

# Assignment2

## 1. Problem & Purpose

- i. 두 문자열이 같을 경우에는 0x0A를, 그렇지 않을 경우에는 0x0B를 4000번지에 저장하는 Strcmp 함수 작성하기
- ii. 배열을 Array[10] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1}을 역순으로 4000번지부터 저장하시오
- iii. 11+13+...+27+29를 계산하여 결과값을 4000번지에 저장하고 각 구현방법의 성능을 비교해보자(Loop을 이용한 방법[11을 shift 연산으로 구현하고 숫자는 #1만 사용], 일 반화된 식  $n(n+10)$  이용, Unrolling 이용)
- iv. Branch와 conditional execution의 차이점과 성능 차이

## 2. Used Instruction

- I. 2-1: LDR // LDRB // STR // CMP // MOV // MOVNE // B // BEQ // DCB // DCD // END
  - i. LDR Rd, operand1 : operand1의 메모리 위치의 값을 word 크기만큼 Rd에 불러온다.
  - ii. LDRB Rd, operand1 : operand1의 메모리 위치의 값을 byte 크기만큼 Rd에 저장한다.
  - iii. STR Rd, [R0, offset] : R0으로부터 offset만큼 이동한 위치에 R0의 값을 word 크기만큼 저장한다.
  - iv. CMP Rd, operand1 : Rd – operand1을 한 state를 cpsr에 업데이트한다.
  - v. MOV Rd operand1 : operand1에 있는 값을 Rd에 저장한다.
  - vi. MOVNE Rd operand1 : Z가 0인 경우(CMP로 비교한 두 값이 다를 경우) operand1에 있는 값을 Rd에 저장한다.
  - vii. B Label1 : Label1의 위치로 이동한다.
  - viii. BEQ Label1 : Z가 1인 경우(CMP로 비교한 두 값이 같을 경우) Label1의 위치로 이동한다.
  - ix. Value1 DCB expr{expr} : expr의 정보를 1byte 단위로 저장하고 Value1을 통해 해당 위치를 읽어올 수 있다.

- x. Value1 DCD expr{expr} : expr의 정보를 4byte 단위로 저장하고 Value1을 통해 해당 위치를 읽어올 수 있다.
- xi. END : Assembly code가 끝났음을 의미하는 Instruction

## II. 2-2 : LDR // STR // MOV // CMP // B // BEQ // DCD // END

- i. LDR Rd, operand1 : operand1의 메모리 위치의 값을 word 크기만큼 Rd에 저장한다.
- ii. STRB Rd, [R0, offset] : R0으로부터 offset만큼 이동한 위치에 R0의 값을 byte 크기만큼 저장한다.
- iii. MOV Rd operand1 : operand1에 있는 값을 Rd에 저장한다.
- iv. CMP Rd, operand1 : Rd – operand1을 한 state를 cpsr에 업데이트한다.
- v. B Label1 : Label1의 위치로 이동한다.
- vi. BEQ Label1 : Z가 1인 경우(CMP로 비교한 두 값이 같을 경우) Label1의 위치로 이동한다.
- vii. Value1 DCD expr{expr} : expr의 정보를 4byte 단위로 저장하고 Value1을 통해 해당 위치를 읽어올 수 있다.
- viii. END : Assembly code가 끝났음을 의미하는 Instruction

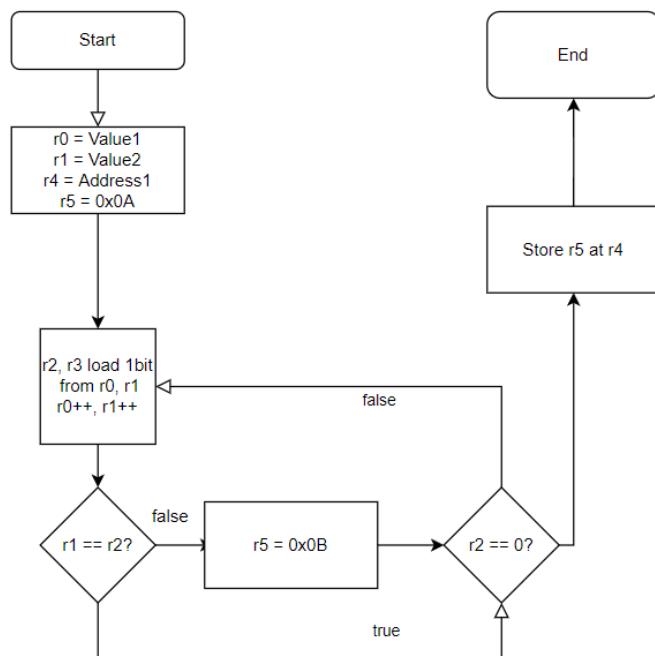
## III. 2-3 : LDR // STR // ADD // ADDNE // MUL // LSL // MOV // CMP // BNE // BEQ // DCD // END

- i. LDR Rd, operand1 : operand1의 메모리 위치의 값을 word 크기만큼 Rd에 저장한다.
- ii. STR Rd, [R0, offset] : R0으로부터 offset만큼 이동한 위치에 R0의 값을 word 크기만큼 저장한다.
- iii. ADD Rd, R0(, R1 ) : Rd에 R0와 R1을 더한 값을 저장한다. R1이 없을 경우 Rd = Rd + R0로 저장한다.
- iv. ADDNE Rd, R0(, R1 ) : Z가 0인 경우(CMP로 비교한 두 값이 다를 경우) Rd에 R0와 R1을 더한 값을 저장한다. R1이 없을 경우 Rd = Rd + R0로 저장한다.

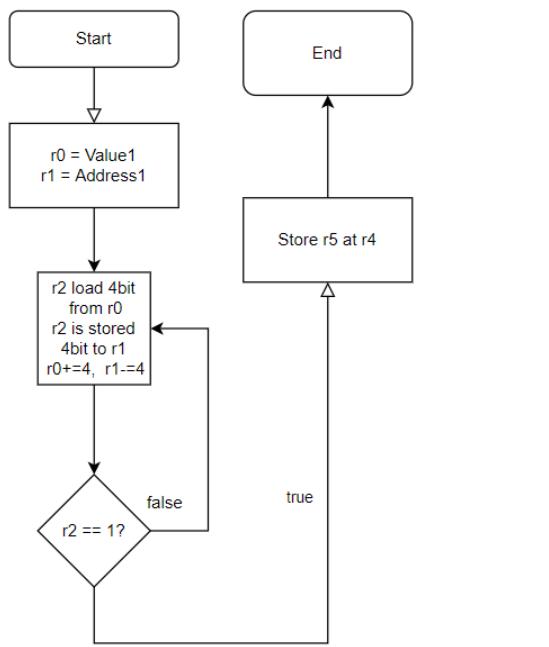
- v. MUL Rd, R0, R1 : Rd에 R0와 R1의 곱셈 값을 저장한다.
- vi. Rd, LSL, operand1 : Rd를 operand1만큼 왼쪽으로 비트 쉬프트한 값을 사용한다.
- vii. MOV Rd operand1 : operand1에 있는 값을 Rd에 저장한다.
- viii. CMP Rd, operand1 : Rd – operand1을 한 state를 cpsr에 업데이트한다.
- ix. BNE Label1 : Z가 0인 경우(CMP로 비교한 두 값이 다를 경우) Label1의 위치로 이동한다.
- x. BEQ Label1 : Z가 1인 경우(CMP로 비교한 두 값이 같을 경우) Label1의 위치로 이동한다.
- xi. Value1 DCD expr{expr} : expr의 정보를 4byte 단위로 저장하고 Value1을 통해 해당 위치를 읽어올 수 있다.
- xii. END : Assembly code가 끝났음을 의미하는 Instruction

### 3. Design(Flow chart)

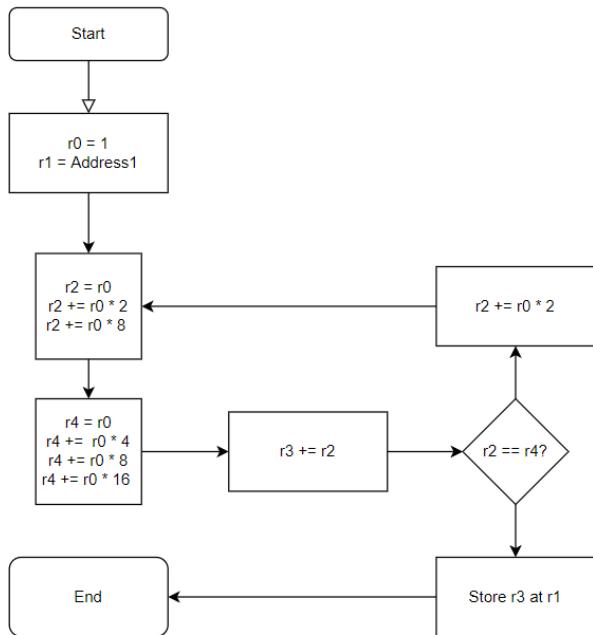
- i. 2-1 flow chart



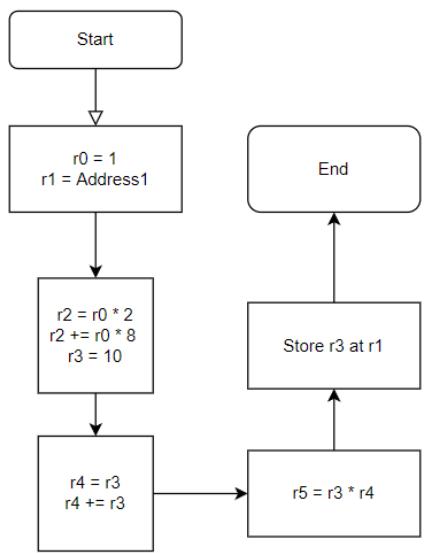
ii. 2-2 flow chart



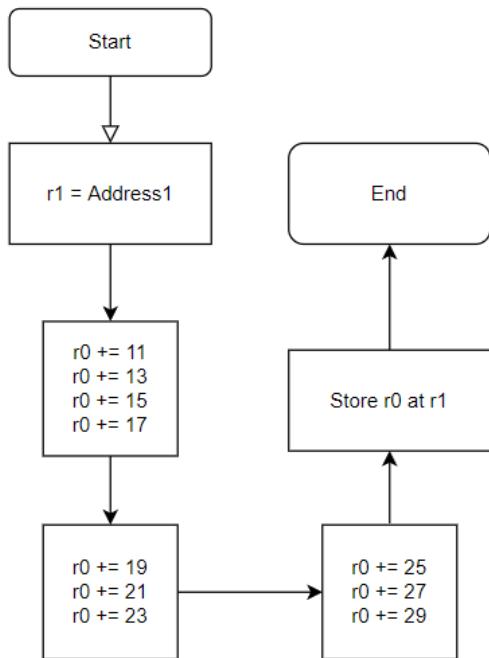
iii. 2-3-1 flow chart



iv. 2-3-2 flow chart



v. 2-3-3 flow chart



## 4. Conclusion

i. 2-1 result

C:\Users\82109\Documents\asem\2\_1\assignment2\_1.uvproj - μVision

**Registers**

Register	Value
R0	0x00000034
R1	0x00000040
R2	0x00000000
R3	0x00000000
R4	0x00004000
<b>R5</b>	<b>0x0000000A</b>
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000010
CPSR	0x00000003
User/System	0x00000000
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000010
Mode	Supervisor
States	10
Sec	0.00000000

**Disassembly**

```

0x00000008 E59F403C LDR R4, [PC, #0x003C]
    7: MOV r5, #0xA ;state
    8:
    9: Main
0x0000000C E3A0500A MOV R5, #0x0000000A
    10: LDRB r2, [r0], #1 ;load lbit char from Value1
0x00000010 E4D02001 LDRB R2, [R0], #0x0001
    < ...

```

**assignment2\_1.s**

```

1 AREA AHMEx, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1 ;Load Value1's address at r0
5 LDR r1, =Value2 ;Load Value2's address at r1
6 LDR r4, Address1 ;Address to store state(is same)
7 MOV r5, #0xA ;state
8
9 Main
10 LDRB r2, [r0], #1 ;load lbit char from Value1
11 LDRB r3, [r1], #1 ;load lbit char from Value2
12
13 CMP r2, r3 ;compare r2, r3
14 MOVNE r5, #0xB ;if r2 != r3, r5 = 0xB
15
16 CMP r2, #0 ;compare r2, 0
17 BEQ Endline ;if r2 == 0, goto Endline
18 B Main ;else goto Main
19
20 Endline
21 STR r5, [r4] ;store state at Address1
22 MOV pc, lr
23
24
25 Value1 DCB "Hello_world",0 ;Compare String
26 Value2 DCB "Hello_world",0 ;CompareString
27
28 Address1 DCD &4000
29
30 END

```

**Command**

```

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 88 Bytes (0%)
Include "C:\\\\Users\\\\82109\\\\Documents\\\\asem\\\\2_1\\\\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
< ...
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory 1**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

r0, r1에 Value1과 Value2가 저장된 위치가 들어갔고, r4에는 저장할 위치, r5에는 현재 상태인 0xA가 정상적으로 들어갔다.

C:\Users\82109\Documents\asem\W\_2\_1\assignment2\_1.uvproj - µVision

**Registers**

Register	Value
R0	0x00000035
R1	0x00000041
R2	0x00000048
R3	0x00000048
R4	0x00004000
R5	0x00000004
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x0000001C
CPSR	0x60000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x0000001C
Mode	Supervisor
States	17
Sec	0,00000000

**Disassembly**

```

0x0000000018 E1520003 CMP R2,R3
14: MOVNE r5, #0x0B ;if r2 != r3, r5 = 0x0B
15:
0x000000001C 13A0500B MOVNE R5,#0x0000000B
16: CMP r2, #0 ;compare r2, 0
0x0000000020 E3520000 CMP R2,#0x00000000
17: BEQ Endline ;if r2 == 0, goto Endline

```

**assignment2\_1.s**

```

1 AREA ARMEx, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1 ;Load Value1's address at r0
5 LDR r1, =Value2 ;Load Value2's address at r1
6 LDR r4, Address1 ;Address to store state(is same)
7 MOV r5, #0x0A ;state
8
9 Main
10 LDRB r2, [r0], #1 ;load 1bit char from Value1
11 LDRB r3, [r1], #1 ;load 1bit char from Value2
12
13 CMP r2, r3 ;compare r2, r3
14 MOVNE r5, #0x0B ;if r2 != r3, r5 = 0x0B
15
16 CMP r2, #0 ;compare r2, 0
17 BEQ Endline ;if r2 == 0, goto Endline
18 B Main ;else goto Main
19
20 Endline
21 STR r5, [r4] ;store state at Address1
22 MOV pc, lr
23
24
25 Value1 DCB "Hello_world",0 ;Compare String
26 Value2 DCB "Hello_world",0 ;CompareString
27
28 Address1 DCD &4000
29
30 END

```

**Memory**

Address: 0x00004000

0x00004000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation t1: 0.000000

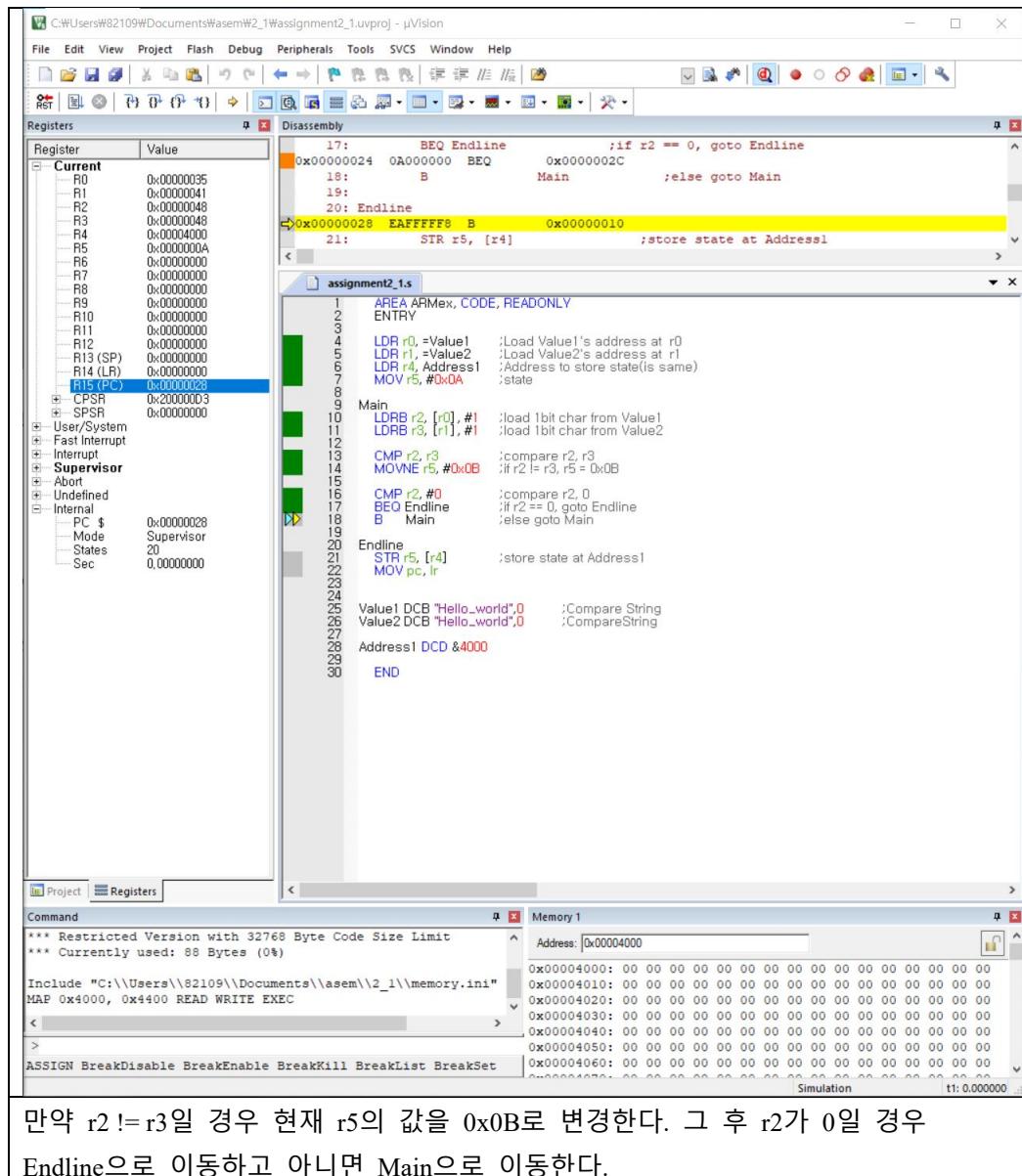
**Command**

```

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 88 Bytes (0%)
Include "C:\\\\Users\\\\82109\\\\Documents\\\\asem\\\\W_2_1\\\\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
< >
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

r2, r3는 각각 r0, r1에서 1바이트를 읽어오고 이 주소를 1 늘린다. 그 후 데이터를 비교하여 현재 cpsr를 업데이트 한다.



만약 r2 != r3일 경우 현재 r5의 값을 0x0B로 변경한다. 그 후 r2가 0일 경우 Endline으로 이동하고 아니면 Main으로 이동한다.

C:\Users\82109\Documents\asem\2\_1\assignment2\_1.uvproj - µVision

**Registers**

Register	Value
R0	0x00000040
R1	0x0000004C
R2	0x00000000
R3	0x00000000
R4	0x00004000
R5	0x00000004
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000030
CPSR	0x60000033
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000030
Mode	Supervisor
States	167
Sec	0,00000000

**Disassembly**

```

0x00000028 EAFFFFF8 B      0x00000010
21:     STR r5, [r4]          ;store state at Address1
0x0000002C E5845000 STR    R5,[R4]
22:     MOV pc, lr
0x00000030 ELA0F00E MOV    PC,R14
0x00000034 6C6C6548 STCVSL p5,CR6,[R12],#-0x0120
0x00000038 6F775F6F SWIVS  0x00775F6F

```

**assignment2\_1.s**

```

1 AREA ARMEx, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1           ;Load Value1's address at r0
5 LDR r1, =Value2           ;Load Value2's address at r1
6 LDR r4, Address1          ;Address to store state(is same)
7 MOV r5, #0x0A              ;state
8
9 Main
10 LDRB r2, [r0], #1         ;load 1bit char from Value1
11 LDRB r3, [r1], #1         ;load 1bit char from Value2
12 CMP r2, r3                ;compare r2, r3
13 MOVNE r5, #0x0B           ;if r2 != r3, r5 = 0xB
14 CMP r2, #0                 ;compare r2, 0
15 BEQ Endline               ;if r2 == 0, goto Endline
16 B Main                     ;else goto Main
17
18 Endline
19 STR r5, [r4]               ;store state at Address1
20
21 MOV pc, lr
22
23
24 Value1 DCB "Hello_world",0 ;Compare String
25 Value2 DCB "Hello_world",0 ;CompareString
26
27 Address1 DCD &4000
28
29
30 END

```

**Memory**

Address: 0x00004000

0x00004000: 0A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

현재 상태인 r5를 r4의 위치(4000번지)에 저장한다.

## ii. 2-2 result

C:\Users\82109\Documents\asem\2\_2\assignment2\_2.uvproj - μVision

**Registers**

Register	Value
R0	0x00000020
R1	0x00004024
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000008
CPSR	0x00000003
User/System	0x00000000
Fast Interrupt	0x00000000
Interrupt	0x00000000
<b>Supervisor</b>	0x00000000
Abort	0x00000000
Undefined	0x00000000
Internal	0x00000000
PC \$	0x00000008
Mode	Supervisor
States	6
Sec	0.00000000

**Disassembly**

```

0x00000000 E59F0044 LDR R0,[PC,#0x0044]
5: LDR r1, Address ;Address to store
6:
7: Main
0x00000004 E59F103C LDR R1,[PC,#0x003C]
8: LDR r2, [r0], #4 ;load int data from Value1(order)
0x00000008 E4902004 LDR R2,[R0],#0x0004

```

**assignment2\_2.s**

```

1 AREA AHMEx, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1 ;Load Value1's address at r0
5 LDR r1, Address1 ;Address to store
6
7 Main
8 LDR r2, [r0], #4 ;Load int data from Value1(order)
9 STR r2, [r1], #-4 ;Store int data at Address1(reverse order)
10 CMP r2, #1
11 BEQ Endline ;If r2 == 1, goto Endline
12 B Main ;Else goto Main
13
14 Endline
15 MOV pc, lr
16
17
18 Value1 DCD 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 ;Compare String
19 Address1 DCD &4024 ;Address to store array
20
21
22 END

```

**Books** | **Funct...** | **Regis...** | **Project**

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DRAMP3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\\Users\\82109\\Documents\\asem\\2_2\\Objects\\assignment2_2.o"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 80 Bytes (0%)
Include "C:\\Users\\82109\\Documents\\asem\\2_2\\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory**

Address: 0x00004000

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation tt: 0.000000

r0에 배열이 들어갔고, r1에는 4024번지의 주소가 들어갔다.

C:\Users\82109\Documents\asem\W\_2\assignment2\_2\assignment2\_2.uvproj - µVision

**Registers**

Register	Value
R0	0x00000024
R1	0x00004020
R2	0x0000000A
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000010
CPSR	0x00000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000010
Mode	Supervisor
States	11
Sec	0,00000000

**Disassembly**

```

0x00000008 E4902004 LDR R2,[R0],#-4 ;Store int data at Addressl(reverse order)
9: STR r2, [r1], #-4
10: 
0x0000000C E4012004 STR R2,[R1],#-0x0004 ;Store int data at Addressl(reverse order)
11: CMP r2, #1
12: BEQ Endline ;if r2 == 1, goto Endline
13: CMP r2, #1
14: B Main ;else goto Main
15: Endline
16: MOV pc, lr
17: 
18: Value1 DCD 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 ;Compare String
19: Address1 DCD &4024 ;Address to store array
20: 
21: END
22: 
23: 

```

**assignment2\_2.s**

```

1 AREA ARMEx, CODE, READONLY
2 ENTRY
3 
4 LDR r0, =Value1 ;Load Value1's address at r0
5 LDR r1, Address1 ;Address to store
6 
7 Main
8 LDR r2, [r0], #4 ;Load int data from Value1(order)
9 STR r2, [r1], #-4 ;Store int data at Addressl(reverse order)
10 CMP r2, #1
11 BEQ Endline ;if r2 == 1, goto Endline
12 B Main ;else goto Main
13 
14 Endline
15 MOV pc, lr
16 
17 
18 Value1 DCD 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 ;Compare String
19 Address1 DCD &4024 ;Address to store array
20 
21 END
22 
23 

```

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DARMP3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\Users\82109\Documents\asem\W_2\Objects\assignment2_2\assignment2_2.o"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 80 Bytes (0%)
Include "C:\Users\82109\Documents\asem\W_2\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC

```

**Memory 1**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 0A 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation t1: 0.000000

r0에 있던 데이터를 r2를 거쳐 r1으로 이동시킨다. 이때 r0의 주소는 4byte 늘리고, r1의 주소는 4byte 줄인다.

C:\Users\82109\Documents\asem\W\_2\assignment2\_2.uvproj - µVision

**Registers**

Register	Value
R0	0x00000024
R1	0x00004020
R2	0x00000004
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000018
CPSR	0x20000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000018
Mode	Supervisor
States	13
Sec	0,00000000

**Disassembly**

```

12: BEQ Endline      ;if r2 == 1, goto Endline
0x00000014 0A000000 BEQ    0x0000001C
13: B Main          ;else goto Main
14:
15: Endline
0x00000018 EAFFFFA B    0x00000008
16: MOV pc, lr

```

**assignment2\_2.s**

```

1 AREA ARMEx, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1      ;Load Value1's address at r0
5 LDR r1, Address1     ;Address to store
6
7 Main
8 LDR r2, [r0], #4      ;Load int data from Value1(order)
9 STR r2, [r1], #-4     ;Store int data at Address1(reverse order)
10
11 CMP r2, #1          ;if r2 == 1, goto Endline
12 BEQ Endline
13 B Main              ;else goto Main
14
15 Endline
16 MOV pc, lr
17
18 Value1 DCD 10, 9, 8, 7, 6, 5, 4, 3, 2, 1      ;Compare String
19
20 Address1 DCD &4024      ;Address to store array
21
22 END

```

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\\Users\\82109\\Documents\\asem\\W_2\\Objects\\assignment2_2"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 80 Bytes (0%)
Include "C:\\Users\\82109\\Documents\\asem\\W_2\\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 0A 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

만약 현재 저장한 데이터 r2가 1이 맞으면 Endline으로 이동하고, 아닐 경우 Main으로 이동한다.

Screenshot of the µVision IDE showing the assembly code, registers, memory dump, and command window.

**Registers** pane:

Register	Value
Current	
R0	0x00000048
R1	0x0003FFC
R2	0x00000001
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x0000001C
CPSR	0x60000033
SPSR	0x00000000

**Disassembly** pane:

```

1 AREA ARMex, CODE, READONLY
2 ENTRY
3
4 LDR r0, =Value1      ;Load Value1's address at r0
5 LDR r1, Address1    ;Address to store
6
7 Main
8 LDR r2, [r0], #4     ;load int data from Value1(order)
9 STR r2, [r1], #-4    ;Store int data at Address1(reverse order)
10 CMP r2, #1          ;if r2 == 1, goto Endline
11 BEQ Endline
12 B Main              ;else goto Main
13
14 Endline
15 MOV pc, lr
16
17
18 Value1 DCD 10, 9, 8, 7, 6, 5, 4, 3, 2, 1      ;Compare String
19 Address1 DCD &4024      ;Address to store array
20
21
22 END

```

**Memory** pane:

Address: 0x00004000

Address	Value
0x00004000	01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00
0x00004010	05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00
0x00004020	09 00 00 00 0A 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

**Command** pane:

```

*** Error: 'C:\Keil_v5\ARM\BIN\DRAMP3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\Users\82109\Documents\asem\2_2\Objects\assign
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 80 Bytes (0%)
Include "C:\Users\82109\Documents\asem\2_2\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Text** pane (bottom right): 사진에서 볼 수 있듯이 4000번지부터 데이터가 정렬되어서 저장되어있다.

### iii. 2-3-1 result

C:\Users\82109\Documents\asemv2\_3\_1\assignment2\_3.uvproj - µVision

**Registers**

Register	Value
R0	0x00000001
<b>R1</b>	0x00004000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
<b>R15 (PC)</b>	0x00000003
CPSR	0x00000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000008
Mode	Supervisor
States	4
Sec	0.00000000

**Disassembly**

```

9:      ADD r2, r2, r0, LSL #3 ;r2 = 0001 + 0010 + 1000 = 1011
10:
11:
12:      0x00000010 E0B22180 ADD     R2,R2,R0,LSL #3
13:      MOV r4, r0               ;r4 = 00001
14:      0x00000014 E1A04000 MOV     R4,R0
15:      ADD r4, r4, r0, LSL #2 ;r4 = 00001 + 00100 = 00101
16:
17:      MOV r3, #0
18:
19:      Loop
20:      ADD r3, r2               ;r3 = r3 + r2
21:      CMP r2, r4               ;if(r2 != 29) r2 = r2+2 and Loop / else goto Endline
22:      ADDNE r2, r0, LSL #1
23:      BNE Loop
24:      BEQ Endline
25:
26:
27:      Endline
28:      STR r3, [r1]             ;Store Data
29:      MOV pc, lr
30:
31:      Address1 DCD &4000       ;Address to store array
32:
33:      END
34:

```

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\Users\82109\Documents\asem\2_3_1\Objects\assi
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
Include "C:\Users\82109\Documents\asem\2_3_1\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
< >
> ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory 1**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

r0에는 비트 수프트에 사용될 1이, r1에는 저장할 위치 4000번지가 들어갔다.

C:\Users\82109\Documents\asem\W2\_3\_1\Assignment2\_3.uvproj - µVision

**Registers**

Register	Value
R0	0x00000001
R1	0x00004000
R2	0x00000008
R3	0x00000000
<b>R4</b>	<b>0x00000010</b>
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
<b>R15 (PC)</b>	<b>0x00000024</b>
CPSR	0x00000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000024
Mode	Supervisor
States	11
Set	0,00000000

**Disassembly**

```

15:      ADD r4, r4, r0, LSL #4 ;r4 = 00001 + 00100 + 01000 + 10000 = 11101
16:      E0844200 ADD R4,R4,R0,LSL #4
17:      MOV r3, #0
18:      Loop
19:      E3A03000 MOV R3,#0x00000000
<

```

**assignment2\_3.s**

```

1 AREA ARMex, CODE, READONLY ;Assignment3 using Loop
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1 ;Address to store
6
7 MOV r2, r0 ;r2 = 0001
8 ADD r2, r2, r0, LSL #1 ;r2 = 0001 + 0010 = 0011
9 ADD r2, r2, r0, LSL #3 ;r2 = 0001 + 0010 + 1000 = 1011
10
11
12 MOV r4, r0 ;r4 = 00001
13 ADD r4, r4, r0, LSL #2 ;r4 = 00001 + 00100 = 00101
14 ADD r4, r4, r0, LSL #3 ;r4 = 00001 + 00100 + 01000 = 01101
15 ADD r4, r4, r0, LSL #4 ;r4 = 00001 + 00100 + 01000 + 10000 = 11101
16
17 MOV r3, #0
18
19 Loop
20 ADD r3, r2 ;r3 = r3 + r2
21 CMP r2, r4 ;if(r2 != 29) r2 = r2+2 and Loop / else goto Endline
22 ADDNE r2, r0, LSL #1
23 BNE Loop
24 BEQ Endline
25
26
27
28 Endline
29 STR r3, [r1] ;Store Data
30 MOV pc, lr
31
32 Address1 DCD &4000 ;Address to store array
33
34 END

```

**Books** | **Funct...** **Regis...** **Project**

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\Users\82109\Documents\asem\W2_3_1\Objects\assignment2_3.o"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
Include "C:\Users\82109\Documents\asem\W2_3_1\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
< >
> ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation t1: 0.000000

r2, r4는 비트 쉬프트를 통해 덧셈을 하여 11과 29를 만든다.

C:\Users\82109\Documents\asem\W2\_3\_1\assignment2\_3.uvproj - µVision

**Registers**

Register	Value
<b>R0</b>	0x00000001
R1	0x00000400
R2	0x00000008
R3	0x00000008
R4	0x0000001D
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
<b>R15 (PC)</b>	0x00000030
SPSR	0x00000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
<b>Internal</b>	
PC \$	0x00000030
Mode	Supervisor
States	14
Set	0,00000000

**Disassembly**

```

23: ADDNE r2, r0, LSL #1
24: BNE Loop
25: BEQ Endline
26:
27:

1 AREA ARMEx, CODE, READONLY ;Assignment3 using Loop
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1 ;Address to store
6
7 MOV r2, r0 ;r2 = 0001
8 ADD r2, r2, r0, LSL #1 ;r2 = 0001 + 0010 = 0011
9 ADD r2, r2, r0, LSL #3 ;r2 = 0001 + 0010 + 1000 = 1011
10
11
12 MOV r4, r0 ;r4 = 00001
13 ADD r4, r4, r0, LSL #2 ;r4 = 00001 + 00100 = 00101
14 ADD r4, r4, r0, LSL #3 ;r4 = 00001 + 00100 + 01000 = 01101
15 ADD r4, r4, r0, LSL #4 ;r4 = 00001 + 00100 + 01000 + 10000 = 11101
16
17 MOV r3, #0
18
19 Loop
20 ADD r3, r2 ;r3 = r3 + r2
21
22 CMP r2, r4 ;if(r2 != 29) r2 = r2+2 and Loop / else goto Endline
23 ADDNE r2, r0, LSL #1
24 BNE Loop
25 BEQ Endline
26
27
28 Endline
29 STR r3, [r1] ;Store Data
30 MOV pc, lr
31 Address1 DCD &4000 ;Address to store array
32
33 END
34

```

**Books** | **Funct...** **Regis...** **Project**

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\\Users\\82109\\Documents\\asem\\W2_3_1\\Objects\\assignment2_3.o"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
Include "C:\\Users\\82109\\Documents\\asem\\W2_3_1\\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
< >
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory**

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation t1: 0.000000

r3에 r2를 저장하고 CMP를 통해 r2와 r4(29)를 비교한다. 29가 아닐 경우에는 r2에 2를 더하고 Loop를 돌고, 같을 경우 EndLine으로 이동한다.

C:\Users\82109\Documents\asem\W2\_3\_1\assignment2\_3.uvproj - µVision

**Registers**

Register	Value
<b>R15 (PC)</b>	0x00000040
R0	0x00000001
R1	0x00000400
R2	0x0000001D
R3	0x000000C8
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
CPSR	0x60000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000040
Mode	Supervisor
States	75
Set	0.00000000

**Disassembly**

```

29:      STR r3, [r1]          ;Store Data
0x0000003C E5813000 STR R3,[R1]
30:      MOV pc, lr
0x00000040 E1A0F0E MOV PC,R14
0x00000044 00004000 ANDEQ R4,R0,R0
0x00000048 00000000 ANDEQ R0,R0,R0
0x0000004C 00000000 ANDEQ R0,R0,R0

```

**assignment2\_3.s**

```

1 AREA ARMex, CODE, READONLY ;Assignment3 using Loop
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1 ;Address to store
6
7 MOV r2, r0 ;r2 = 0001
8 ADD r2, r2, r0, LSL #1 ;r2 = 0001 + 0010 = 0011
9 ADD r2, r2, r0, LSL #3 ;r2 = 0001 + 0010 + 1000 = 1011
10
11
12 MOV r4, r0 ;r4 = 00001
13 ADD r4, r4, r0, LSL #2 ;r4 = 00001 + 00100 = 00101
14 ADD r4, r4, r0, LSL #3 ;r4 = 00001 + 00100 + 01000 = 01101
15 ADD r4, r4, r0, LSL #4 ;r4 = 00001 + 00100 + 01000 + 10000 = 11101
16
17 MOV r3, #0
18
19 Loop
20 ADD r3, r2 ;r3 = r3 + r2
21
22 CMP r2, r4 ;if(r2 != 29) r2 = r2+2 and Loop / else goto Endline
23 ADDNE r2, r0, LSL #1
24 BNE Loop
25 BEQ Endline
26
27
28 Endline
29 STR r3, [r1] ;Store Data
30 MOV pc, lr
31 Address1 DCD &4000 ;Address to store array
32
33
34 END

```

**Command**

```

*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\Users\82109\Documents\asem\W2_3_1\Objects\assignment2_3.o"
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 72 Bytes (0%)
Include "C:\Users\82109\Documents\asem\W2_3_1\memory.ini"
MAP 0x4000, 0x4400 READ WRITE EXEC
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

**Memory 1**

Address	Value
0x00004000	C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000040A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation t1: 0.000000

마지막에 현재 r3을 4000번지에 저장한다.

#### iv. 2-3-2 result

C:\Users\82109\Documents\asem\2\_3\_2\assignment2\_3\_2.uvproj - µVision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers Disassembly

Register	Value
R0	0x00000001
R1	0x00004000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000003
CPSR	0x00000003
+ SPSCR	0x00000000
User/System	
+ Fast Interrupt	
+ Interrupt	
+ Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000008
Mode	Supervisor
States	4
Sec	0.0000000

Disassembly

```

0x00000000 E3A00001 MOV R0,#0x00000001
5: LDR r1, Address1 ;Address to store
6:
0x00000004 E59F101C LDR R1,[PC,#0x001C]
7: MOV r2, r0, LSL #1 ;r2 = 0010
0x00000008 E1A02080 MOV R2,R0,LSL #1
8: ADD r2, r0, LSL #3 ;r2 = 0010 + 1000 = 1010

```

assignment2\_3\_2.s

```

1 AREA ARMex, CODE, READONLY ;Assignment3 using n(n+10)
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1 ;Address to store
6
7 MOV r2, r0, LSL #1 ;r2 = 0010
8 ADD r2, r0, LSL #3 ;r2 = 0010 + 1000 = 1010
9
10 MOV r3, #10 ;n
11
12 MOV r4, r2 ;r4 = 10
13 ADD r4, r3 ;r4 = 10 + n
14 MUL r5, r4, r3 ;r5 = (10 + n)n
15
16 STR r5, [r1] ;Store Data(r5)
17
18 MOV pc, lr
19 Address1 DCD &4000 ;Address to store array
20
21 END

```

Registers Call Stack + Locals

Name	Location/Value	Type
_as...	0x00000000	void f()

Command

```

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 44 Bytes (0%)
Include "C:\\Users\\82109\\Documents\\asem\\2_3_2\\memory.in"
MAP 0x4000, 0x4400 READ WRITE EXEC

```

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

Call Stack + Locals Memory 1 Simulation t1: 0.000

r0에는 비트 수프트에 사용될 1이, r1에는 저장할 위치 4000번지가 들어갔다.

C:\Users\82109\Documents\asem\W2\_3\_2\assignment2\_3\_2.uvproj - µVision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers

Register	Value
R0	0x00000001
R1	0x00004000
R2	0x0000000A
<b>R3</b>	<b>0x0000000A</b>
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
<b>R15 (PC)</b>	<b>0x00000014</b>
CPSR	0x00000003
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
<b>Supervisor</b>	
Abort	
Undefined	
Internal	
PC \$	0x00000014
Mode	Supervisor
States	7
Sec	0,00000000

Disassembly

```

0x00000010 E3A0300A MOV R3,#0x0000000A
12:      MOV r4, r2           ;r4 = 10
→ 0x00000014 E1A09002 MOV R4,R2
13:      ADD r4, r3           ;r4 = 10 + n
0x00000018 E0844003 ADD R4,R4,R3
14:      MUL r5, r4, r3       ;r4 = (10 + n)n
15:

```

assignment2\_3\_2.s

```

1 AREA ARMex, CODE, READONLY    ;Assignment3 using n(n+10)
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1  ;Address to store
6
7 MOV r2, r0, LSL #1 ;r2 = 0010
8 ADD r2, r0, LSL #3 ;r2 = 0010 + 1000 = 1010
9
10 MOV r3, #10          ;n
11
12 MOV r4, r2           ;r4 = 10
13 ADD r4, r3           ;r4 = 10 + n
14 MUL r5, r4, r3       ;r4 = (10 + n)n
15
16 STR r5, [r1]          ;Store Data(r5)
17
18 MOV pc, lr
19 Address1 DCD &4000      ;Address to store array
20
21 END

```

Address1 DCD &4000 ;Address to store array

Memory 1

Address	Value
0x00004000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

r2는 비트 쉬프트 연산을 통해 10이 저장되었고, r3에는 n값인 10이 저장되었다.

C:\Users\82109\Documents\asem\W2\_3\_2\assignment2\_3\_2.uvproj - µVision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers Disassembly

Register	Value
R0	0x00000001
R1	0x00004000
R2	0x00000004
R3	0x00000004
R4	0x00000014
R5	0x000000C8
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000024
CPSR	0x00000033
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000024
Mode	Supervisor
States	13
Sec	0,00000000

```

14:      MUL r5, r4, r3    ;r4 = (10 + n)n
15:      E0050394 MUL    R5,R4,R3
16:      STR r5, [r1]      ;Store Data(r5)
17:      E5815000 STR    R5,[R1]
18:      MOV pc, lr
19:      E10E0F0E MOV    PC,R14
<

```

assignment2\_3\_2.s

```

1 AREA ARMex, CODE, READONLY ;Assignment3 using n(n+10)
2 ENTRY
3
4 MOV r0, #1
5 LDR r1, Address1 ;Address to store
6
7 MOV r2, r0, LSL #1 ;r2 = 0010
8 ADD r2, r0, LSL #3 ;r2 = 0010 + 1000 = 1010
9
10 MOV r3, #10 ;n
11
12 MOV r4, r2 ;r4 = 10
13 ADD r4, r3 ;r4 = 10 + n
14 MUL r5, r4, r3 ;r4 = (10 + n)n
15
16 STR r5, [r1] ;Store Data(r5)
17
18 Address1 DCD &4000 ;Address to store array
19
20
21 END

```

Memory 1

Address	Value
0x00004000	C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

r4에는  $n + 10$ 의 값, 즉 20이 저장되고, r5에는  $n(n + 10)$ 의 곱연산의 값 200이 저장된다. 그리고 이 데이터를 4000번지에 저장한다.

#### v. 2-3-3 result

C:\Users\82109\Documents\asem\W2\_3\Assignment2\_3.uvproj - µVision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers

Register	Value
R0	0x00000000
R1	0x00004000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000004
CPSR	0x00000003
+ User/System	
+ Fast Interrupt	
+ Interrupt	
+ Supervisor	
+ Abort	
+ Undefined	
Internal	
PC \$	0x00000004
Mode	Supervisor
States	3
Sec	0.0000000

Disassembly

```

0x00000000 E59F102C LDR R1,[PC,#0x002C]
 6:      MOV r0, #11 ;ADD 11 ~ 29
→ 0x00000004 E3A0000B MOV R0,#0x0000000B
 7:      ADD r0, #13
 8:      ADD r0, #15
 9:      ADD r0, #17
10:     ADD r0, #18
11:     ADD r0, #21
12:     ADD r0, #23
13:     ADD r0, #25
14:     ADD r0, #27
15:     ADD r0, #29
16:
17:     STR r0, [r1] ;Store Data
18:     MOV pc,lr
19: Address1 DCD &4000 ;Address to store array
20: END
21:
22:

```

assignment2\_3.s

```

1 AREA ARMex, CODE, READONLY ;Assignment3 using unRolling
2 ENTRY
3
4 LDR r1, Address1 ;Address to store
5
6 | MOV r0, #11 ;ADD 11 ~ 29
7 ADD r0, #13
8 ADD r0, #15
9 ADD r0, #17
10 ADD r0, #18
11 ADD r0, #21
12 ADD r0, #23
13 ADD r0, #25
14 ADD r0, #27
15 ADD r0, #29
16
17 STR r0, [r1] ;Store Data
18 MOV pc,lr
19 Address1 DCD &4000 ;Address to store array
20 END
21
22

```

Command

```

*** Error: 'C:\Keil_v5\ARM\BIN\DRAMP3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\\Users\\82109\\Documents\\asem\\W2_3\\Objects\\ass
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 56 Bytes (0%)

Include "C:\\\\Users\\\\82109\\\\Documents\\\\asem\\\\W2_3\\\\memory.in
MAP 0x4000, 0x4400 READ WRITE EXEC
|<----->
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet

```

Memory 1

Address	Value
0x00004200	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004210	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004220	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004230	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004240	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004260	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004270	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004280	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00004290	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x000042A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation tt: 0.000

r1에는 저장할 위치 4000번지가 들어갔다.

The screenshot shows the Keil uVision IDE interface with the following windows:

- Registers**: Shows the current register values. The PC register is highlighted with a blue selection bar.
- Disassembly**: Displays the assembly code for the program. The highlighted line is:
 

```
18:          MOV pc, lr
0x00000030  E1A0F0E  MOV    PC,R14
```
- Assembly Editor**: Shows the source assembly file content:
 

```
1 AREA ARMex, CODE, READONLY ;Assignment3 using unRolling
2 ENTRY
3
4 LDR r1, Address1 ;Address to store
5
6 MOV r0, #11 ;ADD 11 ~ 29
7 ADD r0, #13
8 ADD r0, #15
9 ADD r0, #17
10 ADD r0, #19
11 ADD r0, #21
12 ADD r0, #23
13 ADD r0, #25
14 ADD r0, #27
15 ADD r0, #29
16
17 STR r0, [r1] ;Store Data
18 MOV pc, lr
19
20 Address1 DCD &4000 ;Address to store array
21
22 END
```
- Memory**: A memory dump window showing memory starting at address 0x00004200.
- Command**: A window displaying build errors and warnings, including:
 

```
*** Error: 'C:\Keil_v5\ARM\BIN\DAARM3.DLL' not found
Running with Code Size Limit: 32K
Load "C:\\Users\\82109\\Documents\\asem\\\\2_3_3\\Objects\\ass

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 56 Bytes (0%)

Include "C:\\Users\\82109\\Documents\\asem\\\\2_3_3\\memory.in
MAP 0x4000, 0x4400 READ WRITE EXEC
<           >
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet | 0x000042A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

A text box at the bottom right contains the Korean note: r0에 11부터 29의 합을 저장하였고, 이 값을 r1에 저장하였다.

#### vi. 2-3 compare

- 위의 예제들을 비교하였을 때 Loop > unrolling > 일반식 순으로 state를 사용하였다. 이를 통해 우리는 프로그램을 설계할 때 일반적인 식으로 나타낼 수 있을 경우 일반적인 식으로 나타내고, 아니면 풀어서 작성하는 것이 유리하다는 것을 알 수 있다.

#### vii. Branch와 Conditional execution

- Branch의 경우 Pipeline에서 기존의 데이터를 버려야 하기 때문에 2번의 연산 손실이 발생한다. Branch를 포함해 3번의 연산 손실이 발생하기 때문에 Conditional execution을 3번 이하로 사용할 수 있을 경우 Conditional execution을 사용하고 그

외의 경우에는 Branch를 사용하는 것이 효율적이다. ↴

## 5. Reference

- i. 이준환 교수님/어셈블리프로그램설계및실습/광운대학교(컴퓨터정보공학부)/2021