Пензенский государственный университет
Кафедра «Вычислительная техника»

ОТЧЕТ

по лабораторной работе №8
По дисциплине: «Арифметические и логические основы вычислительной техники»
На тему: «Умножение в цифровых процессорах»

Выполнили:
студенты группы 20ВВ2
xxxxxxxxxxxxxxxxxx

Принял:
xxxxxxxxxxxxxxxx

Пенза, 2021

Лабораторное задание:

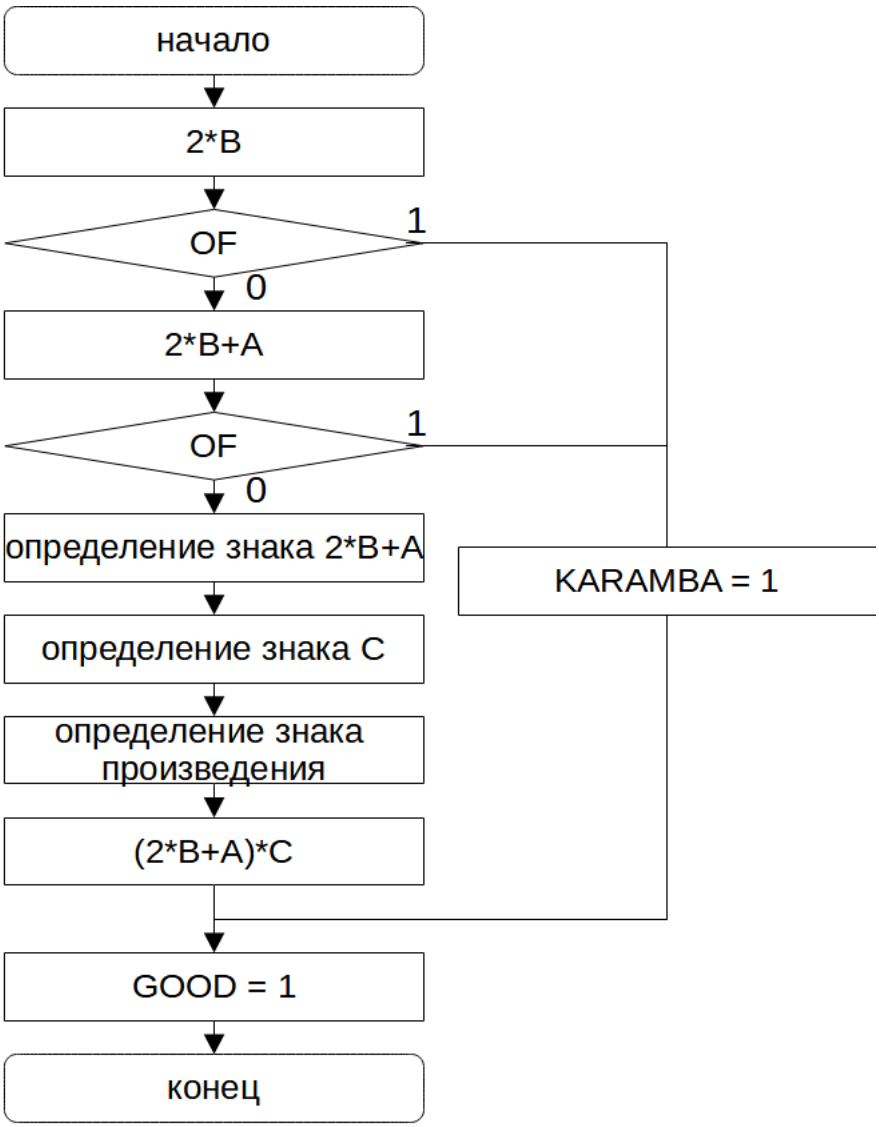
Написать на языке ассемблера программу вычисления выражения **Y=(2*B+A) * C**.

Процессор имеет разрядность – 16 бит. Исходные данные (значения переменных заданного выражения - целые, 16-битовые со знаком) располагаются в оперативной памяти, результат вычисления также поместить в оперативную память.

Заданные ограничения:

1. Умножение и деление на константы делать только с использованием операций сдвига, сложения и вычитания.
2. Умножение переменных при вычислении выражения сделать с использованием операций сложения, вычитания, сдвига, т.е. наложено ограничение − нельзя использовать команду умножения процессора i8086.
3. При вычислении выражения должны быть обработаны все возможные исключения.

Общий алгоритм вычисления выражения приведен на блок схеме:



Чтобы выполнить полное тестирование программы, нужны следующие расчёты:

1. Нормальное завершение, 2 случая (для Y>0 и Y<0), установка флага GOOD=1.
2. Переполнение при 2*B, 1 случай, аварийное завершение, установка флага KARAMBA=1.
3. Переполнение при 2*B+A, 1 случай, аварийное завершение, установка флага KARAMBA=1.

Всего потребуется четыре прогона программы.

Листинг программы:

```
data segment
A dw ?
B dw ?
C dw ?
Y1 dw ?
```

```
Y2 dw ?
GOOD db ?
KARAMBA db ?
mask dw 8000h ;маска для определения знака=1000 0000 0000 0000
data ends

code segment
assume cs: code, ds:data, ss: nothing
start:
mov ax, data
mov ds,ax
mov GOOD, 0 ;флаг нормального завершения
mov KARAMBA, 0 ;флаг аварийного завершения

;расчет суммы
mov ax, B ;ax=B
sal ax, 1 ;ax=2*B
jo METKA_KARAMBA ;аварийное завершение при переполнении 2*B
mov bx, A
add ax, bx   ;2*B+A
jo METKA_KARAMBA ;аварийное завершение при переполнении
2*B+A
mov bx, C

;проверка знаков
mov dx, mask
mov cx, dx
;определение знака 2*B+A
and dx, ax       ;выполняет логическое И между всеми битами
операндов; если 1 И 1 = 1, значит, 2*B+A - отрицательное
число
jz plus_SUM
not ax   ;инверсия и +1 (след. шаг), если число
отрицательное
inc ax
mov dx, 1

plus_SUM:
and cx, bx   ;определение знака C
jz plus_C
not bx       ;инверсия и +1 (след. шаг), если число
отрицательное
inc bx
mov cx, 1
```

```
plus_C:
xor dx, cx   ;1*1=0, 0*0=0, 1*0=1
push dx      ;сохранение знака произведения в стек


;выполнение умножения
xor dx, dx
mov cx, 15
;множимое 2*B+A в ax, множитель C в bx
mul_C:
rcr bx,1 ; начало цикла алгоритма умножения – сдвиг
регистра множителя
jnc a_1 ; проверка выдвинутого разряда - если бит равен 0,
то сдвиг
add dx, ax ; иначе сложение

a_1:
rcr dx, 1 ; сдвиг сумматора
loop mul_C ; уменьшение счетчика на 1

rcr bx,1 ;дополнительный сдвиг регистра множителя
rcr dx,1 ; дополнительный сдвиг регистра сумматора
rcr bx,1 ;дополнительный сдвиг регистра множителя

;извлечение знака
pop cx
test cl, 1   здесь не понял…
jz LABELRESULT   ;если знак 0, то выход
not dx
not bx
inc bx

LABELRESULT:
; Итог: в dx – старшая часть, в bx – младшая часть
произведения
mov Y1, dx ; загрузка старшей части в память
mov Y2, bx ; загрузка младшей части в память

jmp METKA_GOOD; переход на завершение

METKA_KARAMBA:
mov KARAMBA, 1 ; Аварийное завершение

METKA_GOOD:
mov GOOD, 1 ; Нормальное завершение
```

```
quit:
mov ax,4c00h ;возврат в операционную систему
int 21;
code ends
end start
```

Прогоны программы:

1) Нормальное завершение, Y > 0.
   A = 9914_{10} = $26BA_{16}$ = 0010 0110 1011 1010_2
   B = 9311_{10} = $245F_{16}$ = 0010 0100 0101 1111_2
   C = 15425_{10} = $3C41_{16}$ = 0011 1100 0100 0001_2
   Y1=$1A3C_{16}$
   Y2=$6D78_{16}$

2*B

ax	0	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1
sal ax, 1	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1	0

2*B+A

ax	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1	0
bx	0	0	1	0	0	1	1	0	1	0	1	1	1	0	1	0
ax+bx	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0

2*B+A – положительное число, C – положительное число

dx = 0; cx = 0; знак произведения = 0

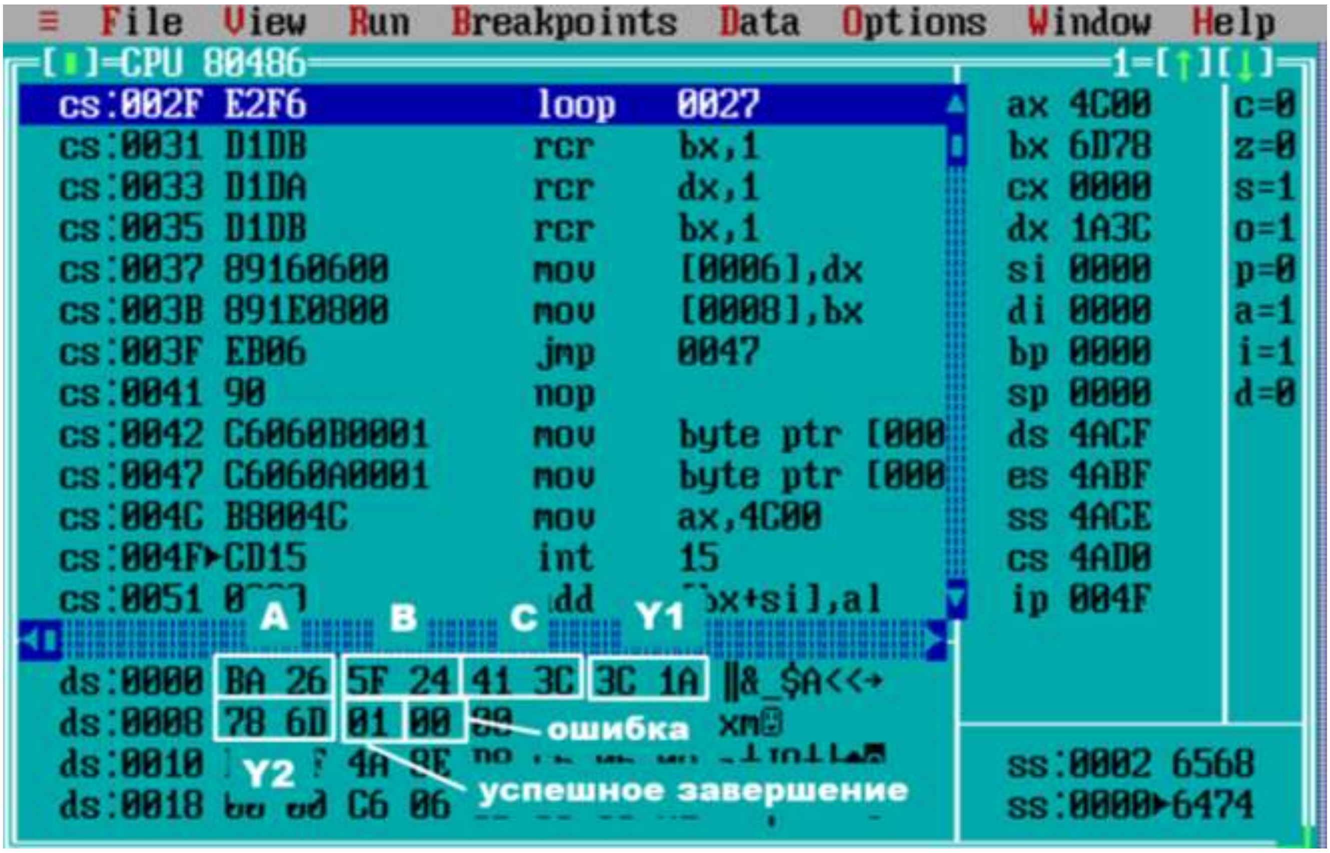
(2*B+A)*C

CF	dx																bx															счетчик			
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0	1	1	1	1	1
	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0	1				
	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0	1	1	1	0
	0	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1	1	0	1
	0	0	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	1	1	0	0
1	0	0	0	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	1	1
1	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	1	0	1	0
1	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	1
1	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1				
1	0	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0
0	0	0	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1

1	0	0	0	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	1	1	0
0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	1	0	1
0	0	1	1	1	0	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	1				
1	0	0	1	1	1	0	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1	0	1	0	0
1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	1				
1	0	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1	0	0	1	1
0	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	1				
0	0	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1	0	0	1	0
1	1	1	0	1	0	0	0	1	1	1	1	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	1				
1	0	1	1	0	1	0	0	0	1	1	1	1	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	1
1	0	0	1	1	0	1	0	0	0	1	1	1	1	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	1	0	0	0	1	1	1	1	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	0	0	0				

$Y1=0001\ 1010\ 0011\ 1100_2=1A3C_{16}$

$Y2=0110\ 1101\ 0111\ 1000_2=6B78_{16}$



Проверка: $(2*9311 + 9914) * 15425 = 440167800_{10} = 1A3C6D78_{16}$

Результат верный.

2) Нормальное завершение, Y < 0.
   $A = 4276_{10} = 10B4_{16} = 0001\ 0000\ 1011\ 0100_2$
   $B = -4532_{10} = -11BA$
   $[B]_2 = EE46_{16} = 1110\ 1110\ 0100\ 0110_2$
   $C = 15551_{10} = 3CBF_{16} = 0011\ 1100\ 1011\ 1111_2$
   $Y1 = FB8D_{16} = 1111\ 1011\ 1000\ 1101_2$
   $Y2 = 02C0_{16} = 0000\ 0010\ 1100\ 0000_2$

2*B

ax	1	1	1	0	1	1	1	0	0	1	0	0	0	1	1	0
sal ax, 1	1	1	0	1	1	1	0	0	1	0	0	0	1	1	0	0

2*B+A

ax	1	1	0	1	1	1	0	0	1	0	0	0	1	1	0	0
bx	0	0	0	1	0	0	0	0	1	0	1	1	0	1	0	0
add ax, bx	1	1	1	0	1	1	0	1	0	1	0	0	0	0	0	0

2*B+A – отрицательное число, требуется инверсия, C – положительное число

dx = 1; cx = 0; знак произведения = 1

ax	1	1	1	0	1	1	0	1	0	1	0	0	0	0	0	0
not ax	0	0	0	1	0	0	1	0	1	0	1	1	1	1	1	1
inc ax	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0

(2*B+A)*C

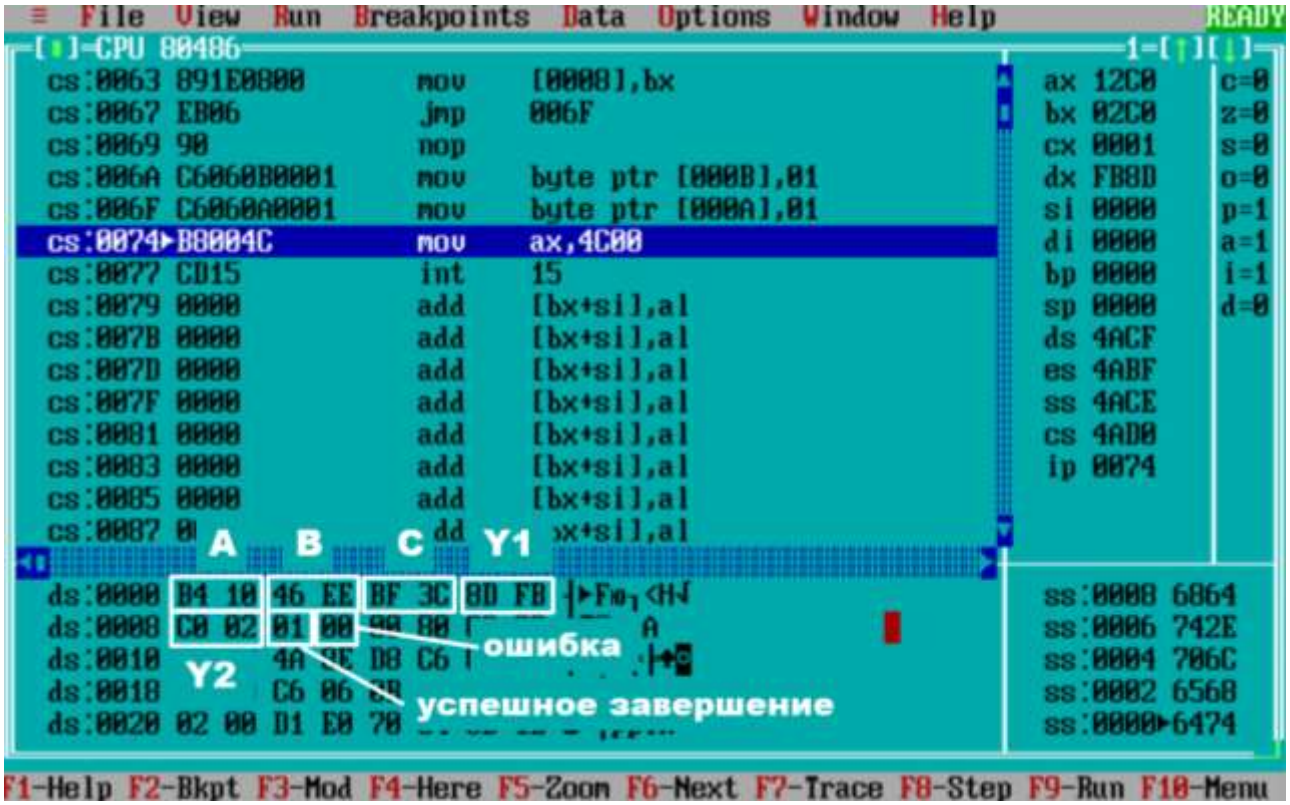
CF	dx																bx																счетчик			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1				
0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	1	1	0
0	0	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1					
0	0	0	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1
0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1				
0	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	0	0
0	0	0	1	0	0	0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1			
0	0	0	0	1	0	0	0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	1	1
0	0	0	1	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1			
0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	1	0	1	0
0	0	0	1	0	0	1	0	0	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1				
0	0	0	0	1	0	0	1	0	0	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1	0	0	1
1	0	0	0	0	1	0	0	1	0	0	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1	0	0	0
0	0	0	0	1	1	0	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1				
0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1
1	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1	0
0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	1
0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1				
1	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0
1	0	0	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1				
1	0	0	0	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1
1	0	0	1	0	0	0	0	1	1	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1	1				

1	0	0	0	1	0	0	0	0	1	1	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0
1	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1				
1	0	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0				

(2*B+A)*C =1111 1011 1000 1101 0000 0010 1100 0000_2= $FB8D02C0_{16}$

Y1=1111 1011 1000 1101_2=$FB8D_{16}$

Y2=0000 0010 1100 0000_2=$02C0_{16}$

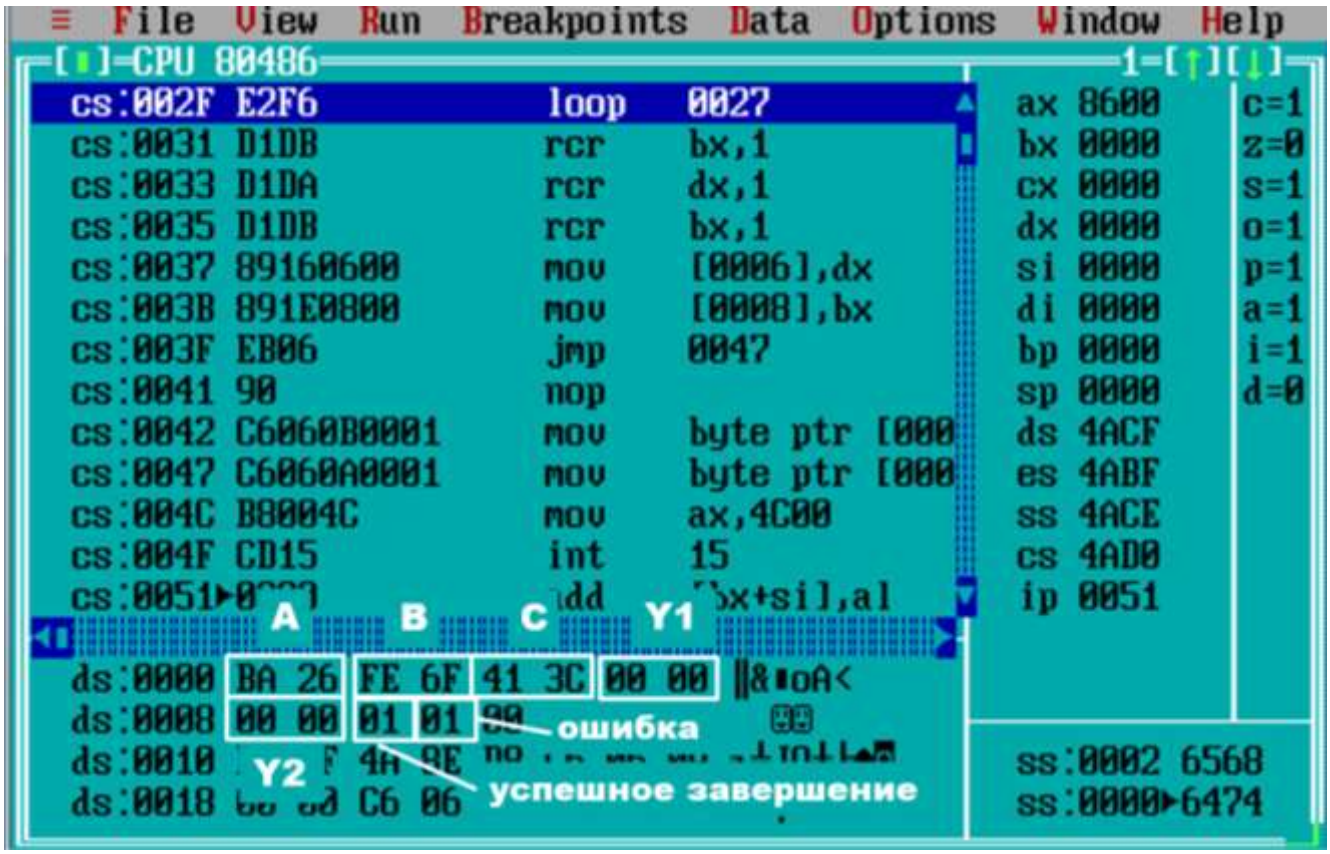


(2*(-4532) + 4276)*15551 = - 74458188_{10} = - $470244C_{16}$

3) Переполнение при 2*B, 1 случай, аварийное завершение, установка флага KARAMBA=1.
   A = 9914_{10} = $26BA_{16}$ = 0010 0110 1011 1010_2
   B = 28670_{10} = $6FFE_{16}$ = 0110 1111 1111 1110_2
   C = 15425_{10} = $3C41_{16}$ = 0011 1100 0100 0001_2
   Y1 = $3DD5_{16}$
   Y2 = $5C36_{16}$

2*B

ax	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0
sal ax, 1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	0



Возникло переполнение, установка флага KARAMBA=1.

4) Переполнение при 2*B+A, 1 случай, аварийное завершение, установка флага KARAMBA=1.
   A = $31282_{10} = 7A32_{16} = 0111\ 1010\ 0011\ 0010_2$
   B = $9311_{10} = 245F_{16} = 0010\ 0100\ 0101\ 1111_2$
   C = $15425_{10} = 3C41_{16} = 0011\ 1100\ 0100\ 0001_2$
   Y1 = $2DE1_{16}$
   Y2 = $BEF0_{16}$

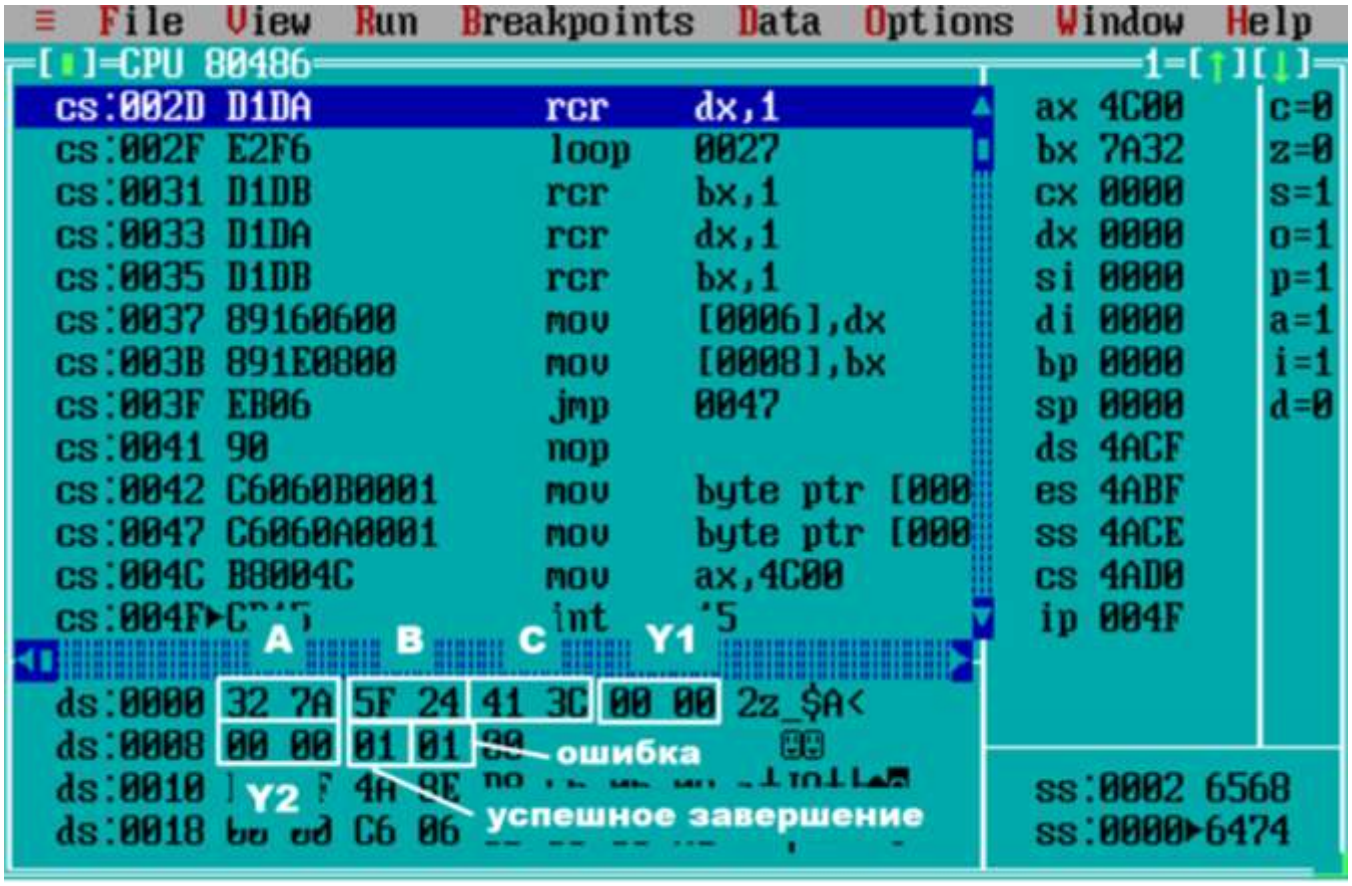
2*B

ax	0	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1
sal ax, 1	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1	0

2*B+A

ax	0	1	0	0	1	0	0	0	1	0	1	1	1	1	1	0
bx	0	1	1	1	1	0	1	0	0	0	1	1	0	0	1	0
ax+bx	1	1	0	0	0	0	1	0	1	1	1	1	0	0	0	0

Возникло переполнение, установка флага KARAMBA=1.



Вывод: создали программу на языке ассемблера для вычисления выражения Y=(2*B+A)*C. Умножение было реализовано через операции сдвига, сложения и вычитания. Возможные исключения обработаны.