# Microprocessor vs. Microcontroller

CSE0420 – Embedded Systems

By

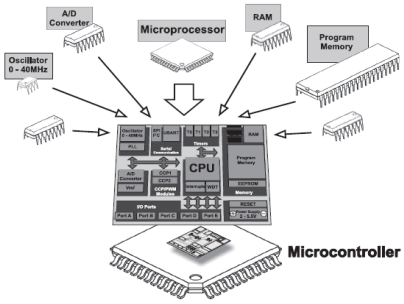
Z. Cihan TAYŞİ

## Outline

- Microprocessor vs. Microcontroller
- Components of microcontrollers
  - Digital I/O
  - Timers
  - Communication buses,
  - and so on…
- Programming schemes
- Peripherals
- Project ideas

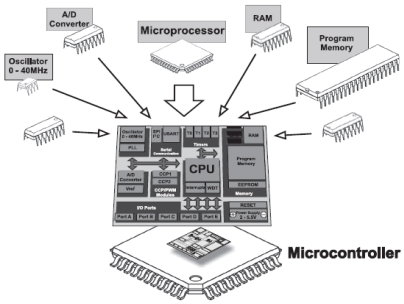
# Microprocessor vs. Microcontroller

- To summarize, a microcontroller is a (stripped-down) processor which is equipped with
  - memory,
  - timers,
  - I/O pins,
  - communication buses,
  - and other on-chip peripherals.




# Microprocessor vs. Microcontroller

- The driving element behind all this is **cost**;
  - Integrating all elements on one chip saves space
  - Also leads to both lower manufacturing costs and shorter development times.
- This saves both **time and money**, which are key factors in embedded systems.



# Microprocessor vs. Microcontroller

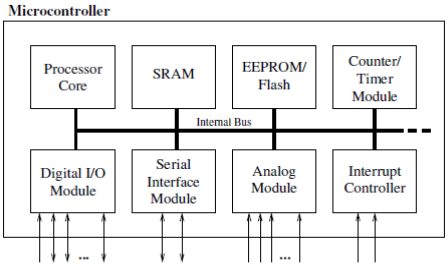
- Additional advantages of the integration are
  - easy upgradability,
  - lower power consumption,
  - and higher reliability,


- **On the downside,**
  - using a microcontroller to solve a task in software that could also be solved with a hardware solution **will not give** you the same speed that the hardware solution could achieve.

# Introduction to Microcontrollers

- **Components**
  - Core, memory, digital I/O, etc.
- **Communication interfaces**
  - UART, SPI, I2C, etc.
- **Programming scheme**
  - Howto program, download, debug,
- **Peripherals**
  - switches, buttons, displays, leds, so on.

# Microcontroller Components




- All components are connected via an internal bus and are all integrated on one chip.
- The modules are connected to the outside world via I/O pins.

# Microcontroller Components

- Processor Core
  - The CPU of the controller. It contains the arithmetic logic unit, the control unit, and the registers (stack pointer, program counter, accumulator register, register file, ...)
- Memory
  - The memory is sometimes split into program memory and data memory.
  - In larger controllers, a DMA controller handles data transfers between peripheral components and the memory.
- Interrupt Controller
  - Interrupts are useful for interrupting the normal program flow in case of (important) external or internal events. In conjunction with sleep modes, they help to conserve power.

# Microcontroller Components

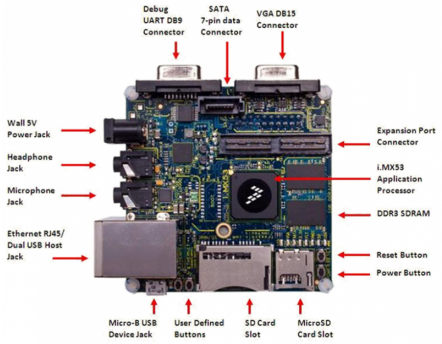
- Interrupt Controller
  - Interrupts are useful for interrupting the normal program flow in case of (important) external or internal events. In conjunction with sleep modes, they help to conserve power.
- Digital I/O
  - Parallel digital I/O ports are one of the main features of microcontrollers. The number of I/O pins varies from 3-4 to over 100, depending on the controller family and the controller type.

# Microcontroller Components

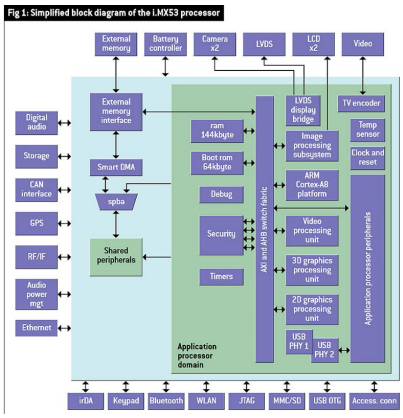
- Timer/Counter
  - Most controllers have at least one and more likely 2-3 Timer/Counters, which can be used to timestamp events, measure intervals, or count events.
  - Many controllers also contain PWM (pulse width modulation) outputs, which can be used to drive motors or for safe breaking (antilock brake system, ABS). Furthermore the PWM output can, in conjunction with an external filter, be used to realize a cheap digital/analog converter.

# Microcontroller Components

- Analog I/O
  - Apart from a few small controllers, most microcontrollers have integrated analog/digital converters,
  - They differ in the number of channels (2-16) and their resolution (8-12 bits).
  - The analog module also generally features an analog comparator.
  - In most cases, the microcontroller includes digital/analog converters.




# Microcontroller Components




- Interfaces
  - Controllers generally have at least one serial interface which can be used to download the program and for communication with the development PC in general.
  - Since serial interfaces can also be used to communicate with external peripheral devices, most controllers offer several and varied interfaces like ***SPI and SCI***.
  - Many microcontrollers also contain integrated bus controllers for the most common (field)busses. ***I2C and CAN*** controllers lead the field here. Larger microcontrollers may also contain ***PCI, USB, or Ethernet*** interfaces.

# Microcontroller Components

- Watchdog Timer
  - Since safety-critical systems form a major application area of microcontrollers, it is important to guard against errors in the program and/or the hardware. The watchdog timer is used to reset the controller in case of software "crashes".
- Debugging Unit
  - Some controllers are equipped with additional hardware to allow remote debugging of the chip from the PC. So there is no need to download special debugging software, which has the distinct advantage that erroneous application code cannot overwrite the debugger

# Programming on Microcontrollers

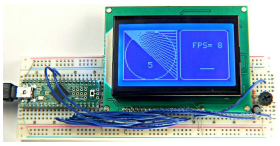
- Bare machine / Bare metal
  - system boots directly into monolithic, single-purpose software, without loading a separate operating system
  - usually written in assembly and/or C
- Board Support Package
  - A board support package (BSP) is essential code for a given computer hardware device that will make that device work with the computer's OS.
  - The BSP contains a small program called a boot loader or boot manager that places the OS and device drivers into memory.
  - The contents of the BSP depend on the particular hardware and OS.
- Application on a operating system
  - Industrial PC, Arm based-microcontrollers, etc.
  - RTOS,Linux, Windows Embedded

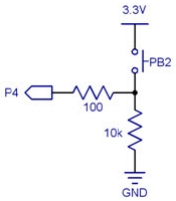
# Programming on Microcontrollers

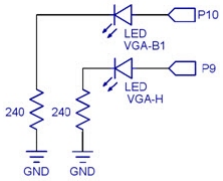
- Programming is done mostly on a PC
  - a special editor / IDE
- Cross compilation is required.
  - building an executable for a different architecture
- Transfer binary code to microcontroller
  - A programmer is required.
- Debugging
  - no printf!!
  - peripherals (leds, communication channels)
  - remote debugging

# Peripherals

- Displays
  - 7 segment
  - dot matrix lcd
- Buttons
- Leds





# Peripherals

- Sensors
  - pressure
  - temperature
  - light
  - humidity
  - …
- Actuators
  - relays
  - servo
  - step motor
  - …



# Projects – I

- Phases
  - Draft proposal
    - a basic product specification
    - just tell us what you are planning to build and features that it will have!
  - Detailed proposal
    - HW/SW partitioning and system design
    - explain which hardware you will use, and give your reasons
    - explain how you going to implement these features
  - Development
    - Iteration & implementation, detailed design, integration
  - Presentation
    - you will hand out a report
    - Also make a presentation of your project including a demo!

## Projects – II

- Draft proposal – 5th week (**next week**)
- Detailed proposal – 6th week
- Development – till your presentation time ☺
- Presentations – starts from 10th week

- Proposals and the report should be printed about and signed by the students!

## Projects – III

- Example project
  - a smart pod
- Draft report – **just an example!!!**
  - I will build an embedded system to be used in an existing pod to monitor the environmental conditions of a flower. The system will monitor light, humidity, temperature and conductivity in the soil and report these info to a PC through wi-fi connection.
  - There will be a control software in the PC. It will check the condition of the soil and generate alarms, if conditions do not meet requirements of the flower in the pot. The control software will support multiple smart ports. Identification of the pots will be done by blink of a led on each pot.

# Projects – IV

- Which hardware to use !?
  - it will be defined by specification of your system.
- What we have…
  - Almost everything, but it is **limited in numbers**
  - You can **always ask for help** about hardware
- Development boards for several microcontrollers
  - 8-bit/16-bit/32-bit
  - Atmel atmega
  - Texas Instruments MSP series
  - ARM based board; Friendly arm, i.Mx53 boards
  - Ardunio
- Special embedded hardware
  - ESP8266 – wi-fi boards