

Пензенский государственный университет
Кафедра «Вычислительная техника»

ОТЧЕТ

по лабораторной работе №2

по дисциплине: "Алгебраические и логические основы вычислительной
техники"

на тему: "Перевод чисел из одной системы счисления в другую.

Формат с ФТ представления чисел в цифровых процессорах"

Выполнил:
xxxxxxxxxxxx

Принял:
xxxxxxxxxxxx

Пенза, 2021

Ход работы

1. Перевела числа $a = 65$ и $b = -85$ в двоичную систему счисления

$$\begin{array}{r|l} 65 & 16 \\ \hline 64 & 4 \\ \hline 1 & \end{array} \quad \begin{array}{r|l} 85 & 16 \\ \hline 80 & 5 \\ \hline 5 & \end{array}$$
$$a = 65_{10} = 41_{16} = 1000001_2 \quad b = -85_{10} = -55_{16} = -1010101_2$$

2. Представила a и b как операнды в прямом и дополнительном коде в процессорах разной разрядности.

Для представления в дополнительном коде двоичная запись числа b была инвертирована, затем результат был сложен с 1.

8-ми разрядный процессор

прямой код:

$$[a]_1 = 0100\ 0001$$

$$[b]_1 = 1101\ 0101$$

дополнительный код:

$$[a]_2 = 0100\ 0001$$

$$[b]_2 = 1010\ 1011$$

16-ти разрядный процессор

прямой код:

$$[a]_1 = 0000\ 0000\ 0100\ 0001$$

$$[b]_1 = 1000\ 0000\ 0101\ 0101$$

дополнительный код:

$$[a]_2 = 0000\ 0000\ 0100\ 0001$$

$$[b]_2 = 1111\ 1111\ 10101011$$

32-разрядный процессор

прямой код:

$$[a]_1 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0100\ 0001$$

$$[b]_1 = 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0101\ 0101$$

дополнительный код:

$$[a]_2 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0100\ 0001$$

$$[b]_2 = 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1010\ 1011$$

3. Полученные представления чисел разместила в оперативной памяти.

Для этого написала следующую программу:

```

Файл Праща Формат Вид Справка
data segment
    a8 db 41h
    a16 dw 41h
    a32 dd 41h

    b8 db 0A8h
    b16 dw 0FFA8h
    b32 dd 0FFFFFFA8h
data ends

code segment
assume cs:code, ds:data, ss:nothing

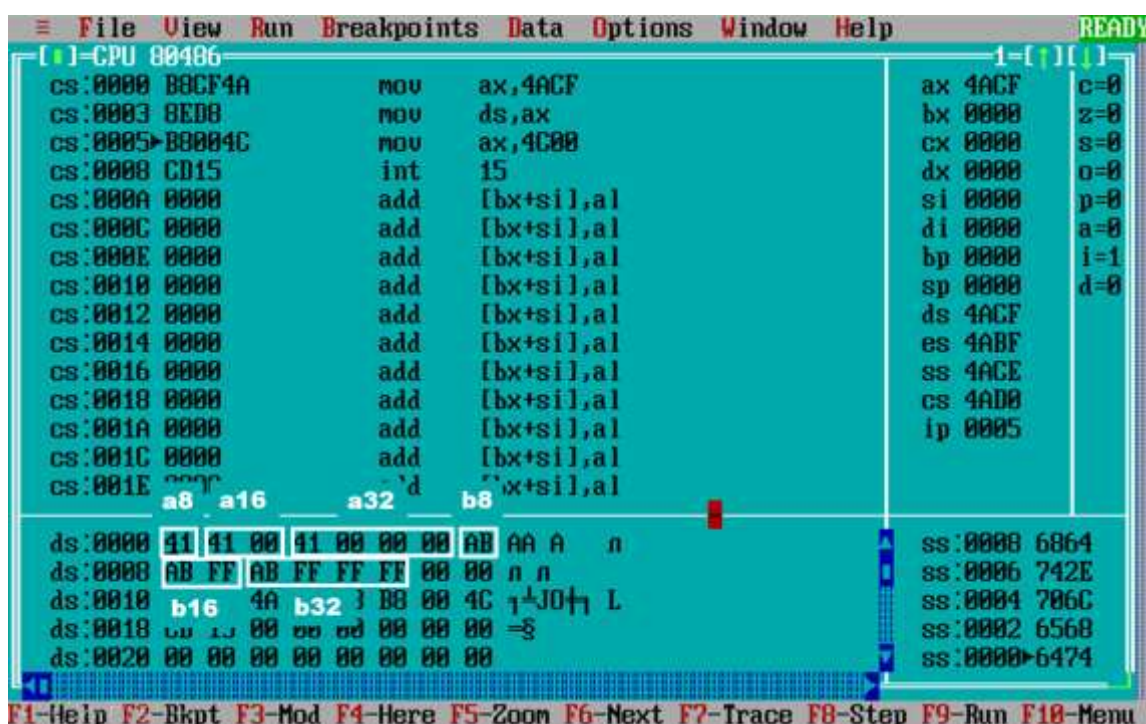
start:    mov ax, data
          mov ds, ax

quit:     mov ax, 4c00h
          int 21

code ends
end start

```

Представление в оперативной памяти:



- Увеличила в оперативной памяти каждого процессора значение младших байтов чисел на 1001_2 .

Измененные значения:

$$\begin{array}{r}
 + 1000001_2 \\
 \underline{1001_2} \\
 1001010_2
 \end{array}$$

$1001010_2 = 4A_{16} = 74_{10}$

$$\begin{array}{r}
 + 10101011_2 \\
 \underline{1001_2} \\
 10110100_2 \\
 \text{инверсия} \\
 + 01001011_2 \\
 \underline{1_2} \\
 01001100_2
 \end{array}$$

$$-1001100_2 = -4C_{16} = -76_{10}$$

Перевод результата для представления в памяти:

$$\begin{array}{r}
 01001100_2 \xrightarrow{\text{инверсия}} 10110011_2 \\
 + 10110011_2 \\
 \underline{1_2} \\
 10110100_2 \\
 10110100_2 = B4_{16}
 \end{array}$$

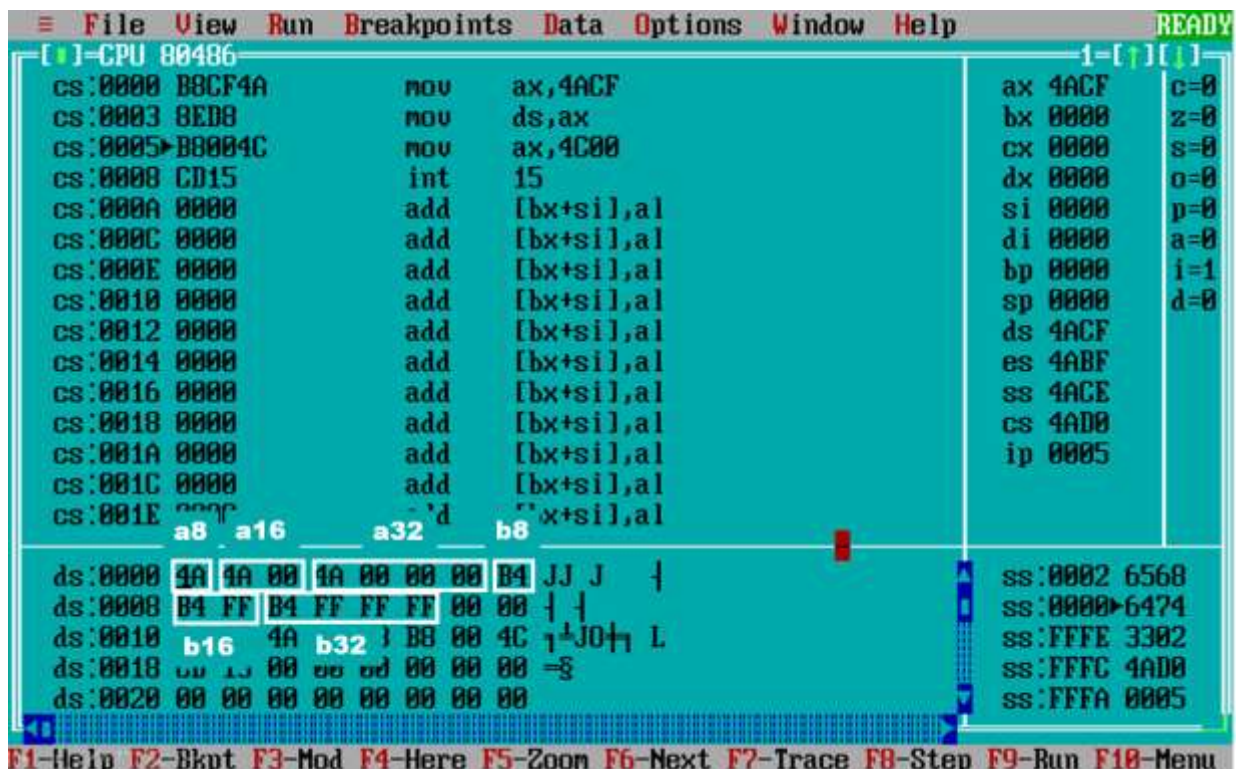
```

data segment
    a8 db 4Ah
    a16 dw 4Ah
    a32 dd 4Ah

    b8 db 0B4h
    b16 dw 0FFB4h
    b32 dd 0FFFFFFB4h
data ends

```

Представление в оперативной памяти:



5. Перевела числа $c = 0,65$ и $d = -0,85$ в двоичную систему счисления.

$c = 0,65$
 $0,65 * 16 = 10,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$
 $0,40 * 16 = 6,4$

$d = -0,85$
 $0,85 * 16 = 13,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$
 $0,60 * 16 = 9,6$

$$c = 0,65_{10} = 0, A6666666_{16} = 0, 1010\ 0110\ 0110\ 0110\ 0110\ 0110\ 0110\ 0110_2$$

$$d = -0,85_{10} = -0, D9999999A_{16} = -0, 1101\ 1001\ 1001\ 1001\ 1001\ 1001\ 1001\ 1010_2$$

6. Представила с и d как операнды в прямом и дополнительном коде в процессорах разной разрядности.

16-разрядный процессор

прямой код:

$$[c]_1 = 0101\ 0011\ 0011\ 0011$$

$$[d]_1 = 1110\ 1100\ 1100\ 1101$$

дополнительный код:

$$[c]_2 = 0101\ 0011\ 0011\ 0011$$

$$[d]_2 = 1001\ 0011\ 0011\ 0011$$

32-разрядный процессор

прямой код:

$$[c]_1 = 0101\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011$$

$$[d]_1 = 1110\ 1100\ 1100\ 1100\ 1100\ 1100\ 1100\ 1101$$

дополнительный код:

$$[c]_2 = 0101\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011$$

$$[d]_2 = 1001\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011$$

7. Полученные представления чисел разместила в оперативной памяти. Для этого написала следующую программу:


```

Файл  Правка  Формат  Вид  Справка
data segment
    c16 dw 5333h
    c32 dd 53333333h

    d16 dw 9333h
    d32 dd 93333333h
data ends

code segment
assume cs:code, ds:data, ss:nothing

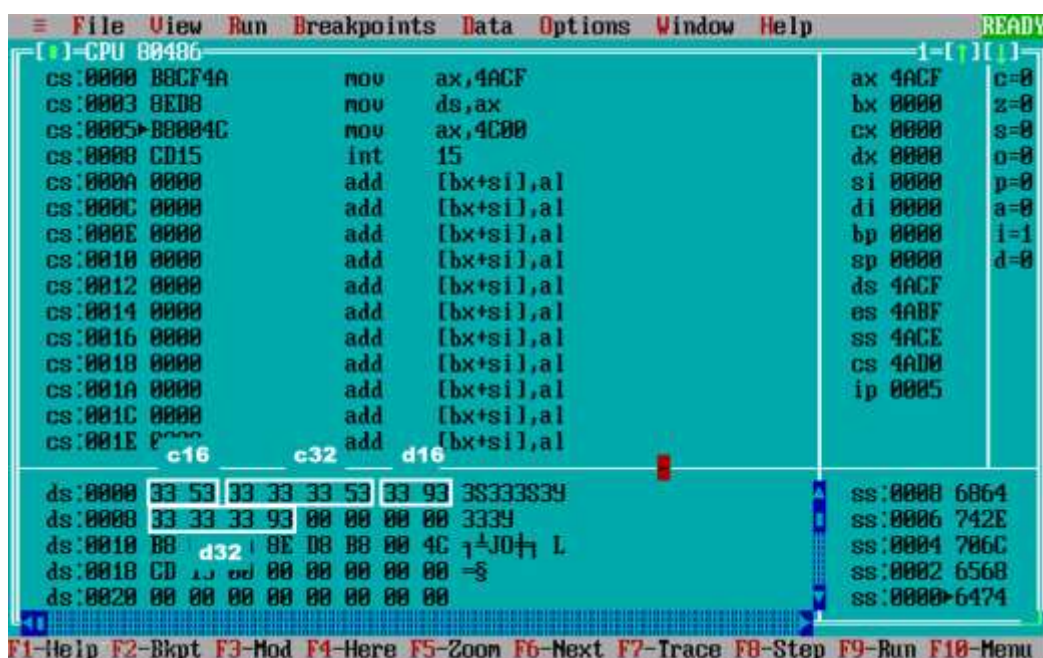
start:    mov ax, data
          mov ds, ax

quit:     mov ax, 4c00h
          int 21

code ends
end start

```

Представление в оперативной памяти:



- Увеличила в оперативной памяти каждого процессора значение младших байтов чисел на 1001_2 .

Изменение значения с:

$$C = 0, A678_{16} = 10 \cdot 16^{-1} + 6 \cdot 16^{-2} + 6 \cdot 16^{-3} + 8 \cdot 16^{-4} = 0,65024_{10}$$

$$C = 0, A6666678_{16} = 10 \cdot 16^{-1} + 6 \cdot 16^{-2} + 6 \cdot 16^{-3} + 6 \cdot 16^{-4} + 6 \cdot 16^{-5} + 6 \cdot 16^{-6} + 7 \cdot 16^{-7} + 8 \cdot 16^{-8} = 0,65_{10}$$

$$\begin{array}{r}
 + \quad 1001 \ 0011 \ 0011 \ 0011_2 \\
 \quad \quad \quad 1001_2 \\
 \hline
 1001 \ 0011 \ 0011 \ 1100_2
 \end{array}
 \xrightarrow{\text{инверсия}}
 \begin{array}{r}
 + \quad 0110 \ 1100 \ 1100 \ 0011_2 \\
 \quad \quad \quad 1_2 \\
 \hline
 0110 \ 1100 \ 1100 \ 0100_2
 \end{array}$$

Представление результата в памяти: $1001\ 0011\ 0011\ 1100_2 = 933C_{16}$

$$d = -0, D9999988_{16} = 13 \cdot 16^{-1} + 9 \cdot 16^{-2} + 9 \cdot 16^{-3} + 9 \cdot 16^{-4} + 9 \cdot 16^{-5} + 9 \cdot 16^{-6} + 8 \cdot 16^{-7} + 8 \cdot 16^{-8} = -0,849997_{10}$$

Представление результата в памяти: $1001\ 0011\ 0011\ 0011\ 0011\ 0011\ 0011\ 1100_2 = 93333333_{16}$



The screenshot shows the CPU 80486 emulator interface. The main window displays assembly code with addresses from CS:0000 to CS:001E. The code includes instructions like `mov ax, 4ACF`, `mov ds, ax`, `mov ax, 4C00`, `int 15`, and a series of `add [bx+si], al` instructions. To the right, a register window shows the current state of registers: `ax 4ACF`, `bx 0000`, `cx 0000`, `dx 0000`, `si 0000`, `di 0000`, `bp 0000`, `sp 0000`, `ds 4ACF`, `es 4ABF`, `ss 4ACE`, `cs 4AD0`, and `ip 0005`. Below the assembly code, a memory dump shows the contents of memory locations starting from DS:0000. The bottom status bar indicates various function keys like F1-Help, F2-Bkpt, etc.

9. Перевела числа $f = 65,85$ и $g = -85,65$ в двоичную систему счисления.

$f = 65,85$		$d = -85,65$	
$\begin{array}{r l} 65 & 16 \\ -64 & 4 \\ \hline 1 & \end{array}$	$0,85 \cdot 16 = 13,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$ $0,60 \cdot 16 = 9,6$	$\begin{array}{r l} 85 & 16 \\ -80 & 5 \\ \hline 5 & \end{array}$	$0,65 \cdot 16 = 10,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$ $0,40 \cdot 16 = 6,4$

$$f = 41, D9999A_{16} = 1000001, 1101\ 1001\ 1001\ 1001\ 1010_2$$

$$g = -55, A66666_{16} = -1010101, 1010\ 0110\ 0110\ 0110\ 0110_2$$

10. Представила f и g как операнды в прямом и дополнительном коде в процессорах разной разрядности.

16-разрядный процессор

прямой код:

$[f]_1 = 0100\ 0001\ 1101\ 1010$ $[g]_1 = 1101\ 0101\ 1010\ 0110$

дополнительный код:

$[f]_2 = 0100\ 0001\ 1101\ 1010$ $[g]_2 = 1010\ 1010\ 0101\ 1010$

32-разрядный процессор

прямой код:

$[f]_1 = 0000\ 0000\ 0100\ 0001\ 1101\ 1001\ 1001\ 1010$

$[g]_1 = 1000\ 0000\ 0101\ 0101\ 1010\ 0110\ 0110\ 0110$

дополнительный код:

$[f]_2 = 0000\ 0000\ 0100\ 0001\ 1101\ 1001\ 1001\ 1010$

$[g]_2 = 1111\ 1111\ 1010\ 1010\ 0101\ 1001\ 1001\ 1010$

11. Полученные представления чисел разместила в оперативной памяти.
Для этого написала следующую программу:

```
part1.asm - Блокнот
Файл Правка Формат Вид Справка
data segment
    f16 dw 41DAh
    f32 dd 0041D99Ah

    g16 dw 0AASAh
    g32 dd 0FFAA599Ah
data ends

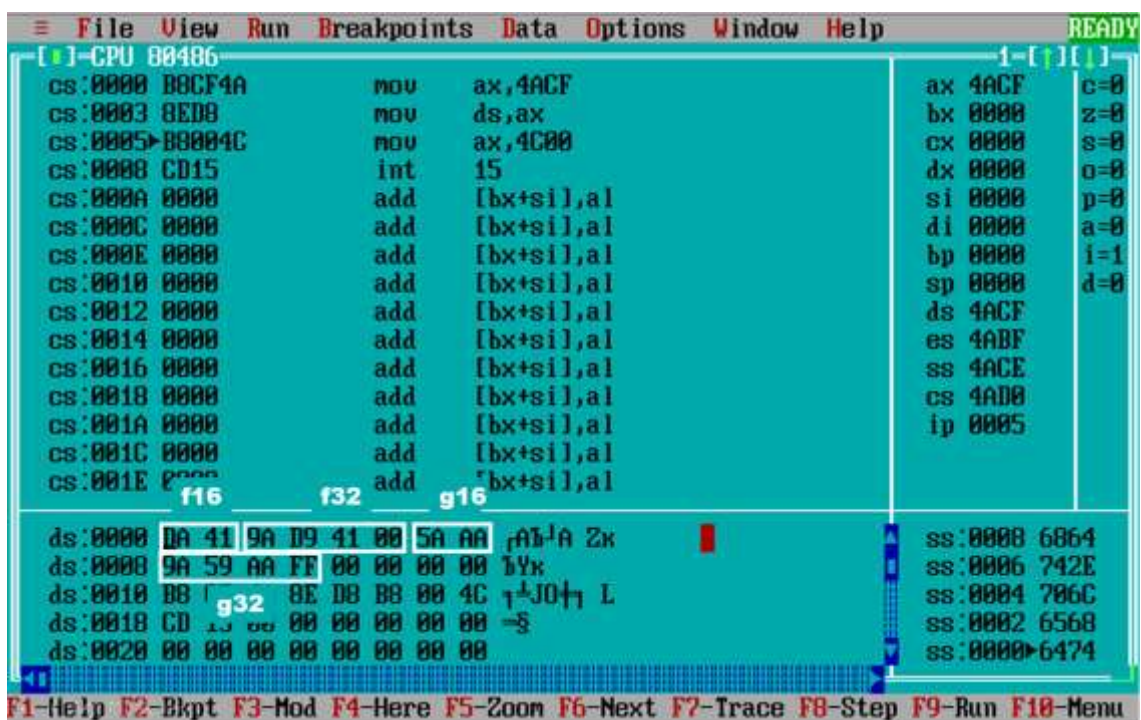
code segment
assume cs:code, ds:data, ss:nothing

start:    mov ax, data
          mov ds, ax

quit:     mov ax, 4c00h
          int 21

code ends
end start
```

Представление в оперативной памяти:



12. Увеличила в оперативной памяти каждого процессора значение младших байтов чисел на 1001_2 .

Изменение значения f:

$$\begin{array}{r} 0100\ 0001\ 1101\ 1010_2 \\ + \quad \quad \quad 1001_2 \\ \hline 0100\ 0001\ 1110\ 0011_2 \end{array}$$

$$f = 41,Е3_{16} = 4 \cdot 16^1 + 1 \cdot 16^0 + 14 \cdot 16^{-1} + 3 \cdot 16^{-2} = 65,88672_{10}$$

$$\begin{array}{r} 0000\ 0000\ 0100\ 0001\ 1101\ 1001\ 1001\ 1010_2 \\ + \quad \quad \quad \quad \quad \quad \quad \quad 1001_2 \\ \hline 0000\ 0000\ 0100\ 0001\ 1101\ 1001\ 1010\ 0011_2 \end{array}$$

$$f = 41,D9A3_{16} = 4 \cdot 16^1 + 1 \cdot 16^0 + 13 \cdot 16^{-1} + 9 \cdot 16^{-2} + 10 \cdot 16^{-3} + 3 \cdot 16^{-4} = 65,85014_{10}$$

Изменение значения g:

$$\begin{array}{r} 1010\ 1010\ 0101\ 1010_2 \\ + \quad \quad \quad 1001_2 \\ \hline 1010\ 1010\ 0110\ 0011_2 \end{array} \xrightarrow{\text{инверсия}} \begin{array}{r} 0101\ 0101\ 1001\ 1100_2 \\ + \quad \quad \quad 1_2 \\ \hline 0101\ 0101\ 1001\ 1101_2 \end{array}$$

$$g = -55,9D_{16} = -(5 \cdot 16^1 + 5 \cdot 16^0 + 9 \cdot 16^{-1} + 13 \cdot 16^{-2}) = -85,61328_{10}$$

Представление результата в памяти: $1010\ 1010\ 0110\ 0011_2 = AA63_{16}$

$$\begin{array}{r} 1111\ 1111\ 1010\ 1010\ 0101\ 1001\ 1001\ 1010_2 \\ + \quad \quad \quad \quad \quad \quad \quad \quad 1001_2 \\ \hline 1111\ 1111\ 1010\ 1010\ 0101\ 1001\ 1010\ 0011_2 \end{array}$$

инверсия

$$\begin{array}{r} 0000\ 0000\ 0101\ 0101\ 1010\ 0110\ 0101\ 1100_2 \\ + \quad \quad \quad \quad \quad \quad \quad \quad 1_2 \\ \hline 0000\ 0000\ 0101\ 0101\ 1010\ 0110\ 0101\ 1101_2 \end{array}$$

$$g = -(55,A65D)_{16} = -(5 \cdot 16^1 + 5 \cdot 16^0 + 10 \cdot 16^{-1} + 6 \cdot 16^{-2} + 5 \cdot 16^{-3} + 13 \cdot 16^{-4}) = -85,64986_{10}$$

Представление результата в памяти: $1111\ 1111\ 1010\ 1010\ 0101\ 1001\ 1010\ 0011_2 = FFAA59A3_{16}$

Представление в памяти:

```
part1.asm – Блокнот
Файл Правка Формат Вид Справка
data segment
    f16 dw 41E3h
    f32 dd 0041D9A3h

    g16 dw 0AA63h
    g32 dd 0FFAA59A3h
data ends

code segment
```

```

File View Run Breakpoints Data Options Window Help
[ ]-CPU 80486- 1-[ ]1111-
cs:0000 B8CF4A mov ax,4ACF ax 4ACF c=0
cs:0003 8ED8 mov ds,ax bx 0000 z=0
cs:0005 B8004C mov ax,4C00 cx 0000 s=0
cs:0008 CD15 int 15 dx 0000 o=0
cs:000A 0000 add [bx+si],al si 0000 p=0
cs:000C 0000 add [bx+si],al di 0000 a=0
cs:000E 0000 add [bx+si],al bp 0000 i=1
cs:0010 0000 add [bx+si],al sp 0000 d=0
cs:0012 0000 add [bx+si],al ds 4ACF
cs:0014 0000 add [bx+si],al es 4ABF
cs:0016 0000 add [bx+si],al ss 4ACE
cs:0018 0000 add [bx+si],al cs 4AD0
cs:001A 0000 add [bx+si],al ip 0005
cs:001C 0000 add [bx+si],al
cs:001E 0000 add [bx+si],al

ds:0000 E3 41 A3 D9 41 00 63 AA yAr-1A cK
ds:0008 A3 59 AA FF 00 00 00 00 rYk
ds:0010 BB 0E D8 B8 00 4C 1 J0 L
ds:0018 CD 00 00 00 00 00 00 -S
ds:0020 00 00 00 00 00 00 00

ss:0008 6864
ss:0006 742E
ss:0004 706C
ss:0002 6568
ss:0000 6474

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

```

Вывод: получила опыт в переводе целых, дробных и смешанных чисел из десятичной системы счисления в двоичную и из двоичной системы в десятичную с использованием шестнадцатеричной системы счисления в качестве промежуточной; рассмотрела формат представления чисел с фиксированной точкой в цифровых процессорах.