**과목명: 시스템프로그래밍**

**2분반**

**<<Project #3>>**

**서강대학교 수학/컴퓨터공학**

**[학번] 20141303**

**[이름] 이윤제**

목 차

1. **프로그램 개요**
2. **프로그램 설명**
   1. 프로그램 흐름도
3. **모듈 정의**
   1. loader 모듈
      1. 기능
      2. 사용 변수
      3. 함수 목록
4. **전역 변수 정의**
5. **알고리즘 / 구현 방식**
6. **코드 설명**
   1. loader.h
   2. loader.c
7. **프로그램 개요**

**프로젝트2에서 구현한 프로그램에 object 파일을 메모리에 적재하는 loader/linker 기능과 적재된 code 부분을 수행하는 run, 이를 debug할 수 있게 도와주는 bp 기능을 구현하는 프로젝트입니다.**

1. **프로그램 설명**

**2-1. 프로그램 흐름도**

Base 모듈

Opcode 모듈

Main 모듈

Shell 모듈

Memory 모듈

Assembler 모듈

Error 모듈

Loader 모듈

프로그램의 흐름도는 이전 장의 그림과 같다.

새로 추가된 모듈인 Loader 모듈을 제외하면 기존에 사용하던 모듈들이기 때문에 추가적인 설명은 불필요하다고 생각했습니다.

Loader 모듈에서 proj1, 2 에서 정의한 대부분의 함수에 대한 사용을 필요로 하기 때문에 기존에 shell 바로 위에 있던 assembler 모듈을 include 하게 했습니다.

1. **모듈 정의**

**3.1. Loader 모듈**

3.1.1 기능

입력으로 주어진 object 파일들에 대해 기존에 설정된 program address를 이용하여 적재할 메모리 위치를 잘 설정하고, 파일 내용에 맞게 linking/loading을 수행한다.

3.1.2 사용 변수

|  |
| --- |
| //progaddr  int prog\_addr = 0;  //symbol list root for maximum 3 files  struct symbol\_node\* symbol\_root[3] = {NULL, }; |

3.1.3 함수 목록

|  |  |
| --- | --- |
| int progaddr(int addr); |  |
| int find\_loc\_by\_name(char\* name); |  |
| int find\_loc\_by\_idx(char\* idx); |  |
| void insert\_loader\_symbol(struct symbol\_node\* root, char\* name, int loc); |  |
| void clear\_loader\_symbol(); |  |
| int loader\_pass1(char\* fname1, char\* fname2, char\* fname3); |  |
| int loader\_pass2(char\* fname1, char\* fname2, char\* fname3); |  |
| int loader(char\* fname1, char\* fname2, char\* fname3); |  |
| int bp(); |  |
| int runn(); |  |

1. **전역 변수 정의**

**전역 변수를 포함한 사용 변수들에 대해서 각 모듈의 [사용 변수]에서 설명했습니다.**

1. **알고리즘 / 구현 설명**

기본적으로 교재와 강의자료에 설명된 내용에 따라 진행했습니다.

Progaddr 함수는 기존에 0으로 초기화했던 전역변수 prog\_addr 의 값을 변경해주기만 하면 됩니다.

Loader의 Pass1에서는 각 obj 파일을 하나씩 읽으며, 형식에 맞게 출력하는 것을 목표로 했는데, 우선 이미 등장했던 symbol이 다시 등장하면 duplicate symbol 에러를 내보내고 함수를 종료하도록 했습니다.

이렇게 symbol들을 다 쌓고 나면 for loop를 넘기기 전에 각 symbol들에 대한 이름과, load될 주소를 출력해주고 넘어갑니다.

각 obj 파일의 길이는 tot 변수에 쌓아주고, 마지막에 출력합니다.

1. **코드 및 구현 설명**

**5.1 Loader.h**

|  |
| --- |
| #include "assembler.h"  int progaddr(int addr);  int find\_loc\_by\_name(char\* name);  int find\_loc\_by\_idx(char\* idx);  void insert\_loader\_symbol(struct symbol\_node\* root, char\* name, int loc);  void clear\_loader\_symbol();  int loader\_pass1(char\* fname1, char\* fname2, char\* fname3);  int loader\_pass2(char\* fname1, char\* fname2, char\* fname3);  int loader(char\* fname1, char\* fname2, char\* fname3);  int bp();  int runn(); |

**5.2 Loader.c**

|  |
| --- |
| #include "loader.h"  //progaddr  int prog\_addr = 0;  //symbol list root for maximum 3 files  struct symbol\_node\* symbol\_root[3] = {NULL, };  //set progaddr  int progaddr(int addr) {  if(!is\_valid\_addr(addr)) return ERROR;  prog\_addr = addr;  return SUCCESS;  }  //find loc value for symbol with symbol name  int find\_loc\_by\_name(char\* name){  int i;  for(i = 0; i < 3; i++) {  struct symbol\_node\* r = symbol\_root[i];  while(r != NULL && strcmp(r->name, name) != 0) {  r = r->next;  }  if(r!=NULL) return r->LOC;  }  return ERROR;  }  //find loc value for symbol with symbol idx  int find\_loc\_by\_idx(char\* idx){  int i;  for(i = 0; i < 3; i++) {  struct symbol\_node\* r = symbol\_root[i];  while(r != NULL && strcmp(r->idx, idx) != 0) {  r = r->next;  }  if(r!=NULL) return r->LOC;  }  return ERROR;  }  //insert symbol node to symbol list  void insert\_loader\_symbol(struct symbol\_node\* root, char\* name, int loc) {  struct symbol\_node\* new\_ = (struct symbol\_node\*)malloc(sizeof(symbol\_node));  new\_->next = NULL;  strcpy(new\_->name, name);  new\_->LOC = loc;  if(root==NULL) {  root = new\_;  }  else {  new\_->next = root->next;  root = new\_;  }  return;  }  //clear all symbol list  void clear\_loader\_symbol() {  int i;  for(i = 0; i < 3; i++) {  struct symbol\_node\* r = symbol\_root[i];  struct symbol\_node\* tmp;  while(r != NULL) {  tmp = r;  r = r->next;  free(tmp);  }  }  return;  }  //pass1  int loader\_pass1(char\* fname1, char\* fname2, char\* fname3) {  int i, j, val, len, tot = 0;  char line[50];  printf("control symbol address length\nsection name\n");  printf("-----------------------------------\n");  int fi;  for(fi = 0; fi < 3; fi++) {  char\* cur;  if(fi==0) cur = fname1;  else if(fi==1) cur = fname2;  else cur = fname3;  if(cur == NULL) continue;  FILE\* fp;  struct symbol\_node\* r;  fp = fopen(cur, "r");  if(fp == NULL) continue;  fscanf(fp, "%s", line);    for(j=1; j<7; j++) printf("%c", line[j]);  printf("\t\t");  char valstr[7], valname[7];  for(j=7;j<13;j++) valstr[j-7] = line[j];  valstr[6] = 0;  val = str\_to\_hex(valstr);  val += prog\_addr;  len = str\_to\_hex(line + 13);  printf("%04x\t%04x\n", val, len);  tot += len;  fscanf(fp, "%s", line);  int walk = 1, mx = strlen(line);  while(walk < mx) {  for(i=walk; i< walk+6;i++) valname[i - walk] = line[i];  for(i=walk + 6; i< walk+12;i++) valstr[i - walk - 6] = line[i];  val = str\_to\_hex(valstr);  val += prog\_addr;  if(find\_loc\_by\_name(valname) != ERROR) {  printf("Error! Duplicate Symbol\n");  return ERROR;  }  else {  r = symbol\_root[fi];  insert\_loader\_symbol(r, valname, val);  }  walk += 12;  }  r = symbol\_root[fi];  while(r != NULL) {  printf("\t\t%s\t%04x\n", r->name, r->LOC);  r = r->next;  }  }  printf("-----------------------------------\n");  printf("\t\t total length ");  printf("%04x\n", tot);  return SUCCESS;  }  int loader\_pass2(char\* fname1, char\* fname2, char\* fname3) {  return SUCCESS;  }  int loader(char\* fname1, char\* fname2, char\* fname3){  int err = loader\_pass1(fname1, fname2, fname3);  if(err == ERROR) return ERROR;  clear\_loader\_symbol();  return SUCCESS;  }  int bp() {  printf("not completed...\n");  return SUCCESS;  }  int runn() {  printf("not completed...\n");  return SUCCESS;  } |