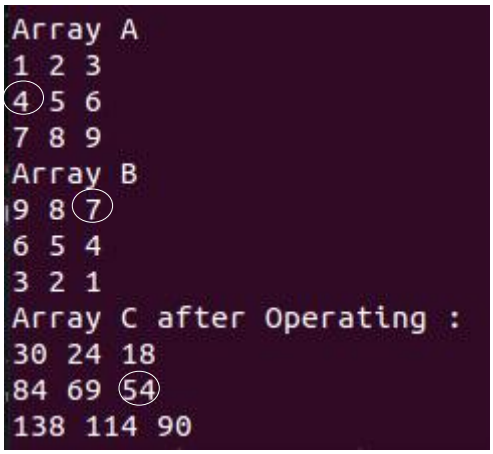
[숙제 8] lab4 실습 내용

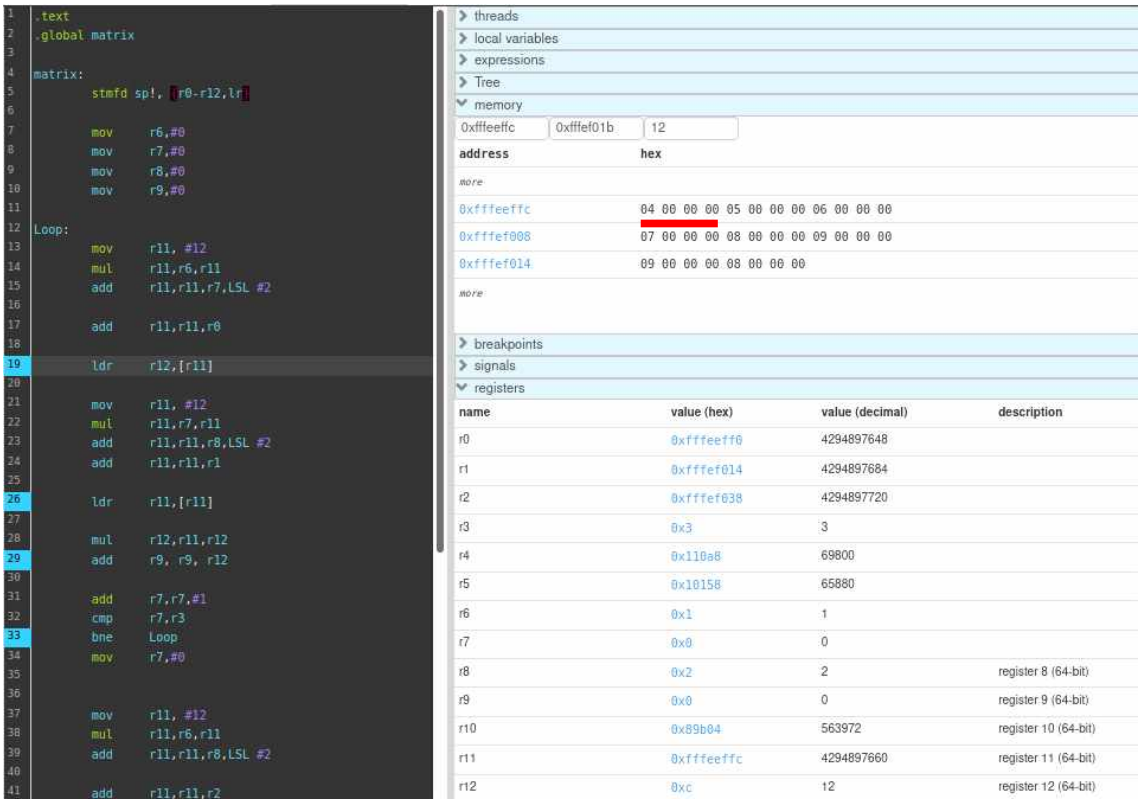
(1) 실습 A 관련(LAB4 P20)

소스 코드(matrix.s) 18번 라인에 break point를 주고, r6(s) = 1, r7(u) = 0, r8(v)=2가 되는 시점부터 trace 하세요. lab4 자료 P20 아래 그림 상에서 1번 값과 2번 값이 곱해져 3번에 저장되는 과정에서, 1번 ~ 3번에 해당하는 메모리 번지 값(R11), 메모리 내용 값(1번 및 2번 읽어온 값, 3번에 저장한 값)을 캡처하시오. 캡쳐 결과가 어느 것에 해당 하는지 표시하세요.

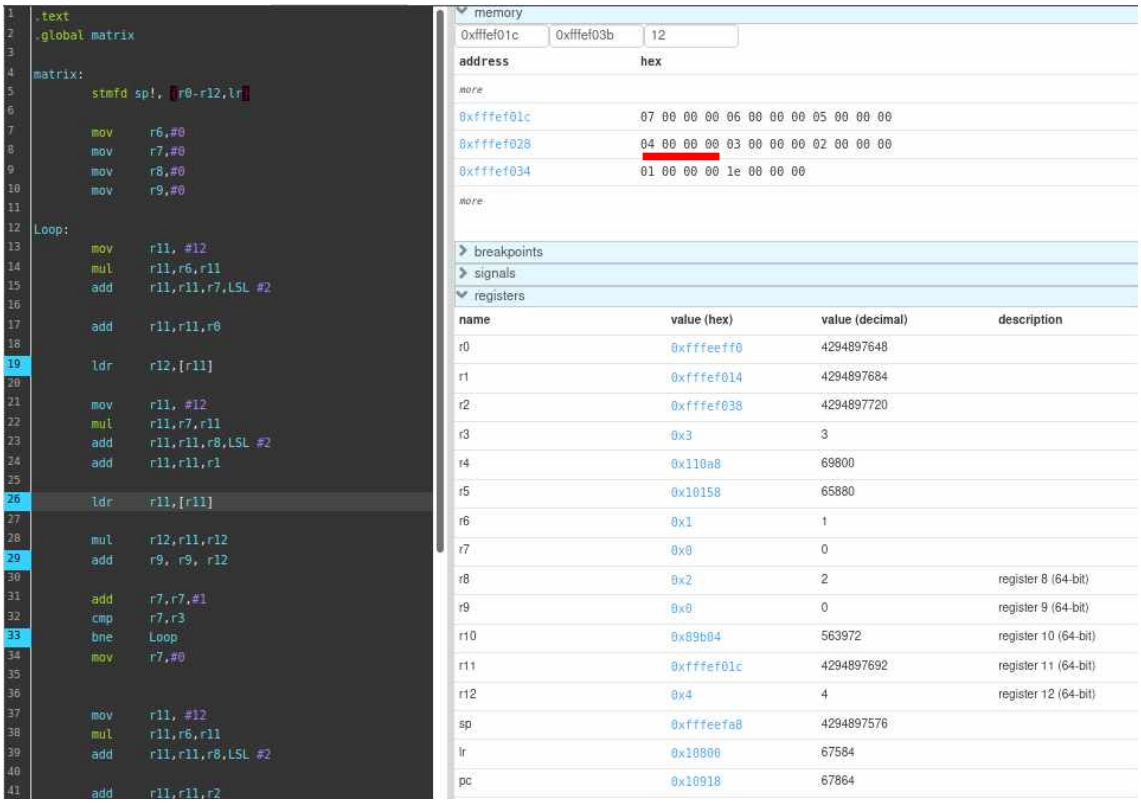
1번값 : 4, 2번값 : 7, 3번값 : 54



1번



2번



```
.text
.global matrix

matrix:
        stmfd sp!, {r0-r12,lr}

        mov     r6,#0
        mov     r7,#0
        mov     r8,#0
        mov     r9,#0

Loop:
        mov     r11, #12
        mul     r11,r6,r11
        add     r11,r11,r7,LSL #2

        add     r11,r11,r0

        ldr     r12,[r11]

        mov     r11, #12
        mul     r11,r7,r11
        add     r11,r11,r8,LSL #2
        add     r11,r11,r1

        ldr     r11,[r11]

        mul     r12,r11,r12
        add     r9, r9, r12

        add     r7,r7,#1
        cmp     r7,r3
        bne     Loop
        mov     r7,#0


        mov     r11, #12
        mul     r11,r6,r11
        add     r11,r11,r8,LSL #2

        add     r11,r11,r2
```

memory

0xfffef01c	0xfffef03b	12

address	hex
more	
0xfffef01c	07 00 00 00 06 00 00 00 05 00 00 00
0xfffef028	04 00 00 00 03 00 00 00 02 00 00 00
0xfffef034	01 00 00 00 1e 00 00 00
more	

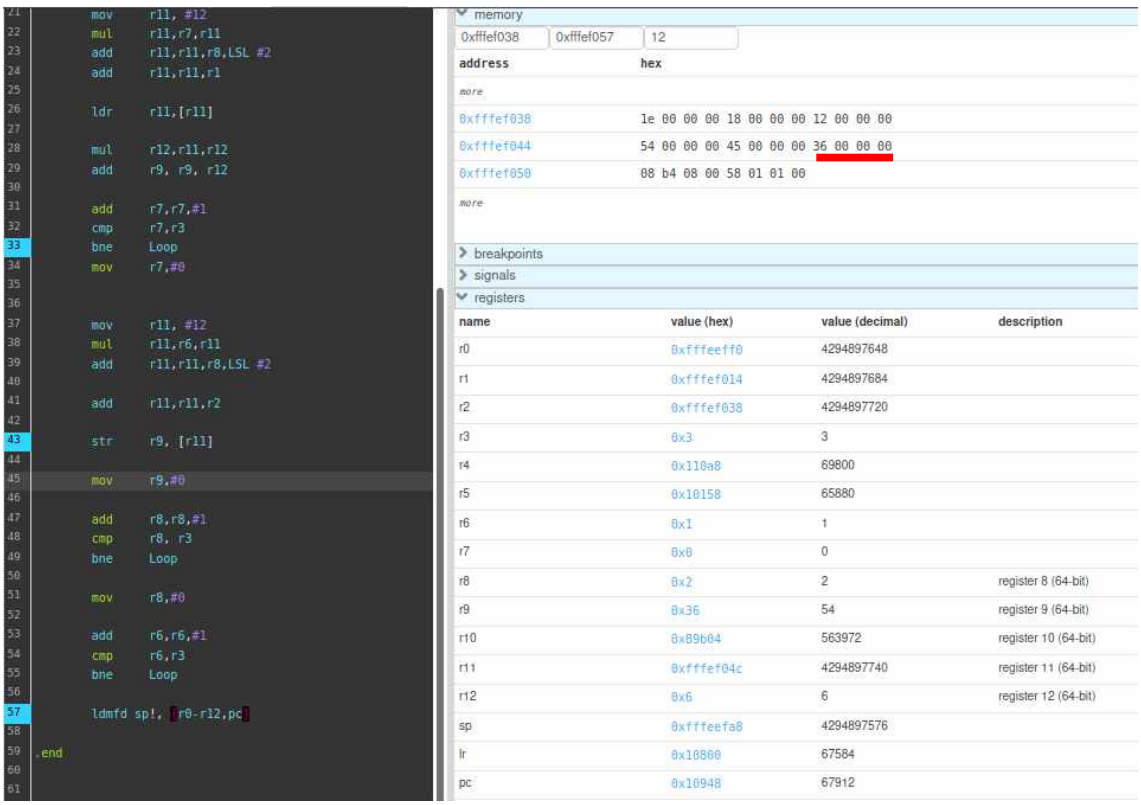
breakpoints

signals

registers

name	value (hex)	value (decimal)	description
r0	0xfffeeff0	4294897648	
r1	0xfffef014	4294897684	
r2	0xfffef038	4294897720	
r3	0x3	3	
r4	0x110a8	69800	
r5	0x10158	65880	
r6	0x1	1	
r7	0x0	0	
r8	0x2	2	register 8 (64-bit)
r9	0x0	0	register 9 (64-bit)
r10	0x89b04	563972	register 10 (64-bit)
r11	0xfffef01c	4294897692	register 11 (64-bit)
r12	0x4	4	register 12 (64-bit)
sp	0xfffeefa8	4294897576	
lr	0x10800	67584	
pc	0x10918	67864	

3번



```
        mov     r11, #12
        mul     r11,r7,r11
        add     r11,r11,r8,LSL #2
        add     r11,r11,r1

        ldr     r11,[r11]

        mul     r12,r11,r12
        add     r9, r9, r12

        add     r7,r7,#1
        cmp     r7,r3
        bne     Loop
        mov     r7,#0


        mov     r11, #12
        mul     r11,r6,r11
        add     r11,r11,r8,LSL #2

        add     r11,r11,r2

        str     r9, [r11]

        mov     r9,#0

        add     r8,r8,#1
        cmp     r8, r3
        bne     Loop

        mov     r8,#0

        add     r6,r6,#1
        cmp     r6,r3
        bne     Loop

        ldmfd sp!, {r0-r12,pc}

.end
```

memory

0xfffef038	0xfffef057	12

address	hex
more	
0xfffef038	1e 00 00 00 18 00 00 00 12 00 00 00
0xfffef044	54 00 00 00 45 00 00 00 36 00 00 00
0xfffef050	08 b4 08 00 58 01 01 00
more	

breakpoints

signals

registers

name	value (hex)	value (decimal)	description
r0	0xfffeeff0	4294897648	
r1	0xfffef014	4294897684	
r2	0xfffef038	4294897720	
r3	0x3	3	
r4	0x110a8	69800	
r5	0x10158	65880	
r6	0x1	1	
r7	0x0	0	
r8	0x2	2	register 8 (64-bit)
r9	0x36	54	register 9 (64-bit)
r10	0x89b04	563972	register 10 (64-bit)
r11	0xfffef04c	4294897740	register 11 (64-bit)
r12	0x6	6	register 12 (64-bit)
sp	0xfffeefa8	4294897576	
lr	0x10800	67584	
pc	0x10948	67912	

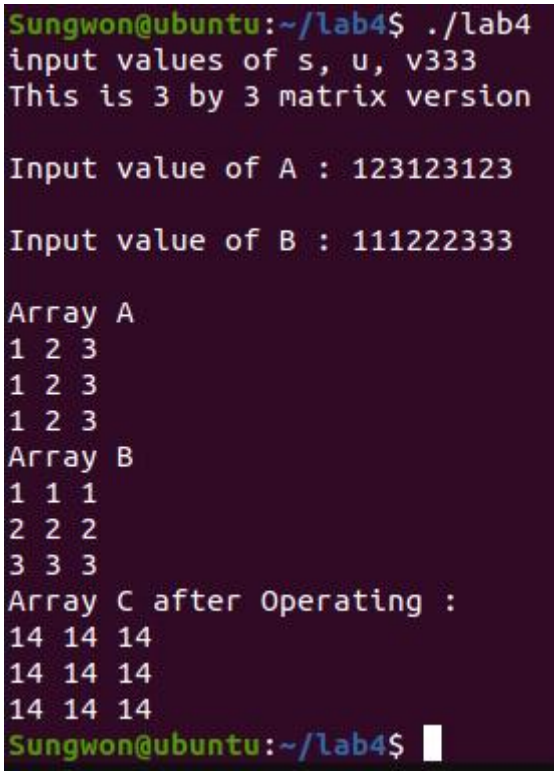
(2) 실습 B 관련(LAB4 P21)

C 프로그램상에서 다른 크기의 s ,u, v 를 입력받고 malloc 을 이용하여 matrix A[s,u], B[u,v], C[s,v] 를 위한 배열 메모리를 할당합니다 . 두 matrix A, B 의 각 원소값을 입력받아 각각 초기화합니다. 매트릭스 인덱스 s, u, v 값이 저장된 3 x 1 array 1개 D 를 선언합니다. C 프로그램에서 Matrix A, B, C 에 대한 pointer값 3 개, D 의 시작번지를 argument 로 하여 어셈블리 함수를 호출합니다. 어셈블리 코드에서는 matrix 곱셈을 수행하여 결과를 Matrix C 에 저장합니다. (Lab4 자료 P22, 23, 24, 27 내용대로 구현하세요). Test를 아래와 같이 수행하여 결과를 제출하세요.

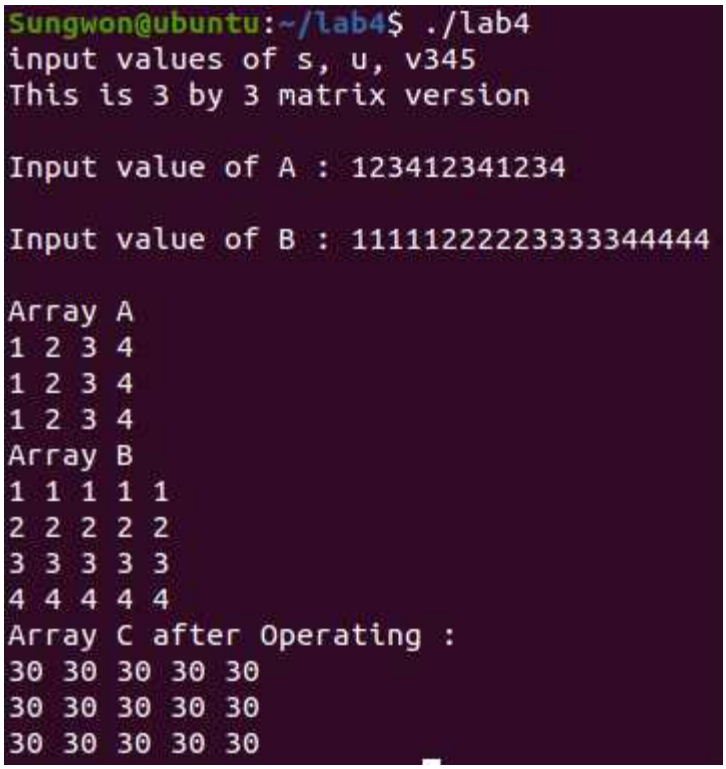
```
.text
.global matrix

matrix:
        stmfd sp!, {r0-r12,lr}          # 사용할 레지스터 스택영역에 push

        ldr     r4, [r3, #4]            # u 값을 r4에 로드
        ldr     r5, [r3, #8]            # v 값을 r3에 로드
        ldr     r10, [r3, #0]           # s 값을 r10에 로드
        mov     r6,#0                   # s까지의 인덱스
        mov     r7,#0                   # u까지의 인덱스
        mov     r8,#0                   # v까지의 인덱스
        mov     r9,#0                   # 곱연산을 저장함

Loop:
        mov     r11, r0                 # A의 원소를 저장할 레지스터
        ldr     r11, [r11, r6, lsl#2]

        ldr     r11, [r11, r7, lsl#2]   # r6, r7 인덱스에 대한 값을 r11에 로드

        mov     r12, r1                 # B의 원소를 저장할 레지스터
        ldr     r12, [r12, r7, lsl#2]

        ldr     r12, [r12, r8, lsl#2]   # r7, r8 인덱스에 대한 값을 r12에 로드

        mul     r12, r11, r12           # A와 B의 원소를 곱해 r12에 저장
        add     r9, r9, r12             # 곱한 값을 r9에 누적시킴

        add     r7, r7, #1              # r7 인덱스를 증가시킴
        cmp     r7, r4
        bne     Loop                    # 인덱스값이 u와 같을 때 까지 반복
        mov     r7, #0                  # r7에 0을 저장
```

```
        mov     r11, r2                 # C의 원소를 저장할 레지스터
        ldr     r11, [r11, r6, lsl#2]

        add     r11, r11, r8, lsl#2     # r6, r8 인덱스에 대한 값을 r11에 지정

        str     r9, [r11]               # 연산이 끝난 r9을 r11위치에 저장

        mov     r9, #0                  # r9 초기화

        add     r8,r8,#1                # r8 인덱스 증가
        cmp     r8, r5
        bne     Loop                    # 인덱스값이 v와 같을 때 까지 반복

        mov     r8,#0                   # r8 인덱스 초기화

        add     r6,r6,#1                # r6 인덱스 증가
        cmp     r6,r10
        bne     Loop                    # 인덱스값이 s와 같을 때 까지 반복

        ldmfd sp!, {r0-r12,pc}          # 사용한 레지스터 스택에서 pop, 함수 호출
                                          이전으로 되돌림

.end
```

● s, u, v 값을 전부 3 으로 입력하고 Matrix 배열 메모리를 할당받고 위와 같이 초기화 한후 Matrix 곱셈 수행후 결과 출력 후 캡쳐 첨부



● s=3, u=4, v=5 로 입력하고 Matrix 배열 메모리를 할당받고 위와 같이 초기화 한후 Matrix 곱셈



수행후 결과출력 후 캡쳐 첨부
- s, u, v 는 1보다 크고 9보다 작은 수를 사용
- 어셈블리 코드에 주석을 추가하세요. 결과 화면을 캡쳐하세요. - C 및 어셈블리 소스 코드를 파일로 제출하세요. 보고서와 소스코드를 압축하여 하나의 file(파일이름명:HW8-학번-이름)로 스마트캠퍼스에 제출하세요.