SJSU SAN JOSÉ STATE UNIVERSITY

Charles W. Davidson College of Engineering

Department of Computer Engineering

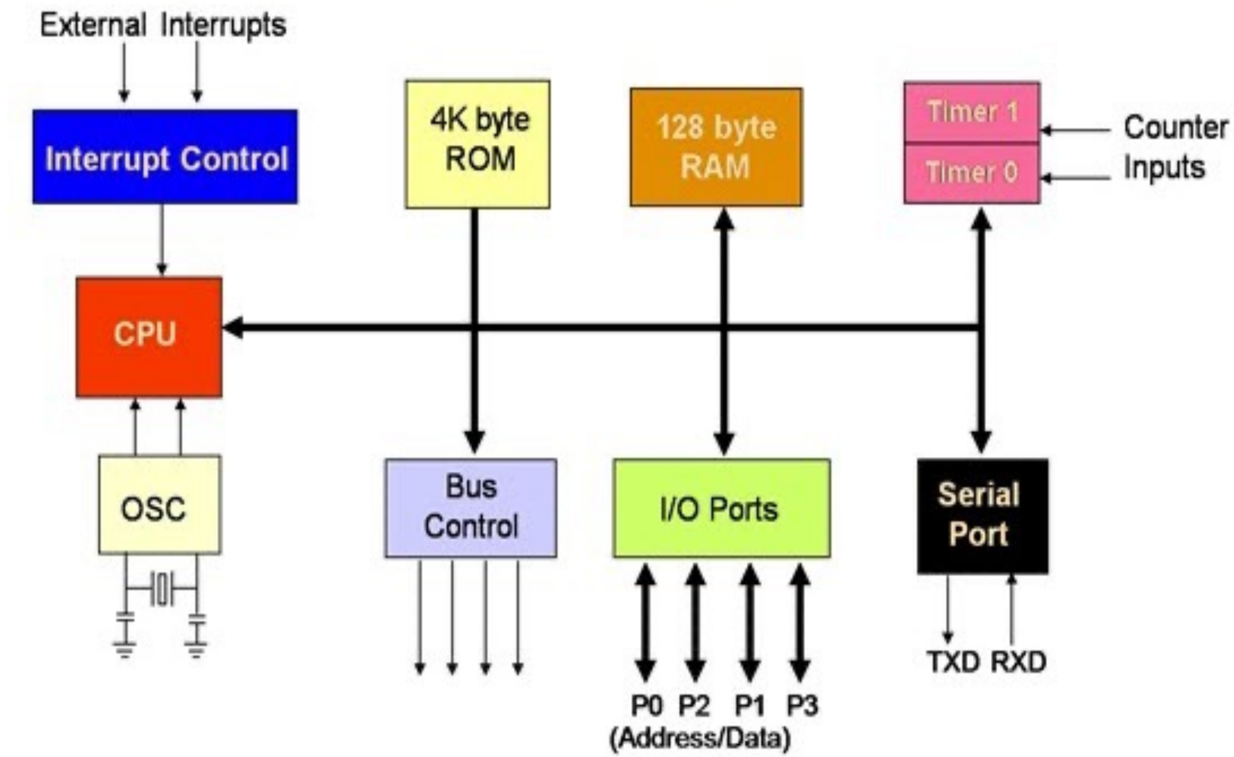
Real-Time Embedded System Co-Design
CMPE 146 Section 1
Fall 2024

MCU Architecture

SJSU SAN JOSÉ STATE UNIVERSITY

Basic MCU Architecture

Intel 8051 8-bit MCU Architecture



- Maximum operating frequency: ~20 MHz
- Single bus
- A few peripherals

Diagram source: https://www.elprocus.com/8051-microcontroller-architecture-and-applications/

Advanced MCU Architecture

- High-performance processor core
  - Can run at hundreds of MHz
- Two-level bus infrastructure
  - High-speed system bus
  - Lower-speed peripheral bus
- Many peripherals
- Debug interface
  - Also for profiling
- Clock generator
  - Multiple outputs
  - Highly programmable

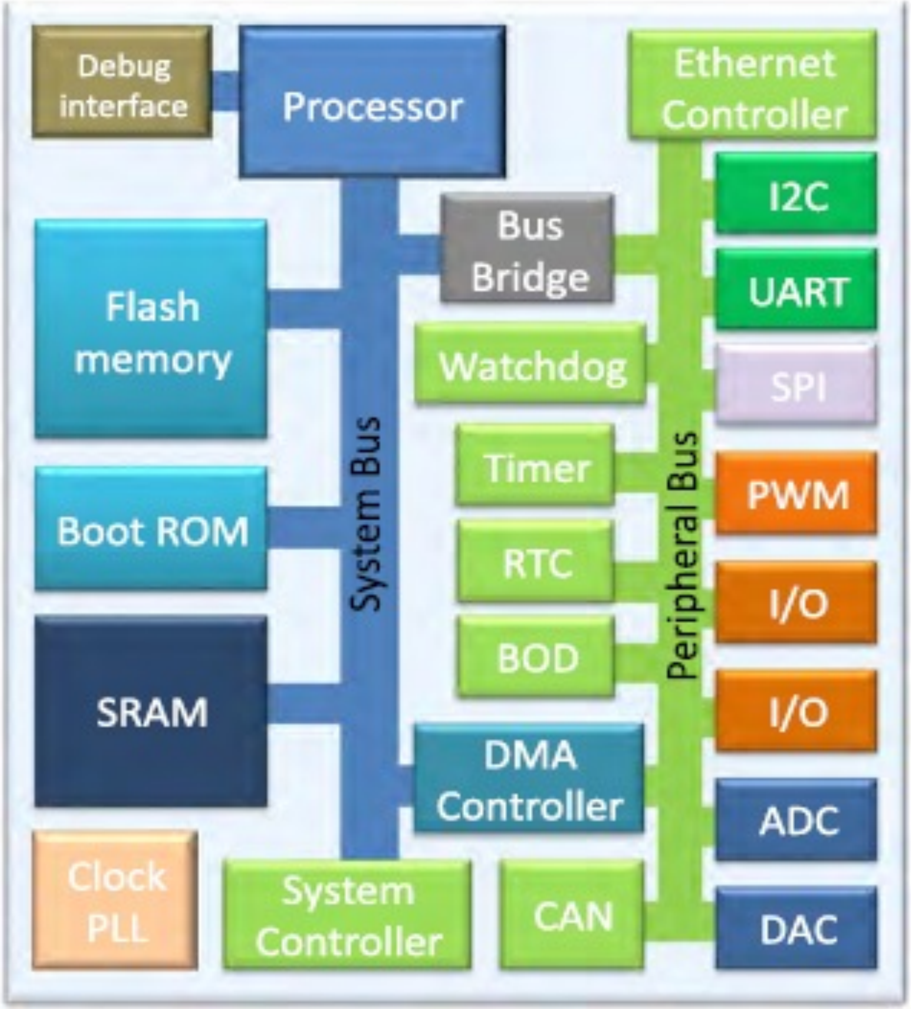


Diagram source: Joseph Yiu, The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors, third edition, Elsevier, Inc., 2014.

Features for Embedded Applications

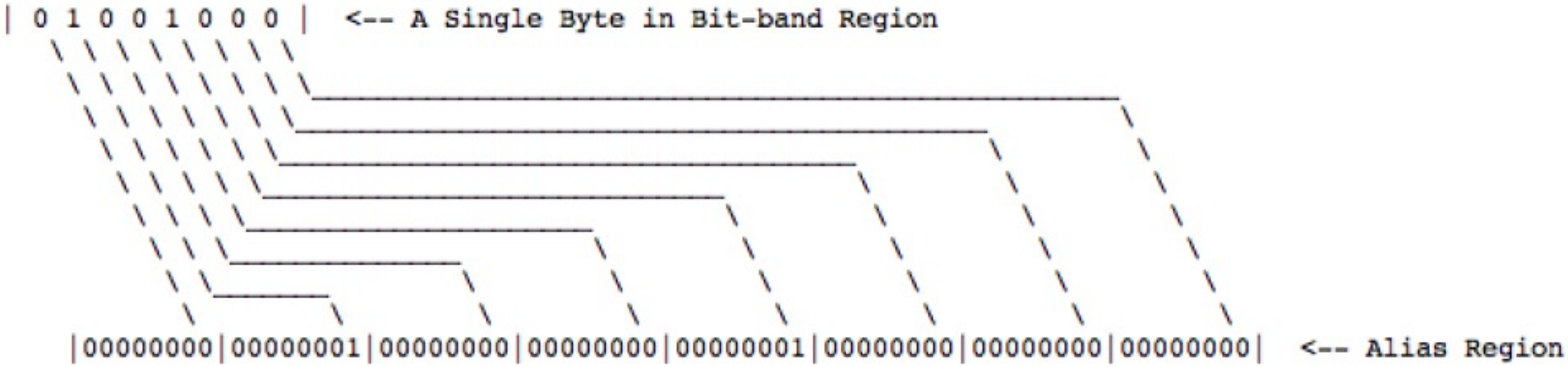
- Simpler processor core
- Memory supports of different types
- Low power consumption
- Lots of interrupt inputs
- Plenty of peripherals
- Security
- Clock generation

Processor Core

- RISC (Reduced Instruction Set Computer) design
  - Load-store architecture
    - Separate instructions for memory access and computation
  - Short fixed length instructions (mostly)
  - Large register file
    - Frequent memory access can be costly
  - Single-clock-cycle instructions (mostly)
- Harvard architecture
  - Separate instruction and data buses
  - Improves execution performance
- Simple and short pipeline
  - For example, ARM Cortex-M3/M4 processors have 3 stages
    - High-end processors have many more stages
  - Less circuitry
- Multiply-Accumulate (MAC) instructions for digital signal processing (DSP)
  - One instruction, two operations

Processor Core (cont’d)

- Special bit-access operations
  - Special instructions for specific operations (in some processors)
  - Bit-Banding feature
    - In ARM Cortex-M3/M4 processors
    - Words in an alias region are mapped to individual bits in a memory region
    - Atomic operations
    - C friendly; no special instruction for compiler to deal with



On-Chip Memory

- Flash memory
  - Program and data storage
  - Non-volatile, programmable (writable with special circuitry)
- ROM
  - Non-volatile, read-only, non-programmable
  - Typically holds
    - System boot loader
    - Flash programming instructions
    - Built-In Self Test (BIST)
    - On-chip peripheral drivers
- SRAM
  - Volatile, readable, writable
  - Provides stack space
  - Can be used as memory cache
  - Typically holds
    - Working program data
    - Instructions (for debugging or better performance)

Low Power Consumption

- Simple RISC core
  - Short pipeline, small instruction set, few advanced features
    - Less circuitry: fewer transistors, smaller area
  - Better code density
    - More power-efficient
- Power consumption of a CMOS circuit can be approximated as

$$P \approx \alpha\, C_L\, V_{DD}^2\, f$$

  where α is the switching activity, C_L is the load capacitance, V_{DD} is the supply voltage and f is the operating frequency.
  - Therefore, in real time, we have two parameters V_{DD} and f to tweak
- Frequency adjustment
  - Put system to sleep; stop the clock when there is nothing to do
    - There is delay penalty when waking up
  - Slow down frequency when speed is not needed

Low Power Consumption (cont'd)

- Voltage adjustment
  - Turn off components when they are not needed
    - There is a delay penalty when need to turn them back on
    - Also reduces the leakage current
  - A good way to add more circuitry without incurring too much power
- Accelerator integration
  - Do things with dedicated hardware, not with CPU instructions
  - Fast, energy-efficient
  - There is communication overhead

Interrupts

- Able to handle many internal and external interrupt sources
  - Can be more than a hundred in total
  - Low interrupt latency
    - For example, ARM Cortex-M3/M4 processors take 12 cycles

- Let processor sleep most of the time and be waken by interrupts
  - Constant polling
    - Wastes energy
      - Input states usually don’t change most of the time
    - May miss short events if polling frequency is not high enough
    - Creates unpredictable latency
      - More input signals to poll, more unpredicitabe
      - Could be much worse than using interrupt
  - Peripherals can generate interrupt when service is needed
  - Change of state in I/O pins can generate interrupt to wake up processor
- Abnormal operations (exception) can also generate interrupts
  - No need to actively check for errors by software

Security

- What must be protected
  - Code
    - Could be reverse-engineered to gain insights on how things work
    - Could be modified
    - Could be cloned to counterfeit systems
  - Data
    - Confidential personal information could be stolen
  - Functionality
    - System malfunction could occur if system is under attack
- What can be done
  - Secure communication channels
  - Secure boot process to ensure nothing has been modified
  - Monitor activities
  - Encrypt sensitive data
- Hardware support
  - Random number generator
  - Checksum generator
  - Encryption/Decryption accelerator

Clock Generation

- On-chip clocks can be generated by
  - Fixed oscillator module
  - Programmable phase-locked loop (PLL) module
- It is all about lowering power consumption
  - System clock rate can be from 0 to hundreds of MHz
- Different peripherals can have their own operating frequencies depending on the application
  - Each module may have its own clock signal
  - Programmable clock dividers provide different frequencies