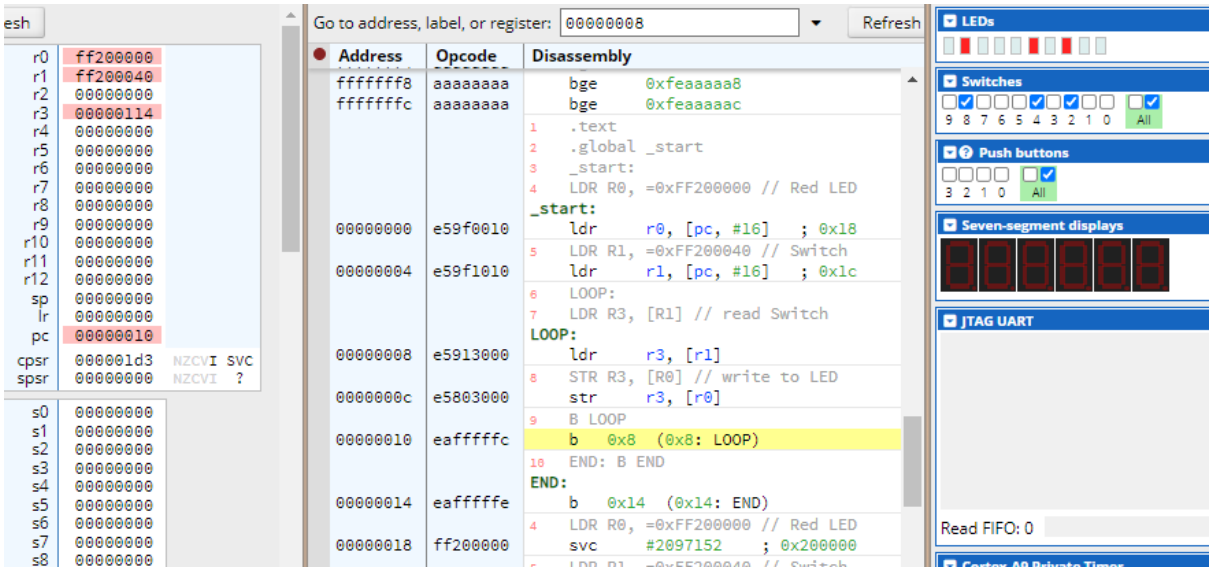
Hw5

2017253019 안희영

1.



LDR R0, =0xFF200000 // r0에 0xFF200000(RedLed출력값 주소)를 저장

LDR R1, =0xFF200040 // r1에 0xFF200040(스위치 입력값 주소) 저장

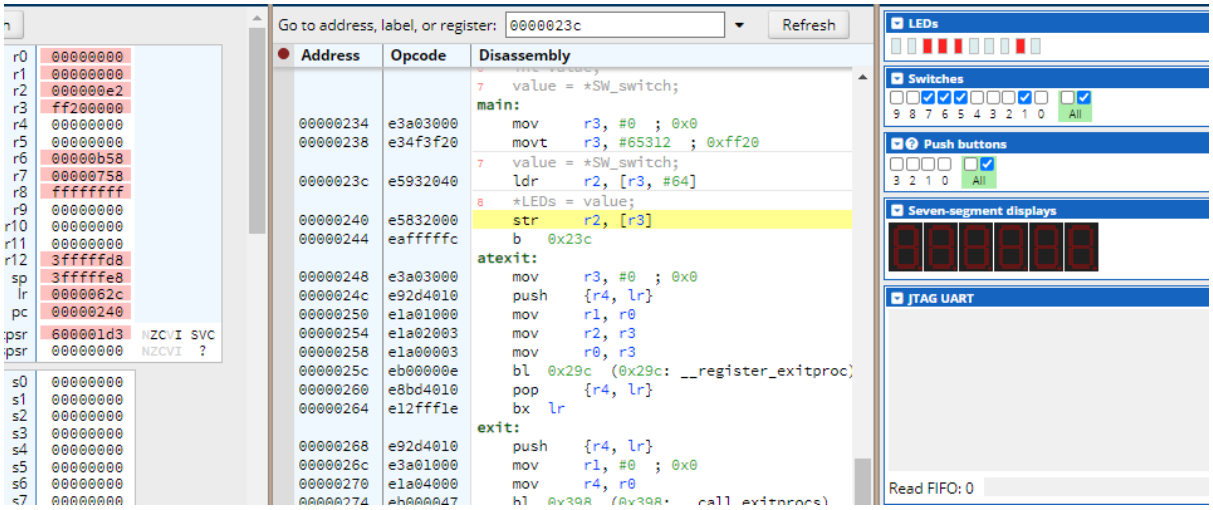
LOOP: //루프시작점

LDR R3, [R1] // 주소가 r1에 저장된값(스위치의 입력값) 을 r3에 불러옴

STR R3, [R0] // r3에 저장된 값을 메모리주소가 r0값인곳에 저장

B LOOP // loop로 이동

2.



volatile int *LEDs = (int *) 0xFF200000; //Red Led의 주소값을 저장한 포인터

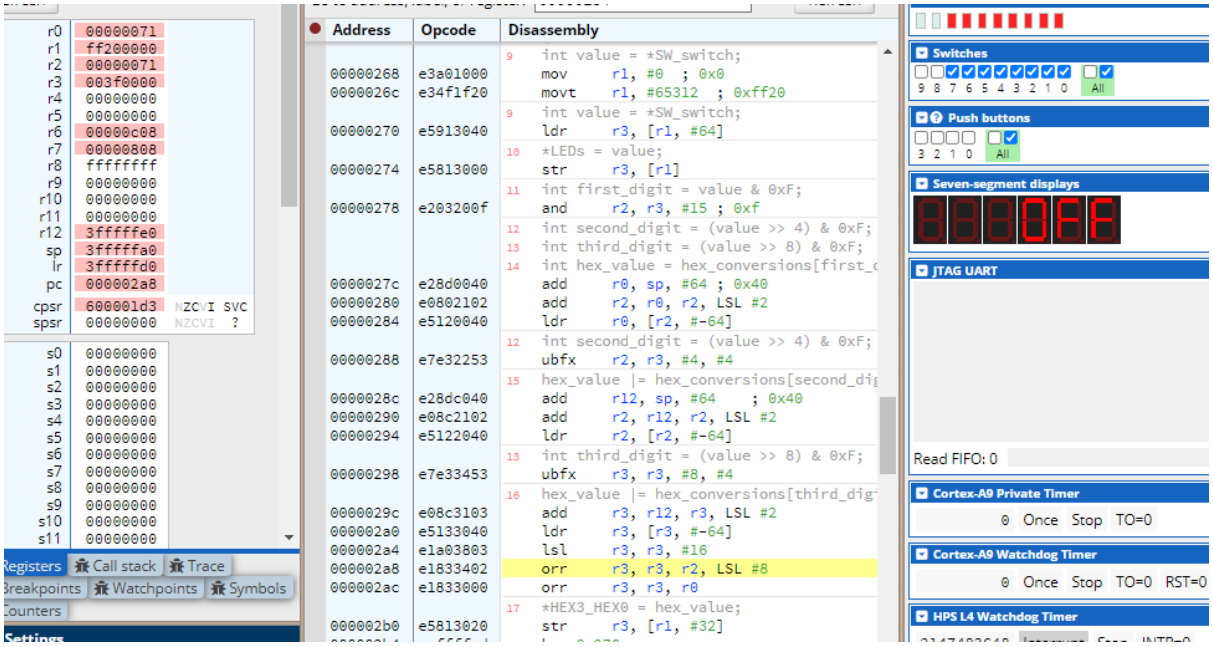
volatile int *SW_switch = (int *) 0xFF200040; // 스위치 입력값의 주소를 저장한 포인터

while(1)//이하반복

value = *SW_switch; //스위치 입력 포인터에 저장된 값을 value에 저장

*LEDs = value; // led 주소의 값을 value 값으로 할당

3.



volatile int *LEDs = (int *) 0xFF200000; // led의 출력값 포인터

volatile int *HEX3_HEX0 = (int *) 0xFF200020; //hex led 출력값 포인터

volatile int *SW_switch = (int *) 0xFF200040; //스위치 입력값 주소 포인터

int hex_conversions[16] = {0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x67, 0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71}; //각 숫자 인덱스에 따른 hex 출력 (0~F까지)

int value = *SW_switch; //스위치값 입력

*LEDs = value; //led로 스위치 입력값 출력

int first_digit = value & 0xF; // hex 출력값 첫번째 자리수 (f와 and연산으로 4비트만 남김)

int second_digit = (value >> 4) & 0xF; // value를 4비트 쉬프트하여 f와 and연산. Hex 두번째 출력값 저장

int third_digit = (value >> 8) & 0xF;//value를 8비트 쉬프트하고 f와 and연산하여 세번째 자리 저장

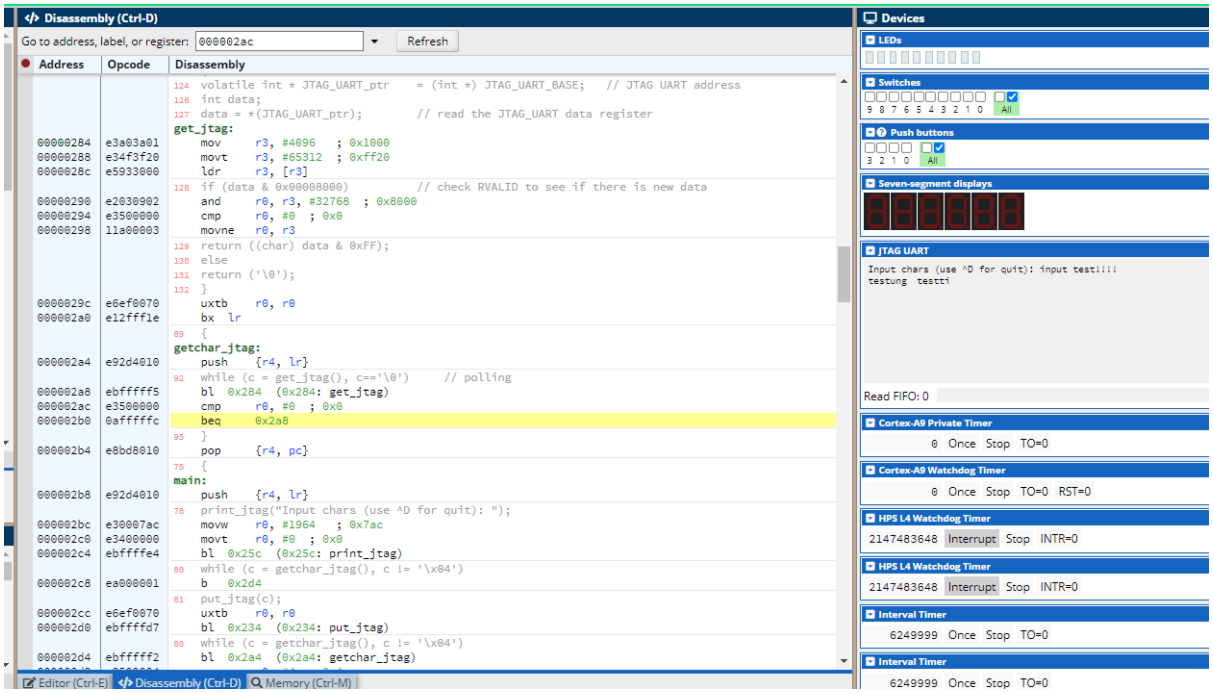
int hex_value = hex_conversions[first_digit]; // 첫째자리에 맞는 hex출력값 저장

hex_value |= hex_conversions[second_digit] << 8;//둘째 자리의 hex led 출력값을 8비트 쉬프트하여 or연산으로 추가함.

hex_value |= hex_conversions[third_digit] << 16; //셋째 자리의 hex led 출력값을 16비트 쉬프트후 or 연산으로 추가.

*HEX3_HEX0 = hex_value;//hex led 출력 주소에 최종 hex 출력값을 저장

3.



DATA: 실제 송수신 데이터

RAVAIL : 수신 FIFO 저장 문자 수

RVALID: 읽어올 데이터 유무 확인

WSPACE : 송신 FIFO 가용 공간

Data 레지스터에는 인수 c의 값(입,출력값)이 저장되어 전달됨

```
if (data & 0x00008000)                  // check RVALID to see if there is new data
                return ((char) data & 0xFF);
```

//rvalid값이 1이라면 data 레지스터 값(1바이트)만을 읽어옴.

```
if (control & 0xFFFF0000)
```

// wspace를 확인하여 출력 가능한 빈 공간이 있다면 출력, 그렇지 않다면 출력하지 않음

RAVAIL: 사용되지 않음