# 어셈블리프로그래밍설계 및 실습 보고서

## 실험제목: **Control flow & data processing**

실험일자: 2023년 10월 10일 (화)

제출일자: 2023년 10월 16일 (월)


학 과: 컴퓨터공학과

담당교수: 이준환 교수님

실습분반: 화 6,7 목 5

학 번: 2020202037

성 명: 염정호

1. 제목 및 목적 (3%)

A. 제목

Control flow & data processing

B. 목적

이번 실험에서는 레지스터를 이용한 사칙 연산을 구현 본다. 또한 loop를 이용해 입력받은 문자열의 크기를 구해주는 stlen함수를 구현한다. 1부터 10까지의 수를 덧셈하는 프로그램을 3가지 방법으로 작성해 보고 state수와 memory사용량을 비교한다.

2. 설계 (Design) (50%)

A. Pseudo code

**Problem1**

```
                AREA ARMex, CODE, READONLY
        ENTRY

start
        LDR r4, TEMPADDR1
        mov r0, #5 ; a
        mov r1, #4 ; b
        mov r2, #3 ; c
        mov r3, #2 ; d

        sub r5,r0,r1        ; r5 = a -   b
        mul r6,r3,r3        ; r6 = d * 2
        add r6,r6,r2        ; R6 = c+ (d * 2)
        add r7,r5,r6        ; R7 = (a-b) + (c+(d*2))

        str r7,[r4]
TEMPADDR1 & &00040000

END
                end
```

## Problem2

```
                AREA ARMex,CODE,READONLY
        ENTRY
start
        LDR r4, TEMPADDR1
        LDR r0, =string
        MOV R1, #0
    BL strlen ; Return after loop operation
        str R1,[r4]
        B EXIT
strlen
        LDRB R2, [R0], #1
        CMP R2, #0 ; Compare if the string is zero
        ADDNE R1, R1, #1 ; Add if the character is non-zero
        BNE strlen; Loop Repeat
        BX LR
EXIT

TEMPADDR1 & &00040000
string DCB "HELLO",0 ;Save string in strlen

END
        end
```

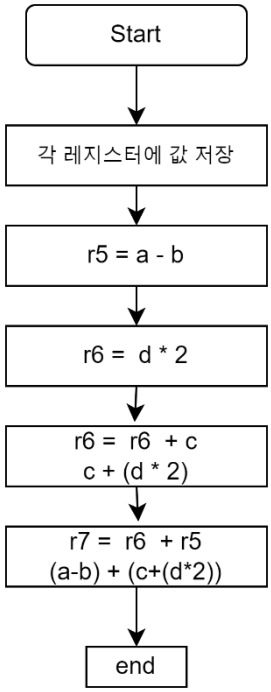
## Problem3

```
                AREA ARMex,CODE,READONLY
        ENTRY
start
        LDR r4, TEMPADDR1
        MOV r0, #0
        MOV r1, #1
    BL for_loop ; Return after loop operation
        B done

for_loop
        cmp r1, #11
        BEQ done
        ADD r0,r1,r0
        ADD r1,r1,#1
        B for_loop
done
        str r0,[r4]
TEMPADDR1 & &00040000

END
        end
////////////////////////////////////////////////////////////////////////////////////////////////

                AREA ARMex,CODE,READONLY
        ENTRY
start
        LDR r4, TEMPADDR1
        MOV r0, #10
        add r1,r0,#1
        mul r0,r1,r0
        LSR r0,r0,#1    ; divide r0,2
```

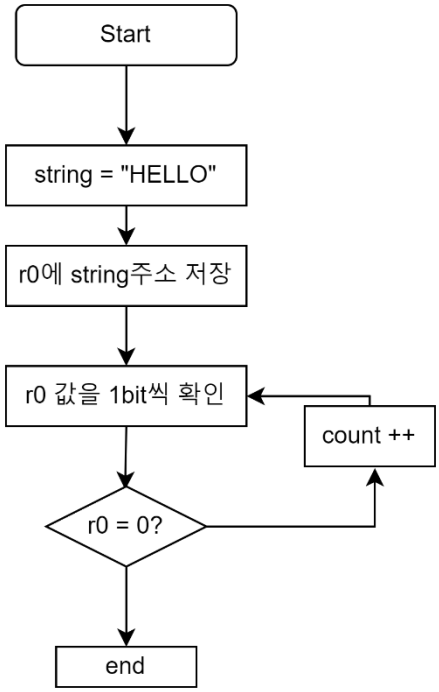
```
        str r0,[r4]
TEMPADDR1 & &00040000
END
        end////////////////////////////////////////////////////////////////////////////////////////
////
                AREA ARMex,CODE,READONLY
        ENTRY
start
        LDR r4, TEMPADDR1
        add r0,r0,#1
        add r0,r0,#2
        add r0,r0,#3
        add r0,r0,#4
        add r0,r0,#5
        add r0,r0,#6
        add r0,r0,#7
        add r0,r0,#8
        add r0,r0,#9
        add r0,r0,#10

                str r0,[r4]
TEMPADDR1 & &00040000
END
        end
```

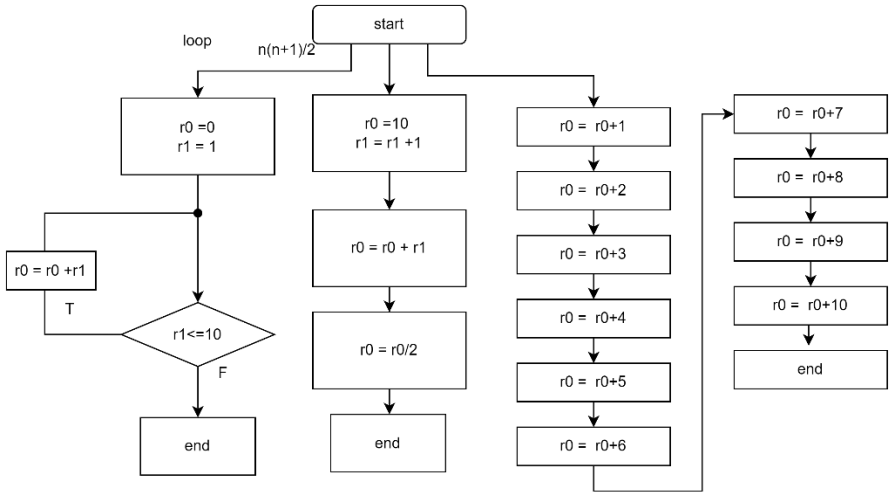
B. Flow chart 작성

Problem1



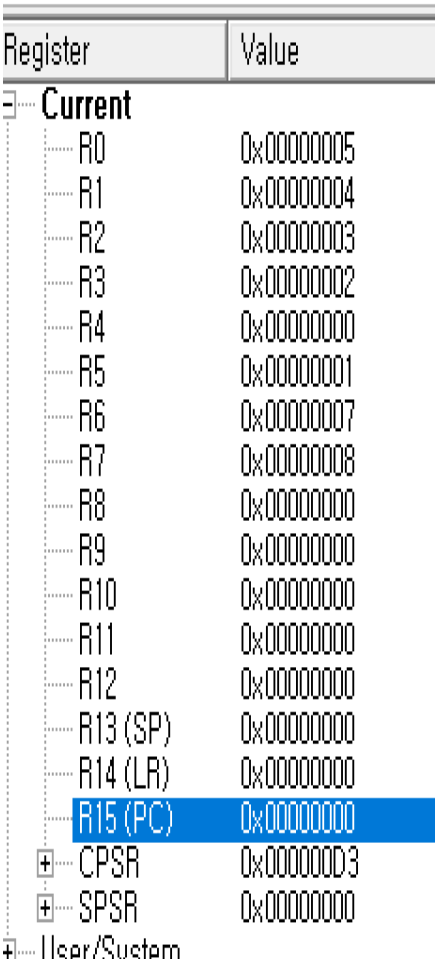

Problem2


Start
string = "HELLO"
r0에 string주소 저장
r0 값을 1bit씩 확인
count ++
r0 = 0?
end

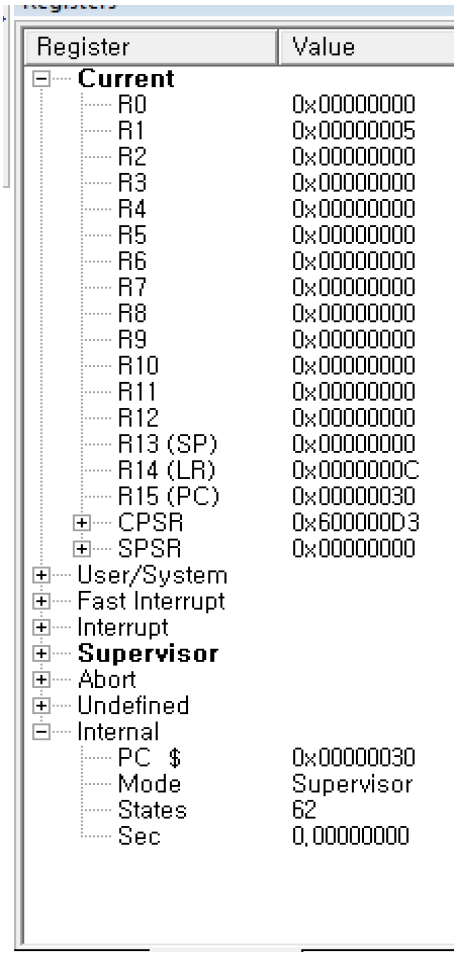

start
loop
n(n+1)/2
r0 =0
r1 = 1
r0 = r0 +r1
T
r1<=10
F
end
r0 =10
r1 = r1 +1
r0 = r0 + r1
r0 = r0/2
end
r0 = r0+1
r0 = r0+2
r0 = r0+3
r0 = r0+4
r0 = r0+5
r0 = r0+6
r0 = r0+7
r0 = r0+8
r0 = r0+9
r0 = r0+10
end

C. Result

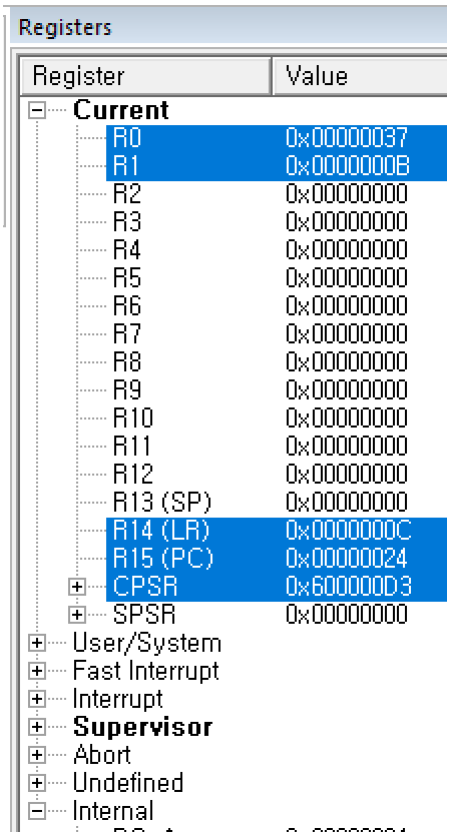
Problem1



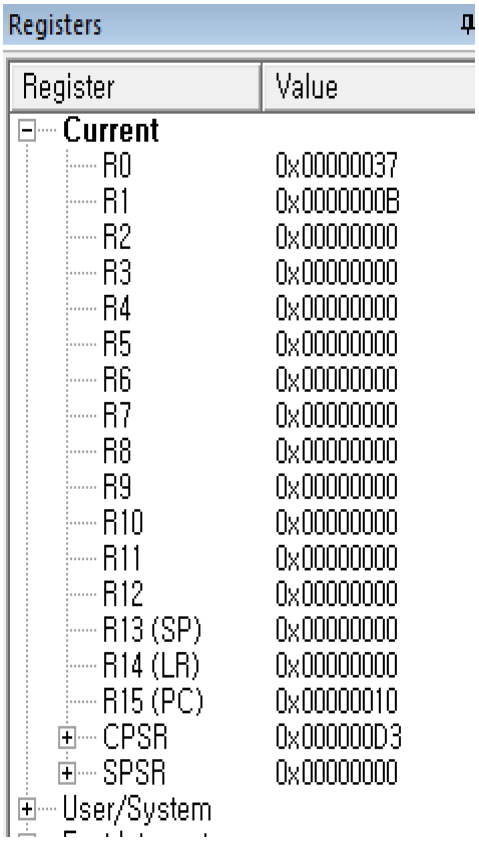

Problem2



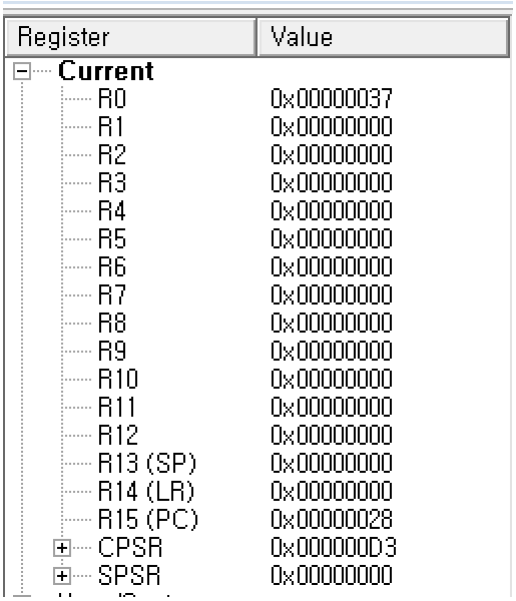
Problem3-1




Problem3-2




Problem3-3



D. Performance

Problem1

Code size 44, state 15

Problem2

Code size 48, state 62

| Problem3-1 | Problem3-2 | Problem3-3 |
| --- | --- | --- |
| Code size 48, state 85 | Code size 28, state 11 | Code size 52, state 16 |

## 3. 고찰 및 결론

### A. 고찰 (35%)

실험 2번을 진행중 막히는 부분이 많았다. 첫번째로 string label를 설정 했을 때 값이 메모리 어딘가 저장된다는 것이었다. 어셈블리 프로그램의 경우에는 당연히 저장할 메모리 주소를 전달해 해당 메모리에 값이 저장될 줄 알았으나 값을 저장 한 후 해당 메모리 위치를 레지스터를 통해 파악해야 했다.

두 번째로 반복문을 실행시킬 때 종료 조건과 종료 시 어디로 가야할지를 정해줘야 한다는 점이 새로운 개념으로 작용해서 Branch함수를 사용하는데 어려움이 있었다,.'

### B. 결론 (10%)

. 이번 실험을 통해 루프 문의 활용 방법을 배울 수 있었다. 또한 실제 c에 사용되는 코드를 어셈블리어로 구현해 보니 C++이 상당한 high level language처럼 느껴졌다.

실험 2번에서 메모 접근과, 레지스터 활용하는 부분이 많아서 state가 크게 나올 것이라 예상했는데 실험 3-1번의 state수가 더 많았다. 아마 state수는 코드가 실행되는 횟수이고 메모리 참조는 속도와 관련이 있기 때문에 이러한 결과가 나왔다.

3번의 실험에서는 2번이 code size와 state모두 적게 나온 것을 확인 할 수 있었는데 간단한 문제의 경우 반복문을 사용하거나, 여러 식을 나열해서 사용하는 것 보다 함수를 이용해서 한번에 처리하는 것이 더 cost소요가 적다는 것을 배웠다.

## 4. 참고문헌 (2%)

이준환/어셈블리프로그래밍실습/광운대학교/2023