شرح پروژه اول معماری کامپیوتر

فروردین ماه ۱۴۰۱

استاد مربوطه : دکتر مهران رضایی

تهیه کننده : محمد امین صابری – ۹۹۳۶۲۳۰۲۶

دانشکده مهندسی کامپیوتر دانشگاه اصفهان

بهار ۱۴۰۱

فهرست مطالب

هدف پروژه :

این پروژه با هدف آشنایی بیشتر با زبان اسمبلی و نحوه اجرای آن می باشد . بنابراین به ساخت اسمبلری می پردازیم که با دریافت دستورات اسمبلی آدرس هر خط را در مبنای ده چاپ و ذخیره کند .

راهکار ارائه شده :

توابعی را طراحی می کنیم که هر خط از دستورات را دریافت کرده و بسته به آنکه شامل چه المان هایی می باشد در متغیر های مخصوص به خود ذخیره کند .

الگوریتم و کد پیاده سازی شده :

```
#include<stdio.h>
#include<string.h>

char** lable = NULL;
int* Final_Address;
char matrix[100][100] = { 0 };
int second_counter = 0;

int lable_finder(char* offset)
{ ...
int R_type(char* registers, char* command)
{ ...
int I_type(char* registers, char* command, int index)
{ ...
int J_type(char* registers, char* command)
{ ...
int director(char* registers, char* command)
{ ...
int  matrix_maker()
{ ...

void lable_checker(char* word, int index)
{ ...
void lable_maker(int c)
{ ...
void read_line(int size)
{ ...
void save_output(int count)
{ ...
int main()
{ ...
```

این برنامه شامل ۳ متغیر global ، ۶ تابع که مقدار عددی int را بر میگردانند و چهار تابع void می باشد که در ادامه نحوه کارکرد هر کدام توضیح داده می شود .

اولین تابع فراخوانی شده ()matrix_maker فایل ورودی assemble program.as را در ماتریسی دو بعدی به نام matrix ذخیره می‌کند و تعداد خطوط این برنامه را بر می‌گرداند .

در ادامه ارایه‌ای به اندازه تعداد خطوط فایل می‌سازیم که با فراخوانی تابع ()lable_maker در صورت وجود برچسب در اول هر خط آن را در متغیر lable ذخیره کند .

تابع ()lable_checker عمل یکتا بودن برچسب را بر عهده دارد یعنی اگر یک برچسبی دوبار تعریف شده باشد ، ارور duplicated داده می شود و برنامه متوقف می‌شود .

تابع ()read_line تعداد خطوط را به عنوان ورودی دریافت می‌کند تا یک دور همه خطوط را بخواند . در این تابع هربار که حلقه اجرا می شود instruction در رشته commend و همه فیلد ها را در رشته registers ذخیره می شود . سپس بررسی می شود که instruction شامل کدام نوع از دستورات J, R, I می باشند که بسته به آن وارد یکی از توابع ()R_type J_type, I_type یا ()director می‌شود .

الگوریتم استفاده شده در توابع ذکر شده J, I, R یکسان است به شکلی که بسته به نوع دستور رجیستر های آن را مقدار دهی کرده و opcode آن را مشخص می کنیم . نکته کلیدی در این الگوریتم استفاده از اپراتور >> شیف به چپ و | Bitwise) (OR یا بیت به بیت) می باشد به طوری که هر چهار بیت مربوط به ریجستر ها و opcode را بر اساس موقعیتی که در ۳۲ بیت ادرس دارند به سمت چپ شیف می دهد و آن ها را or می گیرد و در نهایت در آرایه گلوبال Final_Address ذخیره می‌کند .

وظیفه تابع lable_finder یافتن offset ای می باشد که مقدار آن برچسب است و آدرس برچسب را بر می‌گرداند .

آخرین تابع مورد استفاده save_output هست که آرایه ادرس‌ها را که در Final_Address قرار دارد را در فایل مورد نظر خودمان ذخیره می‌کنیم .

نحوه اجرا از طریق cmd :

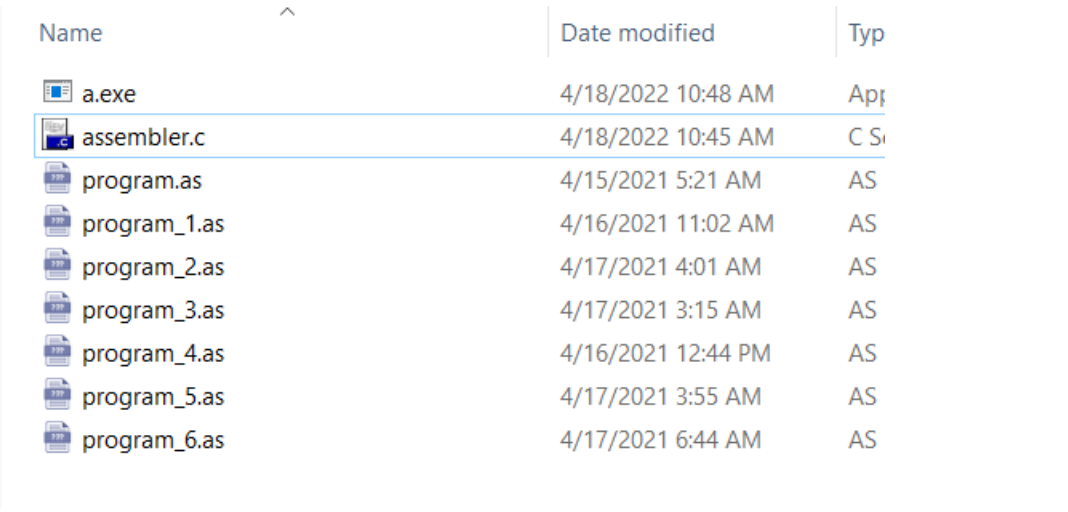
ابتدا cmd را اجرا کرده و به folder برنامه c خود می رویم .

```
Command Prompt
Microsoft Windows [Version 10.0.19042.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MOHAMMADAMIN>cd C:\Users\MOHAMMADAMIN\Desktop\project1_MohammadAmin_saberi

C:\Users\MOHAMMADAMIN\Desktop\project1_MohammadAmin_saberi>
```

دستور gcc -std=c99 assembler.c را اجرا می کنیم تا کد c ما با ورژن ۹۹ ران شود و ارور ها را نادیده بگیرد .

```
C:\Users\MOHAMMADAMIN\Desktop\project1_MohammadAmin_saberi>gcc -std=c99 assembler.c
```

مشاهده می شود که در folder پروژه فایل a.exe به وجود آمده است .



در ادامه تنها با تایپ a برنامه اجرا می شود و از ما ادرس فایل دستورات را می خواهد .

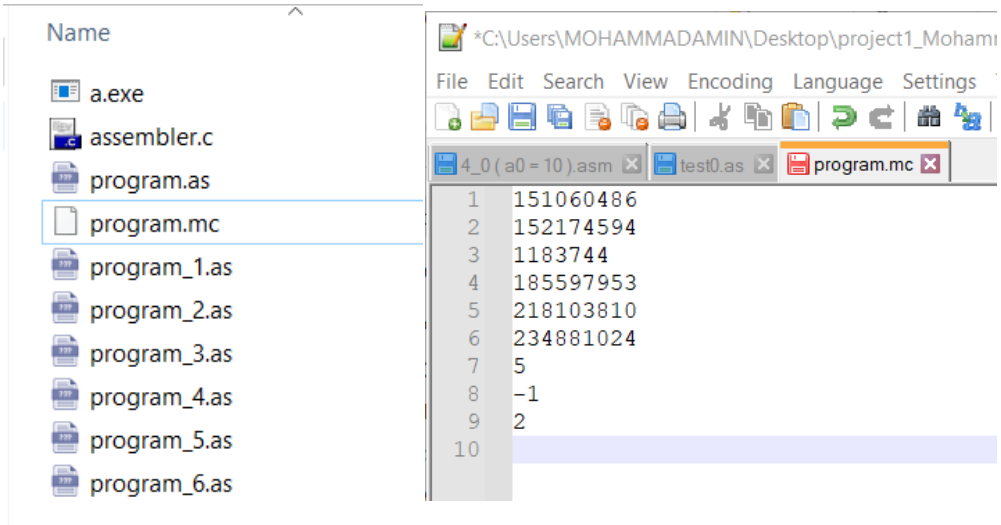
```
C:\Users\MOHAMMADAMIN\Desktop\project1_MohammadAmin_saberi>a

Enter your file's address :
```

ادرس فایل ورودی را میزنیم ، فقط به جای یک \ بک اسلش باید \\ دوتا زد .

```
Command Prompt - a

Enter your file's address : C:\\Users\\MOHAMMADAMIN\\Desktop\\project1_MohammadAmin_saberi\\program.as
        lw      1,0,five        # load reg1 with 5 (symbolic address)
        lw      2,1,2           # load reg2 with -1 (numeric address)
start   add     1,1,2           # decrement reg1
        beq     0,1,done        # goto end of program when reg1==0
        j       start           # go back to beginning og the loop
done    halt                    # end of program
five    .fill   5
neg1    .fill   -1
stAddr  .fill   start           # will contain the address of start (2)
151060486
152174594
1183744
185597953
218103810
234881024
5
-1
2
Enter your saving file's address and name :
```

مشاهده می‌شود که فایل ورودی یکبار چاپ شده تا از درست بودن آن مطمئن شویم و در ادامه آدرس فایلی که می‌خواهیم در آنجا ذخیره شود و اسم آن را وارد می‌کنیم .

```
Enter your saving file's address and name : C:\\Users\\MOHAMMADAMIN\\Desktop\\project1_MohammadAmin_saberi\\program.mc
```



مشاهده می شود که آدرس‌ها در فایل program.mc ذخیره شده‌اند .

خطاهای تعریف شده :

- در صورت استفاده از برچسب تعریف نشده :

This lable : … is not identified, Please try again later …

- بزرگتر بودن عدد offset از ۱۶ بیت یعنی بزگتر از 65535 :

This offset has more than 16 bits, Please try again later …

- آدرس دهی اشتباه برای ورودی دادن فایل program.as :

Wrong file or path !!! Please try again later …

- برچسب تکراری :

The lable … is dublicated, Please try again later …