# Embedded computing for scientific and industrial imaging applications

Lecture 1 - Goal of course, overview, requirements.

HanByul Yang
(Senior Engineer @ Samsung Medison)

- 양한별 (HanByul Yang)
- Senior Engineer @ Samsung Medison
- yhbyhb@yonsei.ac.kr

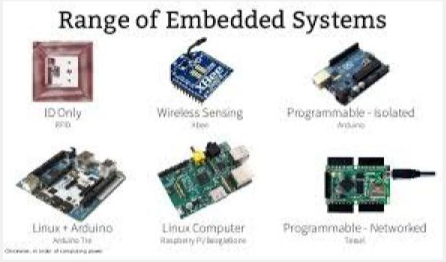
# Assignments and grading

- See course webpage / class notes for schedule and assignments.
- 3 homework assignments and a final project
- These will be turned in by pushing to a GitHub repository.

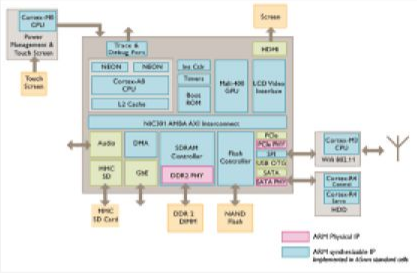
# Embedded computing?

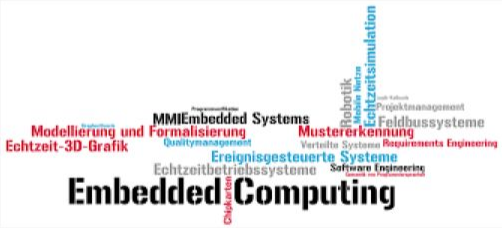
embedded computing

All | Images | News | Videos | Maps | More | Search tools | View saved | SafeSearch










# Embedded system?

Google embedded systems

All Images Videos Books News More Search tools View saved SafeSearch

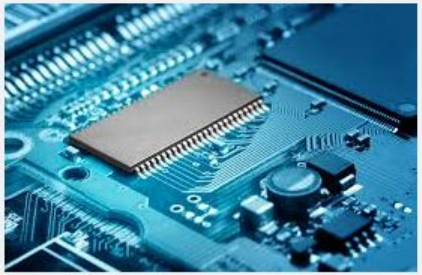

Wallpapers

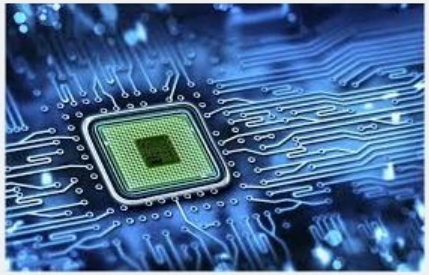

Examples

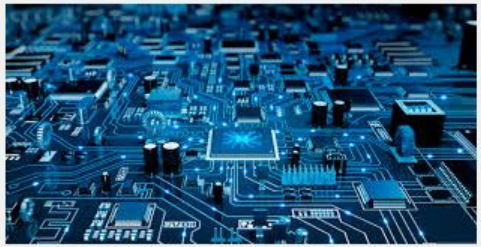
Logo

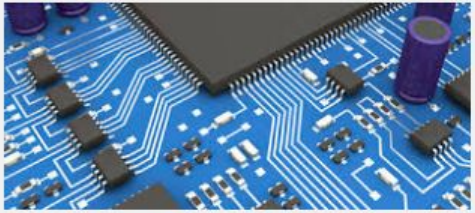

Design

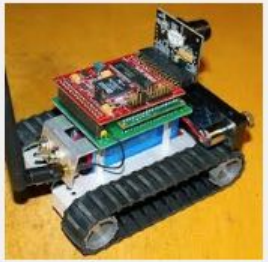

Projects












# Embedded system

- Any computer that is a component in a larger system and that relies on its own microprocessor  - *Wayne Wolf, Princeton University, What_is_embedded_computing*

- An embedded system is a computer system with a dedicated function within a larger mechanical or electrical system.often with real-time computing constraints. - *https://en.wikipedia.org/wiki/Embedded_system*

# Embedded system

- Low power consumption, small size, rugged operating ranges, and low per-unit cost. limited processing resources,
- Modern embedded systems are often based on microcontrollers (i.e. CPUs with integrated memory or peripheral interfaces)
- dedicated to specific tasks, reduce the size and cost of the product and increase the reliability and performance.
- range from portable devices to large stationary installations (smart phones, vehicles, medical devices such as MRI, CT and US)
- Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

# Example - AP (Application processor)

- Memory, Display system/controller
- Multimedia en/decoding codec
- 2D/3D accelerator engine,
- ISP(Image Signal Processor),
- Camera, Audio, Modem,
- High & low speed Serial/Parallel connectivity interface

⇒ SOC(System-On-Chip)

*ref : https://news.samsung.com/kr/482*

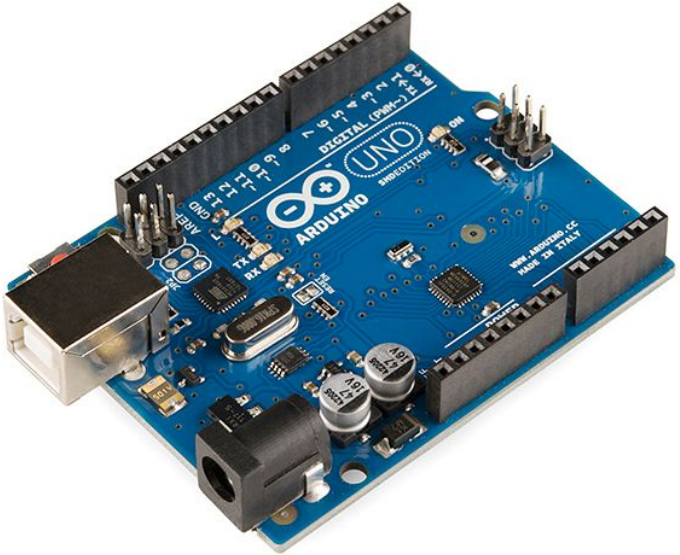

SAMSUNG
SAMSUNG
Galaxy Note7


SAMSUNG

# Diagnostic ultrasound imaging system

- Ultrasound system
  - Computing unit - ASIC, FPGA, DSP
  - Storage - RAMs and ROMs
  - User Interface - Ultrasound transducer
- Industrial embedded computer
  - Computing unit - CPU, GPU
  - Storage - RAMs, ROMs, HDD and SSD
  - Communication - ethernet, WIFI, USB
  - User Interface -  Key panel, Keyboard and trackball

# Examples


DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
POWER
ANALOG IN
WWW.ARDUINO.CC
MADE IN ITALY

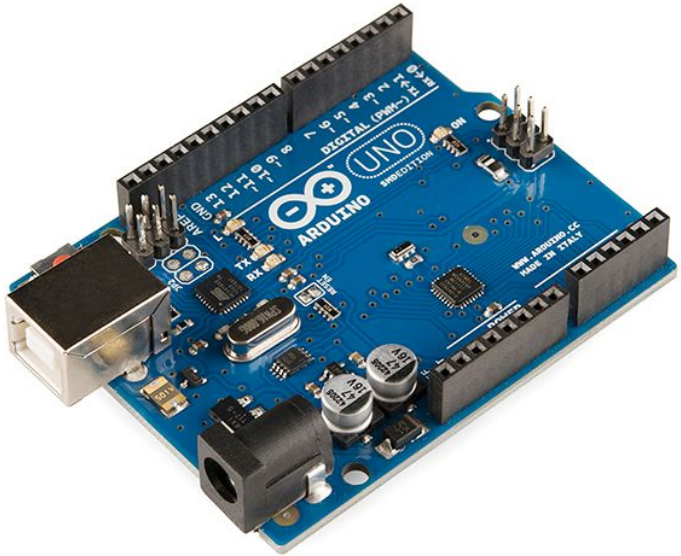

EDAC

# Raspberry Pi

- https://www.raspberrypi.org/
- Credit card-sized single-board computers
- Promoting the teaching of basic computer science in schools and developing countries
- Pi 1 released in Feb 2012.
  Pi 3 released in Feb 2016.
- Raspbian, Ubuntu, Windows 10 IOT
- Python and Scratch
- 8 million in 2016
- https://github.com/raspberrypi

# Arduino

- https://www.arduino.cc/
- open-source electronics platform based on easy-to-use hardware and software.
- 700,000 official boards  in 2013.
- https://github.com/arduino
- C / C++



# Goal of this course

Essential skills for embedded computing

- Essential to know if you eventually want to work on embedded systems.
- Extremely useful for any embedded computing project, even on a laptop.

Strategy

- Concentrate on basics, simple motivating examples.
- Focusing hands-on experience.
- Learn what’s out there to help select what’s best for your needs.

# Focus and Topics

Efficiently using your computing units

- Basic computer architecture
- Languages issues, e.g. compiled vs. interpreted, object oriented, etc.
- Specific languages: C
- Parallel computing with OpenMP

Efficient programming and good software practices

- Version control system : **Git** and **GitHub**
- **C** with visual studio
- Debuggers, code development and testing

# Class materials

- All class materials are on GitHub repository.
  - Recommend having a GitHub account.
- You can clone the repository or download from webpage
  - https://github.com/CSE6000/Fall2016

# Prerequisites

Some programming experience in some language,
e.g., Python, Matlab, C/C++, Java. Swift, C#

You should be comfortable:

- editing a file containing a program and executing it,
- using basic structures like loops, if-then-else, input-output
- writing subroutines or functions in some language

You are not expected to know C

Some basic knowledge of linear algebra - vector or matrices addition, multiplication, solving a linear system

Some comfort level for learning new software and willingness to dive into lots of new things

# Requirements and recommendations

- Requirements
  - Git
  - Any C compiler
  - GitHub account
- Recommendations
  - Microsoft Windows 10
  - MS Visual Studio 2015 Community
  - Git for Windows

Note : Linux is often required for embedded computing, but learning linux is not part of this class