```c
/***********************************************************************
 *
 * Seven-segment display library for AVR-GCC.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Copyright (c) 2019-2020 Tomas Fryza
 * Dept. of Radio Electronics, Brno University of Technology, Czechia
 * This work is licensed under the terms of the MIT license.
 *
 **********************************************************************/

/* Includes ----------------------------------------------------------*/
#define F_CPU 16000000
#include <util/delay.h>
#include "gpio.h"
#include "segment.h"

/* Variables ---------------------------------------------------------*/
// Active-low digit 0 to 9
uint8_t segment_value[] = {
    // abcdefgDP
    0b00000011,     // Digit 0
    0b10011111,     // Digit 1
    0b00100101,     // Digit 2
    0b00001101,     // Digit 3
    0b10011001,     // Digit 4
    0b01001001,     // Digit 5
    0b01000001,     // Digit 6
    0b00011111,     // Digit 7
    0b00000001,     // Digit 8
    0b00001001      // Digit 9
};

// Active-high position 0 to 3
uint8_t segment_position[] = {
    // p3p2p1p0....
    0b00010000,     // Position 0
    0b00100000,     // Position 1
    0b01000000,     // Position 2
    0b10000000      // Position 3
};


/* Function definitions ----------------------------------------------*/
void SEG_init(void)
{
    /* Configuration of SSD signals */
    GPIO_config_output(&DDRD, SEGMENT_LATCH);
    GPIO_config_output(&DDRD, SEGMENT_CLK);
    GPIO_config_output(&DDRB, SEGMENT_DATA);
}

/*--------------------------------------------------------------------*/
```

```
54  void SEG_update_shift_regs(uint8_t segments, uint8_t position)
55  {
56      uint8_t bit_number;
57      segments = segment_value[segments];     // 0, 1, ..., 9
58      position = segment_position[position];  // 0, 1, 2, 3
59
60      // Pull LATCH, CLK, and DATA low
61      GPIO_write_low(&PORTD, SEGMENT_LATCH);
62      GPIO_write_low(&PORTD, SEGMENT_CLK);
63      GPIO_write_low(&PORTB, SEGMENT_DATA);
64
65      // Wait 1 us
66      _delay_us(1);
67
68      // Loop through the 1st byte (segments)
69      // a b c d e f g DP (active low values)
70      for (bit_number = 0; bit_number < 8; bit_number++)
71      {
72          // Output DATA value (bit 0 of "segments")
73          if ((segments & 1) == 0)
74              GPIO_write_low(&PORTB, SEGMENT_DATA);
75          else
76              GPIO_write_high(&PORTB, SEGMENT_DATA);
77
78          // Wait 1 us
79          _delay_us(1);
80
81          // Pull CLK high
82          GPIO_write_high(&PORTD, SEGMENT_CLK);
83
84          // Wait 1 us
85          _delay_us(1);
86
87          // Pull CLK low
88          GPIO_write_low(&PORTD, SEGMENT_CLK);
89
90          // Shift "segments"
91          segments = segments >> 1;
92      }
93
94      // Loop through the 2nd byte (position)
95      // p3 p2 p1 p0 . . . . (active high values)
96      for (bit_number = 0; bit_number < 8; bit_number++)
97      {
98          // Output DATA value (bit 0 of "position")
99          if ((position % 2) == 0)
100             GPIO_write_low(&PORTB, SEGMENT_DATA);
101         else
102             GPIO_write_high(&PORTB, SEGMENT_DATA);
103
104        // Wait 1 us
105        _delay_us(1);
106
```

```
107         // Pull CLK high
108         GPIO_write_high(&PORTD, SEGMENT_CLK);
109 
110         // Wait 1 us
111         _delay_us(1);
112 
113         // Pull CLK low
114         GPIO_write_low(&PORTD, SEGMENT_CLK);
115 
116          // Shift "position"
117          position = position >> 1;
118     }
119 
120     // Pull LATCH high
121     GPIO_write_high(&PORTD, SEGMENT_LATCH);
122 
123     // Wait 1 us
124     _delay_us(1);
125 }
126 
127 /*-------------------------------------------------------------------*/
128 /* SEG_clear */
129 void SEG_clear(void)
130 {
131     uint8_t bit_number, segments = 0b11111111, position = 0;
132 
133     // Pull LATCH, CLK, and DATA low
134     GPIO_write_low(&PORTD, SEGMENT_LATCH);
135     GPIO_write_low(&PORTD, SEGMENT_CLK);
136     GPIO_write_low(&PORTB, SEGMENT_DATA);
137 
138     // Wait 1 us
139     _delay_us(1);
140 
141     // Loop through the 1st byte (segments)
142     // a b c d e f g DP (active low values)
143     for (bit_number = 0; bit_number < 8; bit_number++)
144     {
145         // Output DATA value (bit 0 of "segments")
146         if ((segments & 1) == 0)
147             GPIO_write_low(&PORTB, SEGMENT_DATA);
148         else
149             GPIO_write_high(&PORTB, SEGMENT_DATA);
150 
151         // Wait 1 us
152         _delay_us(1);
153 
154         // Pull CLK high
155         GPIO_write_high(&PORTD, SEGMENT_CLK);
156 
157         // Wait 1 us
158         _delay_us(1);
159 
```

```
160         // Pull CLK low
161         GPIO_write_low(&PORTD, SEGMENT_CLK);
162 
163         // Shift "segments"
164         segments = segments >> 1;
165     }
166 
167     // Loop through the 2nd byte (position)
168     // p3 p2 p1 p0 . . . . (active high values)
169     for (bit_number = 0; bit_number < 8; bit_number++)
170     {
171         // Output DATA value (bit 0 of "position")
172         if ((position % 2) == 0)
173             GPIO_write_low(&PORTB, SEGMENT_DATA);
174         else
175             GPIO_write_high(&PORTB, SEGMENT_DATA);
176 
177         // Wait 1 us
178         _delay_us(1);
179 
180         // Pull CLK high
181         GPIO_write_high(&PORTD, SEGMENT_CLK);
182 
183         // Wait 1 us
184         _delay_us(1);
185 
186         // Pull CLK low
187         GPIO_write_low(&PORTD, SEGMENT_CLK);
188 
189         // Shift "position"
190         position = position >> 1;
191     }
192 
193     // Pull LATCH high
194     GPIO_write_high(&PORTD, SEGMENT_LATCH);
195 
196     // Wait 1 us
197     _delay_us(1);
198 
199 }
200 
201 /*-------------------------------------------------------------------*/
202 /* SEG_clk_2us */
203 void SEG_clk_2us(void)
204 {
205     // Wait 1 us
206     _delay_us(1);
207 
208     // Pull CLK high
209     GPIO_write_high(&PORTD, SEGMENT_CLK);
210 
211     // Wait 1 us
212     _delay_us(1);
```

```
213
214     // Pull CLK low
215     GPIO_write_low(&PORTD, SEGMENT_CLK);
216 }
```