Serielle Schnittstelle (RS232)

G. Kemnitz*, TU Clausthal, Institut für Informatik

19. Juli 2007

Zusammenfassung

Für den Empfänger ist eine funktionierende Schaltung vorgegeben. Diese ist zu untersuchen und zu testen. Anschließend ist ein Sender zu entwerfen und der Empfänger um Zusatzfunktionen zu erweitern.

1 Funktion

Die meisten der heutigen PCs besitzen (noch) eine serielle RS232-Schnittstelle. Unsere Versuchsbaugruppe hat auch einen Anschluss für eine solche serielle Schnittstelle, über die die Baugruppe mit einem PC kommunizieren kann. Von dem 9-poligen SUBD-Stecker einer seriellen Schnittstelle werden nur 3 Leitungen genutzt:

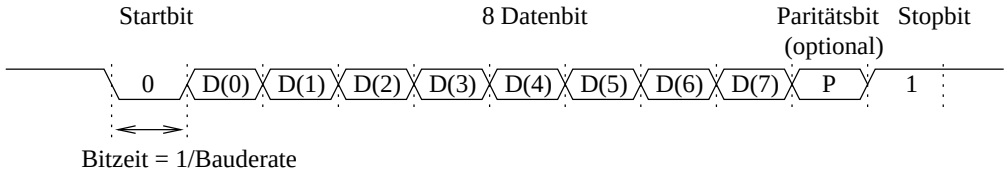
Steckerkontakt 2: RxD, Eingang Empfänger

Steckerkontakt 3: TxD, Ausgang Sender

Steckerkontakt 5: Masse

Eine Null wird als negative und eine Eins als positive Spannung zwischen Signalleitung und Masse übertragen. Die Umwandlung der üblichen Logikpegel (große Spannung, kleine Spannung) in (positive Spannung, negative Spannung) erfolgt in einem speziellen Schaltkreis, in unserem Versuchsaufbau in IC2 auf dem Zusatzboard "Serial I/O1". In der ucf-Datei heißt der Empfängereingang sio_RxD und der Senderausgang sio_TxD.

Die Übertragung über RS232 erfolgt byteweise. Während einer Übertragungspause ist das Sendesignal TxD='1'. Jedes Datenpaket beginnt mit einem Startbit='0' gefolgt von den 8 Datenbits, optional einem Paritätsbit und einem Stopbit='1'. Danach kann sich eine Pause (Übertragungsleitung gleich Eins) oder das Startbit der nächsten Übertragung anschließen. Das Paritätsbit berechnet sich aus der Modulo-2- oder EXOR-Summe der Datenbits und dient zur Fehlererkennung.



Der Sender für die serielle Schnittstelle besteht aus

- einer Schaltung zur Bereitstellung des Taktes
- einem Automaten zur Umwandlung eines Bytes in einen Datenstrom entsprechend Protokoll.

*Tel. 05323/727116

Der Basistakt CLK unserer Übungsbaugruppe hat eine Frequenz von 50MHz. Dieser Takt wird üblicherweise heruntergeteilt auf einen Takt mit der 16-fachen Baudrate und einem Tastverhältnis 1:1:

.

```
process(CLK)
  variable Teiler: integer range 0 to cTeiler-1:=0;
 begin
  if CLK'event and CLK='1' then
    if Teiler=cTeiler-1 then
      Teiler:=0;
      Baudtakt_x16 <= not Baudtakt_x16;
    else
      Teiler:=Teiler+1;
    end if;
  end if;
 end process;
```

Gebräuchliche Baudraten und Teilerkonstanten zeigt nachfolgende Tabelle:

Baudrate	cTeiler (Basistakt 50 MHz)
2,4 kBaude	648
4,8 kBaude	324
9,6 kBaude	162
19,2 kBaude	81

Der Takt Baudrate_x16 wird in einen weiteren Teiler (4-Bit-Zähler) durch 16 geteilt und als Takt für den Sendeautomaten genutzt.

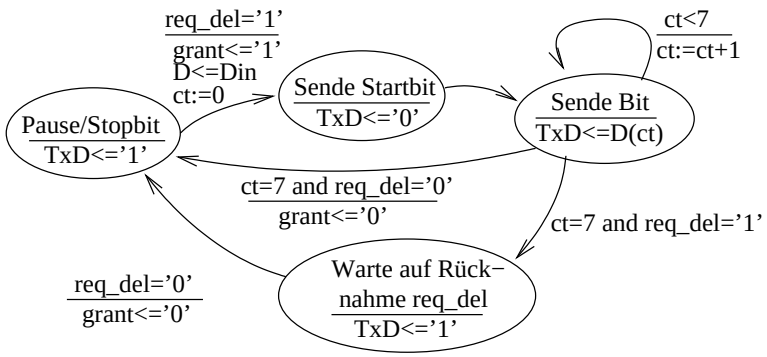
Der Sendeautomat soll als Graph spezifiziert werden. Solange keine Sendung gestartet ist, bleibt der Automat im Pausezustand und hält die Sendeleitung TxD auf Eins. Wenn eine Übertragung gestartet wird, durchläuft er die Zustandsfolge:

- Sende Startbit: TxD<='0'
- Sende Bit 0: TxD<=D(0)
- ...
- Sende Bit 7: TxD<=D(7)
- Sende Paritätsbit: TxD<=D(0) xor D(1) xor ... xor D(7)
- Sende Stopbit (TxD<='1')

Die zu versendenden Daten kommen in der Regel von einer Schaltung, die nicht mit dem Baudratentakt arbeitet. Das erfordert ein Handshake-Protokoll:

- Datenquelle setzt das Anforderungssignal req auf Eins und stellt die Daten auf dem Bus Din bereit
- Der serielle Sender übernimmt, sobald er zur Übernahme bereit ist, die zu versendenden Daten und bestätigt mit grant<='1'
- Versenden der Daten
- Warten auf Rücknahme der Anforderung und Rücknahme der Bestätigung
- Warten auf die nächste Anforderung

Sende- und Handshakeablauf sind zu einem Ablaufgraphen zu vereinigen, wobei das Anforderungssignal nur in abgetasteter Form verarbeitet werden darf.



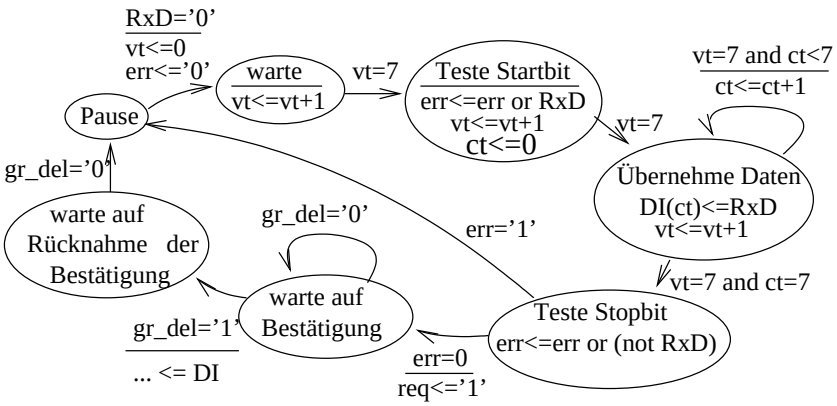
Anmerkungen:

- Der dargestellte Automat versendet die Daten ohne Paritätsbit.
- Der Sender ist als gekoppelter Automat beschrieben. Der Hauptautomat steuert den als Variable modellierten Zähler ct. Der Automat könnte auch so umgebaut werden, dass die versendeten Bits nicht durch einen Zähler, sondern den Automatenzustand unterschieden werden.
- Der Zähler muss vom Typ Integer range 0 to 7 sein, damit er eines von 8 Bits aus D auswählen kann. D muss entsprechend den Typ std_logic_vector(7 downto 0) haben.

Der Empfänger arbeitet mit derselben Baudrate wie der Sender. Dadurch kennt er die Bitzeit. Den Startzeitpunkt einer Übertragung bekommt er vom Sender nur indirekt in Form der fallenden Flanke zu Beginn der Übertragung mitgeteilt. Aus der Kenntnis der Baudrate und der Startflanke muss er die Übernahmezeitpunkte der einzelnen Bits bestimmen. Die Lösung beinhaltet, dass der 1:16-Vorteiler einbezogen wird. Der Empfänger startet wie der Sender im Pausezustand. Statt auf eine Sendeanforderung wartet er auf die fallende Flanke auf der Empfangsleitung RxD. RxD wird dabei mit der mehrfachen, im Beispiel mit der 16fachen Baudrate abgetastet. Sobald die Datenleitung auf Null wechselt, schaltet der Automat einen Zustand weiter und setzt den Taktvorteiler auf Null. Alle weiteren Zustandsübergänge erfolgen, wenn der Vorteiler die Hälfte seines Endwertes erreicht, d.h. etwa in der Mitte zwischen den Signalwechseln des Senders, wenn die gesendeten Daten stabil sind. Beim ersten Zustandswechsel wird eine Null auf der Empfangsleitung erwartet (Startbit). Beim 2. bis 9. Zustandswechsel werden die Datenbits übernommen. Abschließend wird kontrolliert, dass das Stopbit Eins ist. Falls zur Fehlererkennung zusätzlich ein Paritätsbit mit übertragen wird, ist auch das auszuwerten.

Auch die Übergabe empfangender Daten an die weiterverarbeitende Schaltung erfolgt üblicherweise über Handshake:

- Nach fehlerfreiem Empfang eines Bytes Schreibanforderung stellen
- Auf Bestätigung warten und Anforderung zurücknehmen
- Auf Rücknahme der Bestätigung warten.



Anmerkungen:

- Kein Paritätsbit
- vt ist ein Zähler modulo 16
- ct muss wie beim Sender vom Integer range 0 to 7 sein, damit es ein Bit aus dem 8-Bit-Vektor DI auswählen kann
- das Bestätigungssignal der nachfolgenden Einheit ist, um laufzeitbedingte Fehlfunktionen auszuschließen, mit dem Takt des Empfängers abzutasten.

2 Test eines Beispielentwurfs

2.1 Projekt Anlegen

Legen Sie ein neues Verzeichnis

```
H:\TGP\Afg_RS232
```

an und kopieren Sie aus dem Netz die Dateien

- RS232.npl
- Empfang.vhd
- Praktikum.ucf

in das Verzeichnis und öffnen Sie das Projekt.

2.2 Beispielentwurf Ansehen und Testen

1. Rekonstruieren Sie den Graphen des Empfangsautomaten aus der Datei Empfang.vhd und vergleichen Sie den gefunden Graphen mit dem oben beschriebenen Graphen (Hausaufgabe).
   (a) Empfang mit oder ohne Auswertung des Paritätsbits?
   (b) Welche Baudrate ist eingestellt?
   (c) Enthält der Automat eine Handshake-Funktion?
   (d) Wie funktioniert die zuschaltbare Echofunktion?
2. Übersetzen Sie den Entwurf und programmieren Sie den Schaltkreis.

3. Starten Sie auf dem PC das Programm Hyperterminal

   (a) Startmenue: Start=>Programme=>Zubehör=>Kommunikation=HyperTerminal oder Ausführen von C:\Programme\WindowsNT\hypertrm.exe

   (b) Im Fenster "Beschreibung der Verbindung" für Name beliebige Zeichenkette, z.B. "x" eingeben und o.k.

   (c) Im Fenster "Eigenschaften von <Name>" COM1 lassen und o.k.

   (d) Im Fenster "Eigenschaften von COM1" Standardwerte wiederherstellen: (9600 Bits pro Sekunde, 8 Datenbits, kein Paritätsbit, 1 Stopbit, keine Flusssteuerung) und o.k.

4. Verbindung Testen: SW1 auf 1 stellen. Nacheinander die Ziffern 0 bis 9 senden und kontrollieren, dass auf led(7 downto 0) die zugehörigen ASCII-Codes 30h bis 39h angezeigt werden. Echo abschalten (SW1 auf 0 umschalten) und Test wiederholen.

5. Experimentieren: Andere Zeichen ausprobieren; Verbindungsparameter ändern (Baudrate verdoppeln/halbieren, Paritätsbit einstellen) und schauen, wie sich das auf die empfangenen Daten auswirkt.

3 Entwurfsaufgaben

1. Entwerfen Sie einen Sendeautomaten, der bei Betätigung der Taste BTN0 das Byte SW(8 downto 0) an den PC sendet. Achtung: Entprellung nicht vergessen!

2. Modifizieren Sie die Empfängerschaltung so, dass auf den beiden linken 7-Segmentziffern die Anzahl der empfangenen Bytes und auf den beiden rechten 7-Segmentziffern das letzte empfangene Byte als 2-stellige Hexadezimalzahlen dargestellt werden.

4 Mögliche Probleme und Fehlersuche

Wenn die serielle Kommunikation mit dem PC nicht funktioniert, kann das Problem auch auf der PC Seite liegen:

1. Wenn nie Daten an den PC gesendet werden, bleibt das Programm Hyperterminal möglicherweise hängen (zeitweise keine Reaktion auf Mouse- und Tastaturereignisse). Ursache sind wahrscheinlich irgendwelche Autodetect-Funktionen von Windows 2000. Problembeseitigung: Daten an den PC senden, z.B. durch Einschalten der Echo-Funktion. Alternativ kann auch der Nachbarrechner unter Linux gebootet und die Verbindung wie im Assemblerkurs über Minicom hergestellt werden.

2. Kein Empfang gesendeter Zeichen: Ursache kann auch ein falsches Stop- oder Paritätsbit sein. Hyperterminal/Minicom zeigen eingehende Daten mit falschem Paritäts- oder Stopbit nicht an. Falls dieses Problem bei Ihnen auftritt, ist der vernünftigste Weg zur Fehlereingrenzung ChipScope. Der integrierten Logikanalysator (ila) sollte das serielle Sende- und Empfangssignal sowie die Automatenzustände beobachten. Zur Taktung ist die 4-fache Baudrate, abzugreifen am 1:16 Vorteiler des Senders, geeignet.

5 Aufräumen

- Über Menüpunkt "Project, Cleanup Project Files" automatisch generierte Design-Files löschen.
- Netzteil zur Spannungsversorgung aus der Steckdose ziehen.
- Modelsim und Projektnavigator beenden.