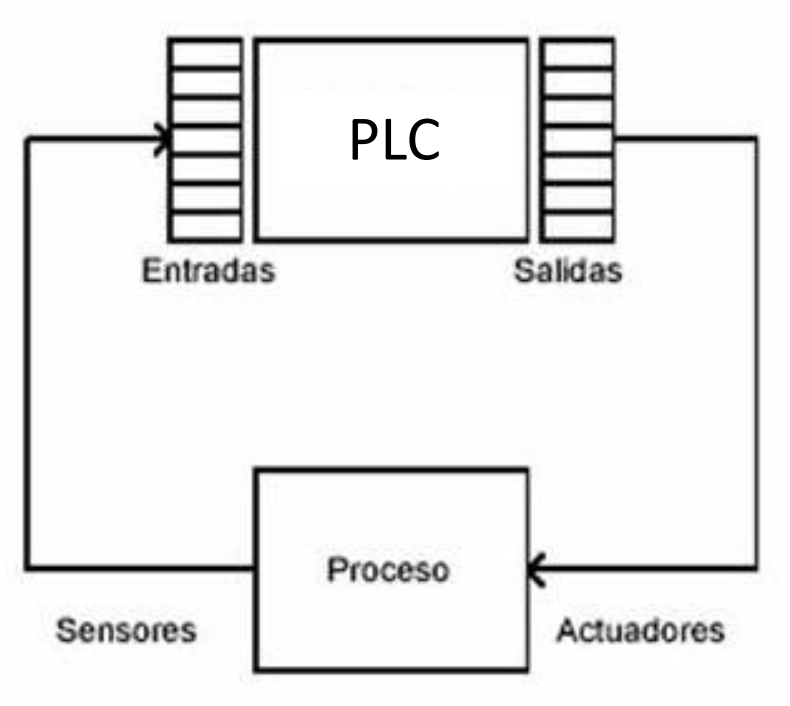
Introducción - Repaso

Controladores Lógicos Programables

¿Qué es un PLC?

- **P**rogrammable **L**ogic **C**ontroller
  - Controlador
  - Programable
  - Hardware + Software
  - Aplicación: Automatización / Control Industrial
  - Interconexión con proceso/máquina a controlar (“campo”) estandarizado

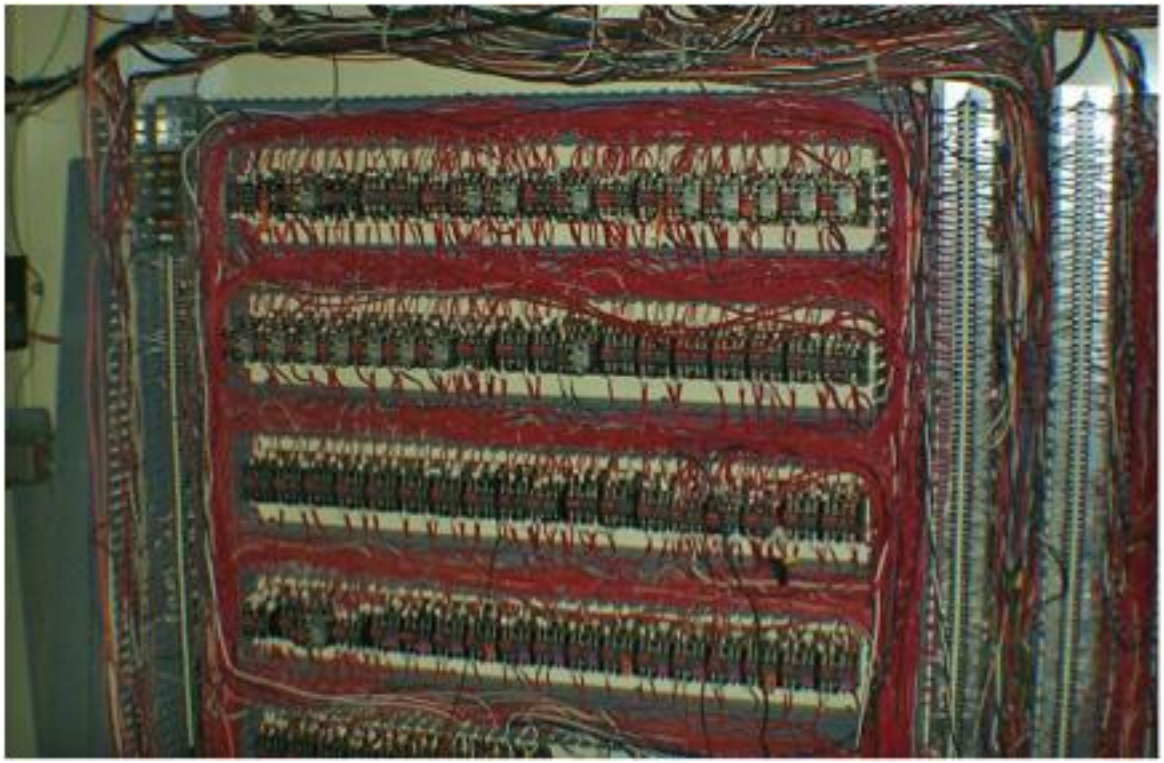
¿Qué es un PLC?



- Proceso representado en el PLC por las E/S
- Cantidad de señales define el tamaño del PLC

Historia de los PLCs

Tablero con lógica de relé



Fuente: http://www.xl-technology.com/control-upgrades.html

Arquitectura de un PLC

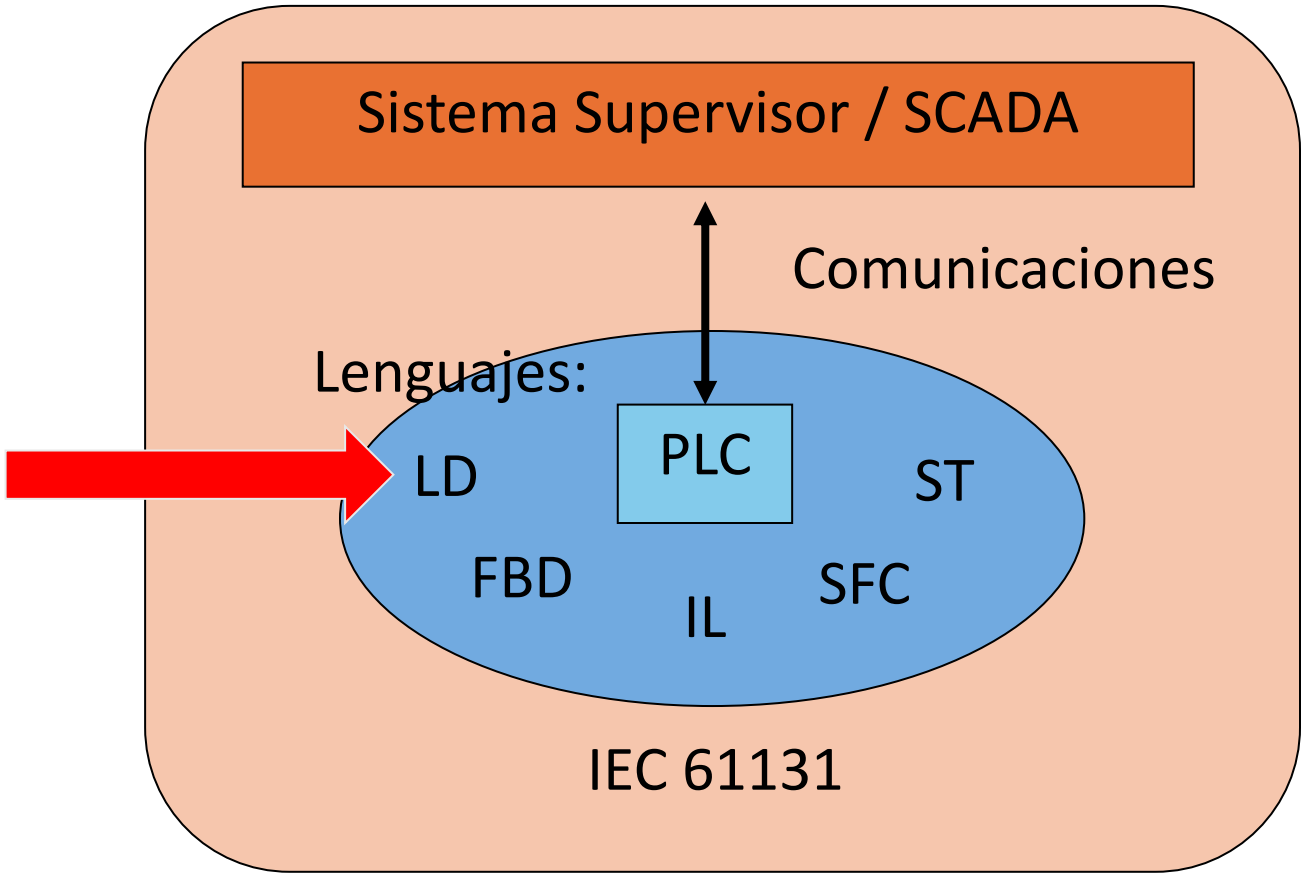
Un PLC se compone de una CPU, memoria, entradas y salidas

Diferencias con PC:

- El PLC tiene menos recursos, no tiene interfaz gráfica
- El hardware del PLC es de mejor calidad, mayor confiabilidad
- El sistema operativo del PLC es más robusto

IMPORTANTE: La seguridad de un sistema no depende de un PLC estándar

Programa del Curso



Proyectos / Aplicaciones

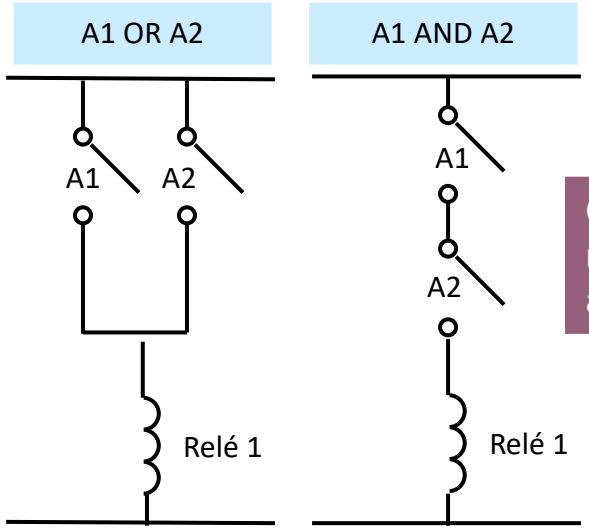
Lenguage LD
(Ladder Diagram)

Controladores Lógicos Programables

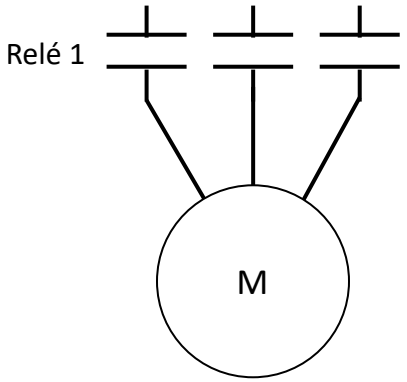
Orígenes históricos

Ejemplo 1: encendido de motor M por distintas lógicas de dos llaves A1 y A2

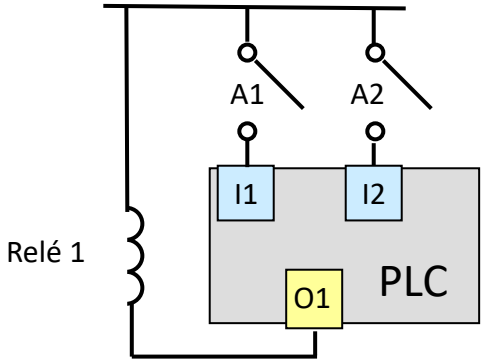
Lógica de relé



Cambio de lógica (OR a AND) requiere recableado paralelo a serie



PLC



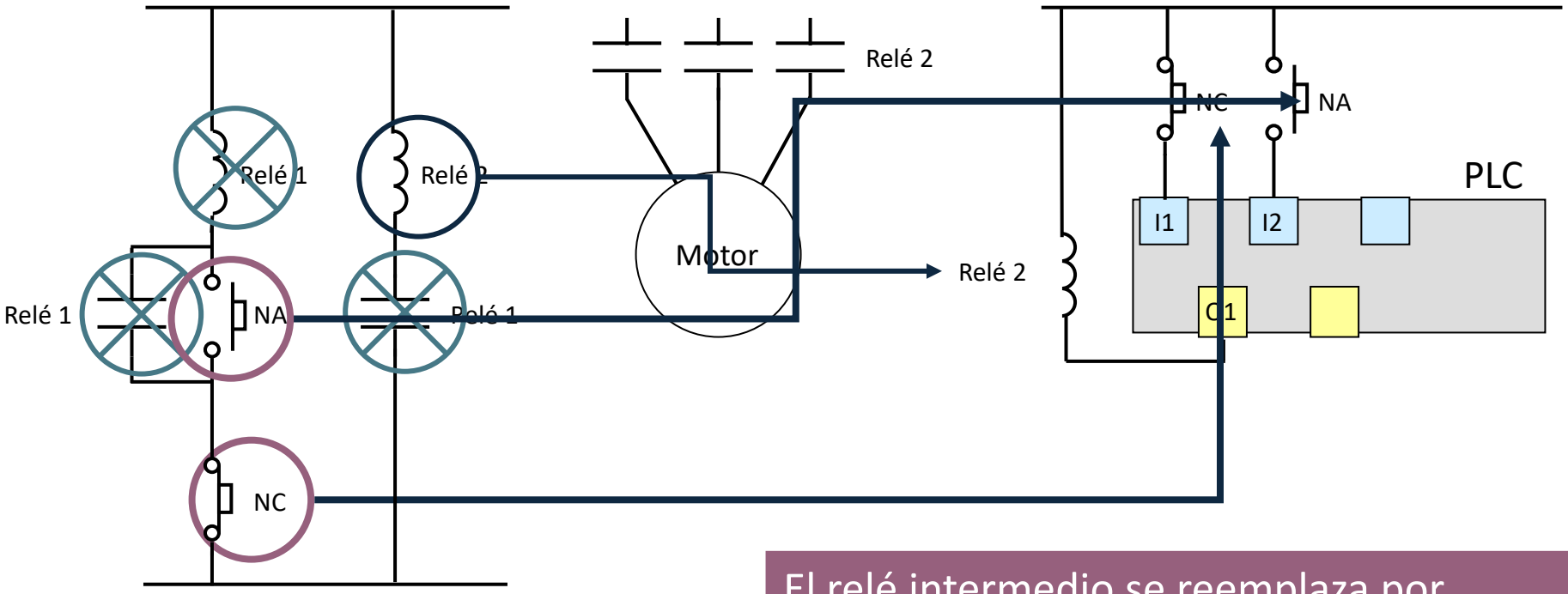
Cambio de lógica (OR a AND) se resuelve por software del PLC

Orígenes históricos

Ejemplo 2: encendido-apagado de motor por pulsadores

Circuito por lógica de relé

Circuito por PLC



El relé intermedio se reemplaza por software en el PLC

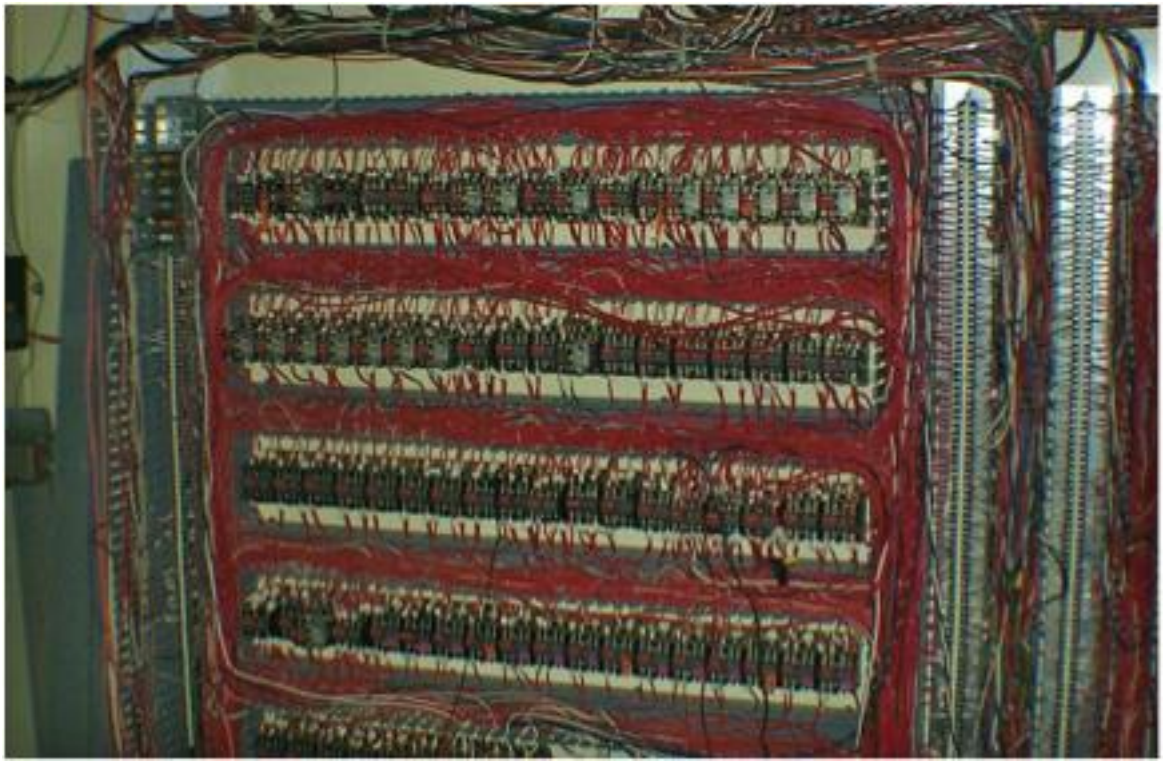
Orígenes históricos

Objetivos del PLC:

- Aumentar la confiabilidad
- Aumentar la flexibilidad
- Mantener la facilidad de soporte

Historia de los PLCs

Tablero con lógica de relé



Tablero con PLC

Fuente: http://www.xl-technology.com/control-upgrades.html

Orígenes históricos

Objetivos del PLC:

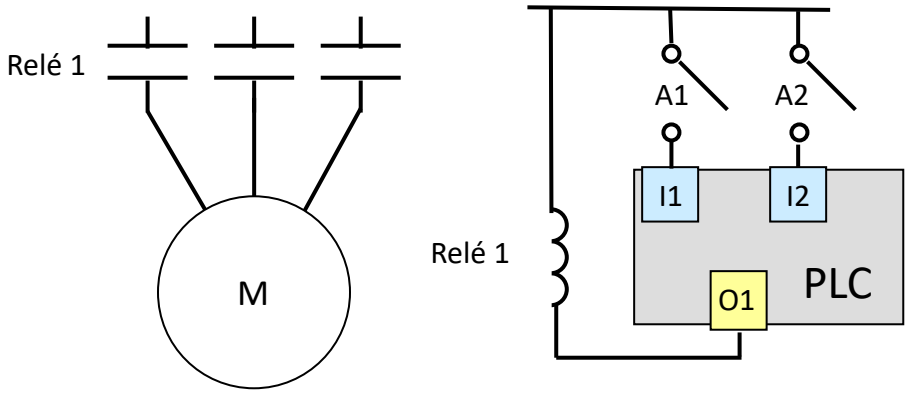
- Aumentar la confiabilidad
- Aumentar la flexibilidad
- Mantener la facilidad de soporte

Lenguaje de programación fácilmente entendido por electricistas de planta

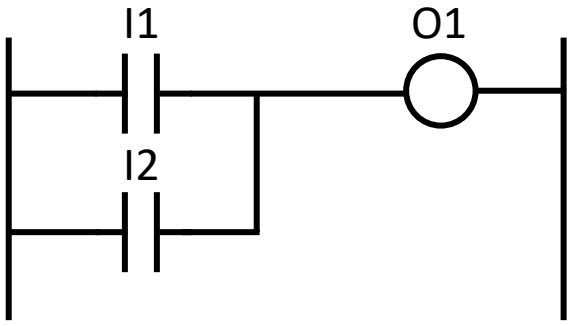
LADDER (LD)

Orígenes históricos

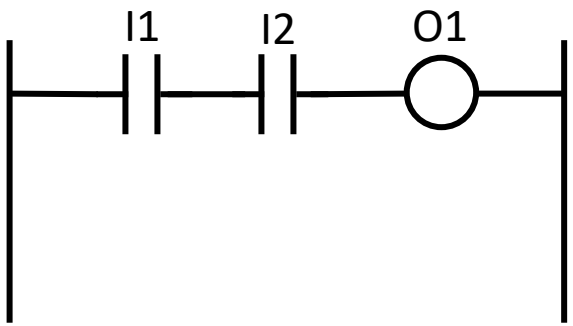
Programas LD del ejemplo 1



Encendido por I1 OR I2

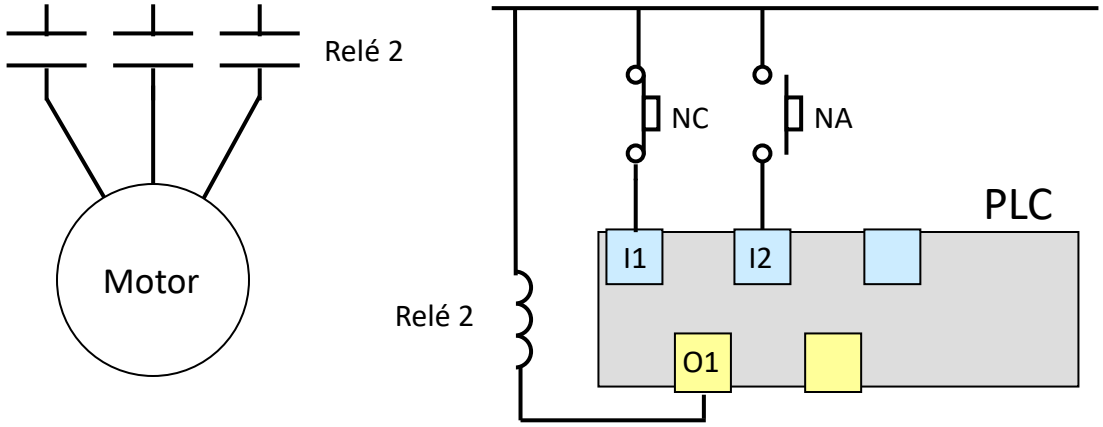


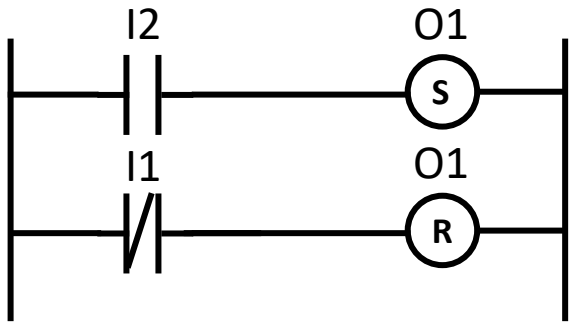
Encendido por I1 AND I2



Orígenes históricos

Programa LD del ejemplo 2





Estructura programa LD

Lenguaje gráfico

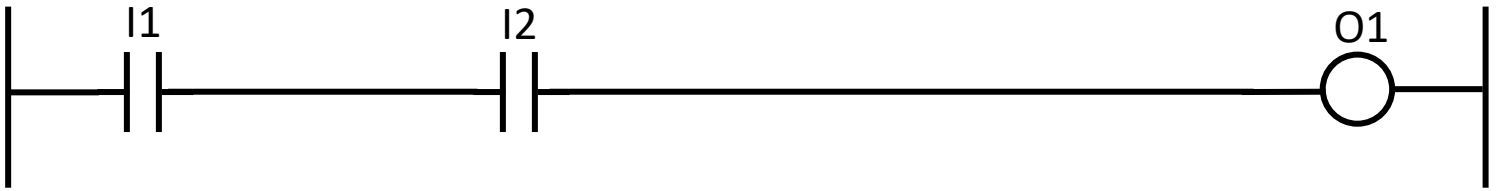
Programa consiste en una secuencia de escalones (rungs)

Estructura de cada escalón:

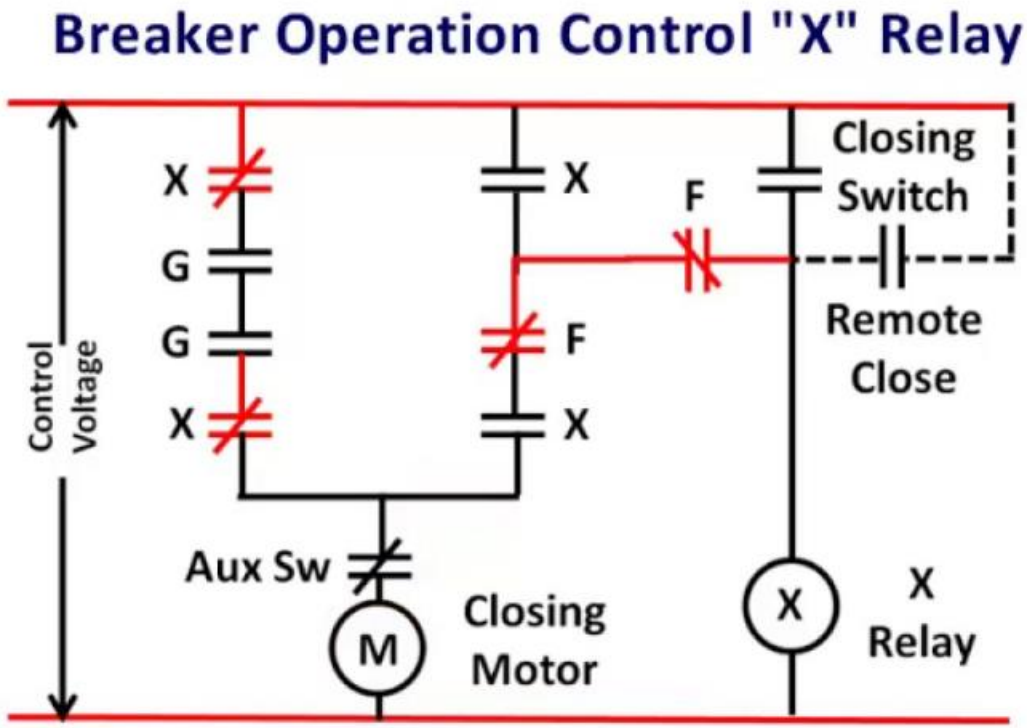
1. Comienza en una barra de alimentación izquierda (positivo de la fuente)
2. Condiciones y acciones, conectadas por líneas de conexión
3. Termina en una barra de alimentación derecha (negativo de la fuente)

Los escalones se ejecutan de arriba hacia abajo

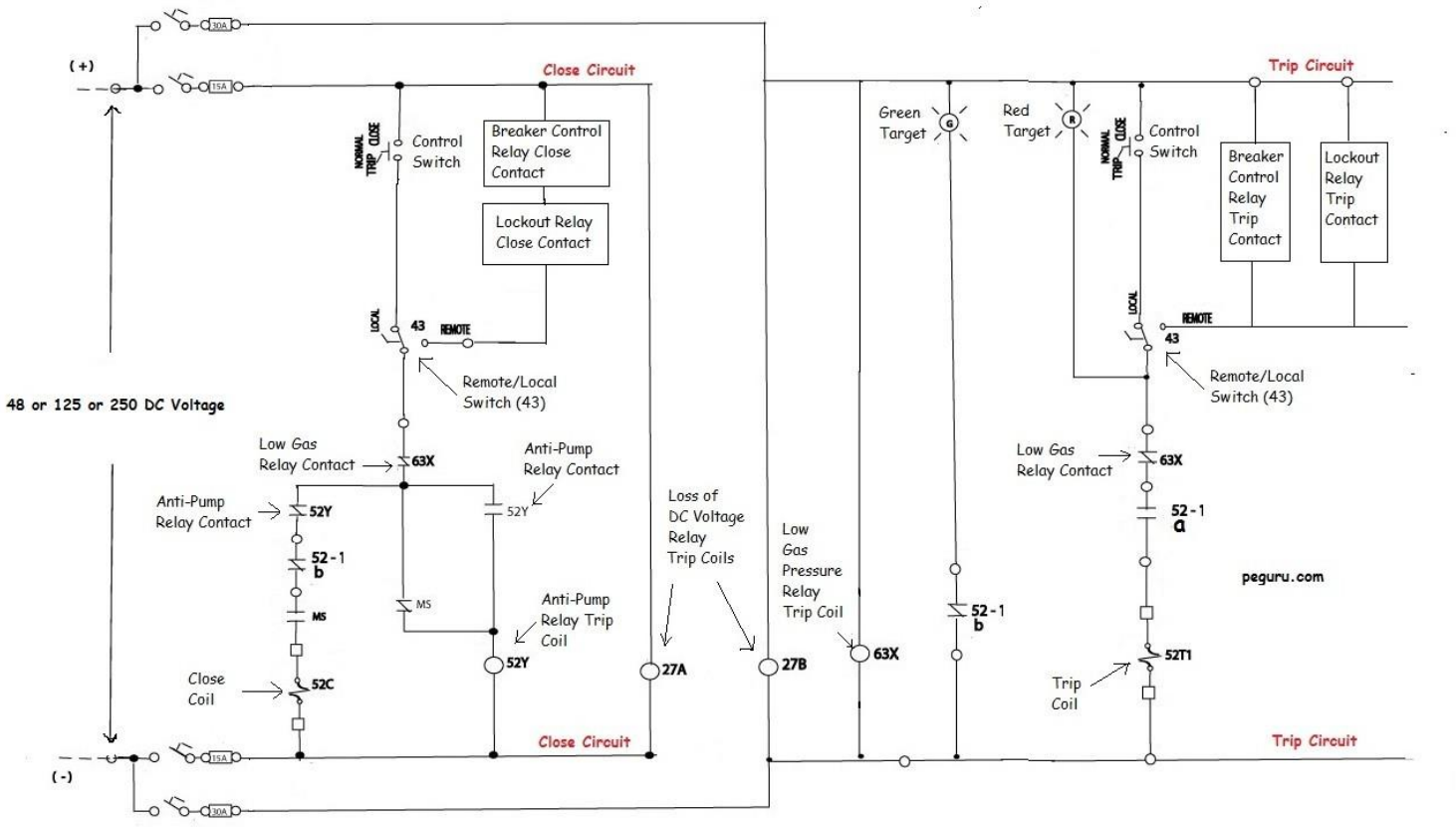
Cada escalón se ejecuta de izquierda a derecha



Circuitos de Lógica de Relé



Circuitos de Lógica de Relé



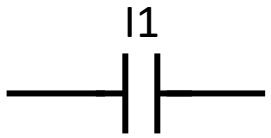
Diferencias entre Fabricantes

Diferencias entre fabricantes: implementación y nomenclatura de las instrucciones

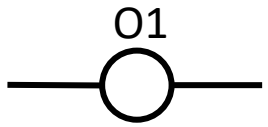
Este curso usa la nomenclatura de los PLCs del laboratorio, según estándar

Símbolos Básicos

Contacto (entrada)



Bobina (salida)



Cada símbolo tiene asociado un bit de memoria

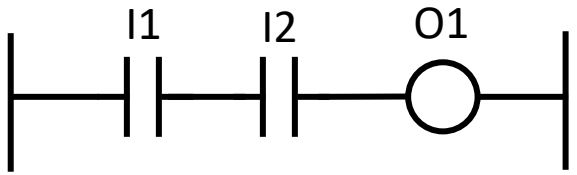
Un bit de memoria se refiere por:

- Su dirección
- Su etiqueta (label)

Operaciones Básicas

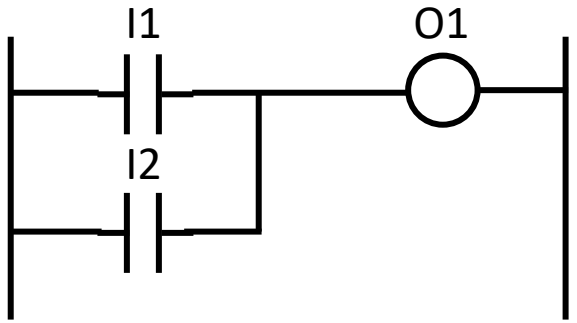
AND (conexión serie):

(O1 = 1) si (I1 = 1) y (I2 = 1)



OR (conexión paralelo):

(O1 = 1) si (I1 = 1) o (I2 = 1)



Instrucciones con BITs

Contacto directo

Contacto normalmente abierto

- Verdadero si bit vale 1

Contacto invertido

Contacto normalmente cerrado

- Verdadero si bit vale 0

Bobina directa

Análoga a la bobina de un relé

- Si el escalón es 1, escribe 1 en bit asociado
- Si el escalón es 0, escribe 0 en bit asociado

Bobina invertida

Función inversa de bobina directa

SET (o LATCH)

Instrucción de salida retentiva

- Si el escalón es 1, escribe 1 en bit asociado
- Si el escalón es 0, no hace nada

RESET (UNLATCH)

Se usa en conjunto con el SET

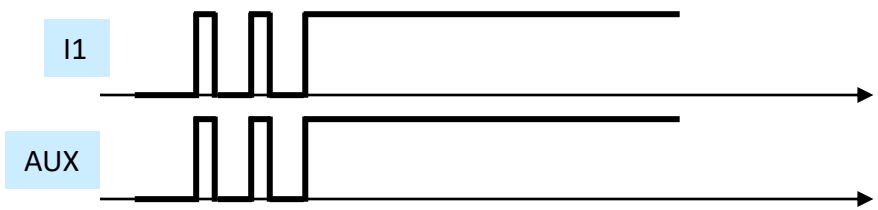
- Si el escalón es 1, escribe 0 en bit asociado
- Si el escalón es 0, no hace nada

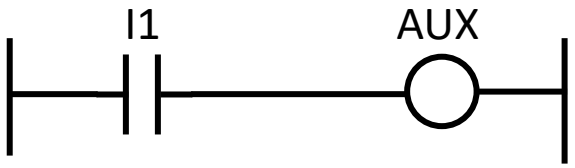
Instrucciones con BITs

Ejemplo: BOBINA vs. SET/RESET

Contador de pulsos con rebotes

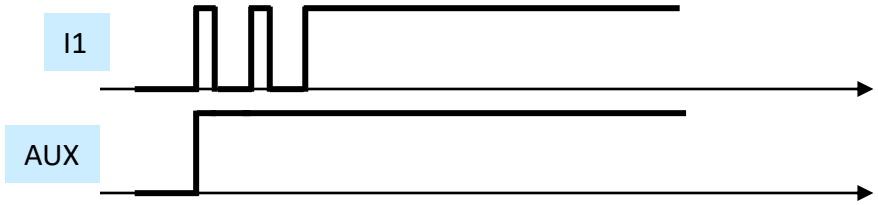
Solución con bobina

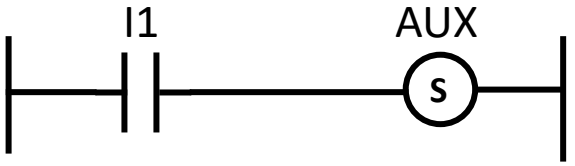




Siguen instrucciones para contar pulsos

Solución con SET

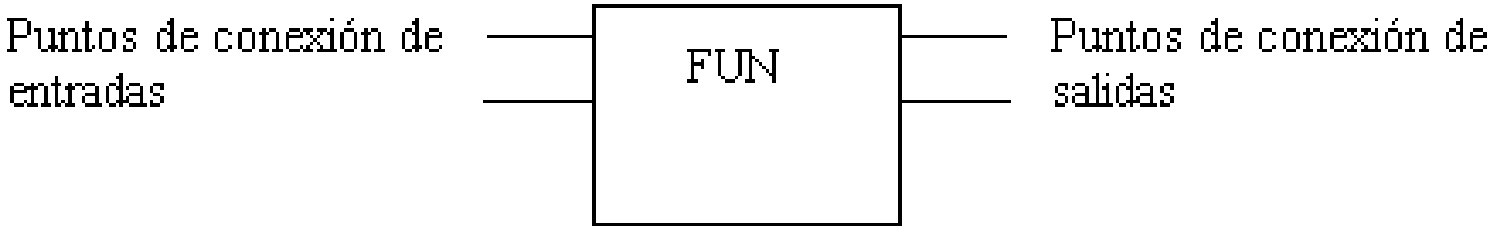




Siguen instrucciones para contar pulsos
y realizar RESET luego de filtrado

Bloques Funcionales

- Las instrucciones de aquí en adelante se representan gráficamente como bloques funcionales
- Bloque funcional: objeto gráfico que se representa por un rectángulo, con puntos de conexión de entradas, conexión de salidas y un identificador
- El identificador describe función del bloque
- Las entradas y salidas son datos. El tipo de dato de cada una depende del bloque.



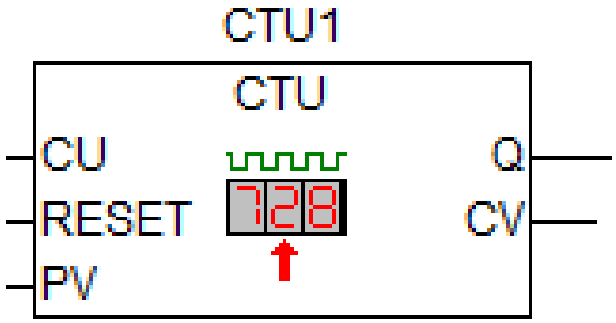
Contadores

- Existen tres tipos:
  - UP - Counters: CTU
  - DOWN – Counters: CTD
  - UP-DOWN Counters: CTUD

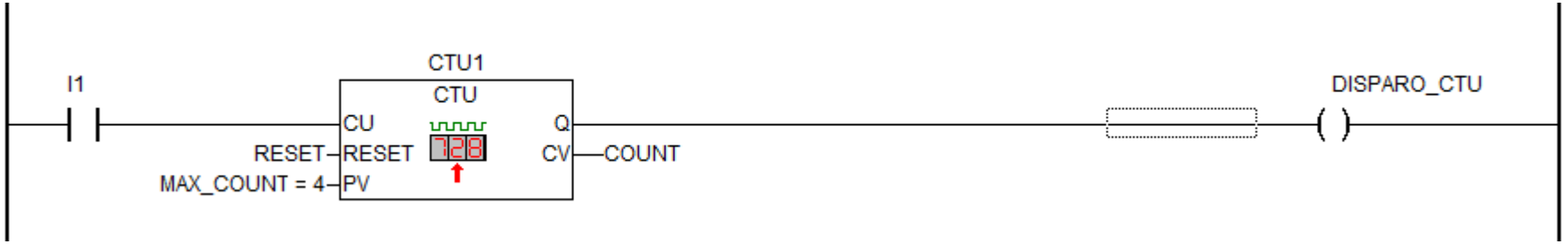
- Rango de cuenta: depende de fabricante. En PLCs de laboratorio:
  - CTU cuenta desde 0 a 32767
  - CTUD cuenta desde 0 a 32767

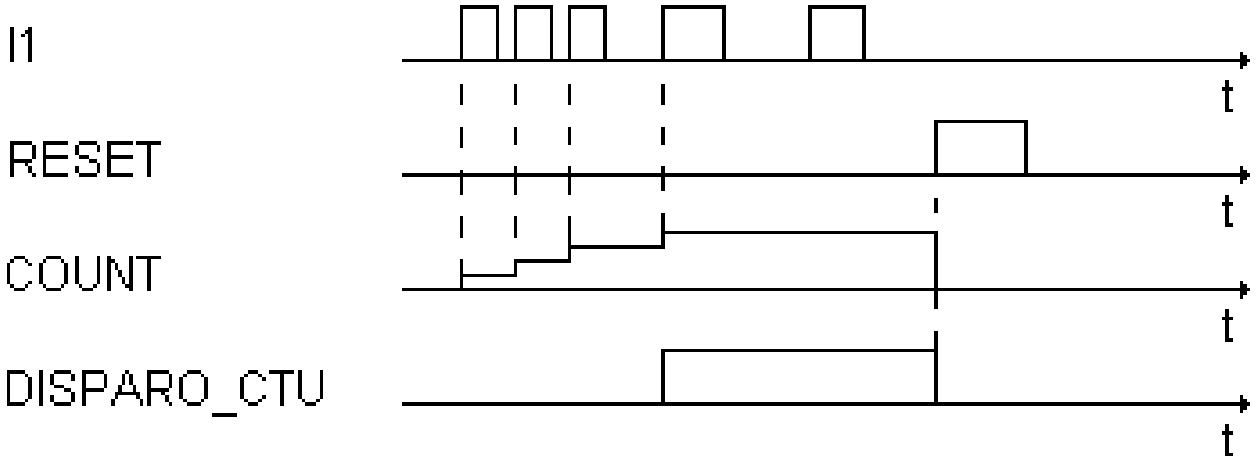
Contadores

- Entrada CU (bit): “Pulso”, se conecta al tren de pulsos que se cuentan
- Entrada RESET (bit): escribe 0 en acumulador
- Entrada PV (Word): límite máximo de cuenta (Preset Value)
- Salida Q (bit): “Done”, indica si acumulador >= PV
- CV (Word) = cuenta, acumulador (Current Value)



Contadores: Ejemplo





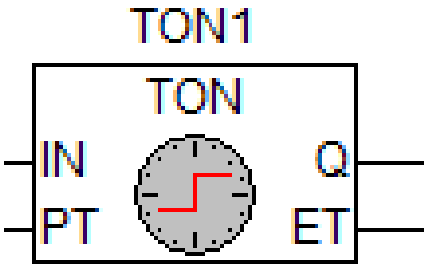
- Observación: el contador se incrementa a intervalos de tiempo variables (dependen del pulso I1)

Timers

- Timer: instrucción destinada a esperar cierto tiempo antes de una acción
- Tres tipos de timers:
  - TON: Timer On Delay (retardo en el encendido): luego que la entrada pasa a 1 durante X seg, la salida pasa a 1.
  - TOF: Timer Off Delay (retardo en el apagado): luego que la entrada pasa 0 durante X seg, la salida pasa a 0.
  - Timers retentivos: no existen en los PLCs de lab. Cuenta el tiempo que la entrada es 1, congelando la cuenta con cambios de 1 a 0.

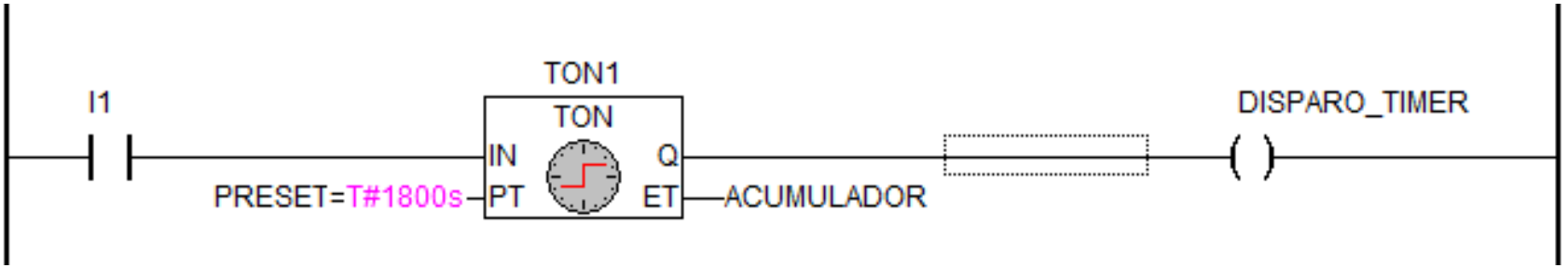
Timers: TON

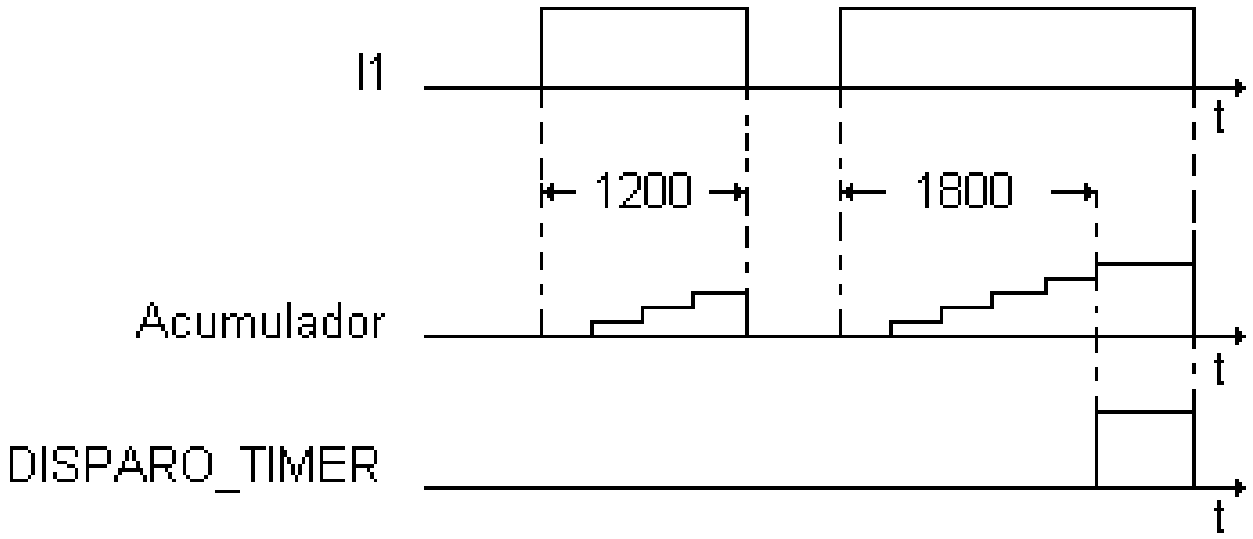
- Timer On Delay TON:



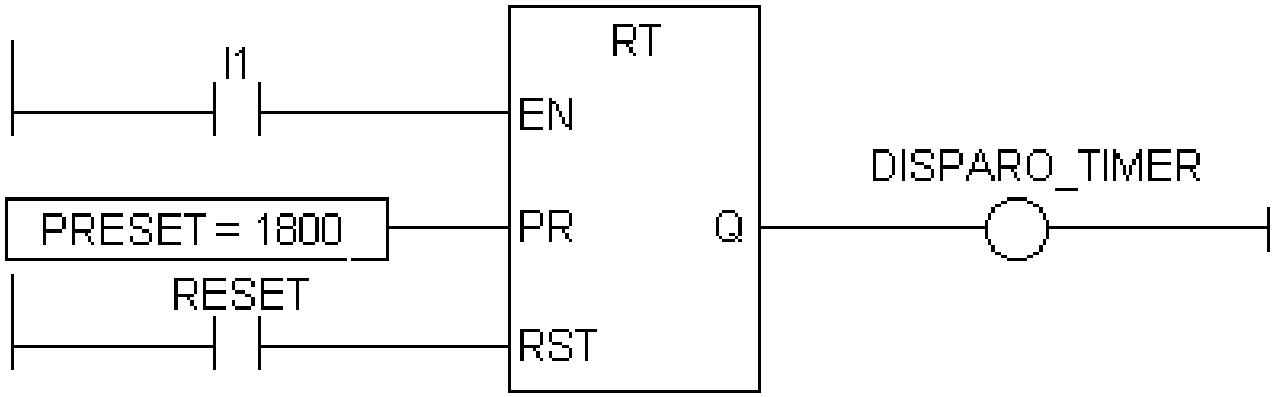
- Entrada IN (bit): “Habilitación”, se conecta al pulso que el timer retarda
- Entrada PT (time): “Preset”, determina valor del acumulador para el que se ejecuta acción del timer
- Salida Q (bit): “Done”, indica la expiración del tiempo de retardo
- Salida ET (time): “Elapsed Time”, la cuenta del tiempo transcurrido

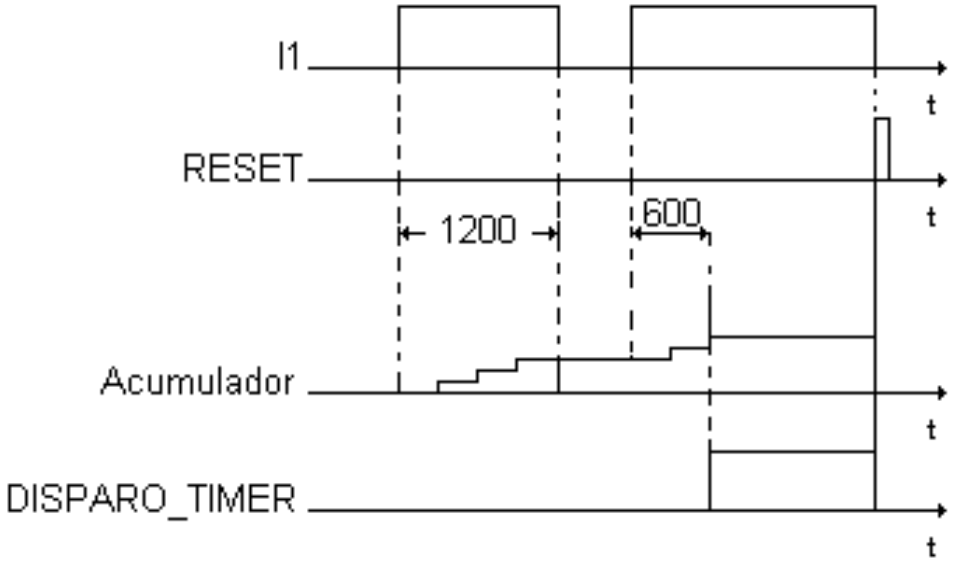
Timers: Ejemplo TON





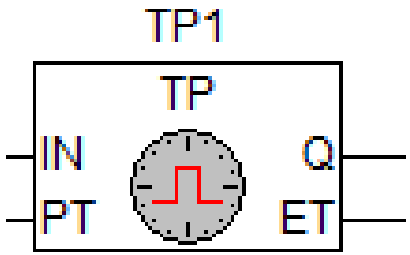
Timers: Ejemplo Retentivo





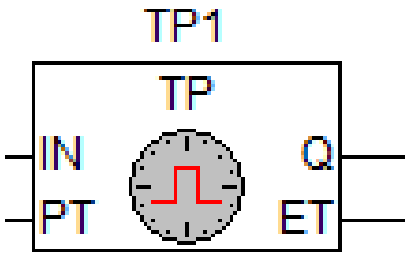
Timers: TP

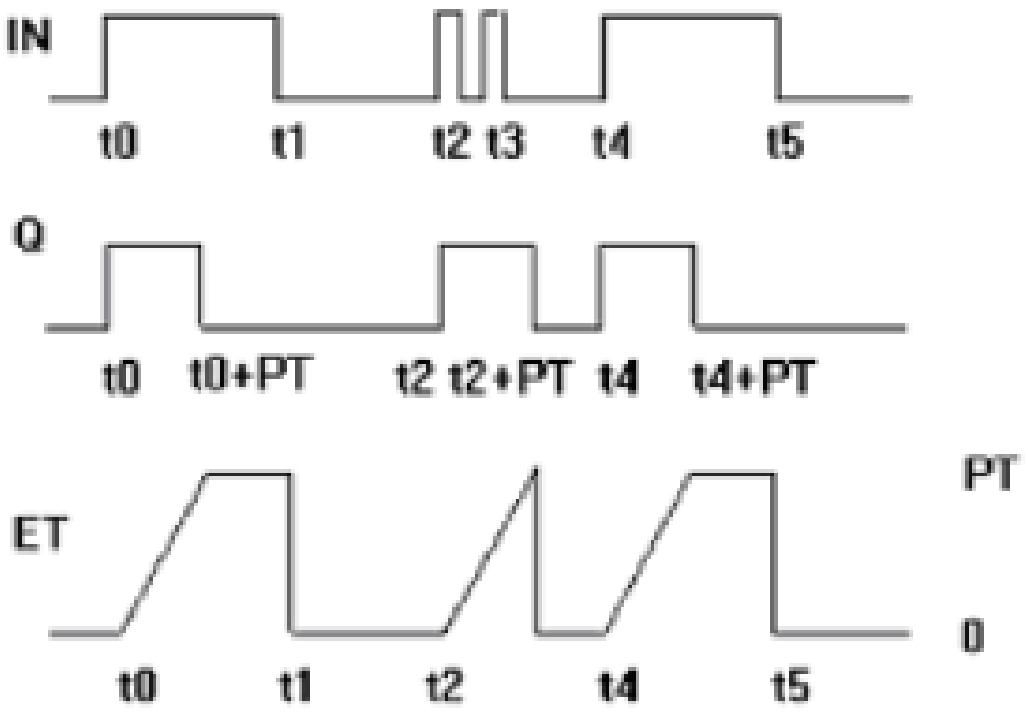
- Timer de Pulso TP:



- Entrada IN (bit): entrada que genera el pulso cuando pasa a 1
- Entrada PT (time): “Preset”, duración del pulso de salida
- Salida Q (bit): se mantiene en 1 durante el tiempo PT luego de 1 en la entrada
- Salida ET (time): “Elapsed Time”, la cuenta del tiempo transcurrido

Timers: Ejemplo TP





Timers

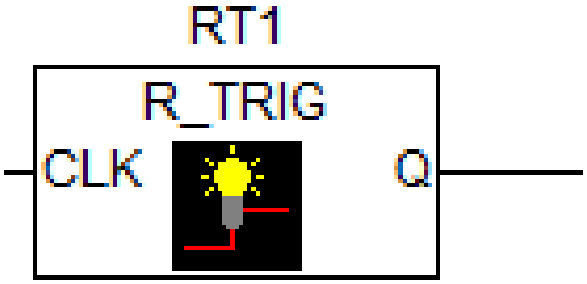
- La cuenta del timer es independiente de la ejecución del programa (a cargo del sistema operativo)
- La cuenta del contador SI depende de la ejecución del programa

Timers

- Precisión
  - Demora en la entrada: 1 tiempo de ciclo
  - Demora en ejecución del timer: 1 tiempo de ejecución (máx 1 ciclo)
  - Demora en la salida: 1 tiempo de ciclo
  - Total: 3 tiempos de ciclo
  - Ejemplo: tiempo de ciclo de 5 mseg lleva a error de 15 mseg
- Retardos en filtrado y en la electrónica

Triggers: R_TRIG

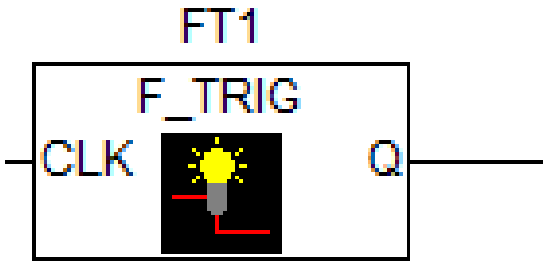
- Trigger Ascendente (Rising):



- Detección de flanco ascendente
- Entrada CLK (bit): activa la salida cuando pasa a 1 (flanco)
- Salida Q (bit): se mantiene en 1 durante 1 ciclo de ejecución ante flanco en CLK, luego pasa a 0

Triggers: F_TRIG

- Trigger Ascendente (Falling):



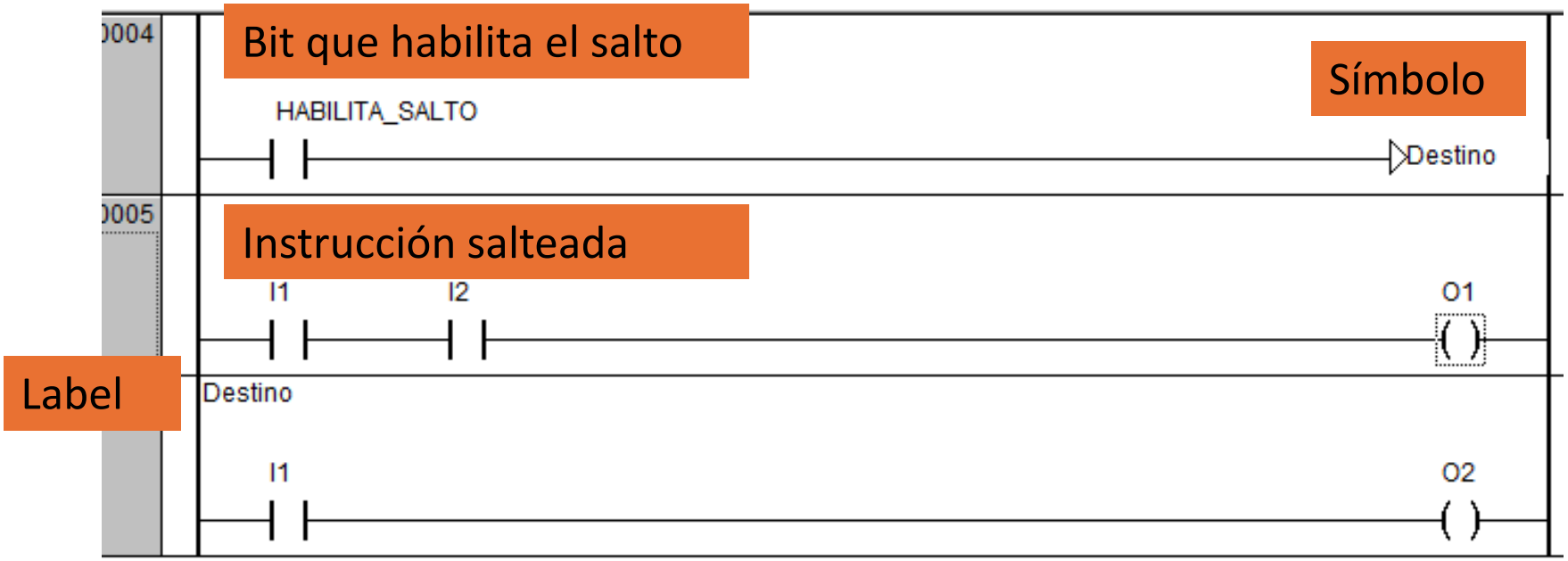
- Detección de flanco descendente
- Entrada CLK (bit): activa la salida cuando pasa a 0 (flanco)
- Salida Q (bit): se mantiene en 1 durante 1 ciclo de ejecución ante flanco en CLK, luego pasa a 0

Control de Flujo

- JSR (Jump to subroutine)
- En PLC de laboratorio, salto condicional a valor TRUE de un bit, definido por:
  - Jump symbol: símbolo asociado a instrucción JUMP
  - Bit que define el salto
  - Jump label: posición donde salta (siempre hacia adelante)

Control de flujo

- Esquema instrucción JUMP PLC lab:



Otras Instrucciones

- Existen muchas instrucciones en forma de bloques funcionales (varias se estudian en otras partes del curso)
- Desde el punto de vista del programa LADDER se clasifican en:
  - Instrucciones de entrada: evalúan si el escalón es verdadero o falso
  - Instrucciones de salida: se ejecutan según resultado del escalón

Otras Instrucciones

- Ejemplos de instrucciones de entrada:
  - Instrucciones de comparación: Igual (EQ), Mayor (GT), etc. Salida verdadera o falso en función de comparación de entradas (tipo Bool, Word, etc.)
- Ejemplos de instrucciones de salida:
  - Operaciones aritméticas o lógicas: ADD, AND, MUL, etc. Entradas y salida tipo Word, Real, etc.
  - Operaciones de movimiento de memoria: permiten copiar áreas de memoria
  - Funciones de control PID
  - Funciones de comunicaciones: permiten intercambio de mensajes entre PLCs

Ejemplo

Se desea escribir un programa que controle el encendido - apagado de una bomba.

La bomba será encendida si:

1) Se pulsa el botón de arranque.

2) La protección térmica está deshabilitada.

3) Está abierto el botón de alarma.

4) Está abierto el botón de parada.

Desde un tiempo T después del encendido, no puede haber ni sobre corriente ni baja corriente. Expresado de otra forma, desde un tiempo T después del arranque, la corriente I debe cumplir $I_{MIN} < I < I_{MAX}$, siendo I_{MIN} e I_{MAX} límites prefijados.

Ejemplo

El motor de la bomba se apagará si:

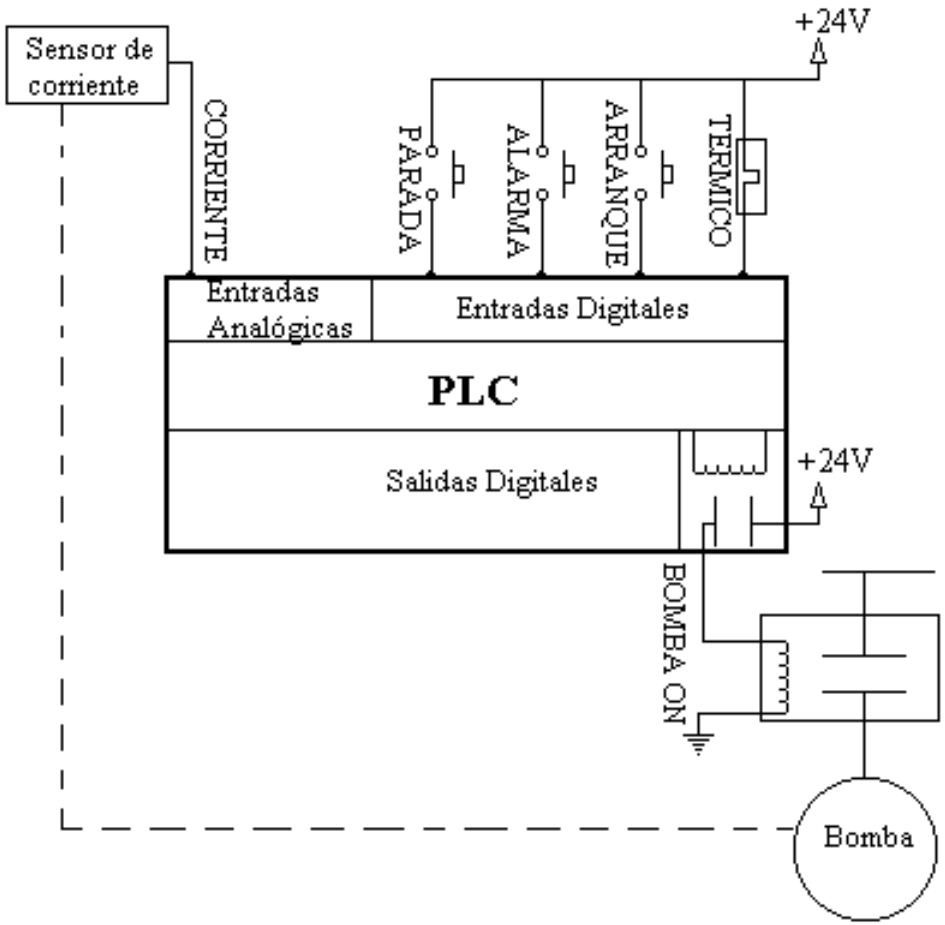
1) Se pulsa el botón de parada.

2) Se cierra la protección térmica.

3) Se pulsa el botón de alarma.

4) Los límites de corriente no son los correctos.

Conexiones al PLC



Ejemplo – Parte 1

Ejemplo – Parte 1

Ejemplo – Parte 1

La bomba será encendida si:

1) Se pulsa el botón de arranque.
2) La protección térmica está deshabilitada.
3) Está abierto el botón de alarma.
4) Está abierto el botón de parada.

Ejemplo – Parte 1

La bomba será encendida si:

1) Se pulsa el botón de arranque.
2) La protección térmica está deshabilitada.
3) Está abierto el botón de alarma.
4) Está abierto el botón de parada.

Ejemplo – Parte 1

La bomba será encendida si:

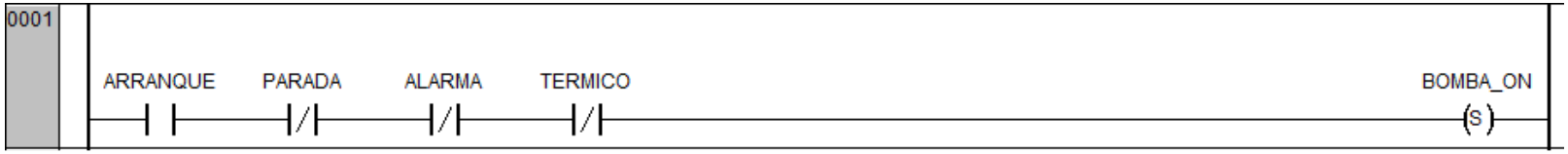
1) Se pulsa el botón de arranque.
2) La protección térmica está deshabilitada.
3) Está abierto el botón de alarma.
4) Está abierto el botón de parada.

Ejemplo – Parte 1

La bomba será encendida si:

1) Se pulsa el botón de arranque.
2) La protección térmica está deshabilitada.
3) Está abierto el botón de alarma.
4) Está abierto el botón de parada.

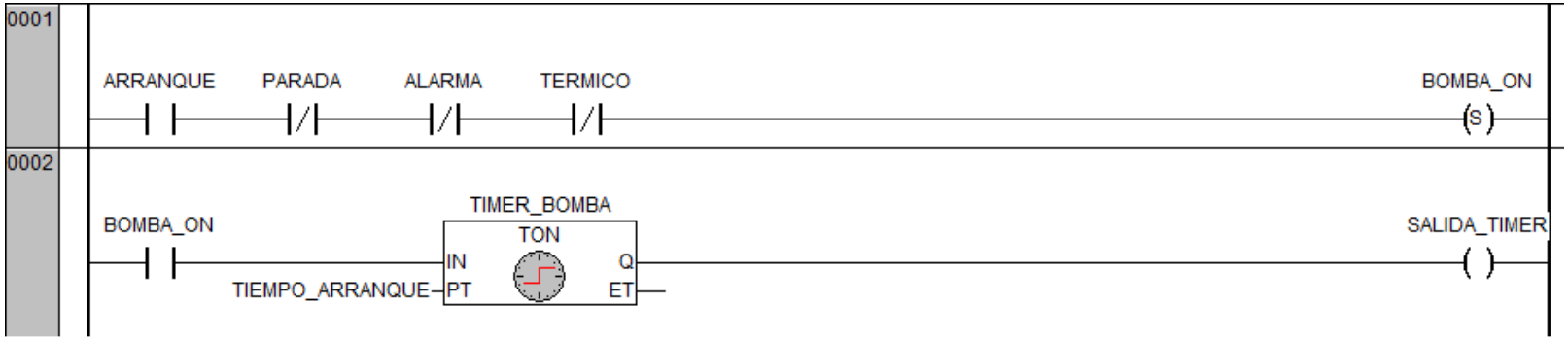
Ejemplo – Parte 1



La bomba será encendida si:

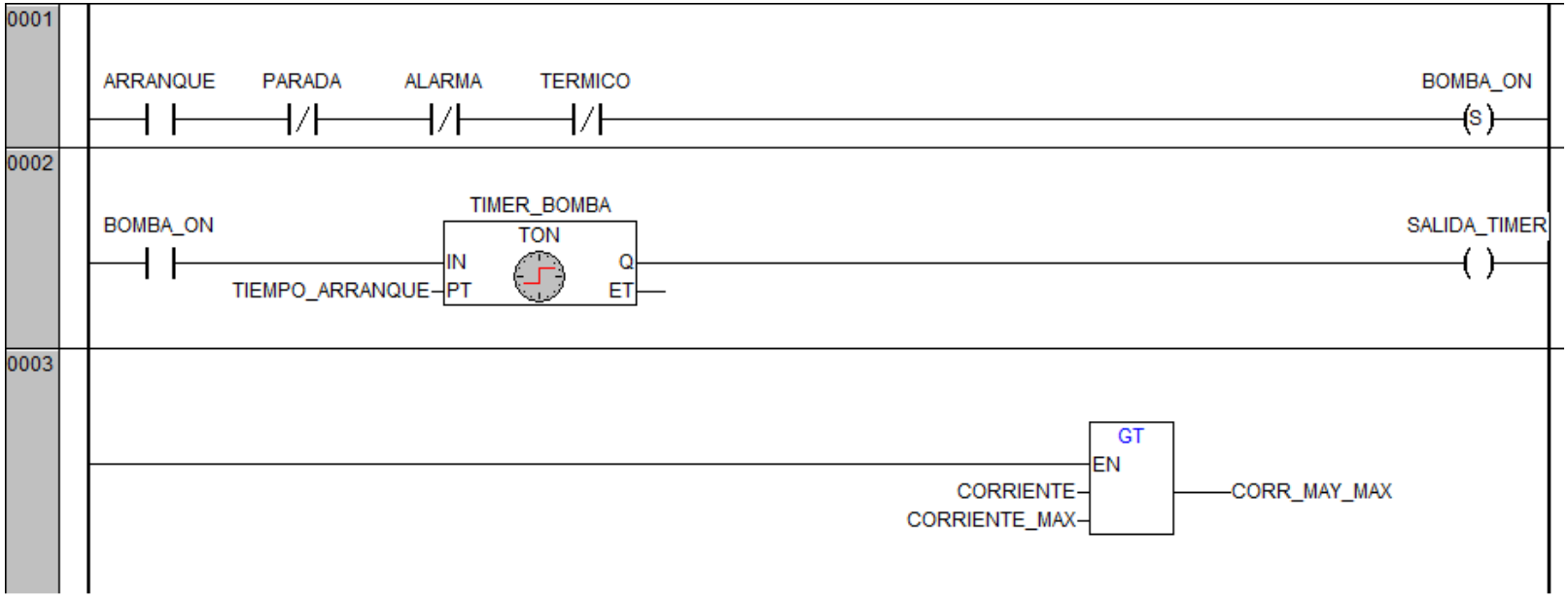
1) Se pulsa el botón de arranque.
2) La protección térmica está deshabilitada.
3) Está abierto el botón de alarma.
4) Está abierto el botón de parada.

Ejemplo – Parte 1



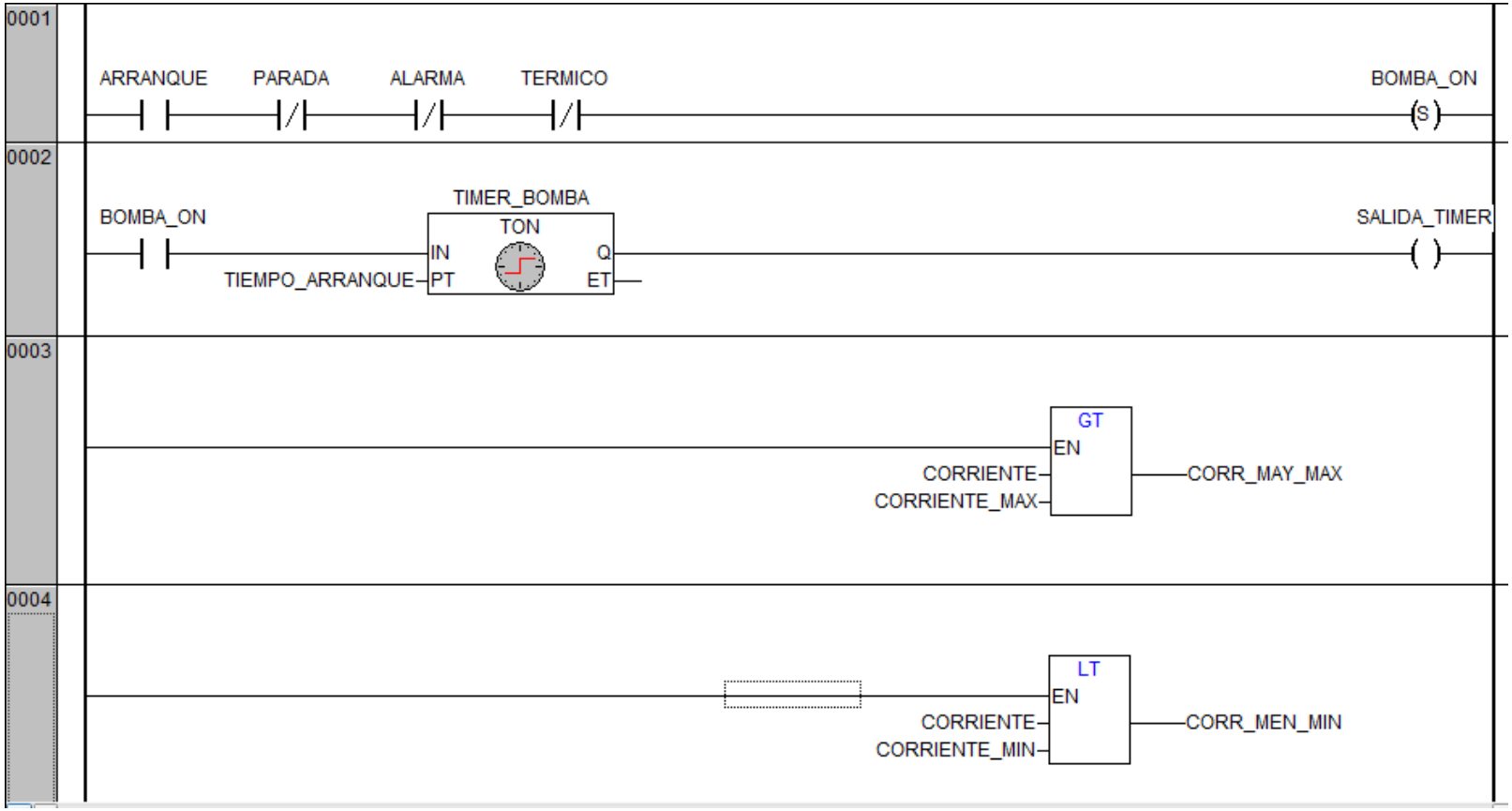
Desde un tiempo T después del encendido, no puede haber ni sobre corriente ni baja corriente. Expresado de otra forma, desde un tiempo T después del arranque, la corriente I debe cumplir $I_{MIN} < I < I_{MAX}$, siendo I_{MIN} e I_{MAX} límites prefijados.

Ejemplo – Parte 1

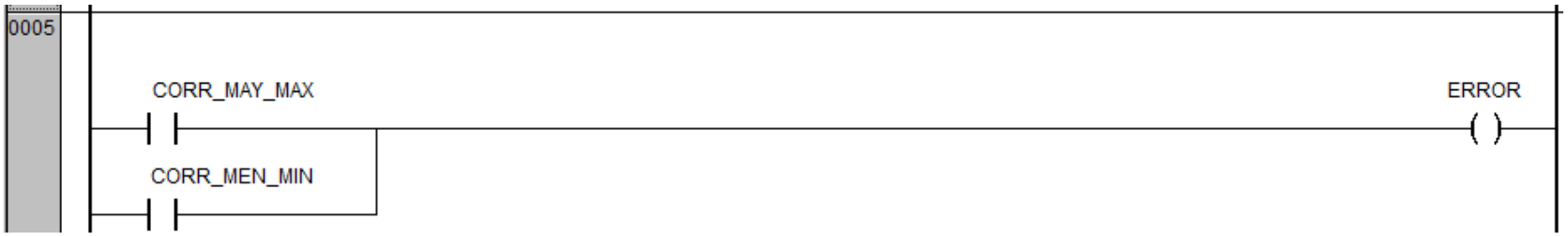


Desde un tiempo T después del encendido, no puede haber ni sobre corriente ni baja corriente. Expresado de otra forma, desde un tiempo T después del arranque, la corriente I debe cumplir $I_{MIN} < I < I_{MAX}$, siendo I_{MIN} e I_{MAX} límites prefijados.

Ejemplo – Parte 1

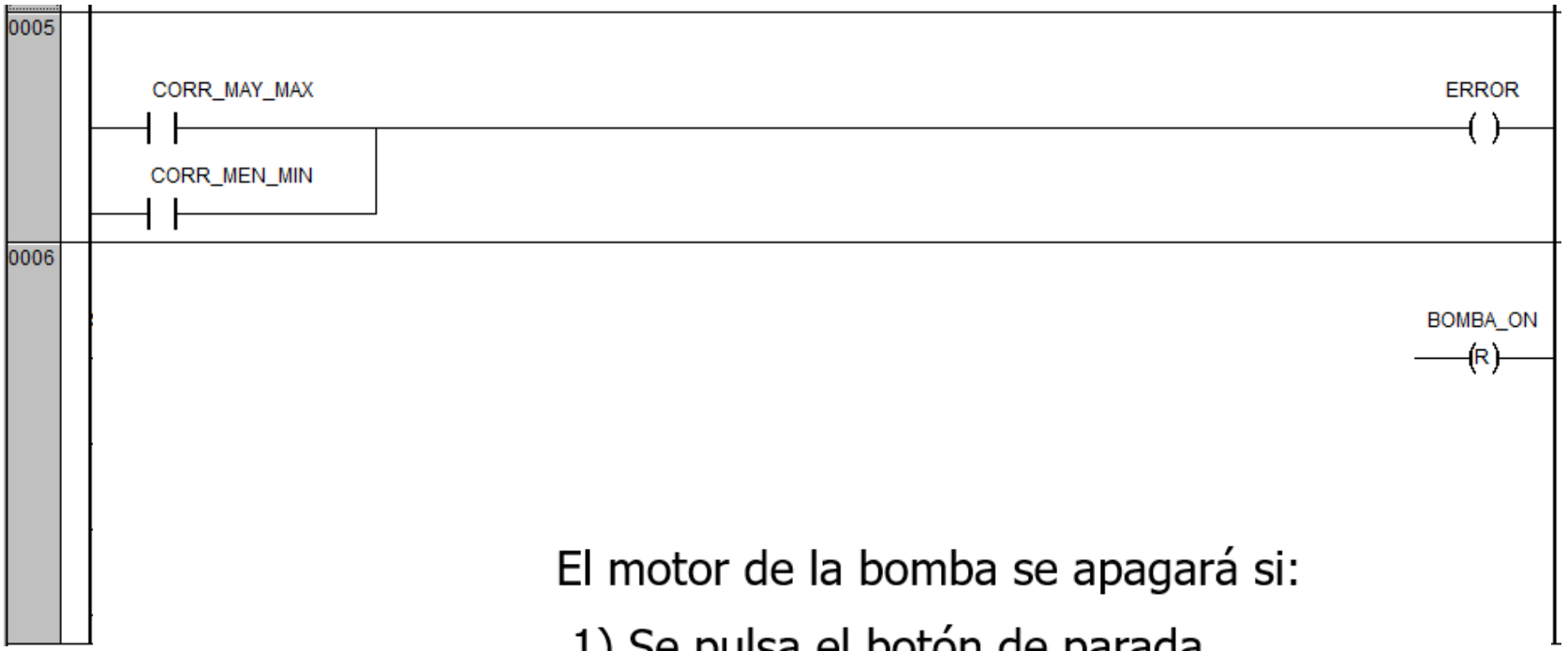


Ejemplo – Parte 2



Desde un tiempo T después del encendido, no puede haber ni sobre corriente ni baja corriente. Expresado de otra forma, desde un tiempo T después del arranque, la corriente I debe cumplir $I_{MIN} < I < I_{MAX}$, siendo I_{MIN} e I_{MAX} límites prefijados.

Ejemplo – Parte 2



El motor de la bomba se apagará si:

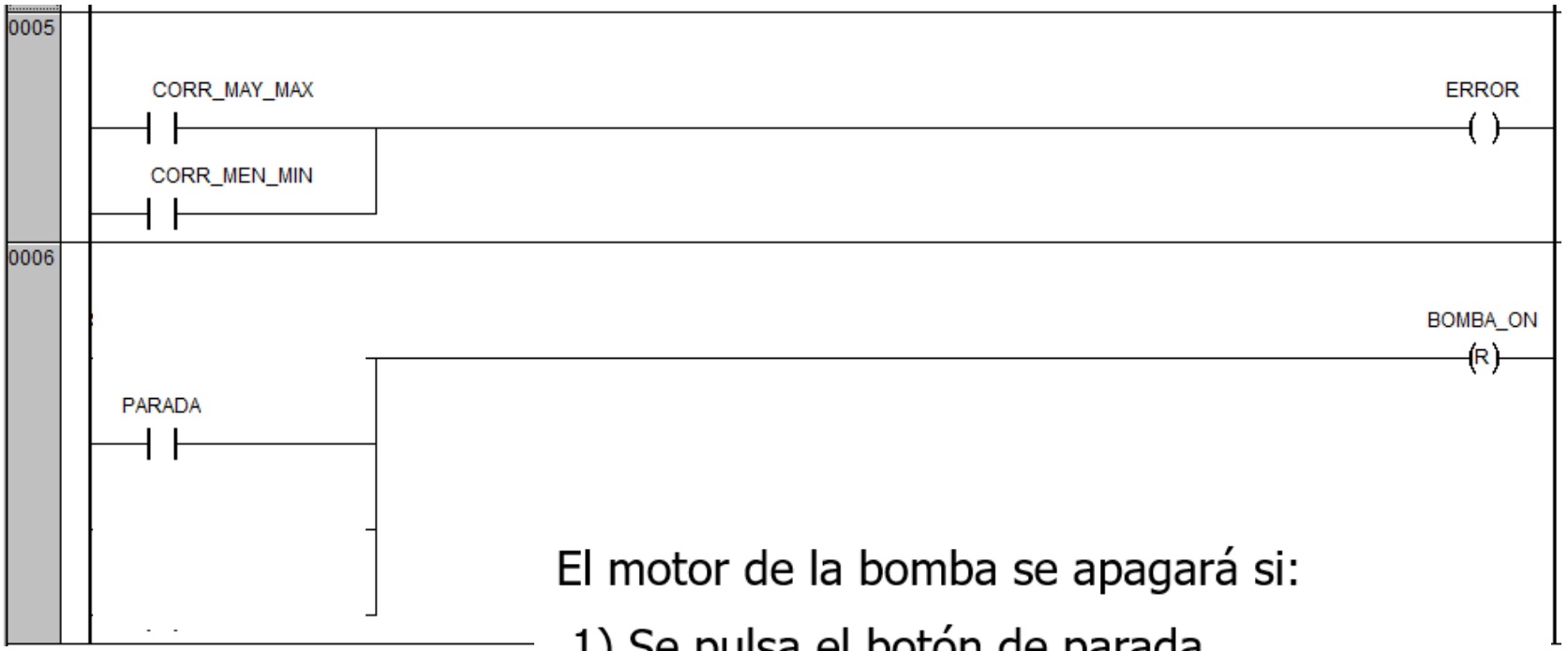
1) Se pulsa el botón de parada.

2) Se cierra la protección térmica.

3) Se pulsa el botón de alarma.

4) Los límites de corriente no son los correctos.

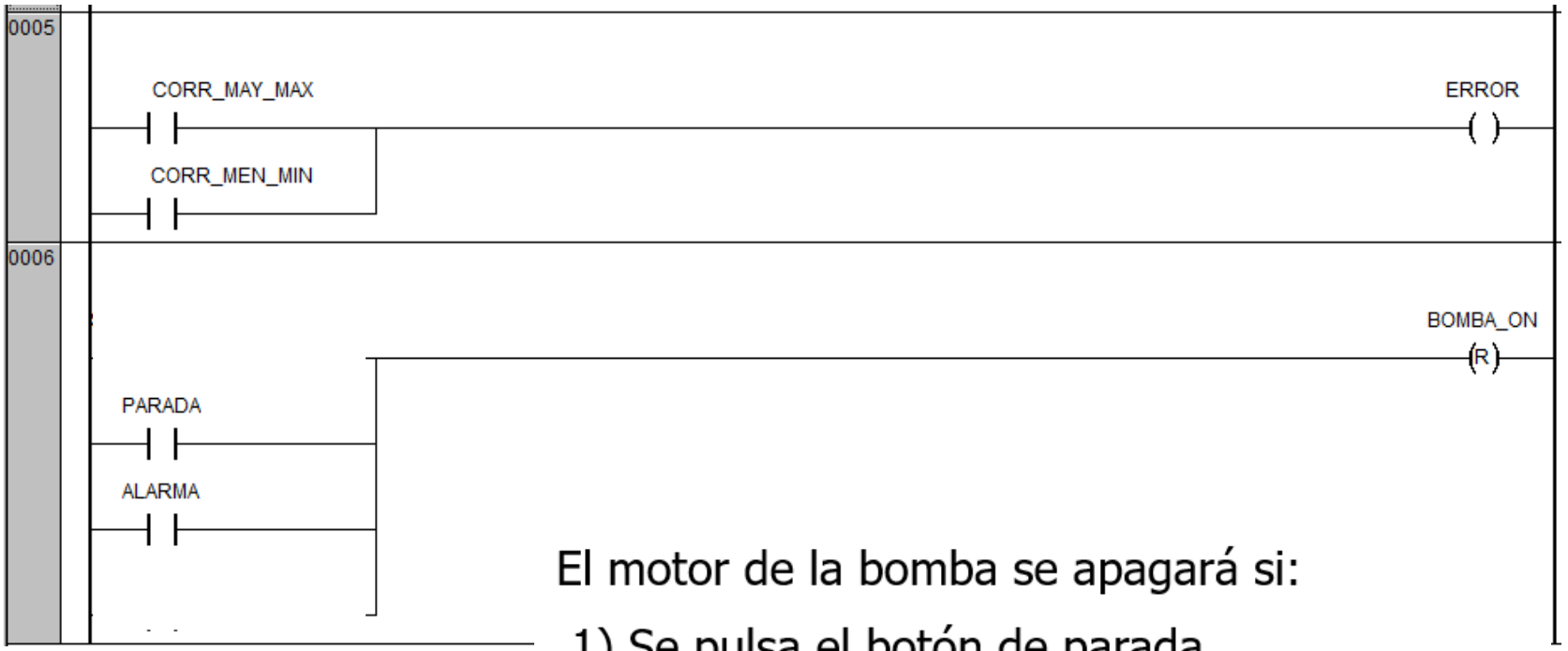
Ejemplo – Parte 2



El motor de la bomba se apagará si:

1) Se pulsa el botón de parada.
2) Se cierra la protección térmica.
3) Se pulsa el botón de alarma.
4) Los límites de corriente no son los correctos.

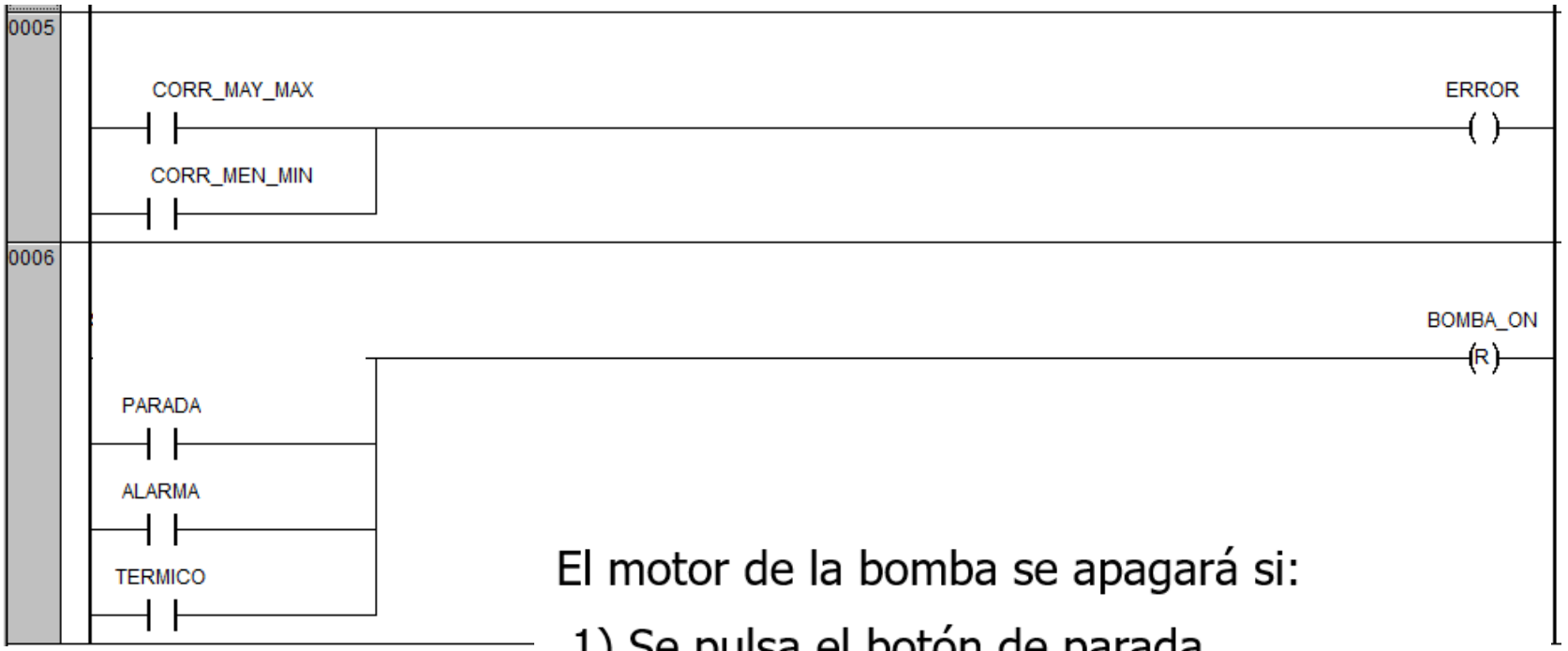
Ejemplo – Parte 2



El motor de la bomba se apagará si:

1) Se pulsa el botón de parada.
2) Se cierra la protección térmica.
3) Se pulsa el botón de alarma.
4) Los límites de corriente no son los correctos.

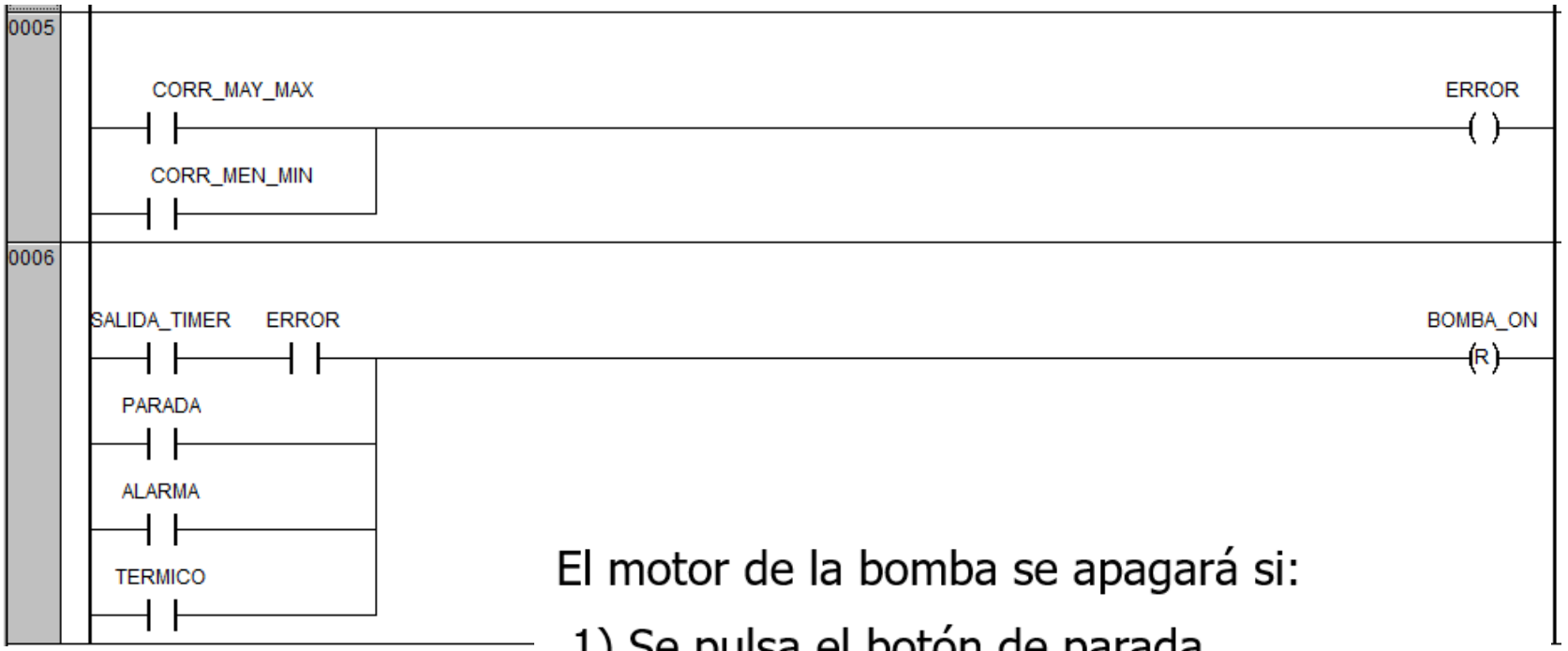
Ejemplo – Parte 2



El motor de la bomba se apagará si:

1) Se pulsa el botón de parada.
2) Se cierra la protección térmica.
3) Se pulsa el botón de alarma.
4) Los límites de corriente no son los correctos.

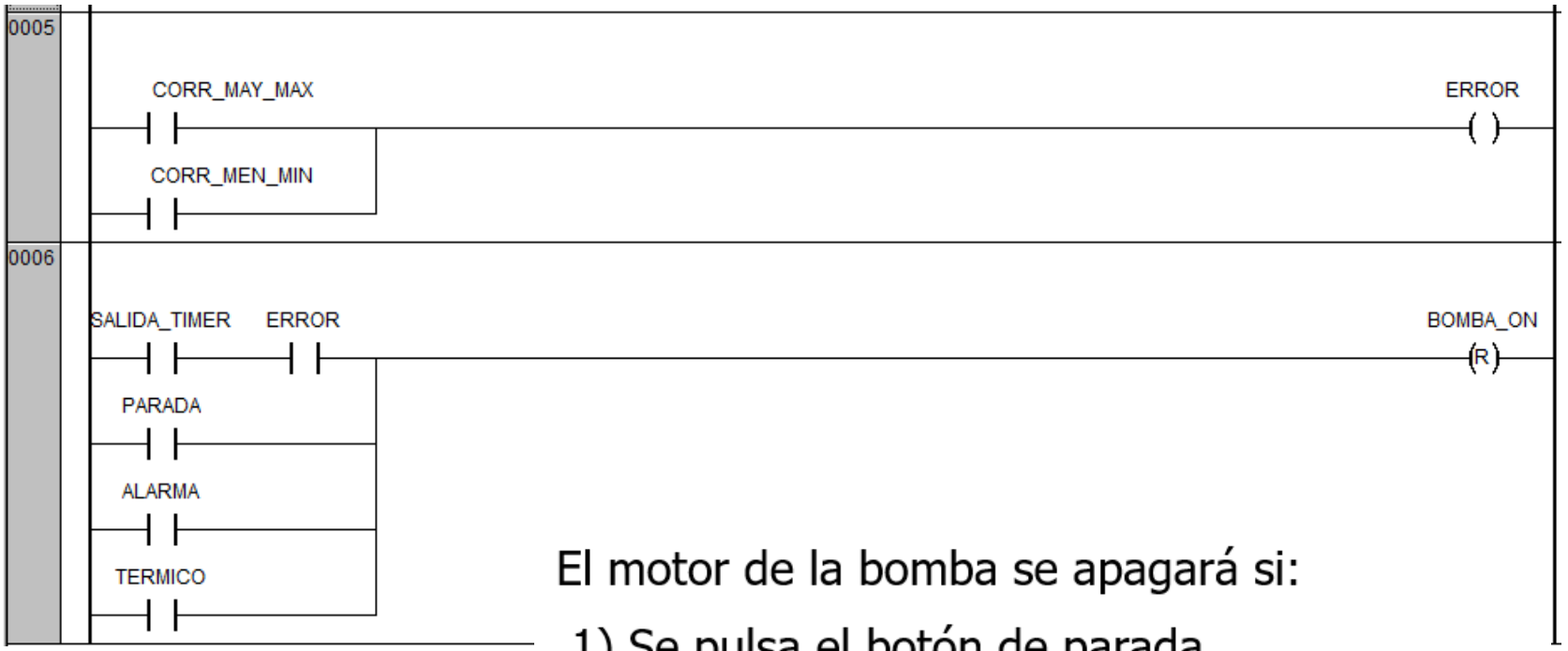
Ejemplo – Parte 2



El motor de la bomba se apagará si:

1) Se pulsa el botón de parada.
2) Se cierra la protección térmica.
3) Se pulsa el botón de alarma.
4) Los límites de corriente no son los correctos.

Ejemplo – Parte 2



El motor de la bomba se apagará si:

1) Se pulsa el botón de parada.
2) Se cierra la protección térmica.
3) Se pulsa el botón de alarma.
4) Los límites de corriente no son los correctos.

Diagrama de Estados

- STD – State Transitions Diagram
- No es un lenguaje
- Es una metodología para representar cierta lógica de funcionamiento en base a estados y transiciones

Ejemplo

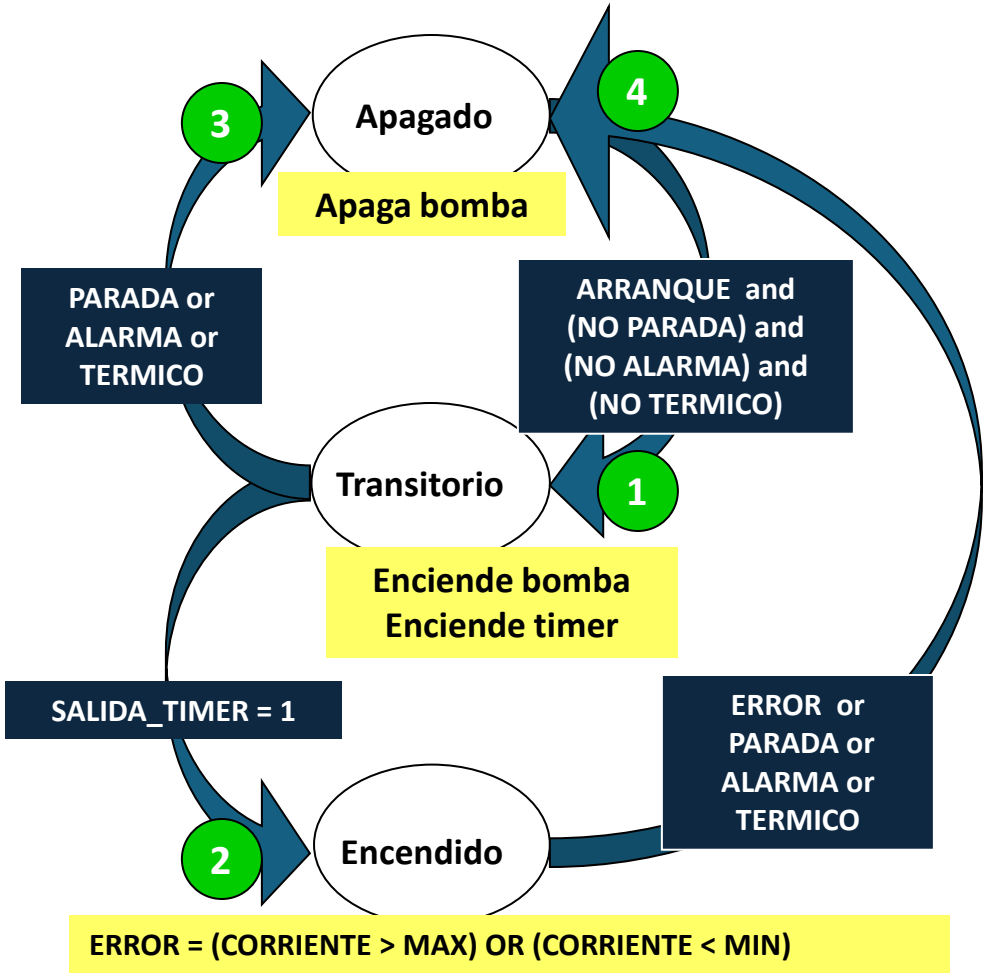
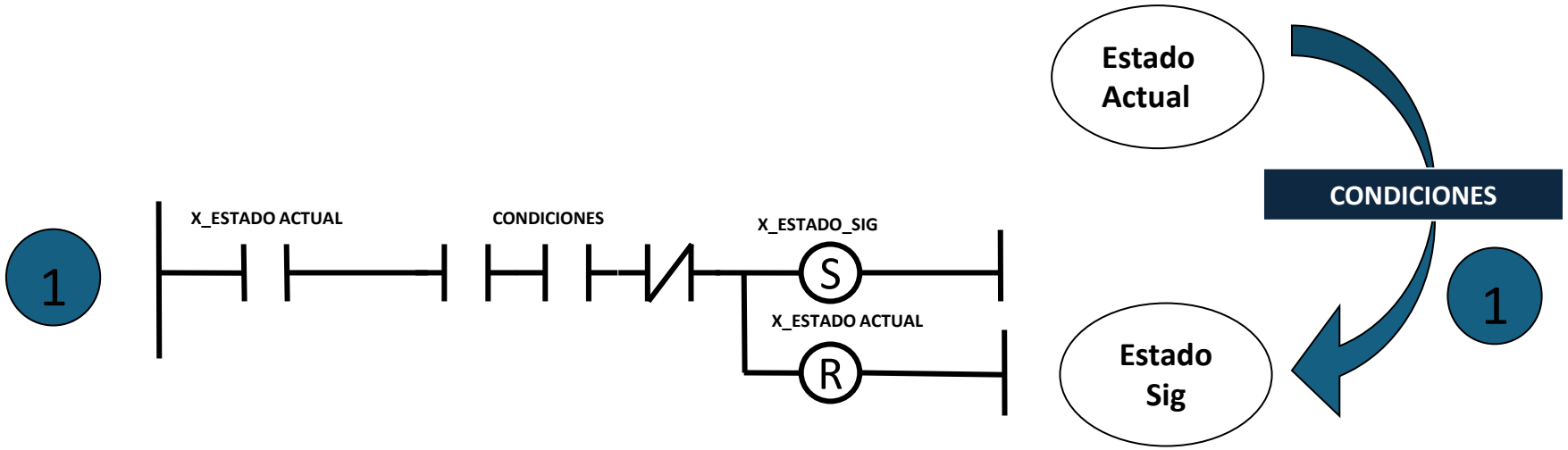
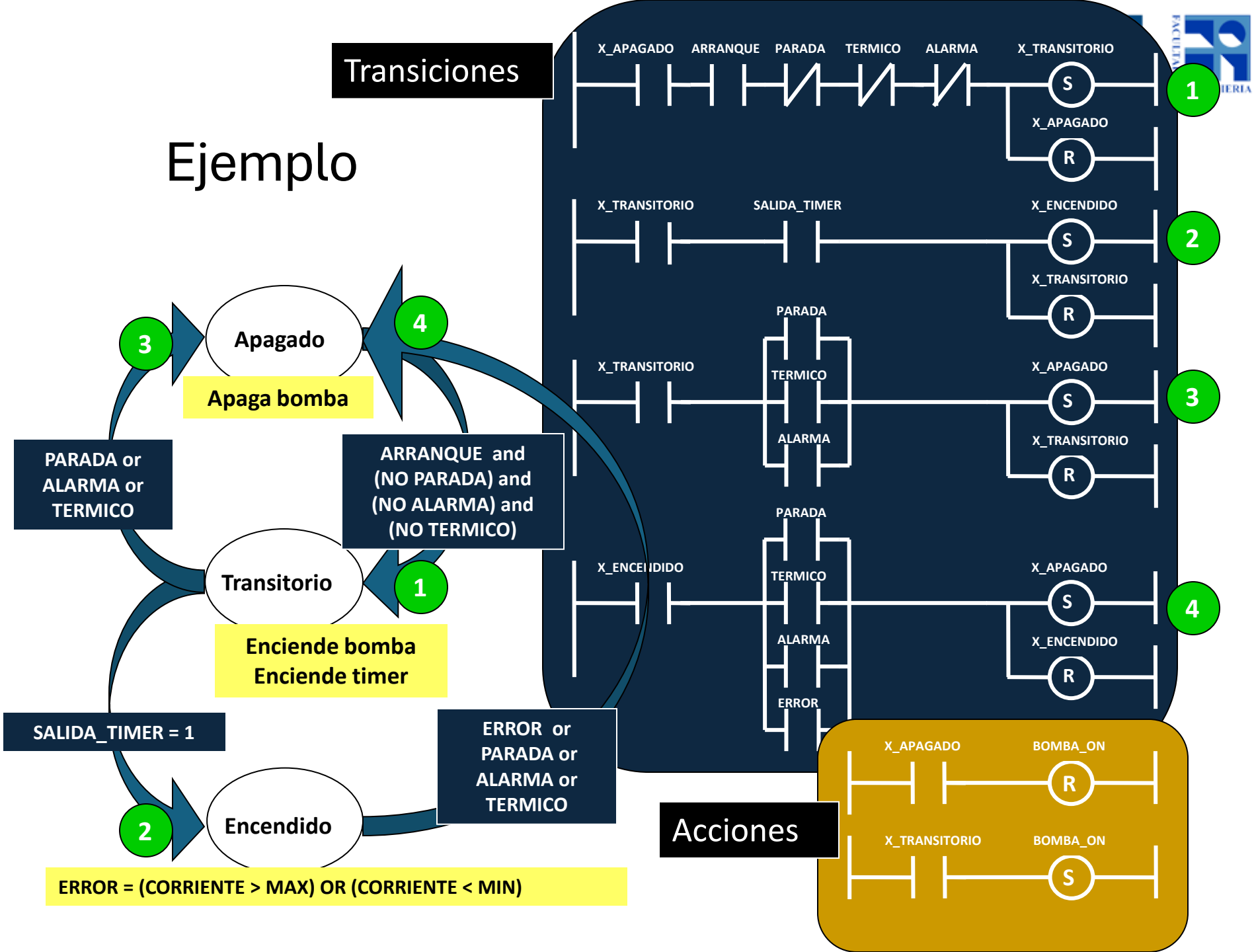

Apagado
Apaga bomba
Transitorio
Enciende bomba
Enciende timer
Encendido
1
2
3
4
PARADA or
ALARMA or
TERMICO
ARRANQUE and
(NO PARADA) and
(NO ALARMA) and
(NO TERMICO)
SALIDA_TIMER = 1
ERROR or
PARADA or
ALARMA or
TERMICO
ERROR = (CORRIENTE > MAX) OR (CORRIENTE < MIN)

Diagrama de Estados en LD

- Método para convertir un diagrama de estados al lenguaje LD



Ejemplo


Transiciones
X_APAGADO
ARRANQUE
PARADA
TERMICO
ALARMA
X_TRANSITORIO
S
1
X_APAGADO
R
X_TRANSITORIO
SALIDA_TIMER
X_ENCENDIDO
S
2
X_TRANSITORIO
R
PARADA
X_TRANSITORIO
TERMICO
X_APAGADO
S
3
ALARMA
X_TRANSITORIO
R
PARADA
X_ENCENDIDO
TERMICO
X_APAGADO
S
4
ALARMA
X_ENCENDIDO
R
ERROR
Apagado
Apaga bomba
Transitorio
Enciende bomba
Enciende timer
Encendido
PARADA or ALARMA or TERMICO
ARRANQUE and (NO PARADA) and (NO ALARMA) and (NO TERMICO)
SALIDA_TIMER = 1
ERROR or PARADA or ALARMA or TERMICO
ERROR = (CORRIENTE > MAX) OR (CORRIENTE < MIN)
Acciones
X_APAGADO
BOMBA_ON
R
X_TRANSITORIO
BOMBA_ON
S

Ventajas / Desventajas


- Método sistemático


- Minimiza/elimina errores posteriores

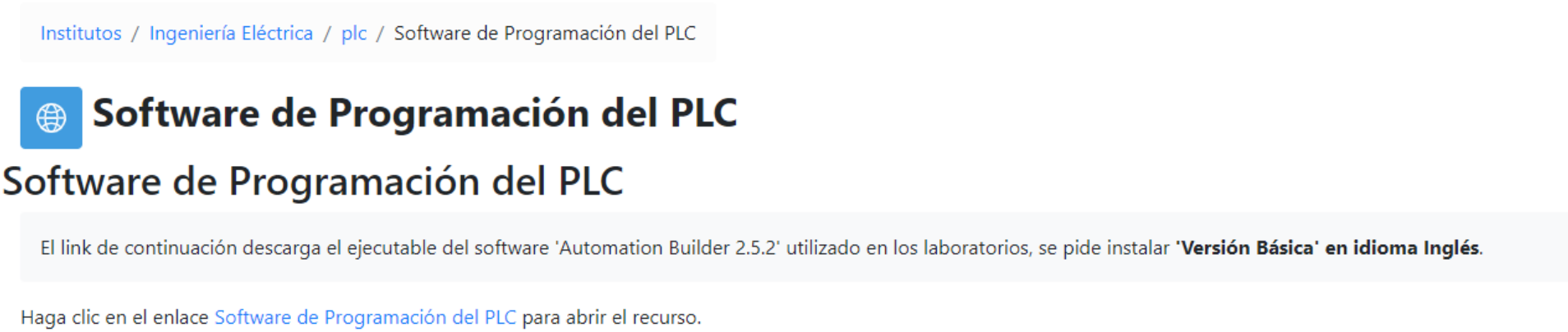

- LD disponible en todos los PLCs


- Poco intuitivo, difícil de comprender sin el diagrama de estados asociado


- Cambios requieren estudio del diagrama previamente

Ambiente de Desarrollo (desde EVA)

- Automation Builder
- Instalar versión **2.5** **Básica** en **Inglés**
- En EVA: plc: Software de Programación del PLC | FING



Ambiente de Desarrollo

- Automation Builder
- Instalar versión **2.5** **Básica** en **Inglés**
- http://new.abb.com/plc/automationbuilder/platform/software