어셈블리 프로그래밍 설계 및 실습

실험제목: Floating Point

실험일자: 2017년 10월 25일 (수)

제출일자: 2017년 11월 1일 (수)

학 과: 컴퓨터공학과

담당교수: 이형근 교수님

실습분반: 월 5, 수 6, 7

학 번: 2014722075

성 명: 이 동 준

1. 제목 및 목적 (3%)
   1. 제목

Floating Point

* 1. 목적

Floating point에 대해 이해하고 표현방법을 알아본다. 그리고 Floating point 숫자 간의 덧셈 방법을 알아보고 설계한다

1. 설계 (Design) (50%)
   1. Pseudo code

load first value

load second value

load address

if(first value == 0)

result = second value

stop()

if(second value ==0)

result = first value

stop()

r3<- sign bit of first value

r4<- sign bit of second value

r5<- exponent of first value

r6<- exponent of second value

r7<- mantissa of first value(set most left bit)

r8<- mantissa of second value(set most left bit)

r7<- make fraction using mantissa of first bit

r8<- make fraction using mantissa of second bit

compare r5,r6 // large exponent - small exponent = shiftnum of small fraction

if(r5 > r6)

r9 = r5 – r6

r8 = r8/2

if(r5 < r6)

r9 = r6 – r5

r7 = r7/2

r5 <- r6

compare r3,r4 // compare signbit

if(r3==r4)

samesignbit()

if(r3!=r4)

notsame()

samesignbit() //case of equal signbit

r2 = r7 + r8 , update carry

if(carry set) exponent++

if(carry clear) make mantissa

set mantissa on mantissa position

set exponent on exponent position

make floating point after addition all of value

store value in memory

stop()

notsame() // case of not equal

compare r7,r8

if(r7>r8) r2 = r7 - r8

if(r7<r8) r2 = r8 – r7

loop //find 1. of fraction

compare r2, 0x80000000

if(r2 < 0x80000000)

exponent—

r2 = r2\*2

compare r2, 0x80000000

if( r2 < 0x80000000)

go to loop

set mantissa on mantissa position

set exponent on exponent position

addition of matissa, signbit and exponent

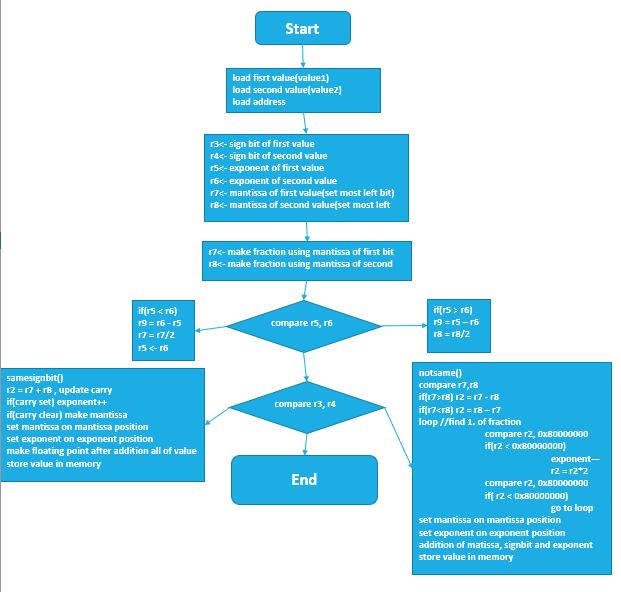
store value in memory

stop()

stop()

end

* 1. Flow chart 작성



* 1. Result

\* case of same sign bit

 r0 – first value, r1 – second value, r2 – result

-> r0 : 1.5, r1 : 3.25 ~> r2 : 4.75

\* case of not same sign bit

 r0 – first value, r1 – second value, r2 – result

-> r0 : 20, r1 : -5 ~> r2 : 15

* 1. Performance

code size : 232 Bytes, state : 55 (case of same sign bit)

1. 고찰 및 결론
   1. 고찰 (35%)

처음에는 Floating point에 대해서 개념이 너무 부족하였다. 심지어 Addition까지 수행해야 하기 때문에 상당한 시간 투자를 통해 설계를 하였다. 각 비트가 나타내는 수들을 어떻게 따로따로 연산을 할 지와 덧셈연산에서 특히 sign bit가 다를 경우는 상당한 난이도가 있었다. 다양한 시도 끝에 가장 적합한 방법으로 설계를 하였다.

Floating point에 대한 개념을 확실히 잡아야 한다. 더불어 logical shift의 활용을 더욱 익혀야겠다.

* 1. 결론 (10%)

Floating point가 가지고 있는 32bit안에서 각 bit들이 나타내는 의미들을 이해하고, 덧셈연산 하나지만 sign bit가 같을 경우와 다를 경우를 나누어 생각을 해야 한다. 즉 양수와 양수, 음수와 음수의 연산과 양수와 음수의 연산을 나누어 다른 방식의 덧셈을 시행해야 한다.

1. 참고문헌 (2%)

광운대학교 – 어셈블리프로그래밍 설계 및 실습 – 이형근교수님 – 강의자료