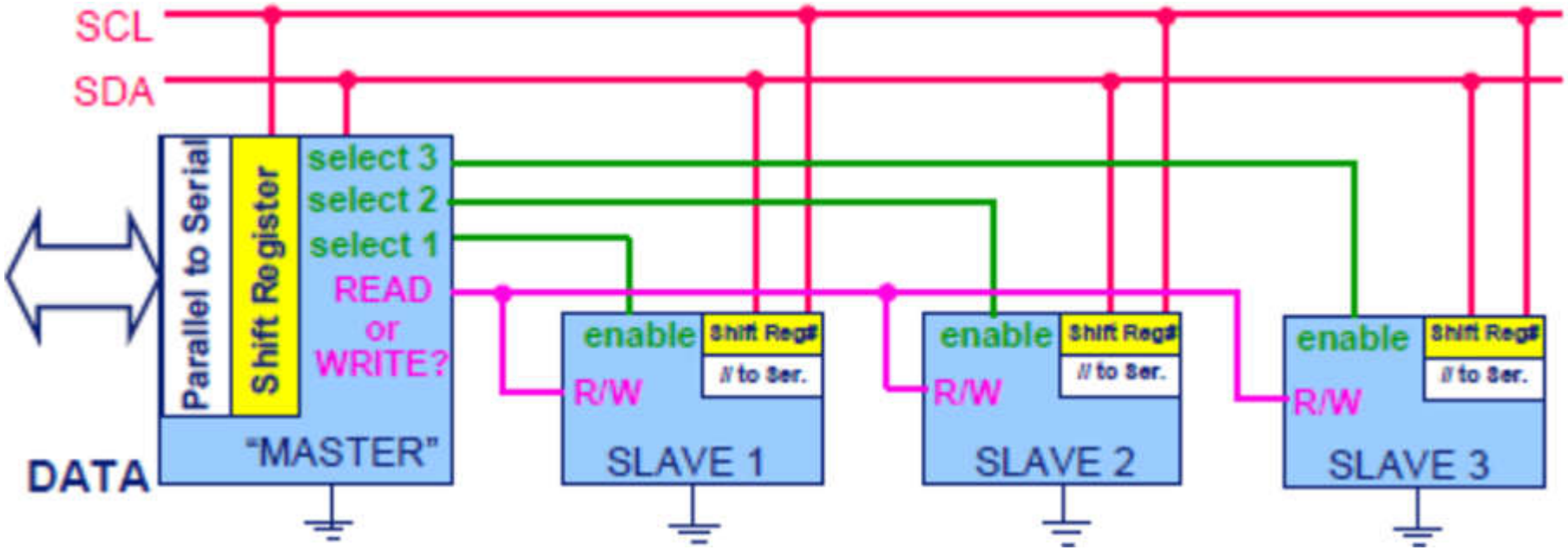
I2C Bus (Inter-IC Bus)
For SW Engineers

Tuan Nguyen-viet

General Concept for Serial Communications

- The general concept of serial bus communication is shown in Figure below.
- The most popular serial bus communication protocols available (the Year 2009) in the market are:
  - SPI, UART, **I2C**, CAN, USB, IEE1394, and so on.



I^2C or I2C

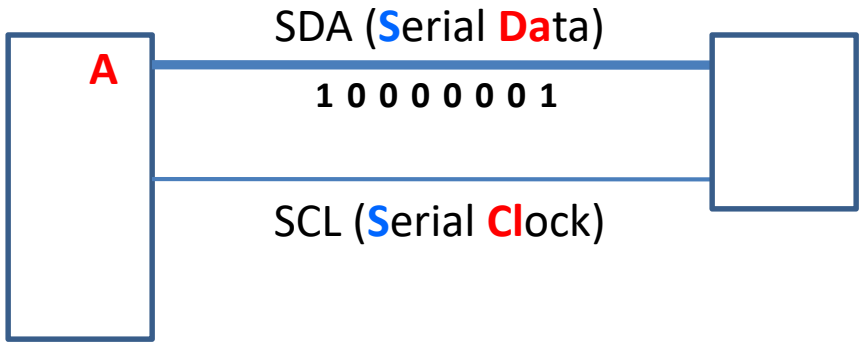
- Pronounced
  - “eye-squared-see”, (The ² symbol is a superscript 2)
  - "I-squared-C"
- Sometimes called
  - “eye-two-see”
  - "I-two-C”

- Summary:
  - pronounced "I-squared-C" or "I-two-C”

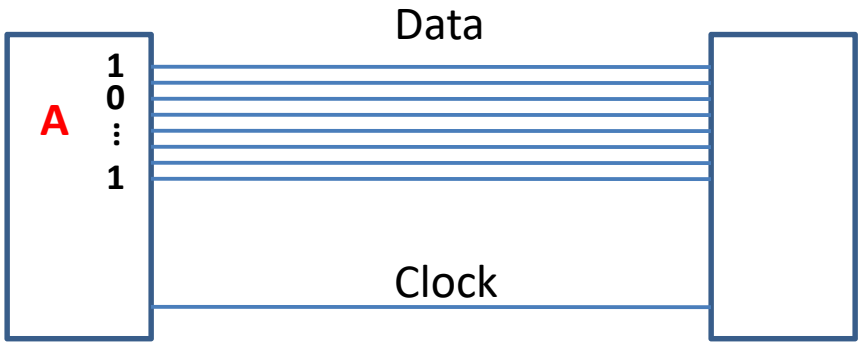
I2C – 'De Facto' Application

- The I2C bus is a **de facto** **world standard** that is now
  - implemented in over **1000** different **IC**s
    - manufactured by more than **50 companies**. (the Year 2011).

Many Parallel Pins vs 2 Pins



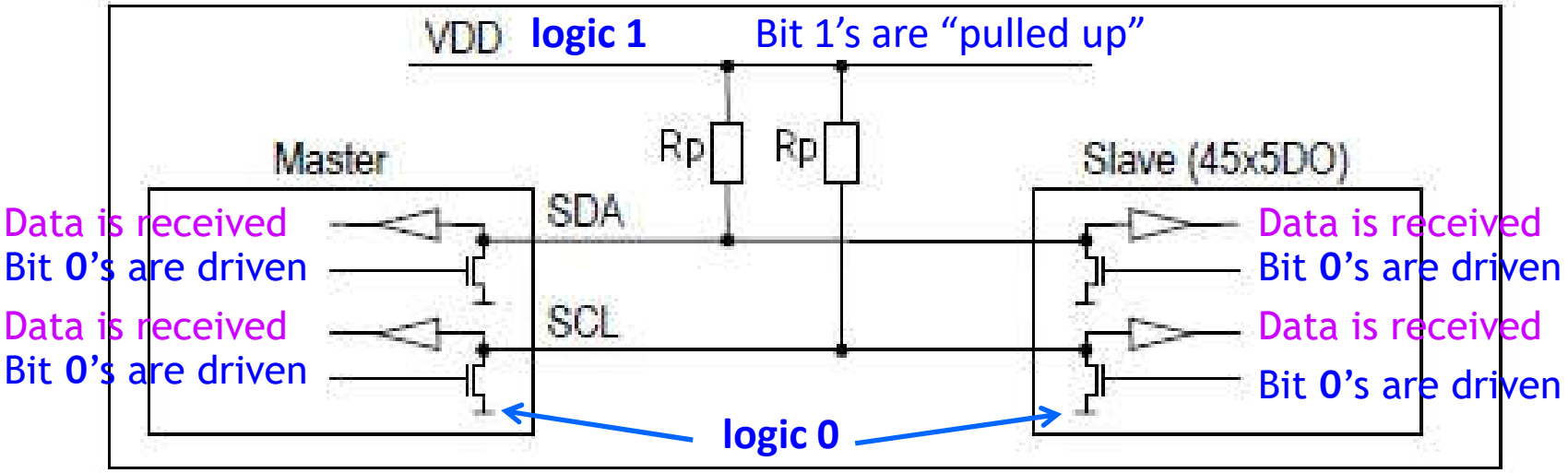
I2C Serial Interface



Parallel Interface

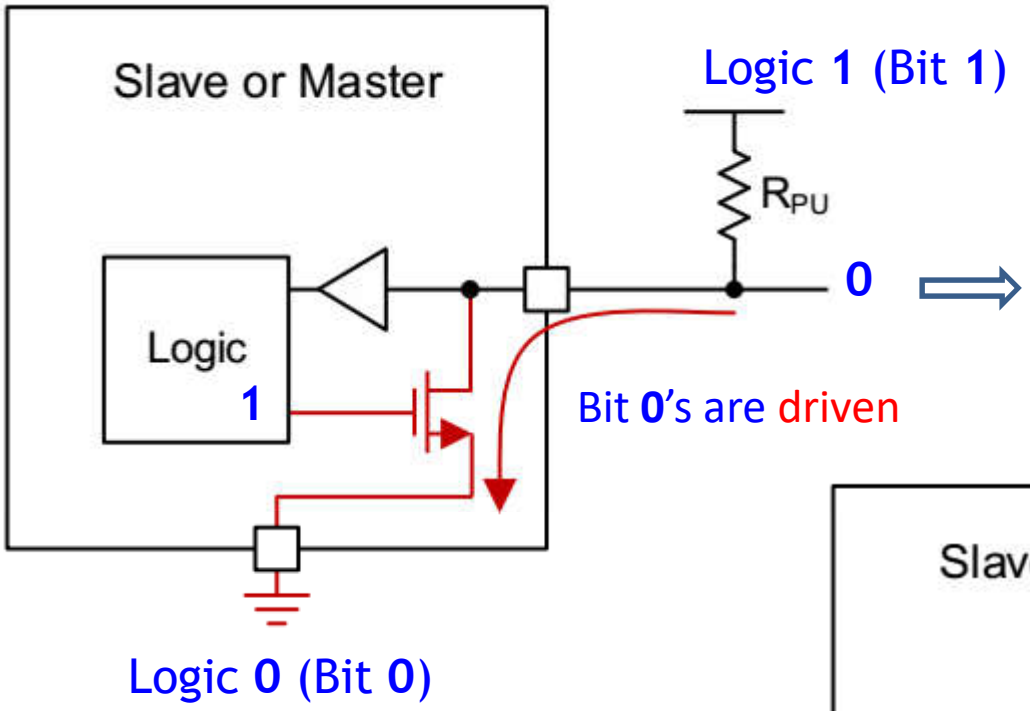
ASCII Alphabet			
A	1000001	N	1001110
B	1000010	O	1001111
C	1000011	P	1010000
D	1000100	Q	1010001
E	1000101	R	1010010
F	1000110	S	1010011
G	1000111	T	1010100
H	1001000	U	1010101
I	1001001	V	1010110
J	1001010	W	1010111
K	1001011	X	1011000
L	1001100	Y	1011001
M	1001101	Z	1011010

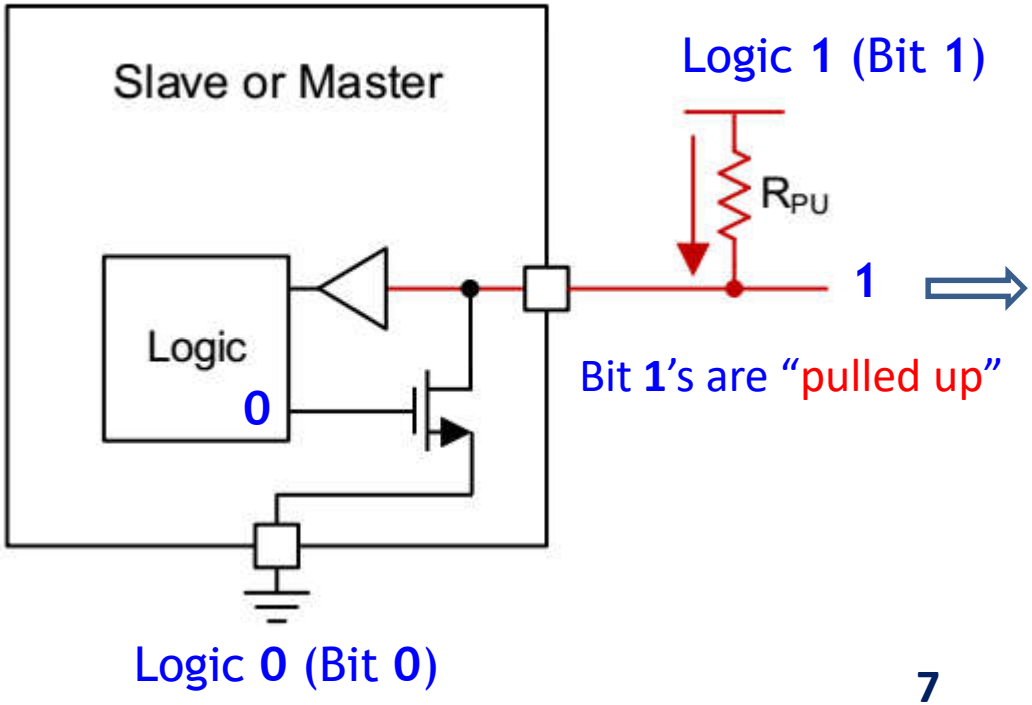
How can a Device transfer or receive on 2 wires?



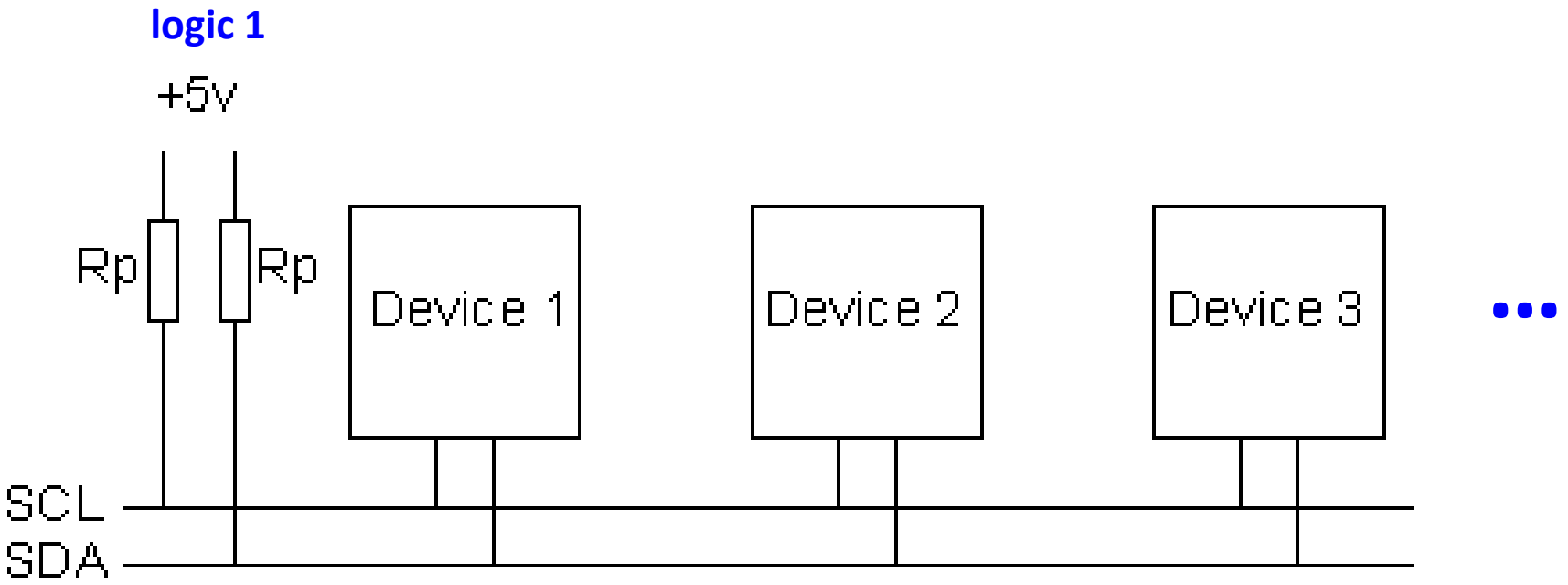
- Pull-ups and high-impedance mode pins
  - Wires default to being "**on**" or **high** (**logic 1**),
    - Any Device can make a wire go "**off**" or **low** (**logic 0**).
  - This is super clever.
    - SPI and UART can't do this.

How can a Device transfer or receive on 2 wires? (2)





How can a Device transfer or receive on 2 wires? (3)

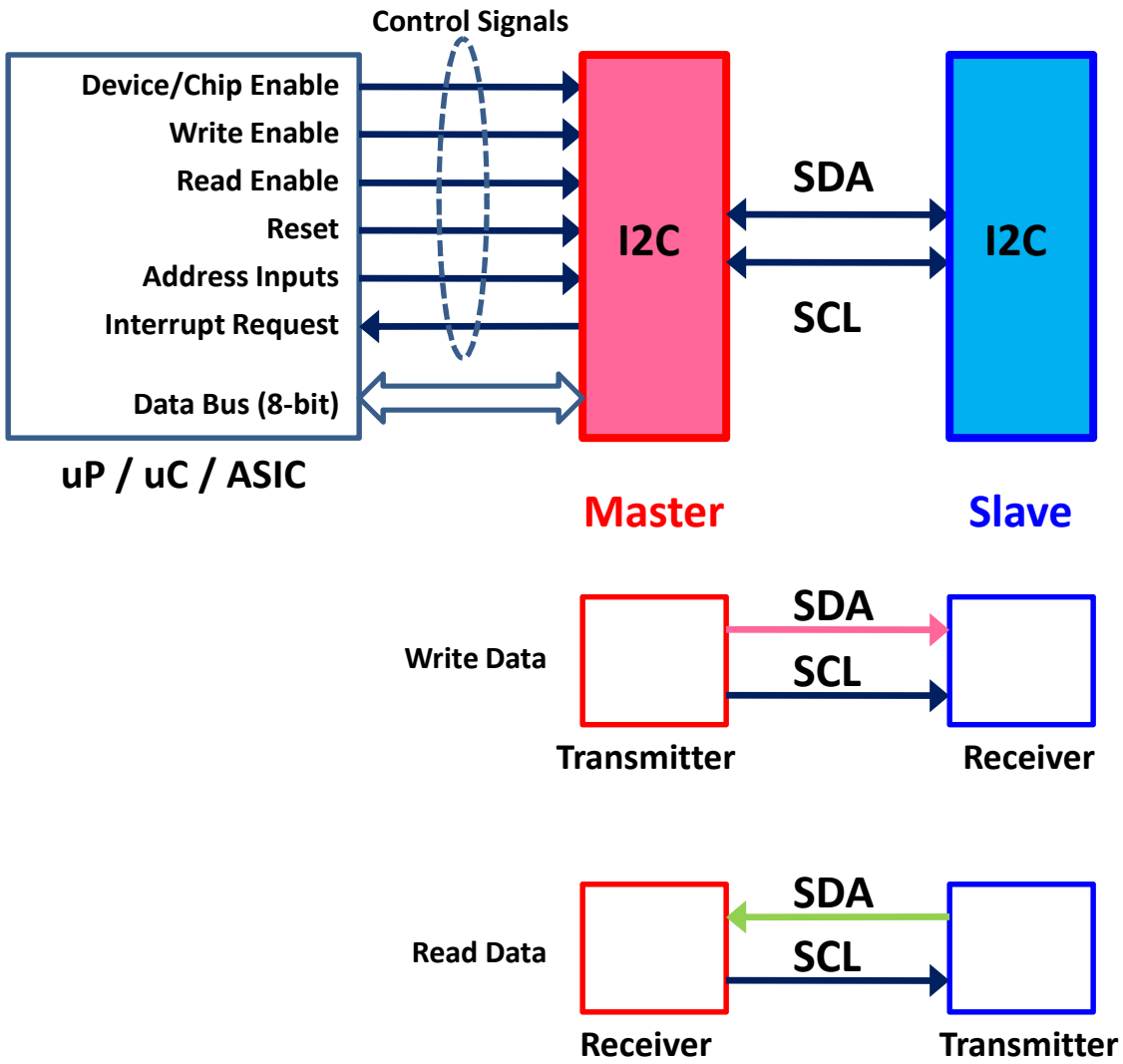


Bidirectional and Half Duplex

- **Bidirectional** serial data line (SDA)
  - communication is **half duplex**
  - bus arbitration
    - Arbitration is used when more than one master start data transfer at the same time.

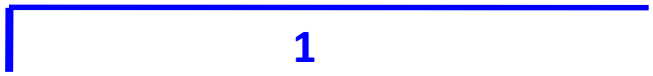
- and a serial clock line (SCL),
  - which provides synchronization.
  - **Bidirectional**

Master / Slave – Transmitter / Receiver



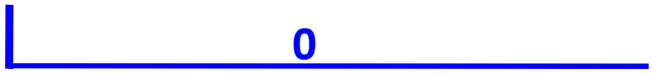
I2C Clock - SCL

- Not a “traditional” clock
- **Normally** is kept **High** (**logic 1**) using a pull-up



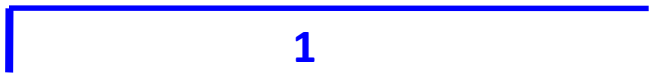
- Pulsed by the **master** during data transmission

- **Master** could be either the **transmitter** or **receiver**
- Slave device can hold clock **Low** (**logic 0**) if needs more time
  - Allows for **Flow Control**



I2C Data - SDA

- **Normally** is kept **High** (**logic 1**) using a pull-up



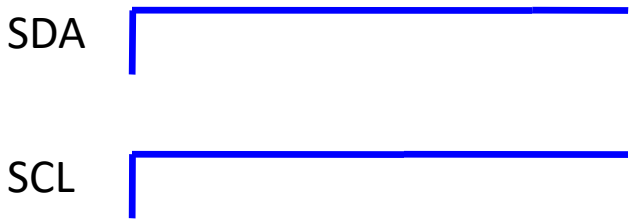
- Pulsed by the **Master** / **Slave** during data transmission for Data bits

- **Master** / **Slave** could be either the **transmitter** or **receiver**

Bus Operation

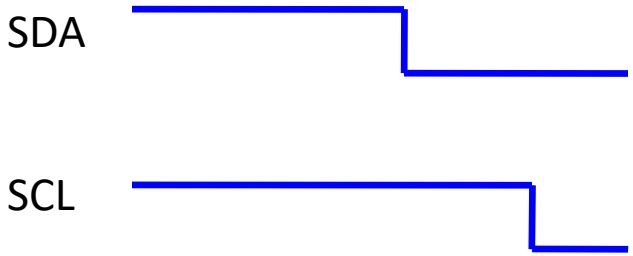
Idle State on I2C Bus

- SCL is **high** (**logic 1**)
  - and SDA is **high** (**logic 1**).



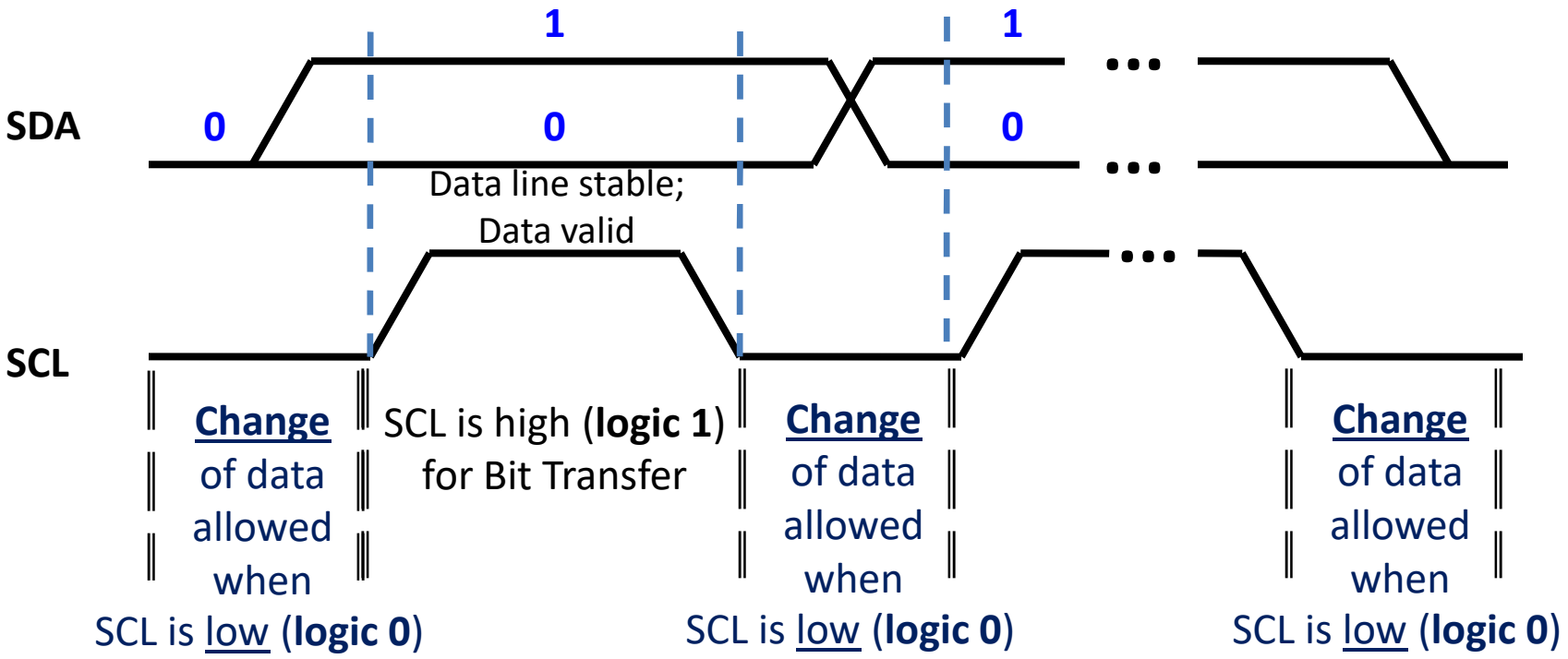
I2C Start Condition

- Master pulls SDA **low** (**logic 0**)
  - while SCL is **high** (**logic 1**)



Bit Transfer on I2C Bus

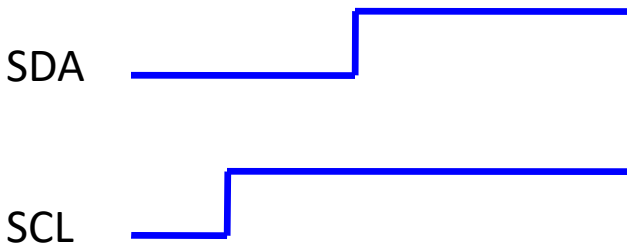
- **Normal** SDA changes only happen
  - when SCL is low (**logic 0**)
- In **normal** data transfer,
  - the data line (SDA) only changes **Logic** (**logic 0/1 => logic 1/0**)
    - when the clock (SCL) is low (**logic 0**)



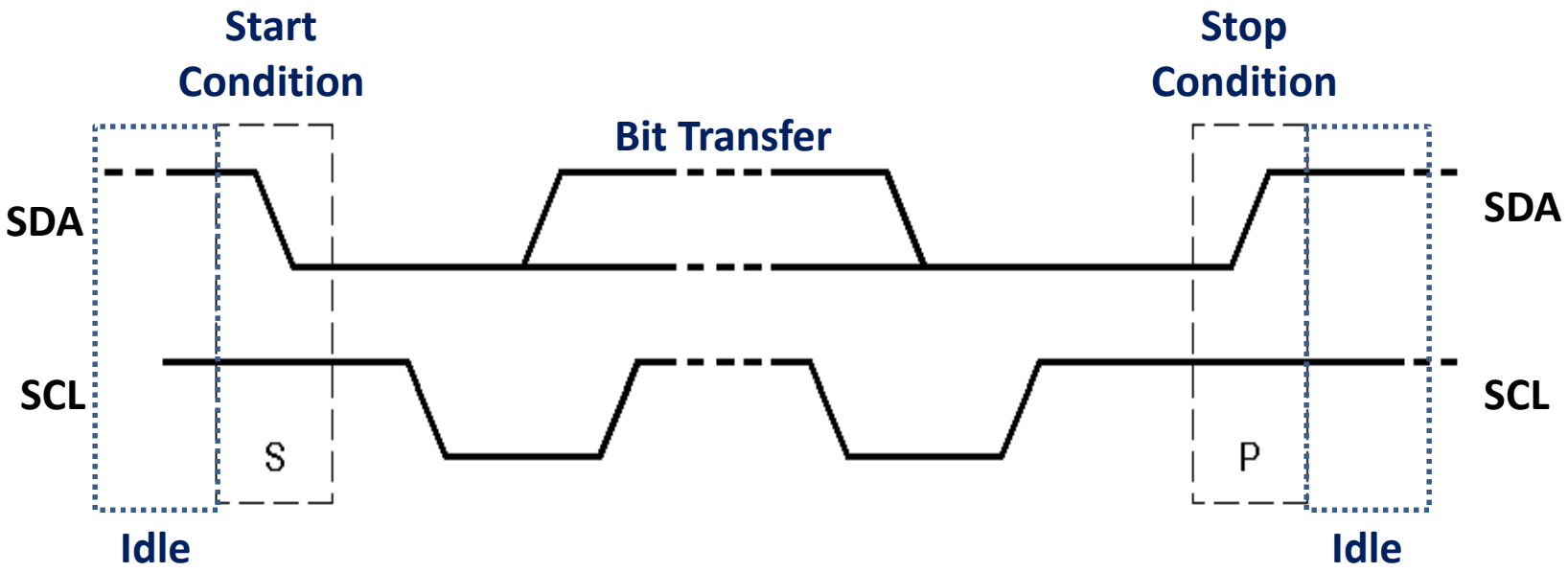
I2C Stop Condition

- Master pulls SDA **high** (**logic 1**)
  - while SCL is **high** (**logic 1**)

- Also used to abort transactions

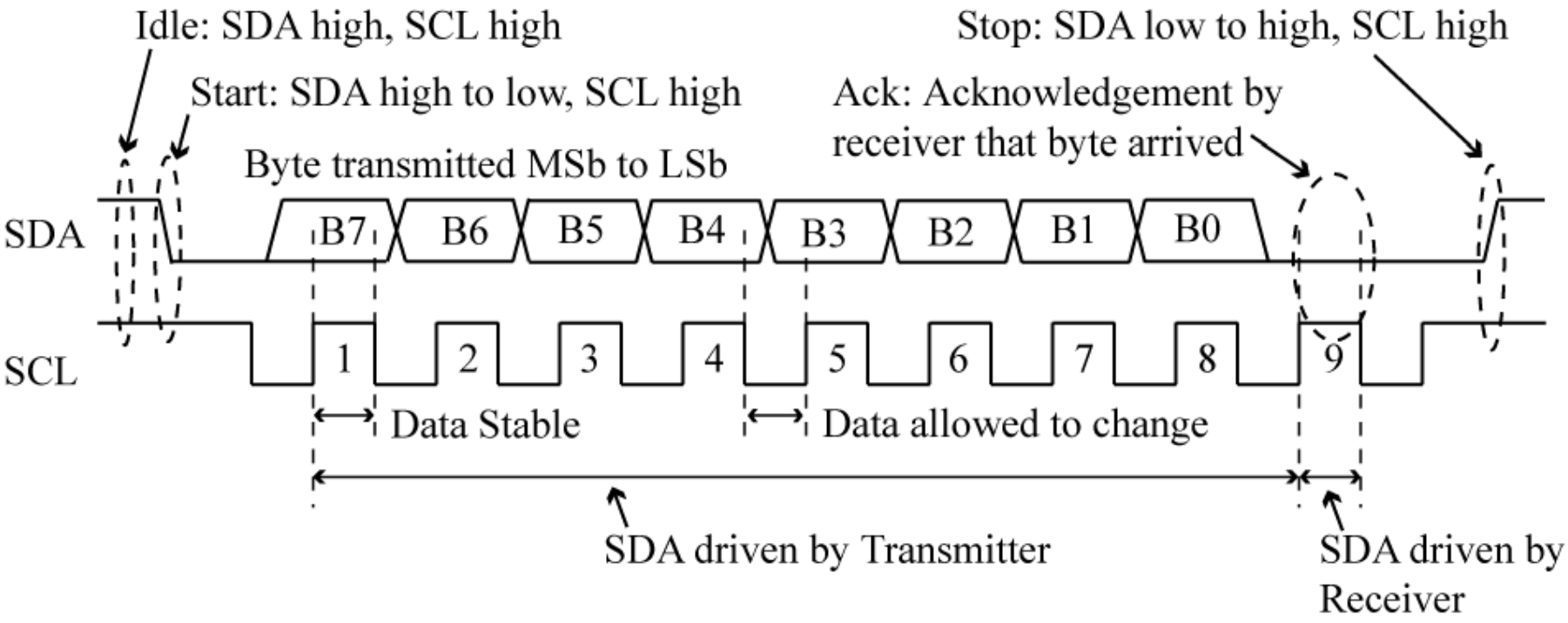


Idle / Start / Stop Conditions and Bit Transfer

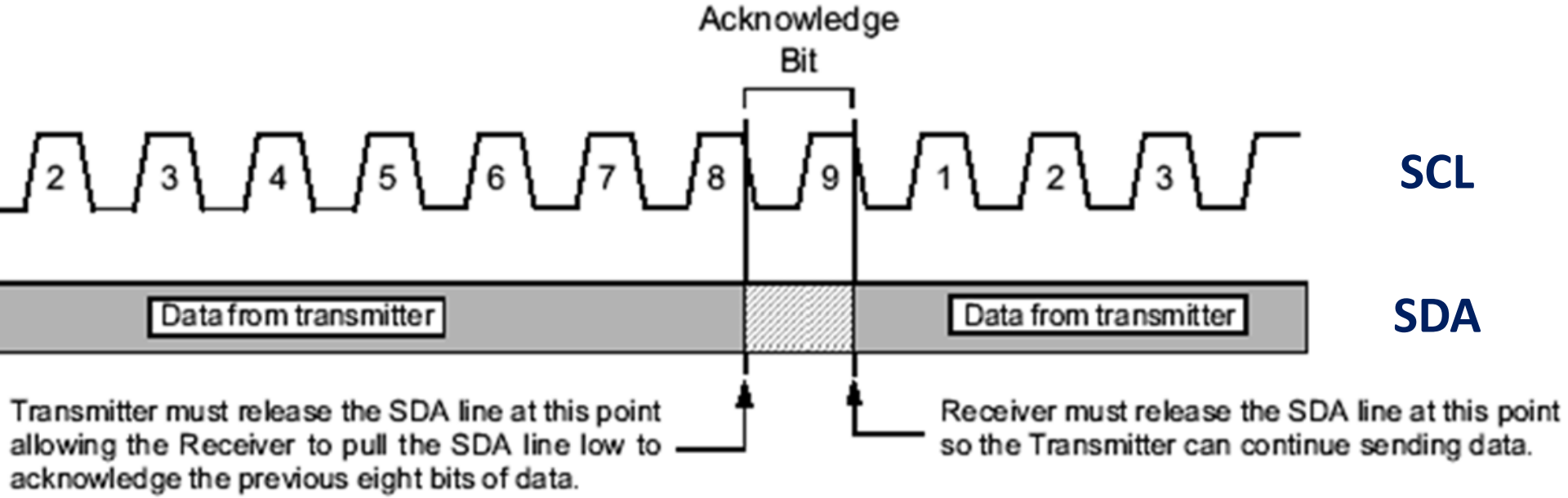


Idle / Start / Stop Conditions and Bit Transfer (2)

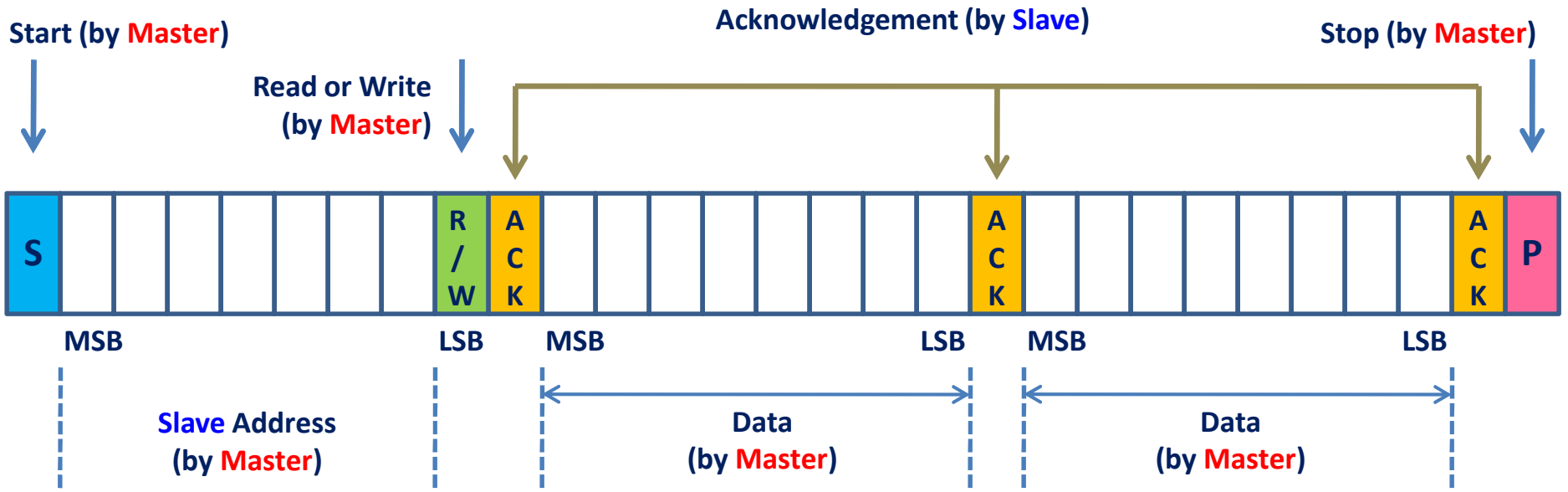
- Multiple bytes sent in a transaction;
  - every 8 bits leading to a 9th bit that is an acknowledge.

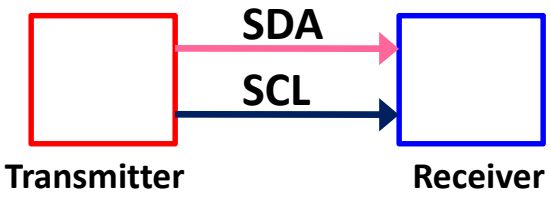


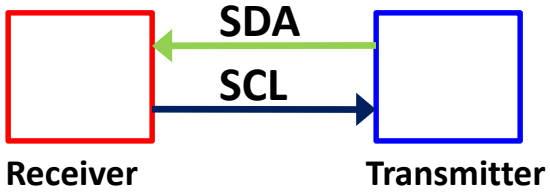
Acknowledgement Timing for Receiver

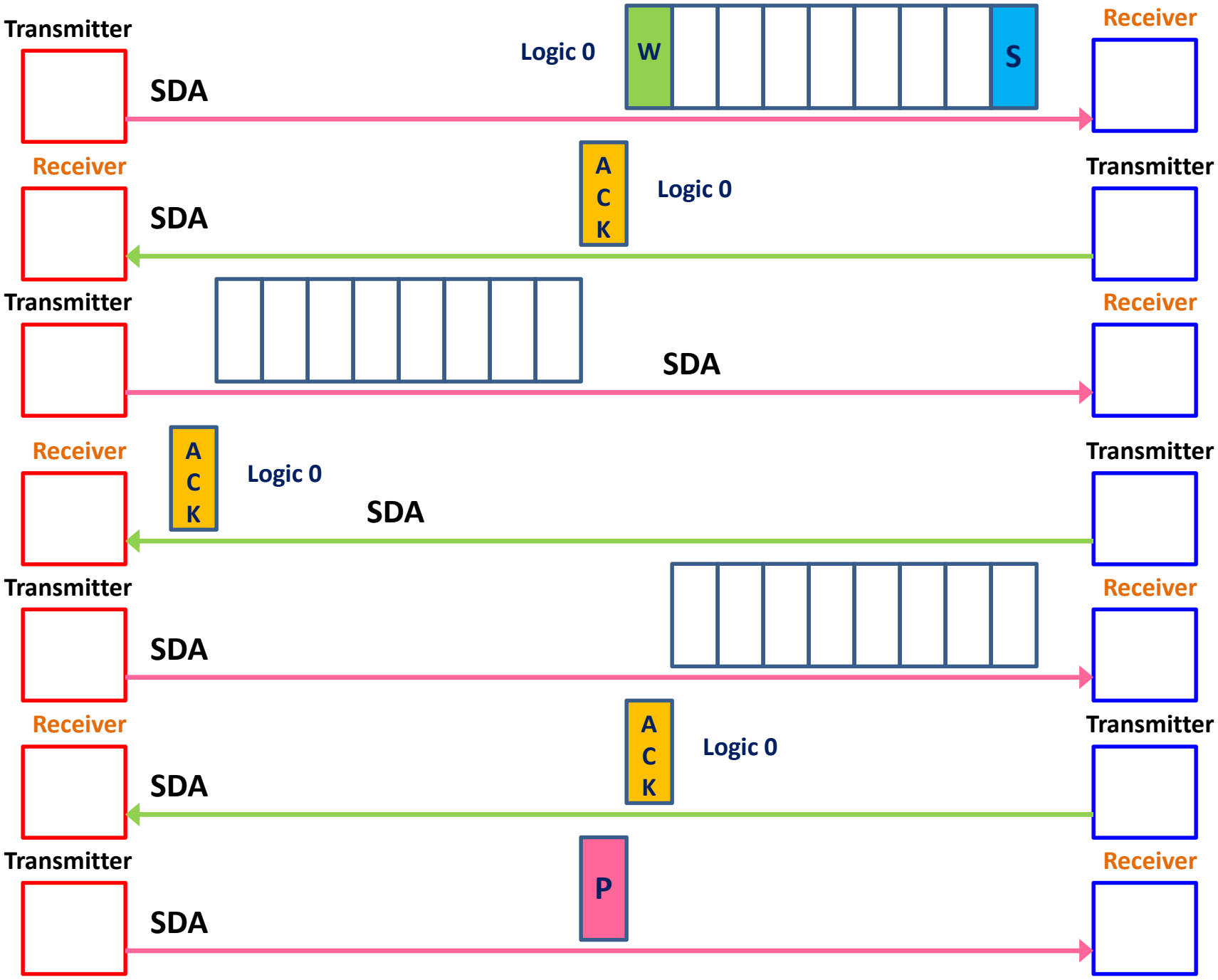


Example: Write Data from Master to Slave

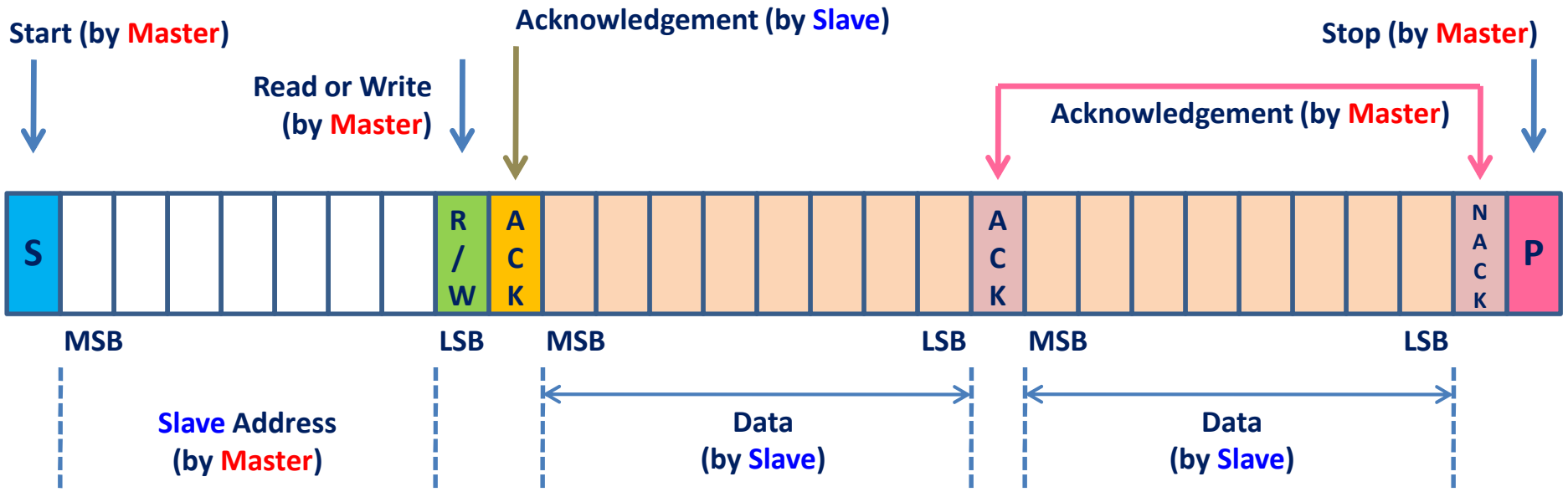


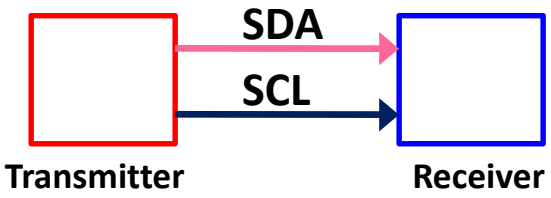


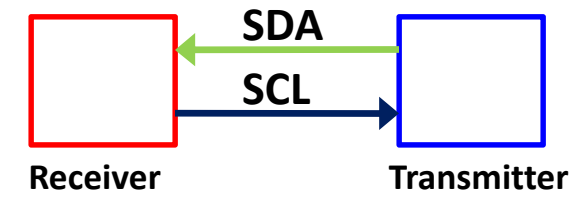


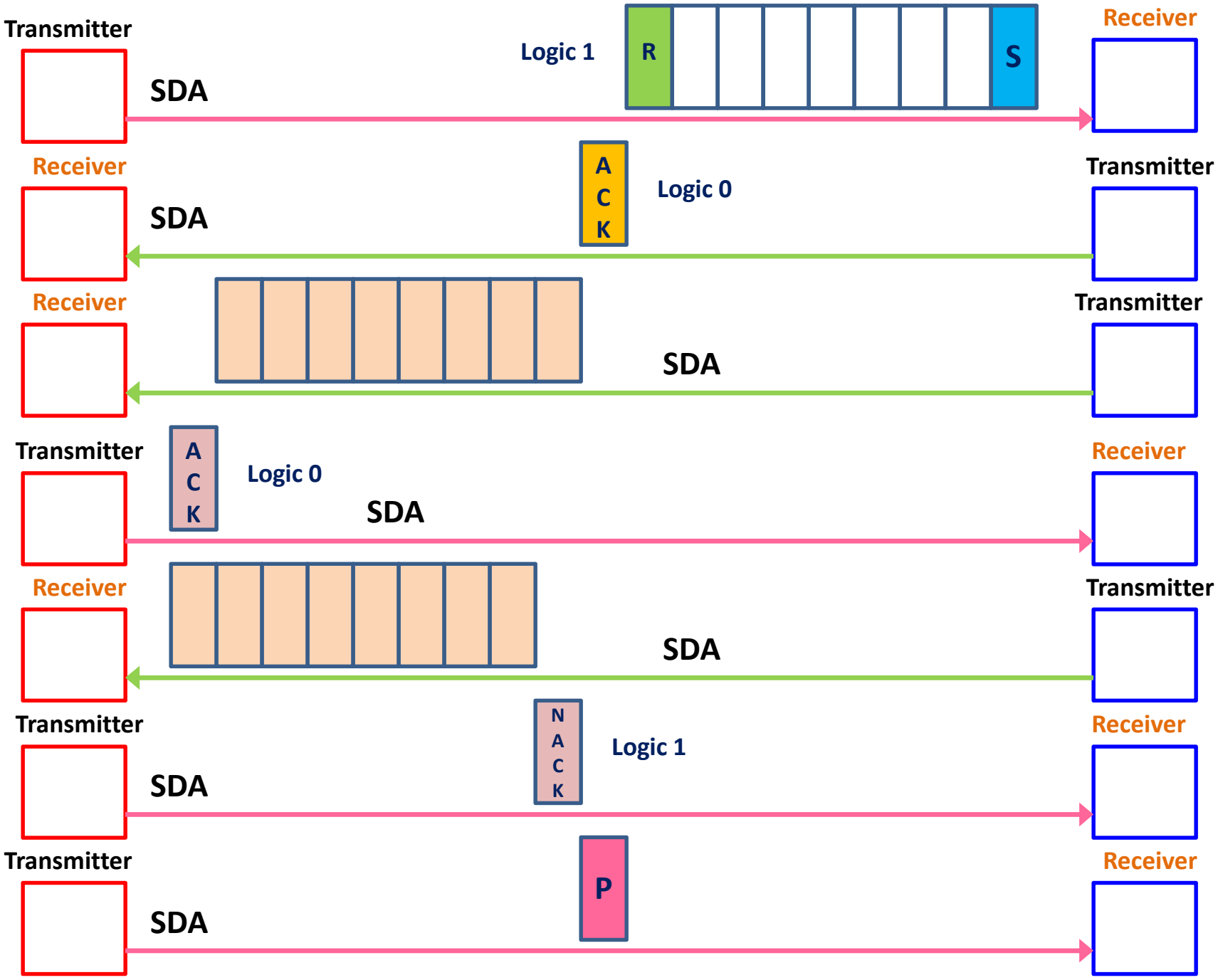

Transmitter
Logic 0
W
S
Receiver
SDA
Receiver
A
C
K
Logic 0
Transmitter
SDA
Transmitter
Receiver
SDA
Receiver
A
C
K
Logic 0
Transmitter
SDA
Transmitter
Receiver
SDA
Receiver
A
C
K
Logic 0
Transmitter
SDA
Transmitter
P
Receiver
SDA

Example: Read Data from Slave to Master







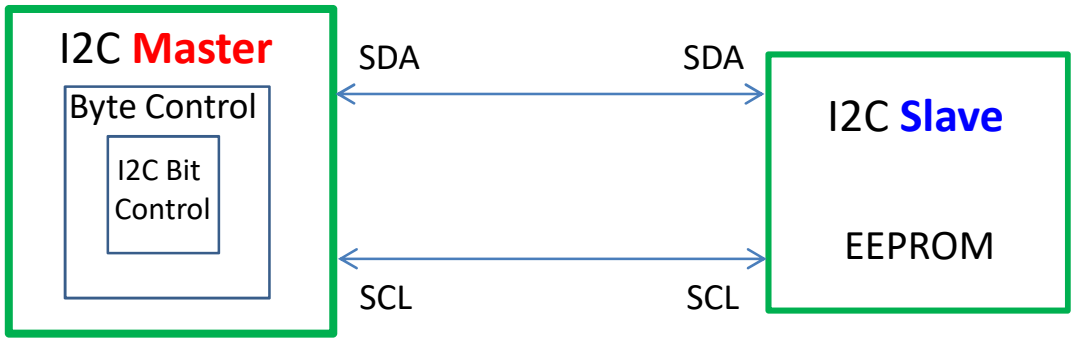

Transmitter
Logic 1
R
S
Receiver
SDA
Receiver
A
C
K
Logic 0
Transmitter
SDA
Receiver
Transmitter
SDA
Transmitter
A
C
K
Logic 0
Receiver
SDA
Receiver
Transmitter
SDA
Transmitter
N
A
C
K
Logic 1
Receiver
SDA
Transmitter
P
Receiver
SDA

Transaction

- Transmitter/Receiver differs from Master/Slave
  - Master initiates transactions
  - Slave responds

- Transmitter sets data on SDA line, Receiver acks
  - For a **write**,
    - Master is Transmitter
  - For a **read**,
    - Slave is Transmitter

Read/Write



1) Generate **Start** signal
2) Write Slave address + Write bit.
3) Write memory location
4) **Stop**

WR

5) Generate **Start** signal
6) Write Slave address + Read bit
7) Write memory location
8) Read from the memory

RD

Address Transmission

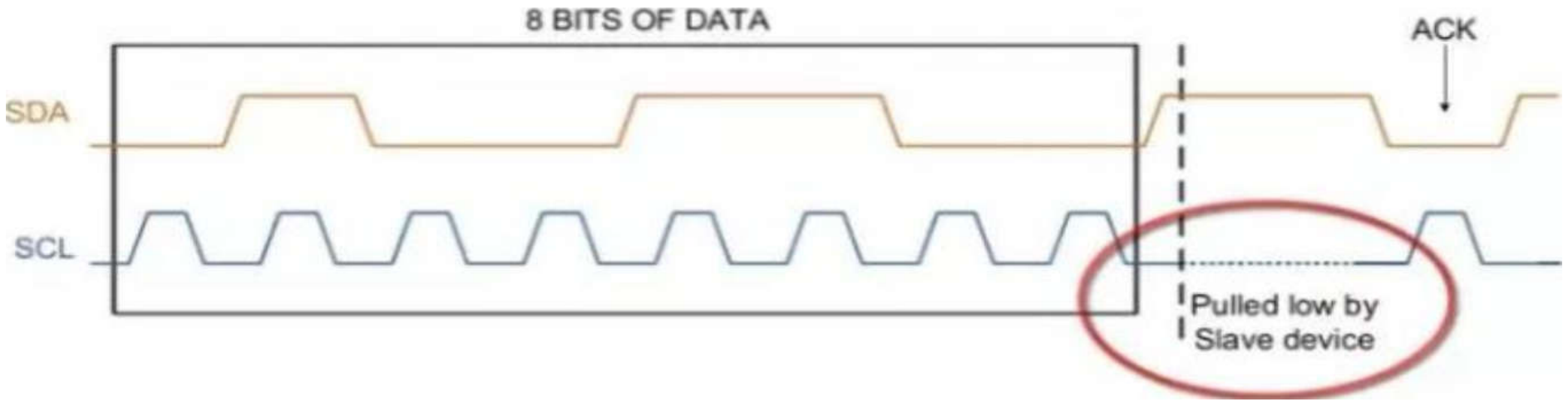
- Data is always sampled on the **rising** clock edge
- Address is **7** bits
- An 8-th bit (**R/W**) indicated read or write
  - High (**logic 1**) for **read**
  - Low (**logic 0**) for **write**

Data Transmission

- Transmitted just like address (8 bits)
- For a **write**,
  - Master transmits, Slave acknowledges
- For a **read**,
  - Slave transmits, Master acknowledges
- Transmission continues
- Subsequent bytes sent
- Continue until Master creates **Stop** condition

Clock Stretching

- During Communication, on a byte level,
  - **device** may be able to receive data at fast rate
    - but it needs more time
      - to store a **received byte**
      - or to prepare a **next byte** to be transmitted.
- Slaves can then hold the **SCL** line **low** (**logic 0**)
  - which is known as ***Clock Stretching***.
- During this time, Master goes into a **Wait** state.



Addressing

- Each node has a unique **7** (or 10) bit address
- Peripherals often have fixed and programmable address portions
- Addresses starting with 0000 or 1111 have special functions:
  - 0000000 is a General Call Address
  - 0000001 is a Null (CBUS) Address
  - 1111XXX Address Extension
  - 1111111 Address Extension – Next Bytes are the Actual Address

Addressing (2)

- Upper four bits are assigned by device manufacturer
  - and are hardcoded in the device.
- Lower three bits are used in different ways by manufacturer.
  - E.g.,
    - ❑ Control byte (contains slave address):

      7 6 5 4 3 2 1 0

      1 0 1 0 B0 A1 A0 R/W

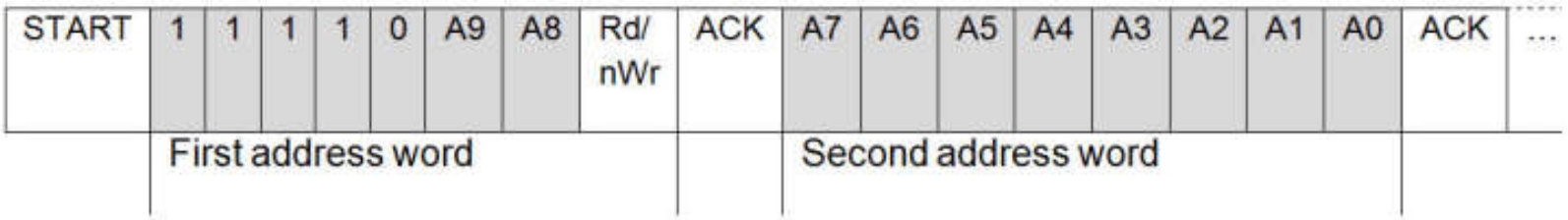
    - ❑ B0 is block select (upper/lower 32K).
    - ❑ A1, A0 are chip selects,

      four devices on one bus.

Addressing (3)

Address	Purpose
0000000 0	General Call – addresses all devices supporting the general call mode
0000000 1	Start Byte
0000001 X	CBUS addresses
0000010 X	Reserved for different bus formats
0000011 X	Reserved for future purpose
00001XX X	High-speed Master code
11110XX X	10-bits slave addressing
11111XX X	Reserved for future purposes

10-bit Addressing

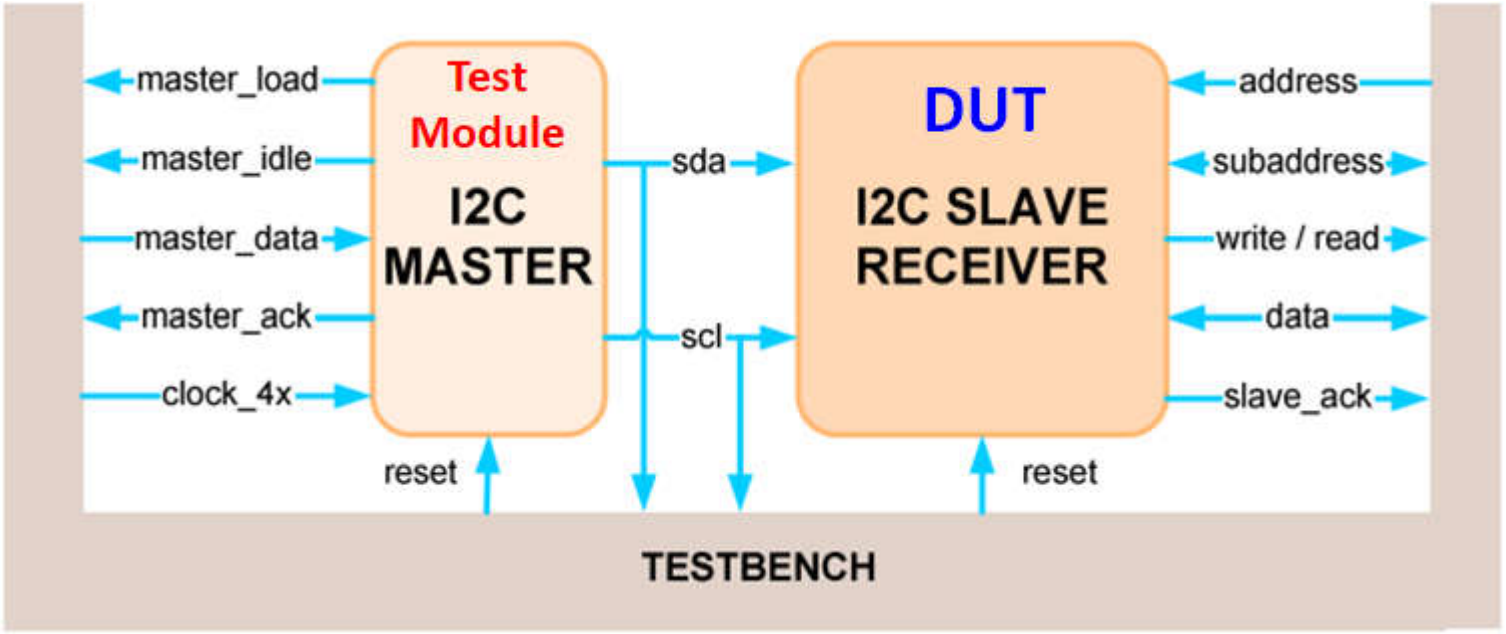


10 bits address:

A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

I2C BUS COMMUNICATION TEST

An Example of DUT and Testbench for I2C



Backup Slide

BACKUP SLIDE

Thank You