Weekly Report 4

Embedded Micro System Design (EEL 5930)

**Project : ASM2MIF Compiler Design**

**Author : Sourindu Chatterjee**

**Date : November 11, 2015**

|  |  |  |
| --- | --- | --- |
| Tasks | % of completion | % Of difficulty |
| Flex definations | **100%** | **15%** |
| Flex Rules | **80%** | **50%** |
| Flex User Code | **80%** | **60%** |
| Asm instructions | **100%** | **70%** |
| mIF instructions | **90%** | **70%** |

**MIF Instructions**

**Code SCOMP Instruction:**

**variable B : 5555**

**Variable C : 1111**

**VariABle A : 0000**

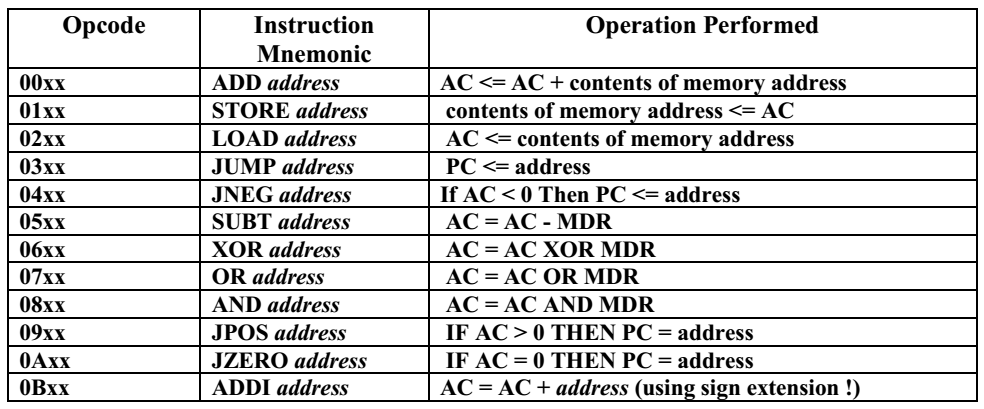
**LOAD B**

**ADD C**

**STORE A**

**L0: JUMP L0**

**Instruction Set:**



**/\* Scanner for assembler to MIF file converter \*/**

**%{**

**#include <stdio.h>**

**#include <string.h>**

**#include <math.h>**

**#include <errno.h>**

**#include <stdlib.h>**

**#include <time.h>**

**#include <ctype.h>**

**#define DEBUG 0**

**int state=0; /\* end of line prints out IW \*/**

**int icount =0; /\* number of instructions \*/**

**int vcount =0, printcount =0; /\* number of variables \*/**

**int pp =1; /\*\* preprocessor flag \*\*/**

**int vvflag = 0; /\*\* variable flag for second run\*\*/**

**int varflag = 0; /\*\* variable flag for first run \*\*/**

**int valflag = 0;**

**int magcount = 0;**

**int ccflag = 0;**

**char buf[256];**

**char opis[6],lblis[4],immis[4], varname[20][10];**

**struct inst {int adr; int imm; char \*txt; char \*opc;} iw;**

**struct init {char \*name; char \*code;} op\_table[20] =**

**{**

**"ADD" , "00",**

**"SUBT" , "05",**

**"AND" , "08",**

**"OR" , "07",**

**"STORE" , "01",**

**"ADDI" , "0B",**

**"XOR" , "06",**

**"JUMP" , "03",**

**"LOAD" , "02",**

**"0" , "0"**

**};**

**FILE \*fid;**

**int add\_symbol(int value, char \*symbol);**

**int lookup\_symbol(char \*symbol);**

**void list\_symbols();**

**void conv2hex(int value, int Width);**

**char\* lookup\_opc(char \*opc);**

**%}**

**DIGIT [0-9]+**

**VAR [a-zA-Z][a-z0-9\_]\***

**COLON [ \t]\*":"[ \t]\***

**LABEL [Ll][0-9]+[:]**

**COMMENT "--"\***

**%%**

**([\r\n]+)|"\r\n"+ {if (pp) printf( "\n");**

**else{ if ((state==2) && (pp==0)) /\* print out an instruction at end of line \*/**

**{conv2hex(iw.adr,8);printf(" : %s",iw.opc); conv2hex(iw.imm,8);**

**printf("; -- %s %s\n",opis,immis);**

**}**

**state=0;iw.imm=0;**

**}}**

**{COMMENT} {printf("\nFound Comment");}/\*Buffer the Comment\*/**

**(?i:variable) {if (pp) {printf("\nThe Variable "); varflag = 1;}}**

**{VAR} {if (pp) {if(varflag){varflag = 0;valflag = 1; strcpy(varname[magcount++], yytext); printf(" %s",yytext);} else{ printf(" %s", yytext); add\_symbol(vcount, yytext);}}**

**else {if(vvflag){vvflag = 0;state=2;iw.imm=lookup\_symbol(yytext)+20;strcpy(immis,yytext);}**

**else if(ccflag){ccflag = 0;state=2;iw.imm=lookup\_symbol(strcat(yytext,":"));**

**strcpy(immis,yytext);}}}**

**{DIGIT} { if (pp){if(valflag){valflag =0; printf( " := %d\n",atoi(yytext)); strcpy(varname[magcount],yytext);}}}**

**STORE|ADD|LOAD { if (pp) printf( "%d) Instruction : %s",printcount++, yytext );**

**else { state=1;iw.adr=icount++;iw.opc=lookup\_opc(yytext); vvflag = 1;}}**

**JUMP { if (pp) printf( "%d) Instruction : %s",printcount++, yytext );**

**else { state=1;iw.adr=icount++;iw.opc=lookup\_opc(yytext); ccflag = 1;}}**

**{LABEL} { if (pp) {printf( "A label: %s lenth=%d Icount=%d\n", yytext , yyleng,printcount);**

**add\_symbol(printcount, yytext);}}**

**{COLON} /\*do nothing\*/**

**[" "\t]+ {/\* eat up whitespace printf("\nSpace Found");\*/}**

**. {printf("\nUnknown Charecter : %s", yytext);}**

**%%**

**int yywrap(void) { return 1; };**

**void main( argc, argv )**

**int argc;**

**char \*\*argv;**

**{**

**++argv, --argc; /\* skip over program name \*/**

**if ( argc > 0 )**

**yyin = fopen( argv[0], "r" );**

**else**

**yyin = stdin;**

**yylex();**

**if (yyin != NULL) fclose(yyin);**

**pp=0;**

**printf("\nDEPTH = 256; %% Memory depth and width are required %%");**

**printf("\nWIDTH = 16; %% Enter a decimal number %%");**

**printf("\nADDRESS\_RADIX = HEX; %% Address and value radixes are optional %%");**

**printf("\nDATA\_RADIX = HEX; %% Enter BIN, DEC, HEX, or OCT; unless %%");**

**printf("\n %% otherwise specified, radixes = HEX %%");**

**printf("\n -- Specify values for addresses, which can be single address or range");**

**printf("\nCONTENT");**

**printf("\n BEGIN");**

**printf("\n");**

**printf("\n[00..FF] : 0000; %% Range--Every address from 00 to FF = 0000 (Default) %%\n");**

**yyin = fopen( argv[0], "r" );**

**yylex();**

**printf("\n");**

**}**

**/\* define a linked list of symbols \*/**

**struct symbol {**

**char \*symbol\_name;**

**int symbol\_value;**

**struct symbol \*next;**

**};**

**struct symbol \*symbol\_list; /\* first element in symbol list \*/**

**extern void \*malloc();**

**int add\_symbol(int value, char \*symbol)**

**{**

**struct symbol \*wp;**

**if(lookup\_symbol(symbol) >= 0 ) {**

**printf("--- Warning: symbol %s already defined \n", symbol);**

**return 0;**

**}**

**wp = (struct symbol \*) malloc(sizeof(struct symbol));**

**wp->next = symbol\_list;**

**wp->symbol\_name = (char \*) malloc(strlen(symbol)+1);**

**strcpy(wp->symbol\_name, symbol);**

**if (symbol[0]!='L') vcount++;**

**wp->symbol\_value = value;**

**symbol\_list = wp;**

**return 1; /\* it worked \*/**

**}**

**int lookup\_symbol(char \*symbol)**

**{ int found = -1;**

**struct symbol \*wp = symbol\_list;**

**for(; wp; wp = wp->next) {**

**if(strcmp(wp->symbol\_name, symbol) == 0)**

**{if (DEBUG) printf("-- Found symbol %s value is: %d\n",symbol, wp->symbol\_value);**

**return wp->symbol\_value;}**

**}**

**if (DEBUG) printf("-- Symbol %s not found!!\n",symbol);**

**return -1; /\* not found \*/**

**}**

**char\* lookup\_opc(char \*opc)**

**{ int k;**

**strcpy(opis,opc);**

**for (k=0;op\_table[k].name !=0;k++)**

**if (strcmp(opc,op\_table[k].name)==0) return (op\_table[k].code);**

**printf("\*\*\*\*\*\*\* Ups, no opcode : %s --> exit \n",opc);exit(1);**

**}**

**void list\_symbols()**

**{**

**struct symbol \*wp = symbol\_list;**

**printf("--- Print the Symbol list: ---\n");**

**for(; wp; wp = wp->next)**

**if (wp->symbol\_name[0]=='L') {**

**printf("-- Label : %s line = %d\n",wp->symbol\_name, wp->symbol\_value);**

**} else {**

**printf("-- Variable : %s memory @ %d\n",wp->symbol\_name, wp->symbol\_value);**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\*\*\*\* CONV\_STD\_LOGIC\_VECTOR(value, bits) \*\*\*\*\*\*\*\*\*\*\*/**

**void conv2hex(int value, int Width)**

**{**

**int k, t;**

**int W;**

**extern FILE \*fid;**

**t = value;**

**for (k = Width - 4; k >= 0; k-=4) {**

**W = (t >> k) % 16;**

**printf( "%1x", W);**

**}**

**}**

**====:Output Console:====**

