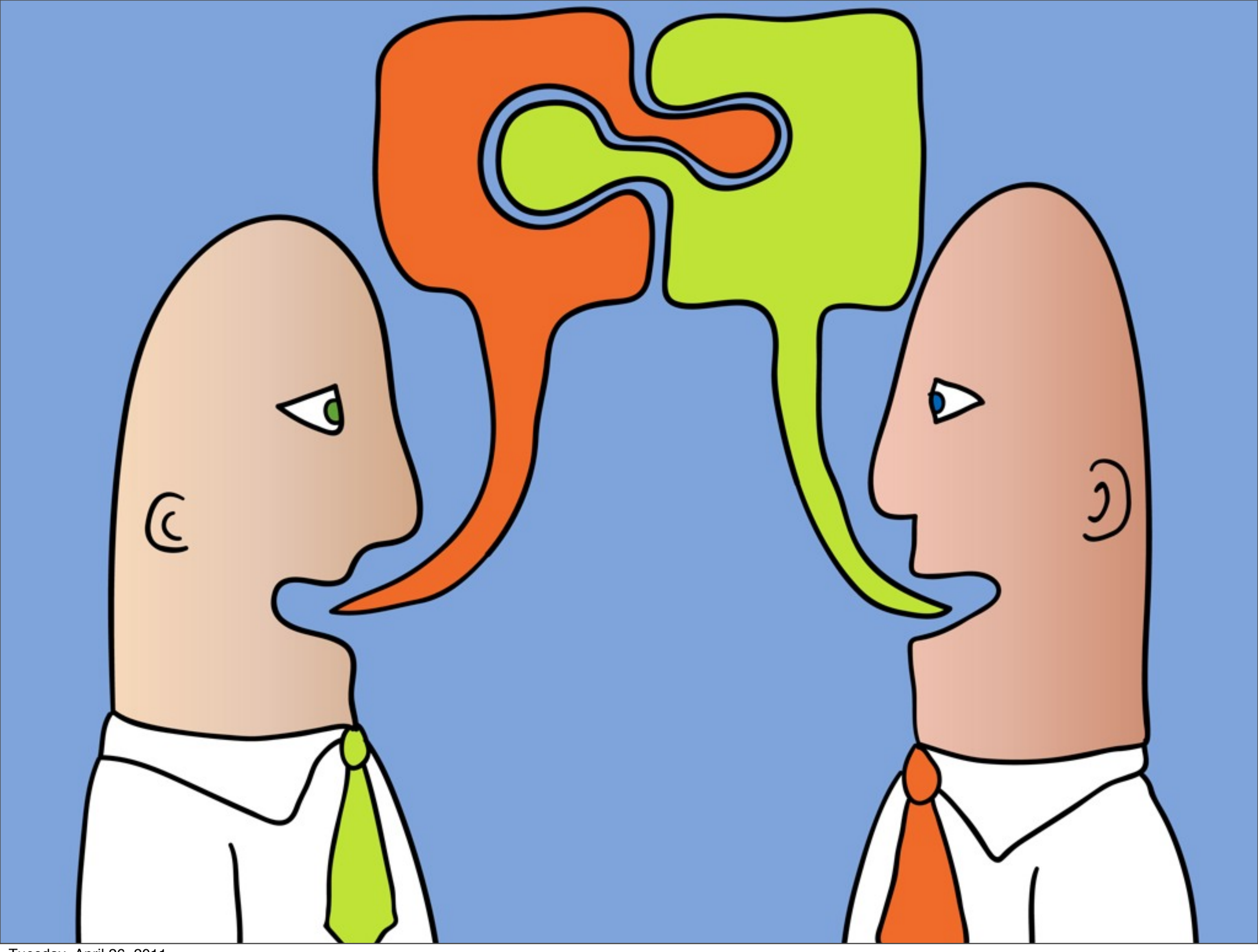
Communication

Press Play: Interactive Device Design | April 25, 2011



Context of Communication

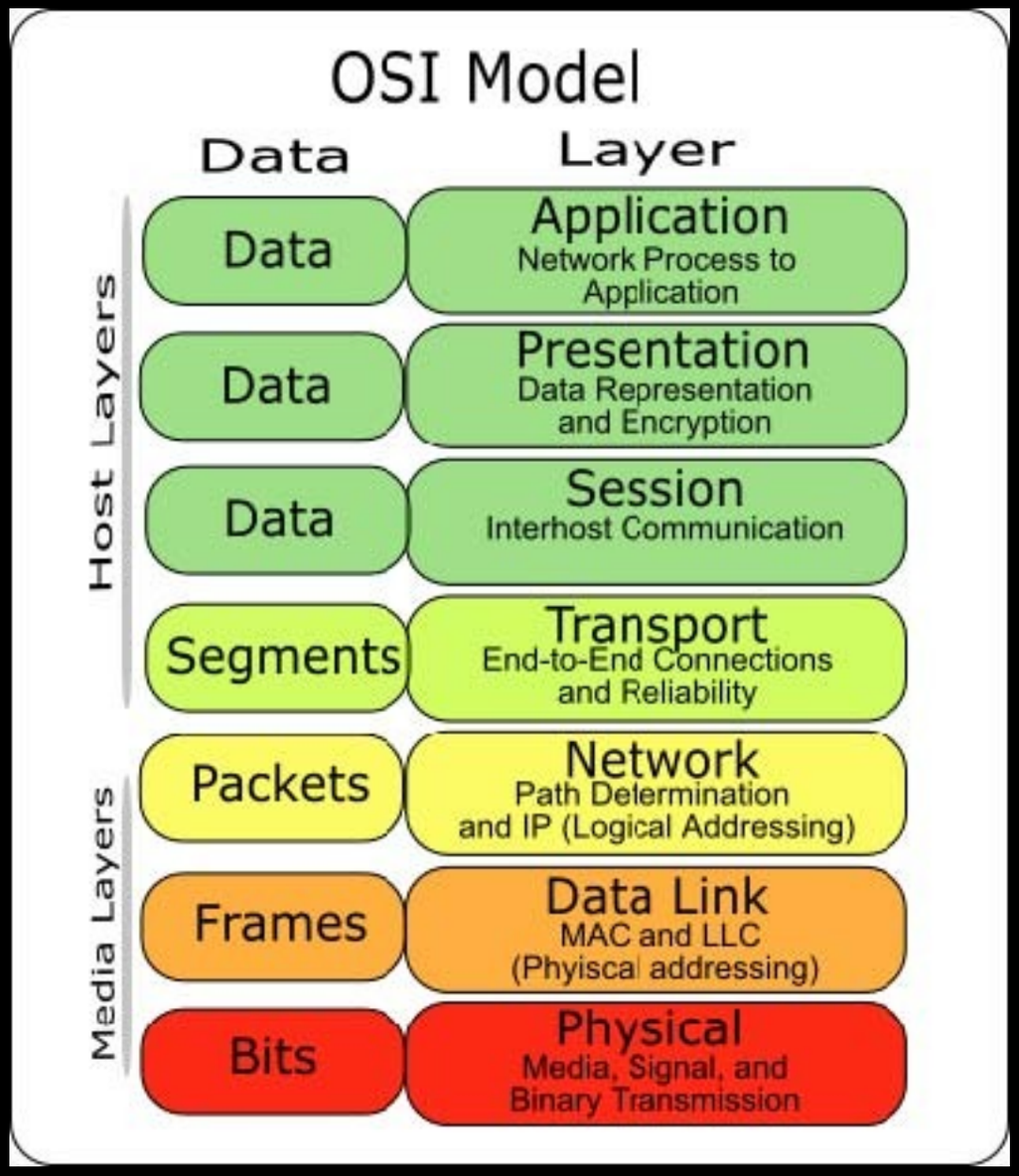
Conversation - Rules of Conduct

- Communication is holding a conversation
  - Interprocessor 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
  - They govern how we start a conversation, who speaks when, how fast, how often, etc.

Context of Communication

Open Systems Interconnection



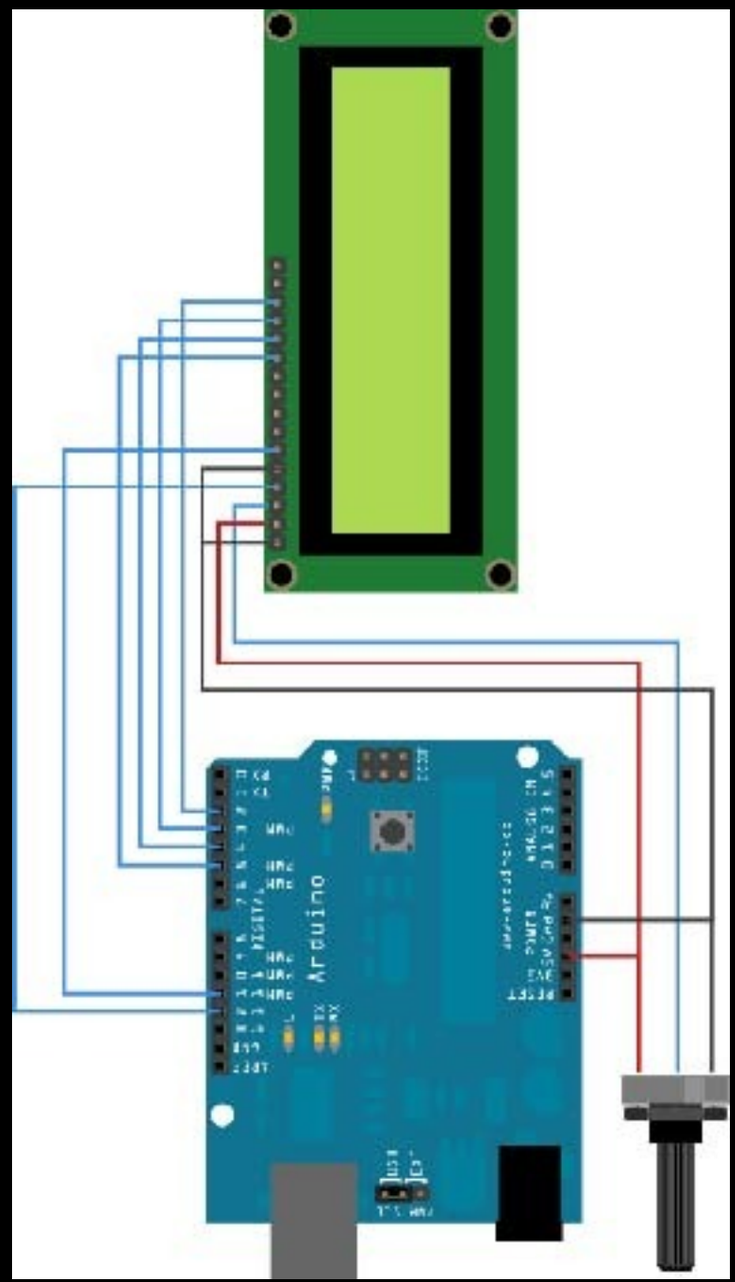
Serial, WiFi, Ethernet

Voltages, Wires, Timing

Types of Interface

Parallel

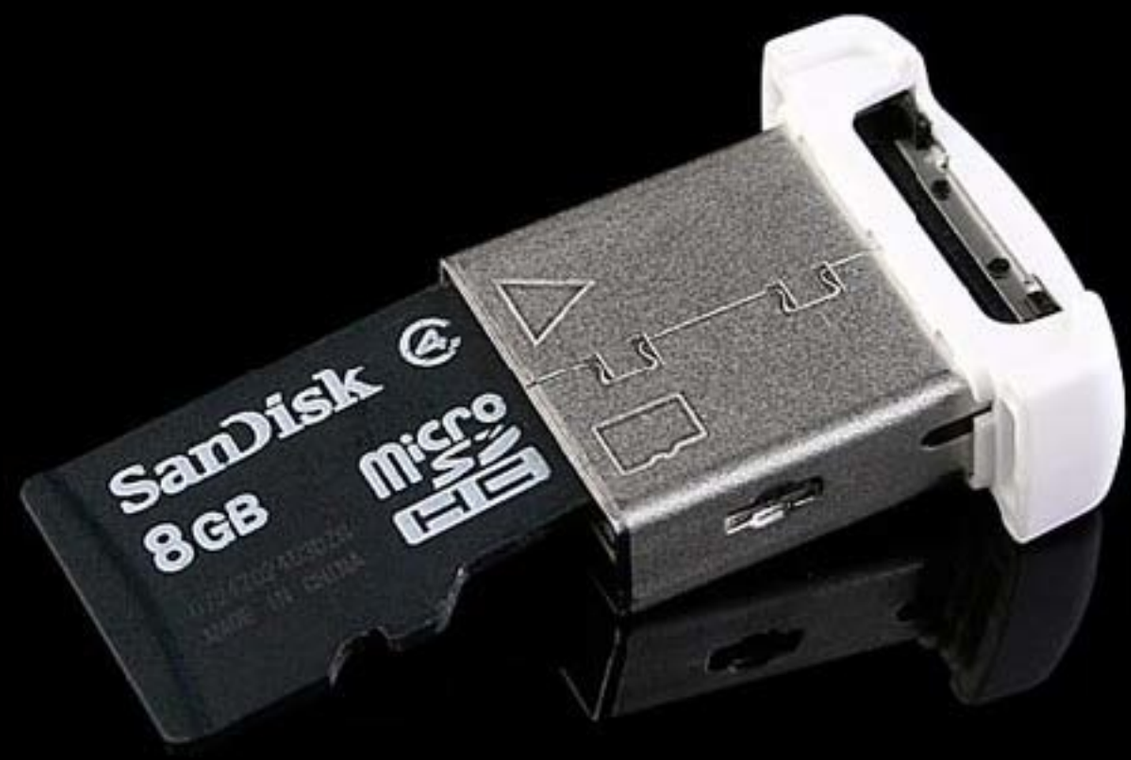
- Examples
  - Graphical LCD
  - SCSI, Firewire
- Advantages
  - Faster in Theory
- Drawbacks
  - Crosstalk
  - Clock Skew
  - Wire per Bit



Types of Interface

Serial

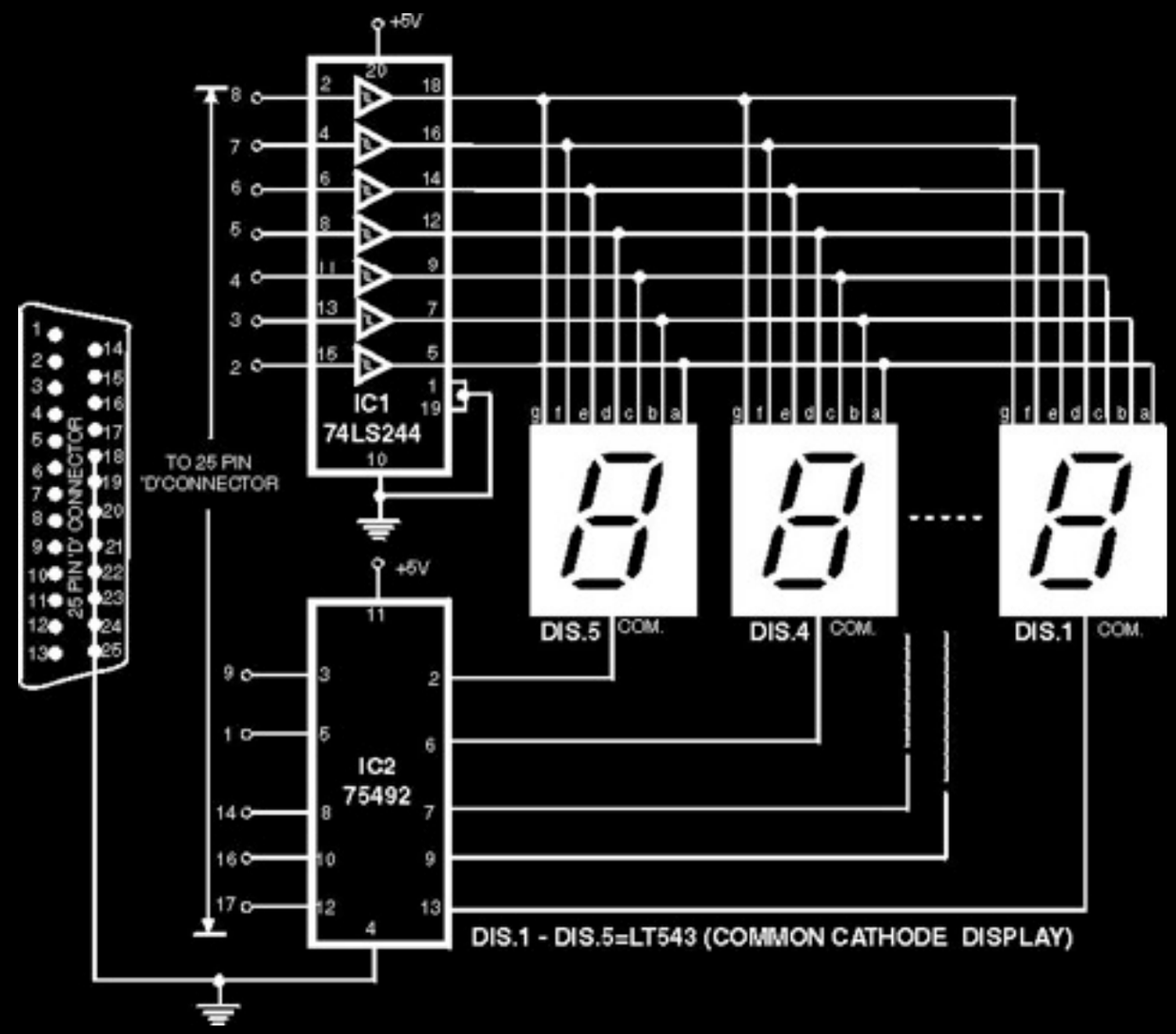
- Examples
  - USB
  - SPI and I²C
- Advantages
  - Clock Faster
  - Fewer Wires
- Drawbacks
  - Overhead of Negotiation



Conserve Resources

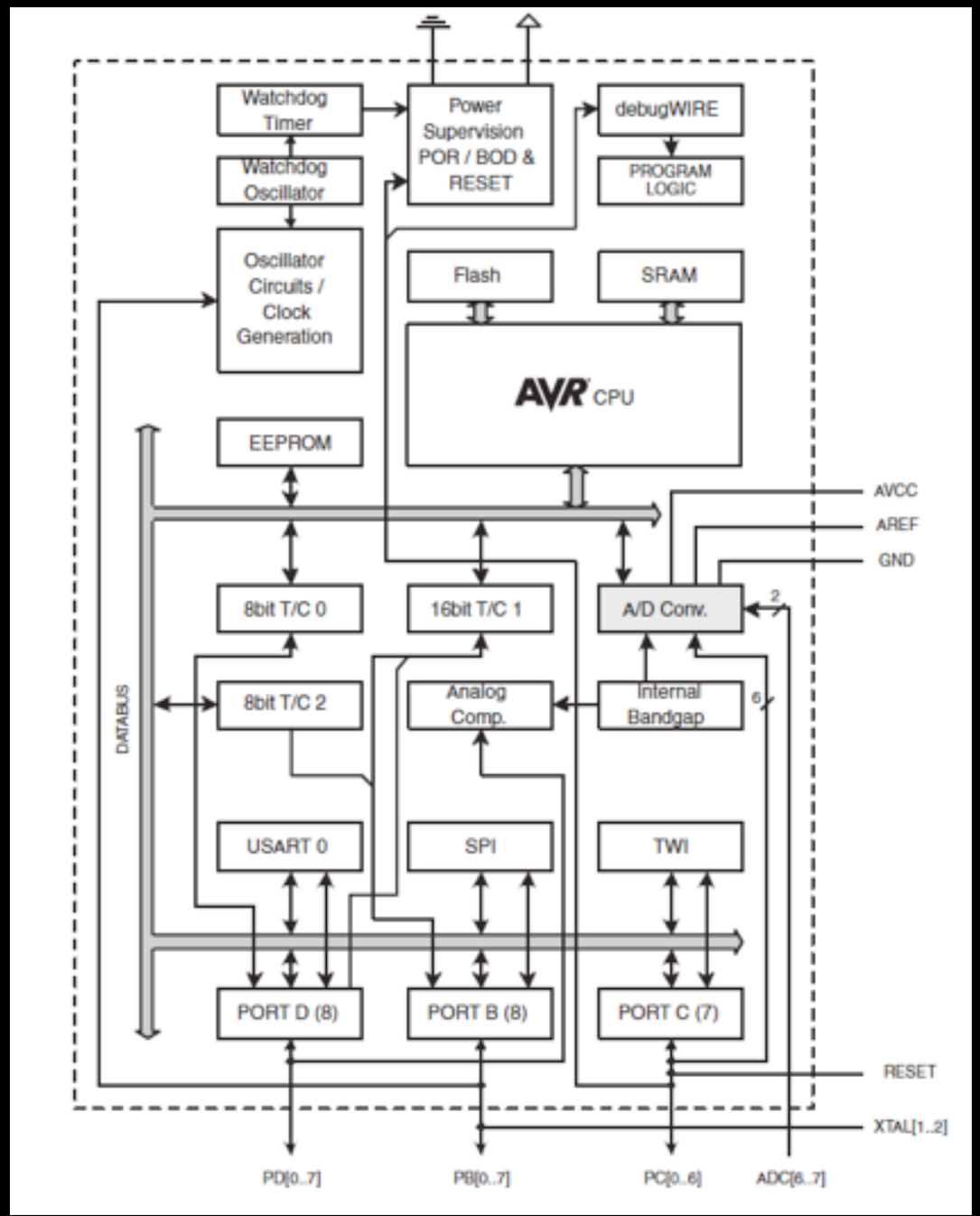
Ride the Bus

- Types of Interface
  - Both Serial and Parallel
  - SPI and I^2C
- Internal or External
- 1 line per device
  - Chip select



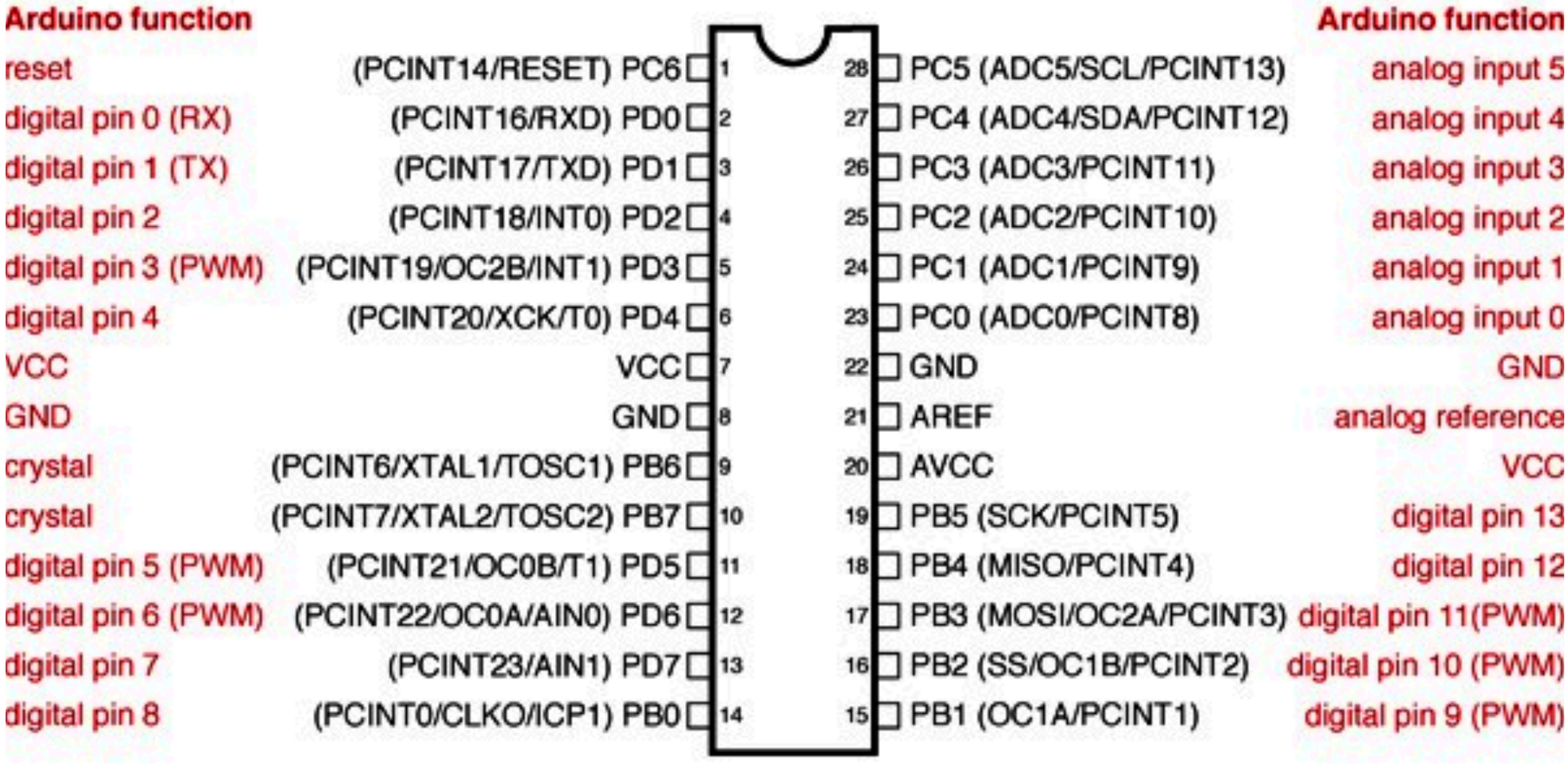
Microprocessor
Communication

- Atmega 328 supports:
  - Digital and Analog I/O
  - Master/Slave SPI interface
  - 2 wire serial interface bus (I2C)
  - Programmable Serial USART



Microprocessor
Communication

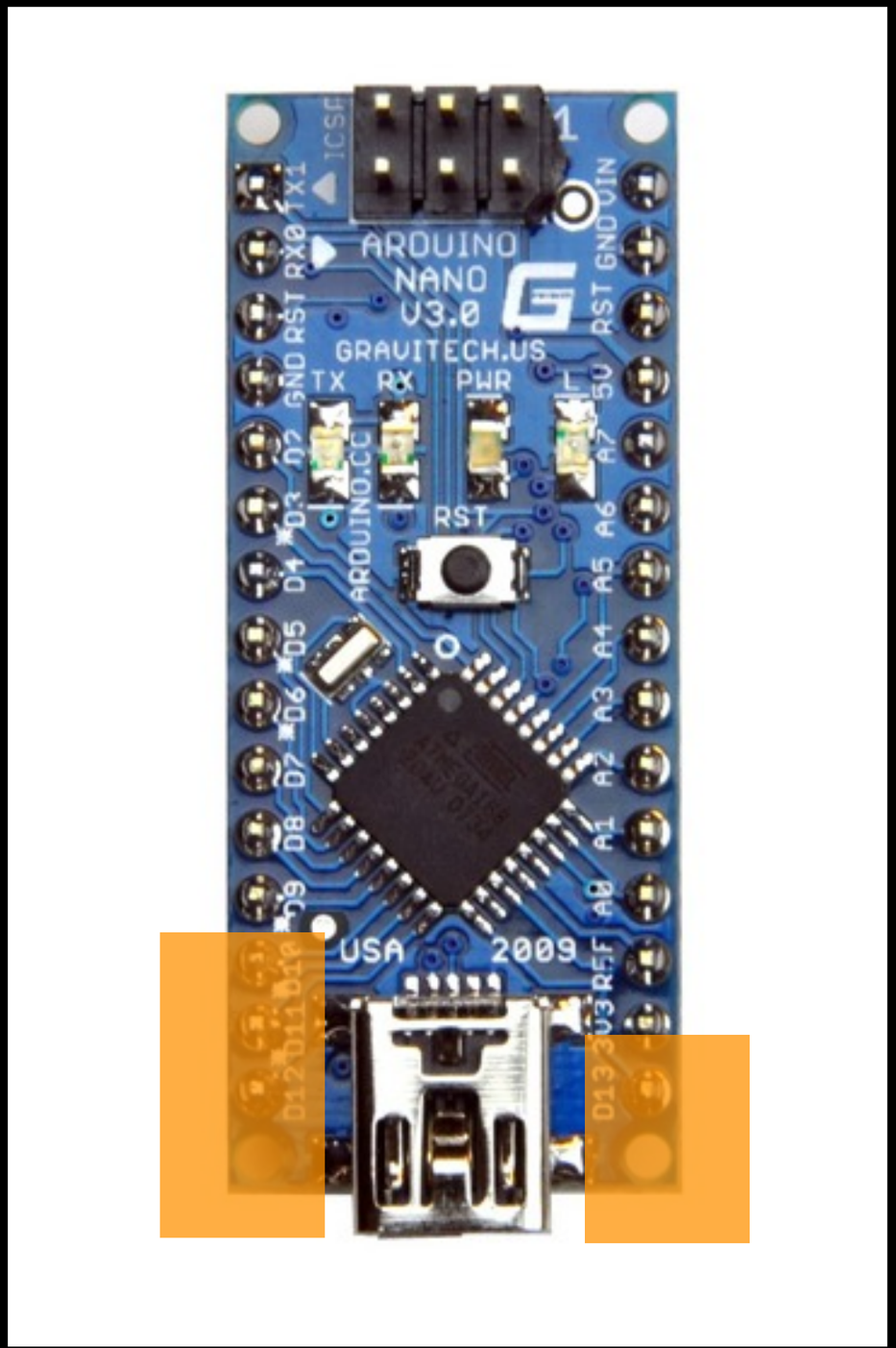
Atmega168 Pin Mapping



Digital Pins 11,12 & 13 are used by the ICSP header for MISO, MOSI, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

Microprocessor

Communication



SS
MOSI
MISO

SCK

How does a MCU communicate?

Bits & Bytes

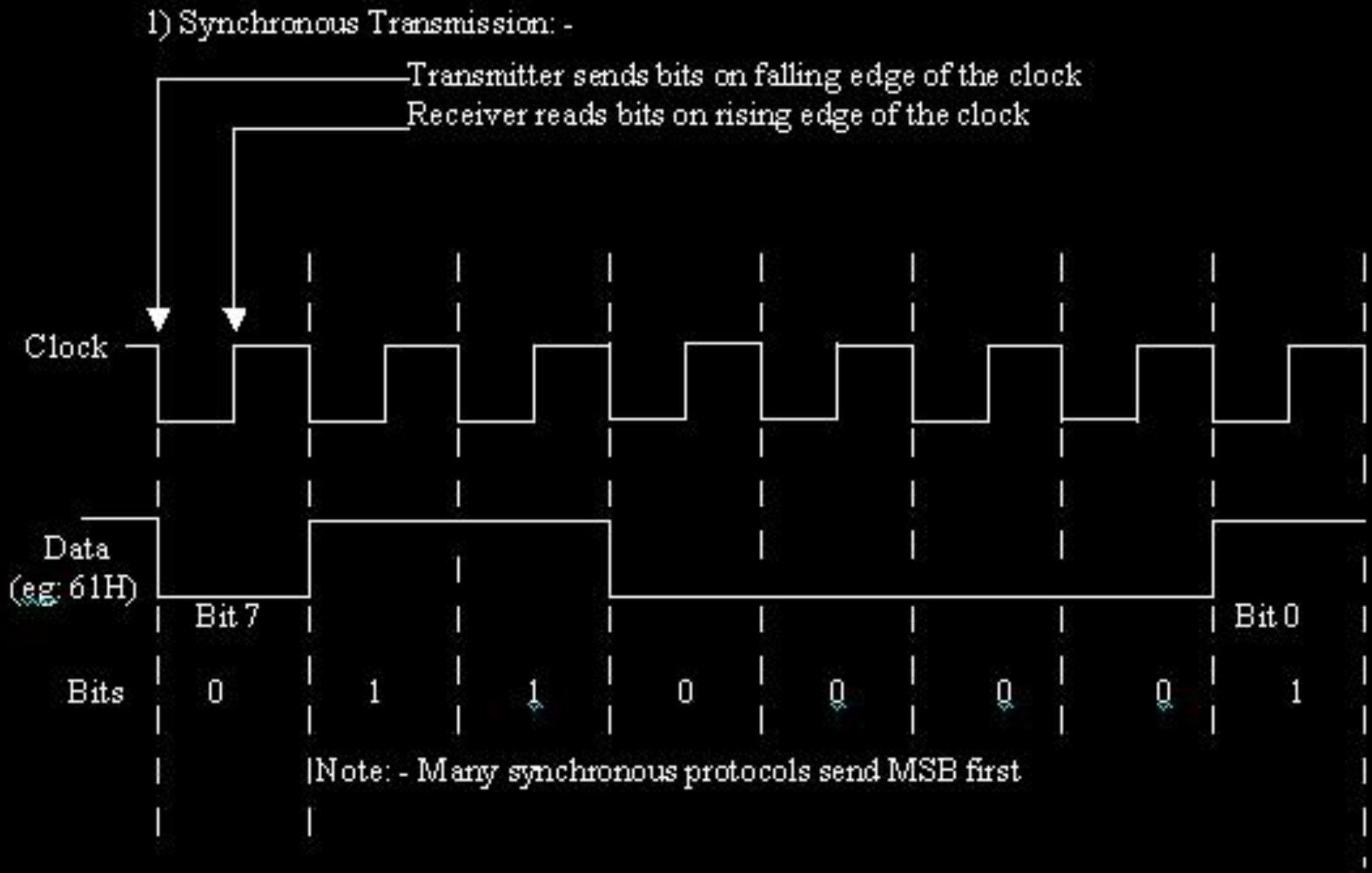
- 1 Bit
  - 0 = LOW
  - 1 = HIGH
- 2 Bits
  - 00 = 0
  - 01 = 1
  - 10 = 2
  - 11 = 3
- 3 Bits
  - 000 = 0
  - 001 = 1
  - 010 = 2
  - 011 = 3
  - 100 = 4
  - 101 = 5
  - 110 = 6

How does a MCU communicate?

Parallel vs. Serial

Serial Communication

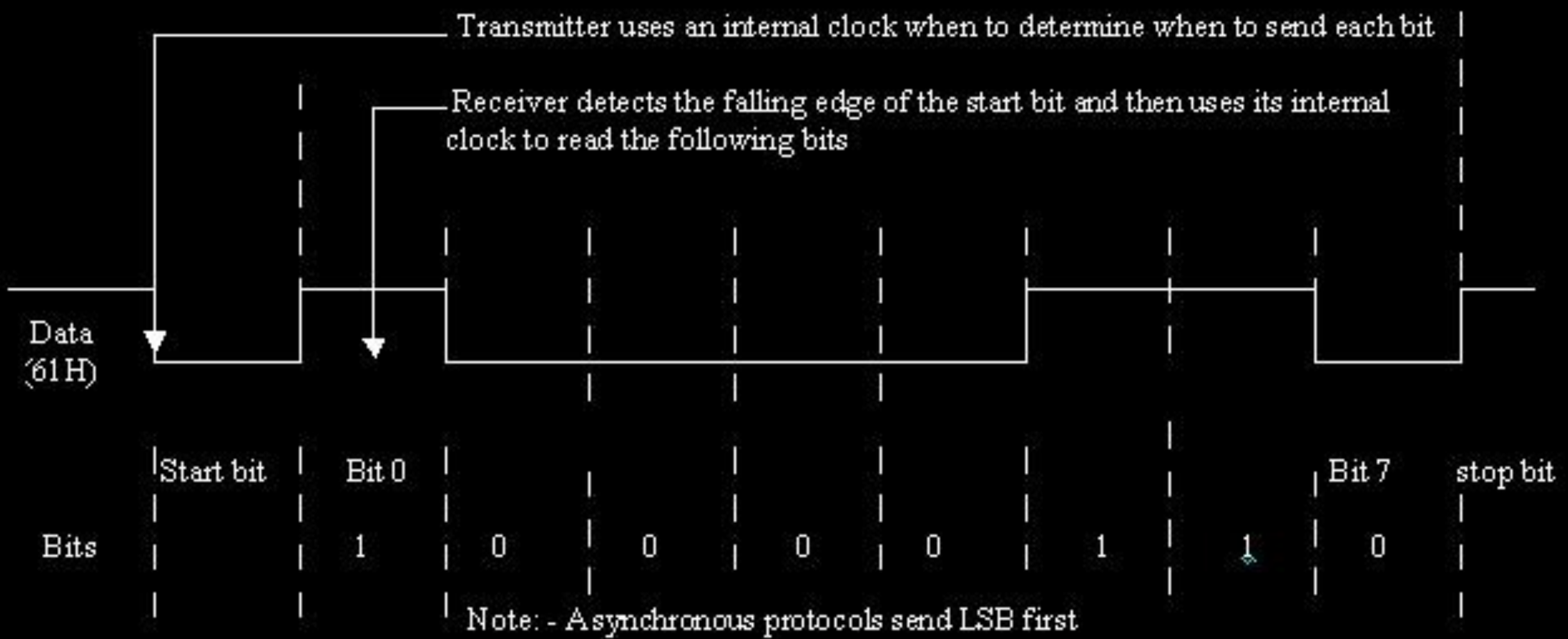
Synchronomous



Serial Communication

Asynchronomous

2) Asynchronous Transmission: -



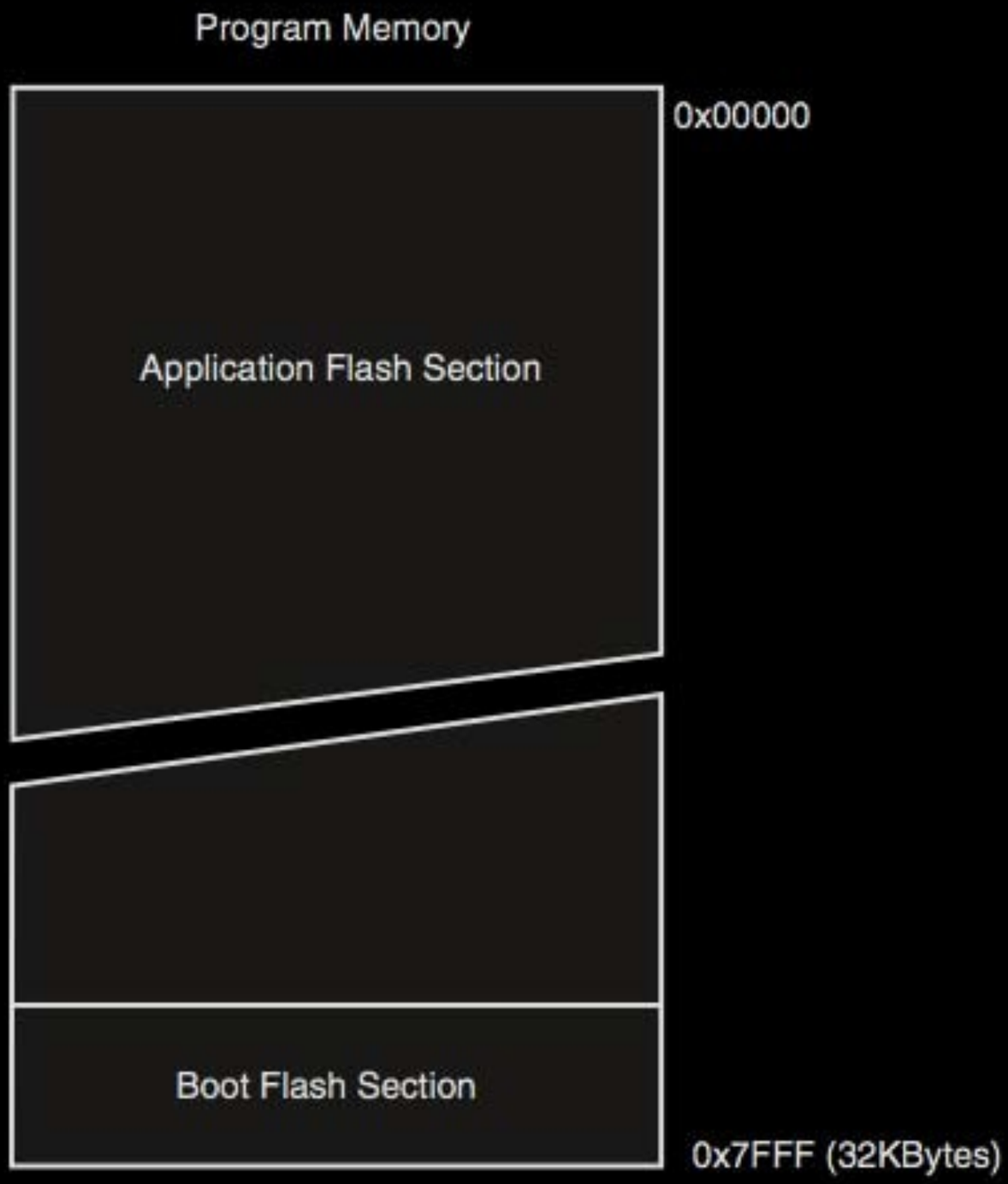
Serial Peripheral Interface

Configuration and Use

- How do we configure the microcontroller (uC) for SPI?
- How do we use SPI to communicate?

Atmega 328 Memory Map

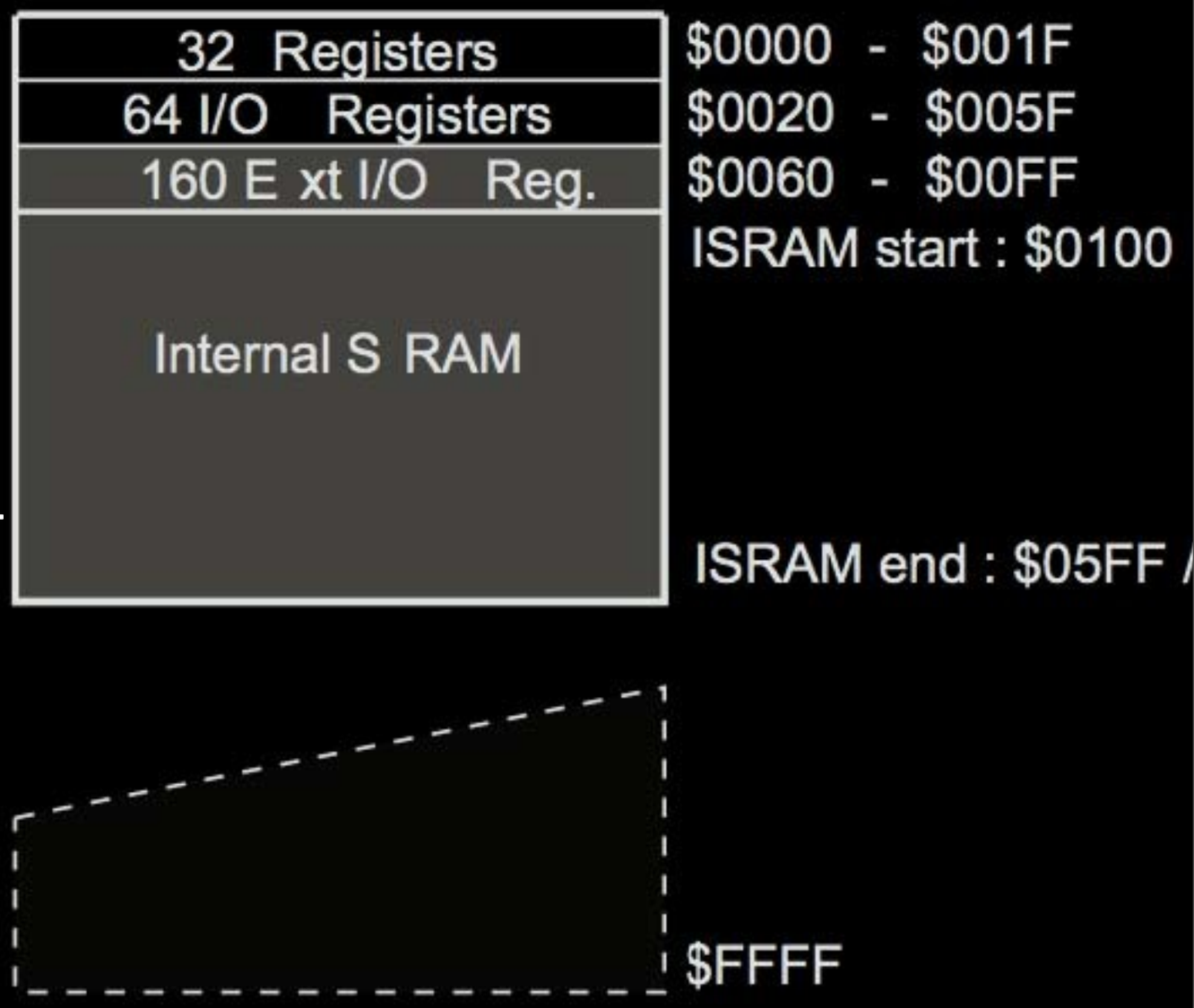
Program Memory



Atmega 328 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 μC operating settings.
- Writing values into these registers changes the settings.



Serial Peripheral Interface

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	0	0	0	0	0	0	0	

- 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?
- Set SPCR |= 0101 0000

- 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

Serial Peripheral Interface

Configuration

- SCLK, MOSI, & MISO are pre-assigned
- We only connect them