어셈블리 프로그래밍 설계 및 실습
보고서

실험제목: Pseudo Instructions

실험일자: 2018년 11월 01일 (목)

제출일자: 2018년 11월 08일 (목)

학 과: 컴퓨터공학과

담당교수: 이준환 교수님

실습분반: 화 5, 수 6,7

학 번: 2015722025

성 명: 정용훈

1. 제목 및 목적

A. 제목

Pseudo Instructions

B. 목적

Pseudo instruction이 사용자가 작성한 코드가 아닌 assembler에서 어떻게 실제 instruction으로 변환되는지 직접 확인하며 이해한다. 또한 Disassembly를 해석하는 방법을 이해하며 Label의 이름이 실제 메모리 주소라는 것을 이해한다.

2. 설계 (Design)

A. Pseudo code

Strcpy

(1) 레지스터 r0에 문자열을 저장할 주소 Load

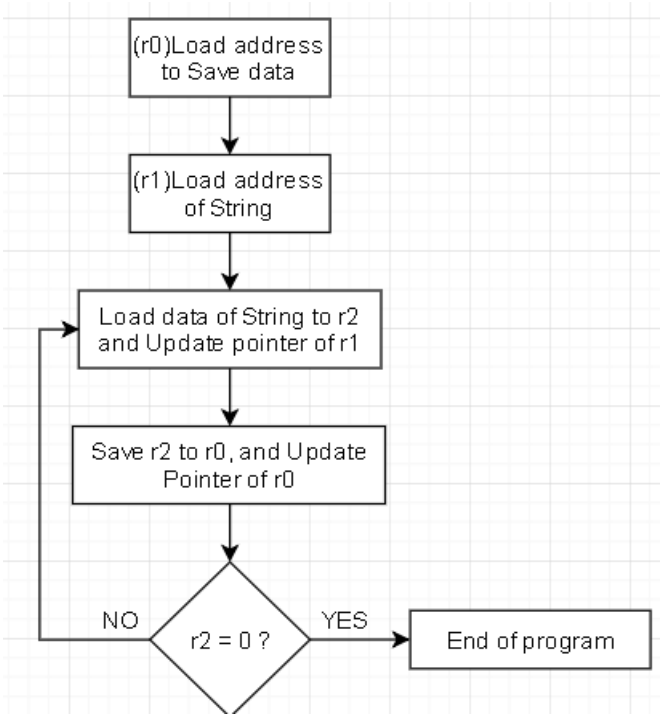
(2) 레지스터 r1에 문자열의 정보가 담긴 주소를 Load

(3) R2에 r1에서 문자열의 정보를 가져오고 pointer를 한 칸 밀어서 update

(4) R2의 정보를 r0에 저장하며 r0의 pointer를 한 칸 밀어서 update

(5) 레지스터 r2의 값과 문자열의 끝을 나타내는 정수 0을 비교하여 다르면 다시 (3)실행, 같으면 함수 종료

B. Flow chart 작성



이번 과제는 코드를 작성하는 것이 목적이 아니라 pseudo code를 배우기 위한 간단한 코드로 Flow chart또한 굉장히 간단하게 나오는 것을 확인할 수 있습니다. 간단한 strcpy함수를 설명하자면 저장할 주소와 String의 정보가 담긴 주소를 먼저 불러와 r2레지스터에 String의 정보를 순차적으로 가져와 0과 비교하여 같지 않으면 문자열이 끝나지 않았다는 뜻으로 문자를 불러오는 명령을 반복하여 수행하며 해당 문자를 r0메모리에 순차적으로 저장하게 되며 문자열 복사를 구현할 수 있습니다. R2와 값을 비교하여 0값에 도달하게 되면 프로그램을 종료하도록 설계 되어있습니다.

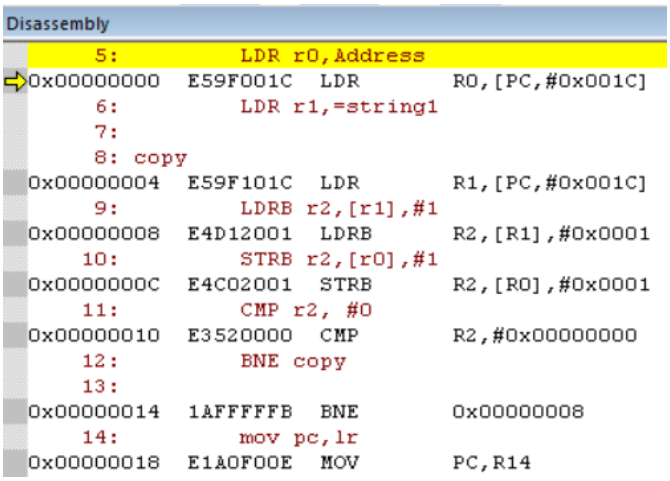
C.Result

```
0x00040000: 48 65 6C 6C 6F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

위 이미지는 문자열이 16진법 아스키코드로 들어가 있는 모습입니다. 아스키 코드 표를 맞춰 변환하면 Hello,0로 입력한 String의 정보가 그대로 복사된 것을 확인할 수 있습니다.

3. Pseudo instruction & Disassembly

이번 실습에서는 코드 구현의 목적보다는 pseudo instruction과 Disassembly에 관련된 내용을 주 목적으로 다루고 있습니다 Disassembly를 통하여. 32bit의 명령어들을 확인할 수 있으며 해당 구조의 대한정보는 앞선 강의를 통하여 배웠습니다. 가장 먼저 이번 코드를 작성하면서 사용된 pseudo instruction은 LDR과 STR이 있으며 Disassembly를 통하여 나타내면 아래 사진과 같은 정보를 얻을 수 있습니다.



각각의 32bit instruction이 쓰인 정보를 보면 7가지로 나눌 수 있습니다.

E59F001C, E59F101C, E4D12001, E4C02001, E3520000, 1AFFFFFB, E1A0F00E으로 나눌 수 있으며 해당 정보를 32bit binary로 바꿔주면 아래와 같습니다.

E59F001C = 1110 0101 1001 1111 0000 0000 0001 1100 (LDR)

E59F101C = 1110 0101 1001 1111 0001 0000 0001 1100 (LDR)

E4D12001 = 1110 0100 1101 0001 0010 0000 0000 0001 (LDRB)

E4C02001 = 1110 0100 1100 0000 0010 0000 0000 0001 (STRB)

E3520000 = 1110 0011 0101 0010 0000 0000 0000 0000 (CMP)

1AFFFFFB = 0001 1010 1111 1111 1111 1111 1111 1011 (CMP)

E1A0F00E = 1110 0001 1010 0000 1111 0000 0000 1110 (MOV)

위 명령어에 맞는 ARM Instruction Set Format을 보면 Disassembly에서 명령어에 대한 32bits Format이 왜 저런 형태로 나오는지 이해할 수 있습니다. 아래 그림은 ARM Instruction Set Format의 그림입니다.

31 28 16 8 0

ARM Instruction Set Format										
Cond	0 0	I	Opcode	S	Rn	Rd	Operand2			
Cond	0 0 0 0 0 0	A	S	Rd	Rn	Rs	1 0 0 1	Rm		
Cond	0 0 0 0 1	U	A	S	RdHi	RdLo	Rs	1 0 0 1	Rm	
Cond	0 0 0 1 0	B	0 0	Rn	Rd	0 0 0 0	1 0 0 1	Rm		
Cond	0 1	I	P	U	B	W	L	Rn	Rd	Offset
Cond	1 0 0	P	U	S	W	L	Rn	Register List		
Cond	1 0 1	L	Offset							

기본적인 Instruction Set Format으로 첫번째부터 순서데로 Data processing/PSR Transfer, Multiply, Long Multiply, Swap, Load/Store Byte/Word, Load/Store Multiple, 마지막으로 Branch에 관련된 Format입니다. Format에 맞춰 보았을 때 작성한 코드에서 첫번째 두번째 LDR의 다른 점으로는 Rd의 값이 다르기 때문에 32bits에서도 Rd를 제외한 나머지 부분은 LDR의 Format을 따르고 있는 것을 확인할 수 있습니다. 또한 LDR과 LDRB의 차이로는 offset과 I, P, U, B, W, L이 다른 값을 가지며 공통적인 Format은 지키는 것을 확인할 수 있습니다.

또한 pseudo instruction은 실질적으로 프로그램에서 인식할 때 사용자가 작성한 코드형태가 아니라 변경되어 프로그램이 인식할 수 있는 코드로 변경되는데 위 코드 중 LDR r0, Address 같은 경우 변경된 형태가 R0, [PC, #0x001C]로 변경되는 것을 확인할 수 있습니다. 두번째로 나온 코드가 어셈블러가 인식하는 코드로써 주소의 값을 직접 다루어 동작하는 것을 확인할 수 있습니다. 같은 예로 LDRB, STRB같은 경우 사용자는 #1을 사용하여 코드를 작성하지만 어셈블러는 #0x0001과 같은 형태로 값을 바꾸어 인식하는 것을 확인할 수 있습니다.

4. 고찰 및 결론

A. 고찰

코드 작성에는 어려움이 없었던 실습입니다. 해당 실습의 목적은 pseudo instruction과 Disassembly를 통한 32bit 코드의 이해와 해석이 목적이기 때문입니다. Disassembly를 통하여 프로그램에서 pseudo instruction이 실제 프로그램에서 쓰이기 위하여 변경되는데 변경된 instruction이 무엇을 의미하는지 처음 해석할 때는 수월하지 않았습니다. 하지만 각각 코드의 특징을 살펴보았을 때 단순한 값으로 인식하는 것이 아니라 항상 사용하였던 주소의 형태로 값을 변경하여 접근하는 것이 보였고 LDR같은 경우는 PC값이 프로그램 동작에 관여한다는 것을 알게 되었습니다.

B.결론

어셈블리 과목을 배우면서 instruction format은 학기 초에 배운 내용이었습니다. 솔직히 말해서 당시에는 완벽하게 이해하지 못하고 언뜻 무엇을 의미하는지 정도만 알고 있는 상태였습니다. 해당 실습을 통하여 Format에 대한 구조와 뜻을 조금 더 명확하게 알게 되었으며 pseudo instruction의 동작과 실질적으로 변경되어 프로그램에 전달되는 코드도 배우게 되었습니다. 이번 실습으로 배운 이론만으로는 다른 곳에 응용하고 사용하기는 어렵다고 예상되지만 이런 개념을 통하여 개발자가 개발하게 되면 프로그램에 대해 좀더 깊게 생각하며 개발할 수 있는 계기가 될 수 있다고 생각 합니다.

5.참고문헌

이준환/ARM instruction set - Pseudo instructions/광운대학교/2018

이준환/ARM Instruction Set – Instruction/광운대학교/2018