시스템 프로그래밍 프로젝트 3

1. 프로젝트 문제 및 목표

프로젝트 1, 2 에서 구현한 셸(shell)에 linking과 loading 기능을 추가하는 프로그램이다. 프로젝트 2 에서 구현된 assemble 명령을 통해서 생성된 object 파일을 link시켜 메모리에 올리는 일을 수행한다.

2. 요구사항

2.1 프로젝트 목표 설정

- 이미 제출한 프로젝트 2 에 아래의 기능들을 추가해야 한다.
- 구현해야 할 사항들
  1. 주소 지정 명령어 (progaddr)
  2. Linking Loader (loader)
  3. 프로그램 실행 (run)
  4. debug 명령어 (bp)

2.2 합성

프로젝트 2 에서 구현한 셸(shell)에 linking과 loading 기능을 추가하는 프로그램으로, 프로젝트 2 에서 구현된 assemble 명령을 통해서 생성된 object 파일을 link시켜 메모리에 올리는 일을 수행한다. 주소 지정 명령어, Linking Loader, 프로그램 실행 명령어, debug 명령어 등을 구현해야 하는데 이를 위해 필요한 자료구조 및 알고리즘을 구상하여 전체 프로그램을 설계한다.

2.3 제작 / 2.4 시험

1) 주소 지정 명령어

sicsim> **progaddr [address]**

- loader 또는 run 명령어를 수행할 때 시작하는 주소를 지정한다.
- sicsim이 시작되면 default로 progaddr는 0x00 주소로 지정한다.

2) Linking Loader

sicsim> **loader [object filename1] [object filename2] […]**

- filename1, filename2, … 에 해당하는 object 파일을 읽어서 linking 작업을 수행 후, 가상 메모리(1M)에 그 결과를 기록한다.
- 교재에 나오는 Pass1과 Pass2를 수행한다.
- 파일 개수는 3개까지만 고려한다.
- Loader 실행이 후, load map을 화면에 출력합니다. 강의자료 chapter3 pp.18-21 참조
- obj 파일에 에러가 존재할 경우 및 정상적으로 link, loader가 동작하지 않는 경우는 고려하지 않는다.
- 프로젝트 1에서 사용되는 명령어들이 사용되므로 프로젝트 1의 명령어가 제대로 구현이 안되어 있을 시 점수를 받을 수 없다.

ex) sicsim> loader proga.obj progb.obj progc.obj

proga.obj, progb.obj, progc.obj 를 가지고서 가상 메모리에 그 결과를 기록하고 load map을 화면에 출력한다.

Control section 및, symbol name의 출력 순서는 상관 없음)

```
sicsim>progaddr 4000
sicsim>loader proga.obj progb.obj progc.obj
control symbol address length
section name
---------------------------------
PROGA           4000   0063
        LISTA   4040
         ENDA   4054
PROGB           4063   007F
        LISTB   40C3
         ENDB   40D3
PROGC           40E2   0051
        LISTC   4112
         ENDC   4124
---------------------------------
          total length 0133
```

(가상 메모리 시작주소부터 끝주소까지 프로그램이 제대로 올라오지 않으면 0점 처리)

```
sicsim>dump 4000, 4133
04000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04020 03 20 1D 77 10 40 C7 05 00 14 00 00 00 00 00 00 ; . w.@.........
04030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04050 00 00 00 00 00 41 26 00 00 08 00 40 51 00 00 04 ; .....A&....@Q...
04060 00 00 83 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04090 00 00 00 00 00 00 00 00 03 10 40 40 77 20 27 ; ..........@@w '
040A0 05 10 00 14 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
040B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
040C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
040D0 00 00 00 00 41 26 00 00 08 00 40 51 00 00 04 00 ; ....A&....@Q....
040E0 00 83 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
040F0 00 00 00 00 00 00 00 00 00 00 03 10 40 40 77 10 ; ............@@w.
04100 40 C7 05 10 00 14 00 00 00 00 00 00 00 00 00 00 ; @.............
04110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
04120 00 00 00 00 00 41 26 00 00 08 00 40 51 00 00 04 ; .....A&....@Q...
04130 00 00 83 00                                     ; ................
```

ex) copy.obj 파일을 load 한 경우 (copy.obj 파일은 편의를 위해 M 레코드가 없으므로 항상 0 번지부터 올라간다고 가정한다.)

```
sicsim>progaddr 0
sicsim>loader copy.obj
control symbol address length
section name
---------------------------------
COPY              0000   1077
---------------------------------
          total length 1077
```

3) debug 명령어

sicsim> bp [address]

- sicsim에 breakpoint를 지정한다.
- breakpoint는 현재 프로그램의 Loc 기준으로 한다.
- Breakpoint는 bp 명령어를 통해서 프로그램 길이만큼 지정할 수 있다.
- run을 수행하면 breakpoint까지 프로그램이 실행되고 프로그램이 정지한다.
- 다음 run의 실행은 정지된 breakpoint부터 시작, 그 다음 breakpoint까지 진행된다.
- Breakpoint가 없으면 프로그램 끝까지 실행한다.
- 입력되는 bp의 범위는 메모리에 올라가는 프로그램 길이와 동일하다고 가정한다.

- 입력으로 들어오는 bp가 Loc값과 다른 경우는 고려하지 않는다. (ex. 강의자료 chapter3 pp.8 에서 bp가 2, 4, 5와 같은 경우는 고려하지 않는다)

```
sicsim>loader copy.obj
control symbol address length
section name
---------------------------------
COPY            0000   1077
---------------------------------
          total length 1077
sicsim>bp 3
           [ok] create breakpoint 3
sicsim>bp 2a
           [ok] create breakpoint 2a
sicsim>bp 1024
           [ok] create breakpoint 1024
```

sicsim> bp clear

- sicsim에 존재하는 breakpoint를 전부 삭제한다.

```
sicsim>bp clear
           [ok] clear all breakpoints
```

sicsim> bp

- sicsim에 존재하는 breakpoint를 전부 화면에 출력한다.

```
sicsim>bp
            breakpoint
            ----------
            3
            1046
            2A
```

4) 프로그램 실행

sicsim> run

- loader 명령어의 수행으로 메모리에 load된 프로그램을 실행한다.
- progaddr 명령어로 지정한 주소부터 실행된다.
- 실행 결과로써 register 상태를 화면에 출력한다. 출력되는 register는 A, X, L, PC, B, S, T 이다.
- 모든 레지스터는 프로그램이 load되면 0으로 초기화 된다. 단, PC는 프로그램의 시작 주소, L은 프로그램의 길이로 초기화 한다.
- Breakpoint까지 실행되고 Breakpoint가 없으면 프로그램 끝까지 실행한다.

시스템 프로그래밍 프로젝트 #3

- 수행하는 프로그램은 제공되는 copy.obj를 사용하며 다른 경우는 고려하지 않는다. copy.obj는 0번지부터 프로그램에 올라간다고 가정한다.
- copy.obj파일은 강의자료 chapter3 pp.8의 코드로부터 만든 오브젝트파일이다.
- run 을 했을 시, 멈추는 break point 지점이 프로그램 흐름과 같아야 한다.
- 메모리에 프로그램이 Load 되어 있지 않은 상황에서의 실행(run)은 고려하지 않아도 된다. (segmentation fault 에 대한 예외처리를 하지 않아도 된다.)

- copy.obj 역 어셈블시 명령어가 I/O 관련 명령어인 경우는 다음과 같이 처리한다.

TD: 다음 instruction으로 넘어가되 CC(condition code)는 '<'로 변경되었다고 가정한다.

RD: input device로부터 아무것도 받지 못했다고 가정한다. 즉, 다음 명령어인 COMPR A, S의 결과 CC가 '='이라 가정한다.

WD: 다음 instruction으로 넘어간다.

ex) 아래 예시는 copy.obj 파일을 0번지부터 올린 후 bp가 3, 2A, 1046일때의 run을 4번 수행시킨 결과

```
A : 000000  X : 000000
L : 001077 PC : 000003
B : 000000  S : 000000
T : 000000
            Stop at checkpoint[3]
A : 000000  X : 000000
L : 00000A PC : 001046
B : 000033  S : 000000
T : 001000
            Stop at checkpoint[1046]
A : 000046  X : 000003
L : 00002A PC : 00002A
B : 000033  S : 000000
T : 000003
            Stop at checkpoint[2A]
```

```
A : 000046  X : 000003
L : 00002A PC : 001077
B : 000033  S : 000000
T : 000003
          End Program
```

ex) breakpoint를 설정하고 프로그램을 load 한 뒤 run을 했을 때의 결과에 대한 예시

3. 기타

3.1 환경 구성

Linux (gcc) : 반드시 gcc를 이용해서 C언어로 프로그램 하십시오.

특히 C언어가 아닌 C++ 등 다른 언어를 사용하거나, 도스 및 윈도우에서 작성한 경우 0점 처리합니다.

참고) 컴파일 시, make 파일에 gcc -Wall 옵션을 사용하여 warning 을 철저히 확인하시기 바랍니다. (Warning 발생시 건당 1점 감점 처리함. 최대 5점 감점)

3.2 팀 구성

개별 프로젝트입니다.

3.3 수행 기간: 4월 12일(월) 00:00 ~ 4월 30일(금) 23:59

Late는 없습니다. 기한 내에 제출하셔야 합니다.

3.4 제출물 (5가지 파일중 하나라도 없는 경우에는 0점 처리함, 꼼꼼하게 확인하고 보내주세요)

1) 프로그램 소스 및 헤더파일
2) Makefile
3) 프로그램 다큐멘테이션 리포트: 소스 및 프로그램의 구현방법을 설명한 Document. 반드시 예제 파일에 준해서 작성할 것
4) 프로그램의 컴파일 방법 및 실행방법에 대한 간단한 내용을 적은 README파일
5) 테스트 파일 (ex) proga.obj, progb.obj, progc.obj, copy.obj)

그 외 필요 파일이 포함되지 않아 프로그램이 동작하지 않는 경우 0점

3.5 제출 방법 (형식을 지키지 않을 경우 감점 처리함)

시스템 프로그래밍 프로젝트 #3

sp<본인 학번>_proj3 이름의 디렉터리를 만들고, 여기에 위에서 설명한 모든 파일들을 넣은 후, **디렉터리를 tar로 압축하여 한 파일로 만들어** 사이버캠퍼스 과제란에 압축한 파일을 제출하시기 바랍니다. (압축파일 내에 반드시 디렉터리가 포함되어 있어야 하며, 바이너리파일 및 코어파일을 제외할 것.)

제출주소: 사이버캠퍼스 과제란

파일형식: sp학번_proj3.tar (ex. sp20211234_proj3.tar)

ex) sp20211234_proj3/

README → 컴파일 방법 및 실행방법에 대한 간단한 내용을 적은 파일

Document.doc →(또는 Document.docx)

opcode.txt → 프로젝트#2에서 제공된 opcode 파일.

20211234.c → 소스 파일이 여러 개인 경우 main 함수가 있는 파일의 이름을 학번.c 로 합니다.

20211234.h → 최소 한 개 이상의 헤더 파일. 하나인 경우 학번.h.

Makefile → 실행파일은 20191234.out처럼 학번.out 이름으로 고정할 것.

proga.obj, progb.obj, progc.obj → linking loader에 대한 example.

copy.obj → run 명령어에 대한 example.

tar 명령어는 아래와 같이 사용합니다.

tar 파일로 묶을 때 지난 project와 동일하게 -z 옵션을 사용하지 않습니다.

ex) tar -cvf **sp<학번>_proj3.tar** 만든 디렉토리명

tar 파일의 이름은 다음과 같이 되어야 합니다.

sp<학번>_proj3.tar

ex) sp20211234_proj3.tar

makefile에 반드시 포함되어야 할 script : make, make clean

주의사항

+ 제출형식(tar file 이름 형식, 내용물 등)이 잘못되었을 시, 감점 10%

+ 종종 윈도우 알집으로 tar로 압축하시는 경우가 있습니다. 해당 경우도 10% 감점합니다.

3.6 Source code 관련

Segmentation fault

실행 불가 시 : 0점

명령 수행 시 : 그 부분점수 0점

Compile error

0점

Warning

1점 감점

Average case

기본 예제 파일 수행

주석

주석이 없거나, 알아볼 수 없는 경우 감점 시키겠습니다.

타인이 알아볼 수 있는 형태로 주석을 달아주십시오.

또한 주석이 깨져 있는 경우에 대해서도 감점.

Copy

0점 처리!!

3.7 프로젝트에 대한 질문 사항은 eclass 질문게시판을 이용해 주세요. 중복 질문이 덜 하도록, 제목에 질문의 요지를 포함해 주세요.

3.8 다른 사람의 프로그램을 일 부분이라도 copy해서 제출한 사람은 이 과목의 기말 성적이 F가 나갑니다.

*** 본 프로젝트는 시간이 많이 소요됩니다. 반드시 일찍 시작해서 프로젝트 수행 시 나타나는 질문을 미리 해결해야 프로젝트를 잘 마치실 수 있습니다. 한주 전에 프로젝트 마감을 목표로 진행하는 것이 중요합니다!

* 프로젝트에 대한 질문사항은 사이버캠퍼스 질의응답 게시판을 이용해 주시거나 조교에게 연락해 주시기 바랍니다.

박주훈 : sowild@sogang.ac.kr