Harddisk macht Spaß

(Hart, schnell und sicher, c't 8 bis 10/86)

Ihr Projekt c't-HDC . . . hat mir viel Spaß gemacht und mein bescheidenes Wissen über Festplatten bedeutend erweitert. Leider mußte ich beim Testen der Hardware in der Software Fehler feststellen, die anschließend verbessert aufgelistet werden. Die vorgestellte Hardware versah ihren Dienst sofort fehlerfrei, womit man dem Projekt eine hohe Nachbausicherheit zusichern kann. Es ist also kein reines 'Könnerprojekt'.

Ihr vorgestellter Z80-Treiber läuft in der dargestellten Form nicht unter Turbo-Pascal, da

1.) in inlines eingeflochtene Variablenadressen innerhalb der innersten Prozedur definiert werden müssen. In Ihrer Version wurde für Buffer irgendein Wert hineinkompiliert.

2.) Konstantenwerte immer zwei Bytes an Platz einnehmen, womit die Definition mit 'hdcdat' leider nicht möglich ist. Das Programm stürzte mir an dieser Stelle immer ab. Nachdem ich die wahre Portadresse (bei mir 80h) eingesetzt hatte, ging's.

Thomas Lüth, Braunschweig

1.) In inlines eingeflochtene Variablenadressen brauchen nicht innerhalb der innersten Prozedur definiert zu werden – die Übergabe funktioniert auch bei globaler Deklaration einwandfrei. Allerdings liegt in diesem Programmteil wirklich ein Fehler vor. Die von Ihnen vorgeschlagene Parameterübergabe (call by reference) übergibt die Adresse, an der die Adresse des Buffers steht. Das entspricht dem inline (.. / $2A / buffer /. . .), in Z80-Mnemonics: LD HL,(buffer). Ändert man $2A auf $21, also den Code auf LD HL,buffer, paßt es auch wieder für die abgedruckte Version, da dort die Adresse des Buffers direkt übergeben wurde.

2.) Konstantenwerte werden in Turbo-Pascal tatsächlich als 2-Byte-Werte eingesetzt, allerdings nur bei den Versionen 1.0 und 2.0. Das verwendete Turbo 3.0 setzt nur ein einziges Byte ein, wenn die Konstante als Integer-Zahl im Wertebereich 0..255 definiert wird.

Herr Lüth hat die Software, die er Interessenten auf Wunsch auch auf eingesandter Diskette zukommen läßt, in diversen Punkten überarbeitet. Das genaue Disk-Format sollte vorher mit ihm abgesprochen werden: Thomas Lüth, Karlstr. 69, 3300 Braunschweig, 05 31/34 04 23.

```
-------------------------------------}
type bufarray = array[1..1024] of byte;

{-------------------------------------
    Lese Sektor-Puffer
-------------------------------------}
procedure Get_Data_From_HDC;
  procedure read(var buffer : buffarray);
  var i : integer;
  begin
    for i:=1 to (Sektor_Laenge div 128) do
    inline($2a/buffer/$01/$80/$80/$ed/$b2);
  end;

begin
  HDC_Wait;
  HDC_DRQ;
  read(buffer);
end;

{-------------------------------------
    Fuelle Sektor-Puffer
-------------------------------------}
procedure Send_Data_to_HDC;
  procedure send(var buffer : buffarray);
  var i : integer;
  begin
    for i:=1 to (Sektor_Laenge div 128) do
    inline($2a/buffer/$01/$80/$80/$ed/$b3);
  end;

begin
  HDC_DRQ;
  send(buffer);
  HDC_Wait;
end;
```

Aufgemotzt

(c't 6/87, S. 94)

Inzwischen konnte die Rechnerkarte für den C64 mit der 16-Bit-CPU 65SC816 auch in den neusten Revisionen der C64-Platine getestet werden, die auch an dem neuen 64poligen PLA erkennbar sind. In diesen Boards trägt die C64-CPU zwar die Bezeichnung '8500', sie ist aber identisch mit dem 6510 und kann ebenfalls von der 65SC816-CPU emuliert werden. Auch die neueren C64-Modelle lassen sich zum c't816 umbauen.

Die 8502-CPU, die im C128 eingesetzt wird, besitzt dagegen eine vom 6510 abweichende Pinbelegung. Die 65SC816-Karte läuft daher nicht im C128.

In der Projektbeschreibung wurde leider nicht erwähnt, daß die Karte erst bei Bestückung mit mindestens drei RAM-Bausteinen (IC25 bis IC27) lauffähig ist. Bei diesem Ausbau steht bereits die RAM-Disk mit einer Kapazität von 28 KByte zur Verfügung.

Wer noch die Datassette benutzt, wird feststellen, daß die Karte im 16-Bit-Betrieb den Kassettenmotor einschaltet. Dieser Effekt verschwindet, wenn man Pin 12 von IC9 (74LS395) mit Pin 24 des 6510 verbindet.

Im Schaltplan auf Seite 96 liegt der OE-Eingang des EPROMs auf Masse. In der Realität liegt dieser Eingang an Pin 5 von IC7, also am invertierten Takt des 65SC816. Das Listing zeigt eine Version der Start-Routine, die bereits Sprünge in den RAM-Disk-Treiber enthält. Es kann nicht unverändert abgetippt werden.

Noch eine Klärung zu den EPROMs: Lieferbar sind zur Zeit zwei Versionen (siehe Rubrik Software-Service). Das Startup-EPROM enthält nur eine Routine, die die 16-Bit-CPU aktiviert und das Kernal und das BASIC in das schnelle Simulations-RAM (IC25, 26) kopiert. Die zweite Version enthält zusätzlich den RAM-Disk-Treiber. Mit Erscheinen von c't 9/87 wird eine dritte EPROM-Version erhältlich sein, die einen Bildschirmeditor und einen makrofähigen Assembler für den vollen Befehlssatz der 16-Bit-CPU enthält. Mitgeliefert wird natürlich auch ein ausführliches Handbuch, das den Umgang mit Editor und Assembler beschreibt und den Befehlssatz der 65SC816-CPU erläutert.

Rasante Wurzel

(c't 6/87, S.160)

Im BASIC-Listing auf Seite 160 sind uns leider zwei Fehler unterlaufen. In Zeile 40 muß es heißen:

40 ZN = (X / YN - YN)

Und in Zeile 60 muß

60 YN = YN + ZN / 2

eingegeben werden. Damit wird die Wurzel auch in BASIC richtig berechnet.

Algebra – kein Problem

(c't 6/87, Seite 44)

Für meine Diplomarbeit arbeitete ich auf einem Apple II+-kompatiblen Rechner mit Z80-Karte und dem Programm muSIMP/muMATH. Dabei stieß ich auf einen Fehler in der Divisionsroutine von muMATH, der mich einige schlaflose Nächte kostete (man sucht Fehler ja immer erst bei sich). Als Korrektur sind ab Adresse 1B76 die im Listing angegebenen Speicherstellen zu ändern. Damit sollte es jedem möglich sein, eine eventuell fehlerhafte Version zu korrigieren.

Ansonsten kann ich die Vorzüge von muSIMP/muMATH nur bestätigen, denn muSIMP war eine sehr geeignete Programmiersprache für das Thema meiner Arbeit 'Bestimmung mehrfach rekursiver Pseudozufallszahlen-Generatoren mit maximaler kleinster Periode auf einem Mikrocomputer'.
Richard Hafke, Grünstadt

```
1B76  2B        DEC HL          ; jetzt "Übertragsbyte" des Dividenden
1B77  7E        LD  A,(HL)      ; testen, ob es
1B78  E6 01     AND 01          ; auf 1 gesetzt ist
1B7A  28 04     JR  Z,04        ; nein, dann vergleichen
1B7C  35        DEC (HL)        ; ja, dann löschen und da jetzt
1B7D  C3 97 1B  JP  1B97        ; Divisor(Dividend, subtrahieren
1B80  23        INC HL          ; Zeiger wieder auf MS-Byte des Divisors
1B81  1A        LD  A,(DE)      ; MS-Byte des Divisors mit
1B82  BE        CP  (HL)        ; MS-Byte des Dividenden vergleichen
1B83  DA 97 1B  JP  C,1B97      ; Divisor < Dividend, subtrahieren
1B86  C2 AC 1B  JP  NZ,1BAC     ; Divisor > Dividend, shiften
```