2019-2 Embedded System Design Practice Lab #3

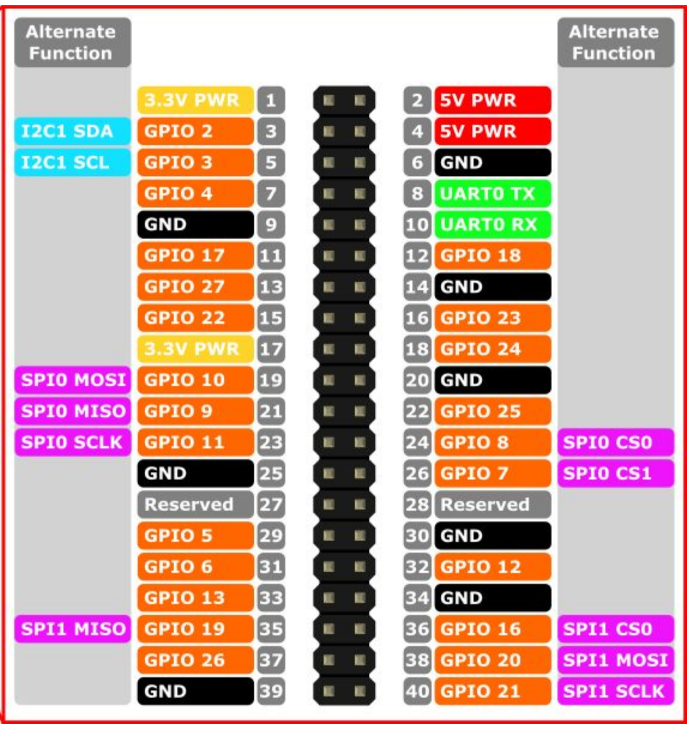
email: ysjoo@cs.kookmin.ac.kr

RPi GPIO control in C

NOTE:

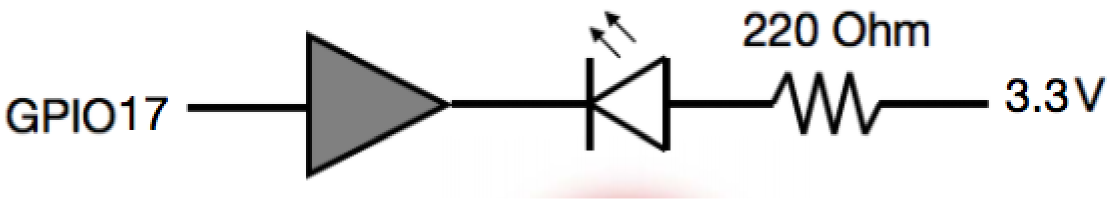
- **Be sure to take off your watches, rings, and any metal accessories before practice.**
- **When power is ON, changing wiring or inserting & removing components to the breadboard is strongly prohibited.**

- GPIO (General Purpose Input/Output) pin map



Alternate Function					Alternate Function
	3.3V PWR	1	2	5V PWR	
I2C1 SDA	GPIO 2	3	4	5V PWR	
I2C1 SCL	GPIO 3	5	6	GND	
	GPIO 4	7	8	UART0 TX	
	GND	9	10	UART0 RX	
	GPIO 17	11	12	GPIO 18	
	GPIO 27	13	14	GND	
	GPIO 22	15	16	GPIO 23	
	3.3V PWR	17	18	GPIO 24	
SPI0 MOSI	GPIO 10	19	20	GND	
SPI0 MISO	GPIO 9	21	22	GPIO 25	
SPI0 SCLK	GPIO 11	23	24	GPIO 8	SPI0 CS0
	GND	25	26	GPIO 7	SPI0 CS1
	Reserved	27	28	Reserved	
	GPIO 5	29	30	GND	
	GPIO 6	31	32	GPIO 12	
	GPIO 13	33	34	GND	
SPI1 MISO	GPIO 19	35	36	GPIO 16	SPI1 CS0
	GPIO 26	37	38	GPIO 20	SPI1 MOSI
	GND	39	40	GPIO 21	SPI1 SCLK

1. Make a prototype for the below circuit.



- Active low circuit: LED will be on with the GPIO output 0 (= 0V), off with GPIO output 1 (= 3.3V).

2. Use below Linux shell commands to check if the circuit is accurately built. The LED should be turned on and off as the write command is executed.

- **$ gpio readall**
- **$ gpio -g mode 17 out**
- **$ gpio readall**
- **$ gpio -g write 17 0**
- **$ gpio -g write 17 1**
- **$ gpio write 0 0**
- **$ gpio write 0 1**

- **IMPORTANT: your GPIO extension board (T-shape PCB) uses BCM numbering. However, the gpio command uses the wPi numbering by default. In order to use the BCM numbering, you should add -g option, for example:**
    - **gpio -g mode 17 out   ==   gpio mode 0 out**

3. Watch the below video.
LED breathing example
https://www.youtube.com/watch?v=ZT6siXyIjvQ
https://www.youtube.com/watch?v=gu4RfI4nzJA

LED PWM control
https://www.youtube.com/watch?v=jQ3JHknsM4o

4. Test below python code.

<onoff.py>

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)

on_time = 0.1 # time led is ON in seconds
off_time = 0.9 # time led is OFF in seconds

while True:
    GPIO.output(17, GPIO.HIGH)
    time.sleep(on_time)
    GPIO.output(17, GPIO.LOW)
    time.sleep(off_time)
```

- run vi or nano or any text editor you prefer.
- copy and paste the above code.
- $ python onoff.py

- Change the LED on/off_item value so that the LED brightness is reduced by ⅓.
- If you feel the LED is repeatedly turning ON and OFF, you may consider further reducing the period (= on_time + off_time).

5. Run the below python code.

<led_pwm.py>

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time

LedPin = 17

GPIO.setmode(GPIO.BCM)       # Use BCM numbering
GPIO.setup(LedPin, GPIO.OUT)   # Set pin mode as output
GPIO.output(LedPin, GPIO.LOW)  # Set pin to low(0V)

p = GPIO.PWM(LedPin, 1000)     # set Frequency to 1KHz
p.start(0)                     # Start PWM output, Duty Cycle = 0

try:
        while True:
                for dc in range(0, 101, 2):   # Increase duty cycle: 0~100
                        p.ChangeDutyCycle(dc)     # Change duty cycle
                        time.sleep(0.01)
                time.sleep(0.05)
                for dc in range(100, -1, -2): # Decrease duty cycle: 100~0
                        p.ChangeDutyCycle(dc)
                        time.sleep(0.01)
                time.sleep(0.05)
except KeyboardInterrupt:
        p.stop()
        GPIO.output(LedPin, GPIO.HIGH)    # turn off all leds
        GPIO.cleanup()
```

6. Modify and compile the below C code to make your LED breathing.
< led_pwm.c>

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <string.h>
#include <sys/time.h>

const char *PATH_GPIO_EXPORT = "/sys/class/gpio/export";
const char *PATH_GPIO_UNEXPORT = "/sys/class/gpio/unexport";
const char *PATH_GPIO_17_DIRECTION = "/sys/class/gpio/gpio17/direction";
const char *PATH_GPIO_17_VALUE = "/sys/class/gpio/gpio17/value";

#define GPIO_NUM 17

typedef enum {
	OFF = 0,
	ON
} gpio_state ;

void gpio_init();
void gpio_exit();
void set_gpio_state(gpio_state state);
void delay_micro(int delay_micros);

FILE *GPIO_EXPORT;
FILE *GPIO_17_DIRECTION;
FILE *GPIO_17_VALUE;

int main()
{
	int time = 0;
	gpio_state state = OFF;

	gpio_init();

/*************** insert your code here *****************/
	while(1) {
		set_gpio_state(ON);
		delay_micro(1000000); // 1 sec delay
		set_gpio_state(OFF);
		delay_micro(1000000); // 1 sec delay
	}

/*****************************************************/

	gpio_exit();
}

void gpio_init()
{
	if ((GPIO_EXPORT = fopen(PATH_GPIO_EXPORT, "w")) == NULL) {
		printf("%s open failed\n", PATH_GPIO_EXPORT);
		exit(0);
	}

	fprintf(GPIO_EXPORT, "%d", GPIO_NUM);
	fclose(GPIO_EXPORT);

	if ((GPIO_17_DIRECTION = fopen(PATH_GPIO_17_DIRECTION, "w")) == NULL)
{
```

```
                printf("%s open failed\n", PATH_GPIO_17_DIRECTION);
                exit(0);
        }
        fprintf(GPIO_17_DIRECTION, "out");
        fclose(GPIO_17_DIRECTION);

        if ((GPIO_17_VALUE = fopen(PATH_GPIO_17_VALUE, "w")) == NULL) {
                printf("%s open failed\n", PATH_GPIO_17_VALUE);
                exit(0);
        }
}

void gpio_exit()
{
        FILE *GPIO_UNEXPORT;

        fclose(GPIO_17_VALUE);

        if ((GPIO_UNEXPORT = fopen(PATH_GPIO_UNEXPORT, "w")) == NULL) {
                printf("%s open failed\n", PATH_GPIO_UNEXPORT);
                exit(0);
        }
        fprintf(GPIO_UNEXPORT, "%d", GPIO_NUM);
        fclose((int) GPIO_UNEXPORT);
}


void set_gpio_state(gpio_state state)
{
        fprintf(GPIO_17_VALUE, "%d", state);
        fflush(GPIO_17_VALUE);
}


void delay_micro(int delay_micros)
{
        struct timeval now, pulse;
        int cycles, micros;

        cycles = 0;
        gettimeofday(&pulse, NULL);
        micros = 0;
        while (micros < delay_micros) {
                ++cycles;
                gettimeofday(&now, NULL);
                if (now.tv_sec > pulse.tv_sec)
                        micros = 1000000L;
                else
                        micros = 0;
                micros = micros + (now.tv_usec - pulse.tv_usec);
        }
}
```

```
$ sudo bash
$ gcc -o led_pwm.out led_pwm.c
$ ./led_pwm.out
```