

# RTOS Assignment 4: 7-Segment Display

Dan Blanchette

January 30 2023

## 1 Program Tasks and Functions

### 1.1 Tasks

#### 1.1.1 void Task 1: Priority 5

Task one was designed to be the task that handles the count-up and increments the left and right displays accordingly. As a partial solution, I created three nested for loops with the left segment being tied to the inner-most loop. The outer loop, I concluded, could be used to increment the right segment's digits by passing the iterator from the outer-most loop to the numbers function. As a Test for the segment's refresh rate, I kept the number passed as zero to the numbers function. I had two task delays implemented. The outer loop has a delay of 15 divided by the port tick rate, while the innner loop has a delay of 10 also divided by the port tick rate.

#### 1.1.2 void Task 2: Priority 4

This task was implemented to address the requirement that the D13 LED should blink in sync with the second count. When tested, the pico's D13 LED blinked in time with the second change of the right segment. However, this was using delays, and the requirement was to sync it with the task that is handled by the counter.

#### 1.1.3 void Task 3: Priority N/A

This was in testing as a possible countdown method for the assignment. It has not been tested or fully implemented.

### 1.2 Functions

#### 1.2.1 void setup 7seg()

This function has the code to set up the GPIO pin initialization for the seven-segment display.

### 1.2.2 void numbers(const int)

In this function, I set up the GPIO pins to display the digit patterns on the seven-segment display. The implementation of this function uses a switch statement to allow for ease of access for each digit by passing an integer value as a function parameter.

## 2 Block Diagram

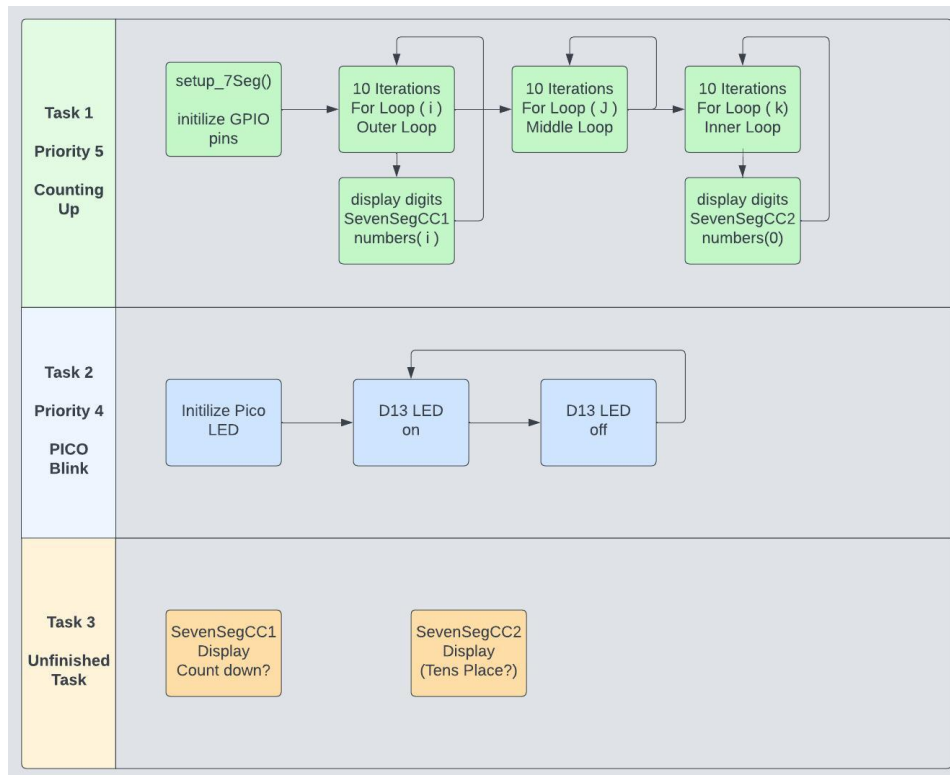


Figure 1: block diagram

## 3 Code

```
1 /**
2  * @file main.c
3  * @author Dan Blanchette
4  * @brief This program will count down from 42 to 00 and back up
   from 00 to 42 using a 7-segment display.
```

```

5  *      As an additional feature, the D13 LED on the Pico Feather
        will be synced with the second count.
6  * @version 0.1
7  * @date Started: 2023-02-3, Due: 2023-02-8
8  * Total Hours: 21 (coding and research)
9  * @copyright Copyright (c) 2023
10 *
11 */
12 #include "main.h"
13
14 // GPIO pin setup
15 #define SevenSegCC1 11
16 #define SevenSegCC2 10
17
18 #define SevenSegA 26
19 #define SevenSegB 27
20 #define SevenSegC 29
21 #define SevenSegD 18
22 #define SevenSegE 25
23 #define SevenSegF 7
24 #define SevenSegG 28
25 #define SevenSegDP 24
26
27
28
29
30
31 /*FUNCTION PROTOTYPES*/
32 // setup 7-seg I/O
33 void setup_7seg();
34 // draws the number pattern for the display
35 void numbers(const int);
36
37
38 /*TASKS*/
39 // Counts up to 9 for now
40 void task_1()
41 {
42     while (true)
43     {
44         setup_7seg();
45         printf("Start Countdown Loop\n");
46         // CC1: is the right display segment
47         // CC2: is the left display segment
48
49         // counts down two repetitions
50         int i = 0;
51         for (i; i < 10; i++)
52         {
53             gpio_put(SevenSegCC1, 1);
54             gpio_put(SevenSegCC2, 0);
55             // 7-seg numbers
56             numbers(i);
57             vTaskDelay(15 / portTICK_PERIOD_MS);
58             gpio_put(SevenSegCC1, 0);
59             gpio_put(SevenSegCC2, 1);
60             for (int j = 0; j < 10; j++)

```

```

61     {
62         inner-most loop holds left display steady
63         for (int k = 0; k < 10; k++)
64         {
65             numbers(0);
66             vTaskDelay(10 / portTICK_PERIOD_MS);
67         }
68     }
69 }
70 }
71 }
72
73 // pico blink
74 void task_2()
75 {
76     // blinks D13 LED on Pico
77     const uint LED_PIN = PICO_DEFAULT_LED_PIN;
78     while(true)
79     {
80         gpio_init(LED_PIN);
81         gpio_set_dir(LED_PIN, GPIO_OUT);
82         // Flash pico LED at top of loop
83
84         // pico led on
85         gpio_put(LED_PIN, 1);
86         vTaskDelay(500 / portTICK_PERIOD_MS);
87         // pico led off
88         gpio_put(LED_PIN, 0);
89         vTaskDelay(500 / portTICK_PERIOD_MS);
90     }
91 }
92
93 // counting down
94 void task_3()
95 {
96     while (true)
97     {
98
99         printf("Start Countdown Loop\n");
100        // CC1: is the right display segment
101        // CC2: is the left display segment
102
103        // counts down two repetitions
104        int i = 10;
105        for (i; i > 0; i--)
106        {
107            gpio_put(SevenSegCC1, 1);
108            gpio_put(SevenSegCC2, 0);
109            // 7-seg numbers
110            numbers(i);
111            vTaskDelay(20 / portTICK_PERIOD_MS);
112
113            gpio_put(SevenSegCC1, 0);
114            gpio_put(SevenSegCC2, 1);
115            for (int j = 0; j < 10; j++)
116            {
117                for (int k = 0; k < 10; k++)

```

```

118         {
119             numbers(0);
120             vTaskDelay(10 / portTICK_PERIOD_MS);
121         }
122     }
123 }
124 }
125 }
126
127 int main()
128 {
129     // Use for debugging
130     stdio_init_all();
131     // This first task function's format is meant as a reference
132     xTaskCreate(
133         task_1, // fucntion to be called
134         "Task_1", // Name of Task
135         256, // Stack Size
136         NULL, // Parameter to pass to a function
137         5, // Task Priority (0 to configMAX_PRIORITIES - 1)
138         NULL // Task handle (check on status, watch memory
139             usage, or end the task)
140     );
141     // xTaskCreate(task_2, "Task_2", 256, NULL, 4, NULL);
142     // tell the scheduler to start running
143     vTaskStartScheduler();
144     while (1){}
145 }
146
147 /* FUNCTION DEFINITIONS */
148 // setup for 7 segment display's GPIO pins
149 void setup_7seg()
150 {
151     // initialize digital pin LED_BUILTIN as an output.
152     gpio_init(SevenSegA);
153     gpio_init(SevenSegB);
154     gpio_init(SevenSegC);
155     gpio_init(SevenSegD);
156     gpio_init(SevenSegE);
157     gpio_init(SevenSegF);
158     gpio_init(SevenSegG);
159     // This GPIO pin activates the decimal point on the 7 segment
160     display
161     gpio_init(SevenSegDP);
162
163     gpio_init(SevenSegCC1);
164     gpio_init(SevenSegCC2);
165
166     gpio_set_dir(SevenSegA, GPIO_OUT);
167     gpio_set_dir(SevenSegB, GPIO_OUT);
168     gpio_set_dir(SevenSegC, GPIO_OUT);
169     gpio_set_dir(SevenSegD, GPIO_OUT);
170     gpio_set_dir(SevenSegE, GPIO_OUT);
171     gpio_set_dir(SevenSegF, GPIO_OUT);
172     gpio_set_dir(SevenSegG, GPIO_OUT);
173     gpio_set_dir(SevenSegDP, GPIO_OUT);

```

```

173     gpio_set_dir(SevenSegCC1, GPIO_OUT);
174     gpio_set_dir(SevenSegCC2, GPIO_OUT);
175 }
176 // "Draws" numbers to the 7-Segment Display
177 void numbers(const int segNum)
178 {
179     // segNUM (int) is passed to the function and selects the
180     // pattern
181     switch (segNum)
182     {
183     case 0:
184         /*
185          A
186
187          F | G | B   | |
188          E |___| C   |__|
189          D
190          */
191         gpio_put(SevenSegA, 1);
192         gpio_put(SevenSegB, 1);
193         gpio_put(SevenSegC, 1);
194         gpio_put(SevenSegD, 1);
195         gpio_put(SevenSegE, 1);
196         gpio_put(SevenSegF, 1);
197         gpio_put(SevenSegG, 0);
198         break;
199
200     case 1:
201
202         // display #1
203         /*
204          | B
205          | C
206          */
207         gpio_put(SevenSegA, 0);
208         gpio_put(SevenSegB, 1);
209         gpio_put(SevenSegC, 1);
210         gpio_put(SevenSegD, 0);
211         gpio_put(SevenSegE, 0);
212         gpio_put(SevenSegF, 0);
213         gpio_put(SevenSegG, 0);
214         break;
215
216     case 2:
217         // display #2 on the right segment
218         /*
219          --|
220          |__
221          */
222         gpio_put(SevenSegA, 1);
223         gpio_put(SevenSegB, 1);
224         gpio_put(SevenSegC, 0);
225         gpio_put(SevenSegD, 1);
226         gpio_put(SevenSegE, 1);
227         gpio_put(SevenSegF, 0);
228         gpio_put(SevenSegG, 1);

```

```

229         break;
230     case 3:
231         // display #3 on the right segment
232         /*  --
233            --|
234            --|
235            */
236         gpio_put(SevenSegA, 1);
237         gpio_put(SevenSegB, 1);
238         gpio_put(SevenSegC, 1);
239         gpio_put(SevenSegD, 1);
240         gpio_put(SevenSegE, 0);
241         gpio_put(SevenSegF, 0);
242         gpio_put(SevenSegG, 1);
243         break;
244     case 4:
245         // display #4 on the right segment
246         /*  |__|
247            |
248            */
249         gpio_put(SevenSegA, 0);
250         gpio_put(SevenSegB, 1);
251         gpio_put(SevenSegC, 1);
252         gpio_put(SevenSegD, 0);
253         gpio_put(SevenSegE, 0);
254         gpio_put(SevenSegF, 1);
255         gpio_put(SevenSegG, 1);
256         break;
257     case 5:
258         // display #5 on the right segment
259         /*  --
260            |__
261            --|*/
262         gpio_put(SevenSegA, 1);
263         gpio_put(SevenSegB, 0);
264         gpio_put(SevenSegC, 1);
265         gpio_put(SevenSegD, 1);
266         gpio_put(SevenSegE, 0);
267         gpio_put(SevenSegF, 1);
268         gpio_put(SevenSegG, 1);
269         break;
270     case 6:
271         // display #6 on the right segment
272         /*  --
273            |__
274            |__|*/
275         gpio_put(SevenSegA, 1);
276         gpio_put(SevenSegB, 0);
277         gpio_put(SevenSegC, 1);
278         gpio_put(SevenSegD, 1);
279         gpio_put(SevenSegE, 1);
280         gpio_put(SevenSegF, 1);
281         gpio_put(SevenSegG, 1);
282         break;
283     case 7:
284         // display #7 on the right segment

```

```

286     /*
287         --
288         |
289         |
290     */
291     gpio_put(SevenSegA, 1);
292     gpio_put(SevenSegB, 1);
293     gpio_put(SevenSegC, 1);
294     gpio_put(SevenSegD, 0);
295     gpio_put(SevenSegE, 0);
296     gpio_put(SevenSegF, 0);
297     gpio_put(SevenSegG, 0);
298     break;
299 case 8:
300     // display #8 on the right segment
301     /*
302         --
303         |__|
304         |__|*/
305     gpio_put(SevenSegA, 1);
306     gpio_put(SevenSegB, 1);
307     gpio_put(SevenSegC, 1);
308     gpio_put(SevenSegD, 1);
309     gpio_put(SevenSegE, 1);
310     gpio_put(SevenSegF, 1);
311     gpio_put(SevenSegG, 1);
312     break;
313 case 9:
314     // display #9 on the right segment
315     /*
316         --
317         |__|
318         |
319         |
320     */
321     gpio_put(SevenSegA, 1);
322     gpio_put(SevenSegB, 1);
323     gpio_put(SevenSegC, 1);
324     gpio_put(SevenSegD, 0);
325     gpio_put(SevenSegE, 0);
326     gpio_put(SevenSegF, 1);
327     gpio_put(SevenSegG, 1);
328     break;
329 default:
330     // this is for debug purposes
331     printf("Please enter a value between 0-9");
}
}

```