# 어셈블리프로그램설계및실습 보고서

## 과제 주차: 4주차


학 과: 컴퓨터공학과

담당교수: 이형근 교수님

실습분반: 화요일 6, 7

학 번: 2021202058

성 명: 송채영


제 출 일: 2022.09.29(목)

## 1. Problem Statement

기본 명령어를 사용하는 예제를 통해 어셈블리어 프로그래밍을 이해한다. ARM 조건부 실행 코드를 보는 방법을 이해하고 이를 어셈블리어 프로그램으로 구현하는 능력을 습득한다. 또한 원하는 데이터를 메모리로 저장하고 가져올 수 있도록 어셈블리어 프로그래밍 능력을 습득한다.

## 2. Design

이번 과제에서는 CMP, LDR, STRB, LDRB 명령어를 사용하였다. 우선 CMP는 Rn - N을 compare하는 명령어이다. LDR은 register에 word를 load 하는 명령어이다. 다음으로 STRB는 registre로 부터 byte를 저장하는 명령어이다. 마지막으로 LDRB는 register에 byte를 load하는 명령어이다.

-problem 1

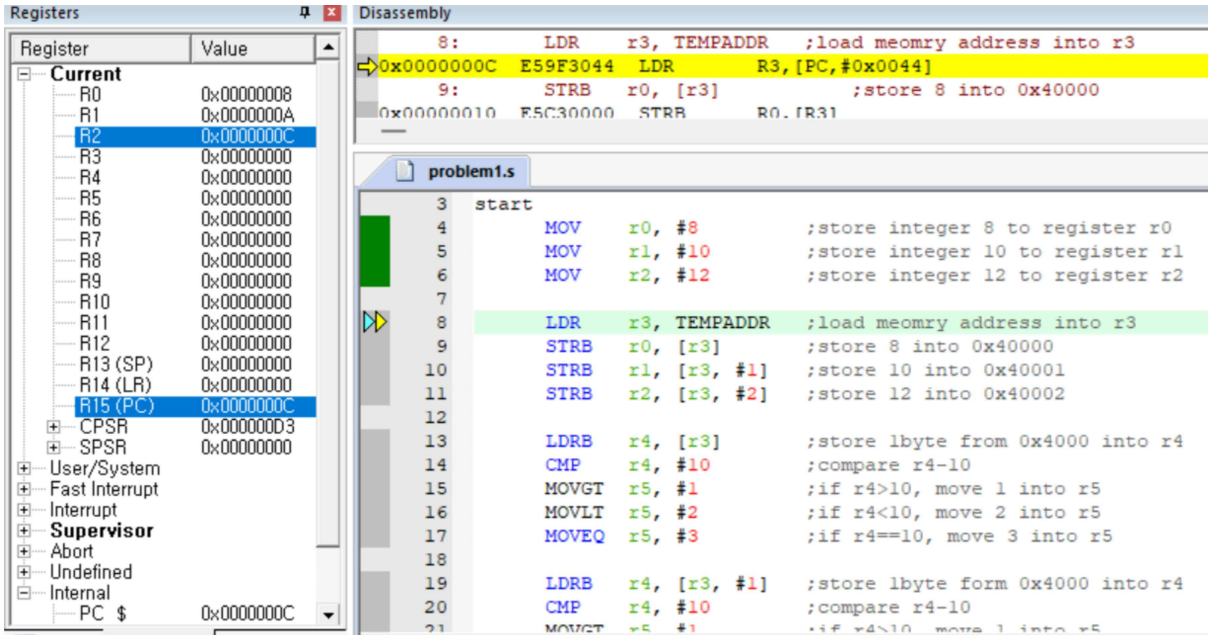
레지스터 r0, r1, r2에 MOV 명령어를 통해 0x0A(10)보다 작은수, 같은수, 큰수를 저장한다. -> TEMPADDR의 값을 40000으로 설정하고 주소값을 r3에 불러온다. -> 메모리에 저장된 값을 r4에 1 byte 단위로 불러와 그 값과 0x0A(10)을 비교한다. -> r4 > 10인 경우 r5에 1을 저장한다. r4 < 10인 경우 r5에 2를 저장한다. r4 = 10인 경우 r5에 3을 저장한다. ->다음의 과정을 반복한다. -> lr의 값을 pc에 저장한다. -> 프로그램을 종료한다. 의 순으로 과제를 수행할 것이다.

-problem 2

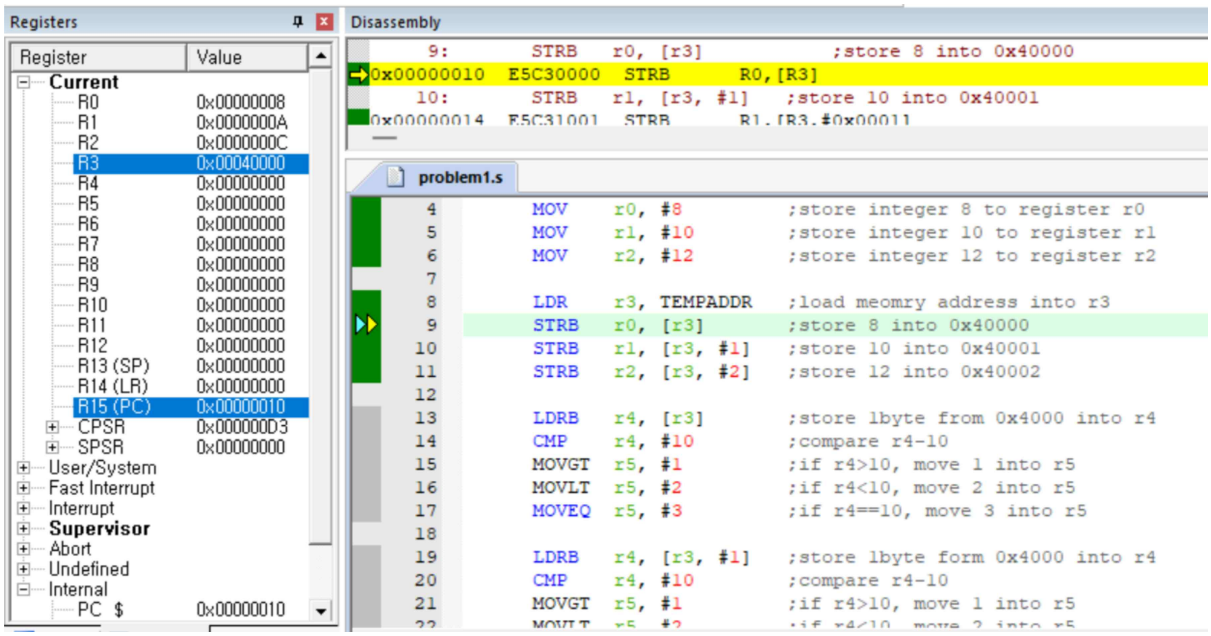
레지스터 r0, r1, r2, r3에 1, 2, 3, 4를 차례대로 저장한다. -> TEMPADDER의 값을 40000으로 설정하고 주소값을 r4에 불러온다. -> 레지스터에 저장된 값들을 r4의 메모리 공간에 저장한다. -> r4에 저장된 값들을 r5로 불러온다. -> r4의 메모리에 역순으로 값을 저장한다. -> r4에 저장된 값들을 r6로 불러온다. -> lr의 값을 pc에 저장한다. -> 프로그램을 종료한다. 의 순으로 과제를 수행할 것이다.

## 3. Conclusion

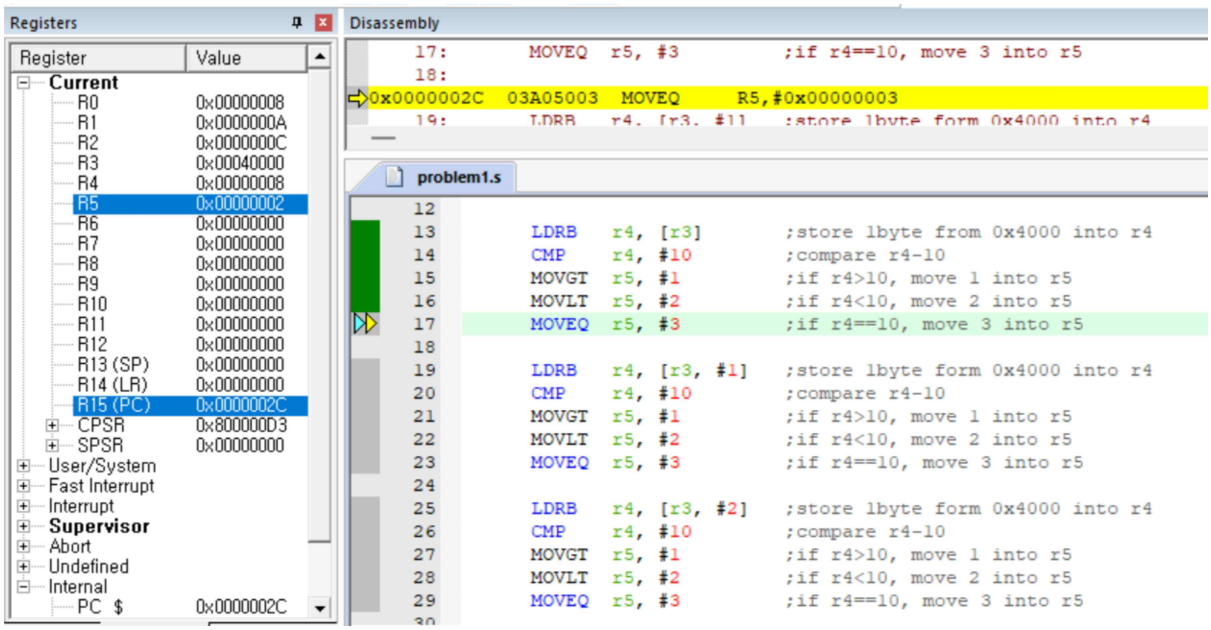
- problem 1



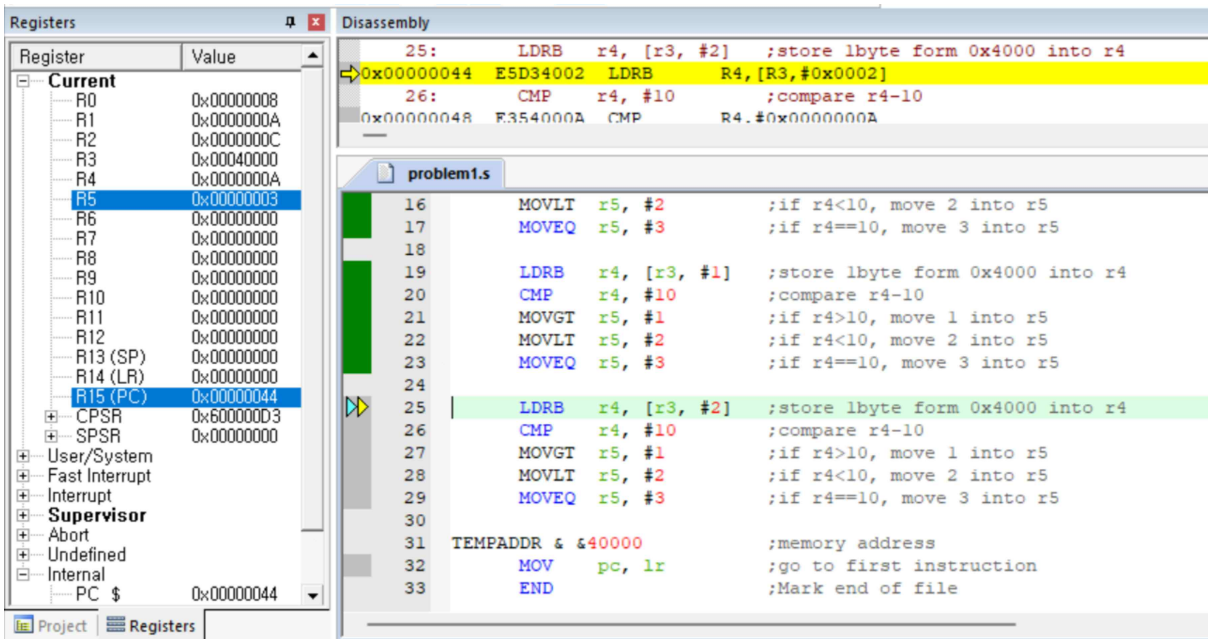

MOV r0, #8, MOV r1, #10, MOV r2, #12를 통해 register r0, r1, r2에 각각 정수 8, 10, 12를 저장한다. R0, R1, R2의 value가 8, A, C로 바뀐 것을 확인할 수 있다.



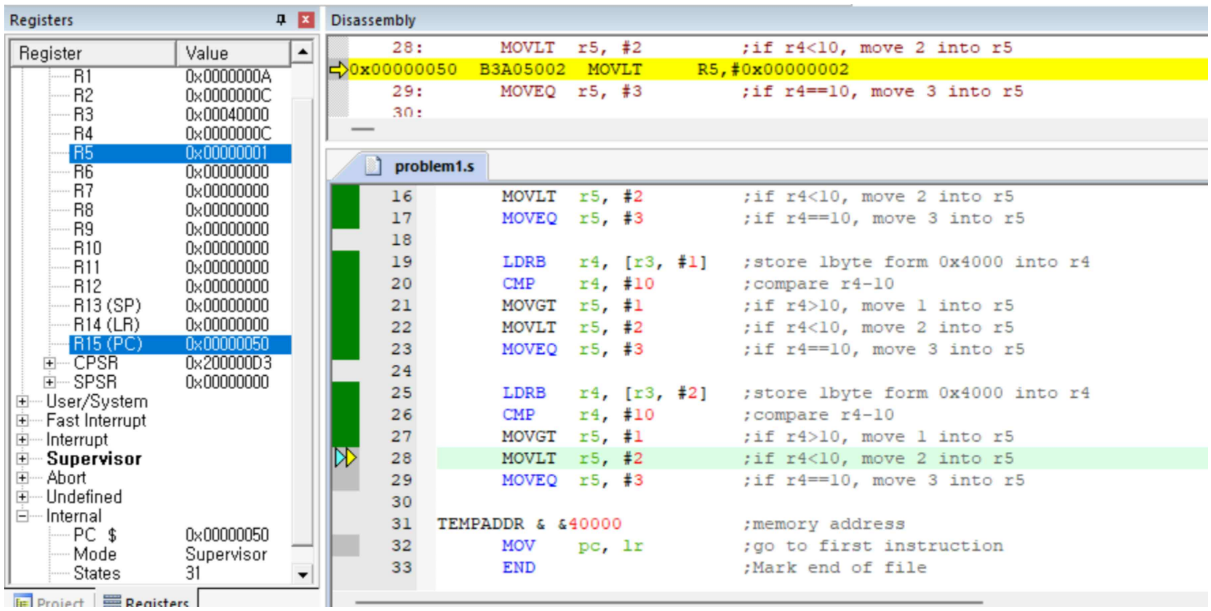

LDR r3, TEMPADDER를 통해 R3에 주소값을 불러온 것을 확인할 수 있다.




r4에 10보다 작은 수가 저장돼 있기 때문에 r5에 2가 저장되는 것을 볼 수 있다.

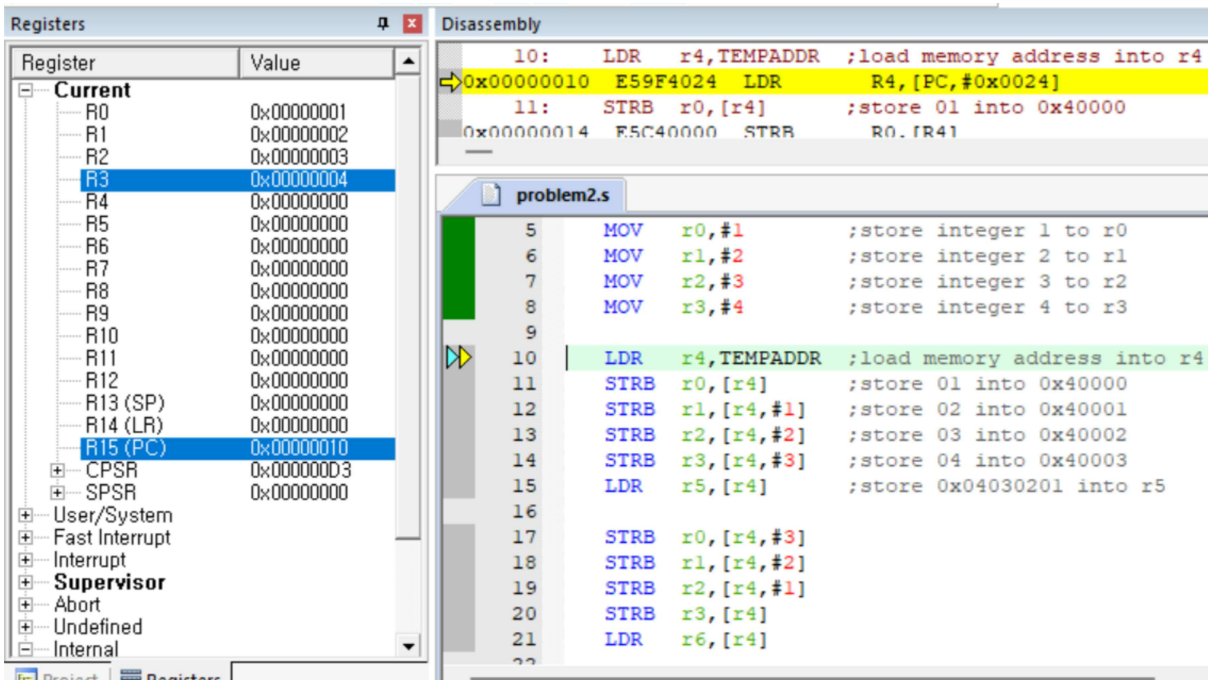



r4에 10과 같은 수가 저장돼 있기 때문에 r5에 3이 저장되는 것을 볼 수 있다.

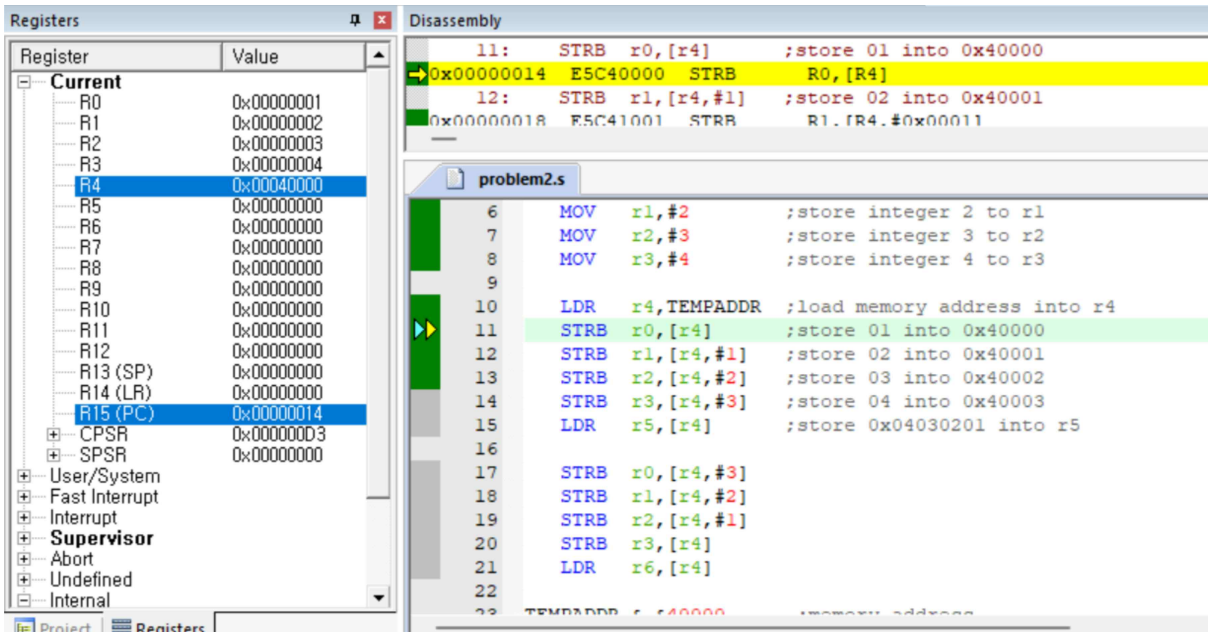



r4에 10보다 큰 수가 저장돼 있기 때문에 r5에 1이 저장되는 것을 볼 수 있다.

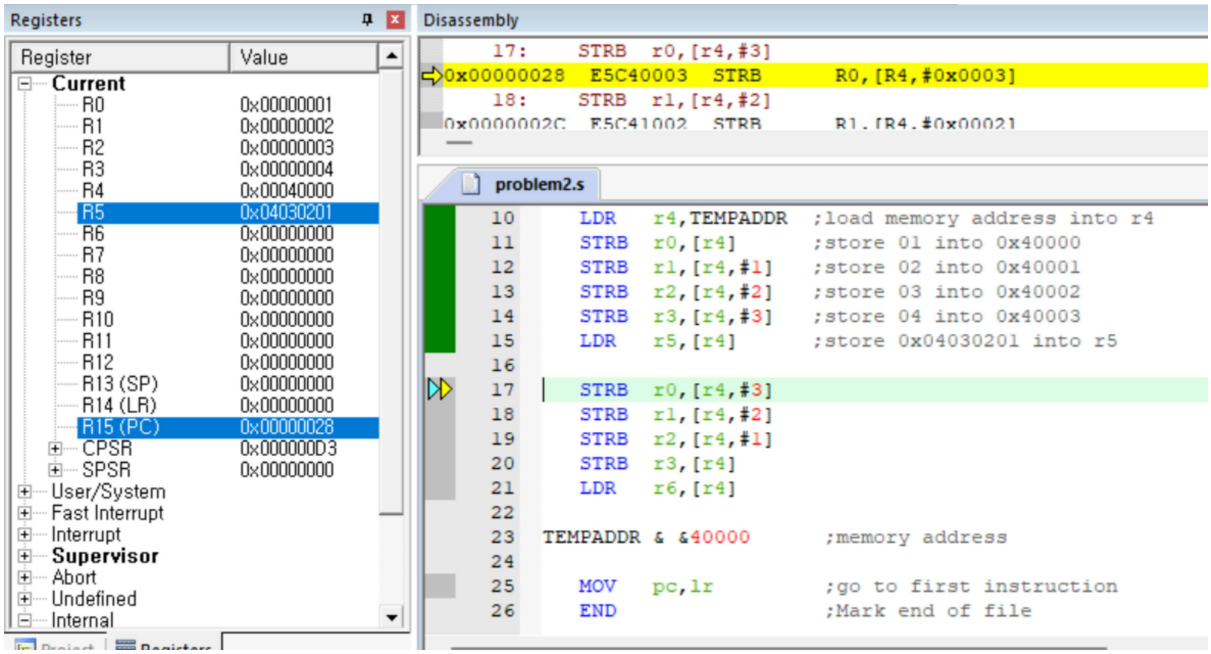
- problem 2



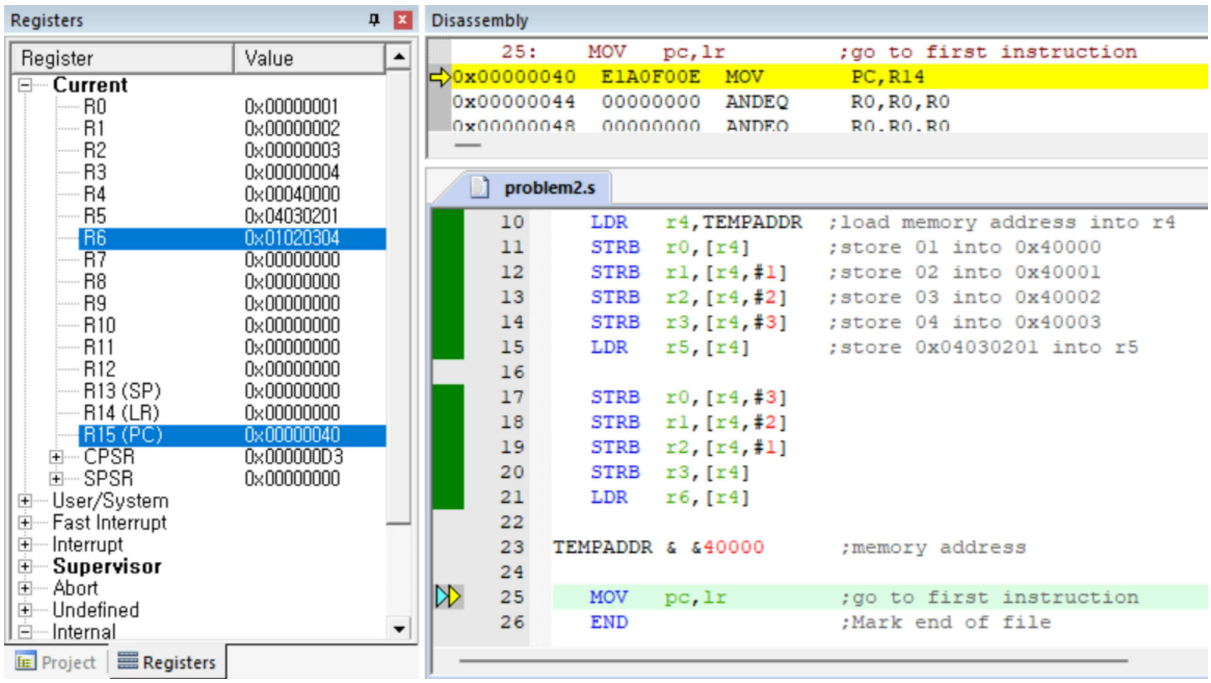

MOV r0, #1, MOV r1, #2, MOV r2, #3, MOV r3, #4를 통해 register r0, r1, r2에 각각 정수 1, 2, 3, 4를 저장한다. R0, R1, R2, R3의 value가 1, 2, 3, 4로 바뀐 것을 확인할 수 있다.




LDR r4, TEMPADDER를 통해 R4에 주소값을 불러온 것을 확인할 수 있다.



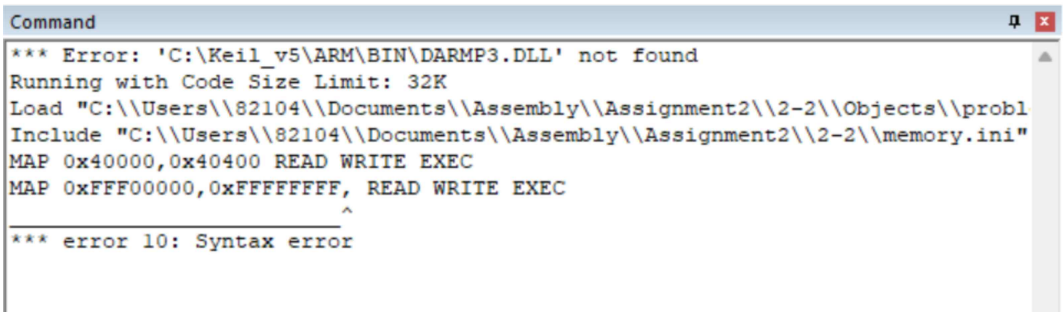

little-endian의 특성을 이용해서, r4의 메모리에 저장된 값들을 r5에 불러왔다. R5에 0x04030201이 저장된 것을 확인할 수 있다.




little-endian의 특성을 이용해서, r4의 메모리에 역순으로 저장한 값들을 r6에 불러왔다. R6에 0x01020304가 저장된 것을 확인할 수 있다.

## 4. Consideration

문제만 보고 코드를 짜기엔 처음 보는 명령어들도 많아 다소 막막했다. example로 나와 있는 예시 코드들을 직접 수행해보면서 코드가 어떻게 돌아가고 명령어가 어떤 역할을 하는지 익힐 수 있었다. 또한 저장되는 것을 직접 확인할 수 있어 이해가 빨리 됐던 것 같다. 전 시간에 배운 Register INI file을 추가하는 것을 깜빡해 65 error가 났었다. 하지만 아래의 사진과 같은 Syntax error가 나 확인해보니 0xFFFFFFFF뒤에 ,를 넣어주어 오류가 나는 것이었다. 이 오류를 통해 Simulation output의 에러 위치가 화살표로 표시 된다는 것을 알 수 있었고, 다음 실습 때는 잘 확인해 같은 오류가 나지 않도록 해야겠다.




## 5. Reference

이형근 교수님/어셈블리프로그램설계및실습/광운대학교(컴퓨터정보공학부)/2022