Communication

Press Play: Interactive Device Design | July 21, 2011

Check In: MP3 Player Designs

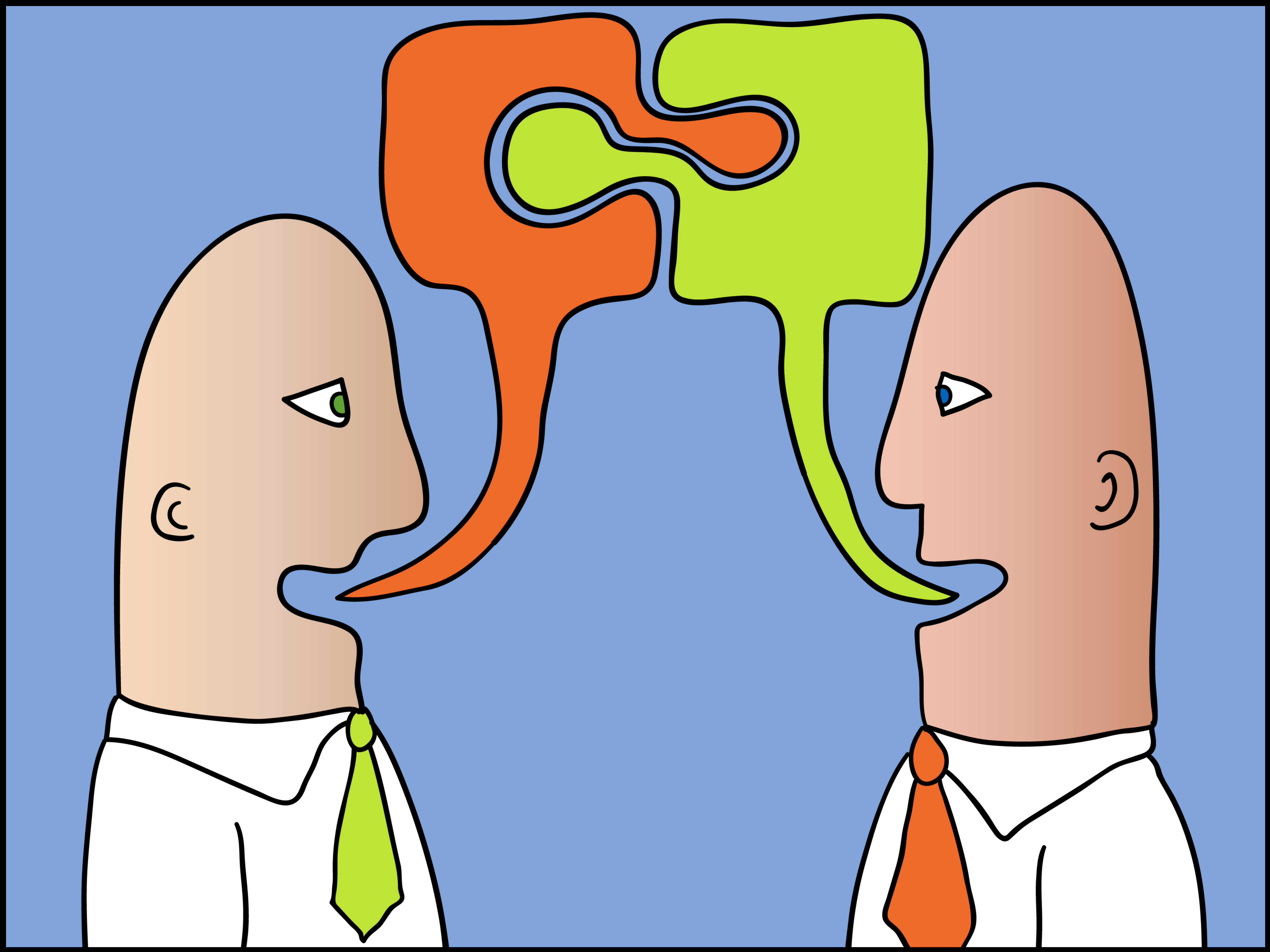
Start with the big picture: what's it going to do?

Mine your observations or own needs for ideas.

Given that, what functions should it perform?

What interactions do those functions imply?

Map/sketch/diagram how those will work.



Context of Communication

Conversation - Rules of Conduct

Communication is holding a conversation

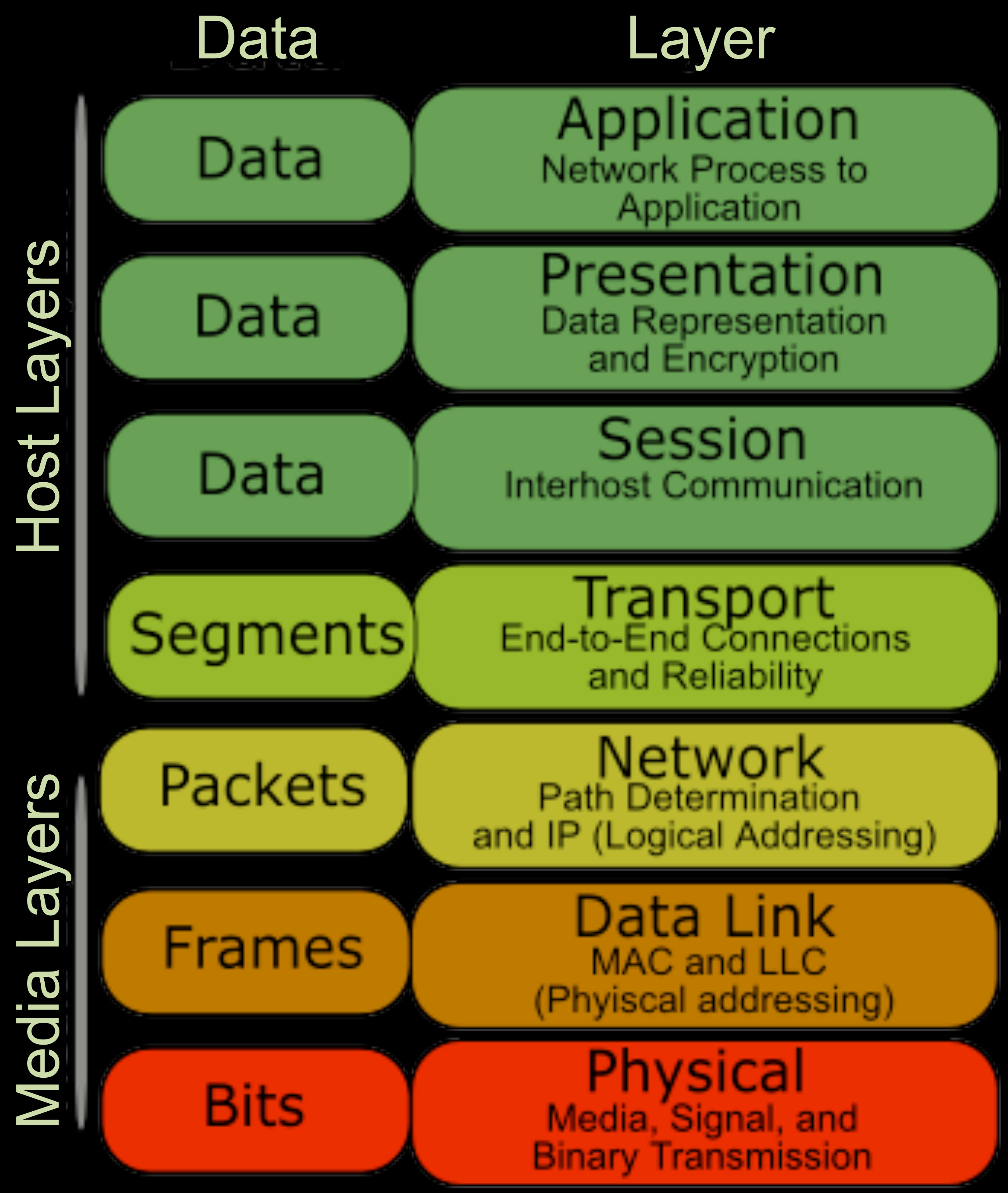
- □ Inter-processor communication is 'peer-to-peer'
- □ Processor-to-device conversation is 'master-slave'

A protocol is a set of rules of conduct that we agree to uphold during the conversation

- □ It governs how we start a conversation, who speaks when, how fast, how often, etc.

Context of Communication

Open Systems Interconnection



	Data	Layer
Host Layers	Data	Application Network Process to Application
	Data	Presentation Data Representation and Encryption
	Data	Session Interhost Communication
	Segments	Transport End-to-End Connections and Reliability
Media Layers	Packets	Network Path Determination and IP (Logical Addressing)
	Frames	Data Link MAC and LLC (Phyiscal addressing)
	Bits	Physical Media, Signal, and Binary Transmission

Serial, WiFi, Ethernet

Voltage, Wires, Timing

Types of Interface

Parallel

Examples

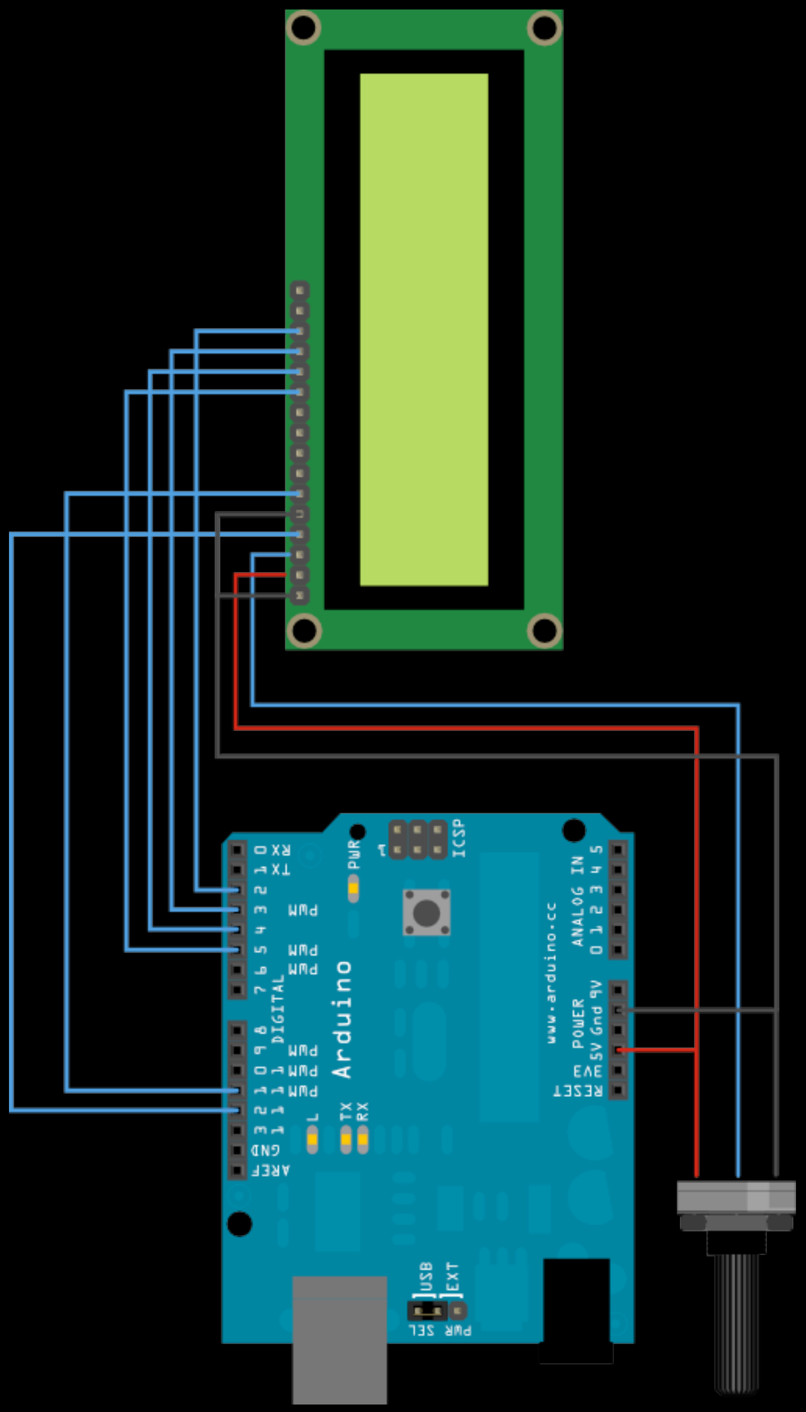
- □ Graphical LCD
- □ SCSI, Firewire

Advantages

- □ Faster in Theory

Drawbacks

- □ Crosstalk
- □ Clock Skew
- □ Wire per Bit



Types of Interface

Serial

Examples

- USB, SATA
- SPI and I^{2}C

Advantages

- Clock Faster
- Fewer Wires

Drawbacks

- Overhead of Negotiation

Conserve Resources

Ride the Bus

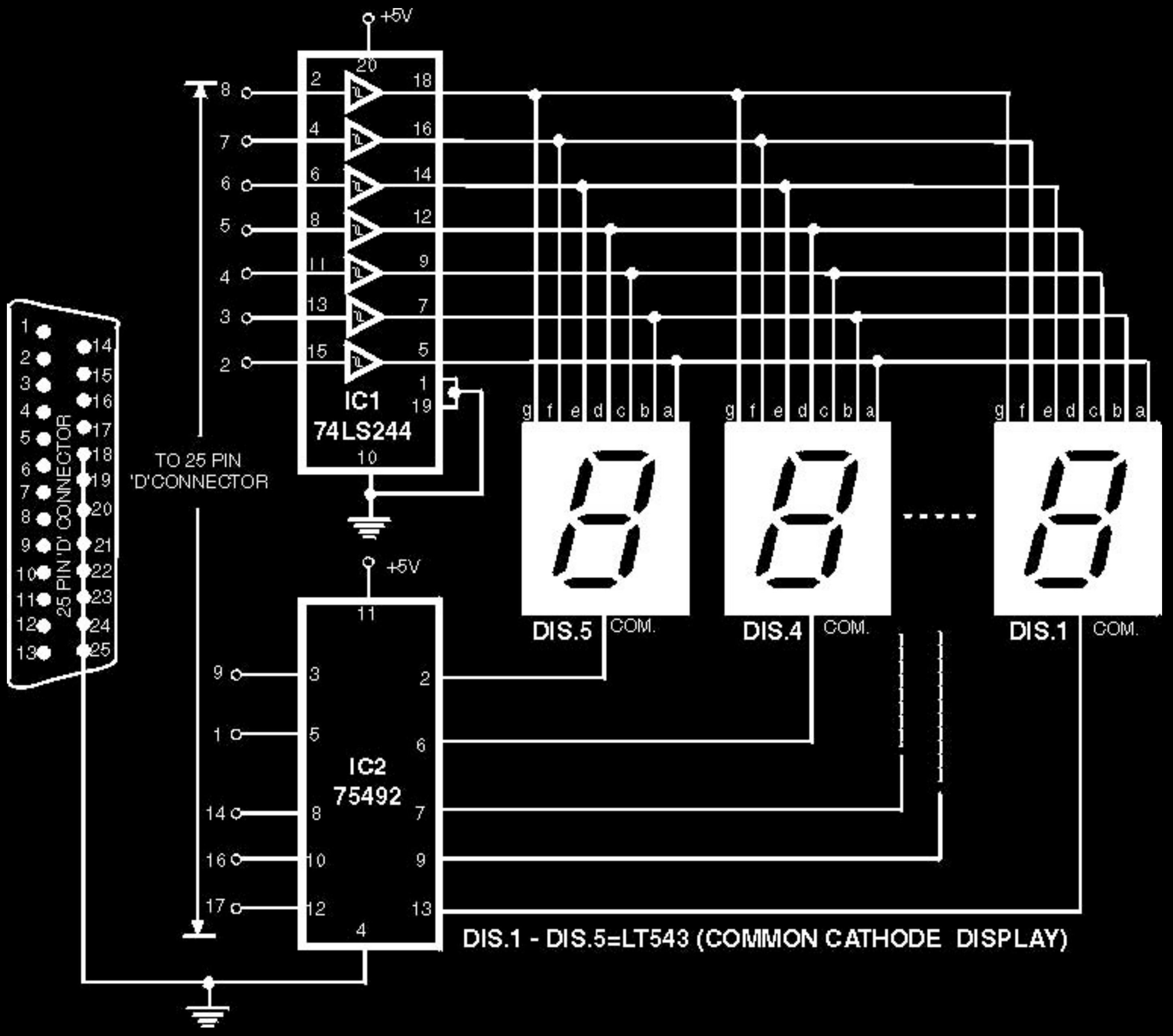
Types of Interface

□ Parallel or Serial

Internal or External

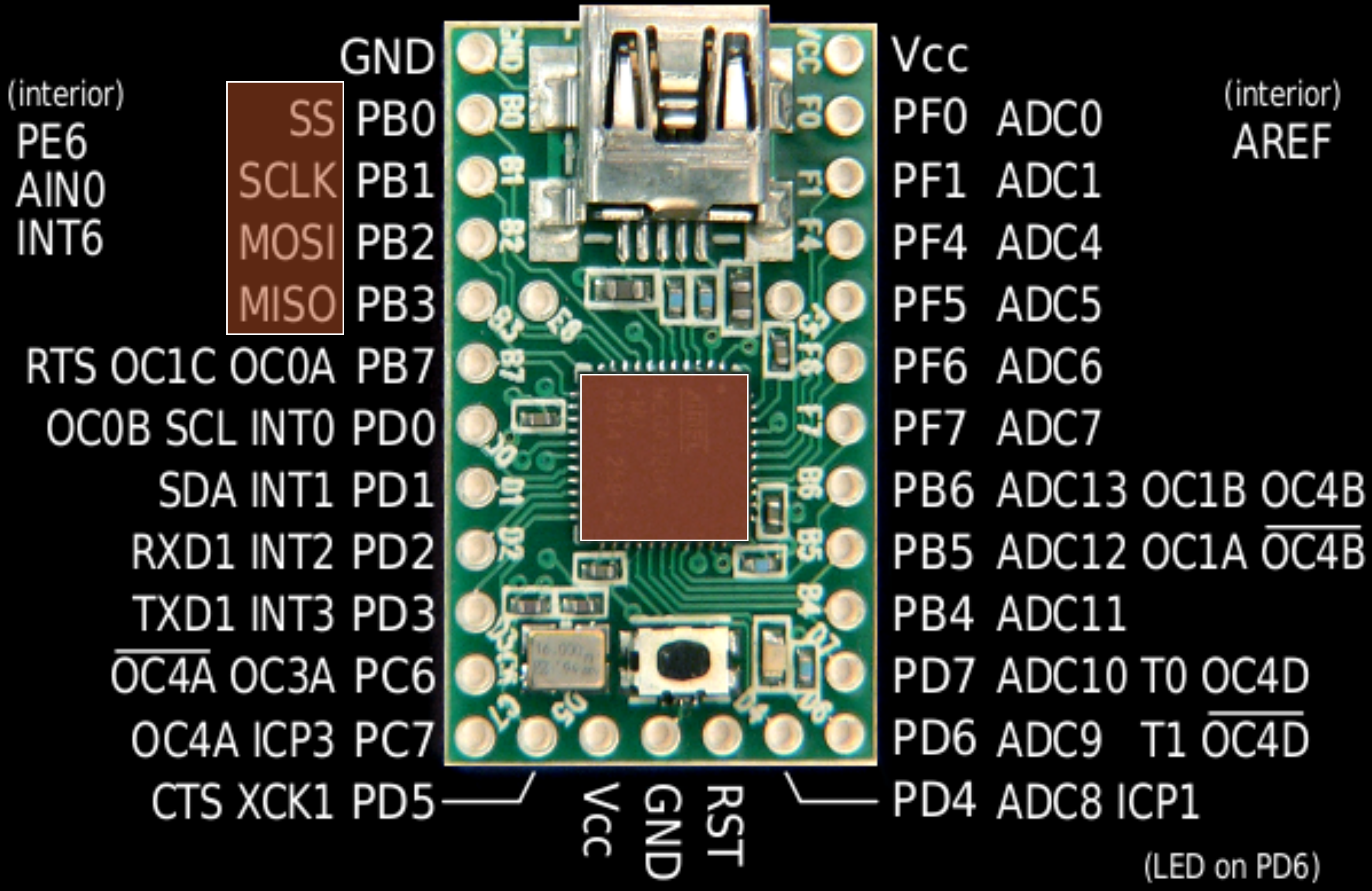
One Line per Device

□ Chip Select



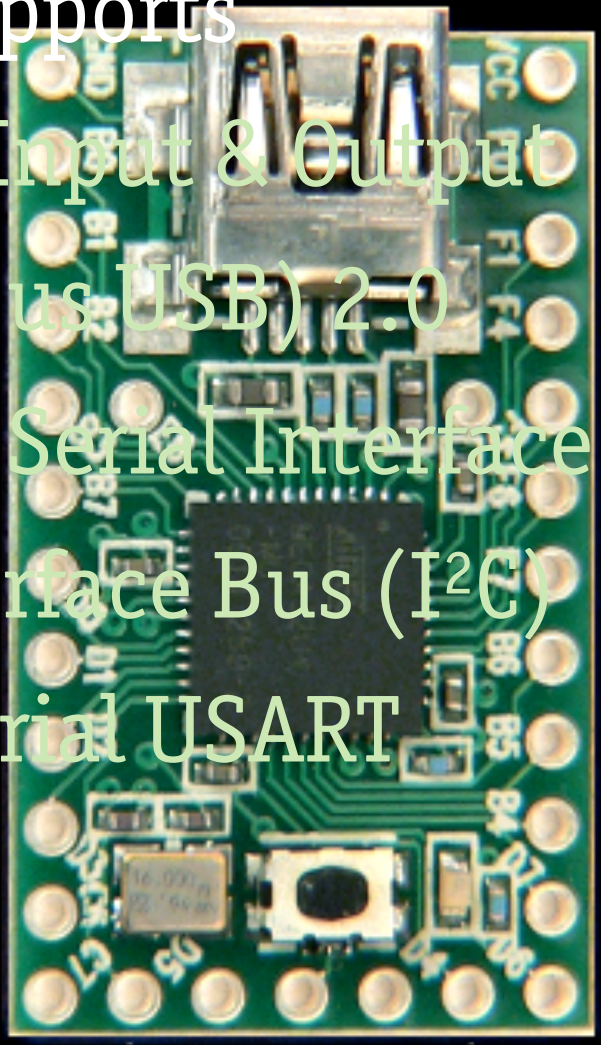
Microprocessor
Communication

SPI Bus ATmega32U4



Microprocessor

Communication



The ATmega32U4 Supports

□ Digital & Analog Input & Output

□ Universal Serial Bus USB) 2.0

□ Master/Slave SPI Serial Interface

□ 2-Wire Serial Interface Bus (I²C)

□ Programmable Serial USART

How Does an MCU Communicate?

Bits and Bytes

1 Bit

0 = LO

1 = HI

2 Bits

00 = 0

01 = 1

10 = 2

11 = 3

3 Bits

000 = 0 ◀

001 = 1

010 = 2 ◀

011 = 3

100 = 4 ◀

101 = 5

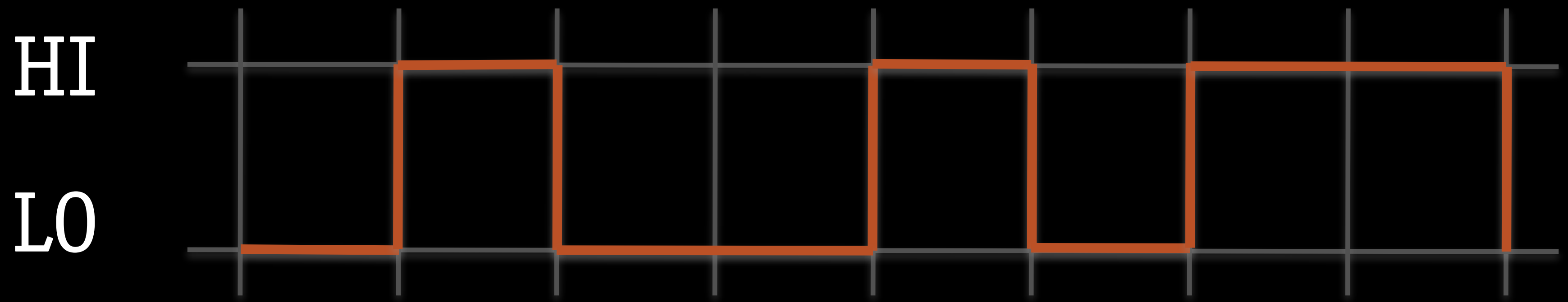
110 = 6

111 = 7

4 Bits

1000 = 8 ◀

...



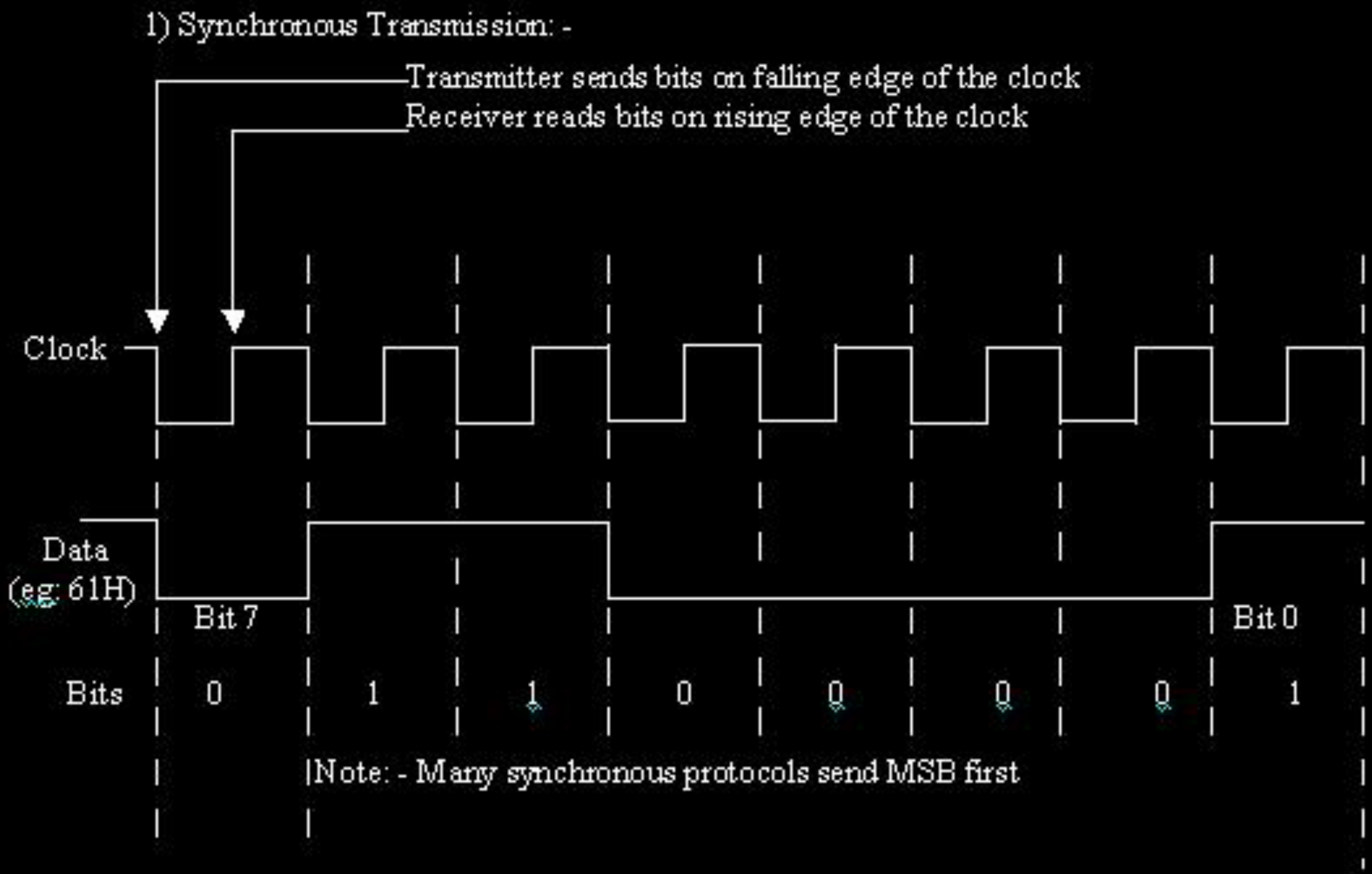
How Does an MCU Communicate?

Parallel and Serial

Synchronous and Asynchronous

Serial Communication

Synchronous



1) Synchronous Transmission: -

Transmitter sends bits on falling edge of the clock

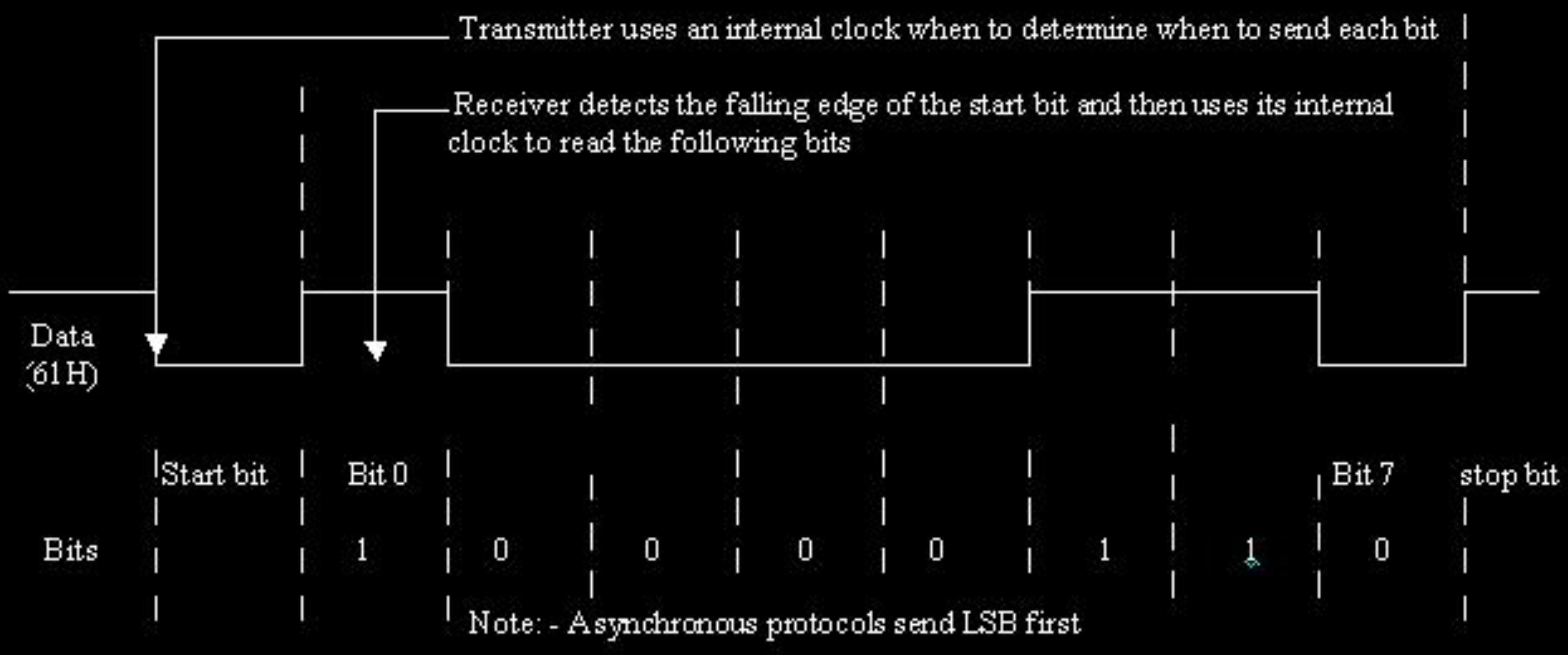
Receiver reads bits on rising edge of the clock

Note: - Many synchronous protocols send MSB first

Serial Communication

Asynchronous

2) Asynchronous Transmission: -



Transmitter uses an internal clock when to determine when to send each bit

Receiver detects the falling edge of the start bit and then uses its internal clock to read the following bits

Note: - Asynchronous protocols send LSB first

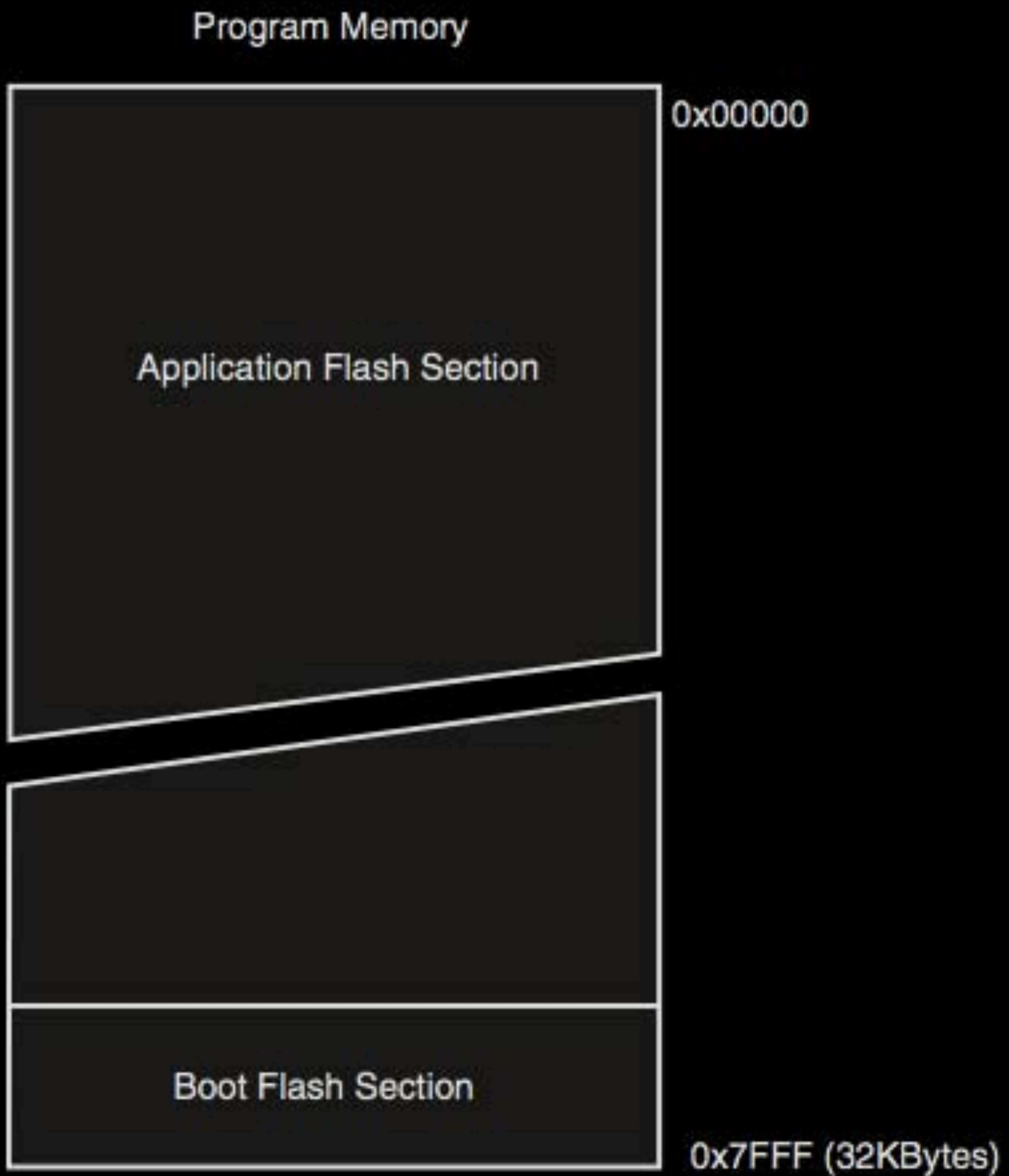
Serial Peripheral Interface

Configuration and Use

- □ How do we configure the MCU for SPI?
- □ How do we use SPI to communicate?

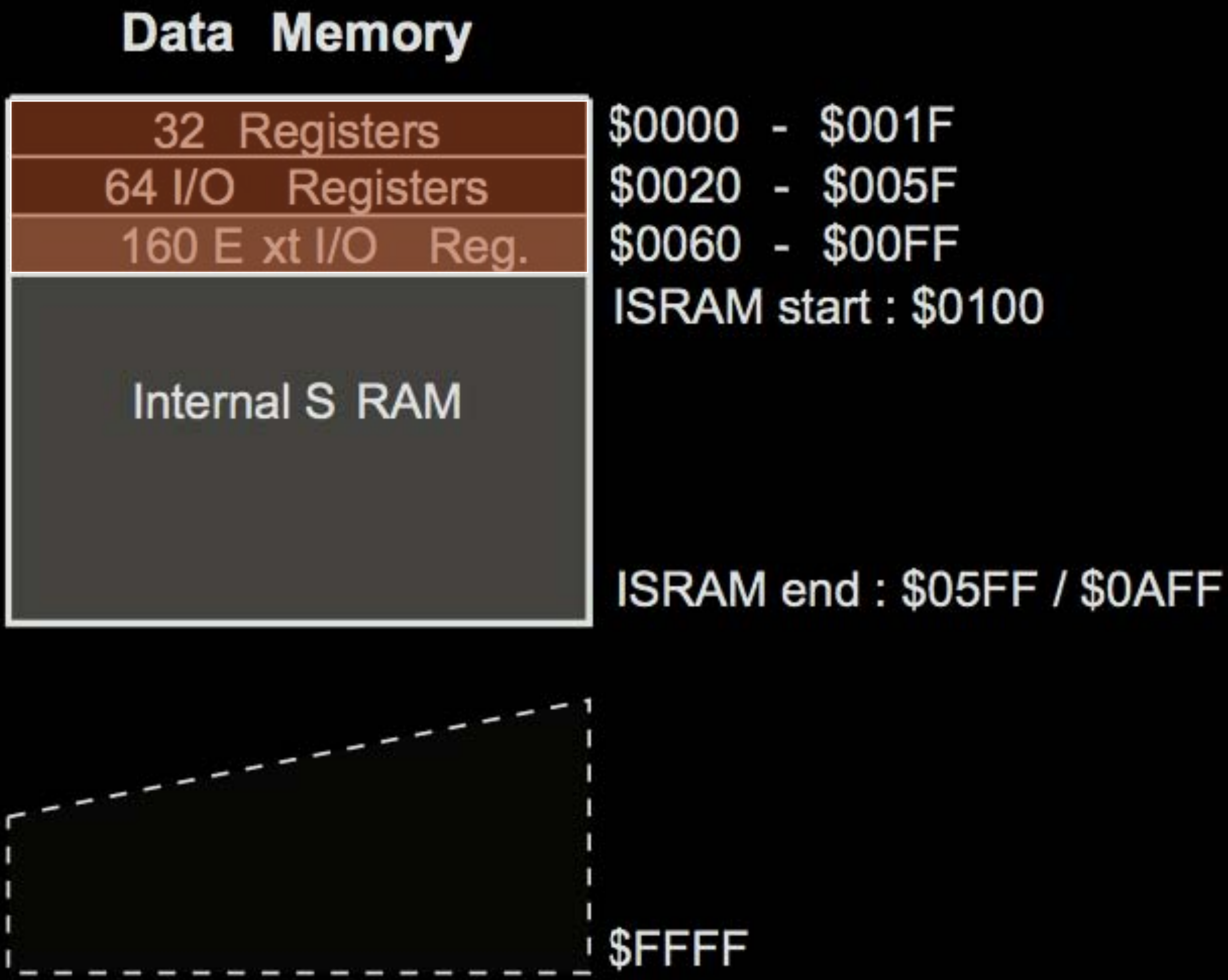
ATmega32U4 Memory Map

Program Memory



ATmega32U4 Memory Map

Data Memory



Data Memory

32 Registers $0000 - $001F

64 I/O Registers $0020 - $005F

160 E xt I/O Reg. $0060 - $00FF

ISRAM start : $0100

Internal S RAM

ISRAM end : $05FF / $0AFF

$FFFF

ATmega32U4 Memory Map

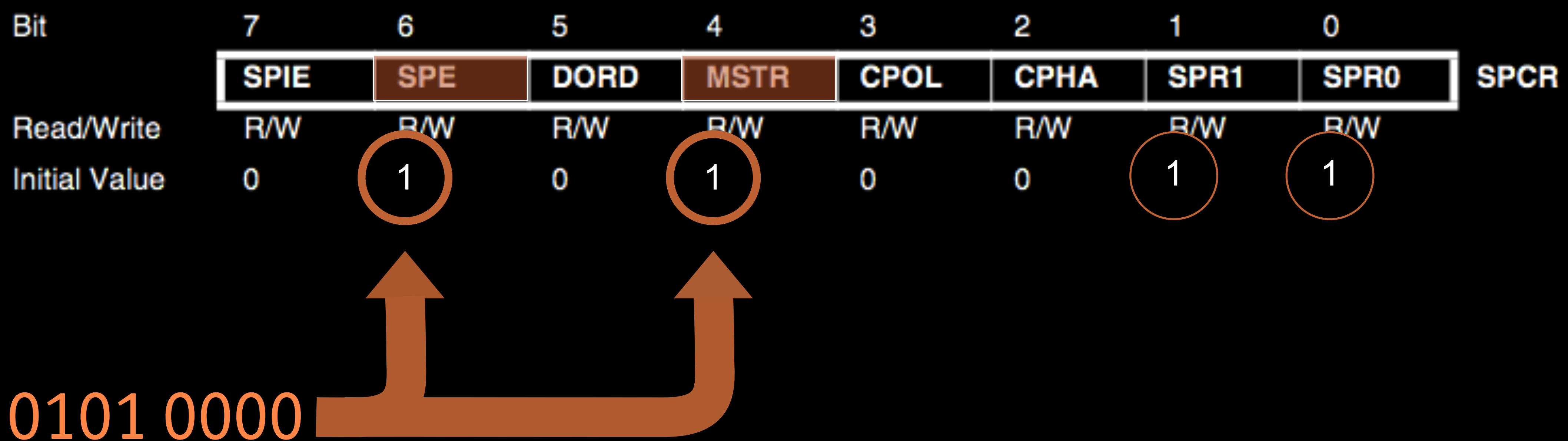
Access Data & Settings

- A register is a small amount of storage available on the CPU whose contents can be accessed more quickly than storage available elsewhere.
- Certain registers are reserved for MCU operating settings.
- Writing values into these registers changes the settings.

Serial Peripheral Interface

An Example of Configuration

SPI Control Register – SPCR



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	1	0	1	0	0	1	1	

Set SPCR = 0101 0000

SPE – "Enable SPI mode"

MSTR – "I control the clock"

Settings Registers

Bitwise Operators

How to set bits without over-writing an entire register?

Bitwise Operators

AND = “&”
OR = “|”
NOT = “~”
XOR = “^”

Bitwise AND

0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1

Bitwise OR

0 | 0 = 0
0 | 1 = 1
1 | 0 = 1
1 | 1 = 1

We set the SPCR using a bitwise OR: SPCR |= 0101 0000

Serial Peripheral Interface

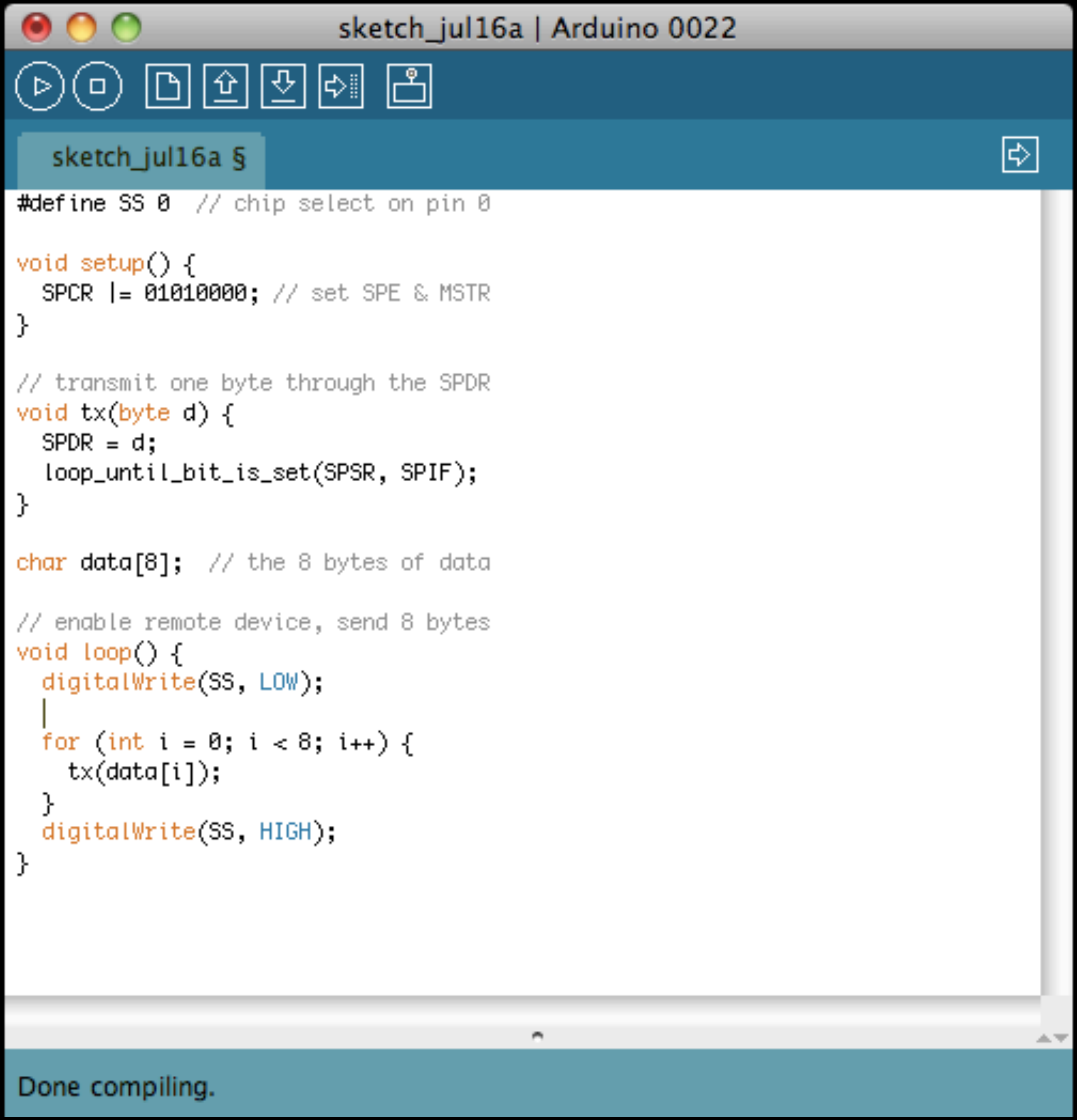
An Example of Use

Where are:

- □ SCLK
- □ MOSI
- □ MISO

They are pre-assigned

We only connect them

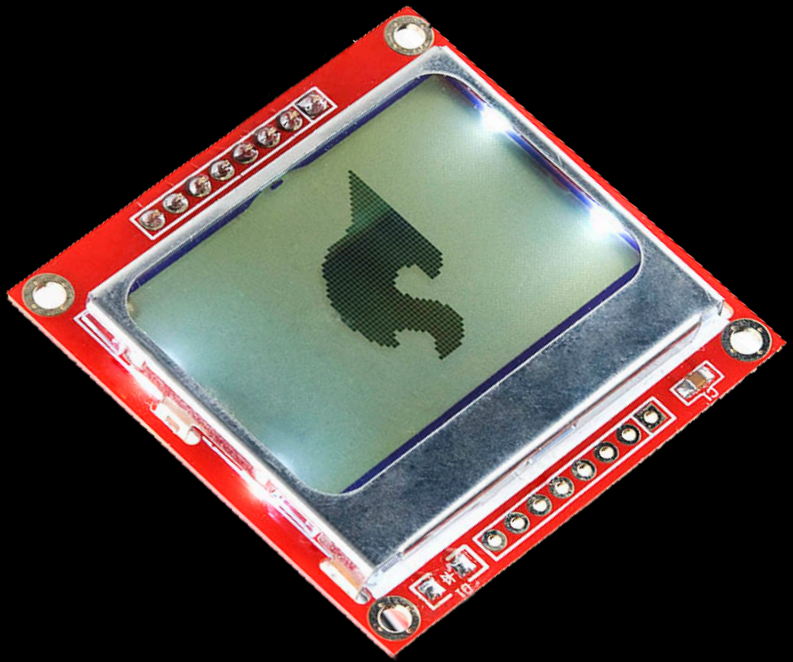


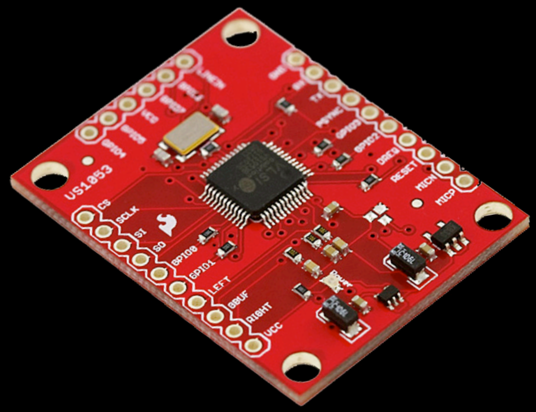
```
#define SS 0  // chip select on pin 0

void setup() {
  SPCR |= 01010000; // set SPE & MSTR
}

// transmit one byte through the SPDR
void tx(byte d) {
  SPDR = d;
  loop_until_bit_is_set(SPSR, SPIF);
}

char data[8];  // the 8 bytes of data

// enable remote device, send 8 bytes
void loop() {
  digitalWrite(SS, LOW);

  for (int i = 0; i < 8; i++) {
    tx(data[i]);
  }
  digitalWrite(SS, HIGH);
}
```

Done compiling.

Communication Devices

Hardware You Can Use





Lab 5 Preview: Etch-a-Sketch

Modify Your Teensy to Output 3.3V

Writing to a Graphical LCD (GLCD)

Using a microSD Card for Storage