21MIS1001 02-08-2024

Sabbarish S Monday

SWE4001

System Programming

LAB-3

Pass-2 Assembler

CODE:

#include<bits/stdc++.h>

using namespace std;

class Pass2{

string programName;

int startAddress;

int lenPro;

int loctr;

vector<map<string,string>> instructOp;

map<string, string>opTab = {

{"ADD", "1A"},

{"AND", "40"},

{"COMP", "28"},

{"DIV", "24"},

{"J", "3C"},

{"JEQ", "30"},

{"JGT", "34"},

{"JLT", "38"},

{"JSUB", "48"},

{"LDA", "00"},

{"LDCH", "50"},

{"LDL", "08"},

{"LDX", "04"},

{"MUL", "20"},

{"OR", "44"},

{"RD", "D8"},

{"RSUB", "4C"},

{"STA", "0C"},

{"STCH", "54"},

{"STL", "14"},

{"STSW", "E8"},

{"STX", "10"},

{"SUB", "1C"},

{"TD", "E0"},

{"TIX", "2C"},

{"WD", "DC"}

};

public:

map<string, string> sumTabdata;

void writeProCode(){

fstream pgm("objpgm.txt",ios::app);

if(pgm.is\_open()){

pgm << "H" << "\t" << setw(6) << left << programName

<<"\t"<< setw(6) << setfill('0') << right << hex << startAddress<<"\t"

<< setw(6) << lenPro << "\n";

while (!instructOp.empty()) {

string textRec = "T";

int d = startAddress;

int recordStart =d;

string objectCode;

while (!instructOp.empty() && objectCode.length() < 60) {

auto instruction = instructOp.front();

instructOp.erase(instructOp.begin());

objectCode += instruction["opcode"];

d+= 3;

}

pgm << textRec<<"\t" << setw(6) << setfill('0') << hex << recordStart <<"\t"

<< setw(2) << setfill('0') << (objectCode.length() / 2) <<"\t"

<< objectCode << "\n";

}

pgm << "E" <<"\t"<< setw(6) << setfill('0') << startAddress << "\n";

}

}

void writeObjCode(vector<string> readLine){

fstream obj("objcode.txt",ios::app);

if(obj.is\_open()){

if(readLine.size()==4){

for(auto d: readLine){

obj<<d<<"\t";

}

loctr = stoi(readLine[0],nullptr, 16);

}else if(readLine[1]=="START"){

startAddress = stoi(readLine[2],nullptr, 16);

programName = readLine[0];

obj<<"\t\t\t";

for(auto d: readLine){

obj<<d<<"\t";

}

}else if(readLine[0]=="END"){

lenPro = loctr - startAddress;

obj<<"\t\t";

obj<<readLine[0]<<"\t\t\t\t"<<readLine[1];

}

if(opTab.find(readLine[1]) !=opTab.end()){

map<string, string> instruction;

instruction["address"] = to\_string(loctr);

instruction["opcode"] = opTab[readLine[1]] + sumTabdata[readLine[2]];

instructOp.push\_back(instruction);

loctr = stoi(readLine[0],nullptr, 16);

obj<<readLine[0]<<"\t\t"<<readLine[1]<<"\t"<<readLine[2]<<"\t\t"<<opTab[readLine[1]]+sumTabdata[readLine[2]];

}

obj<<"\n";

}

}

void readSymTab(){

for (auto itr = sumTabdata.begin(); itr != sumTabdata.end(); ++itr) {

cout << itr->first

<< '\t' << itr->second << '\n';

}

}

void loadSymtab(){

fstream symTabF("symtab.txt",ios::in);

if(symTabF.is\_open()){

string symBuffer;

while(getline(symTabF,symBuffer)){

stringstream ss(symBuffer);

string buffer;

bool isFirst = true;

string var;

string add;

while(ss>>buffer){

if(isFirst){

add = buffer;

isFirst = false;

}else{

var = buffer;

isFirst = true;

}

}

sumTabdata.insert({var, add});

}

cout<<"Symtab loaded"<<endl;

}

}

void splitandstore(string line){

stringstream ss(line);

string buffer;

int index = 0;

vector <string> lineBuffer;

while(ss>>buffer){

lineBuffer.push\_back(buffer);

}

writeObjCode(lineBuffer);

}

void readIntermediateFile(){

fstream inputF("intermediate.txt", ios::in);

if(inputF.is\_open()){

cout<<"LOG: file open success"<<endl;

string buffer;

while(getline(inputF,buffer)){

splitandstore(buffer);

}

cout<<endl;

}

}

};

int main(){

Pass2 p2;

p2.loadSymtab();

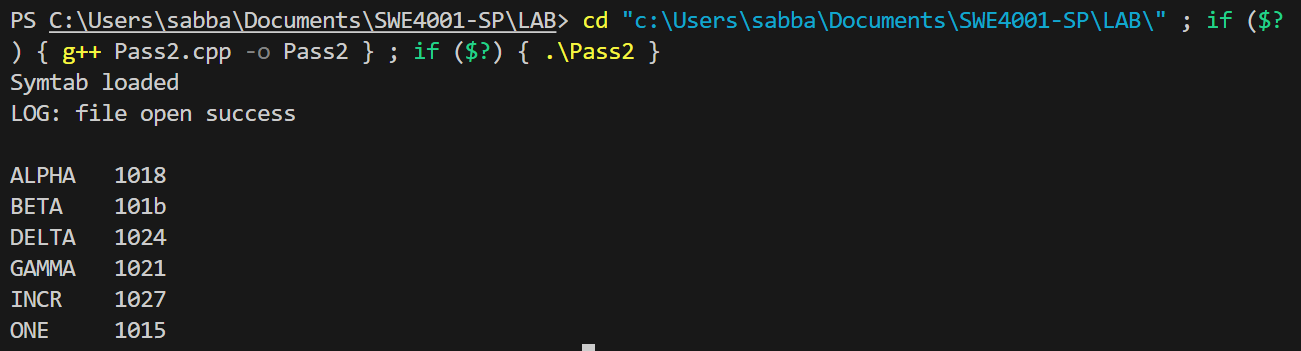
p2.readIntermediateFile();

p2.readSymTab();

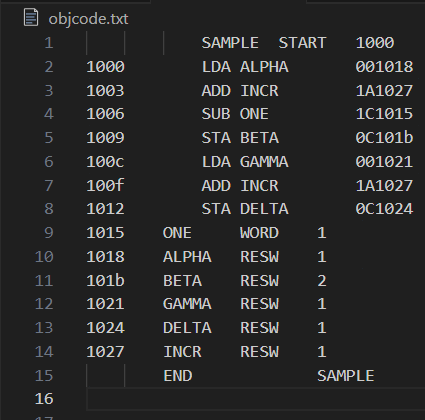
p2.writeProCode();

}

Output:



Objcode.txt



Objpgm

