Exercise #03 REPORT

Name: 송예린

Student ID: 2171023

1. **Exercise Objective (10%)**

GPIO 는 Memory Map 을 통해서 input, output 을 Control 한다. 포인터의 개념을 이해한다면 임베디드 시스템의 General Purpose input/output 을 어떻게 조합해서 바꿔야 동작하는 지도 이해할 수 있을 것이다. 포인터와 Memory Map 을 활용해서 GPIO 를 제어해본다면 더 잘 이해할 수 있을 것이라고 생각된다. LED 가 색상이 바뀌며 깜빡이는 실습을 통해 GPIO 를 Control 할 예정이다.

2. **Implementation (60%)**

1) Port P1 에서 빨간색 LED 가 깜빡이는 동안 Port P2 는 빨간색, 초록색, 파란색 순대로 LED 가 깜빡이는 실습

P1, P2 별로 gpio_dir, gpio_out 포인터를 선언하고 direction Base address 값을 포인터가 가리키게 한다. 이때 direction Base address 은 Reference Manual 에서 offset 이 몇 byte 만큼 떨어져 있는 지를 보고 사용할 수 있다. 값 0 번째 핀을 output 으로 사용하기 위해서 Port1 direction register 값을 00000001 로 세팅해줘야 한다 포인터 gpio_p1dir 를 이용하여 해당 주소에 0x01(16 진수)을 넣어준다. While 문 안에 p1 과 마찬가지로 *gpio_p2dir=0x01 을 세팅 후 포인터 gpio_p1out, gpio_p2out 로 output register Baes address 값에 0x01 을 넣어줘서 빨간 색 불이 켜지게 한다. for 문으로 딜레이를 주고 포인터 gpio_p1out, gpio_p2out 로 output register Baes address 값에 0x00 으로 바꿔서 불이 꺼지게 한다. P1 은 LED 의 색상이 변하지 않으므로 while 문 밖에서 p1dir=0x01 으로 셋팅 후 while 문 안에서 *gpio_p1out=0x00 – 딜레이(for 문) - *gpio_p1out=0x01 을 반복하여 깜빡이게 한다. P2 는 output 으로 사용하는 핀 번호가 계속 바뀌므로 while 문 안에 *gpio_p2dir 을 반복적으로 작성하고 값을 바꿔줘야 한다. 빨간색은 00000001 -> 0x01 초록색은 00000010 -> 0x02 파란색은 00000100 -> 0x04 이다. 해당 값을 dir 주소에 세팅 후 안에서 *gpio_p2out 로 output register 주소값에 해당 값을 넣어준다. for 문으로 딜레이를 준 후 *gpio_p1out=0x00 으로 불이 꺼지게 한다. 다시 for 문으로 딜레이를 준 후 초록색, 파란색의 값을 넣어서 같은 방식으로 작성한다.

3. Discussion & Conclusion (10%)

P1, P2 에 LED 색상이 따로 변하는 실습을 통해 GPIO 를 제어해보았다. 포인터로 주소값에 접근하고 direction Base address 값에 1byte 중 핀 번호에 해당하는 bit 를 1 로 바꾸고 다시 0 으로 변경하는 방식으로 LED 를 깜빡였다.

어려웠던 부분은 주소 값에 접근하여 값을 변경한다는 개념이 잘 와닿지 않았다 처음에 while 문 밖에서 *gpio_p2dir=0x01, gpio_p2dir=0x02, gpio_p2dir=0x04;로 작성하였는데 파란색 불만 깜빡였었다 그 이유를 생각해보니 포인터는 주소 값에 접근하여 값을 변경하는데 같은 주소를 가리키는 포인터로 값을 변경했기 때문에 dir 주소에 0x04 만 남았던 것이다. 코드를 다시 보니 포인터를 선언할 때 4 개의 포인터(p1 dir, p1 output, p2 dir, p2 output) 만 선언 후 Base address 값을 가리키게 했다면 더 간소화할 수 있었을 것 같다. 같은 주소값을 가리키고 있기 때문에 여러 포인터를 선언할 필요는 없다고 생각한다.

이 실습을 통해 General Purpose input/output 이 어떻게 동작하는 지 직접적으로 이해할 수 있었고 포인터에 대해 다시 정리할 수 있었다. 개인적으로 매우 흥미로운 실험이었다.

4. Reference(s)

5. Code (20%)

FILE NAME (ex: main.c)

```
        WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;// stop watchdog timer
        volatile unsigned char *gpio_p2dir1, *gpio_p2out1, *gpio_p2dir2, *gpio_p2out2, *gpio_p2dir3,
*gpio_p2out3, *gpio_p1out,*gpio_p1dir;
        int i;
        gpio_p2dir1=(unsigned char*)(0x40004C05);
        gpio_p2out1=(unsigned char*)(0x40004C03);
    gpio_p2dir2=(unsigned char*)(0x40004C05);
    gpio_p2out2=(unsigned char*)(0x40004C03);
    gpio_p2dir3=(unsigned char*)(0x40004C05);
    gpio_p2out3=(unsigned char*)(0x40004C03);
    gpio_p1dir=(unsigned char*)(0x40004C04);
    gpio_p1out=(unsigned char*)(0x40004C02);
    *gpio_p1dir=0x01;
        while(1){


            *gpio_p2dir1=0x01;
            *gpio_p2out1=0x01;
        *gpio_p1out=0x01;
            for(i=0;i<100000; i++){
                ;
```

```
        }
        *gpio_p2out1=0x00;
    *gpio_p1out=0x00;
        for(i=0;  i<100000; i++){
            ;
        }
    *gpio_p2dir2=0x02;
    *gpio_p2out2=0x02;
    *gpio_p1out=0x01;
    for(i=0;i<100000;  i++){
        ;
    }
    *gpio_p2out2=0x00;
    *gpio_p1out=0x00;
    for(i=0;  i<100000; i++){
        ;
    }
    *gpio_p2dir3=0x04;
    *gpio_p2out3=0x04;
    *gpio_p1out=0x01;
    for(i=0;i<100000;  i++){
        ;
    }
    *gpio_p2out3=0x00;
    *gpio_p1out=0x00;
    for(i=0;  i<100000; i++){
        ;
    }
     }
```