Aufgabe 3: Serielle Schnittstelle (RS232)

G. Kemnitz*, TU Clausthal, Institut für Informatik

7. Juli 2014

Zusammenfassung

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

1 Funktion

Die PCs im Übungsraum und unsere Versuchsbaugruppe sind über ein serielles Schnittstellenkabel miteinander verbunden, über das byteweise Daten übertragen werden können. 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 IC14. In der ucf-Datei heißt der Empfängereingang RxD und der Senderausgang 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 Stoppbit='1' (vgl. Abbildung 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.

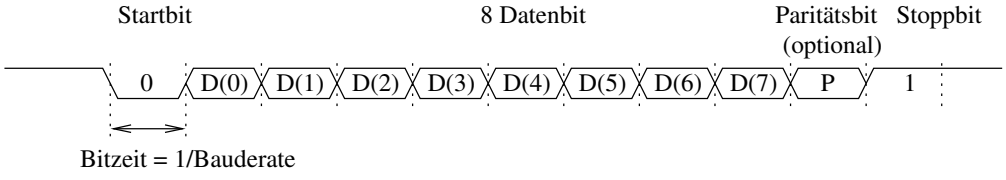

Abbildung 1: Rahmenformat RS232

Der Sender für die serielle Schnittstelle besteht aus

*Tel. 05323/727116

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

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 counter : integer range 0 to cTeiler - 1 := cTeiler - 1;
begin
  if rising_edge(clk) then
    if counter = 0 then
      clk_16x_baud <= not clk_16x_baud;
      counter := cTeiler - 1;
    else
      counter := counter - 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 clk_16x_baud wird in einem 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 Stoppbit (TxD<='1')

Die zu versendenden Daten werden in der Regel von einer Schaltung, die nicht mit dem Baudratentakt arbeitet, bereitgestellt. 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 Handshake-Ablauf sind zu einem Ablaufgraphen zu vereinigen, wobei das Anforderungssignal nur in abgetasteter Form verarbeitet werden darf (Abbildung 3).

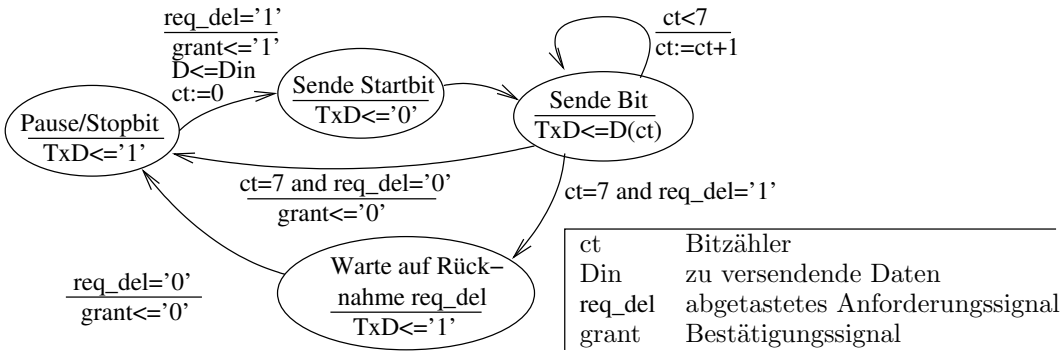

Abbildung 2: Ablaufgraph des Senders mit Handshake

Anmerkungen:

- Der dargestellte Automat versendet die Daten ohne Paritätsbit.
- Der Sender ist durch einen Operationsablaufgraphen beschrieben, in dessen Knoten ein Zähler gesteuert und an dessen Kanten der Zählstand mit ausgewertet wird. Die Automatenbeschreibung könnte auch so abgewandelt 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 des abgetasteten Empfangssignals »RxD_del«. Das Signal »RxD« wird dabei mit der mehrfachen, im Beispiel mit der 16-fachen 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 Stoppbit 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 (Abbildung 3):

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

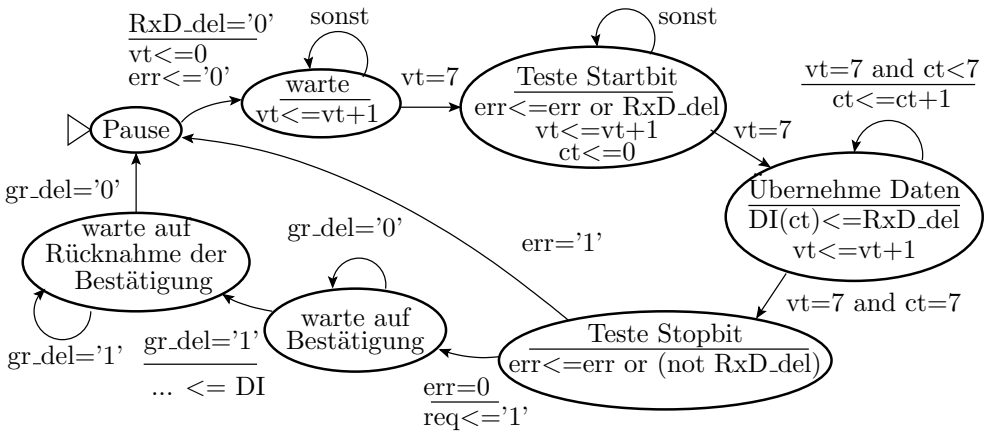

Abbildung 3: Empfänger mit Handshake als Ablaufgraph

Anmerkungen:

- Kein Paritätsbit
- vt ist ein Zähler modulo 16
- ct muss wie beim Sender vom Typ 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 des Beispielentwurfs

Laden Sie die Dateien a3_rs232.xise, a3_rs232.vhd und a3_rs232.ucf aus dem Netz herunter und kopieren Sie sie in ein Verzeichnis in ihrem home-Verzeichnis. Öffnen Sie dann das Projekt in ISE.

1. Rekonstruieren Sie den Graphen des Empfangsautomaten aus der Datei a3_rs232.vhd und zeichnen Sie ihn auf das Abgabeblatt. Vergleichen Sie den gefundenen Graphen mit dem oben beschriebenen Graphen.

   (a) Empfang mit oder ohne Auswertung des Paritätsbits?
   (b) Welche Baudrate ist eingestellt?
   (c) Enthält der Automat eine Handshake-Funktion?
   (d) Wie funktioniert die eingebaute Echofunktion?

2. Übersetzen Sie den Entwurf und programmieren Sie den Schaltkreis.

3. Starten Sie auf dem PC ein Terminalprogramm, unter Linux »GtkTerm«

   (a) Anwendungen => Acessories => Serial port terminal
   (b) Einstellen der Übertragungsparameter über das Menü »Configuration => Port öffnen«.
   (c) Überprüfen der Einstellungen: Port: /dev/ttyS0, Speed: 9600 Bits pro Sekunde, Parity: even, Bits: 8, Stopbits: 1, Flow control: none

   Under Windows use »putty«

- »Start« ▷ »All Programs« ▷ »...« ▷ »Putty« ▷ »serial« auswählen ▷ Schnittstelleneinstellungen über »Connection ▷ Serial« vornehmen ▷ »Open«

4. Verbindung Testen: Nacheinander die Ziffern 0 bis 9 senden und kontrollieren, dass auf led(7 downto 0) die zugehörigen ASCII-Codes 30h bis 39h angezeigt werden.

3 Entwurfsaufgaben

1. Schließen Sie den Logikanalysator an die Pins "DB0" (RxD) und "ADR0" (TxD) auf der Ansteckbaugruppe an Stecker A2 an. Laden Sie die Datei a3_rs232.xml aus dem Netz herunter. Tragen Sie in die Datei eine sinnvolle Abtastrate und Triggerbedingung ein, um die Übertragung vom PC zur Baugruppe aufzeichnen zu können. Begründen Sie Ihre Wahl der Abtastrate.

2. Entwerfen Sie einen Sendeautomaten, der bei Betätigung der Taste BTN0 das Byte SW(7 downto 0) an den PC sendet. Achtung: Entprellung nicht vergessen! Schreiben Sie den Sendeautomaten in eine neue Datei und verwenden Sie folgende Entity-Beschreibung:

```
entity a3_rs232_sender is
  port(
    clk_16x_baud : in  std_logic;
    txd          : out std_logic;
    data         : in  std_logic_vector(7 downto 0);
    req          : in  std_logic;
    grant        : out std_logic
  );
end entity;
```

   Simulieren Sie ihren Automaten mit Hilfe des Testrahmens a3_rs232_tb.vhd aus dem Netz und testen Sie die Schaltung auf der Baugruppe. Der Logikanalysator kann zur Problemfindung verwendet werden.

3. Modifizieren Sie die Empfängerschaltung so, dass auf den beiden linken 7-Segmentziffern das letzte empfangene Byte als 2-stellige Hexadezimalzahl dargestellt wird. Ergänzen Sie die Überprüfung des Paritätsbits im Zustand 9 des Automaten und geben sie einen möglichen Fehler auf LED11 der "Ampelsteuerung/Zahlenschloss"-Ansteckbaugruppe am Stecker B1 aus.

4 Abnahmekriterien

- Zeichnungen des Sende- und Empfangsautomaten auf dem Abgabeblatt
- Vorführung der Messergebnisse des Logikanalysators
- Simulation und Vorführung des Sendeautomaten
- Änderungen am Empfangsautomaten.