Introduktion til seriel kommunikation.
Senest redigeret d. 1/11- 2011. / Valle

Se genial om 8051 seriel kommunikation: http://www.edsim51.com/8051Notes/8051/serial.html

I AT89C4051 er der indbygget en Seriel Port, også kaldet en **UART**. Det står for Universal Asyncron Receive Transmit.

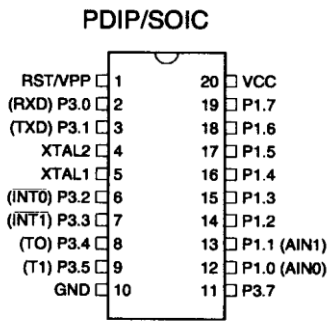
Porten er rimelig let at bruge, når man har forstået princippet. Efter opsætning eller konfigurering er det blot at få programmet til at skrive en byte til et bestemt register, kaldet SBUF, (for Seriel Buffer) for at sende den. Hvis en byte er modtaget, læses det også fra registeret SBUF, men der er dog tale om to forskellige registre, med samme navn.

Eks på kode:

```
Mov SBUF, A        ; Indholdet af reg A kopieres til SBUF reg. og sendes

Mov A, SBUF        ; Læs modtaget data til reg A.
```

Men først skal programmet lave en opsætning, så den serielle del af Controlleren bliver aktiv, - og der skal bestemmes, hvilken med hvilken hastighed bittene skal sendes. Såkaldt Baud-rate. Og til det bruges timer 1.

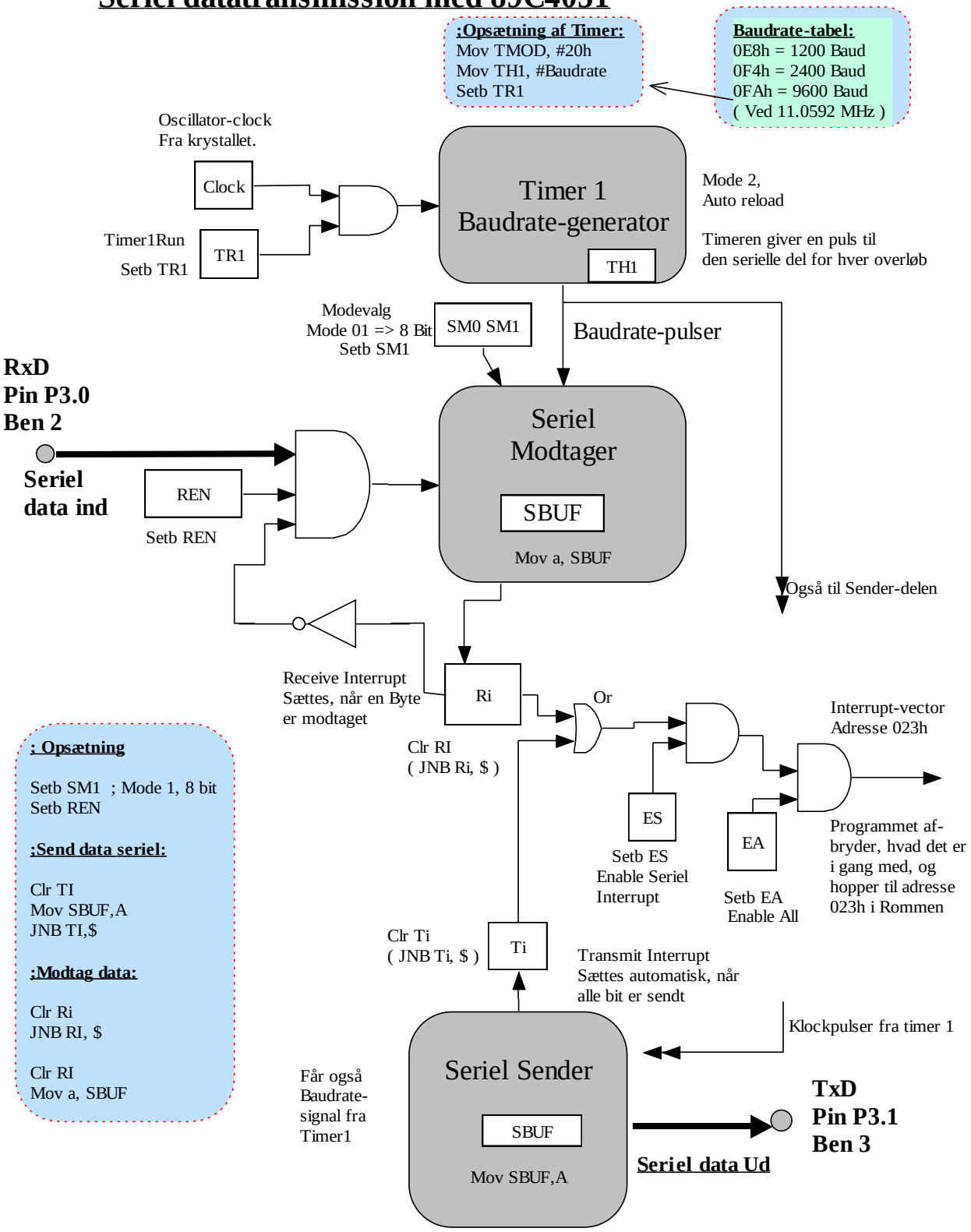
Skrives herefter til SBUF, sendes data umiddelbart efter automatisk ud på udgangsbenet, P3.1. Når UARTèn er færdig med at sende en byte, sættes et flag, eller et bit, som programmet kan tjekke og dermed vide, når den serielle del er færdig med at sende en byte.



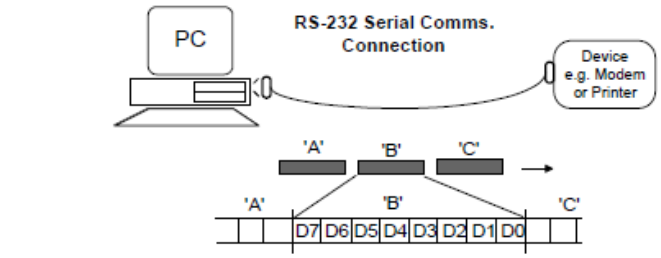
Der bruges to ben af port 3 til kommunikationen. På P3.0, pin 2 på IC'en, findes RxD, Receive data, og på P3.1, pin 3, TxD, Transmit Data.

Følgende ses et "billede" af opsætningen af den serielle del:

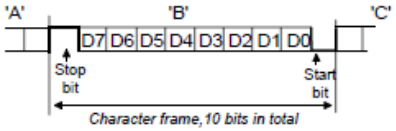
Seriel datatransmission med 89C4051



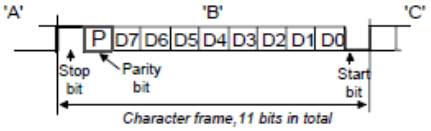
Valle
30/9-03
17/3-04
27/10-04



a) Sequence without framing

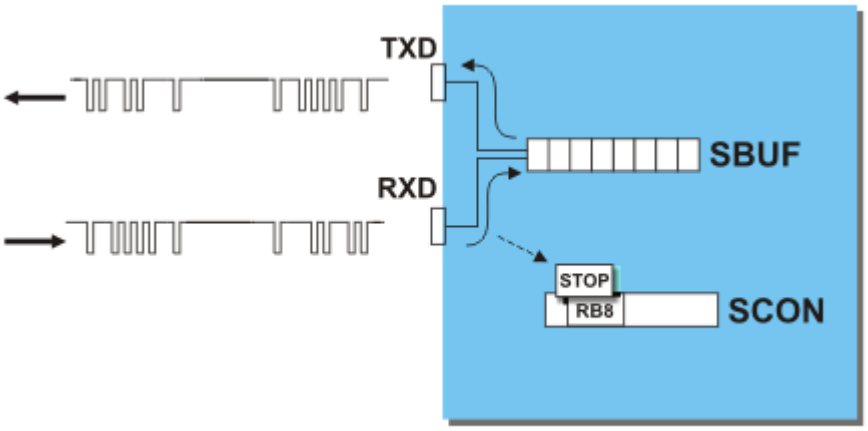


b) Framed data

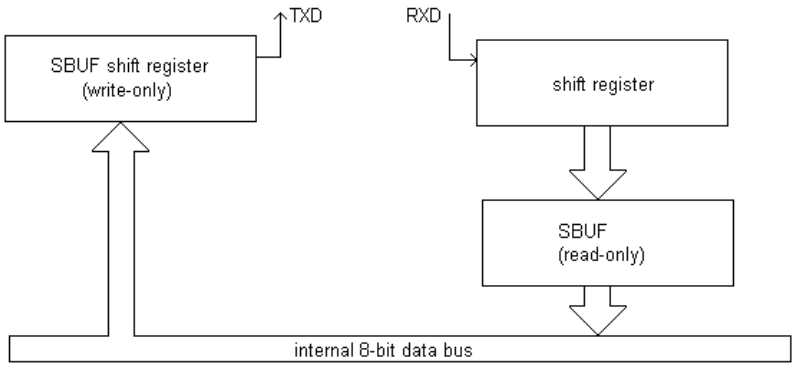


c) Framed data including a parity bit

Kilde: http://www-2.dc.uba.ar/materias/oc1verano/2006/download/8051.pdf

I controlleren ledes et modtaget signal ind til et register, kaldet SBUF. Hvis man vil sende et serielt signal, sendes det ud af et register, der også kaldes SBUF. I realiteten er der 2 fysiske registre der hedder SBUF	

Her et skema over de to registre:



Kilde: http://www.edsim51.com/8051Notes/8051/serial.html

Seriel kommunikation udføres med et antal bit pr sekund. Der er fra gammel tid (Teletext og teletype) valgt fx 150 bit pr sek., også kaldet Baud (udtales ” Boo ”) eller Baudrate.
Senere, efterhånden som elektronikken blev hurtigere, blev Baudraten fordoblet, - indtil flere gange.

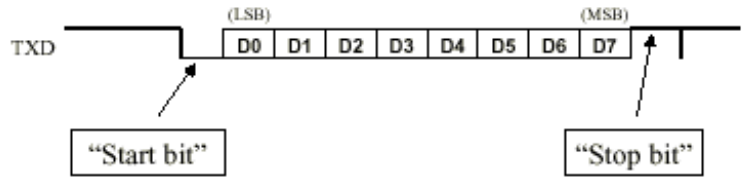
Her ses et skema med nogle gængse baudrates.

Jeg plejer at bruge 1200 Baud.

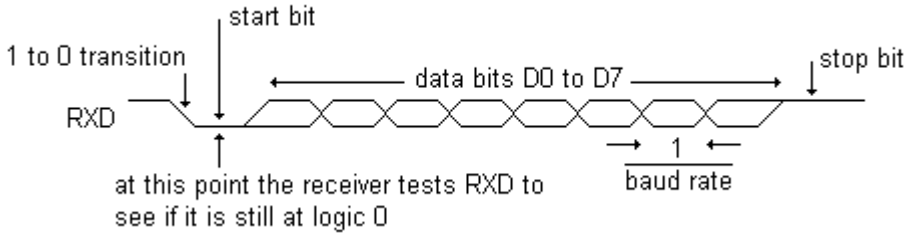
Baud	Bit Width	Frame Width (1 start bit, 8 data bits)	Real Data Rate * (1 start bit, 8 data bits, no parity, 1 stop bit)
4800	208.33us	1875us	480 Bytes/s 0.47 KBytes/s
9600	104.17us	937.5us	960 Bytes/s 0.94 KBytes/s
19200	52.08us	468.75us	1920 Bytes/s 1.88 KBytes/s
38400	26.04us	234.38us	3840 Bytes/s 3.75 KBytes/s
57600	17.36us	156.25us	5760 Bytes/s 5.63 KBytes/s
115200	8.68us	78.13us	11520 Bytes/s 11.25 KBytes/s

Dataledningen mellem to processorer, dvs. fra en TxD til en anden processors RxD, er NH, dvs. normalt høj. Når der sendes et signal, startes med at ledningen bliver lav. Dette kaldes et Startbit. Herefter sendes de 8 bit, og evt. et paritetsbit. Sluttelig et Stopbit, der er et 1-tal. Herefter er dataledningen høj indtil næste startbit sendes.

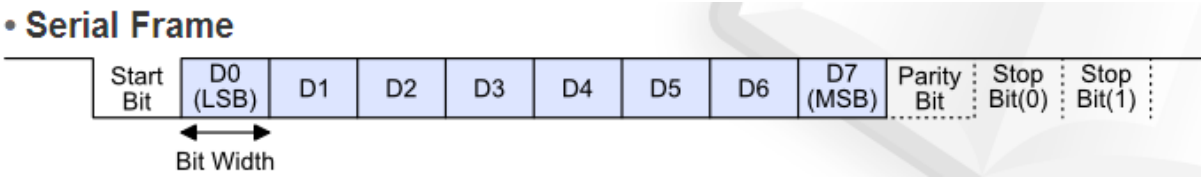
Figuren viser, at i alt 10 bit sendes.
LSB sendes først.



Her er et par flere eksempler på at vise, hvordan et signal sendes:

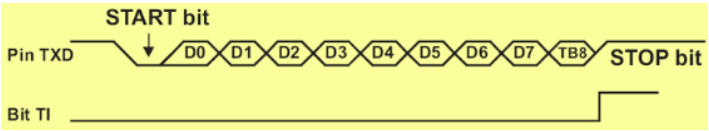


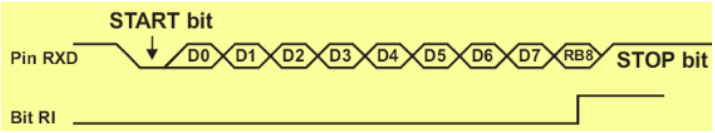
http://www.edsim51.com/8051Notes/8051/serial.html



http://www.labbookpages.co.uk/electronics/serialPort.html

$$Baudrate_{Mode1\&3} = \frac{2^{SMOD}}{32} \cdot \frac{Osc\ frekv.}{12 \cdot (256 - TH1)}$$





http://www.scribd.com/doc/19533035/8051-Tutor

Obs: 1200 Baud = 0E6h @ 1200 MHz, = 230 dec.

UART'en i AT89C4051 kan indstilles til at arbejde på 4 måder, såkaldte Modes.

Mode0: Mode1: 8 Bit. (Den vi bruger). Mode2: 9 Bit Mode3: 9 Bit	8051 Serial Port Mode 1 • 8-bit **UART** mode with *variable baud rate* • Popular interface usually associated with RS-232 • **10-bits** are required to transfer **8-bits** of data – 2 bits are used for synchronization – Start bit and Stop bit • Signal out of the serial port (NOT on the interface) (LSB) (MSB) TXD D0 D1 D2 D3 D4 D5 D6 D7 "Start bit" "Stop bit"

Opsætningen foregår i registeret SCON. Det står for Seriel Control-register. Her vælges den mode, man ønsker, UART'en skal arbejde i:

Mode 1, 8-bit.

I mode 1 sendes / modtages asynkront på 1 ledning, dvs. uden at man overfører klock-frekvensen.

Dette kræver dog, at både sender og modtager arbejder med samme format (Startbit, databit & stopbit) og med samme Baudrate.

Synkroniseringen mellem sender og modtager sker vha. startbittet. Normalt er der "Højt" på TxD, som er forbundet til en anden processors RxD. Når der i modtageren registreres et 0, dvs. en faldende flanke på modtageren, registreres dette som starten på en datapakke. I midten af det

modtagne startbit tjekkes igen om der stadig er ”nul” på RxD, eller det blot var støj. Er der stadig lav, opfattes det af modtageren som start på data.

Herefter ventes til midten på første databit, bit 0, LSB, hvor værdien på RxD, 0 eller 1, clockes ind i input-skifteregisteret. Dette sker igen for følgende bit osv.

Idet der aftastes i midten af et bit-varighed opnås en pæn støjundertrykkelse. Det er absolut nødvendig, at sender og modtager bruger samme Baudrate.

Først gennemgås de forskellige bit i SCON registeret.

SCON Registeret: Navne og betydning:

Bit 7	Bit 6	5	4	3	2	1	Bit 0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI
= 0	= 1		Enabler mod-tagning			Transmit over.	Byte modtaget.

Nemmeste opsætning i et program sker med koderne:

```
Setb SM1
Setb REN
```

Eller:

```
Mov SCON, #??h
```

Transmit bit

Når senderen er færdig med at sende en byte, sætter den en ”Transmit over”- flag, kaldet TI , eller Transmit Interrupt bit.

Recieve bit

I modtageren sætter UART’en et bit, eller et flag, kaldet RI, Receive Interrupt flaget.
RI-flaget skal cleares, før der kan modtages en ny byte.

Baudrate

Baudraten styres i både sender og modtager af timer 1-overløb. Den indbyggede Timer / Tæller tæller skal bringes til at tælle op fra en indstillelig værdi, og ved overløb (FFFFh) sendes en puls til den serielle port. Timer 1 skal indstilles til mode 2, Auto Reload, som er beregnet til at generere baudrate til den serielle del. Auto Reload betyder, at den automatisk igen ved overløb starter fra den indstillede værdi.

Timeren indstilles med følgende kode:

```
Mov TMOD, #20h
Mov Th1, #0E8h          ( for 1200 Baud )
Setb Tr1
```

Dette skema viser en oversigt over gængse Baudrates og indstillingerne: (Bit Smod er default = 0)

Baudrate	Timer Reload værdi	Krystalfrekvens MHz	SMOD bit
110	072h	6,0	0
150	040h	11,0592	0
300	0A0h	11,0592	0
600	0D0h	11,0592	0
1200	0E8h	11,0592	0
2400	0F4h	11,0592	0
4800	0FAh	11,0592	0
9600	0FAh	11,0592	1
9600	0FDh	11,0592	0
19,2K	0FDh	11,0592	1
57,6K	0FFh	11,0592	1

Der kan tillades en frekvensafvigelse på +- 3 %.

```
;@ 11.059MHz           smod    TH1
;Baud rate:  1200      0       0e8h
;Baud rate:  2400      0       0f4h
;Baud rate:  9600      0       0fdh
;Baud rate: 19200      1       0fdh

;@ 12MHz               smod    TH1
;Baud rate:  1200      0       0e6h
;Baud rate:  2400      0       0f3h
;Baud rate:  9600      1       0f9h
```

Eksempel på Sendeprogram

```
Mov SCON, #40h      ; Mode = 1, 8 bit data, Recieve disabled.
Mov TMOD, #20h      ; Timer 0 disabled
                    ; Timer 1 mode 2 Auto Reload
                    ; C/T = 0
                    ; Gate = 0
Mov PCON, #80h      ; Sæt bit SMOD i reg PCON, = Ikke dele med 2, kun 16
Mov TH1, #100h-6    ; Genloadværdi = 6,
```

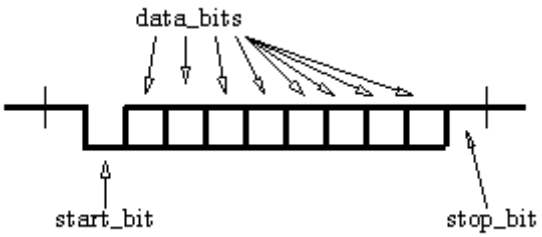
```
                        ; 11,0592 MHz / 12 / 16 / 9600 Baud = 6 !
                        ; Tælleren loades med 100h – 6 !

Setb TR1                ; Start timer 1, Timer1 Run ( I TCON Registeret )
Clr TI                  ; Clr Transmit Interrupt bit.
Mov SBUF, A             ; A-værdi til SBUF, og start automatisk sendefunktion

JNB TI, $               ; Vent til sendebuffer er tom, TI bliver sat,
                        ;  $ = Samme linie
```

Eksempel på modtageprogram

```
Mov SCON, #50h          ; Mode 1, 8 databit, Receive enable
Mov TMOD, #20h          ; Timer 0 disabled
                        ; Timer 1 = mode 2
                        ; C/T = 0
                        ; Gate = 0
Mov PCON, #00h          ; SMOD = 0 ( Tæller / 2 )
Mov TH1, #100h-3        ; Genloadværdi = 3
                        ; ( 11,0592 MHz / 12 / 16 / 2 / 9600 Baud )
Setb TR1                ; Start timer 1
JNB RI, $               ; Vent på RI flag
Clr RI                  ; Slet Receive flag
Mov A, SBUF             ; læs modtaget byte til reg A
```

Hvis der er længere mellem to uC, der skal kommunikere, kan man bruge standarden RS232, eller RS485.

RS232:

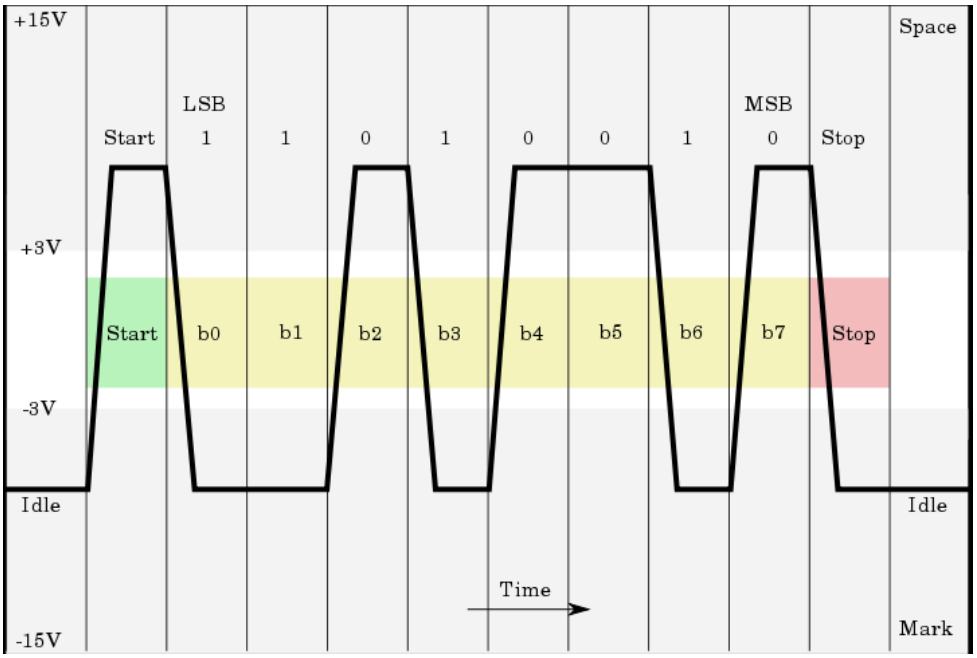
Signaler kan sendes direkte fra uC til uC, dvs. med signalniveauer på 0 og 5 Volt. Dette medfører dog begrænset rækkevidde. Men det kan sagtens bruges til at koble to uC sammen på samme print. Fx kan 1 uC kontrollere et tastatur, og sende de indtastede data til en anden uC.	

Ønskes større rækkevidde, kan bruges lidt større spændinger i signalet. Fx kan standarden, eller "protokollen" der kaldes RS232 anvendes. Her er signalernes spænding noget anderledes. Fx +/- 12 Volt. Et 1, sendes som minus 12 volt, et nul som et + 12 volt.

Her ses et billede af signalerne i en transmission af en pakke på 8 databit efter RS232.

Evt. kan sendes en 9. bit, en Paritetsbit.

Transmissionsafstanden er dog ikke over ca. 10 meter !

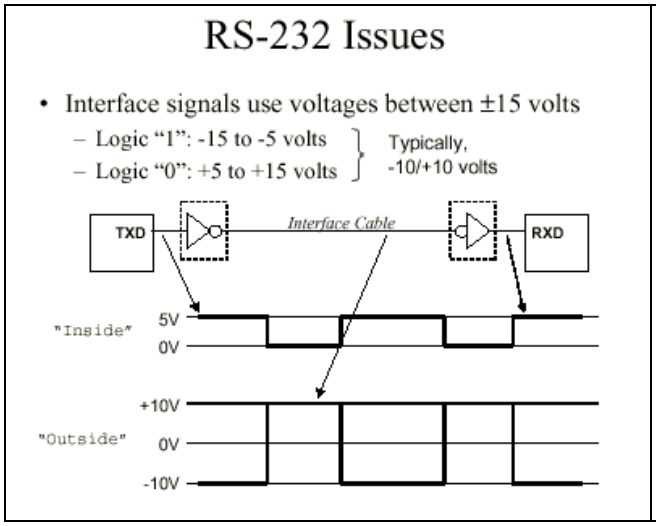


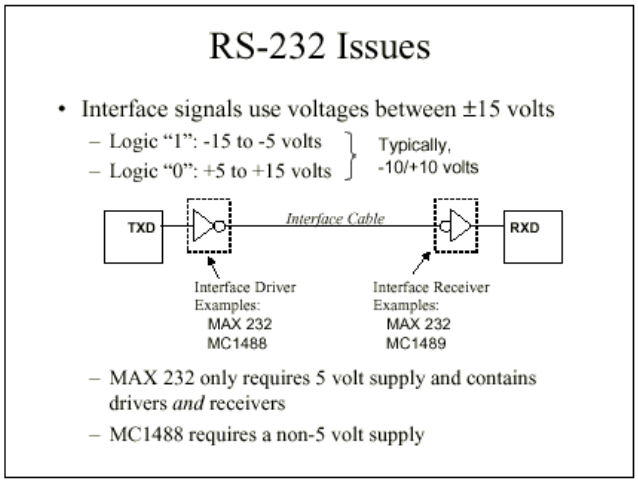
Grafen er fra : http://en.wikipedia.org/wiki/RS-232

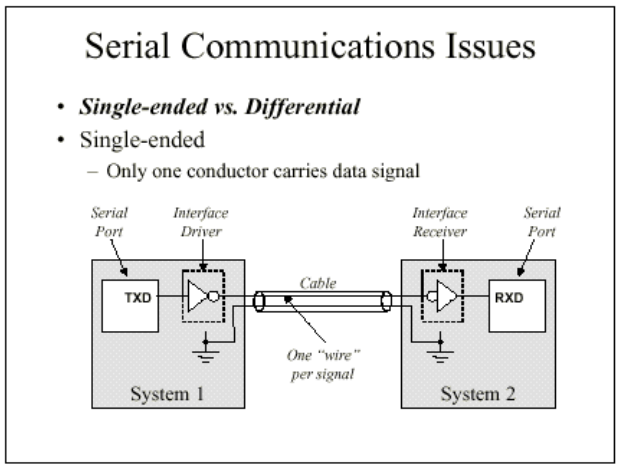
De data, dvs. de 8 bit-pakker, der blev brugt til at sende data til printere i "gamle dage" var ordnet efter ASCII-tabellen.

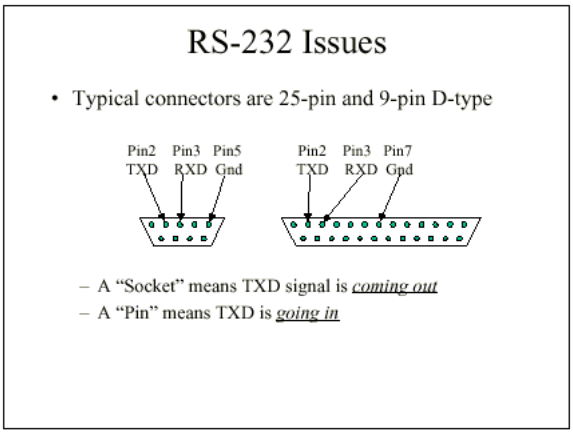
Tjek kredsen MAX232, der kan bruges til at omforme fra 5 Volt signaler til +/- 12 Volt.

I RS232 protokollen sendes signalet på 1 ledning, men en nul-ledning skal også forbindes. Tillige blev brugt et antal handshake-signaler.







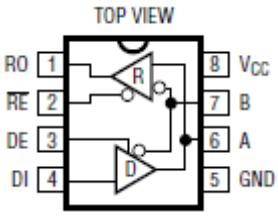


RS485:

RS485-protokollen.

Ønskes en meget stabil og langtrækkende signaloverførsel, kan med fordel vælges at bruge RS485.

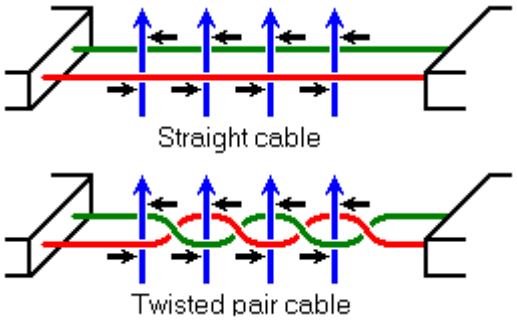
Standarden RS485 benytter et balanceret signal, på kun 2-ledere. Rækkevidden er ca. op til 1 km. på blot 5 Volt. Det er dog en betingelse, at lederne er snoede.



Til at kode og til at afkode signalet bruges en lille IC. Signalet overføres fra kredsens A-signal og B-signal.

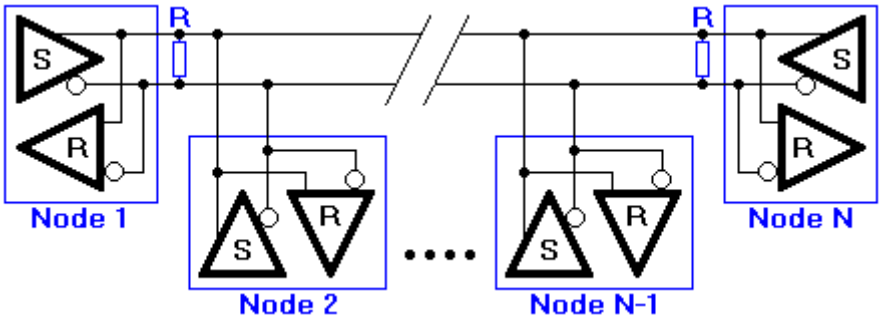
Kredsen kan kobles som enten sender eller modtager.

De blå pile repræsenterer magnetfelt, de sorte induceret støj.



Billede fra : http://www.lammertbies.nl/comm/info/RS-485.html

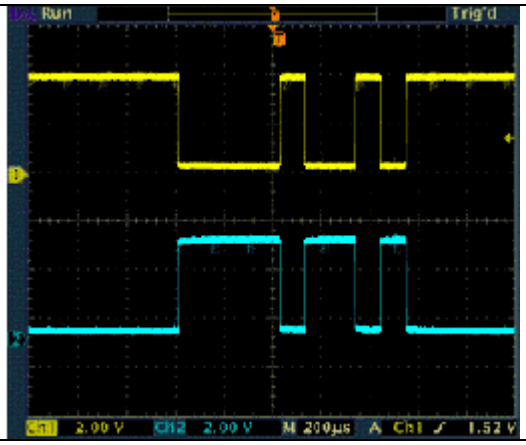
Her ses et eksempel på hvordan flere sendere / modtagere kan kobles sammen på et netværk.



Normally, an RS-485 receiver output is "1" if A > B by +200mV or more, and "0" if B > A by 200mV or more.

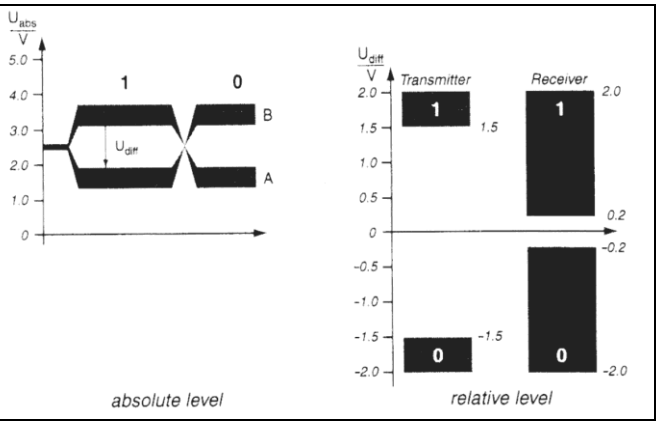
RS485-bussen sender data differential over et twisted pair kabel. Dvs. at når den ene ledning bliver positiv, bliver den anden nul, og modsat.

Kablet termineres med 120 Ohm i hver ende



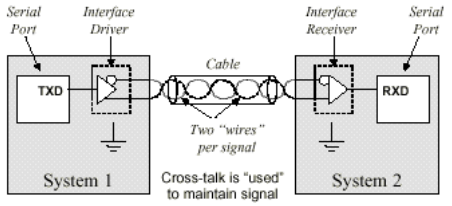
Kilde: http://www.maxim-ic.com/appnotes.cfm/an_pk/723/

Her ses en graf over signalerne:



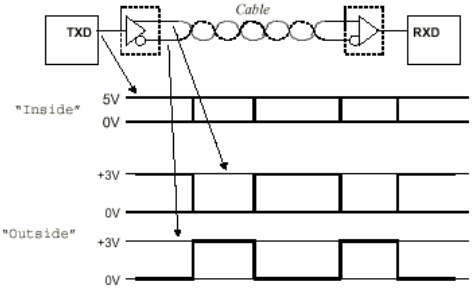
Serial Communications Issues

- ***Single ended vs. Differential***
- Differential
  - A conductor pair (twisted-pair) used



RS-422/485 Issues

- Signals use voltage differences between ±2 and ±6 volts



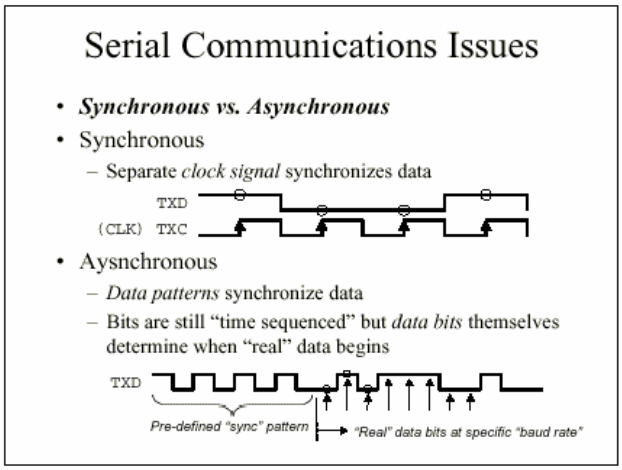
Bilag: Yderligere materiale:

Synkron / Asynkron transmission.

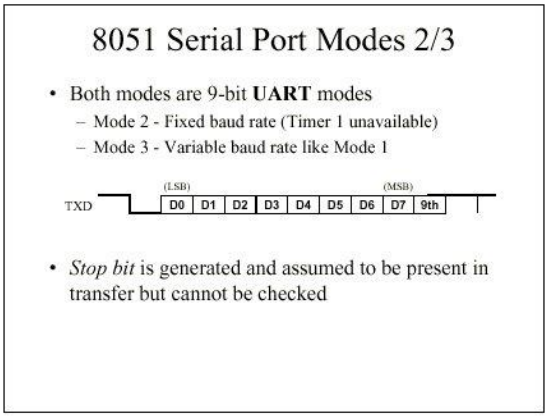
Vha af kontrolbit i et register, vælges hvilken mode, den serielle UART skal konfigureres til.

Ved synkron sending, sendes tillige en klocksignal. Her vil vi gennemgå asyncron.

Mode 0.



Mode 2 / 3:



Modes & Kontrolbit

Den Serielle port kan sættes op i 4 modes, mode 0 til 3. Modes og andre opsætninger foretages ved at sætte nogle bit i registeret kaldet SCON, Seriel port Control-registeret.

Vha. Mode-valg indstilles den serielle ports funktion.

I **Mode 0** kan der vha. et skifteregister, SIPO- eller PISO - register laves udvidelser i antallet af I/O-linier.

I **Mode 1** sendes 8 bit, ingen paritetsbit !!

I **Mode 2 & 3** bruges udover de 8 databit en 9. databit som kan bruges til paritetsbit.

Her følger en nærmere forklaring af bittenes betydning:

Bit	Forklaring
SM0	Mode-valg-bit
SM1	Mode-valg-bit
SM2	Multiprocessorbit SM2 bit styrer "Multiprocessor-data-udveksling i mode 2 & 3. Er SM2 sat, bliver modtage interrupt flaget RI, Receive Interrupt ikke sat, når det 9. modtagne bit, RB8 er nul. Bruges i systemer med flere processorer, hvor en processor sender data ud til en af max 255 tilsluttede under-processorer. Er SM2 bit lav, bliver flaget RI sat, uafhængig af det 9. databit i hvert modtaget ord. Hvis sat op, bliver der når RI bliver sat, udført et interrupt. (Se i interrupt-afsnittet.) Ønskes næsten altid = 0 !!!! I hvertfald her !!
REN	Receive Enable Bit. Sættes denne, frigives / tillades modtagelse af data på RxD. = 0 disabler modtagelse.
TB8	Bruges i mode 2 & 3. I mode 2 og 3 sendes altid 9 databit, hvoraf sidste tages fra bit TB8.
RB8	Bruges i mode 2&3. I mode 2 & 3 gemmes det 9. bit = paritetsbit i bit RB8.
TI	Transmit Interrupt Flag. Transmit færdig !. Flaget sættes automatisk når et ord er sendt. Dette kan fx udløse et interrupt i styreprogrammet, eller programmet kan blot vente på at dette bit sættes. I Mode 0 sættes TI ved slutningen af det 8. Bit. I de øvrige modes sættes TI ved begyndelsen af det 9. Bit (Stopbit / paritetsbit ???) Softwaren skal resette TI igen !
RI	Receive Interrupt flag. Modtage færdig-flag.

Nærmere beskrivelse af MODE-BIT
SM0 og SM1 er Mode – valg bit.

Mode	Styrebit	SM2	Forklaring af portens funktion i mode:
0	SM0 = 0 SM1 = 0	Skal være 0	8 bit skifteregister, Bruges fx til udvidelse af antal I/O-linier Sende/Modtagefrekvens : Krystallet / 12 RB8 bruges ikke. TI, Transmit Interrupt sættes ved slutningen af det 8. Bit. RI-bittet sættes ved slutningen af det 8. bit
1	SM0 = 0 SM1 = 1	Sat	8 bit UART Sendefrekvens / Modtagefrekvens indstilles med timer 1 og bit SMOD.. Hvis SM2 er sat, udføres der kun et interrupt (RI flaget sættes) hvis der modtages gyldig stopbit efter et modtaget ord. Det modtagne stopbit lagres i RB8. TI bittet sættes ved begyndelsen af det 9. Bit ved sending. Receivebit RI sættes i midten af stopbittet.
2	SM0 = 1 SM1 = 0		9 bit UART. Sende/modtage frekvens Krystal / (32 eller 64) 9. bit er paritetsbit, Skal lagres i RB8 i reg. SCON. TI, Transmit Interrupt bit sættes ved begyndelsen af det 9. Bit. Skal resettes af programmet. RI, Receive Interrupt bit sættes i midten af stopbittet.
3	SM0 = 1 SM1 = 1		9 bit UART. Sende/modtage frekvens indstilles med timer 1 og bit SMOD. 9. bit er paritetsbit, TI, Transmit Interrupt bit sættes ved begyndelsen af det 9. Bit. Skal resettes af programmet. RI, Receive Interrupt bit sættes i midten af stopbittet.

Baudrate

Baudraten styres for både sender og modtager af timer 1-overløb. Den indbyggede Timer / Tæller tæller op fra en indstillelig værdi, og ved overløb (FFFFh) sendes en puls til den serielle port. Timer 1 skal indstilles til mode 2, Auto Reload, som er beregnet til at generere baudrate til den serielle del.

I den serielle del deles timer-overløb-pulserne igen med 16, og evt. afhængig af bit SMOD yderligere med 2.

Er SMOD sat til 0, deles altså med 32, er SMOD = 1 deles med 16.

SMOD findes som bit 7 i PCON registeret, Power Control Register.

```
        Mov PCON, #00h          ; SMOD = bit 7 i PCON = 0, altså deles med (16 * 2)
```

Baudraten bestemmes altså som krystallets frekvens / 12 og så i den serielle del igen med 16 eller 32. Anvendes en 11.0592 MHz krystal, passer det med de gængse baudrates.

Baudraten kan beregnes vha flg. formel:

$$Baudrate = \frac{2^{SMOD} \cdot Tælleroverløb}{32}$$

Interrupt:

Interrupt vektor for seriel transmission er 23h !!

Assemblerprogram eksempel.

```
/* Program Hoved
Programmet er lavet af:   osv.

*/

$NOMOD51                ; Se bort fra 8051 registre
$INCLUDE (AT892051.INC) ; Benyt istedet ATMEL89C2051 definitioner defineret i .INC-filen.


; definitioner

        Baudrate EQU =0E8h      ; 1200 baud ved 11,0592 MHz

;------------------------------

        Org 0h          ; programstart
        Jmp Start       ; hop over Interrupt vektorer


        Org 23h         ; Seriel interrupt vektor
        Call Serint     ; seriel interrupt rutine
        RETI



Start:  Org 30h

        Mov SP, #30h            ; Flyt stackpointeren til 30 h i RAM
```

; Opsæt timer til Baudrate generering

```
        Mov TMOD, #20h      ; vælg Auto reload
        Mov TH1, #Baudrate  ; Load genloadværdi
        Setb TR1            ; Sæt timer 1 igang
```

; Opsæt sender

```
        Setb SM1            ; Mode 1, 8 bit
```

(Her placeres så senderprogrammet)

;Opsæt Modtager

```
        Setb SM1            ; Mode 1, 8 bit
        Setb REN            ; Sæt recieve enable bit
```

Opgave

Arbejd sammen 2 grupper. Opbyg på et fumlebræt 1 uC som sender, og på et andet 1 uC som modtager.

Senderen skal have ledninger fra P1, der kan sættes ned til nul. Det mønster, der så læses med jævne mellemrum på port 1 skal sendes til modtageren.

Modtageren skal læse den sendte byte, og tænde 8 lysdioder i samme kombination, som ”kontakterne” hos senderen.