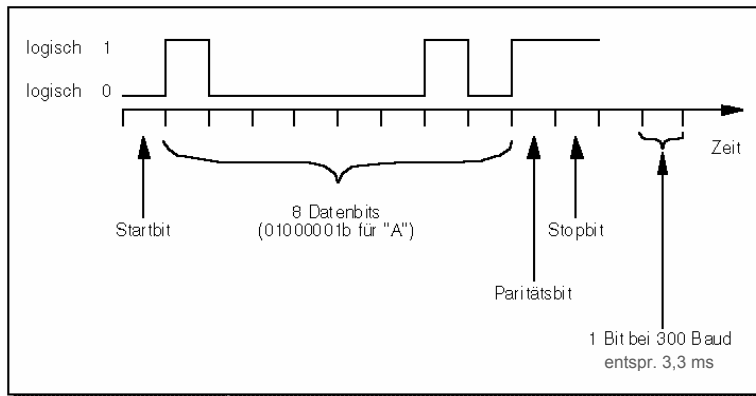


2. Anmerkungen zu seriellen Kommunikation

Die Kommunikation über die serielle Schnittstelle (RS-232) erfolgt i.d.R. *asynchron*, d.h. die Synchronisation bzw. Taktung muss bei der seriellen Übertragung deshalb auf der Datenleitung durch die Daten selbst erfolgen. Einem zu übertragenden "Daten-Wort" werden deshalb Steuerinformationen gemäß dem RS-232-Protokoll voran- und nachgestellt. Diese eingebetteten Synchronisationsinformationen bestehen aus einem so genannten *Startbit*, das den Beginn einer Dateneinheit, und mindestens einem *Stopbit*, welches das Ende einer Dateneinheit angibt.

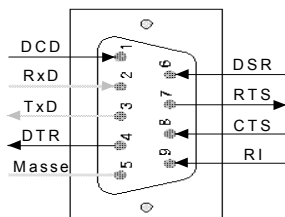


*Das asynchrone Protokoll der seriellen Übertragung
(8 Datenbits, 2 Stopbits, ungerade Parität)*

Zur Kommunikation über die serielle Schnittstelle wird eine speziell hierfür entwickelte Klasse (Serial) bereitgestellt werden, die es ermöglicht, über die serielle Schnittstelle Daten auszutauschen.

Der Anschluss

Als Anschluss für die serielle Schnittstelle, auch als V24- und RS232C-Schnittstelle bezeichnet, sind i.d.R. 9-polige Sub-D-Steckverbindungen gebräuchlich.



Steckerbelegung der seriellen Schnittstelle

Leitung	Kürzel	Bedeutung
Transmitted Data	TxD	Datensendeleitung
Received Data	RxD	Datenempfangsleitung
Request To Send	RTS	Über diesen Handshake-Ausgang teilt der Empfänger dem Sender mit, ob er bereit ist, Daten zu empfangen.
Clear To Send	CTS	(Modem) diese Leitung auf logisch 1, sobald es zum Empfang der Daten bereit ist.
Data Set Ready	DSR	die Aktivierung dieses Signal mit, dass er betriebsbereit ist. Bei lokalen Verbindungen dient dieser Eingang (anderen Seite)
Data Terminal Ready	DTR	gang informiert der Empfänger; dass er betriebsbereit ist. Bei lokalen Verbindungen ist dieser Anschluss üblicherweise mit der DSR-Leitung

Schnittstellen-Steuerung

- ☐ Bei jeder Art von Datenaustausch sind für eine erfolgreiche Kommunikation Absprachen über den Ablauf des Datentransfers notwendig.
- ☐ Um eine verlustfreie Datenübertragung zu ermöglichen müssen sich Sender und Empfänger darüber einigen, wann Daten gesendet werden.
- ☐ Im Allgemeinen unterscheidet man *Software-Handshake* und *Hardware-Handshake*.

Definition

Verfahren, die den Ablauf der Datenkommunikation regeln, werden als **Protokollverfahren** bezeichnet. Mit ihnen werden die zeitliche und inhaltliche Steuerung des Datenstroms festgelegt.

Neben der zeitlichen und inhaltlichen Synchronisation des Datenaustausches für eine Kommunikation überhaupt sind weitergehende Aufgaben zu erfüllen:

- ☐ Gewährleistung einer fehlerfreien Übertragung bzw. das Ermöglichen der Fehlererkennung
- ☐ Quittierung von ankommenden Daten
- ☐ Vermeiden des Überlaufes von Daten im Empfänger

Beispiel 1: XON/XOFF-Protokoll

Das *XON/XOFF-Protokoll* ist ein einfaches Protokollverfahren zur Vermeidung von Überlauf Fehlern bei der seriellen Datenübertragung.

- ☐ Die Synchronisation wird mit Hilfe der Steuerzeichen "XON" und "XOFF" realisiert

	ASCII	Tastatur	hex	Wirkung
XON	DC1	CTRL-Q	11	DÜ einschalten
XOFF	DC3	CTRL-S	13	DÜ abschalten

- ☐ In der minimalen Beschaltung benötigt man nur eine Zweidrahtverbindung (Abb. 2.3.3)

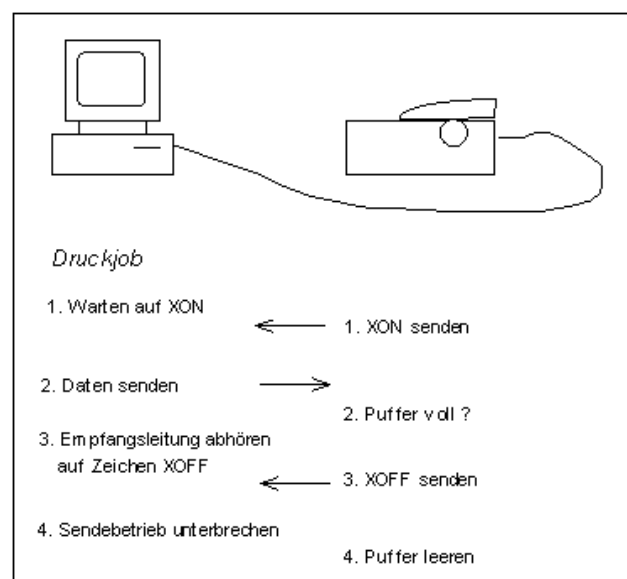
Ablauf:

Wird ein am Computer angeschlossener Drucker auf *on line* geschaltet oder wird er vom Computer zurückgesetzt (reset), so sendet der Drucker das Zeichen XON zum Computer.

Der Computer sendet daraufhin die Daten in der eingestellten Übertragungsgeschwindigkeit. Der Drucker fängt schon während der Datenübertragung an zu drucken.

Ist die Übertragungsgeschwindigkeit größer als die Druckgeschwindigkeit, so füllt sich der Puffer im Drucker. Wird ein bestimmter Füllstand im Puffer erreicht (oder wird der Drucker auf *off line* geschaltet), sendet der Drucker das Zeichen XOFF. Der Computer stoppt daraufhin den Datentransfer.

Der Drucker kann nun in Ruhe aus dem Puffer drucken. Wenn der Druckpuffer fast leer ist sendet der Drucker wieder XON zum Computer und der Datentransfer wird fortgesetzt. Dieser Vorgang wiederholt sich solange, bis der Rechner alle Daten ausgegeben sind.



Beispiel 2: ETX/ACK-Protokoll

- ❑ Im *ETX/ACK-Protokoll* werden die Daten in Form von Blöcken übergeben.
- ❑ Die Länge der Datenblöcke ist abhängig von der Größe des Empfangspuffers.
- ❑ Der Sender schickt einen Datenblock, gefolgt von dem Zeichen **ETX** (*end of text*, ASCII-Code 03).
- ❑ Das Ausgabesystem wartet mit der Übertragung des nächsten Datenblockes, bis der Empfänger mit dem Zeichen **ACK** (*acknowledge*, ASCII-Code 06) den Empfang bestätigt hat.
- ❑ Im Gegensatz zum XON/XOFF-Protokoll wird hier eine zusätzliche Steuerleitung eingesetzt.

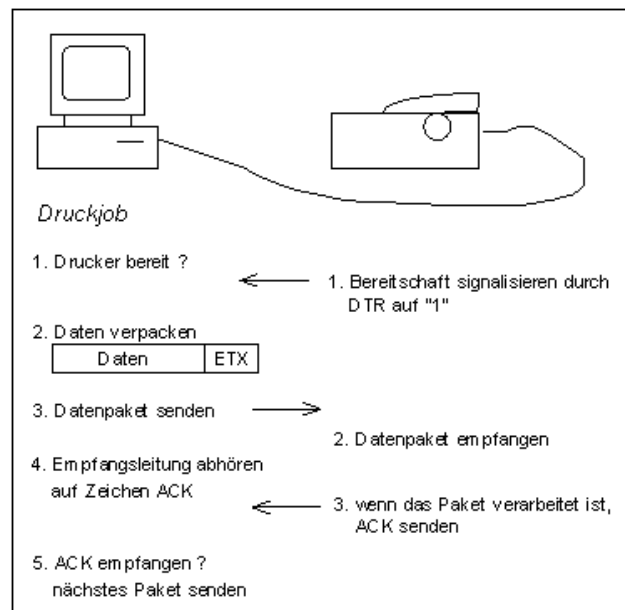
Ablauf:

Ist der Empfänger zur Datenübernahme bereit, signalisiert er diesen Zustand der Gegenstation, indem er seine Ausgangssteuerleitung DTR auf High-Pegel setzt. Gleichzeitig sendet der Empfänger das Steuerzeichen ACK (06h).

Der Sender übermittelt daraufhin das Datenpaket, das mit dem ASCII-Zeichen „ETX“ (03h) abgeschlossen ist.

Nachdem der Empfänger die Daten verarbeitet hat, meldet er seine erneute Empfangsbereitschaft, indem er sich mit „ACK“ zurückmeldet.

Der Sender kann nun das nächste Datenpaket übermitteln.



Beispiel 3: READY/BUSY-Protokoll

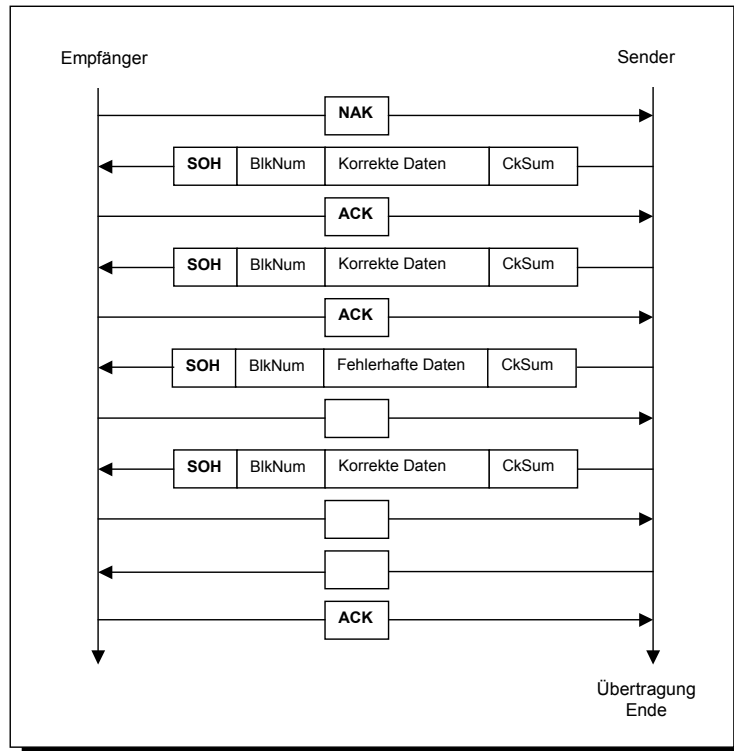
- ❑ Dieses Protokoll benutzt zur Steuerung des Datentransfers Steuerleitungen. (Abb. 2.3.2)
- ❑ Ist der Empfänger als DCE (z.B. Modem) geschaltet, so wird das Leitungspaar **DSR** (*data set ready*) und **DTR** (*data terminal ready*) benutzt. Handelt es sich bei beiden den Kommunikationspartnern um ein DTE (Rechner), so wird das Leitungspaar **RTS** (*request to send*) und **CTR** (*clear to send*) benutzt.
- ❑ Das entsprechende Signal ist im EIN-Zustand, wenn das Gerät bereit (*ready*) ist, d.h. der Empfänger ist in der Lage, Daten zu empfangen. Ist der Empfänger nicht mehr bereit (*busy*), wird das Signal in den AUS-Zustand gebracht. Daraufhin muss der Sender den Datentransfer stoppen und so lange warten, bis das Signal wieder in den EIN-Zustand geht.

Ablauf:

Ist der Sender (Rechner I) zur Datenübertragung bereit, so meldet er dies dem Empfänger (Rechner II), indem er seine Steuerleitung RTS in den EIN-Zustand versetzt. Der Empfänger registriert diesen Zustand und zeigt seine Empfangsbereitschaft durch den EIN-Zustand seiner RTS-Leitung an. Der Sender kann dann mit der Übertragung beginnen, wenn er den EIN-Zustand auf seiner CTS-Leitung erkennt. Sobald der Empfänger sein RTS-Signal zurücknimmt (RTS = AUS), unterbricht der Sender die Übertragung und wartet auf die erneute Empfangsbereitschaft.

Beispiel 4: Das XModem-Protokoll

Über der Byte-Ebene sitzt die Block-Ebene, mit deren Hilfe Datenblöcke vom Sender zum Empfänger gesandt werden. Dabei handelt es sich um ein reines Software-Protokoll, denn von der Hardware macht sich diese Ebene unabhängig, indem sie auf die Sende- und Empfangsroutinen der Byte-Ebene zurückgreift.



Das Kommunikationsprotokoll auf der Block-Ebene

Die Datenübertragung erfolgt blockweise. Zu einem solchen Block gehören immer drei Informationen: ein Erkennungszeichen (*Token*), das dem Block vorangeht und seinen Inhalt beschreibt, die *Länge des Blocks* und natürlich der *Datenblock* selbst.

Das Protokoll geht dabei von folgenden Token aus:

- SOH** = 01h **Start Of Header**: kennzeichnet, dass ein Block folgt
- EOT** = 04h **End Of Transmission**: kennzeichnet, dass die Übertragung beendet ist
- CAN** = 18h **CANcel Transfer**: kennzeichnet, dass die Übertragung abgebrochen wurde

Damit eine fehlerfreie Übertragung eines Blockes für den Empfänger auch erkennbar wird, sendet der Sender am Ende des Übertragungsblockes eine *Prüfsumme* (Checksum).

Aufbau eines Blocks:

SOH	BlockNr	255-BlockNr	Datenbereich	Checksum
-----	---------	-------------	--------------	----------

Wenn die Datenübertragung beendet ist, also die Datei vollständig übertragen wurde, wird vom Sender das Zeichen EOT anstatt des normalerweise erwarteten SOH-Zeichens gesendet.

Der Empfänger ermittelt aus den ihm übertragenen Daten eine Prüfsumme und vergleicht diese mit dem empfangenen Prüfsummenwert. Der Empfänger gibt dem Sender nach der Übertragung des Blockes ein Feedback in Form eines fest vereinbarten Zeichens. Er sendet dabei ein **ACK**-Zeichen (**Acknowledge**), wenn die Übertragung fehlerfrei war, bzw. ein **NAK**-Zeichen (**Non-Acknowledge**) im Fehlerfall.

Die Feedback-Zeichen des Empfängers:

- ACK** = 06h **Acknowledge**: kennzeichnet, dass ein Block fehlerfrei übertragen wurde
- NAK** = 15h **Non-Acknowledge**: kennzeichnet eine fehlerhafte Übertragung.