```c
/***********************************************************************
 *
 * Decimal counter with 7-segment output.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Copyright (c) 2018-2020 Tomas Fryza
 * Dept. of Radio Electronics, Brno University of Technology, Czechia
 * This work is licensed under the terms of the MIT license.
 *
 **********************************************************************/

/* Includes ----------------------------------------------------------*/
#include <avr/io.h>         // AVR device-specific IO definitions
#include <avr/interrupt.h>  // Interrupts standard C library for AVR-GCC
#include "timer.h"          // Timer library for AVR-GCC
#include "segment.h"        // Seven-segment display library for AVR-GCC

uint8_t singles = 0;
uint8_t decimals = 0;


/* Function definitions ----------------------------------------------*/
/**
 * Main function where the program execution begins. Display decimal
 * counter values on SSD (Seven-segment display) when 16-bit
 * Timer/Counter1 overflows.
 */
int main(void)
{
    // Configure SSD signals
    SEG_init();



    // Test of SSD: display number '3' at position 0
    SEG_update_shift_regs(3, 0);

    //SEG_clear();


    /* Configure 8-bit Timer/Counter0
     * Set prescaler and enable overflow interrupt */
    TIM0_overflow_4ms();
    TIM0_overflow_interrupt_enable();

    /* Configure 16-bit Timer/Counter1
     * Set prescaler and enable overflow interrupt */
    TIM1_overflow_262ms();
    TIM1_overflow_interrupt_enable();

    // Enables interrupts by setting the global interrupt mask
    sei();

```

```c
    // Infinite loop
    while (1)
    {
        /* Empty loop. All subsequent operations are performed exclusively
         * inside interrupt service routines ISRs */
    }

    // Will never reach this
    return 0;
}

/* Interrupt service routines ------------------------------------*/
/**
 * ISR starts when Timer/Counter0 overflows. Display value on SSD.
 */
ISR(TIMER0_OVF_vect)
{
    static uint8_t position = 0;
    if (position == 0)
        SEG_update_shift_regs(singles, 0); //first position
    else
        SEG_update_shift_regs(decimals, 1); //second position
    position = !position; //change position (0 1)
}

/**
 * ISR starts when Timer/Counter1 overflows. Increment decimal counter.
 */
ISR(TIMER1_OVF_vect)
{
    // number AB = 0-5 0-9
    // LEd counter 0-59
    singles++;
    if(singles > 9)
    {
        singles = 0;
        decimals++;
        if(decimals > 5)
            decimals = 0;
    }
}
```