CE5045
Embedded System Design

I. Introduction to Course

Instructor: Dr. Chen, Tseng-Yi

Computer Science & Information Engineering

Course Information (I/II)

- This course is for graduate students who are interesting in embedded system kernels, I/O subsystem development, and embedded component integration.

- Topics of this course cover
  - ✓ Understanding the Linux O.S. kernel.
    - Process, memory management, file system, etc.
  - ✓ Developing I/O drivers and integrating the drivers to Linux kernel.
    - Linux-based mobile system (e.g., Android development)
  - ✓ Establishing embedded application on Arduino platform
    - Internet-of-things applications (e.g., RFID module integration)

Course Information (II/II)

- The objectives of this course
  - Enable students to learn the hand-on experience in Linux kernel modification, I/O driver development, and internet-of-things platform establishment.
- Teaching materials
  - ✓ No required textbook for now.
  - ✓ All slides can be downloaded on LMS system.
  - ✓ Reference books for this course
    - Daniel P. Bovet and Marco Cesati, Understanding The Linux Kernel 3rd Edition
    - Wolfgang Mauerer, Professional Linux Kernel Architecture, 1st Edition
    - Edward A. Lee and SanjitA. Seshia: Introduction to Embedded Systems, A Cyber-Physical Systems Approach, Second Edition
  - ✓ Hardware platform for labs
    - Arduino UNO R3 and TI MSP 430

Evaluation

- Grading
  - ✓ Attendance 10% (Class participation)
  - ✓ Assignment 20% (3-5 labs)
    - All assignment will be announced before June 12 and their deadline will be at the end of June
  - ✓ Midterm report 30%
    - Trace Linux kernel and give an oral presentation
  - ✓ Final project 40%
    - Build an IoT services or applications based on Arduino or MSP430 platforms.
- Notice
  - ✓ You may fail this course if you miss (more than) ¼ of the whole classes
  - ✓ Academic dishonesty (e.g. cheating, plagiarism, and etc.) will be taken seriously, and heavy penalty can be imposed.

About This Class

- Class information
  - ✓ Class schedule: Wednesday 11:00~11:50 and Thursday 15:00~16:50
  - ✓ Classroom: A205 (Wednesday) and A203 (Thursday)
  - ✓ All handouts will be in English

- Instructor information
  - ✓ Name: Tseng-Yi Chen (陳增益)
  - ✓ Office: E6-B535
  - ✓ E-mail: tychen@g.ncu.edu.tw
  - ✓ Office tel.:35334
  - ✓ Office hours: Mon. 13:00~17:00 and Fri. 11:00~13:00

Course Progress

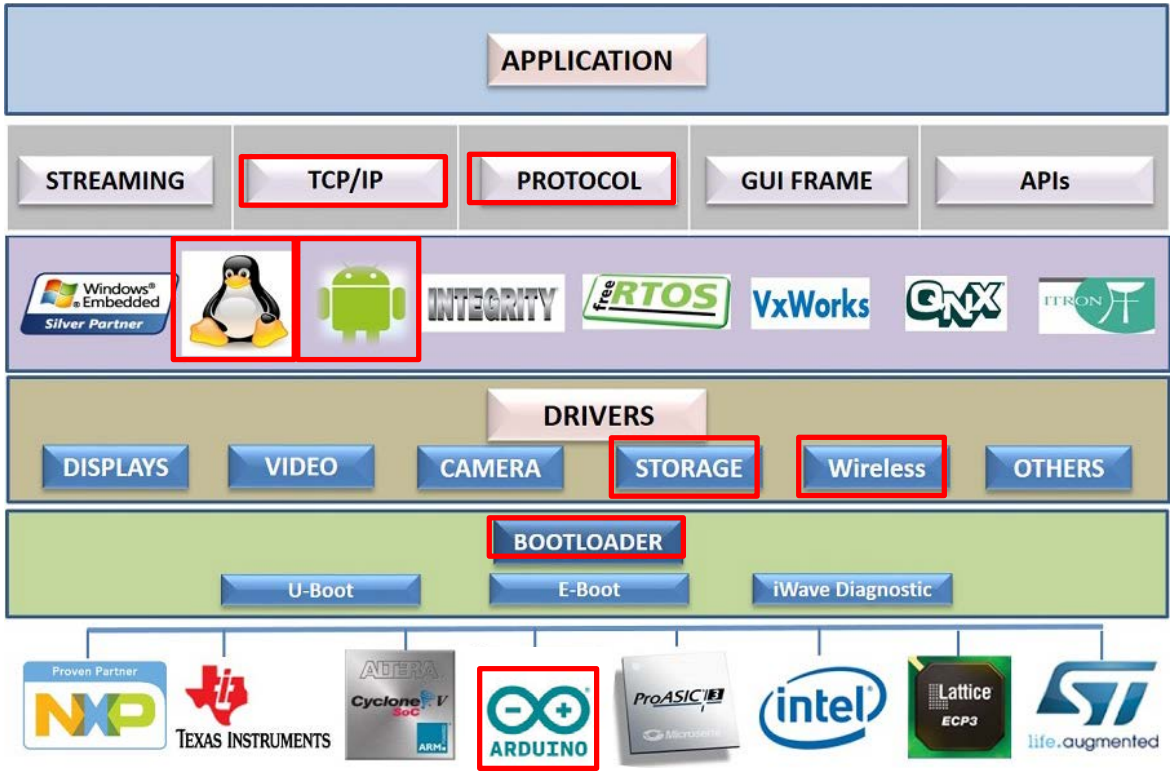
Date	Progress
3/4-5	Introduction to Course
3/11-12	Embedded O.S. Intro.
3/18-19	Process Management
3/25-26	Process Scheduling
4/1-2	4/1 Memory Management(I) 4/2 Spring break
4/8-9	Memory Management(II)
4/15-16	Signal and Interruption
4/22-23	Midterm report

Date	Progress
4/29-30	I/O Driver Development
5/6-7	Cross-compiler Introduction
5/13-14	Android O.S. Introduction
5/20-21	Android I/O and Native Lib.
5/27-28	Arduino Platform Intro.
6/3-4	Arduino Development Tool
6/10-11	Arduino I/O module Dev.
6/17-18	6/18 Final project demo
6/24-25	6/24 Final project demo 6/25 Dragon Boat Festival

Remarks

- This course assumes that students have basic knowledge of Operating Systems and C/C++ programming language.
- However, this course does not assume that students are familiar with embedded operating system development.

Overview of System Architecture



- Embedded operating system
  - ✓ Embedded Linux
  - ✓ Android O.S.
- Drivers
  - ✓ Storage device
  - ✓ Wireless module
- Hardware platform
  - ✓ Arduino UNO R3

CE5045
Embedded System Design

II. Introduction to Embedded System

Instructor: Dr. Chen, Tseng-Yi

Computer Science & Information Engineering

What is Embedded System?

- Computing systems are everywhere.

- Most of us think of “desktop” computers
  - ✓ PC’s
  - ✓ Laptops
  - ✓ Mainframes
  - ✓ Servers

- But there’s another type of computing system
  - ✓ Far more common...

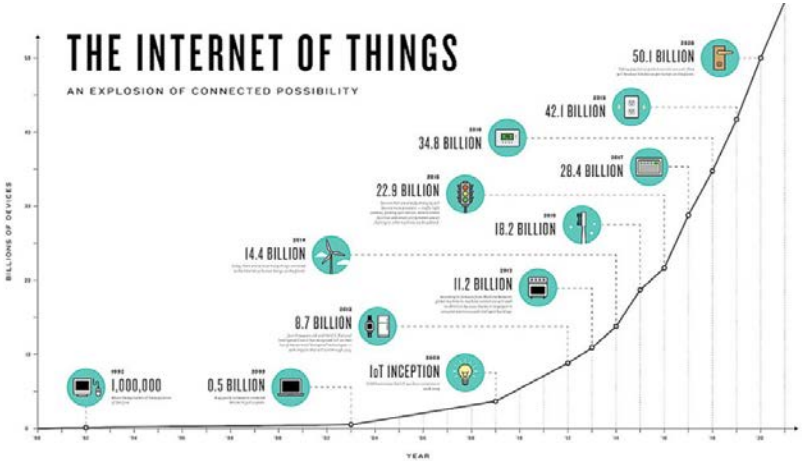
Embedded System Everywhere

- Embedded computing systems
  - ✓ Computing systems embedded within electronic devices
  - ✓ Hard to define. Nearly any computing system other than a desktop computer
  - ✓ Billions of units produced yearly, versus millions of desktop units

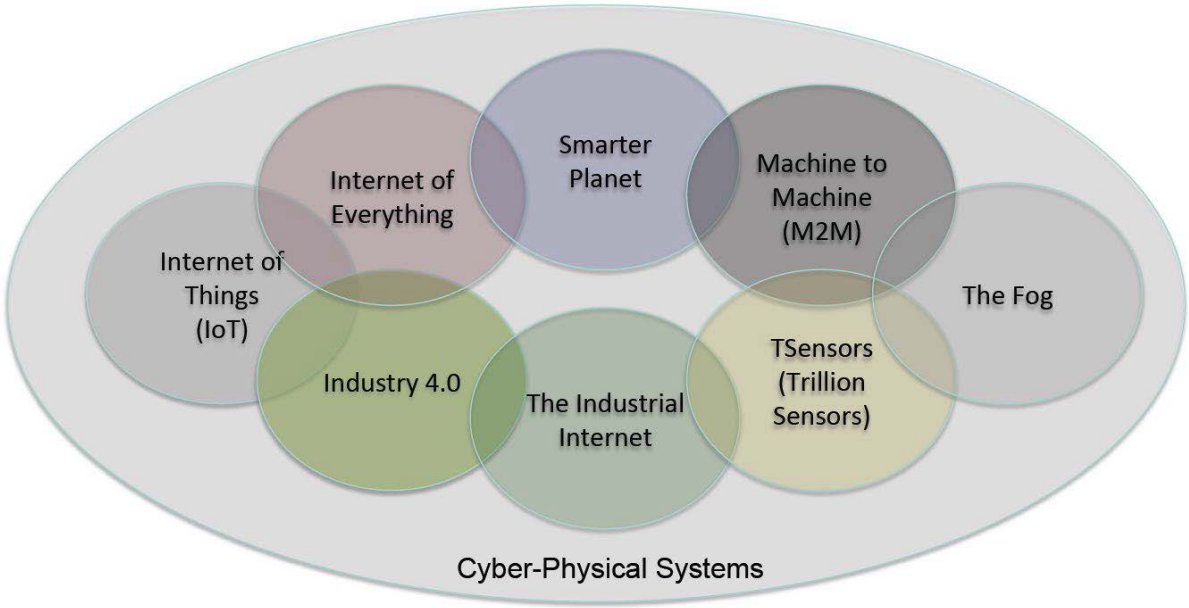
Computers are in here...

and here...

and even here...



Many Names – Similar Meanings



A “short list” of Embedded Systems

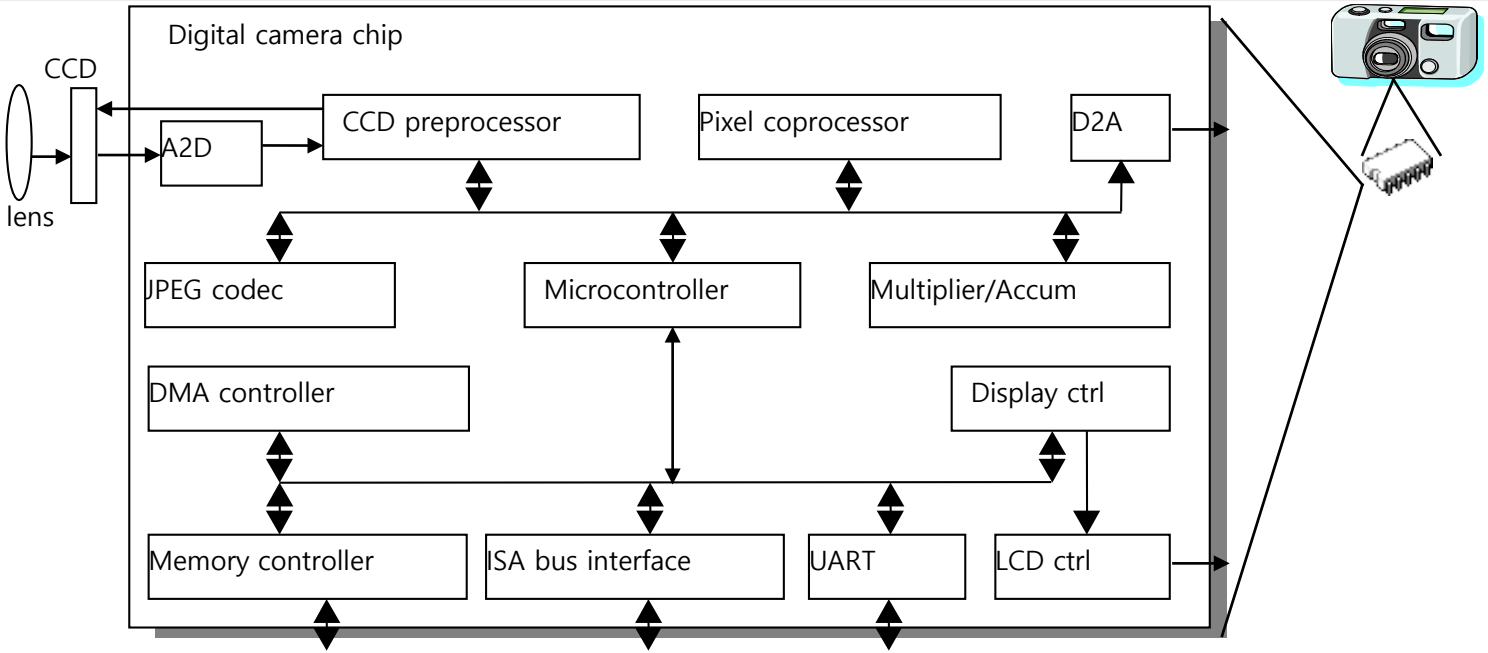
Anti-lock brakes
Auto-focus cameras
Automatic teller machines
Automatic toll systems
Automatic transmission
Avionic systems
Battery chargers
Camcorders
Cell phones
Cell-phone base stations
Cordless phones
Cruise control
Curbside check-in systems
Digital cameras
Disk drives
Electronic card readers
Electronic instruments
Electronic toys/games
Factory control
Fax machines
Fingerprint identifiers
Home security systems
Life-support systems
Medical testing systems
Modems
MPEG decoders
Network cards
Network switches/routers
On-board navigation
Pagers
Photocopiers
Point-of-sale systems
Portable video games
Printers
Satellite phones
Scanners
Smart ovens/dishwashers
Speech recognizers
Stereo systems
Teleconferencing systems
Televisions
Temperature controllers
Theft tracking systems
TV set-top boxes
VCR’s, DVD players
Video game consoles
Video phones
Washers and dryers

Embedded system never died.

Characteristics of Embedded Systems

- Single-functioned
  - ✓ Executes a single program, repeatedly
- Tightly-constrained
  - ✓ Low cost, low power, small, fast, etc.
- Reactive and real-time
  - ✓ Continually reacts to changes in the system's environment
  - ✓ Must compute certain results in real-time without delay

An Embedded System Example



- Single-functioned – always a digital camera
- Tightly-constrained – Low cost, low power, small, fast
- Reactive and real-time – only to a small extent

Quick Question

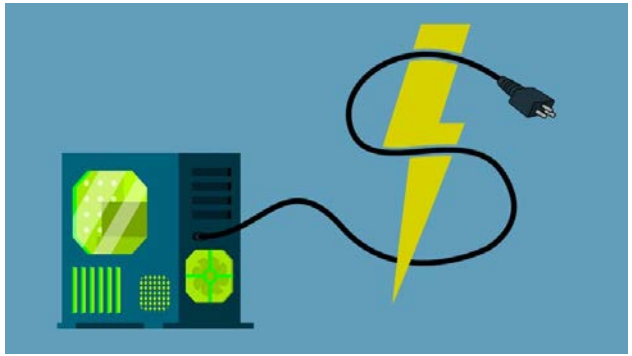
- Can smartphone be considered as an embedded platform?

Quick Question

- It is a complete system. However, smartphones do contain several embedded systems, like the modem core and the single-chip WiFi+BT+GPS solutions.

Design Challenge

- Obvious design goal:
  - ✓ Construct an implementation with desired functionality
- Key design challenge:
  - ✓ Simultaneously optimize numerous design metrics
- Design metric
  - ✓ A measurable feature of a system's implementation
  - ✓ Optimizing design metrics is a key challenge



Energy consumption? Performance?

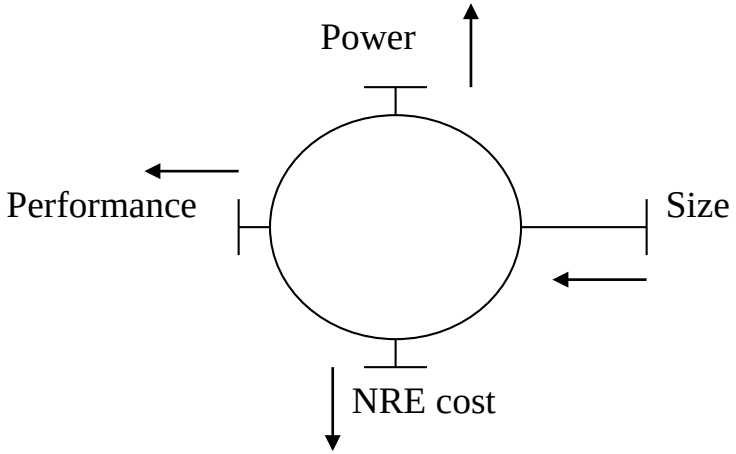
Design Metrics (I/II)

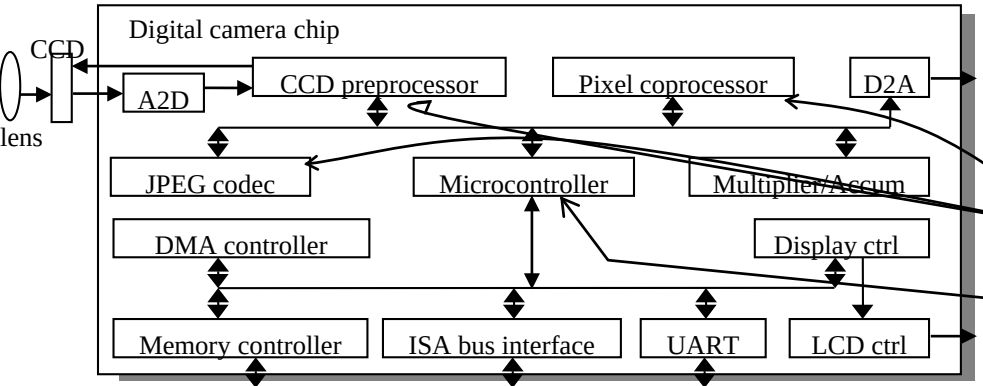
- Common metrics
  - ✓ Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  - ✓ NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
  - ✓ Size: the physical space required by the system
  - ✓ Performance: the execution time or throughput of the system
  - ✓ Power: the amount of power consumed by the system
  - ✓ Flexibility: the ability to change the functionality of the system without incurring heavy NRE cost

Design Metrics (II/II)

- Common metrics
  - ✓ Time-to-prototype: the time needed to build a working version of the system
  - ✓ Time-to-market: the time required to develop a system to the point that it can be released and sold to customers
  - ✓ Maintainability: the ability to modify the system after its initial release
  - ✓ Correctness, safety, many more

Improving One May Worsen Others



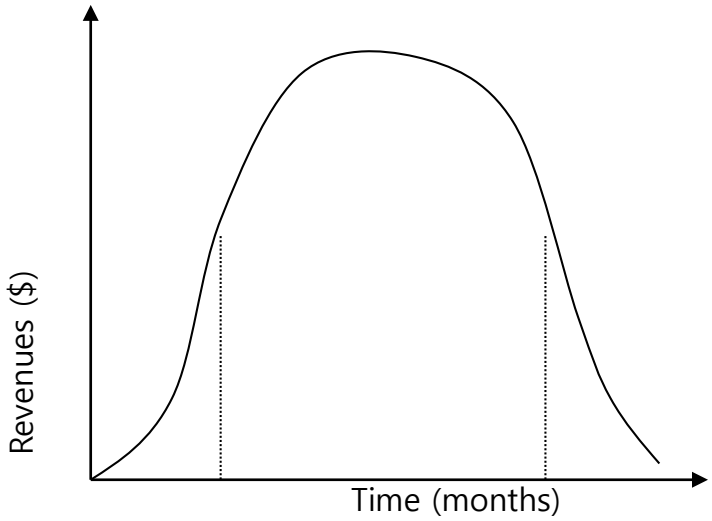


- Expertise with both **software and hardware** is needed to optimize design metrics
  - ✓ Not just a hardware or software expert, as is common
  - ✓ A designer must be comfortable with various technologies in order to choose the best for a given application and constraints

Hardware

Software

Time-to-market



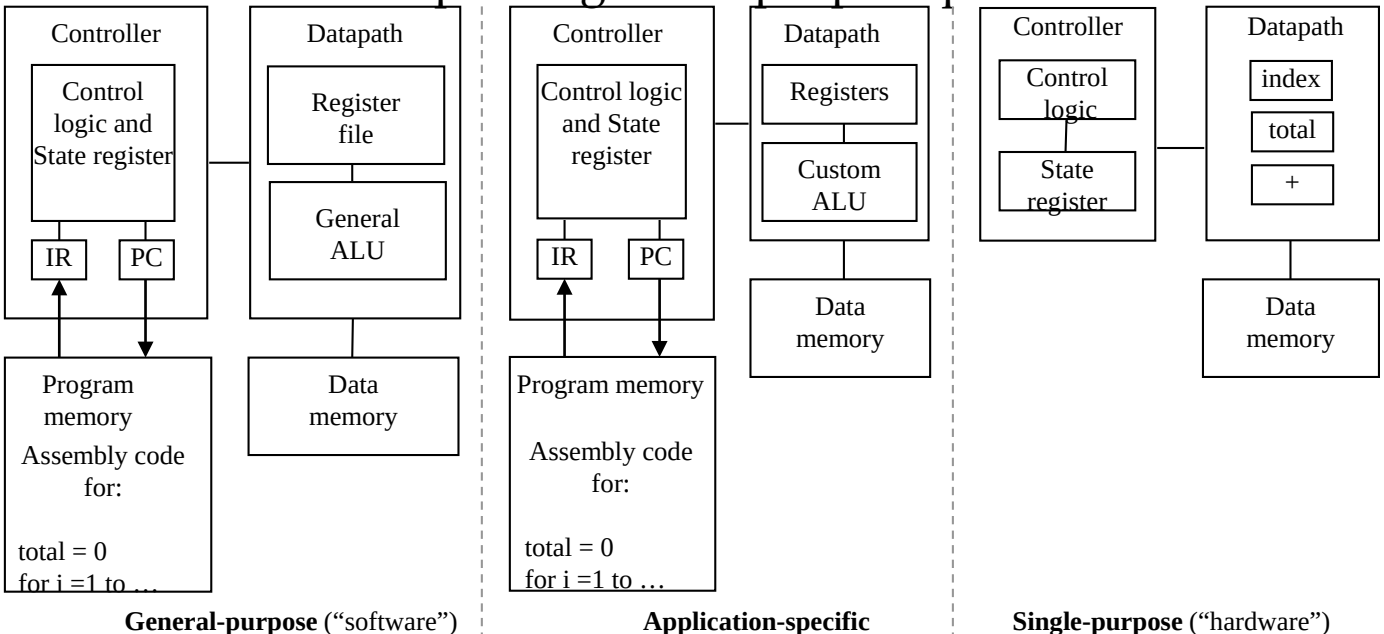
- Time required to develop a product to the point it can be sold to customers
- Market window
  - ✓ Period during which the product would have highest sales
- Average time-to-market constraint is about 8 months
- Delays can be costly

Performance

- Widely-used measure of system, widely-abused
  - ✓ Clock frequency, instructions per second – not good measures for customer
  - ✓ Digital camera example – a user cares about how fast it processes images, not clock speed or instructions per second
- Latency (different from response time)
  - ✓ Time between task start and end
  - ✓ e.g., Camera's A and B process images in 0.25 seconds
- Throughput
  - ✓ Tasks per second, e.g. Camera A processes 4 images per second
  - ✓ Throughput can be more than latency seems to imply due to concurrency, e.g. Camera B may process 8 images per second (by capturing a new image while previous image is being stored).
- *Speedup* of B over S = B's performance / A's performance
  - Throughput speedup = 8/4 = 2

Processor Technology (I/II)

- The architecture of the computation engine used to implement a system's desired functionality
- Processor does not have to be programmable
  - ✓ "Processor" *not* equal to general-purpose processor

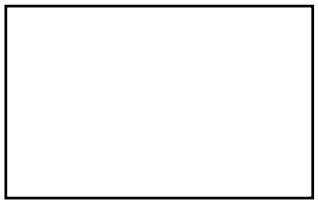


Processor Technology (II/II)

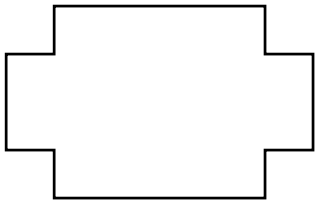
- Processors vary in their customization for the problem at hand

Desired functionality

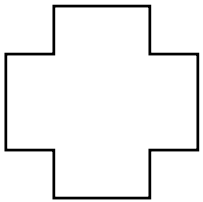
```
total = 0
for i = 1 to N  loop
   total += M[i]
end loop
```



General-purpose processor



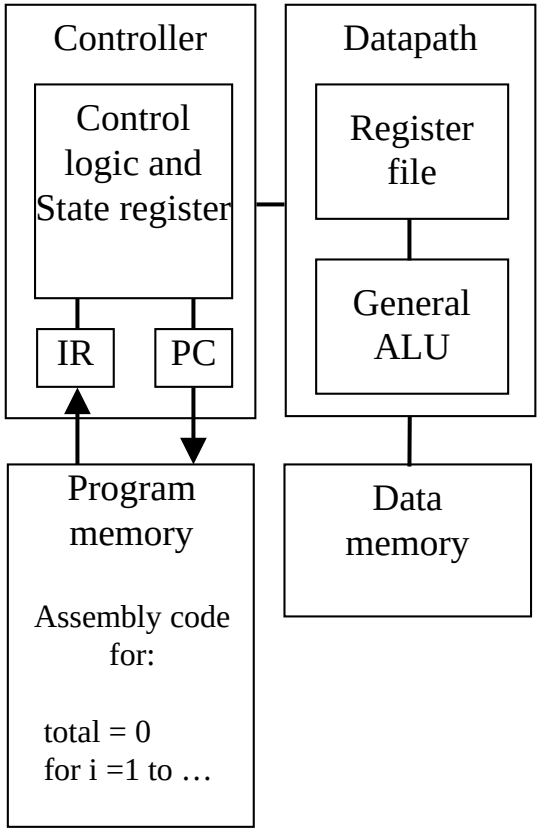
Application-specific processor



Single-purpose processor

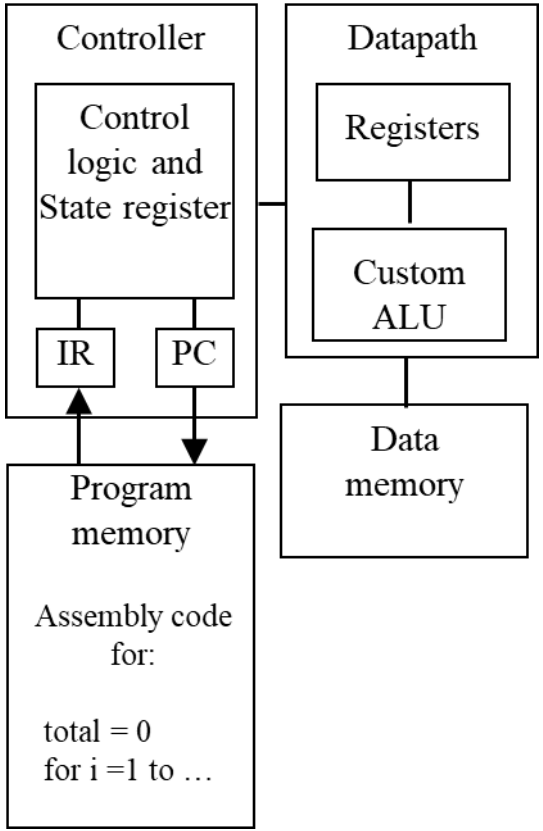
General-purpose Processors

- Programmable device used in a variety of applications
  - ✓ Also known as “microprocessor”
- Features
  - ✓ Program memory
  - ✓ General datapath with large register file and general ALU
- User benefits
  - ✓ Low time-to-market and NRE costs
  - ✓ High flexibility
- “Intel i-series” the most well-known, but there are hundreds of others



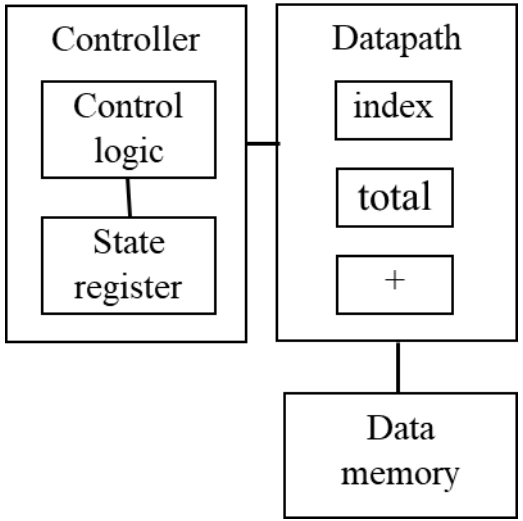
Application-specific processors

- Programmable processor optimized for a particular class of applications having common characteristics
  - ✓ Compromise between general-purpose and single-purpose processors
- Features
  - ✓ Program memory
  - ✓ Optimized datapath
  - ✓ Special functional units
- Benefits
  - ✓ Some flexibility, good performance, size and power



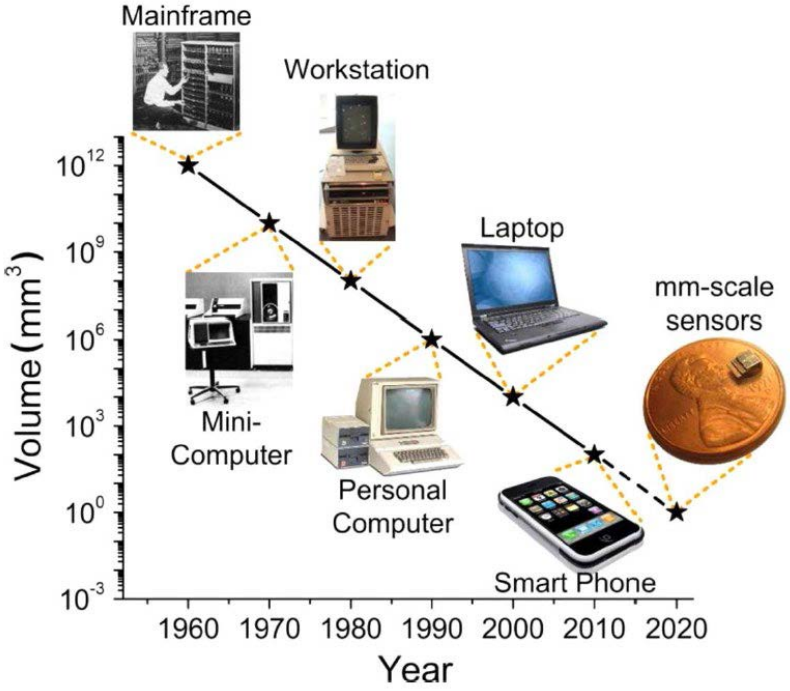
Single-purpose processors

- Digital circuit designed to execute exactly one program
  - ✓ a.k.a. coprocessor, accelerator or peripheral
- Features
  - ✓ Contains only the components needed to execute a single program
  - ✓ No program memory
- Benefits
  - ✓ Fast
  - ✓ Low power
  - ✓ Small size



Bell's Law

- New class of Computers Emerges Every 10 Years



Streaming information to/from the physical world:

- Smart Dust
- Sensor Networks
- Cyber-Physical Systems
- Internet-of-Things

Ever cheaper, ever smaller, ever more networked, tighter integrated edge devices

Trends

- Embedded systems are communicating with each other, with servers or with the cloud. Communication is increasingly wireless.

- **Higher degree of integration** on a single chip or integrated components
  - ✓ Memory + processor + I/O-units + (wireless) communication
  - ✓ Use of networks-on-chip for communication between units
  - ✓ Use of homogeneous or heterogeneous multiprocessor systems on a chip (MPSoC).
  - ✓ Use of integrated microsystems that contain energy harvesting, energy storage, sensing, processing and communication ("zero power systems")
  - ✓ The complexity and amount of software is increasing

- **Low power and energy constraints** (portable or unattended devices) are increasingly important, as well as temperature constraints (overheating)