TRABAJO 2 SISTEMAS ELECTRÓNICOS DIGITALES

MICROS

CURSO 2023/24

RODRIGO MARTÍN VARGAS -

ÁLVARO GÓMEZ AGUDO - 55881

MARIO GÓMEZ SÁNCHEZ-CELEMÍN - 55887

GRUPO A404

Índice

1. Introducción

Un microcontrolador es un circuito integrado que contiene todos los componentes de un computador. Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es la que le confiere la denominación de «controlador incrustado» *(embedded controller*). Se dice que es "la solución en un chip" porque su reducido tamaño minimiza el número de componentes y el costo.

El microcontrolador fue inventado en Texas Instruments en la década de 1970, alrededor del mismo tiempo que el primer microprocesador estaba siendo inventado en Intel. Los primeros microcontroladores eran simplemente microprocesadores con una función de memoria, como la memoria RAM y ROM. Más tarde, los microcontroladores se desarrollaron en una amplia gama de dispositivos diseñados para aplicaciones de sistemas embebidos específicos en dispositivos tales como automóviles, teléfonos móviles y electrodomésticos.

Existen en el mercado varias marcas reconocidas por sus características, comercialización, soporte técnico, difusión, usos en la industria etc. Entre ellas tenemos INTEL, MOTOROLA, MICROCHIP, PHILLIPS, NATIONAL y ATMEL.

En nuestro caso utilizaremos el microcontrolador STM32F411E-DISCO de la familia STM32. La cual esta basada en nucleaos ARM Cortex-M de 32 bits. Dentro de la misma familia podemos encontrar modelos que cuentan con un núcleo M0 hasta M7, con distintos periféricos y distintas cantidades de memoria RAM y Flash, con distintas prestaciones y por lo tanto, distintos precios, para utilizar en proyectos Maker o hobby, hasta las aplicaciones profesionales más exigentes

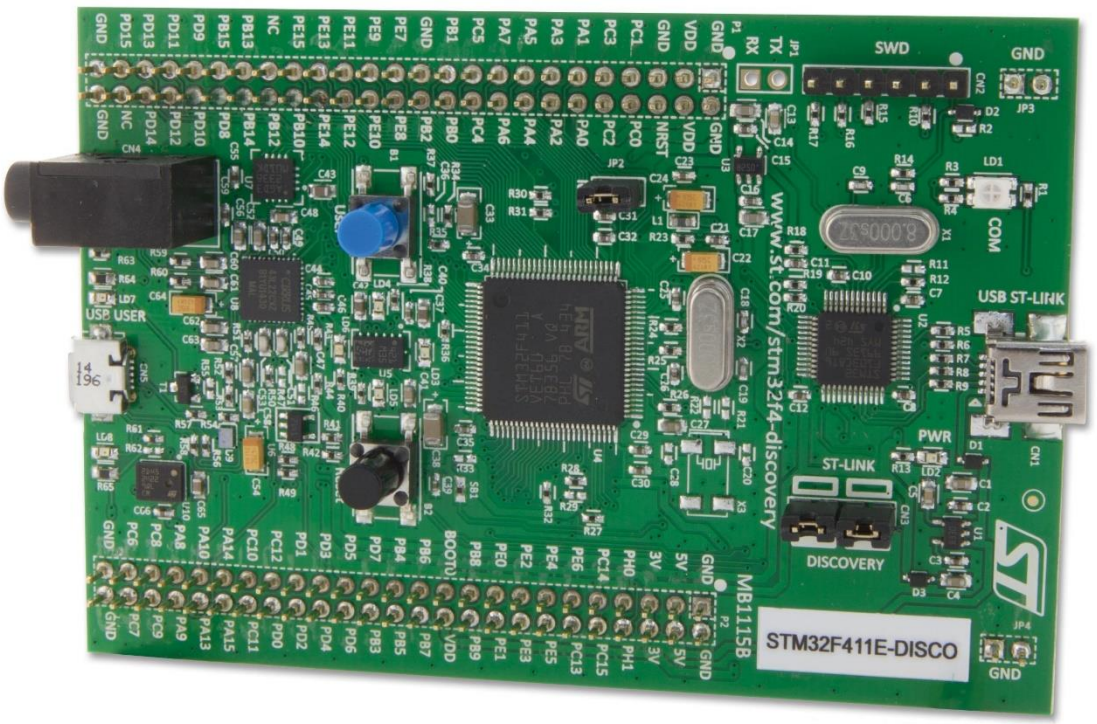


Ilustración 1: Microcontrolador STM32F411E-DISCO

2. Sensores utilizados

LDR: Es un componente electrónico que se utiliza como sensor de luz. También se denominan fotorresistencias. Tienen una resistencia eléctrica muy elevada cuando están en la oscuridad o con poca luz.

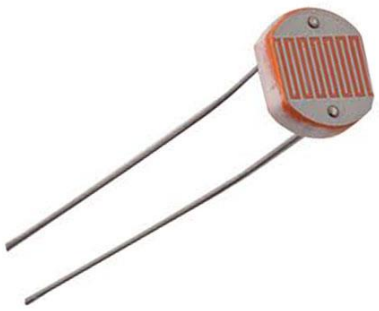


Ilustración 2: LDR

Servomotor: Un servomotor es un motor de corriente continua, pero en vez de conseguir un giro continuo, está diseñado para conseguir que gire un determinado ángulo en respuesta a una señal de control, y que se mantenga fijo en esa posición. Esta señal de control es dada por los pines digitales PWM.

Ilustración 3: Servomotor

Potenciómetro: Un potenciómetro es un componente electrónico similar a los resistores, pero cuyo valor de resistencia en vez de ser fijo es variable, permitiendo controlar la intensidad de corriente a lo largo de un circuito conectándolo en paralelo ó la caida de tensión al conectarlo en serie.

Ilustración 4: Potenciómetro

Botón: El botón de un dispositivo electrónico funciona por lo general como un interruptor eléctrico, es decir en su interior tiene dos contactos, al ser pulsado uno, se activará la función inversa de la que en ese momento este realizando.

Ilustración 5: Botón

Led: Un LED es un diodo emisor de luz, es decir, un tipo particular de diodo que emite luz al ser atravesado por una corriente eléctrica.

Ilustración 6: LEDS

Ultrasonido: el ultrasonido es un sensor utilizado para medir distancias. Su uso se basa en un trigger o gatillo, pin que al accionarse manda un pulso el cual se recibe a través del pin echo y en base a la diferencia de tiempo entre pulsos y la velocidad de sonido se determina la distancia.

Ilustración 8: Ultrasonido

3. Maqueta

Para implementar la aplicación domótica con el microcontrolador hemos realizado una maqueta que simula un representa un garaje y realiza diferentes funciones de manera autónoma.

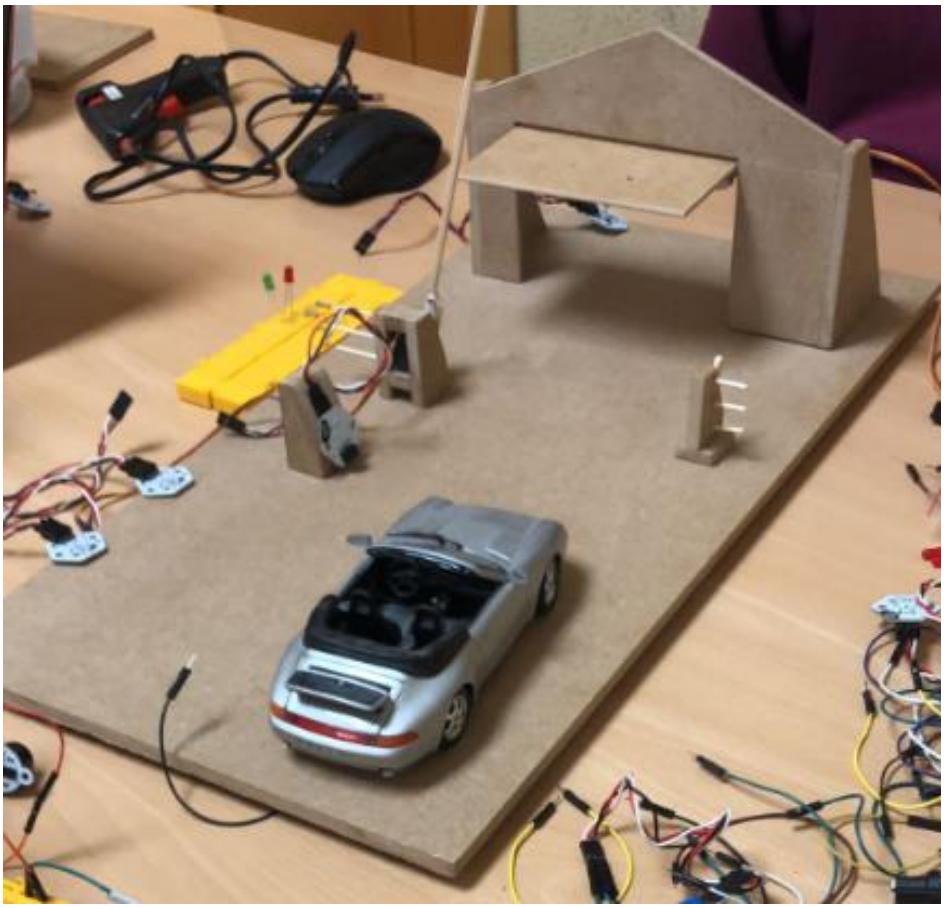


Ilustración 7: Maqueta garaje

En primer lugar, cuando el coche se aproxima a la casa un ultrasonido lo detecta y va iluminando un sistema de luces led que representa la posición en la que se encuentra. A continuación, se encuentra con la barrera cerrada. Esta barrera tiene dos modos de control, los cuales se eligen mediante un pulsador. En el primer caso, se detiene el coche, el sensor de infrarrojos lo detecta y da la orden al servomotor de abrir la barrera. En el segundo, se regula manualmente la acción mediante un potenciómetro.

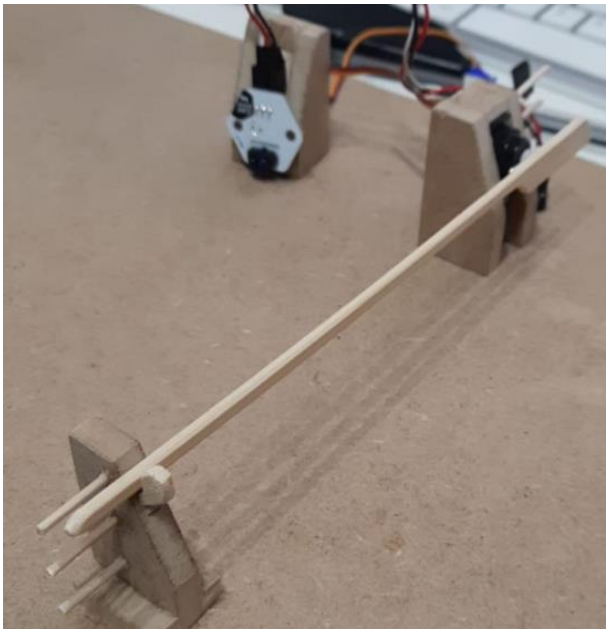


Ilustración 8: Barrera con sensor infrarrojos y servo

Por último, el coche llega a la puerta de la casa y debe introducir una clave en la FPGA. En caso de que sea correcta se iluminara un led verde y se abrirá la puerta mediante otro servomotor.

Ilustración 9: Puerta casa accionada por servomotor

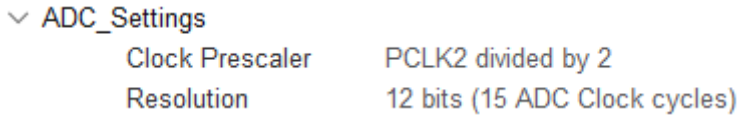
Además, la casa tiene un led que se regula mediante un sensor LDR. Se mantiene encendida y cuando detecta luz natural se apaga.

Ilustración 10: Funcionamiento iluminación de la casa

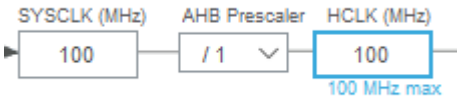
4. Especificaciones

Antes de proceder a la parte de programación es necesario mostrar los ajustes utilizados en algunos de los elementos.

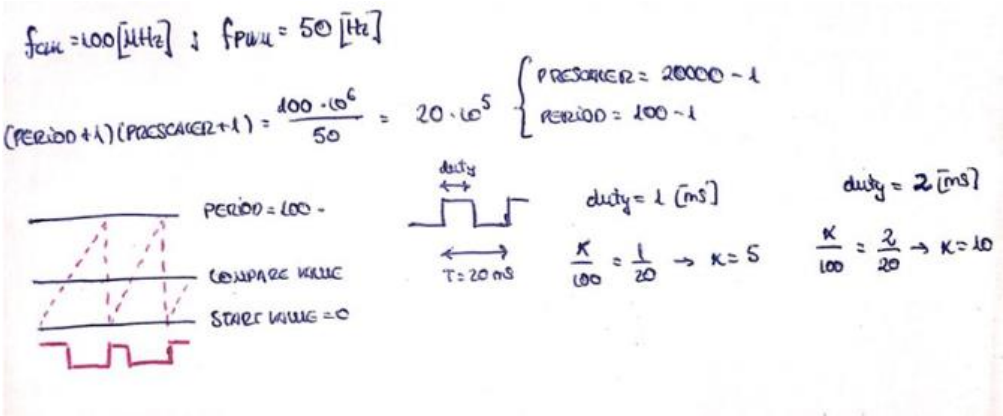
- Se ha utilizado una resolución de 12 bits para las entradas analógicas (LDR y potenciómetro)



- La frecuencia del reloj es de 100 MHz



- El servomotor funciona a 50 Hz y con un ancho de pulso de entre 1 y 2 milisegundos



- El ultrasonido mide la distancia en función del tiempo de rebote del ultrasonido por ello es función de la velocidad de sonido.

$f_{clk} = 100\,[MHz] \;;\; f_{salida} = 1\,[MHz]$

$(PERIODO+1)(PRESCALER+1) = \frac{100 \cdot 10^6}{1 \cdot 10^6} = 100$

- $PERIODO = 0$.
- $PRESCALER = 100 - 1$

$distancia = t_{medición} \cdot vel_{sonido} = \frac{Periodos}{2 \cdot 10^6} \cdot 343 \cdot 100\,[cm/s] = \frac{Periodos}{59}$

Con el número de periodos que tarda en llegar el impulso desde que el trigger se acciona hasta que el echo lo recoge y dividiendo por 2 veces la frecuencia que llega al timer se obtiene el tiempo en segundos de cada medición. La distancia en centímetros se obtiene al multiplicar el tiempo por la velocidad del sonido en centímetro por segundo (34300 cm/s).

5. Programación

En primer lugar se muestran las conexiones utilizadas en la placa mediante el archivo .ioc.

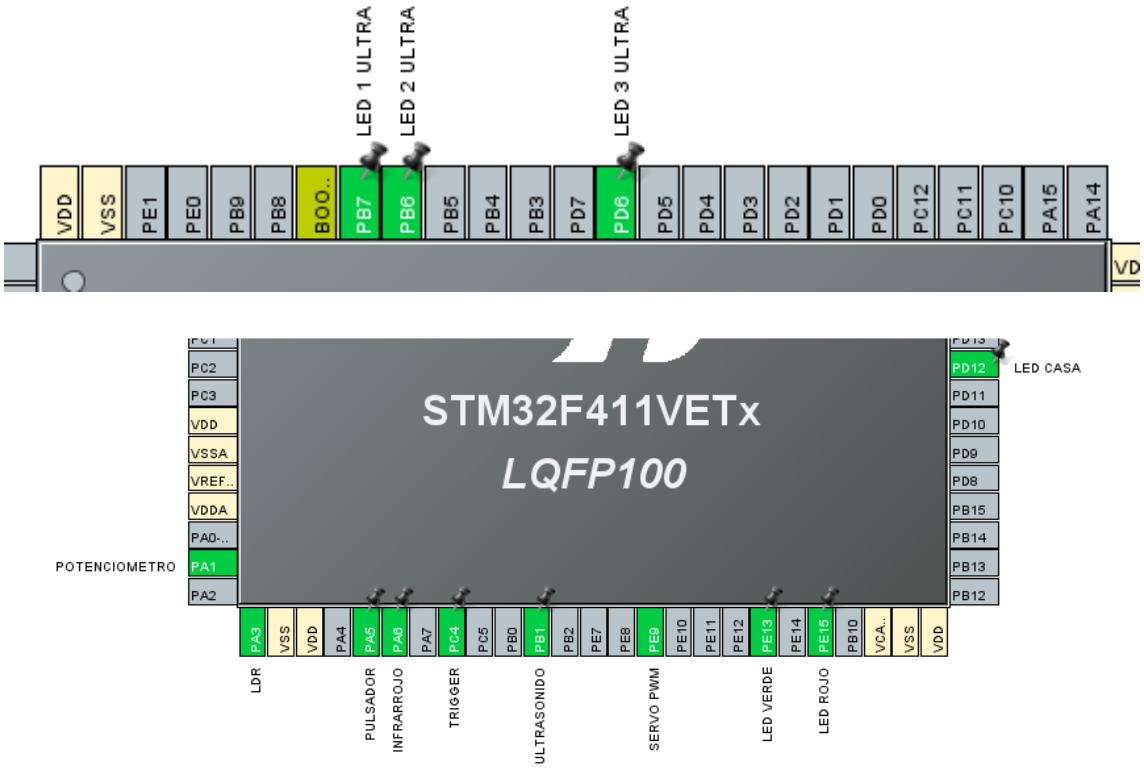


Ilustración 11: Entradas y salidas de la placa

Agrupando las entradas en la tabla 1 y las salidas en la tabla 2:

Tabla 1: Entradas micro

Elemento	Potenciómetro	LDR	Pulsador	Sensor IR	Ultrasonido
Pin	PA1	PA3	PA5	PA6	PC4

Tabla 2: Salidas micro

Elemento	PWM	Led Verde	Led Rojo	Led Casa	Leds carretera	Ultrasonido (TIM)
Pin	PE9	PE13	PE15	PD12	PD6, PB6, PB7	PB1

Una vez que se conocen las conexiones del micro con la parte hardware y las especificaciones de los elementos utilizados se procede a recoger en una lista las funciones utilizadas junto con la descripción de su funcionamiento. Todas ellas se han programado en el fichero main.c.

- **Variables utilizadas**

```
uint32_t valor_adc[2];              // almacena los valores de los sensores analógicos
uint32_t pot_val=0,ldr_val=0;       // dan nombre a los valores de los sensores analógicos
uint32_t duty=0;                    // valor del ancho de pulso del pwm del servomotor
volatile uint8_t modo=0;           // indica el modo de apertura de la barrera (0 - automático, 1 - manual)
volatile uint8_t dma_completo = 0; // finalización de las lecturas dma de los sensores analógicos
volatile uint8_t bajar_barrera=0; // se utiliza como callback para evitar el uso de DELAYS
uint8_t es_pv=0;                 // variable para configuracón de temporizador de ultrasonido
int distancia=0;                 // da valor a la distancia medida por el sensor de ultrasonido
uint32_t valor1,valor2,periodo;  // variable para configuracón de temporizador de ultrasonido
```

- **Servo_manual:** Esta función se encarga del control manual de la apertura de la valla. Para ello recibe el valor del potenciómetro y en función del valor recibido genera un pulso PWM con un ancho de 1 o 2 milisegundos.

```
uint32_t servo_manual(uint32_t value){
    if(value>0 && value<3500){ // cerrado
        HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,1);
        HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,0);
        return 5;
    }
    else if(value>=3500){ // abierto
        HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,0);
        HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,1);
        return 8;
    }
    else return 8;
}
```

- **Servo_automático:** Esta función se encarga del control automático de la apertura de la valla. Para ello se utiliza el sensor de infrarrojos de tal modo que si se recibe un valor lógico alto genera un pulso PWM para mover el servomotor y abrir la barrera.
  Para la lectura del sensor se ha utilizado un código antirrebotes realizando 4 lecturas cada 25 milisegundos, evitando así falsos positivo.
  Para evitar utilizar DELAYS que bloqueen el código se ha utilizado una especia de Callback (bajar_barrera), ejecutando la orden desde el main.

```
void servo_automatico(){

    int contador=0,tiempo=0;

        if((HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_6))&&(bajar_barrera==0)){
            tiempo=HAL_GetTick();
            while(contador<4){
                if((HAL_GetTick()-tiempo)>=25){
                    if(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_6)==0){
                        contador=0;break;
                    }
                    else {
                        tiempo=HAL_GetTick();
                        contador++;
                    }
                }
                if((contador==4)&&(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_6)==1)){
                    __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,5);
                    HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,1);
                    HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,0);
                    bajar_barrera=1;
                }

            }
        }
        contador=0;
}
```

- **Interrupciones para el pulsador:** Para la lectura del pulsador se han utilizado interrupciones (callbacks), cambiando del modo automático (modo=0) a modo manual (modo=1) con cada pulsación detectada.

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin){
    if(GPIO_Pin==GPIO_PIN_5){
        if(modo==0)modo=1;
        else if(modo==1)modo=0;
    }
}
```

- **Encender_led:** Esta función se encarga de encender la luz de la casa en función de la luminosidad detectada por el LDR, devolviendo un valor lógico alto o bajo.

```
int encender_led(uint32_t value){
    if(value<3000)return 1;
    else return 0;
}
```

- **Medición ultrasonidos:** En primer lugar para la medición del ultrasonidos se necesita configurar un temporizador para recibir el tiempo de captura, y determinar el pin del trigger y activar la interrupción del timer. En la función de la interrupción se captura el momento en el que se lanza el ultrasonido y se cambia la polaridad del impulso a recoger para cuando el sonido vuelva. Cuando se recoge la llegada del ultrasonido por flanco de bajada la diferencia de periodos se almacena en la variable periodo y se vuelve a cambiar la polaridad de la medición para la siguiente medida.

```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim){
        if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_4){
                if(es_pv==0)
                {
                        valor1=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_4);
                        es_pv=1;
                        __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_4,TIM_INPUTCHANNELPOLARITY_FALLING);

                }
                else if(es_pv==1){
                                valor2=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_4);
                                __HAL_TIM_SET_COUNTER(htim,0);
                                if(valor2>valor1){
                                        periodo=valor2-valor1;
                                }
                                es_pv=0;
                                __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_4,TIM_INPUTCHANNELPOLARITY_RISING);
                }
        }

}
```

Con la diferencia de periodos obtenida entre la salida y la llegada se puede obtener la distancia como se ha explicado anteriormente dividiendo esta por 58.

```
distancia=periodo/58;
```

Para accionar el trigger se utiliza la siguiente sentencia en el while del main:

```
HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,GPIO_PIN_SET);
HAL_Delay(10);
HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,GPIO_PIN_RESET);
```

```
void encender_leds_carretera(int distancia){
     if((distancia>=0)&&(distancia<=7)){
                HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,GPIO_PIN_SET);
                HAL_GPIO_WritePin(GPIOB,GPIO_PIN_6,GPIO_PIN_RESET);
                HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,GPIO_PIN_RESET);
          }

          if((distancia>=8)&&(distancia<=15)){
                  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,GPIO_PIN_RESET);
                  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_6,GPIO_PIN_SET);
                  HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,GPIO_PIN_RESET);
          }

          if((distancia>=16)&&(distancia<=24)){
                  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,GPIO_PIN_RESET);
                  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_6,GPIO_PIN_RESET);
                  HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,GPIO_PIN_SET);
          }

          if(distancia>24){
              HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,GPIO_PIN_RESET);
              HAL_GPIO_WritePin(GPIOB,GPIO_PIN_6,GPIO_PIN_RESET);
              HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,GPIO_PIN_RESET);
          }
}
```

La utilización de todas estas funciones permite que el bucle principal sea bastante claro y legible.

```
int tiempo_barrera=0;
int aux=0;
while (1)
{
   pot_val=valor_adc[0];
   ldr_val=valor_adc[1];

    if(modo==1) __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,servo_manual(pot_val));
    else servo_automatico();

    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,encender_led(ldr_val));

    // ENVIO DE SEÑAL DEL TRIGGER
    HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,GPIO_PIN_SET);
    HAL_Delay(10);
    HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,GPIO_PIN_RESET);

    HAL_TIM_IC_Start_IT(&htim3,TIM_CHANNEL_4);

    if((bajar_barrera==1)&&(aux==0)){
        tiempo_barrera=HAL_GetTick();
        aux=1;
    }

    if(HAL_GetTick()-tiempo_barrera>3000){
       __HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1,8); // cerrado
       HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,0);
       HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,1);
       bajar_barrera=0;
       aux=0;
    }

    distancia=periodo/58;
    encender_leds_carretera(distancia);
```