Lab 1

by
Garett Loghry

CS466 – Embedded Systems
LABORATORY REPORT

Computer Science, Washington State University Vancouver
January 27th, 2023

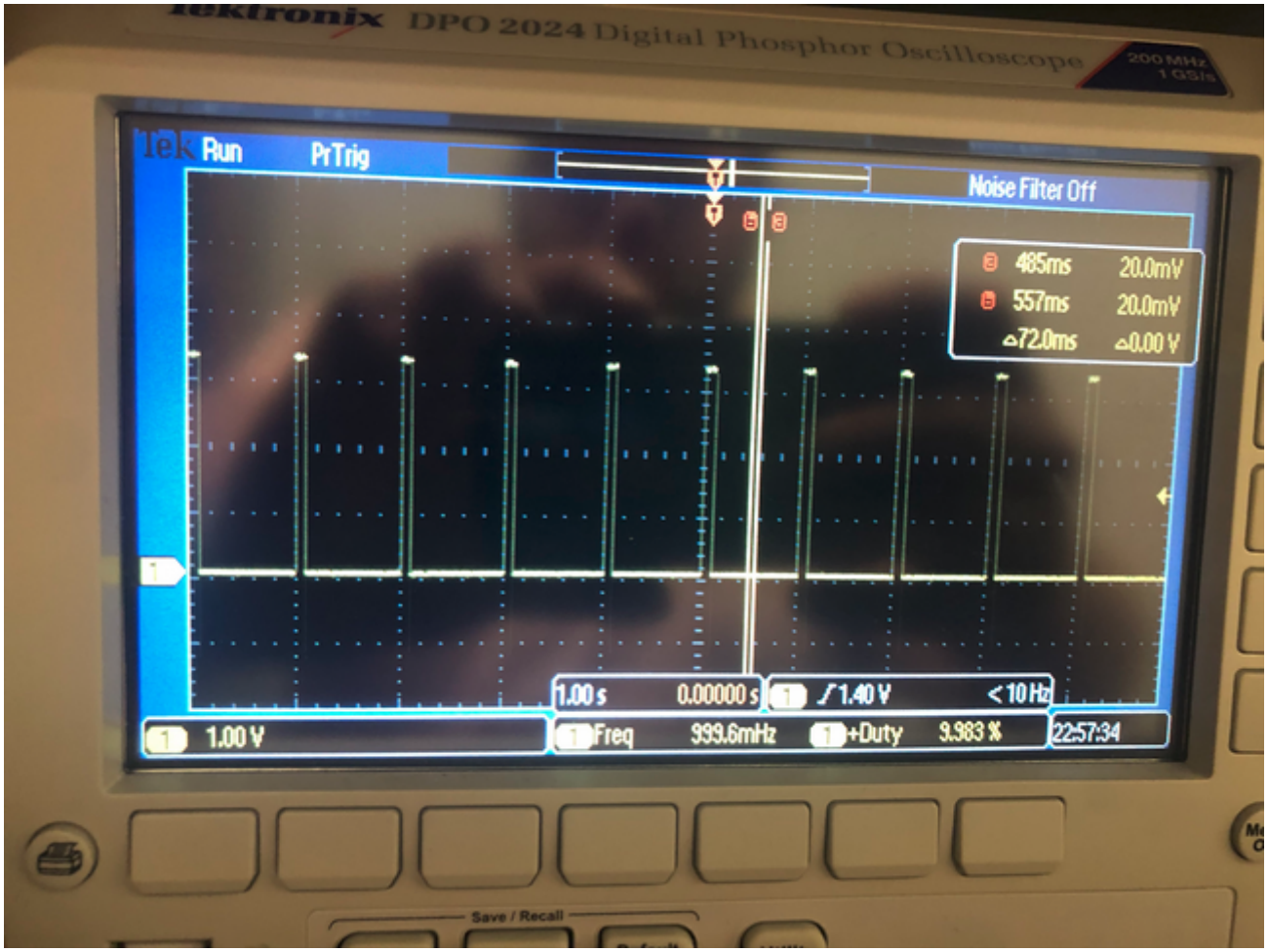
Objective:
The objective of lab 1 is to verify our installation of the Pico SDK and start to build our fundamentals of embedded system designs. This lab spanned multiple weeks, and while the actual work wasn't overly complicated, there were a variety of small details to make sure that we got right, and some common mistakes were pointed out that should help us going forward.

Apparatus:
- Raspberry Pi Pico
- Breadboard (protoboard)
- 2 DIP buttons
- Wires
- Oscilloscope (Tektroinx – not sure of model)

Method:
There were a variety of pre-lab work that needed to get done, such as installing FreeRTOS, downloading the compiler, and pulling from the GitHub repository. Some of this was laid out in lab 1, some of it was part of the README.md file that was supplied with the repository we pulled. From there we installed a premade "blinky" program onto the Pico. This would be our basis for the code for lab 1. From there, we made alterations to the presupplied code and expanded the "blinky" program.

Data:
All code for this lab is supplied as an addendum to this report. Comments are made for each problem solution, and preprocessor directives are used to separate each problem. Each section should have something commented like `/* Problem 12 */` for example. Some problems have questions that the lab asks, and it is assumed that these questions are to be answered here. Some have subsections that appear to be more like comments (such as part 6) and don't ask any specific question; they are skipped if no question has been asked.

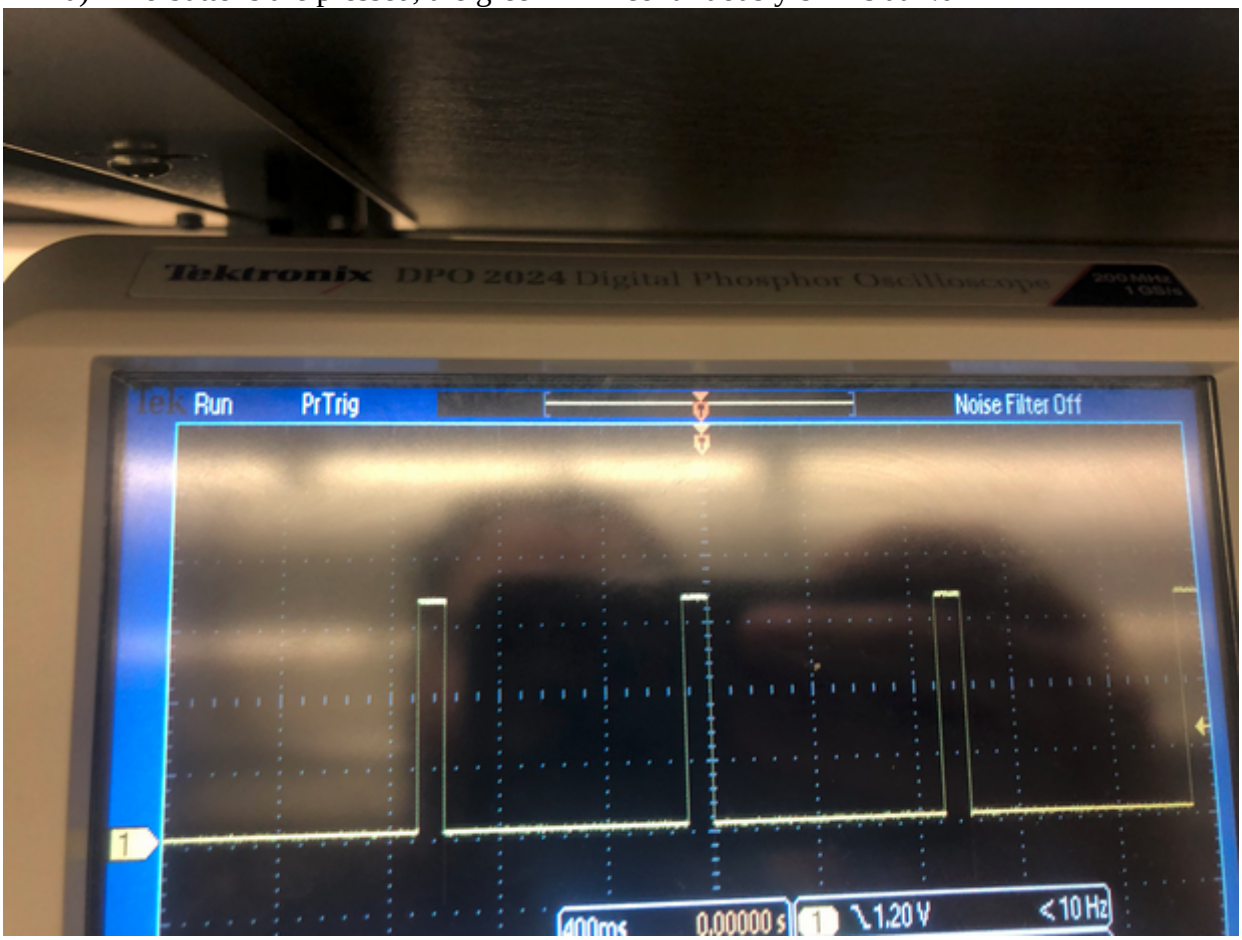
> Problem 7:
> The expected operation of the lab1.c provided program is to blink the LED in a 1Hz frequency with a 10% duty load.
>
> Problem 10:
> Scope data included below. Scope shows a `Freq 999.6mHz` and a `+Duty 9.983 %`. This matches our expected 1Hz and 10% duty within an acceptable margin of error.



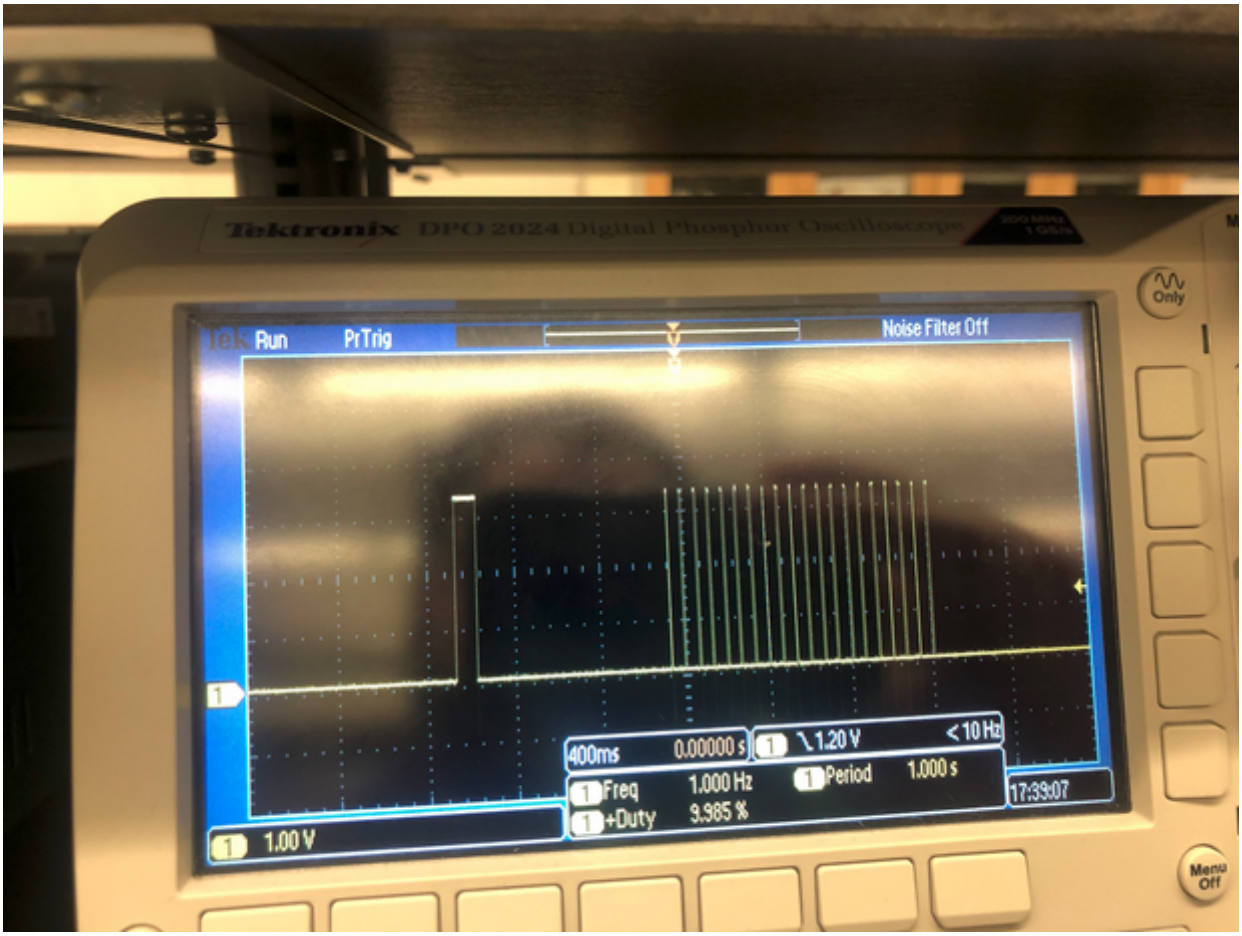
Problem 11:
I solved problem 11 in the simplest way I knew how. Turn on the LED when the button is pressed. This was initially just going to be a test, but I found when I had both buttons wired and pressed both buttons, it would light the LED the normal amount, but if I pressed a single button the LED would get dimmer. This happens, I believe, as a result of an unintended PWM. The while() loop is continuous, and the LED gets marginally dimmer when it has to rapidly turn on and off (in the case that just one button was pressed.) After talking to Miller this wasn't the expected solution, but it is simple and to the point, and so I kept it as is.

Problem 13:
Problem 13 piggy backs from problem 12. Each part will be outlined below. Including images is a little rough here, but I did my best. While still getting used to the scope, I've found that one image is lacking critical frequency data, but it is still supplied. Also, The single button presses were counted for correctness, but I wasn't sure how to verify frequency. It *appears* correct, but more work is needed to be comfortable with the scope.

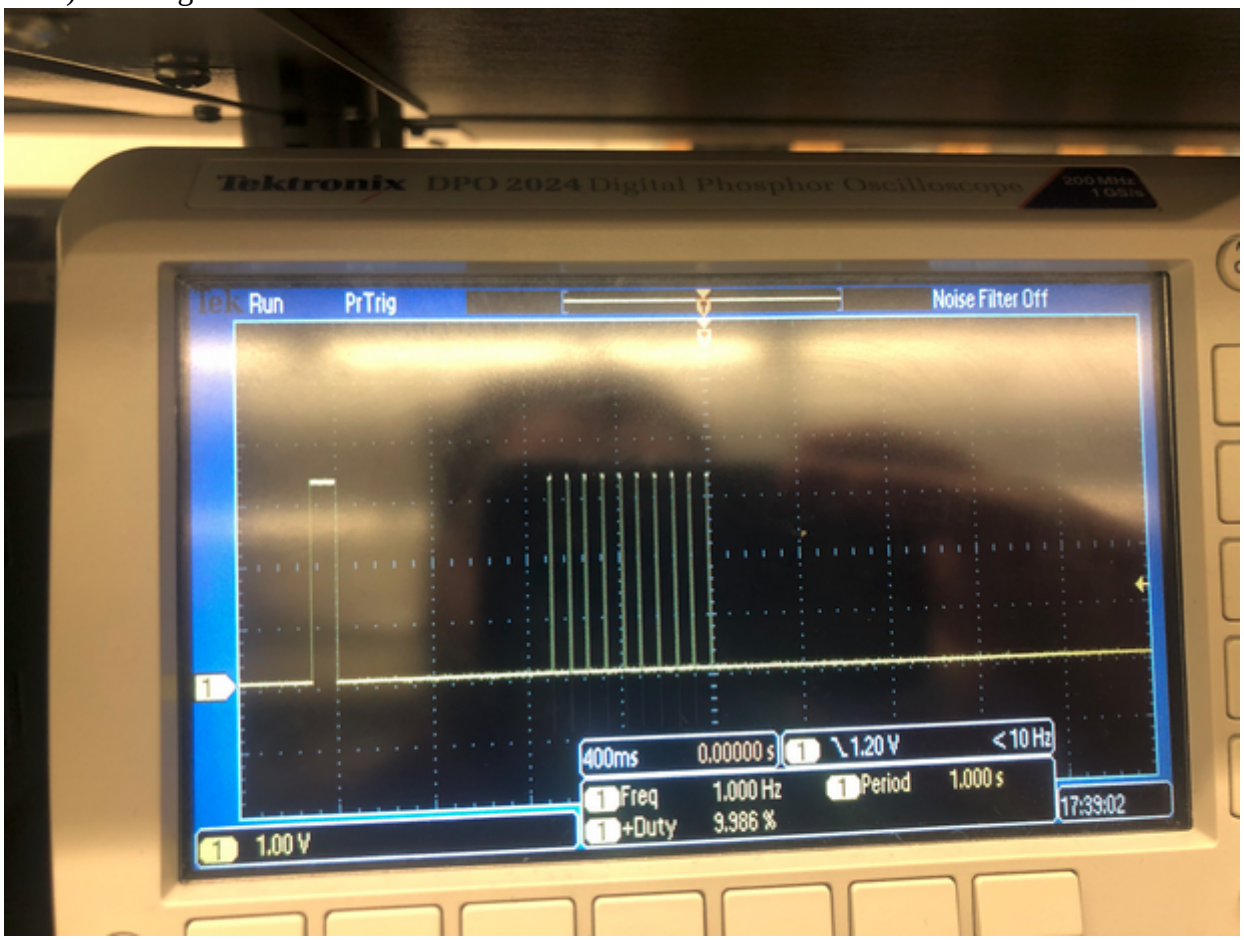
a) If no buttons are pressed, the green LED continuously blinks at 1.0Hz



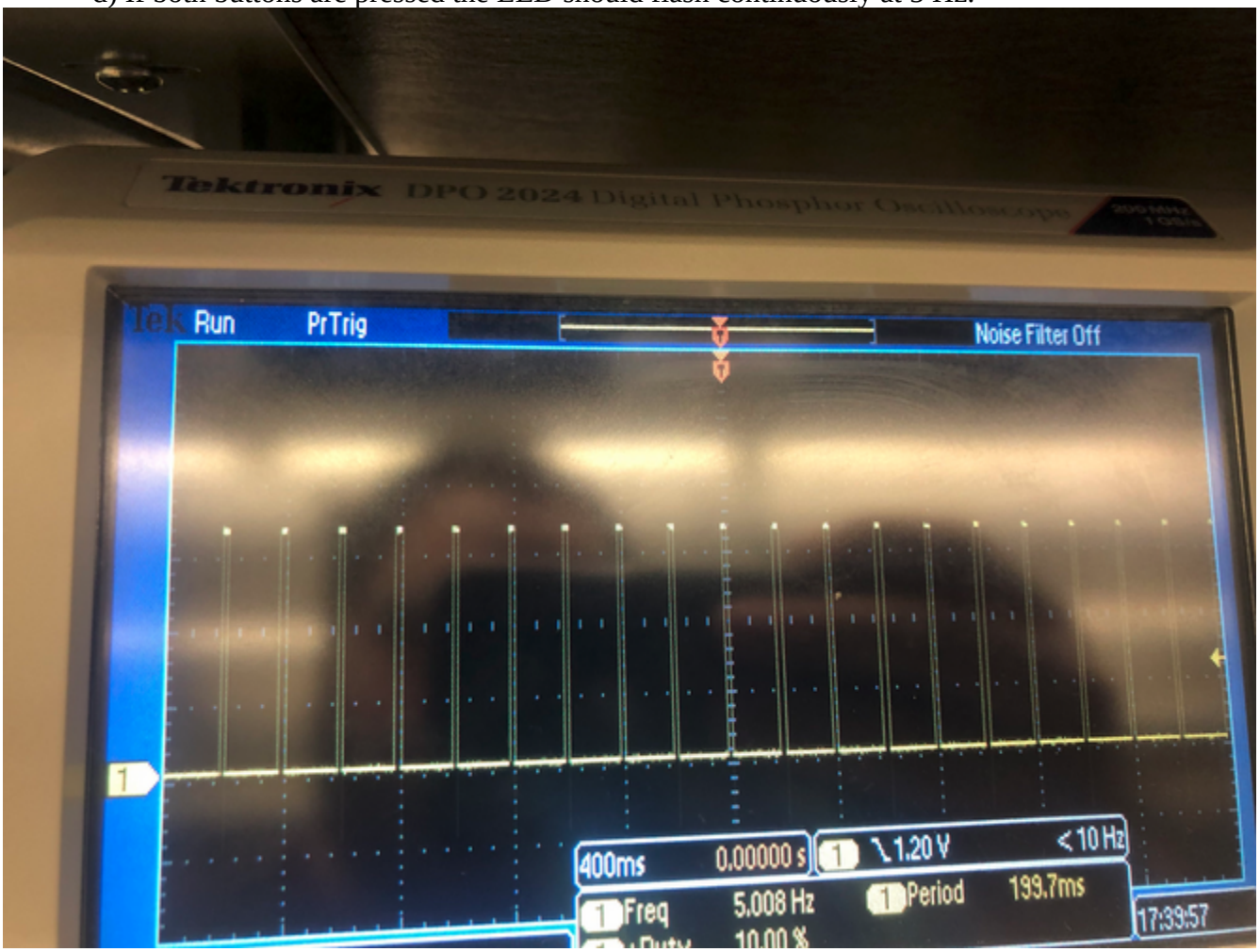
b) Pressing SW1 will cause the led to flash 20 times at 15Hz



c) Pressing SW2 will cause the led to flash 10 times at 13Hz



d) If both buttons are pressed the LED should flash continuously at 5 Hz.



Results and Analysis:
Little to nothing is required for analysis for this lab, as much of the work was refamiliarizing ourselves with the tools and equipment, as well as verifying that we have a working test bench. All questions asked by the lab where answered in the **Data** section of this report other than part 11 subsection c. It asks “Lab Writeup: Describe the results and any conclusions from your test.” in relation to devising a way to check if switch 1, switch 2, or both are shorted to ground. That section will be answered below.

Problem 11:
Part c.) Looking back, it seems the instructor wanted us to think about how difficult it can be to work with a single LED, but multiple buttons/switches. The pseudo PWM that I devised, while somewhat clever, breaks down when you start to add more and more switches. How could you know if button 2 is pressed vs button 23? The answer given in later sections about having each button attached to a different frequency seems much smarter and well thought out, and it seems likely that this is closer to the industry standard for these types of things.

Conclusion:
This lab was intended to refamiliarize ourselves with working with breadboards, buttons/switches, wires, and other things that are closely related to embedded systems. In that regard I think I learned a lot from this lab and got out what was intended for me to get. The instructor provided some helpful insights about how and why certain solutions weren't optimal. Looking back on my solution to problem 12, I have a huge delay in every for() loop, and have introduced a massive amount of timing unknowns and too much jitter to be actually usable. I one of the core things I learned from this lab is that I must be more mindful for timing and resources than I have to be in other areas of programming. This type of development is very different than working in python, and I've still got a lot to learn, but I'm confident I'm on my way.

lab1.c code addendum starts here

```
/**
 * @brief CS466 Lab1 Blink proigram based on pico blink example
 *
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

#ifndef true
#define true 1
#endif

#ifndef false
#define false 0
#endif

#define LED_PIN 25
#define GPIO_18 18
#define SW1 17
#define SW2 16

#include "pico/stdlib.h"

void my_gpio_init(void) {
    gpio_init(LED_PIN);
    gpio_set_dir(LED_PIN, GPIO_OUT);
    /* New code starts here */
    gpio_init(GPIO_18);
    gpio_set_dir(GPIO_18, GPIO_OUT);

    gpio_init(SW1);
    gpio_set_dir(SW1, GPIO_IN);
    gpio_pull_up(SW1);

    gpio_init(SW2);
    gpio_set_dir(SW2, GPIO_IN);
    gpio_pull_up(SW2);
    /* end of new code added to init */
}

void led_control(bool isOn) {
    gpio_put(LED_PIN, isOn);
    gpio_put(GPIO_18, isOn);
}

bool isPressed(uint button) { return (!gpio_get(button)); }

int main() {
    my_gpio_init();

    while (true) {
#if 0   /* ORIGINAL CODE GIVEN */
```

```
        gpio_put(LED_PIN, 1);
        gpio_put(LED_PIN, 0);
#endif

#if 0   /* PROBLEM 10 */
        led_control(1);
        sleep_ms(100);
        led_control(0);
        sleep_ms(900);
#endif

#if 0   /* PROBLEM 11 */
        led_control(gpio_get(SW1));
        led_control(gpio_get(SW2));
#endif

#if 1   /* PROBLEM 12 */
        if (isPressed(SW1)) {
            for (int i = 0; i < 20; i++) {
                /* UNKONWN REQUIREMENT
                 * Both buttons pressed while in the middle of one loop
                 * Breaking and moving onto both to both buttons function
                 */
                if (isPressed(SW2)) break;

                /* Want 15Hz blink, best approx */
                led_control(1);
                sleep_ms(7);                // 100/15 rounded
                led_control(0);
                sleep_ms(60);               // 900/15
            }
            /* Weird logic, if switch 1 held and
             * switch 2 not pressed do nothing.
             * Required to act as a trigger.
             * KNOWN BUG: If held for about 2s will still trigger again
             *            but if held for about 1s or 5s it won't.
             *            Not sure why.
             */
            while (isPressed(SW1) && !isPressed(SW2));
        }

        if (isPressed(SW2)) {
            for (int i = 0; i < 10; i++) {
                /* UNKONWN REQUIREMENT
                 * Both buttons pressed while in the middle of one loop
                 * Breaking and moving onto both to both buttons function
                 */
                if (isPressed(SW1)) break;

                /* Want 13Hz blink, best approx */
                led_control(1);
                sleep_ms(8);                // 100/13 rounded
                led_control(0);
                sleep_ms(69);               // 900/13
```

```
            }
            /* Weird logic, if switch 2 held and
             * switch 1 not pressed do nothing.
             * Required to act as a trigger.
             * KNOWN BUG: If held for about 2s will still trigger again
             *            but if held for about 1s or 5s it won't.
             *            Not sure why.
             */
            while (isPressed(SW2) && !isPressed(SW1));
        }

        while (isPressed(SW1) && isPressed(SW2)) {
            /* Want 5Hz blink when both switches pressed */
            led_control(1);
            sleep_ms(20);
            led_control(0);
            sleep_ms(180);
        }

        while (!isPressed(SW1) && !isPressed(SW2)) {
            /* Want 1Hz blink when no switches pressed */
            led_control(1);
            sleep_ms(5);
            led_control(0);
            sleep_ms(995);
        }
#endif

#if 0   /* working alternate of problem 12 */
        bool trigger1 = isPressed(SW1);
        bool trigger2 = isPressed(SW2);
        if (trigger1
        } else {
            sleep_ms(1);
        }
#endif
    }
}
```