CO3053 – Embedded Systems

- Introduction to Embedded Systems -

Course Introduction

- General Information
  - CO3053 - Embedded Systems

- Instructor
  - **Pham Hoang Anh, Ph.D**
  - Faculty of Computer Science and Engineering
  - HCMC University of Technology
  - Email: anhpham@hcmut.edu.vn
  - Phone: (84)(8) 38647256 (Ext. 5843)
  - Course materials: BKeL

Learning Outcome

- Interpret the embedded software development process
  - Describe each step in the process
  - Sort the steps in the process in correct order
- Express a software design idea in forms of flowchart and state machine
  - Describe a system by state machine
  - Describe an algorithm by flowchart
- Apply round-robin and event-driven models to develop an embedded software
  - Apply round-robin model
  - Apply event-driven model
- Explain the real-time constraints in embedded systems applications
  - Recognize the time response constraints
  - Analyze time response constraint corresponding to each application.

Tentative Contents

- What is embedded system?
- Embedded Platform Architecture
- Embedded System Development Process
- Programming models such as round robin, event-driven, RTOS
- Methods to present design such as flowchart, and state diagram.
- Misc. topics of emerging technologies
- Selected topics of embedded system development for ARM

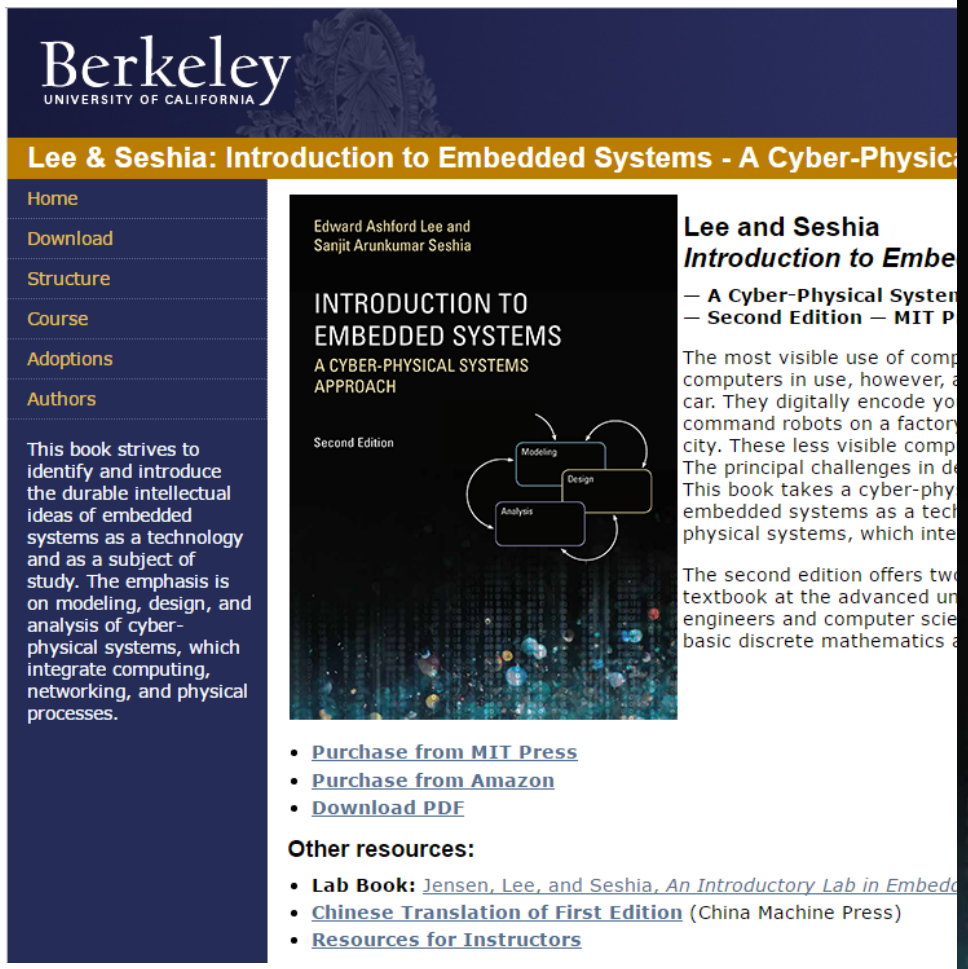
Grading Policy

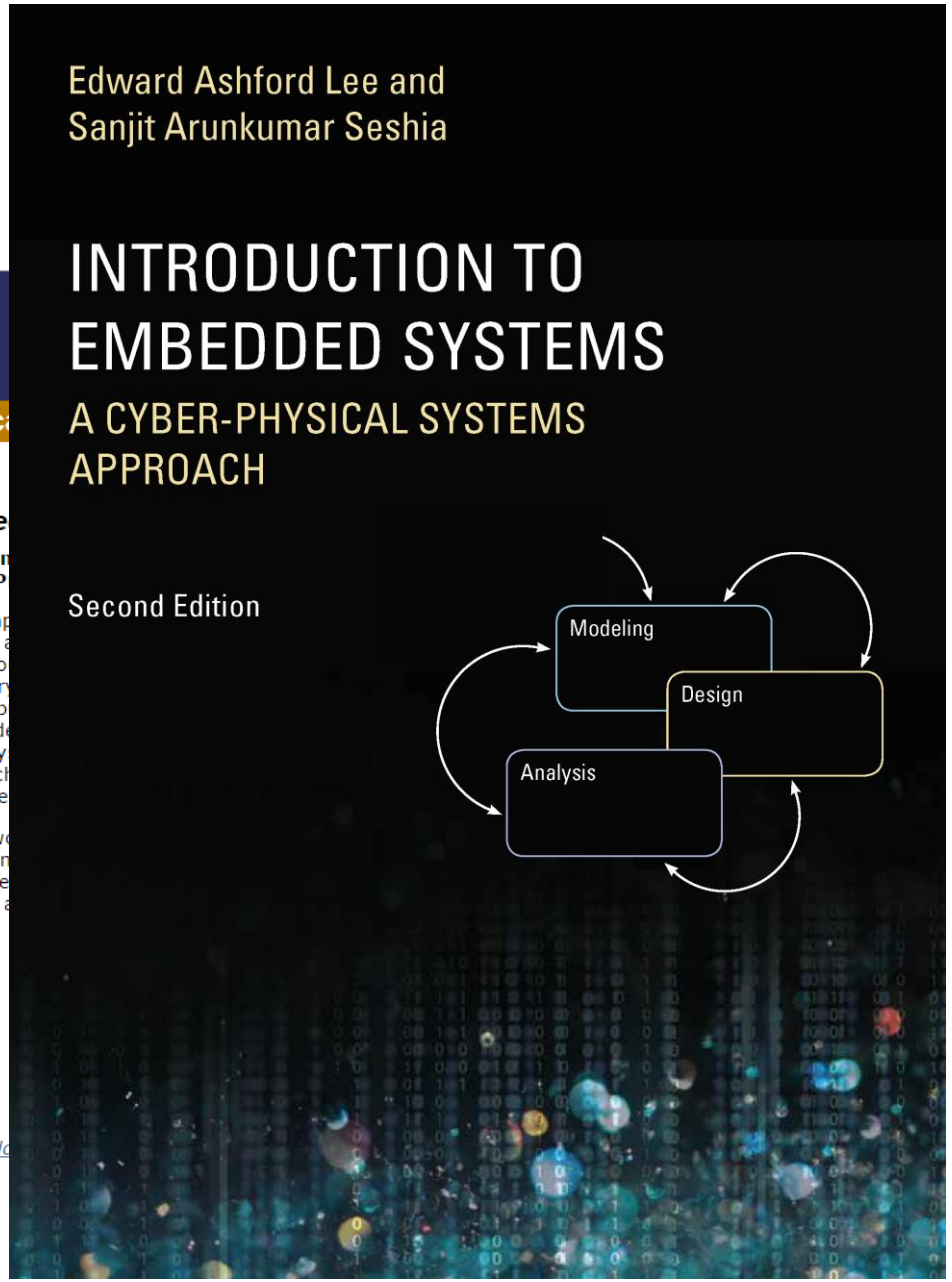
- Grading
  - Lab: 20%
  - Midterm: 20%
  - Project: 20%
  - Final Exam: 40%
- Assignment
  - Submit before due date
- Presentation
  - Journal Articles
  - Related Topics

https://www.hackster.io/projects?ref=topnav
http://www.electronicshub.org/iot-project-ideas

Textbooks

- http://leeseshia.org/





Labs

- ESP32
- STM32
- MicroBit
- Intel Edison
- Robotics Experiments

anhpham@hcmut.edu.vn

Course Project - Evaluation Criteria

- **Report**
  - Final Report (Project Description, Design, Implementation)
  - Group Meeting (Minutes)
  - Task Assignment
  - Presentation

- **Poster**
  - Image
  - Power Point (A0 - A1 size)

- **Implementation and Demonstration**
  - Source Code (+ Userguide)
  - Video (mp4, avi, youtube)

Presentation Topics

- ROS/RTOS
- Android Things
- Cyber-Physical Systems
- Internet of Things

Project Topics

- Smart Mirror
  - Voice Control by Alexa Amazon
- Indoor Localization
  - Wifi or BLE
- 17DOF Humanoid Robot
  - Hardware is available
- Driverless Automotive
  - LIDAR …
- Bluetooth Mesh Network
  - Gateway Implementation
  - Application Development
- LoRa Testbed Development
  - Sensor Nodes, Gateway
- Position Tracking
  - LoRa + GPS
- Robotic vacuum cleaner
  - Movement Control
  - Movement Planning
  - …
- Air Quality Monitoring Hub
  - Solar Power
  - GPRS/3G Communication

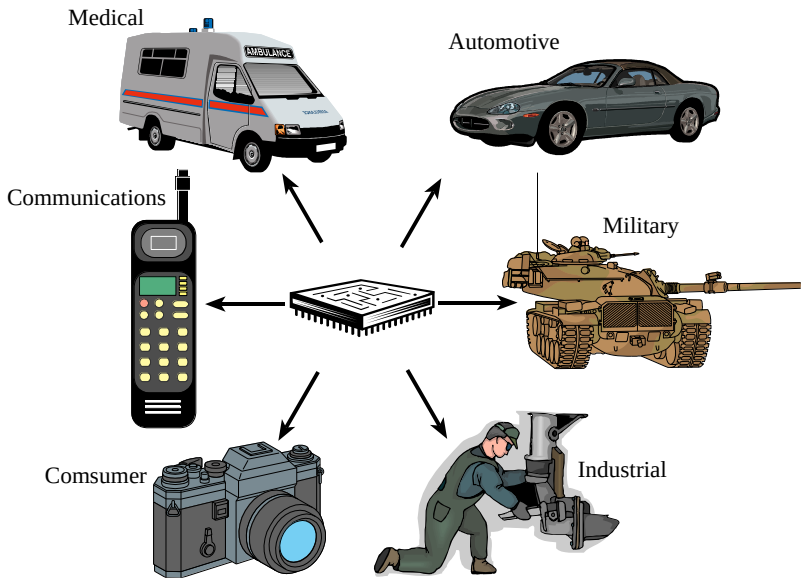
Question and Discussion?

Review

- ADC
- UART
- SPI
- I2C
- PWM

CO3053 – Knowledge Review

ADC , UART, I2C , SPI , PWM



Contents

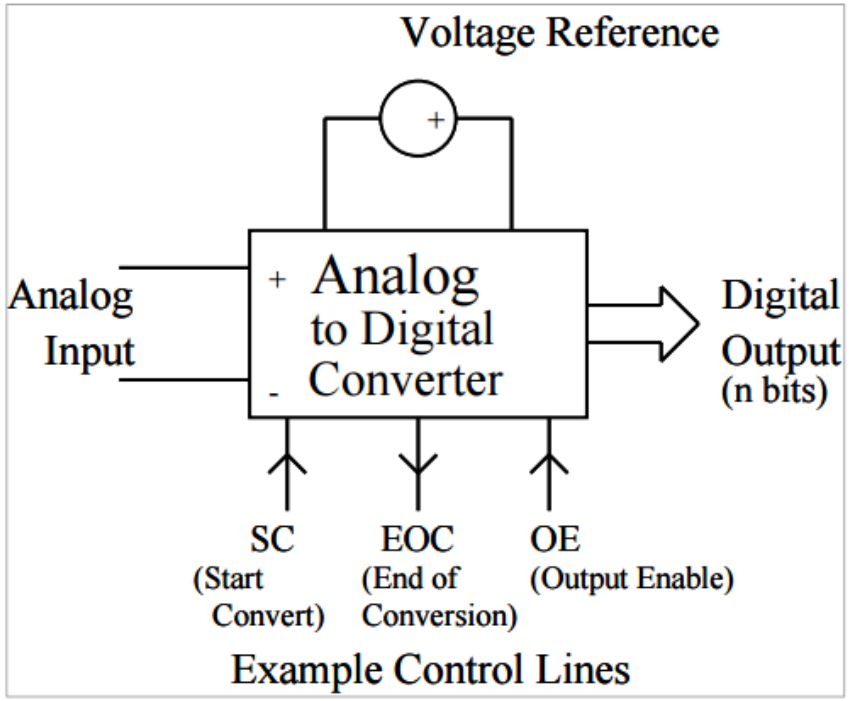
- Analog to Digital Conversion (ADC)
- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I2C)
- Pulse Width Modulation (PWM)

Analog-To-Digital Conversion (ADC)

- Embedded system applications are often required to interface with analog signals.
- They must be able to convert **input analog signals**, for example from microphone or temperature sensor, **to digital data.**
- They must also be able to convert digital signals to analog form, for example if driving a loudspeaker or dc motor
- We will first consider conversion from analog-to-digital, before later looking at digital-to-analog conversion

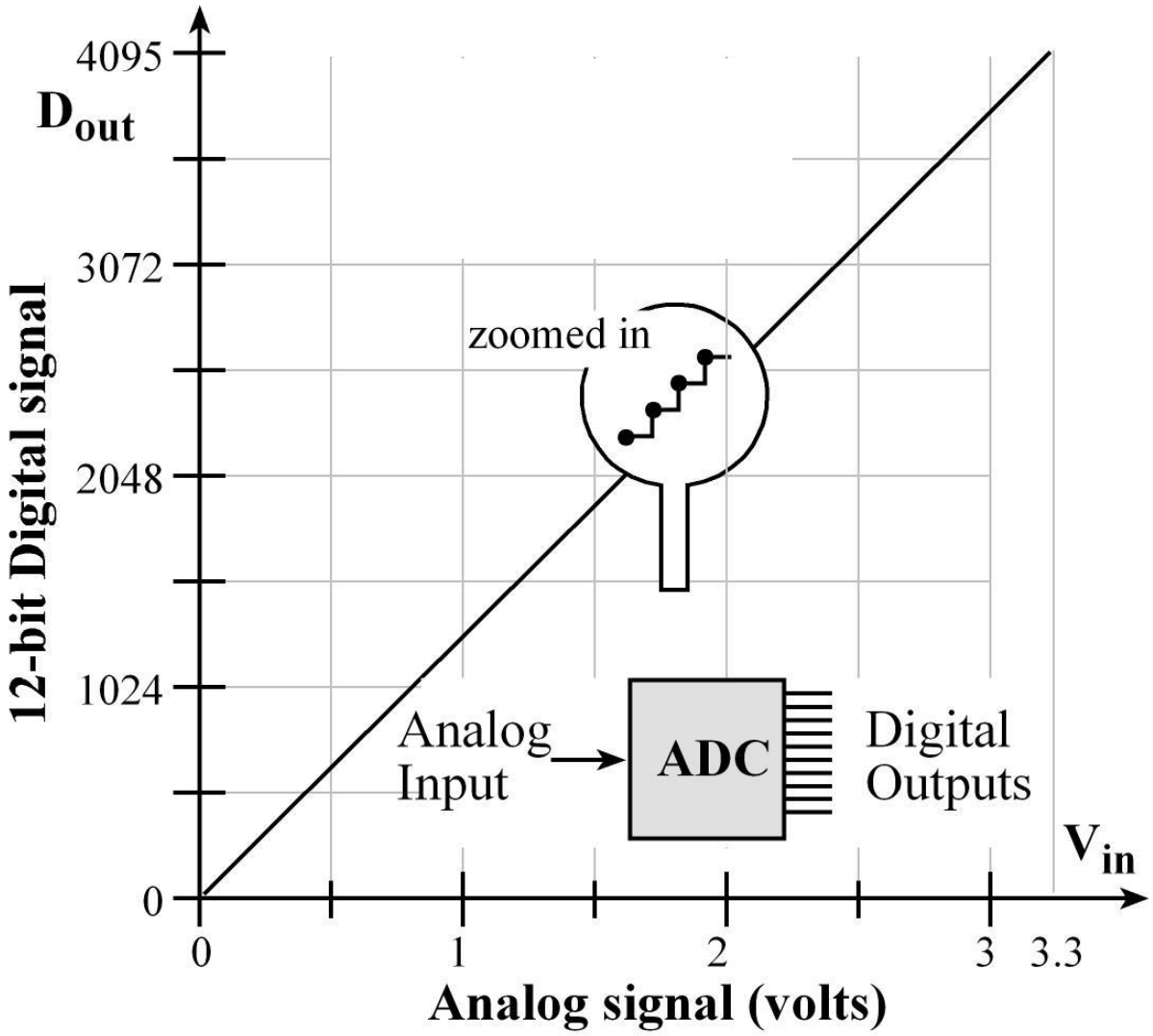
Concepts of Analog-To-Digital Conversion

- An analog-to-digital convertor (ADC) is an electronic circuit whose digital output is proportional to its analog input.
- Effectively it "measures" the input voltage, and gives a binary output number proportional to its size.
- The input range of the ADC is usually determined by the value of **voltage reference**.



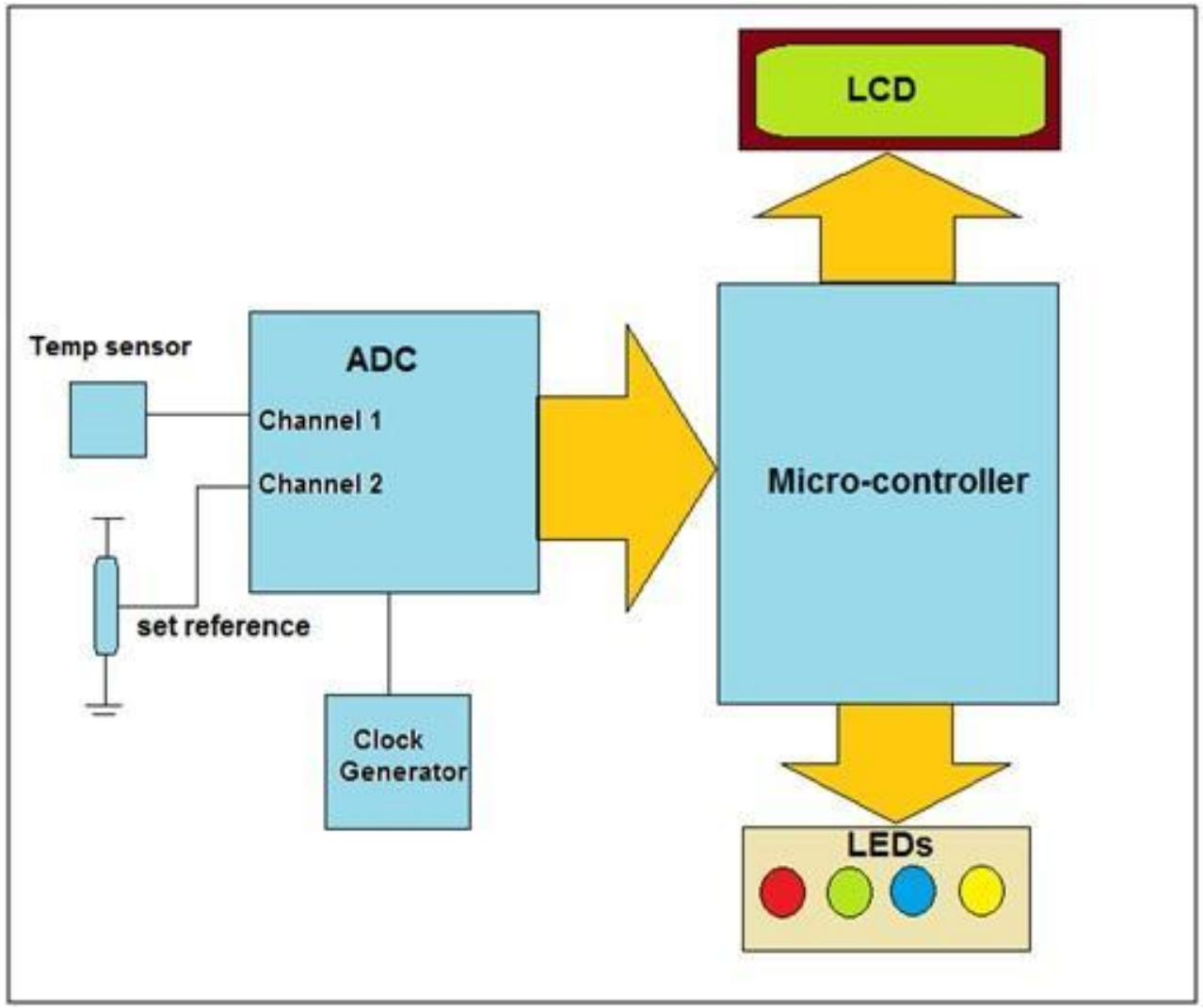
Analog-To-Digital Conversion

- Example



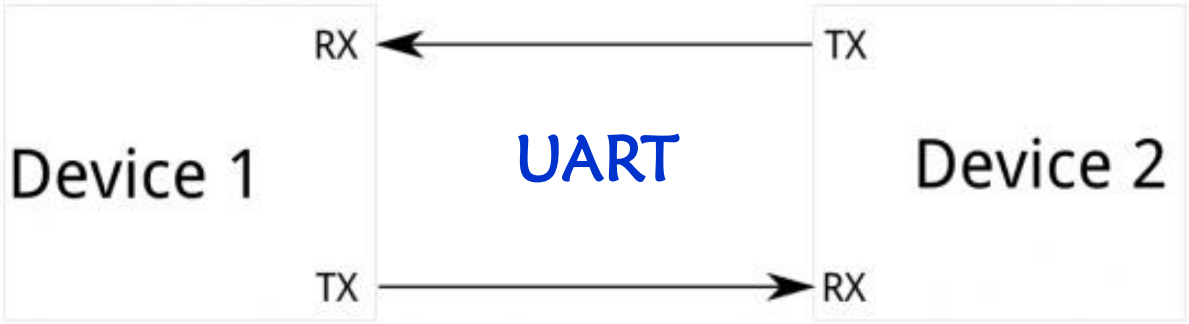
Data Acquisition System

- Example



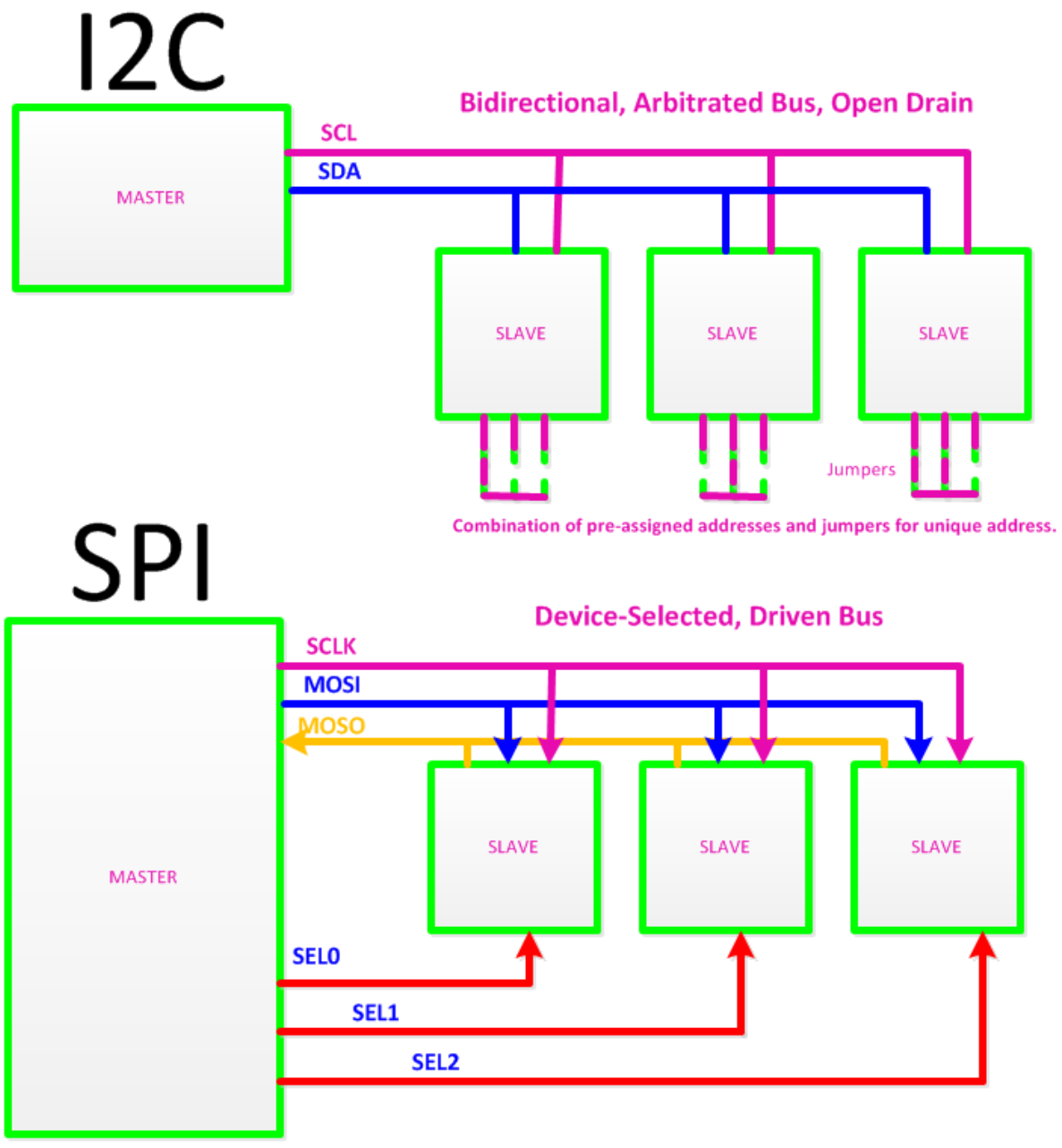
What's Wrong with Serial Ports?

- A common serial port, the kind with TX and RX lines, is called "**asynchronous**" (not synchronous) because there is no control over when data is sent or any guarantee that both sides are running at precisely the same rate.
  - Since computers normally rely on everything being synchronized to a single "clock", this can be a problem when two systems with slightly different clocks try to communicate with each other.



→ Synchronous Serial Communication

SPI vs. I2C

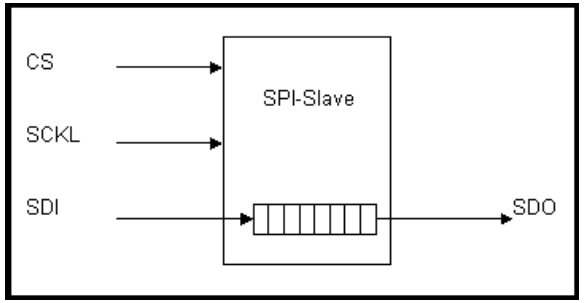

I2C
Bidirectional, Arbitrated Bus, Open Drain
MASTER
SCL
SDA
SLAVE
SLAVE
SLAVE
Jumpers
Combination of pre-assigned addresses and jumpers for unique address.
SPI
Device-Selected, Driven Bus
MASTER
SCLK
MOSI
MOSO
SLAVE
SLAVE
SLAVE
SEL0
SEL1
SEL2

Serial Peripheral Interface (SPI)

- Developed by Motorola
  - Also known as MicroWire (National Semiconductor), QSPI (Queued), MicrowirePlus.
  - Synchronous Serial Communication.

- Primarily used for **serial communication** between **a host processor and peripherals**.

- Can also connect 2 processors via SPI



- SPI works in a master slave configuration with the master being the host microcontroller for example and the slave being the peripheral

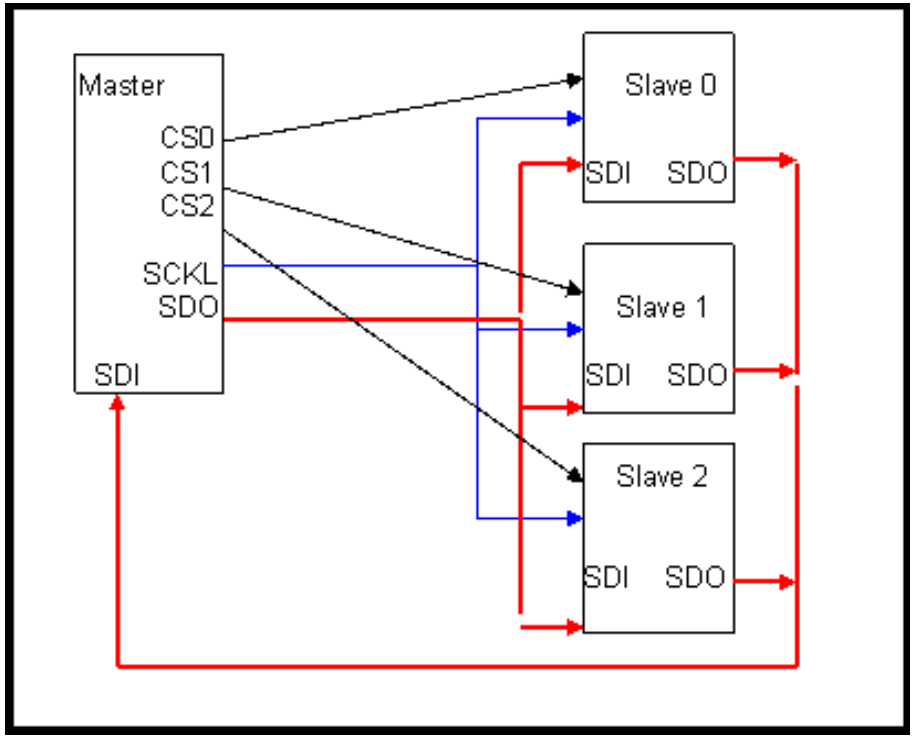
SPI Operation

- For SPI, there are Serial Clocks (SCLK), Chip Select lines (CS), Serial Data In (SDI) and Serial Data Out(SDO).
- There is only one master, there number of slaves depends on the number of chip select lines of the master.
- Synchronous operation, latch on rising or falling edge of clock, SDI on rising edge, SDO on falling edge.
- Master sends out clocks and chip selects. Activates the slaves it wants to communicate with.
- Operates in **1 to 2 MHz** range.

SPI – Master Slave Setup

- In this setup, there are 3 slave devices. The SDO lines are tied together to the SDI line of the master.
- The master determines which chip it is talking to by the CS lines. For the slaves that are not being talked to, the data output goes to a Hi Z state

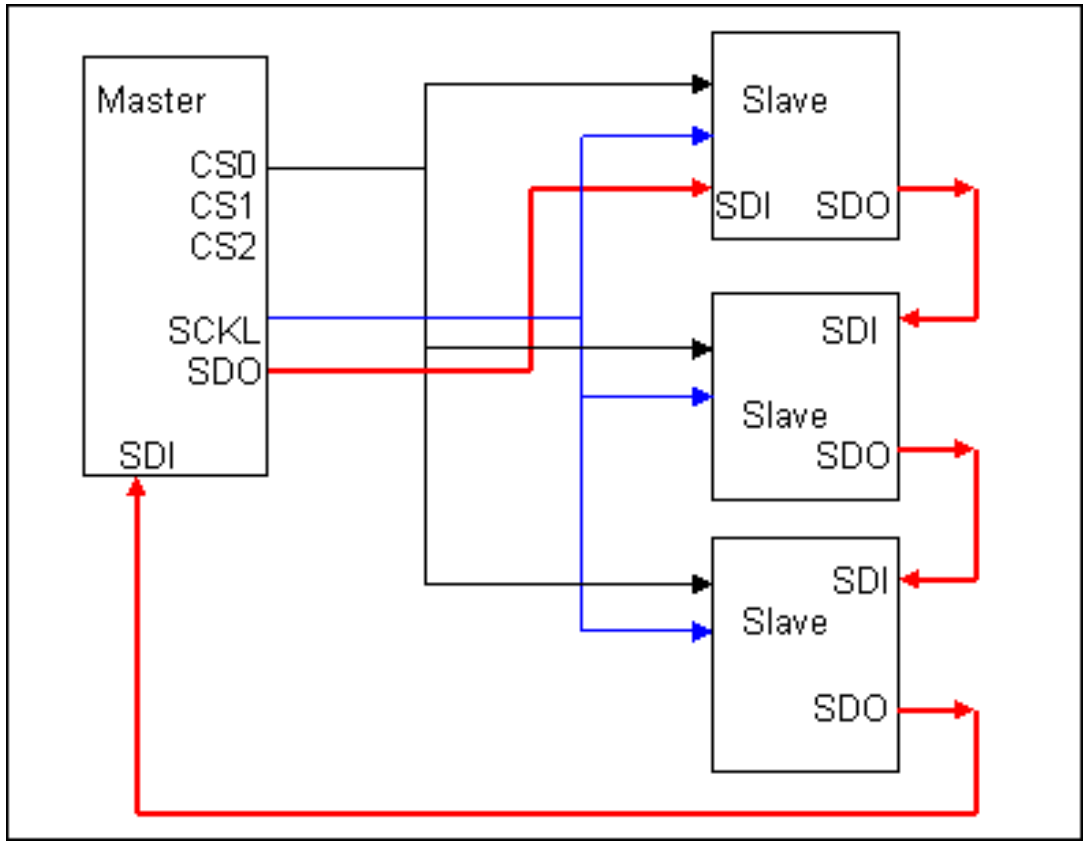
Multiple Independent Slave Configuration



SPI – Master Slave Setup

- In this example, each slave is cascaded so that the output of one slave is the input of another. When cascading, they are treated as one slave and connecting to the same chip select.

Multiple slave cascaded configuration



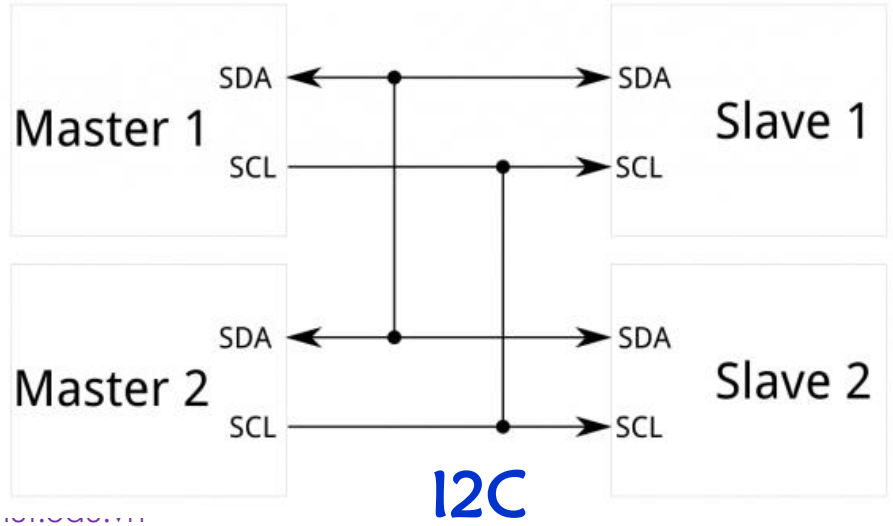
SPI Peripherals Types

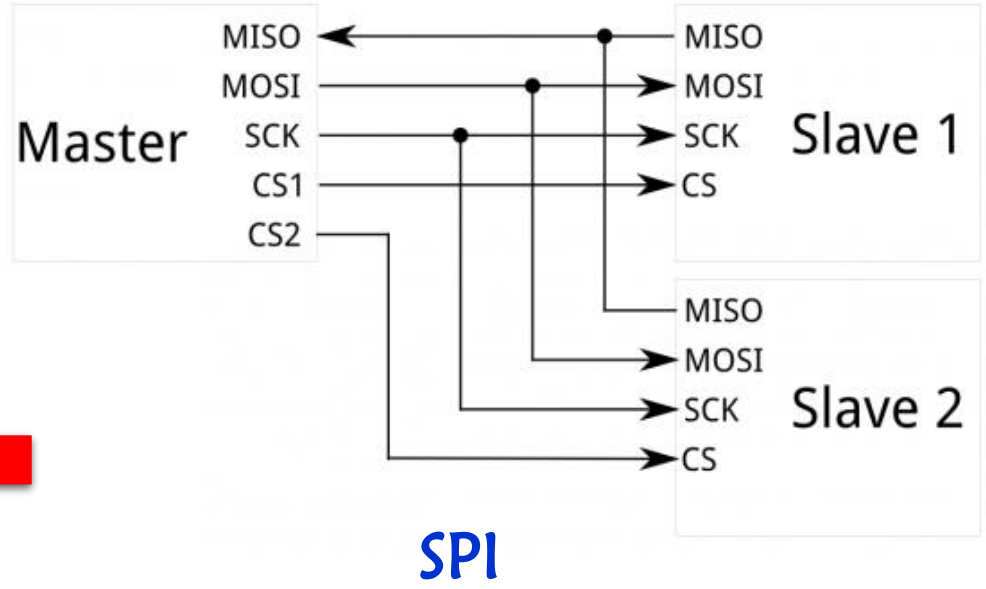
- **Types**
  - Converters (ADC, DAC)
  - Memories (EEPROM, RAM's,Flash)
  - Sensors (Temperature, Humidity, Pressure)
  - Real Time Clocks
  - Misc-Potentiometers, LCD controllers, UART's, USB controller, CAN controller, amplifiers.

- **Vendors that make these peripherals**
  - Atmel –EEPROM, Dig. POT's
  - Infineon- Pressure Sensors, Humidity Sensors
  - Maxim- ADC, DAC, UART,
  - TI- DSP's, ADC, DAC
  - National Semiconductor-Temperature Sensors, LCD/USB controllers

What's Wrong with SPI?

- The most obvious drawback of SPI is **the number of pins** required. Connecting a single master to a single slave with an SPI bus requires four lines; each additional slave requires one additional chip select I/O pin on the master.
- SPI only **allows one master on the bus**, but it does support an arbitrary number of slaves (subject only to the drive capability of the devices connected to the bus and the number of chip select pins available).





Inter-Integrated Circuit (I2C)

- I²C requires a mere **two wires**, like asynchronous serial, but those two wires can support up to **1008 slave devices**. Also, unlike SPI, I²C can support a multi-master system, allowing more than one master to communicate with all devices on the bus.

- Data rates fall between asynchronous serial and SPI.
  - Most I²C devices can communicate at **100kHz or 400kHz**.

- There is some overhead with I²C.
  - For every 8 bits of data to be sent, one extra bit of meta data (the "ACK/NACK" bit, which we'll discuss later) must be transmitted.

- The hardware required to implement I²C is more complex than SPI, but less than asynchronous serial.

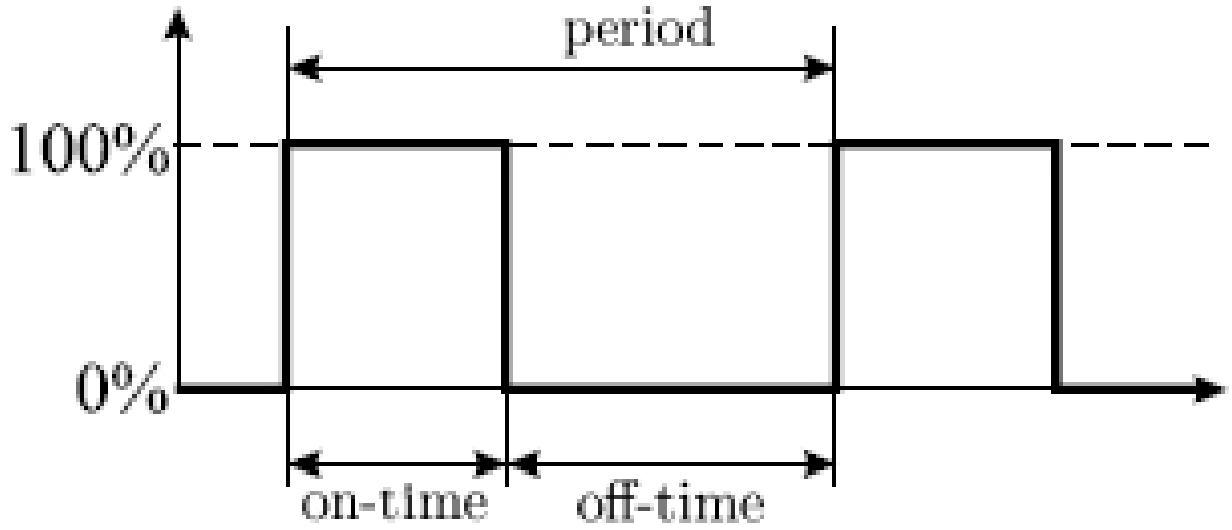
Pulse Width Modulation (PWM)

- **Pulse Width Modulation (PWM)** is a technique that conforms a signal width, generally pulses based on modulator signal information.
- The general purpose of PWM is to control power delivery, especially to inertial electrical devices.
- The on-off behavior changes the average power of signal.
- Output signal alternates between on and off within a specified period.
- If signal toggles between on and off quicker than the load, then the load is not affected by the toggling.
- A secondary use of PWM is to encode information for transmission.

Pulse Width Modulation (PWM)

- Duty Cycle
  - **D = on-time/period**
- V_{LOW} is often zero.

$$\mathbf{V_{AVG} = DV_{HI} + (1 - D)V_{LOW}}$$



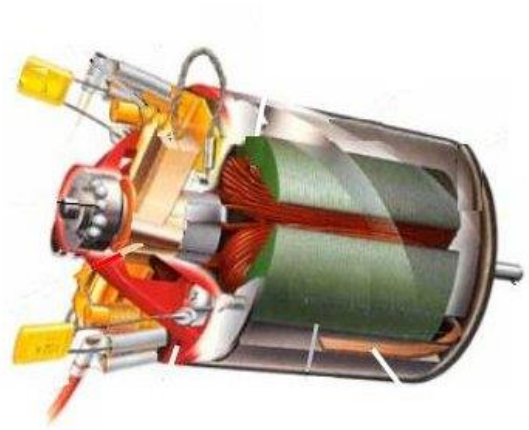
Pulse Width Modulation

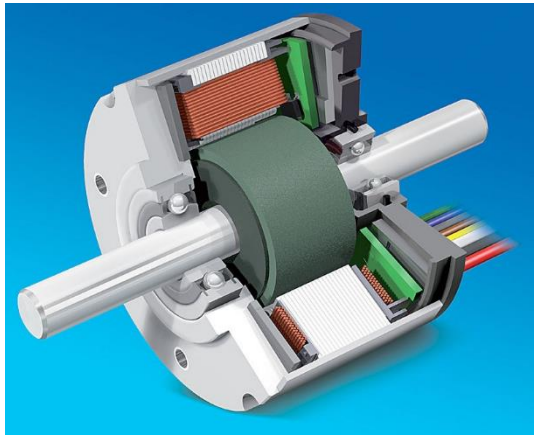
- **Advantages**
  - Average value proportional to duty cycle, D.
  - Low power used in transistors used to switch the signal.
  - Fast switching possible due to MOSFETS and power transistors at speeds in excess of 100 kHz.
  - Digital signal is resistant to noise.

- **Disadvantages**
  - Cost
  - Complexity of circuit
  - Radio Frequency Interference
  - Voltage spikes
  - Electromagnetic noise

Applications to DC Motors

- The voltage supplied to a DC motor is proportional to the duty cycle.
- Both brushed and brushless motors can be used with PWM
- Both analog and digital control techniques and components are available.





PWM with 555 Timer

- Potentiometer is used to adjust the duty cycle