Alexander Akomer (Partner: Shaheen Ebrahimi)

CSCE 462 – 500 Liu

1/28/2022

Lab 1 Report

1) **What is RPI 3 CPU clock speed? Why is it important to know?**
   The Raspberry Pi 3 has a clock speed of 1200 MHz. This is an important statistic to be aware of because it is an indicator of the processor's performance.

2) **What is the range of voltage that represents logic low on Raspberry Pi?**
   The Raspberry Pi interprets anything within the range of 0 – 1.8V as logical low.

3) **What is the range of voltage that represents logic high on Raspberry Pi?**
   The Raspberry Pi interprets anything within the range of 1.8 – 3.3V as logical high.

4) **Compare the Raspberry Pi 3 with a standard PC, can you use Pi as a desktop PC? What's an application for RPi?**
   The Raspberry Pi 3 is a much weaker computer in terms of memory, processing power, storage, and clock speeds. It also runs on much lower power consumption than a typical desktop PC. This means that it is unable to handle some of the more powerful tasks a regular desktop PC might be able to accomplish, but it functions perfectly for low-power and simple processes. In conjunction with some external memory, an RPi can function perfectly as a small personal cloud device, or can function as a Wi-Fi adapter to machines that typically require a physical connection (like printers).

5) **Discuss the difference between polling and interrupt. What situation should we use polling and what situation should we use interrupts?**
   Polling typically makes use of an infinite loop, constantly checking for an action to occur to respond to it while interrupts are a way of waiting for an action to occur without constantly checking through the use of signals. Interrupts are better than polling in a large majority of cases. Interrupts cause the performance of the microcontroller to be far better because it does not have to continuously check to see whether an action has occurred and will instead just receive a single notification to perform further actions. This means that there are no busy-wait cycles in an interrupt scheme. However, polling can have an advantage in the instance that the way for the interrupt to be triggered is on a frequent I/O interruption of a short duration, especially if these occur on near-regular intervals. In this case, polling method can have an efficiency advantage.

6) **What is hardware interrupt? What is software interrupt?**
   A hardware interrupt is a signal sent to the CPI from beyond the computer – an external device. A software interrupt occurs when an application on the machine needs attention from the OS – for instance, there is a divide-by-zero error in some code.

Code:

```
import RPi.GPIO as GPIO
from time import sleep

#A: 19, B: 16, C: 12, D: 6, E: 13, F: 20, G:21
sevseg_io = [19,16,12,6,13,20,21] #GPIO ports used for seven segment display
rgb1_io = [22,27,17]
rgb2_io = [25,24,23]
button_io = 18
numbers = [[19,16,12,6,13,20],
        [16,12],
        [19,16,6,13,21],
        [19,16,12,6,21],
        [16,12,20,21],
        [19,12,6,20,21],
        [19,12,6,13,20,21],
        [19,16,12],
        [19,16,12,6,13,20,21],
        [19,16,12,6,20,21]]

def init():
   setup_GPIO()
   reset()
   GPIO.output(rgb2_io[1], 1)
   GPIO.output(rgb1_io[0], 1)

def reset():
   flush_display()
   flush_led()

def flush_led():
   GPIO.output(rgb1_io, 0)
   GPIO.output(rgb2_io, 0)

def flush_display():
   GPIO.output(sevseg_io, 0)

def setup_GPIO():
   GPIO.setwarnings(False)
   GPIO.setmode(GPIO.BCM)
   GPIO.setup(button_io, GPIO.IN)
   GPIO.setup(rgb1_io, GPIO.OUT)
   GPIO.setup(rgb2_io, GPIO.OUT)
   GPIO.setup(sevseg_io, GPIO.OUT)
```

```
def change_color(rgb, r, g, b):
    GPIO.output(rgb, 0)
    GPIO.output(rgb[0], r)
    GPIO.output(rgb[1], g)
    GPIO.output(rgb[2], b)

def blink():
    for i in range(0,3):
        GPIO.output(rgb2_io[2], 1)
        sleep(0.5)
        GPIO.output(rgb2_io[2], 0)
        sleep(0.5)
    change_color(rgb2_io, 1, 0, 0)

def countdown():
    for i in range(9,-1,-1):
        GPIO.output(numbers[i], 1)
        if i <= 4 and i%2 == 0:
            change_color(rgb1_io, 0, 0, 1)
        elif i <= 4 and i%2 != 0:
            change_color(rgb1_io, 0, 0, 0)
        sleep(1)
        GPIO.output(numbers[i], 0)

def logic(pin):
    GPIO.output(rgb2_io[1], 0)
    GPIO.output(rgb2_io[2], 1)
    blink()
    change_color(rgb1_io, 0, 1, 0)
    countdown()
    change_color(rgb1_io, 1, 0, 0)
    change_color(rgb2_io, 0, 1, 0)
    sleep(10)

def polling_method(): # This is the polling method
    init()
    while True:
        if not GPIO.input(button_io):
            logic(0)

def interrupt_method(): # This is the interrupt method
    init()
    GPIO.add_event_detect(button_io, GPIO.RISING, logic)

#polling_method() # choose one of these to uncomment
#interrupt_method() # choose one of these to uncomment
```