UNIVERSITEIT
iYUNIVESITHI
STELLENBOSCH
UNIVERSITY

100

1918 · 2018

forward together · saam vorentoe · masiye phambili

Computer Systems / Rekenaarstelsels 245 - 2020

Lecture 17

Serial Communication – I²C / Seriële Kommunikasie – I²C

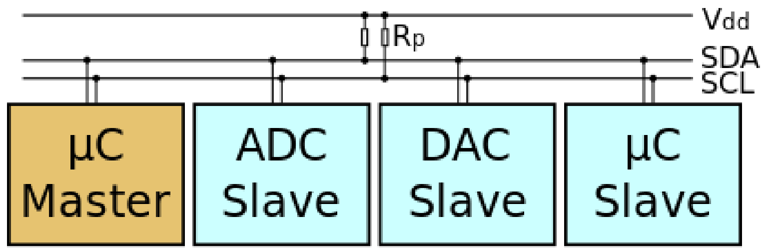
Dr Rensu Theart & Dr Lourens Visagie

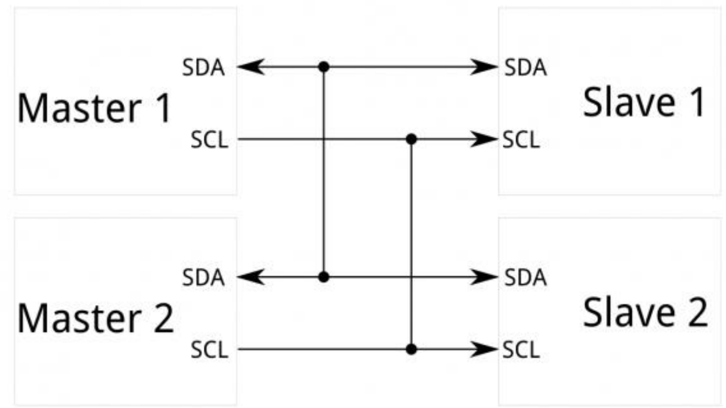
Standard Serial Communication
Standaard Seriële Kommunikasie

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

I2C signals
I2C seine

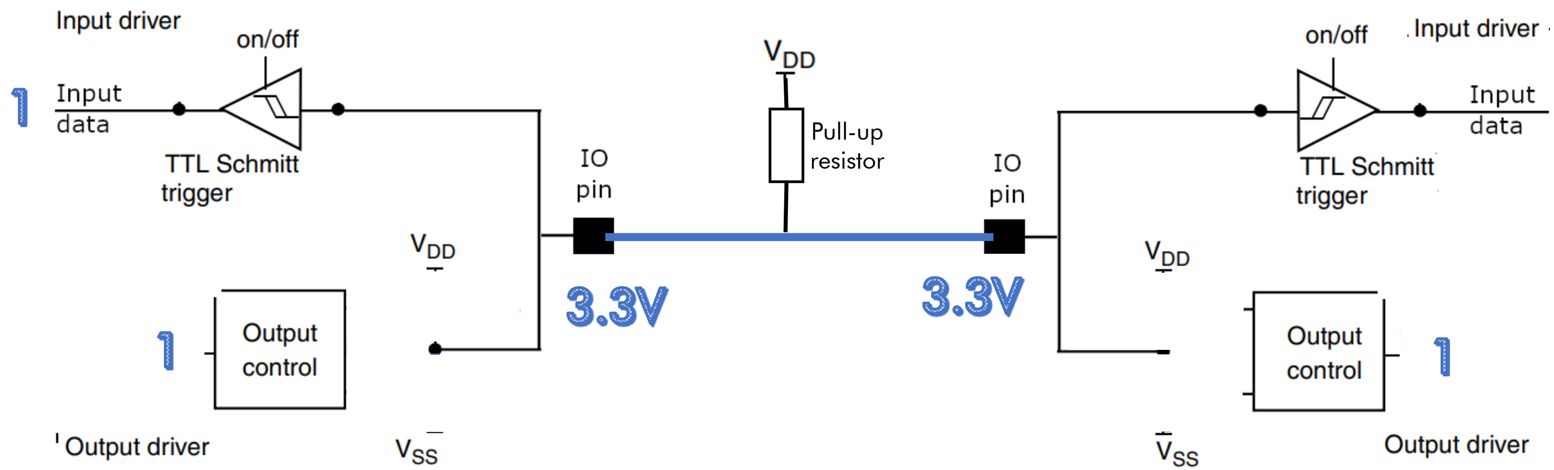
- I2C = Inter-Integrated Circuit
- Two communication lines, connected across multiple devices
  - Serial Clock (SCL)
  - Serial Data (SDA)
  - (Ground connection across all devices as well)
- Supports multiple masters, but normally a single master
- Master is the only one that controls clock signal (SCL)
- Any device can change the voltage level of bus signals – need to avoid bus contention





I2C Bus

- Microcontroller GPIO makes use of so-called **Open-drain** configuration for SCL and SDA bus signals: Signal can be pulled low by a device on the bus, but not driven high (if it is not outputting a “0”, the line is left floating)
- Open-drain IO port: P-MOS transistor is never activated.
- Outputting ‘0’ results in N-MOS activating, pulling output pin to 0V.
- Outputting ‘1’, causes pin voltage to float



- Eliminates possibility of **bus contention** (one device outputting ‘1’ while another one pulls it low does not create a short-circuit)
- **Pull-up resistor** on SCL and SDA lines: If no device is pulling the line to ground, the pull-up resistor(s) will ensure it is at a logical high condition
- Choice of pull-up resistor is not always straightforward – more on this later.

Address frame
Adres raam

- I2C makes use of addressing – each slave device has a unique address. Usually address is 7-bit (but 10-bit addressing is also possible)
- 7-bit address : $2^7 = 128$ possible addresses
- 10-bit address: $2^{10} = 1024$ possible addresses
- Two types of “transactions” on the I2C bus:
  - Master write: Master wants to send data to a particular slave
  - Master read: Master would like a particular slave to return data to it

Master write
Meester skryf

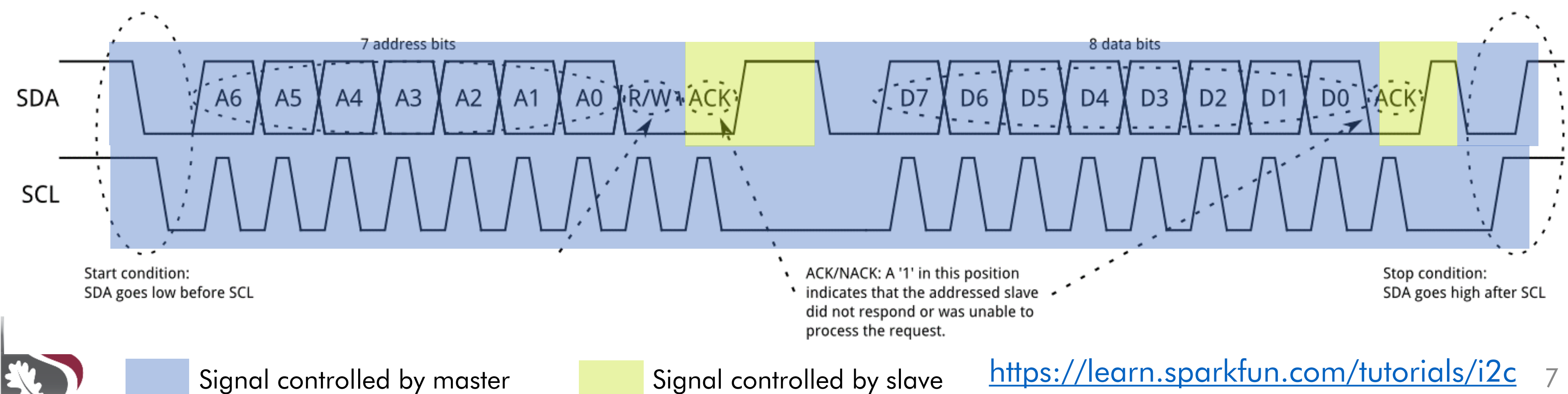
- All transactions start with an I2C "start" condition, followed by an address frame: Master identifies which slave it wants to communicate with, and also if it is a read or write transaction

Master action

1. Keep SCL high, and pull SDA low → I2C Start condition
2. Send 7x clocks and output intended slave address on SDA
3. Output write(0) bit
4. Release SDA line and sense ACK
5. Send 8x clocks and output 8x data bits
6. Release SDA line and sense ACK
7. Release SCL line and then SDA → I2C Stop

Slave action

1. Detect I2C start (all slaves)
2. Read transmitted slave address (all slaves)
3. Read read/write bit (all slaves)
4. Slave with address match pulls SDA line low (ACK)
5. Slave reads data bits (only slave with address match)
6. Slave with address match pulls SDA line low (ACK)
7. Detect I2C stop (all slaves)



https://learn.sparkfun.com/tutorials/i2c

Master read
Meester lees

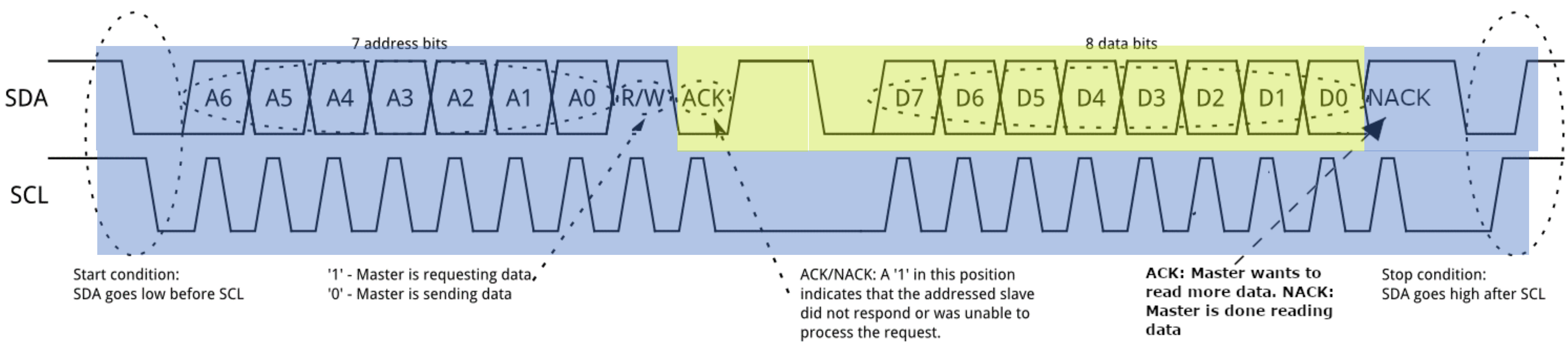
- Master reads start off exactly the same, but master will output a ‘1’ for the R/W bit. Master still controls clock signal, but senses the state of SDA instead of actively controlling it

Master action

1. Keep SCL high, and pull SDA low → I2C Start condition
2. Send 7x clocks and output intended slave address on SDA
3. **Output read(1) bit**
4. Release SDA line and sense ACK
5. Send 8x clocks and **read** 8x data bits
6. **Send ACK bit** (want to read more bytes) **or NACK** (done reading)
7. Release SCL line and then SDA → I2C Stop

Slave action

1. Detect I2C start (all slaves)
2. Read transmitted slave address (all slaves)
3. Read read/write bit (all slaves)
4. Slave with address match pulls SDA line low (ACK)
5. **Slave outputs data bits** (only slave with address match)
6. **Slave senses ACK/NACK**
7. Detect I2C stop (all slaves)



Signal controlled by master

Signal controlled by slave

https://learn.sparkfun.com/tutorials/i2c

I2C

- Data always transmitted with one byte (8 bits) per frame
- Most significant bit transmitted first
- Complicated hardware implementation

- Hardware implementation handles start/stop detection, address match, read and write transactions, and generates ACKs and NACKs
- Interpretation of data is done in software

- When reading data from I2C device, there is usually multiple data items that can be read. How do you (the microcontroller master) instruct the slave which data item you want to read?
- Answer: First perform an I2C write transaction with the data "identifier"
- Usually the term "register" is used for this (the data item in the I2C slave device you are interested in). Note that this has nothing to do with the CPU registers, or memory-mapped peripheral registers. It is implemented external to the microcontroller on the slave device. From our perspective we can only read and write

I2C Example
I2C Voorbeeld

- Example: MEMS accelerometer on STM development board (LSM303DLHC)
- We want to read the X,Y,Z acceleration vector components
- Each of X, Y and Z is a 16-bit value, consisting of low (L) and high (H) byte
- I2C master write:
  - I2C start
  - Slave address: 0011001_2
  - R/W bit: 0 (write)
  - Data byte: 0xA8
    = 0x28 (OUT_X_L_A) | 0x80
  - I2C stop
- I2C master read:
  - I2C start
  - Slave address: 0011001_2
  - R/W bit: 1 (read)
  - Read data byte (OUT_X_L_A returned)
  - Read data byte (OUT_X_H_A returned)
  - Read data byte (OUT_Y_L_A returned)
  - …
  - I2C stop

Table 17. Register address map

Name	Slave address	Type	Register address		Default	Comment
			Hex	Binary		
Reserved (do not modify)	Table 14		00 - 1F	--	--	Reserved
CTRL_REG1_A	Table 14	rw	20	010 0000	00000111	
CTRL_REG2_A	Table 14	rw	21	010 0001	00000000	
CTRL_REG3_A	Table 14	rw	22	010 0010	00000000	
CTRL_REG4_A	Table 14	rw	23	010 0011	00000000	
CTRL_REG5_A	Table 14	rw	24	010 0100	00000000	
CTRL_REG6_A	Table 14	rw	25	010 0101	00000000	
REFERENCE_A	Table 14	rw	26	010 0110	00000000	
STATUS_REG_A	Table 14	r	27	010 0111	00000000	
OUT_X_L_A	Table 14	r	28	010 1000	output	
OUT_X_H_A	Table 14	r	29	010 1001	output	
OUT_Y_L_A	Table 14	r	2A	010 1010	output	
OUT_Y_H_A	Table 14	r	2B	010 1011	output	
OUT_Z_L_A	Table 14	r	2C	010 1100	output	
OUT_Z_H_A	Table 14	r	2D	010 1101	output	

Table 14. SAD+Read/Write patterns

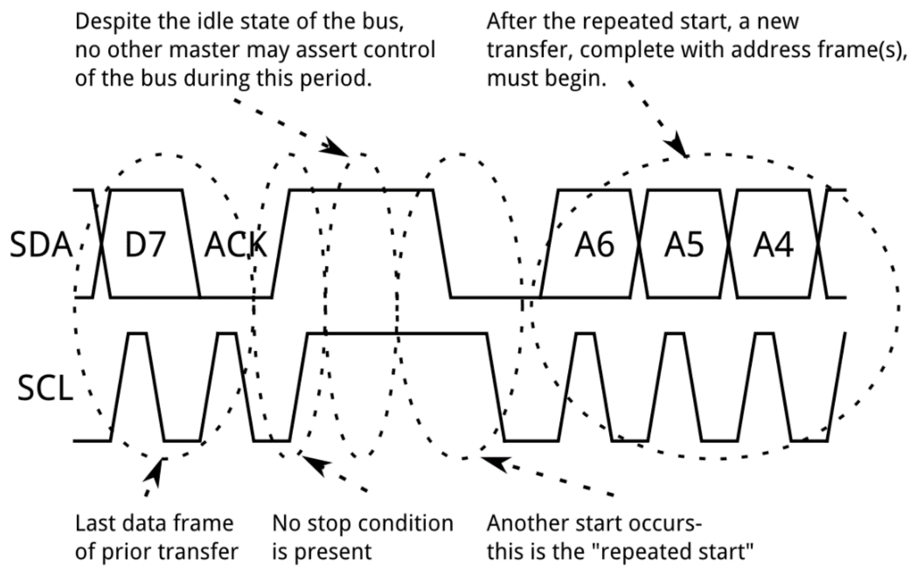
Command	SAD[7:1]	R/W	SAD+R/W
Read	0011001	1	00110011 (33h)
Write	0011001	0	00110010 (32h)

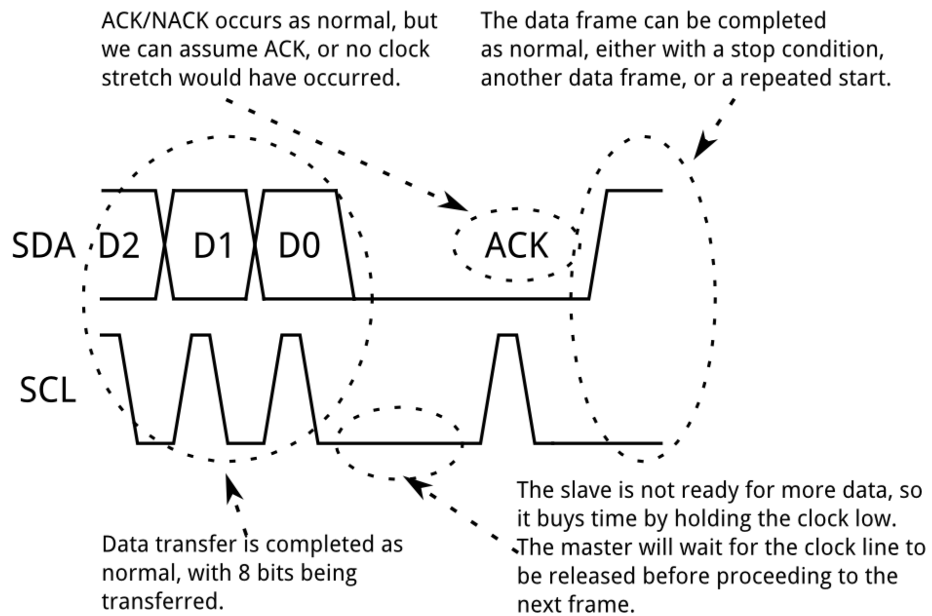
Register index from where data is returned will auto-increment (this is device specific. Read the datasheet!)

I2C Advanced Concepts
I2C Gevorderde Konsepte

- Repeated Start condition
  - Used by master to perform multiple I2C transactions (read or write), without releasing the bus

- Clock stretching
  - Used by slave to "buy time" before responding to read request
  - Master releases clock signal after address or data frame
  - Slave keeps clock low. During this time master may not control the clock





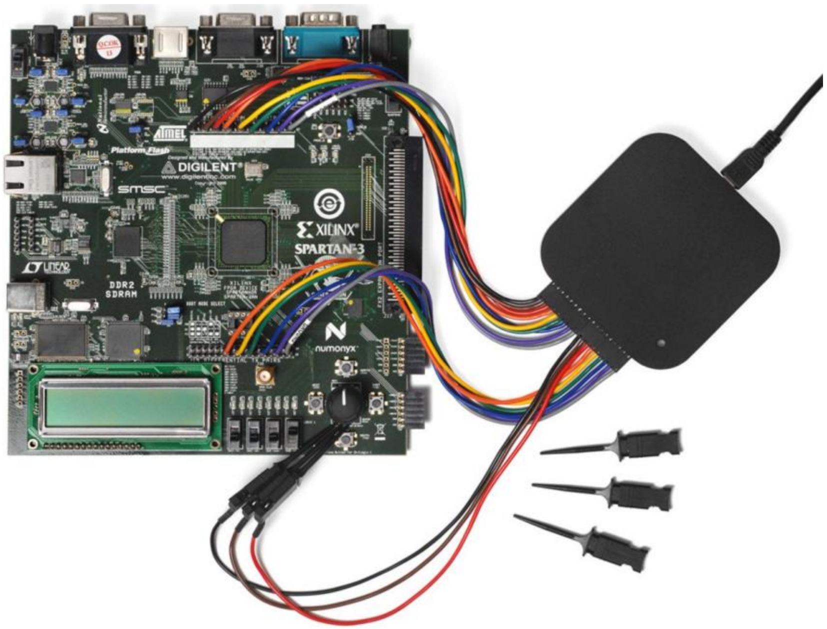
https://learn.sparkfun.com/tutorials/i2c

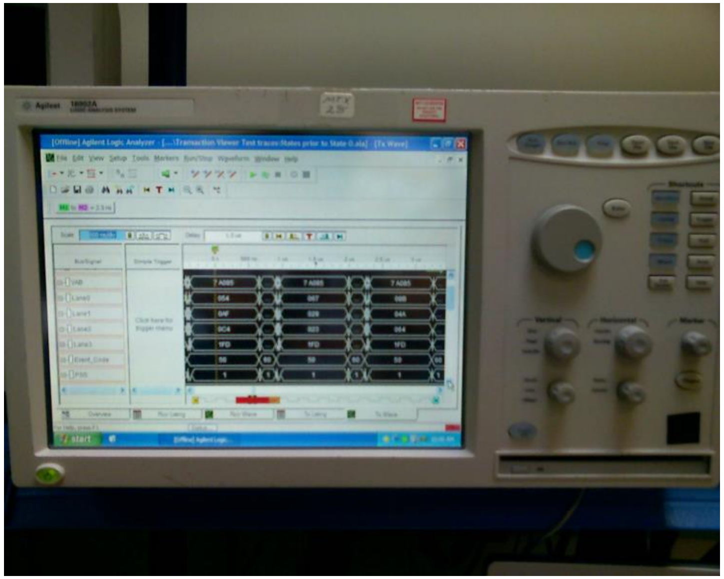
Clock speed and Signal Levels
Klokspoed en Seinvlakke

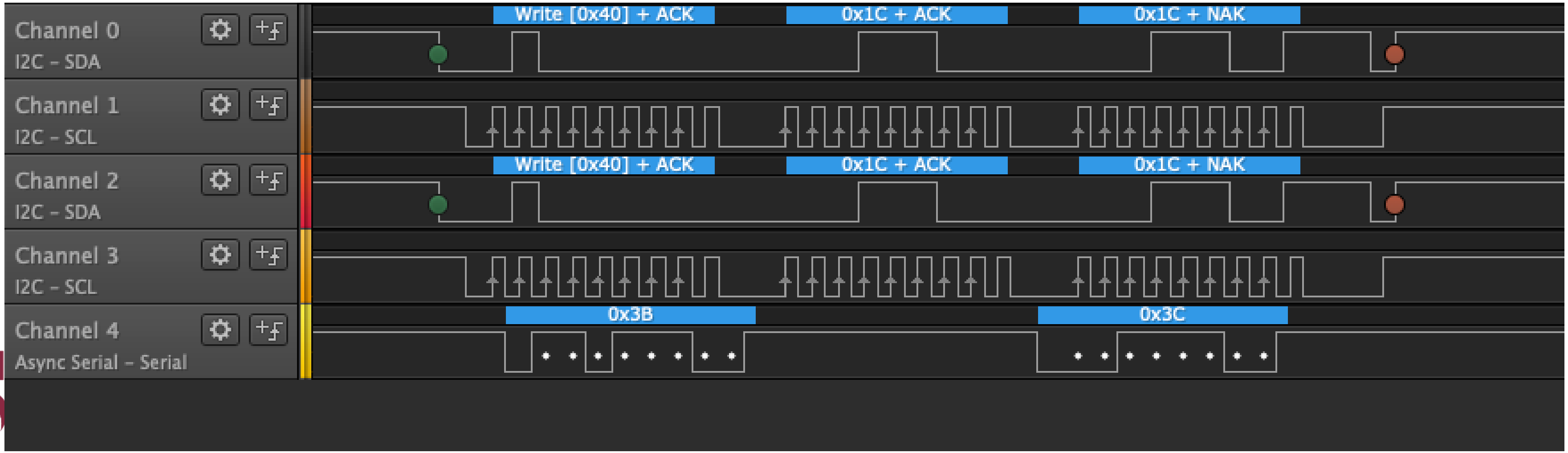
- Bus speeds of 100 kHz to 400 kHz are typical
- Newer standard introduces higher speeds (up to 3.4 MHz)
- Signal levels: Typically 5V or 3.3V.
  - Connect pull-up resistors to lowest supply voltage
  - Use signal level-shifter if needed
  - Use logic buffer if there is a possibility that one device on the bus may be powered down – prevents back-powering of dead logic
- Pull-up resistor value is a function of bus speed and capacitance

Logic analyser
Logika analiseerder

- Useful debugging tool…





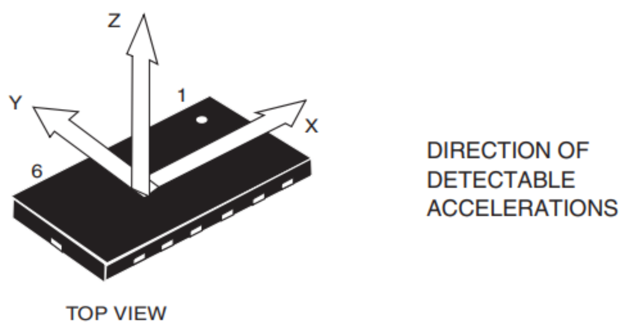


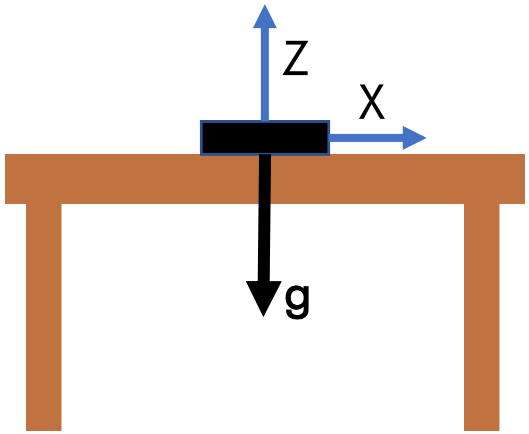
Further reading
Verdere leesmateriaal

- https://learn.sparkfun.com/tutorials/i2c
- Texas Instruments: Application Report SLVA704 - Understanding the I2C Bus - https://www.ti.com/lit/an/slva704/slva704.pdf

3D Linear Accelerometer
3D Liniêere Versnellingsensor

- The sensor measures linear acceleration
- While it is stationary (on the Earth surface), it measures the Earth's gravitational acceleration ($1\text{g} = 9.81\text{m/s}^2$)

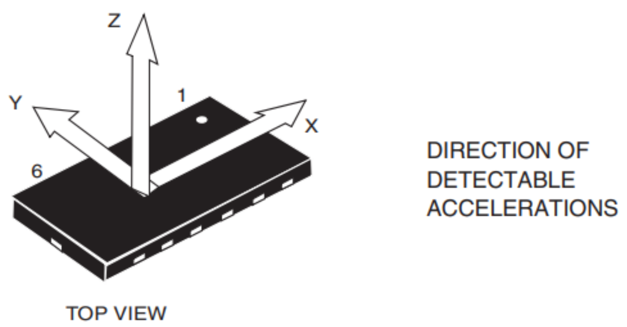


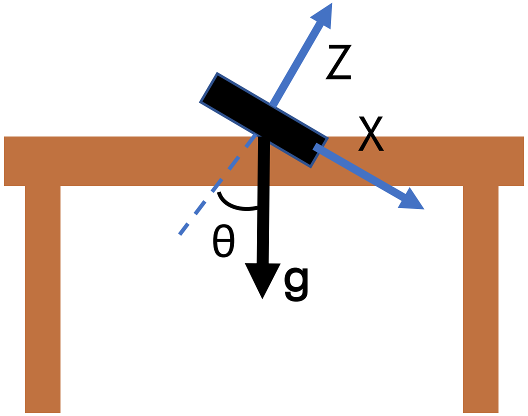


$$\boldsymbol{a} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} g$$

3D Linear Accelerometer
3D Liniêere Versnellingsensor

- The sensor measures linear acceleration
- While it is stationary (on the Earth surface), it measures the Earth's gravitational acceleration ($1g = 9.81 m/s^2$)
- When I rotate it, the 1g vector transfers to the non-Z components





$$\boldsymbol{a} = \begin{bmatrix} \sin\theta \\ 0 \\ -\cos\theta \end{bmatrix} g$$