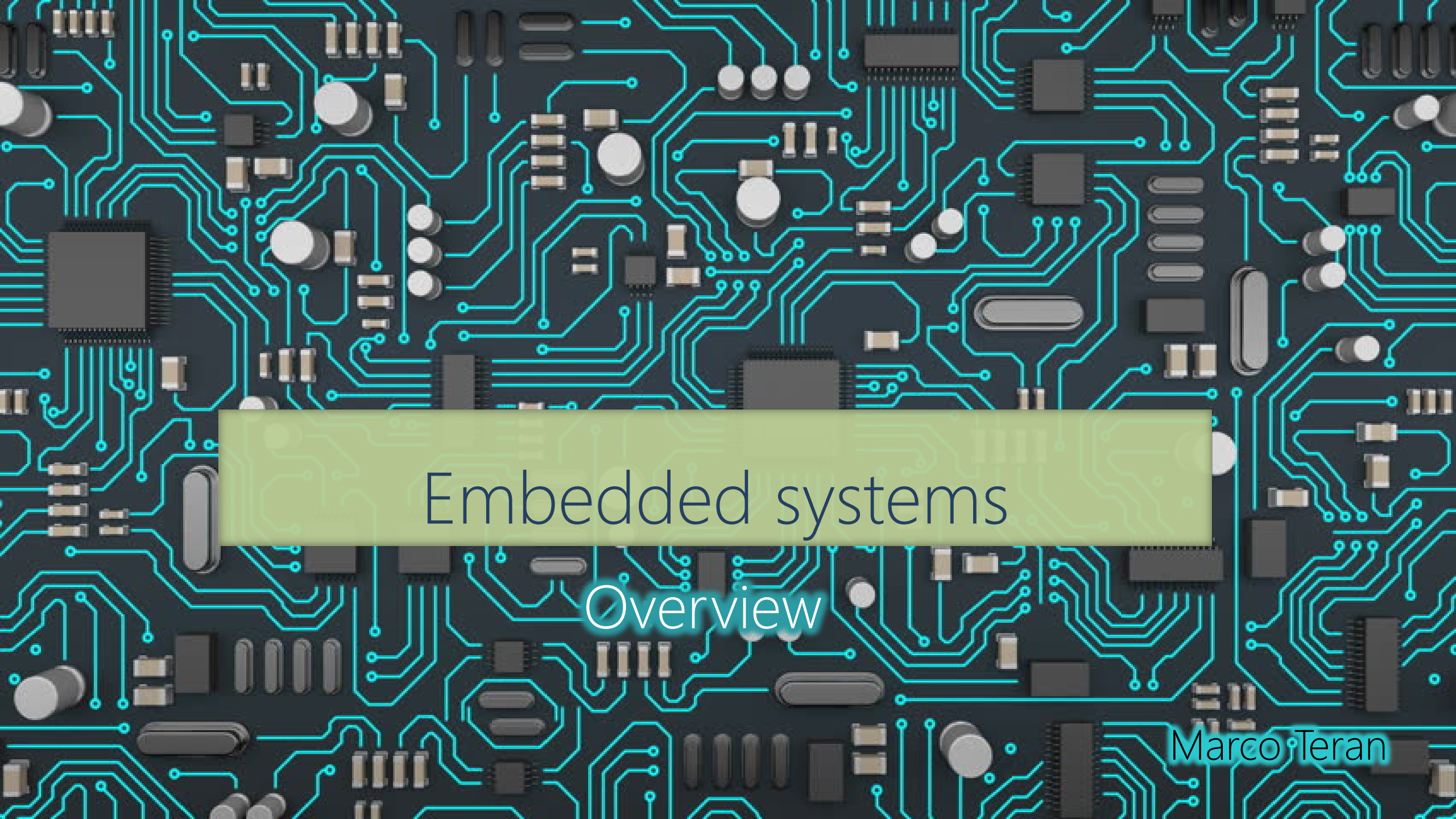
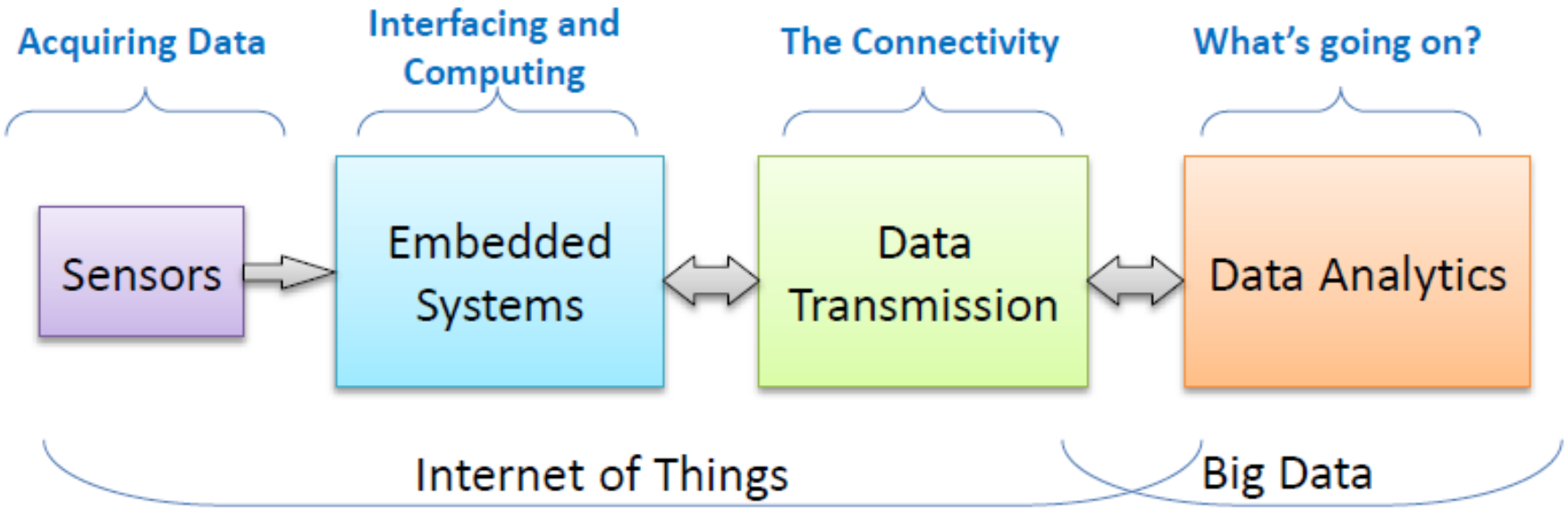


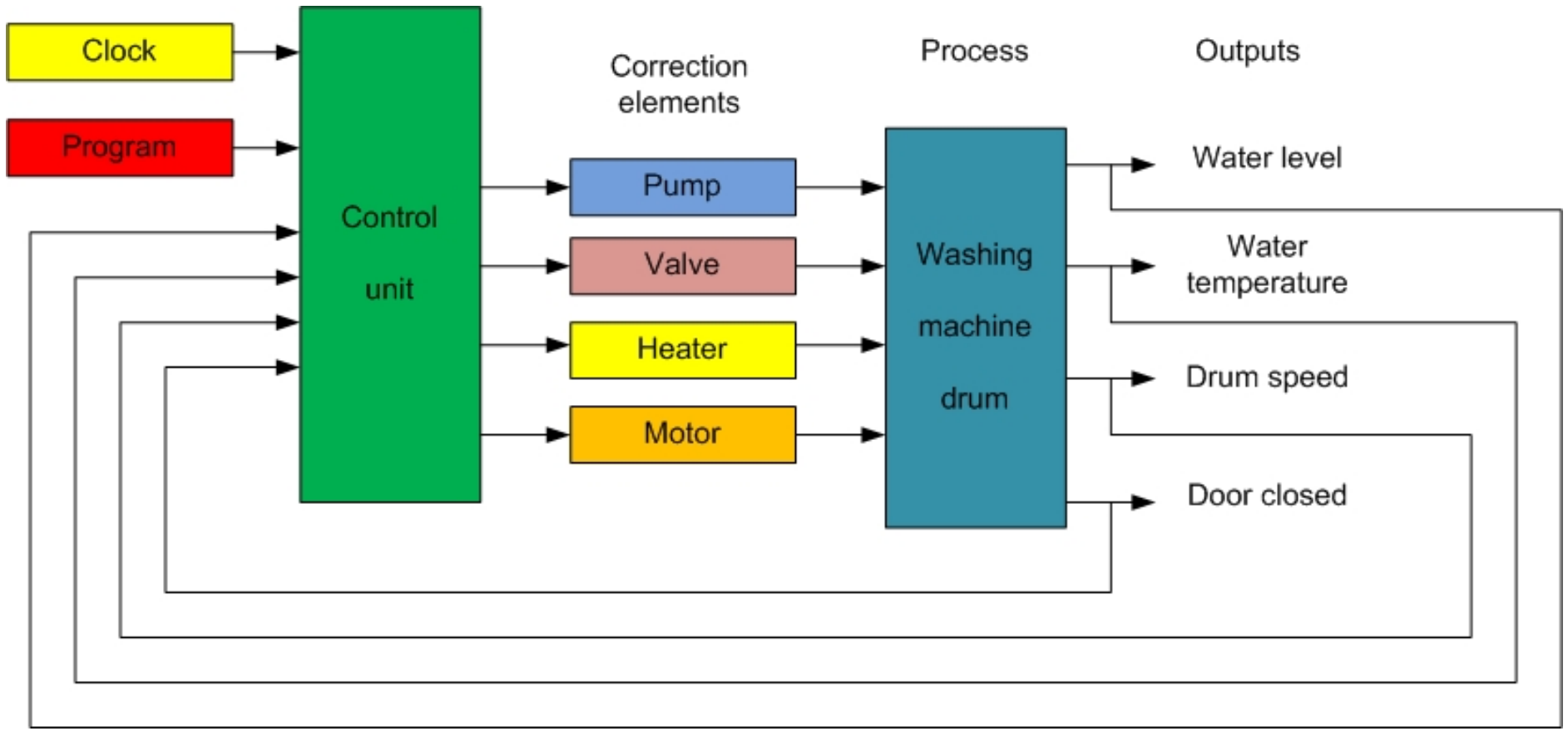
Embedded systems

Overview

Marco Teran

A Basic IoT Model





A Washing Machine

Introduction

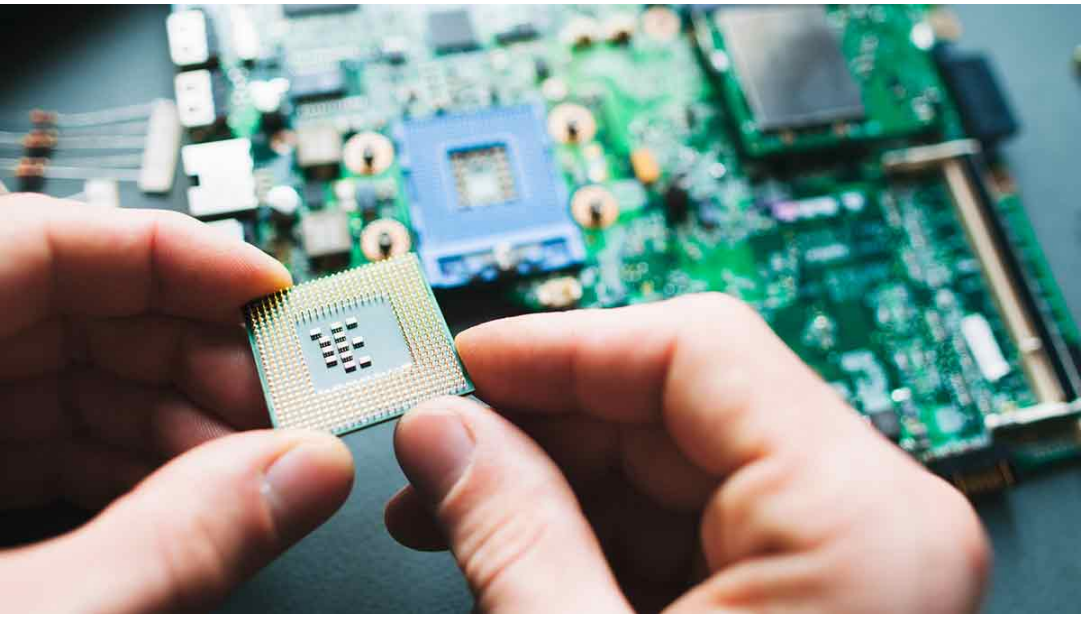
- Computing systems are everywhere
- Most of us think of “desktop” computers
  - PC’s
  - Laptops
  - Mainframes
  - Servers
- But there’s another type of computing system
  - Far more common…








Embedded Systems



- Embedded systems are computer-based systems which are embedded inside another device (car, home, TV, etc.)
- Interactions with embedded systems are different than with standard computers
- Interact with users via simple interface:
  - Digital camera, TV, cellphone, …
- Interact with another device, invisible to user:
  - Disk drive, memory stick, anti-lock braking system, …

An embedded system is a computer-based system which is specifically designed to perform one or several dedicated functions in real-time.

Embedded Systems

"Dortmund" Definition: [Peter Marwedel]

Embedded systems are information processing systems embedded into a larger product

Berkeley: [Edward A. Lee]

Embedded software is software integrated with physical processes. The technical problem is managing time and concurrency in computational systems.

Definition: Cyber-Physical (cy-phy) Systems (CPS) are integrations of computation with physical processes [Edward Lee, 2006].

An **embedded system** on plug-in card with a processor, memory, power supply, and external interfaces. An embedded is a controller programmed and controlled by a real-time operating system (RTOS) with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints

Forestry Machines

- Networked computer system
  - Controlling arms & tools
  - Navigating the forest
  - Recording the trees harvested
  - Crucial to efficient work
- “Tough enough to be out in the woods”

Smart buildings

Examples

- Integrated cooling, lightning, room reservation, emergency handling, communication
- Goal: “Zero-energy building”
- Expected contribution to fight against global warming

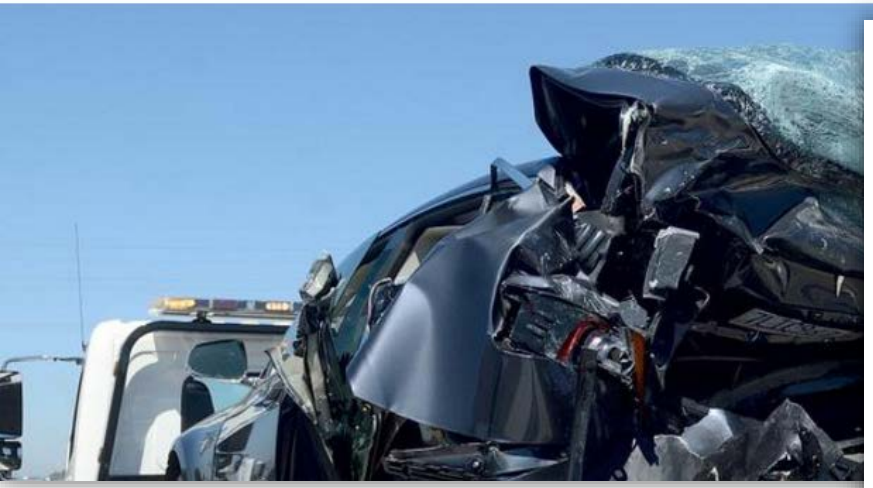
ES 2016 Grand Failure

AUGUST 15 TSLA: 2

Rare fatal accident in a Tesla Model S rear-ended by a large SUV in California

Fred Lambert - 2 months ago @FredericLambert

TESLA TESLA MODEL S TESL



Tesla in fatal California crash was on Autopilot

31 March 2018 Share

del X died shortly after the crash

a says a vehicle involved in a fatal crash in California
, raising further questions about the safety of self-

Model X cars crashed into a roadside barrier and caught fire

Serious safety lapses led to Uber's fatal self-driving crash, new documents suggest

The fatal crash in Tempe, Arizona, in 2018 comes into sharper focus

By Andrew J. Hawkins | @andyjayhawk | Nov 6, 2019, 11:45am EST

SHARE

Image: ABC 15

Google's discont
Amazon, Best B

Can ES get Hacked?

AFTER JEEP HACK, CHRYSLER RECALLS 1.4M VEHICLES FOR BUG FIX

RESEARCHERS HACKED A MODEL S, BUT TESLA'S ALREADY RELEASED A PATCH

HACKERS TAKE DOWN POWER STATIONS IN UKRAINE

MAY BE THE FIRST CYBERATTACK TO SUCCESSFULLY CAUSE A BLACKOUT

By Kelsey D. Atherton Posted January 5, 2016

Apagón en Venezuela: Maduro denuncia un "ciberataque"

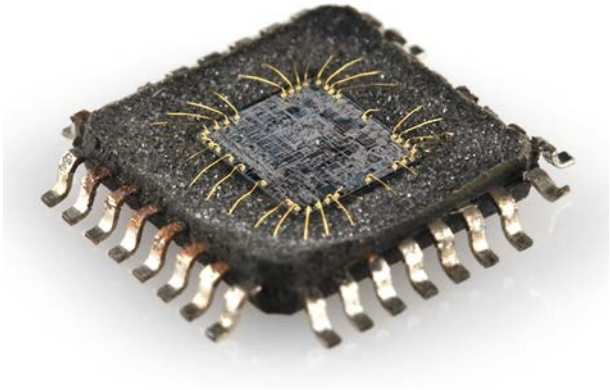
Por Yaiza Martín-Fradejas · última actualización: 10/03/2019

Los venezolanos se enfrentan a un **tercer día sin electricidad**. En Caracas, un grupo de personas salió a

A “short list” of embedded systems

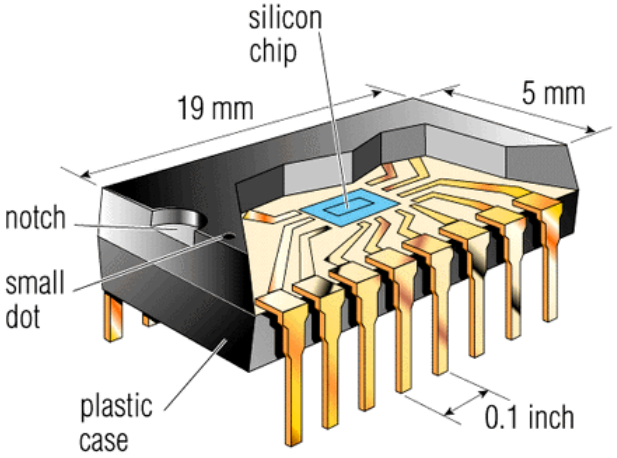
- Anti-lock brakes
- Auto-focus cameras
- Automatic teller machines
- Automatic toll systems
- Automatic transmission
- Avionic systems
- Battery chargers
- Camcorders
- Cell phones
- Cell-phone base stations
- Cordless phones
- Cruise control
- Curbside check-in systems
- Digital cameras
- Disk drives
- 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

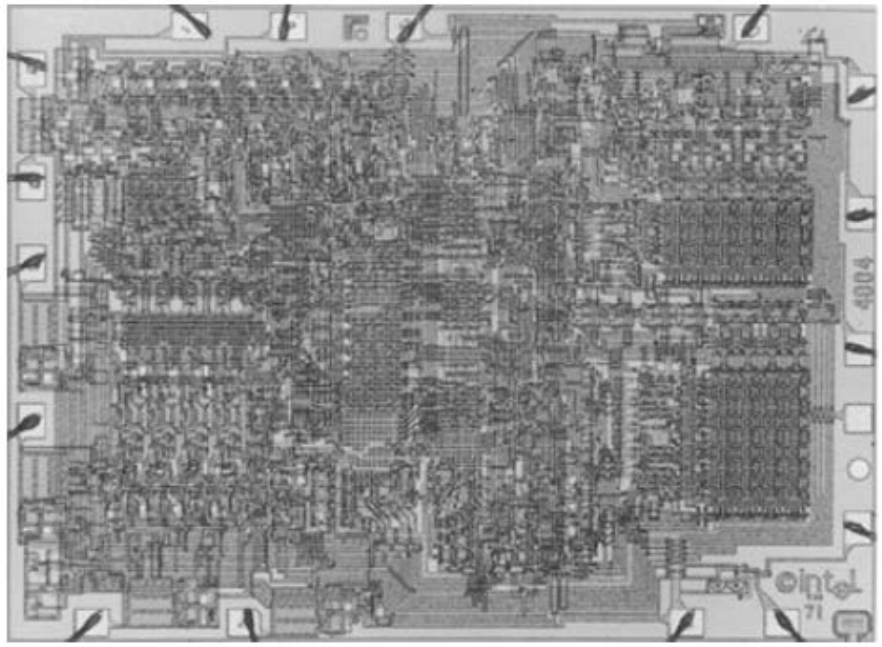
Integrated Circuit (IC)

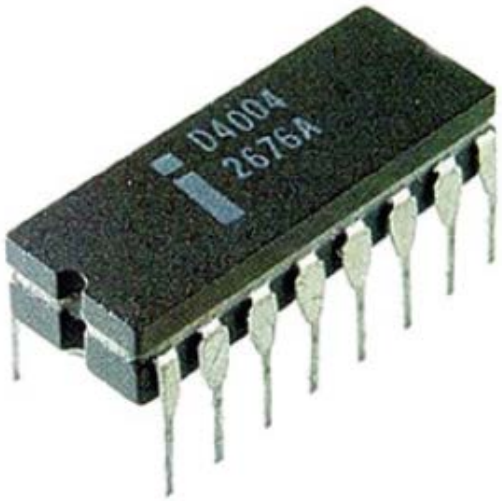


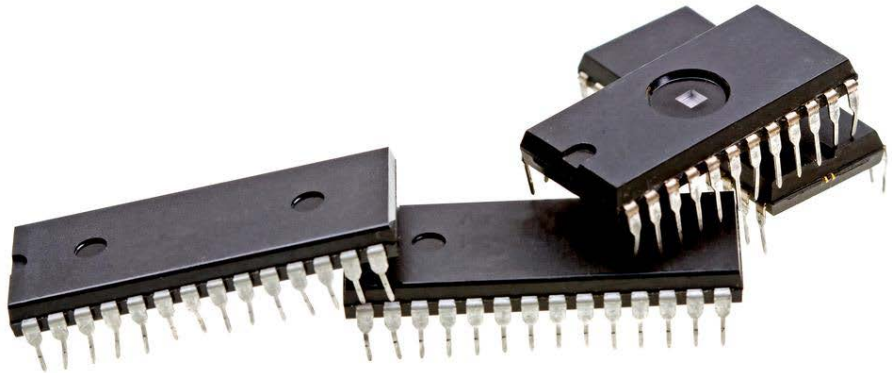
Also known as microchips, ICs are capable of performing specific tasks

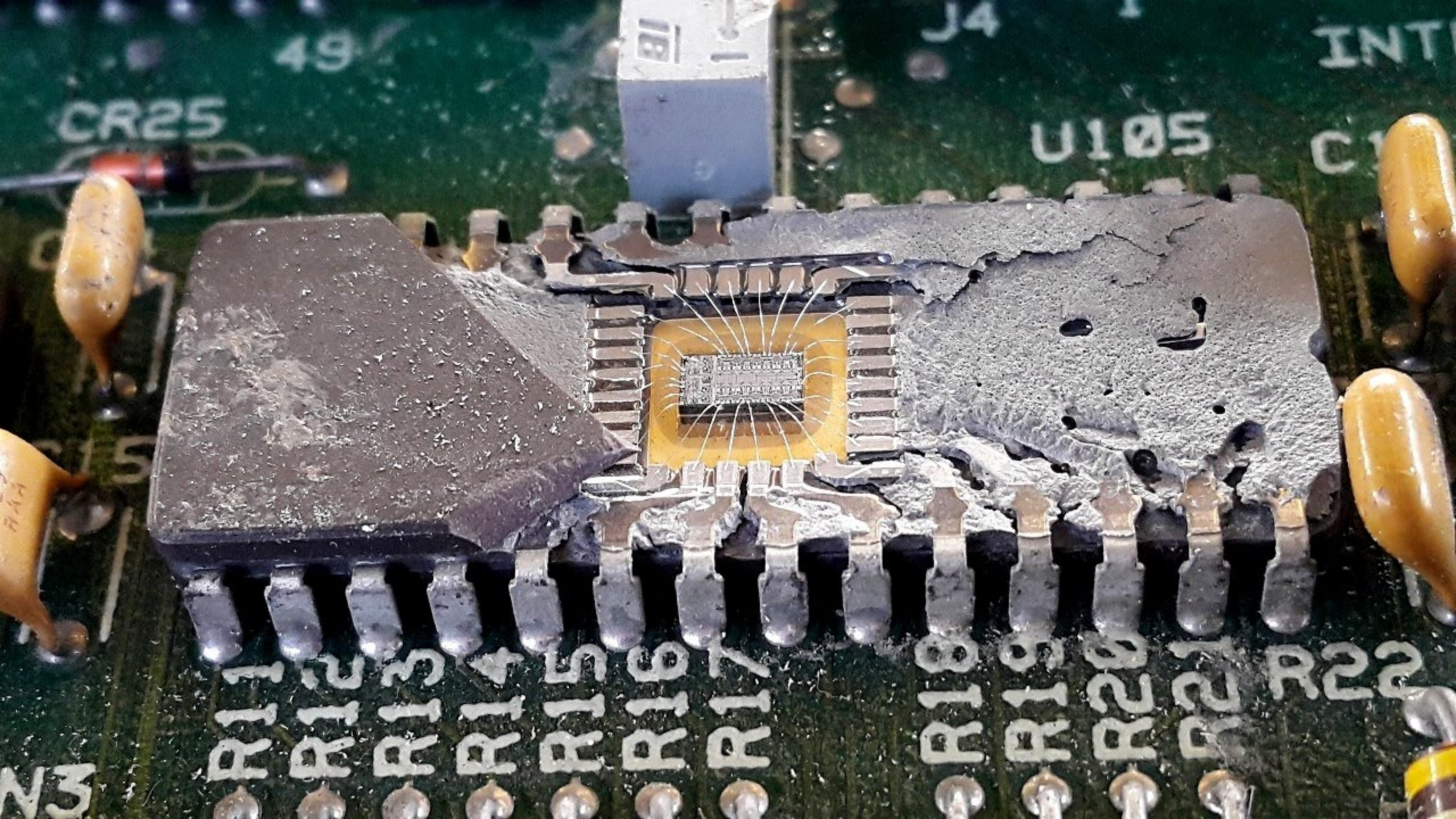
- There are many different types of IC
- A programmable IC is called a microcontroller
- ICs come in all different shapes and sizes
- Most come in a dual in line (DIL) package







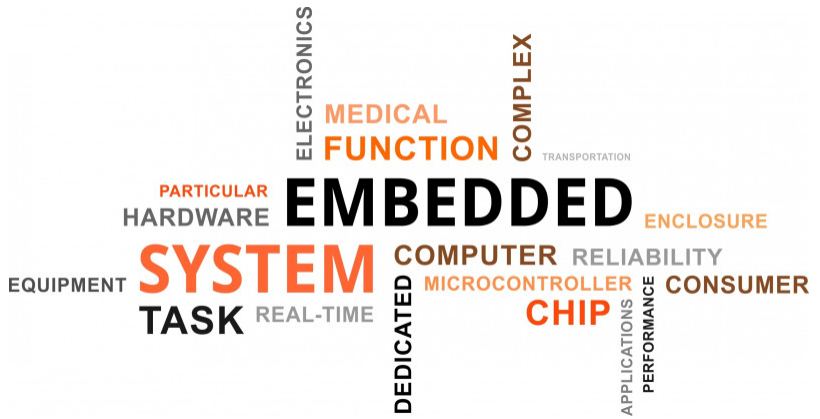


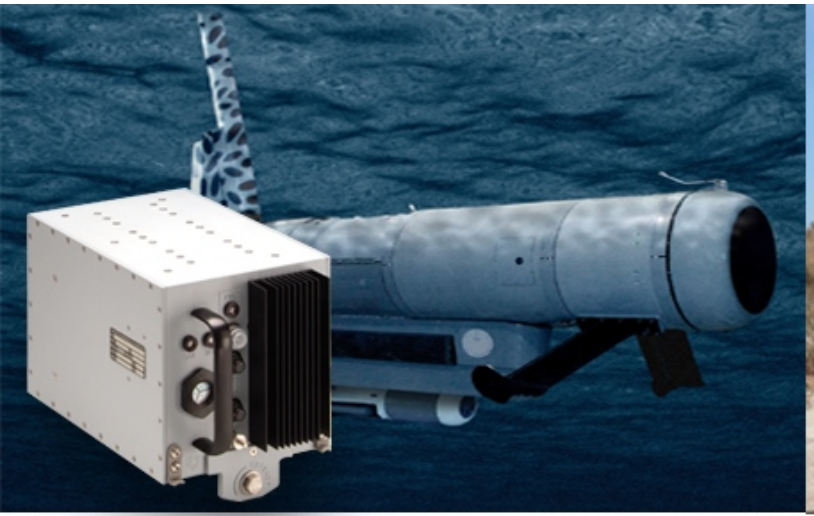

49
J4
CR25
U105
R22

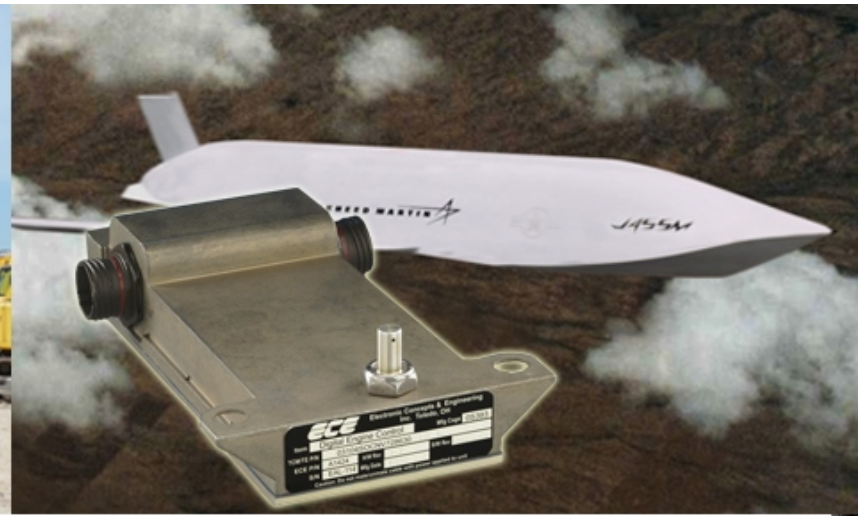
Embedded Systems characteristics

- Interact **directly** with the *physical environment*
  - ABS applies brakes, disk drive controls motors

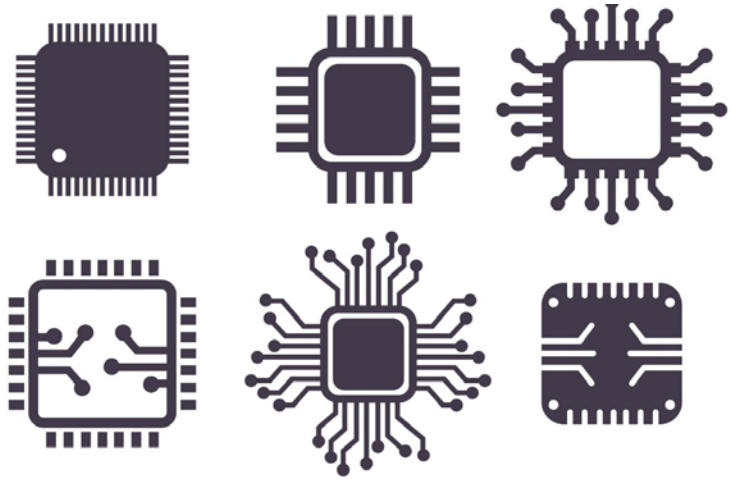
- **Hard timing constraints** (unlike most computer software): Perform real time operations
  - Must compute certain results in real-time without delay
  - ABS fails if it is too slow
  - MP3 player sound bad if conversion is slow



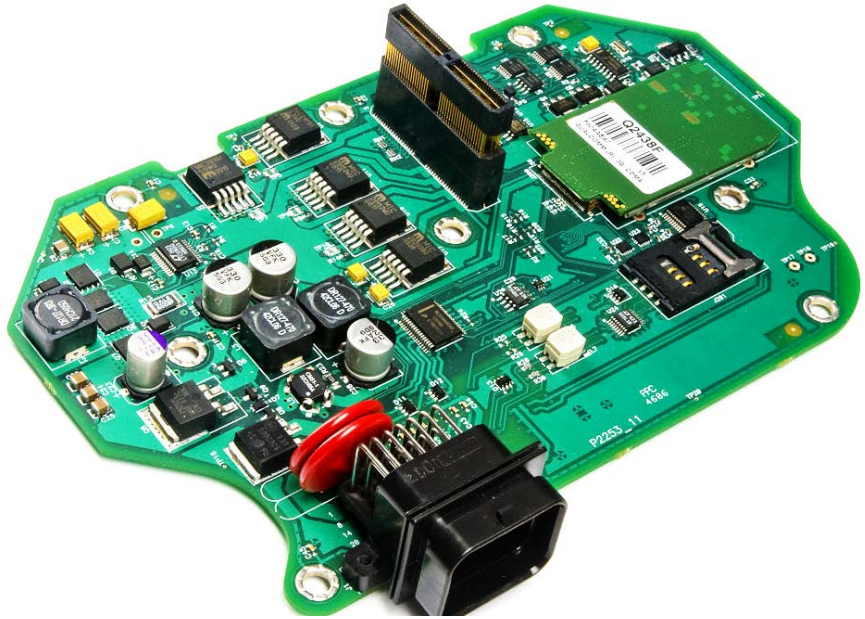




Embedded Systems characteristics



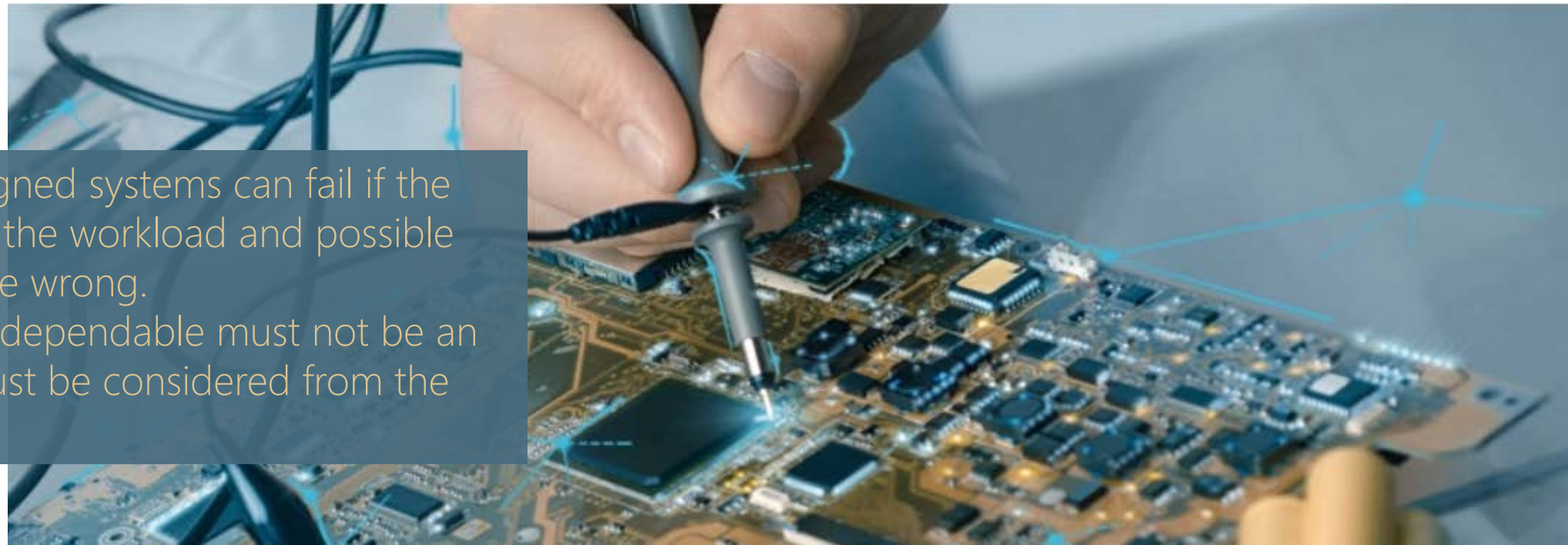
- **Application-specific**, not general-purpose: designed to execute a *specific task*
  - ABS only applies breaks, MP3 only plays music
  - Some systems violate this (cellphones w/ apps)
- **Dependability**
- Single-functioned (dedicated System)
  - Executes a single program, repeatedly
- Tightly-constrained (Efficient)
  - Comprises the least required resources
  - Low cost, low power, small, fast, etc.
- Often interfaced with *sensors* in order to get feedback of process parameters
- Designed to operate under harsh environments
- Reactive and real-time
  - Continually reacts to changes in the system's environment



Dependability

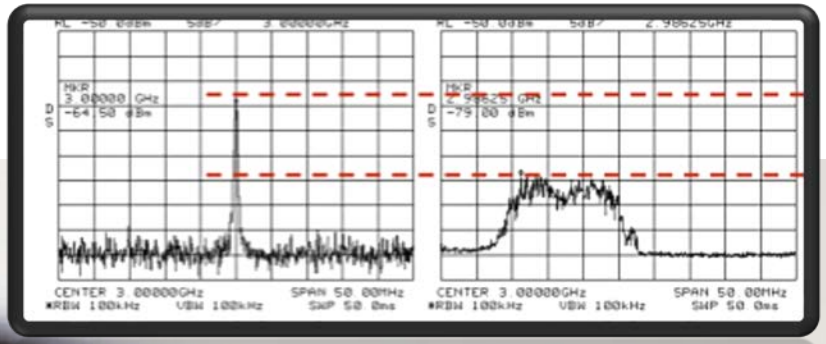
Embedded Systems must be dependable,

- Reliability *R(t)* probability of system working correctly provided that is was working at *t=0*
- Maintainability *M(d)* probability of system working correctly **d** time units after error occurred.
- Availability *A(t)* probability of system working at time **t**
- Safety no harm to be caused
- Security confidential and authentic communication



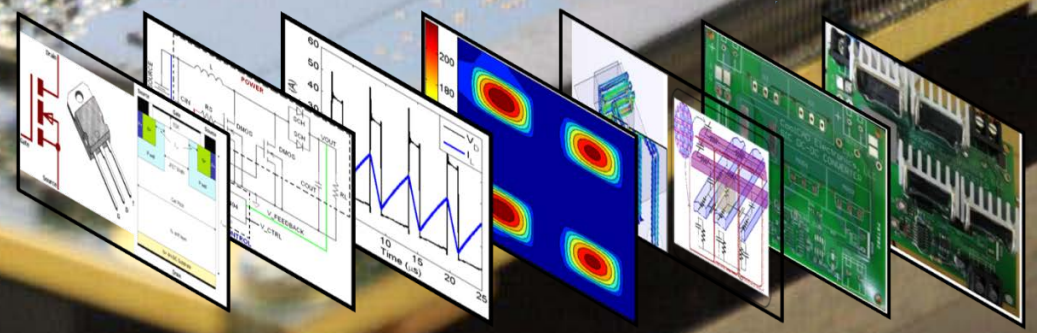
- Even perfectly designed systems can fail if the assumptions about the workload and possible errors turn out to be wrong.
- Making the system dependable must not be an after-thought, it must be considered from the very beginning

Efficiency



ES must be **efficient**

- Code-size efficient
- Run-time efficient
- Weight efficient
- Cost efficient
- Energy efficient



Cost Constraints

- **Embedded design methodology:** Be Efficient!
  - Vast majority of embedded products are in cost critical markets
- Very different from traditional software engineering
  - Correct functionality is all that counts
  - Performance, memory, power, not important
  - Moore's law will save you eventually

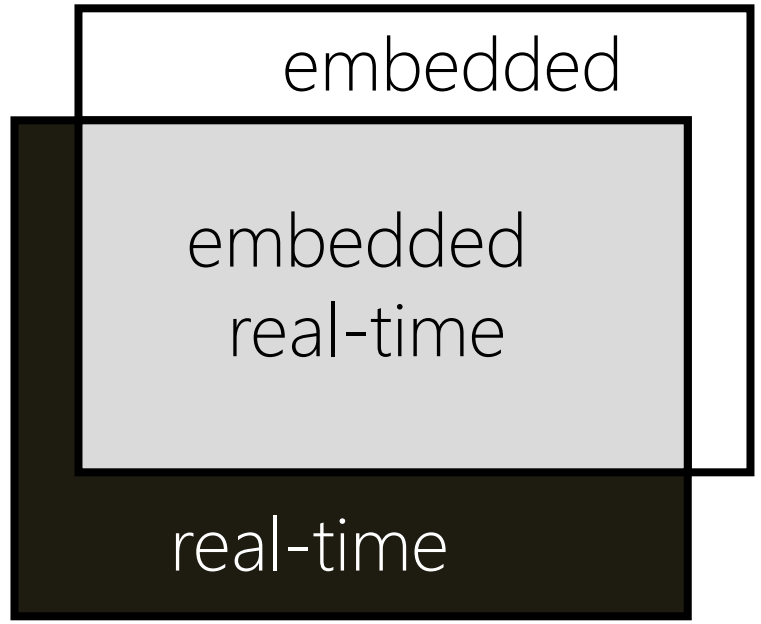
Embedded system designers should be control freaks

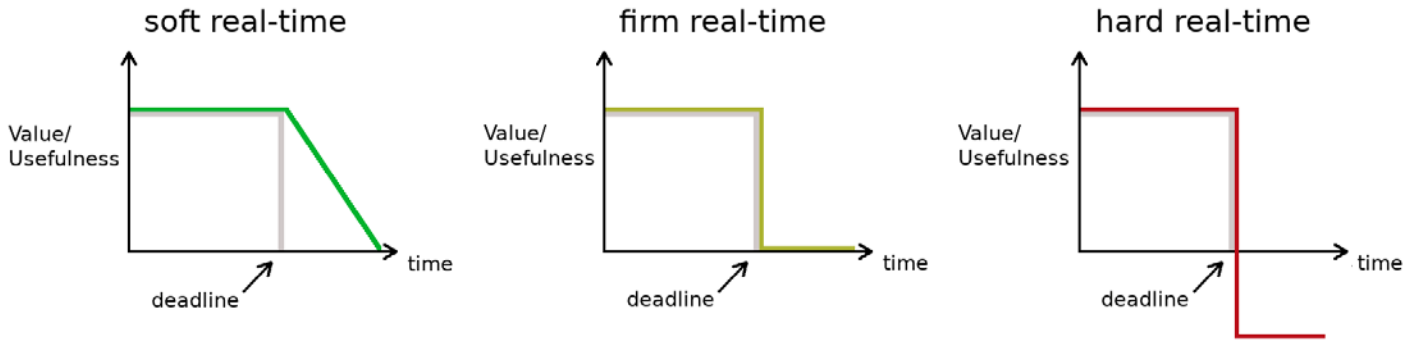
- Need to have explicit control over timing, memory
- Why trust a compiler/interpreter/OS?

Real-time constraints

Many ES must meet **real-time constraints**

- A real-time system must react to stimuli from the controlled object (or the operator) within the time interval **dictated** by the environment.
- For real-time systems, right answers arriving too late are wrong.
- **"A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe"** [Kopetz, 1997].
- All other time-constraints are called **soft**.
- A guaranteed system response must be explained without statistical arguments

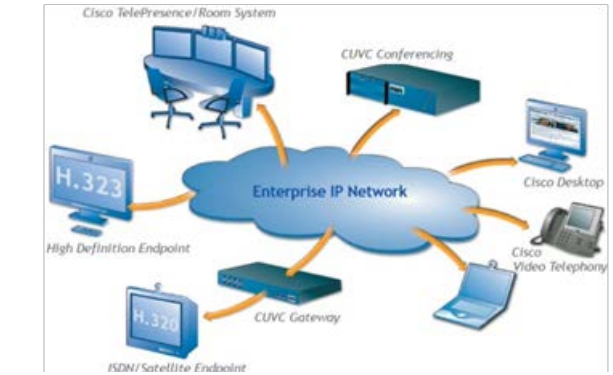


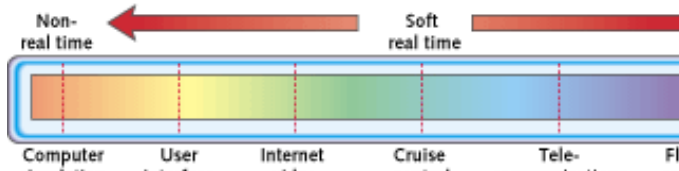


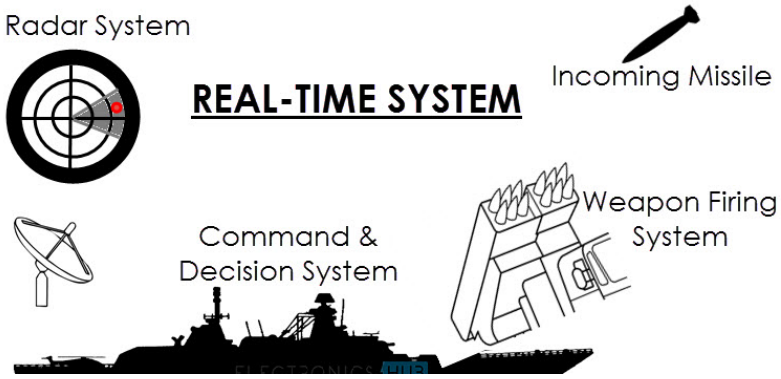
Real-Time Systems

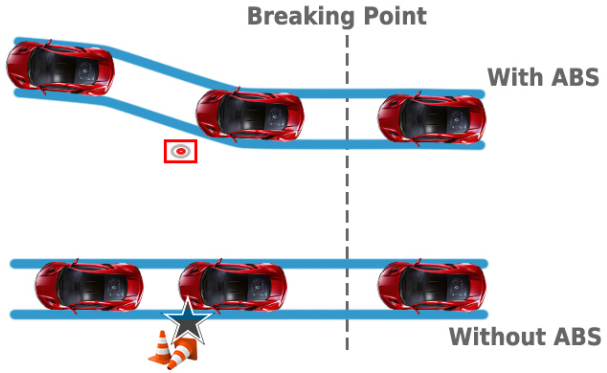
Embedded and Real-Time?

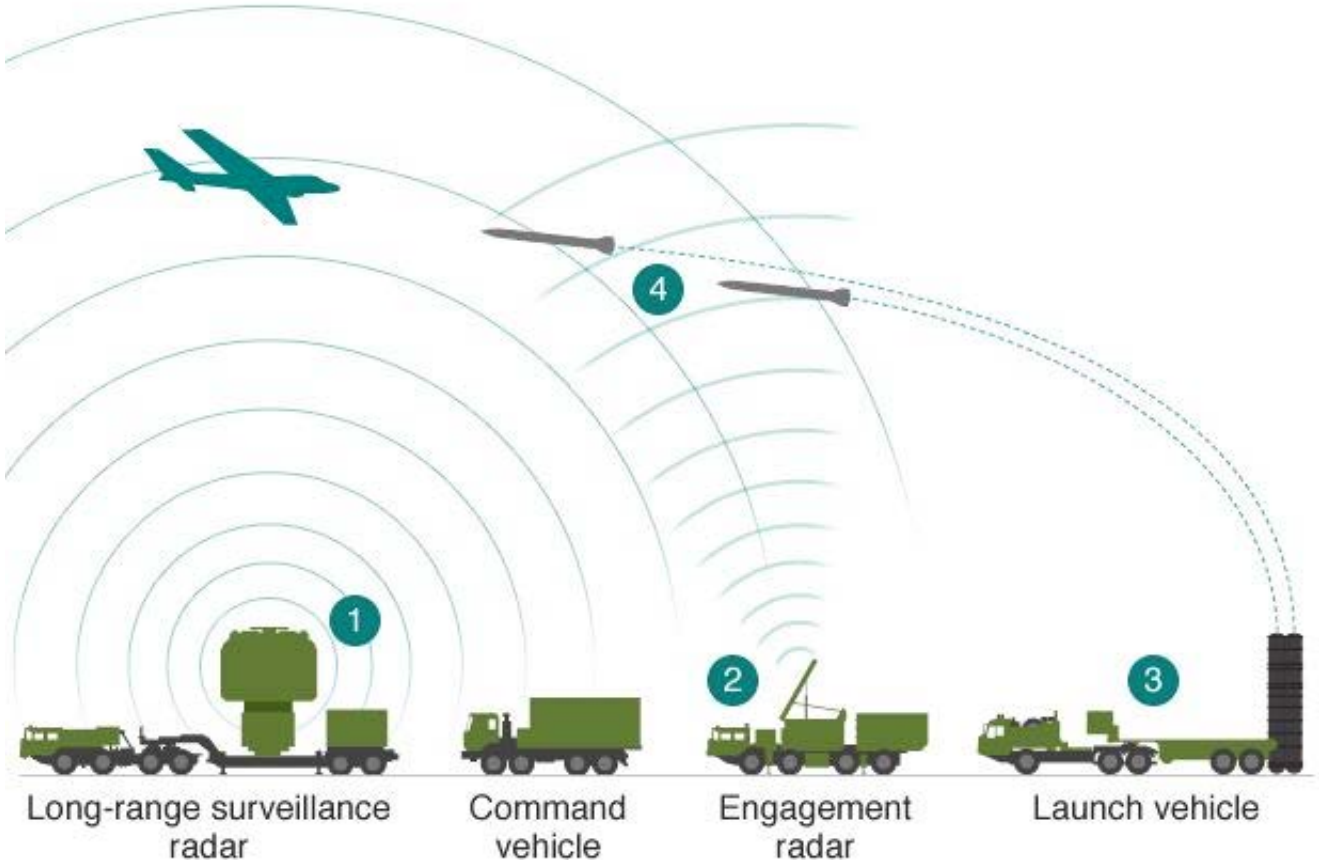
- Most embedded systems are real-time
- Most real-time systems are embedded

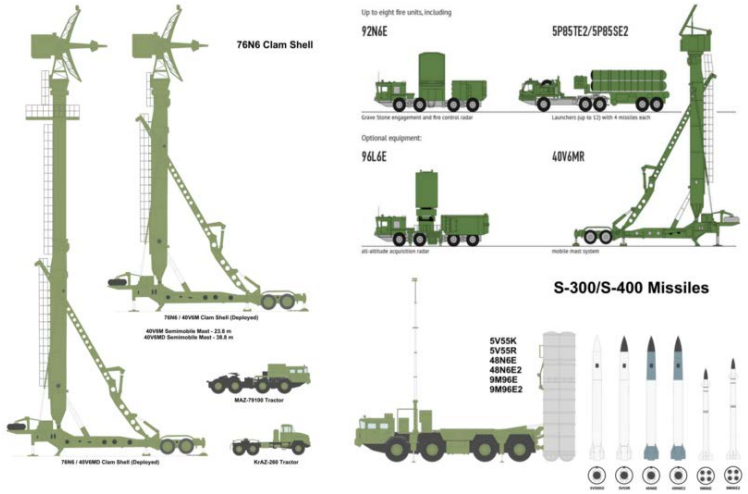








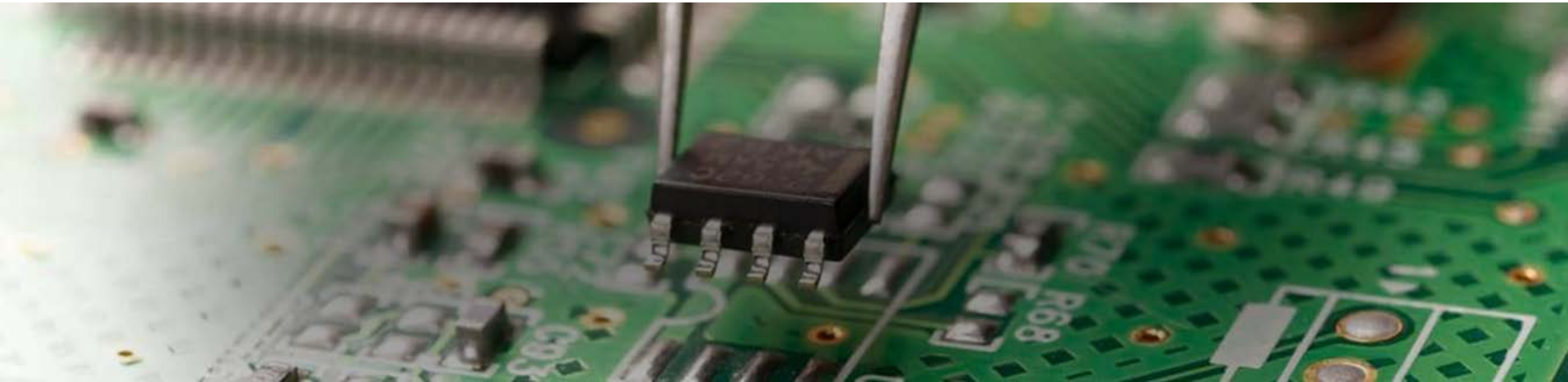

4
1
2
3
Long-range surveillance radar
Command vehicle
Engagement radar
Launch vehicle


76N6 Clam Shell
92N6E
5P85TE2/5P85SE2
96L6E
40V6MR
S-300/S-400 Missiles
5V55K
5V55R
48N6E
48N6E2
9M96E
9M96E2

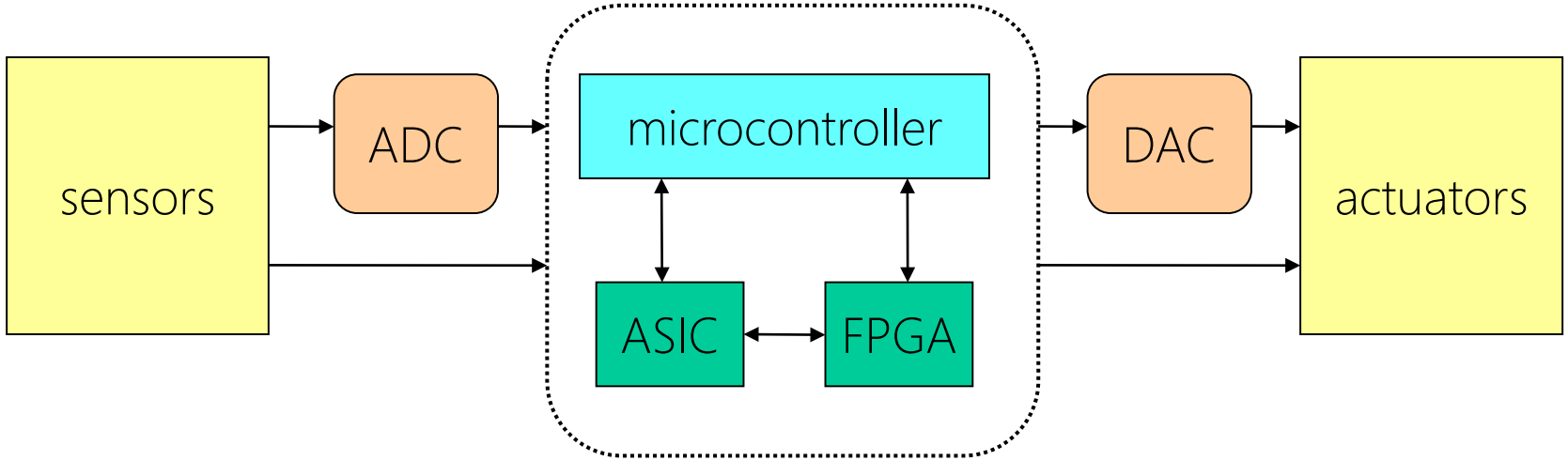
Dedicated systems

- **Dedicated** towards a certain **application**
  Knowledge about behavior at design time can be used to minimize resources and to maximize robustness

- **Dedicated user interface**
  (no mouse, keyboard and screen)

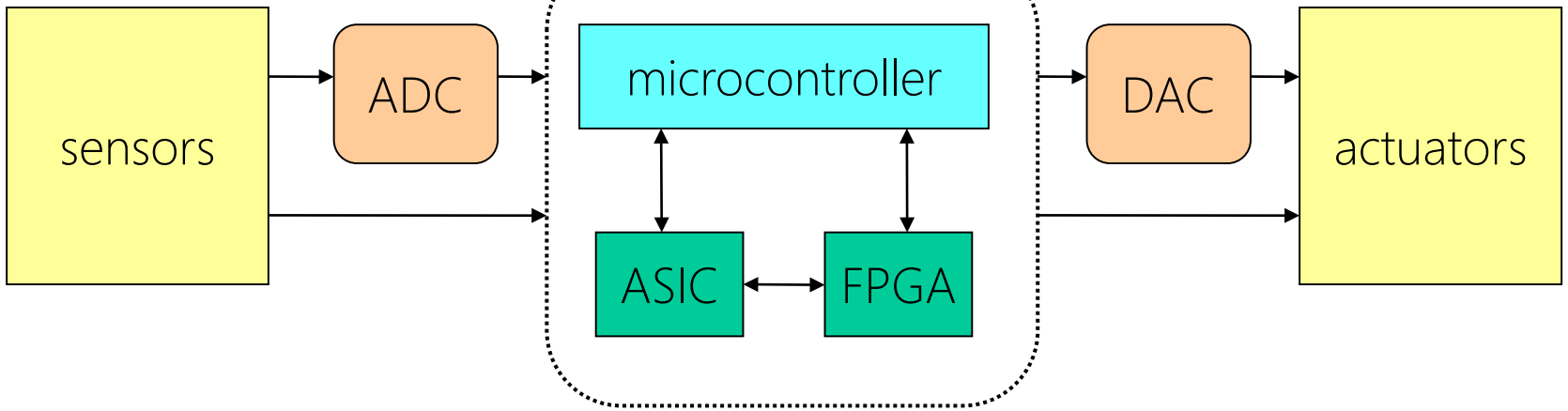


Embedded System Structure

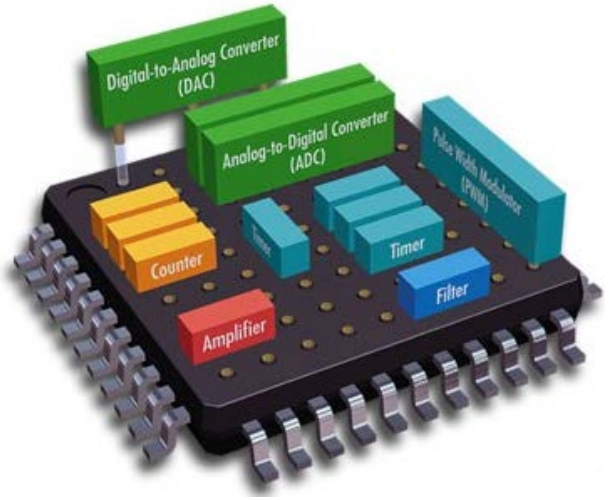


- Sensors receive data from the world
  - Analog (light sensor, microphone, etc.) and digital (buttons)
- Actuators cause events in the world
  - Analog (motors, speakers) and digital (lights)
- Application-Specific Integrated Circuit (ASIC) - special-purpose hardware
- Field Programmable Gate Array (FPGA) - reconfigurable hardware

Embedded System Structure

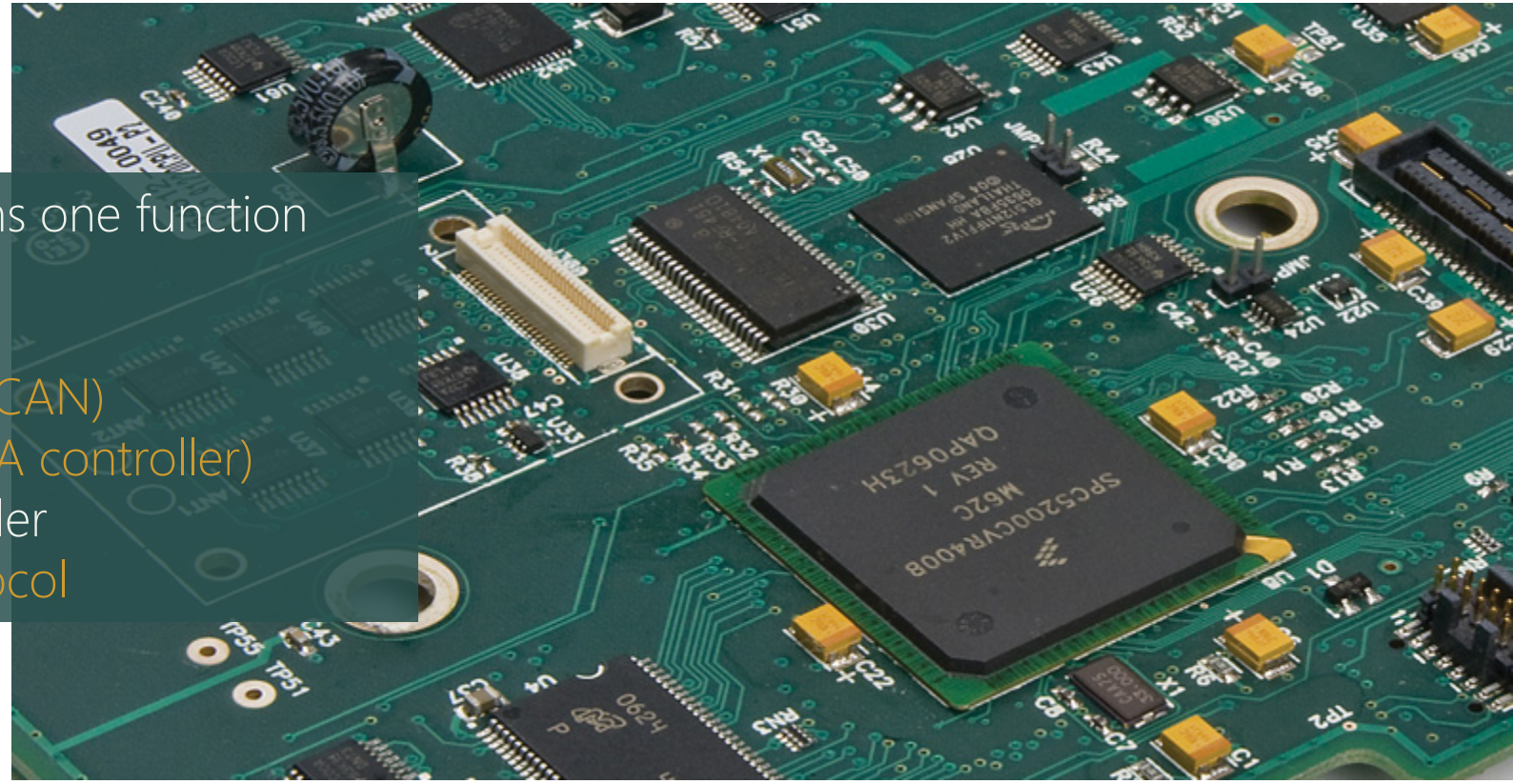


Embedded system hardware is frequently used in a loop
("hardware in a loop")

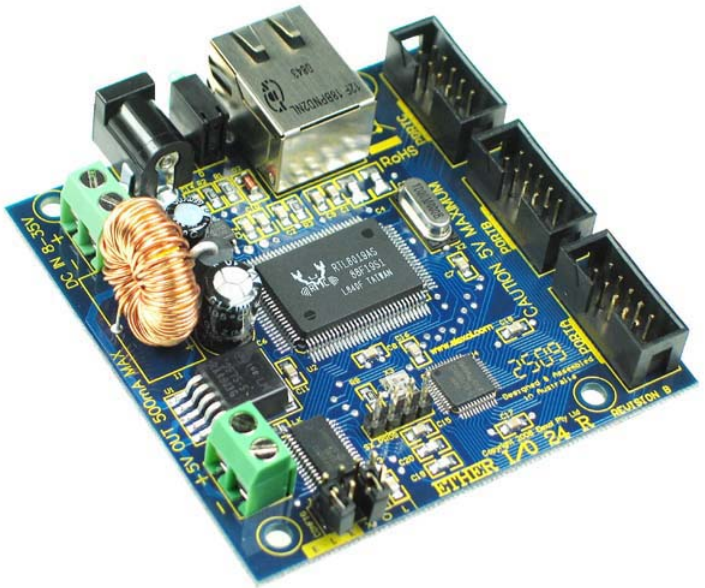


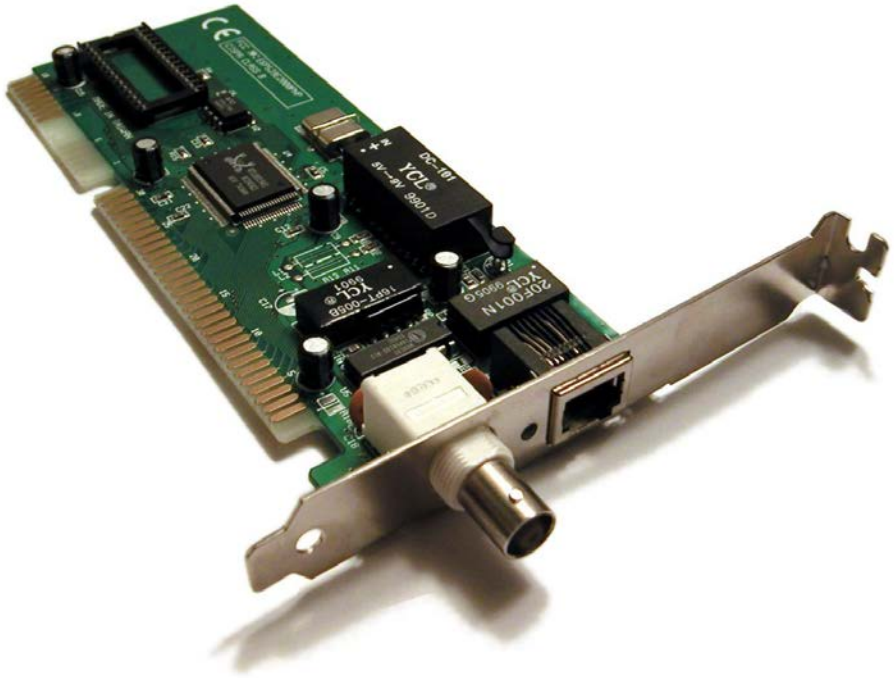
Intellectual Property (IP) Core

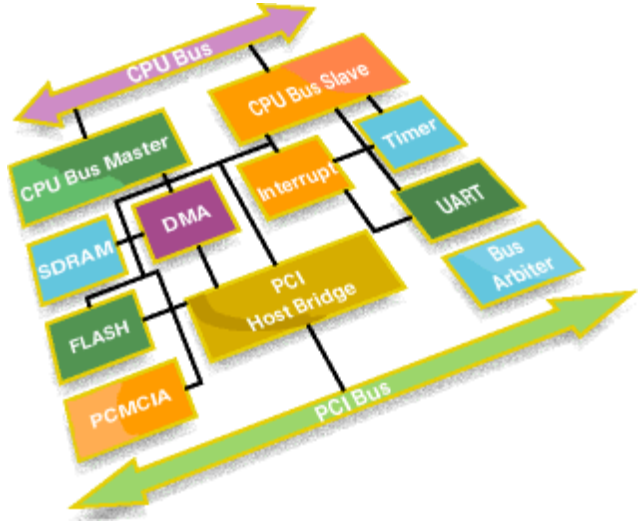
- An integrated circuit which performs one function
- Cheap in high volume
- Very useful for common tasks
  - Network controllers (ethernet, CAN)
  - Audio/Video (audio codec, VGA controller)
- Must interact with the microcontroller
  - Consider communication protocol



Intellectual Property (IP) Core

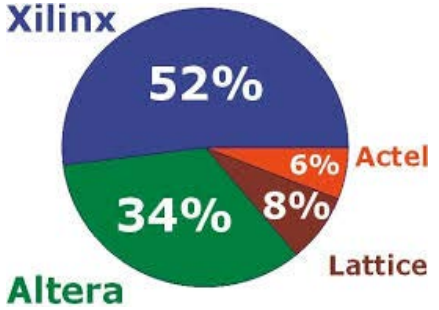


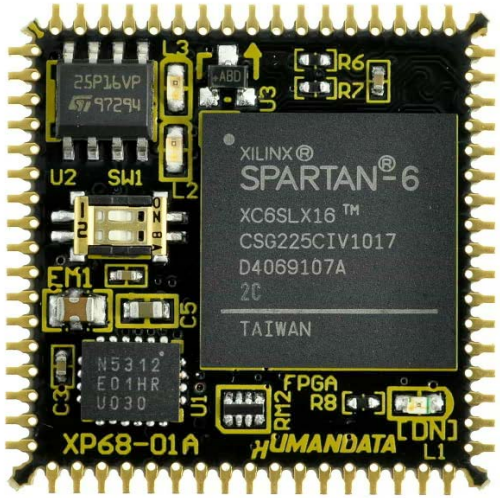


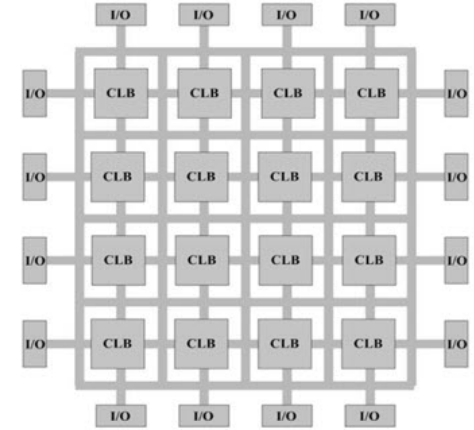


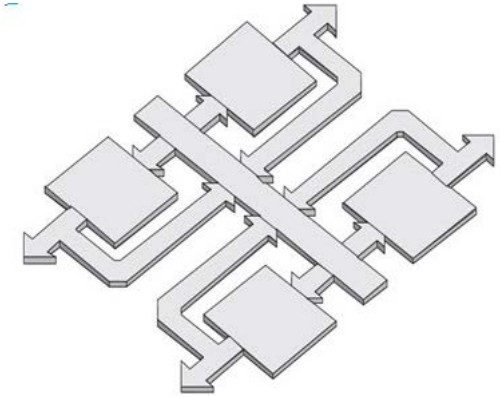
Field Programmable Gate Array (FPGA)

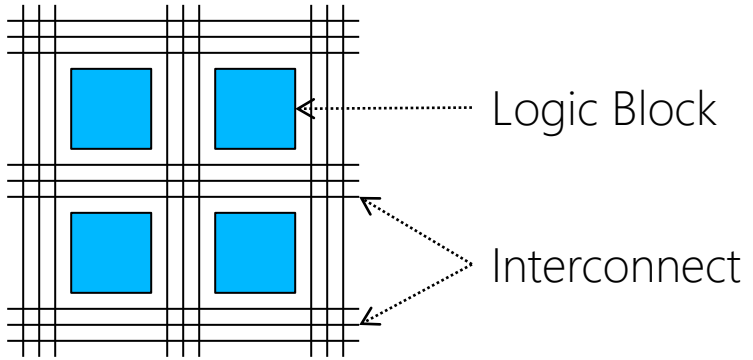
- Hardware which can be reconfigured via RAM
- Faster than SW, slower than ASIC
- No fabrication needed











Controllers

Four important factors for the controller

- Number of transistors -> size, cost, power
- Number of clock cycles -> power
- Time to MARKET-> cost, acceptance
- Nonrecurring engineering cost (NRE) -> cost, acceptance

Ideal: Minimize all factors at the same time!

Tasks of the controller

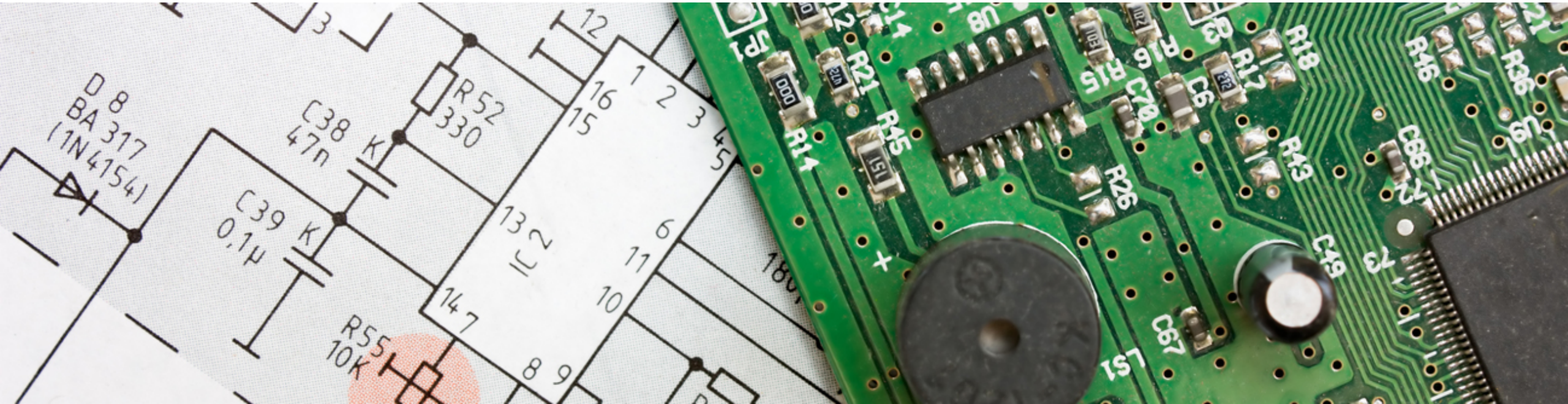
- Running of (real time) data processing and communication protocols
- Perform and control the application program
- Energy management of the node
  - Different operation modes available (active, idle, listen, sleep, etc.)

Embedded processors (MicroControllers)

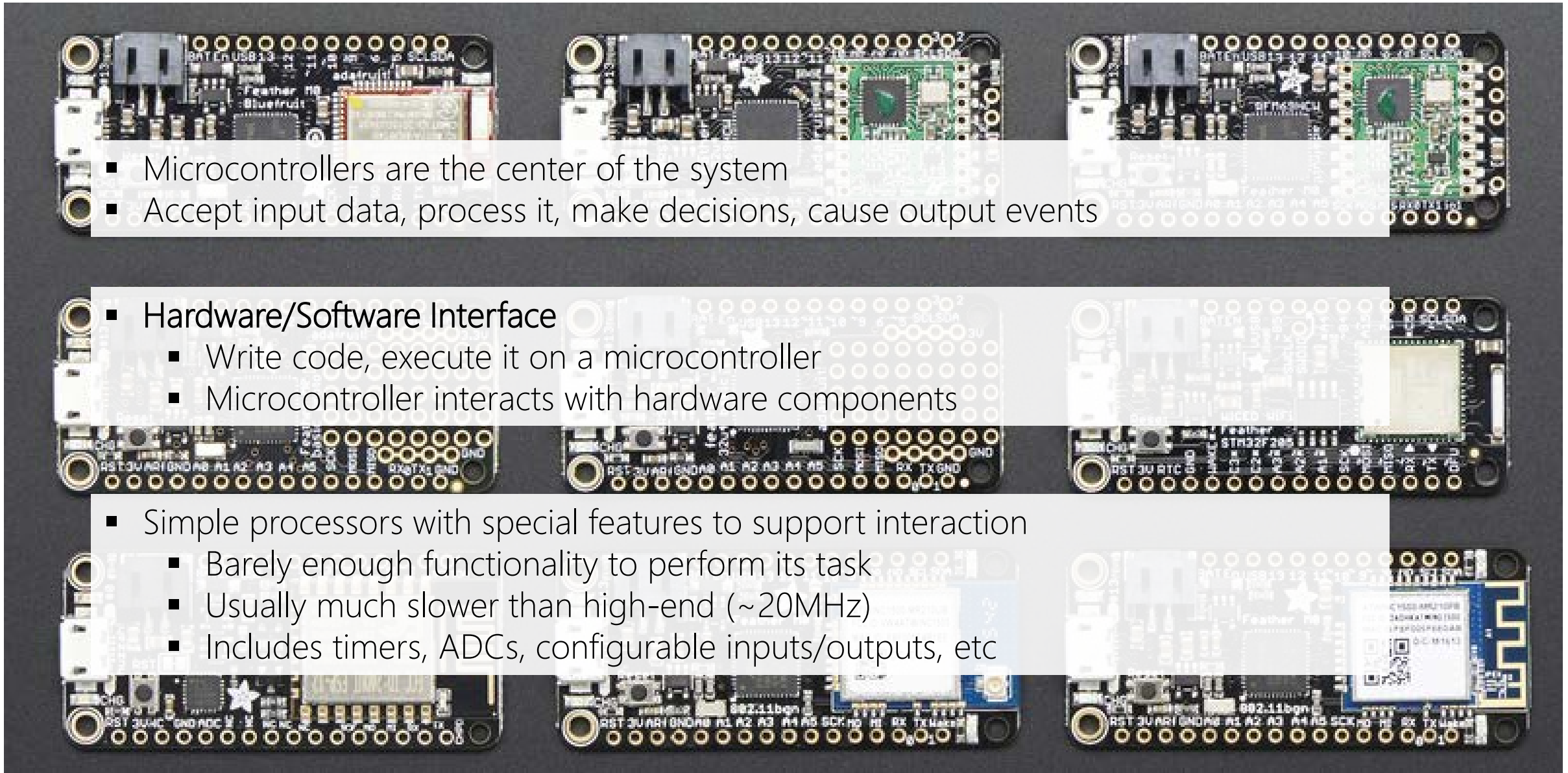
- Programmed once by manufacturer of system
- Executes a single program (or a limited suite) with few parameters
- Task-specific
  - can be optimized for specific application
- Interacts with environment in many ways
  - direct sensing and control of signal wires
  - communication protocols to environment and other devices
  - real-time interactions and constraints

Microcontrollers

- Processing power: 4-bit, 8-bit, 16-bit, 32 bit
- Microcontrollers.com: “The highest rate of new product success is in the 8-bit microcontroller market. The lowest rate of success is in the 64- and 32-bit microcontroller markets.”
- Specific features: communications, keyboard handling, signal processing, video processing



Microcontrollers



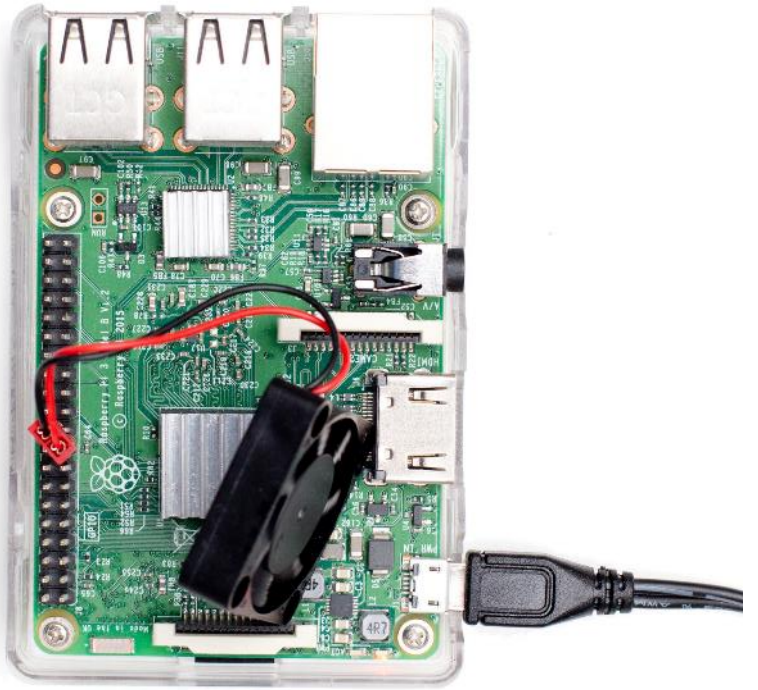
- Microcontrollers are the center of the system
- Accept input data, process it, make decisions, cause output events

- Hardware/Software Interface
  - Write code, execute it on a microcontroller
  - Microcontroller interacts with hardware components

- Simple processors with special features to support interaction
  - Barely enough functionality to perform its task
  - Usually much slower than high-end (~20MHz)
  - Includes timers, ADCs, configurable inputs/outputs, etc

Microcontrollers

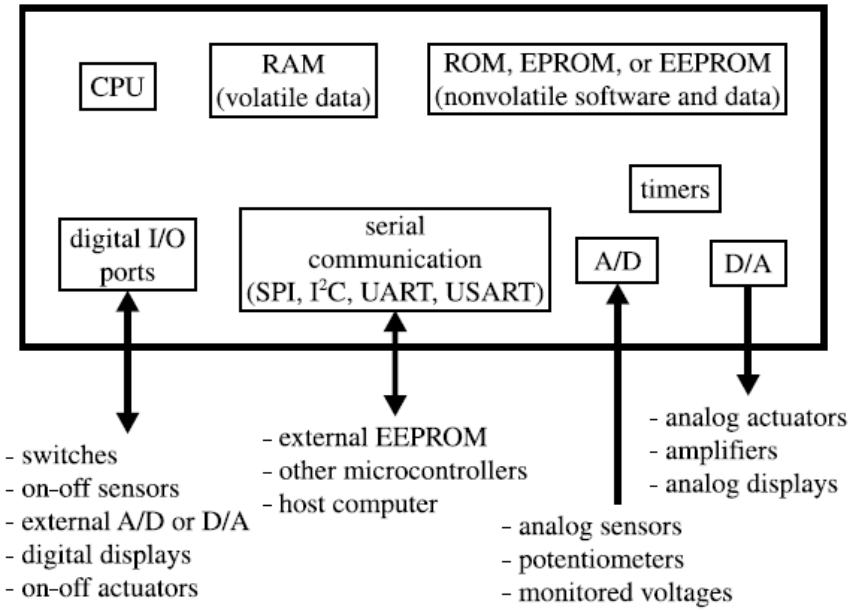
- A microcontroller IC is known as a peripheral interface controller (PIC)
  - These programmable ICs are very adaptable
  - They can be programmed to perform multiple processes
  - This reduces the amount of components needed to perform a task (component redundancy)



What are the benefits of having a smaller circuit board?

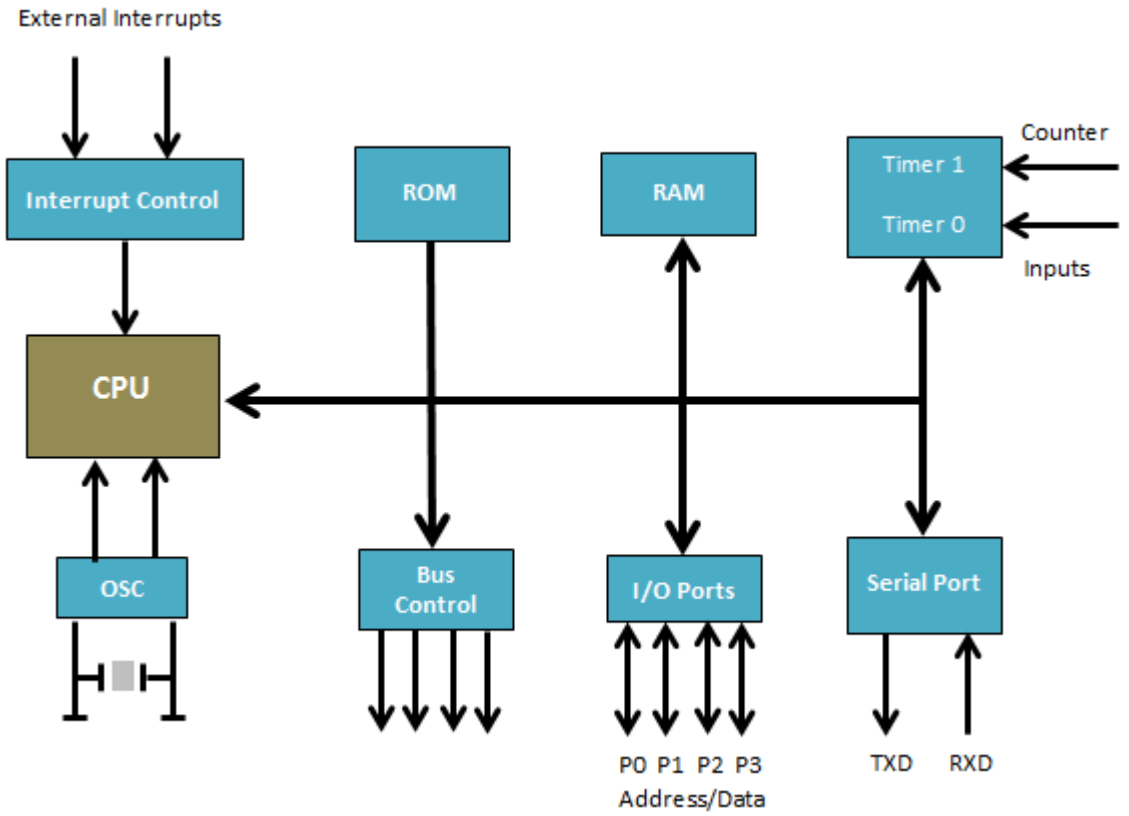
Microcontrollers Components

- CPU
  - Ranging from small and simple 4-bit processors to complex 32- or 64-bit processors.
- In-circuit programming and debugging support.
- Clock generator
  - often an oscillator for a quartz timing crystal, resonator or RC circuit.
- Random-access memory (RAM)
  - Type of volatile memory for data storage.
- Read-only memory (ROM),
  - Erasable programmable read only memory (EPROM)
  - Electrically Erasable Programmable Read-Only Memory (EEPROM)
  - Flash memory for program and operating parameter storage.

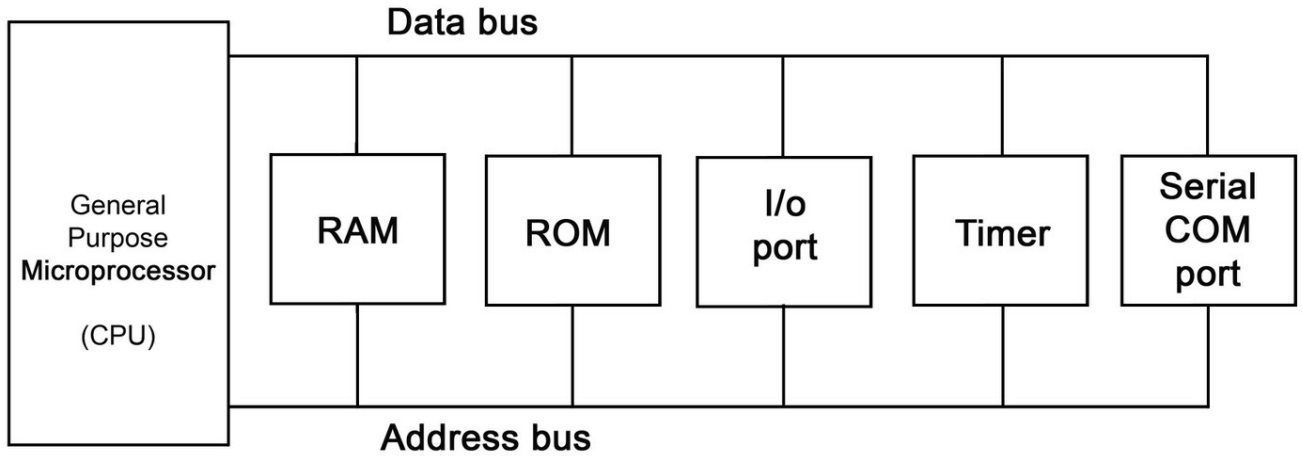


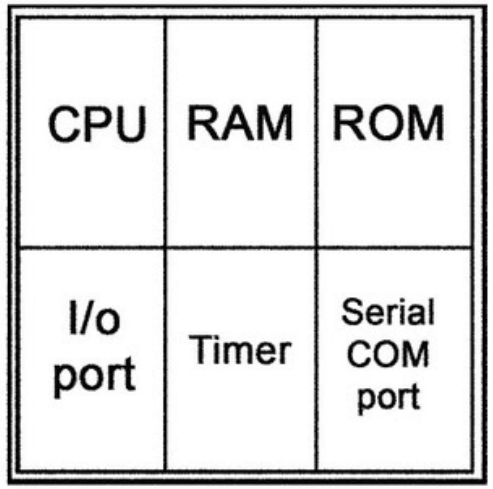
Microcontrollers Components

- Discrete input and output bits, allowing control or detection of the logic state of an individual package pin.
- Serial input/output such as serial ports (UARTs).
  - I^2C,
  - Serial Peripheral Interface (SPI)
  - Controller Area Network (CAN) for system interconnect
- Peripherals such as timers, event counters.
- Pulse width modulation (PWM) generators.
- Analog-to-digital converters and digital-to-analog converters.



A Single Chip Microcontroller Contains





Microprocessors and Microcontrollers

- A microprocessor unit (MPU) is a processor on one silicon chip.
- Microcontrollers are used in embedded computing.
- A Microcontroller unit (MCU) is a microprocessor with some added circuitry on one silicon chip.

General-Purpose vs. DSP

- General-Purpose
  - Used in for any application
  - Many features included

- Digital Signal Processors (DSP)
  - Made to support DSP functions
  - Vector instructions
  - Cheaper but more limited

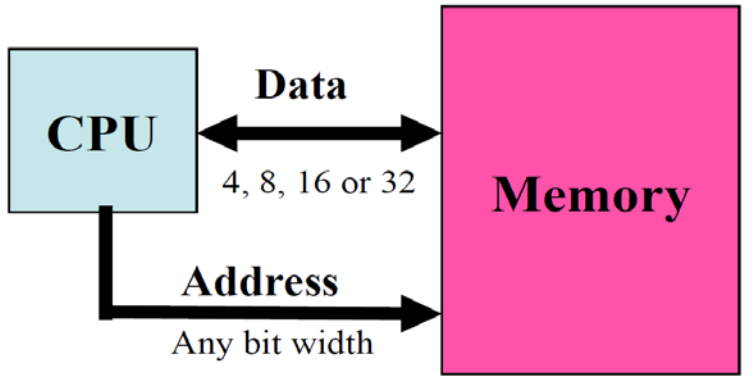
MCU Types

MCUs can be classified according:

- MCU bits (e.g. 8-bit, 16-bit, 32-bit)
- Instruction Set Architecture (ISA)

MCU Bit Definition

The number of bits describing the data path defines Microcontroller Bit Definition



Microcontroller Features

- Processor speed: Fundamental measure of processing rate of device
  - Value of interest is in MIPS, not MHz
- Supply voltage/current: Measure of the amount of power required to run the device
- Multiple modes (sleep, idle, etc)

Common Memory Types in MCUs

- RAM
- EEPROM
- Flash Memory

Popular Microcontrollers

- Intel 8051
- Microchip PIC
- Atmel AVR
- ARM

Atmel AVR

AVR

ATX Mega

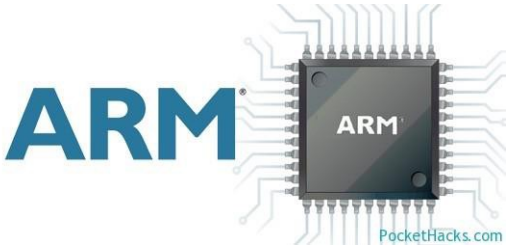
ATmega 328P

PIC 18F877A

8051

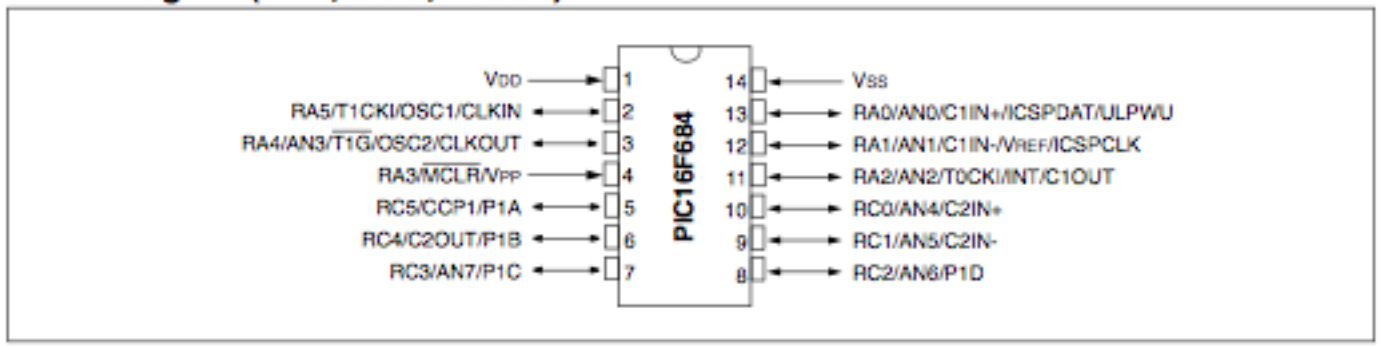
Arduino

ARM



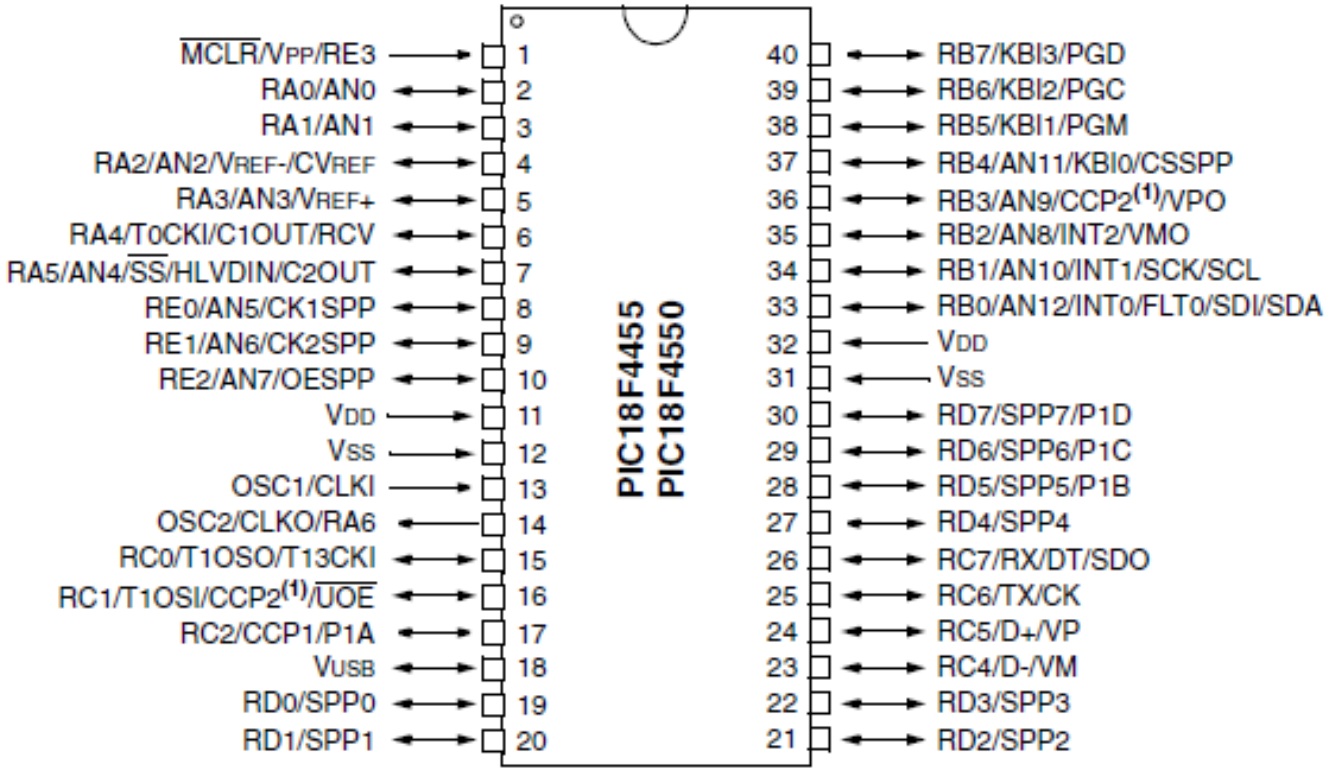
Microchip PIC16F684

14-Pin Diagram (PDIP, SOIC, TSSOP)



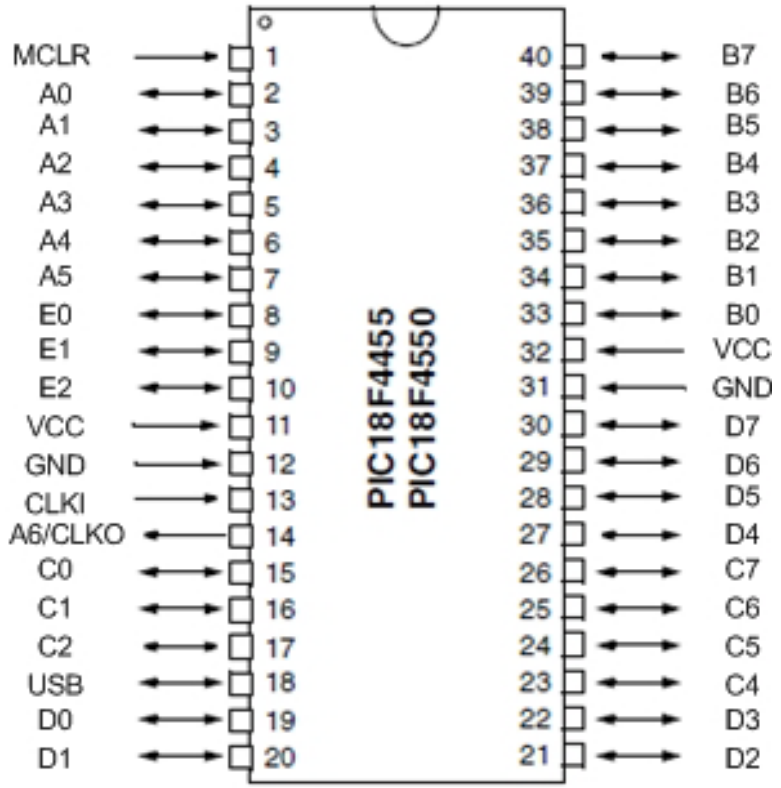
- 14 pins – power, ground, 12 I/O
- Dual Inline Package (DIP), fits into breadboards
- 4MHz default clock rate, 20MHz max
- Timers, ADC, Analog comparator

Microchip PIC18F4550



- This is a general purpose 8-bit microcontroller
- Developed by Microchip Technology Inc.
- Uses an 8-bit data bus which makes it possible to access 8 bits of data in a single machine instruction cycle.
- This microcontroller has a speed of 12 MIPS.
- This microcontroller has 40 pins.

Simplified Pin Diagram



16F87x Family: Features

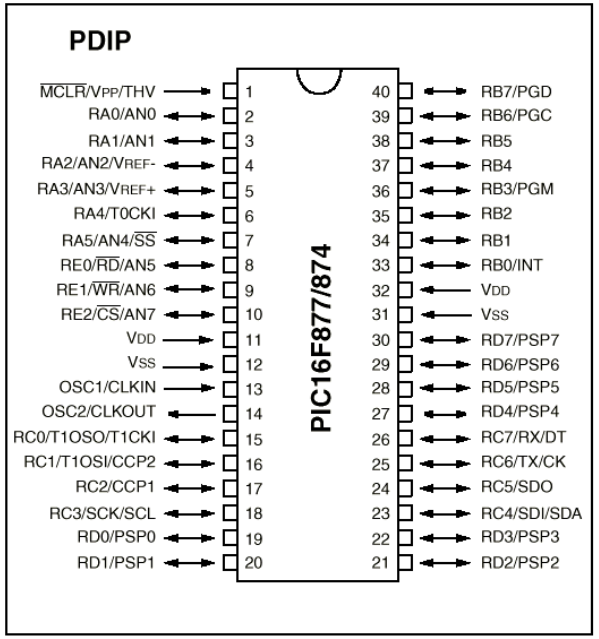
Devices Included in this Data Sheet:

- PIC16F873
- PIC16F874
- PIC16F876
- PIC16F877

Microcontroller Core Features:

- High-performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM data memory
- Pinout compatible to the PIC16C73B/74B/76/77
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and
  Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low-power, high-speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming™ (ICSP) via two pins
- Single 5V In-Circuit Serial Programming capability
- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial and Industrial temperature ranges
- Low-power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

Pin Diagram

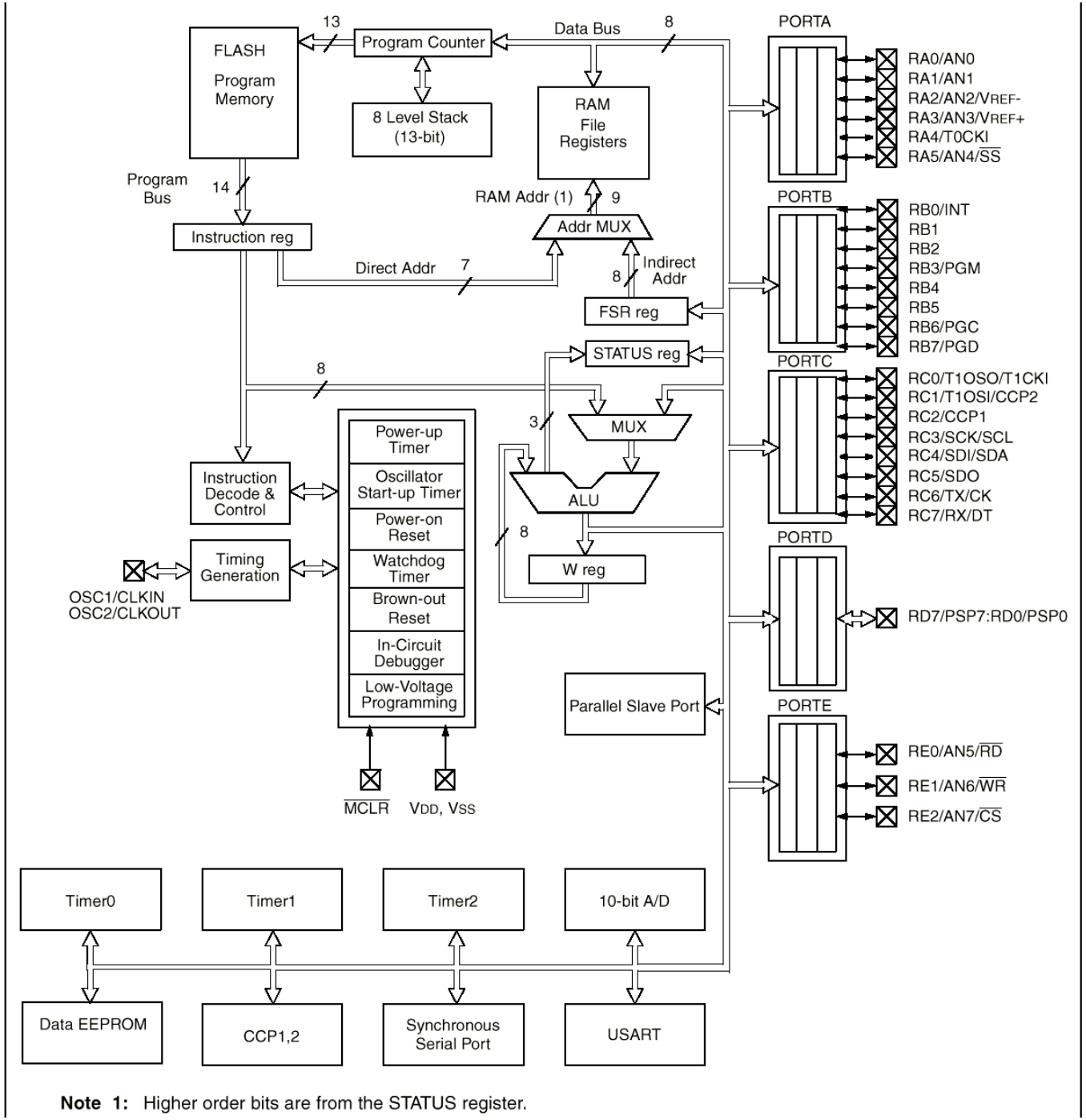


Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI™ (Master Mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

16F87x Family: Features

Typical Microcontroller Elements



Note 1: Higher order bits are from the STATUS register.

Why study the ARM architecture (and the Cortex-M in particular)?

Lots of manufacturers ship ARM products

Actel
POWER MATTERS

ATMEL

CYPRESS
PERFORM

DUST NETWORKS

ember

NXP

Texas Instruments

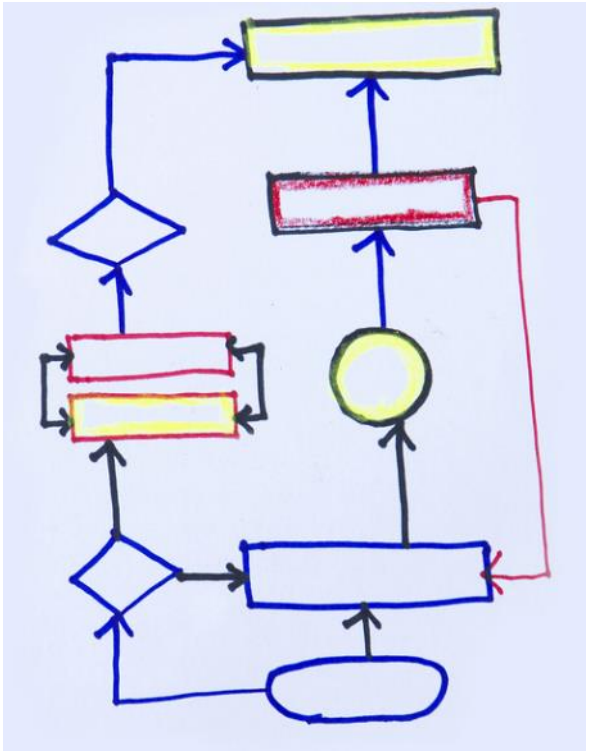
Software Translation

- Machine language: CPU instructions represented in binary
- Assembly language: CPU instructions with mnemonics
  - Easier to read
  - Equivalent to machine language
- High-level language: Commonly used languages (C, C++, Java, etc.)
  - Much easier to use

All software must be translated into the machine language of the microcontroller

Programming a PIC

- A PIC's functions are controlled by a program or code that is received via a download cable
  - Specialist software enables the program to be drawn graphically as a flowchart
  - It can also be written in a coding language, such as BASIC
  - Most PICs can be reprogrammed multiple times, allowing for corrections and updates
- What might be the drawback of using a PIC for a simple one-off task?



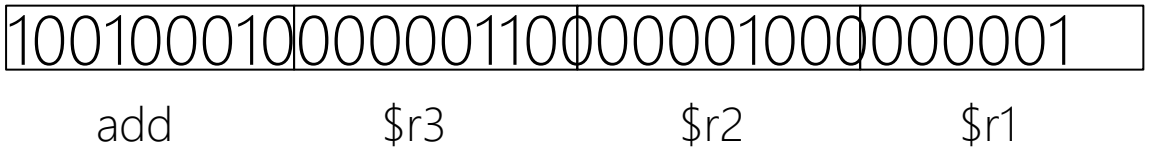
Compilation Example

a = b + c;

Compiler

```
lw $r1, ($s1)        ⟶  Load b from memory
lw $r2, ($s2)        ⟶  Load c from memory
add $r3, $r2, $r1    ⟶  Add b and c
sw $r3, ($s3)        ⟶  Store result a in memory
```

Assembler



Why C for Embedded Systems?

- Easier to use than Assembly language
- Allows more control than higher-level languages (Java, Python, etc.

C example

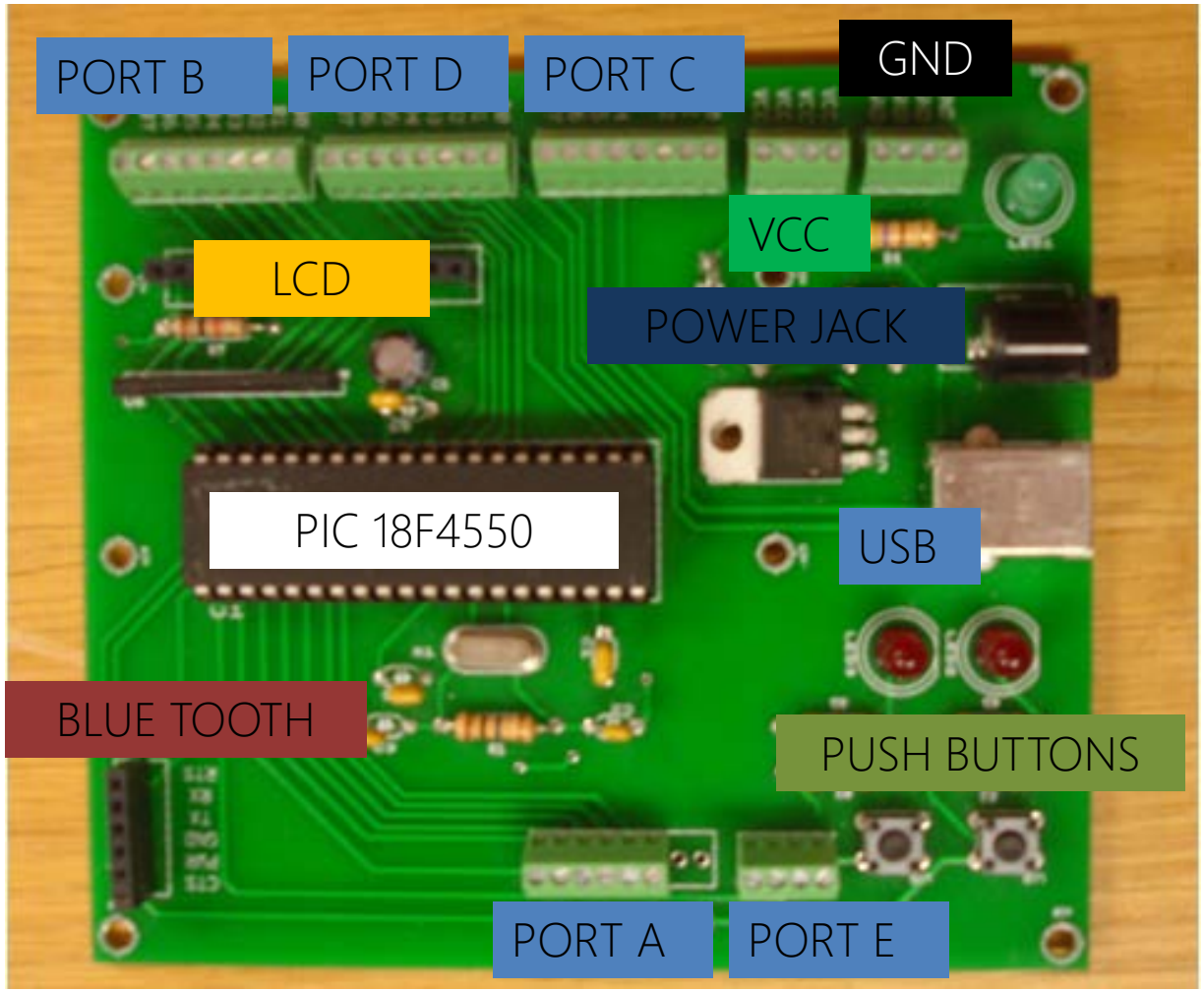
```
int a, b, c;
a = b + c;
```

Python example

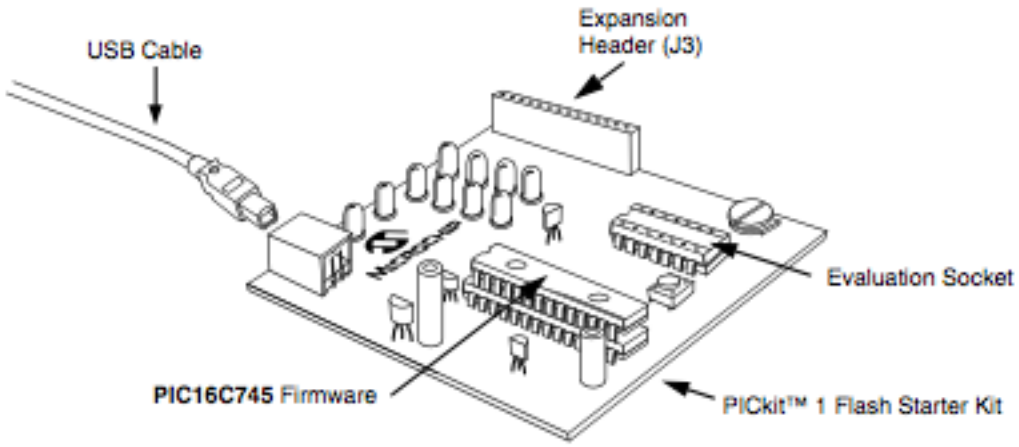
```
a = b + c
```

How much memory is used in each example?

Microcontroller development tools: Hardware



PICKIT 1 Flash Starter Kit



- Comes with a PIC16F684
- Allows programming via USB cable
- Includes 8 LEDs, pushbutton, trim pot, expansion header

Microcontroller development tools: Software

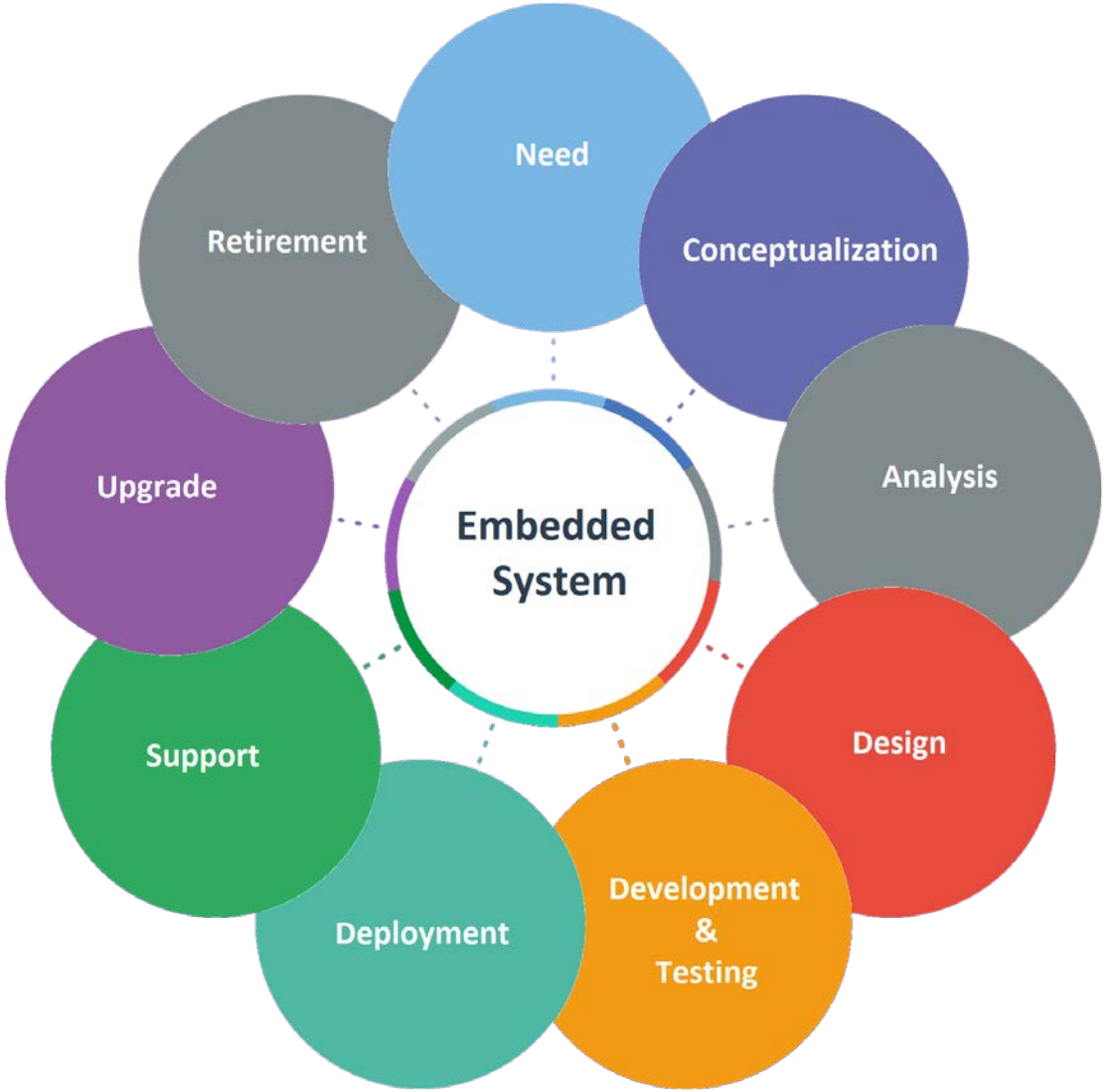
- IDE (Integrated Development Environment): A software application that facilitates all aspects developing of embedded software.
  - Source code editor.
  - A compiler.
  - A debugger.

- Software interface for loading the compiled code into the microcontroller.

How to design Microcontroller Based System?

- Controls some process or aspect of the environment: Microcontrollers Vs. DSPs
- DSPs optimized for math [multiplies]
- Embedded controller may not be a microcontroller per se but is used for special purpose control application
- Typical applications: temperature control, smart instrument, GPS, digital lock, cell phone, etc. .
- Exercise: write down five µController based devices

How to design Microcontroller Based System?



Considerations in embedded system design

- An embedded system receives input from its environment through sensors, processes this input and acts upon its environment through actuators
- Besides the usual software and hardware design issues the embedded system designer must consider the properties of the sensors and actuators and the environment itself
- The ultimate test of an embedded systems are the laws of physics

Examples

- Personal information products: Cell phone, pager, watch, pocket recorder, calculator
- Laptop components: mouse, keyboard, modem, fax card, sound card, battery charger
- Home appliances: door lock, alarm clock, thermostat, air conditioner, tv remote, hair dryer, VCR, small refrigerator, exercise equipment, washer/dryer, microwave oven
- Toys; video games, cars, dolls, etc.

Design Procedure of Microcontroller based System

- Understand design specification
  - Operating conditions, actuators, sensors, and tasks
- Select an appropriate target microcontroller
  - Need to capture main features of microcontroller from the design specification
  - Select the proper controller
- System design and test
  - Design other circuits
  - Test individual functions with circuits
  - Debug and complete programming

Ex: Design Home Energy Controller

- I live in the four bedrooms and two floors house.
- I would like to design home energy systems with following features:
  - Each room has a temperature sensor, a cooler, and a heater
  - Each room has two lights and their control switches
  - In the living room, we have a LCD monitor and home theater audio systems
  - Home energy systems need to provide user interface for setting temperature, light control, cooler and heater control
- System design and test
  - Design other circuits
  - Test individual functions with circuits
  - Debug and complete programming

Microcontroller Market

- Shipments- > 16 Billion in 2000, 8 bit > 1/2 market
- Major Players: Microchip 16Fxx, Intel 8051, Motorola MC68HC05, National COP800, SGS/Thomson ST62, Zilog Z86Cxx: 8 bit comparison

Digikey: Microcontroller

Manufacturer	Who make it?	Analog Devices Inc, Arduino, Atmel, Cypress Semiconductor Corp, Epson Electronics America Inc, and so on
Series	Group of family	568xx, 56F8xx,
Core Processor	Processor Technology	ACE1001, ARM, C28x, Z8
Core Size	Data bit and number of core	4, 6, 8, 16, 32 bit, Duel, Quad, Single, Tri-core
Speed	Clock Speed	10MHz ~ 1000 MIPS
Connectivity	Communication	Ethernet, SCI, USB, CAN, and so on.
Peripherals	Internal function blocks	LCD, PWM, WDT, DMA, Motor control
Program Memory	Memory size	

Starting point: “Microcontroller System Design”

- Hardware design
  - Evaluation hardware boards
  - Reference schematic circuit
  - Application notes
- Software
  - C compiler, assembler, and tool chains
  - Example codes, modular, and flowchart
- Debugging Tools: emulator, JTAG programmer