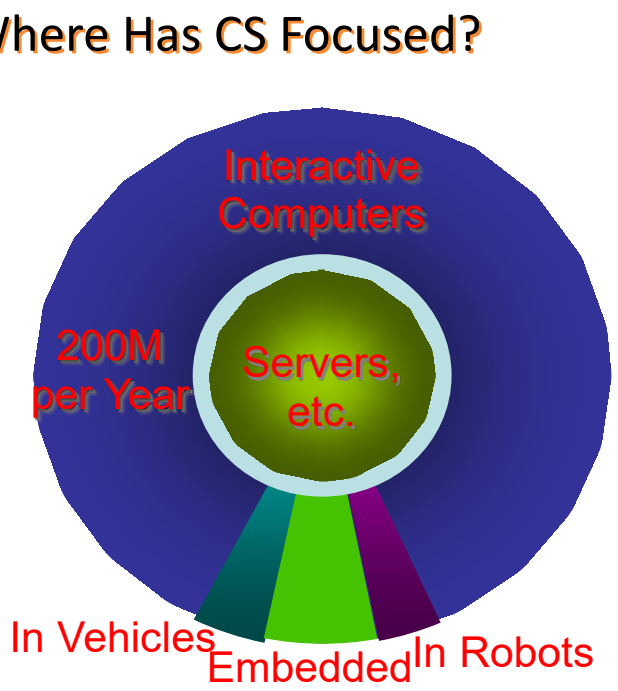
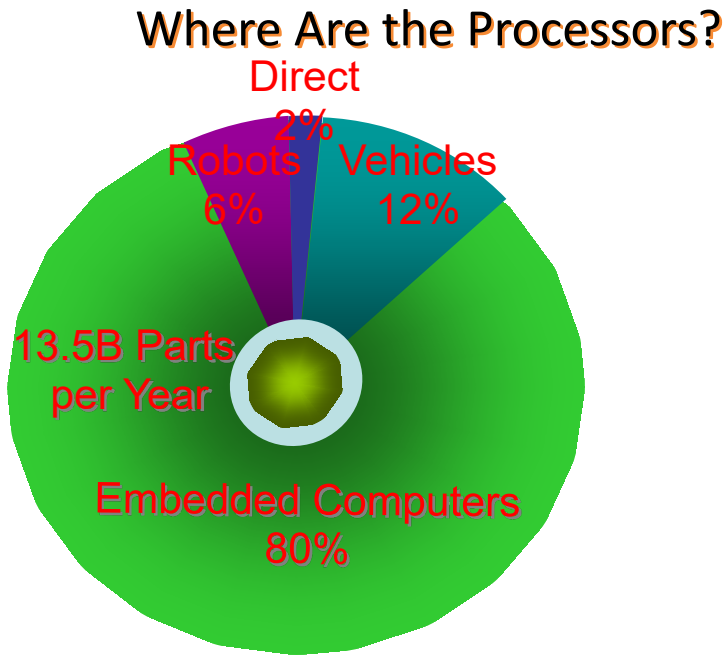
Introduction to Embedded Systems

Where are the Processors?

Estimated 98% of 8 Billion CPUs produced annualy used for embedded apps



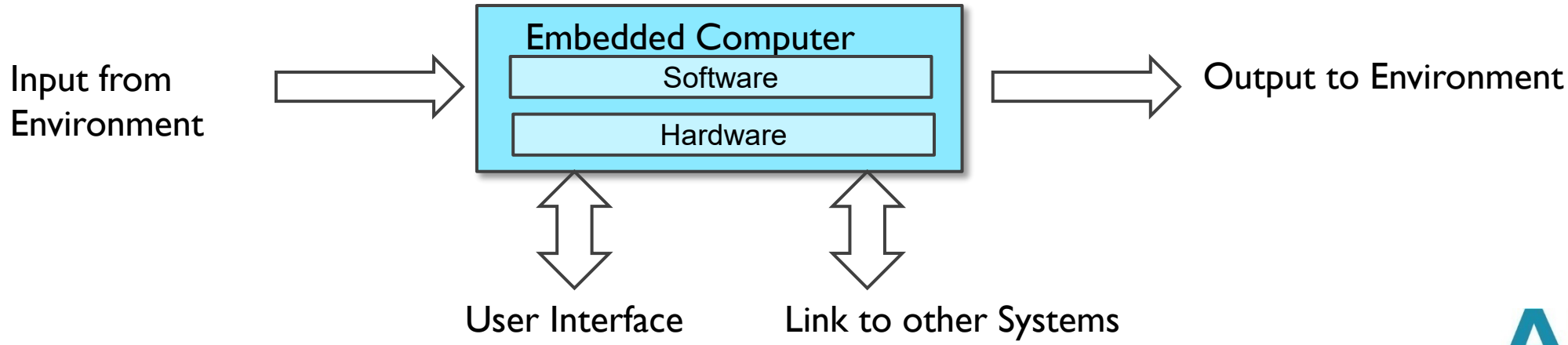


Look for the Processors…the Opportunities Will Follow!

Source: DARPA/Intel (Tennenhouse)

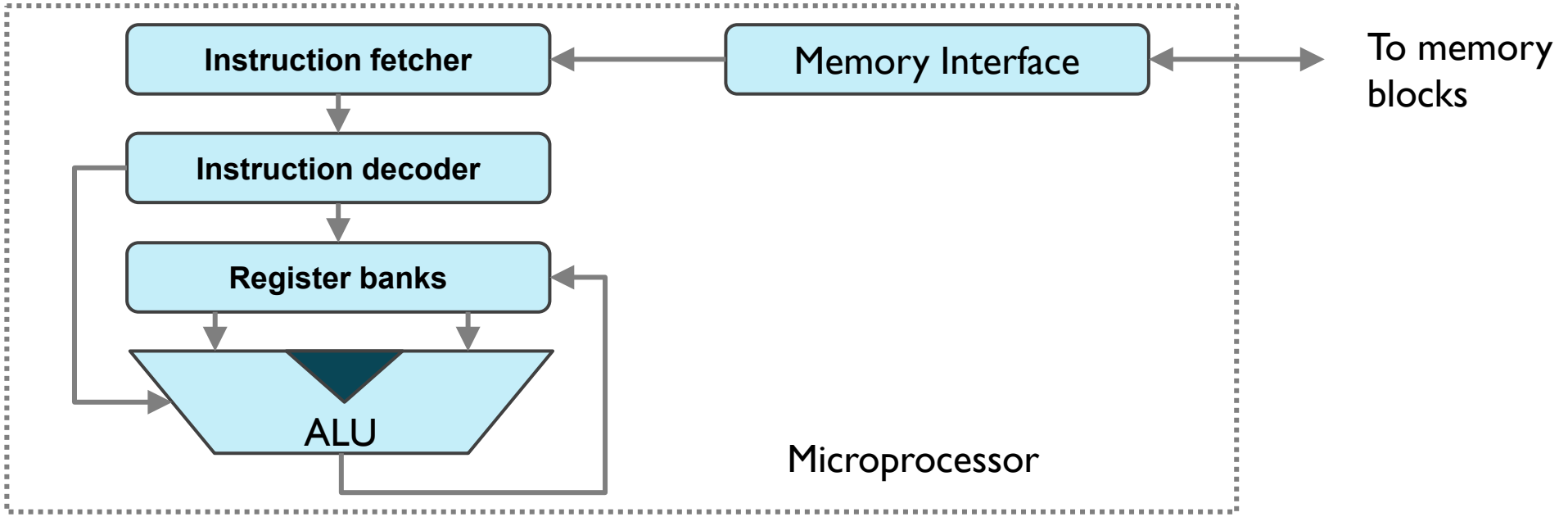
Introduction to Embedded Systems

- What is an Embedded System?
  - Application-specific computer system
  - Built into a larger system
  - Often with real-time computing constraints
- Why add a computer to a larger system?
  - Better performance
  - More functions and features
  - Lower cost e.g. through automation
  - More dependability



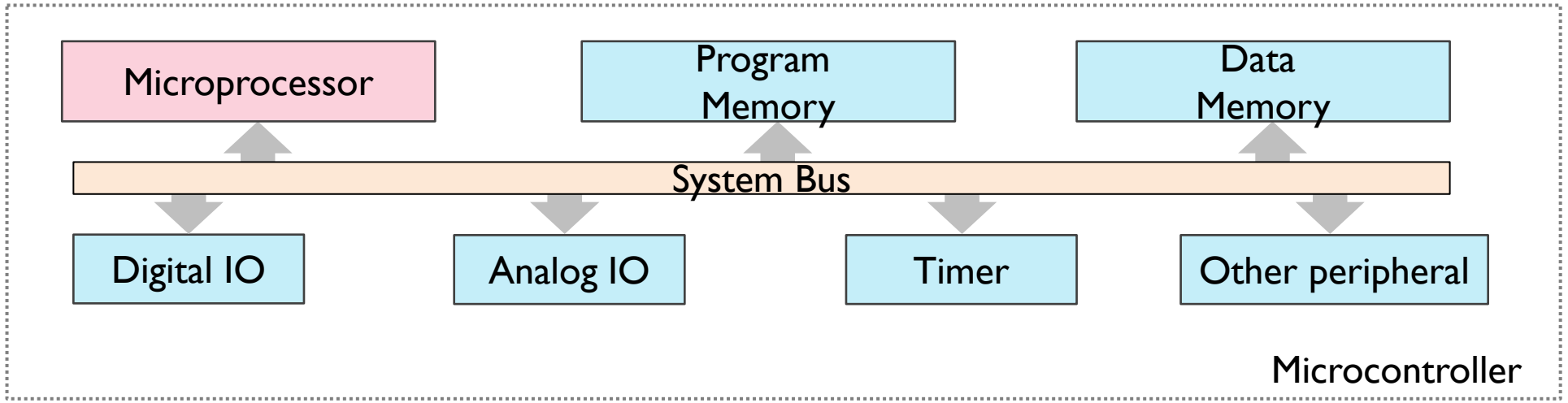
CPUs vs. MCUs vs. Embedded Systems

- Microprocessor (CPU)
  - Defined typically as a single processor core that supports at least instruction fetching, decoding, and executing
  - Normally can be used for general purpose computing, but needs to be supported with memories and Input/Outputs(IOs)



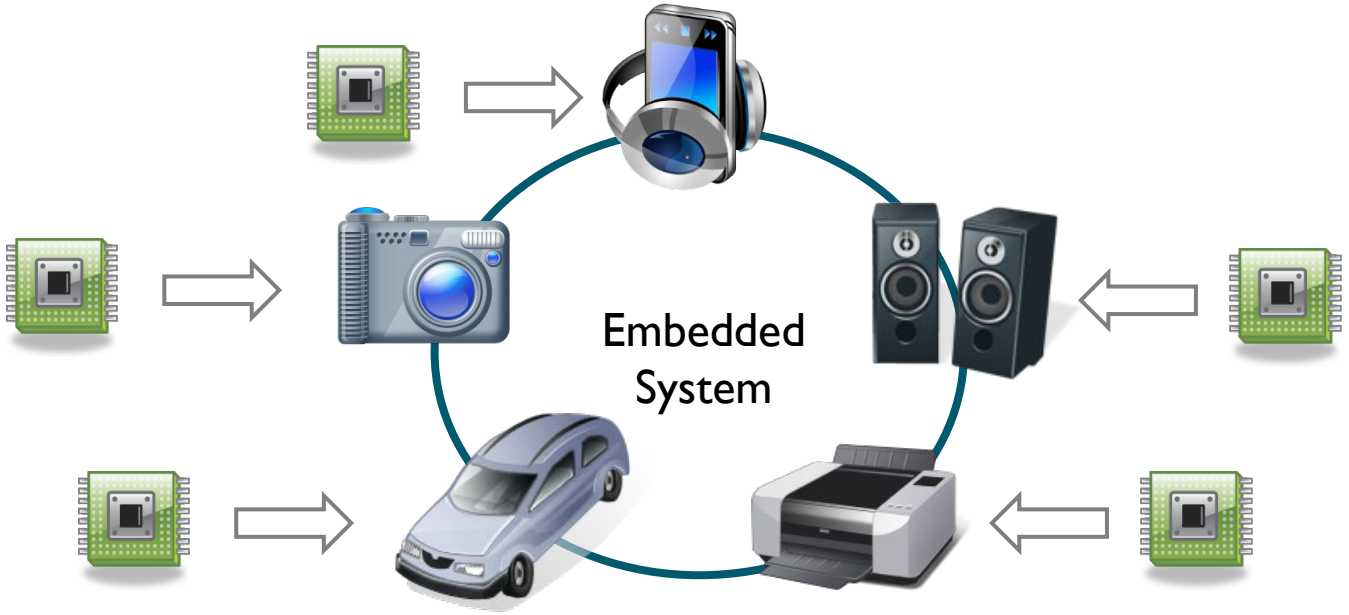
CPUs vs. MCUs vs. Embedded Systems

- Microcontroller (MCU)
  - Typically has a single processor core
  - Has memory blocks, Digital IOs, Analog IOs, and other basic peripherals
  - Typically used for basic control purpose, such as embedded applications



CPUs vs. MCUs vs. Embedded Systems

- Embedded System
  - Typically implemented using MCUs
  - Often integrated into a larger mechanical or electrical system
  - Usually has real-time constraints



Attributes of Embedded Systems

- Interfacing with larger system and environment
  - Analog signals for reading sensors
  - Typically use a voltage to represent a physical value
  - Power electronics for driving motors, solenoids
  - Digital interfaces for communicating with other digital devices
  - Simple - switches
  - Complex – displays
- Concurrent, reactive behaviours
  - Must respond to sequences and combinations of events
  - Real-time systems have deadlines on responses
  - Typically must perform multiple separate activities concurrently

Attributes of Embedded Systems

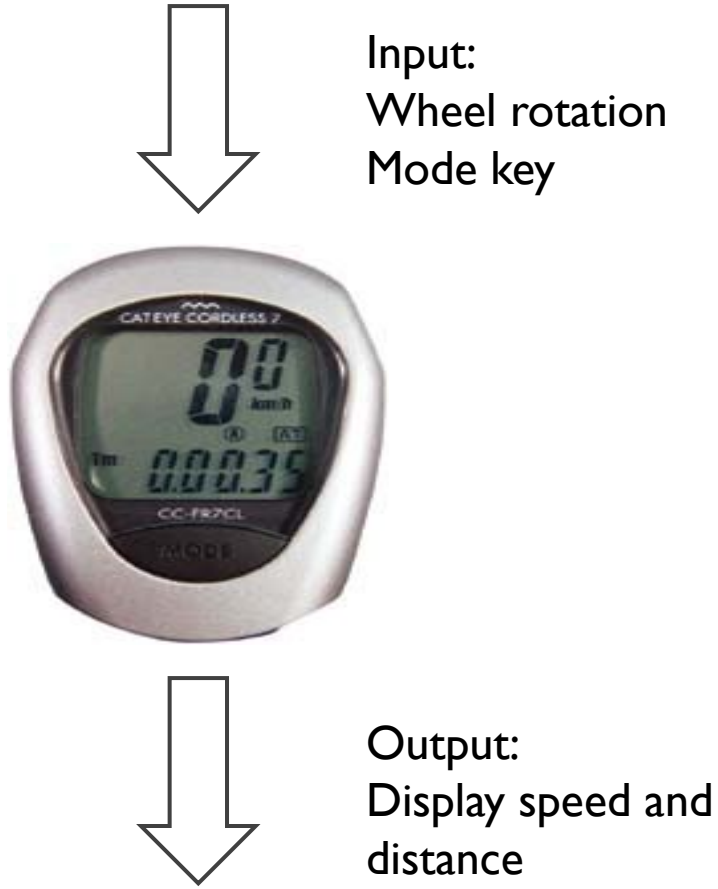
- Fault handling
  - Many systems must operate independently for long periods of time, requiring them to handle likely faults without crashing
  - Often fault-handling code is larger and more complex than the normal-case code
- Diagnostics
  - Help service personnel determine problems quickly

Functions of Embedded Systems

- Closed-loop control system
  - Monitor a process, adjust an output to maintain desired set point (temperature, speed, direction, etc.)
- Sequencing
  - Step through different stages based on environment and system
- Signal processing
  - Remove noise, select desired signal features
- Communications and networking
  - Exchange information reliably and quickly

Example Embedded System: Bike Computer

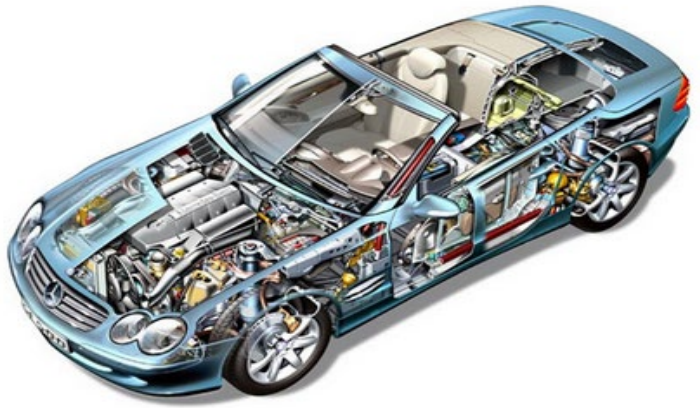
- Functions
  - Speed and distance measurement
- Constraints
  - Size
  - Cost
  - Power and Energy
  - Weight
- Inputs
  - Wheel rotation indicator
  - Mode key
- Output
  - Liquid Crystal Display
- Use Low Performance Microcontroller
  - 8-bit, 10 MIPS



Gasoline Automobile Engine Control Unit

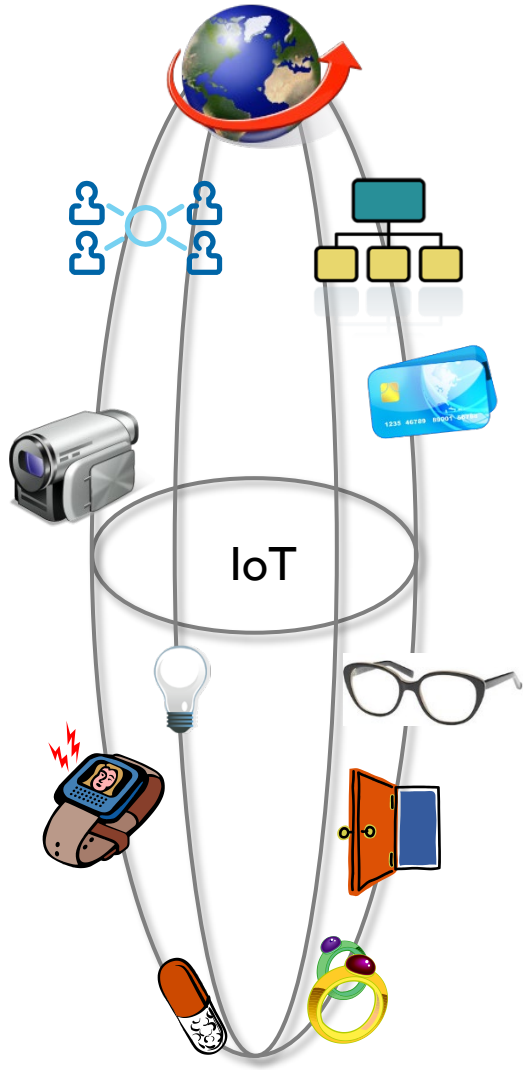
- Functions
  - Fuel injection
  - Air intake setting
  - Spark timing
  - Exhaust gas circulation
  - Electronic throttle control
  - Knock control
- Constraints
  - Reliability in harsh environment
  - Cost
  - Weight

- Many Inputs and Outputs
  - Discrete sensors & actuators
  - Network interface to rest of car
- Use High Performance Microcontroller
  - E.g. 32-bit, 3 MB flash memory, 150 - 300 MHz



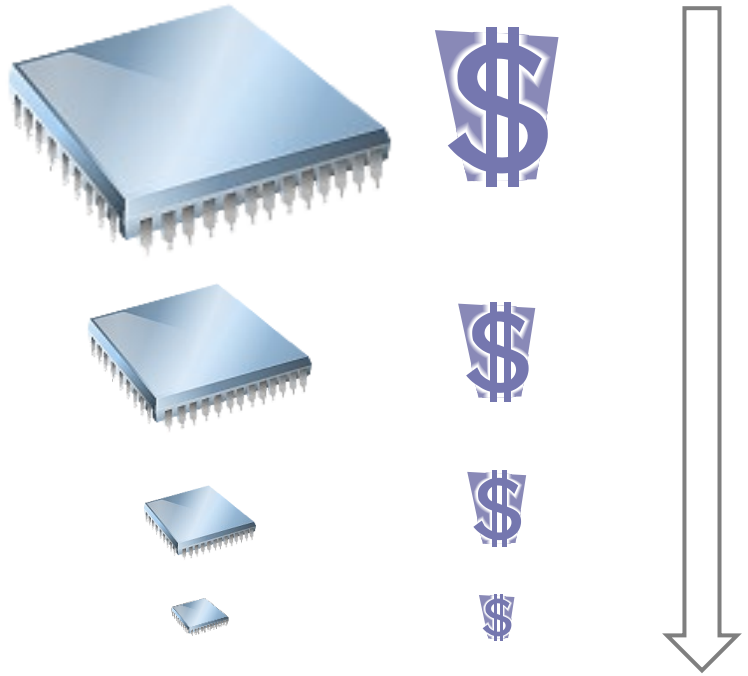
Internet of Things

- Internet of Things (IoT)
  - IoT as a term generally refers to a world in which a large range of objects are addressable via the network
  - Objects can include
  - Smart buildings and home appliances, e.g. washing machines, TVs, fridges, cookers, doors, chairs…
  - Civil engineering structures, e.g. bridges, railways …
  - Wearable devices, e.g. smart watches, smart glasses, rings, clothes …
  - Medical devices, e.g. embedded pills
  - And possibly every THING in the world…



Internet of Things

- Why IoT?
  - Items can have more functionalities and become more intelligent
  - Items can be managed in an easier way
  - More information become available
- Why IoT is becoming more realistic?
  - Embedded chips are becoming
  - Cheaper
  - Smaller
  - Lower power
  - Communication is becoming faster



Challenges of Internet of Things

- Large amount of chips required
  - Chips have to become even more cheaper, smaller
- Big data demand
  - Large volume of data will be generated, data centre storage needs to be increased
- Computation requirement
  - Requires high performance e.g. for cloud computing
- Power consumption
  - Low power chips, longer battery life, and maybe wireless charging…
- Security
  - Large amount of private data need to be protected
- Standards
  - Official standards are required, such as network protocol

Options for Building Embedded Systems

Dedicated Hardware

Software Running on Generic Hardware

Implementation	Design Cost	Unit Cost	Upgrades & Bug Fixes	Size	Weight	Power	System Speed
Discrete Logic	low	mid	hard	large	high	?	very fast
ASIC	high ($500K/ mask set)	very low	hard	tiny - 1 die	very low	low	extremely fast
Programmable logic – FPGA, PLD	low to mid	mid	easy	small	low	medium to high	very fast
Microprocessor + memory + peripherals	low to mid	mid	easy	small to med.	low to moderate	medium	moderate
Microcontroller (int. memory & peripherals)	low	mid to low	easy	small	low	medium	slow to moderate
Embedded PC	low	high	easy	medium	moderate to high	medium to high	fast

Benefits of Embedded Systems

- Greater performance and efficiency
  - Software makes it possible to provide sophisticated control
- Lower costs
  - Less expensive components can be used
  - Manufacturing costs reduced
  - Operating costs reduced
  - Maintenance costs reduced
- More features
  - Many not possible or practical with other approaches
- Better dependability
  - Adaptive system which can compensate for failures
  - Better diagnostics to improve repair time

Constraints of Embedded Systems

- Cost
  - Competitive markets penalize products which don't deliver adequate value for the cost
- Size and weight limits
  - Mobile (aviation, automotive) and portable (e.g. handheld) systems
- Power and energy limits
  - Battery capacity
  - Cooling limits
- Environment
  - Temperatures may range from -40°C to 125°C, or even more

Impact of Constraints

- Microcontrollers used (rather than microprocessors)
  - Include peripherals to interface with other devices, respond efficiently
  - On-chip RAM, ROM reduce circuit board complexity and cost
- Programming language
  - Programmed in the C language rather than the Java language (resulting in smaller and faster code, so less expensive MCU)
  - Some performance-critical code may be in assembly language (a lower level language)
- Operating system
  - Typically no OS, but instead simple scheduler (or even just interrupts + main code (foreground/background system)
  - If OS is used, likely to be a lean RTOS

Building Embedded Systems using MCUs

- In most embedded systems, MCUs are chosen to be the best solution, since they offer:
  - Low development and manufacturing cost
  - Easy porting and updating
  - Light footprint
  - Relatively low power consumption
  - Satisfactory performance for low-end products
- We will learn how to develop a variety of embedded systems, using an easy-to-start MCU design suite: mbed™ platform
  - Open software library tools
  - Low cost hardware platforms
  - Online Integrated development environment (IDE)

What is mbed Platform

- mbed is a platform used for developing applications based on ARM Cortex-M microprocessors
- The mbed platform includes:
  - mbed Software Development Kit (SDK), consists of
  - C/C++ software libraries, such as peripheral drivers, networking, RTOS and runtime environment
  - Software tools, such as build tools, test and debug scripts
  - mbed Hardware Development Kit (HDK), consist of
  - Recipes to build custom hardware devices, such as interface firmware and schematics that can be used to easily create development boards
  - mbed hardware platforms – off-the-shelf development boards
  - mbed supports an online IDE, which provides a free instant-access web-based toolchain for application development

Useful Resources

- mbed official website:
- http://www.mbed.org

Why Are We:

- Using C instead of Java (or Python, or your other favorite language)?
  - C is the de facto standard for embedded systems because of:
  - Precise control over what the processor is doing
  - Modest requirements for ROM, RAM, and MIPS, so much cheaper system
  - Predictable behavior, no OS (e.g., garbage collection) preemption
- Learning assembly language?
  - The compiler translates C into assembly language. To understand whether the compiler is doing a reasonable job, you need to understand what it has produced.
  - Sometimes, we may need to improve performance by writing assembly versions of functions.
- Required to have a microcontroller board?
  - The best way to learn is hands-on.