

02 Serielle Schnittstelle

1 Ziel

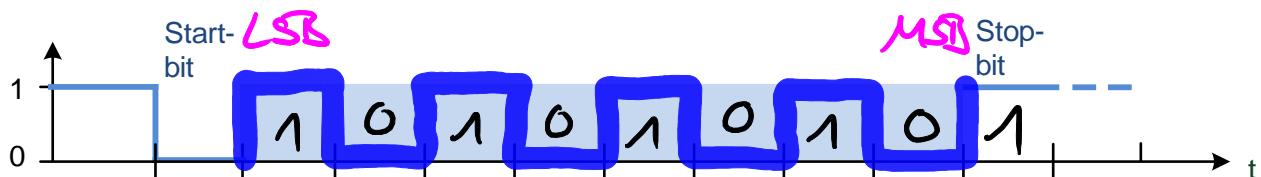
- Einarbeitung in die Bedienung eines Oszilloskops.
- Bestimmung der wichtigsten Parameter und Einrichten einer asynchronen seriellen Schnittstelle.

Hintergrund: Die asynchrone serielle Schnittstelle wird im professionellen Bereich nach wie vor häufig verwendet, um Geräte (z.B. Router oder Embedded Controller) zu konfigurieren.

- 2 Vorbereitung** ~~* 9. Variante Start-Bit gehört auf Octo = 1011; E~~
~~115'200 : 10 = 11520~~ ~~$\frac{1}{11520} = 0.0000\ 868 = 86,8\ \text{Mikrosek}$~~
- Studieren Sie den Anhang A zur asynchronen seriellen Schnittstelle und beantworten Sie folgende Fragen für eine Schnittstelle mit 115'200 Baud; 8 Daten-Bit, ohne Parity und einem Stop-Bit:
 - Wie gross ist die Bit-Dauer T? ~~8 Daten-Bit + 1 Stop-Bit = 9 Bit~~

~~* $115'200 : 9 = 11'800$ $\frac{1}{11'800} = 0.0000\ 78125 = 78,125\ \text{Mikrosek.}$~~

- Zeichnen Sie eine Signalfolge ein, bei der sich 1- und 0-Pegel abwechseln.



- Welche Frequenz hat das Rechtecksignal ausgedrückt durch T und als numerischer Wert in Hz?

~~$HZ = \frac{1}{T} = 0.000078125 = 11.800 HZ$ * $0.000086805 = 11520,07 HZ$~~

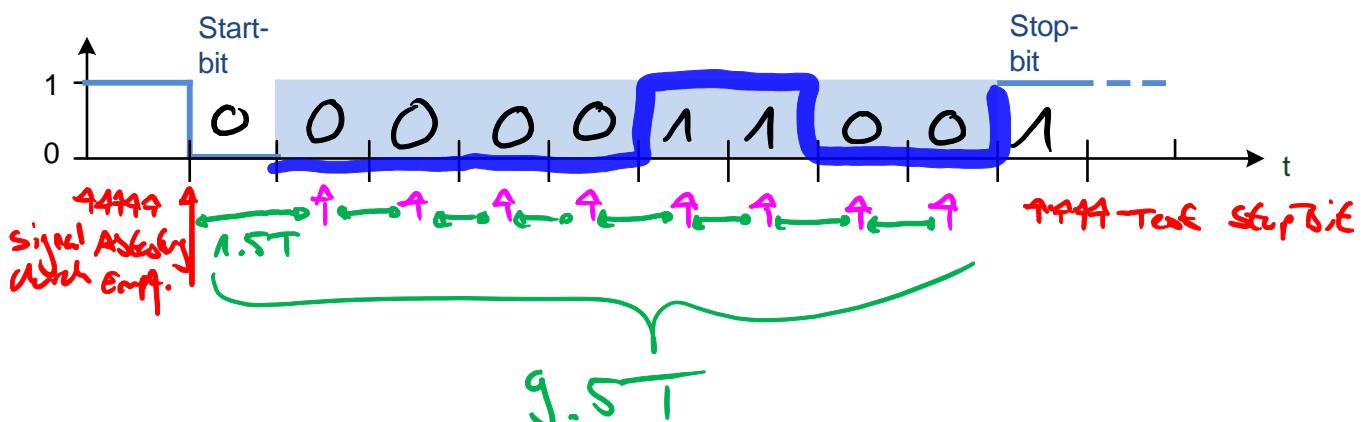
- Welchem Zeichen entspricht diese Signalfolge?

~~1|01010101010 = Dec 85 ASCII: 'U'~~

- Wählen Sie in der ASCII-Tabelle ein beliebiges Zeichen aus, bestimmen Sie den Binärwert und zeichnen Sie das Zeitdiagramm der Übertragung in der folgenden Grafik ein.

Zeichen: ~~O~~

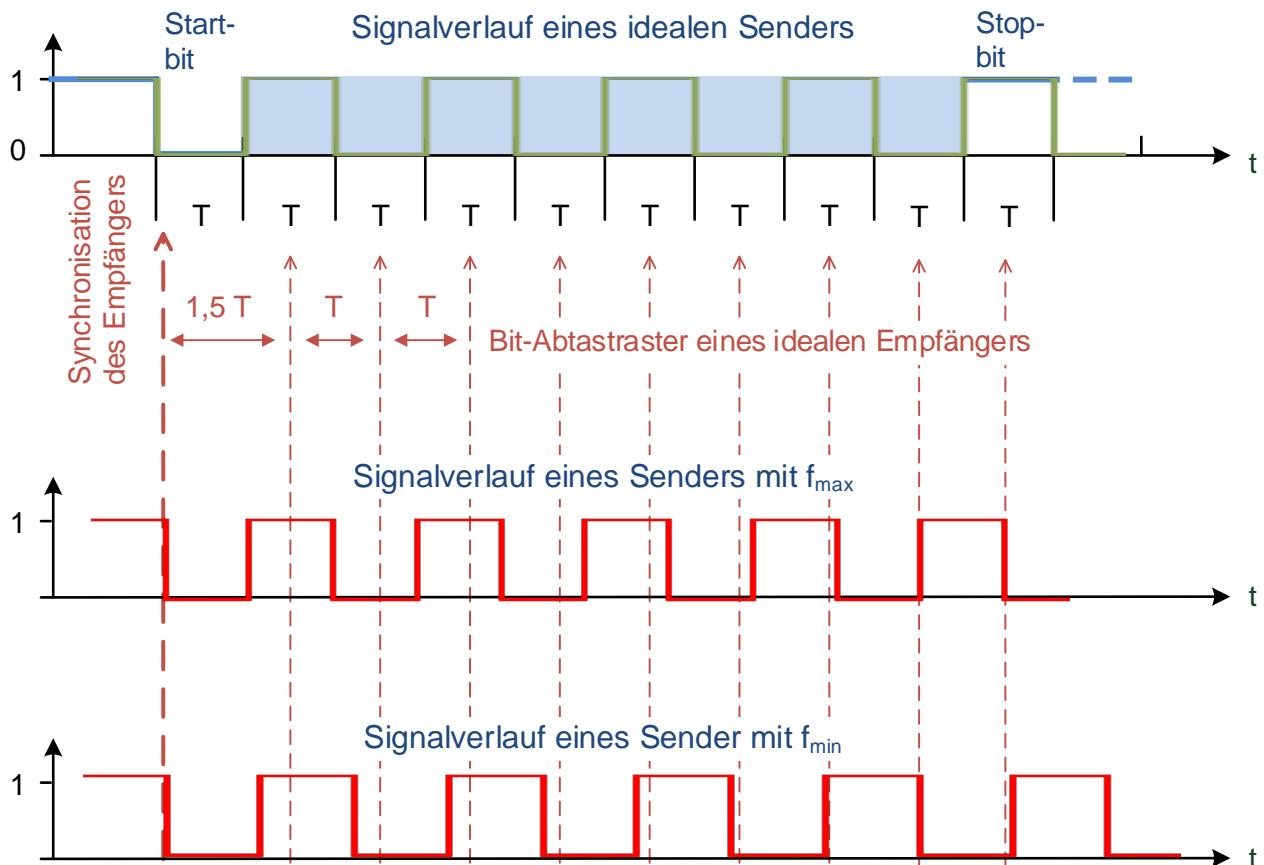
ASCII-Wert: ~~Dec: 48 = 00110000~~



- Wie gross darf der Fehler des Senders sein, damit ein idealer Empfänger noch richtig funktioniert?
Formulieren Sie die Bedingung!

max. 0.5T auf 0.5T (siehe oben)

- Die folgende Grafik zeigt die beiden Grenzsituationen. Geben Sie die zugehörigen minimalen und maximalen Frequenzen an.



$$f_{\max} = \frac{2B}{10 \cdot 10.5} = 2 \cdot 115 \cdot 200 : 10 \cdot 10.5 = 241.920$$

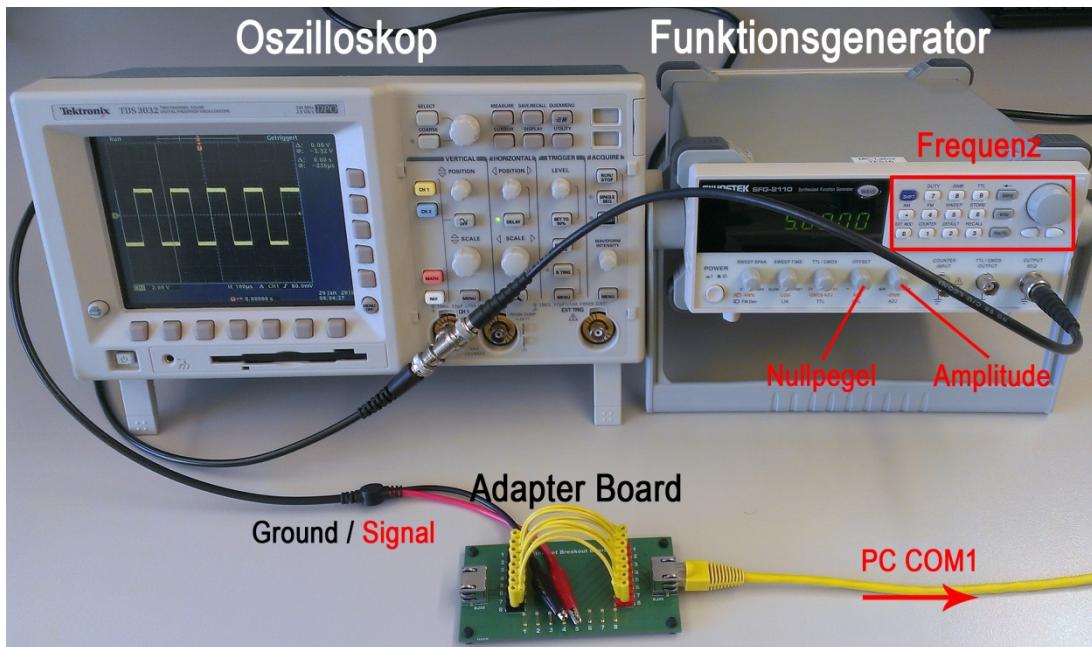
$$f_{\min} = \frac{2B}{10 \cdot 9.5} = 2 \cdot 115 \cdot 200 : 10 \cdot 9.5 = 218.880$$

Zeigen Sie Ihre Vorbereitungen dem Laborbetreuer.



3 Praktikum

- Schliessen Sie den Funktionsgenerator an das Oszilloskop an. Falls Sie noch nie mit einem Oszilloskop gearbeitet haben, lassen Sie sich die Bedienung erklären.
- Stellen Sie mit Hilfe des Oszilloskops ein Rechtecksignal ein, das die richtigen Signal-Pegel für eine RS232-Schnittstelle und die richtige Frequenz für 115'200 Baud hat (siehe Abschnitt 2).



- Starten Sie einen PC unter Linux.
Führen Sie Signal und Ground des Funktionsgenerators über ein Adapter-Board auf die COM1-Schnittstelle eines PCs. Auf dem Adapter-Board sind folgende Signale vorhanden (Bezeichnungen sind aus Sicht des PCs zu lesen):

RJ45	Abkürzung	Beschreibung
1	DSR	Data Set Ready
2	DCD	Data Carrier Detect
3	DTR	Data Terminal Ready
4	GND	Signal Ground
5	RD	Received Data
6	TD	Transmitted Data
7	CTS	Clear To Send
8	RTS	Request To Send

Starten Sie auf dem PC das Terminal-Emulations-Programm putty und überprüfen Sie die Einstellungen (COM1, 115'200 Baud, 8 Daten-Bit, keine Parity-Bit, 1 Stop-Bit, keine Flow-Control).

- Variieren Sie am Drehknopf die Frequenz in beiden Richtungen, bis der Empfang nicht mehr richtig funktioniert. Bestimmen Sie aus den Grenzwerten die Bitzeiten und vergleichen Sie diese mit Ihren Berechnungen.

Mit der Hälfte der Baud-Rate wird das "U" in Putty ausgegeben. Wichtig ist dabei, dass Putty entsprechend konfiguriert ist. und die Function auf Rechteckige Signale entsprechend 53.6



Erklären Sie die Abweichungen der Messung dem Dozierenden.

- Verkleinern Sie am Drehknopf die Amplitude bis der Empfang nicht mehr richtig funktioniert. Bestimmen Sie den Grenzwert und vergleichen Sie diesen mit dem Standard.

1.809 ist noch in Ordnung, sobald man unter diesem Wert ist, kann er nicht mehr richtig unterscheiden ob es eine 0 oder 1 ist

- Entfernen Sie den Funktionsgenerator. Verbinden Sie stattdessen den Konsolen-Eingang (ttyS0) einer Embedded Linux Box (ELB) über das Adapter-Board mit der COM1-Schnittstelle des PCs.

Wir wollen nur das Echo sehen aber keine Ausgaben der Shell. Darum starten wir auf der ELB ein Programm, das nichts macht, ausser Zeichen von der Konsole entgegenzunehmen:

```
cat > /dev/null
```

- Drücken Sie auf der Tastatur das Zeichen, das Sie in der Vorbereitung Abschnitt 2 gewählt hatten und zeichnen Sie den Signalverlauf mit dem Oszilloskop auf. Vergleichen Sie die gemessene Kurvenform mit dem von Ihnen gezeichneten Zeitdiagramm. Gibt es Unterschiede?

umgekehrt

- Vergleichen Sie TD und RD beim Drücken der <Enter> Taste. Erklären Sie, warum diese nicht gleich sind.



Weitere Ideen (eigene sind erwünscht):

- Schliessen Sie das Oszilloskop an, so dass Sie TD und RD aufzeichnen können.
- Messen Sie den Delay zwischen TD/RD.
- Vergleichen Sie die Signalpegel des PC und der Embedded Linux Box.
- Stellen Sie die Bitrate um (Programm stty und Terminal-Einstellungen).
- Verbinden Sie 2 PCs und erstellen Sie mit Hilfe des Adapter-Boards selber eine Null-Modem-Verbindung zwischen zwei COM-Schnittstellen (kann auf demselben PC sein).
- Untersuchen Sie die Flow-Control-Mechanismen z.B.
 - Öffnen Sie z.B. auf dem Adapter-Board alle Leitungen außer 4, 5 und 6. Überprüfen Sie die Funktion mit und ohne Flow-Control (siehe Terminal-Einstellungen).
 - Testen Sie Soft-Flow-Control (mit Steuer-Zeichen)
 - Testen Sie Hard-Flow-Control (mit Signal-Leitungen)

Anhang: Serielle asynchrone Übertragung

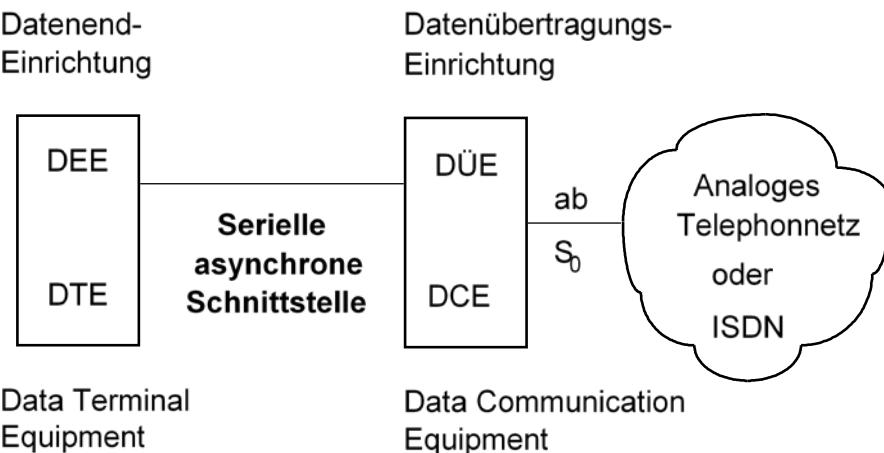
A.1 Typische Einsatzgebiete

Typische Einsatzgebiete für serielle asynchrone Schnittstellen sind:

- Anschluss von Modems und ISDN-Adaptern an eine DTE
- Anschluss von Terminals an einen Rechner
- Einfache PC-PC-Verbindungen
- Feldbusse in der Automatisierungstechnik
- Anschluss von Endgeräten an den PC (Drucker, Digitalkameras usw.)
- Verbinden von Systemeinheiten in Embedded Systems wie Mikrokontroller, Modems, Drucker, GPS usw.

A.2 Herkunft

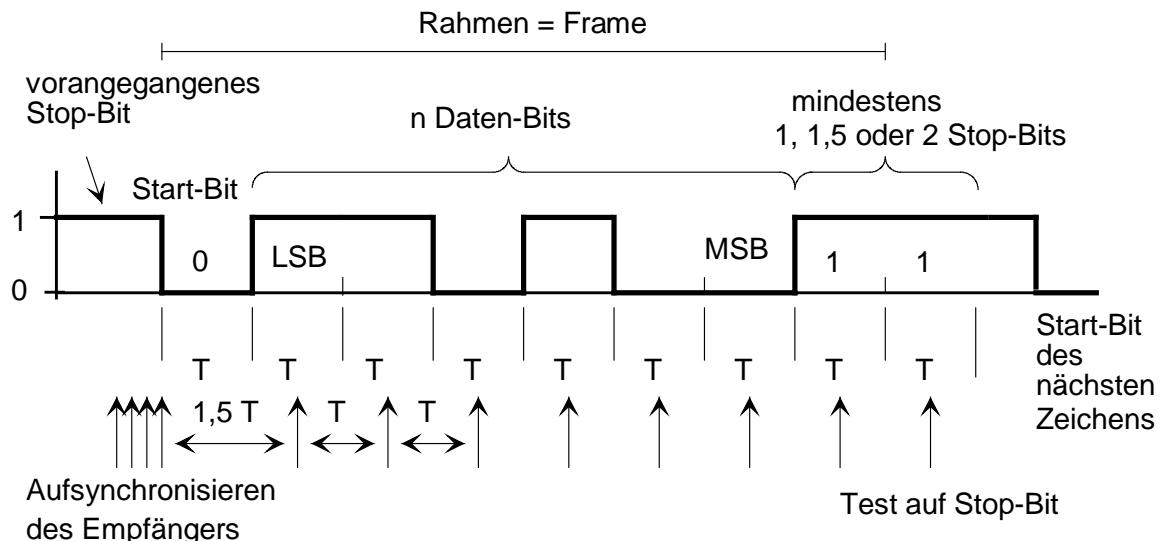
Serielle asynchrone Schnittstellen wurden ursprünglich entwickelt, um Datenübertragungseinrichtungen (z.B. Modems) an Datenendeinrichtungen (z.B. Terminals, PCs) anzuschliessen. Daraus sind die Eigenschaften der Schnittstellen auf diese Anwendung ausgelegt.



Mit der starken Verbreitung von PC und Embedded Systemen wurden solche serielle asynchrone Schnittstellen häufig für andere Zwecke, wie z.B. zum Verbinden zweier Rechner und der Anschaltung anderer Systemeinheiten benutzt.

A.3 Das Funktionsprinzip im Detail

Die Daten werden **seriell pro Byte** in der folgenden Reihenfolge übertragen: das **niederwertigste** Daten-Bit (LSB = Least Significant Bit) **zuerst**, das **höchstwertige** Daten-Bit (MSB = Most Significant Bit) **zuletzt**. Die Übertragung wird als **asynchron** bezeichnet, **weil kein Takt** für die Bitsynchronisation übertragen wird.



Der Sender und der Empfänger besitzen eigene, unabhängige Taktquellen mit annähernd gleicher Frequenz. Jedes übertragene Zeichen wird separat mit Hilfe eines Start- und Stop-Bits synchronisiert. Für jedes Zeichen sendet der Sender ein Start-Bit, n Daten-Bits und zum Abschluss 1-2 Stop-Bits.

Am Anfang jedes Zeichens synchronisiert sich der Empfänger auf den Sender. Dazu wird die Leitung mit einer höheren Frequenz als der Bitrate abgetastet, bis die fallende Flanke des Start-Bits erkannt wird. Dann werden die n Daten-Bits asynchron transferiert, d.h. der Empfänger muss die Sendefrequenz kennen. Kleine Taktdifferenzen zwischen Sender und Empfänger werden durch Stop-Bits aufgefangen. Zwischen Sender und Empfänger müssen Taktfrequenz, Anzahl übermittelte Daten-Bits und Anzahl Stop-Bits abgemacht sein.

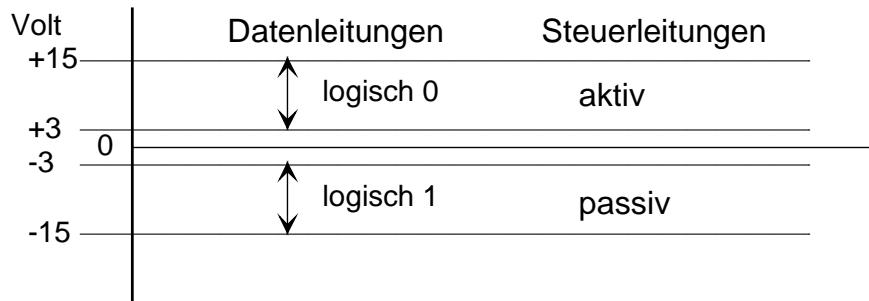
Die n Daten-Bits bestehen aus Nutzdaten und in vielen Fällen aus einem zusätzlichen Parity-Bit für die Erkennung von einfachen Übertragungsfehlern. Der Sender bildet eine Parity über die Nutzdaten (Even oder Odd Parity) und überträgt dieses zusätzliche Bit als letztes Bit. Der Empfänger überprüft das Parity-Bit der empfangenen Nutzdaten mit dem selbst gebildeten Parity-Bit. Zwischen Sender und Empfänger muss natürlich abgemacht sein, ob mit geradem oder ungeradem Parity-Bit gearbeitet wird.

A.4 Die Schnittstelle RS232D

EIA-RS232 bezeichnet Normen der EIA (Electronic Industries Association). RS bedeutet Recommended Standard. Sie beinhalten die mechanischen, die elektrischen und die funktionalen Eigenschaften. D bezeichnet die Version. Die EIA-RS232 Schnittstelle ist elektrisch identisch mit der ITU-T-Norm V.28.

a) Elektrische Eigenschaften

Bei RS232D werden für die Datenleitungen Spannungspiegel von +3...+15 Volt dem logisch Null und -3...-15 Volt dem logisch Eins zugeordnet. Für die Steuer- und Meldeleitungen ist diese Zuordnung umgekehrt. Die symmetrischen Spannungspiegel bewirken, dass der DC-Signalanteil klein gehalten wird. Die hohe Pegeldifferenz ergibt einen grossen Störabstand. Auf diese Art werden Leitungslängen von 10-100 m und Übertragungsraten von 1-20 KBit/s erreicht (offiziell 15 m, 20 KBit/s).



Ein Nachteil ist, dass spezielle Treiber- und Empfänger-Bausteine und die Spannungen +12 und -12 Volt verwendet werden müssen. Es gibt allerdings Bausteine, welche die notwendigen Spannungen intern aus den 5 Volt generieren (z.B. MAX 232).

b) Mechanische Eigenschaften

In der Norm RS232D wird ein 25poliger Cannon-Stecker verwendet. In der Praxis wird jedoch meist ein 9poliger Cannon-Stecker (Subminiatur D male) verwendet, der mit den PCs als defakto Standard eingeführt wurde.

		Signale und Stecker beim PC	
Data Carrier Detect	(DCD)	1 ○	Data Set Ready (DSR)
Receive Data	(RD)	2 ○	Request to Send (RTS)
Transmit Data	(TD)	3 ○	Clear to Send (CTS)
Data Terminal Ready	(DTR)	4 ○	Calling Indicator/Ring Indicator (RI)
Signal Ground		5 ○	

c) Funktionale Eigenschaften der Schnittstellen

In der Praxis werden meistens nur zwischen drei und neun dieser Signale verwendet. In der folgenden Tabelle sind die neun wichtigsten dieser Signale dargestellt:

Übertragungsleitungen:

Transmit Data (TD): Datensendeleitung von der DTE zur DCE.

Receive Data (RD): Datenempfangsleitung von der DCE zur DTE.

Flow-Control-Signale:

Request to Send (RTS): Anforderung des DTE an das DCE sich für den Sendebetrieb vorzubereiten.

Clear to Send (CTS): Mit diesem Signal zeigt das DCE an, dass der Sender eingeschaltet ist und dass sie bereit ist, Daten zu senden.

Modem-Control-Signale:

Data Terminal Ready (DTR): Zeigt an, dass das DTE betriebsbereit ist.

Data Set Ready (DSR): Zeigt an, dass das DCE betriebsbereit ist.

Data Carrier Detect (DCD): Mit diesem Signal zeigt das DCE an, dass ein Trägersignal empfangen wird.

Ring Indicator (RI): Meldung des DCE an die DTE, dass ein Rufsignal anliegt.

A.5 Verbinden von Rechnern über serielle Schnittstellen

Da diese Schnittstellen-Normen auch im Nahbereich verwendet werden, fehlt in vielen Fällen die Datenübertragungseinrichtung (z.B. ein Modem) und es werden zwei Rechner (DTE) direkt zusammengeschaltet. In diesem Fall müssen bestimmte Leitungen wie z.B. Sende- und Empfangs-Datenleitungen ausgekreuzt werden.

Beim Zusammenschalten unterschiedlicher DTEs kommt es vor, dass das eine Gerät bestimmte Signale nicht besitzt, das andere Gerät diese Signale aber prüft. In solchen Fällen werden die Signale so beschaltet, dass das prüfende Gerät einen plausiblen Zustand vorfindet.

Damit die Datenübertragung z.B. zwischen zwei PC ohne Datenverluste funktioniert, muss eine Flussteuerung realisiert werden. Dazu gibt es 2 Möglichkeiten:

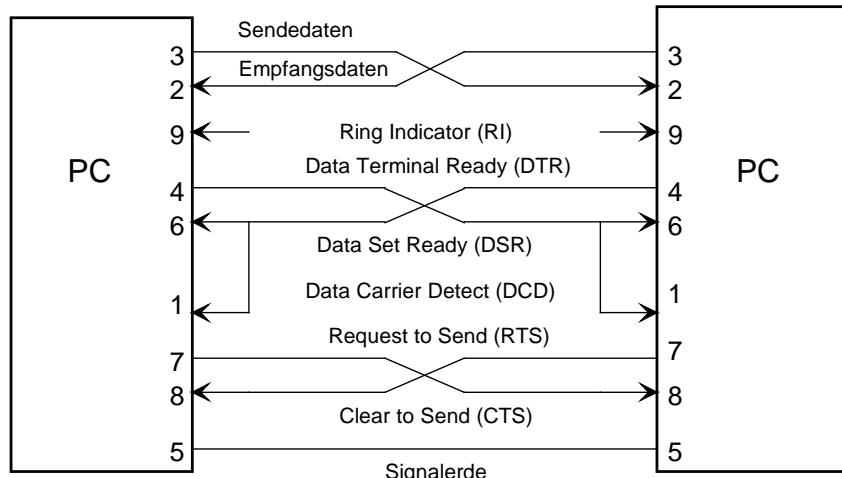
1. Die Fluss-Steuerung zwischen Sender und Empfänger wird hardware-mässig über die vorgesehenen Leitungen realisiert.
2. Sender und Empfänger realisieren die Fluss-Steuerung software-mässig mit Meldungen.

a) Hardwaremässige Fluss-Steuerung

Die Signale müssen wie folgt verbunden werden (Stecker-Nummern beziehen sich auf den 9poligen Stecker des PC):

- Sende- und Empfangsleitungen müssen ausgekreuzt werden.
- Das Signal "Data Terminal Ready" wird mit den Signalen "Data Set Ready" und "Data Carrier Detect" des Partners verbunden.
- Das Signal "Request to Send" wird mit dem Signal "Clear to Send" des Partners verbunden.

Eine solche Verbindung wird als "**Nullmodem**" bezeichnet.



b) Software-mässige Fluss-Steuerung

Anstelle der hardwaremässigen Steuerung mit den oben beschriebenen Signalen ist es auch möglich, die Steuerung von Sender und Empfänger softwaremässig zu realisieren. Dazu werden die ASCII-Zeichen XON (<Ctrl>Q) und XOFF (<Ctrl>S) verwendet.

Mit XON (DC1, Wert 11h) meldet der Empfänger, dass er bereit ist Daten zu empfangen und mit XOFF (DC3, Wert 13h), dass er bis auf Widerruf keine weiteren Daten entgegennehmen kann.

Die Hardware-Steuerleitungen sind nicht notwendig.

