

Защищено:
Большаков С.А.

Демонстрация ЛР:
Большаков С.А.

"__" _____ 2020 г.

"__" _____ 2020 г.

**Отчет по лабораторной работе №7 по курсу
Системное программирование**

"Ввод, вывод и перевод адреса "

(есть ли дополнительные требования - НЕТ)

11
(количество листов)
Вариант № 1

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5Ц-62**

Гусев С.Р.

(подпись)

"__" _____ 2020 г.

СОДЕРЖАНИЕ

1. Цель выполнения лабораторной работы № 7.....	3
2. Порядок и условия проведения работы № 7	3
3. Описание ошибок, возникших при отладке № 7.....	4
4. Блок-схема программы	4
5. Текст программы на языке Ассемблера	6
6. Результаты работы программы.....	12
7. Выводы по ЛР № 7	12

1. Цель выполнения лабораторной работы № 7

Разработать и отладить программу на языке Ассемблер для ввода с клавиатуры четырехразрядного числа (короткого адреса NEAR) в шестнадцатеричном представлении (доступные шестнадцатеричные цифры – 0123456789ABCDEF). Введенное значение переводиться в машинное представление в виде отдельного слова (2 байта - DW). Полученное значение выводится потом на экран также в шестнадцатеричном представлении, но заново переведенное из машинного формата. Кроме того, число выводится в десятичном формате (нужно выполнить программный перевод из одной системы счисления в другую).

2. Порядок и условия проведения работы № 7

Студенты разрабатывают работоспособную программу на языке Ассемблер по заданию ЛР, выполняют следующие действия (порядок выполнения работы):

- Знакомятся и осмысливают задание на ЛР.
- Разрабатывают алгоритм реализации задачи (блок-схема программы можно оформить в MS VISIO, MS WORD или на листе бумаги).
- Выполняют написание текста программы на языке Ассемблер и вводят его в отдельном текстовом редакторе (можно использовать текстовый редактор ASM_ED.EXE – есть на сайте) или в интегрированной оболочке (например, в QC).
- Выполняют отладку программы в отладчике (TD.EXE), демонстрируют преподавателю умение работать в отладчике, выполняя различные действия (выполнение по шагам, просмотр данных и т.д.).
- Формируют исполнимый модуль программы заданного типа (COM или EXE см. задание).
- Демонстрируют преподавателю работоспособную программу.
- По требованию преподавателя, если нужно, вносят изменения в программу и демонстрируют знание действий необходимых для создания исполнимого модуля (это предварительная сдача ЛР).
- Оформляют отчет по данной лабораторной работе в соответствии с требованиями приведенными ниже и на основе шаблона отчетов по ЛР.
- На основе отчета по ЛР (распечатанного) выполняют защиту ЛР у преподавателя (ответы на контрольные вопросы), после чего в журнале отмечается: срок сдачи ЛР, срок защиты ЛР, оценка за защиту данной ЛР и выполнение дополнительных требований к ЛР. На защите задаются вопросы, перечисленные в разделе “Контрольные вопросы по каждой ЛР и общие вопросы”, а также вопросы по листингу программы (отметьте себе, не по тексту программы, по листингу).

Работа считается выполненной полностью и в срок, если студент полностью сдал и защитил отчет ЛР в срок. Если студент сделал работу с дополнительными требованиями, то это обязательно отмечается в журнале ЛР и учитывается в оценке при подведении итогов семестра по данной дисциплине и на экзамене. Если студент выполнил все ЛР с дополнительными требованиями и получил отметки не ниже “хорошо”, то на зачете он освобождается от решения задачи (задачи на зачете заключаются в написании процедуры на языке Ассемблер или командного файла) и может претендовать на получение автоматической оценки по курсовой работе - ОТЛИЧНО, при своевременной ее сдаче.

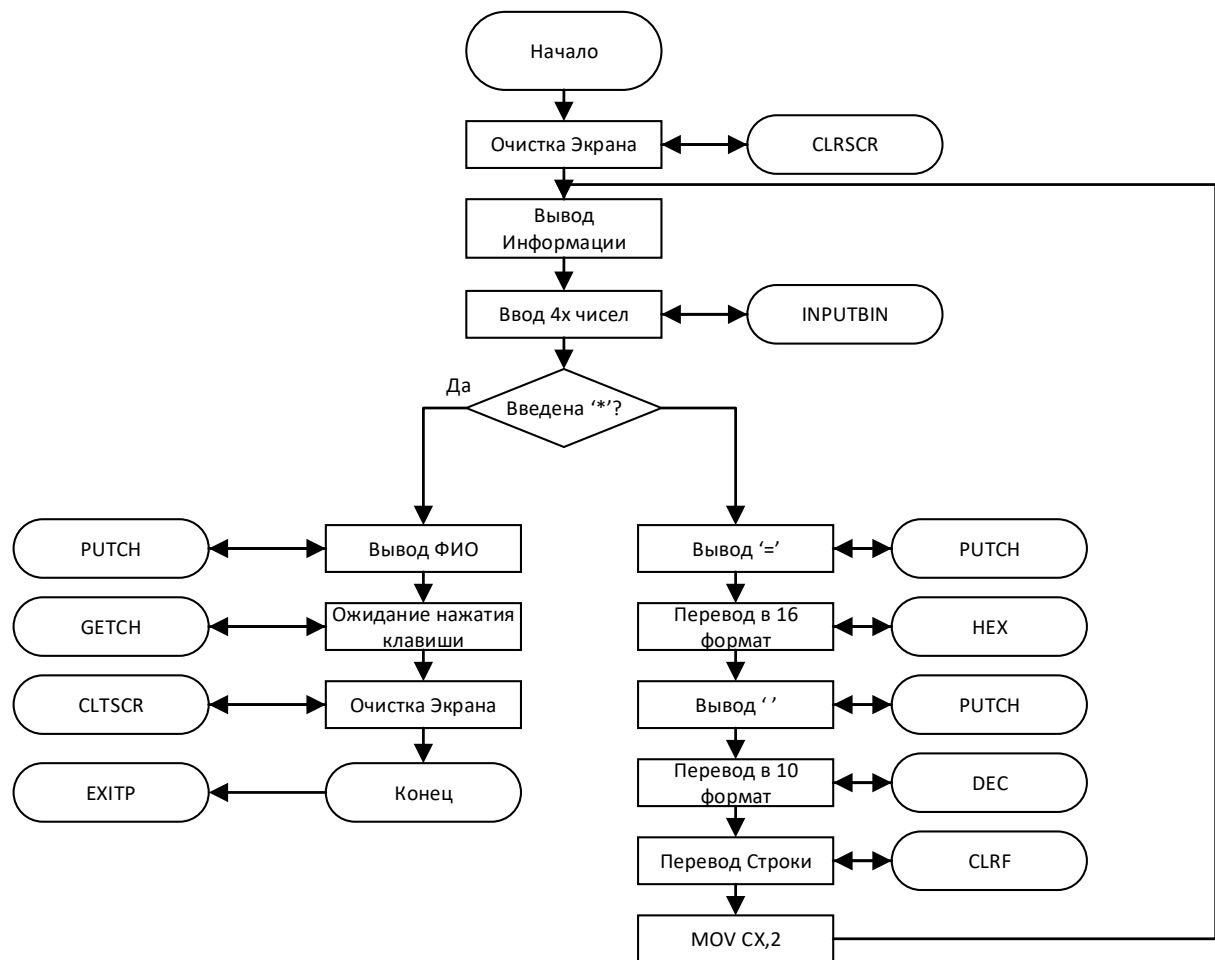
Если преподаватель обнаруживает (поверьте, сделать очень просто), что программа и отчет по ней сделаны несамостоятельно (проще - списаны), то он отмечает данный факт в журнале, а на зачете в этом случае задаются дополнительные вопросы по лабораторным работам, методическому пособию и материалам лекций. Кроме того, студент в этом случае не вправе рассчитывать на оценку по курсовой работе выше чем – удовлетворительно.

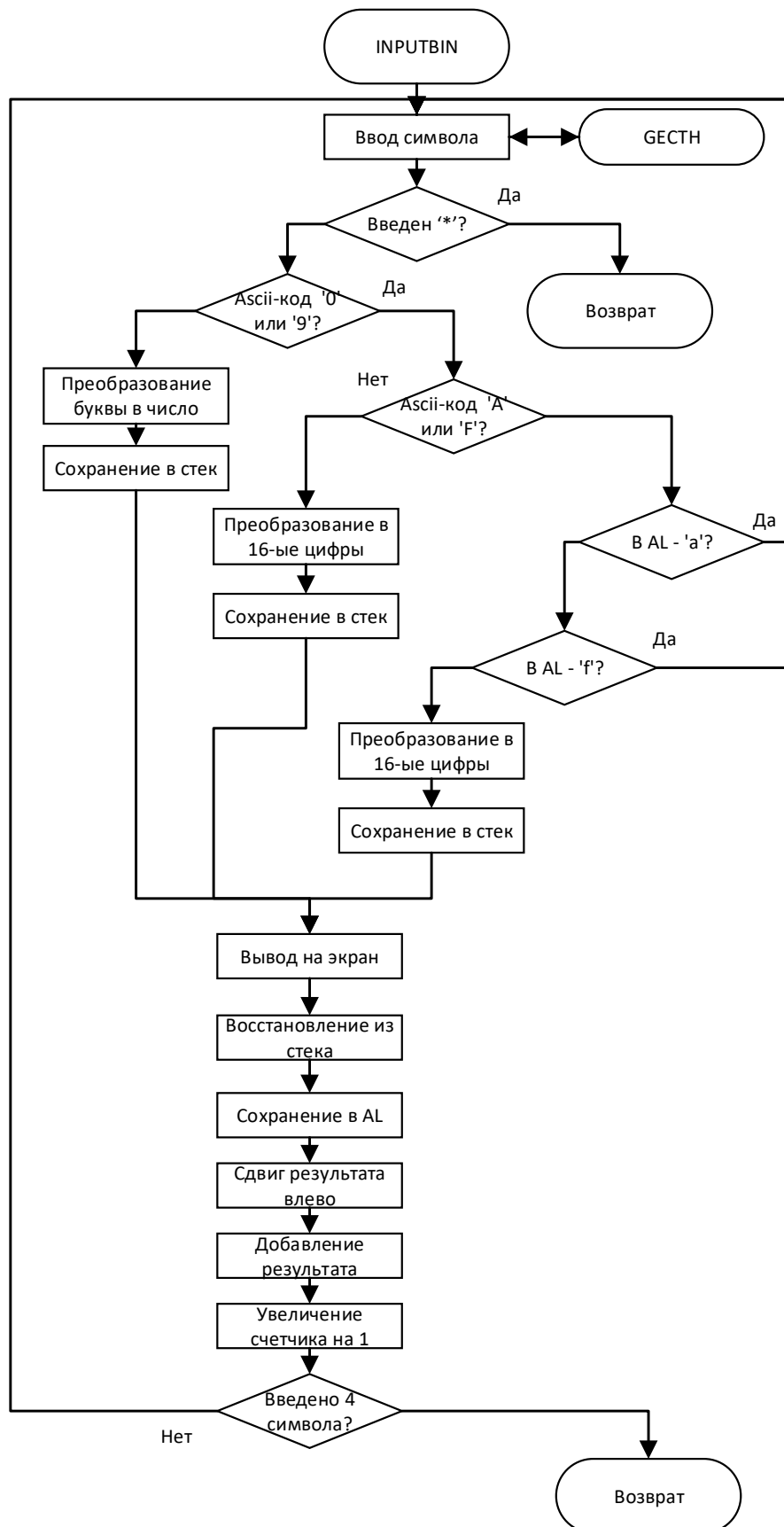
Для выполнения цикла лабораторных работ по курсу полезно познакомиться с указанными выше разделами методических указаний к ЛР, подготовленных преподавателем (отдельный документ – есть на сайте – оранжевая кнопка).

3. Описание ошибок, возникших при отладке № 7

№ п/п	Проявление ошибки	Причина ошибки	Способ устранения
1.	Ошибка компиляции	Неправильное использование оператора 'je'	Использовать оператор правильно
2.	Зависание программы, бесконечный вывод в консоль	Неправильно указано количество итераций цикла	Занести верное количество итераций цикла в регистр cx

4. Блок-схема программы





5. Текст программы на языке Ассемблера

```

3          assume cs:CODE
4 0000          CODE SEGMENT 'CODE'
5          ; -----
6 0000 8B A0 A1 AE E0 A0          E2+ LABTITLE DB 'Лабораторная работа #7$'
7          AE E0 AD A0 EF 20 E0+
8          A0 A1 AE E2 A0 20 23+
9          37 24
10 0017 8D AE A2 A0 EF 20          E1+ HELP1 DB 'Новая строка - enter$'
11          E2 E0 AE AA A0 20 2D+
12          20 65 6E 74 65 72 24
13 002C 82 EB E5 AE A4 20          A8+ HELP2 DB 'Выход из      программы - esc$'
14          A7 20 AF E0 AE A3 E0+
15          A0 AC AC EB 20 2D 20+
16          65 73 63 24
17 0045 82 A2 A5 A4 A8 E2          A5+ ENTRMSG DB 'Введите число: $'
18          20 E7 A8 E1 AB AE 3A+
19          20 24
20 0055 30 31 32 33 34 35 36+ HEX_TABLE DB '0123456789ABCDEF$'
21          37 38 39 41 42 43 44+
22          45 46 24
23 0066 41          CURNTSM BL DB 'A'
24 0067 1B          EXITSM BL DB 27 ; esc
25 0068 00          PRGRMFLAG DB 0
26 0069 2710 03E8 0064 000A + k    DW 10000,1000,100,10,1
27          0001
28          ; -----
29 0073          START:
30          ; Загрузка      сегментного регистра данных DS
31 0073 0E          push cs
32 0074 1F          pop ds
33          ; Очитка экрана
34 0075 E8 00B4          call CLRSCR
35          ; Вывод заголовка
36 0078 E8 00A6          call CLRF
37 007B BA 0000r          mov dx ,      offset LABTITLE
38 007E E8 00BA          call PUTSTR
39 0081 E8 009D          call CLRF
40          ; Вывод справки
41 0084 E8 009A          call CLRF
42 0087 BA 0017r          mov dx ,      offset HELP1
43 008A E8 00AE          call PUTSTR
44 008D E8 0091          call CLRF
45 0090 BA 002Cr          mov dx ,      offset HELP2
46 0093 E8 00A5          call PUTSTR
47 0096 E8 0088          call CLRF
48 0099 E8 0085          call CLRF
49 009C          MAINLP:
50          ; Вывод строки для ввода
51 009C BA 0045r          mov dx , offset ENTRMSG
52 009F E8 0099          call PUTSTR
53          ; Ввод двухбайтового слова в 16-м виде
54 00A2 E8 00A0          call INPUTBIN
55          ; Проверка на '*'
56 00A5 2E: 80 3E 0068r 01          cmp PRGRMFLAG , 1
57 00AB 74 46          je EXITP

```

```

58 00AD 53          push BX
59          ; Вывод '='
60 00AE B2 20          mov dl , '='
61 00B0 E8 0083          call PUTCH
62 00B3 B2 3D          mov dl , '='
63 00B5 E8 007E          call PUTCH
64 00B8 B2 20          mov dl , ''

```

65	00BA E8 0079	call PUTCH
66		; Вывод двубайтового слова в 16-м виде
67	00BD 8B D3	mov dx , bx
68	00BF 52	push dx
69	00C0 2E: 88 36 0066r	mov CURNTSMBL , dh
70	00C5 E8 0031	call HEX ; Первый байт
71	00C8 5A	pop dx
72	00C9 8A F2	mov dh , dl
73	00CB 2E: 88 36 0066r	mov CURNTSMBL , dh
74	00D0 E8 0026	call HEX ; Второй байт
75	00D3 B2 68	mov dl , 'h'
76	00D5 E8 005E	call PUTCH
77		; Вывод ' ()'
78	00D8 B2 20	mov dl , ''
79	00DA E8 0059	call PUTCH
80	00DD B2 28	mov dl , '('
81	00DF E8 0054	call PUTCH
82		; Вывод двухбайтового слова в 10-м виде
83	00E2 5B	pop BX
84	00E3 E8 00BD	call DEC
85		; Вывод ')'
86	00E6 B2 29	mov dl , ')'
87	00E8 E8 004B	call PUTCH
88		; Продолжение цикла
89	00EB E8 0033	call CLRF
90	00EE B9 0002	mov cx , 2
91	00F1 E2 A9	loop MAINLP
92	00F3	EXITP:
93		; Вывод из программы
94	00F3 B0 00	mov al , 0
95	00F5 B4 4C	mov ah , 4CH
96	00F7 CD 21	int 21H
97		; -----
98		; Процедуры
99		; -----
100		; Перевод символа в код
101		; -----
102	00F9	HEX PROC
103	00F9 2E: 8A 16 0066r	mov dl , CURNTSMBL
104	00FE BB 0055r	lea bx , HEX_TABLE ; Загрузка таблицы
105	0101 8A C2	mov al , dl
106	0103 D0 E8 D0 E8 D0 E8	D0+ shr al , 4 ; Отбрасывание младших разрядов
107	E8	
108	010B D7	xlat ; Преобразование в 16 систему
109	010C 8A D0	mov dl , al
110	010E E8 0025	call PUTCH
111	0111 2E: 8A 16 0066r	mov dl , CURNTSMBL
112	0116 8A C2	mov al , dl
113	0118 24 0F	and al , 00001111b ; Отбрасывание старших разрядов
114	011A D7	xlat ; Преобразование в 16 систему

115	011B 8A D0	mov dl , al
116	011D E8 0016	call PUTCH
117	0120 C3	ret ; Выход
118	0121	HEX ENDP
119		; -----
120		; Перевод на новую строку
121		; -----
122	0121	CLRF PROC
123	0121 B2 0A	mov dl , 10
124	0123 E8 0010	call PUTCH
125	0126 B2 0D	mov dl , 13
126	0128 E8 000B	call PUTCH


```

127 012B C3          ret ; Выход
128 012C          CLRFB ENDP
129                ; -----
130                ; Очистка экрана
131                ; -----
132 012C          CLRSCR PROC
133 012C E8 FFF2      call CLRFB
134 012F B4 00        mov ah , 0H
135 0131 B0 03        mov al , 3H
136 0133 CD 10        int 10H
137 0135 C3          ret ; Выход
138 0136          CLRSCR ENDP
139                ; -----
140                ; Вывод одного символа на экран (из dl)
141                ; -----
142 0136          PUTCH PROC
143 0136 B4 02        mov ah , 02H
144 0138 CD 21        int 21H
145 013A C3          ret
146 013B          PUTCH ENDP
147                ; -----
148                ; Вывод строки (из dx)
149                ; -----
150 013B          PUTSTR PROC
151 013B B4 09        mov ah , 09H
152 013D CD 21        int 21H
153 013F C3          ret
154 0140          PUTSTR ENDP
155                ; -----
156                ; Ввод символа с клавиатуры (в al)
157                ; -----
158 0140          GETCH PROC
159 0140 B4 08        mov ah , 08H
160 0142 CD 21        int 21H
161 0144 C3          ret
162 0145          GETCH ENDP
163                ; -----
164                ; Ввод слова и перевод адреса в двоичное число
165                ; -----
166 0145          INPUTBIN PROC
167 0145 BD 0000      mov bp , 0 ; Счетчик введенных символов
168 0148 BB 0000      mov bx , 0 ; Число
169 014B          INLOOP:
170                ; Ввод символа
171 014B E8 FFF2      call GETCH

```

```

172 014E 8A D0        mov dl , al
173 0150 EB 19 90      jmp CHECK
174 0153          SHIFT:
175                ; Если символ правильный, то он выводится
176 0153 E8 FFE0      call PUTCH
177                ; Восстановление из стека введенной цифры
178 0156 58          pop ax
179                ; Занесение 0 в ah, в al цифра
180 0157 B4 00        mov ah , 0
181                ; Сдвиг предыдущего результата на 4 влево
182 0159 D1 E3        shl bx , 1
183 015B D1 E3        shl bx , 1
184 015D D1 E3        shl bx , 1
185 015F D1 E3        shl bx , 1
186                ; Добавление введенной цифры к предыдущему результату
187 0161 03 D8        add bx , ax
188 0163 45          inc bp ; Счетчик ввода

```

```

189                                ; Если введены 4   цифры, то выходим
190 0164 83 FD 04                cmp  bp    , 4
191 0167 74 39                   je  EXIT
192 0169 EB E0                    jmp  INLOOP
193 016B                          CHECK:
194 016B 2E: 3A 06 0067г         cmp  al    , EXITSMBL ; Если введен не EXITSMBL то
                                проверяются цифры и буквы
195 0170 75 09                   jne  NUMBER
196 0172 2E: C6 06 0068г 01      mov  PRGRMFLAG , 1 ; Иначе установка флага выхода из программы и
возврат из процедуры
197 0178 EB 28 90                jmp  EXIT
198 017B                          NUMBER:
199                                ; Проверка цифр
200                                ; Если ascii-код   введенного символа <'0' или '9'>, то проверка
больших букв
201 017B 3C 30                   cmp  al ,    '0'
202 017D 72 09                   jb  BIGCHR
203 017F 3C 39                   cmp  al ,    '9'
204 0181 77 05                   ja  BIGCHR
205                                ; Иначе вычитание символа '0' из   ascii-кода введенной цифры
(преобразование "буквы" в+
206                                число)
207 0183 2C 30                   sub  al ,    '0' ; al - число
208                                ; Сохранение числа в стеке
209 0185 50                       push ax
210                                ; Переход на накопление результата в bx
211 0186 EB CB                    jmp  SHIFT
212 0188                          BIGCHR:
213                                ; Если ascii-код введенного символа <'A' или >'F',   то проверка
маленьких букв
214 0188 3C 41                   cmp  al , 'A'
215 018A 72 09                   jb  SMALLCHR
216 018C 3C 46                   cmp  al , 'F'
217 018E 77 05                   ja  SMALLCHR
218                                ; Преобразование букв в 16-е цифры
219 0190 2C 37                   sub  al , 'A'-10 ; 'B'-'A'+10=11
220                                ; Сохранение числа   в стеке
221 0192 50                       push ax
222 0193 EB BE                    jmp  SHIFT
223 0195                          SMALLCHR:
224 0195 3C 61                   cmp  al , 'a'
225 0197 72 B2                   jb  INLOOP
226 0199 3C 66                   cmp  al , 'f'
227 019B 77 AE                   ja  INLOOP
228 019D 2C 57                   sub  al , 'a'-10 ; 'b'-'a'+10=11

```

Turbo Assembler Version 3.1
lab.asm

04/19/17 09:50:21

Page 5

```

229 019F 50                       push ax
230 01A0 EB B1                    jmp  SHIFT
231 01A2                          EXIT:
232 01A2 C3                      ret
233 01A3                          INPUTBIN ENDP
234                                ; -----
235                                ; Перевод из 16-ой   в 10-ую систему
236                                ; -----
237 01A3                          DEC PROC

```

Warning lab.asm(222) Reserved word used as symbol: DEC

```

238 01A3 8B C3                   mov  ax    , bx ; Число
239 01A5 BF 0000                 mov  di    , 0 ; Счетчик цикла
240 01A8                          LOOP:

```

Warning lab.asm(225) Reserved word used as symbol: LOOP

```

241 01A8 BA 0000                 mov  dx    , 0 ; Старший байт делимого
242 01AB 2E: 8B 9D      0069г     mov  bx    , k[di] ; Делитель
243                                ; Деление

```

```

244             ; dx:ax / bx = ax(dx)
245             ; ax - частное, dx      - остаток
246 01B0 F7 F3      div bx
247             ; Остаток в стек
248             ; Остаток считается новым числом
249 01B2 52          push dx
250             ; Вывод результата деления
251 01B3 05 0030     add ax , '0'
252 01B6 8A D0       mov dl , al
253 01B8 E8 FF7B     call PUTCH
254             ; В ax (младший байт делимого) помещается остаток
255 01BB 58          pop ax
256 01BC 47          inc di
257 01BD 47          inc di
258             ; 5 цифр - 10 байт
259 01BE 83 FF 0A     cmp di , 10
260 01C1 72 E5       jb LOOP
261 01C3 C3          ret
262 01C4             DEC ENDP
263
264             END START

```

Warning lab.asm(249) Open segment: CODE

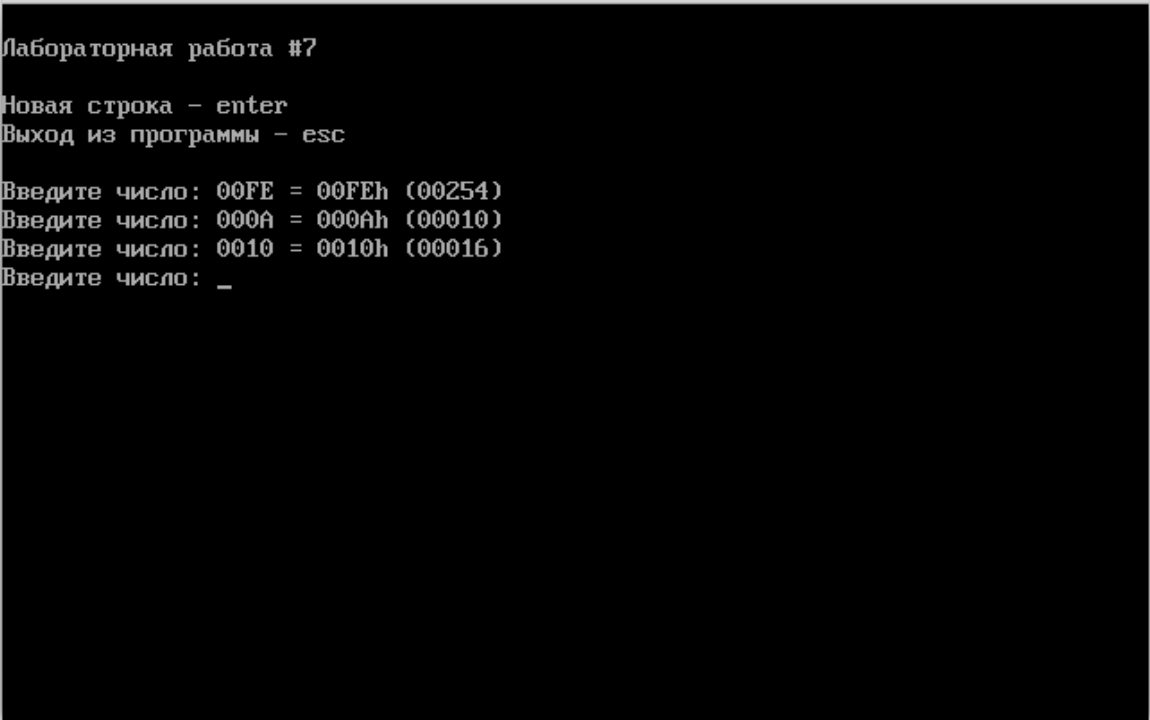
Turbo Assembler Version 3.1 04/19/17 09:50:21 Page 6
Symbol Table

Symbol Name	Type	Value	Cref	(defined at #)
??DATE	Text	"04/19/17"		
??FILENAME	Text	"lab"		
??TIME	Text	"09:50:21"		
??VERSION	Number	030A		
@CPU	Text	0101H		
@CURSEG	Text	CODE	#4	
@FILENAME	Text	LAB		
@WORDSIZE	Text	2	#4	
BIGCHR	Near	CODE:0188	202 204 #212	
CHECK	Near	CODE:016B	173 #193	
CLRF	Near	CODE:0121	36 39 41 44 47 48 89 #122 133	
CLRSCR	Near	CODE:012C	34 #132	
CURNTSMBL	Byte	CODE:0066	#23 69 73 103 111	
DEC	Near	CODE:01A3	84 #237	
ENTRMSG	Byte	CODE:0045	#17 51	
EXIT	Near	CODE:01A2	191 197 #231	
EXITP	Near	CODE:00F3	57 #92	
EXITSMBL	Byte	CODE:0067	#24 194	
GETCH	Near	CODE:0140	#158 171	
HELP1	Byte	CODE:0017	#10 42	
HELP2	Byte	CODE:002C	#13 45	
HEX	Near	CODE:00F9	70 74 #102	
HEX_TABLE	Byte	CODE:0055	#20 104	
INLOOP	Near	CODE:014B	#169 192 225 227	
INPUTBIN	Near	CODE:0145	54 #166	
K	Word	CODE:0069	#26 242	
LABTITLE	Byte	CODE:0000	#6 37	
LOOP	Near	CODE:01A8	#240 260	
MAINLP	Near	CODE:009C	#49 91	
NUMBER	Near	CODE:017B	195 #198	
PRGRMFLAG	Byte	CODE:0068	#25 56 196	
PUTCH	Near	CODE:0136	61 63 65 76 79 81 87 110 116	124 126 #142
176 253				
PUTSTR	Near	CODE:013B	38 43 46 52 #150	
SHIFT	Near	CODE:0153	#174 211 222 230	
SMALLCHR	Near	CODE:0195	215 217 #223	
START	Near	CODE:0073	#29 264	

Groups & Segments Bit Size Align Combine Class Cref (defined at #)

CODE 16 01C4 Para none CODE 3 #4
Turbo Assembler Version 3.1 04/19/17 09:50:21

6. Результаты работы программы



```
Лабораторная работа #7  
Новая строка - enter  
Выход из программы - esc  
Введите число: 00FE = 00FEh (00254)  
Введите число: 000A = 000Ah (00010)  
Введите число: 0010 = 0010h (00016)  
Введите число: _
```

7. Выводы по ЛР № 7

В ходе этой лабораторной работы я научился переводить числа из одной системы счисления в другую в ассемблере, а также закрепил свои знания по работе с регистрами.