

## 5. Praktikum

### 1. Aufgabe TSR-Treiber INT60h für Serielle Schnittstelle

Es ist ein TSR-Programm zu entwickeln, das durch seinen Aufruf im Speicher resident gemacht wird. Durch erneuten Aufruf soll das TSR-Programm aus dem Speicher entfernt werden. (einfache Überprüfung durch Kennung genügt)

Das TSR-Programm soll die Funktionen eines eigengestrickten Treibers für Senden und Empfangen mit der seriellen Schnittstelle ermöglichen.

Es sind sozusagen die Funktionen **01= Senden**, **02= Empfangen** und **03= Status** des BIOS INT 14h durch eigene und **natürlich bessere !!!** Funktionen des INT 60h nachzubilden (Software - Interrupt)

( Skript *Seriell* Seite 7 – 10 )

#### Funktionen des INT 60h Serielle Schnittstelle

Funktion	Fktn-Nr	Übergabe	Rückgabe	Bemerkung
Installations-Test	AH = 0	AL = 60h	AX = 6000h Treiber installiert und resident	
Zeichen senden	AH = 1	AL = Zeichen	AH.7 = 0 o.k. AH.7 = 1 Fehler AH.6-0 LSR-Status	Zeichen in AL bleibt erhalten
Zeichen empfangen	AH = 2		AL = Zeichen AH.7 = 0 o.k. AH.7 = 1 Fehler AH.6-0 LSR-Status AX= FF00h kein Zeichen empfangen	
Status abfragen	AH = 3		AH = LSR-Status AL bleibt erhalten	Zum Überprüfen ob Senden möglich
UART Initialisierung	AH != 0 1 2 3		Die eigentliche Initialisierung erfolgt beim resident machen des TSR-Treibers	zum LSR löschen und Empf-Puffer leeren

Für dieses TSR-Treiber-Programm steht im Rahmen des Praktikums nur eine **minimale 3 Draht Verbindung** ( gekreuzt, aber keine Brücken für Steuersignale im Stecker d.h. kein 3 Draht Null-Modem !! ) zur Verfügung. Daher keine Verwendung der **Modem-Steuersignale** und damit kein **Hardware - Handshake** möglich.

Übertragungs-Parameter: 9600Bd, keine /gerade Parität, 2 Stoppbit, 8 Datenbit

Über 4 Parameter nach dem Programm-Aufruf soll die Schnittstelle COM1 / COM2 eingestellt, der Selbsttest-Modus Loopback ein/aus - der 16Byte FIFO des UART ein/aus – geschaltet und keine oder gerade Parität eingestellt werden.

Aufruf: TSRname 2LFP => COM2 / Loopback ein / FIFO ein / gerade Parität  
alle anderen ( Default ) COM1 / Loopback aus / FIFO aus/ keine Parität

Die Einstellung der Parität per Programmaufruf ist in der Lösungsvorgabe hpra5\_11 noch nicht enthalten !!!!

Zumindest eine der beiden folgenden Varianten ist zu realisieren :

a) Variante mit **Empfangs-Polling** ( serpoll.asm )

b) Variante mit **Empfangs-Interrupt** und Empfangs-Ringpuffer ( Lösungsvorgabe hpra5\_11.asm )

Ein empfangenes Zeichen im UART löst den Empfangs-Interrupt aus und in der zugehörigen Interrupt-Service-Routine ISR sollte das empfangene Zeichen ( in AL ) und der zugehörige Übertragungs-Status aus dem LSR-Registers ( in AH ) in einen Word-Ringpuffer gespeichert werden.

Eine Abfrage ob “Zeichen empfangen“ liest dann nur noch aus dem Ringpuffer aus und kann anhand der Statusinformation die Fehlerfreiheit beim Empfang des Zeichens untersuchen.

Beindet sich kein Zeichen im seriellen Empfangs-Ringpuffer ( Interrupt -> hpra5\_11.asm ) oder im UART-Empfangs-Puffer RBR ( Polling -> serpoll.asm ), so soll vom Software-Interrupt 60h in AX= FF00h zurückgeliefert werden.

**Optional** kann auch entsprechend INT 14h AH = 04 eine **Initialisierung** mit beliebigen Parametern ausprogrammiert werden ( noch nicht in Lösungsvorgabe vorhanden )

## 2. Aufgabe: Terminalemulation mit TSR-Treiber INT60h Serielle Schnittstelle

Zum Testen des TSR-INT60h-Treibers sollen in einem transienten **Terminalemulations-Programm** (z.B. aus Vorlesung TERMEMU3.asm oder TERMCOMx.asm ) die BIOS- INT 14h-Funktionen durch die entsprechenden INT 60h-Funktionen ersetzt werden- Beendigung mit Strg-C.  
Zuvor muss aber natürlich der TSR-INT60h-Treiber resident gemacht worden sein.

### Hinweis:

Um möglichst keine Konflikt-Fehlerquellen zu erhalten, sollte aus dem zu verändernden Terminalemulations - programm der Teil mit direkter Programmierung des UART und die Initialisierung entfernt werden. Denn die Schnittstelleninitialisierung mit der Berücksichtigung der Varianten COMx/ Loopback/ FIFO soll beim Installieren des TSR-INT60h-Treibers erfolgen, wobei die Varianten durch Parameter hinter dem Programmaufruf festgelegt werden. (eine nachträgliche Initialisierung bzw Umschaltung könnte z.B. mit Hilfe des DOS-Multiplexers INT 2Fh realisiert werden )

### Testmöglichkeiten:

- die Terminalemulation kann durch die UART-Option Loopback ohne Kommunikations-Partner mit "sich selber" erfolgen

- mit einer Verbindung zwischen den beiden seriellen Schnittstellen COM1 und COM2 und je einer DOS-Box

- Aufbau einer seriellen Kommunikation mit dem Betriebssystem Monitor 51 des MVUS 80535 ( keine Parität, Monitor 51 arbeitet aber mit Software-Handshake )

### Lösungsvorgabe hpra5\_2l.asm

## 3. Aufgabe: Datei-Übertragung mit TSR-Treiber INT60h Serielle Schnittstelle

Es ist ein transientes Programm für den **Dateitransfer** über die serielle Schnittstelle mit den Funktionen des TSR-Treibers INT60h zu entwerfen.

Der Zugriff auf die zu übertragende **ASCII-Datei** kann mit den DOS-Handle-Funktionen 3Ch – 43h INT 21h erfolgen ( *Skript Systemunterlagen Seite 35* ).

Die **Übertragungssicherheit** soll durch einen **Software-Handshake** zwischen den beiden Rechnern sichergestellt werden.

Das Programm arbeitet **entweder als Datei-Sender oder Datei-Empfänger** ( und damit ist LOOPBACK hier vom Prinzip her nicht mehr funktionsfähig !!! und scheidet als Testmöglichkeit aus.)

Der **Software-Handshake** wird durch den Austausch von in ihrer Bedeutung festgelegten **ASCII-Steuer - signalen** zwischen den beiden Partnern realisiert. ( = nicht transparente Datenübertragung )

### Übliche Steuerzeichen für Software-Handshake

Name	ASCII-Code	Bedeutung
ACK	06h	Acknowledge positive Rueckmeldung
NAK	15h	neg. Ackn. negative Rueckmeldung
STX	02h	Start of Text
ETX	03h	End of Text
XON	11h	Device Control 1 EIN
XOFF	13h	Device Control 3 AUS

### Übertragungs-Steuerung ( Handshake-Ablauf )

Die Empfangsbereitschaft testet der Sender zu Beginn mit **STX**, worauf der Empfänger mit **XON** seine Bereitschaft signalisiert

Der Empfänger sendet für jedes korrekt empfangene Byte ( Parität + Rahmen wird vom UART automatisch geprüft ) das ASCII-Steuerzeichen **ACK** zurück. Bei fehlerhaftem Empfang wird mit einem **NAK** der Sender zum nochmaligen Senden des verfälschten Zeichens aufgefordert.

Alle fehlerfrei **empfangenen** / **zu sendenden** Zeichen werden von ihren **Empfangs-** / **Sende** - Programmen **vor/nach** dem **Schreiben/Lesen in/aus** der Datei auf dem **Bildschirm** ausgegeben.

Das Ende der Datei und damit der Übertragung signalisiert der Sender mit **ETX** und danach werden beide Programme beendet.

Nach vollständiger Übertragung der Datei werden Send- und Empfangs-Programm immer beendet.

Ein Übertragungsfehler in den Steuerzeichen führt nach einer Meldung zum sofortigen Abbruch der Übertragung und Beendigung des Programms, welches das falsche Steuerzeichen empfangen/erkannt hat. Eine weitere Möglichkeit zur Beendigung des Programms bzw. dem Abbruch der Übertragung ist ein Strg C von der Tastatur.

#### **Wartezeit**

Erfolgt auf ein ASCII-Steuerzeichen oder ein übertragenes Zeichen keine entsprechende Reaktion des Übertragungs-Partners, so soll nach einer Wartezeit von 2 Minuten das wartende Programm mit Fehlermeldung beendet werden.

Für den Programmstart bedeutet dies zum Beispiel nach 2 Minuten den Abbruch, wenn in dieser Zeit Empfänger / Sender nicht die Steuerzeichen STX / XON empfangen haben.

*(eine Möglichkeit für Wartezeit siehe INT 15h, Fktn 83h Skript Systemunterlagen Seite 24)*

#### **ASCII- und Binär- Dateien**

Aufgrund der Verwendung von **ASCII-Steuerzeichen** für den Software-Handshake sind nur reine ASCII-Dateien (bei denen die verwendeten Steuerzeichen nicht auftreten können) übertragbar.

Die Bytes aus Binär-Dateien müssen daher für die Übertragung mittels Software-Handshake erst in einen speziellen **Hex-Code** umgewandelt (beispielsweise INTEL-Hex-Code) und vom Empfänger natürlich entsprechend decodiert werden.

Jeweils ein Halbbyte wird dazu in das ASCII-Zeichen seines HEX-Wertes umcodiert und somit werden für ein binäres Byte zwei ASCII-Zeichen übertragen und es können somit die Bitmuster der Steuerzeichen nicht mehr auftreten.

(Was bedeutet das für die Fehlerminderung bzw. Codesicherung – Hamming-Distanz ?????. Wird dadurch die minimale Hamming-Distanz erhöht? – Wird die Übertragungssicherheit erhöht? – Rechnergrundlagen !!)

**Hinweis:** Beim Dateitransfer kann das zu testende Programm nicht mehr im Loopback-Modus mit sich selber getestet werden. Befinden sich aber auf dem Rechner zwei serielle Schnittstellen COM1 und COM2 so kann mit einer gekreuzten 3-Draht-Verbindung zwischen den beiden Steckern und je einer DOS-Box für Send- und Empfangs-Programm die Übertragung auf einem einzigen Rechner durchgeführt werden.

Aber !! nicht jeder Programmabsturz muß dabei vom auszutestenden Programm herrühren !!!

Zum **Testen** der Dateiübertragung am besten die ASCII-Quelldatei **\*\*\*\*.asm** des eigenen Programms benutzen und nach der Übertragung durch Assemblieren auf Fehlerfreiheit testen.

#### **Lösungsvorgabe hpra5\_3l.asm**

#### **Unterschiede der beiden seriellen Schnittstellentreiber:**

<b>BIOS INT 14h</b>	<b>TSR INT 60h</b>
Bedient Verbindung Rechner (DTE) – Modem (DCE) Setzt damit 9-Draht-Verbindung voraus Übertragungssteuerung mittels Modem-Steuersignale und damit Hardware-Handshake Rechner-Rechner-Verbindung setzt Null-Modem voraus, entweder das Universal-Null-Modem mit allen Leitungen gekreuzt oder das 3-Draht-Null-Modem mit Brücken in den Steckern	Dient Verbindung Rechner (DTE) – Rechner (DTE) Setzt nur minimalste gekreuzte 3-Draht-Verbindung voraus Damit Übertragungs-Steuerung nur noch mittels Software-Handshake möglich

**Hinweis:** Bei interruptgetriebenem Empfang kann am augenblicklichen LSR-Status (Funktion 03) nicht erkannt werden, ob ein Zeichen empfangen wurde, da LSR-Status mit Interrupt-Empfang nicht zeitlich konsistent ist.

Deswegen bei Interrupt-Empfang für Empfangs-Prüfung nicht Funktion 03 aufrufen sondern direkt mit Funktion 02 arbeiten.