PEPS32

Ich habe noch einen alten EPROM-Simulator „PEPS“ von der Firma Conitec, der derzeit nur unter CP/M, DOS bzw. mit einer alten Windows Version verwendet werden kann. Angeschlossen wurde der „PEPS“ ursprünglich über eine parallele Schnittstelle.

Da ich ihn aber gern an meinem Arbeitsrechner unter Windows 7 64 bit bzw. Windows 8.1 64bit nutzen möchte, habe ich nach einer Möglichkeit gesucht, den Simulator neben der Übertragung über die parallel Schnittstelle LPT1, zusätzlich eine Möglichkeit zu schaffen, die serielle Schnittstelle, bzw. (über einen USB-UART-Wandler) über USB an den PC anzuschließen und die Daten so an den „PEPS“ zu übertragen.

Ursprünglich wurde zur Übertragung eine parallele Schnittstelle unter MS-DOS oder ganz früher unter CP/M eine PIO genutzt. Inzwischen ist die parallele Schnittstelle am PC leider auch weitestgehend ausgestorben.

Die Nutzung der parallelen Schnittstelle (Port 378h) ist aber in meinem Programm weiterhin möglich und auch um ein Vielfaches schneller (sofern die LPT noch im PC vorhanden ist). Notfalls gibt es auch kleine Zusatzkarten für den PC, die eine LPT-Schnittstelle bereitstellen.

Bei der seriellen Übertragung werden die Daten über einen kleinen „Umweg“, sprich eine kleine Zusatzschaltung zum „PEPS“ übertragen.

Entweder Variante 1: über einen vorhandenen echten COM-Port zu einem AVR (ATTiny2313 mit RS232-Schnittstellenbaustein). Der Rx-Eingang am AVR ist für TTL-Pegel ausgelegt, d.h. es muss ein kleiner RS232 Wandler vorgeschaltet werden.

Oder Variante 2: über einen USB-Anschluss mit USB-UART-Wandler zum AVR (ATTiny2313).
Ich verwende hier das kleine Modul UM 2102 vom ELV (Best.Nr.: 91859). Dieser bringt dann am Ausgang gleich die gewünschten TTL-Pegel für den Rx-Eingang des AVR.

Am AVR werden nur die ankommenden seriellen Daten sofort parallel am Port B ausgegeben.
Es werden nur 4 Datenleitungen gebraucht.
Dabei gibt es folgende Zuordnung:

PORTB.0 = M0	(bzw. D0 bei LPT1)
PORTB.1 = M1	(bzw. D1 bei LPT1)
PORTB.2 = DATA	(bzw. D2 bei LPT1)
PORTB.3 = CLOCK	(bzw. D3 bei LPT1)

Über diese Leitungen werden die Daten direkt zum „PEPS“ weiter gereicht.

Die Übertragungsgeschwindigkeit wurde auf 115200 Baud festgelegt (115200/8/n/1).

Leider ist selbst bei dieser Geschwindigkeit die Übertragung im Vergleich zur parallelen Übertragung über den LPT-Port sehr langsam.

1kB Datenübertragung dauert seriell ca. 15 sec, parallel < 1sec.
Bei paralleler Übertragung werden die Datenleitungen D0-D3 de LPT-Port genutzt.

Anleitung zum Datenübertragungs-Programm PEPS32.EXE

Das Programm ist in FreeBASIC geschrieben und arbeitet auf der Konsolenebene.

Verwendung bei paralleler Datenübertragung (LPT-Port):
Um die evtl. vorhandene parallel Schnittstelle zu verwenden, ist ein Treiber (inpout32.dll) eines Fremdanbieters zu nutzen, der den direkten Zugriff auf den LPT1-Port (378h) gestattet.
http://www.highrez.co.uk/Downloads/InpOut32/default.htm
Dieser Treiber muss sich im Programmverzeichnis vom EPSI32, bzw. im System32 Verzeichnis im Windows befinden.
Durch Verwendung des Fremd-Treibers muss „EPSI32.EXE“ evtl. mit Administratorrechten ausgeführt werden. Eine evtl. Abfrage ist mit ja zu beantworten.
Der Programmaufruf erfolgt z.B. in einem CMD-Fenster mit zwei Parametern:
Parameter 1: Dateiname.Endung
Parameter 2: P für Verwendung der parallelen Schnittstelle LPT1 oder
COM-Port-Nummer (von 1…20)

Beispiel*:* ***PEPS32 test.bin P***
(Datei test.bin soll parallel über LPT1 an EPSI übertragen werden)

Verwendung bei serieller Datenübertragung (COM-Port oder USB =virtueller COM-Port):

Bei einer echten COM-Schnittstelle ist kein zusätzlicher Treiber nötig. Bei einem USB-UART Wandler (der einen virtuellen COM-Port bereitstellt) ist der zum Wandler zugehörige Treiber im Vorfeld zu installieren und der vergebene virtuelle COM-Port aus der Systemsteuerung auszulesen.

Aufgerufen wird das Programm zur Datenübertragung dann mit:
Beispiel: ***PEPS32 test.bin 5***
(Datei test.bin soll seriell über COM5 an EPSI übertragen werden)

Empfehlenswert ist die Dateiendung ***.bin*** zu verwenden. Es sind auch andere Endungen möglich, allerdings müssen die Daten im Binärformat gespeichert sein.
Die Dateigröße ist nach der Größe des simulierten EPROM natürlich zu beachten.

Verwendung unter ASIDE 1.14

Da ich zeitweise gern das Programm ASIDE v1.14 von Tassilo Heeg nutze und aus dem Programm heraus ein direkter Aufruf eines EPROM-Simulators vorgesehen ist, habe auch diese Möglichkeit getestet. Nach entsprechender Eingabe der Aufrufparameter unter … EPROM-Emulator… Z80

Beispiel: ***peps32 *.bin p*** (bei Parallelport)

funktioniert das sehr gut. Man muss nicht jedes Mal die Programmierumgebung verlassen und die neu erstellte Binärdatei wird sofort nach dem erfolgreichen Übersetzungslauf zum „PEPS“ übertragen.

Infos:

```
'*****************************************************************
'*EPROM - Simulator PEPS
'* File: peps32.exe    Datum:19-02-17  Autor: Klaus Wilfling
'* Version: 0.3
'* EPSI-Interface für PC über parallele oder serielle Schnittstelle mit AVR
'* Bit 0 : M0  (Modus-Bit 0)
'* Bit 1 : M1  (Modus-Bit 1)
'* Bit 2 : Data
'* Bit 3 : Clock
'*****************************************************************
'* Modus       M1      M0      Bedeutung
'*       0       0       0       Adresse hochzählen
'*       1       0       1       Datenbyte einschieben
'*       2       1       0       Datenbyte übernehmen
'*       3       1       1       Simulationsmodus
'*****************************************************************
```