CIRCUITOS DIGITALES Y MICROCONTROLADORES 2022

Facultad de Ingeniería UNLP

Máquinas de Estados Finitos y su implementación en Software

Ing. José Juárez

Abril 2021

Modelización con Máquinas de Estados

- **Abstracción:** Se trata de definir un problema complejo en un conjunto de principios básicos. Si podemos modelar el sistema en base a estos principios, podremos entender mejor el problema, separando "que es lo que se está haciendo" de los detalles del "como se esta haciendo".
- **Definición 1:**Una una Máquina de Estados Finitos (MEF) es un modelo abstracto del "comportamiento" del sistema, basado en principio simples.
- **Definición 2:** una Máquina de Estados Finitos (MEF) es un modelo matemático (Teoría general de autómatas) usado para describir el comportamiento de un sistemas que puede ser representado por un número finito de estados, un conjunto de entradas y una función de transición que determina el estado siguiente en función del estado actual y de las entradas.
- **Ventajas de la implementación con "maquina de estados"**
  - Hace más sencilla la comprensión del sistema y su funcionamiento,
  - Desde el punto de vista del software es más sencillo de mantener ya que se pueden agregar o quitar estados sin modificar el resto
  - Es más sencillo de depurar,
  - Es más sencillo de verificar
  - Es más sencillo de Optimizar

Modelización con Máquinas de Estados

- **El modelo :**
  - Contiene los estados, las entradas y las reglas bien definidas para cambiar de estado
  - El siguiente estado depende de las entradas y del estado actual
  - Se puede especificar mediante un "diagrama de estados" o "tabla de transiciones de estados"
  - Cada transición implica diferentes respuestas o acciones del sistema

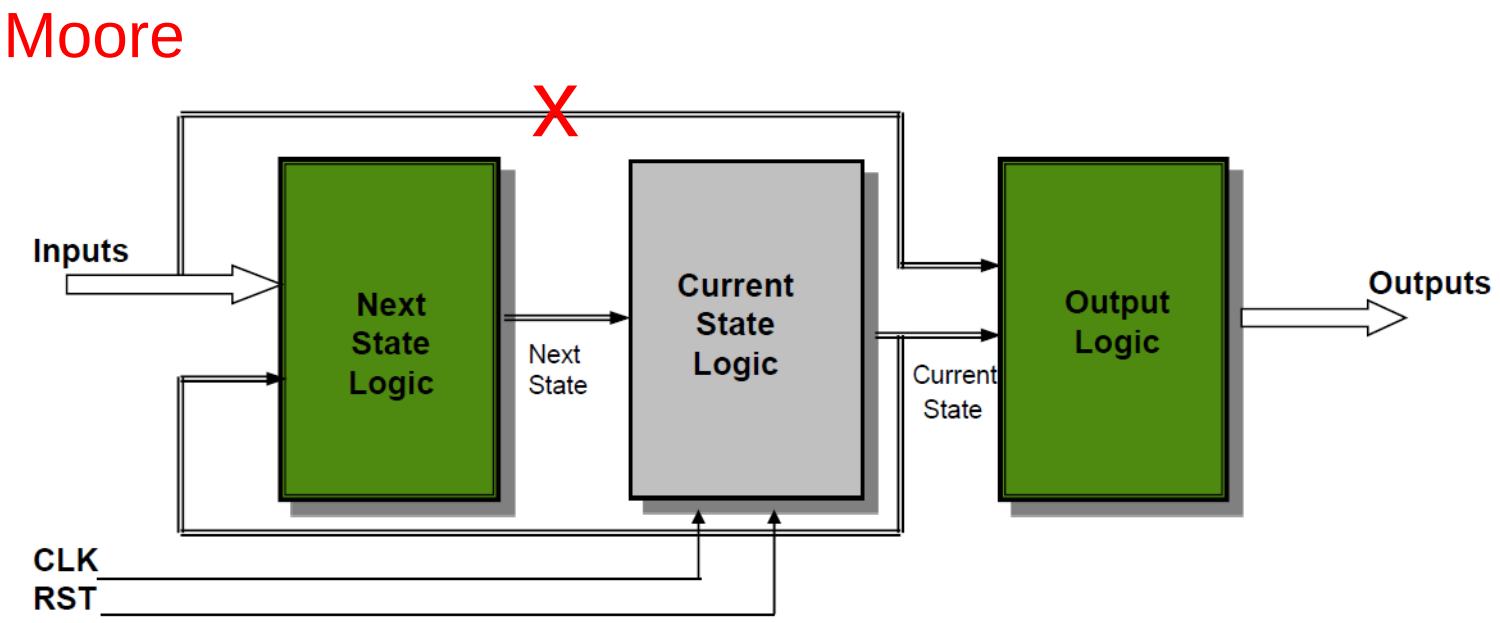
    => 2 posibilidades para actualizar las salidas

- **Mealy**
  - La salida depende del estado actual y de las entradas
  - Son propensos a este modelo, los sistemas donde la salida provoca el cambio de estado. Por ejemplo, el movimiento de las articulaciones de un robot producen el cambio de estado (parado-sentado)
- **Moore**
  - La salida del sistema depende solo del estado actual
  - Puede haber múltiples estados con la misma salida, pero para cada estado el significado es diferente. La salida guarda estrecha relación con el estado. Por ejemplo, un controlador de Semáforo

- Ambos modelos son intercambiables pero es mejor optar por la forma que representa de manera más natural el problema.

Implementación: Hardware

- Modelo General de un circuito secuencial sincrónico:
  (Modelo de Mealy)



Cortesia: Cristian Cisterna: Escuela ACSE

Especificación

- 1-Descripción matemática (6-tupla)

MEF={S, so, Sf, Σ, δ, Γ}

S: Conjunto Finito de estados del sistema

so $\in$ S: Estado inicial

Sf $\subseteq$ S: conjunto de estados finales

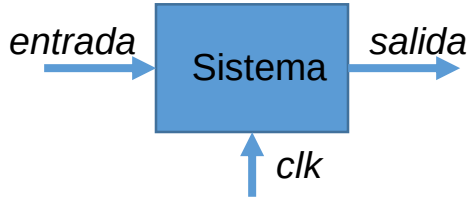
Σ: alfabeto finito de entradas o eventos que producen cambios de estado

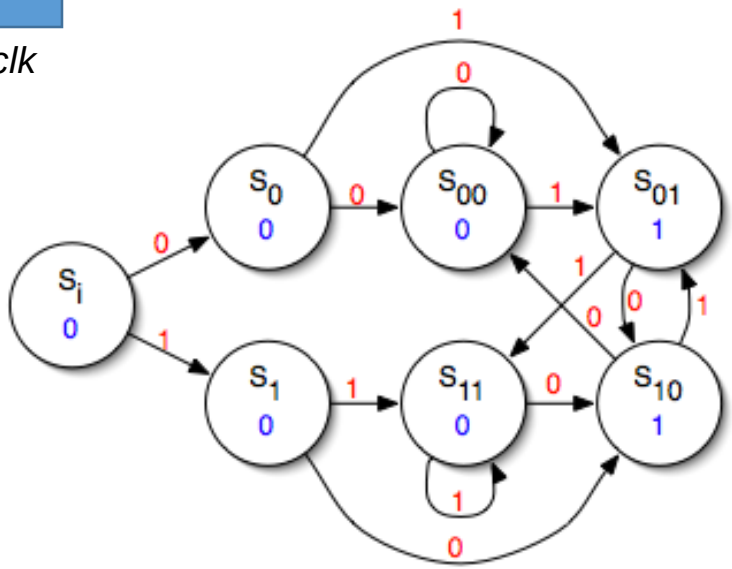
δ: función de transición Sx$\Sigma \rightarrow$ SxΓ

Γ: Conjunto finito de salidas

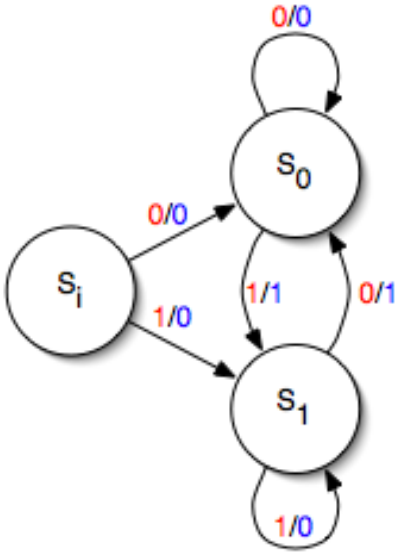
Especificación

- 2-Descripción por diagramas de estados
  - Ejemplo: Detector de flanco: la salida es la XOR de las dos entradas mas recientes
  - **Moore:**





  - **Mealy:**

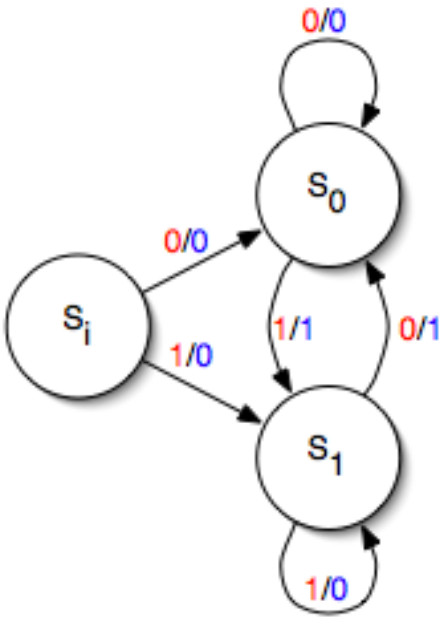


En rojo: entrada
En azul: salida

Especificación

- 3-Descripción por tabla de transición de estados

Estados\Entradas	1	0
Si	S1/0	S0/0
S0	S1/1	S0/0
S1	S1/0	S0/1



Implementación: MEF en software

- Del análisis de la especificación podemos extraer los siguientes elementos a implementar:

1) Definir el conjunto de estados

   estados: (S0, S1,…,Sn)

2) Definir el conjunto de entradas (alfabeto)

   entradas: (i0, i1,…,im)

3) Definir el conjunto de salidas

   salidas: (O0, O1,…,Ok)

4) Definir una función de transición de estados

   tabla : [1…n] [1…m]

5) Definir procedimiento para establecer el estado inicial

```
Iniciar_MEF {
  estado=Si  //i e 0..n
  salida=Oj  //j e 0..k
}
```

Implementación: MEF en software

6) Definir procedimiento para actualizar la MEF

```
Actualizar_MEF(){
  Leer(&entradas);
  estado=tabla[estado][entradas];
  Act_Salidas(estado,entradas);
 }
```

Puede ser Bloqueante o No-bloqueante

Implementación por tablas

7) Definir procedimientos para ejecutar la MEF

```
Ejecutar_MEF{

  Iniciar_MEF();
  repetir siempre{
        Actualizar_MEF();
  }
}
```

Sin temporizar

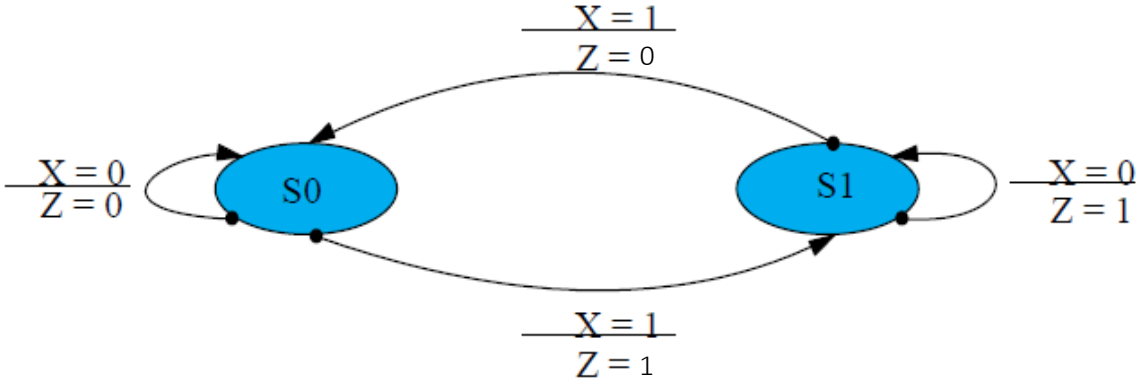
```
Ejecutar_MEF{

  Iniciar_MEF();
  repetir cada T segundos{
        Actualizar_MEF();
  }
}
```

temporizada

Implementación: switch-case

- Implementación a partir de la especificación:

  si examinamos un diagrama de estados, por ejemplo :



surge intuitivamente el siguiente razonamiento:

Estado S0:

```
If   (X==0)
 then S0 y Z=0
else  S1 y Z=1
```

Estado S1:

```
If   (X==0)
  then S1 y Z=1
else  S0 y Z=0
```

Notar que si fuera Moore Las salidas van fuera del if-else

¿cómo seleccionar entre S0 o S1? => switch- case

Implementación: switch-case

```
void ActualizarMEF(void)
{
  X=leerEntradas();
  switch(estado){
       Case S0:
              if(X==1)
               { estado=S1; Z=1;}
              else
               { estado=S0; Z=0;}
              break;
       Case S1:
              if(X==1)
               { estado=S0; Z=0;}
              else
               { estado=S1; Z=1;}
              break;
  }
}
```

Si hay múltiples entradas => otro switch

Implementación: switch-case

Definición y Declaración de Variables de estado →

```
typedef enum{S0,S1}state;
state estado;
```

Método de Inicialización →

```
Iniciar_MEF(){
 estado=S0; Z=0;
}
```

Ejecución de la MEF →

```
void main(void){
  Iniciar_MEF();

  while(1){
    Actualizar_MEF();
   }
}
```

Implementación: con punteros a función

```
typedef enum{S0,S1} state;

void fS0(void);
void fS1(void);
void (*MEF[])(void)={fS0,fS1};
state estado;

void main(void){
  Iniciar_MEF();
  while(1){
      Actualizar_MEF();
 }
}
void ActualizarMEF(void){
  X=leerEntradas();
 (*MEF[estado])();  //Ejecuta la función correspondiente
}
```

```
void fS0(void)
{
  if(X==1){
   estado=S1; Z=1;}
 else {
   estado=S0; Z=0;}
}

void fS1(void)
{
 if(X==1){
   estado=S0; Z=0;}
 else {
   estado=S1; Z=1;}
}
```

Implementación: con punteros a función

- ***Diferencias:***
  - En implementaciones con **switch()** los **case** de cada estado se evaluarán secuencialmente, equivale a una cadena de **if** consecutivos de resolución. No tarda lo mismo en ejecutar las actualizaciones según el caso.
  - En implementaciones con **punteros a función o tablas** el tiempo de acceso a las funciones es el mismo independientemente del valor de la variable de estado, equivale a un desvío selectivo de la ejecución del programa.
  - En general implementaciones con punteros o tablas de transición permiten uniformidad en el tiempo de acceso, son más compactas, pero ocupan más memoria.
- ***Otros tipos de implementaciones:***
  - Tabla de transición, listas enlazadas y otras.

Implementación: MEF en Software

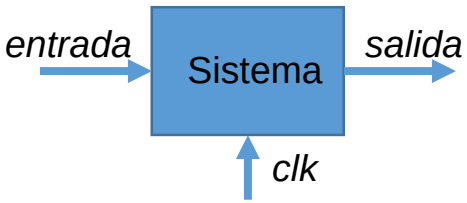
- ***MEF Temporizadas:*** Actualización periódica con Timer (polling periódico)

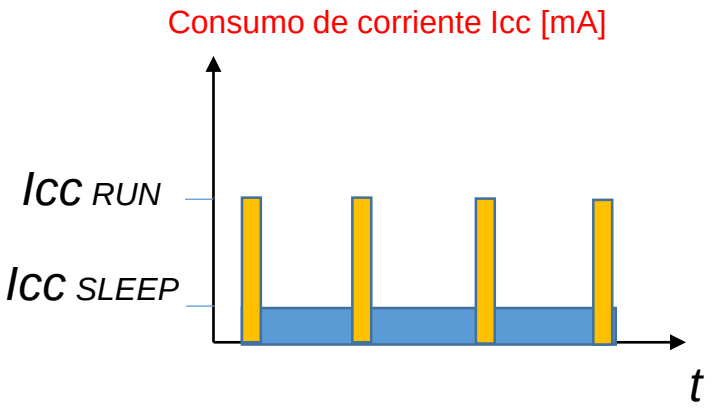
```
void main(void){
  Iniciar_MEF();

  while(1){
     if(FLAG_TIMER){
          Actualizar_MEF();
          FLAG_TIMER=0;
     }
     sleep();
}
```

Modificado x ISR() cada T seg.

Debe ser
No-bloqueante

Similar a la implementación en Hardware





Implementación: MEF en Software

- ***MEF jerárquicas:*** Un estado puede contener otra MEF

```
void ActualizarMEF(void)
{
  switch(estado){
       Case S0:
              Actualizar_MEF_S0(&params);
              break;
       Case S1:
              if(X==1)
               { estado=S0; Z=0;}
              else
               { estado=S1; Z=1;}
              break;
  }
}
```

Tiene que heredar todas las variables del estado superior

Implementación: MEF en Software

- ***MEF cooperativas:*** varias MEF independientes en ejecución simultánea

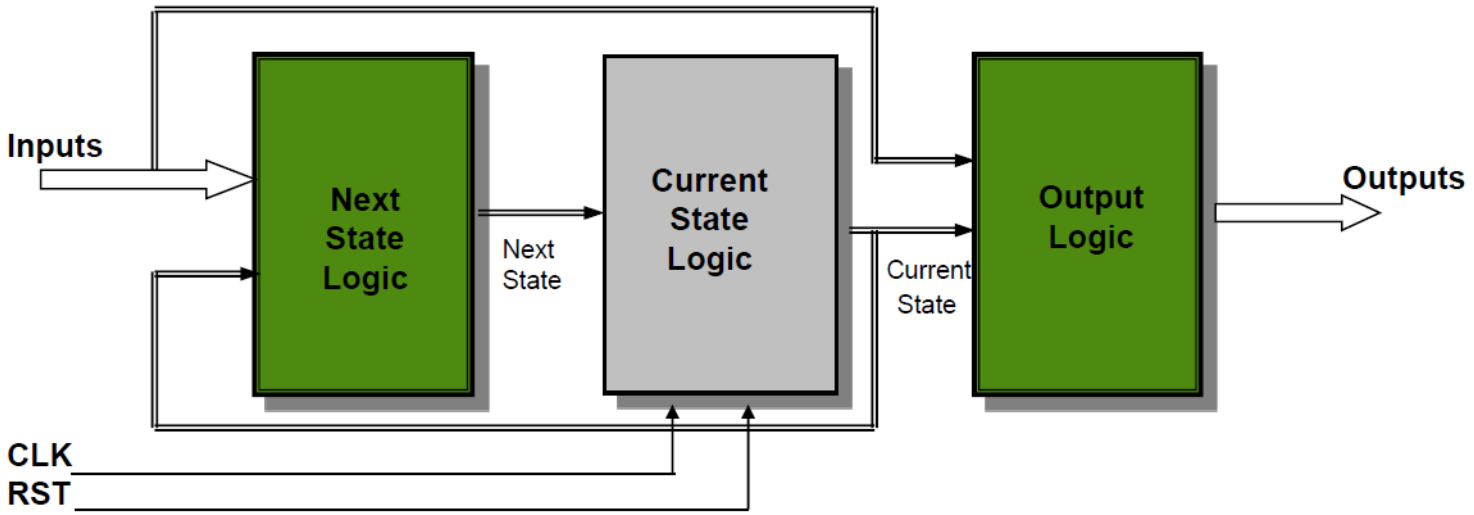
```
void main(void){
  Iniciar_MEF_1();
  Iniciar_MEF_2();

  while(1){
    Actualizar_MEF_1();
    Actualizar_MEF_2();
 }
}
```

1-No-bloqueantes
2-Independientes (2 diagramas)
3-Se pueden comunicar x mensajes asincrónicos
4-Se ejecutan hasta completar (run to completion)
5-Son cooperativas (corrutinas)

- ***Statecharts:*** Método gráfico (especificación UML) para especificar MEFs generalizadas (jerárquicas y concurrentes)

Implementación: MEF en Hardware

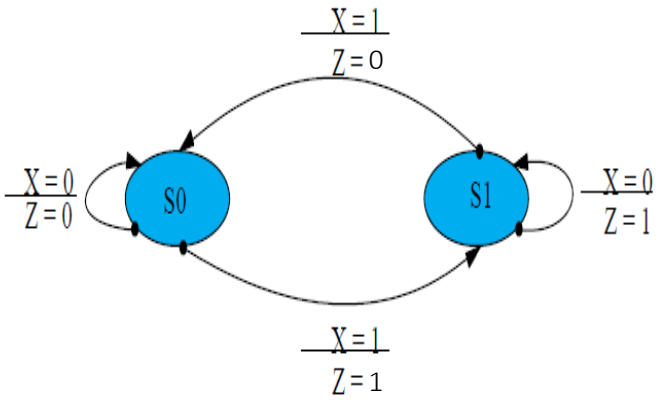
- Volvamos al modelo general de un circuito secuencial sincrónico: (Modelo de Mealy)
- Veamos como se implementaría el ejemplo anterior



Cortesia: Cristian Cisterna: Escuela ACSE

Implementación: MEF en Hardware

Partimos de la misma especificación:



La MEF se implementa en un circuito Secuencial descrito
En VHDL utilizamos las sentencias Process y Case-When

```
type state is (S0,S1);
...
process(clk, rst)
begin
 if(rst= '1')then
  state <= S0;
  Z <= '0';
 elsif(rising_edge(clk)) then
  case state is
   when S0 =>
     if(X='0') then
       state <= S0;
       Z <= '0' ;
     elsif(X='1') then
       state <= S1 ;
       Z <= '1';
     end if;
   when S1 =>
     if (X = '0') then. . .
     end if;
   end case;
 end if;
end process;
```

Implementación: comparación sintáctica

En C :

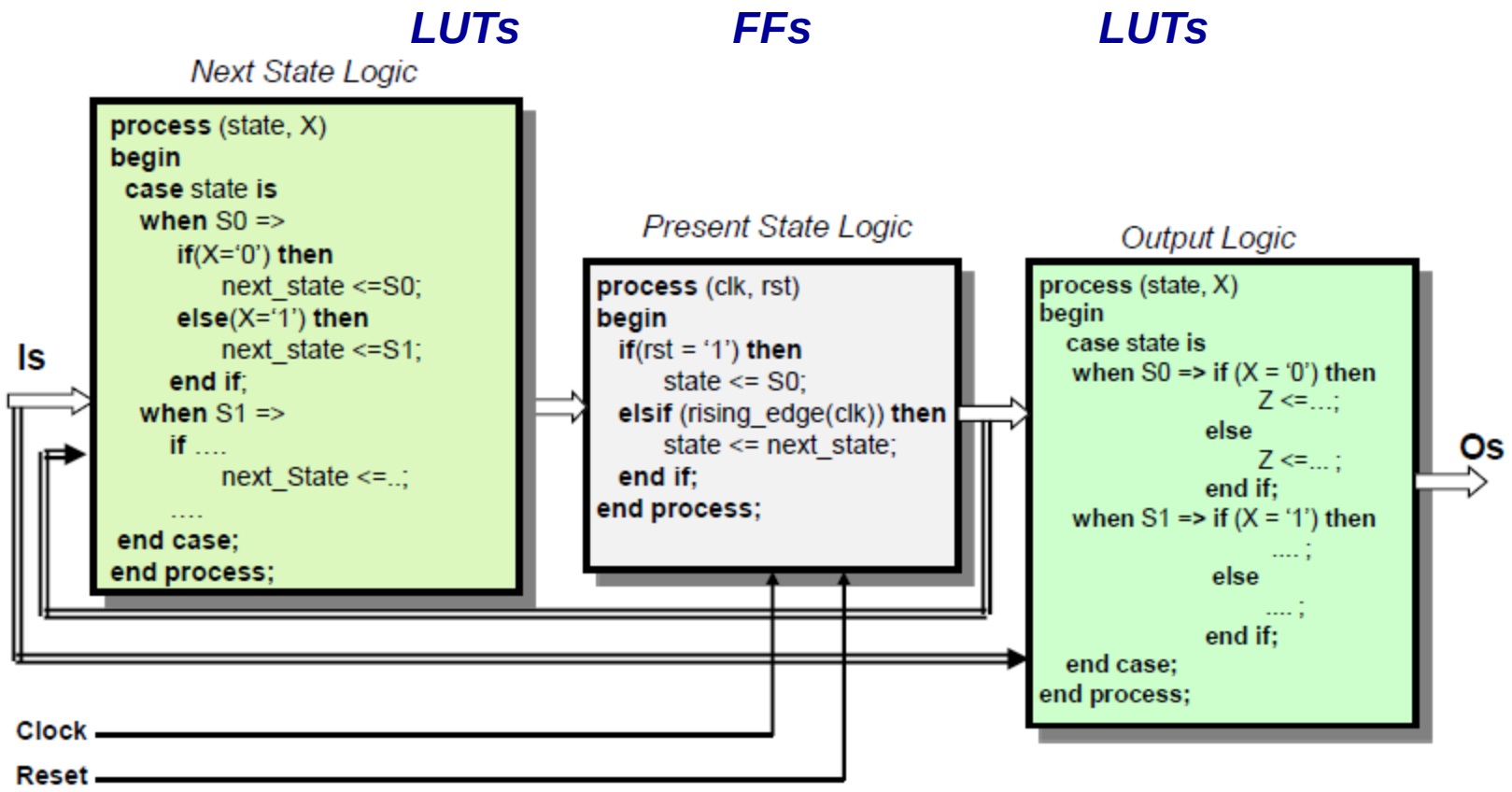
```
typedef enum{S0,S1} state;
state estado;

Iniciar_MEF(){
 estado=S0; Z=0;
}
void ActualizarMEF(void)
{
  switch(estado){
  Case S0:
        if(X==1)
         { estado=S1; Z=1;}
        else
         { estado=S0; Z=0;}
        break;
  Case S1:
        if(X==1)
         { estado=S0; Z=0;}
        else
         { estado=S1; Z=1;}
        break;
  }
}
```

En VHDL :

```
type state is (S0,S1);
...
process(clk, rst)
begin
 if(rst= '1')then
  state <= S0;
  Z <= '0';
 elsif(rising_edge(clk)) then
  case state is
   when S0 =>
     if(X='0') then
       state <= S0;
       Z <= '0' ;
     elsif(X='1') then
     state <= S1 ;
       Z <= '1';
     end if;
   when S1 =>
     if (X = '0') then. . .
     end if;
   end case;
 end if;
end process;
```

Implementación: MEF en Hardware

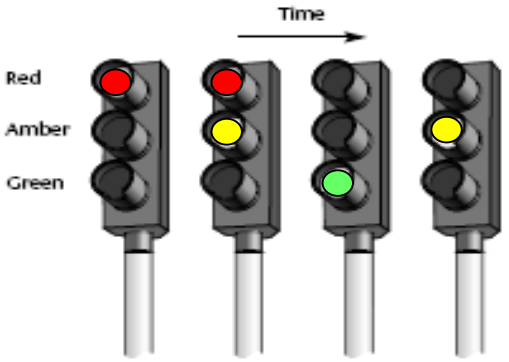
- Método formal en VHDL:



Cortesia: Cristian Cisterna: Escuela ACSE

Ejemplo 1

- Semáforo (MEF temporizada)



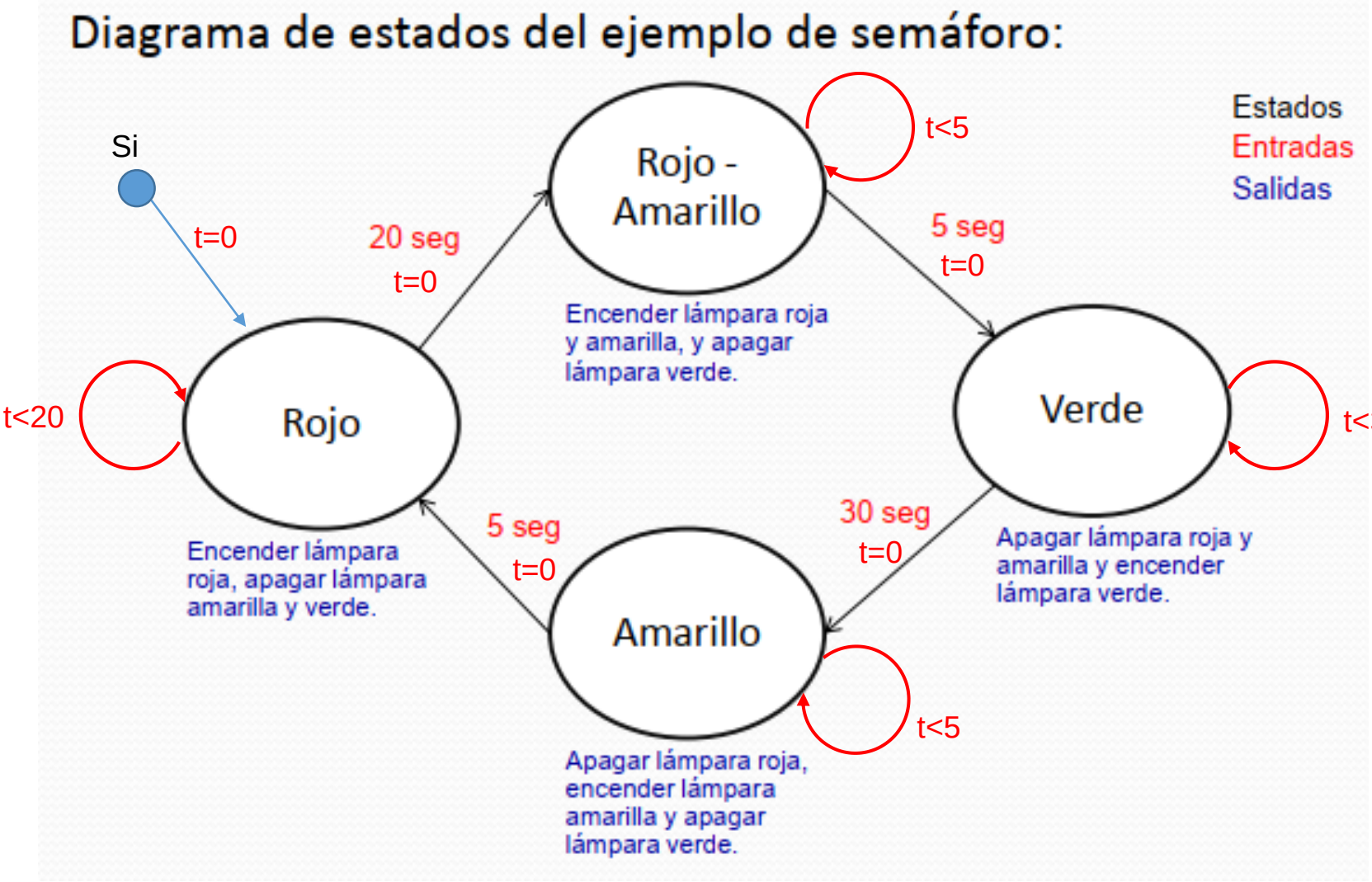
Estados y salidas (Moore):

- 1-rojo
- 2-rojo-amarillo
- 3-verde
- 4-amarillo

Cambio de estados x evento tiempo (entrada)

- Rojo: 20seg
- Rojo-amarillo: 5seg
- verde: 30seg
- Amarillo: 5 seg

Semáforo: Implementación MEF



Semáforo: Implementación MEF

```
// Posibles estados del sistema
typedef enum {RED, RED_AMBER,GREEN, AMBER} eLight_State;

// Tiempo de duración de cada estado
// (tiempo en segundos)
//
#define RED_DURATION 20
#define RED_AMBER_DURATION 5
#define GREEN_DURATION 30
#define AMBER_DURATION 5
```

Semáforo: Implementación MEF

```
// ------ variables privadas----------------------------------

static eLight_State Light_state; // Estado actual del sistema
static long Time_in_state; // Tiempo en el estado

/*-------------------------------------------------------------*-
TRAFFIC_LIGHTS_Init()
-*-------------------------------------------------------------*/

void TRAFFIC_LIGHTS_Init ( eLight_State START_STATE)
{
  Light_state = START_STATE; // poner un estado inicial
}

void main(void)
{
  MCU_Init();

  TRAFFIC_LIGHTS_Init (RED); // inicializar MEF

  for(;;)  {
        TRAFFIC_LIGHTS_Update(); // MEF temporizada aprox. c/1 seg
        Delay_ms(1000);
  }
}
```

(Moore)

```
void TRAFFIC_LIGHTS_Update(void)
{
 switch (Light_state)
 {
  case RED:
   Red_light = ON;
   Amber_light = OFF;
   Green_light = OFF;
   if(++Time_in_state == RED_DURATION)
    {
      Light_state = RED_AMBER;
       Time_in_state = 0;
       }
   break;
  case RED_AMBER:
    Red_light = ON;
    Amber_light = ON;
    Green_light = OFF;
    if (++Time_in_state == RED_AMBER_DURATION)
     {
      Light_state = GREEN;
      Time_in_state = 0;
     }
   break;
case GREEN:
      Red_light = OFF;
      Amber_light = OFF;
      Green_light = ON;
      if (++Time_in_state == GREEN_DURATION)
        {
         Light_state = AMBER;
         Time_in_state = 0;
        }
      break;
case AMBER:
     Red_light = OFF;
      Amber_light = ON;
      Green_light = OFF;
      if (++Time_in_state == AMBER_DURATION)
       {
        Light_state = RED;
        Time_in_state = 0;
       }
       break;
  }
}
```

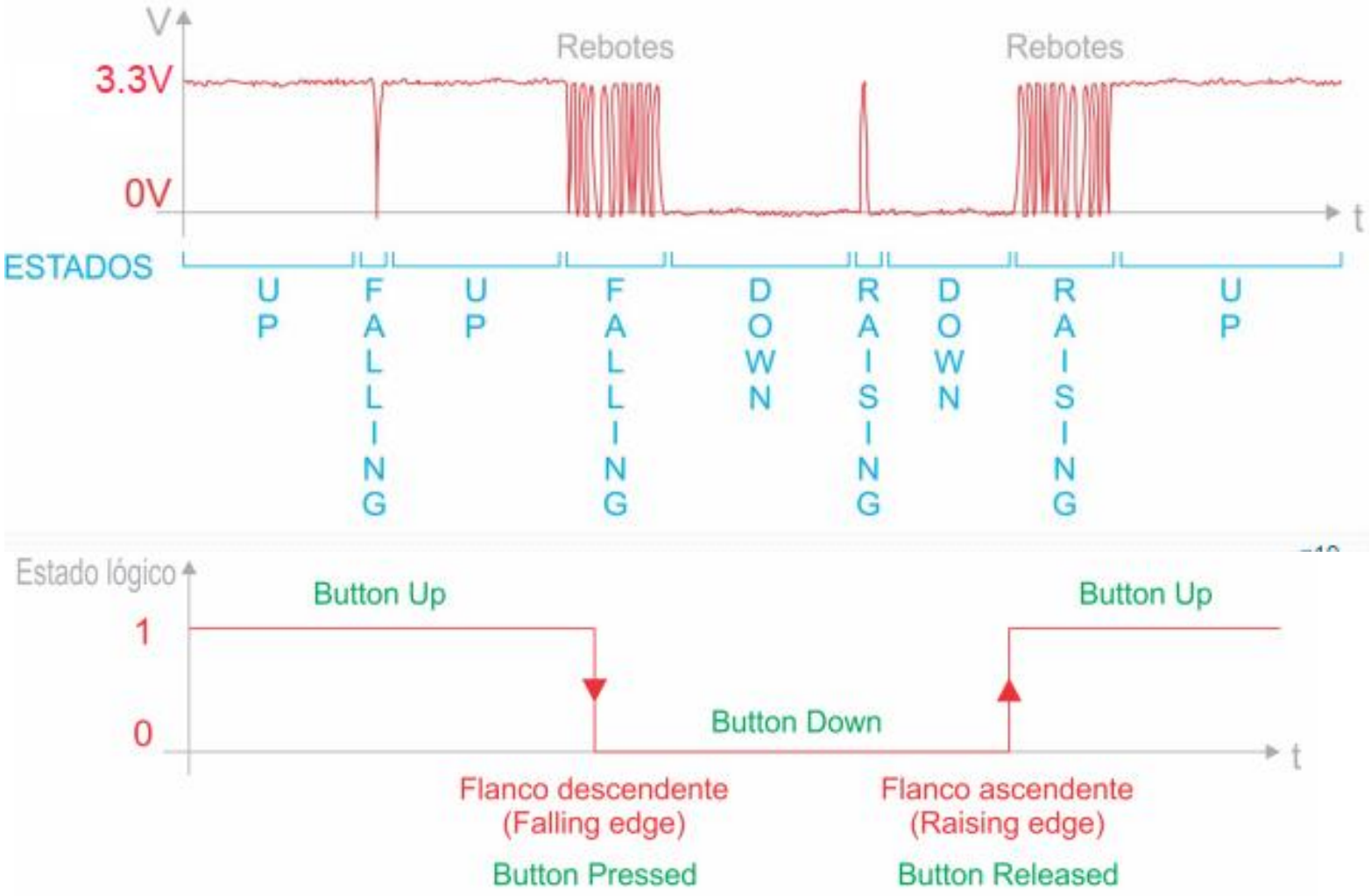
salidas

entrada

Cambio estado

Ejemplo 2

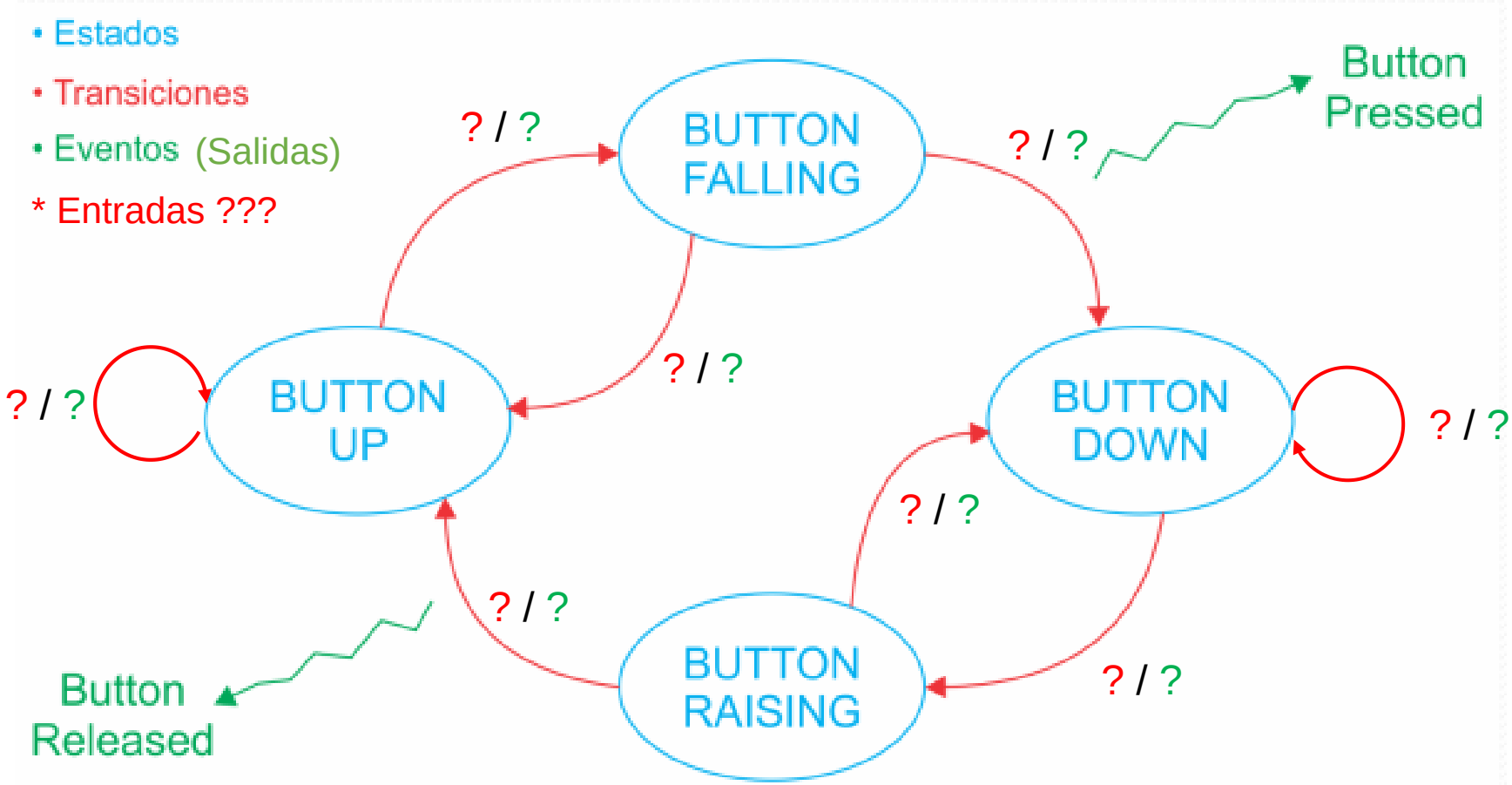
- Detección de pulsador con anti-rebote



Ejemplo 2

- Implementación MEF (Mealy)

Tarea: Completar especificación e implementar



Bibliografía

- *"Real-Time Systems Design and Analysis".* P. Laplante 3rd Ed. John Wiley & Sons 2004.
- *"Embedded Microcomputer System, Real-Time Interfacing".* J. Valvano 2nd Ed. Thomson 2007.
- *"Fundamentos de lógica digital con diseño VHDL -"* 2nd Ed. Brown, Vranesic. 2006.
- *"Patterns for time-triggered Embedded Systems"*, Michael J. Pont. 2014 (pttes_2014.pdf descarga gratis en la web)
- *"Practical UML Statecharts in C/C++, 2nd Edition: Event-Driven Programming for Embedded Systems",* Miro Samek, Newnes 2008, pdf descarga gratis en la web: https://www.state-machine.com/