Steuerung der Datenkommunikation (der seriellen Schnittstelle) durch Standard-Protokolle

Bei jeder Art von Datenaustausch / von Datenkommunikation sind für eine erfolgreiche Kommunikation zwischen 2 Kommunikationspartnern (z.B. Sender und Empfänger über eine serielle Schnittstelle) Absprachen über den Ablauf des Datenaustausch notwendig, d. h. die **Vereinbarung eines Protokolls** muss festgelegt werden.

Ein **Protokoll** regelt ganz allgemein

- den Ablauf der Datenkommunikation insgesamt, u.a.
- durch die zeitliche und inhaltliche Steuerung des Datenstroms.

Viele Protokolle (bzw. korrekt funktionierende Datenkommunikationen) benötigen vor dem eigentlichen Datenaustausch (Inhalt, Nutzdaten) eine Abstimmung / ein Verfahren der 2 Kommunikationspartner, um den korrekten Datenaustausch sicherzustellen. Dieses Verfahren nennt man **Handshake-Verfahren** (oder kurz: **Handshake**).

Ein allgemeines Beispiel für eine Handshake-Verfahren:

Sender fragt Empfänger:	„Bist du bereit zum Empfangen?“	
Empfänger meldet Sender:	„Ja“	(oder „Nein“ ???)*
Sender meldet Empfänger:	„Dann sende ich jetzt 10 Zeichen. Ok?“	
Empfänger meldet Sender:	„Ok, ich warte auf Empfang.“	
Sender sendet:	10 Zeichen	
Empfänger meldet Sender:	„Ok, 10 Zeichen erhalten.“	
Sender fragt Empfänger:	„Bist du bereit zum Empfangen?“	
Empfänger meldet Sender:	… (usw.)	

** Das abgebildete Protokoll regelt nur den Best-Case, also wenn alles „glatt“ läuft, also alles „nach Plan“. Ein Protokoll muss aber auch die Worst - Case - Fälle beschreiben und die Behandlung der Fälle festschreiben. Im Beispiel z.B. den Fall, dass der Empfänger gar nicht mit „Ja“ sondern mit „Nein“ antwortet. Ein weiterer Worst - Case wäre der Fall, dass der Empfänger gar nicht antwortet.*

Man unterscheidet zwischen **Software-Handshake** und **Hardware-Handshake** und **Mischformen**.

Beim **Software-Handshake** sendet der Sender und Empfänger zur Steuerung des Datenflusses **spezielle (vereinbarte und eingefügte) Zeichen über die Datenleitungen TxD und RxD**. Entsprechend werden für die Datenübertragung bei einem Software-Handshake lediglich drei Leiteradern (RxD, TxD und gemeinsamer GND) benötigt. Der Hauptvorteil liegt also darin, dass keine gesonderten (zusätzlichen) Schnittstellenleitungen im Verbindungskabel erforderlich sind.

Manchmal sind vor der eigentlichen (Nutz-) Datenübertragung durchaus mehrfache Hin- und Herübertragungen von Handshakeinformationen (synonym auch Steuercodes genannt) notwendig bzw. sinnvoll. Die Prüfungen auf Empfang und Korrektheit der vereinbarten Zeichen (oder auch ganze Zeichenfolgen) muss über Softwareauswertung (= Programmcode) erfolgen.

Bekannte **Standardprotokolle** für ein **Software-Handshake-Protokoll** sind das **XON-XOFF-Protokoll** oder das **ETX-ACK-Protokoll**.

Beim **Hardware-Handshake** signalisieren sich die beiden Geräte (Sender und Empfänger) ihren jeweiligen Status über die bei der seriellen Schnittstelle (RS-232) vorhandenen zusätzlichen **Steuerleitungen RTS, CTS, DTR und DSR**.

Ein bekanntes **Standardprotokoll** für ein **Hardware-Handshake-Protokoll** ist das **Ready-Busy-Protokoll**.

Zusätzlich wird bei Hardware-Handshake - Protokollen meist auch ein Software-Handshake - Protokoll ergänzend mit eingesetzt, z.B. für die reine Datenübertragung, was das Protokoll dann aber zu einem kombinierten Protokoll (Mischform) macht.

Praxiseinsatz der Protokolle:

Die allermeisten Software- und Hardware-Handshake-Protokolle sind speziell für die modellierten Systemen vereinbart und sind damit auch spezielle Unikate.

Bewährte Standard-Protokolle können jedoch eine gute (weil bewährte) Vorlage liefern, um dann notwenige spezielle Anforderungen der bestimmten Systeme nur noch zu ergänzen.

Wenn Hardware-Handshake möglich ist, weil eine entsprechendes Verbindungskabel vorliegt bzw. die Schnittstelle entsprechende Ein- und Ausgänge verwalten kann, dann ist zu überlegen / zu entscheiden, ob man ein **kombiniertes Software-Hardware-Handshake-Protokoll (Mischform)** einsetzt, um die Datenleitungen zu entlasten, d.h. darüber dann nur Dateninhalte zu übertragen.

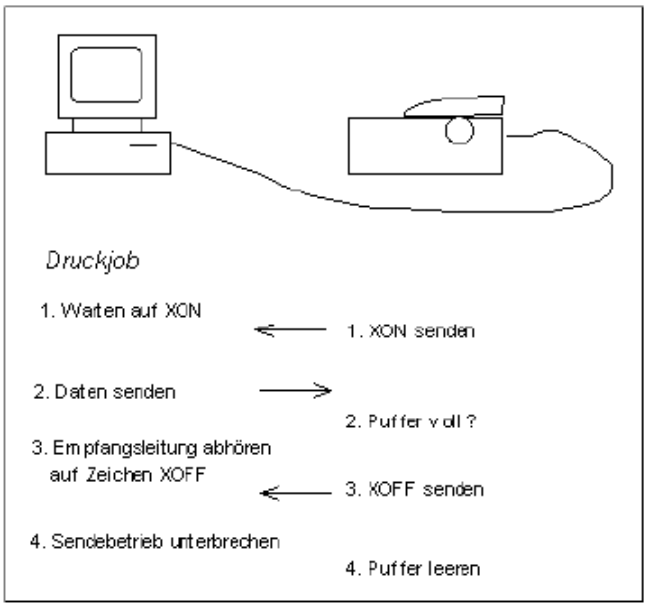
XON-XOFF-Protokoll (exemplarisch gelistet für ein Software-Handshake)

Das XON-XOFF-Protokoll ist ein Software-Handshake-Protokoll und ist eine Vereinbarung zur Übertragung von Text über die serielle Schnittstelle. Es ist ein einfaches **Protokollverfahren zur Vermeidung von Überlauf-fehlern der Pufferspeicher der Hardwareschnittstelle** (beim Sender und / oder Empfänger) bei der seriellen Datenübertragung. Eine 3-Draht-Verbindung reicht aus.

Beim **XON-XOFF-Protokoll** sendet der Empfänger zur Steuerung (= Synchronisation) des Datenflusses spezielle Zeichen an den Sender:

- XON = ASCII-Code 0x11 = 11h = 17 = DC1 = Device Control 1
- XOFF = ASCII-Code 0x13 = 13h = 19 = DC3 = Device Control 3

Das XON-XOFF-Protokoll ist (früher) im Zusammenspiel mit serieller Datenübertragung zwischen einem Computer (PC) und einem Drucker verwendet worden.



Ablauf des **XON-XOFF-Protokolls:**

Wird ein am PC angeschlossener Drucker auf online geschaltet, sendet er das **Zeichen XON** über die Datenleitung zum PC.

Der PC wartet auf das Zeichen XON und sendet bei Erhalt (in der eingestellten Übertragungsgeschwindigkeit) die Daten zum Drucker.

Der Drucker speichert die Daten in einen Pufferspeicher und beginnt gleichzeitig auch mit dem Drucken der Daten (aus dem Pufferspeicher).
Kann der Drucker die eintreffenden Daten nicht mehr (schnell genug) verarbeiten (der Pufferspeicher des Druckers ist (fast) voll), sendet er das **Zeichen XOFF** zum PC.

Der PC "hört" während seiner Daten-Sendung immer auch seine RxD-Leitung ab. Empfängt der PC das Zeichen XOFF, stoppt bzw. unterbricht er die (weitere) Sendung.

Der Drucker kann nun ohne weiteren Dateneingang erst mal die Daten aus dem Pufferspeicher drucken (= den Pufferspeicher leeren). Ist er wieder empfangsbereit, d. h. der Puffer ist leer, signalisiert er das wieder durch Senden von **Zeichen XON**.

Der Sender (PC) „hört“ während seiner Pause auf dieses Zeichen und sendet danach weiteren Text aus seiner Daten-Sendung, die er insgesamt drucken will.

Dieser Vorgang wiederholt sich solange, bis alle Daten gesendet und auch am Drucker ausgedruckt sind.

Ready-Busy-Protokoll (exemplarisch gelistet für ein Hardware-Handshake)

Die Ready-Busy-Prozedur (-Protokoll) ist die einfachste und in der Praxis wohl am häufigsten verwendete Prozedur.

Zur Signalisierung des Status "Ready"(bereit zum Datenempfang) oder "Busy" (das Gerät ist dann beschäftigt und kann gerade keine Daten mehr empfangen) wird im einfachsten Fall nur eine Leitung benötigt, nämlich DTR (Data Terminal Ready). Ist diese Leitung durch den Empfänger gesetzt (HIGH), so besteht Empfangsbereitschaft. Die Gegenstelle kann das an dem Eingang DSR „abhören“. DSR auf HIGH signalisiert demnach dem Sender die Empfangsbereitschaft des Empfängers. DTR nicht gesetzt (LOW) hingegen signalisiert den Busy-Status.

Nach diesem Protokoll tauschen sich Empfänger und Sender gegenseitig die Betriebsbereitschaft(en) aus. Außerdem dienen die korrespondierenden Leitungen RTS und CTS dazu den Datensende-Request des anzukündigen und vom Empfänger zu bestätigen. Das Datensenden vom Sender zum Empfänger wird durch das Signal DCD (Data Carrier Detect) angekündigt. Ein Handshake existiert hier dann nicht mehr. Es wird jetzt davon ausgegangen, dass der Empfänger empfangsbereit ist, wenn vorher DTR (Data Terminal Ready) aktiviert wurde.

Allgemeine Anwendung von Protokollen

Auch wenn bei der Beschreibung der Protokolle hier die serielle Schnittstelle benutzt wurde, sollte klar sein, dass Protokolle prinzipiell bei jeder Art der Datenübertragung und -kommunikation zur Anwendung kommen müssen.

Das beschrieben XON-XOFF-Protokoll ist z.B. also auch zwischen 2 anderen Kommunikationspartnern (z.B. 2 PC) und für ganz andere Aufgabenstellungen / Datenübertragungsarten (z.B. bei Client - Server - Systemen über Netzwerk und TCP/IP) anwendbar, und ist nicht auf die serielle Schnittstelle beschränkt.