Project 3: MIPS Decoder

September 19, 2017

1 Description

Due Saturday Oct 7, 2017, 11:59 pm. Read and decode a file of MIPS machine code commands.

2 Problem Statement

The basic instructions of MIPS (the Core Instruction Set) can be converted to machine code. For example, the MIPS instruction `add $s3, $s3, $s0` can be interpreted as `0x0040006c`. The goal of this project is to read a file of MIPS commands written in machine code, and then to extract from the machine code the opcode, func (if there is one), MIPS inst (eg `lw`), the instruction format (R or I instructions only, J format need not be considered), the corresponding values of `rd`, `rs`, and `rt`, and finally, the Immediate value.

2.1 Input

Each input will be followed by a newline. Sample file input:

```
24020004
3c011001
34240000
0040a020
00408820
```

2.2 Sample Output

```
Machine code,opcode (hex),func (hex),MIPS Inst,Format,rd,rs,rt,Imm (hex)
24020004,9,-,addiu,I,-,0,2,0004
3c011001,d,-,ori,I,-,1,4,0001
34240000,f,-,lui,I,-,-,1,0000
0040a020,0,20,add,R,20,2,0,-
00408820,0,20,add,R,17,2,0,-
```

Table 1: Sample Output: out.csv

Machine code	opcode (hex)	func (hex)	MIPS Inst	Format	rd	rs	rt	Imm (hex)
24020004	9	-	addiu	I	-	0	2	0004
3c011001	d	-	ori	I	-	1	4	0001
34240000	f	-	lui	I	-	-	1	0000
0040a020	0	20	add	R	20	2	0	-
00408820	0	20	add	R	17	2	0	-

3 File Input/Output

3.1 File Input: line-by-line

```c
// File Name:   readFileStringly.c
// Description: Reads a file line by line
#include <stdio.h>
#include <stdlib.h>
#define  MAX_NUM_CHARS  20L //max number of chars to read
#define  MAXN  100L       //size of array to store chars read

int main(int argc, char* argv[])   //char** argv also ok
{
    char lineRead[MAXN];
  //open a file from cmd line for reading
    FILE* inFile = fopen(argv[1], "r");

    if( inFile == NULL){
      perror("Error  opening file");  //prints to console
      return(-1);   //report an error if problem
    }

    while( fgets(lineRead, MAX_NUM_CHARS, inFile) != NULL)  {
      puts(lineRead);
  }
    fclose(inFile);
    return 0;
}
```

3.2 File Input: character-by-character

```c
// File Name:   readFileCharly.c
// Description: Reads a file character by character
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[])
  char c;

  FILE* inFile = fopen(argv[1], "r");   //open a file from user for
     reading
  if( inFile == NULL) exit(1);  //you should print out a reasonable
     message here

  c=fgetc(inFile); //if(c == EOF) puts("nothing to do");

  while( c != EOF )  { //read 1 char, stop if at EOF.
    c=fgetc(inFile);
    printf("read %c \n", c);  //for testing purposes. Do not include
       in code
  }
  puts("done");

  fclose(inFile);
  return 0;
}
```

4 Rubric

```
Proj 03 MIPS Machine Code Translator
====================================

Rubric:

1. Program compiles successfully:             4
2. Program reads and parses input correctly:  6
3. Correct output:                            6
4. Submitted as a compressed file:            2
5. whitespace and comments:                   2


Programs that don't compile or run (or run in an infinite loop) will get 0 points.

Coding conventions:
http://www.gnu.org/prep/standards/standards.html
http://www.cs.swarthmore.edu/~newhall/unixhelp/c_codestyle.html
http://ieng9.ucsd.edu/~cs30x/indhill-cstyle.html


Total points:  10

points for above:
-----------------
1. compiles:                           x/4
2. reads input:                        x/6
3. correct output:                     x/6
4. Canvas assignment proper submission x/2 (compressed, 1st 2 lines = name/email)
5. whitespace/comments                 x/2

total:     x/10


Instructor Comments:
====================


Program console output appended below:
======================================
%
```