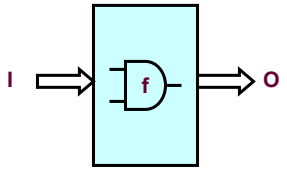
Automatismos secuenciales

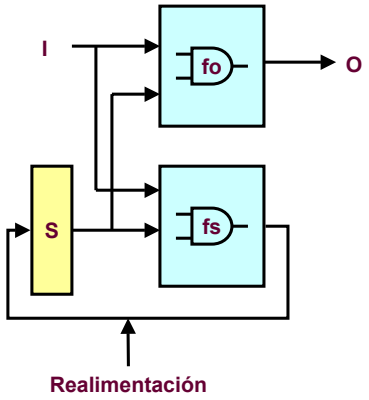
Automatismo combinacional

- **Las salidas del sistema de control sólo dependen de las entradas al sistema de control.**
- **Tipos de variables:**
  - Variables de entrada (I)
  - Variables de salida (O)
- **Una sola función lógica:**
  - O=f(I)
- **Entre los cambios de la entrada y salida hay un retardo que depende de la implantación tecnológica: PLC, puertas lógicas, relés, etc.**



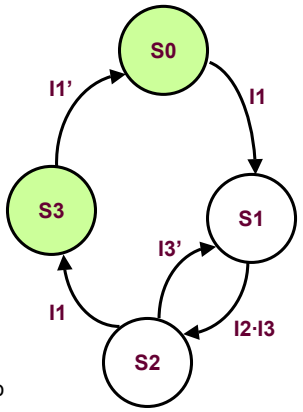
Automatismo secuencial

- **Las salidas del sistema de control dependen de las entradas y de valores almacenados internamente en el sistema de control.**
  - El sistema puede almacenar su historia.
- **Tipos de variables lógicas:**
  - Variables de entrada (I)
  - Variables de estado (S): variables internas que almacenan el estado del sistema. Normalmente aparece una memoria para almacenarlas.
  - Variables de salida (O)
- **Dos funciones lógicas: O=fo(I,S), S=fs(I,S)**
- **El sistema está realimentado.**



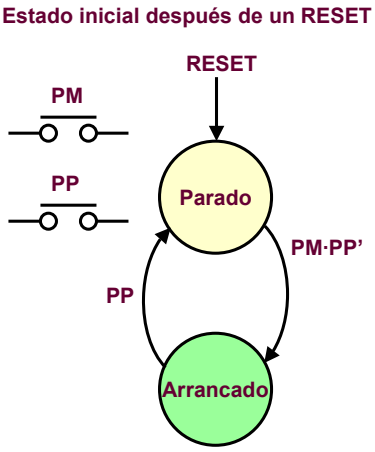
Máquina de estados

- **Cada combinación posible de valores ({0,1}) de las variables de estado representa un estado del sistema (S0, S1).**
- **La evolución de las variables de estado se puede representar mediante una máquina de estados:**
  - Los círculos indican los estados posibles. La etiqueta asociada indica el nombre de cada estado.
  - Las flechas indican el cambio de un estado a otro. Las etiquetas asociadas indican la condición (función lógica) que deben cumplir las variables de entrada parar ir de un estado a otro.
- **El número de estados posibles determina el mínimo número de variables de estado necesarias (Ver codificación).**

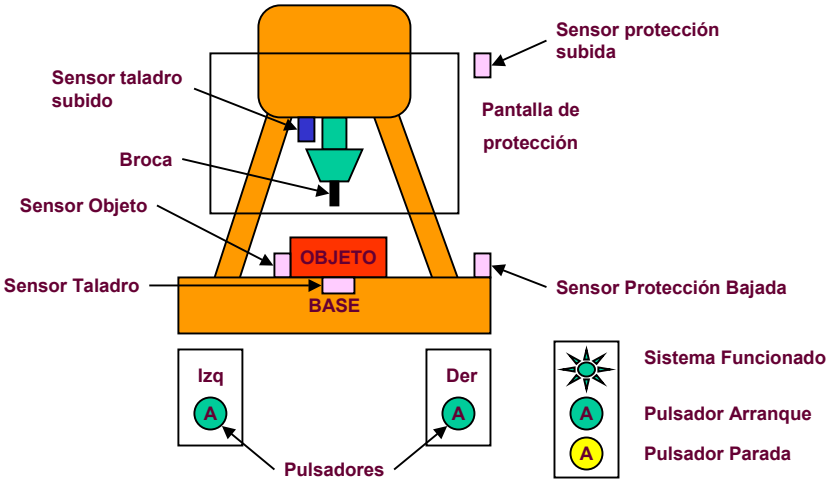


Ejemplo: Marcha/Paro de motor

- El pulsador de *Marcha (PM)* provoca que el sistema pase de estado de parado a marcha. La señal de arranque del motor está asociada al estado de *Arrancado.*
- El pulsador de *Paro (PP)* provoca que el sistema pase del estado de *Arrancado* a *Parado.*
- Para guardar los dos posibles estados internos del control sólo necesitamos un bit (*0-Parado, 1-Arrancado*).
- La función lógica del motor es *m=Arrancado.*



Ejemplo de programación más complejo



Especificación del control de la taladradora

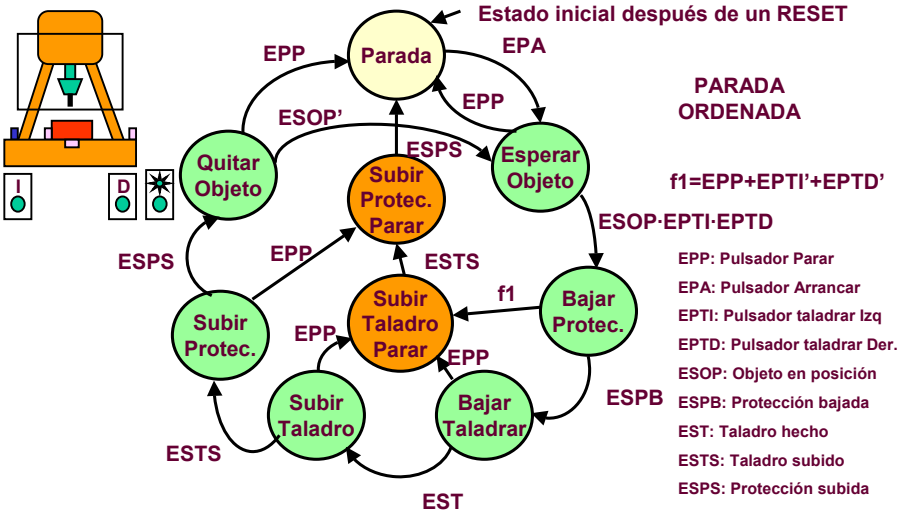
- **1. El sistema arranca al pulsar el pulsador de arranque (Se señaliza mediante una bombilla).**
- **2. Se coloca el objeto a taladrar según la posición. Se activa el sensor de objeto en posición.**
- **3. El operador pulsa los botones de taladrar izquierdo y derecho.**
  - Seguridad en la operación
- **4. Se baja la cabina de protección mediante un motor. Mediante un sensor se detecta que ha llegado a su posición de bajada.**
  - Una vez bajada, el operador puede liberar los pulsadores.
- **5. La taladradora comienza a girar y a descender.**
- **6. Cuando completa el agujero se activa el sensor de taladro realizado.**
- **7. Se sube el taladro y a continuación la protección.**
- **8. Si en cualquier momento se pulsa paro, la taladradora se para, después de haber subido el taladro y la protección si es necesario.**

Identificación entradas/salidas y asignación.

ENTRADAS	LOGICA	ETIQUETA	DIRECCIÓN
Reset	N	RESET	E124.0
Pulsador de Arranque	P	EPA	E124.1
Pulsador de Parada	P	EPP	E124.2
Pulsador Taladrar Izquierdo	P	EPTI	E124.3
Pulsador Taladrar Derecho	P	EPTD	E124.4
Sensor Objeto en Posición	P	ESOP	E124.5
Sensor Protección Bajada	P	ESPB	E124.6
Sensor Taladro Hecho	P	EST	E124.7
Sensor Taladro subido	P	ESTS	E125.0
Sensor Protección Subida	P	ESPS	E125.1

SALIDAS			
Bombilla Sistema Funcionando	P	SBSF	A124.0
Relé Motor Bajar Protección	P	SMBP	A124.1
Relé Motor Subir Protección	P	SMSP	A124.2
Relé Motor Bajar Taladradora	P	SMBT	A124.3
Relé Motor Subir Taladradora	P	SMDT	A124.4
Relé Motor Taladradora	P	SMT	A124.5

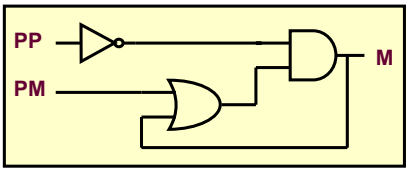
Ejemplo máquina de estados: control taladradora



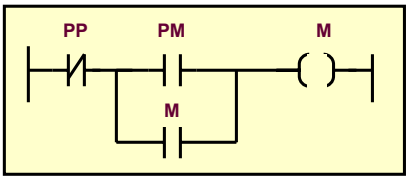
Realimentación

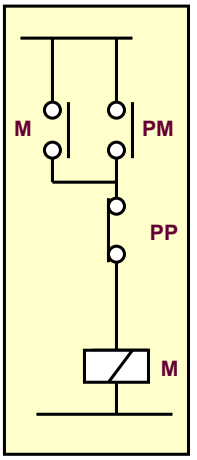
- **¿Cómo puede trabajar S=f(S,I)?**
- **Teóricamente S a la derecha y a la izquierda son la misma.**
- **Si S=f(S,I), ¿cuál es el estado final, dado un estado inicial S0?**
  - Difícil de resolver directamente
  - Puede no haber solución: no se puede obtener un estado estable.
- **En la práctica hay retardos:**
  - S a la derecha es el estado en el instante anterior
  - S a la izquierda es el nuevo estado calculado, para el estado siguiente.
- **La solución para automatismos es romper la realimentación**
  - La ecuación se divide en dos:
    - Sig=f(S,I) Primero se calcula el estado siguiente
    - S=Sig Segundo, el estado siguiente se asigna al estado actual
  - Las dos asignaciones se realizan en momentos diferentes
    - ¿Cómo? Señal de sincronización

Marcha/Paro con realimentación directa

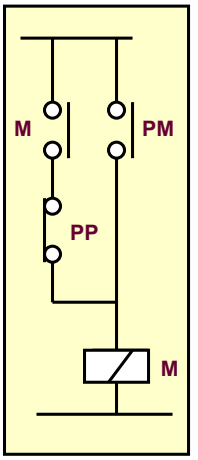


¡¡SOLO PARA CASOS MUY SENCILLOS!!



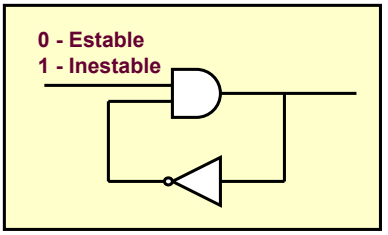


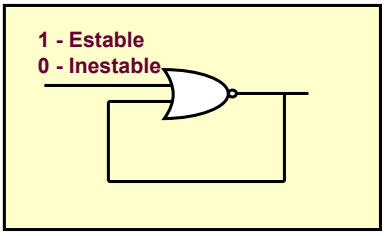
Paro/Reset dominante

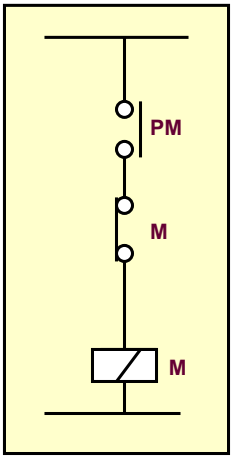


Marcha/Set dominante

Casos problemáticos

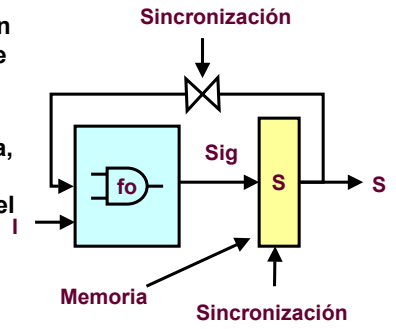


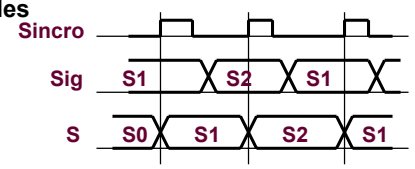




Sincronización

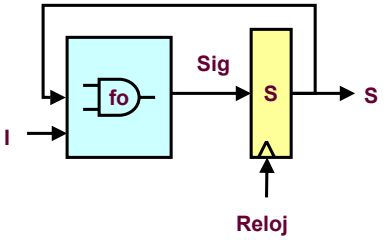
- Mientras la señal de sincronización está desactivada: S es diferente de Sig. Sig se calcula a partir de S. S no puede cambiar de valor.
- Cuando la sincronización se activa, en S se graba el valor Sig. En la realimentación se debe mantener el valor anterior de S, para que no cambie Sig.
- Los estados cambian de forma síncrona con la señal de sincronización y no con las señales de entrada.
- Fácil de implantar:
  - Hardware: registros con reloj.
  - Programa: ciclo de scan.

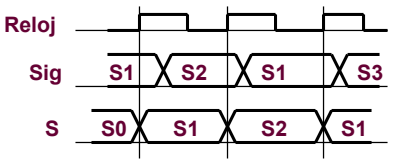




Sincronización en hardware

- Se utilizan registros gobernados por una señal de reloj. El registro tiene n bits de entrada y n bits de salida.
- Cuando llega el flanco de subida del reloj (o de bajada) la entrada del registro se carga en la salida.
- Aunque la salida del registro ha cambiado, la entrada Sig no cambia hasta que pasa un cierto tiempo porque fo tiene retardo. Cuando cambia Sig ha desaparecido el flanco del reloj.
- Sólo hay que preocuparse de diseñar Sig=f(S,O).

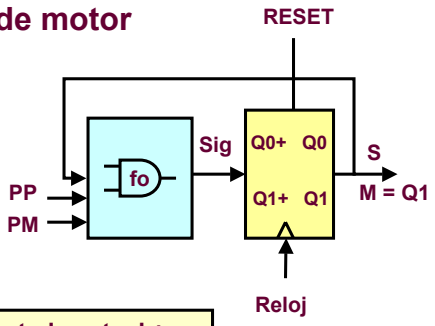




Ejemplo: arranque/parada de motor

- 2 estados: 2 bits de memoria

Sig = {Q1+, Q0+}
S = {Q1,Q0}
S0 = 01
S1 = 10
Q0+ = Q0·PM' + Q1·PP + Q0'·Q1'
Q1+ = Q1·PP' + Q0·PM·PP'



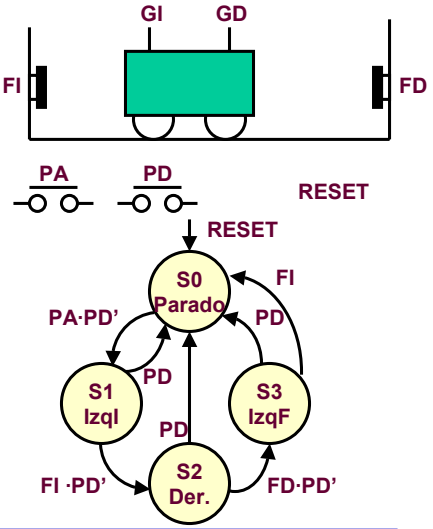
Sig = Mantenerse en el estado actual +
+ Condiciones para ir a él desde otro estado

- RESET: cuando esta señal se activa, todos los elementos de memoria toman un valor conocido. En el caso de los registros lo normal es tomar valor 0.

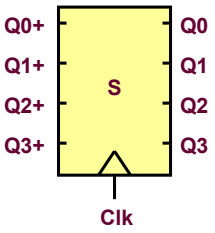
Simplificación a 1 sólo bit de estado:
Sig = S·PP'+PM·PP'
Sig = PP'(S+PM)
M = S
Ecuación del relé

Ejemplo: Control de un móvil

- Al pulsar *Arrancar PA*, el móvil si está parado parte hacia la izquierda. Se detiene cuando se activa el final de carrera FI. A continuación, inicia su marcha hacia la derecha hasta llegar a FD. Nuevamente, inicia su marcha hacia la izquierda y se detiene al llegar a ella.
- Si se pulsa *Detener PD* el móvil se para inmediatamente.



Ejemplo (II)



- **Sig = {Q3+,Q2+,Q1+,Q0+}**
- **S = {Q3, Q2, Q1, Q0}**
  - S0 = 0001
  - S1 = 0010
  - S2 = 0100
  - S3 = 1000

La codificación y las ecuaciones lógicas fuerzan que cuando un bit está a 1 el resto está a 0

- **Q0+ = Q0·PA' + Q1·PD + Q2·PD + Q3·PD + Q3·FI +Q0'·Q1'·Q2'·Q3'**
- **Q1+ = Q0·PA·PD' + Q1·PD'·FI'**
- **Q2+ = Q1·FI·PD' + Q2·PD'·FD'**
- **Q3+ = Q2·FD·PD' + Q3·PD'·FI'**
- **GI = S1 + S3 = Q1 + Q3**
- **GD = S2 = Q2**

Recuerde: Inicialmente habría que poner GD = Q0'Q1'Q2Q3', pero siempre se cumple que cuando Q2=1 entonces Q0=Q1=Q3=0 en funcionamiento normal, debido a la codificación utilizada

- **El RESET pondría todos los registros a 0. Al desactivar el reset Q0 se activaría.**