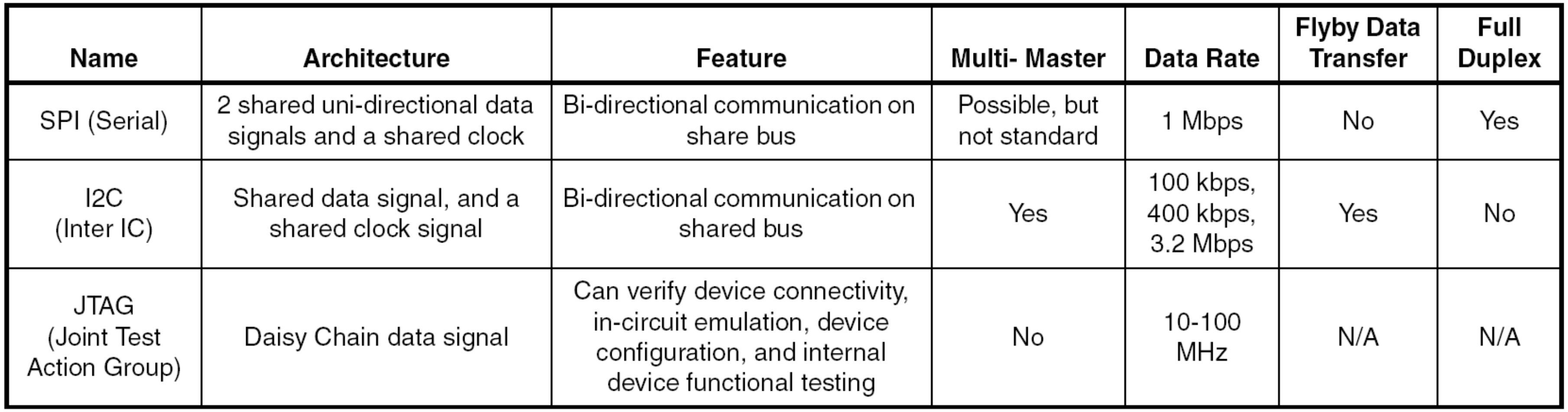
Le reti Seriali Sincrone

SPI Serial Peripheral Interface

I2C Inter Integrated Circuit (Inter IC)

JTAG Joint Test Action Group

Comparazione tra le caratteristiche dei protocolli seriali



Name	Architecture	Feature	Multi- Master	Data Rate	Flyby Data Transfer	Full Duplex
SPI (Serial)	2 shared uni-directional data signals and a shared clock	Bi-directional communication on share bus	Possible, but not standard	1 Mbps	No	Yes
I2C (Inter IC)	Shared data signal, and a shared clock signal	Bi-directional communication on shared bus	Yes	100 kbps, 400 kbps, 3.2 Mbps	Yes	No
JTAG (Joint Test Action Group)	Daisy Chain data signal	Can verify device connectivity, in-circuit emulation, device configuration, and internal device functional testing	No	10-100 MHz	N/A	N/A

Using serial buses instead of parallel buses can cut component costs, space, and power consumption.

Lower system performance is typical, but could be acceptable in many applications.

Complete systems can be built using either SPI or I2C.

JTAG is typically used only during product development, manufacturing, or servicing.

SPI: Serial Peripheral Interface

Using a serial bus in a system has many advantages over a parallel bus.

1. Lower component cost
2. Smaller printed circuit board
3. Simplified design
4. Typically, lower power consumption

The trade-off for these advantages is inherently lower data transfer rates than a parallel bus system.
In serial buses only one wire carries data into and/or out of the device.
In some cases there is one signal for carrying data into the device another different signal for carrying data out of the device.
On the other hand in parallel buses many signals transfer data simultaneously into and out of the device.

SPI: Serial Peripheral Interface

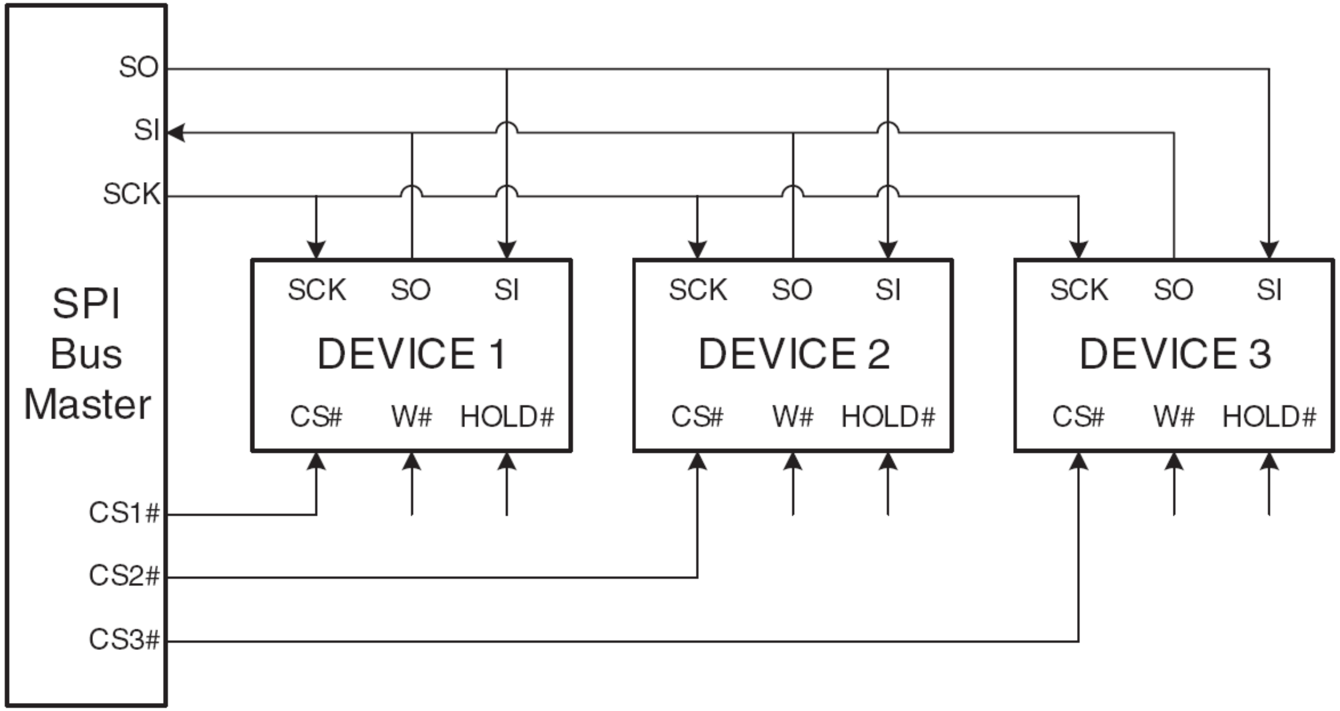
SPI is an acronym for *Serial Peripheral Interface bus. SPI allows for full-duplex data transfers. Typically, there* is only one bus master, all other SPI devices are slaves.
Some vendors (example: Xilinx™) provide a means for SPI bus arbitration, but this is not a requirement of the SPI standard.
SPI has a shared data bus.
The master transmits data on the SO signal line and receives data on the SI signal line, this allows the bus master to simultaneously transmit and receive data.
All data transfers must take place between the Bus Master and slaves.
Data transfers directly between two slave devices (Flyby transfers) are not allowed.
The SPI Bus Master signals are shown below.

SCK SPI Bus Clock output by SPI Bus Master

SI Input Data to Bus Master

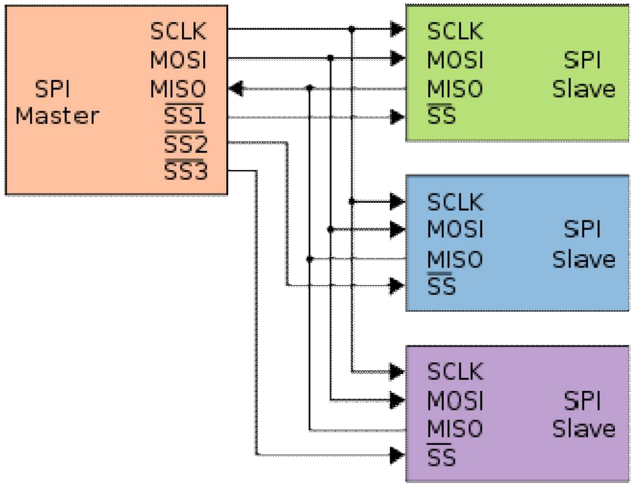
SO Output Data from Bus Master

SPI: Serial Peripheral Interface

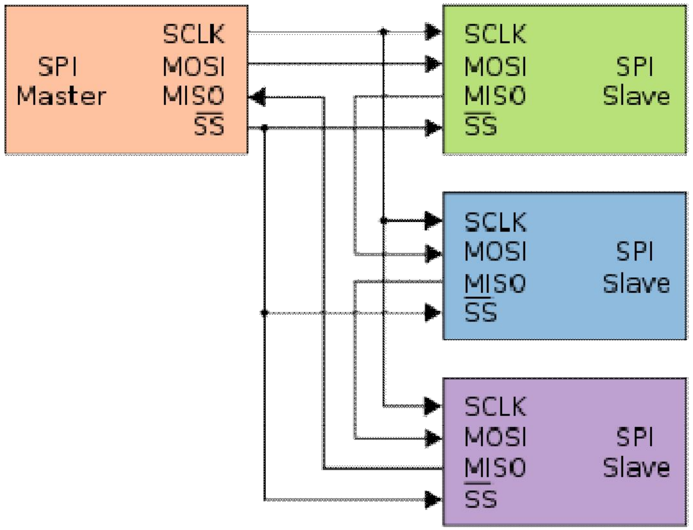


example of an SPI system with one bus master and three slave devices.
The SPI Bus Master outputs an individual device select signal for each SPI device slave.

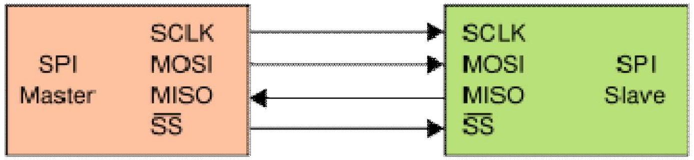
SPI: Serial Peripheral Interface Possibili configurazioni



Configurazione a tre slave con CS#

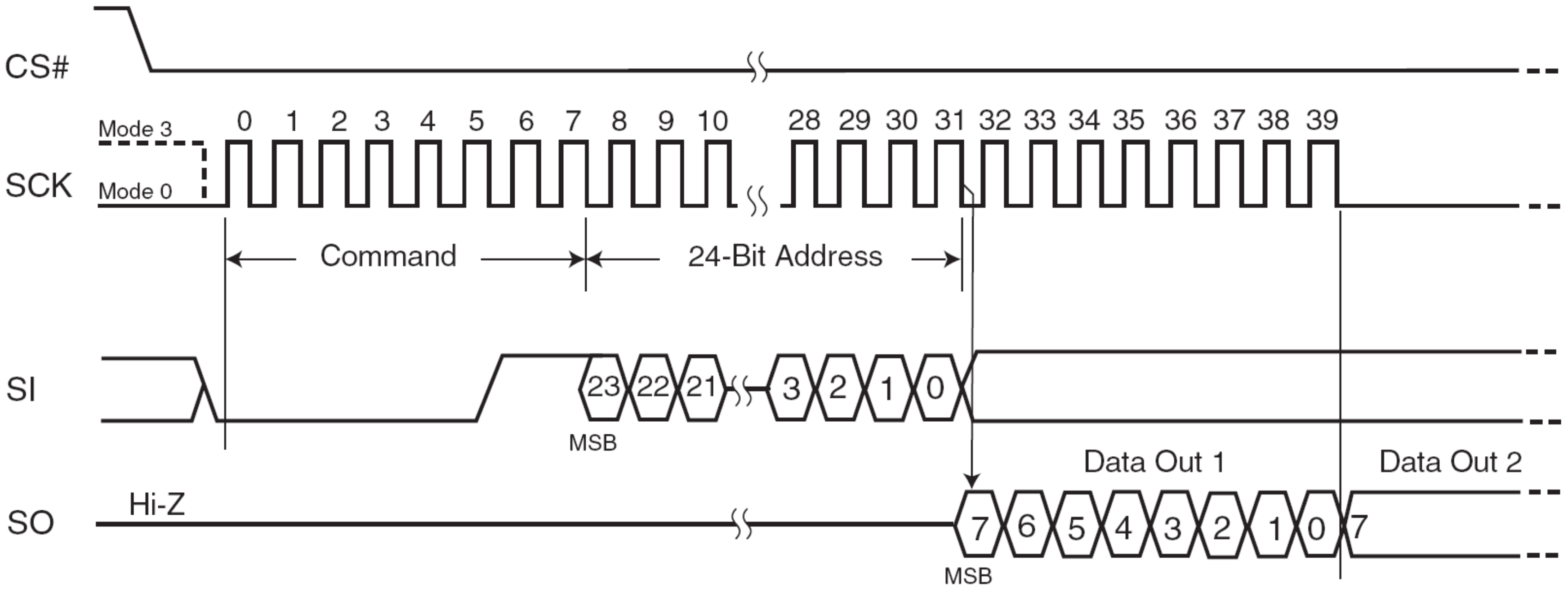


Configurazione con tre slave daisy chained



Configurazione a singolo slave

SPI: Serial Peripheral Interface



SPI Bus Cycle

example of an SPI read bus cycle of a *Spansion* SPI Flash.

For Spansion SPI devices, data is sampled on the rising edge of SCK and changes on the falling edge of SCK.

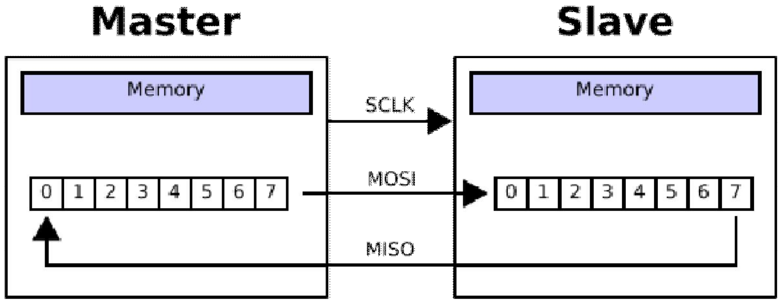
SPI: Serial Peripheral Interface

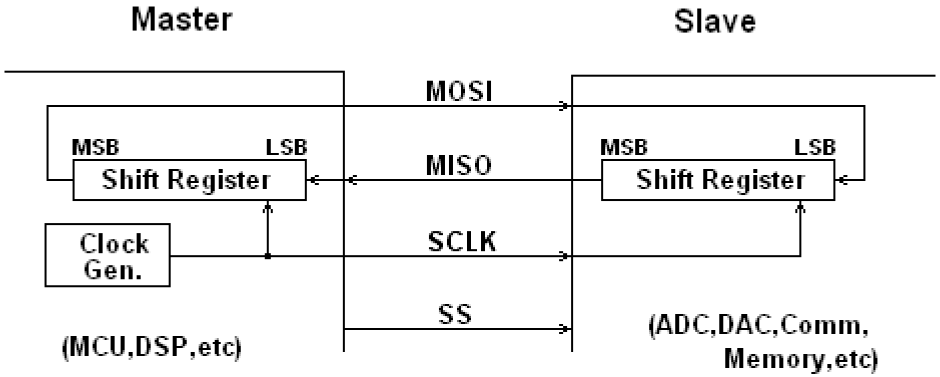
The bus cycle events are as follows:

1. CS# is asserted low to select the Flash.
2. The SPI read command is send.
3. The internal 24-bit Flash memory address is send.
4. The first Flash data byte is read out.

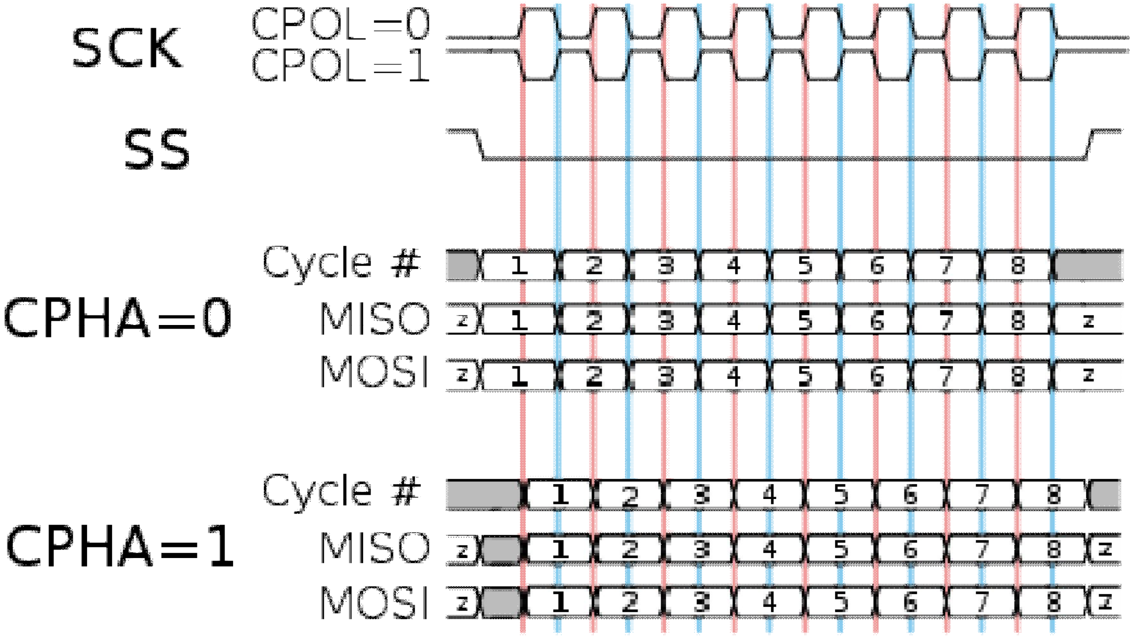
In Figure at the left of the SCK signal are shown the options of *Mode 0 and Mode 3. Depending on the* system SCK should be high (Mode3) when idle. In other systems SCK it should be low (Mode0) when idle.
In both mode 0 and 3 data is clocked in on the rising edge of clock and out on the falling edge of clock.
The SPI bus is a loose standard. Some devices can have a reversed SCK polarity. Note that some SPI devices can be connected in a daisy chain manner, but again this is not a requirement of the SPI standard.
It is possible to daisy chain SPI devices with the data output of the first SPI device going into the data input of the 2nd SPI and so on.
Note that, like with JTAG, none of the daisy chained devices would have independent access to the bus master.

SPI: Serial Peripheral Interface



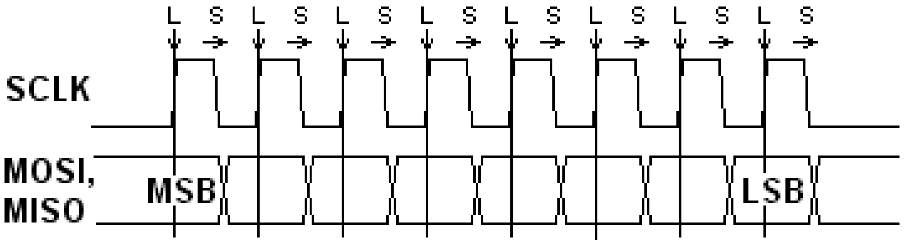


Tempistiche

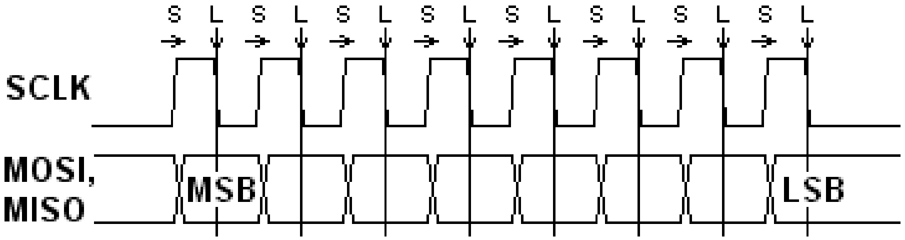


SPI: Serial Peripheral Interface

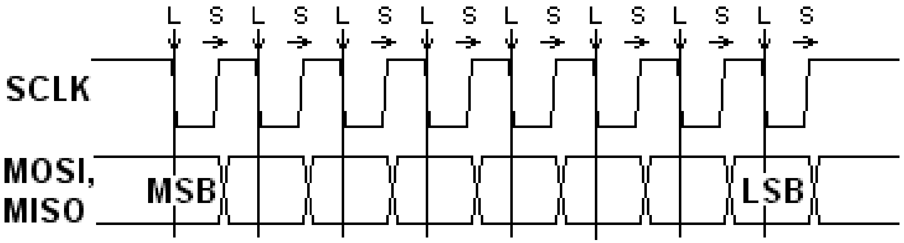
Modo 1



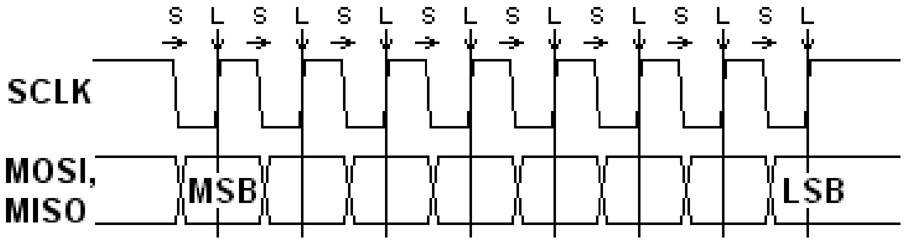
Modo 2



Modo 3



Modo 4



I2C: Inter Integrated Circuit

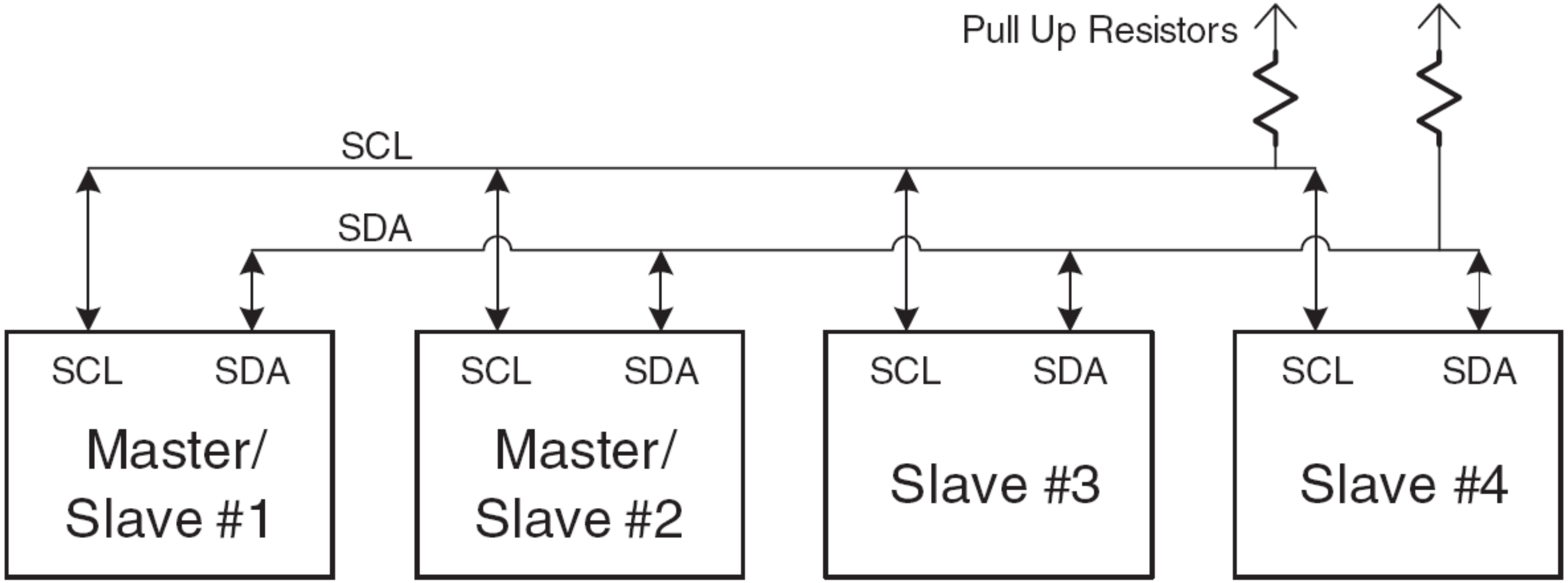
I2C is an acronym for *Inter IC bus. I2C allows for multiple bus masters and flyby data transfers.*
Standard I2C can support data transfer up to 100 kbs. There is also fast I2C at 400 kbs and **high speed I2C at 3.4 Mbps**.
The I2C Bus Master signals are shown below:

SCL -> SPI Bus Clock.

SDA -> Data

I2C bus has only two signals: SCL and SDA.
They are both bi-directional and open collector.
Pull-up resistors on SCL and SDA are required.
SCL is used for clock and wait. SDA is used for address and data.
Since there is only one data line full-duplex cannot be supported.

I2C: Inter Integrated Circuit

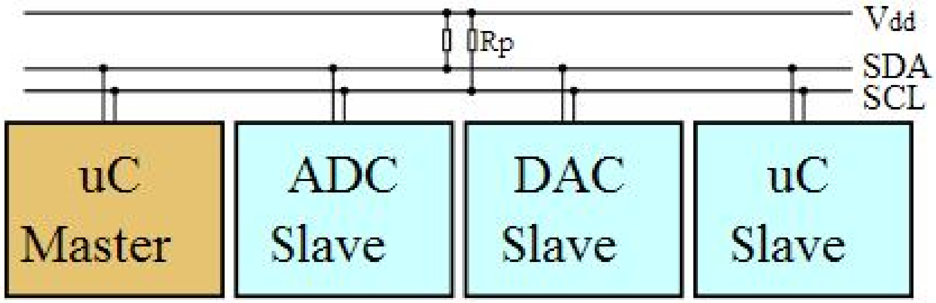


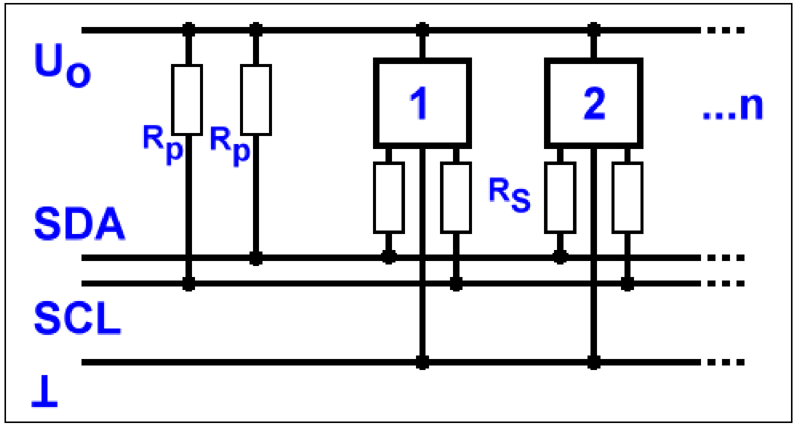
example I2C system is shown below with two Master/Slave devices and two Slave only devices.

The I2C bus does not have device select signals, but selects an I2C device by sending a device select byte (adressing).

Therefore, all I2C devices must be preprogrammed with a unique I2C bus address before they are used on the I2C bus. The I2C protocol supports up to 127 addressable devices.

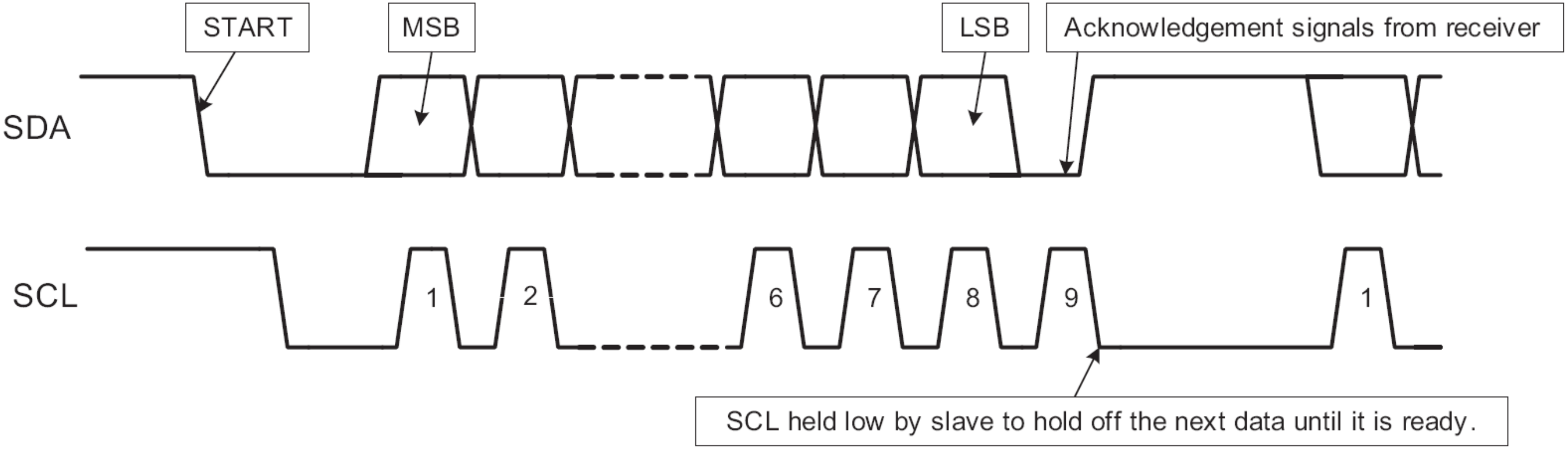
I2C: Inter Integrated Circuit





Resistenze di adattamento e protezione dei dispositivi

I2C: Inter Integrated Circuit

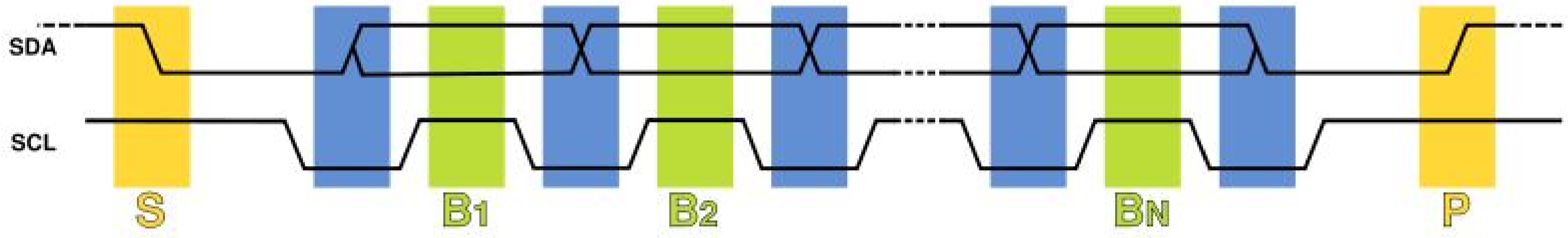

START
MSB
LSB
Acknowledgement signals from receiver
SDA
SCL
1
2
6
7
8
9
1
SCL held low by slave to hold off the next data until it is ready.

The I2C clock signal (SCK) is generated by the current master, but all the slaves can hold the SCK signal low until they are ready to allow it to go high.

In this way a rising SCK is delayed until the slowest I2C device is ready.

Figure shows the transfer of a byte to a slave, the slave acknowledges the transfer, and then holds down SCL until it is ready for the next byte.

I2C: Inter Integrated Circuit



S = Start Bit
Campionamenti sulle linee verdi, commutazione sulle linee blu
P = Stop Bit
Bi = Bit del Byte o del messaggio (più in generale)

I2C: Inter Integrated Circuit

Bus arbitration is done with the SDA line.

All of the potential bus masters simultaneously try to drive the SDA signal.

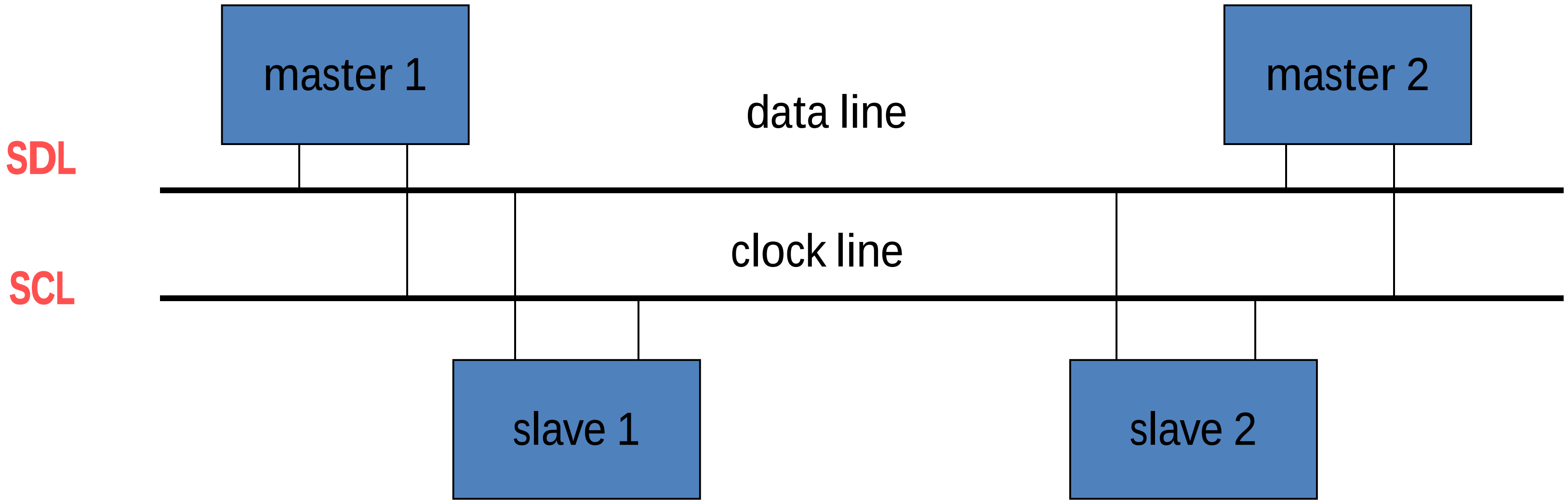
When any bus master detects that they have driven a '1', but detected a '0' on the SDA signal it has lost the bus arbitration. Perfettamente analogo al CAN!!!!!!

It will immediately give up driving the SDA signal and will go into slave mode in case it was the addressed slave device.

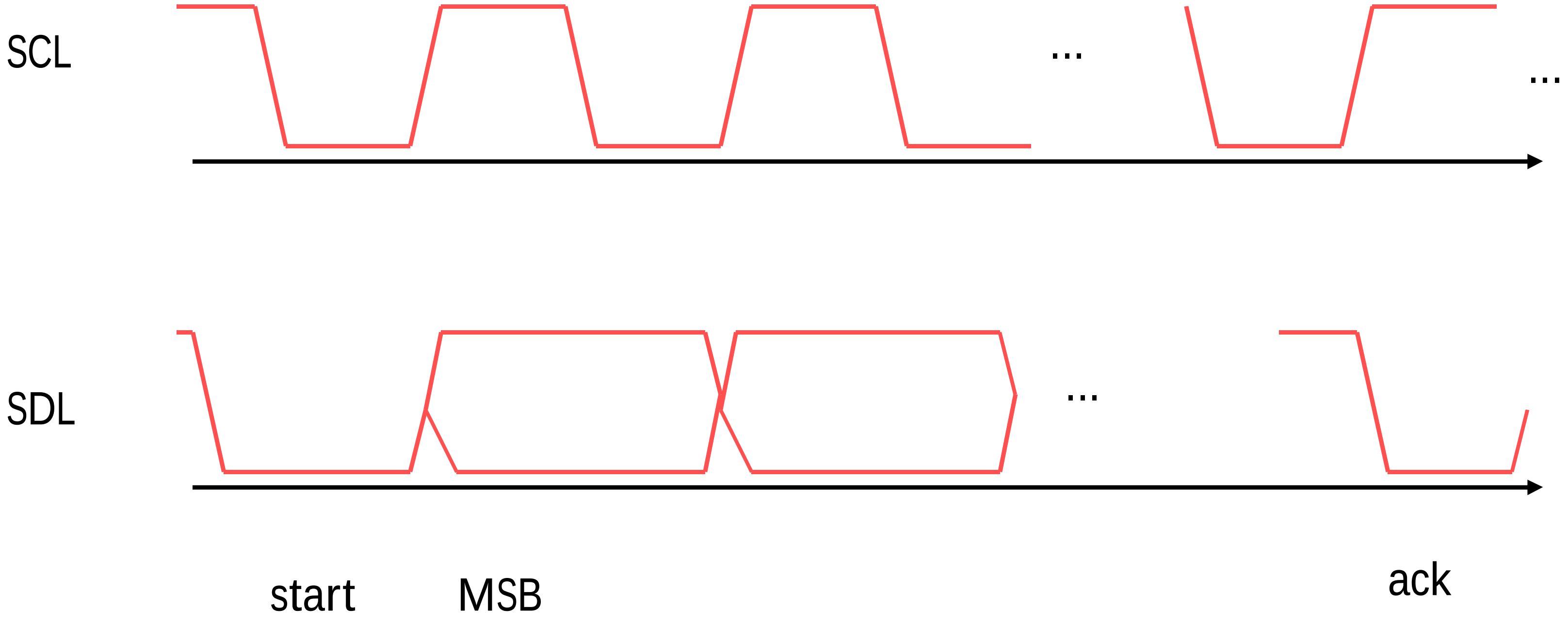
I^{2}C bus

- Designed for low-cost, medium data rate applications.
- Characteristics:
  - serial;
  - multiple-master;
  - fixed-priority arbitration.
- Several microcontrollers come with built-in I^{2}C controllers.

I^{2}C physical layer

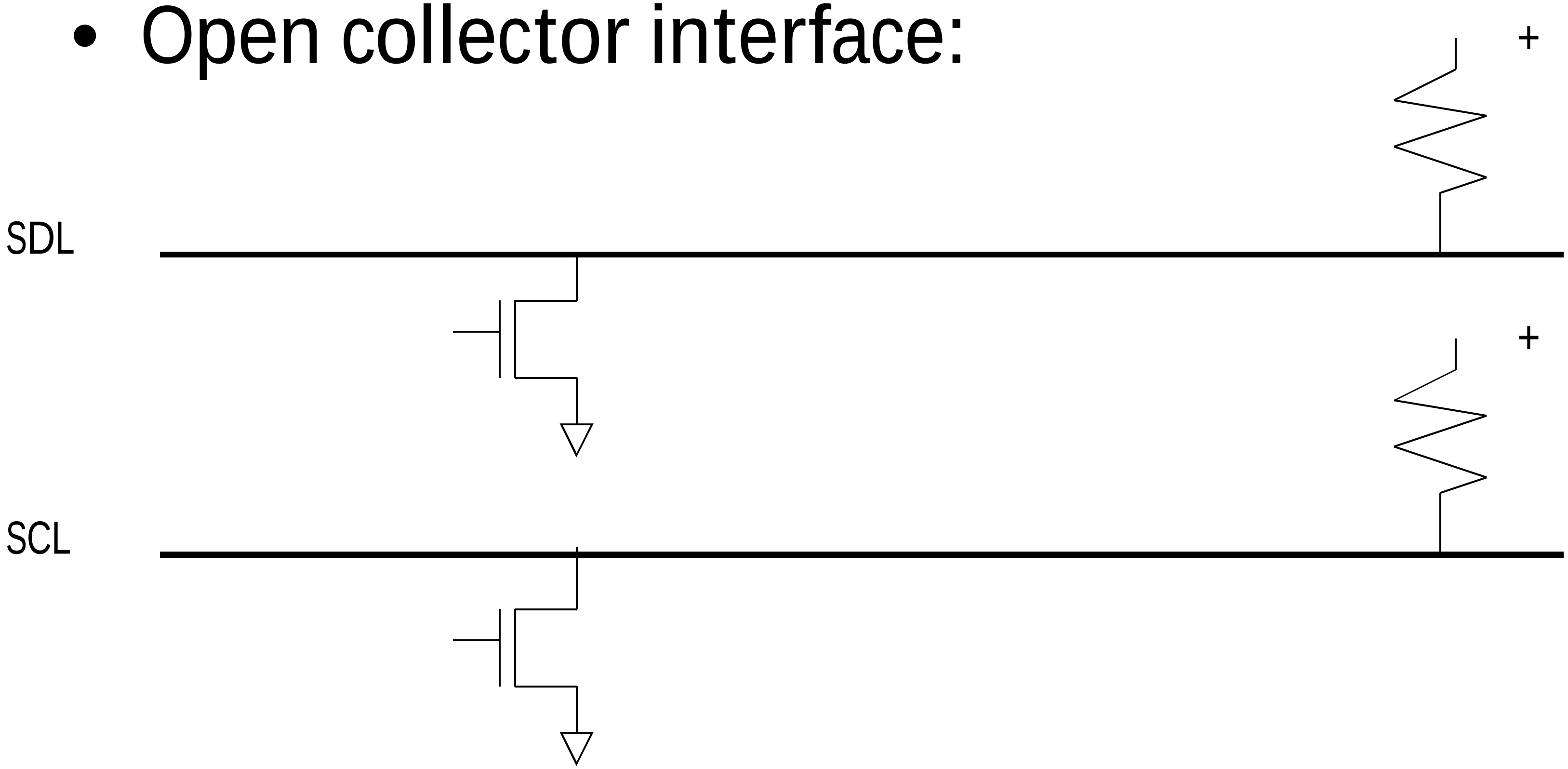


I^{2}C data format



I^{2}C electrical interface

- Open collector interface:



I^{2}C signaling

- Sender pulls down bus for 0.
- Sender listens to bus---if it tried to send a 1 and heard a 0, someone else is simultaneously transmitting.
- Transmissions occur in 8-bit bytes.

I^{2}C data link layer

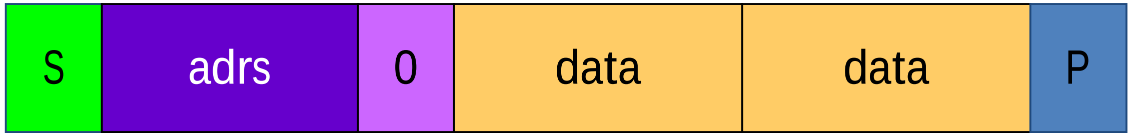
- Every device has an address (7 bits in standard, 10 bits in extension).
  - Bit 8 of address signals read (1)or write (0).
- General call address allows broadcast.

I^2C bus arbitration

- Sender listens while sending address.
- When sender hears a conflict, if its address is higher, it stops signaling.
- Low-priority senders relinquish control early enough in clock cycle to allow bit to be transmitted reliably.

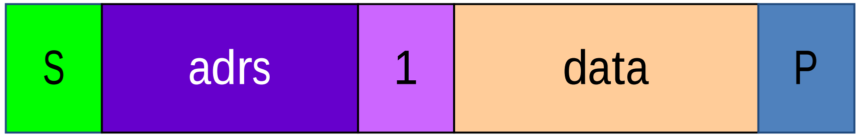
I^{2}C transmissions

multi-byte write



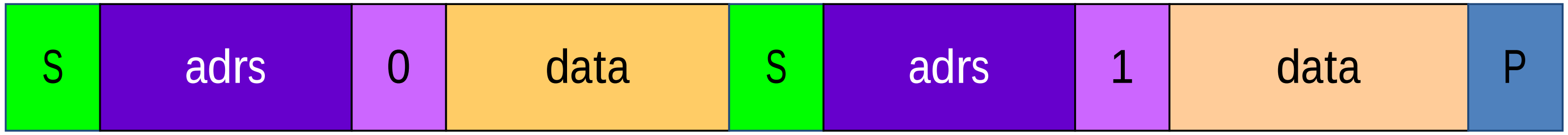
S	adrs	0	data	data	P

read from slave



S	adrs	1	data	P

write, then read



S	adrs	0	data	S	adrs	1	data	P

JTAG: Joint Test Action Group

The Joint Test Action Group (JTAG) method of connection is the **IEEE standard 1149**.
It is also known as “JTAG boundary scan”.

JTAG is commonly used for the following applications:

1. Board Assembly Test (verifies the connectivity of device pins to the PCB)
2. Development Tool (in-circuit emulator)
3. System Debug (provides a “back door” into the system)
4. Testing internal device circuitry (not be discussed here)

Typically, JTAG is a feature found in relatively high pin count devices, but not in low pin count devices.

I2C and SPI can be found in both high pin count devices like microcontrollers and in low pin count devices like A/D converters.

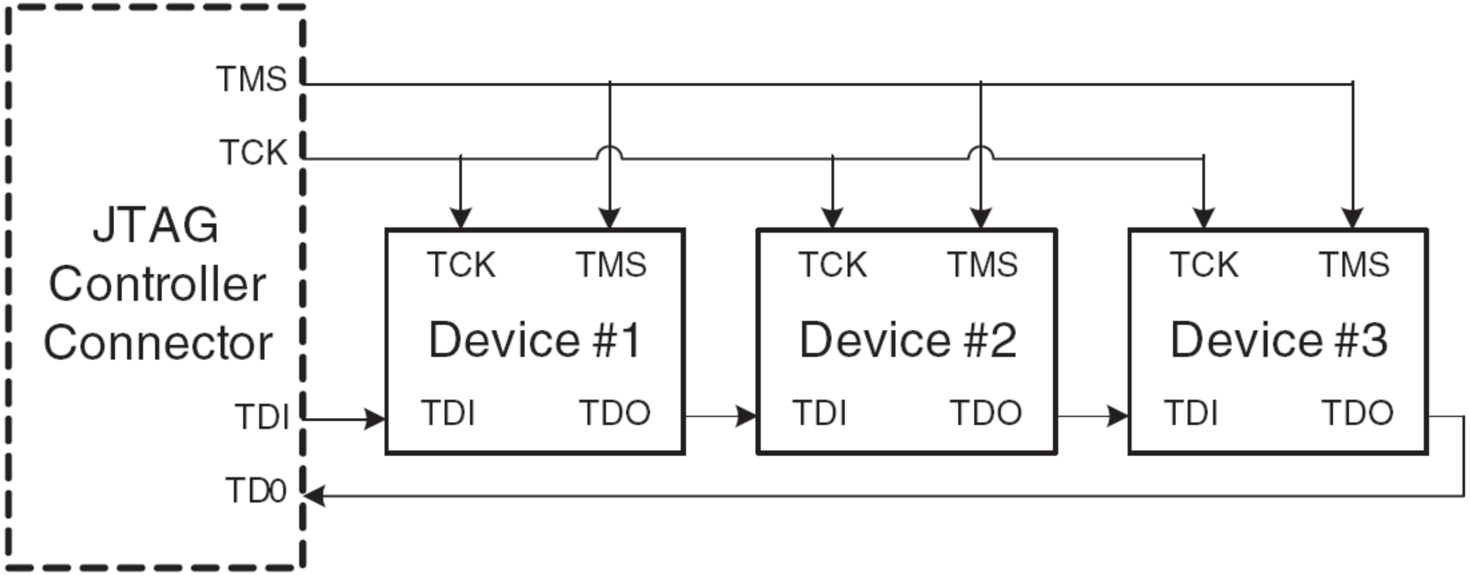
JTAG: Joint Test Action Group

JTAG uses four (plus an optional reset signal) wires to pass data through devices in a daisy chain.

These signals are shown below:

1. TDI (Test Data In) - Daisy Chained
2. TDO (Test Data Out) - Daisy Chained
3. TCK (Test Clock) - Shared
4. TMS (Test Mode Select) - Shared
5. TRST (Test ReSeT) optional

JTAG: Joint Test Action Group



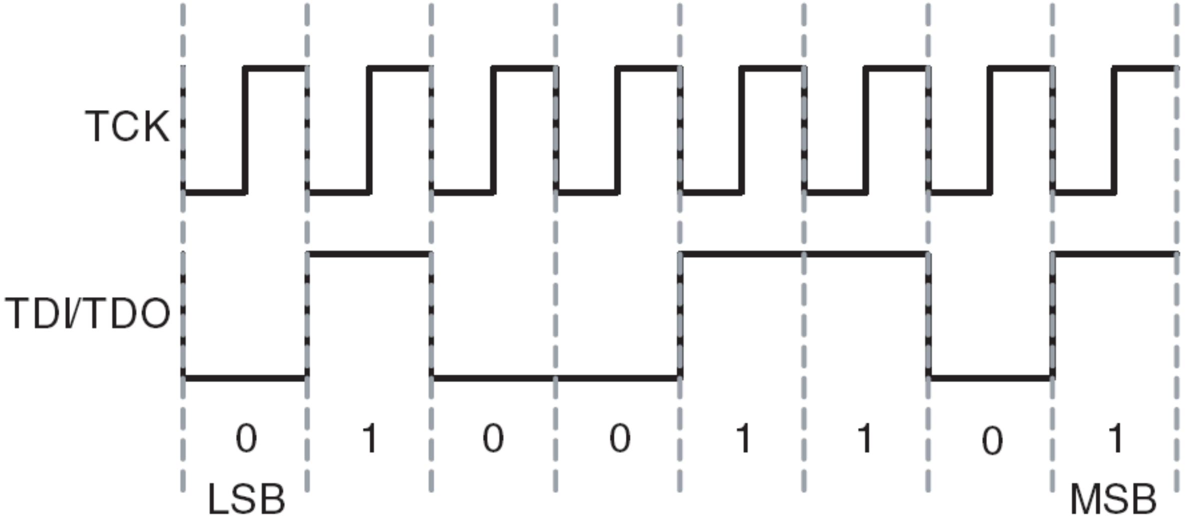
Example of a JTAG circuit.
There is a JTAG Controller Connection and three JTAG devices.

A JTAG Controller is connected at the connector and it drives TCK and TDI into DEVICE #1.

The TDI data eventually passed out of the DEVICE #1 TDO pin into DEVICE #2 and so on until the daisy chain is closed back at the JTAG Controller at TDO.

The buses signals TMS and TCK control the data transfer.

JTAG: Joint Test Action Group



Data outputs change on the falling edge of TCK and data is sampled on rising edge of TCK.

TMS and TRST are not shown.

Data transfers with JTAG take more clocks than the other serial buses, because in JTAG the data must pass sequentially through the devices in the chain.