21MIS1001 13-08-24

Sabbarish S Tuesday

SWE-4001

----------------------------------------------------------------------------------------------------------------------------------------------------

Lab 2 – Pass-1 Assembler

----------------------------------------------------------------------------------------------------------------------------------------------------

CODE:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAXLINE 100

#define MAXSYMTAB 100

#define MAXOPTAB 10

typedef struct {

char label[20];

int address;

} Symbol;

typedef struct {

char mnemonic[10];

int format;

} Opcode;

Symbol symtab[MAXSYMTAB];

int symtabCount = 0;

Opcode optab[MAXOPTAB] = {

{"LDA", 3},

{"ADD", 3},

{"SUB", 3},

{"STA", 3},

{"WORD", 3},

{"RESW", 3},

{"RESB", 1},

{"END", 0},

{"START", 0}

};

int optabCount = 9;

void addSymbol(char \*label, int address) {

strcpy(symtab[symtabCount].label, label);

symtab[symtabCount].address = address;

symtabCount++;

}

int findOpcodeFormat(char \*opcode) {

int i;

for (i = 0; i < optabCount; i++) {

if (strcmp(optab[i].mnemonic, opcode) == 0) {

return optab[i].format;

}

}

return -1; // If the opcode is not found

}

void writeIntermediateFile(FILE \*intermediateFile, int loc, char \*label, char \*opcode, char \*operand) {

fprintf(intermediateFile, "%04X\t%-7s\t%-7s\t%s\n", loc, label, opcode, operand);

}

void writeSymbolTable(FILE \*symtabFile) {

int i;

for (i = 0; i < symtabCount; i++) {

fprintf(symtabFile, "%04X\t%s\n", symtab[i].address, symtab[i].label);

}

}

int main() {

FILE \*inputFile = fopen("input.txt", "r");

FILE \*intermediateFile = fopen("intermediate.txt", "w");

FILE \*symtabFile = fopen("symtab.txt", "w");

if (!inputFile || !intermediateFile || !symtabFile) {

perror("Error opening file");

return EXIT\_FAILURE;

}

// Initialize variables outside the loop

char line[MAXLINE];

char label[20];

char opcode[10];

char operand[20];

int locctr = 0;

int startAddress = 0;

int foundStart = 0;

while (fgets(line, sizeof(line), inputFile)) {

line[strcspn(line, "\n")] = '\0';

if (line[0] == '/' || line[0] == '\0') {

continue;

}

// Reset label, opcode, and operand for each line

label[0] = '\0';

opcode[0] = '\0';

operand[0] = '\0';

// Parse the line to separate label, opcode, and operand

int numFields = sscanf(line, "%s %s %s", label, opcode, operand);

if (numFields == 2) { // No label, just opcode and operand

strcpy(opcode, label);

strcpy(operand, opcode);

label[0] = '\0';

} else if (numFields == 1) { // Only opcode, no label, no operand

strcpy(opcode, label);

label[0] = '\0';

}

if (strcmp(opcode, "START") == 0) {

startAddress = strtol(operand, NULL, 16);

locctr = startAddress;

fprintf(intermediateFile, " %-7s\t%-7s\t%s\n", label, opcode, operand);

foundStart = 1;

continue;

}

if (strcmp(opcode, "END") == 0) {

fprintf(intermediateFile, " %-7s\t%-7s\t%s\n", label, opcode, operand);

break;

}

if (label[0] != '\0') {

addSymbol(label, locctr);

}

int format = findOpcodeFormat(opcode);

if (format > 0) {

writeIntermediateFile(intermediateFile, locctr, label, opcode, operand);

locctr += format;

} else if (strcmp(opcode, "RESW") == 0) {

writeIntermediateFile(intermediateFile, locctr, label, opcode, operand);

locctr += 3 \* atoi(operand);

} else if (strcmp(opcode, "RESB") == 0) {

writeIntermediateFile(intermediateFile, locctr, label, opcode, operand);

locctr += atoi(operand);

} else if (strcmp(opcode, "WORD") == 0) {

writeIntermediateFile(intermediateFile, locctr, label, opcode, operand);

locctr += 3;

} else {

fprintf(stderr, "Invalid opcode: %s\n", opcode);

break;

}

}

if (foundStart) {

writeSymbolTable(symtabFile);

} else {

fprintf(stderr, "Error: No START directive found.\n");

}

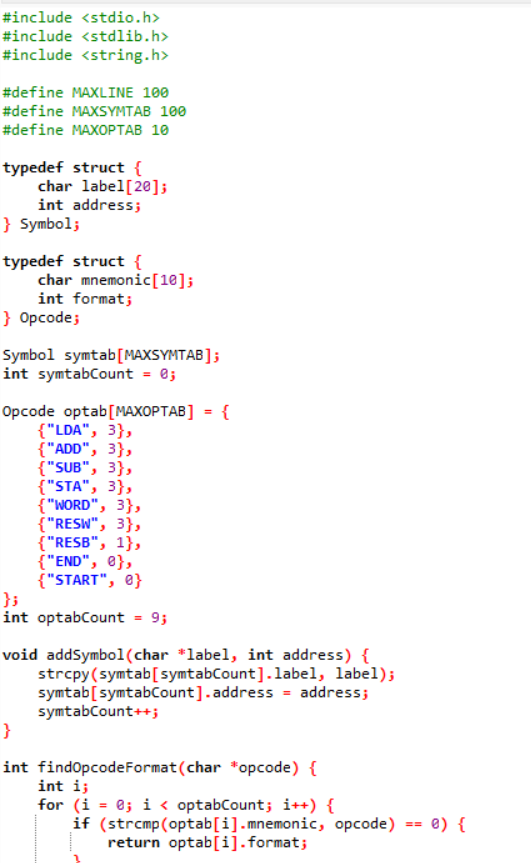
fclose(inputFile);

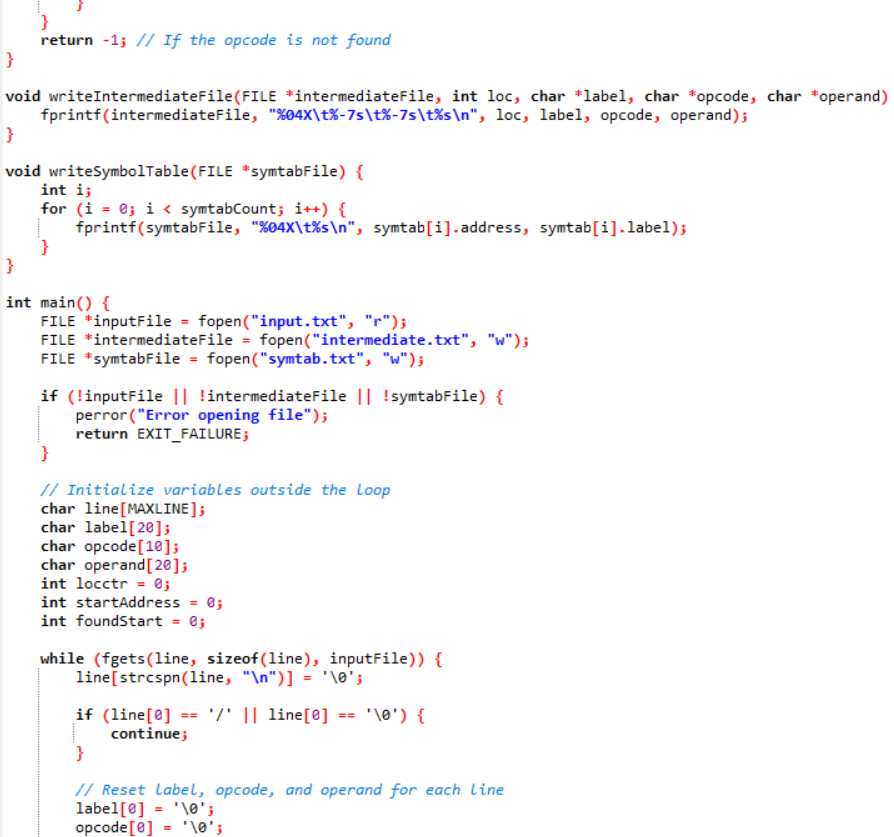
fclose(intermediateFile);

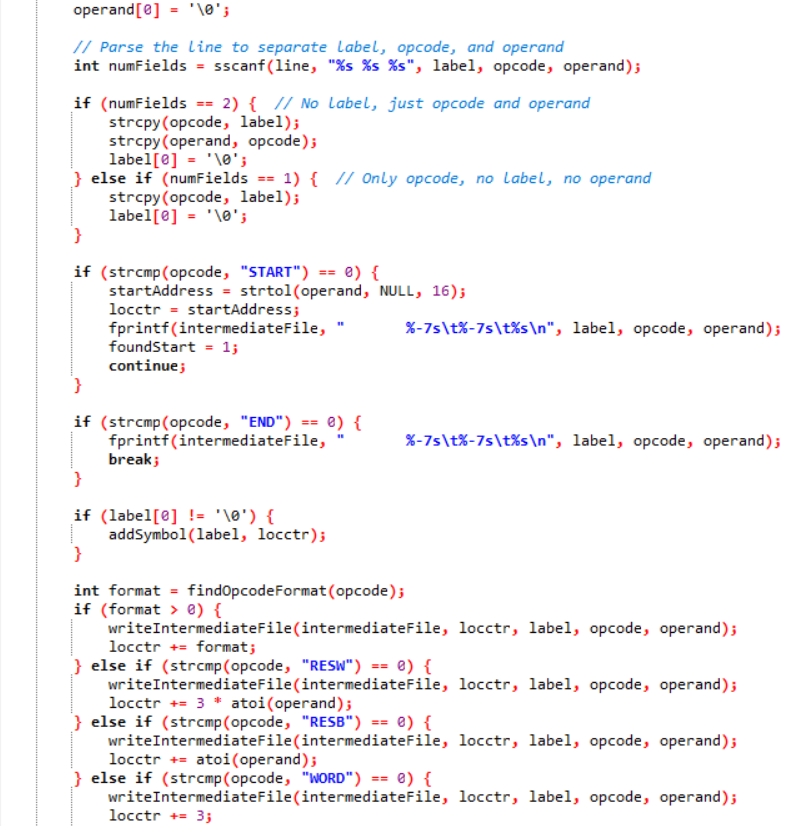
fclose(symtabFile);

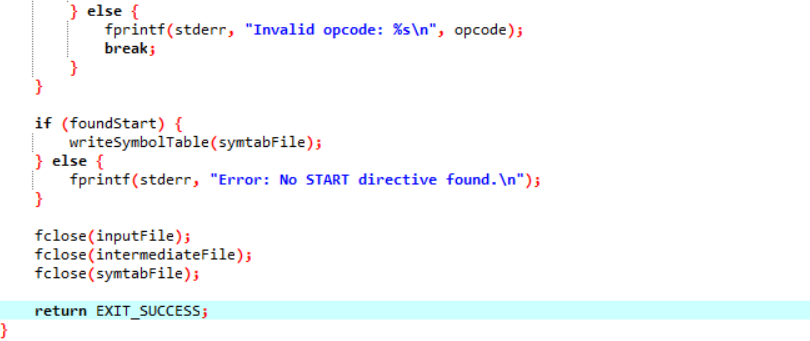
return EXIT\_SUCCESS;

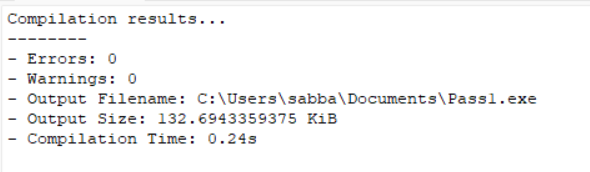
}



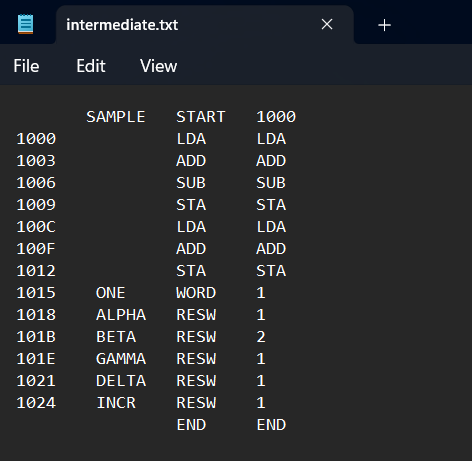








Intermediate.txt



Symtab.txt

