Basics of I2C: Advanced Topics

TIPL 6104
TI Precision Labs – Digital Communications

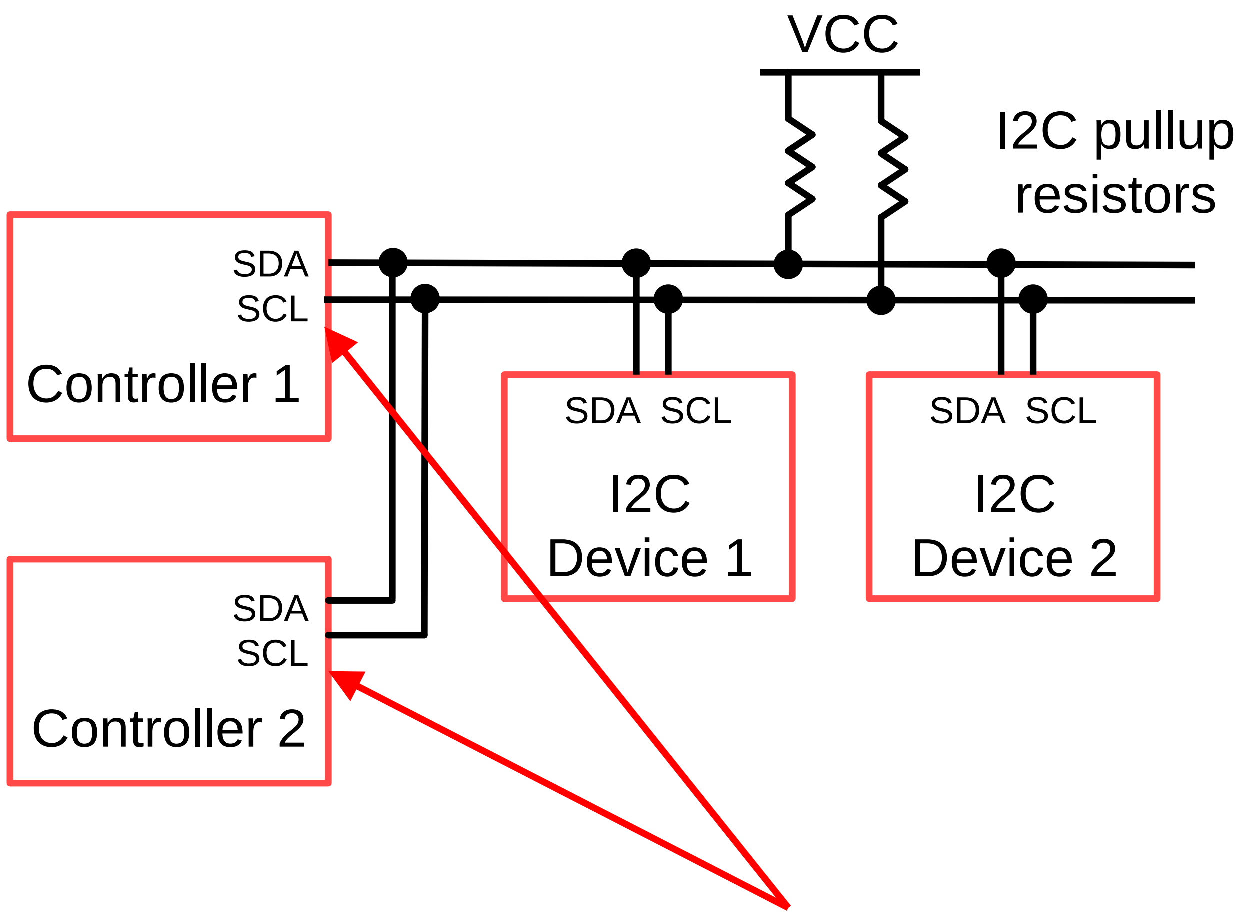
Prepared by Joseph Wu

Presented by Alex Smith

Basics of I2C – Advanced Topics

- **Clock Synchronization and Arbitration**
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

Clock Synchronization and Arbitration

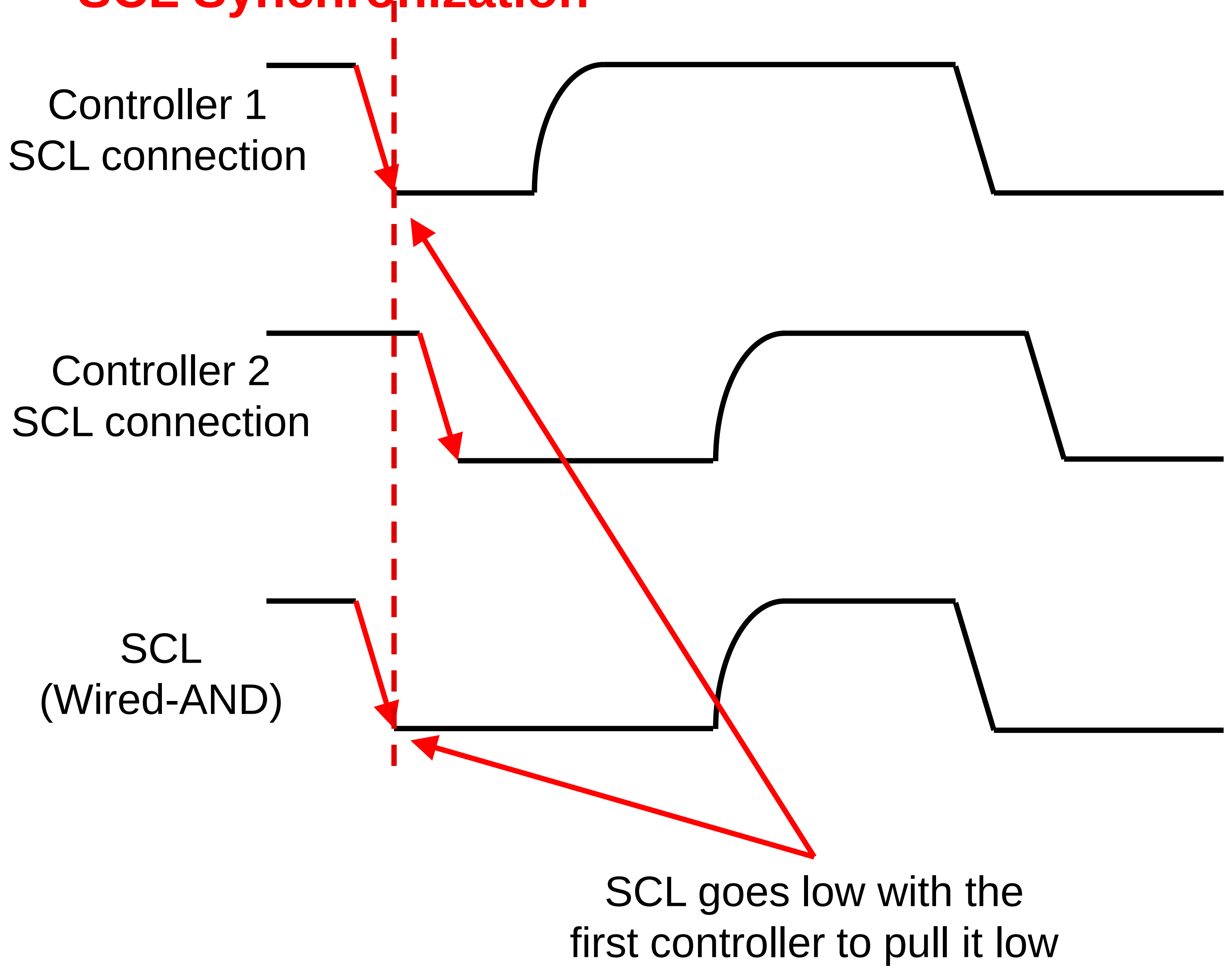


If both controllers try to use the bus at the same time, there must be a way to resolve contention without disrupting communications

- Multiple controllers on the bus means there can be more than one device trying to claim the bus at the same time
- The open-drain wired-AND connection allows for two devices to try to claim the bus without disruptive contention
- I2C uses clock synchronization and then arbitration to determine which controller claims the bus

Clock Synchronization and Arbitration

SCL Synchronization



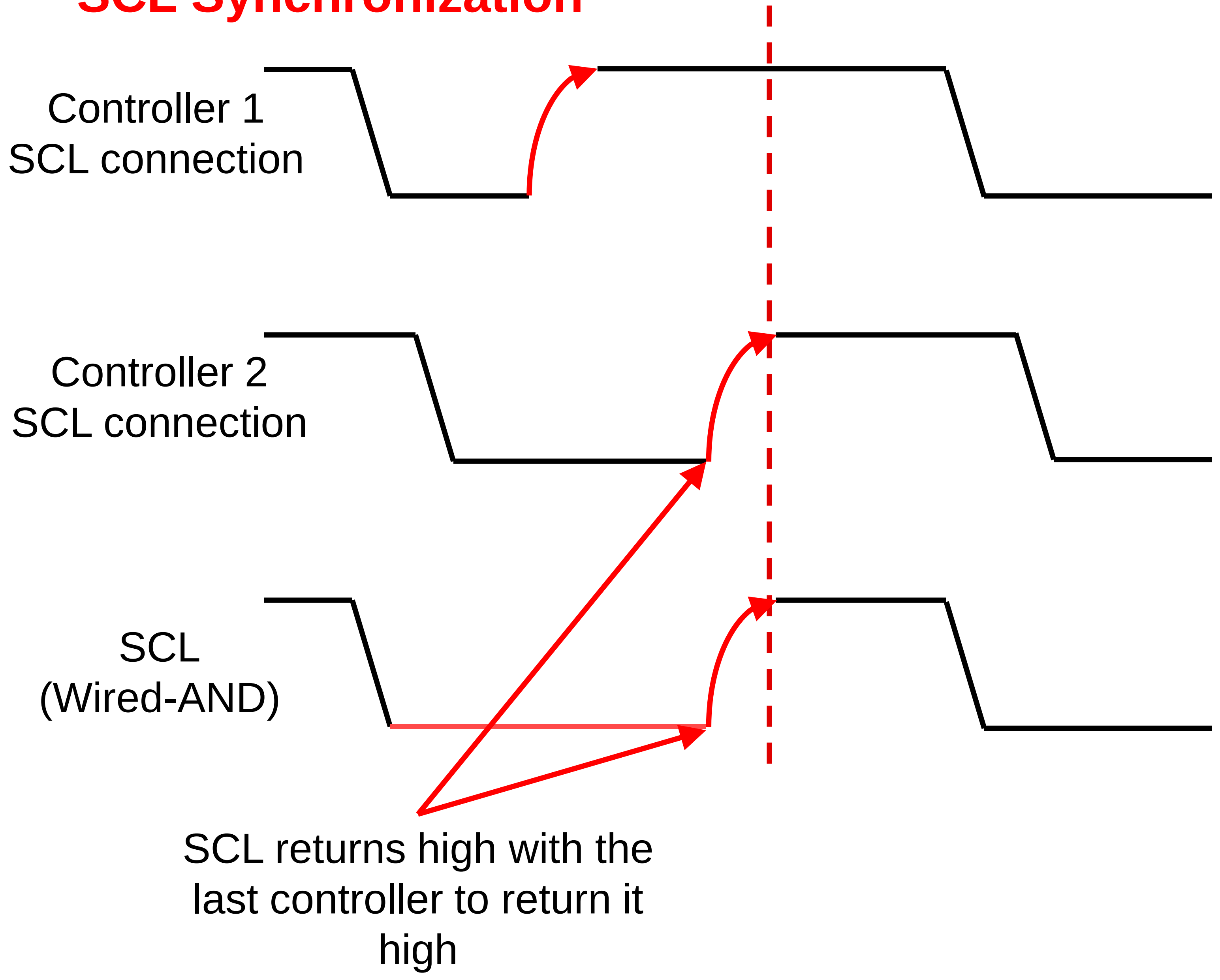
- Clock synchronization is done when the I2C controllers try to claim the bus
- Two controllers try to initiate START condition near the same time
- Wired-AND connection; SCL is low if any controller pulls SCL low; SCL is high only if both controllers are setting the line high

Truth Table

Controller 1 SCL	Controller 2 SCL	Resulting SCL
0	0	0
0	1	0
1	0	0
1	1	1

Clock Synchronization and Arbitration

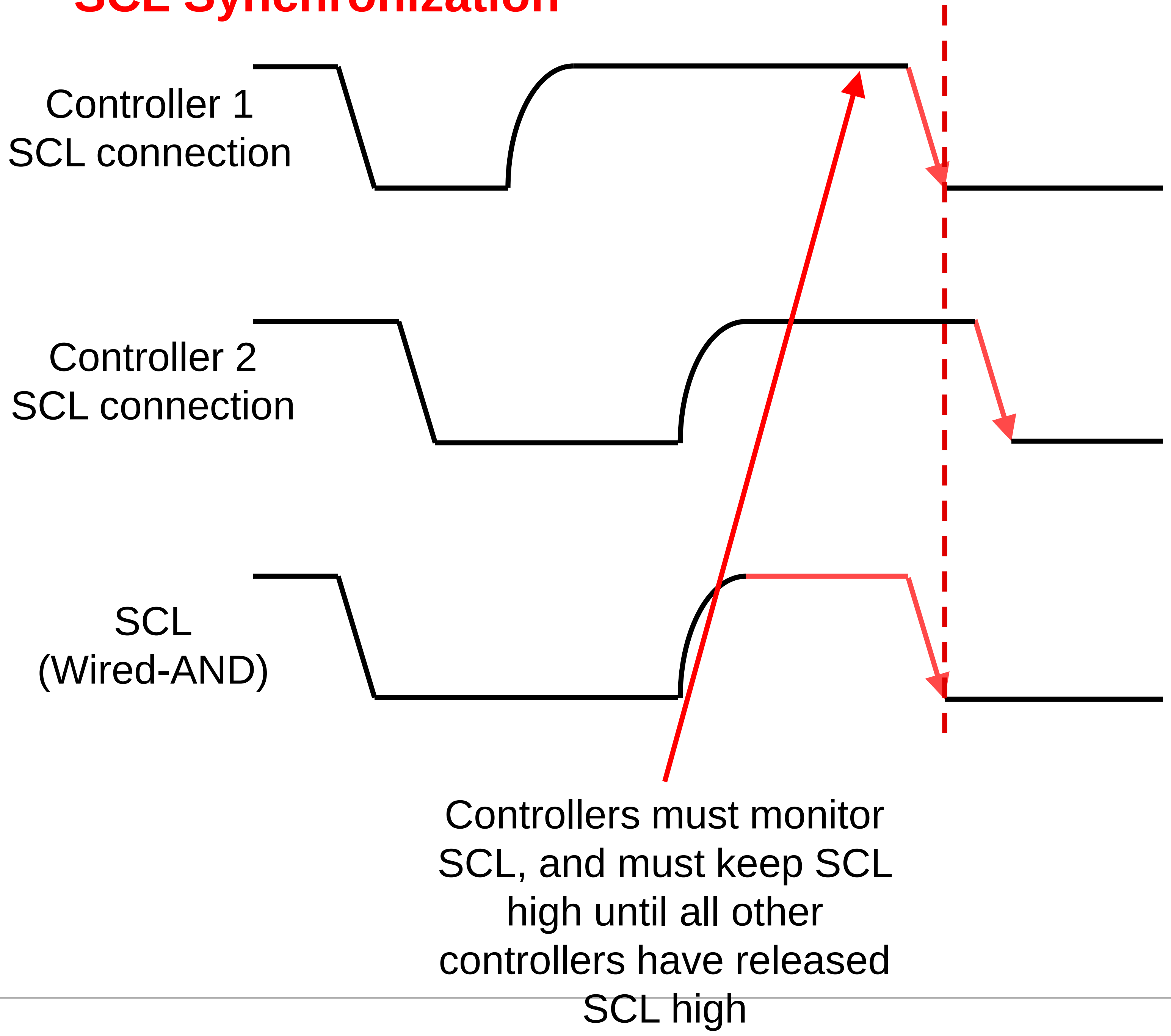
SCL Synchronization



- SCL synchronization continues when the controller devices release SCL
- All active controllers still need to monitor SCL to ensure they are able to complete the SCL pulse
- If another controller has also pulled down on SCL, the other controllers cannot proceed with the SCL pulse and must wait until SCL is released
- SCL stays low for as long as the longest period of time for which SCL is pulled low by any controller

Clock Synchronization and Arbitration

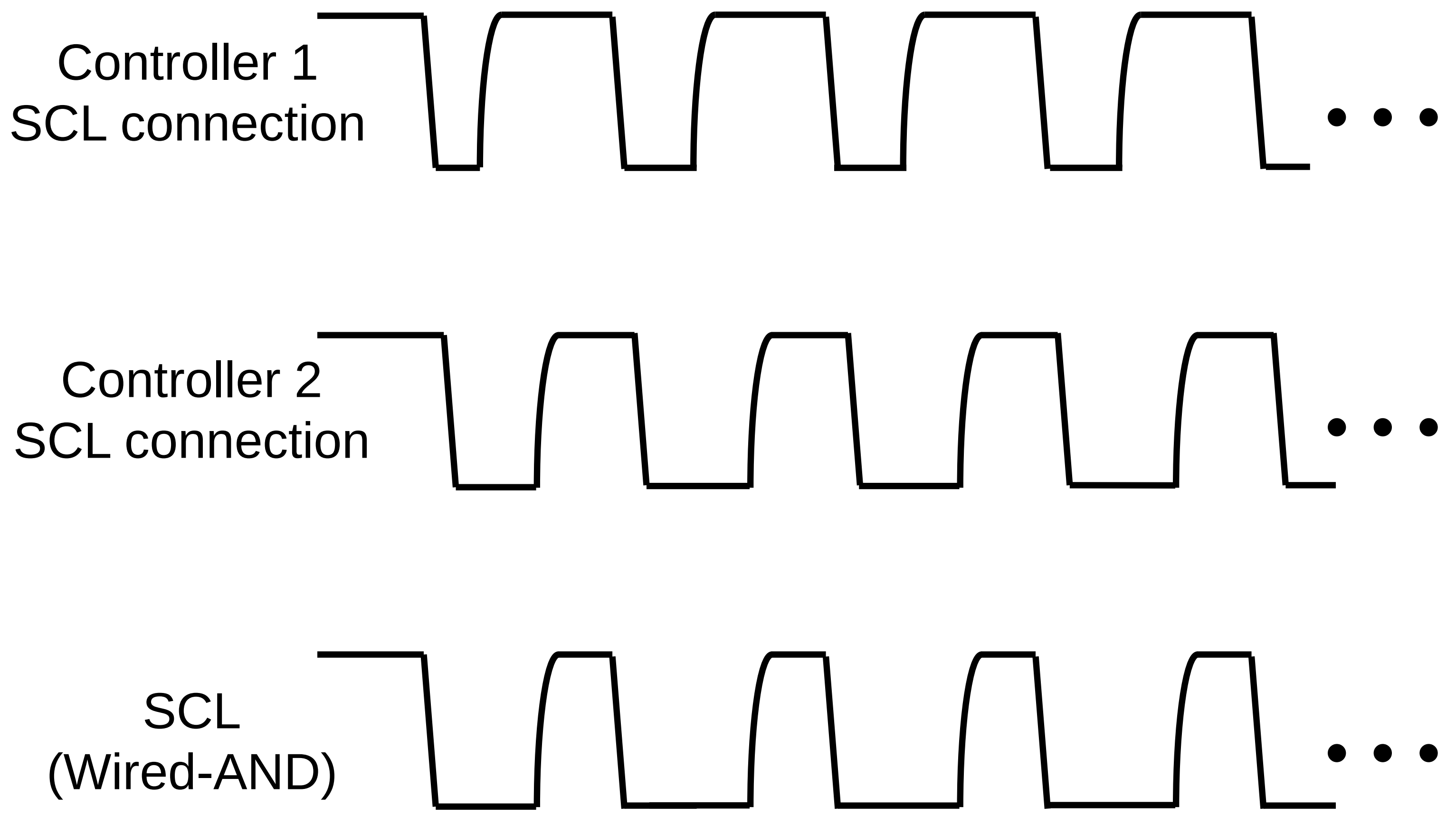
SCL Synchronization


Controller 1
SCL connection
Controller 2
SCL connection
SCL
(Wired-AND)
Controllers must monitor SCL, and must keep SCL high until all other controllers have released SCL high

- Synchronization continues as all controllers have released SCL high
- After the rising edge of SCL, all controllers pull down on SCL to complete the SCL pulse
- The first controller that completes the SCL high time determines the high time period of the pulse

Clock Synchronization and Arbitration

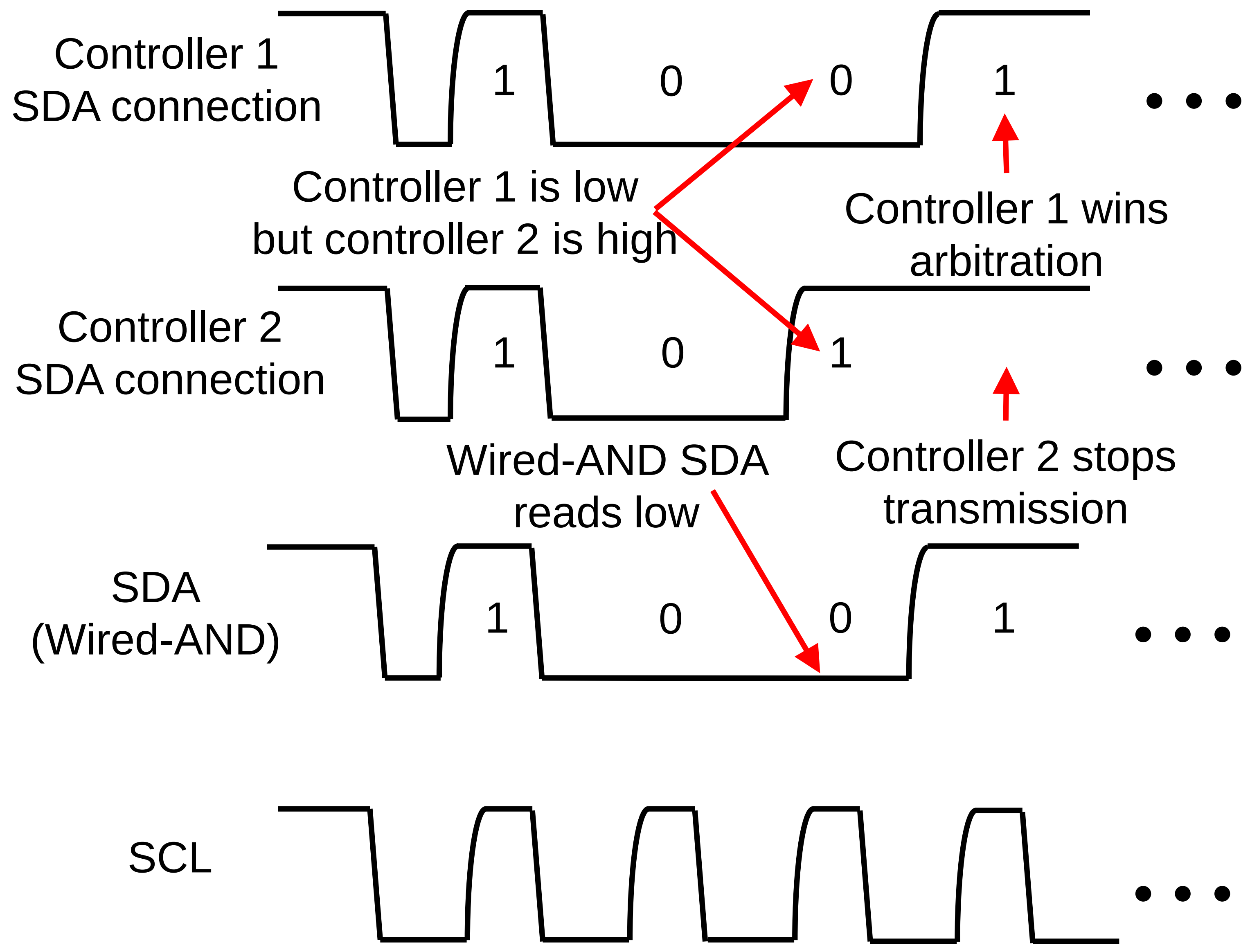
SCL Synchronization



- SCL continues to be synchronized to the same clock with the same method
- SCL clock is generated with its low period determined by the controller with the longest clock low period
- The high period is determined by the controller with the shortest clock high period.

Clock Synchronization and Arbitration

SDA Arbitration

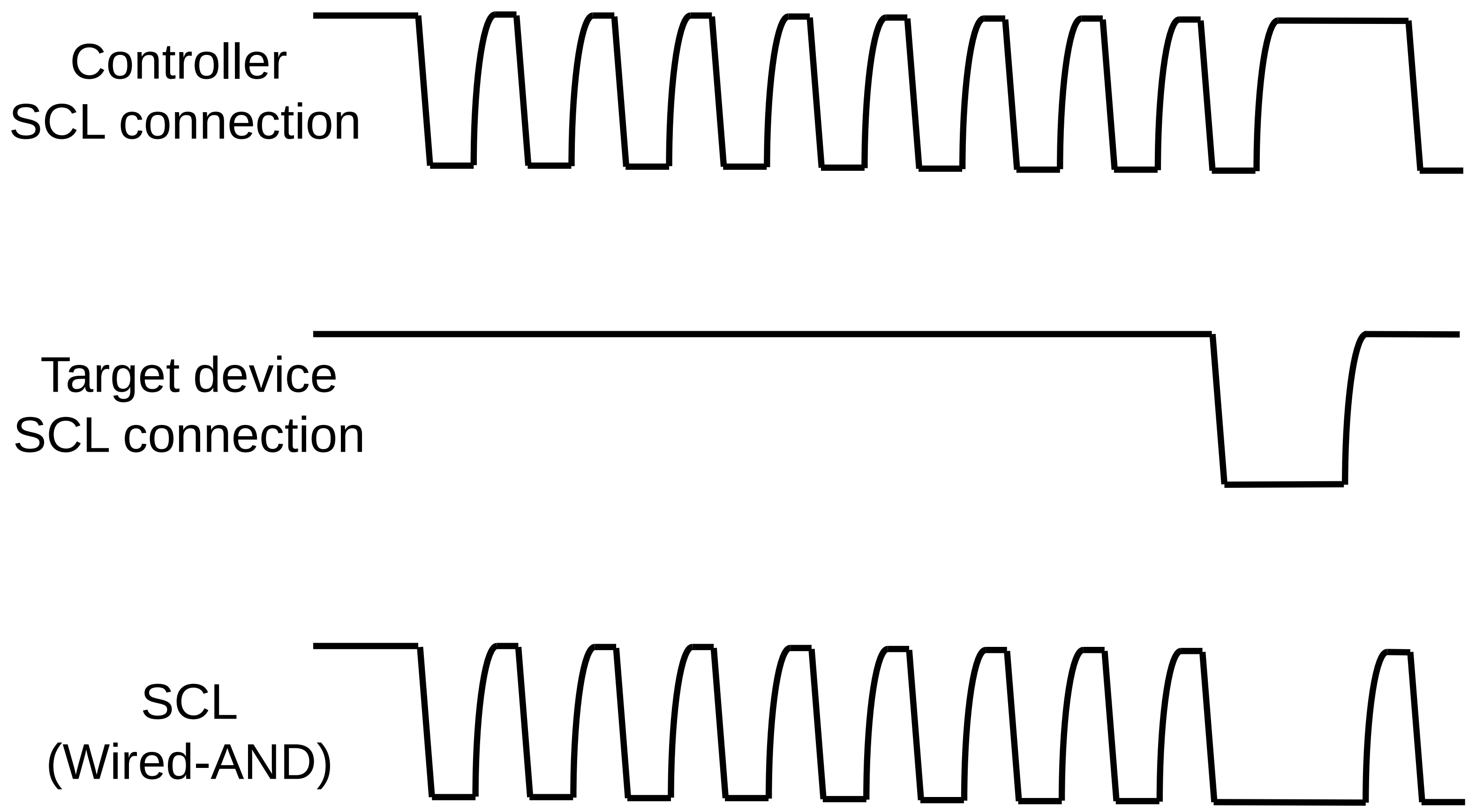


- Arbitration is done on SDA using a synchronized SCL
- Both controllers transmit data on SDA normally, but both controllers monitor SDA bit by bit
- The first controller to transmit a low bit, while the other controller transmits a high bit wins arbitration
- The controller that does not win arbitration stops transmission and waits for the STOP

Basics of I2C – Advanced Topics

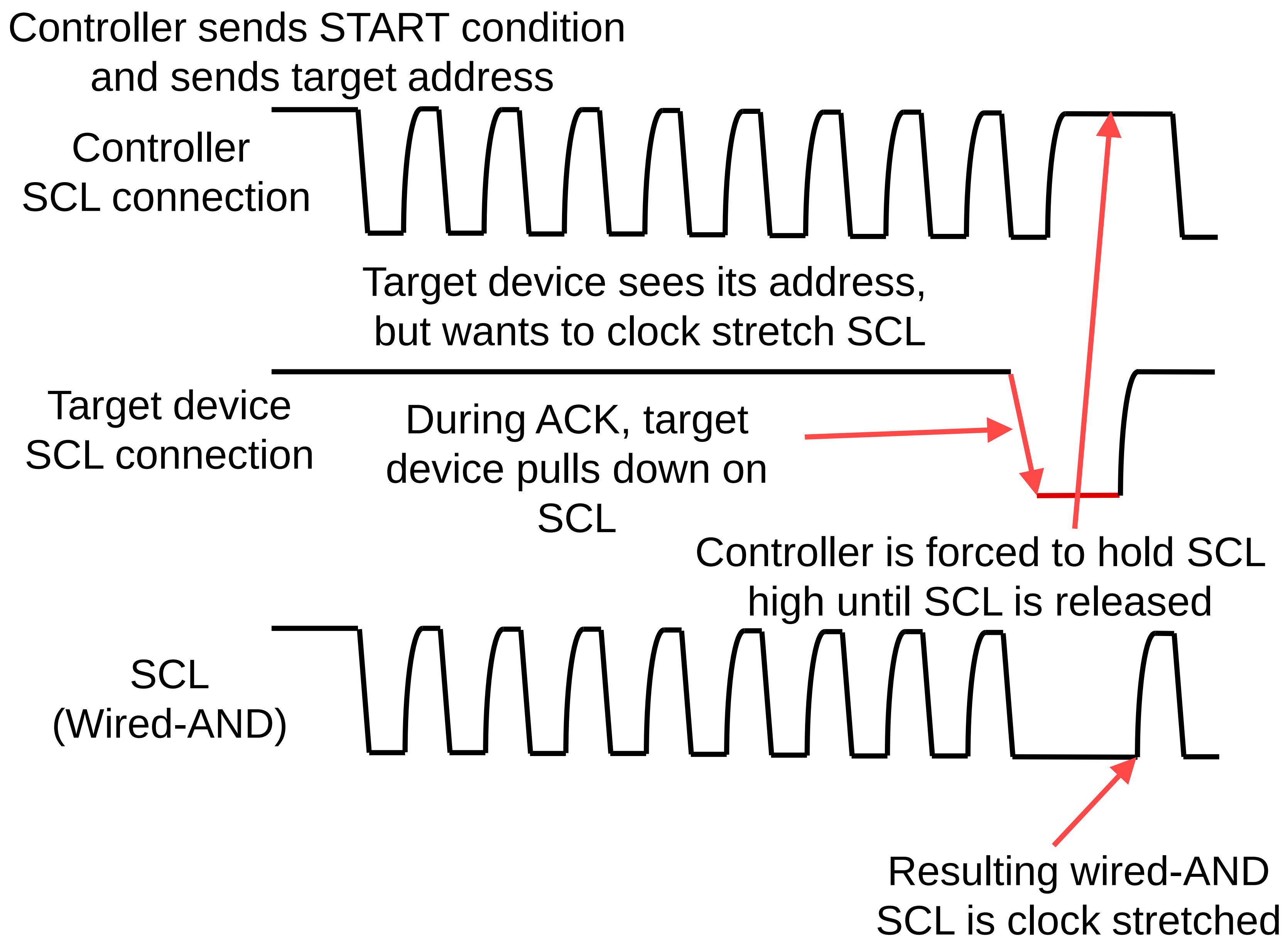
- Clock Synchronization and Arbitration
- **Clock Stretching**
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

Clock Stretching



- Target devices may clock stretch SCL to slow down I2C communications
- SCL held low for a time period by target device, often on ACK pulse
- controller must monitor SCL and extends SCL pulse to accommodate target's clock stretching
- There is no time limit to the target's clock stretching in the specification

Clock Stretching


Controller sends START condition and sends target address
Controller SCL connection
Target device sees its address, but wants to clock stretch SCL
Target device SCL connection
During ACK, target device pulls down on SCL
Controller is forced to hold SCL high until SCL is released
SCL (Wired-AND)
Resulting wired-AND SCL is clock stretched

- Clock stretching example starts with controller sending target address
- After address sent, responding target stretches SCL during ACK
- Controller monitors SCL and cannot proceed to next clock pulse until target releases SCL
- Wired-AND SCL shows stretched clock at after address byte sent without disrupting communication

Basics of I2C – Advanced Topics

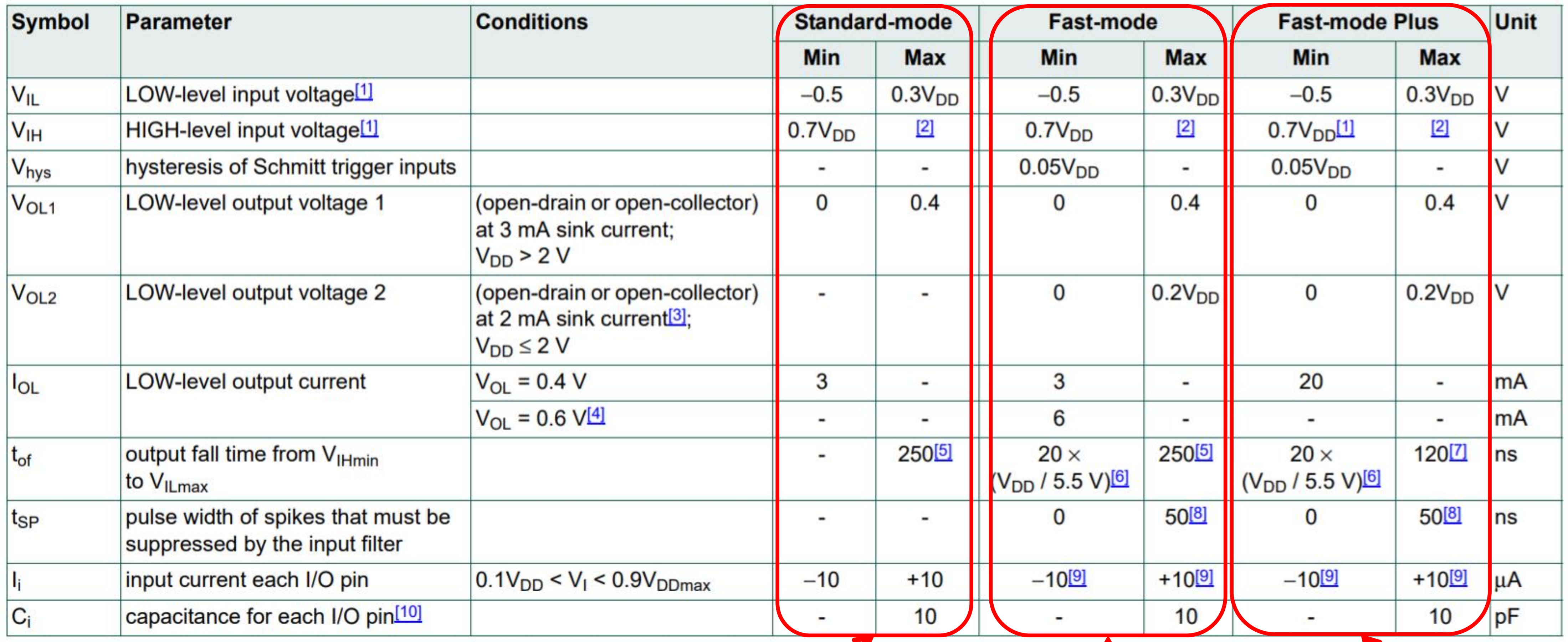
- Clock Synchronization and Arbitration
- Clock Stretching
- **Electrical and Timing Specifications**
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

Electrical and Timing Specifications

SDA and SCL I/O Characteristics

Table 9. Characteristics of the SDA and SCL I/O stages

n/a = not applicable.



Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
V_{IL}	LOW-level input voltage[1]		–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage[1]		$0.7V_{DD}$	[2]	$0.7V_{DD}$	[2]	$0.7V_{DD}$[1]	[2]	V
V_{hys}	hysteresis of Schmitt trigger inputs		-	-	$0.05V_{DD}$	-	$0.05V_{DD}$	-	V
V_{OL1}	LOW-level output voltage 1	(open-drain or open-collector) at 3 mA sink current; $V_{DD} > 2$ V	0	0.4	0	0.4	0	0.4	V
V_{OL2}	LOW-level output voltage 2	(open-drain or open-collector) at 2 mA sink current[3]; $V_{DD} \leq 2$ V	-	-	0	$0.2V_{DD}$	0	$0.2V_{DD}$	V
I_{OL}	LOW-level output current	$V_{OL} = 0.4$ V	3	-	3	-	20	-	mA
		$V_{OL} = 0.6$ V[4]	-	-	6	-	-	-	mA
t_{of}	output fall time from V_{IHmin} to V_{ILmax}		-	250[5]	20 × (V_{DD} / 5.5 V)[6]	250[5]	20 × (V_{DD} / 5.5 V)[6]	120[7]	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		-	-	0	50[8]	0	50[8]	ns
I_i	input current each I/O pin	$0.1V_{DD} < V_I < 0.9V_{DDmax}$	–10	+10	–10[9]	+10[9]	–10[9]	+10[9]	µA
C_i	capacitance for each I/O pin[10]		-	10	-	10	-	10	pF

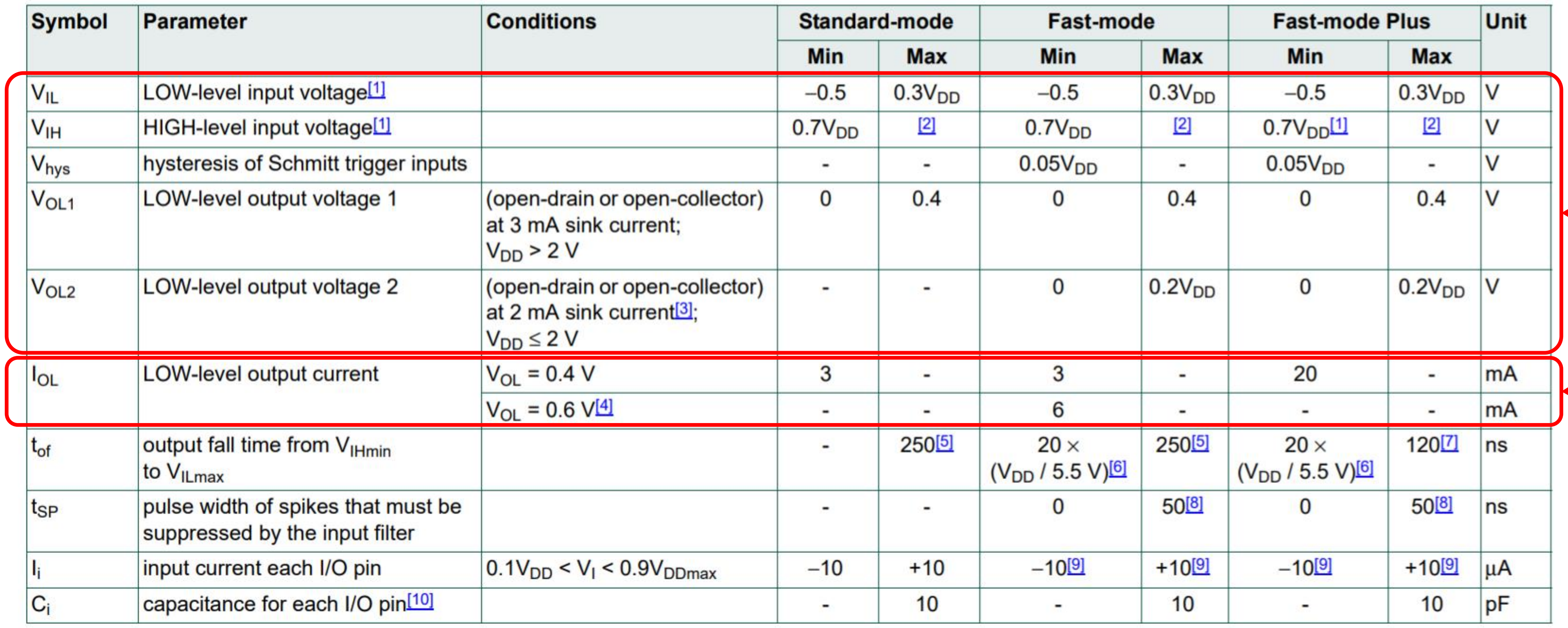
Standard mode Fast mode Fast-mode Plus

Electrical and Timing Specifications

SDA and SCL I/O Characteristics

Table 9. Characteristics of the SDA and SCL I/O stages

n/a = not applicable.



Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
V_{IL}	LOW-level input voltage[1]		–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage[1]		$0.7V_{DD}$	[2]	$0.7V_{DD}$	[2]	$0.7V_{DD}$[1]	[2]	V
V_{hys}	hysteresis of Schmitt trigger inputs		-	-	$0.05V_{DD}$	-	$0.05V_{DD}$	-	V
V_{OL1}	LOW-level output voltage 1	(open-drain or open-collector) at 3 mA sink current; $V_{DD} > 2$ V	0	0.4	0	0.4	0	0.4	V
V_{OL2}	LOW-level output voltage 2	(open-drain or open-collector) at 2 mA sink current[3]; $V_{DD} \leq 2$ V	-	-	0	$0.2V_{DD}$	0	$0.2V_{DD}$	V
I_{OL}	LOW-level output current	$V_{OL} = 0.4$ V	3	-	3	-	20	-	mA
		$V_{OL} = 0.6$ V[4]	-	-	6	-	-	-	mA
t_{of}	output fall time from V_{IHmin} to V_{ILmax}		-	250[5]	$20 \times (V_{DD} / 5.5\ V)$[6]	250[5]	$20 \times (V_{DD} / 5.5\ V)$[6]	120[7]	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		-	-	0	50[8]	0	50[8]	ns
I_i	input current each I/O pin	$0.1V_{DD} < V_I < 0.9V_{DDmax}$	–10	+10	–10[9]	+10[9]	–10[9]	+10[9]	µA
C_i	capacitance for each I/O pin[10]		-	10	-	10	-	10	pF

Input and output voltage levels

Output current

Electrical and Timing Specifications

SDA and SCL Bus Line Characteristics

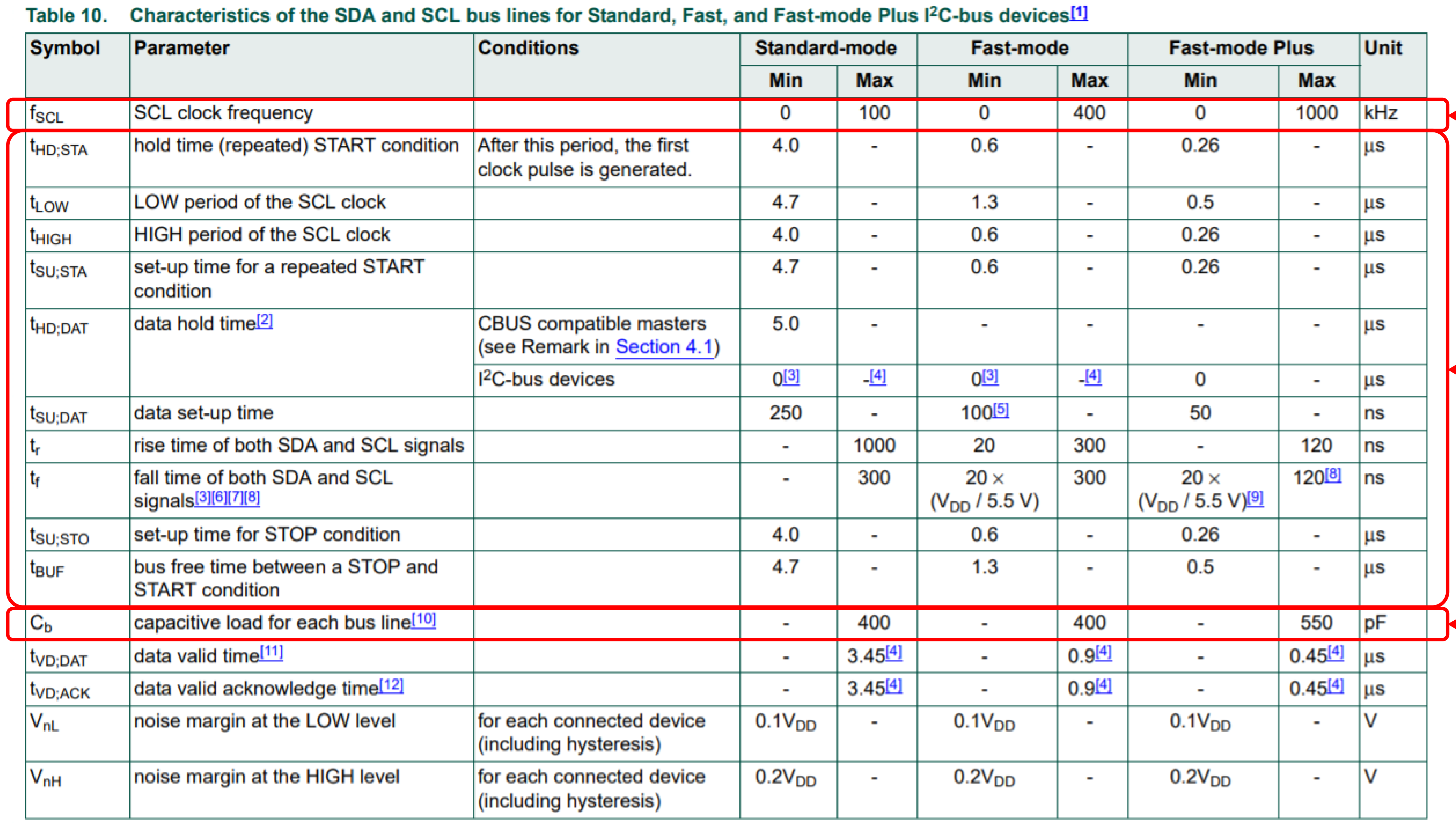


Table 10. Characteristics of the SDA and SCL bus lines for Standard, Fast, and Fast-mode Plus I^2C-bus devices[1]

Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
f_{SCL}	SCL clock frequency		0	100	0	400	0	1000	kHz
$t_{HD;STA}$	hold time (repeated) START condition	After this period, the first clock pulse is generated.	4.0	-	0.6	-	0.26	-	µs
t_{LOW}	LOW period of the SCL clock		4.7	-	1.3	-	0.5	-	µs
t_{HIGH}	HIGH period of the SCL clock		4.0	-	0.6	-	0.26	-	µs
$t_{SU;STA}$	set-up time for a repeated START condition		4.7	-	0.6	-	0.26	-	µs
$t_{HD;DAT}$	data hold time[2]	CBUS compatible masters (see Remark in Section 4.1)	5.0	-	-	-	-	-	µs
		I^2C-bus devices	0[3]	-[4]	0[3]	-[4]	0	-	µs
$t_{SU;DAT}$	data set-up time		250	-	100[5]	-	50	-	ns
t_r	rise time of both SDA and SCL signals		-	1000	20	300	-	120	ns
t_f	fall time of both SDA and SCL signals[3][6][7][8]		-	300	20 × (V_{DD} / 5.5 V)	300	20 × (V_{DD} / 5.5 V)[9]	120[8]	ns
$t_{SU;STO}$	set-up time for STOP condition		4.0	-	0.6	-	0.26	-	µs
t_{BUF}	bus free time between a STOP and START condition		4.7	-	1.3	-	0.5	-	µs
C_b	capacitive load for each bus line[10]		-	400	-	400	-	550	pF
$t_{VD;DAT}$	data valid time[11]		-	3.45[4]	-	0.9[4]	-	0.45[4]	µs
$t_{VD;ACK}$	data valid acknowledge time[12]		-	3.45[4]	-	0.9[4]	-	0.45[4]	µs
V_{nL}	noise margin at the LOW level	for each connected device (including hysteresis)	$0.1V_{DD}$	-	$0.1V_{DD}$	-	$0.1V_{DD}$	-	V
V_{nH}	noise margin at the HIGH level	for each connected device (including hysteresis)	$0.2V_{DD}$	-	$0.2V_{DD}$	-	$0.2V_{DD}$	-	V

Maximum SCL frequency

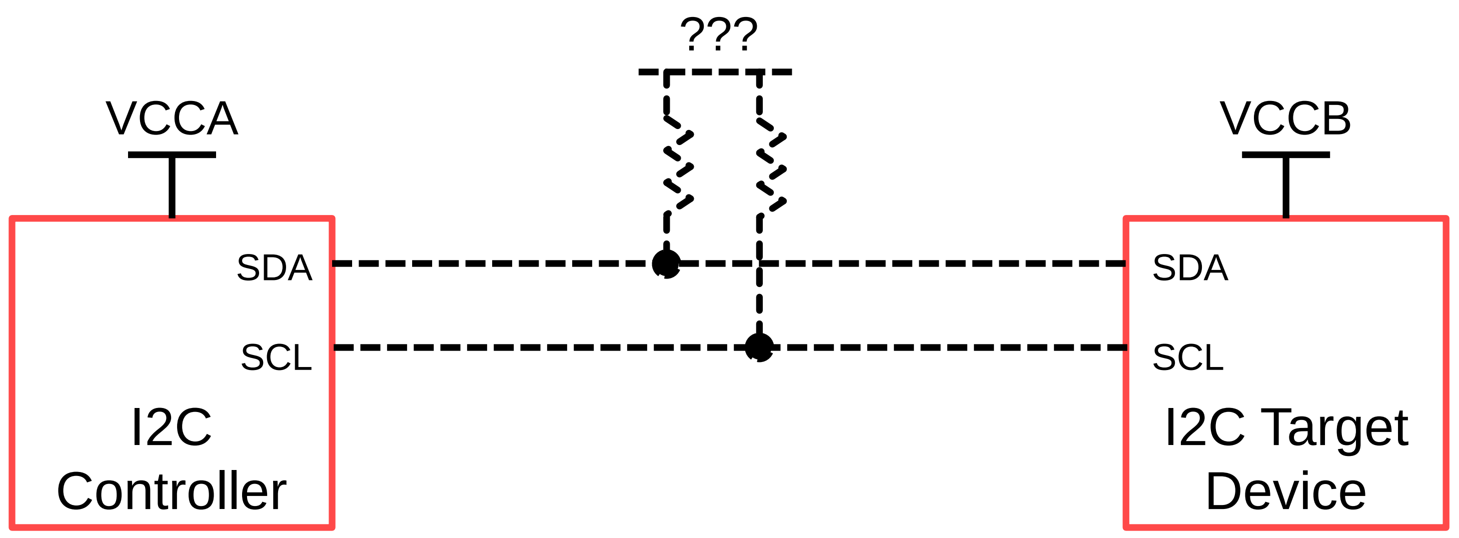
Bus timing, including setup and hold times

Maximum bus capacitive load

Basics of I2C – Advanced Topics

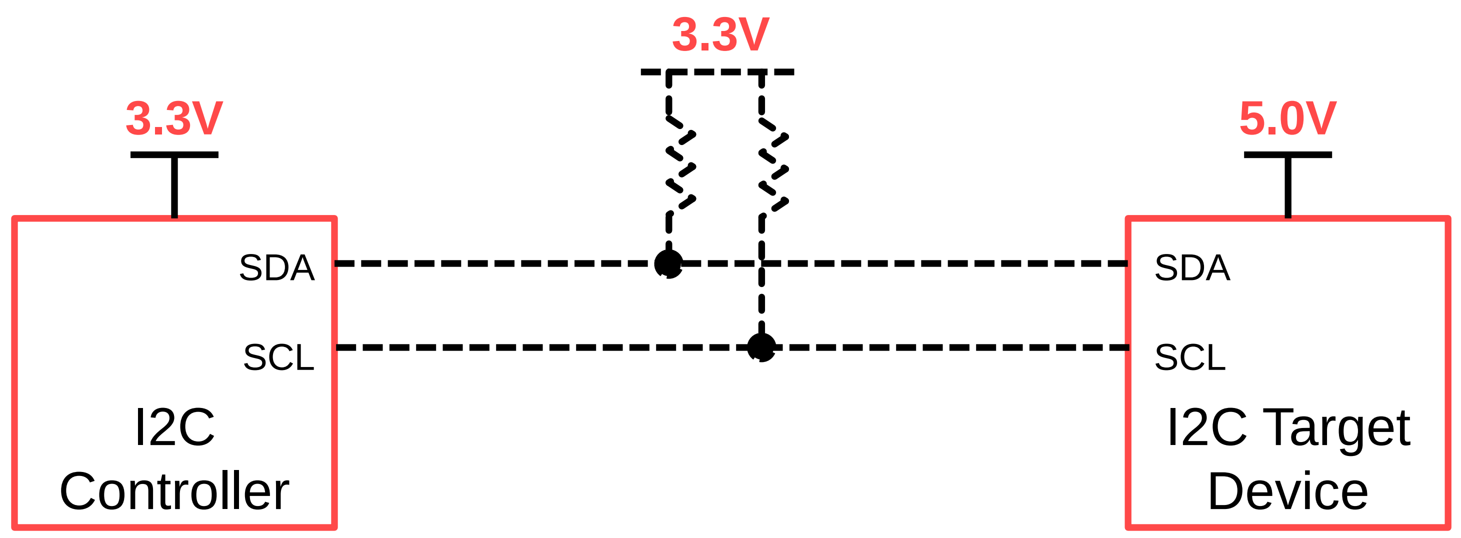
- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- **Voltage Level Translation**
- Pullup Resistor Sizing
- Other Similar Protocols

Voltage Level Translation

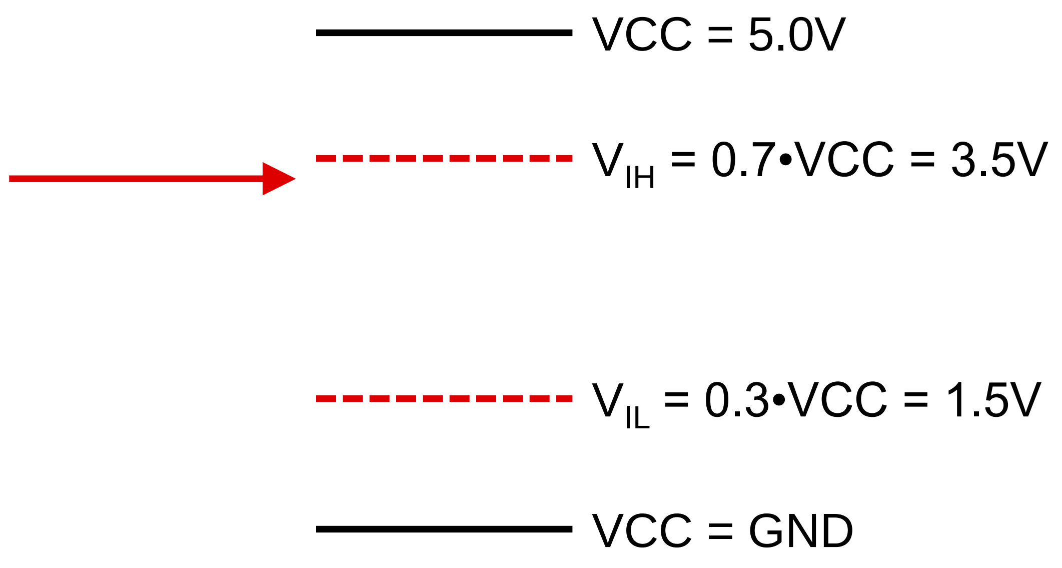


- Mismatched controller and target device voltages may cause problems in communication or even damage to devices
- The pullup resistor connection determines if the output voltage mismatch overdrives the input or under-drives the input to the other device

Voltage Level Translation



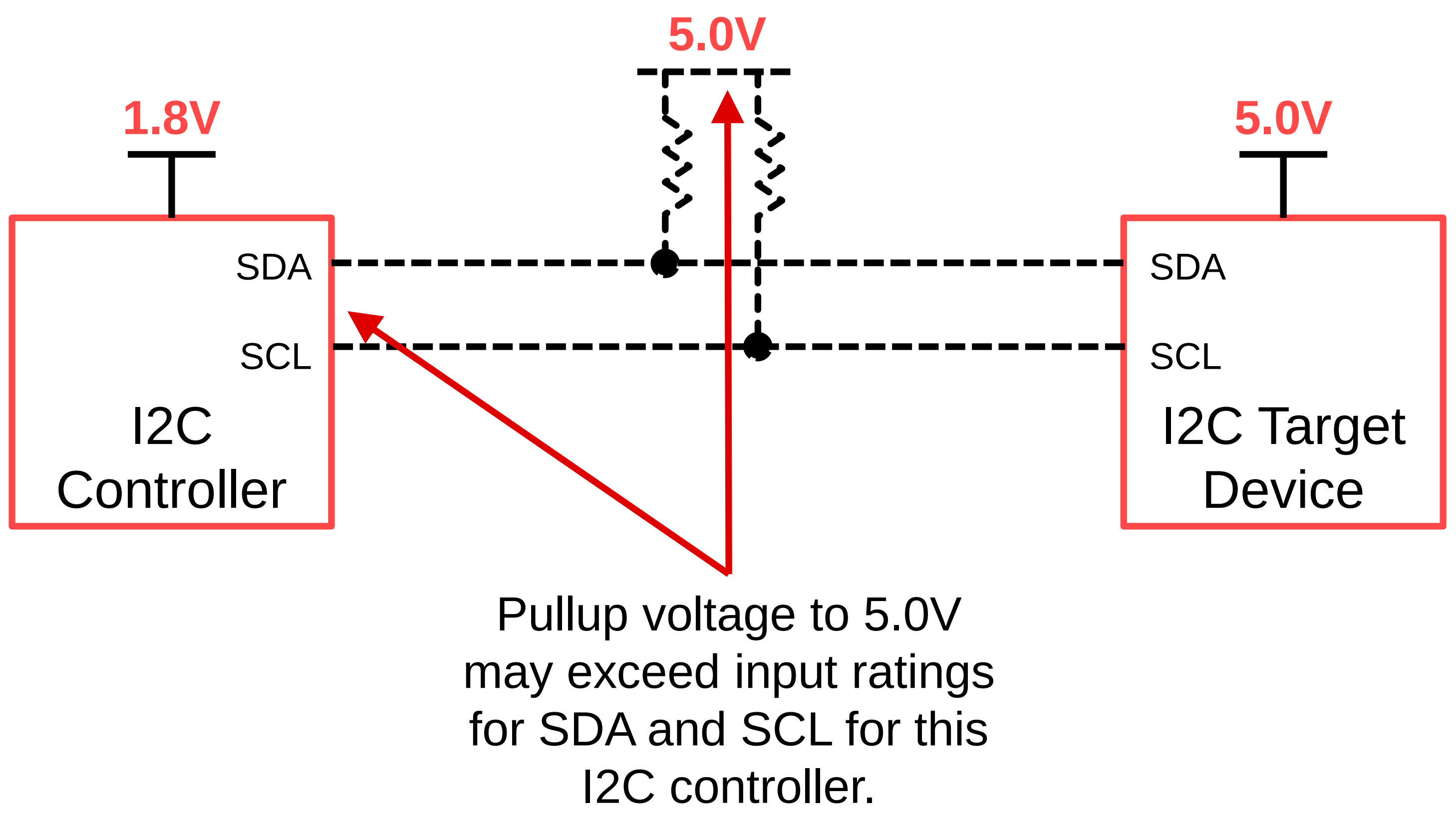
I2C input voltage level



At 3.3V, controller I2C bus can't be pulled up high enough to meet the input high voltage minimum (V_{IL}) for the target device

- Controller and pullups set to 3.3V
- Target device set to 5.0V
- A high-level input voltage is defined as 0.7 ▪ VCC
- For this example, the pullup only goes up to 3.3V, while the target device high-level input voltage is 3.5V based on it's supply
- Input does not go high enough to ensure correct communication

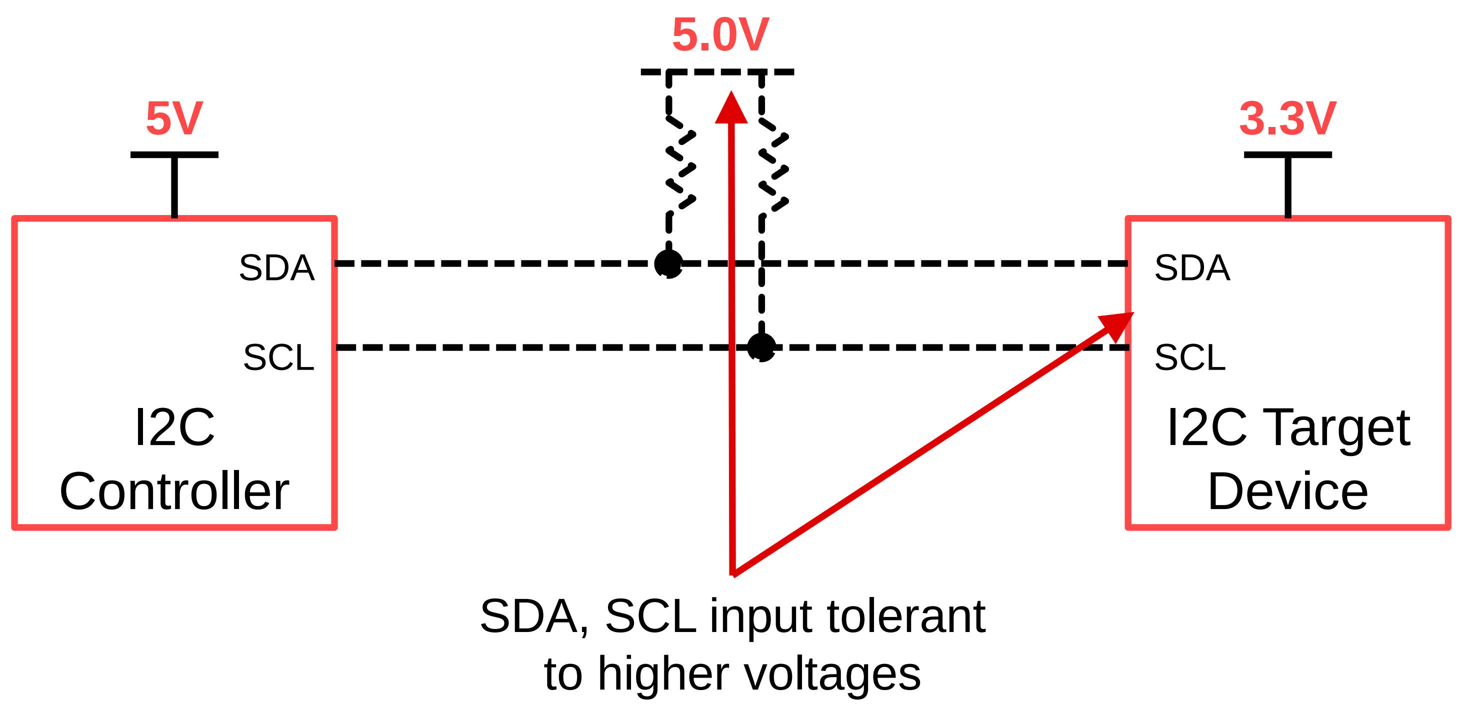
Voltage Level Translation



5.0V
1.8V
5.0V
SDA
SCL
I2C Controller
SDA
SCL
I2C Target Device
Pullup voltage to 5.0V may exceed input ratings for SDA and SCL for this I2C controller.

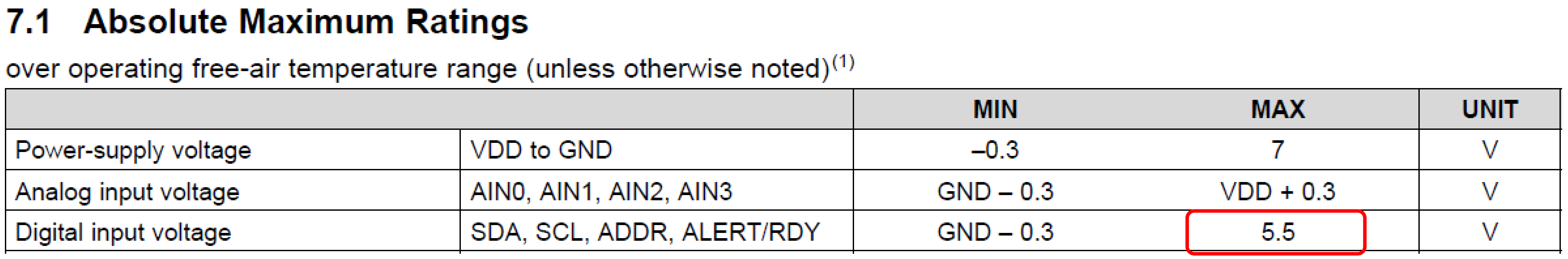
- Controller set to 1.8V
- Target device and pullups set to 5.0V
- High-level bus voltage pulls up to 5.0V, which may be outside the input range of the SDA and SCL of the controller

Voltage Level Translation



- Controller and pullups set to 5V
- Target device set to 3.3V
- In this example, the target device has SDA and SCL pins that are tolerant to higher voltages
- This communication is ok, even if the target device SDA and SCL pins are raised higher than supply

Example from ADS1115:

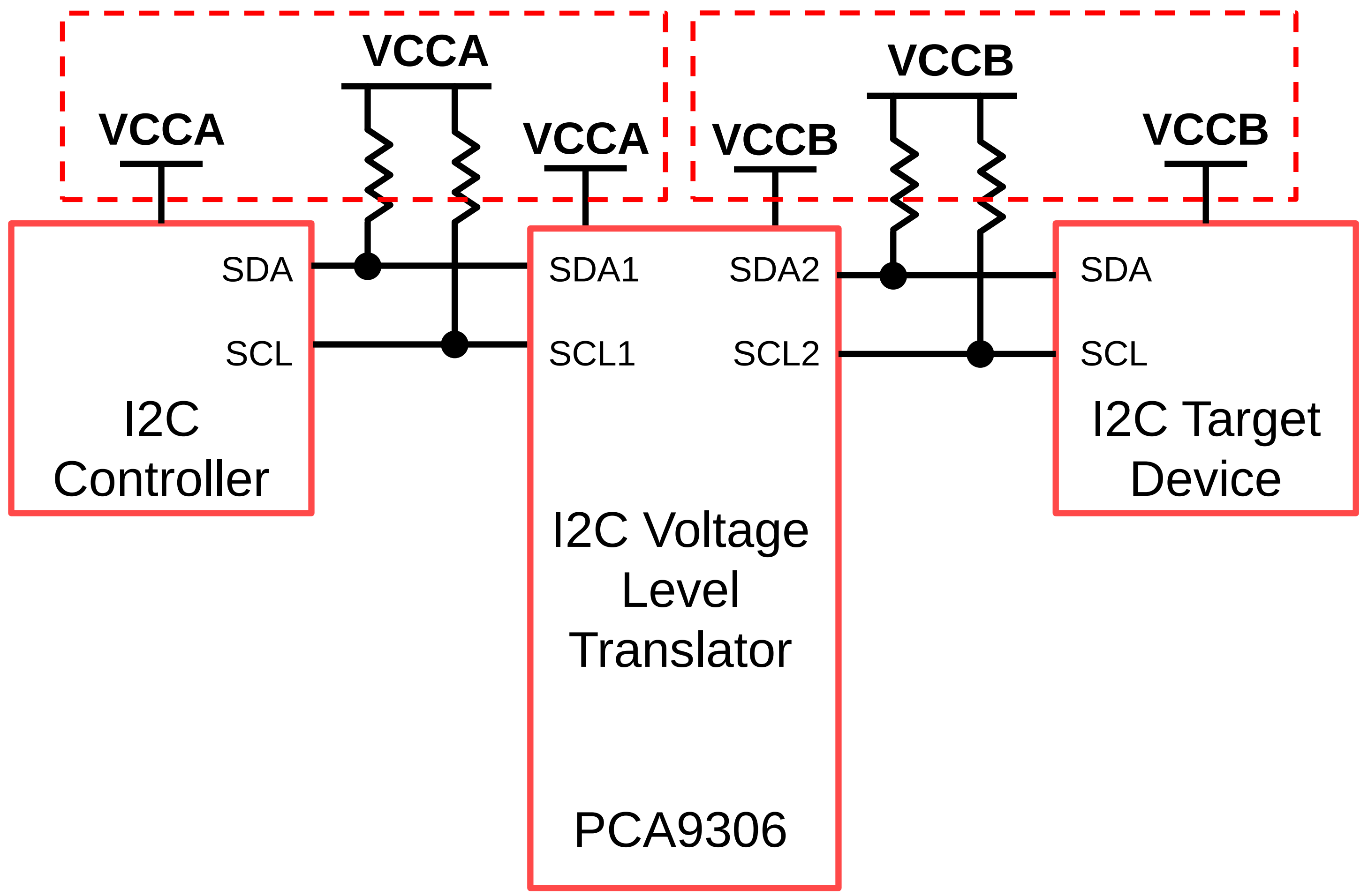


7.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)(1)

		MIN	MAX	UNIT
Power-supply voltage	VDD to GND	–0.3	7	V
Analog input voltage	AIN0, AIN1, AIN2, AIN3	GND – 0.3	VDD + 0.3	V
Digital input voltage	SDA, SCL, ADDR, ALERT/RDY	GND – 0.3	5.5	V

Voltage Level Translation

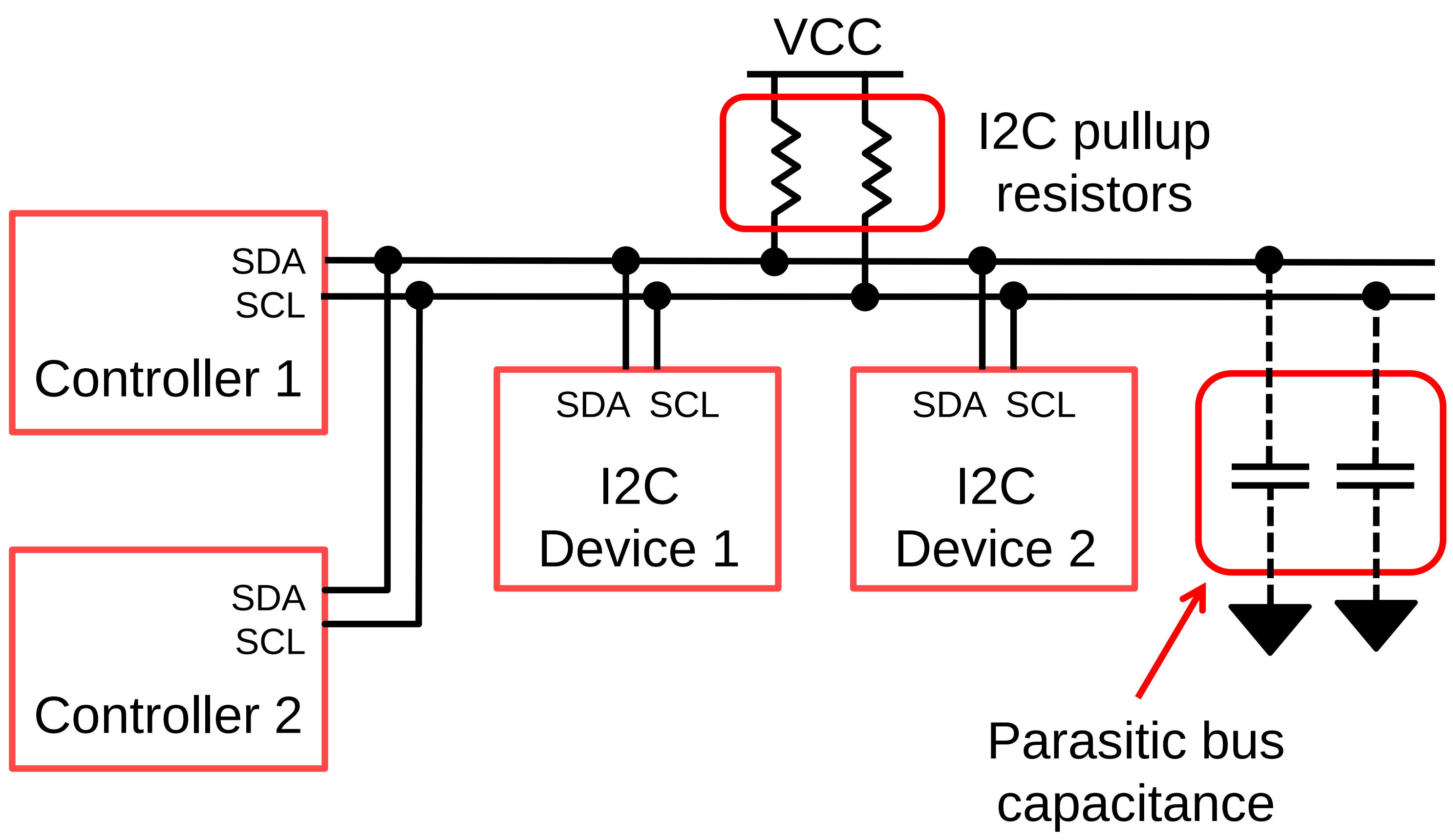


- Voltage level translators are a solution to mismatched supplies
- Requires two sets of pullups, one for each voltage level
- PCA9306 is a common I2C voltage level translator

Basics of I2C – Advanced Topics

- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- **Pullup Resistor Sizing**
- Other Similar Protocols

Pullup Resistor Sizing



- Transition time for I2C determined by pullup resistor sizing, total bus capacitance, and current sink from I2C devices
- Typical pullup resistors 1kΩ to 10kΩ
- Lower resistance pullup resistance means higher power
- Higher pullup resistance means lower speed
- Min and max pullup resistance can be calculated for optimal performance

Pullup Resistor Sizing

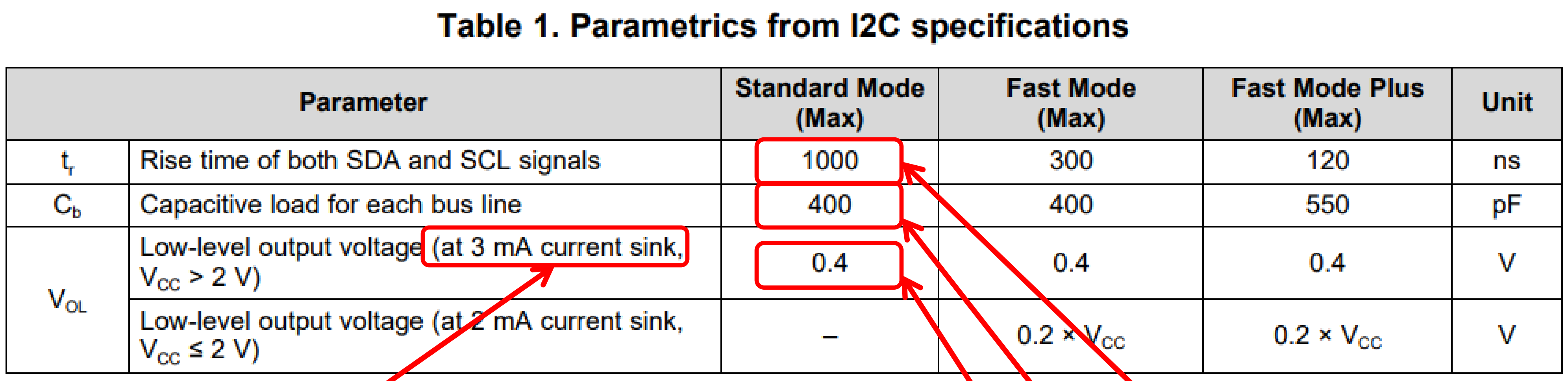


Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \leq 2$ V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V

Current sink for device I2C connections (I_{OL})

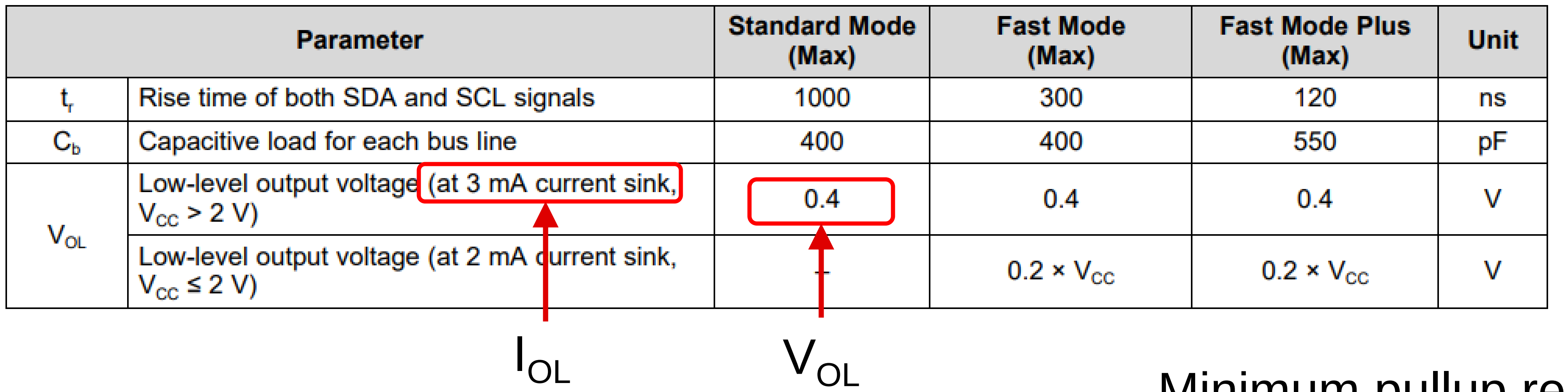
Maximum rise time for I2C signals

Maximum capacitive load on bus

Digital output voltage low level (V_{OL})

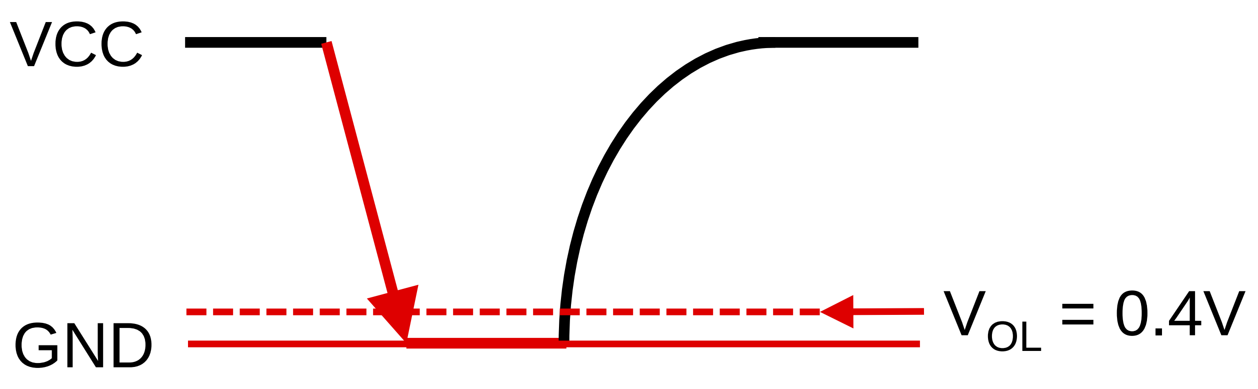
Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications



Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, V_{CC} ≤ 2 V)	–	0.2 × V_{CC}	0.2 × V_{CC}	V

SDA, SCL Voltage



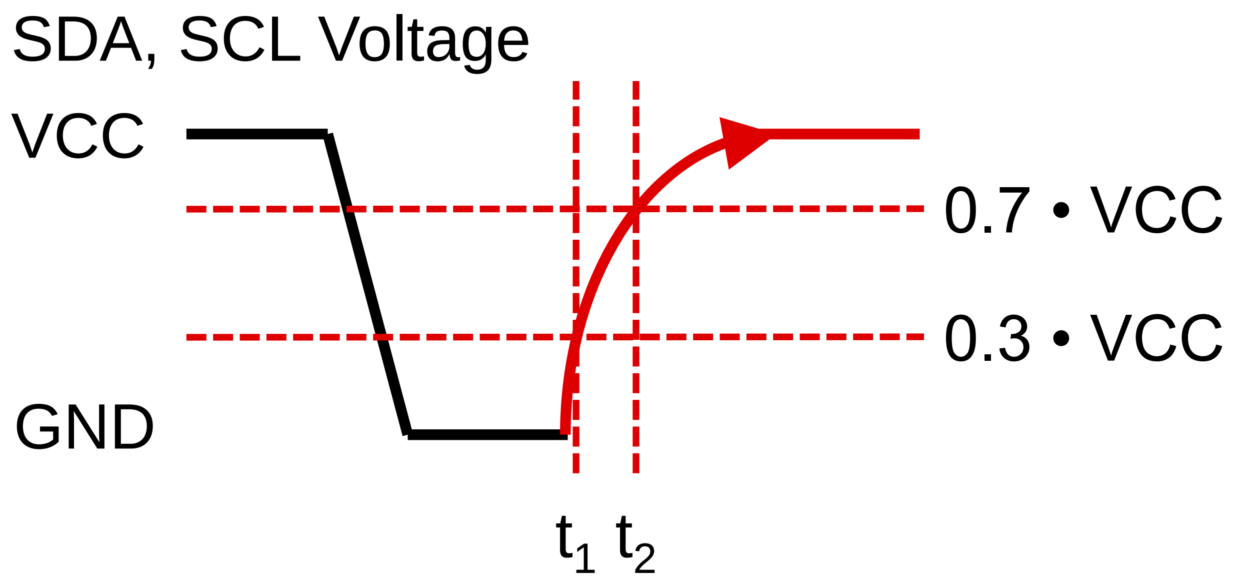
Minimum pullup resistance based on VCC to V_{OL} and the pulldown current I_{OL}

$$R_P(\min) = \frac{(V_{CC} - V_{OL}(\max))}{I_{OL}}$$

Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, $V_{CC} > 2$ V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \leq 2$ V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V



Maximum pullup resistance based on the exponential voltage rise:

$$V(t) = V_{CC} \times \left(1 - e^{\frac{-t}{RC}}\right)$$

From the digital input low voltage:

$$V_{IL} = 0.3 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t_1}{R_P C_B}}\right)$$

To the digital input high voltage:

$$V_{IH} = 0.7 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t_2}{R_P C_B}}\right)$$

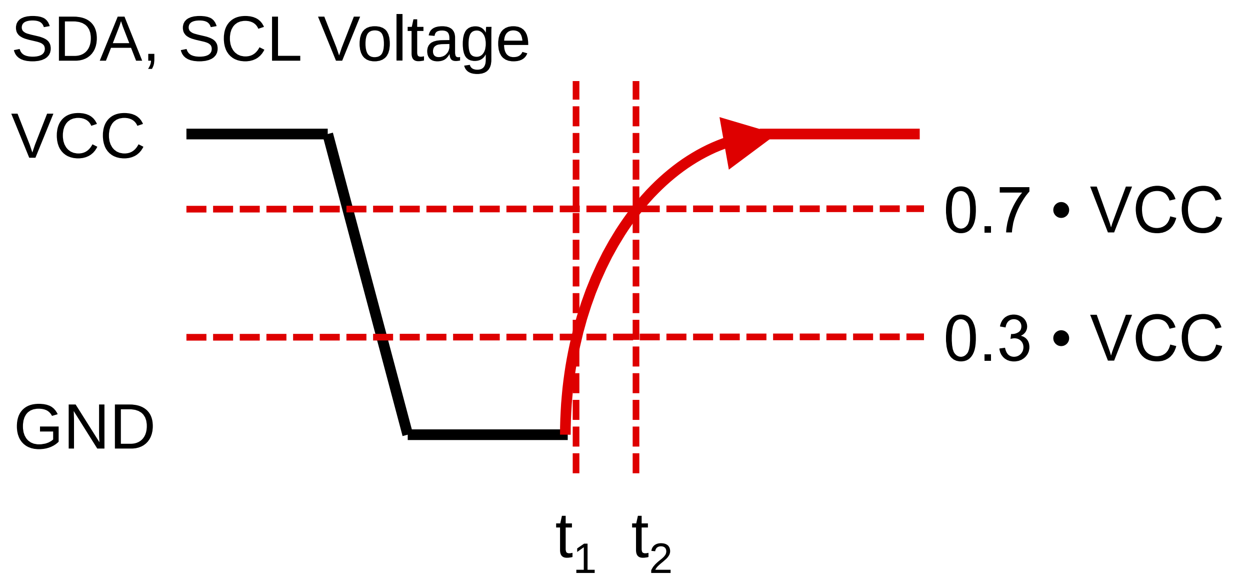
Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, V_{CC} ≤ 2 V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V

Max t_{RISE}

Max bus capacitance



Solve for the rise time ($t_2 - t_1$), can be calculated as:

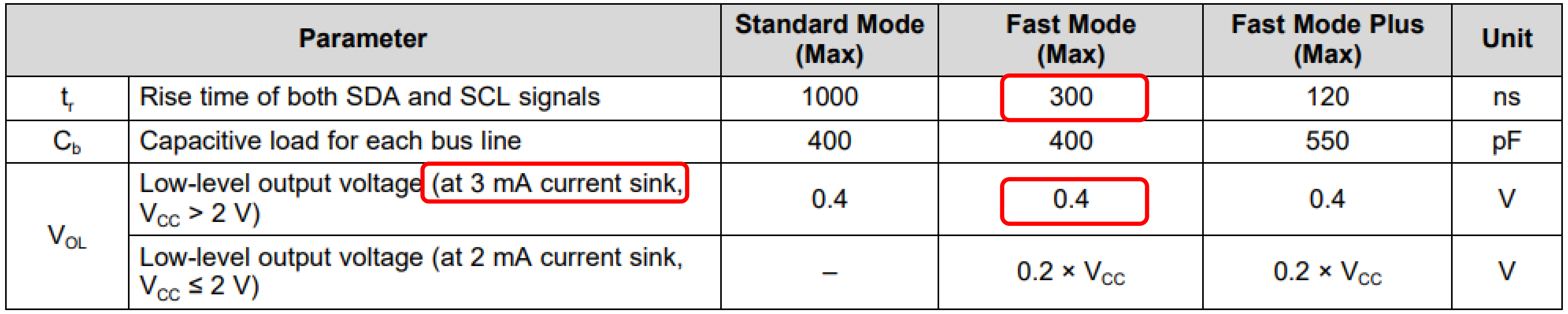
$$t_{RISE} = t_2 - t_1 = 0.8473 \bullet R_p \bullet C_b$$

The maximum pullup resistance can be calculated from the maximum rise time and the bus capacitance:

$$R_P(\max) = \frac{t_{RISE}}{(0.8473 \bullet C_b)}$$

Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications



Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, V_{CC} ≤ 2 V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V

Example:

Find the minimum and maximum pullup resistance for Fast mode I2C Communication with:

- Cb = 200 pF and VCC = 3.3V

Solution:

$$R_P(\text{min}) = \frac{(V_{CC} - V_{OL}(\text{max}))}{I_{OL}} = \frac{(3.3V - 0.4V)}{3 \times 10^{-3}A} = 966.667\Omega$$

$$R_P(\text{max}) = \frac{t_{RISE}}{(0.8473 \bullet C_b)} = \frac{300 \times 10^{-9}s}{0.8473 \bullet 200 \times 10^{-12}F} = 1.77k\Omega$$

From “I2C Bus Pullup Resistor Calculation”
By Rajan Arora (Application Report SLVA689)

TEXAS INSTRUMENTS

Basics of I2C – Advanced Topics

- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- **Other Similar Protocols**

Other Similar Protocols

SMBus - System Management Bus

- Commonly used in computer motherboards for power source management
- Some minor specification differences with I2C in voltage levels, sink current, frequency, and timing
- Added features:
  - ✓ Dynamic address allocations
  - ✓ 35ms timeout to the bus
  - ✓ Packet error checking (PEC) with CRC-8 checksum
  - ✓ SMBAlert#

Other Similar Protocols

PMBus - Power Management Bus

- Variant of SMBus
- Originally defined by Intel and Duracell
- Used in digital management of power supplies
- Many pre-defined domain specific commands
- Standard command space has many readable and writeable device attributes regarding voltage current, and power; gives measurements, status, and warnings

Other Similar Protocols

IPMI – Intelligent Platform Management Interface

- Standardized message-based hardware management interface for a computer motherboard or server
- Run by a Baseboard Management Controller (BMC) or Management Controller (MC)
- Independent of computer's CPU, firmware, and operating system.
- BMC is always running IPMI even when the main system is off, allows for remote management of a system

Other Similar Protocols

ATCA - Advanced Telecom Computing Architecture

- Follow-on to Compact PCI (cPCI), used in advanced rack-mounted telecommunications hardware
- Includes fault tolerant scheme for thermal management

DDC - Display Data Channel

- Allows for monitor or display to inform the host about identity and capabilities
- Bidirectional host may control monitor display functions

CBUS compatibility

- Compatible with reserved I2C address and bus line DLEN
- No longer used

Thanks for your time!
Please try the quiz.

Quiz: Basics of I2C: Advanced Topics

1. To prevent bus contention between controllers, I2C uses clock synchronization and then arbitration. How is arbitration done between controllers?
   a. The first controller to send the START condition is allowed to continue it’s communication
   b. The first device to complete the address byte and receive the ACK from the target device continues its communication
   c. Clock synchronization decides the controller that wins the arbitration
   d. Contending controllers synchronize clocks and continue communication. The first controller to send an SDA low bit that is not matched by the other controller wins arbitration

Quiz: Basics of I2C: Advanced Topics

1. To prevent bus contention between controllers, I2C uses clock synchronization and then arbitration. How is arbitration done between controllers?
   a. The first controller to send the START condition is allowed to continue it’s communication
   b. The first device to complete the address byte and receive the ACK from the target device continues its communication
   c. Clock synchronization decides the controller that wins the arbitration
   [✓ d. Contending controllers synchronize clocks and continue communication. The first controller to send an SDA low bit that is not matched by the other controller wins arbitration]

Quiz: Basics of I2C: Advanced Topics

2. The PCA9306 is an example of what?
   a. A voltage translator used to bridge devices with two different supply voltages
   b. A system for clock stretching during the ACK from target device after the controller sends the I2C address
   c. A communication protocol similar to I2C used for power management

Quiz: Basics of I2C: Advanced Topics

2. The PCA9306 is an example of what?
   - [✓ a. A voltage translator used to bridge devices with two different supply voltages]
   - b. A system for clock stretching during the ACK from target device after the controller sends the I2C address
   - c. A communication protocol similar to I2C used for power management

Quiz: Basics of I2C: Advanced Topics

3. Which of the following parameters is not a factor in calculating the minimum and maximum values for I2C pullup resistors?
    a. The I2C supply voltage attached to the pullup resistor
    b. The total bus capacitance on the I2C line
    c. Setup and hold times for SDA and SCL
    d. The I2C device sink current for SDA and SCL
    e. The digital input high and digital input low levels for the SDA and SCL lines

Quiz: Basics of I2C: Advanced Topics

3. Which of the following parameters is not a factor in calculating the minimum and maximum values for I2C pullup resistors?
   a. The I2C supply voltage attached to the pullup resistor
   b. The total bus capacitance on the I2C line
   [✓ c. Setup and hold times for SDA and SCL]
   d. The I2C device sink current for SDA and SCL
   e. The digital input high and digital input low levels for the SDA and SCL lines

Thanks for your time!

Basics of I2C: Advanced Topics

TIPL 6104
TI Precision Labs – Digital Communications

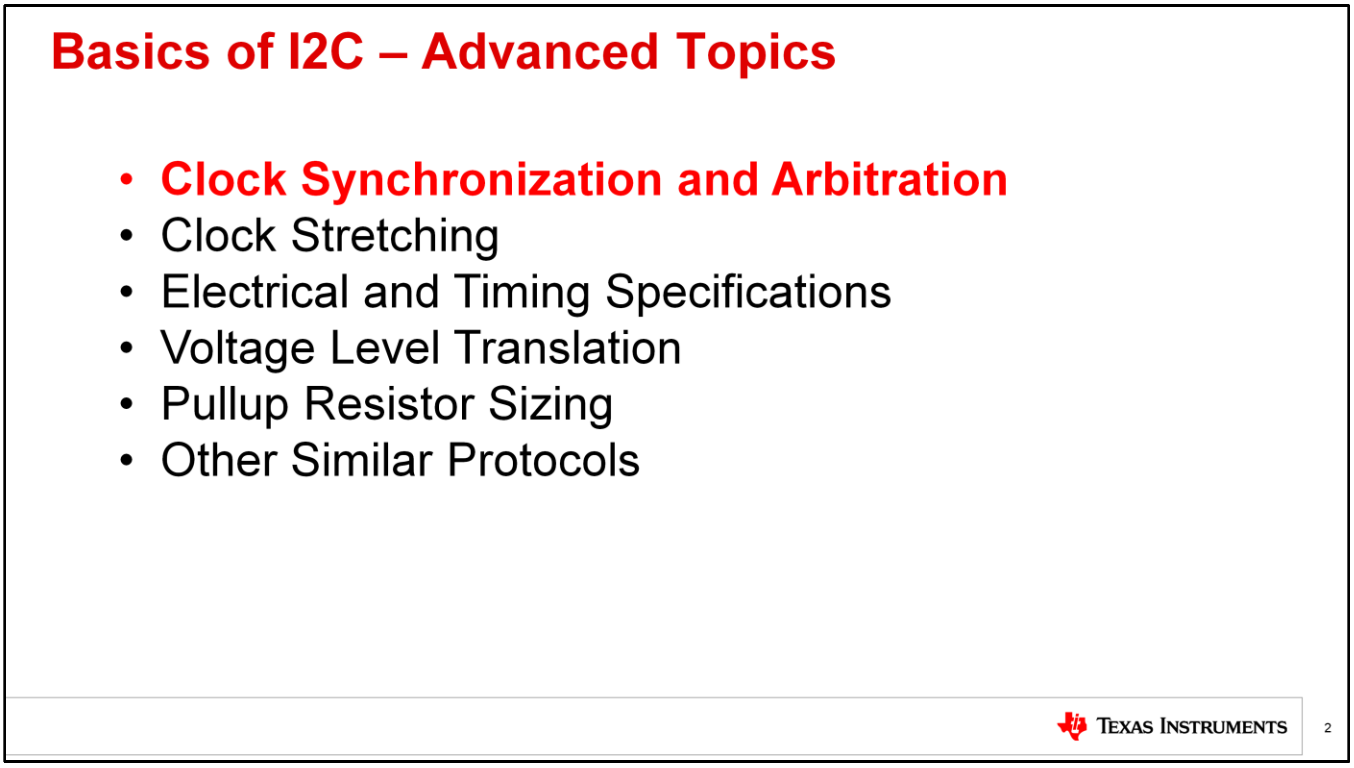
Prepared by Joseph Wu

Presented by Alex Smith

Texas Instruments 1

Hello, and welcome to our in-depth look at communications with precision data converters. In other I2C videos, we describe the some of the protocol basics of I2C and use an example to show how you might communicate with a precision data converter. With those videos, you should be able to understand how I2C works and how to read and debug basic system communications.

However, those videos only scratch the surface of the I2C protocol. This video will cover some advanced topics of I2C. We won’t go into too much depth. However, we’ll introduce some topics that will allow you to understand what they are when you come across them. This information can be found in the I2C bus specification, and you can find more details there.

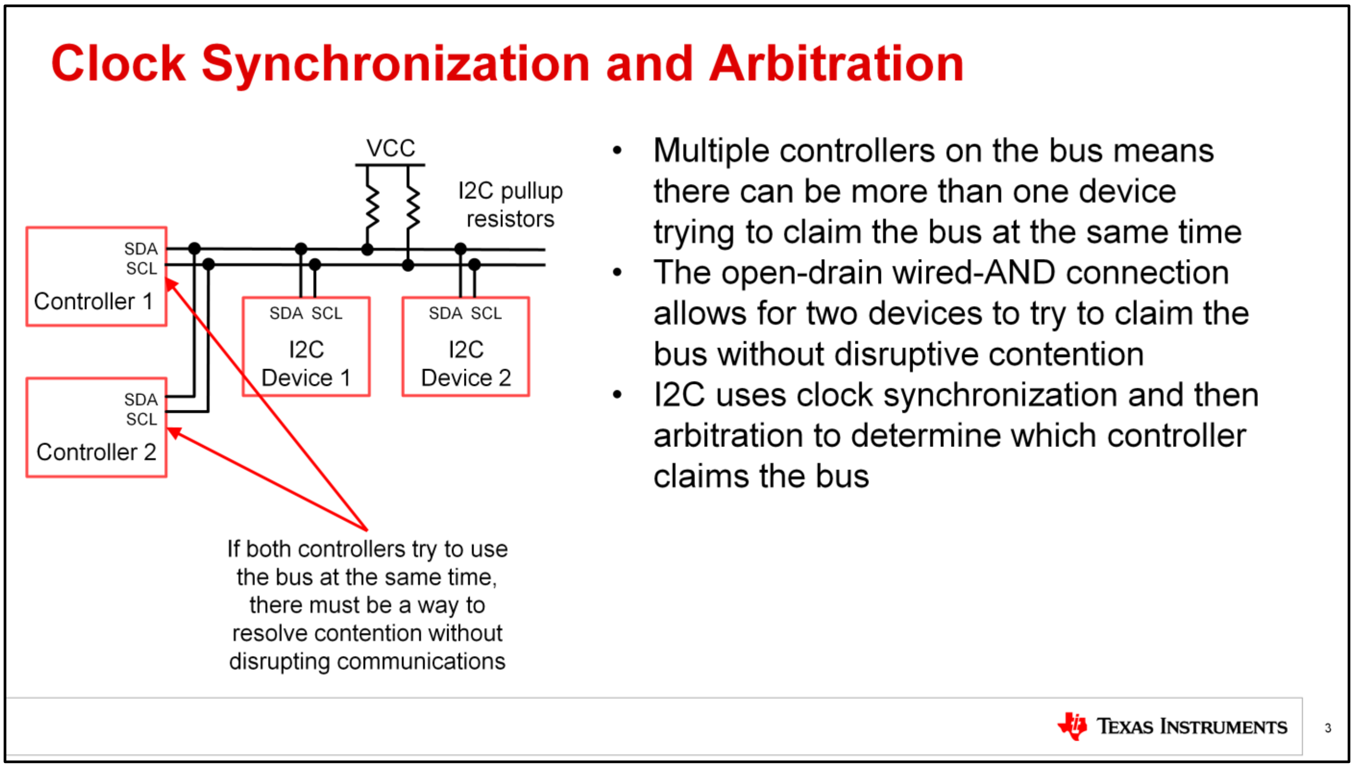


Basics of I2C – Advanced Topics

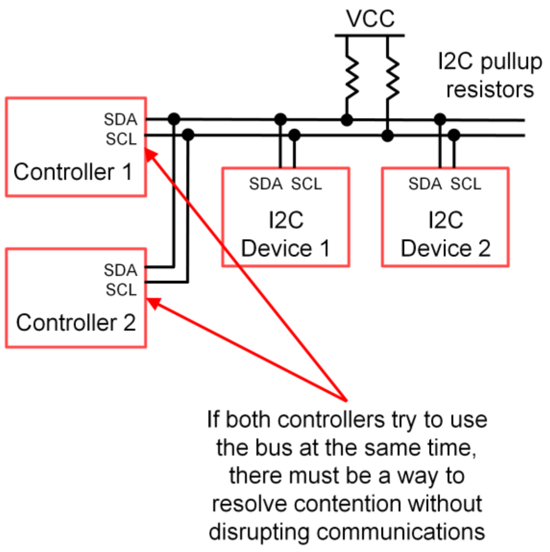
- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

Texas Instruments 2

The first I2C topic in this presentation is clock synchronization and arbitration between controller devices on the bus.



Clock Synchronization and Arbitration


VCC
I2C pullup resistors
SDA
SCL
Controller 1
SDA
SCL
Controller 2
SDA SCL
I2C Device 1
SDA SCL
I2C Device 2
If both controllers try to use the bus at the same time, there must be a way to resolve contention without disrupting communications

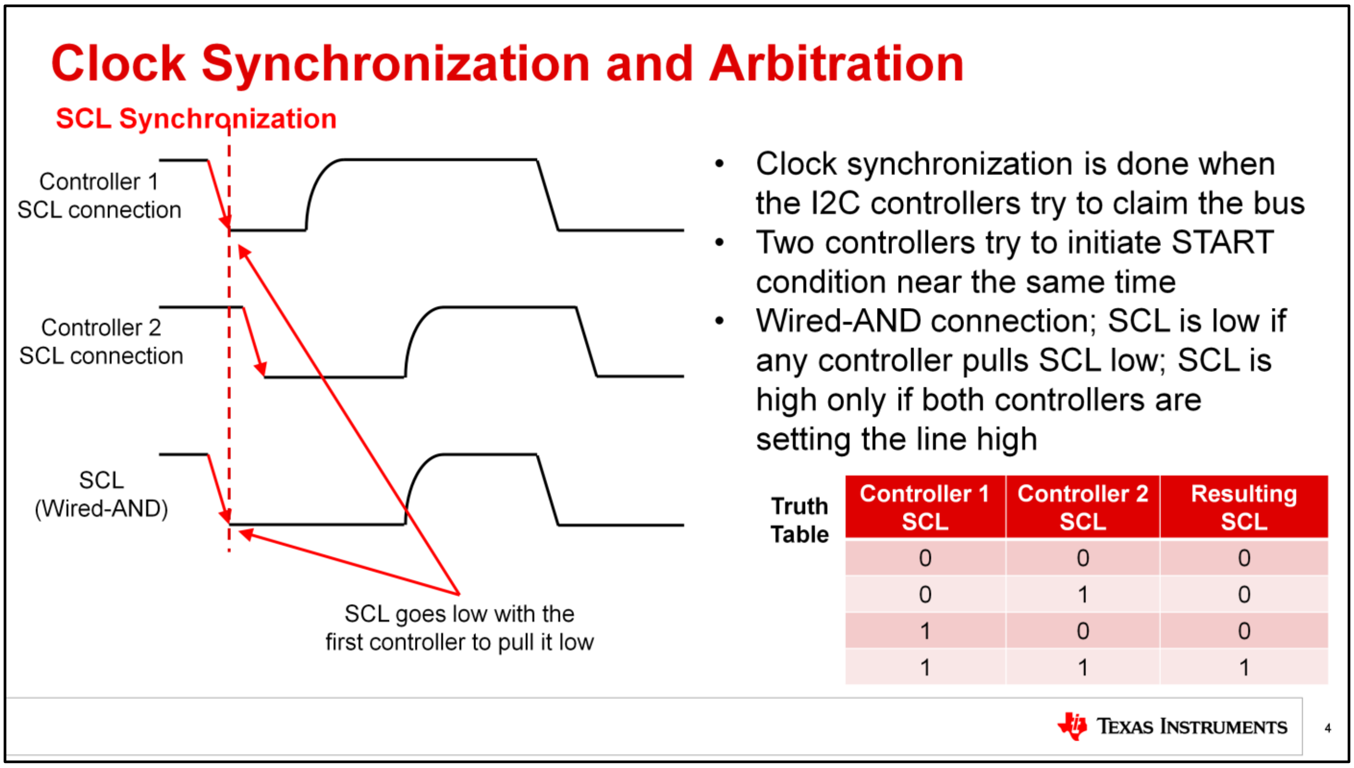
- Multiple controllers on the bus means there can be more than one device trying to claim the bus at the same time
- The open-drain wired-AND connection allows for two devices to try to claim the bus without disruptive contention
- I2C uses clock synchronization and then arbitration to determine which controller claims the bus

TEXAS INSTRUMENTS 3

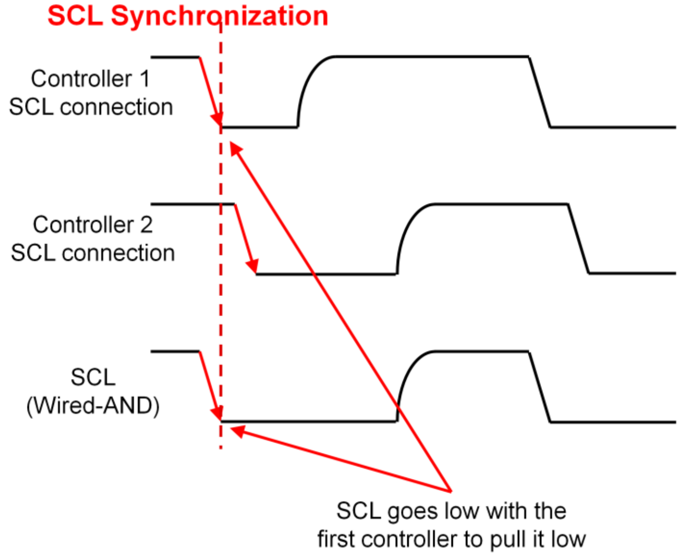
In I2C, there may be multiple controllers on the same bus. Because of this, there may be two or more devices trying to claim the bus for communication at the same time. This requires multiple active controllers to resolve which device controls the bus.

I2C uses a method of clock synchronization and arbitration to ensure that one controller that gains control and does so without compromising its communication. Because I2C uses open-drain connections to SDA and SCL, the connections result in a "wired-AND" connection, where the line gives a logical AND of the device outputs. This is helpful in arbitration without disruption to the communication. In systems with only one controller, this arbitration isn't necessary.

We'll describe in detail how multiple controllers synchronize clocks for I2C communication. We'll also describe how controllers use arbitration to determine which controller wins the bus without disruptive contention.



Clock Synchronization and Arbitration


SCL Synchronization
Controller 1 SCL connection
Controller 2 SCL connection
SCL (Wired-AND)
SCL goes low with the first controller to pull it low

- Clock synchronization is done when the I2C controllers try to claim the bus
- Two controllers try to initiate START condition near the same time
- Wired-AND connection; SCL is low if any controller pulls SCL low; SCL is high only if both controllers are setting the line high

Truth Table

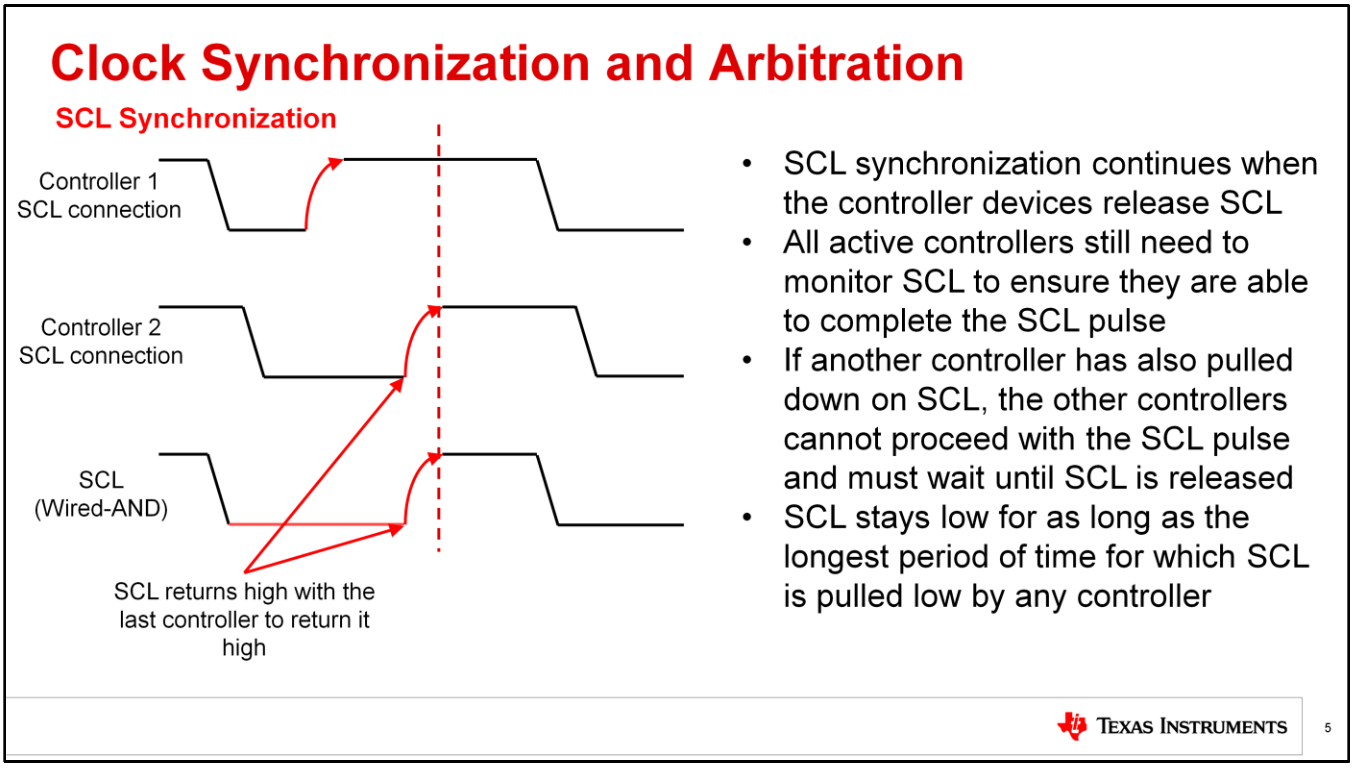
Controller 1 SCL	Controller 2 SCL	Resulting SCL
0	0	0
0	1	0
1	0	0
1	1	1

TEXAS INSTRUMENTS 4

To prevent bus contention, clock synchronization is first performed using the SCL line and the open-drain connections from the controllers on the bus. This wired-AND connection is low if any of the controllers pull SCL low. This connection is the logical AND of the two controller device's SCL connection. The output of SCL is high only if both controller devices have released the open-drain connection high. A truth table of this logical wired-AND is shown in the lower right corner.

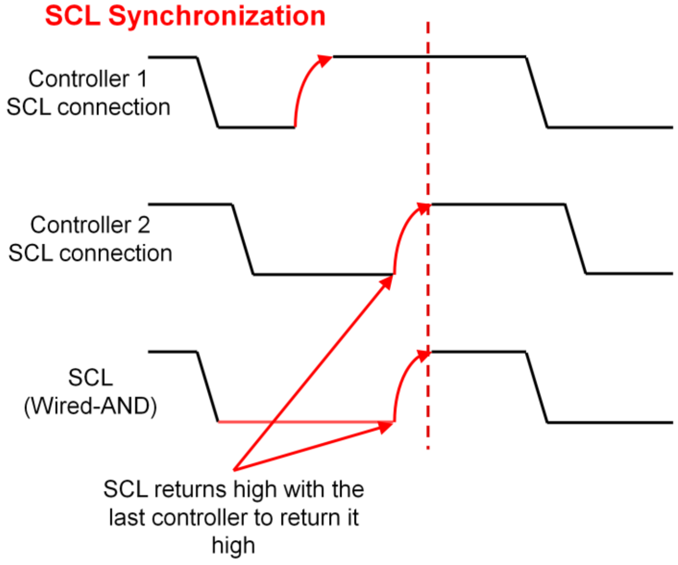
During a START condition where two controllers are trying to claim the bus, there is a high to low transition on SCL. Here is an example where two controller devices are trying to claim the bus at or near the same time.

Here, controller 1 device initiates a START condition shortly before controller device 2 does the same. controller 1 pulls SCL down before controller 2. With the wired-AND connection, SCL pulls low as soon as controller 1 pulls down on SCL.



Clock Synchronization and Arbitration

SCL Synchronization



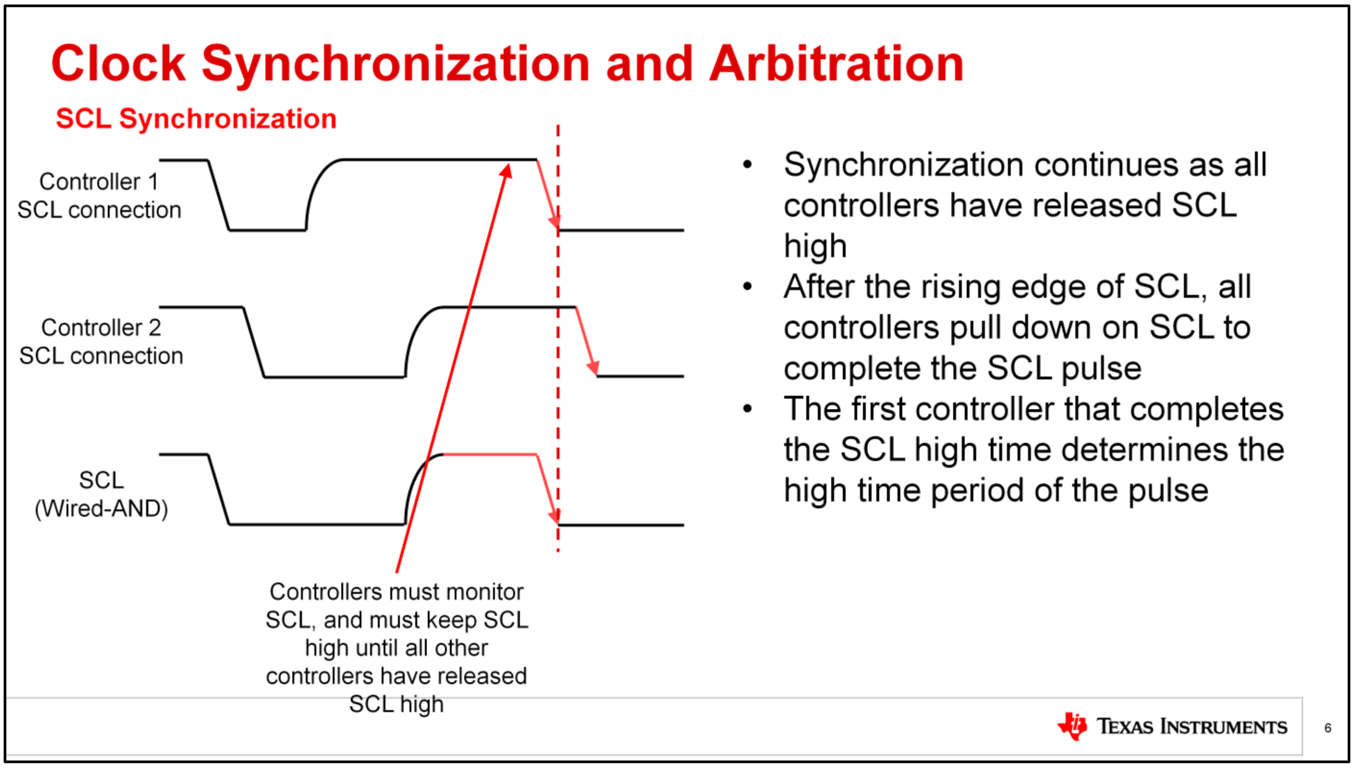
- SCL synchronization continues when the controller devices release SCL
- All active controllers still need to monitor SCL to ensure they are able to complete the SCL pulse
- If another controller has also pulled down on SCL, the other controllers cannot proceed with the SCL pulse and must wait until SCL is released
- SCL stays low for as long as the longest period of time for which SCL is pulled low by any controller

TEXAS INSTRUMENTS 5

After the START condition, controller 1 releases SCL to go high. However controller 2 is still holding SCL low. Because of the wired-AND connection, SCL remains low until controller 2 releases the SCL high. At the same time, controller 1 is still monitoring SCL and must wait for the other controller to release the clock. controller 1 cannot advance the SCL pulse until the SCL is available when controller 2 has released it.

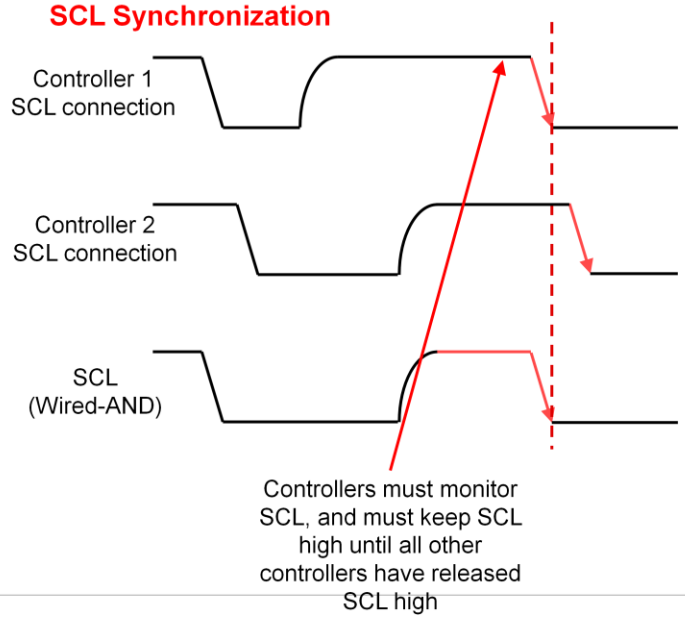
When multiple controllers are competing for the bus, SCL stays low for as long as the longest period of time that any controller pulls down SCL. Only after all the controllers have released the SCL can the line be released high for the serial clock pulse. This synchronizes the start of the serial clock for all controllers.

For clock synchronization, each controller device must monitor the SCL line and react to cases where the SCL does not match its expected SCL output.



Clock Synchronization and Arbitration

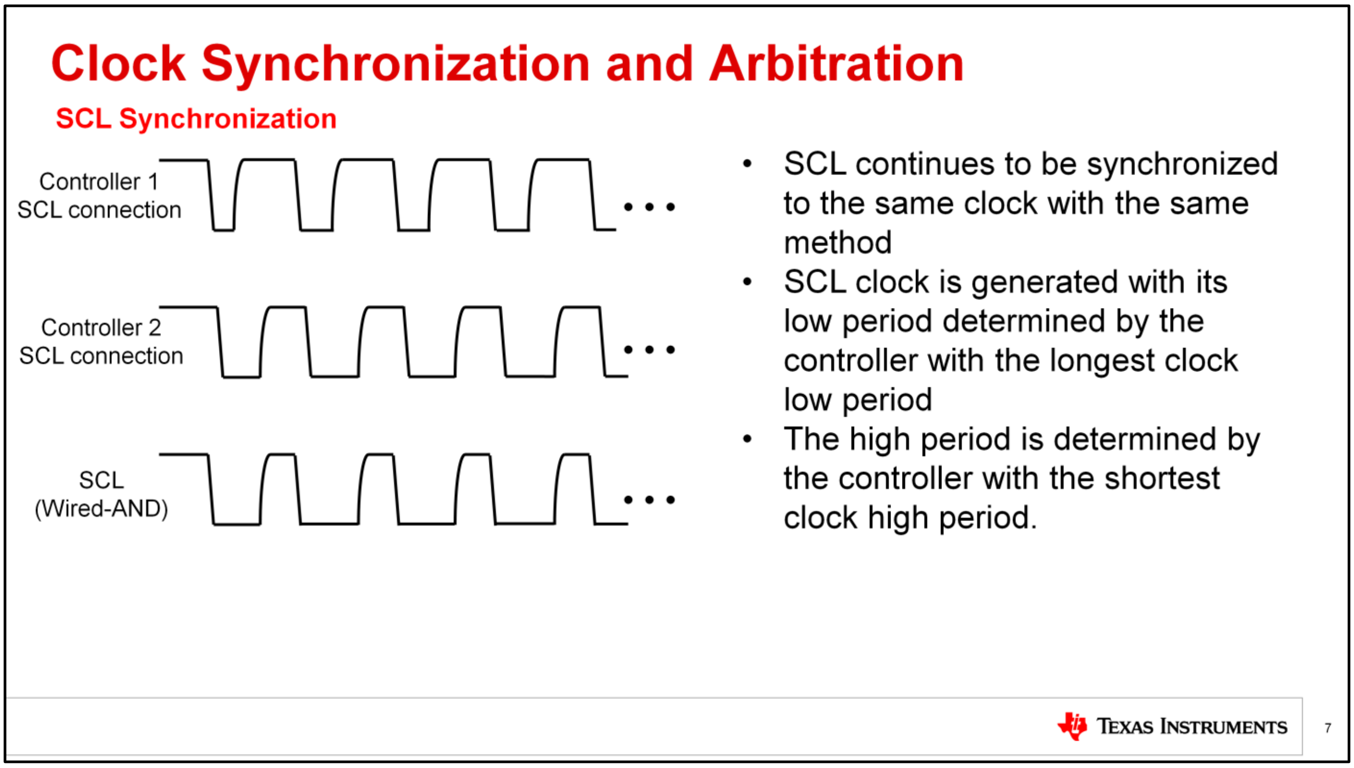
SCL Synchronization


Controller 1
SCL connection
Controller 2
SCL connection
SCL
(Wired-AND)
Controllers must monitor SCL, and must keep SCL high until all other controllers have released SCL high

- Synchronization continues as all controllers have released SCL high
- After the rising edge of SCL, all controllers pull down on SCL to complete the SCL pulse
- The first controller that completes the SCL high time determines the high time period of the pulse

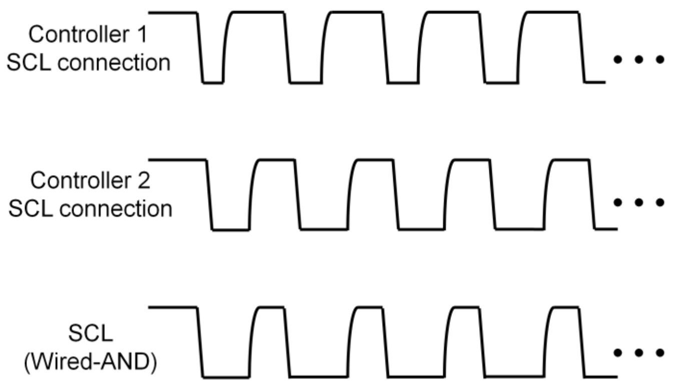
TEXAS INSTRUMENTS 6

Similarly, after the start of the serial clock pulse, all the controllers pull down on SCL to complete the serial clock pulse. Again, with the wired-AND connection, SCL is then pulled down with the first controller that responds with pulling down SCL. The first controller that completes the SCL high time period determines the high time of SCL from the wired-AND connection.



Clock Synchronization and Arbitration

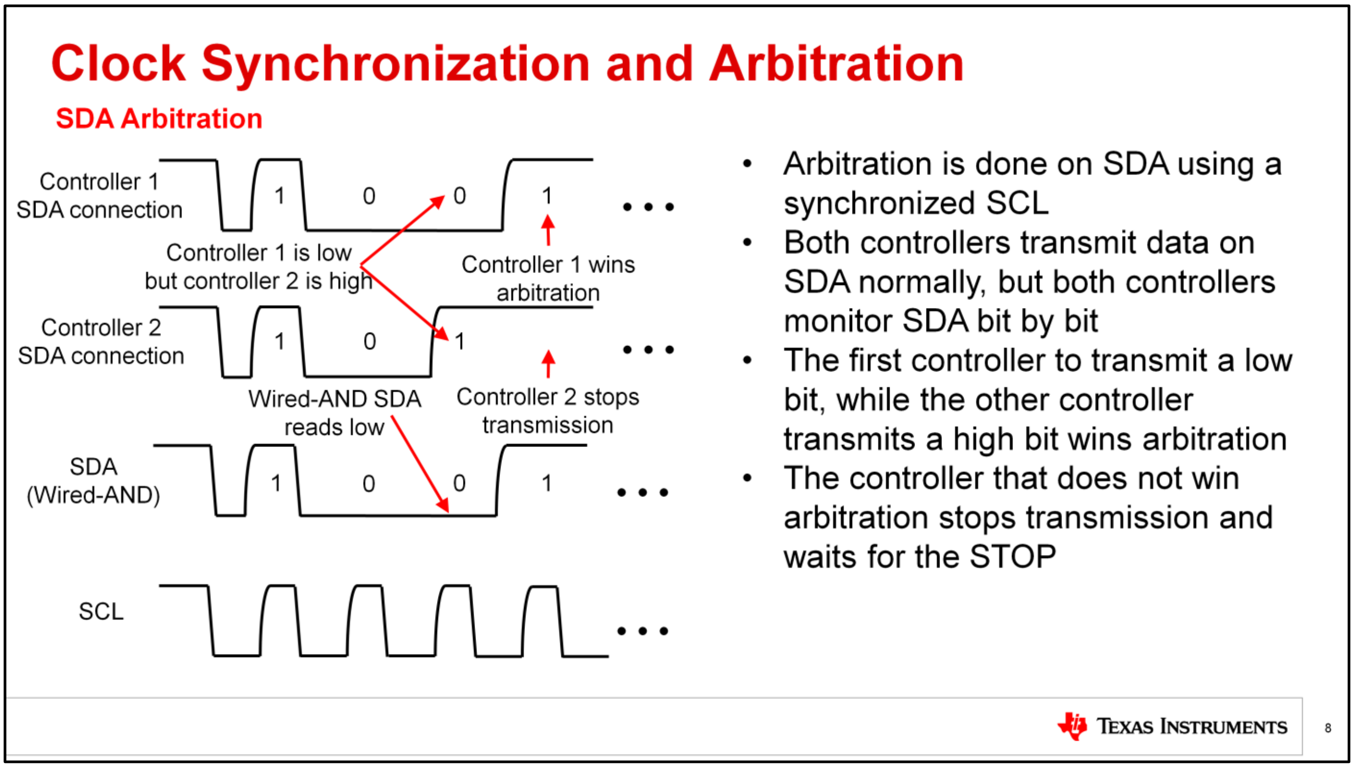
SCL Synchronization



- SCL continues to be synchronized to the same clock with the same method
- SCL clock is generated with its low period determined by the controller with the longest clock low period
- The high period is determined by the controller with the shortest clock high period.

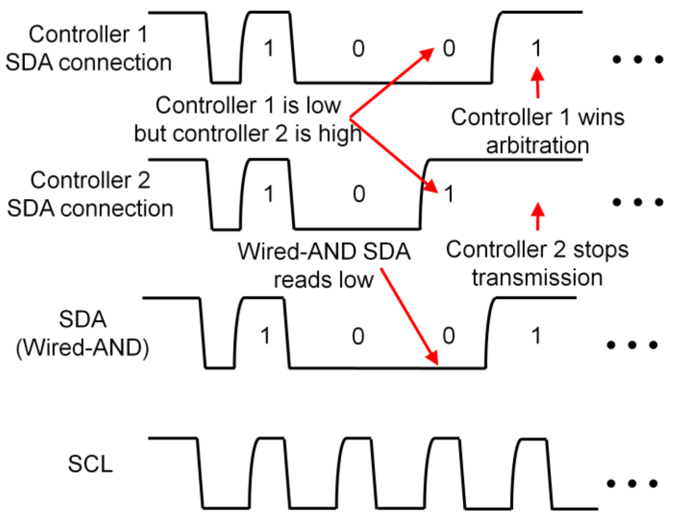
The synchronization of the SCL clock continues for subsequent clock pulses between all active controllers. Each SCL clock pulse is generated with its LOW period determined by the controller with the longest clock LOW period and the HIGH period is determined by the controller with the shortest clock HIGH period.

Again, clock synchronization works because the controllers monitor each pulse of the SCL line and react to cases where the SCL line does not match the state that the controller expects.



Clock Synchronization and Arbitration

SDA Arbitration


Controller 1 SDA connection
1
0
0
1
Controller 1 is low but controller 2 is high
Controller 1 wins arbitration
Controller 2 SDA connection
1
0
1
Wired-AND SDA reads low
Controller 2 stops transmission
SDA (Wired-AND)
1
0
0
1
SCL
• • •

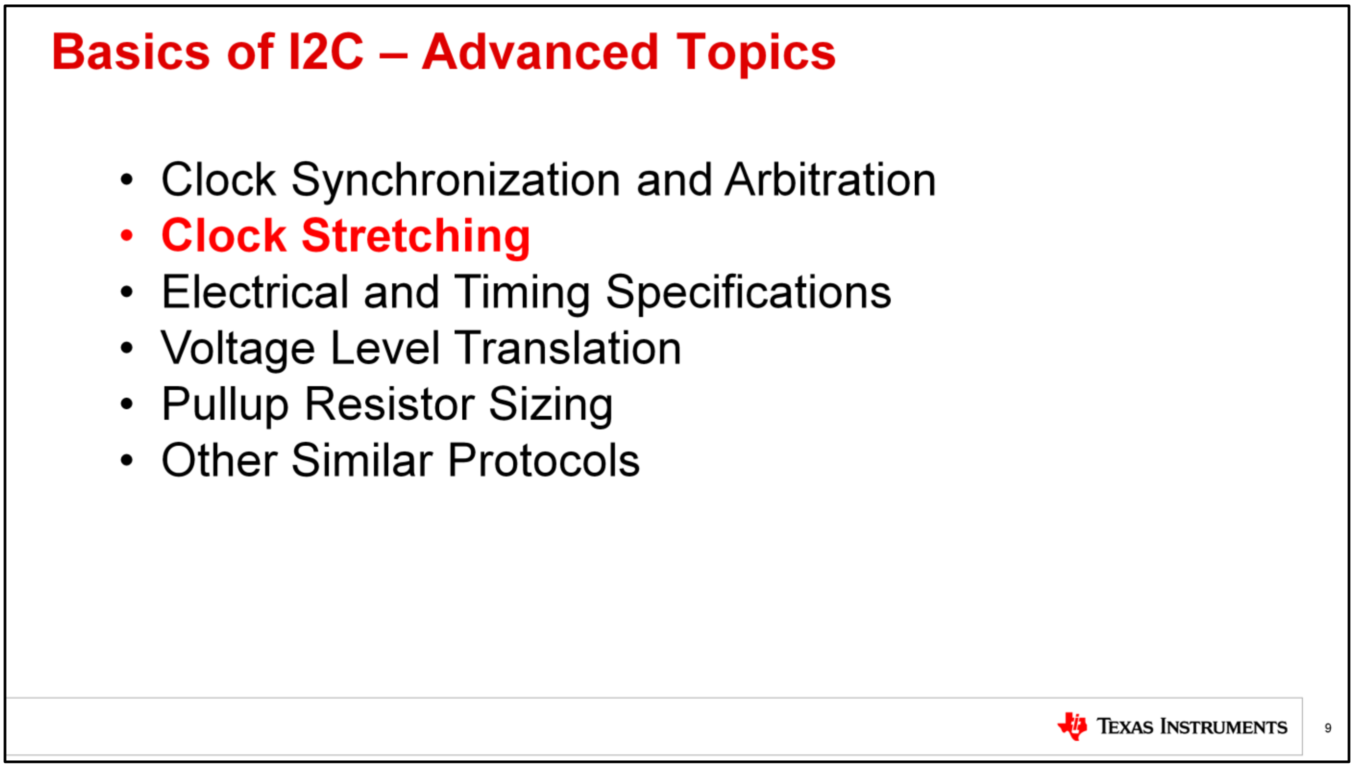
- Arbitration is done on SDA using a synchronized SCL
- Both controllers transmit data on SDA normally, but both controllers monitor SDA bit by bit
- The first controller to transmit a low bit, while the other controller transmits a high bit wins arbitration
- The controller that does not win arbitration stops transmission and waits for the STOP

TEXAS INSTRUMENTS 8

Now that the serial clocks are synchronized, arbitration is done on SDA. Both controllers transmit data normally on SDA, sending their communication to the intended target device. Similar to SCL, SDA is a wired-AND connection.

Both controllers also monitor SDA for the resulting communication. The first controller to transmit a low bit while the other controller transmits a high bit wins arbitration. With the wired-AND connection, the controller that wins arbitration does not have its communication disrupted. The controller device that loses arbitration stops its transmission and the controller device that wins arbitration continues its communication uninterrupted.

In this method of arbitration, both controllers are transmitting data at the same time. The controller that matches the wired-AND result for SDA is the controller that wins arbitration. The controller that is disrupted by the wired-AND result for SDA stops transmission and releases the I2C bus.

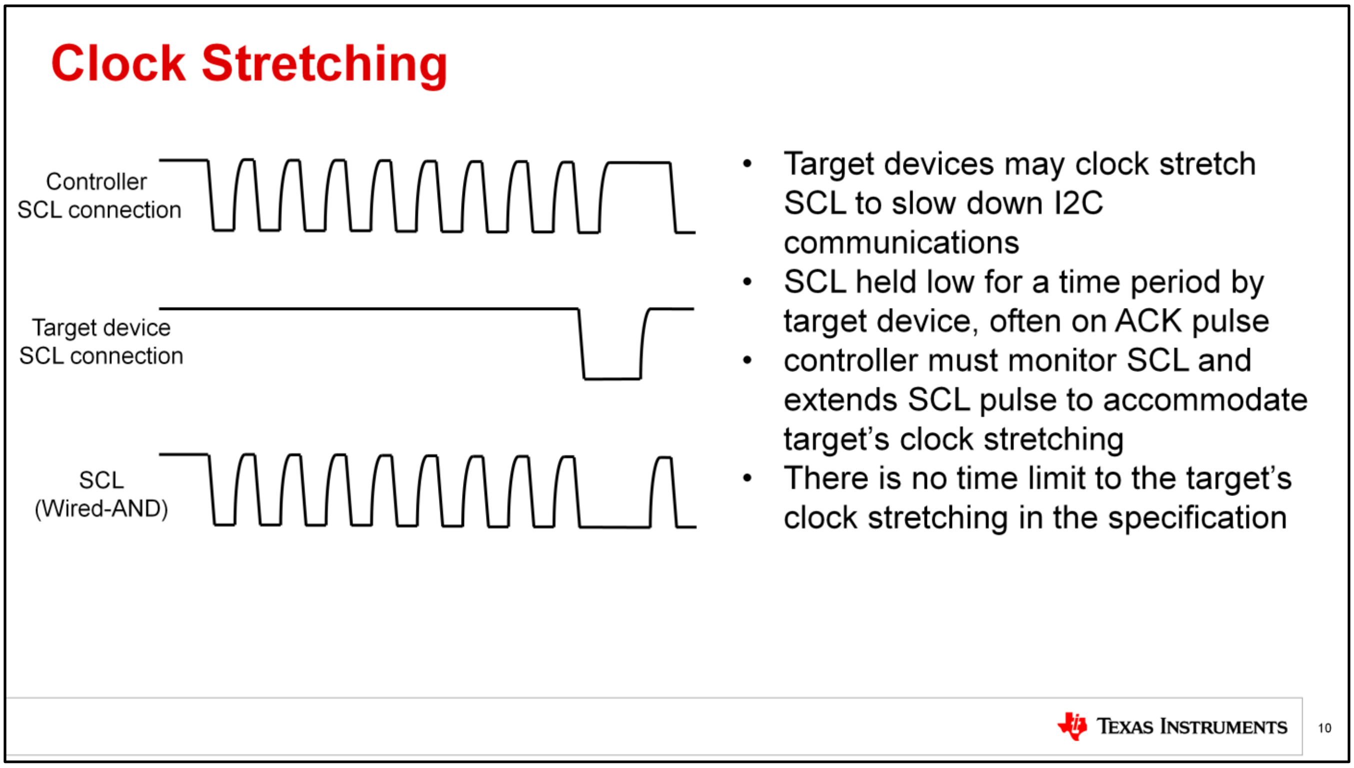

Basics of I2C – Advanced Topics

- Clock Synchronization and Arbitration
- **Clock Stretching**
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

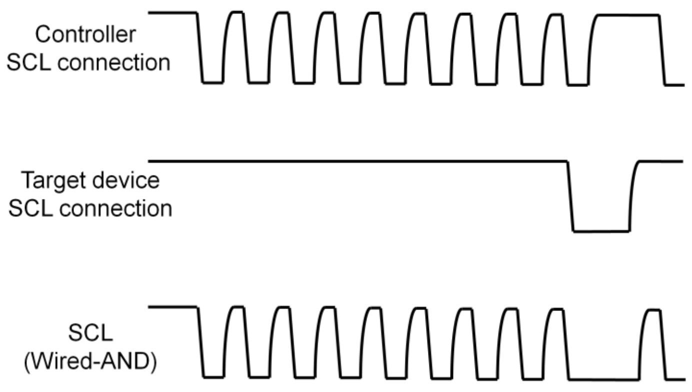
TEXAS INSTRUMENTS 9

The next topic in this presentation is clock stretching.

It’s not always the controller that controls the SCL serial clock. In some cases the target device can slow down the communication. The next set of slides show how target devices can use clock stretching to slow things down.



Clock Stretching



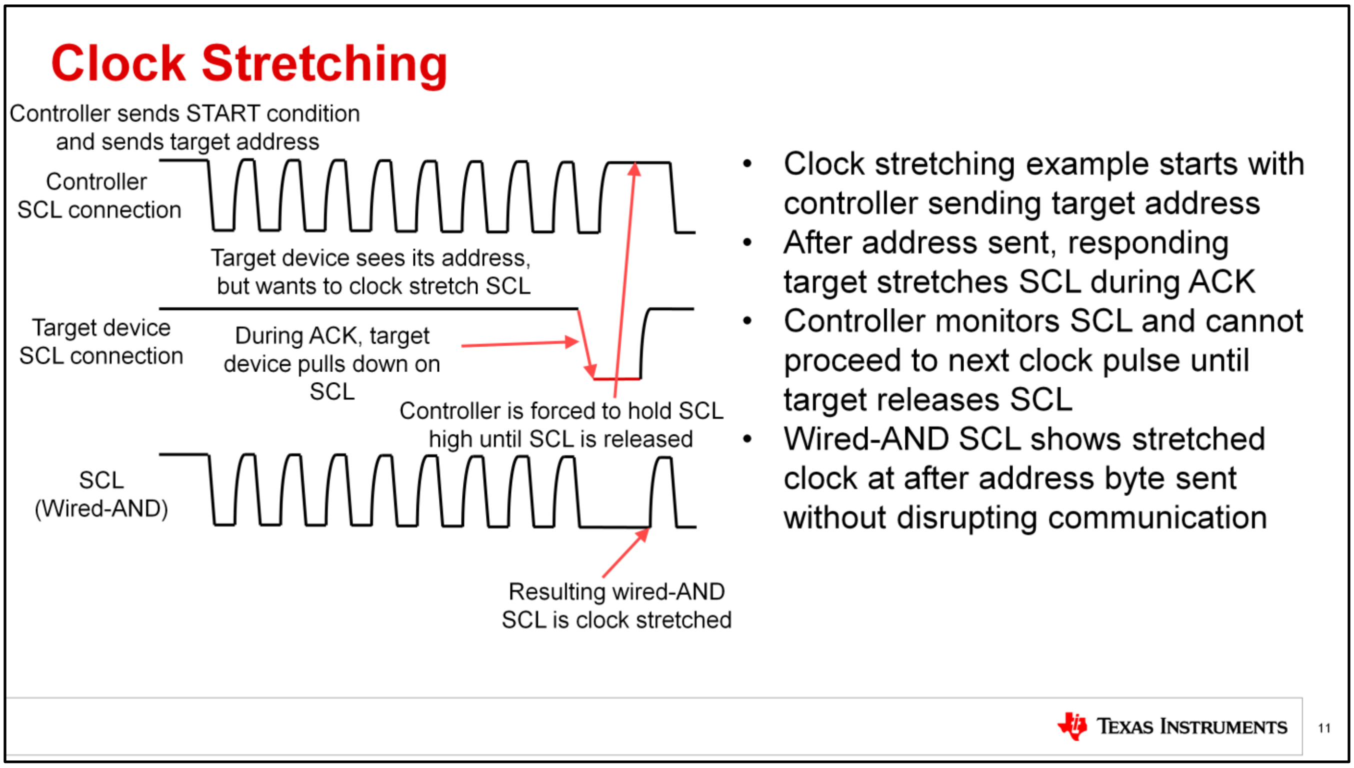
- Target devices may clock stretch SCL to slow down I2C communications
- SCL held low for a time period by target device, often on ACK pulse
- controller must monitor SCL and extends SCL pulse to accommodate target's clock stretching
- There is no time limit to the target's clock stretching in the specification

TEXAS INSTRUMENTS 10

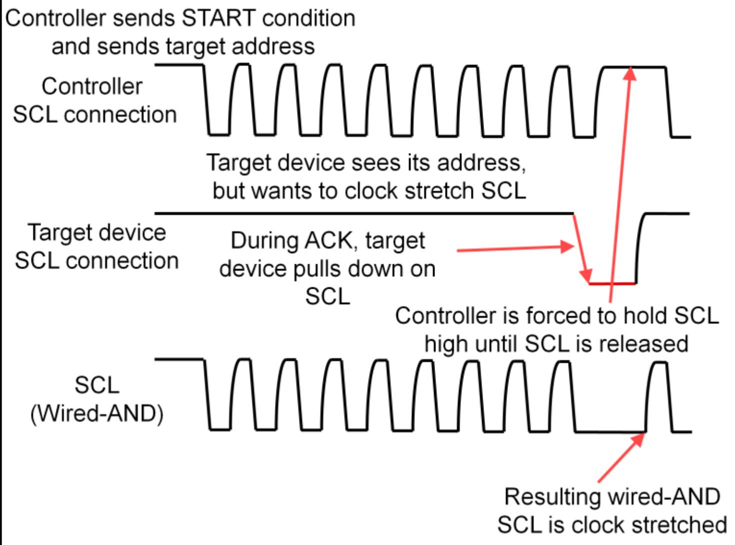
In general, the SCL line and therefore the I2C clock rate, is controlled by the controller. However, there may be times where the target device is unable to comply with the clock rate. The target device may need extra time to process a command or send data. In such cases, the target device may try to slow down the communication through clock stretching.

After a target device receives a byte of data in transmission, it may hold down SCL longer so that the controller is required to adjust the clock. This is similar to clock synchronization. The controller monitors SCL and is forced to extend the SCL pulse if it determines that SCL is still low after the controller has released it. If clock stretching is supported by the controller, any SCL pulse can be clock stretched by the target device. However, the general implementation of clock stretching is done with the SCL pulse around the ACK bit.

According to the I2C specification, there is no time limit to the target holding down SCL for clock stretching. Other similar specifications (like SMBus) have time limits for how long SCL can be held low.



Clock Stretching



Controller sends START condition and sends target address
Controller SCL connection
Target device sees its address, but wants to clock stretch SCL
Target device SCL connection
During ACK, target device pulls down on SCL
Controller is forced to hold SCL high until SCL is released
SCL (Wired-AND)
Resulting wired-AND SCL is clock stretched

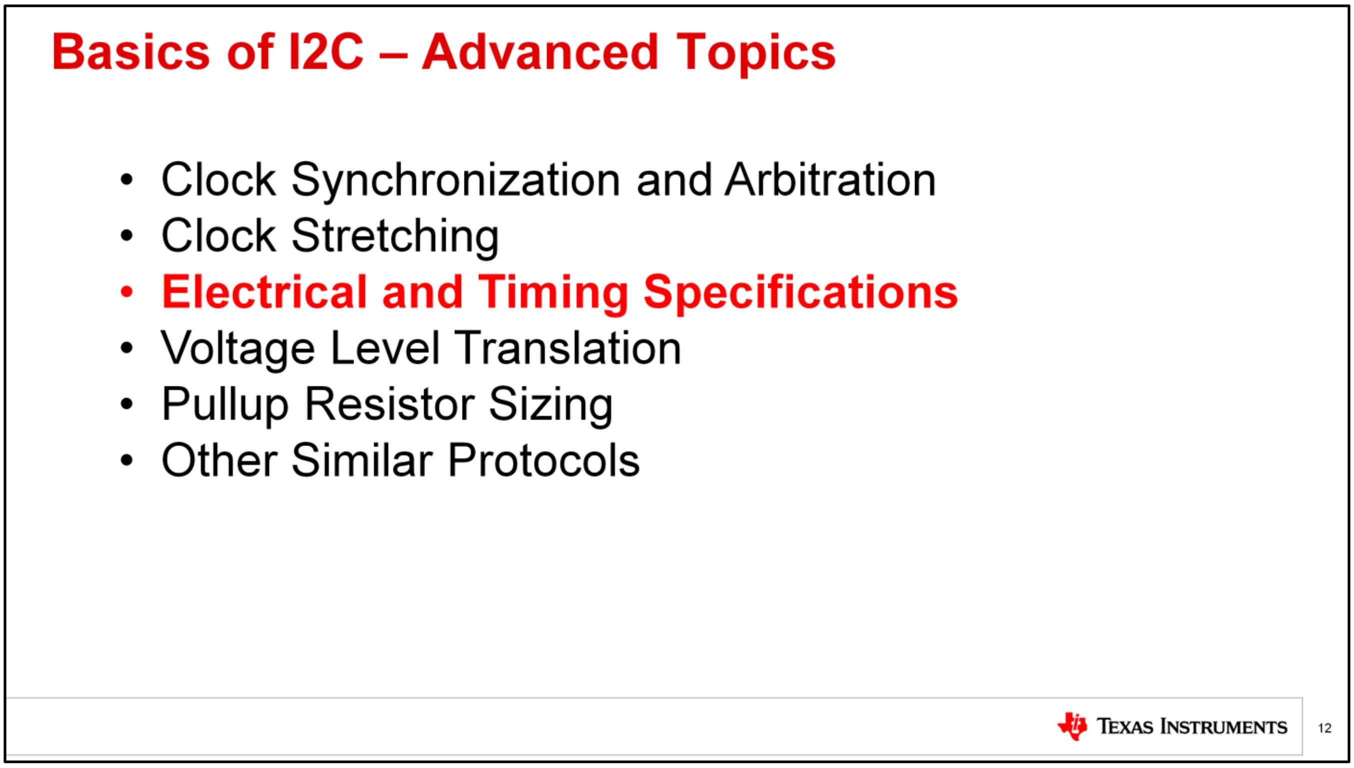
- Clock stretching example starts with controller sending target address
- After address sent, responding target stretches SCL during ACK
- Controller monitors SCL and cannot proceed to next clock pulse until target releases SCL
- Wired-AND SCL shows stretched clock at after address byte sent without disrupting communication

TEXAS INSTRUMENTS 11

Here's an example of the target device clock stretching SCL. In this example, the controller issues a START and sends the target device address.

When the target device recognizes the controller is sending the proper target address, the target device then begins to ACK the address. If the target device needs to slow down communications, it can pull down on SCL. This is the only instance the target device can control the SCL.

If the controller responds to clock stretching, it monitors SCL and sees that SCL remains low even though the controller has released SCL. Because of this, the controller cannot continue with the SCL pulse until the SCL is released by the target. The controller continues to monitor SCL. Once SCL is released high, the controller can then continue past the target device's ACK and continue with the next byte transmission. The resulting wired-AND connection of SCL shows the SCL stretched. Data transmission is delayed by the target device without disrupting communication.



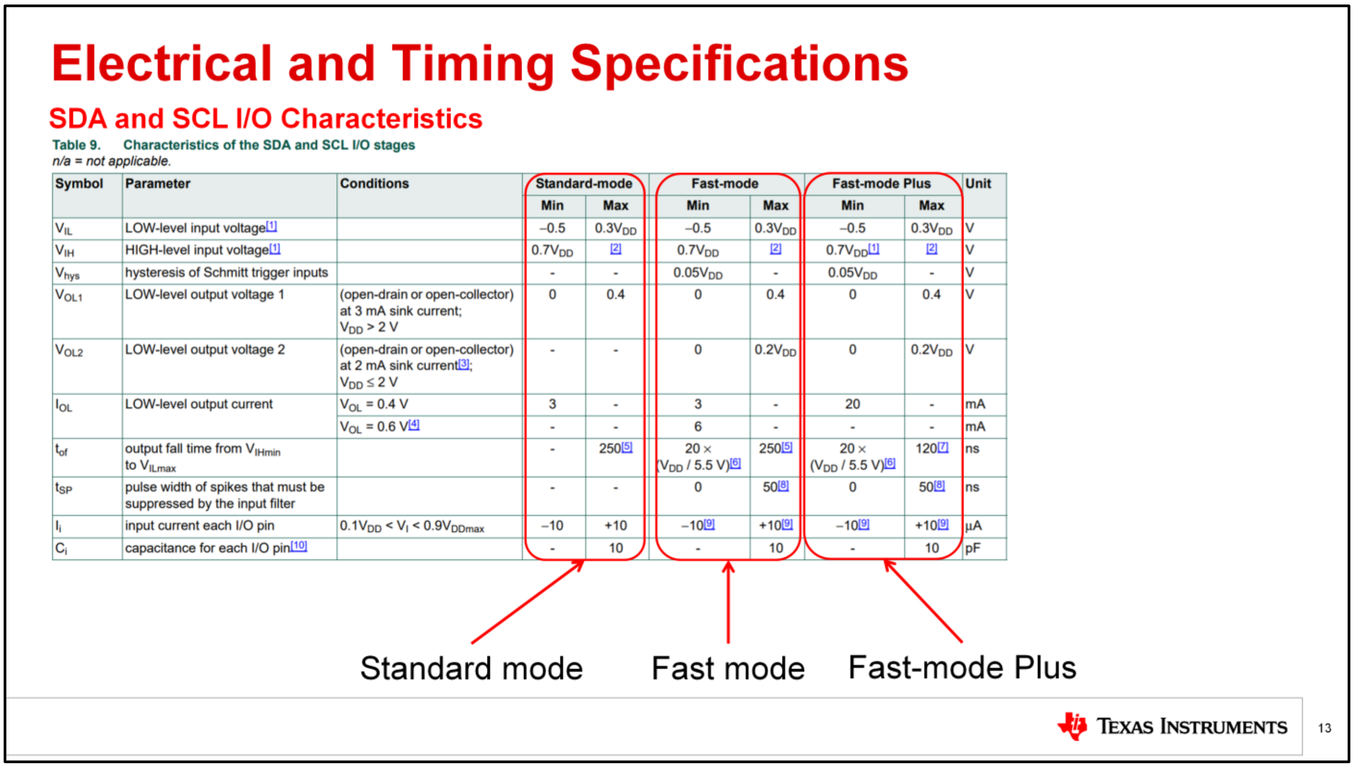
Basics of I2C – Advanced Topics

- Clock Synchronization and Arbitration
- Clock Stretching
- **Electrical and Timing Specifications**
- Voltage Level Translation
- Pullup Resistor Sizing
- Other Similar Protocols

12

The datasheet for every I2C device will have electrical and timing specifications that cover the characteristics for the I2C bus. Because I2C is a common protocol, these specifications should be matched from device to device. This section will discuss the electrical and timing characteristics and how they are shown in the I2C specification.

We won’t go into detail about each of the specifications, but we’ll give an overview of how these specifications are organized. Datasheets for I2C devices will cover what you need to know to operate our devices. However, you can search out the I2C specifications and read more about each of these characteristics.



Electrical and Timing Specifications

SDA and SCL I/O Characteristics

Table 9. Characteristics of the SDA and SCL I/O stages

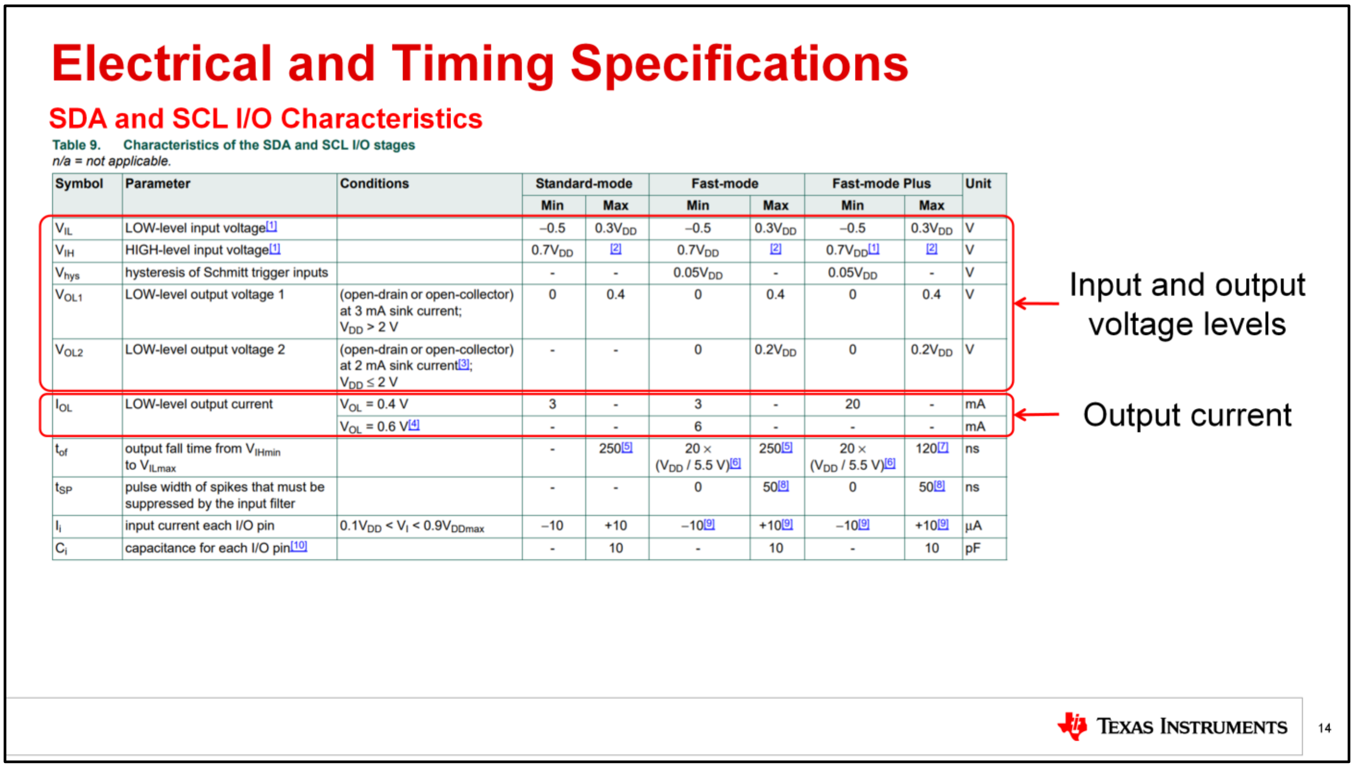
n/a = not applicable.

Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
V_{IL}	LOW-level input voltage[1]		–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage[1]		$0.7V_{DD}$	[2]	$0.7V_{DD}$	[2]	$0.7V_{DD}$[1]	[2]	V
V_{hys}	hysteresis of Schmitt trigger inputs		-	-	$0.05V_{DD}$	-	$0.05V_{DD}$	-	V
V_{OL1}	LOW-level output voltage 1	(open-drain or open-collector) at 3 mA sink current; $V_{DD} > 2$ V	0	0.4	0	0.4	0	0.4	V
V_{OL2}	LOW-level output voltage 2	(open-drain or open-collector) at 2 mA sink current[3]; $V_{DD} \leq 2$ V	-	-	0	$0.2V_{DD}$	0	$0.2V_{DD}$	V
I_{OL}	LOW-level output current	$V_{OL} = 0.4$ V	3	-	3	-	20	-	mA
		$V_{OL} = 0.6$ V[4]	-	-	6	-	-	-	mA
t_{of}	output fall time from V_{IHmin} to V_{ILmax}		-	250[5]	$20 \times (V_{DD} / 5.5\ V)$[6]	250[5]	$20 \times (V_{DD} / 5.5\ V)$[6]	120[7]	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		-	-	0	50[8]	0	50[8]	ns
I_i	input current each I/O pin	$0.1V_{DD} < V_I < 0.9V_{DDmax}$	–10	+10	–10[9]	+10[9]	–10[9]	+10[9]	µA
C_i	capacitance for each I/O pin[10]		-	10	-	10	-	10	pF

Standard mode Fast mode Fast-mode Plus

TEXAS INSTRUMENTS 13

As an example, here we show Table 9 from the I2C specifications. This table shows the input/output characteristics for the I2C bus lines. First, you can see from the columns that the specifications are different for different I2C speed modes. Minimums and maximums are listed for standard mode, fast mode, and fast-mode plus. Because the devices operate at different speeds, these specifications are different to accommodate the differences in voltage and timing.



Electrical and Timing Specifications

SDA and SCL I/O Characteristics

Table 9. Characteristics of the SDA and SCL I/O stages

n/a = not applicable.

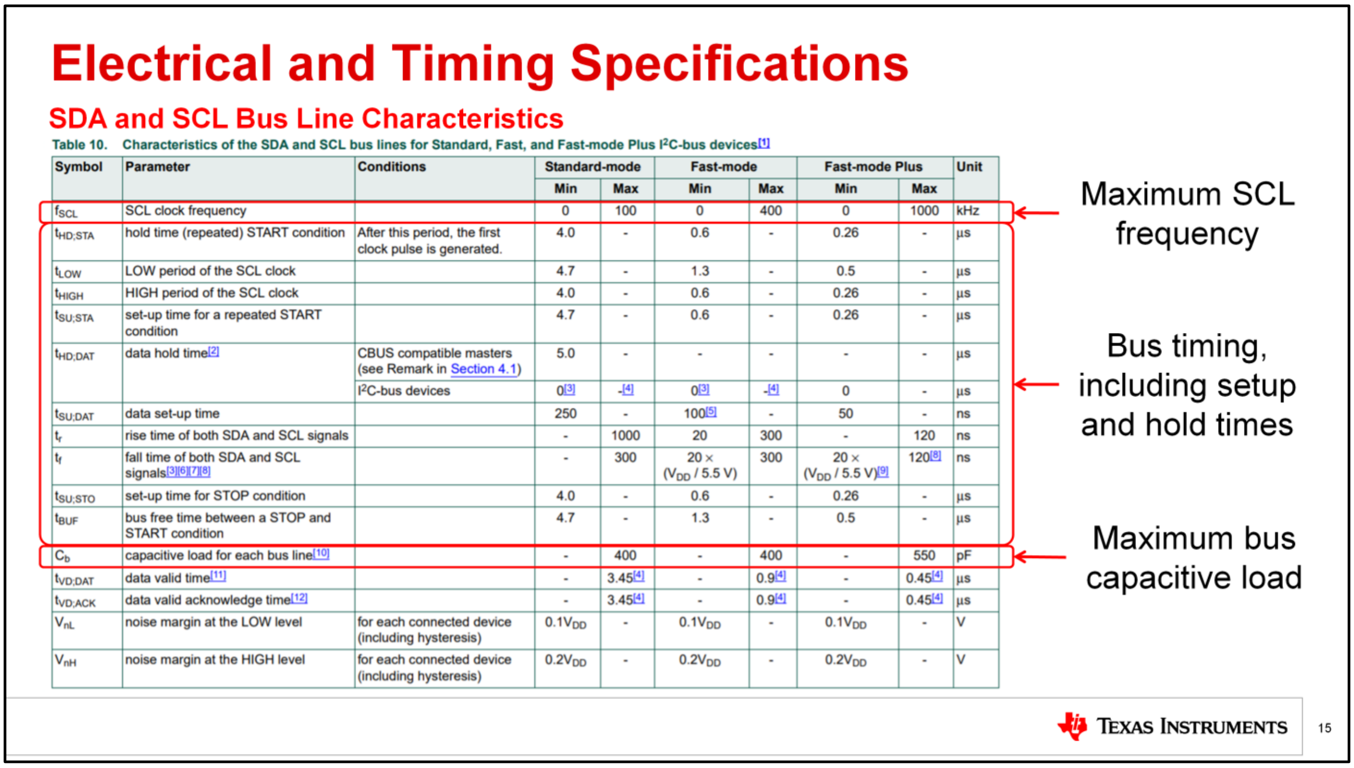
Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
V_{IL}	LOW-level input voltage[1]		–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	–0.5	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage[1]		$0.7V_{DD}$	[2]	$0.7V_{DD}$	[2]	$0.7V_{DD}$[1]	[2]	V
V_{hys}	hysteresis of Schmitt trigger inputs		-	-	$0.05V_{DD}$	-	$0.05V_{DD}$	-	V
V_{OL1}	LOW-level output voltage 1	(open-drain or open-collector) at 3 mA sink current; $V_{DD} > 2$ V	0	0.4	0	0.4	0	0.4	V
V_{OL2}	LOW-level output voltage 2	(open-drain or open-collector) at 2 mA sink current[3]; $V_{DD} \leq 2$ V	-	-	0	$0.2V_{DD}$	0	$0.2V_{DD}$	V
I_{OL}	LOW-level output current	$V_{OL} = 0.4$ V	3	-	3	-	20	-	mA
		$V_{OL} = 0.6$ V[4]	-	-	6	-	-	-	mA
t_{of}	output fall time from V_{IHmin} to V_{ILmax}		-	250[5]	20 × (V_{DD} / 5.5 V)[6]	250[5]	20 × (V_{DD} / 5.5 V)[6]	120[7]	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		-	-	0	50[8]	0	50[8]	ns
I_i	input current each I/O pin	$0.1V_{DD} < V_i < 0.9V_{DDmax}$	–10	+10	–10[9]	+10[9]	–10[9]	+10[9]	µA
C_i	capacitance for each I/O pin[10]		-	10	-	10	-	10	pF

Input and output voltage levels

Output current

Texas Instruments 14

Highlighting some of the parameters, Table 9 gives specifications for low level and high level input and output voltages for SCL and SDA. This ensures that each I2C bus line has a voltage range that correctly transmits and receives high and low levels. This table also gives the minimum output current that the device open drains pull down on SCL and SDA.



Electrical and Timing Specifications

SDA and SCL Bus Line Characteristics

Table 10. Characteristics of the SDA and SCL bus lines for Standard, Fast, and Fast-mode Plus I²C-bus devices[1]

Symbol	Parameter	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Unit
			Min	Max	Min	Max	Min	Max	
f_{SCL}	SCL clock frequency		0	100	0	400	0	1000	kHz
$t_{HD;STA}$	hold time (repeated) START condition	After this period, the first clock pulse is generated.	4.0	-	0.6	-	0.26	-	µs
t_{LOW}	LOW period of the SCL clock		4.7	-	1.3	-	0.5	-	µs
t_{HIGH}	HIGH period of the SCL clock		4.0	-	0.6	-	0.26	-	µs
$t_{SU;STA}$	set-up time for a repeated START condition		4.7	-	0.6	-	0.26	-	µs
$t_{HD;DAT}$	data hold time[2]	CBUS compatible masters (see Remark in Section 4.1)	5.0	-	-	-	-	-	µs
		I²C-bus devices	0[3]	-[4]	0[3]	-[4]	0	-	µs
$t_{SU;DAT}$	data set-up time		250	-	100[5]	-	50	-	ns
t_r	rise time of both SDA and SCL signals		-	1000	20	300	-	120	ns
t_f	fall time of both SDA and SCL signals[3][6][7][8]		-	300	20 × (V_{DD} / 5.5 V)	300	20 × (V_{DD} / 5.5 V)[9]	120[8]	ns
$t_{SU;STO}$	set-up time for STOP condition		4.0	-	0.6	-	0.26	-	µs
t_{BUF}	bus free time between a STOP and START condition		4.7	-	1.3	-	0.5	-	µs
C_b	capacitive load for each bus line[10]		-	400	-	400	-	550	pF
$t_{VD;DAT}$	data valid time[11]		-	3.45[4]	-	0.9[4]	-	0.45[4]	µs
$t_{VD;ACK}$	data valid acknowledge time[12]		-	3.45[4]	-	0.9[4]	-	0.45[4]	µs
V_{nL}	noise margin at the LOW level	for each connected device (including hysteresis)	$0.1V_{DD}$	-	$0.1V_{DD}$	-	$0.1V_{DD}$	-	V
V_{nH}	noise margin at the HIGH level	for each connected device (including hysteresis)	$0.2V_{DD}$	-	$0.2V_{DD}$	-	$0.2V_{DD}$	-	V

Maximum SCL frequency

Bus timing, including setup and hold times

Maximum bus capacitive load

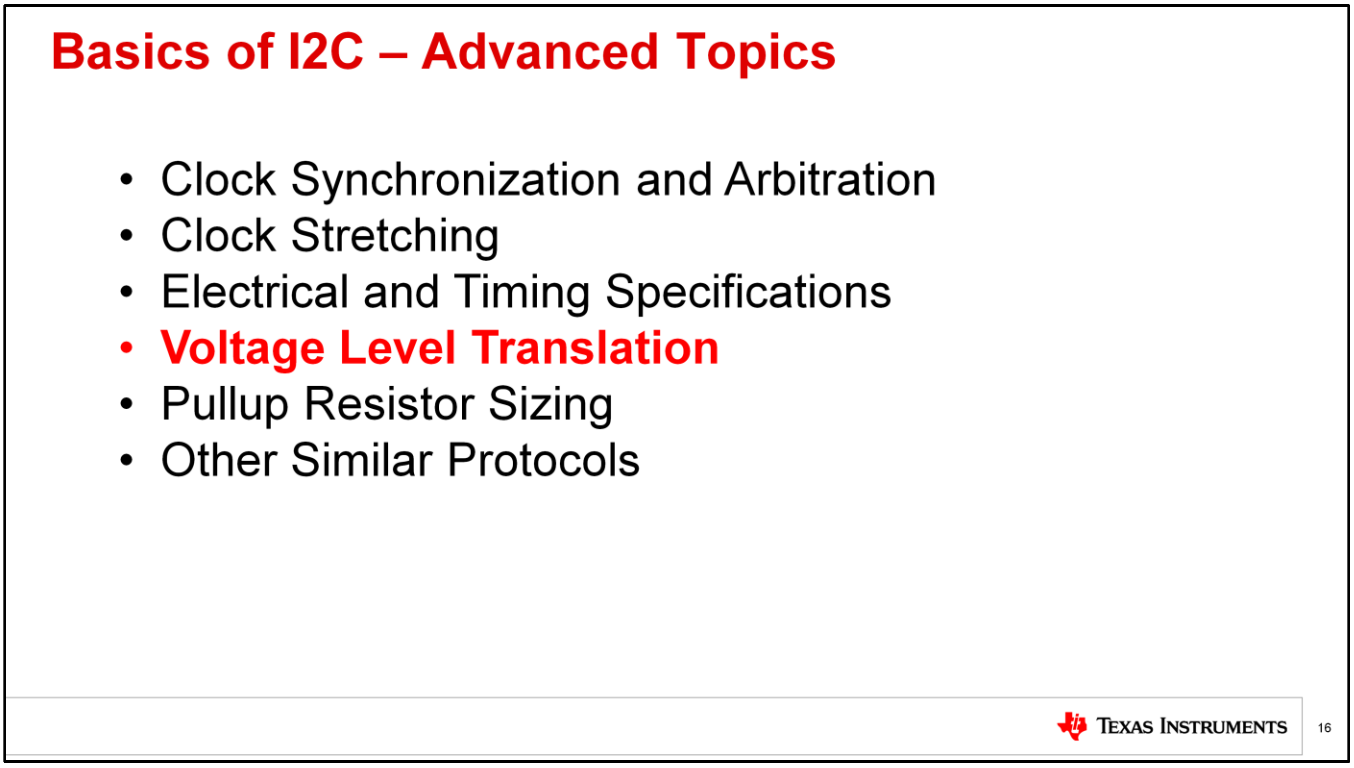
Texas Instruments

15

Table 10 of the I2C specification gives additional minimums and maximums for the SDA and SCL bus timing. The first key parameter gives the maximum SCL clock frequency for each of the I2C speed modes. Much of the rest of the table gives various setup and hold times for the SDA in relation to SCL. There is also timing information for the START and STOP conditions.

One last key parameter shows the maximum capacitive load allowed on the I2C bus lines. With the high signals based on pullup resistances, the load capacitance may determine the speed at which the I2C bus communicates. Later in this presentation, we'll show how this bus capacitance can be used to determine a range for the I2C pullup resistances.

In whatever I2C devices you use, these SCL and SDA bus line characteristics can be found in their respective datasheets. The datasheets will give enough of these characteristics to setup the device correctly. Again, for further information you can search out the I2C specifications and read more about these characteristics

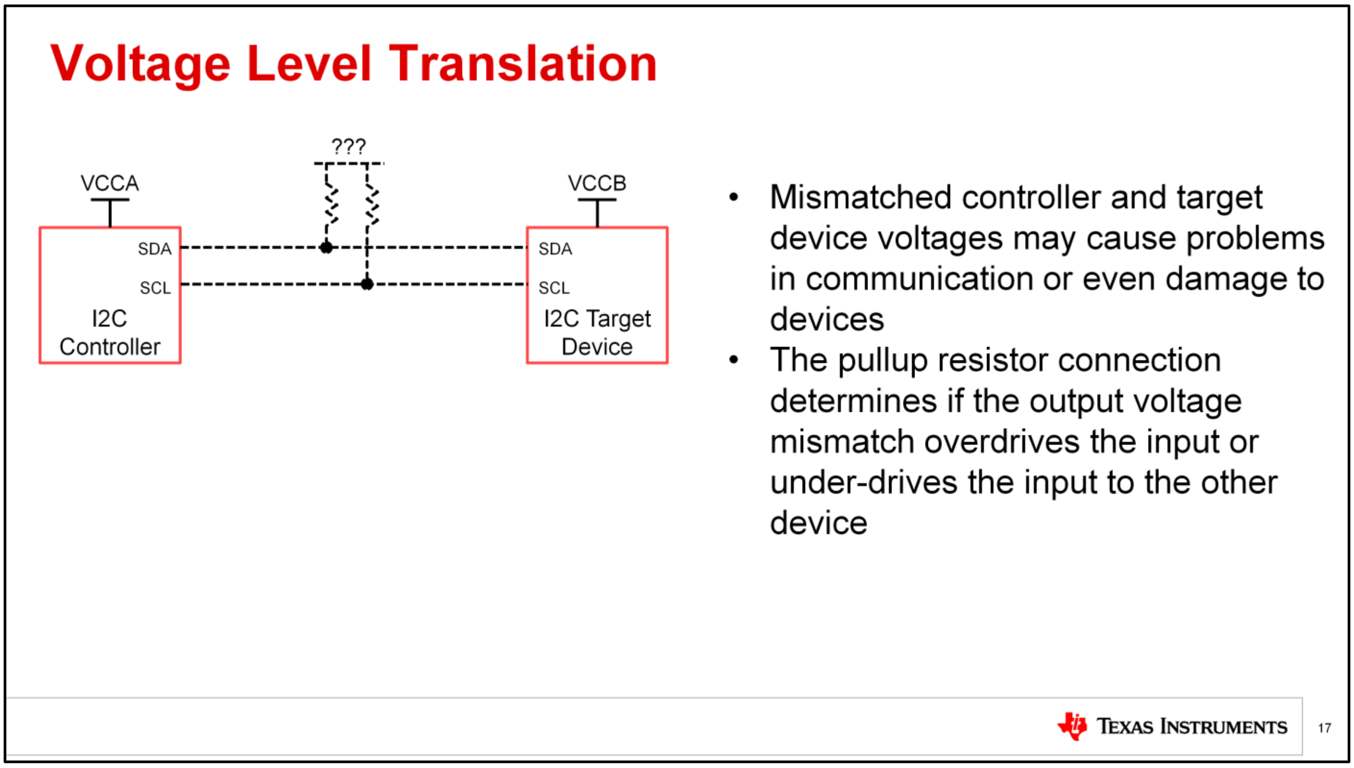


Basics of I2C – Advanced Topics

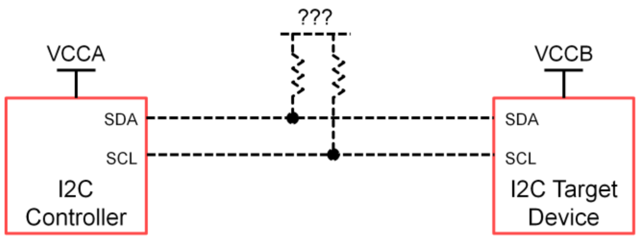
- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- **Voltage Level Translation**
- Pullup Resistor Sizing
- Other Similar Protocols

TEXAS INSTRUMENTS 16

Larger systems may have multiple power sources with multiple voltages. These different voltages may power different I2C controllers and target devices. Here, we'll talk about voltage level translation and how these different I2C voltages may (or may not) interact.



Voltage Level Translation

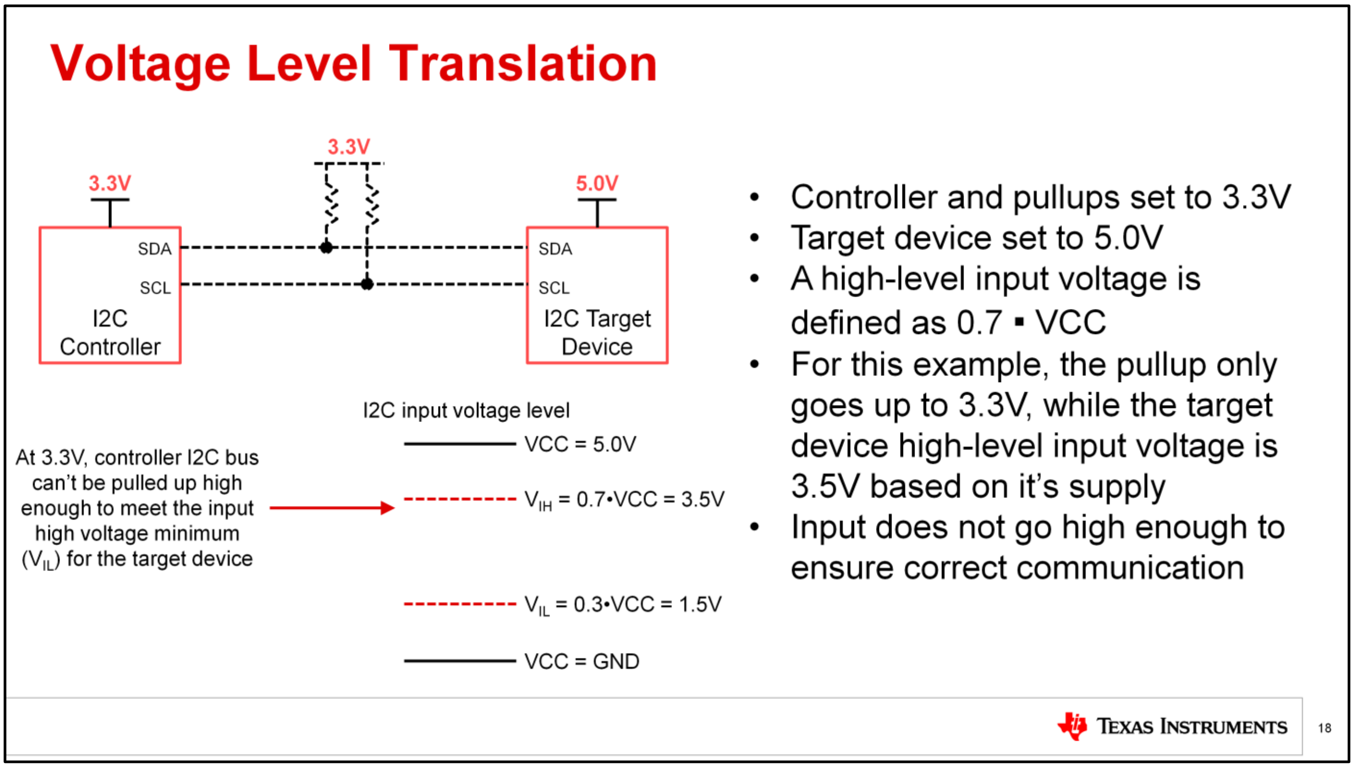


- Mismatched controller and target device voltages may cause problems in communication or even damage to devices
- The pullup resistor connection determines if the output voltage mismatch overdrives the input or under-drives the input to the other device

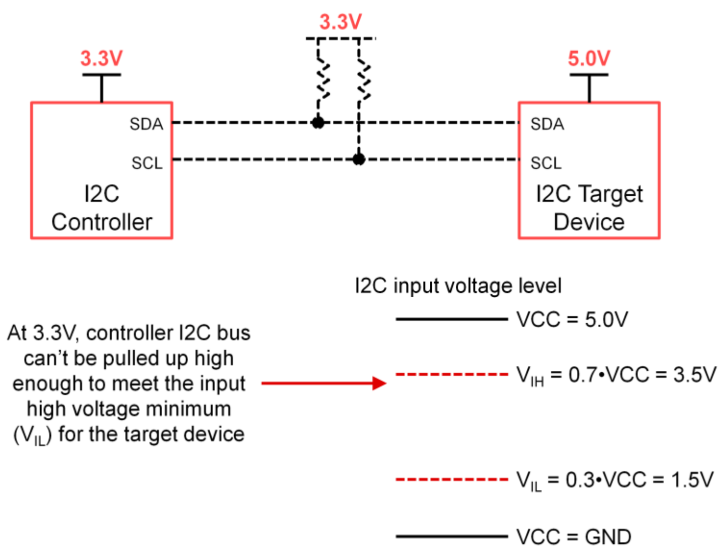
TEXAS INSTRUMENTS 17

One common problem with designing large systems is the mixing of different voltage levels within the system. For example, what happens when the controller and the target device do not run on the same voltage?

Mismatched voltages in the supply can disrupt communication or even damage a device. The connection of the pullup resistors determines if the output voltage of one overdrives or underdrives the input of the next device. Several examples can show some of the consequences of the mismatch.



Voltage Level Translation



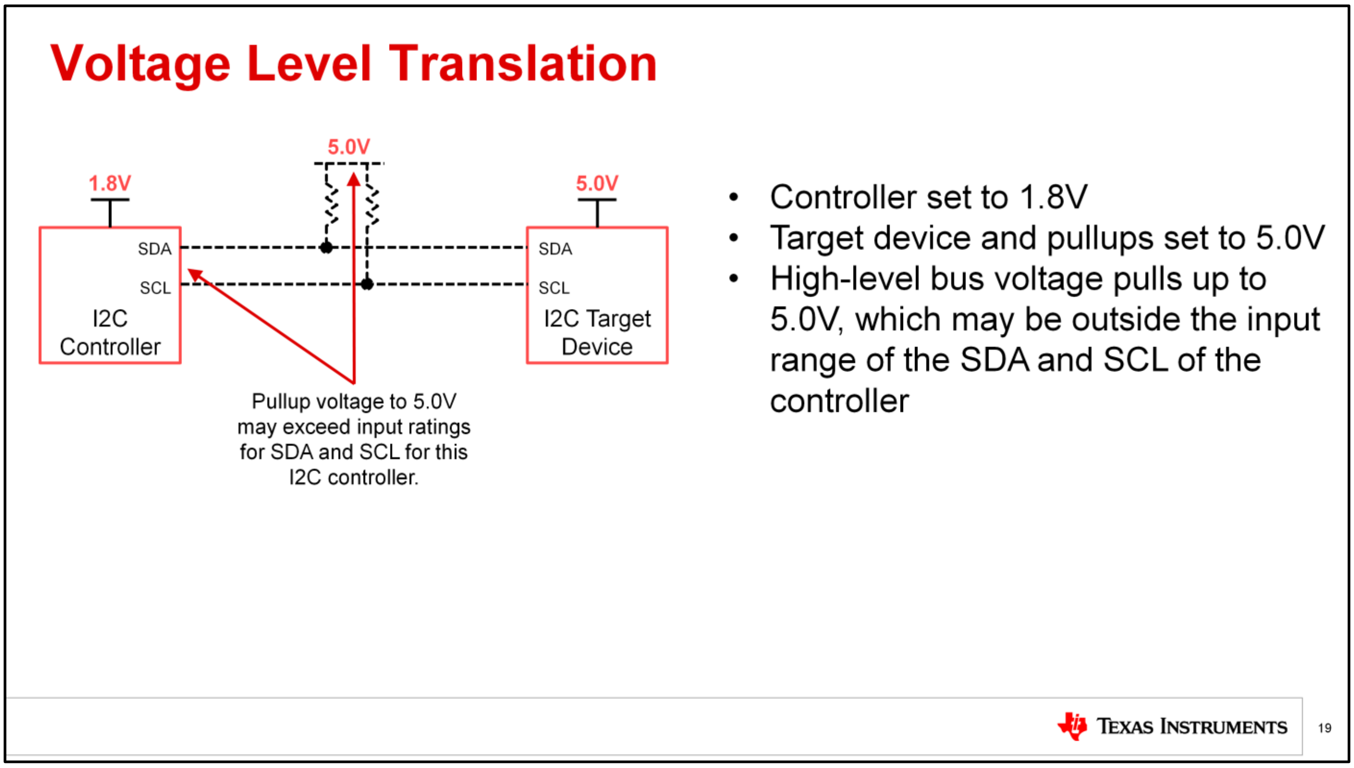
- Controller and pullups set to 3.3V
- Target device set to 5.0V
- A high-level input voltage is defined as 0.7 ▪ VCC
- For this example, the pullup only goes up to 3.3V, while the target device high-level input voltage is 3.5V based on it's supply
- Input does not go high enough to ensure correct communication

TEXAS INSTRUMENTS 18

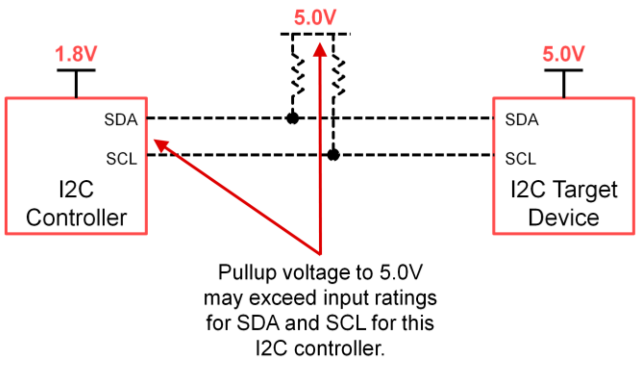
Here's one example of supply mismatch with different I2C devices. In this example, the controller and the pullups are set to 3.3V, while the target device is set to 5.0V.

In the I2C specification, there are minimum and maximum voltages required for a digital input voltage to be accurately interpreted as a digital high or low. For example, the SDA and SCL are interpreted as a digital input low voltage when the input goes below the maximum 0.3 ▪ VCC. Also, the SDA and SCL are interpreted as a digital input high voltage when the input goes above the minimum of 0.7 ▪ VCC. This latter specification is important for the mismatched supplies.

With the pullups tied to the lower supply of 3.3V, the resistors are never able to pull up higher than the minimum required voltage of 3.5V. In this case, neither the SDA, nor the SCL are ensured to be interpreted as a digital high. This would potentially prevent communication between the devices.



Voltage Level Translation

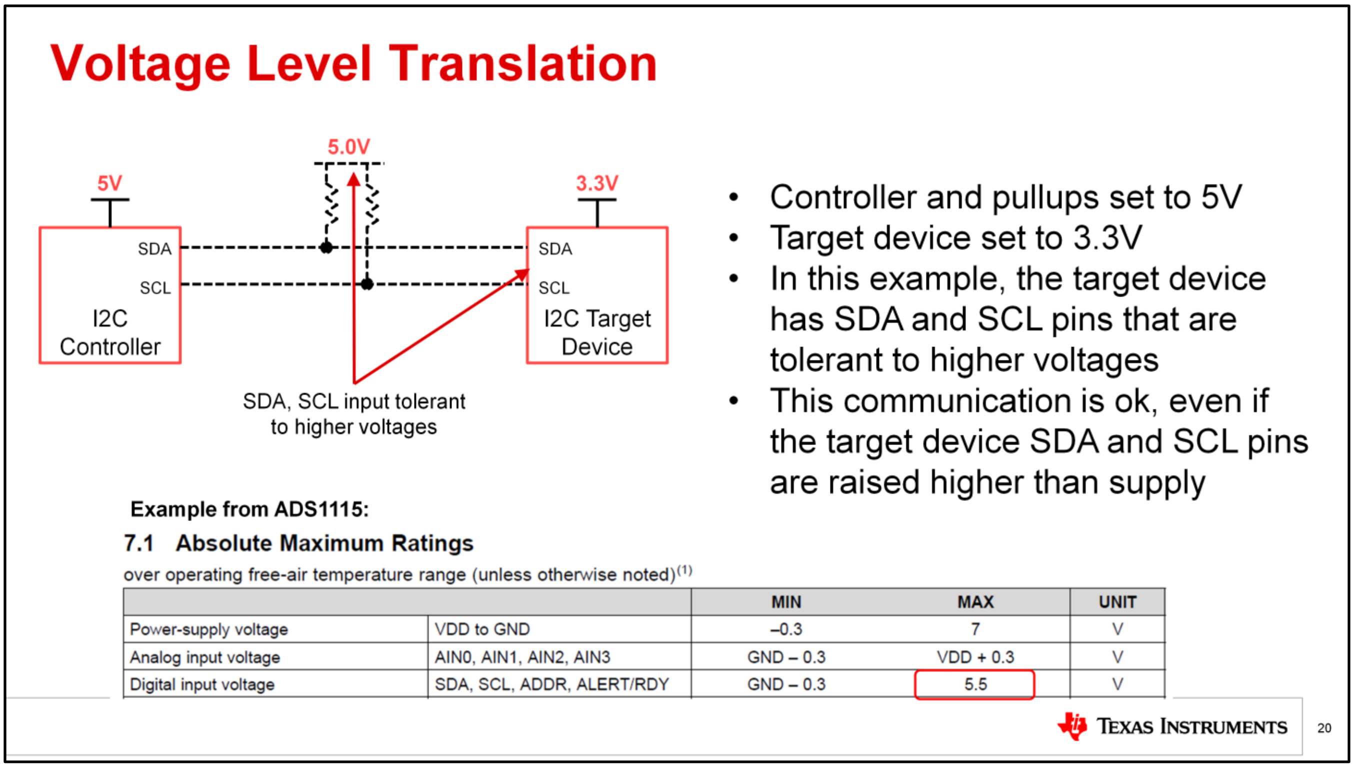

5.0V
1.8V
5.0V
SDA
SCL
I2C Controller
SDA
SCL
I2C Target Device
Pullup voltage to 5.0V may exceed input ratings for SDA and SCL for this I2C controller.

- Controller set to 1.8V
- Target device and pullups set to 5.0V
- High-level bus voltage pulls up to 5.0V, which may be outside the input range of the SDA and SCL of the controller

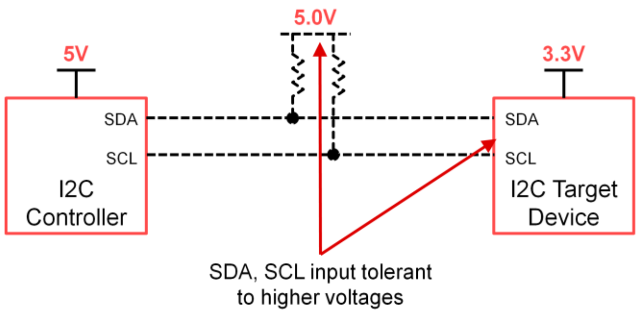
TEXAS INSTRUMENTS 19

Here’s another example where the controller is set to 1.8V, but the pullups and the target device are set to 5.0V.

In this example, the I2C bus lines are able to be pulled up to 5.0V. However, the controller device may not accept voltages that high. If the difference between the device voltages are too great, the lower voltage device may be susceptible to damage.



Voltage Level Translation



- Controller and pullups set to 5V
- Target device set to 3.3V
- In this example, the target device has SDA and SCL pins that are tolerant to higher voltages
- This communication is ok, even if the target device SDA and SCL pins are raised higher than supply

Example from ADS1115:

7.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)(1)

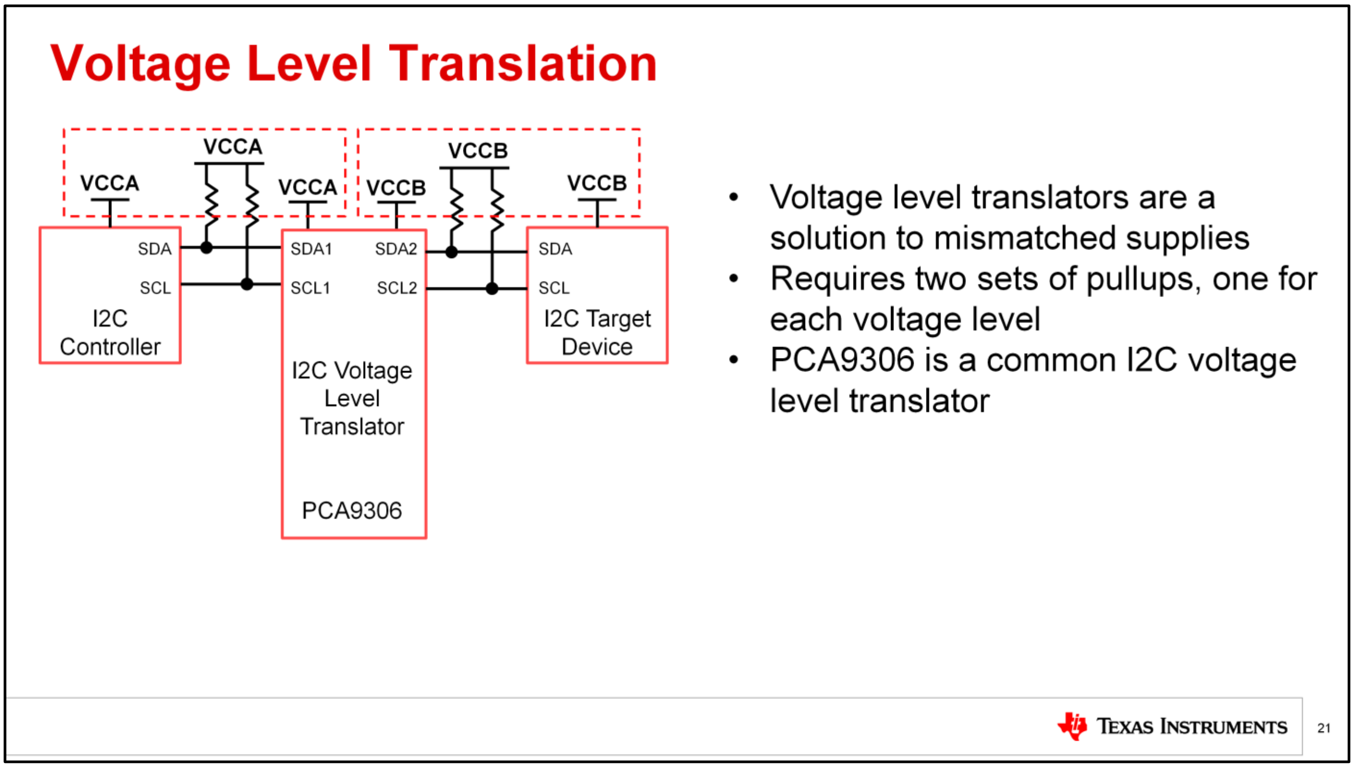
		MIN	MAX	UNIT
Power-supply voltage	VDD to GND	–0.3	7	V
Analog input voltage	AIN0, AIN1, AIN2, AIN3	GND – 0.3	VDD + 0.3	V
Digital input voltage	SDA, SCL, ADDR, ALERT/RDY	GND – 0.3	$\boxed{5.5}$	V

TEXAS INSTRUMENTS 20

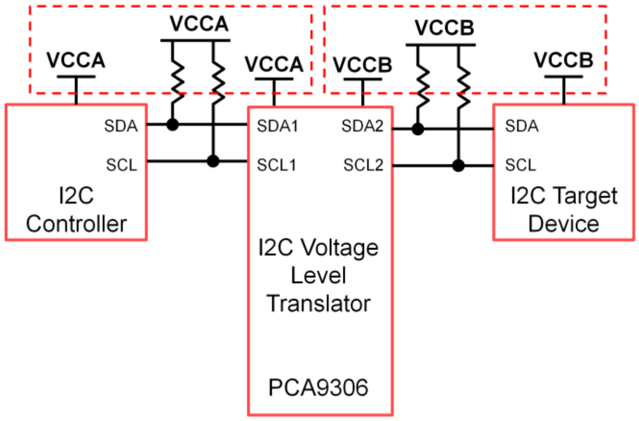
Here's an example where the controller and pullups are set to 5V, but the target device is set to 3.3V.

The I2C bus lines are able to be pulled up to 5.0V, exceeding the target device supply. However, the target device has inputs tolerant to higher voltages. This is a feature in some I2C devices. This may allow for direct connections between the I2C bus with pullups to the higher voltage supply. Check with the device datasheets for this possible feature.

The ADS1115 is just one device that has SDA and SCL lines that are tolerant to voltages higher than the supply. Looking at the Absolute Maximum Table from the datasheet, the maximum digital input voltage is 5.5V, regardless of the supply voltage. With this type of I2C line, the target device can tolerate pullup voltages higher than the supply. This allows for I2C communication between the devices even with different supply voltages.



Voltage Level Translation

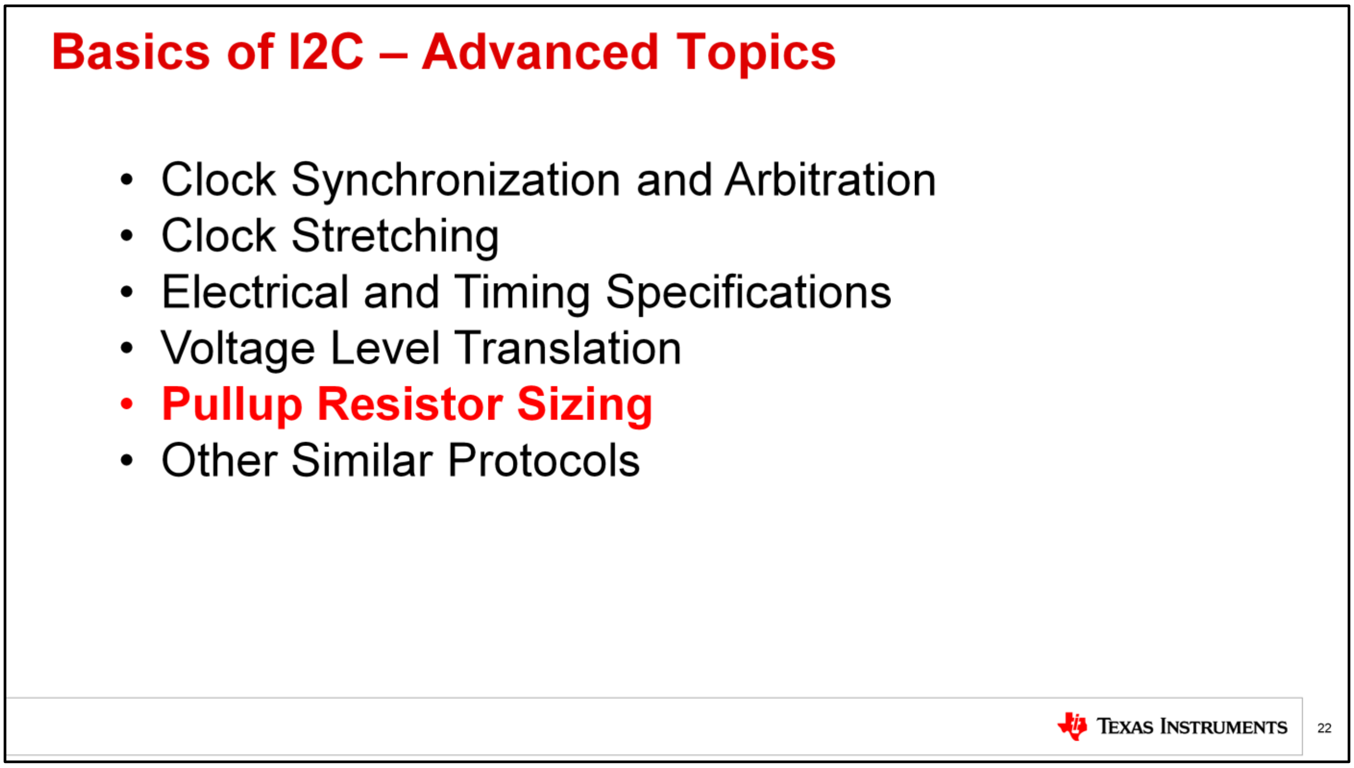


- Voltage level translators are a solution to mismatched supplies
- Requires two sets of pullups, one for each voltage level
- PCA9306 is a common I2C voltage level translator

TEXAS INSTRUMENTS 21

With mismatched supply voltages, the best option may be to use a special device to bridge the two supplies.

This figure shows an example of using an I2C voltage level translator to bridge the communication between two different supply voltages. There are two sets of pullups, one for each voltage level. As a common voltage translator, the PCA9306 allows for communication between different supply levels.

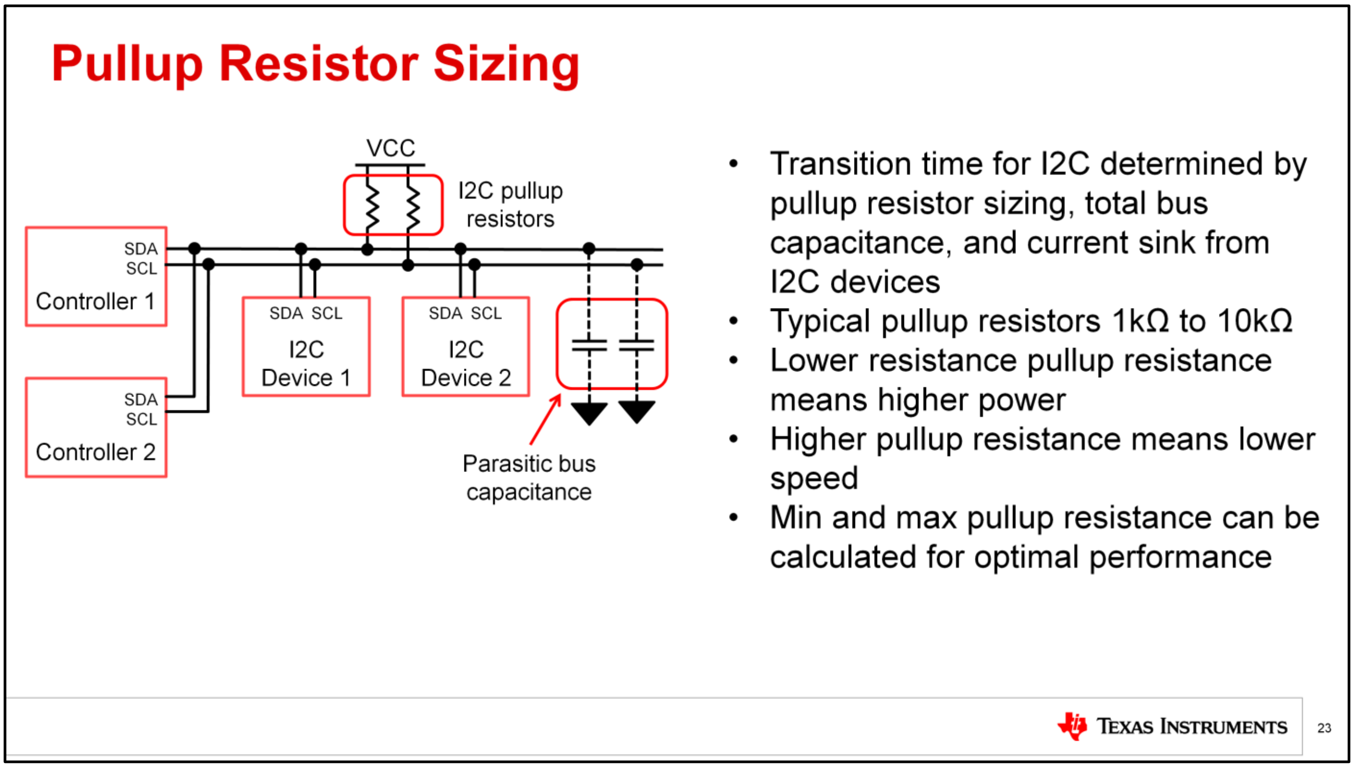


Basics of I2C – Advanced Topics

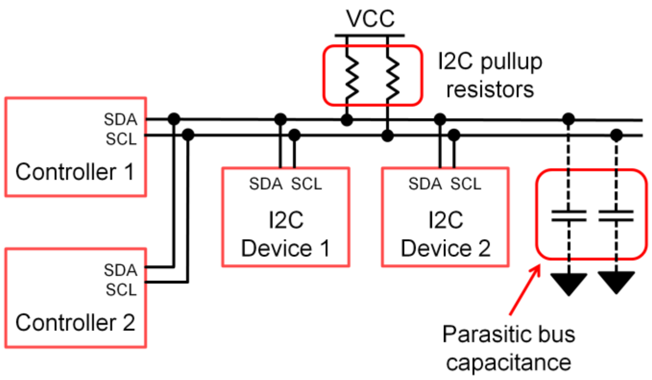
- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- **Pullup Resistor Sizing**
- Other Similar Protocols

TEXAS INSTRUMENTS 22

Another I2C advanced topic involves the pullup resistances required for I2C communication. To ensure that the bus speed is fast enough to meet the protocol bus speed, you may need to calculate values for the pullup resistances. In this section we'll show how to calculate a minimum and maximum value for the pullup resistances based on the I2C specifications.



Pullup Resistor Sizing

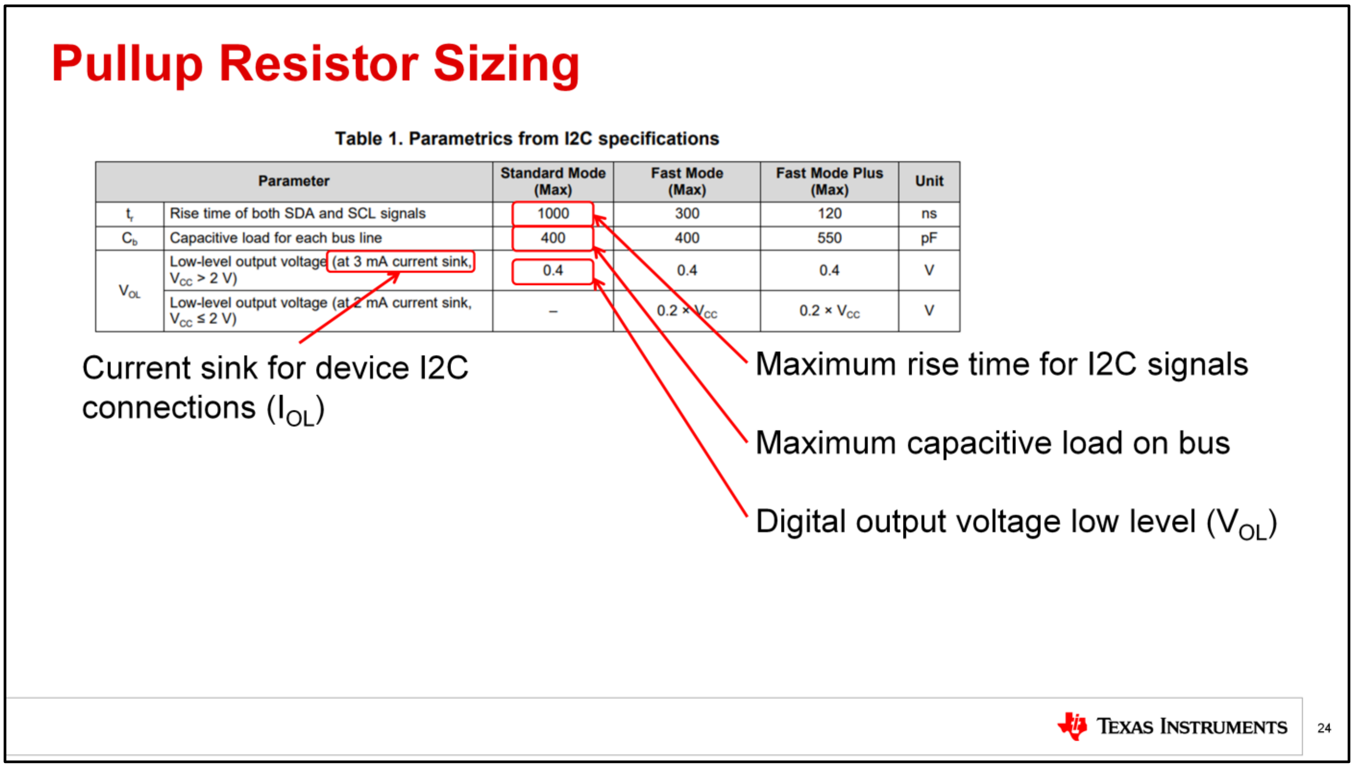


- Transition time for I2C determined by pullup resistor sizing, total bus capacitance, and current sink from I2C devices
- Typical pullup resistors 1kΩ to 10kΩ
- Lower resistance pullup resistance means higher power
- Higher pullup resistance means lower speed
- Min and max pullup resistance can be calculated for optimal performance

Texas Instruments 23

With the open-drain connections of SDA and SCL, transitions from these lines from high to low and from low to high are dependent on bus capacitance, current sink from the device connection, and the pullup resistor magnitude.

The normal pullup resistor recommendation is 1kOhms to 10kOhms. However, with higher resistances, the I2C communication is slower. With lower resistances, the I2C communication requires more power. Based on the several different parameters, we can calculate a minimum and maximum resistance for the I2C bus speed



Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, $V_{CC} > 2$ V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \le 2$ V)	–	0.2 × V_{CC}	0.2 × V_{CC}	V

Current sink for device I2C connections (I_{OL})

Maximum rise time for I2C signals

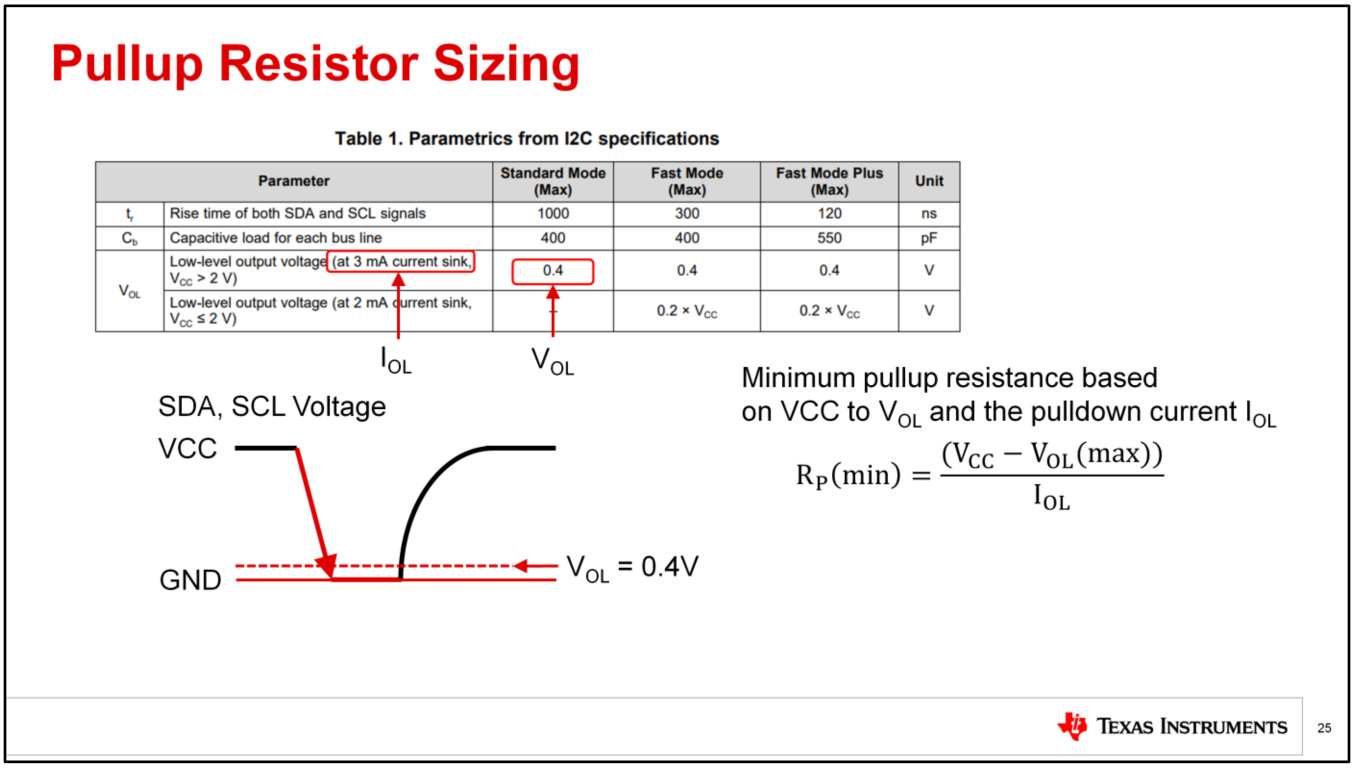
Maximum capacitive load on bus

Digital output voltage low level (V_{OL})

Texas Instruments 24

First, let's look at a table listing parametric characteristics from the I2C specifications.

We can focus on the specifications for the Standard mode. It lists the maximum rise time for the I2C bus, the maximum capacitive load on the bus, and it lists the low level output voltage listed as V_{OL}, which are given for different voltage levels for different speed modes of I2C. The table also lists the output current sunk by the device, which we'll call I_{OL}. We'll use all of these parameters to help determine the pullup resistance values.



Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, V_{CC} ≤ 2 V)	–	0.2 × V_{CC}	0.2 × V_{CC}	V

Minimum pullup resistance based on VCC to V_{OL} and the pulldown current I_{OL}

$$R_P(\min) = \frac{(V_{CC} - V_{OL}(\max))}{I_{OL}}$$

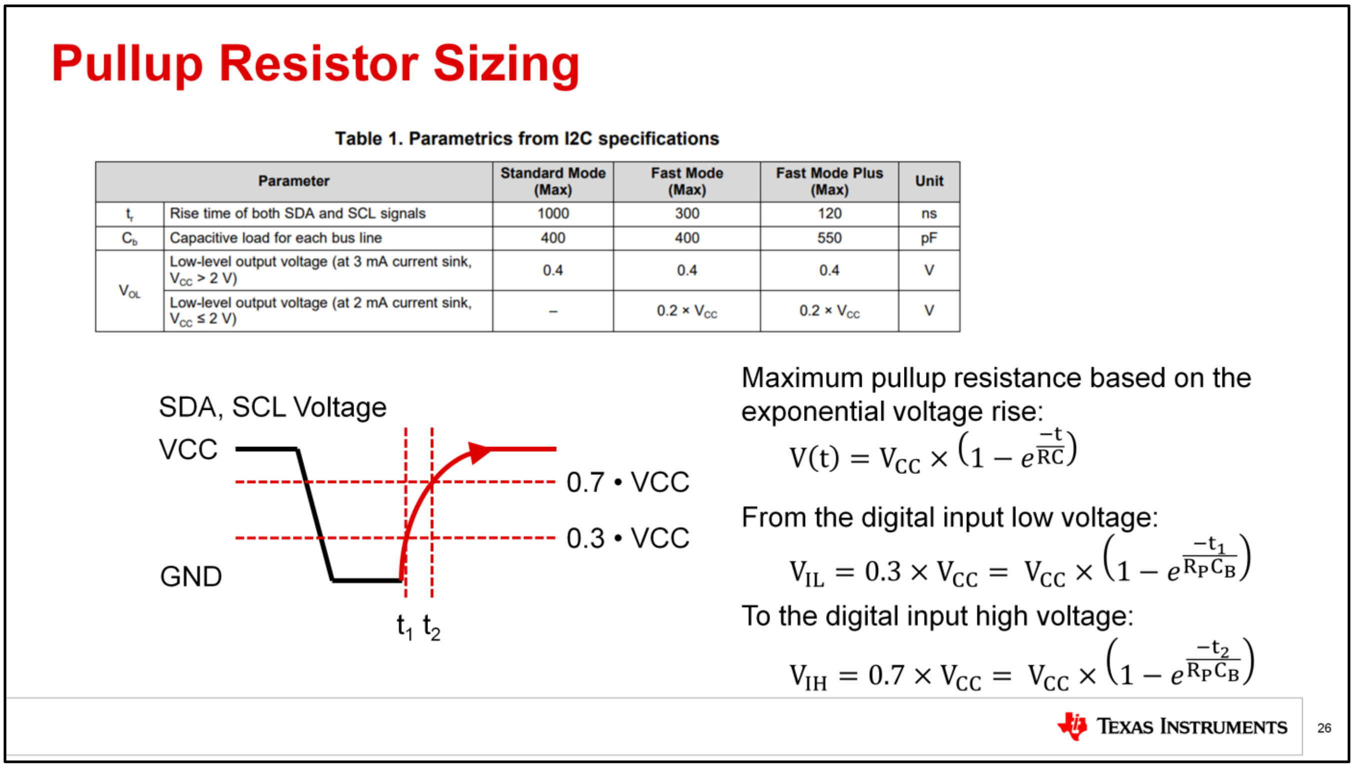
TEXAS INSTRUMENTS 25

Here we have an open drain connection to the I2C bus and we show the output waveform. The SDA and SCL bus transition low from the current pulling from the device.

The positive supply is connected to the bus voltage VCC when the device releases the SDA or SCL line. When active, the device drain pulls the bus line output to near ground. The output must drop to the output low-level voltage V_{OL}.

The device pulls the bus line low with current I_{OL}. Based on this current, we can calculate the minimum resistance needed for the pullup. If the resistance is smaller, the output current can't pull the output voltage of the bus low enough to be recognized as a digital low. This is shown in the equation in the bottom right:

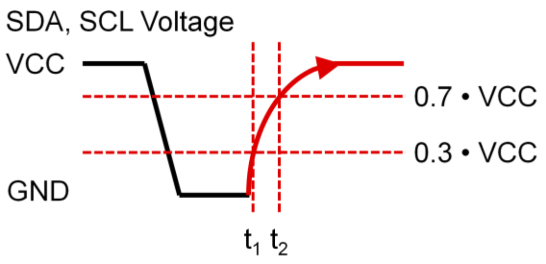
Rp(min) = $(V_{CC}-V_{OL}(\text{max}))/I_{OL}$



Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, $V_{CC} > 2$ V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \leq 2$ V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V



Maximum pullup resistance based on the exponential voltage rise:

$$V(t) = V_{CC} \times \left(1 - e^{\frac{-t}{RC}}\right)$$

From the digital input low voltage:

$$V_{IL} = 0.3 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t_1}{R_P C_B}}\right)$$

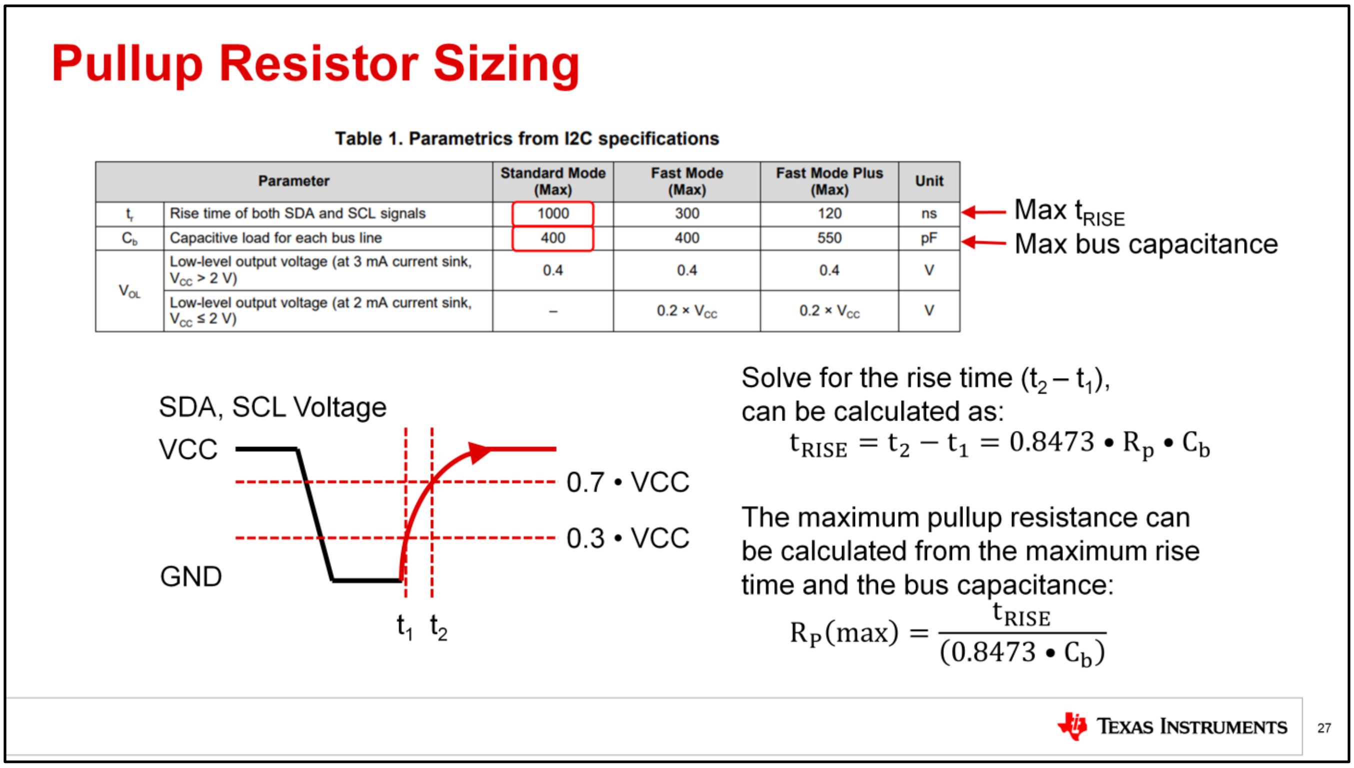
To the digital input high voltage:

$$V_{IH} = 0.7 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t_2}{R_P C_B}}\right)$$

Texas Instruments 26

Then, the open-drain connection releases the output current, and the resistors pull the bus connection high. The bus line output waveform shows an exponential rise. As the resistor pulls the voltage up from ground, the voltage settles based on the bus capacitance (Cb). The maximum pullup resistance is limited by the bus capacitance because of the I2C standard rise time specification. With a high resistance, the pullup output rises too slowly, and may not reach the logical high fast enough.

The equation for the exponential rise over time is shown with the pull up resistance. The rise time is based on the transition from the digital input low voltage of 0.3 times the supply voltage to the digital input high voltage of 0.7 times the supply voltage.



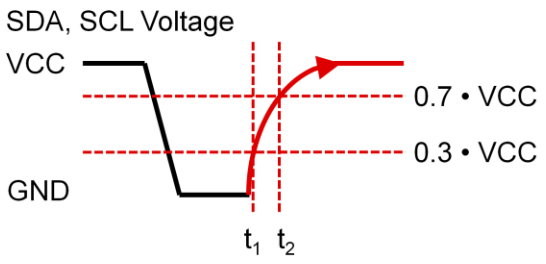
Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, V_{CC} ≤ 2 V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V

Max t_{RISE}

Max bus capacitance



Solve for the rise time ($t_2 - t_1$), can be calculated as:

$$t_{RISE} = t_2 - t_1 = 0.8473 \bullet R_p \bullet C_b$$

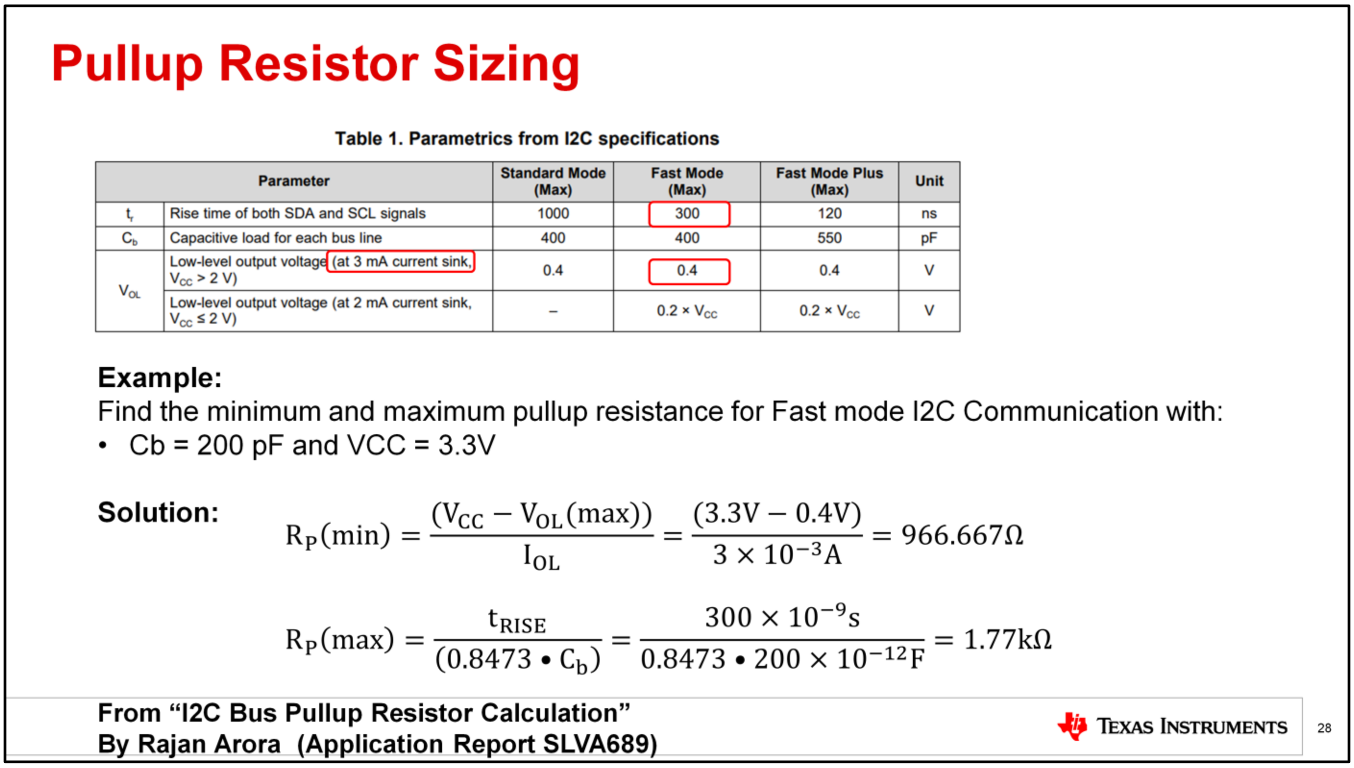
The maximum pullup resistance can be calculated from the maximum rise time and the bus capacitance:

$$R_P(\max) = \frac{t_{RISE}}{(0.8473 \bullet C_b)}$$

TEXAS INSTRUMENTS 27

From the exponential equations, the rise time can be solved in terms of the maximum pullup resistance and the bus capacitance.

Again, the rise time is based on the bus line's rise time from 0.3 times VCC to 0.7 times VCC.



Pullup Resistor Sizing

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, $V_{CC} > 2$ V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \le 2$ V)	–	$0.2 \times V_{CC}$	$0.2 \times V_{CC}$	V

Example:
Find the minimum and maximum pullup resistance for Fast mode I2C Communication with:
- Cb = 200 pF and VCC = 3.3V

Solution:

$$R_P(\min) = \frac{(V_{CC} - V_{OL}(\max))}{I_{OL}} = \frac{(3.3\text{V} - 0.4\text{V})}{3 \times 10^{-3}\text{A}} = 966.667\Omega$$

$$R_P(\max) = \frac{t_{RISE}}{(0.8473 \bullet C_b)} = \frac{300 \times 10^{-9}\text{s}}{0.8473 \bullet 200 \times 10^{-12}\text{F}} = 1.77\text{k}\Omega$$

From "I2C Bus Pullup Resistor Calculation"
By Rajan Arora (Application Report SLVA689)

Texas Instruments 28

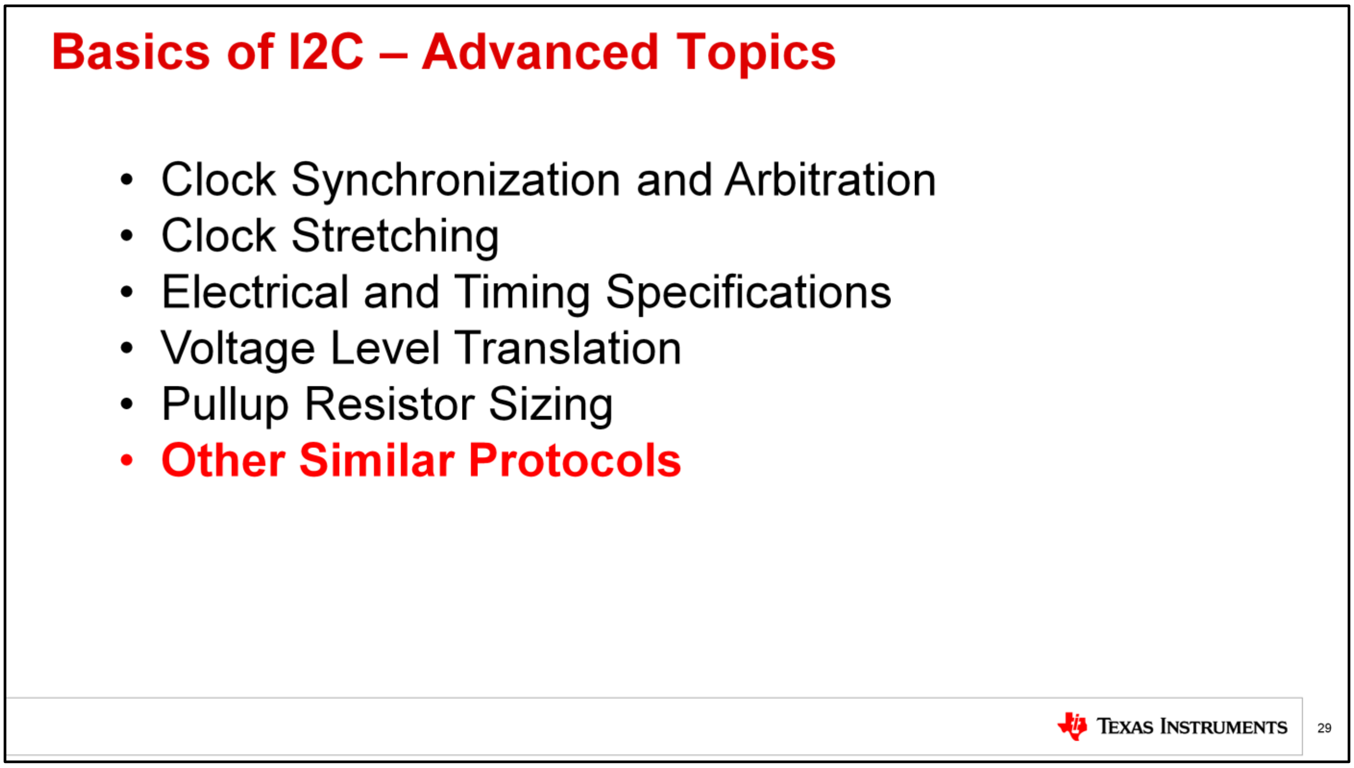
Using the equations developed over the previous few slides, we can calculate the minimum and maximum pullup resistance for a fast mode I2C communication bus. In this example, we can calculate the minimum and maximum pullup resistance with a 200pF bus capacitance and supply voltage of 3.3V.

Solving for the minimum pullup resistance, subtract the output low voltage of 0.4V from the supply voltage of 3.3V. Then divide by the current pulled by the bus line of 3mA. This results in 967 Ohms.

Then solve for the maximum bus resistance. Take the rise time of 300 nanoseconds and divide by the quantity of 0.8473 times 200 picoFarads. This gives a maximum resistance of 1.77 kOhms.

This may appear to be a narrow range. However, this is because we've designed the pullup resistor sizing to operate with the high bus capacitance of 200pF. If the design could ensure a lower bus capacitance, the maximum resistance could be increased, reducing the power dissipated on the I2C bus.

For a more detailed description of I2C pullup resistor calculations see Application Report SLVA689, "I2C Bus Pullup Resistor Calculation" By Rajan Arora.



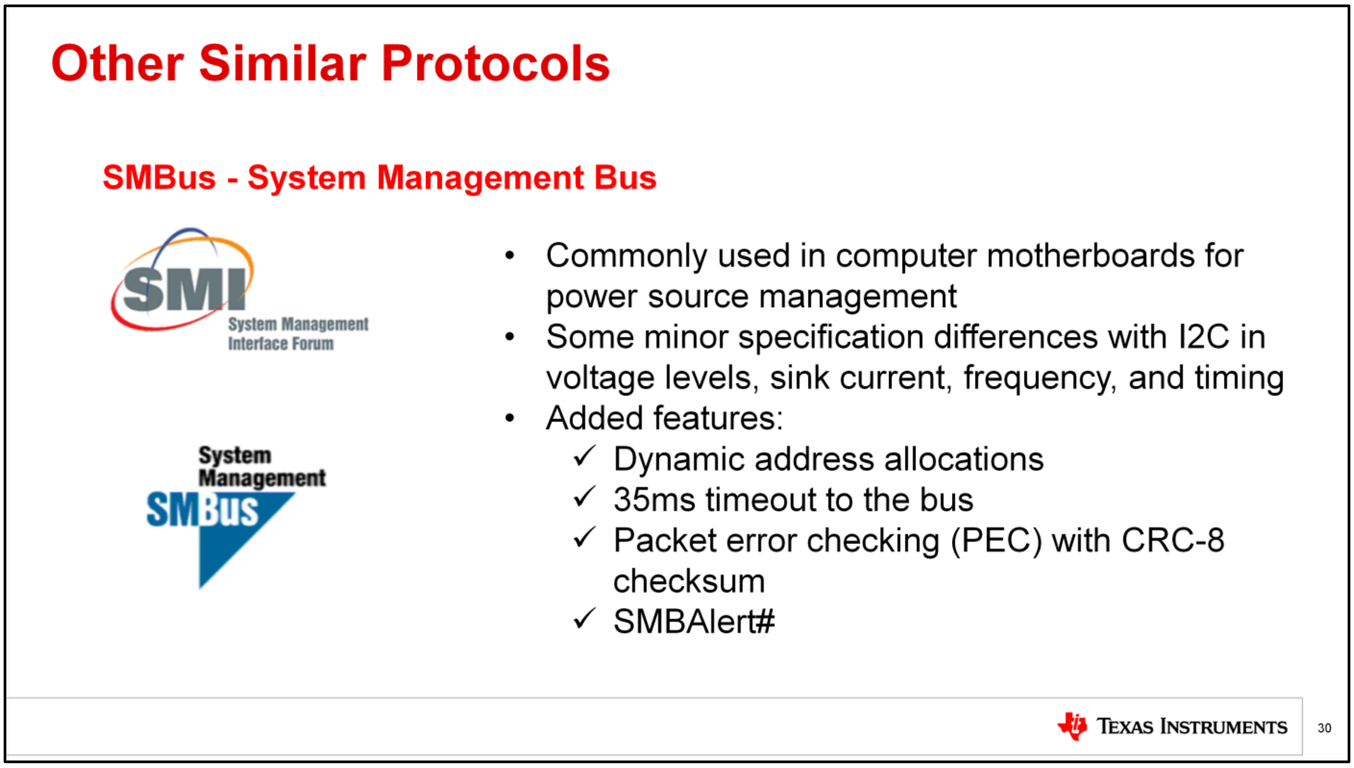
Basics of I2C – Advanced Topics

- Clock Synchronization and Arbitration
- Clock Stretching
- Electrical and Timing Specifications
- Voltage Level Translation
- Pullup Resistor Sizing
- **Other Similar Protocols**

TEXAS INSTRUMENTS 29

The I2C specification discusses several other communications protocols based on I2C. These other protocols may be similar and compatible with I2C communication and may be used for specific applications. They may also have defined sets of commands and application-specific extensions for their systems.

Just as in the I2C specification, we'll briefly describe these other protocols, but we'll leave it to you to dig deeper into their systems, applications, and uses.



Other Similar Protocols

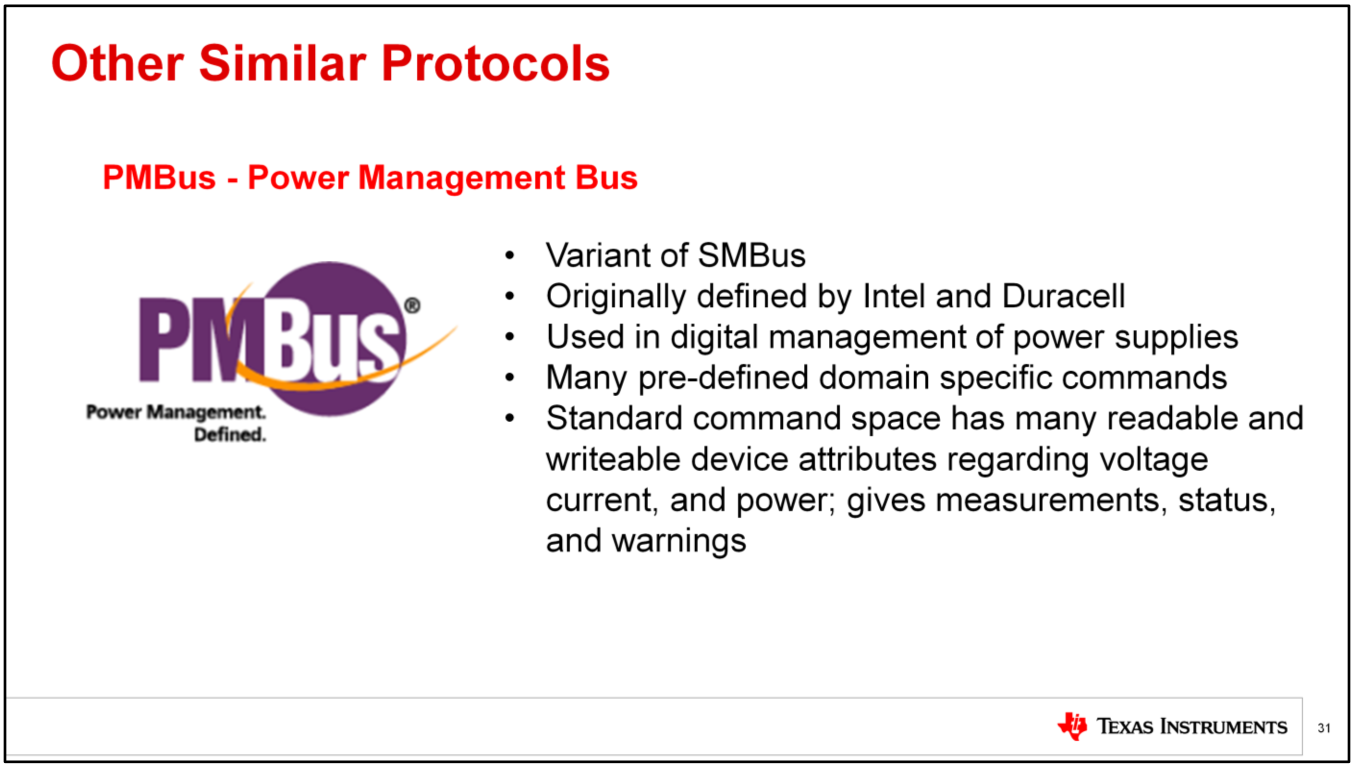
SMBus - System Management Bus

- Commonly used in computer motherboards for power source management
- Some minor specification differences with I2C in voltage levels, sink current, frequency, and timing
- Added features:
  - ✓ Dynamic address allocations
  - ✓ 35ms timeout to the bus
  - ✓ Packet error checking (PEC) with CRC-8 checksum
  - ✓ SMBAlert#

Texas Instruments 30

The first of these similar protocols is the System Management bus or SMBus. It is commonly used in servers and computer motherboards for power source management. It's very similar to I2C in the communication protocol, and can be understood by an I2C controller.

This protocol has some additional features in comparison to I2C. First, it can dynamically set addresses, allowing for quick communications at the startup of a system. Also it has a 35ms timeout on the bus which prevents one device from indefinitely tying up the bus. It also has a packet error checking for error detection in data communication. There is also an additional line called SMBAlert that is used by target devices as an interrupt to tell the controller about certain events detected by the target device.



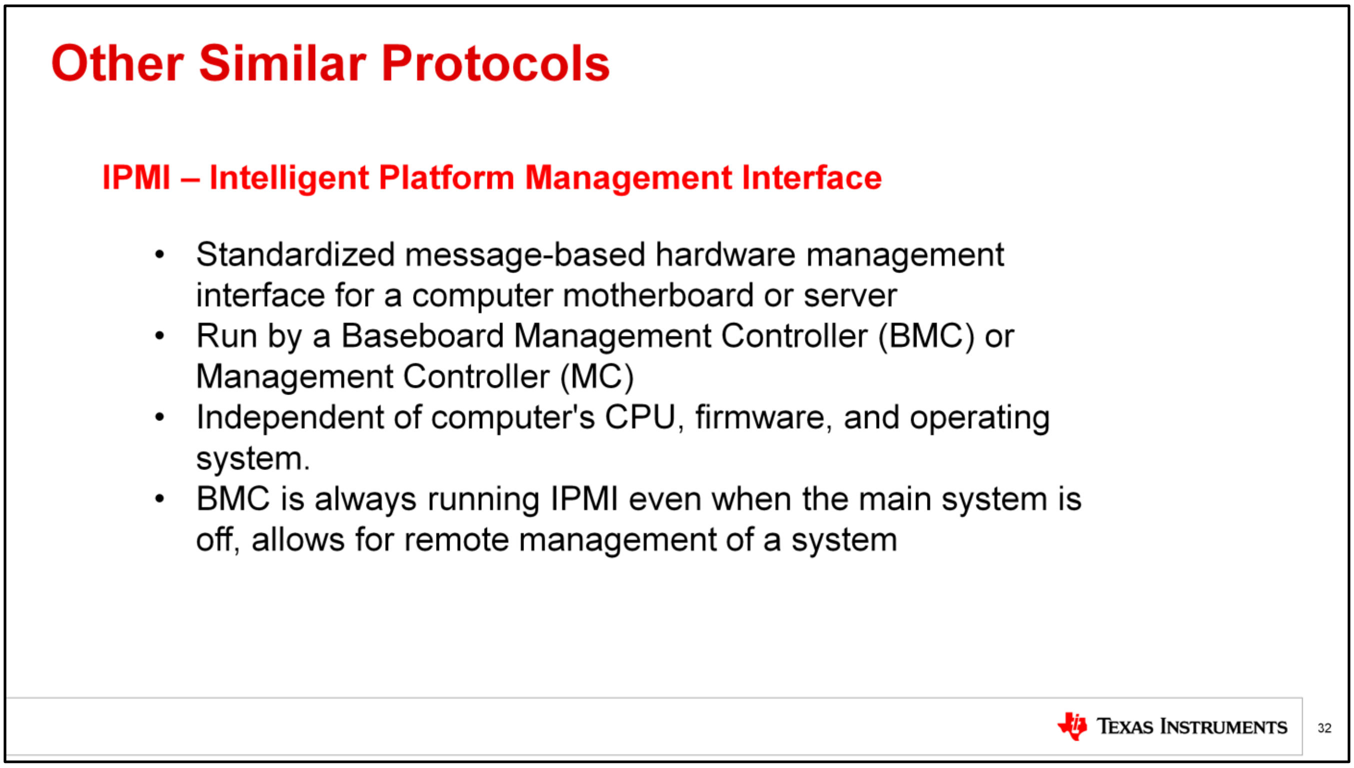
Other Similar Protocols

PMBus - Power Management Bus

- Variant of SMBus
- Originally defined by Intel and Duracell
- Used in digital management of power supplies
- Many pre-defined domain specific commands
- Standard command space has many readable and writeable device attributes regarding voltage current, and power; gives measurements, status, and warnings

Texas Instruments 31

PMBus is basically a variant of SMBus defined by Intel and Duracell. It is used in the digital management of power supplies. This protocol also defines specific commands to retrieve data about voltage, current, and power in the system.



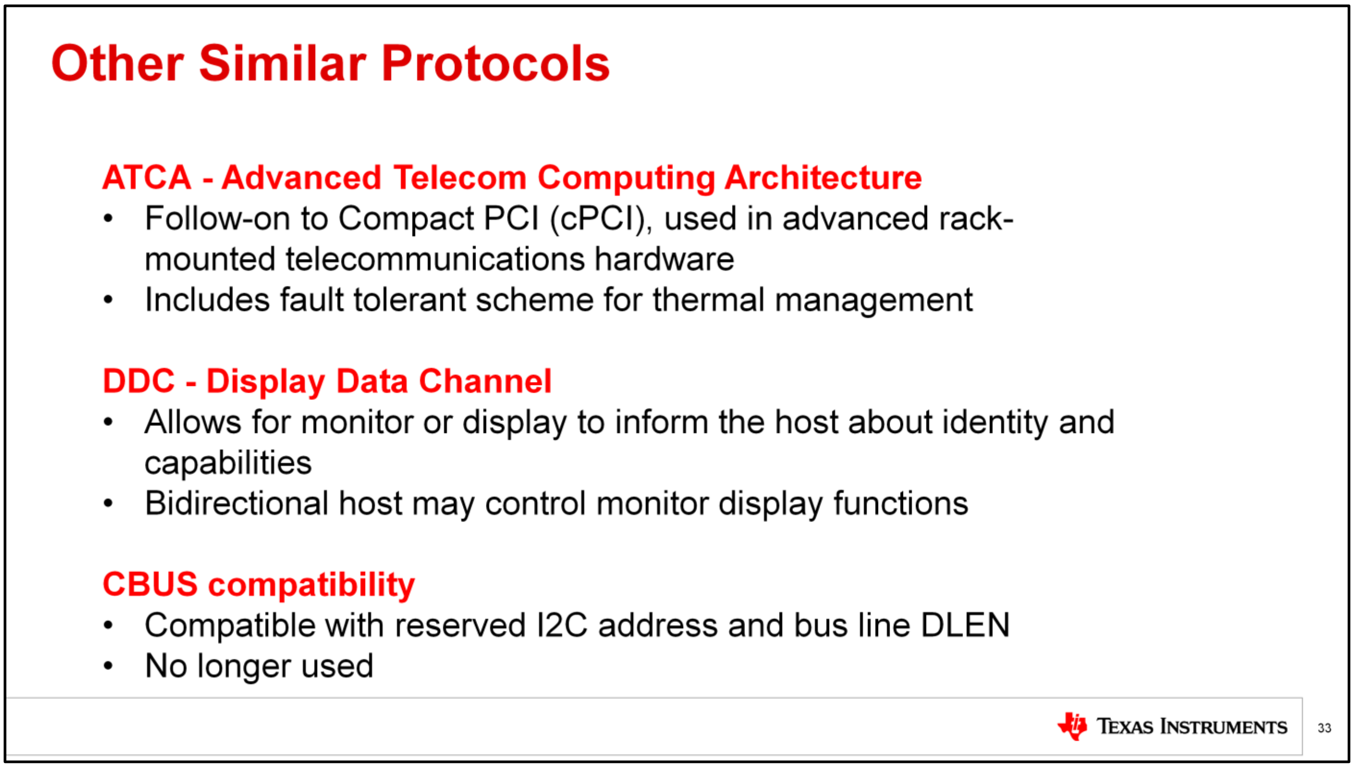
Other Similar Protocols

IPMI – Intelligent Platform Management Interface

- Standardized message-based hardware management interface for a computer motherboard or server
- Run by a Baseboard Management Controller (BMC) or Management Controller (MC)
- Independent of computer's CPU, firmware, and operating system.
- BMC is always running IPMI even when the main system is off, allows for remote management of a system

TEXAS INSTRUMENTS 32

IPMI is another I2C based protocol used by a baseboard management controller or BMC. It uses a standardized message based interface for a computer motherboard or server. The BMC is always running even when the main system is off. This allows for operation, measurement, and remote management of a system.


Other Similar Protocols

ATCA - Advanced Telecom Computing Architecture
- Follow-on to Compact PCI (cPCI), used in advanced rack-mounted telecommunications hardware
- Includes fault tolerant scheme for thermal management

DDC - Display Data Channel
- Allows for monitor or display to inform the host about identity and capabilities
- Bidirectional host may control monitor display functions

CBUS compatibility
- Compatible with reserved I2C address and bus line DLEN
- No longer used

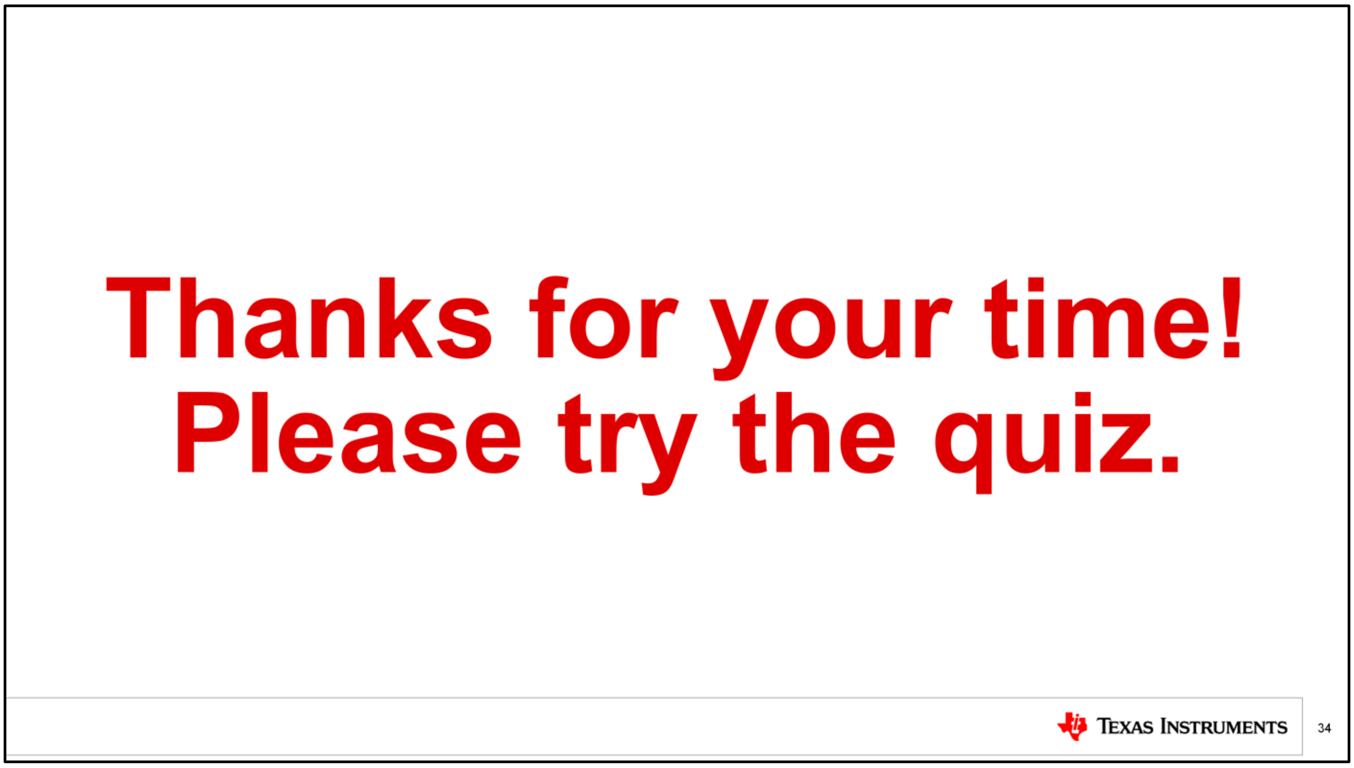
TEXAS INSTRUMENTS 33

There are several other similar protocols discussed in the I2C specifications.

ATCA is a follow-on to Compact PCI and used in rack mounted telecom hardware.

DDC is a monitor or display information protocol that is used by hosts for control of display functions.

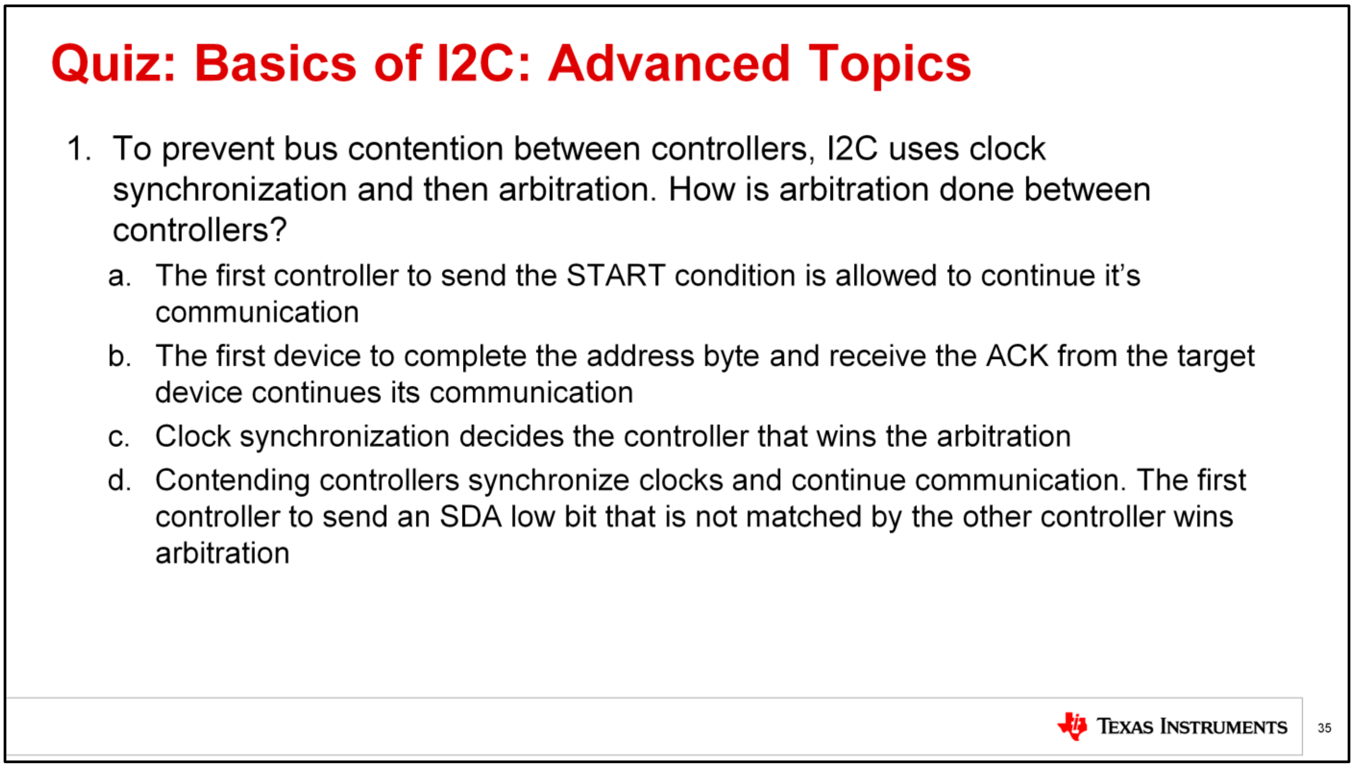
Finally CBUS is another protocol that is derived from I2C. As mentioned in the reserved address section, it is no longer used.



Thanks for your time!
Please try the quiz.

Texas Instruments 34

That concludes this video – thank you for watching! Please try the quiz to check your understanding of this video's content.

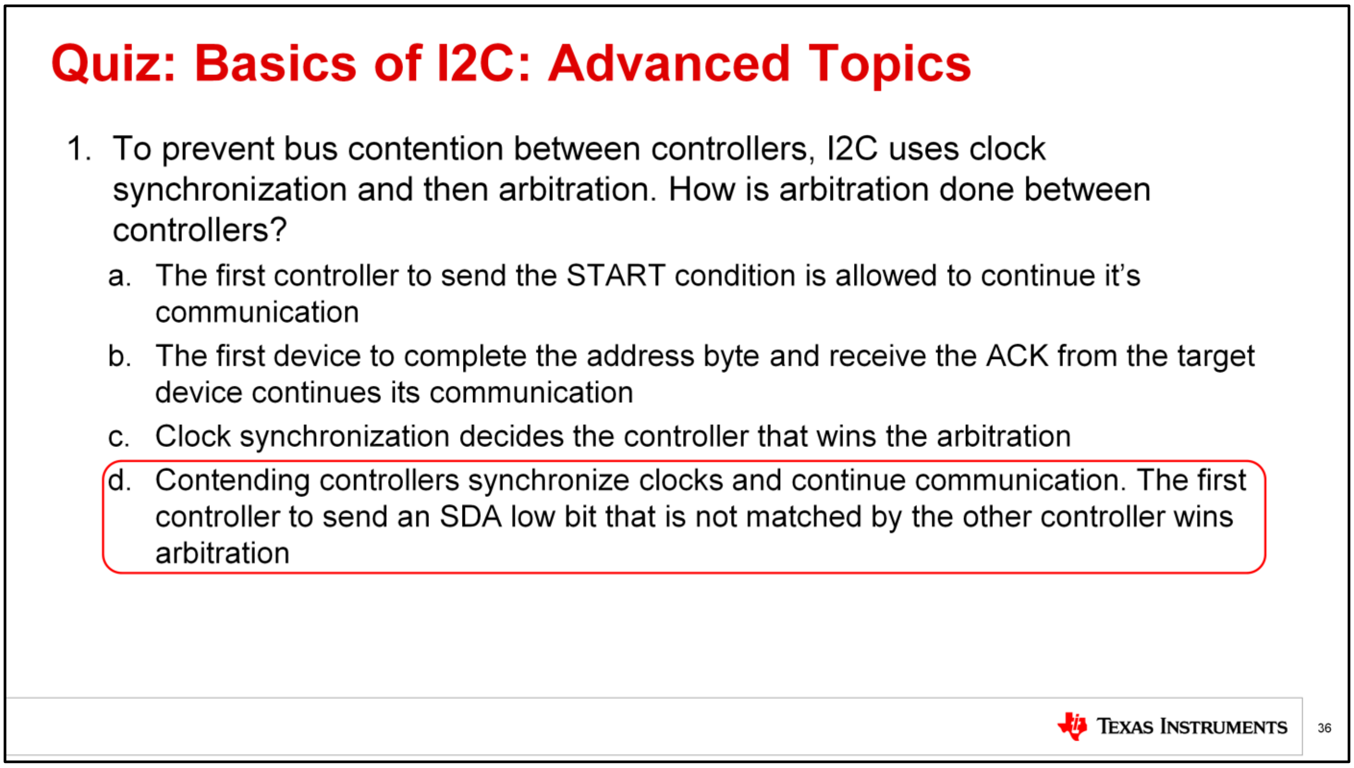


Quiz: Basics of I2C: Advanced Topics

1. To prevent bus contention between controllers, I2C uses clock synchronization and then arbitration. How is arbitration done between controllers?
   a. The first controller to send the START condition is allowed to continue it's communication
   b. The first device to complete the address byte and receive the ACK from the target device continues its communication
   c. Clock synchronization decides the controller that wins the arbitration
   d. Contending controllers synchronize clocks and continue communication. The first controller to send an SDA low bit that is not matched by the other controller wins arbitration

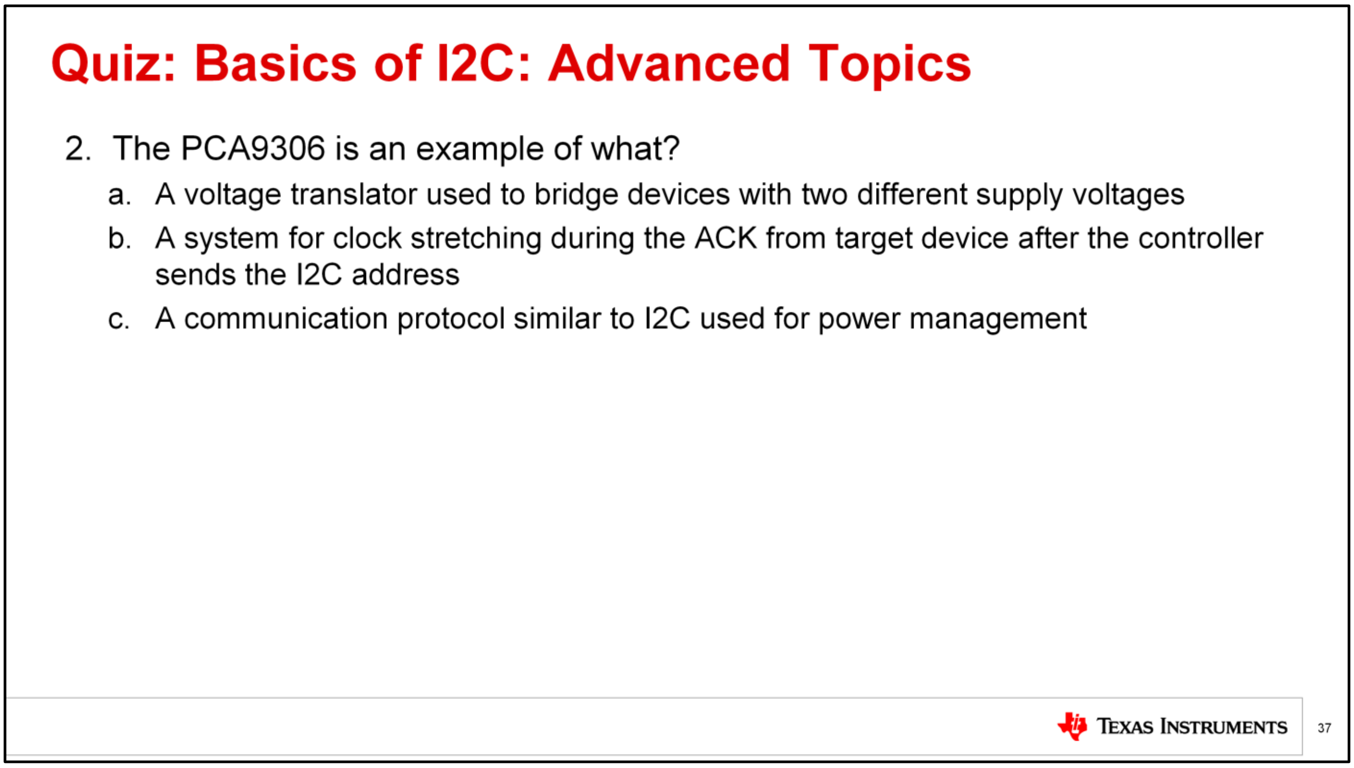
TEXAS INSTRUMENTS

35



Quiz: Basics of I2C: Advanced Topics

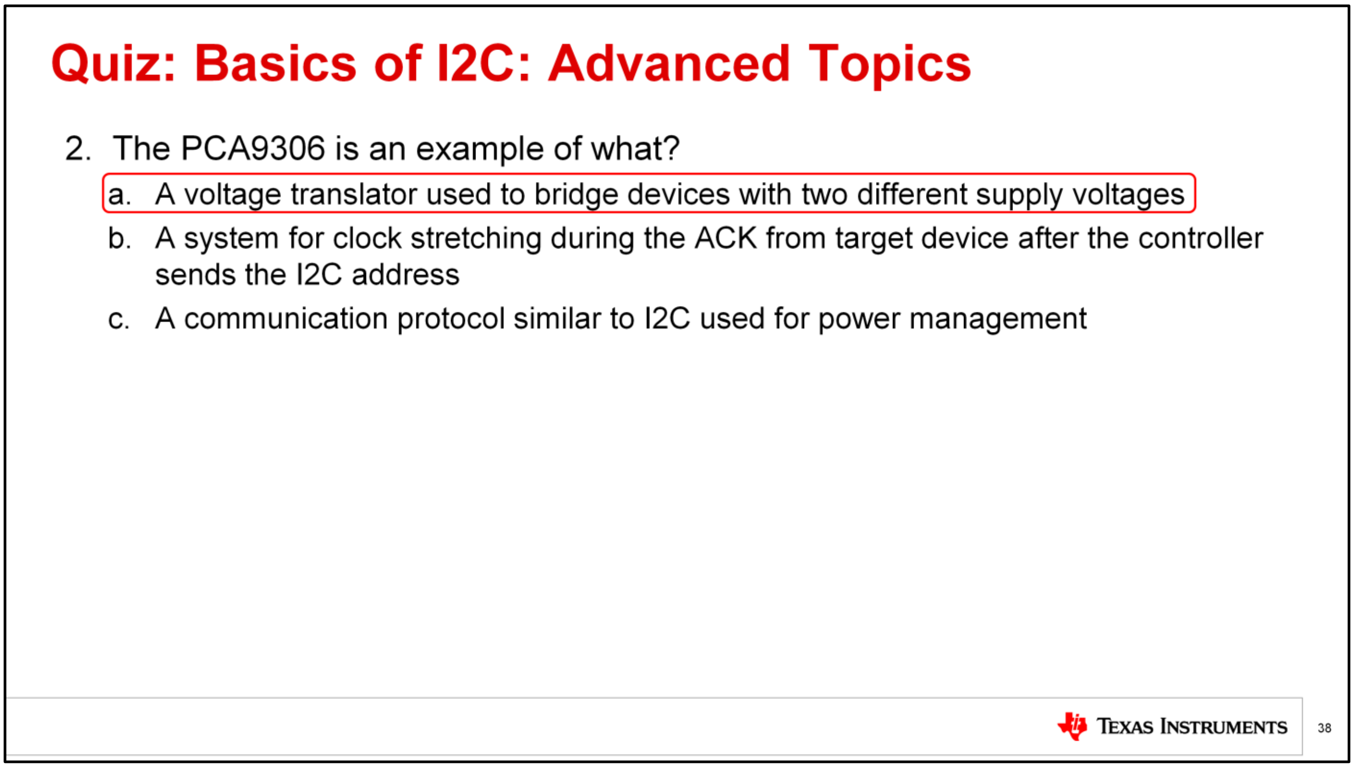
1. To prevent bus contention between controllers, I2C uses clock synchronization and then arbitration. How is arbitration done between controllers?
   a. The first controller to send the START condition is allowed to continue it's communication
   b. The first device to complete the address byte and receive the ACK from the target device continues its communication
   c. Clock synchronization decides the controller that wins the arbitration
   [✓ d. Contending controllers synchronize clocks and continue communication. The first controller to send an SDA low bit that is not matched by the other controller wins arbitration]



Quiz: Basics of I2C: Advanced Topics

2. The PCA9306 is an example of what?
   a. A voltage translator used to bridge devices with two different supply voltages
   b. A system for clock stretching during the ACK from target device after the controller sends the I2C address
   c. A communication protocol similar to I2C used for power management

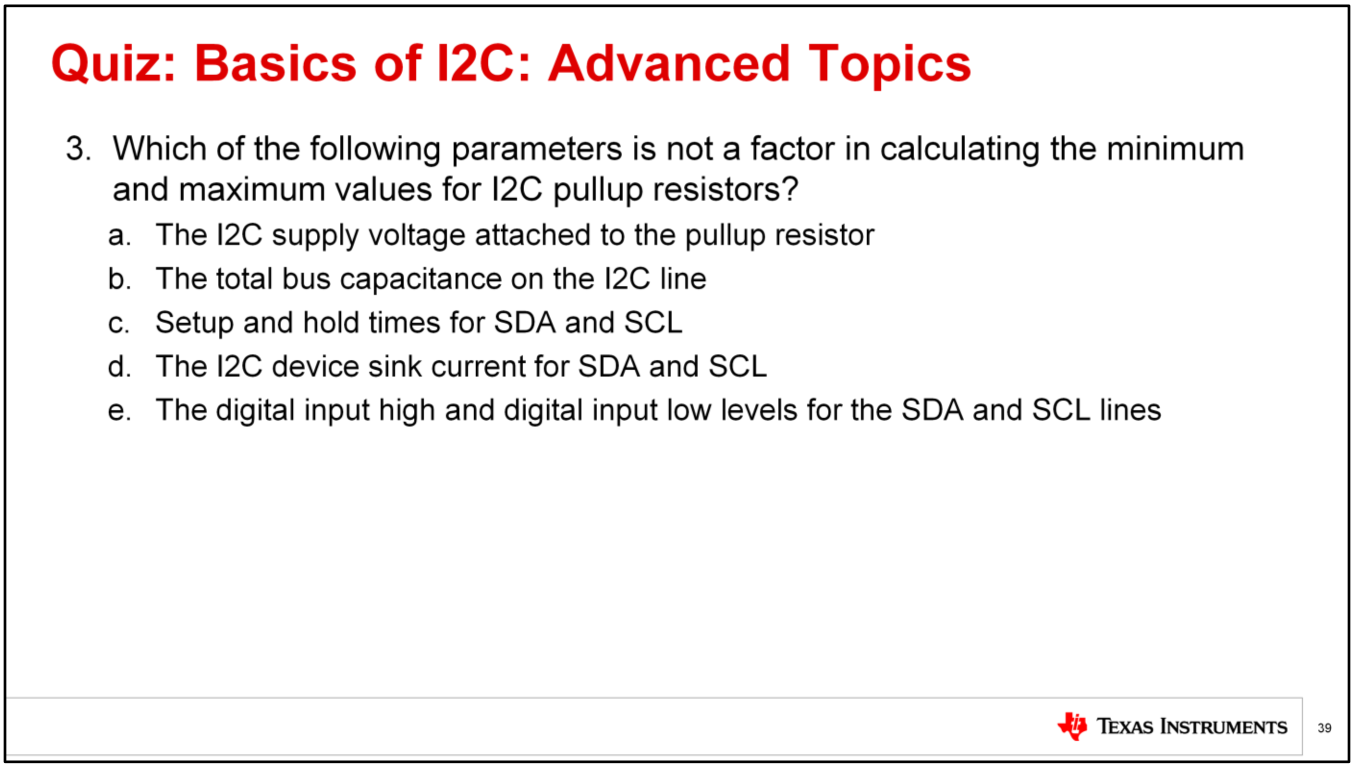
TEXAS INSTRUMENTS 37



Quiz: Basics of I2C: Advanced Topics

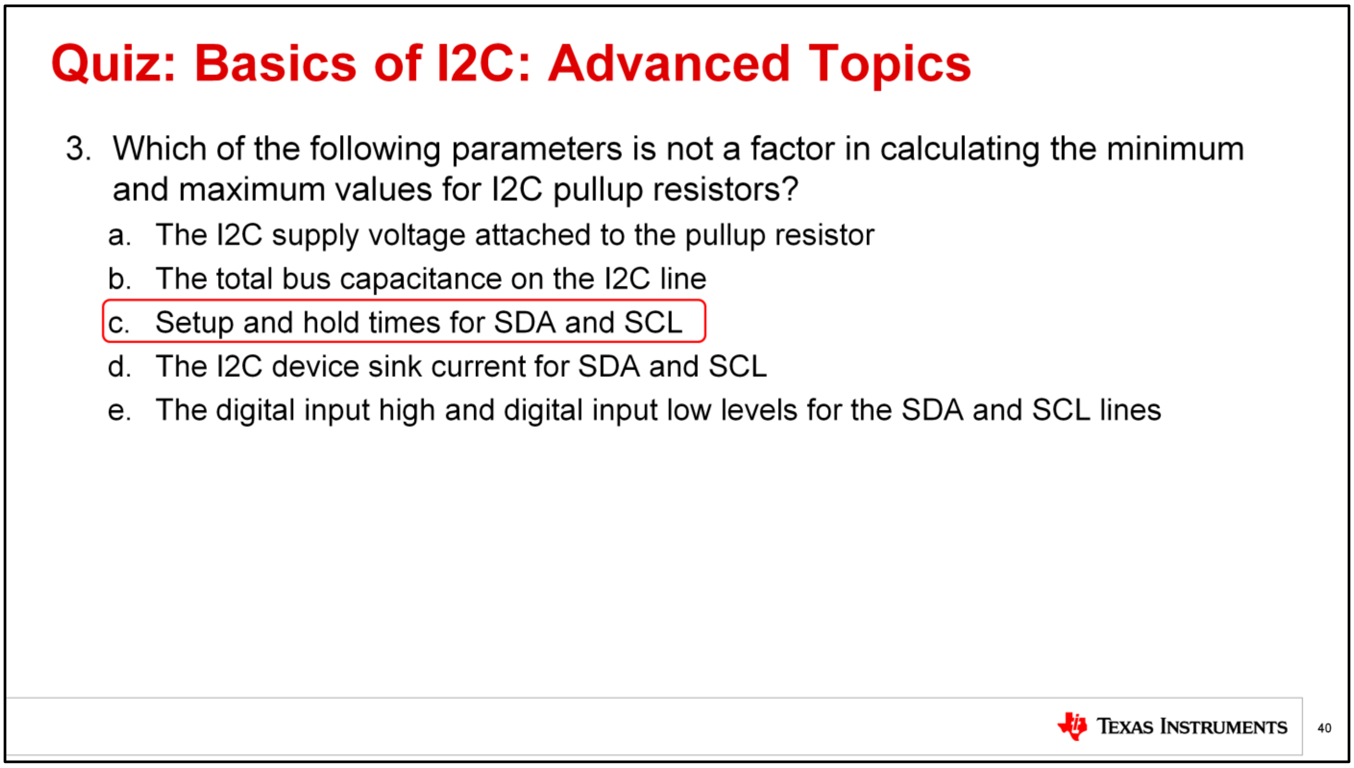
2. The PCA9306 is an example of what?
   - [✓ a. A voltage translator used to bridge devices with two different supply voltages]
   - b. A system for clock stretching during the ACK from target device after the controller sends the I2C address
   - c. A communication protocol similar to I2C used for power management

Texas Instruments



Quiz: Basics of I2C: Advanced Topics

3. Which of the following parameters is not a factor in calculating the minimum and maximum values for I2C pullup resistors?
   a. The I2C supply voltage attached to the pullup resistor
   b. The total bus capacitance on the I2C line
   c. Setup and hold times for SDA and SCL
   d. The I2C device sink current for SDA and SCL
   e. The digital input high and digital input low levels for the SDA and SCL lines



Quiz: Basics of I2C: Advanced Topics

3. Which of the following parameters is not a factor in calculating the minimum and maximum values for I2C pullup resistors?
   - a. The I2C supply voltage attached to the pullup resistor
   - b. The total bus capacitance on the I2C line
   - [✓ c. Setup and hold times for SDA and SCL]
   - d. The I2C device sink current for SDA and SCL
   - e. The digital input high and digital input low levels for the SDA and SCL lines

Thanks for your ti