강의명: 임베디드 시스템
숙제 번호: 1
숙제 제목: Digital I/O programming(디지털 입출력)
학생 이름: 한규현

1. 프로그램 led1-flash

1.1 프로그램 코드 쓰기

```
#include  "mbed.h"
DigitalOut  led1 (LED1);                // led1 = LED1

int  main ()
{
    while  (true) {
        led1  = 0 ;                     // on LED1
        thread_sleep_for (500);         // wait 500 ms
        led1  = 1 ;                     // off LED1
        thread_sleep_for (500);         // wait 500 ms
    }
}
```

1.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기
LED를 조작하는 Digital input/output은 Class DigitalIn/Out 두 클래스가 관리한다. 본 과제에서는 출력을 하면 되므로 DigitalOut의 클래스를 사용했고, 이번 문항의 경우 led1을 제어하는 것이므로 DigitalOut led1 (LED1); 이라고 코드를 작성했다.
FRDM-K64F의 내부 LED1은 common anode 방식을 따르기 때문에 LOW일 때 led가 켜지게 된다. 이러한 이유로 led1 = 0 일 때 LED1이 켜지고 led1 = 1일 때 LED1이 꺼지도록 프로그램을 작성했다. LED1이 ON/OFF 하는 과정에서 0.5sec wait을 주기 위해 thread_sleep_for(500)을 넣었고, 그 과정을 무한 반복하기 위해 while(true)와 같이 코드를 작성했다.

1.3 하드웨어 구성 사진 첨부하기

1.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기

https://youtu.be/ZcEyG8lOY8g

2. 프로그램 led1-flash-sos

2.1 프로그램 코드 쓰기

```
#include  "mbed.h"

DigitalOut  led1 (LED1);                // led = LED1

int  main ()
{
    int  i ;

    while  (true) {
        for (i  = 0 ; i  < 3 ; i ++){
            led1  = 0 ;
            thread_sleep_for (300);
            led1  = 1 ;
            thread_sleep_for (300);
        }
        thread_sleep_for (1000);

         for (i  = 0 ; i  < 3 ; i ++){
            led1  = 0 ;
            thread_sleep_for (1000);
            led1  = 1 ;
            thread_sleep_for (300);
        }
        thread_sleep_for (1000);
    }
}
```

2.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기

앞선 1번 문제에서와 같이 LED1 = 0 이면 켜지고, LED1 = 1이면 꺼진다. 첫 번째 for문은 S를 표현한다. 0.3초 짧게 LED를 켜고, 0.3초 짧게 LED를 끄도록 thread_sleep_for(300)을 사용해서 코드를 작성했다. S 출력이 끝나면 thread_sleep_for(1000)을 실행해서 사이에 텀을 두게 코드를 작성했다. 이후 for문으로 O를 출력하도록 프로그램을 작성했고, 이 for에서는 앞선 for과 동일한 로직이지만, 켜지는 시간을 1.0sec로 해서 O를 표현했다. while(true)로 SOS가 계속해서 출력되도록 했다.

2.3 하드웨어 구성 사진 첨부하기

2.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기

https://youtu.be/Fevxt45JaLY

3. 프로그램 led-d0-flash

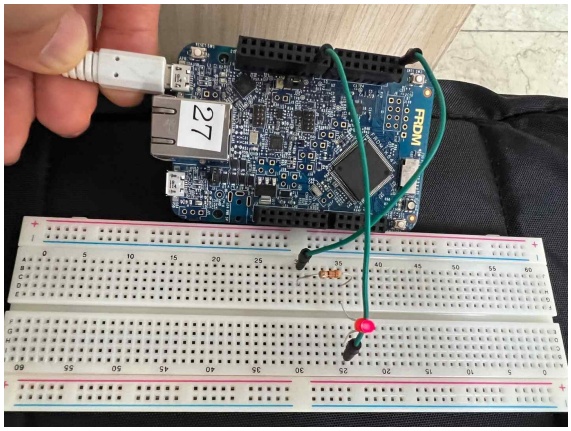
3.1 프로그램 코드 쓰기

```
#include  "mbed.h"

DigitalOut  led (D0);                 // led = LED1

int  main ()
{
    while  (true) {
        led  = 1 ;                   //on LED
        thread_sleep_for (500);  // wait 500 ms
        led  = 0 ;                    // off LED
        thread_sleep_for (500);  // wait 500 ms
    }
}
```

3.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기

D0 pin에 LED를 연결하여 깜빡이게 해야하므로 Digital led(D0); 이라고 작성했다. 그리고 common cathode 방식을 사용했기 때문에 led = 0 일 때 꺼지고, led = 1일 때 켜지도록 프로그램을 작성했다. thread_sleep_for(500)을 led 변수의 값이 바뀔 때마다 작성함으로써 켜고 꺼짐 사이 시간을 0.5sec로 구성했다.

3.3 하드웨어 구성 사진 첨부하기



3.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기
https://youtu.be/mPFHkJKEFmU

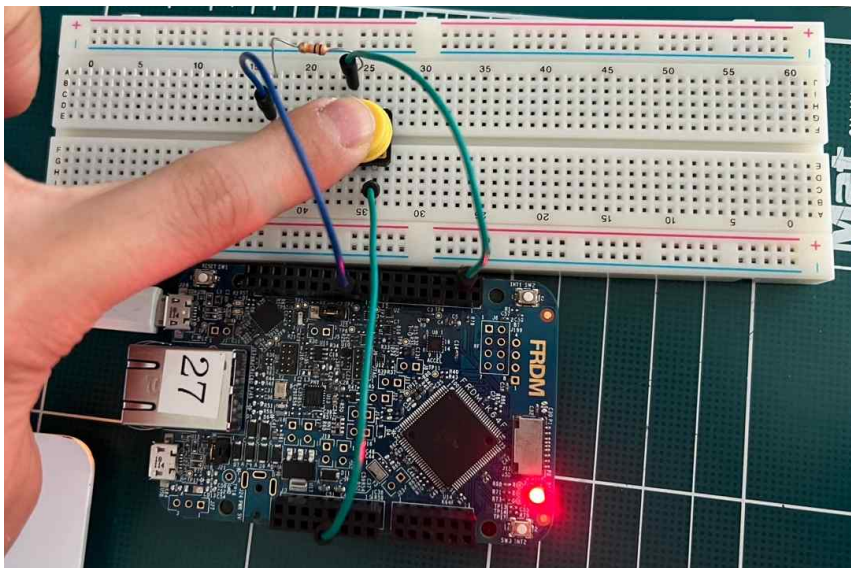
4. 프로그램 switch-d1-input

4.1 프로그램 코드 쓰기

```
#include "mbed.h"
DigitalIn sw(D1, PullDown);          // sw = D1(PullDown)
DigitalOut led1(LED1);               // led1 = LED1
int main()
{
    while (true) {
        if(sw == 1) {
            led1 = 0;
        }   else {
            led1 = 1;
        }
    }
}
```

4.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기
이번에는 외부에서 버튼으로 Input을 받아야하기 때문에 DigitalIn sw(D1, PullDown);이라고 코드를 작성해서 스위치가 눌리는 Input을 받도록 했다. 이전 문제들과 마찬가지로 LED를 제어하기 때문에 DigitalOut led1(LED);를 작성해 Output 출력을 했다. sw변수는 Input을 위한 변수이고 구체적으로 sw = 1이면 버튼이 눌린 상황으로 led1 = 0 으로 불이 켜지게 했다. 만약 그렇지 않은 상황에서는 led1 = 1로 설정해서 불이 들어오지 않도록 프로그램을 작성했다. Input은 언제 들어올지 모름으로 while(true)로 작성해 계속 Input을 받을 수 있도록 했다.

4.3 하드웨어 구성 사진 첨부하기



4.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기

https://youtu.be/DvkbcLwqkYk

5. 프로그램 led-with-switch

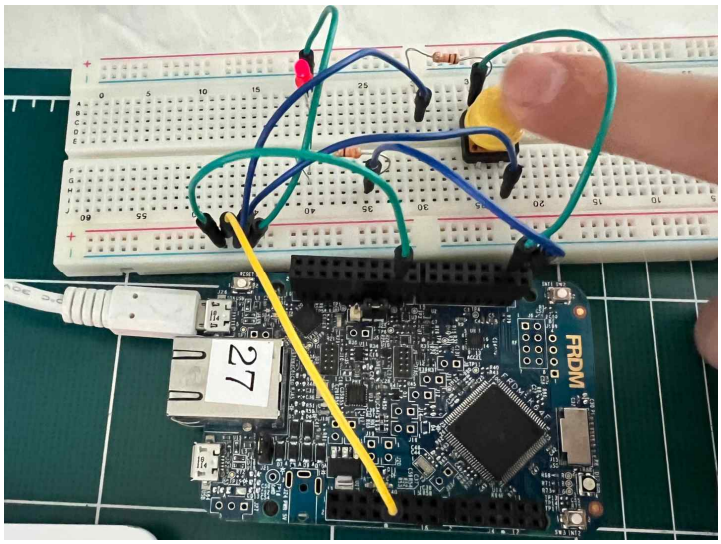
5.1 프로그램 코드 쓰기

```
#include "mbed.h"

DigitalOut led(D0);                 // led = LED1
DigitalIn sw(D1);
int main()
{
    while (true) {
        if(sw == 1) {
            led = 1;
            thread_sleep_for(200);
            led = 0;
            thread_sleep_for(200);
            led = 1;
            thread_sleep_for(200);
            led = 0;
        }
    }
}
```

5.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기

DigitalOut 으로 D0 pin에 연결된 LED를 출력하고, DigitalIn으로 D1 pin에 연결된 버튼 스위치에 입력을 받도록 한다. 이때 버튼 스위치가 눌리게 되면 LED가 두 번 켜져야함으로 버튼 스위치가 눌리는 상태인 sw == 1이 되면, led = 1로 LED가 0.2초 켜졌다 꺼졌다를 2번 반복하도록 코드를 작성했다. switch의 input이 언제 들어올지 모르기 때문에 while(true)로 작성하여 input을 상시로 받도록 했다.

5.3 하드웨어 구성 사진 첨부하기



5.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기

https://youtu.be/aubuPSnJoR0

6. 프로그램 7-segment

6.1 프로그램 코드 쓰기

```
#include "mbed.h"
// display = bus of D0-D7
BusOut display(D0, D1, D2, D3, D4, D5, D6, D7);
int main()
{
    while (true) {
        for(int i =0; i <=16; i++){
            switch (i){
                case 0:
                display =0x3F;
                break;
```

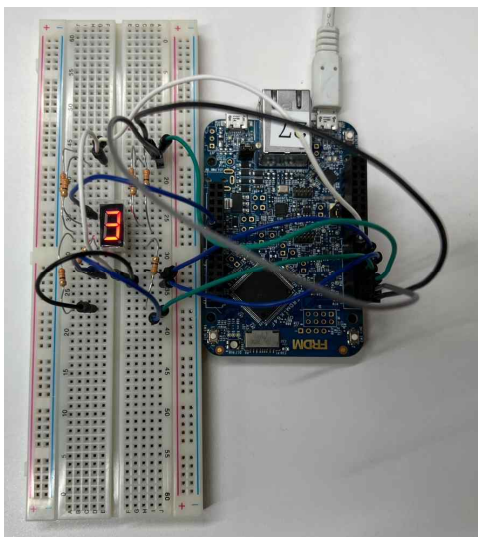
```
            case 1:
            display =0x06;
            break;
            case 2:
            display =0x5B;
            break;
            case 3:
            display =0x4F;
            break;
            case 4:
            display =0x66;
            break;
            case 5:
            display =0x6D;
            break;
            case 6:
            display =0x7D;
            break;
            case 7:
            display =0x07;
            break;
            case 8:
            display =0x7F;
            break;
            case 9:
            display =0x6F;
            break;
            case 10:
            display =0x77;
            break;
            case 11:
            display =0x7C;
            break;
            case 12:
            display =0x39;
            break;
            case 13:
            display =0x5E;
            break;
```

```
                case 14:
                display =0x79;
                break;
                case 15:
                display =0x71;
                break;
                default:
                break;
            }
            thread_sleep_for(1000);
        }
    }
}
```

6.2 프로그램 작성 아이디어 혹은 이유 설명 쓰기

BusOut이란 Class를 활용해 D0~D7에 연결된 8개의 LED를 관리하도록 했다. i = 0부터 계속 증가하면서 0-F까지의 수를 출력할 수 있도록 case문으로 프로그램을 작성했다. 모든 출력이 마치게 되면 1초의 정지 시간을 가진 뒤, 다시 0-F까지 출력될 수 있도록 프로그램을 작성했다.

6.3 하드웨어 구성 사진 첨부하기



6.4 프로그램 수행 사진/동영상(Youtube 링크) 첨부하기

https://youtu.be/Dl6YTD5W_Wc

끝.