

```

1 Program Tastint4;
2 (*****
3 (**          Tastint2 8.6.1998          **)
4 (**          Abfangen des Hardware-Interrupts 9          **)
5 (** - Demonstration der Folgen einer Nichtfreigabe des Interrupt- **)
6 (**   Controllers                                          **)
7 (**                                                    **)
8 (** - Ausgabe des Inhalts des Tastaturpuffers bzw. der niederen 7 Bits **)
9 (**                                                    **)
10 (** - Aufruf der alten Tastaturroutine nur dann, wenn die Taste Nr. 25 **)
11 (**   gedrückt wurde (Zeichen "p")                        **)
12 (**                                                    **)
13 (** TastInt4:                                             **)
14 (** - Restaurieren des alten Interrupt-Vektors nun INNERHALB der **)
15 (**   Schleife EXITPROC, also NACH Programmende.          **)
16 (** Dann wird das Programm durch Division durch Null zum Absturz ge- **)
17 (**   bracht.                                              **)
18 (** Beobachten sie die Folgen!!!!!! (Es gibt keine!!!!) **)
19 (**                                                    **)
20 (** Achtung!!! Um in Windows die Folgen zu sehen, müssen Sie das **)
21 (**   übersetzte Programm Tastint3.exe in einem DOS-Fenster **)
22 (**   starten.                                             **)
23 (** Windows95 ist nämlich so freundlich, diesen Fehler   **)
24 (**   in irgendeiner Weise mit dem Pascal-System abzufangen. **)
25 (**                                                    **)
26 (*****
27 {$M 1024,0,256} { Begrenzung des Speicherplatzbedarfs für das residente
28                   Programm }
29
30 Uses DOS, CRT;
31
32 Type T_Prozedur = Procedure; { Prozedurtyp, ist im Grunde genommen ein Zeiger, der
33                               auf den ersten Speicherplatz einer Prozedur im
34                               Codesegment zeigt }
35
36 VAR pAlter_KBD_Intr : Pointer;
37     Alter_KBD_Intr  : T_Prozedur ABSOLUTE pAlter_KBD_Intr;
38                               { Alte Keyboard-Interrupt-Prozedur im ROM-BIOS
39                               Diese Prozedur kann zunächst nicht über einen Namen aufgerufen werden,
40                               da sie keine Pascalprozedur ist. Sie beginnt am absoluten Spei-
41                               cherplatz pAlter_KBD_Intr. Dieser Speicherplatz muß bei der
42                               Programminitialisierung erst ermittelt werden. Damit läßt sich nun
43                               die ROM-Prozedur mit dem Namen Alter_KBD_Intr wie eine normale
44                               Pascal-Prozedur aufrufen }
45
46     pAlte_Exit_Prozedur : Pointer; { Beim Verlassen des Pascalprogramms, ob regulär oder infolge
47                                     eines Abbruchs durch einen Laufzeitfehler wird standard-
48                                     mäßig eine Prozedur aufgerufen, die am Speicherplatz ExitProc
49                                     beginnt. Der Zeiger ExitProc ist in Pascal definiert.
50                                     Sollen über den Standardausstieg hinaus weitere Aktionen
51                                     durchgeführt werden, wie zum Beispiel das Restaurieren
52                                     "verbogener" Interruptvektoren, so kann der Zeiger ExitProc
53                                     auf die eigene Prozedur gerichtet werden.}
54
55 Var Zei      : Char;
56     Absturz  : real;
57
58
59
60 Procedure KBDNeu; Interrupt;
61 Var Taste   : Byte;
62 Begin
63     Taste := Port[$60]; { Auslesen des Tastaturpuffers }

```

```

64      Begin
65          Gotoxy(10,15); Clreol;           { Zeile 15 ab Spalte 15 löschen }
66          Write(' Taste Nr.','Taste AND $7F); { Höchstwertiges Bit ausblenden }
67          if Taste < 128 then             { Oberstes Bit ist Flag, ob Taste }
68              Write(' gedrückt')         { drückt (Flag = 0) oder losgelassen }
69          else                             { ist (Flag = 1) }
70              Write(' losgelassen');
71
72          Gotoxy(10,16); Clreol;
73          Write(' Tastaturpufferinhalt: ',Taste);
74
75          Gotoxy(10,18);
76          Write(' Falls die Zeile "port[$20] := $20] auskommentiert ist,');
77          Gotoxy(10,19);
78          Write(' dann war das war die letzte Aktion in diesem Programm! ');
79          Gotoxy(10,20);
80          Write(' Der Interrupt-Controller wird nicht mehr freigegeben!');
81          Gotoxy(10,22);
82          Write(' Bei Druck auf die Taste "p" wird die alte BIOS-Routine aufgerufen,');
83          Gotoxy(10,23);
84          Write(' die das Zeichen dem Pascalprogramm übergibt.')
85      End;
86      { Hier wird nun nicht mehr die alte Tastaturroutine aufgerufen, die von
87        sich aus den Interruptcontroller wieder freigibt. Der Rechner reagiert daher nicht mehr und
88        muß durch Hardware Reset wieder gestartet werden }
89      if taste = 25 then { Taste mit Buchstaben p gedrückt!}
90      Begin
91          ASM
92          pushf
93          END;
94          alter_KBD_intr;
95      End
96      Else
97          Port[$20] := $20; { Interrupt Controller freigeben! Zum Test bitte
98                           durch Kommentarklammern unwirksam machen}
99
100 end;
101
102
103
104 Procedure Programm_Exit; far;
105 {+-----+
106 | Diese Prozedur wird beim Verlassen des Programms aufgerufen, also auch |
107 | beim Verlassen durch einen Laufzeitfehler oder durch <Ctrl><C>.         |
108 | Dabei wird die alte Interrupt-Prozedur wiederhergestellt und der Inter- |
109 | rupt-Controller wieder freigegeben, falls der Abbruch durch <Ctrl><C>   |
110 | erfolgte. In diesem Fall wird nämlich die alte Interrupt-Prozedur, die  |
111 | den Interrupt-Controller freigibt nicht mehr aufgerufen.               |
112 +-----+}
113 Begin
114     ExitProc := pAlte_Exit_Prozedur; { Vektor auf alte Standard-Exit-Prozedur
115                                     belegen }
116     SetIntVec($9,pAlter_KBD_Intr); { Interruptvektor Nr. 9 = Zeiger auf alte
117                                   Serviceroutine des Tastaturinterrupts wieder
118                                   herstellen }
119     Port[$20] := $20; { Interrupt Controller sicherheitshalber
120                       freigeben }
121 End;
122
123 (*-----*)
124
125 Begin (*Hauptprogramm*)
126     GetIntVec($9,pAlter_KBD_Intr); { Vektor auf die alte Tastaturinterrupt-Serviceroutine in der

```

```
127          absoluten Prozedurvariablen Alter_KBD_intr retten.
128          Damit kann die alte Prozedur mit diesem Bezeichner
129          aufgerufen werden }
130  Setintvec($9, @KBDNeu);      { Vektor Nr. 9 auf eigene Routine richten }
131
132  pAlte_Exit_Prozedur := ExitProc; { Exit-Handler installieren }
133  ExitProc             := @Programm_Exit;
134
135
136  ClrScr;
137  Textcolor(green);
138  Highvideo;
139  Writeln('*****      Test der neuen Tasturroutine      *****');
140  Normvideo;
141  Writeln;
142  Writeln('Druck auf beliebige Taste außer der Taste mit dem Buchstaben p ');
143  Writeln('läßt die Tastennummer auf dem Bildschirm erscheinen');
144  Writeln('In der nächsten Zeile ist der Inhalt des Tastaturpuffers zu sehen. ');
145  Writeln('Das oberste Bit ist offensichtlich 1, wenn die Taste losgelassen wird. ');
146  Writeln;
147  Writeln('Ende      :   Bitte p drücken');
148
149  Zei := Readkey;      { Diese Prozedur, die eine Tasteneingabe verarbeitet,
150                      muß nur einmal aufgerufen werden, da sie nur in Aktion tritt,
151                      wenn die neue Interruptprozedur die alte aufruft. Dies ist
152                      dann der Fall, wenn die Taste "p" gerückt wird, mit der das
153                      Programm beendet werden soll. }
154
155
156  Absturz := 0;
157  Absturz := 1/Absturz;
158  { Alter KBD-Interruptvektor wird von Pascal mit PROGRAMM_EXIT restauriert }
159  end.
160
161
```