Proyecto 2: Manejo de dimmer con PWM

Alumnos:
- Garré, Sebastián Nehuen
- Mosler, Santiago Agustín
- Mozo, Demian Nehuel
- Trigo, Nicolás Daniel

Sistemas Digitales - 06 / 06 / 2025

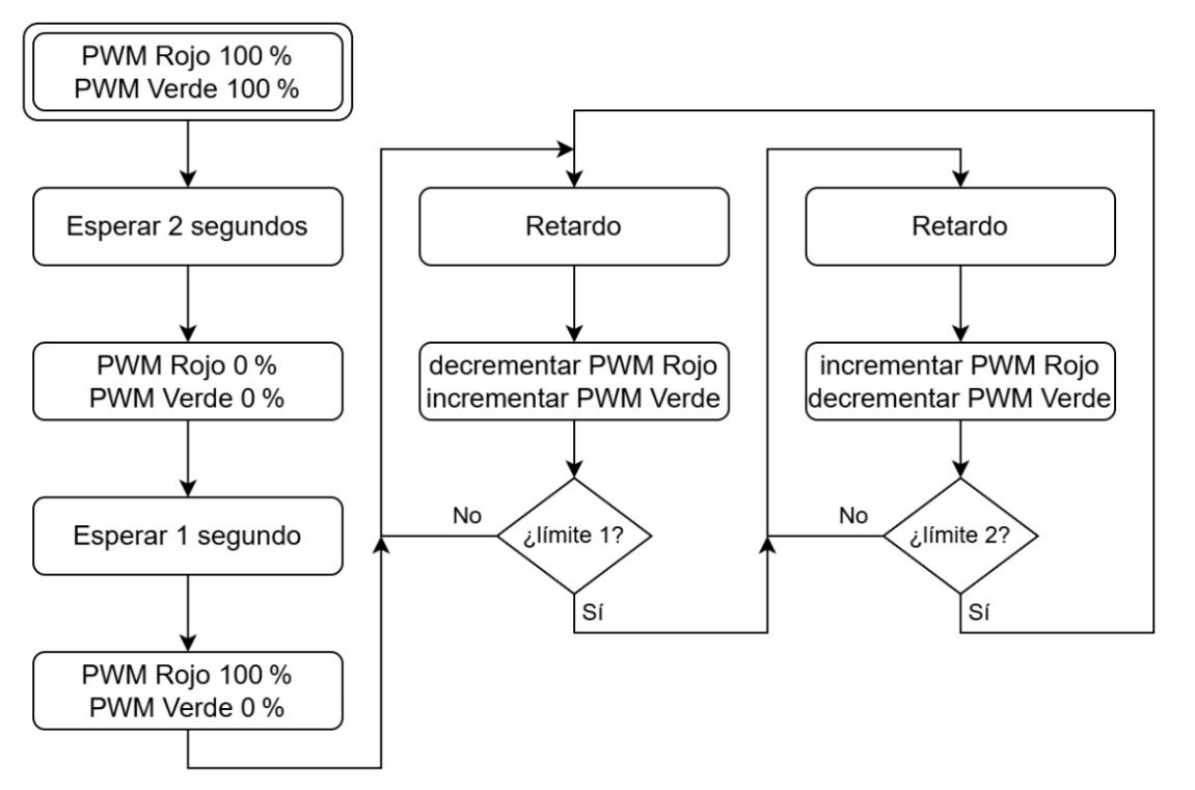
Introducción

El objetivo es **aprender PWM (modulación por ancho de pulso)** y **manejo de retardos temporales** mediante un ejemplo práctico. Se requiere configurar jumpers, pines y programar el comportamiento deseado.

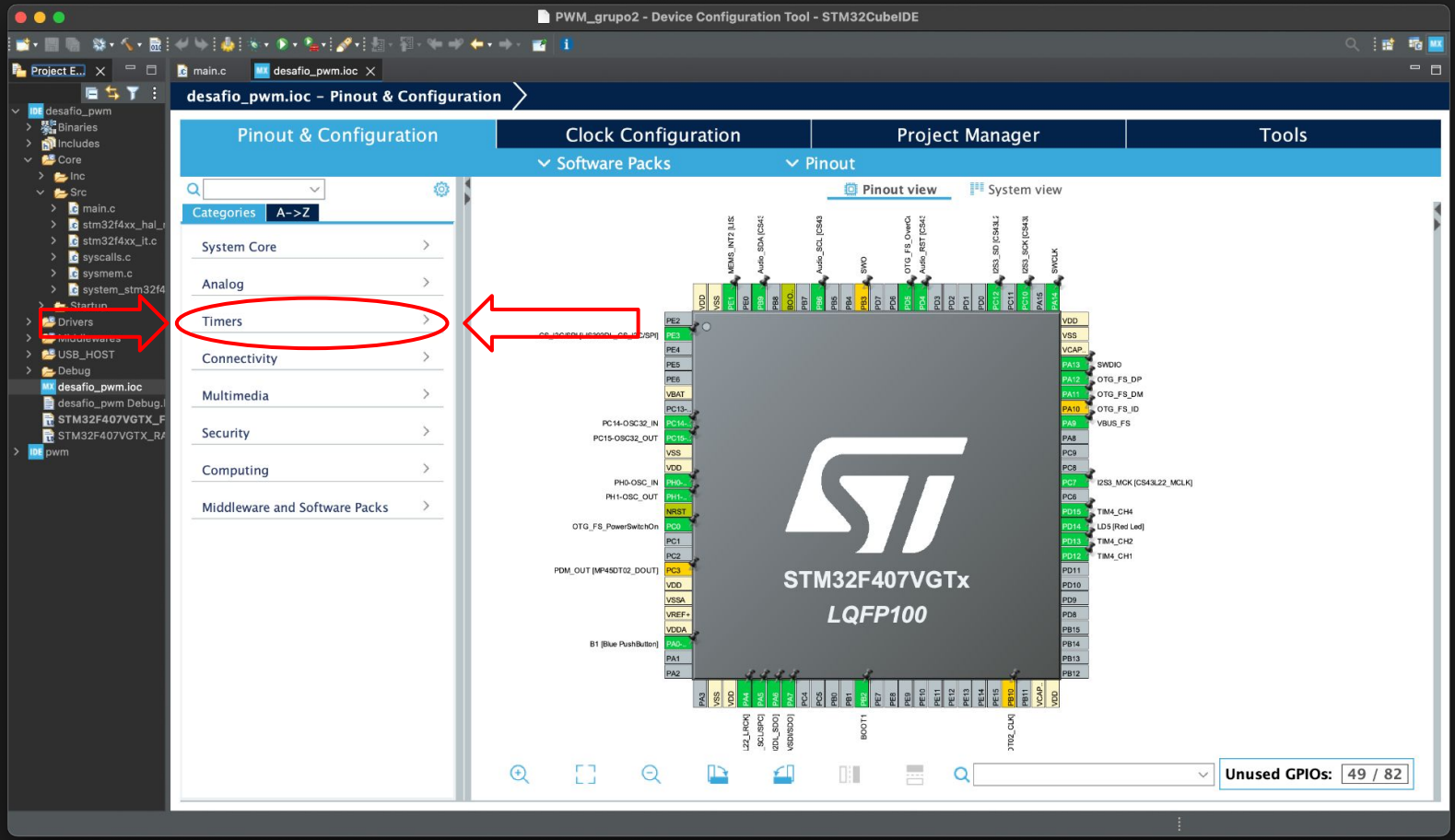
Funcionamiento:

1) LEDs rojo y verde al 100% (2 seg).
2) Ambos apagados (1 seg).
3) LED rojo al 100%, verde apagado.
4) Transición:
   Rojo baja intensidad progresivamente .
   Verde sube intensidad progresivamente .
5) Se invierte el proceso.
6) El ciclo se repite indefinidamente.

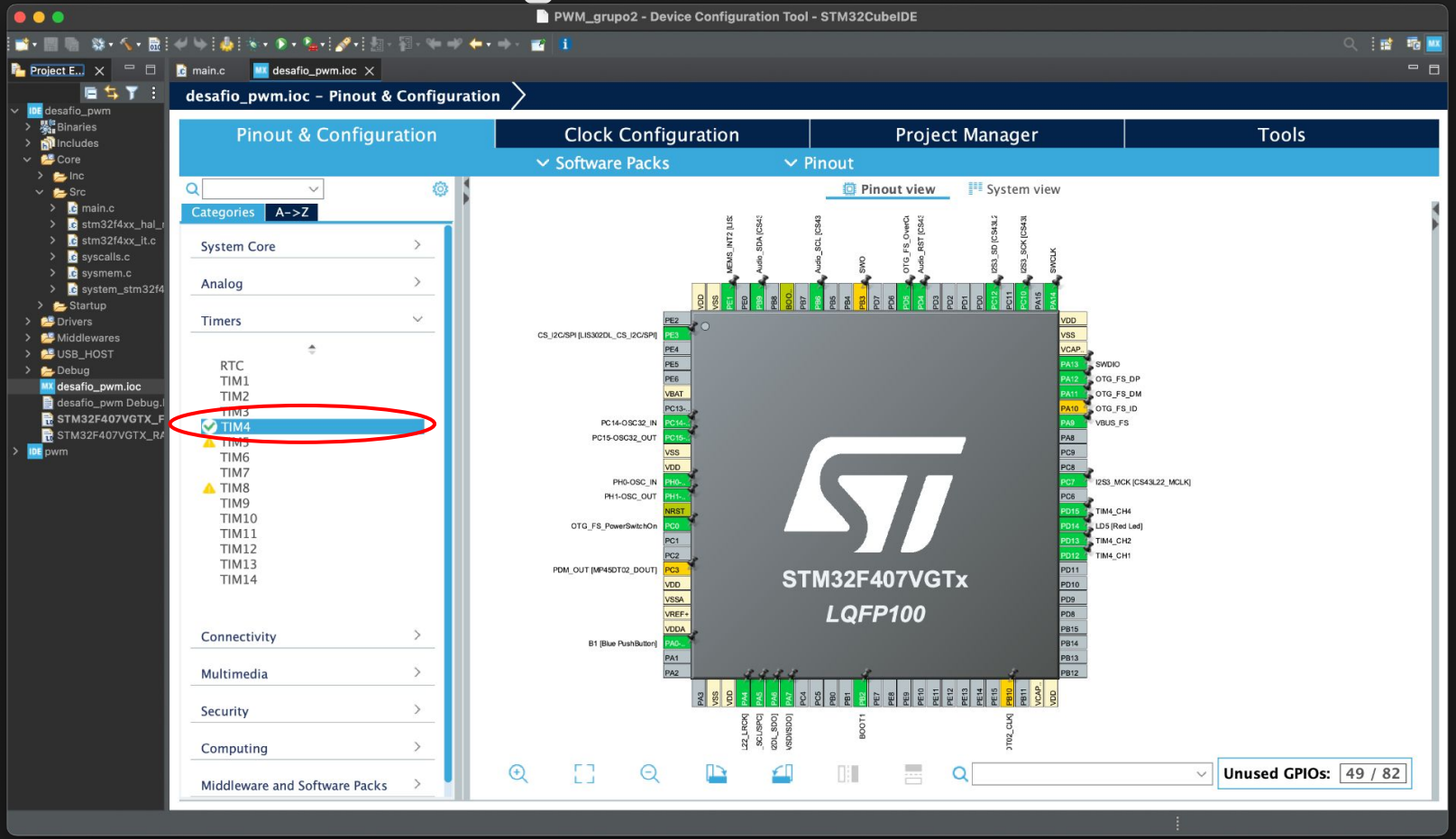
Diagrama de flujo



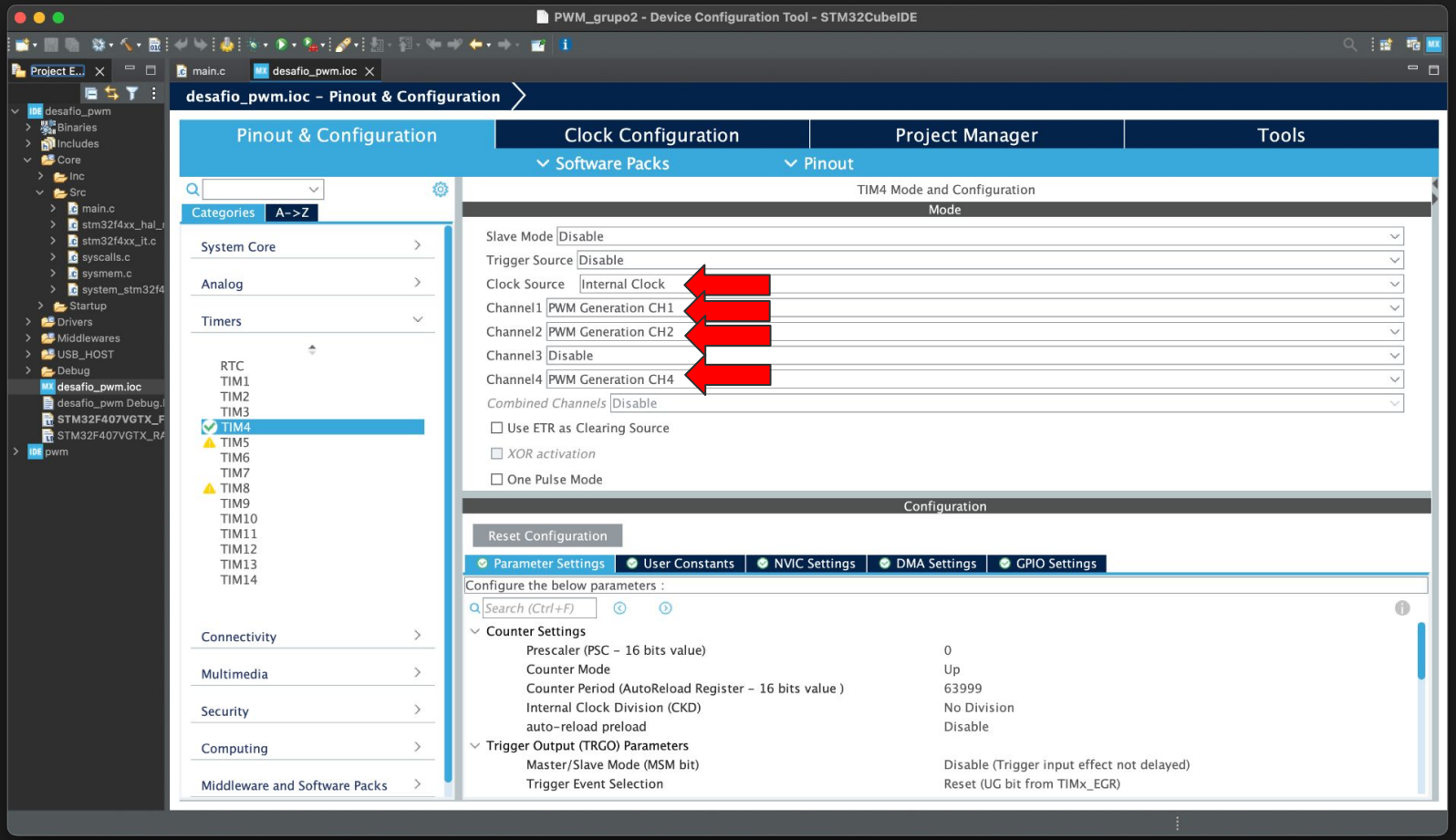
Configuración del timer



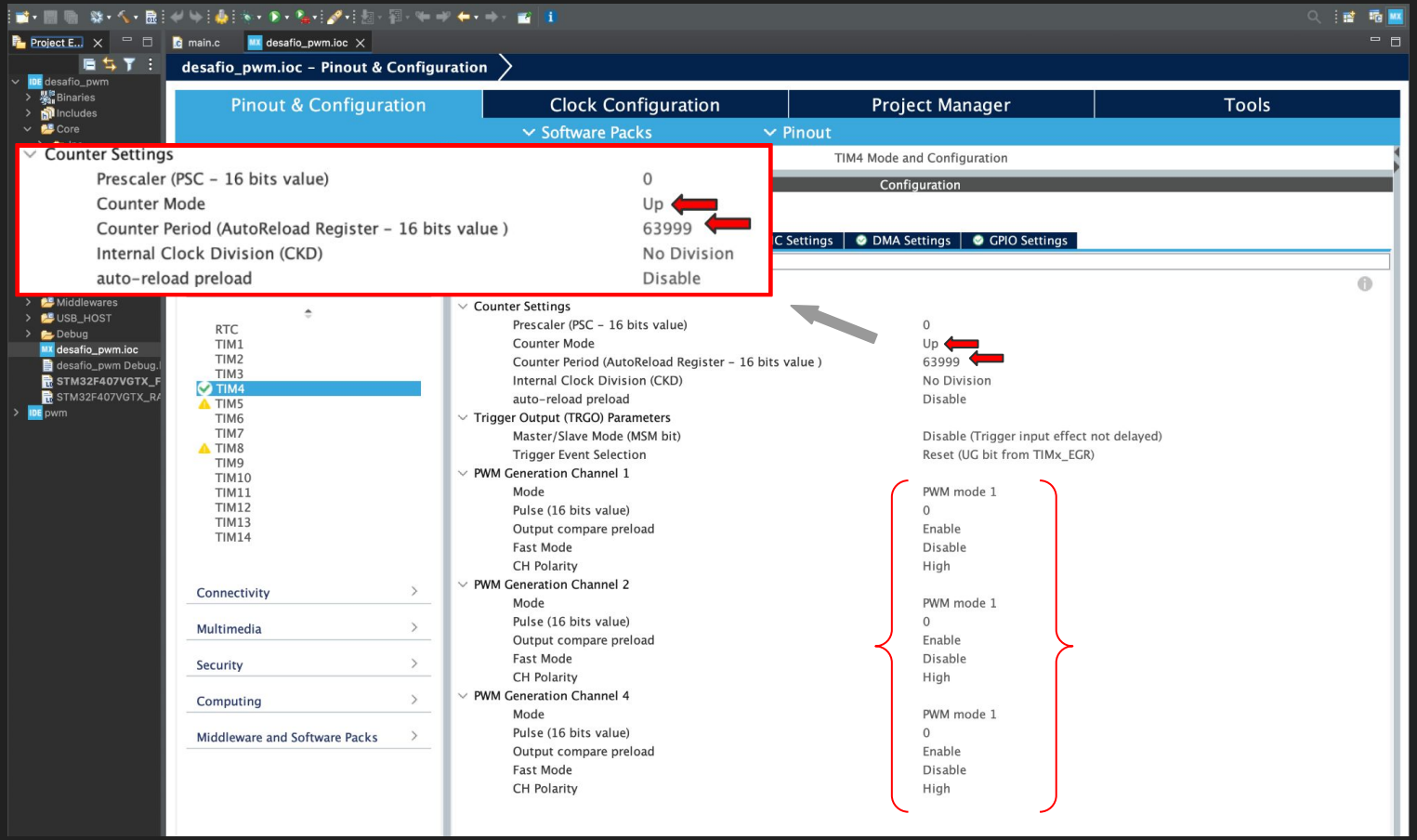
Configuración del timer



Configuración del timer



Configuración del timer



Definiciones en código (en `main.c`)

```
/* Private define ------------------------------------------------------------*/

#define TENCENDIDO 2000 // Tiempo en ms luces encendidas al inicio
#define TAPAGADO   1000 // Tiempo en ms luces apagadas antes de transición
#define TCAMBIO      20 // Retardo entre cambios de PWM
#define VALCAMBIO   640 // Paso de cambio de ciclo de trabajo (1% de 64000)
```

Loop principal

```
/* USER CODE BEGIN 2 */
  // CONDICIONES INICIALES
  inicio();
  /* USER CODE END 2 */

  /* Infinite loop */
  while (1) {
    /* USER CODE BEGIN 3 */
    transicion1(); //  gradualmente disminuye rojo, aumenta verde
    transicion2(); //  gradualmente disminuye verde, aumenta rojo
  }
  /* USER CODE END 3 */
}
```

Función inicio

```
void inicio(void) {
  // Obtenemos el valor máximo de ciclo de trabajo de la configuración del hw
  uint16_t ciclomaximo = TIM4->ARR; // Valor máximo de ciclo de trabajo
  // Configuramos el PWM para que inicie con los LED encendidos
  TIM4->CCR1 = ciclomaximo; // LED verde al 100 %
  TIM4->CCR2 = ciclomaximo; // LED rojo al 100 %
  // Arrancamos el PWM
  HAL_TIM_PWM_Start(&htim4,TIM_CHANNEL_1); // Inicio de la modulación PWM, LED verde
  HAL_TIM_PWM_Start(&htim4,TIM_CHANNEL_2); // Inicio de la modulación PWM, LED rojo
  // Mantenemos los LED encendidos por un tiempo
  HAL_Delay(TENCENDIDO); // Retardo de TENCENDIDO milisegundos
  // Apagamos los LED
  TIM4->CCR1 = 0; // LED verde al 0 %
  TIM4->CCR2 = 0; // LED rojo al 0 %
  // Mantenemos los LED apagados por un tiempo
  HAL_Delay(TAPAGADO); // Retardo de TAPAGADO milisegundos
  // Encendemos el LED rojo
  TIM4->CCR2 = ciclomaximo; // LED rojo al 100 %
}
```

Ejemplo función de transición

```
/*-------------  transición 1 -------------*/
// Disminuye rojo, aumenta verde
void transicion1(void) {
  // Obtenemos valores de la configuración del hw
  uint16_t ciclomaximo = TIM4->ARR; // Valor máximo de ciclo de trabajo
  uint16_t cicloverde = TIM4->CCR1; // Valor actual de ciclo de trabajo en LED verde
  uint16_t ciclorojo = TIM4->CCR2; // Valor actual de ciclo de trabajo en LED rojo
  do {
    // Esperamos un tiempo antes de cambiar las intencidades de lso LED
    HAL_Delay(TCAMBIO); // Espera por TCAMBIO milisegndos
    // Disminuimos un poco la intensidad del LED rojo
    ciclorojo = (ciclorojo > VALCAMBIO) ? ciclorojo - VALCAMBIO
                                        : 0; // Decremento con saturación en 0 %
    TIM4->CCR2 = ciclorojo; // Actualiza el ciclo de trabajo en LED rojo
    // Aumentaqmos un poco la intensidad del LED verde
    cicloverde = (ciclomaximo - cicloverde > VALCAMBIO) ?
                                                        cicloverde + VALCAMBIO
                    : ciclomaximo; // Incremento con saturación en 100 %
    TIM4->CCR1 = cicloverde; // Actualiza el ciclo de trabajo en LED verde
    // Verficamos si repetimos o invertimos
  } while (ciclorojo > 0 && cicloverde < ciclomaximo); // Repetimos mientras no se alcance un límite
}
```

Desafío

1) Inicia con LED amarillo al 100%
2) Aumenta rojo a 100% progresivamente
3) Disminuye amarillo a 0% progresivamente
4) Aumenta verde a 100% progresivamente
5) Disminuye rojo a 0% progresivamente
6) Aumenta amarillo a 100% progresivamente
7) Disminuye verde a 0% progresivamente
8) Repetir indefinidamente pasos 2–7.

Loop principal

```
  /* USER CODE BEGIN 2 */
   // CONDICIONES INICIALES
   inicio(); // (amarillo al 100%)
   /* USER CODE END 2 */

   /* Infinite loop */
   while (1) {
     /* USER CODE BEGIN 3 */
     transicion1(); // Aumenta rojo a 100%
     transicion2(); // Disminuye amarillo a 0%
     transicion3(); // Aumenta verde a 100%
     transicion4(); // Disminuye rojo a 0%
     transicion5(); // Aumenta amarillo a 100%
     transicion6(); // Disminuye verde a 0%
   }
   /* USER CODE END 3 */
}
```

Función `inicio()`

```c
/*------------- Inicio -------------*/
void inicio(void) {
  // Obtenemos el valor máximo de ciclo de trabajo
  uint16_t ciclomaximo = TIM4->ARR;

  // Iniciamos con LED amarillo al 100%
  TIM4->CCR4 = ciclomaximo; // LED amarillo al 100%

  // Arrancamos los PWM
  HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1); // Verde
  HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_2); // Rojo
  HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_4); // Amarillo
}
```

Ejemplo función de transición

```c
/*------------- Transición 1: Aumenta rojo a 100% -------------*/
void transicion1(void) {
  uint16_t ciclomaximo = TIM4->ARR;
  uint16_t ciclorojo = TIM4->CCR2;

  do {
    HAL_Delay(TCAMBIO);
    ciclorojo = (ciclomaximo - ciclorojo > VALCAMBIO) ? ciclorojo + VALCAMBIO
                                                      : ciclomaximo;
    TIM4->CCR2 = ciclorojo;
  } while (ciclorojo < ciclomaximo);
}
```