

```

1  program HolIntVek;
2  {+-----+
3  | 1.10.95, 7.4.1998
4  | Ausgabe der Interruptvektoren auf dem Bildschirm
5  |
6  |
7  |
8  +-----+}
9  uses crt,dos;
10
11
12  {Im folgenden wird der Datentyp PTR_SegOf definiert, der aus den beiden Variablen
13  offset und segment besteht. Diese beiden Variablen werden im RAM in dieser Reihenfolge
14  abgelegt. In dieser Reihenfolge liegen im Speicher auch Offset und Segment von Adreß-
15  vektoren. Wenn p irgendeine Variable vom Typ Pointer ist, liegt hier auch zuerst der
16  Offsettingteil und dann der Segmentteil zu je 2 Byte auf aneinanderliegenden Speicherplätzen.
17
18  Pascal erlaubt nun durch Aufruf des Typs Ptr_SegOf mit dem Argument p vom Typ Pointer die
19  Typumwandlung in zwei Recordkomponenten des Typs WORD (= 2Byte). Diesen umgewandelten Typ
20  kann man dann z.B. zwei Variablen vom Typ WORD zuordnen, z.B.:
21
22  o := PTR_SegOf(p).Offset;
23  s := PTR_SegOf(p).Segment;}
24
25  Type Ptr_segof = record
26      offset,
27      segment    : Word;
28  end;
29
30
31
32  VAR p      : Pointer;    { In diese Variable wird der aktuelle Interrupt-
33                          vektor abgespeichert }
34
35      s,o    : Word;       { Diesen beiden Variablen wird der Zahlenwert der
36                          Adressteile Segment und Offset zugewiesen, die
37                          dann mit der Standard-Ausgabeprozedur Write am
38                          Bildschirm angezeigt werden können}
39
40      i      : Byte;       { Laufvariable zur Numerierung der Interrupts }
41      Hexzahl : String;     { Zur Darstellung der Registerinhalte als Hexzahl}
42
43
44  (*-----*)
45  Function Hexaus(Hex:Byte): String;
46  {+-----+
47  | 1.10.95, 7.4.1998
48  | Wandelt eine Zahl zwischen 0 und 255 in einen am Bildschirm anzeigbaren
49  | String um, der die Hexzahl darstellt
50  +-----+}
51  Var Zahl: Byte;
52      s   : String;
53      Procedure Halbbyteaus(Zahl:Byte);    {Zeichen für 4 Bit erzeugen }
54      Begin
55          If Zahl > 9 Then
56              Begin
57                  s := s + CHR(Zahl + 55);
58              End
59          Else
60              s := s + chr(Zahl + 48);
61      End;
62
63  Begin

```

```
64      s := '';
65      Zahl := Hex DIV 16;
66      Halbbyteaus(Zahl);
67      Hex := Hex MOD 16;
68      Halbbyteaus(Hex);
69      Hexaus := s;
70 End;
71
72
73
74 Begin
75   clrscr;
76   Writeln('Interrupt Nr      Adresse Hex      Adresse dezimal');
77   Writeln('Dez      Hex      Segment:Offset      Segment:Offset');
78   For i := 0 to 255 do
79     Begin
80       GetIntVec(i,p);           {i ist die Nummer des Interrupts, p der Adreß-
81                                vektor auf die zugehörige Interrupt-Service-
82                                Routine}
83       s := Ptr_segof(p).segment; { mit diesem Trick kann man die Typenumwandlung }
84       o := Ptr_segof(p).offset;  { vom 4-Byte-Zeiger zum 2-Wort-Record machen }
85
86       hexzahl := '';
87       hexzahl := hexaus(s div 256) + hexaus(s mod 256) + ':' +
88                 hexaus(o div 256) + hexaus(o mod 256);
89
90
91       Writeln(i:3,'      ',hexaus(i),'      ',hexzahl,'      ',s,':',o);
92       if (i+1) mod 22 = 0 then
93         Begin
94           readln;
95           clrscr;
96           Writeln('Interrupt Nr      Adresse Hex      Adresse dezimal');
97           Writeln('Dez      Hex      Segment:Offset      Segment:Offset');
98         End;
99       End;
100   readln;
101 End.
102
```