Introduction to Embedded Systems & Microcontroller Programming

Microcontroller Programming

Week 1

Vivatsathorn Thitasirivit

Rev. 1.0 (Feb 2023), complementary with COMP103 guidebook

https://vtneil.com

Go to course's website
https://vtneil.com/courses/pre_comp103/

Go to course's website
https://vtneil.com/courses/pre_comp103/

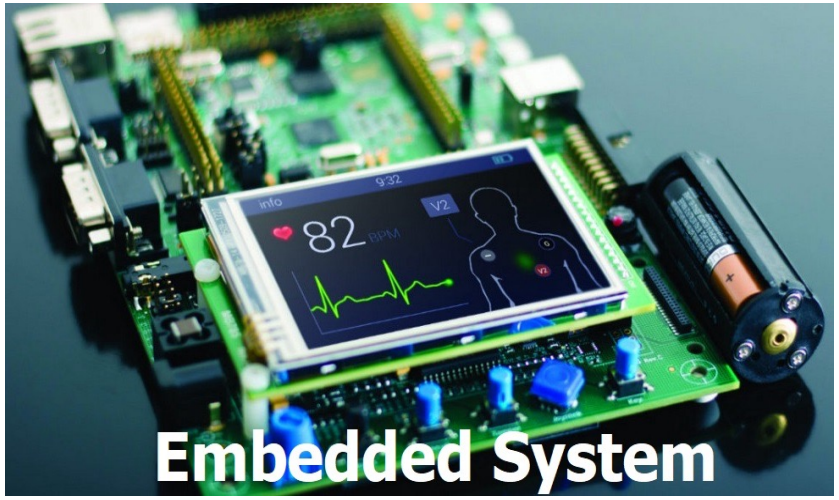
Embedded Systems

What are they? and why?

An **embedded system** can be thought of a hardware system with software embedded in it for it to function as designed.

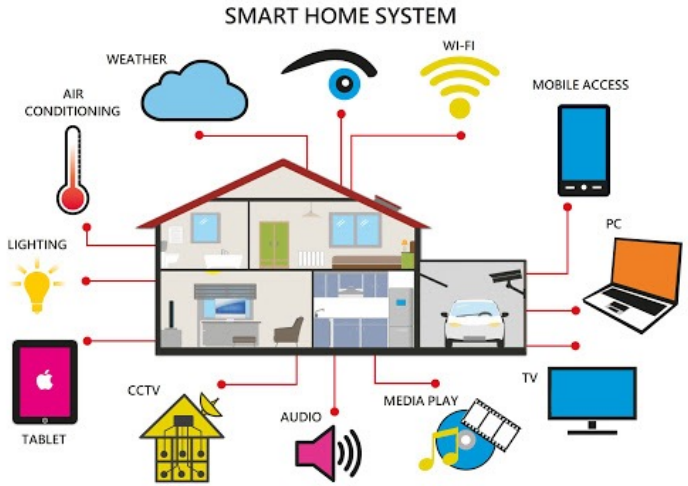
An embedded system can be either an independent system or a part of larger system.

It usually is built on **microcontrollers** or **microprocessors**, which is designed to perform specific tasks.



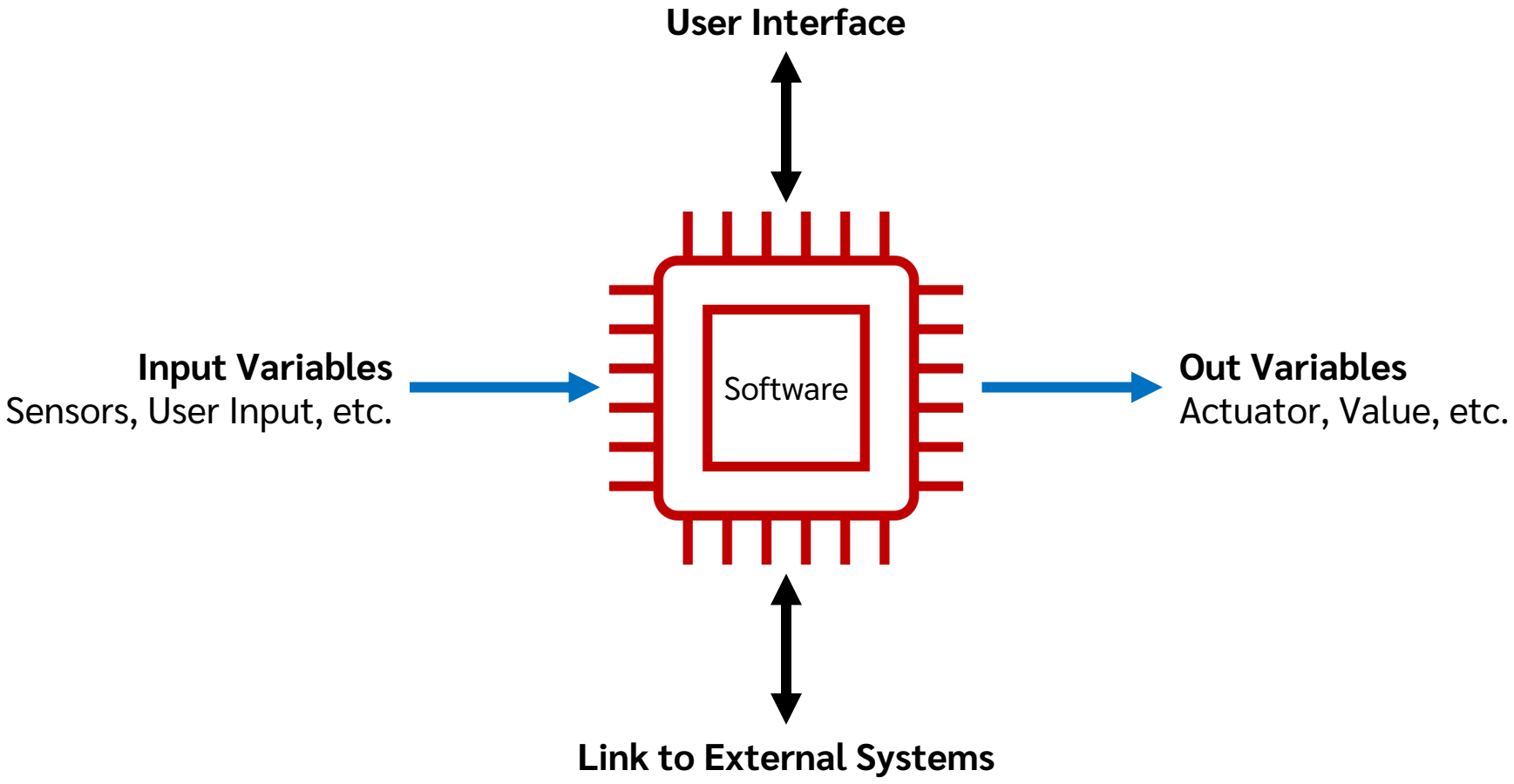
Embedded Systems

Applications in IoT



Embedded Systems

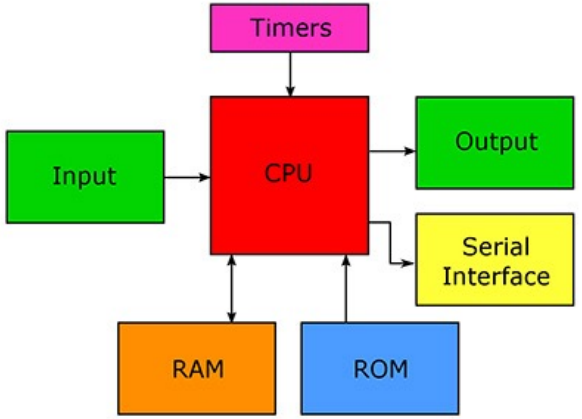
How do they work?



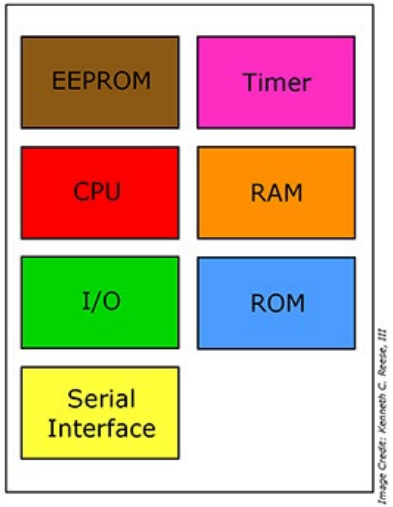
Embedded Systems

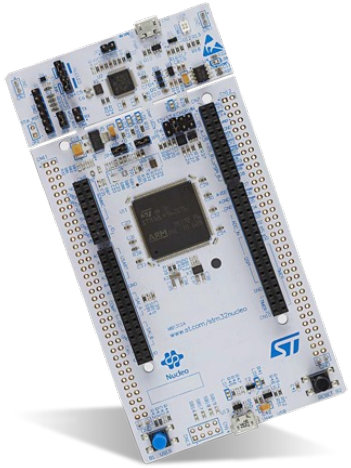
What are differences from our Computer?

Microprocesser: CPU
and several supporting chips.



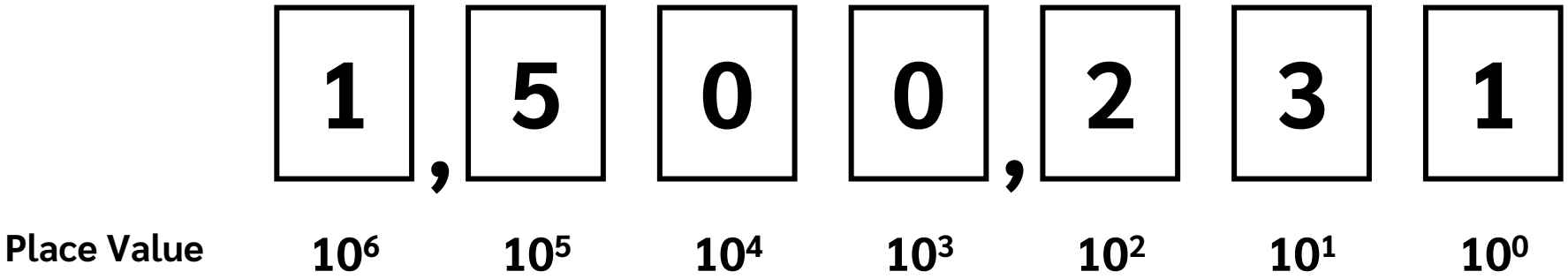
Microcontroller: CPU
on a single chip.





Embedded Systems

Human Counting: Number Representations



The value of 1,500,231 (in decimal system) is:

$(1 \times 10^6) + (5 \times 10^5) + (2 \times 10^2) + (3 \times 10^1) + (1 \times 10^0)$
$= 1{,}000{,}000 + 500{,}000 + 200 + 30 + 1$

Embedded Systems

Digial Computer: Number Representations

	0	1	0	0	0	1	1	0
Place Value	2^7 = 128	2^6 = 64	2^5 = 32	2^4 = 16	2^3 = 8	2^2 = 4	2^1 = 2	2^0 = 1

The value of 0100 0110 (in binary system) is:

$(1 \times 2^6) + (1 \times 2^2) + (1 \times 2^1)$
$= 64 + 4 + 2$
$=$ **70** (in decimal system)

Embedded Systems

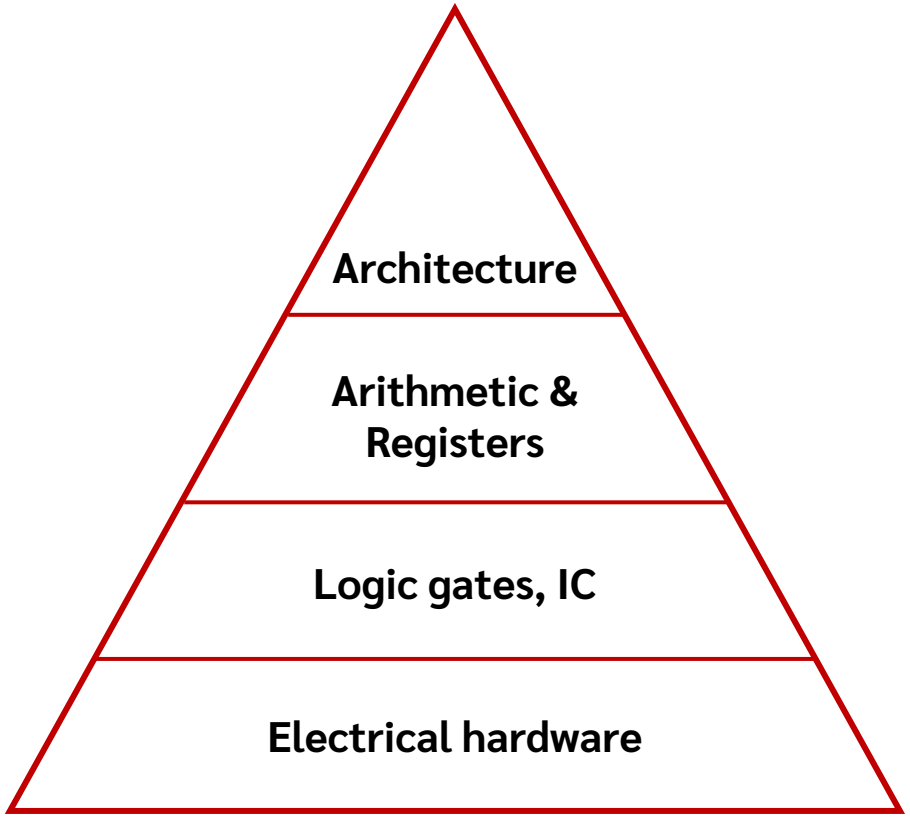
Computer Architecture

Layer 0 A computer is built up systematically from the lowest layer of physical (electrical) layer.

Layer 1 The logic gates are crafted from that layer.

Layer 2 The logical, arithmetic binary operations and registers are, then, developed from those logic gates and active components.

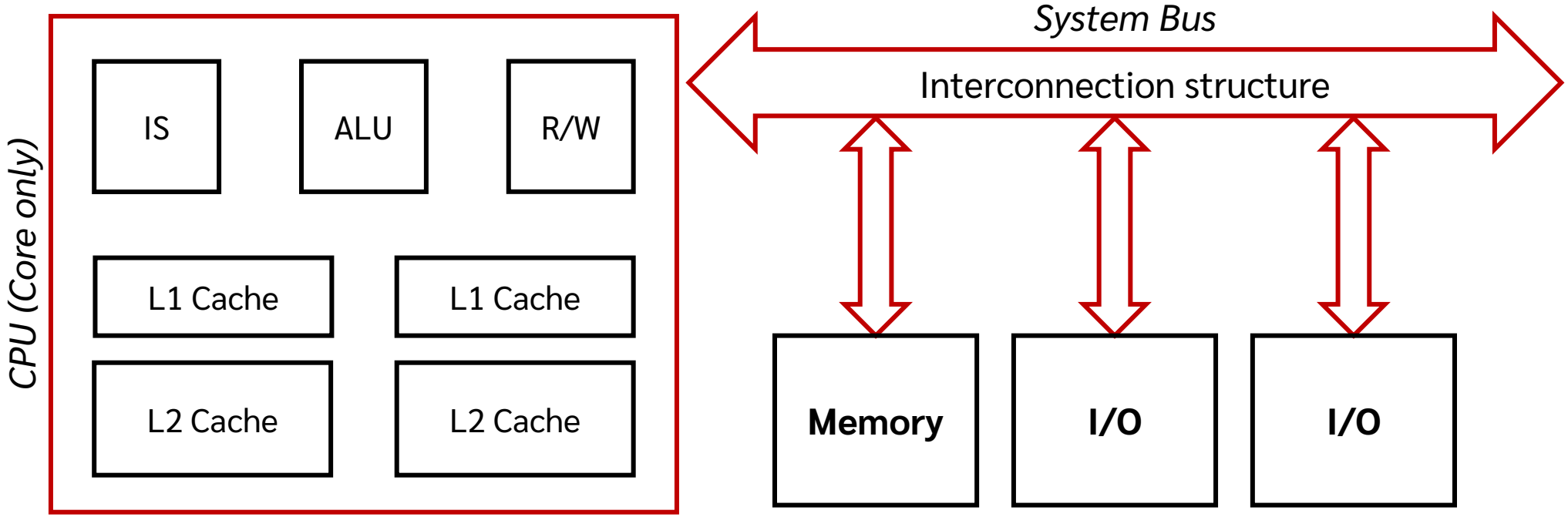
Layer 3 The processor level (architecture level) is the highest level in computer design hierachy. This layer consists of three basic types: CPU, memory, and I/O. These components are interconnected using a **bus**.



Embedded Systems

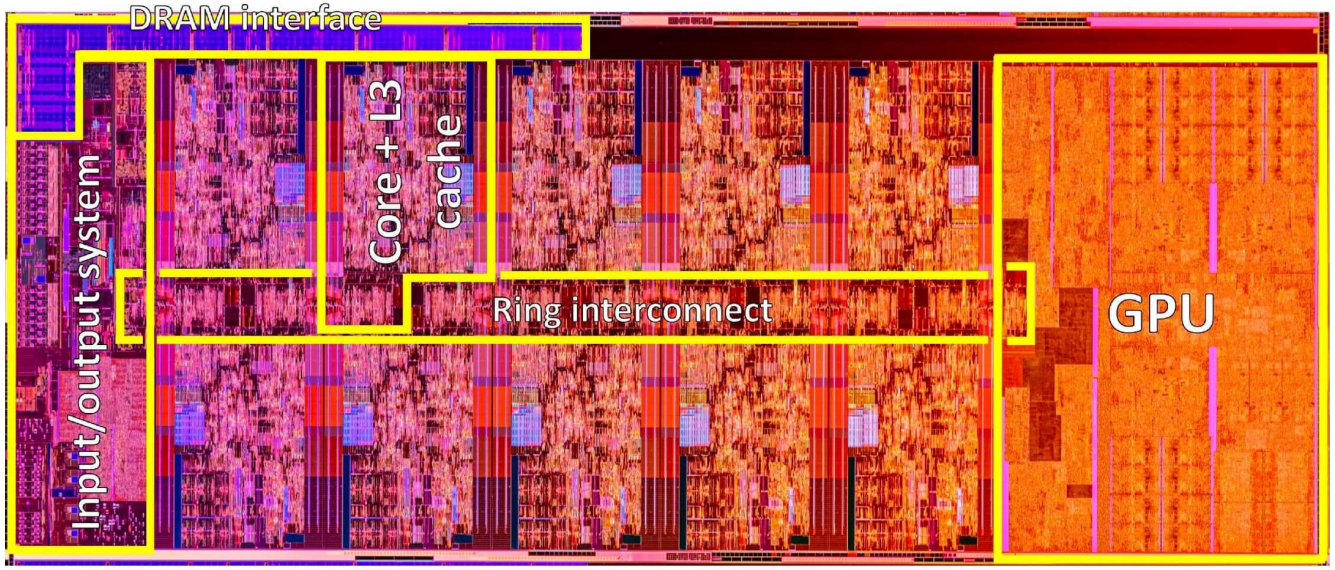
Computer Architecture

A **bus** is an interconnection system used in many computer architectures. It allows each individual part of the computer to communicate with each other.

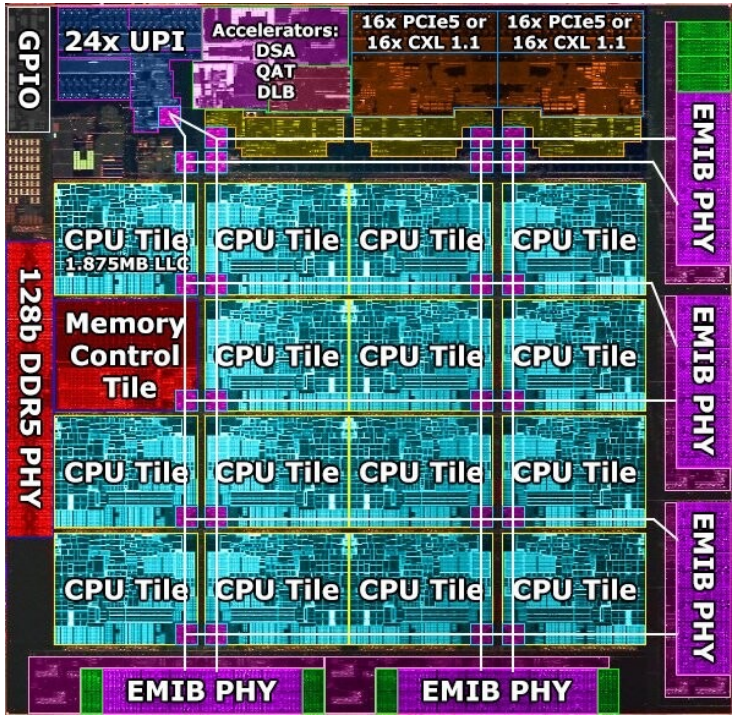


Embedded Systems

Computer Architecture



Intel i9-10900K Chip Die



Intel Xeon-4 Chip Die

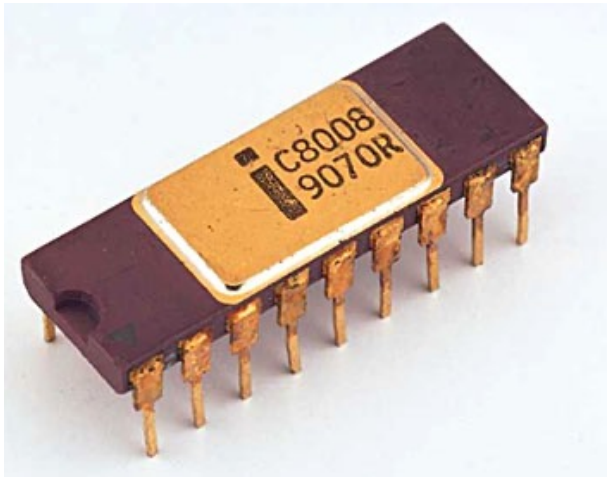
Embedded Systems

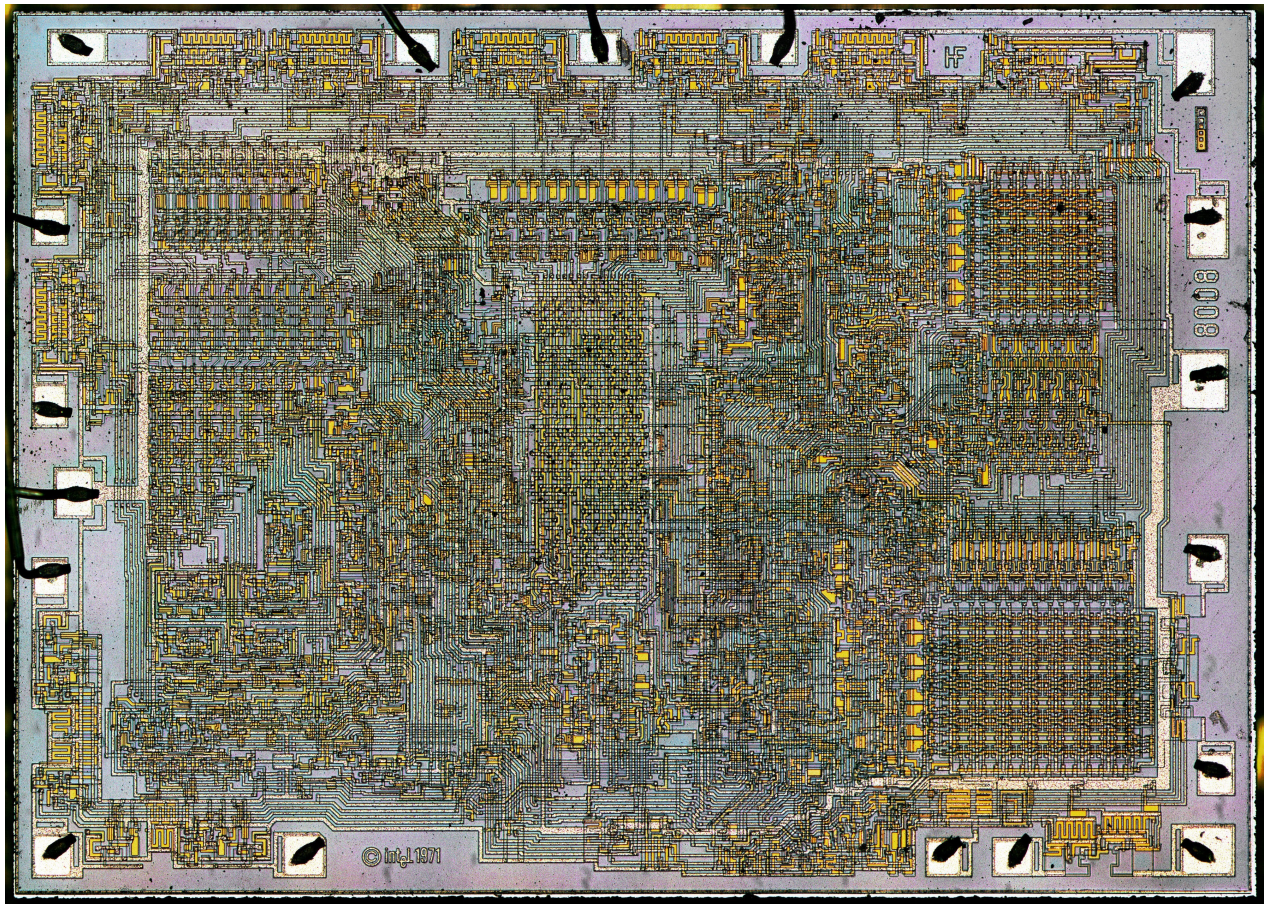
Computer Architecture

Intel Core Gen 12 (Alder Lake)

Embedded Systems

Computer Architecture





Intel 8008 (1972), *~50 years old*

Embedded Systems

Architecture of a Microcontroller

A microcontroller's **architecture** is quite different from our personal computers. It usually has specific and limited functionality.

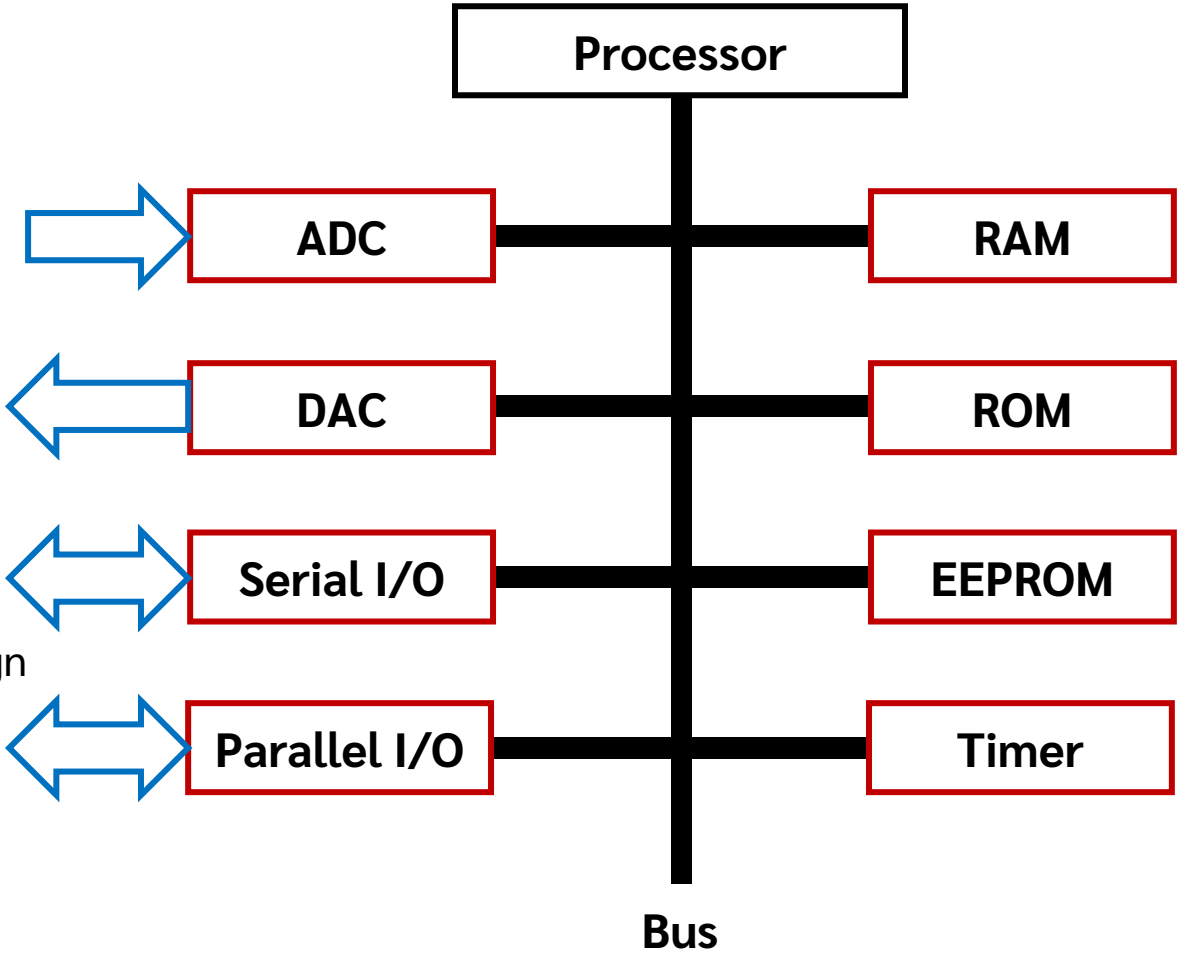
For internal processes, traditional set of hardware:

1. Processor/CPU
2. RAM
3. ROM, EEPROM
4. Timer (Clock/Oscillator)

are used for basic logical and arithmetic calculations.

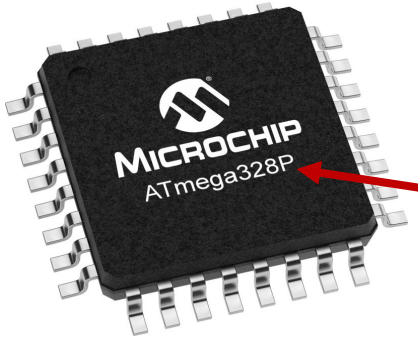
For external processes, which the microcontrollers are design for these types of usages, these components are used:

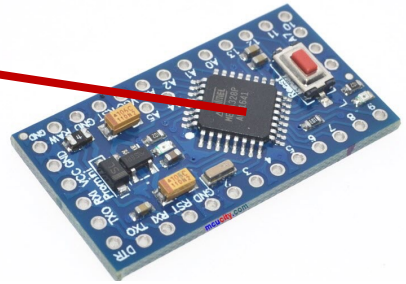
1. Analog-to-Digital Converter
2. Digital-to-Analog Converter
3. Serial Input/Output ports (Serial I/O)
4. Parallel Input/Output ports (Parallel I/O: shift-registers)



Microcontroller Form Factors

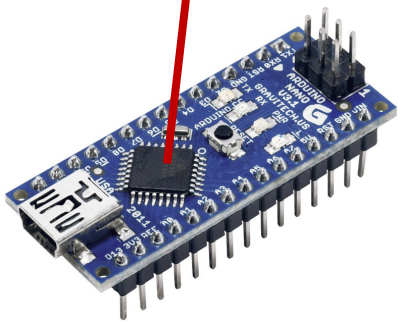
AVR ATMel ATmega328P Microcontroller Chip Form Factors



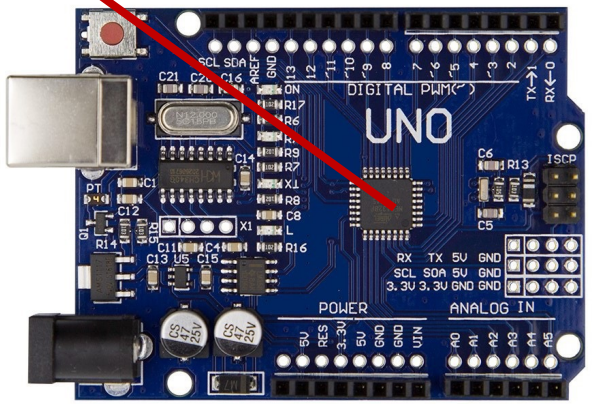


Arduino Pro Mini

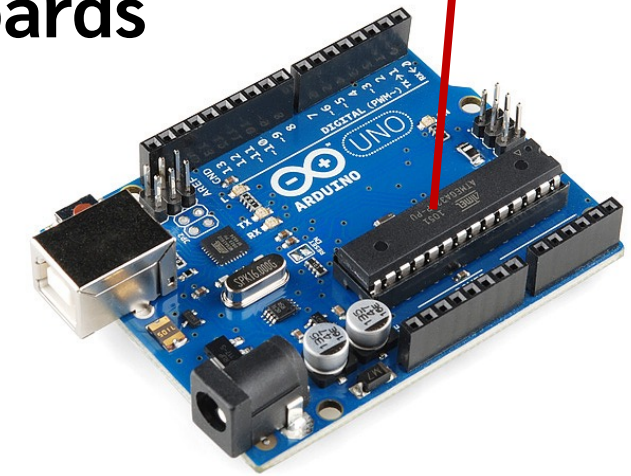
Development Boards



Arduino Nano



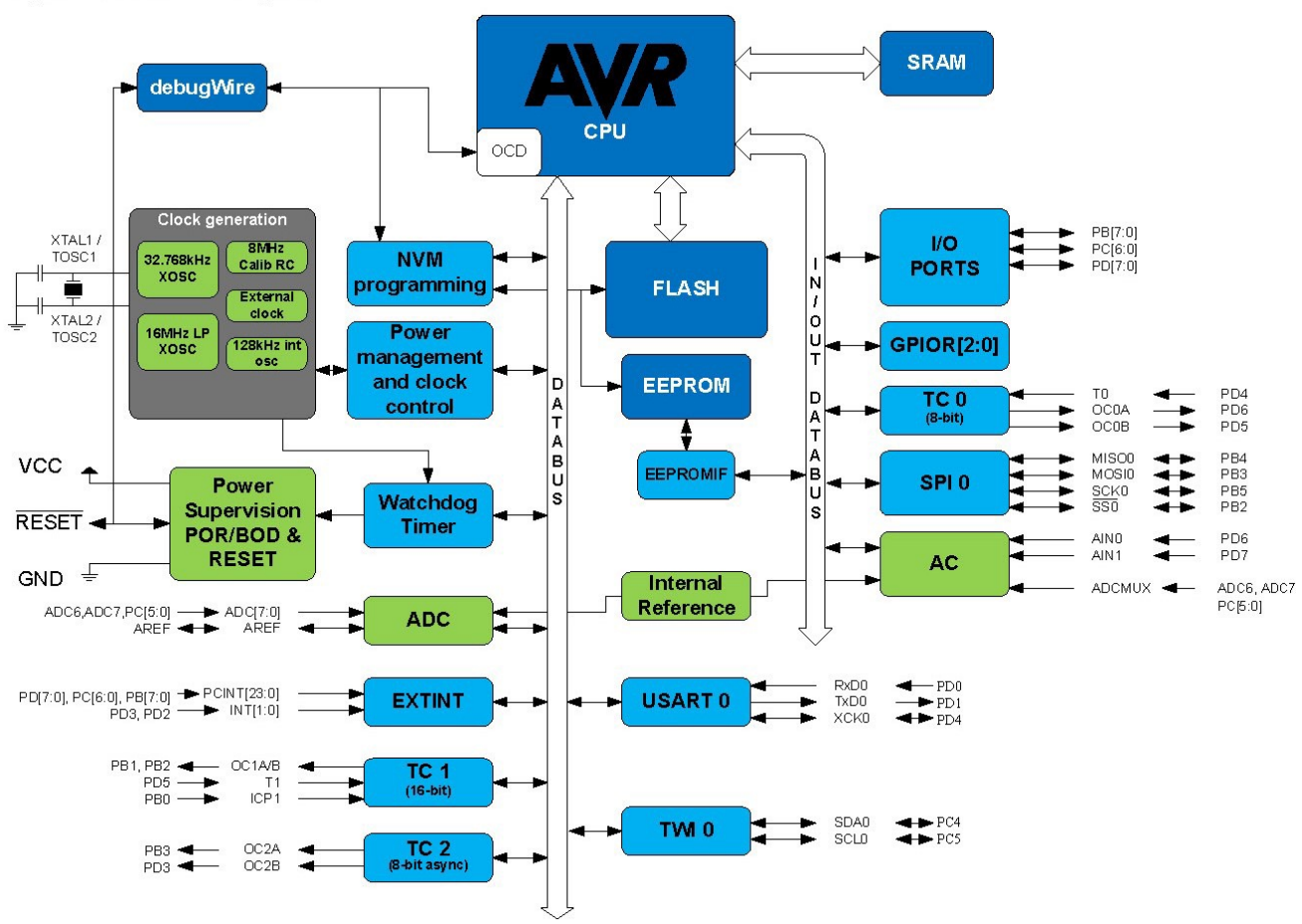
Arduino UNO Rev3



Integrated Circuits

Integrated Circuits (IC) Package Types *(Source: components101)*

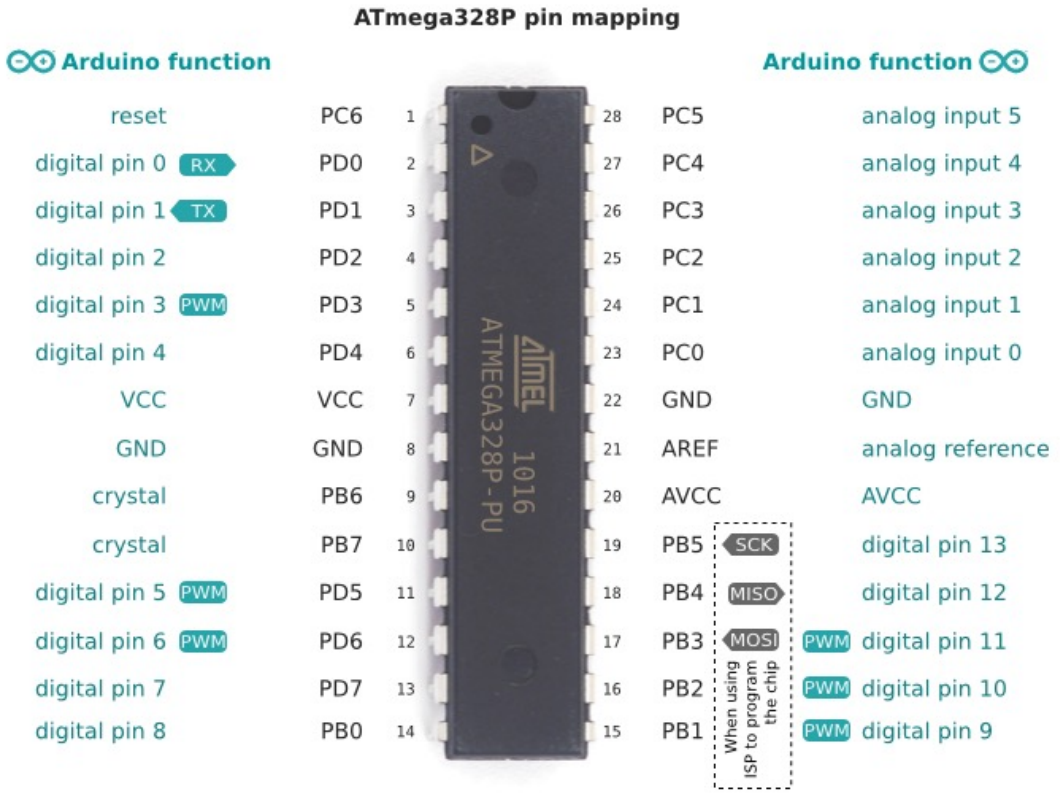
ATmega328P's Architecture



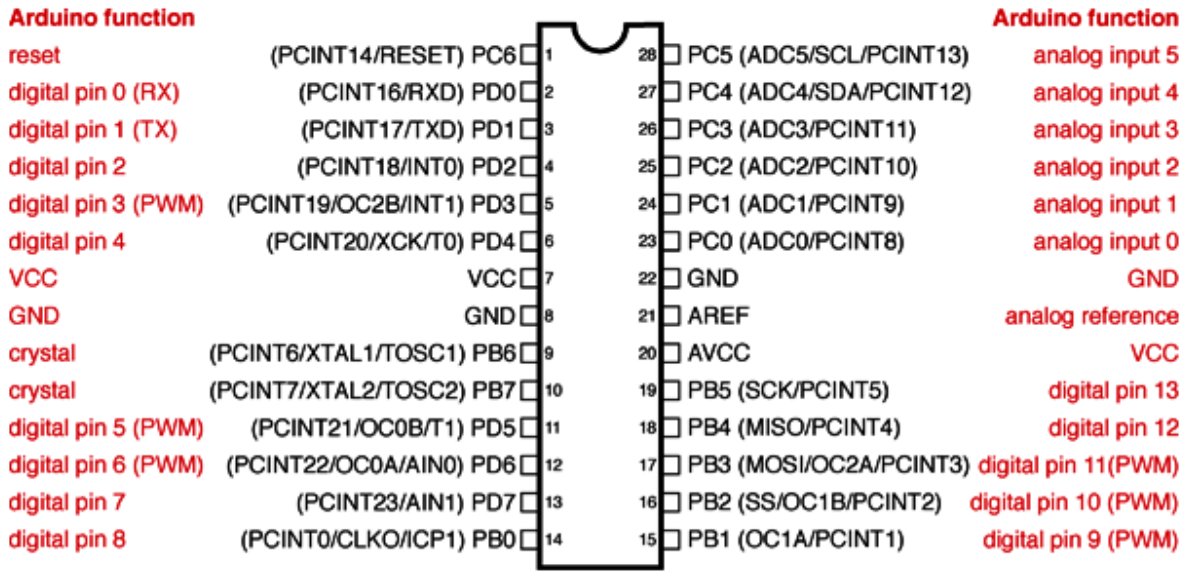
ATmega328P Block Diagram

Embedded Systems

ATmega328P's Pin Mapping



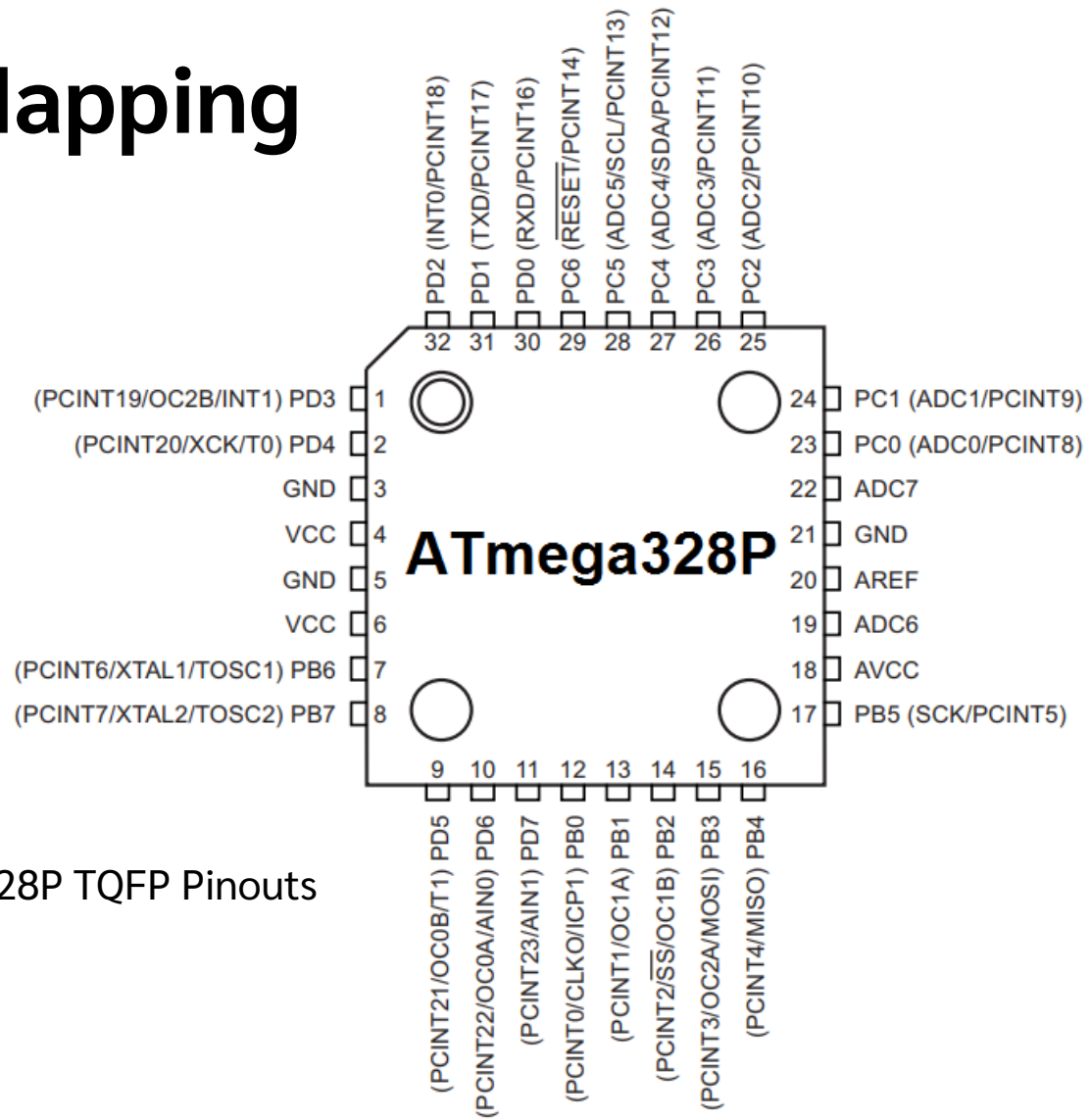
ATmega328P Through-hole Pinouts



ATmega328p Through-hole Pinouts (Detailed)

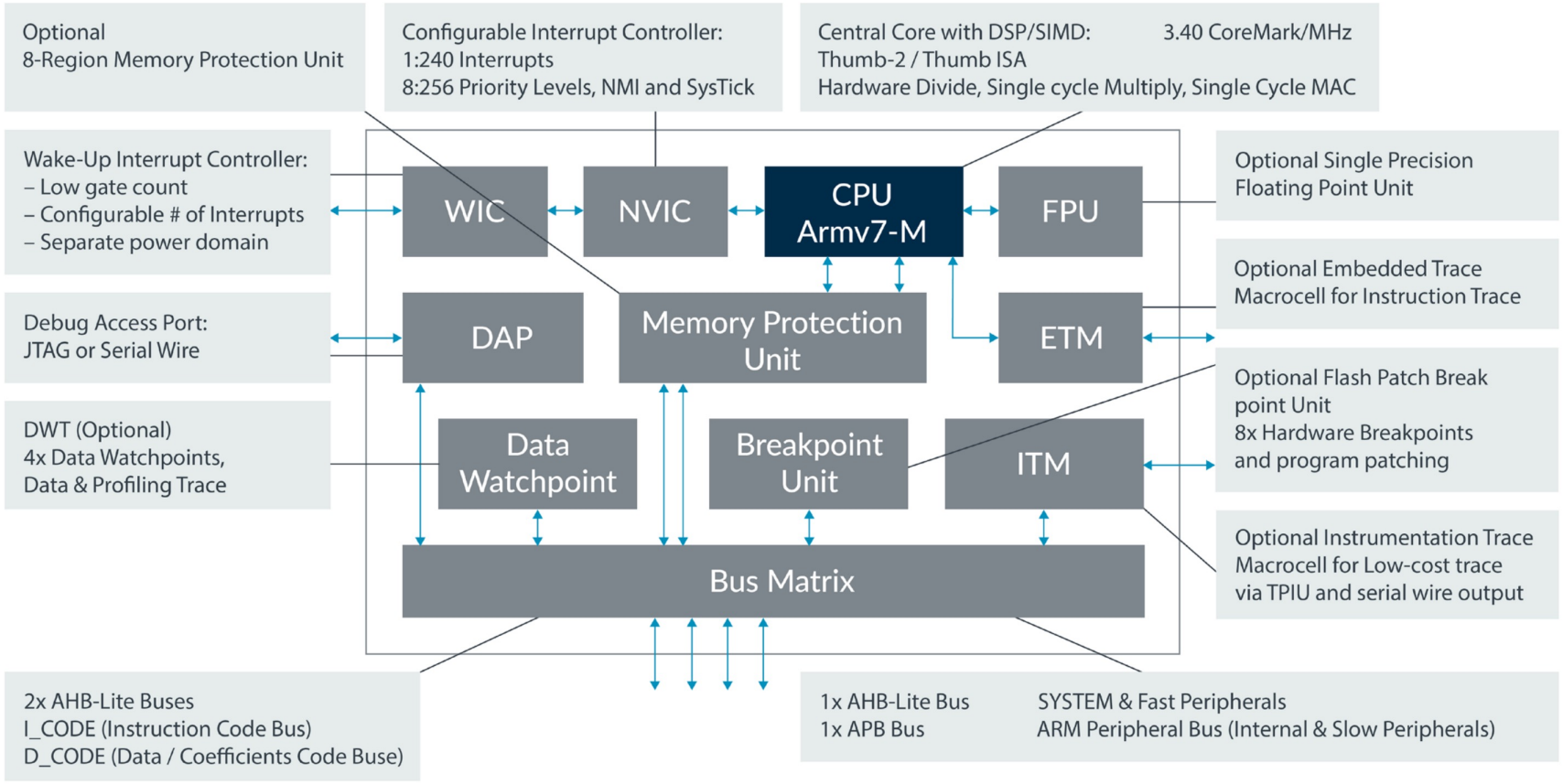
Embedded Systems

ATmega328P's Pin Mapping



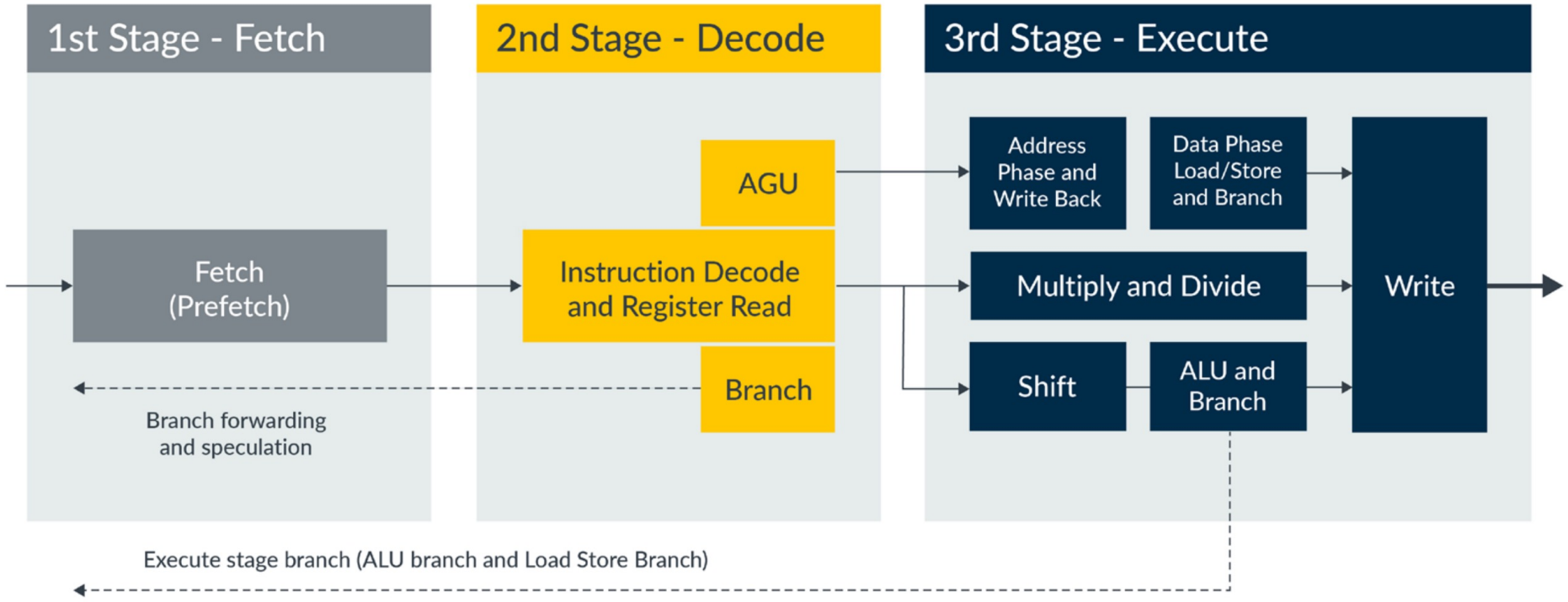
ATmega328P TQFP Pinouts

Architecture of a Microcontroller



ARM Cortex-M4 Instruction Pipeline (Source: ARM datasheet)

Architecture of a Microcontroller



ARM Cortex-M4 Instruction Pipeline (Source: ARM datasheet)

Hardware Communication

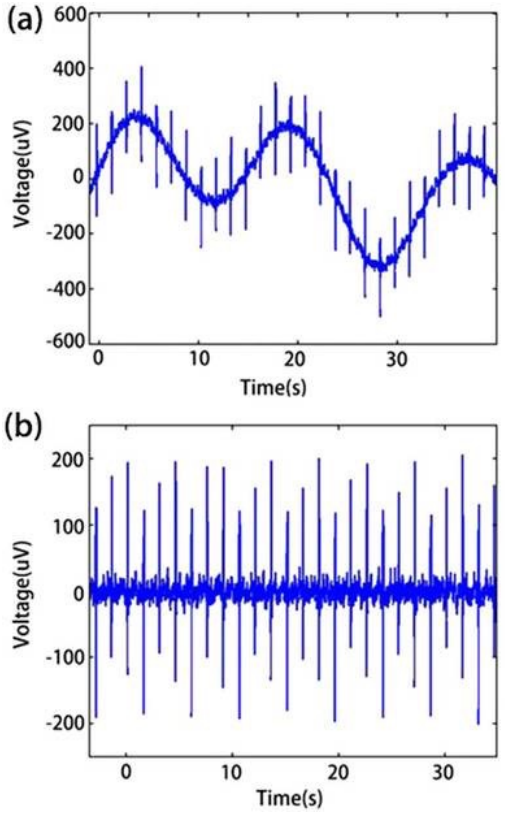
What is Signal?

Signal is anything that can carry information from one place to another.

In electronics, signal refers to any time-varying voltage, current, or electromagnetic wave that is **meaningful** and carries **information**.

Types of signals by quantification:

1. Analog Signal
2. Digital Signal



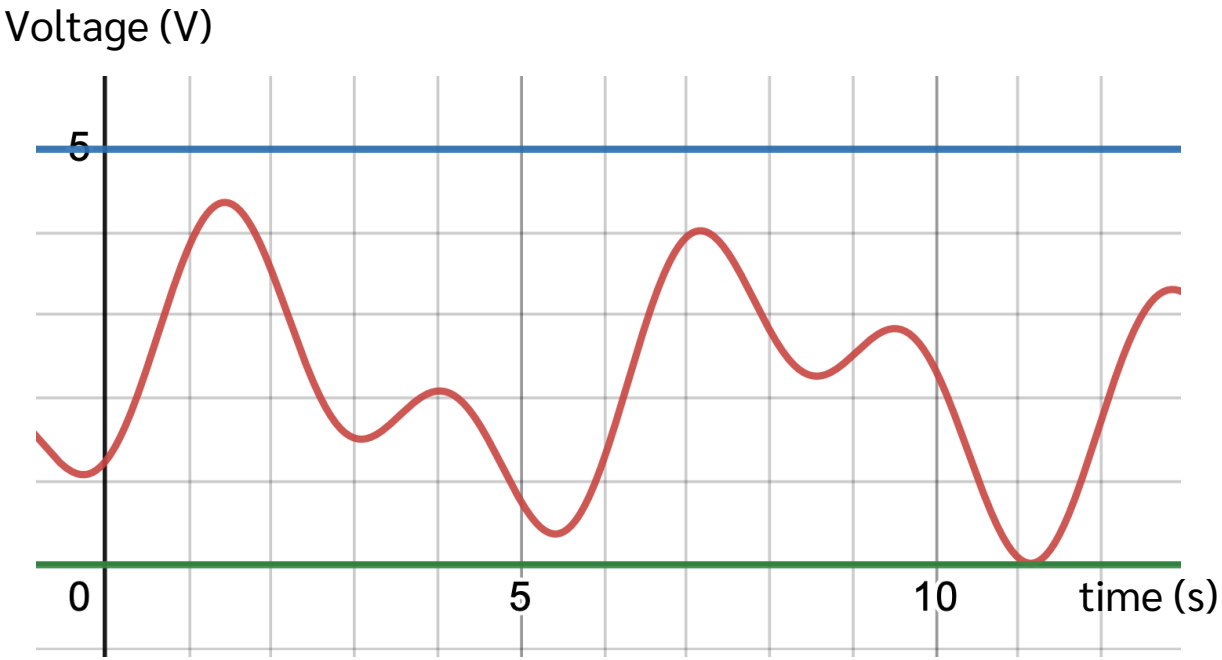
(c)

Hardware Communication

Analog Signal

Analog signal is a quantity by itself. The amplitude of the signal represents quantity. For example, an analog modulator has a range of voltage output of 0.00 V to 5.00 V.

The computer does not implicitly understand what analog signals are as the computers are designed to work in digital format. The hardware responsible for signal conversion is called "**Analog-to-Digital Converter**" (ADC).



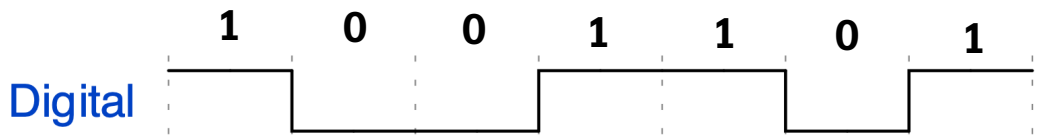
Hardware Communication

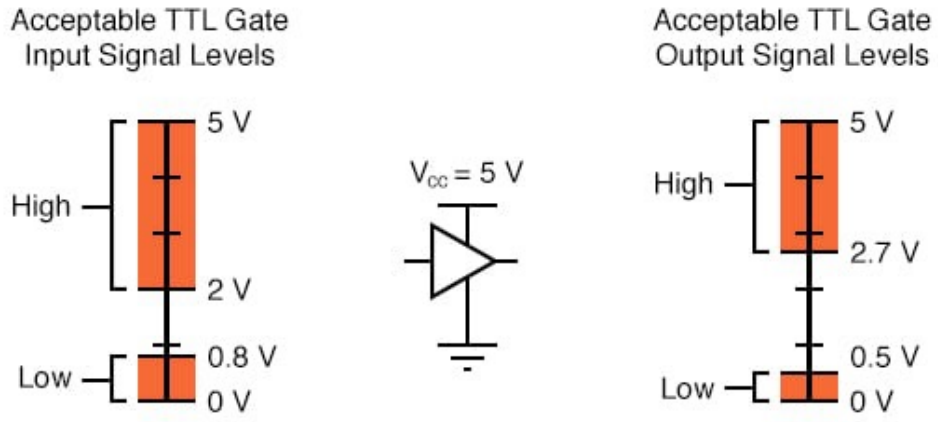
Digital Signal

Digital Signal is a more sophisticated way of representing data. The quantification is based on binary system. The voltages have threshold ranges.

For example, for input gate,
a range of *2 – 5 V means HIGH (“1”)*
and a range of *0.0 – 0.8 V means LOW (“ 0 “).*

Binary system is just a way of representing number. As humans, we use base-10 (digits 0 – 9). But in computer, electrical signal is identified either HIGH or LOW, so the base-2 (digits 0 – 1) is used.



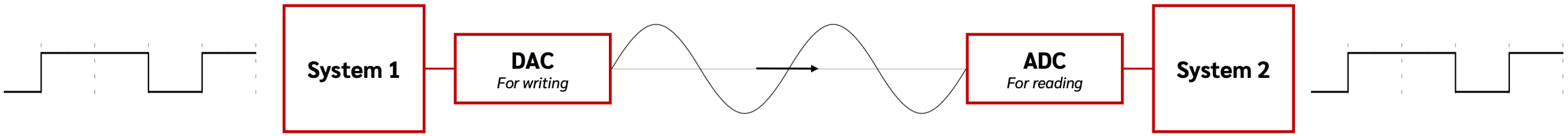


Hardware Communication

Analog-Digital Conversion

The hardware responsible for signal conversion is called **“Analog-to-Digital Converter”** (ADC).

For the inverse: The **Digital-to-Analog Converter** (DAC) is used.

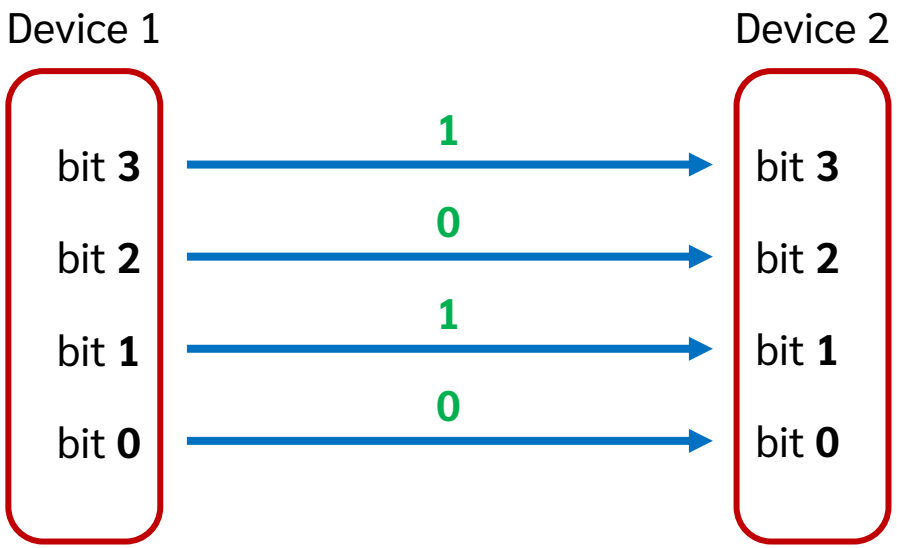


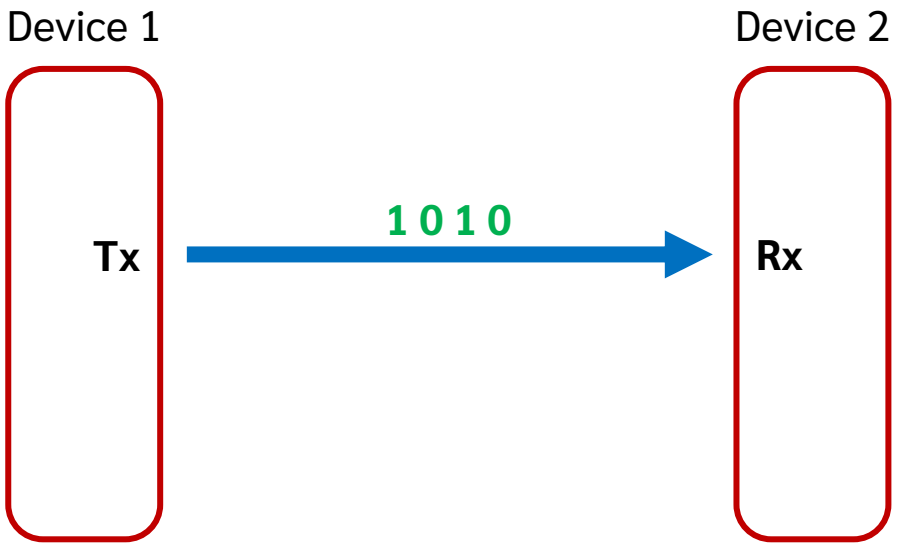
Microcontroller Connectivity

Wired Communication: Serial vs Parallel

Parallel Communication uses *n* number of wires to transmit *n*-bit data per packet.

On the other hand**, Serial Communication** uses at least 1 wire to transmit any bits of data as long as you have allocated enough time for it to send.





The binary data we need to send is "1010."

Microcontroller Connectivity

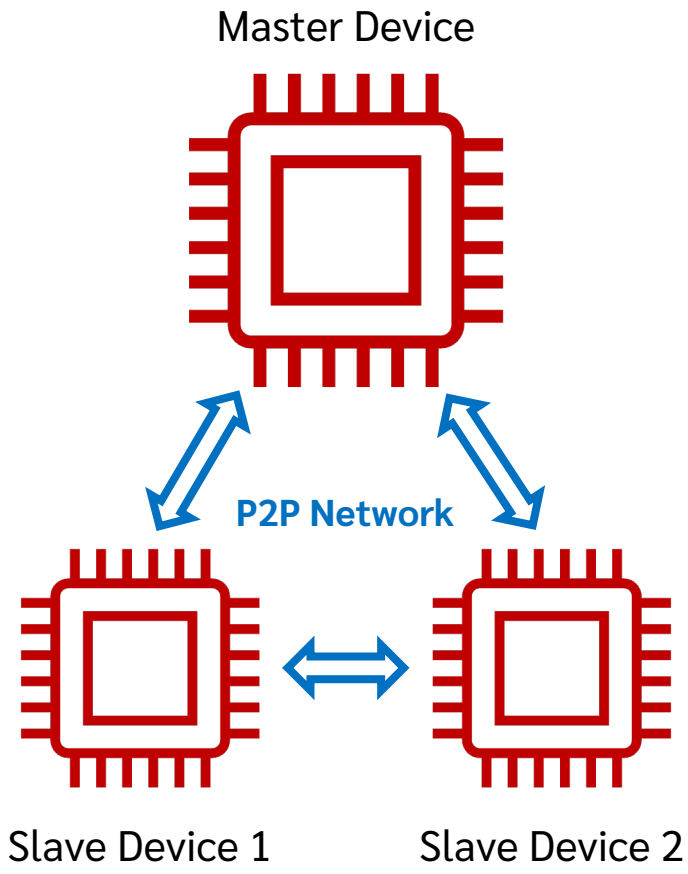
Wired Communication Protocols

A **communication protocol** is used when you want to communicate between two or more devices.

The protocols are like languages. If you use different languages in the network, everyone wouldn't understand what others are saying.

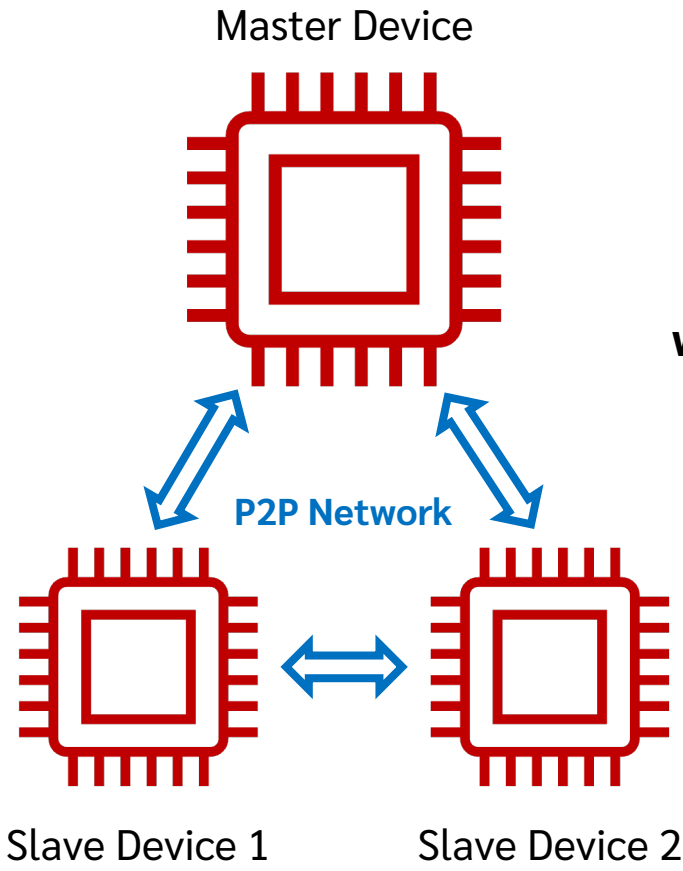
The computer network also works in the same fashion.

In this network example, every devices must communicate with each other. How would you design the network?

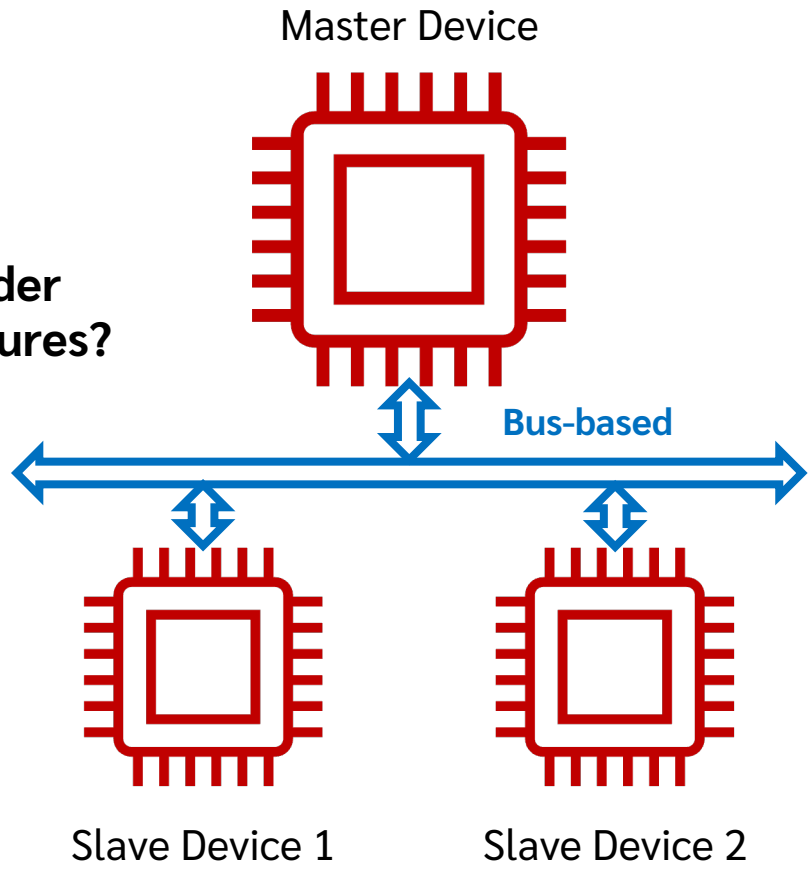


Microcontroller Connectivity

Wired Communication Protocols



What should you consider when designing architectures?



Microcontroller Connectivity: Wired Protocols

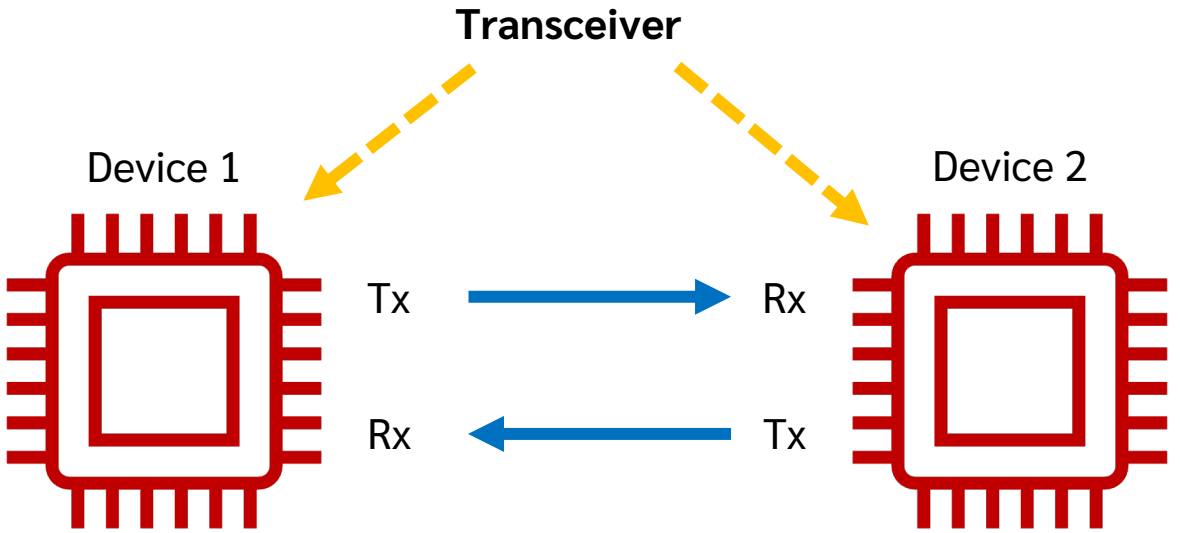
Universal Asynchronous Receiver/Transmitter (UART)

UART protocol is the simplest form of serial communication standard. It is **full duplex** as one device can transmit to another and vice versa *at the same time* without **signal collision**.

It needs 2 wires: One for transmitting (Tx) and one for receiving (Rx).

Note that Common ground (GND) between two devices is always required in every protocol.

The Common GND is for Voltage referencing.

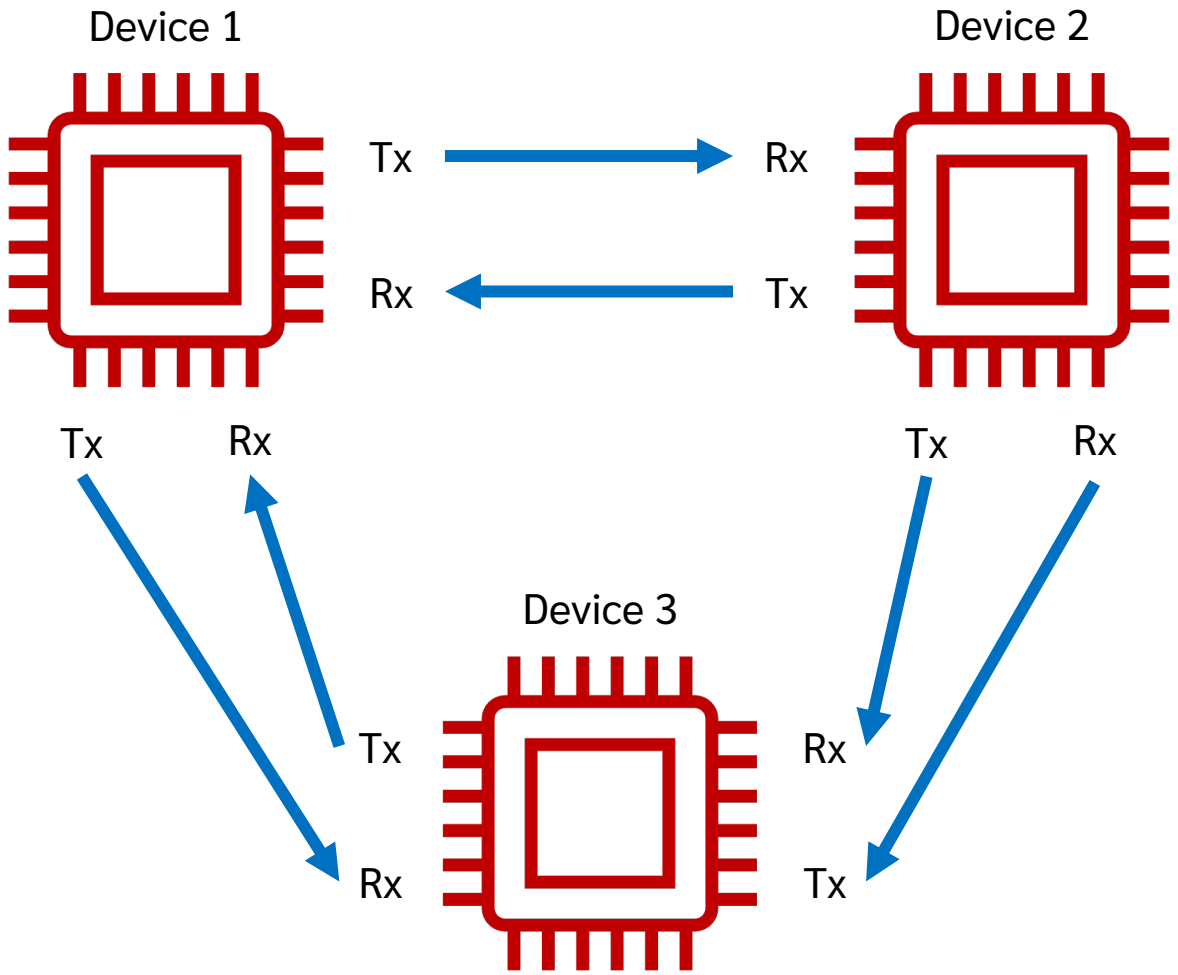


Microcontroller Connectivity: Wired Protocols

Universal Asynchronous Receiver/Transmitter (UART)

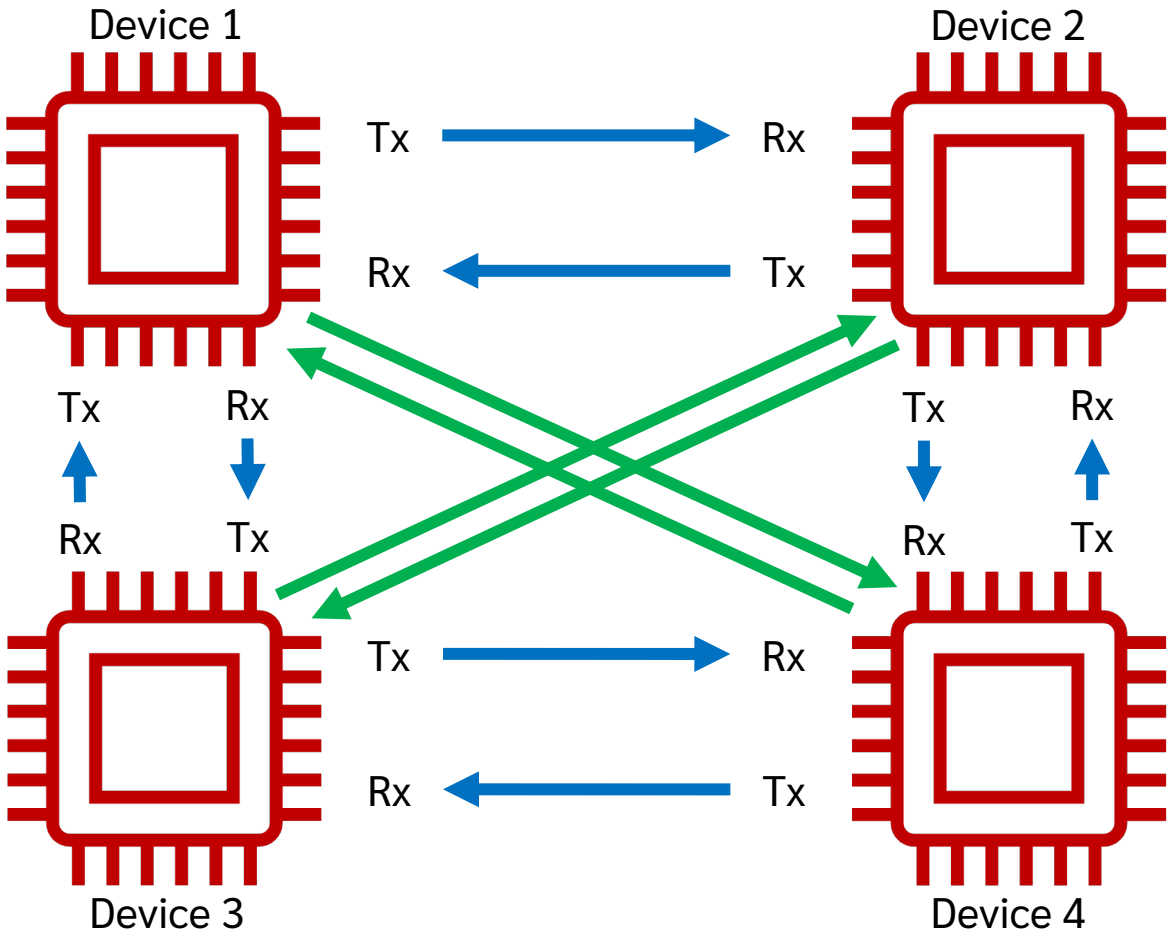
What about if you wanted to connect 3 devices?

It requires 6 wires just to make 3 devices talk to each other.



Microcontroller Connectivity: Wired Protocols

Universal Asynchronous Receiver/Transmitter (UART)



What about if you wanted to connect 4 devices?

What a *MESS*!!!

Imagine 10 devices? How many pins would each device needs if you want everyone to talk to everyone?

The answer is 2C(n, 2) where n is number of devices.

*10 devices: 2 * (10!)/(8! 2!) = 90 wires!!*

Microcontroller Connectivity: Wired Protocols

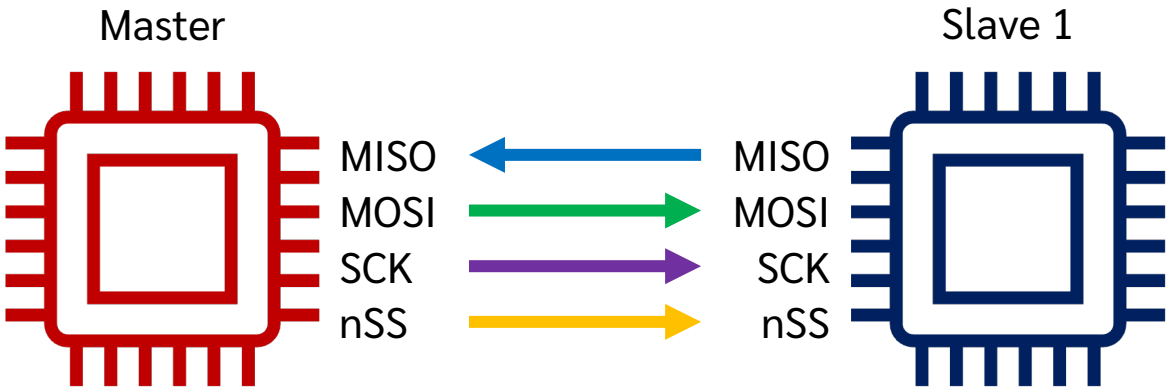
Serial Peripheral Interface (SPI)

SPI is an upgraded version of UART. It reduces the hassle of connecting every pair possible in the network by adding "**Chip-Select**" **(Slave-Select)** pin for each slave devices.

*Also, the UART has problems sending long messages over the network, so SPI adds "**Clock**" into action.*

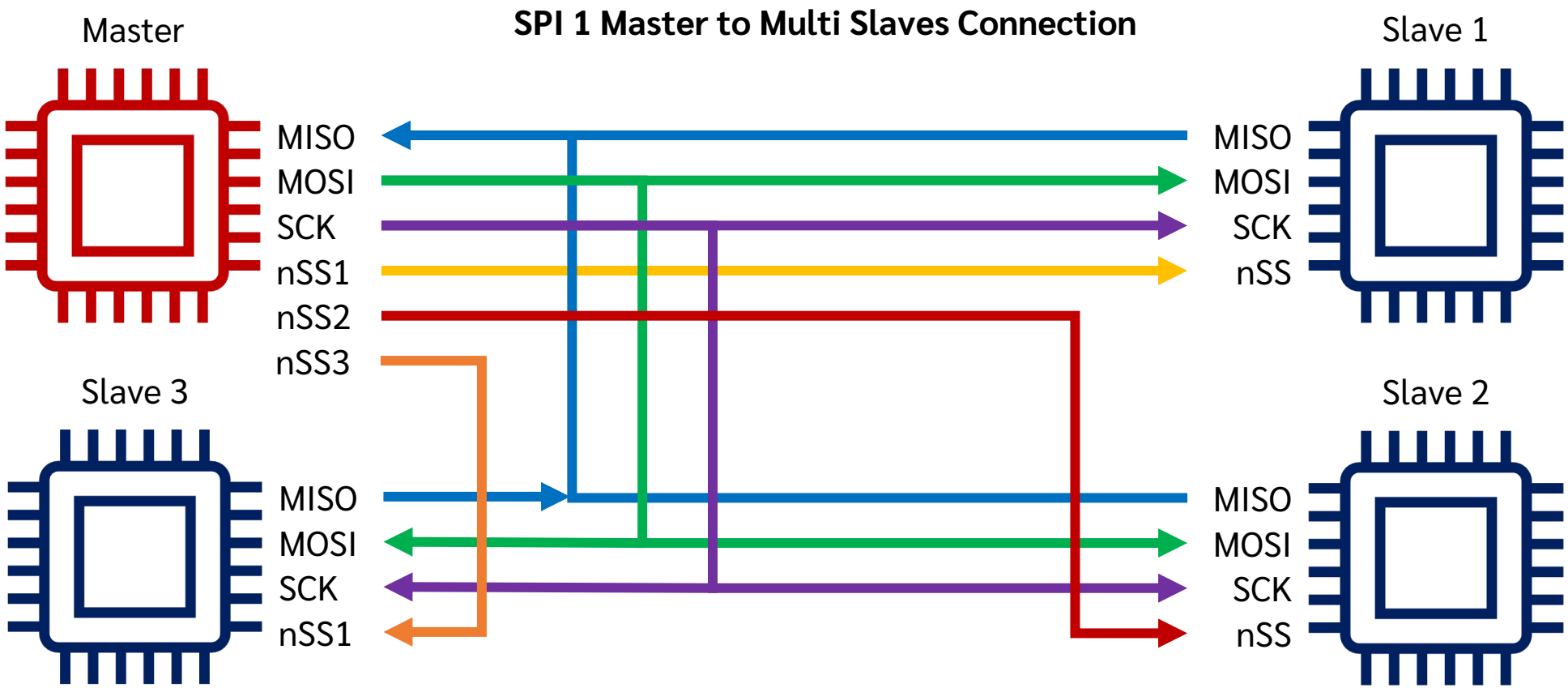
Tx and Rx pins can be connected in parallel. The names are changed to **Master-Out-Slave-In** (MOSI) and **Master-In-Slave-Out** (MISO) respectively.

All wires use **Push-pull** output, meaning the line is driven HIGH and LOW by MOSFET.



Microcontroller Connectivity: Wired Protocols

Serial Peripheral Interface (SPI)



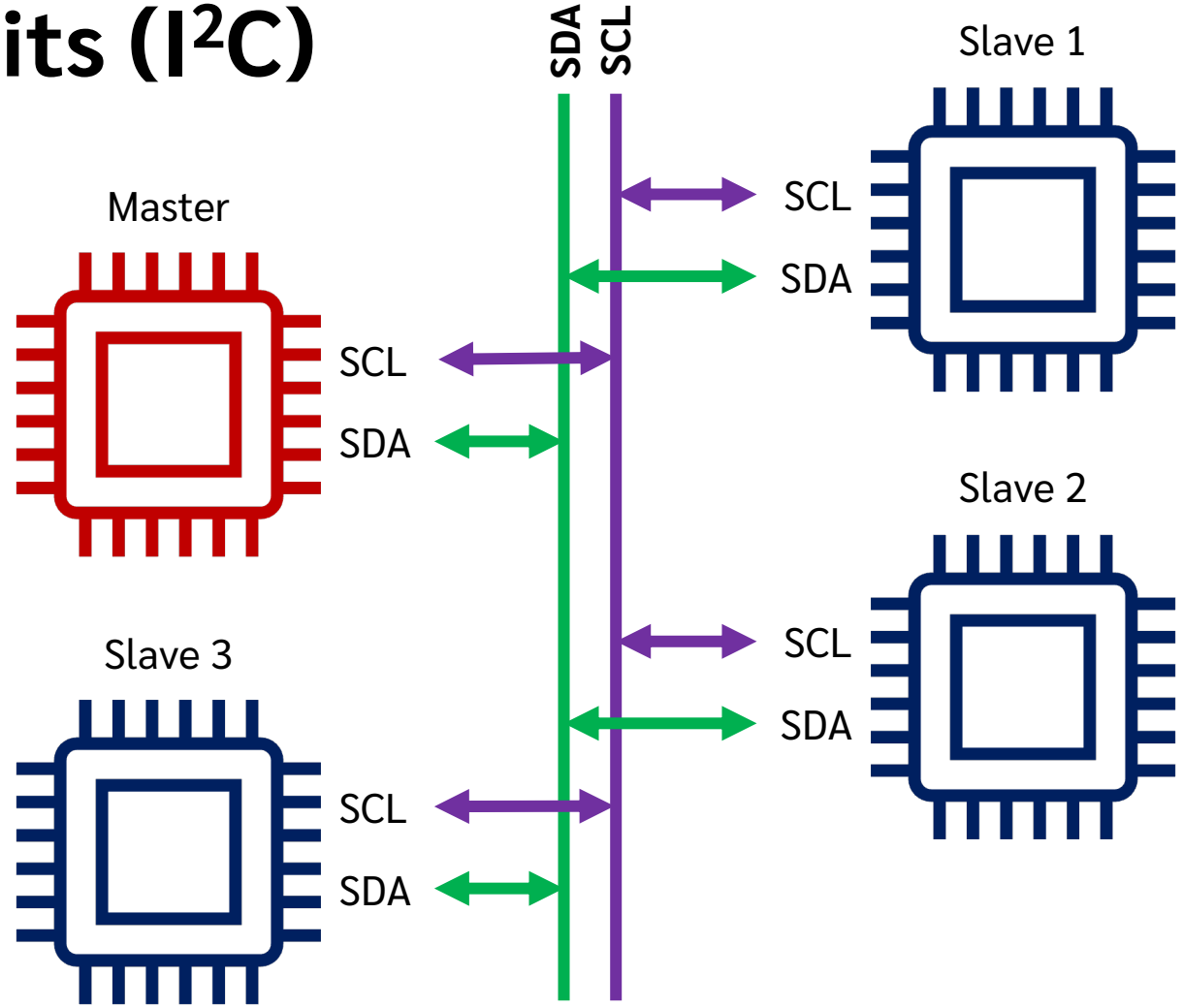
Microcontroller Connectivity: Wired Protocols

Inter-Integrated Circuits (I²C)

I²C is a *bus-based* multi-master multi-slave communication. This protocol is more complex than the last two.

It uses a central clock from master like SPI. But the data line can be accessed by every device on the bus.

SDA and SCL uses **Open-Drain** output, meaning the line has static HIGH, and when the line is needed, the current is *drained* to make the line LOW. This is also called **Wire-AND**.



New Naming Conventions

Nowadays educators, engineers, designers, and community members are encouraged to discontinue the use of the terms MOSI/MISO/SS and in their place use SDO/SDI/CS.

Master Devices are now often called "**Controller Devices**."

Slave Devices are now often called "**Peripheral Devices**."

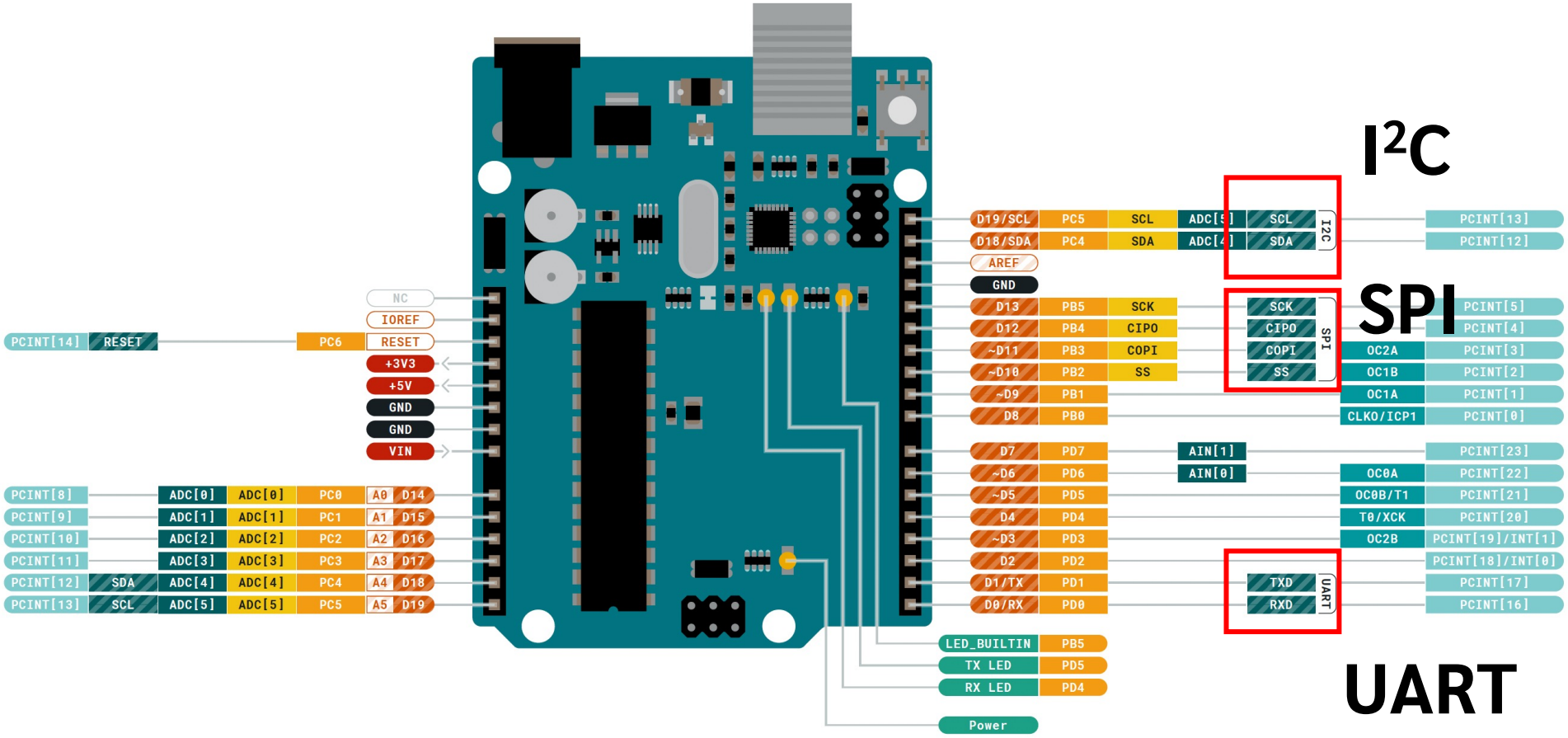
Master-In-Slave-Out (MISO) is equivalent to **Serial Data Out** (SDO).
There are also uses of "Controller-In-Peripheral-Out (CISO)."

Master-Out-Slave-In (MOSI) is equivalent to **Serial Data In** (SDI).
There are also uses of "Controller-Out-Peripheral-In (COSI)."

Slave Select (SS) is equivalent to **Chip Select** (CS).

Microcontroller Connectivity

Peripherals: Microcontroller



Microcontroller Connectivity

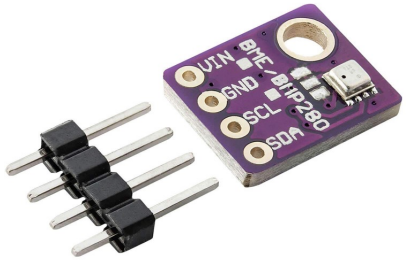
Peripherals: CanSat

Ground Station

LoRa

LoRa Module

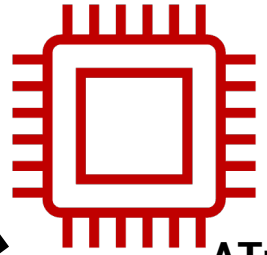
SPI



BME280

Temperature, Humidity, Pressure Sensor

I²C



ATmega328P

Voltage Reader

Analog

ADC_0

Voltage

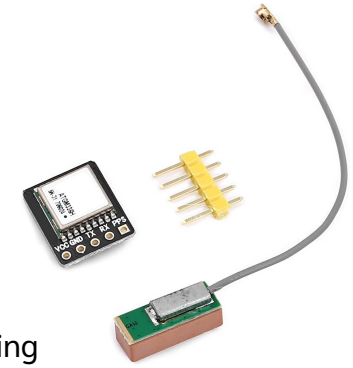
Battery Voltage

vUART (Software serial emulation)

LEDs and Buzzer

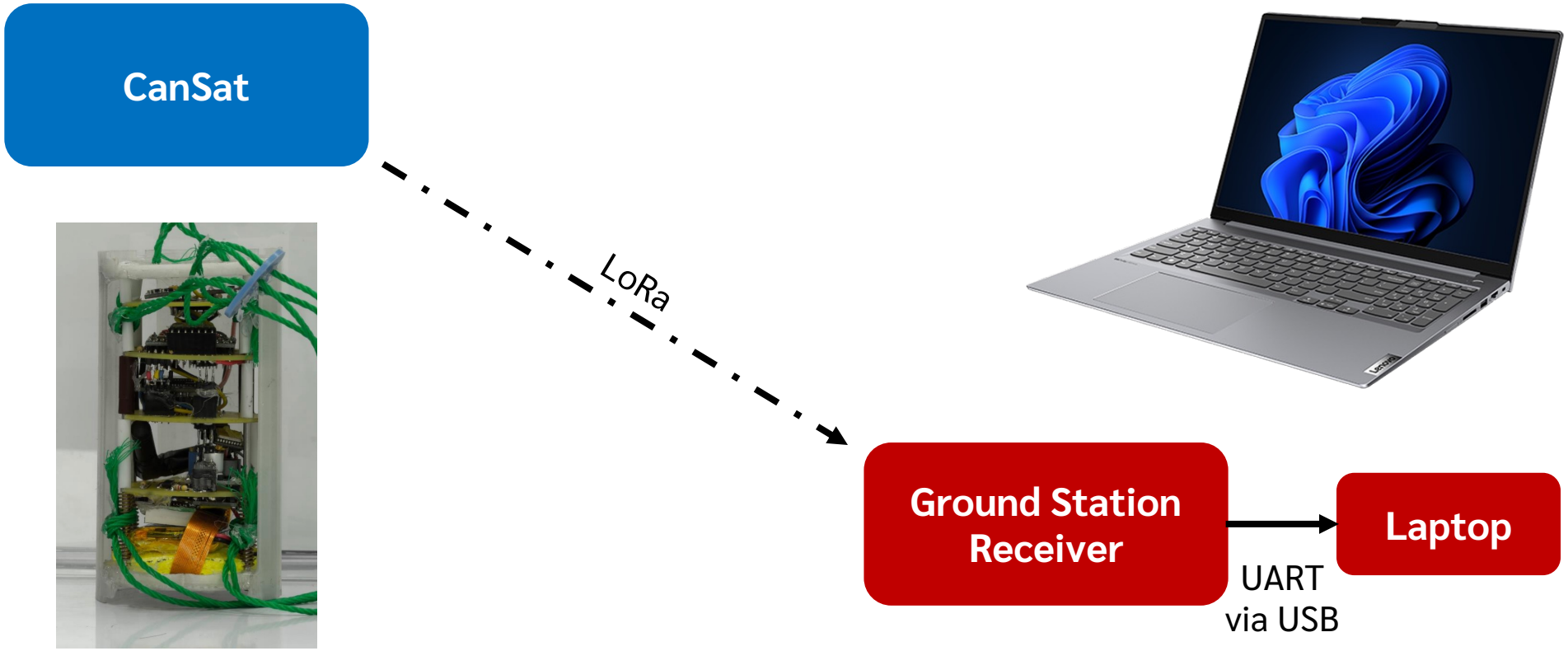
ATGM336H

GPS Sensor



Microcontroller Connectivity

CanSat-Ground Station

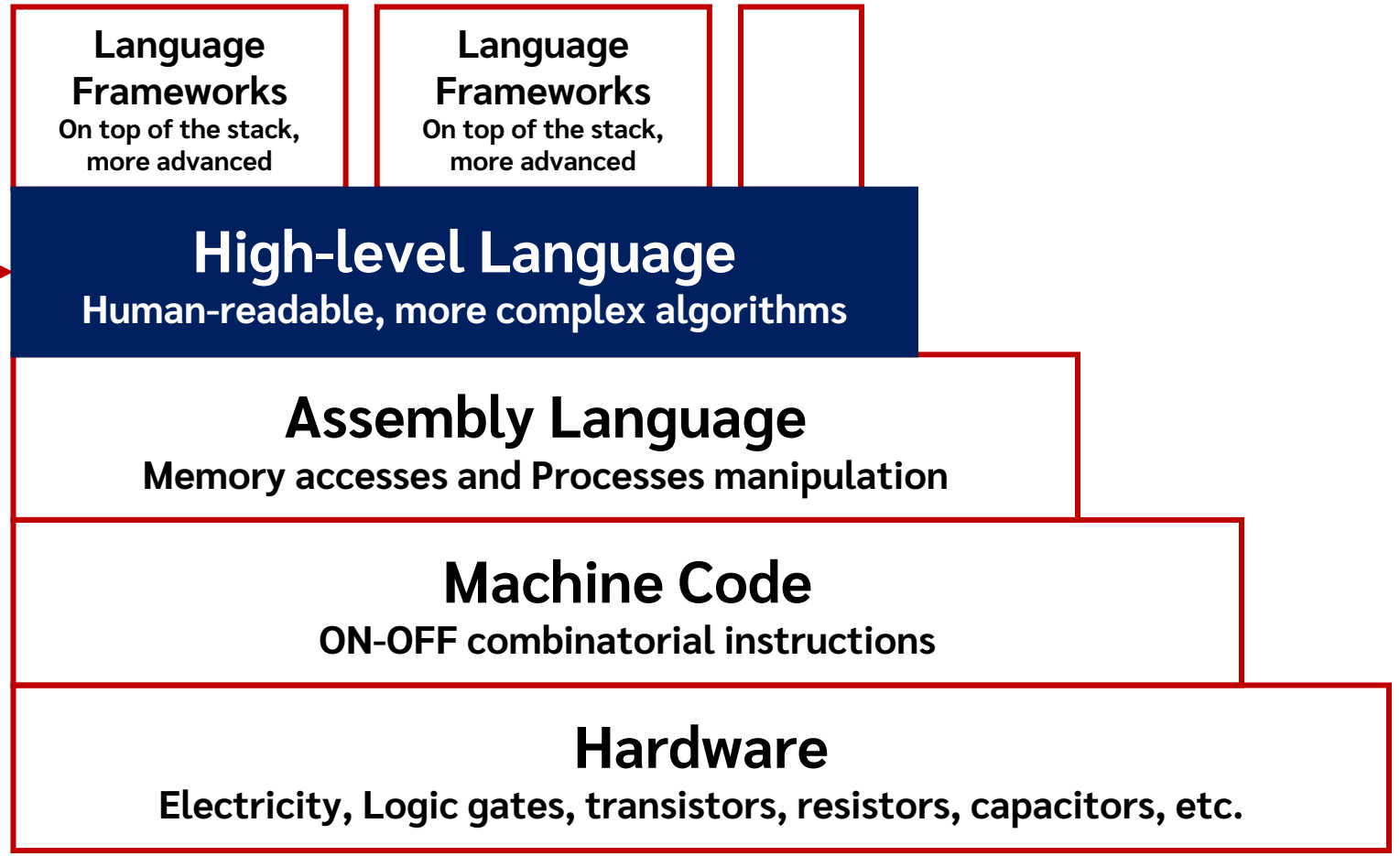


A Program

Computer Programming

Programming Language Level

We will be focusing here...

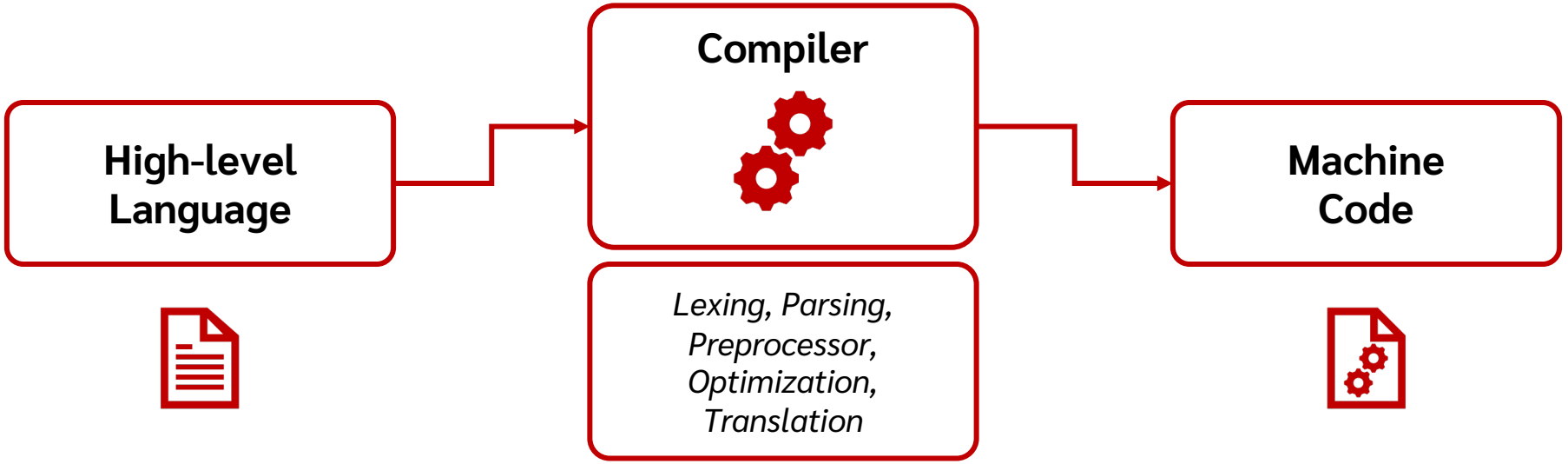


Computer Programming

Programming Language Level

1. Compiler

Example: C, C++



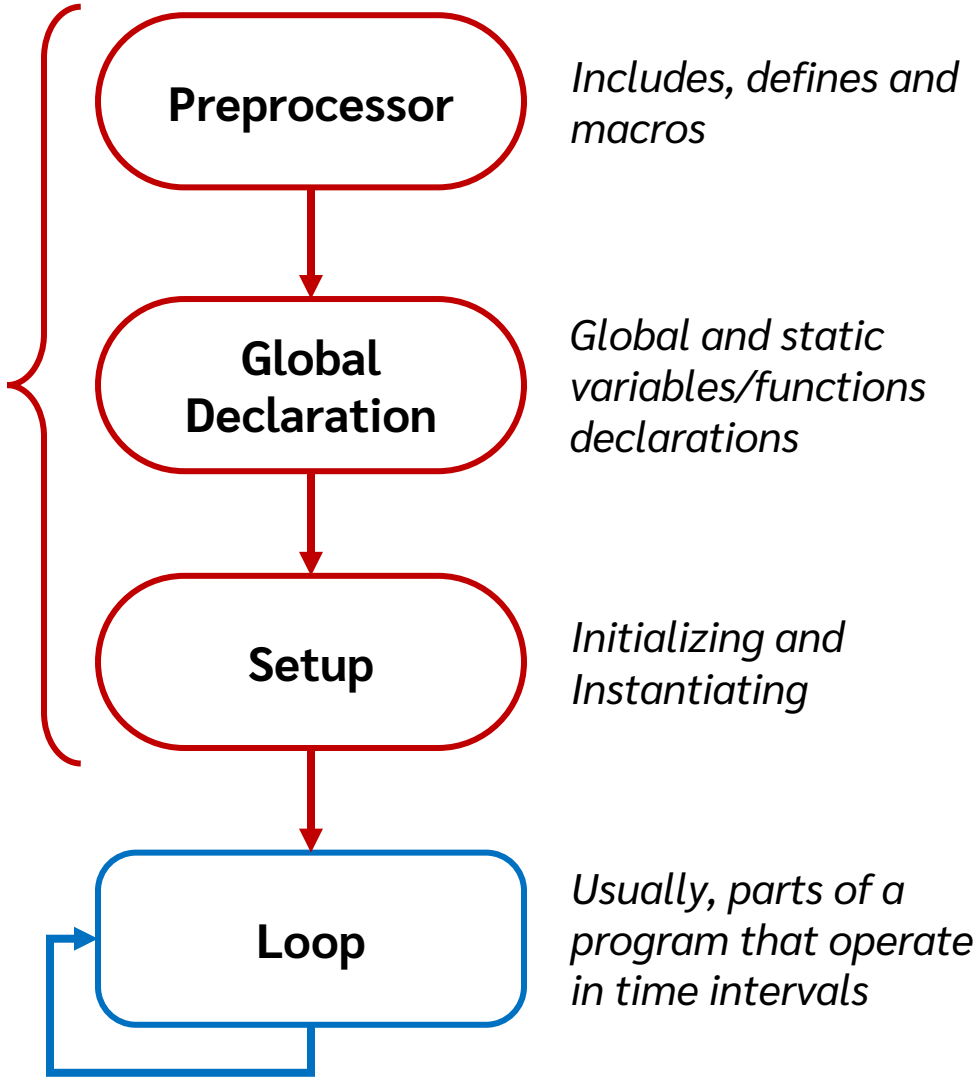
Microcontroller Program Flow

Setup-Loop Variant

In microcontroller programming, the scheme of operation is usually:

1. setup phase,
2. loop phase.

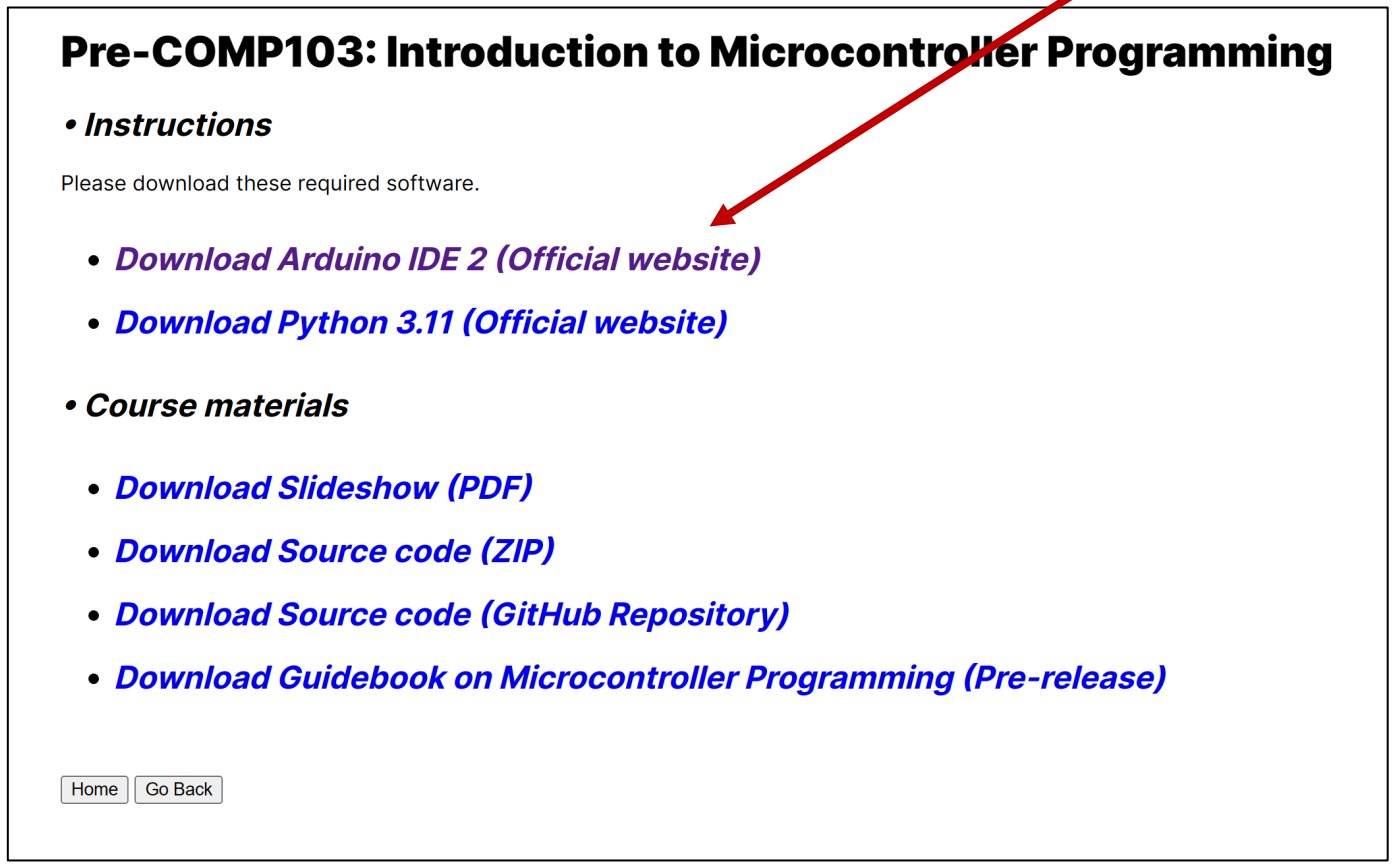
The scheme is standard across various designers and manufacturers. Because of that, the implementation is very similar across different brands. Arduino's Hardware Abstraction Layer (HAL) implements the scheme and create a user-friendly programming interface.



Arduino IDE

How to setup Arduino IDE (For Windows)

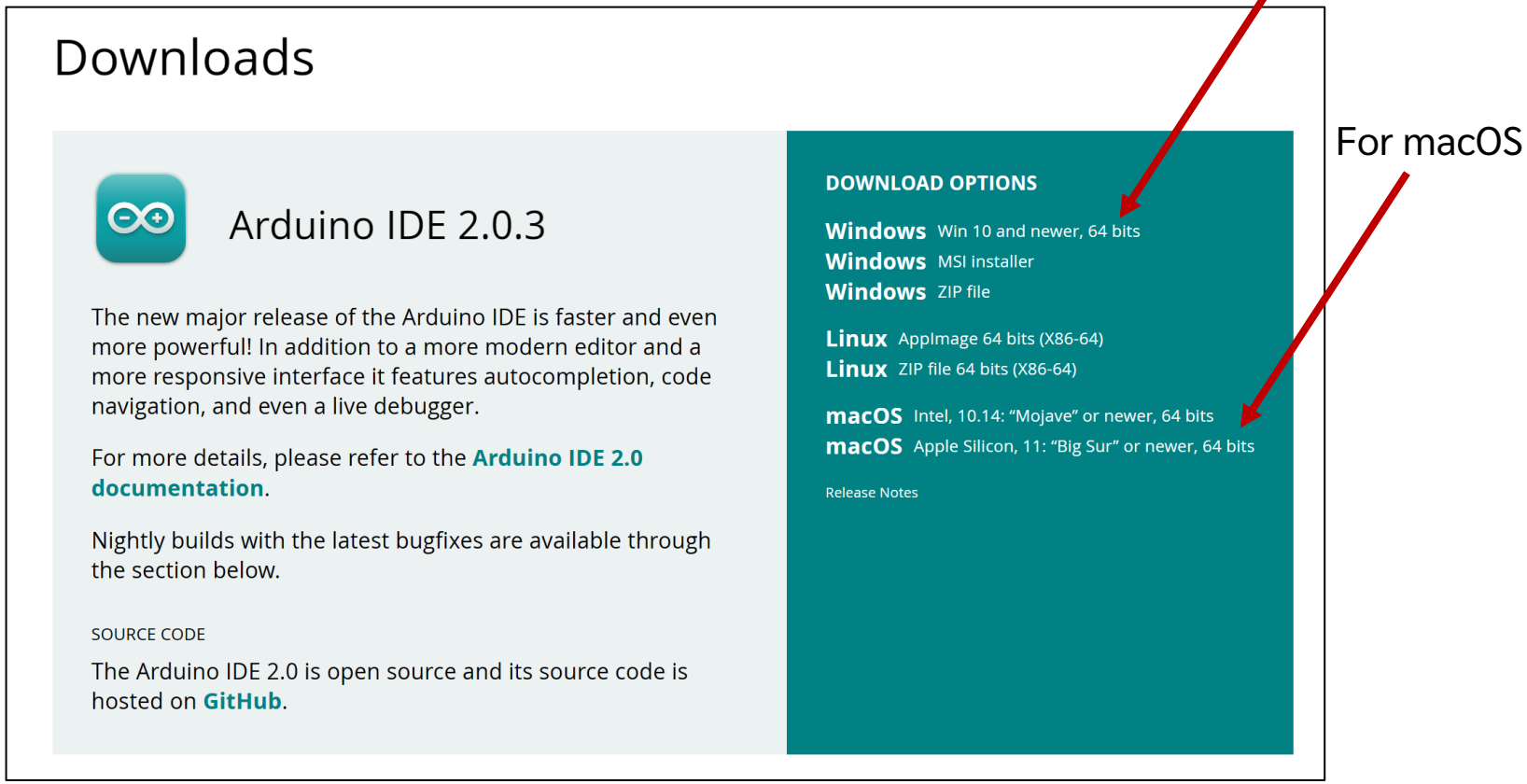
In course's website



Arduino IDE

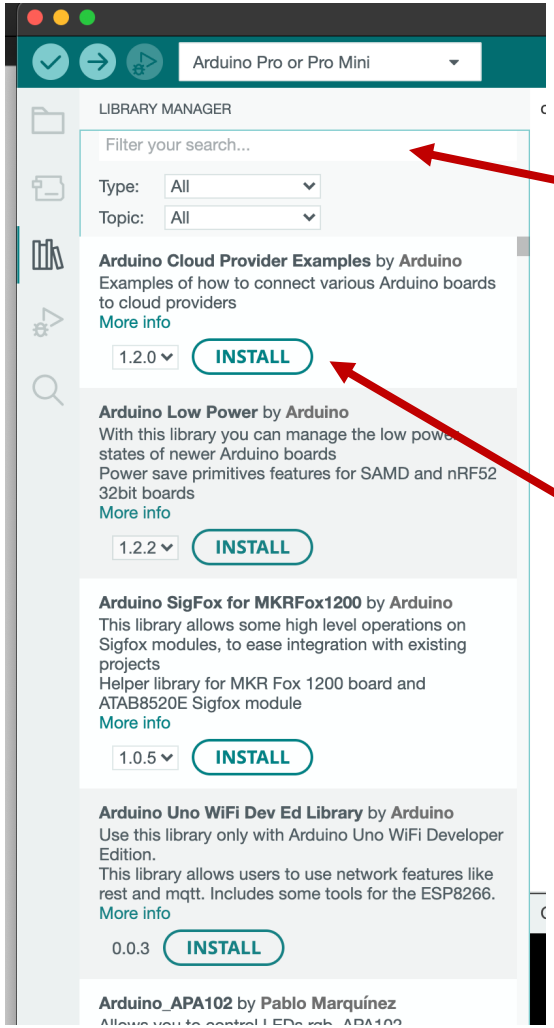
How to setup Arduino IDE (For Windows)

Arduino Website



Arduino IDE

Install Required Library



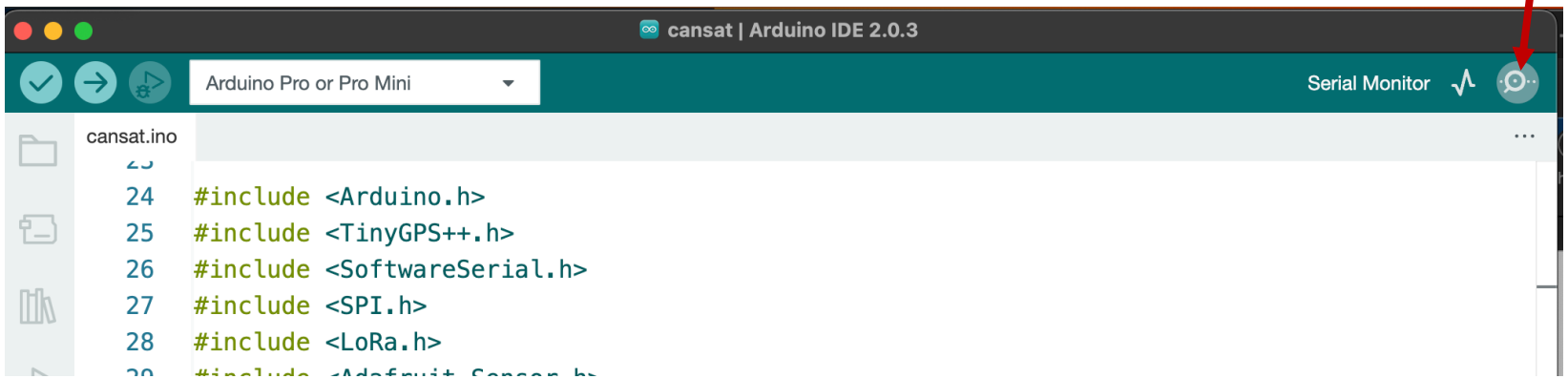
Search for Library Name

Click Install

Arduino IDE

Using Serial Monitor

Click here to open Serial Monitor



Arduino IDE

Using Serial Monitor

Click here to view Serial Monitor