Implementación de Antirrebote con Máquina de Estado Finito (MEF)

Este método de implementación utiliza una MEF de tipo Mealy para crear un *driver* para captura de eventos de pulsado de teclas mecánicas con antirrebote.

Deseamos que el microcontrolador detecte el estado del botón sin dar falsos positivos.

El comportamiento de la señal de un pulsador al ser pulsado y liberado es el que se muestra en la figura 1

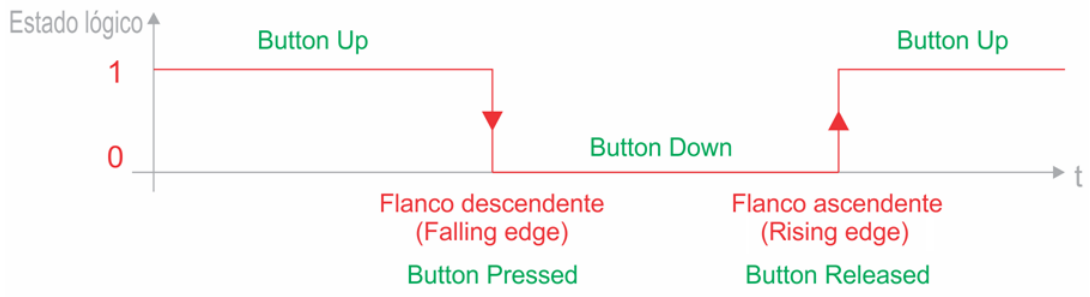

Figura 1

La estrategia que utilizaremos para asegurarnos de que estamos capturando un cambio de estado del pulsador, descartando la posibilidad de que se deba a una falsa detección por rebote mecánico (*debounce*), se puede observar en la figura 2. **Haremos una doble verificación** de estado durante 40 ms para detectar el estado de decrecimiento de la señal (***FALLING***) (flanco descendente), que se produce al hacer la transición del estado Alto (***UP***) al estado Bajo (***DOWN***), y **haremos una doble verificación** de 40 ms para detectar el estado de crecimiento de la señal (***RISING***) (flanco ascendente), que se produce al hacer la transición del estado Bajo (***DOWN***) al estado Alto (***UP***). La detección de cambio de estado se dará por válida si da positiva en ambas verificaciones de 40 ms cada una.

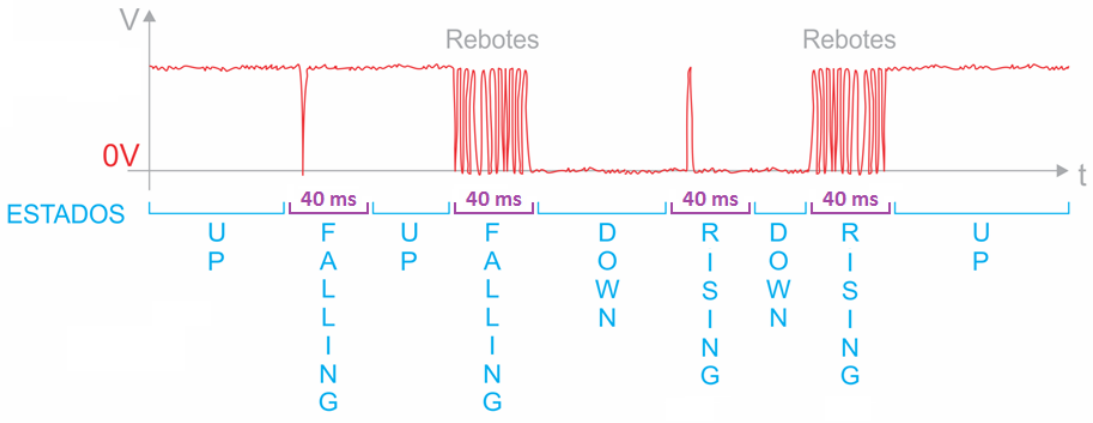

Figura 2

Conformación de Nuestra máquina de estados:

ESTADOS:

BUTTON_UP: Mientras el botón está liberado.
BUTTOTN_FALLING: Mientras está ocurriendo el flanco descendente.
BUTTON_RISING: Mientras está ocurriendo el flanco ascendente.
BUTTON_DOWN: Mientras el botón esta presionado.

EVENTOS:

buttonPressed(): Cuando se detecta estado de botón presionado, como válido.

buttonReleased(): Cuando se detecta estado de botón liberado, como válido.

CONDICIONES DE TRANSICIÓN ENTRE ESTADOS:

1. Si está en estado **BUTTON_UP** y se presiona el pulsador, pasa al estado **BUTTOTN_FALLING**.
2. Si está en estado **BUTTOTN_FALLING**, espera un tiempo de 40 ms y vuelve a leer el pulsador, si realmente estaba presionado pasa al estado **BUTTON_DOWN** y llama a la función (evento) *buttonPressed()*. Si no estaba presionado vuelve al estado **BUTTON_UP**.
3. Si está en estado **BUTTON_DOWN** y liberan el botón pasa al estado **BUTTON_RISING**.
4. Si está en estado **BUTTON_RISING**, espera un tiempo de 40 ms y vuelve a leer el pulsador, si realmente estaba liberado pasa al estado **BUTTON_UP** y llama a la función (evento) *buttonReleased()*. Si no estaba liberado vuelve al estado **BUTTON_DOWN**.

DIAGRAMA DE ESTADOS DE NUESTRA MAQUINA DEBUNCE:

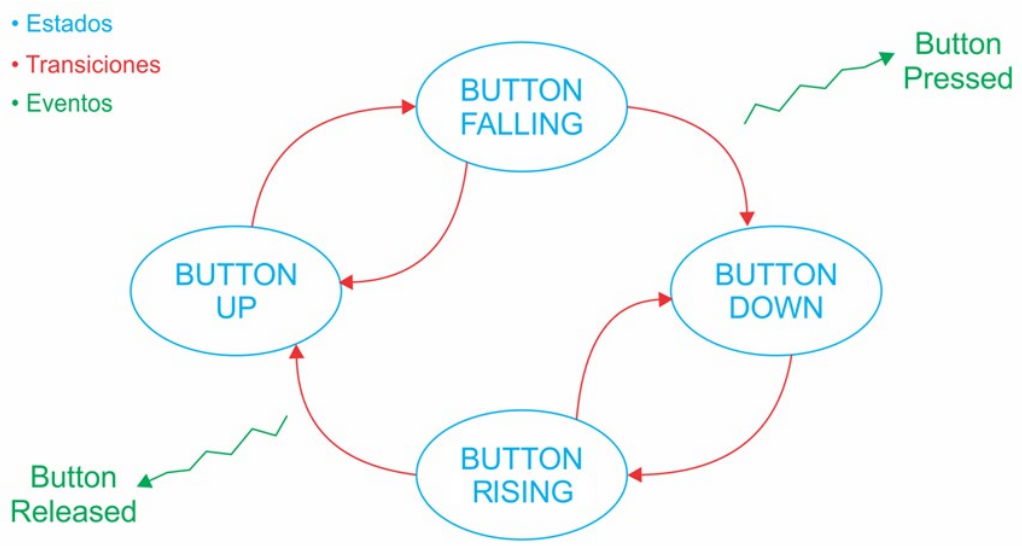

Figura 3

Ejemplo de aplicación de MEF

EJEMPLO DE IMPLEMENTACIÓN DE UN SEMÁFORO

ESTADOS:

1. **Rojo**
2. **Rojo-Amarillo**
3. **Verde**
4. **Amarillo**

Resolvemos mediante la implementación una MEF de Moore:

ENTRADA:

1. **Tiempo transcurrido**

POSIBLES SALIDAS DEL SISTEMA:

1. En estado **Rojo**: encender lámpara roja, apagar lámpara amarilla y verde.
2. En estado **Rojo-Amarillo**: encender lámpara roja y amarilla, apagar lámpara verde.
3. En estado **Verde**: apagar lámpara roja y amarilla, encender lámpara verde.
4. En estado **Amarillo**: apagar lámpara roja, encender lámpara amarilla y apagar lámpara verde.

Al ser una máquina de Moore, la salida depende únicamente del estado actual.

CAMBIOS DE ESTADO:

Dependen únicamente del **estado actual** y del **tiempo transcurrido** (entrada).

DIAGRAMA DE ESTADOS:

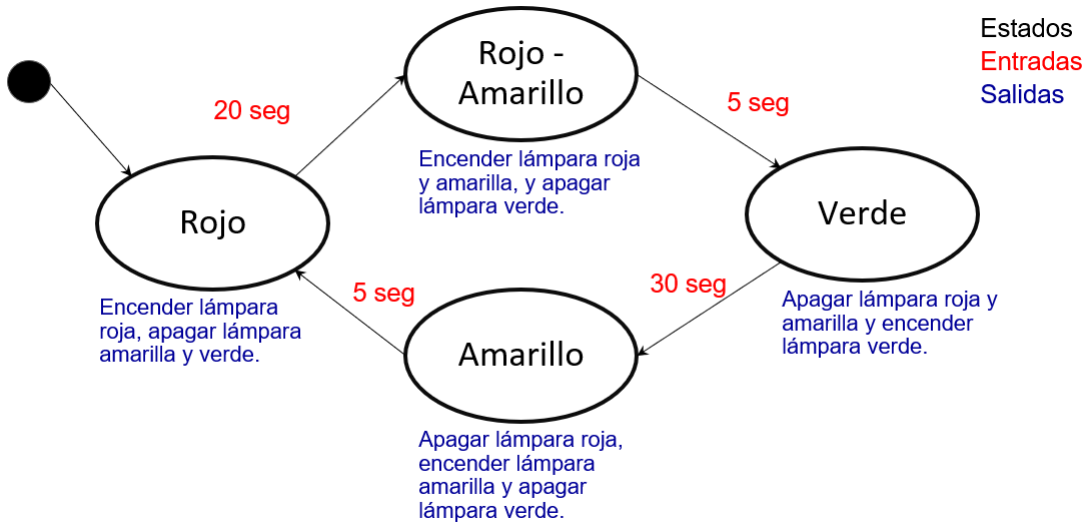

Figura 4

Implementación en C

```
// Nuevo tipo de datos enumerado llamado estadoMEF_t
typedef enum{
   ESTADO_INICIAL,
   ESTADO_1,
   ESTADO_2,
   ESTADO_N
} estadoMEF_t;

// Variable de estado (global)
estadoMEF_t estadoActual;

// Prototipos de funciones
void inicializarMEF(void );
void actualizarMEF(void);
```

```
/// Programa principal
int main (void){
        ...
        inicializarMEF();
        ...
        while(1){
                ...
                actualizarMEF();
                ...
        }
        return 0;
}

// Función Inicializar MEF
void inicializarMEF(void){
        estadoActual = ESTADO_INICIAL;
  // Resto de la inicializaciones
}


// Función Actualizar MEF
void actualizarMEF(void){
        switch (estadoActual) {
                case ESTADO_INICIAL:
                // Actualizar salida del estado
                        gpioWrite(...)
    ...
                        // Chequear condiciones de transición de estado
                        if(condicionesDeTransición == TRUE){
                                // Cambiar a otro estado
                                estadoActual = ESTADO_N;
                        }

                break;
                case ESTADO_1:
                        ...
                break;

                case ESTADO_N:
                                ...

                        break;

                        default:
                                //Si algo modificó la variable estadoActual
                                // a un estado no válido llevo la MEF a un
                                // lugar seguro, por ejemplo, la reinicio:
                                controlDeErrores();
                        inicializarMEF();

                        break;
                }
        }
```