



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ, Информатика и системы управления

КАФЕДРА ИУ7, Программное обеспечение ЭВМ и информационные технологии

ЛАБОРАТОРНАЯ РАБОТА №2

ПО ДИСЦИПЛИНЕ

“Операционные системы”

Студент ИУ7-54Б
(Группа)

_____ **А.А.Андреев**
(Подпись, дата) (И.О.Фамилия)

Преподаватель

_____ **Н.Ю. Рязанова**
(Подпись, дата) (И.О.Фамилия)

2021 г.

Оглавление

| | |
|--|----------|
| Оглавление | 1 |
| Задание | 2 |
| Решение | 4 |
| Перевод из реального режима в защищенный и обратно | 4 |
| Перевод из защищенного режима в реальный и обратно | 8 |

Задание

Написать программу, переводящую компьютер в защищенный режим (32-разрядный режим работы компьютеров на базе процессоров Intel). Программа начинает работать в реальном режиме. Для перевода в защищенный режим выполняются необходимые действия. В защищенном режиме программа работает на нулевом уровне привилегий.

В защищенном режиме программа должна:

- определить объем доступной физической памяти;
- осуществить ввод с клавиатуры строки с выводом введенной строки на экран;
- получить информацию на экране от системного таймера или в виде мигающего курсора, или в виде количества тиков с момента запуска программы на выполнение, или в виде значения реального времени.

Затем программа корректно возвращается в реальный режим с соответствующими сообщениями.

Для реализации поставленной задачи необходимо:

Создать две системные таблицы – глобальную таблицу дескрипторов (GDT) для описания сегментов физической памяти, с которыми будет работать запущенная программа и таблицу дескрипторов прерываний (IDT), в которой заполняются дескрипторы прерываний, которые необходимы для выполнения поставленной задачи.

Заполнить дескрипторы в обоих таблицах необходимой информацией.

Заполнить селекторы значениями смещения к соответствующим дескрипторам сегментов.

Перевести компьютер из реального в защищенный режим, установив флаг ре в 1.

В защищенном режиме определить объем доступной физической памяти следующим образом – первый мегабайт пропустить; начиная со второго мегабайта сохранить байт или слово памяти, записать в этот байт или слово сигнатуру, прочитать сигнатуру и сравнить с сигнатурой в программе, если сигнатуры совпали, то это – память. Вывести на экран полученной количество байтов доступной памяти.

Для ввода строки с клавиатуры необходимо написать обработчик прерывания от клавиатуры. Доступ к обработчику осуществляется через

предварительно заполненный дескриптор прерывания от клавиатуры в IDT.

Для получения информации от системного таймера необходимо написать обработчик прерывания от системного таймера. Доступ к обработчику осуществляется через предварительно заполненный необходимой информацией дескриптор прерывания от таймера в IDT.

Если в таблице дескрипторов прерываний были пропущены первые 32 дескриптора (так сделано фирмой Microsoft), то необходимо перепрограммировать контроллер прерывания на новый базовый вектор.

При переходе в защищенный режим необходимо открыть линию A20.

Возвращение в реальный режим должно выполняться корректно с использованием 32-разрядных операндов.

При переходе из режима в режим выдавать соответствующие сообщения.

В защищенном режиме информация для вывода на экран записывается непосредственно в видеобуфер.

Для возвращения в реальный режим выполнить необходимые действия.

Литература

Рудаков П.И., Финогенов К.Г. Программирование на ASSEMBLER. – М.: Диалог МИФИ, 2001, с.640

Зубков С. В. Программирование Assembler. - М.: Мир, 2004, с.685"

Первая часть: перевод в защищенный режим и обратно с подсчетом памяти

Вторая часть: ввод с клавиатуры и вывод тиков

Можно сдавать по частям или вместе.

Решение

В решении представлены листинги перевода из реального режима в защищенный и обратно (Листинг 1) со скриншотом работы перевода (Рисунок 1), перевода из защищенного режима в реальный и обратно (Листинг 2) со скриншотом работы перевода (Рисунок 2).

Перевод из реального режима в защищенный и обратно

Листинг 1: Код перевода из реального режима в защищенный и обратно с выводом соответствующего сообщения

```
1. . 386P
2.
3. ; Структура дескриптора сегмента.
4. DESCRIPTOR STRUC
5.     LIMIT    DW    0
6.     BASE_L   DW    0
7.     BASE_M   DB    0
8.     ATTR_1   DB    0
9.     ATTR_2   DB    0
10.    BASE_H   DB    0
11. DESCRIPTOR ENDS
12.
13. ; Сегмент данных.
14. ; USE16 - использование 16-битных адресов по умолчанию.
15. DATA SEGMENT USE16
16.     GDT_NULL DESCRIPTOR<0,0,0,0,0,0>
17.     GDT_DATA DESCRIPTOR<DATA_SIZE-1,0,0,92H,0,0>
18.     GDT_CODE DESCRIPTOR<CODE_SIZE-1,0,0,98H,0,0>
19.     GDT_STACK DESCRIPTOR<255,0,0,92H,0,0>
20.     GDT_SCREEN DESCRIPTOR<4095,8000H,0BH,92H,0,0>
21.     GDT_DATA32 DESCRIPTOR<OFFFFFH,0,0,92H,11001111B,0>
22.     GDT_SIZE=$-GDT_NULL
23.
24.     PDESCRIPTOR DF    0
25.
26.     MSGRM DB    '!REAL MODE!'
27.     MSGRM_LEN=$-MSGRM
28.     MSGPM DB    '!PROTECTED MODE!'
29.     MSGPM_LEN=$-MSGPM
30.
31.     PARAM=1CH
32.     COL=30
33.     ROW=12
34.
35.     DATA_SIZE=$-GDT_NULL
36. DATA ENDS
37.
38. TEXT SEGMENT 'CODE' USE16
39. ASSUME CS:TEXT, DS:DATA
40. MAIN PROC
```

```

41.      MOV AX, DATA
42.      MOV DS, AX
43.      MOV AX, 0B800H
44.      MOV ES, AX
45.
46.      MOV DI, ROW * 160 + COL * 2
47.      MOV BX, OFFSET MSGRM
48.      MOV CX, MSGRM_LEN
49.      MOV AH, PARAM
50.
51.      PRINTRM:
52.          MOV AL, BYTE PTR [BX]
53.          INC BX
54.          STOSW
55.          LOOP    PRINTRM
56.
57.      XOR EAX, EAX
58.      MOV AX, DATA
59.      SHL EAX, 4
60.      MOV EBP, EAX
61.      MOV BX, OFFSET GDT_DATA
62.      MOV WORD PTR [BX].BASE_L, AX
63.      SHR EAX, 16
64.      MOV BYTE PTR [BX].BASE_M, AL
65.
66.      XOR EAX, EAX
67.      MOV AX, CS
68.      SHL EAX, 4
69.      MOV BX, OFFSET GDT_CODE
70.      MOV WORD PTR [BX].BASE_L, AX
71.      SHR EAX, 16
72.      MOV BYTE PTR [BX].BASE_M, AL
73.
74.      XOR EAX, EAX
75.      MOV AX, SS
76.      SHL EAX, 4
77.      MOV BX, OFFSET GDT_STACK
78.      MOV WORD PTR [BX].BASE_L, AX
79.      SHR EAX, 16
80.      MOV BYTE PTR [BX].BASE_M, AL
81.
82.      MOV DWORD PTR PDESCR + 2, EBP
83.      MOV WORD PTR PDESCR, GDT_SIZE - 1
84.      LGDT    PDESCR
85.
86.      CLI
87.
88.      MOV EAX, CR0
89.      OR EAX, 1
90.      MOV CR0, EAX
91.
92.      DB 0EAH
93.      DW OFFSET CONTINUE
94.      DW 16
95.

```

```

96.          CONTINUE:
97.          MOV AX, 8
98.          MOV DS, AX
99.
100.         MOV AX, 24
101.         MOV SS, AX
102.
103.         MOV AX, 32
104.         MOV ES, AX
105.
106.         MOV AX, 40
107.         MOV GS, AX
108.
109.         MOV DI, (ROW + 1) * 160 + COL * 2
110.         MOV BX, OFFSET MSGPM
111.         MOV CX, MSGPM_LEN
112.         MOV AH, PARAM
113.
114.         PRINTPM:
115.             MOV AL, BYTE PTR [BX]
116.             INC BX
117.             STOSW
118.             LOOP    PRINTPM
119.
120.
121.         MOV GDT_DATA.LIMIT, 0FFFFH
122.         MOV GDT_CODE.LIMIT, 0FFFFH
123.         MOV GDT_STACK.LIMIT, 0FFFFH
124.         MOV GDT_SCREEN.LIMIT, 0FFFFH
125.
126.         PUSH DS
127.         POP DS
128.         PUSH ES
129.         POP ES
130.         PUSH SS
131.         POP SS
132.
133.         DB 0EAH
134.         DW  OFFSET GO
135.         DW  16
136.
137.         GO:
138.             MOV EAX, CR0
139.             AND EAX, 0FFFFFFFEH
140.             MOV CR0, EAX
141.             DB 0EAH
142.             DW  OFFSET RETURN
143.             DW  TEXT
144.
145.         RETURN:
146.             MOV AX, DATA
147.             MOV DS, AX
148.             MOV AX, STK
149.             MOV SS, AX
150.

```

```

151.      STI
152.
153.      MOV DI, (ROW + 2) * 160 + COL * 2
154.      MOV BX, OFFSET MSGRM
155.      MOV CX, MSGRM_LEN
156.      MOV AH, PARAM
157.
158.      PRINTRMAG:
159.          MOV AL, BYTE PTR [BX]
160.          INC BX
161.          STOSW
162.          LOOP    PRINTRMAG
163.
164.          MOV AX, 4C00H
165.          INT 21H
166.
167.      MAIN      ENDP
168.      CODE_SIZE=$-MAIN
169.      TEXT      ENDS
170.
171.      STK SEGMENT STACK 'STACK'
172.      DB 256 DUP(' ')
173.      STK ENDS
174.      END MAIN

```

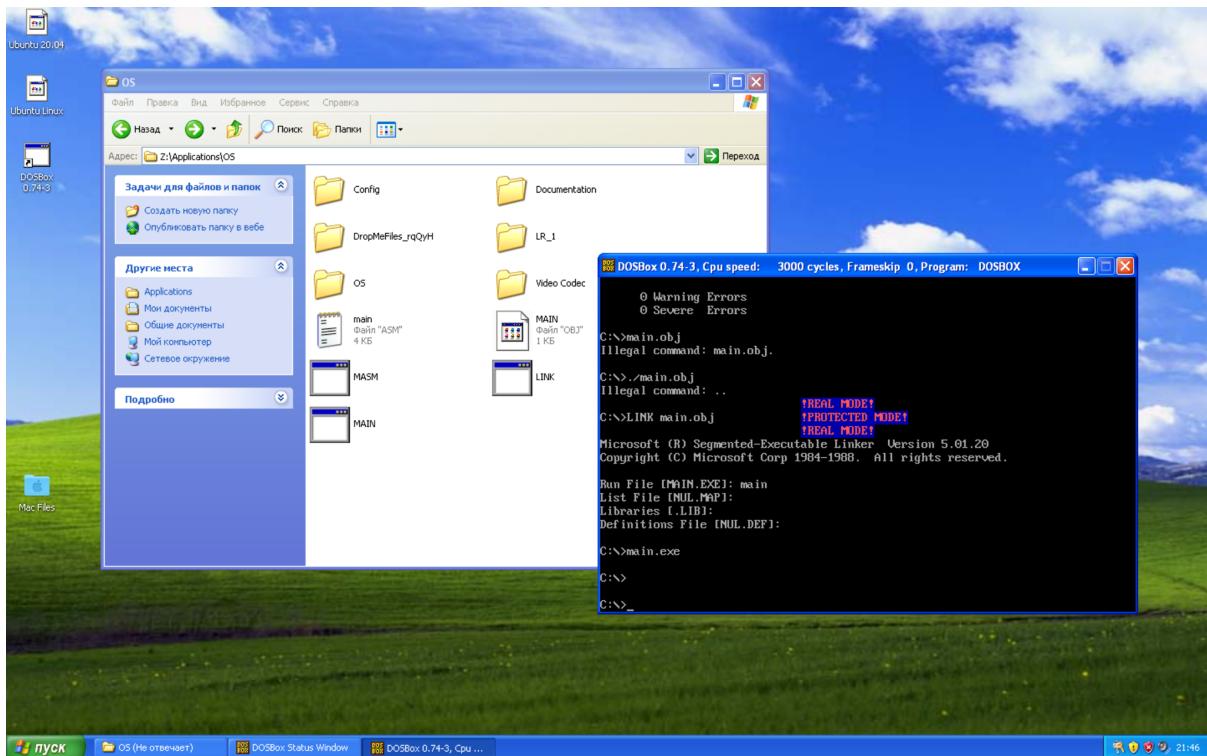


Рисунок 1: Работы программы при переводе из реального в защищенный и обратно

Перевод из защищенного режима в реальный и обратно

Листинг 2: Код перевода из реального режима в защищенный и обратно с выводом соответствующего сообщения

```
1. .386P
2.
3. ; Структура дескриптора сегмента.
4. DESCRIPTOR STRUC
5.     LIMIT    DW    0
6.     BASE_L   DW    0
7.     BASE_M   DB    0
8.     ATTR_1   DB    0
9.     ATTR_2   DB    0
10.    BASE_H   DB    0
11. DESCRIPTOR ENDS
12.
13. ; Сегмент данных.
14. ; USE16 - использование 16-битных адресов по умолчанию.
15. DATA SEGMENT USE16
16.     GDT_NULL DESCRIPTOR<0,0,0,0,0,0>
17.     GDT_DATA DESCRIPTOR<DATA_SIZE-1,0,0,92H,0,0>
18.     GDT_CODE DESCRIPTOR<CODE_SIZE-1,0,0,98H,0,0>
19.     GDT_STACK DESCRIPTOR<255,0,0,92H,0,0>
20.     GDT_SCREEN DESCRIPTOR<4095,8000H,0BH,92H,0,0>
21.     GDT_DATA32 DESCRIPTOR<OFFFFFH,0,0,92H,11001111B,0>
22.     GDT_SIZE=$-GDT_NULL
23.
24.     PDESCRIPTOR DF    0
25.
26.     MSGRM DB    '!REAL MODE!'
27.     MSGRM_LEN=$-MSGRM
28.     MSGPM DB    '!PROTECTED MODE!'
29.     MSGPM_LEN=$-MSGPM
30.
31.     PARAM=1CH
32.     COL=30
33.     ROW=12
34.
35.     DATA_SIZE=$-GDT_NULL
36. DATA ENDS
37.
38. TEXT SEGMENT 'CODE' USE16
39.     ASSUME CS:TEXT, DS:DATA
40.     MAIN PROC
41.         MOV AX, DATA
42.         MOV DS, AX
43.         MOV AX, 0B800H
44.         MOV ES, AX
45.
46.         MOV DI, ROW * 160 + COL * 2
47.         MOV BX, OFFSET MSGRM
48.         MOV CX, MSGRM_LEN
49.         MOV AH, PARAM
50.
```

```

51.      PRINTRM:
52.          MOV AL, BYTE PTR [BX]
53.          INC BX
54.          STOSW
55.          LOOP    PRINTRM
56.
57.          XOR EAX, EAX
58.          MOV AX, DATA
59.          SHL EAX, 4
60.          MOV EBP, EAX
61.          MOV BX, OFFSET GDT_DATA
62.          MOV WORD PTR [BX].BASE_L, AX
63.          SHR EAX, 16
64.          MOV BYTE PTR [BX].BASE_M, AL
65.
66.          XOR EAX, EAX
67.          MOV AX, CS
68.          SHL EAX, 4
69.          MOV BX, OFFSET GDT_CODE
70.          MOV WORD PTR [BX].BASE_L, AX
71.          SHR EAX, 16
72.          MOV BYTE PTR [BX].BASE_M, AL
73.
74.          XOR EAX, EAX
75.          MOV AX, SS
76.          SHL EAX, 4
77.          MOV BX, OFFSET GDT_STACK
78.          MOV WORD PTR [BX].BASE_L, AX
79.          SHR EAX, 16
80.          MOV BYTE PTR [BX].BASE_M, AL
81.
82.          MOV DWORD PTR PDESCR + 2, EBP
83.          MOV WORD PTR PDESCR, GDT_SIZE - 1
84.          LGDT    PDESCR
85.
86.          CLI
87.
88.          MOV EAX, CR0
89.          OR  EAX, 1
90.          MOV CR0, EAX
91.
92.          DB  OEAH
93.          DW  OFFSET CONTINUE
94.          DW  16
95.
96.          CONTINUE:
97.          MOV AX, 8
98.          MOV DS, AX
99.
100.         MOV AX, 24
101.         MOV SS, AX
102.
103.         MOV AX, 32
104.         MOV ES, AX
105.

```

```

106.      MOV AX, 40
107.      MOV GS, AX
108.
109.      MOV DI, (ROW + 1) * 160 + COL * 2
110.      MOV BX, OFFSET MSGPM
111.      MOV CX, MSGPM_LEN
112.      MOV AH, PARAM
113.
114.      PRINTPM:
115.          MOV AL, BYTE PTR [BX]
116.          INC BX
117.          STOSW
118.          LOOP    PRINTPM
119.
120.
121.      MOV GDT_DATA.LIMIT, 0FFFFH
122.      MOV GDT_CODE.LIMIT, 0FFFFH
123.      MOV GDT_STACK.LIMIT, 0FFFFH
124.      MOV GDT_SCREEN.LIMIT, 0FFFFH
125.
126.      PUSH DS
127.      POP  DS
128.      PUSH ES
129.      POP  ES
130.      PUSH SS
131.      POP  SS
132.
133.      DB  0EAH
134.      DW  OFFSET GO
135.      DW  16
136.
137.      GO:
138.          MOV EAX, CRO
139.          AND EAX, 0FFFFFFFEH
140.          MOV CRO, EAX
141.          DB  0EAH
142.          DW  OFFSET RETURN
143.          DW  TEXT
144.
145.      RETURN:
146.          MOV AX, DATA
147.          MOV DS, AX
148.          MOV AX, STK
149.          MOV SS, AX
150.
151.          STI
152.
153.      MOV DI, (ROW + 2) * 160 + COL * 2
154.      MOV BX, OFFSET MSGRM
155.      MOV CX, MSGRM_LEN
156.      MOV AH, PARAM
157.
158.      PRINTRMAG:
159.          MOV AL, BYTE PTR [BX]
160.          INC BX

```

```

161.           STOSW
162.           LOOP      PRINTRMAG
163.
164.           MOV AX, 4C00H
165.           INT 21H
166.
167.           MAIN     ENDP
168.           CODE_SIZE=$-MAIN
169.           TEXT     ENDS
170.
171.           STK SEGMENT STACK 'STACK'
172.           DB 256 DUP(' ')
173.           STK ENDS
174.           END MAIN

```

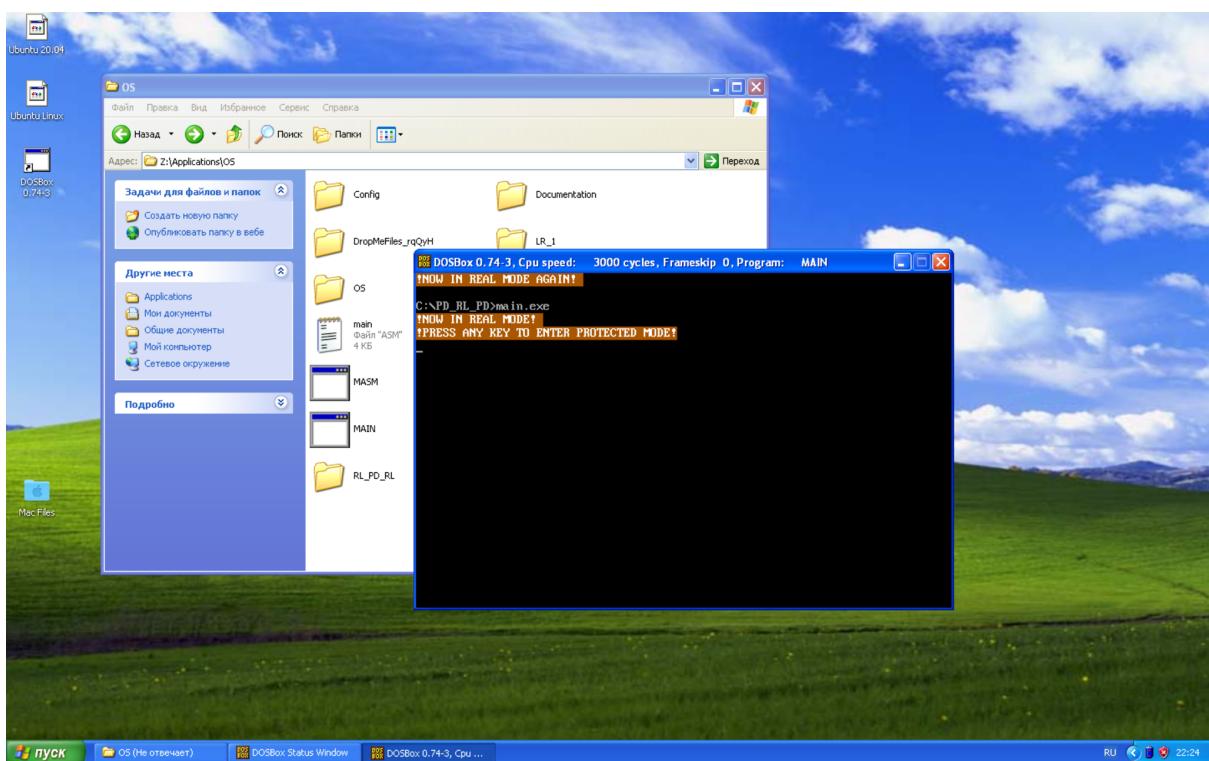


Рисунок 2: Работы программы при переводе из защищенного в реальный и обратно до клика на любую клавишу

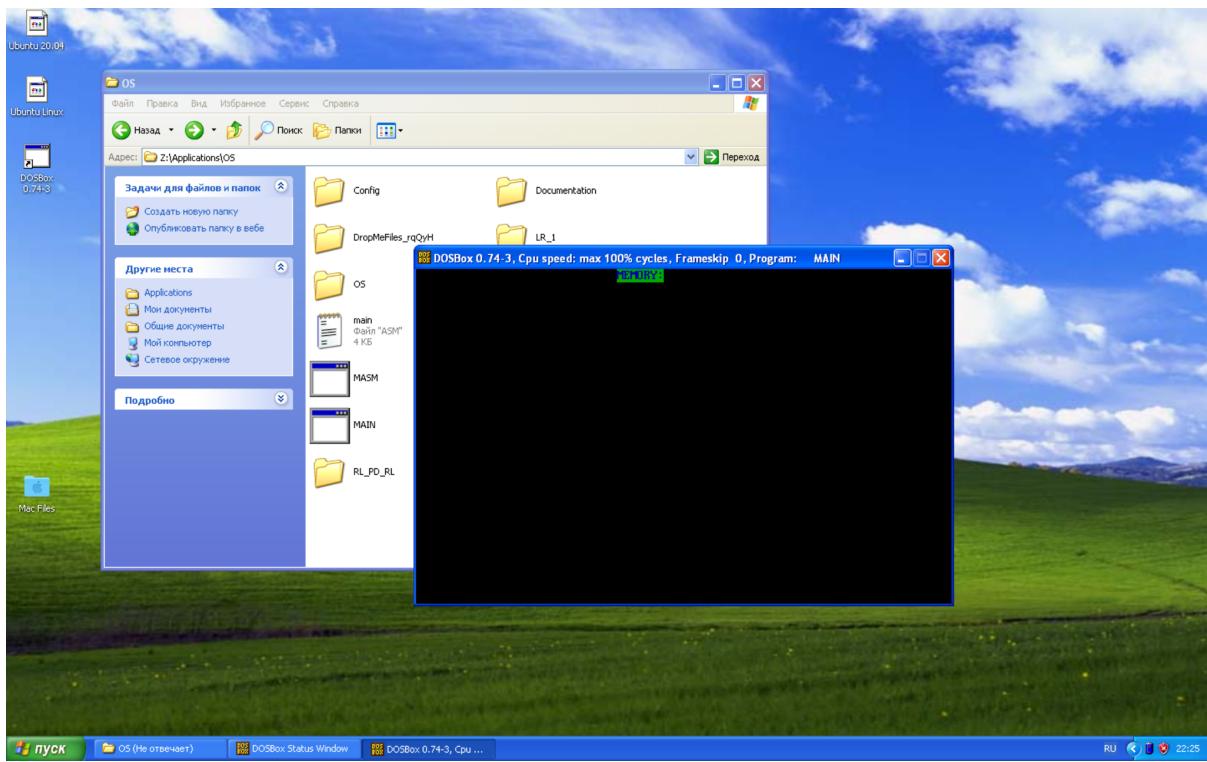


Рисунок 3: Работы программы при переводе из защищенного в реальный и обратно
после клика на любую клавишу во время работы

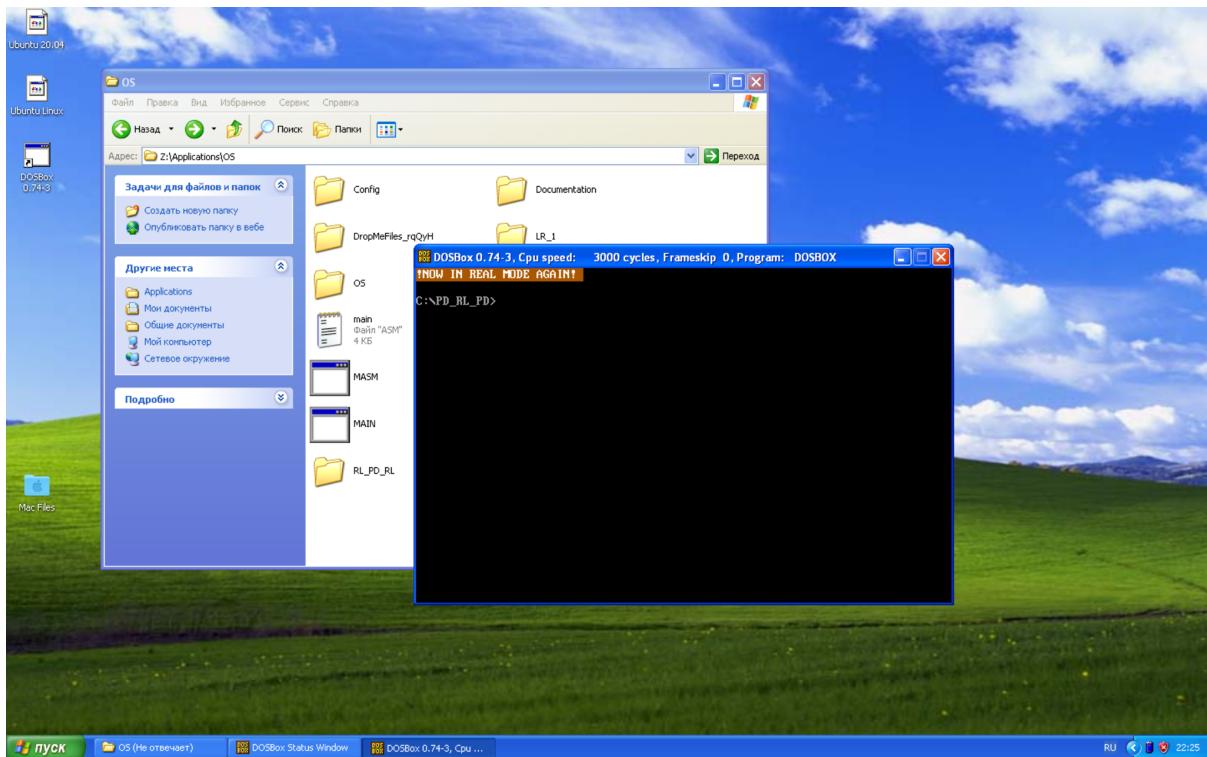


Рисунок 4: Работы программы при переводе из защищенного в реальный и обратно
после работы