어셈블리 프로그래밍 설계 및 실습

실험제목: Basic Example

실험일자: 2020 년 09 월 8 일 (화)

제출일자: 2020 년 09 월 13 일 (일)

학　　과: 컴퓨터공학과
담당교수: 이준환 교수님
학　　번: 2019202052
성　　명: 김호성

1. 제목 및 목적

A. 제목
Basic Exapmle

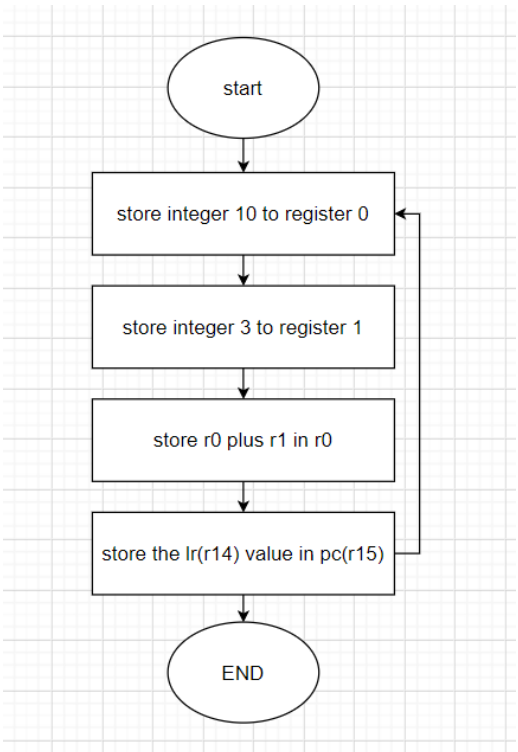
B. 목적
예제 코드를 수행하여 기본적인 동작에 대해 이해하고, register 값을 확인하는 법을 숙지하기 위해 해당 과제를 진행한다.

2. 설계 (Design)

A. Pseudo code

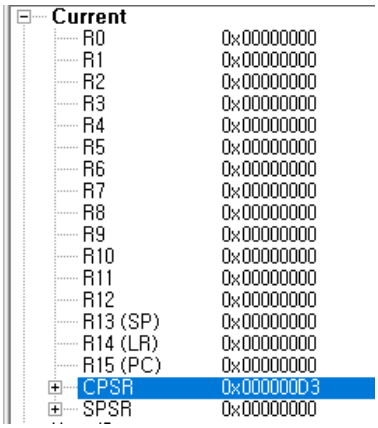
MOV R[0] ← 10
MOV R[1] ← 3
ADD R[0] ← R0, R1
MOV PC ← LR

B. Flow chart 작성

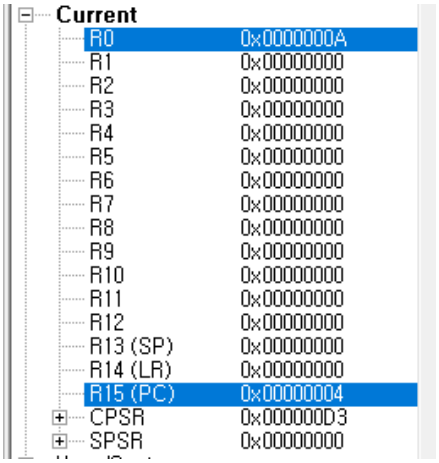


C. Result

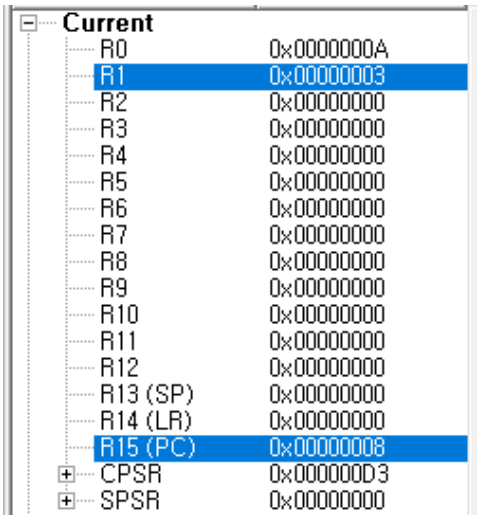
1. 초기 세팅



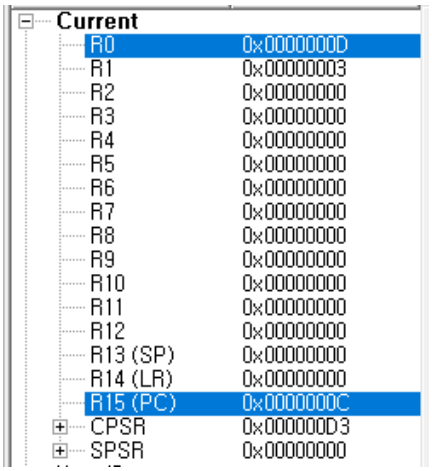
2. MOV R[0] ← 10



3. MOV R[0] ← R[0], R[1]



4. MOV PC ← LR



5. Memory.ini

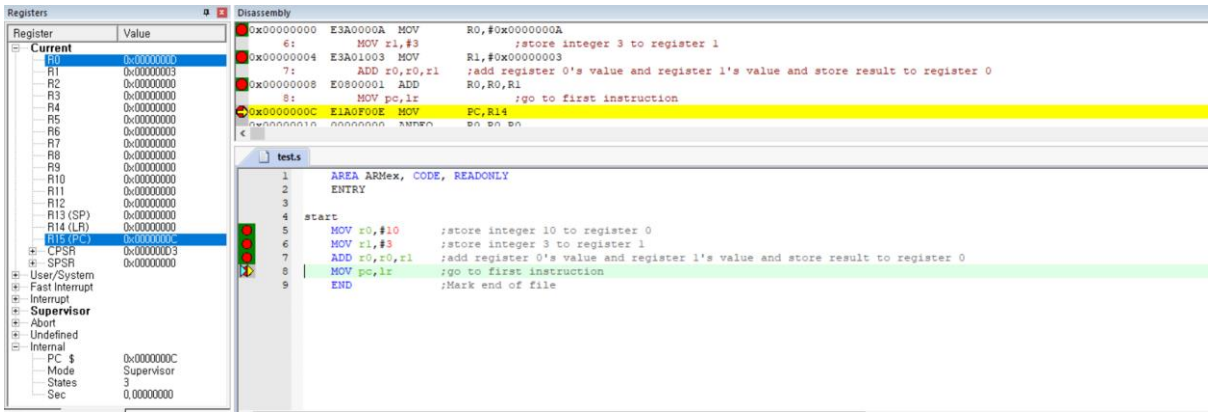
```
Running with Code Size Limit: 32K
Load "C:\\Users\\jjaa7\\Desktop\\2019202052_KimHoseong\\Objects\\2019202052_KimHoseong.axf"
Include "C:\\Users\\jjaa7\\Desktop\\2019202052_KimHoseong\\memory.ini"
MAP 0x40000, 0x40400 READ WRITE EXEC
MAP 0xFFF00000, 0xFFFFFFFF READ WRITE EXEC
```

D. Performance

Performance = Code size * Code size * states

Total RO Size (Code + RO Data)	16 (	0.02kB)
Total RW Size (RW Data + ZI Data)	0 (	0.00kB)
Total ROM Size (Code + RO Data + RW Data)	16 (	0.02kB)

Code size = 16



State = 3

결론: performance = 16*16*3 = 768

3. 고찰 및 결론

A. 고찰

- 레지스터의 value 의 경우 16 진수를 사용한다. [10 = A, 13 =D]

- MOV pc lr 은 이번 과제에선 코드의 진행을 가장 처음으로 돌리는 역할을 한다.

- states 의 경우 | 진행 전 pc 의 value – 진행 후 pc 의 value | /4 만큼 증가한다.

B. 결론

- MOV pc lr 의 의미는 r14(lr) 값을 r15(pc)에 저장하라는 명령이다. 여기서 pc 는 program counter 의 약자로 다음에 수행할 명령어의 주소를 가지는 instruction 으로, pc 의 value 를 제어하지 않을 경우 자동으로 value 값이 4 씩 증가하면서 코드를 진행한다. 여기서 lr 의 value 는 처음 초기화된 0x00000000 값을 가지고 있고, pc 의 value 에 lr 의 value 가 저장되면서 무한루프가 형성된다.

- 어셈블리코드 문법의 규칙은
  label <space> opcode <space> operands <space> ; comment
  이 순서를 지켜줘야 한다. 그래서 코드를 보면 label 을 쓰지 않더라도 tab 이나 스페이스바를 눌러 label 의 공간을 확보한 것을 알 수 있다.

4. 참고문헌

- Basic_Example.pdf
- 1.MDK-ARM_Setup_&_Basic_Example.pdf
- MDK-ARM_Setup_&_Basic_Example_ver1.pdf