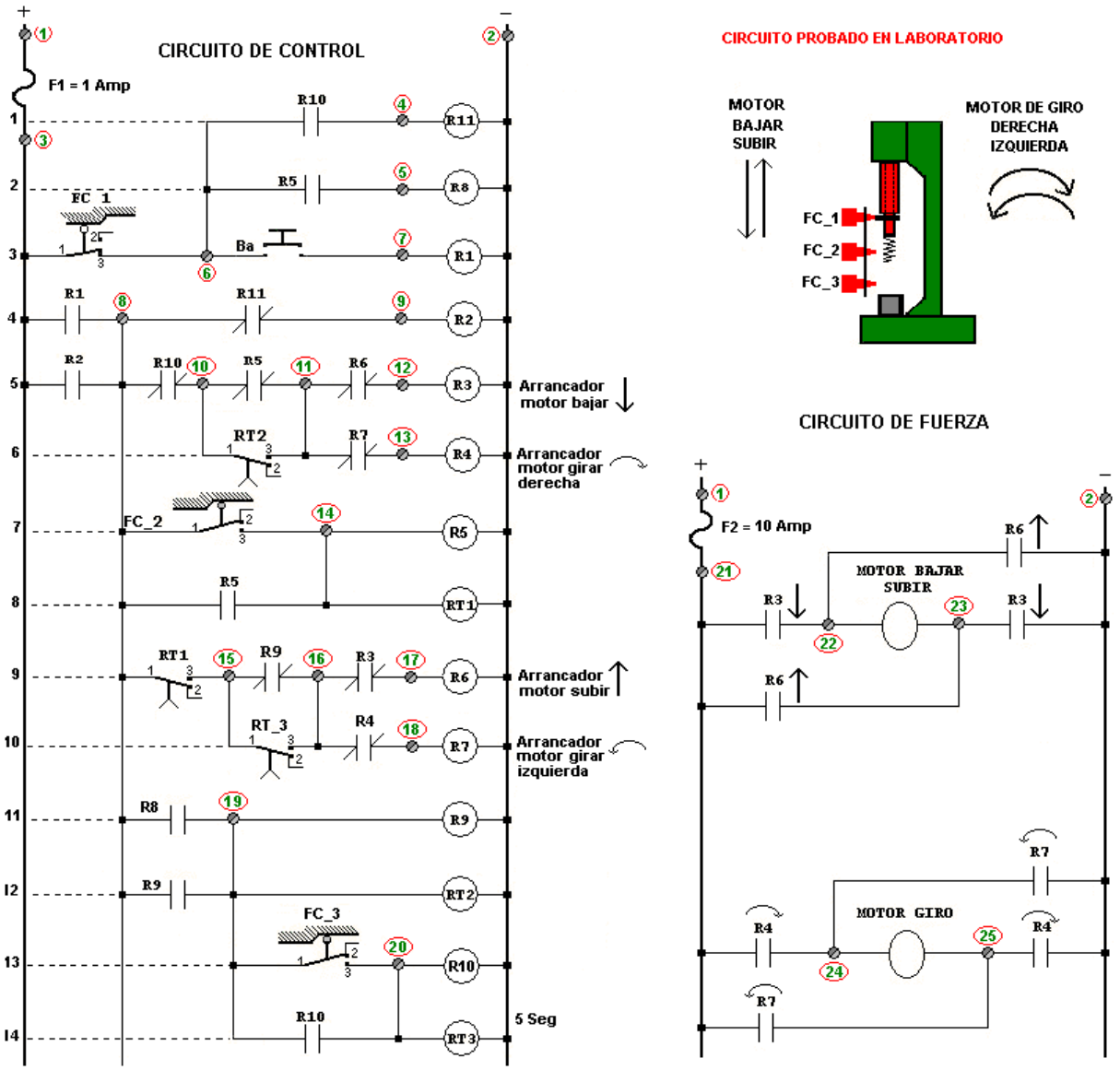


Figura 1.9.5

EJEMPLO 3:

Diseñe el circuito de control para controlar un carro de DC en el cual se requiere tener botones de avance hacia la derecha y avance hacia la izquierda.

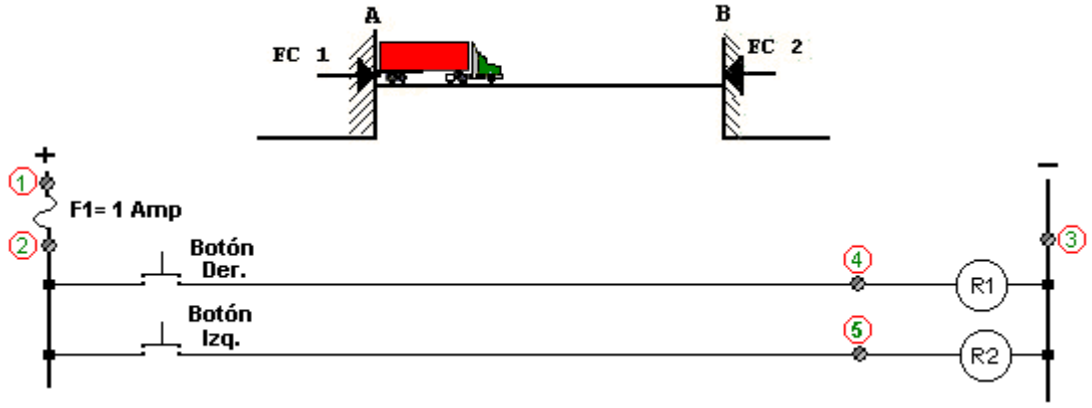


Figura 1.9.6a

Ahora observe que si se oprime el botón de avance hacia la derecha y llegar este hasta el final de su carrera, esto es, hasta el tope de su carrera, y se continúa oprimiendo dicho botón, el carro se forzaría con lo cual se puede quemar; realice las protecciones correspondientes.

Para ello, se colocaron dos finales de carrera en los extremos, verifique el circuito siguiente.

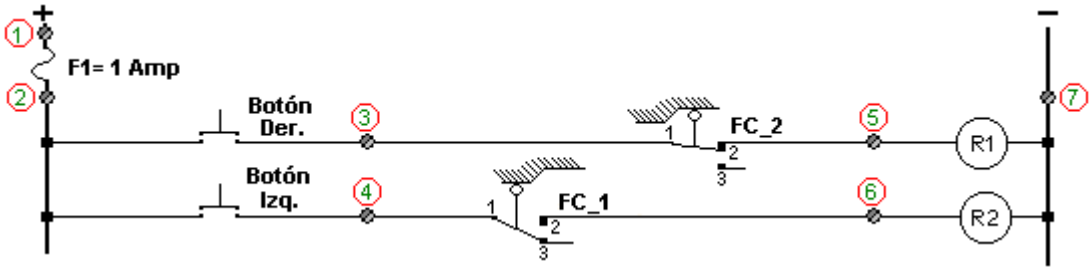


Figura 1.9.6b

Observe como en el circuito anterior se logro protegerlo cuando el operador continua oprimiendo el botón cuando el carro llego hasta su final de carrera, pero ahora se tiene el problema de que si el carro se encuentra en la mitad del camino y se oprimen ambos botones se produce un corto circuito.

Realice las protecciones correspondientes.

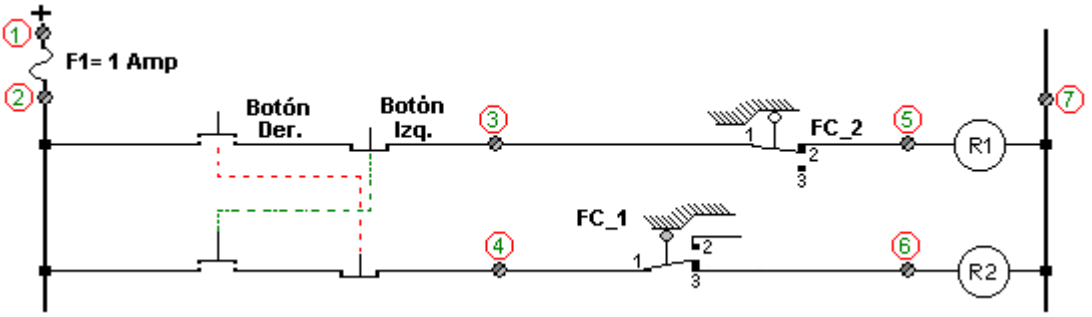


Figura 1.9.6c

Ahora se desea proteger el circuito colocando interlock eléctricos.

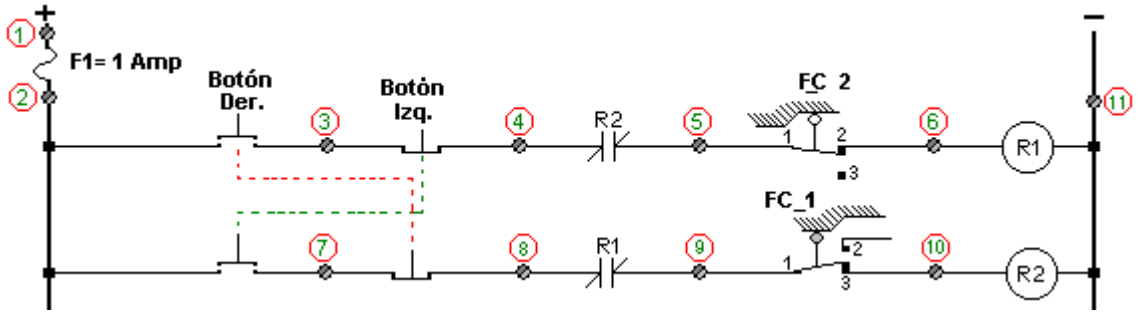


Figura 1.9.6d

Ahora se desea que al dar un pulso al botón derecho avance y pare al llegar al final de su carrera, lo mismo aplica para el botón de la izquierda.

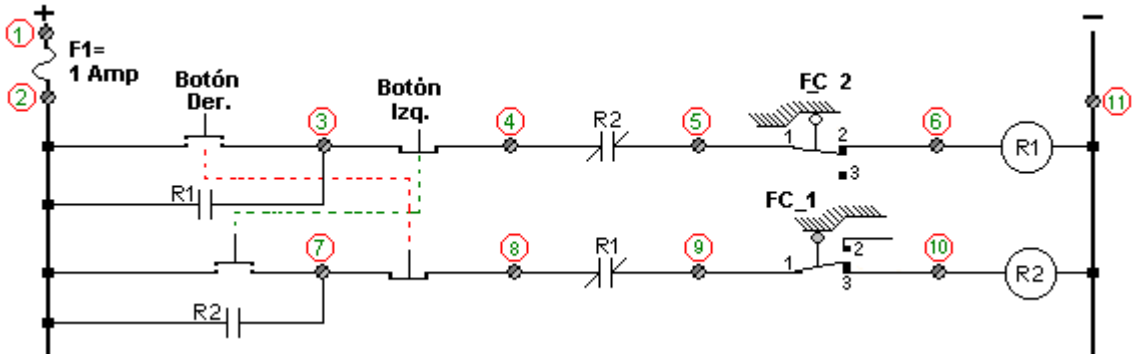


Figura 1.9.6e

Por último se desea tener un solo botón de avance, de tal manera que se de un pulso de inicio avance y al llegar al final de su carrera, pare el motor, dure un tiempo parado e invierta la rotación parando siempre en la posición de reposo. Considere que para dar inicio de ciclo se requiere que el carro este en la posición de reposo, esto significa que el final de carrera FC_1 esté activado.

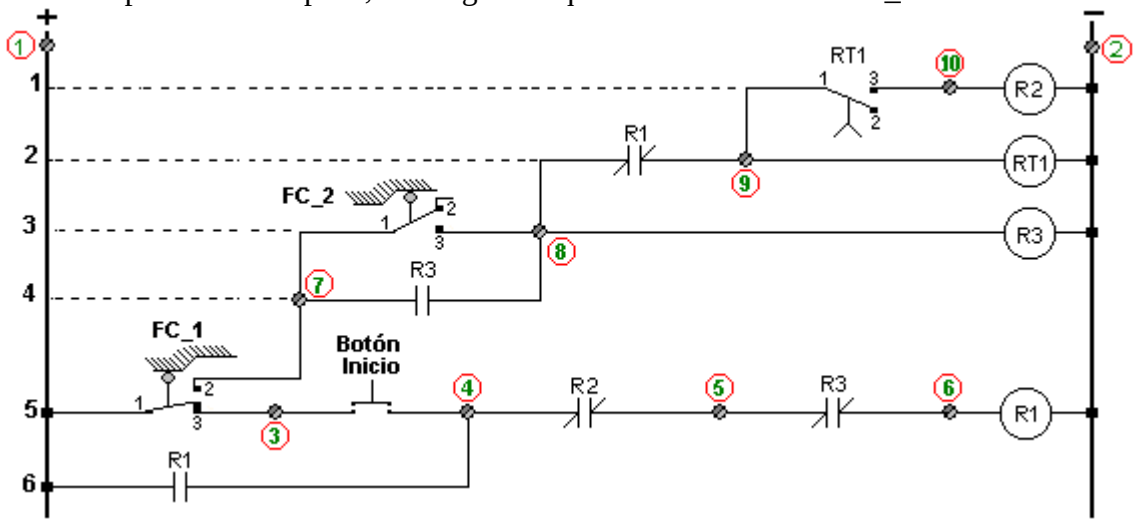


Figura 1.9.6f

EJEMPLO 4:

Se tiene un carro con un motor de corriente directa y se desea transportar material de" **A** hacia **B**" al llegar al punto **B** deberá pararse, durar un tiempo parado y regresar, esto es, invertir el sentido de giro del motor y al llegar al punto **A** pararse durar un tiempo y salir de nuevo. Además se desea poner un selector de tres posiciones, de tal manera que en la posición 1 realice un ciclo, en la posición 2 realice tres ciclos y en la posición tres trabaje en automático.

También se desea poner un boton de paro, si éste se oprime cuando el carro va hacia delante deberá continuar hacia adelante y terminar el ciclo y cuando llegue a la posición de reposo deberá pararse y quedar todo en condiciones de iniciar de nuevo. Si el carro viene de regreso, al llegar a la posición de reposo deberá terminar su ciclo.
La solución se encuentra en la figura 1.9.7

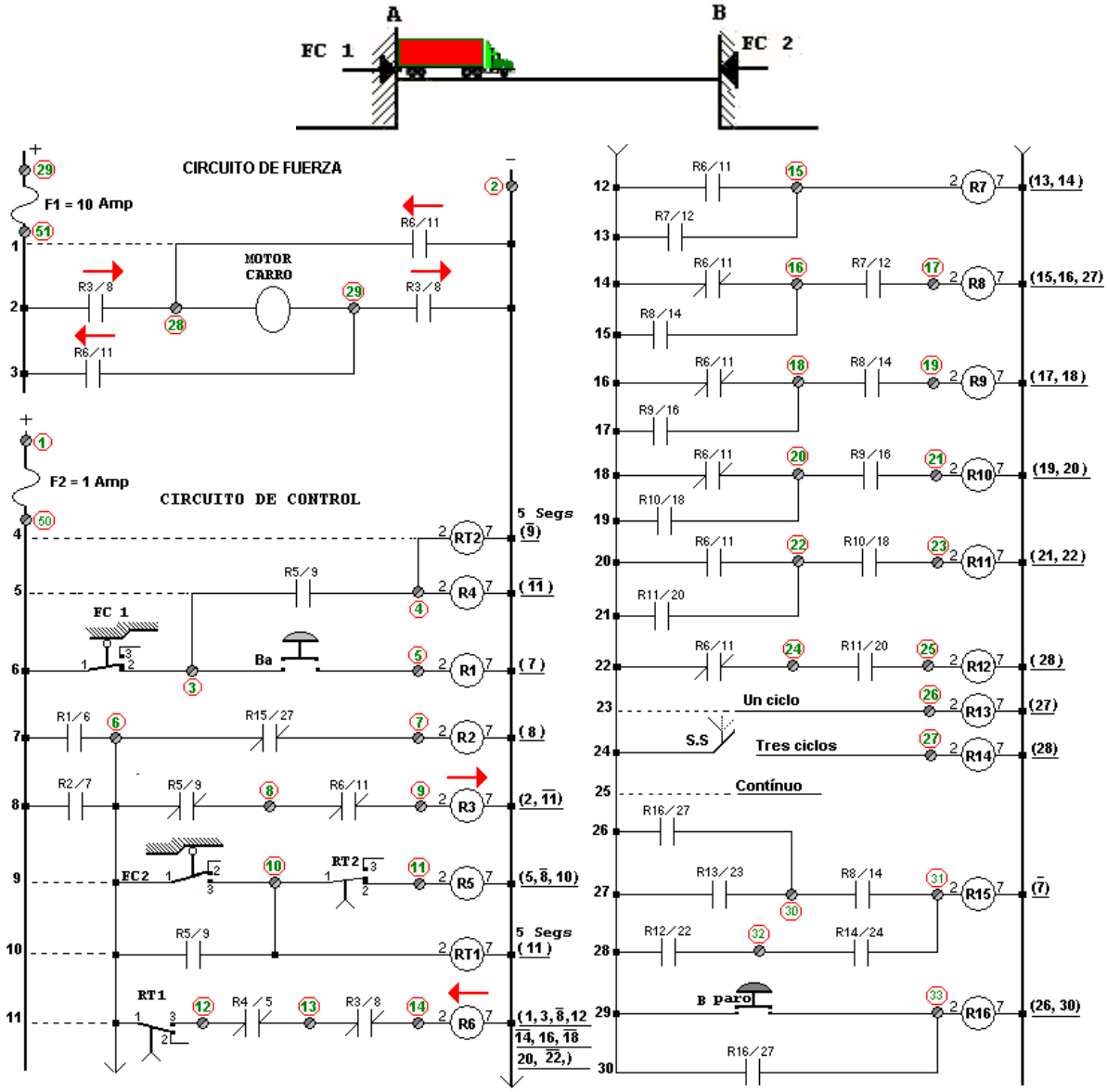


Figura 1.9.7

EJEMPLO 5: Diseñe el circuito de control para una maquina que trabaja con la secuencia descrita en la figura 1.9.8

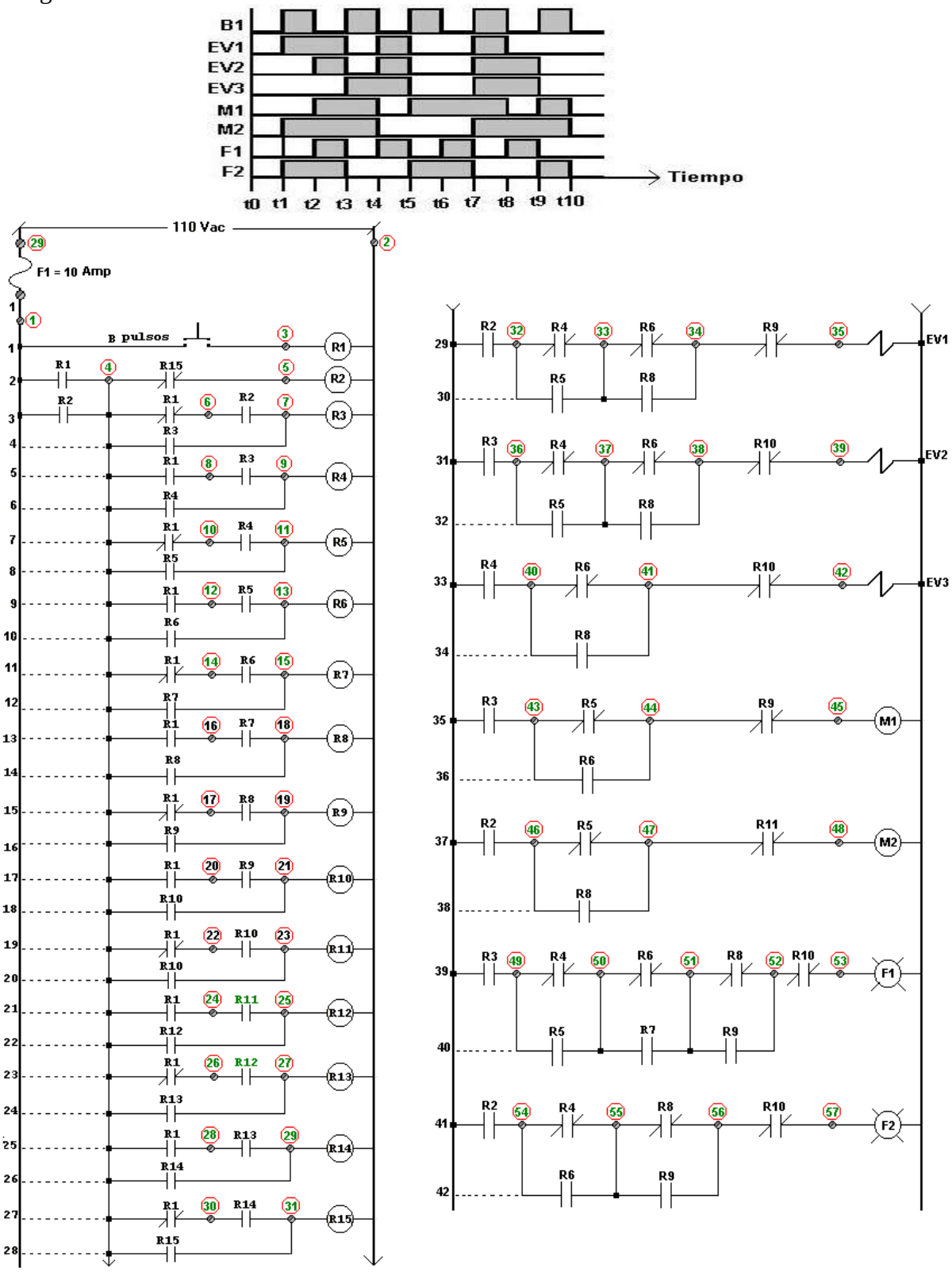


Figura 1.9.8

EJEMPLO 6:

1.- Se tienen tres botones normalmente abiertos;

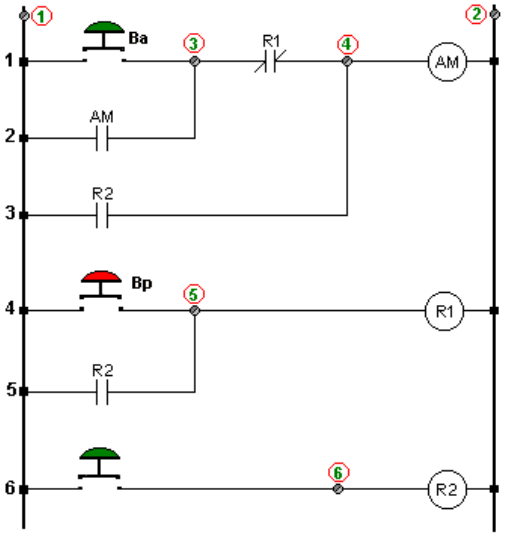
Y se requiere que:

a.- Al oprimir el botón de arranque, se energice un motor M1

b.- Al oprimir el botón de paro se detenga el motor M1

c.- Al oprimir el botón de ajuste (jogueo) si el motor esta parado que se energice y al soltarlo se pare, esto es para dar pulsos; además, si el motor esta energizado y si oprimimos el botón de ajuste (jogueo) que el motor continúe energizado y al soltarlo se pare.

SOLUCIÓN 1



SOLUCIÓN 2

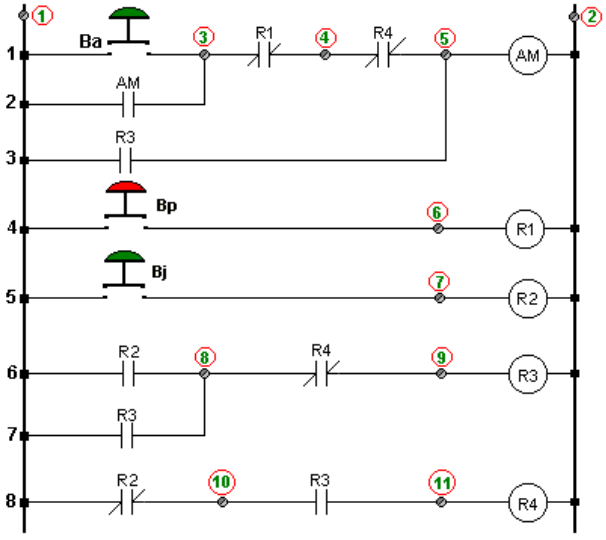


Figura 1.9.9

EJEMPLO 7:

Se tiene una cisterna y un depósito, además se tienen dos bombas B1 y B2, y se desea bombear agua desde la cisterna al depósito de tal manera que las bombas trabajen en forma alternada, esto es, que trabaje la bomba B1 y en seguida la bomba B2 de acuerdo a los niveles presentados en la figura siguiente.

Condiciones de los niveles para trabajar las bombas;

Las bombas estarán en condiciones de trabajar si el liquido de la cisterna llega al nivel B y parar si llega al nivel A, además deberán cumplirse las condiciones que el nivel del tanque deberá estar entre los niveles de C y D, esto es, si el nivel del tanque llega al punto D deberá parar y arrancar hasta que llegue al punto C.

Por último, se desea una estación de botones, de arranque y paro del sistema.
(Ver solución en figura 1.9.10)

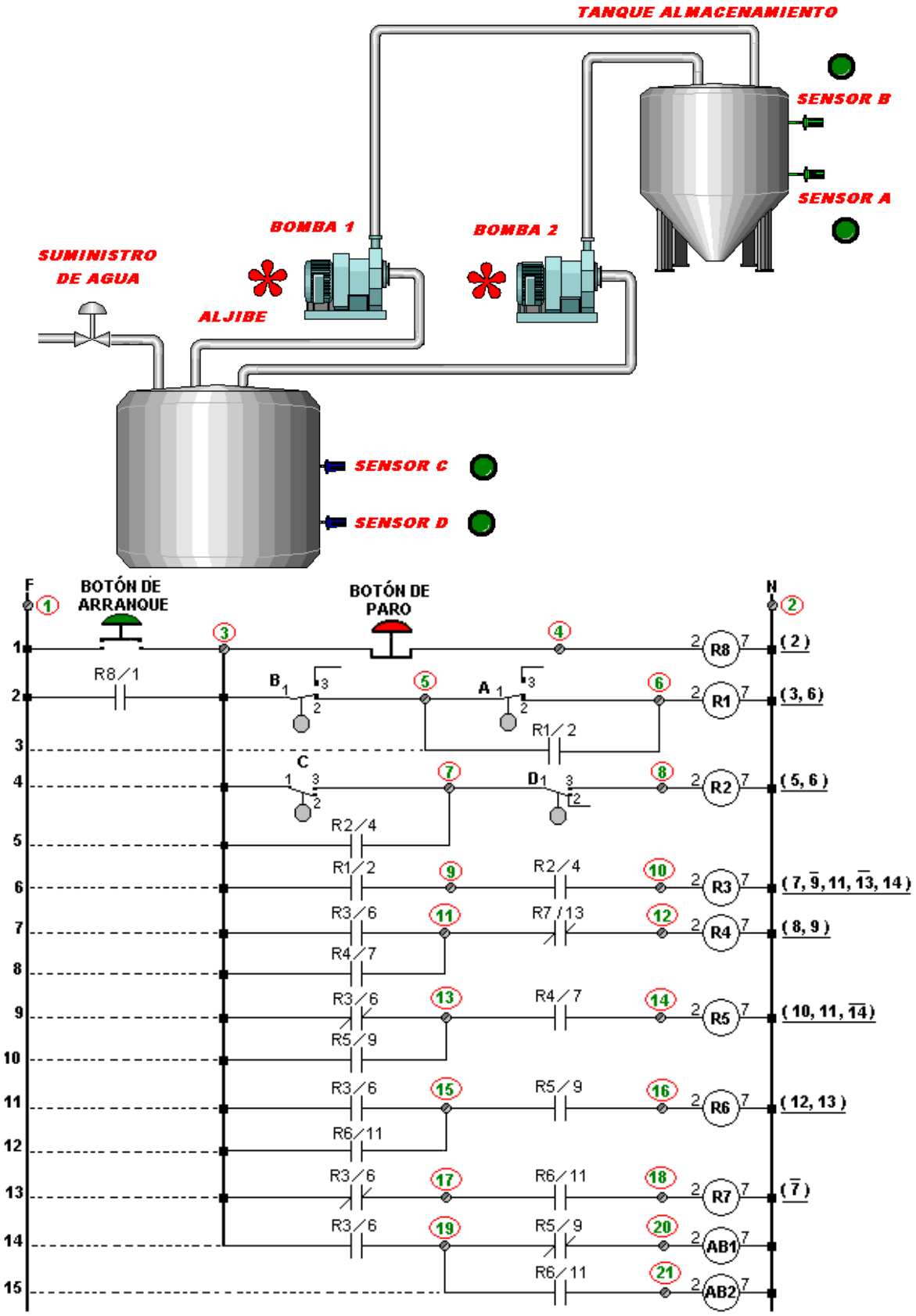

TANQUE ALMACENAMIENTO
SENSOR B
SENSOR A
BOMBA 1
BOMBA 2
SUMINISTRO DE AGUA
ALJIBE
SENSOR C
SENSOR D
F
N
BOTÓN DE ARRANQUE
BOTÓN DE PARO
R8/1
R1/2
R2/4
R3/6
R4/7
R5/9
R6/11
R7/13
R8
R1
R2
R3
R4
R5
R6
R7
AB1
AB2
(2)
(3, 6)
(5, 6)
(7, 9, 11, 13, 14)
(8, 9)
(10, 11, 14)
(12, 13)
(7)

Figura 1.9.10

EJEMPLO 8:

Diseñe el circuito de control para realizar la mezcla de dos productos energizando las electroválvulas 1 y 2 cuando el tanque este lleno (detectado por el sensor 1) se deberá desenergizar dichas electroválvulas entrando el motor para realizar dicha mezcla durante un tiempo de 5 minutos, al terminar dicho tiempo, se desenergiza dicho motor entrando un tiempo de 5 segundos para estabilizar el sistema, posteriormente deberá entrar la electroválvula 3 para realizar la descarga, cuando el sensor 2 detecta el nivel bajo se deberá cerrar la electroválvula tres (desenergizandola) y deberá proceder a realizar un nuevo batch.

El sistema deberá contar con un botón de inicio y un botón de paro con la salvedad de que al oprimir el botón de paro, deberá continuar con el ciclo hasta terminarlo y pararse. (Ver solución en figura 1.9.11)

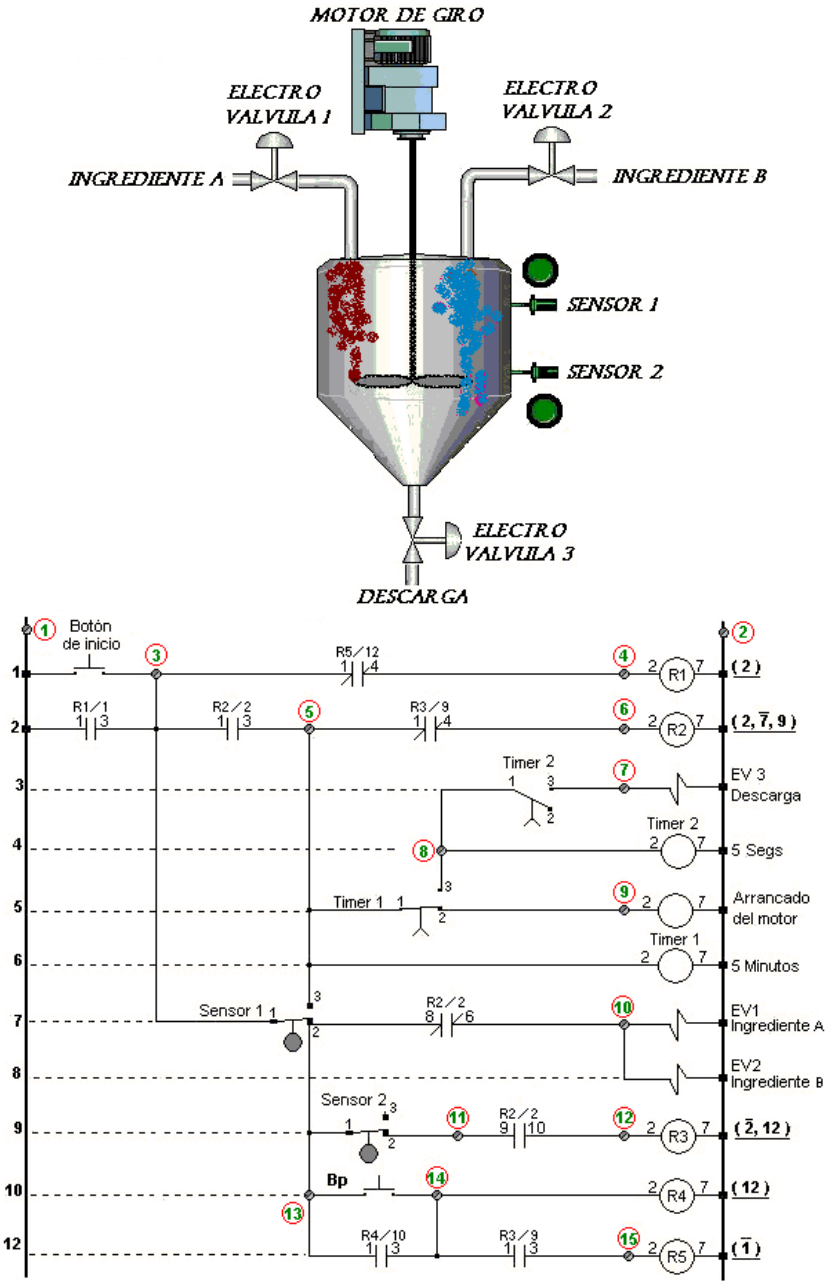


Figura 1.9.11

EJEMPLO 9:

Diseñe el circuito de control para realizar la mezcla de dos productos energizando la electroválvula 1 cuando el tanque llegue al nivel 2 (detectado por el sensor 2) se deberá desenergizar la electroválvula 1 y energizar la electroválvula 2 entrando el ingrediente B, cuando llega al nivel 1, se desenergiza la electroválvula 2 y se energiza el motor de alta velocidad para realizar la mezcla durante un tiempo de 5 minutos, al terminar dicho tiempo, se desenergiza el motor de alta velocidad entrando a una velocidad más baja con un tiempo de 2 segundos para estabilizar el sistema, posteriormente deberá entrar la electroválvula 3 para realizar la descarga continuando el motor de baja velocidad, cuando el sensor 2 detecta el nivel bajo se deberá cerrar la electroválvula tres (desenergizandola) y dura un tiempo de 3 segundos y deberá proceder a realizar un nuevo batch.

El sistema deberá contar con un botón de inicio y un botón de paro con la salvedad de que al oprimir el botón de paro, deberá continuar con el ciclo hasta terminar el batch y pararse.

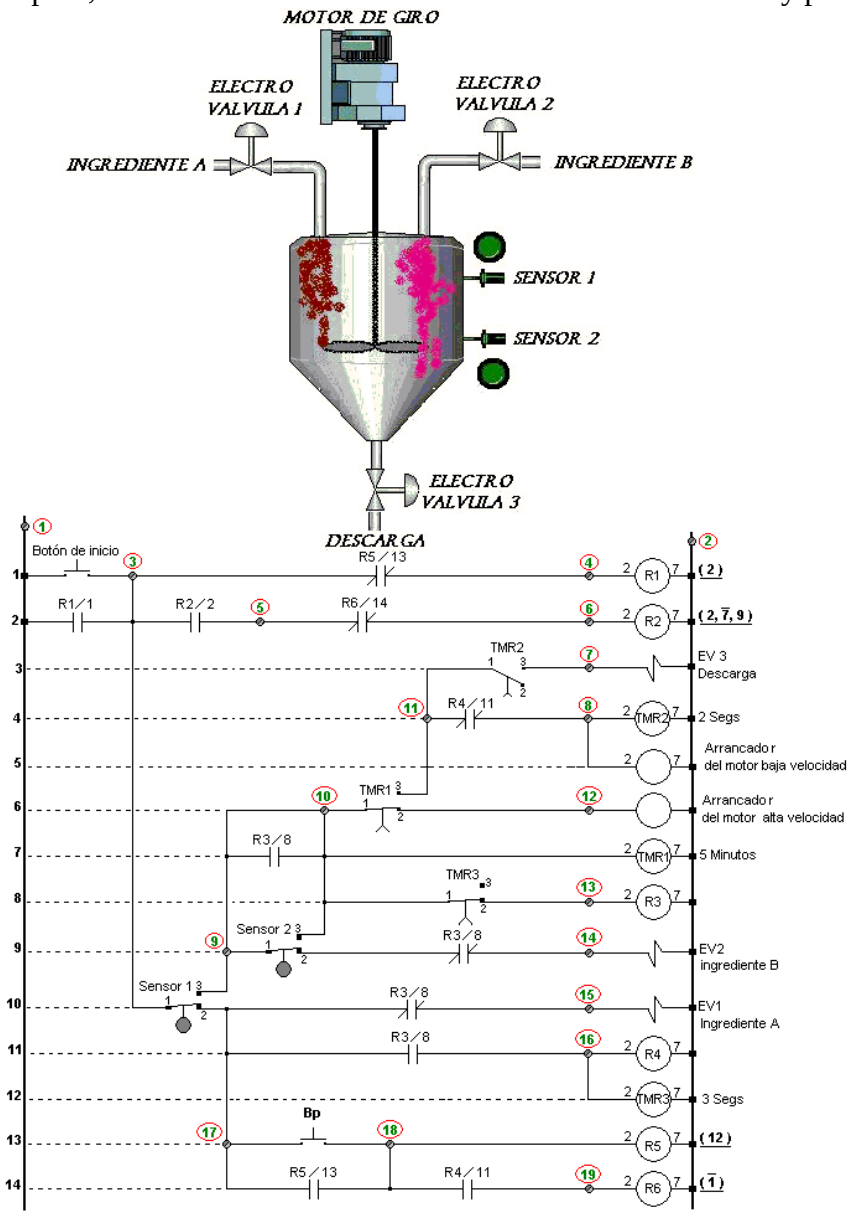
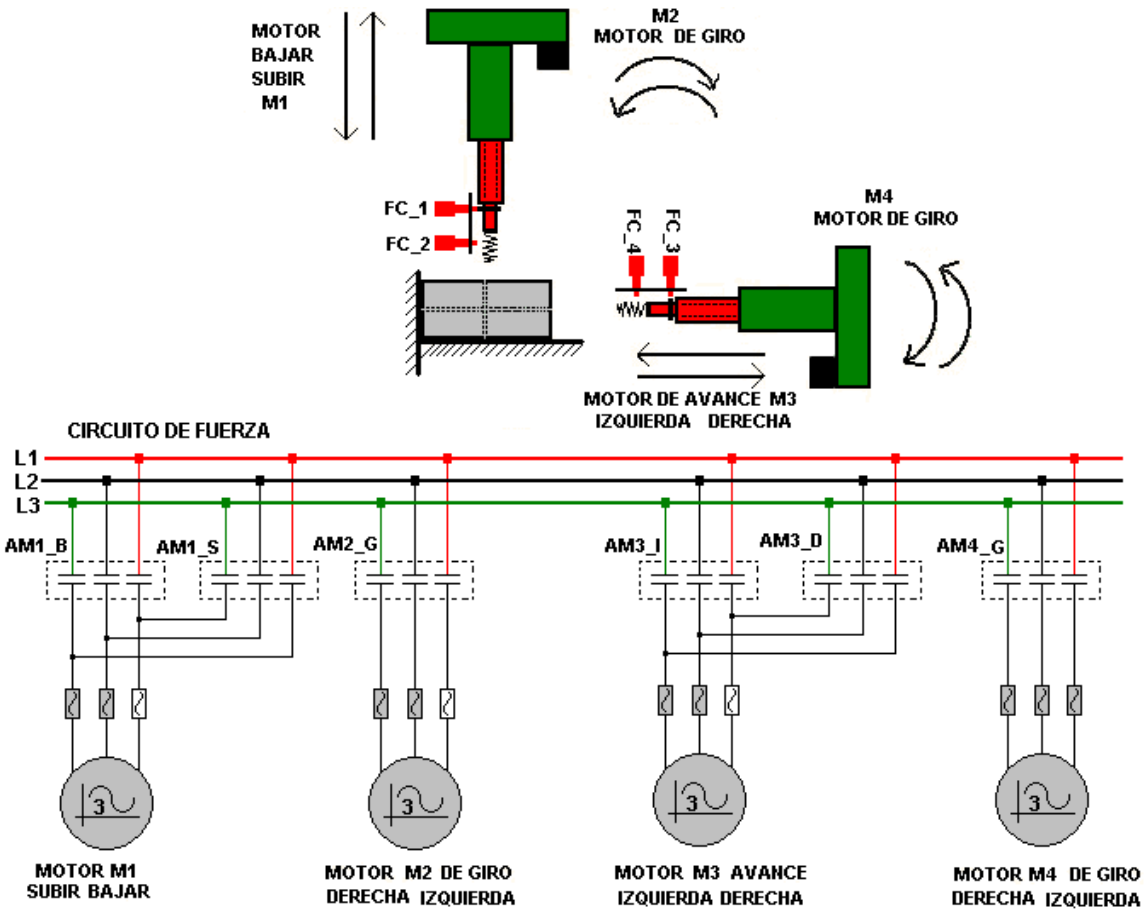


Figura 1.9.12

EJEMPLO 10:

Se desea barrenar una pieza de madera, tal como lo muestra la figura siguiente;



Se cuenta con un botón de inicio, de tal manera que al oprimir el botón de inicio, se energice el motor M1 y M2 el motor M1 sirve para bajar y el motor M2 sirve para girar la broca, al llegar al final de su carrera, deberá parar el motor de bajar y dar un tiempo para que pierda la inercia de bajar (el motor de giro continúa girando), cuando termina el tiempo deberá entrar el motor M1 pero ahora deberá subir (invertir la rotación) y al llegar a la posición de reposo se paran los dos motores M1 y M2 e inmediatamente deberá entrar el M3 (avance de taladro) y el motor M4 de giro de broca, al llegar a final de su carrera, (cuando ya termino de barrenar deberá parar el motor de avance y el de giro deberá continuar, al mismo tiempo deberá entrar un tiempo y al terminar este tiempo deberá entrar el motor M3 de regreso (invertir la rotación) y al llegar a la posición de reposo, deberá parar ambos motores terminando con ello el ciclo.

NOTA:

Para poder dar inicio de ciclo se requiere que ambos taladros por seguridad se encuentren en reposo.

NOTA:

Como puede observar se tienen dos motores reversibles M1 y M3, realice el circuito con protecciones de interlock eléctricas y mecánicas.

(Ver solución en figura 1.9.13)

SOLUCION:

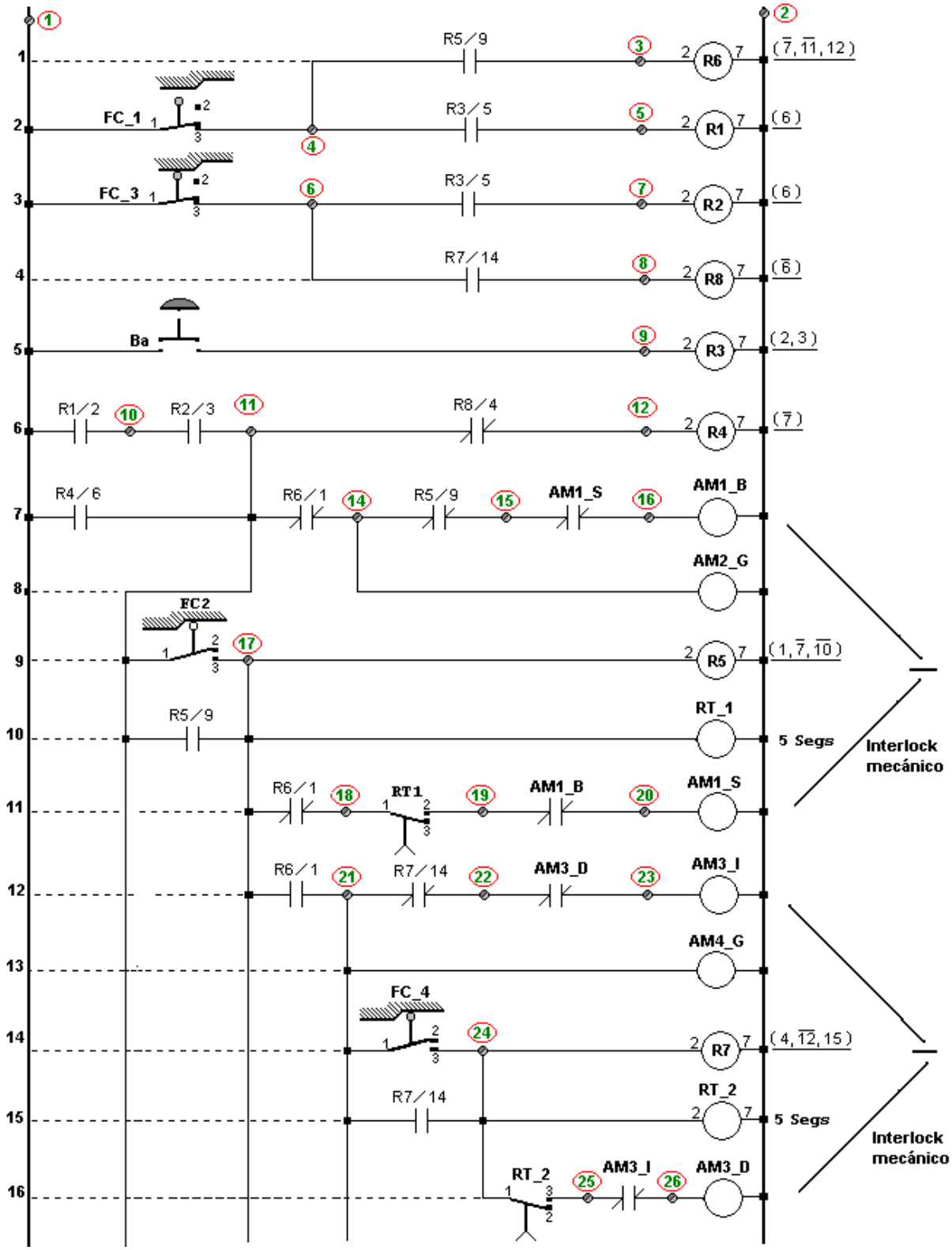
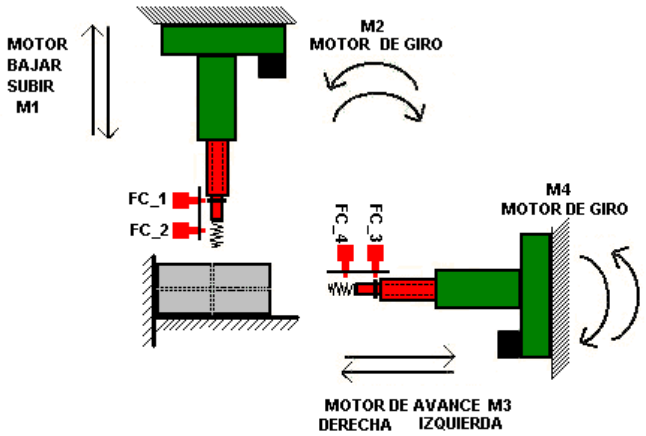
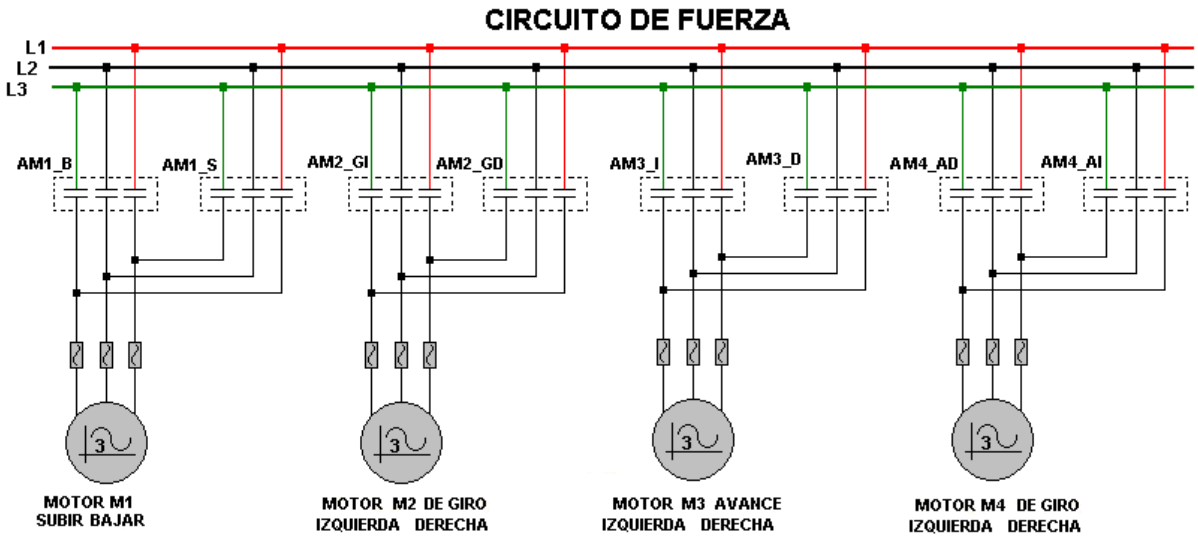


Figura 1.9.13

EJEMPLO 11:

Se desea barrenar una pieza de madera, tal como lo muestra la figura siguiente;





Realice el circuito de control del problema anterior (No #3), pero ahora se desea que los motores de giro también paren e inviertan la rotación.

NOTA: Para poder dar inicio de ciclo se requiere que ambos taladros por seguridad se encuentren en reposo.

NOTA: Como puede observar se tienen dos motores reversibles M1 y M3, realice el circuito con protecciones de interlock eléctricas y mecánicas.

SOLUCIÓN CIRCUITO DE CONTROL:

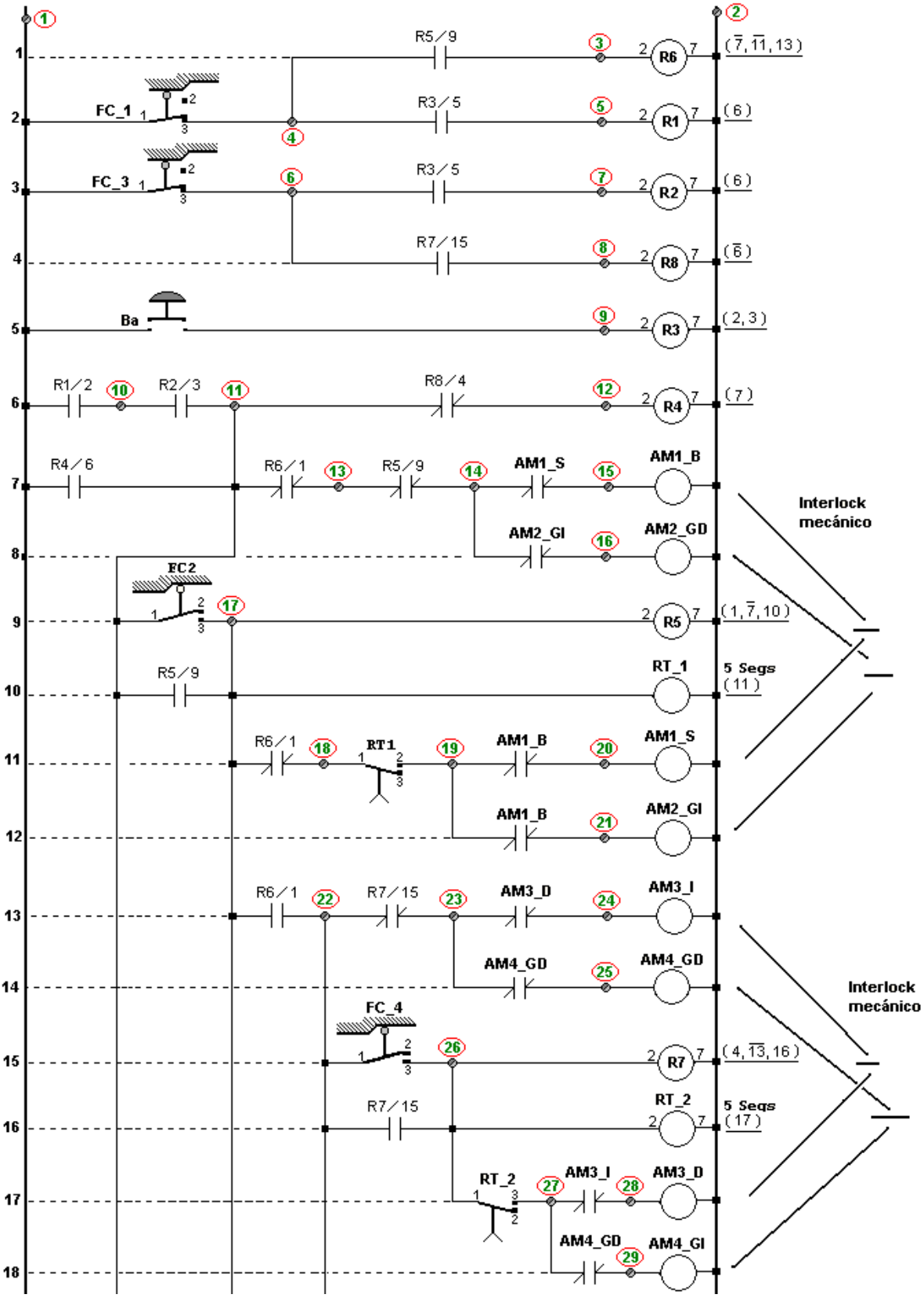


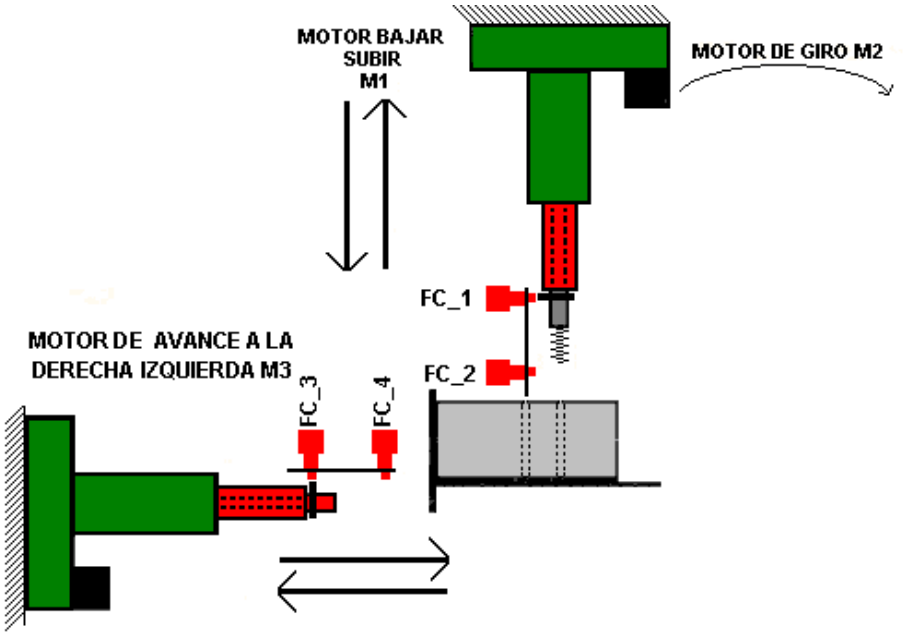
Figura 1.9.14

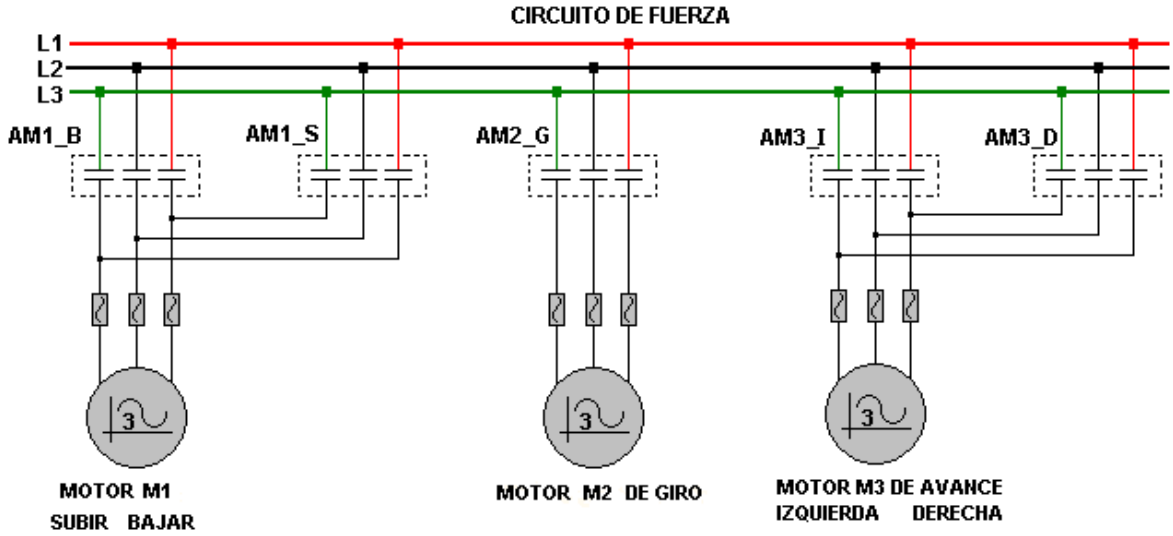
EJEMPLO 12:

Se desea barrenar una pieza de madera, tal como lo muestra la figura siguiente;

Se cuenta con un botón de inicio, de tal manera que al oprimir el botón de inicio, se energice el motor M1 y M2 el motor M1 sirve para bajar y el motor M2 sirve para girar la broca, al llegar al final de su carrera, deberá parar el motor de bajar y dar un tiempo para que pierda la inercia de bajar (el motor de giro continúa girando), cuando termina el tiempo deberá entrar el motor M1 pero ahora deberá subir (invertir la rotación) y al llegar a la posición de reposo se paran los dos motores M1 y M2 e inmediatamente deberá entrar el M3 (avance de la pieza), al llegar a final de su carrera, deberá entrar de nuevo el taladro para realizar el segundo barreno y al llegar arriba deberá regresar el motor de avance terminando el ciclo

<u>NOTA</u>: Para poder dar inicio de ciclo se requiere que el sistema este en reposo por seguridad.





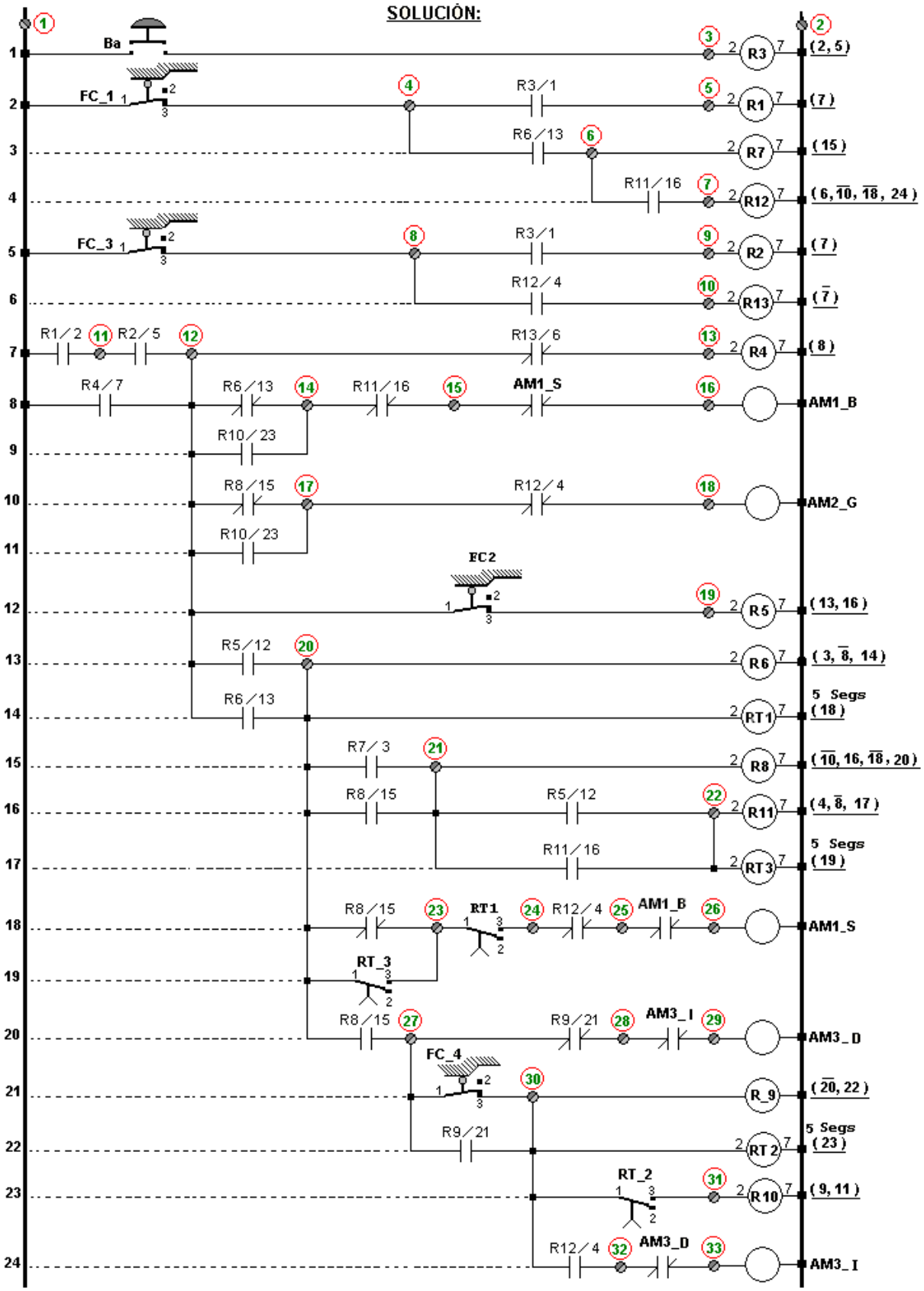

SOLUCIÓN:
Ba
FC_1
FC_3
FC2
FC_4
R3/1
R6/13
R11/16
R12/4
R1/2
R2/5
R13/6
R4/7
R10/23
R8/15
R5/12
R7/3
R9/21
AM1_S
AM1_B
AM2_G
AM3_I
AM3_D
RT1
RT_3
RT_2
R3
R1
R7
R12
R2
R13
R4
R5
R6
RT1
R8
R11
RT3
R_9
RT2
R10
(2, 5)
(7)
(15)
(6, 10, 18, 24)
(7)
(7)
(8)
(13, 16)
(3, 8, 14)
5 Segs (18)
(10, 16, 18, 20)
(4, 8, 17)
5 Segs (19)
(20, 22)
5 Segs (23)
(9, 11)

Figura 1.9.15

EJEMPLO 13:

Se desea barrenar una pieza de madera, tal como lo muestra la figura siguiente;

Se cuenta con un botón de inicio, de tal manera que al oprimir el mismo, se energice el motor M1 el cual sirve para bajar y girar la broca, al llegar al final de su carrera, deberá parar el motor de bajar y dar un tiempo para que pierda la inercia de bajar , cuando termina el tiempo deberá entrar el motor M1 pero ahora deberá subir (invertir la rotación) y al llegar a la posición de reposo se para e inmediatamente deberá entrar el M2, al llegar a final de su carrera, deberá parar durar un tiempo (para perder la inercia) y después de un tiempo invierte la rotación (regresando) y al llegar a la posición de reposo deberá entrar el motor 3 (M3), al llegar al final de su carrera deberá parar durar un tiempo y después invertir la rotación regresar y al llegar a la posición de reposo terminar el ciclo quedando listo para iniciar un ciclo de nuevo.

NOTA: Para poder dar inicio de ciclo se requiere que el sistema este en reposo por seguridad.

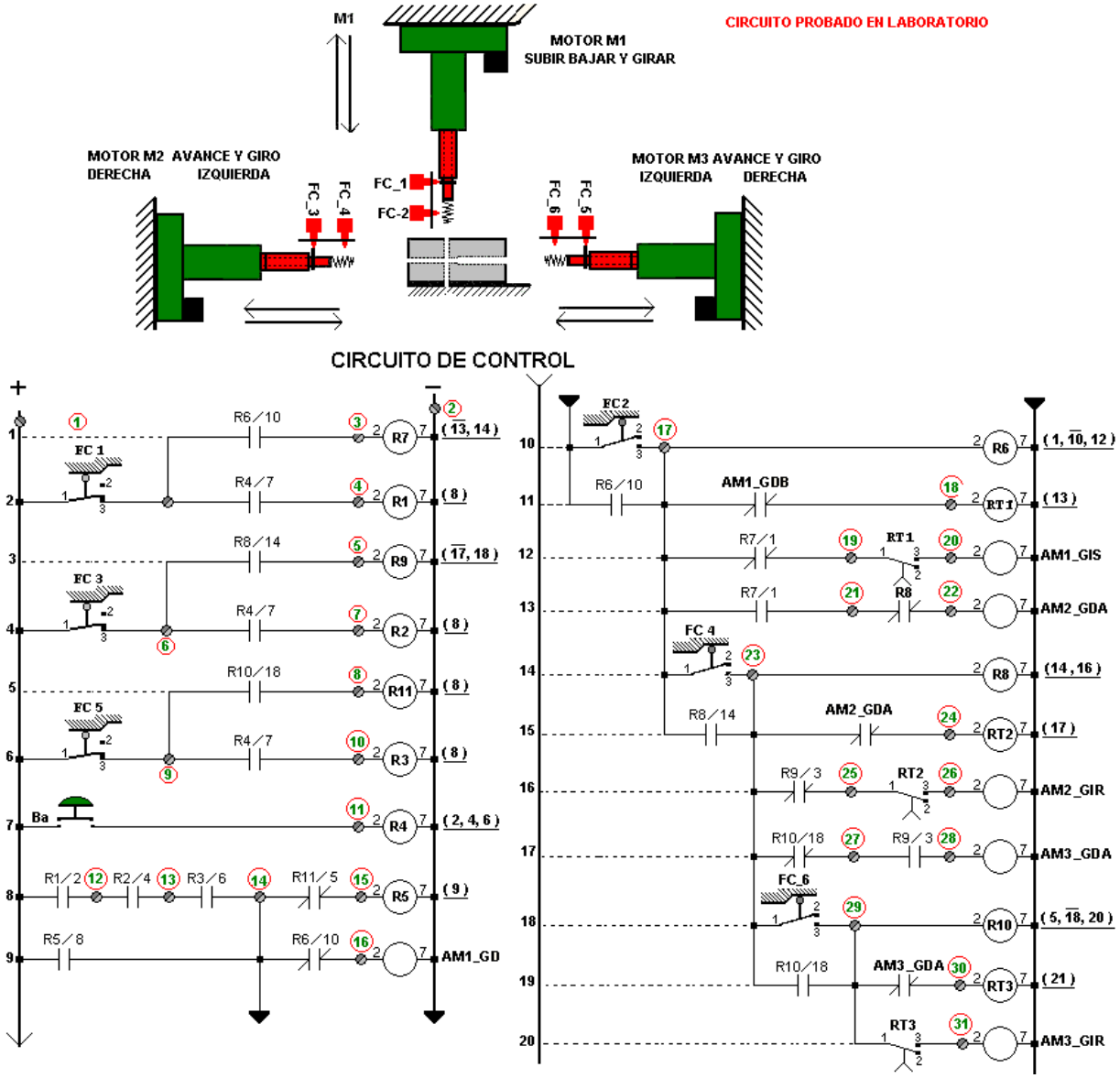
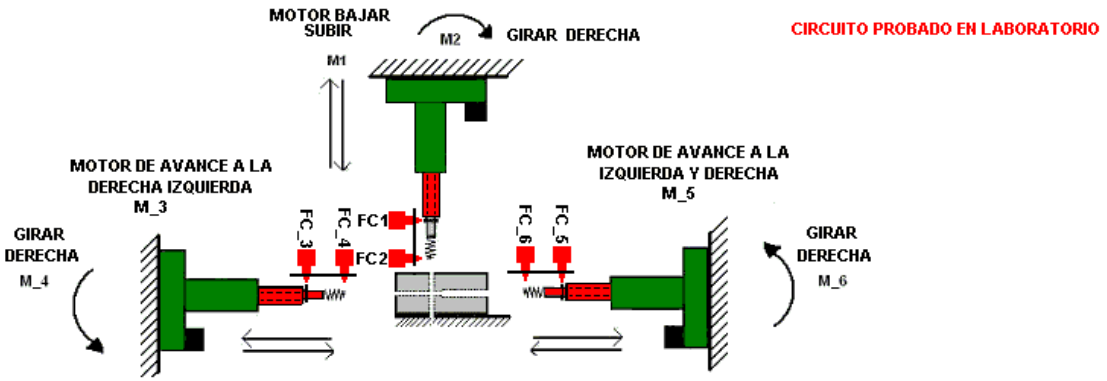


Figura 1.9.16

EJEMPLO 14: Realice el circuito anterior, pero ahora se desea poner motores independientes para el movimiento y el giro, pero ahora se desea que los motores de giro no inviertan la rotación y que no paren al llegar a su final de carrera sino que se paren hasta que lleguen hasta su posición de reposo.

NOTA: Para dar inicio se requiere que el sistema esté en reposo por seguridad.



CIRCUITO DE CONTROL

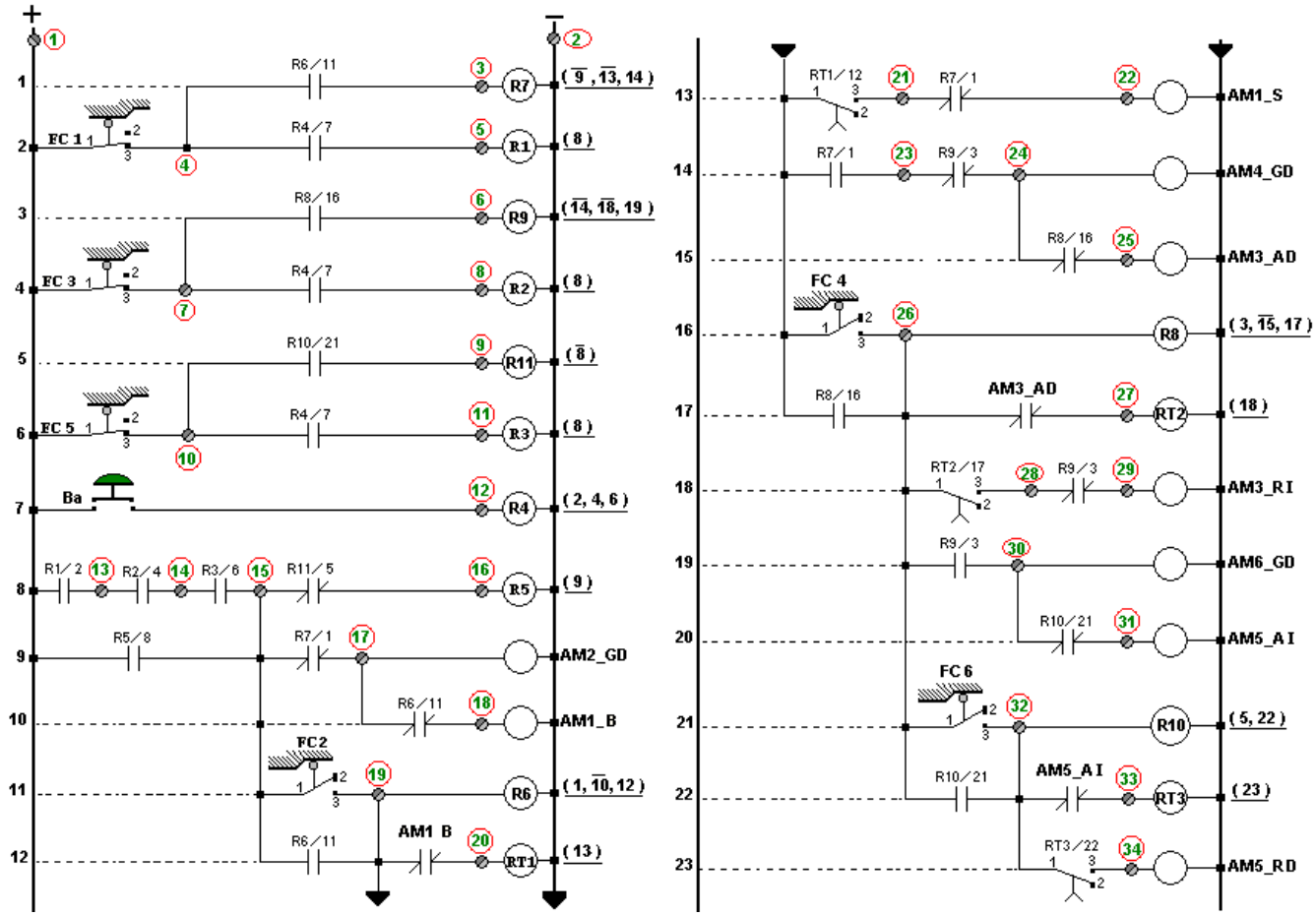


Figura 1.9.17

EJEMPLO 15:

Realice el circuito control anterior, pero ahora se desea que al llegar el motor 5 a la posición de reposo, que pare, dure un tiempo para cambiar de pieza y reinicie el ciclo en forma automática. Para parar el sistema, coloque un botón de paro, el cual parará al terminar el ciclo.

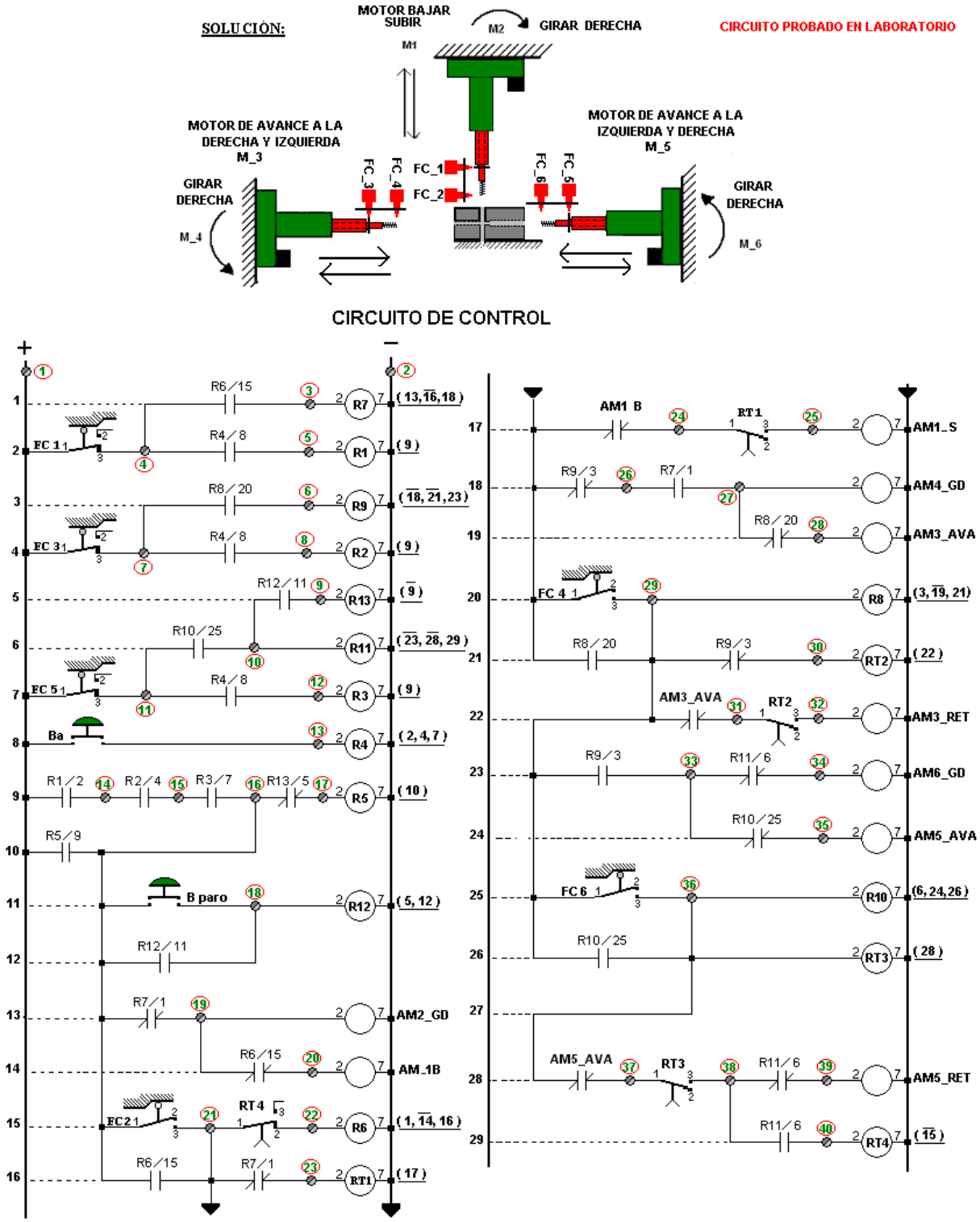


Figura 1.9.18

EJEMPLO 16:

Realice el circuito de control en el que se desea que el motor M1 baje y sujete la pieza, parando el motor M1, posteriormente, entre el motor M3 y el motor M4 avanzando hacia la derecha y realizando el barreno correspondiente, al llegar al final de su carrera para el motor de avance dura un tiempo e invierte la rotación, al llegar a la posición de reposo paran ambos motores y entran los motores M5 y M6 avanzando y girando hacia la derecha, al llegar al final de su carrera para el motor de avance, dura un tiempo e invierte la rotación, al llegar a la posición de reposo paran ambos motores entrando el motor 1 invirtiendo su rotación, al llegar a la posición de reposo para el motor M1 dura un tiempo y reinicia el ciclo. Para parar, se requiere un botón de paro, el cual al oprimirlo deberá parar al terminar el ciclo.

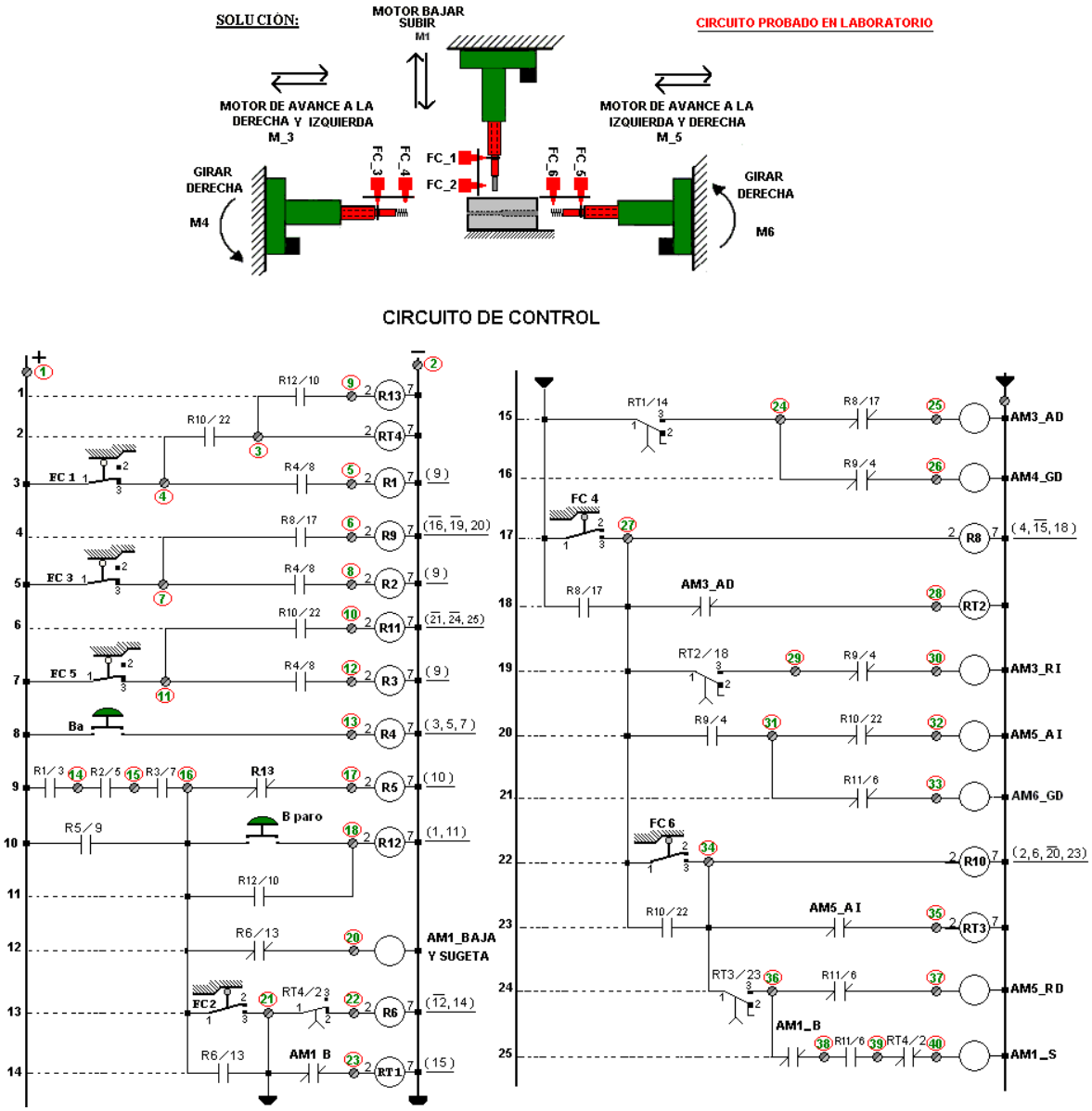


Figura 1.9.19

CAPÍTULO 2 CONTROL NEUMÁTICO

Introducción:

En la actualidad, la necesidad de automatizar la producción no es tarea únicamente de las grandes empresas, sino que también incumbe a las pequeñas industrias. Incluso la industria artesanal se ve obligada a desarrollar métodos de producción que excluyan el trabajo manual y no dependan de la habilidad del trabajador. La fuerza muscular y la habilidad manual deben sustituirse por la fuerza y precisión mecánica. La fuerza neumática puede realizar muchas funciones mejor y más rápidamente, de forma más regular y sobre todo durante más tiempo sin sufrir los efectos de la fatiga.

Comparando el trabajo humano con el de un elemento neumático, se comprueba la inferioridad del primero en lo referente a capacidad de trabajo.

No obstante, sustituir actividades manuales por dispositivos mecánicos y neumáticos, sólo es un paso dentro del proceso de automatización de la producción industrial.

Este paso está encaminado, al igual que otros muchos, a obtener el máximo provecho con un costo mínimo. La utilización de la máquina adecuada en cada caso será la forma de evitar la adquisición de costosos equipos que encarezcan el producto de forma desproporcionada, pudiéndose dar el caso de que una máquina especial construida con elementos que se producen en serie y que se adaptan exactamente a las necesidades del proceso de fabricación, resulten más económicos que una máquina estándar. Otro factor importante que se debe tomar en cuanta es la escasez de personal para ciertos tipos de trabajos. Visto a largo plazo, se advierte la tendencia regresiva en el número de empleados de las industrias que realizan trabajos muy repetitivos, no solamente es debido a la creciente automatización, sino a que en un futuro próximo no se encontrará personal para ciertos tipos de trabajos.

La energía neumática no es utilizable en todos los casos de automatización. Las posibilidades técnicas de la neumática están sometidas a ciertas limitaciones en lo que se refiere a fuerza, espacio, tiempo (precisión) y velocidad en el proceso de la producción. Esta tecnología tiene su ventaja más importante en la flexibilidad y variedad de aplicaciones en casi todas las ramas de la producción industrial.

La automatización de dispositivos, maquinaria y procesos industriales aplicando la técnica neumática, ha sido posible debido a la existencia de una gran variedad de elementos de trabajo, mando y regulación, que permiten una construcción económica, sencilla y confiable.

2.1- Que es automatizar

Es liberar al hombre de procesos repetitivos que requieran poco ó ningún esfuerzo mental. Sobre todo, en el desarrollo de trabajos en los cuales hay que observar forzosamente un determinado orden de procesos individuales; algunos dispositivos adecuados pueden suplir ésta actividad humana de forma más rápida, con una calidad constante y una adecuada planeación de la producción.

Para la correcta utilización de los elementos neumáticos en la automatización industrial, es necesario conocer la estructura y el funcionamiento de éstos equipos. Al mismo tiempo aprender normas, definiciones de conceptos y ser capaz de proyectar y montar sencillos, medianos y avanzados automatismos con sus respectivos mandos.

2.2- Automatización con circuitos neumáticos.

Neumática:

Parte de la mecánica que estudia el comportamiento de los gases sometidos a presión. Se dice que la neumática es la rama de la ingeniería más nueva, estamos hablando que se inicio en los años 50s, cuando se empezó de una manera sistemática a automatizar los procesos de control. En la actualidad es inconcebible hablar de automatización sin pensar en equipos neumáticos. La razón de ello es lo simple y práctico de sus elementos.

2.2.1- Propiedades del aire comprimido.

Puede que sea sorprendente que la neumática se haya difundido tan rápidamente y en poco tiempo. Este fenómeno se debe, entre otras cosas, a que en ciertos problemas de automatización es más sencillo y económico utilizar aire que cualquier otro tipo de fluido.

VENTAJAS DEL AIRE COMPRIMIDO
Abundante
Transportable
Almacenable
Temperatura
Antideflagrante
Limpio
Constitución de los elementos
A prueba de sobrecargas

DESVENTAJAS
Compresible
Fuerza
Escape
Costos

2.2.2- Ventajas del aire comprimido.

Abundante:

El aire comprimible está disponible en cualquier parte y en cantidades prácticamente inagotables.

Transportable:

El aire comprimido puede transportarse por tuberías a largas distancias. No es necesario preocuparse por el retorno del aire.

Almacenable:

Un compresor no tiene que funcionar constantemente. El aire comprimido puede almacenarse en un acumulador (depósito), desde donde se puede recurrir a él. Además, también es posible almacenar el aire comprimido en botellas (cilindros).

Temperatura:

El aire comprimido es indiferente a las oscilaciones de la temperatura. De esta manera es factible trabajar de modo seguro incluso con temperaturas extremas.

Antideflagrante:

El aire comprimido no ofrece peligro de explosión o de incendio. En consecuencia no es necesario adoptar costosas medidas de seguridad contra explosiones.

Limpio:

El aire comprimido es limpio, por lo que no se produce contaminación alguna por fugas de aire en las tuberías o en las unidades de trabajo. Precisamente esta característica es indispensable en la industria alimenticia, maderera, textil y en la fabricación de curtidos.

Montaje:
Los elementos de montaje son fáciles de montar, por lo que los sistemas neumáticos no son costosos.

Velocidad:
El aire comprimido es un medio de trabajo sumamente veloz, por lo que es factible alcanzar altas velocidades de trabajo.

Regulación:
La potencia y la velocidad de los elementos neumáticos pueden ser regulados sin mayor problema.

Seguridad a sobrecargas:
Las herramientas neumáticas y los elementos de trabajo pueden someterse a esfuerzos hasta quedar inmovilizados, por lo que son seguros frente a sobrecargas.

2.2.3- Desventajas del aire comprimido:

Para delimitar con precisión los posibles campos de aplicación de la neumática, es preciso conocer también sus *desventajas.*

Compresión:
El aire, dado que es comprimible, no permite obtener velocidades homogéneas y constantes de los cilindros.

Fuerza:
Los sistemas neumáticos sólo son económicos hasta determinadas potencias. Dado el límite usual de la presión de trabajo 7 kgsF/cm2

Evacuación de aire:
La evacuación del aire produce ruidos. Sin embargo, dicho problema ha podido ser solucionado en buena parte mediante el uso de silenciadores.

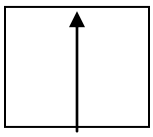
Costos:
El aire comprimido es un medio energético relativamente costoso. No obstante, buena parte de los elevados costos energéticos son compensados por la economía de los elementos y por el alto rendimiento de los mismos.

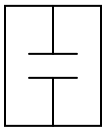
2.3.- Representación esquemática de las válvulas:

Al igual que en la electricidad o en la electrónica los elementos de control se representan por símbolos para hacer más práctico el desarrollo de circuitos. Estos símbolos de ninguna manera representan el sentido o construcción del elemento, su labor es únicamente dar una idea de su funcionamiento.

Las válvulas se representan por medio de vías y de posiciones. Las posiciones se indican por medio de cuadros, donde cada cuadro indica una posición.

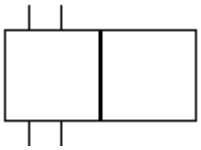
Las vías se representan por líneas con flechas donde la flecha indica el sentido del flujo del aire comprimido.





Las posiciones cerradas se representan por líneas transversales, la unión de dos líneas o tuberías se representan por un punto.

Las conexiones (entradas y salidas) se representan por medio de líneas unidos en una posición que también indica la posición de reposo de la válvula.



La otra posición se obtiene desplazando lateralmente los cuadros hasta que las conexiones coincidan.

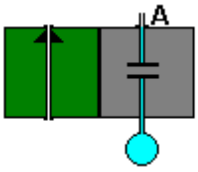


Figura a

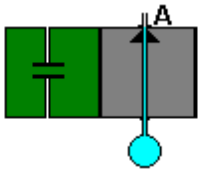


Figura b

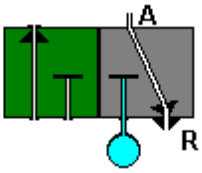


Figura c

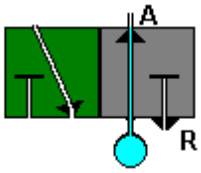
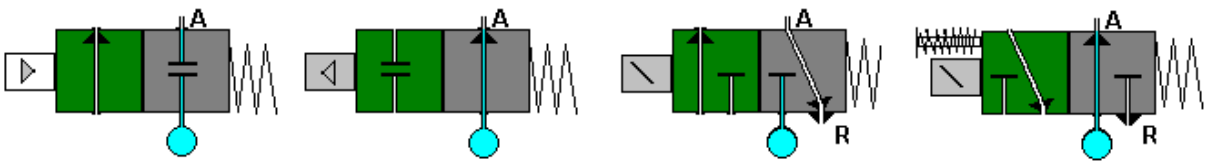


Figura d

Cuando en el escape, el triangulo viene pegado a la válvula indica que no tiene rosca para recuperar el aire, mientras que cuando el triangulo viene despegado indica que tiene rosca para poder recuperar el aire en el escape.

En la figura a se encuentra una válvula de dos vías dos posiciones normalmente cerrada, en la Figura b se encuentra una válvula de dos vías dos posiciones normalmente abierta, en la figura c se encuentra una válvula de tres vías dos posiciones normalmente cerrada y en la figura d se encuentra una válvula de tres vías dos posiciones normalmente abierta.

Finalmente se presentan las mismas válvulas con diferentes tipos de accionamiento y retorno por muelle.



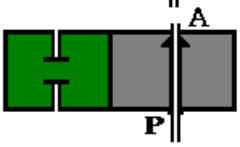
A continuación se presentan los símbolos de algunas válvulas distribuidoras.

2.3.1 - Simbología neumática

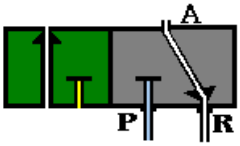
VALVULAS DE MANDO Y REGULACION

Válvula distribuidora 2/2
cerrada en posición de reposo

Válvula distribuidora 2/2
abierta en posición de reposo



Válvula distribuidora 3/2
cerrada en posición de reposo



Válvula distribuidora 3/2
abierta en posición de reposo

Válvula distribuidora 4/2
(Memoria)

Válvula distribuidora 5/2
(Memoria)

Válvula distribuidora 3/3
posición central bloqueada

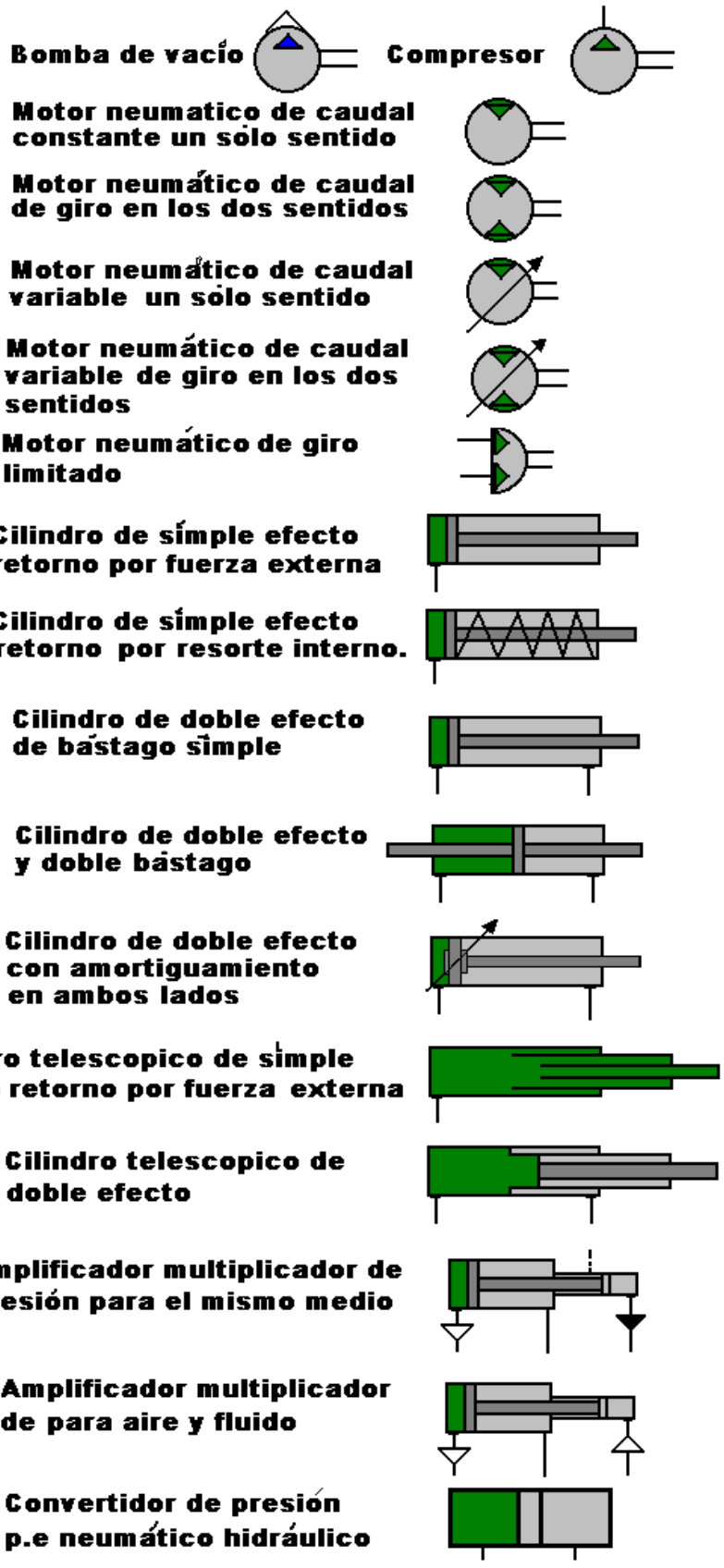
Válvula distribuidora 4/3
cerrada en posición central

Válvula distribuidora 4/3
posición central en flotación

Válvula distribuidora 5/3
cerrada en posición central

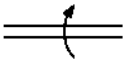
Válvula distribuidora
varias posiciones

TRANSFORMACION DE ENERGIA

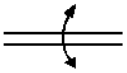


MANDOS MECÁNICOS

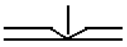
Arbol con giro en un sentido.



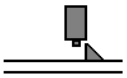
Arbol con giro en los dos sentidos.



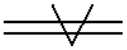
Enclavamiento.



Bloqueo
(reprecentación esquemática
del enclavamiento).

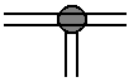


Dispositivo de desenclavamiento
instantaneo.

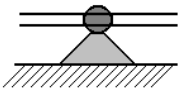


Articulación simple.

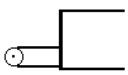
Articulación con leva.

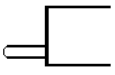


Articulación con un punto fijo.



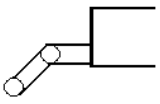
RODILLO





LEVA

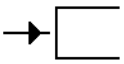
RODILLO
ESCAMOTEABLE



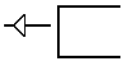
MUELLE

ACCIONAMIENTOS NEUMÁTICOS

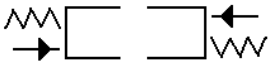
PRESIÓN DIRECTA



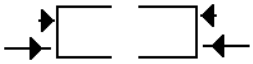
PRESIÓN INDIRECTA



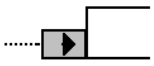
CENTRADO
POR MUELLE



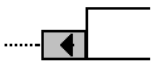
CENTRADO POR
PRESIÓN



PRESIÓN INDIRECTA
(Servopilotaje)

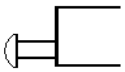


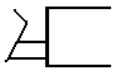
PRESIÓN INDIRECTA
(Servopilotaje)



ACCIONAMIENTOS MUSCULARES

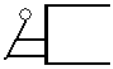
PULSADOR

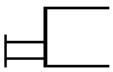




PEDAL

PALANCA





GENERAL

DENOMINACIÓN DE RACORES

Conexiones → A, B, C

Alimentación de presión → P

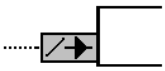
Escapes → R, S, T

Fugas → L

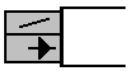
Conexiones de pilotaje → Z, Y, X

ACCIONAMIENTOS COMBINADOS

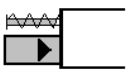
ACCIONAMIENTO POR BOBINA
POR PRESIÓN (Servopilotaje)



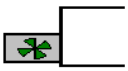
ACCIONAMIENTO POR BOBINA
POR PRESIÓN



ACCIONAMIENTO POR PRESIÓN
O MANUAL

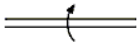


ACCIONAMIENTO DISPONIBLE
(no normalizado)

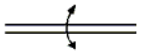


MANDOS MECANICOS

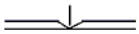
Arbol con giro en un sentido.



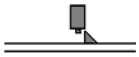
Arbol con giro en los dos sentidos.



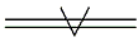
Enclavamiento.



Bloqueo
(reprecentación esquemática
del enclavamiento).

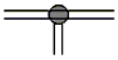


Dispositivo de desenclavamiento
instantaneo.

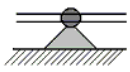


Articulación simple.

Articulación con leva.



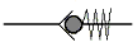
Articulación con un punto fijo.



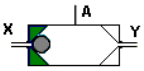
VALVULAS DE BLOQUEO

VALVULA ANTIRRETORNO

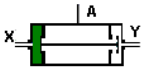
VALVULA ANTIRRETORNO
CON MUELLE



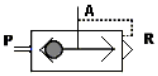
VALVULA SELECTORA DE
CIRCUITO



VALVULA DE
SIMULTANEIDAD



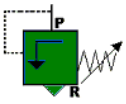
VALVULA DE ESCAPE
RAPIDO



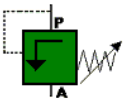
VALVULA DE CIERRE

VALVULAS DE PRESION

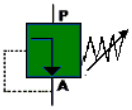
VALVULA LIMITADORA DE
PRESION



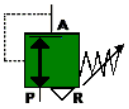
VALVULA DE SECUENCIA
AJUSTABLE



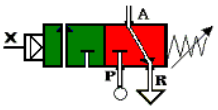
VALVULA DE SECUENCIA
AJUSTABLE SIN ESCAPE

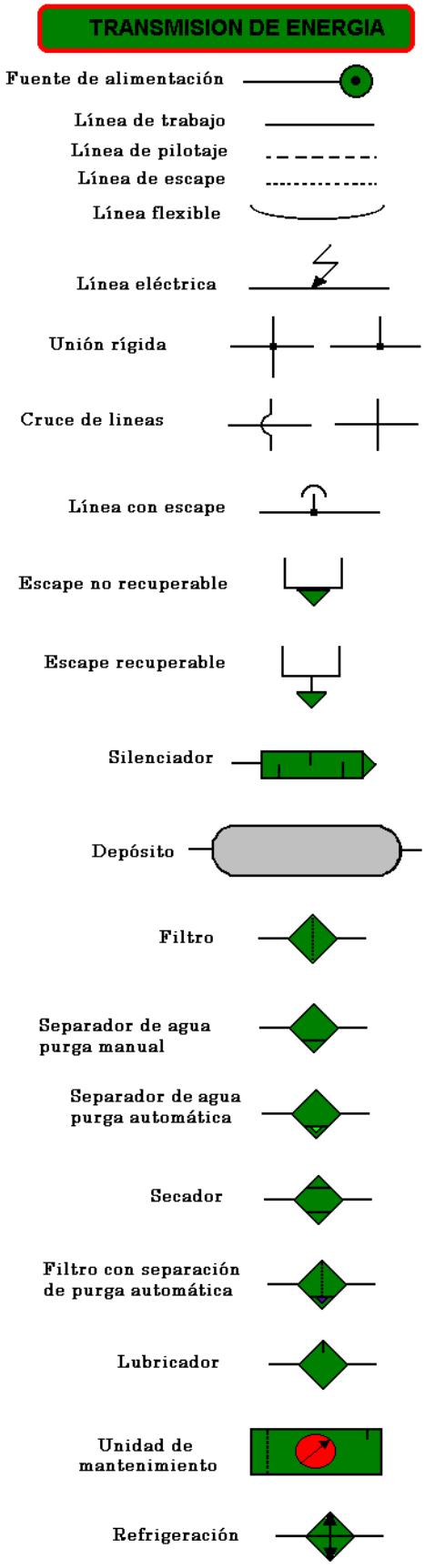


REGULADOR DE PRESION
AJUSTABLE CON ESCAPE



VALVULA DE SECUENCIA
CON ESCAPE



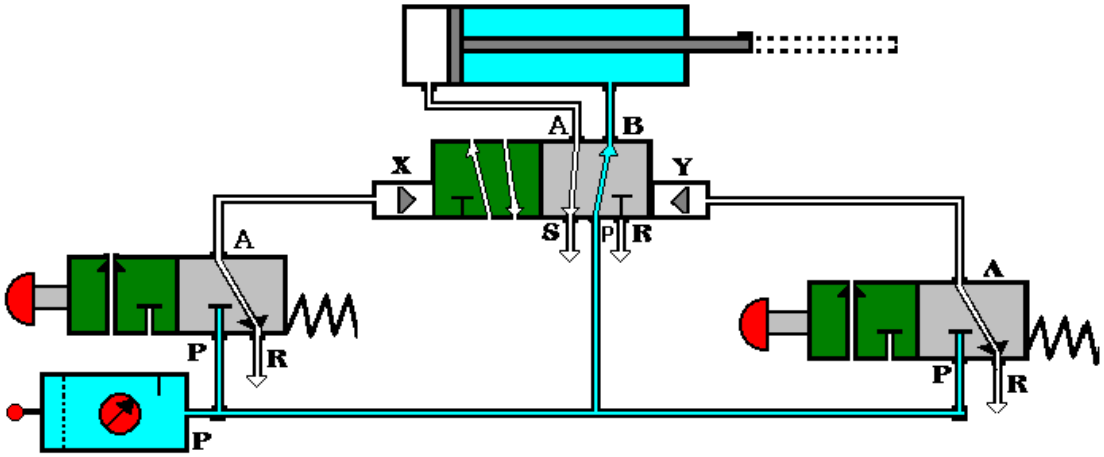

TRANSMISION DE ENERGIA
Fuente de alimentación
Línea de trabajo
Línea de pilotaje
Línea de escape
Línea flexible
Línea eléctrica
Unión rígida
Cruce de lineas
Línea con escape
Escape no recuperable
Escape recuperable
Silenciador
Depósito
Filtro
Separador de agua
purga manual
Separador de agua
purga automática
Secador
Filtro con separación
de purga automática
Lubricador
Unidad de
mantenimiento
Refrigeración

2.4 - Ejercicios neumáticos.

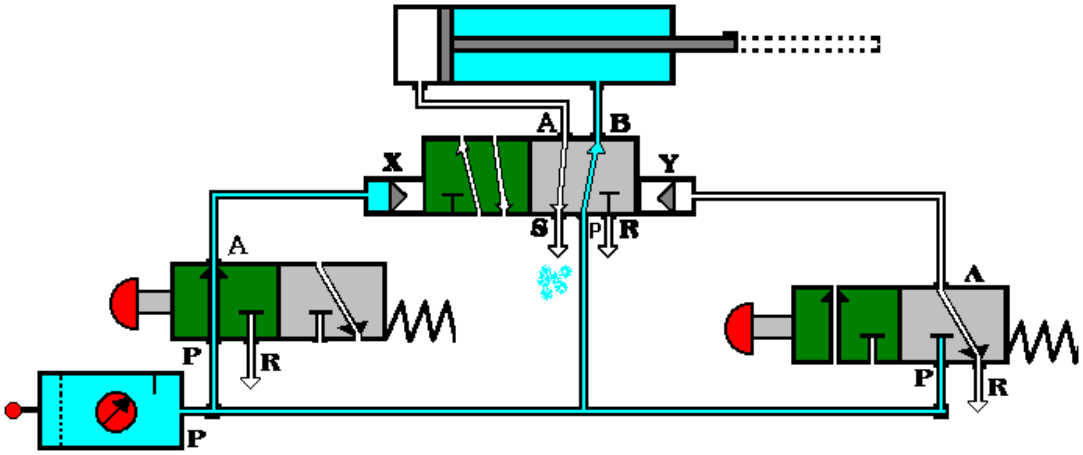
EJERCICIO Nº 1

Se tiene una prensa con un cilindro de doble efecto y se desea sujetar una pieza para realizar en ella algún trabajo, por lo tanto se deberán utilizar dos botones neumáticos uno para que baje la prensa y uno para que suba una vez que se realizó el trabajo en la pieza.

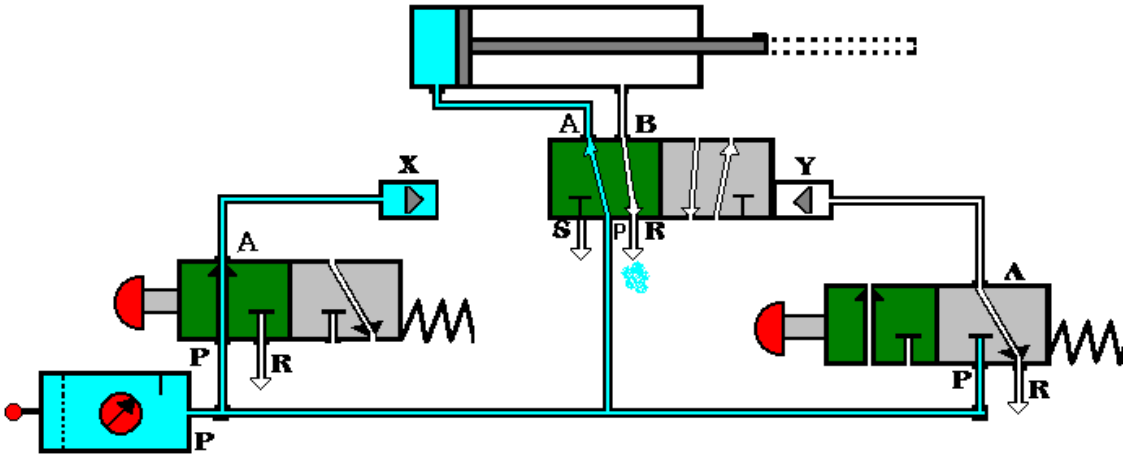
A continuación presentamos una solución paso a paso Ver figuras paso a paso).



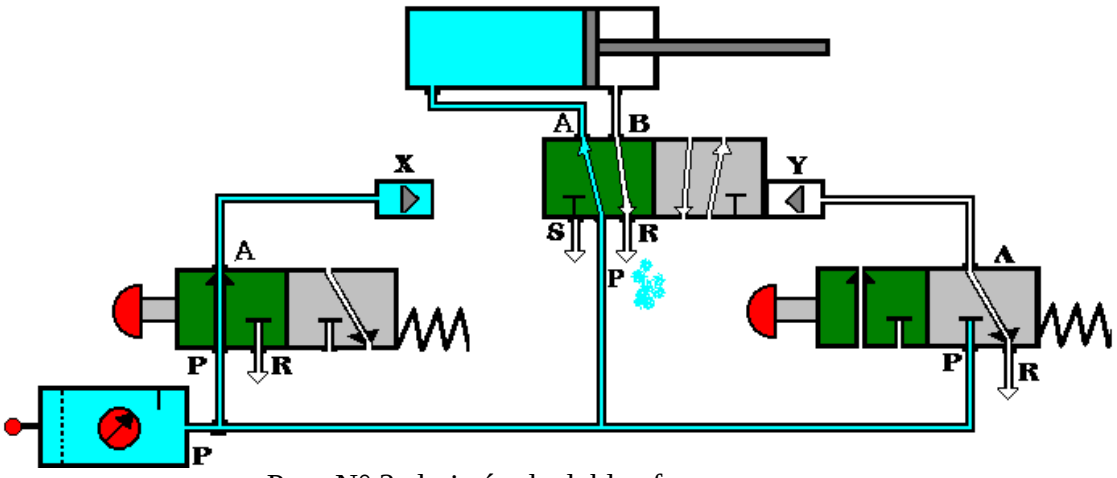
Paso N° 0 Circuito de control neumático en reposo.



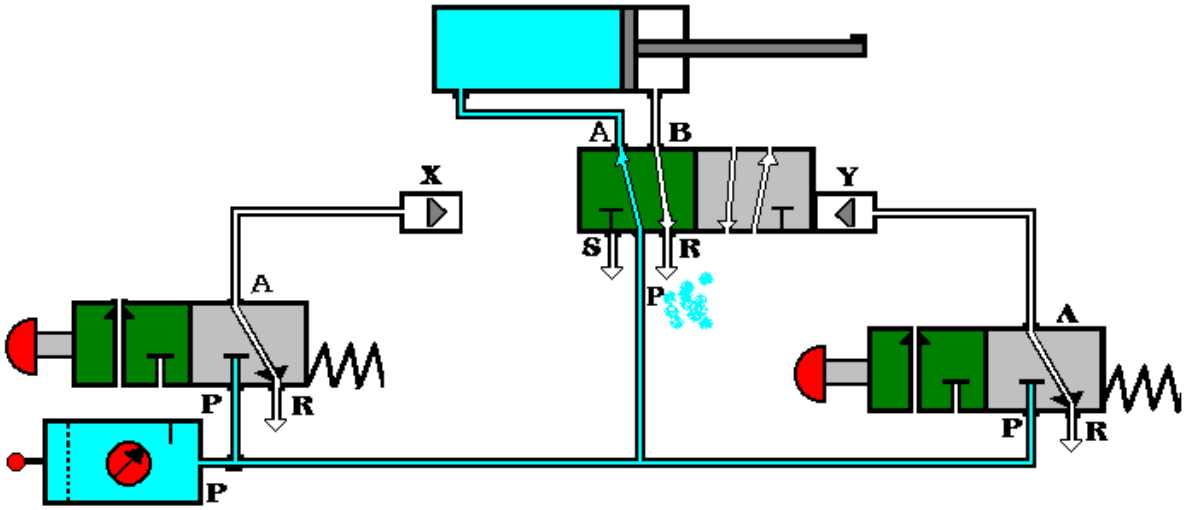
Paso N° 1 se activa el botón de inicio



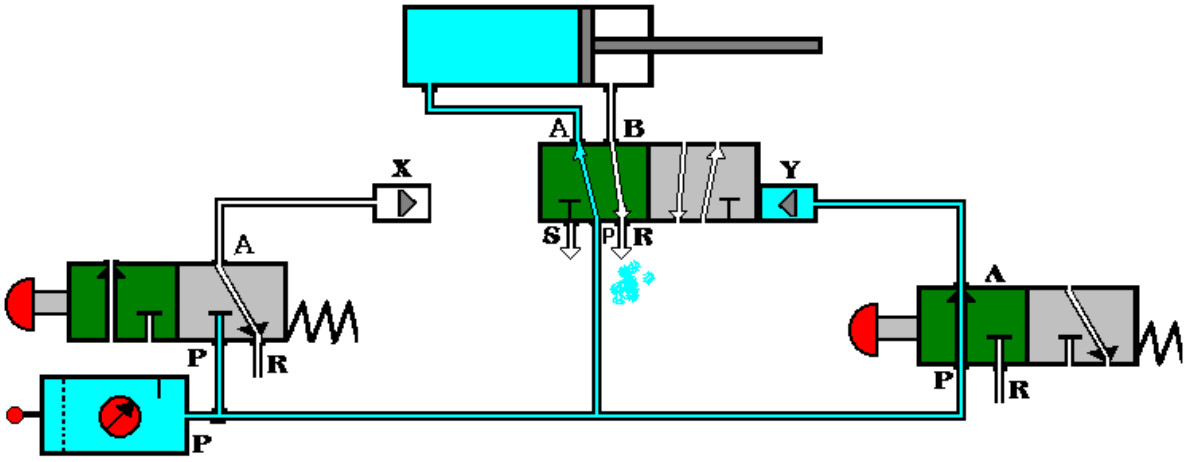
Paso N° 2 se activa la válvula de memoria 5/2.



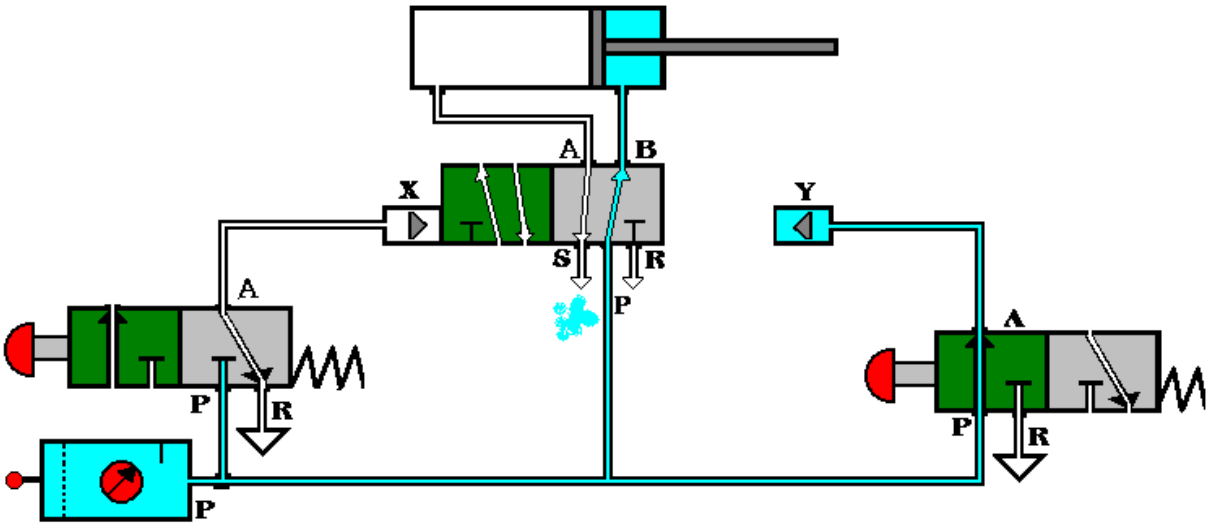
Paso N° 3 el pistón de doble efecto avanza.



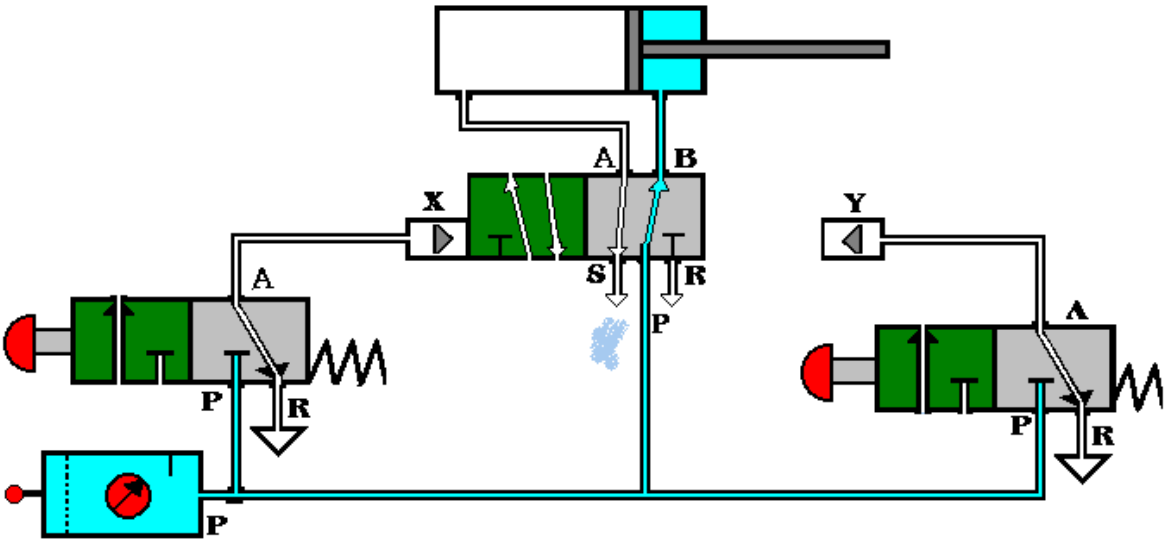
Paso N° 4 se desactiva el botón de inicio y regresa a su posición de reposo.



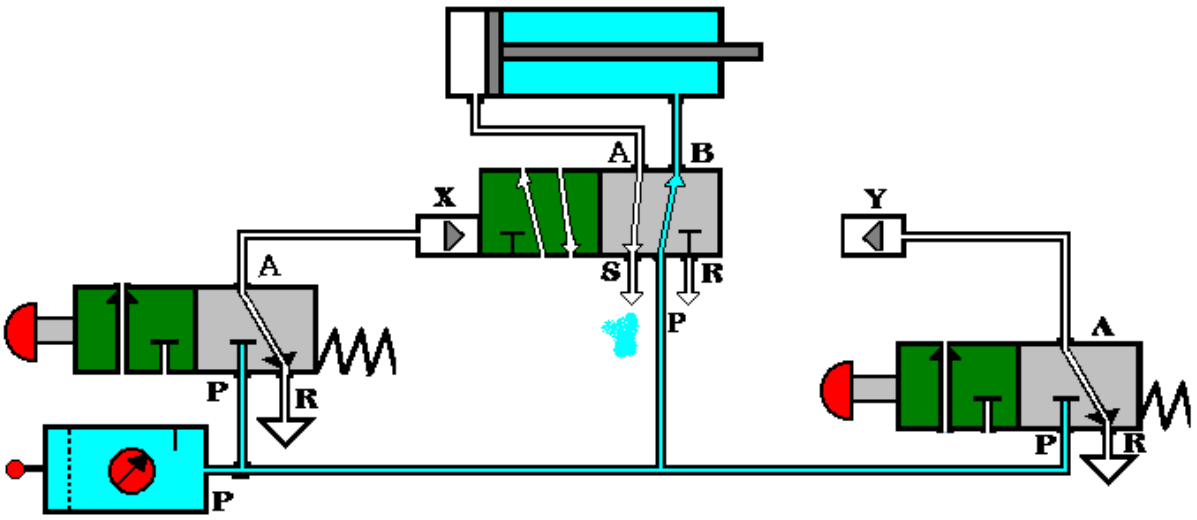
Paso N° 5 se activa el botón de retorno.



Paso N° 6 la válvula de memoria se desactiva regresando a su posición de reposo.



Paso N° 7 Se desactiva el botón de retorno.



Paso N° 8 se regresa el pistón a la posición original.

EJERCICIO Nº 2

Diseñe el circuito de control neumático, anterior pero ahora que al llegar a su final de carrera el pistón se regrese en forma automática (Ver figura 2.4.2)..

SOLUCIÓN:

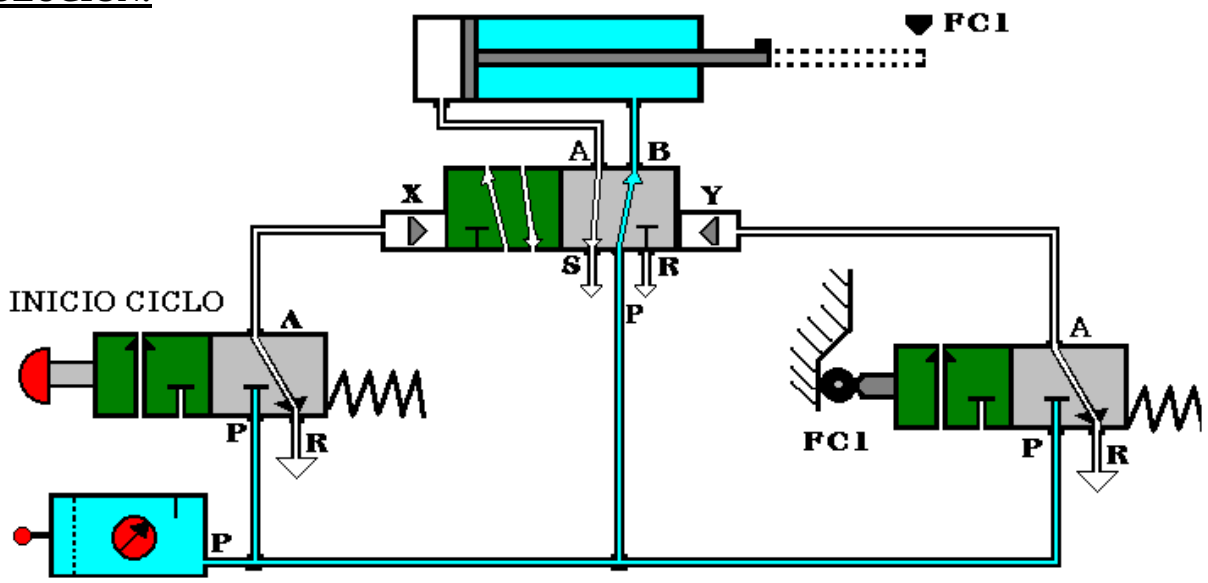


FIGURA N° 2.4.2

EJERCICIO Nº 3

Como podrá usted observar en el ejemplo de la figura anterior, si el pistón realiza su trabajo muy rápido y no alcanza el operador a soltar la mano del botón , el pistón sale de nuevo pudiendo dañar la pieza al repetir el ciclo, por lo tanto realice el circuito de protección para que no repita el ciclo (Ver figura 2.4.3).

SOLUCIÓN:

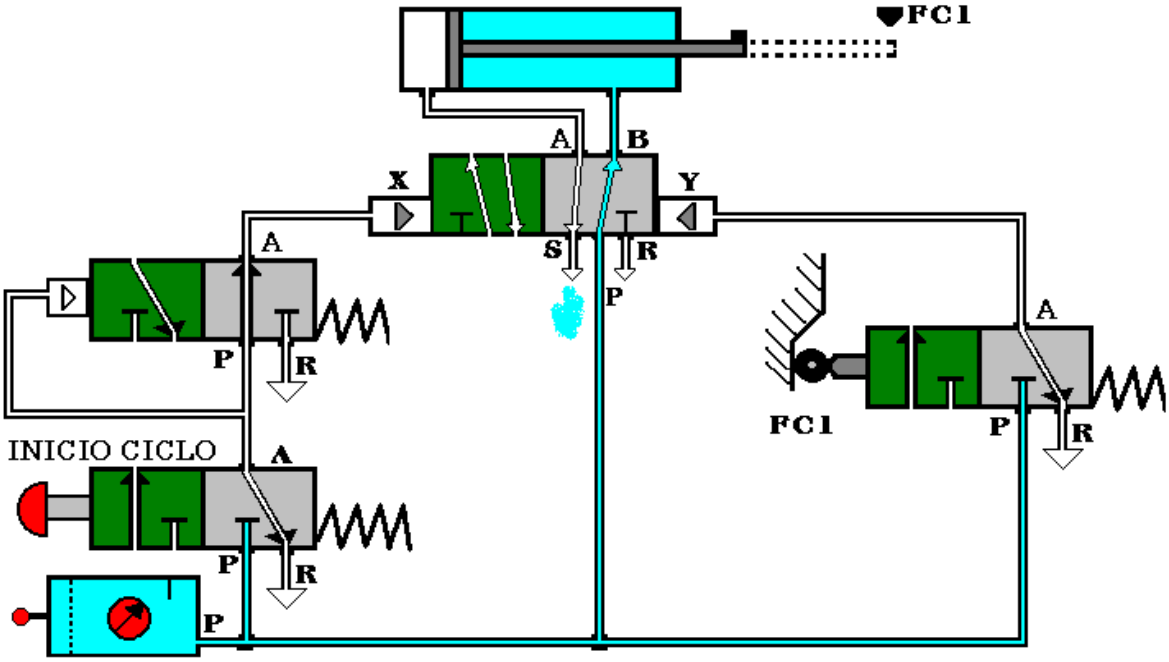


FIGURA N° 2.4.3

EJERCICIO Nº 4

En el circuito anterior se puede presentar el caso de que el pistón después de estar un tiempo parado tienda a bajar un poco, o que por alguna razón no llega a la posición de reposo con lo que las piezas trabajadas no tienen la calidad requerida. Diseñe el circuito de control para que si el pistón no esta en la posición de reposo no inicie el ciclo. (Ver figura 2.4.4). **SOLUCION**

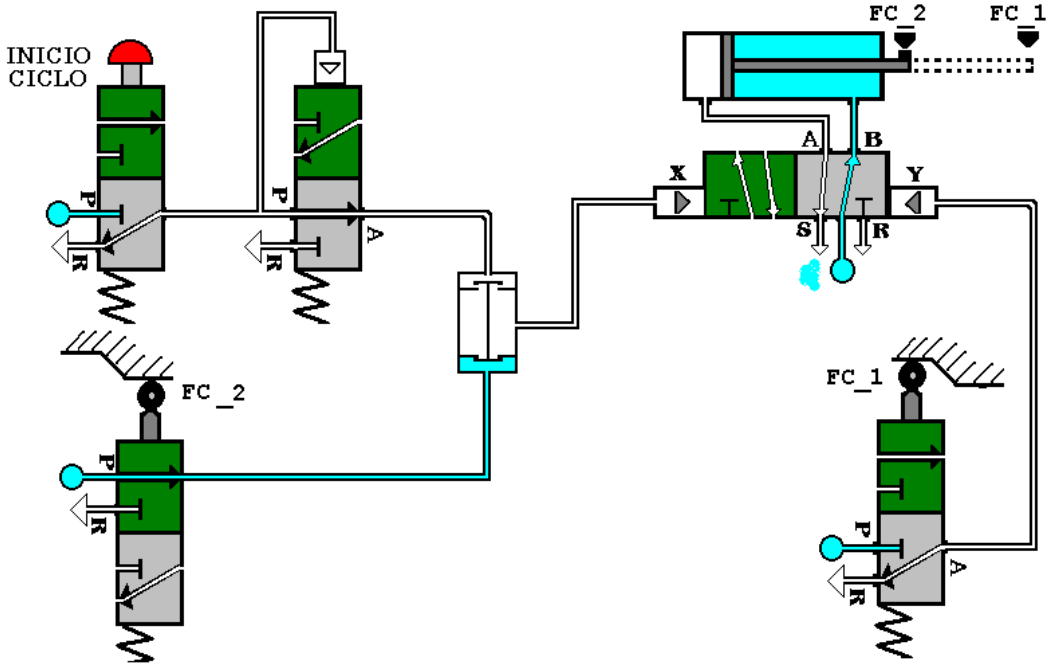


FIGURA N° 2.4.4

EJERCICIO Nº 5

Realice el circuito de control para el ejercicio anterior pero ahora se requiere pegar dos piezas, por lo que el pistón lleva en la punta un troquel con una resistencia para calentar el troquel de tal manera que al bajar dure un tiempo abajo y luego suba en forma automática (Ver Fig. 2.4.5).

SOLUCION

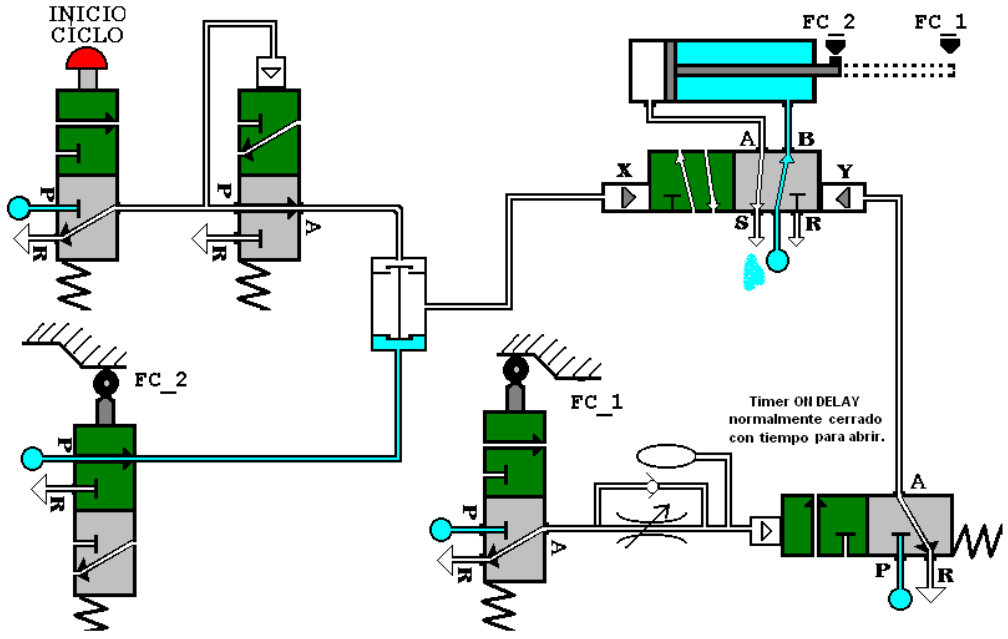


FIGURA N° 2.4.5

EJERCICIO Nº 6

Ahora se requiere ponerle un alimentador de piezas en automático, por lo que se requiere que al llegar al reposo el pistón baje de nuevo. Para parar el circuito coloque un botón de paro de tal manera que en el momento que sea, si se oprime el botón de paro se continúe el ciclo y al terminarlo se pare (Ver figura 2.4.6).

SOLUCIÓN:

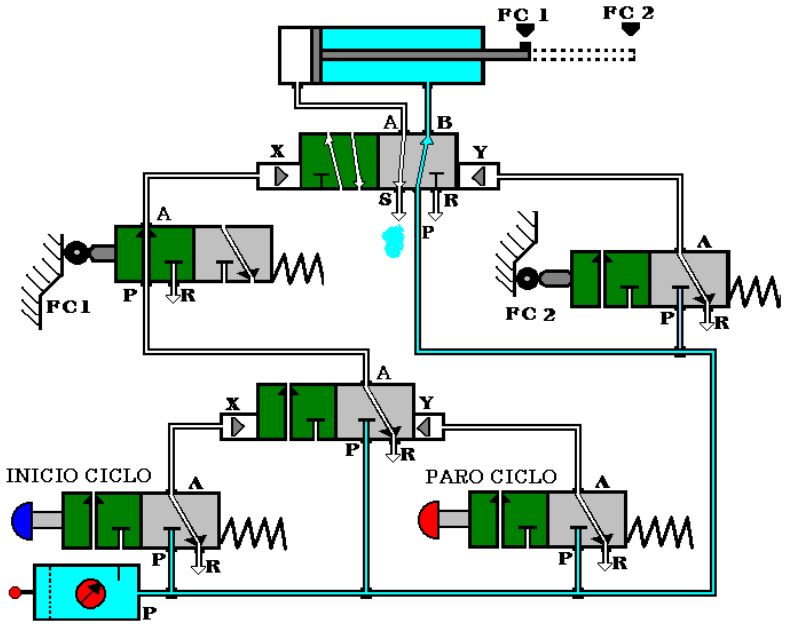


FIGURA N° 2.2.6

EJERCICIO Nº 7

En el ejemplo del circuito anterior se puede observar que si damos el impulso de inicio de ciclo y el final de carrera FC_1 no esta activado no iniciara el ciclo, y también observamos que si se activa por alguna razón el final de carrera FC_1 iniciara el ciclo en ese instante, con lo cual puede provocar un accidente. Diseñe el arreglo correspondiente para evitar este problema (Ver Fig 2.2.7).

SOLUCIÓN:

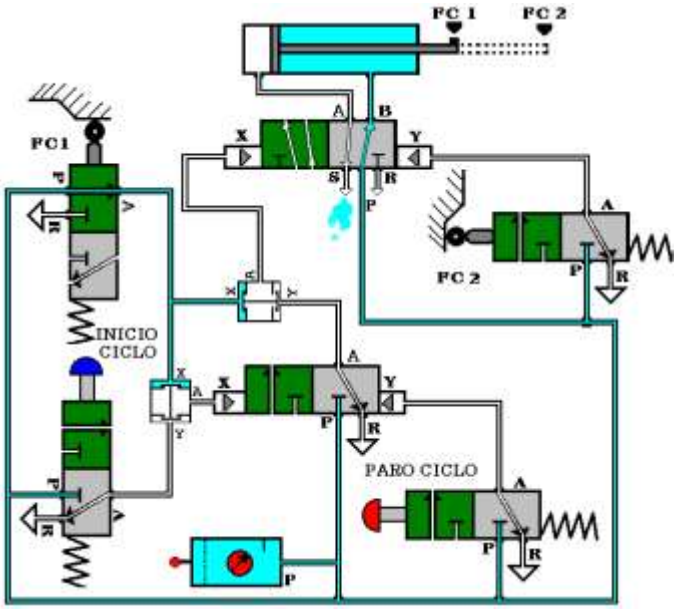


FIGURA N° 2.4.7

EJERCICIO Nº 8

Diseñe el circuito de control neumático para que al dar inicio, entre un pistón a sujetar la pieza, cuando está sujeta que entre un segundo pistón a barrenar y al llegar al final de su carrera se regrese y cuando llegue al reposo se suelte la pieza terminando el ciclo. Además, se deberá tener cuidado de que no repita ciclo y que estén en reposo ambos pistones para poder iniciar.

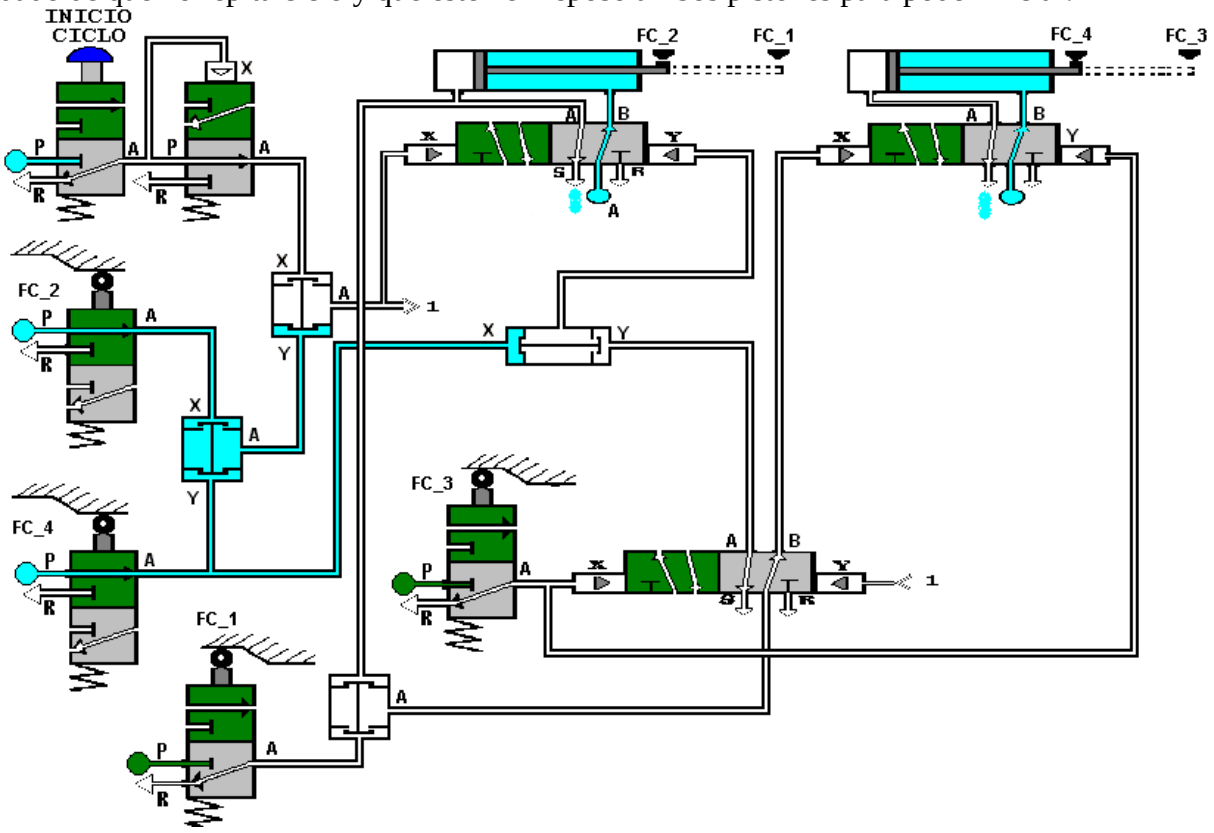


FIGURA N° 2.4.8

2.5- Regulación de velocidad.

La figura N° 2.4.9 nos muestra la manera de regular la velocidad de avance de un pistón de simple efecto con válvulas reguladoras unidireccionales

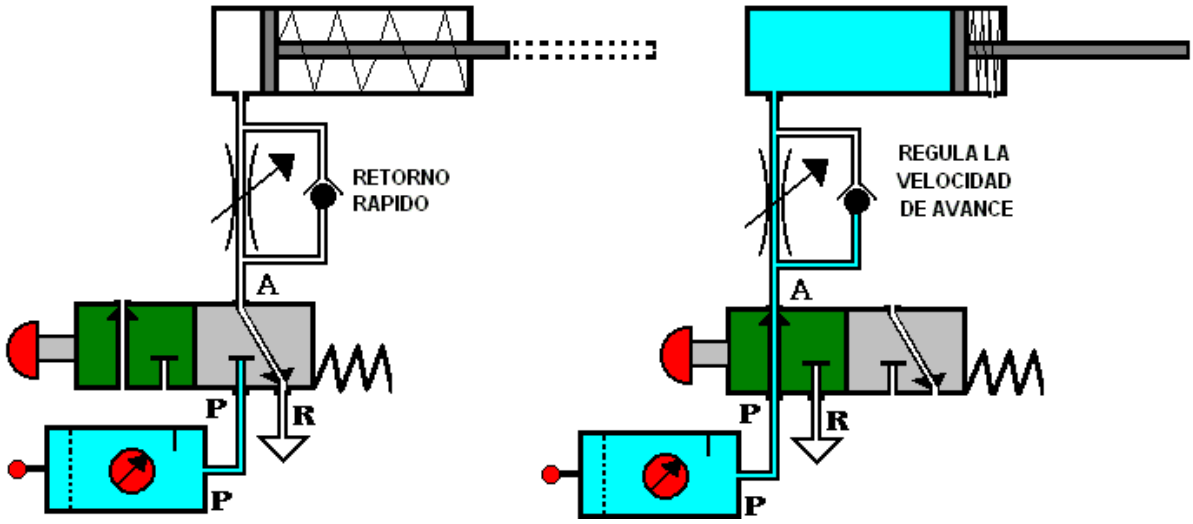


FIGURA N° 2.4.9

La figura N° 2.4.10 nos muestra la forma de regular la velocidad de retorno de un pisón de simple efecto con válvulas reguladoras unidireccionales.

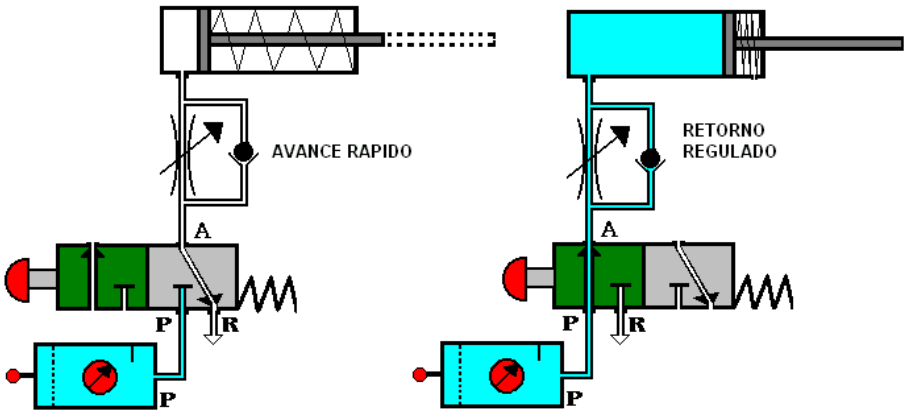


FIGURA N° 2.4.10

La figura N° 2.4.11 nos muestra la manera de regular tanto la velocidad de avance como de retorno cuando se requiere que ambas tengan siempre la misma velocidad.

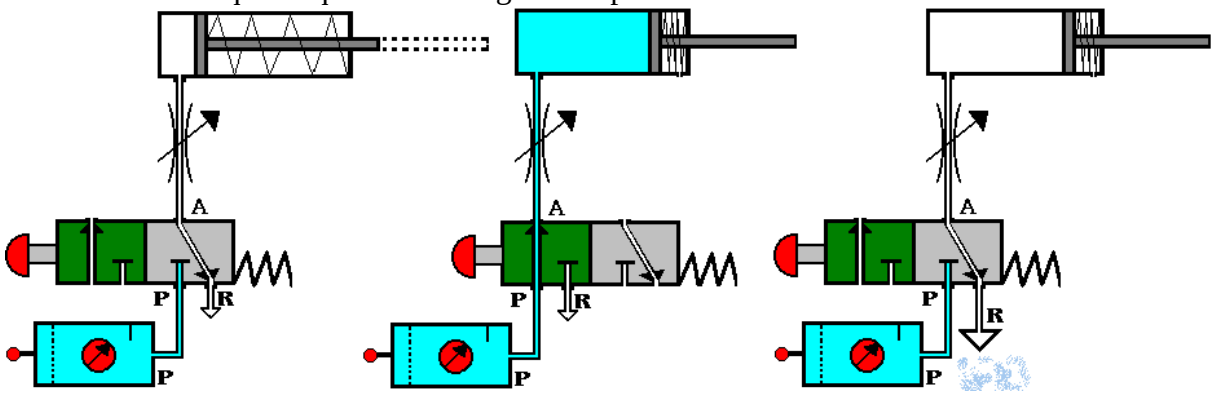


FIGURA N° 2.4.11

La figura N° 2.4.12 nos muestra como regular tanto la velocidad de avance como la de retorno en forma independiente de un pistón de simple efecto.

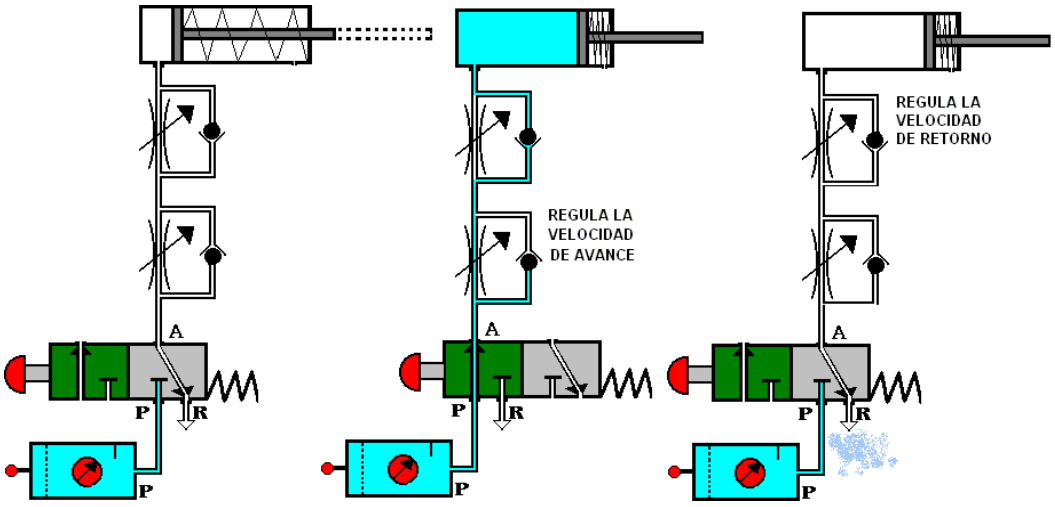


FIGURA N° 2.4.12

2.5- Ejercicios con regulación de velocidad.

EJERCICIO Nº 9

La figura N° 2.5.1 nos muestra la manera de regular la velocidad de avance de un pistón de doble efecto; observe como para regular la velocidad de avance se regula siempre el escape.

SOLUCIÓN:

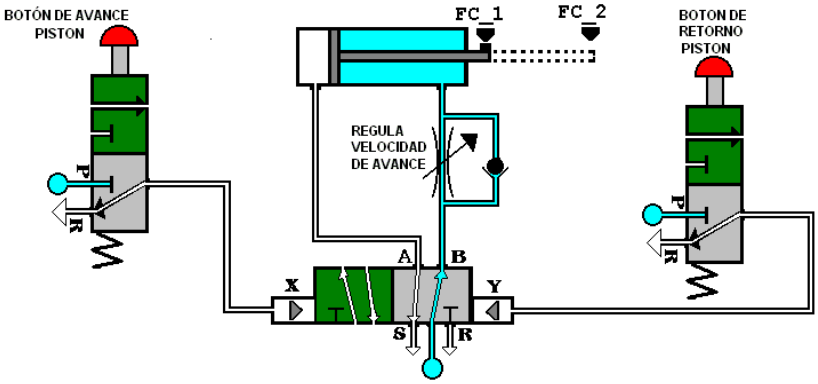


FIGURA N° 2.5.1

EJERCICIO Nº 10

La figura N° 2.5.2 nos muestra la manera de regular la velocidad de retorno de un pistón de doble efecto; observe que para regular la velocidad de retorno se regula siempre el escape.

SOLUCIÓN:

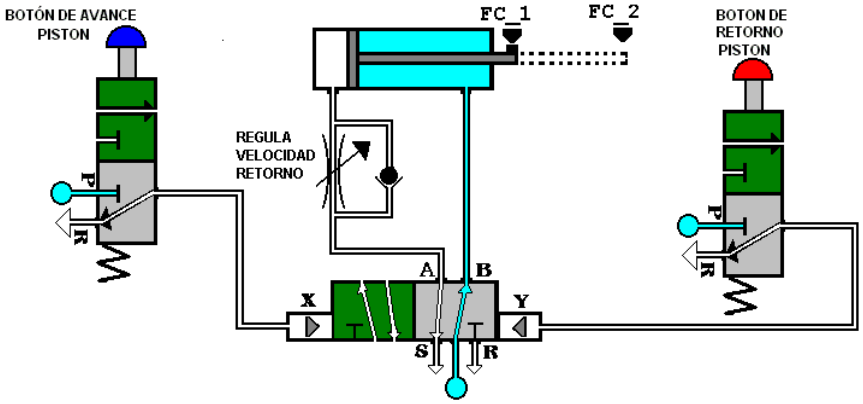


FIGURA N° 2.5.2

EJERCICIO Nº 11

La figura N° 2.5.3 nos muestra la manera de regular tanto la velocidad de avance como la de retorno en forma independiente (con válvulas reguladores unidireccionales) de un pistón de doble efecto. Observe que siempre se regula el escape. **SOLUCIÓN:**

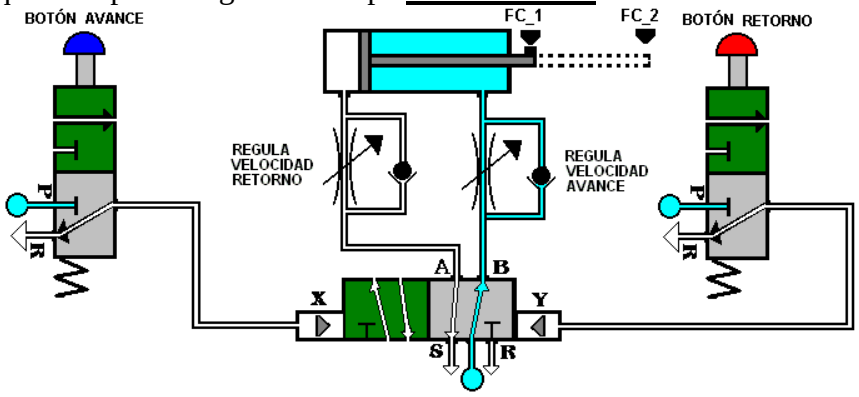


FIGURA N° 2.5.3

EJERCICIO Nº 12

La figura N° 2.5.4 nos muestra la manera de regular la velocidad tanto de avance como la de retorno cuando se requiere que ambas velocidades sean siempre la misma con válvula reguladora unidireccional. **SOLUCIÓN:**

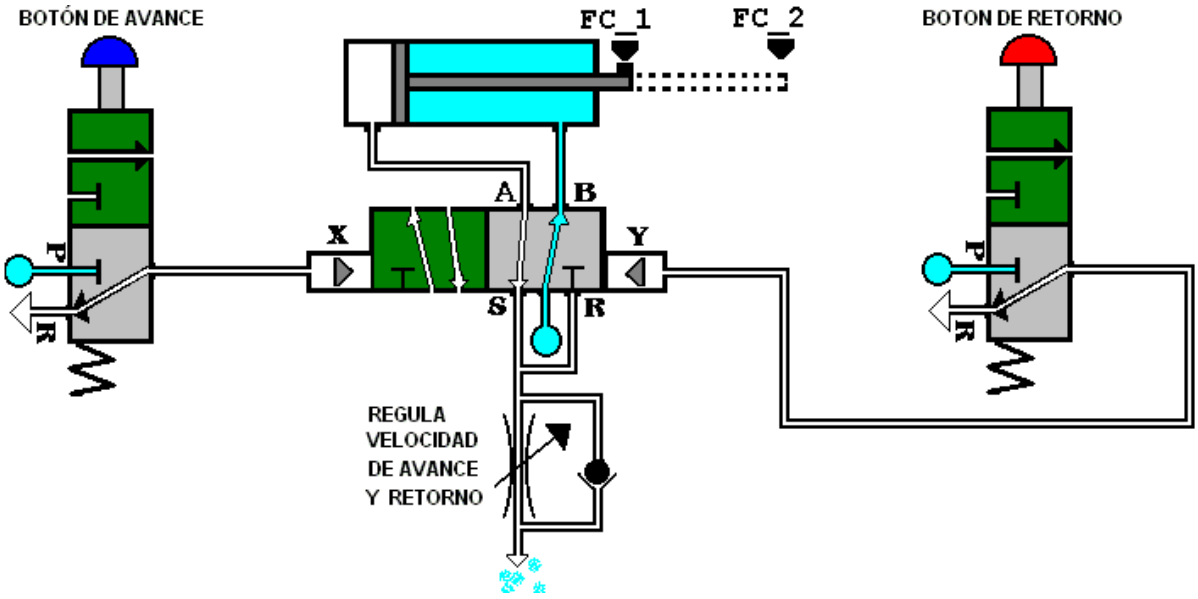


FIGURA N° 2.5.4

En la figura N° 2.4.17 se observa la manera de regular la velocidad de avance y retorno con una válvula reguladora bidireccional cuando se requiere que tanto el avance como el retorno sea la misma velocidad.

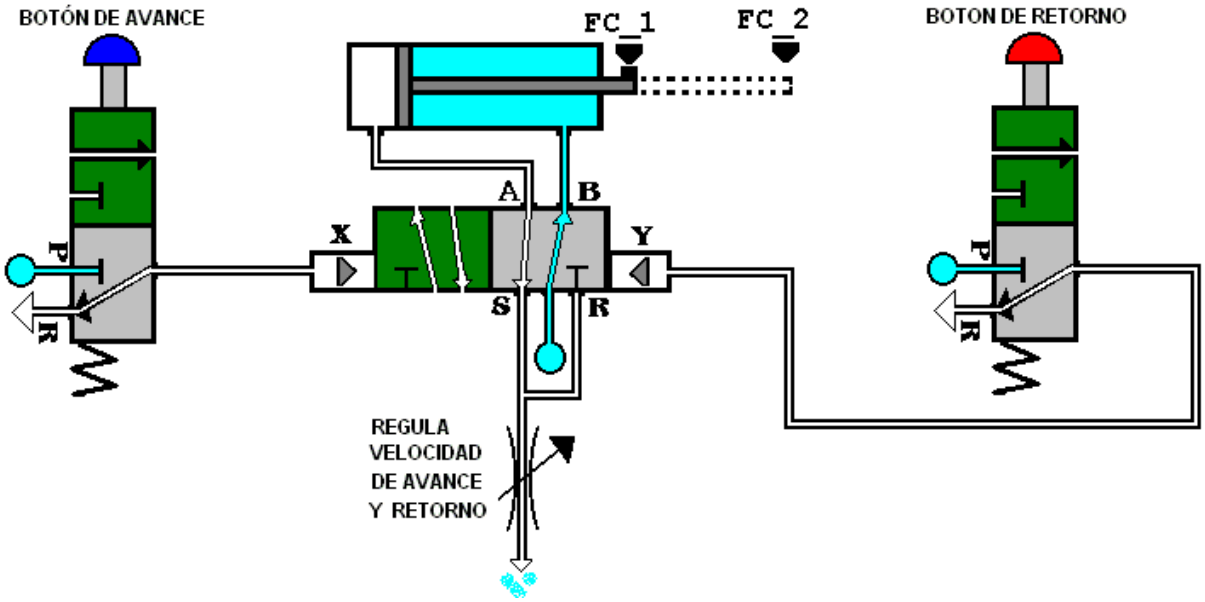


FIGURA N° 2.4.17

2.6 Automatizaciones neumáticas.

EJERCICIO Nº 13

Se quiere automatizar el proceso de pegado de piezas, en el cual se tienen dos pistones uno para sujetar y otro para pegar, los dos pistones son de doble efecto y con doble actuación, el pistón 1 sujeta y el pistón 2 pega, además se requiere de un selector de dos posiciones en una posición deberá realizar un ciclo y en la otra posición deberá trabajar en continuo, para poder iniciar el ciclo se requiere de un botón de inicio ciclo, si el selector se tiene en continuo para parar se deberá tener dos opciones, una cambiando el selector a un ciclo o bien oprimiendo o dando un impulso al botón de paro.

NOTA; el ciclo deberá parar siempre en condiciones de reposo y para iniciar ciclo se requiere que ambos pistones estén en reposo, ver figura 2.6.1.

SOLUCIÓN:

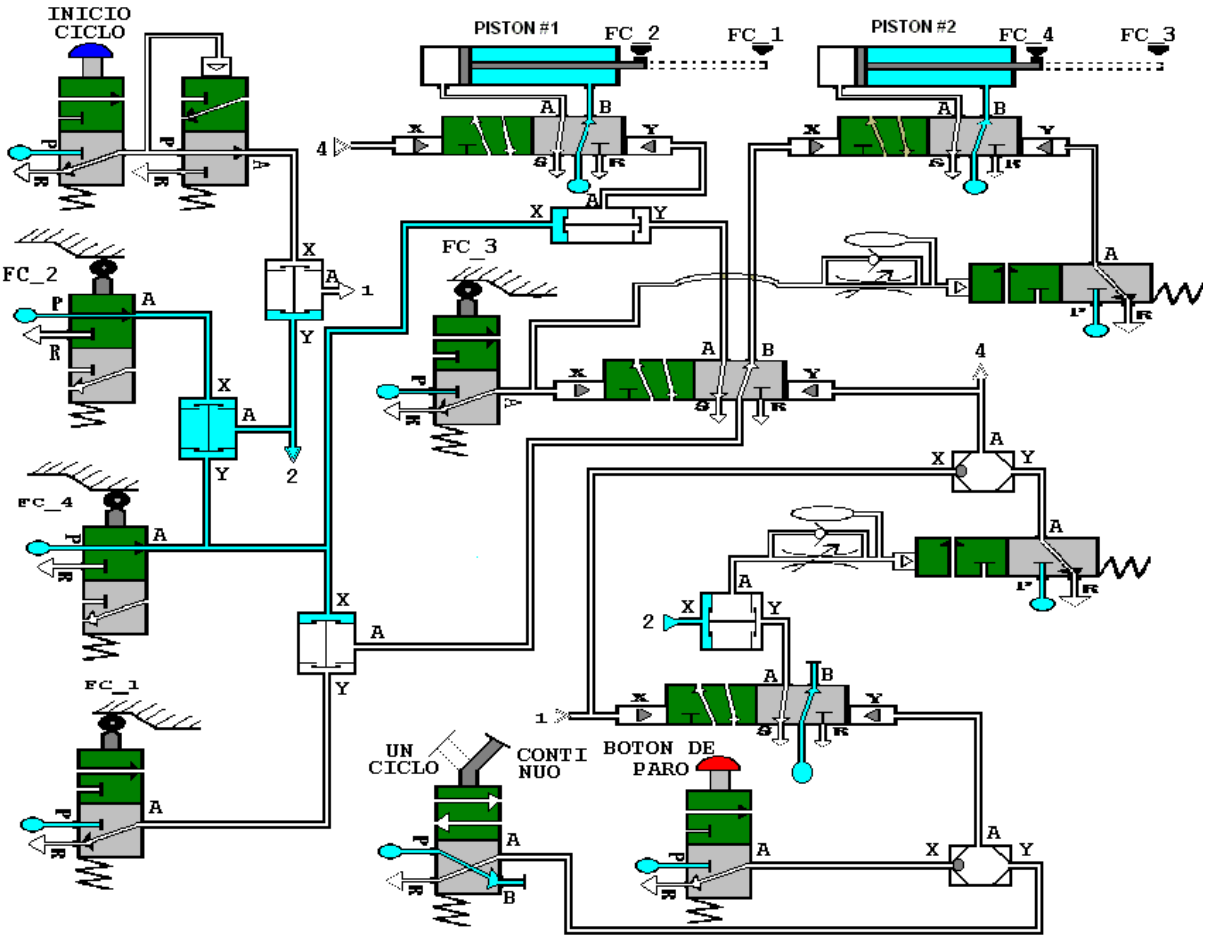


FIGURA N° 2.6.1

EJERCICIO Nº 14

Solución al problema anterior utilizando pistones de Simple efecto para el pistón sujetador y pistón remachador. Pero ahora se desea que al terminar el ciclo, de un tiempo para quitar la pieza e iniciar de nuevo, con esto estamos asegurando que la producción sea constante durante el curso, para parar el ciclo coloque un botón de paro. También se desea colocar un contador neumático en lugar de hacer el contador. El siguiente circuito a sido realizado, dibujado y probado con el Software de Fluid Sim de Festo versión libre de demostración, ver figura 2.6.2.

SOLUCIÓN:

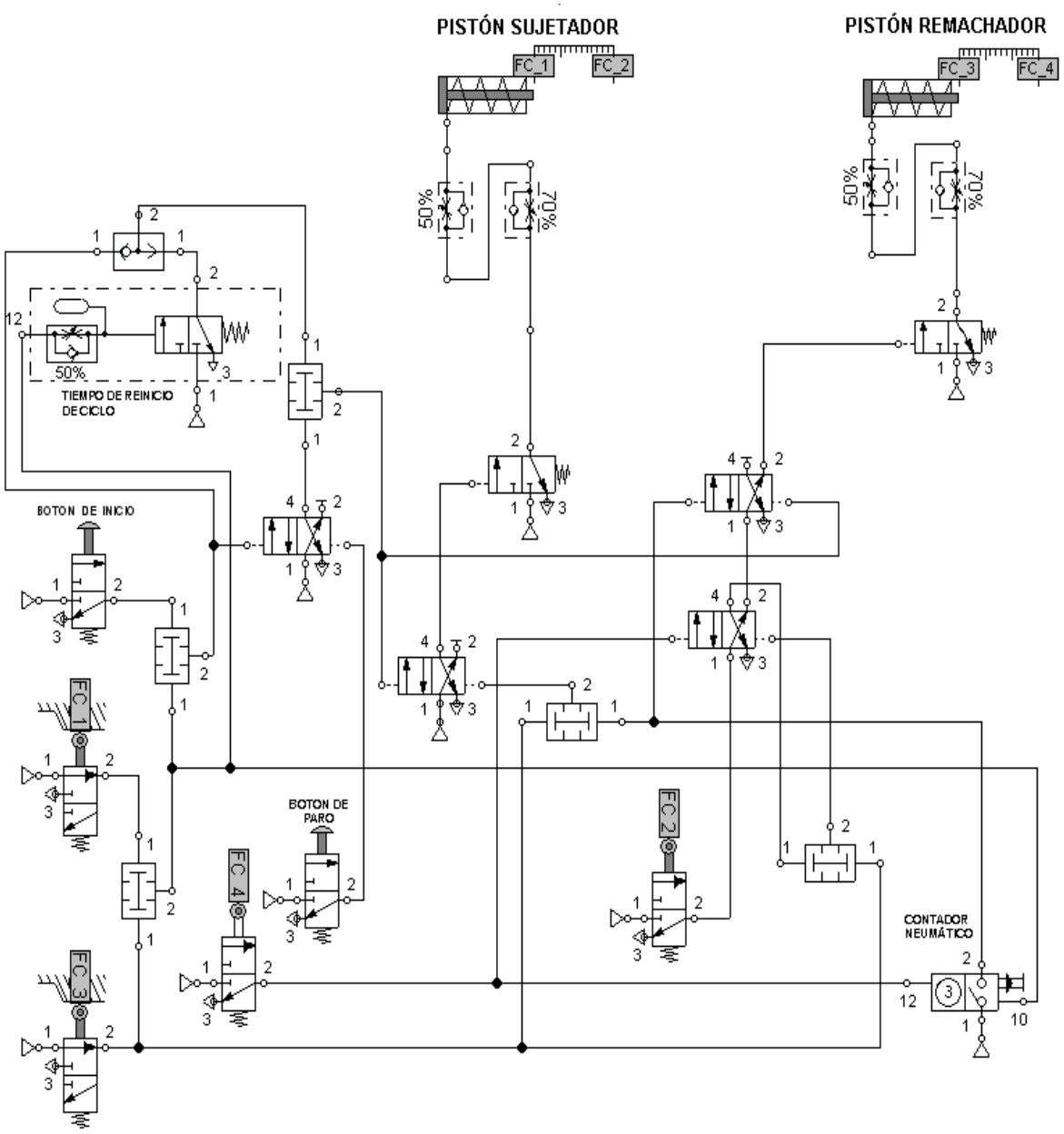


FIGURA N° 2.6.2

EJERCICIO Nº 15

Se quiere automatizar el proceso de encorchado de botellas, para esto el operador debe colocar manualmente el corcho en la boca de la botella, una vez hecho esto se oprime el botón de inicio de ciclo y sale un pistón para sujetar la pieza, cuando la pieza esta fija, baja un segundo pistón y golpea el corcho tres veces consecutivas para que entre completamente, una vez hecho esto el pistón sujetador regresa y suelta la pieza finalizando el ciclo.

Se van a utilizar pistones de doble efecto y un contador neumático que nos indica cuando el pistón remachador completa los tres ciclos. El siguiente circuito a sido realizado dibujado y probado con el software de Fluid Sim de Festo versión libre de demostración, ver figura 2.6.3

SOLUCIÓN:

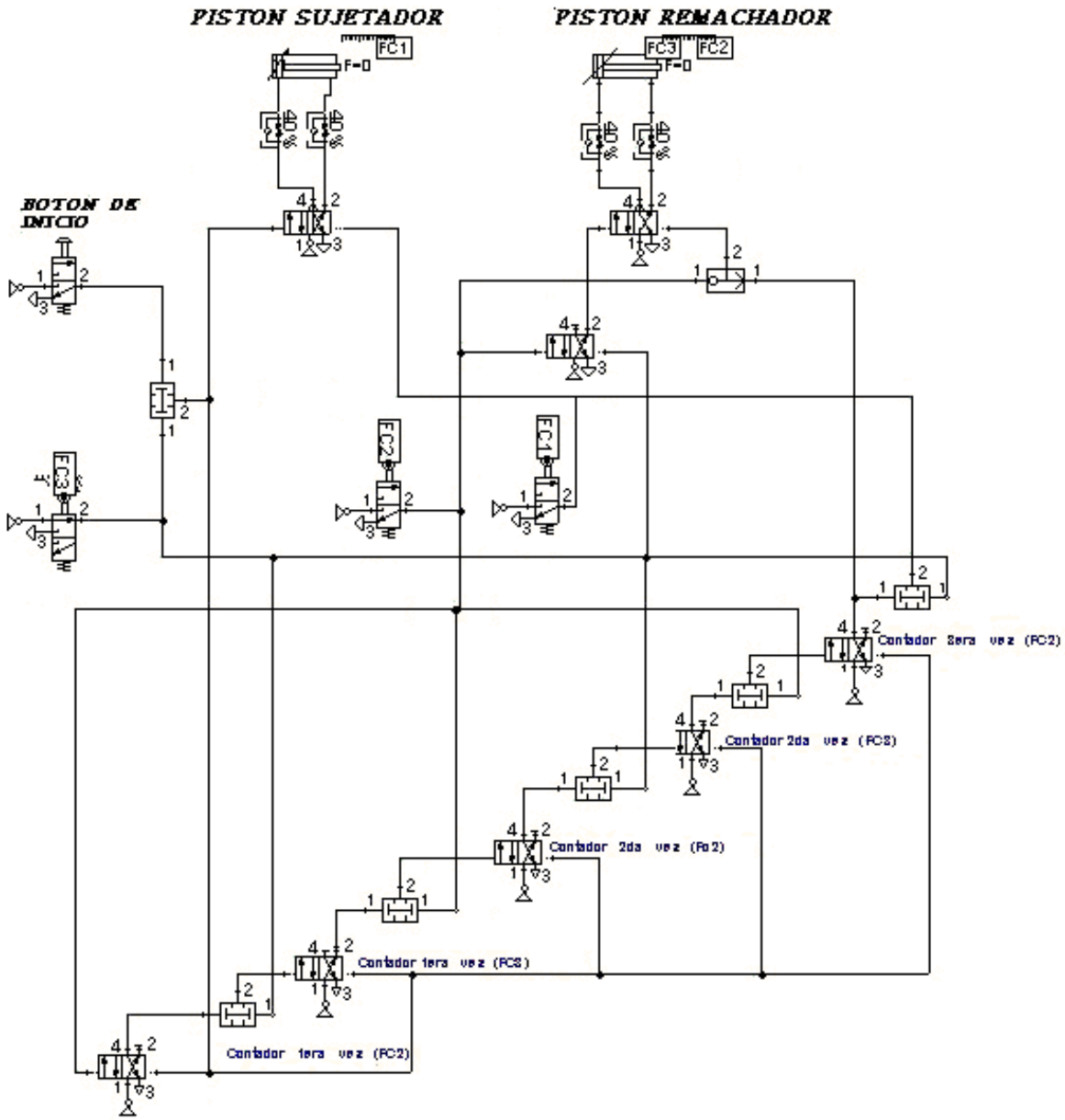


FIGURA N° 2.6.3

EJERCICIO Nº 16

Se quiere automatizar el proceso de encorchado de botellas, para esto el operador debe colocar manualmente el corcho en la boca de la botella, una vez hecho esto se oprime el botón de inicio de ciclo y sale un pistón para sujetar la pieza, cuando la pieza esta fija, baja un segundo pistón y golpea el corcho tres veces consecutivas para que entre completamente, una vez hecho esto el pistón sujetador regresa y suelta la pieza.

Se va a utilizar un pistón de doble efecto y uno de simple efecto y un contador neumático que nos indique cuando el pistón remachador completa los tres ciclos. El siguiente circuito a sido realizado dibujado y probado con el Software de Fluid Sim de Festo versión libre de demostración, ver figura 2.6.4.

SOLUCIÓN:

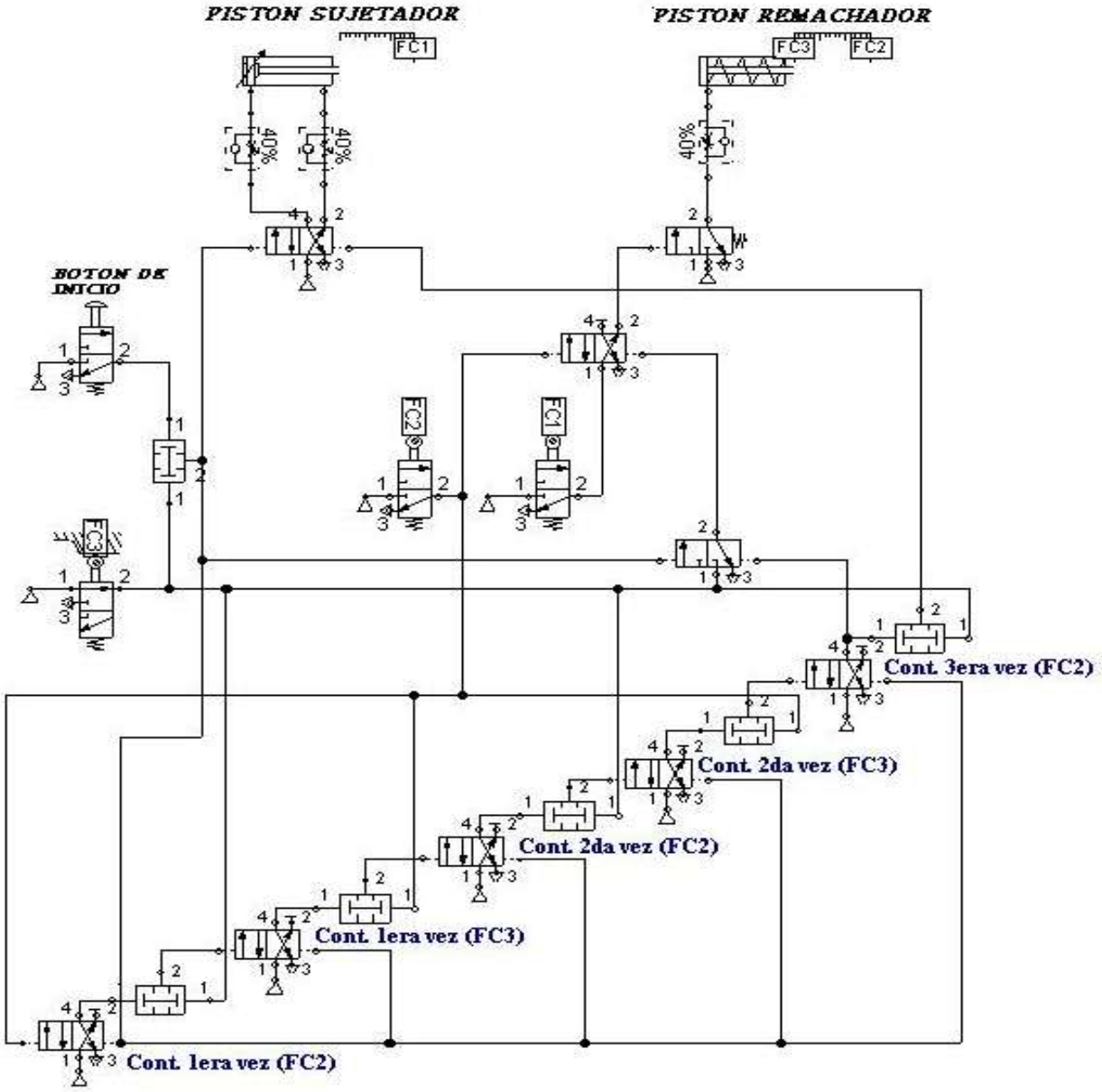


FIGURA N° 2.6.4

EJERCICIO Nº 17

Solución al problema anterior utilizando pistones de Simple efecto para el pistón sujetador y de doble efecto para el pistón remachador. El siguiente circuito a sido realizado, dibujado y probado con el Software de Fluid Sim de Festo versión libre de demostración, ver figura 2.6.5.

SOLUCIÓN:

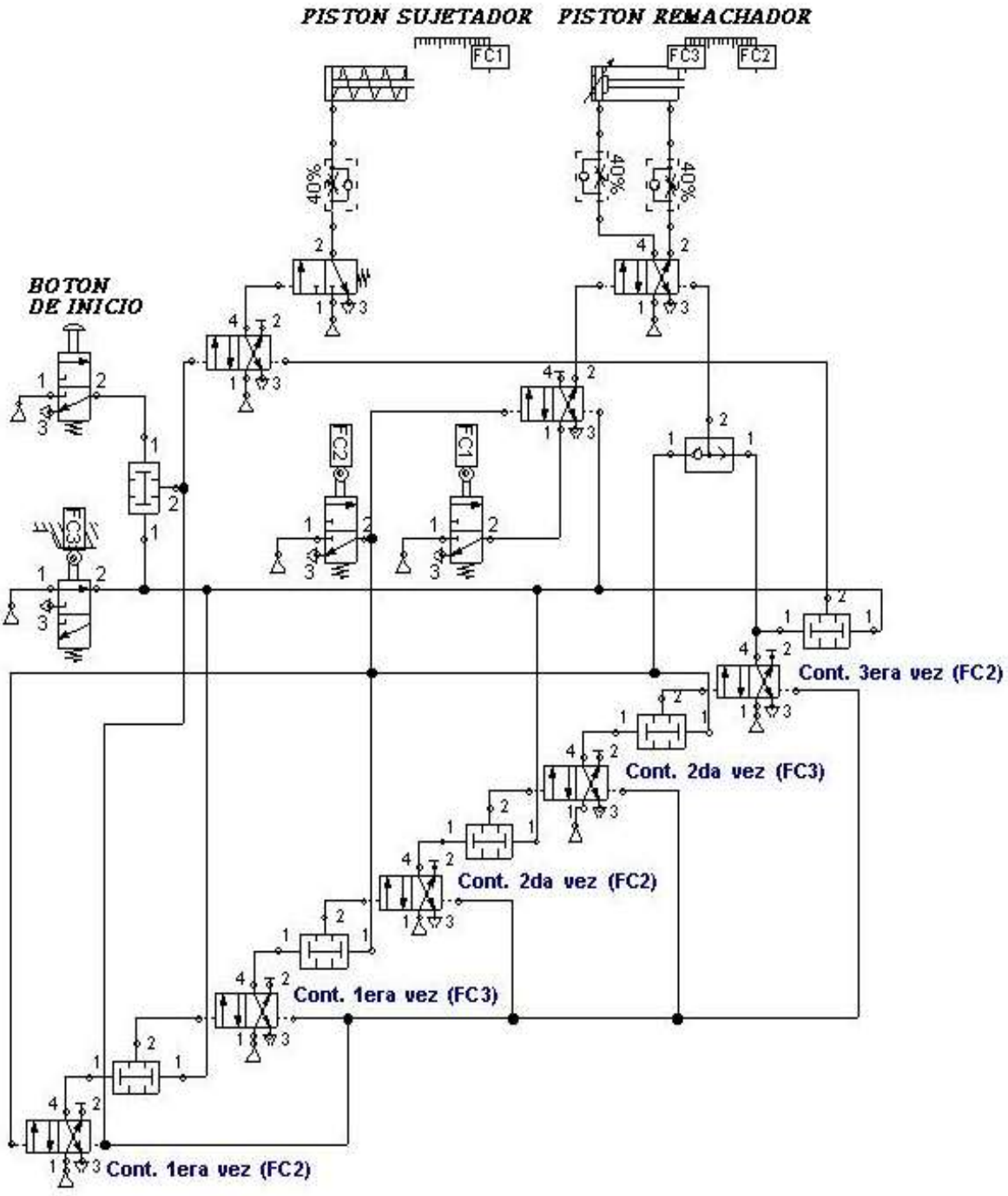


FIGURA N° 2.6.5

CAPÍTULO 3 CONTROL ELECTRO NEUMÁTICO

Introducción:

El impresionante progreso que se ha tenido en los últimos años en el área de automatización ha tenido una considerable afectación sobre las técnicas de control. Los problemas a resolver son cada vez más complejos y los imperativos respecto a los mandos son, naturalmente más delicados. Precisión, rapidez y seguridad son los requisitos que se deben tener en cuenta hoy en día al realizar una automatización. Una gran cantidad de dispositivos electrónicos han sido desarrollados y comercializados, pero sin embargo no se han desarrollado los procesos de enseñanza para que el diseñador pueda aplicarlos de una manera más eficiente.

En muchas ocasiones nos encontramos con dispositivos donde inclusive desarrollan tan sólo las funciones que se puedan efectuar con ellas y el fabricante se abstiene de dar alguna aplicación para nos ser involucrado en caso de que la persona que utiliza dicho dispositivo realice una aplicación peligrosa que puede afectar al equipo y a la persona que lo utiliza.

En la actualidad, la mayoría de las automatizaciones son realizadas por medio de control electro neumático, esto es, la parte de potencia es realizada por equipo neumático y la parte de control por medios eléctricos o electrónicos, dentro del control eléctrico podemos realizarlos por control electromecánico y por medios electrónicos por electrónica de compuertas lógicas o por control programable a través de microcontroladores o por medio de control programable PLC.

En el presente capítulo presentamos la posibilidad de realizar los ejercicios propuestos por medio de control electromecánico. También y antes de iniciar el estudio del control electroneumático por medio de control electromecánico, queremos dejar bien claro que aunque parezca obsoleto realizar dicho control la idea es de que se comprendan las bases del control y de esta forma quedemos en condiciones de iniciar el estudio del control programable por medio de PLCs a través de los diagramas de escalera.

Recomendamos no brincarse este capítulo ya que nos dará las bases para realizar automatizaciones por medio de control programable PLCs.

3.1- Ejemplos con circuitos electro neumáticos.

EJEMPLO 3.1:

Se tiene una prensa con un cilindro de doble efecto y se desea sujetar una pieza para realizar en ella algún trabajo, por lo tanto se deberán utilizar dos botones eléctricos uno para que baje la prensa y otro para que suba una vez que se realizó el trabajo en la pieza.

Realice el circuito de control electro neumático.

SOLUCIÓN:

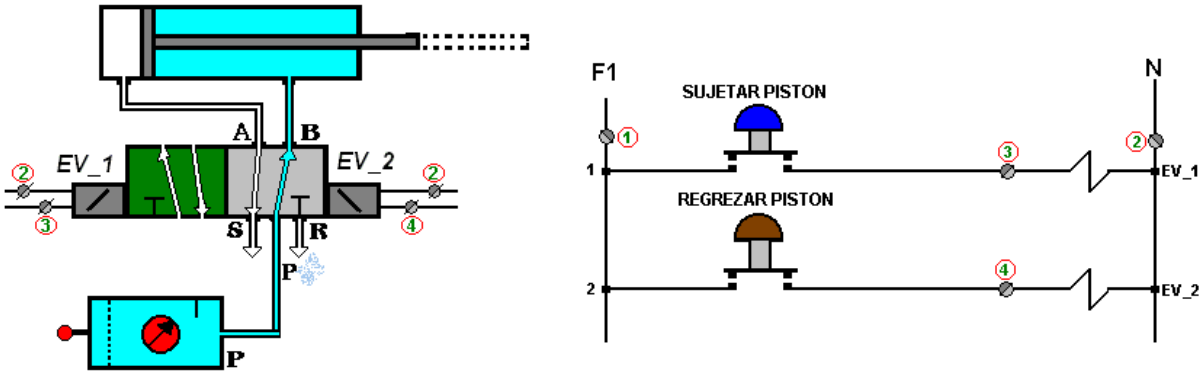


FIGURA # 3.1

EJEMPLO 3.2:

Realice el mismo circuito pero ahora con una electro válvula con una sola solenoide y retorno por resorte.

SOLUCIÓN:

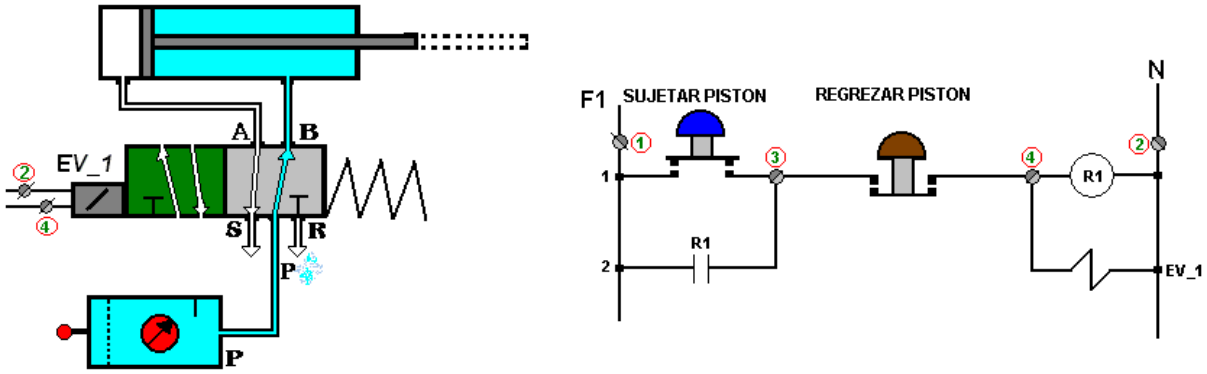


FIGURA # 3.2

EJEMPLO 3.3:

Diseñe el circuito de control electro neumático para el ejercicio No 1 con electro válvula de doble solenoide pero ahora que al llegar al final de su carrera se regrese en forma automática.

SOLUCIÓN:

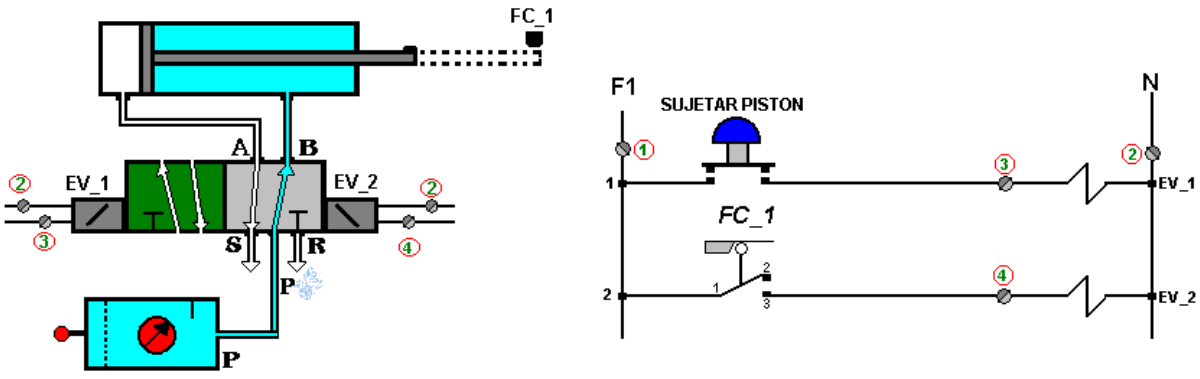


FIGURA # 3.3

EJEMPLO 3.4.

Diseñe el circuito de control electro neumático para el ejercicio No 2 con electro válvula de una solenoide y retorno por muelle pero ahora que al llegar al final de su carrera se regrese en forma automática.

SOLUCIÓN:

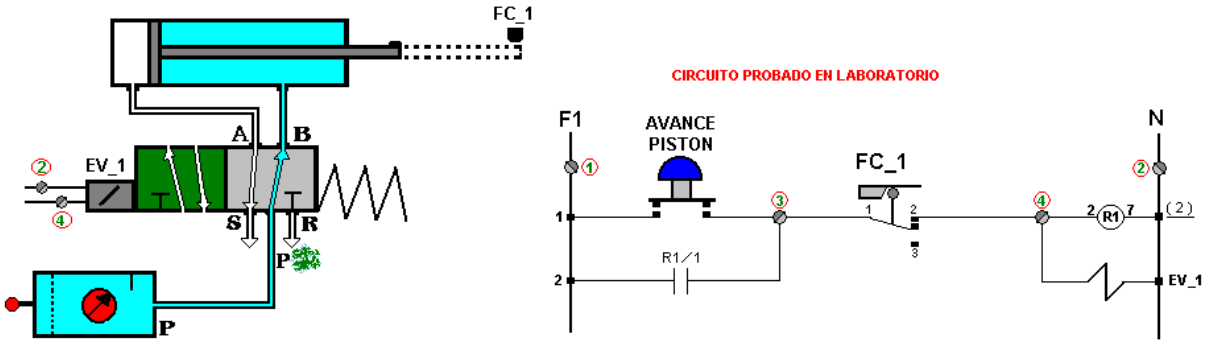


FIGURA # 3.4

EJEMPLO 3.5.

Como podrá usted observar en el ejemplo de la figura anterior, si el pistón realiza su trabajo muy rápido y no alcanza el operador a soltar la mano del botón , el pistón sale de nuevo pudiendo dañar la pieza al repetir el ciclo, por lo tanto realice el circuito de protección para que no repita el ciclo.

SOLUCIÓN:

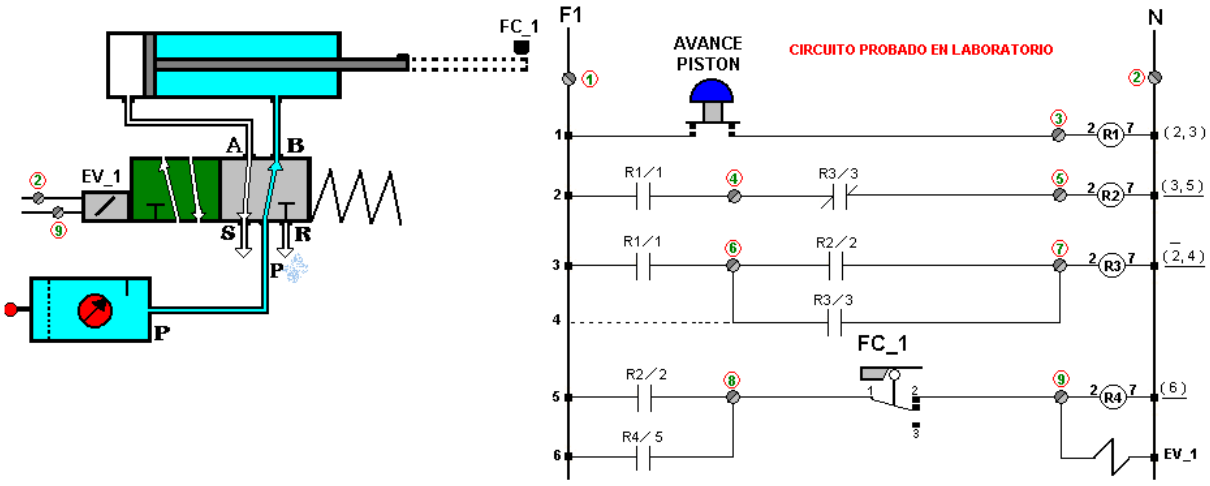


FIGURA # 3.5

EJEMPLO 3. 6:

En el circuito anterior se puede presentar el caso de que el pistón después de estar un tiempo parado tienda a bajar un poco, o que por alguna razón no llega a la posición de reposo con lo que las piezas trabajadas no tienen la calidad requerida. Diseñe el circuito de control para que si el pistón no esta en la posición de reposo no inicie el ciclo, además de que tenga la condición de que no repita ciclo.

SOLUCIÓN:

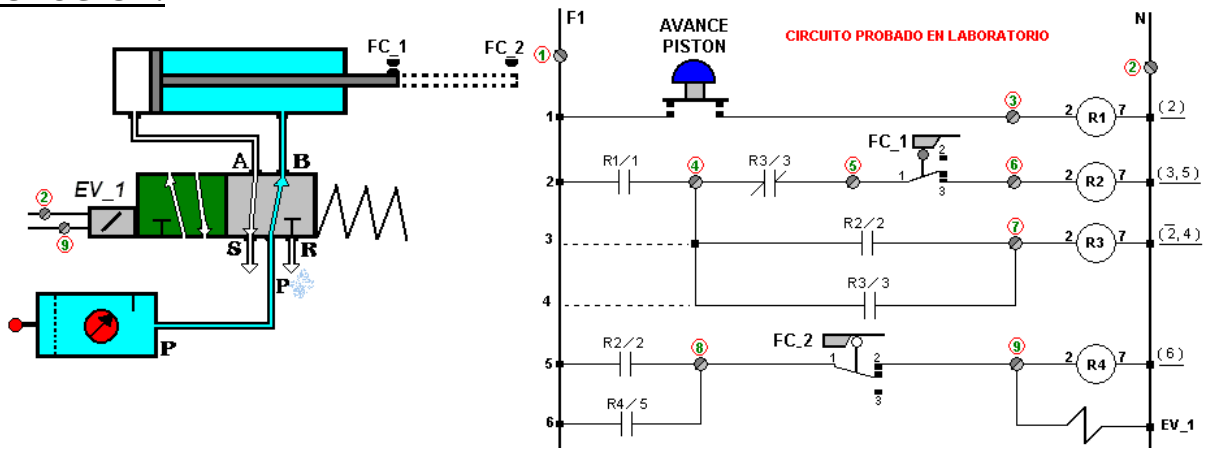


FIGURA # 3.6

EJEMPLO 3.7:

Realice el circuito de control para el ejercicio anterior pero ahora se requiere pegar dos piezas, por lo que el pistón lleva en la punta un troquel con una resistencia para calentar el troquel de tal manera que al bajar dure un tiempo abajo y luego suba en forma automática.

SOLUCIÓN:

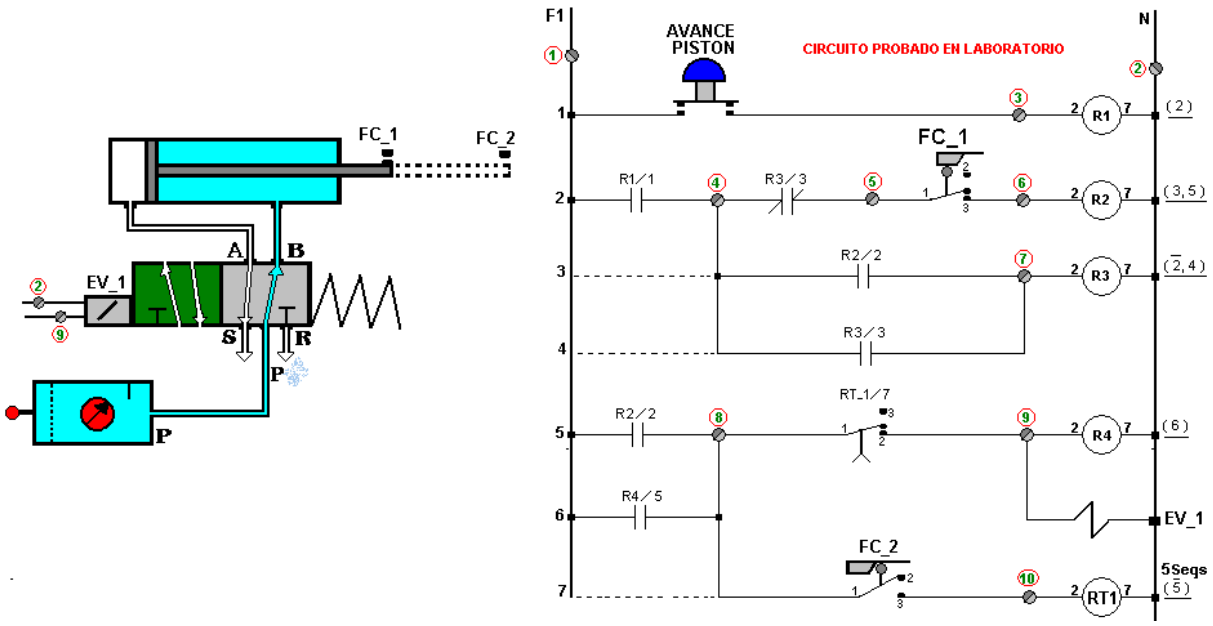


FIGURA # 3.7

EJEMPLO 3.8:

Si analizamos el problema anterior, en el cual el operador pone la pieza a pegar con una mano y con la otra da el inicio de ciclo. Imaginemos que el operador después de estar realizando esta operación durante un periodo prolongado y como usted podrá observar bastante fastidioso, comete el error de no retirar la mano derecha con la que pone la pieza y da inicio de ciclo, como el pistón baja muy rápido le sujetara la mano con el pistón que trae un troquel caliente provocándole el accidente correspondiente.

Realice la protección necesaria para evitar este problema.

SOLUCIÓN:

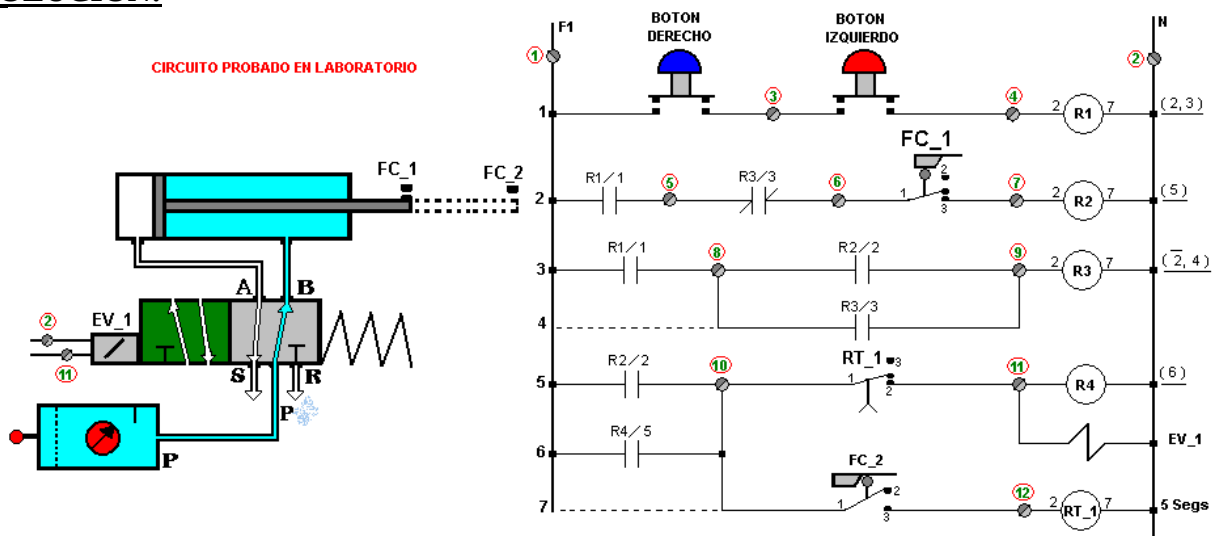


FIGURA # 3.8

EJEMPLO 3.9:

En el ejemplo anterior, se realizó la protección requerida pero, ahora, se presentó el siguiente problema, el operador tiene que estar utilizando las dos manos y éstas, separadas de la maquina como máxima protección, pero, sucede que como nadie lo supervisa y es muy cansado el estar oprimiendo los dos botones, éste (el operador) deja activado un botón con cinta adhesiva y trabaja con un sólo botón, provocando de nuevo el accidente.

Diseñe una protección para que si el operador activa un botón y después de un segundo oprime el segundo botón no se active el pistón, con lo cual nos aseguramos de que el operador utilizará las dos manos y retiradas de la máquina.

SOLUCIÓN:

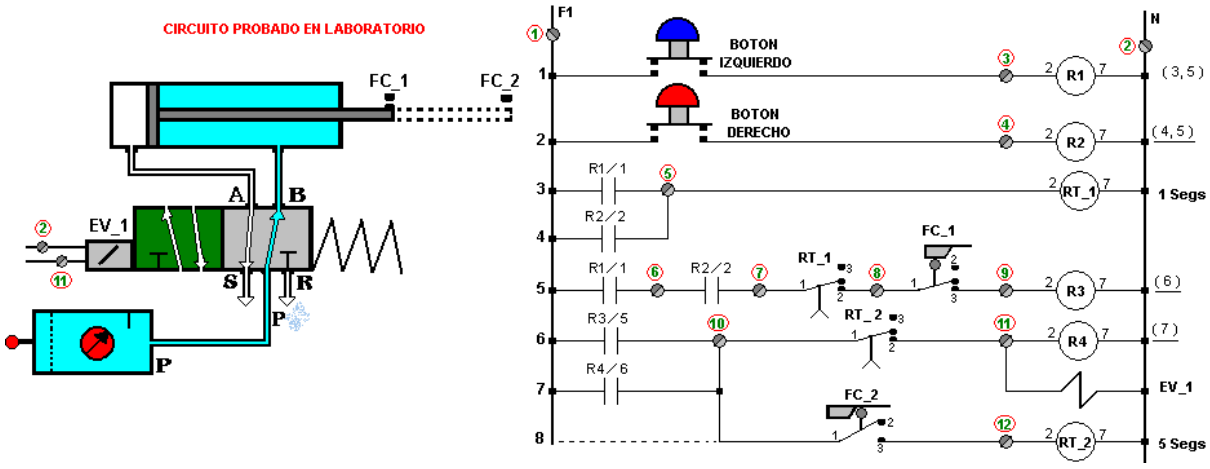


FIGURA # 3.9

EJEMPLO 3.10:

Ahora se requiere ponerle un alimentador de piezas en automático, por lo que se necesita que al llegar al reposo el pistón baje de nuevo. Para detener el circuito coloque un botón de paro de tal manera que en cualquier momento en que se oprima el botón de paro continúe el ciclo y al llegar a la posición de reposo se detenga.

SOLUCIÓN:

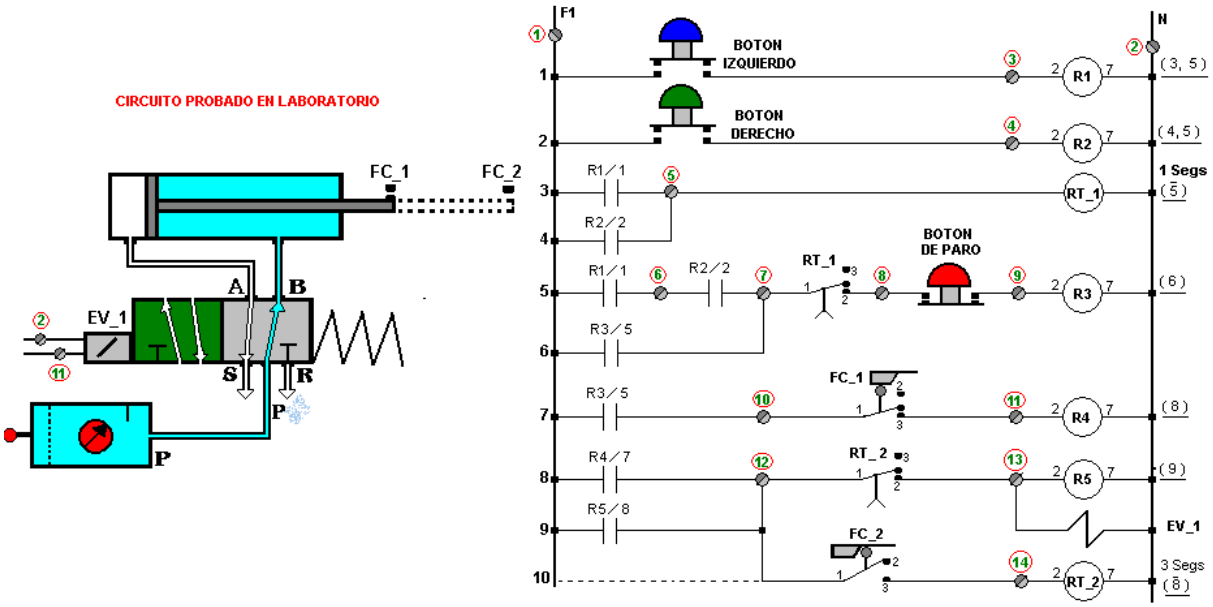


FIGURA # 3.10

EJEMPLO 3.11:

En el ejemplo del circuito anterior se puede observar que si damos el impulso de inicio de ciclo y el final de carrera FC_1 no esta activado, no iniciará el ciclo y también observamos que si se activa por alguna razón el final de carrera FC_1 iniciará el ciclo en ese instante, con lo cual puede provocar un accidente.

Diseñe el arreglo correspondiente para evitar este problema (Ver figura 3.11).

SOLUCIÓN:

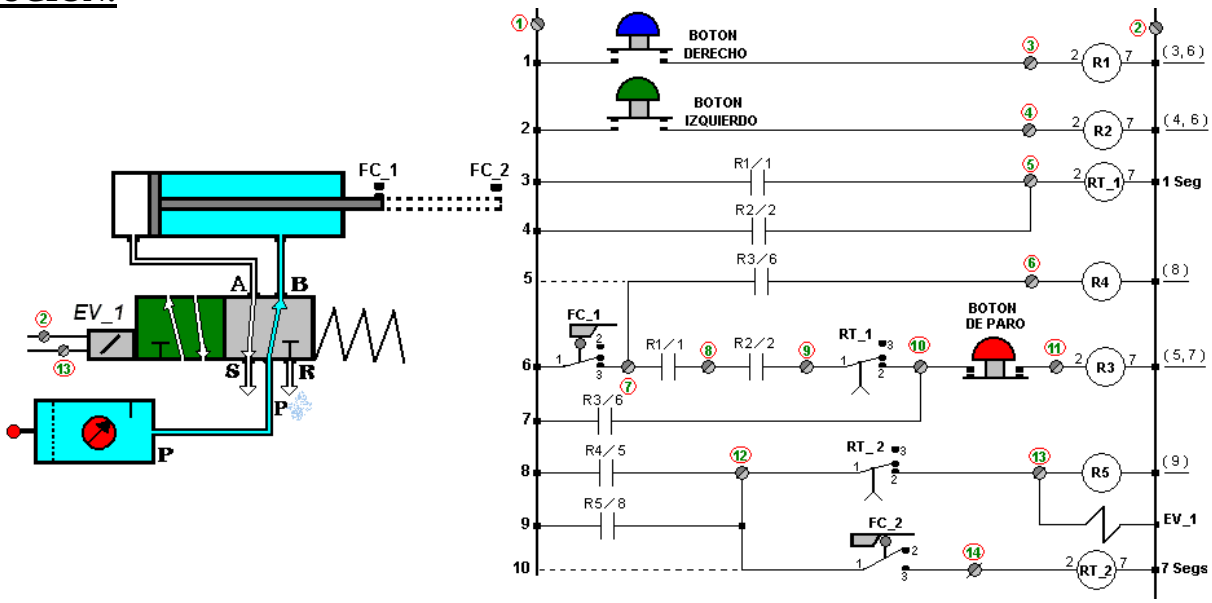
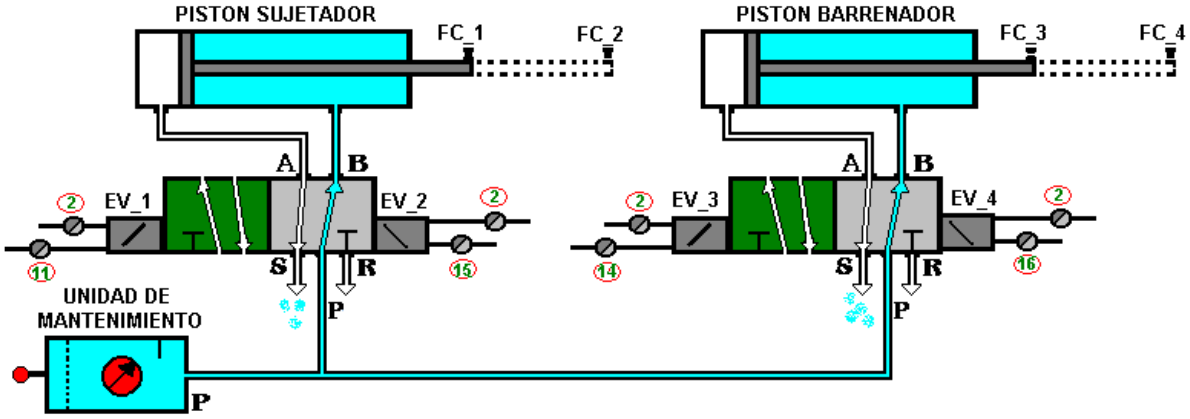


FIGURA # 3.11

EJEMPLO 3.12:

Diseñe el circuito de control electro neumático para que al dar inicio, entre un pistón a sujetar la pieza, cuando está sujeta que entre un segundo pistón a barrenar y al llegar al final de su carrera el segundo pistón se regrese y al llegar al reposo se suelte la pieza terminando el ciclo. Además, se deberá tener cuidado de que no repita ciclo y que estén en reposo ambos pistones para poder iniciar.

NOTA: Las electro válvulas tienen accionamiento eléctrico en ambos lados.



CIRCUITO PROBADO EN LABORATORIO

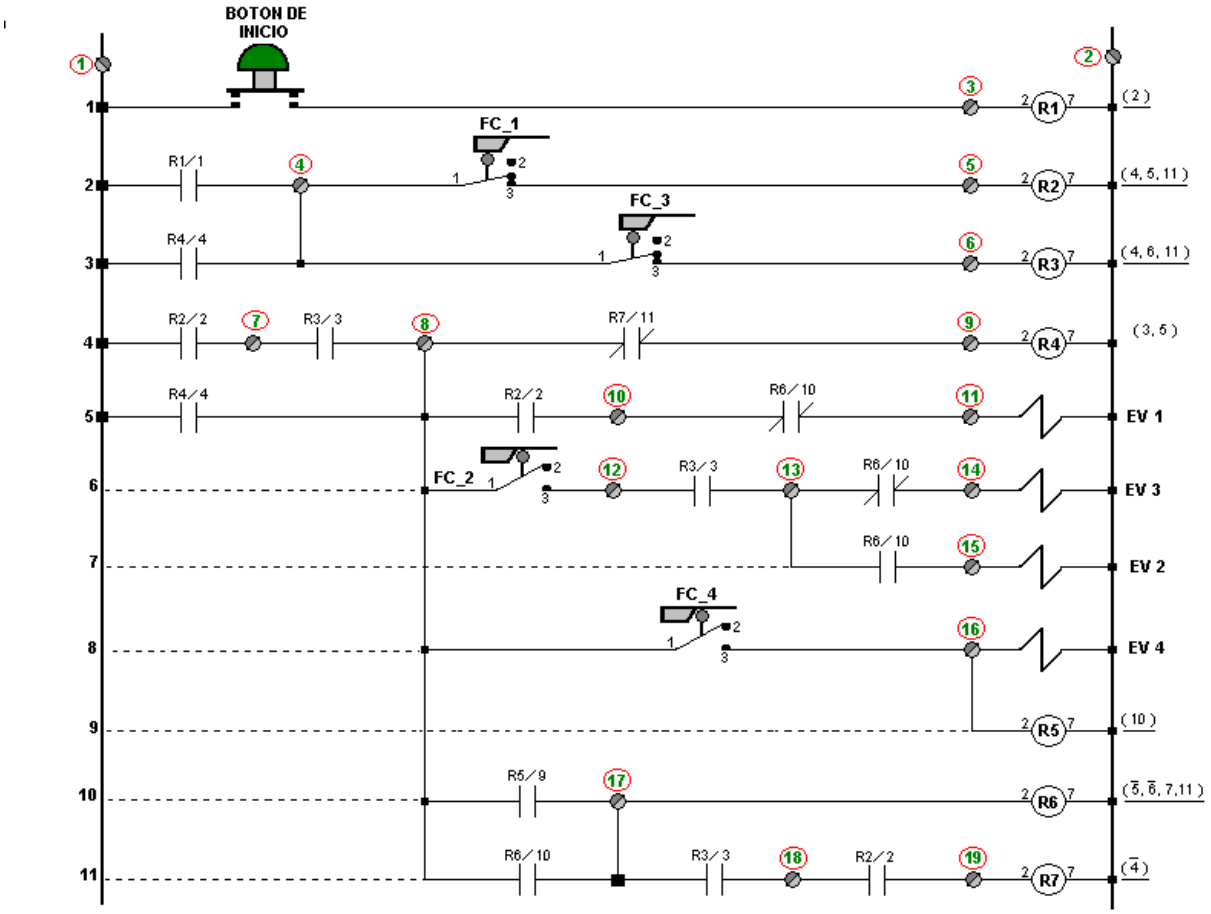
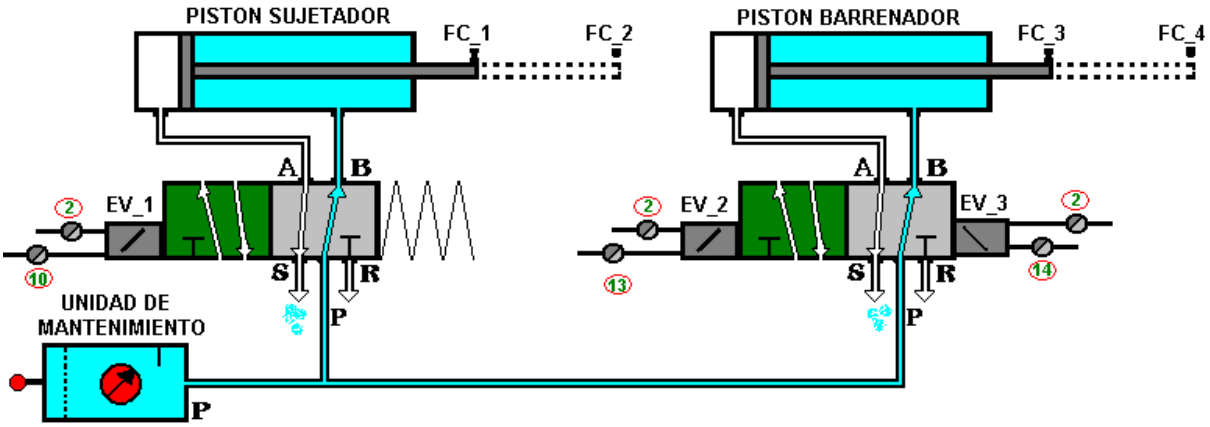


FIGURA # 3.12

EJEMPLO 3.13:

Diseñe el circuito de control electro neumático del ejemplo No 3.12 pero ahora considere que la electro válvula del pistón sujetador tiene accionamiento eléctrico y retorno por muelle.

SOLUCIÓN:



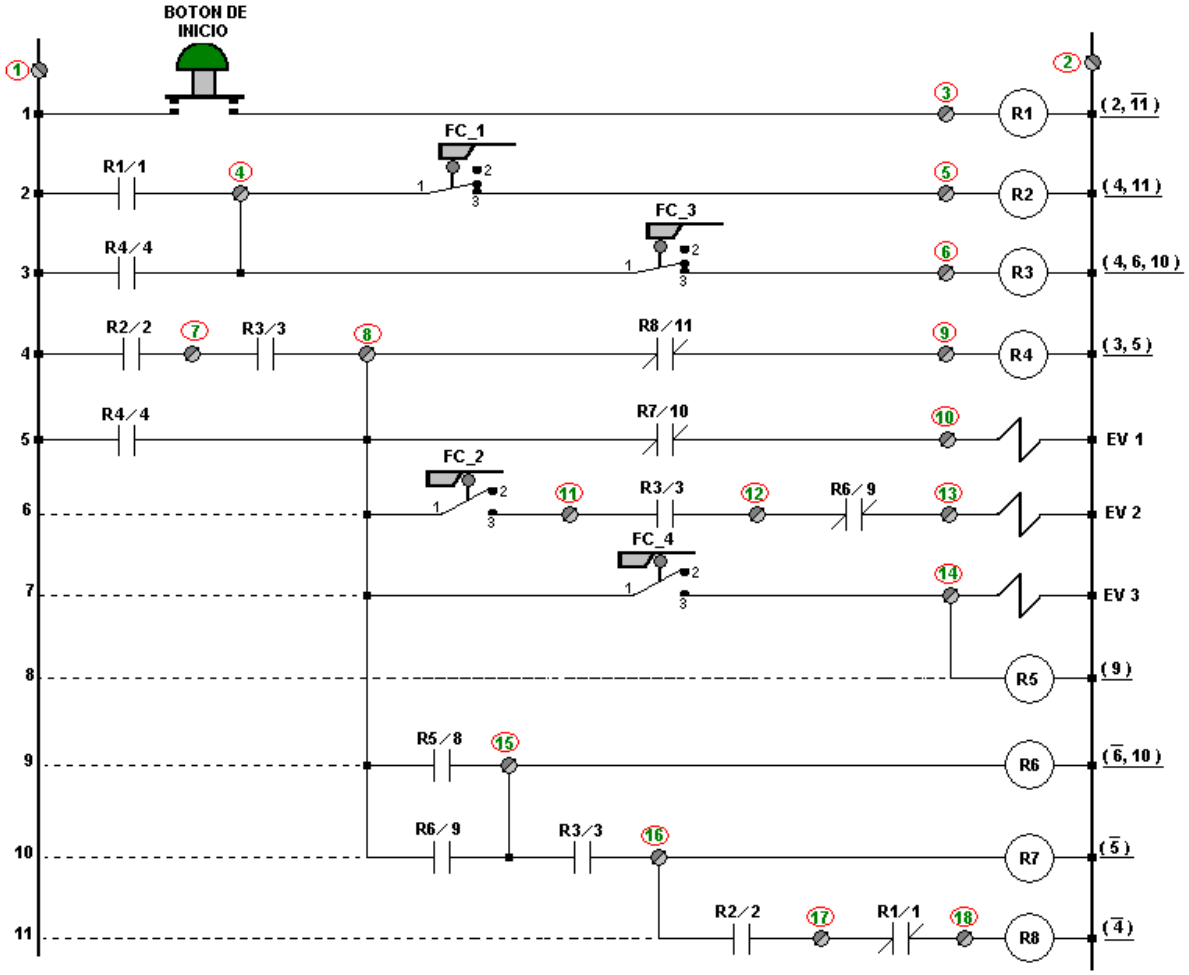
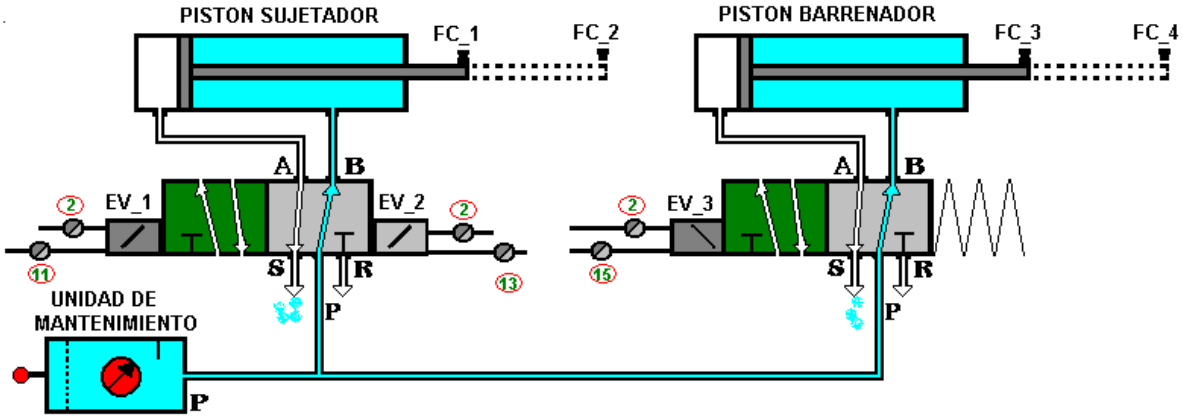


FIGURA # 3.13

EJEMPLO 3.14:

Diseñe el circuito de control electro neumático del ejemplo No 3.12 pero ahora considere que la electro válvula del pistón sujetador tiene accionamiento eléctrico en ambos lados y el pistón barrenador tiene accionamiento eléctrico y retorno por muelle (Ver figura 3.14).

SOLUCIÓN:



CIRCUITO PROBADO EN LABORATORIO

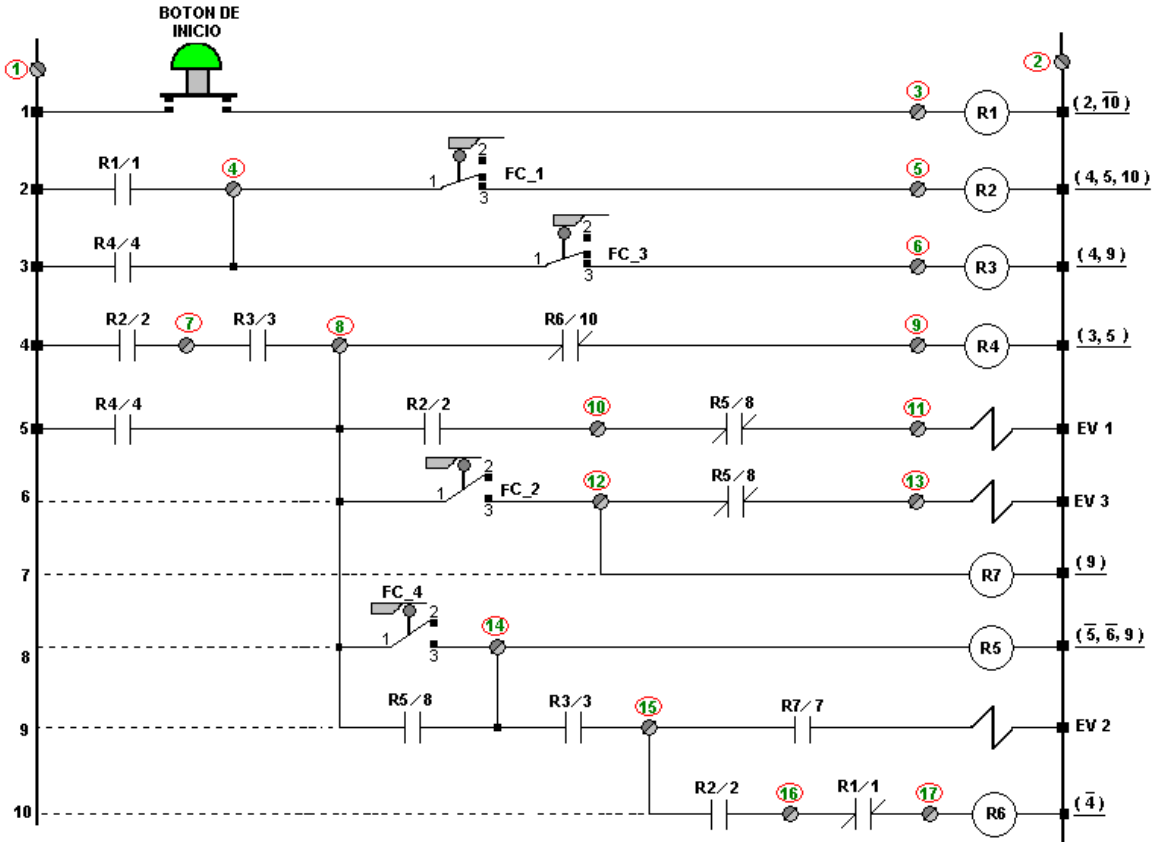
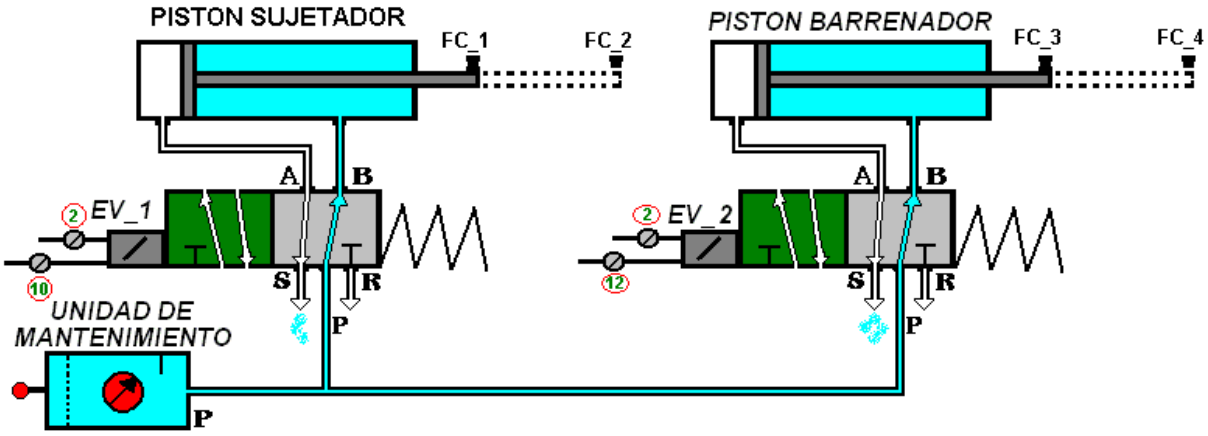


FIGURA # 3.14

EJEMPLO 3.15:

Diseñe el circuito de control electro neumático del ejemplo No 3.12 pero ahora considere que ambas electro válvulas tienen accionamiento eléctrico y retorno por muelle (Ver figura 3.15).

SOLUCIÓN:



CIRCUITO PROBADO EN LABORATORIO

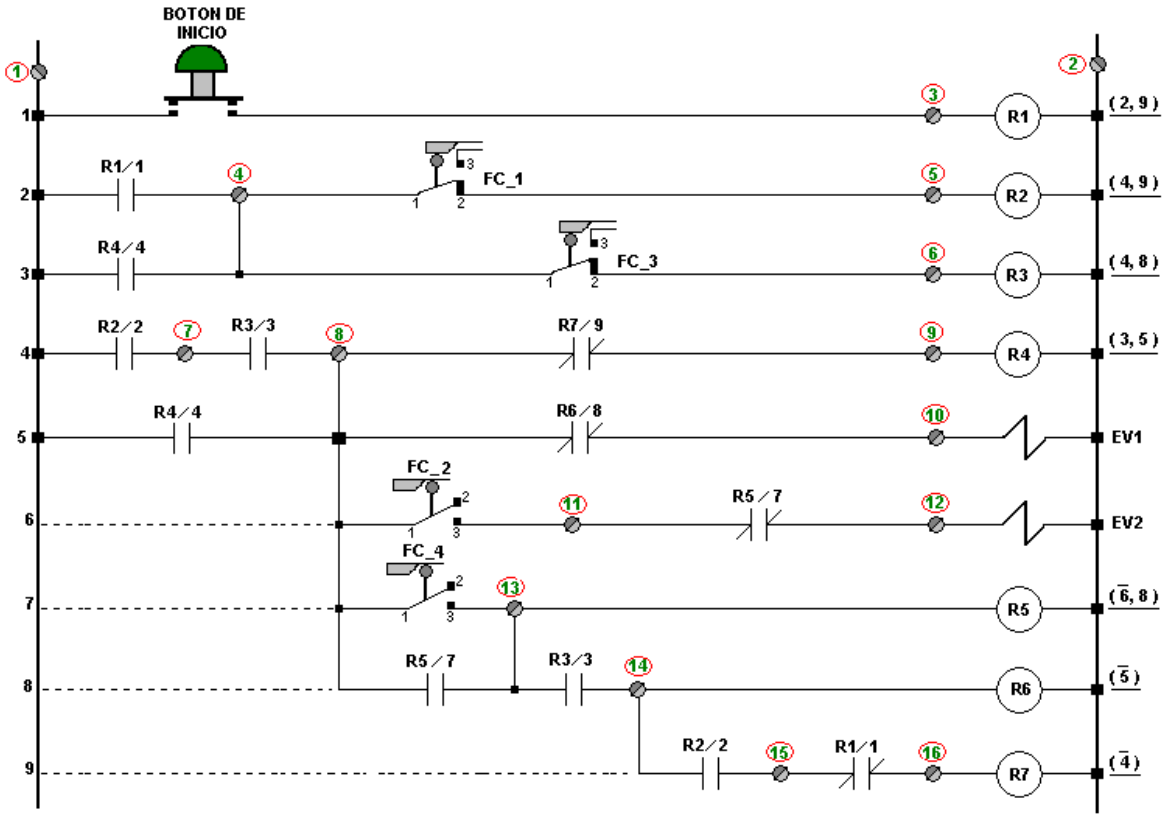


FIGURA # 3.15

EJEMPLO 3. 16:

Diseñe el circuito de control para el ejemplo No 3.12 pero ahora se desea que al dar inicio de ciclo salga el pistón sujetador y al llegar al final de su carrera se regrese y al llegar a la posición de reposo salga el pistón barrenador y al llegar al final de su carrera se regrese y cuando llegue a la posición de reposo termine el ciclo (Ver figura 3.16).

SOLUCIÓN:

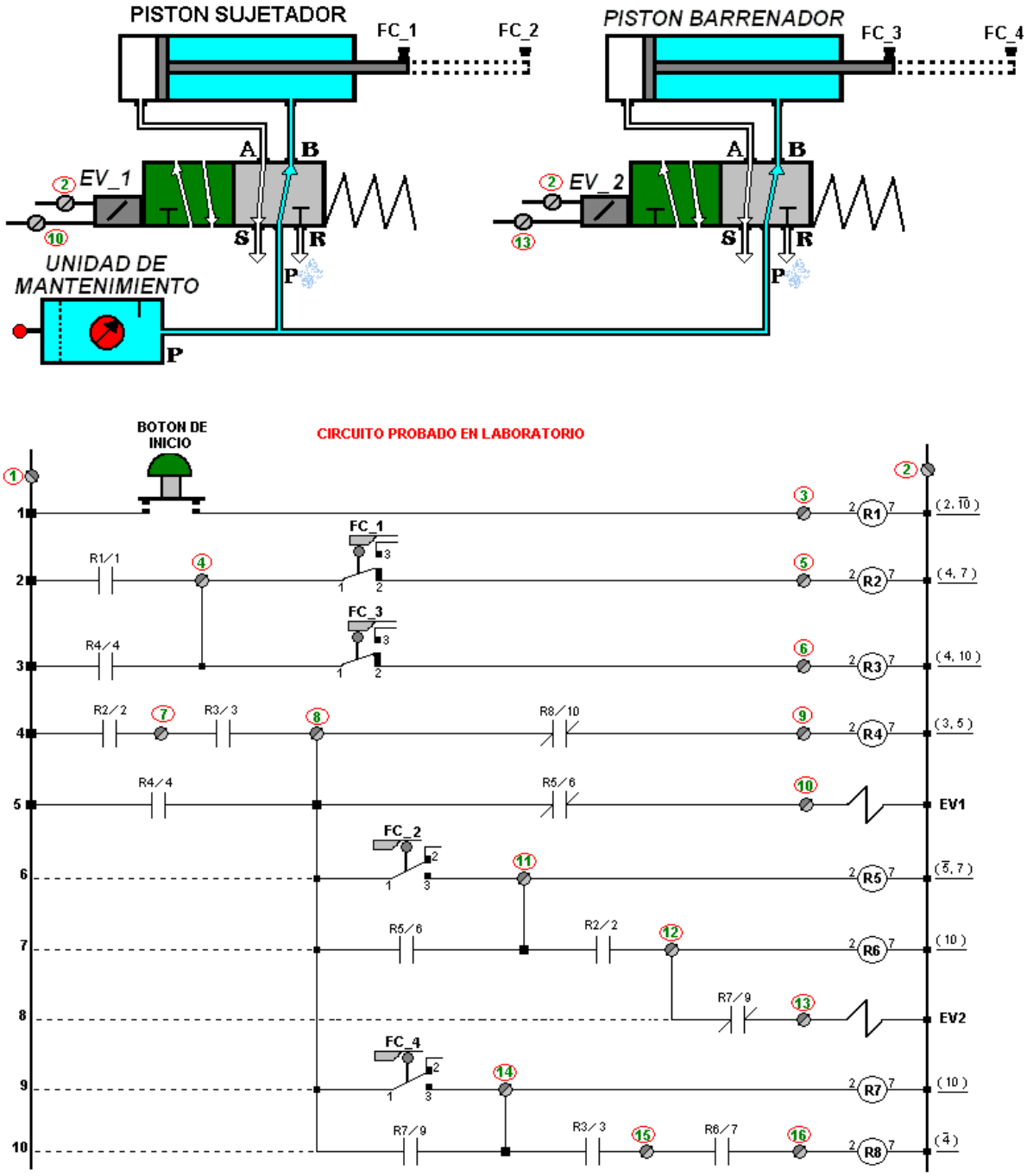


FIGURA # 3.16

EJEMPLO 3.17:

Diseñe el circuito de control del ejercicio No 3.15 pero ahora se desea que al terminar el ciclo, esto es, que al llegar el pistón sujetador al reposo dure un tiempo y reinicie el ciclo. Para detener el ciclo coloque un botón de paro y que se detenga al terminar el ciclo (Ver figura 3.17).

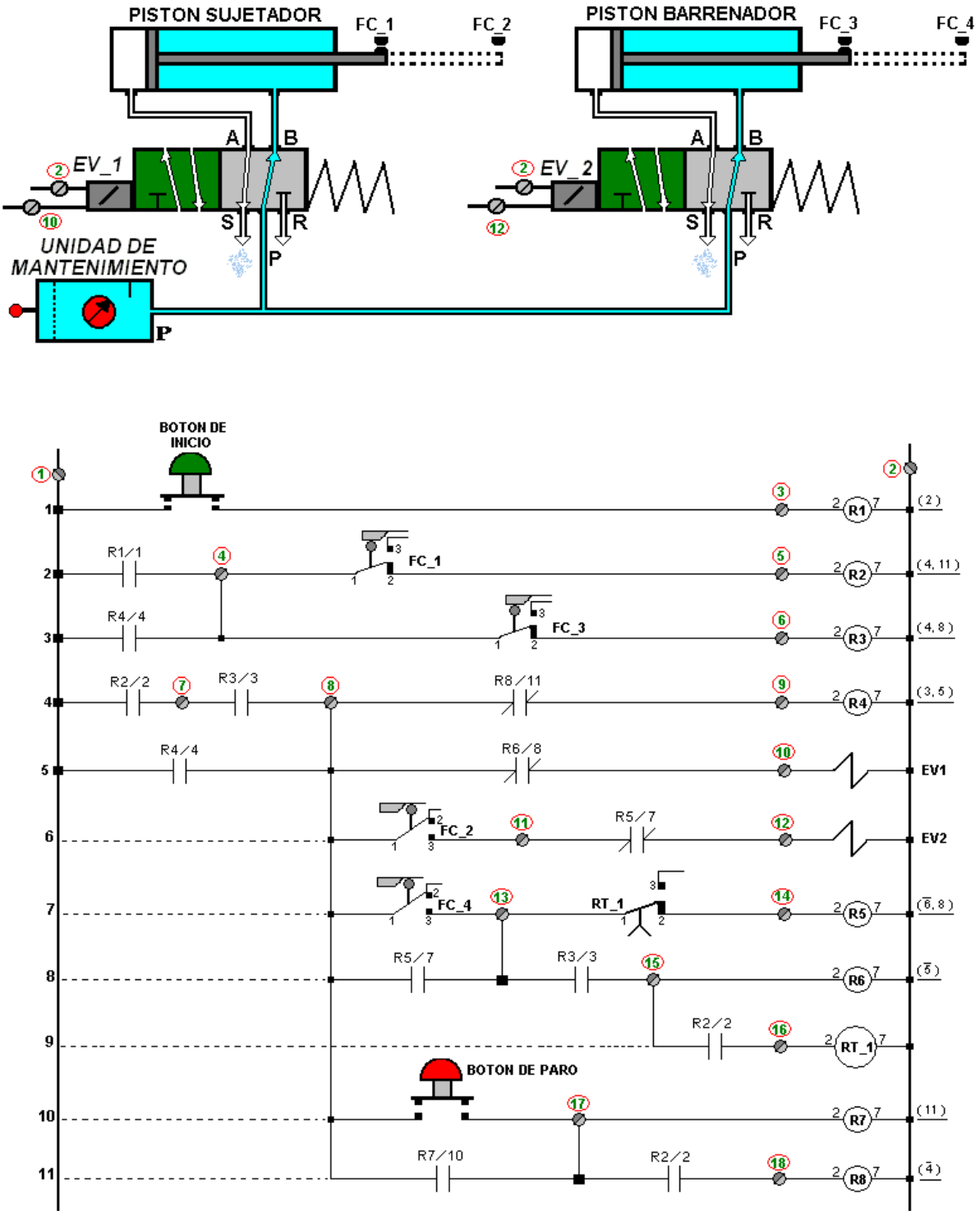


FIGURA # 3.17

EJEMPLO 3.18:

Diseñe el circuito de control para el ejercicio No 16 pero ahora se desea colocar un selector de dos posiciones; en una posición trabajara en un ciclo y en la otra posición en automático. (Ver figura 3.18).

SOLUCIÓN:

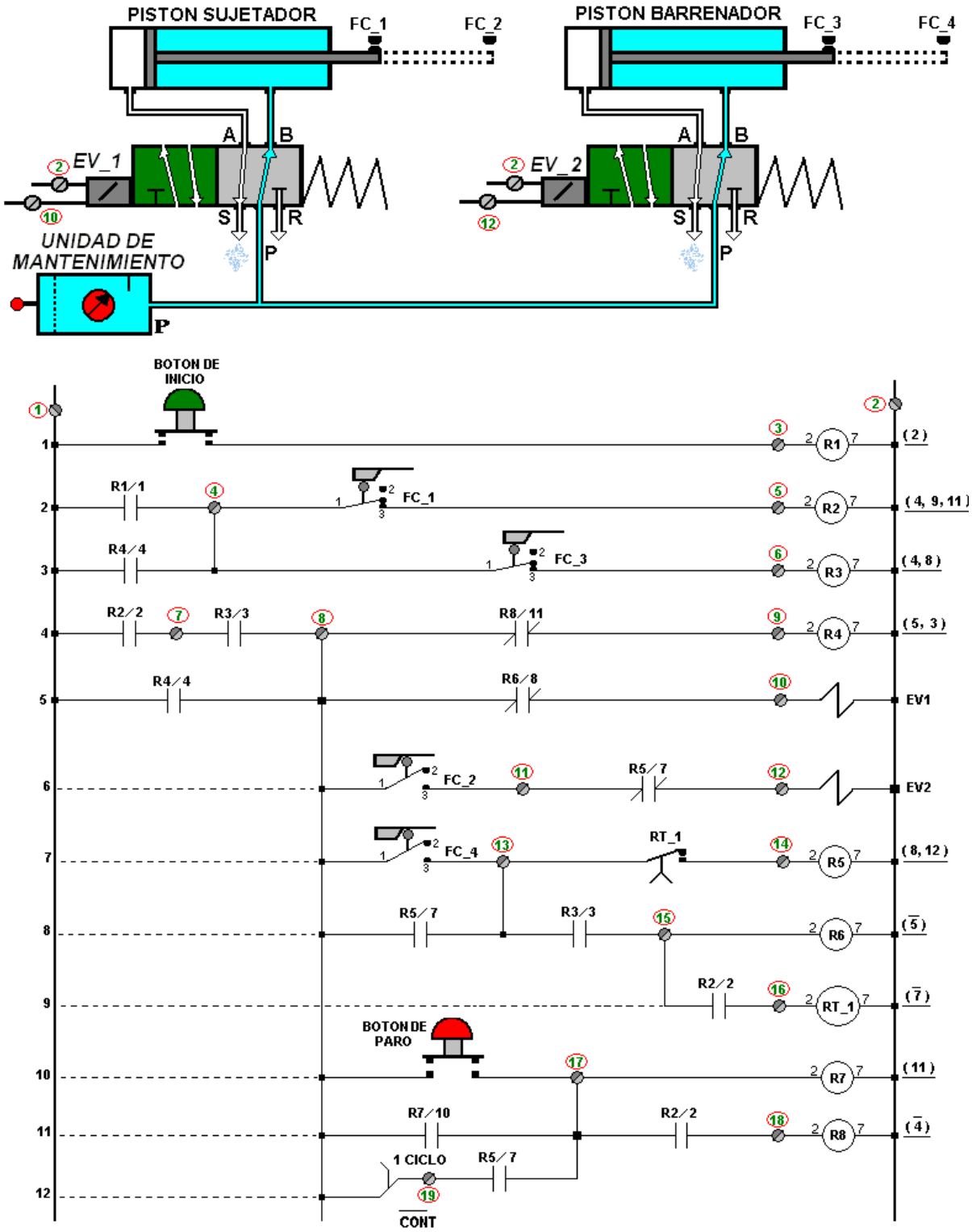


FIGURA # 3.18

EJEMPLO 3.19:

Diseñe el circuito de control electro neumático para tres pistones los cuales deberán trabajar en forma de contador Johnson, esto es, al dar inicio de ciclo sale el primer pistón y al llegar al final de su carrera sale el segundo pistón y al llegar al final de su carrera el segundo pistón, sale el tercer pistón y al llegar al final de su carrera el tercer pistón se regresa el tercer pistón y cuando llegue a la posición de reposo el tercer pistón, regresa el segundo pistón y al llegar el segundo pistón a la posición de reposo se regresa el primer pistón y finalmente al llegar a la posición de reposo el primer pistón termina el ciclo quedando todo en condiciones de iniciar de nuevo el ciclo.

NOTA: Las condiciones para dar inicio de ciclo son que los tres pistones estén en reposo.
(Ver figura 3.19)

SOLUCIÓN:

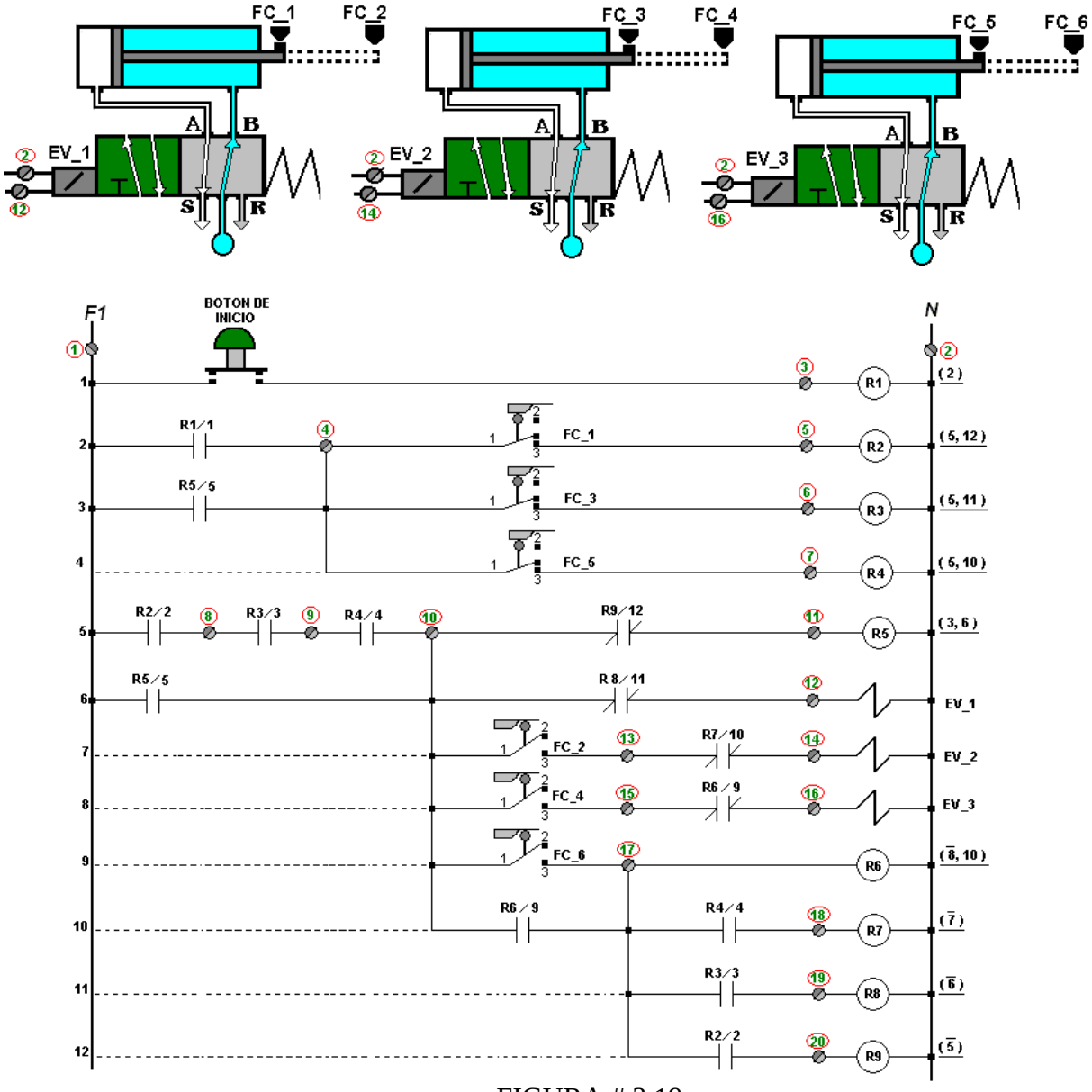


FIGURA # 3.19

EJEMPLO 3.20:

Diseñe el circuito de control electro neumático para el circuito del problema 3.19 pero ahora invertido, esto es, al dar inicio de ciclo sale el primer pistón, al llegar el final de su carrera sale el segundo y al llegar al final de su carrera el segundo sale el tercero y al llegar al final de su carrera se regresa el primero y al llegar a la posición de reposo se regresa el segundo y al llegar a la posición de reposo se regresa el tercero y al llegar al reposo el tercero se termina el ciclo. Se aplican las mismas condiciones de inicio de ciclo (Ver figura 3.20).

SOLUCIÓN:

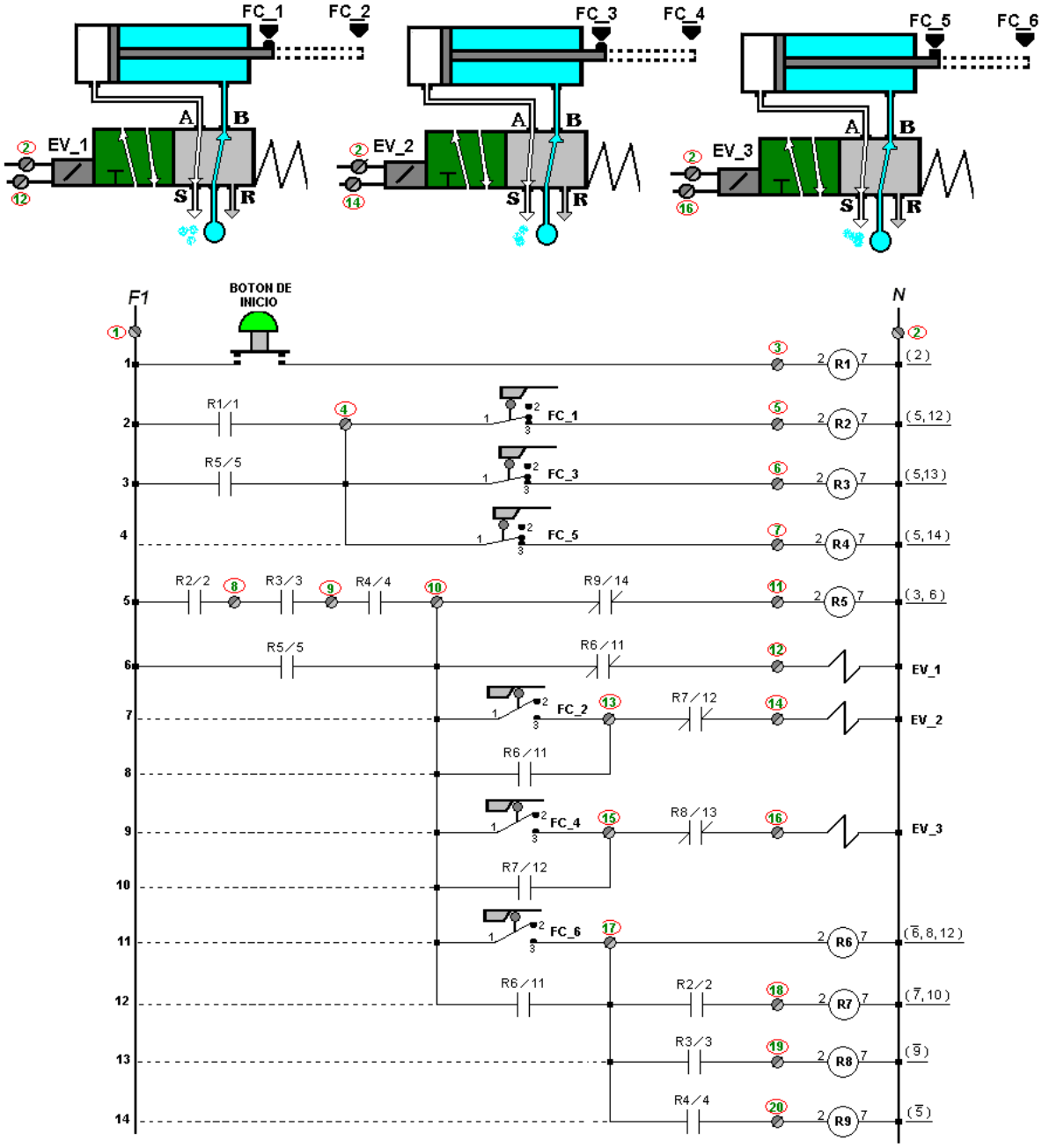


FIGURA # 3.20

EJEMPLO 3.21:

Se quiere automatizar el proceso de encorchado de botellas, para esto el operador debe colocar manualmente el corcho en la boca de la botella, una vez hecho esto se oprime el botón de inicio de ciclo y sale un pistón para sujetar la pieza, cuando la pieza esta fija, baja un segundo pistón y golpea el corcho tres veces consecutivas para que entre completamente, una vez hecho esto el pistón sujetador regresa y suelta la pieza finalizando el ciclo (Ver figura 3.21).

SOLUCIÓN:

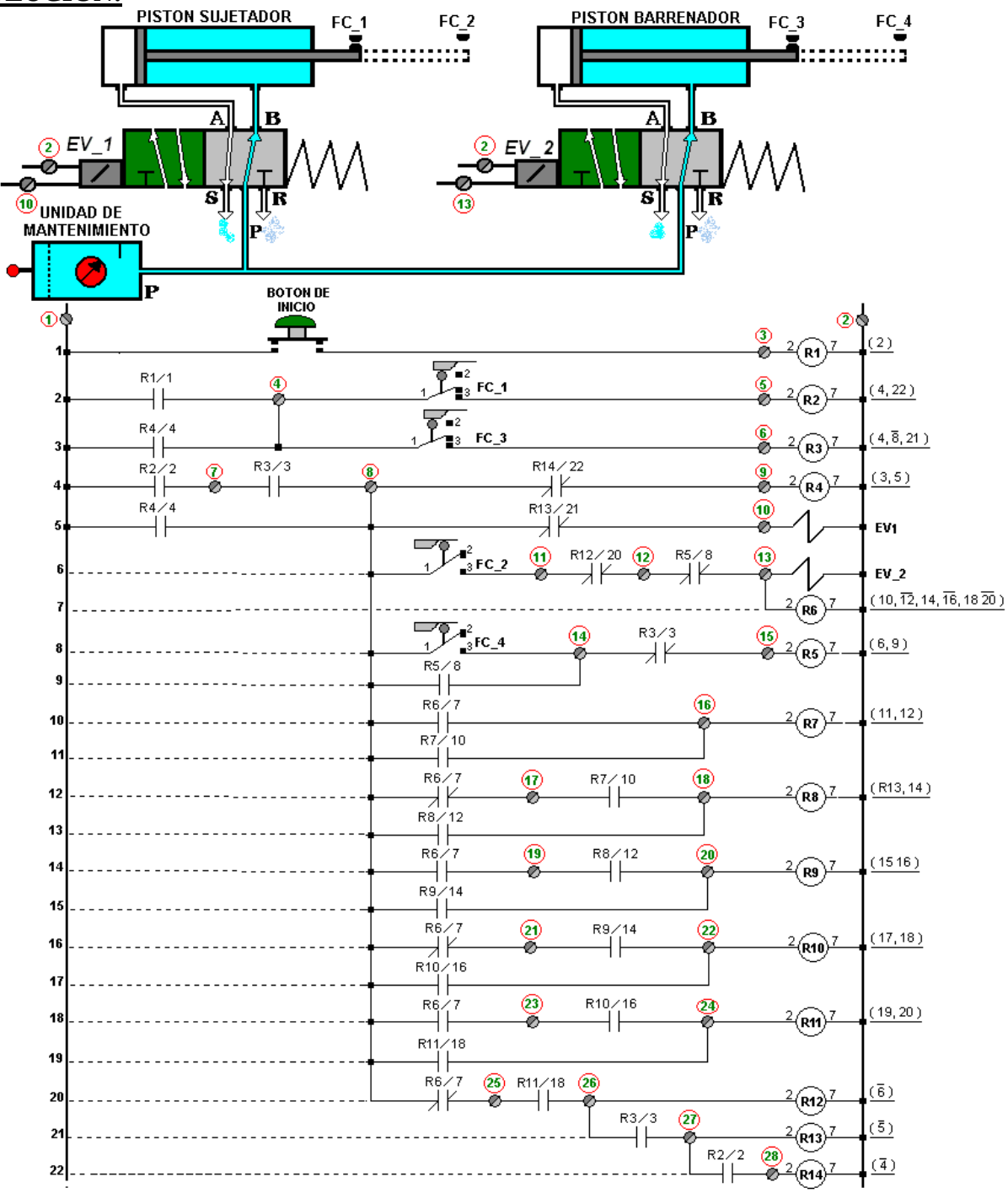


FIGURA # 3.21

EJEMPLO 3.22

Realice el circuito anterior, pero ahora se desea que al terminar el ciclo dure un tiempo para quitar la pieza y reinicie de nuevo, con esto tendremos una producción constante. Para parar el ciclo coloque un botón de paro el cual deberá parar al terminar el ciclo (Ver figura 3.22).

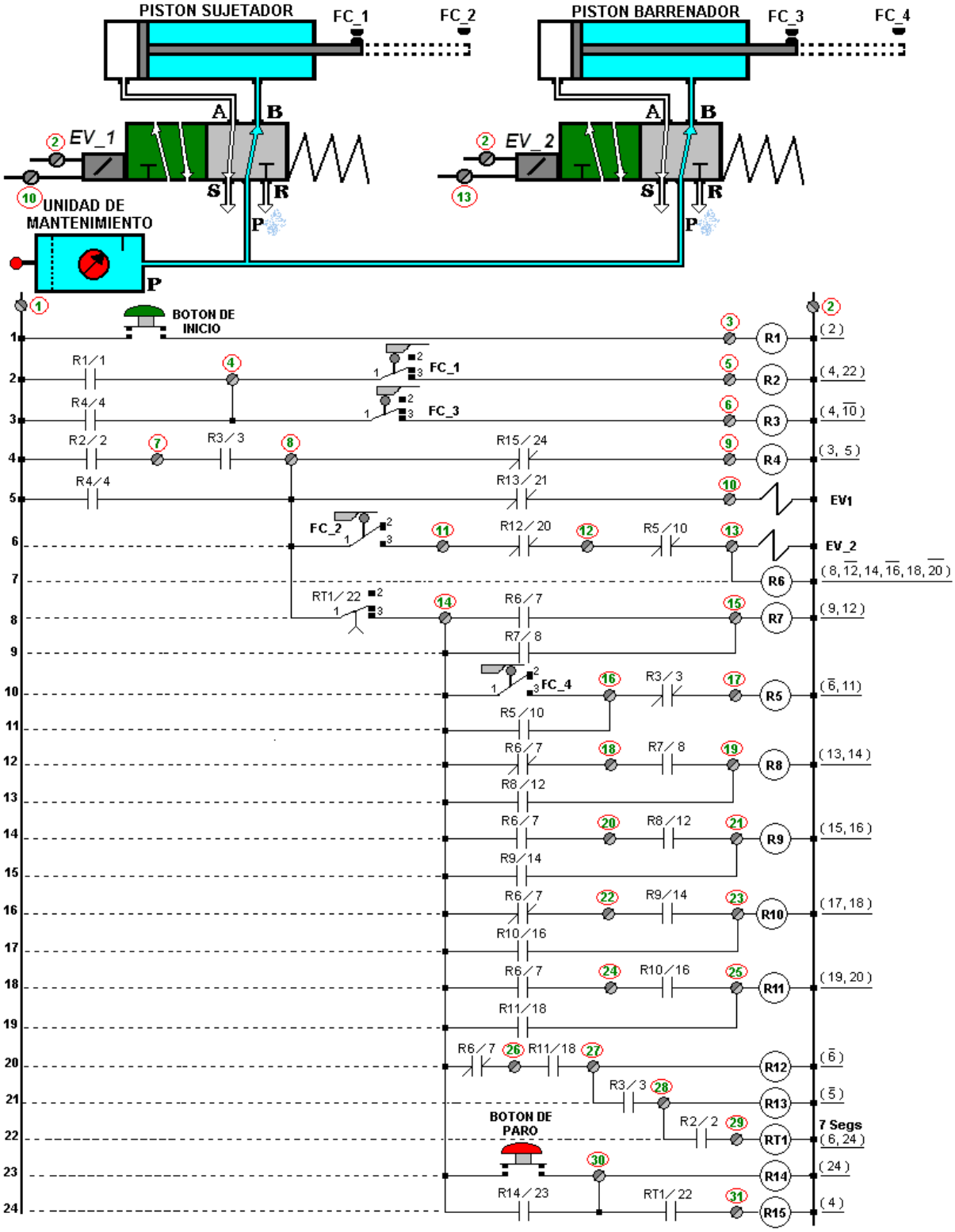
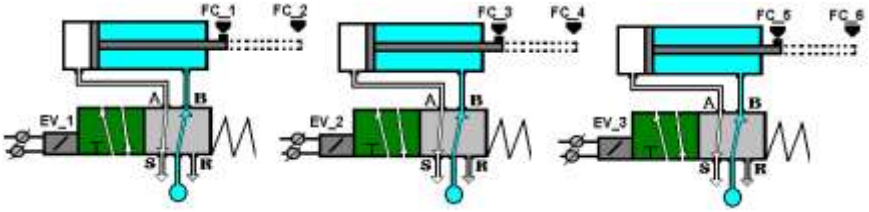


FIGURA # 3.22

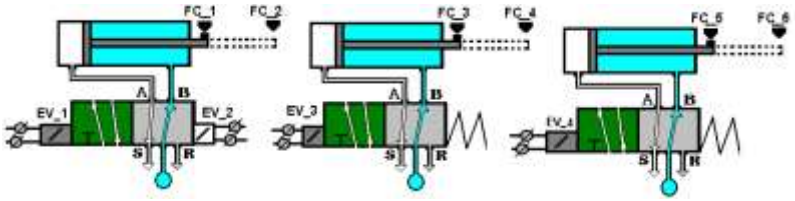
EJEMPLO 3.23a PROBLEMA PROPUESTO

Diseñe el circuito de control para los siguientes circuitos de tal manera que funcione como contador de anillo, esto es con un botón de inicio y si todos tres pisones se encuentran en reposo, que salga un pistón 1 y al llegar al final de su carera se regresa y al llega a la posición de reposo sale el pistón 2 y al llega l final de su carrera, se regresa y al llegar a la posición de reposo sale el pistón tres y al llegar al reposo sale de nuevo el pistón 1 y así sucesivamente. Para parar se requiere un botón de paro y una vez que se activa deberá terminar el ciclo, esto es, cuando regresa el pistón tres se termina el ciclo.



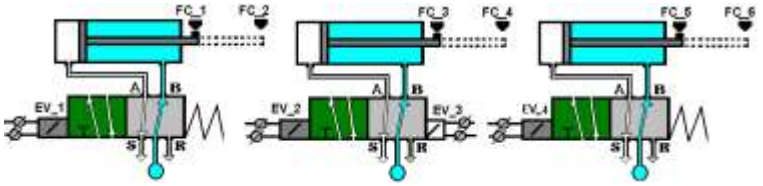
EJEMPLO 3.23b PROBLEMA PROPUESTO

El siguiente problema es idéntico al anterior pero observe como el pistón 1 tiene doble accionamiento eléctrico.



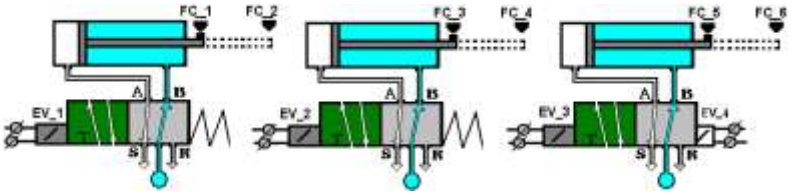
EJEMPLO 3.23b PROBLEMA PROPUESTO

El siguiente problema es idéntico al anterior pero observe como el pistón 2 tiene doble accionamiento eléctrico.



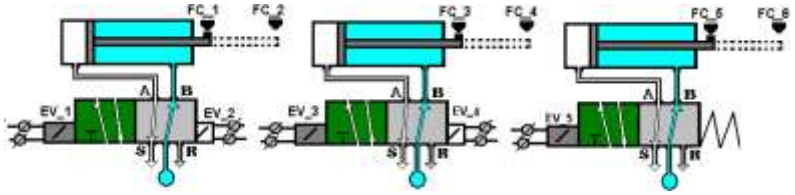
EJEMPLO 3.23c PROBLEMA PROPUESTO

El siguiente problema es idéntico al anterior pero observe como el pistón 3 tiene doble accionamiento eléctrico.



EJEMPLO 22d: PROBLEMA PROPUESTO

El siguiente problema es idéntico al anterior pero observe como el pistón 1 y el pistón 2 tiene doble accionamiento eléctrico.



CAPÍTULO 4 CONTROL PROGRAMABLE

Introducción a los P.L.C's

El desarrollo e implementación de los relevadores hace muchos años, fue el primer paso hacia la automatización, esto trajo como consecuencia incrementos en la producción. La aplicación de los relevadores hizo posible añadir lógica a la operación de las máquinas y de esta manera reducir la carga de trabajo al usuario y en algunos casos la necesidad de operadores humanos. Los relevadores hicieron posible establecer automáticamente una secuencia de operación, programar tiempos de retraso, conteo de eventos, o hacer que un evento dependa de que ocurra otro.

Los relevadores con todas sus ventajas, tienen también sus desventajas. Tienen un período de vida limitado (especificado por el fabricante), su naturaleza electromecánica dictamina que después de un tiempo de uso, éstos serán inservibles. Sus partes conductoras de corriente pueden desgastarse (falsos contactos), quemarse (abrirse) o fundirse (pegarse) en un momento dado, modificando la secuencia establecida y requiriendo su reemplazo.

Podemos decir con seguridad que el inconveniente más importante con la lógica de los relevadores, es su naturaleza fija. La lógica de un tablero de control construido a base de relevadores es realizada por los ingenieros o técnicos de diseño, implementándola y colocando los relevadores en la platina (base del tablero) de control y realizando el sistema de cableado de acuerdo a un diseño previo.

Mientras que la máquina controlada por el tablero con relevadores continúe llevando a cabo los mismos pasos y en la misma secuencia todo esta bien, pero cuando se requiera realizar una modificación en la secuencia de operación y/o realizar un cambio en el proceso de producción, la lógica del tablero debe ser rediseñada (recableado).

Sí el cambio es muy grande, una opción más económica puede ser desechar el panel actual y construir uno nuevo.

El problema anteriormente descrito fue el mayor que enfrentaron los productores de automóviles a mediados de los años sesentas. Con el transcurso de los años se habían automatizado altamente las operaciones de producción mediante el uso de relevadores. Con los cambios de modelos anuales que se tenían, el proceso y la secuencia cambiaba, y por lo tanto se requería un reemplazo total del cableado, además de que el tiempo de vida de los relevadores llegaba a su límite por lo que se veían en la necesidad de poner equipo nuevo en cada cambio de modelo con lo cual se invertía una gran cantidad de trabajo y material sin tomar en cuenta la gran cantidad de tiempo de producción perdido.

Las condiciones estaban dadas para la creación de un dispositivo electrónico capaz de solucionar estos problemas. En estos tiempos (años sesentas) apareció el microprocesador con lo cual les dio la idea a los fabricantes de automóviles, principalmente a la compañía General Motor, de que la clase de control que ellos requerían podrían llevarse a cabo con estos dispositivos mediante el uso de una computadora. Las computadoras en si mismas no eran deseables para aplicaciones industriales por muchas razones. Por lo que se plantearon los requerimientos básicos para el desarrollo de este nuevo producto de uso industrial, las cuales fueron las siguientes:

- Todos los componentes del sistema debían ser capaces de operar en un ambiente industrial sin necesidad de algún otro equipo de soporte, de hardware o de ambiente.

- Contaría con memoria interna para almacenar un programa de control y datos. La memoria no perdería el programa aunque al dispositivo de control le faltara el voltaje de alimentación.

- El dispositivo de control deberá ser fácil y rápidamente programable y reprogramable por el usuario con un mínimo de interrupción de su servicio.

- El sistema debe ser de fácil mantenimiento y reparación. Deberá diseñarse con indicadores de status y modularidad para facilitar las reparaciones y la búsqueda de errores.

- El sistema deberá ocupar menor espacio que los sistemas de relevador y deberá consumir menor potencia que los sistemas de control por relevadores.

- El PLC deberá ser capaz de comunicarse con un sistema central de datos para propósitos de monitoreo.

- Deberá ser capaz de trabajar con 120Volts de corriente alterna y con elementos estándar de control, con interruptores de presión, interruptores de límite, etc.

- Las señales de salida deberán ser capaces de energizar arrancadores para motores y válvulas solenoides que operen a 120 Volts de C.A.

- Deberá ser expandible desde su mínima configuración hasta su máxima, con un mínimo de alteración y de tiempo perdido.

- Deberá ser competitivo en costos de venta e instalación, respecto de los sistemas de base a relevador.

- La estructura de memoria empleada deberá ser expandible a un mínimo de 4000 palabras o elementos de memoria.

La ingeniería electrónica estaba frente a un gran reto: Primero, diseñar un equipo que, como una computadora pudiese efectuar el control y también ser fácilmente reprogramado pero adecuado al ambiente industrial. Segundo, diseñar un equipo que tomara en cuenta la experiencia de los especialistas en control electromecánico pero con equipo electrónico. El reto fue enfrentado y a finales de los años sesentas se entregó el primer dispositivo, que le daría forma a lo que hoy se le conoce como Controlador Lógico Programable (PLC), en las plantas ensambladoras de automóviles de Detroit, en los Estados Unidos. Los PLC's entraron al mercado europeo dos años después.

Actualmente cada fabricante de PLC's excede con mucho estos requerimientos básicos y lo han convertido en una alternativa de gran versatilidad y seguridad para los sistemas industriales a base de control electromecánico.

Las funciones de relevador son muy variadas: Lógica, secuencia, manejo de datos, control de temperatura, presión, nivel, flujo, temporización y conteo.

Trabajan en ambientes de temperatura muy altos para equipos normales de control (60 grados centígrados) y niveles de humead del 95% sin condensación. Generalmente no requieren especificaciones especiales de instalación. Cada elemento de control esta diseñado para aplicaciones industriales.

Los PLCs actuales no solamente cumplen estos requisitos sino que los superan. El PLC actual es una computadora de propósito específico que proporciona una alternativa más flexible y funcional para los sistemas de control industriales. La figura siguiente nos muestra en una forma general las funciones básicas de un PLC.

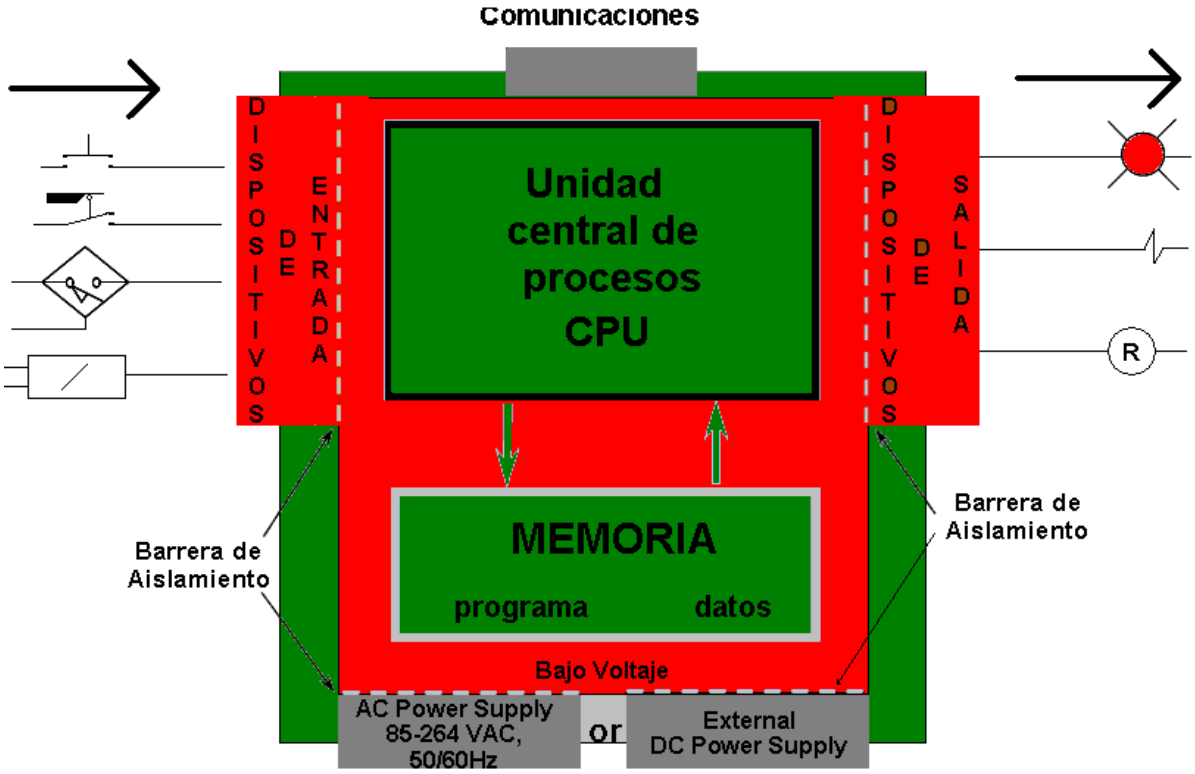


Figura 4.1

Debido a la gran aceptación que ha tenido el PLC, se ha dado una definición formal por la NEMA (National Electrical Manufacturers Association) descrita como sigue:

"El PLC es un aparato electrónico operado digitalmente que usa una memoria programable para el almacenamiento interno de instrucciones, las cuales implementan funciones especificas tales como lógicas, secuenciales, temporización, conteo y aritméticas, para controlar a través de módulos de entradas y salidas digitales y analógicas, varios tipos de máquinas o procesos. Una computadora digital que es usada para ejecutar las funciones de un controlador programable, se puede considerar bajo este rubro. Se excluyen los controles secuenciales mecánicos".

De una manera más general podemos definir al Controlador Lógico Programable a toda máquina electrónica, diseñada para controlar en tiempo real y en medio industrial procesos secuenciales de control. Su programación y manejo puede ser realizado por personal con conocimientos eléctricos o electrónicos sin previos conocimientos sobre informática.

También se le puede definir como "una caja negra" en la que existen unos terminales de entrada a los que se les conectarán pulsadores, finales de carrera, foto celdas, sensores, etc.... unos terminales de salida a los cuales se les conectarán bobinas de los contactores para motores, focos pilotos, electro válvulas etc... de tal forma que la actuación de estos últimos está en función de las señales de entrada que estén activadas en cada momento, según el programa almacenado. Esto quiere decir que los elementos tradicionales como relevadores auxiliares, relevadores de enclavamiento, temporizadores, contadores... son internos. La tarea del usuario se reduce a realizar el programa que no es más que la relación entre las señales de entradas que se tienen que cumplir para activar cada salida.

Los PLC's los podemos dividir para su estudio en cuatro partes importantes:

1.- Unidad central de procesos.
2.- Módulos de entradas y salidas.
3.- Interfase de comunicación.
4.- Fuente de alimentación.

4.1.- La unidad central de procesos (CPU)

El procesador o unidad central de procesos con sus siglas en ingles CPU es el cerebro del controlador. El hardware del PLC está constituido por un microprocesador, circuitos de memoria y circuitos periféricos. Gracias al microprocesador, el PLC puede ejecutar una serie de instrucciones (programa) en un tiempo muy corto (milésimas de segundo), realizar operaciones aritméticas y lógicas, simular dispositivos de campo como temporizadores, contadores, programadores cíclicos; realizar transferencia de información entre el sistema de entrada / salida y la memoria, así como entablar comunicación con el usuario por medio de terminales de programación y de datos o bien con otros dispositivos inteligentes. Con el avance de la tecnología de los semiconductores y el desarrollo de las circuitos impresos, los CPU's son cada vez más compactos, más rápidos y con más opciones.

4.2.- Memorias

Una vez que un programa o lista de instrucciones se ha introducido en el PLC, éste se guarda en la memoria del CPU hasta que es modificado por el usuario. El programa grabado en la memoria no se borra, debido a que el PLC cuenta con un compartimiento para colocar una batería que abastece de energía a la memoria todo el tiempo evitando así que ésta se pierda. El tiempo de vida de la batería es limitado pero puede durar de 5 a 8 años en condiciones normales de uso.}

La memoria del PLC se puede encontrar en cuatro diferentes versiones. RAM, PROM, EPROM Y EEPROM. (Ver figura 4.1).

La memoria RAM (Memoria de Acceso Aleatorio) es de lectura y escritura, pero es volátil; es decir que al faltarle el voltaje de alimentación, ésta pierde toda la información que tenía almacenada; aunque una característica importante de esta memoria es que puede trabajar a velocidades más altas que las otras memorias. Es por esto que se utilizan como memoria de

almacenamiento del programa del PLC y los datos que este va almacenando. La memoria RAM se puede grabar, leer y borrar cuantas veces sea necesario y el PLC esta diseñado para hacerlo, ya sea por medio del programador de bolsillo o por medio de una computadora con el software adecuado.

La memoria PROM (Memoria Programable de solo lectura) es de solo lectura y no es volátil; es decir que se programa de fábrica por primera y única vez y posteriormente solo se puede leer. No necesita un voltaje de alimentación para asegurar la información que se graba. Este tipo de memoria se utiliza para grabar el programa inicial de arranque del PLC; a este programa se le conoce como el BIOS del PLC y es fundamentalmente para el funcionamiento de mismo. Gracias al BIOS nosotros podemos darnos cuenta si el PLC está trabajando (RUN) o esta detenido (STOP.), si se detectó una falla o trabaja normalmente, si la batería de respaldo de la memoria RAM está baja, si existe comunicación con otro dispositivo, etc.

Las memorias EPROM y EEPROM son de lectura y escritura y no son volátiles. La diferencia entre ambas está en que la memoria EPROM se puede borrar y grabar solo con aparatos especiales (borrador de rayos ultravioleta y programador de EPROM'S), mientras que la memoria EEPROM no necesita de ningún dispositivo especial para ser borradas y grabadas, el PLC que las utiliza está capacitado para hacerlo, esto es se pueden borrar eléctricamente. Cualquiera de éstos dos dispositivos sirven para conservar una copia del programa que se encuentra en la memoria RAM del PLC y de esta manera el usuario se asegura de conservar el programa por tiempo indefinido, y utilizarla en el caso de que el programa que se encuentra grabado en la memoria RAM sufra alteraciones. En algunos PLC's, el BIOS se encuentra grabado en la memoria EPROM.

4.3.- Módulos de entradas y salidas.

Los módulos de entradas y salidas juegan un papel importante en la estructura del PLC; sirven de enlace entre el mundo exterior y el CPU. El procesador conoce el estado físico y actúa sobre los dispositivos instalados en el campo, gracias a los módulos. Existen actualmente un número muy grande de dispositivos que le pueden mandar información al PLC para su proceso y control, están por ejemplo los sensores de posición, presión, flujo, humedad, PH, temperatura, nivel, etc. Y todos ellos pueden enviar al PLC una señal eléctrica diferente que el módulo se va a encargar de traducir para que el PLC la pueda entender y procesar. De la misma manera existe una gran variedad de actuadores sobre los cuales el PLC puede tener control; alarmas sonoras, electro válvulas, arrancadores de motores, luces pilotos, etc, y que el módulo de salida se encarga de traducir en señales eléctricas generadas por el PLC a niveles entendibles por cada actuador.

Los módulos de entrada y salida ofrecen también aislamiento eléctrico entre el PLC y el campo, evitando con esto daño interno al controlador por causa de disturbios eléctricos en el campo: el voltaje de aislamiento es normalmente de 1,500 VDC.

Existen dos tipos de señales eléctricas que los módulos pueden manejar tanto de entrada como de salida: señales discretas y señales analógicas. La señal discreta es aquella señal cuyo valor esta entre dos estados: encendido o apagado, lleno o vacío, arriba o abajo, presente o ausente, etc,. Y que eléctricamente para el PLC se traduce en presencia o ausencia de voltaje. La señal analógica, a diferencia de la discreta, puede tener un valor determinado dentro de muchos valores posibles (rango).

Las señales eléctricas que manejan los módulos pueden ser tanto de alterna (AC) como de directa (DC); entre las señales de AC más comunes se encuentran 120VAC y VDC, la señal de DC

más común es de 24VDC tanto de lógica positiva como de lógica negativa. Esto quiere decir que el módulo puede entregar o recibir (según sea de entrada o de salida), voltaje (24VDC, 120VAC o 220VAC) o ausencia de voltaje (línea común de la fuente o neutro).

Una característica importante de los módulos de salida es su capacidad para conectarse a cargas inductivas o capacitivas. Sí nuestra carga es de corriente directa (lo más común es a 24V) el módulo esta diseñado con salidas a transistor, con lo cual tenemos un rango de vida mayor y una alta velocidad de conmutación, si la salida es de corriente alterna (lo más común es 110 o 220v) el módulo esta diseñado con salidas a triac, con lo cual tenemos una gran velocidad y durabilidad por ser de estado sólido, cuando en campo tenemos tanto señales de DC y AC y conmutación de señales para otros equipos como variadores de velocidad, se encuentran módulos con salidas de contacto por relevador.

Para los módulos analógicos existen rangos estándares para la recepción y transmisión de señales analógicas, por ejemplo para el voltaje tenemos rangos de 0 a 5 VDC, -5 a +5 VDC, -10 a + 10 VDC. Para corriente existen rangos de 0 a 20 mA de DC, de 4 a 20 mA de DC y de –20 a + 20 mA de DC.

A continuación presentamos las conexiones típicas para los diferentes voltajes de entrada presentados anteriormente, primeramente presentamos un circuito típico de entrada a 24Vdc lógica positiva y negativa.

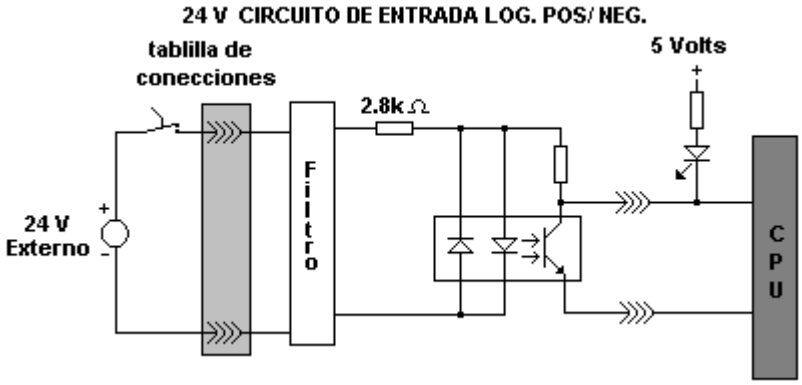


FIGURA # 4.3.1a

En seguida presentamos un circuito típico de entrada a 117 Vac.

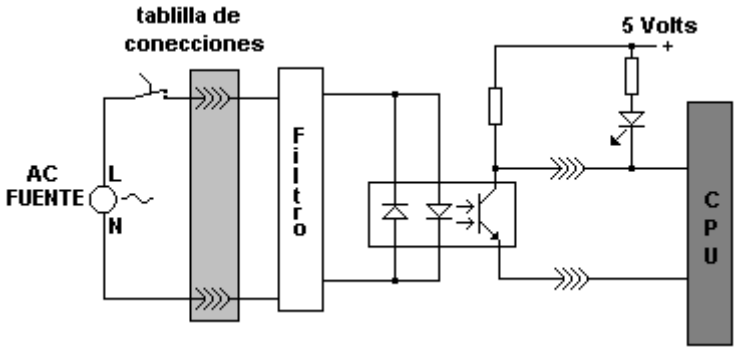


FIGURA # 4.3.1b

Los siguientes circuitos nos presentas las conexiones típicas de módulos de salida, primeramente iniciaremos presentando el circuito de salidas a transistor.

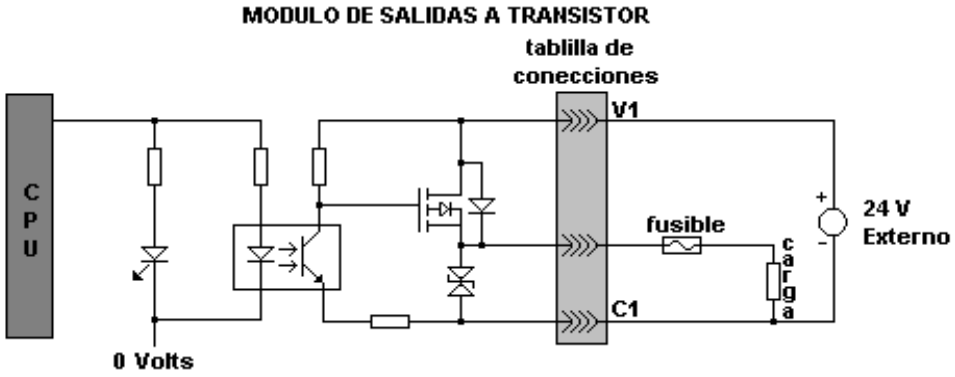


FIGURA # 4.3.1c

El siguiente circuito nos muestra las conexiones típicas para un módulo de salidas a triacs.

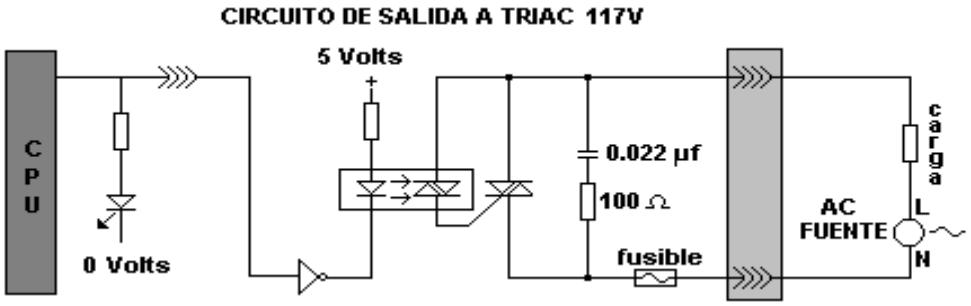


FIGURA # 4.3.1d

El siguiente circuito nos muestra las conexiones típicas de salida a relevador, observe como se puede conectar con este tipo de salidas a cargas con D.C o A.C

.

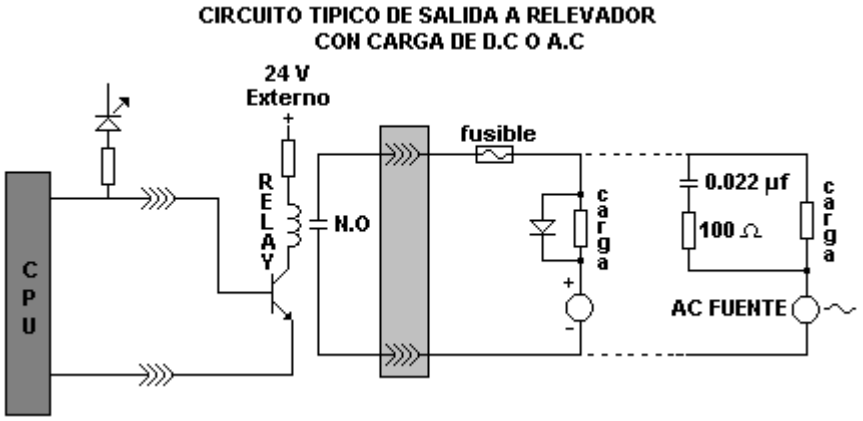


FIGURA # 4.3.1e

4.4.- Conclusiones generales sobre los PLC's.

Los PLC's fueron diseñados específicamente para:

- Operar en un ambiente industrial. Está constituido para trabajar confiablemente a pesar de variaciones que puedan tener por temperatura o ruido eléctrico.

- Usarse por el personal de mantenimiento y/o de planta. El empleo de un PLC no requiere del conocimiento de un lenguaje de programación específico, ya que en su manera elemental puede programarse utilizando el original sistema de diagramas de escalera, que es familiar a la mayoría de técnicos e ingenieros tanto eléctricos como electrónicos.

- Recibir mantenimiento por parte de los técnicos o electricistas de la planta.

- Se pueden reutilizar cuando el proceso a controlar ha quedado fuera de producción y ahora la maquina realiza otros procesos.

Ventajas de los PLC's

- Se pueden realizar cambios en las secuencias de operación de los procesos de producción de una manera relativamente sencilla.
- El empleo de un menor número de recursos humanos, reduciendo los problemas de tipo laboral.
- Los costos de la automatización se reducen en comparación con los que tienen con el control electromecánico a medida que los procesos son más complejos.
- Debido a la tecnología electrónica utilizada por los PLC's, las necesidades de espacio también se reducen en comparación al requerido con los paneles de relevadores.
- Se reduce la cantidad de dibujos ya que el software puede imprimir el programa y el usuario solo realiza los diagramas de conexiones de campo al PLC.
- Se reduce la cantidad de cableado, así como el calibre del mismo ya que el PLC realiza solo funciones de control y no de fuerza, los cuales están perfectamente definidos y separados.

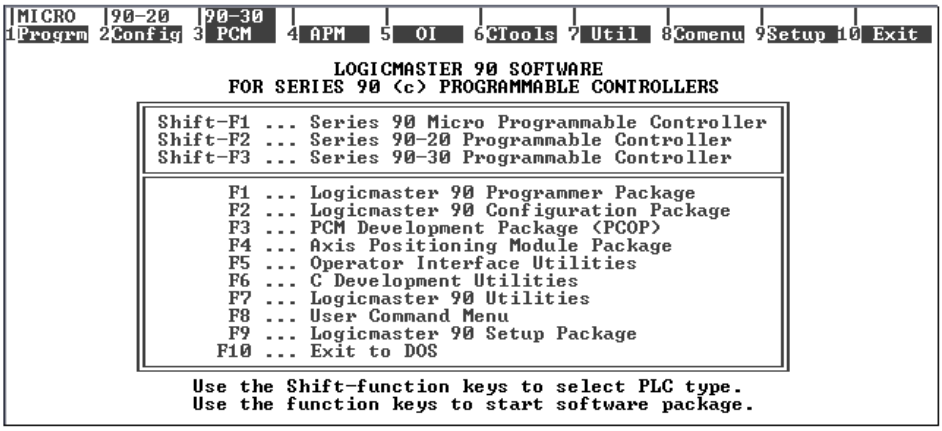
Desventajas de los PLC's

- Se requiere un programador de bolsillo para su programación.
- Aún cuando también se pueden programar por medio de una computadora, se requiere de la inversión de la misma y de un software que la mayoría de las veces resulta ser caro.
- Para comunicarse el PLC con la computadora se requieren interfaces de comunicación que también resultan ser caros.
- Los software de programación no son compatibles entre diferentes marcas de PLC por lo que se requiere de conocimientos de los mismos para marcas diferentes y en algunos casos de software diferentes para PLC de las mismas marcas pero con modelos diferentes.
  A continuación procederemos entrar en detalle con la programación de el PLC de la Familia de Ge fanuc de la serie 90-30, aunque debemos hacer la aclaración que este software es utilizado también en los modelos micro 90 y modelo 90-20.

4.5.- Configuración del software de programación para PLC's de la marca GE Fanuc de la serie micro 90, 90-20 y 9030.

En la siguiente sección mostraremos los diferentes paquetes de programación que ofrece Ge Fanuc para los controladores de la serie micro 90, 90-20 y 90-30. En este capitulo hablaremos del método de programación y del grupo de instrucciones que componen el lenguaje.

Primero haremos la aclaración que este Software de programación esta en la versión de MS-DOS, luego ya estando en ella tecleamos **LM90** y presionamos enter. Al hacer esto nos muestra la siguiente pantalla. Pantalla N° 4.5.1



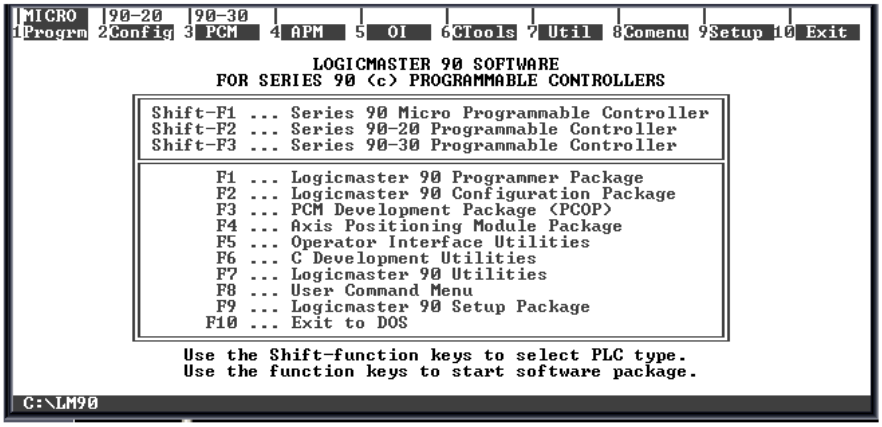
PANTALLA N° 4.5.1

En esta pantalla podemos observar los diferentes paquetes con que consta este Software.

Para entrar a cualquiera de estos diez paquetes de programación, se oprime directamente la tecla F1, F2, F3, etc. Observe que en la parte superior se encuentran una parte sombreada y otra no y con las teclas F1, F2,,,, se entra directamente a los diferentes paquetes, y con la tecla SHIFT + F1 se entra a la parte superior, así, si queremos seleccionar el Modelo Micro 90 oprimimos las teclas SHIFT + F1 y entramos directamente a este modelo, con las teclas SHIFT + F2 entramos al modelo 90-20 y con SHIFT + F3 entramos al modelo 90-30.

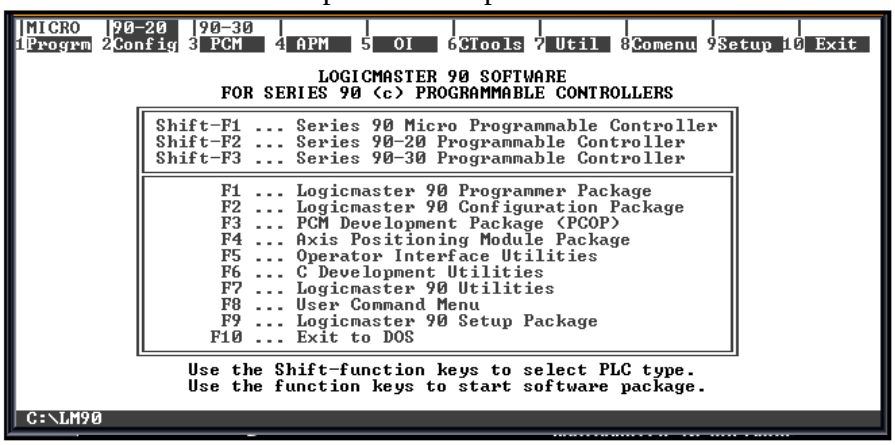
Una vez seleccionado el modelo deseado, seleccionamos uno de los diez paquetes de programación directamente con las teclas F1, F2, F3,,, etc.

Para seleccionar por ejemplo el PLC de la familia del Micro 90 oprimimos SHIFT + 1 observamos que se ha seleccionado el PLC adecuado como lo podemos observar en la pantalla N° 4.5.2



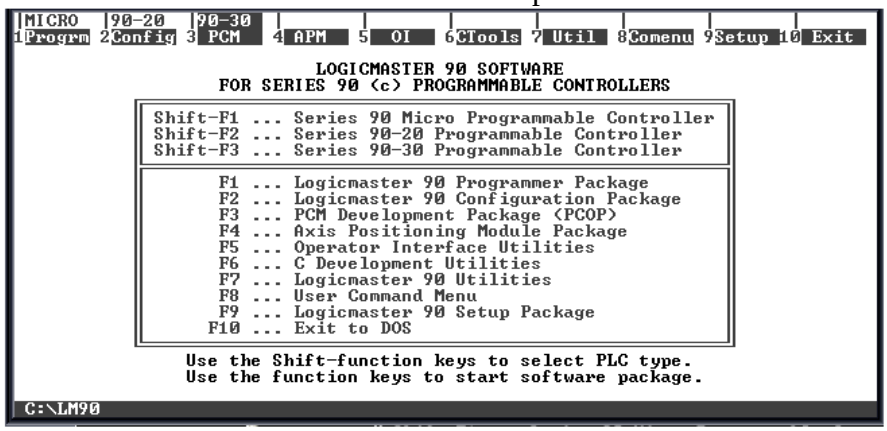
PANTALLA Nº 4.5.2

Si queremos seleccionar el PLC de la familia 90-20 se oprime la tecla SHIFT + F2 y nos aparece la selección adecuada tal como aparece en la pantalla N° 4.5.3.



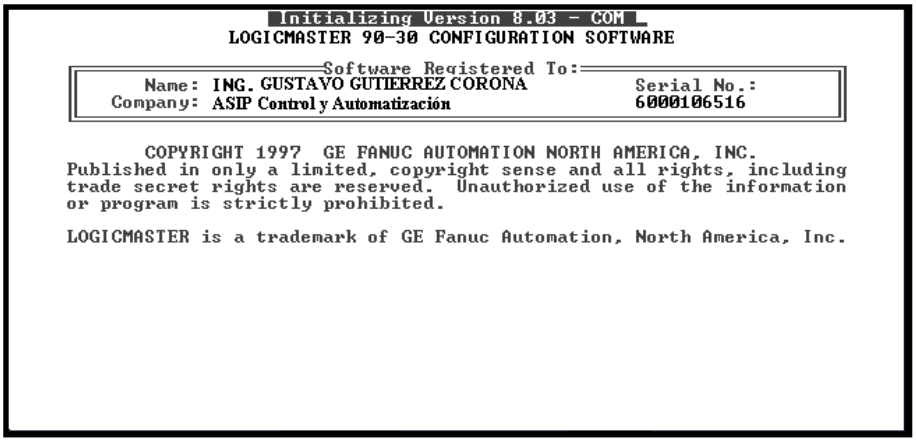
PANTALLA N° 4.5.3

Si queremos seleccionar el PLC de la familia de 90-30 oprimimos las teclas SHIFT + F3 y nos muestra el PLC seleccionado como nos lo muestra la pantalla N°4.5.4



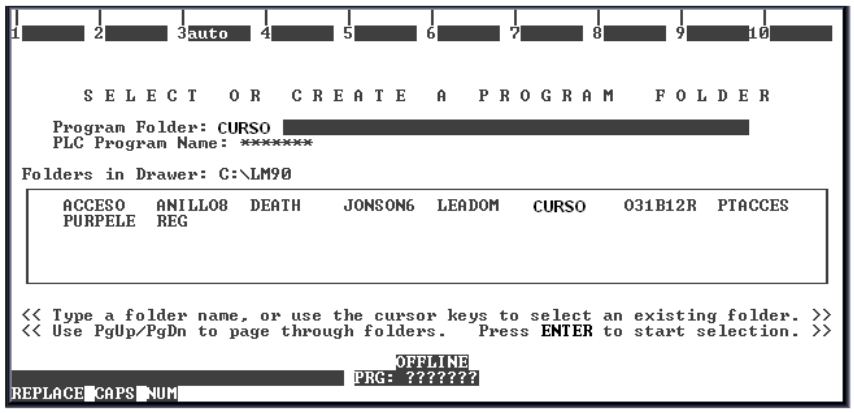
PANTALLA N° 4.5.4

Una vez seleccionado el PLC de la familia 90-30 que es el equipo que hemos seleccionado para este trabajo, procederemos a configurar el sistema, esto es, seleccionar los diferentes módulos para nuestra aplicación con su respectivo direccionamiento. Para ejecutar esta función observamos la pantalla N° 4 en la que nos muestra con F2 el paquete de configuración, por lo tanto oprimimos la tecla F2 y nos muestra la Pantalla N° 4.5.5. En la que nos muestra la versión del software y el registro y número de serie del mismo.



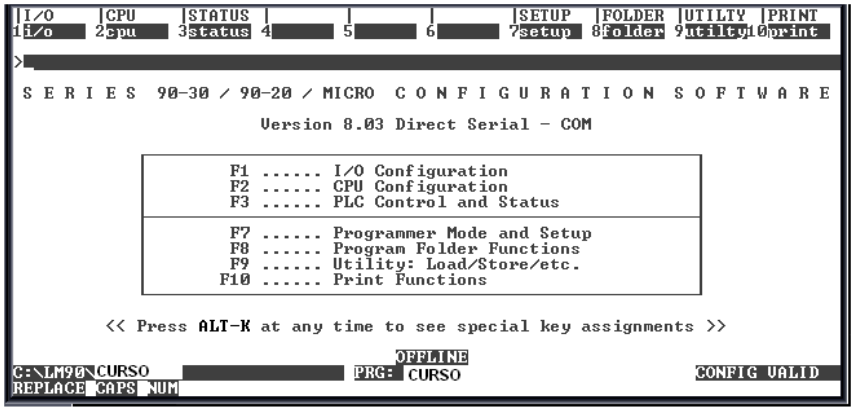
PANTALLA N°4.5. 5

La pantalla N° 4.5.5 es inicialización y por lo tanto desaparece para mostrarnos la pantalla N° 4.5.6 con el archivero con los diferentes folders contenidos en el software.



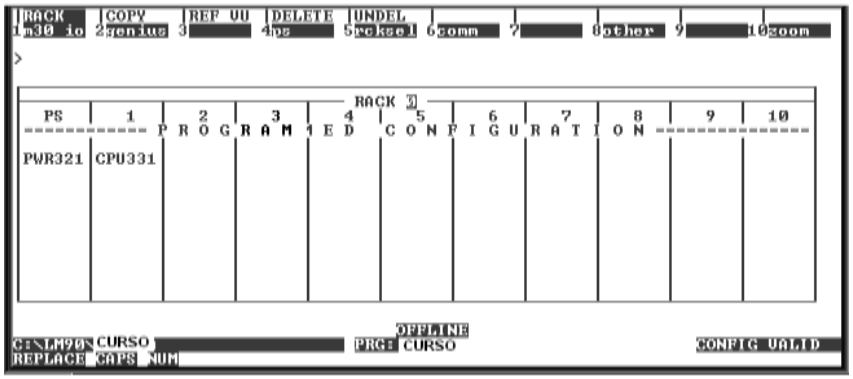
PANTALLA N°4.5. 6

Una vez mostrado el archivero con los diferentes fólder, seleccionamos el fólder deseado y si no se encuentra en el archivero, en este caso particular el fólder deseado se llama CURSO . En este caso particular ya se encuentra en el archivero nuestro programa y solo procedemos a seleccionarlo con el cursor y le oprimimos la tecla enter y nos muestra la pantalla N° 4.5.7, sí queremos generar un fólder nuevo, escribimos el nuevo fólder con un máximo de 7 letras y le pulsamos enter.



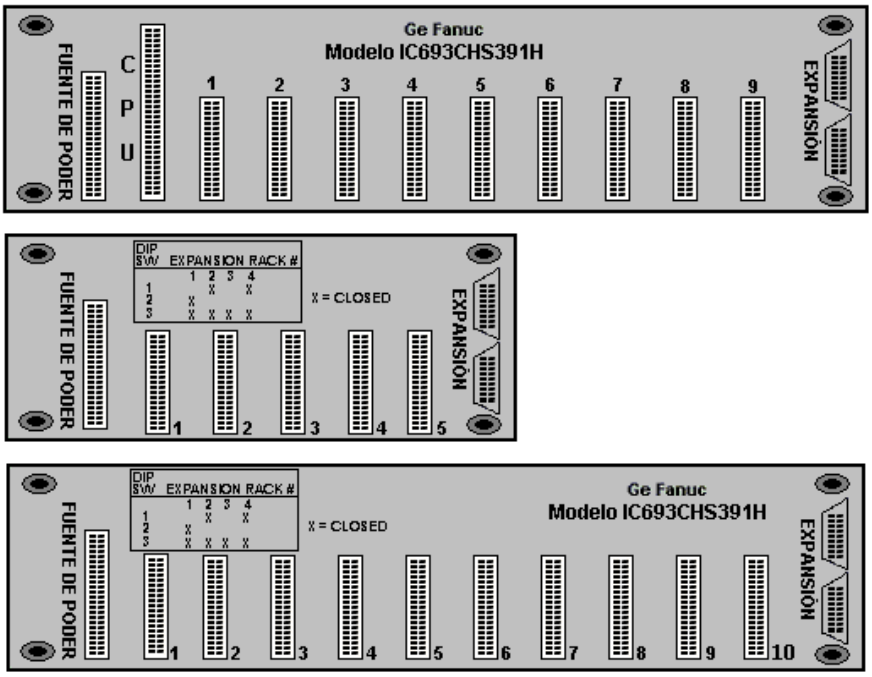
PANTALLA N° 4.5.7

Una vez seleccionado nuestro fólder verificamos que el nombre del mismo aparezca en la parte inferior al centro, como podemos observar por default el software nos entrega la siguiente configuración. Pantalla 4.5.8



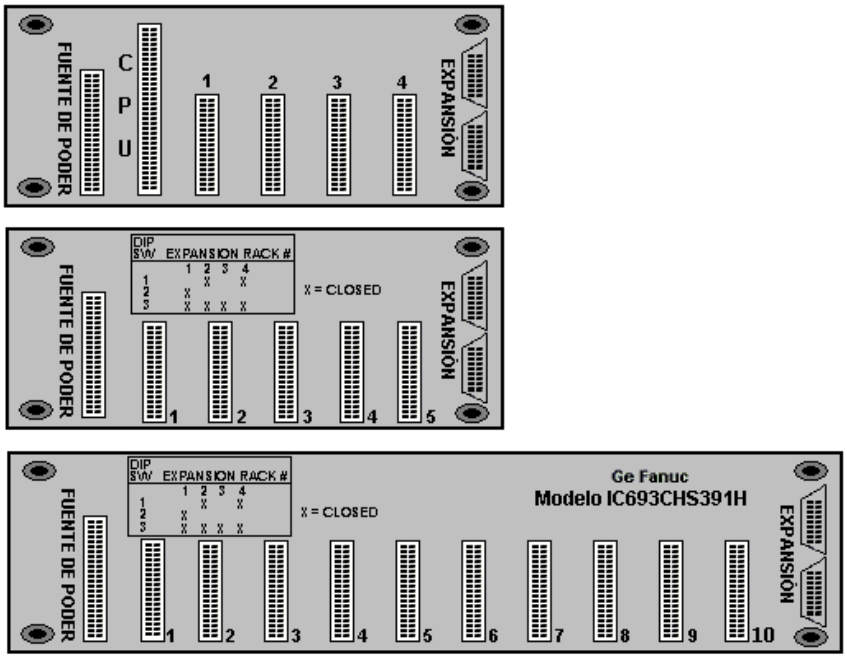
PANTALLA N°4.5. 8

Ahora debemos tomar en cuenta las diferentes opciones con que se cuenta en cuanto a los racks que dispone para realizar una automatización y que son las siguientes. Rack de 10 ranuras donde la CPU requiere ser externo, y cuenta con 9 ranuras para ser utilizadas, a su vez ésta tiene dos alternativas, un Rack de expansión de 5 ranuras o una de 10 y así sucesivamente hasta agotar la capacidad de la CPU. (Ver pantalla 4.5.9).



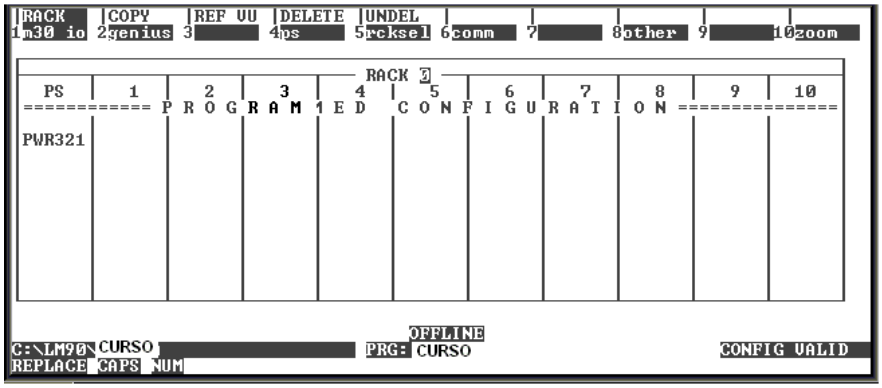
PANTALLA N° 4.5.9

En la pantalla Nº 4.5.10 se observa la segunda alternativa, en la cual el Rack maestro es de 5 ranuras y la CPU requiere una posición independiente con lo cual tenemos cuatro ranuras disponibles con las mismas expansiones explicadas para el modelo anterior.



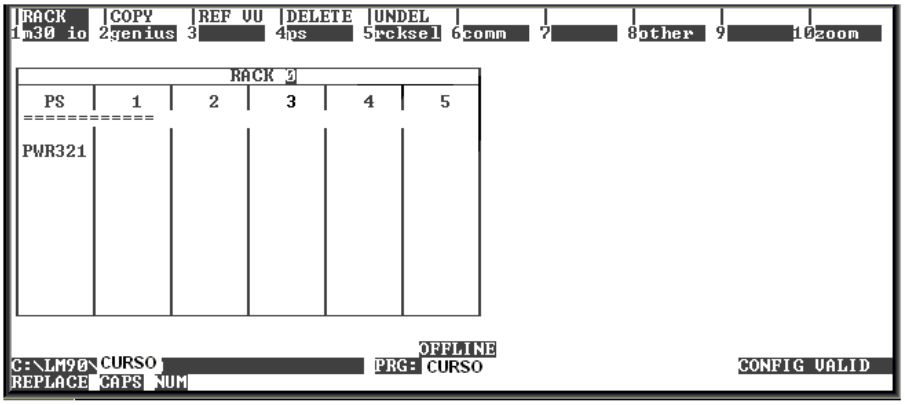
PANTALLA N° 4.5.10

En seguida tenemos la pantalla N 4.5.11 con un rack de 10 ranuras en la cual el CPU se encuentra integrado teniendo más capacidad de entradas y salidas pero con las cuales no se puede tener expansión.



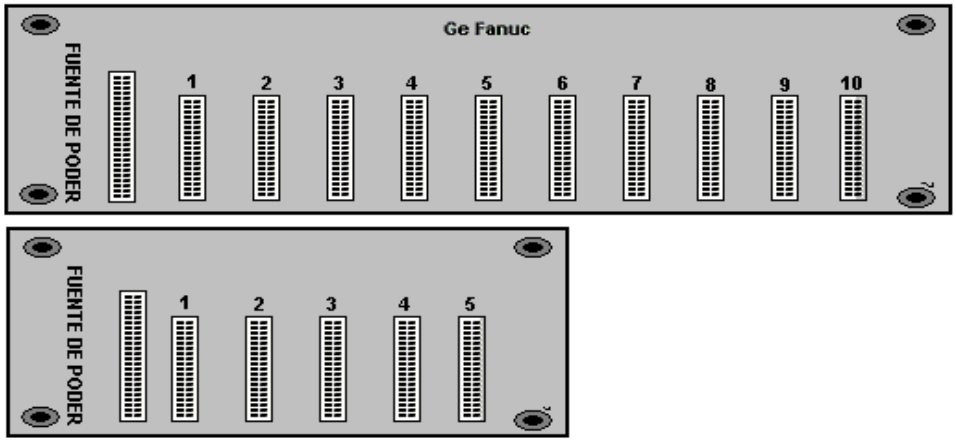
PANTALLA N° 4.5.11

Finalmente tenemos la pantalla N 4.5.12 en la cual se muestra un rack de 5 ranuras con CPU integrado pero sin capacidad de expansión.



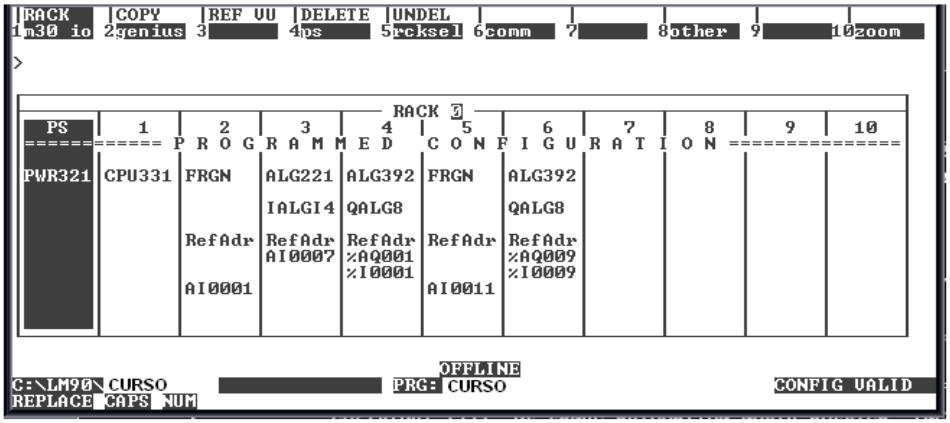
PANTALLA N° 4.5.12

En la pantalla N 4.5.13 se observa el acomodo de los módulos de entradas y salidas que se explicaron en las pantallas N 4.5.11 y 4.5.12.



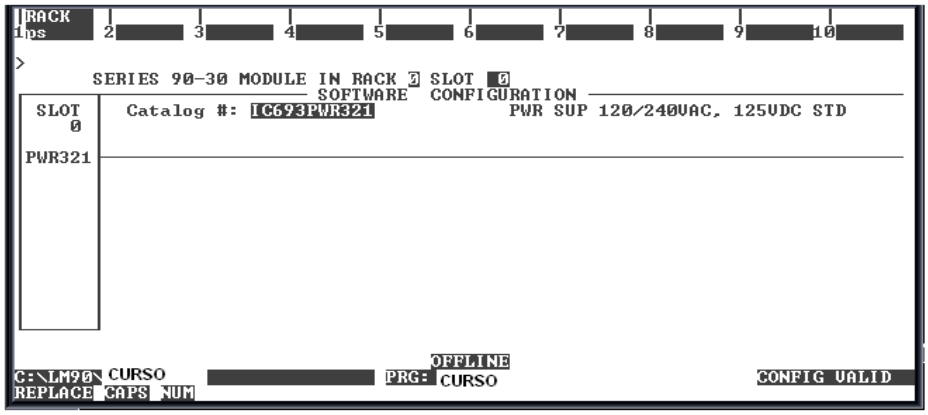
PANTALLA N° 4.5.13

Ahora con la tecla F1 seleccionamos el modulo de fuente de poder, como nos lo muestra la pantalla N 4.5.14.



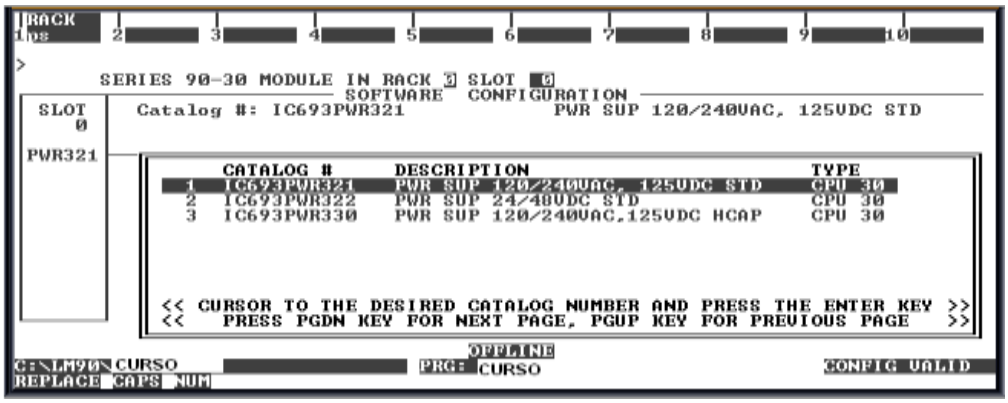
PANTALLA N° 4.5.14

En la pantalla 14 podemos observar como nos despliega nuestro rack seleccionado en este caso un rack de 10 ranuras iniciando siempre por la fuente de poder modelo IC693PWR321 con la tecla F 10 (Z00m) nos muestra las características de la misma tal como lo muestra la pantalla Nº 4.5.15



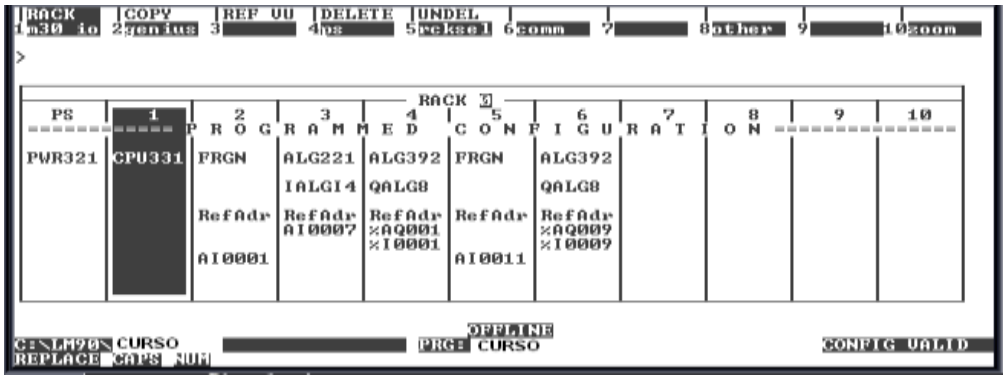
PANTALLA N° 4.5.15

Si la fuente mostrada en la pantalla N° 4.5.15 no es la deseada con la tecla F1 nos despliega las diferentes opciones que podemos elegir para nuestra aplicación, en este caso es una fuente de alimentación de 120 VAC, 125VDC tal como lo muestra la pantalla N° 4.5.16.



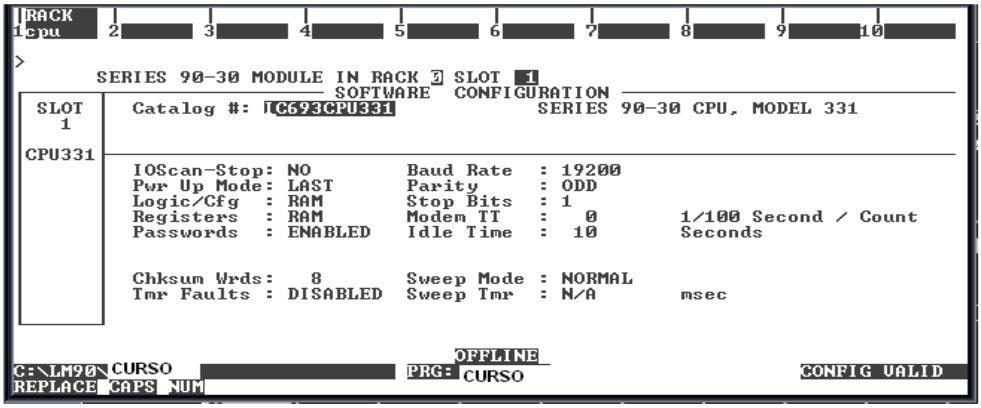
PANTALLA N° 4.5.16

Una vez seleccionada nuestra fuente de poder nos regresamos con la tecla escape hasta la pantalla principal (ver pantalla N° 4.5.8) y con el cursor a la derecha nos posicionamos en la ranura de nuestro CPU tal como lo muestra la pantalla N° 4.4.17.



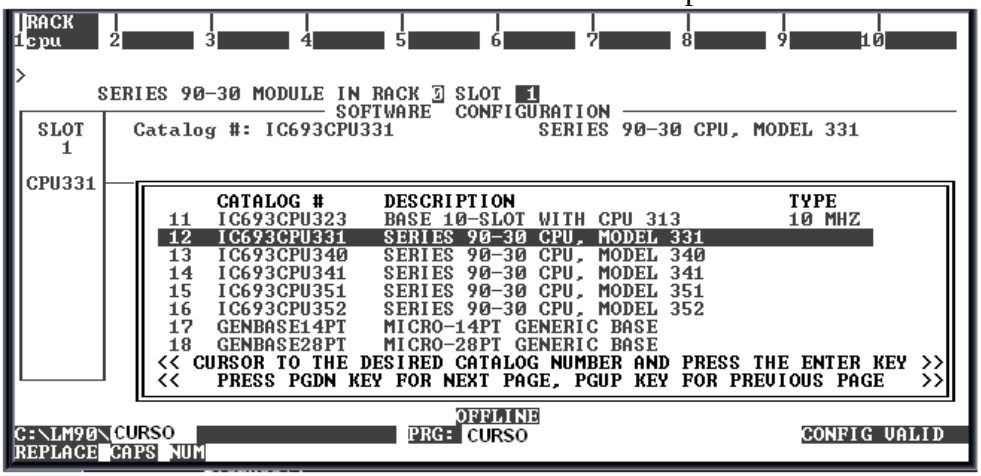
PANTALLA N° 17

Una vez en la ranura del CPU con la tecla F 10 (zoom) podemos ver las características del mismo tal como se muestra, en la pantalla N° 18 en la cual se ven las características internas de la CPU



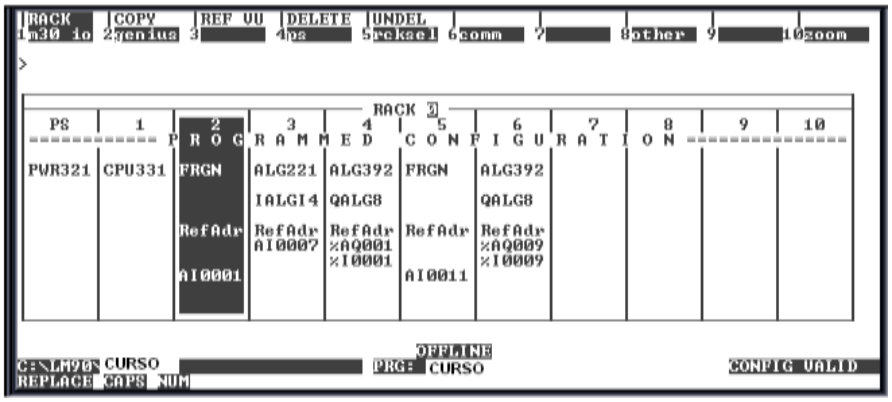
PANTALLA N° 4.5.18

Si nuestro CPU se requiere diferente al mostrado en la pantalla N° 4.5.18 podemos acceder a otras alternativas de CPU con la tecla F1 como se muestra en la pantalla N° 4.5.19.



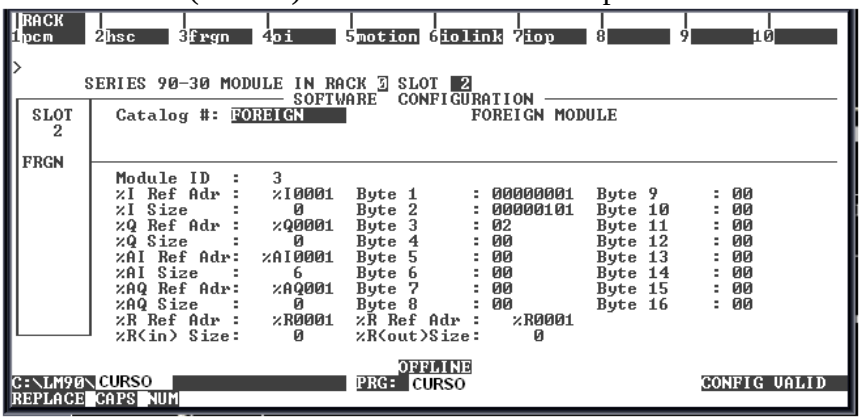
PANTALLA N° 4.5.19

En la pantalla N° 4.5.19 podemos seleccionar las diferentes opciones que nos ofrece el software. Para esta aplicación en particular seleccionamos el CPU IC693CPU331 que nos proporciona la capacidad adecuada para esta aplicación. Para poder seleccionar la tarjetas de entradas y salidas regresamos en la misma forma con la tecla "escape" hasta la pantalla del rack principal (ver pantalla N° 4.5.8) luego con el cursor hacia la derecha nos posicionamos hasta la ranura N° 2 que es donde inicia la configuración de los módulos. Tal como aparece en la pantalla N° 4.5.20.



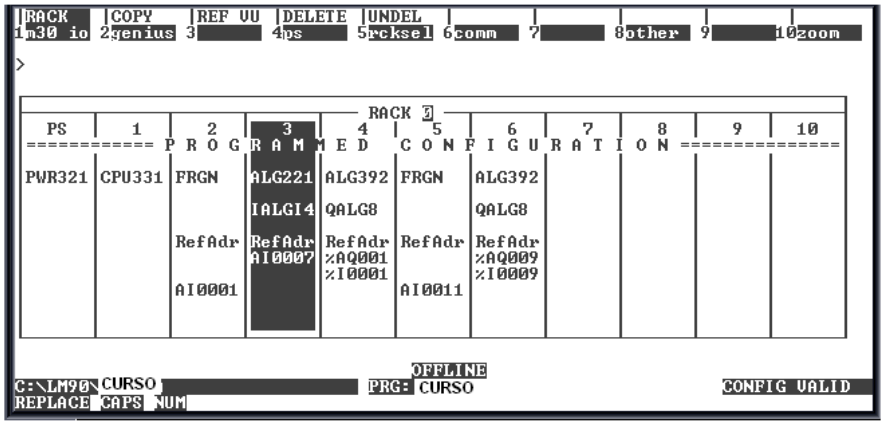
PANTALLA N° 20

Para configurar el direccionamiento del módulo de entradas y salidas analógicas es necesario oprimir la tecla F 10 (Zoom) tal como lo muestra la pantalla N° 4.5.21.



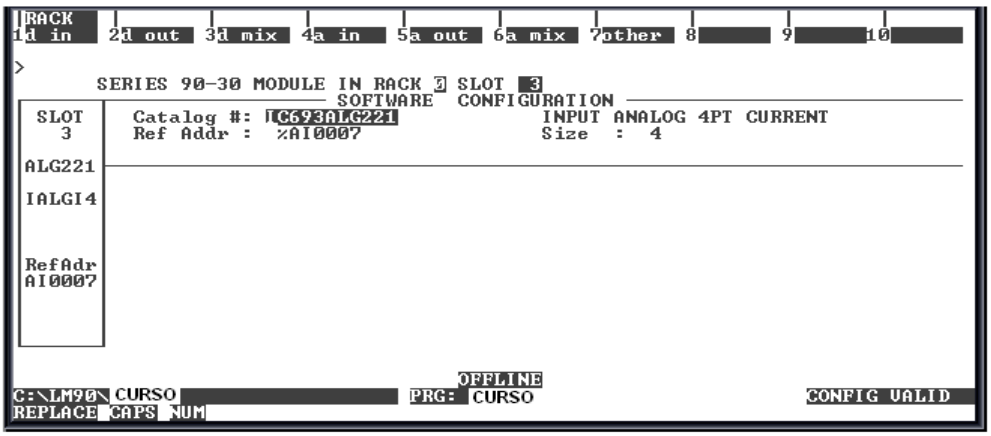
PANTALLA N° 21

Para configurar el módulo de entradas analógicas nos posicionamos en la ranura N° 4.5.3 no sin antes regresar con la tecla ESC hasta ver el rack principal como lo muestra la pantalla N° 4.5.22.



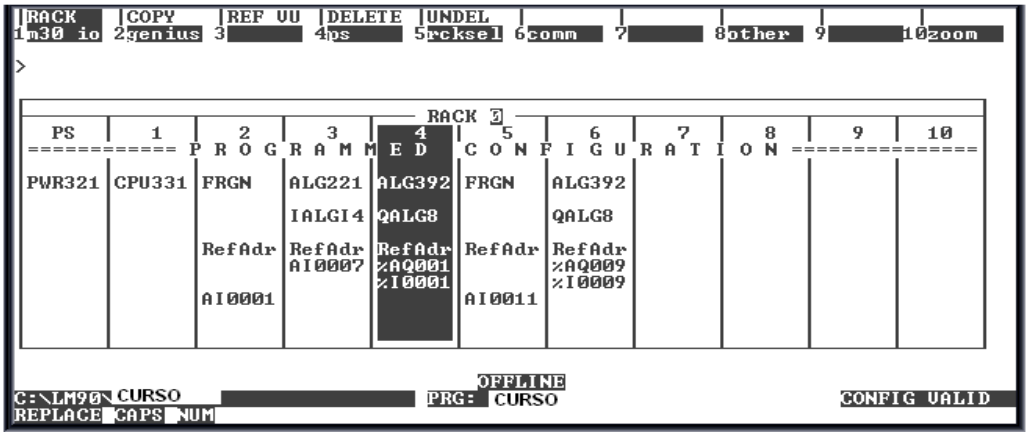
PANTALLA N° 4.5.22

Para ver el direccionamiento de este módulo oprimimos la tecla F 10 (Zoom) en este caso es un modulo de 4 entradas analógicas e individualmente se configura el tipo de señal que operara en corriente de (4 a 20 mA) como es nuestra aplicación mostrada en la pantalla N° 4.5.23.



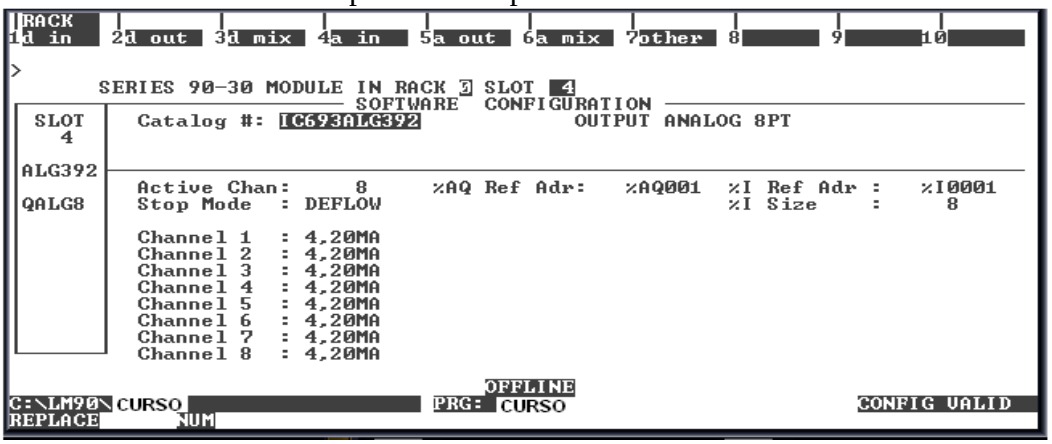
PANTALLA N° 4.5.23

Para configurar el siguiente modulo de la misma forma se regresa al rack principal (ver pantalla N° 4.5.8) en este caso se configura el modulo de salidas analógicas tal como se muestra en la pantalla N° 4.5.24.



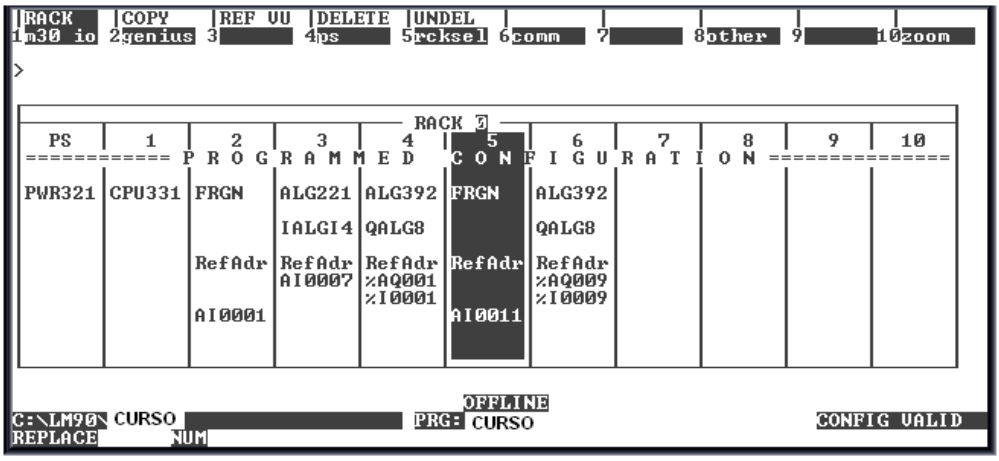
PANTALLA N° 4.5.24

En esta pantalla con la tecla F 10 (Zoom) podemos observar el direccionamiento y características del mismo tal como aparece en la pantalla N° 4.5.25.



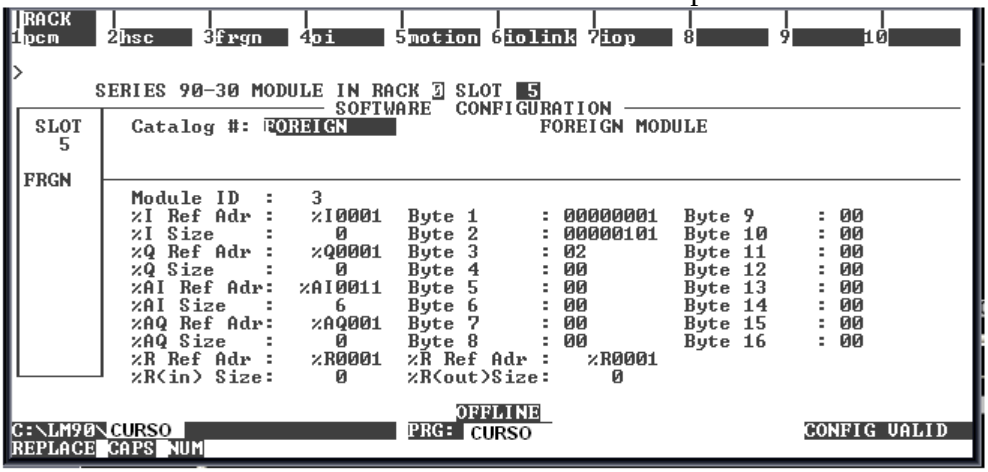
PANTALLA N° 4.5.25

En nuestra aplicación es necesario hacer uso de módulos fabricados por terceros en este caso por Horner por esa razón en el software aparece como modulo extranjero y aparece como tal en la pantalla N° 4.5.26.



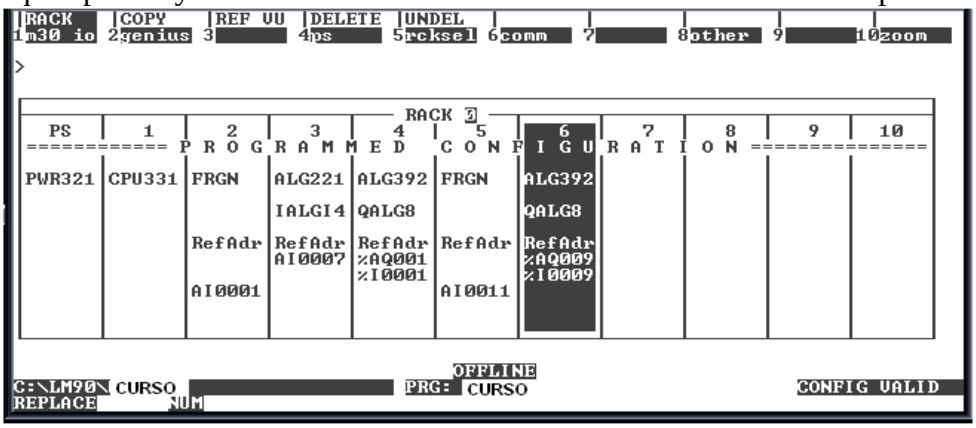
PANTALLA N° 4.5.26

Una vez en la ranura deseada con la tecla F10 (Zoom) se configura el direccionamiento así como las características del mismo tal como se muestra en la pantalla N° 4.5.27.



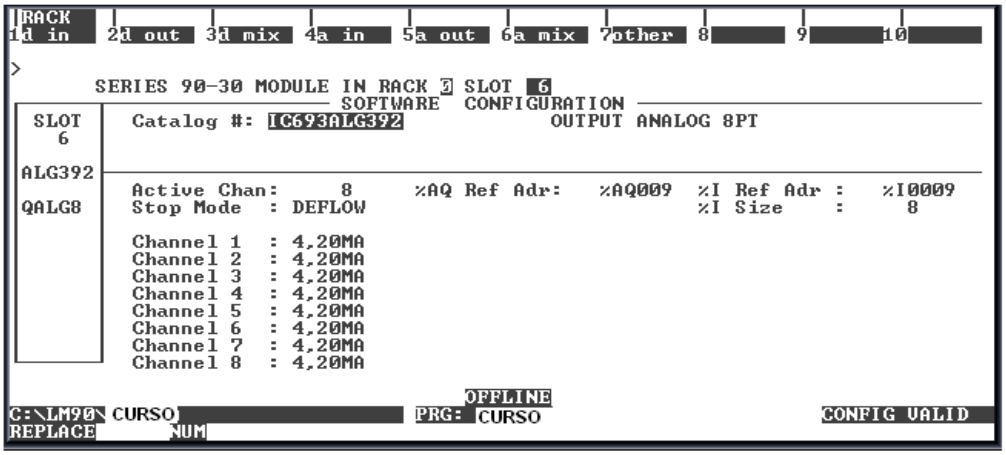
PANTALLA N° 4.5.27

Para configurar el último módulo analógico se coloca una en el rack principal con la tecla escape hasta que aparece y con el cursor hacia la derecha tal como lo muestra la pantalla N° 4.5.28.



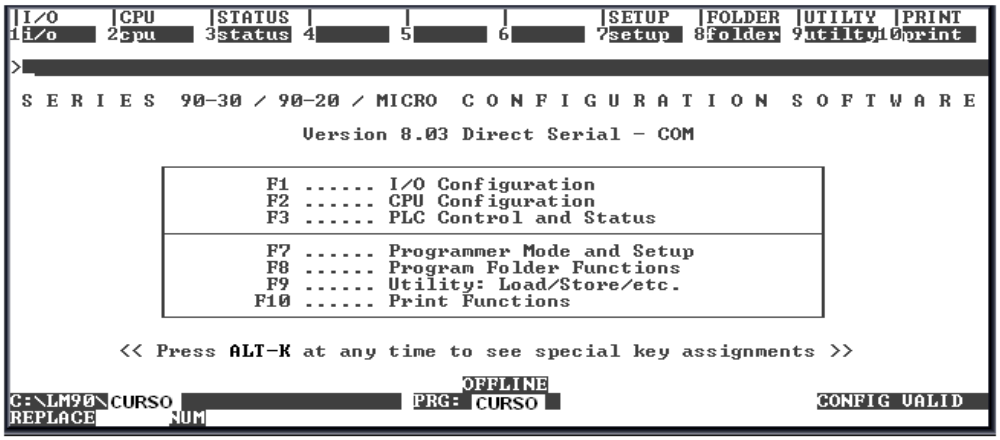
PANTALLA N° 4.5.28

Ahora para ver su configuración interna del modulo oprimimos la tecla F 10 (Zoom) tal como aparece en la pantalla N° 4.5.29.



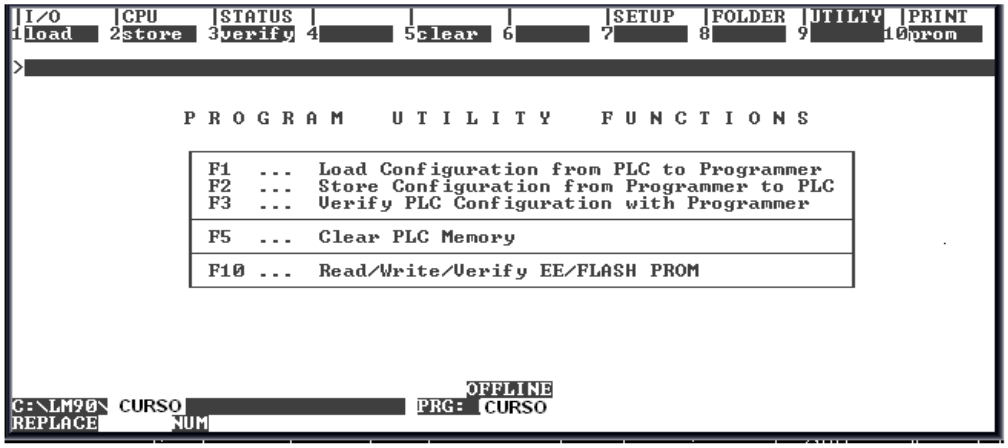
PANTALLA N° 4.5.29

Una vez configuramos todos los módulos seleccionados para nuestra aplicación con su respectivo direccionamiento, procedemos a guardarlo. Esto se realiza con la tecla escape hasta la pantalla principal tal como aparece en la pantalla N° 4.5.30.



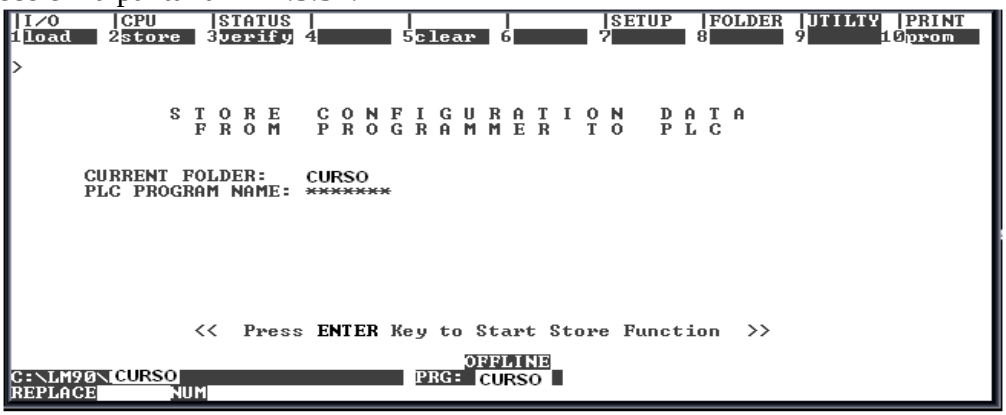
PANTALLA N° 4.5.30

En este momento procedemos a cargar nuestra configuración a la CPU oprimiendo la tecla F9 (utility) y nos muestra la pantalla N° 4.5.31.



PANTALLA N° 4.5.31

Ahora oprimimos la tecla F2 para almacenar la configuración de la computadora al PLC tal como aparece en la pantalla N° 4.5.32.



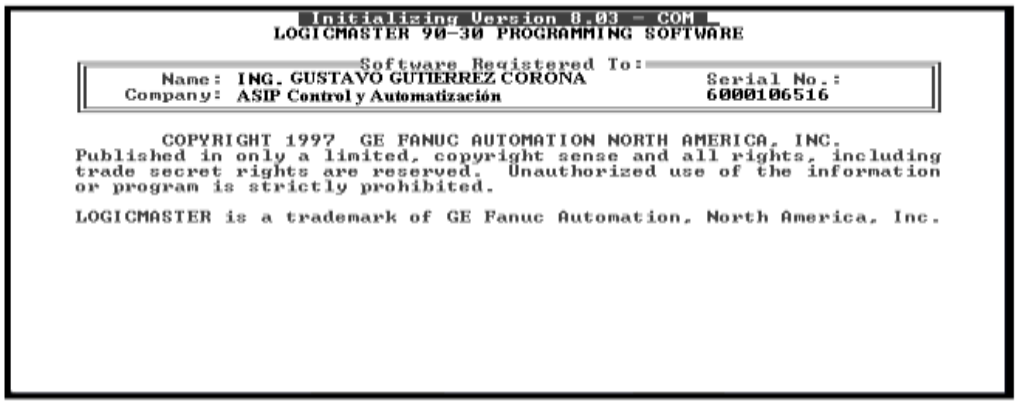
PANTALLA N° 4.5.32

Ahora se oprime enter con lo cual queda cargado la configuración al CPU del PLC.

Finalmente cargada nuestra configuración procedemos a salir del paquete del software hasta la pantalla principal con la tecla escape hasta encontrar la pantalla N° 4.5.33.

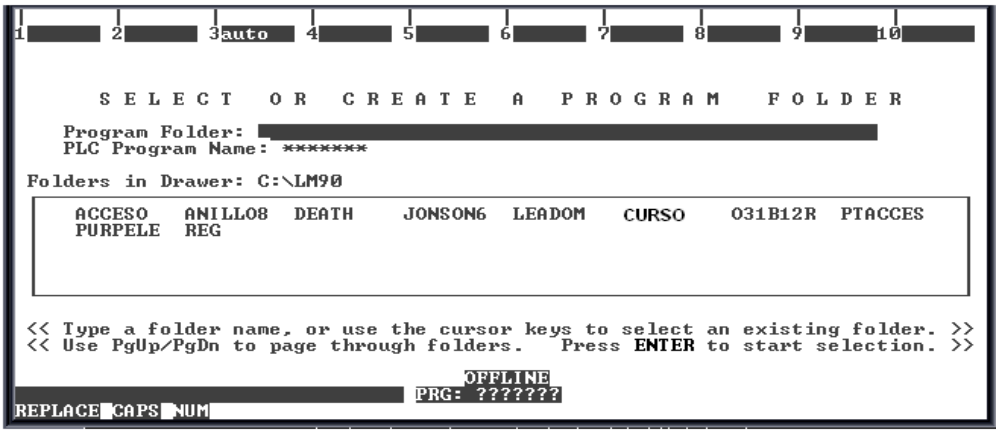
PANTALLA N° 4.5.33

Una vez en esta pantalla principal, con la tecla F1 entramos al paquete de programación tal como aparece en la pantalla N° 4.5.34 la cual nos muestra el propietario de la licencia y el número de serie, misma que se muestra en forma transitoria.



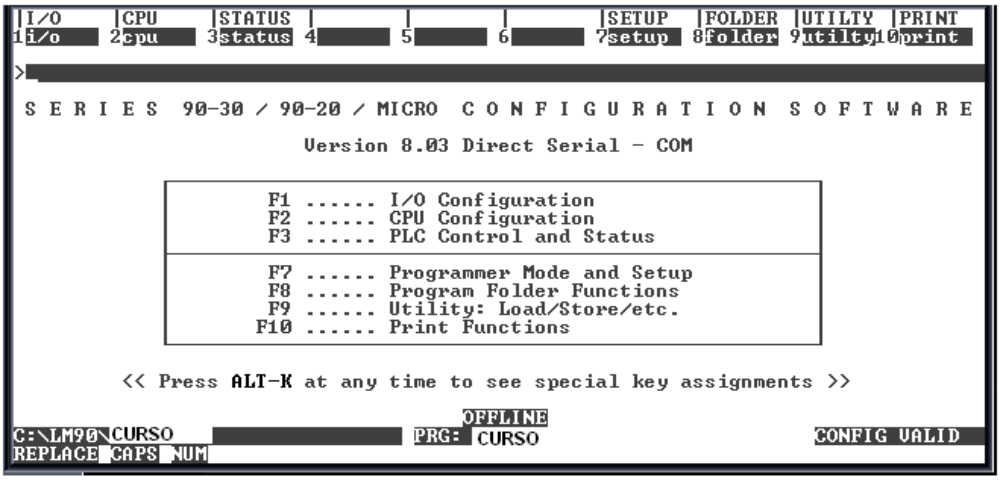
PANTALLA N° 4.5.34

Una vez estabilizado esta pantalla nos muestra el archivo con todos los folders, contenidos en este paquete de software. Tal como lo muestra la pantalla N° 4.5.35 y con el cursor nos posicionamos en el fólder deseado en este caso el de ***"CURSO"*** y oprimimos enter.



PANTALLA N° 4.5.35

Ahora si oprimimos la tecla F1 entonces nos aparecerá la pantalla Nº 4.5.36.



PANTALLA N° 4.5.36

Descripción de las diferentes teclas mostradas en la pantalla N° 4.5.37

Función	Descripción
Program	Crea, edita o monitorea un programa lógico
Tables	Muestra y cambia datos de referencia
Status	Selecciona el estado del programa Esta tecla incluye la muestra las fallas I/O de las entradas y salidas y del PLC
Setup	Muestra y cambia los puertos y otros parámetros de programación
Folder	Carga almacena o borra, limpia asegura o realiza respaldos de un programa en el fólder
Print	Imprime el programa de un fólder.

(Ver figura 4.5.37)

Automatización con control programable

Introducción:

A continuación presentamos los circuitos de control electromecánico con su equivalente en control programable en el que las entradas se representan con la literal (**I**) y las salidas se representan con la letra (**Q**), aquí es importante hacer notar que esta representación de entradas y salidas corresponden a la Marca Ge Fanuc, las cuales cambian para otros PLC de otras Marcas.

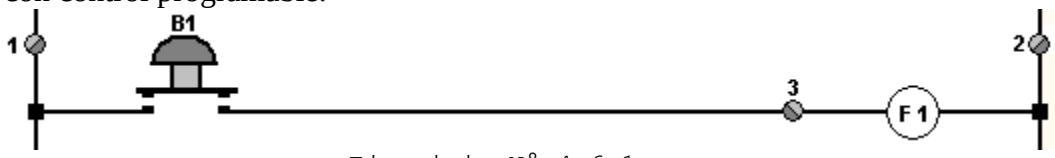
PROGRAMACIÓN DEL PLC MARCA Ge Fanuc.

4.6 Ejemplos de circuitos combinacionales con control programable.

EJERCICIO Nº 1:

A continuación presentamos el ejercicio N° 4.6.1 en el cual se tiene un botón normalmente abierto y un foco y se desea que al oprimir el botón se prenda el foco y al soltarlo se apague.

Primeramente realizamos el circuito de control electromecánico y posteriormente su equivalente con control programable.

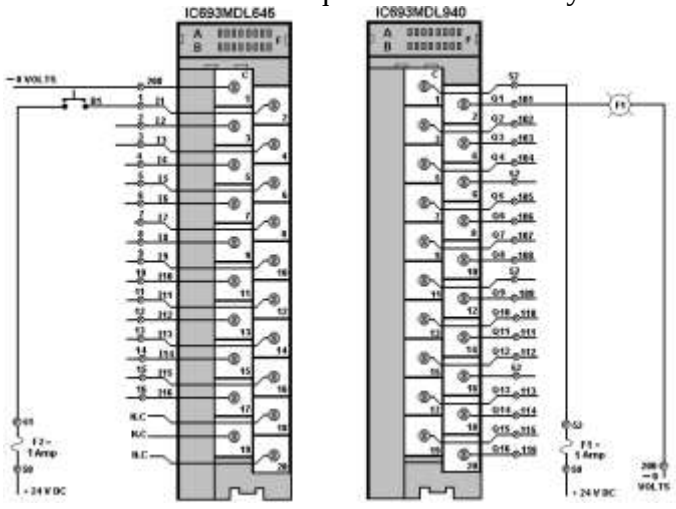


Ejercicio N° 4.6.1

Equivalente con control programable.

```
|[  START OF LD  PROGRAM CURSO 1 ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]
| << RUNG 4  STEP #0001 >>
 BOTON_1                                                  FOCO_1
|%I0001                                                   %Q0001
+--] [-----------------------------------------------------------( )-
 [ END OF PROGRAM LOGIC      ]
```

Para este circuito se requiere una entrada y una salida. Observe que para los siguientes ejercicios estamos utilizando un módulo de 16 entradas de 24 Vdc lógica positiva modelo IC693MDL645. Mientras que para las salidas se esta utilizando un módulo de 16 salidas a relevador de 2 Amp por punto modelo IC693MDL940, el cual tiene cuatro comunes, cada común corresponde a cuatro salidas, observe como la carga en este caso un foco, esta conectado a una fuente de alimentación de 24 V dc. Para este circuito se requiere una entrada y una salida.

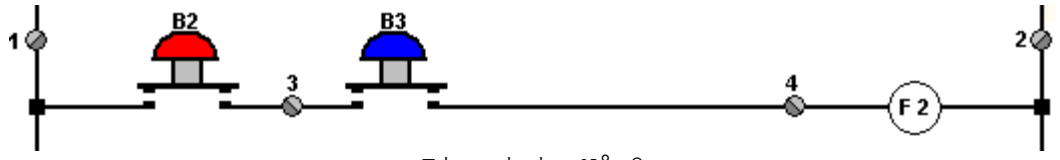


Cableado para el ejercicio N° 4.6.1

EJERCICIO Nº 2:

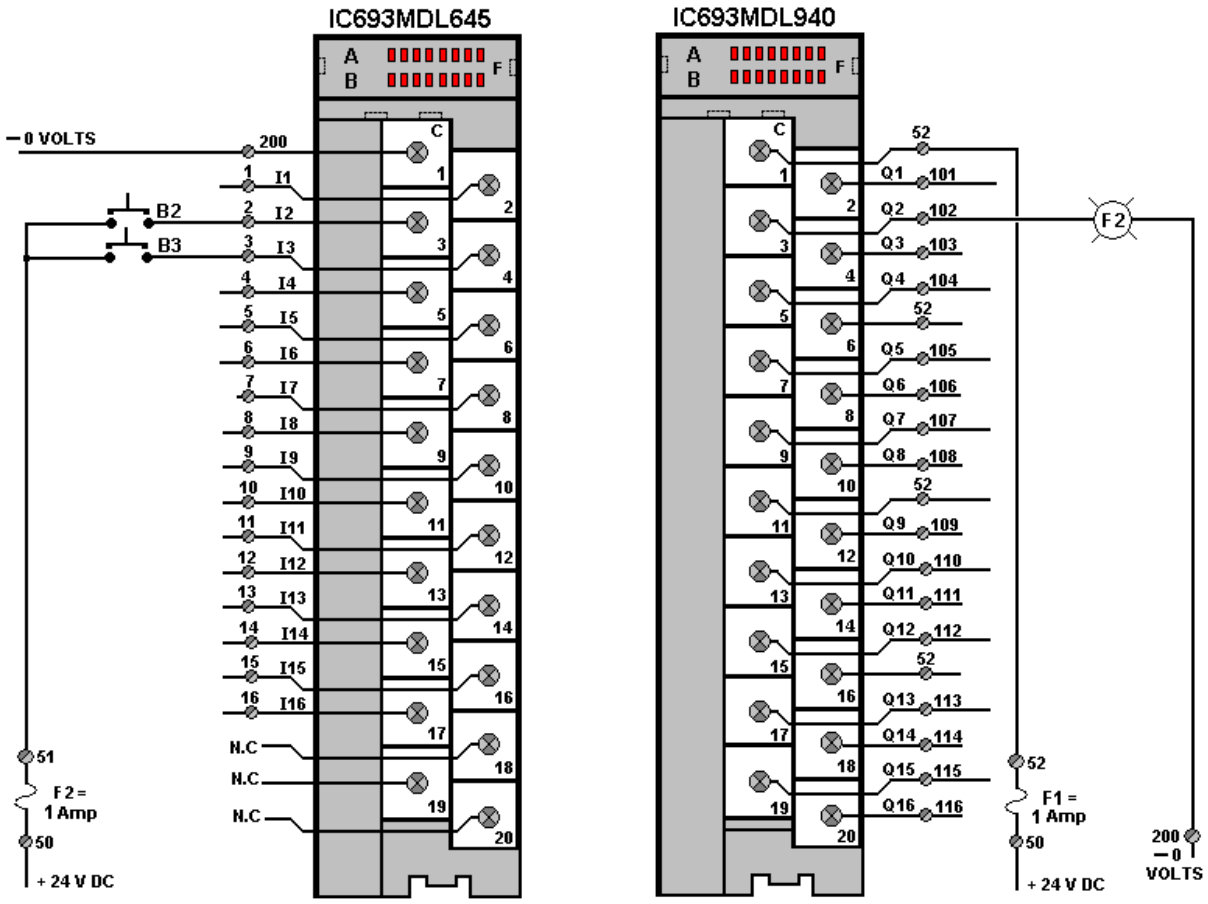
En el ejercicio N° 4.6.2 se presentan dos botones en serie, de tal manera que para que se energice el foco 2 se requiere que se activen ambos botones. También se presenta su circuito equivalente con control programable.

Para este circuito se requieren dos entradas y una salida.



Ejercicio N° 2

```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]

| << RUNG 4  STEP #0001 >>
|BOTON_2 BOTON_3                                                  FOCO_2
|%I0002  %I0003                                                   %Q0002
+--] [-----] [-----------------------------------------------------------( )-
| [      END OF PROGRAM LOGIC      ]
```

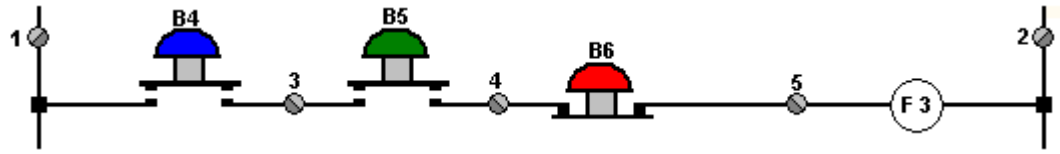


Cableado para el ejercicio N° 4.6.2

EJERCICIO Nº 3:

El ejercicio N° 4.6.3 nos muestra un circuito con tres botones en serie los cuales deberán activarse el botón 4 y el botón 5 además que no se active el botón 6 para que se energice el foco 3. También se presenta su circuito equivalente con control programable.

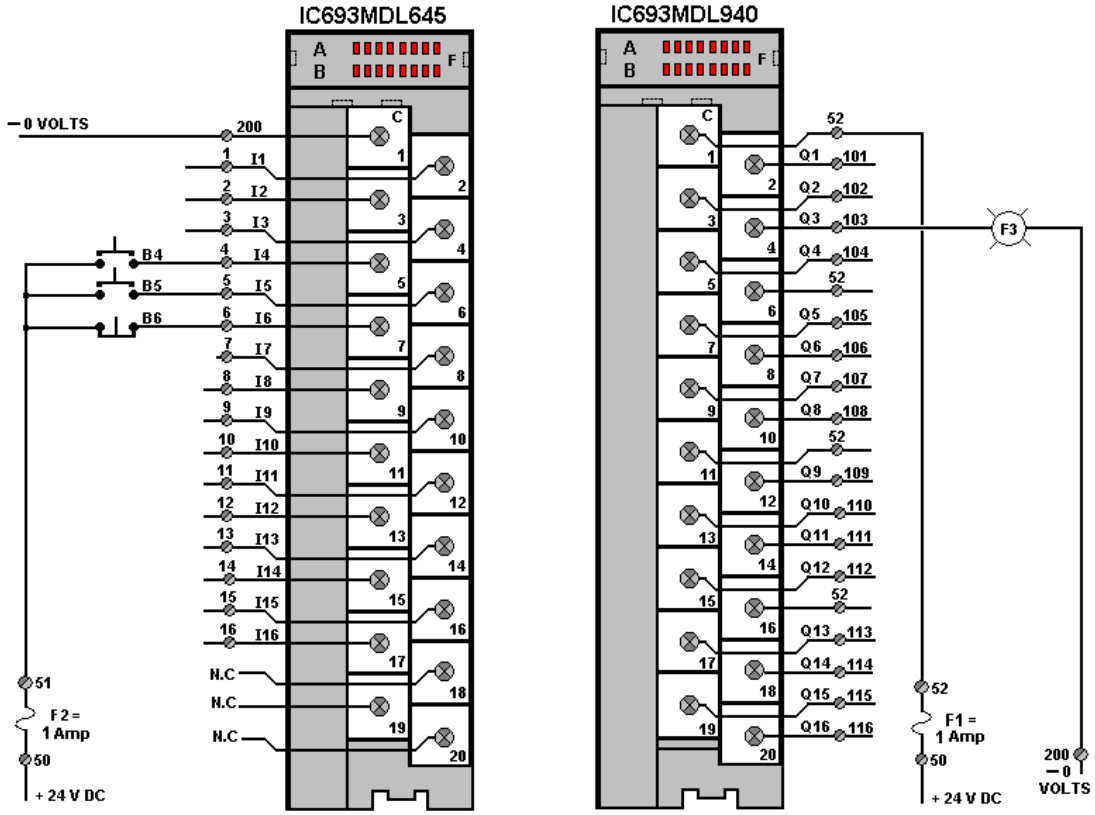
Para este circuito se requieren tres entradas y una salida.



Ejercicio N°4.6.3

Observe como el contacto de I6 se programó normalmente abierto ya que la entrada al módulo se cableo con un botón cerrado.

```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]

| << RUNG 4  STEP #0001 >>
|BOTON_4  BOTON_5  BOTON_6                                        FOCO-3
|%I0004  %I0005  %I0006                                          %Q0003
+--] [-----] [-----] [-----------------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
```



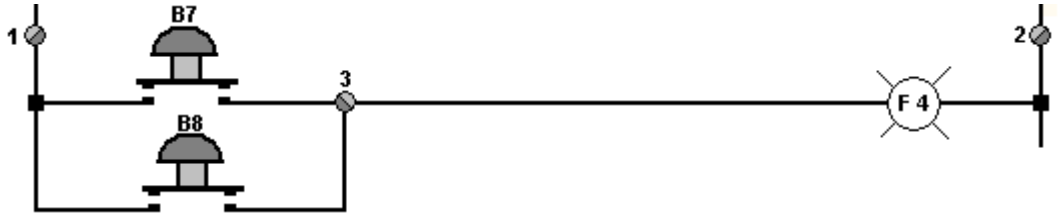
Cableado para el ejercicio N° 4.6.3

EJERCICIO Nº 4:

En el ejercicio N° 4.6.4 se presentan dos botones en paralelo, en este circuito se aprecia que para que se energice el foco 4 se requiere que se active cualquiera de los dos botones o ambos.

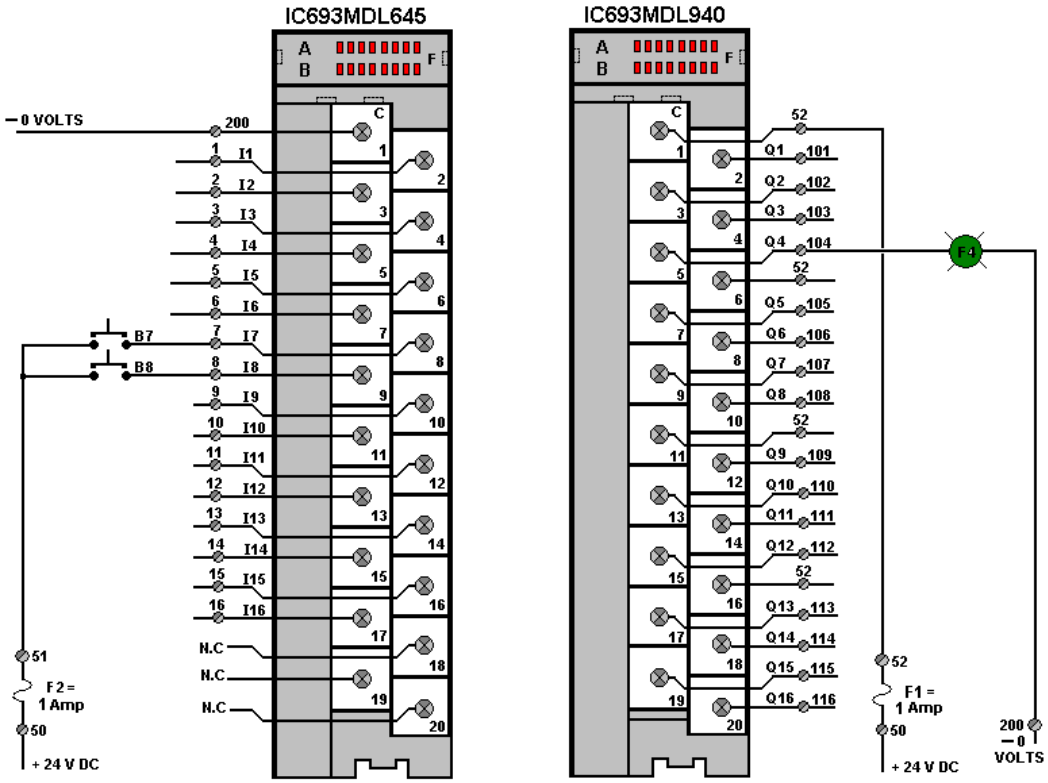
También se presenta su circuito equivalente con control programable.

Para este circuito se requieren dos entradas y una salida.



Ejercicio N° 4.6.4

```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]
| << RUNG 4  STEP #001 >>
| FOCO_7                                                        FOCO_4
|%I0007                                                         %Q0004
+--] [--+-------------------------------------------------------------( )--
| FOCO_8|
|%I0008 |
+--] [--+
|[      END OF PROGRAM LOGIC      ]
```

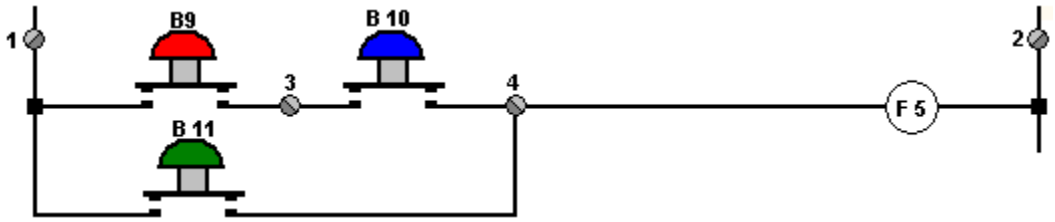


Cableado para el ejercicio N° 4.6.4

EJERCICIO Nº 5:

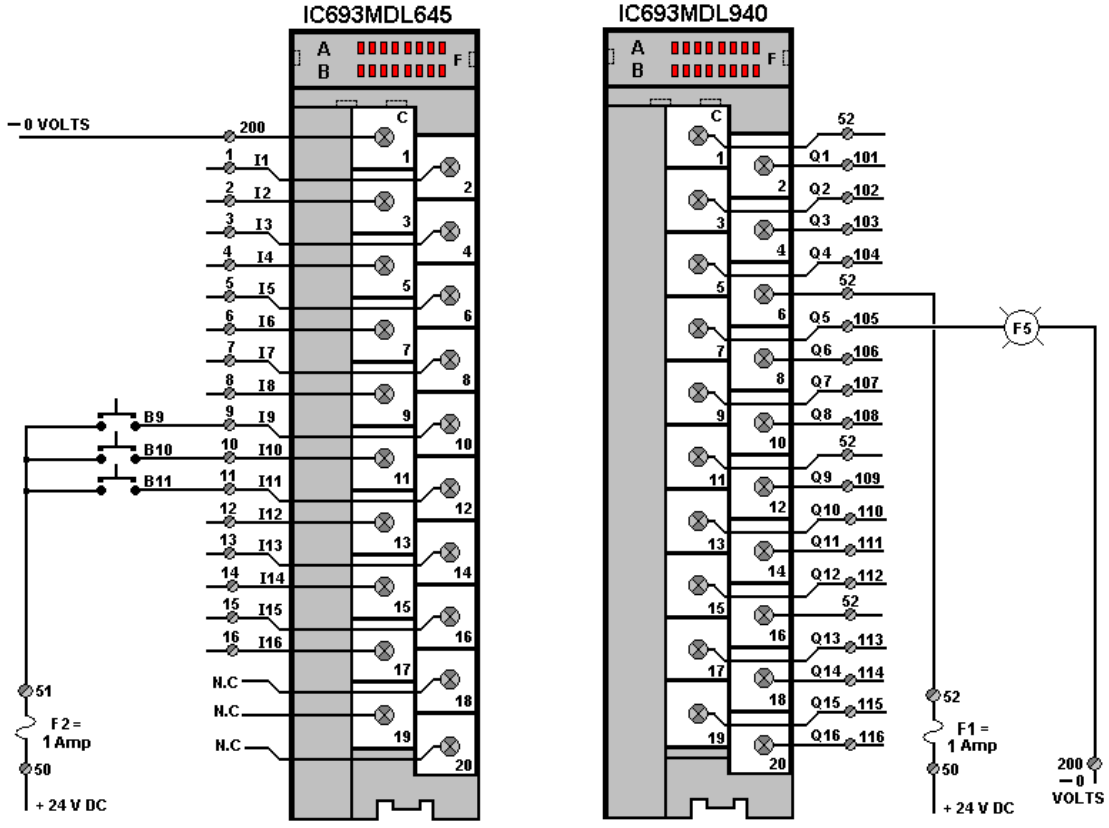
En el ejercicio N° 4.6.5 se presenta un circuito con dos botones en serie abiertos y un botón abierto en paralelo, de tal manera que para que se energice el foco 5 se requiere que se activen el botón 9 y el botón 10 o bien que se active el botón 11.

Para este circuito se requieren tres entradas y una salida.



Ejercicio N° 4.6.5

```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]
| << RUNG 4  STEP #001 >>
 BOTON_9 BOT_10                                                  FOCO_5
|%I0009  %I0010                                                  %Q0005
+--] [-----] [--+-------------------------------------------------( )--
|BOT_11         |
|%I0011         |
+--] [----------+
|[      END OF PROGRAM LOGIC      ]
```

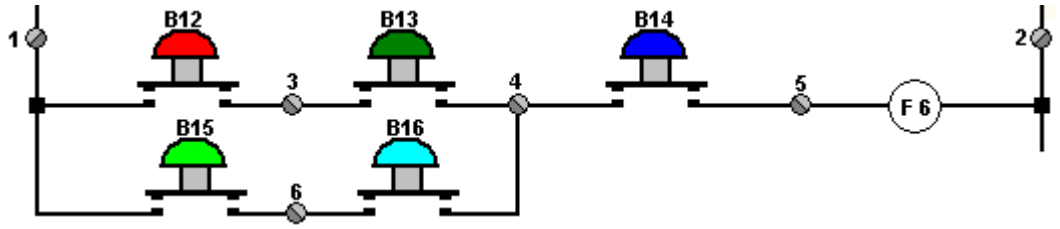


Cableado para el ejercicio N° 4.6.5

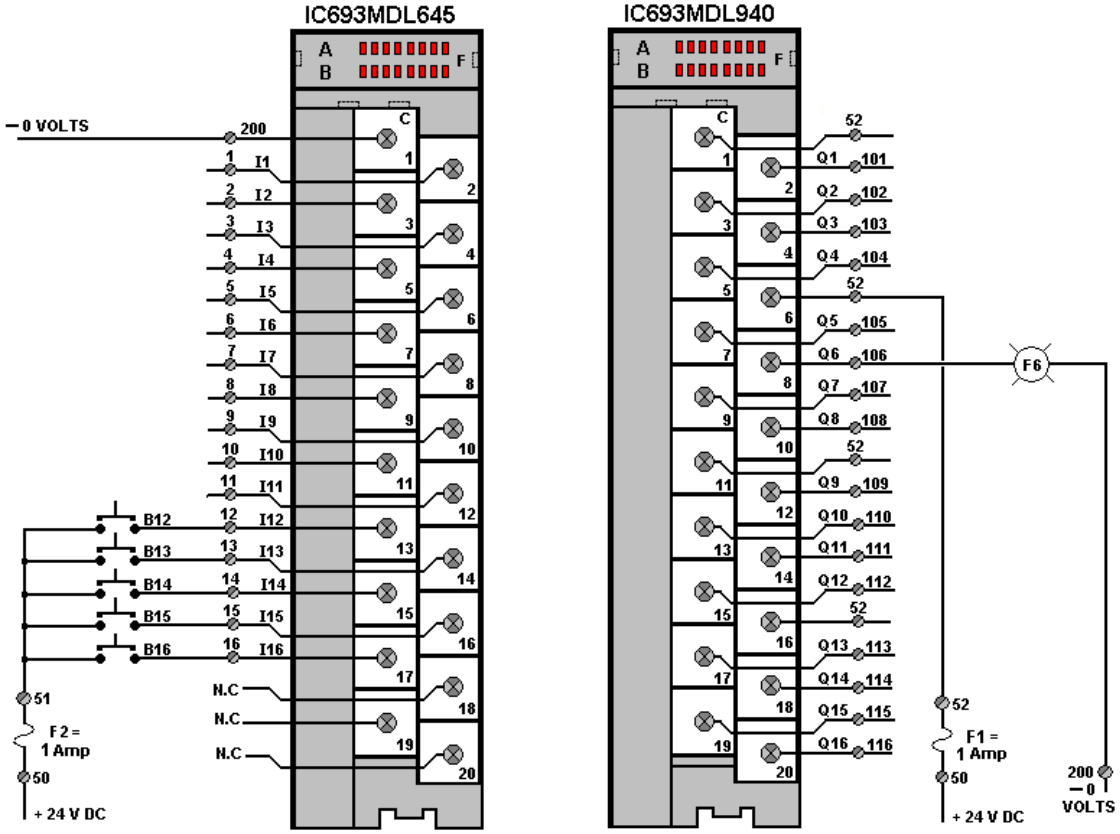
EJERCICIO Nº 6:

En el ejercicio N° 4.6.6 se presenta un circuito con dos botones en serie abiertos y dos botones abiertos en paralelo, de tal manera que para que se energice el foco 5 se requiere que se activen el botón 12 y el botón 13 o bien que se active el botón 15 y el botón 16 además en cualquiera de las combinaciones anteriores se requiere que se active el botón 14.

Para este circuito se requieren cinco entradas y una salida.




```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]
| << RUNG 4  STEP #001 >>
  BOT_12   BOT_13    BOT_14                                        FOCO_6
| %I0012   %I0013    %I0014                                        %Q0006
+---] [------] [--+----] [-----------------------------------------( )--
|BOT_15   BOT_16 |
|%I0015   %I0016 |
+--] [------] [--+
|[      END OF PROGRAM LOGIC      ]
```

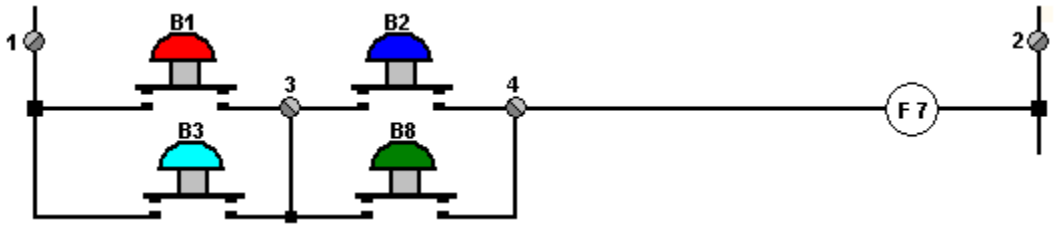


Cableado para el ejercicio N° 4.6.6

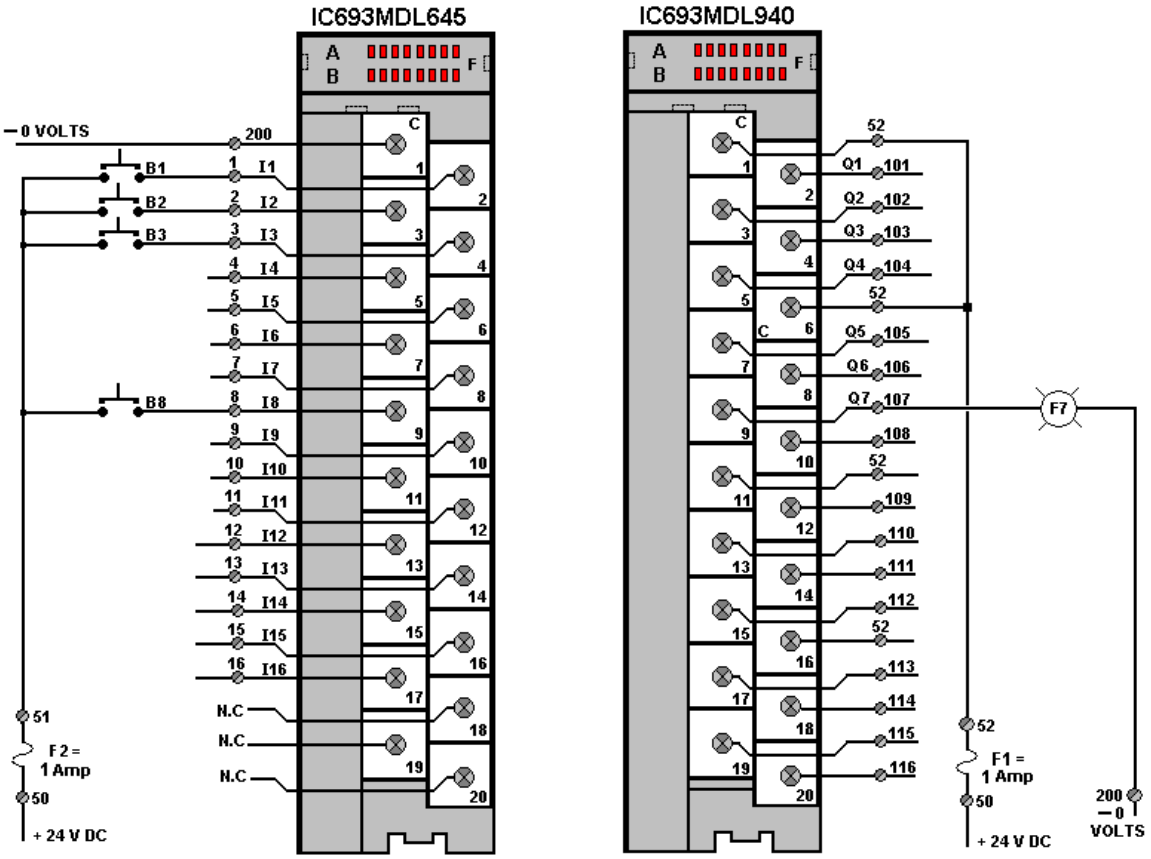
EJERCICIO Nº 7:

En el ejercicio N° 4.6.7 se requiere para que se energice el foco 7 se deberán activar el botón 1 y el botón 2, o bien el botón 1 y el botón 8, también el foco 7 se energiza activando el botón 3 y el botón 2 o bien el botón 3 y el botón 8.

Para este circuito se requieren cuatro entradas y una salida.




```
|[  START OF LD  PROGRAM CURSO   ]
|[     VARIABLE DECLARATIONS     ]
|[       BLOCK DECLARATIONS      ]
|[     START OF PROGRAM LOGIC    ]
| << RUNG 4  STEP #001 >>
   BOTON_1     BOTON_2                                                FOCO_7
|  %I0001      %I0002%                                                 Q0007
+----] [----+----] [----+-----------------------------------------------( )--
|  BOTON_3  |  BOTON_8  |
|  %I0003   |  %I0008   |
+----] [----+----] [----+
|[      END OF PROGRAM LOGIC      ]
```



Cableado para el ejercicio N° 4.6.7

EJERCICIO Nº 8:

Nota: El presente ejemplo se desarrolló en el capitulo #1 con control electromecánico, ahora le daremos una solución programable).

Se tienen dos botones y se tienen cuatro focos, los cuales deberán prender en las siguientes condiciones:

1.- Si los dos botones se encuentran abiertos se enciende el foco número uno.
2.- Si se activa el botón uno se enciende el foco número dos y se apaga el foco uno.
3.- Si se activa el botón dos se apaga el foco dos y se enciende el foco número tres.
4.- Si se activan los dos botones se apaga el foco número tres y se prende el foco número cuatro.

Como se puede observar nunca podrán encender más de un foco y siempre deberá estar encendido al menos uno. Para la solución del presente problema haga uso de los relevadores.

Como podemos observar en este ejercicio se tienen solo dos variables y los tiempos son arbitrarios por lo que es muy sencillo desarrollar las ecuaciones correspondientes. Ver figura 4.6.8 Tomando en cuenta que los botones son con un contacto abierto utilizamos relevadores.

$$F1 = \overline{B1} \cdot \overline{B2} = \overline{R1}\ \overline{R2}$$

$$F2 = B1 \cdot \overline{B2} = R1\ \overline{R2}$$

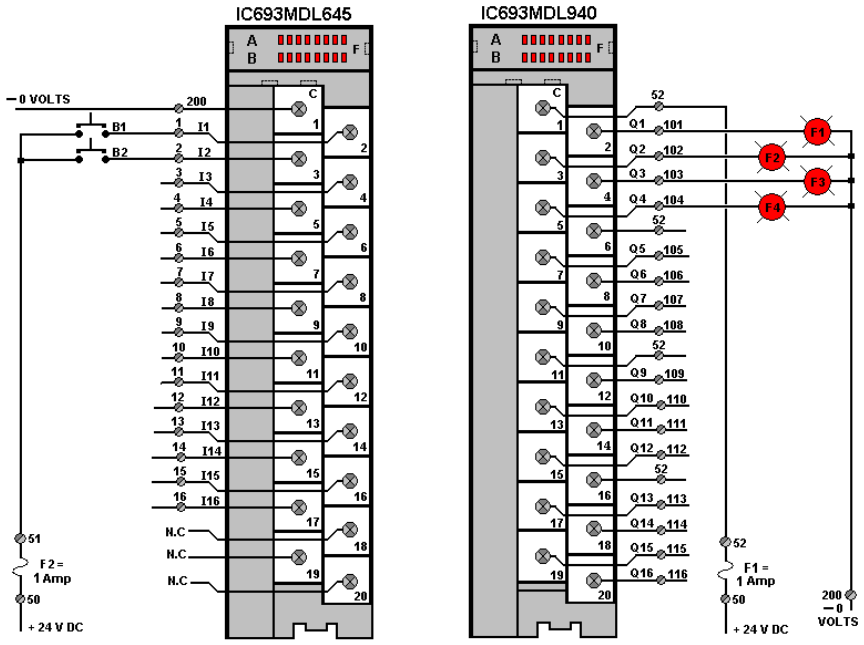
$$F3 = \overline{B1} \cdot B2 = \overline{R1}\ R2$$

$$F4 = B1 \cdot B2 = R1\ R2$$

Figura 4.6.8

Para leer estas ecuaciones seria de la siguiente manera;

1º .- Para que se energice el F1 se requiere que B1 y B2 se encuentren ausentes lo cual se representa con la negación indicada cor una línea en la parte superior.
2º .-Para que se energice el F2 se requiere que B1 se haga presente y que B2 este ausente.
3º .-Para que se energice el F3 se requiere que B1 este ausente y B2 se haga presente.
4º .-Para que se energice el F4 se requiere que B1 y B2 se hagan presentes.



Cableado para el ejercicio N° 4.6.8

A continuación presentamos una primera solución a este problema. Observe como en este ejercicio se están utilizando relevadores internos (M), los cuales no son cableados y pertenecen al sistema.

SOLUCIÓN A:

```
|[  START OF LD  PROGRAM CURSO8   |
| << RUNG 4  STEP #0001 >>
|BOTON_1
|%I0001                                                          %M0001
+--] [-----------------------------------------------------------( )--
| << RUNG 5  STEP #0003 >>
|BOTON_2
|%I0002                                                          %M0002
+--] [-----------------------------------------------------------( )--
| << RUNG 6  STEP #0005 >>
|                                                                FOCO_1
|%M0001  %M0002                                                  %Q0001
+--]/[-----]/[-----------------------------------------------------( )--
| << RUNG 7  STEP #0008 >>
|                                                                FOCO_2
|%M0001  %M0002                                                  %Q0002
+--] [-----]/[-----------------------------------------------------( )--
| << RUNG 8  STEP #0011 >>
|                                                                FOCO_3
|%M0001  %M0002                                                  %Q0003
+--]/[-----] [-----------------------------------------------------( )--
| << RUNG 9  STEP #0014 >>
|                                                                FOCO_4
|%M0001  %M0002                                                  %Q0004
+--] [-----]/[-----------------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```

SOLUCIÓN B:

A continuación presentamos una segunda solución a este problema.

```
|[  START OF LD  PROGRAM CURSO9   ]
|[     VARIABLE DECLARATIONS      ]
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|BOTON_1 BOTON_2                                                 FOCO_1
|%I0001  %I0002                                                  %Q0001
+--]/[-----]/[-----------------------------------------------------( )--
| << RUNG 5  STEP #0004 >>
|BOTON_1 BOTON_2                                                 FOCO_2
|%I0001  %I0002                                                  %Q0002
+--] [-----]/[-----------------------------------------------------( )--
| << RUNG 6  STEP #0007 >>
|BOTON_1 BOTON_2                                                 FOCO_3
|%I0001  %I0002                                                  %Q0003
+--]/[-----] [-----------------------------------------------------( )--
|
| << RUNG 7  STEP #0010 >>
|BOTON_1 BOTON_2                                                  FOCO_4
|%I0001  %I0002                                                   %Q0004
+--] [-----] [-----------------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```

EJERCICIO Nº 9:

El siguiente problema es similar al ejemplo número 8 solo que ahora tendremos tres botones y 8 focos, los cuales trabajaran bajo las siguientes condiciones.

1.- Si los tres botones se encuentran ausentes se prendera el foco 1.
2.- Si se oprime el botón 1 se apaga el foco uno y prende el foco 2.
3.- Si se oprime el botón 2 se apagara el foco 2 y prendera el foco 3.
4.- Si se oprime el botón 3 se apagara el foco 3 y prendera el foco 4.
5.- Si se oprimen los botones 1 y 2 se apagara el foco 4 y prendera el foco 5.
6.- Si se oprimen los botones 1 y 3 se apagara el foco 5 y prendera el foco 6.
7.- Si se oprimen los botones 2 y 3 se apagara el foco 6 y prendera el foco 7.
8.- Si se oprimen los botones 1 y 2 y 3 se apagara el foco 7 y prendera el foco 8.

Como se puede observar nunca podrán encender mas de un foco y siempre deberá estar encendido al menos uno. Para la solución del problema haga uso de los relevadores.

Como podemos observar en este ejercicio se tienen solo tres variables por lo que es muy sencillo desarrollar las ecuaciones correspondientes. Tomando en cuenta que los botones son con un contacto abierto utilizamos relevadores.

NOTA: Sí, no se sigue la secuencia indicada, el resultado es el mismo ya que pertenece a la categoría de los combinacionales.

SOLUCIÓN A:

CONTROL DE FOCOS

```
|[  START OF LD  PROGRAM CURSO11  ]
|[     VARIABLE DECLARATIONS      ]
              V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        BOT_1
          %I0002        BOT_2
          %I0003        BOT_3
          %Q0001        FOCO_1
          %Q0002        FOCO_2
          %Q0003        FOCO_3
          %Q0004        FOCO_4
          %Q0005        FOCO_5
          %Q0006        FOCO_6
          %Q0007        FOCO_7
          %Q0008        FOCO_8
```

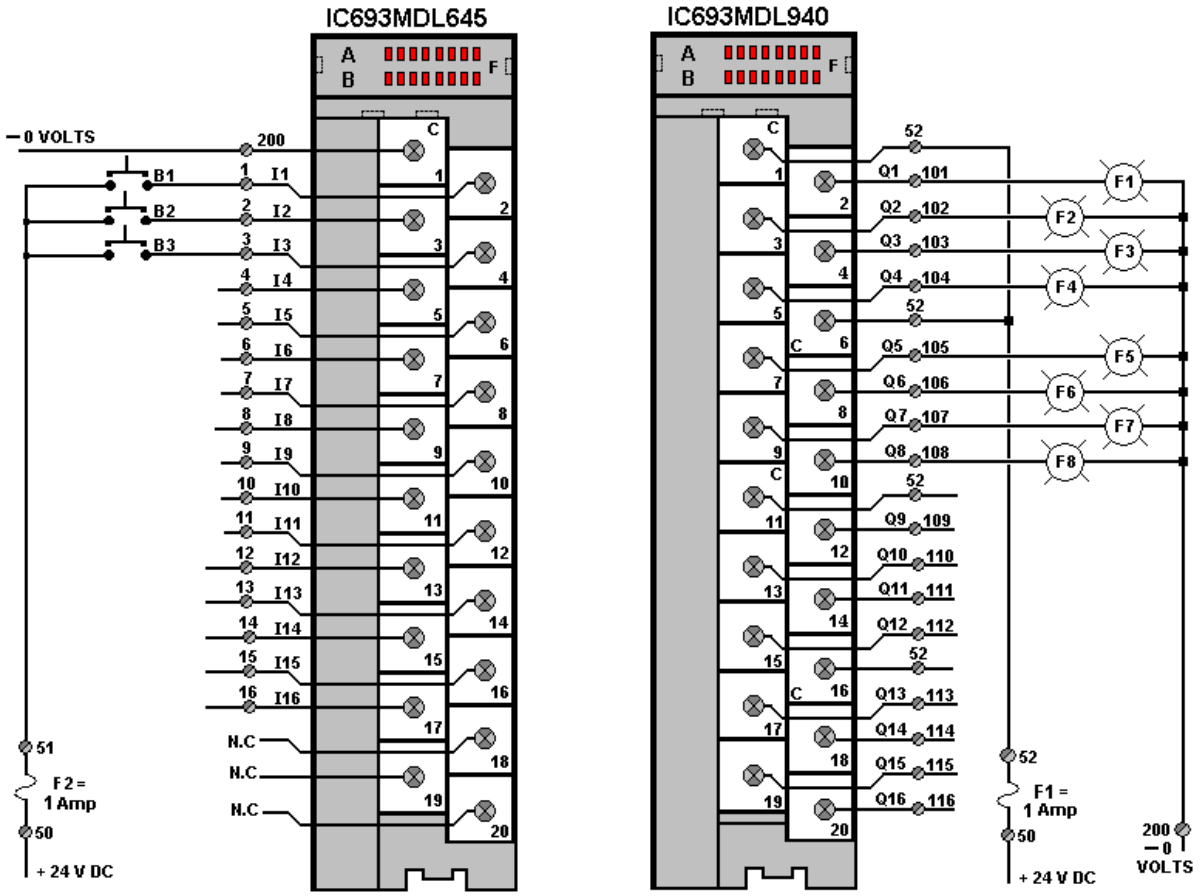
Observe como aquí las **(M)** representan relevadores internos, los cuales se puede decir que no cuestan, en cambio las **(Q)** representan salidas del PLC las cuales representan valor real del PLC.

```
PROGRAM NAME
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
| BOT_1
|%I0001                                                          M0001
+--] [-------------------------------------------------------------( )--
| << RUNG 5  STEP #0003 >>
| BOT_2
|%I0002                                                         %M0002
+--] [-------------------------------------------------------------( )--
| << RUNG 6  STEP #0005 >>
| BOT_3
|%I0003                                                         %M0003
+--] [-------------------------------------------------------------( )--
|
| << RUNG 7  STEP #0007 >>
                                                                FOCO_1
|%M0001  %M0002  %M0003                                         %Q0001
+--]/[-----]/[-----]/[---------------------------------------------( )--
|
| << RUNG 8  STEP #0011 >>
|                                                               FOCO_2
|%M0001  %M0002  %M0003                                         %Q0002
+--] [-----]/[-----]/[---------------------------------------------( )--
|
| << RUNG 9  STEP #0015 >>
|                                                               FOCO_3
|%M0001  %M0002  %M0003                                         %Q0003
+--]/[-----] [-----]/[---------------------------------------------( )--
|
| << RUNG 10  STEP #0019 >>
|                                                               FOCO_4
|%M0001  %M0002  %M0003                                         %Q0004
+--]/[-----]/[-----] [---------------------------------------------( )--
|
| << RUNG 11  STEP #0023 >>
|                                                               FOCO_5
|%M0001  %M0002  %M0003                                         %Q0005
+--] [-----] [-----]/[---------------------------------------------( )--
|
| << RUNG 12  STEP #0027 >>
|                                                               FOCO_6
|%M0001  %M0002  %M0003                                         %Q0006
+--] [-----]/[-----] [---------------------------------------------( )--
|
| << RUNG 13  STEP #0031 >>
|                                                               FOCO_7
|%M0001  %M0002  %M0003                                         %Q0007
+--]/[-----] [-----] [---------------------------------------------( )--
|
| << RUNG 14  STEP #0035 >>
|                                                               FOCO_8
|%M0001  %M0002  %M0003                                         %Q0008
+--] [-----] [-----] [---------------------------------------------( )--
|
| [      END OF PROGRAM LOGIC      ]
```

SOLUCIÓN B:

```
|[  START OF LD  PROGRAM CURSO12  |
|[     VARIABLE DECLARATIONS      ]
             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        BOT_1
          %I0002        BOT_2
          %I0003        BOT_3
          %Q0001        FOCO_1
          %Q0002        FOCO_2
          %Q0003        FOCO_3
          %Q0004        FOCO_4
          %Q0005        FOCO_5
          %Q0006        FOCO_6
          %Q0007        FOCO_7
          %Q0008        FOCO_8
          ----------    ---------------     ---------------------------
          CURSO12        PROGRAM NAME
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_1
|%I0001  %I0002  %I0003                                         %Q0001
+--]/[-----]/[-----]/[-------------------------------------------------( )--
| << RUNG 5  STEP #0005 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_2
|%I0001  %I0002  %I0003                                         %Q0002
+--] [-----]/[-----]/[-------------------------------------------------( )--
| << RUNG 6  STEP #0009 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_3
|%I0001  %I0002  %I0003                                         %Q0003
+--]/[-----] [-----]/[-------------------------------------------------( )--
| << RUNG 7  STEP #0013 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_4
|%I0001  %I0002  %I0003                                         %Q0004
+--]/[-----]/[-----] [-------------------------------------------------( )--
| << RUNG 8  STEP #0017 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_5
 %I0001  %I0002  %I0003                                         %Q0005
+--] [-----] [-----]/[-------------------------------------------------( )--
| << RUNG 9  STEP #0021 >>
| BOT_1   BOT_2   BOT_3                                          FOCO_6
|%I0001  %I0002  %I0003                                         %Q0006
+--] [-----]/[-----] [-------------------------------------------------( )--
| << RUNG 10  STEP #0025 >>
 BOT_1   BOT_2   BOT_3                                          FOCO_7
 %I0001  %I0002  %I0003                                         %Q0007
+--]/[-----] [-----] [-------------------------------------------------( )--
| << RUNG 11  STEP #0029 >>
|  BOT_1   BOT_2  BOT_3                                          FOCO_8
 %I0001  %I000  %I0003                                          %Q0008
+--] [-----] [-----] [-------------------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
```

A continuación se presenta el diagrama de conexiones de los módulos de entradas y de salidas del PLC



Cableado para el ejercicio N° 4.6.9

EJERCICIO Nº 10:

Se desea automatizar una sala de cine, para ello se realizó el estudio de la generación de calor por persona cuando la sala esta al 33%, 66%, y 99% de su capacidad, así como el consumo de aire de acuerdo a el área instalada y se llego ha las siguientes conclusiones;

1º.- Si el cine se encuentra con una concurrencia de menos del 33% de su capacidad, la temperatura deberá estar aproximadamente a 20º C, lo cual es una temperatura agradable y no hay necesidad de ningún control.

2º.- Si la asistencia es arriba del 33%, la temperatura llega a 22º C y deberá entrar a trabajar un sistema de aire lavado con un motor de 5 H.P, que le vamos ha llamar "motor pequeño", con este motor deberá bajar la temperatura de 22 º C y con ello parar el motor.

3º.- Si la asistencia a la sala de cine es del 66 % la temperatura sube hasta 24 º C lo cual significa que la máquina pequeña no pudo con la carga térmica y deberá salir y entrar una máquina más grande de 10 H. P, "le vamos ha llamar motor grande", la cual deberá bajar la temperatura.

4º.- Si a la sala de cine entran el 99 % la temperatura sube hasta 26 º C esto significa que la máquina grande tampoco pudo con la carga térmica y deberá entrar a ayudar la máquina pequeña.

Como condición se pide que para que trabajen las máquinas deberá estar circulando agua en las máquinas como sistema de enfriamiento para que no se quemen.

SOLUCIÓN:

Para la solución del presente problema primero debemos identificar las variables a controlar:

A = Detector de flujo de agua en el sistema.

T1= Termostato para detectar la temperatura de 22 º C.

T2= Termostato para detectar la temperatura de 24 º C.

T3= Termostato para detectar la temperatura de 26 º C

Ahora podemos proceder a realizar las correspondientes ecuaciones tomando en cuenta que son pocas las variables.

$M1 = (AT1\overline{T2} + AT3)$ = Si factorizamos $M1 = A (T1 \overline{T2} + T3)$

$M2 = AT2$

Es importante saber leer las ecuaciones considerando, no las literales, sino, las variables lógicas que se están utilizando; una manera de leer estas ecuaciones seria:

Para que se energice la máquina 1 que es una máquina pequeña se requiere primero que haya agua circulando en el sistema, además que se haga presente la temperatura uno, esto es que la temperatura llegue a 22 º C y que no se haga presente la temperatura T2 esto es, que no alcance los 24 º C; También podrá entrar la máquina uno si hay agua circulando en el sistema y la temperatura T3 se hace presente o sea que llegue hasta 26 º C.

Para que entre la máquina dos se requiere que haya agua circulando en el sistema y que se haga presente la temperatura T2 esto es, que llegue hasta 24 º C.

Primero vamos a mostrar en Figura 4.6.9 la simbología para los sensores utilizados;

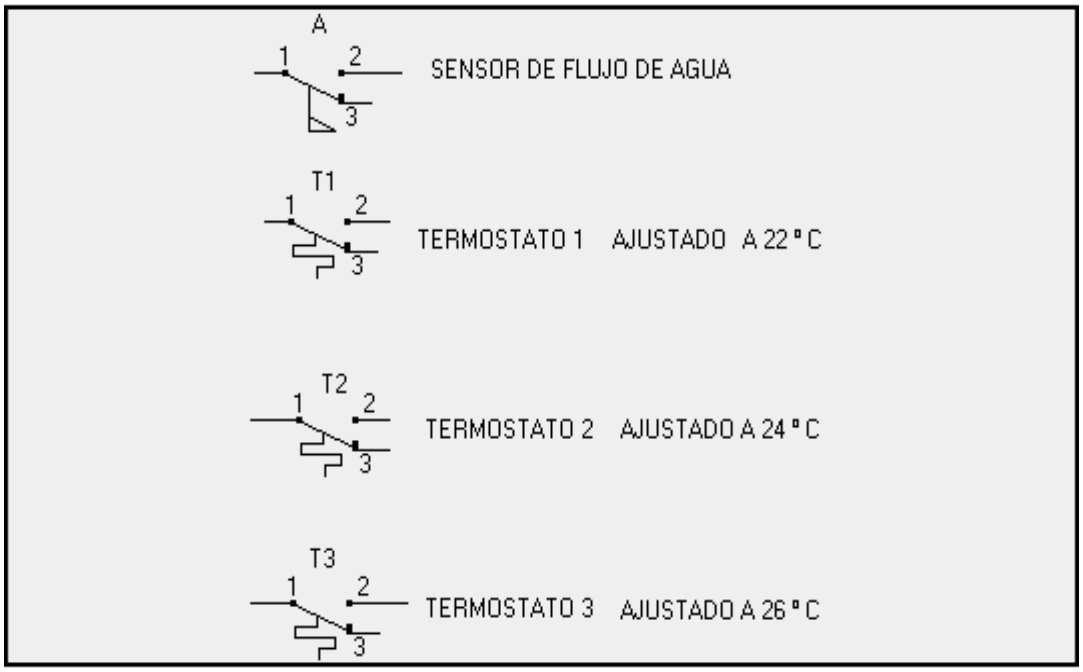
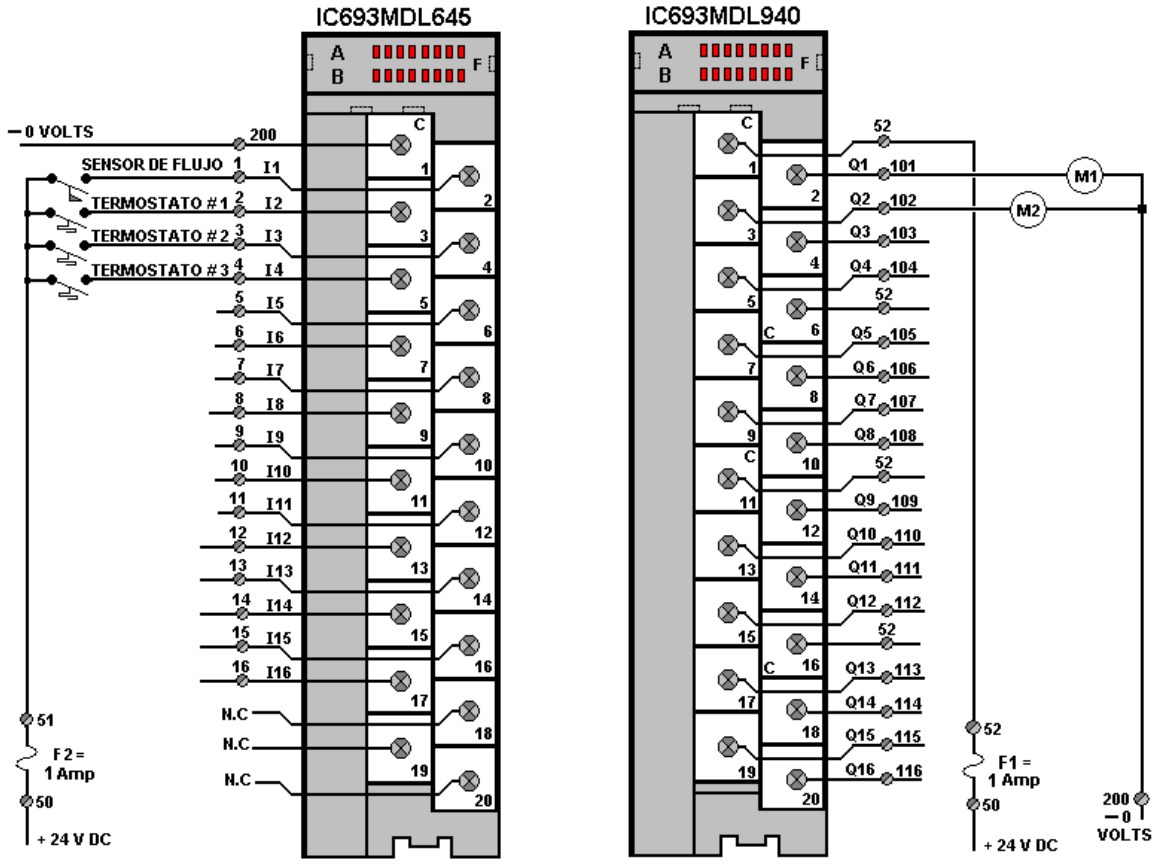


Figura 4.6.9

A continuación presentamos una solución con control programable.

```
        CONTROL DE AIRE ACONDICIONADO

|[  START OF LD  PROGRAM CURSO10  ]
|[     VARIABLE DECLARATIONS      ]
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]

| << RUNG 4  STEP #0001 >>
|
|  I1      I2       I3                                          MAQ_1
|SENS_FL TERMO_1 TERMO_2                                      %Q0001
+--] [-----] [-----]/[--+-------------------------------------( )--
|                        |
|  I1       I4           |
|SENS_FL TERMO_3         |
+--] [-----] [----------+
|
| << RUNG 5  STEP #0008 >>
|
|   I1     I3                                                  MAQ_2
|SENS_FL TERMO_2                                              %Q0002
+--] [-----] [-----------------------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
|
```



Cableado para el ejercicio N° 4.6.10

4.7 - Formas de insertar, cortar y pegar contactos en un circuito.

En el presente tema analizaremos la forma de insertar contactos o insertar líneas o eliminar contactos o eliminar líneas o eliminar programas completos, o importar información de un fólder a otro.

A continuación se presenta la forma de modificar una variable en una línea.
Sí se tiene una línea como la siguiente y se desea modificar un contacto de I3 por I6.

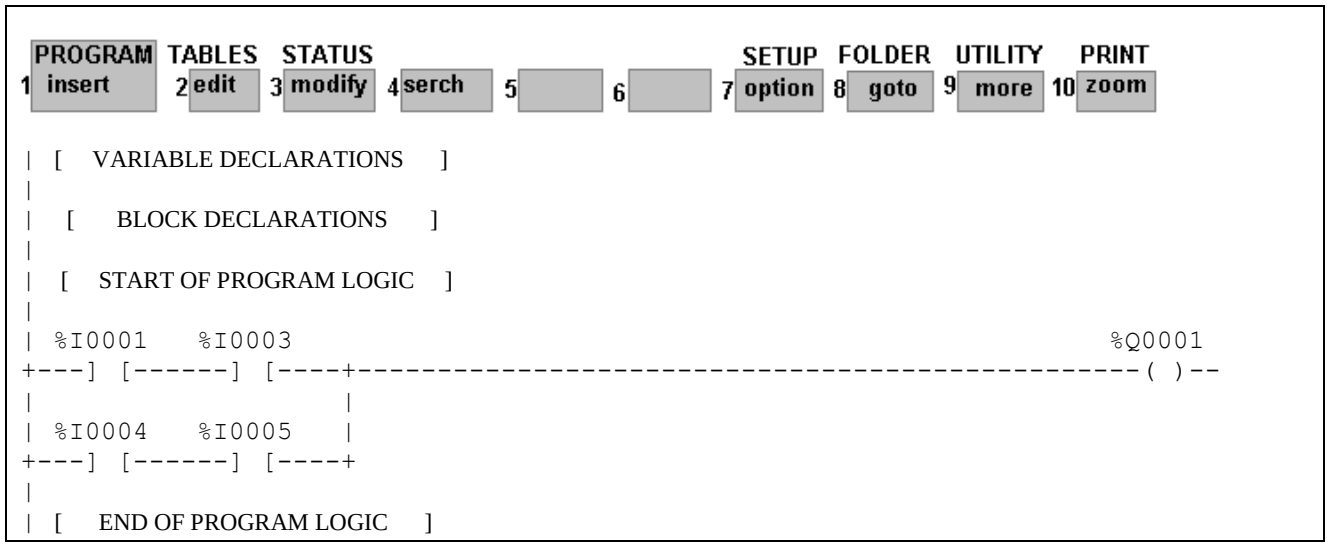
Primeramente debemos de tomar en cuenta que en línea esto es, estando en modo de **RUN** se pueden realizar dos pasos:

1º.- Cambiar una variable, esto es, una I1 por una I3 ó M5 ó una Q3.
2º.- Modificar un contacto, esto es cambiar un contacto abierto por uno cerrado y viceversa.

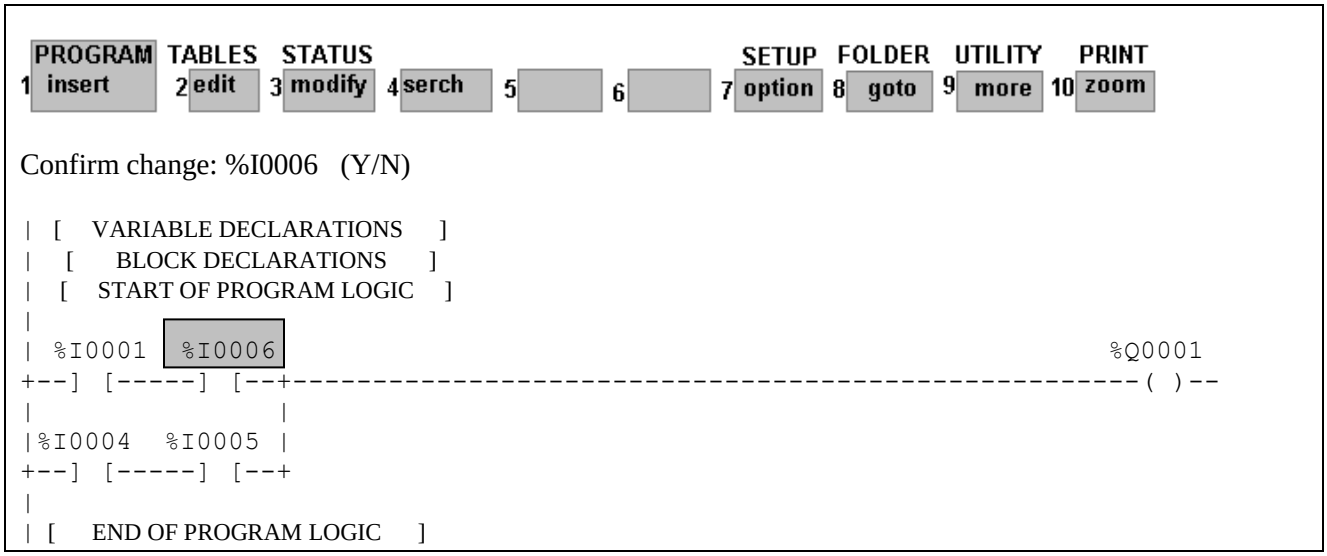
Que es lo que **no se puede hacer en modo de RUN**.

1º.- Borrar una función.
2º.- Insertar una función.

EJERCICIO N° 4.7.1 (Cambiar una variable en modo de RUN).



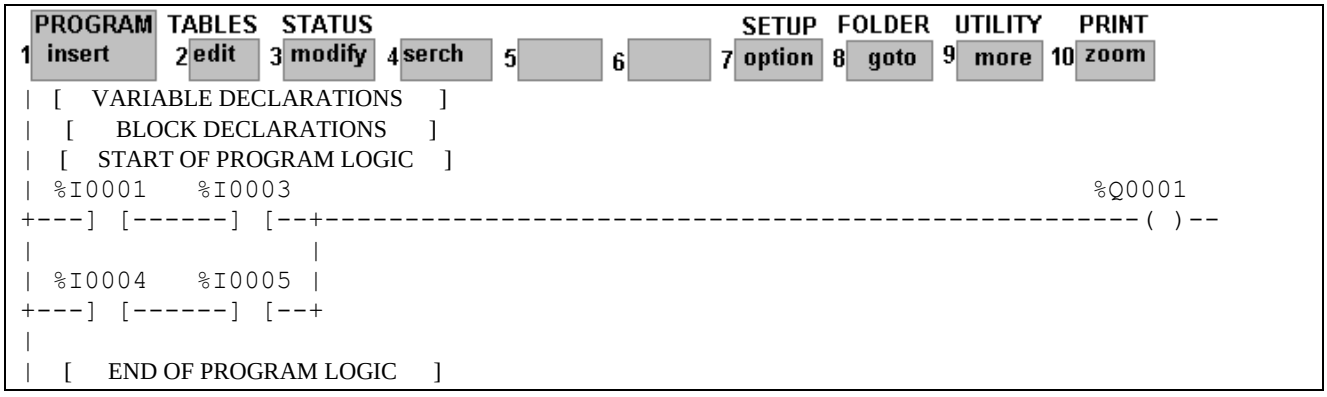
Para cambiar una variable simplemente se coloca el cursor en la variable a modificar en este caso se coloca el cursor en I3 y se pone I6 y oprimimos la tecla ENT con lo cual nos pregunta el programa si confirmamos el cambio a lo cual contestamos oprimiendo la tecla **Y**.



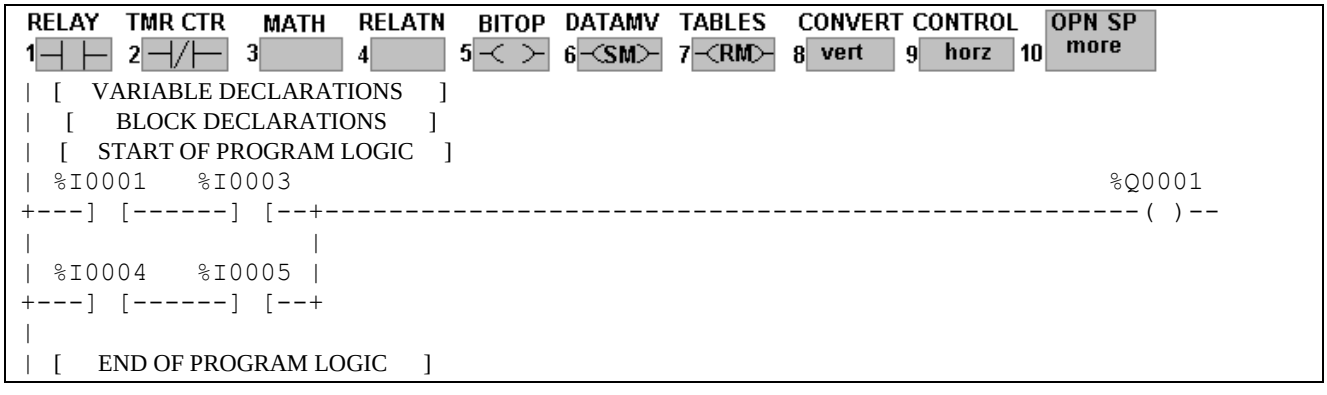
EJERCICIO N° 4.7.2 (Modificar un contacto abierto por un cerrado y viceversa en modo RUN).

A continuación se presenta la forma de modificar un contacto abierto por un cerrado o viceversa en una línea.

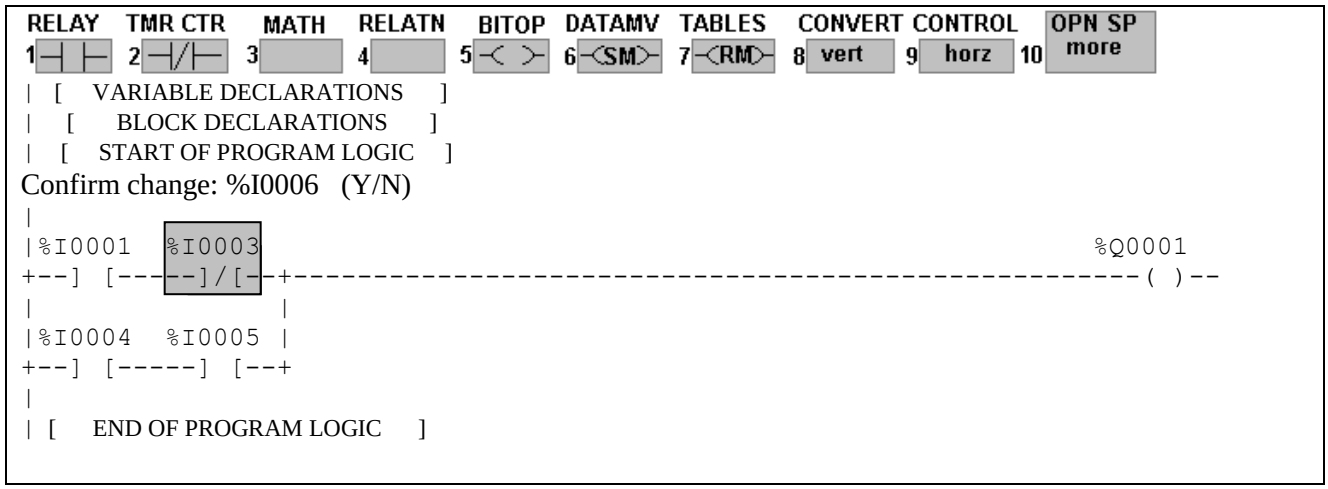
Sí se tiene una línea como la siguiente y se desea modificar el contacto abierto de I3 por un cerrado del mismo I3.



Primeramente oprimimos la tecla F3 (modify.) y nos aparece la siguiente pantalla.



Posteriormente y estando el cursor en el contacto I3 se oprime la tecla F2 contacto cerrado).



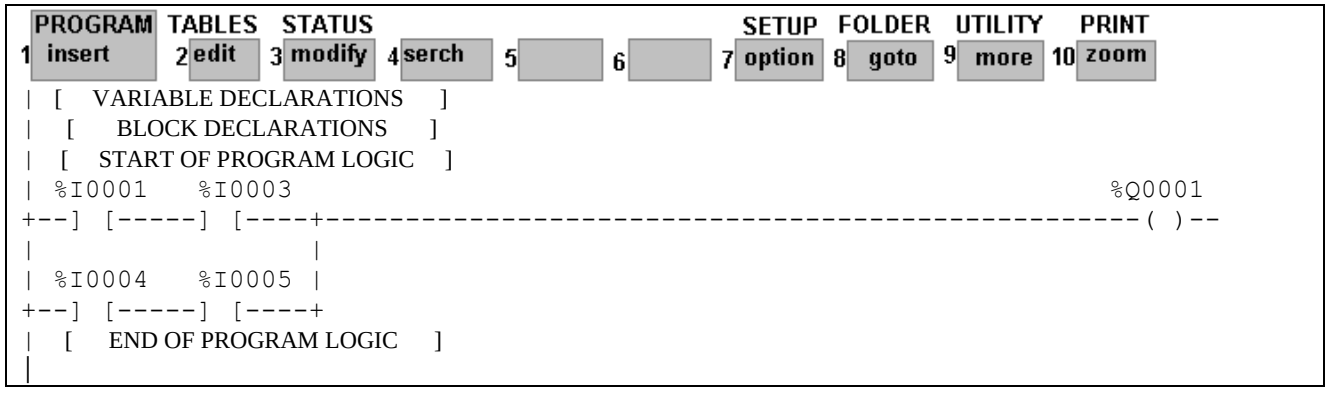
Finalmente oprimimos la tecla ENTER para terminar la operación.

EJERCICIO N° 4.7.3 (Insertar un contacto en paralelo).

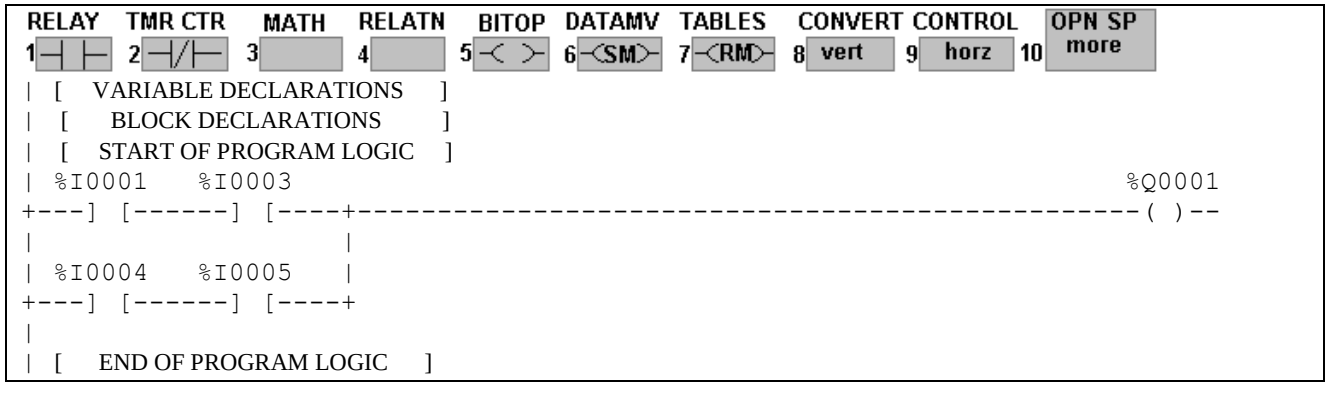
NOTA: Los siguientes ejercicios se pueden realizar solo en modo de **STOP**

A continuación se presenta la forma de insertar contactos en una línea.

Sí se tiene una línea como la siguiente y se desea introducir un contacto en paralelo con I3.

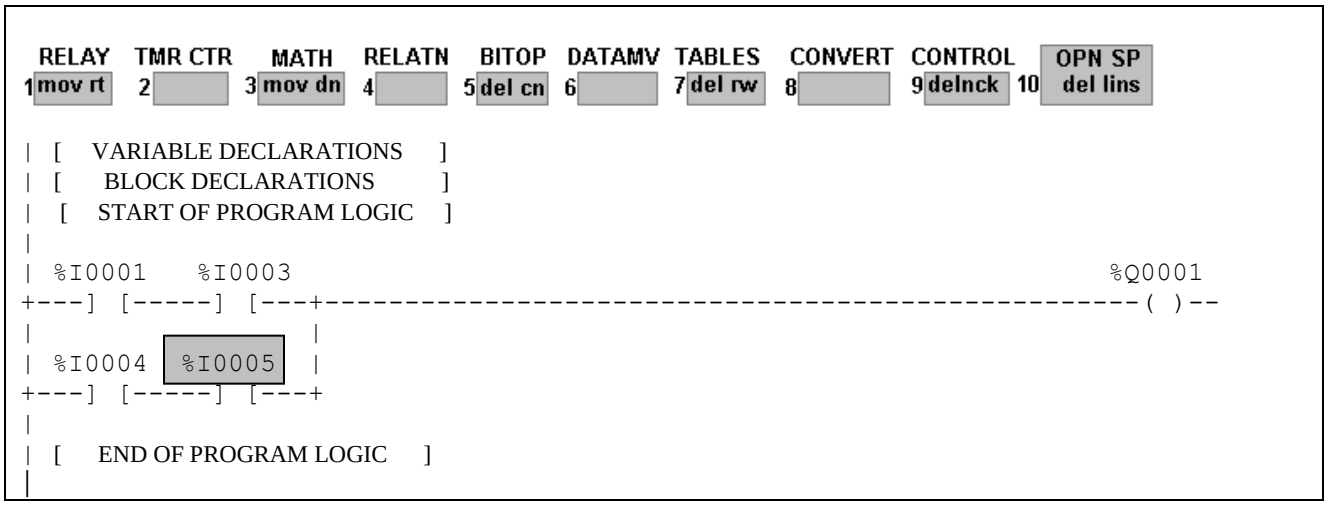


Primero procedemos a editar la línea con F2 tal como lo muestra la pantalla siguiente.

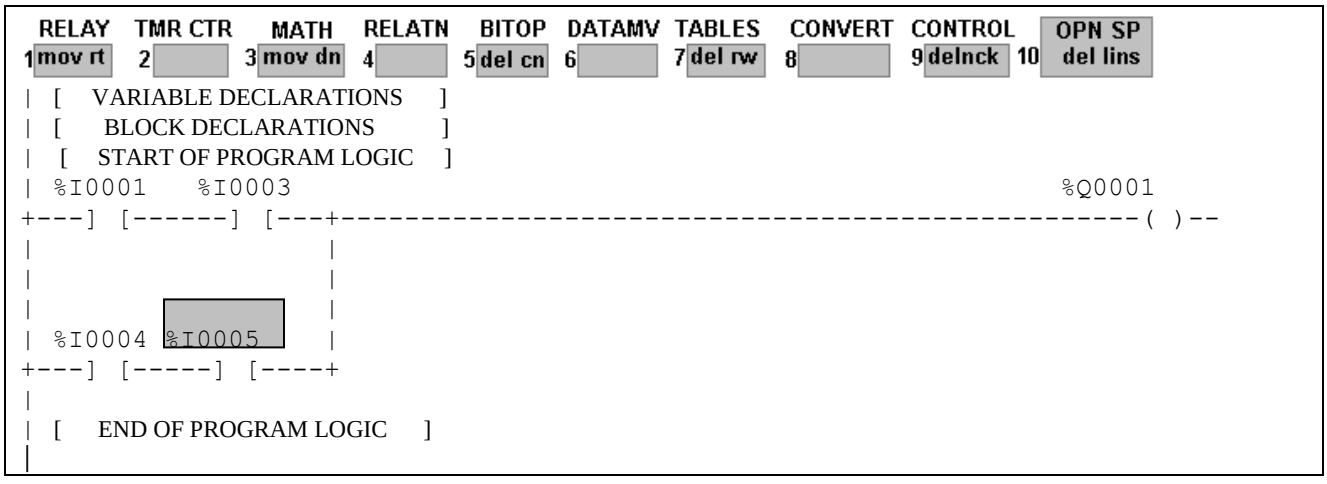


En seguida colocamos el cursor en I5 para bajar la línea donde se encuentra los contactos I4 y I5 para poder colocar el contacto en paralelo con I3, y oprimimos las teclas shft + F10 (OPN SP), con lo cual nos muestra la siguiente pantalla.

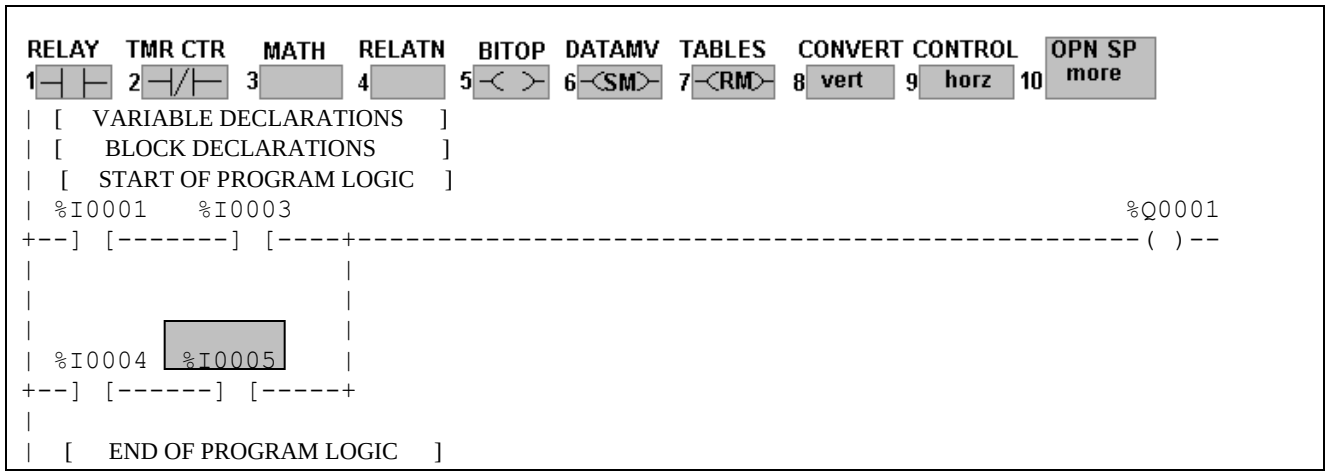
NOTA: Recuerde que los iconos de la parte superior de la pantalla que se encuentran sin sombrear se activan oprimiendo primero la tecla shft y luego la función correspondiente y las que se encuentran sombreados directamente la función (F1, 2, 3 etc).



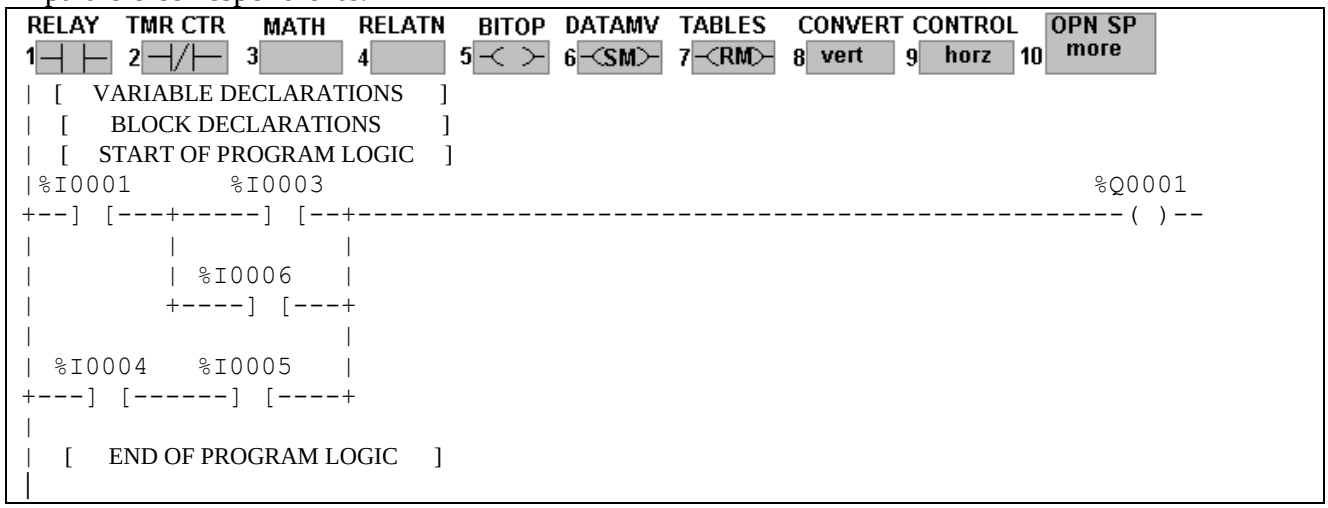
Luego procedemos a oprimir la tecla F3 (mov dn), con lo cual nos muestra la pantalla siguiente donde observamos que se abrió un espacio para colocar el contacto.



Ahora sí, ya tenemos el espacio para colocar el contacto que queremos; como en los iconos mostrados, no se encuentran contactos, oprimimos la tecla shft F1 (RELAY), mostrándonos la siguiente pantalla.



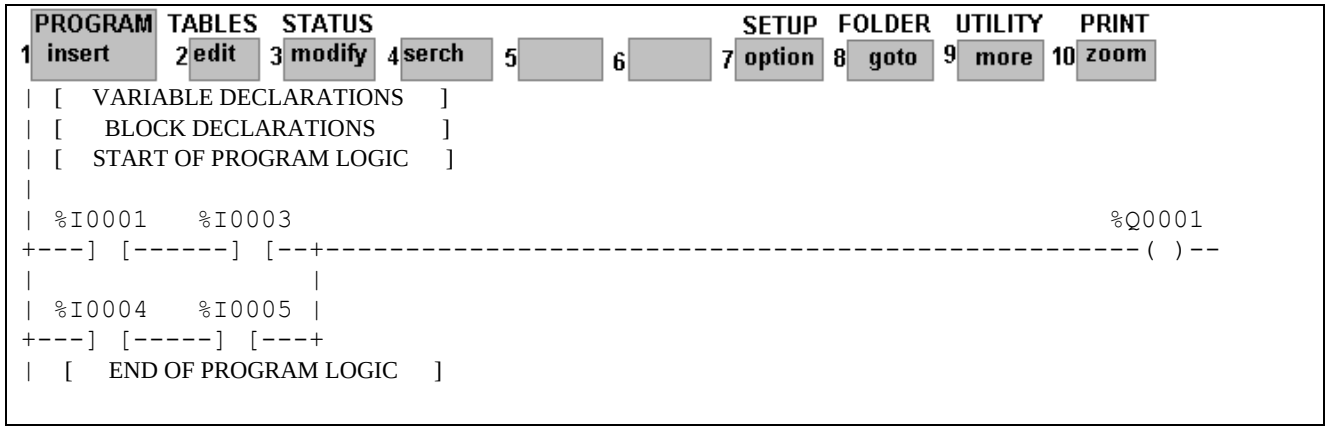
Ahora si, procedemos a colocar nuestro contacto colocando el cursor debajo de I3 y poniendo su paralelo correspondiente.



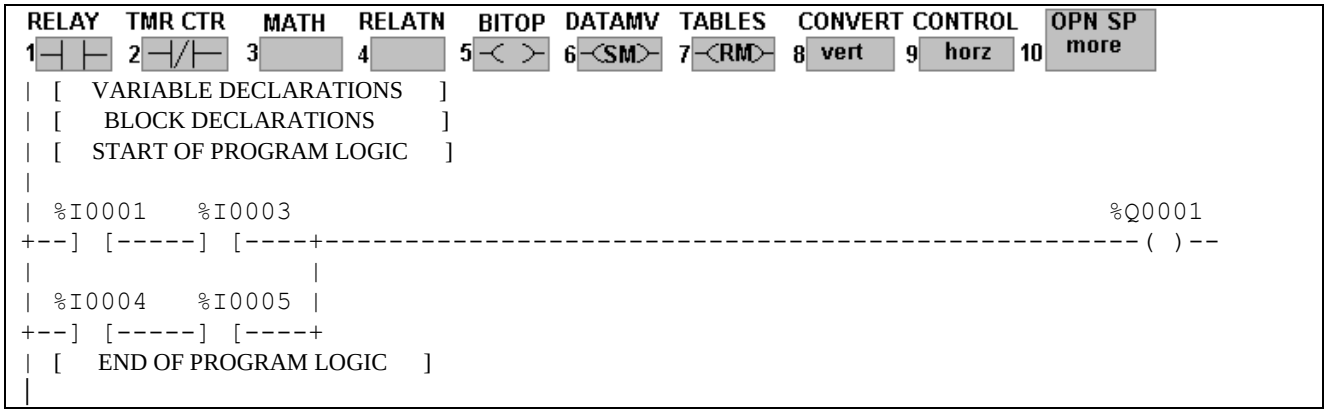
EJERCICIO N° 4.7.4 (Insertar un contacto en serie).

A continuación se presenta la forma de insertar contactos en una línea.

Sí se tiene una línea como la siguiente y se desea introducir un contacto en serie con I1 y I3.

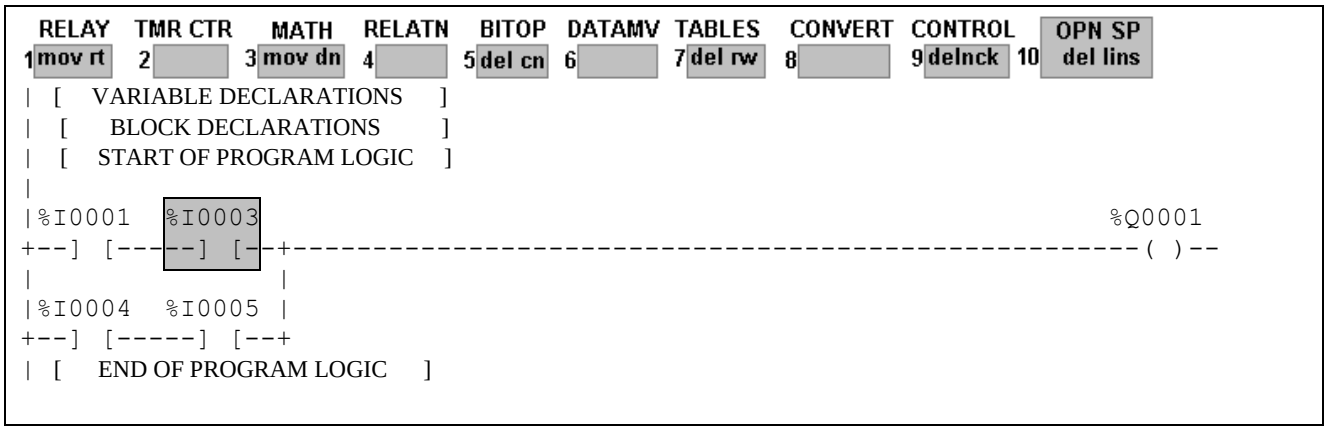


Primero procedemos a editar la línea con F2 tal como lo muestra la pantalla siguiente

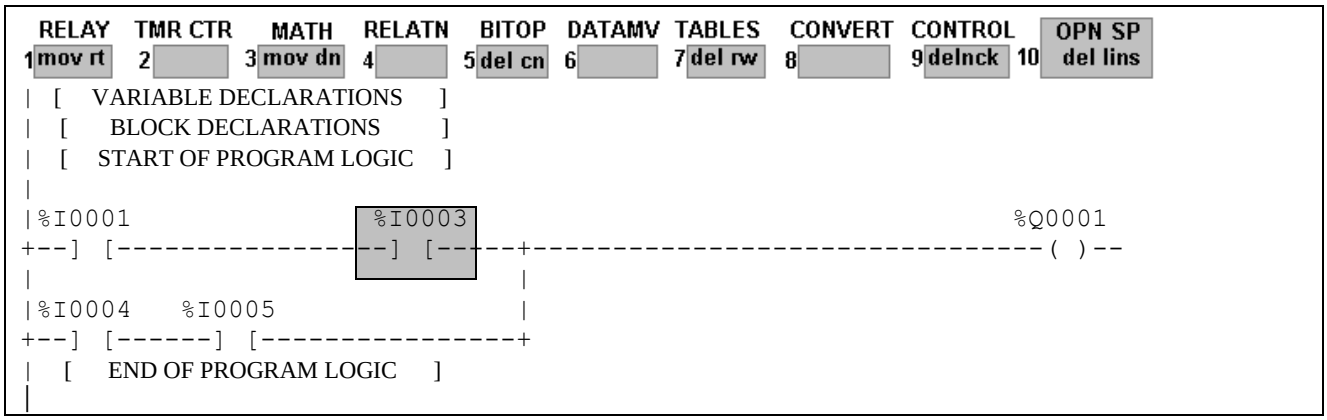


En seguida colocamos el cursor en I3 para mover la línea donde se encuentra los contactos I1 y I3 para poder colocar el contacto en serie con I3, y oprimimos las teclas shft + F10 (OPN SP), con lo cual nos muestra la siguiente pantalla.

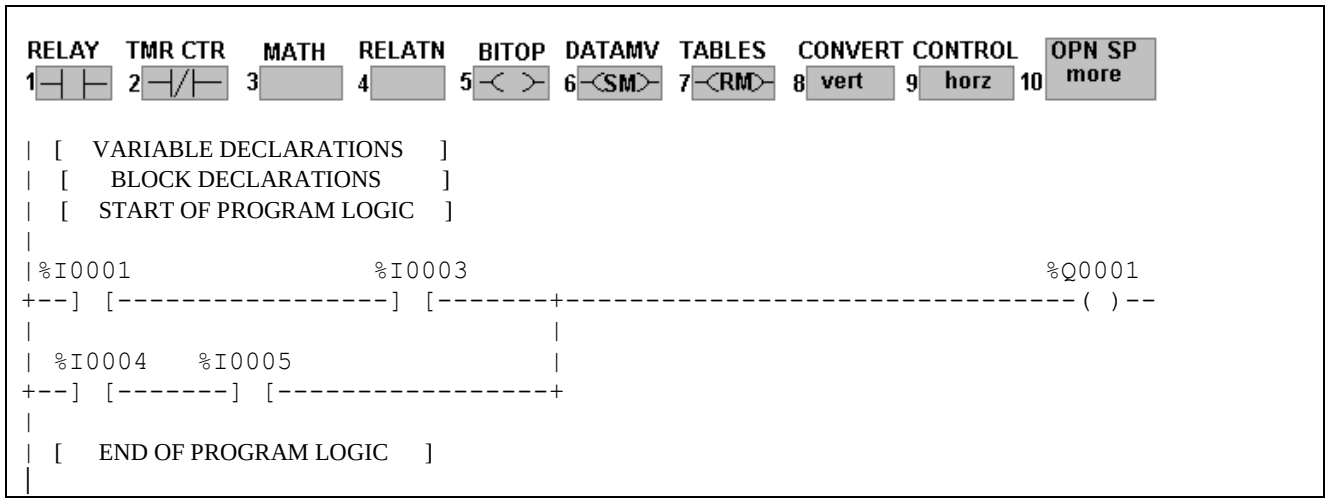
NOTA: Recuerde que los iconos de la parte superior de la pantalla que se encuentran sin sombrear se activan oprimiendo primero la tecla shft y luego la función correspondiente y las que se encuentran sombreados directamente la función (F1, 2, 3 etc).



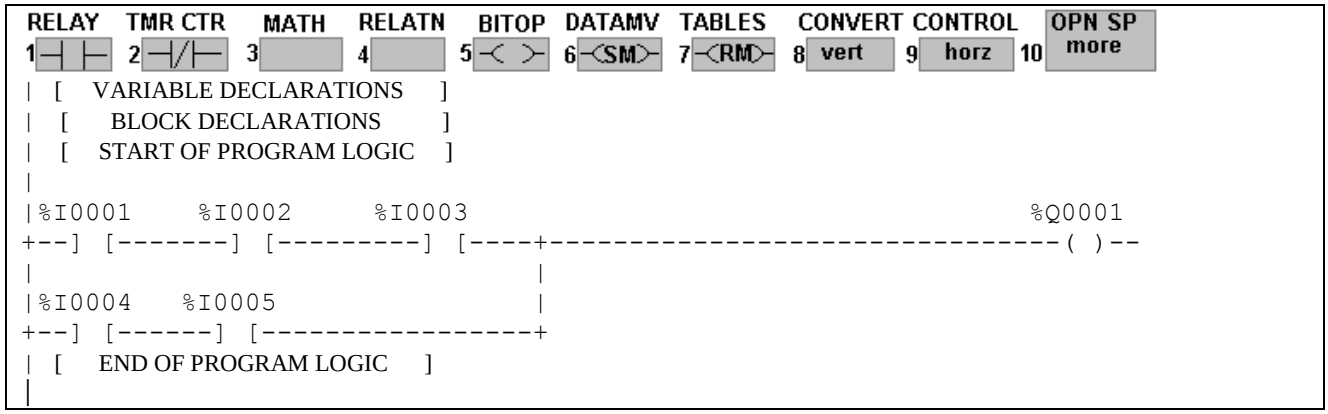
Luego procedemos a oprimir la tecla F1 (mov rt), con lo cual nos muestra la pantalla siguiente donde observamos que se abrió un espacio para colocar el contacto.



Ahora sí, ya tenemos el espacio para colocar el contacto que queremos; como, en los iconos mostrados no se encuentran contactos, oprimimos la tecla shft F1 (RELAY), mostrándonos la siguiente pantalla.



Ahora si, procedemos a colocar nuestro contacto colocando el cursor inmediatamente después de I1 y poniendo su contacto correspondiente.

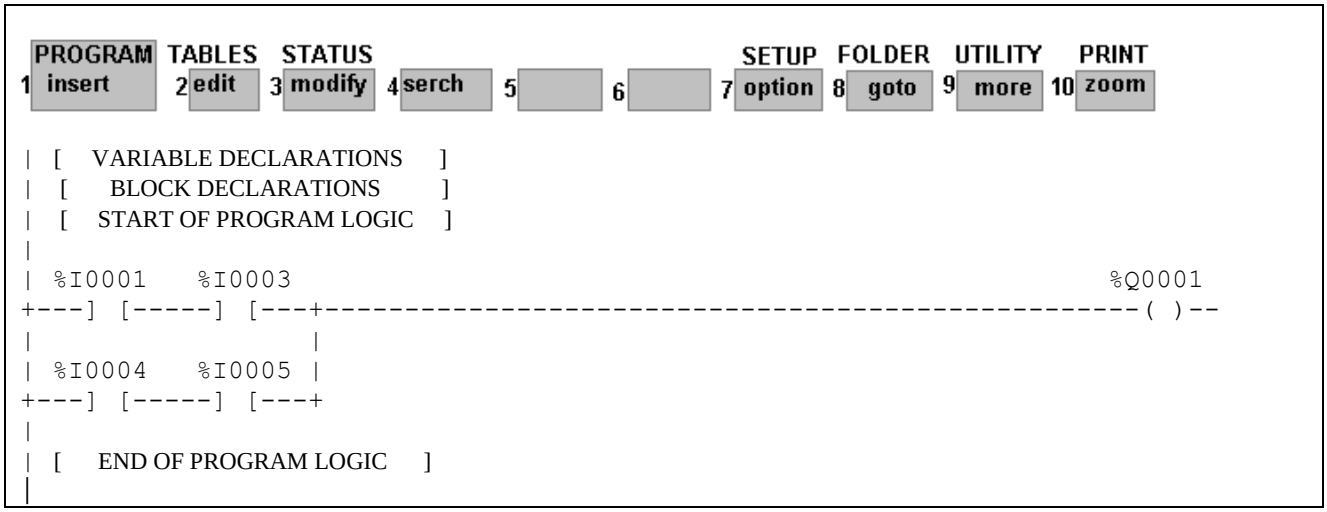


EJERCICIO N° 4.7.5 (Eliminar un contacto).

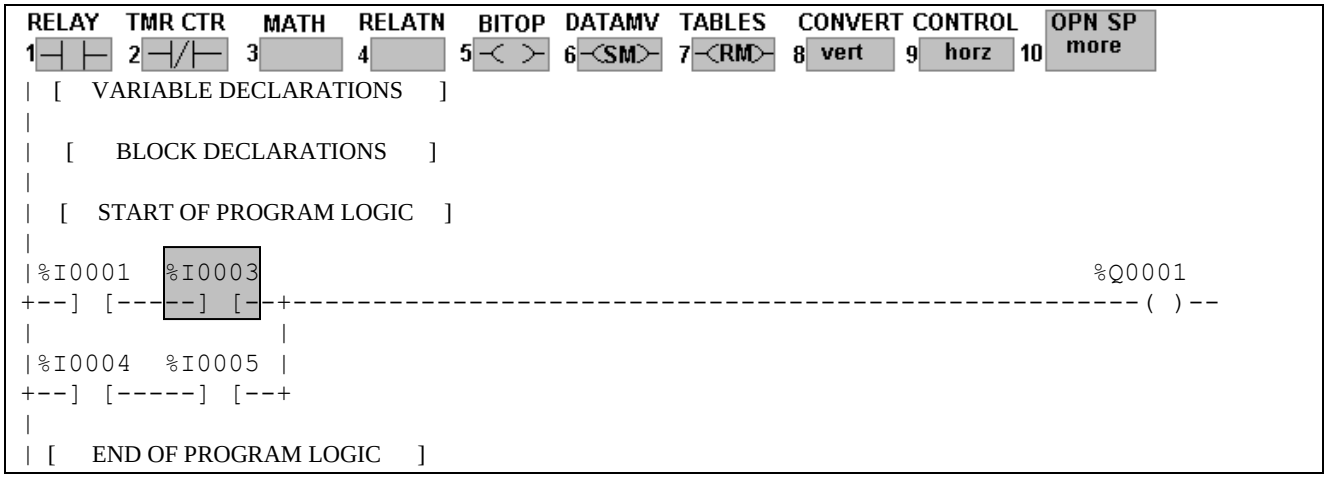
A continuación veremos la forma de eliminar un contacto dentro de una línea. Recuerde que a diferencia del control electromecánico donde cada eslabón correspondía a una línea, en el control programable una línea corresponde a todos los elementos que intervienen en el control de una bobina.

Primero para eliminar un contacto de una línea, se coloca el cursor en el elemento que se quiere eliminar y se edita la línea con **F2**, luego se oprimen las teclas **"ALT D "** o bien F9 que es una línea horizontal. (Ver funcionamiento de teclas rápidas en ANEXO 1).

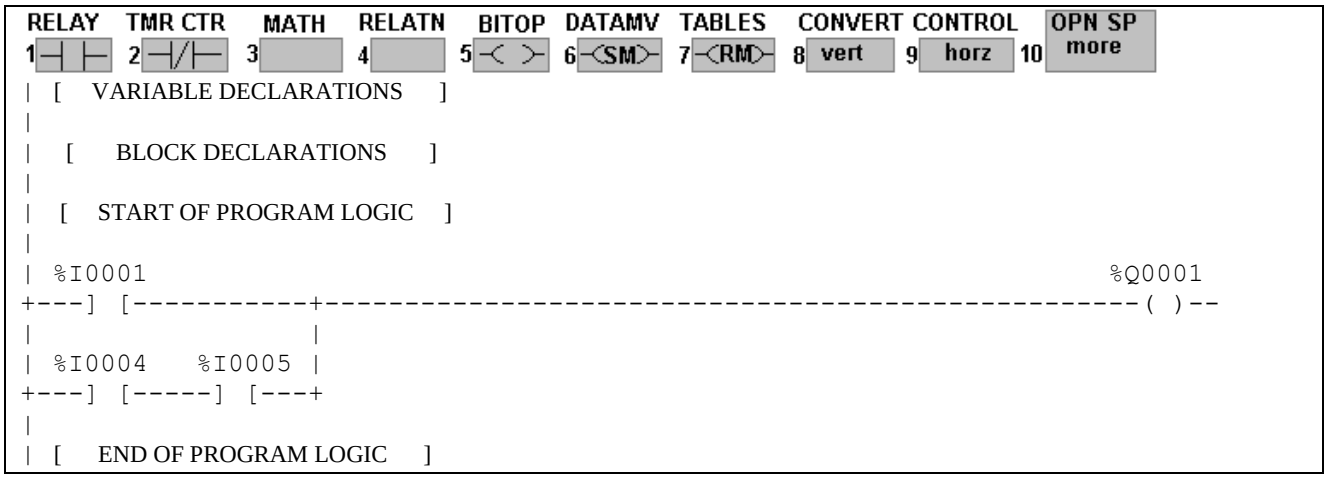
Observe la siguiente línea donde se requiere eliminar el contacto de I3



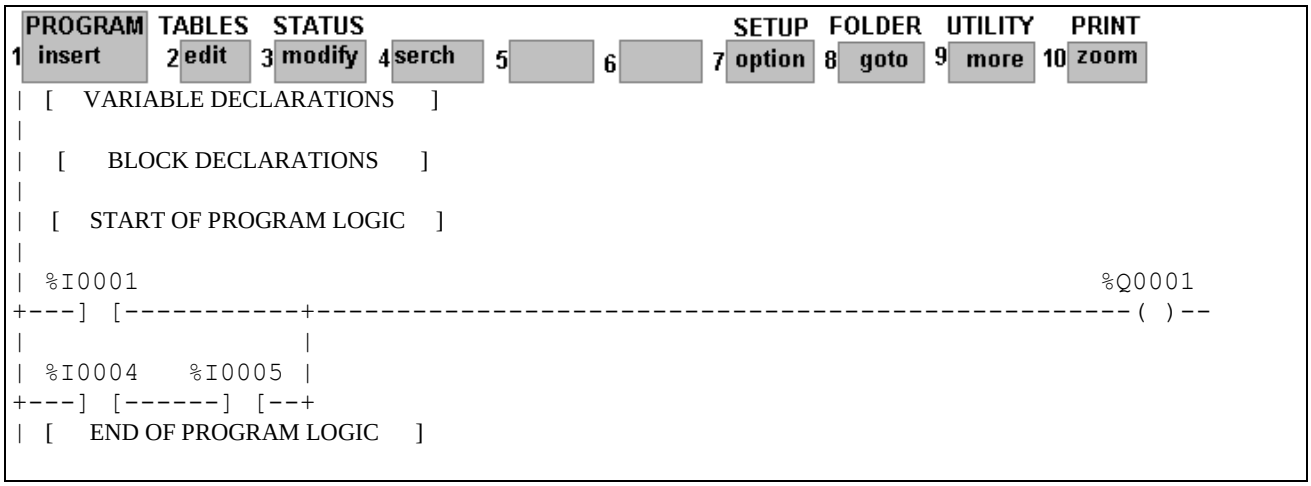
Primeramente con la tecla F2 (**EDIT),** luego colocamos el cursor en I3.



En seguida se oprimen las teclas ALT + D y se elimina el contacto, o bien esto mismo lo podemos realizar con la tecla F9 (horz) borrando el elemento y se procede a colocar la línea horizontal en el hueco que quedo al eliminar el contacto de **I3**.

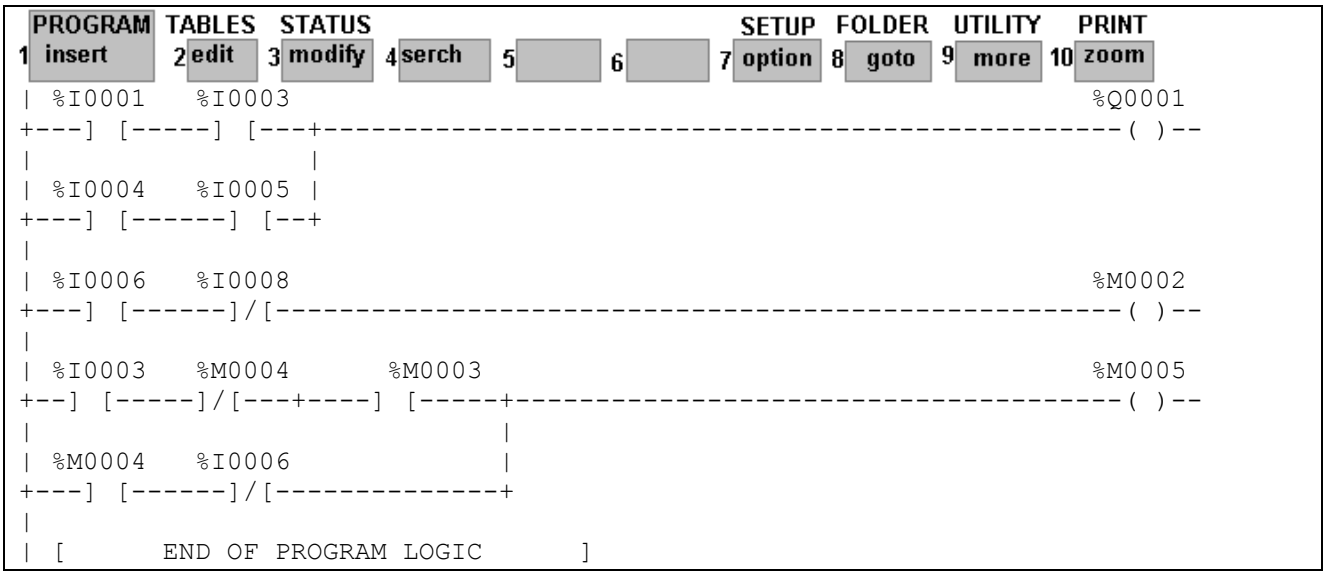


Finalmente oprimimos la tecla de ESC para terminar.

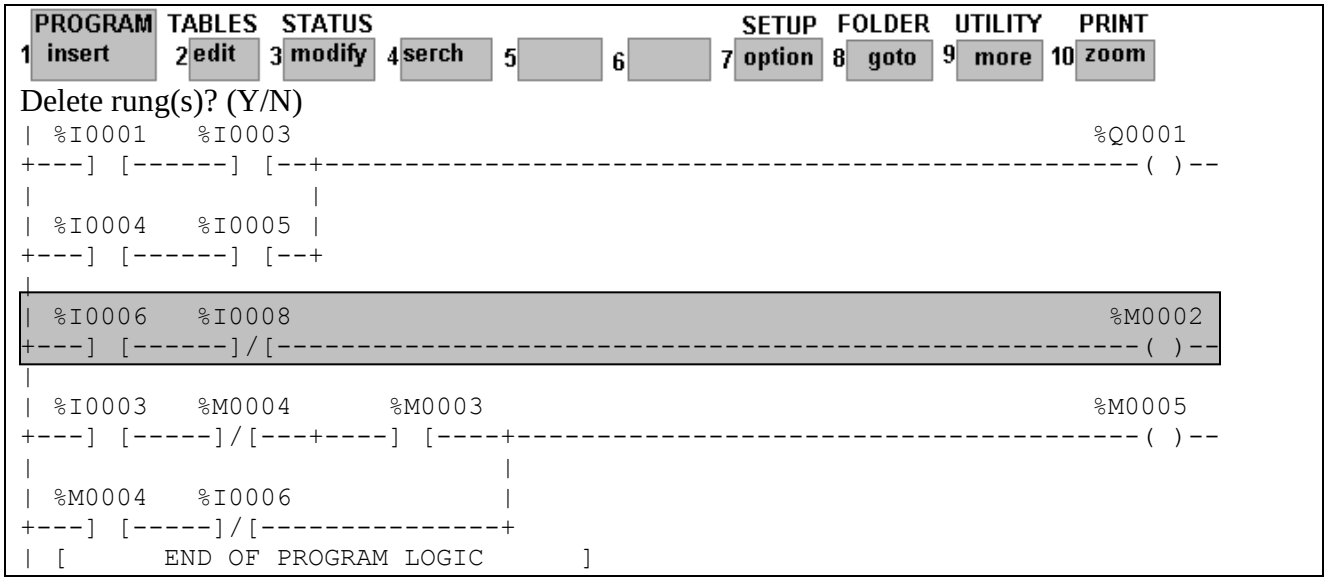


EJERCICIO N° 4.7.6 (Eliminar una línea).

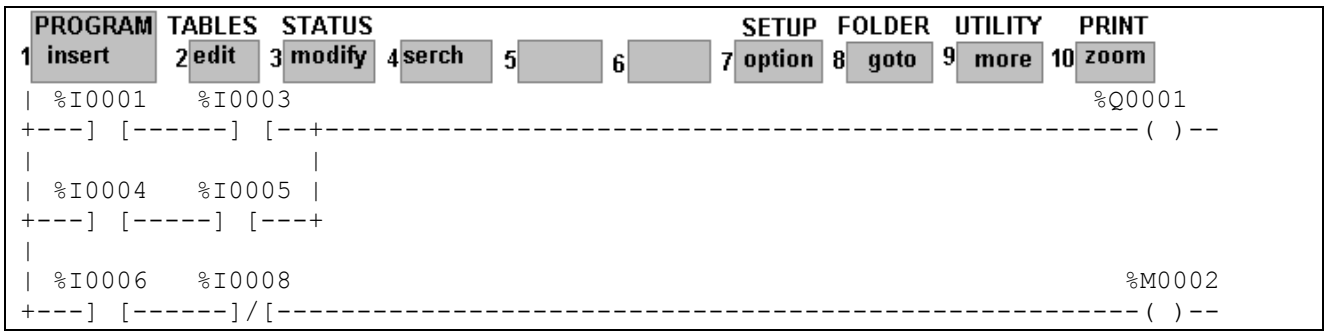
Para eliminar una línea completa se coloca el cursor en cualquier elemento de la línea que se desea eliminar y sin editar se oprimen las teclas rápidas ALT+D con lo cual se elimina la línea completa. Ver la pantalla siguiente donde se desea eliminar la línea que controla la bobina de M2.



Colocamos el cursor en cualquier elemento de la línea que controla la bobina de M2 y sin editar activamos las teclas rápidas **ALT+D** y nos muestra un mensaje sí, deseamos borrar la línea completa.



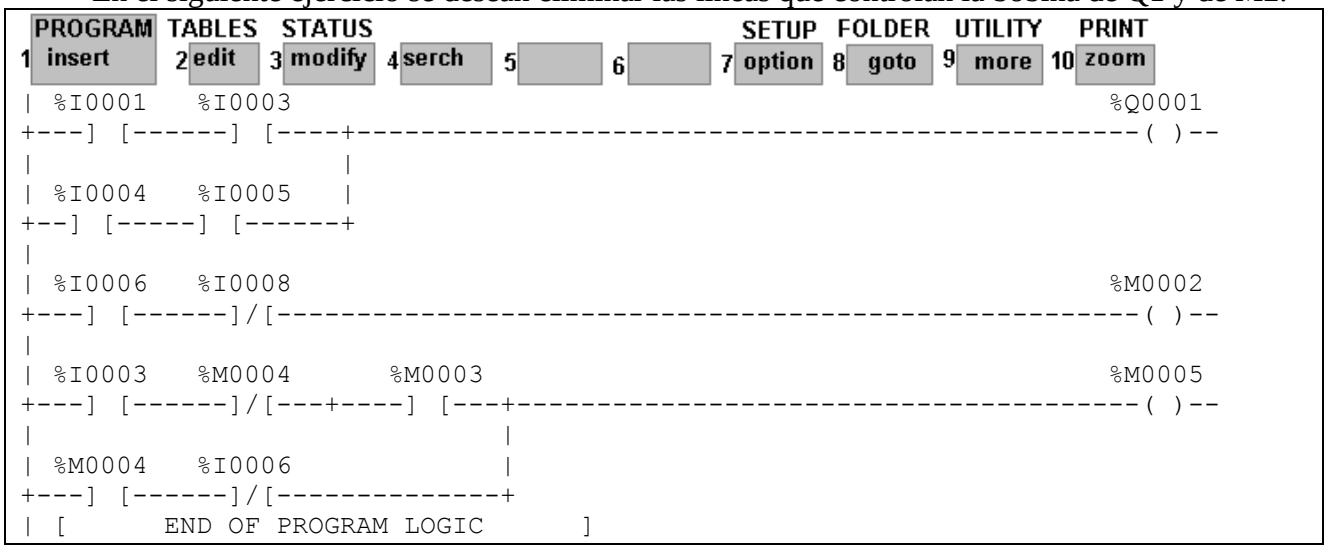
Entonces si realmente deseamos eliminar la línea oprimimos la tecla **Y** con lo cual se elimina la línea tal como se ve en la pantalla siguiente.



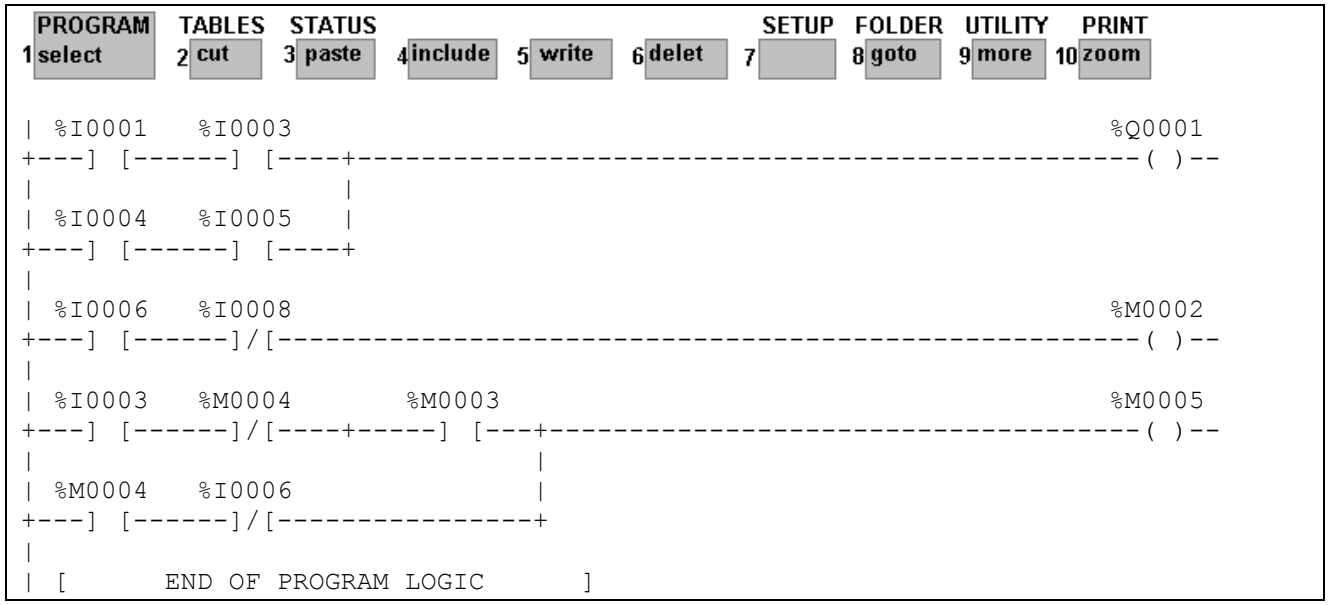
EJERCICIO N° 4.7.7 (Eliminar varias líneas).

Para eliminar varias líneas de un programa primeramente se seleccionan las líneas que se deseen eliminar y con el icono **DELETE** se borran ver el siguiente ejercicio.

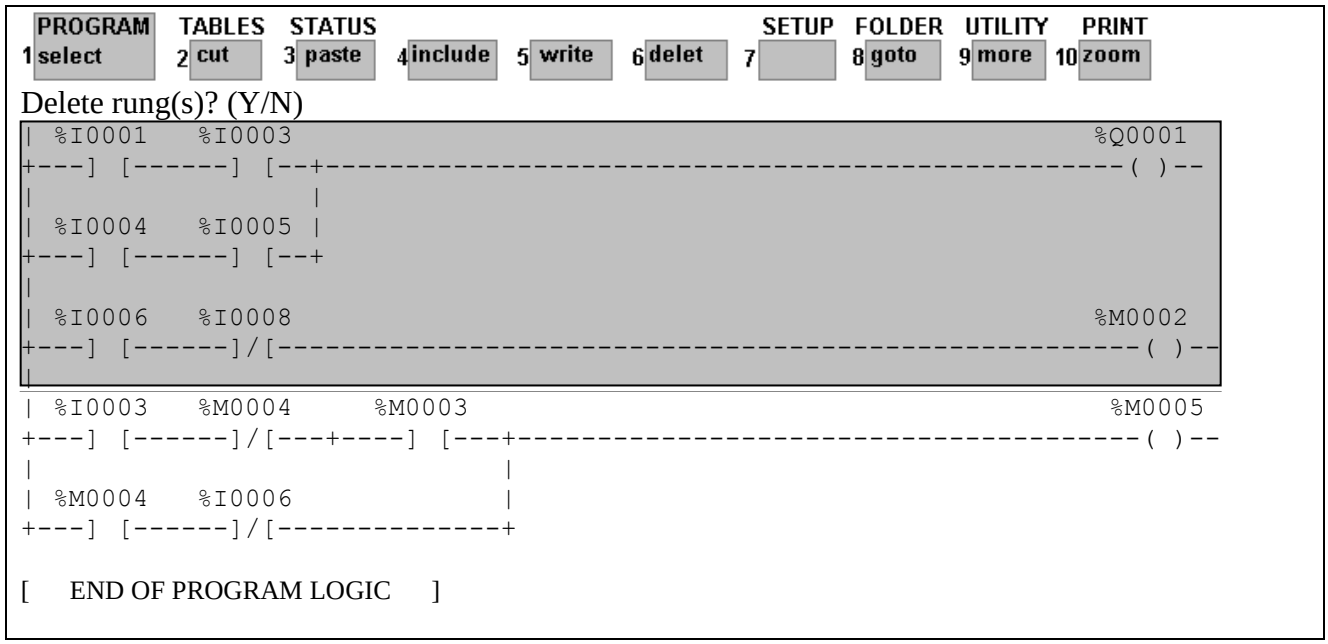
En el siguiente ejercicio se desean eliminar las líneas que controlan la bobina de Q1 y de M2.



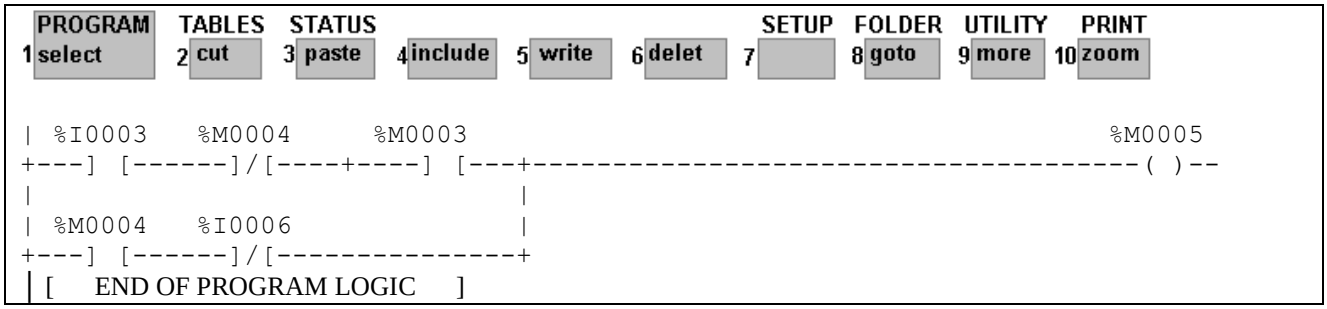
Primeramente observamos que en los iconos de la parte superior con **F9** aparece (**more**) la cual oprimimos y nos muestra los siguientes cambios.



Ahora con **F1 (select)** seleccionamos la dos línea donde se encuentra la bobina Q1 y con el cursor hacia abajo seleccionamos la siguiente linea donde se encuentra la bobina M2 y procedemos a oprimir la tecla **F6 (delete)** y nos pregunta si deseamos eliminar a lo cual se oprime la tecla Y.



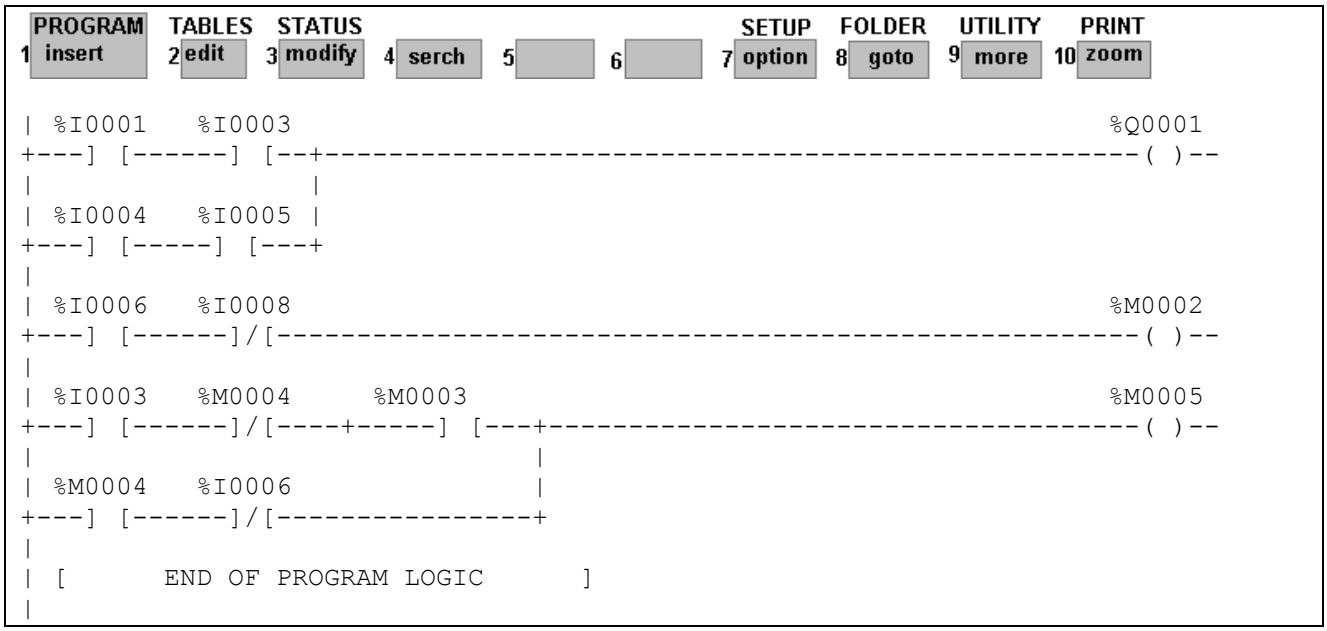
Con lo cual se borran las dos líneas quedando la figura siguiente.



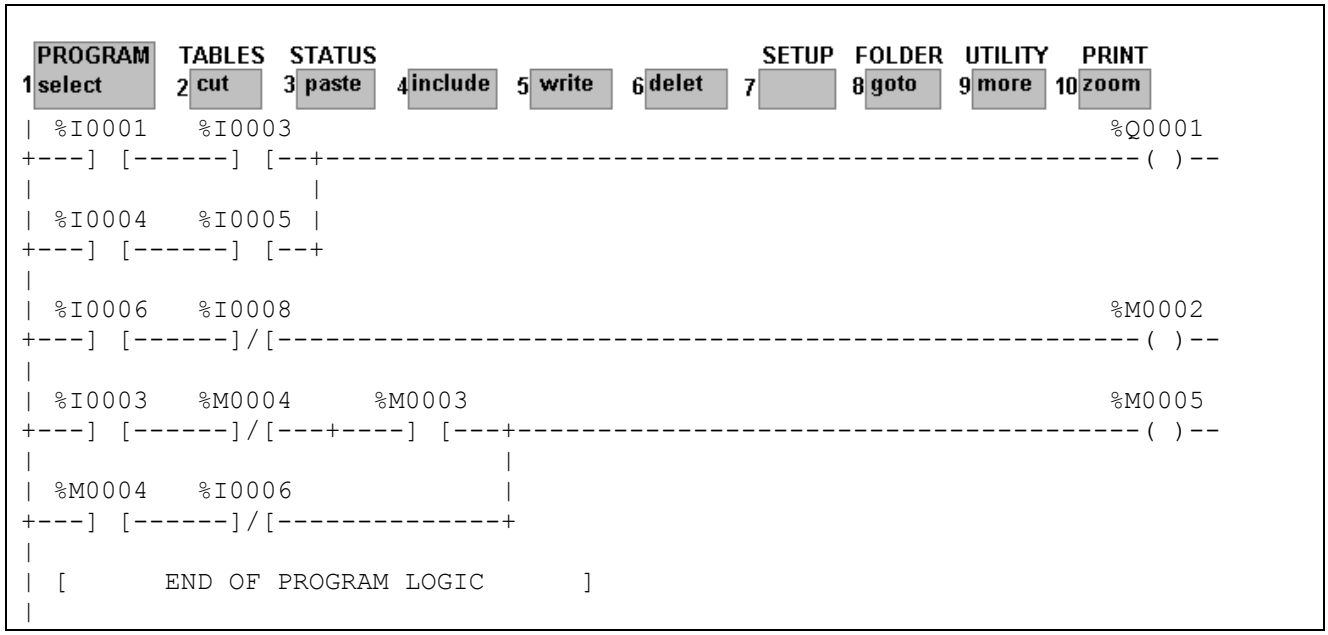
EJERCICIO N° 4.7.8 (Copiar una o varias líneas).

El proceso de cortar o pegar es muy similar, primero se seleccionan la línea o las líneas que se desean borrar o pegar luego se cortan y posteriormente se pegan tantas veces como se deseen.

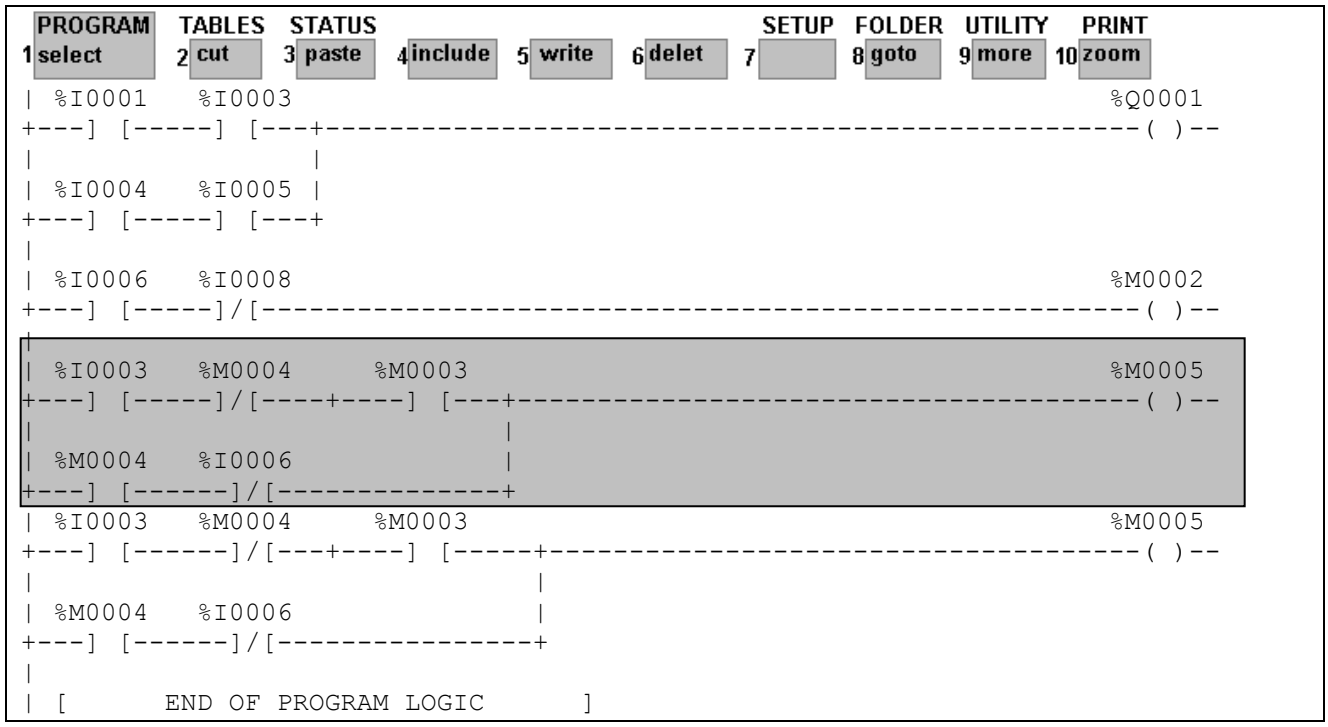
Observe el siguiente ejercicio en el cual se desea pegar una línea igual a la línea que controla M5.



Primero seleccionamos con F9 more con lo cual nos muestra los siguientes iconos



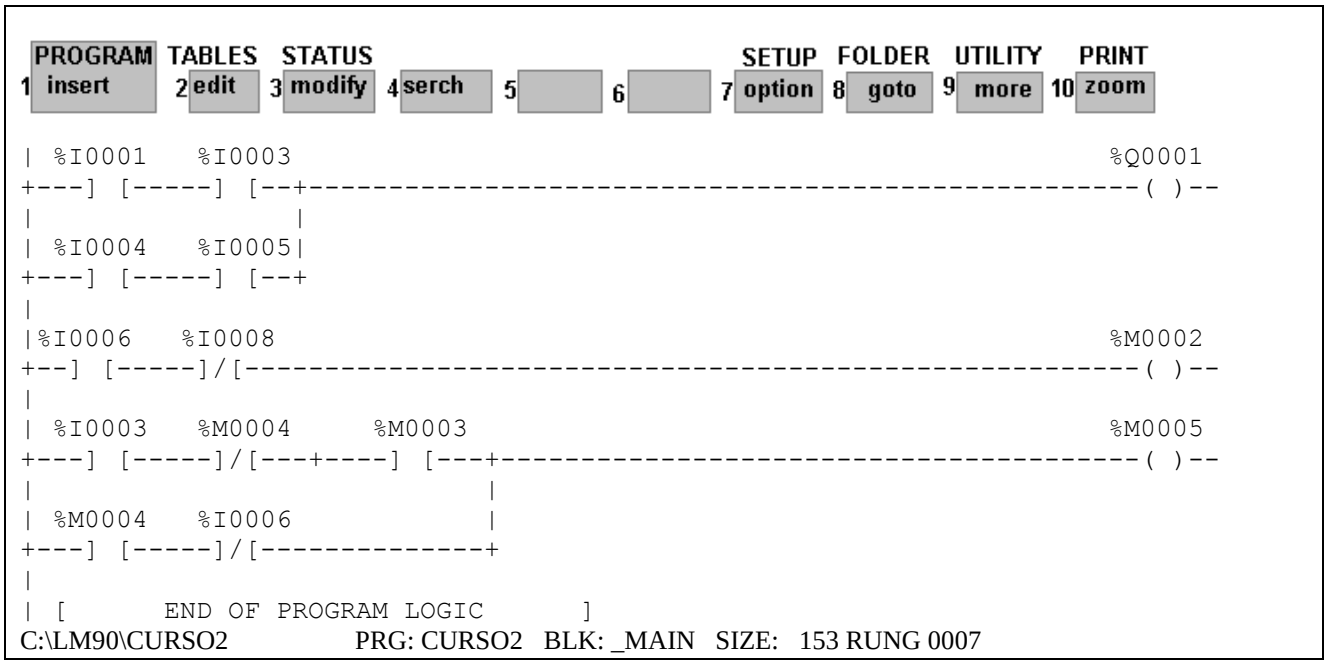
Ahora con F1 seleccionamos la línea que queremos copear, primero la cortamos con F2 (cut) y la pegamos con F3 (paste), luego la oprimimos de nuevo con lo cual tenemos la línea pegada.



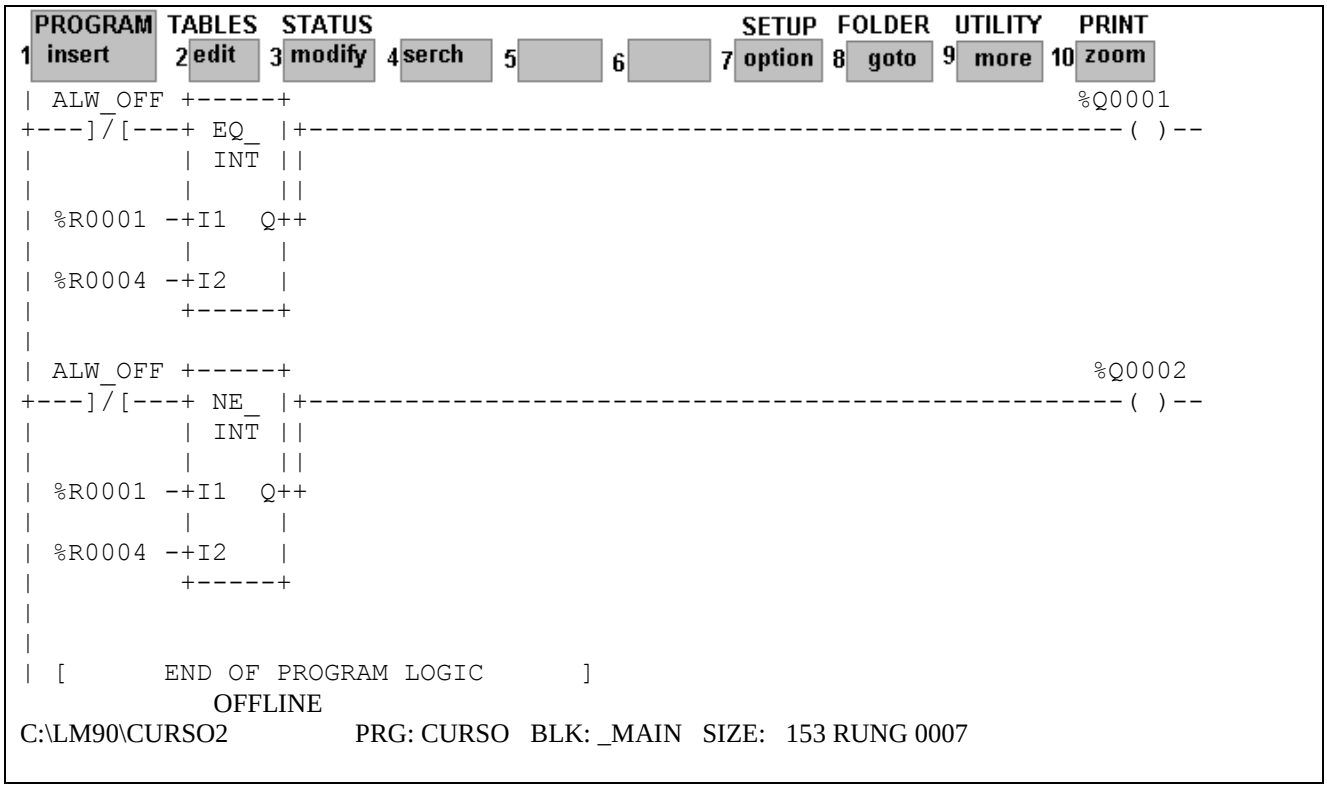
EJERCICIO N° 4.7.9 (Copiar una parte de un programa hacia otro programa).

Imaginemos que tenemos el siguiente fólder llamado CURSO 2 al cual queremos insertarle el programa que esta en el fólder de CURSO o parte del fólder llamado CURSO.

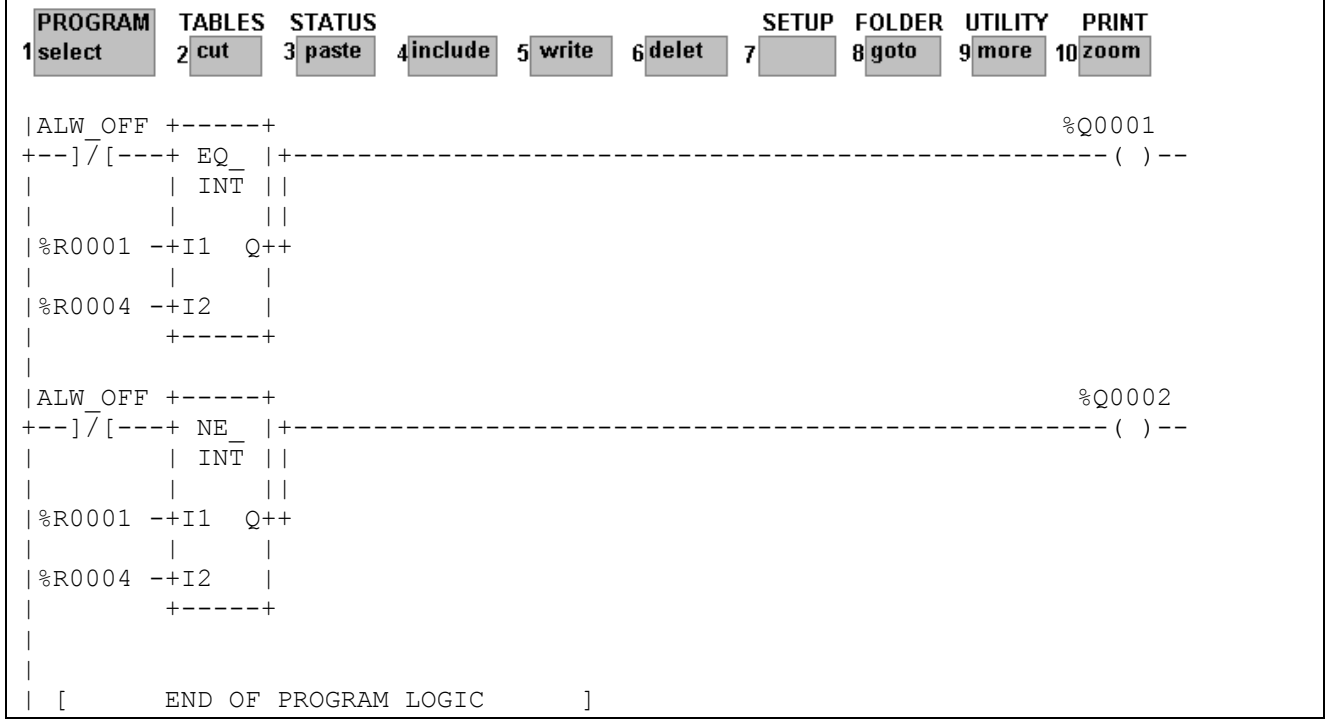
Observe la siguiente pantalla en la parte inferior del centro se encuentra el nombre del fólder.



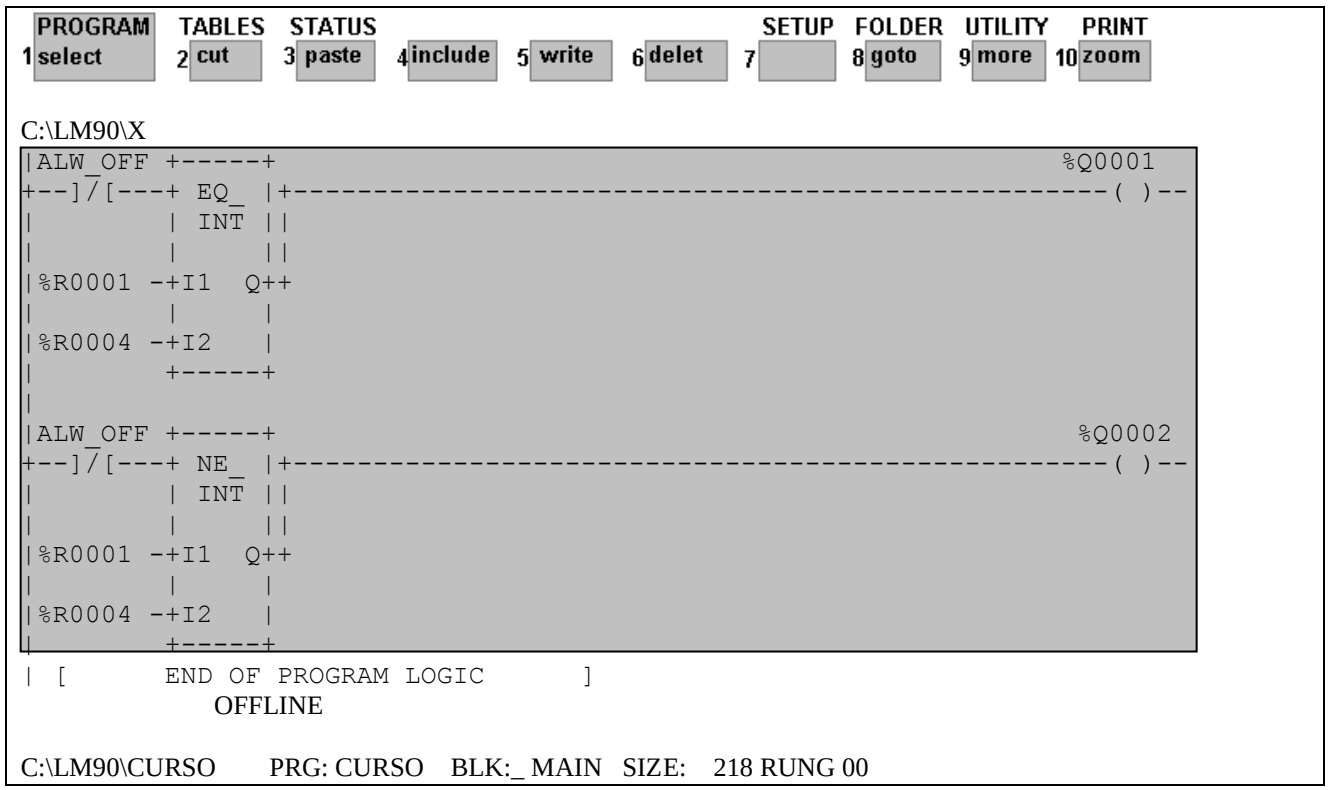
En la siguiente pantalla se muestra el programa del folder CURSO. Ver parte inferior.



Primeramente nos colocamos en el fólder de CURSO posteriormente oprimimos la tecla F9 (more) y nos aparece la siguiente pantalla.

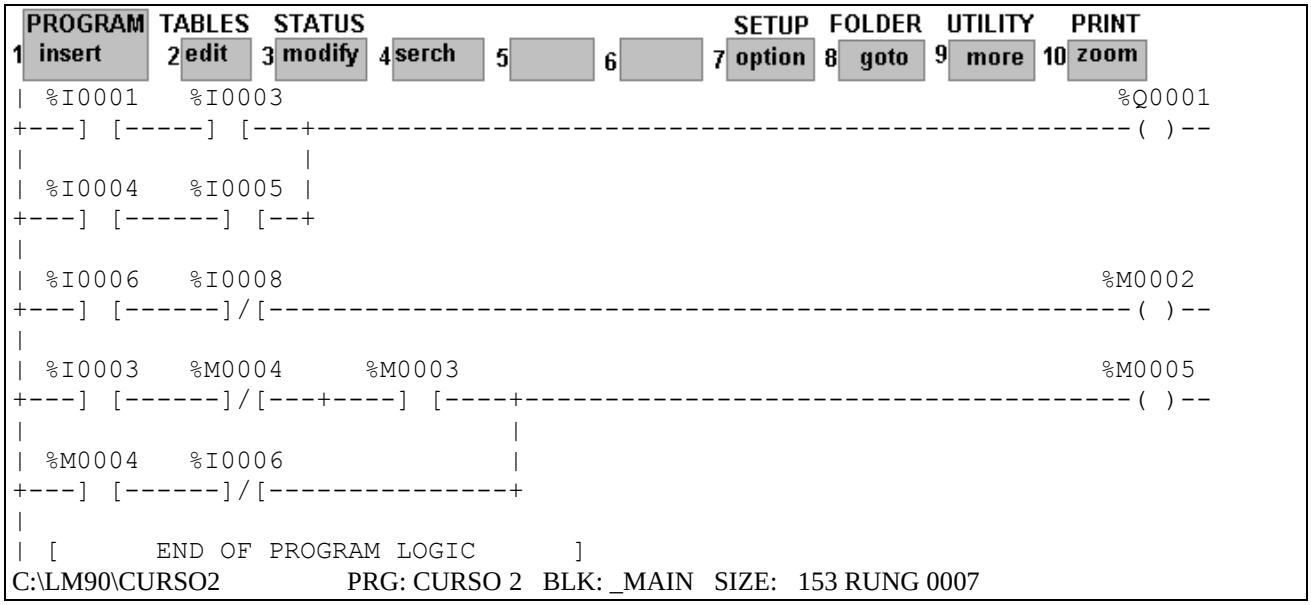


Posteriormente seleccionamos la parte del circuito que queremos trasladar de un fólder a otro, una ves seleccionado con la tecla **F1 (select)** le indicamos la ruta por donde vamos a trasladar dicha información en este caso es de la siguiente manera C:\LM90\X. y oprimimos la tecla **F5 (write).**

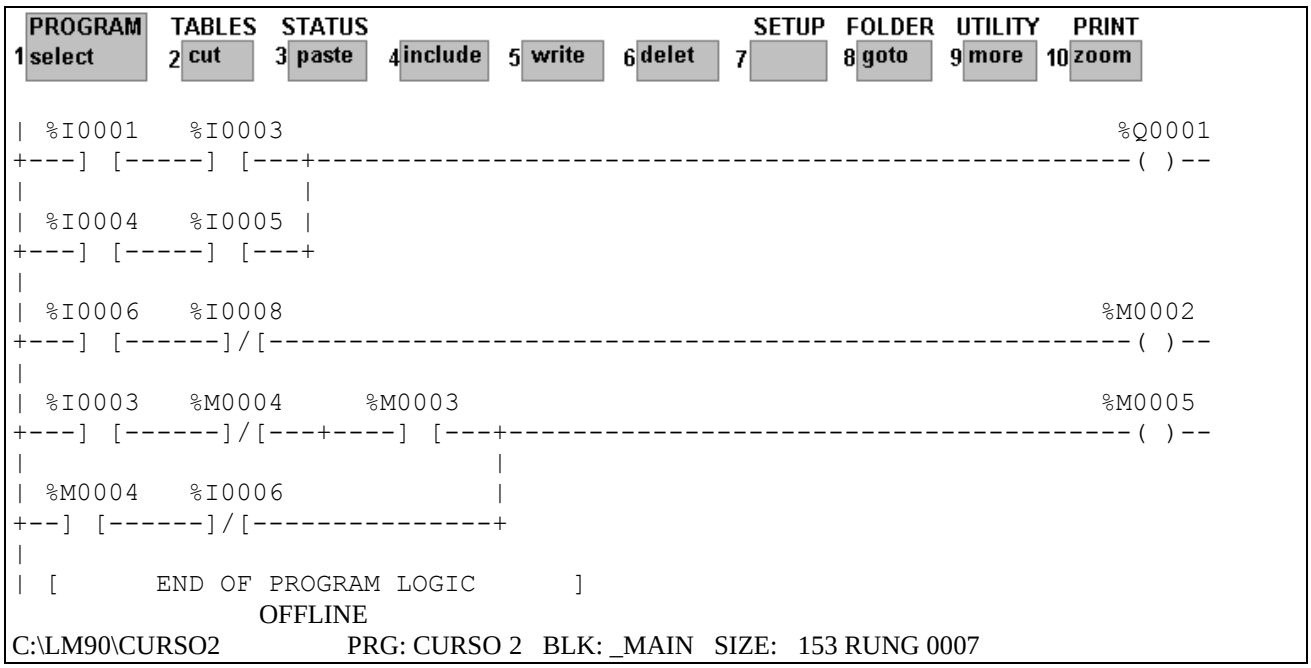


Posteriormente nos cambiamos de fólder y nos colocamos en el fólder de CURSO 2 y colocamos el cursor donde deseamos instalar la parte seleccionada del fólder anterior.

Primero, observe como nos instalamos en el fólder de CURSO 2

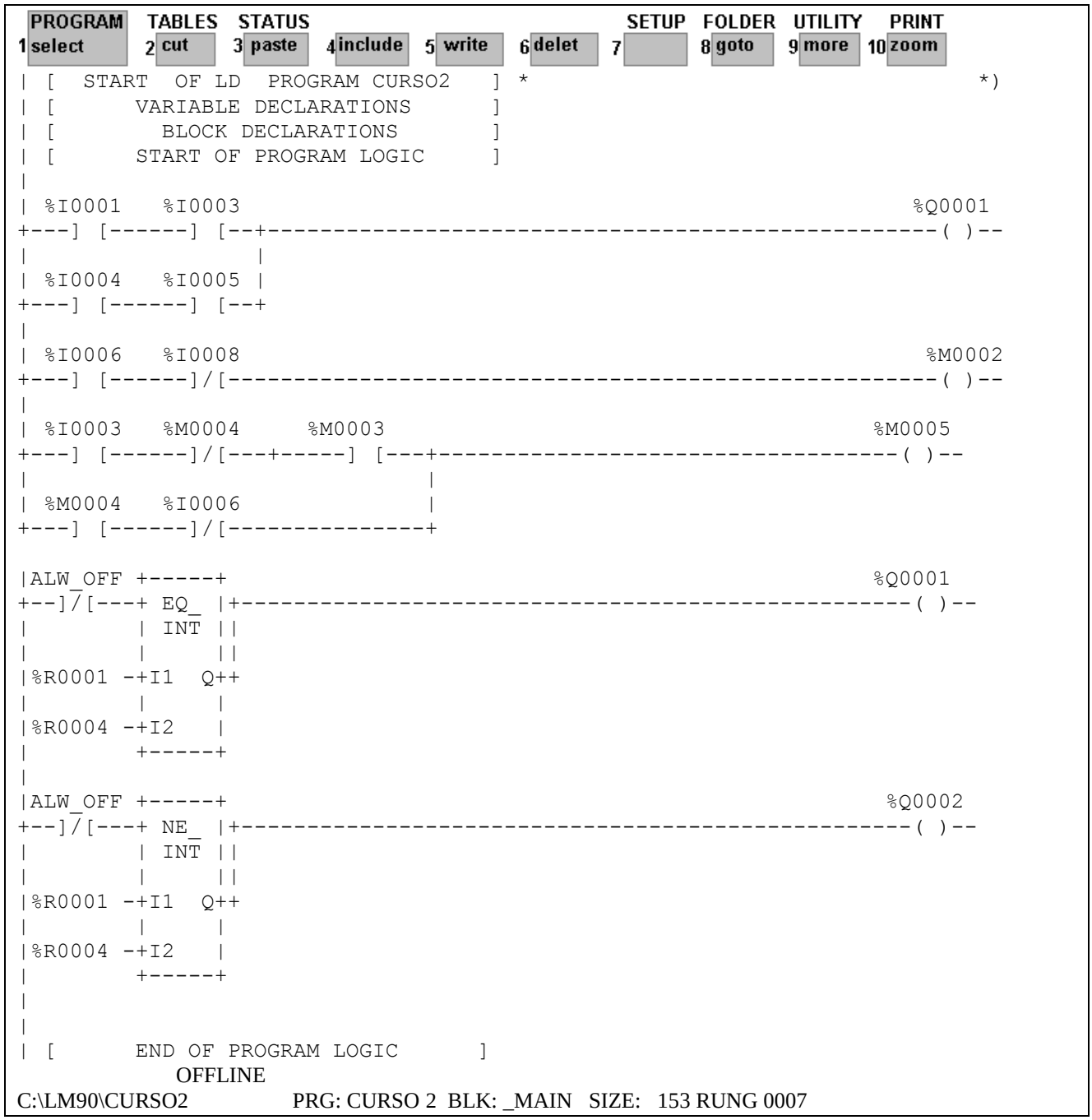


Finalmente colocamos el cursor en la parte donde deseamos instalar la parte seleccionada y con la tecla F9 (more) nos aparece



Ahora oprimimos la tecla C:\LM90\X y oprimimos la tecla F4 (include) con lo cual nos aparece la pantalla siguiente con la inclusión de la parte seleccionada.

NOTA: La ruta C:\LM90\X la x indica tan solo una dirección, para trasladar otra información se utiliza cualquier otra variable ejemplo b, c, d, f, etc.

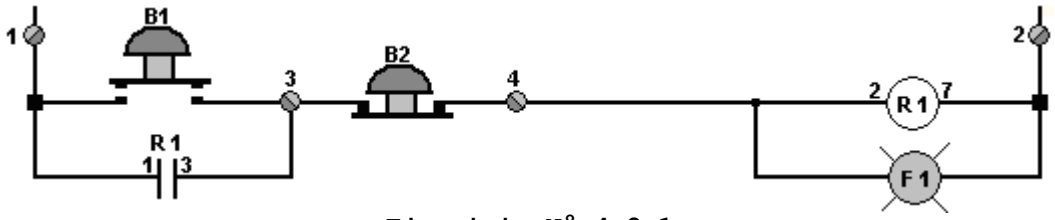


4.8 - Programación de circuitos donde intervienen las memorias.

EJERCICIO Nº 4.8.1

A continuación presentamos el circuito de la ecuación de la memoria en la cual al activar el botón 1 se energiza el relevador R1, este a su vez cierra su contacto con las patitas 1 y 3 el cual hace la función de contacto de sostén o contacto de memoria, de tal manera que si soltamos el botón 1 el relevador continuara energizado. Para desenergizar el relevador R1 se requiere que se active el botón 2 desenergizando el relevador R1 y al soltarlo permanece desenergizado.

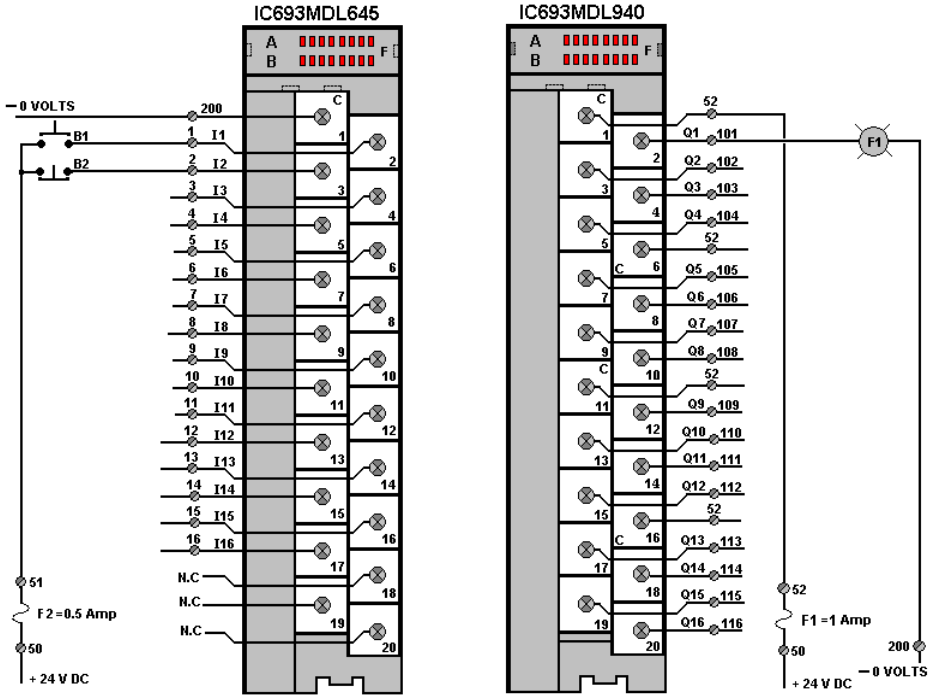
A continuación presentamos el circuito de control electromecánico y electrónico por medio de compuertas lógicas y sus diferentes formas de realizarlo con control programable.



Ejercicio N° 4.8.1

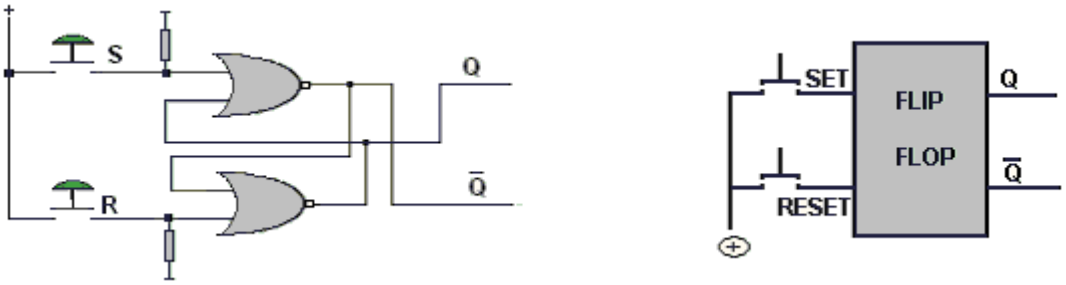
```
|[  START OF LD  PROGRAM CURSO   ]
|[     START OF PROGRAM LOGIC     ]
|  SET       RESET                                                FOCO_1
| %I0001    %I0002                                                %Q0001
+---] [---+---] [-------------------------------------------------( )--
| FOCO_1  |
| %Q0001  |
+---] [---+
|[      END OF PROGRAM LOGIC      ]
```

Aquí podemos observar que **I 2** se representó con un contacto abierto, esto es debido a que el botón de paro equivalente con **I 2** es un botón cerrado, lo que quiere decir que **I 2** ya se encuentra cerrado por efecto del cableado de control.



Cableado para el ejercicio N° 4.8.1

En electrónica en muy común encontrar la ecuación de la memoria en forma de flip flop en la cual se tiene dos entradas de Set y Reset y dos salidas de Q1 Y Q1 negada.



Ejercicio N° 4.8.1

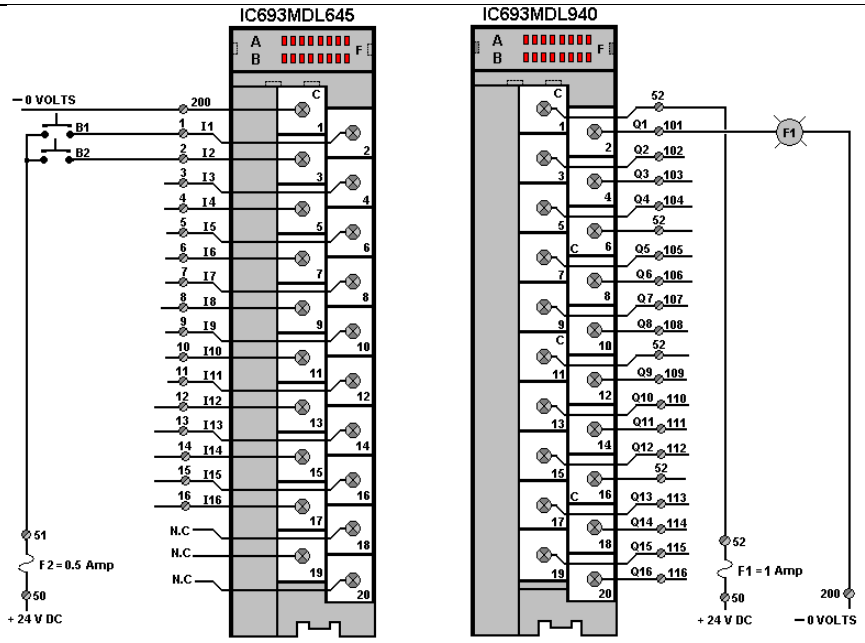
Aquí es importante hacer notar que la ecuación original de la memoria representada por:

$Q = (S + Q) \overline{R}$ observamos que el botón de paro (Reset) equivalente al control electromecánico es un botón cerrado, sin embargo en la representación electrónica se representa con el SET y el RESET son entradas abiertas, para lo cual el PLC tiene su equivalente tal como se presenta a continuación.

En este circuito al activar I1 se setea Q1 y al soltar el botón 1 representado por I1 permanece activado Q1, al darle un impulso a I2 se resetea Q1 y al soltarlo permanece desenergizado Q1.

A continuación, presentamos la forma de realizarlo con control programable.

```
|[  START OF LD  PROGRAM CURSO   ]
| << RUNG 4  STEP #0001 >>
|   SET                                                          FOCO_1
| %I0001                                                         %Q0001
+---] [-------------------------------------------------------------(S)-
| << RUNG 5  STEP #0003 >>
|  RESET                                                         FOCO_1
| %I0002                                                         %Q0001
+---] [-------------------------------------------------------------(R)-
|[      END OF PROGRAM LOGIC      ]
```

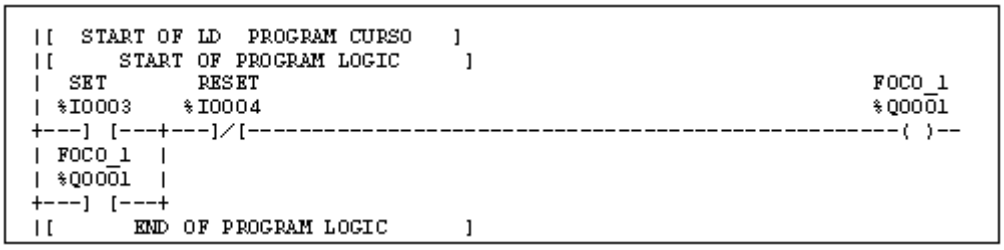


Cableado para el ejercicio N° 4.8.1

A continuación se presentan tres casos los cuales hacemos la aclaración que pudieran ser muy peligrosos y definitivamente prohibitivos, en caso de realizarlos se deberán tomar las medidas correspondientes.

1º Caso: Lo consideraremos como el anteriormente analizado.

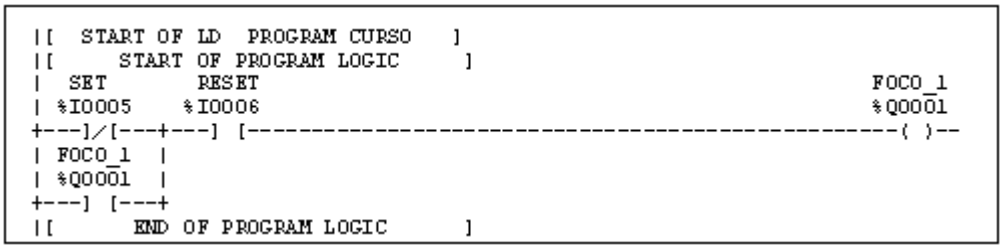
2º Caso: Ahora imaginemos que en el ejercicio 4.8.1 en lugar de un foco es un arrancador para un motor, y se daña el botón de paro I2 el cual es normalmente cerrado y en el almacén no se tienen botones normalmente cerrados pero si se cuenta con botones normalmente abiertos y se desea adaptarlo, realice el circuito correspondiente.



Ahora explicaremos la razón del porque este circuito puede ser peligroso: en el caso de que existiera un falso contacto en el botón de paro, el arrancador del motor continua energizado y no tenemos posibilidad de pararlo, lo cual puede ser muy peligroso porque no sabemos en que momento puede resultar el falso contacto. **NOTA**: Observe que se realizó un nuevo cableado con entradas diferentes para realizar la práctica, en el cual al botón de arranque se conecto a la entrada I3 y el botón de paro se conecto a la entrada I4.

3º caso.

Ahora imaginemos que en el ejercicio 4.8.1 que el botón que se daño es el botón de arranque, esto es el botón normalmente abierto y en el almacén se encuentran botones normalmente cerrados; realice la adaptación correspondiente.



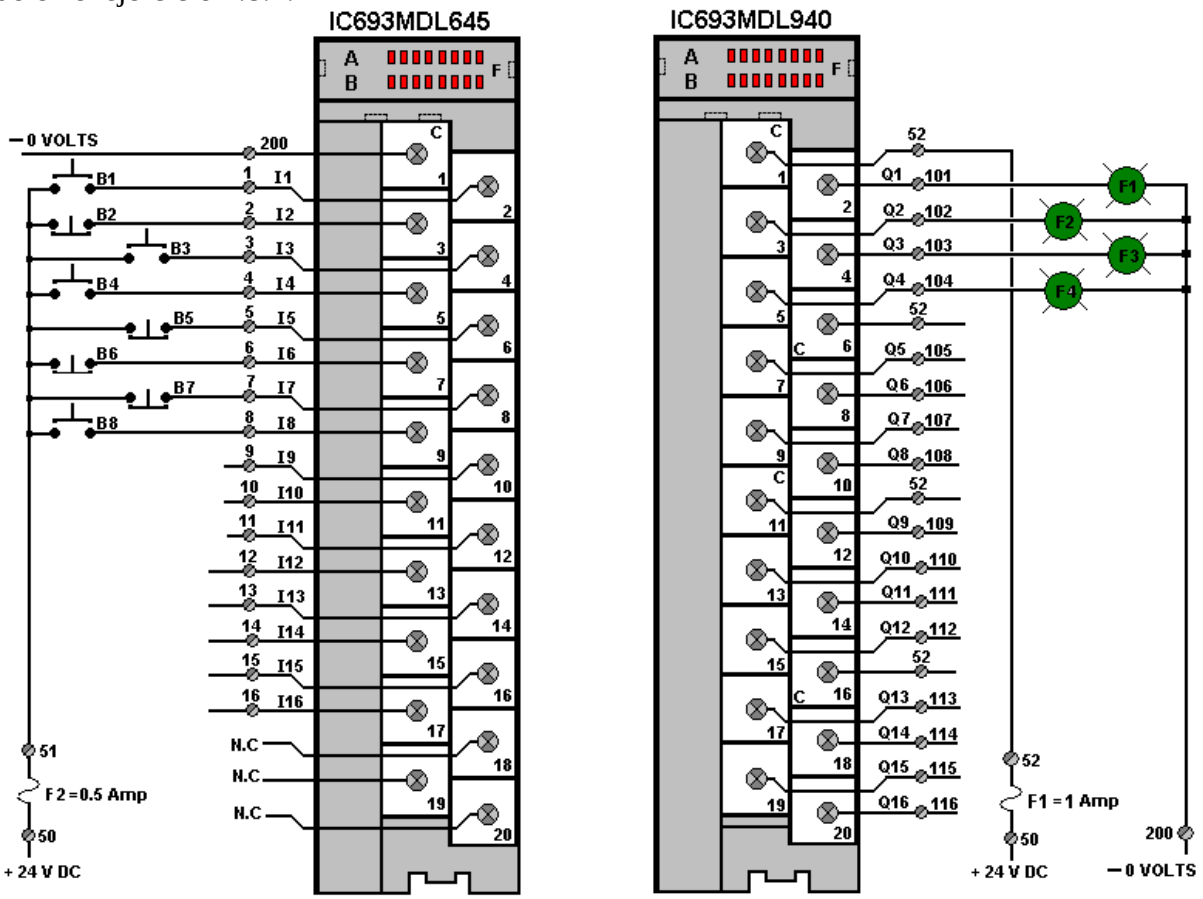
En este caso, en caso de existir un falso contacto en el botón de arranque una vez que ya halla sido arrancado el motor, no lo podemos parar ya que si oprimimos el botón de paro este, se parará en forma momentánea pero al desactivar el botón de paro arrancaría de nuevo, en caso de accidente el operador ante la eventualidad del accidente tiende a parar el motor con el botón de paro pero solo con pulsos de paro y se puede dar la situación de que no se quede oprimiendo el botón, lo cual resultaría muy peligroso. **NOTA**: Observe que se realizó un nuevo cableado con entradas diferentes para realizar la práctica, en el cual al botón de arranque se conecto a la entrada I5 y el botón de paro se conecto a la entrada I6.

4º caso. Ahora imaginemos el segundo caso en el cual se daño el botón de paro y se sustituyo por un botón normalmente abierto y como en condiciones normales el sistema estuvo trabajando bien y no se hicieron las correcciones debidas esto es sustituir el botón adecuado, posteriormente se daño el botón de arranque el cual es normalmente abierto y en el almacén solo existen botones normalmente cerrados, y se desea que la maquina continúe trabajando, realice las adecuaciones al circuito. Finalmente observe que se desea arrancar un motor con un botón de arranque normalmente cerrado y pararlo con un botón normalmente abierto. **NOTA**: Observe que se realizó un nuevo cableado con entradas diferentes para realizar la práctica, en el cual al botón de arranque se conecto a la entrada I7 y el botón de paro se conecto a la entrada I8.

A continuación presentamos el circuito correspondiente.

```
|[  START OF LD  PROGRAM CURSO   ]
|[     START OF PROGRAM LOGIC     ]
|  SET        RESET                                              FOCO_1
| %I0007     %I0008                                              %Q0001
+---]/[---+---]/[-------------------------------------------------( )--
| FOCO_1  |
| %Q0001  |
+---] [---+
|[      END OF PROGRAM LOGIC      ]
```

Finalmente presentamos las respectivas conexiones al PLC para realizar estos cuatro casos, pero haciendo la aclaración que el único circuito que cumple la norma es el circuito original mostrado en el ejercicio 4.8.1.



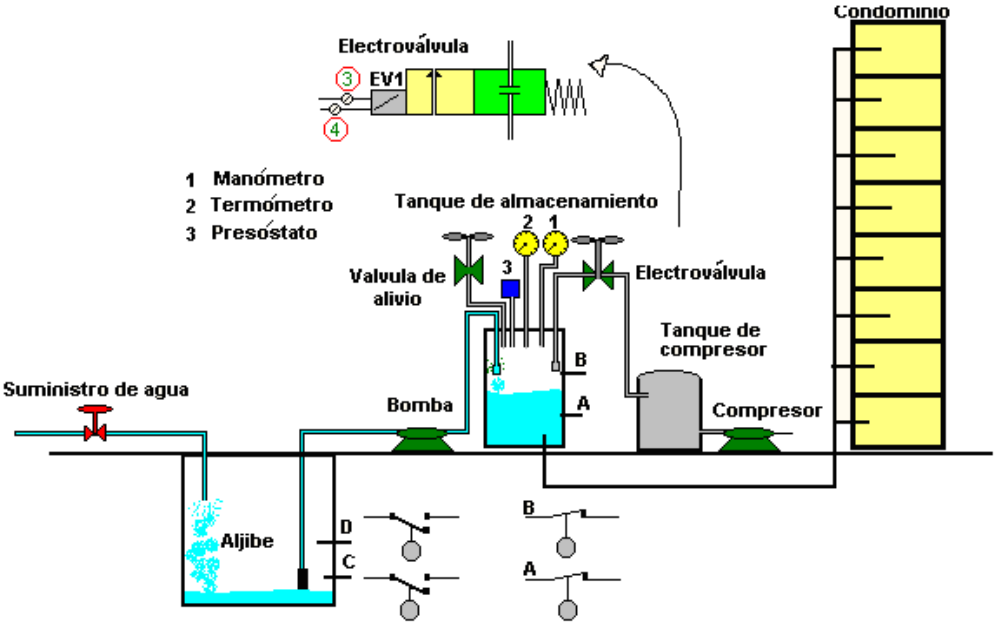
Ejercicio 4.8.1

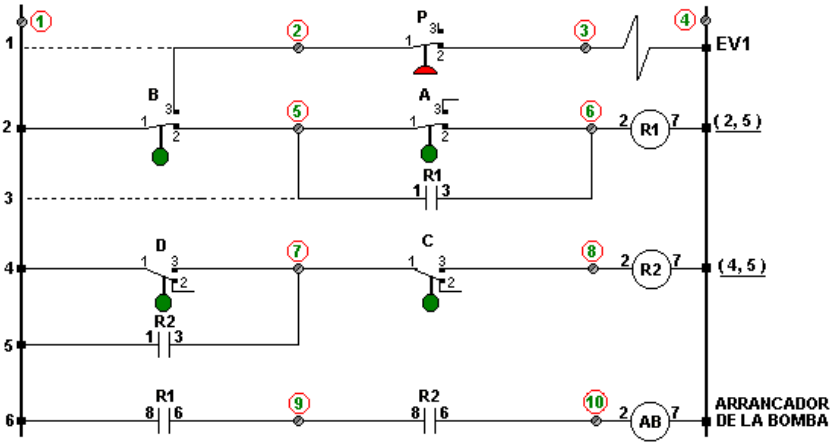
EJERCICIO Nº 4.8.2

A continuación presentamos un circuito ya expuesto en el capítulo dos, con solución electromecánica, ahora vamos a realizar una solución con control programable. (Ver dibujo de ejercicio 4.8.2). Queremos suministrar agua a un edificio de siete pisos, en el cual se tiene un depósito cerrado sometido a presión, el cual por norma deberá tener un manómetro para estar viendo la presión, un termómetro para estar viendo la temperatura, una válvula de seguridad para que en caso de falla de los sistemas eléctricos pueda abrir y desalojar la presión y un presóstato para estar controlando la presión, a la que se requiere que suministre agua a presión a todos los pisos del edificio, bajo las siguientes condiciones;

Tomando en cuenta los controles anteriores ahora se desea que cuando el nivel del tanque de presión se encuentre en el punto B se cheque la presión en el tanque y si ésta presión se encuentra por debajo de los 7 Kg/cm2 que se energice una electro válvula para inyectarle aire a presión al depósito y que cuando la presión sea igual o mayor a los 7 Kg/cm2 se des-energice dicha electroválvula. Con un estudio previo se comprobó que con una presión de 7 Kg/cm2 se inyectará agua a todos los pisos del condominio con la misma presión.

NOTA: Observe que la electro válvula es una de 2 vías dos posiciones normalmente cerrada.





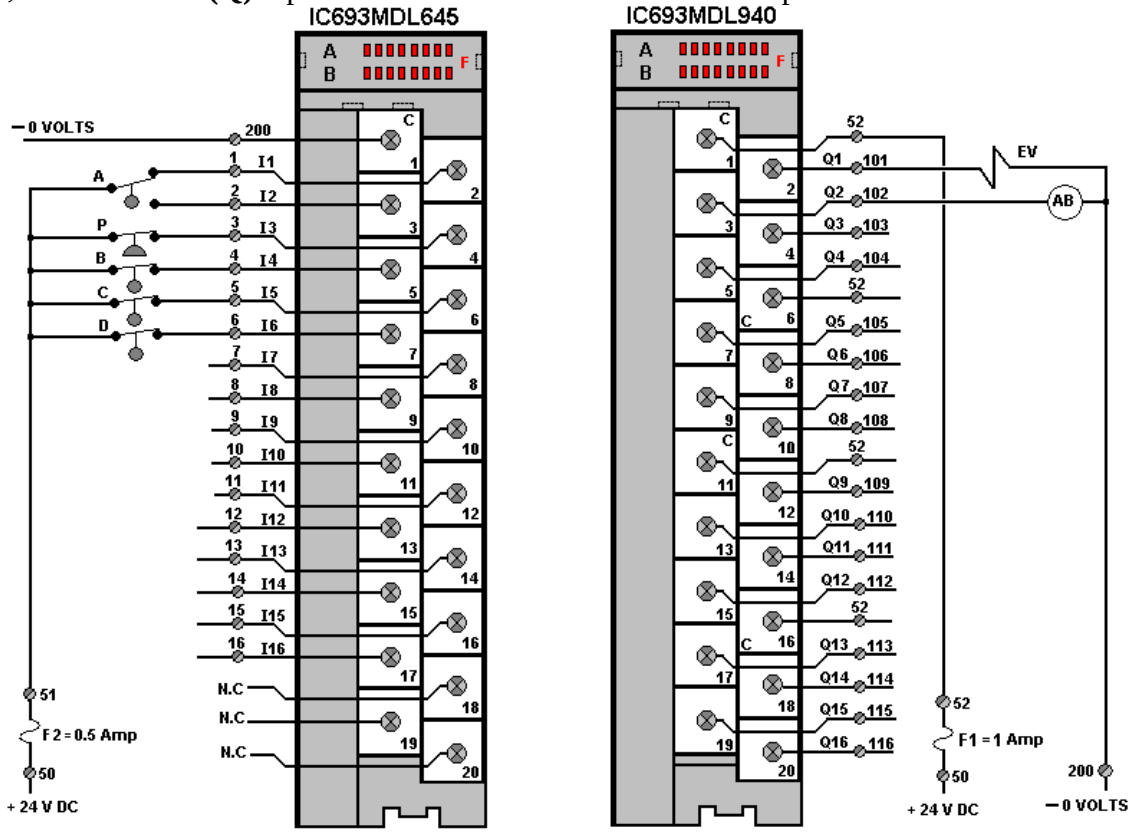
Ejercicio N° 4.8.2

Solución con control electromecánico para el problema anterior

A continuación presentamos su equivalente con control programable.

```
|[  START OF LD  PROGRAM CURSO6   ]
|[     VARIABLE DECLARATIONS      ]
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|%I0002  %I0004                                                %M0001
+--] [--+--] [--+-----------------------------------------------( )--
|       |       |
|       |%M0001 |
|       +--] [--+
| << RUNG 5  STEP #0006 >>
|%I0001  %I0003                                                %Q0001
+--] [-----] [-----------------------------------------------( )--
|
| << RUNG 6  STEP #0009 >>
|%I0005  %I0006                                                %M0002
+--] [--+--] [-----------------------------------------------( )--
|       |
|%M0002 |
+--] [--+
|
| << RUNG 7  STEP #0013 >>
|%M0001  %M0002                                                %Q0002
+--] [-----] [-----------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```

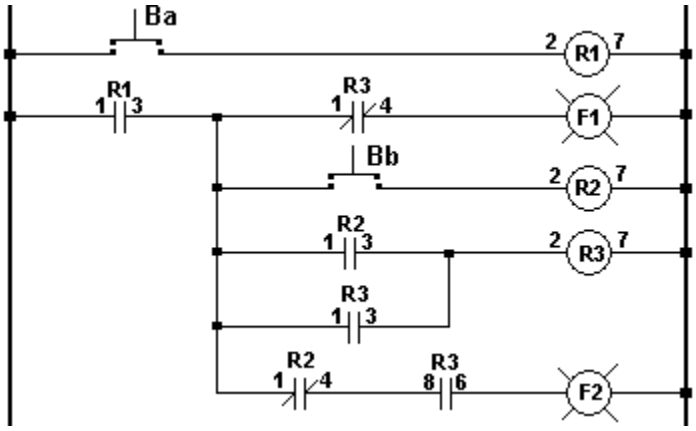
Observe como aquí las **(M)** representan relevadores internos, los cuales se puede decir que no cuestan, en cambio las **(Q)** representan salidas del PLC las cuales representan valor real del PLC.



Cableado para el ejercicio N° 4.8.2

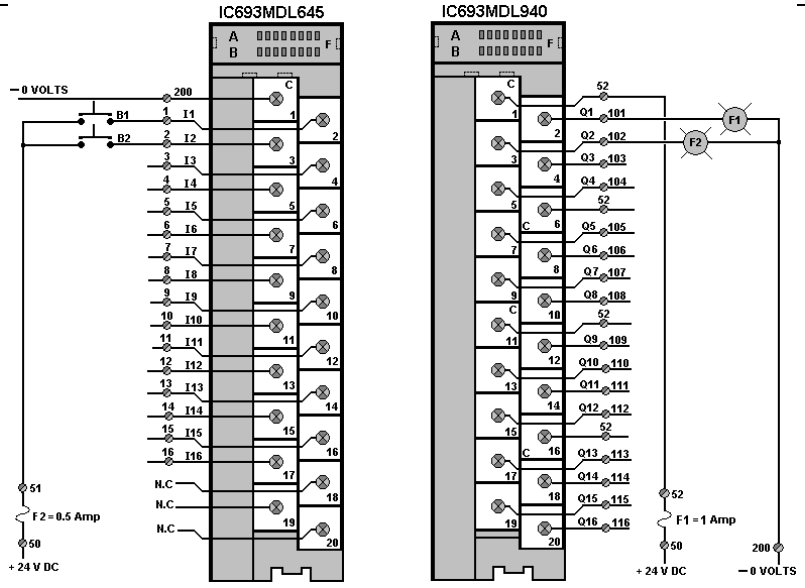
EJERCICIO Nº 4.8.3

Ejemplo clásico de la secuencia se tienen dos botones A y B y se desea que al oprimir A se encienda el foco 1; luego al oprimir B y mantener oprimida a A se apague el foco 1; finalmente al volver a abrir el botón B se encienda el foco 2 y al soltar el botón A se apague.



EJEMPLO CLASICO DE LA SECUENCIA

```
|[  START OF LD  PROGRAM CURSO 14  |
  BOTON_A                                                    FOCO_1
|%I0001  %M0001                                            %Q0001
+--] [--+--]/[-----------------------------------------------( )--
|       |
|       |BOTON_B
|       |%I0002                                            %M0001
|       +--] [--+-------------------------------------------( )--
|       |       |
|       |%M0001 |
|       +--] [--+
|       |
|       |BOTON_B                                          FOCO_2
|       |%I0002   %M0001                                  %Q0002
|       +--]/[-----] [-------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```



Cableado para el ejercicio N° 4.8.3

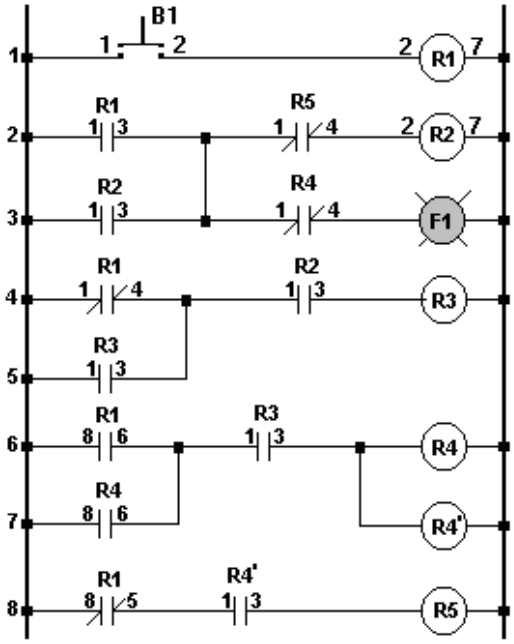
EJEMPLO 4.8.4

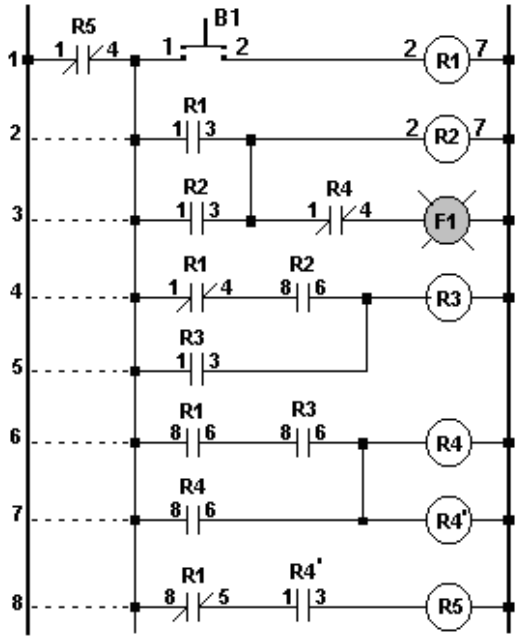
Siguiendo la misma metodología realice el siguiente circuito.

Al oprimir el botón 1, que se energice el Foco 1 y que al soltarlo que se quede prendido.

Luego al oprimir de nuevo el botón 1, que se apague y al soltarlo permanezca apagado y que todo quede en condiciones para repetir el ciclo.

Como podemos observar, este es un Flip Flop tipo "T"

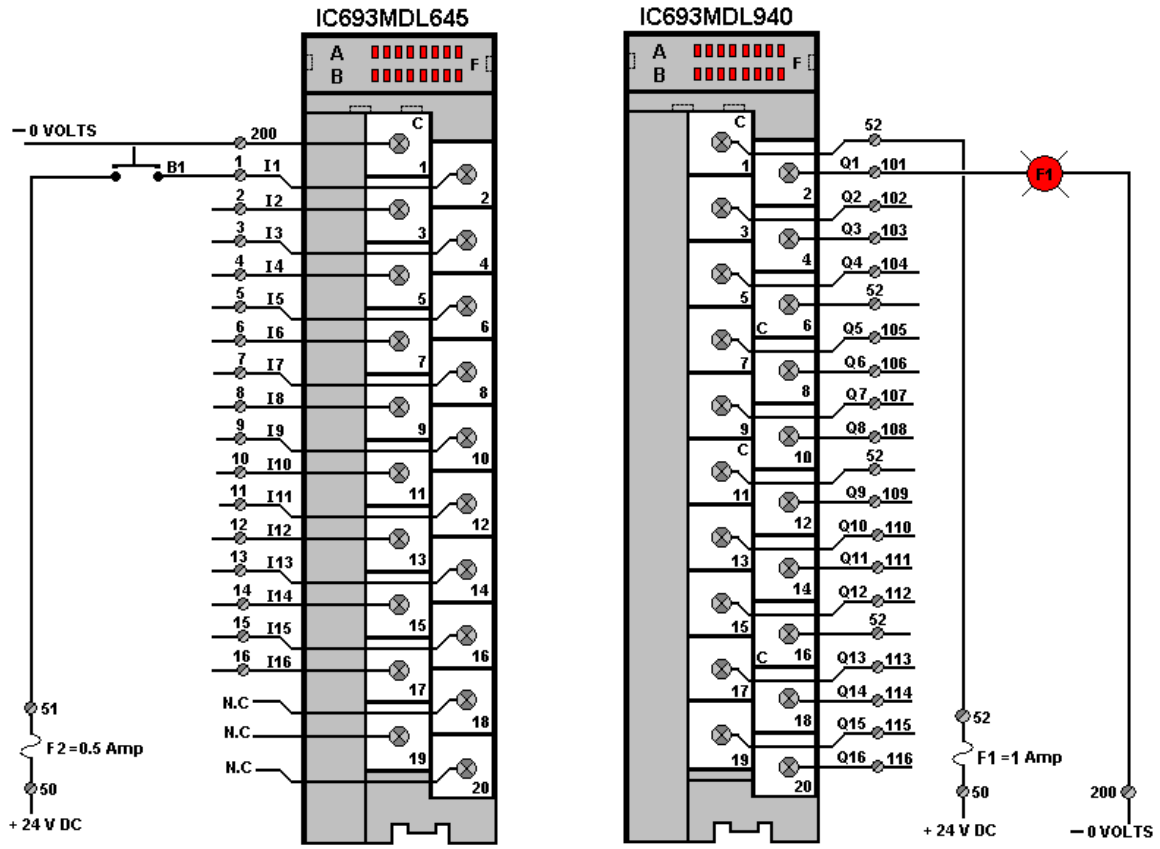





```
|[     START OF PROGRAM LOGIC     ]
|BOTON_1
|%I0001                                                          %M0001
+--] [-----------------------------------------------------------------( )--
|%M0004   %M0001    %M0003                                        %Q0001
+--]/[--+---] [---+---]/[-----------------------------------------( )-
|       | FOCO_1  |
|       | %Q0001  |
|       +---] [---+
|       |        FOCO_1
|       |%M0001  %Q0001                                          %M0002
|       +--]/[-----] [--+-----------------------------------------( )-
|       |               |
|       |%M0002         |
|       +--] [----------+
|       |
|       |%M0001  %M0002                                          %M0003
|       +--] [-----] [--+-----------------------------------------( )-
|       |               |
|       |%M0003         |
|       +--] [----------+
|       |
|       |%M0001  %M0003                                          %M0004
|       +--]/[-----] [-----------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```

A continuación presentamos una segunda solución para el problema anterior

```
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|BOTON_1
|%I0001                                                          %M0001
+--] [-----------------------------------------------------------( )--
|                                                                FOCO_1
|%M0001  %M0003                                                  %Q0001
+--] [--+--]/[-----------------------------------------------------( )--
|%Q0001 |
+--] [--+
|         FOCO_1
|%M0001  %Q0001  %M0004                                          %M0002
+--]/[-----] [--+--]/[-------------------------------------------( )--
|%M0002         |
+--] [----------+
|
|%M0001  %M0002                                                  %M0003
+--] [--+--] [-----------------------------------------------------( )--
|%M0003 |
+--] [--+
|
|%M0001  %M0003                                                  %M0004
+--]/[-----] [-----------------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```



Cableado para el ejercicio N° 4.8.4

EJEMPLO 4.8.5:

Diseñe el circuito de control que realice las siguientes funciones:

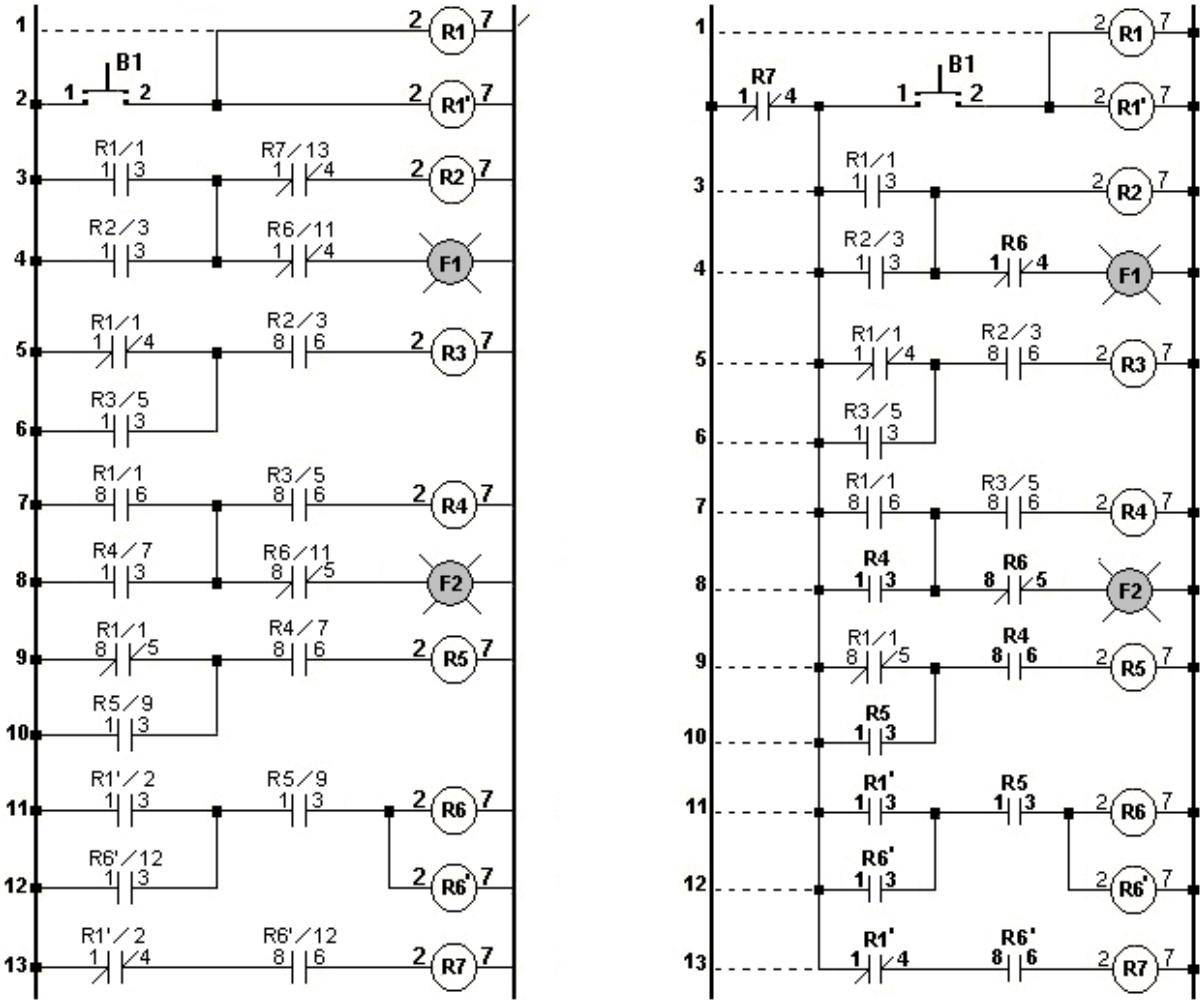
1.- Que al oprimir el botón 1, prenda el Foco 1 y al soltarlo permanezca prendido.

2.- Al oprimir de nuevo el botón 1, que se energice el foco 2 y que el Foco 1 continué energizado.

3.- Al oprimir de nuevo el botón 1, se apaguen los dos focos y al soltarlo quede el sistema listo para reiniciar la operación.

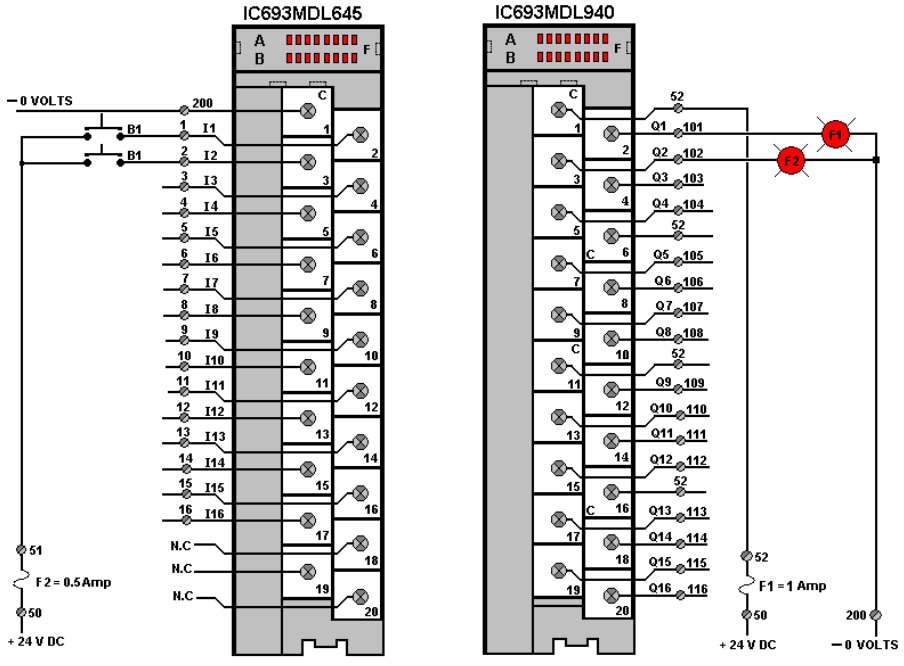
SOLUCIÓN:

Aquí presentamos dos alternativas de solución, primero con control electro mecánico y posteriormente con control programable.



Solución con control electro mecánico

```
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|BOTON_1
|%I0001                                                          %M0001
+--] [-----------------------------------------------------------( )-
|                                                                FOCO_1
|%M0001  %M0004                                                  %Q0001
+--] [--+--]/[-----------------------------------------------------( )-
|%Q0001 |
+--] [--+
|
|%M0001  %Q0001                                                  %M0002
+--]/[--+--] [-----------------------------------------------------( )-
|%M0002 |
+--] [--+
                                                                 FOCO_2
|%M0001  %M0002                                                  %Q0002
+--] [--+--] [-----------------------------------------------------( )-
|%Q0002 |
+--] [--+
|
|%M0001  %Q0002                                                  %M0003
+--]/[--+--] [-----------------------------------------------------( )-
|%M0003 |
+--] [--+
|
|%M0001  %M0003  %M0005                                          %M0004
+--] [-----] [--+--]/[-------------------------------------------( )-
|%M0004         |
+--] [----------+
|
|%M0001  %M0004                                                  %M0005
+--]/[-----] [-----------------------------------------------------( )--
|[      END OF PROGRAM LOGIC      ]
```



Cableado para el ejercicio N°4.8.5

EJEMPLO 4.8.6

Diseñe el circuito de control que realice las siguientes funciones:

1.- Que al oprimir el botón 1, prenda el Foco 1 y al soltarlo permanezca prendido.
2.- Al oprimir de nuevo el botón 1 que se energice el foco 2 y que el Foco 1 continué energizado.
3.- Al oprimir de nuevo el botón 1 se energice el foco 3 y los dos focos anteriores permanezcan encendidos.
4.- Al oprimir de nuevo el botón 1 se apaguen los tres focos y al soltarlo permanezcan apagados y que el sistema quede el sistema listo para reiniciar la operación.

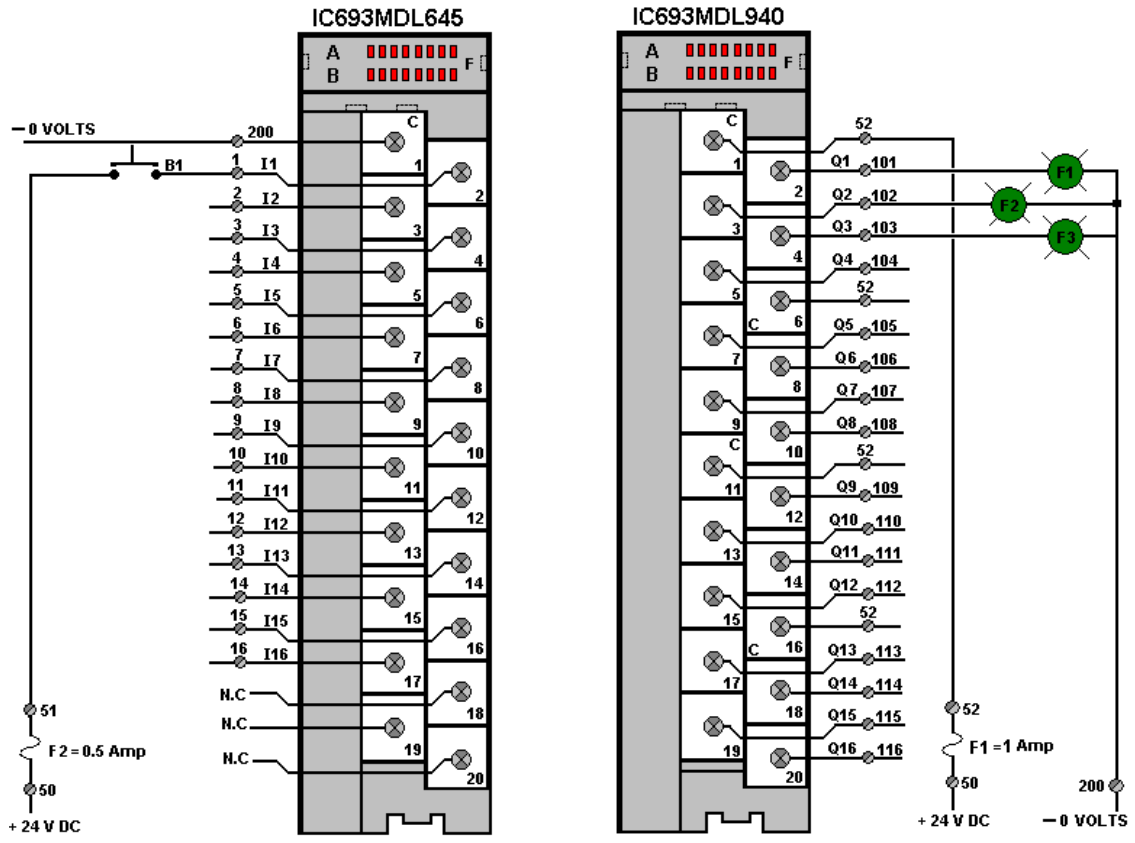
Como puede observar éste es un contador.

```
[  START OF LD  PROGRAM CURSO19  ]
|[     VARIABLE DECLARATIONS      ]

             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001                     BOTON_1
          %Q0001                     FOCO_1
          %Q0002                     FOCO_2
          %Q0003                     FOCO_3

|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
 BOTON_1
|%I0001                                                          %M0001
+--] [-------------------------------------------------------------( )--
|
|                                                                 FOCO_1
|%M0001  %M0005                                                   %Q0001
+--] [--+--]/[---------------------------------------------------( )--
|       |
|%Q0001 |
+--] [--+
|
|%M0001  %Q0001                                                   %M0002
+--]/[--+--] [---------------------------------------------------( )--
|       |
|%M0002 |
+--] [--+
|                                                                 FOCO_2
|%M0001  %M0002                                                   %Q0002
+--] [--+--] [---------------------------------------------------( )--
|       |
|%Q0002 |
+--] [--+
|%M0001  %Q0002                                                   %M0003
+--]/[--+--] [---------------------------------------------------( )--
|       |
|%M0003 |
+--] [--+
```

```
|                                                                  FOCO_3
|%M0001   %M0003                                                  %Q0003
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%Q0003 |
+--] [--+
|
|%M0001   %Q0003                                                  %M0004
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0004 |
+--] [--+
|
|%M0001   %M0004   %M0006                                         %M0005
+--] [-----] [--+--]/[-------------------------------------------( )--
|               |
|%M0005         |
+--] [----------+
|
|%M0001   %M0005                                                  %M0006
+--]/[-----] [-----------------------------------------------------( )--
|
|[       END OF PROGRAM LOGIC       ]
```

SOLUCIÓN:

A continuación presentamos la solución de éste problema sólo que de aquí en adelante se presentarán sólo las soluciones con control programable así como su cableado.



Cableado para el ejercicio N° 4.8.6

EJEMPLO 4.8.7

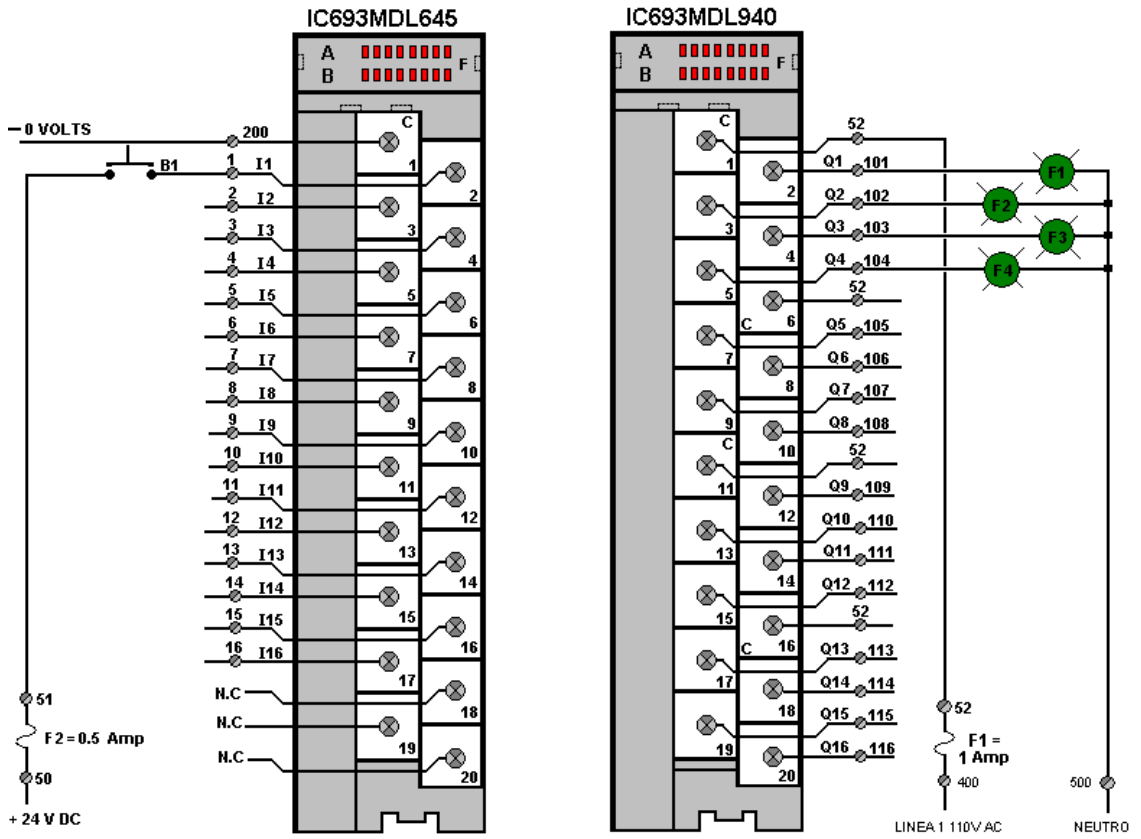
Diseñe el circuito de control de un contador de Binario de cuatro bits, considere que ahora los focos sean de 110VAC.

SOLUCIÓN:

```
|[  START OF LD  PROGRAM PRAC_3   ]       |
|[     VARIABLE DECLARATIONS      ]
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|%I0001                                                              %M0001
+--] [-------------------------------------------------------------( )--
| << RUNG 5  STEP #0003 >>
|
|%M0001  %M0017                                                     %M0002
+--] [--+--]/[-----------------------------------------------------( )--
|       |
|%M0002 |
+--] [--+
| << RUNG 6  STEP #0007 >>
|
|%M0002  %M0004  %M0008  %M0012  %M0016                           %Q0001
+--] [--+--]/[--+--]/[--+--]/[--+--]/[---------------------------( )--
|       |       |       |       |
|       |%M0006 |%M0010 |%M0014 |
|       +--] [--+--] [--+--] [--+
| << RUNG 7  STEP #0019 >>
|
|%M0001  %M0002                                                      %M0003
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0003 |
+--] [--+
| << RUNG 8  STEP #0023 >>
|
|%M0001  %M0003                                                      %M0004
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0004 |
+--] [--+
| << RUNG 9  STEP #0027 >>
|
|%M0004  %M0008  %M0016                                              %Q0002
+--] [--+--]/[--+--]/[-------------------------------------------( )--
|       |       |
|       |%M0012 |
|       +--] [--+
|
| << RUNG 10  STEP #0033 >>
|
|%M0001  %M0004                                                      %M0005
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0005 |
+--] [--+
|
| << RUNG 11  STEP #0037 >>
```

```
|%M0001  %M0005                                                    %M0006
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0006 |
+--] [--+
| << RUNG 12  STEP #0041 >>
|
|%M0001  %M0006                                                    %M0007
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0007 |
+--] [--+
|
| << RUNG 13  STEP #0045 >>
|%M0001  %M0007                                                    %M0008
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0008 |
+--] [--+
| << RUNG 14  STEP #0049 >>
|
|%M0008  %M0016                                                    %Q0003
+--] [-----]/[-----------------------------------------------------( )--
| << RUNG 15  STEP #0052 >>
|
|%M0001  %M0008                                                    %M0009
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0009 |
+--] [--+
| << RUNG 16  STEP #0056 >>
|
|%M0001  %M0009                                                    %M0010
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0010 |
+--] [--+
| << RUNG 17  STEP #0060 >>
|
|%M0001  %M0010                                                    %M0011
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0011 |
+--] [--+
| << RUNG 18  STEP #0064 >>
|
|%M0001  %M0011                                                   %M0012
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0012 |
+--] [--+
| << RUNG 19  STEP #0068 >>
|
|%M0001  %M0012                                                    %M0013
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0013 |
+--] [--+
```

```
| << RUNG 20  STEP #0072 >>
|
|%M0001  %M0013                                                    %M0014
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0014 |
+--] [--+
| << RUNG 21  STEP #0076 >>
|
|%M0001  %M0014                                                    %M0015
+--]/[--+--] [-----------------------------------------------------( )--
|       |
|%M0015 |
+--] [--+
| << RUNG 22  STEP #0080 >>
|
|%M0001  %M0015                                                  %M0016
+--] [--+--] [-----------------------------------------------------( )--
|       |
|%M0016 |
+--] [--+
| << RUNG 23  STEP #0084 >>
|
|%M0001  %M0016                                                    %M0017
+--]/[-----] [-----------------------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
```

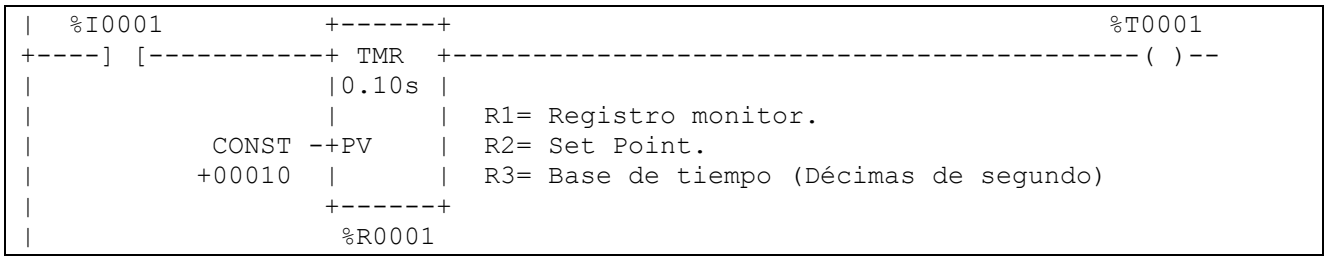


Cableado para el ejercicio N° 4.8.7

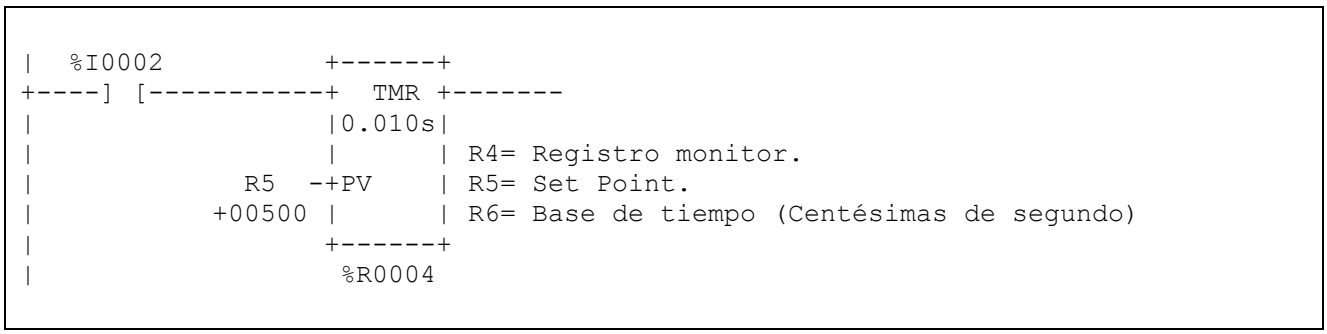
4.9 - Programación de circuitos donde interviene el tiempo
Funcionamiento del timer.

El timer se representa por un rectángulo, en el cual en la parte lateral izquierda se coloca el set point, este es el valor del tiempo que queremos encontrar, en la parte del centro se coloca la base de tiempo con la cual se desea trabajar y esta puede ser en décimas, centésimas o milésimas de segundo y por último en la parte inferior se representa el registro monitor o sea el reloj real acumulado del timer, finalmente del lado de la derecha se coloca la bobina que energizará el timer cuando el valor de este registro sea igual o mayor al valor del set point. Es importante aclarar que ésta bobina es recomendable representarla con la letra (T) de timer pero también se puede utilizar cualquier otra variable utilizada en este software como puede ser la de relevador interno (M) o de salida (Q). Incluso si se desea, se puede dejar sin colocar bobina de salida. También es importante mencionar que estos tres datos se deben colocar en un registro (R), y como podemos observar por cada timer se requieren tres registros. (Tres localidades de memoria).

NOTA: Para activar un timer se requiere un contacto como elemento de control.



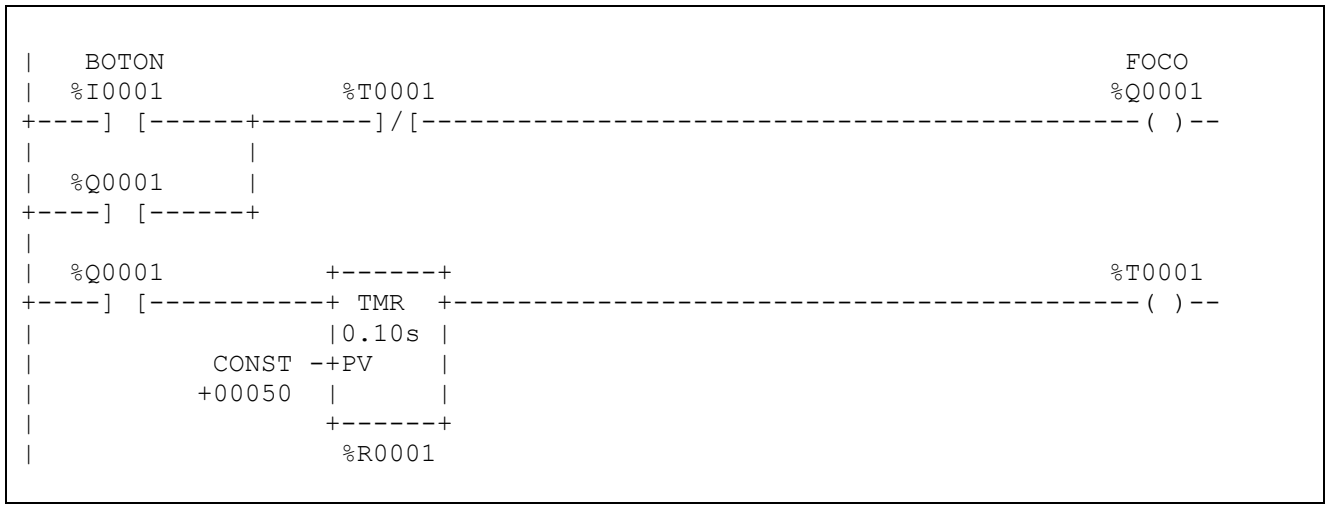
Funcionamiento del circuito mostrado; Al cerrar el contacto de I1 empieza a correr el tiempo, cuando el tiempo transcurrido monitoreado por R1 se iguala o se hace mayor que el valor de la constante, (en este caso 10), se energiza la salida T1. Para éste PLC en particular, el tiempo se calcula multiplicando la base de tiempo por el valor de la constante, por lo tanto se multiplica 0.1 X 10 = 1 seg.



Funcionamiento del circuito mostrado; Al cerrar el contacto de I2 empieza a correr el tiempo en este caso esta en centésimas de seg. Observe que no se tiene salida y por lo tanto se puede utilizar el timer para comparar el tiempo transcurrido R4 con un valor del registro R5 previamente especificado.

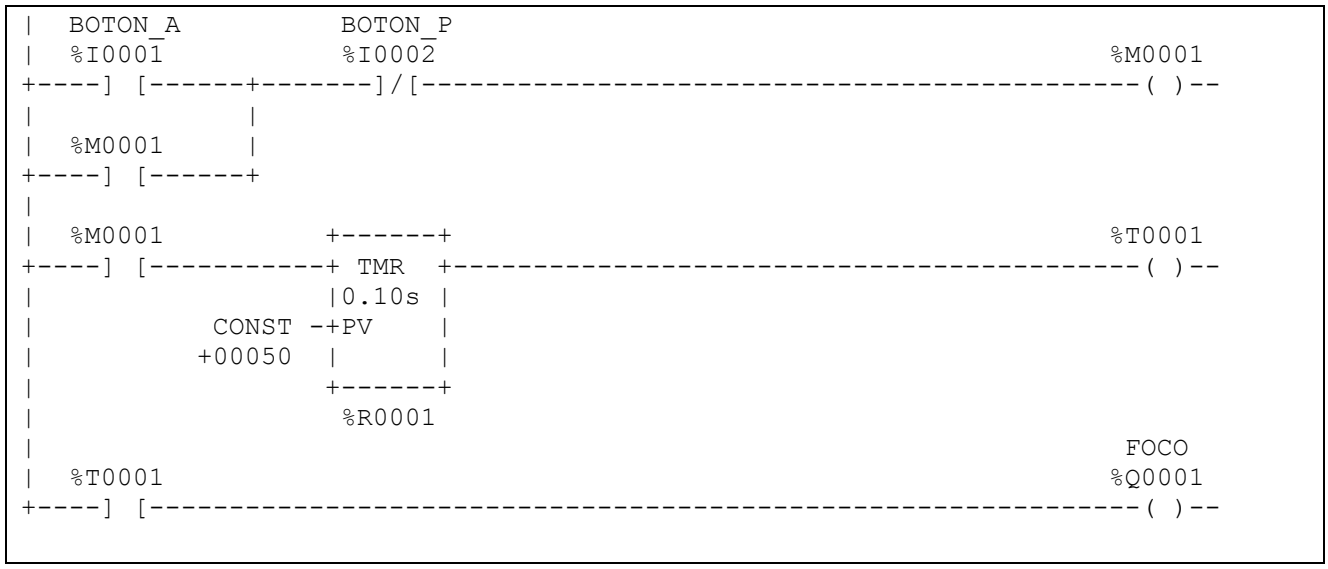
EJERCICIO N° 4.9.1

Diseñe el circuito de control para que al oprimir un botón se prenda un foco y cinco seg. después se apague.



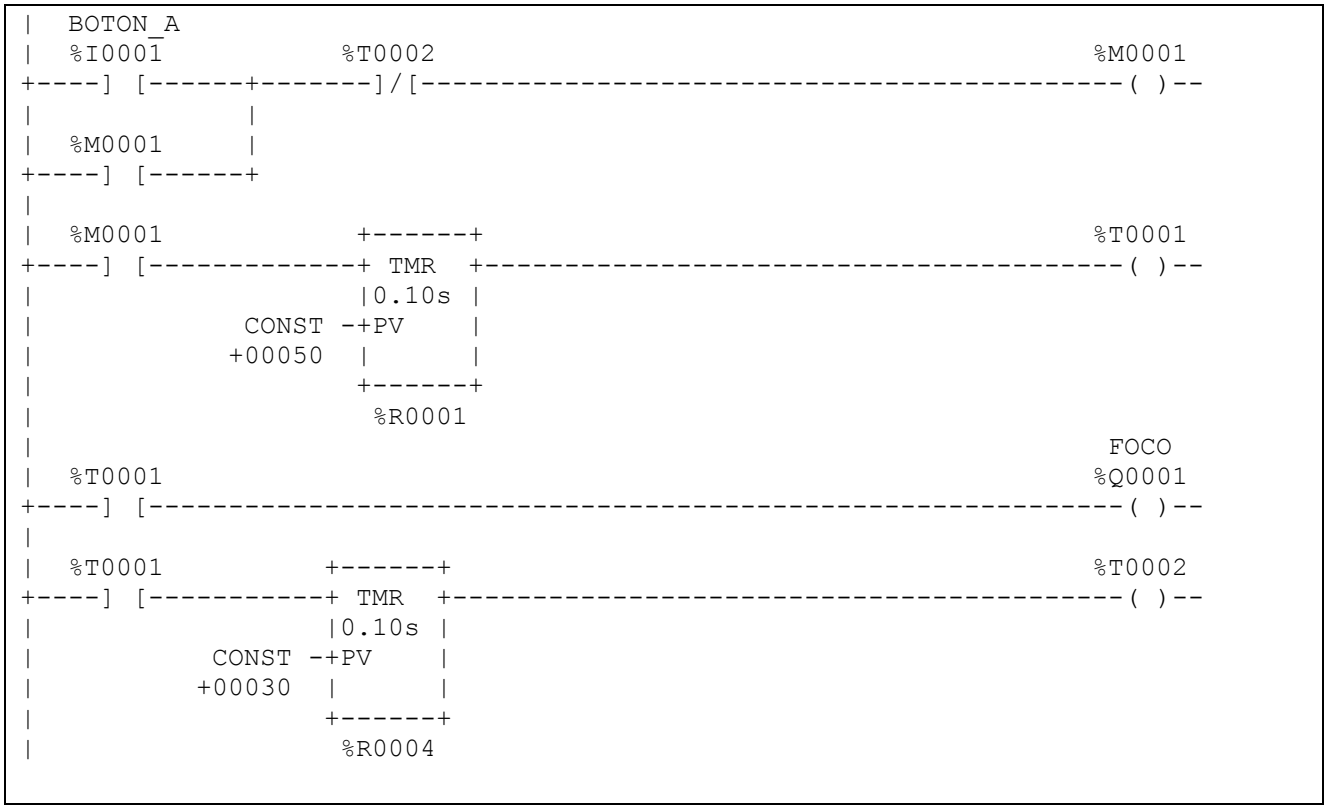
EJERCICIO N° 4.9.2

Diseñe el circuito de control para que al oprimir un botón empiece a transcurrir el tiempo y a los cinco seg. se encienda y permanezca encendido, para apagar el foco utilice un botón de paro.



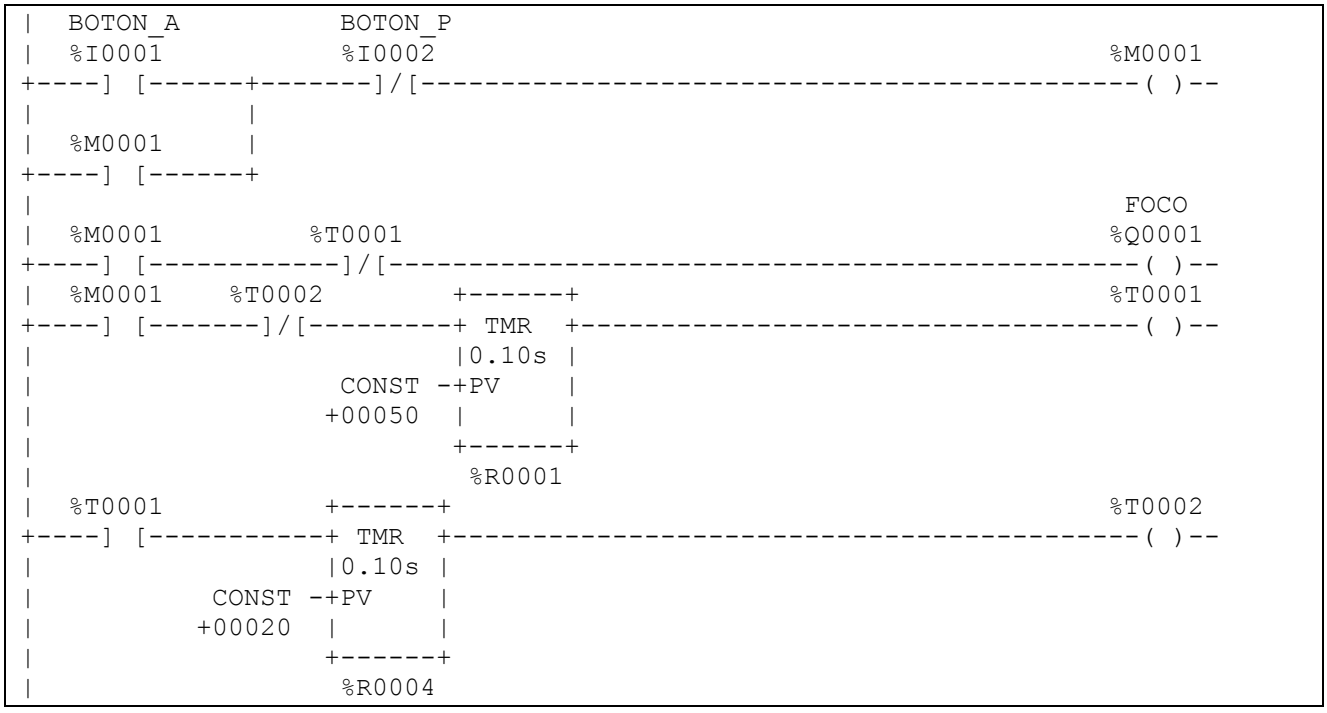
EJERCICIO N° 4.9.3

Diseñe el circuito de control para que al oprimir un botón empiece a transcurrir el tiempo y a los cinco seg. se encienda y permanezca encendido durante 3 seg, al concluir este tiempo se apague y así permanezca.



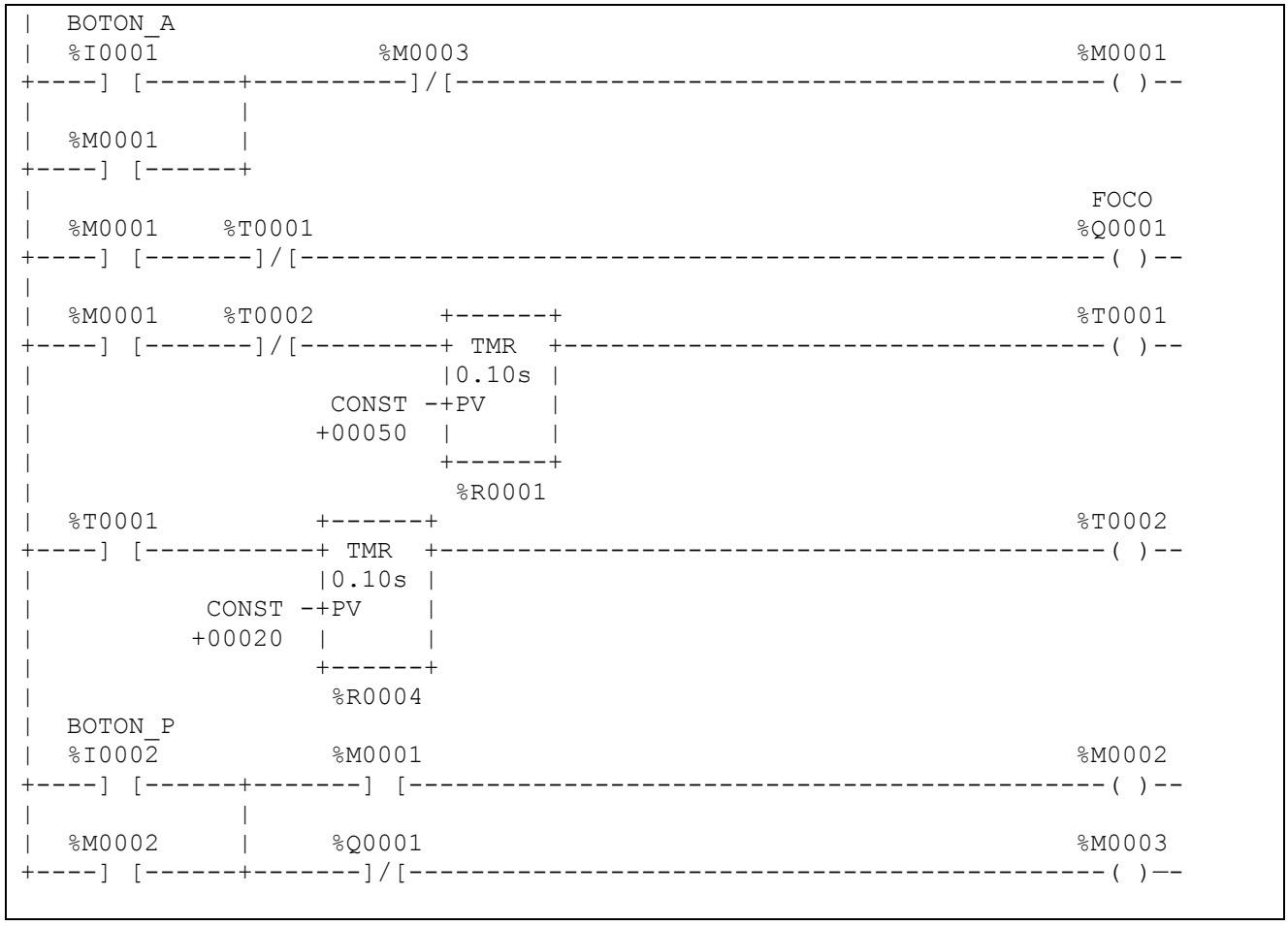
EJERCICIO N° 4.9.4

Diseñe el circuito de control para que al oprimir un botón se encienda un foco y deberá permanecer encendido durante cinco seg. y se deberá apagar, luego deberá permanecer dos seg. apagado y encender de nuevo y así sucesivamente. Para apagarlo utilice un botón de paro.



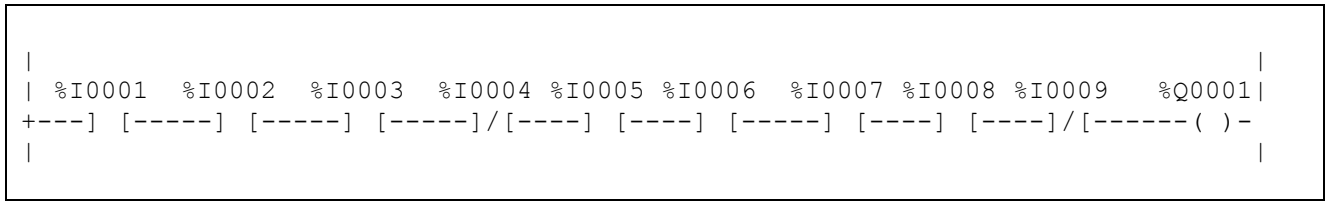
EJERCICIO N° 4.9.5

Diseñe el circuito de control para que al oprimir un botón se prenda un foco y deberá permanecer prendido durante cinco seg. y se deberá apagar, luego deberá permanecer dos segundos apagado y prender de nuevo y así sucesivamente. Para apagarlo utilice un botón de paro, pero ahora si el foco esta prendido cuando se oprime el botón de paro, deberá permanecer encendido y cuando se apague que ya no prenda, igualmente, si el foco esta apagado cuando se oprime el botón de paro que permanezca apagado.



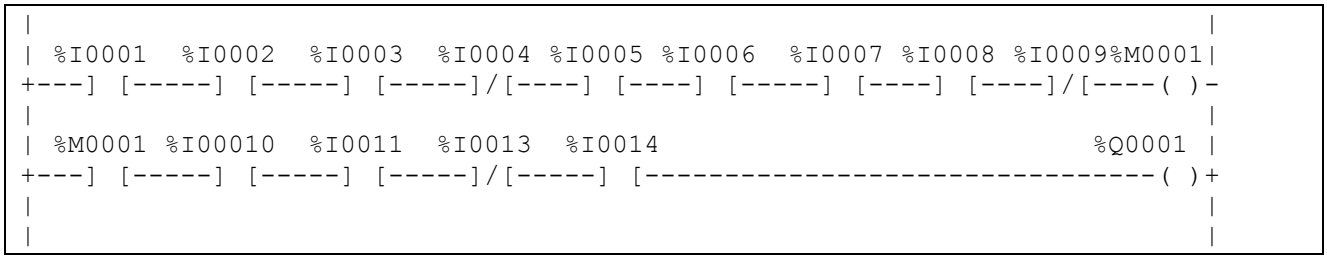
EJERCICIO N° 4.9.6

En el siguiente ejercicio se observa la cantidad máxima de contactos abiertos o cerrados (9) más una bobina que se pueden colocar en una línea.

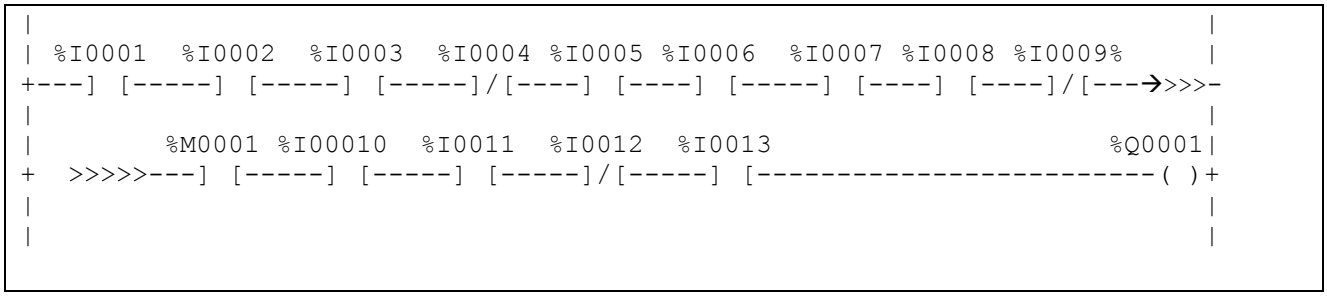


Si por alguna razón se desea colocar más de 9 contactos en serie hay dos alternativas:

a) Se coloca una bobina de relevador interno y en la siguiente línea se coloca un contacto de la bobina y se continúa con los contactos restantes.



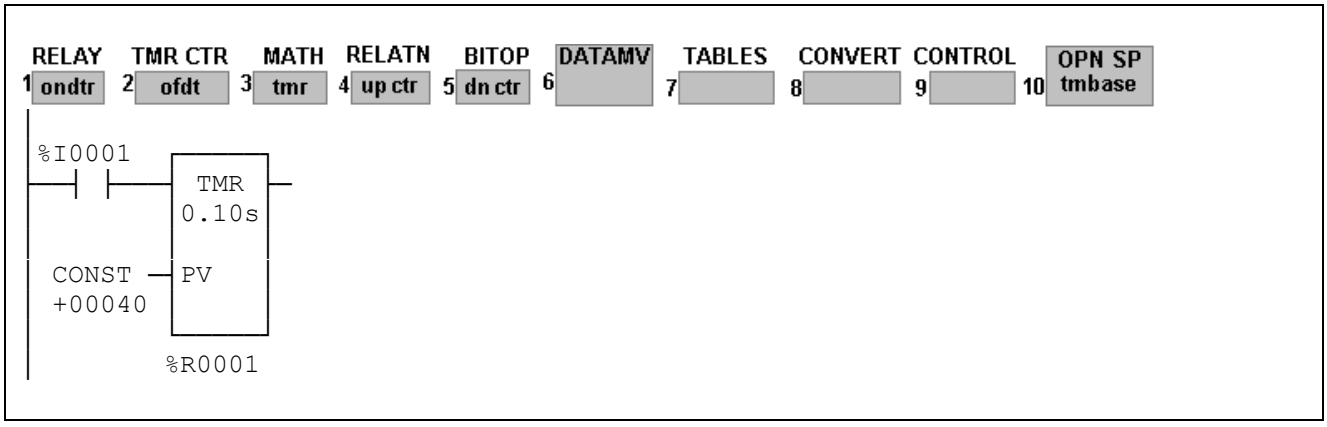
b) Se colocan los contactos en una línea y se coloca la función y se continúa en la siguiente línea.



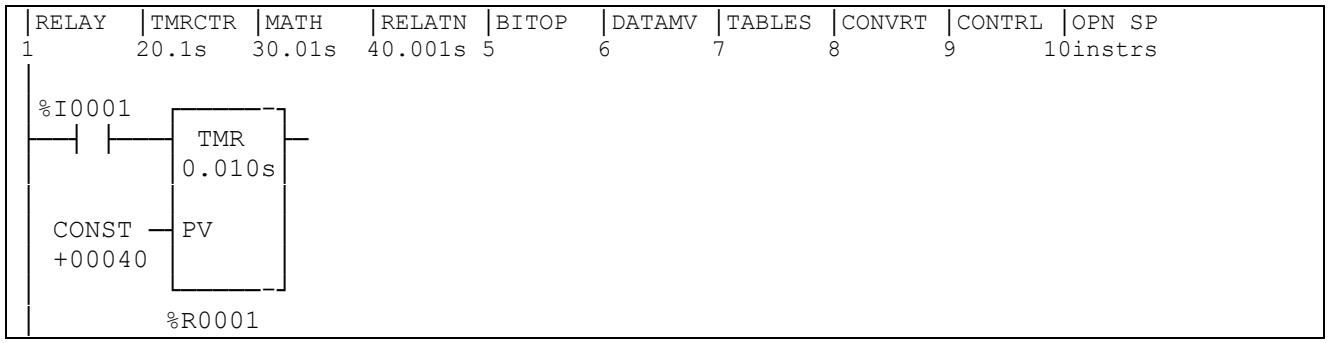
EJERCICIO N° 4.9.7

En el siguiente ejercicio se muestra la forma de colocar la base de tiempo para los timers:

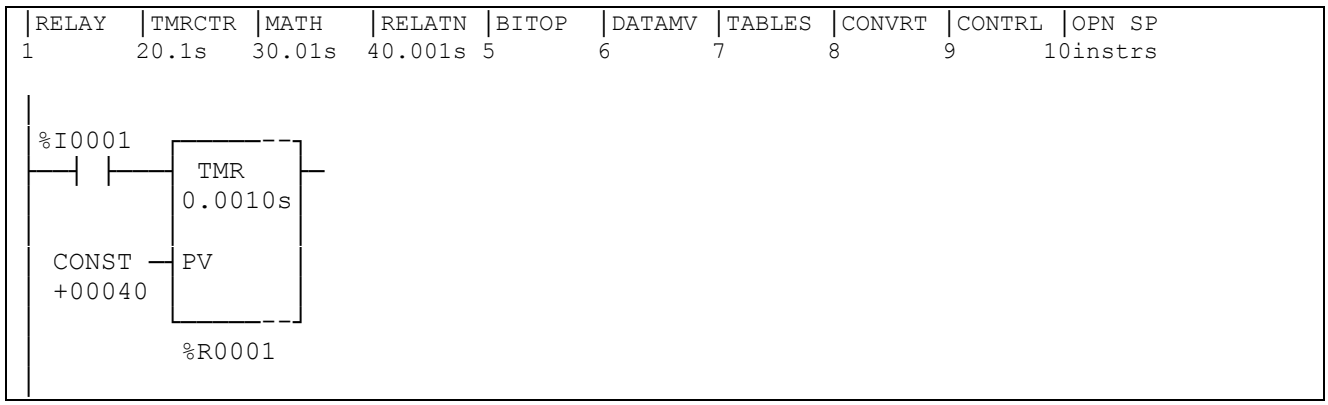
a) Primero se realiza el circuito del timer.



b) En seguida se oprime la tecla F10 y nos aparecen los siguientes iconos, si se desea obtener décimas de segundo se oprime la tecla F3.



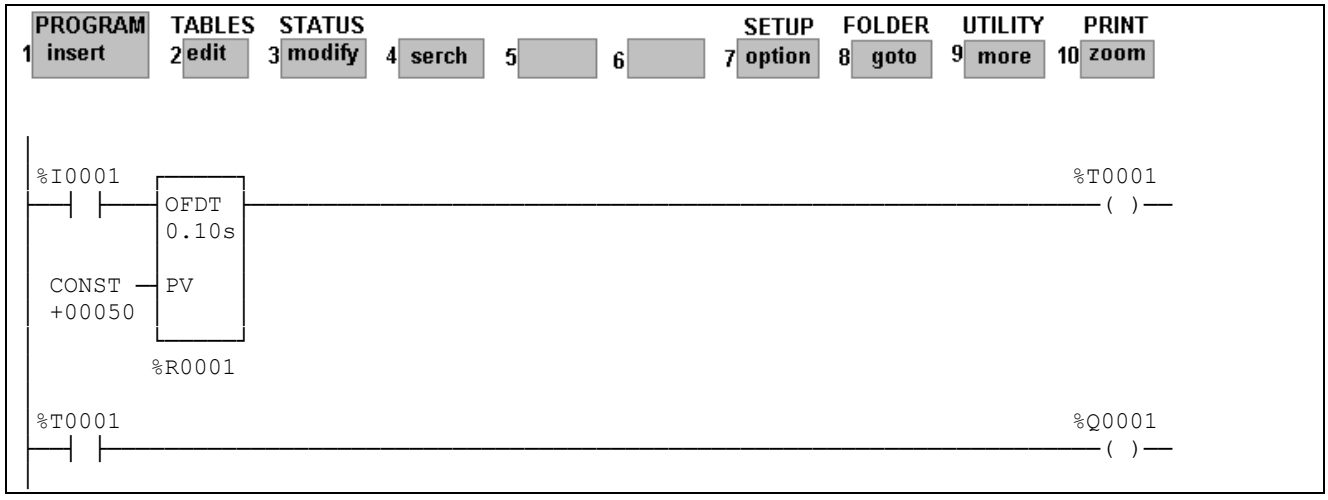
c.- si se desea tener como base de tiempo en centésimas de segundo se oprime la tecla F4.



EJERCICIO N° 4.9.8

Realice el circuito de control para que al oprimir un botón se prenda un foco y continúe energizado hasta que transcurran 5 segs.

Funcionamiento; al oprimir el botón I1, se energiza el timer y éste a su vez energiza la bobina T1 la cual energiza inmediatamente la bobina Q1, cuando soltamos el botón se desenergiza el timer iniciando su conteo, cuando completa el tiempo de 5 segs se desenergiza la bobina T1 y éste a su vez desenergiza Q1.



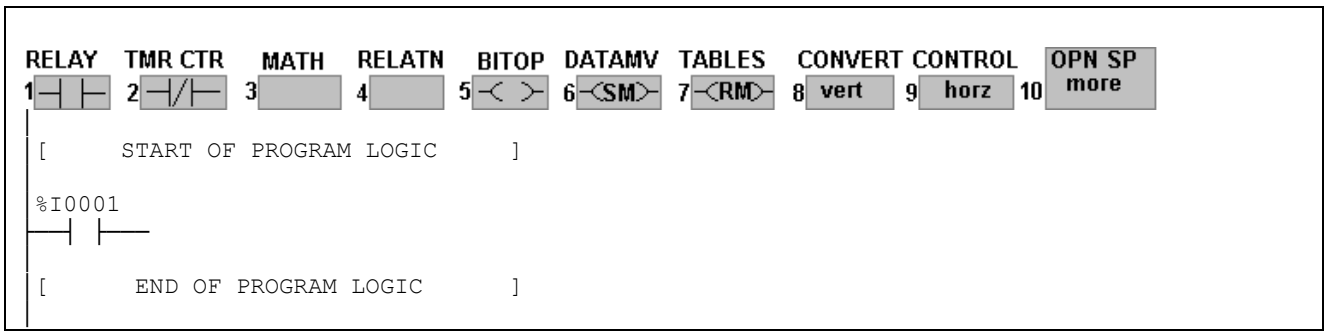
EJERCICIO N° 4.9.9

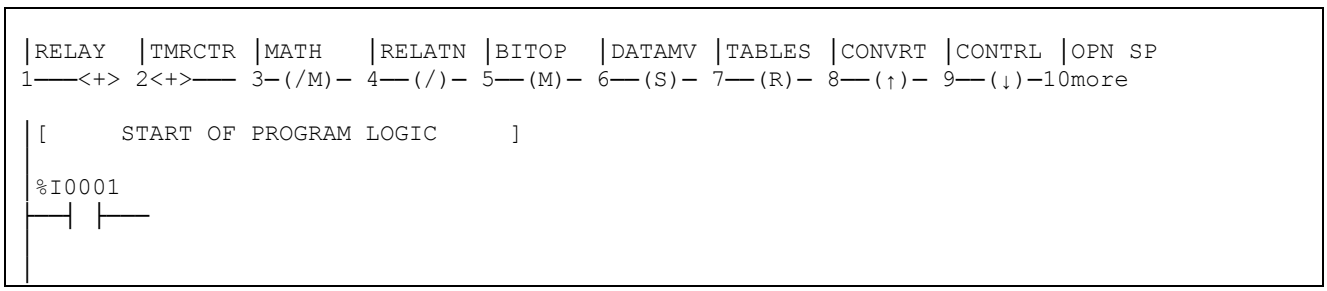
Realice el circuito de control para que al oprimir un botón se prenda un foco continúe energizado hasta que transcurran 5 seg, posteriormente se apaga y permanece apagado hasta que se oprima de nuevo I1.

Como podemos observar en el ejercicio anterior, el tiempo empieza a correr a partir de que se desactiva el botón, lo que quiere decir que el tiempo empieza a transcurrir cuando se suelta el botón; en el presente ejercicio se desea que el tiempo empiece a correr en cuanto se active el botón sin importar cuanto tiempo permaneció activado.

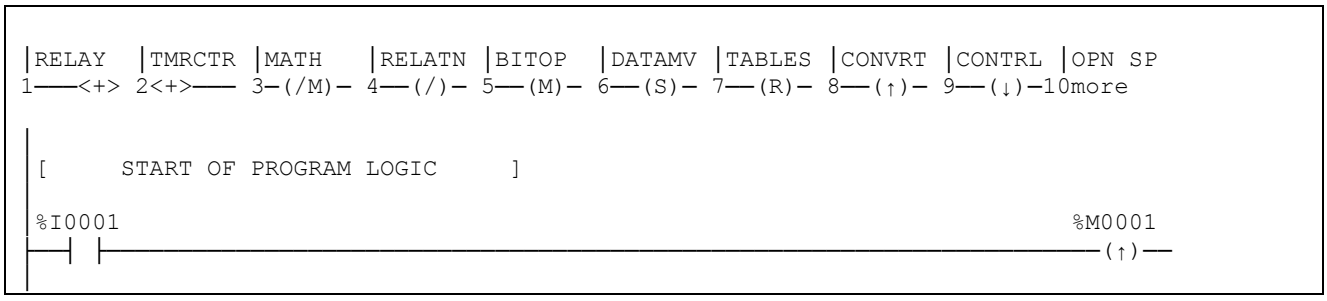
Primeramente vamos a explicar como funciona el circuito de one shot.

Al activar el botón I1 se debe energizar la bobina M1 en el franco se subida, dando sólo un disparo y se desenergiza aun cuando la energía continué en la bobina. Observe que en este circuito no se observa el ícono con el one shot, por lo tanto oprimimos la tecla shft F10 (more) y nos muestra la siguiente figura.

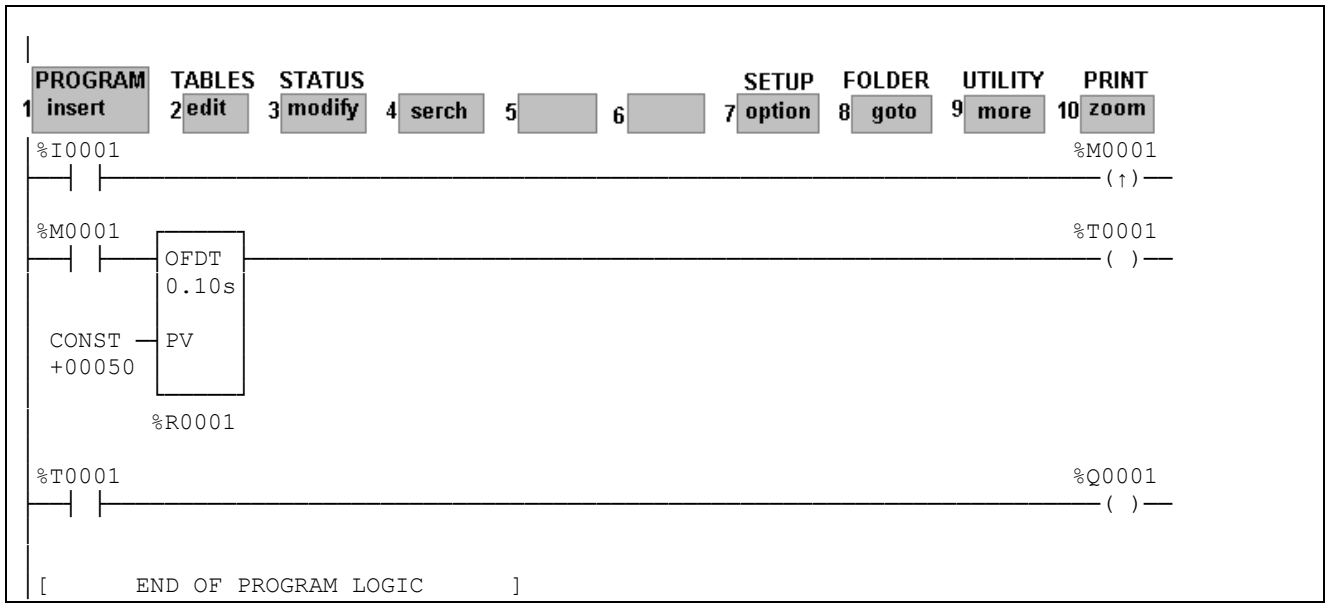




Ahora si podemos observar con F8 el ícono con la flecha hacia arriba indicándonos que se energiza sólo un pulso en el franco de subida.



Funcionamiento; al oprimir el botón I1, se energiza la bobina M1 un impulso, éste a su vez energiza y desenergiza el timer, al desenergizarse el timer empieza a transcurrir el tiempo.



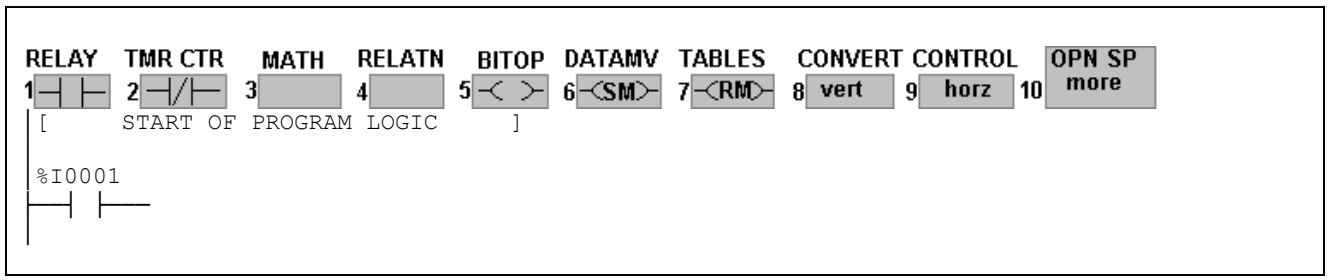
EJERCICIO N° 4.9.10

A continuación presentamos el funcionamiento de un timer especial cuyo nombre es watch dog delay y cuyo funcionamiento es como sigue:

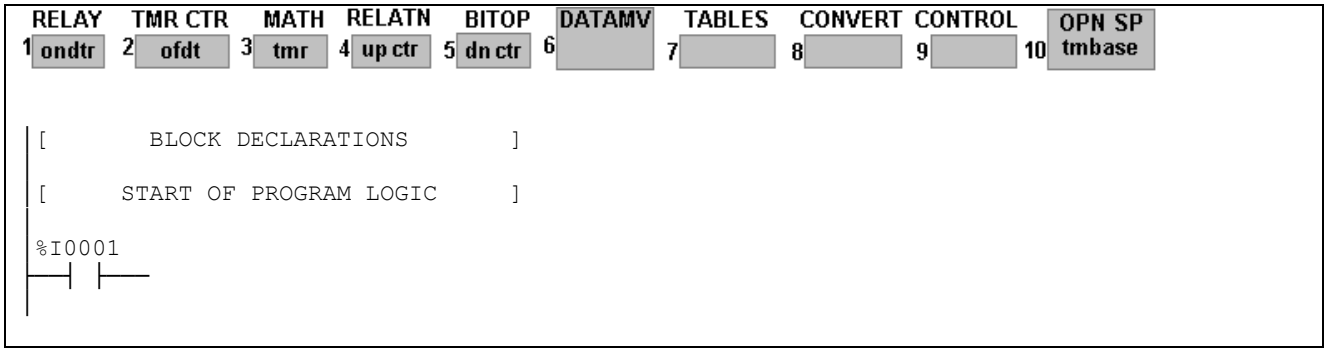
Al cerrar el contacto de I1 empieza a transcurrir el tiempo y al desactivarlo se detiene el tiempo, si oprimimos de nuevo el contacto de I1 continúa transcurriendo el tiempo y así sucesivamente. Para resetear el timer se oprime el botón I2.

Para programarlo siga las instrucciones que a continuación se muestran.

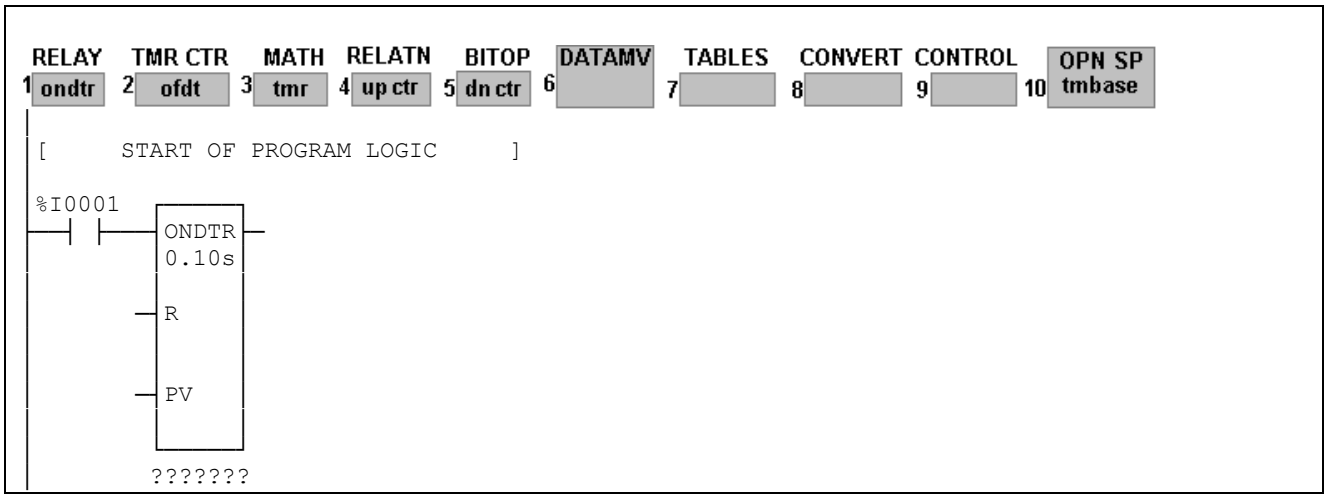
Primero oprimimos la tecla insertar y colocamos el contacto de I1 coco ya se explico este paso anteriormente y tenemos la siguiente pantalla.



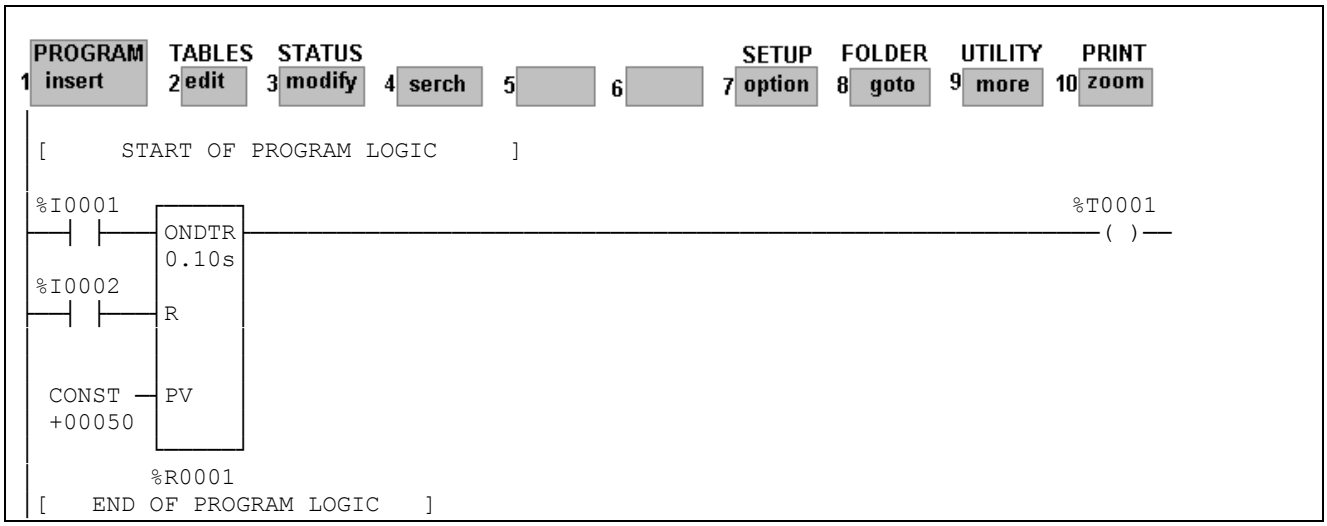
Luego, oprimimos shft F2 para obtener la función de timer como nos muestra la siguiente figura.



Ahora oprimimos la tecla F1 para activar ondtr tal como lo vemos en la siguiente pantalla



Finalmente completamos el circuito.



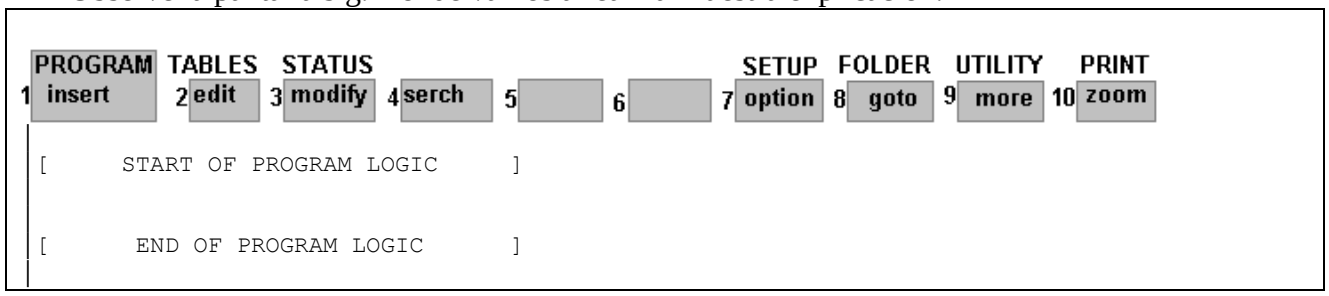
4.10 Programación de circuitos con contadores

Para programar los contadores primeramente mencionaremos que existen dos tipos de contadores los ascendentes y los descendentes, y al igual que los timer requieren de tres registros para su utilización.

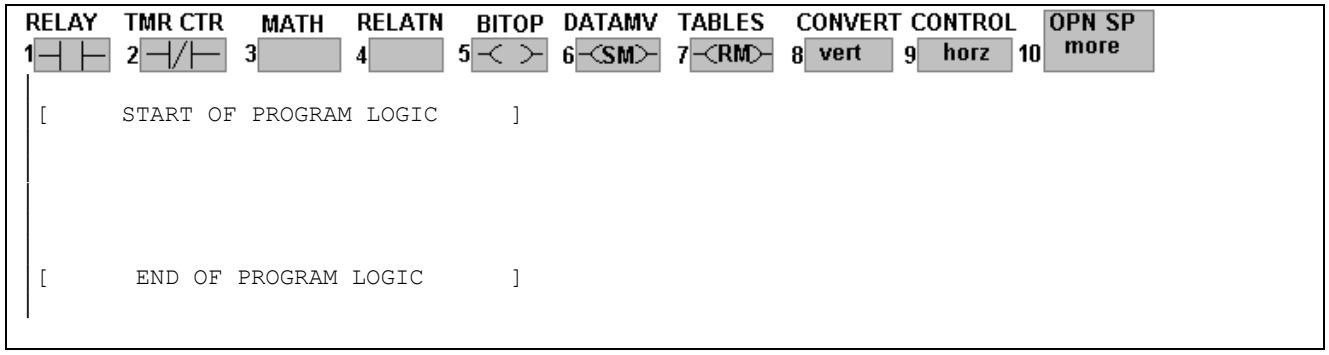
Funcionamiento:

El contador requiere de dos estradas una de clock y una de reset, también tiene un PV (valor de preset), un registro monitor y una salida, la cual al igual que el timer si se quiere se puede programar o no. Para activar el contador se requiere de un elemento de control para activarlo.

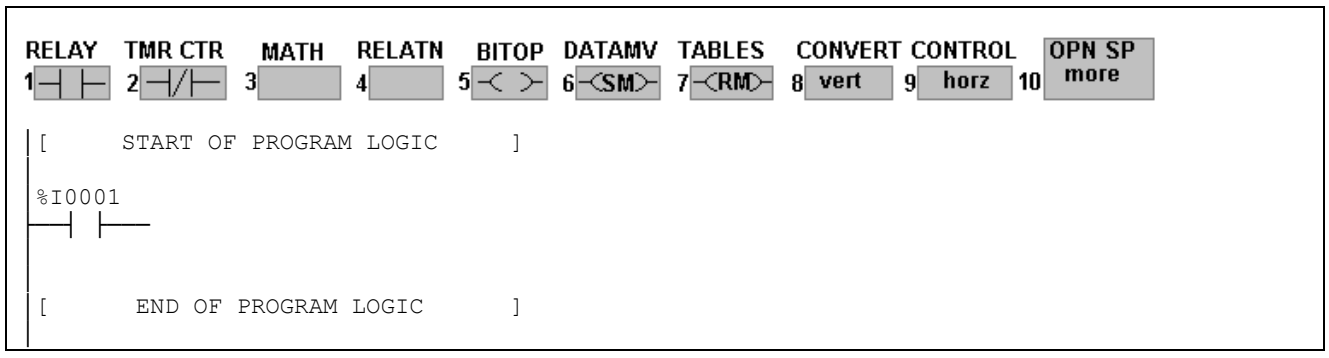
Observe la pantalla sig. Donde vamos a realizar nuestra explicación.



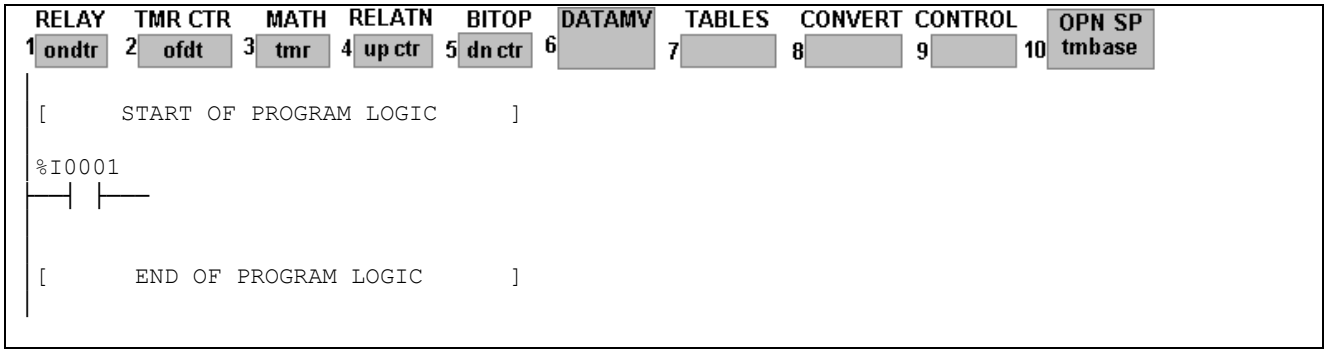
Primero colocamos la función F1 (insertar) para iniciar una sesión tal como lo vemos en la pantalla sig.



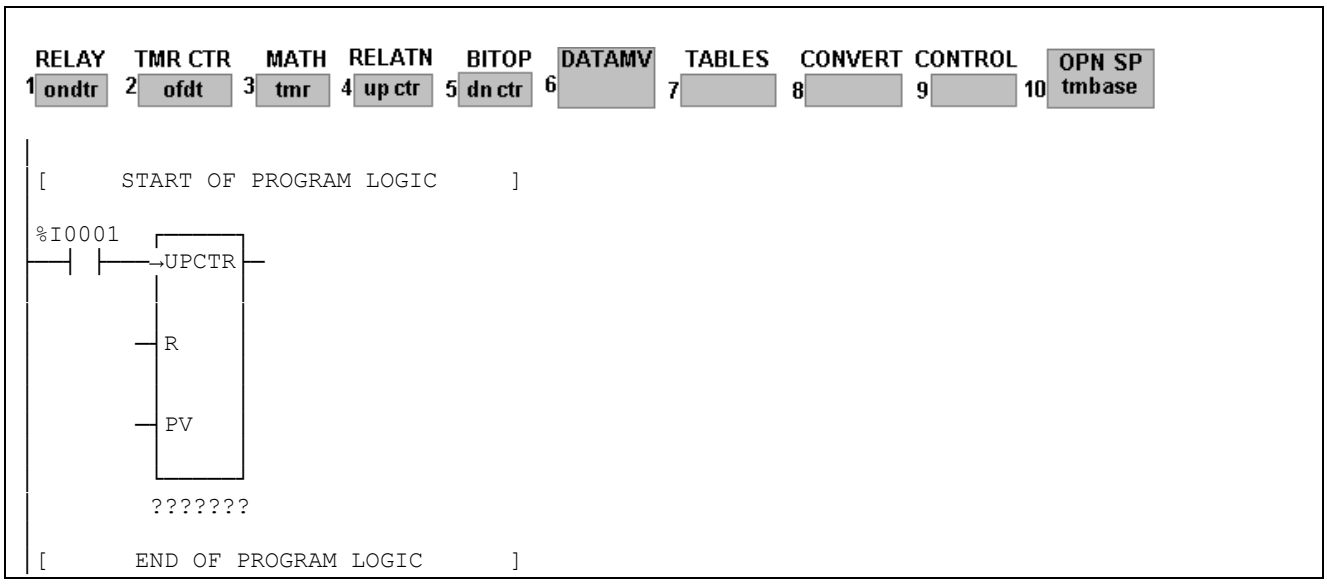
Posteriormente colocamos un elemento de control, en este caso vamos a colocar un contacto de I1 para dar pulsos al contador, tal como lo muestra la sig. pantalla.



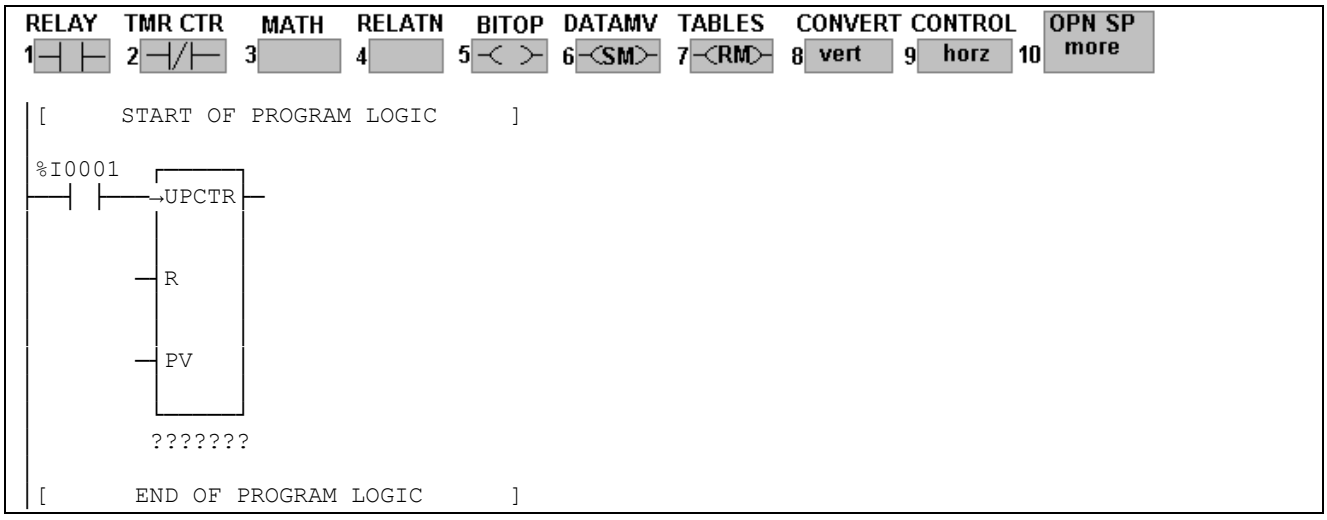
En seguida oprimimos la tecla shft F2 de TMRCTR con lo cual se muestra la sig. pantalla



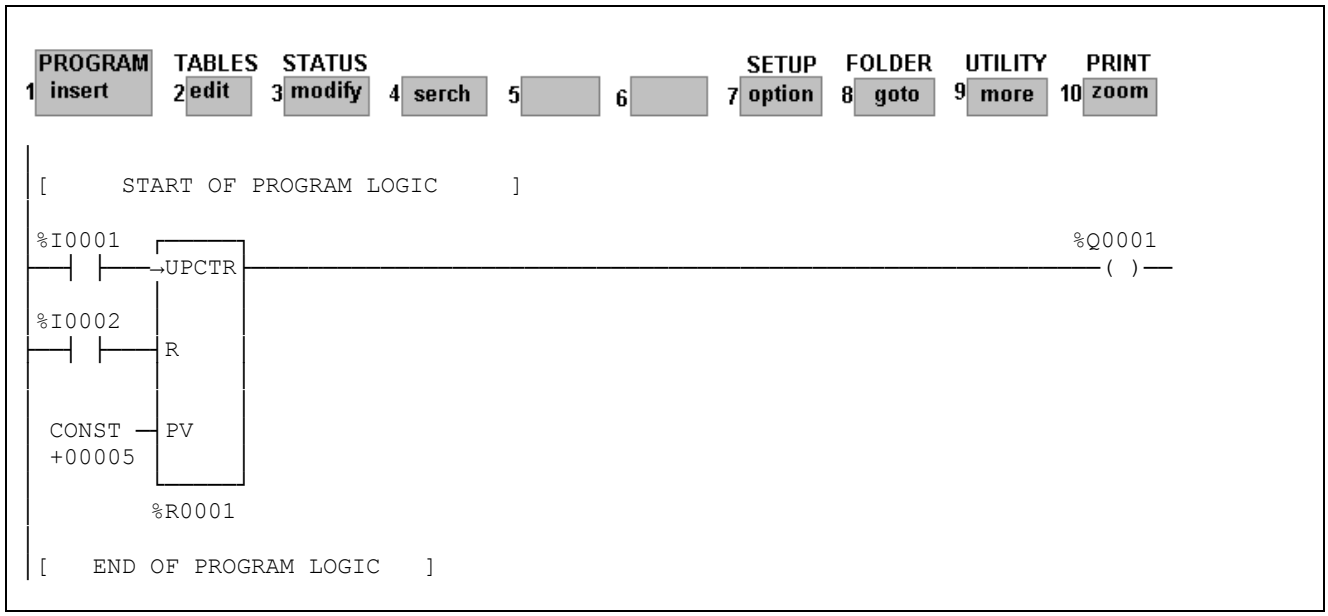
En la pantalla anterior observamos como aparecen los contadores ascendentes y descendentes, procedemos a oprimir la tecla shft F3 para llamar al contador ascendente.



Para completar los datos que nos muestra este bloque oprimimos la tecla shft F1 que es una función relevador, con lo cual nos muestra la siguiente pantalla



Finalmente procedemos a completar la información del bloque de contador.



Ahora procederemos con su funcionamiento. Al oprimir el contacto I1 empieza a contar cada vez que se oprima el botón I1 y al llegar a 5 pulsos el contador energizara Q1.

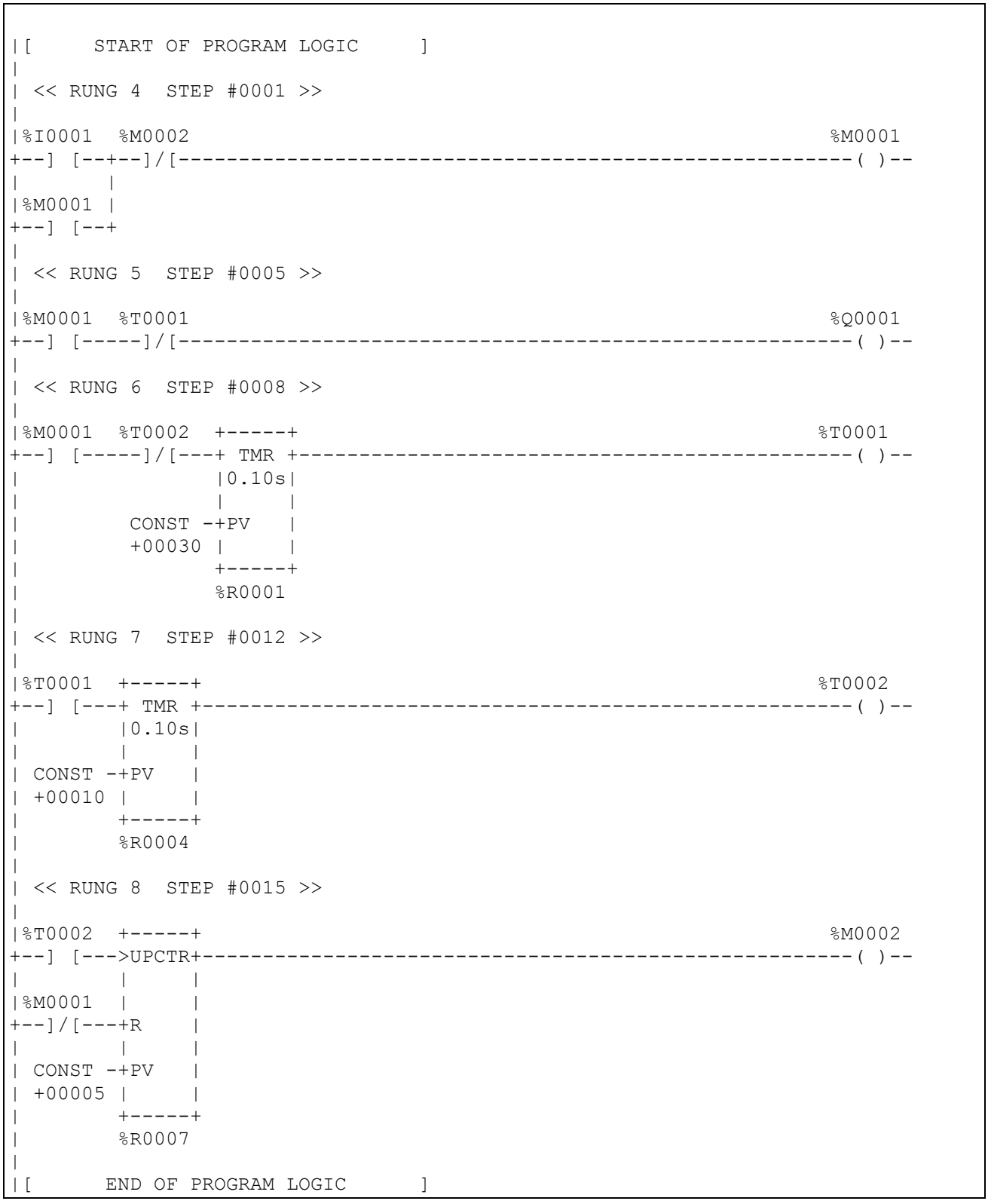
Para resetearlo oprimimos el botón I2, quedando el circuito listo para realizar otro conteo.

NOTA: El funcionamiento del contador descendente DWCTR es similar al anterior, con la observación de que el conteo es descendente y cuando el contador llega a cero entrega su salida.

NOTA: Cuando se tiene en un circuito varios contadores y timer se debe tener cuidado de no repetir o utilizar registros ya ocupados en el mismo circuito ya que esto traería resultados inesperados, recuerde que por cada timer y contadores se requiere de tres localidades del registro.

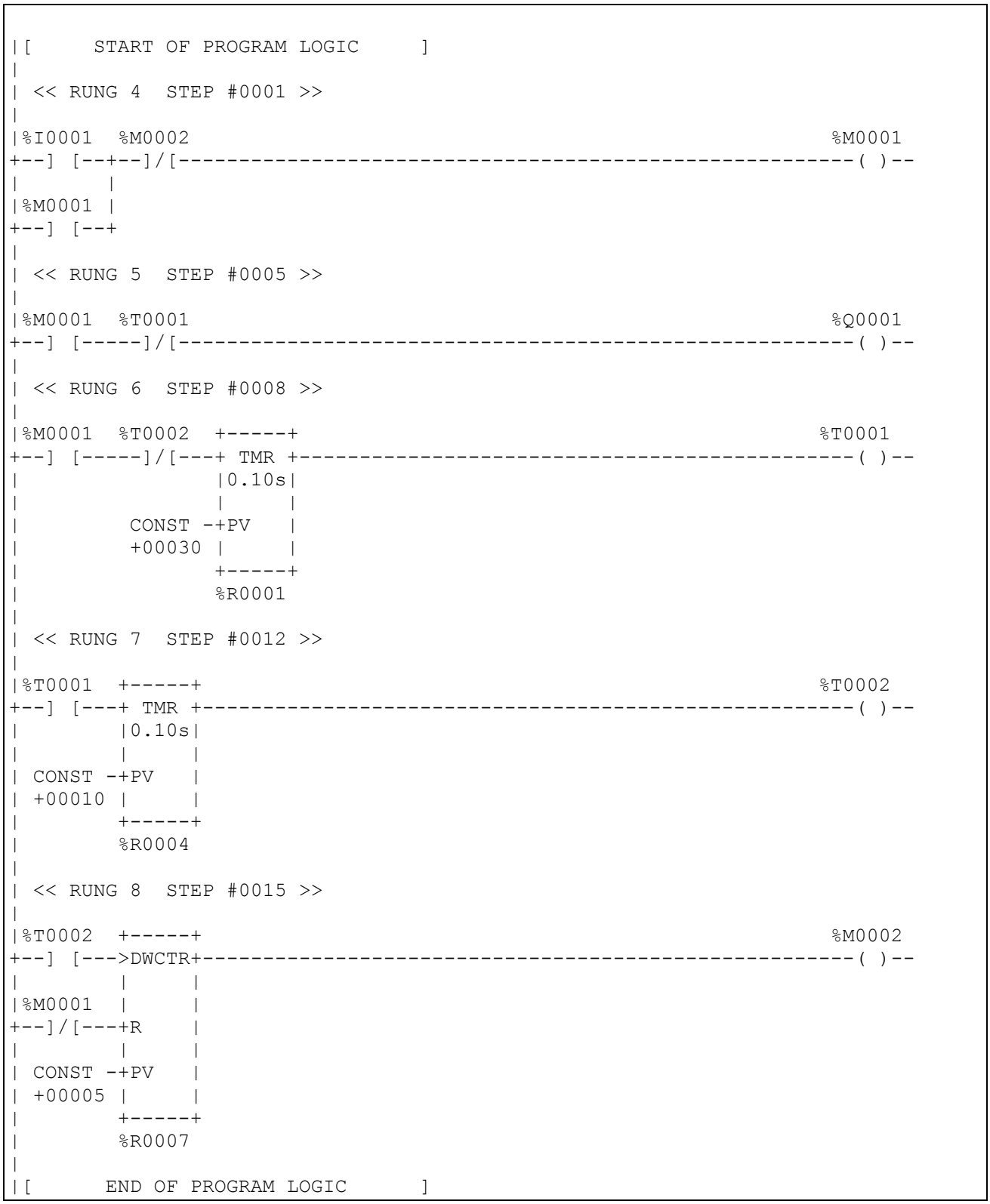
EJERCICIO 4.10.1

Diseñe el circuito de control en el cual se requiere al oprimir un botón se prenda un foco que dure 3 seg. prendido y que se apague, deberá durar 1 seg. apagado y volver a prender, durante 5 veces, apagarse y quedar en condiciones de iniciar de nuevo si se oprime el botón I1.



EJERCICIO 4.10.2

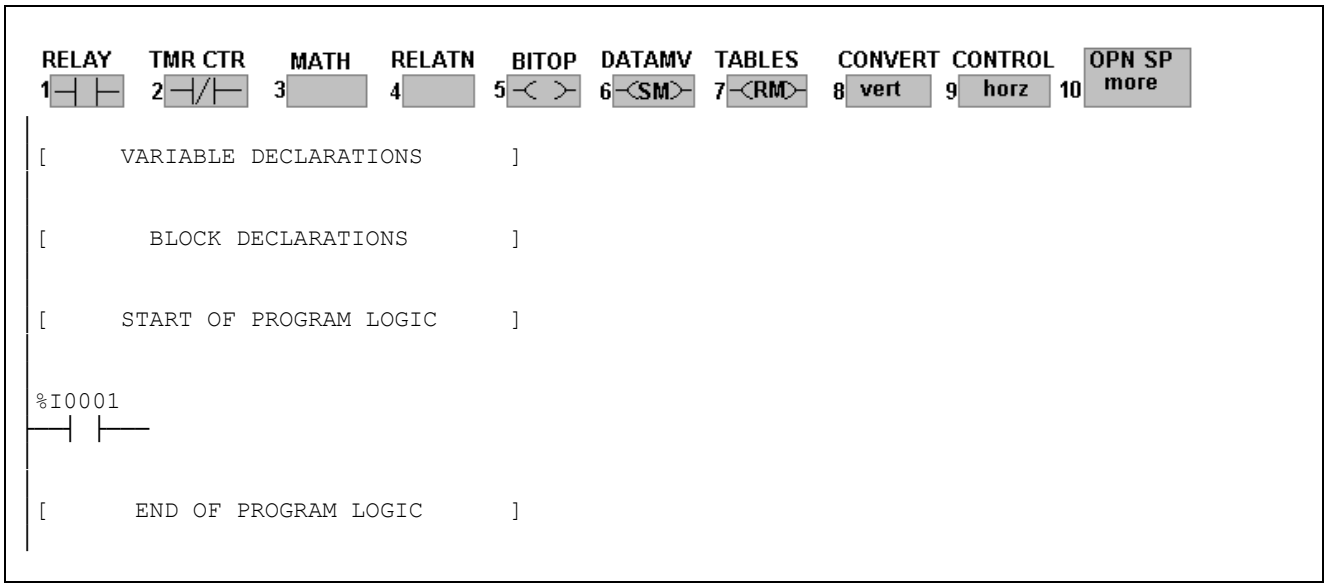
Diseñe el circuito del problema anterior utilizando el contador descendente.



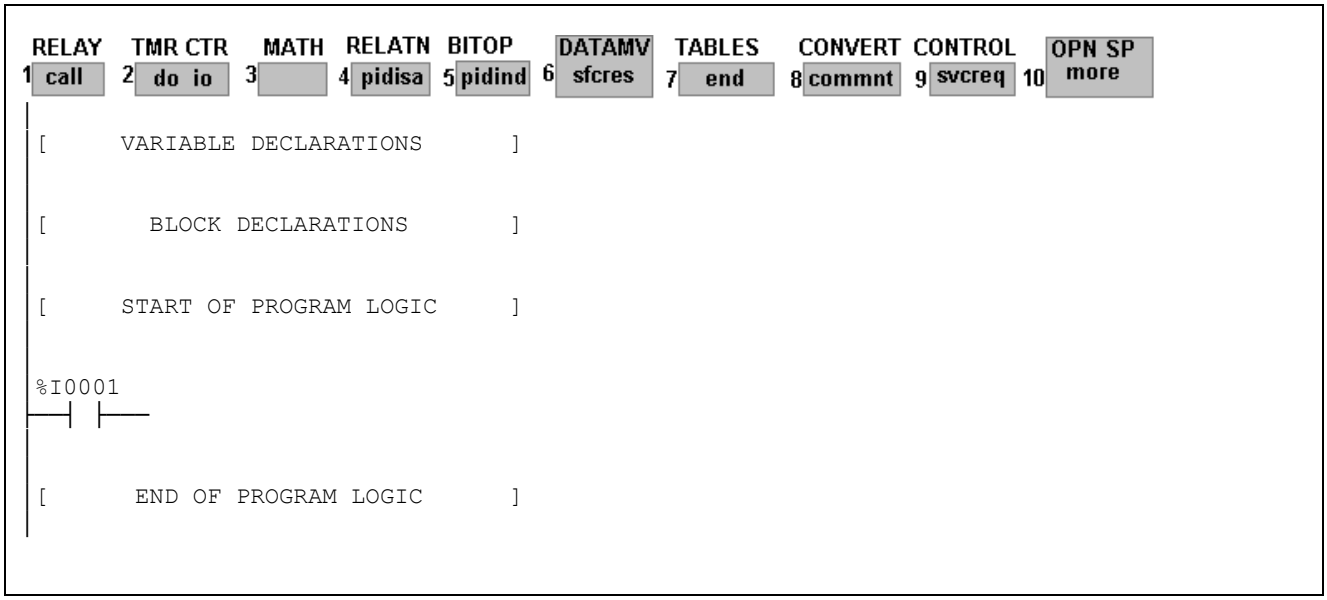
4.11 Programación de circuitos con la función master control y la función jump.

El funcionamiento de este master control es similar a la función electromecánico de permisivo, esto significa que si se hace presente este master control todo lo que esta dentro se reseteara, observe que se debe poner un inicio y un final.

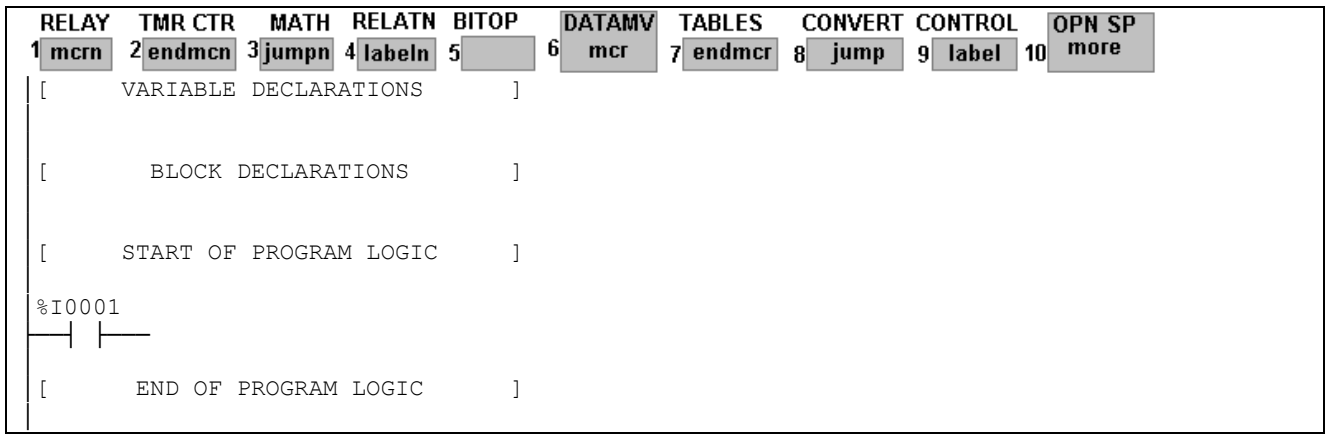
Primero oprimimos la tecla insertar para iniciar una sesión, con lo cual nos aparece la siguiente pantalla y colocamos un contacto necesario para habilitar el master control.



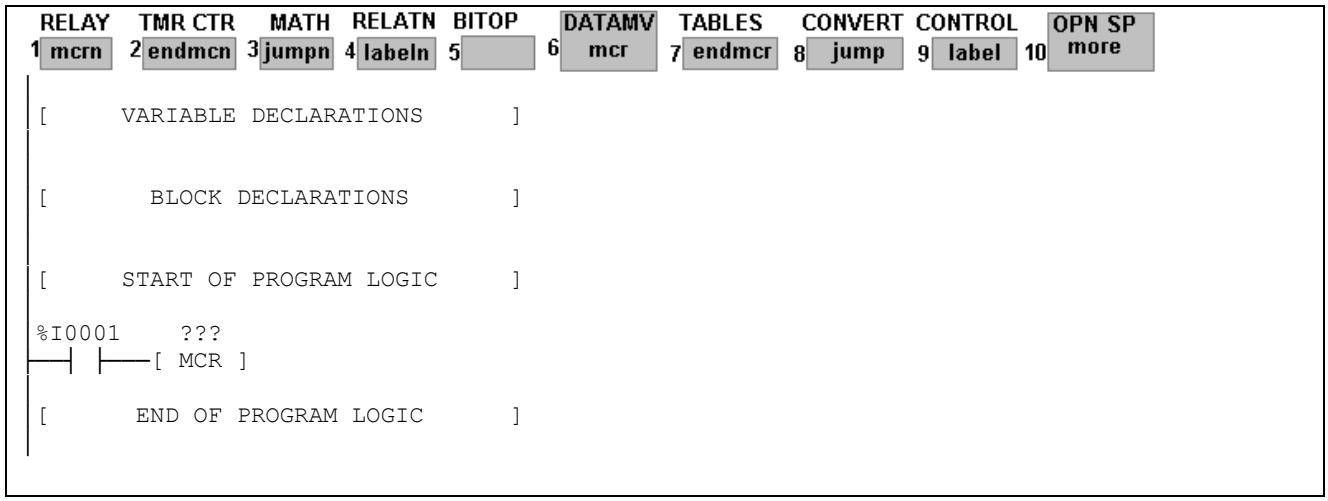
Para colocar el master control se utiliza la tecla shift+ F9 (CONTRL), con lo que nos aparece la sig. Pantalla.



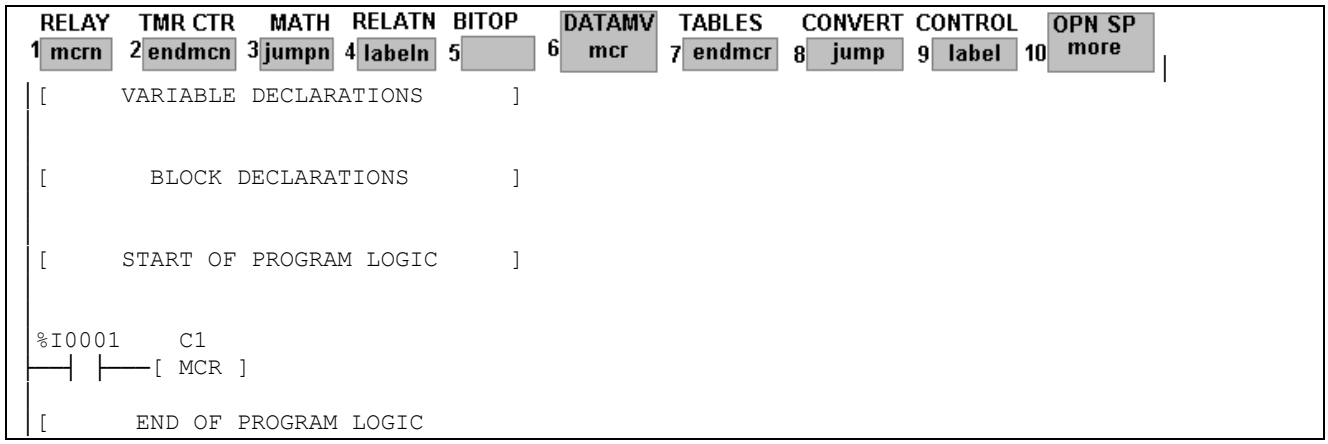
Luego la tecla F10 (more), con lo cual nos aparece la sig. Pantalla.



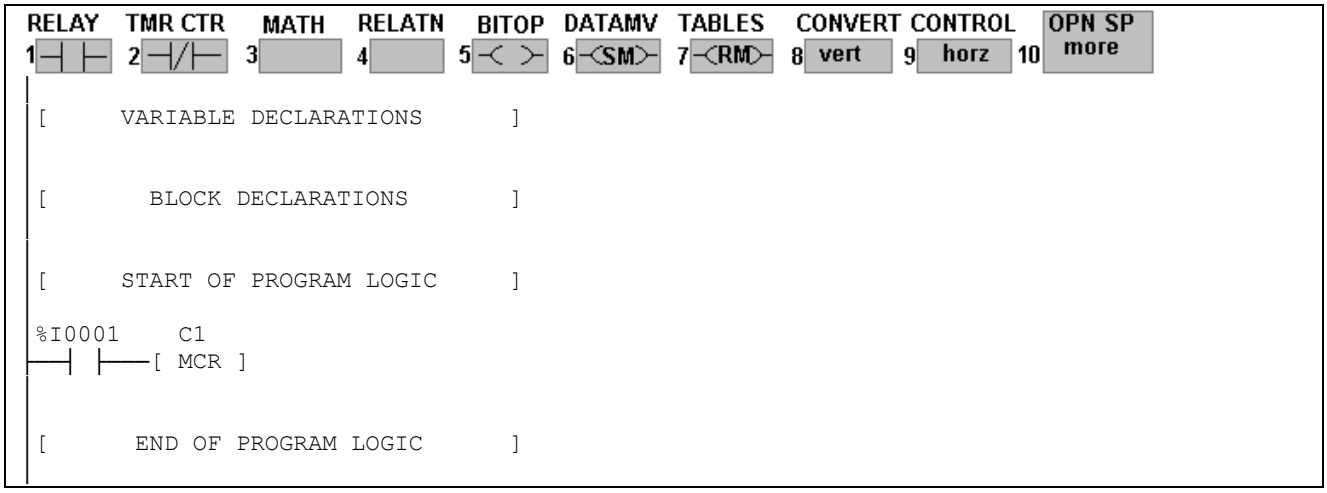
Para luego oprimir la tecla F6(MCR), con lo cual nos aparece la sig. Pantalla.



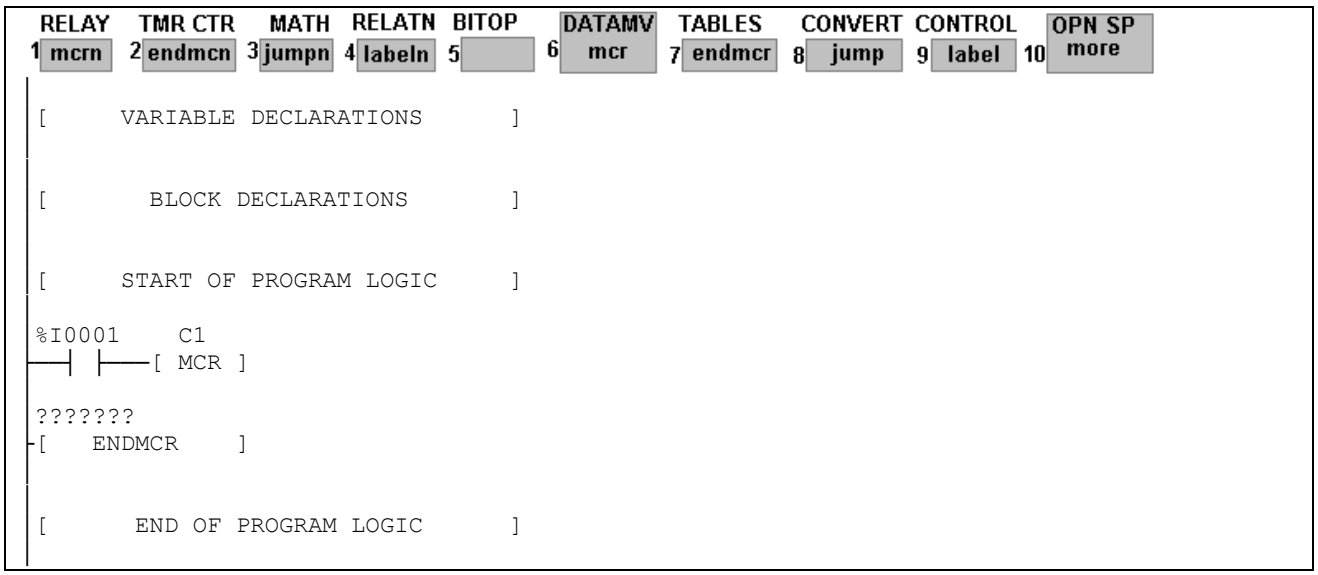
Para diferenciar un master control de otro se le pone una identificación en este caso se puso C1.



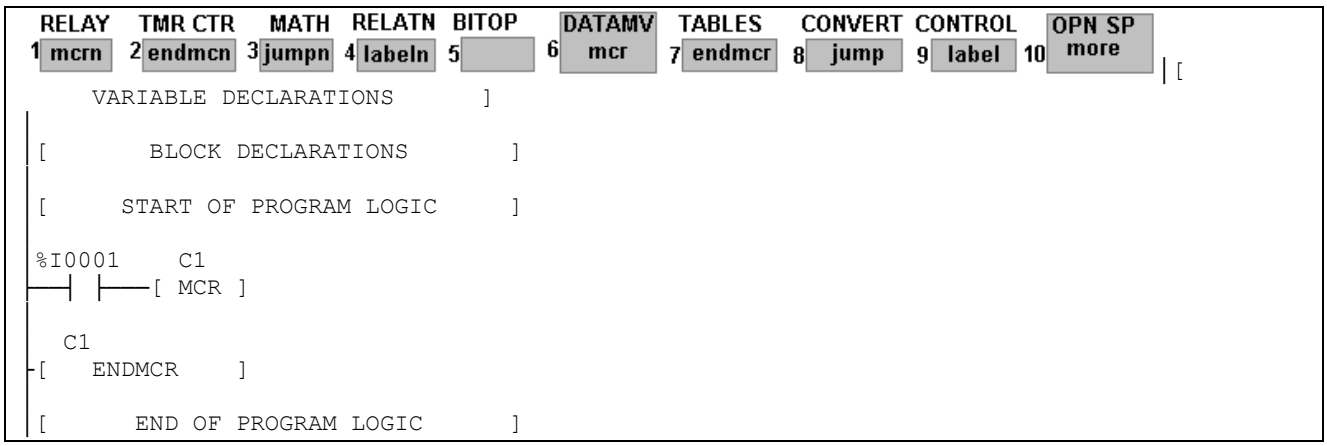
Una vez colocado el master control se procede a colocar el final del master control, para ello se oprime la tecla ENT, y aparece la siguiente pantalla



Aquí realizamos el mismo proceso de colocar el master control pero ahora colocamos el (END MCR)



Finalmente le colocamos la identificación misma del master control que en este caso fue C1



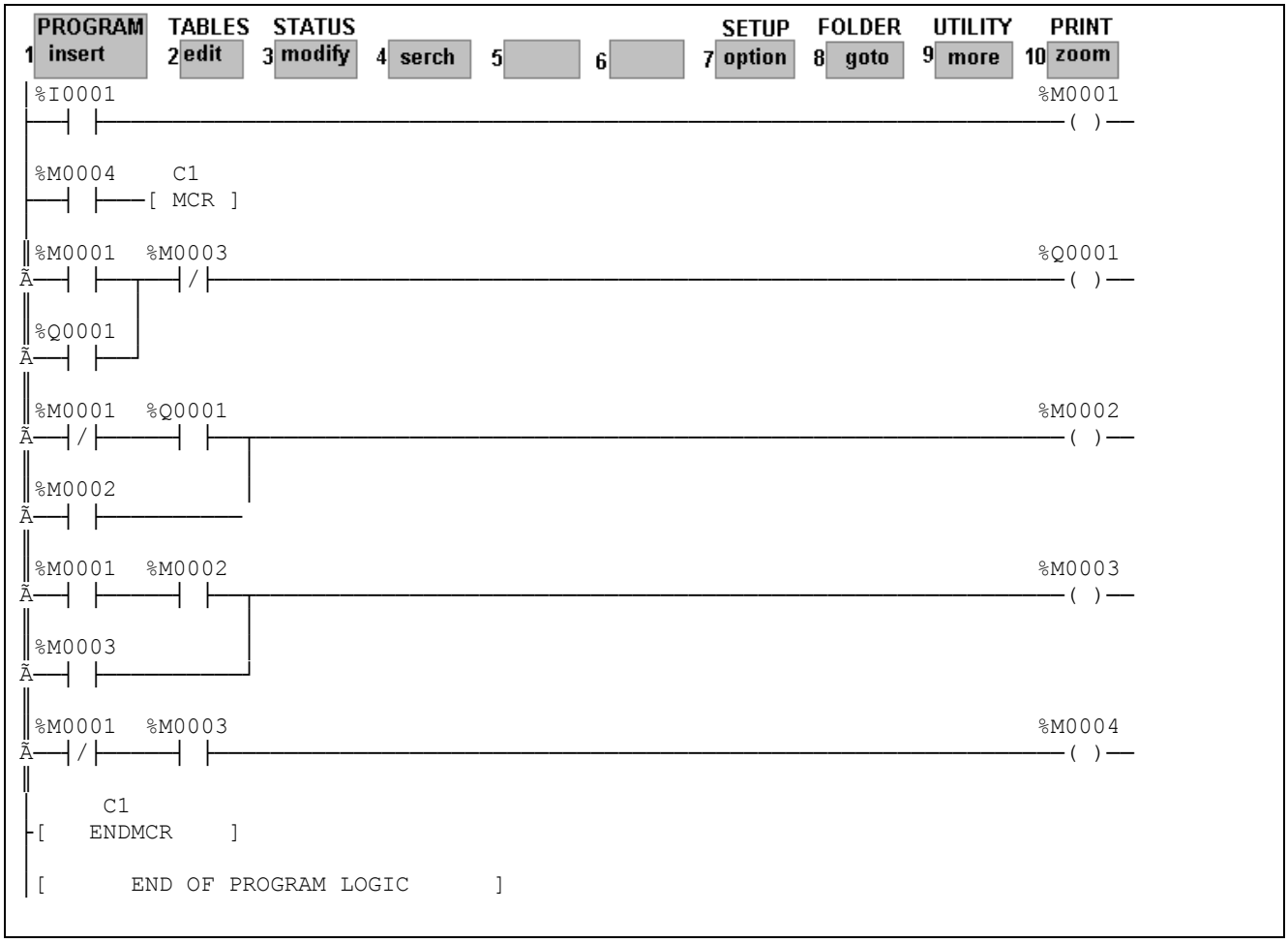
Por ultimo oprimimos la tecla ESC para finalizar

EJERCICIO 4.11.1

Diseñe el circuito de control que realice las siguientes funciones:

1.- Que al oprimir el boton 1, prenda el Foco 1 y al soltarlo permanezca prendido.
2.- Al oprimir de nuevo el boton 1 se apague el foco 1 y permanezca apagado y quede en condiciones de iniciar de nuevo.
3.- Al oprimir de nuevo el boton 1 se energice el foco 1 y así sucesivamente

SOLUCION:



EJERCICIO 4.11.2

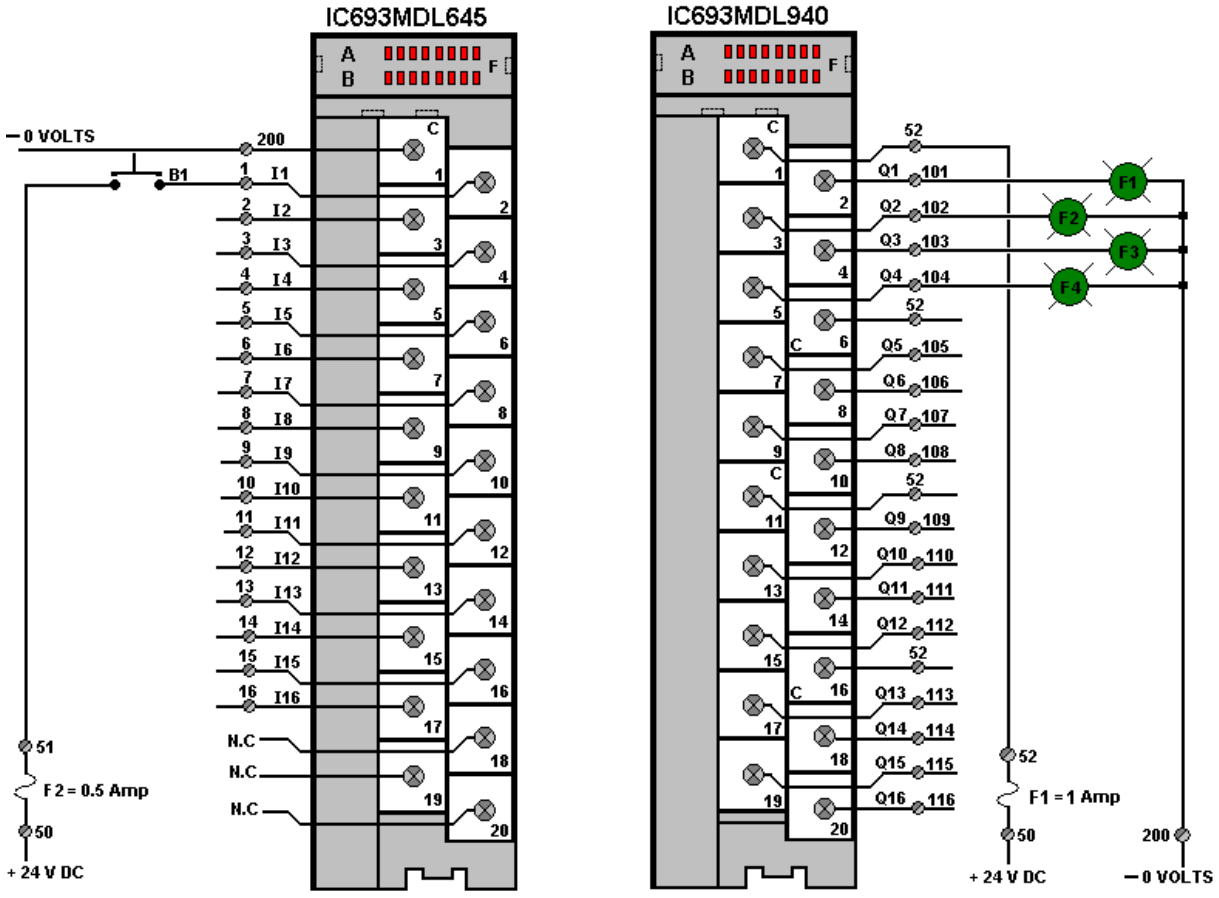
Realice el circuito de control para un control de anillo de cuatro bits utilizando el master control.

```
|[  START OF LD  PROGRAM

|[     VARIABLE DECLARATIONS      ]

              V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
```

```
          ---------     --------     --------------------------------
          %I0001                     BOTON_1
          %Q0001                     FOCO_1
          %Q0002                     FOCO_2
          %Q0003                     FOCO_3


|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
 BOTON_1
|%I0001                                                              %M0001
+--] [-------------------------------------------------------------( )--
|
|%M0010    C1
+--] [---[ MCR ]
|
|%M0001                                                              %M0002
+--] [--+---------------------------------------------------------( )--
|       |
|       |                                                          FOCO_1
|%M0002 |%M0004                                                   %Q0001
+--] [--+--]/[-----------------------------------------------------( )--
|
%M0001  %M0002                                                   %M0003
|--]/[-----] [--+-----------------------------------------------( )--
|              |
|%M0003        |
|--] [----------+
|
|%M0001  %M0003                                                    %M0004
|--] [-----] [--+-----------------------------------------------( )--
|              |                                                   FOCO_2
|%M0004        |%M0006                                            %Q0002
|--] [----------+--]/[-----------------------------------------( )--
|
|%M0001  %M0004                                                   %M0005
|--]/[-----] [--+-----------------------------------------------( )--
|              |
|%M0005        |
|--] [----------+
|
| %M0001  %M0005                                                   %M0006
|--] [-----] [--+-----------------------------------------------( )--
|              |                                                   FOCO_3
|%M0006        |%M0008                                            %Q0003
|--] [----------+--]/[-----------------------------------------( )—
|
|%M0001  %M0006                                                    %M0007
|--]/[-----] [--+-----------------------------------------------( )--
|              |
|%M0007        |
|--] [----------+
|
|%M0001  %M0007                                                    %M0008
|--] [-----] [--+-----------------------------------------------( )--
|              |
|%M0008        |                                                   %Q0004
|--] [----------+-----------------------------------------------( )--
```

```
|
|%M0001   %M0008                                                    %M0009
|--]/[-----] [--+-----------------------------------------------------( )--
|               |
|%M0009         |
|--] [----------+
|
|%M0001   %M0009                                                    %M0010
|--] [-----] [-----------------------------------------------------( )--
|
|       C1
[    ENDMCR     ]
|
|[       END OF PROGRAM LOGIC       ]
|
```

A continuación se presenta la forma de cablear el presente ejercicio:



Ejercicio 4.11.2

EJERCICIO 4.11.3

Diseñe el circuito de control de un contador de Johnson de cuatro bits utilizando la función master control.

```
            V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     -------------------------------
          %I0001                     BOTON_1
          %Q0001                     FOCO_1
          %Q0002                     FOCO_2
          %Q0003                     FOCO_3
          %Q0004                     FOCO_4


|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|
%I0001                                                          %M0001
+--] [-----------------------------------------------------------( )--
|
| << RUNG 5  STEP #0003 >>
|
|%M0017    C2
+--] [---[ MCR ]
|
* << RUNG 6  STEP #0005 >>
*
*%M0001                                                          %M0002
*--] [--+-------------------------------------------------------( )--
*       |
*       |                                                       FOCO_1
*%M0002 |%M0010                                                 %Q0001
*--] [--+--]/[-----------------------------------------------------( )--
*
* << RUNG 7  STEP #0012 >>
*
*%M0001  %M0002                                                  %M0003
*--]/[-----] [--+-------------------------------------------------( )--
*               |
*%M0003         |
*--] [----------+
*
* << RUNG 8  STEP #0016 >>
*
*%M0001   %M0003                                                 %M0004
*--] [-----] [--+-------------------------------------------------( )--
*               |
*               |                                               FOCO_2
*%M0004         |%M0012                                         %Q0002
*--] [----------+--]/[-----------------------------------------( )--
*
* << RUNG 9  STEP #0024 >>
```

```
*
*%M0001  %M0004                                                    %M0005
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0005         |
*--] [----------+
*
* << RUNG 10  STEP #0028 >>
*%M0001  %M0005                                                    %M0006
*--] [-----] [--+-----------------------------------------------------( )--
*               |                                                  FOCO_3
*%M0006         |%M0014                                            %Q0003
*--] [----------+--]/[-----------------------------------------------( )--
*
* << RUNG 11  STEP #0036 >>
*%M0001  %M0006                                                    %M0007
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0007         |
*--] [----------+
*
* << RUNG 12  STEP #0040 >>
*%M0001  %M0007                                                    %M0008
*--] [-----] [--+-----------------------------------------------------( )--
*               |                                                  FOCO_4
*%M0008         |%M0016                                            %Q0004
*--] [----------+--]/[-----------------------------------------------( )--
*
* << RUNG 13  STEP #0048 >>
*%M0001  %M0008                                                    %M0009
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0009         |
*--] [----------+
* << RUNG 14  STEP #0052 >>
*%M0001  %M0009                                                    %M0010
*--] [-----] [--+-----------------------------------------------------( )--
*               |
*%M0010         |
*--] [----------+
*
* << RUNG 15  STEP #0056 >>
*%M0001  %M0010                                                    %M0011
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0011         |
*--] [----------+
*
* << RUNG 16  STEP #0060 >>
*%M0001  %M0011                                                    %M0012
*--] [-----] [--+-----------------------------------------------------( )--
*               |
*%M0012         |
*--] [----------+
*
* << RUNG 17  STEP #0064 >>
*
```

```
*%M0001  %M0012                                                    %M0013
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0013         |
*--] [----------+
*
* << RUNG 18  STEP #0068 >>
*%M0001  %M0013                                                    %M0014
*--] [-----] [--+-----------------------------------------------------( )--
*               |
*%M0014         |
*--] [----------+
*
* << RUNG 19  STEP #0072 >>
*%M0001  %M0014                                                    %M0015
*--]/[-----] [--+-----------------------------------------------------( )--
*               |
*%M0015         |
*--] [----------+
*
* << RUNG 20  STEP #0076 >>
*
*%M0001  %M0015                                                     %M0016
*--] [-----] [--+-----------------------------------------------------( )--
*               |
*%M0016         |
*--] [----------+
*
* << RUNG 21  STEP #0080 >>
*%M0001  %M0016                                                    %M0017
*--]/[-----] [-----------------------------------------------------( )--
*
| << RUNG 22  STEP #0083 >>
|
|     C2
+[   ENDMCR    ]
|[       END OF PROGRAM LOGIC       ]
```

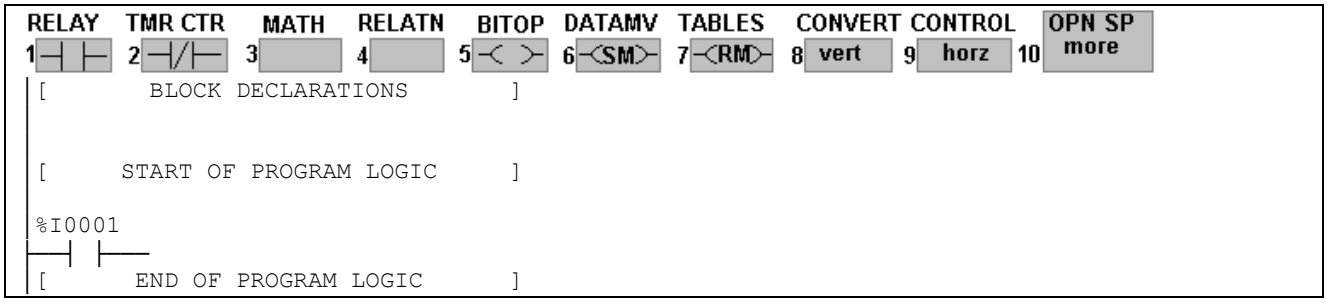
4.12 Programación de circuitos con secuenciadores

Marca GE Fanuc.

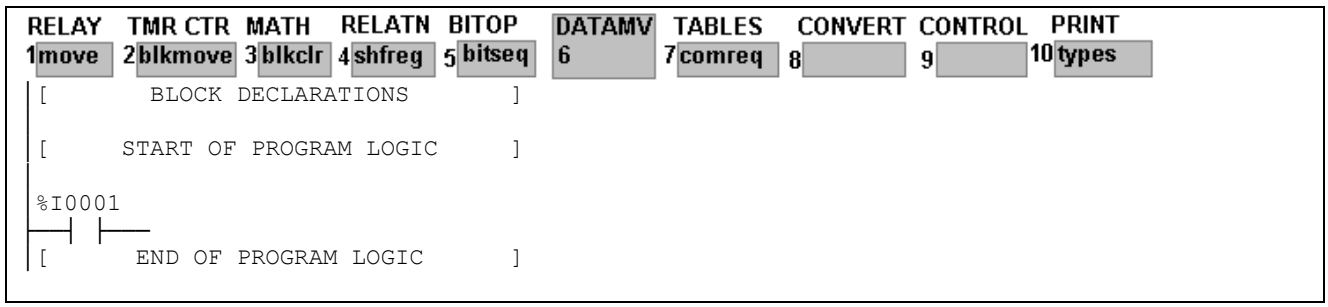
El funcionamiento de los secuenciadores es similar al registro de corrimiento, en los cuales se tiene una entrada de clock, una entrada de dirección y una entrada de reset.

Funcionamiento:

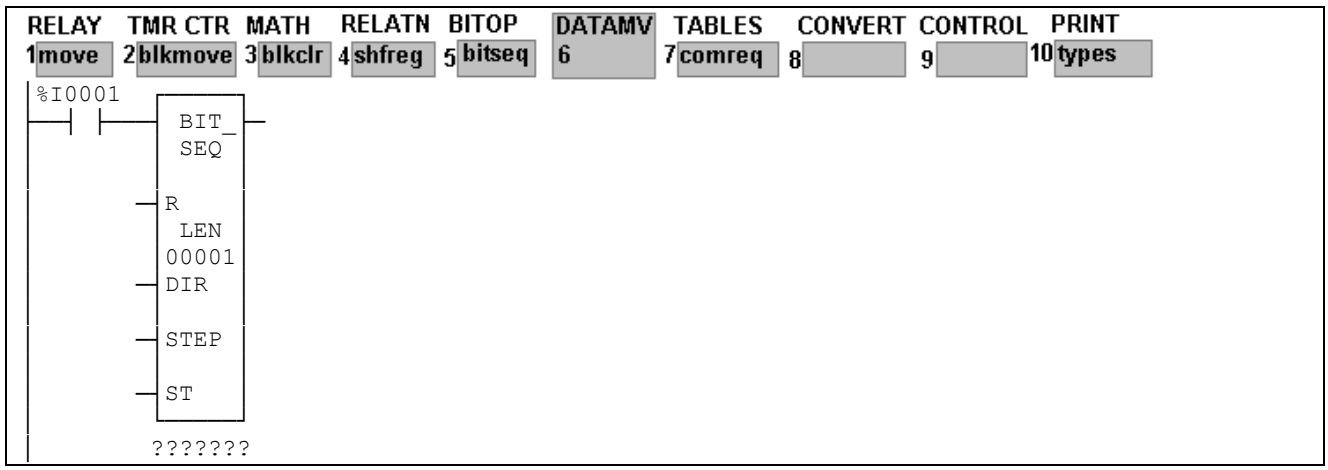
Primeramente colocamos la entrada de clock tal como lo muestra la siguiente figura.



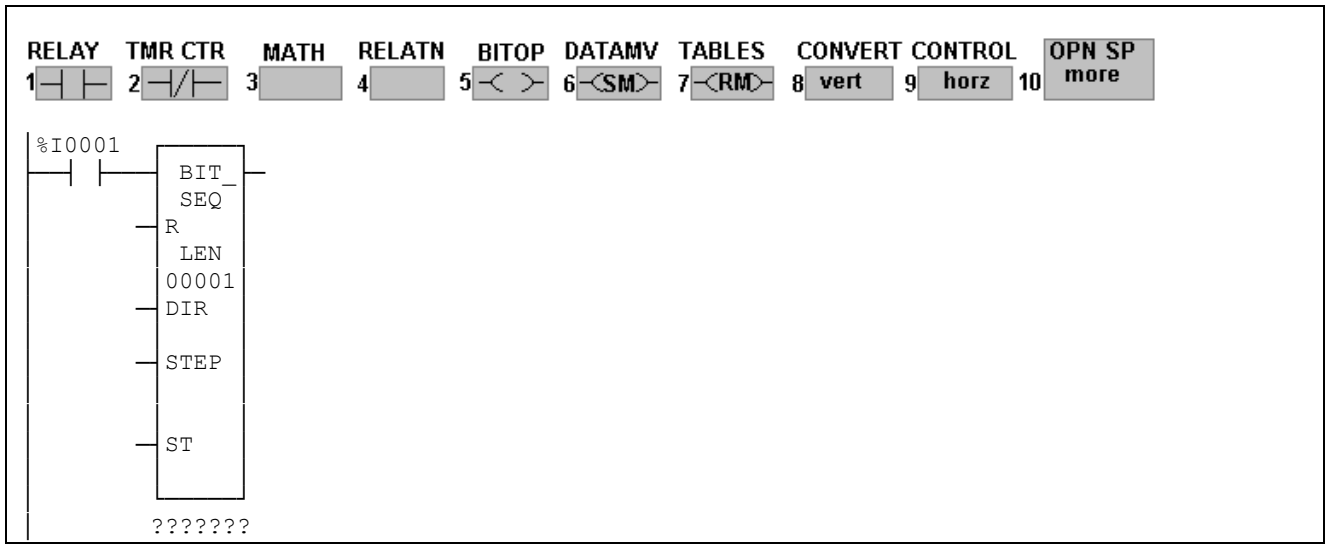
Posteriormente oprimimos las teclas shft + F6 donde se observa un DATA MOVE, y con lo cual nos muestra la siguiente pantalla.



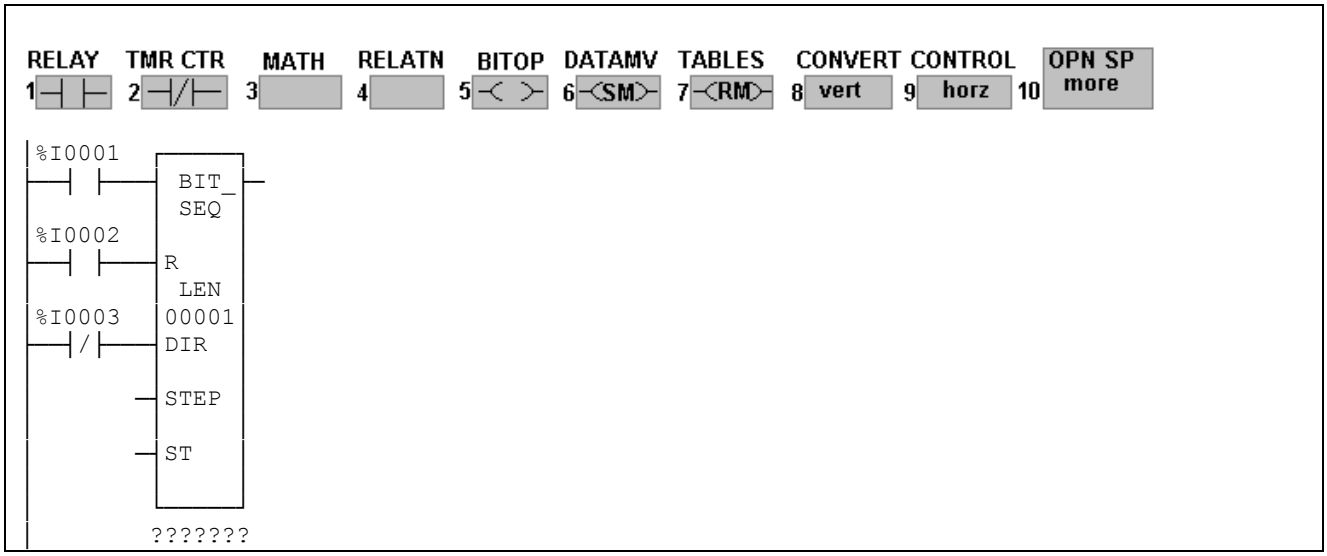
Después seleccionamos F5 (bistec), con lo cual nos muestra la sig. Pantalla,



En seguida oprimimos las teclas shft + F1 con lo cual obtenemos la función relevador para completar nuestro circuito



Por último procedemos a completar la información que pide:



Como podemos observar en la pantalla anterior la línea I1 esta conectada a el clock, la línea I2 esta conectada al Reset y la entrada I3 esta conectada a el direccionamiento, también en este punto es importante hacer notar que con I3 cerrado indica que el direccionamiento es en el sentido de las manecillas del reloj, con el contacto abierto indica el sentido contrario a las manecillas del reloj.

Ahora continuamos con el llenado de la información de este bloque en el cual el STEP indica el puntero del Reset, y deberá ser una constante, esto significa que cuando nosotros reseteamos con I2 este block, la dirección que tomara será la que le indiquemos en el STEP, en este caso colocaremos el valor de 1 para que al resetearlo la dirección del secuenciador se coloque en la dirección de 1.

La entrada ST nos indica la variable que queremos mover y pueden ser la Q, M, T, G.

En la parte inferior deberemos colocar el registro monitor, esto es, el lugar donde colocaremos dicha información, en este caso se requieren de cuatro registros (localidad de memoria). y se coloca R1, con lo cual quedan utilizados R1, R2, R3, y R4. También este registro nos sirve para visualizar la posición del secuenciador.

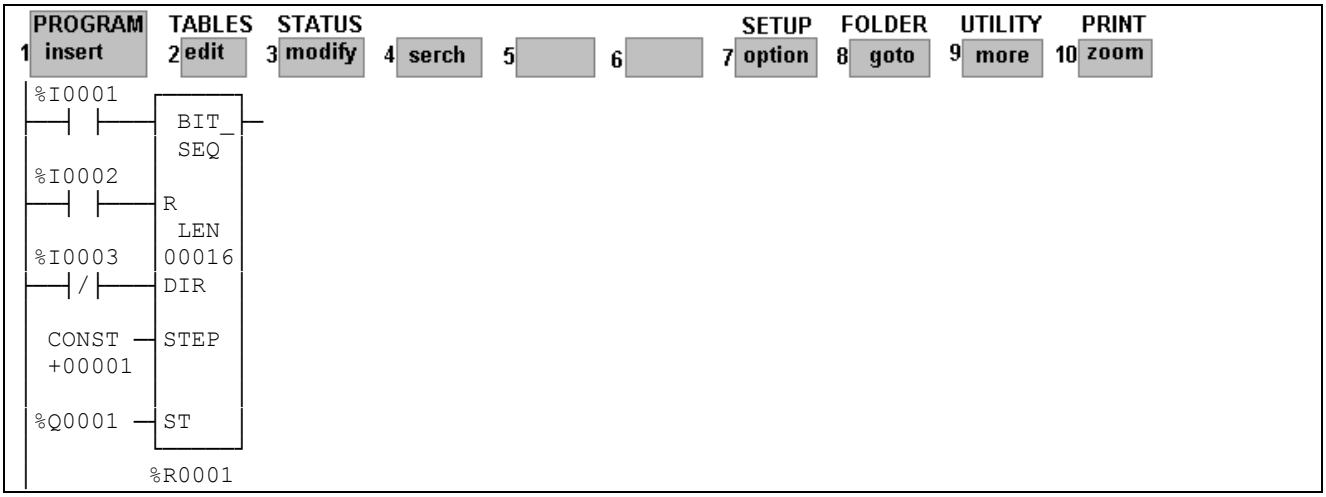
Finalmente, en la parte del centro (LEN) se coloca la longitud del secuenciador por ejemplo si queremos mover un dato introducido por el clock 16 direcciones a la derecha, se coloca en (LEN) 16.

NOTA:

1.- La longitud máxima que le podemos dar a este secuenciador es de 2 a la 16 = 32,768.

2.- La variable a mover representada por ST se toma por palabras completas de 8 bits, de tal manera que si en un secuenciador se toman tres direcciones de la variable a mover por ejemplo las salidas de Q1, para el siguiente secuenciador se deberá iniciar con la dirección de 9, ya que las primeras 8 direcciones fueron utilizadas por el primer secuenciador.

3.- En este PLC de Ge Fanuc la dirección cero no existe.



Funcionamiento:

Si colocamos un dato en Q1, esto es, si a Q1 le damos el valor de uno, este dato se puede mover por cada pulso que le demos al clock, este dato se moverá una dirección hacia la derecha por cada pulso del clock (tomando en cuenta que I3 se encuentra cerrado y está conectado a la entrada de (DIR). Cuando le damos un pulso de reset el valor en Q1 se moverá a la dirección del puntero de reset en este caso Q1 se colocara en la dirección de 1. Si la entrada de dirección se abre esto es, si la entrada I3 se coloca abierta la dirección de movimiento de datos será en sentido contrario a las manecillas de reloj.

EJERCICIO 4.12.1

Diseñe el circuito de control en el que al oprimir un botón se prenda un foco 1 y un segundo después se prenda el foco 2 y se apague el foco 1, un segundo después se apaga el foco 2 y prende el foco 3 y así sucesivamente con 16 focos, de tal manera que cuando se prende el foco 16 un segundo después se apaga el foco 16 y se prende el foco 1 etc, como se puede observar éste es un circuito contador de anillo de 16 bits.

```
|[  START OF LD  PROGRAM  SEC_1   ]
|[     VARIABLE DECLARATIONS      ]
|
|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|
|PB_INI  PB_PARO
|%I0001  %I0002                                                  %M0100
+--] [--+--]/[-----------------------------------------------------( )--
|       |
|%M0100 |
+--] [--+
```

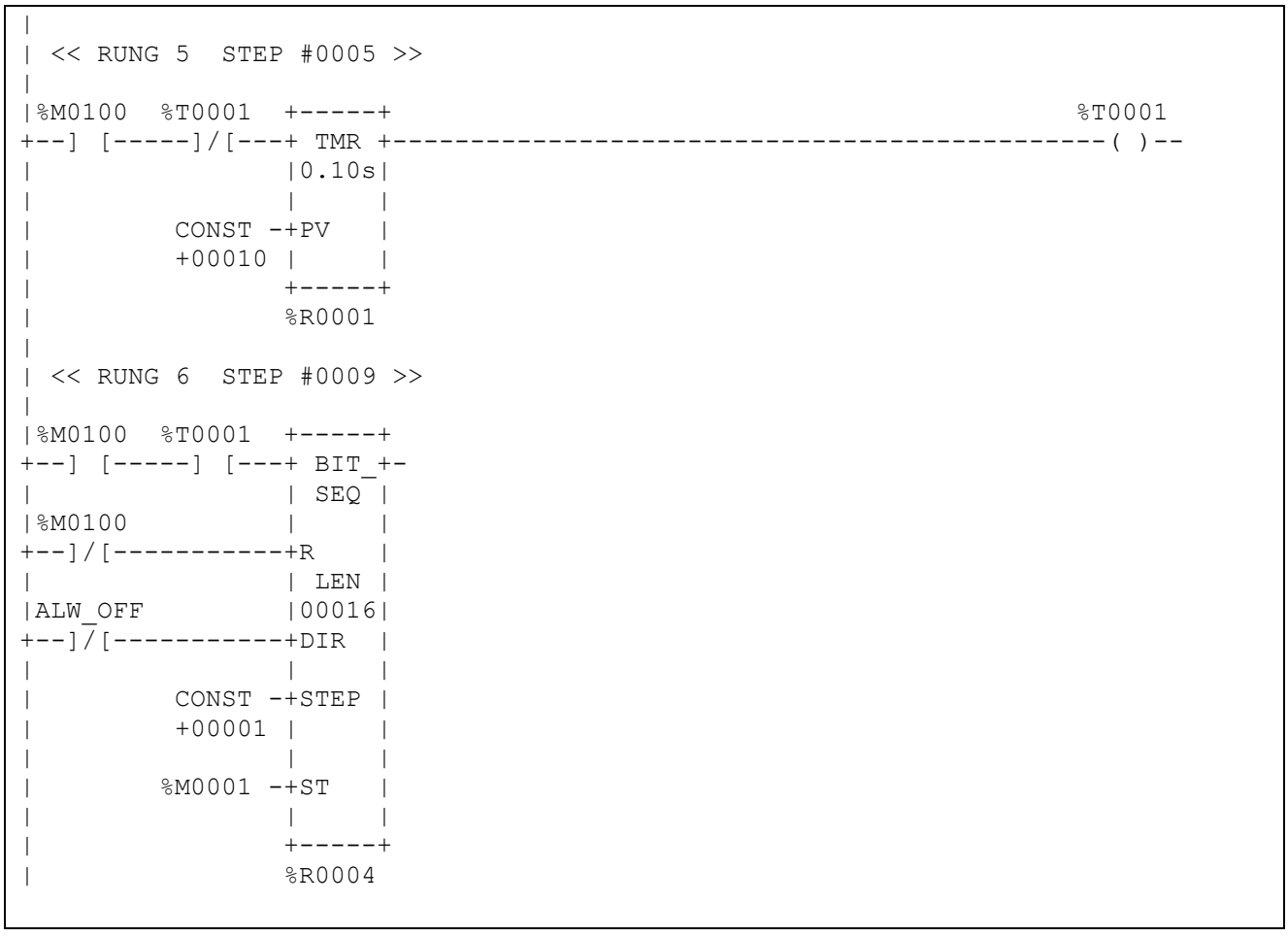
OBSERVACIONES:

En la línea anterior (RUNG 4) se tiene como I1 un botón de inicio normalmente abierto el cual nos sirve para dar inicio de ciclo, con I2 tenemos un botón de paro normalmente abierto, de tal manera que cuando se active abrirá el contacto, tumbando la memoria. Al oprimir I1 se energiza la

bobina M100 y se sostiene, también este contacto de M100 esta conectado al reset del secuenciador en la (RUNG) 6, de tal manera que mientras no se de inicio ciclo el secuenciador esta reseteado, al dar inicio de ciclo se abre el contacto de M100 que esta en el reset del secuenciador permitiendo que esté en condiciones de realizar su función.

También este contacto de M100 esta en serie con el timer de tal manera que habilita al timer y este a su vez empieza a correr el tiempo cuando completa el tiempo de 1 seg. se resetea dando un pulso al secuenciador el cual a su vez activa la salida de M2 y desactiva la salida m1, así sucesivamente.

Observe como estando el secuenciador reseteado, éste toma la dirección de M1 en 1 de tal manera que al dar inicio de ciclo M1 esta cerrado y por lo tanto energiza la salida Q1 que se encuentra en la línea 7 (RUNG) 7.



NOTA: En la dirección del secuenciador se coloca un contacto cerrado de S8 el cual es un contacto sin bobina que siempre esta cerrado para que el sentido del secuenciador siempre sea en el sentido de las manecillas del reloj.

En la siguiente línea, se observa como al dar inicio de ciclo se cierra el contacto de M100, y como el secuenciador esta reseteado, se encuentra su salida de M1 cerrado con lo cual se energiza la salida Q1, un segundo después el secuenciador pasa a su segundo paso cerrando su contacto de M2 energizando Q2 y desenergizando Q1 y así sucesivamente.

```
| << RUNG 7  STEP #0014 >>
|%M0100  %M0001                                                    %Q0001
+--] [--+--] [-----------------------------------------------------( )--
|       |
|       |%M0002                                                    %Q0002
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0003                                                    %Q0003
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0004                                                    %Q0004
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0005                                                    %Q0005
|       +--] [-----------------------------------------------------( )--
|       |
|        %M0006                                                    %Q0006
|       +--] [-----------------------------------------------------( )--

|       |%M0007                                                    %Q0007
|       +--] [-----------------------------------------------------( )--
|
|       |%M0008                                                    %Q0008
|       +--] [-----------------------------------------------------( )--
| << RUNG 8  STEP #0045 >>
|%M0100  %M0009                                                    %Q0009
+--] [--+--] [-----------------------------------------------------( )--
|       |
|       |%M0010                                                    %Q0010
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0011                                                    %Q0011
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0012                                                    %Q0012
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0013                                                    %Q0013
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0014                                                    %Q0014
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0015                                                    %Q0015
|       +--] [-----------------------------------------------------( )--
|       |
|       |%M0016                                                    %Q0016
|       +--] [-----------------------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
```

EJERCICIO 4.12.2

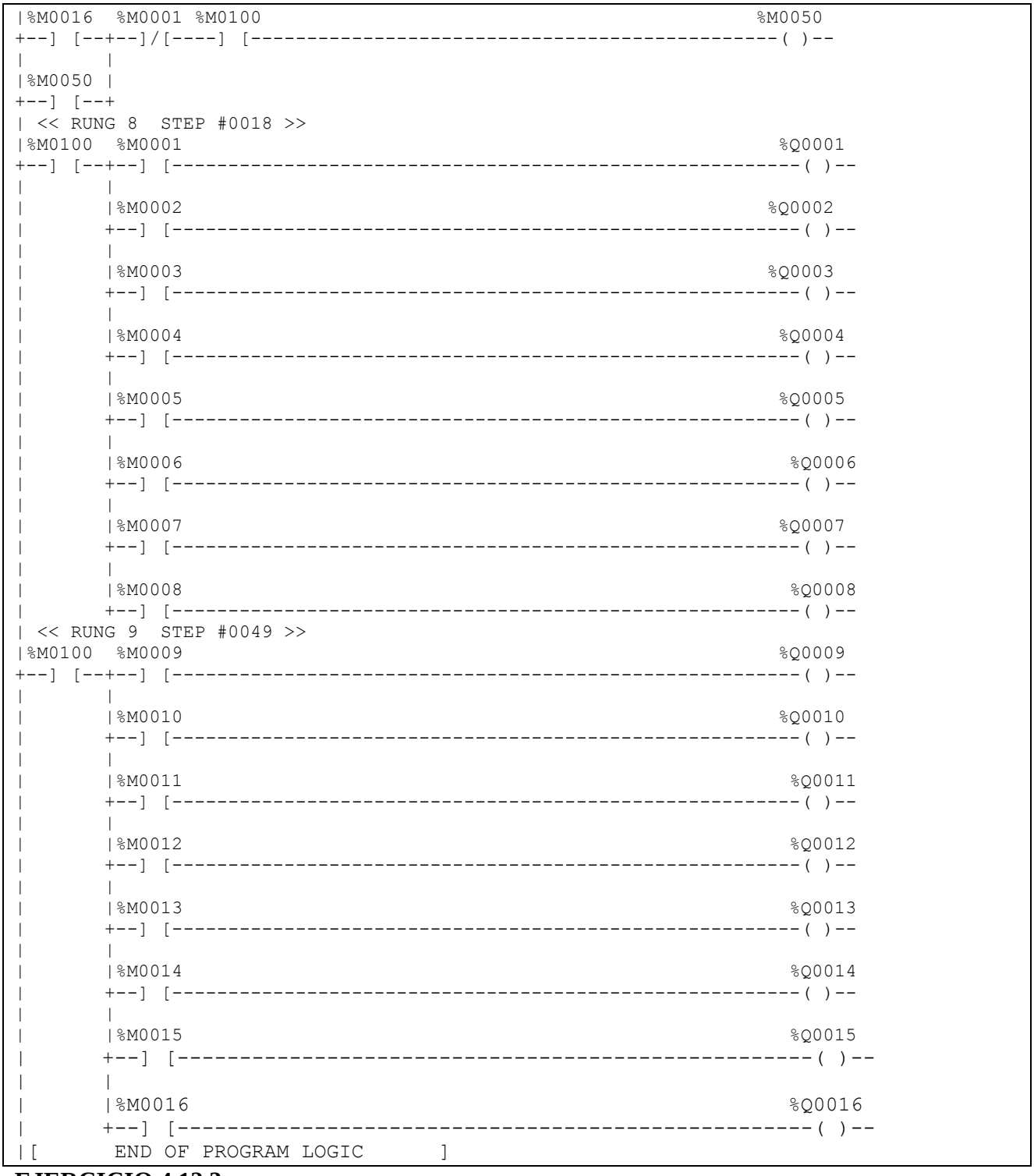
Diseñe el circuito de control para controlar un foco en control de péndulo de 16 bits esto es, al dar inicio de ciclo se activa el bit 1 con el siguiente impulso del clock se desactiva el bit uno y prende el dos hasta llegar al bit 16 , el siguiente paso es regresarse y prender el bit 15 y así sucesivamente hasta llegar al bit uno y luego regresarse y prender el bit número dos. Como podemos observar éste es un contador tipo péndulo.

```
|[  START OF LD  PROGRAM          ]
|
|[     VARIABLE DECLARATIONS      ]
|
|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|
|%I0001  %I0002                                                  %M0100
+--] [--+--]/[-----------------------------------------------------( )--
|       |
|%M0100 |
+--] [--+
|
| << RUNG 5  STEP #0005 >>
|
|%M0100  %T0001  +-----+                                         %T0001
+--] [-----]/[---+ TMR +-----------------------------------------( )--
|                |0.10s|
|                |     |
|          CONST -+PV   |
|          +00010 |     |
|                 +-----+
|                 %R0001
|
| << RUNG 6  STEP #0009 >>
|%M0100  %T0001  +-----+
+--] [-----] [---+ BIT_+-
|                | SEQ |
|%M0100          |     |
+--]/[-----------+R    |
|                | LEN |
|%M0050          |00016|
+--]/[-----------+DIR  |
|                |     |
|          CONST -+STEP |
|          +00001 |    |
|         %M0001 -+ST   |
|                 +-----+
|                 %R0004
| << RUNG 7  STEP #0014 >>
```

OBSERVACIONES:

Observe como el circuito que se presenta para la solución del presente problema es similar al anterior solo que ahora se ha insertado la línea (RUNG 7) en la cual, al entrar M16 se memoriza con M50, la cual a su vez se colocó en la dirección del secuenciador de tal manera que al abrir su

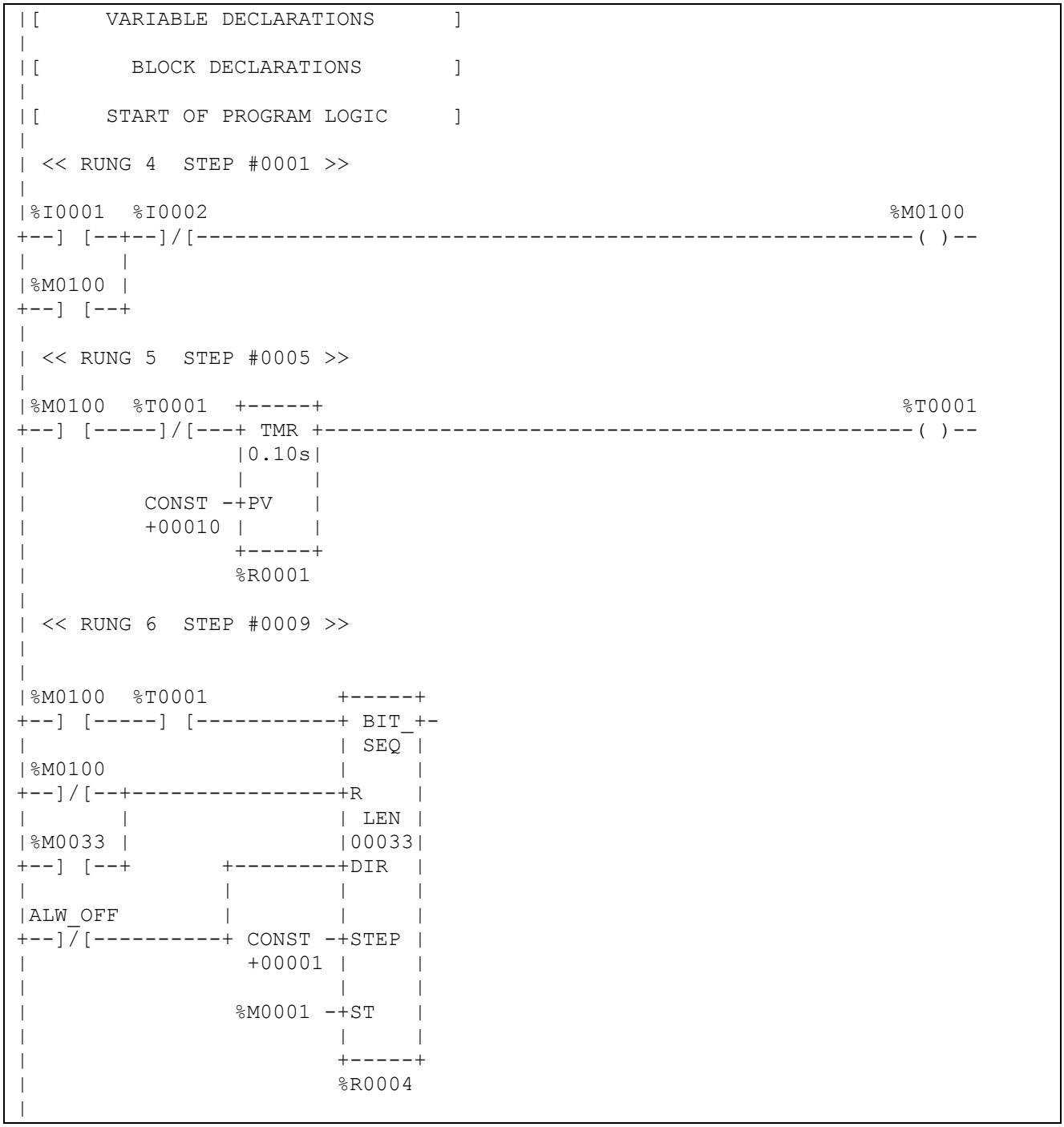
contacto el secuenciador invierta su dirección hasta llegar a M1 la cual tumba la memoria, invirtiendo su dirección.



EJERCICIO 4.12.3:

Diseñe el circuito de control para controlar 16 salidas que trabaje como un contador tipo Johnson, en el cual se desea que con el primer pulso de clock se active la salida uno, con el siguiente pulso de clock se activa la salida dos y la salida uno se queda activada y así sucesivamente

hasta la salida 16, con el siguiente pulso del clock se desactiva el bit 16 y así sucesivamente hasta que se desactivan los 16 bits, con el siguiente pulso de clock inicia un nuevo ciclo. Como podemos observar éste es un contador tipo Johnson.



Observe como ahora para ésta solución, la longitud del secuenciador se prolongó a 33 y la dirección se cambio por un contacto cerrado de S8 siempre cerrado de tal manera que la dirección del siempre será en sentido de las manecillas del reloj, observe también que el secuenciador se

resetea cundo pasa a la dirección de 33, inclusive como se puede observar como la longitud es de 33, no hay necesidad de resetearlo con M33.

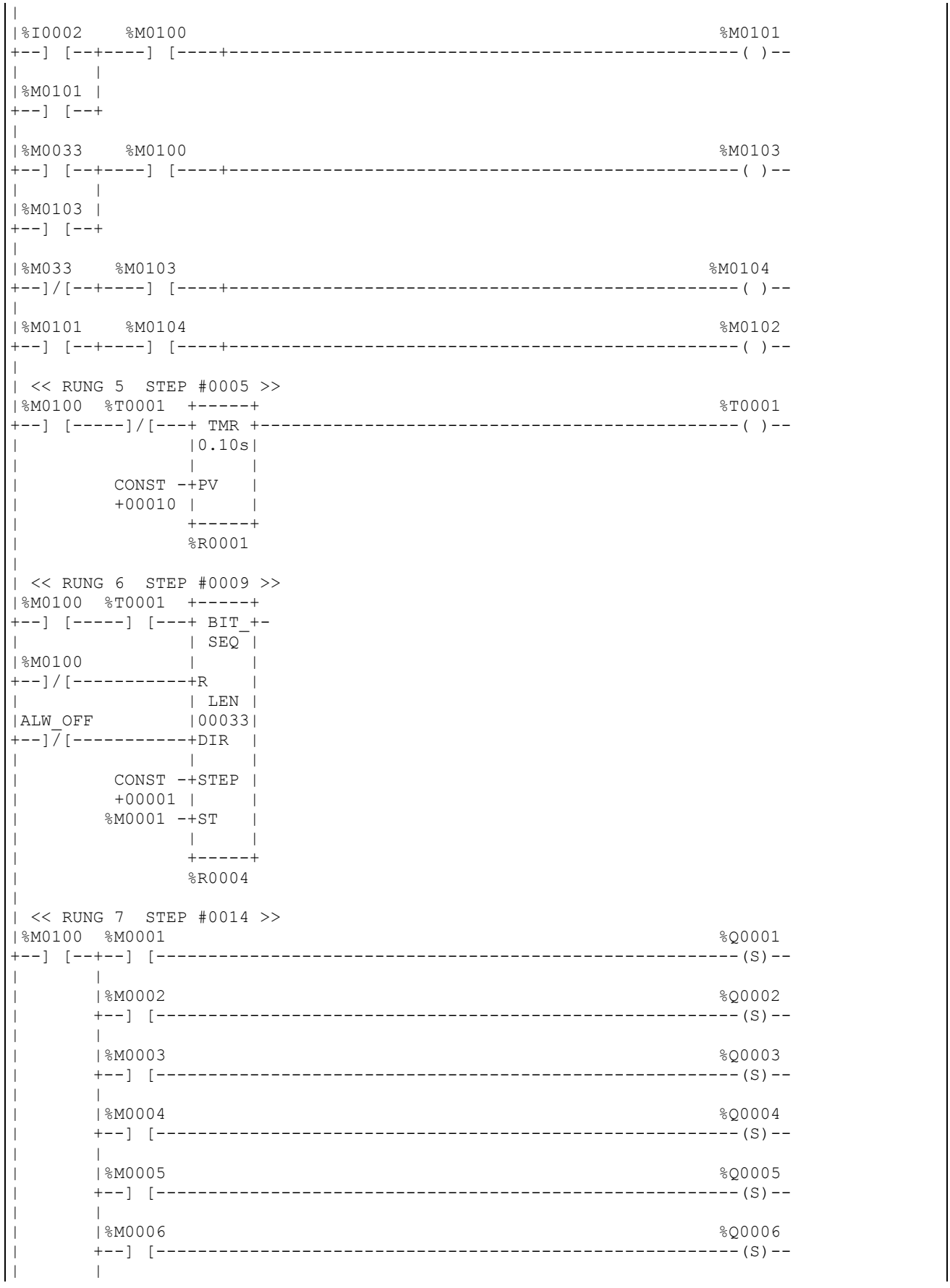
```
| << RUNG 7  STEP #0015 >>
|%M0100  %M0001                                                %Q0001
+--] [--+--] [-------------------------------------------------------(S)--
|       |
|       |%M0002                                                %Q0002
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0003                                                %Q0003
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0004                                                %Q0004
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0005                                                 %Q0005
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0006                                                %Q0006
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0007                                                %Q0007
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0008                                                %Q0008
|       +--] [-------------------------------------------------------(S)--
| << RUNG 8  STEP #0046 >>
|%M0100  %M0009                                                %Q0009
+--] [--+--] [-------------------------------------------------------(S)--
|       |
|       |%M0010                                                %Q0010
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0011                                                 %Q0011
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0012                                                %Q0012
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0013                                                %Q0013
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0014                                                %Q0014
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0015                                                %Q0015
|       +--] [-------------------------------------------------------(S)--
|       |
|       |%M0016                                                %Q0016
|       +--] [-------------------------------------------------------( )--
|
| << RUNG 9  STEP #0077 >>
|%M0100  %M0017                                                %Q0016
+--] [--+--] [-------------------------------------------------------(R)--
|       |
|       |%M0018                                                %Q0015
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0019                                                %Q0014
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0020                                                %Q0013
|       +--] [-------------------------------------------------------(R)--
```

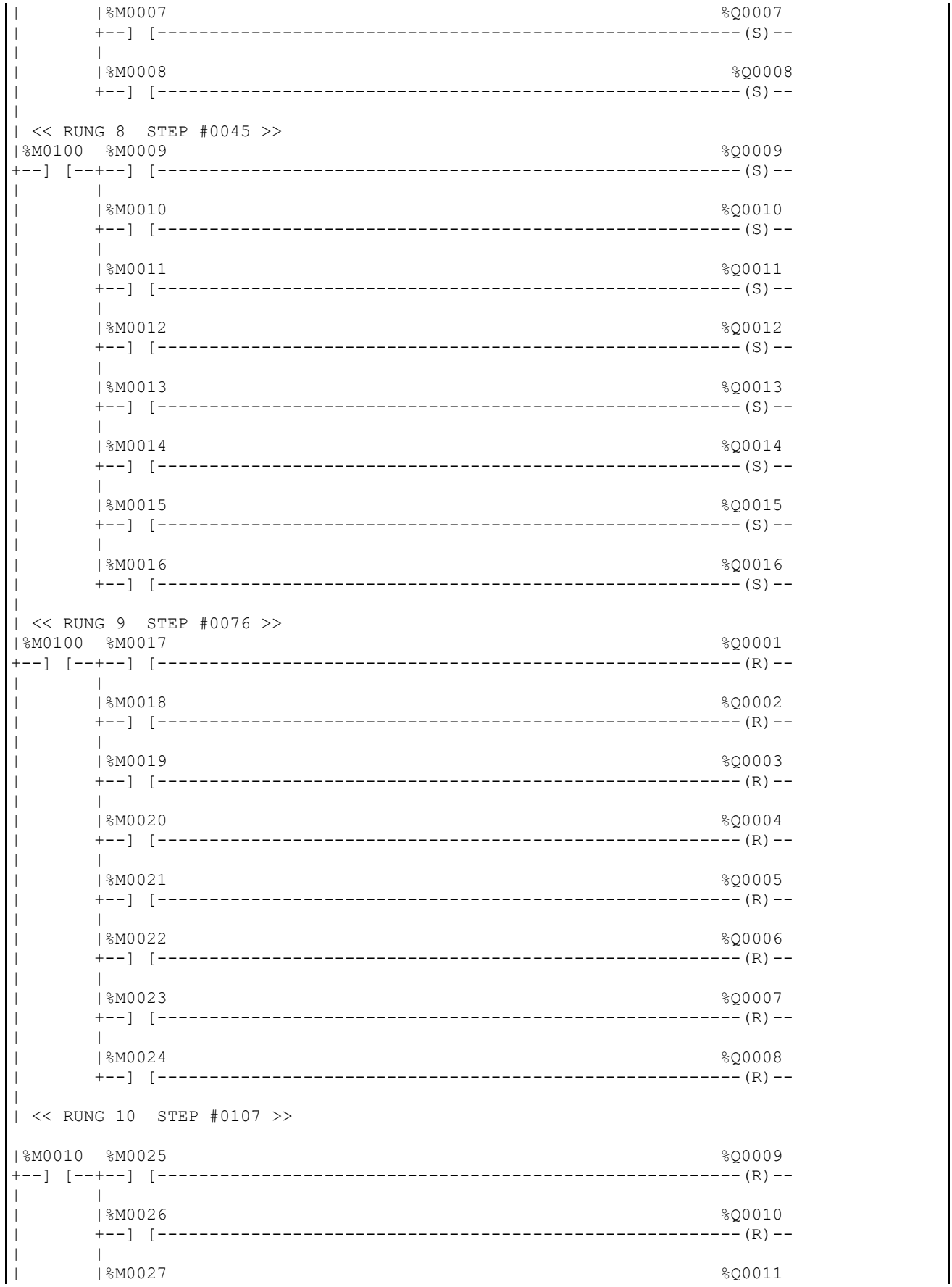
```
|       |
|       |%M0021                                                    %Q0012
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0022                                                    %Q0011
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0023                                                    %Q0010
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0024                                                    %Q0009
|       +--] [-----------------------------------------------------(R)--
|
| << RUNG 10  STEP #0108 >>
|%M0010  %M0025                                                    %Q0008
+--] [--+--] [-----------------------------------------------------(R)--
|       |
|       |%M0026                                                    %Q0007
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0027                                                    %Q0006
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0028                                                    %Q0005
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0029                                                    %Q0004
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0030                                                    %Q0003
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0031                                                    %Q0002
|       +--] [-----------------------------------------------------(R)--
|       |
|       |%M0032                                                    %Q0001
|       +--] [-----------------------------------------------------(R)--
|[      END OF PROGRAM LOGIC      ]
```

EJERCICIO 4.12.4

Diseñe el circuito de control del ejercicio número tres, sólo que ahora se desea que el contador trabaje a la inversa, esto es, al energizarse el bit 16 y con el siguiente pulso del clock ahora se desactiva el bit 1 hasta desactivarse los 16, iniciando un nuevo ciclo con el siguiente pulso. Como podrá observar este es un contador Johnson invertido. Como parte adicional a este circuito se desea colocar un botón de paro de tal manera que al dar un impulso de paro en cualquier instante del recorrido del secuenciador, al terminar el ciclo se pare y que quede en condiciones de iniciar de nuevo.

```
|[  START OF LD  PROGRAM   GEF    |
|[     VARIABLE DECLARATIONS      ]
|
|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|%I0001        %M0102                                      %M0100
+--] [--+-----+--]/[--+-------------------------------------------( )--
|       |
|%M0100 |
+--] [--+
```


|%I0002 %M0100 %M0101
+--] [--+----] [----+----( )--
|%M0101 |
+--] [--+
|%M0033 %M0100 %M0103
+--] [--+----] [----+----( )--
|%M0103 |
+--] [--+
|%M033 %M0103 %M0104
+--]/[--+----] [----+----( )--
|%M0101 %M0104 %M0102
+--] [--+----] [----+----( )--
| << RUNG 5 STEP #0005 >>
|%M0100 %T0001 +-----+ %T0001
+--] [-----]/[---+ TMR +----( )--
|0.10s|
CONST -+PV |
+00010 |
+-----+
%R0001
| << RUNG 6 STEP #0009 >>
|%M0100 %T0001 +-----+
+--] [-----] [---+ BIT_+-
| SEQ |
|%M0100 |
+--]/[-----------+R |
| LEN |
|ALW_OFF |00033|
+--]/[-----------+DIR |
CONST -+STEP |
+00001 |
%M0001 -+ST |
+-----+
%R0004
| << RUNG 7 STEP #0014 >>
|%M0100 %M0001 %Q0001
+--] [--+--] [----(S)--
|%M0002 %Q0002
+--] [----(S)--
|%M0003 %Q0003
+--] [----(S)--
|%M0004 %Q0004
+--] [----(S)--
|%M0005 %Q0005
+--] [----(S)--
|%M0006 %Q0006
+--] [----(S)--


|%M0007 %Q0007
+--] [----------(S)--
|%M0008 %Q0008
+--] [----------(S)--
<< RUNG 8 STEP #0045 >>
|%M0100 %M0009 %Q0009
+--] [--+--] [----------(S)--
|%M0010 %Q0010
+--] [----------(S)--
|%M0011 %Q0011
+--] [----------(S)--
|%M0012 %Q0012
+--] [----------(S)--
|%M0013 %Q0013
+--] [----------(S)--
|%M0014 %Q0014
+--] [----------(S)--
|%M0015 %Q0015
+--] [----------(S)--
|%M0016 %Q0016
+--] [----------(S)--
<< RUNG 9 STEP #0076 >>
|%M0100 %M0017 %Q0001
+--] [--+--] [----------(R)--
|%M0018 %Q0002
+--] [----------(R)--
|%M0019 %Q0003
+--] [----------(R)--
|%M0020 %Q0004
+--] [----------(R)--
|%M0021 %Q0005
+--] [----------(R)--
|%M0022 %Q0006
+--] [----------(R)--
|%M0023 %Q0007
+--] [----------(R)--
|%M0024 %Q0008
+--] [----------(R)--
<< RUNG 10 STEP #0107 >>
|%M0010 %M0025 %Q0009
+--] [--+--] [----------(R)--
|%M0026 %Q0010
+--] [----------(R)--
|%M0027 %Q0011

```
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0028                                                    %Q0012
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0029                                                    %Q0013
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0030                                                    %Q0014
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0031                                                    %Q0015
|       +--] [-------------------------------------------------------(R)--
|       |
|       |%M0032                                                    %Q0016
|       +--] [-------------------------------------------------------(R)--
|
|[      END OF PROGRAM LOGIC      ]
|
```

4.13 Programación de circuitos con subrutinas.

¿Por qué son importantes las subrutinas?

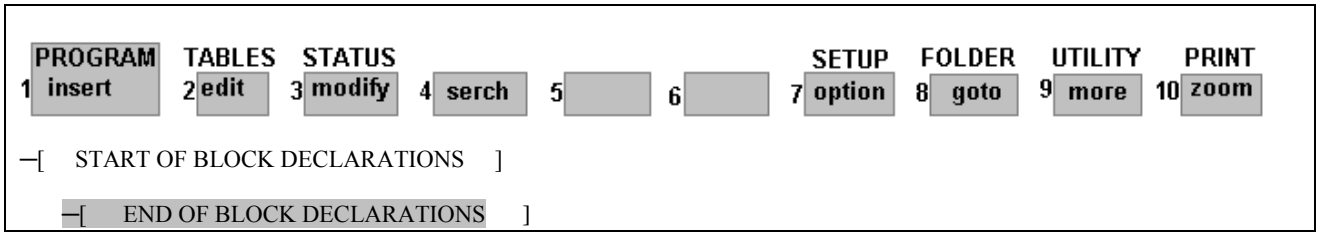
Hasta este momento todos los problemas planteados los hemos resuelto en el (main program) (programa principal). Pero cuando se tiene un problema mucho más extenso en programación conviene dividirlo en varios subprogramas o subrutinas por las cuales se pueda resolver más fácilmente dicho problema. La ventaja de dichas subrutinas ofrece una mayor panorámica de cómo resolverlo, e identificar más fácilmente donde ocurre un problema en caso de presentarse.

Si por ejemplo tenemos una maquina donde se requiere que su funcionamiento se haga manual, en un ciclo o en tres ciclos, conviene hacer un programa con subrutinas, asignando a cada segmento una subrutina, con lo cual podremos encontrar más fácilmente fallas y hacer un programa mucho más entendible y organizado para hacer posteriores modificaciones al mismo..

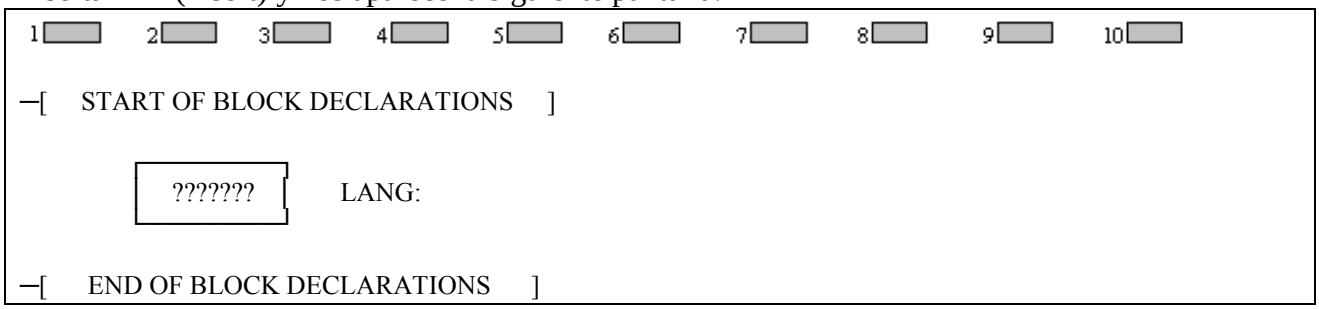
Primeramente para realizar un programa con subrutinas procedemos a declarar los BLOCK DECLARATION y esto se realiza colocando el cursor en (BLOCK DECLARATION)**.**

```
 PROGRAM  TABLES  STATUS                                  SETUP  FOLDER  UTILITY  PRINT
1 insert  2 edit  3 modify  4 serch  5        6        7 option  8 goto  9 more  10 zoom  [
START OF LD  PROGRAM SUB_RUT  ]
|
|[   VARIABLE DECLARATIONS    ]
|
|[    BLOCK DECLARATIONS     ]
|
[   START OF PROGRAM LOGIC    ]
|
|[    END OF PROGRAM LOGIC    ]
```

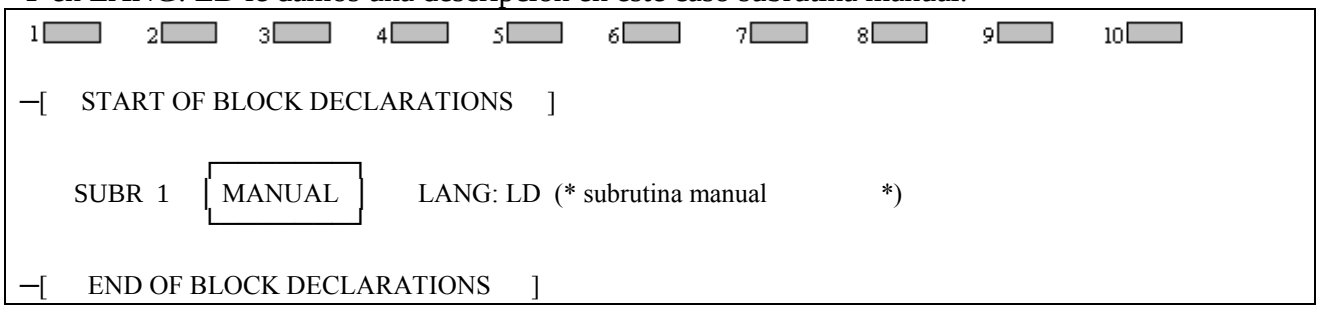
Ahora activamos la tecla **F10** (ZOOM) y nos presenta la siguiente pantalla



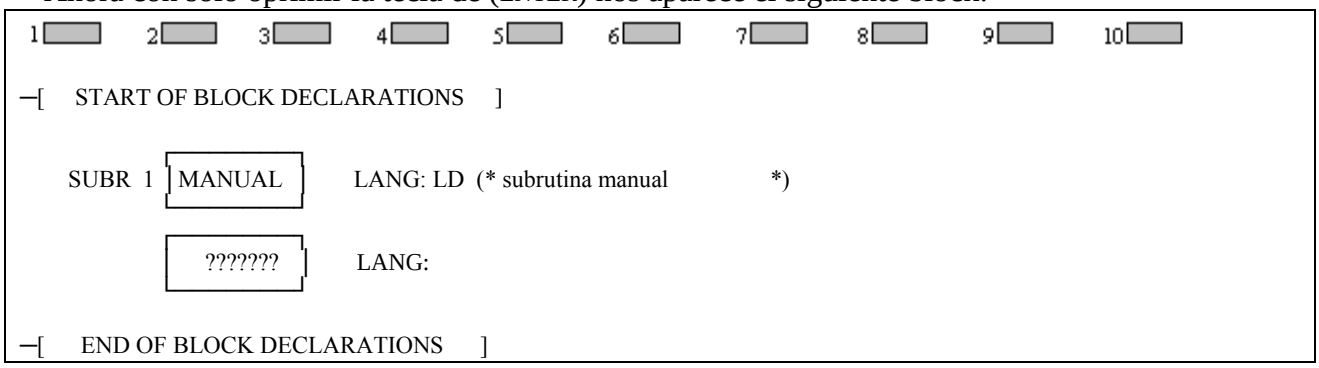
Ahora procedemos a colocar el cursor en (END OF BLOCK DECLARATIONS) y oprimimos la tecla insertar **F1** (insert) y nos aparece la siguiente pantalla.



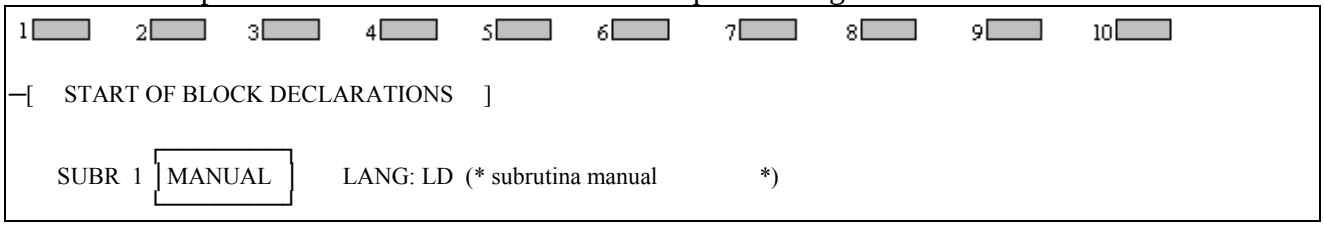
Ahora procedemos a darle un nombre a este block, en este caso le ponemos manual.
Y en LANG: LD le damos una descripción en este caso subrutina manual.



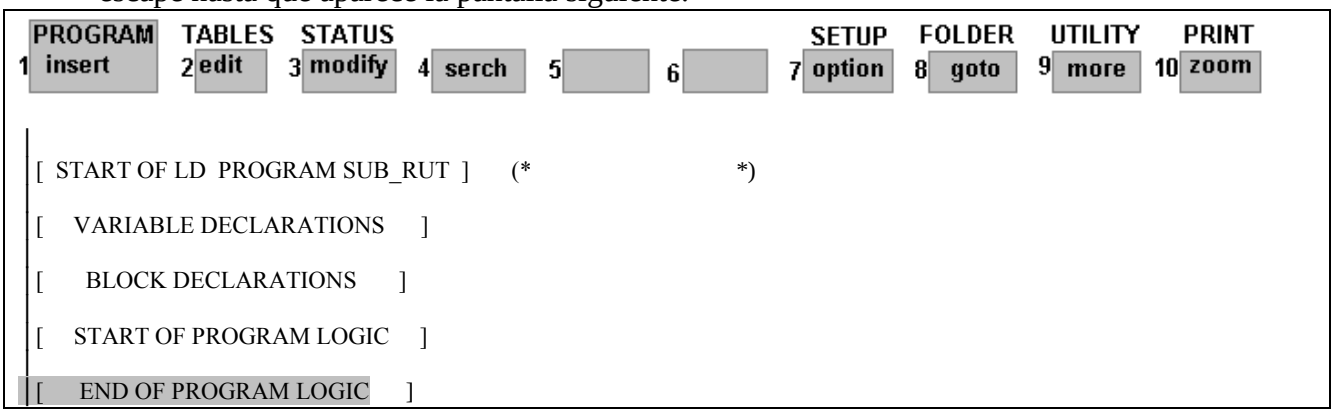
Ahora con solo oprimir la tecla de (ENTER) nos aparece el siguiente block:



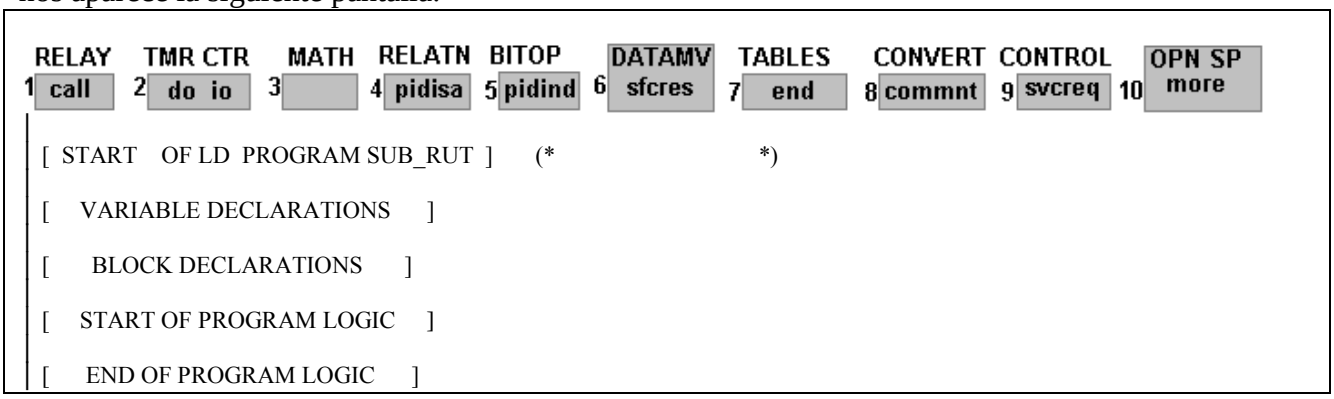
Ahora procedemos a darle nombre a este bloque de la siguiente manera



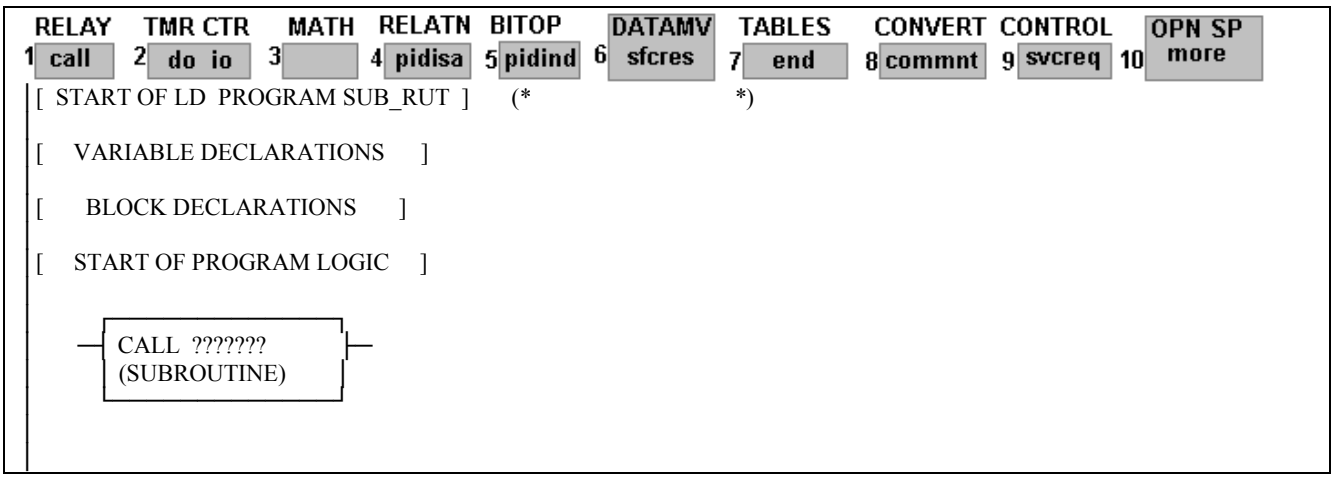
Finalmente, cuando ya están declarados los (BLOCK DECLARATIONS) se oprime la tecla escape hasta que aparece la pantalla siguiente.



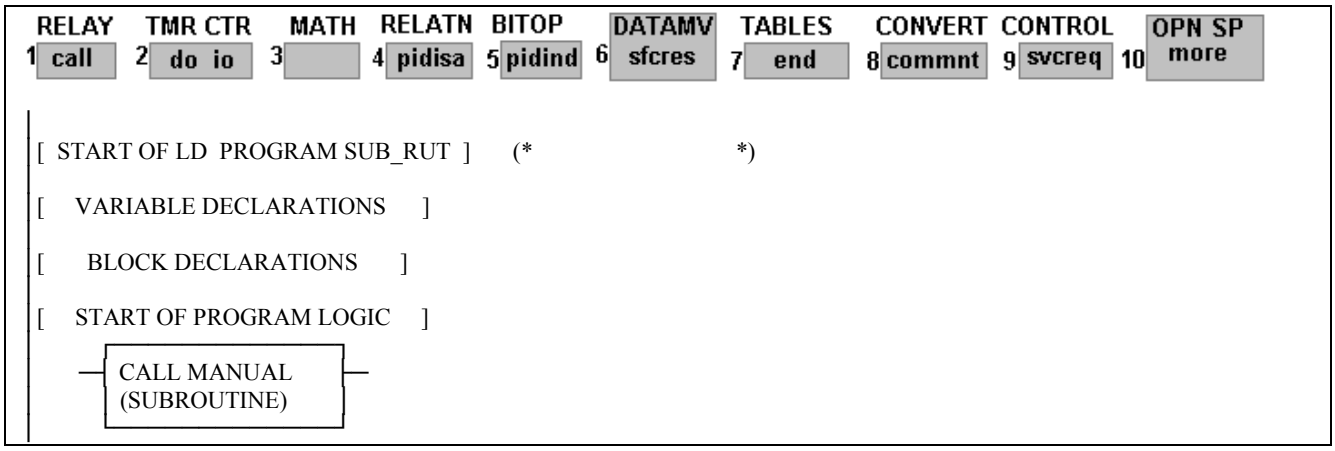
Ahora colocamos el cursor en (END OF PROGRAM LOGIC) y oprimimos la tecla **shift F9** (CONTROL) y nos aparece la siguiente pantalla.



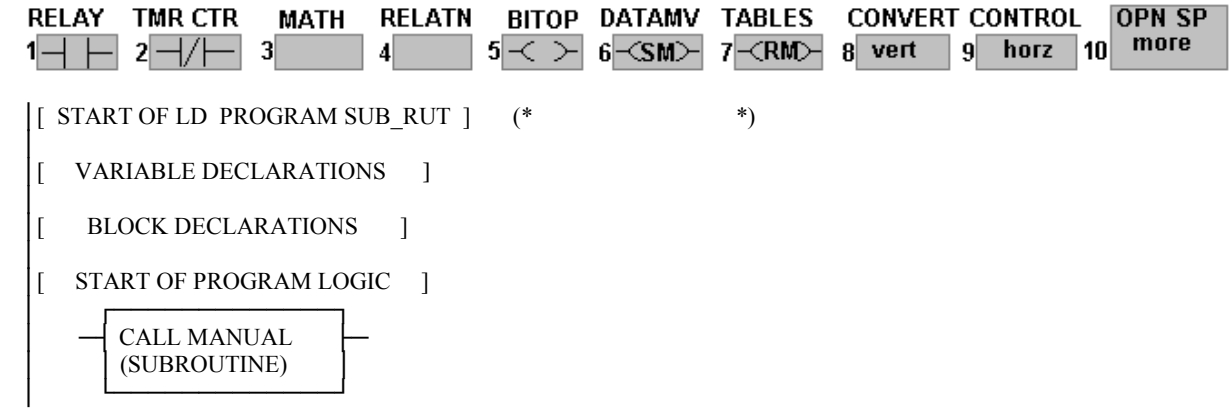
Y ahora oprimimos la tecla **F1** (CALL) y nos aparece la siguiente pantalla.



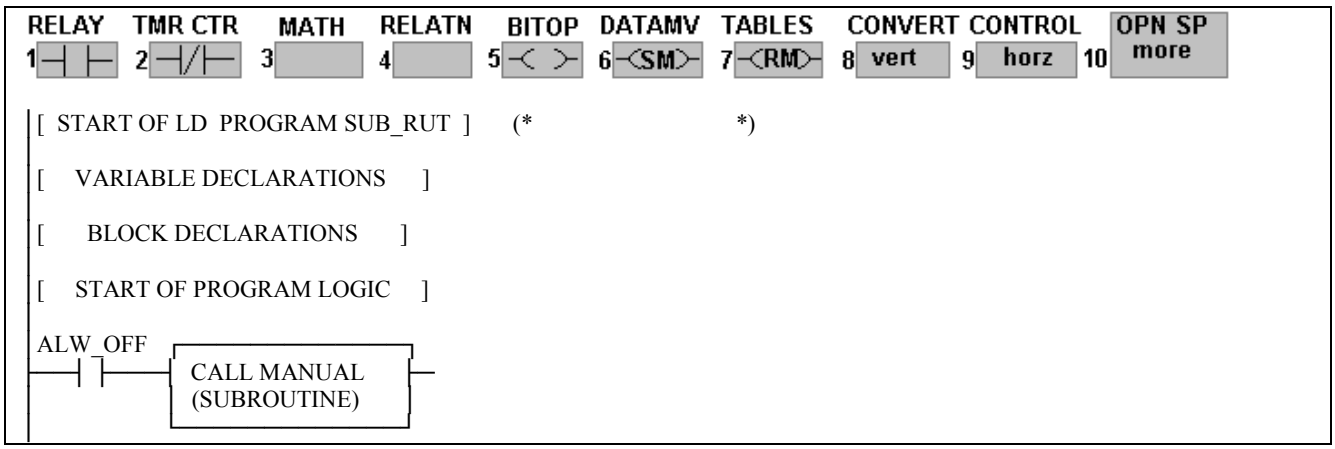
Ahora procedemos a llamar el block manual y nos aparece la siguiente pantalla



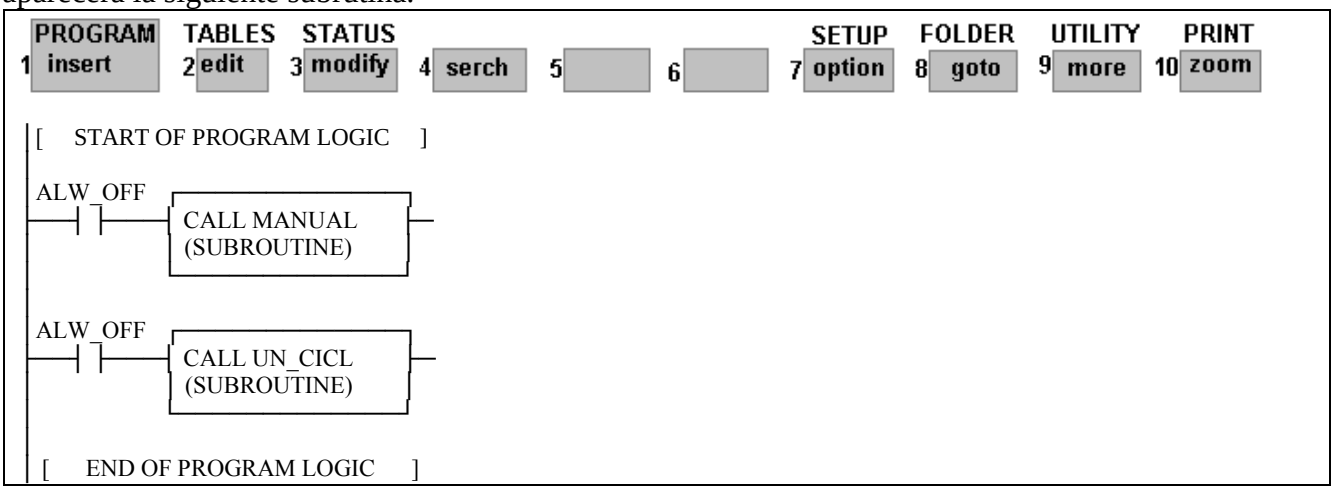
A continuación procedemos a colocar el elemento de control que habilita la subrutina, pero como no aparecen en esta pantalla los contactos de control y como estos pertenecen a los relevadores, oprimimos la tecla shift (F1) y nos aparece la siguiente pantalla.



Ahora si estamos en condiciones de colocar el contacto de control



En seguida procedemos a declarar las siguientes subrutinas, siguiendo los pasos anteriores nos aparecerá la siguiente subrutina.



Con esto terminamos las declaraciones de las subrutinas y procedemos a realizar aplicaciones con los mismos.

4.14 - Programación de circuitos con funciones de relación.

MARCA Ge Fanuc.

Las funciones de relación, son funciones matemáticas de comparación y que en circuitos de control son muy útiles: Su funcionamiento es muy sencillo y no creemos que tengan mayor problema para su aplicación.

A continuación presentamos las diferentes funciones de relación.

En la figura 6.9.1 podemos observar una función EQ (igual) de tal manera que se compara el valor que tiene el registro R1 y lo compara con el valor de R2 si estos valores ***SON IGUALES*** se tendrá una salida en Q1.

```
| [  START OF LD  PROGRAM CURSO2   ]  (*                                *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]

| << RUNG 4  STEP #0001 >>
|ALW_OFF +-----+
|        |     |                                                    %Q0001
+--]/[---+ EQ_ |+----------------------------------------------------( )--
|        | INT ||
|        |     ||
|%R0001 -+I1  Q++
|        |     |
|%R0002 -+I2   |
|        +-----+
```

Figura 6.9.1

La figura 6.8.2 se muestra una función lógica que compara si ***NO SON IGUALES*** R1 y R2, si no son iguales tendremos una salida en Q2.

```
| [   START OF LD PROGRAM CURSO2   ] (*                                    *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 5  STEP #0004 >>
|ALW_OFF +-----+
|        |     |                                                      %Q0002
+--]/[---+ NE_ |+-----------------------------------------------------( )--
|        | INT ||
|        |     ||
|%R0001 -+I1  Q++
|        |     |
|%R0002 -+I2   |
|         +-----+
```

Figura 6.9.2

La figura 6.9.3 nos muestra una función que compara si R1 es ***MAYOR QUE*** R2 sí es así, entonces tenemos una salida en Q3, cualquier otro resultado la salida será falsa.

```
| [   START OF LD PROGRAM CURSO2   ] (*                                    *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]

| << RUNG 6  STEP #0007 >>
|ALW_OFF +-----+
|        |     |                                                      %Q0003
+--]/[---+ GT_ |+-----------------------------------------------------( )--
|        | INT ||
|        |     ||
|%R0001 -+I1  Q++
|        |     |
|%R0002 -+I2   |
|         +-----+
```

Figura 6.9.3

La figura 6.9.4 nos muestra una función que compara si R1 es ***MAYOR O IGUAL QUE*** R2 sí es así, entonces tenemos una salida en Q4, cualquier otro resultado la salida será falsa.

.

```
| [   START OF LD PROGRAM CURSO2   ] (*                                    *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 7 STEP #0010 >>
|ALW_OFF +-----+
|        |     |                                                      %Q0004
+--]/[---+ GE_ |+-----------------------------------------------------( )--
|        | INT ||
|        |     ||
|%R0001 -+I1  Q++
|        |     |
|%R0002 -+I2   |
|         +-----+
```

Figura 6.9.4

La figura 6.8.5 nos muestra una función que compara si R1 es ***MENOR QUE*** R2 sí es así, entonces tenemos una salida en Q5, cualquier otro resultado la salida será falsa.

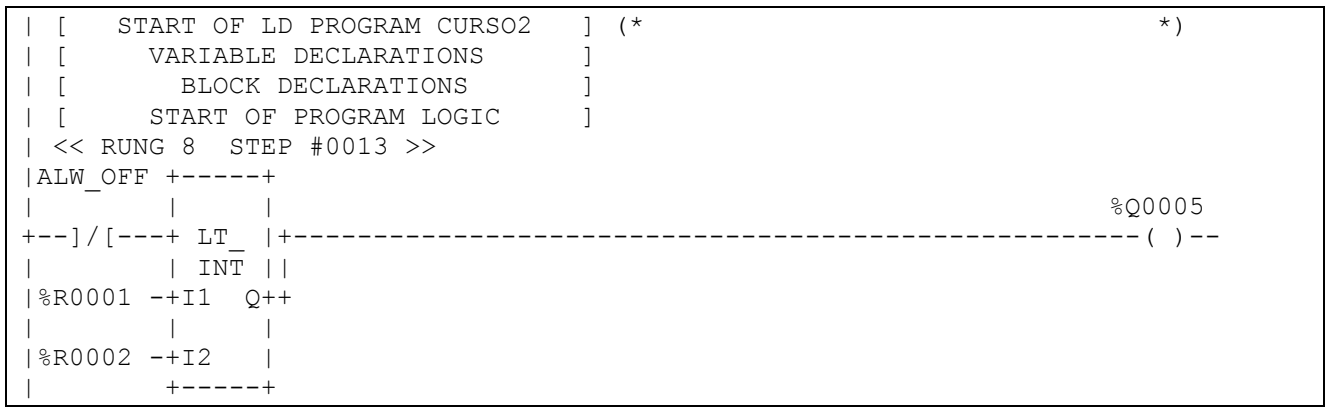


Figura 6.9.5

La figura 6.8.6 nos muestra una función que compara si R1 es ***MENOR O IGUAL QUE*** R2 sí es así, entonces tenemos una salida en Q6, cualquier otro resultado la salida será falsa.

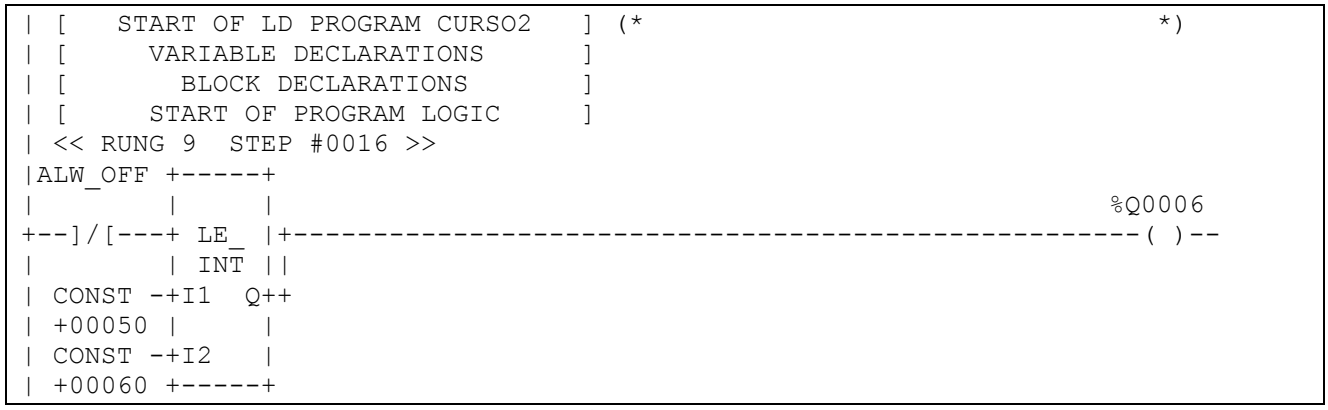


Figura 6.8.6

La figura 6.8.7 nos muestra una función que compara si R1 esta ***EN EL RANGO DE*** R2 sí es así, entonces tenemos una salida en Q7, cualquier otro resultado la salida será falsa.

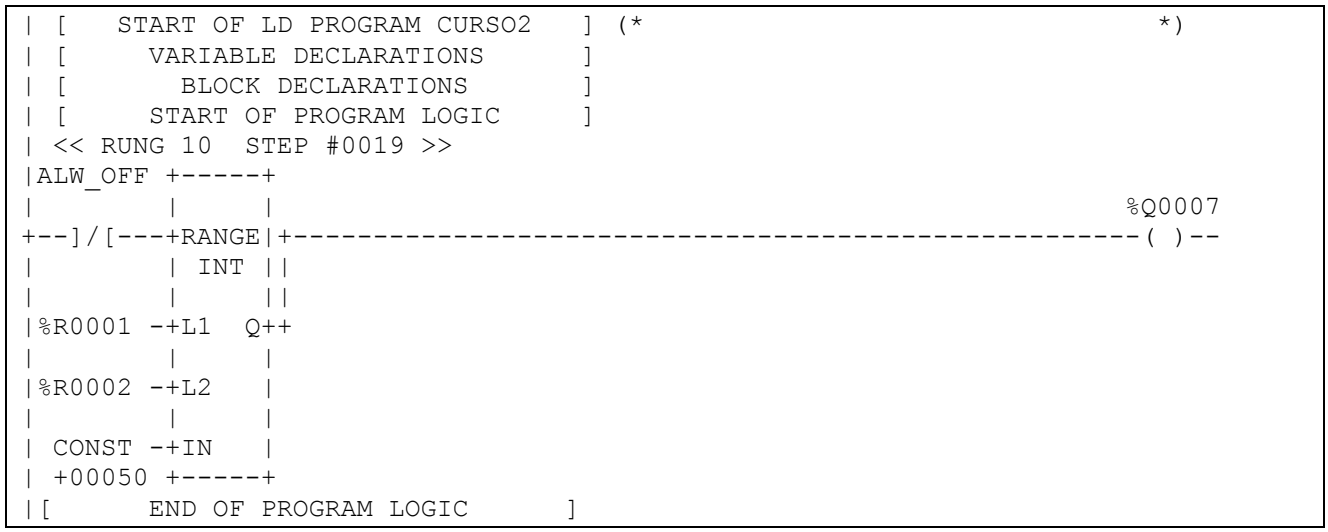


Figura 6.8.6

Por supuesto que en estos ejemplos hemos hablado de comparaciones con registros pero también se pueden comparar constantes.

4.15 - Programación de circuitos con funciones matemáticas. MARCA Ge Fanuc.

La figura 6.10.1 nos muestra una función que suma el valor de R1 con el valor de R2, si el valor del registro en R1 es igual a 5 y el valor de R2 es igual a 6 la suma la deposita en el R3 dando por sumatoria R3 = 11.

NOTA: Si el resultado de la operación no es entera entonces el resultado será el valor superior mas cercano, si se desea que el valor la presente en decimales, entonces se deberá colocar una CPU que realiza operaciones con punto flotante, esto es con decimales.

```
| [  START OF LD  PROGRAM CURSO6   ]     (*                              *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4 STEP #0001 >>
| %I0001   +-----+
+---] [----+ ADD_+-
|          | INT |
|          |     |
| %R0001  -+I1  Q+-%R0003
|          |     |
| %R0002  -+I2   |
|          +-----+
| [      END OF PROGRAM LOGIC      ]
```

Figura 6.10.1

La figura 6.10.2 nos muestra una función que resta el valor de R1 con el valor de R2, si el valor del registro en R1 es igual a 5 y el valor de R2 es igual a 3 la resta la deposita en el R3 dando por sumatoria R3 = 2

```
| [  START OF LD  PROGRAM CURSO7   ]     (*                              *)
| [     VARIABLE DECLARATIONS      ]
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|%I0001  +-----+
+--] [---+ SUB_+-
|         | INT |
|         |     |
|%R0001 -+I1  Q+-%R0003
|         |     |
|%R0002 -+I2   |
|         +-----+
|
       [      END OF PROGRAM LOGIC      ]
```

Figura 6.10.2

La figura 6.10.3 nos muestra una función que multiplica el valor de R1 con el valor de R2, si el valor del registro en R1 es igual a 5 y el valor de R2 es igual a 2 la multiplicación la deposita en el registro R3 dando por multiplicación R3 = 10.

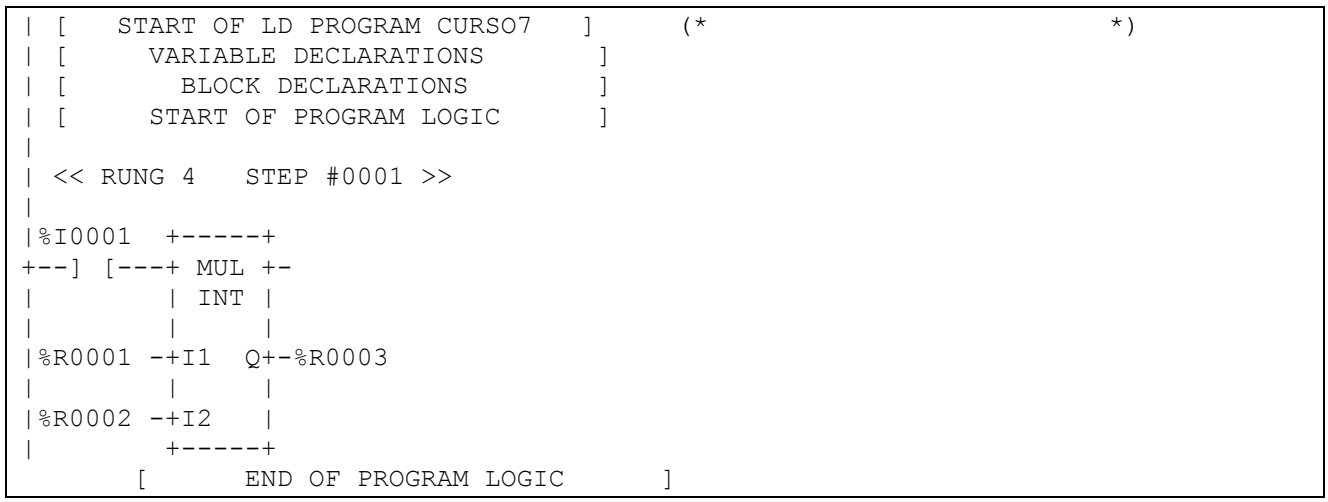


Figura 6.10.3

La figura 6.10.4 nos muestra una función que divide el valor de R1 con el valor de R2, si el valor del registro en R1 es igual a 6 y el valor de R2 es igual a 2 la división la deposita en el registro R3 dando por división R3 = 3.

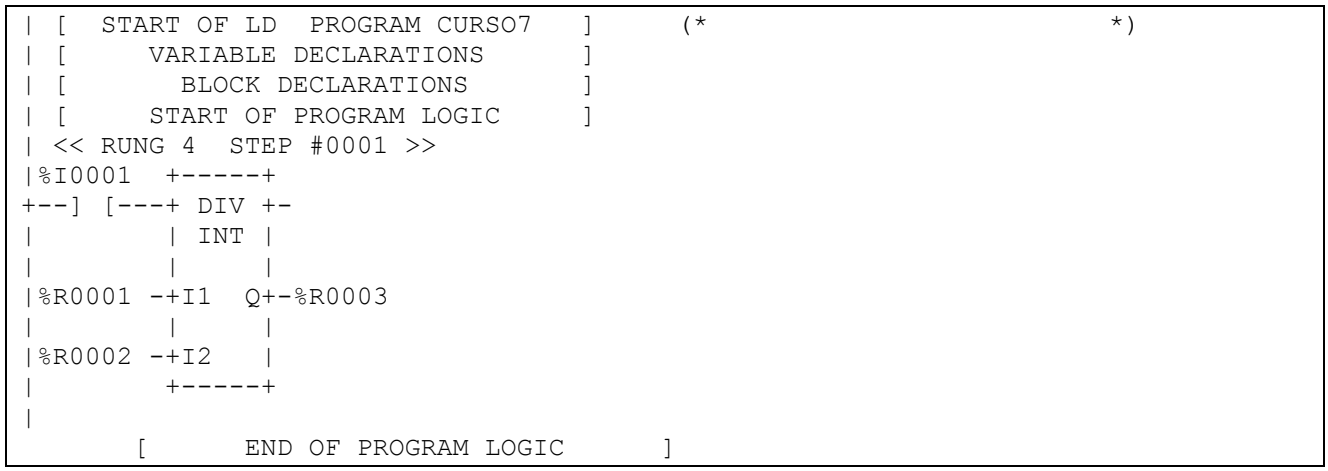
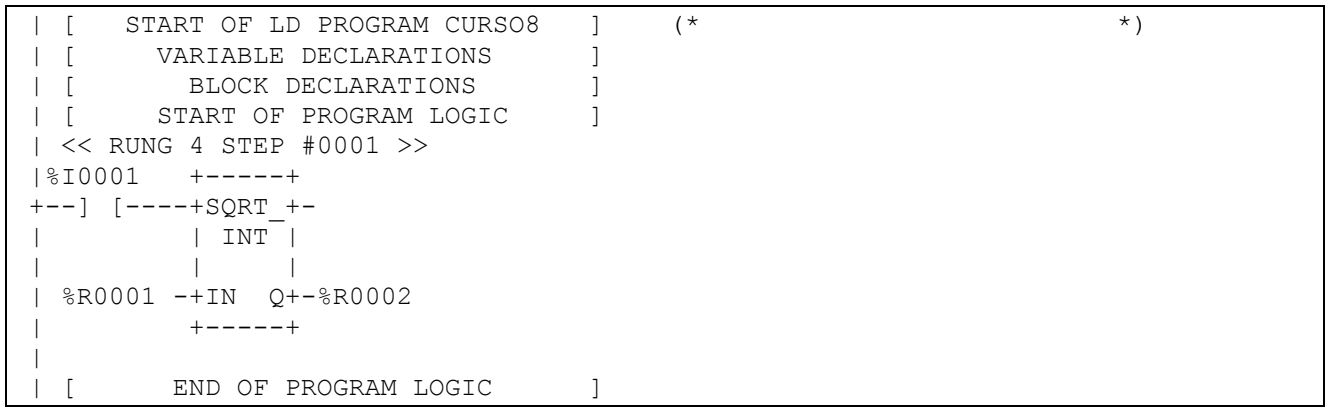


Figura 6.10.4

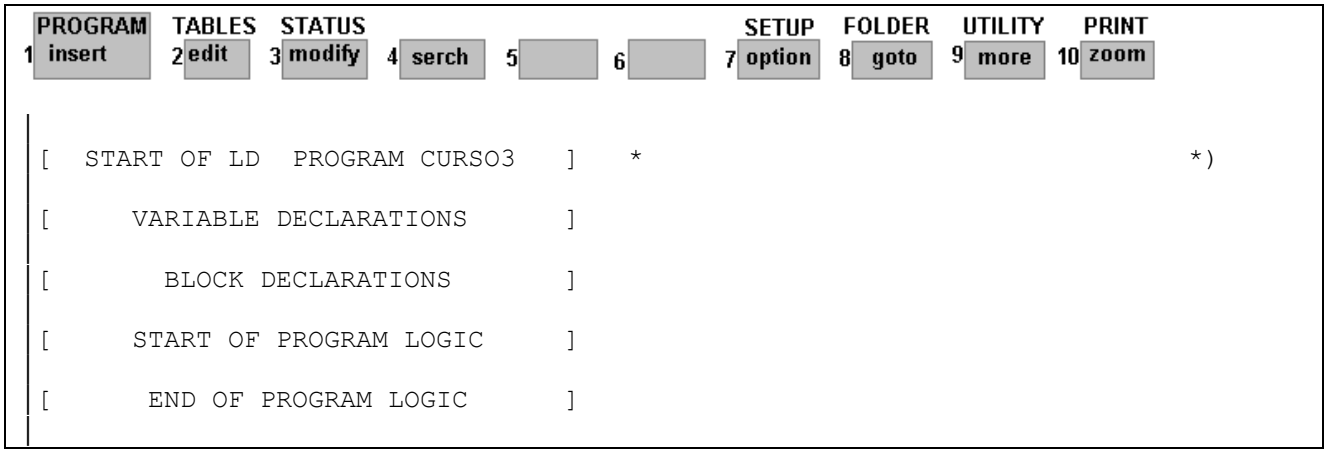
La figura 6.10.5 nos muestra una función que realiza la raíz cuadrada de R1, si el valor de R1 es de 4 el resultado de esta operación la depositará en R2, la cual seria R2 = 2.



4.16 - Programación de circuitos con registros de corrimiento. MARCA Ge Fanuc.

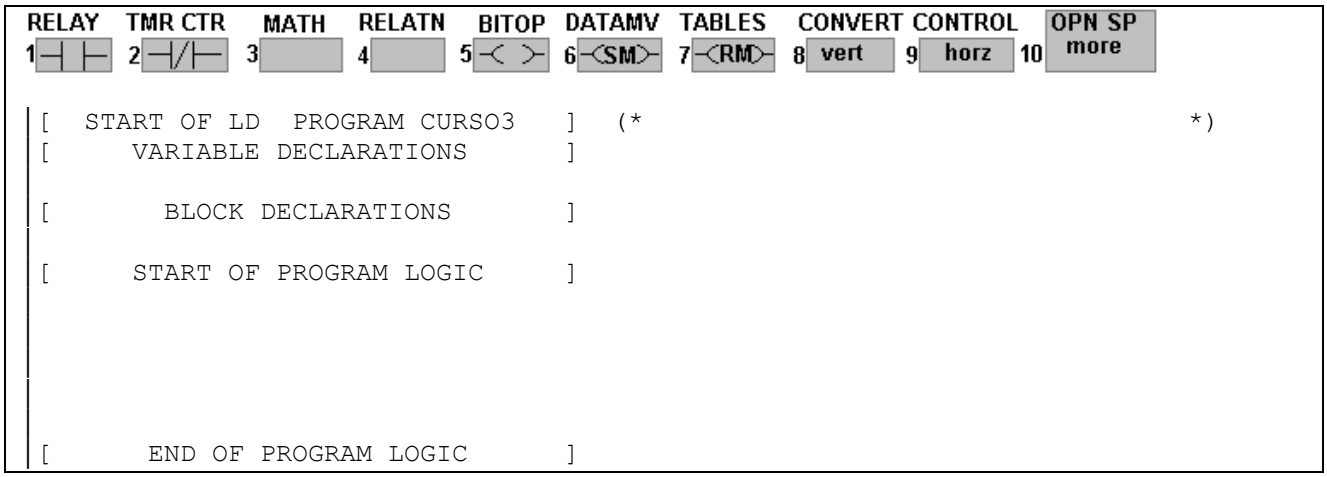
Los registros de corrimiento tienen una gran cantidad de aplicaciones. Primeramente vamos a proceder a explicar como funcionan y posteriormente procederemos a realizar alguna aplicación de los mismos.

Para realizar este ejercicio vamos a partir desde el principio, esto es, vamos a iniciar con un ejercicio totalmente nuevo. Por lo tanto tenemos una pantalla como se muestra en la pantalla 6.11.1



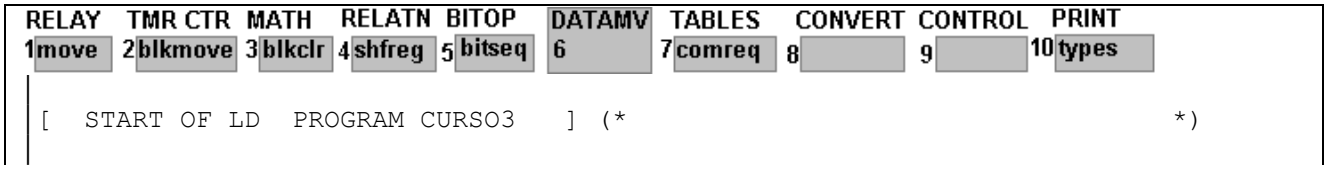
Pantalla 4.16.1

Primeramente, colocamos el cursor donde dice ***END OF PROGRAM LOGIC*** y oprimimos la tecla F1(insert). Y nos aparece la pantalla 6.11.2.



Pantalla 4.16.2

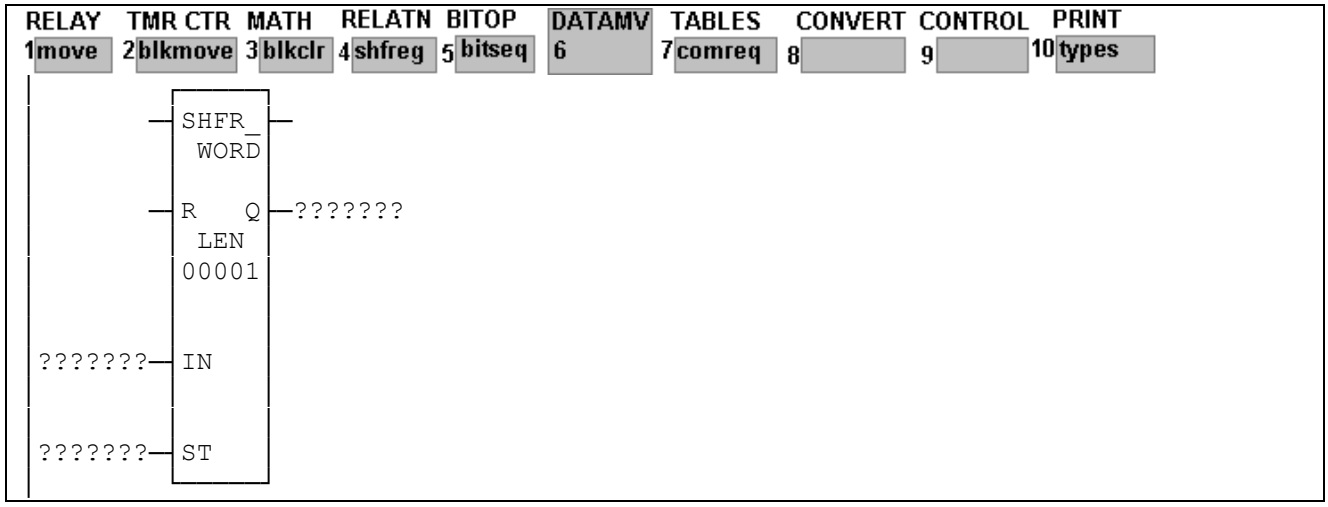
Ahora oprimimos la tecla SHF F6 (data move) y nos aparece la pantalla 6.11.3



```
[      VARIABLE DECLARATIONS       ]

[        BLOCK DECLARATIONS        ]

[      START OF PROGRAM LOGIC      ]


[       END OF PROGRAM LOGIC       ]
```

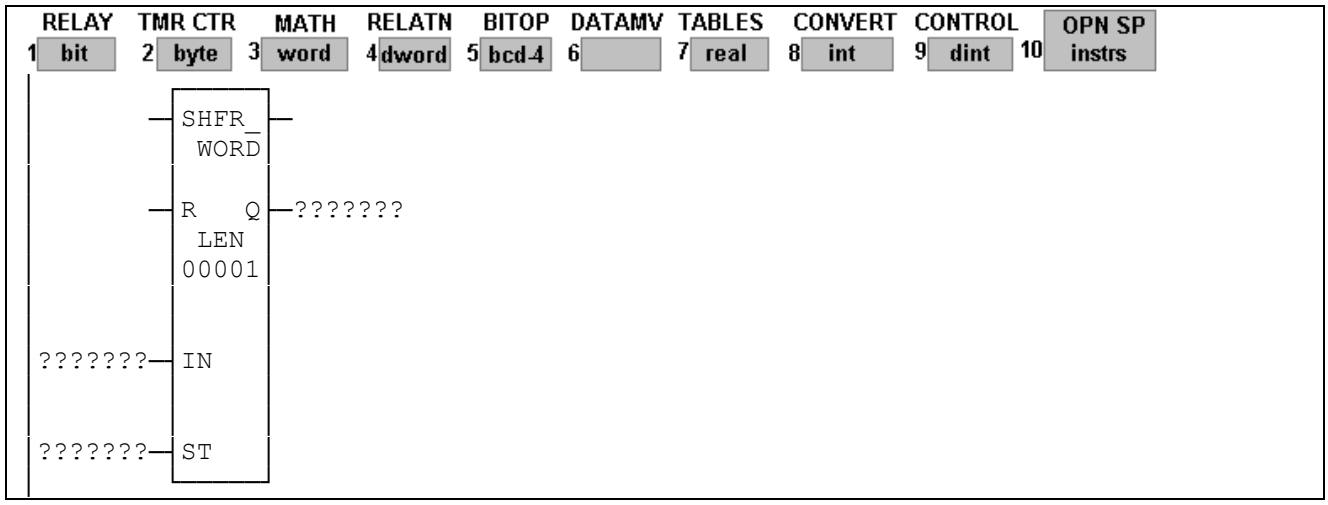
Pantalla 4.16.3

Ahora oprimimos la tecla F4 (shfreg), y nos aparece la pantalla 6.8.4 en la cual nos muestra el bloque completo con un registro de corrimiento.



Pantalla 4.16.4

Pero observe que este es un registro de corrimiento de WORD y nosotros por ahora queremos mover bits por lo tanto observando los iconos de la parte superior se observa que con F10 hay types (tipos) y aparece la pantalla 6.11.5.



Pantalla 4.16.5.

Ahora oprimimos la tecla F1 (bit) y nos aparece la pantalla que se muestra en 6.11.6.

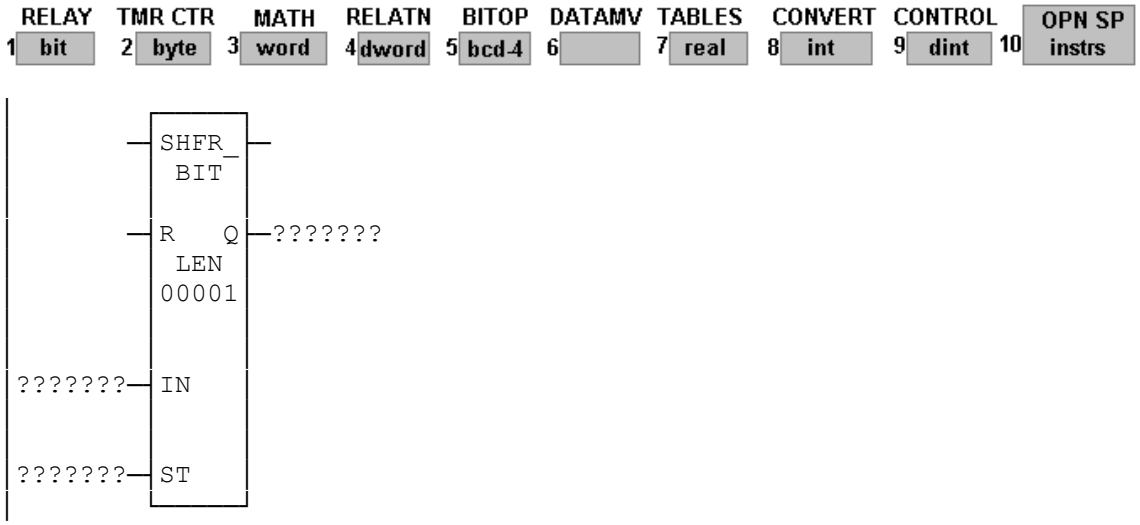


Figura 4.16.6

Finalmente oprimimos la tecla F1 (RELAY) con lo cual procederemos a completar su programación. Observe como M1 es una entrada de clock, el cual le dará pulsos al registro de corrimiento para que avance. Ver figura 6.11.7.

Luego viene FST_SCN que es el reset del registro de corrimiento, en este caso lo estamos reseteando con el primer pulso de S1, (ver detalle de los contactos especiales de S en anexos 1).

En la parte inferir del registro se observa IN esta es la entrada de datos del registro. Y nosotros estamos colocando una entrada digital de I2. Un poco más abajo aparece ST que es la variable que queremos mover en este caso decidimos mover las Q.

En la parte central del registro esta el LEN que es la longitud del registro en este caso se decidió que tuviera una longitud de 8.

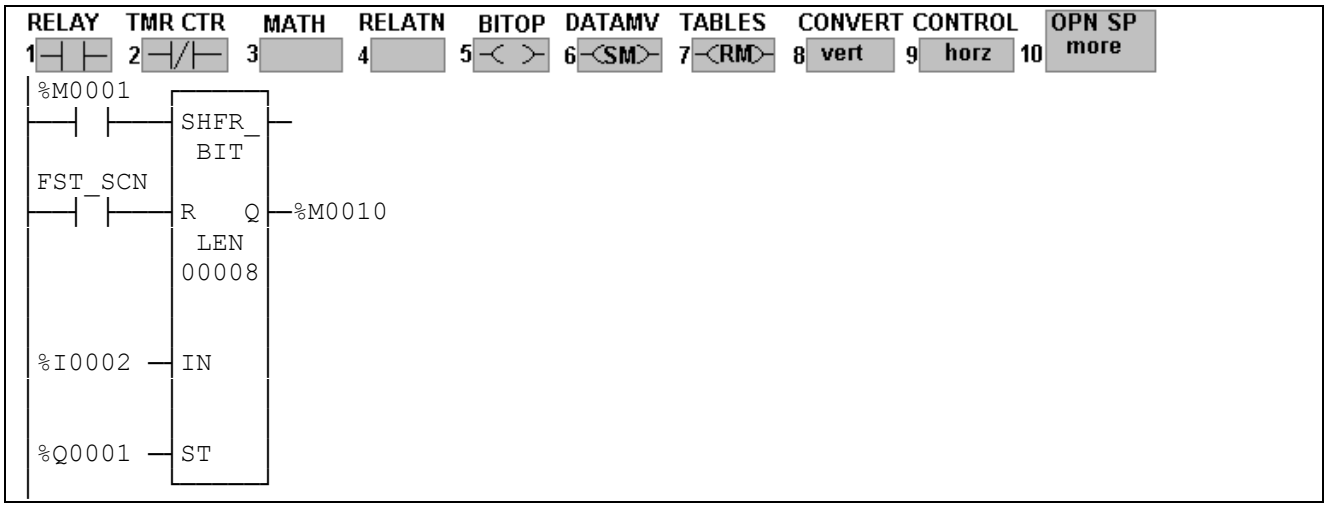


Figura 4.16.7

Finalmente observe como se inserto una línea con I1 de entrada que energiza M1 con un one shot (un disparo) para obligar al registro que solo de un paso de avance. Ver figura 6.11.8

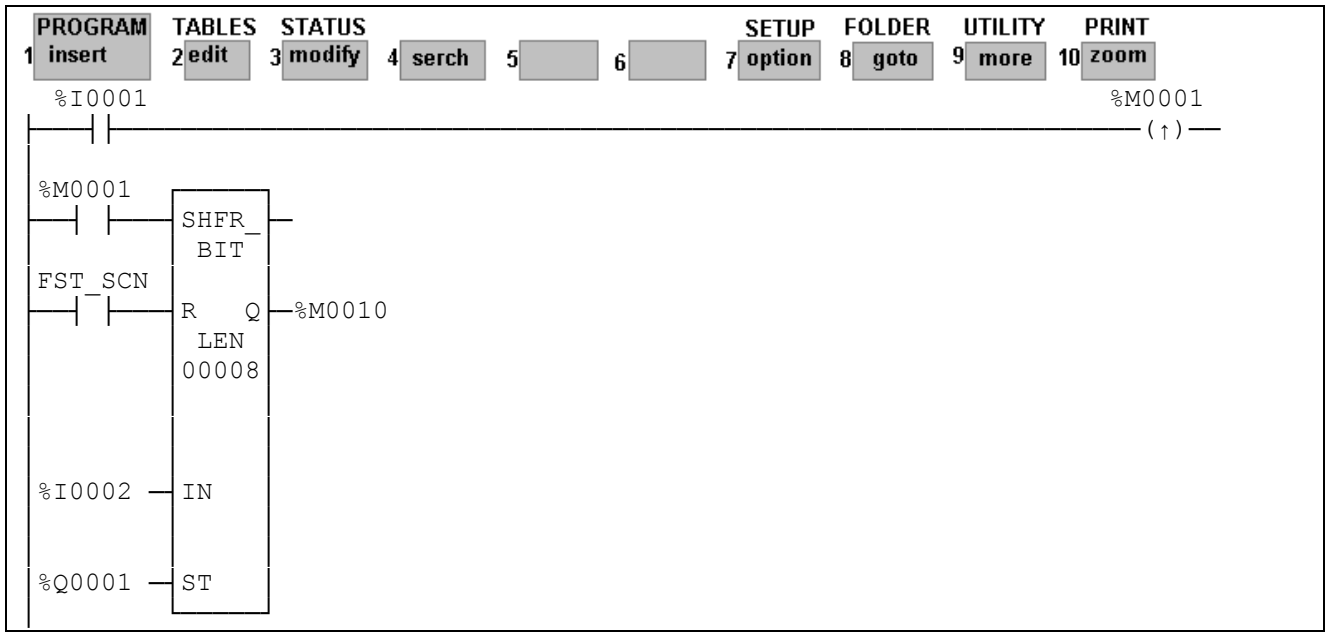


Figura 4.16.8

Funcionamiento general del registro:

Cuando damos un pulso con I1 de clock al registro, el registro introducirá lo que tenga en la entrada de datos si se tenia un uno, bueno meterá un uno, si se tenia un cero bueno mete cero. Hay que hacer la aclaración de que estos datos tal como se encuentra ahora con una longitud de 8, al llegar a este bit empezaran a salirse del registro.

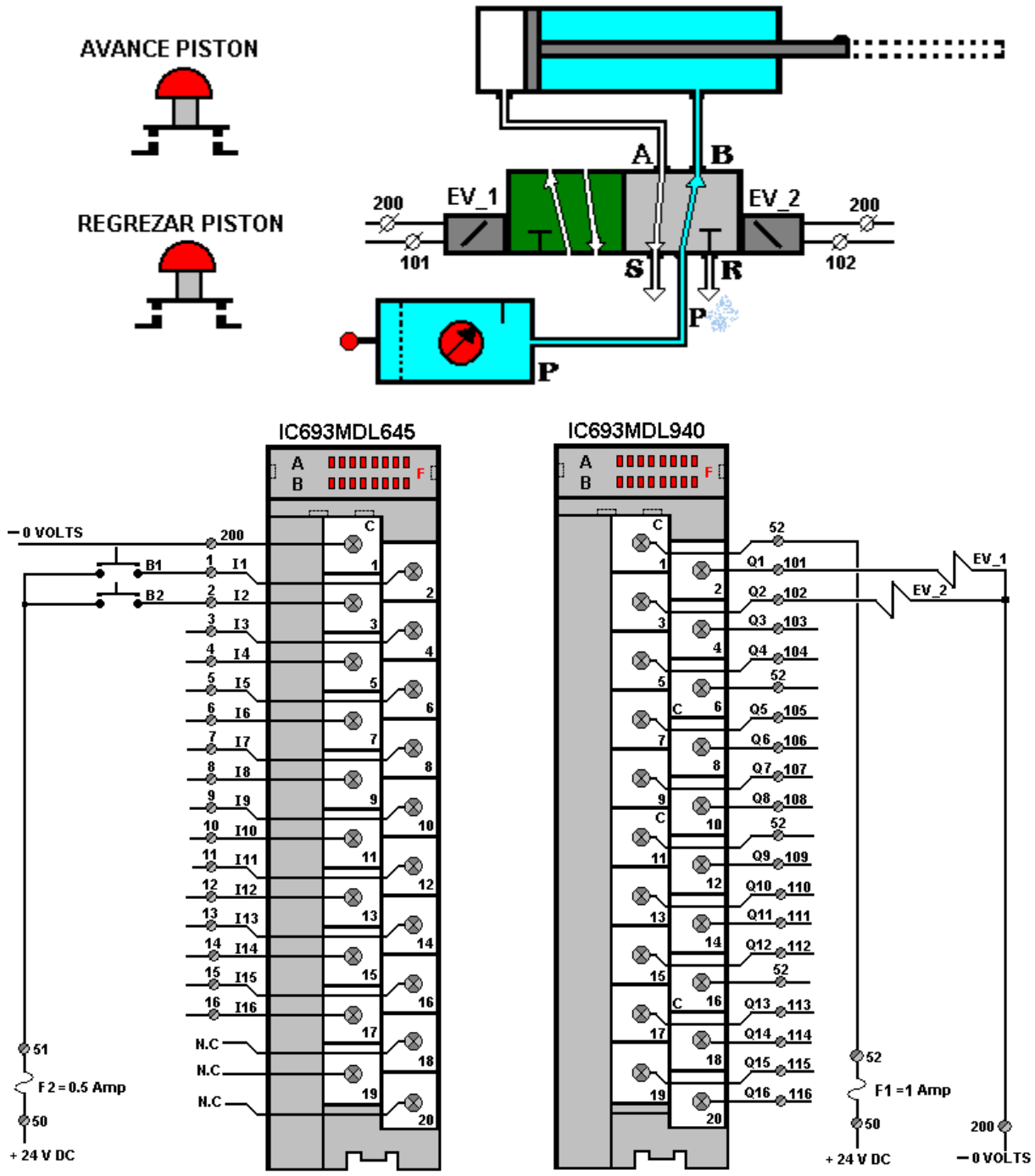
4.17 - Programación de circuitos de automatización con edición de variables Mca Ge Fanuc.

EJEMPLO 4.17

Se tiene una prensa con un cilindro de doble efecto y se desea sujetar una pieza para realizar en ella algún trabajo, por lo tanto se deberán utilizar dos botones eléctricos uno para que baje y sujete la prensa, una vez que se realizó el trabajo con otro botón que regrese a su posición.

Realice el cableado al PLC, el circuito de control electro neumático con control programable, declaración de variables y normatividad técnica.

SOLUCIÓN:

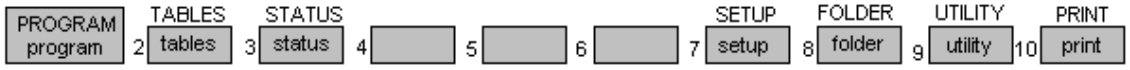


EJEMPLO 4.17

Como podemos observar en la figura anterior, se a dibujado el pistón de doble efecto con su respectiva electro válvula de 5 vías 2 posiciones con accionamiento eléctrico en ambos lados (EV_1 y EV_2) y su correspondiente identificación del cableado, para la electro válvula EV_1 los números (86 y 200) y para la electro válvula EV_2 con (87 y 200), las cuales fueron cableados en el modulo de 16 salidas a relevador y direccionadas con Q1 y con Q2. Para las entradas se conectaron en un modulo de 16 entradas, el botón B1 sujetar pistón se direcciono en la entrada I1 con cableado (100 y 1) y el botón de regresar pistón en I2 con cableado (100 y 2).

Una vez que se realizó el cableado, se procede a diseñar el programa de control.

Primero nos colocamos en la siguiente pantalla, en la cual se le a dado un nombre al fólder en este caso CURSO 9.



S E R I E S 90-30 / 90-20 / MICRO P R O G R A M M I N G S O F T W A R E

Version 9.02 Direct Serial – COM

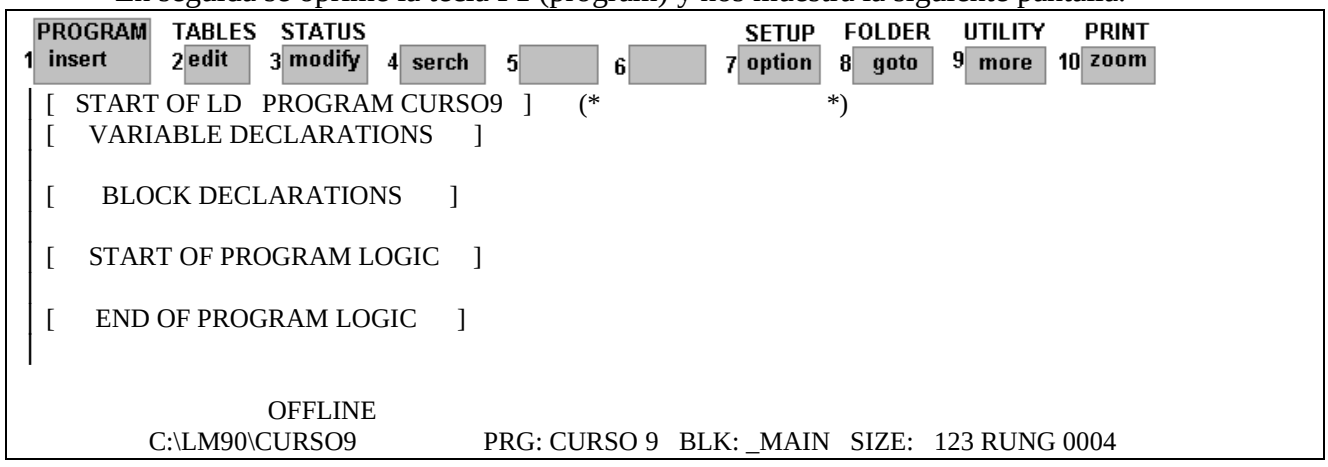
F1 ..... Program Display/Edit
F2 ..... Reference Tables
F3 ..... PLC Control and Status

F7 ..... Programmer Mode and Setup
F8 ..... Program Folder Functions
F9 ..... Utility: Load/Store/etc.
F10 .... Print Functions

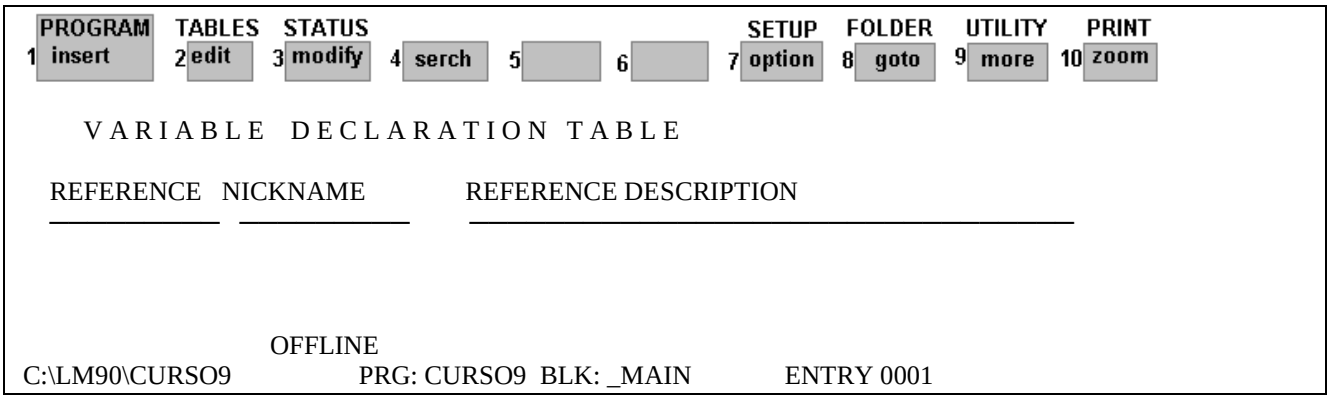
<< Press ALT-K at any time to see special key assignments >>

OFFLINE
C:\LM90\CURSO9 PRG: CURSO9

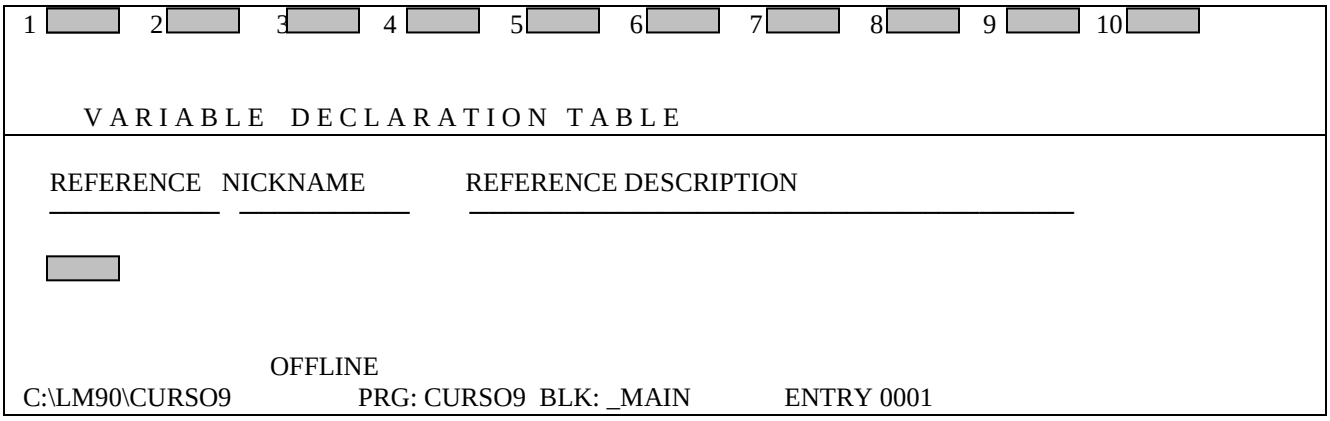
En seguida se oprime la tecla F1 (program) y nos muestra la siguiente pantalla.



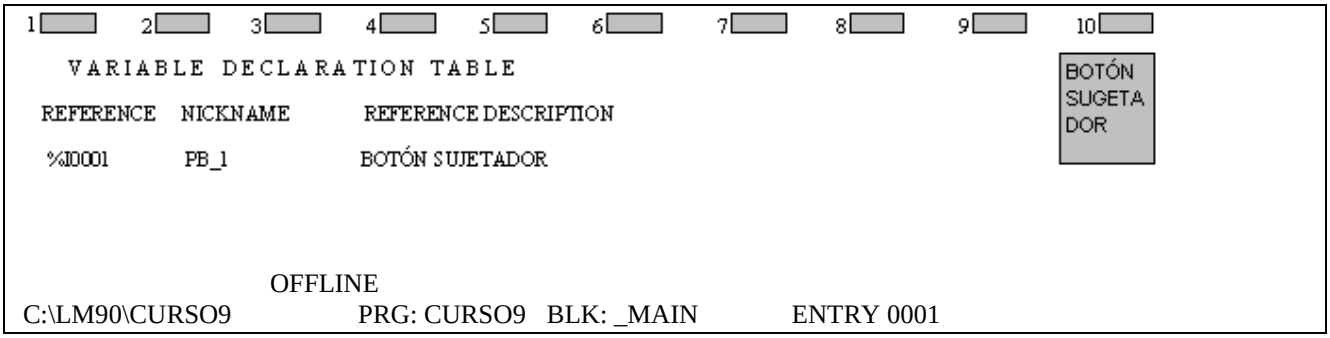
Ahora colocamos el cursor en VARIABLE DECLARATIONS y nos muestra la siguiente pantalla.



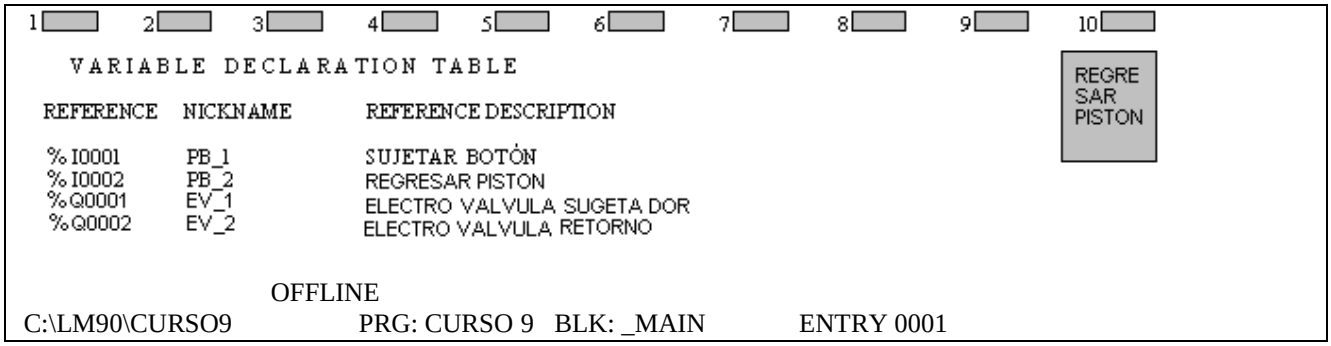
En seguida oprimimos la tecla F1 (insert) y nos muestra la siguiente pantalla.



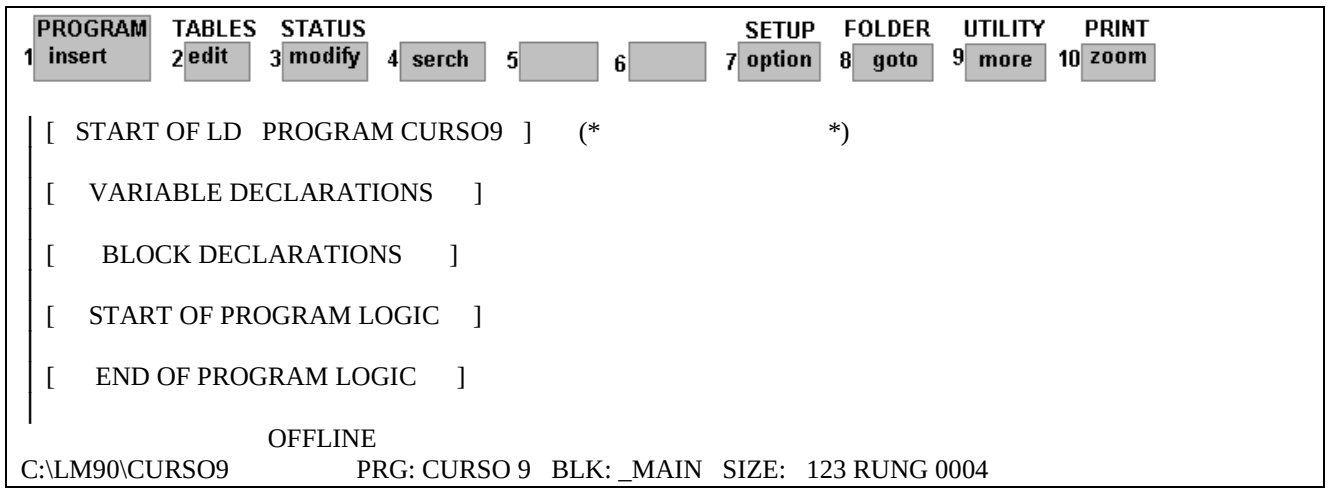
En seguida en REFERENCE colocamos la variable I1, en NICKNAME colocamos un nombre corto para definir la I1 (PB_1) y finalmente en REFERENCE DESCRIPTION describimos el significado completo de I1 (BOTÓN SUJETADOR) observando siempre el recuadro de la derecha para que quede bien descrito ya que es lo que veremos en el diagrama de escalera.



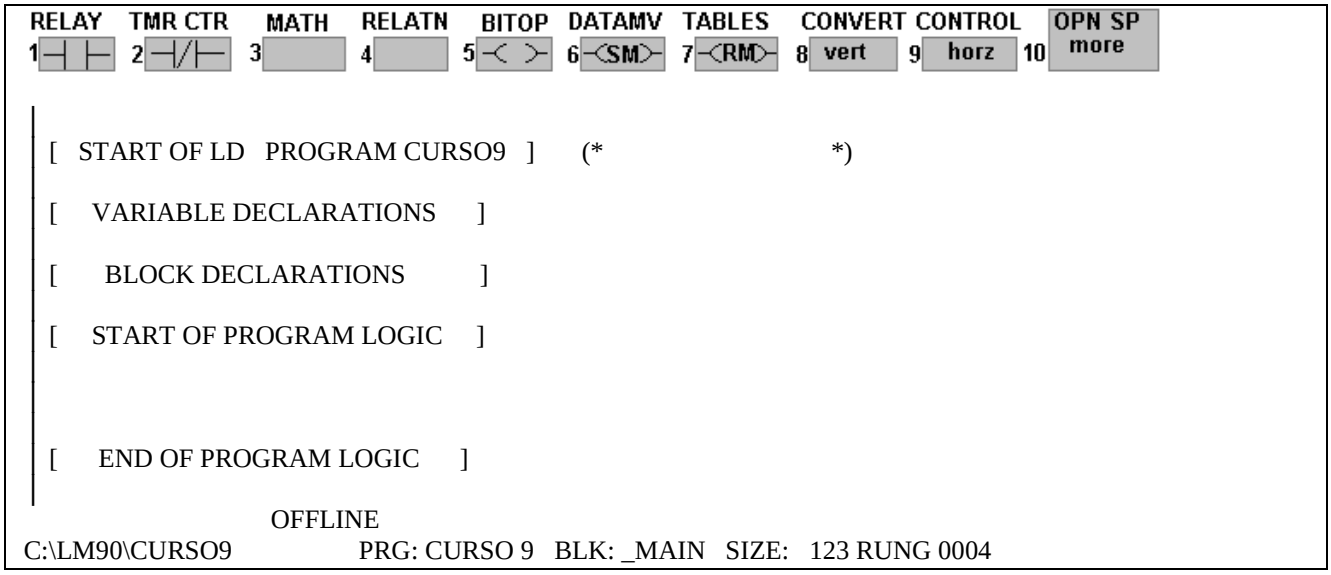
En seguida definimos la variable para I2 y las salidas Q1 y Q2.



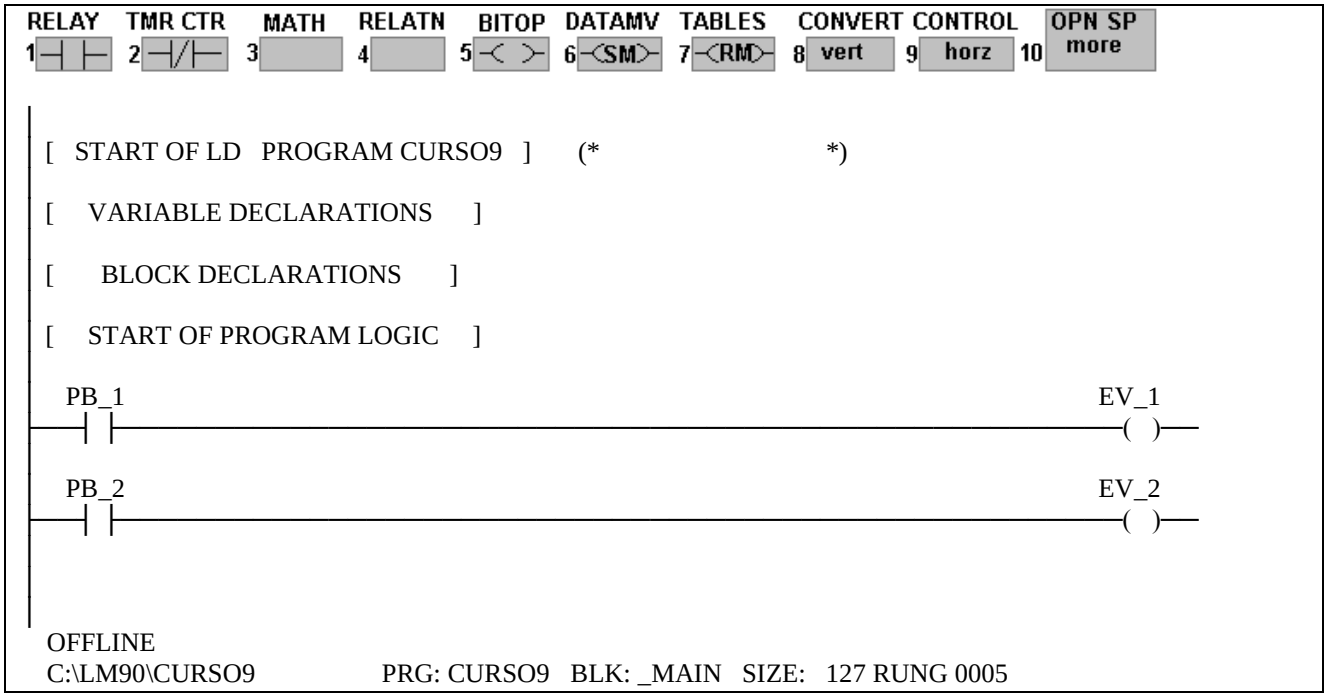
Una vez declaradas las variables, se oprime la tecla ESC y nos muestra la siguiente pantalla.



Ahora colocamos el cursor en (END OF PROGRAM), oprimimos la tecla F1 (insert) y nos muestra la siguiente pantalla.

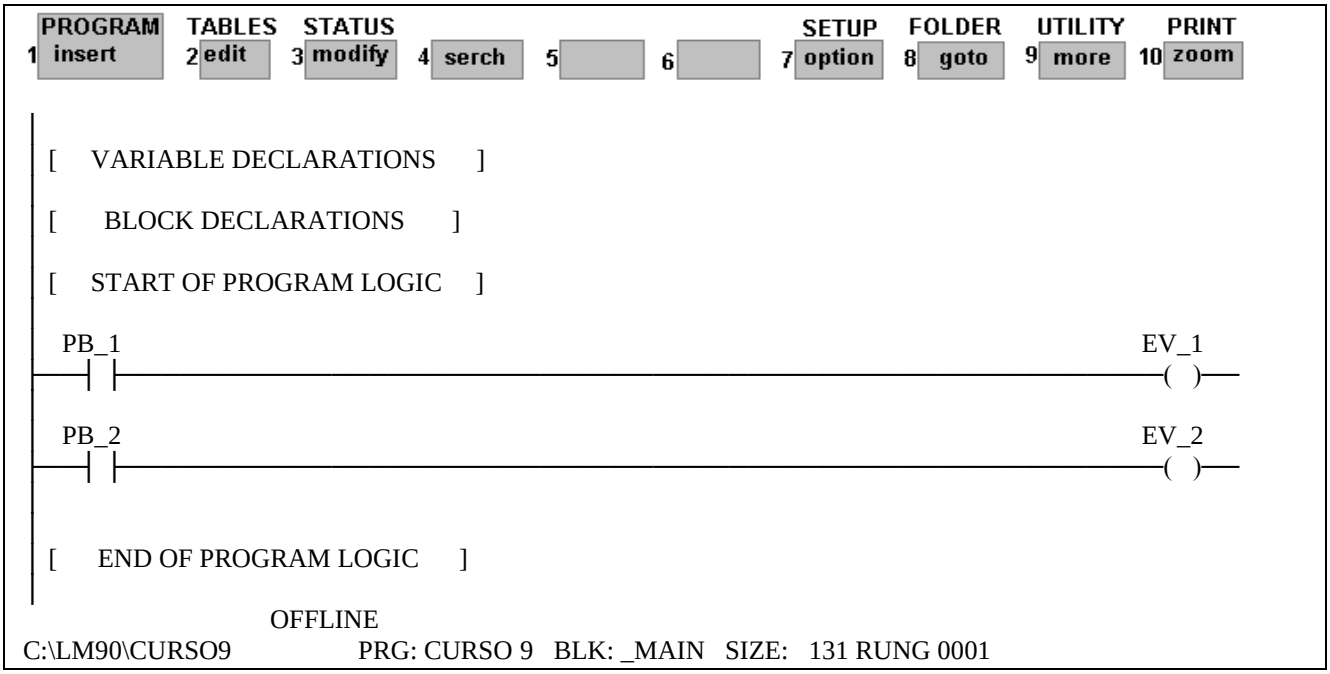


En seguida realizamos el circuito de control.

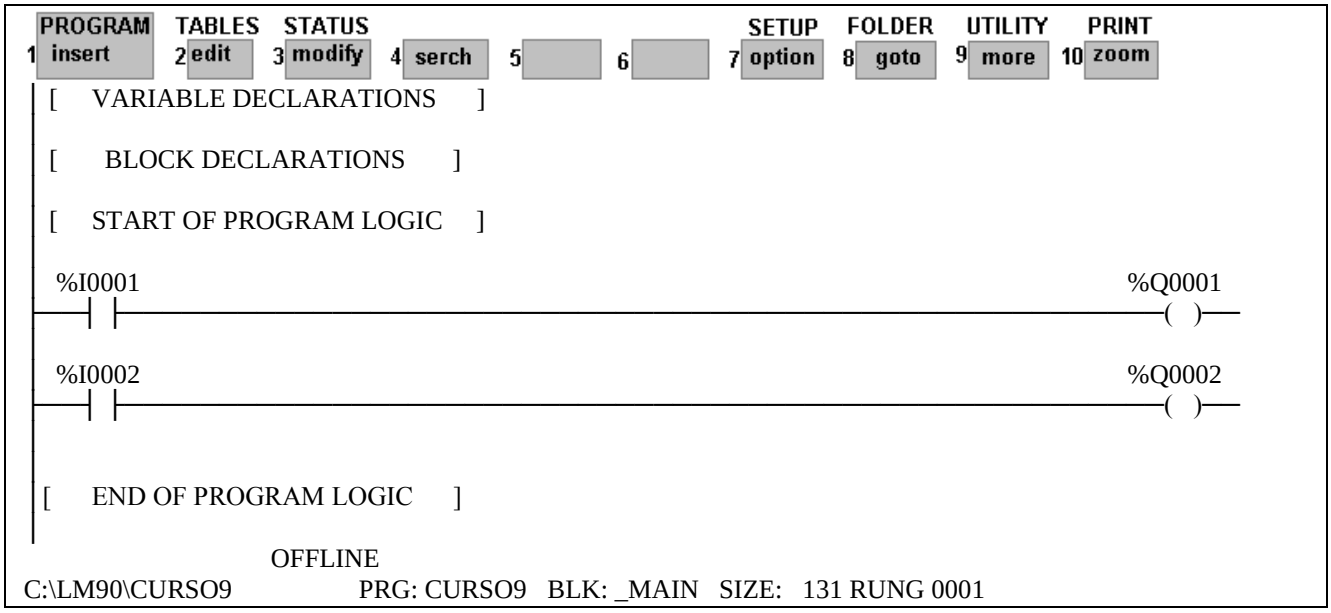


NOTA: Recuerde para iniciar una nueva línea, se debe oprimir la tecla ENT.

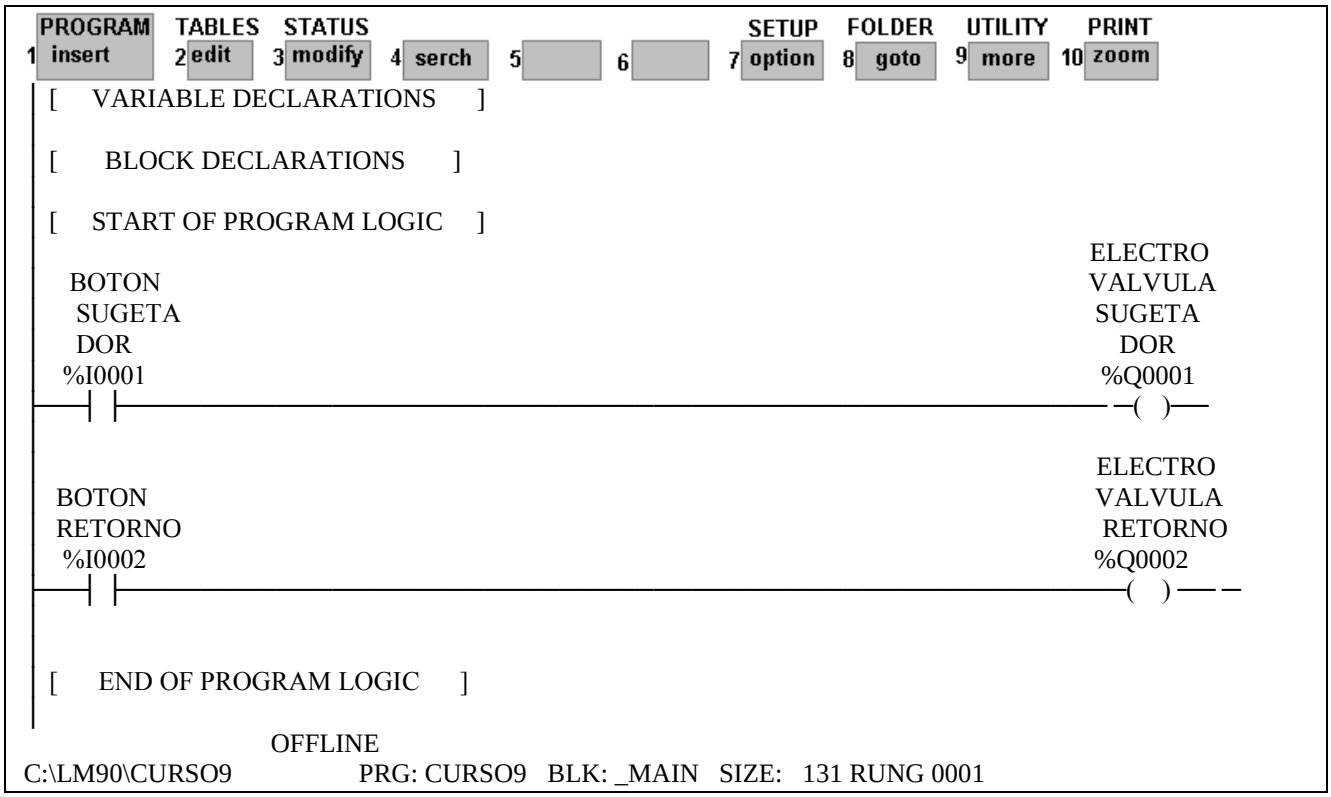
Finalmente oprimimos la tecla ESC para terminar la programación.



Para ver el significado complete se oprimen las teclas ALT+N tantas veces sea necesario hasta encontrar la forma que mas se adapte a sus necesidades.



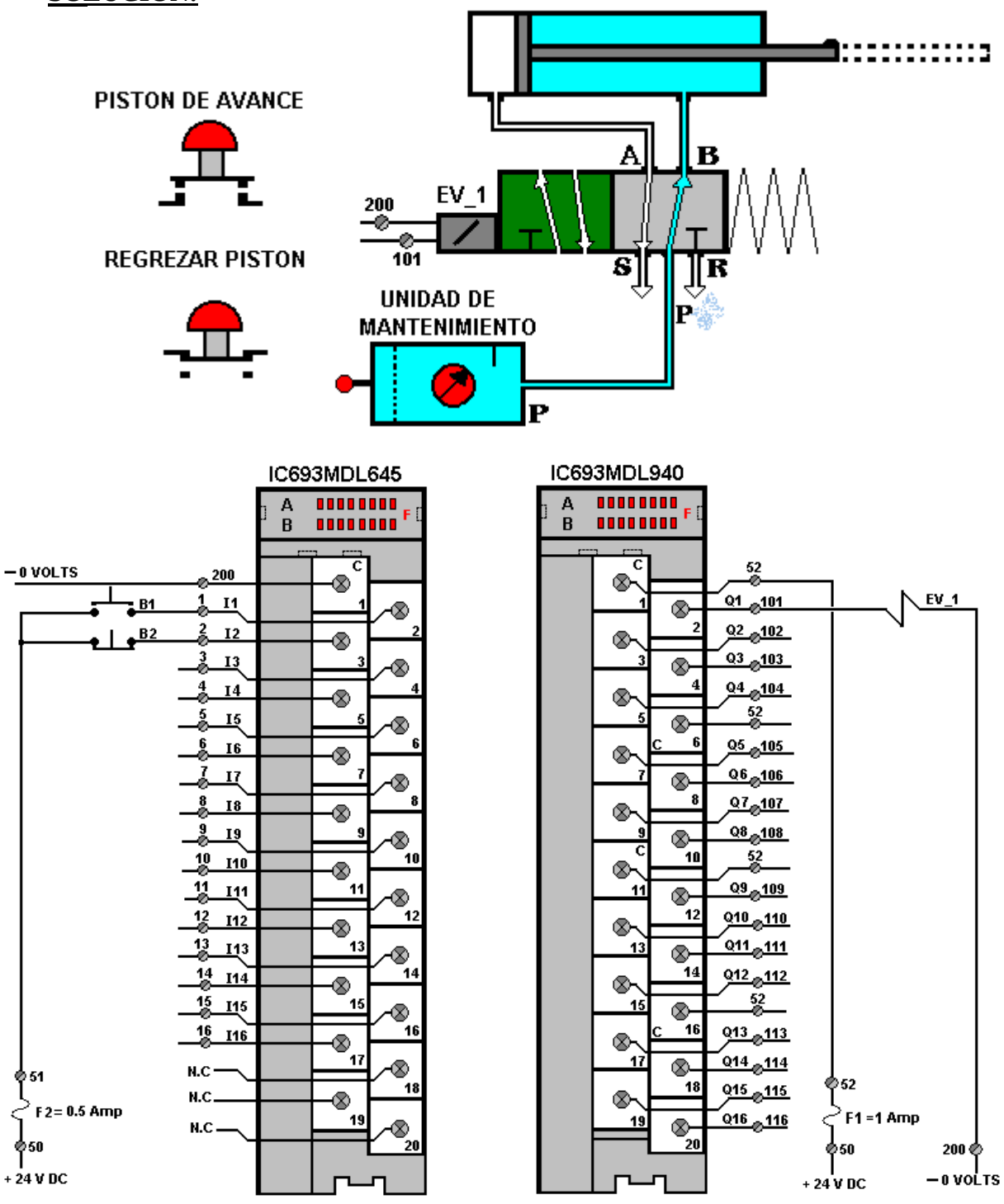
Al oprimir las teclas ALT+N



EJERCICIO 4.17.2

Realice el mismo circuito pero ahora con una electro válvula con una sola solenoide (EV_1) y retorno por muelle.

SOLUCIÓN:



EJEMPLO 4.17.2

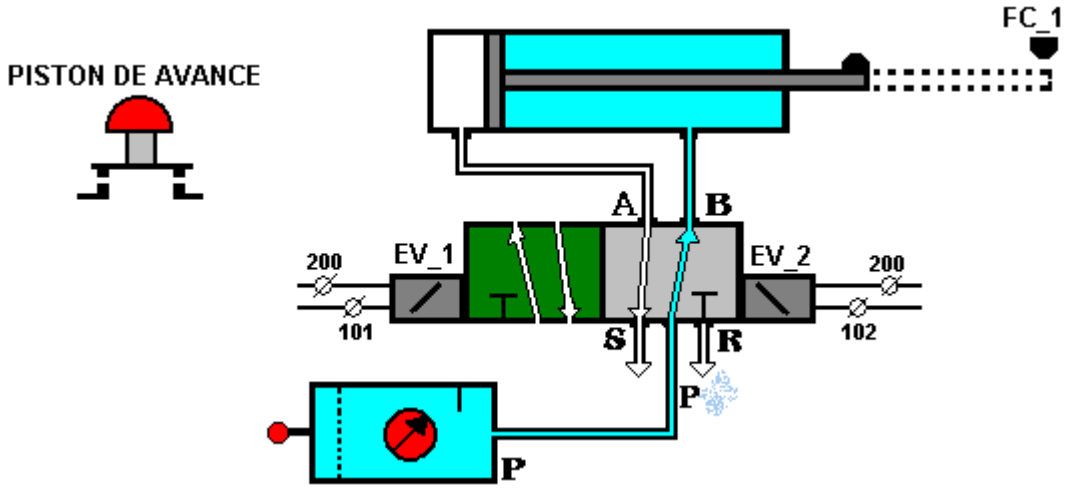
A continuación presentaremos solo el diseño del programa de control, esperando que el alumno siga las instrucciones del ejemplo anterior para el desarrollo del presente circuito.
Observe como el contacto de I2 se coloco abierto ya que el Botón PB_2 se cableo cerrado.

```
| [   START OF LD    PROGRAM CUR_10    ]        (*                                  *)
|
| [        VARIABLE DECLARATIONS       ]
|
| [          BLOCK DECLARATIONS        ]
|
| [        START OF PROGRAM LOGIC      ]
|
| << RUNG 4       STEP #0001 >>
|
|  PB_1     PB_2                                                          EV_1
+---] [--+--] [-------------------------------------------------------------( )--
|        |
|  EV_1  |
+---] [--+
|
| [      END OF PROGRAM LOGIC       ]

Program: CUR_10                  C:\LM90\CUR_10                  Block: _MAIN
```

EJERCICIO 4.17.3

Diseñe el circuito de control electro neumático para el ejercicio No 1 con electro válvula de doble solenoide pero ahora que al llegar al final de su carrera se regrese en forma automática.

SOLUCIÓN:



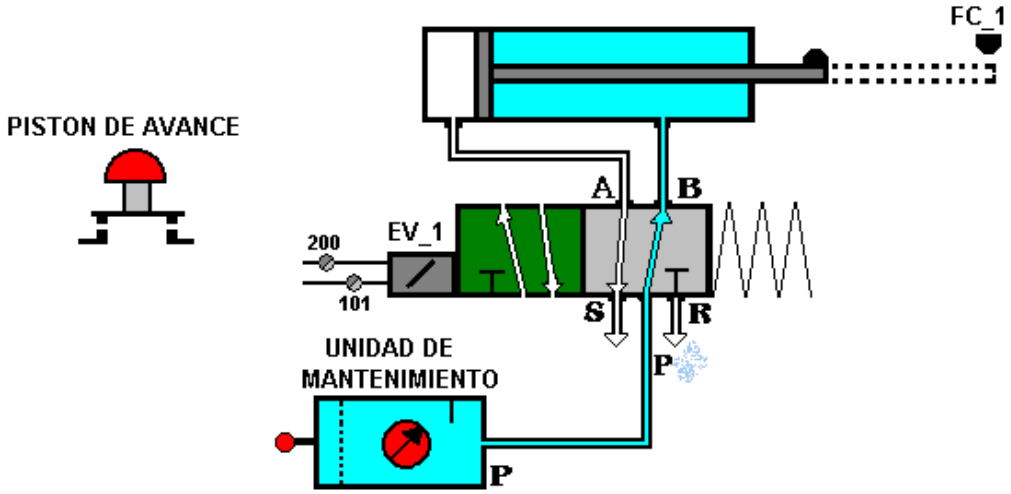

```
| [   START OF LD    PROGRAM CUR_11    ]        (*                          *)
| [     VARIABLE DECLARATIONS       ]
          V A R I A B L E   D E C L A R A T I O N   T A B L E
        REFERENCE     NICKNAME          REFERENCE DESCRIPTION
        ---------     --------      --------------------------------
        %I0001        PB_1          BOTON AVANCE
        %I0002        FC_1          FINAL DE CARRERA 1
        %Q0001        EV_1          EV_1
        %Q0002        EV_2          EV_2
                      I D E N T I F I E R    T A B L E
```

```
IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
 ----------     ---------------     --------------------------------
          CUR_11          PROGRAM NAME

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4     STEP    #0001 >>
|
| PB_1                                                          EV_1
+--] [-------------------------------------------------------------( )--
|
| << RUNG 5     STEP    #0003 >>
|
| FC_1                                                          EV_2
+--] [-------------------------------------------------------------( )--
|
| [      END OF PROGRAM LOGIC      ]
  Program: CUR_11        C:\LM90\CUR_11             Block: _MAIN
```

EJERCICIO 4.17.4

Como podrá usted observar en el ejemplo de la figura anterior, si el pistón realiza su trabajo muy rápido y no alcanza el operador a soltar la mano del botón , el pistón sale de nuevo pudiendo dañar la pieza al repetir el ciclo, por lo tanto realice el circuito de protección para que no repita el ciclo. (Observe que el pistón tiene solenoide con retorno por muelle).

SOLUCIÓN:



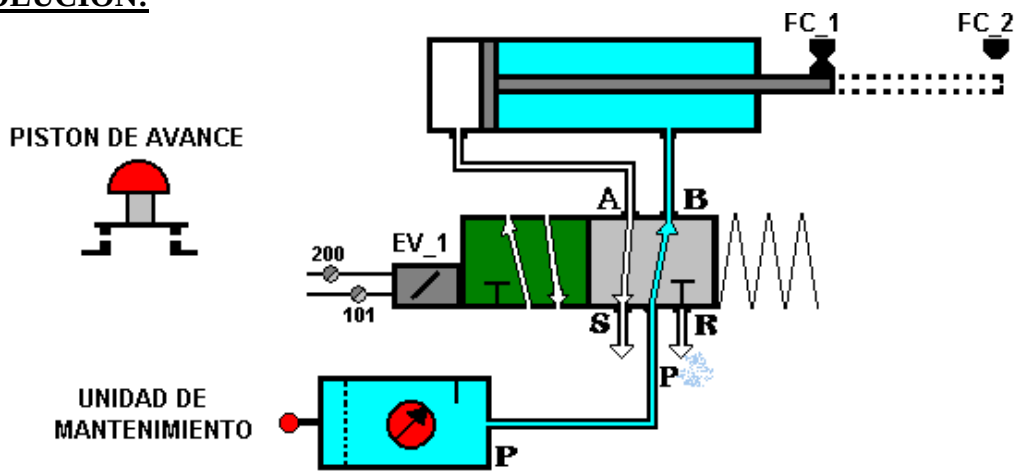

```
| [  START OF LD  PROGRAM CUR_12   ]    (*                                *)
| [     VARIABLE DECLARATIONS      ]
            V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        PB_1         BOTON AVANCE
          %I0002        FC_1         FINAL DE CARRERA 1
          %Q0001        EV_1         ELECTRO VAL 1
          %M0001                     UN DISPARO
                        I D E N T I F I E R    T A B L E
```

```
   IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
   ----------     ---------------     --------------------------------
| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|BOTON
|AVANCE
| PB_1                                                                %M0001
+--] [-------------------------------------------------------------------(^)--
| << RUNG 5  STEP #0003 >>
|
|          FINAL
|UN        DE CARRE
|DISPARO   RA 1                                                        EV_1
|%M0001    FC_1                                                        Q1
+--] [--+--]/[-----------------------------------------------------------( )--
|       |
|ELECTRO|
|  VAL1 |
| EV_1  |
+--] [--+
| [      END OF PROGRAM LOGIC      ]
Program: CUR_12          C:\LM90\CUR_12              Block: _MAIN
```

EJEMPLO 4.17.5

En el circuito anterior se puede presentar el caso de que el pistón después de estar un tiempo parado tienda a bajar un poco, o que por alguna razón no llega a la posición de reposo con lo que las piezas trabajadas no tienen la calidad requerida. Diseñe el circuito de control para que si el pistón no esta en la posición de reposo no inicie el ciclo, además de que tenga la condición de que no repita ciclo.

SOLUCIÓN:



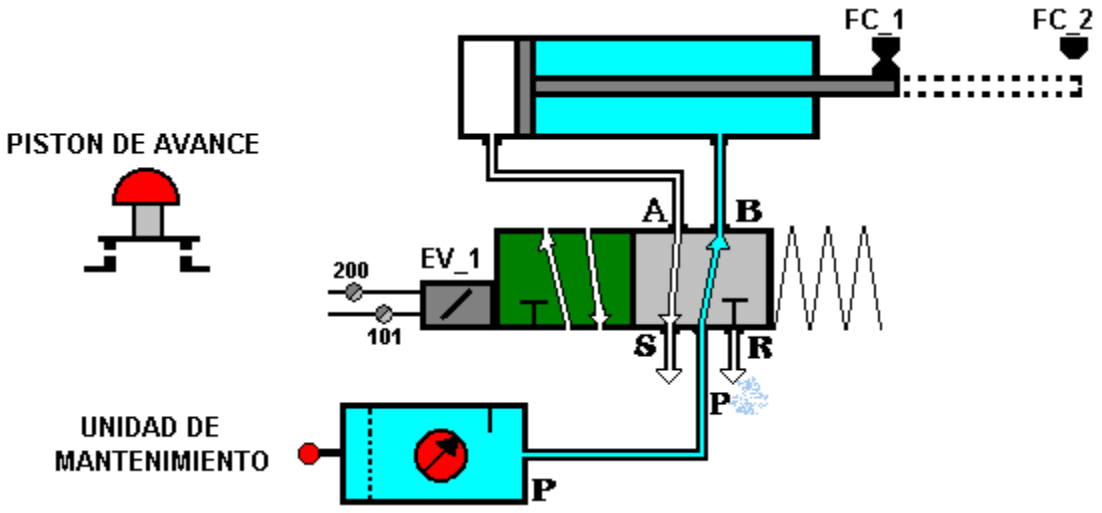

```
|[  START OF LD  PROGRAM CURSO9   ]    (*                          *)
|[     VARIABLE DECLARATIONS      ]

            V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        FC1_P1R      FC1 PISTON1 REPOSO
```

```
           %I0002         FC2_P1A       FC2 PISTON1 ADLANTE
           %I0003         PB_INI        BOTON INICIO
           %Q0001         EV_1          ELECTRO VALVULA
                          I D E N T I F I E R    T A B L E
  IDENTIFIER     IDENTIFIER TYPE           IDENTIFIER DESCRIPTION
----------     ---------------     --------------------------------
           CURSO9          PROGRAM NAME
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4   STEP #0001 >>
|
|BOTON
|INICIO
|PB_INI                                                        %M0001
+--] [-------------------------------------------------------------(^)--
|
| << RUNG 5   STEP #0003 >>
|
|         FC1      FC2
|         PISTON1  PISTON1
|         REPOSO   ADLANTE
|%M0001   FC1_P1R  FC2_P1A                                      EV_1
+--] [-----] [--+--]/[-----------------------------------------( )--
|               |
|ELECTRO        |
|VALVULA        |
| EV_1          |
+--] [----------+
|[      END OF PROGRAM LOGIC      ]
|
Program: CURSO9                   C:\LM90\CURSO9
```

EJEMPLO 4.17.6

Realice el circuito de control para el ejercicio anterior pero ahora se requiere pegar dos piezas, por lo que el pistón lleva en la punta un troquel con una resistencia para calentar el troquel de tal manera que al bajar dure un tiempo abajo y luego suba en forma automática.

SOLUCIÓN:



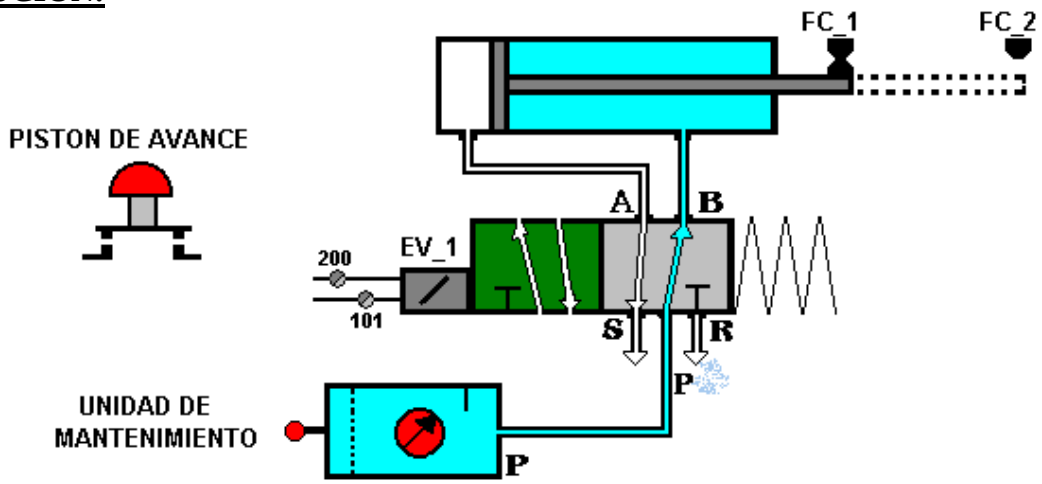
```
| [  START OF LD  PROGRAM  EJE1    ]       (*                          *)

| [     VARIABLE DECLARATIONS      ]

             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        FC1_P1R      FC1 PISTON1 REPOSO
          %I0002        FC2_P1A      FC2 PISTON1 ADLANTE
          %I0003        PB_INI       BOTON INICIO
          %Q0001        EV_1         ELECTRO VALVULA
          %T0001        TPO_PEG      TIEMPO PEGADO

                        I D E N T I F I E R    T A B L E

IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
 ----------     ---------------     --------------------------------
          EJE1           PROGRAM NAME

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|PB_INI
|%I0003                                                        %M0001
+--] [-------------------------------------------------------------(^)--

| << RUNG 5  STEP #0003 >>
|        FC1_P1R TPO_PEG                                        EV_1
|%M0001  %I0001  %T0001                                        %Q0001
+--] [-----] [--+--]/[-----------------------------------------------( )--
| EV_1          |
|%Q0001         |
+--] [----------+

| << RUNG 6  STEP #0008 >>
| EV_1   FC2_P1A                                              TPO_PEG
|%Q0001  %I0002  +-----+                                       %T0001
+--] [-----] [---+ TMR +-------------------------------------------( )--
|                |0.10s|
|                |     |
|         CONST -+PV   |
|         +00030 |     |
|                +-----+
|                %R0001

    Program: EJE1          C:\LM90\EJE1               Block:    MAIN
```

EJEMPLO 4.17.7

Si analizamos el problema anterior, en el cual el operador pone la pieza a pegar con una mano y con la otra da el inicio de ciclo. Imaginemos que el operador después de estar realizando esta operación durante un periodo prolongado y como usted podrá observar bastante fastidioso, comete el error de no retirar la mano derecha con la que pone la pieza y da inicio de ciclo, como el pistón baja muy rápido le sujetara la mano con el pistón que trae un troquel caliente provocándole el accidente correspondiente.

Realice la protección necesaria para evitar este problema. Para resolver este problema, se requiere utilizar ambas manos, esto es dar inicio ciclo con dos botones, uno en cada mano.

SOLUCIÓN:




```
| [  START OF LD  PROGRAM EJEM_8   ]     (*                              *)
| [     VARIABLE DECLARATIONS      ]

             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        PB_DER       BOTON DERECHO
          %I0002        PB_IZQ       BOTON IZQ.
          %I0003        FC1_P1R      FC1 PISTON1 REPOSO
          %I0004        FC2_P1A      FC2 PISTON1 ADELAN.
          %Q0001        EV1_P1       ELECTRO VALVULA
          %T0001        TPO_PEG      TIEMPO PEGADO

                        I D E N T I F I E R    T A B L E
          IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
        ----------     ---------------     --------------------------------
          EJEM_8          PROGRAM NAME

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|
|                FC1     FC2
|BOTON   BOTON   PISTON1 PISTON1 TIEMPO
|DERECHO IZQ.    REPOSO  ADELAN. PEGADO
|PB_DER  PB_IZQ  FC1_P1R FC2_P1A TPO_ PEG                          EV1_P1
+--] [-----] [-----] [-----]/[--+--]/[--------------------------------( )-
```

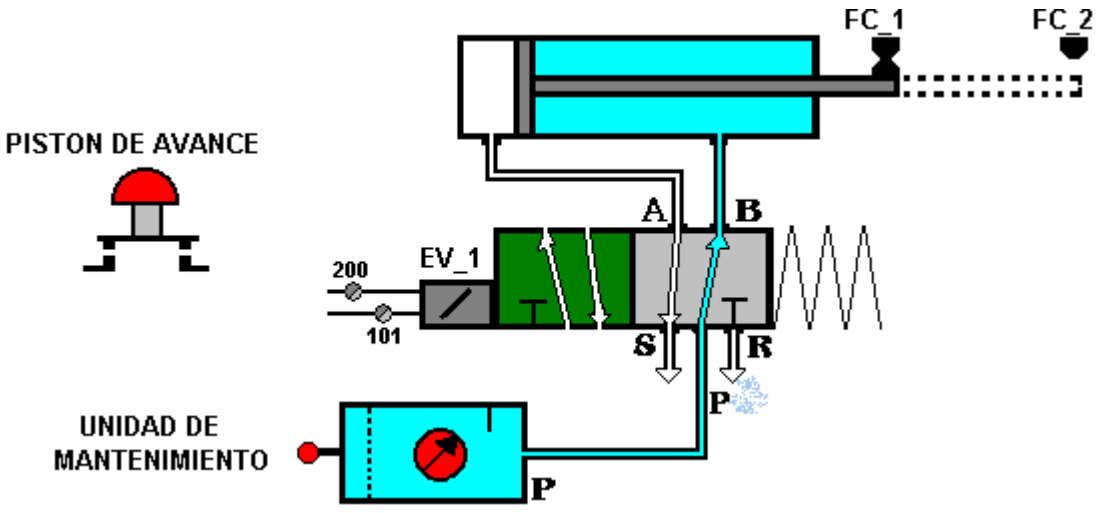
```
|ELECTRO                        |
|VALVULA                        |
|EV1_P1                         |
+--] [-------------------------+
|
|         FC2      FC1
|ELECTRO PISTON1  PISTON1
|VALVULA ADELAN.  REPOSO
|EV1_P1   FC2_P1A FC1_P1R +-----+                                        TPO_PEG
+--] [-----] [-----]/[---+ TMR +----------------------------------------( )--
|                         |0.10s|
|                         |     |
|                  CONST -+PV   |
|                  +00030 |     |
|                         +-----+
|                         %R0001
Program: EJEM_8             C:\LM90\EJEM_8                  Block: _MAIN
```

EJEMPLO 4.17.8

En el ejemplo anterior, se realizó la protección requerida pero, ahora, se presentó el siguiente problema, el operador tiene que estar utilizando las dos manos y éstas, separadas de la maquina como máxima protección, pero, sucede que como nadie lo supervisa y es muy cansado el estar oprimiendo los dos botones, este (el operador) deja activado un botón con cinta adhesiva y trabaja con un solo botón, provocando de nuevo el accidente.

Diseñe una protección para que si el operador activa un botón y después de un segundo oprime el segundo botón no se active el pistón, con lo cual nos aseguramos de que el operador utilizara las dos manos y retiradas de la maquina.

SOLUCIÓN:

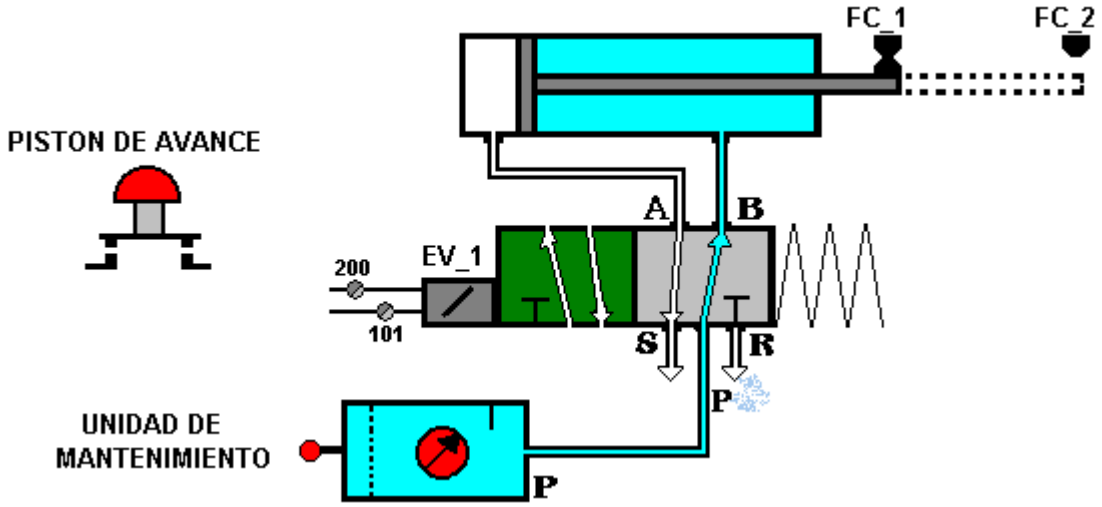


```
|[  START OF LD  PROGRAM EJEMP_9  ]     (*                                *)
|[     VARIABLE DECLARATIONS      ]
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        PB_DER       BOTON DERECHO
          %I0002        PB_IZQ       BOTON IZQ.
          %I0003        FC1_P1R      FC1 PISTON1 REPOSO
          %I0004        FC2_P1A      FC2 PISTON1 ADELAN.
          %Q0001        EV1_P1       ELECTRO VALVULA
          %T0001        TPO_PEG      TIEMPO PEGADO
          %T0002        TPO_SEG      TIEMPO SEG_MAN
| << RUNG 4  STEP #0001 >>
|
|BOTON
|DERECHO
|PB_DER          +-----+                                          TPO_SEG
+--] [--+--------+ TMR +-----------------------------------------------( )--
|       |        |0.10s|
|BOTON  |        |     |
|IZQ.   |        |     |
|PB_IZQ |        |     |
+--] [--+ CONST -+PV   |
|       +00010 |     |
|              +-----+
|              %R0004
| << RUNG 5  STEP #0005 >>
|BOTON   BOTON   TIEMPO
|DERECHO IZQ.    SEG_MAN
|PB_DER  PB_IZQ  TPO_SEG                                          %M0001
+--] [-----] [-----]/[-----------------------------------------------( )--
| << RUNG 6  STEP #0009 >>
|        FC1     FC2
|        PISTON1 PISTON1 TIEMPO
|        REPOSO  ADELAN. PEGADO
|%M0001  FC1_P1R FC2_P1A TPO_PEG                                  EV1_P1
+--] [-----] [-----]/[--+--]/[-----------------------------------------( )--
|                       |
|ELECTRO                |
|VALVULA                |
|EV1_P1                 |
+--] [-----------------+
| << RUNG 7  STEP #0015 >>
|        FC2     FC1
|ELECTRO PISTON1 PISTON1
|VALVULA ADELAN. REPOSO
|EV1_P1  FC2_P1A FC1_P1R +-----+                                  TPO_PEG
+--] [-----] [-----]/[---+ TMR +-----------------------------------( )--
|                        |0.10s|
|                        |     |
|                 CONST -+PV   |
|                 +00030 |     |
|                        +-----+
|
|                        %R0001
|[      END OF PROGRAM LOGIC      ]
```

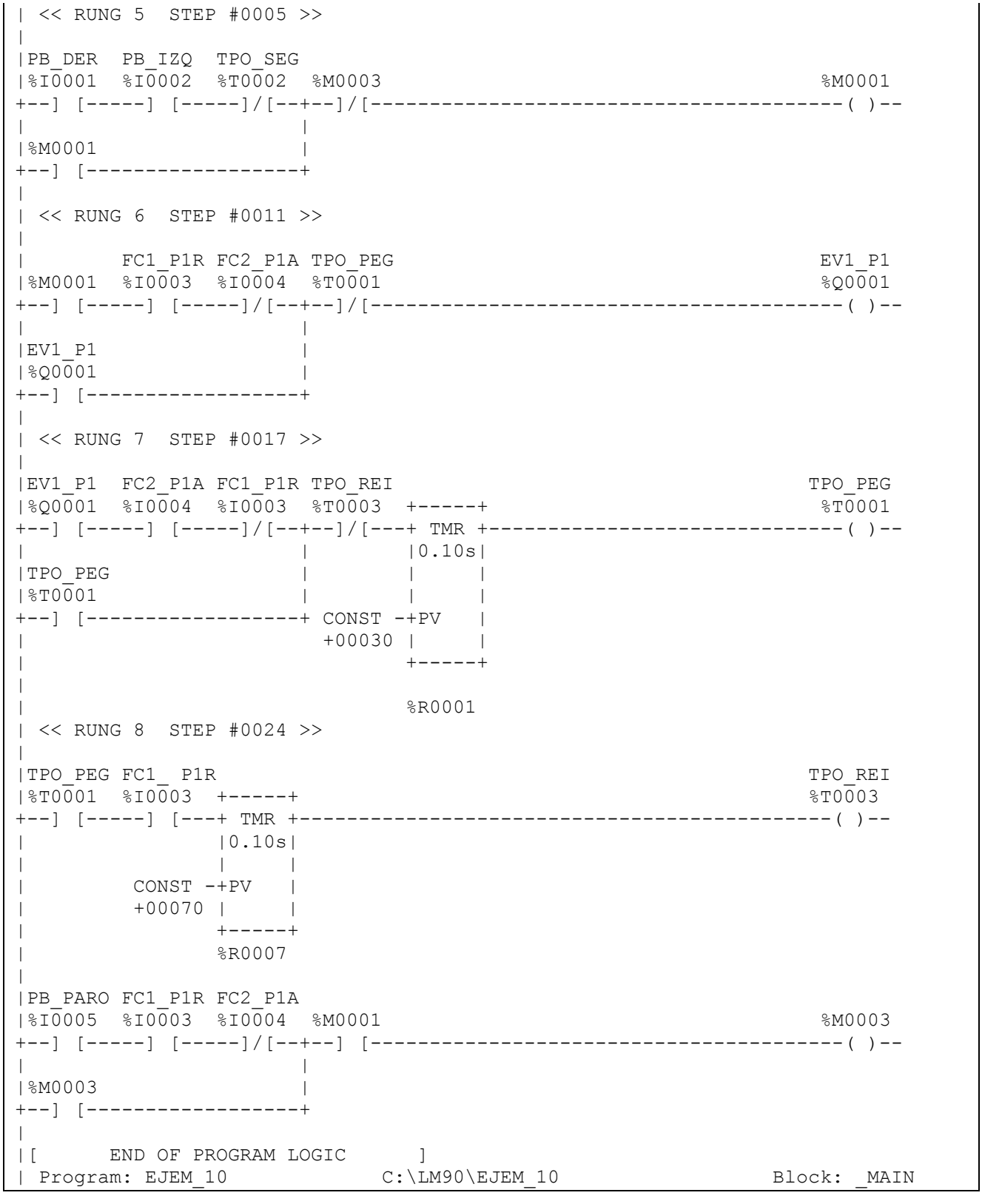
EJEMPLO 4.17.9

Ahora se requiere ponerle un alimentador de piezas en automático, por lo que se necesita que al llegar al reposo dure un tiempo y el pistón baje de nuevo. Para parar el circuito coloque un botón de paro de tal manera que en cualquier momento en que se oprima el botón de paro continúe el ciclo y al llegar a la posición de reposo se pare.

SOLUCIÓN:




```
|[  START OF LD  PROGRAM EJEM_10  ]       (*                          *)
|[     VARIABLE DECLARATIONS      ]
              V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     -------------------------------
          %I0001        PB_DER       BOTON DERECHO
          %I0002        PB_IZQ       BOTON IZQ.
          %I0003        FC1_P1R      FC1 PISTON1 REPOSO
          %I0004        FC2_P1A      FC2 PISTON1 ADELAN.
          %I0005        PB_PARO      BOTON PARO CICLO
          %Q0001        EV1_P1       ELECTRO VALVULA
          %M0001                     CIC_INI CIADO
          %M0003                     PARO CICLO
          %T0001        TPO_PEG      TIEMPO PEGADO
          %T0002        TPO_SEG      TIEMPO SEG_MAN
          %T0003        TPO_REI      TIEMPO REI_CIC
                        I D E N T I F I E R    T A B L E
         IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
         ----------     ---------------     -------------------------------
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|PB_ DER                                                          TPO_SEG
|%I0001          +-----+                                           %T0002
+--] [--+--------+ TMR +-----------------------------------------------( )--
|       |        |0.10s|
|PB_IZQ |        |     |
|%I0002 |        |     |
+--] [--+ CONST -+PV   |
|       +00010  |     |
|               +-----+
|               %R0004
```

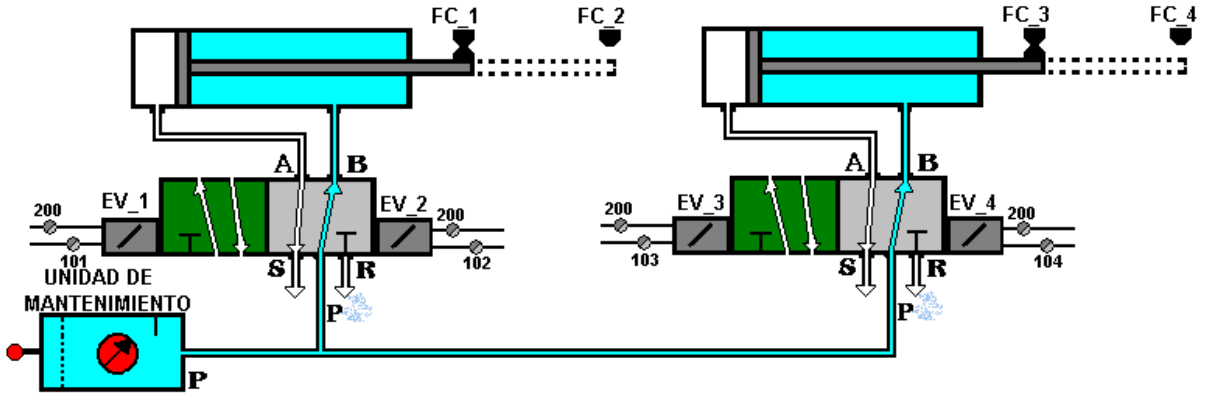

| << RUNG 5 STEP #0005 >>
|PB_DER PB_IZQ TPO_SEG
|%I0001 %I0002 %T0002 %M0003 %M0001
|%M0001
| << RUNG 6 STEP #0011 >>
| FC1_P1R FC2_P1A TPO_PEG EV1_P1
|%M0001 %I0003 %I0004 %T0001 %Q0001
|EV1_P1
|%Q0001
| << RUNG 7 STEP #0017 >>
|EV1_P1 FC2_P1A FC1_P1R TPO_REI TPO_PEG
|%Q0001 %I0004 %I0003 %T0003 %T0001
|TPO_PEG
|%T0001
CONST -+PV
+00030
0.10s
%R0001
| << RUNG 8 STEP #0024 >>
|TPO_PEG FC1_ P1R TPO_REI
|%T0001 %I0003 %T0003
CONST -+PV
+00070
0.10s
%R0007
|PB_PARO FC1_P1R FC2_P1A
|%I0005 %I0003 %I0004 %M0001 %M0003
|%M0003
|[END OF PROGRAM LOGIC   ]
| Program: EJEM_10 C:\LM90\EJEM_10 Block: _MAIN

EJEMPLO 4.17.10

Diseñe el circuito de control electro neumático para que al dar inicio, entre un pistón a sujetar la pieza, cuando está sujeta la pieza, que entre un segundo pistón a barrenar y al llegar al final de su carrera el segundo pistón se regrese y al llegar al reposo se suelte la pieza terminando el ciclo. Además, se deberá tener cuidado de que no repita ciclo y que estén en reposo ambos pistones para poder iniciar.

NOTA: Las electro válvulas tienen accionamiento eléctrico en ambos lados.

SOLUCIÓN:



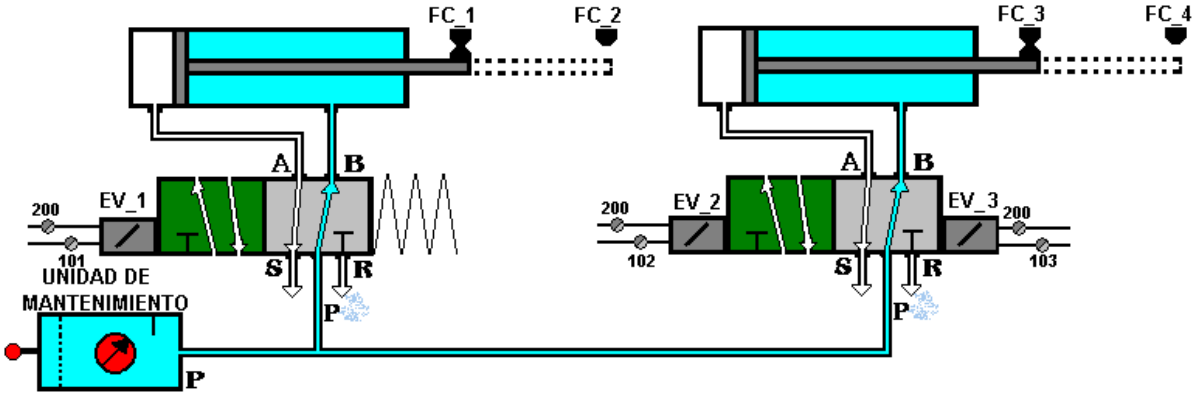

```
|[  START OF LD  PROGRAM EJEM_10  ]    (*                                *)
|[     VARIABLE DECLARATIONS      ]
             V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     -------------------------------
          %I0001        FC1_P1R      FC1 PISTON1 REPOSO
          %I0002        FC2_P1A      FC2 PISTON1 ADELAN.
          %I0003        FC3_P2R      FC3 PISTON2 REPOSO
          %I0004        FC4_P2A      FC4 PISTON2 ADLANTE
          %I0005        PB_INIC      BOTON INICIO
          %Q0001        EV1_P1A      ELECTRO VAL_P1A
          %Q0002        EV2_P1R      ELECTRO VAL_P1R
          %Q0003        EV3_P2A      ELECTRO VAL_P2A
          %Q0004        EV4_P2R      ELECTRO VAL_P2R
          %M0001                     CIC_INI CIADO
          %M0002                     PISTON2 RETORNO
          %M0003        FIN_CIC      FIN_CIC

                        I D E N T I F I E R    T A B L E
        IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
        ----------     ---------------     --------------------------------
          EJEM_10        PROGRAM NAME

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|
|         FC1     FC3
|BOTON    PISTON1 PISTON2                                          CIC_INI
|INICIO   REPOSO  REPOSO  FIN_ CIC                                   CIADO
|%I0005   %I0001  %I0003  %M0003                                     %M0001
+--] [-----] [-----] [--+--]/[-----------------------------------------( )--
```

```
|                      |
|CIC_INI               |
|CIADO                 |
|%M0001                |
+--] [------------------+
|
| << RUNG 5  STEP #0007 >>
|        FC1     FC3
|CIC_INI PISTON1 PISTON2 PISTON2                                  ELECTRO
|CIADO   REPOSO  REPOSO  RETORNO                                  VAL_P1A
|%M0001  %I0001  %I0003  %M0002                                    %Q0001
+--] [-----] [-----] [-----]/[-----------------------------------------( )--
|
| << RUNG 6  STEP #0012 >>
|        FC2     FC3
|CIC_INI PISTON1 PISTON2 PISTON2                                  ELECTRO
|CIADO   ADELAN. REPOSO  RETORNO                                  VAL_P2A
|%M0001  %I0002  %I0003  %M0002                                   %Q0003
+--] [-----] [-----] [-----]/[-----------------------------------------( )--
|
| << RUNG 7  STEP #0017 >>
|        FC2     FC4
|CIC_INI PISTON1 PISTON2 CIC_ INI                                 PISTON2
|CIADO   ADELAN. ADLANTE CIADO                                    RETORNO
|%M0001  %I0002  %I0004  %M0001                                    %M0002
+--] [-----] [-----] [--+--] [-----------------------------------------( )--
|                      |
|PISTON2               |
|RETORNO               |
|%M0002                |
+--] [------------------+
|
| << RUNG 8  STEP #0023 >>
|        FC2     FC4
|CIC_INI PISTON1 PISTON2                                           ELECTRO
|CIADO   ADELAN. ADLANTE                                           VAL_P2R
|%M0001  %I0002  %I0004                                            %Q0004
+--] [-----] [-----] [-------------------------------------------------( )--
|
| << RUNG 9  STEP #0027 >>
|        FC2     FC3
|CIC_INI PISTON1 PISTON2 PISTON2                                  ELECTRO
|CIADO   ADELAN. REPOSO  RETORNO                                  VAL_P1R
|%M0001  %I0002  %I0003  %M0002                                    %Q0002
+--] [-----] [-----] [-----] [-----------------------------------------( )--
|
| << RUNG 10  STEP #0032 >>
|                FC1     FC3
|CIC_INI PISTON2 PISTON1 PISTON2
|CIADO   RETORNO REPOSO  REPOSO                                    FIN_CIC
|%M0001  %M0002  %I0001  %I0003                                     %M0003
+--] [-----] [-----] [-----] [-----------------------------------------( )--
|
|[      END OF PROGRAM LOGIC      ]
Program: EJEM_10             C:\LM90\EJEM_10                Block: _MAIN
```

EJEMPLO 4.17.11

Diseñe el circuito de control electro neumático del ejemplo No 11 pero ahora considere que la electroválvula del pistón sujetador tiene accionamiento eléctrico y retorno por muelle.

SOLUCIÓN:



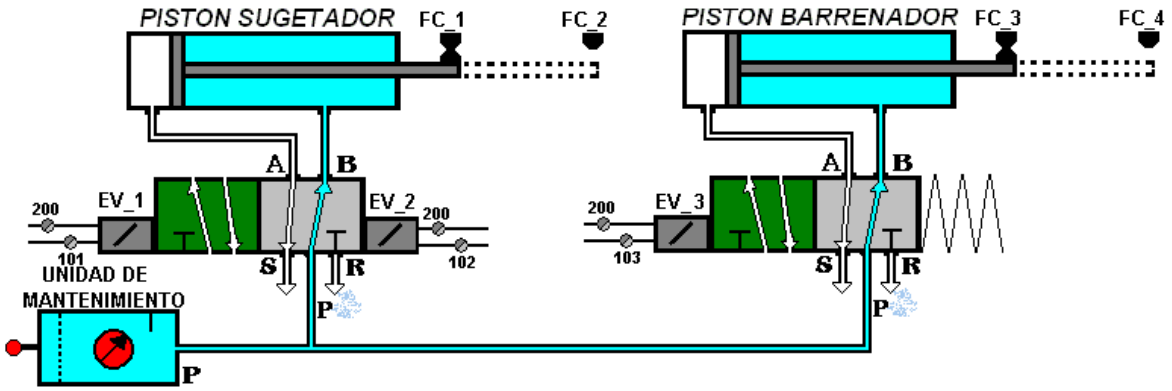

```
|[  START OF LD  PROGRAM EJEM_12  ]       (*                          *)
|[     VARIABLE DECLARATIONS      ]
              V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     ------------------------------
          %I0001        FC1_P1R      FC1_REP PISTON1
          %I0002        FC2_P1A      FC2_ADE PISTON1
          %I0003        FC3_P2R      FC3_REP PISTON2
          %I0004        FC4_P2A      FC4_ADE PISTON2
          %I0005        PB_INIC      BOTON INICIO
          %Q0001        EV1_P1A      ELECTRO VAL_P1A
          %Q0002        EV2_P2R      ELECTRO VAL_P2R
          %Q0003        EV3_P2A      ELECTRO VAL_P2A
          %M0001                     CIC_INI CIADO
          %M0002                     PISTON2 RETORNO
          %M0003        RET_P1       RET_P1
          %M0004        FIN_CIC      FIN CICLO.
                        I D E N T I F I E R    T A B L E
      IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
      ----------     ---------------     ----------------------------
        EJEM_12        PROGRAM NAME         MCS1           MCR

|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|
|BOTON   FC1_REP FC3_REP FIN
|INICIO  PISTON1 PISTON2 CICLO.                                    CIC_INI
|PB_INIC FC1_P1R FC3_P2R FIN_CIC                                    CIADO
|%I0005  %I0001  %I0003  %M0004                                     %M0001
+--] [-----] [-----] [--+--]/[-----------------------------------( )--
|                        |
|CIC_INI                 |
|CIADO                   |
|%M0001                  |
+--] [------------------+
| << RUNG 5  STEP #0007 >>
|
|CIC_INI
```

```
|CIADO
|
|%M0001   MCS1
+--]/[---[ MCR ]
* << RUNG 6  STEP #0009 >>
*FC1_REP FC3_REP ELECTRO
*PISTON1 PISTON2 RET_P1                                          VAL_P1A
*FC1_P1R FC3_P2R RET_P1                                          EV1_P1A
*%I0001  %I0003  %M0003                                          %Q0001
*--] [-----] [--+--]/[-----------------------------------------( )--
*               |
*ELECTRO        |
*VAL_P1A        |
*EV1_P1A        |
*%Q0001         |
*--] [----------+
*
* << RUNG 7  STEP #0014 >>
*FC2_ADE FC3_REP PISTON2                                                  ELECTRO
*PISTON1 PISTON2 RETORNO
*FC2_P1A FC3_P2R EV3_P2A                                        VAL_P2A
*%I0002  %I0003  %M0002                                         %Q0003
*--] [-----] [-----]/[---------------------------------------( )--
*
* << RUNG 8  STEP #0018 >>
*FC2_ADE FC4_ADE
*PISTON1 PISTON2                                               PISTON2
*FC2_P1A FC4_P2A                                               RETORNO
*%I0002  %I0004                                                %M0002
*--] [-----] [--+-----------------------------------------------( )--
*               |
*PISTON2        |
*RETORNO        |
*%M0002         |
*--] [----------+
*
* << RUNG 9  STEP #0022 >>
*        FC2_ADE FC4_ADE                                        ELECTRO
*PISTON2 PISTON1 PISTON2                                        VAL_P2A
*RETORNO FC2_P1A FC4_P2A                                        EV3_P2A
*%M0002  %I0002  %I0004                                         %Q0003
*--] [-----] [-----] [---------------------------------------( )--
*
* << RUNG 10  STEP #0026 >>
*FC2_ADE FC3_REP PISTON2
*PISTON1 PISTON2 RETORNO
*FC2_P1A FC3_P2R RET_P1                                           RET_P1
*%I0002  %I0003  %M0002                                           %M0003
*--] [-----] [-----] [--+-----------------------------------------( )--
*                      |
*RET_P1                |
*RET_P1                |
*%M0003                |
*--] [------------------+
* << RUNG 11  STEP #0031 >>
*
*PISTON2         FC1_REP FC3_REP                                 FIN
*RETORNO RET_P1  PISTON1 PISTON2                                 CICLO.
*RET_P1  RET_P1  FC1_P1R FC3_P2R                                 FIN_CIC
*%M0002  %M0003  %I0001  %I0003                                  %M0004
*--] [-----] [-----] [-----] [-----------------------------------( )--
| << RUNG 12  STEP #0036 >>
| MCS1
+[   ENDMCR    ]
|[      END OF PROGRAM LOGIC      ]
```

EJEMPLO 4.17.12

Diseñe el circuito de control electro neumático del ejemplo No 12 pero ahora considere que la electro válvula del pistón sujetador tiene accionamiento eléctrico en ambos lados y el pistón barrenador tiene accionamiento eléctrico y retorno por muelle.

SOLUCIÓN:



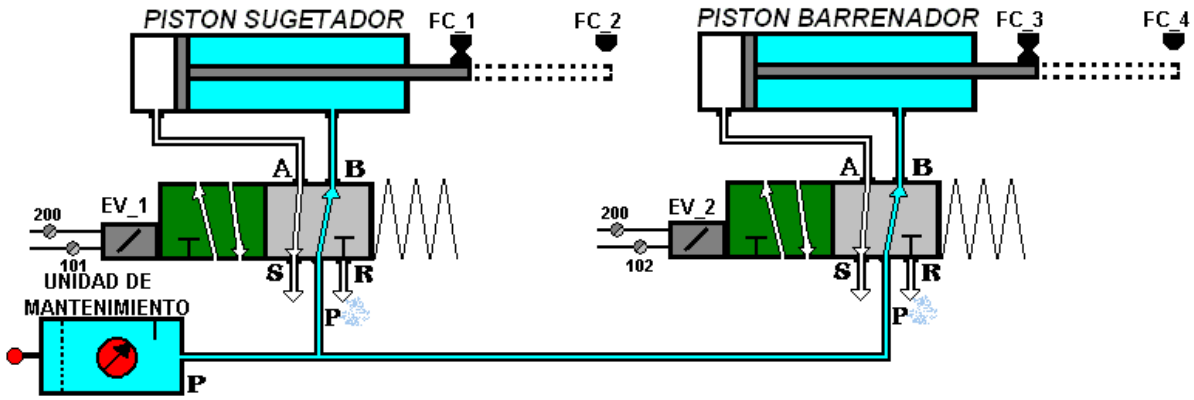

```
|[  START OF LD  PROGRAM EJEM_13  ]       (                          *)
|[     VARIABLE DECLARATIONS      ]
             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        FC1_P1R      FC1_REP PISTON1
          %I0002        FC2_P1A      FC2_ADE PISTON1
          %I0003        FC3_P2R      FC3_REP PISTON2
          %I0004        FC4_P2A      FC4_ADE PISTON2
          %I0005        PB_INIC      BOTON INICIO
          %Q0001        EV1_P1A      ELECTRO VAL_P1A
          %Q0002        EV2_P1R      ELECTRO VAL_P1R
          %Q0003        EV3_P2A      ELECTRO VAL_P2A
          %M0001                     CIC_INI CIADO
          %M0002        RET_P2       RETORNO P2
          %M0003        FI_CIC       FIN CICLO

                        I D E N T I F I E R    T A B L E

    IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
    ----------     ---------------     --------------------------------
    EJEM_13        PROGRAM NAME
          MCS1           MCR
|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|
|BOTON   FC1_REP FC3_REP FIN
|INICIO  PISTON1 PISTON2 CICLO
|PB_INIC FC1_P1R FC3_P2R FI_CIC                                    %M0001
+--] [-----] [-----] [--+--]/[-------------------------------------( )--
|                        |
|CIC_INI                 |
|CIADO                   |
|%M0001                  |
+--] [------------------+
```

```
|
| << RUNG 5  STEP #0007 >>
|
|CIC_INI
|CIADO
|%M0001   MCS1
+--]/[---[ MCR ]
|
* << RUNG 6  STEP #0009 >>
*
*FC1_REP FC3_REP RETORNO
*PISTON1 PISTON2 P2
*FC1_P1R FC3_P2R RET_P2                                          EV1_P1A
*--] [-----] [-----]/[-----------------------------------------( )--
*
* << RUNG 7  STEP #0013 >>
*
*FC2_ADE FC3_REP RETORNO
*PISTON1 PISTON2 P2
*FC2_P1A FC3_P2R RET_P2                                          EV3_P2A
*--] [-----] [--+--]/[-----------------------------------------( )--
*              |
*ELECTRO        |
*VAL_P2A        |
*EV3_P2A        |
*--] [----------+
*
* << RUNG 8  STEP #0018 >>
*
*FC2_ADE FC4_ADE
*PISTON1 PISTON2
*FC2_P1A FC4_P2A                                                  RET_P2
*--] [-----] [--+-------------------------------------------------( )--
*              |
*RETORNO        |
*P2             |
*RET_P2         |
*--] [----------+
*
* << RUNG 9  STEP #0022 >>
*
*FC2_ADE FC3_REP RETORNO
*PISTON1 PISTON2 P2
*FC2_P1A FC3_P2R RET_P2                                           EV2_P1R
*--] [-----] [-----] [-------------------------------------------( )--
*
* << RUNG 10  STEP #0026 >>
*
*
*RETORNO FC1_REP FC3_REP
*P2      PISTON1 PISTON2
*RET_P2  FC1_P1R FC3_P2R                                            FI_CIC
*--] [-----] [-----] [-----------------------------------------------( )--
*
| << RUNG 11  STEP #0030 >>
|
| MCS1
+[   ENDMCR    ]
|
|[      END OF PROGRAM LOGIC      ]
Program: EJEM_13                C:\LM90\EJEM_13                 Block: _MAIN
```

EJEMPLO 4.17.13

Diseñe el circuito de control electro neumático del ejemplo No 13 pero ahora considere que ambas electro válvulas tienen accionamiento eléctrico y retorno por muelle.

SOLUCIÓN:



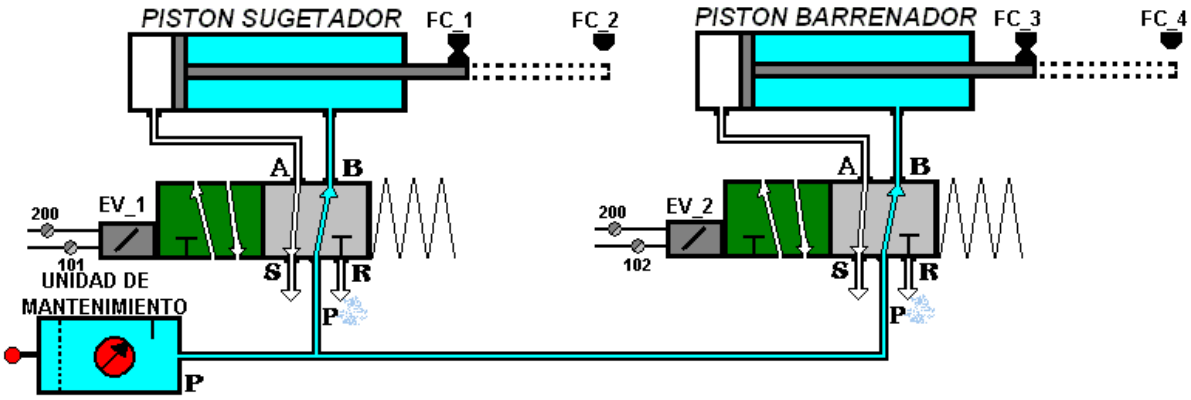

```
|[  START OF LD  PROGRAM EJEM_14  ]     (*                          *)
|[     VARIABLE DECLARATIONS      ]
            V A R I A B L E   D E C L A R A T I O N   T A B L E
         REFERENCE     NICKNAME          REFERENCE DESCRIPTION
         ---------     --------     -------------------------------
         %I0001        FC1_P1R      FC1_REP PISTON1
         %I0002        FC2_P1A      FC2_ADE PISTON1
         %I0003        FC3_P2R      FC3_REP PISTON2
         %I0004        FC4_P2A      FC4_ADE PISTON2
         %I0005        PB_INIC      BOTON INICIO
         %Q0001        EV1_P1A      ELECTRO VAL_P1A
         %Q0002        EV2_P2A      ELECTRO VAL_P2A
         %M0001                     CIC_INI CIADO
         %M0002        RET_P2       RETORNO P2
         %M0003        RET_P1       RETORNO P1
         %M0004        FIN_CIC      FIN_CIC
                       I D E N T I F I E R    T A B L E
IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
      ----------     ---------------     --------------------------------
       EJEM_14        PROGRAM NAME
          MCS1           MCR

|[       BLOCK DECLARATIONS       ]
|
|[     START OF PROGRAM LOGIC     ]
|
| << RUNG 4  STEP #0001 >>
|
|BOTON   FC1_REP FC3_REP
|INICIO  PISTON1 PISTON2 FIN_CIC
|PB_INIC FC1_P1R FC3_P2R FIN_CIC                                    %M0001
+--] [-----] [-----] [--+--]/[-------------------------------------( )--
|                       |
|CIC_INI                |
|CIADO                  |
|%M0001                 |
+--] [------------------+
|
| << RUNG 5  STEP #0007 >>
```

```
|CIC_INI
|CIADO
|%M0001   MCS1
+--]/[---[ MCR ]
|
* << RUNG 6  STEP #0009 >>
*
*FC1_REP FC3_REP RETORNO
*PISTON1 PISTON2 P1
*FC1_P1R FC3_P2R RET_P1                                          EV1_P1A
*--] [-----] [--+--]/[-----------------------------------------------( )--
*               |
*ELECTRO        |
*VAL_P1A        |
*EV1_P1A        |
*--] [----------+
*
* << RUNG 7  STEP #0014 >>
*
*FC2_ADE FC3_REP RETORNO
*PISTON1 PISTON2 P2
*FC2_P1A FC3_P2R RET_P2                                         EV2_P2A
*--] [-----] [--+--]/[-----------------------------------------------( )--
*               |
*ELECTRO        |
*VAL_P2A        |
*EV2_P2A        |
*--] [----------+
*
* << RUNG 8  STEP #0019 >>
*FC2_ADE FC4_ADE
*PISTON1 PISTON2
*FC2_P1A FC4_P2A                                                 RET_P2
*--] [-----] [--+-------------------------------------------------( )--
*               |
*RETORNO        |
*P2             |
*RET_P2         |
*--] [----------+
*
* << RUNG 9  STEP #0023 >>
*FC2_ADE FC3_REP RETORNO
*PISTON1 PISTON2 P2
*FC2_P1A FC3_P2R RET_P2                                          RET_P1
*--] [-----] [-----] [--+-----------------------------------------( )--
*                      |
*RETORNO                |
*P1                     |
*RET_P1                 |
*--] [------------------+
*
* << RUNG 10  STEP #0028 >>
*RETORNO FC1_REP FC3_REP
*P1      PISTON1 PISTON2
*RET_P1  FC1_P1R FC3_P2R                                         FIN_CIC
*--] [-----] [-----] [-------------------------------------------( )--

| << RUNG 11  STEP #0032 >>
|
| MCS1
+[   ENDMCR    ]
|[      END OF PROGRAM LOGIC      ]
Program: EJEM_14                C:\LM90\EJEM_14             Block: _MAIN
```

EJEMPLO 4.17.14

Diseñe el circuito de control para el ejemplo No 14 pero ahora se desea que al dar inicio de ciclo salga el pistón sujetador y al llegar al final de su carrera se regrese y al llegar a la posición de reposo salga el pistón barrenador y al llegar al final de su carrera se regrese y al llegar a la posición de reposo termine el ciclo.

SOLUCIÓN:



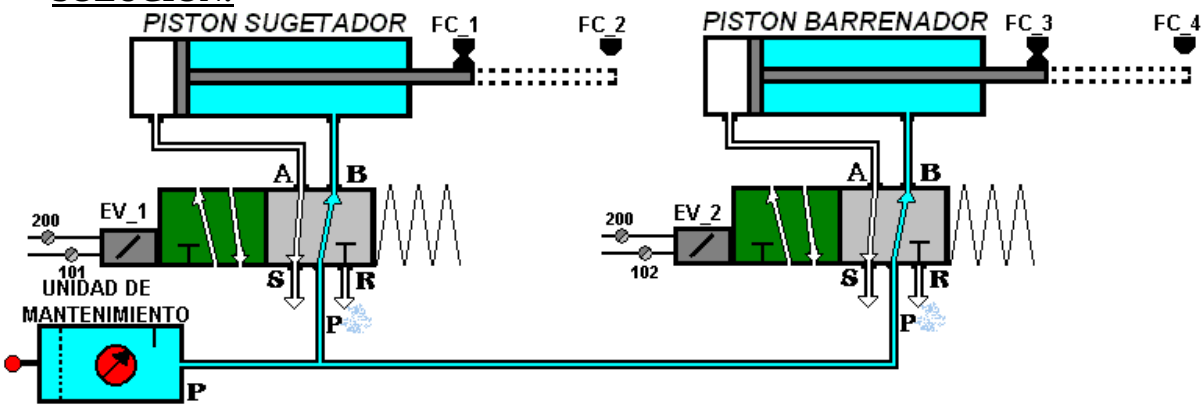

```
|[  START OF LD  PROGRAM EJEM_15  ]       (*                      *)
|[     VARIABLE DECLARATIONS      ]
            V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     ------------------------------
          %I0001        FC1_P1R      FC1_REP PISTON1
          %I0002        FC2_P1A      FC2_ADE PISTON1
          %I0003        FC3_P2R      FC3_REP PISTON2
          %I0004        FC4_P2A      FC4_ADE PISTON2
          %I0005        PB_INIC      BOTON INICIO
          %Q0001        EV1_P1A      ELECTRO VAL_P1A
          %Q0002        EV2_P2A      ELECTRO VAL_P2A
          %M0001        CIC_INI      CICLO INICIAD
          %M0002        RET_P1       RET_P1
          %M0003        RET_P2       RET_P2
          %M0004        FIN_CIC      FIN_CIC

                        I D E N T I F I E R    T A B L E

          IDENTIFIER     IDENTIFIER TYPE        IDENTIFIER DESCRIPTION
          ----------     ---------------     --------------------------
          EJEM_15        PROGRAM NAME
          MCS1           MCR
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|
|BOTON   FC1_REP FC3_REP
|INICIO  PISTON1 PISTON2 FIN_CIC
|PB_INIC FC1_P1R FC3_P2R FIN_CIC                              CIC_INI
+--] [-----] [-----] [--+--]/[-----------------------------------( )--
|                       |
|CICLO                  |
|INICIAD                |
|CIC_INI                |
+--] [------------------+
|
| << RUNG 5  STEP #0007 >>
|
|CICLO
```

```
|INICIAD
|CIC_INI  MCS1
+--]/[---[ MCR ]
|
* << RUNG 6  STEP #0009 >>
*
*FC1_REP FC3_REP
*PISTON1 PISTON2 RET_P1
*FC1_P1R FC3_P2R RET_P1                                          EV1_P1A
*--] [-----] [--+--]/[-----------------------------------------( )--
*               |
*ELECTRO        |
*VAL_P1A        |
*EV1_P1A        |
*--] [----------+
*
* << RUNG 7  STEP #0014 >>
*
*FC2_ADE FC3_REP
*PISTON1 PISTON2
*FC2_P1A FC3_P2R                                                 RET_P1
*--] [-----] [--+-------------------------------------------------( )--
*               |
*RET_P1         |
*RET_P1         |
*--] [----------+
*
* << RUNG 8  STEP #0018 >>
*
*FC1_REP FC3_REP
*PISTON1 PISTON2 RET_P1  RET_P2
*FC1_P1R FC3_P2R RET_P1  RET_P2                                  EV2_P2A
*--] [-----] [-----] [--+--]/[-----------------------------------( )--
*                       |
*ELECTRO                |
*VAL_P2A                |
*EV2_P2A                |
*--] [------------------+
*
* << RUNG 9  STEP #0024 >>
*
*FC1_REP FC4_ADE
*PISTON1 PISTON2 RET_P1
*FC1_P1R FC4_P2A RET_P1                                           RET_P2
*--] [-----] [-----] [--+-------------------------------------------( )--
*                       |
*RET_P2                 |
*RET_P2                 |
*--] [------------------+
*
* << RUNG 10  STEP #0029 >>
*
*        FC1_REP FC3_REP
*RET_P2  PISTON1 PISTON2
*RET_P2  FC1_P1R FC3_P2R                                         FIN_CIC
*--] [-----] [-----] [-------------------------------------------( )--
*
| << RUNG 11  STEP #0033 >>
|
| MCS1
+[   ENDMCR    ]
|[      END OF PROGRAM LOGIC      ]
Program: EJEM_15                C:\LM90\EJEM_15             Block: _MAIN
```

EJEMPLO 4.17.15

Diseñe el circuito de control del ejercicio No 15 pero ahora se desea que al terminar el ciclo, esto es, que al llegar el pistón sujetador al reposo dure un tiempo y reinicie el ciclo. Para parar el ciclo coloque un botón de paro y que pare al terminar el ciclo.

SOLUCIÓN:




```
|[  START OF LD  PROGRAM EJEM_17  ]       (*                                  *)
|[     VARIABLE DECLARATIONS      ]
             V A R I A B L E   D E C L A R A T I O N   T A B L E

          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     --------------------------------
          %I0001        FC1_P1R      FC1_REP PISTON1
          %I0002        FC2_P1A      FC2_ADE PISTON1
          %I0003        FC3_P2R      FC3_REP PISTON2
          %I0004        FC4_P2A      FC4_ADE PISTON2
          %I0005        PB_INIC      BOTON INICIO
          %I0006        PB_PARO      PB_PARO
          %Q0001        EV1_P1A      ELECTRO VAL_P1A
          %Q0002        EV2_P2A      ELECTRO VAL_P2A
          %M0001        CIC_INI      CICLO INICIAD
          %M0002        RET_P1       RET_P1
          %M0003        RET_P2       RET_P2
          %M0004        FIN_CIC      FIN_CIC
          %M0005        PAR_CIC      PAR_CIC
          %T0001        T_REI_C      TIEMPO REI_CIC
                        I D E N T I F I E R    T A B L E
        IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
        ----------     ---------------     ----------------------------
          EJEM_17        PROGRAM NAME
          MCS1           MCR
|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|BOTON   FC1_REP FC3_REP
|INICIO  PISTON1 PISTON2 FIN_CIC
|PB_INIC FC1_P1R FC3_P2R FIN_CIC                                   CIC_INI
+--] [-----] [-----] [--+--]/[-------------------------------------( )--
|                       |
|CICLO                  |
|INICIAD                |
|CIC_INI                |
+--] [------------------+
| << RUNG 5  STEP #0007 >>
|CICLO
|INICIAD
|CIC_INI  MCS1
```

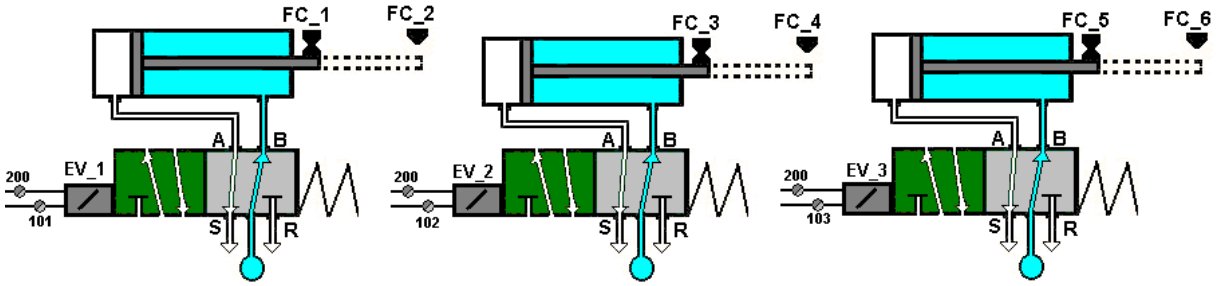
```
+--]/[---[ MCR ]
 *FC1_REP FC3_REP
 *PISTON1 PISTON2 RET_P1
*FC1_P1R FC3_P2R RET_P1                                          EV1_P1A
*--] [-----] [--+--]/[-------------------------------------------( )--
*               |
*ELECTRO        |
*VAL_P1A        |
*EV1_P1A        |
*--] [----------+
* << RUNG 7  STEP #0014 >>
*FC2_ADE FC3_REP TIEMPO
*PISTON1 PISTON2 REI_CIC
*FC2_P1A FC3_P2R T_REI_C                                        RET_P1
*--] [-----] [--+--]/[-------------------------------------------( )--
*               |
*RET_P1         |
*RET_P1         |
*--] [----------+
* << RUNG 8  STEP #0019 >>
*FC1_REP FC3_REP
*PISTON1 PISTON2 RET_P1  RET_P2
*FC1_P1R FC3_P2R RET_P1  RET_P2                                  EV2_P2A
*--] [-----] [-----] [--+--]/[-------------------------------------( )--
*ELECTRO                |
*VAL_P2A                |
*EV2_P2A                |
*--] [------------------+
* << RUNG 9  STEP #0025 >>
*FC1_REP FC4_ADE         TIEMPO
*PISTON1 PISTON2 RET_P1  REI_CIC
*FC1_P1R FC4_P2A RET_P1  T_REI_C                                 RET_P2
*--] [-----] [-----] [--+--]/[-------------------------------------( )--
*                       |
*RET_P2                 |
*RET_P2                 |
*--] [------------------+
*
* << RUNG 10  STEP #0031 >>
*        FC1_REP FC3_REP
*RET_P2  PISTON1 PISTON2
*RET_P2  FC1_P1R FC3_P2R +-----+                                  T_REI_C
*--] [-----] [-----] [---+ TMR +-------------------------------------( )--
*                       |0.10s|
*                       |     |
*                 CONST -+PV   |
*                 +00050 |     |
*                        +-----+
*                        %R0001
* << RUNG 11  STEP #0036 >>
*PB_PARO
*PB_PARO                                                         PAR_CIC
*--] [--+-----------------------------------------------------------( )--
*PAR_CIC|
*PAR_CIC|
*--] [--+
* << RUNG 12  STEP #0039 >>
*                FC1_REP FC3_REP
*PAR_CIC RET_P2  PISTON1 PISTON2
*PAR_CIC RET_P2  FC1_P1R FC3_P2R                                 FIN_CIC
*--] [-----] [-----] [-----] [-------------------------------------( )--
| << RUNG 13  STEP #0044 >>
| MCS1
+[   ENDMCR    ]
|[      END OF PROGRAM LOGIC      ]
```

EJEMPLO 4.17.16

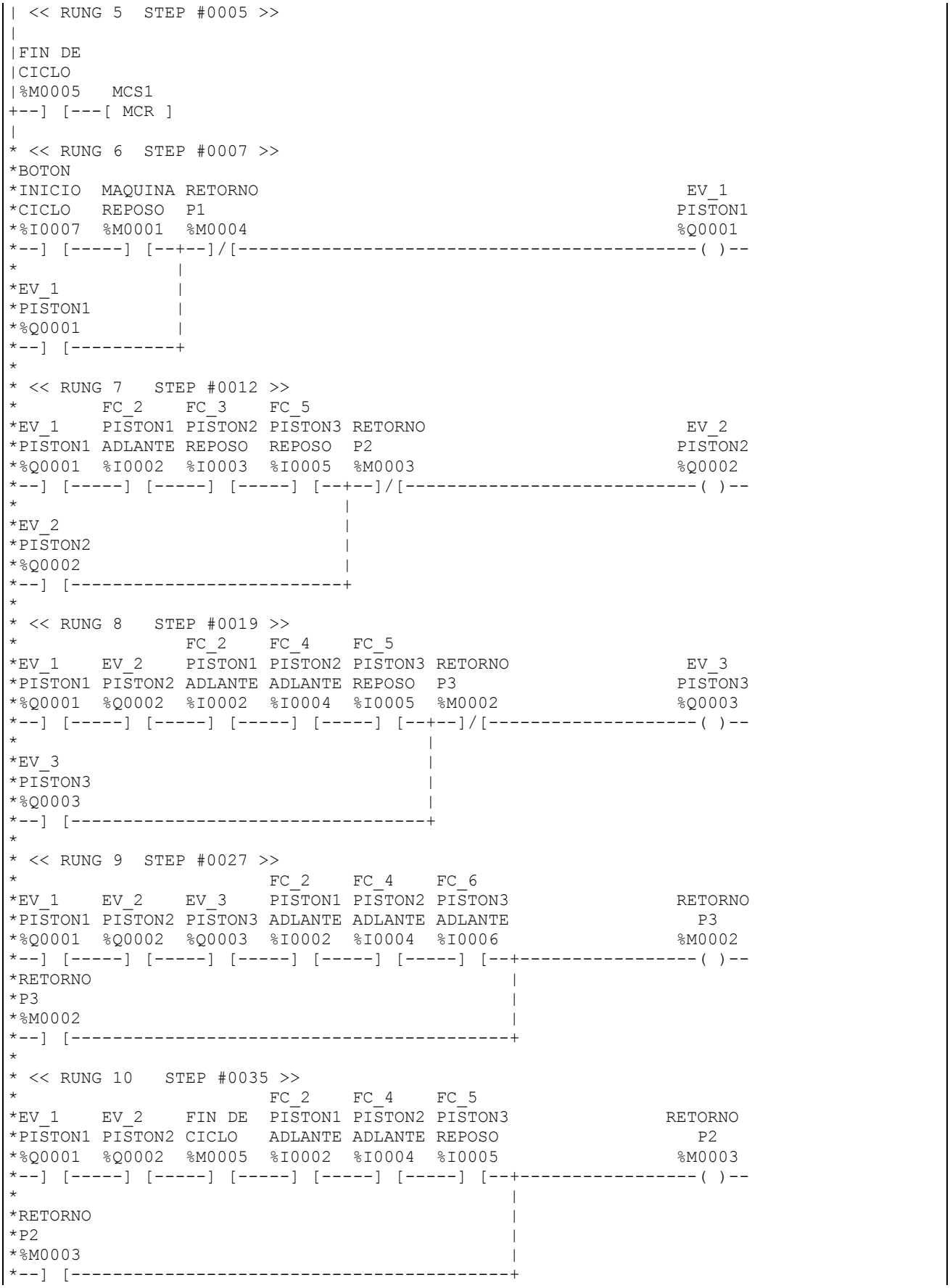
Diseñe el circuito de control electro neumático para tres pistones los cuales deberán trabajar en forma de contador Johnson, esto es, al dar inicio de ciclo sale el primer pistón y al llegar al final de su carrera sale el segundo pistón y al llegar al final de su carrera el segundo pistón, sale el tercer pistón y al llegar al final de su carrera el tercer pistón se regresa el tercer pistón y al llegar a la posición de reposo el tercer pistón, regresa el segundo pistón y al llegar el segundo pistón a la posición de reposo se regresa el primer pistón y al llegar a la posición de reposo el primer pistón termina el ciclo quedando todo en condiciones de iniciar de nuevo el ciclo.

NOTA: Las condiciones para dar inicio de ciclo son que los tres pistones estén en reposo.

SOLUCIÓN:




```
| [  START OF LD  PROGRAM CUR_13   ]       (*                        *)
| [     VARIABLE DECLARATIONS      ]
             V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     -------------------------------
          %I0001        FC1_P1R      FC_1 PISTON1 REPOSO
          %I0002        FC2_P1A      FC_2 PISTON1 ADLANTE
          %I0003        FC3_P2R      FC_3 PISTON2 REPOSO
          %I0004        FC4_P2A      FC_4 PISTON2 ADLANTE
          %I0005        FC5_P3R      FC_5 PISTON3 REPOSO
          %I0006        FC6_P3A      FC_6 PISTON3 ADLANTE
          %I0007        PB_INIC      BOTON INICIO CICLO
          %Q0001        EV1_P1       EV_1 PISTON1
          %Q0002        EV2_P2       EV_2 PISTON2
          %Q0003        EV3_P3       EV_3 PISTON3
          %M0001        MAQ_REP      MAQUINA REPOSO
          %M0002        RET_P3       RETORNO P3
          %M0003        RET_P2       RETORNO P2
          %M0004        RET_P1       RETORNO P1
          %M0005        FIN_CIC      FIN DE CICLO

                        I D E N T I F I E R   T A B L E
   IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
   ----------     ---------------     --------------------------------
          CUR_13         PROGRAM NAME
          MCS1           MCR

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4  STEP #0001 >>
|
|FC_1    FC_3    FC_5                                                 MAQUINA
|PISTON1 PISTON2 PISTON3                                              REPOSO
|REPOSO  REPOSO  REPOSO                                               %M0001
|%I0001  %I0003  %I0005                                               %M0001
+--] [-----] [-----] [-----------------------------------------------( )--
|
```

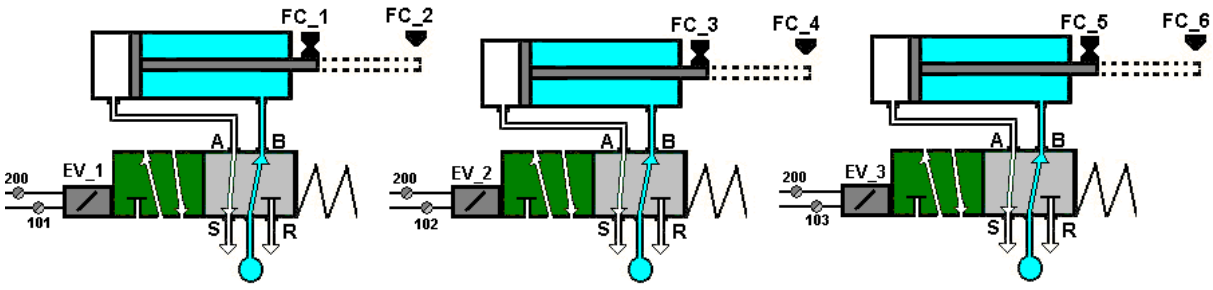

| << RUNG 5 STEP #0005 >>
|FIN DE
|CICLO
|%M0005   MCS1
+--] [---[MCR]
* << RUNG 6 STEP #0007 >>
*BOTON
*INICIO MAQUINA RETORNO EV_1
*CICLO   REPOSO P1 PISTON1
*%I0007 %M0001 %M0004 %Q0001
*--] [-----] [--+--]/[----------( )--
*EV_1
*PISTON1
*%Q0001
*--] [----------+
* << RUNG 7   STEP #0012 >>
* FC_2 FC_3 FC_5
*EV_1 PISTON1 PISTON2 PISTON3 RETORNO EV_2
*PISTON1 ADLANTE REPOSO REPOSO P2 PISTON2
*%Q0001 %I0002 %I0003 %I0005 %M0003 %Q0002
*--] [-----] [-----] [-----] [--+--]/[----------( )--
*EV_2
*PISTON2
*%Q0002
*--] [----------+
* << RUNG 8   STEP #0019 >>
* FC_2 FC_4 FC_5
*EV_1 EV_2 PISTON1 PISTON2 PISTON3 RETORNO EV_3
*PISTON1 PISTON2 ADLANTE ADLANTE REPOSO P3 PISTON3
*%Q0001 %Q0002 %I0002 %I0004 %I0005 %M0002 %Q0003
*--] [-----] [-----] [-----] [-----] [--+--]/[----------( )--
*EV_3
*PISTON3
*%Q0003
*--] [----------+
* << RUNG 9 STEP #0027 >>
* FC_2 FC_4 FC_6
*EV_1 EV_2 EV_3 PISTON1 PISTON2 PISTON3 RETORNO
*PISTON1 PISTON2 PISTON3 ADLANTE ADLANTE ADLANTE P3
*%Q0001 %Q0002 %Q0003 %I0002 %I0004 %I0006 %M0002
*--] [-----] [-----] [-----] [-----] [-----] [--+----------( )--
*RETORNO
*P3
*%M0002
*--] [----------+
* << RUNG 10   STEP #0035 >>
* FC_2 FC_4 FC_5
*EV_1 EV_2 FIN DE PISTON1 PISTON2 PISTON3 RETORNO
*PISTON1 PISTON2 CICLO ADLANTE ADLANTE REPOSO P2
*%Q0001 %Q0002 %M0005 %I0002 %I0004 %I0005 %M0003
*--] [-----] [-----] [-----] [-----] [-----] [--+----------( )--
*RETORNO
*P2
*%M0003
*--] [----------+

```
*
* << RUNG 11   STEP #0043 >>
*                  FC_2    FC_3    FC_5
*EV_1     RETORNO  PISTON1 PISTON2 PISTON3                        RETORNO
*PISTON1  P2       ADLANTE REPOSO  REPOSO                          P1
*%Q0001  %M0003  %I0002  %I0003  %I0005                          %M0004
*--] [-----] [-----] [-----] [-----] [--+-------------------------( )--
*                                      |
*RETORNO                               |
*P1                                    |
*%M0004                                |
*--] [----------------------------------+
*
* << RUNG 12  STEP #0050 >>
*RETORNO MAQUINA                                                  FIN DE
*P1      REPOSO                                                    CICLO
*%M0004  %M0001                                                  %M0005
*--] [-----] [-----------------------------------------------------( )--
*
| << RUNG 13    STEP #0053 >>
| MCS1
+ [   ENDMCR    ]
| [      END OF PROGRAM LOGIC      ]
```

EJEMPLO 4.17.17

Diseñe el circuito de control electro neumático para el circuito del problema 19 pero ahora invertido, esto es, al dar inicio de ciclo sale el primer pistón, al llegar el final de su carrera sale el segundo y al llegar al final de su carrera el segundo sale el tercero y al llegar al final de su carrera se regresa el primero y al llegar a la posición de reposo se regresa el segundo y al llegar a la posición de reposo se regresa el tercero y al llegar al reposo el tercero se termina el ciclo. Se aplican las mismas condiciones de inicio de ciclo.

SOLUCIÓN:



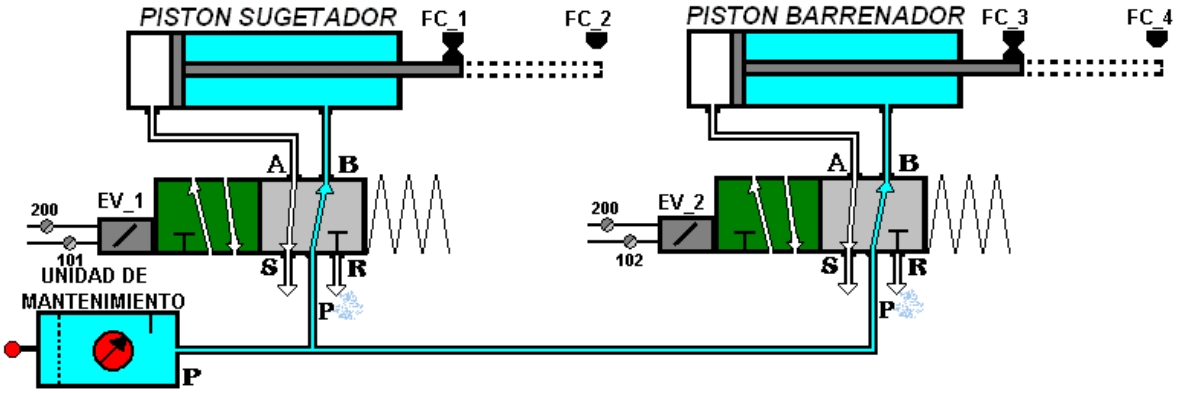
```
|[  START OF LD  PROGRAM CUR_14   ]       (*                      *)
|[     VARIABLE DECLARATIONS      ]

            V A R I A B L E   D E C L A R A T I O N   T A B L E
         REFERENCE     NICKNAME          REFERENCE DESCRIPTION
         ---------     --------     -------------------------------
         %I0001        FC1_P1R      FC_1 PISTON1 REPOSO
         %I0002        FC2_P1A      FC_2 PISTON1 ADLANTE
         %I0003        FC3_P2R      FC_3 PISTON2 REPOSO
         %I0004        FC4_P2A      FC_4 PISTON2 ADLANTE
         %I0005        FC5_P3R      FC_5 PISTON3 REPOSO
         %I0006        FC6_P3A      FC_6 PISTON3 ADLANTE
         %I0007        PB_INIC      BOTON INICIO CICLO
         %Q0001        EV1_P1       EV_1 PISTON1
         %Q0002        EV2_P2       EV_2 PISTON2
         %Q0003        EV3_P3       EV_3 PISTON3
         %M0001        MAQ_REP      MAQUINA REPOSO
         %M0002        RET_P1       RETORNO P1
         %M0003        RET_P2       RETORNO P2
         %M0004        RET_P3       RETORNO P3
         %M0005        FIN_CIC      FIN DE CICLO

                       I D E N T I F I E R    T A B L E
   IDENTIFIER     IDENTIFIER TYPE          IDENTIFIER DESCRIPTION
   ----------     ---------------     --------------------------------
         CUR_14          PROGRAM NAME
         MCS1            MCR

| [       BLOCK DECLARATIONS       ]
| [     START OF PROGRAM LOGIC     ]
| << RUNG 4   STEP #0001 >>
|
|FC_1    FC_3    FC_5
|PISTON1 PISTON2 PISTON3
|REPOSO  REPOSO  REPOSO
|FC1_P1R FC3_P2R FC5_P3R                                        MAQ_REP
+--] [-----] [-----] [-----------------------------------------( )--
|
| << RUNG 5   STEP #0005 >>
|
|FIN DE
|CICLO
|FIN_CIC  MCS1
+--] [---[ MCR ]
|
* << RUNG 6  STEP #0007 >>
*
*BOTON
*INICIO  MAQUINA RETORNO
*CICLO   REPOSO  P1
*PB_INIC MAQ_REP RET_P1                                         EV_P1
*--] [-----] [--+--]/[-----------------------------------------( )--
*               |
*EV_1           |
*PISTON1        |
*EV1_P1         |
*--] [----------+
*
* << RUNG 7   STEP #0012 >>
*        FC_2    FC_3    FC_5
*EV_1    PISTON1 PISTON2 PISTON3 RETORNO
*PISTON1 ADLANTE REPOSO  REPOSO  P2
*EV1_P1  FC2_P1A FC3_P2R FC5_P3R RET_P2                        EV_2
*--] [-----] [-----] [-----] [--+--]/[-------------------------( )--
*                            |
```

```
*EV_2                          |
*PISTON2                       |
*EV2_P2                        |
*--] [--------------------------+
*
* << RUNG 8   STEP #0019 >>
*                FC_2    FC_4    FC_5
*EV_1    EV_2    PISTON1 PISTON2 PISTON3 RETORNO
*PISTON1 PISTON2 ADLANTE ADLANTE REPOSO  P3
*EV1_P1  EV2_P2  FC2_P1A FC4_P2A FC5_P3R RET_P3                  EV_3
*--] [-----] [-----] [-----] [-----] [--+--]/[-------------------( )--
*                                       |
*EV_3                                   |
*PISTON3                                |
*EV3_P3                                 |
*--] [----------------------------------+
*
* << RUNG 9   STEP #0027 >>
*                         FC_2    FC_4    FC_6
*EV_1    EV_2    EV_3    PISTON1 PISTON2 PISTON3
*PISTON1 PISTON2 PISTON3 ADLANTE ADLANTE ADLANTE
*EV1_P1  EV2_P2  EV3_P3  FC2_P1A FC4_P2A FC6_P3A                RET_P1
*--] [-----] [-----] [-----] [-----] [-----] [--+----------------( )--
*                                              |
*RETORNO                                       |
*P1                                            |
*RET_P1                                        |
*--] [-----------------------------------------+
*
* << RUNG 10  STEP #0035 >>
*                         FC_1    FC_4    FC_6
*EV_1    EV_2    FIN DE  PISTON1 PISTON2 PISTON3
*PISTON1 PISTON2 CICLO   REPOSO  ADLANTE ADLANTE
*EV1_P1  EV2_P2  FIN_CIC FC1_P1R FC4_P2A FC6_P3A                RET_P2
*--] [-----] [-----] [-----] [-----] [-----] [--+----------------( )--
*                                              |
*RETORNO                                       |
*P2                                            |
*RET_P2                                        |
*--] [-----------------------------------------+
*
* << RUNG 11  STEP #0043 >>
*
*                FC_1    FC_3    FC_6
*EV_1            PISTON1 PISTON2 PISTON3
*PISTON1         REPOSO  REPOSO  ADLANTE
*EV1_P1  %M0006  FC1_P1R FC3_P2R FC6_P3A                        RET_P3
*--] [-----] [-----] [-----] [-----] [--+------------------------( )--
*                                       |
*RETORNO                                |
*P3                                     |
*RET_P3                                 |
*--] [----------------------------------+
*
* << RUNG 12   STEP #0050 >>
*RETORNO MAQUINA
*P3      REPOSO
*RET_P3  MAQ_REP                                                FIN_CIC
*--] [-----] [---------------------------------------------------( )--
| << RUNG 13   STEP #0053 >>
| MCS1
[   ENDMCR    ]
| [      END OF PROGRAM LOGIC      ]
|
```

EJEMPLO 4.17.18

Se quiere automatizar el proceso de encorchado de botellas, para esto el operador debe colocar manualmente el corcho en la boca de la botella, una vez hecho esto se oprime el botón de inicio de ciclo y sale un pistón para sujetar la pieza, cuando la pieza esta fija, baja un segundo pistón y golpea el corcho tres veces consecutivas para que entre completamente, una vez hecho esto el pistón sujetador regresa y suelta la pieza finalizando el ciclo.

SOLUCIÓN:




```
            V A R I A B L E   D E C L A R A T I O N   T A B L E
          REFERENCE     NICKNAME          REFERENCE DESCRIPTION
          ---------     --------     -------------------------------
          %I0001        FC1_P1R      FC1 PISTON1 REPOSO
          %I0002        FC2_P1A      FC2 PISTON1 ADLANTE
          %I0003        FC3_P2R      FC3 PISTON2 REPOSO
          %I0004        FC4_P2A      FC4 PISTON2 ADLANTE
          %I0005        BOT_INI      BOTON INICIO
          %Q0001        EV1_P1       EV1 PISTON1
          %Q0002        EV2_P2       EV2 PISTON2
          %M0001        CIC_INI      CICLO INICIA DO
          %M0002        RET_P2       RETORNO PISTON2
          %M0003        TER_GOL      TERCER GOLPE
          %M0004        RET_P1       RETORNO PISTON1
          %M0005        FIN          FIN CICLO

|[       BLOCK DECLARATIONS       ]
|[     START OF PROGRAM LOGIC     ]

| << RUNG 4  STEP #0001 >>
*         FC1_P1R  FC3_P2R                                    CICLO
*BOTON  PISTON1  PISTON2                                    INICIA DO
|INICIO  REPOSO  REPOSO FIN                                 CICLO
|%I0005  %I0001  %I0003  %M0005                             %M0001
+--] [-----] [-----] [--+--]/[-------------------------------( )--
|CICLO                  |
|INICIA                 |
|DO                     |
|%M0001                 |
+--] [-----------------+
| << RUNG 5  STEP #0007 >>
|CICLO
|INICIA
|DO
|%M0001   MCS1
+--]/[---[ MCR ]

* << RUNG 6  STEP #0009 >>
```

```
*CICLO
*INICIA  RETORNO                                                EV1
*DO      PISTON1                                               PISTON1
*%M0001  %M0004                                                %Q0001
*--] [-----]/[-------------------------------------------------( )--
*
* << RUNG 7  STEP #0012 >>
*FC2     FC3
*PISTON1 PISTON2 RETORNO TERCER                                 EV2
*ADLANTE REPOSO  PISTON2 GOLPE                                 PISTON2
*%I0002  %I0003  %M0002  %M0003                                %Q0002
*--] [-----] [--+--]/[-----]/[---------------------------------( )--
*EV2            |
*PISTON2        |
*%Q0002         |
*--] [----------+
*
* << RUNG 8  STEP #0018 >>

*FC2     FC4
*PISTON1 PISTON2                                               PISTON2
*ADLANTE ADLANTE                                               RETORNO
*%I0002  %I0004                                                %M0002
*--] [-----] [-------------------------------------------------( )--
*
* << RUNG 9  STEP #0021 >>
* TERCER                                                       RETORNO
* GOLPE                                                        PISTON2
*%M0002  +-----+                                               %M0003
*--] [--->UPCTR+-------------------------------------------------( )--
*CICLO   |     |
*INICIA  |     |
*DO      |     |
*%M0001  |     |
*--]/[---+R    |
*        |     |
* CONST -+PV   |
* +00003 |     |
*        +-----+
*        %R0001
* << RUNG 10  STEP #0025 >>

*        FC3
*TERCER  PISTON2                                               RETORNO
*GOLPE   REPOSO                                                PISTON1
*%M0003  %I0003                                                %M0004
*--] [-----] [-------------------------------------------------( )--
*
* << RUNG 11  STEP #0028 >>

*        FC1     FC3
*TERCER  PISTON1 PISTON2                                         FIN
*GOLPE   REPOSO  REPOSO                                        CICLO
*%M0003  %I0001  %I0003                                        %M0005
*--] [-----] [-----] [---------------------------------------( )--
*
* << RUNG 12  STEP #0032 >>

* MCS1
+[   ENDMCR    ]
|[      END OF PROGRAM LOGIC      ]
```

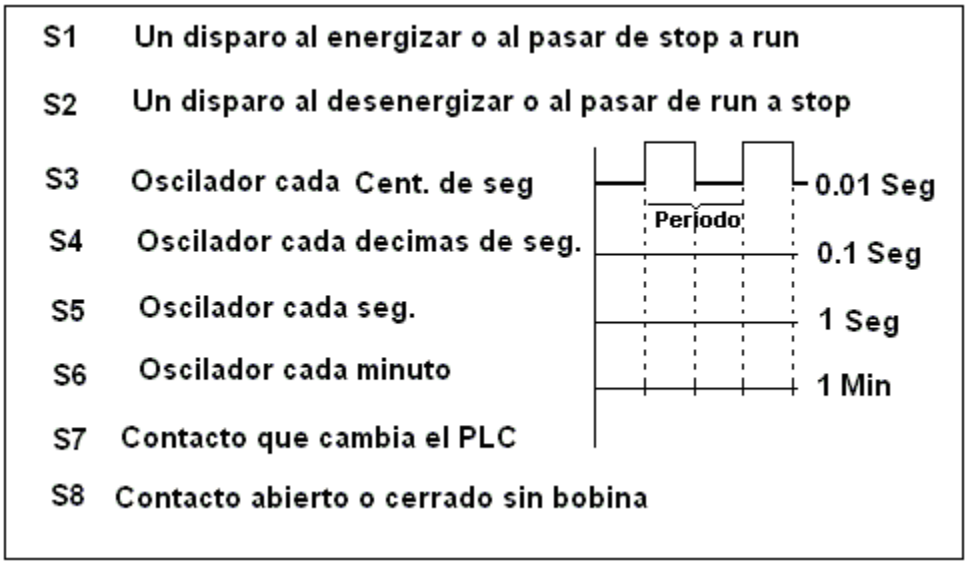
Anexo 1.

MARCA Ge Fanuc.

Teclas de función rápida. Para ver esta tabla se oprimen las teclas **(ALT+K)**

FUNCTION	KEY	FUNCTION	KEY
Abort	Alt-A	Exit Package	Ctrl-Break
Clear Field	Alt-C	Zoom Out	Esc
Change Programmer Mode	Alt-M	Previous Command Line	Ctrl-Home
Change PLC Run/Stop State	Alt-R	Next Command Line	Ctrl-End
Toggle Status Area	Alt-E	Cursor Left Within Field	Ctrl <——
List Directory Files	Alt-L	Cursor Right Within Field	Ctrl ——>
Print Screen	Alt-P	Decrement Reference Address	Ctrl-D
Help	Alt-H	Increment Reference Address	Ctrl-U
Key Help	Alt-K	Change/Incr Field Contents	Tab
Instruction Mnemonic Help	Alt-I	Change/Decr Field Contents	Shift-Tab
Start Teach Mode	Alt-T	Accept Field Contents	Enter
Stop Teach Mode	Alt-Q	Display Last System Error	Ctrl-E
Playback File (n = 0 - 9)	Alt-n	Toggle Discrete Reference	Keypad -
Toggle Discrete Reference	F12	Override Discrete Reference	Keypad *
Override Discrete Refer	F11	Accept Rung	Keypad +
Toggle Text Editor Bell	Alt-B	Accept Rung	Enter
Delete Rung Element	Alt-D	Previous Rung	Ctrl-PgUp
Store Block to PLC & Disk	Alt-S	Next Rung	Ctrl-PgDn
Display Zoom Level	Alt-X	Horizontal Shunt	~
Update Disk	Alt-U	Vertical Shunt	|
Variable Table Window	Alt-V	Go to Next Operand Field	Tab
Go To Operand's Ref. Table	Alt-F2	Toggle Editor Display Mode	Alt-N

Contactos especiales.



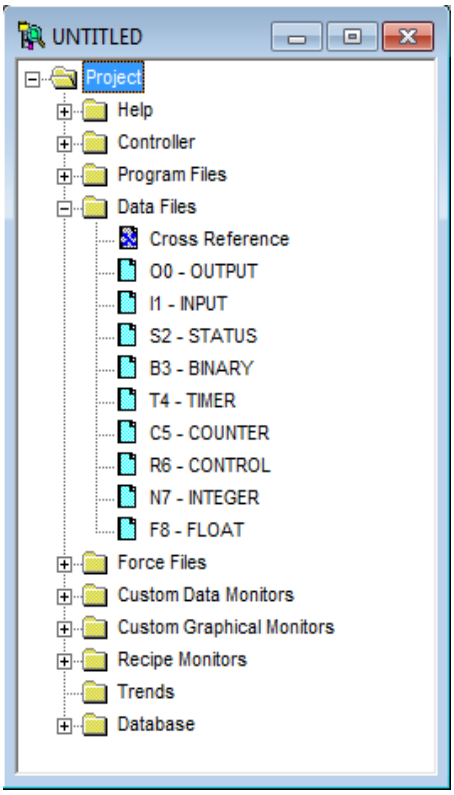
CAPITULO 5
INTRODUCCIÓN A LA PROGRAMACIÓN DE PLCs ALLEN BRADLEY FAMILIA MICROLOGIX 1200

En esta sexta edición de este libro de automatización, presentamos por primera ocasión esta nueva familia de controladores lógicos programables considerando que en la inmensa industria de la automatización podemos encontrar un sin fin de marcas y familias de controladores, cada una con un entorno de programación diferente. Aunque generalmente podemos encontrar un común denominador en el diagrama de escalera, el software de cada marca tiene sus particularidades.

Consideramos que la programación de la serie MicroLogix de Allen Bradley no es la excepción, y por ello hemos incluido un capitulo completo de esta familia de PLC's ya que tiene un entorno de programación reconocido por su gran versatilidad y gran aceptación en el mercado nacional además de que nos permite disponer de muchas herramientas dentro del programa que muchos otros controladores no ofrecen.

El propósito de este tema es introducir al estudiante en el ambiente y las capacidades que nos proporciona en este caso el RSLogix 500 para el PLC MicroLogix 1200. A pesar de ser un software amigable para el usuario y estar basado en Windows OS, es bastante diferente a otros paquetes de programación, por lo cual, es imperativo el hábil manejo de las instrucciones y conocimiento de los distintos tipos de archivo que se pueden manejar.

Archivos de Datos (Data Files).- Estos archivos de datos contienen la información del estado de las instrucciones, salidas y entradas del programa. Son registros de funciones especificas y generales de 16 bits, en el MicroLogix 1200 podemos almacenar hasta un total de 256 elementos (0-255). Las tablas de datos de estos archivos se van generando automáticamente al direccionar cada instrucción, ya sea un solo bit o una palabra de bit completa (16 bits). Al crear un nuevo archivo de esta manera se le asigna un tipo de variable global en el programa, si desea hacer el archivo una variable local debe hacer el cambio manualmente.



Existen 9 tipos de Archivos de Datos, cada uno tiene una función particular dentro del programa. Los datos de estos archivos se pueden arrastrar desde las tablas los mismos hacia las instrucciones o escribiendo sobre la instrucción el parámetro requerido.

Salidas (Output O0).- Este registro de propósito general almacena el estado de las salidas del controlador y de sus módulos externos de salidas conectados a él, ya sean salidas analógicas o digitales. Pueden direccionarse bit por bit o por registro hacia distintos tipos de instrucciones, las cuales veremos más adelante. El número de salidas que tenemos disponibles está delimitado por el modelo de controlador y sus módulos de salidas externas.

Entradas (Inputs I1).- Este registro de propósito general almacena el estado de las entradas del controlador y de sus módulos externos de entradas conectados a él, ya sean entradas analógicas o digitales. Pueden direccionarse bit por bit o por registro hacia distintos tipos de instrucciones. El número de entradas que tenemos disponibles está delimitado por el modelo de controlador y sus módulos de entradas externas.

Estado (Status S2).- En este archivo se tiene acceso a una gran cantidad de información, desde errores, tiempos de escaneo, sobreflujos, depuración, banderas, etc. Refiérase a la carpeta Proyecto ► Database ► Address/Symbol para poder direccionar y/o monitorear algún parámetro que sea de utilidad. En esta sección también puede cambiar tipo de alcance de la variable según convenga; global o local.

Binario (Binary B3).- Estos registros son de gran importancia y generalmente usted tendrá que hacer uso de varios archivos binarios en su diagrama de escalera. Una de las ventajas de trabajar con un PLC es la cantidad de memoria que puede procesar, con estos archivos binarios usted tiene a su disposición 256 registros de 16 bits (MicroLogix 1200), lo que le permite manejar más de 4000 elementos de control para su programa, ya sean contactos de relevadores, bobinas internas y demás instrucciones.

Temporizador (Timer T4).- Es un registro de propósito especifico. La instrucción de temporización siendo vital en la automatización, tiene un registro de control para cada instrucción que requiera implementar. Los parámetros de entrada de esta instrucción; Timer, Time Base, Preset, Accum, se detallaran más adelante.

Al utilizar esta instrucción puede disponer de 3 bits de instrucciones:

13 = Done Bit (DN).- Este bit se establece en “1” lógico cuando el temporizador ha terminado su instrucción, entiéndase por terminado que su acumulador ha llegado al tiempo establecido previamente.

14 = Timer Timing Bit (TT).- Se establece en “1” lógico cuando el temporizador ha empezado su cuenta.

15 = Enable Bit (EN).- Este bit se establece en “1” lógico cuando el temporizador es activado.

Los bits de instrucciones, el preset y el acumulador necesitan una palabra de memoria cada uno.

Contador (Counter C5).- Al igual que el archivo T4, este registro de propósito especifico almacena los valores de cada parámetro del contador que se requiere para su implementación. Estos parámetros son; Counter, Preset, Accum, los cuales se detallaran mas adelante.

Al utilizar esta instrucción puede disponer de 3 bits de instrucciones:

11 = Count Down Overflow Bit (UN)

12 = Count Up Overflow Bit (OV)

13 = Done Bit (DN).- Este bit se establece en "1" lógico cuando el acumulador del contador ha llegado al valor prestablecido.

14 = Count Down Enable Bit (CD).- Este bit se establece en "1" lógico cuando el contador ascendente es activado.

15 = Count Up Enable Bit (CU).- Este bit se establece en "1" lógico cuando el contador descendente es activado.

Control (Control R6).- Este registro de propósito especifico almacena los datos de otras instrucciones que no son ni de temporización ni de conteo, tales como las instrucciones de desplazamiento de bits, secuenciadores, ASCII, etc. En algunas otras instrucciones no es necesario este archivo.

Almacena la longitud, el puntero de posición y los siguientes bits de control dependiendo de la instrucción a la cual se asigne.

Entero (Integer N7).- Este archivo de propósito general almacena un numero entero de -32768 a 32767 pudiendo ser decimal, binario, hexadecimal, octal o ASCII. El procesador de MicroLogix 1200 cuenta con 256 registros donde en cada uno se guarda un valor en el rango mencionado.

Flotante (Float F8).- Este archivo de propósito general almacena un numero de tipo flotante de 1.1754944e-38 hasta 3.40282347e+38. El procesador de MicroLogix 1200 cuenta con 256 registros donde en cada uno se guarda un valor en el rango mencionado.

Archivos de Funciones (Function Files).- Los archivos de funciones son una combinacion de registros de archivos de estado y de control, contienen informacion de ciertas instrucciones que pudieran ser mas complejas y necesitan un extenso numero de parametros a configurar, también proveen información acerca del hardware del controlador.

5.- INSTRUCCIONES

En esta sección hablaremos de las instrucciones básicas, archivos de función, y archivos de datos que están presentes en todos los modelos de los controladores de la línea MicroLogix 1200. Así como su forma de programar en el software con un sencillo ejemplo.

5.1.- INSTRUCCIONES DE UN BIT

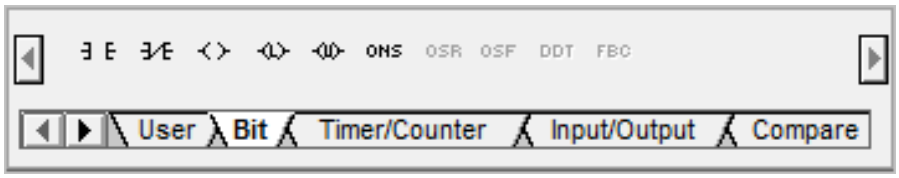
Son seis las instrucciones básicas que manejan un solo bit: contacto normalmente abierto, contacto normalmente cerrado, salida externa, detector de impulso, salida enclavada y salida desenclavada.

Estas instrucciones operan únicamente sobre un bit de datos. Durante la operación, el procesador puede ponerlo en condición falsa o verdadera, basado en la continuidad lógica de las

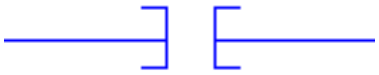
líneas del programa y de acuerdo al direccionamiento asignado en el mismo programa. Podemos direccionar un bit tantas veces como nuestro programa lo requiera. El límite sería la memoria.

Aunque existen algunos archivos de datos que por naturaleza manejan un solo bit, en todos los demás archivos de datos se pueden direccionar valores de un solo bit.

Cada instrucción en los diagramas de escalera tiene un nombre diferente a lo conocido comúnmente, este nombre es de tres letras y en el argot, se les conoce como Mnemónicas.



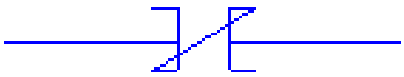
Contacto normalmente abierto: **XIC (Examine si es cerrado)**. Es una instrucción de entrada cuyo valor se asocia con un dispositivo externo o interno, la simbolización en el programa es:



Esta instrucción sólo puede tener 2 valores: 0 o 1; en donde:

0 = condición falsa de la instrucción
1 = condición verdadera de la instrucción

Contacto normalmente cerrado: XIO (Examine si es abierto). Es una instrucción de entrada cuyo valor se asocia con un dispositivo externo, la simbolización en el programa es:



Esta instrucción sólo puede tener 2 valores: 0 o 1; en donde:

0 = condición verdadera de la instrucción.
1 = condición falsa de la instrucción.

Salida externa: OTE. Es una instrucción de salida cuyo valor se asocia con un dispositivo externo o interno, la simbolización en el programa es:

Esta instrucción sólo puede tener 2 condiciones: Falsa o Verdadera; en donde:

Falsa = genera un valor de 0 en la dirección asignada a la instrucción.

Verdadera = genera un valor de 1 en la dirección asignada a la instrucción.

Evite duplicar instrucciones OTE con la misma dirección en un mismo programa. No es conveniente utilizar más de una vez una instrucción OTE con la misma dirección, ya que provocaría resultados no deseados.

Detector de impulso: OSR. Es una instrucción de entrada que monitorea la ocurrencia de un evento una sola vez durante un scan (One shot). La simbolización en el programa es:

Dirección del bit: **B3/0**

Esta instrucción se utiliza cuando se requiere iniciar un evento (salida), en el momento en que ocurra el cambio de estado de falso a verdadero en la entrada de la instrucción. No importa si la entrada es verdadera o falsa, lo que importa es la transición de falso a verdadero. Cada que detecta un cambio de falso a verdadero en la entrada, la instrucción OSR presenta una condición verdadera durante un ciclo de programa solamente, Figura 4.1. La salida de la instrucción es verdadera durante un ciclo de programa, independientemente del estado en que permanezca la entrada.

La dirección del bit que se utilice en esta instrucción debe ser única, no puede ser utilizada dos veces en el mismo programa.

Ejemplo:

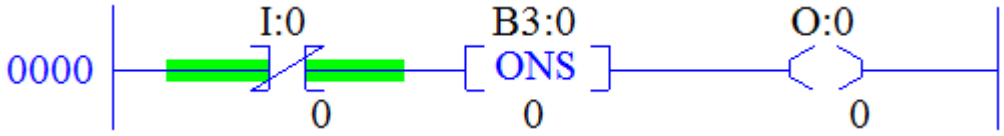
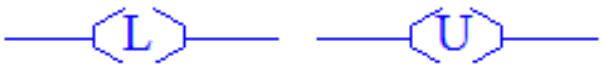


Figura 4.1

Salida Latch: OTL y salida Unlatch: OTU. Estas instrucciones de salida pertenecen al grupo que manejan un solo bit. La simbolización de las instrucciones Latch y Unlatch en el programa son respectivamente:



Estas son instrucciones de salida mantenidas que se pueden utilizar en pareja para la tabla de bit de datos que controlan. Se les llama salidas mantenidas porque se necesita de las dos instrucciones para poder cambiar de estado a un bit.

SÍ asignamos una dirección a la instrucción **OTL** que corresponda a una terminal de salida externa, el dispositivo de salida alambrado a esta terminal se energizará cuando el bit de memoria sea puesto en 1. Una instrucción **OTU** con la misma dirección de salida que la instrucción **OTL**

anterior, pondrá en cero el bit de memoria y por consecuencia el dispositivo de salida alambrado se desenergizará.

La instrucción **OTL** hace cambiar el estado del bit asignado de 0 a 1, cuando la condición de las entradas que manejan a **OTL** cambia de falso a verdadero, y permanecerá en 1 aunque **OTL** vuelva a cambiar de estado verdadero a falso. Cuando se presenta una instrucción **OTU** con el mismo bit de asignación que **OTL**, y su condición de entrada cambia de falso a verdadero, el bit en cuestión cambia de 1 a 0 y permanece en esa posición independientemente de la condición posterior de **OTU**. Figura 4.2.

Ejemplo:

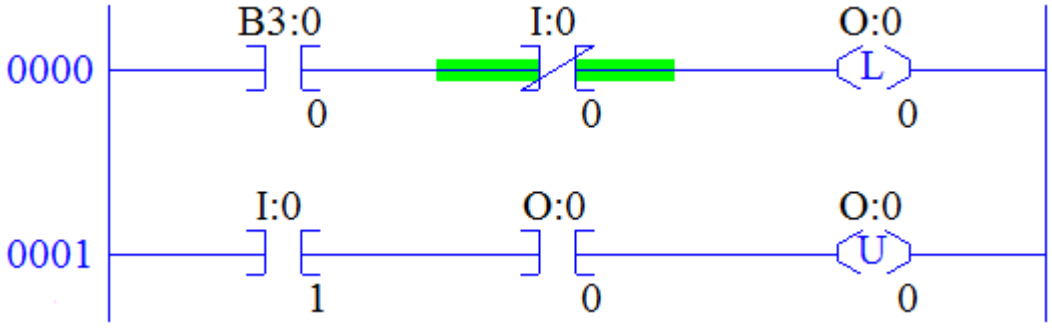
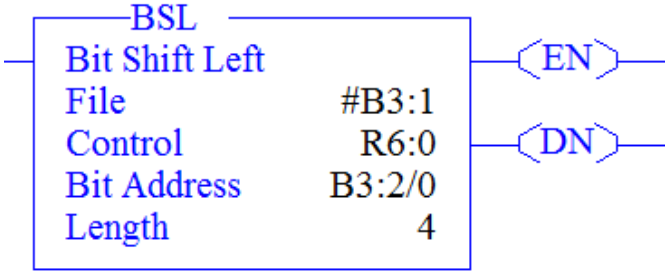


Figura 4.2

5.2.- INSTRUCCIONES DE DESPLAZAMIENTO DE BIT (BIT SHIFT)

Estas instrucciones son útiles para cargar y descargar información en un arreglo de bits, ya sea de 1 bit a la vez o palabras de bits.

BSL. Desplazamiento de bit a la izquierda (bit shift lef).- En cada transición de falso a verdadero, esta salida carga un bit de dato a un arreglo de bits, mueve a la izquierda el patrón de datos a través del arreglo y desecha el ultimo bit.



File.- Es la dirección del arreglo de bit que se quiere desplazar. Se debe usar el indicador (#) en la dirección.

Control.- Es la única dirección de la estructura de control en el área de control de la memoria que guarda el estado de los bits de instrucción, el tamaño de el arreglo y el puntero.

Bit Address.- Es la ubicación del bit que será añadido al arreglo.

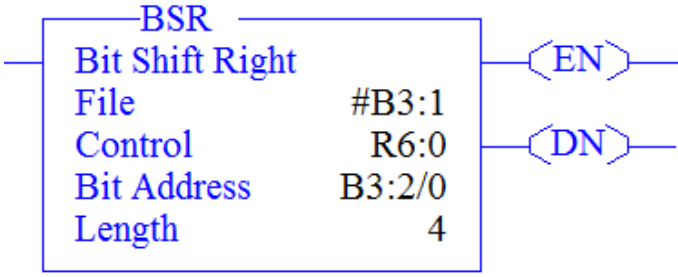
Length.- Es el número total de bits a ser desplazados por el BSL. Los bits a la izquierda del último bit en el arreglo hasta la siguiente palabra no podrán ser utilizados.

***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.

BSR. Desplazamiento de bit a la derecha (bit shift right).- En cada transición de falso a verdadero, esta salida carga un bit de dato a un arreglo de bits, mueve a la derecha el patrón de datos a través del arreglo y desecha el ultimo bit.

File.- Es la dirección del arreglo de bit que se quiere desplazar. Se debe usar el indicador (#) en la dirección.

Control.- Es la única dirección de la estructura de control en el área de control de la memoria que guarda el estado de los bits de instrucción, el tamaño de el arreglo y el puntero.



Bit Address.- Es la ubicación del bit que será añadido al arreglo.

Length.- Es el número total de bits a ser desplazados por el BSL. Los bits a la izquierda del último bit en el arreglo hasta la siguiente palabra no podrán ser utilizados.

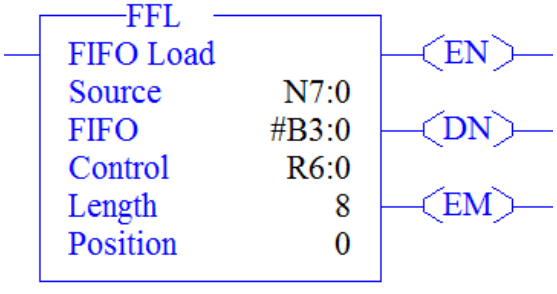
***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.

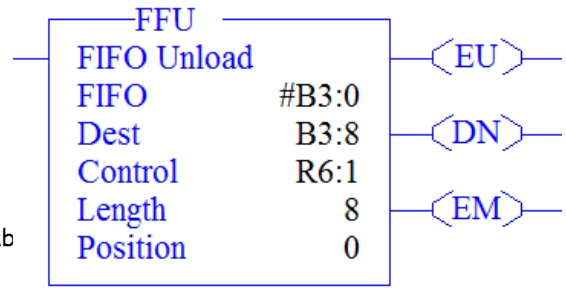
FFL/FFU. Primero en entrar, primero en salir (First In First Out cargar/descargar).- Las instrucciones de salida FFL y FFU son usadas en pares. El FFL carga una palabra en un archivo creado por el usuario llamado "FIFO stack" en sucesivas transiciones falso-verdadero de la entrada lógica de la línea de control. El FFU descarga las palabras provenientes del FIFO stack en el mismo orden en el que entraron.

Source.- Es una dirección de palabra o constante de programa (-32768 a 32767) que guarda el siguiente valor a ser introducido en el FIFO stack.

Destination.- Es una dirección de palabra que guarda el valor existente de el FIFO stack.

FIFO.- Es la dirección del stack. Debe ser una dirección de palabra indexada en la entrada, salida, estado, bit o archivo entero. La misma dirección es programada para el FFL y el FFU.





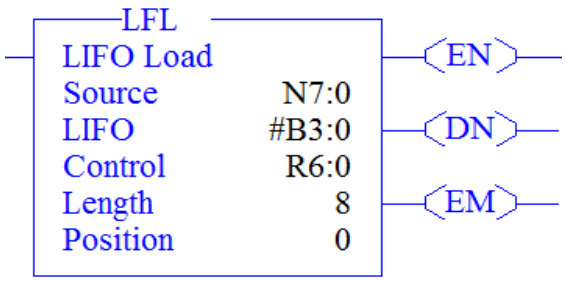
Length.- Es el máximo número de elementos en el stack, hasta un máximo de 128 palabras. El mismo número es programado para el FFL y el FFU.

Position.- Es la siguiente ubicación disponible donde la instrucción carga datos en el stack. El mismo número es programado para el FFL y el FFU.

Control.- Es una dirección de archivo de control. El bit de estado, la longitud del stack y la posición son guardados en este elemento. El mismo número es programado para el FFL y el FFU.

***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.

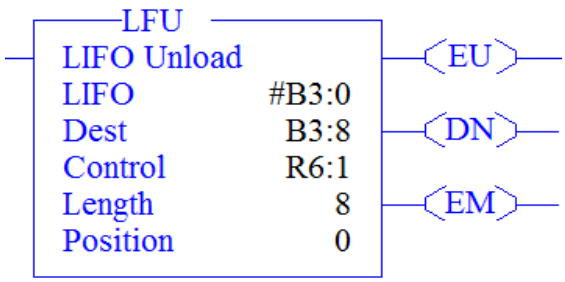
LFL/LFU. Ultimo en entrar, primero en salir (Last In First Out cargar/descargar).- Las instrucciones de salida LFL y LFU son usadas en pares. El LFL carga una palabra en un archivo creado por el usuario llamado "LIFO stack" en sucesivas transiciones falso-verdadero de la entrada lógica de la línea de control. El LFU descarga las palabras provenientes del LIFO stack invirtiendo el orden en el que entraron.



Source.- Es una dirección de palabra o constante de programa (-32768 a 32767) que guarda el siguiente valor a ser introducido en el LIFO stack.

Destination.- Es una dirección de palabra que guarda el valor existente de el LIFO stack.

FIFO.- Es la dirección del stack. Debe ser una dirección de palabra indexada en la entrada, salida, estado, bit o archivo entero. La misma dirección es programada para el LFL y el LFU.



Length.- Es el máximo número de elementos en el stack, hasta un máximo de 128 palabras. El mismo número es programado para el LFL y el LFU.

Position.- Es la siguiente ubicación disponible donde la instrucción carga datos en el stack. El mismo número es programado para el LFL y el LFU.

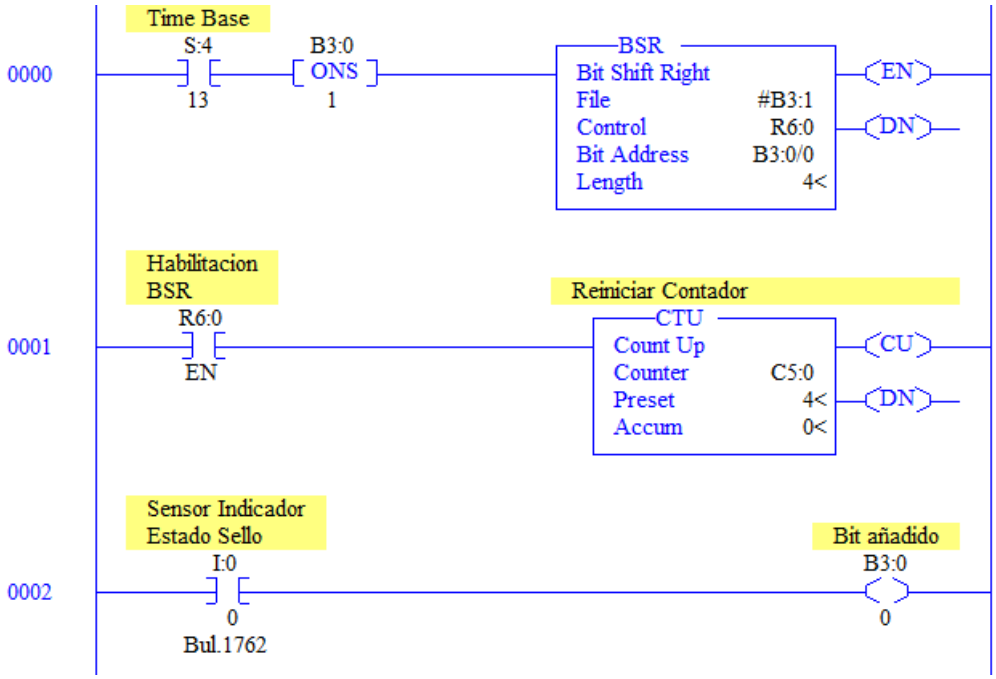
Control.- Es una dirección de archivo de control. El bit de estado, la longitud del stack y la posición son guardados en este elemento. El mismo número es programado para el LFL y el LFU.

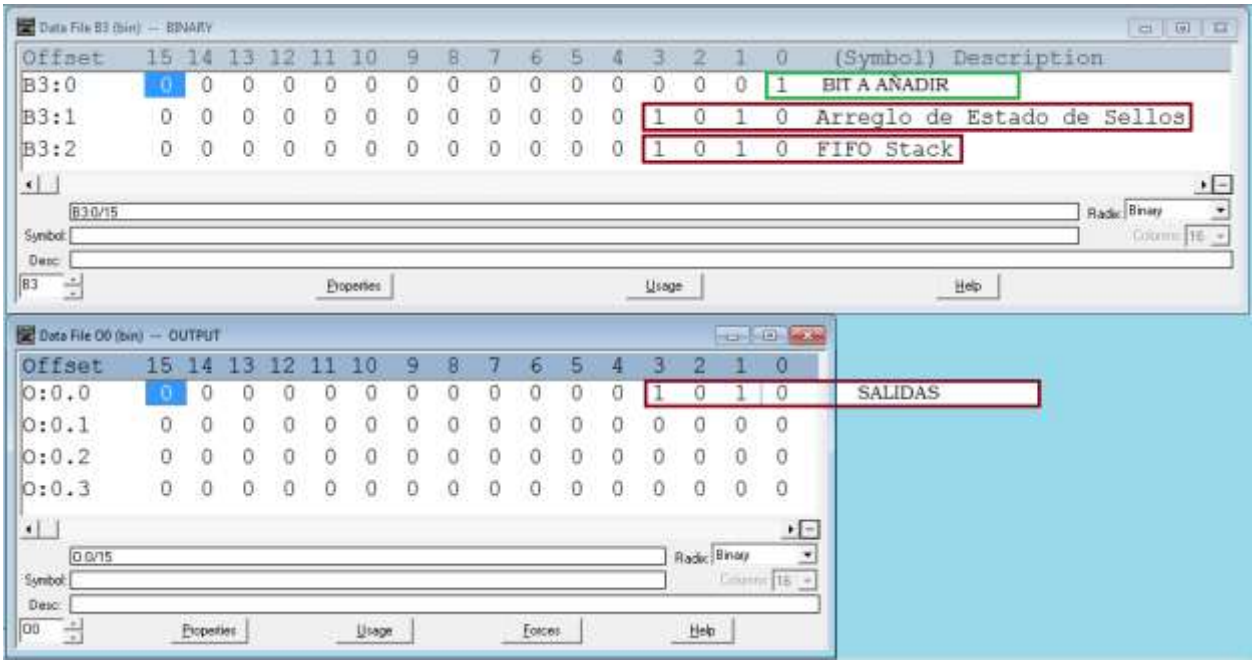
***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.

Ejercicio 1: En una banda transportadora se verifica el estado del sellado de las botellas en una envasadora, "0" mal sellado y "1" buen sellado.

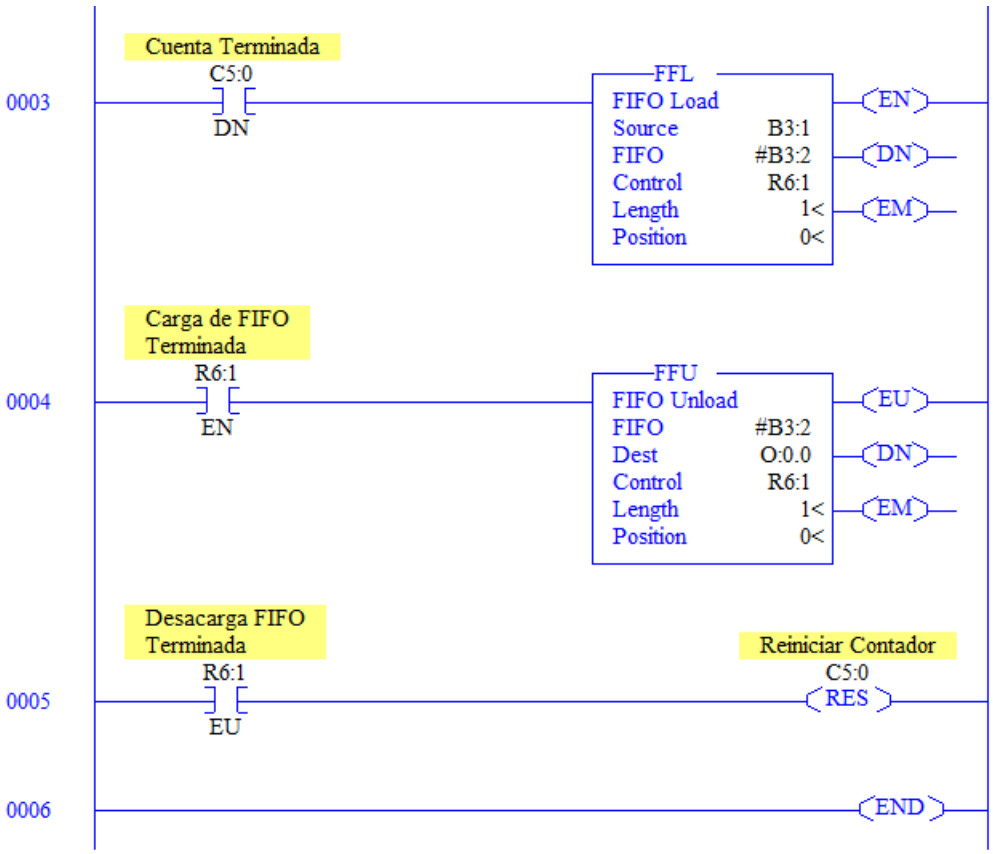
Simule el estado del sello y lleve un registro, cada 4 botellas un FIFO accionara 4 electroválvulas que a su vez realizaran un proceso dependiendo del estado de los sellos.

Solución: Utilice una base de tiempo S:4/13 para simular la llegada y partida de una botella. En cada transición falso-verdadero un BSR desplazara a la derecha un bit B3:0/0 de dato a través de un arreglo #B3:1, el estado de este bit dependerá de una entrada I:0/0 que envía una señal de mal o buen sello ("0" y "1" lógicos respectivamente). La habilitación R6:0/EN del BSR dará cada pulso al contador C5:0 para indicarnos la cantidad de sellos que se han analizado.





Cuando el acumulador de C5:0 llegue a 4, la línea 3 de nuestro programa será verdadera y el FFL cargara nuestro arreglo B3:1 a un stack #B3:2 de longitud 1, al cargar el arreglo R6:1/EN habilitara a FFU quien descargara el valor guardado del stack FIFO a las salidas O:0.0. Finalizando la descarga R6:1/EU reiniciara el contador para evaluar 4 nuevos valores de sellado.



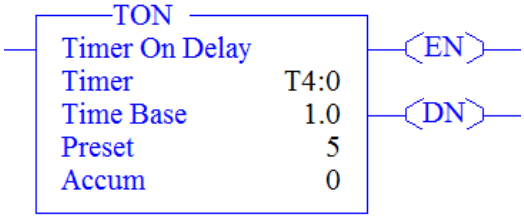
5.3.- INSTRUCCIONES DE TEMPORIZACION Y CONTEO

Las instrucciones de salida de temporización y conteo nos permiten controlar operaciones basadas en tiempo o número de eventos, son quizá las instrucciones más usadas y unas de las de mayor importancia en los sistemas secuenciales pese a lo sencillas y fáciles de usar dentro del control programable.

TON. Temporizador con retardo a la conexión (Timer On-Delay).- Esta instrucción se utiliza para encender o apagar una salida después de un tiempo (en intervalos ya sea de segundos o milésimas de segundo) previamente establecido (Preset) al ser activado el temporizador. Esta instrucción de salida comienza a contar el tiempo cuando la línea de entrada de control es "verdadera". Espera la cantidad de tiempo especificada, mantiene el seguimiento del acumulado de intervalos que han ocurrido "Accum" siempre y cuando la condición de la línea de control siga siendo verdadera, de lo contrario el valor del acumulador se restablecerá a 0. Por último, el bit DN se establece en "1" lógico cuando el valor del acumulador llega al valor preestablecido.

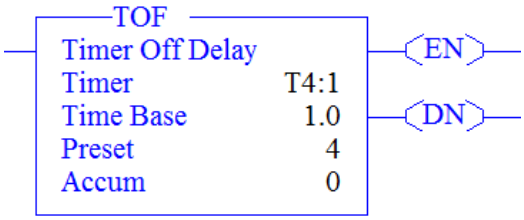
TOF. Temporizador con retardo a la desconexión (Timer Off-Delay).- Esta instrucción se utiliza para encender o apagar una salida después de un tiempo (en intervalos ya sea de segundos o milésimas de segundo) previamente establecido (Preset) al ser desactivado el temporizador. Esta instrucción de salida comienza a contar el tiempo cuando la línea de entrada de control hace una transición verdadero-falso. Espera la cantidad de tiempo especificada, mantiene el seguimiento del acumulado de intervalos que han ocurrido "Accum" siempre y cuando la condición de la línea de control siga siendo falsa, de lo contrario el valor del acumulador se restablecerá a 0. Por último, el bit DN se establece en "1" lógico cuando el valor del acumulador llega al valor preestablecido.

RTO. Temporizador con retardo a la conexión retentivo (Retentive Timer On-Delay).- Funciona de la misma manera que el TON, a diferencia de este, el RTO mantiene el valor del acumulador aun si la condición de la línea de control cambie a falso. Si la línea se establece como verdadera nuevamente continuara la cuenta desde donde se perdió la continuidad de la línea. En otras palabras el RTO lleva el control de cuánto tiempo la condición de la línea ha sido verdadera.



Para configurar un temporizador cualquiera que sea su tipo se necesita solo 4 parámetros:

Timer.- Es el direccionamiento del espacio de memoria que ocupara el temporizador.

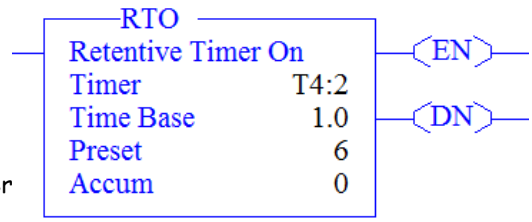


Time Base.- Determina la duración de cada intervalo de tiempo.

Preset.- Es el valor del punto en el cual el acumulador activara el bit DN (done).

Accum.- Es el número de intervalos de tiempo que se ha alcanzado.

***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.



La cantidad de tiempo que se puede contar está limitado por el valor de la dirección del Preset (16 bits=0 - 32767), tenga en cuenta la base de tiempo que va a implementar.

Milésimas de segundo. 0.001 x 32767 = 32.767 segundos

Centésimas de segundo. 0.01 x 32767 = 327.67 segundos

Segundos. 1.0 x 32676 = 32,767 segundos

CTU. Contador ascendente (Count Up).- Esta instrucción de salida cuenta de forma ascendente cada transición falso-verdadero en la entrada de la línea de control, cuando el valor acumulado (Accum) alcanza el valor preestablecido (Preset) se activa una salida DN. Después de activar la salida, el acumulador seguirá llevando la cuenta hasta que otra instrucción RES con la misma dirección sea utilizada.

La frecuencia de la señal de entrada no deberá exceder el valor máximo de tiempo de escaneo.

CTD. Contador descendente (Count Down).- Esta instrucción de salida cuenta de forma descendente cada transición falso-verdadero en la entrada de la línea de control, cuando el valor acumulado (Accum) alcanza el valor preestablecido (Preset) se activa una salida DN. Después de activar la salida, el acumulador seguirá llevando la cuenta hasta que otra instrucción RES con la misma dirección sea utilizada.

La frecuencia de la señal de entrada no deberá exceder el valor máximo de tiempo de escaneo.

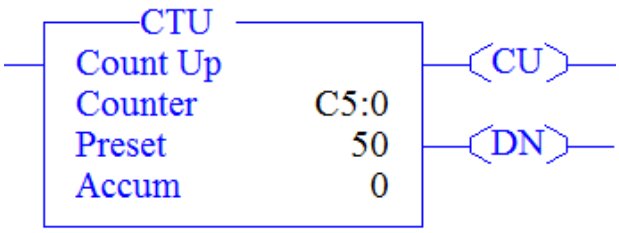
Para configurar un contador se necesitan solo 3 parámetros:

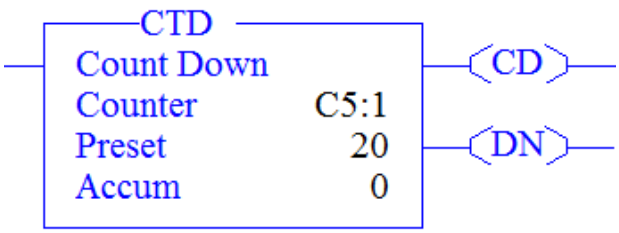
Counter.- Es el direccionamiento que ocupara en la memoria el contador.

Preset.- Es el valor del punto en el cual el acumulador activara el bit DN (done).

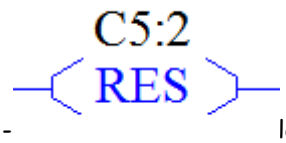
Accum.- Es el número de transiciones que han ocurrido.

***Nota:** los parámetros de la imagen son solo ejemplos, varían dependiendo del programa.





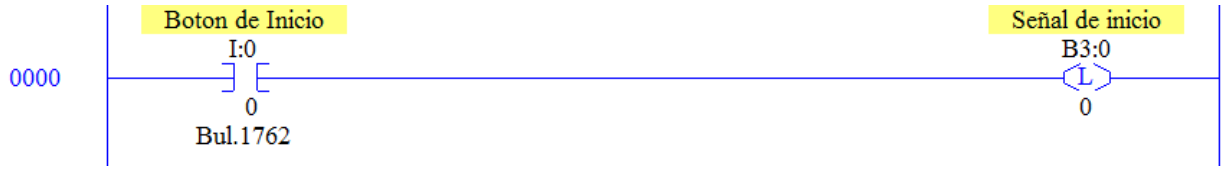
RES. Restablecer (Reset).- Esta instrucción es usada para restablecer el valor de al acumulador de un temporizador o un contador a cero, esta acción se realiza cuando la condición de la línea de control de RES sea verdadera. Para reiniciar un contador o un temporizador con esta instrucción recuerde que debe tener la misma dirección del elemento a restablecer.



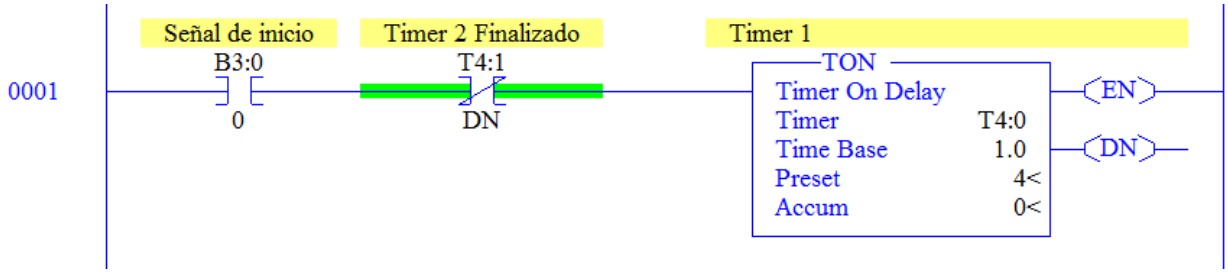
Ejercicio 2: Con un botón de inicio, construya un oscilador astable, 4 segundos activo y 2 segundos inactivo, cuando el oscilador realice 5 ciclos deberá parar hasta que se oprima el botón nuevamente.

Solución: Utilice dos temporizadores, uno de 4 segundos y otro de 2 segundos después de realizar el circuito de control con un contador ascendente restablezca la cuenta al llegar a 5 ciclos. El ciclo debe poder reiniciar nuevamente.

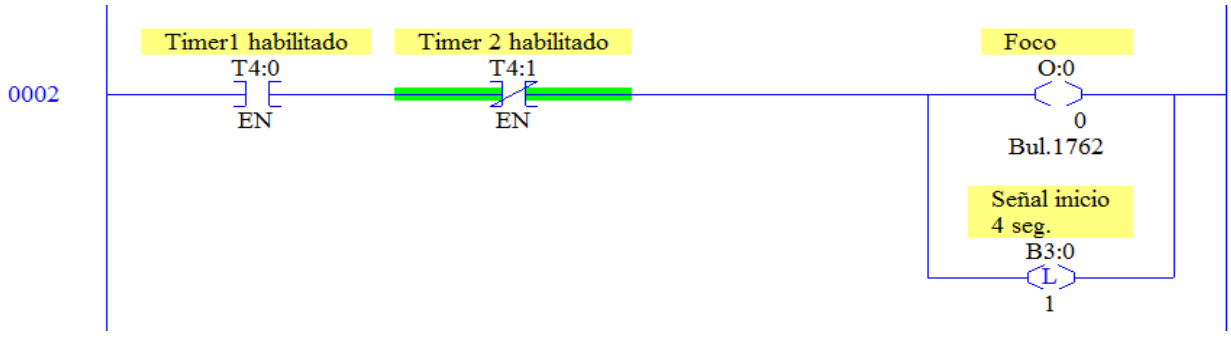
Utilice un contacto normalmente abierto con dirección I:0.0 (Botón de Inicio) el cual accione una salida interna enclavada B3:0/0 (Señal de inicio), esta señal nos permitirá hacer que el ciclo sea continuo.



Lleve la señal de inicio a un contacto normalmente abierto y la señal T4:1/DN del segundo temporizador (Timer 2) a un contacto normalmente cerrado, estos estarán en serie en la entrada del primer temporizador (Timer 1) haciendo que este inicie su cuenta al oprimir el Botón de Inicio. El Timer 1 será el que controle los 4 segundos del oscilador astable.

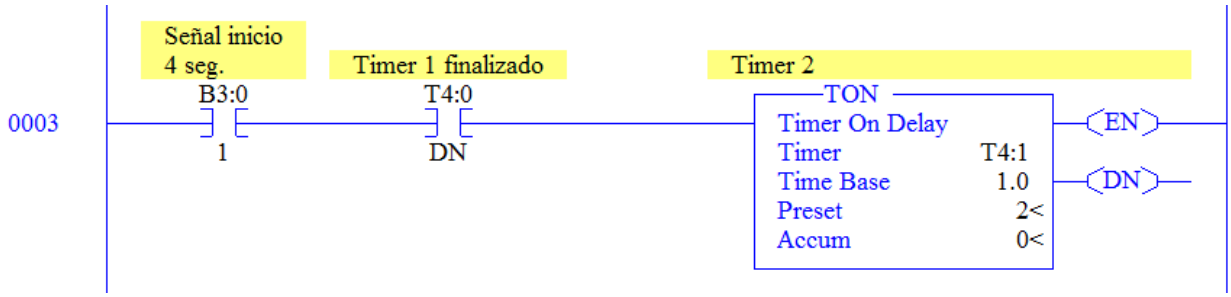


Al ser habilitado el Timer 1 la señal T4:0/EN cerrara un contacto normalmente abierto, y junto a un contacto normalmente cerrado de T4:1/EN del Timer 2 energizaran una salida externa O:0.0 (Foco) y a una salida interna enclavada B3:0/1 (Señal inicio 4 seg).

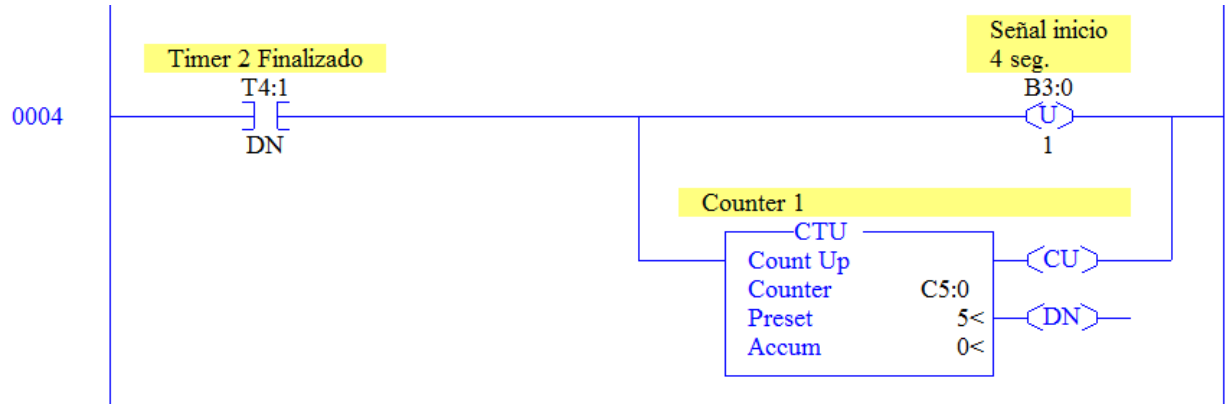


La señal de inicio de 4 segundos en un contacto normalmente abierto y la señal T4:0/DN (Timer finalizado) también en un contacto abierto, nos indican que han finalizado los 4 segundos

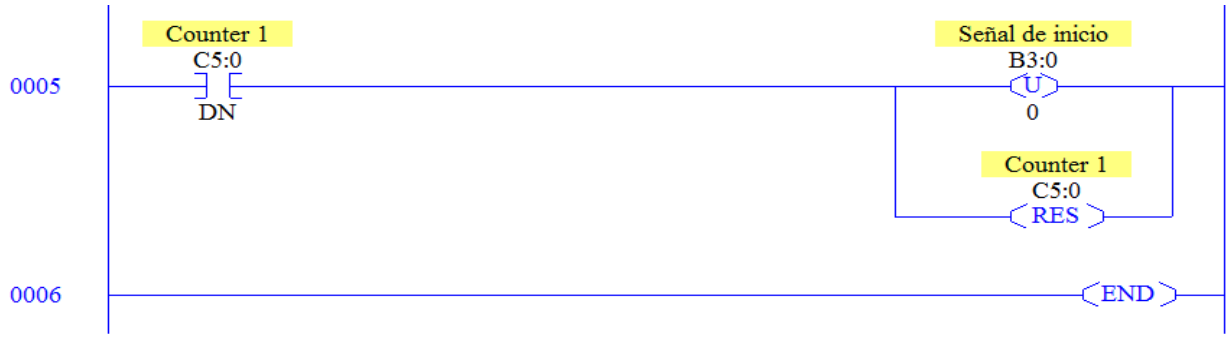
en activo del Foco, de esta manera inicializamos a el Timer 2. Al ser activado el Timer 2, abre el contacto de la línea 0002 Timer 2 habilitado, lo que desenergiza el Foco e inicia la cuenta de los 2 segundos inactivos del astable.



Al pasar los 2 segundos del Timer 2 la señal T4:1/DN (Timer 2 Finalizado) activara la salida interna desenclavada de Señal inicio 4 seg y a su vez hará una transición falso-verdadero en el contador C5:0 (Counter 1) el cual llevara el conteo de los ciclos acumulados. Al desenclavar la Señal inicio 4 seg reiniciamos el Timer 2 pues el contacto normalmente abierto de la línea 0003 se abre. Ya reiniciado Timer 2, el contacto Timer 2 finalizado en la línea 0001 se cierra nuevamente y permite iniciar el ciclo ya que Señal de inicio sigue enclavada.



Si el acumulador de Counter 1 es igual a 5 se activara la señal C5:0/DN, cerrando el contacto normalmente abierto y así activara la salida interna desenclavada B3:0/0 junto con la instrucción RES de Counter 1 para reiniciar el valor del acumulador. Al estar desenclavada Señal de inicio, el Timer 1 ya no iniciara una nueva cuenta hasta que Botón de inicio se vuelva a activar.

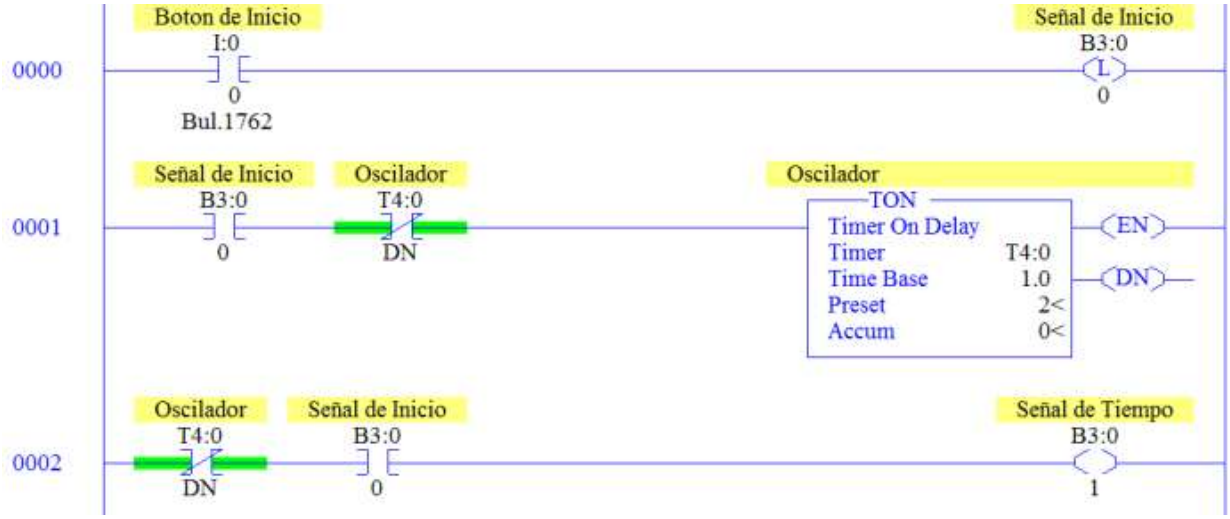


Ejemplo:

Muchas veces usted solo necesitara un oscilador simple el cual mande un impulso en determinado tiempo, en ese caso, tome en cuenta el siguiente ejemplo para evitar la construcción de osciladores más complejos cuando su aplicación no lo necesite.

Llamaremos a este ejemplo como **Oscilador simple**, para futuras referencias.

Utilice un Botón de Inicio (I:0.0) para enclavar una salida retentiva (B3:0/0) llamada Señal de Inicio y utilícela en la entrada de un temporizador (T4:0). En la línea 0002 una salida interna (B3:0/1), Señal de Tiempo, se activara al momento de dar un pulso a Botón de Inicio, ya que la señal "bit done" (T4:0/DN) no está activada y el contacto sigue cerrado y Señal de Inicio ha sido activada, cerrando a su vez un contacto normalmente abierto. En ese mismo instante, el acumulador del Oscilador comenzara a contar, dependiendo de la base de tiempo y el tiempo preestablecido que use, la señal de T4:0/DN, al ser establecida en "1" lógico estará desactivando al mismo oscilador al abrir un contacto normalmente cerrado que se encuentra en la línea 0001, de esta manera el ciclo vuelve a iniciar y ya hemos realizado nuestra base de tiempo. El oscilador solo se detendrá al desenclavar la salida B3:0/0 Señal de Inicio.



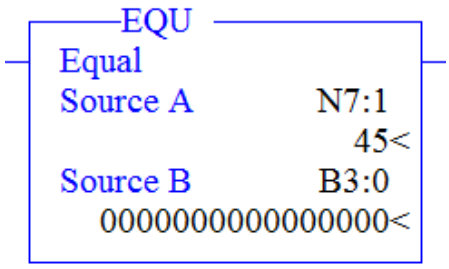
5.4.- INSTRUCCIONES DE COMPARACION

Estas instrucciones se utilizan para comparar el valor de dos datos, A y B. Las operaciones disponibles son: Igual, Diferente, Menor que, Menor o Igual que, Mayor que, Mayor o Igual que, Igual con máscara y Limite.

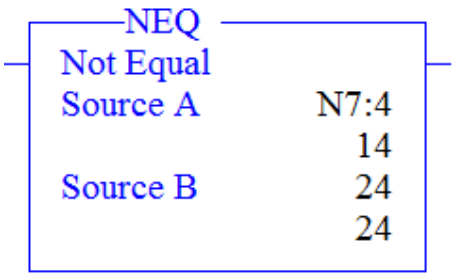
Source A.- Para la fuente "A", el usuario debe introducir la dirección de una palabra, B3, T4, C5, N7, etc.

Source B.- Para la fuente "B", el usuario debe introducir la dirección de una palabra, B3, T4, C5, N7 o una constante de programa; binario, hexadecimal o decimal.

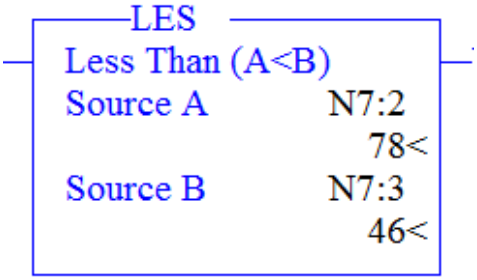
EQU. Igual (Equal, A = B).- Compara dos valores específicos. Si "A" es igual a "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



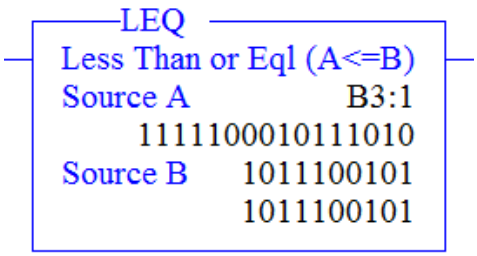
NEQ. Diferente (Not Equal, A ≠ B).- Compara dos valores específicos. Si "A" es diferente a "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



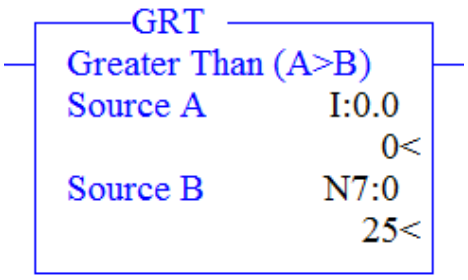
LES. Menor que (Less Than, A < B).- Compara dos valores específicos. Si "A" es menor que "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



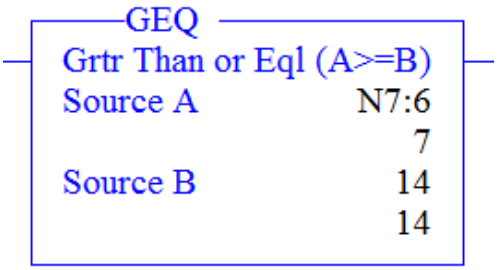
LEQ. Menor o Igual que (Less Than or Eql, A<=B).- Compara dos valores específicos. Si "A" es menor o igual que "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



GTR. Mayor que (Greater Than, A>B).- Compara dos valores específicos. Si "A" es mayor que "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



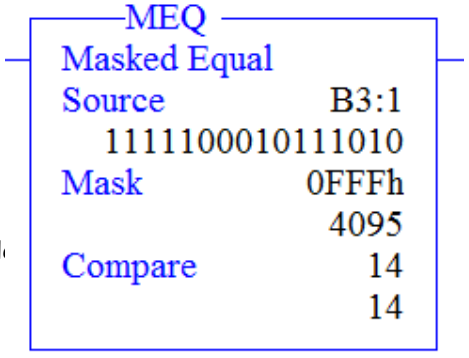
GEQ. Mayor o Igual que (Greater Than or Equal, (A>=B).- Compara dos valores específicos. Si "A" es mayor o igual que "B", esta instrucción de entrada será verdadera y la salida será energizada. Ninguna otra condición afectara el estatus de la salida.



MEQ. Comparación Igual con mascara (Masked Comparision for Equal).- Esta instrucción condicional compara 16 bits de datos de una dirección con 16 bits de una dirección de referencia a través de una máscara. Si los valores son iguales la instrucción es verdadera.

Esta mascara nos permite comparar solo un segmento de el registro de bits, es el equivalente a realizar una operación X-OR en lógica booleana.

Source.- Es la dirección del valor que se va a comparar.



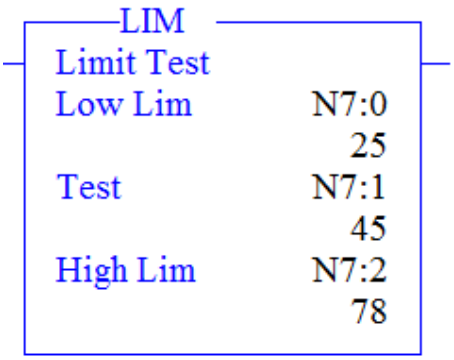
Mask.- Es el valor de la máscara por el cual la instrucción mueve los datos. Puede ser un valor hexadecimal, binario o decimal.

Compare.- Es un valor entero o una dirección de referencia.

LIM. Prueba de Limite (Limit Test).- Use esta instrucción cuando requiera realizar una prueba para valores dentro o fuera de un rango en especifico. Los limites alto y bajo (High and Low) pueden ser una palabra o una constante de programa.

Dependiendo en cómo se definan los parámetros, el estatus de la salida de la instrucción será el siguiente:

Si el límite inferior (low limit) tiene un valor más grande que el límite alto, la instrucción es falsa cuando el valor de prueba está dentro de los límites. Si el valor de prueba es igual o está fuera de los límites, la instrucción es verdadera.



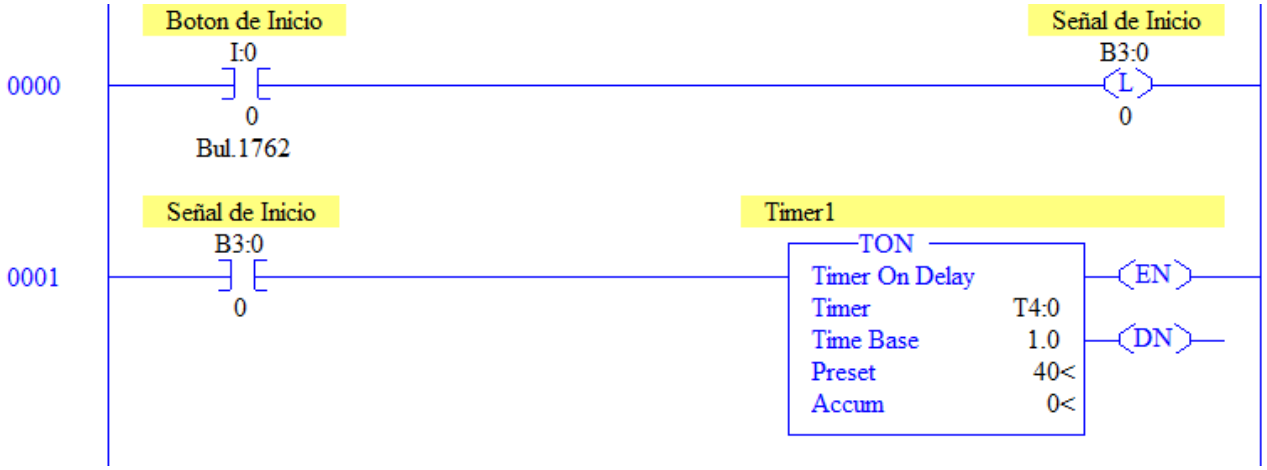
Si el límite inferior (low limit) tiene un valor igual o menor que el limite alto, la instrucción es verdadera cuando el valor de prueba está dentro de los limites o es igual a cualquiera de ellos. Si el valor de prueba está fuera de los límites, la instrucción es falsa.

Ejercicio 3: Construya un contador ascendente de décadas de 4 bits.

Solución: Utilice 4 comparadores, cada comparador controlara una salida.

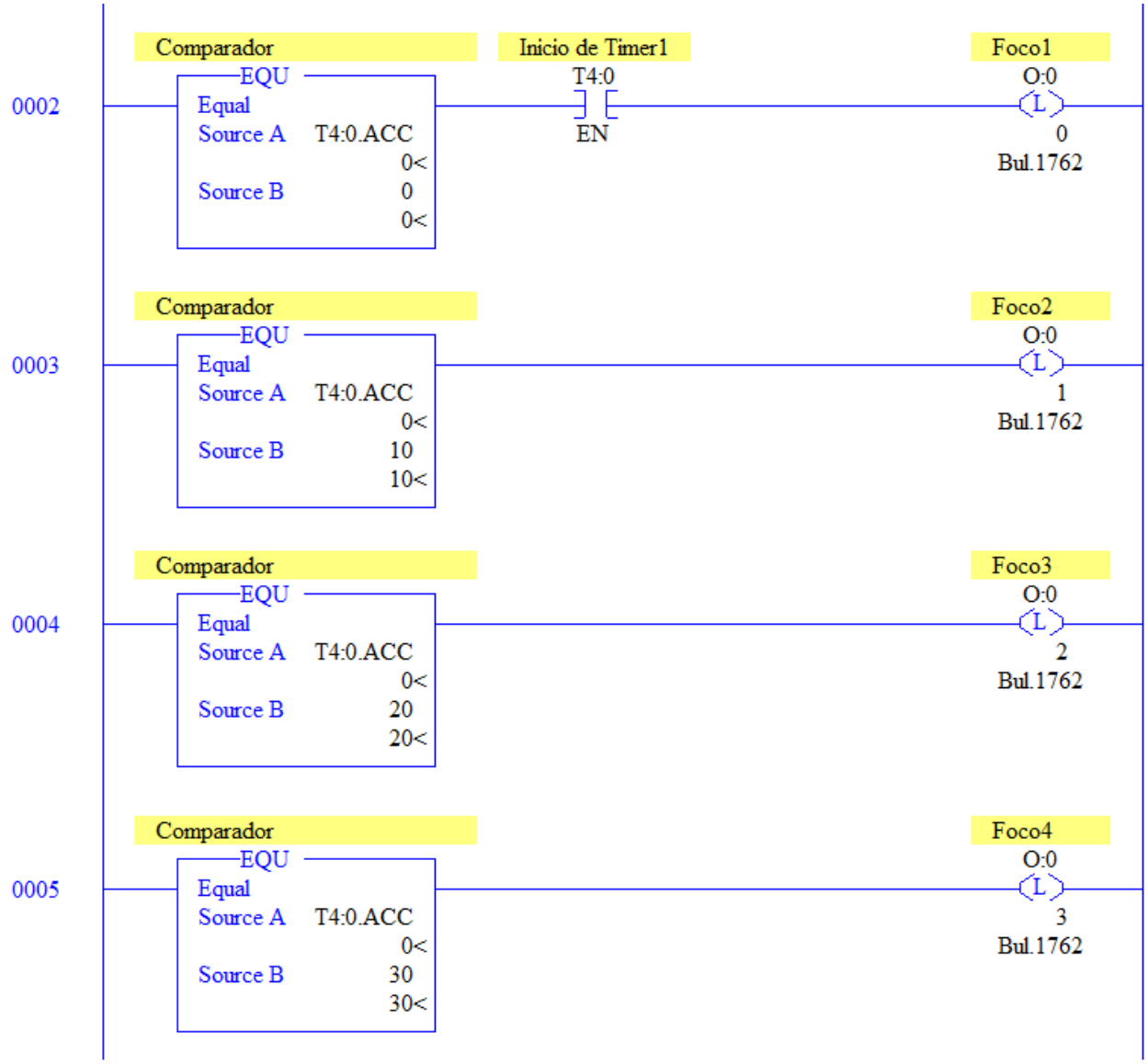
Diagrama

Con un contacto normalmente abierto I:0.0 (Botón de Inicio) activaremos una salida interna B3:0/0 (Señal de Inicio), con esta señal activaremos a un temporizador T4:0 (Timer1) y así iniciaremos nuestra secuencia.

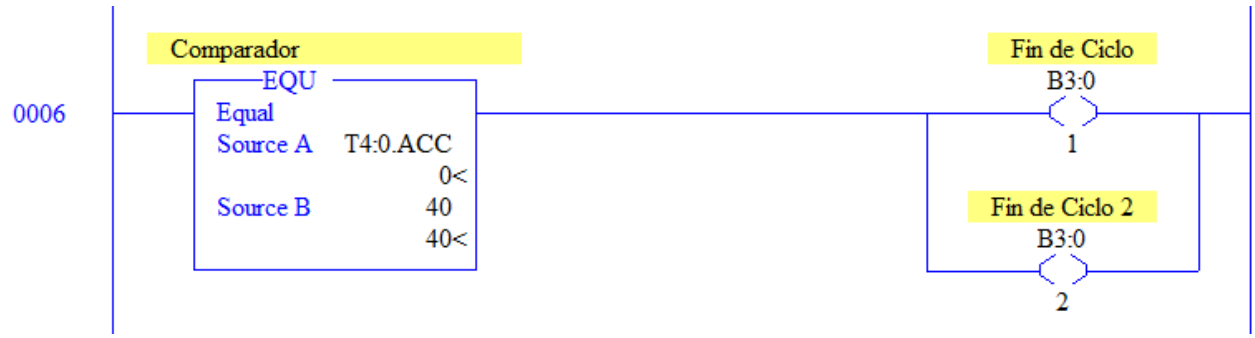


Cuando Timer1 sea activado el acumulador (T4:0.ACC) comenzara su cuenta, tomaremos la dirección de este acumulador y la utilizaremos en todas las direcciones Source A de nuestros comparadores, así, mientras el acumulador siga incrementando activara todas las salidas al llegar al valor indicado en el Source B de cada comparador. Usaremos salidas retentivas, pues en un comparador EQU la salida solo será verdadera mientras los dos valores sean iguales.

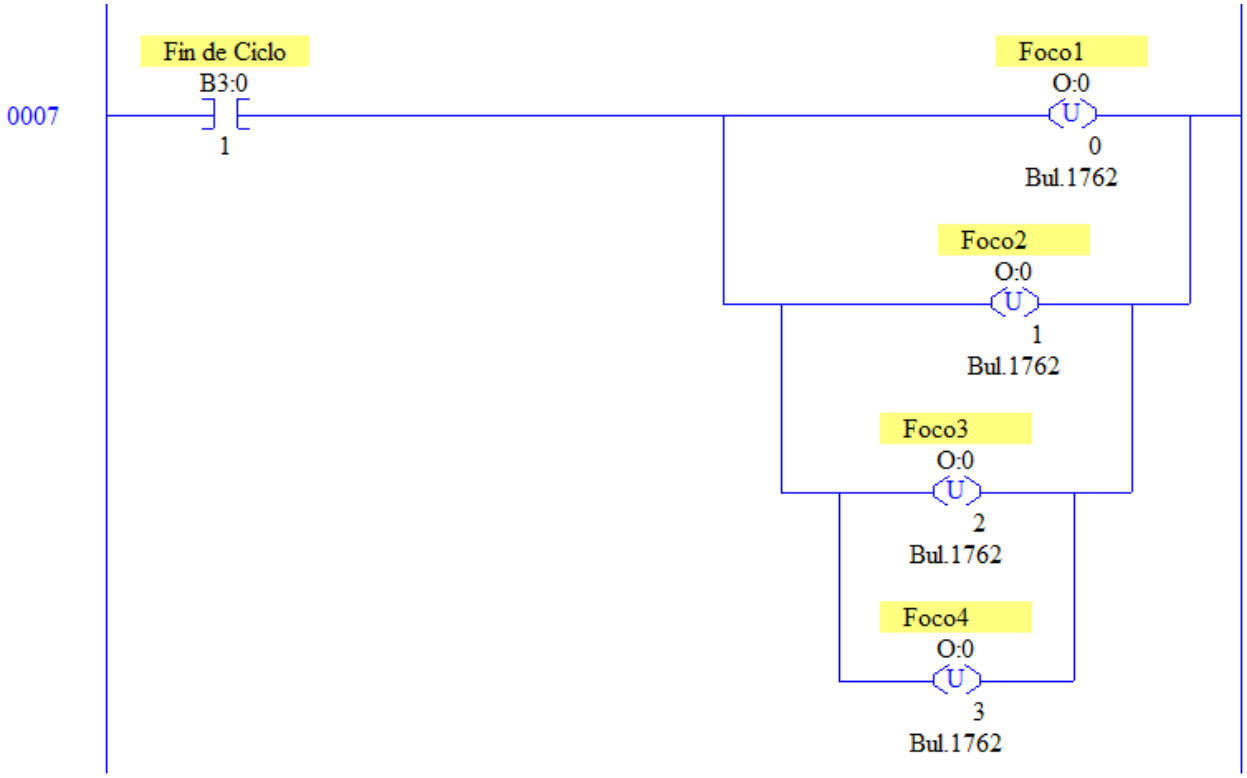
La señal T4:0/EN (Inicio de Timer1) la usaremos para que la salida del Foco1 no se active sin antes oprimir el Boto de Inicio.



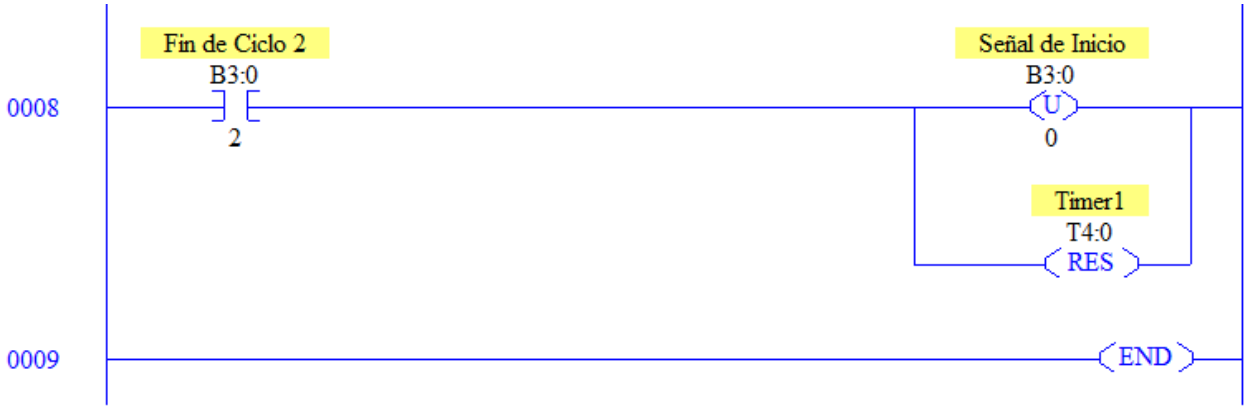
Al finalizar nuestra secuencia utilizaremos un contador final para desenclavar todas las salidas, reiniciar el valor del acumulador de Timer1 y desenclavar a Señal de Inicio. Usamos dos salidas internas B3:0/1 (Fin de Ciclo) y B3:0/2 (Fin de Ciclo 2) pues no podemos usar más de 5 ramas en paralelo, esto excede el límite del procesador.



Fin de ciclo desenclava las salidas 0, 1, 2 y 3 de O:0.0.



Fin de Ciclo 2 desenclava a Señal de Inicio y reinicia el acumulador de Timer1.



***Nota:** Intente utilizar otro tipo de comparador y diseñe un contador diferente. Este es solo un ejemplo ilustrativo, muchas de las consideraciones que se deben tener al diseñar un diagrama de escalera fueron pasadas por alto.

5.5.- INSTRUCCIONES DE CONTROL

Estas instrucciones son usadas para cambiar el orden en el que el procesador escanea un programa de escalera. Son usadas especialmente para reducir el tiempo de escaneo, crear un programa más eficiente y depurarlo con mayor facilidad**.**

Estas instrucciones añaden gran versatilidad al diseño del programa, evitan tener bloques redundantes dentro de la estructura lógica y además, le permite al usuario usar otros subprogramas a su conveniencia.

JMP. Saltar a Etiqueta (Jump to Label).- Cuando la condición de la línea de entrada de esta instrucción es verdadera, el procesador salta hacia adelante o hacia atrás a su instrucción LBL correspondiente. El saltar a través del programa permite omitir una parte del programa hasta que sea requerido ejecutar la misma parte del programa repetidas veces. Utilice un decimal de 0-999 para etiquetarlo.

LBL. Etiqueta (Label).- Esta instrucción de entrada es el objetivo de cualquier instrucción JMP que tenga su misma etiqueta. Debe ser la primera instrucción de la línea y siempre será evaluada como 1 lógico, sin bits de control.

Puede utilizar varias instrucciones JMP que dirijan al programa hacia la misma etiqueta, pero no utilice la misma dirección para la instrucción LBL, esto producirá un error en el tiempo de compilación.

Considere las veces que realiza movimientos dentro del lazo de las instrucciones JMP y LBL. Si se hacen muchos saltos podría ocasionar que el “watchdog timer” agote su tiempo de espera y el procesador se vaya a falla. Puede usar un contador, un temporizador o el registro de escaneo del programa (S3: bits 0 a 7) para limitar la cantidad de tiempo dentro de estas instrucciones.

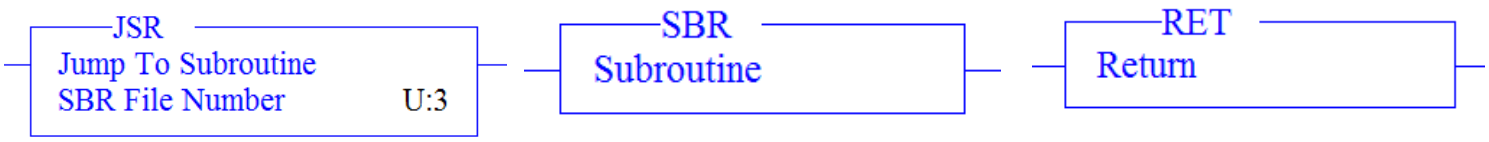
No utilice saltos dentro de una zona de MCR. Las instrucciones programadas dentro de la zona de MCR empezando con la instrucción LBL y terminando con el final de MCR siempre serán evaluadas como si la zona de MCR fuera verdadera, sin importar el estado de la instrucción de inicio de MCR.

JSR. Saltar a Subrutina (Jump to Subroutine).- Cuando la condición de la línea de entrada de esta instrucción de salida sea verdadera, causa que el procesador salte al archivo de subrutina indicado. Cada subrutina debe tener un único número de archivo (decimal, 3 – 255).

Es posible anidar hasta 8 niveles de subrutinas, cada subrutina anidada dirigirá el flujo de programa desde el programa principal hacia una subrutina y después a la siguiente.

SBR. Subrutina (Subroutine).- Una subrutina se utiliza para guardar segmentos recurrentes de un programa que deberá ser ejecutado desde varios puntos dentro de la aplicación. Esta instrucción debe ser programada como la primera línea de una subrutina. Su uso es opcional, pero es recomendado. Al igual que LBL esta instrucción siempre es evaluada como verdadera sin bits de control.

RET. Regreso de Subrutina (Return from Subroutine).- Esta instrucción de salida marca el final de una subrutina. La línea de control de **RET** debe ser condicional si precede el final de la subrutina. El procesador continua con la ejecución del programa en la instrucción siguiente al **JSR** que provoco la salida del programa, ya sea en una subrutina anterior o en el programa principal.



MCR. Master Control Reset.- Esta instrucción de salida es usada para controlar segmentos el diagrama de escalera, donde todas las salidas no retentivas serán deshabilitadas al mismo tiempo y durante el mismo tiempo. Para delimitar su rango de acción es usada en pares, una para definir el inicio y otra para el final.

Cuando la entrada del MCR sea falsa, todas las salidas no retentivas dentro de la zona de control serán deshabilitadas. Cuando la condición de la línea sea verdadera, todas las líneas serán escaneadas de acuerdo a las condiciones de las mismas.

No utilice ninguna condición en la instrucción MCR final, esta debe ser la única instrucción en la línea.

TDN. Fin Temporal (Temporary End).- Use esta instrucción para depurar progresivamente el programa u omitir el resto de de la subrutina o programa en curso.

Cuando la línea de entrada de esta instrucción de salida es verdadera, se detiene el escaneo del programa, se actualizan las entradas y salidas y continua el escaneo desde la línea 0 del programa principal.

Si la línea de entrada es falsa, el procesador continua el escaneo normalmente.

Usar esta instrucción en una subrutina anidada termina la ejecución de todas las subrutinas anidadas.

SUS. Suspender (Suspend).- Use esta instrucción para depurar o diagnosticar el programa.

Cuando la línea de control es verdadera, esta instrucción manda al controlador a modo "Suspend Idle". El ID suspendido se coloca en una palabra (S:7) de el archivo de estado. El archivo suspendido es colocado en una palabra (S:8) de el archivo de estado. Todas las salidas son desenergizadas. Introduzca un numero ID de -32768 a 32767.

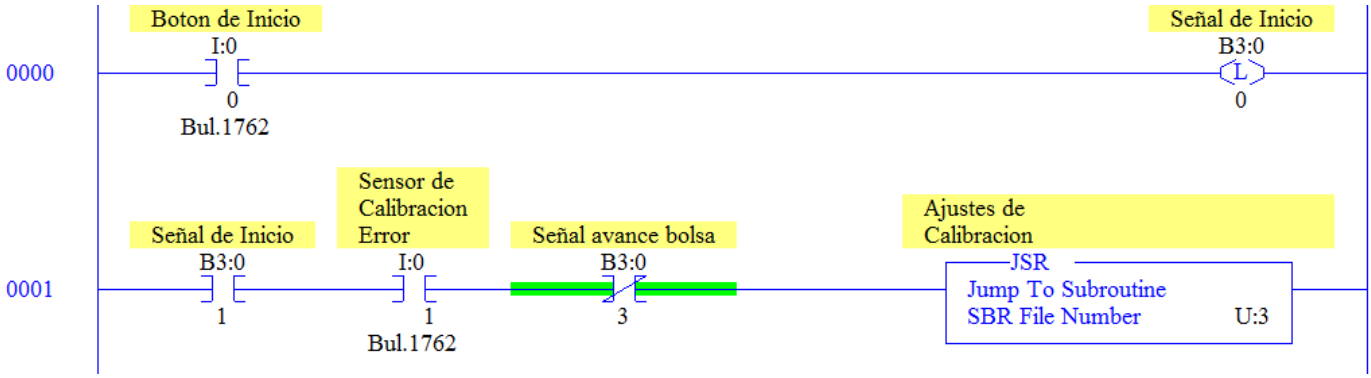
Ejercicio 4: En un proceso dado se tienen 4 etapas; calibración, avance de bolsa, pesaje y corte. Las 4 etapas se realizan en cada ciclo de programa y todas conllevan un tiempo, pero la parte de la calibración solo es necesaria si el sensor de calibración requiere de un ajuste.

Elabore un programa que cumpla con esos sencillos requerimientos, aumentando la eficiencia en el escaneo del programa.

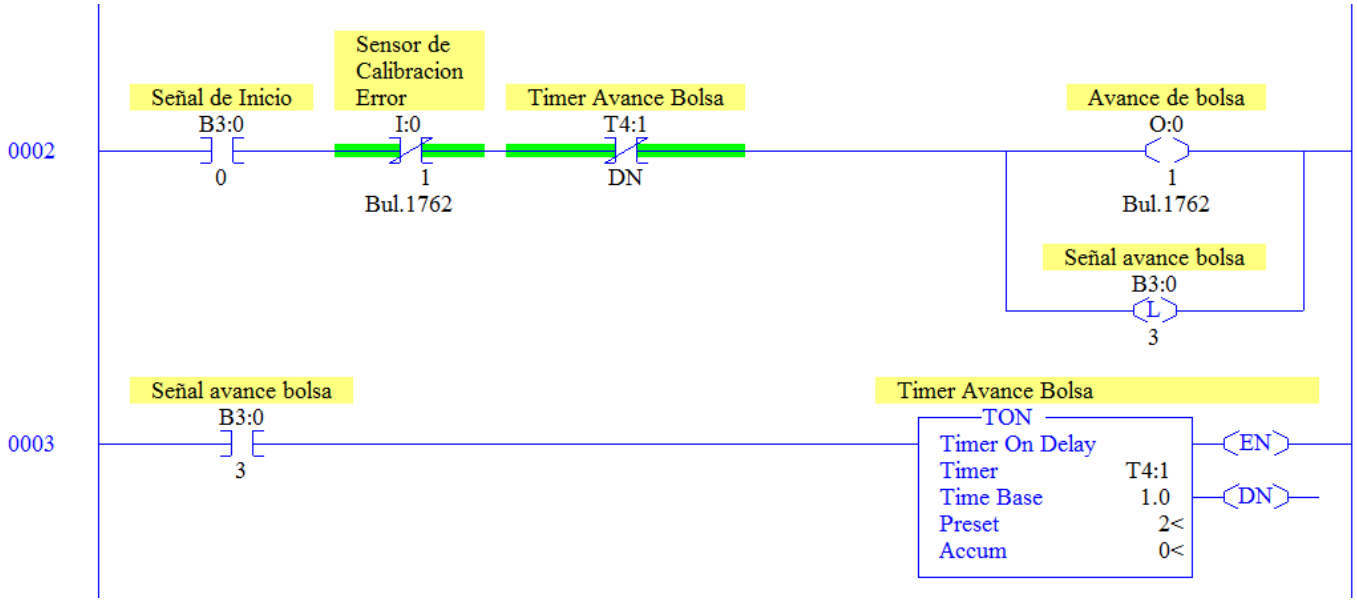
Solución: Utilice algún tipo de salto para entrar a una parte del programa que se necesitará solo en algún tipo de circunstancia.

Diagrama

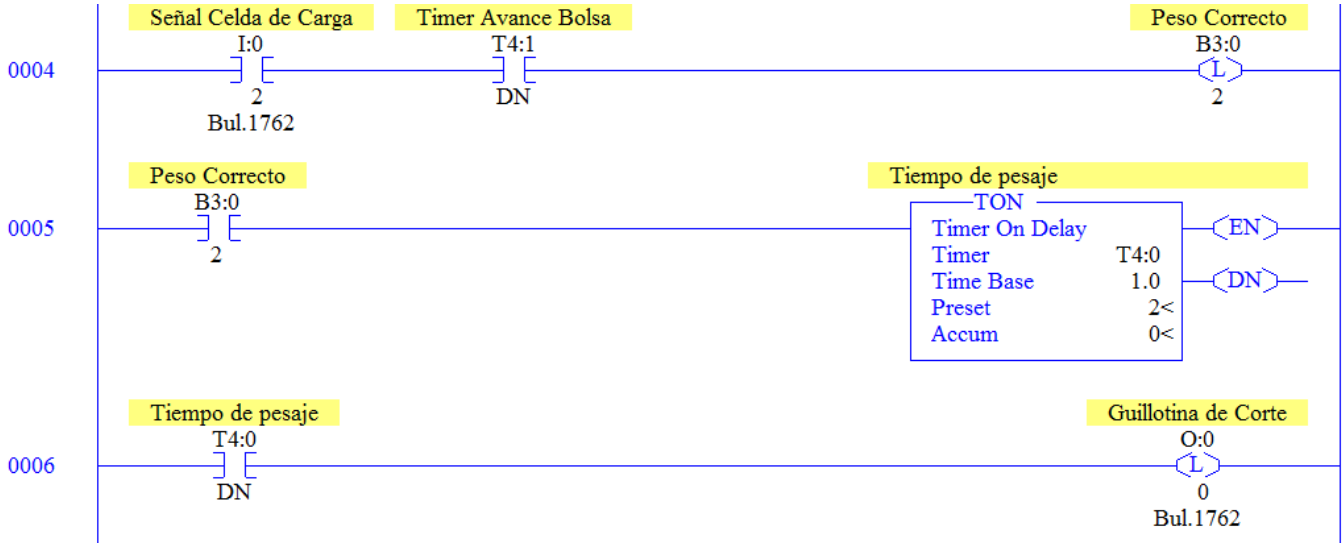
Después de dar inicio a nuestro programa (línea 0000), analizaremos la condición de la línea de entrada de una subrutina llamada "Ajustes de Calibración", uno de las señales será "Sensor de Calibración" este contacto nos indicara si el sistema necesita algún ajuste, de esta manera solo estaremos calibrando el equipo cuando sea necesario (línea 0001).

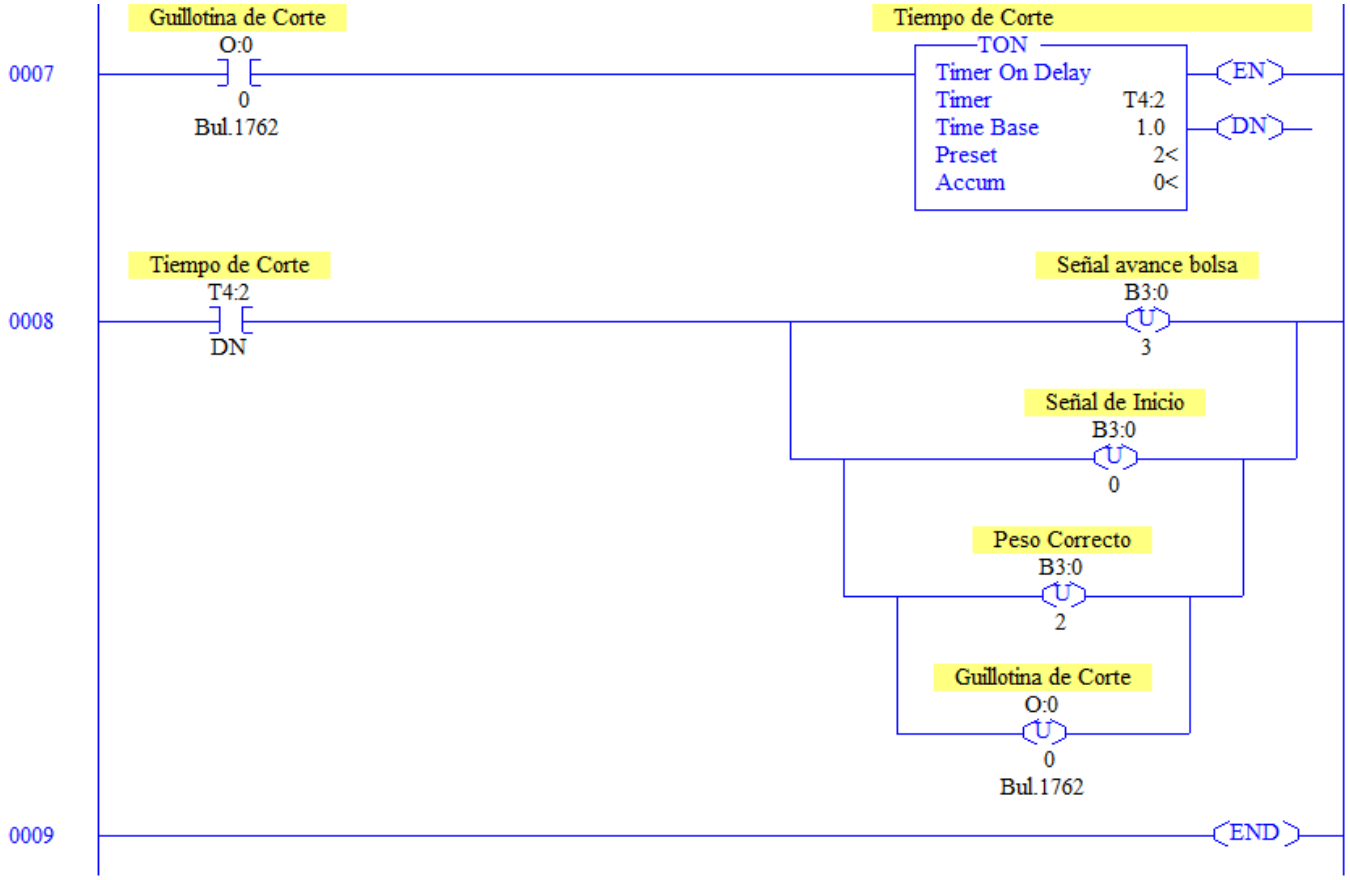


Si el estado del sistema es optimo activaremos el arrancador de un motor llamado "Avance bolsa" este estará activado durante 2 segundos y después el mismo temporizador "Timer Avance Bolsa" detendrá el avance en la línea anterior (línea 0002 y 0003).



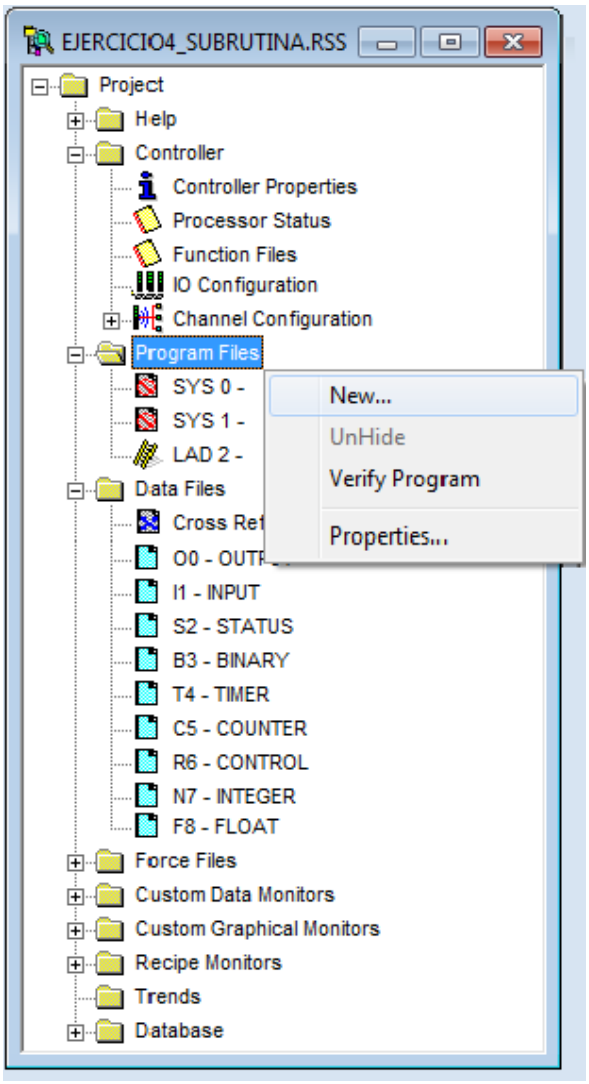
Cuando Avance de bolsa se detenga y una señal proveniente de una celda de carga* este activada, una salida interna retentiva se energizara. Esta salida llamada Peso Correcto es nuestra señal de habilitación para proseguir con la etapa de corte. Después, finalizar el proceso desenclavando todas las salidas retentivas previamente activadas. Los temporizadores utilizados en estas etapas son para definir cuanto tiempo estarán activadas nuestras salidas antes de comenzar la etapa siguiente (línea 0004 a 0009).

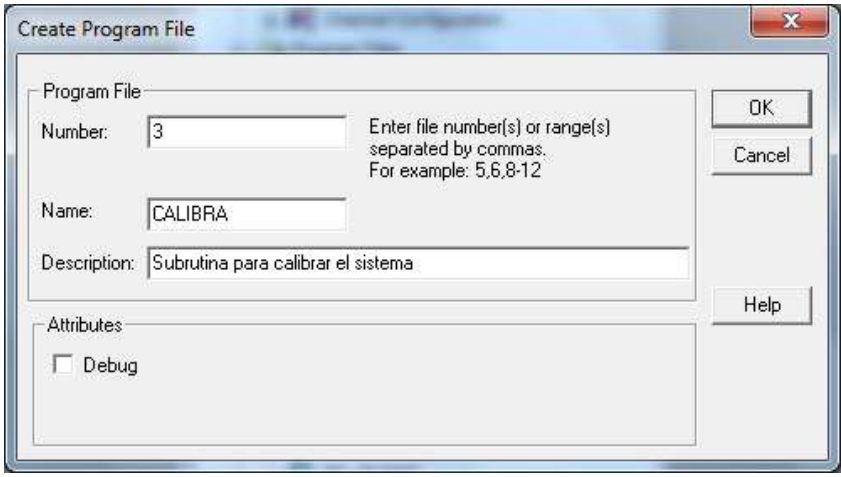


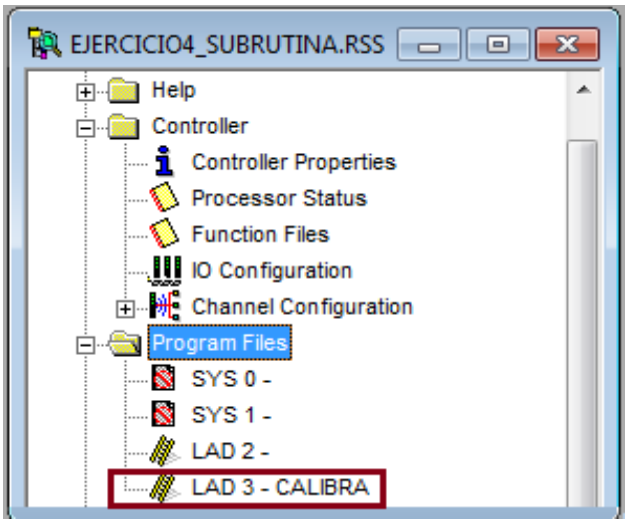


La subrutina de este programa será empleada como una retroalimentación. Estaremos monitoreando un valor de offset. Con una instrucción ADD sumaremos los valores de dos señales analógicas, una proveniente del sensor y otra de un potenciómetro con valores positivos y negativos. Enviaremos el valor a una salida la cual a su vez estará conectada a nuestro sensor de calibración. En el momento en el que el sensor este calibrado óptimamente, el contacto de Error de calibración en la línea de entrada de la subrutina se abrirá y nos enviara de regreso al programa principal.

Para crear este archivo de subrutina diríjase a la carpeta de Project ► Program Files ► New. De clic derecho sobre la carpeta de program files e introduzca un numero decimal de 3 a 255 para referenciar la subrutina.







5.6.- INSTRUCCIONES MATEMATICAS

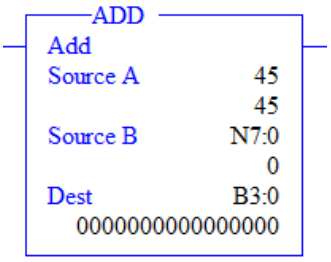
El procesador de MicroLogix 1200 nos permite realizar algunos cálculos usando expresiones u operaciones matemáticas específicas.

Son instrucciones de salida, cuando la línea de entrada sea verdadera la salida realiza la operación indicada y guarda el resultado en una dirección de destino

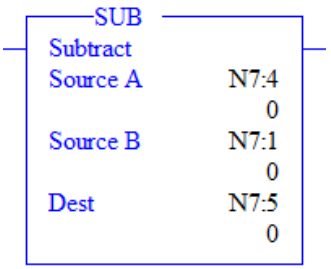
Source A, B: Tanto la fuente A como la fuente B pueden ser constantes de programa o direcciones que contienen valores, sin embargo A y B no pueden ser constantes a la vez.

Dest: Es la dirección de 16 bits donde se guardara el resultado de la operación. Si el resultado excede el límite permitido – 32,768 a 32767, el procesador establece el bit S:0/1 (overflow bit) and S:5/0 (overflow trap bit, major error 0020). Monitoree el bit S:5/0 en el programa para evitar esta peligrosa situación.

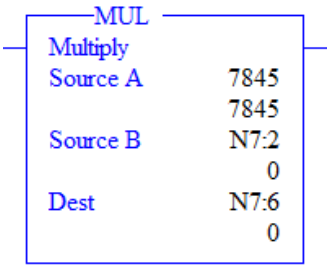
ADD. Suma (Addition).- Realiza una suma entre dos números y los guarda en la dirección indicada.



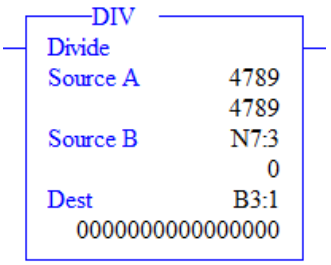
SUB. Resta (Substract).- Realiza una resta entre dos números y los guarda en la dirección indicada.



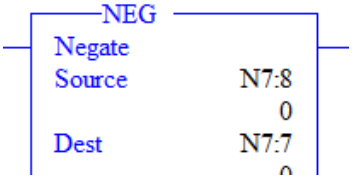
MUL. Multiplicación (Multiply).- Realiza una resta entre dos números y los guarda en la dirección indicada.



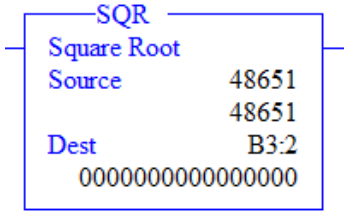
DIV. División (Divide).- Realiza una resta entre dos números y los guarda en la dirección indicada.



NEG. Negación (Negate).- Esta instrucción cambia el signo de un número que se encuentra en una dirección y lo guarda en una diferente. Ambas direcciones tienes que ser direcciones de palabra.



SQR. Raíz Cuadrada (Square Root).- Obtiene la raíz



cuadrada de un número y lo guarda en la dirección indicada. Se debe usar una constante de programa para el valor de la fuente. Es posible usar número negativos.

5.7.- INSTRUCCIONES LOGICAS Y DE MOVIMIENTO

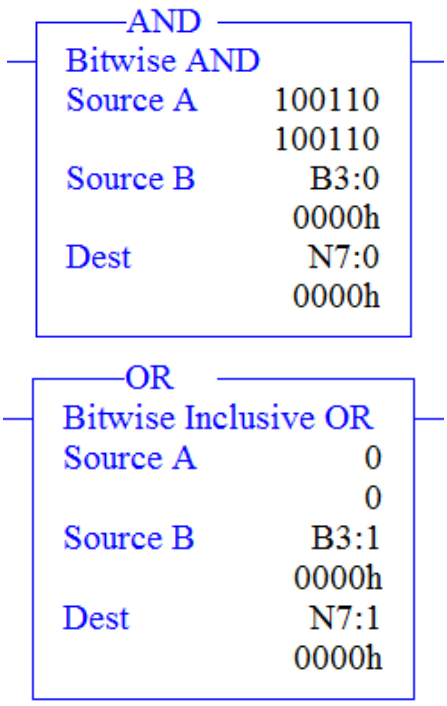
Estas instrucciones de salida le permiten realizar operaciones lógicas o de movimiento sobre palabras individuales.

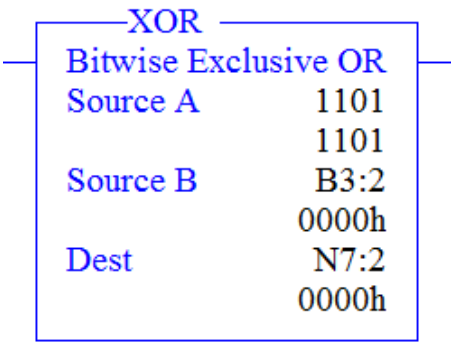
Son 4 operaciones lógicas que se pueden implementar AND, OR, X-OR, NOT. Cuando la condición de la línea de entrada es verdadera, la fuente A y B realizan la operación indicada bit por bit y guarda el resultado en la dirección de destino.

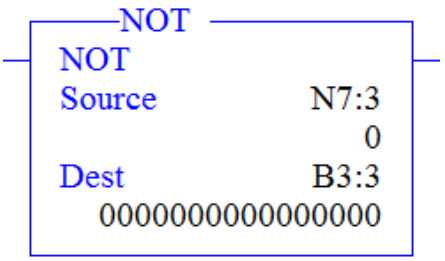
En la instrucción **NOT** el operando de la fuente no puede ser una constante.

Source A, B.- Pueden ser tanto direcciones de palabra o constantes de programa, sin embargo, ambas no pueden ser constantes al mismo tiempo.

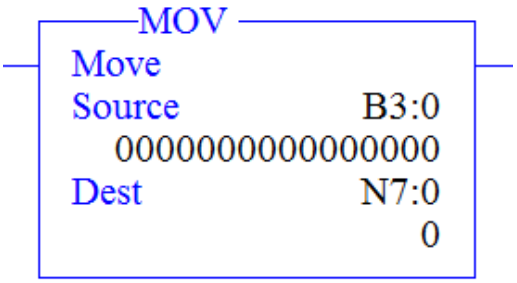
Dest.- Debe ser una dirección de palabra, este es el lugar de destino donde se almacena el resultado de la operación.







MOV. Mover (Move).- Cuando la línea de entrada de esta instrucción de salida es verdadera, se mueve una copia de la fuente hacia la palabra de destino en cada escaneo de programa. El valor original de la fuente permanece intacto y sin cambios en su dirección.

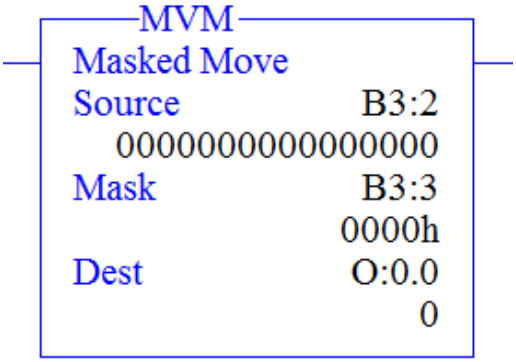


Source.- Es la dirección de el dato que se va a mover. La fuente puede ser una constante.

Destination.- Es la dirección final de la información que se va a mover.

Si usted desea mover una palabra de datos sin afectar las banderas matemáticas, use la instrucción Copy (**COP**) con una longitud de una palabra (16 bits) en lugar de la instrucción **MOV.**

MVM. Mover con Mascara (Masked Move).- Al igual que la instrucción MOV te permite mover un dato de una palabra a otra de destino. A diferencia de MOV, MVM permite seleccionar solo un segmento de la información que se va a mover a través de una mascara.



Source.- Es la dirección de el dato que se va a mover. La fuente puede ser una constante.

Destination.- Es la dirección final de la información que se va a mover.

Mask.- Es la instrucción de la máscara a través de la cual se moverán los datos. La máscara puede ser un valor

hexadecimal, binario o decimal, el procesador realizara la conversión necesaria y mostrara el valor hexadecimal.

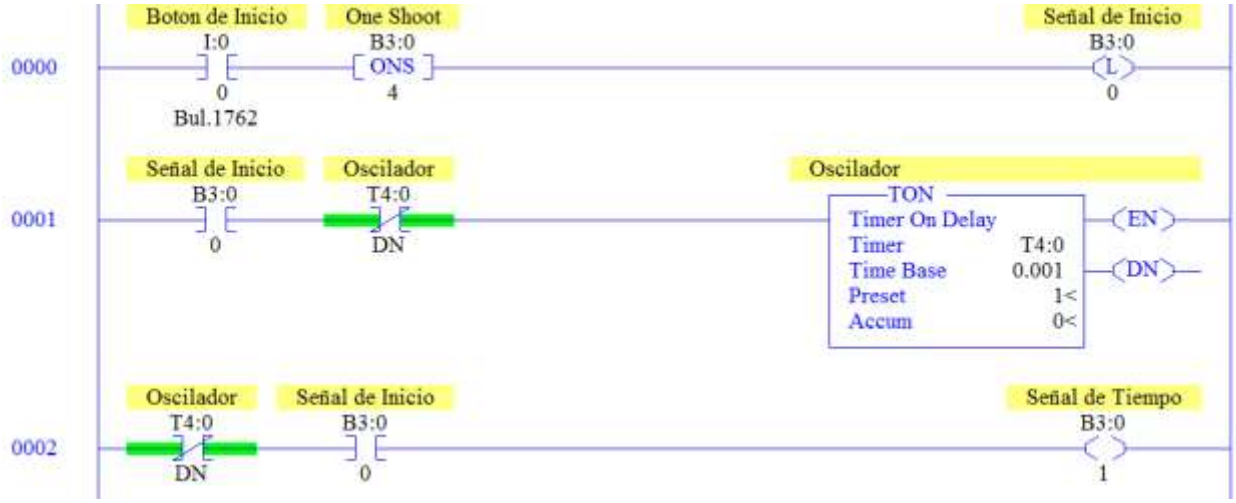
Ejercicio 5: Con las herramientas que ha adquirido hasta el momento, obtenga la potencia de un número.

Solución: Debido a que en el procesador del MicroLogix 1200 no podemos usar ninguna instrucción matemática para calcular la potencia de un número, utilice la instrucción **MUL** para obtener el resultado.

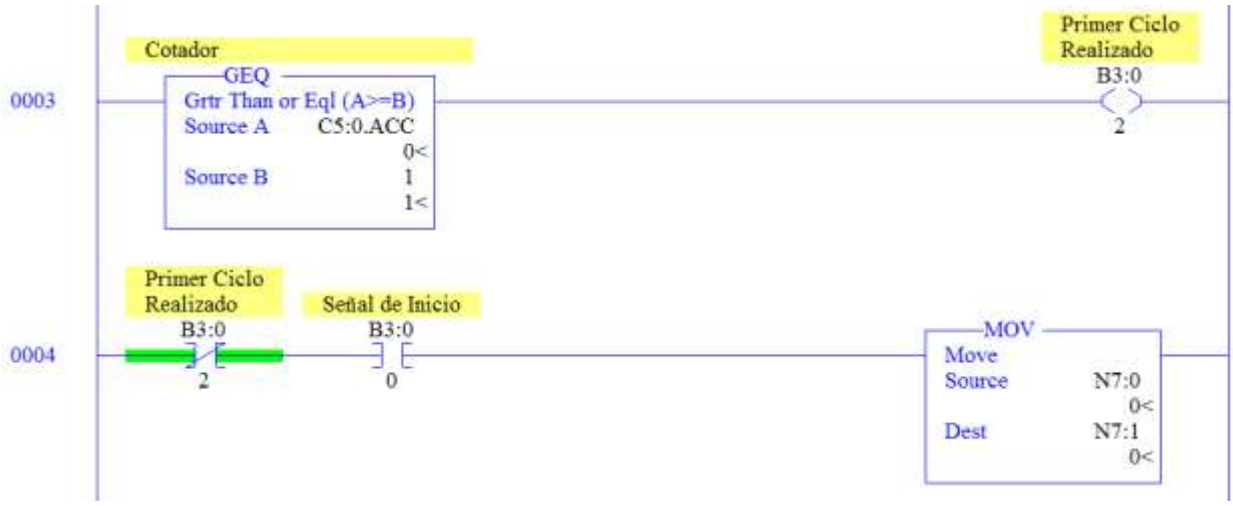
Diagrama:

Con la misma estructura en nuestro Botón de Inicio de ejercicios anteriores, iniciaremos un oscilador simple el cual también ya habíamos construido. El One Shot B3:0/4 de la línea 0000 es para evitar una señal en falso debida a que Botón de Inicio haya permanecido activado hasta el siguiente escaneo de programa, de esta manera nos aseguramos a no hacer más de un cálculo de la potencia de un numero por cada número que introducimos.

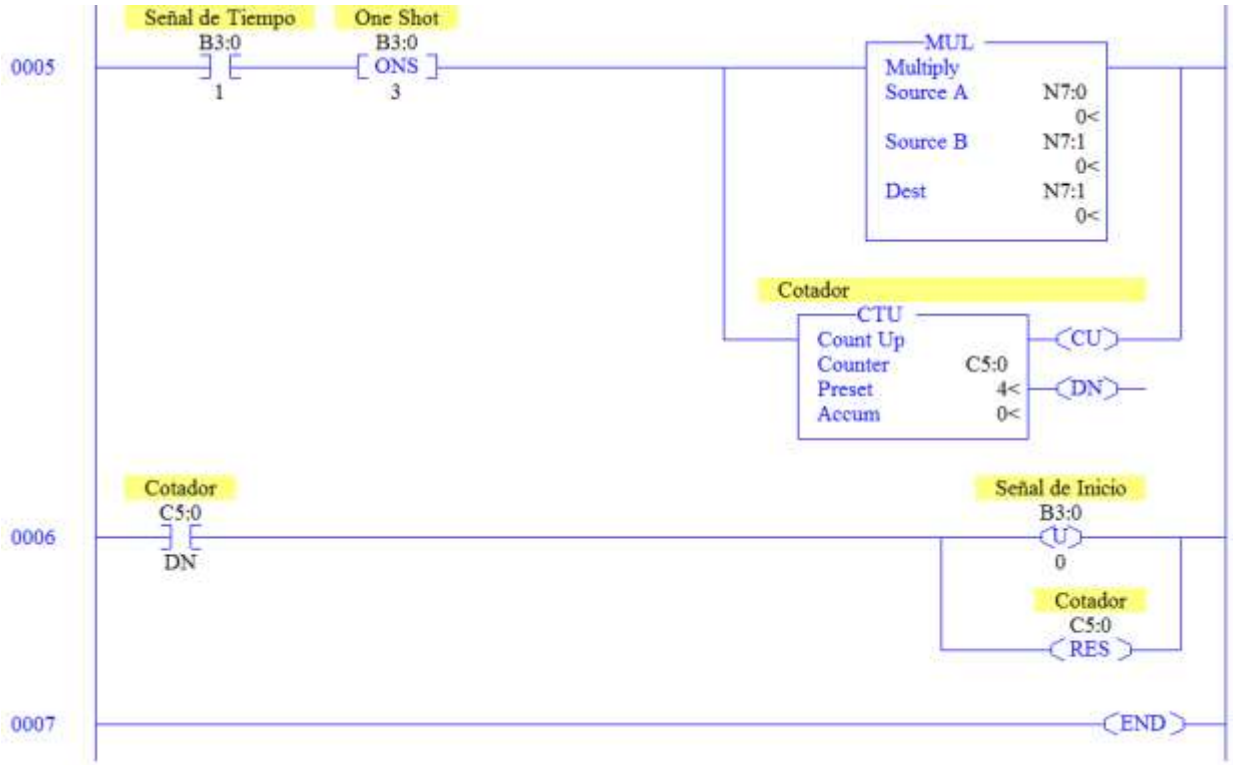
Usamos una base de tiempo pequeña en el Oscilador (1 milésima de segundo), queremos que el cálculo se realice lo más rápido posible y solo necesitamos la transición de falso a verdadero en la línea de entrada de la instrucción MUL en 0005.



Para efectuar la operación deseada, haremos un diagrama sencillo que nos permita visualizar fácilmente nuestro resultado. Ya que la potencia de un numero es simplemente ese número multiplicado cierto número de veces por sí mismo, resolveremos el problema de la siguiente manera; N7:0 será nuestro numero a elevar a una potencia dada, en el escaneo de programa comprobamos si es el primer paso que realiza nuestro diagrama, con un **GEQ** monitoreamos si el acumulador del contador es mayor o igual a cero, si es así, moveremos el valor de N7:0 hacia N7:1. Recuerde que solo realizamos este paso en la primera ocasión, de esta manera N7:1 nos indica por cual número debemos realizar las siguientes multiplicaciones.



Por último utilizaremos una instrucción **MUL** para multiplicar N7:0 y N7:1 una vez por cada transición falso-verdadero de nuestra Señal de Tiempo. El Contador que se encuentra en la misma rama que **MUL** es el que nos indica cuantas veces realizaremos la multiplicación, dicho de otra forma, su Preset es la potencia a la cual elevaremos el numero en N7:0. Dese cuenta que al ya no realizar la instrucción **MOV**, N7:1 guarda el resultado de la multiplicación para poder seguir multiplicando ese valor. Nuestra secuencia terminara cuando el bit DN del Contador se establezca en "1" lógico y reinicie al mismo Contador y desenclave la Señal de Inicio.



5.8.- INSTRUCCIONES DE SECUENCIADOR

Las instrucciones de secuenciador son usadas en operaciones concretas que se repiten una y otra vez, como en maquinas ensambladoras automáticas.

La ventaja principal de un secuenciador es su capacidad para manejar información en una sola instrucción, controlar hasta 16 salidas discretas en una sola línea y ahorrar memoria de programa.

Se puede usar archivos de entero o binarios definidos por el usuario con las instrucciones de secuenciador.

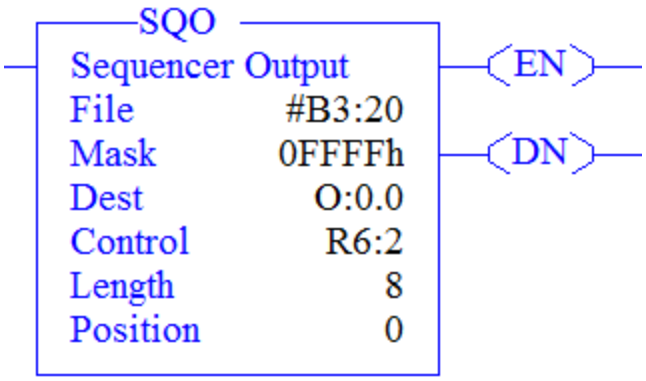
Si la aplicación requiere de más de 16 bits se pueden múltiples secuenciadores en paralelo, teniendo en cuenta que no se traslapen los rangos de estos.

Las instrucciones de secuenciador alteran el contenido de S:24 (index register).

SQO. Secuenciador de Salida (Sequencer Output).- En cada transición falso-verdadero, esta instrucción se mueve un paso a través del archivo de secuencia programado, transfiriendo la información filtrada por una máscara a una palabra de destino. El bit DONE (DN) se establece en "1" lógico cuando la última palabra de el archivo ha sido transferida. En la siguiente transición, la instrucción se restablece en el paso 1.

También es posible utilizar la instrucción RES para restablecer el secuenciador. Todos los bits de control se restablecerán en cero, así como la posición del mismo.

File.- Es la dirección del archivo del secuenciador. Debe usar un indicador (#) antes de el archivo. Este archivo guarda la información de referencia para monitorear las entradas



Mask.- Es una constante o valor hexadecimal de la dirección de la máscara por la cual se moverá la información. Si la máscara es un archivo, su longitud será igual a la del archivo del secuenciador.

Si el valor es decimal o binario, el procesador realizara la conversión necesaria para mostrar el valor hexadecimal.

Destination.- Es la dirección de la palabra o archivo de salida para SQO hacia donde la instrucción mueve la información del archivo de secuenciador.

Control.- Es una dirección de archivo de control. El bit de estado, la longitud del stack y la posición son guardados en este elemento. No use esta dirección para ninguna otra instrucción.

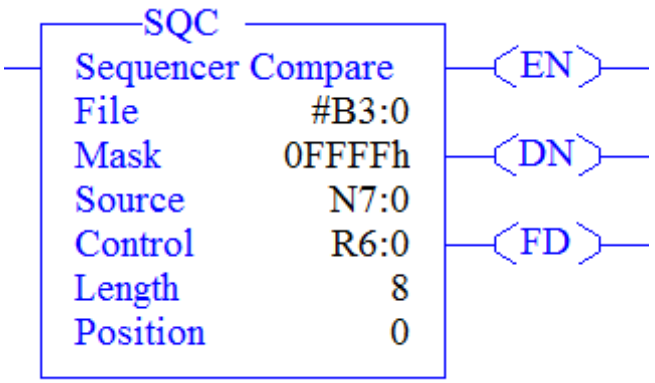
Length.- Es el numero de pasos del secuenciador empezando en la posición 1. Con un máximo de 255 palabras. La posición 0 es la posición inicial. Al reiniciar la cuenta la instrucción comienza

desde la posición número 1, si se ingresa un numero de longitud igual a 10, el número de elementos será de 11 al incluir la dirección inicial.

Position.- Es el paso o la ubicación en la cual se encuentra el archivo de secuenciador.

SQC. Secuenciador de Comparación (Sequenccer Comapre).- En cada transición falso-verdadero, esta instrucción se mueve un paso en el archivo de secuenciador direccionado a través de una máscara, en cada paso compara el valor de una palabra fuente con la palabra del archivo en curso, de encontrar una igualdad el bit FOUND (FD) se establece en "1" lógico. La instrucción termina al comparar la última palabra del archivo de secuenciador y el bit DN se establece en "1" lógico.

File.- Es la dirección del archivo del secuenciador. Debe usar un indicador (#) antes de el archivo. Este archivo guarda la información de referencia para monitorear las entradas



Mask.- Es una constante o valor hexadecimal de la dirección de la máscara por la cual se moverá la información. Si la máscara es un archivo, su longitud será igual a la del archivo del secuenciador.

Si el valor es decimal o binario, el procesador realizara la conversión necesaria para mostrar el valor hexadecimal.

Source.- Es la dirección de la palabra de entrada o archivo del cual el SQC obtiene la información para comparar con el archivo de secuenciador.

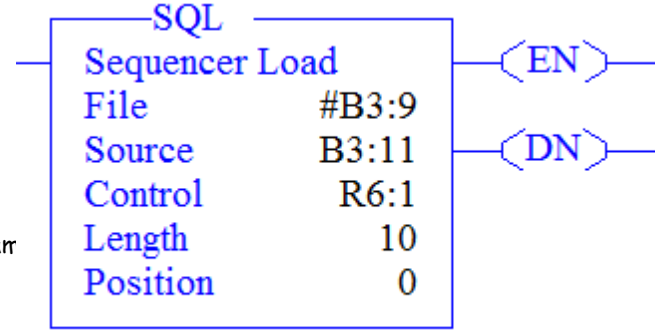
Control.- Es una dirección de archivo de control. El bit de estado, la longitud del stack y la posición son guardados en este elemento. No use esta dirección para ninguna otra instrucción.

Length.- Es el numero de pasos del secuenciador empezando en la posición 1. Con un máximo de 255 palabras. La posición 0 es la posición inicial. Al reiniciar la cuenta la instrucción comienza desde la posición número 1, si se ingresa un numero de longitud igual a 8, el número de elementos será de 9 al incluir la dirección inicial.

Position.- Es el paso o la ubicación en la cual se encuentra el archivo de secuenciador.

SQL. Secuenciador de almacenamiento (Sequencer Load).- En cada transición falso-verdadero, esta instrucción se mueve un paso en el archivo de secuenciador y almacena una palabra fuente en la palabra del archivo en el que se encuentra.

File.- Es la dirección del archivo del secuenciador. Debe usar un indicador (#) antes de el archivo. Este archivo guarda la información de referencia para monitorear las entradas.



Source.- Puede ser una dirección de palabra, una dirección de archivo o una constante de programa. Tambien es posible utilizar los datos del archivo de entradas y salidas. Si la fuente es una dirección de archivo, la longitud del archivo se iguala al archivo del secuenciador de carga. Los dos archivos avanzaran automáticamente por el valor de posición.

Control.- Es una dirección de archivo de control. El bit de estado, la longitud del stack y la posición son guardados en este elemento. No use esta dirección para ninguna otra instrucción.

Length.- Es el numero de pasos del secuenciador empezando en la posición 1. Con un máximo de 255 palabras. La posición 0 es la posición inicial. Al reiniciar la cuenta la instrucción comienza desde la posición número 1, si se ingresa un numero de longitud igual a 8, el número de elementos será de 9 al incluir la dirección inicial.

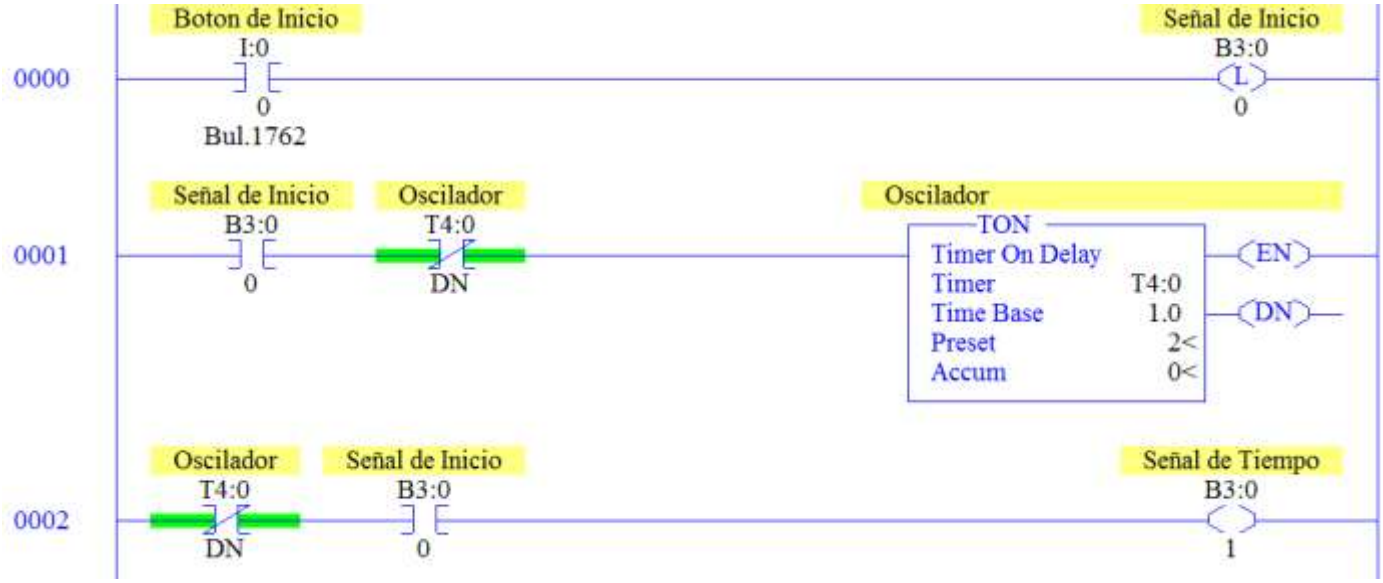
Position.- Es el paso o la ubicación en la cual se encuentra el archivo de secuenciador.

Ejercicio 6: Construya un Contador BCD

Solucion: Utilice un secuenciador de salida SQO, recuerde que esta instrucción le permite manejar una mayor cantidad de bits, de esta forma será más simple implementar su secuencia.

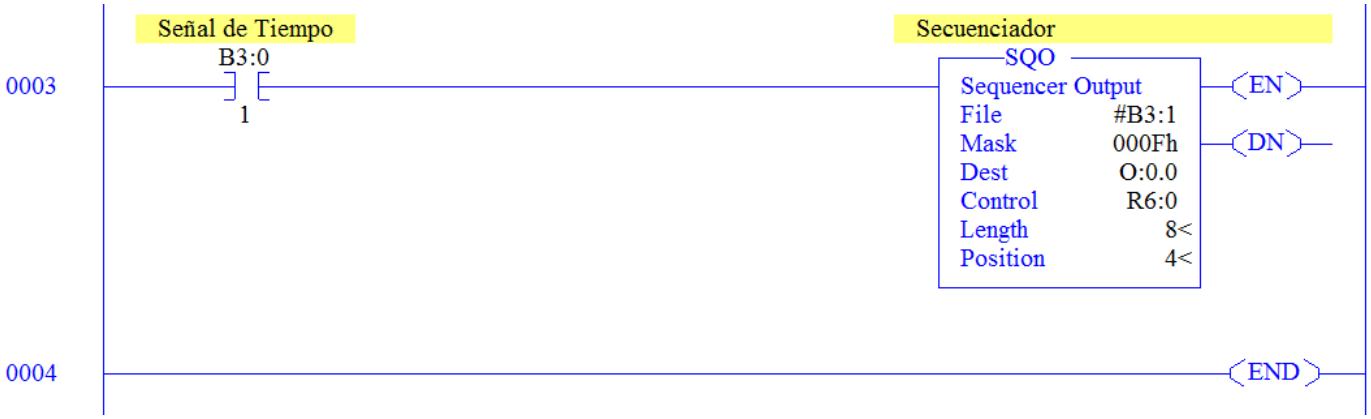
Diagrama:

Utilice el Oscilador Simple que se implemento en el tema 4.2, con esta señal se realizara el control en la línea de entrada del secuenciador.



Con la Señal de Tiempo se realizara el control en la línea de entrada del secuenciador. En cada transición falso-verdadero, el **SQO** moverá su posición a través del archivo de secuenciador,

de esta forma solo tenemos que programar en los archivos binarios la secuencia deseada y el secuenciador enviara a las salidas los datos que vayamos a introducir.



Usaremos una mascara con valor 000F hexadecimal para solo enviar los primeros 4 valores de el archivo binario.

Recuerde que la posición inicial del secuenciador es 0, pero al reiniciar se establecerá la posición en 1. Es importante tener esto en cuenta para no traslapar las palabras de los archivos y que nuestra secuencia se ejecute de la manera deseada.