Youwei Zhang

# Code

Excellent work!

9.5/10

# Testing

Very good!

4/4

/\* File: preprocessor.c

\* Author: Youwei Zhang

\* School: Dalhousie University

\* Dept: Electrical and Computer Engineering

\* Course: ECED 3403 Computer Architecture

\*

\* Purpose： This file is the preprocessor that translates legacy XM2 instructions into

\* XM3 instructions, then output an XM3 and report all error

\* Last Modified: 2020.5.30

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define RECORDLENGTH 255

#define TOKENLENGTH 15

#define TABLECONVERTS 35



#define TABLECOLUMN 4

#define NOMORE "XXXNOMORE"

enum

{

NOEXIST = -3,

COMMTKN = -2,

NOMATCH = -1,

ONE\_INST = 0,

TWO\_INST = 1,



INST\_OPER = 2,

INST\_LHOPER = 3,

INST\_RHOPER = 4

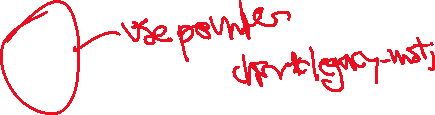
};

struct Convert\_Struct

{

char \_legacy\_inst[TOKENLENGTH];

int \_type;



char \_new\_inst[TOKENLENGTH];

char \_new\_oper[TOKENLENGTH];

};

struct Convert\_Struct convert\_table[TABLECONVERTS] = {

{"SPL0", ONE\_INST, "SETPRI", "#0"}, {"SPL1", ONE\_INST, "SETPRI", "#2"}, {"SPL2", ONE\_INST, "SETPRI", "#3"}, {"SPL3", ONE\_INST, "SETPRI", "#3"}, {"SPL4", ONE\_INST, "SETPRI", "#4"}, {"SPL5", ONE\_INST, "SETPRI", "#5"}, {"SPL6", ONE\_INST, "SETPRI", "#6"}, {"SPL7", ONE\_INST, "SETPRI", "#7"}, {"CLC", ONE\_INST, "CLRCC", "C"}, {"CLN", ONE\_INST, "CLRCC", "N"}, {"CLZ", ONE\_INST, "CLRCC", "Z"}, {"CLV", ONE\_INST, "CLRCC", "V"}, {"SEC", ONE\_INST, "SETCC", "C"}, {"SEN", ONE\_INST, "SETCC", "N"}, {"SEZ", ONE\_INST, "SETCC", "Z"}, {"SEV", ONE\_INST, "SETCC", "V"}, {"RET", ONE\_INST, "MOV", "R5,R7"}, {"BEQ", TWO\_INST, "CEX", "EQ,#1,#0"}, {"BZ", TWO\_INST, "CEX", "EQ,#1,#0"}, {"BNE", TWO\_INST, "CEX", "NE,#1#0"}, {"BNZ", TWO\_INST, "CEX", "NE,#1,#0"}, {"BGE", TWO\_INST, "CEX", "GE,#1,#0"}, {"BLT", TWO\_INST, "CEX", "LT,#1,#0"}, {"BC", TWO\_INST, "CEX", "CS,#1,#0"}, {"BNC", TWO\_INST, "CEX", "CC,#1,#0"}, {"BN", TWO\_INST, "CEX", "MI,#1,#0"}, {"CALL", INST\_OPER, "BL", NOMORE}, {"PUSH", INST\_LHOPER, "ST", "-R6"}, {"JUMP", INST\_LHOPER, "MOV", "R7"}, {"PULL", INST\_RHOPER, "LD", "R6+"}, {"CLR.B", INST\_RHOPER, "MOVL", "#0"}, {"CLR", INST\_RHOPER, "MOVLZ", "#0"}, {"CLR.W", INST\_RHOPER, "MOVLZ", "#0"}, {NOMORE, NOMATCH, NOMORE, NOMORE}};

void Translate(FILE \*xm3, int legacy\_num, char \*oper\_token);



int XM2Check(char token[]);

int main(int argc, char \*\*argv)

{

if (argc != 2)

{

printf("Invalid Input, please try again\n");



return 0;

}

FILE \*xm3 = fopen("solution.asm", "w+");

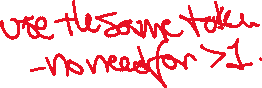
FILE \*xm2 = fopen(argv[1], "r");

char record[RECORDLENGTH];

char delimit[]=" \t\r\n\v\f";

while (fgets(record, sizeof record, xm2) != NULL)

{ // READ record from XM-2 input file



char \*first\_token = strtok(record, delimit);

int legacy\_num = XM2Check(first\_token);



if (legacy\_num > NOMATCH)

{ //If first token is INST Token

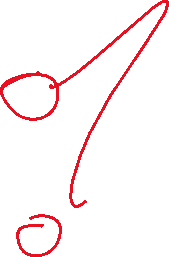


char \*second\_token = strtok(NULL, delimit);

Translate(xm3, legacy\_num, second\_token ? second\_token : NOMORE);

}

else if (legacy\_num == NOMATCH)



{



fprintf(xm3, "%s ", first\_token);

char \*second\_token = strtok(NULL, delimit);

legacy\_num = XM2Check(second\_token);

if (legacy\_num > NOMATCH)

{ //If second token is INST token

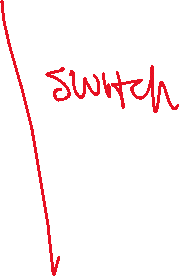
char \*third\_token = strtok(NULL, delimit);



Translate(xm3, legacy\_num, third\_token ? third\_token : NOMORE);

}

else if (legacy\_num > NOEXIST) //Avoid (NULL) in solution file



fprintf(xm3, "%s", second\_token);

}

else if (legacy\_num == COMMTKN)

fprintf(xm3, "%s", first\_token);

char \*token = strtok(NULL, delimit);

while (token != NULL)

{

fprintf(xm3, " %s", token);

token = strtok(NULL, delimit);

}

fprintf(xm3, "\n");

}

fclose(xm2);

fclose(xm3);

return 0;

}

/\*

\* Function: XM2Check

\* ------------------

\* Check whether the input instruction is legacy XM2 instruction

\*

\* token[]: Input instruction toke

\*

\* returns: the index of give instruction in convert\_table

\* returns NOEXIST if it's a comment or NULL

\* returns NOMATCH if it's not match the instruciton in convert\_table

\*/

int XM2Check(char token[])

{

if (token == NULL)

return NOEXIST;

else if (token[0] == ';')

return COMMTKN;

int length = strlen(token)+1;

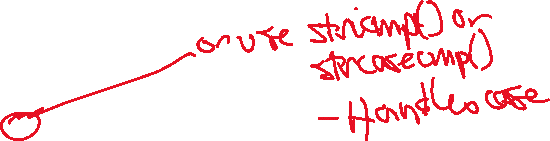
int i;

char temp[length];

for (i = 0; i < length; i++)

{

temp[i] = toupper(token[i]);



}

for (i = 0; i < TABLECONVERTS; i++)

{

if (!strcmp(convert\_table[i].\_legacy\_inst, temp))

return i;

}

return NOMATCH;

}

/\*

\* Function: Translate

\* -------------------

\* Translate the legacy record to xm3 module record, write new record into output file and show

\* error message if oper\_token is missing

\*

\* xm3: Output file

\* legacy\_num: the index of insrtuction in convert\_table

\* oper\_token: the token after instruction, oper token if instruction is not belong to ONE\_INST

\*

\* returns: Void

\*/

void Translate(FILE \*xm3, int legacy\_num, char \*oper\_token)

{

switch (convert\_table[legacy\_num].\_type)

{



case ONE\_INST:

fprintf(xm3, "%s %s", convert\_table[legacy\_num].\_new\_inst,

convert\_table[legacy\_num].\_new\_oper);

if (strcmp(oper\_token, NOMORE) != 0)

fprintf(xm3, " %s", oper\_token);

break;

case TWO\_INST:

fprintf(xm3, "%s %s\n", convert\_table[legacy\_num].\_new\_inst,

convert\_table[legacy\_num].\_new\_oper);

if (!strcmp(oper\_token, NOMORE) || oper\_token[0] == ';')

fprintf(xm3, "Mssing Legacy Operand");

else

fprintf(xm3, "BRA %s", oper\_token);

break;

case INST\_OPER:

if (!strcmp(oper\_token, NOMORE) || oper\_token[0] == ';')

fprintf(xm3, "Mssing Legacy Operand");

else

fprintf(xm3, "%s %s", convert\_table[legacy\_num].\_new\_inst,

oper\_token);

break;

case INST\_LHOPER:

if (!strcmp(oper\_token, NOMORE) || oper\_token[0] == ';')

fprintf(xm3, "Mssing Legacy Operand");

else

fprintf(xm3, "%s %s,%s", convert\_table[legacy\_num].\_new\_inst,

oper\_token, convert\_table[legacy\_num].\_new\_oper);

break;

case INST\_RHOPER:

if (!strcmp(oper\_token, NOMORE) || oper\_token[0] == ';')

fprintf(xm3, "Mssing Legacy Operand");



else

fprintf(xm3, "%s %s,%s", convert\_table[legacy\_num].\_new\_inst,

convert\_table[legacy\_num].\_new\_oper, oper\_token);

break;

default:

break;

}

}