Praktikum Mikrorechner 9 (serielle Schnittstelle)

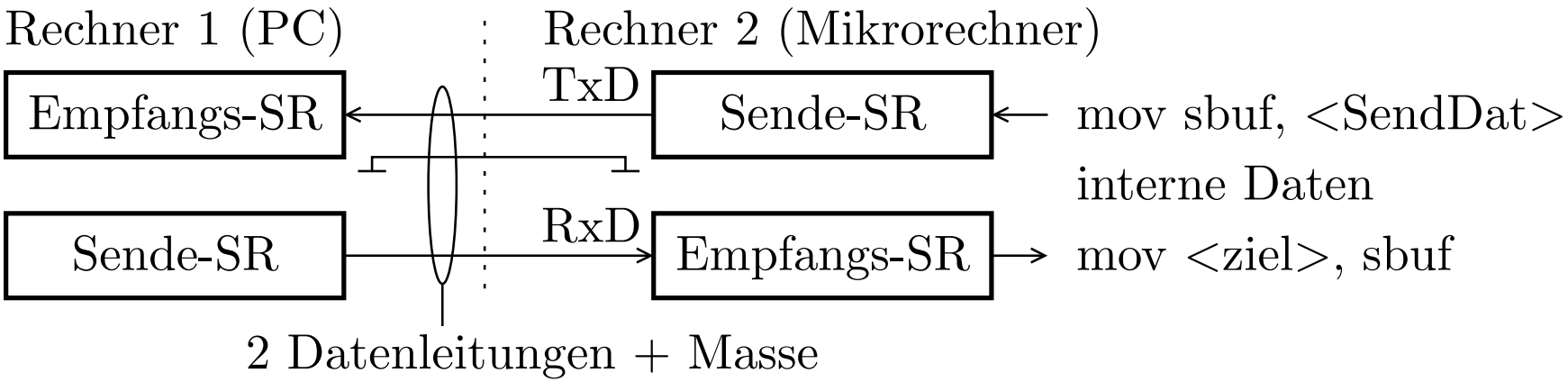
Prof. G. Kemnitz

Institut für Informatik, Technische Universität Clausthal

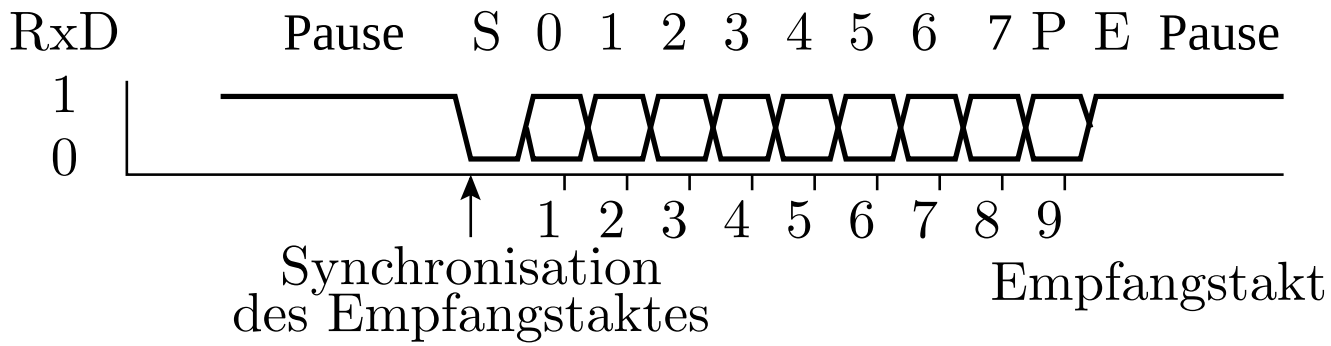
5. November 2014

Serielle Schnittstelle

Asynchrone serielle Schnittstelle (USART)



Protokoll



S: Startbit

0 bis 7: Datenbits

P: Paritätsbit falls aktiviert

E: Stopbit

SFR der asynchronen seriellen Schnittstelle

sbuf (Adresse 9ch)

Schreibzugriff: Beschreiben des Sendepuffers und Übertragungsstart; darf erst nach Abschluss der vorherigen Übertragung erfolgen

Lesezugriff: Lesen des Empfangsregisters; liefert nur nach Empfangsabschluss sinnvolle Werte

SCON (Adresse 98h)

sm0	sm1	sm2	ren	tb8	rb8	ti	ri

sm0 bis sm2 Betriebsart (Einstellung beibehalten)

ren receive enable (bleibt 1)

tb8, rb8 in speziellen Betriebsmodi 9. Datenbit

ti Sendeflag: vor Versenden eines Bytes vom Programm löschen; wird von der Hardware nach Empfang eines Bytes gesetzt

ri Empfangsflag: wird nach Empfangsabschluss von der Hardware gesetzt und muss nach Auslesen des Bytes vom Programm gelöscht werden

Baudrate mit Timer 1 auf 9,6 kbaud eingestellt

Benutzung der seriellen Schnittstelle

- vor der seriellen Ein- und Ausgabe Interrupt abgeschaltet! Dann funktioniert allerdings der Monitor, d.h. der Schrittbetrieb, Unterbrechung mit ctr-C etc. nicht mehr.

```
clr es   ; seriellen Interrupt ausschalten
         ; Komunikation mit dem Monitor aus
...      ; serielle Ein- und Ausgabe
setb es  ; seriellen Interrupt einschalten
         ; Kommunikation mit dem Monitor ein
```

putchar und getchar

- Zeichen Senden
  - Sendeflag löschen
  - Zeichen in den Sendepuffer schreiben
  - warte, bis Sendeflag gesetzt ist (dauert ca. 1 ms)

```
clr ti
mov sbuf, #char ; oder dadr|@ri|...
jnb ti, $       ; $ - Adresse des Befehls
```

- Zeichen Empfangen
  - Warte bis Empfangsflag gesetzt ist
  - Zeichen aus dem Sendepuffer in eine Variable kopieren
  - Empfangsflag löschen ;

```
jnb ri, $       ; kann längere Zeit dauern
mov dadr, sbuf  ; Empfangswert speichern
clr ri
```

```
EndeZeichen equ 1bh
Monitor     equ 0e300h
org 100h
 clr es       ; seriellen Interrupt aus
loop:
 ; ---------------------------------------
 jnb ri, $    ; auf ein Zeichen warten
 mov P1, sbuf; Zeichen auf Port 1 ausgeben
 clr ri       ; Empfangsflag löschen
 ; ---------------------------------------
 clr ti       ; Sendeflag löschen
 mov sbuf, P1; Zeichen zurücksenden
 jnb ti, $    ; warte, bis versendet
 ; ---------------------------------------
 ; wiederhole, bis Endezeichen
 mov a, P1
 cjne a, #EndeZeichen, loop;
 setb es      ; seriellen Interrupt ein
 ljmp Monitor; Rücksprung zum Monitor
```

Zeichentabelle

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Aufgaben

Aufgabe 9.1: Sende Zeichenkettenkonstante

Entwickeln Sie ein Programm, das die im Befehlsspeicher einprogrammierte Zeichenkettenkonstante:

```
org 200h
String:
 db "Das ist ein Text", 0ah, 0dh, 0
```

über die serielle Schittstelle ausgibt:

- initialisiere einen Zeiger mit der Adresse String
- Wiederhole, bis der Zeiger auf die abschließende Null zeigt
  - Lese das Zeichen, auf das der Zeiger zeigt
  - versende es über die serielle Schnittstelle
  - erhöhe den Zeiger um Eins

Aufgabe 9.2: Umwandlung eines Zeichens in eine Hexadezimalzahl

Entwickeln Sie ein Programm mit folgendem Algorithmus:

- Wiederhole, bis ESC (1bh) empfangen wird
  - warte auf ein Zeichen
  - zurücksenden einer Zeichenkette aus den zwei Hexadezimalziffern des Zeichens, gefolgt von einem Zeilenumbruch "\n" (0ah, 0dh).

Beispiel:

Empfang	Senden
'A' (41h)	"41\n" (34h, 31h, 0ah, 0dh)
'1' (31h)	"31\n" (33h, 31h, 0ah, 0dh)