Edizioni G-Tronic
9RSAA

Regla de Gottardo

Aplicado en TIA PORTAL V21
Vol. 1
Primera edición 2026

Esta publicación está dirigida a la comunidad científica; por lo tanto, no es un libro de texto para aprender a programar PLC Siemens con TIA PORTAL.
El objetivo de esta edición monográfica es definir un axioma, desarrollar sus implicaciones teóricas y demostrar los teoremas resultantes.

Escrito, editado y publicado
Por

ing. Prof. Dott. Marco Gottardo PhD

Serie de publicaciones sobre automatización industrial.

Regla de Gottardo

Esta edición fue editada y publicada en marzo de 2026 por:
Ing. Prof. Dr. Marco Gottardo Ph.D.

Marco Gottardo, C.F. GTTMRC68R06G224I,
Via Colombo 14, 30030 Vigonovo (VE) Italia.
E-mail: ad.noctis@gmail.com

ISBN-13: 9798185081174
Editorial: G-Tronic di Marco Gottardo editore

Índice:

Prefacio

Esta publicación no se concibe como una obra o libro de texto popular, sino como un tratado académico o un volumen monográfico, destinado a introducir y debatir una propuesta teórica específica destinada a la evaluación de la comunidad científica.
El objetivo principal del trabajo es presentar y formalizar el siguiente axioma: "Cuando un contacto es delegado para liberar un auto-bloqueo, se captura en el software en un estado opuesto al representado en el funcional."
El propósito de la discusión es analizar esta proposición, verificar su coherencia lógica y proponerla como un axioma dentro del marco teórico expuesto por el autor. La formalización del axioma constituye el primer paso hacia una posible estructuración más amplia del modelo conceptual, dentro de la cual la proposición pueda desarrollarse y demostrarse posteriormente en forma de teorema.
El presente trabajo también pretende establecer una formulación clara y formalmente reconocible de la tesis propuesta[1], con la intención de atribuir su autoría de manera explícita y rastreable en el contexto de la literatura científica.
Aunque adopta un lenguaje deliberadamente simple y directo, el contenido mantiene el nivel de precisión terminológica y rigor argumentativo requerido en los campos científicos y académicos, para garantizar la claridad de la exposición sin comprometer la formalidad del discurso teórico.

[1] **Tesis:** Formulación para la definición de los estados lógicos de los contactos de liberación de los coches retenidos basada en el estado indicado en los diagramas funcionales. Cuando un contacto físico en el campo es delegado para liberar un autoenganching, este se adquiere en el software en lugar de cómo se representa en el funcional. En consecuencia, se necesitan reglas claras y bien definidas para la representación funcional.

Enmarcamiento formal del problema

En el control industrial discreto, la función de auto-holding o sellado realiza una memoria biestable controlada por dos entradas:

- S: Comando Start (SET)
- R: Commando de arresto (RESET)
- Q: Estado de salida (Bobina)

En el formalismo PLC (Escalera del FBD), La estructura clásica es:

- NO contacto, normalmente abierto en el campo, de Inicio
- Contacto NC, normalmente cerrado en el campo, de Stop
- Contacto NO con enganche automático en paralelo con Inicio
- Bobina de salida

Un primer axioma, necesario para la aplicación de la regla del Gotardo (terminología docente muy extendida en el campo técnico italiano), establece:
En presencia simultánea de los comandos de excitación y desenergización, debe prevalecer la desenergización.
En términos lógicos:

$$\boldsymbol{S = 1 \ \wedge \ R = 1 \Rightarrow Q = 0}$$

Esta es una regla de prioridad de RESET, crucial por razones de seguridad funcional.

Axiomatización en álgebra booleana
Consideremos el álgebra booleana axiomática:

$$\mathcal{B} = (B, +, \cdot, ', 0, 1)$$

con los siguientes axiomas fundamentales:

1. Conmutatividad

$$A + B = B + A; A \cdot B = B \cdot A$$

2. Asociatividad

$$(A + B) + C = A + (B + C)$$

3. Distributividad

$$A \cdot (B + C) = A \cdot B + A \cdot C$$

4. Complementación

$$A + A' = 1; A \cdot A' = 0$$

5. Identità

$$A + 0 = A; A \cdot 1 = A$$

Formalización de la autocontención
Ecuación recursiva de estado

Estado de la salida en ese momento $k + 1$**è:**

$$Q_{k+1} = (S + Q_k) \cdot R'$$

Esta es la formalización canónica de la autocontención con prioridad para reiniciar.
Interpretación:

- $S + Q_k$→ activación inicial o mantenimiento
- R'→ Condición de no RESET

Primera demostración del dominio del Gottardo

Queremos demostrar formalmente:

$$\boldsymbol{S = 1 \land R = 1 \Rightarrow Q_{k+1} = 0}$$

Sustituimos en la ecuación:

$$\boldsymbol{Q_{k+1} = (1 + Q_k) \cdot 1'}$$

Ya que:

$$\boldsymbol{1 + Q_k = 1}$$

e

$$\boldsymbol{1' = 0}$$

más:

$$\boldsymbol{Q_{k+1} = 1 \cdot 0 = 0}$$

Probado.

Propiedad estructural: Prioridad de RESET

Ahora mostremos que la función es equivalente a una forma explícita de prioridad:

$$\boldsymbol{Q_{k+1} = S \cdot R' + Q_k \cdot R'}$$

Aplicación de la distributividad:

$$\boldsymbol{(S + Q_k) \cdot R' = S \cdot R' + Q_k \cdot R'}$$

Coleccionismo:

$$\boldsymbol{Q_{k+1} = R' \cdot (S + Q_k)}$$

Observación:
Se $\boldsymbol{R = 1 \Rightarrow R' = 0}$, allora:

$$\boldsymbol{Q_{k+1} = 0}$$

independientemente de y $\boldsymbol{SQ_k}$.
Esta es la formalización matemática de la regla de Gottardo.

Comparación con el pestillo clásico SR

Pestillos SR sin prioridad:

$$\boldsymbol{Q_{k+1} = S + Q_k \cdot R'}$$

Se $\boldsymbol{S = R = 1}$:

$$\boldsymbol{Q_{k+1} = 1 + Q_k \cdot 0 = 1}$$

Aquí, prevalece el EETS → no cumple con la norma del Gottardo.

Forma canónica de Shannon

Aplicamos la expansión respecto a R:

$$\boldsymbol{Q_{k+1} = R \cdot f_{R=1} + R' \cdot f_{R=0}}$$

Calculamos:

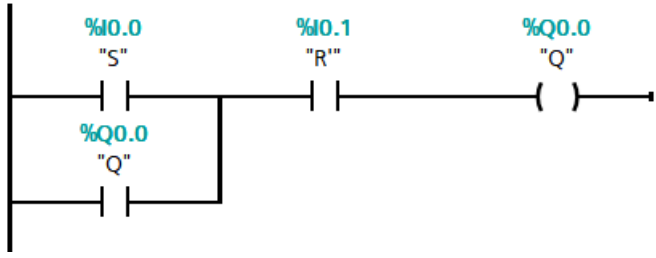
- **Se $\boldsymbol{R = 1 \Rightarrow Q_{k+1} = 0}$**
- **Se $\boldsymbol{R = 0 \Rightarrow Q_{k+1} = S + Q_k}$**

Así que:

$$\boldsymbol{Q_{k+1} = R \cdot 0 + R' \cdot (S + Q_k)}$$
$$\boldsymbol{Q_{k+1} = R' \cdot (S + Q_k)}$$

Forma mínima

Interpretación en el PLC de Siemens (Portal TIA)
En la escalera del Portal TIA:



La estructura implementa exactamente:

$$\boldsymbol{Q = (S + Q) \cdot R'}$$

En el bucle de escaneo:

1. Lectura de entrada
2. Evaluación lógica
3. Actualización de imagen de salida

La prioridad del RESET está garantizada estructuralmente, no temporalmente.

Demostración de la estabilidad del sistema

Busquemos estados estables:

$$\boldsymbol{Q_{k+1} = Q_k = Q}$$
$$\boldsymbol{Q = (S + Q) \cdot R'}$$

Caso 1: $R = 1$

$$\boldsymbol{Q = 0}$$

Equilibrio único.
Caso 2: $R = 0$

$$\boldsymbol{Q = S + Q}$$

Se $\boldsymbol{S = 0 \Rightarrow Q = Q}$(memoria)
Se $\boldsymbol{S = 1 \Rightarrow Q = 1}$
Sistema coerente.

Teorema de seguridad (Formulación)
Teorema:
La función booleana

$$\boldsymbol{Q_{k+1} = (S + Q_k) \cdot R'}$$

es la única función mínima de tres variables que satisface:

1. Autorretención
2. Reset dominante
3. Ausencia de estados prohibidos
4. Idempotencia respecto a $R = 1$

Demostración (sintética):
Imponer las restricciones en la tabla de verdad y minimizarlas mediante aplicaciones de Karnaugh produce un mínimo único que implica una cobertura equivalente a la forma anterior.

Extensión: forma matricial en el espacio de estados
Podemos escribir:

$$\boldsymbol{Q_{k+1} = f(Q_k, S, R)}$$

Sistema dinámico discreto no lineal sobre campo booleano:

$$\boldsymbol{Q_{k+1} = R'S + R'Q_k}$$

Estructuralmente equivalente a:

$$\boldsymbol{Q_{k+1} = R'(S + Q_k)}$$

que representa un sistema lógico de primer orden con entrada de cancelación dominante.

Síntesis de aplicaciones didácticas

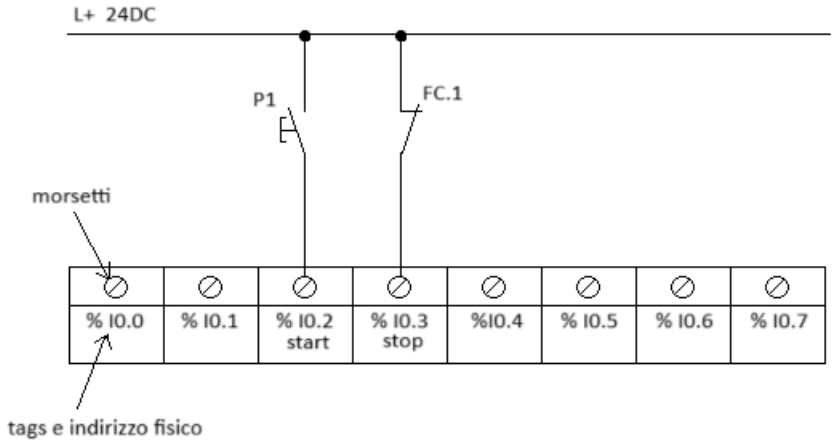
La regla del Gotardo no es un artificio gráfico de escalera, pero:

1. una propiedad estructural de la función booleana
2. una restricción de prioridad en el dominio discreto
3. una condición de seguridad funcional
4. Una elección de diseño coherente con las normativas de seguridad de las máquinas

Su formalización axiomática permite:

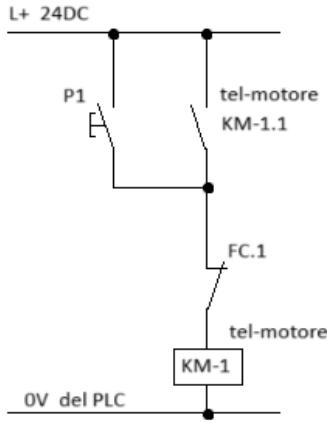
- Análisis formal
- Prueba de equidad
- Verificación simbólica
- Síntesis automática

Aplicación de campo partiendo de bloques terminales de entrada.
El trabajo del programador comienza con una fase de entrevista, en la que se recopilan, documentan y formalizan las especificaciones técnicas del sistema a construir mediante las firmas del cliente.
Estos no deberían cambiar durante el transcurso de la obra, que, sin embargo, ocurre puntualmente, perturbando el trabajo ya en curso.
Cuando empieces a programar el sistema o la máquina, debes proporcionar al programador dos documentos, el diagrama de cableado y el P&ID.
La primera muestra las conexiones entre el equipo sensorial y las entradas del PLC, así como entre las salidas y los actuadores o sus dispositivos de control (relés, contactores, válvulas solenoides, etc.).
La imagen muestra un extracto de un diagrama de cableado típico[2] proporcionado al programador en la fase inicial del desarrollo de la obra.



El botón P1 lleva la señal de control de viaje a la entrada %I0,2, es decir, el tercer bit del primer bloque de terminales de entrada, mientras que el interruptor electromecánico de límite está conectado a la siguiente entrada digital %I0,3.
Probablemente el programador realiza un "comando de auto-retención", que mantiene el movimiento hasta el límite asignado.
Lo que se muestra en este dibujo no debería aparecer en el diagrama de cableado:

[2] En el diagrama de cableado, las entradas y salidas se muestran como rectángulos, organizados en filas de ocho o un byte. Tienen un campo de dirección y un campo para las etiquetas (etiqueta simbólica de la variable). A simple vista, si el rectángulo está en la parte inferior de la página, son entradas; si está arriba, son salidas.
Las variables internas como Merker y las definidas en las bases de datos no se muestran en los esquemas.

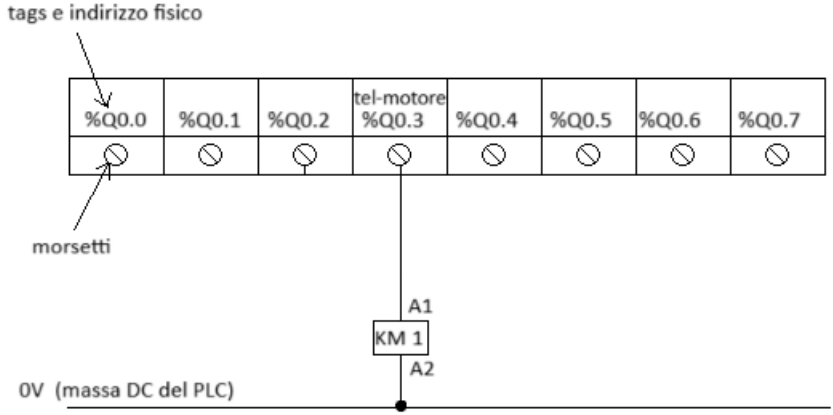


Esto no aparece en el esquema porque representa la lógica implementada internamente[3].
Es un flip flop electromecánico capaz de almacenar un bit hasta que ocurre el evento de desenganche.
En este caso, el[4] comando monostable de la unidad se almacena en el bit interno KM-1, preferiblemente declarado dentro de un Bloque de Datos, en lugar de como un Merker en la Tabla de Etiquetas.

Cuando se pulsa el botón P1, la señal positiva, a 24V DC, se transmite al nodo inferior.
En el nodo se encuentra con un dispositivo electromecánico cableado normalmente cerrado, y la señal pasa a través de él energizando la bobina.
La bobina energizada cierra su contacto situado paralelo al botón de control.
El operador puede soltar el botón, pero el carrete se mantiene con su propio contacto colocado en paralelo.
Alguna parte móvil del coche interceptará el FC.1 que, al cortar la línea, libera el coche retenido.
Como veremos en varias partes del texto, esta simbología, aunque presenta fuertes similitudes, no es un diagrama eléctrico y toma el nombre de diagrama funcional.
Muchas cosas que serían obligatorias para los diagramas de cableado son insignificantes en diagramas funcionales, por ejemplo, muchos de los numerados.
No tiene sentido numerar un cable que en realidad no existe ya que está

[3] En informática, implementar es sinónimo de realizar, y por tanto de programar.
[4] Las señales de entrada pueden ser de naturaleza monoestable o biestable. Las señales monoestables son impulsivas, como las generadas por botones, que generan el estado VERDADERO solo mientras el operador permanece con el dedo. Las señales bistables, en cambio, son típicas de los conmutadores. Cada acción corresponde a una transición de un estado estable a uno complementario, donde permanece hasta una acción posterior del operador. En automatización, se prefiere la acción monoestable y se crea cualquier autocontrol en el software para que los autómatas funcionen de forma autónoma cuando se presentan condiciones. Además, los sistemas autocontrolados son más adecuados para la liberación automática durante intervenciones de seguridad.

representado por software.
Las mismas conexiones tienen una correspondencia directa con el software, por ejemplo, el paralelo representa la operación booleana[5] OR y la serie la operación AND.
El diagrama de cableado, de las salidas, da sentido a la conexión de la bobina del contactor.



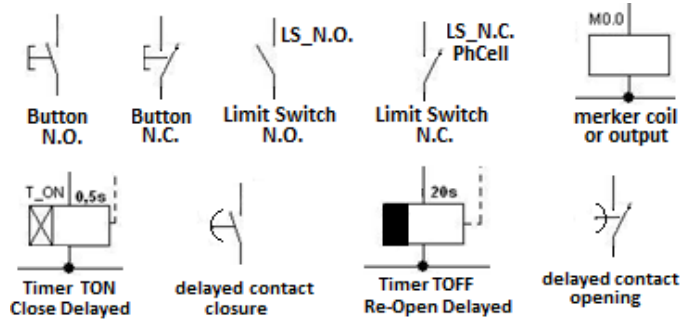
Es en este diagrama donde debe numerarse la longitud del cable entre el terminal %Q0.3 y el terminal A1 del contactor, aunque no esté presente aquí por simplicidad.
Los mismos terminales A1 y A2 estarán representados en el diagrama real en modo estándar como el "punto tachado" al que se combina la redacción consistente con el resto del diagrama, por ejemplo X1-1, etc.

[5] Las operaciones booleanas son aquellas contempladas en el álgebra booleana. George Boole fue un matemático y lógico inglés nacido en 1815, mucho antes de la llegada de los ordenadores. Aplicó álgebra normal a sistemas de numeración en base dos, postulando algunos elementos ausentes y esenciales, como la tautología, que permite simplificar señales redundantes. Por poner un ejemplo, la expresión "hoy llueve o no llueve" es una tautología, que puede lograrse mediante la paralela "O" de las dos posibilidades que pueden expresarse con el mismo contacto en la situación normalmente abierta o normalmente cerrada. Dado que este paralelo siempre es cierto (TRUE), puede eliminarse o mejor reemplazarse por un cable proveniente de la línea de corriente continua de +24V. Se conecta directamente a la bobina del relé de cierre de expresión. Así, se implementa la minimización booleana más importante, como se puede encontrar en la técnica del mapa de Karnaugh. Un compendio de las reglas básicas de la lógica booleana es la contribución de otro científico, De Morgan, que introduce las igualdades y, por tanto, la sustituibilidad entre OR y AND si se sujeta a las restricciones apropiadas de negación de señal.

Los elementos y símbolos fundamentales de la lógica funcional.
La lógica funcional expresa las acciones que debe realizar el controlador y es una ayuda útil para el desarrollo de software. Los elementos básicos son:

1. Línea eléctrica, horizontal en la parte superior, indicada por L+ o L si funciona con corriente continua o alterna. Como no es un circuito eléctrico sino un algoritmo que debe implementarse en la lógica interna del PLC, a través del software, tiene más sentido que siempre sea continuo.
2. Línea de retorno, indicada por M (tierra como cierre de una línea L+) o N (neutro como cierre de una línea alterna).
3. Caídas, que derivan hacia abajo del flujo lógico. En el real funcional, cada segmento comienza desde un descenso y no deriva horizontalmente del segmento anterior.
4. Contacto de cierre (es decir, cableado normalmente abierto), con una apariencia de contacto seco dibujado hacia la izquierda del descenso.
5. Contacto de apertura (es decir, normalmente cerrado con cable), con una apariencia de contacto seco dibujado hacia la derecha del descenso.
6. La bobina temporizada se retrasa en la excitación y su contacto al abrir y cerrar.
7. La bobina de temporización se retrasa en la desenergización y su contacto al abrir y cerrar.
8. Contador, representado como una bobina rectangular de carga pero con entradas para arriba, abajo, ajuste y RESET. El valor predefinido también debe representarse.

Los símbolos útiles para escribir una funcional se resumen en la imagen de abajo. Como podemos ver, los triángulos en los topes de los extremos, como las barras de cierre, no son esenciales, ya que los distintos contactos representan solo el estado booleano de un bit dentro o presente en el bloque terminal, no el objeto electromecánico en el campo.

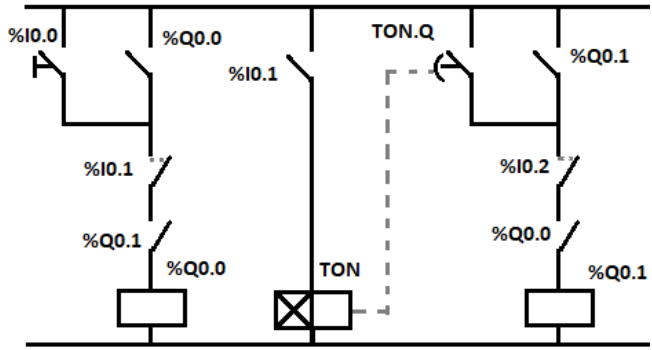


Dado que, en este contexto, el diagrama funcional no es un esquema sino un algoritmo a seguir para implementar el programa Ladder, la cantidad de símbolos y las reglas del dibujo se simplifican considerablemente.

Primero, los símbolos presentes en la funcional representan las señales que proporcionan los objetos electromecánicos.

El diagrama funcional es una representación esquemática e intuitiva del programa que tendrá que implementarse con el software. Cumple la misma función que el diagrama de flujo [6]en lenguajes de alto nivel.

Solo en una primera aproximación la función rotada 90 grados representa la implementación del software. En general, esto no es cierto. En este sentido, debe considerarse la **Regla del Gottardo.**

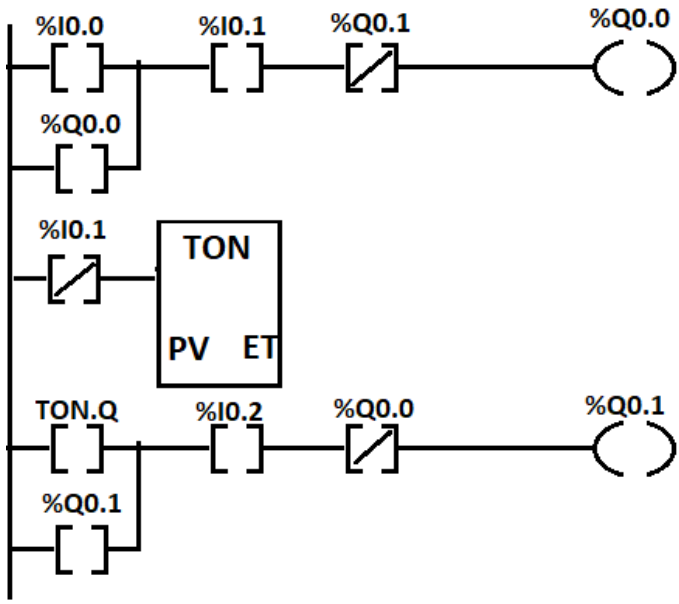


Regola del Gottardo: Quando un contatto* è delegato allo sgancio di un'auto ritenuta, viene acquisito nel software al contrario di come è rappresentato nel funzionale.

***si intende contatto fisico di tipo fine corsa elettromeccanico. Quando il medesimo segnale perviene da una virtualizzazione o è in qualche modo simulato la regola ne viene confermata in quanto l'eventuale contatto negato presente allo sgancio non è un contatto reale o non è un fine corsa elettromeccanico.**

[6] Los diagramas de flujo no deben aplicarse en la fase de desarrollo de los programas PLC porque son insuficientes. Estos se sustituyen, cuando es posible, por diagramas funcionales. Los conjuntos de símbolos para diagramas de flujo son ovalados para salidas y llegadas, rectángulos de varios tipos para acciones, diamantes para los tomadores de decisiones tipo FI, flechas para indicar las direcciones de los flujos. Por analogía, los diagramas funcionales tienen su propio conjunto limitado de símbolos: el contacto abierto a la izquierda del descenso, el contacto cerrado a la derecha del descenso, la bobina, las bobinas temporizadas, mientras que las lógicas se hacen con paralelos para los OR y series para los AND.

El diagrama funcional propuesto conduce a la implementación del código estandarizado IEC 61131 que se indica a continuación.
Debe prestarse atención a los contactos %I0.1 y %I0.2, que en la función se muestran en el lado derecho del rápel, por lo que normal 1 -> pulsado 0.
En estos segmentos de la Escalera, en cambio, normalmente están abiertos, y esto se explica en la mencionada "regla del Gotardo".[7]
La necesidad de esta norma surge tras treinta años de experiencia en el ámbito educativo y no es menos trivial que afirmar que dos líneas paralelas nunca se cruzan.



Nota sulla sicurezza intrinseca:
Un sistema in autoritenuta si trova in sicurezza intrinseca quando la rottura o la mancanza del sensore di fine corsa invia al controllore il medesimo stato dell'intervento.

Técnica de demostración con contranominales: Para la regla del Gotardo aplicamos la técnica de la demostración contranominal, que consiste en partir de la negación de la tesis para llegar a la negación de la hipótesis, y no en contradicción con la hipótesis.
Ambas tesis son la necesidad de negar el contacto de liberación del coche de sujeción respecto a cómo se representa en el funcional.

[7] Hasta que se demuestre lo contrario, el autor utiliza el estatus de investigador con doctorado para validar su existencia y la demostración formulada aquí.

La negación de la tesis es que no se niega el estado de contacto físico en el pitch.

La hipótesis, sin embargo, es que los contactos físicos en el campo delegados para la liberación del coche de sujeción normalmente están con cables cerrados.

Supongamos, absurdamente, que se niega al estado en el campo del contacto físico delegado a desenganchar.

Se puede observar que ninguna señal llega al terminal delegado para leerla.

Dado que esto está conectado, según la lógica interna del software, mediante una combinación booleana AND, el resultado combinatorio, sea cual sea el estado del botón de arranque, será cero, impidiendo que la bobina se bloquee en autoenganche.

La **negación de la hipótesis** es que el estado del contacto de liberación delegado al coche de sujeción no debe ser denegado respecto a cómo se muestra en el funcional.

Al no negar el estado de este contacto, se implementa el teorema de doble negación booleano[8] realizado por la afirmación "not(not(unhooked) = unhooked", y por tanto el motor no arranca porque está desenganchado.

la doble negación permite eliminar una negación de la expresión, es decir, colocar el estado del contacto en el segmento escalera 7 abierto en lugar de cerrado, validando definitivamente la regla del Gotardo, hasta que se demuestre lo contrario.

C.Q.D.

Por favor, ten en cuenta: La hipótesis menciona "contactos cerrados en el campo", por lo que no debería aplicarse cuando los contactos mencionados no existen, es decir, cuando están virtualizados en un panel HMI.

Demostración de lo contrario: El contacto entrelazado no debe revertirse porque no satisface las suposiciones iniciales al demostrar la tesis.

[8] Demostrable tanto mediante los sistemas deductivos de Hilbert como en la lógica proposicional clásica incluso con la única ayuda de tautologías. Según las reglas de la doble negación, teorémicas en álgebras booleanas y combinatorias, entonces la doble negación permite eliminar una negación de la expresión, validando definitivamente la regla del Gotardo, hasta que se demuestre lo contrario.

Declaración técnica correcta

En el caso de **contacto físico de un interruptor electromecánico de límite cableado (NC) normalmente** cerrado y delegado a la **liberación de un dispositivo auto-retenedor**, se aplica la siguiente regla:

En el software PLC, el contacto debe programarse respecto a su representación funcional.

En la práctica:

- En el campo → contacto NC (cerrado en reposo).
- En escalera→ debe insertarse como un contacto normalmente abierto (—| |—) asociado al bit de entrada.

Esto se debe a que el PLC lee **los niveles de lógica** eléctrica, no la simbología funcional.

Análisis de por qué es necesaria la inversión

Estado físico real (interruptor de límite NC)

Estadofísico	**Terminal PLC**	**Bits de entrada**
No presionado	1	TRUE
Prensado	0	FALSE

Pero funcionalmente:

- Presionado = condición de disparo
- No presionado = máquina habilitada

Así, el significado funcional se invierte respecto al nivel lógico leído.

Forma correcta de autorretención l

Ecuación correcta:
Q_{n+1}= (START ∨ Q_n) ∧ ENTRADA

Donde:

- INPUT es el bit leído por el PLC (NC cableado).
- Cuando se pulsa el interruptor de límite → ENTRADA = 0 → el AND cancela toda → liberación garantizada.

Si el programador lo niega en el software:
Q_{n+1}= (START ∨ Q_n) ∧ ¬ENTRADA

entonces:

- En reposo → ENTRADA = 1 → ¬1 = 0 → la máquina nunca arranca.
- Pulsado → INPUT = 0 → ¬0 = 1 → la máquina puede arrancar con el interruptor de límite activo (error grave).

Y este es exactamente el núcleo de la demostración contranominal reportada en el texto.

Corolario: Qn+1 representa el estado asumido por la salida en el ciclo de escaneo tras la evaluación del estado lógico de la ubicación de memoria, según el tiempo de ejecución del PLC. En sistemas PLC que cumplen con el modelo de ejecución del lenguaje IEC 61131-3, las entradas se muestrean al inicio del ciclo de escaneo y se copian a la imagen de proceso de las entradas, mientras que las asignaciones de salida actualizan la imagen de proceso de las salidas durante la ejecución del programa. Solo al final del ciclo se transfiere la imagen de las salidas a los terminales físicos. Esto implica que el programa puede observar el estado actualizado de una bobina en el mismo ciclo, mientras que el campo lo verá en el siguiente ciclo.

Traducción lógica de la demostración

Hipótesis:
El contacto físico es contacto cero en el campo.

Tesi:
No debería ser invertido en software.

Demostración contranominal:

- Si lo niegas en el software,
- Entonces evitarías que el pestillo se enganche o activarías la máquina en la condición de liberación,
- Por lo tanto, la hipótesis de un correcto funcionamiento sería falsa,
- por lo tanto, no debe ser negado.

La referencia a la doble negación:

$$\neg(\neg S) = S$$

solo sirve para formalizar que:
NC físico + NO contacto en la escalera = doble negación implícita → comportamiento consistente.

Cuando NO se aplica

La nota al pie del libro es correcta:
La regla solo se aplica **si el contacto es electromecánico físico NC cableado en el campo**.

No se aplica cuando:

- La señal es virtual (HMI).
- La señal se simula.
- El contacto físico es NO.
- Estos son enclavamientos lógicos internos.

Resumen operativo (de técnico de planta)

Si el dispositivo de seguridad es:

1. El NC físico cableado → NO se invierte en el PLC.
2. NO se evalua → física según la función.
3. Lo virtual → no es ninguna regla del Gotardo.

Ahora podemos desarrollar los siguientes puntos:

- reescribiendo la demostración en una forma rigurosa de lógica proposicional pura (sin narrativa),
- Muestra el caso con temporizador TON como en el diagrama presentado arriba
- Analizar el error típico en TIA Portal al confundir la representación funcional con el estado eléctrico.

sin perjuicio de la regla del Gotardo, ya presente en el texto con ISBN: 9798345038970, procedamos desarrollando el análisis de los tres puntos enumerados.

Mantenemos como axioma la formulación de la Regla del Gottardo, ya presente en muchos textos, y procedemos de forma estructurada en los tres puntos requeridos.

Demostración en lógica proposicional estricta

Definiciones formales

Que así sea:

- F = estado físico del contacto NC en el campo
- F=1 → circuito cerrado (condición normal)
- F=0 → circuito abierto (interruptor de límite pulsado)

I = bit leído por el dispositivo cableado PLCENC:

$$I=F$$

- Q = salida auto-cerrada
- S = inicio

Autocontrol correcto:

$$Q_{n+1}= (S \vee Q_n) \wedge I$$

Tesis (Regla del Gotardo)

No niegues que estoy en software.

Demostración contranominal

Supongamos que la tesis es falsa: el contacto está denegado en el software.

Entonces:

$$Q_{n+1} = (S \vee Q_n) \wedge \neg I$$

Caso 1 — Condición normal (F = 1 → I = 1):

$$Q_{n+1} = (S \vee Q_n) \wedge 0 = 0$$

La sujeción no puede ser sujeta → comportamiento no conforme.

Caso 2 — Interruptor de límite de límite activo (F = 0 → I = 0):

$$Q_{n+1} = (S \vee Q_n) \wedge 1$$

La máquina puede arrancar en condiciones de disparo → grave infracción funcional.

Por lo tanto, la negación de la tesis implica la negación de la hipótesis del correcto funcionamiento.

Q.E.D.

Caso con TON (como en el diagrama adjunto)

En tu diagrama **aparece UN MONTÓN** que retrasa el desacoplamiento o la activación.

Te lo recordamos:

TON.Q=1 solo después PT con IN = 1

Se il finecorsa NC viene negato nel software:

- En condiciones normales, → IN = 0 → TON nunca arranca.
- En condiciones de viaje → IN = 1 → TONELADA de pieza cuando no debería.

Consecuencias:

- Invertir los tiempos.
- Auto-cierre que se activa con el interruptor de límite pulsado.
- Retrasos en el viaje en lugar de retrasos en el inicio.

Con los temporizadores, la violación se vuelve aún más insidiosa porque introduce dinámicas temporales inconsistentes.

Error típico en el Portal TIA

Error común en los estudiantes:

Confondere:

- Símbolo gráfico funcional (NC dibujado)
- Estado lógico leído por PLC

En TIA:

- Contacto —|/|— no significa "contacto físico NC".
- Significa "condición lógica denegada".

Muchos programadores:

1. Cableado NC en el campo.
2. Colocan —|/|— en la escalera.
3. Reciben doble error de inversión.

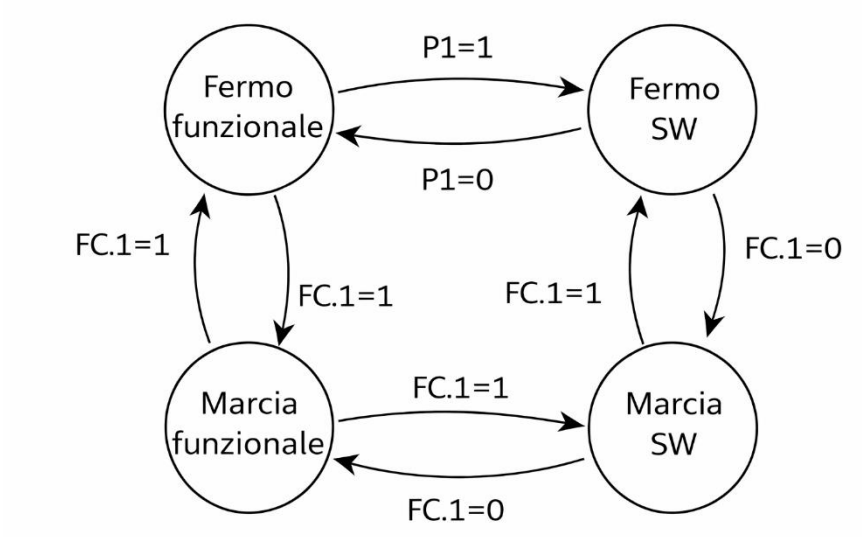
Efecto:

- Una máquina que no arranca.
- O empieza en condiciones de seguridad abierta.
- Diagnósticos engañosos.

La Regla del Gotardo sirve precisamente para evitar esta ambigüedad entre los niveles físico y booleano.

Representación de autómatas de estados finitos

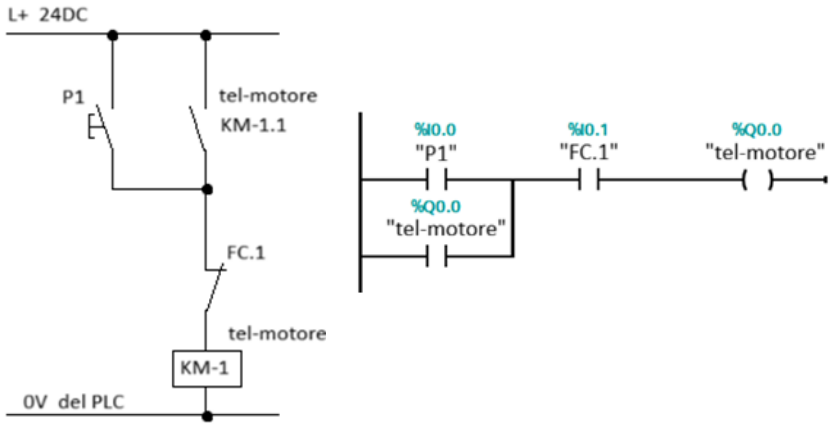
El autómata de estados finitos representado en el grafo formaliza el comportamiento de un circuito con **autocontención electromecánica** cuando se adquiere dentro de un sistema de control programable según el principio llamado *regla de Gotardo*. Este principio establece que, si un contacto se delega a la liberación de un autoenganching, su adquisición lógica en el software ocurre con polaridad inversa respecto a la representación funcional presente en el esquema pseudoelectromecánico.



Regola del Gottardo

El sistema en cuestión consiste en un circuito de control de un contactor, indicado por KM-1, equipado con autoenganche mediante un contacto auxiliar del mismo contactor en paralelo con el botón de arranque P1.
La cadena de registro se realiza mediante el contacto FC.1 colocado en serie con el carrete.
En el dominio funcional electromecánico, el comportamiento del sistema es bistable: la activación momentánea del comando de inicio energiza la bobina contactor, que luego mantiene su estado

mediante el contacto auto-enganchado; Abrir el contacto de parada interrumpe la corriente en la bobina y vuelve a su estado de reposo.
Cuando este comportamiento se implementa dentro de un PLC, la lógica debe interpretarse a la luz del patrón cíclico de escaneo y el uso de imágenes de proceso de entradas y salidas.
En este contexto, el autómata introduce una distinción conceptual entre el **estado funcional del circuito físico** y el **estado de software interno al controlador**, produciendo así cuatro estados distintos que describen la combinación de ambas representaciones.
Los estados llamados *Parada* Funcional y *Corrida* Funcional describen el comportamiento esperado del sistema físico, mientras que los *estados SW Stop* y *SW Run* representan el estado de la variable lógica interna que realiza el autoenganche en el programa PLC.
En el funcionamiento del autómata, pulsar el botón P1 determina una transición del dominio funcional parado al estado correspondiente del software, en el que la lógica auto-enganchada es adquirida por el controlador.



Este paso refleja el hecho de que, durante el ciclo de escaneo, la activación de la entrada provoca la asignación de la bobina lógica que implementa el autoenganche.
Una vez establecida esta condición, el autómata evoluciona hacia el estado de marcha, en el que tanto el modelo funcional como el

modelo de software representan consistentemente la excitación del contactor.

La presencia del contacto FC.1, por otro lado, introduce la condición de liberación autobloqueante. En el ámbito funcional, abrir el contacto inmediatamente provoca la interrupción de la cadena de suministro de la bobina. Sin embargo, en el dominio del software, este evento se captura como un cambio en la entrada que provoca la transición al estado de espera solo mediante la actualización de la lógica de auto-bloqueo en el ciclo de escaneo. En esta fase, emerge el principio expresado por la regla del Gotardo: el contacto que representa funcionalmente la condición de parada se trata en el software con una polaridad lógica opuesta a la representación eléctrica, ya que la variable interna que mantiene el autoenganche debe desactivarse cuando el contacto está abierto.

Por tanto, el autómata representa una formalización del desalineamiento semántico entre la representación electromecánica del circuito y su implementación lógica en el PLC.

A través de la distinción entre estados funcionales y estados de software, es posible describir rigurosamente el proceso por el cual se establece y posteriormente libera el bloqueo, destacando cómo la adquisición de las entradas y la actualización de las salidas en el ciclo de escaneo introducen una transformación lógica que justifica la inversión conceptual del contacto de parada. En este sentido, el autómata proporciona una base formal para interpretar la regla del Gotardo como una propiedad emergente del modelo de ejecución de controladores programables y de la traducción de circuitos de relé en lógica de software.

Analisi critica del libro (ISBN 9798345038970)

Valutazione tecnica, non editoriale.

Fortalezas

1. Introduce la formalización lógica (contranominal, doble negación).
2. Conecta cableado físico y álgebra booleana.
3. Estimula el razonamiento estructurado, no solo la "programación por imitación".
4. Bien para los estudiantes de ITI/ITS que necesitan entender por qué.

Valoración general

Desde un punto de vista técnico:

1. Conceptualmente sólido.
2. Configuración lógica correcta.
3. Útil para formación técnica y profesional.

Desde un punto de vista educativo:

1. Bueno para estudiantes motivados.
2. Puede ser complejo para quienes no tienen una base en lógica formal.
3. Se habría beneficiado de un mayor rigor simbólico y menos retórica.

No es un manual industrial avanzado, pero para el objetivo de ITI/IPSIA/ITS, escuelas técnicas italianas, está por encima de la media de los textos introductorios.

Formalización según IEC 61131-3

IEC 61131-3 define los lenguajes (LD, FBD, ST, SFC) y el modelo de ejecución cíclica.

Modelo lógico de autocontrol

Variables:

```
VAR
    Start     : BOOL;   // Botón NO
    StopNC    : BOOL;   // Entrada de contacto físico NC
    Motor     : BOOL;   // Salida auto-cerrada
END_VAR
```

Ingesta física (Regla del Gotardo – hipótesis del libro):

- STOP contacto físico por cable NC.
- Entrada → Descanso = TRUE.
- Activada → Entrada = FALSE.

Implementazione corretta in Structured Text

```
Motor := (Start OR Motor) AND StopNC;
```

Esta forma es:

- Determinista,
- reset-dominant,
- coerente con logica fail-safe.

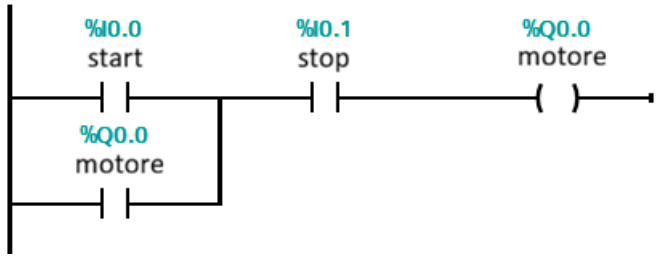
Implementación incorrecta (violación)

```
Motor := (Start OR Motor) AND NOT StopNC;
```

Consecuencias:

- En reposo → NOT StopNC = FALSE → El motor nunca arranca.
- En condicion en parada → NOT StopNC = TRUE → Posible inicio peligroso.

Forma correcta equivalente de escalera



Nota: El contacto StopNC es **NO en la escalera**, porque representa el nivel lógico de lectura, no la naturaleza física del dispositivo.

Tabla de Verdad Completa

Consideramos:

$$Q_{n+1} = (S \vee Q_n) \wedge I$$

Donde:

- S = Start
- I = StopNC
- Qn = Estado anterior

Caso corretto

S	Q_n	I (NC)	Q_{n+1}	Significado
0	0	1	0	Empresa
1	0	1	1	Inicio
0	1	1	1	Retención
1	1	1	1	Retención
X	X	0	0	STOP siempre dominante

Propiedad fundamental:

$$I = 0 \Rightarrow Q_{n+1} = 0$$

Reset dominante garantito.

Caso incorrecto (Negación de software)

$$Q_{n+1} = (S \vee Qn) \wedge \neg I$$

S	Q_n	I	Q_{n+1}
0	0	1	0
1	0	1	0 □
0	1	1	0 □
1	0	0	1 □
0	1	0	1 □

Infracciones:

- Incapaz de arrancar en estado normal.
- Posible inicio con STOP activo.

Análisis desde una perspectiva de seguridad funcional

Referencias regulatorias:

- EN ISO 13849-1
- IEC 62061

Principio de seguridad relevante

Fail-safe principle:

La pérdida de energía o la apertura del circuito deben conducir al estado seguro.

Con contacto NC:

- Rotura de cable → Entrada = 0

- PLC Derecho FALSE
- AND Cancelar salida
- Motor se paradas

Si violas la Regla del Gotardo:

- Rotura de cable → Entrada = 0
- NOT 0 = 1
- Posible consentimiento para la marcha

Esto es un compromiso:

- Categoría de circuito
- Nivel de rendimiento (PL)
- Integridad funcional

Dominanza del reset

En los sistemas de seguridad, el RESET debe ser:

- prioritario,
- inenmascarable,
- No autorreiniciable.

La forma correcta:

$$Q=Fset \wedge Fsafe$$

donde:

$$Fsafe = \bigwedge_i Ii$$

Cada entrada de seguridad está en global Y.

Conexión con la seguridad de los PLC

En PLC, el sistema de seguridad (es decir, CPU F):

1. Los contactos de seguridad se tratan como señales de "alto = seguro".
2. La negación arbitraria en el software rompe la consistencia SIL/PL.

La Regla del Gottardo, en este sentido, no solo es didáctica sino coherente con la arquitectura certificada.

Evaluación técnica en profundidad del libro

Análisis más detallado.

Corrección conceptual

Interpretación correcta del informe:

1. Nivel eléctrico
2. Nivel lógico
3. Representación por Ladder

Ajuste correcto de dominante por RESET

Rigor formal

Parcialmente estricto:

1. Usa correctamente el contranominal.
2. Pero falta una formalización sistemática con tablas de verdad.
3. Las referencias a los sistemas de Hilbert son teóricamente correctas pero no están formalmente desarrolladas.

Nivel educativo

Para los objetivos de ITI/IPSIA/ITS escuela tecnica Italiana:

- Por encima de la media.
- Estimula el razonamiento lógico.
- Introduce conceptos de álgebra booleana aplicada.

Límite:

- A veces lenguaje redundante.
- El formalismo no siempre lineal.

Valoración general

Desde un punto de vista técnico-profesional:

Es un buen texto didáctico operativo con ambición lógico-formal.

No es un manual avanzado de seguridad funcional, pero está bien estructurado para:

1. Entiende la autodisciplina,
2. evitar errores típicos en PLC,
3. desarrollar razonamiento booleano aplicado.

Formalización de la regla del Gotardo en el álgebra booleana axiomática

Definiciones y suposiciones

Sea una autocontención definida como controlada por:

- $S \in \{0,1\}$ — comando START (NO)
- $F \in \{0,1\}$ — estado físico del contacto de la salida NC
- $I \in \{0,1\}$ $I \in \{0,1\}$ — valor lógico leído por PLC
- $Qn \in \{0,1\}$ — estado de la bobina en el ciclo n
- $Qn+1$ — estado en el siguiente ciclo

Hipótesis física (contacto NC por cable):

$$I - F$$

con:

1. $F=1 \rightarrow$ contacto cerrado (condición normal)
2. $F=0 \rightarrow$ contacto abierto (disparo)

Autocontrol correcto

Definición:

$$Q_{n+1} = (S \vee Q_n) \wedge I$$

Axiomas de álgebra booleana utilizados

Se consideran los axiomas clásicos (de Huntington):

1. Conmutatividad
   $A \vee B = B \vee A$
   $A \wedge B = B \wedge A$
2. Distributividad
   $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$

3. Elementos neutros
   $A \wedge 1 = A$
   $A \vee 0 = A$
4. Elementos nulos
   $A \wedge 0 = 0$
   $A \vee 1 = 1$
5. Complementación
   $A \wedge \neg A = 0$
   $A \vee \neg A = 1$

Propiedad de RESET dominante

Proposición 1
La función es dominante en RESET respecto a I.

Demostración

Que así sea $I = 0$.

$Q_{n+1} = (S \vee Q_n) \wedge 0$

Por axioma 4:

$A \wedge 0 = 0$

entonces:

$Q_{n+1} = 0$

Independientemente de S e Qn

Q.E.D.

Infracción de las normas de Gottardo

Supongamos la denegación de software:

$Q_{n+1} = (S \vee Q_n) \wedge \neg I$

Sea I = 1 (Condición normal).

$Q_{n+1} = (S \vee Q_n) \wedge 0 = 0$

La función pierde la capacidad de establecer.

Sea I = 0 (Lanzamiento).

$Q_{n+1} = (S \vee Q_n) \wedge 1 = S \vee Q_n$

El RESET ya no es dominante.

Conclusión formal:
La negación de software destruye la propiedad de dominancia del RESET.

Modelado de sistemas de estados finitos (FSM)

Definición del sistema

Definimos la máquina de estados:

$M = (X,U,\delta)$

donde:

- $X = \{OFF,ON\}$
- $U = \{S,I\}$
- $\delta : X \times U \rightarrow X$

Función de transición correcta

Estado actual	S	I	Siguiente estado
OFF	0	1	OFF
OFF	1	1	ON
ON	0	1	ON
ON	1	1	ON
*	*	0	OFF

Propiedades:

$I = 0 \Rightarrow \delta(x, S, I) = OFF$

El estado OFF es absorbente en comparación con I = 0.

Caso incorrecto (denegación de software)

Estado	S	I	Siguiente estado
OFF	1	1	OFF
OFF	1	0	ON

Se genera una transición peligrosa:

$(OFF, S=1, I=0) \rightarrow ON$

El sistema no es monótono respecto a la variable de seguridad.

Propiedad formal violada

Definimos las propiedades de seguridad:

$I = 0 \Rightarrow X = OFF$

En el sistema equivocado, esta propiedad no es invariante.

Análisis de doble canal – Categoría 3/4

Referencia: EN ISO 13849-1

Categoría de Arquitectura 3

Características:

- Dos canales independientes.
- Monitorización de la consistencia.
- Un solo fallo no provoca la pérdida de la función de seguridad.

Definimos:

- I1 = canal A
- I2 = canal B

Función de seguridad:

$F_{safe} = I_1 \wedge I_2$

Autorretención:

$Q_{n+1} = (S \vee Qn) \wedge I_1 \wedge I_2$

Propiedades de tolerancia a fallos

Caso: Rotura de cable en el canal A

$I_1=0$

Entonces:

$Q_{n+1} = 0$

Sistema seguro.

Caso de reversión incorrecta de software

$Q_{n+1} = (S \vee Qn) \wedge \neg I_1 \wedge \neg I_2$

Rotura de cable:

$I_1=0 \Rightarrow \neg I_1=1$

Posible consentimiento para la marcha.

Pérdida por categoría.

Categoría 4

Requiere:

- Redundancia
- Diagnóstico continuo
- Detección de acumulación de fallos

Modelo lógico:

$F_{safe} = (I_1 \wedge I_2) \wedge D$

donde D = diagnósticos.

La Regla del Gotardo es una condición necesaria para mantener la semántica "1 = seguro".

La inversión en software altera la correspondencia semántica entre:

- nivel eléctrico,
- nivel lógico,
- Nivel de certificación PL.

Conclusiones de ingeniería

1. La Regla del Gotardo garantiza formalmente:

    1. reset dominante,
    2. invariancia del estado seguro,
    3. monotonía respecto a la variable de seguridad.

2. En álgebra booleana se puede demostrar mediante axiomas clásicos.
3. En la teoría de autómatas es una propiedad del estado absorbente[9].
4. En seguridad funcional, es un requisito estructural mantener las categorías PL y SIL[10].

---

[9] En el contexto de la teoría de autómatas, un **estado** *absorbente* es un estado de un autómata de estados finitos caracterizado por la propiedad de que, una vez alcanzado, el sistema ya no puede evolucionar a diferentes estados. Formalmente, dado un autómata determinista , se dice que un estado es absorbente si para todo símbolo de entrada . Esta propiedad implica que todas las transiciones que salen del estado conducen de vuelta al propio estado, convirtiéndolo en un punto de estabilidad del sistema dinámico descrito por el autómata. En términos de modelado, los estados absorbentes se usan frecuentemente para representar condiciones terminales, condiciones de error irrecuperable o configuraciones de equilibrio de las que no se espera una evolución adicional del comportamiento del sistema.$A = (Q, \Sigma, \delta, q_0) q_a \in Q x \in \Sigma \delta(q_a, x) = q_a$

[10] El **Nivel de Rendimiento (PL)** y **el Nivel de Integridad de Seguridad (SIL)** son métricas regulatorias utilizadas para cuantificar el nivel de fiabilidad requerido de las funciones de seguridad en los sistemas de control. El Nivel de Rendimiento, definido en ISO 13849-1, expresa la capacidad de una función de seguridad para reducir el riesgo a través de cinco niveles discretos, indicados desde PL hasta (menor) a PL y (máximo), determinados en función de la probabilidad media de fallo peligroso por hora y la estructura arquitectónica del sistema. El Nivel de Integridad de Seguridad, definido en IEC 61508 y también adoptado en la industria por IEC 62061, representa de forma similar la probabilidad de que una función de seguridad realice correctamente su tarea cuando sea necesaria; se divide en cuatro niveles crecientes, SIL 1–SIL 4, asociados a intervalos cuantitativos de probabilidad de fallo peligroso. Ambas clasificaciones son herramientas para la evaluación de riesgos y el diseño de sistemas de seguridad, permitiéndote especificar y verificar el grado de integridad que requieren las funciones de seguridad.

La tesis de Gottardo

Fundamentos lógico-formales de la regla del Gotardo en PLC y sistemas de seguridad

Una discusión sobre álgebra booleana, teoría de autómatas, lógica temporal y seguridad funcional

Resumen

La llamada ***Regla del Gottardo***, formulada en el contexto de la programación PLC con contactos físicos normalmente cerrados delegados a la liberación de un sistema auto-retenible, puede interpretarse no solo como una regla práctica de taller, sino como una propiedad estructural demostrable en álgebra booleana, que puede modelarse mediante autómatas de estados finitos y verificarse mediante lógica temporal formal.

En este trabajo se demuestra que esta regla garantiza la dominancia del RESET, la preservación del invariante de seguridad y la consistencia semántica entre el nivel eléctrico y el nivel lógico, siendo una condición necesaria —aunque no suficiente— para el mantenimiento de los requisitos de seguridad previstos por las normas EN ISO 13849-1 e IEC 62061.

Estructura lógica de la función de autocontención

Consideremos un sistema de control de motor con un botón de arranque normalmente abierto y un contacto físico de parada normalmente cerrado, cableado en serie, leído por un PLC conforme a IEC 61131-3.

Indicamos con:

1. S el comando de inicio,
2. F el estado físico del contacto NC,
3. I el valor lógico leído por el PLC,
4. Qn el estado de la bobina en el ciclo nº

Dado que el contacto está cableado NC en el campo, la lectura de variables coincide con el estado eléctrico real:

$I = F$

Con la convención:

$F=1 \Rightarrow$ circuito cerrado (condición normal)
$F=0 \Rightarrow$ circuito abierto (disparo)

La función correcta de autocontención se define por:

$$Q_{n+1} = (S \vee Q_n) \wedge I$$

Esta expresión incorpora la estructura clásica de latch con el RESET dominante implícito en la conjunción final.

Demostración en álgebra booleana axiomática

El álgebra booleana clásica, basada en los axiomas de Huntington, proporciona las herramientas para una demostración rigurosa.

Queremos demostrar que la función anterior tiene la propiedad de dominancia de la liberación.

Sea $I = 0$. Entonces:

$$Q_{n+1} = (S \vee Q_n) \wedge 0$$

Para el axioma del elemento nulo:

$$A \wedge 0 = 0$$

sigue inmediatamente:

$$Q_{n+1} = 0$$

La variable de salida es independiente de S y Qn. El RESET es dominante.

Ahora analicemos la función obtenida al negar el contacto en el software:

$$Q_{n+1} = (S \vee Q_n) \wedge \neg I$$

Poniendo I=0 obtenemos:

$$Q_{n+1} = (S \vee Q_{n)} \wedge 1 = S \vee Q_n$$

El RESET pierde la dominancia. La propiedad de seguridad ya no es una consecuencia axiomática de la estructura.

De ello se deduce que la regla del Gotardo no es una convención gráfica, sino una propiedad estructural de la función booleana.

Modelado como sistema de estados finitos

Se define el autómata determinista:

$M = (X, U, \delta)$

donde el espacio de estados es X= {OFF, ON}, el conjunto de entradas es U= {S, I} y la función de transición es:

$\delta(x, S, I) =$

- OFF se I=0
- ON se I=1 ∧ (S=1 ∨ x=ON)
- OFF altrimenti

La propiedad fundamental del sistema es que el estado OFF es absorbente respecto a la condición I=0. Formalmente:

$$I = 0 \Rightarrow \delta(x, S, I) = OFF \ \forall x, S$$

Si se introduce la denegación inadecuada del contacto, el autómata admite la transición:

$$(OFF, S=1, I=0) \rightarrow ON$$

La condición de seguridad ya no es invariante. El sistema pierde monotonía respecto a la variable de seguridad.

Verificación por lógica temporal

La propiedad deseada puede expresarse en Lógica Temporal Lineal (LTL).

Sea definida la proposición atómica:

$$\text{Seguro: } =(Q=\text{APAGADO})$$

La propiedad fundamental se convierte en:

$$G(I=0 \rightarrow \text{Seguro})$$

donde G es el operador "globalmente".

En el modelo correcto, la fórmula es válida en todos los caminos temporales.

En el modelo equivocado, existe un camino tal que:

$$I=0 \wedge S=1 \wedge Qn=OFF$$

Pero:

$$Q_{n+1}=ON$$

La fórmula LTL está falsificada. Por tanto, la regla puede considerarse una condición necesaria para la validez del invariante de seguridad en lógica temporal.

En los terminis CTL:

$$AG(I=0 \rightarrow AX(Q=OFF))$$

es cierto solo en la formulación correcta.

Extensión a sistemas de doble canal de categoría 3/4

En el marco regulatorio de la EN ISO 13849-1, la Categoría 3 requiere redundancia con la detección de fallos.

Se introducen dos canales independientes, I1 e I2, ambos con NC físicos.

La función se convierte en:

$$Q_{n+1}= (S \vee Q_n) \wedge I_1 \wedge I_2$$

Supongamos que hay un solo fallo con una rotura de cable en el primer canal:

$$I_1=0$$

Por propiedad axiomática:

$$Q_{n+1}=0$$

Se mantiene la función de seguridad.

Si niegas las señales en el software:

$$Q_{n+1}= (S \vee Q_n) \wedge \neg I_1 \wedge \neg I_2$$

La ruptura de cables produce:

$$I_1= 0 \Rightarrow \neg I_1 = 1$$

El fracaso ya no es inherentemente seguro. La estructura pierde la propiedad de seguridad y puede comprometer el Nivel de Rendimiento.
En la Categoría 4, donde también se requiere detección de acumulación de fallos, la consistencia semántica "1 = condición segura" es fundamental para el análisis probabilístico y la correcta implementación de mecanismos diagnósticos.

Análisis probabilístico y relación con la SIL

En el contexto de la IEC 62061, la probabilidad de un fallo peligroso depende de la capacidad del sistema para pasar automáticamente a un estado seguro en caso de fallo.

Con los contactos NC y la función final AND, la probabilidad de fallo peligroso depende de la probabilidad de fallo simultáneo de los canales.

En el software, la anulación única no crea un apagado seguro. La probabilidad de fallo peligroso aumenta, modificando la PFHd y, por tanto, la SIL alcanzable.

La Regla del Gotardo se convierte así en una condición estructural para mantener baja la tasa peligrosa de fallo.

Conclusiones generales

La Regla del Gotardo, lejos de ser una convención gráfica o un simple hábito de enseñanza, puede interpretarse como:

una propiedad algebraica de la dominancia del reset, una propiedad estructural de un autómata determinista, un invariante que puede expresarse en lógica temporal, condición necesaria para mantener el Nivel de Rendimiento en sistemas redundantes.

Su validez es estricta dentro de la hipótesis del contacto físico NC por cable en el campo. Aparte de esta hipótesis, la regla sigue siendo válida, sobre la base de la prueba contranominal, y debe reconsiderarse a la luz de la semántica de la señal.