**과목명: 시스템프로그래밍**

**1분반**

**<<Project #3>>**

**서강대학교 [컴퓨터공학과]**

**[20181617]**

**[김채연]**

목 차

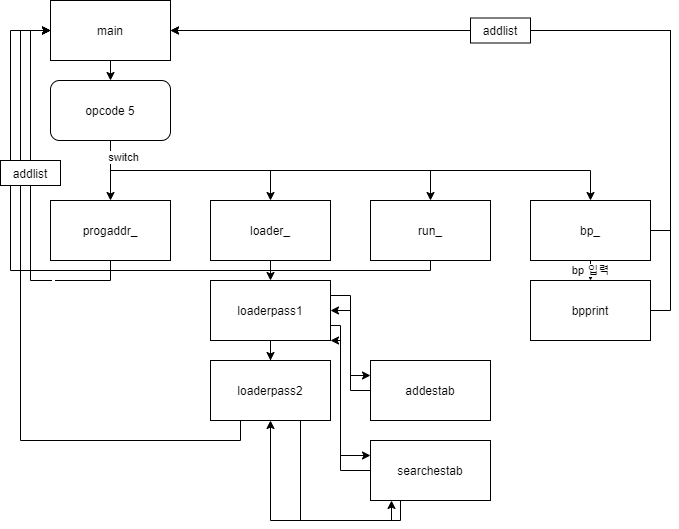
1. **프로그램 개요**
2. **프로그램 설명**
   1. 프로그램 흐름도
   2. 프로그램 기능
3. **모듈 정의**
4. **전역 변수 정의**
5. **코드 설명**

**1. 프로그램 개요**

프로젝트 1,2 에서 구현한 셀에 linking과 loading 기능을 추가하는 프로그램이다. 프로젝트2에서 구현된 assemble 명령을 통해서 생성된 object 파일을 link 시켜 메모리에 올리는 일을 수행한다.

**2. 프로그램 설명**

2.1 프로그램 흐름도



2.2 프로그램 기능

1) 주소 지정 명령어

1. progaddr [address]

loader 또는 run 명령어를 수행할 때 시작하는 주소를 지정한다.

2) Ling Loader

1. loader [object filename1][object filename2][…]

filename1, filename2, …에 해당하는 object 파일을 읽어서 linking 작업을 수행 후 가상 메모리에 그 결과를 기록한다.

3) debug 명령어

1. bp [address or clear]

-bp address

sicsim에 breakpoint를 지정한다.

-bp clear

sicsim에 존재하는 breakpoint를 전부 삭제한다.

-bp

sicsim에 존재하는 breakpoint를 전부 화면에 출력한다.

4) 프로그램 실행

1. run

loader 명령어의 수행으로 메모리에 load된 프로그램을 실행한다.

**3. 모듈 정의**

1) stdio.h

표준 입출력 함수를 담고있는 헤더파일

2) string.h

C형식 문자열 관련 함수를 담고있는 헤더 파일

3) stdlib.h

난수생성, 문자열 형식을 다른 형식으로 변환, 의사 난수생성, 동적 메모리 관리 등의 함수를 담고있는 헤더 파일

4) sys/stat.h

파일의 상태를 확인하기 위한 자료형, 구조체, 상수와 관련된 함수를 담고있는 헤더 파일

5) dirent.h

파일 시스템의 디렉터리를 나타내기 위한 구조체를 정의하는 헤더 파일

6) enum

열거형이라고 부르며 정수형 상수에 이름을 붙여서 코드를 작성하고 이해하는 것을 쉽게 도와준다.

7) typedef

어떠한 자료형 이름에 새 이름을 부여하여 정의하는 것이다.

8) struct

하나 이상의 변수를 묶어서 좀 더 편리하게 사용할 수 있도록 도와준다.

3.1 함수 설명

1) void option5(char \*input) : progaddr, run, loader, bp 명령어를 처리하는 함수

char array[4][50] : 입력된 input 명령어를 ‘ ’으로 나눠 저장하는 변수

int len : input의 길이를 저장하는 변수

int I, j, k : for문을 돌릴 때 필요한 변수

int result : 함수들의 결과를 받는 변수

2) int progaddr\_(char \* address) : progaddr 명령어를 처리하는 함수

3) int loader\_(char\*file1 , char\* file2,char\* file3) : loader 명령어를 처리하는 함수

int I, k : for문을 돌릴 때 필요한 변수

char file[3][50] : 3개의 file 이름을 다시 저장하는 변수

int result : 함수들의 결과를 받는 변수

4) int run\_() : run 명령어를 처리하는 함수

int start : 프로그램의 시작주소

int end : 프로그램의 끝

int m : opcode의 address값

int row : memory의 row

int col : memory의 col

int value : memory[row][col]의 값

int code : value를 4로 나누어 명령어들의 opcode와 비교하는 변수

int indirect : indirect addressing일 때의 주소 값

int I : for문을 돌릴 때 필요한 변수

5) int bp\_(char\* input) : bp 명령어를 처리하는 함수

int num : 16진수를 10진수로 바꿔 저장하는 변수

int count : bp의 개수를 저장하는 변수

6) void bpprint() : 저장된 bp를 출력하는 함수

int count : bp의 개수를 저장하는 변수

int I : for문을 돌릴 때 필요한 변수

7) int loaderpass1(char\* filename, int filenum) : loader 과정중 pass1을 처리하는 함수

FILE \* fp : 파일 포인터

char array[100] : 파일을 읽을 때 필요한 string

char ctemp[8] : array를 6개씩 따로 저장하는 변수

int pstart : 프로그램의 시작주소

int pend : 프로그램의 끝

int result : 함수의 결과 값을 받는 변수

int linecount : 줄 수를 세는 변수

int I,j : for문을 돌릴 때 필요한 변수

char name[10] : 프로그램,symbol의 이름을 저장하는 변수

8) int loaderpass2(char\* filename, int filenum) : loader 과정 중 pass2를 처리하는 함수

FILE \* fp : 파일 포인터

char array[100] : 파일을 읽을 때 필요한 string

int I : for문을 돌릴 때 필요한 변수

char ctemp[8] : array를 6개 씩 따로 저장하는 변수

int num : opcode에서 숫자를 읽어 int로 바꿔 저장하는 변수

int result : 함수의 결과 값을 받는 변수

int value : opcode에서 숫자를 읽어 int로 바꿔 저장하는 변수

char r[10][6] : reference 이름을 저장하는 변수

int linecount : 줄 수를 세는 변수

int j : for문을 돌릴 때 필요한 변수

char ccnum[3] : array를 2개 씩 따로 저장하는 변수

char \*toktemp : strtok을 할 때 쓰는 변수

int num2 : opcode에서 숫자를 읽어 int로 바꿔 저장하는 변수

int row : memory row

int col : memory col

int irow : memory row 이전 것 저장

int icol : memory col 이전 것 저장

int a : 16진수의 두번째 자리 저장

int b : 16진수의 첫번째 자리 저장

9) void addestab(char\* input, int address, int filenum) : estab에 symbol을 추가하는 함수

ESTABNODE\* new

char \*result : strtok을 할 때 쓰는 변수

10) int searchestab(char\* input) : estab에서 symbol을 찾는 함수

ESTABNODE\* temp : estab에서 한 개의 노드를 표시하는 변수

**4. 전역 변수 정의**

1) ESTABNODE

estab의 해쉬테이블을 구성하는 NODE, typedef를 이용하여 정의함. name은 symbol의 이름, address는 주소, next는 다음 노드를 가리킨다.

2) ESTABNODE\* estab[3]

estab의 해쉬테이블 head. 3개의 파일이므로 3개 정의함.

3) int prostart

전체 program의 시작주소를 저장하는 변수이다.

4) int csaddr[3]

각 file의 program의 시작주소를 저장하는 변수이다.

4) int cslth[3]

각 file의 program의 길이를 저장하는 변수이다.

5) int A~Treg

각 레지스터 값을 저장하는 변수이다.

6) char conditioncode

CC의 값을 저장하는 변수이다.

7) int bparray[100]

bp를 저장하는 변수이다.

8) int totallength

전체 program의 길이를 저장하는 변수이다.

**5. 코드 설명**

1) void option5(char \*input)

프로젝트3의 구현 명령어인 progaddr, loader, run, bp를 구분하는 함수이다. input으로 사용자가 입력한 값을 넘겨받는다. 그다음 ‘ ’로 input을 구분하여 array에 따로 저장한다. array[0]과 각 명령어들과 비교하여 switch문을 통하여 명령어에 맞는 함수를 불러준다. 명령어의 결과가 올바르면 addlist 해준다. 만약 이중에 일치하는 명령어가 없으면 다음 option함수를 불러 input을 다시 넘겨준다.

2) int progaddr\_(char \* address)

progaddr 명령어를 처리하는 함수이다. 넘겨받은 address string을 strtol을 통해 prostart 전역변수에 int형으로 다시 저장한다. 만약 prostart가 주어진 범위가 아니라면 다시 0으로 설정되고 에러메세지를 출력하고 -1을 리턴한다. 범위내라면 0을 리턴한다.

3) int loader\_(char\*file1 , char\* file2,char\* file3)

loader 명령어를 처리하는 함수이다. 만약 file이 입력되지 않는다면 에러메세지를 출력하고 -1을 리턴한다. 파일이 제대로 입력되었다면 csaddr, cslth estab, register 값들을 모두 초기화한다. 들어온 파일 개수만큼 loaderpass1함수를 실행하고 loaderpass1의 결과가 -1이면 -1을 리턴한다. 결과가 -1이 아니라면 loaderpass2함수를 뒤이어 실행하고 마찬가지로 결과가 -1이면 -1을 리턴한다. loader가 성공적으로 실행되었다면 symbol name과 address, length들을 출력해준다. 이때 PCreg에는 프로그램의 시작주소가, totallength와 Lreg에는 프로그램의 총 길이가 저장된다.

4) int run\_()

run 명령어를 처리하는 함수이다. 만약 아무런 프로그램이 loader되지 않았다면 에러 메시지를 출력하고 -1을 리턴한다. 제대로 입력되었다면 PCreg가 프로그램의 총길이가 되기전까지 while문을 이용하여 반복하여 메모리 값을 읽어온다. 먼저 메모리를 하나 읽어 4로 나누어 mnemonic의 opcode를 4로 나눈 값과 비교하여 switch문을 통하여 구분한다. 각 mnemonic은 case로 나누어져 있다. 명령어를 읽어온 메모리의 4로 나눈 나머지값을 통해 addressing을 알 수 있고, 그 다음 메모리의 xbpe 값을 통해 형식을 구분하고 b,p를 구분하였다. 그리고 그다음 메모리를 통해 address값을 읽어왔다. 이를 이용하여 각 명령어에 맞게 처리해주었다. 그리고 도중에 PCreg가 bp의 값중에 일치하는 것이 있으면 run을 종료하고 각 레지스터의 값을 출력해주었다.

5) int bp\_(char\* input)

bp명령어를 처리하는 함수이다.bp도 3가지로 구분하여 bp만 입력되었을 때는 저장된 모든 bp를 출력하는 함수를 호출하였다. bp clear가 입력되었을 때는 모든 bp를 없애도록 bparray[0]에 저장된 bp의 개수를 0으로 초기화했다. bp address가 입력되었을 때는 bparray[0]을 1더해준뒤 그 순서번째의 bparray에 address값을 저장했다.

6) void bpprint()

bp를 모두 출력해주는 함수이다. bparray[0]에 저장되어있는 bp의 개수만큼 for문을 이용하여 16진수 형태로 출력한다.

7) int loaderpass1(char\* filename, int filenum)

loader과정 중의 pass1을 처리하는 함수이다. 파일이 존재하지 않으면 에러메세지를 출력하고 -1을 리턴한다. 그다음 csaddr을 저장하고 한줄씩 파일을 읽는다. 첫 줄의 프로그램이름과 시작주소, 길이를 각각 estab, csaddr, cslth에 저장한다. 그 후 E가 나올때까지 while문을 이용하여 계속해서 프로그램을 한줄씩 읽어준다. D가 나온다면 해당 symbol 이름을 estab에 프로그램별로 따로 저장을 해준다. 이때 symbol이름이 이미 estab에 있으면 에러메세지를 출력해주고 return -1한다. E가 나오면 그다음 프로그램의 csaddr을 저장해주고 끝낸다.

8) int loaderpass2(char\* filename, int filenum)

loader과정 중의 pass2를 처리하는 함수이다. 파일이 끝날때까지 파일을열어서 한줄씩 읽어온다. 만약 T가 나온다면 6글자를 읽어 시작 address로 설정한다. 그다음 2글자뒤부터 2글자씩 읽어와 int형식으로 바꾼뒤 해당 address의 memory에 저장해준다. R이 나온다면 reference 이름과 순서를 r에 저장해준다. M이 나온다면 6글자를 읽어서 address로 설정하고 그 뒤 2글자는 5와 6으로 구분하여 몇 개의 메모리를 바꿀지 구분한다. 그다음 부호와 숫자를 읽어서 해당 숫자의 r에저장된 reference의 address 값을 가져온다. 이때 reference가 estab에 없을 때 에러메세지를 출력하고 -1을 리턴해준다. 부호에 따라 더하거나 빼서 메모리값을 변경해준다.

9) void addestab(char\* input, int address, int filenum)

symbol을 estab에 저장해주는 함수이다. file순서에 따라 따로 저장을 한다. linkedlist를 이용하여 head가 NULL이라면 head에 저장하고 이미 있다면 next가 없는 가장 끝 NODE에 값을 저장해준다.

10) int searchestab(char\* input)

해당 input이 estab에 있는지 찾아주는 함수이다. 모든 estab의 node와 값을 비교하여 있다면 input의 address값을 리턴해주고 만약 찾지 못한다면 -1을 리턴한다.