어셈블리프로그램설계및실습 보고서

과제 주차: 7주차

학 과: 컴퓨터공학과

담당교수: 이형근 교수님

실습분반: 화요일 6, 7

학 번: 2021202058

성 명: 송채영

제 출 일: 2022.10.31(월)

1. Problem Statement

레지스터와 메모리간 블록 단위의 데이터를 저장하고 가져오는 것에 대해 이해한다. 어셈블리어의 서브 루틴(함수)의 사용 방법에 대해 익힌다. 프로그래밍 수준에서 스택 명령어의 필요성과 효용성에 대해 이해하고 응용해본다. 여러 종류의 스택 저장 방식에 대해 이해하고 각 방식에 따른 메모리 접근 방법을 실질적으로 확인해보며 그 차이를 이해한다.

2. Design

LDM은 Load memory pointed by base register to multiple registers를, STM은 Store multiple register to memory pointed by base register를 의미한다. Addressing mode에는 IA, IB, DA, DB가 있으며 I는 increase를 D는 decrease를 의미하고. 주소 증가의 방향을 뜻한다. Stack에는 Full ascending, Full descending, Empty ascending, Empty descending이 있으며, Ascending은 낮은 주소에서 시작해 높은 주소로 가는 것을 의미하고 descending은 높은 주소에서 시작해 낮은 주소로 가는 것을 뜻한다.

-problem 1

Address1에 메모리를 저장한다. -> r0, r1, r2에 각각 1, 1, 11을 저장한다. (이때 11을 저장하는 이유는 비교해주기 위해서이다.) -> r3 = r0 * r1, r0 = r0 + 1, r3 = r1의 과정을 반복한다. (r0가 11이 될 때 까지) -> r0가 11이면 r3의 값을 stack에 저장한다. -> 프로그램을 종료한다.

-problem 2

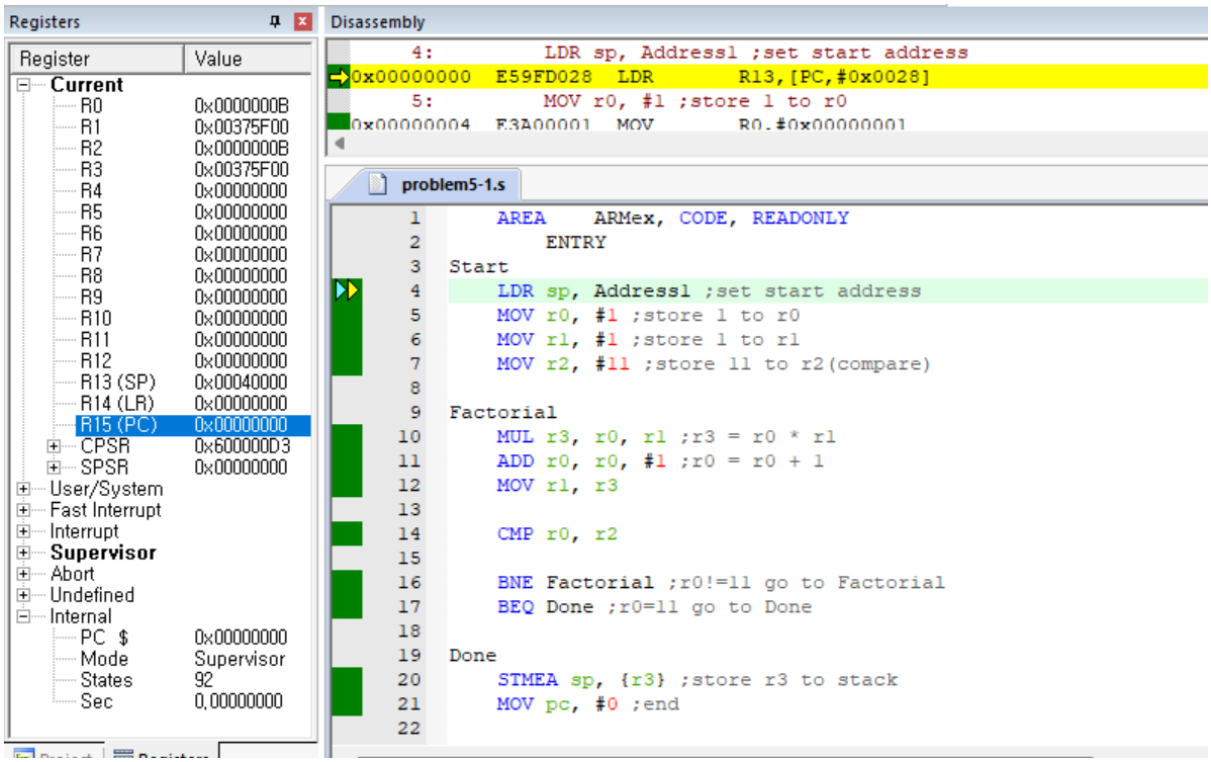
Address1에 메모리를 저장한다. -> r0부터 r7까지 각각 0에서 7을 저장한다. -> 저장된 값을 stack에 push한 후 레지스터의 바꾼 값을 불러온다. -> 프로그램을 종료한다.

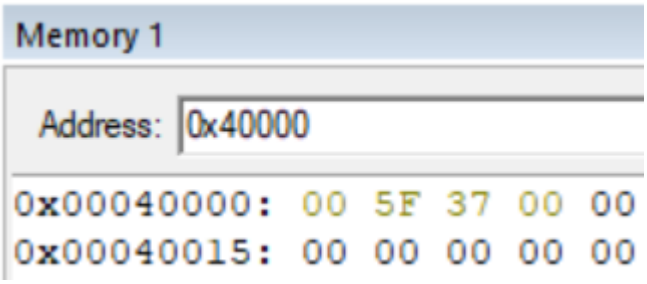
- Problem 3

Address1에 메모리를 저장한다. -> r0부터 r7까지 각각 10에서 17을 저장한다. -> r8에 160을 저장한다. -> 저장된 값을 stack에 push한다. -> doregister함수에서 index값을 더해준다. -> r9 레지스터에 값을 더해준다. -> r9와 r8이 같을 때 까지 doGCD의 과정을 반복한다. -> r9와 r8이 같으면 Done으로 가 r9에 r4를 더해 메모리에 저장한다. -> 프로그램을 종료한다.

3. Conclusion

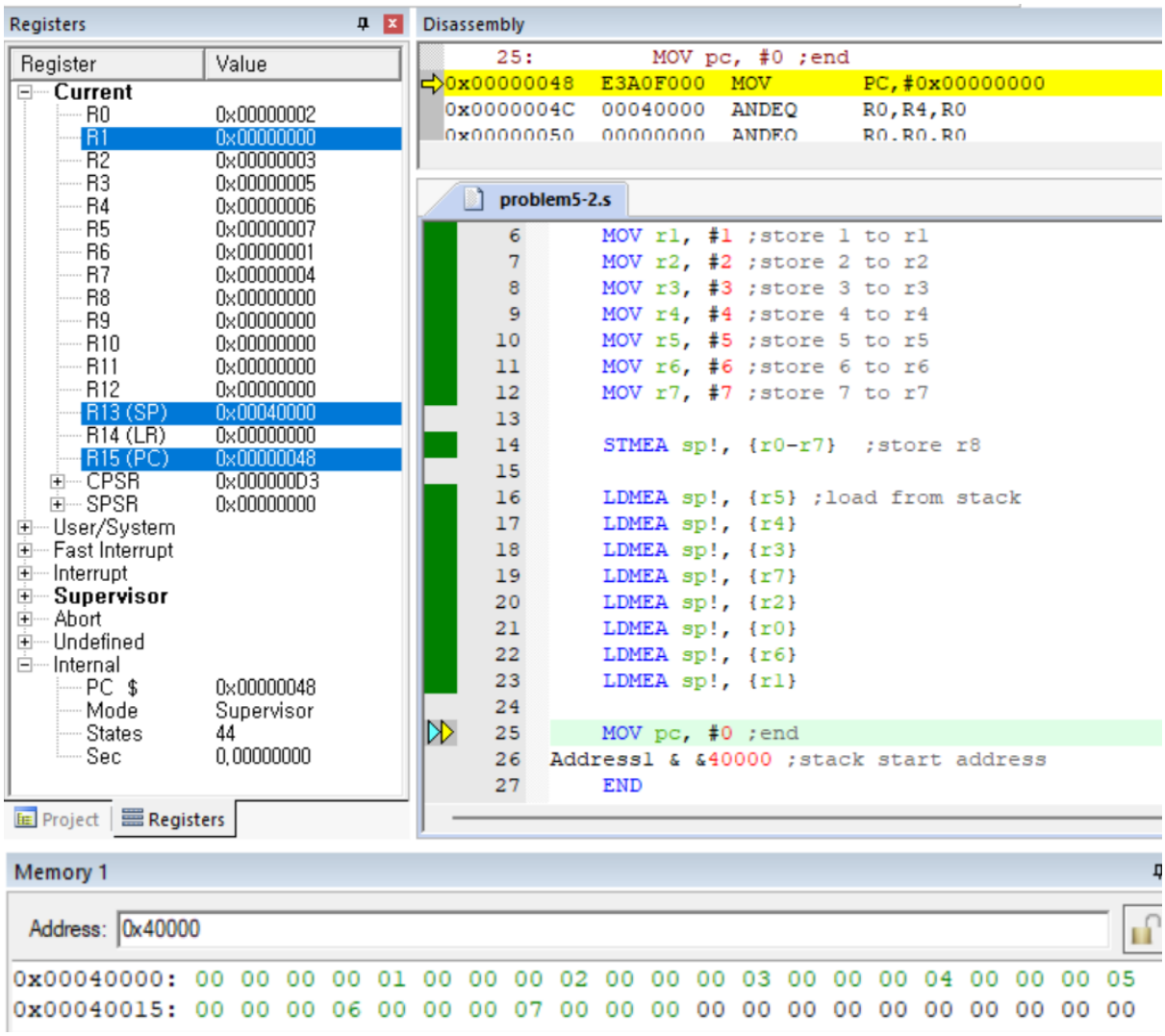
- problem 1





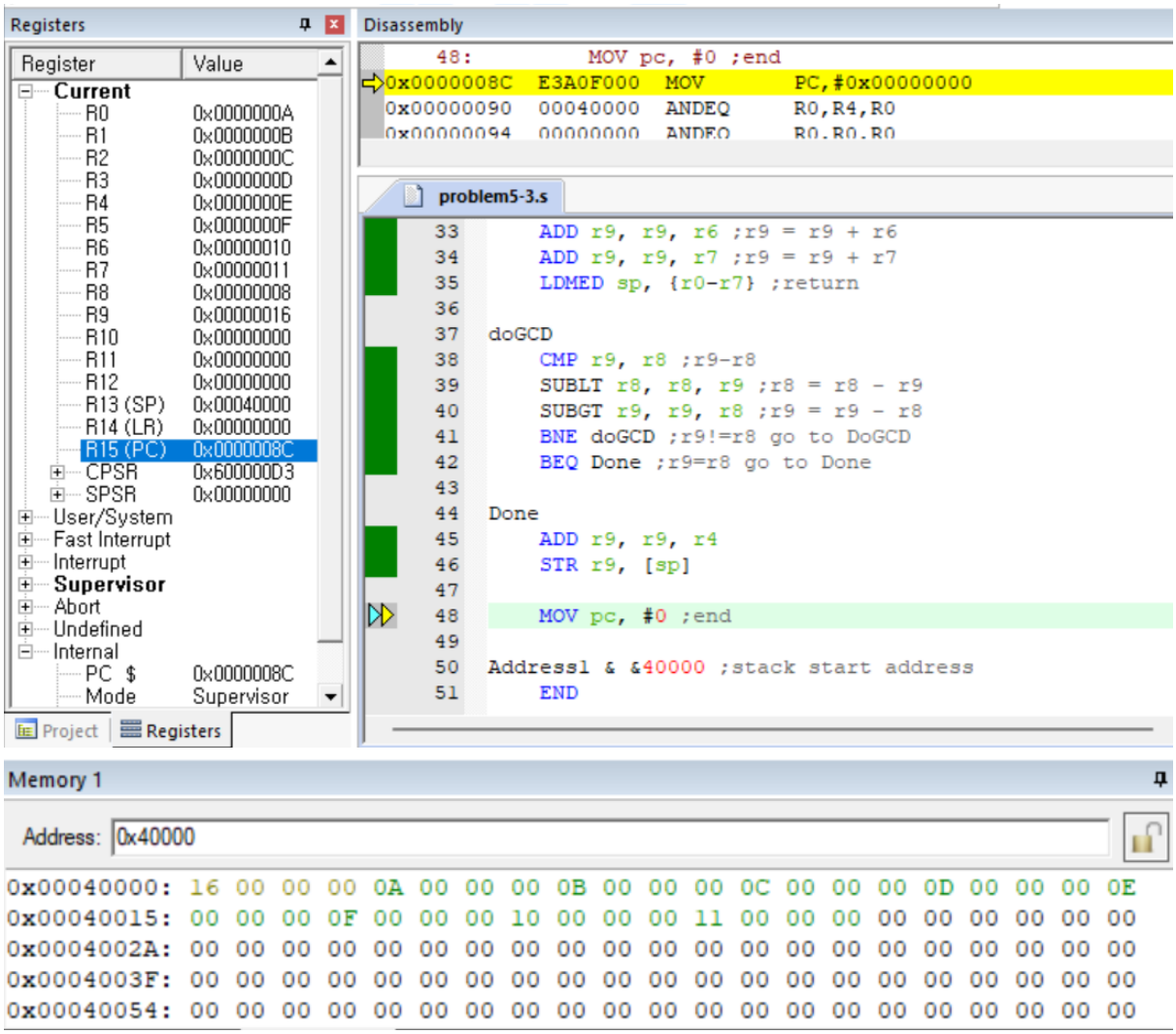
스택과 재귀함수를 이용하여 r3에 10!의 값인, 375f00이 메모리에 저장된 것을 볼 수 있다. 밑에 사진에서도 마찬가지로 40000번지에 결과가 저장된 것을 확인할 수 있다.

- problem2



r0 부터 r7까지 바뀐 값인 r5에는 7, r4에는 6, r3에는 5, r7에는 4, r2에는 3, r0에는 2, r6에는 1, r1에는 0이 저장된 것을 확인할 수 있다. 마찬가지로 40000번지에 결과가 저장된 것을 확인할 수 있다.

- Problem 3



r0부터 r7에 10부터 17이, r8에는 160이 저장되어 있는 것을 알 수 있다. 또한 메모리 주소 즉 40000번지에 r9와 r4를 더한 값이 저장되어 있는 것을 알 수 있다.

4. Consideration

Stack을 사용하지 않고 코드를 구현할 수 있었지만, 이번 실습 과제의 목표는 stack을 사용하는 것이기 때문에 r13, sp를 활용하였다. FA, FD, EA, ED의 개념이 헷갈려 개념부터 공부하고 시작하였다. 시험 전에 이해가 안됐던 부분이었는데 실습을 하면서 이해가 된 것 같아 시험 전에 미리 실습을 진행해 보지 못한 점이 아쉬웠다. 하지만 강의 자료를 공부하고 실습을 진행하니 수월하게 진행할 수 있었다. 2번 과제를 진행하면서, 처음에는 LDR 명령어를 이용해 레지스터에 메모리를 저장해주었는데 이때 메모리는 40000번지에 저장하였다. 그 결과 40000번지에 제대로 출력이 되지 않았다. 40020번지로 저장했을 경우에는 값이 제대로 출력이 되었다. 차이를 잘 모르겠고, stack을 사용하여 짜보고 싶어 stack을 이용하여 과제를 수행하니 40000번지에 제대로 저장이 되는 것을 확인할 수 있었다.

5. Reference

이형근 교수님/어셈블리프로그램설계및실습/광운대학교(컴퓨터정보공학부)/2022