상명대학교 컴퓨터과학과 "EA0003: 시스템소프트웨어" 실습

- 실습 번호: lab02

- 실습 디렉터리: ~/lab02 이름 이경헌

- 실습 날짜: 2017년 05월 1일

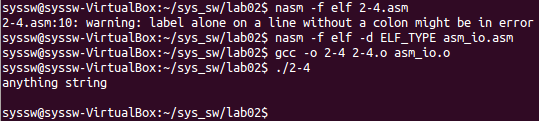
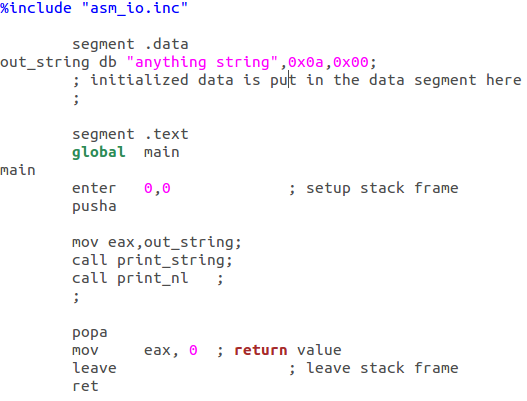
- 실습 제목: ASM IO 라이브러리 학번: 201311229

- 주의사항: 최종 실습 파일을 E-campus에 제출할 때 파일 이름은 반드시 lab-02-asm-io-학번-이름.dox 형태로 할 것.

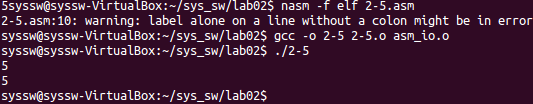
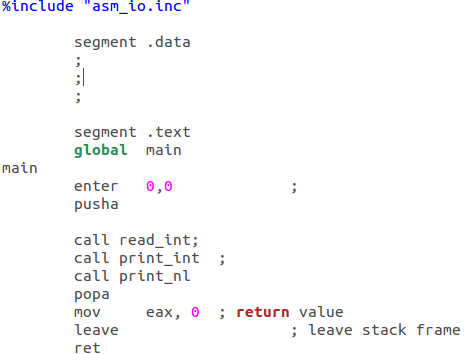
- 실습 내용:

1. (과제준비) 터미널에서 명령 “mkdir lab02”를 수행하여 자신의 홈 디렉터리 아래에 실습 디렉터리 (~/lab02)를 만든다. 이번 과제에서 만들어지는 모든 파일들은 이 디렉터리에 두도록 한다.

1. (ASM IO 라이브러리) ASM IO 라이브러리의 기능을 확인해보자
   1. 웹 브라우저를 실행하여 E-campus에서 라이브러리 파일 asm\_io.tar.gz 과 샘플 소스 파일 asm-io-sample-programs.zip을 다운로드 받으시오 (주: 인터넷에서 다운로드한 파일은 ~/Downloads 디렉터리에 저장됨)
   2. 명령 “cd lab02”를 수행하여 실습 디렉터리로 이동한 후, 명령 “gzip –d –c ~/Downloads/asm\_io.tar.gz | tar xvf –“ 를 실행하여 라이브러리 파일들을 생성하시오 (이 파일들은 이후 실습에서도 계속 필요하니 삭제하지 말 것).
   3. 명령 “unzip ~/Downloads/asm-io-sample-programs.zip“ 를 사용하여 샘플 소스 파일들을 생성하시오 (이 파일들은 이후 실습에서도 계속 필요하니 삭제하지 말 것.
   4. (2-4.asm) 명령 ”cp skeleton.asm 2-1.asm” 를 실행하여 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 2-4을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오.

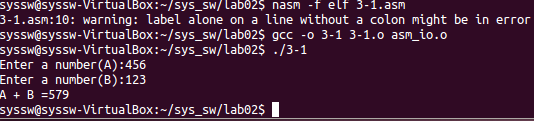
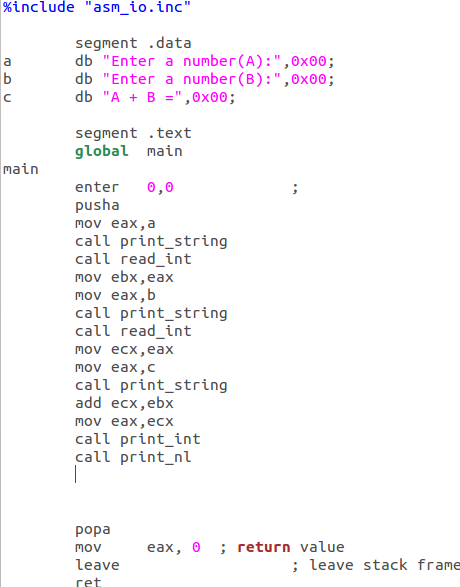


* 1. (2-5.asm) skeleton.asm 파일을 이용하여, 다음과 같은 프로그램 2-5.asm 을 생성하시오. 이 프로그램은 먼저 read\_int 루틴을 사용하여 정수 하나를 입력 받은 후, 이를 print\_int 루틴을 사용해 출력해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 2-5를 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오.



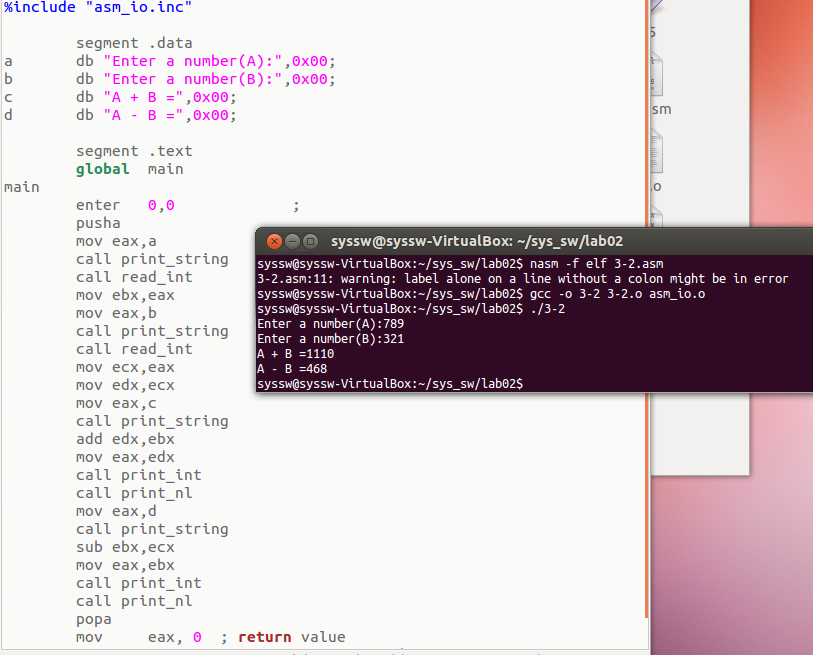
1. (산술 연산) 간단한 숫자 계산을 하는 어셈블리 프로그램을 작성해 보자
   1. (3-1.asm) 두 개의 정수를 입력 받아, 두 수의 합을 계산하여 출력하는 어셈블리 프로그램 3-1.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다 (예에서 붉은색 글자는 사용자 입력에 해당함). 이 프로그램을 어셈블, 링크하여 실행 파일 3-1을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 덧셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴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



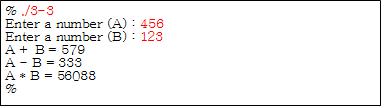


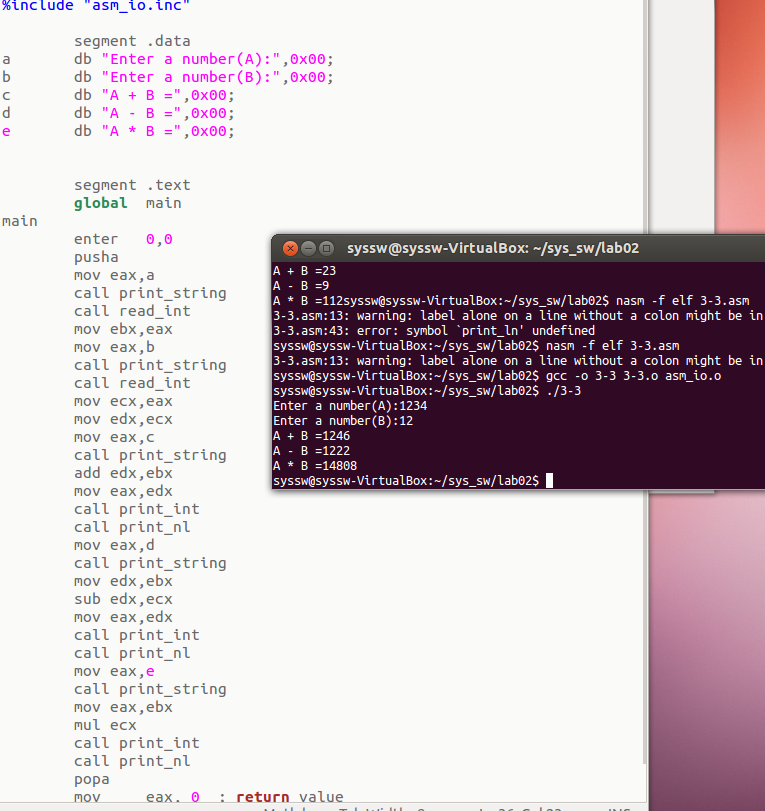
* 1. (3-2.asm) 위의 3-1.asm 에 두 수의 차를 출력하는 기능을 추가한 어셈블리 프로그램 3-2.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 3-2를 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 뺄셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴 사용만으로는 출력이 어려우므로, 이 범위를 넘어가지 않는 예를 사용하여 시험해 볼 것)..



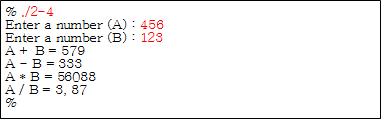


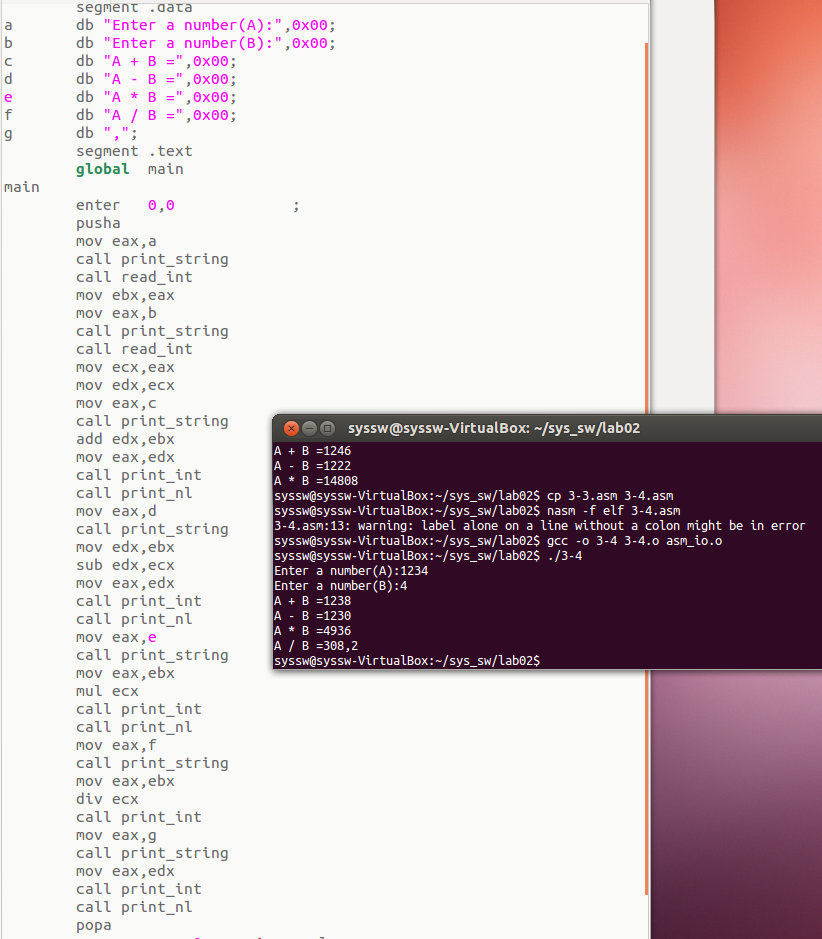
* 1. (3-3.asm) 위의 3-2.asm 에 두 수의 곱을 출력하는 기능을 추가한 어셈블리 프로그램 3-3.asm 을 작성하시오. 단, 이 프로그램은 “반드시” 아래와 같이 동작해야 한다. 이 프로그램을 어셈블, 링크하여 실행 파일 3-3을 생성한 후 이를 실행시키는 과정을 담은 화면을 캡쳐하여 아래에 첨부하시오 (주: 곱셈 결과가 32-bit 정수의 범위를 넘어가는 경우 단순히 print\_int 루틴 사용만으로는 출력이 어려우므로, 이 범위를 넘어가지 않는 예를 사용하여 시험해 볼 것).





* 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.

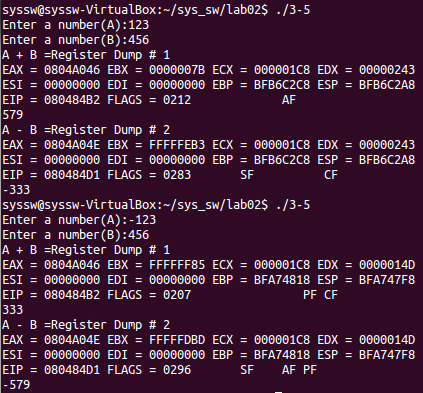




* 1. (3-5.asm) 위의 3-2.asm 프로그램 중, 각 add / sub 명령 수행 직후마다 dump\_regs 매크로를 사용하여 현재의 레지스터 값들을 출력하는 프로그램 3-5.asm 을 작성하고, 이 프로그램을 어셈블, 링크하여 실행 파일 3-5 를 생성하시오. 프로그램 3-5 를 다양한 입력값을 주어 덧셈과 뺄셈을 수행해 보고, 이 과정에서 CF 비트가 1 이 되는 수행에 대해 화면 캡쳐하여 첨부하고, 왜 CF 플래그가 1 이 되었는지 이유를 설명하시오. (OF 와 SF 가 1 이 되는 경우도 각각 첨부하고 이유를 설명하시오)

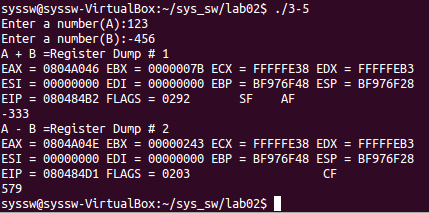
1. CF 가 1 이 되는 경우

연산에 의해서 operand 의 부호가 바뀌게 되면 CF가 1이 된다.

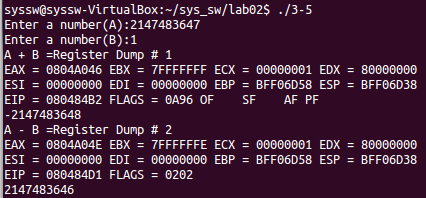


1. SF 가 1 이 되는 경우

연산 결과 값이 음수일 때 SF는 1이 된다



1. OF 가 1 이 되는 경우

결과값이 레지스터의 크기를 넘어갈 경우2의 31승이 넘어갈 경우 OF가 1이 된다. 

끝.