상명대학교 컴퓨터과학과 "EA0003: 시스템소프트웨어" 실습

- 과제 번호: lab02

- 과제 디렉터리: ~/sys\_sw/lab02 이름: 임현

- 과제 날짜: 2016년 4월 28일

- 과제 제목: ASM IO 라이브러리 학번: 201511054

- 과제 내용:

1. (과제준비) 자신의 홈 디렉터리 아래에 과제 디렉터리 (~/sys\_sw/lab02)를 만든다. 이번 과제에서 만들어지는 모든 파일들은 이 디렉터리에 두도록 한다.



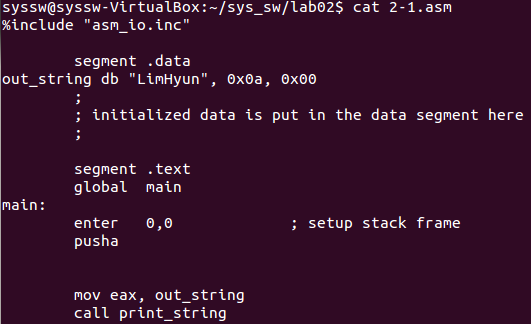
1. (ASM IO 라이브러리) ASM IO 라이브러리의 기능을 확인해보자
   1. ASM IO 라이브러리가 담긴 압축 파일 asm\_io.tar.gz 를 실습 디렉터리로 복사한 후 명령 “gzip –d –c asm\_io.tar.gz | tar xvf –“ 를 사용하여 관련 파일들을 생성하시오 (이 파일들은 이후 실습에서도 계속 필요하니 삭제하지 말 것). 생성된 파일들에 대해 파일 이름, 크기, 간단한 설명을 아래에 적으시오.

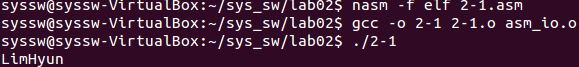


asm\_io.asm, 크기 : 8.972 bytes, 설명 : asm\_io.inc를 사용하는 예제를 보여주는 어셈블리 소스 파일

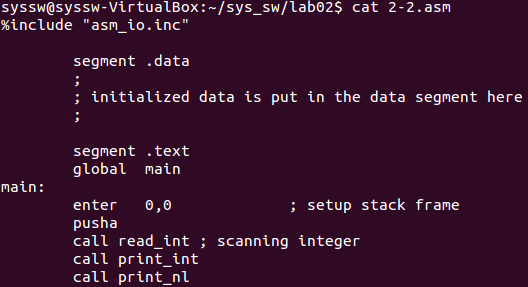
asm\_io.inc, 크기 : 563 bytes, 설명 : Input/Output의 Routines을 나타내는 헤더파일.

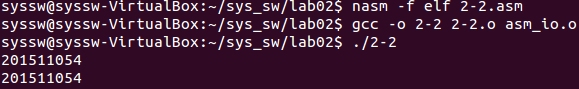
* 1. (2-1.asm) skeleton.asm 파일을 바탕으로 소스 파일 2-1.asm 을 생성하시오. 이 프로그램의. data 섹션에 out\_string 이라는 레이블을 붙여 임의의 문자열을 하나 정의하고, print\_string 루틴을 사용하여 이 문자열을 출력하는 프로그램을 작성하시오. 이 프로그램을 어셈블, 링크하여 실행 파일 2-1을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오.





* 1. (2-2.asm) skeleton.asm 파일을 이용하여, 다음과 같은 프로그램 2-2.asm 을 생성하시오. 이 프로그램은 먼저 read\_int 루틴을 사용하여 정수 하나를 입력 받은 후, 이를 print\_int 루틴을 사용해 출력해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 2-2를 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오.





1. (산술 연산) 간단한 숫자 계산을 하는 어셈블리 프로그램을 작성해 보자
   1. (3-1.asm) 두 개의 정수를 입력 받아, 두 수의 합을 계산하여 출력하는 어셈블리 프로그램 3-1.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다 (예에서 붉은색 글자는 사용자 입력에 해당함). 이 프로그램을 어셈블, 링크하여 실행 파일 3-1을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 덧셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴 사용만으로는 출력이 어려우므로, 이 범위를 넘어가지 않는 예를 사용하여 시험해 볼 것)..

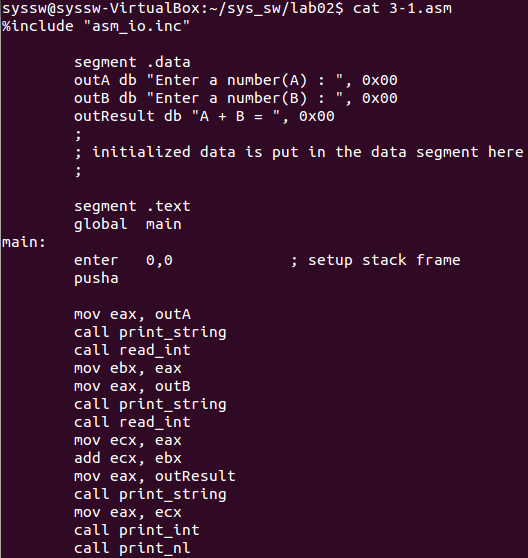
% ./3-1

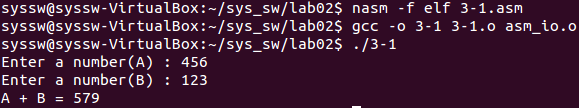
Enter a number (A) : 456

Enter a number (B) : 123

A + B = 579

%





* 1. (3-2.asm) 위의 3-1.asm 에 두 수의 차를 출력하는 기능을 추가한 어셈블리 프로그램 3-2.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 3-2를 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 뺄셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴 사용만으로는 출력이 어려우므로, 이 범위를 넘어가지 않는 예를 사용하여 시험해 볼 것)..

% ./3-2

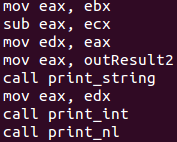
Enter a number (A) : 456

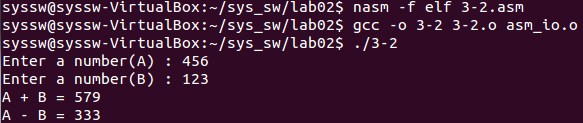
Enter a number (B) : 123

A + B = 579

A - B = 333

%





* 1. (3-3.asm) 위의 3-2.asm 에 두 수의 곱을 출력하는 기능을 추가한 어셈블리 프로그램 3-3.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 3-3을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 곱셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴 사용만으로는 출력이 어려우므로, 이 범위를 넘어가지 않는 예를 사용하여 시험해 볼 것).

% ./3-3

Enter a number (A) : 456

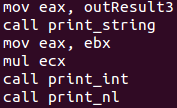
Enter a number (B) : 123

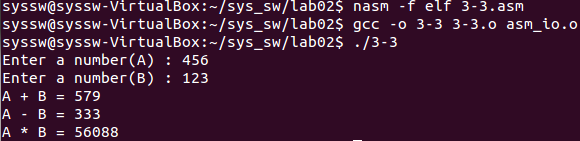
A + B = 579

A - B = 333

A \* B = 56088

%





* 1. (3-4.asm) 위의 3-3.asm 에 두 수를 나눈 몫과 나머지를 출력하는 기능을 추가한 어셈블리 프로그램 3-4.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 3-4 를 생성하고, 이를 실행시킨 결과를 화면 캡쳐하여 첨부하시오.

% ./2-4

Enter a number (A) : 456

Enter a number (B) : 123

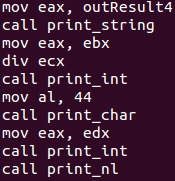
A + B = 579

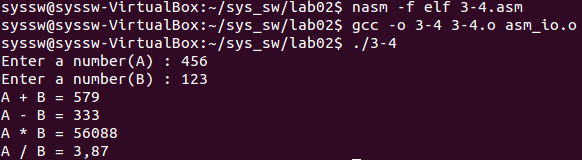
A - B = 333

A \* B = 56088

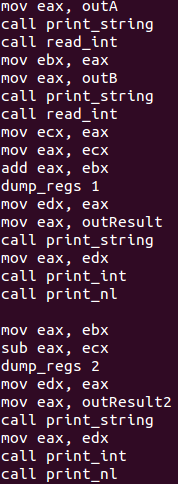
A / B = 3, 87

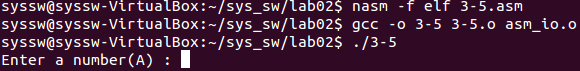
%



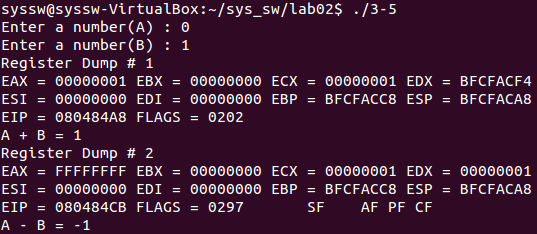


* 1. (3-5.asm) 위의 3-2.asm 프로그램 중, 각 add / sub 명령 수행 직후마다 dump\_regs 매크로를 사용하여 현재의 레지스터 값들을 출력하는 프로그램 3-5.asm 을 작성하고, 이 프로그램을 어셈블, 링크하여 실행 파일 3-5 를 생성하시오. 프로그램 3-5 를 다양한 입력값을 주어 덧셈과 뺄셈을 수행해 보고, 이 과정에서 CF 비트가 1 이 되는 수행에 대해 화면 캡쳐하여 첨부하고, 왜 CF 플래그가 1 이 되었는지 이유를 설명하시오. (OF 와 SF 가 1 이 되는 경우도 각각 첨부하고 이유를 설명하시오)



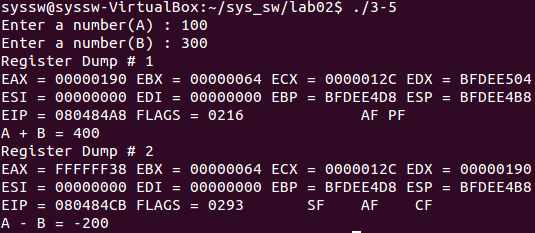


1. CF 가 1 이 되는 경우



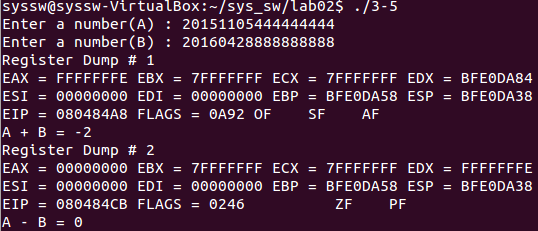
최상위 비트가 더 높은 자리로 Carry 되거나 최상위 비트로부터 Borrow이 발생한 경우

1. SF 가 1 이 되는 경우



최상위 비트가 1이 되는 경우, 즉 음수가 되었을 경우

1. OF 가 1 이 되는 경우



최대 정수보다 큰 수가 입력되거나, 최소 정수보다 작은 정수일 경우

끝.