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

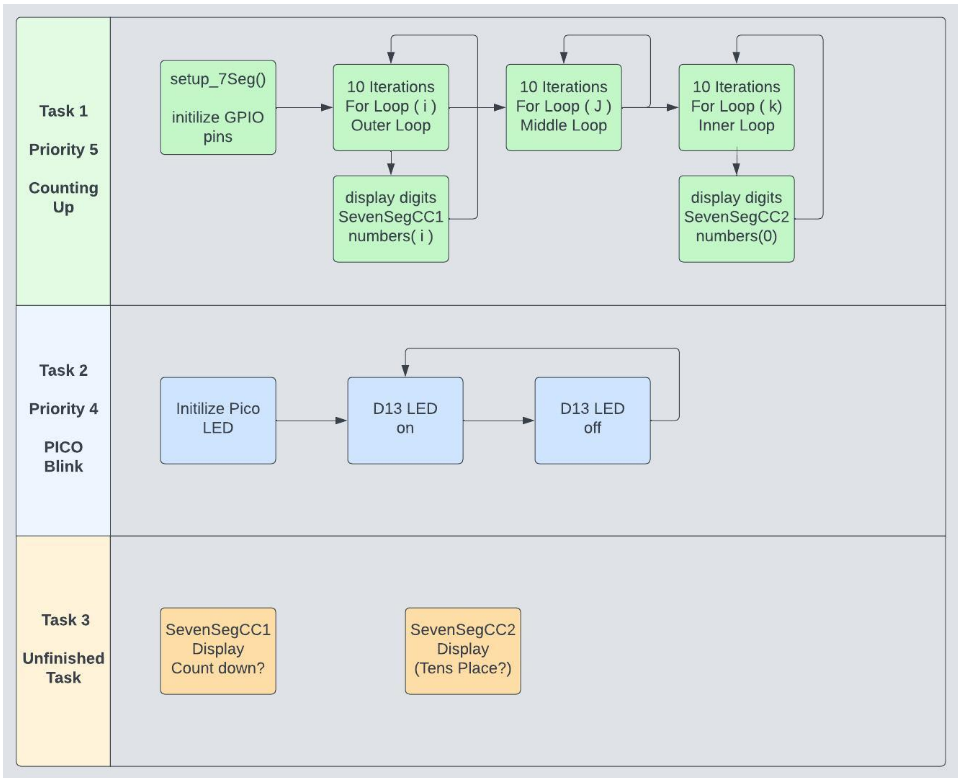
1.2.1 void setup 7seg()

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



Task 1
Priority 5
Counting Up
setup_7Seg() initilize GPIO pins
10 Iterations For Loop (i) Outer Loop
display digits SevenSegCC1 numbers(i)
10 Iterations For Loop (J) Middle Loop
10 Iterations For Loop (k) Inner Loop
display digits SevenSegCC2 numbers(0)
Task 2
Priority 4
PICO Blink
Initilize Pico LED
D13 LED on
D13 LED off
Task 3
Unfinished Task
SevenSegCC1 Display Count down?
SevenSegCC2 Display (Tens Place?)

Figure 1: block diagram

3 Code

```
/**
 * @file main.c
 * @author Dan Blanchette
 * @brief  This program will count down from 42 to 00 and back up
    from 00 to 42 using a 7-segment display.
 *      As an additional feature, the D13 LED on the Pico Feather
    will be synced with the second count.
 * @version 0.1
 * @date Started: 2023-02-3, Due: 2023-02-8
 * Total Hours: 21 (coding and research)
 * @copyright Copyright (c) 2023
 *
 */
#include "main.h"

// GPIO pin setup
#define SevenSegCC1 11
#define SevenSegCC2 10

#define SevenSegA 26
#define SevenSegB 27
#define SevenSegC 29
#define SevenSegD 18
#define SevenSegE 25
#define SevenSegF 7
#define SevenSegG 28
#define SevenSegDP 24





/*FUNCTION PROTOTYPES*/
// setup 7-seg I/O
void setup_7seg();
// draws the number pattern for the display
void numbers(const int);


/*TASKS*/
// Counts up to 9 for now
void task_1()
{
  while (true)
  {
    setup_7seg();
    printf("Start CountDown Loop\n");
    // CC1: is the right display segment
    // CC2: is the left display segment

    // counts down two repetitions
    int i = 0;
    for (i; i < 10; i++)
    {
      gpio_put(SevenSegCC1, 1);
      gpio_put(SevenSegCC2, 0);
      // 7-seg numbers
      numbers(i);
      vTaskDelay(15 / portTICK_PERIOD_MS);
      gpio_put(SevenSegCC1, 0);
      gpio_put(SevenSegCC2, 1);
      for (int j = 0; j < 10; j++)
      {
                    inner-most loop holds left display steady
        for (int k = 0; k < 10; k++)
        {
          numbers(0);
          vTaskDelay(10 / portTICK_PERIOD_MS);
        }
      }
    }
  }
}

// pico blink
void task_2()
{
  // blinks D13 LED on Pico
  const uint LED_PIN = PICO_DEFAULT_LED_PIN;
  while(true)
  {
    gpio_init(LED_PIN);
    gpio_set_dir(LED_PIN, GPIO_OUT);
    // Flash pico LED at top of loop

    // pico led on
    gpio_put(LED_PIN, 1);
    vTaskDelay(500 / portTICK_PERIOD_MS);
    // pico led off
    gpio_put(LED_PIN, 0);
    vTaskDelay(500 / portTICK_PERIOD_MS);
  }
}

// counting down
void task_3()
{
  while (true)
  {

   printf("Start CountDown Loop\n");
   // CC1: is the right display segment
   // CC2: is the left display segment

   // counts down two repetitions
   int i = 10;
   for (i; i > 0; i--)
   {
      gpio_put(SevenSegCC1, 1);
      gpio_put(SevenSegCC2, 0);
      // 7-seg numbers
      numbers(i);
      vTaskDelay(20 / portTICK_PERIOD_MS);

      gpio_put(SevenSegCC1, 0);
      gpio_put(SevenSegCC2, 1);
      for (int j = 0; j < 10; j++)
      {
        for (int k = 0; k < 10; k++)
        {
          numbers(0);
          vTaskDelay(10 / portTICK_PERIOD_MS);
        }
    }
   }
  }
}

int main()
{
  // Use for debugging
  stdio_init_all();
 // This first task function's format is meant as a reference
  xTaskCreate(
              task_1, // fucntion to be called
             "Task_1", // Name of Task
              256,  // Stack Size
              NULL, // Parameter to pass to a function
              5,  // Task Priority (0 to configMAX_PRIORITIES - 1)
              NULL // Task handle (check on status, watch memory
    usage, or end the task)
              );
  // xTaskCreate(task_2, "Task_2", 256, NULL, 4, NULL);
  // tell the scheduler to start running
  vTaskStartScheduler();

  while (1){}
}

/* FUNCTION DEFINITIONS */
// setup for 7 segment display's GPIO pins
void setup_7seg()
{
    // initialize digital pin LED_BUILTIN as an output.
    gpio_init(SevenSegA);
    gpio_init(SevenSegB);
    gpio_init(SevenSegC);
    gpio_init(SevenSegD);
    gpio_init(SevenSegE);
    gpio_init(SevenSegF);
    gpio_init(SevenSegG);
    // This GPIO pin activates the decimal point on the 7 segment
    display
    gpio_init(SevenSegDP);

    gpio_init(SevenSegCC1);
    gpio_init(SevenSegCC2);

    gpio_set_dir(SevenSegA, GPIO_OUT);
    gpio_set_dir(SevenSegB, GPIO_OUT);
    gpio_set_dir(SevenSegC, GPIO_OUT);
    gpio_set_dir(SevenSegD, GPIO_OUT);
    gpio_set_dir(SevenSegE, GPIO_OUT);
    gpio_set_dir(SevenSegF, GPIO_OUT);
    gpio_set_dir(SevenSegG, GPIO_OUT);
    gpio_set_dir(SevenSegDP, GPIO_OUT);

    gpio_set_dir(SevenSegCC1, GPIO_OUT);
    gpio_set_dir(SevenSegCC2, GPIO_OUT);
}
// "Draws" numbers to the 7-Segment Display
void numbers(const int segNum)
{
  // segNUM (int) is  passed to the function and selects the
    pattern
  switch (segNum)
  {
    case 0:
      /*
         A
         ___         __
      F | G | B     |  |
      E |___| C     |__|
          D
      */
        gpio_put(SevenSegA, 1);
        gpio_put(SevenSegB, 1);
        gpio_put(SevenSegC, 1);
        gpio_put(SevenSegD, 1);
        gpio_put(SevenSegE, 1);
        gpio_put(SevenSegF, 1);
        gpio_put(SevenSegG, 0);
        break;

    case 1:

    // display #1
    /*
      | B
        | C
    */
        gpio_put(SevenSegA, 0);
        gpio_put(SevenSegB, 1);
        gpio_put(SevenSegC, 1);
        gpio_put(SevenSegD, 0);
        gpio_put(SevenSegE, 0);
        gpio_put(SevenSegF, 0);
        gpio_put(SevenSegG, 0);
        break;

    case 2:
      // display #2 on the right segment
      /* __
        __|
        |__
          */
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 0);
      gpio_put(SevenSegD, 1);
      gpio_put(SevenSegE, 1);
      gpio_put(SevenSegF, 0);
      gpio_put(SevenSegG, 1);
      break;
    case 3:
      // display #3 on the right segment
      /* __
        __|
        __|
          */
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 1);
      gpio_put(SevenSegE, 0);
      gpio_put(SevenSegF, 0);
      gpio_put(SevenSegG, 1);
      break;
    case 4:
      // display #4 on the right segment
      /*
        |__|
           |
          */
      gpio_put(SevenSegA, 0);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 0);
      gpio_put(SevenSegE, 0);
      gpio_put(SevenSegF, 1);
      gpio_put(SevenSegG, 1);
      break;
    case 5:
      // display #5 on the right segment
      /*  __
        |__
         __|*/
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 0);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 1);
      gpio_put(SevenSegE, 0);
      gpio_put(SevenSegF, 1);
      gpio_put(SevenSegG, 1);
      break;
    case 6:
       // display #6 on the right segment
      /*  __
        |__
        |__|*/
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 0);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 1);
      gpio_put(SevenSegE, 1);
      gpio_put(SevenSegF, 1);
      gpio_put(SevenSegG, 1);
      break;
    case 7:
      // display #7 on the right segment
      /*
         __
          |
          |
      */
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 0);
      gpio_put(SevenSegE, 0);
      gpio_put(SevenSegF, 0);
      gpio_put(SevenSegG, 0);
      break;
    case 8:
      // display #8 on the right segment
      /*  __
        |__|
        |__|*/
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 1);
      gpio_put(SevenSegE, 1);
      gpio_put(SevenSegF, 1);
      gpio_put(SevenSegG, 1);
      break;
    case 9:
      // display #9 on the right segment
      /*  __
        |__|
          |
          */
      gpio_put(SevenSegA, 1);
      gpio_put(SevenSegB, 1);
      gpio_put(SevenSegC, 1);
      gpio_put(SevenSegD, 0);
      gpio_put(SevenSegE, 0);
      gpio_put(SevenSegF, 1);
      gpio_put(SevenSegG, 1);
      break;

      default:
      // this is for debug purposes
      printf("Please enter a value between 0-9");
  }
}
```