ECEN 5813

Chutao Wei

Curry Buscher

PES Project 1 Code pdf

readme.md

```
# cu-ecen-5813-project-1
**Title:**
PES Project 1 Readme <br/>
**Name:**
Curry Buscher, Chutao Wei <br/>
**Repository Comments:**
This repository contains 3 programs (see more details in PES Project 1.pdf)
<br/>
**Project Comments:**
No issues so far <br/>
### **Installation/Execution/Editing Notes:**<br/>
**Language:**
C<br/>
**Compiler:**
GCC version 7.4.0<br/>
**Environment:**
Ubuntu 18.04.1<br/>
**License:**
MIT<br/>
```

makefile

```
# Name: makefile
# Description: makefile for project 1 for ECEN 5813 Principle of Embedded
System Software
# Toolchain: gcc compiler version 7.4.0


XCC = gcc # Compiler
LD  = ld  # Linker
CFLAGS  = # Compiler Flags
LDFLAGS = -lc # Linker Flags

DEPS = # header files
OBJ1 = program_1.o
OBJ2 = program_2.o
OBJ3 = program_3.o

all: one two three


one: $(OBJ1)
    $(XCC) -o $@ $^ $(CFLAGS)
program_1.o: program_1.c $(DEPS)
    $(XCC)  -c -o $@ $<  $(CFLAGS)


two: $(OBJ2)
    $(XCC) -o $@ $^ $(CFLAGS)
program_2.o: program_2.c $(DEPS)
    $(XCC)  -c -o $@ $<  $(CFLAGS)



three: $(OBJ3)
    $(XCC) -o $@ $^ $(CFLAGS)
program_3.o: program_3.c $(DEPS)
    $(XCC)  -c -o $@ $<  $(CFLAGS)


clean:
    -rm -f program_3 *.o *.s
```

program_1.c

```c
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>


//edited from https://www.geeksforgeeks.org/program-decimal-binary-conversion/
void octal_dec_hex(FILE * fp,int n, int opSize)
{
    n=abs(n);
    int m=pow(2,opSize)-1;
    printf("!!!!!!!!!!!!!!!!!!!!!!!!!!!%i\n",m);
    if(n<(pow(2,opSize))){
        fprintf(fp, "Octal (abs)\t\t0%o\t0%o\t%o\n", abs(n), m, 0);
        fprintf(fp, "Decimal (abs)\t\t%d\t%d\t%d\n", abs(n), m, 0);
        fprintf(fp, "Hexadecimal (abs)\t\t0x%X\t0x%X\t0x%X\n", abs(n), m, 0);
    }
    else{
        fprintf(fp, "Octal (abs)\t\tERROR\tERROR\tERROR\n");
        fprintf(fp, "Decimal (abs)\t\tERROR\tERROR\tERROR\n");
        fprintf(fp, "Hexadecimal (abs)\t\tERROR\tERROR\tERROR\n");
    }
    return;
}

void decToBin(FILE * fp,int n, int opSize)
{
    int value=n;
    n=abs(n);
    //edited from https://stackoverflow.com/questions/25829669/how-do-you-multiply-a-character-in-c-like-python
    char str[opSize+1];
    memset(str, '0', sizeof(str)-1);
    str[sizeof(str)-1] = '\0';

    printf("%s\n",str);

    // array to store binary number
    char binaryNum[opSize+1];
    memset(binaryNum, '0', sizeof(binaryNum)-1);
    binaryNum[sizeof(binaryNum)-1] = '\0';

    // counter for binary array
    int i = 0;
    while (n > 0) {

        // storing remainder in binary array
        if (n%2==0){
            binaryNum[i] = '0';
        }
        else{
            binaryNum[i] = '1';
        }
```

```
    n = n / 2;
    i++;
}

printf("%s\n",binaryNum);
printf("x%d\n",(int) sizeof(binaryNum));

// printing binary array in reverse order
for (int j = 0; j < sizeof(binaryNum)-1; j++){
    str[opSize-j-1]= binaryNum[j]; //2 for 0b
    printf("%s\n",str);
}

//max
char max[opSize+1];
memset(max, '1', sizeof(max)-1);
max[sizeof(max)-1] = '\0';

//min
char min[opSize+1];
memset(min, '0', sizeof(min)-1);
min[sizeof(min)-1] = '\0';

printf("!!!!!!!!!!%i\n",n);
printf("!!!!!!!!!!%i\n",opSize);
if(abs(value)<(pow(2,opSize))){
    fprintf(fp,"Binary (abs) 0b%s\t0b%s\t0b%s\n",str,max,min);
}
else{
    fprintf(fp,"Binary (abs)\t\tERROR\tERROR\tERROR\n");
}
octal_dec_hex(fp,value,opSize);

memset(max, '1', sizeof(max)-1);
max[sizeof(max)-1] = '\0';
max[0]='0';

min[0]='1';


char newstr[opSize+1];
memset(newstr, '0', sizeof(newstr)-1);
newstr[sizeof(newstr)-1] = '\0';

for (int i=0;i<opSize;i++){
    if (str[i]=='0'){
        newstr[i]='1';
    }
    else{
        newstr[i]='0';
    }
}

int lower=-(pow(2,opSize)/2)+1;
int higher=pow(2,opSize)/2-1;
if (value>=lower && value<=higher){
```

```
        fprintf(fp,"Signed One's
Compliment\t0b%s\t0b%s\t0b%s\n",newstr,max,min);
    }
    else{
        fprintf(fp,"Signed One's Compliment\tERROR\t0b%s\t0b%s\n",max,min);
    }

    signed_Twos(fp,value,opSize);

    for (int i=0;i<opSize;i++){
        newstr[i]=str[i];
    }
    memset(min, '1', sizeof(min)-1);
    lower=-(pow(2,opSize)/2)+1;
    if (value>=lower && value<=higher){
        if(value>=0){
            newstr[0]='0';
        }
        else{
            newstr[0]='1';
        }
        fprintf(fp,"Sign-Magnitude\t\t0b%s\t0b%s\t0b%s\n",newstr,max,min);
    }
    else{
        fprintf(fp,"Sign-Magnitude\t\tERROR\t0b%s\t0b%s\n",max,min);
    }
    return;
}

void signed_Twos(FILE *fp, int n, int opSize){
    int value=n;
    if (value<0){
        n=pow(2,opSize)+n;
    }
    //edited from https://stackoverflow.com/questions/25829669/how-do-you-
multiply-a-character-in-c-like-python
    char str[opSize+1];
    memset(str, '0', sizeof(str)-1);
    str[sizeof(str)-1] = '\0';

    printf("%s\n",str);

    // array to store binary number
    char binaryNum[opSize+1];
    memset(binaryNum, '0', sizeof(binaryNum)-1);
    binaryNum[sizeof(binaryNum)-1] = '\0';

    // counter for binary array
    int i = 0;
    while (n > 0) {

        // storing remainder in binary array
        if (n%2==0){
            binaryNum[i] = '0';
        }
        else{
            binaryNum[i] = '1';
```

```
        }
        n = n / 2;
        i++;
    }

    printf("%s\n",binaryNum);
    printf("x%d\n",(int) sizeof(binaryNum));

    // printing binary array in reverse order
    for (int j = 0; j < sizeof(binaryNum)-1; j++){
        str[opSize-j-1]= binaryNum[j]; //2 for 0b
        printf("%s\n",str);
    }

    //max
    char max[opSize+1];
    memset(max, '1', sizeof(max)-1);
    max[sizeof(max)-1] = '\0';
    max[0] = '0';

    //min
    char min[opSize+1];
    memset(min, '0', sizeof(min)-1);
    min[sizeof(min)-1] = '\0';
    min[0] = '1';
    min[sizeof(min)-2] = '1';

    int higher=pow(2,opSize)/2-1;
    int lower=-pow(2,opSize)/2;
    if (value>=lower && value<=higher){
        fprintf(fp,"Signed Two's
Compliment\t0b%s\t0b%s\t0b%s\n",str,max,min);
    }
    else{
        fprintf(fp,"Signed Two's Compliment\tERROR\t0b%s\t0b%s\n",max,min);
    }
    return;
}

int main(int argc,char *argv[]){

    FILE * fp;
    fp = fopen ("ProgramOne.out", "w+");
    int input[11][3]={{-7, 10, 4}, {-7, 9, 4}, {-7, 10, 5}, {-10, 10, 4},
{236, 10, 8}, {0354, 8, 8}, {0xEB, 16, 8}, {-125, 10, 8}, {65400, 10, 8},
{65400, 10, 16}, {-32701, 10, 16}};
    for (int i=0; i<11; i++){
        int value=input[i][0];
        int radix=input[i][1];
        int operand=input[i][2];
        if (operand!=4 && operand!=8 && operand!=16 || radix!=8 && radix!=10
&& radix!=16 || value>pow(2,operand)-1){

            fprintf(fp,"Binary (abs)\t\tERROR\tERROR\tERROR\n");
            fprintf(fp, "Octal (abs)\t\tERROR\tERROR\tERROR\n");
            fprintf(fp, "Decimal (abs)\t\tERROR\tERROR\tERROR\n");
            fprintf(fp, "Hexadecimal (abs)\t\tERROR\tERROR\tERROR\n");
```

```
            fprintf(fp,"Signed One's Compliment\tERROR\tERROR\tERROR\n");


            fprintf(fp,"Signed Two's Compliment\tERROR\tERROR\tERROR\n");
            fprintf(fp,"Sign-Magnitude\t\tERROR\tERROR\tERROR\n");
            fprintf(fp,"\n\n");

        }
        else{
            decToBin(fp,value,operand);
            fprintf(fp,"\n\n");
        }
    }
    return 0;
}
```

program_2.c

```
#include <stdio.h>

#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void programTwo(void){
    FILE * fp;
    fp = fopen ("ProgramTwo.out", "w+");

    int
in[20]={66,114,117,99,101,32,83,97,121,115,32,72,105,33,7,9,50,48,49,57};
    for(int i=0;i<20;i++){
        char *type="";
        if (in[i]<=32){
            type="Whitespace";
        }
        else if (in[i]>=57 && in[i]<=65 || in[i]>=91 && in[i]<=96 ||
in[i]>=33 && in[i]<=47){
            type="Special Character";
        }
        else if (in[i]<=48 && in[i]<=57){
            type="Digit";
        }
        else if (in[i]<=97 && in[i]<=122){
            type="Upper Case";
        }
        else if (in[i]<=65 && in[i]<=90){
            type="LowerCase";
        }
        fprintf(fp,"Code: %3d\tType: %15s\tASCII Char: %c\n", in[i], type,
in[i]);
    }

    fclose(fp);

    return;
}

int main(int argc,char *argv[]){
    programTwo();
    return 0;
}
```

program_3.c

```c
/**
 * @file  program_3.c
 * @brief program 3 for project 1
 *
 * This is the c program for program 3 specified in project 1
 *
 *
 * @author Chutao Wei
 * @date Jan. 23 2020
 * @verison 1.0
 */

#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
// the following binary pattern code is quoted from William Whyte
// and edited a little bit by me
// https://stackoverflow.com/questions/111928/is-there-a-printf-converter-to-print-in-binary-format?page=1&tab=votes#tab-top

#define BYTE_TO_BINARY_PATTERN "0b%c%c%c%c"
#define BYTE_TO_BINARY(byte)  \
  (byte & 0x08 ? '1' : '0'), \
  (byte & 0x04 ? '1' : '0'), \
  (byte & 0x02 ? '1' : '0'), \
  (byte & 0x01 ? '1' : '0')

/**** ch function ****/
void check_0b1111(uint16_t num)
{
    if ((num == 0b1111)|| (num == 0b1110) || (num == 0b1101) || (num == 0b1011) || (num == 0b0111))
    {
        printf(" which is true\n");
    }
    else
    {
        printf(" which is false\n");
    }
    return;
}

/**** main function ****/
int main(void)
{
    // Step 1: Print the original input in hexadecimal
    uint16_t face = 0xFACE;
    uint16_t temp = 0;
    printf("Original value is: 0x%X\n", face);

    // Step 2: Test if 3 of last 4 bits are on
    temp = face&0xF;
    printf("binary value: "BYTE_TO_BINARY_PATTERN,BYTE_TO_BINARY(temp));
    check_0b1111(temp);
```

```
    //Step 3: Reverse the byte order, print the value in hexadecimal
    face =  ((face&0xF000)>>12) +\
            ((face&0x0F00)>>4 ) +\
            ((face&0x00F0)<<4)  +\
            ((face&0x000F)<<12);

    printf("Reverse byte value is: 0x%X\n", face);
    //Step 4: Test if 3 of last 4 bits are on
    temp = face&0xF;
    printf("binary value: "BYTE_TO_BINARY_PATTERN,BYTE_TO_BINARY(temp));
    check_0b1111(temp);

    //Step 5: Rotate the value by six bits to the left
    face = (face<<6) | (face>>(32-6));
    printf("Rotate shift left 6 bits value is: 0x%X\n", face);

    //Step 6: Test if 3 of last 4 bits are on
    temp = face&0xF;
    printf("binary value: "BYTE_TO_BINARY_PATTERN,BYTE_TO_BINARY(temp));
    check_0b1111(temp);

    //Step 7: Rotate the value by four bits to the right
    face = (face>>4) | (face<<(32-4));
    printf("Rotate shift right 4 bits value is: 0x%X\n", face);

    //Step 8: Test if 3 of last 4 bits are on
    temp = face&0xF;
    printf("binary value: "BYTE_TO_BINARY_PATTERN,BYTE_TO_BINARY(temp));
    check_0b1111(temp);

    return 0;
}
```