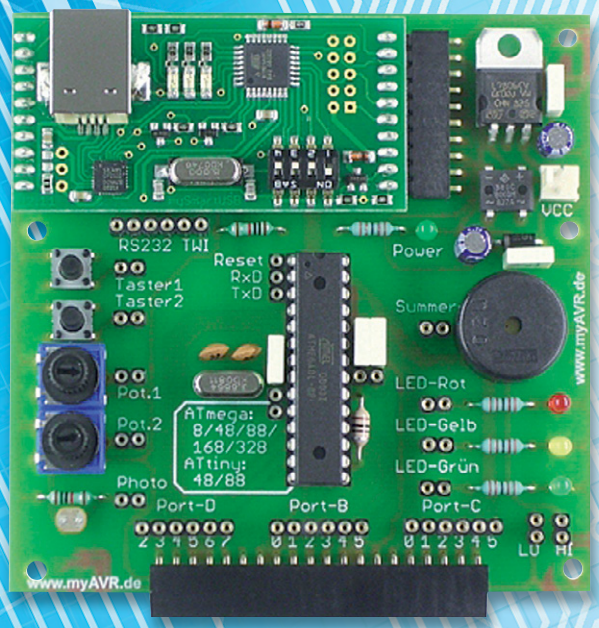
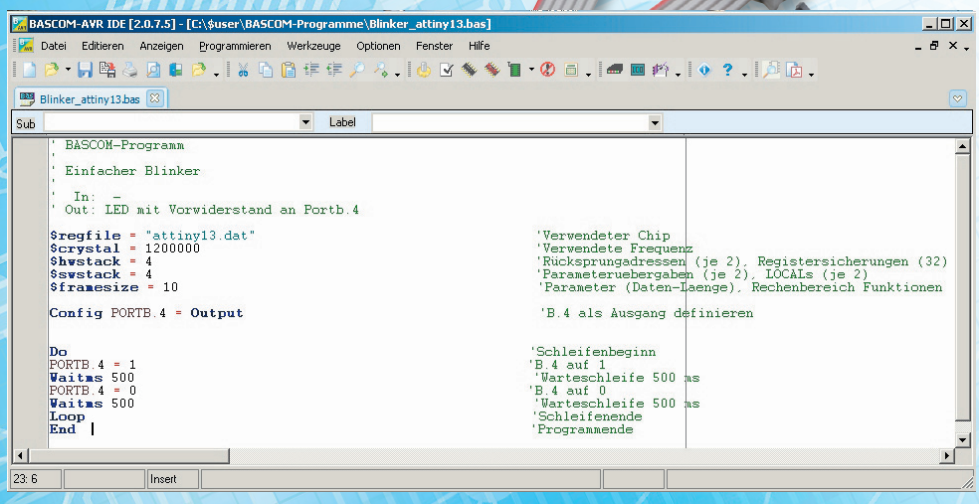
Mikrocontroller-Einstieg

Teil 7: Serielle (UART-)Datenübertragung





mit BASCOM-AVR

Die serielle UART-Schnittstelle hat eine lange Tradition als Möglichkeit für die robuste und „leitungssparende" Datenübertragung zwischen zwei elektronischen Geräten. Mit BASCOM können vom Mikrocontroller Daten zu einem anderen Mikrocontroller, zu einem PC oder zu anderen Geräten gesendet werden und es können Daten von einem Mikrocontroller, PC oder anderen Geräten empfangen werden (Bild 1). In diesem Teil der Artikelserie „Mikrocontroller-Einstieg mit BASCOM-AVR" werden die Grundlagen der seriellen Datenübertragung sowie das Senden von Daten zum PC bzw. an den FS20-Sender FS20 US dargestellt.

Grundlagen

Bei einer parallelen Datenübertragung werden mehrere Bits (zum Beispiel 8 Bit) parallel – also gleichzeitig und quasi nebeneinander – auf mehreren Leitungen von einer Hardware-Einheit zur anderen übertragen. Bei einer seriellen Datenübertragung werden die Bits auf einer Leitung nacheinander übertragen. Man benötigt für die reine Datenübertragung daher nur eine Leitung. Bei einer sogenannten synchronen seriellen Datenübertragung gibt es noch eine zweite Leitung, die ein Taktsignal zur Synchronisierung von Sender und Empfänger überträgt. Bei einer asynchronen seriellen Datenübertragung gibt es keine Taktleitung, was bedeutet, dass Sender und Empfänger sich ohne Taktsignal synchronisieren müssen. Sender und Empfänger müssen sich an gemeinsame Vereinbarungen in Bezug auf die Geschwindigkeit und Reihenfolge der Bitübertragung usw. halten, damit die Kommunikation funktionieren kann. In beiden Fällen (synchron und asynchron) benötigt man zusätzlich eine GND-Verbindung.

Während auch I²C, SPI, 1-Wire, USB usw. serielle Datenverbindungen realisieren, ist mit serieller Datenübertragung meistens die serielle UART-Verbindung gemeint. UART ist die Abkürzung von „Univeral Asynchronous Receiver/Transmitter" (= Universeller asynchroner Empfänger und Sender) und steht für Hardware-Einheiten in Mikrocontrollern oder PCs, welche die Protokollumsetzung zwischen internem Datenbus und seriellem Anschlusspin durchführen. Die meisten Atmel-Mikrocontroller haben eine eingebaute UART-Einheit (oder mehrere). In Bild 2 sieht man, dass beim ATmega88 die Sendeleitung (TXD = Transmit Data) an Pin 3 des Mikrocontrollers und die Empfangsleitung (RXD = Receive Data) an Pin 2 herausgeführt sind. Außerdem kann man mit BASCOM eine Software-UART definieren, wenn man eine zusätzliche UART benötigt bzw. einen ATtiny-Mikrocontroller ohne Hardware-UART benutzt.

Da es bei der asynchronen seriellen Datenübertragung keine Taktleitung gibt, ist es wichtig, dass Sender und Empfänger auf dieselbe Datenübertragungsgeschwindigkeit (Maßeinheit 1 Baud) eingestellt sind. In BASCOM gibt es dafür die Compiler-Direktive $BAUD.

Bei UART werden 5 bis 9 Datenbit für die Übertragung eines Zeichens zu einer Einheit zusammengefasst. Zusätzlich gibt es ein Startbit und ein oder zwei

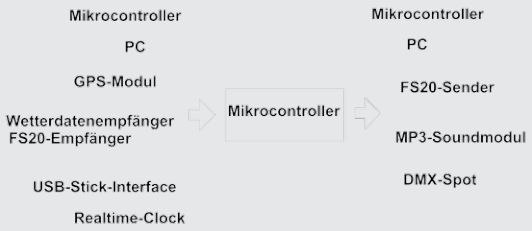


Bild 1: Serielle Kommunikation zur Außenwelt (Beispiele)

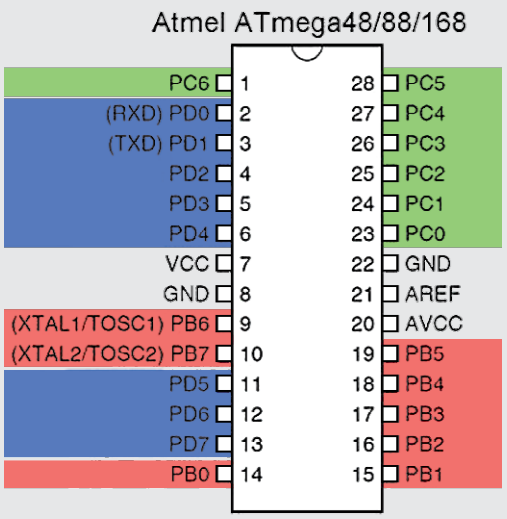


Bild 2: Pinbelegung ATmega48/88/168

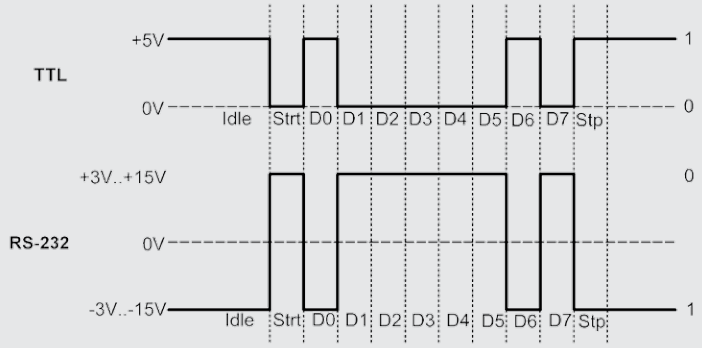


Bild 3: Serielle Datenübertragung (TTL und RS232)

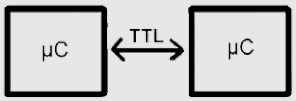


Bild 4: TTL-Verbindung

Stoppbit sowie eventuell ein Paritätsbit, welches eine gewisse Fehlererkennung zulässt. Üblicherweise benutzt man ein Startbit, 8 Datenbit, kein Paritätsbit und ein Stoppbit (8N1). Bild 3 zeigt die serielle Datenübertragung des Zeichens „A" (binär &b0100_0001): Im Ruhezustand (Idle) liegen +5 V am Ausgangspin. Ein Zeichen beginnt mit einem Startbit (0 V). Danach werden die Datenbits (mit dem niedrigstwertigen Bit zuerst) übertragen. Ein Stoppbit (+5 V) beendet die Übertragung eines Zeichens. Anschließend folgt das nächste Zeichen oder (wie hier) zunächst der Ruhezustand.

Außer der Vereinbarung über eine gemeinsame Geschwindigkeit (z. B. 2400 Baud), der Anzahl der Datenbits pro Zeichen (z. B. 8), der Anzahl der Paritäts- und Stoppbits ist der verwendete Spannungspegel wichtig. Mikrocontroller arbeiten mit CMOS-Spannungslevel: 0 V entspricht einer logischen 0 und Betriebsspannung entspricht einer logischen 1 (Bild 3 oben). Wenn der Mikrocontroller mit 5 V betrieben wird, spricht man in Anlehnung an die traditionelle TTL-Technologie (TTL = Transistor-Transistor-Logik) auch von TTL-Level und meint damit die Darstellung einer logischen 1 durch eine Spannung von ca. 5 V und die Darstellung einer logischen 0 durch ca. 0 V. Die genauen Spannungsbereiche und deren Zuordnung zu den Logikpegeln sind in den jeweiligen Datenblättern spezifiziert.

Bei einer Verbindung von zwei Geräten mit TTL-Level können diese direkt verbunden werden (Bild 4). Dabei wird die Sendeleitung des Senders mit dem Empfangspin des Empfängers verbunden und, falls benötigt, umgekehrt

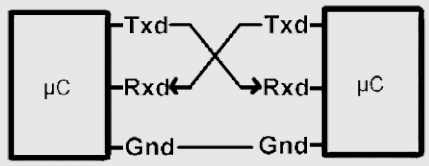


Bild 5: TTL-Kreuzverbindung

Bild 6: Pegelkonvertierung mit MAX232/MAX232A von Maxim Integrated

Bild 7: USB-Verbindung mit FT232R von FTDI bzw. CP2102 von Silicon Labs

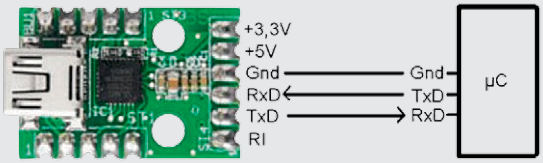


Bild 8: USB-UART-Umsetzer UM2102

Bild 9: Quarz an ATmega88

ebenso. Zusätzlich sind die beiden Geräte mit einer gemeinsamen GND-Leitung zu verbinden (Bild 5).

Bei einer RS232-Schnittstelle wird die serielle Datenübertragung mit Signalpegeln von +3 bis +15 V (für eine logische 0) und -3 bis -15 V (für eine logische 1) realisiert. Im Vergleich zu TTL/CMOS sind also die Spannungspegel anders definiert und außerdem invertiert (Bild 3 unten).

Für die Verbindung von RS232-Geräten mit TTL/CMOS-Geräten gibt es Level-Umsetzer (Levelshifter). In Bild 6 sieht man schematisch einen zwischengeschalteten MAX232-Baustein als Level-Umsetzer. Es gibt auch Umsetzer zwischen einer seriellen UART-Einheit mit TTL/CMOS-Pegel und USB (Bild 7).

ELV bietet für die Verbindung eines PCs mit USB-Anschluss mit einem Mikrocontroller den USB-UART-Umsetzer UM2102 an (Bild 8). Der ELV-Umsetzer UO2102 bietet zusätzlich eine optische Trennung der Geräte. Beim myAVR-Board MK2 ist ein USB-UART-Umsetzer mySmartUSB MK2 bereits enthalten und kann mit Hilfe des myAVR-Tools Progswitch vom Programmer-Modus in den Bridge-Modus umgeschaltet werden. Auf jeden Fall müssen die GND-Anschlüsse der kommunizierenden Geräte verbunden werden. Wenn man nur eine Richtung benötigt, reichen zwei Kabel aus – Datenleitung und GND-Leitung.

Da es bei der asynchronen seriellen Datenverbindung, wie bereits beschrieben, darauf ankommt, dass Sender und Empfänger von derselben Übertragungsgeschwindigkeit ausgehen, damit die Bits richtig abgetastet werden können, ist ein präziser Takt sehr wichtig. Der ungenaue interne Oszillator ist daher nicht geeignet. Man sollte den Mikrocontroller mit einem extern angeschlossenen Quarz betreiben, sobald serielle Datenübertragungen durchgeführt werden. Der Quarz wird, wie in Bild 9 dargestellt, zwischen XTAL1 und XTAL2 mit zwei Kondensatoren (je nach Quarz ca. 22 pF) gegen GND angeschlossen. Die Quarzfrequenz muss mit der $Crystal-Direktive im BASCOM-Programm angegeben werden und in den Fuse-Bits des Mikrocontrollers muss auf „external Crystal" umgestellt werden (in der BASCOM-Entwicklungsumgebung: Programmieren – Zum Chip senden – Manuell programmieren – Lock and Fuse Bits).

BASCOM stellt verschiedene Befehle für die Benutzung serieller Datenverbindungen zur Verfügung (Bild 10). Mit $BAUD wird die Datenübertragungsrate eingestellt. Mit PRINT und PRINTBIN werden Daten seriell vom Mikrocontroller an andere Einheiten gesendet. Hierfür werden in diesem und im nächsten Artikel einige Beispiele gezeigt. Das Empfangen serieller Daten von anderen Einheiten kann mit unterschiedlichen Techniken und BASCOM-Befehlen geschehen, die im weiteren Verlauf der Artikelserie detailliert beschrieben werden.

Senden

Soll (nur) gesendet werden, muss eine Verbindung vom TXD-Pin des Mikrocontrollers zum RXD-Pin des Empfängers hergestellt werden. Zusätzlich werden die GND-Leitungen von Sender und Empfänger verbunden.

Senden vom Mikrocontroller zum PC

Es ist sehr einfach, Daten mit BASCOM seriell zu senden. Nützlich ist das beispielsweise zu Kontrollzwecken während der Programmentwicklung, indem an verschiedenen Stellen des Programms Variableninhalte an einen PC gesendet und dort angezeigt werden. Zur Anzeige auf dem PC kann man sogenannte Terminalprogramme wie zum Beispiel das in BASCOM

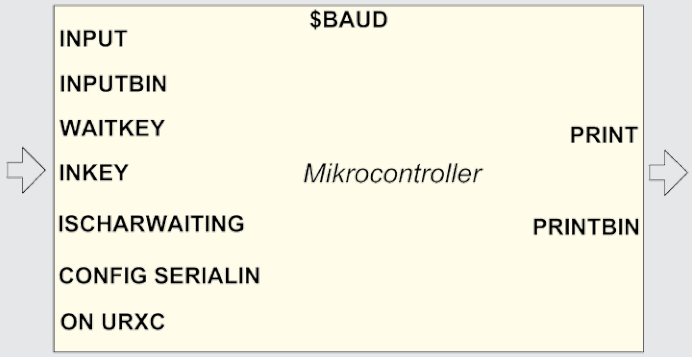


Bild 10: BASCOM-Befehle

Elektronikwissen

Baudratenquarz

Bei der asynchronen seriellen Kommunikation ist es wichtig, dass Sender und Empfänger dieselben Übertragungsparameter (Baudrate, Anzahl Daten-, Paritäts- und Stoppbits) benutzen und – da es keine Taktleitung gibt – dass die Taktfrequenzen der beteiligten Geräte mit der verwendeten Baudrate zusammenpassen.
Die Synchronisierung erfolgt durch das Startbit. Eine halbe Bitlänge nach Ende des Startbits tastet der Empfänger den Spannungspegel ab. Danach jeweils eine Bitlänge später. Wenn die Taktfrequenzen und die Baudrate nicht zusammenpassen, verschiebt sich der Abtastzeitpunkt von der Mitte des empfangenen Bits nach hinten oder nach vorne und es kann zu falschen Werten kommen. Damit es nicht zu den beschriebenen Abtastfehlern kommt, kann man weder einen instabilen internen RC-Oszillator noch einen beliebigen Quarz als Taktgeber für den Mikrocontroller verwenden. Man muss einen Quarz mit zunächst einmal „krumm" aussehender Frequenz verwenden, damit die Übertragung fehlerfrei erfolgen kann. Derartige gut passende Quarze nennt man Baudratenquarze. Übliche Frequenzen von Baudratenquarzen sind 1,8432 MHz, 3,6864 MHz, 7,3728 MHz, 11,0592 MHz, 14,7456 MHz, 18,4320 MHz.
Die zu verwendende Baudrate ist entweder durch einen Kommunikationspartner vorgegeben (z. B. ELV-Modul oder Sensor) oder kann frei gewählt werden, wenn auf keiner Seite etwas vorgegeben ist. Historisch bedingt haben sich Baudraten von 2400, 4800, 9600, 14.400, 19.200, 28.800 usw. etabliert.
Außer dem manuellen Ausrechnen zusammenpassender Taktfrequenz-Baudrate-Paarungen gibt es verschiedene Möglichkeiten, zusammenpassende Werte für Prozessortakt und Baudrate zu ermitteln:

1. Im Datenblatt des verwendeten Mikrocontrollers (beim ATmega88 im Kapitel 20.11) findet man Tabellen mit zusammenpassenden Takt-Baud-Kombinationen
2. In der BASCOM-Entwicklungsumgebung: Programmieren – Ergebnis anzeigen
3. In BASCOM: Optionen – Compiler – Communication
4. In Baudratentabellen bzw. -rechnern im Internet (www.bascom-buch.de/)

integrierte Terminalprogramm oder HTerm von www.der-hammer.info verwenden. Man kann auch Messwerte (Temperaturen o. Ä.) zum PC übertragen (loggen) lassen. Wenn die Messwerte am PC empfangen wurden, können die Werte auch in Tabellenkalkulationsprogramme oder Spezialprogramme zur weiteren Verarbeitung oder grafischen Darstellung importiert werden. Und schließlich gibt es Geräte, die sich mithilfe seriell gesendeter Befehle steuern lassen.

Das folgende Beispielprogramm demonstriert die wesentlichen Sendebefehle PRINT und PRINTBIN:

```
' BASCOM-Programm
'
' Serielle Ausgabe an PC (über USB-UART-Umsetzer UM2102 o.ä.)
' Texte/Zahlen von AVR zum PC senden mit PRINT/PRINTBIN
' Am PC empfangen mit BASCOM-Terminal, HTerm oder anderem Programm
'
'  In:  Taster an B.0
' Out: Serielles Signal an Pin  d.1  = Txd

$regfile    = "M88def.dat"              'Verwendeter Chip
$crystal    = 3686400                   'Verwendete Frequenz. Fuse-Bits auf Externen Quarz einstellen!
$hwstack    = 40                        'Rücksprungadressen (je 2), Registersicherungen (32)
$swstack    = 40                        'Parameteruebergaben (je 2), LOCALs (je 2)
$framesize  = 60                        'Parameter (Daten-Laenge), Rechenbereich Funktionen

$baud = 9600                            '9600 Bits pro Sekunde, 8 Datenbits, Keine Parität, 1 Stoppbit

Config Portb.0 = Input                  'Fuer Taster
Portb.0 = 1                             'Pullup-Widerstand
Taster1 Alias Pinb.0

Dim Zahl As Byte

Print "ELVjournal"
Print "Mikrocontroller-Einstieg ";      'Semikolon unterdrückt CRLF
Print "mit BASCOM-AVR"

Zahl = 65
Print "Mit Print: "
Print Zahl                              '65 als Zeichen "6" und "5" senden
Print Zahl
Print "Mit PRINTBIN: "
Printbin Zahl                           '65 als Zahl senden
Printbin Zahl

Print                                   'sendet nur CRLF = In neue Zeile springen
Print
Print "Mit Print: ";
Zahl = 3
Print Zahl
Print "Mit PRINTBIN: ";
Printbin Zahl

Print
Print
Zahl = 5

Do
Print "Zahl= " ; Zahl
Incr Zahl
Wait 1

If Taster1 = 0 Then                     'Wenn Taster gedrückt dann ..
  Zahl = 0
  Print "Taste  = Zahl auf Null"
```

```
End If

Loop
End
```

Erläuterungen:
Nach der üblichen Beschreibung des verwendeten Mikrocontrollers, der Taktfrequenz und der Stackwerte wird mit $BAUD = 9600 die Datenübertragungsrate festgelegt. Dieselbe Baudrate muss auf jeden Fall im Terminalprogramm bzw. einem Empfänger eingestellt sein. Auch die Anzahl von Datenbits, Paritätsbits und Stoppbits muss beim Empfänger auf dieselben Werte wie beim Sender eingestellt werden.
Mit dem BASCOM-Befehl PRINT werden Texte bzw. Variableninhalte gesendet. Auch Zahlen werden als Zeichen zum Empfänger übertragen. Die Zahl 65 wird mit PRINT also als das Zeichen „6" und dann das Zeichen „5" gesendet. Nach dem Senden mit PRINT wird CR (Carriage Return, 13) und LF (Line Feed, 10) gesendet. Die Schreibmarke im Terminalfenster springt dadurch an den Anfang der nächsten Zeile. Um in derselben Zeile fortzufahren, muss der PRINT-Befehl mit einem Semikolon abgeschlossen werden. Mit PRINTBIN werden Zahlen nicht als Zeichen, sondern als Zahlen übertragen. Ein CRLF wird bei PRINTBIN nicht angehängt.

Bild 11 zeigt die Ausgabe am PC mit dem BASCOM-Terminalprogramm. Bild 12 zeigt die Ausgabe mit dem Terminalprogramm HTerm, welches sehr viele Möglichkeiten der Konfiguration bietet. In Bild 12 ist HTerm so eingestellt, dass zusätzlich zu den Zeichen (wie im BASCOM-Terminal) auch die Dezimalwerte angezeigt werden. Man sieht dadurch sehr schön am Ende der Zeilen die Dezimalzeichen 13 und 10 für CR und LF.

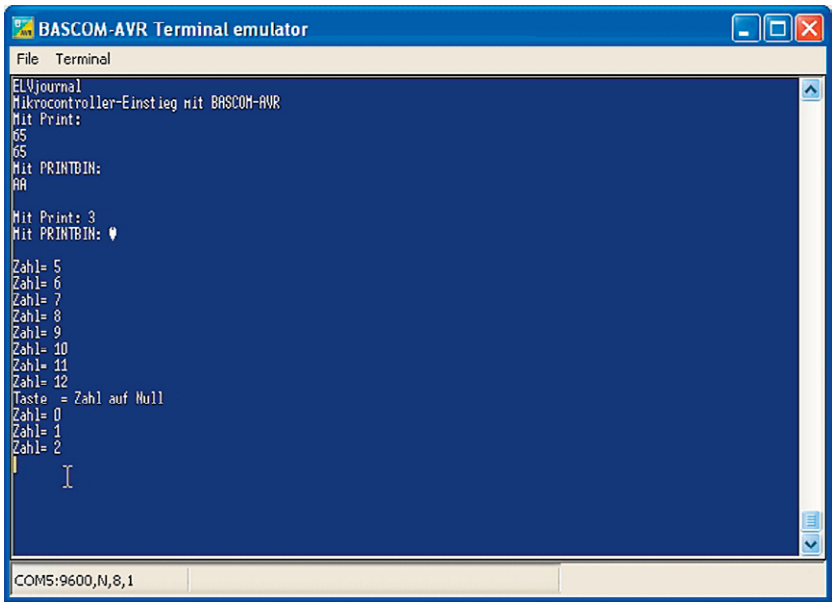


Bild 11: BASCOM-Terminal: Ausgabe durch PRINT/PRINTBIN

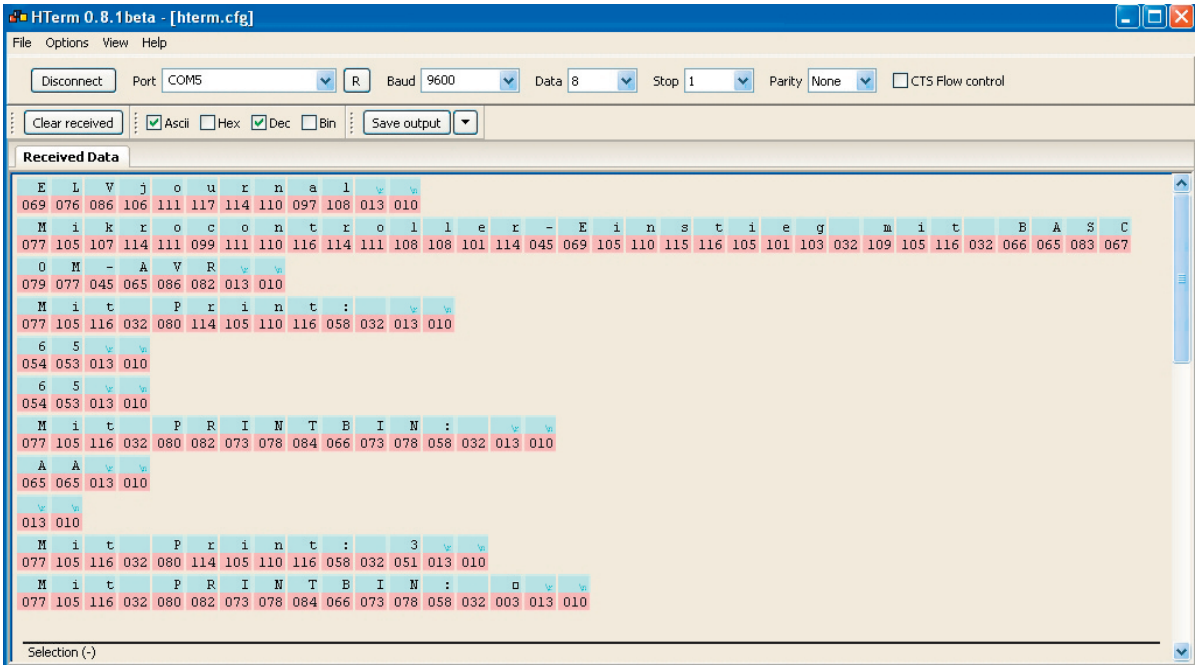


Bild 12: HTerm-Terminal: Ausgabe durch PRINT/PRINTBIN

Daten an FS20-Sender übertragen

Das Produkt FS20 US von ELV stellt einen Baustein dar, mit dem sich FS20-Empfänger mit einfachen Befehlen über die serielle Schnittstelle eines Mikrocontrollers steuern lassen. Dafür reicht es aus, eine Verbindung vom TXD-Pin des Mikrocontrollers zum RXD-Pin des FS20-Moduls herzustellen sowie die jeweiligen GND-Anschlüsse zu verbinden (Bild 13). Das FS20-Modul wird mit einer positiven Versorgungsspannung von 3 bis 5 V betrieben.

Vom BASCOM-Programm aus kann man nun beliebige FS20-Empfänger ansteuern, indem gemäß dem im Datenblatt beschriebenen Protokoll entsprechende Kommandos seriell an das Modul gesendet werden.

```
' BASCOM-Programm
'
' Serielle Ausgabe an FS20 US (UART-FS20-Übersetzer)
' Kommandos von AVR zum FS20US senden
'
'  In:  Taster an B.0 und B.1
' Out: Serielles Signal an Pin  d.1  = Txd
' Out: LED an Portb.2

$regfile  = "M88def.dat"                  'Verwendeter Chip
$crystal = 3686400                        'Verwendete Frequenz. Fuse-Bits auf Externen Quarz einstellen!
$hwstack = 40                             'Rücksprungadressen (je 2), Registersicherungen (32)
$swstack = 40                             'Parameteruebergaben (je 2), LOCALs (je 2)
$framesize    = 60                        'Parameter (Daten-Laenge), Rechenbereich Funktionen

$baud = 9600                              '9600 Bits pro Sekunde, 8 Datenbits, Keine Parität, 1 Stoppbit

Config Portb.0 = Input                    'Taster1
Portb.0 = 1                               'Pullup-Widerstand
Taster1 Alias Pinb.0

Config Portb.1 = Input                    'Taster2
Portb.1 = 1                               'Pullup-Widerstand
Taster2 Alias Pinb.1

Config Portb.2 = Output                   'LED
Led Alias Portb.2

Const Startzeichen = &H02                                   'Das Startzeichen ist immer &h02  (hexadezimal 2)
Dim Befehlslaenge As Byte                                   'Variable für Befehlslaenge
Dim Befehls_id As Byte                                      'Befehl an FS20US-Modul
Dim Hauscode1 As Byte , Hauscode2 As Byte                   'Variablen für Hauscode
Dim Kanaladresse As Byte                                    'Variable für Kanaladresse
Dim Fs20befehl As Byte                                      'Variable für FS20_Befehl
Dim Erweiterungsbyte As Byte                                'Erweiterungsbyte

Befehlslaenge = 6
Befehls_id = &HF1                                           'FS20-Befehl einmal senden: &hF1
Hauscode1 = &HFD                                                        '
Hauscode2 = &H04
Kanaladresse = &H02

Const Aus = &H00                                            'Aus
Const Maximalwert = &H10                                    'Einschalten auf Maximum

Do

If Taster1 = 0 Then
  Led = 1                                                   'Kontroll-LED an
  Fs20befehl = Maximalwert
  Printbin Startzeichen ; Befehlslaenge ; Befehls_id ; Hauscode1 ; Hauscode2 ; Kanaladresse ;
Fs20befehl ; Erweiterungsbyte
End If
```

```
If Taster2 = 0 Then
  Led = 0                                          'Kontroll-LED aus
  Fs20befehl = Aus
  Printbin Startzeichen ; Befehlslaenge ; Befehls_id ; Hauscode1 ; Hauscode2 ; Kanaladresse ;
Fs20befehl ; Erweiterungsbyte
End If

Loop
End
```

Erläuterungen:
Die serielle Kommunikation mit dem FS20 US erfolgt mit 9600 Baud, 8 Datenbit, ohne Paritätsbit und 1 Stoppbit (9600, 8, N, 1). Im BASCOM-Programm wird deshalb mit $BAUD = 9600 die Übertragungsgeschwindigkeit entsprechend eingestellt. Das Protokoll des FS20 US schreibt das Senden von Befehls-ID (z. B. &hF1 für einmaliges Senden), Hauscode1, Hauscode2 und Kanaladresse (entsprechend Tabellen 2 und 3 im Datenblatt), FS20-Befehl (z. B. 0 für Aus; vgl. Tabelle 4 im Datenblatt) sowie Erweiterungsbyte (wie Zeiten) vor. Vorangestellt wird jeweils ein Startzeichen (&h02) und die Befehlslänge. Im BASCOM-Programm wird das Startzeichen als Konstante definiert. Die benötigten Variablen werden deklariert und mit Startwerten belegt.

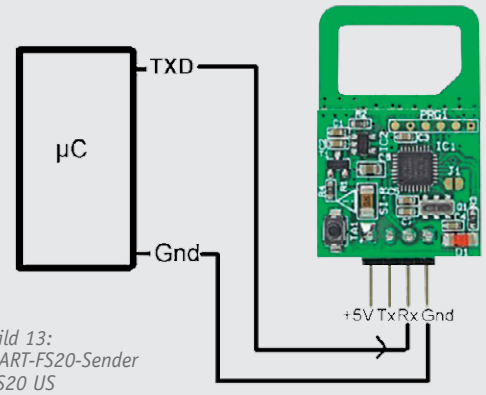


Bild 13:
UART-FS20-Sender
FS20 US

In der Hauptschleife (DO-LOOP) werden zwei Taster abgefragt. Je nachdem, welcher Taster gedrückt wird, ist die jeweilige IF-Bedingung erfüllt. Im entsprechenden Zweig wird zunächst eine Kontroll-LED ein- bzw. ausgeschaltet. Danach wird der Wert für den FS20-Befehlscode (Tabelle 4 im Datenblatt) der Variablen FS20-Befehl zugewiesen. Mit PRINTBIN werden die benötigten Zahlen über die serielle Schnittstelle an das FS20-US-Modul gesendet und dort in einen FS20-Sendebefehl umgewandelt.

Selbstverständlich könnte man die Zahlen auch direkt in den PRINTBIN-Befehl schreiben. Das Programm soll aber als Basis für die Einbindung in eigene Projekte dienen, bei denen man situationsbedingt Werte an Variablen zuweisen und dann den gesamten PRINTBIN-Befehl aufrufen kann. Man könnte beispielsweise ab einer kritischen Temperatur eine mit einer Funk-Steckdose FS20 ST geschaltete Alarm-Leuchte oder eine Sirene ansteuern.

Ausblick

Nachdem in diesem Teil die Grundlagen der seriellen Datenübertragung dargestellt und Beispiele mit den BASCOM-Befehlen PRINT und PRINTBIN gezeigt wurden, wird es im ELVjournal 1/2014 um die Ansteuerung des ELV Soundmoduls MSM3 und eine Steuerung von DMX-Lichtspots gehen. ELV

Weitere Infos:

- Stefan Hoffmann: Einfacher Einstieg in die Elektronik mit AVR-Mikrocontroller und BASCOM. Systematische Einführung und Nachschlagewerk mit vielen Anregungen. ISBN 978-3-8391-8430-1
- www.bascom-buch.de
- www.mcselec.com
- www.atmel.com

Alle Infos zu den Produkten/Bauteilen finden Sie im Web-Shop.
Preisstellung Oktober 2013 – aktuelle Preise im Web-Shop

Empfohlene Produkte/Bauteile:	Best.-Nr.	Preis
BASCOM-(Demo-)Lizenz von MCS Electronics www.mcselec.com	–	–
Atmel-AVRISP-mkII-Programmer	JZ-10 03 55	€ 39,95
oder myAVR-Board MK2	JZ-10 90 00	€ 49,–
Netzteil für myAVR-Board MK2	JZ-10 90 01	€ 6,95
ATmega88	JZ-10 07 62	€ 3,95
100-nF-Kondensator	JZ-10 03 17	€ 0,08
Batteriehalter für 3x Mignon	JZ-08 15 30	€ 0,75
Batterieclip für 9-V-Block-Batterie	JZ-08 01 28	€ 0,30
BASCOM-Buch	JZ-10 90 02	€ 54,–
Experimentier-Board 1202B	JZ-07 72 89	€ 12,95
Schaltdraht-Sortiment	JZ-05 47 68	€ 5,95
LED-Set	JZ-10 63 56	€ 3,95
oder Leuchtdioden	JZ-10 66 60	€ 1,65
und Widerstände	JZ-10 66 57	€ 1,85
Piezo-Signalgeber	JZ-00 73 87	€ 0,95
Mikroschalter und -taster	JZ-10 66 67	€ 2,80
LC-Display, 2x 16 Zeichen	JZ-05 41 84	€ 6,95
oder myAVR-LCD-Add-on		
Pin-Ausrichter	JZ-00 84 63	€ 4,95
USB-UART-Umsetzer UM2102	JZ-09 18 59	€ 5,95
USB-UART-Umsetzer mit Optokopplung UO2102	JZ-10 49 66	€ 12,95
MP3-Soundmodul MSM3	JZ-10 57 29	€ 37,95
UART-FS20 Sender FS20 US	JZ-09 87 89	€ 19,95
Funk-Schaltsteckdose FS20 ST	JZ-10 45 37	€ 29,95
Funk-Dimmer FS20 DI-5	JZ-10 46 46	€ 39,95
DMX-Spot	JZ-10 25 14	€ 39,95
FS20-Diagnosetool FS 20DT	JZ-06 68 13	€ 59,95