Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Томский государственный университет систем
управления и радиоэлектроники»

Факультет систем управления (ФСУ)

Кафедра автоматизированных систем управления (АСУ)

Двоичная арифметика

отчет по лабораторной работе №1 по дисциплине
«Вычислительная техника»

Обучающийся гр. 431-3
_____ Сергиевский Д.В.
«12__» 09_____ 2022 г.

Проверил: доцент каф. АСУ, к.т.н.
_____ Алферов С.М.
«___» _____ 2022 г.

Томск 2022

Содержание

Введение

Цели работы:

- Получить представление о способах хранения числовых данных в памяти ЭВМ.

Задания:

- Написать программу по выводу битовых данных (битов), числовых значений, хранящихся в переменных. Задать значения и ввести на экран биты: целых знаковых и беззнаковых чисел длиной 1,2 и 4 байта; вещественных чисел с плавающей запятой длиной 4 и 8 байт. Объяснить результат.

1 Ход работы

В рамках данной лабораторной работы была написана небольшая программа по выводу битовых данных. В качестве языка программирования был выбран C, поскольку из-за проблем со слухом была пропущена информация о возможности использования C++. Все результаты были получены с использованием онлайн-компилятора (https://www.onlinegdb.com/online_c_compiler).

Код программы приведен в Приложении А.

1.1 Вывод битовых данных

При запуске программа запрашивает ввод значений для требуемых типов данных и выводит их битовые данные. Пример выполнения программы представлен на Рисунке 1.1.

```
signed char (1 byte) = 5
bits: 0 0000101

signed short (2 byte) = 5
bits: 0 0000000.00000101

signed int (4 byte) = 5
bits: 0 0000000.00000000.00000000.00000101

unsigned char (1 byte) = 5
bits: 00000101

unsigned short (2 byte) = 5
bits: 00000000.00000101

unsigned int (4 byte) = 5
bits: 00000000.00000000.00000000.00000101

float (4 byte) = 5
bits: 0 10000001 0100000.00000000.00000000

double (8 byte) = 5
bits: 0 10000000001 0100.00000000.00000000.00000000.00000000.00000000.00000000
```

Рисунок 1.1 - пример вывода программы

1.2 Представление целых чисел

Целые числа представлены в стандартной записи двоичный чисел. В случае знаковых типов младший бит резервируется для знака, а число хранится в дополнительном коде.

Пример битовых данных для некоторых целых чисел представлен на Рисунке 1.2.

```
Целые числа:
      +0 |       +0
0 0000000 | 0 0000000
      +1 |       -1
0 0000001 | 1 1111111
      +3 |       -3
0 0000011 | 1 1111101
      +7 |       -7
0 0000111 | 1 1111001
     +15 |      -15
0 0001111 | 1 1110001
     +31 |      -31
0 0011111 | 1 1100001
     +63 |      -63
0 0111111 | 1 1000001
    +127 |     -127
0 1111111 | 1 0000001
```

Рисунок 1.2 - вывод битовых данных для некоторых целых чисел

1.3 Представление чисел с плавающей точкой

Числа с плавающей точкой представлены в экспоненциальной записи двоичных чисел. Младший бит резервируется для знака мантиссы. К реальному значению порядка добавляется 127, при этом значения меньше 127 трактуются как отрицательные. Мантисса представлена в прямом коде вне зависимости от знака и записывается в обратном порядке следования бит.

Пример битовых данных для некоторых чисел с плавающей точкой представлен на Рисунках 1.3 и 1.4. Для сокращения записи старшие байты мантиссы опущены, поскольку на данном диапазоне обращены в ноль.

```
Мантисса дробных чисел:
          +0.000000 |          -0.000000
0 00000000 0000000 | 1 00000000 0000000
          +1.000000 |          -1.000000
0 01111111 0000000 | 1 01111111 0000000
          +3.000000 |          -3.000000
0 10000000 1000000 | 1 10000000 1000000
          +7.000000 |          -7.000000
0 10000001 1100000 | 1 10000001 1100000
         +15.000000 |         -15.000000
0 10000010 1110000 | 1 10000010 1110000
         +31.000000 |         -31.000000
0 10000011 1111000 | 1 10000011 1111000
         +63.000000 |         -63.000000
0 10000100 1111100 | 1 10000100 1111100
        +127.000000 |        -127.000000
0 10000101 1111110 | 1 10000101 1111110
        +255.000000 |        -255.000000
0 10000110 1111111 | 1 10000110 1111111
```

Рисунок 1.3 - вывод некоторых чисел с разной мантиссой

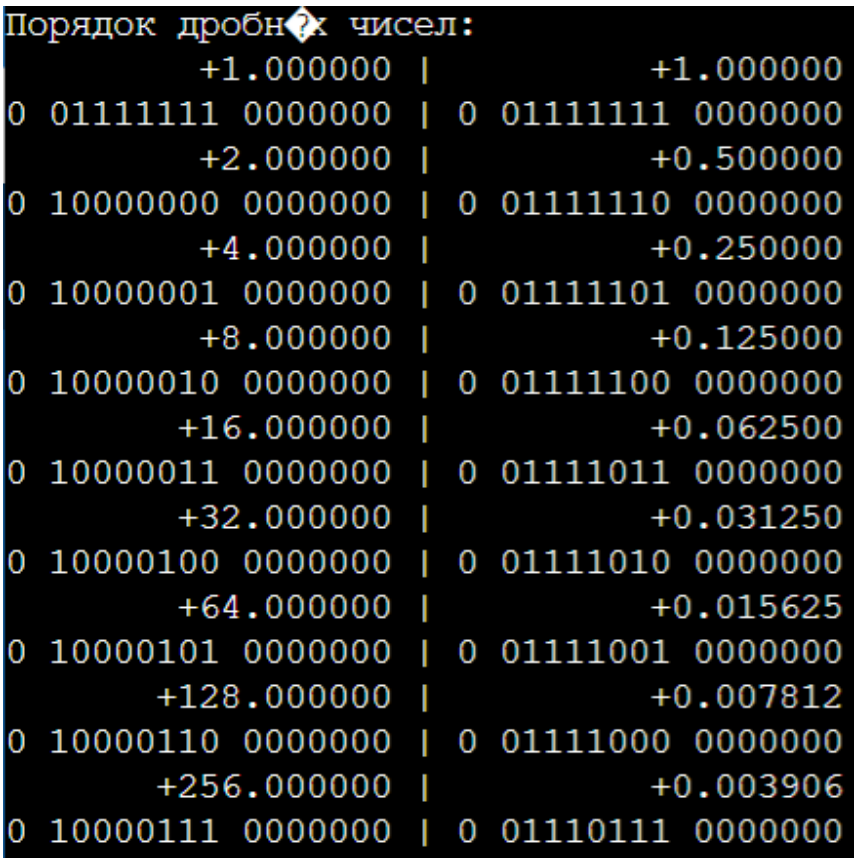

Рисунок 1.4 - вывод некоторых чисел с разным порядком

Вывод

В результате данной лабораторной работы были подтверждены на практике знания о представлении числовых типов данных в памяти компьютера.

Приложение А

Код программы по выводу битовых данных.

```
/*******************************************************************************

                    Online C Compiler.
            Code, Compile, Run and Debug C program online.
Write your code in this editor and press "Run" button to compile and execute it.

*******************************************************************************
/

#include <stdio.h>

void print_bits(void* p, size_t skip_count, size_t print_count)
{
   for(size_t i = 0; i < skip_count+print_count; ++i)
   {
      if(i < skip_count)
         continue;
      printf("%d", (((unsigned char*)p)[(i/8)] >> (i%8) &1));
   }
}

char* _bracket_end(char* start)
{
   char* c = start + 1;
   int count = 1;

   for(; *c != '\0'; ++c)
   {
      count += (*c == '[') - (*c == ']');
```

```
        if(count == 0)
            return c;
    }

    printf("bracket without a pair: %s\n", start);
    return c;
}

int _printf_bits(void* p, char* pattern, size_t index)
{
    int number = 0;
    int escape_flag = 0;

    for(char* c = pattern; ; ++c)
    {

        if(escape_flag)
        {
            print_bits(p, index, number);
            index += number;
            number = 0;

            printf("%c", *c);
            escape_flag = 0;
        }

        if('0' <= *c && *c <= '9')
        {
            number = number*10 + (*c-'0');
            continue;
        }

        if(*c == ' ')
```

```
{
   print_bits(p, index, number);
   index += number;
   number = 0;
   continue;
}

if(*c == '/')
{
   escape_flag = 1;
   continue;
}

if(*c == '[')
{
   if(number == 0)
      number = 1;
   for(; number > 0; --number)
   {
      index = _printf_bits(p, c+1, index);
   }

   c = _bracket_end(c);

   continue;
}

if(*c == '>')
{
   index += number;
   number = 0;
   continue;
}
```

```
        print_bits(p, index, number);
        index += number;
        number = 0;

        if(*c == '\0' || *c == ']')
            return index;

        printf("%c", *c);
    }
}

size_t bit_count_from_pattern(char* pattern)
{
    size_t sum = 0;
    size_t number = 0;
    int escape_flag = 0;

    for(char* c = pattern; ; ++c)
    {
        if(escape_flag)
        {
            escape_flag = 0;
            sum += number;
            number = 0;
            continue;
        }

        if('0' <= *c && *c <= '9')
        {
            number = number*10 + *c - '0';
            continue;
        }
```

```
        if(*c == '/')
        {
            escape_flag = 1;
            continue;
        }

        if(*c == '[')
        {
            if(number == 0)
                number = 1;
            for(; number > 0; --number)
                sum += bit_count_from_pattern(c+1);

            c = _bracket_end(c);
            continue;
        }

        sum += number;
        number = 0;

        if(*c == '\0' || *c == ']')
            return sum;
    }
}

void printf_bits(void*p, char* pattern)
{
    size_t byte_count = bit_count_from_pattern(pattern);
    byte_count = byte_count/8 + (byte_count % 8 != 0);

    unsigned char buf[byte_count];
    for(size_t i = 0; i < byte_count; ++i)
```

```
    {
        buf[i] = 0;
        for(int j = 0; j < 8; ++j)
        {
            buf[i] |= (((unsigned char*)p)[byte_count-i-1] >> (7-j) & 1) << j;
        }
    }

    _printf_bits(&buf, pattern, 0);
}

// *********

void print_data_for_screenshots()
{
    printf("Целые числа:\n");
    char inum = 0;
    for(int i = 1; inum >= 0; inum += i, i *= 2)
    {
        printf("%+9hhd | %+9hhd\n", inum, -inum);
        printf_bits(&inum, "1/ 7");
        printf(" | ");
        inum = -inum;
        printf_bits(&inum, "1/ 7\n");
        inum = -inum;
    }

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

    printf("Мантисса дробных чисел:\n");
    float fnum = 0.0;
    for(int i = 1; i <= 256; fnum += i, i *= 2)
    {
```

```
        printf("%+18f | %+18f\n", fnum, -fnum);
        printf_bits(&fnum, "1/ 8/ 7 2[8>]");
        printf(" | ");
        fnum = -fnum;
        printf_bits(&fnum, "1/ 8/ 7 2[8>]\n");
        fnum = -fnum;
    }

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

    printf("Порядок дробных чисел:\n");
    fnum = 1.0;
    for(int i = 1; i <= 256; i *= 2)
    {
        fnum = 1.0*i;
        printf("%+18f | %+18f\n", fnum, 1/fnum);
        printf_bits(&fnum, "1/ 8/ 7 2[8>]");
        printf(" | ");
        fnum = 1/fnum;
        printf_bits(&fnum, "1/ 8/ 7 2[8>]\n");
    }
}

#define copypaste(type, name, scanf_format, bit_format) \
    type name;\
    printf("\n" #type " (");\
    printf("%ld", sizeof(type));\
    printf(" byte) = ");\
    scanf(scanf_format, &name);\
    printf("bits: ");\
    printf_bits(&name, bit_format);

void interact_with_human()
```

```
{
   copypaste(signed char, s_char, "%hhd", "1/ 7\n")
   copypaste(signed short, s_short, "%hd", "1/ 7.8\n")
   copypaste(signed int, s_int, "%d", "1/ 7 3[.8]\n")

   copypaste(unsigned char, u_char, "%hhu", "8\n")
   copypaste(unsigned short, u_short, "%hu", "8.8\n")
   copypaste(unsigned int, u_int, "%u", "8.8.8.8\n")

   copypaste(float, v_float, "%f", "1/ 8/ 7 2[.8]\n")
   copypaste(double, v_double, "%lf", "1/ 11/ 4 6[.8]\n")

   return;
}


int main()
{
   // print_data_for_screenshots();
   interact_with_human();
   return 0;
}
```