Interfacing using Serial Protocols Using SPI and I2C

Presented by:

K.Pratyush Aditya (200201024)

Vivek Rathi (200201025)

Praneet Koppula (200201032)

Embedded Communication System

- Based on Interconnect
  - -Point-to-point network (data link)
  - -Shared media networks (data highways)

- Based on model
  - -State based Communication System
  - -Event based Communication System

- Based on Bits transferred
  - -Serial Communication
  - -Parallel Communication

Parallel vs. Serial

- Really fast
- Usage of large numbers of pins
  - Requires bigger ICs
- Large number of wires/traces
  - -Space requirements on board (imagine modella)
  - -Large bundle of wires if off board

- Minimum number of pins (1 to 3) and wires
- Requires inter-device agreements and synchronization
  - Speed (baud rate)
  - Bit framing
- Requires extra software or hardware (USART)

Serial Port Communication: Benefits

To interface with a PC, during development and/or in the field

Most PC have serial interface to connect to the peripherals.

Serial interfaces allow processors to communicate without the need for shared memory and semaphores, and the problems they can create.

Serial Port Communication: Benefits

Benefit of serial communications is low pin counts

Can be performed with just one I/O pin, compared to eight or more for parallel communications

Many common embedded system peripherals, such as A2D and D2A converters, LCDs, and temperature sensors,support serial interfaces

Protocols in Serial Port Communication

Name	Sync/ Async	Type	Duplex	Max Devices	Pin Count
RS-232	Async	Peer	Full	2	2
I2C	Sync	Multi-Master	Half	N*	2
SPI	Sync	Multi-Master	Full	N*	3+
Microwire	Sync	Master/Slave	Full	N*	3+

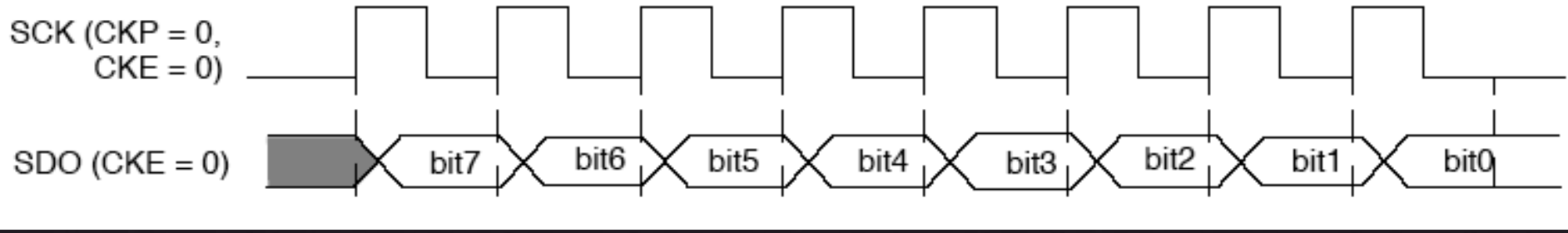
Multi-Master vs. Master/Slave vs. Peer

• Master/slave describes a bus where one device is the master and others are slaves. Master/slave buses are usually synchronous, as the master often supplies the timing clock for data being sent along in both directions.

• A multi-master bus is a master/slave bus that may have more than one master. These buses must have an arbitration scheme that can settle conflicts when
more than one master wants to control the bus at the same time.

• Point-to-point or peer interfaces are where two devices have a peer relation to each other; there are no masters or slaves. Peer interfaces are most often
asynchronous.

Synchronous Serial



- Clock on one wire, and synchronized data on other
- Usually used to connect microcontrollers/processors to peripherals
- SPI (Serial Peripheral Interface) -- 1 wire for each dir + 1 clock wire
- I2C (Inter IC) -- 1 wire for bidirectional data (direction is handled by protocol) + 1 clock wire

I2C Inter-IC

- **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 I^2C Interfaces**
  - Microcontroller, EEPROM, Real-Timer, interface chips, LCD driver, A/D converter

What is I^2C used for?

- **Data transfer between ICs and systems at relatively low rates**
  - "Classic" I^2C 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

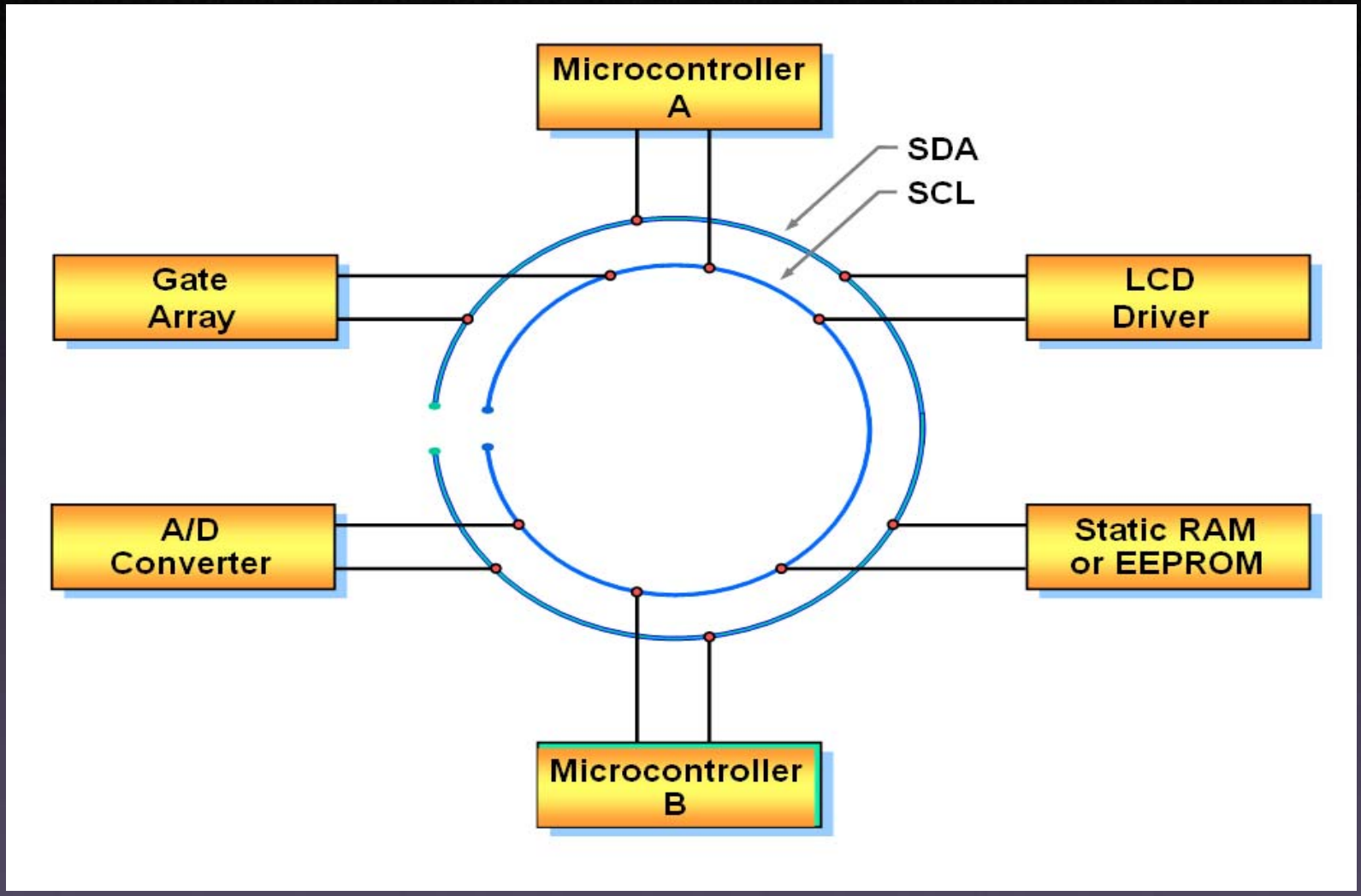
I^2C Bus Characteristics

- **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**

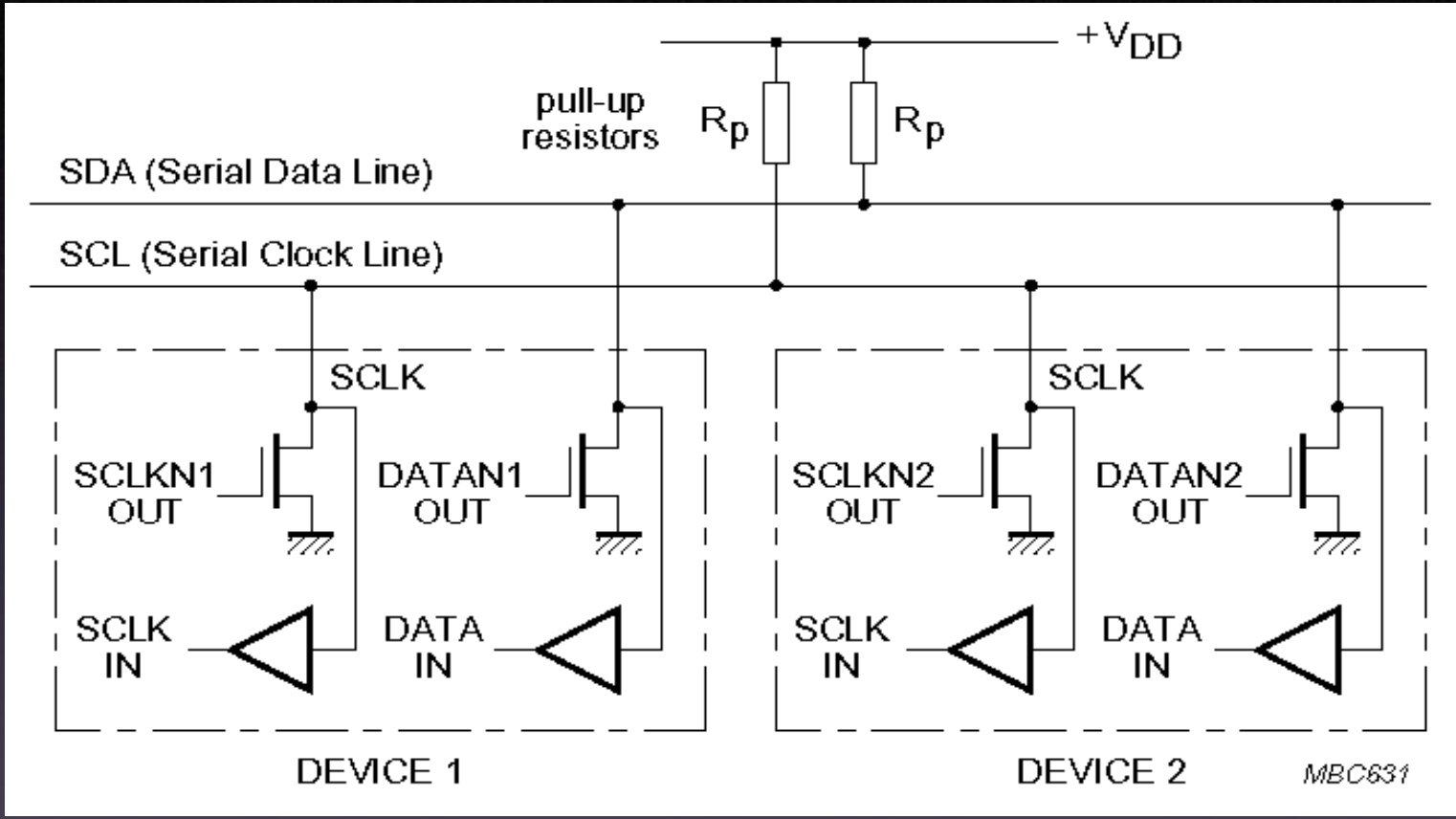
I^2C Bus Characteristics (cont'd)

- **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)**
- **Compatible with different IC technologies**

I²C Bus Configuration Example



I²C Electrical Aspects



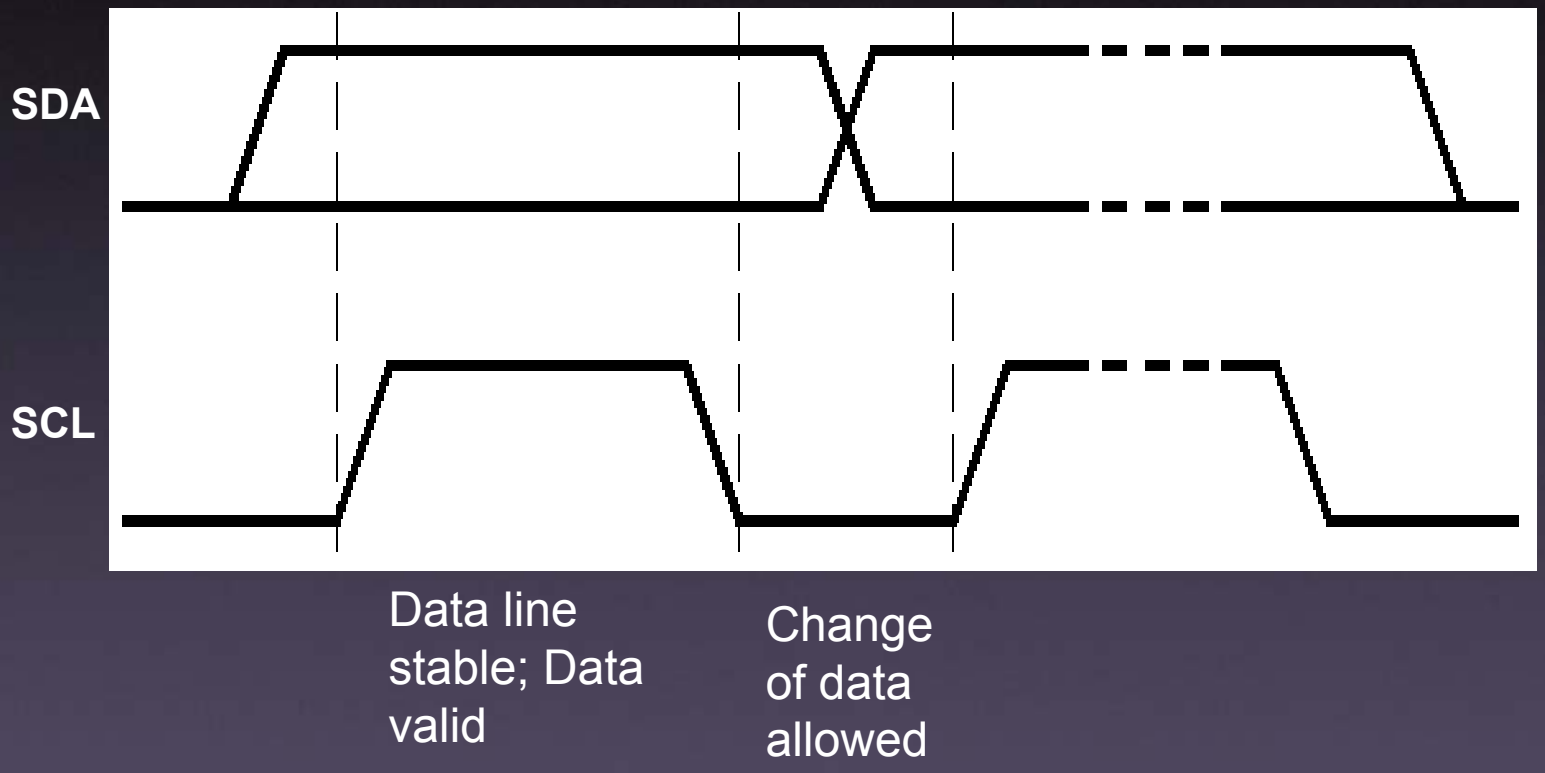
- I²C devices are wire ANDed together.
- If any single node writes a zero, the entire line is zero

I^2C 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

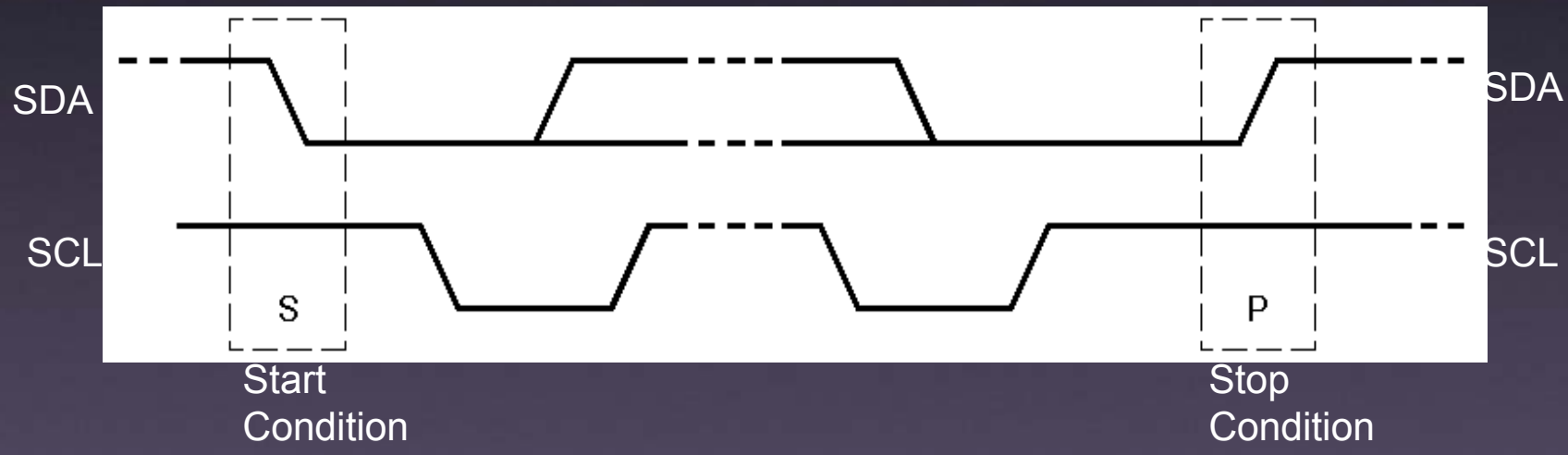
Bit Transfer on the I²C Bus

- 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

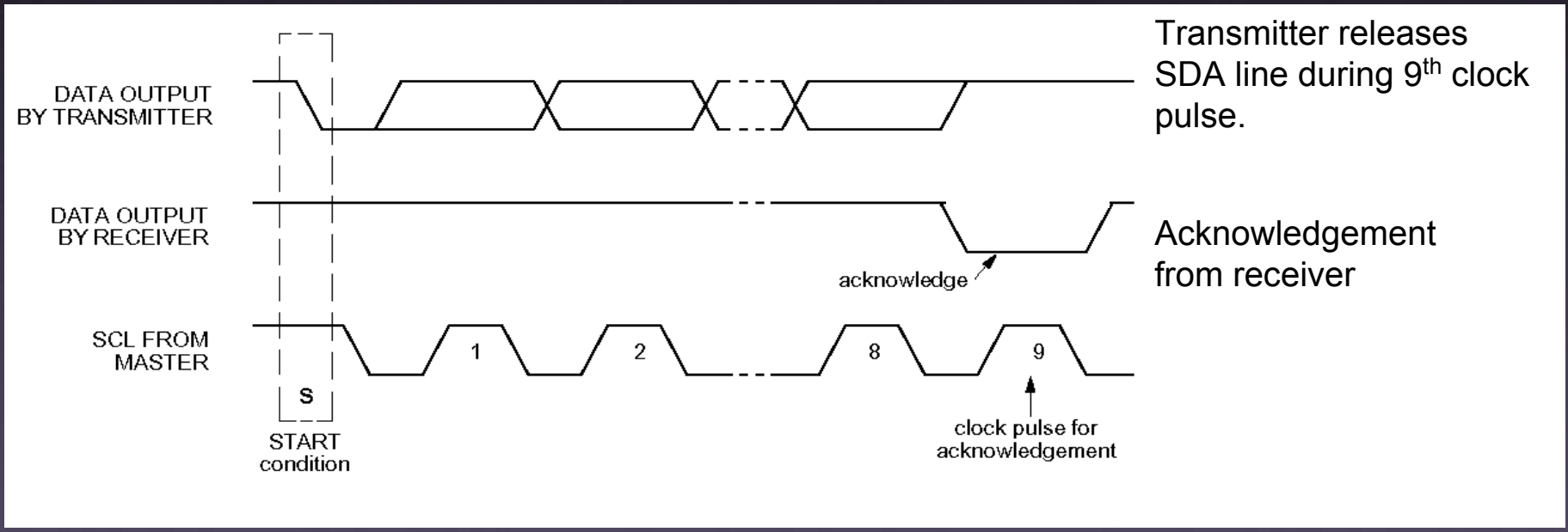
0 – Slave written to by Master

1 – Slave read by Master

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

Data Transfer on the I2C Bus

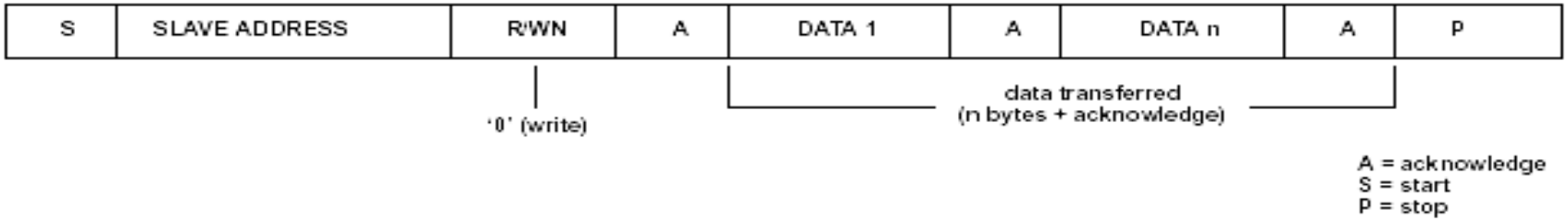
- **Start Condition**
- **Slave address + R/W**
  - Slave acknowledges with ACK
- **All data bytes**
  - Each followed by ACK
- **Stop Condition**



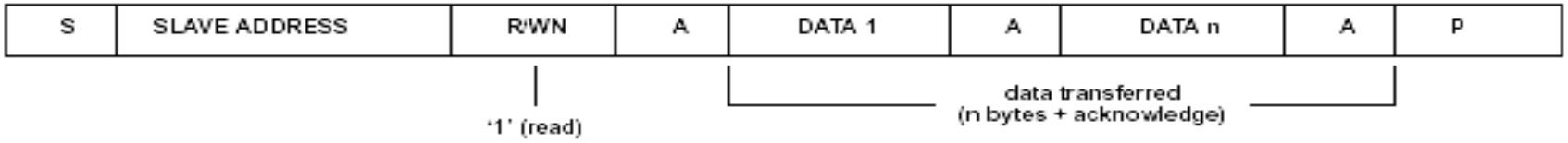
Remember : Clock is produced by Master

Possible Data Formats

a. Master transmitter to slave receiver. Direction is not changed.

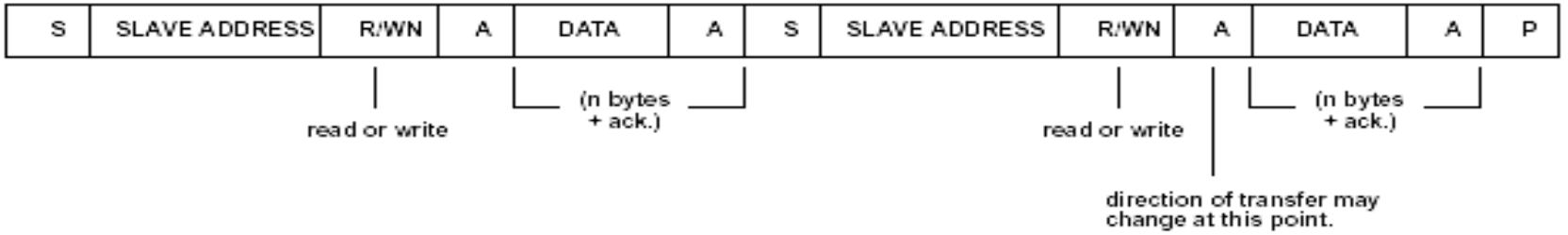


b. Master reads slave immediately after first byte.



At the moment of the first acknowledge, the master transmitter becomes a master receiver and the slave receiver becomes a slave transmitter. This acknowledge is still generated by the slave.

The STOP condition is generated by the master.



During a change of direction within a transfer, the START condition and the slave address are both repeated, but the R/W bit reversed. Start, stop, slave addresses and R/W bits are generated by the master.

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.)

What is SPI?

- SPI stands for Serial Peripheral Interfacing.

- It is a synchronous serial bus developed by Motorola and is present on many of their micro controllers.

- SPI is used to communicate with peripheral devices.

Why SPI?

- As compared with its counterpart I^2C, SPI is more suited for data stream applications.

  Communication between DSPs.
  ADC

- SPI can also achieve significantly higher data rates than I^2C.

- Full Duplex capability.

SPI Bus Configuration.

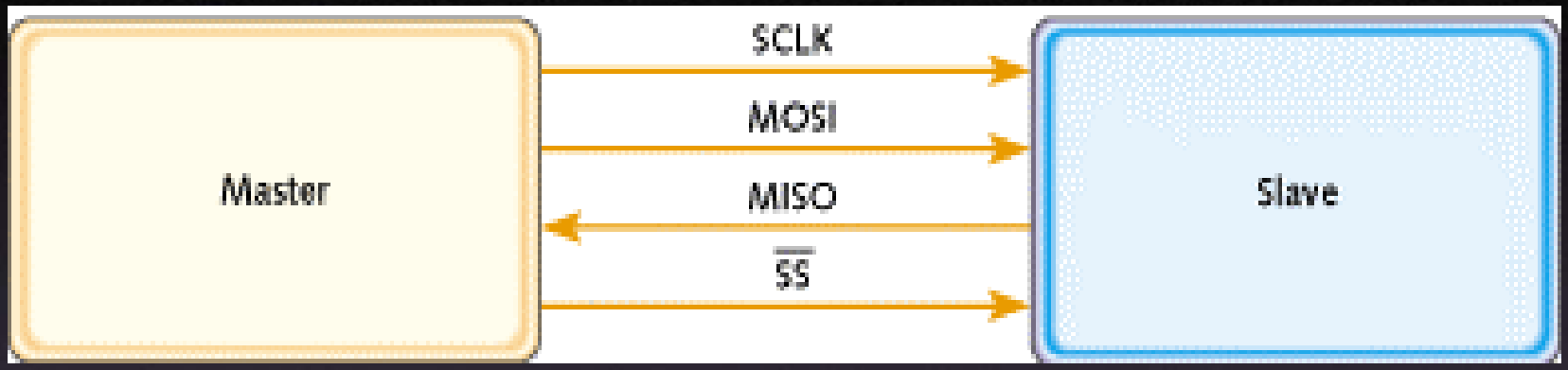


Fig 1.0 Single master, single slave SPI implementation

- Devices communicate using a master/slave relationship.

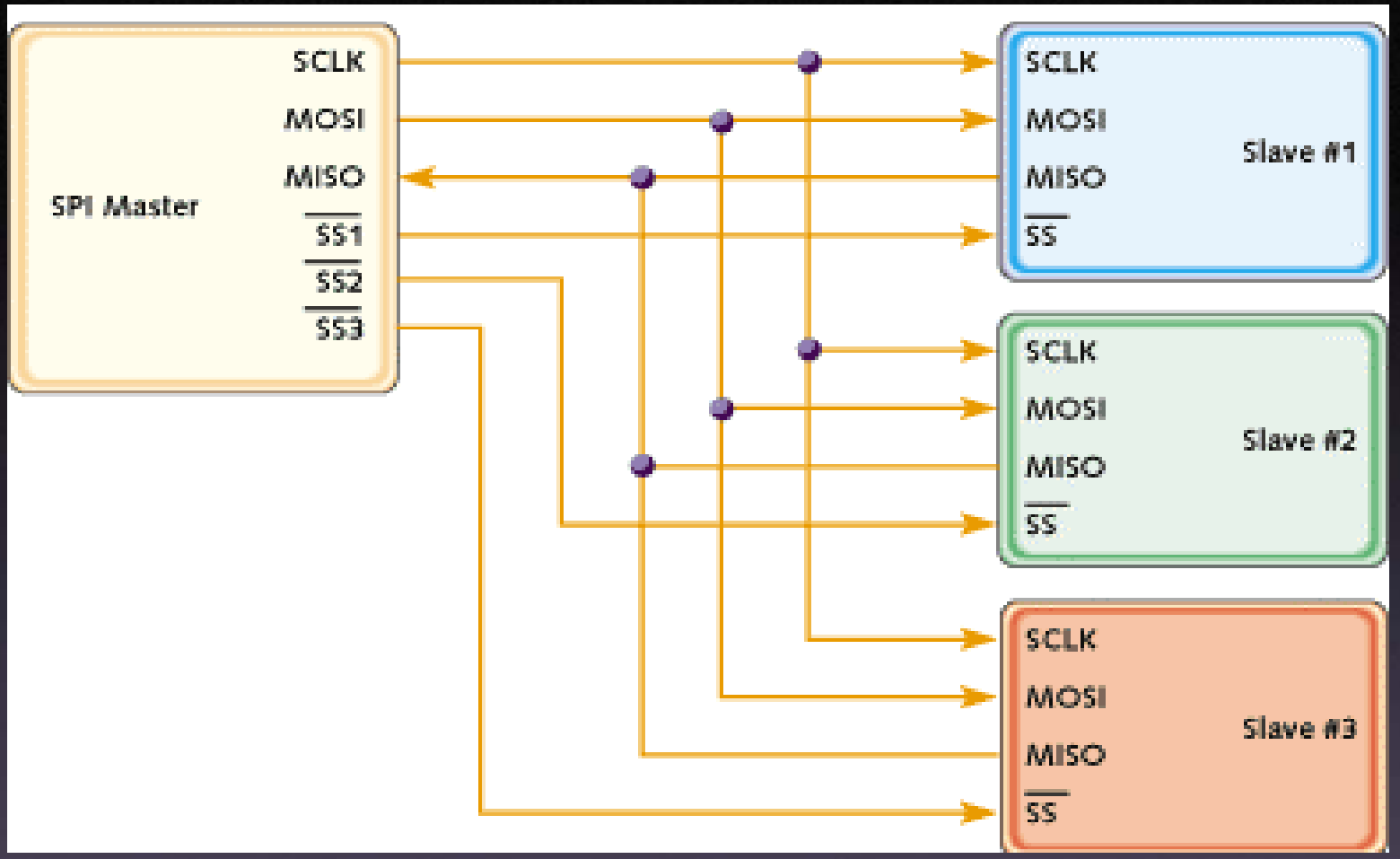


Fig 2.0 Single master, multiple slave SPI implementation

- SPI defines four types of signals.

SPI Signal	Name
MISO	Master In Slave Out
MOSI	Master Out Salve In
SCLK	Serial Clock
SS	Slave Select

SPI Registers

- There are three registers unique to the serial peripheral interface which provide control, status, and data storage.

- The Serial Peripheral Control Register (SPCR), shown in Fig 3, provides control for the SPI.

Bit	7	6	5	4	3	2	1	0	
	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

SPI Control Register

- SPIE : Serial Peripheral Interrupt Enable.
- SPE : Serial Peripheral System Enable.
- DORD : Data Order.
- MSTR : Master Mode Select.
- CPOL : Clock Polarity.
- CPHA : Clock Phase.
- SPR0 : SPI Clock Rate select 0.
- SPR1 : SPI Clock Rate select 1.

SPI Data Register

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	SPDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	X	X	X	X	X	X	X	X	Undefined

The SPI Data Register is a Read/Write Register used for data transfer between the register file and the SPI Shift Register. Writing to the register initiates data transmission. Reading the register causes the Shift Register Receive buffer to be read.

SPI Status Register

Bit	7	6	5	4	3	2	1	0	
	SPIF	WCOL	-	-	-	-	-	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- SPIF : SPI Interrupt Flag
- WCOL : Write Collision.
- SPI2X : To Double the data transmission rate.

SPI Advantages and Disadvantages

•SPI does not have an acknowledgement mechanism to confirm receipt of data. In fact, without a communication protocol, the SPI master has no knowledge of whether a slave even exists. SPI also offers no flow control. If you need hardware flow control, you might need to do something outside of SPI.

•Slaves can be thought of as input/output devices of the master. SPI does not specify a particular higher-level protocol for master-slave dialog. In some applications, a higher-level protocol is not needed and only raw data are exchanged. An example of this is an interface to a simple codec. In other applications, a higher-level protocol, such as a command-response protocol, may be necessary. Note that the master must initiate the frames for both its command and the slave's response.

SPI Advantages and Disadvantages Contd..

- SPI's full duplex communication capability and data rates (ranging up to several megabits per second) make it, in most cases, extremely simple and efficient for single master, single slave applications. On the other hand, it can be troublesome to implement for more than one slave, due to its lack of built-in addressing; and the complexity only grows as the number of slaves increases.

Patent Issues

Philips maintains a license is required for implementing an I2C interface on a chip (IC, ASIC, FPGA, etc). And it is Philips's position that all chips that can talk to the I2C bus must be licensed. It doesn't matter how this interface is implemented. The licensed manufacturer may use firmware bit banging, in-house IP, or purchased IP cores.

The SPI bus is broadly accepted because it has little or no patent issues. This is partly because Motorola, its' creator, provides no specification or central support. Those applying SPI can create hardware and software solutions without patent issues, but also without support or definition of supporting protocols.

Conclusions

References

"Application Note: Using the Serial Peripheral Interface to Communicate between multiple microcomputers", Motorola, ANN91/D Rev. 1, 01/2001.

"Introduction to Serial Peripheral Interface", by David Kalinsky and Roee Kalinsky, Embedded Systems Programming, 02/01/2002.

"Introduction to I2C", by David Kalinsky and Roee Kalinsky, Embedded Systems Programming, 07/31/2001.

"PCF8584 I2C Bus Controller", Philips Semiconductor, Oct 21, 1997.

"The I2C Bus Specification", Version 2.1, Jan 2000, Philips Semiconductor.

http://www.epanorama.net/links/serialbus.html

http://www.i2cchip.com/

Thank You