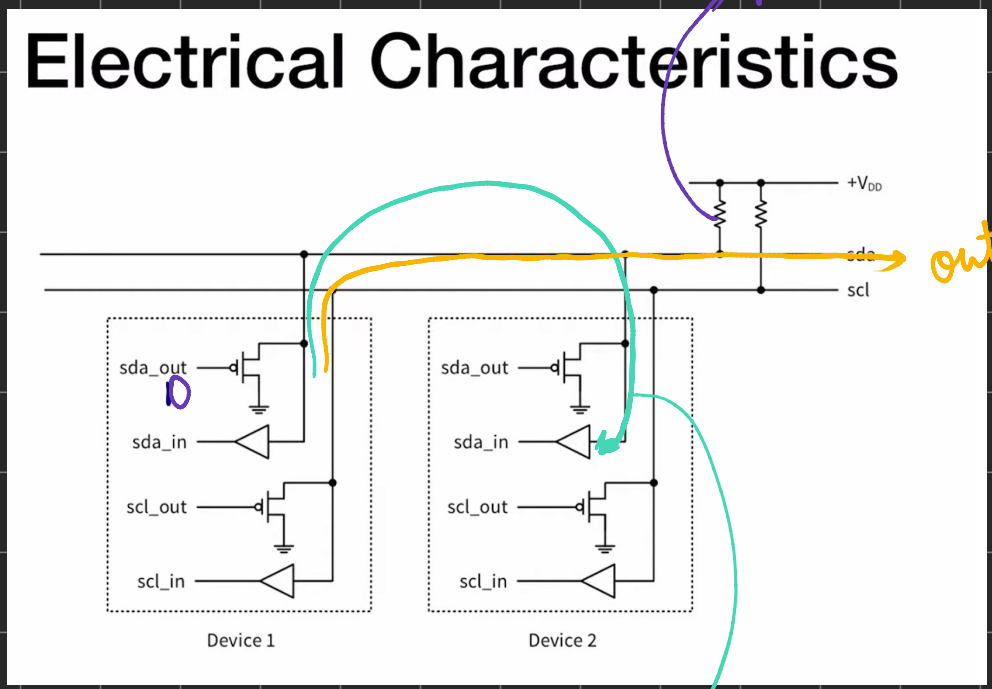
I²C	SPI
• Single bidirectional data line	• 2 data Lines miso / mosi
• I2C clock generated by master	• SPI ——— ——— master.
• Multiple devices on the I2C bus can be reached by unique address	• slave device can be activated by ss_n signal.
• supports multiple master	• contain a single master.

→ uses open drain technology to operate it's Bus.



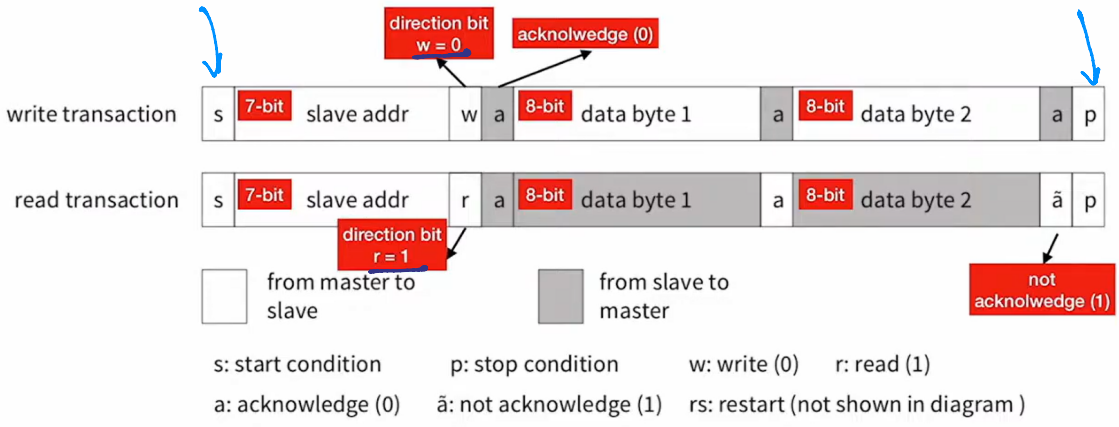
always listening

→ only a master can start data transfer

Basic bus protocol

start condition

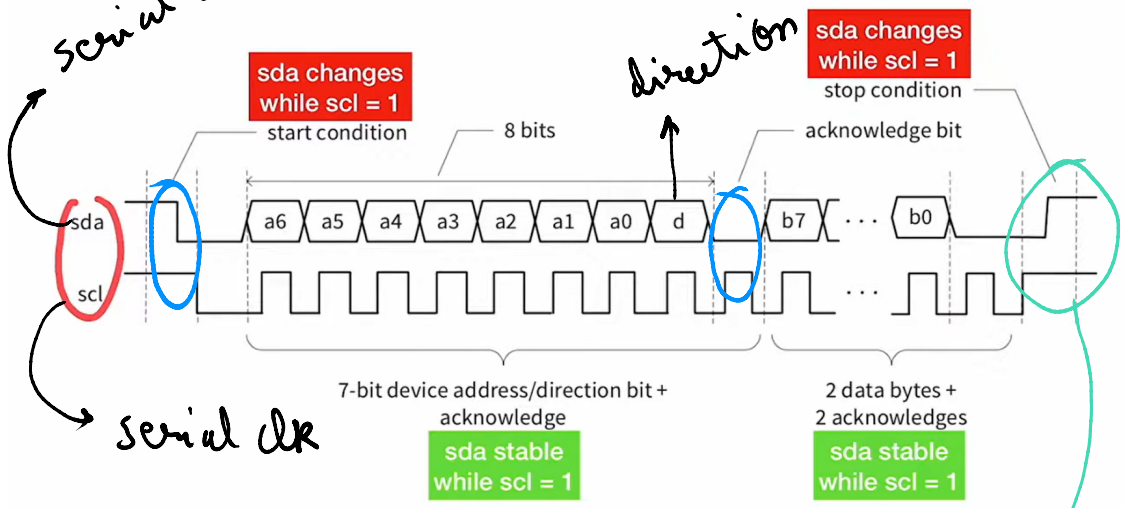
stop condition



Timing

serial data

direction



serial clk

→ ideal bus will be high (1)

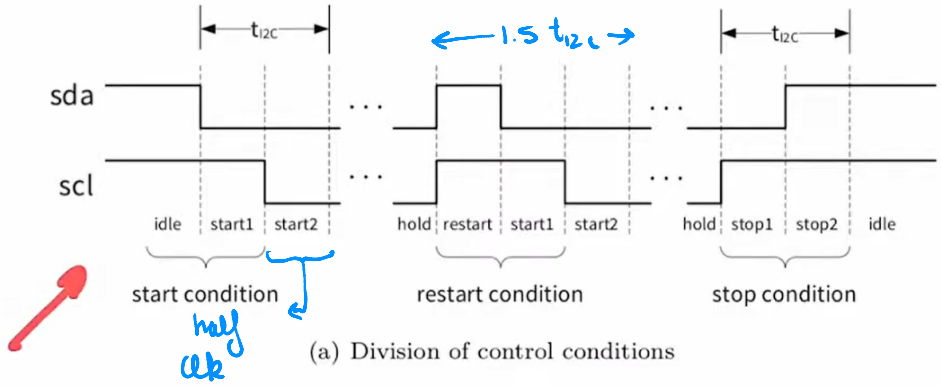
→ when data changes while in high condition (scl = 0 & sda = ~sda), it's a stop condition

I2C Controller

Specifies
- → start condition
- → stop condition
- → Bit level timing

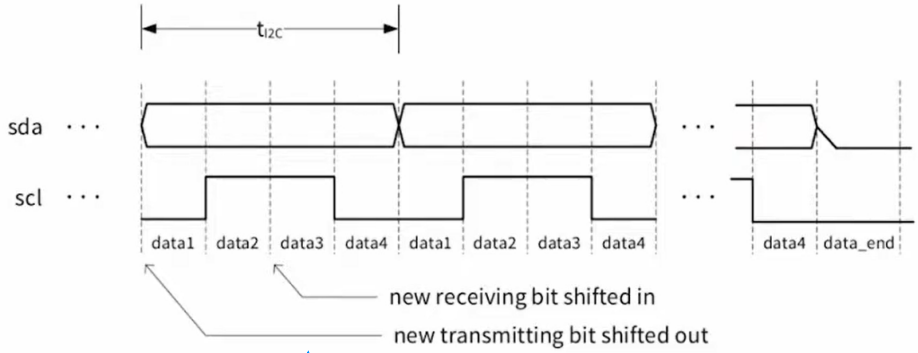
Does not specify:
- → no of bytes in a transaction

I²C Action Phases

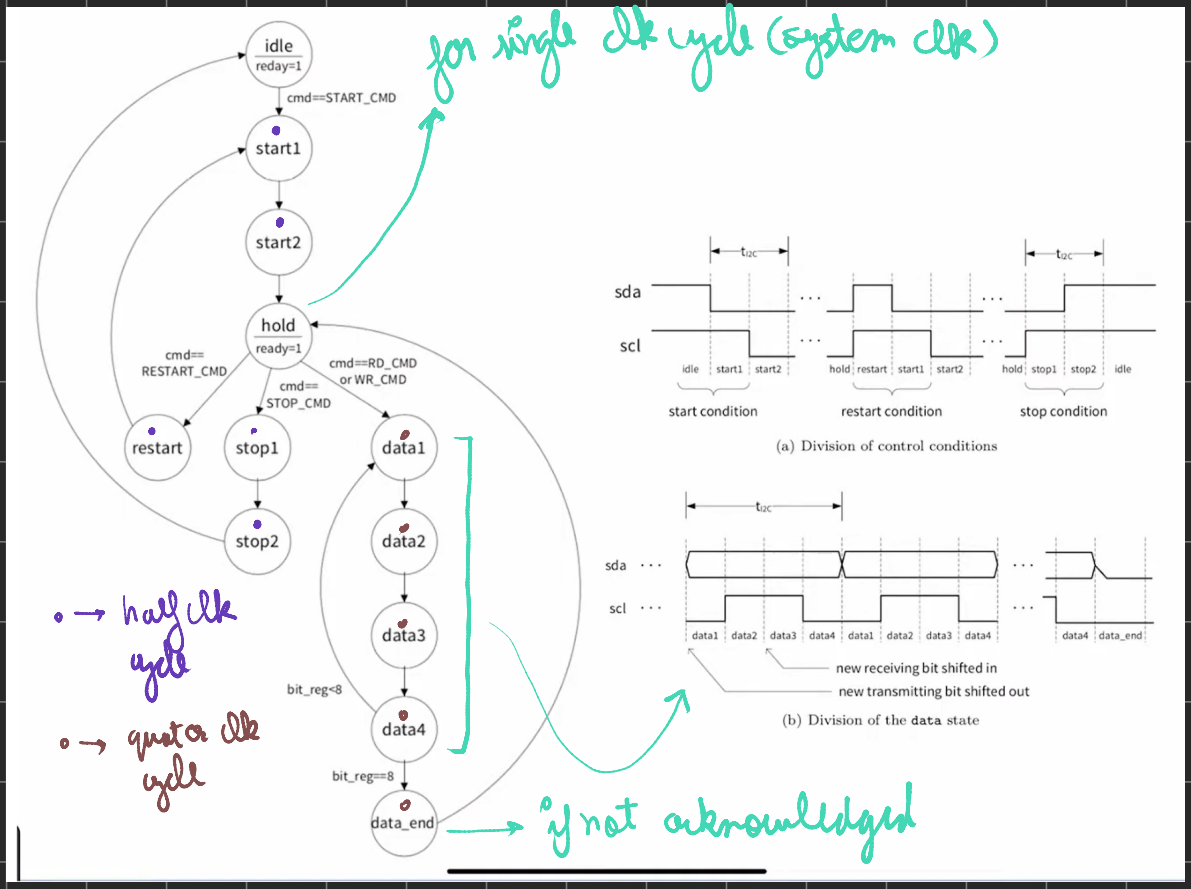


(a) Division of control conditions

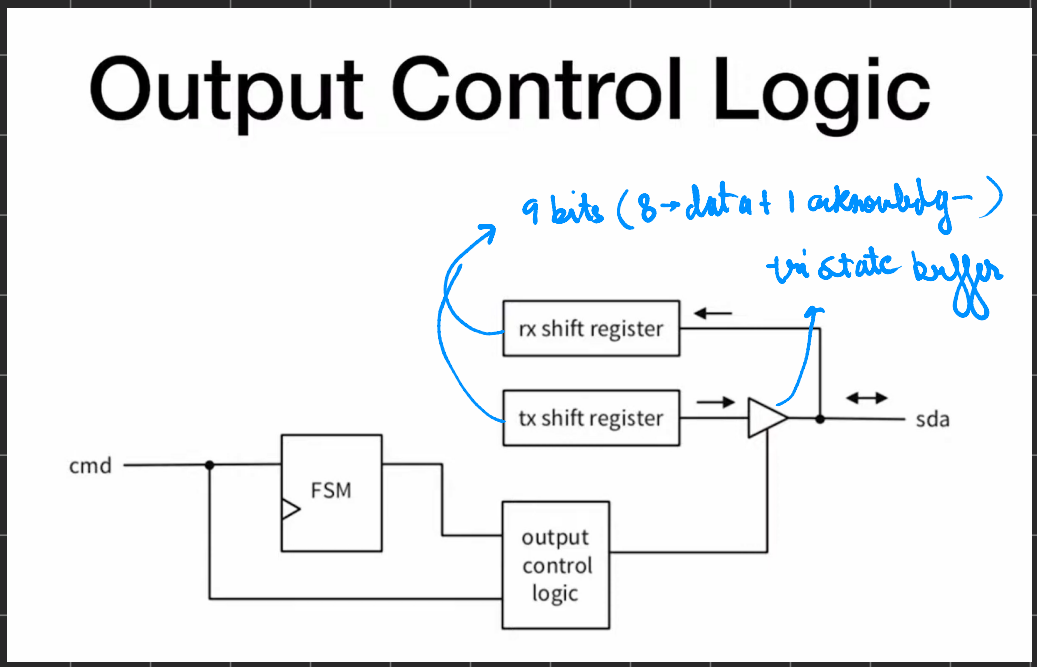
half clk



data receiving



→ cmd to choose between these 5 states



Output control logic

- all the operation in RX & TX register happens simultaneously: at some time

↳ Writing (Master sends data to slave)
→ master sends 8 data bits, then wait to receive 9th (Ack.) from slave.

	Action	When	Detail
①	Load TX Register	before Bit 1.	Load 8 bits of the data to be sent. The 9th bit (ack) is irrelevant
②	Activate tri-state Buffer	clk cycle 1-8	The output control logic puts a '1' to the buffer. connecting the shift register to SDA line.
③	Data Transfer	clk cycle 1-8	Data bits are transferred one at a time, data goes onto SDA line to the slave.
④	Receiving Shift Register operation	clk — 1-8	The RX shift register simultaneously receives the data being sent.
⑤	Deactivate tri-state Buffer	clk cycle 9 (ack)	The FSM identifies the ACK bit stage.

The output control Logic puts a '0' to the buffer, disconnecting the shift register.

(6)	Line Release & slave control	clk cycle 9 (clk)	Disconnecting the buffer release the data line, allowing the slave to take control. The line defaults to '1'.
(7)	Acknowledge Reception	clk cycle 9 (clk)	The slave asserts the signal (sends a '0') if it acknowledged the data. This signal moves the receiving & the RX register.
(8)	Read ACK status	After clk 9	The master reads the LSB of the RX shift register to determine if the slave ack- the data or not.

↳ Reading

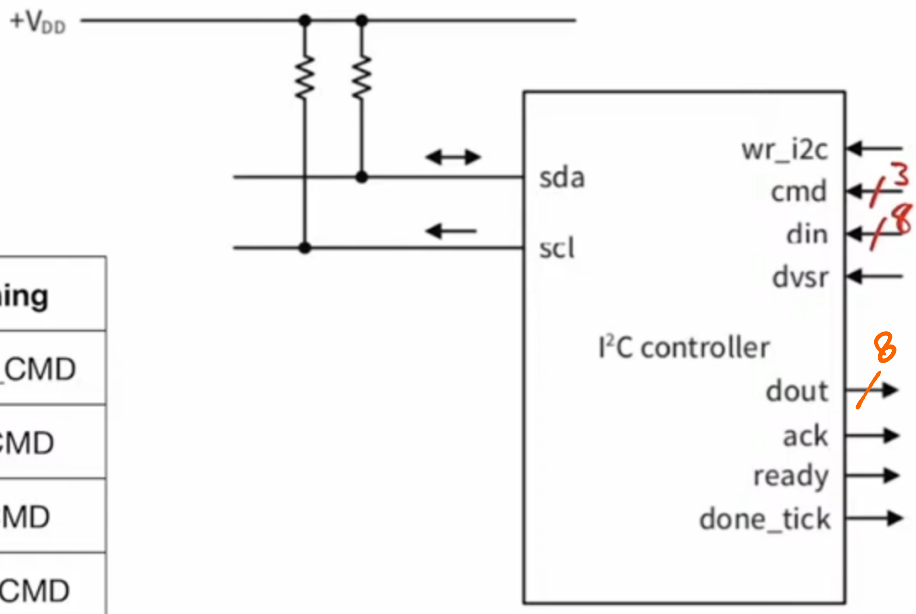
Step	Action/Component	When (Bits)	Details	Citations
Preparation	Load TX Register	Before Bit 1	Load the TX register with "x's" (dummy data, irrelevant 0s or 1s) for the first 8 bits. The 9th bit (ACK position) is loaded with the intended response (typically **'0'** for ACK, or '1' if the Master wants the Slave to stop) 6 8 . 6 8	
1.	Keep Buffer Deactivated	Clock cycles 1-8 (Data)	The Master's output control logic ensures the line is available for the Slave to send data. The buffer remains effectively disconnected (the register data stays at '0' in the source example) 6 7 . 7	
2.	Data Reception	Clock cycles 1-8 (Data)	The Slave sends 8 bits of data. The RX shift register receives this data 6 7 . 6 7	
3.	TX Shift Register Operation	Clock cycles 1-8 (Data)	The TX register shifts out the dummy 'x' bits, which are thrown out because the buffer is disconnected 6 7 . 6 7	
4.	Activate Tri-State Buffer	Clock cycle 9 (ACK)	The Output Control Logic knows it needs to send an ACK/NACK. It asserts the buffer by putting a **'1'** 6 . 6	
5.	Acknowledge Transmission	Clock cycle 9 (ACK)	The Master sends the pre-loaded ACK ('0') or NACK ('1') to the Slave 6 8 . 6 8	
6.	Receiving Shift Register Operation	Clock cycle 9 (ACK)	The RX register simultaneously receives the Master's own transmitted ACK (which is dummy data and ignored) 8 . 8	
7.	Data Ready	After Clock cycle 9	The received data byte is available in the RX shift register 8 . 8	

I2C scl speed

- Std. mode : 0 → 100 kHz
- Fast mode : up to 400 kHz
- High-speed mode : up to 3.4 MHz

$$f_{divisor} = \frac{f_{system}}{4 \times f_{I2C}}$$

I²C Controller



cmd	Meaning
000	START_CMD
001	WR_CMD
010	RD_CMD
011	STOP_CMD
100	RESTART_CMD