

시스템 프로그래밍 개별 프로젝트 #2

1. 프로젝트 문제 및 목표

프로젝트 #1에서 구현한 셸(shell)에 assemble 기능을 추가하는 프로그램입니다. SIC/XE의 assembly program source 파일을 입력 받아서 object파일을 생성하고, 어셈블리 과정 중 생성된 symbol table과 결과물인 object 파일을 볼 수 있는 기능을 제공해야 합니다. 교재의 2.2까지 설명된 SIC/XE 어셈블러의 기능을 구현함을 원칙으로 한다.

2. 요구사항

2.1 프로젝트 목표 설정

- 이미 제출한 프로젝트#1에 아래의 기능들을 추가해야 합니다.
- 구현해야 할 사항들 (다음 페이지에 보다 자세한 설명이 나옵니다.)
 - ① Shell 관련 명령어들 (help, type)
 - ② assembler (assemble)
 - ③ assemble 관련 명령어 (symbol)

2.2 합성

프로젝트 #1에서 구현한 셸(shell)에 assemble 기능을 추가하는 프로그램을 작성하는 프로젝트로, SIC/XE machine의 assembly program source 파일을 입력 받아서 object파일을 생성하고, 어셈블리 과정 중 생성된 symbol table과 결과물인 object 파일을 볼 수 있는 기능을 제공해야 한다. 이와 같은 기능을 제공하는 프로그램을 작성하기 위해 필요한 자료구조와 알고리즘을 구상하여 전체적인 프로그램을 설계한다.

2.3 제작 / 2.4 시험 / 2.5 평가

1) Shell 관련 명령어

① sicsim> help

- 아래와 같이 Shell에서 실행 가능한 모든 명령어들의 리스트를 화면에 출력해준다.

h[elp]

d[ir]

q[uit]

시스템 프로그래밍 프로젝트 #2

```

hi[story]
du[mp] [start, end]
e[dit] address, value
f[ill] start, end, value
reset
opcode mnemonic
opcodelist
{
    assemble filename
    type filename
    symbol
}

```

② sicsim> type filename

- filename에 해당하는 파일을 현재 디렉터리에서 읽어서 화면에 출력한다.
- 현재 디렉터리에 해당 파일이 존재하지 않으면 에러 메시지를 출력한다.
- 시스템 콜을 사용하지 않는다.

걍 fopen 하면 될듯.

ex) sicsim> type a.obj

```

HCOPY 0010000107A
T00101E150C10364820610810334C0000454F46000003000000
E001000

```

2) SIC/XE 어셈블러 명령

① sicsim> assemble filename

- filename에 해당하는 소스 파일을 읽어서 object파일과 리스트 파일을 만든다.
- 소스 파일의 확장자는 .asm입니다.
- 리스트 파일의 파일명은 소스 파일과 동일하고 확장자는 .lst입니다.
- object 파일의 파일명은 소스 파일과 동일하고 확장자는 .obj입니다.
- 소스파일에 에러가 존재할 경우, 리스트 파일과 object파일을 생성하지 않고 에러 내용을 화면에 출력합니다. 에러 발생시 바로 명령이 종료됩니다.

에러의 내용은 디버깅을 위해서 어떤 라인에서 에러가 발생했는지 출력한다.

ex) sicsim> assemble 2_5.asm

i. 파일명을 2_5.lst로 하는 리스트 파일과 2_5.obj로 하는 object파일이 만들어집니다.

ex) sicsim> type 2_5.asm (책 2.2의 figure 2.5참고(3rd edition 기준))

```

5      COPY     START     1000     COPY FILE...
10     FIRST    STL      RETADR   SAVE RETURN ADDRESS

```

시스템 프로그래밍 프로젝트 #2

```
.....  
255          END      FIRST  
sicsim> assemble 2_5.asm  
          output file : [2_5.lst], [2_5.obj]  
sicsim> type 2_5.lst(책 2.2의 figure 2.6참고(3rd edition 기준))  
      5    0000   COPY   START    0  
     10   0000   FIRST   STL     RETADR    17202D  
     12   0003           LDB     #LENGTH   69202D  
.....  
255          END      FIRST  
sicsim> type 2_5.obj(책 2.2의 figure 2.8참고(3rd edition 기준))  
HCOPY 000000001077  
T0000001D17202D69202D.....  
T00001D.....  
....  
E000000
```

② sicsim> symbol

- assemble 과정 중에 생성된 symbol table을 화면에 출력합니다. Symbol table은 각자 설계를 하고, 출력은 아래와 같이 합니다
(출력형식을 꼭 지킬 것)
- 가장 최근에 assemble 한 파일의 symbol table을 출력합니다.
- symbol의 출력은 symbol을 기준으로 내림차순으로 정렬이 되어야 합니다.

```
sicsim> assemble 2_5.asm  
          output file : [2_5.lst], [2_5.obj]  
sicsim> symbol  
(Wt)RETADR(Wt)0030  
=>실제 출력 될시에는  
RETADR 0030  
-즉 하나의 symbol당 한 line을 차지하고, 탭+Symbol+탭+주소값+Wn  
을 의미합니다. 맨마지막 줄에는 Wn(엔터)를 빼주세요.
```

교재의 2.2까지 설명된 SIC/XE 어셈블러의 기능을 구현함을 원칙으로 한다.

원래의 SIC/XE machine은 standard machine에 하위 호환 되어야 하지만 이번 SIC/XE 어셈블러에서는 체크하지 않습니다.

* Compile 해야 되는 기본 소스파일 p55 Figure 2.5

시스템 프로그래밍 프로젝트 #2

3. 환경: 개별 프로젝트입니다.

Linux (gcc) : 반드시 gcc만을 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 확인 하시기 바랍니다. (Warning 발생시 감점 처리함.)

4. Due Date :

4월 8일(월) 11:59시까지 제출.

5. 제출물 (아래 파일들이 모두 포함되어 있어야 함)

- 1) 프로그램 소스 및 헤더파일
- 2) Makefile
- 3) 프로그램 다큐멘테이션 리포트:
이번에는 XE 소스 assemble이 주된 기능인만큼 이에 대한 프로그램 흐름이나 알고리즘 설명(어떻게 구현하였는지)을 꼭 넣어주시기 바랍니다.
- 4) 프로그램의 컴파일 방법 및 실행방법에 대한 간단한 내용을 적은 README파일(.txt)
- 5) 기타 수행에 필요한 파일 (ex) opcode.txt ...)
- 6) 테스트 파일 (ex) 2-5.asm 등

6. 제출 방법

sp<학번>_proj2 이름의 디렉터리를 만들고, 여기에 위에서 설명한 모든 파일들을 넣은 후, 디렉터리를 tar로 압축하여 한 파일로 만들어 사이버캠퍼스에 제출하시기 바랍니다.
(압축파일 내에 반드시 디렉터리가 포함되어 있어야 하며, 바이너리파일 및 코어파일을 제외할 것. 기타 불필요한 파일을 포함시키지 말 것.)

ex) sp20191234_proj2/

README → 컴파일 방법 및 실행방법에 대한 간단한 내용을 적은 파일
Document.doc →(또는 Document.hwp)

20191234.c → 소스 파일이 여러 개인 경우 main 함수가 있는 파일의
이름을 학번.c 로 합니다.

20191234.h → 최소 한 개 이상의 헤더 파일. 하나인 경우 학번.h
Makefile → 실행파일은 20191234.out처럼 학번.out 이름으로 고정할 것.

opcode.txt → 프로젝트#1에서 제공된 opcode 파일.
2_5.asm → 제공되는 테스트파일.

시스템 프로그래밍 프로젝트 #2

tar 명령어는 아래와 같이 사용합니다.

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

ex) tar cvf sp<학번>_proj2.tar 만든 디렉토리명

주의사항

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

+ **제출 마감 시간을 지키지 못했을 경우 0점**

7. Source code 관련

Compile error

Compile error로 실행이 불가능한 경우 : 숙제 전체 0점.

Segmentation fault

실행 불가 시 : 0점

명령 수행 시 : 그 부분점수 0점

Warning

1건당 1점 감점

Average case

기본 예제 파일 수행

주석

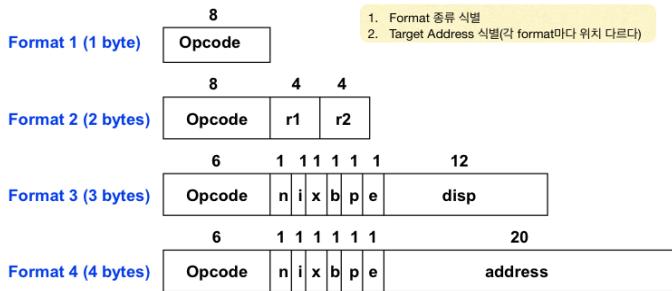
주석이 없거나, 알아볼 수 없는 경우 감점 시키겠습니다.

타인이 알아볼 수 있는 형태로 주석을 달아주십시오.

*** 모든 프로그램은 자동 검증 프로그램에 의해서 검증이 됩니다. 절대로 타인의 프로그램을 참조하지 말도록 하세요. 무조건 F가 나갑니다. ****

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

***** 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!



S Y M L D A N A M E , X \O

S Y M L D A N A M E , X V

S Y M L D A N A M E , X \O

SYM LDA NAME , X \o

token	0	SYM
	1	LDA
	2	NAME
	3	

0 1 2 3 4 5 6 7 8 9 10
0000□□,0000|0

$$\begin{array}{l} i=6 \\ C_{14}=2 \end{array} \quad \text{System} = 11$$

0 1 2 3 4 5 6 7 8 9 0
0 0 0 0, 0 0 0 0 0 0 \ 0

$$\begin{aligned} i &= 4 \\ \text{Count} &= 2 \\ \text{Sum} &= 11 \end{aligned}$$

TOKEN ~ 5>1

i) FIRST FIX

BASE LENGTH

EOF BYTE C'EOF'

ii) YEAH CLEAR X

INPUT BYTE X'4B'

WHOW COMPA A,B

VARI RESW 4

ii) LDCH BUFFER,X

0	1	2	3	4	...
A	B	C	D	E	...

22	23	24	25
W	X	Y	Z

HASH TABLE C1 size는 28개이다.

↑ label의 시작 alphabet의 index + 1개의 head에 붙음

0	A 0 7 14 21	13 N 6 13 20 27
1	B 1 8 15 22	14 D 0 7 14 21
2	C 2 9 16 23	15 P 1 8 15 22
3	D 3 10 17 24	16 Q 2 9 16 23
4	E 4 11 18 25	17 R 3 10 17 24
5	F 5 12 19 26	18 S 4 11 18 25
6	G 6 13 20 27	19 T 5 12 19 26
7	H 0 7 14 21	20 U 6 13 20 27
8	I 1 8 15 22	21 V 0 7 14 21
9	J 2 9 16 23	22 W 1 8 15 22
10	K 3 10 17 24	23 X 2 9 16 23
11	L 4 11 18 25	24 Y 3 10 17 24
12	M 5 12 19 26	25 Z 4 11 18 25

Alphabet num을 7로 나눈 나머지 = index % 7

0	4	8	12	16	20	24
0	7	14	21			

for 0 ~ 6 f

4, 11, 18, 25 처리! E, L, S, Z
 3 10 17 24 D K R Y
 2 9 16 23 C J Q X

0	A H O V
1	B I P W
2	C J Q X
3	D K R Y
4	→ E L S Z
5	F M T
6	G N U

내점자순으로 저장!

1. 2. 3. 4

$$4C_2 = 6$$

$$\left\{ \begin{array}{l} (2,1) \times (3,1) \times (4,1) = 1 \\ (1,2) \times (3,2) \times (4,2) = 1 \\ (1,3) (2,3) (4,3) = 1 \\ (1,4) (2,4) (3,4) = 1 \end{array} \right. \quad \begin{array}{l} \textcircled{1} \\ \textcircled{2} \\ \textcircled{3} \\ \textcircled{4} \end{array}$$

for Z do A

4*11 { while(pMove && pMove == 'Z')

0	0	A H O V
1	1	B Z P W
2	2	C J Q X
3	3	D K R Y
4	4	E L S Z
5	5	F M T
6	6	G N U

0	A	0	7	14	21	13	N	6	13	20	27
1	B	1	8	15	22	14	O	0	7	14	21
2	C	2	9	16	23	15	P	1	8	15	22
3	D	3	10	17	24	16	Q	2	9	16	23
4	E	4	11	18	25	17	R	3	10	17	24
5	F	5	12	19	26	18	S	4	11	18	25
6	G	6	13	20	27	19	T	5	12	19	26
7	H	7	14	21	28	20	U	6	13	20	27
8	I	8	15	22	29	21	V	0	7	14	21
9	J	9	16	23	30	22	W	1	8	15	22
10	K	10	17	24	31	23	X	2	9	16	23
11	L	11	18	25	32	24	Y	3	10	17	24
12	M	12	19	26	33	25	Z	4	11	18	25

p1
p2
p3
p4 }
While (p1 & p2 & p3 & p4) {
print biggest of four;
switch (biggest) {

case 1 :

```
print p1 ;
p1 = p1->link;
if (!p1) break;
if ((p1->s)[0] != 'alpha') break;
```

While (p1 & p2 & p3) {
print biggest of four;

case 2 :

```
if (!p1) p1 = p3 ;
if (!p2) p2 = p3 ;
while (p1 & p2) {
    print bigger one;
```

```
if (p1) print p1 ;
if (p2) print p2 ;
```

Sign Cnt == 1 { 000x 8
 00x0 4
 0x00 2
 x000 1

Cnt == 2 { 00xx 12
 0x0x 10
 0xx0 6
 x00x 9
 x0x0 5
 xx00 3

Cnt == 3 XXX0 7
 XX0X 11
 X0XX 13
 0XXX 14

In TEXT RECORD,
calculate obj code.
using memory.

A. X. L. PC. SW. B. S.T.F
↓
LOCCTR ↓
 BASE.

PC. = LOCCTR

} LIST File of text record 출력은 QUEUE 를 짜자!
크기 count 하며 enqueue 진행.
if } s RES-Directive
 count ≥ MAX-LEN ⇒ 출력!(dequeue).

해야 할 것들

{ tokenize or/and comma issue 해결
BYTE 상수 출력하는 법 찾고
각 case의 obj code 맵

queue를 이용해 obj file of text record 출력.

Description

↳ pass1 은 지나친 numNode on lineNum
LDC
s,e,skip flag 정보 차장 (line by line)
+ Create Symtab.

PASS2에서 numNode의 정보를 활용해 PASS2 효율적으로 진행

PASS2에서는 각 instruction의 Object Code 생성

.lze file format

lineNum /t /t /t LOC/t/t label /t dir/op /t Operand /t/t/t objCode

numNode

token

.obj file format

*Record { obj code 를 이용한 text record 작성.
size
LOC

getObjCode

i) label operation operand

ii) operation

00AB49

$\Rightarrow 9 + 16 \cdot 4 + 16^2 \cdot 11 + 16^3 \cdot 10$

594821

* B Reg을 초기화시키지 않았을 때 Warning

1. no symbol defined

* lst 출력 format

2. + sign 있는 4자리

* + sign 있는 4자리

3. no opcode defined

+ sign 있는 3자리

cf) indirect addressing C3 B 를 저장하는 법?

Okay?.

