COPYRIGHT

DISCLAIMER

This content is provided without warranty or representation of any kind. The use of the content is entirely at your own risk and Stellenbosch University (SU) will have no liability directly or indirectly as a result of this content.

The content must not be assumed to provide complete coverage of the particular study material. Content may be removed or changed without notice.

The video is of a recording with very limited post-recording editing. The video is intended for use only by SU students enrolled in the particular module.

UNIVERSITEIT
iYUNIVESITHI
STELLENBOSCH
UNIVERSITY

100
1918 · 2018

forward together · saam vorentoe · masiye phambili

Computer Systems / Rekenaarstelsels 245 - 2020

Lecture 20

Serial Communication - SPI/
Seriële Kommunikasie - SPI

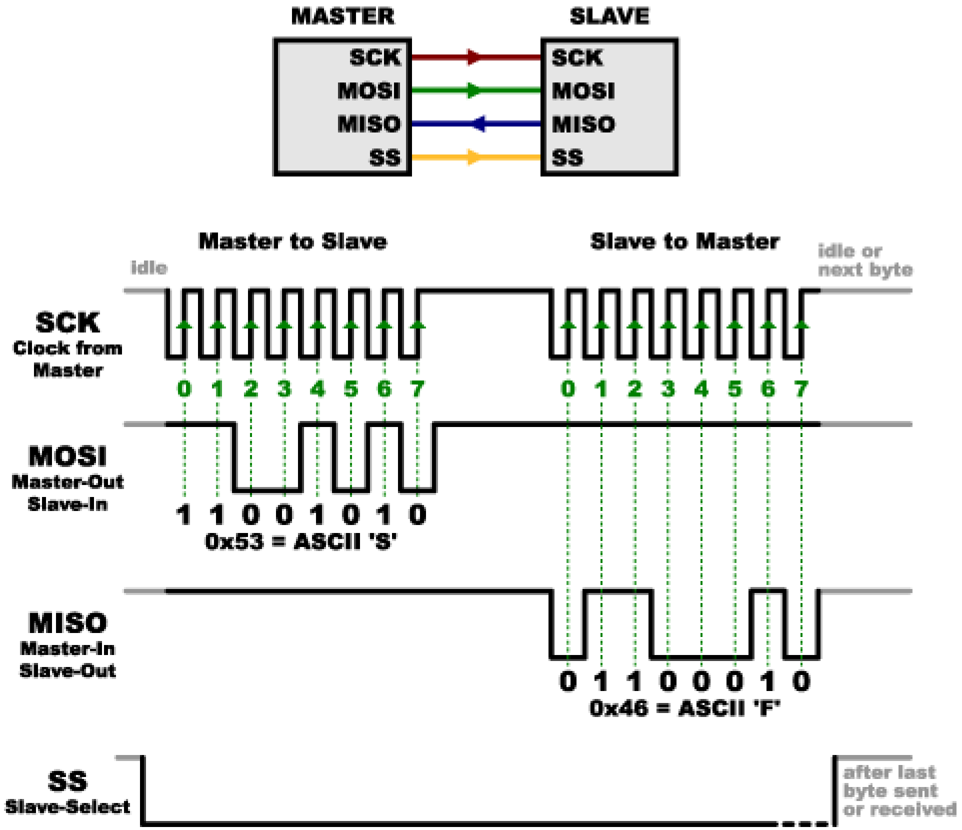
Dr Rensu Theart & Dr Lourens Visagie

Standard Serial Communication
Standaard Seriële Kommunikasie

	Half/Full-duplex	Bus/point-to-point	Synchronous/ Asynchronous
UART (Universal Asynchronous Receiver/Transmitter	Full	Point-to-point	Asynchronous
I^{2}C (Inter-Integrated Circuit)	Half	Bus	Synchronous
SPI (Serial Peripheral Interface)	**Full**	**Bus**	**Synchronous**

SPI

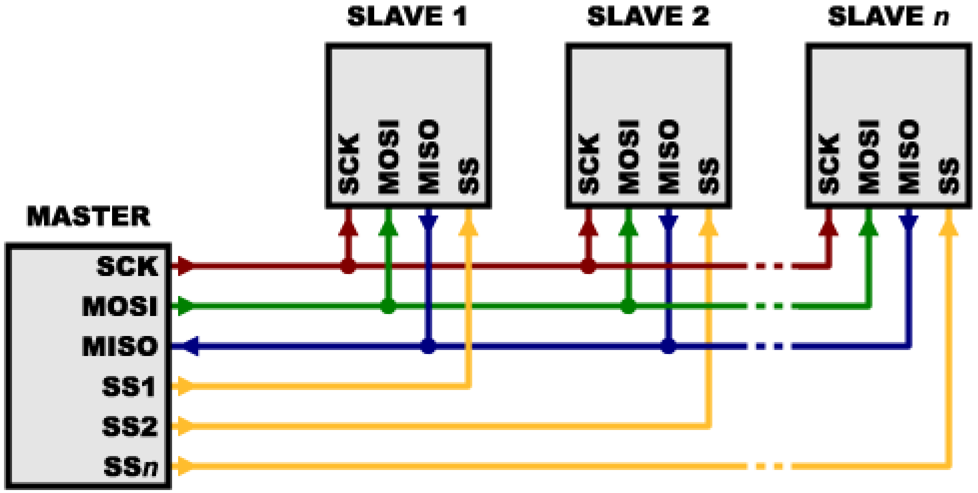
- SPI = Serial Peripheral Interface
- Only one master
- At least 4 signals
- 3 Lines for clock and data
  - SCK: Clock signal (output by master)
  - MOSI: Master Out / Slave In
  - MISO: Master In / Slave Out
- Master controls clock signal
- Master places data on the MOSI line, if it wants to write data to the slave
- If the slave has to reply with data, the master will keep on sending clock signals and the slave will place data on the MISO line
- SPI supports full-duplex communication - data can be sent and received at the same time: Slave outputs data on MISO at the same time master is outputting on MOSI

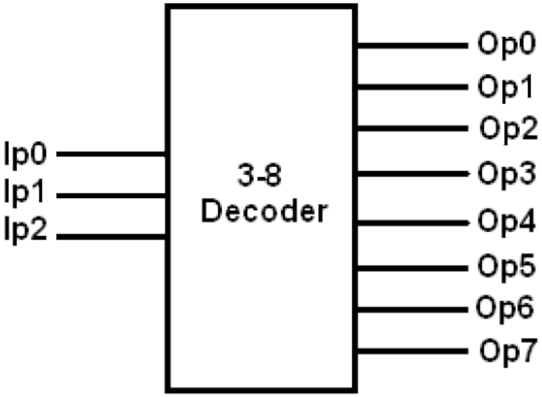


https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi

SPI

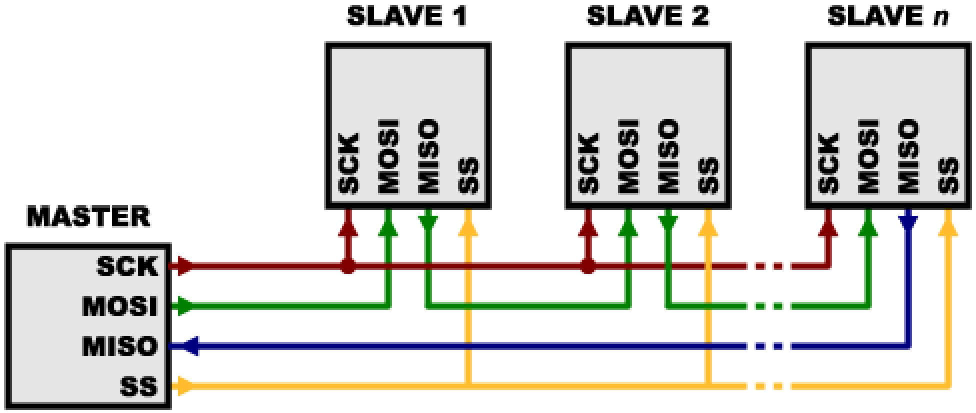
- Each slave has a Slave Select (SS) line (or CS for "Chip Select").
  - Master uses the SS line to select which slave to talk to: SS acts like an enable signal for the slave
  - SS signals are active low (0V = enabled)
- Bus contention can still happen if more than one slave is enabled at the same time (multiple slaves driving the MISO signal at once)
- Total number of signals = 3 + n (n = number of slaves)
- For many slave devices, use a decoder to reduce the number of required IO pins
  - This will also ensure that only a single slave is on at one time

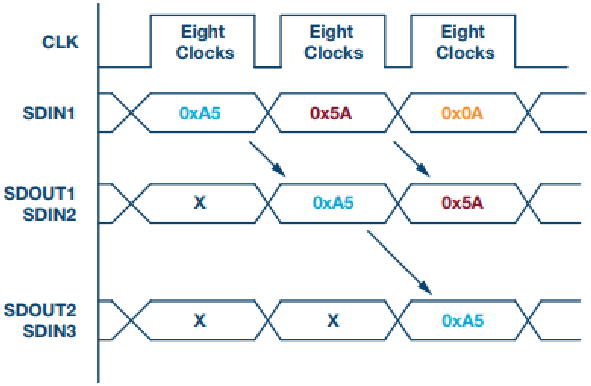


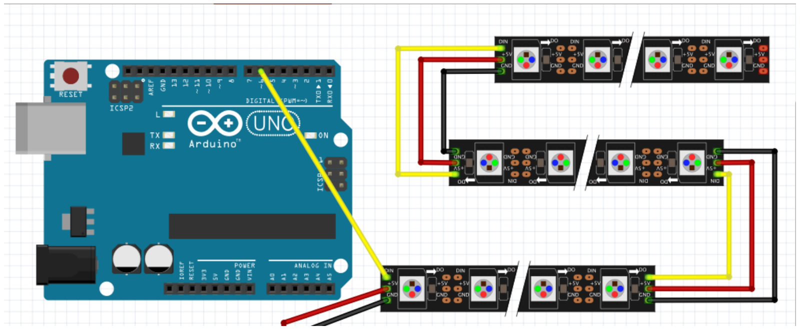


SPI

- Another topology makes use of "daisy-chaining": data output (DOUT) of one slave device is connected to data input (DIN) of next slave
- Only one slave-select (SS) signal connected to all slaves
- All slaves get the same clock signal
- Data is propagated through the slave devices. First byte of information that is transmitted ends up at the last slave device in the chain
- For 3x slave devices, the master has to generate 24 clock cycles to transfer a byte to the last slave
- Slave devices have to support this
- Often used with output-only scenarios – data is not returned to the master
    - Example: LED strips with addressable drivers

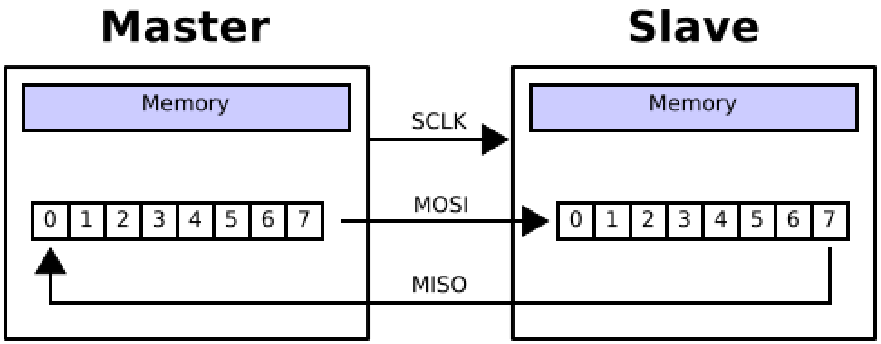






SPI

- Straightforward hardware implementation: SPI is implemented using simple shift registers
- There is no frame "overhead" – no start or stop bits. Data is sent as a continuous bit stream
- Data is always transmitted in multiples of 8 bits (1 byte)
- Shift register can be 8-bit, 16-bit or 32-bit, but number of bytes to transmit or receive is dependent on the application
- Data can be transmitted LSB first or MSB (bit order). This is configured in the device setup. Consult slave datasheet to decide which to use



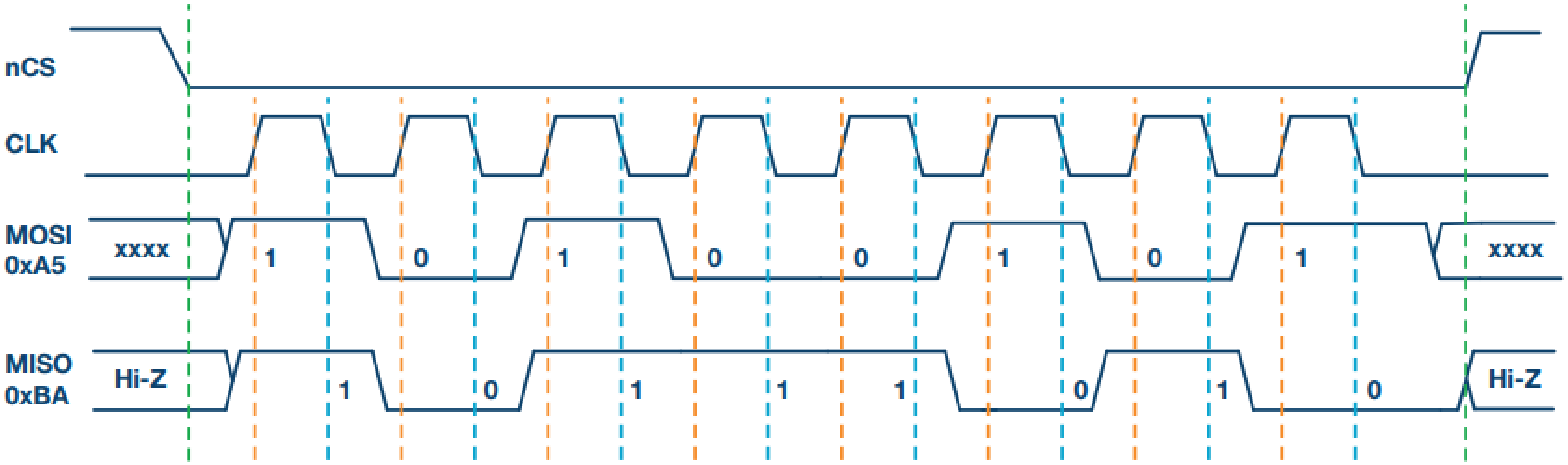
- Voltage levels depends on device (typically TTL levels)
- Speeds of up to 100 MHz possible
- Master controls clock – decides on transfer speed
- Slave will usually have a maximum allowed clock rate (and not a specific expected value)
- Typical clock rate is 16MHz
- At high transfer rates, transmission line effects should be considered

SPI

- Different options for clock phase (CPHA) and polarity (CPOL): determines when data is sampled and shifted out
- Again determined by slave requirements – check the datasheet, and setup SPI peripheral accordingly

Table 1. SPI Modes with CPOL and CPHA

SPI Mode	CPOL	CPHA	Clock Polarity in Idle State	Clock Phase Used to Sample and/or Shift the Data
0	0	0	Logic low	Data sampled on rising edge and shifted out on the falling edge
1	0	1	Logic low	Data sampled on the falling edge and shifted out on the rising edge
2	1	1	Logic high	Data sampled on the falling edge and shifted out on the rising edge
3	1	0	Logic high	Data sampled on the rising edge and shifted out on the falling edge

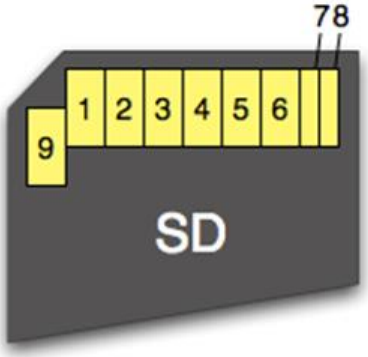


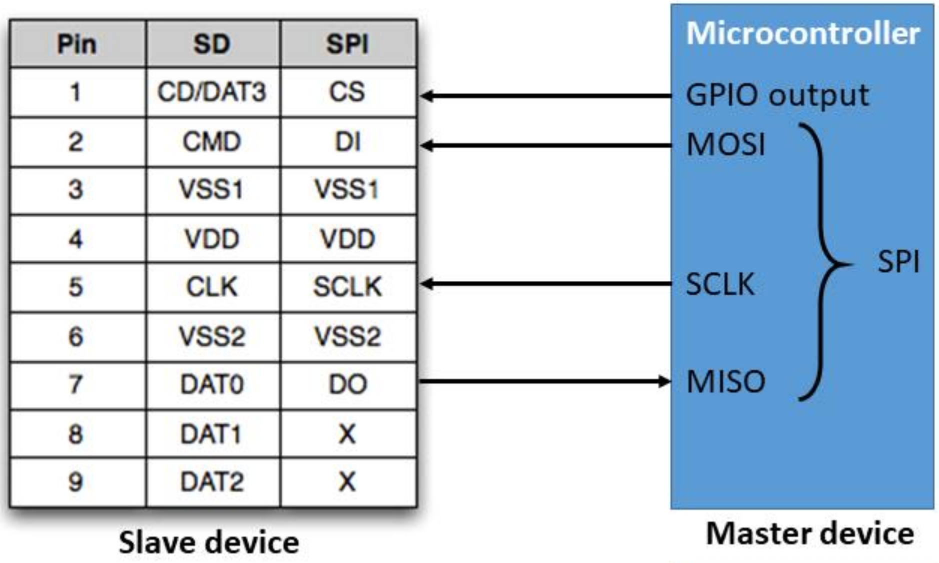
SPI Mode 0, CPOL = 0, CPHA = 0: CLK idle state = low, data sampled on rising edge and shifted on falling edge.

SPI Interfacing examples
SPI Koppelvlak voorbeelde

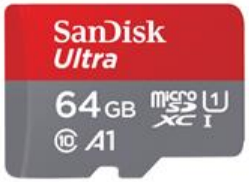
- SD Card
- SD cards can interface through native SD interface: 4 parallel bi-directional data signals
- But all SD cards also support SPI interface
- SPI signals share same pins as native SD interface – commands used to setup the card in one of the two modes
- Fairly complicated command set, and read and write sequences to use SD card through SPI

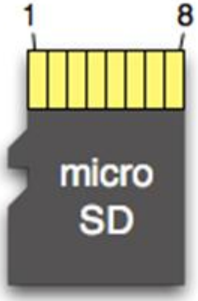
SPI Interfacing examples
SPI Koppelvlak voorbeelde

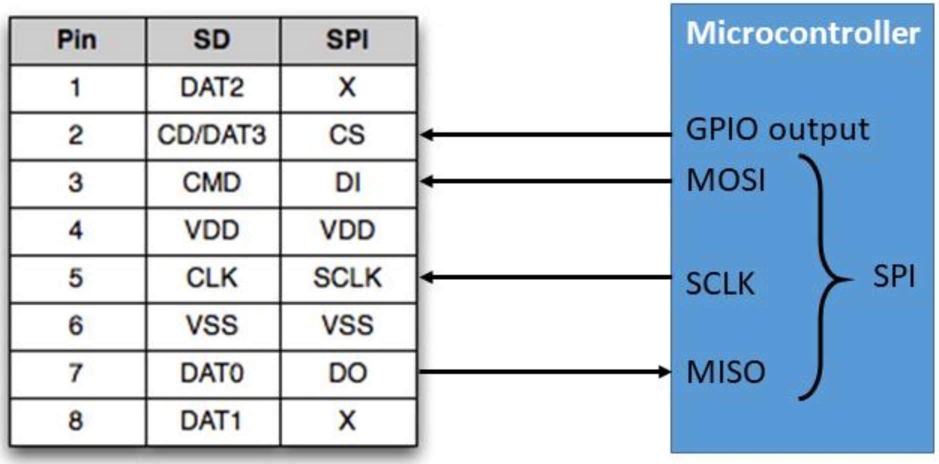




Pin	SD	SPI
1	CD/DAT3	CS
2	CMD	DI
3	VSS1	VSS1
4	VDD	VDD
5	CLK	SCLK
6	VSS2	VSS2
7	DAT0	DO
8	DAT1	X
9	DAT2	X

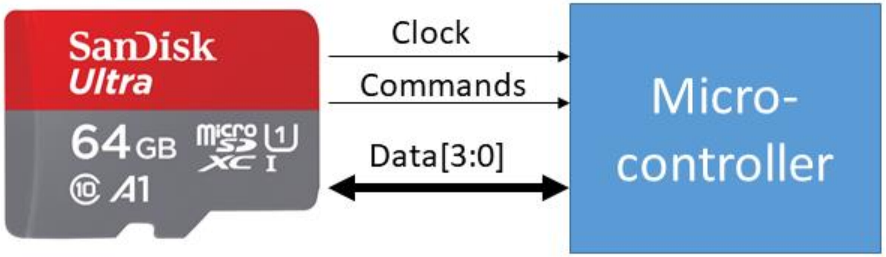
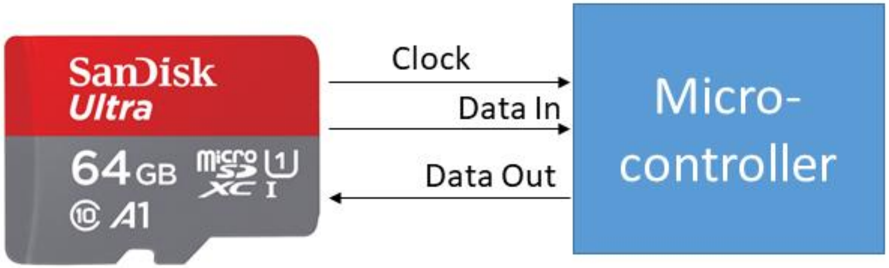






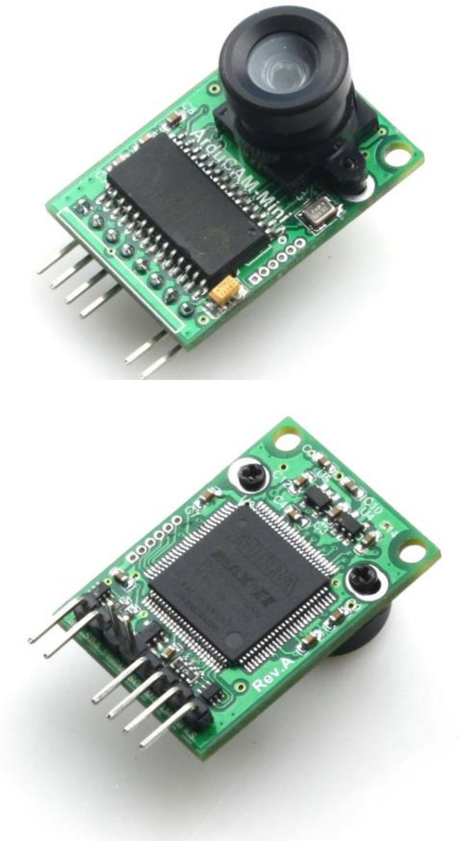
Pin	SD	SPI
1	DAT2	X
2	CD/DAT3	CS
3	CMD	DI
4	VDD	VDD
5	CLK	SCLK
6	VSS	VSS
7	DAT0	DO
8	DAT1	X

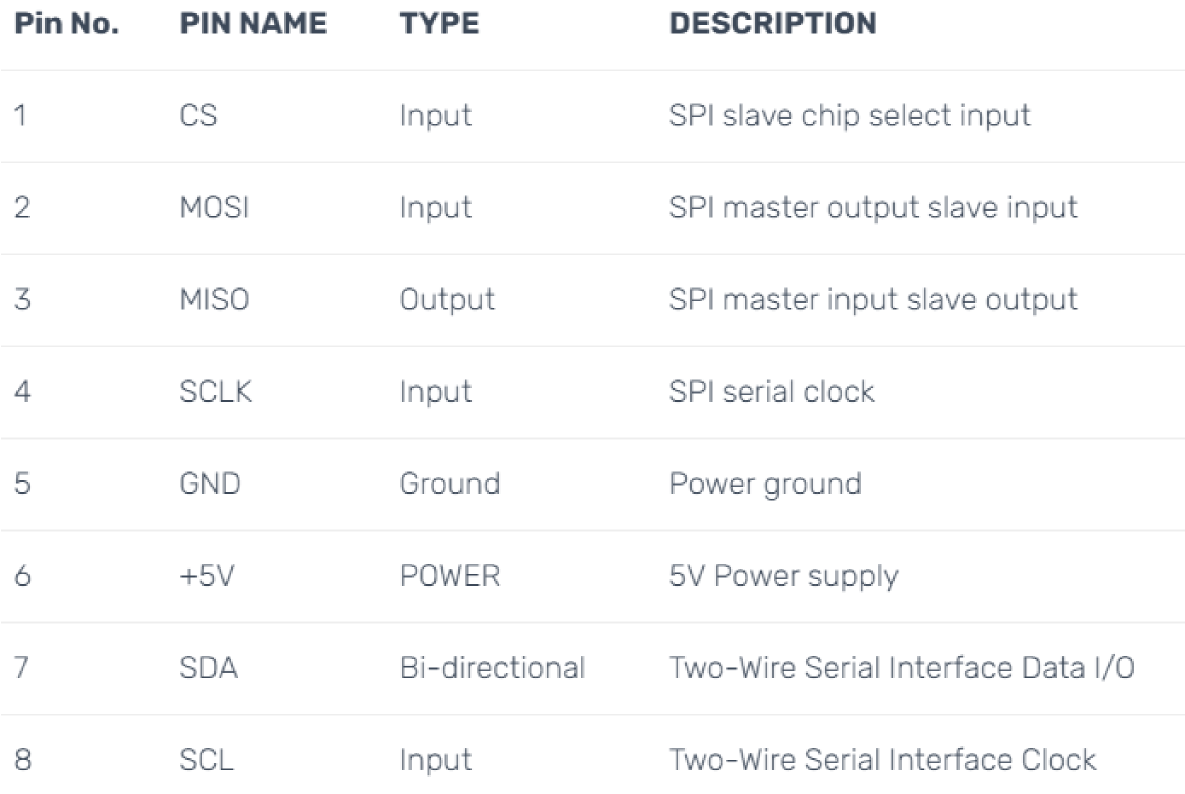
SPI Interfacing examples
SPI Koppelvlak voorbeelde

		✓	✗
SD mode	 SanDisk Ultra 64GB Clock Commands Data[3:0] Micro-controller	Faster access	No dedicated peripheral on microcontroller
SPI	 SanDisk Ultra 64GB Clock Data In Data Out Micro-controller	Dedicated peripheral on microcontroller – easier to integrate	Slower access

SPI Interfacing examples
SPI Koppelvlak voorbeelde

- Camera modules
- I2C for imager sensor configuration changes
- SPI for high speed image data transfer





Pin No.	PIN NAME	TYPE	DESCRIPTION
1	CS	Input	SPI slave chip select input
2	MOSI	Input	SPI master output slave input
3	MISO	Output	SPI master input slave output
4	SCLK	Input	SPI serial clock
5	GND	Ground	Power ground
6	+5V	POWER	5V Power supply
7	SDA	Bi-directional	Two-Wire Serial Interface Data I/O
8	SCL	Input	Two-Wire Serial Interface Clock

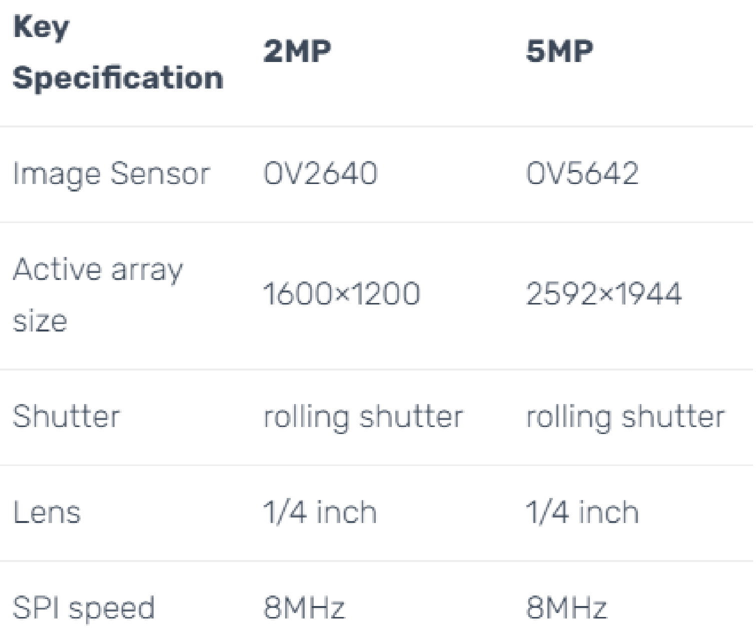
ArduCam Mini

SPI Interfacing examples
SPI Koppelvlak voorbeelde

- Camera modules
- Question: How long will it take to download to download a full 5MP image frame?

- 5MP ~ 2592x1944 resolution
- Assume RGB image, so 1 byte for each of red, green and blue component, per pixel.
- Entire frame = 3 bytes * 2592 * 1944 = 15 116 544 bytes
- Number of bits (SPI clocks) needed: ~120 000 000
- At 8Mhz clock rate, it will take 120/8 = 15s!

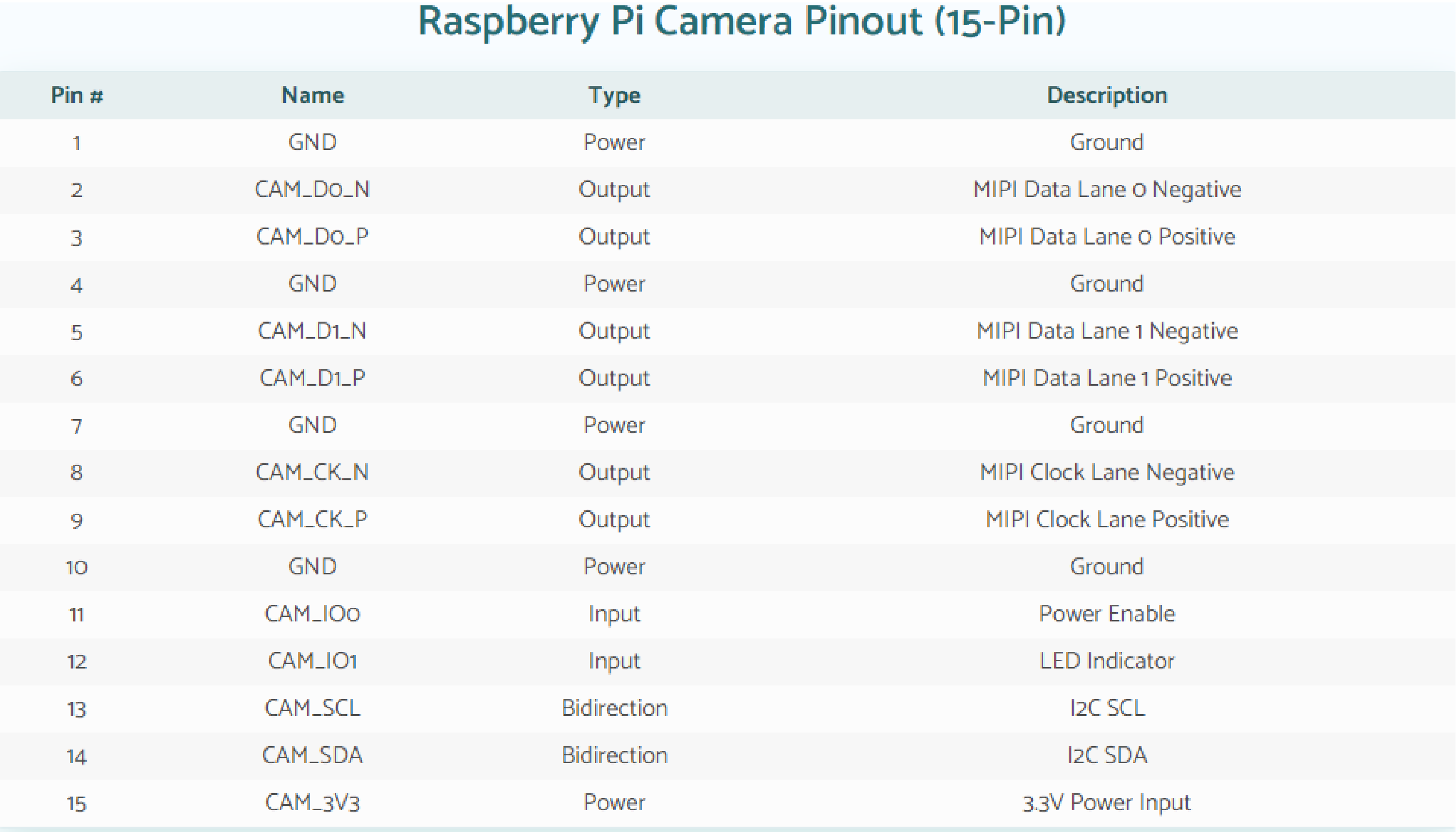
- How do you stream video out of a camera? (30 frames per second?)
  - Compression on image sensor
  - Use multiple parallel data signals
  - Decrease frame resolution



Key Specification	2MP	5MP
Image Sensor	OV2640	OV5642
Active array size	1600×1200	2592×1944
Shutter	rolling shutter	rolling shutter
Lens	1/4 inch	1/4 inch
SPI speed	8MHz	8MHz

SPI Interfacing examples
SPI Koppelvlak voorbeelde

- Camera modules
- Raspberry PI camera module
- Uses Camera Serial Interface (CSI) (Mobile Industry Processing Interface – MIPI - standard)
- Not quite SPI, but similar
- Note the I2C interface for configuration, and differential signalling on camera data



Raspberry Pi Camera Pinout (15-Pin)

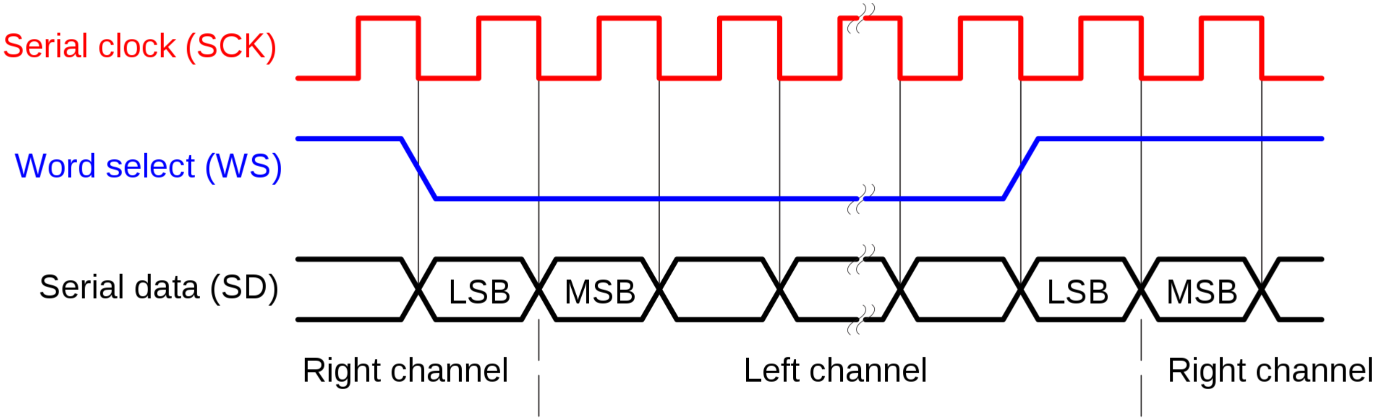
Pin #	Name	Type	Description
1	GND	Power	Ground
2	CAM_D0_N	Output	MIPI Data Lane 0 Negative
3	CAM_D0_P	Output	MIPI Data Lane 0 Positive
4	GND	Power	Ground
5	CAM_D1_N	Output	MIPI Data Lane 1 Negative
6	CAM_D1_P	Output	MIPI Data Lane 1 Positive
7	GND	Power	Ground
8	CAM_CK_N	Output	MIPI Clock Lane Negative
9	CAM_CK_P	Output	MIPI Clock Lane Positive
10	GND	Power	Ground
11	CAM_IO0	Input	Power Enable
12	CAM_IO1	Input	LED Indicator
13	CAM_SCL	Bidirection	I2C SCL
14	CAM_SDA	Bidirection	I2C SDA
15	CAM_3V3	Power	3.3V Power Input

SPI Interfacing examples
SPI Koppelvlak voorbeelde

- I2S: Integrated Inter-IC Sound Bus
- Specification from Philips Semiconductor (now NXP)
- Serial bus interface for connecting digital audio devices
- Commonly used by audio ADC (analog-to-digital) and DAC (digital-to-analog) converters
- NOT the same as I2C! (although the acronyms look similar)
- I2S is similar to SPI. In fact, the same microcontroller hardware can be used for I2S
  - CK (mapped to SPI SCK): Serial clock
  - SD (mapped to MOSI): Serial Data
  - WS (mapped to hardware controlled Slave Select NSS): Word Select – selects between left and right channel data

SPI Interfacing examples
SPI Koppelvlak voorbeelde

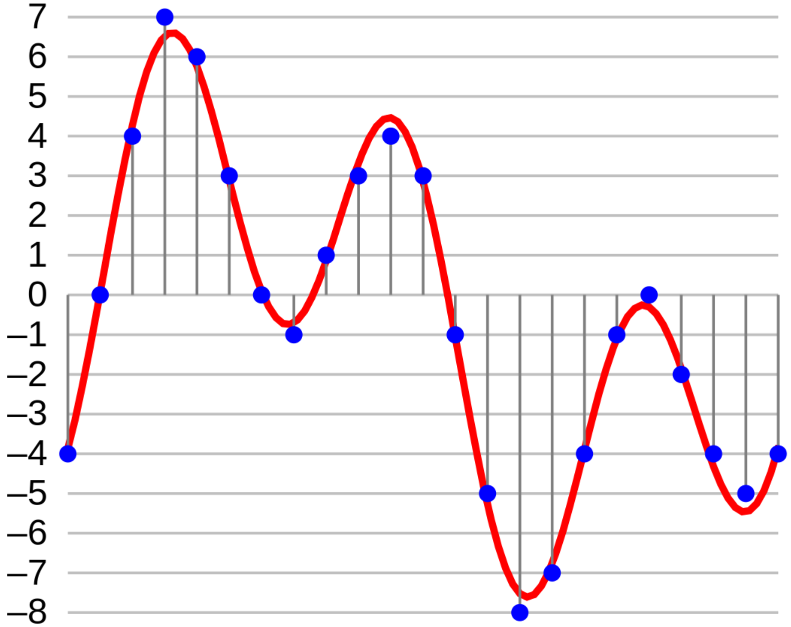
- I2S: Integrated Inter-IC Sound Bus
- Data sent MSB first
- Bit length per sample not specified
- WS (Word select) is like a clock signal, n times slower than SCK (where n is the number of bits per sample)



By wdwd - Own work, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=16579640

SPI Interfacing examples
SPI Koppelvlak voorbeelde

- I2S: Integrated Inter-IC Sound Bus
- Data is represented using Pulse Code Modulation (PCM)
  - Analog signal is sampled at uniform intervals, and quantized to nearest integer (number of bits used to represent integer determines resolution/quantization error)
  - Audio PCM values are signed integer – positive and negative range



CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=635225

SPI Programming
SPI Programmering

- Memory-mapped registers
  - Configuration Registers – sets up the SPI peripheral in various modes
    - Master or Slave
    - Full or half-duplex
    - Receive-only or transmit-only
    - Baud rate (clock rate)
    - Interrupt enable
  - Status Register (SR)
    - Read only bits, indicating data arrived (RXNE), data done transmitting (TXE)
  - Data Register DR
    - Write to DR: transmit data on MOSI line
    - Read from DR: return last data received from MISO line
  - I2S configuration register – when using the SPI peripheral in I2S mode

SPI Programming
SPI Programmering

- Interrupts
  - RXNE interrupt: when data arrived in the receive data register. Need to read the DR to get the received data
  - TXE interrupt: When data has been transferred from the DR into the output shift register (OK to write more data into the DR)

- Software or hardware control of the SS line
  - For software management, your program has to configure a pin as GPIO output and manually set signal level (low = enabled)
  - But the SPI peripheral can automatically control the SS signal for you (hardware control)

- Use DMA (Direct Memory Access) to free up processor (no need for interrupts during transmission)
  - More on this in future lecture

Further reading
Verdere lees

- https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi

Serial Communication – Application comparison
Seriele Kommunikasie – Toepassings vergelyking

	Advantage/Disadvantage	Typically used to interface with
UART	+ VERY common (and cheap) - Low data speeds (500,000 bps maximum) - Only suited for point-to-point communications - Start and –stop bit overhead	• Debugging • Modems (3G, Lorawan, NBIoT) • GPS receivers
I2C	+ Only 2 signals (clock and data) Bus speeds of 100 kHz to 3.4 MHz - Complex hardware implementation	• Analog-to-digital converters (ADCs) • Sensors (magnetic field sensor, temperature, acceleration, angular rate)
SPI	+ High speed (> 10MHz) + Simple hardware implementation - Lots of wires/tracks (at least 4)	• SD card • Camera modules • I2S Audio