

# GPIO - LED and Switch Interface

## Objective

The objective of the assignment is to access microcontroller's port-pins to manipulate LEDs that are connected to a few on-board LEDs of the [SJ2 board](#).

Please [reference this article](#) for good hints related to this assignment.

---

## Part 0:

In this part, we will setup basic skeleton of the code before we access the memory map to manipulate an LED.

```
// file: main.c
#include <stdio.h>
// Separate include files for Clang to sort separately
#include "delay.h"
void main(void) {
    unsigned counter = 0;

    while (1) {
        printf("Running: %u\n", counter);
        ++counter;
        delay__ms(1000U);
    }
}
```

---

## Part 1:

In this part, we will actually reference the LPC user manual and manipulate one of the on-board LEDs.

You can refer Chapter 8 - Table 94 of the user manual.

```
// file: main.c
#include <stdio.h>
// Separate include files for Clang to sort separately
// Add header files required
#include "delay.h"
void blinky_leds(void){
    // 1. Refer to the datasheet and configure the direction and pin using the memory address
    uint32_t *port1_pin_register = (uint32_t *) (MEMORY_ADDRESS);
    ...
    // Set directions to the pins
    ...
    while (1) {
        delay_ms(100);
        //METHOD1: Use bitmasking techniques to set and reset the pins
        //METHOD2: Use the set and clear registers
    }
}
int main(void)
{
    blinky_leds();
    return 0;
}
```

---

## Part 2:

In this part, we will use the LPC memory map to manipulate the on-board LEDs. This will reduce us from cross referencing the LPC user manual.

Use [LPC40xx MCU Memory Map](#) as reference

```
// file: main.c
#include <stdio.h>
// Separate include files for Clang to sort separately
// Add header files required
#include "board_io.h"
#include "delay.h"
#include "gpio.h"
#include "lpc40xx.h"
void blinky_leds(void) {
    // Step 1: Choose pin as GPIO
    LPC_IOCON->(port_pin_number) &= ~7;

    // Step 2: Enable the direction pin
    while (1) {
        delay__ms(100);
        // Use set and clear registers to set and clear pins accordingly
    }
}
```

---

## Part 3:

Create a comprehensive GPIO driver and manipulate the on-board LED's using the on-board switches. You could use [PART1: GPIO Driver](#) as reference.

---

## Extra Credit:

Do something creative with your implemented GPIO driver. You could have the LED blink three times when the switch is pressed.

---

## Go above and beyond

Want to have even more fun and experience multi-tasking functionality of your board's software package? Then follow this link:

- [GPIO Assignment](#)
-

Revision #13

Created 4 years ago by [sree harsha](#)

Updated 1 month ago by [sree harsha](#)