SJSU SAN JOSÉ STATE UNIVERSITY

Charles W. Davidson College of Engineering

Department of Computer Engineering

Real-Time Embedded System Co-Design
CMPE 146 Section 1
Fall 2024

SJSU SAN JOSÉ STATE UNIVERSITY

Introduction to Real-time Embedded System

General-purpose Computer Systems

- These are the computer systems we are familiar with
  - Laptops
  - Desktop PC’s
  - File Servers
  - Supercomputers

Embedded Computer Systems

- There are far more computer systems that are completely encapsulated by the devices they control
  - Toys
  - Microwaves
  - Refrigerators
  - Cameras
  - Cell phones
  - Tablets
  - Robots
  - Cars
  - Airplanes

  ⋮

Features of Embedded Systems

Mostly they have more constraints than general-purpose ones and have some or all of these features:

- Designed for specific purposes
  - Tasks are pre-defined to specific requirements
  - Designs can be highly optimized
- Tight time schedule
  - Tasks must be finished in a short period of time, typically in milliseconds
  - Missing the schedule may cause system failure
- High reliability
  - Need to run smoothly over a wide range of conditions (temperature, humidity, etc.)
  - Systems usually run unattended; no one there to correct the error
- High safety
  - Cannot harm the environment or users
  - Some systems are designed to control fast moving parts or high voltages

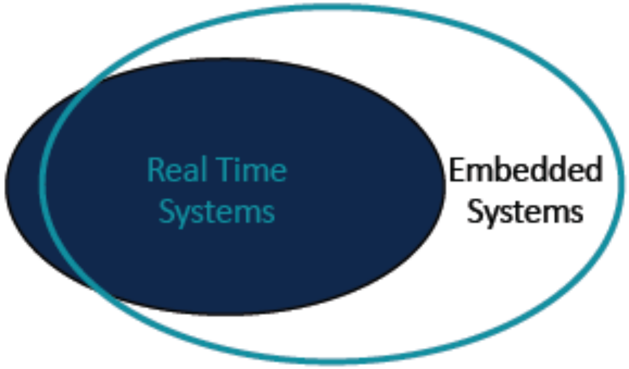
SJSU SAN JOSÉ STATE UNIVERSITY

Features of Embedded Systems (cont'd)

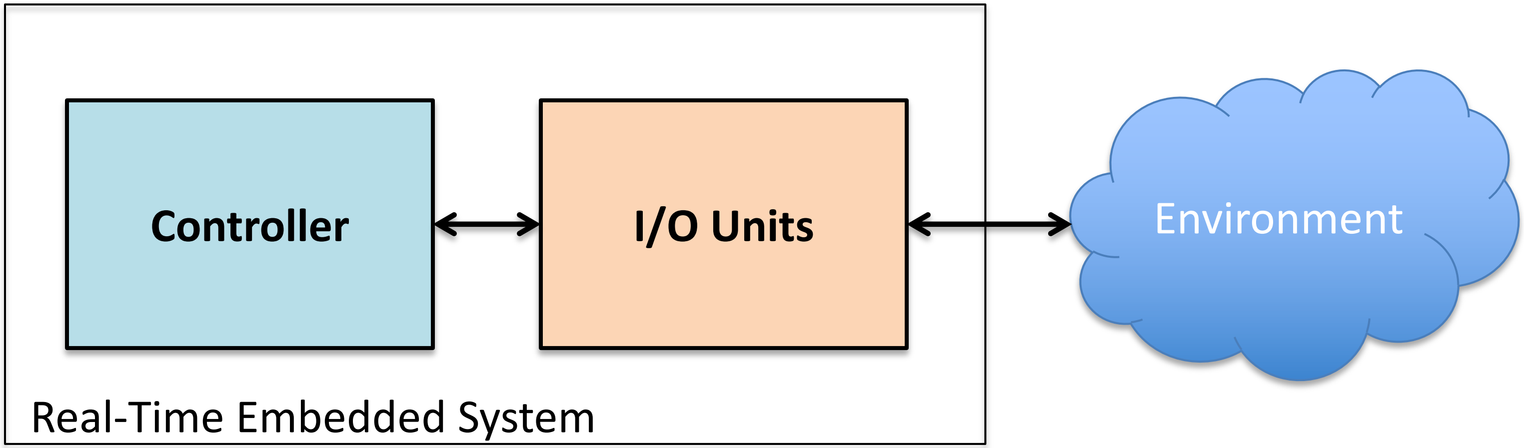
- Low cost
  - Usually quite low, from a few tens of cents to tens of dollars
- Small size
  - Usually the smaller, the better; less cost, weight, area, etc.
- High production volume
  - Often mass-produced, in hundreds of thousands to millions
  - Tiny saving in hardware cost can be multiplied by millions
- Low power consumption
  - Many are battery-powered
    - It would be very annoying to have to often replace or charge the battery
  - Some are powered by harvesting the energy from the environment
    - Sun or room light
    - Movement of device
    - Body temperature changes
    - Electromagnetic wave

Real-Time Systems

- Correctness of the system depends not only on the logical results, but also on the time in which the results are produced
  - A late answer is a wrong answer
- Correct response time depends on applications
  - Some require microseconds while for some others seconds is acceptable
- Hard real-time vs. soft real-time
  - Hard: missing deadline is system failure
    - Examples: Automobile braking system, pacemaker
  - Soft: missing deadline is system performance degradation
    - Examples: computer mouse movement, video frame decoding
- Sometimes we use the two terms “real-time” and “embedded” interchangeably, but note that they refer to different system characteristics



Real-Time Embedded System Structure



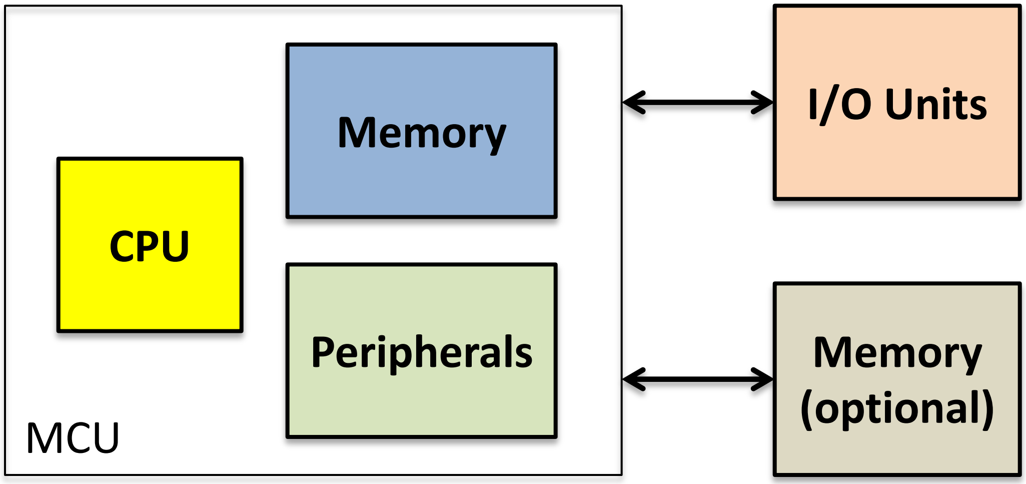
- Controller
  - Provides control logic and timing
  - Can take on different forms
    - Programmable device with instructions in memory
      - Flexible, quick time to market, higher cost
    - Hardwired logic
      - Fast execution, longer development time, lower cost, lower power
- I/O
  - Interfaces with the environment
  - Drives different types of devices, e.g., motor, heating element, display
  - Receives different types of sensing inputs, e.g., switch, light, sound

Programmable Controller

- Programmable controller design has become the dominant choice over the years
  - Due to advances in chip fabrication technology, more transistors can be integrated on a single silicon die to implement complicated control logics
  - Controllers used to be boxes or boards are now single chips
  - Controller speed has increased significantly
  - Many inexpensive and sophisticated development tools are available
  - Time-to-market is much shortened
- Most important and complex component in most embedded computing systems
  - Typically dictates the features and performance of the system
  - Selection of a controller is a very important design decision in most system developments

System Design of Interest

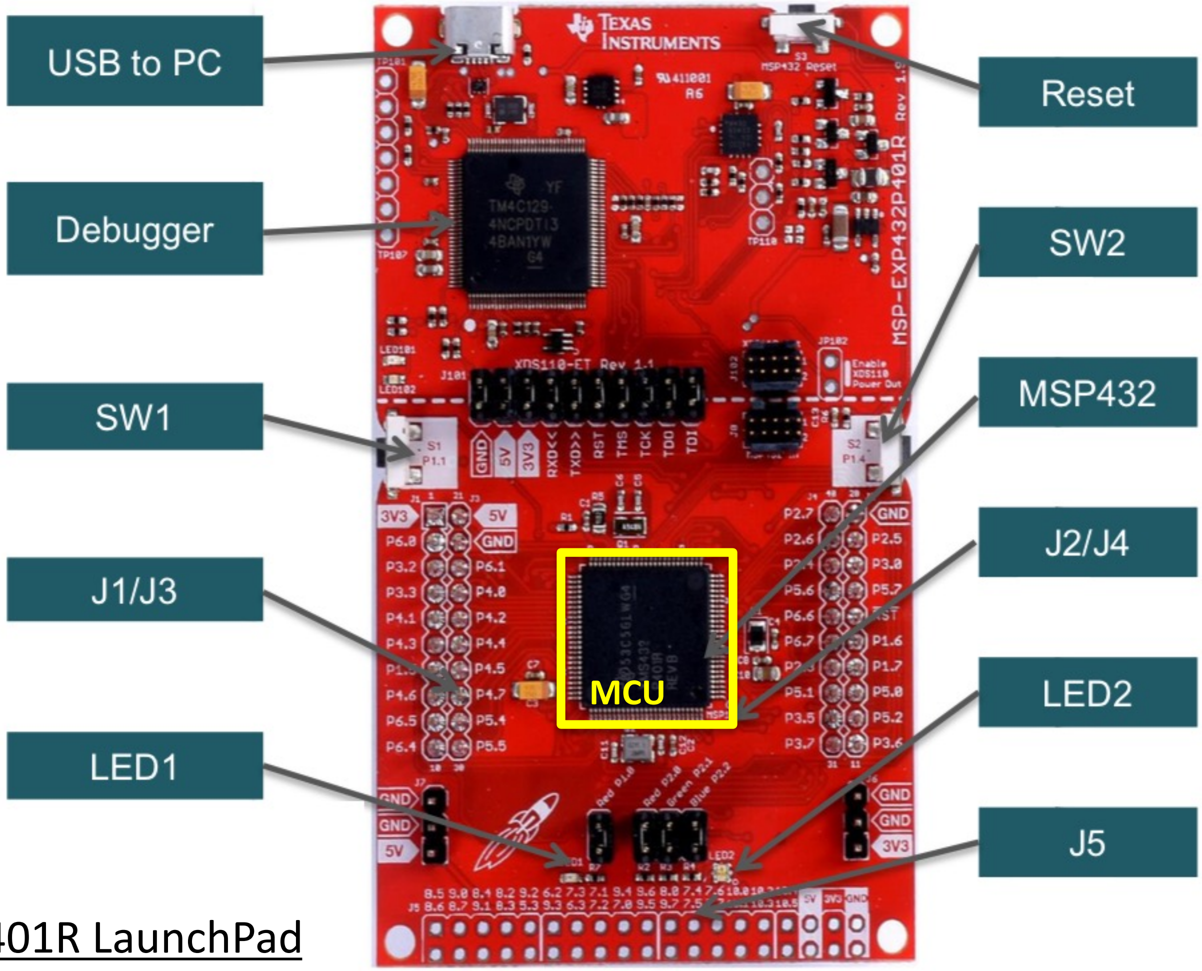
- This course will focus on designs using programmable microcontrollers
  - Most (if not all) of the control logics are contained in a single microchip
- We call such device Microcontroller Unit (MCU)
- MCU is basically a microprocessor (CPU) integrated with
  - small amount of memory for program execution
  - some common peripherals
    - interrupt controller, DMA controller, clock generator, timer, etc.



Real-Time Embedded System

Embedded System Example

- Type of systems we will discuss in this course
  - Small-board form factor
  - Reasonably high-performance MCUs



TI MSP432P401R LaunchPad

Other Embedded System Types

Besides the board-level design, we have two other types of system development, in terms of the final end product

- Chip-Level
- Chassis-Level

SJSU SAN JOSÉ STATE UNIVERSITY

Chip-Level Embedded Systems

- The end product is a fully customized single chip, not a board
- Different implementation methodology
  - Use hardware design language like Verilog or VHDL to describe the circuitry
  - Typically, all control logics are hard-wired and run in parallel
  - Tons of simulation and testing required before tape-out
- EDA (Electronic Design Automation) tools are very expensive
  - Much of the design process is highly automated
  - Very high development cost (tool, labor)
- Long time-to-market
- Can achieve much better performance in size, power and speed
- One popular alternative is FPGA (Field-Programmable Gate Array)
  - Off-the-shelf parts
  - Use same Verilog or VHDL
  - Shorter time-to-market
  - Component cost is much higher
  - Performance can be as good as custom design

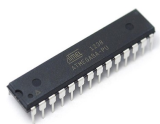
Chassis-Level Embedded Systems

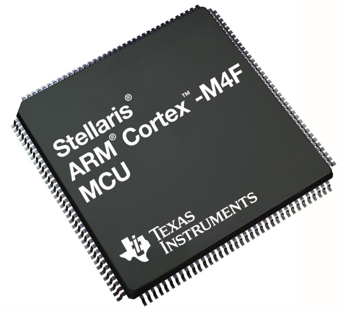
- Some embedded systems for industrial applications are big
  - Circuit boards are contained in one or more chassis, connected by a bus on a backplane

- Controllers and peripherals are contained in separate boards
- Malfunctioning boards can be easily replaced
  - To reduce down time
- Highly reliable
  - Can handle wide range of environment fluctuations
- Low production volume
- Much higher cost

MCUs

- Dozens of manufacturers
  - Microchip, Intel, Infineon, NXP, STMicroelectronics, TI, Renesas, ...
  - No one dominates the market
- Available in a wide range of features, capabilities and sizes for various applications



- Compare with SoC (System-on-Chip)
  - MCU and SoC are quite similar
  - SoC typically has all the memory and peripherals integrated on chip
    - Many include advanced peripherals like WiFi, GPS, accelerometer, etc.
  - Typically designed for one specific application or one customer
    - Most likely not off-the-shelf product like MCUs
  - SoC sometimes is also referred to as ASIC (Application Specific Integrated Circuit)

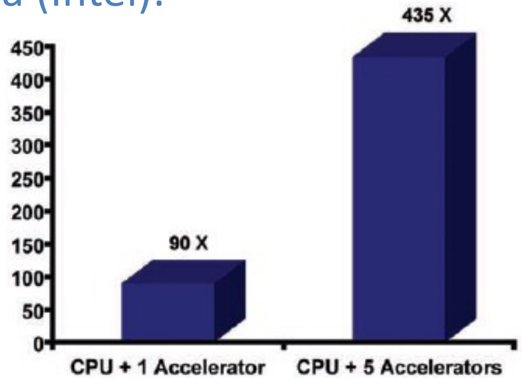
MCU Example

Texas Instruments MSP432P401R Microcontroller

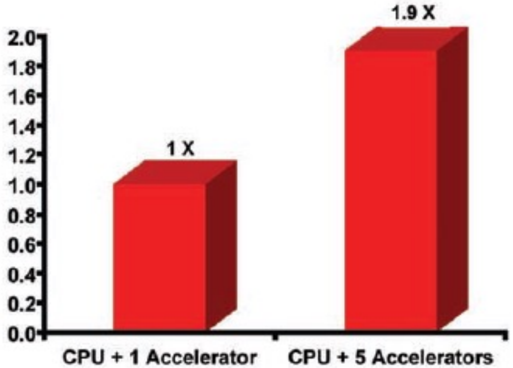
- ARM® 32-bit Cortex®-M4F CPU
- 256-KB Flash memory, 64-KB SRAM
- Floating point unit
- Memory protection unit
- Analog-to-digital converter (ADC)
- Digital-to-Analog converter (DAC)
- DMA (Direct Memory Access) controller
- Timers
- UART, SPI, I2C (serial communication protocols)
- GPIO (General-Purpose I/O)
- AES encryption/decryption accelerator
- Ultra-low-power operating modes

MCU Trends

- Leverage high-end microprocessor designs and continuing advances in chip fabrication technology
- More cores
  - Better computation performance in multitasking
- More memory
  - Non-volatile memory for instruction and data storage
  - SRAM for program execution
- Integration of accelerators
  - DSP (Digital Signal Processing), graphics, or other specific functions
  - One important way to save power and improve performance
  - One study from Altera (Intel):



Execution Speedup

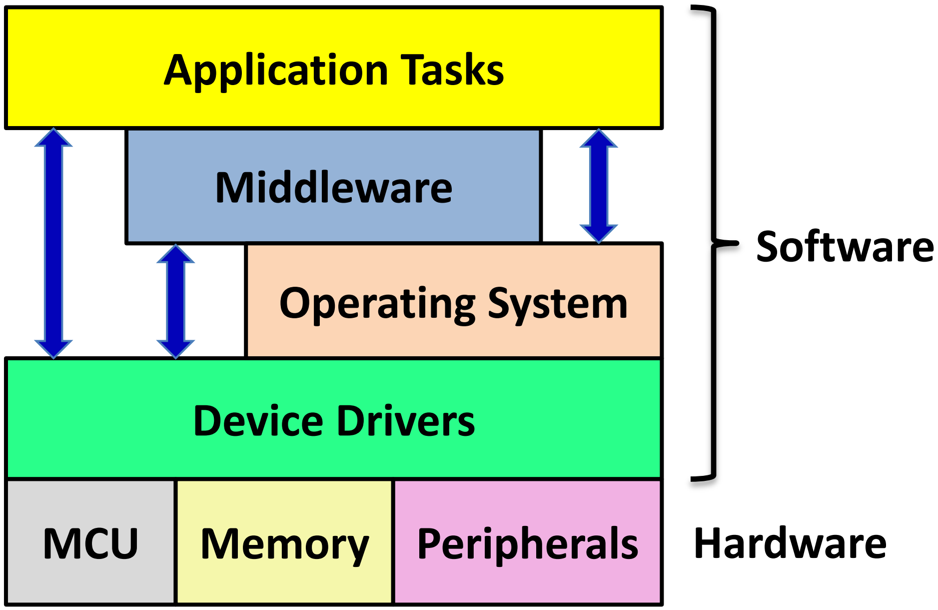


Power Consumption

MCU Trends (cont'd)

- SIMD (Single Instruction Multiple Data) instructions
  – Efficient for processing data arrays
- Power management unit
  – Provides various power modes for minimizing power consumption for different applications
- More integrated peripherals
  – USB, WiFi, Bluetooth, ...
- Addition of configurable logic block
  – Improves performance with dedicated circuitry
- MMU (Memory Management Unit)
  – Better protection in multitasking environment
  – Improves reliability
- Security
  – Random number generator
  – Encryption and decryption logics

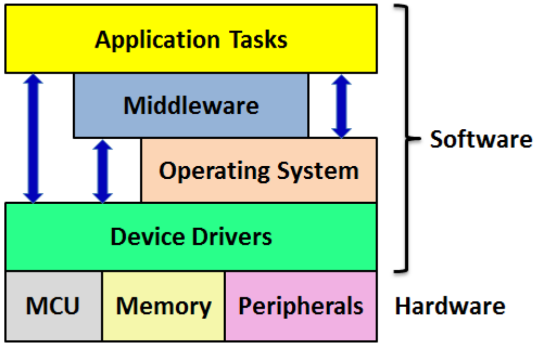
Software Architecture



- Classic layer architecture
  - The higher a layer goes, the farther from the hardware; hardware becomes more abstract
  - Clear functionality boundaries
- Middleware is like libraries
  - Contains well-defined functions/services needed by application tasks

Software Architecture (cont'd)

- Operation System (OS)
  - Suitable for concurrent programming model
    - Especially on multiple cores
  - Applications are more portable



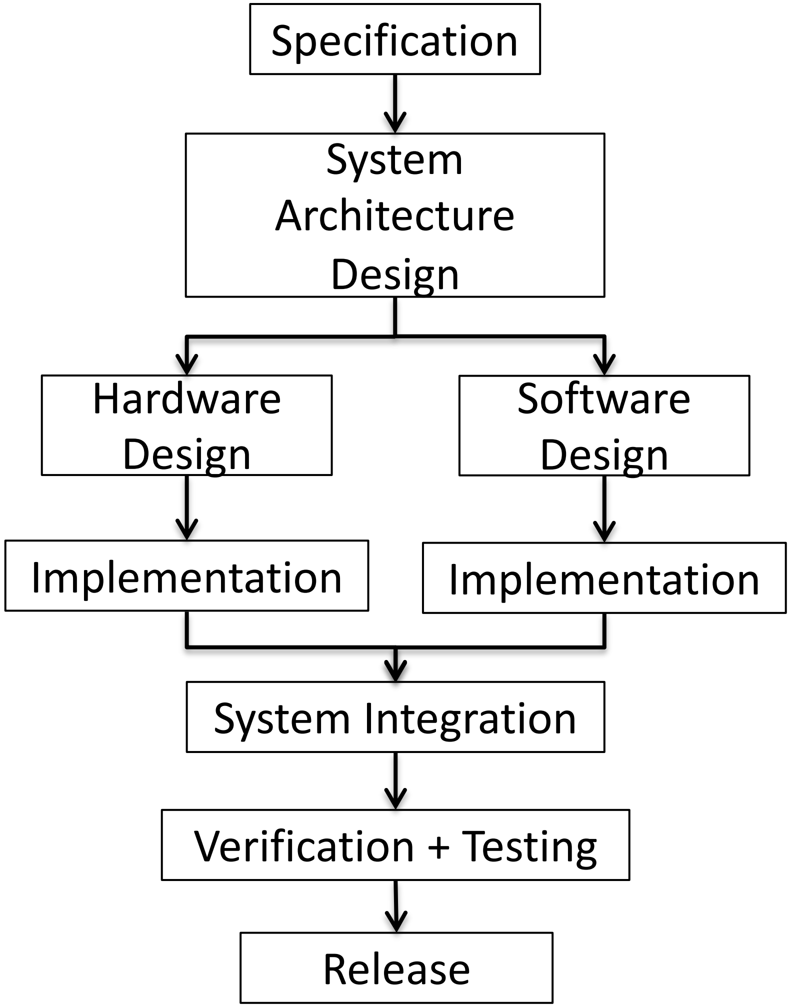
- Software resources are mostly easily accessible
  - Lots of open-sources in middleware and OS
  - Most board or MCU vendors provide device drivers for free
- For less demanding or complicated applications, the middleware and/or OS layers may not be needed
  - Simplify the software structure
  - Lose advantages of portability

Program Structure

Primarily we concern about how software tasks are executed, i.e., the scheduling

- Round-robin scheduler (superloop)
  - A forever loop at the top level to call (execute) one function (task) at a time
- Time-slice
  - Each task is allocated with a fixed amount of time to execute
- Task queues
  - Two queues: ready and waiting
  - Ready queue has a list of tasks to execute
  - Waiting queue has a list of tasks waiting for a trigger event to execute
- Wake-up by interrupts
  - High-priority tasks are waken up by dedicated interrupt sources
  - Works along with other scheduling schemes
- RTOS
  - Rely on a real-time operating system to do the scheduling and synchronization

System Development Process



SJSU SAN JOSÉ STATE UNIVERSITY

Performance Metrics

There are so many. Each application has its own set.

- Power consumption
  - Energy the system consumes per unit time
- Size
- Response time (delay)
  - How fast the system can respond to change of inputs
- Throughput
  - After an initial delay, how fast the system can produce the results
- Unit cost
  - Monetary cost of manufacturing one unit of the system
- NRE (Non-Recurring Engineering) cost
  - One-time monetary cost of designing the system
- Time-to-market
  - Time required to develop a system to the point that it can be sold

Performance Metrics (cont'd)

- Maintainability
  - Ability to service the system
- Time-to-prototype
  - Time needed to build a working version of the system
- Flexibility
  - Ability to change the functionality of the system without incurring heavy NRE cost
- Manufacturability
  - How easy the system can be manufactured
- Correctness
  - How accurate are the results produced by the system
- Safety
  - How safe is the system to the users or environment
- Testability
  - How easy can the system be tested

SJSU SAN JOSÉ STATE UNIVERSITY

New Challenges

- Characterization of response time
  - It is getting harder to predict
  - MCUs are getting more sophisticated
    - Many cores, memory cache interactions, out-of-order execution, branch prediction, etc.
  - Applications become more complex
    - More tasks are involved, creating complicated interactions
- Software tools
  - More accelerators and power-saving hardware features are becoming available
  - Need more sophisticated tools to make use of them seamlessly
- Security
  - Becoming a serious issue as connectivity has increased drastically over the years
  - Built-in security features are no longer a design afterthought

Things to explore in this course

- We are not designing any circuits
- Understanding in detail how hardware works at system level
  - MCU architecture, peripherals, memory system, bus protocols, interfaces, etc.
  - A lot of reading on datasheets and manuals
- Practical software design and implementation
  - Concurrent programming with and without OS
  - Make the most of MCU features
- Optimization techniques
  - Power, program execution