2.3 Atmega-Peripherie/Serielle Schnittstelle

2.3.1 Aufgabe

Mit dem AVR soll ein Tester für eine serielle Schnittstelle oder einen USB-RS232-Adapter aufgebaut werden. Die Baudrate soll mit 9600 fest sein. Der Tester soll jeden Kleinbuchstaben zum Großbuchstaben machen und umgekehrt (andere Zeichen in Ruhe lassen) und sofort zurückschicken.

2.3.2 Serielle Schnittstelle

Die serielle Schnittstelle (genormt nach RS232 und V.24) wird allgemein auch UART (= Universelle asynchrone Schnittstelle) genannt. Beim AVR heißt sie dagegen USART (=Asynchron- und Synchron-Schnittstelle). Das liegt daran, dass sie auch einen Synchron-Modus kennt.

Beim Betrieb einer seriellen Schnittstelle muss man vorher festlegen, wie viele Startbits, Datenbits und Stopbits man verwenden möchte und ob man zusätzlich ein Paritätsbit übertragen muss. Üblich sind 7 oder 8 Datenbits, ein Startbit und ein Stopbit.

Ebenso muss die verwendete Baudrate festgelegt werden. Üblich sind Werte zwischen 50 und 230400. Häufig wird eine Baudrate von 9600 benutzt.

Die serielle Schnittstelle arbeitet mit negativer Logik, das Potential zwischen -3 V und -15 V entspricht einer Eins, das Potential zwischen +3 V und +15 V entspricht einer Null. Im Mikrocontrollerbereich dagegen wird allgemein mit positiver Logik gearbeitet (0 bis 2,5 V als Null, 2,5 V bis 5 V als Eins), so dass ein Pegelwandler wie der MAX232 nötig ist. Beim Benutzen eines USB-nach-RS232-Adapters muss man nachsehen, ob darauf solch ein Pegelwandler eingebaut ist. Falls nicht, muss man den Adapter auf dem Controllerboard ebenfalls entfernen.

Die Schnittstelle enthält nach Norm Datenleitungen (Senden: TxD und Empfangen: RxD, vom PC als Datenendgerät aus gesehen) und Handshake-Leitungen (RTS, CTS, eventuell noch DCD und weitere). Für eine Hardware-Flusskontrolle braucht man diese Handshake-Leitungen; man muss dann einige Portleitungen für diesen Zweck abzweigen.

Es besteht auch die Möglichkeit eines Software-Handshake: Mit Strg-S (ASCII-19, XOFF) wird die Übertragung angehalten; mit Strg-Q (ASCII-17, XON) wird sie wieder aufgenommen[1]. Für einfache Beispiele braucht man jedoch meistens noch keine Flusskontrolle.

2.3.3 Hardware

Beim Pollin-Board kann eine einfache RS232-Verbindung verwendet werden. Es ist kein Nullmodemkabel erforderlich. Das Board arbeitet als Datenübertragungseinrichtung, also wie z. B. ein-Modem und damit passend zur Buchse. Der Jumper JP1 ist zu setzen für Empfang, JP2 für Senden.

Beim RN-Board ist ein spezielles Adapterkabel anzufertigen (Abbildung 1). Es sind keine Jumper zu setzen.

2.3.4 Übertragungspartner

Für die Übertragung zu einem Linux-PC empfiehlt sich das graphische Programm `cutecom`. Bei einem Windows-PC kann man das Programm `hyperterminal` verwenden. Beide Programme erlauben die Einstellung der wichtigsten Übertragungsparameter.

2.3.5 USART am AVR

Im ATmega sind mehrere Register für die serielle Schnittstelle verantwortlich.

[1] Auf der Konsole eines Betriebssystems funktioniert das genauso, dort kann man es ausprobieren.

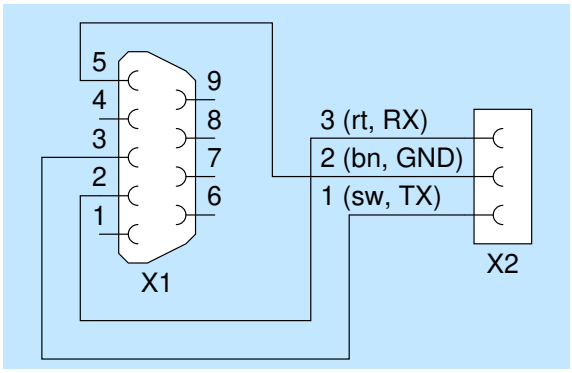

Abbildung 1: Adapterkabel für RN-Board

2.3.5.1 UBRR Das Baudraten-Register UBRR besteht zwei Teilen, dem 8 Bit breiten Low-Byte UBRRL und dem 3 Bit breiten High-Byte UBRRH. Mit ihm kann man die Baudrate einstellen:

$$UBRR = \frac{f_{CPU}}{(Baudrate \cdot 16)} - 1 \tag{1}$$

Bei einem 16-MHz-Controller entspricht dieser Wert der Zeit für ein Bit minus eins. So ist bei einer Baudrate von 9600 die Bitzeit $t_{Bit} = 104\,\mu s$ und $UBRR = 103$. Zu jedem Wert von $UBRR$ gibt es eine Baudrate (siehe Tabelle 1).

	CPU-Frequenz							
Baudrate	4000000	Fehler	8000000	Fehler	12000000	Fehler	16000000	Fehler
50	4999,00	0,00%	9999,00	0,00%	14999,00	0,00%	19999,00	0,00%
75	3332,33	0,01%	6665,67	0,00%	9999,00	0,00%	13332,33	0,00%
100	2499,00	0,00%	4999,00	0,00%	7499,00	0,00%	9999,00	0,00%
150	1665,67	-0,02%	3332,33	0,01%	4999,00	0,00%	6665,67	0,00%
300	832,33	0,04%	1665,67	-0,02%	2499,00	0,00%	3332,33	0,01%
600	415,67	-0,08%	832,33	0,04%	1249,00	0,00%	1665,67	-0,02%
1200	207,33	0,16%	415,67	-0,08%	624,00	0,00%	832,33	0,04%
2400	103,17	0,16%	207,33	0,16%	311,50	-0,16%	415,67	-0,08%
4800	51,08	0,16%	103,17	0,16%	155,25	0,16%	207,33	0,16%
9600	25,04	0,16%	51,08	0,16%	77,13	0,16%	103,17	0,16%
19200	12,02	0,16%	25,04	0,16%	38,06	0,16%	51,08	0,16%
28800	7,68	-3,68%	16,36	2,08%	25,04	0,16%	33,72	-0,80%
57600	3,34	7,84%	7,68	-3,68%	12,02	0,16%	16,36	2,08%
115200	1,17	7,84%	3,34	7,84%	5,51	-7,52%	7,68	-3,68%
230400	-	-	1,17	7,84%	2,26	7,84%	3,34	7,84%

Tabelle 1: Erforderliche Werte für UBBR

$$\frac{f_C PU}{(UBRR + 1) \cdot 16} = Baudrate \tag{2}$$

Bei den neueren Controllern gibt es noch eine Besonderheit: Wenn das Bit 7 von UBRRH gesetzt ist (URSEL-Bit), wird UBRRH zum Control-Register URCSRC.

2.3.5.2 UCR Das Control-Register UCR (bei neueren Controllern UCSRB genannt) legt die Einstellungen der Schnittstelle fest (Tabelle 2)

7	RXCIE	Interrupt nach Empfang
6	TXCIE	Interrupt nach Senden
5	UDRIE	Interrupt, wenn Sende-UDR leer ist
4	RXEN	Empfangen ja
3	TXEN	Senden ja
2	CHR9	(UCSZ2) 9 Bit verwenden ja
1	RXB8	9. Datenbit bei Empfang (Status)
0	TXB8	9. Datenbit bei Sendung (Status)

Tabelle 2: UCR bzw. UCSRB

2.3.5.3 USR Das Status-Register USR (jetzt UCSRA genannt) gibt den Status wieder (Tabelle 3).

7	RXC	RX complete
6	TXC	TX complete
5	UDRE	UDR leer (beim Schreiben auf 0 setzen)
4	FE	Framing Error
3	DOR	Data Overrun: RX-Zeichen nicht abgeholt
2	PE	Parity Error
1	U2X	Doppelte Baudrate ein (Control)
0	MPCM	Multi-Proz.-Betrieb ein(Control)

Tabelle 3: USR bzw. UCSRA

2.3.5.4 UCSRC Mit dem Setzen des URSEL-Bits wird aus dem UBRRH-Register ein zusätzliches Control-Register UCSRC (Tabelle 4).

7	URSEL	Umschaltung zu UBRRH, muss 1 sein
6	UMSEL	Synchronbetrieb (1: ein; 0: aus)
5	UPM1	Paritätseinstellung (1: ein; 0: aus)
4	UPM0	Paritätseinstellung (1: ungerade; 0: gerade)
3	USBS	Zweites Stoppbit (1: ein; 0: aus)
2	UCSZ1	Datenbits (1: 8 oder 7; 0: 6 oder 5)
1	UCSZ0	Datenbits (1: 8 oder 6; 0: 7 oder 5)
0	UCPOL	Phasenlage bei Synchronbetrieb

Tabelle 4: UCSRC

2.3.5.5 UDR Außerdem gibt es noch das Daten-Register UDR für das empfangene oder das zu sendende Datenbyte. In Wirklichkeit sind es zwei Register, je eins für Senden und Empfang, und das Umschalten erfolgt danach, ob man das Register schreibt oder liest.

2.3.5.6 Einstellungen Beim Senden braucht man vor allem zwei Register-Bits:

- UCR.3 (TXEN) = 1
- UDR.6 (TXC) = Transmit Complete

Zum Empfangen braucht man zwei andere Bits:

- UCR.4 (RXEN) = 1
- UDR.7 (RXC) = Receive Complete

2.3.6 Programmierbeispiele

Hier sind einige Programmierbeispiele für Senden und Empfang aufgeführt.

- Senden: `1sende.asm` (cutecom: 9600,keine Parität, 7 Datenbits)
- Empfangen: `2empfange.asm` (cutecom: 9600, Hex input)
- Empfangen mit Interrupt: `3empfange_int.asm`
- Empfangen mit Interrupt: `4empfange_int_led.asm` (cutecom: 9600, Hex input, Parity=Space)
- Senden und Empfangen: `5empf+sende.asm`