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

On-chip Interconnect

- More and more components are integrated on chip
  - A communication infrastructure is warranted for connecting large number of components effectively
- Takes a long time to develop every component from scratch
  - Integrating existing components, generally referred to as IP (Intellectual Property) blocks, can have a lot of savings in design, testing and verification
  - Need an agreement on how the IP blocks connect together
- Advanced Microcontroller Bus Architecture (AMBA) is a popular solution
  - Specification developed by Arm (a UK company)
  - Has become a standard for interfacing on-chip components
- Main features
  - Designed for a wide variety of components
    - Processor cores, co-processors, accelerators, peripherals
  - High performance
  - Low power consumption

AMBA

- Two bus specifications for embedded applications of our interest
  - Advanced High-Performance Bus (AHB)
  - Advanced Peripheral Bus (APB)
- A basic Cortex-M3/M4 processor based system:

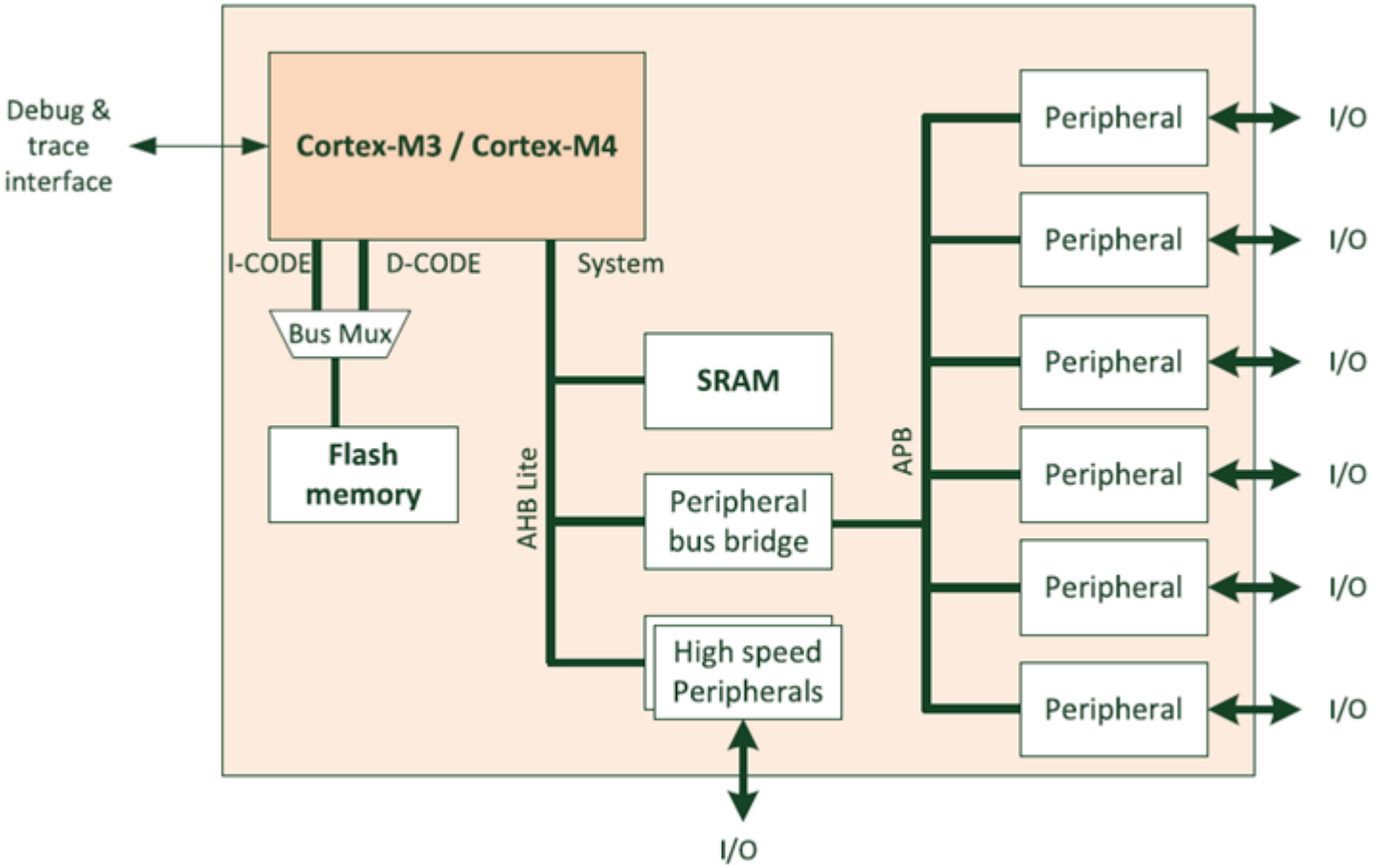


Diagram source: Joseph Yiu, The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors, third edition, Elsevier, Inc., 2014.

AMBA (cont'd)

- Advanced High-Performance Bus (AHB)
  - Designed for high performance
  - A pipelined bus
    - Address, Data and Response stages can overlap in time
  - Multiple masters
    - Essential for systems with multiple processor cores or co-processors
  - Burst transfers
    - Essential for transferring a block of data efficiently
  - Split transactions
    - For slow target devices, transaction can be split up to allow new transaction to take place
  - Wider data bus configurations, i.e., 64, 128 bits
- Advanced Peripheral Bus (APB)
  - A simplified AHB for low-frequency system components
  - Consists of a single bus master called the APB bridge; peripheral devices on the APB are slaves
  - Lower power consumption than AHB
    - Runs at lower clock rate

Numeric Computation Units

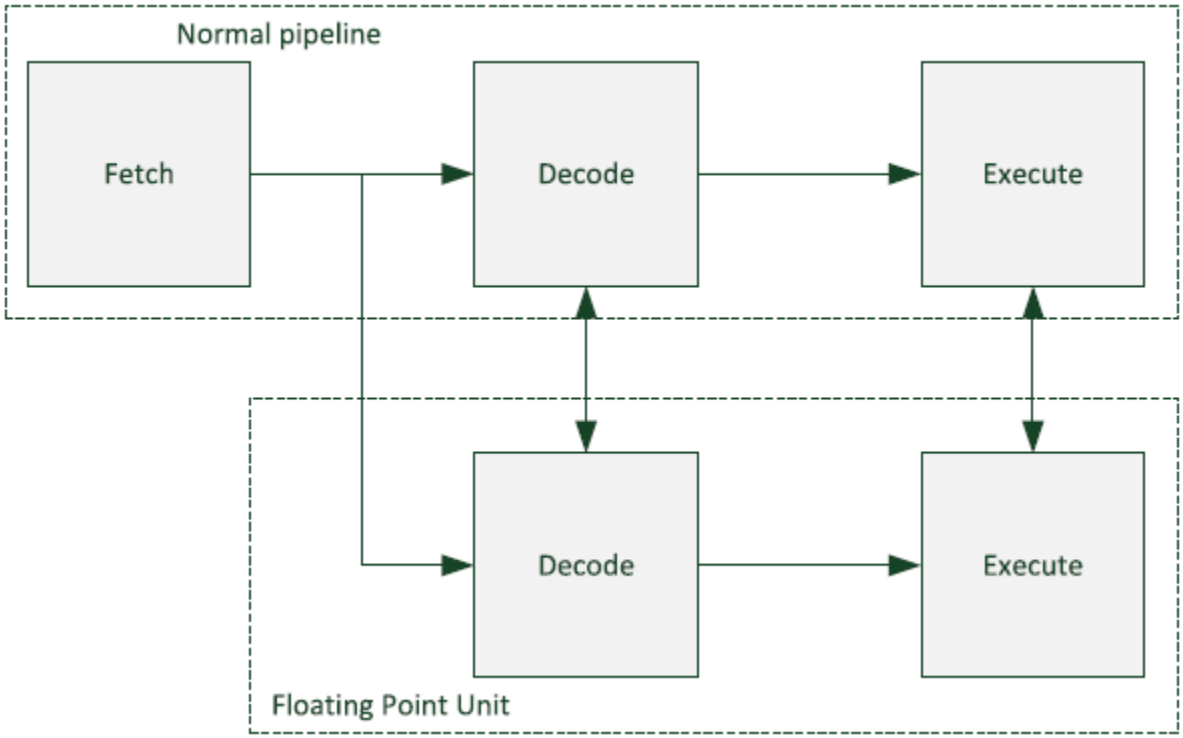
- Two types: Internal and external to the processor core
- Internal to the processor core
  - Tightly integrated with the core
    - Minimal connection overhead
  - Execute with native processor instructions
  - Components
    - ALU
    - Floating Point Unit
    - SIMD Vector Processing Unit

Numeric Computation Units (cont'd)

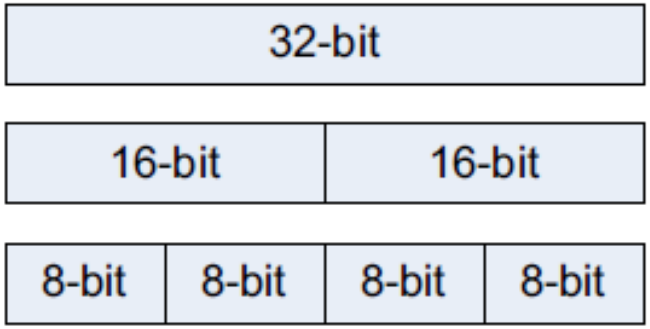
- External to the processor core
  - Attached to the interconnect infrastructure
  - Communication overhead could be significant
    - Data need to be transferred
    - Synchronization with master processor is necessary
  - Some examples:
    - Processor cores, same as the master core
      - For general-purpose computations
      - Independent instruction streams
    - Co-processors
      - For specific types of computations (e.g., network, I/O)
      - Different architecture from main processor core
      - Independent instruction streams
    - Accelerators
      - Specialized functional unit (e.g., checksum generation, encryption/decryption)
      - No program execution (typically)

Floating Point Unit

- Tightly coupled in the core
  - Floating-Point instructions are in the regular instruction stream
  - Separate pipeline but share the same Fetch stage (e.g., on Cortex-M4)
  - Access through a separate register file
- Can be disabled to save power (e.g., on Cortex-M4)



SIMD Processing Unit



- One single instruction can process multiple data items
  - For example, 32-bit processor can process two 16-bit or four 8-bit data items simultaneously
- Tightly coupled in the core
  - SIMD instructions are in the regular instruction stream
  - In the same pipeline as the ALU
  - Access through normal register file
- Well suited for DSP applications
- Needs high-performance memory system to stream in data to the CPU at high speed

Multiple Cores

- For general-purpose computations
- Improve overall system performance with more slower cores
  - Keep clock frequency manageable
  - Spread out workload
    - Reduce local heat density on chip
- Well suited for multi-thread applications
- Software development complexity increases
  - Parallel tasks defining
  - Load balancing
  - Task synchronization
  - Testing and debugging
- Commonly communicate through shared memory
- Typically need a formal OS to manage

Co-Processors

- Offload specific time-consuming tasks from main CPU
  - For example, graphics processing, network protocol handling, I/O management, signal processing, cryptography operations
- May have their own (limited) instruction set
  - For example, GPU

Accelerators

- Handle specific computation-intensive functions
  - For example, FFT (Fast Fourier Transform), DCT (Discrete Cosine Transform), encryption/decryption
- No program to execute
  - Use custom logic
- High performance, low power
  - Custom logic can perform operations much faster than master processor with less energy
  - Can be turned off when idle
- Setup to execute can be quite simple
  - Typically through just a few control registers

Accelerator Example 1

Cyclic Redundancy Check (CRC) checksum generation

- Error-detecting code commonly used in data communication or storage to detect accidental or intentional change in data
- Compute a CRC signature (checksum) of a block of data
  - Checksum = data modulo-2 divisor
  - For example, the 32-bit CRC32-ISO3309 standard uses divisor (polynomial): $f(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
  - Attach the checksum to the block of data
  - For verification, re-compute the checksum again and compare with the stored checksum
- Can be done very quickly on the MSP432 MCU with small amount of circuitry
  1. First initialize the result register with a “seed”
  2. Copy data into the data register of the CRC accelerator
  3. Read the signature with one cycle delay

Accelerator Example 2

Advanced Encryption Standard (AES) encryption and decryption

- A widely used standard
  - Also adopted by the U.S. federal government for classified information
- Based on a design principle known as a substitution–permutation network
  - Iterates over 10-14 rounds
- A symmetric-key algorithm
  - the same key is used for both encrypting and decrypting the data
- Encrypts the data in a block-by-block basis with a long bit-length key
  - Block size can be 128, 192 or 256 bits long
  - Key size can also be one of those lengths
- Can be done quickly on the MSP432 MCU in ~200 cycles
  1. Set up the key in the key register
  2. Write data into the input data register
  3. Wait for the busy flag to be cleared in the status register
  4. Read data from the output data register