BUSSER

Dette kompendium er et forsøg på at beskrive nogle af de forskellige busser, der anvendes inden for elektronikken til kommunikation imellem uC'er, ICér og PC-er.

Kompendiet er ikke færdigt, - men bearbejdes løbende: Der bør omskrives og især forkortes.

Links i dokumentet:

Ældre Busser, Centronics, RS232, RS232,
Fra uC til uC, RS485, Parsnoede Wir – Twisted pair,
I2C,
SPI, 3-wire, Microwire og 4-wire-busserne, Microwire, 3Wire,
1-wire,

De busser, der i dette materiale hovedsagelig er tale om er:

RS232, RS485, 3-wire, SPI, MicroWire, I^2C, 1-Wire.

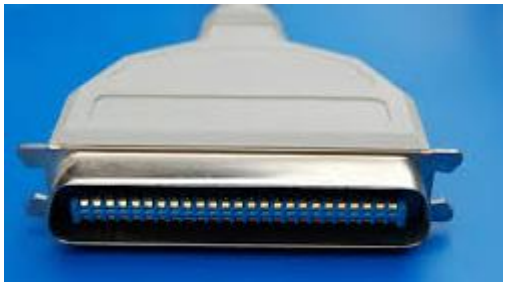
3-wire, SPI og Microwire er i nogen grad overlappende, - og – synes jeg - svære at holde rede på.

Se evt. Youtube om forskellige busser: Youtube om Busser: (6:42)

Ældre Bussystemer: Centronic og RS232

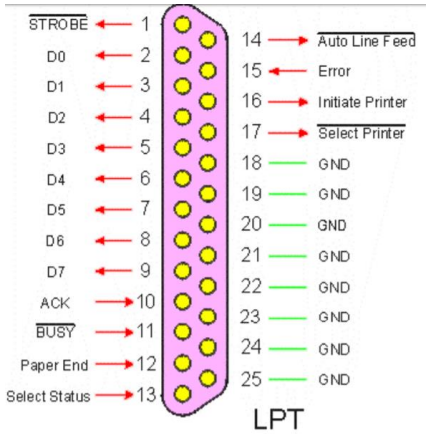
Mellem gamle PC-er og fx en printer foregik kommunikationen enten parallelt eller serielt. Parallel-protokollen hedder Centronic, og er en 8 bit parallel data-overførsel, plus ledninger til forskellige kontrol-signaler, såkaldte handshake-signaler.

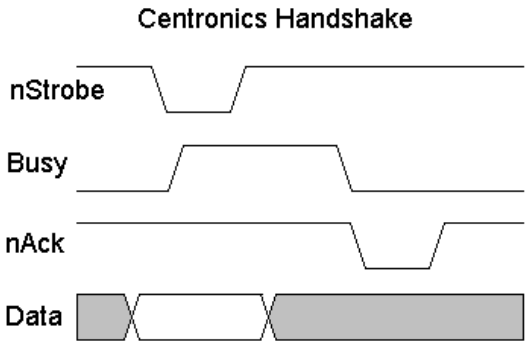
Der blev brugt et 25 pin canon-stik i Computeren og typisk et stik i en printer som vist her til højre.



(8 bit parallel-port)

PC		Printer
DB25 socket		CP36 socket
1	Strobe →	1
2	D0 →	2
3	D1 →	3
4	D2 →	4
5	D3 →	5
6	D4 →	6
7	D5 →	7
8	D6 →	8
9	D7 →	9
10	Acknowledge ←	10
11	Busy ←	11
12	Paper Empty ←	12
13	Select ←	13
14	Auto Linefeed →	14
15	Error ←	32
16	Init →	31
17	Select In →	36
18-25	Signal GND —	19-30
	Shield —	17
	+5 VDC (opt.) —	18





Et billede af nogle af Handshake-signalerne

Parallel kommunikation, men ikke særlig lang rækkevidde!

CP/M (Control Program for Microcomputers) var et operativsystem til microcomputere baseret på Intel 8080/85 og Zilog Z80 microprocessore.

Det blev udviklet af Digital Research, Inc.

Her vist en ” COMET”, som skolen havde mange af i 80érne.

RS232

En anden måde at sende signaler til en printer var via en seriel forbindelse, kaldet RS232.

Nu til dags findes disse stik typisk ikke på vore PC-er. Der bruges i stedet fx USB-stik.

Mange fysikapparater har hidtil været koblet til PC-ere via RS232-stik. Så her kan man i dag blive nødt til at lave noget konvertering fra RS232 til USB.

Der kan købes konvertere, der omdanner USB- til RS232.
Der findes også USB til andre serielle protokoller, endog konvertere til parallel Centronic protokol findes! Dvs. ældre udstyr godt kan kobles på en USB-port.

For at få konverteren til at virke, skal der installeres nogle drivere, og de får PC-en til at ”tro”, at den har en ny hardware-port installeret.

Seriel kommunikation var ikke så hurtig som parallel. Der skulle jo overføres 1 bit ad gangen.

Men fælles var, at rækkevidden kun var få meter.

Nyere bussystemer mellem microcontrollere:

De bussystemer, der er forsøgt beskrevet her i det følgende, er serielle systemer, der kan bruges direkte mellem microcontrollere.

På flg. links findes en fin sammenligning af forskellige bussystemer.
http://www.embedded.com/design/connectivity/4023975/Serial-Protocols-Compared
http://www.maximintegrated.com/app-notes/index.mvp/id/3967

Fra Microcontroller til microcontroller. TxD RxD

I "vores" microcontroller, 8051 eller ATMEGA328, som bruges i Arduino Uno, er der indbygget en **UART**, der står for Universal Asynkron Receiver Transmitter. Denne kan både sende og modtage serielle signaler. Dette betyder, at 2 uC'er kan kobles sammen.

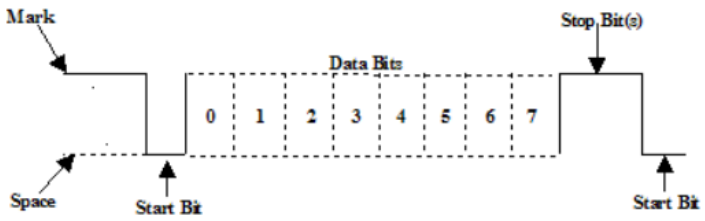
Data kan sendes både:

Synkron	Der sendes både et signal plus et clock-signal, så modtageren ved, når der skal klockes ind i et skifteregister. Der skal bruges 2 signal-ledninger, - og fælles stel, Gnd, dvs. i alt 3 ledninger.
Asyncron	Data afsendes og kommer til modtageren på et tilfældigt tidspunkt. Derfor skal der være "aftalt" en protokol, dvs. at startbit, stopbit, baudrate osv. skal være kendt. Det er nok med 1 signalledning, plus nul.

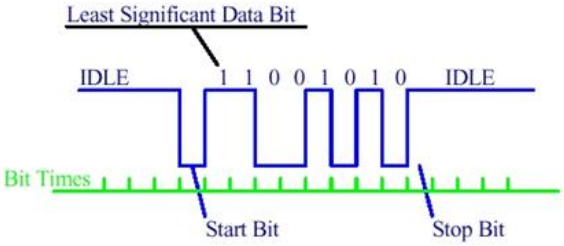
I vores 8-bit uC sendes "pakker" eller bytes asynkront. Der sendes 1 Startbit, 8 databit, og 1 stopbit.

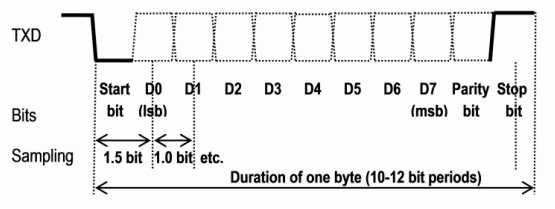
Signaler kan sendes direkte fra uC til uC, dvs. med signalniveauer på 0 og 5 Volt. Tx fra den ene uC skal forbindes til Rx på den anden uC.

Dette medfører dog ret begrænset rækkevidde pga. evt. støjpåvirkning, der kan medføre at signalet forvanskes så modtageren læser forkert.

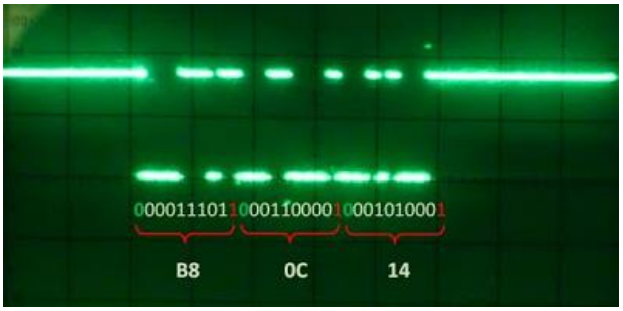


Men det kan sagtens bruges til at koble to uC sammen på samme print og over kortere afstande. Fx kan man lade én uC kontrollere et tastatur, og sende de indtastede data til en anden uC via UART´en.





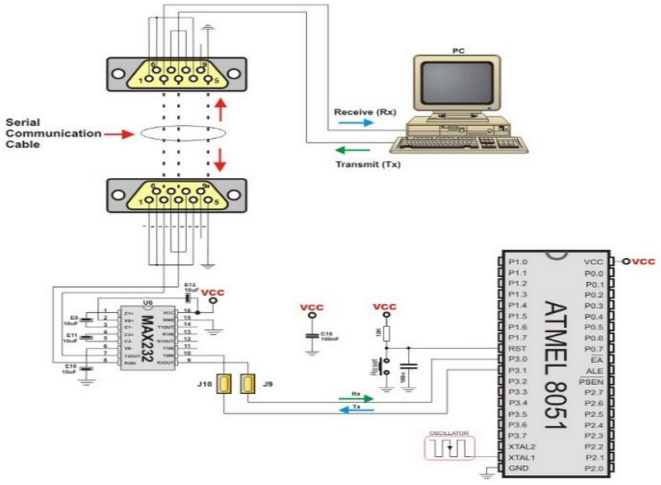
Til højre ses et eksempel på et scoop-billede af tre byte, sendt direkte fra en uC.



RS232

Ønskes større rækkevidde, kan man bruge "protokollen" der kaldes RS232. Her bruges lidt højere spændinger i signalet end de 5 Volt direkte fra uC'en, og "omvendt polaritet".

Der findes kredse, der via ladningspumpe-princippet fra 5 Volt genererer plus/minus 12 Volt. Fx Max232 fra Maxim.



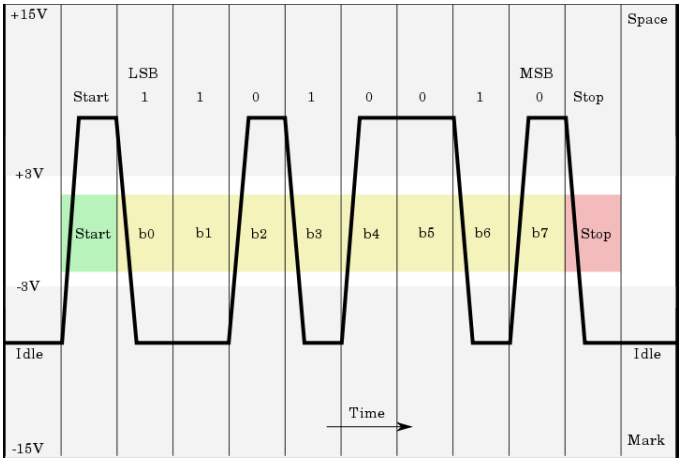
Et "1"-tal sendes som minus 12 volt, og et "0" som plus 12 volt.

Valide spændinger er dog mellem ± (3 og 12 Volt.)

Her ses en graf over signalerne i en transmission af en pakke på 8 databit efter RS232-standarden.

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

Transmissionsafstanden er dog ikke garanteret 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 efter RS232 - standarden i "gamle dage" var ordnet efter ASCII-tabellen.

Opgave:

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 bruges evt. et antal handshake-signaler, til fx at fortælle afsenderen, at modtageren er optaget, dvs. Busy, – eller at den ikke er færdig med at behandle de forrige data !!.

RS485

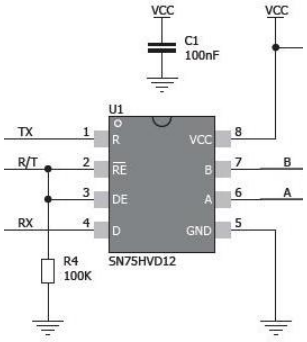
Se Youtube om RS485 (1:08)

Ønskes en meget stabil og meget langtrækkende signaloverførsel, kan man med fordel vælge at bruge standarden kaldet RS485.

Igen bruges eksterne IC-er til at omforme et serielt signal, der skal sendes mellem to uC-er. IC-erne sidder umiddelbart ved Uc-erne, og konverterer 0-5 Volt signal til et balanceret signal.

Ic-erne kan fungere som enten sender eller modtager, Tx eller Rx. Forbindes pin 2 og 3 til en pin på Uc'en vil et højt gøre Ic'en til sender, og et lavt til modtager.

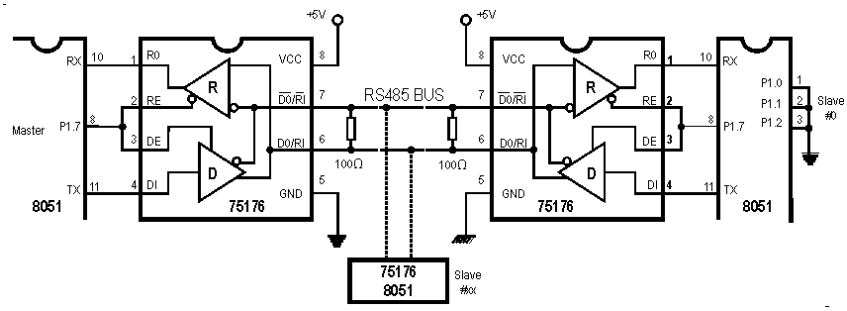
Der findes flere typer IC-er, fra forskellige firmaer, med samme funktion.



Se Kilde:

Her ses et system med en master og et par slaver.

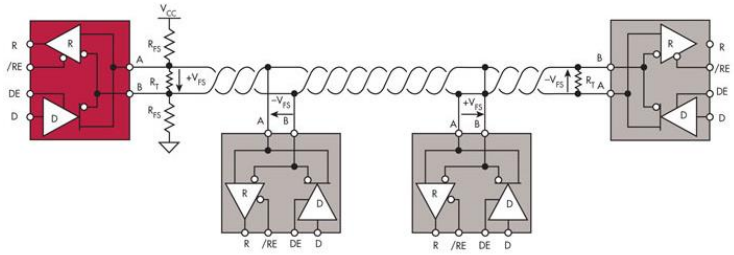
LTC485, SN75176, ADM485 m. fl.



Kilde: http://www.edaboard.com/thread77297.html

Her ses et eksempel på hvordan flere sendere / modtagere kan kobles sammen på et RS485-netværk.

Hver RS485 kreds er forbundet til en uC.



I dette eksempel ses, at signal-ledningerne skal være snoet. Se herom: Og der ses, at der i hver ende skal være en modstand mellem signalledningerne, kaldet A og B. Jeg plejer at bruge en 120 Ohm modstand.

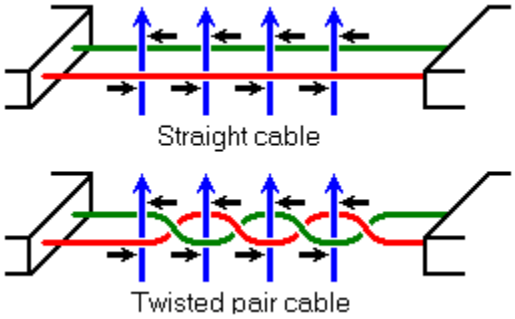
Termineringsmodstand, se: Åben kabel: https://www.physicsclassroom.com/mmedia/waves/free.cfm
Og kortsluttet: https://www.physicsclassroom.com/mmedia/waves/fix.cfm

Og se her, med animation af både åben og kortsluttet:
https://electronics.stackexchange.com/questions/171557/what-and-whys-of-termination

Hvorfor er det vigtigt: ?? Hvordan udbredes pulser i ledninger ? Momentant ?? Hvad er udbredelseshastigheden ??

The speed of light in vacuum is 2.998×10^8 m/s. In coaxial cable, the speed of an electrical signal is about 2/3 of this. Se scop-billeder af reflekteret signal her:

Parsnoning modvirker støj:



Standarden RS485 benytter et balanceret signal, på kun 2-ledere. Stel skal ikke med.

Rækkevidden er ca. op til 1 km. med en spænding på blot 5 Volt. Det er dog en betingelse, at lederne er snoede, for at modvirke indstrålet elektrisk støj.

De blå pile på tegningen repræsenterer indstrålet magnetfelt, og de sorte den inducererede strøm.

På nederste billede ses, at hvis ledningerne er snoet kabel ophæves støj-påvirkningen.

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

*In the picture above, noise is generated by magnetic fields from the environment. The picture shows the magnetic field lines and the noise current in the **RS485** data lines that is the result of that magnetic field. In the straight cable, all noise current is flowing in the same direction, practically generating a looping current just like in an ordinary transformer. When the cable is twisted, we see that in some parts of the signal lines the direction of the noise current is the opposite from the current in other parts of the cable. Because of this, the resulting noise current is many factors lower than with an ordinary straight cable.*

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

Each of the four pairs in a Cat 5 cable has differing precise number of twists per meter to minimize crosstalk between the pairs

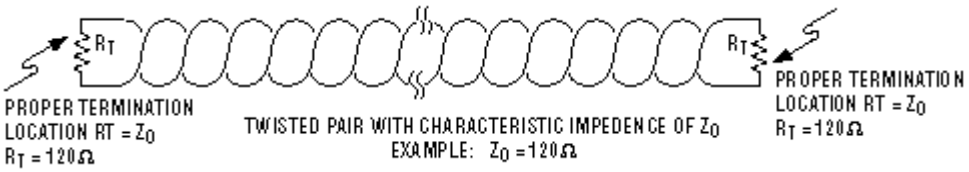
Individual twist lengths

By altering the length of each twist, crosstalk is reduced, without affecting the characteristic impedance. The distance per twist is commonly referred to as pitch, twist rate or twist periodicity.

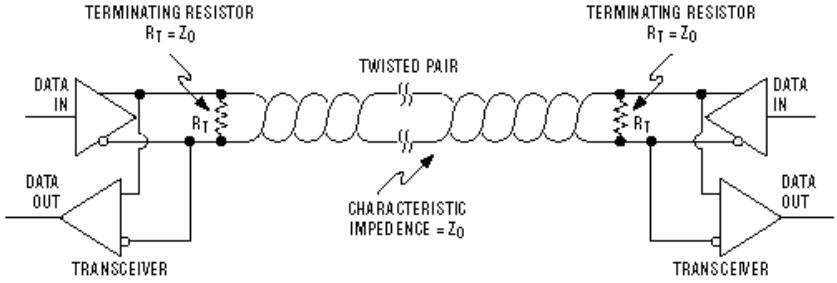
http://en.wikipedia.org/wiki/Category_5_cable

Twist længder er fx. Fra 1,53 cm til 1,94 cm pr twist.

Figuren viser et twisted par med terminering.



http://www.maxim-ic.com/app-notes/index.mvp/id/763

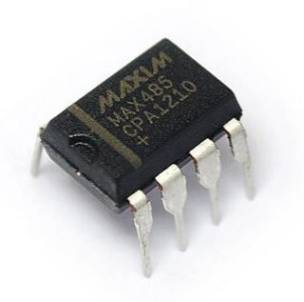


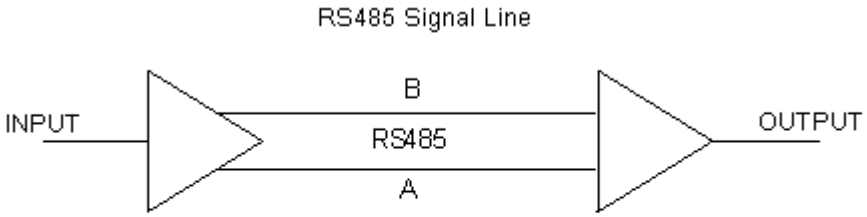
Der bruges normalt termineringsmodstande på 120 Ohm. Svarende til kablets impedans.

A – og B – Signalet:

Til at kode og til at afkode signalet bruges en lille IC. De to ledninger i bussen kaldes for A og B.

De forbindes til kredsens A og B-terminaler.





INPUT	A	B	OUTPUT
0	1	0	0
1	0	1	1

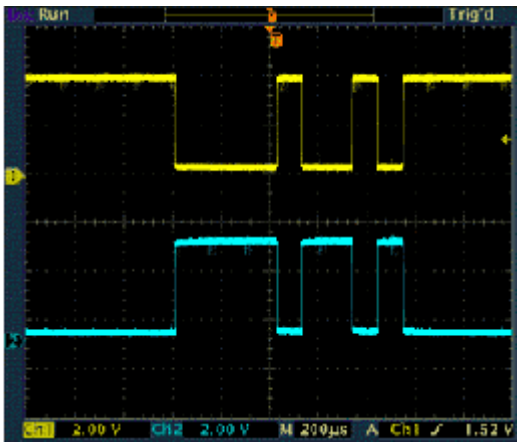
Et digitalt signal på input konverteres til et differential-signal på bussen. Et højt (logisk 1) svarer til at de to output-ledninger har U_A større end U_B. Et logisk 0 er modsat. $U_B > U_A$.

http://www.windmill.co.uk/rs485.html

Normalt er et RS-485 modtagers output "1" hvis spændingen på A er mere end 200mV højere end på B, og "0" hvis B > A med 200mV eller mere.

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 skal termineres med 120 Ohm i hver ende.

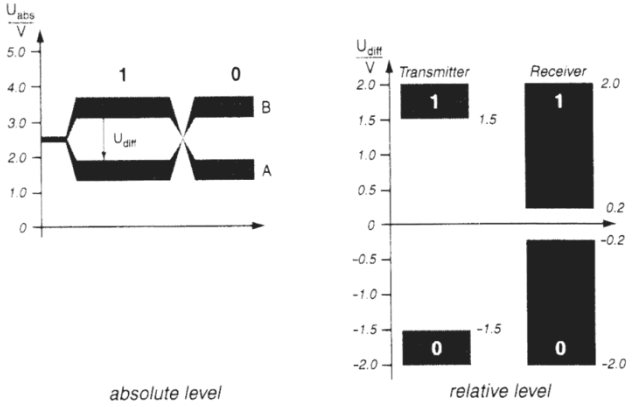


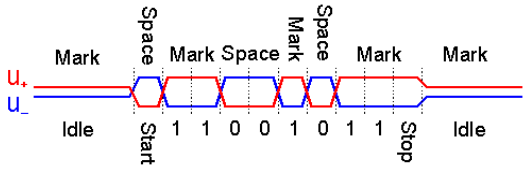
Kilde: http://www.maxim-ic.com/appnotes.cfm/an_pk/723/

Her ses en anden graf over signalerne:

Transmitterens spænding er ”høj” i forhold til den nødvendige spænding for modtageren. Herved opnås også støjimmunitet.

Det højre billede viser signalets spændingsniveauer på sendersiden sammenlignet med den større tolerance på modtagersiden.





I RS485 kan bruges bus-frekvenser op til 12 Mbit/s opnås.

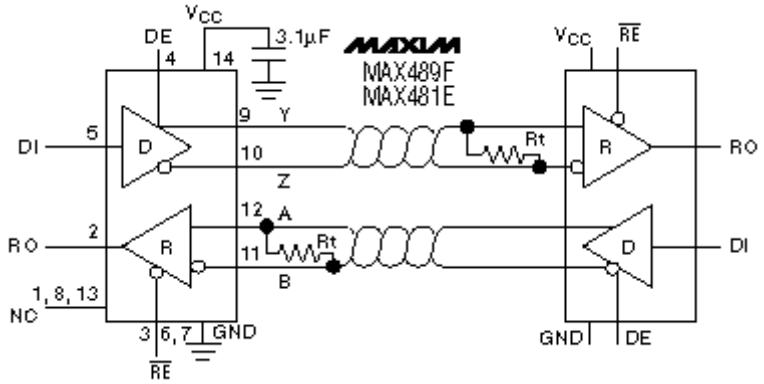
Max buslængde afhænger af overførselshastigheden. Den absolut største længde er 1200 meter ved en datarate på 93,7 Kbit/s med maksimal 32 enheder på bussen. Men ved hjælp af repeatere, dvs. signalforstærkere, kan der kobles 4 grupper a’ 32 devices sammen.

En regel for standard kabler lyder, at Baudraten ganget med kabel-længden i meter skal være mindre end 10^8. Så fx, hvis et kabel er 100 meter lang, fås en max baudrate på 1Mbit/s.

Man behøver ikke altid at bruge TP-kabler. Ved små afstande er alm. telefonkabler gode nok. Eller almindelig 2-ledet lampe-ledning.

Terminering på 120 ohm i begge ender behøves ikke altid ved små længder kabel.

Duplex



Ved simplex sendes kun i den ene retning ad gangen.

Skal der sendes i begge retninger samtidigt, må man bruge IC'er, der kan håndtere det, - eller evt. 4 alm. 8-pins LTC485.

Pins skal forbindes som flg.:

		Koblet som:	
Pin	Pin	Sender	Modtager
1	Reciever Out	NC	Rx
2	Reciever Out enable	5V	0
3	Driver output enable	5V	0
4	Driver input	Tx	0
5	GND	0	0
6	Signal	A	A
7	Signal	B	B
8	Vcc	5V	5V

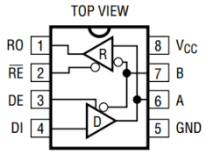
LTC485 Transmitting

INPUTS			LINE CONDITION	OUTPUTS	
$\overline{RE}$	DE	DI		B	A
X	1	1	No Fault	0	1
X	1	0	No Fault	1	0
X	0	X	X	Z	Z
X	1	X	Fault	Z	Z

LTC485 Receiving

INPUTS			OUTPUTS
$\overline{RE}$	DE	A – B	R
0	0	≥0.2V	1
0	0	≤–0.2V	0
0	0	Inputs Open	1
1	0	X	Z

LTC485, DS75176,



http://pdf1.alldatasheet.net/datasheet-pdf/view/8465/NSC/DS75176.html

Læs evt. mere: http://www.emcu.it/MCUandPeriph/RS485/RS485uk.html

RS 485 exists in two versions: 1 TwistedPair or 2 TwistedPairs.

- Single TwistedPair RS 485

  In this version, all devices are connected to a single Twisted Pair. Thus, all of them must have drivers with tri-state outputs (including the Master). Communication goes over the single line in both directions. It is important to prevent more devices from transmitting at same time (software problem).

- Double TwistedPair RS 485

  Here, Master does not have to have tri-state output, since Slave devices transmit over the second twistedpair, which is intended for sending data from Slave to Master.
  Sometimes you can see a RS 485 system in a point-to-point system. It is virtually identical to RS 422; the high impedance state of the RS 485 output driver is not used. The

only difference in hardware of the RS 485 and RS 422 circuits is the ability to set the output to high impedance state.

http://www.pccompci.com/The_RS-_485_Serial_Port.html

I2C, I²C, IIC, eller 2-wire-bus.

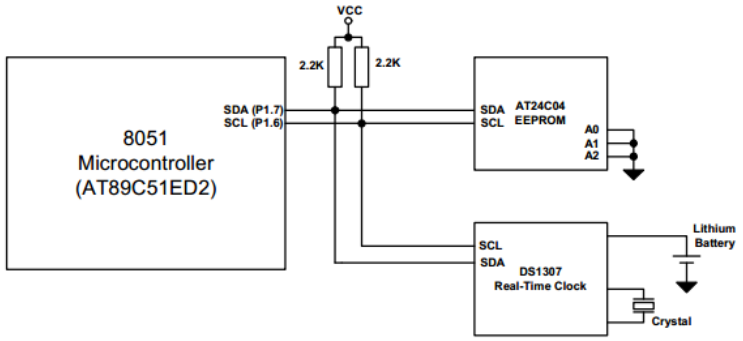
Bussen I2C har flere navne. IIC står for "Inter IC Communications" eller **I**nter **I**ntegrated **C**ontroller. Den kaldes også nogle steder for 2-wire bus.

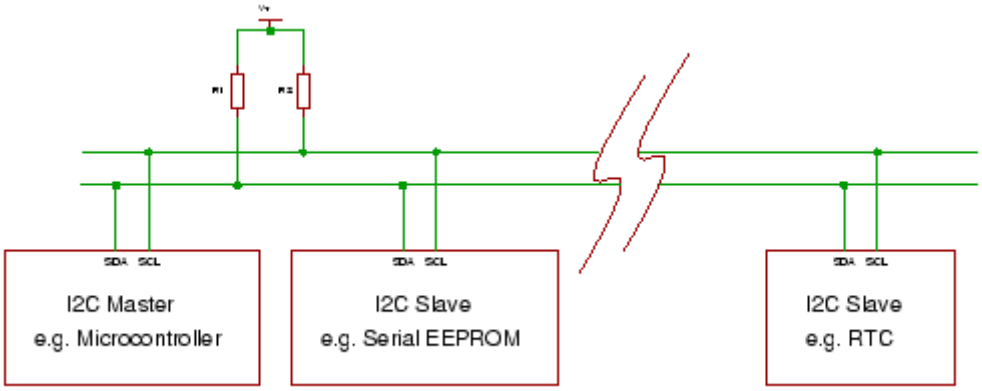
Se video på dette link: http://tronixstuff.com/2010/10/20/tutorial-arduino-and-the-i2c-bus/

I2C bussen er skabt af Philips I 80érne. Dengang var standarden en max clockfrekvens på 100 KHz, men nu findes en hurtigere på 400 KHz.

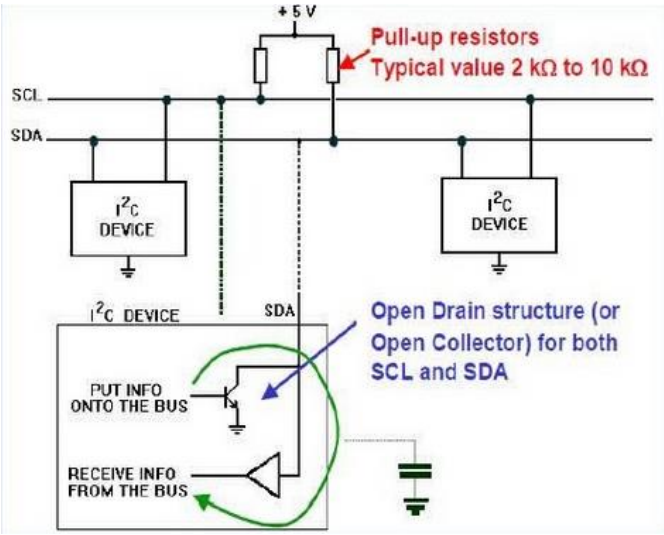
I2C-bussen er en bus, hvor de samme to signal-ledninger bruges til at sende signaler til et antal tilkoblede IC-er.

Der skal bruges 2 signalledere og en nul-leder.





Denne skitse illustrerer hvordan IC-erne koblet på bussen kan trække signalet lavt med en Open Collector eller Open Drain.



http://www.scribd.com/doc/77997878/Final-Project

I2C er en bidirektionel 2 wire-bus, der bruges til kommunikation mellem chips i moderne elektronisk udstyr. fx gemmes opsætninger af kanaler osv. i et TV i EEPROM's via IIC-bussen.

Vha. bussen kan man forbinde en uC til fx en seriel EEPROM, Temperatur IC, port-kredse osv.

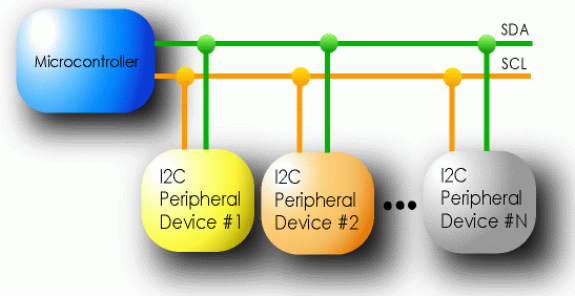
Alle de tilkoblede kredse kaldes slaver.

Al kommunikation sker med et clock-signal og et data-signal. Plus selvfølgelig stel.

Bussen bruges kun til at koble et antal kredse til en kontroller internt på et printkort. Ikke til kommunikation "længere væk".

Det ses, at der kun anvendes 2 wires, der forbindes til alle enhederne i systemet. Plus selvfølgelig nul. Dette er ret smart, idet der så ikke optages et antal pins i controlleren for hver tilkoblet kreds.

I andre bussystemer angiver processoren med en separat ledning til hver tilsluttet kreds, - en Chip Select, - at den er interesseret i at kommunikere med netop denne kreds.

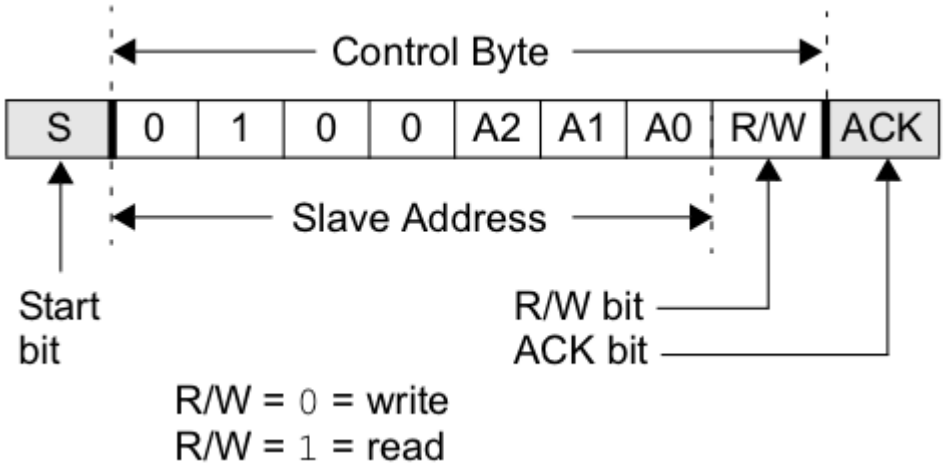


Men i I2C-systemet sker dette valg af tilsluttede kredse også på de to signalledninger.

Kredsene er født med en "unik" ID-kode, også kaldet slaveadresse. Den består af en bit kode som masteren sender ud på bussen. Koden kan være mellem 4 til 7 bit.

De tilsluttede kredse vil derfor kun reagere, når processoren ”kalder dem op” ved at sende deres ID-kode.

De enkelte chips på bussen kaldes op ved deres adresse eller ID. Adressen er (normalt) 7 bit – og det 8. bit fortæller, om man kalder chippen op for skrive eller læseadgang.

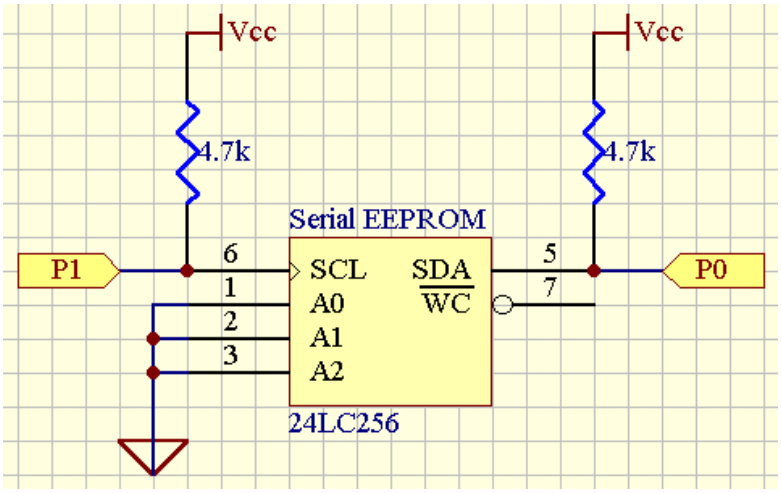


Eksempel med EEPROM

Control Byte Allocation { 24LC256 EEPROM }							
1	0	1	0	A2	A1	A0	R/W

Den her viste en IC, en EEPROM, har kun en 4-bit ID. De 4 mest betydende bit i Slaveadressen er bestemt af fabrikanten, og de næste 3 bit er ført ud til pins, hvilket bevirker, at man fx med dipswitche kan bestemme 3 af IC-ens adressebit. Herved kan man sætte hele 8 af samme IC på bussen. Pin’ene kaldes A2, A1 & A0.

Værdien af bittet R/W fortæller slaven, om der ønskes læse eller skriveadgang.



Forbindelserne ser fx således ud!!

IC-ens ID, kan aflæses i dens datablad.

Den samlede ID bliver altså ???? 000 r/w,

The pull-up resistors are required and can vary from 4.7k to 10k.

Følgende skema angiver forskellige slave-kreds-typer og deres ID-kode. (Eller adresser)
Nogle af kredsene har mere end 4 bit ID.

ASSIGNED I²C BUS ADDRESSES

PART NUMBER	FUNCTION	I²C ADDRESS						
		A6	A5	A4	A3	A2	A1	A0
—	General call address	0	0	0	0	0	0	0
—	Reserved addresses	0	0	0	0	X	X	X
PCD3311/12	Tone generator DTMF/modem/musical	0	1	0	0	1	0	A
PCF8200	Voice synthesizer (male or female)	0	0	1	0	0	0	0
PCF8566	96-segment LCD driver 1:1–1:4 Mux	0	1	1	1	1	1	A
PCF8568	LCD row driver for dot matrix displays	0	1	1	1	1	0	A
PCF8569	Column driver for dot matrix displays	0	1	1	1	1	0	A
PCF8570/71	256 × 8, 128 × 8 static RAM	1	0	1	0	A	A	A
PCF8570C	256 × 8 static RAM	1	0	1	1	A	A	A
PCF8573	Clock/calendar	1	1	0	1	0	A	A
PCF8574	I²C bus to 8-bit bus converter	0	1	0	0	A	A	A
PCF8574A	I²C bus to 8-bit bus converter	0	1	1	1	A	A	A
PCF8576	160-segment LCD driver 1:1–1:4 Mux	0	1	1	1	0	0	A
PCF8577	64-segment LCD driver 1:1–1:2 Mux	0	1	1	1	0	1	0
PCF8577A	64-segment LCD driver 1:1–1:2 Mux	0	1	1	1	0	1	1
PCF8578	Row/column LCD dot-matrix driver	0	1	1	1	1	0	A
PCF8579	Row/column LCD dot-matrix driver	0	1	1	1	1	0	A
PCF8581	128-byte EEPROM	1	0	1	0	A	A	A
PCF8582	256 × 8 EEPROM	1	0	1	0	A	A	A
PCF8583	256 × 8 RAM with clock/calendar	1	0	1	0	0	0	A
PCF8591	4-channel, 8-bit A/D plus 8-bit D/A	1	0	0	1	A	A	A
PCF8594	512-byte EEPROM	1	0	1	0	A	A	A
SAA1064	4-digit LED driver	0	1	1	1	0	A	A
SAA1136	PCM-Audio indent-word interface	0	0	1	1	1	1	0
SAA1300	5-bit high current driver	0	1	0	0	0	A	A
SAA5243/45	Enhanced teletext circuit	0	0	1	0	0	0	1
SAA7191	S-VHS digital multistandard decoder "square pixel"	1	0	0	0	1	A	1
SAA7192	Digital color space converter	1	1	1	0	0	0	A
SAA7199	Digital encoder	1	0	1	1	0	0	0
SAA9020	Field memory controller	0	0	1	0	1	A	A
SAA9051	Digital multi-standard TV decoder	1	0	0	0	1	0	1
SAA9068	(PIPCO) Picture-in-picture controller	0	0	1	0	0	1	A
SAB3035/36/37	(CITAC) CPU interface for tuning and control	1	1	0	0	0	A	A
SAF1135	Data line decoder	0	0	1	0	0	A	A
TDA4670	Picture signal improvement circuit	1	0	0	0	1	0	0
TDA4680	Video processor	1	0	0	0	1	0	0
TDA8421	Hi-fi stereo audio processor	1	0	0	0	0	0	A
TDA8425	Audio processor w/loudspeaker channel	1	0	0	0	0	0	1
TDA8440	Switch for CTV receivers	1	0	0	1	A	A	A
TDA8442	Interface for color decoders	1	0	0	0	1	0	0
TDA8443	YUV/RGB interface circuit	1	1	0	1	A	A	A
TDA8444	Octuple 6-bit DAC	0	1	0	0	A	A	A
TDA8461	PAL/NTSC color decoder	1	0	0	0	1	0	A
TEA6100	FM/IF and tuning interface	1	1	0	0	0	0	1
TEA6300/6310T	Sound fader control circuit	1	0	0	0	0	0	0
TSA5511/12/14	PLL frequency synthesizer for TV	1	1	0	0	0	A	A
TSA6057	Radio tuning PLL frequency synthesizer	1	1	0	0	0	1	A
UMF1009	Frequency synthesizer	1	1	0	0	0	A	A
—	Reserved addresses	1	1	1	1	X	X	X

X = Don't care.
A = Can be connected high or low by the user.

Se evt. flere her:

Nogle kredse er i dag så store, fx EEPROM's, at der i kredsen er større memory, end der kan vælges eller adresseres med 8 bit. For disse kredse er memory'en opdelt i BANKS, eller afdelinger. Her kan A2, A1 og A0 opfattes som Bank-valg, i stedet for IC-valg. Afhængig af størrelse kan der så kobles et færre antal kredse på bussen.

En kreds adresseres så med en byte bestående af 4 bit ID, og 3 bit device, (eller Bank).

Det kunne se sådan ud!

IC type ID				Kreds / Bank			
1	0	0	1	A2	A1	A0	R/W

En kommunikation fra en controller foregår på den måde, at controlleren først udsender en "start". Herefter udsendes en ID-byte, der får den rette IC-enhed til at reagere. I byten er med bit 0 angivet, om der ønskes læse eller skriveadgang. (R / W)

Den kreds, der adresseres, svarer efterfølgende ved at sende et ACKnoledge signal.

Ønskes fx at gemme noget i en EEPROM, sendes der fra controlleren efterfølgende fx den adresse i EEPROM'en, hvor der skal gemmes. Igen kvitterer kredsen, der også kaldes "slaven" med en ACK. Næste byte, controlleren sender gemmes så på korrekte plads.

Når controlleren er færdig, sendes en STOP ordre.

Dette er illustreret i følgende skitse.

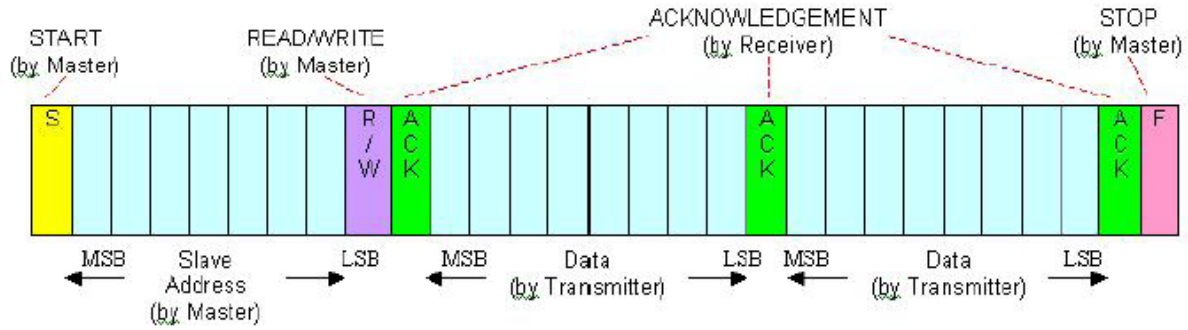
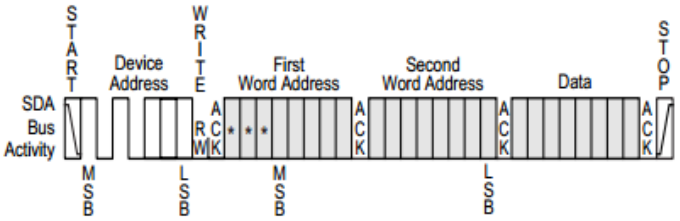


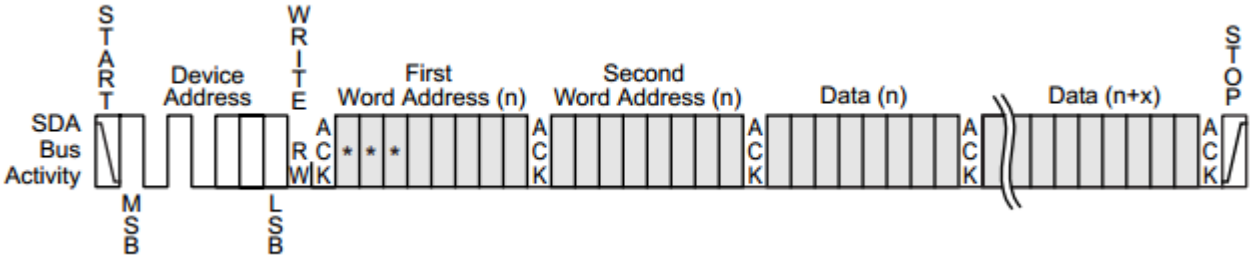
Figure 4: I2C Communication

Det er også mulig at sende flere på hinanden følgende databyte.

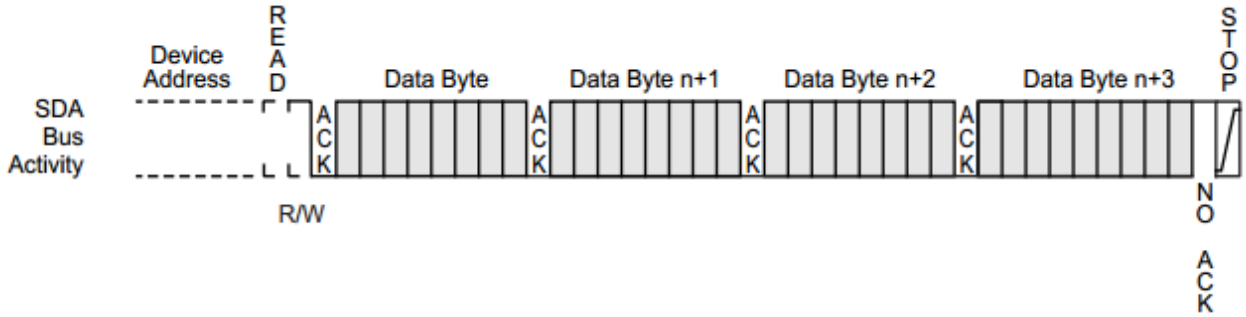
Først når der sendes en Stop-kommando fra controlleren, er transmissionen færdig.



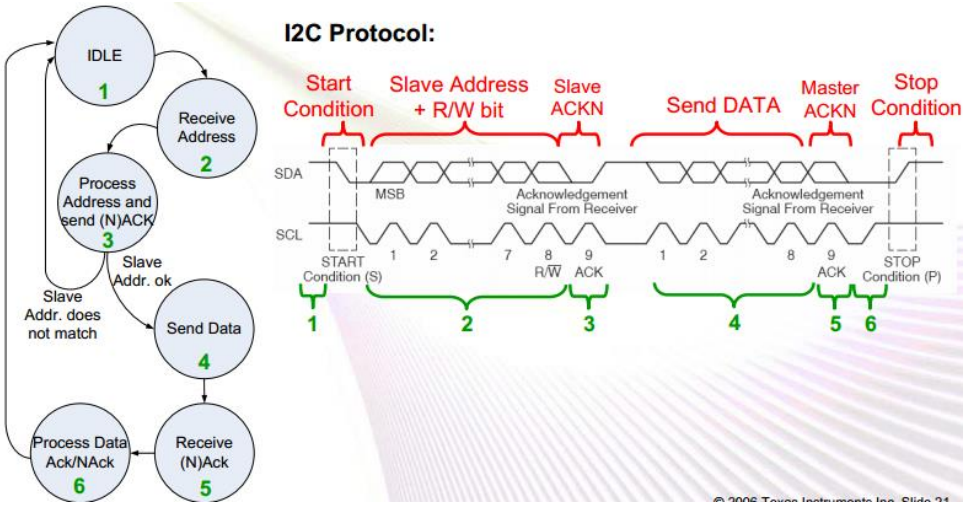
Her følger flere timing-diagrammer:



http://pdf1.alldatasheet.com/datasheet-pdf/view/106552/ISSI/IS24C16.html



http://pdf1.alldatasheet.com/datasheet-pdf/view/106552/ISSI/IS24C16.html



http://www.ti.com/lit/ml/slap117/slap117.pdf

Kort:

I Arduinoverdenen findes biblioteksprogrammer til kommunikationen.

Acknoledge kan opfattes som det 9. bit, hvor slaven trækker dataledningen lav.

Der findes større kredse, hvor det er nødvendigt at sende 2 adressebytes.

Det er ikke nødvendigt, at opretholde en konstant klockfrekvens. Derfor kan processoren godt servicere interruptrutiner undervejs.

Der findes ny protokol med 10 bit adressing og op til 400 Kbit / sek.

Diverse I2C-Kredse:

Følgende en oversigt over nogle forskellige relevante I2C-kredse.

Type	Forklaring
PCF8574	8 bit Port expander Read / Write (Read, = læs port til uC)
PCF8574A	Port expander Read / Write (Read, = læs port til uC), Anden ID-kode !!
DS 1624	Temperaturcensoren fra Dallas kan indstilles til forskellige temperaturmodes, enten kontinuerlig, eller sleepmode, med startsignal hver gang der ønskes en konvertering.
PCF8591	100 KHz 4 kanal 8 bit ADC
DS1307	Real time clock

Serielle EEPROMS

En af de interessante komponenter, der med fordel kan kobles til vores uC, er en EEPROM. Denne er en hukommelse, der kan gemmes data i, uden at de forsvinder når spændingen fjernes.

En **EEPROM** er en hukommelseskreds, der kan kobles til en uC.

Gemte data ”glemmes ikke” selv om spændingen fjernes. Men data kan selvfølgelig overskrives.

Her er vist en oversigt over EEPROM’s til forskellige “bus-familier”.

Alle data er organiseret i bytes a´ 8 bit

Size (in bits)	Bus Style		
	I^2C (24C' Series)	SPI (25C' Series)	Microwire (93C' Series)
128	24C00	-	-
256	-	-	93C06
1,024	24C01	-	93C46
2,048	24C02	-	93C56
4,096	24C04	25C040	93C66
8,192	24C08	25C080	93C76
16,384	24C16	25C160	93C86
32,768	24C32	25C320	-
65,536	24C64	25C640	-
131,072	24C128	-	-
262,144	24C256	-	-

Bemærk, at der fås kredse til hver deres bussystem!! I2C, SPI og Microwire.

Arduino og IIC-bussen:

Det vi gør her er vel nok ”bare” at includere et bibliotek, kaldet Wire.

Se god video, der først forklarer om hvordan der kommunikeres på bussen, deerefter et eksempel på at læse fra en slave: 9:57. Her:

Her findes et program til Arduino, der kan scanne efter en device-adresse

SPI, 3-wire, Microwire og 4-wire-busserne:

Fælles for disse busser, er, at de bruger en separat ledning til at vælge den chip, der ønskes kontakt med.

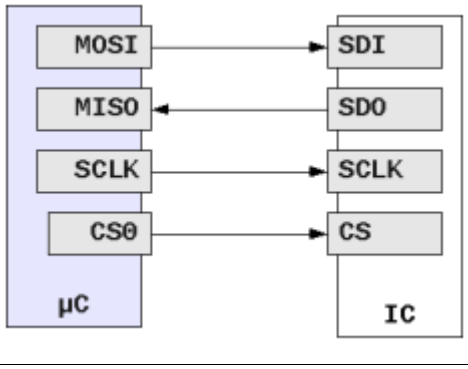
Disse busser arbejder nogenlunde ens. De laves af forskellige firmaer, og har lidt forskellige måder, de skal forbindes på og arbejder på.

Se evt. oversigt her:

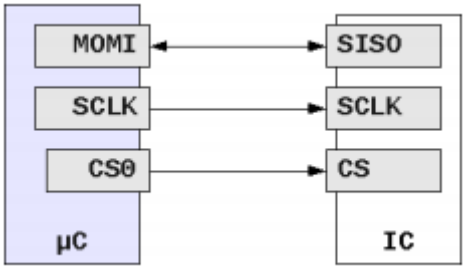
SPI står for Serial Peripheral Interface. SPI-bussen bruger 4 wirer. Chip Select, Clock, og 2 dataledninger, en i hver sin retning.

Men man kan også støde på opsætninger, hvor den ene data-ledning er udeladt, - fx hvis der kun skal læses fra en IC.

Og der findes IC-er til SPI-bussen, hvor data sendes og modtages på samme ledning. Den kan så bare være kaldt "Data". Denne type kan kun arbejde i half-duplex, dvs. data i en retning ad gangen.



SPI 4 wire setup



SPI 3 wire setup.

http://elk.informatik.fh-augsburg.de/da/da-49/da-thoms-cc.pdf

SPI er skabt af Motorola i midt 1980'erne.

Nogle gange kaldes SPI-bussen også for 4-wire serial bus.

SPI bussen kan sammenlignes lidt med microwire bussen. Den bruger lignende signal-navne og commando-protokol. Men den clock'er data ind og ud lidt anderledes. Data kan klockes med meget højere frekvens end microwire. Op til 5 MHz!

I nyere microcontrollere kan der være indbygget en SPI-port!

Fra nettet:

The SPI bus specifies four logic signals:

SCLK: serial clock (output from master);

MOSI; SIMO:	master output, slave input (output from master);
MISO; SOMI:	master input, slave output (output from slave);
SS:	slave select (active low, output from master).

Alternative naming conventions are also widely used:

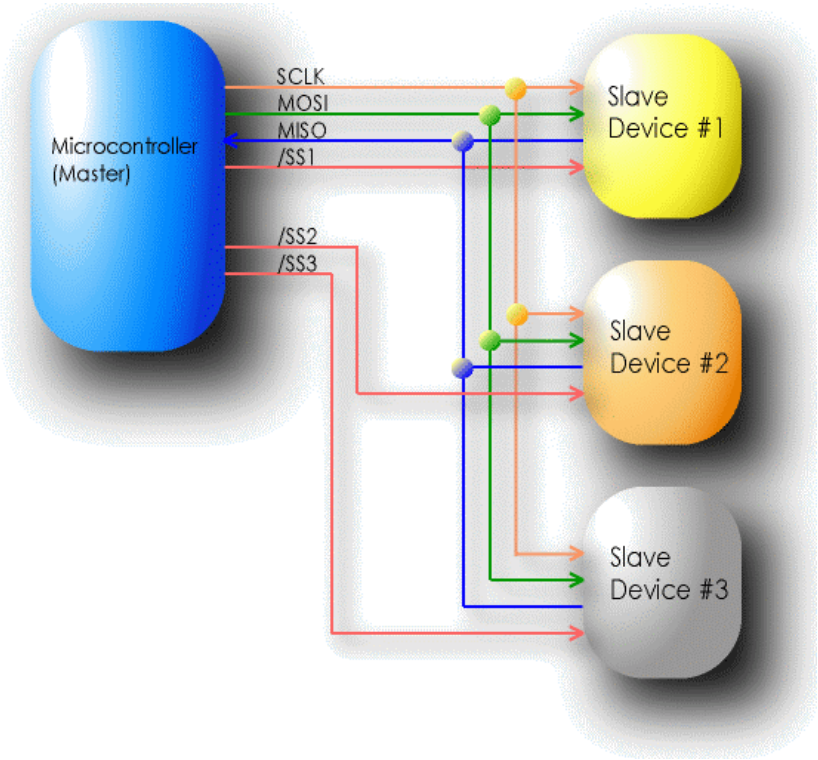
SCK; CLK:	serial clock (output from master)
SDI; DI, DIN, SI:	serial data in; data in, serial in
SDO; DO, DOUT, SO:	serial data out; data out, serial out

The SDI/SDO (DI/DO, SI/SO) convention requires that SDO on the master be connected to SDI on the slave, and vice-versa. Chip select polarity is rarely active high, although some notations (such as SS or CS instead of nSS or nCS) suggest otherwise.

SPI interfacen er command drevet. For at læse data fra Memory, vælger hosten eller masteren en device, vha. chip-select, sender READ kommando, sender adressen af den byte, der ønskes læst, og clocker så data ud af device gennem SO pin.
(Seriel Out)

Ved afslutningen de-selecter hosten igen kredsen.

Der kan være software forskelle mellem fabrikater af SPI-ic'er.



http://www.eeherald.com/section/design-guide/esmod12.html

Fra Nettet

The 3-wire nomenclature is the general category of emcompassing the SPI and Microwire busses. The Microwire is a subset of SPI with slightly different timing and data latching. This last part tripped us up at work when we tried to access an SPI EEPROM with a PIC configured for the Microwire data latching.

SPI

The Serial Peripheral Interface (SPI) is a synchronous serial bus developed by Motorola and present on many of their microcontrollers.
The SPI bus consists of four signals: master out slave in (MOSI), master in slave out (MISO), serial clock (SCK), and active-low slave select (/SS). As a multi-master/slave protocol, communications between the master and selected slave use the unidirectional MISO and MOSI lines, to achieve data rates over 1Mbps in full duplex mode. The data is clocked simultaneously into the slave and master based on SCK pulses the master supplies. The SPI protocol allows for four different clocking types, based on the polarity and phase of the SCK signal. It is important to ensure that these are compatible between master and slave.

In addition to the 1Mbps data rate, another advantage to SPI is if only one slave device is used, the /SS line can be pulled low and the /SS signal does not have to be generated by the master. (This capability is, however, dependent on the phase selection of the SCK.)
A disadvantage to SPI is the requirement to have separate /SS lines for each slave. Provided that extra I/O pins are available, or extra board space for a demultiplexer IC, this is not a problem. But for small, low-pin-count microcontrollers, a multi-slave SPI interface might not be a viable solution.
For more detail on SPI, read David and Roee Kalinsky's "Beginner's Corner: Serial Peripheral Interface" (February 2002).

Microwire

Microwire is a three-wire synchronous interface developed by National Semiconductor and present on their COP8 processor family.

Similar to SPI, Microwire is a master/slave bus, with serial data out of the master (SO), and serial data in to the master (SI), and signal clock (SK). These correspond to SPI's MOSI, MISO, and SCK, respectively. There is also a chip select signal, which acts similarly to SPI's /SS. A full-duplex bus, Microwire is capable of speeds of 625Kbps and faster (capacitance permitting).

Microwire devices from National come with different protocols, based on their data needs. Unlike SPI, which is based on an 8-bit byte, Microwire permits variable length data, and also specifies a "continuous" bitstream mode.

Microwire has the same advantages and disadvantages as SPI with respect to multiple slaves, which require multiple chip select lines. In some instances, an SPI device will work on a Microwire bus, as will a Microwire device work on an SPI bus, although this must be reviewed on a per-device basis.

Both SPI and Microwire are generally limited to on-board communications and traces of no longer than 6 inches, although longer distances (up to 10 feet) can be achieved given proper capacitance and lower bit rates.

http://www.embedded.com/design/connectivity/4023975/Serial-Protocols-Compared

The **Serial Peripheral Interface Bus or SPI** bus is a very loose standard for controlling almost any digital electronics that accepts a clocked serial stream of bits. A nearly identical standard called "Microwire" is a restricted subset of SPI.

SPI is cheap, in that it does not take up much space on an integrated circuit, and effectively multiplies the pins, the expensive part of the IC. It can also be implemented in software with a few standard IO pins of a microcontroller.

Many real digital systems have peripherals that need to exist, but need not be fast. The advantage of a serial bus is that it minimizes the number of conductors, pins, and the size of the package of an integrated circuit. This reduces the cost of making, assembling and testing the electronics.

A serial peripheral bus is the most flexible choice when many different types of serial peripherals must be present, and there is a single controller. It operates in full duplex (sending and receiving at the same time), making it an excellent choice for some data transmission systems.

In operation, there is a clock, a "data in", a "data out", and a "chip select" for each integrated circuit that is to be controlled. Almost any serial digital device can be controlled with this combination of signals.

SPI signals are named as follows:

SCLK - serial clock
MISO - master input, slave output
MOSI - master output, slave input
CS - chip select (optional, usually inverted polarity)

Most often, data goes into an SPI peripheral when the clock goes low, and comes out when the clock goes high. Usually, a peripheral is selected when chip select is low. Most devices have outputs that become high impedance (switched-off) when the device is not selected. This arrangement permits several devices to talk to a single input. Clock speeds range from several thousand clocks per second (usually for software-based implementations), to several million per second.

Most SPI implementations clock data out of the device as data is clocked in. Some devices use that trait to implement an efficient, high-speed full-duplex data stream for applications such as digital audio, digital signal processing, or full-duplex telecommunications channels. On many devices, the "clocked-out" data is the data last used to program the device. Read-back is a helpful built-in-self-test, often used for high-reliability systems such as avionics or medical systems.

In practice, many devices have exceptions. Some read data as the clock goes up (leading edge), others read as it goes down (falling edge). Writing is almost always on clock movement that goes the opposite direction of reading. Some devices have two clocks, one to "capture" or "display" data, and another to clock it into the device. In practice, many of these "capture clocks" can be run from the chip select. Chip selects can be either selected high, or selected low. Many devices are designed to be daisy-chained into long chains of identical devices.
SPI looks at first like a non-standard. However, many programmers that develop embedded systems have a software module somewhere in their past that drives such a bus from a few general-purpose I/O pins, often with the ability to run different clock polarities, select polarities and clock edges for different devices.

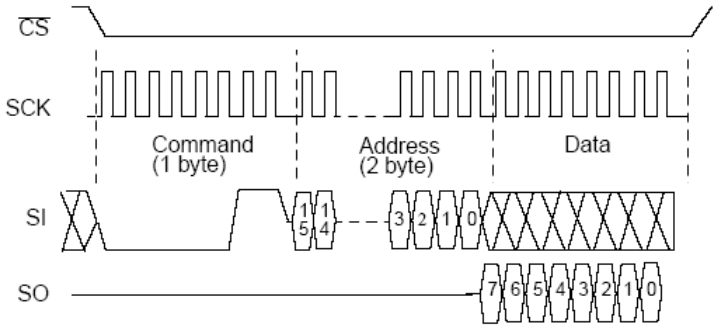
Fra: http://www.coridium.us/ARMhelp/index.htm#page=HwSPIMicrowire.html

SPI-Bussen og 3-wire sammenlignet:

De to busser, 3-wire og SPI er næsten lig hinanden. Her en sammenligning mellem de to:

Serial Interface	Signal names	Signal Descriptions	BUS Speed (Typical)	Data Format
3-WIRE	DQ	Data In	500kHz to 5 MHz	LSB first, MSB last
	Cout	Data Out		
	active-low RST	active-low Reset		
	CLK	Clock		
SPI	SDI	Data In	10MHz	MSB first, LSB last
	SDO	Data Out		
	active-low SS	active-low Slave Select		
	SCK	Clock		

http://www.maxim-ic.com/appnotes.cfm/an_pk/169



Her en skitse af signalerne på ledningerne.

En Chipselect, en clock og to dataledninger.

For mere se:

http://www.mct.de/faq/spi.html

http://www.rpi.edu/dept/ecse/mps/Coding_SPI_sw.pdf

http://www.mikroe.com/en/books/8051book/ch4/ (for SPI indbygget i en uC)

Microwire:

National Semiconductors bus "Microwire" er næsten det samme som SPI. Nogle gange kaldes SPI for "four wire" serial bus.

Det er en af de ældste serielle busser. Supporterer nogle af de billigste serielle hukommelseskredse. Bussen er hurtig nok til mange applikationer, men supporterer ikke så mange kredse

Kræver flere portpins end I2C

Selvom Microwire bussen har nogle fordele, er det den første, der vil forsvinde. (www.xicor.com)

Maxim 3-wire

The Maxim 3-wire interface is found on the DS1620 and some other ICs from Maxim The data flow to and from the DS1620 is multiplexed on only one pin (DQ) while SPI needs two separate signals (MOSI, MISO).

One variant of SPI uses single bidirectional data line (slave out/slave in, called SISO) instead of two unidirectional ones (MOSI and MISO). Clearly, this variant is restricted to a half duplex mode. It tends to be used for lower performance parts, such as small EEPROMs used only during system startup and certain sensors, and Microwire. As of this writing, few SPI master controllers support this mode; although it can often be easily bit-banged in software. When someone says a part supports SPI or Microwire, you can normally assume that means the four-wire version.

However, when someone talks about a part supporting a 3-wire serial bus you should always find out what it means: standard 4-wire SPI, without the chip select pin from that count, since most buses use chip selects but only three wires carry "real" signals; (More, sometimes with an unshared SPI bus segment the device's chip select will be hard-wired as "always selected".) "real" 3-wire SPI; or even a RS232 cable with RXD, TXD, and shield/ground, or an application-specific signaling scheme.

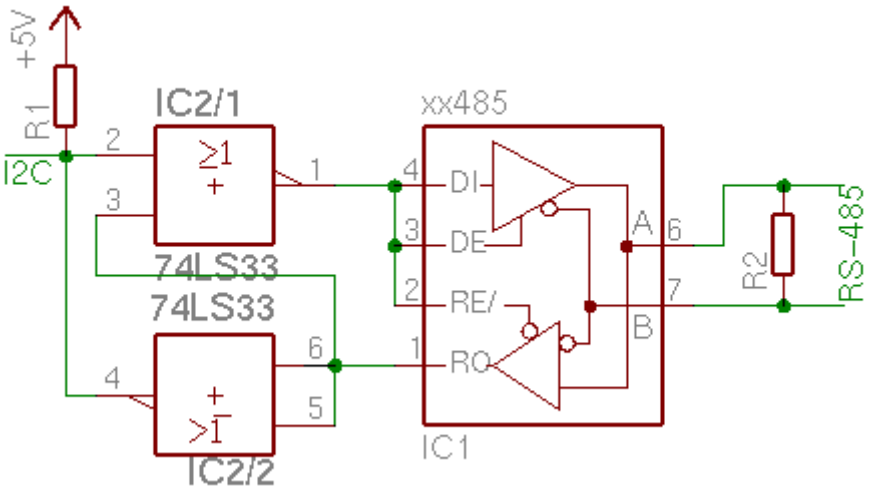
Fra: http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

Andre firmaer kalder bare deres signal-ledninger for Data, Clock og CS (Chip Select)

Ide: Kan man koble en RS485 bus på en I2C-bus?? Se hvordan her:

Konverter fra I2C til 485:

Dette skal dubleres og sættes både på Data og Clock-signalet !!



http://xj900diversion.free.fr/bus/I2C%20-%20RS-485%20adapter.htm

1-wire Bus, MicroLAN, (Kort)

1-wire bussen er fra Dallas Semiconductor / Maxim

Dallas / Maxim 1-wire bussen bruger en enkel datalinje til at sende og modtage data. Derfor kaldes den 1-Wire.

Hver 1-Wire enhed skal også forbindes til fælles ground, og optional til en power supply. Hvis powersupply udelades, kan 1-wire enheden vist strømfødes "parasitisk" fra dataledningen.

1-wire bussen er normalt holdt høj, med en 4,7 K modstand til Ucc. Når der sendes data, trækkes bussen kortvarigt lav i henhold til bittene.

Busprotokollen arbejder med et antal definerede signaltyper. En reset puls, "presence puls", Write 0, Write 1, Read 0, Read 1.

Hvert signal har timing parameter som skal strengt overholdes. Derfor er det vigtigt, at systemet er i stand til at generere akkurate timeslots.

For fuldstændig oversigt over 1-Wire signalling refereres til dokumentation på Dallas/Maxim's hjemmeside.

Data sendes ved at trække data linien lav. Kortere end 15uS er et "1", længere end 60uS = "0".

Kaldes for q-PWM Code Bus System

Kredse:

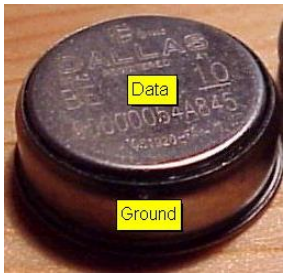
Der finds et antal IC-kredse, der anvender 1-Wire: Fx:

DS18x20, Temperaturcensor I TO92 hus. DS18S20 giver 9 bit / 0,5 grader C opløsning, og DS18B20 giver op til 12 bits opløsning, 0,0625 grader. Begge med tolerancer inden for 0,5 grader.

DS 2401, Bruges til serial nummer, eller ID-chip !. Fås i SMD eller TO92 hus ! IC-en sender en 8 bit familiekode, (Type of device), dernæst 48 bit unik kode, og 8 bit CRC, Cyclic Redundancy Check

Dvs. at den serielle nummerkreds er programmeret med en ud af 2^{48} koder, eller 2,8 E14 koder !

Den kendes også fx som iBUTTON



DS 1920 Temperaturcensor ! og IC til adgangskontrol.

Se: http://en.wikipedia.org/wiki/1-Wire