어셈블리 프로그래밍 설계 및 실습
보고서

실험제목: Report Sample

실험일자: 2018년 09월 13일 (목)

제출일자: 2018년 09월 20일 (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

Data Transfer to or from MEM

B. 목적

어셈블리어의 기본적인 명령어 사용을 high level의 언어로 사용했을 때와 비교하며 예제를 통해 이해한다. 또한 ARM의 조건부 실행 코드와 언어를 이해한다. 특히 원하는 데이터를 메모리에 저장하고 가져오는 것을 어셈블리 프로그래밍을 통하여 익힌다.

2. 설계 (Design)

A. Pseudo code

첫번째 코드

(1). 비교 대상이 될 데이터를 레지스터 r4에 저장 해준다.
(2). r4에 저장된 데이터가 10보다 크면 r5에 1 저장 후 종료, 아니면 다음 조건
(3). r4에 저장된 데이터가 10보다 작으면 r5에 2 저장 후 종료, 아니면 다음 조건
(3). r4에 저장된 데이터가 10이면 r5에 3 저장 후 종료

결과 : 조건에 따라 r5으 값이 바뀌는 것을 확인 할 수 있다.

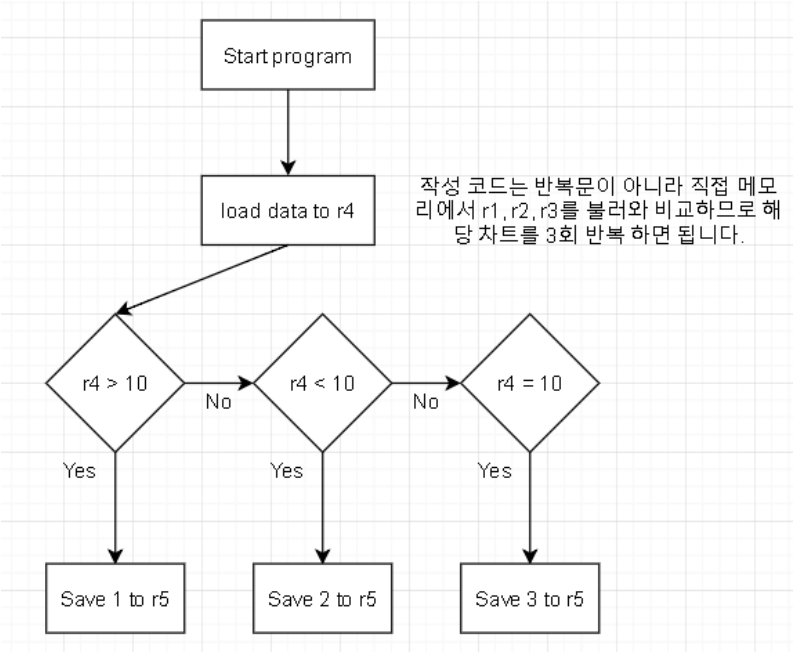
두번째 코드

(1). 메모리에 저장할 내용인 01, 02, 03, 04를 순서데로 r0, r1, r2, r3에 저장
(2). 레지스터 r4에 메모리를 할당해줌(데이터를 저장 할 수 있는 공간을 줌)
(3). 각 레지스터에 저장 되어있는 정보를 r4메모리 공간에 순서데로 저장
(4). r4 메모리에 저장된 정보를 r5에 한번에 불러와 저장.
(5). 각 레지스터에 저장 되어있는 정보를 r4메모리 공간에 역순으로 저장
(6). r4 메모리에 저장된 정보를 r6에 한번에 불러와 저장.

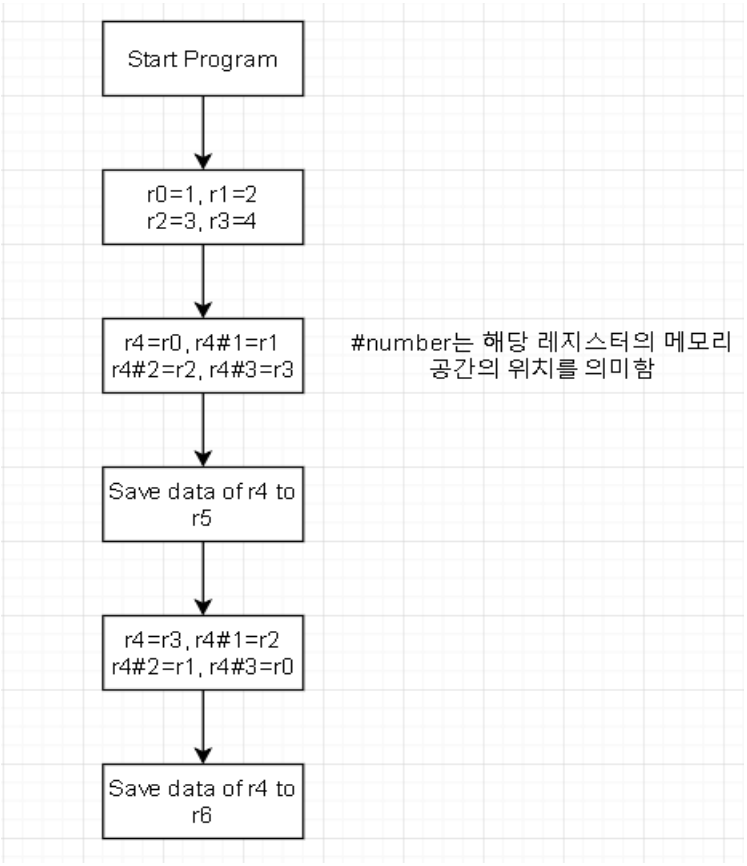
결과 : r5와 r6에 저장된 정보를 보면 역순으로 저장된 것을 확인 할 수 있다.

B. Flow chart 작성

첫번째 코드

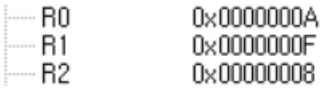


두번째 코드

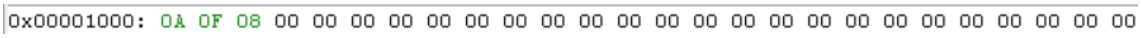


C. Result

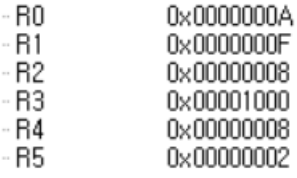
첫번째 코드



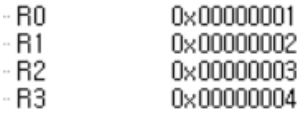
비교 대상이 될 값들을 레지스터에 저장한 화면입니다. 그 후 과제의 조건데로 메모리에서 값들을 load할 수 있도록 아래 사진과 같이 메모리에 값을 저장합니다.



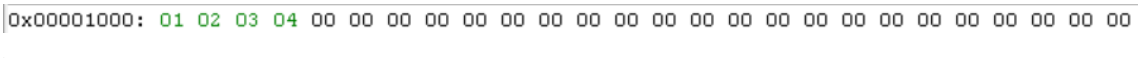
그 후 메모리에 저장된 정보를 r4에 불러와 비교 후 조건에 맞는 값을 아래 이미지와 같이 r5에 저장합니다. 같은 방법으로 메모리에 있는 값들을 순차적으로 비교한 모습입니다.

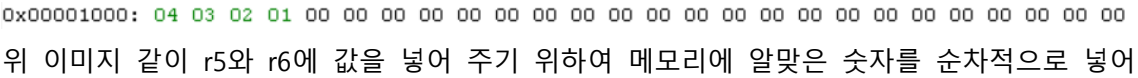


두번째 코드



R5와 R6에 01020304의 값과 04030201의 값을 저장하기 위하여 각 레지스터에 필요한 값을 넣은 모습입니다.





위 이미지 같이 r5와 r6에 값을 넣어 주기 위하여 메모리에 알맞은 숫자를 순차적으로 넣어 준 모습입니다.

정보를 불러온 결과 r5에 04030201이 save되어있고 r6에 01020304의 값이 들어가 있는 것을 확인 할 수 있습니다.

3. 고찰 및 결론

A. 고찰

어셈블리언어를 처음으로 프로그래밍한 계기가 되었습니다. 모든 동작이 낯설고 익숙하지 않았습니다. 좀더 높은 레벨의 언어에서는 쉽게 이해 할 수 있고 많은 동작을 하지 않기 때문에 이해가 쉽지만 간단한 덧셈을 하는 과정도 어셈블리언어가 더 많기 때문에 시간이 좀 걸린 것 같습니다. 또한 조금 아쉬운 점으로는 반복 적으로 수행 할 수 있도록 해주는 명령문을 몰라서 각 레지스터에 값을 수동적으로 넣어주기 때문에 코드의 사이즈가 커지는 문제점이 있다고 생각했습니다. 이에 대한 해결 방안은 C언어의 for나 while과 같은 명령어를 구현하여 코드 사이즈를 줄일 수 있으면 좋다고 생각하였습니다. 그러면 좀더 효율적으로 프로그램을 구현할 수 있다고 생각됩니다.

B. 결론

어셈블리어에서는 변수로 사용하는 레지스터의 개수가 한정적으로 정해진 것을 확인 할 수 있습니다. 이에 따라 좀 더 복잡한 프로그램을 설계하기 위해서는 한계가 있다는 것을 알게 되었고 이를 해결하기위해 메모리에 정보를 주거나 메모리에서 정보를 가져오는 방법을 익히는 계기가 되었습니다. 메모리에서 정보를 주고 받고 하는 방식을 좀 더 익숙하게 사용 할 수 있다면 레지스터의 개수가 13개로 정해져 있다고 해도 충분히 자유롭게 프로그램을 설계 할 수 있다고 생각합니다.

4.참고문헌

이준환 / 어셈블리 프로그램 설계 및 실습 / 광운대학교 / 2018