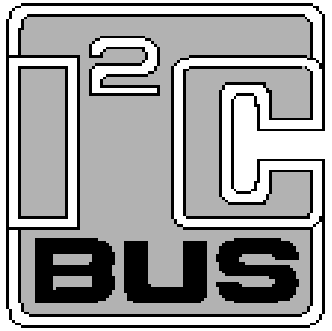
Indian Institute of
Technology Bombay



The I²C Bus

Suryakant Toraskar
smtoraskar.iitbombay@gmail.com
Computer Architecture and Dependable Systems Lab
Department of Electrical Engineering
Indian Institute of Technology Bombay

The I²C Bus

- What is the I²C Bus and what is it used for?
- Bus characteristics
- I²C Bus Protocol
- Data Format
- Typical I²C devices
- Example device
- PIC18F I2C support

What is I²C

- **The name stands for “Inter - Integrated Circuit Bus”**
- **A Small Area Network connecting ICs and other electronic systems**
- **Originally intended for operation on one single board / PCB**
  - Synchronous Serial Signal
  - Two wires carry information between a number of devices
  - One wire use for the data
  - One wire used for the clock
- **Today, a variety of devices are available with I²C Interfaces**
  - Microcontroller, EEPROM, Real-Timer, interface chips, LCD driver, A/D converter

What is I²C used for?

- **Data transfer between ICs and systems at relatively low rates**
  - "Classic" I²C is rated to 100K bits/second
  - "Fast Mode" devices support up to 400K bits/second
  - A "High Speed Mode" is defined for operation up to 3.4M bits/second
- **Reduces Board Space and Cost By:**
  - Allowing use of ICs with fewer pins and smaller packages
  - Greatly reducing interconnect complexity
  - Allowing digitally controlled components to be located close to their point of use

I²C Bus Characteristics

- **Includes electrical and timing specifications, and an associated bus protocol**
- **Two wire serial data & control bus implemented with the serial data (SDA) and clock (SCL) lines**
  - For reliable operation, a third line is required: **Common ground**
- **Unique start and stop condition**
- **Slave selection protocol uses a 7-Bit slave address**
  - The bus specification allows an extension to 10 bits
- **Bi-directional data transfer**
- **Acknowledgement after each transferred byte**
- **No fixed length of transfer**

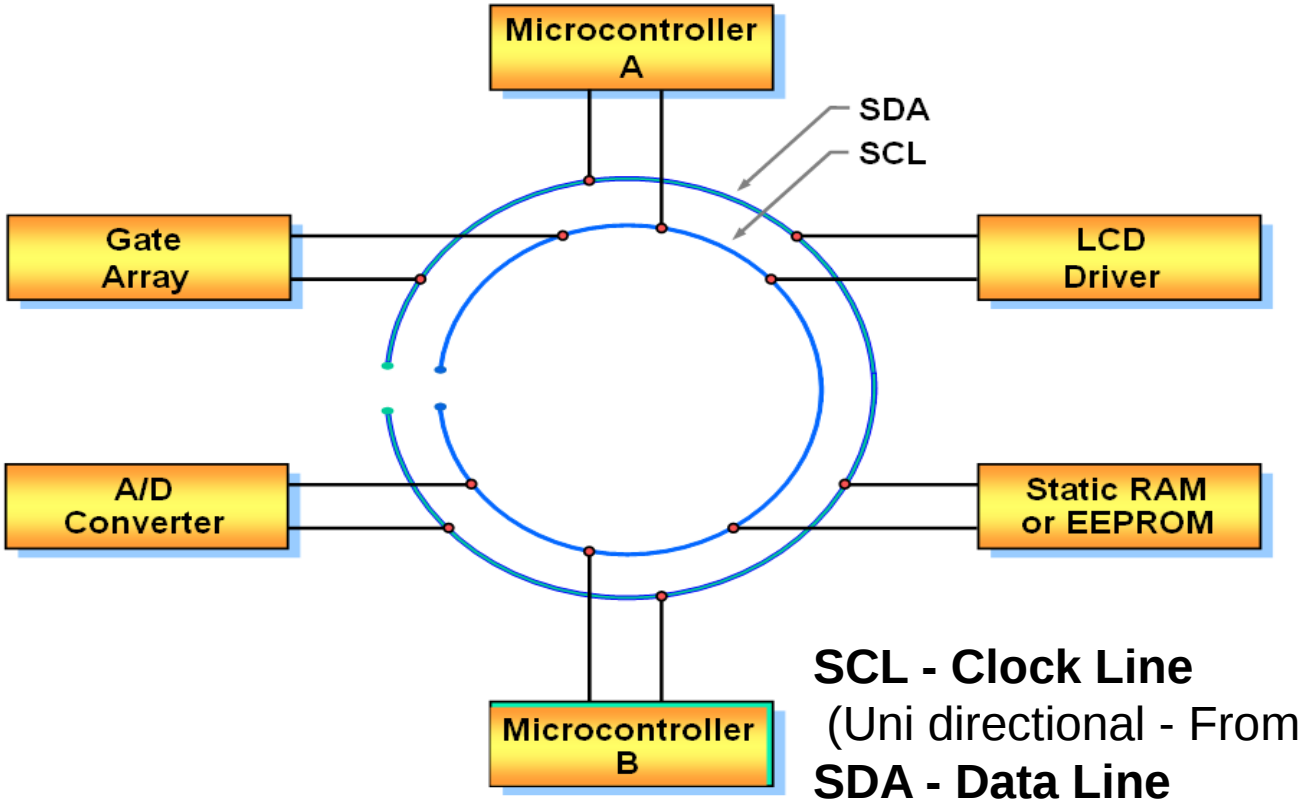
I²C Bus Characteristics (cont'd)

- **True multi-master capability**
  - Clock synchronization
  - Arbitration procedure
- **Transmission speeds up to 100Khz (classic I2C)**
- **Max. line capacitance of 400pF, approximately 4 meters (12 feet)**
- **Allows series resistor for IC protection**
- **Compatible with different IC technologies**

I²C Bus Definitions

- **Master:**
  - Initiates a transfer by generating start and stop conditions
  - Generates the clock
  - Transmits the slave address
  - Determines data transfer direction
- **Slave:**
  - Responds only when addressed
  - Timing is controlled by the clock line

I²C Bus Configuration Example

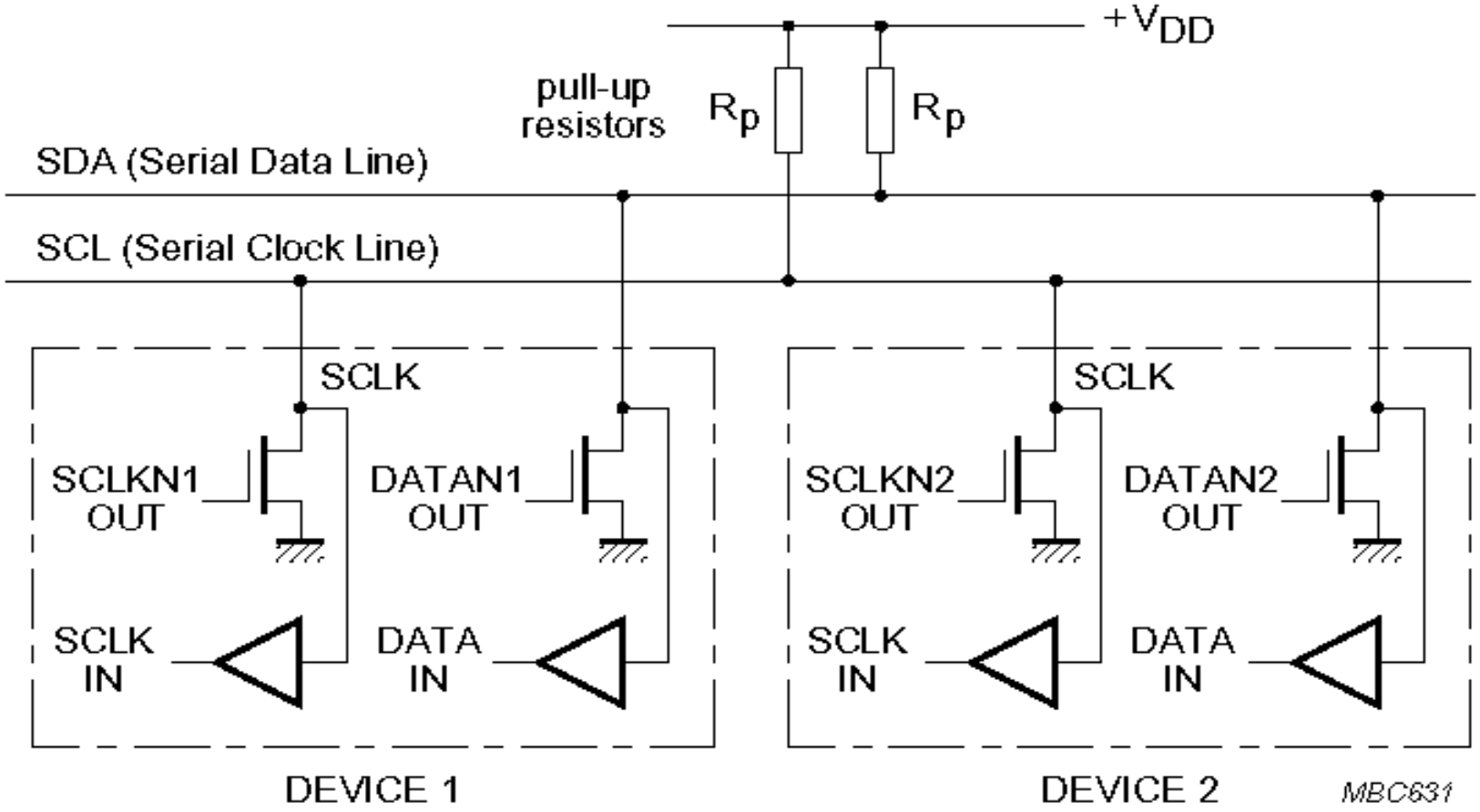


SCL - Clock Line
(Uni directional - From Master to Slave)
SDA - Data Line
(Bidirectional)

I²C Hardware Details

- **Devices connected to the bus must have an open drain or open collector output for serial clock and data signal**
- **The device must also be able to sense the logic level on these pins**
- **All devices have a common ground reference**
- **The serial clock and data lines are connected to Vdd(typically +5V) through pull up resistors**
- **At any given moment the I2C bus is:**
  - Quiescent (Idle), or
  - in Master transmit mode or
  - in Master receive mode.

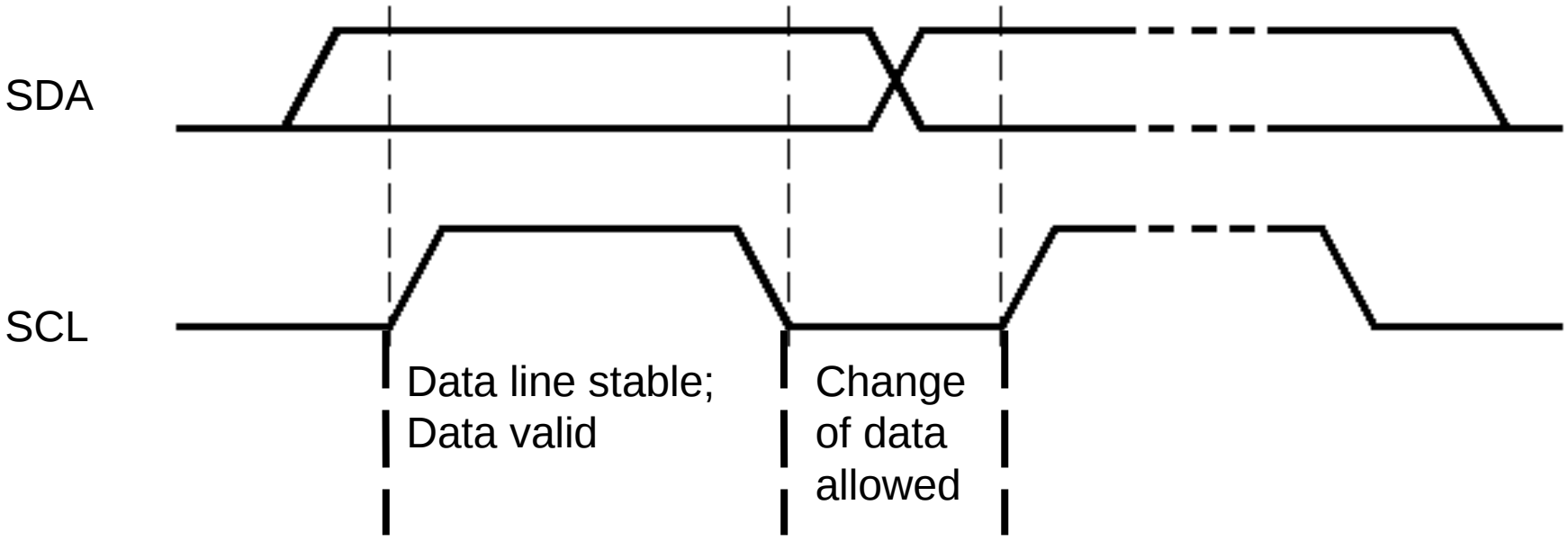
I²C Electrical Aspects



- **I²C devices are wire ANDed together.**
- **If any single node writes a zero, the entire line is zero**

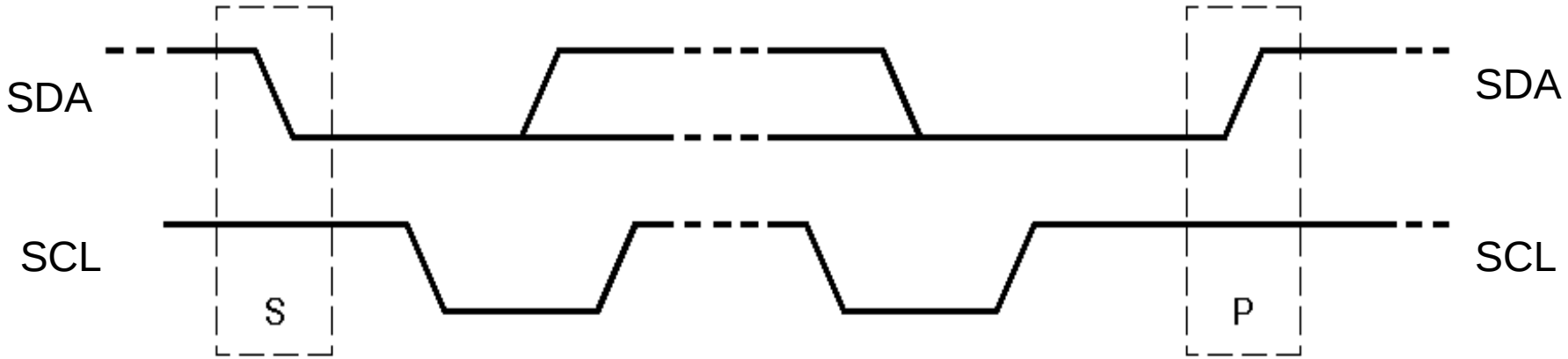
Bit Transfer on the I²C Bus

- In normal data transfer, the data line only changes state when the clock is low



Start and Stop Conditions

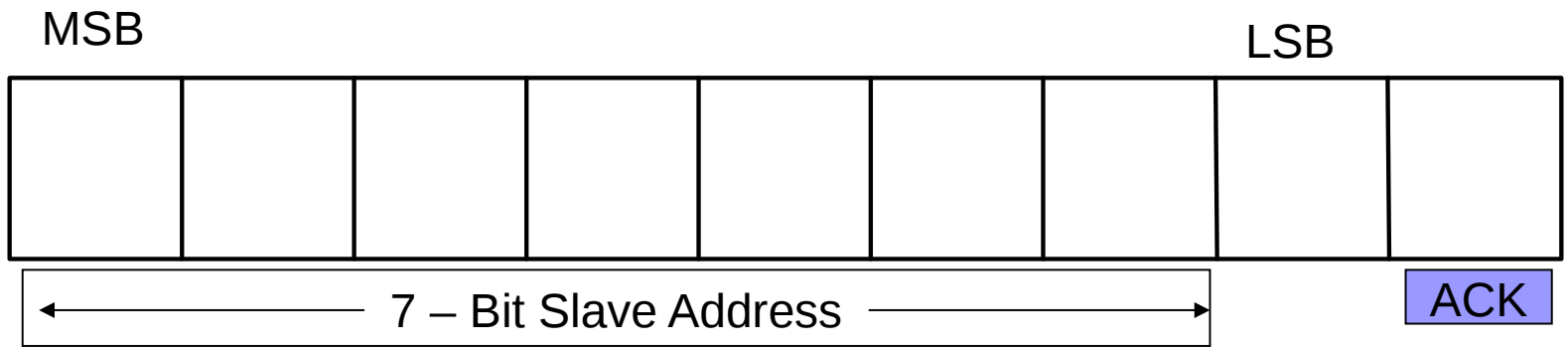
- A transition of the data line while the clock line is high is defined as either a start or a stop condition.
- Both start and stop conditions are generated by the bus master
- The bus is considered busy after a start condition, until a stop condition occurs



I²C 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

First Byte in Data Transfer on the I²C Bus



R/Wr

0 – Slave written to by Master

1 – Slave read by Master

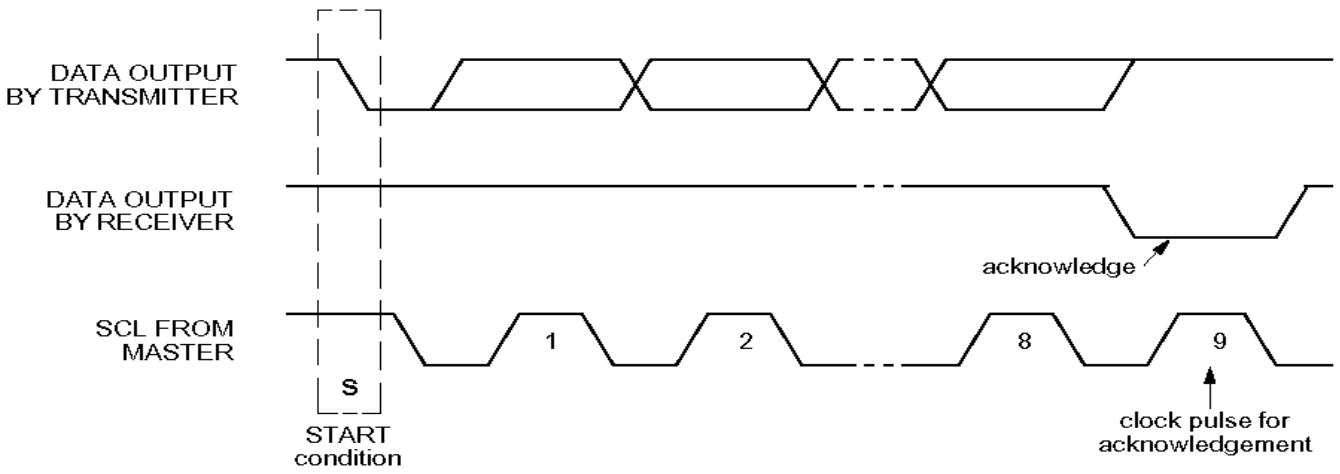
ACK – Generated by the slave whose address has been output.

I²C Bus Connections

- **Masters can be**
  - Transmitter only
  - Transmitter and receiver
- **Slaves can be**
  - Receiver only
  - Receiver and transmitter

Acknowledgements

- Master/slave receivers pull data line low for one clock pulse after reception of a byte
- Master receiver leaves data line high after receipt of the last byte requested
- Slave receiver leaves data line high on the byte following the last byte it can accept



Transmitter releases SDA line during 9th clock pulse.

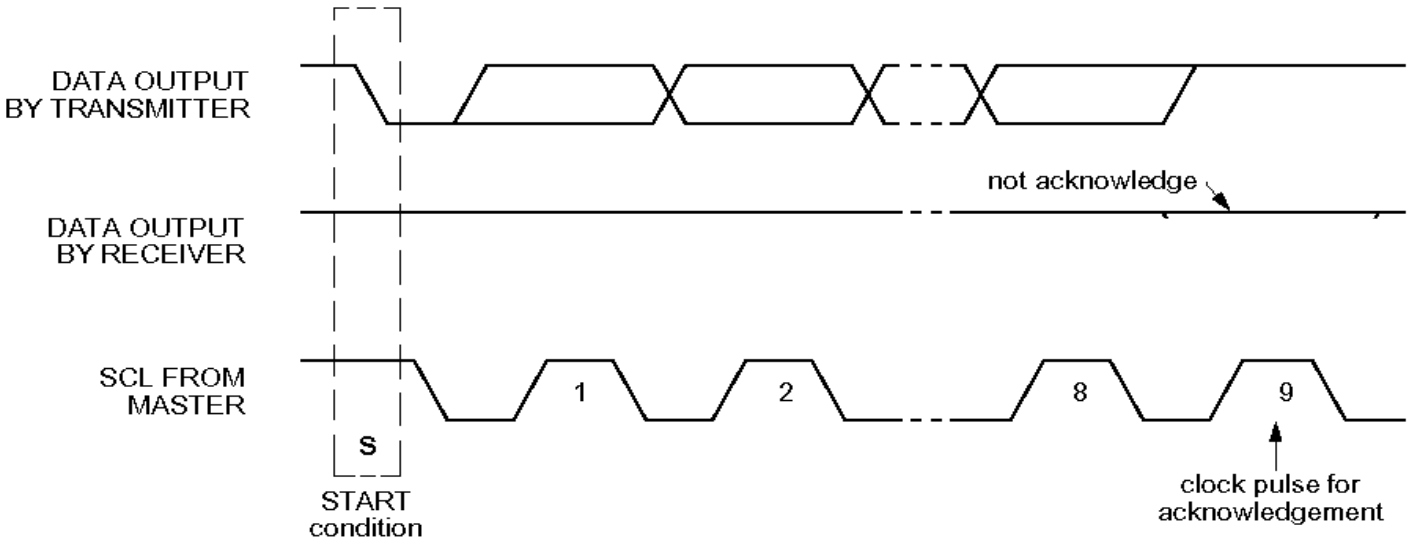
Acknowledgement from receiver

Acknowledgements

- **From Slave to Master Transmitter:**
  - After address received correctly
  - After data byte received correctly
- **From Slave to Master Receiver:**
  - Never (Master Receiver generates ACK)
- **From Master Transmitter to Slave:**
  - Never (Slave generates ACK)
- **From Master Receiver to Slave:**
  - After data byte received correctly

Negative Acknowledge

- Receiver leaves data line high for one clock pulse after reception of a byte



Transmitter releases SDA line during 9th clock pulse.

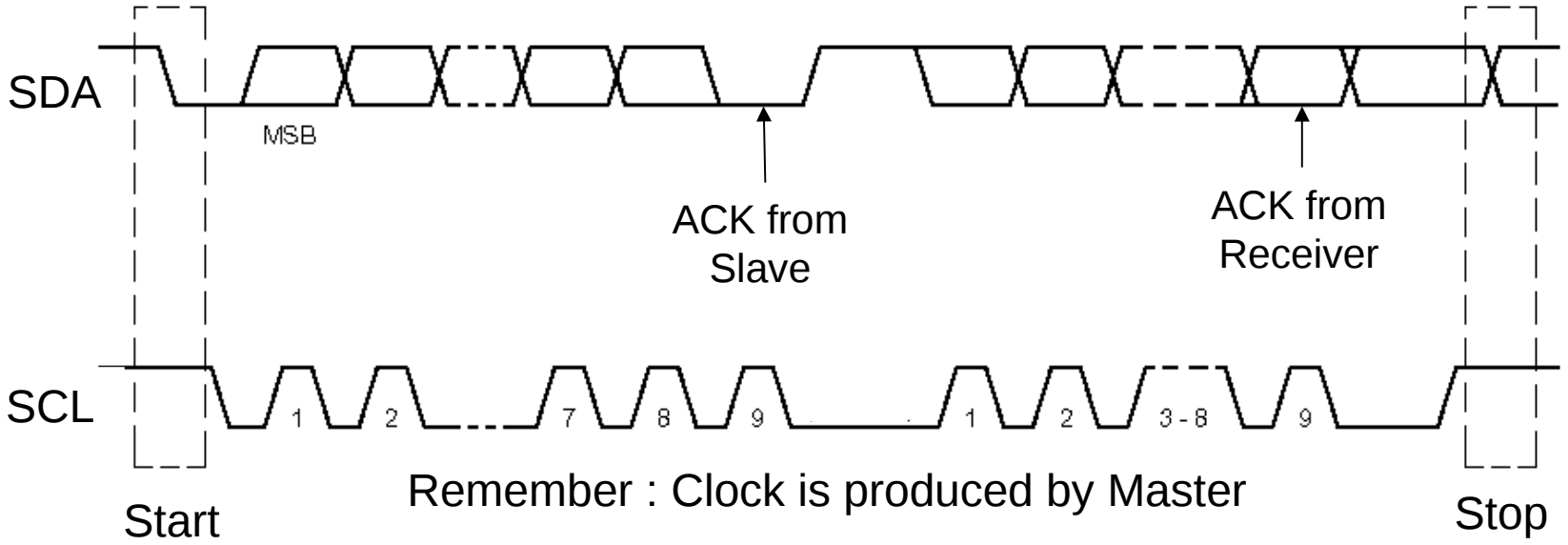
Not acknowledgement (NACK) from receiver

Negative Acknowledge (Cont'd.)

- **From Slave to Master Transmitter:**
  - After address not received correctly
  - After data byte not received correctly
  - Slave Is not connected to the bus
- **From Slave to Master Receiver:**
  - Never (Master Receiver generates ACK)
- **From Master Transmitter to Slave:**
  - Never (Slave generates ACK)
- **From Master Receiver to Slave:**
  - After last data byte received correctly

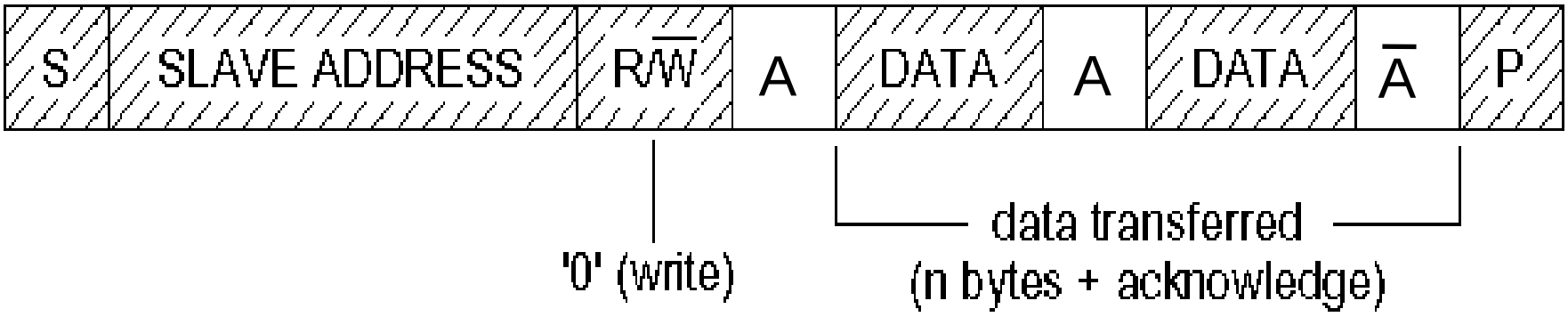
Data Transfer on the I2C Bus

- **Start Condition**
- **Slave address + R/W**
  - Slave acknowledges with ACK
- **All data bytes**
  - Each followed by ACK
- **Stop Condition**



Data Formats

- **Master writing to a Slave**



from master to slave

from slave to master

MBC605

A = acknowledge (SDA LOW)

Ā = not acknowledge (SDA HIGH)

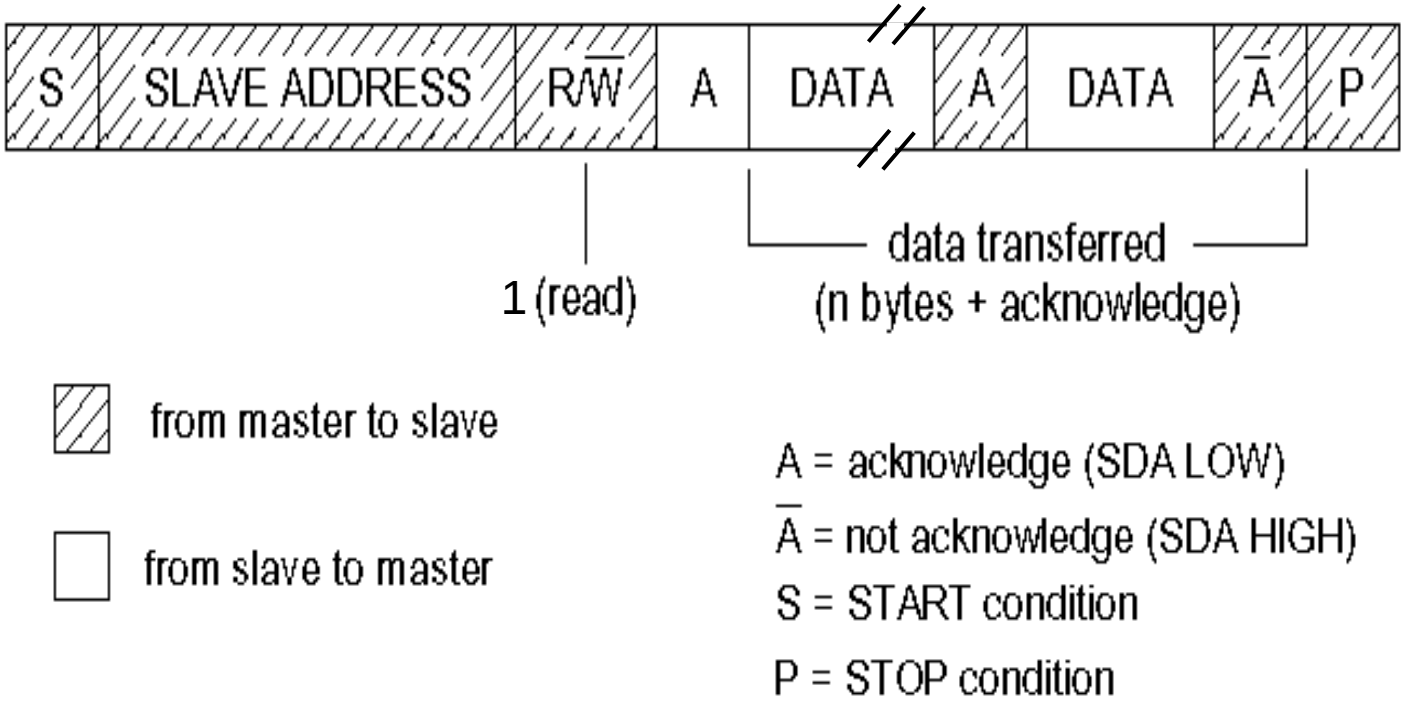
S = START condition

P = STOP condition

Data Formats Cont'd.

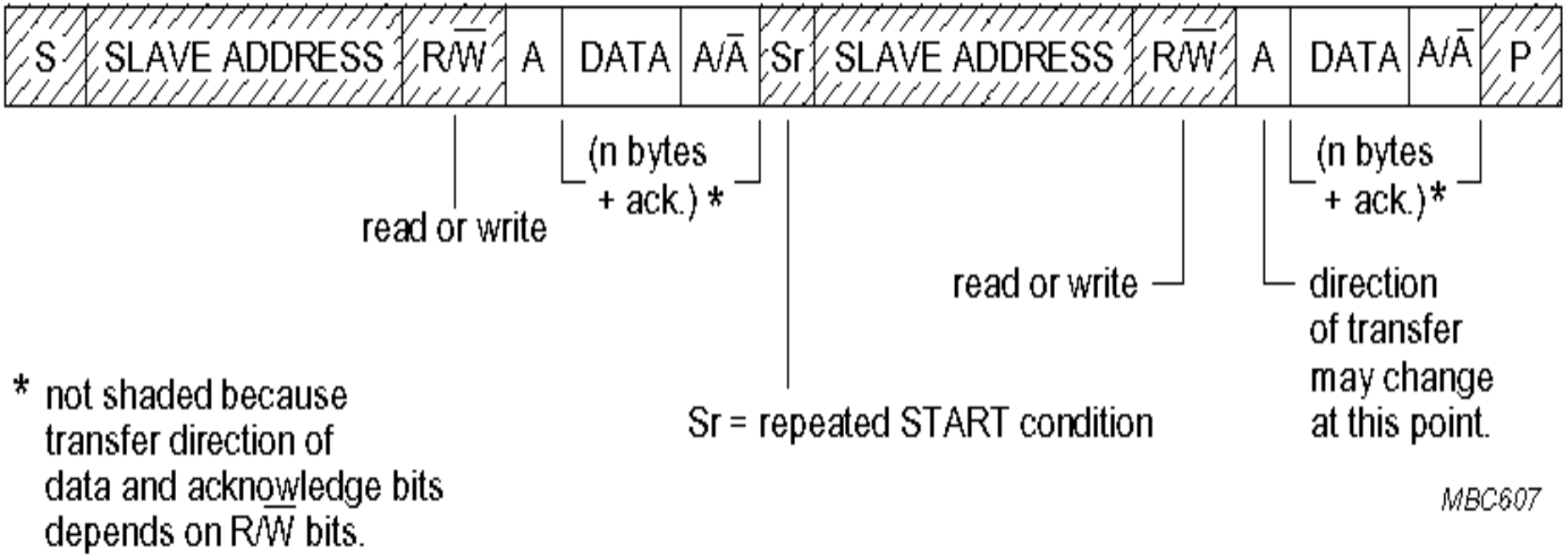
- **Master reading from a Slave** :

Master is Receiver of data and Slave is Transmitter of data.



Data Formats Cont'd.

- **Combined Format**



- A **repeated start** avoids releasing the bus and therefore prevents another master from taking over the bus

Bus Recovery

- **An I²C bus can be "locked" when:**
  - A Master and a Slave get out of synch
  - A Stop is omitted or missed (possibly due to noise)
  - Any device on the bus holds one of the lines low improperly, for any reason
  - A shorted bus line
- **If SCL can be driven, the Master may send extra clocks until SDA goes high, then send a Stop.**
- **If SCL is stuck low, only the device driving it can correct the problem.**

Type of I²C Implementations

- **Byte Oriented Interface**
  - Data is handled one byte at a a time
  - Processor interprets a status byte when an event occurs
  - For instance Philips 8xC554, 8xC591
- **Bit Oriented Interface**
  - Processor is involved in every bus event when the interface is not Idle
- **“Bit Banged”**
  - Implemented completely in software on 2 regular I/O pins of the microcontroller
  - Works for single master systems
  - Not recommended for Slave devices or Multimaster systems

Available I²C Devices

- **Analog to Digital Converters (A/D, D/A):** MMI functions, battery & converters, temperature monitoring, control systems
- **Bus Controller:** Telecom, consumer electronics, automotive, Hi-Fi systems, PCs, servers
- **Bus Repeater, Hub & Expander:** Telecom, consumer electronics, automotive, Hi-Fi systems, PCs, servers
- **Real Time Clock (RTC)/Calendar:** Telecom, EDP, consumer electronics, clocks, automotive, Hi-Fi systems, FAX, PCs, terminals
- **DIP Switch:** Telecom, automotive, servers, battery & converters, control systems
- **LCD/LED Display Drivers:** Telecom, automotive instrument driver clusters, metering systems, POS terminals, portable items, consumer electronics

Available I²C Devices

- **General Purpose Input/Output (GPIO) Expanders and LED Display Control:** Servers, keyboard interface, expanders, mouse track balls, remote transducers, LED drive, interrupt output, drive relays, switch input
- **Multiplexer & Switch:** Telecom, automotive instrument driver clusters, metering systems, POS terminals, portable items, consumer electronics
- **Serial RAM/ EEPROM:** Scratch pad/ parameter storage
- **Temperature & Voltage Monitor:** Telecom, metering systems, portable items, PC, servers
- **Voltage Level Translator:** Telecom, servers, PC, portable items, consumer electronics

End use

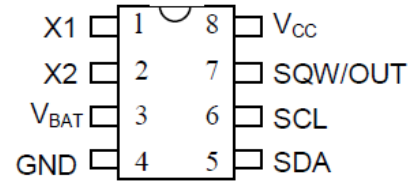
- **Telecom:** Mobile phones, Base stations, Switching, Routers
- **Data processing:** Laptop, Desktop, Workstation, Server
- **Instrumentation:** Portable instrumentation, Metering systems
- **Automotive:** Dashboard, Infotainment
- **Consumer:** Audio/video systems, Consumer electronics (DVD, TV etc.)

Example – DS1307

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
  - 56-byte, battery-backed, nonvolatile (NV)
- RAM for data storage
  - Two-wire serial interface
  - Programmable squarewave output signal
  - Automatic power-fail detect and switchcircuitry
  - Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range:
- -40°C to +85°C

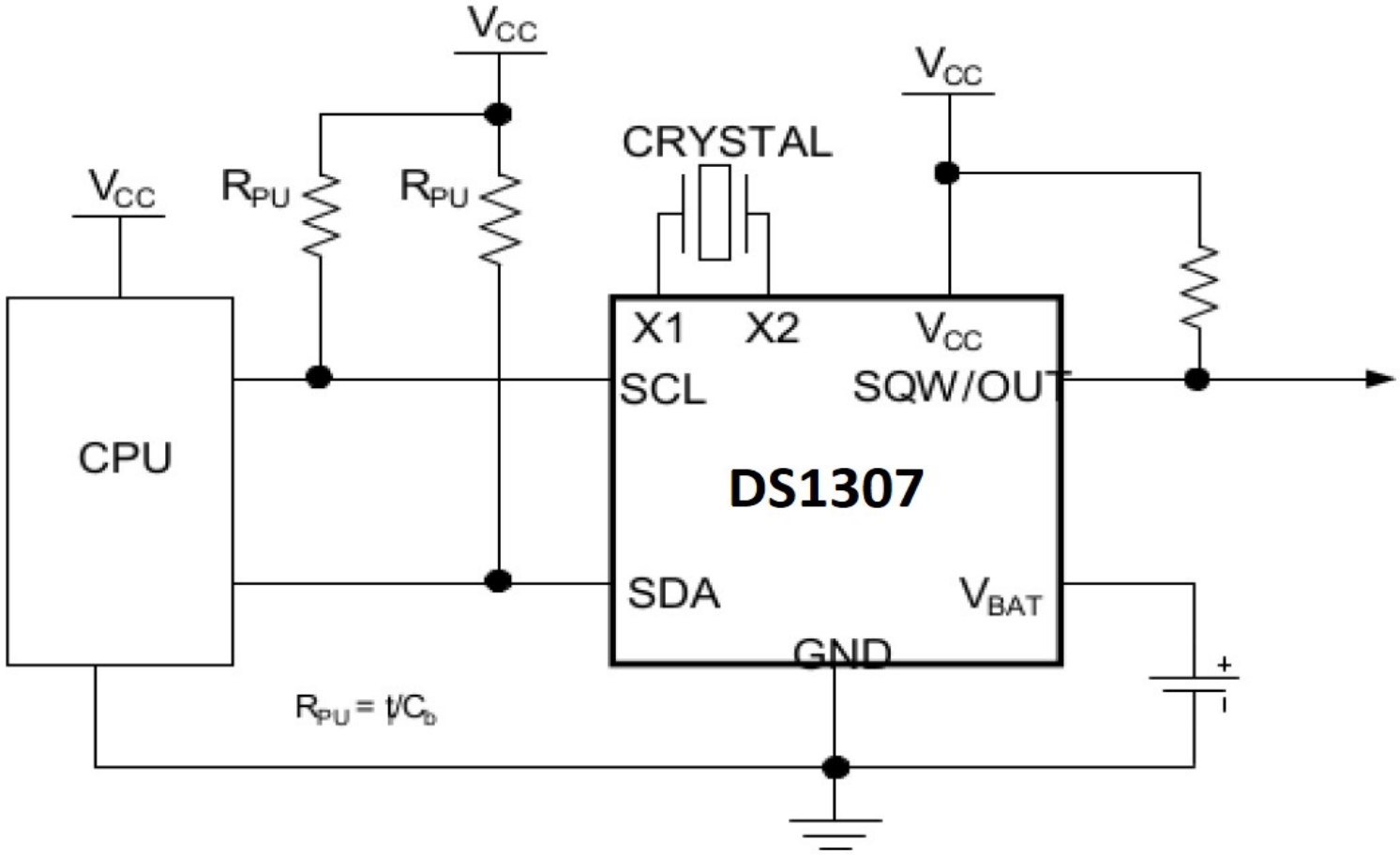
- VCC - Primary Power Supply
- X1, X2 - 32.768kHz Crystal Connection
- VBAT - +3V Battery Input
- GND - Ground
- SDA - Serial Data
- SCL - Serial Clock
- SQW/OUT - Square Wave/Output Driver

PIN ASSIGNMENT



DS1307 8-Pin DIP (300-mil)

DS1307 Hardware



All Registers used are of value 10kΩ.

DS1307 Registers

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/ AM							
03h	0	0	0	0	0	DAY			Day	01–07
04h	0	0	10 Date		Date				Date	01–31
05h	0	0	0	10 Month	Month				Month	01–12
06h	10 Year				Year				Year	00–99
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h–3Fh									RAM 56 x 8	00h–FFh

0 = Always reads back as 0.

You can use memory locations from 08h-3Fh for storage.
The **7-bit slave address of DS1307 is 1101000**.

Control Bits

- **OUT (Output control):** This bit controls the output level of the SQW/OUT pin when the square wave output is disabled. If SQWE = 0, the logic level on the SQW/OUT pin is 1 if OUT = 1 and is 0 if OUT = 0.
- **SQWE (Square Wave Enable):** This bit, when set to a logic 1, will enable the oscillator output. The frequency of the square wave output depends upon the value of the RS0 and RS1 bits. With the square wave output set to 1Hz, the clock registers update on the falling edge of the square wave.
- **RS (Rate Select):** These bits control the frequency of the square wave output when the square wave output has been enabled.

SQUAREWAVE OUTPUT FREQUENCY

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

PIC18F registers for I2C

- Initialize the following Registers:
  - SSPCON1 - Select Various modes for I2C.
  - SSPCON2 - Ack/Nack, Start/Stop and Repeated Start signals.
  - SSPSTAT - Select Standard mode (100 kHz).
  - SSPBUF - Stores Received and Transmitted values.
  - SSPADD - Define the baud rate

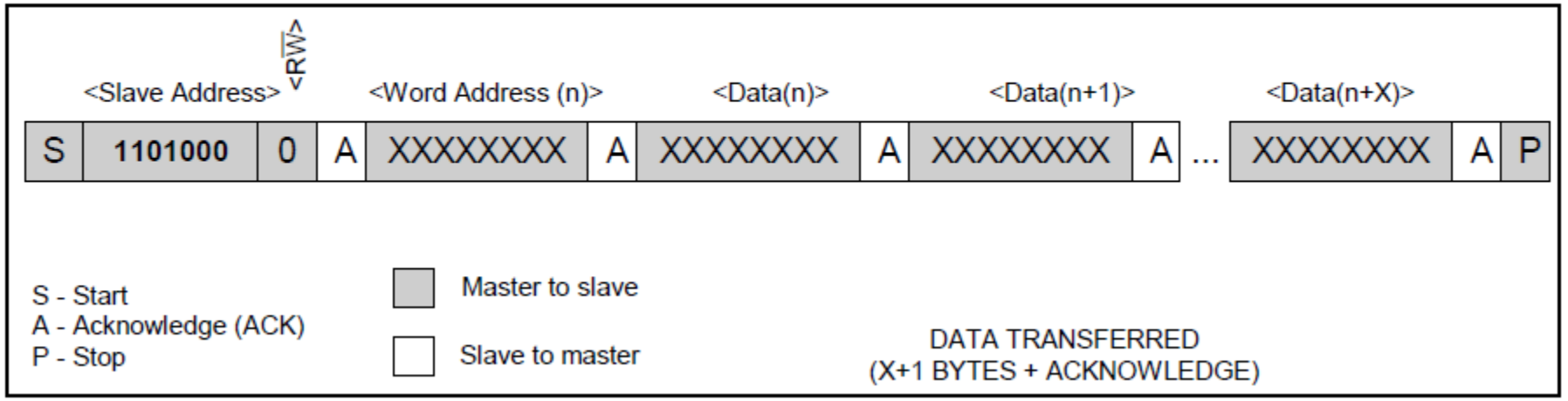
  - SSPIF - Flag to check for completion of the I2C operation

I2C Operation for DS1307

- The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the direction bit (*R/w) which, for a read, is a 1.

- Every 9th bit from start is ACK bit coming from Slave

Figure : Data Write—Slave Receiver Mode



I2C Operation for DS1307

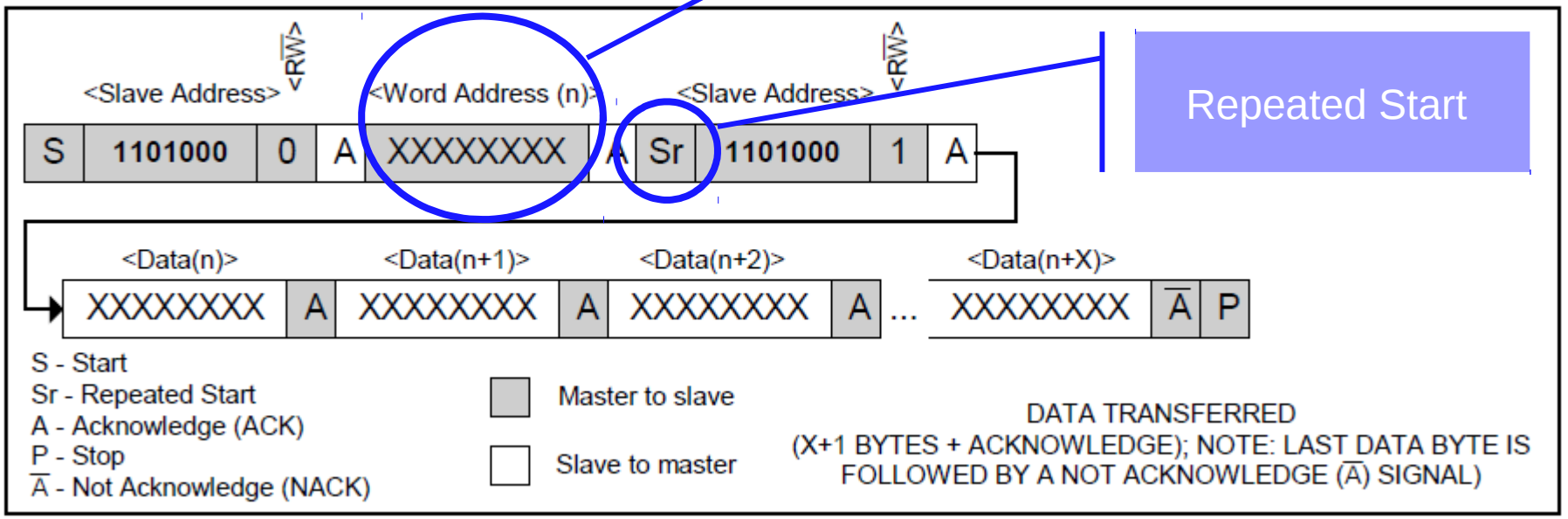
- The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the direction bit (*R/w) which, for a read, is a 1.

- Every 9th bit from start is ACK bit coming from Slave

Location address which is latched inside the device for providing data on next read

Repeated Start

Figure : Data Read (Write Pointer, Then Read)



Thank You