Lab. GPIO and mmap

Assignment 2

Functional Requirements

Write a program that meets the following functional requirements.

1. **When the program starts**:
   - A status LED (i.e., LED0) should turn on to indicate the program is running.
   - The program should continue running for 30 seconds or more.
2. **At the end of the program**:
   - All LEDs must be turned off.
3. **During the execution**:
   - Two LEDs (i.e., LED1 & LED2) serve as outputs designated to specific switches.
     - When the designated **"ON switch"** is pressed, **Turn on LED1 and LED2**.
     - When the designated **"OFF switch"** is pressed, **Turn off LED1 and LED2**.
     - When the designated **toggle switch for LED1** is pressed, **Toggle LED1 state**.
     - When the designated **toggle switch for LED2** is pressed, **Toggle LED2 state**.

NOTE: Switch input should be edge-triggered only (no repeated toggling on hold).

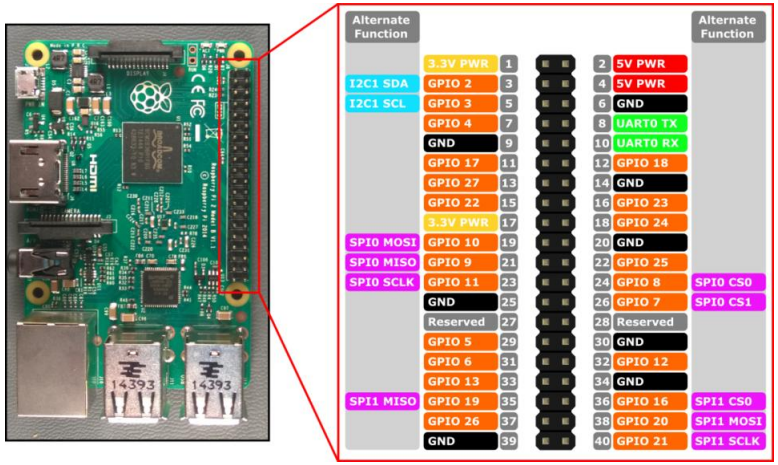
Implementation

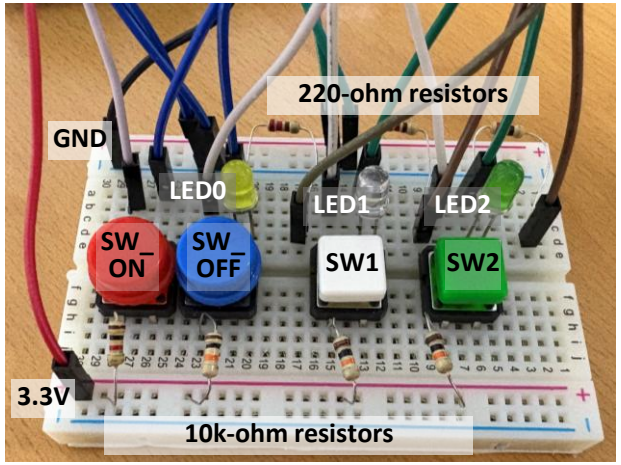
- **You may either:**
  1. build your circuit to match the provided example code, or
  2. modify the code to match your own custom circuit. (without wiringPi)

- **Implementation note:**
  - **<u>Circuit details are intentionally not provided.</u>**
    - The circuit should be designed as part of the assignment.
    - You may freely choose which GPIO pins to use based on your circuit configuration.
  - **<u>The use of wiringPi is strictly prohibited.</u>**
    - You must use the custom GPIO header and register-level access as introduced in class.
  - **<u>Switch input should be edge-triggered only.</u>**
    - Holding a button must not result in repeated toggles.

Circuit Configuration

- Use **4 switches as inputs** and **3 LEDs as outputs**.
- Configure each **switch with a pull-up resistor**.
  - pressing the switch makes the GPIO input read as LOW.
- The example code is provided for reference, and **your pin assignments may differ** from the example.





Example code: ‘assignment2.c’ (1/3)

```c
#include <stdio.h>      // Standard input/output library
#include <time.h>       // Standard time library
#include <stdlib.h>     // Standard library functions
#include <unistd.h>     // POSIX API (close, read, write, etc.)
#include <fcntl.h>      // File control operations (open, O_RDWR, etc.)
#include <sys/mman.h>   // Memory management functions (mmap, munmap, etc.)
#include "gpio.h"       // Custom GPIO library header

// Constants ------------------------------------------------------------------------------------
#define SW_NUM   4
#define SW_ON    26
#define SW_OFF   19
#define SW1      13
#define SW2      6

#define LED_NUM  3
#define LED0     21
#define LED1     20
#define LED2     16

#define RUNTIME  30     // Program runtime in seconds


// Type Definitions -----------------------------------------------------------------------------
typedef struct {
    const int *pins;
    int count;
} GpioList;
```

Please refer to the lab materials for “gpio.h” and “gpio.c”

```
$ gcc assignment2.c gpio.c –o assignment2
$ sudo ./assignment2
```

Example code: ‘assignment2.c’ (2/3)

```c
// Function Prototypes ---------------------------------------------------------------------------
void setGPIO(GpioList inputs, GpioList outputs); // Initializes input and output GPIO pins.
void handleInputs(GpioList inputs, int state[]); // Handles switch inputs and detects press (rising edge).
void handleStatus(int pin_no);                   // Executes predefined actions based on switch input.
void clearOutputs(GpioList outputs);             // Turns off all output GPIO pins (sets to LOW).


// Main Function (Refer to the assignment for descriptions) ---------------------------------------
int main(void) {
    // Initialize switch state tracking (all initially HIGH)
    int prev_states[SW_NUM] = {HIGH, HIGH, HIGH, HIGH};

    // Define GPIO pin lists for switches and LEDs
    const int switch_pins[SW_NUM] = {SW_ON, SW_OFF, SW1, SW2};
    const int led_pins[LED_NUM]   = {LED0, LED1, LED2};

    // Wrap pin arrays into GpioList structures
    GpioList inputs  = {switch_pins, SW_NUM};
    GpioList outputs = {led_pins,    LED_NUM};

    // Set GPIO modes and turn on LED0 as status indicator
    setGPIO(inputs, outputs);
    digitalWrite(LED0, HIGH);

    // Start runtime loop for fixed duration
    time_t start = time(NULL);
    while (difftime(time(NULL), start) < RUNTIME) {
        handleInputs(inputs, prev_states);
    }

    // Turn off all outputs on exit
    clearOutputs(outputs);

    return 0;
}
```

Please refer to the lab materials for “gpio.h” and “gpio.c”

```
$ gcc assignment2.c gpio.c –o assignment2
$ sudo ./assignment2
```

Example code: ‘assignment2.c’ (3/3)

```
68  // Function Definitions ---------------------------------------------------
69  void setGPIO(GpioList inputs, GpioList outputs) {
70      // Initialize wiringPi with BCM GPIO numbering
71      wiringPiSetupGpio();
72
73      // Set input pin modes
74      for (int i = 0; i < inputs.count; i++) {
75          pinMode(inputs.pins[i], INPUT);
76      }
77
78      // Set output pin modes
79      for (int i = 0; i < outputs.count; i++) {
80          pinMode(outputs.pins[i], OUTPUT);
81      }
82  }
83
84  void clearOutputs(GpioList outputs) {
85      // Set all output pins to LOW
86      for (int i = 0; i < outputs.count; i++) {
87          digitalWrite(outputs.pins[i], LOW);
88      }
89  }
90
```

Please refer to the lab materials for the definition of the following functions: **“wiringPiSetGpio”, “pinMode”, “digitalWrite”,** and **“digitalRead”.**

IMPORTANT NOTE:

- **Do NOT Use “wiringPi.h”.**
- **USE the custom GPIO header**, “gpio.h” & “gpio.c”.

```
91  void handleInputs(GpioList inputs, int state[]) {
92      // Poll each input switch and detect edge transition (HIGH → LOW)
93      for (int i = 0; i < inputs.count; i++) {
94          int curr = digitalRead(inputs.pins[i]);
95          if (state[i] == HIGH && curr == LOW) {
96              handleStatus(inputs.pins[i]);   // Handle action on press
97          }
98          state[i] = curr;    // Update stored state
99      }
100     delay(50);  // Debouncing
101 }
102
103 void handleStatus(int pin_no) {
104     // Perform predefined actions
105     if (pin_no == SW_ON) {
106         digitalWrite(LED1, HIGH);   // Turn on all LED1 & LED2
107         digitalWrite(LED2, HIGH);
108     }
109     else if (pin_no == SW_OFF) {
110         digitalWrite(LED1, LOW);    // Turn off all LED1 & LED2
111         digitalWrite(LED2, LOW);
112     }
113     else if (pin_no == SW1) {
114         digitalWrite(LED1, !digitalRead(LED1)); // Toggle LED1
115     }
116     else if (pin_no == SW2) {
117         digitalWrite(LED2, !digitalRead(LED2)); // Toggle LED2
118     }
119 }
120
```

Please refer to the lab materials for “gpio.h” and “gpio.c”

```
$ gcc assignment2.c gpio.c –o assignment2
$ sudo ./assignment2
```

Deliverables

- To receive credit:
  - You must have the actual operation of their program verified by me during class time before the submission deadline of the assignment.
- Please submit:
  - A video of your program running (with the circuit)
  - A zipped folder with all source files (including headers)
    - e.g. team0.zip
      - team0
        - assignment2.c
        - gpio.c
        - gpio.h
        - …