어셈블리프로그램 설계 및 실습

Lab #7

Pseudo Instructions

Kwangwoon University
Embedded System Architecture Lab

학습 목표

- Pseudo instruction이 assembler에 의해 어떻게 실제 instruction으로 변환되는지 직접 확인하고 이해함
- Disassembly를 해석하는 방법을 이해함
  - 32bit 명령어들의 구조 이해
- *Label*의 이름이 실제 메모리 주소임을 이해하고 이를 이용해 어셈블리어 프로그래밍을 진행하는 능력을 습득함

Contents

- Pseudo Instructions
  - Instructions
  - Example
  - Disassembly
- Problem

Instructions

- ADR, ADRL
  - Address를 읽어오는 명령어
  - Ex)
```
Loop:   MOV R0, R1, #1
        …
        ADR R2, Loop
```
  - Ex)
```
Loop:   MOV R0, R1, #1
        …
        ADRL R2, Loop+4
```

- LDR
  - Ex) LDR R2, =Loop → Actually *label* is code memory address !!

Instructions

- LDFD / LDFS (Not recognized on our simulator)
  - Loads a floating-point register
  - Ex) LDFD f1,=3.12E106
  - Ex2) LDFS f1,=3.12E-6

- NOP
  - Short for No Operation
  - NOP is sometimes used as a description for the action performed by a function or a sequence of programming language statements if the function or code has no effect
  - C code example; {}

Example

```
1  cr equ 0x0d                    ;#define cr 0x0d
2
3      area strlen, code, readonly
4          entry
5
6  main
7      ldr r0,=Table    ;load the address of the Table
8      eor r1,r1,r1     ;clear R1 to store count , mov r1,#0
9
10 Loop
11     ldrb r2, [r0], #1    ;load the first byte into R2
12     cmp r2, #cr          ;is it the terminator ?
13     BEQ Done             ; yes => stop loop
14     ADD r1, r1, #1       ; no => increment count
15
16     BAL Loop                ; read next char
17
18 Done
19     str r1, CharCount ;store result
20     mov pc, #0        ; finish
21
22 ;=====Data1 area
23     AREA Data1, DATA
24
25 Table
26     ALIGN
27     dcb "Hello, World", cr
28
29 ;====Result Area
30     AREA Result, DATA
31 CharCount
32     DCB 0                ;storage for count
33
34     END
```

Directive.
→ EQU mean "equal"
→ Like *typedef* in C/C++

Directive.
→ The ALIGN directive aligns the current location within the code to a word (4-byte) boundary.
→ Check *Table*'s memory!

Compile error correction

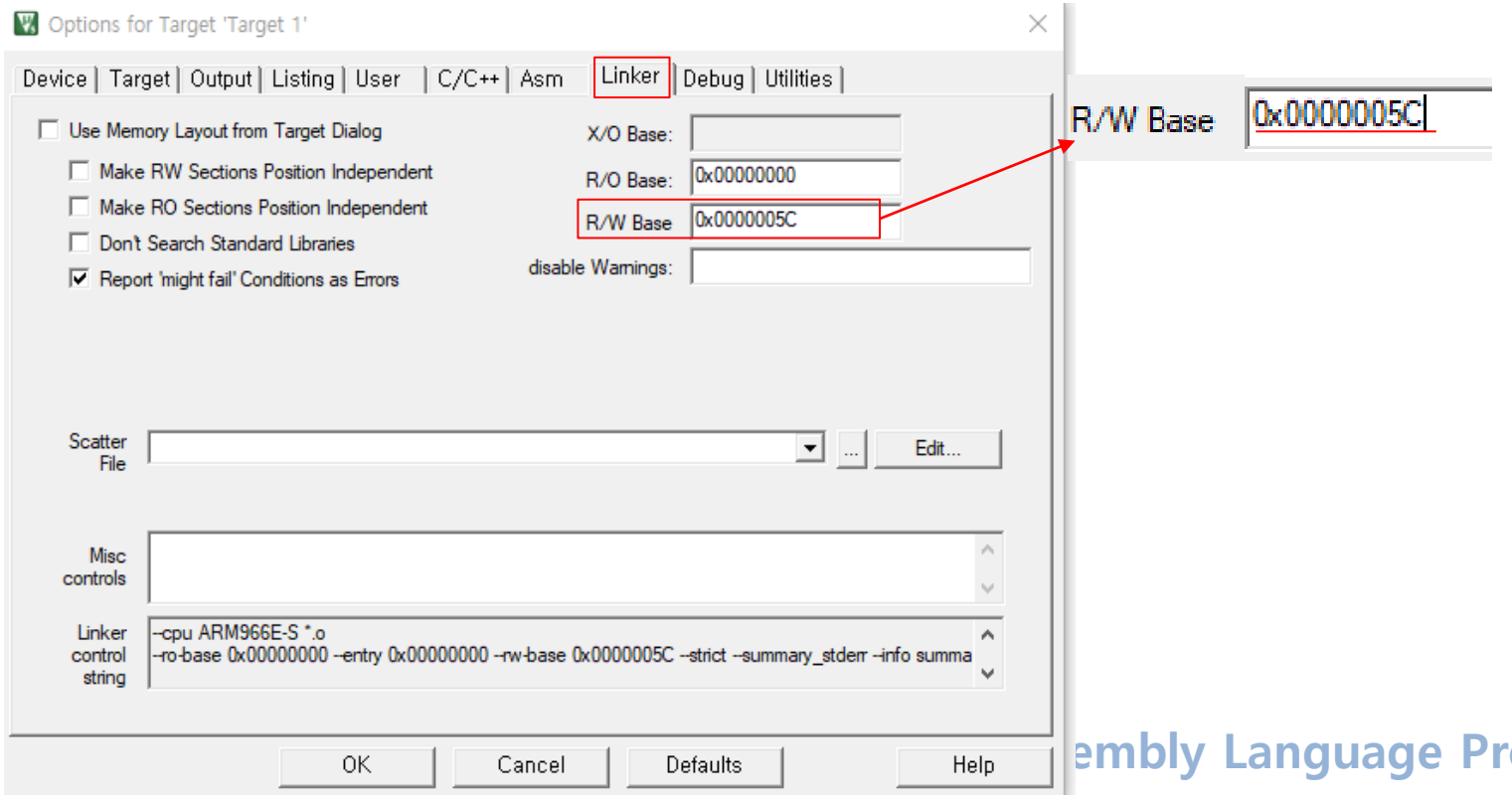
Error message (ex)

```
linking...
.\test2.axf: Error: L6221E: Execution region ER_RO with Execution range [0x00000000,0x0000005c) overlaps with Execution region ER_RW with Execution range [0x00000000,0x00000014)
".\test2.axf" - 1 Errors, 1 Warning(s).
Target not created
```

```
ER_RO with Execution range [0x00000000,0x0000005c)
```

*출력 error는 코드에 따라 달라질 수 있음

- Project → Options for Target 'XXX'



Disassembly

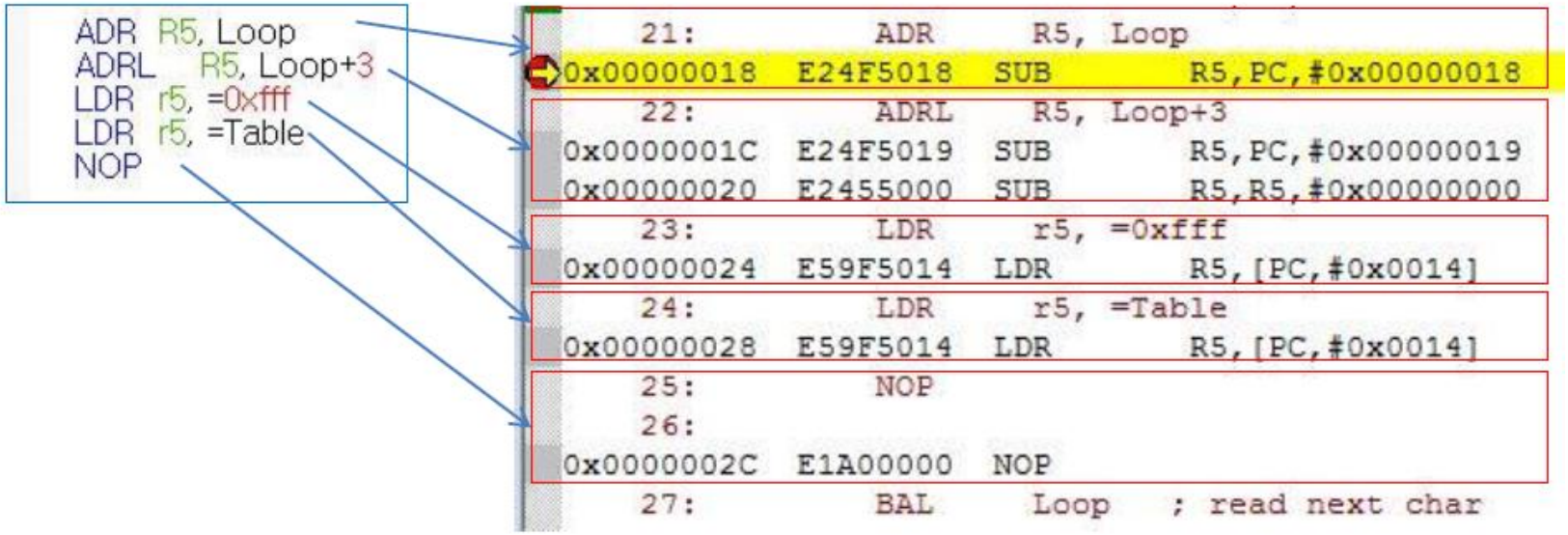
- Example

Disassembly

```
    12:          LDR     R0, =Data1      ; load the address of the lookup table
0x00000000  E59F001C  LDR       R0,[PC,#0x001C]
    13:          EOR     R1, R1, R1      ; clear R1 to store count
    14:
    15: Loop
0x00000004  E0211001  EOR       R1,R1,R1
    16:          LDRB    R2, [R0], #1        ;load the first byte into R2
0x00000008  E4D02001  LDRB      R2,[R0],#0x0001
    17:          CMP     R2, #CR             ;is it the terminator ?
0x0000000C  E352000D  CMP       R2,#0x0000000D
    18:          BEQ     Done    ; yes => stop loop
0x00000010  0A000001  BEQ       0x0000001C
    19:          ADD     R1, R1,      #1      ; no => increment count
    20:
0x00000014  E2811001  ADD       R1,R1,#0x00000001
    21:          BAL     Loop    ; read next char
    22:
    23: Done
0x00000018  EAFFFFFA  B         0x00000008
    24:          STR     R1, CharCount               ; store result
0x0000001C  E58F104C  STR       R1,[PC,#0x004C]
    25:          SWI     &11     ; finish
0x00000020  EF000011  SWI       0x00000011
0x00000024  00000060  ANDEQ     R0,R0,R0,RRX
0x00000028  6C6C6548  STCVSL    p5,CR6,[R12],#-0x0120
0x0000002C  57202C6F  STRPL     R2,[R0,-PC,ROR #24]!
0x00000030  646C726F  STRVSBT   R7,[R12],#-0x026F
0x00000034  0000000D  ANDEQ     R0,R0,R13
```

Disassembly

- Pseudo instructions



Problem

- strcpy함수 구현
  - 함수원형 : **char* strcpy(char* dest, const char* origin);**
    - 메모리 0x40000번지를 strcpy의 dest로 함
      - 정의된 문자열을 4byte 정렬하여 저장
    - strcpy의 리턴은 무시
  - 단, 이번 보고서에는 반드시 disassembly 화면을 캡쳐하여 다음 언급이 반드시 포함돼야 함.
    - Pseudo instructio이 어떻게 변경되고, 변경된 instructio이 어떤 원리로 프로그램에서 동작하는지에 대한 설명
    - 또한, 사용된 모든 명령어의 disassembly 화면을 캡쳐하여 32bit의 코드가 어떤 방식으로 이뤄져 있는지에 대한 해석
    - 임의의 문자열 1개를 정의하여 사용
      - 문자열의 길이는 10 이상

Homework

- 제출기한
  - Soft copy
    - 과제 기한
      - ~ 11.9(월)
      - klas 과제 업로드 기한을 유의하시기 바랍니다.
    - 압축 파일은 보고서와 프로젝트 전체를 모아서 제출(ini파일 포함)
      - ini 파일 관련 오류 발생 시, 0점
    - 압축파일 형식
      - Assignment_(번호)_(학번).zip
        - Ex) Assignment_7_2012722069.zip

Q&A

Thank you