ECE3411 – Fall 2015

Lecture 7a.

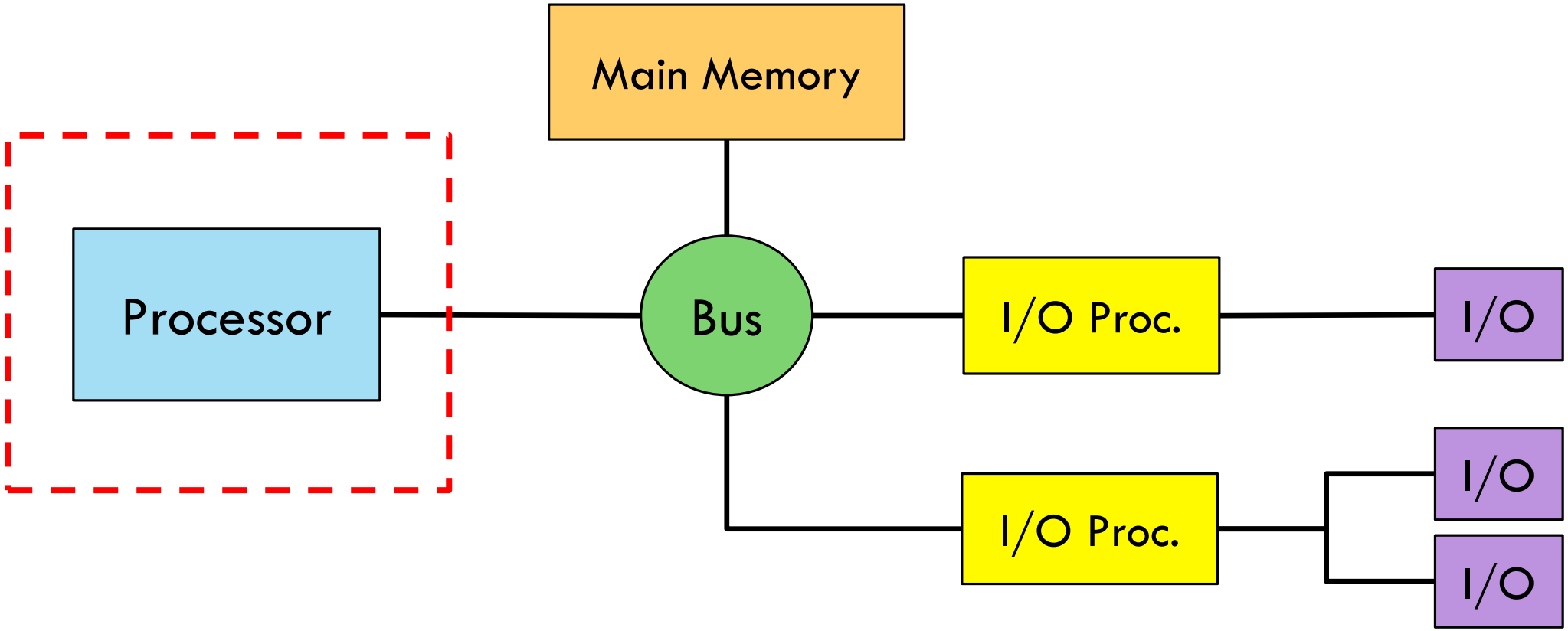
Advanced topics in Embedded Systems Design

Marten van Dijk, Syed Kamran Haider
Department of Electrical & Computer Engineering
University of Connecticut
Email: {vandijk, syed.haider}@engr.uconn.edu

UCONN

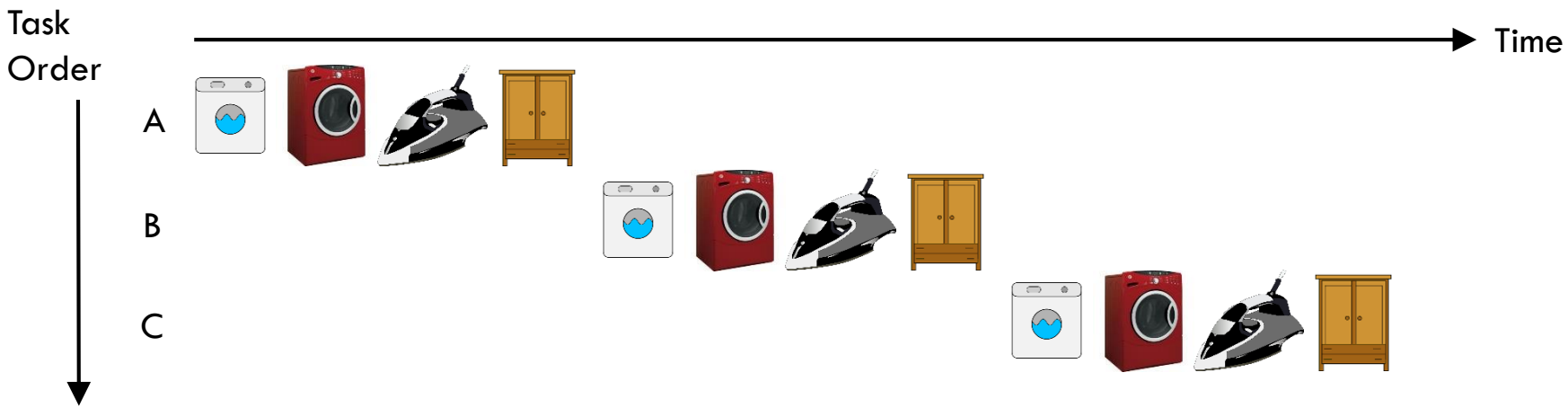
Some of these slides are extracted or copied from TTK4155:
"Industrial & Embedded System Design" offered at NTNU, Norway.

Basic Computer Components

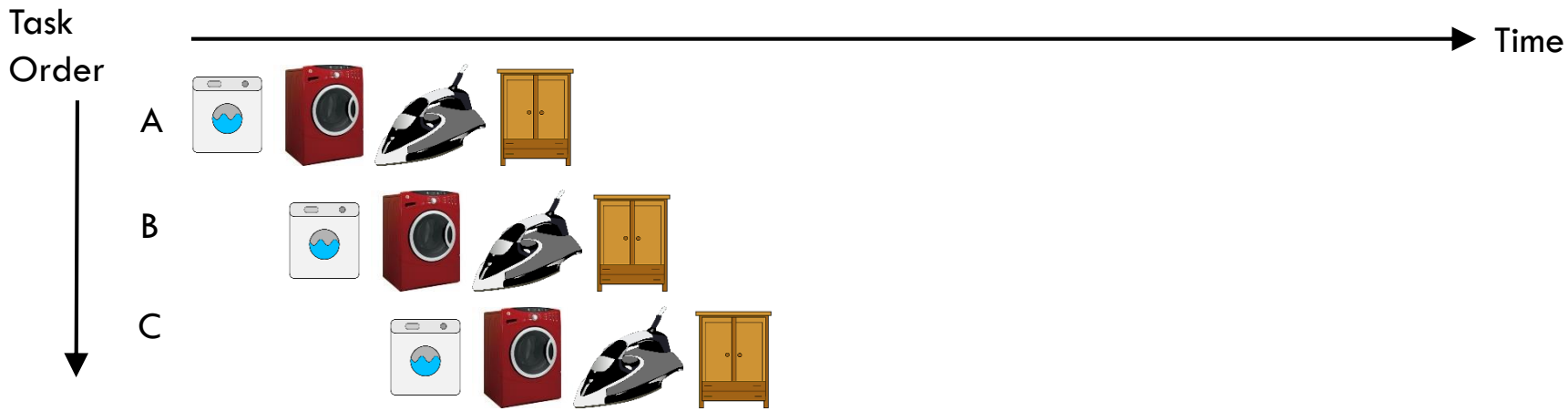


Pipelining in Real life

Non-Pipelined



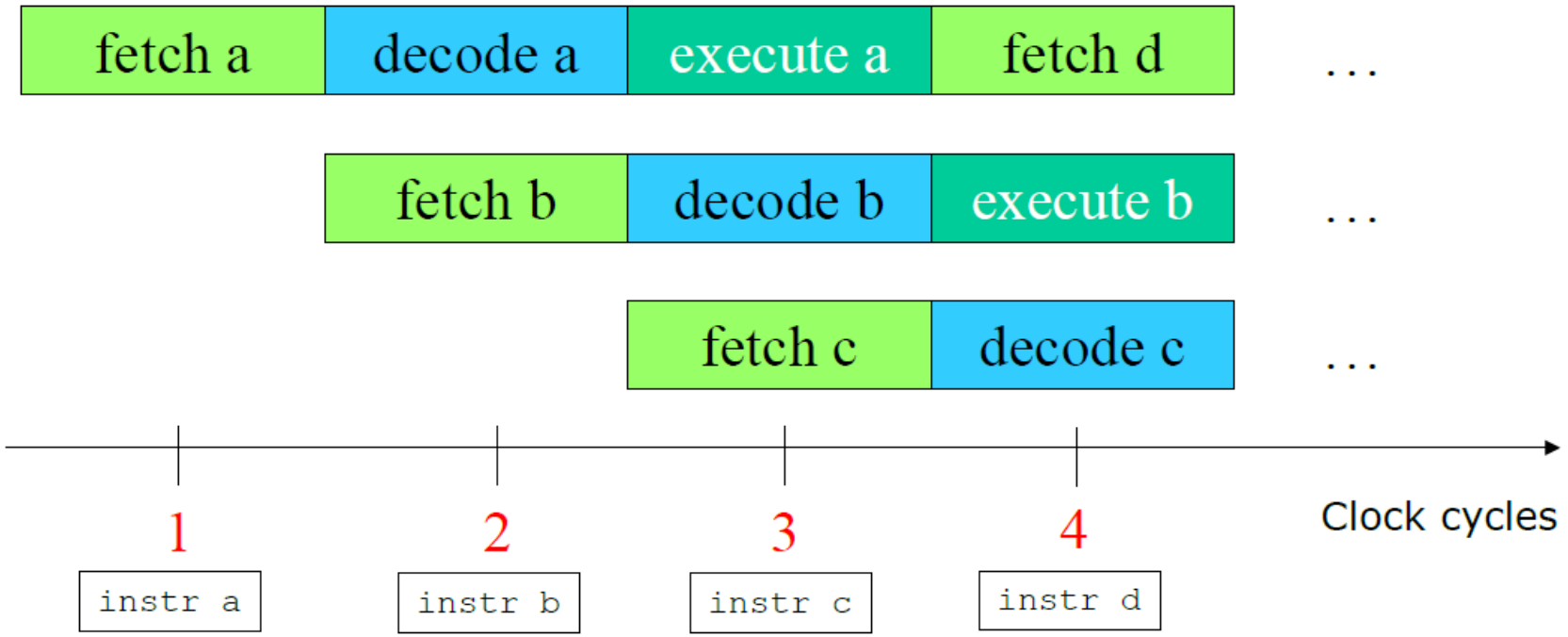
Pipelined



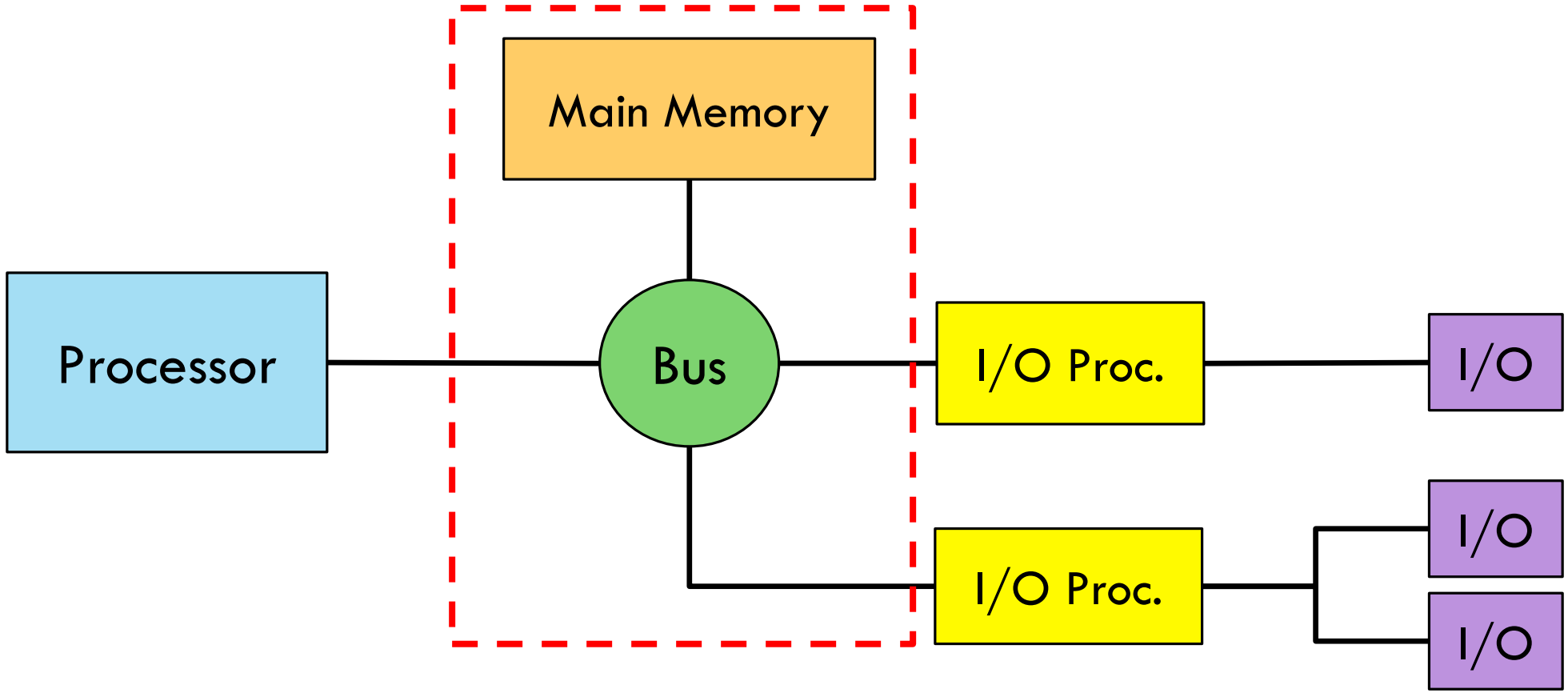
Instruction Pipelining

- Method for increasing the instruction throughput of the processor [IPS]
- Instruction execution may be partitioned into a fixed number of sequential steps, e.g. (ARM7):
  - Fetch (instruction from memory)
  - Decode (opcode and operands)
  - Execute…
- Stages may be executed in parallel, e.g. the CPU works with several instructions at the same time
  → Increased Throughput
- Some instructions and code sequences may reduce the performance gain (pipeline stalls) because of dependencies
  → Developing techniques to avoid stalls is very “hot” in current microprocessor research

3-Stage Pipeline



Basic Computer Components



Main Memory

- Stores instructions and data for the processor
- Connected to the processor via the memory bus

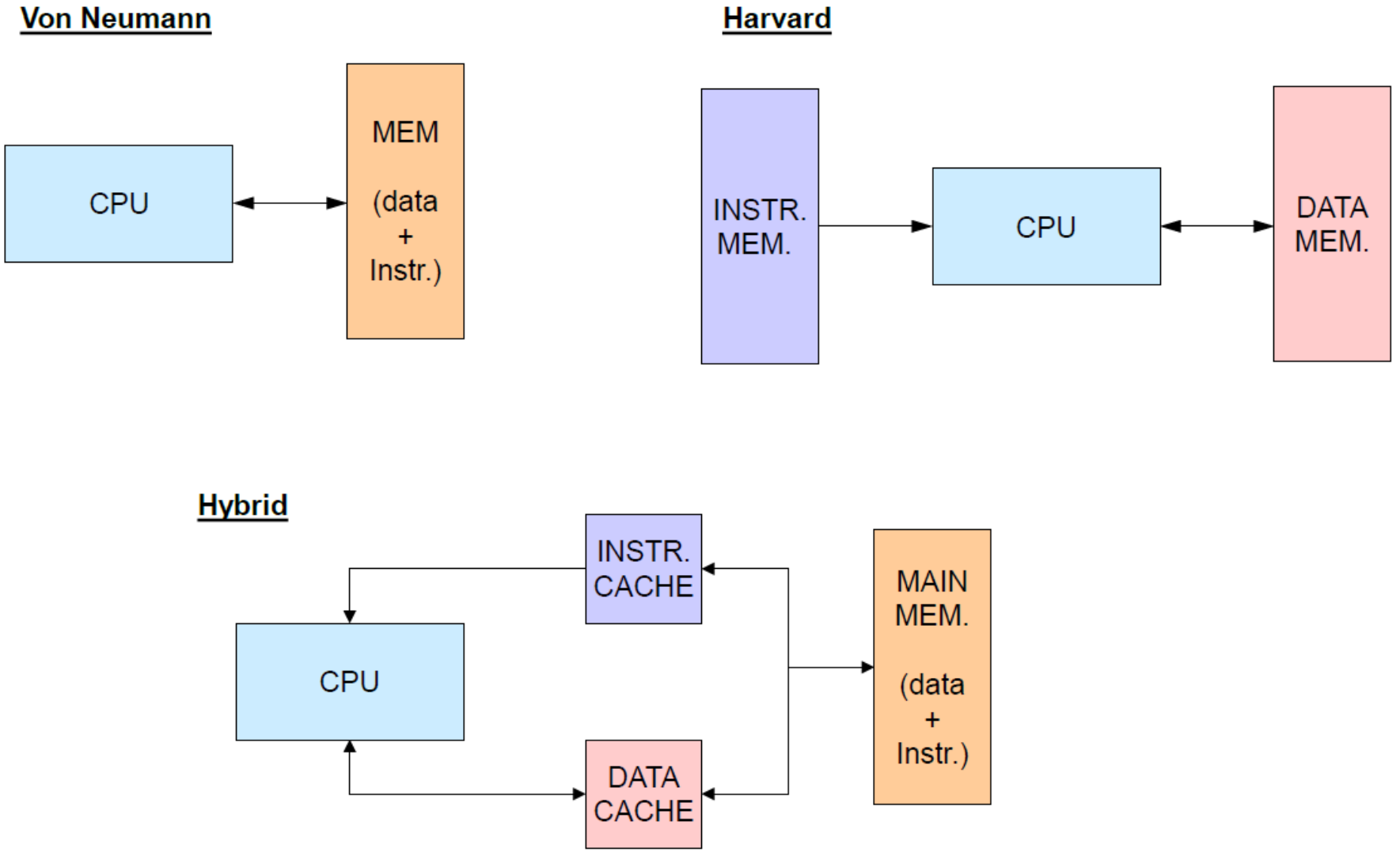
Two main types:

- Volatile, RAM (random bytewise read and write)
  - SRAM
  - DRAM
- Non-volatile, ROM (read (and sometimes write…))
  - OTPROM
  - EPROM
  - EEPROM
  - FLASH
  - FRAM

Memory Architectures

- Von Neumann machine/architecture
  - Memory viewed at a long continuous column of memory cells
  - No principal difference between data and instructions
  - No difference between different data types
  - Common physical storage for instructions and data
- Harvard architecture
  - Principal difference between data and instructions
  - Physically separate memories and buses for data and instructions
  - May have different word length for data and instructions
- Hybrid architecture
  - Harvard architecture between CPU and cache memory
  - In case of cache-miss: Von Neumann between CPU and main memory

Memory Architectures

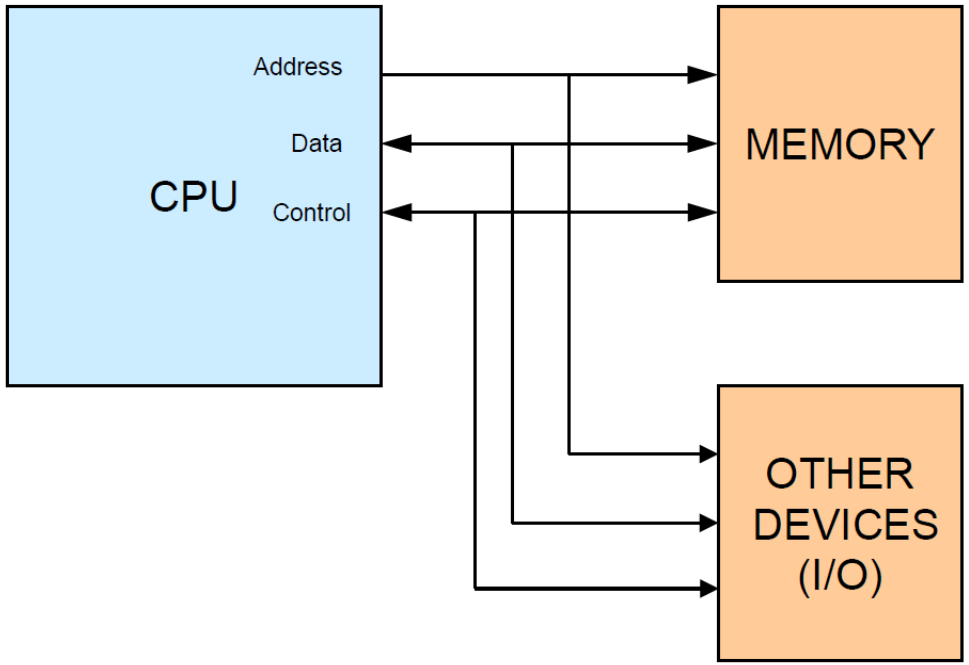


Bus and Communication Interfaces

- Parallel Bus Systems
  - Processor Buses – AVR etc.
  - Industrial Buses
    - VMEbus
    - CompactPCI
    - PC/104
    - …
- Serial Local Buses
  - SPI
  - MicroWire
  - I2C
  - 1-Wire
- Serial Lines (1 to 1, 1 to N)
  - UART
  - RS-232C
  - RS-422
  - USB
- Networks (N to M)
  - CAN
  - RS-485
  - LAN/Ethernet
- Wireless Communication
  - IR/IrDA
  - ISM
  - WiFi
  - Bluetooth
  - Zigbee

Parallel Bus Interfaces

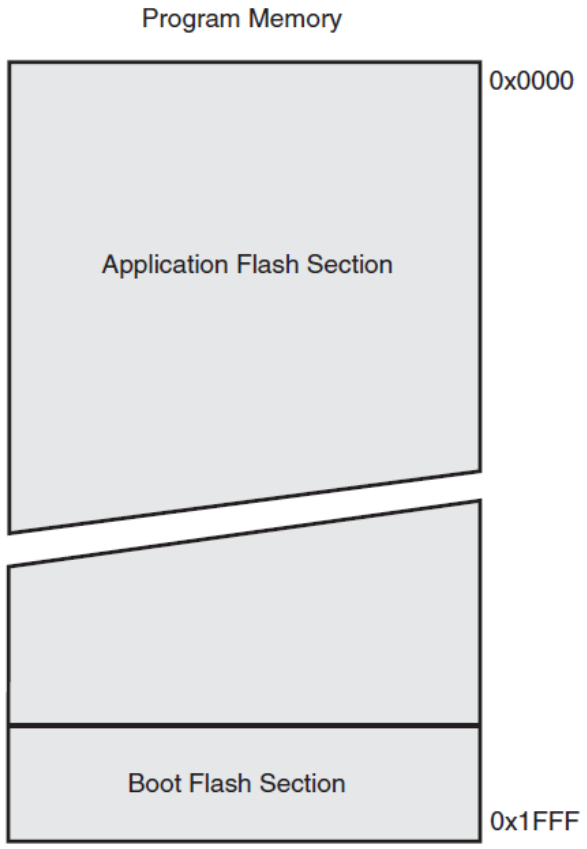
- A bus is defined as a group of signal lines that shares a common function and that connects the processor to the memory and I/O devices in the system
- The “three bus” system is the most common parallel bus architecture:
  - Address
  - Data
  - Control

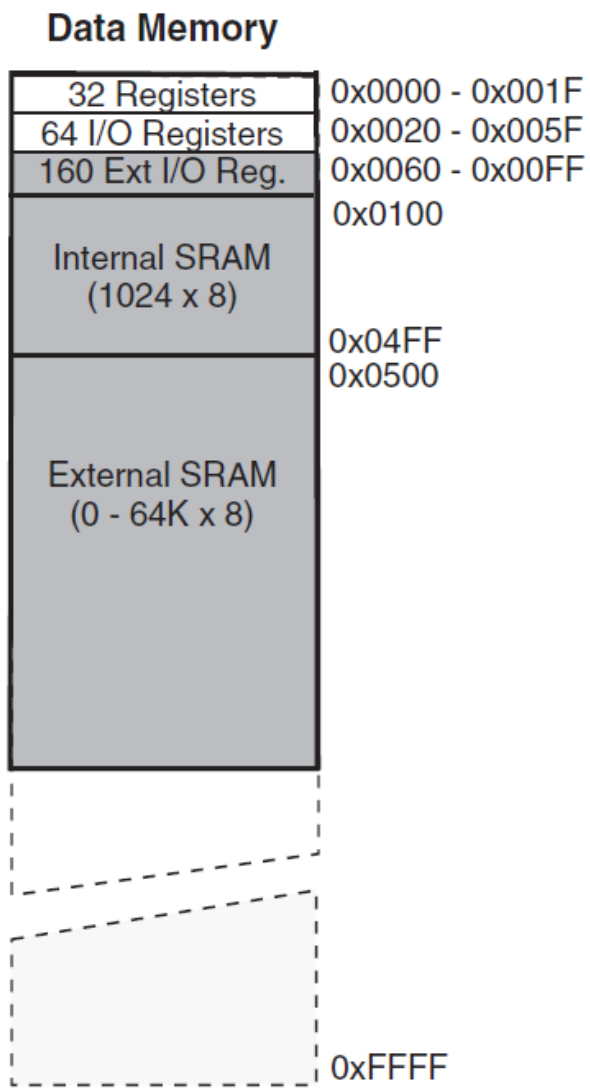


Address Space

- The range of memory locations addressable by a processor.
- Typically reflected by the width of the address bus
  - 16 bit → 2^{16} = 64 KB
  - 32 bit → 2^{32} = 4 GB
- Linear (flat) address space
  - One contiguous block of bytes (words)
  - Logical address = physical address
- Paged, Segmented
  - Organized in pages and/or segments:offsets (logical addresses)
  - Conversion between logical & physical addresses needed

Address space of AVR ATmega162

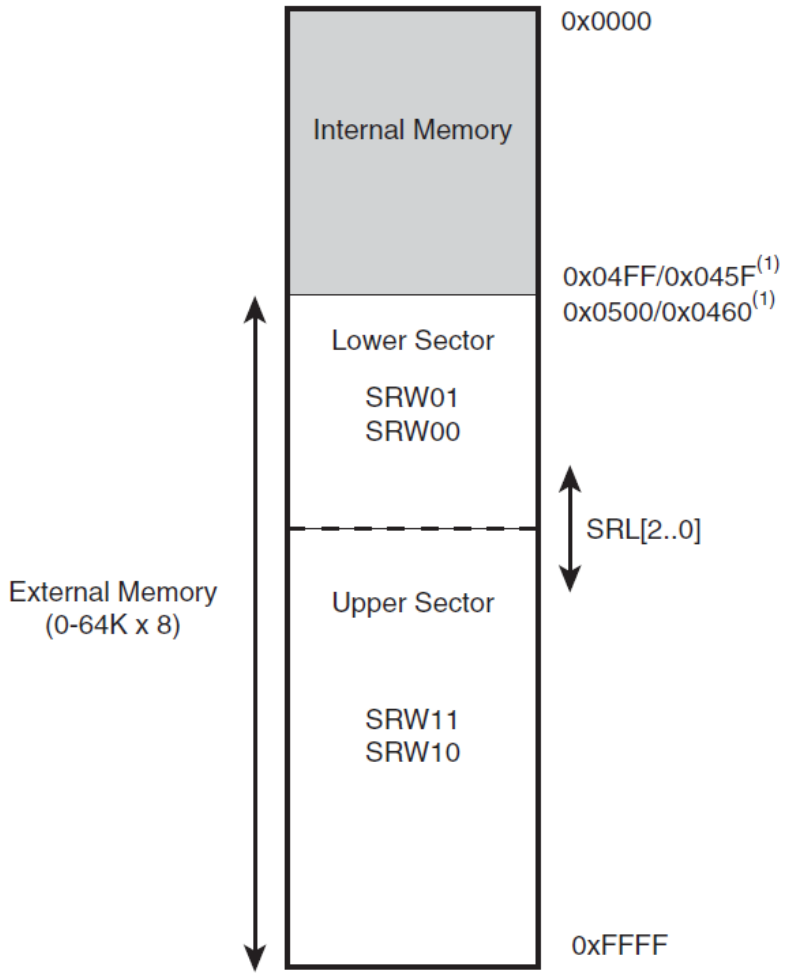




Interfacing External SRAM to ATmega162

The External Memory interface consists of:

- AD7:0 → Multiplexed low-order address bus and data bus
- A15:8 → High-order address bus (configurable number of bits)
- ALE → Address latch enable
- RD → Read strobe
- WR → Write strobe

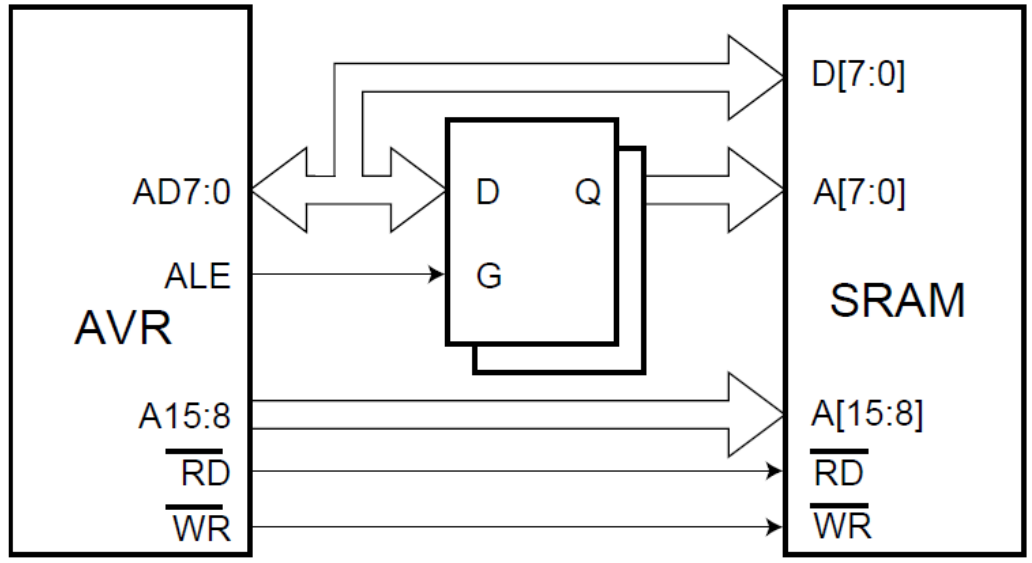


Interfacing External SRAM to ATmega162

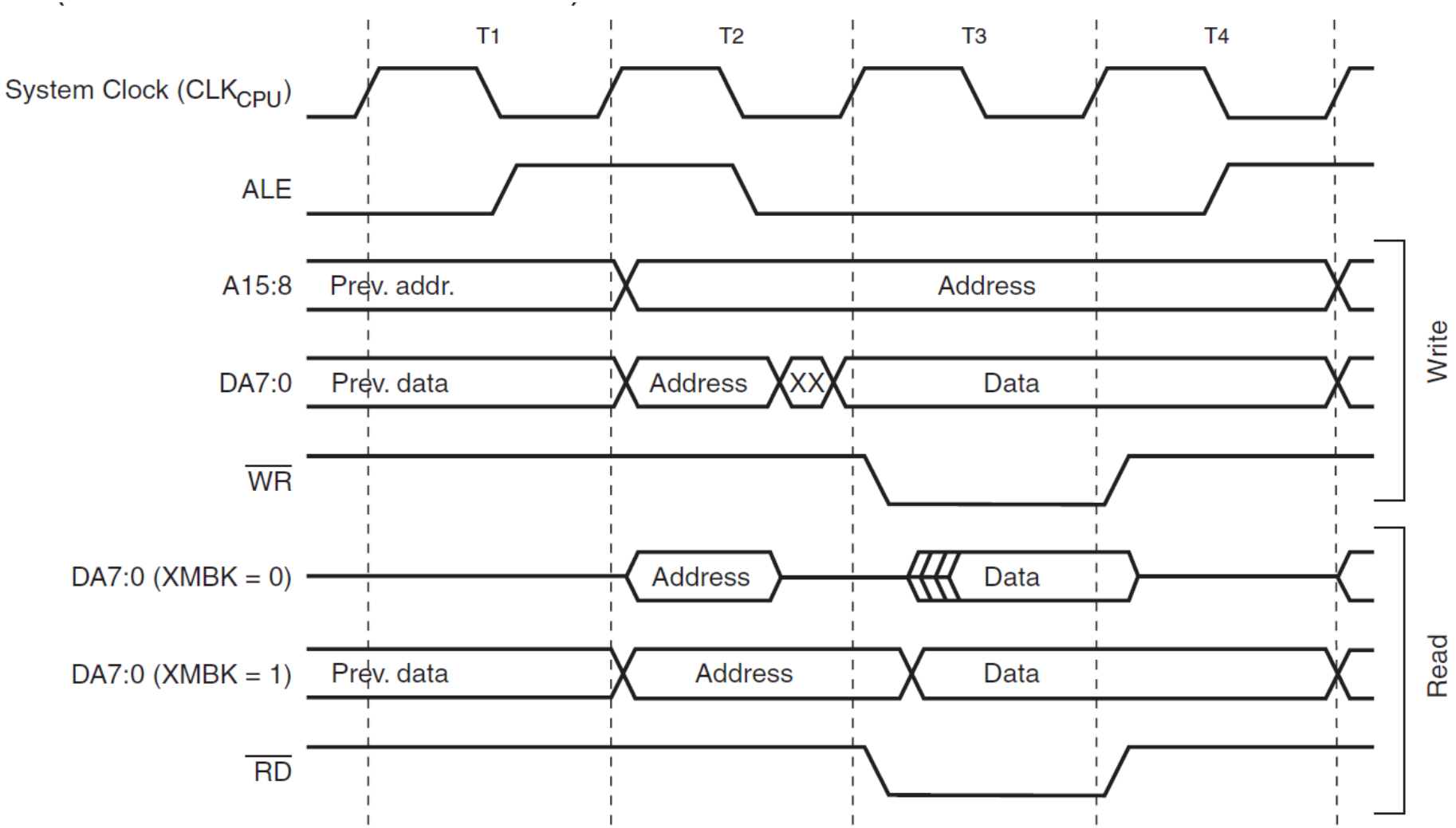
The External Memory interface consists of:

- AD7:0 → Multiplexed low-order address bus and data bus
- A15:8 → High-order address bus (configurable number of bits)
- ALE → Address latch enable
- RD → Read strobe
- WR → Write strobe

External SRAM Connected to the AVR



Processor/Bus cycle ATmega162 (without wait states)



Wait states

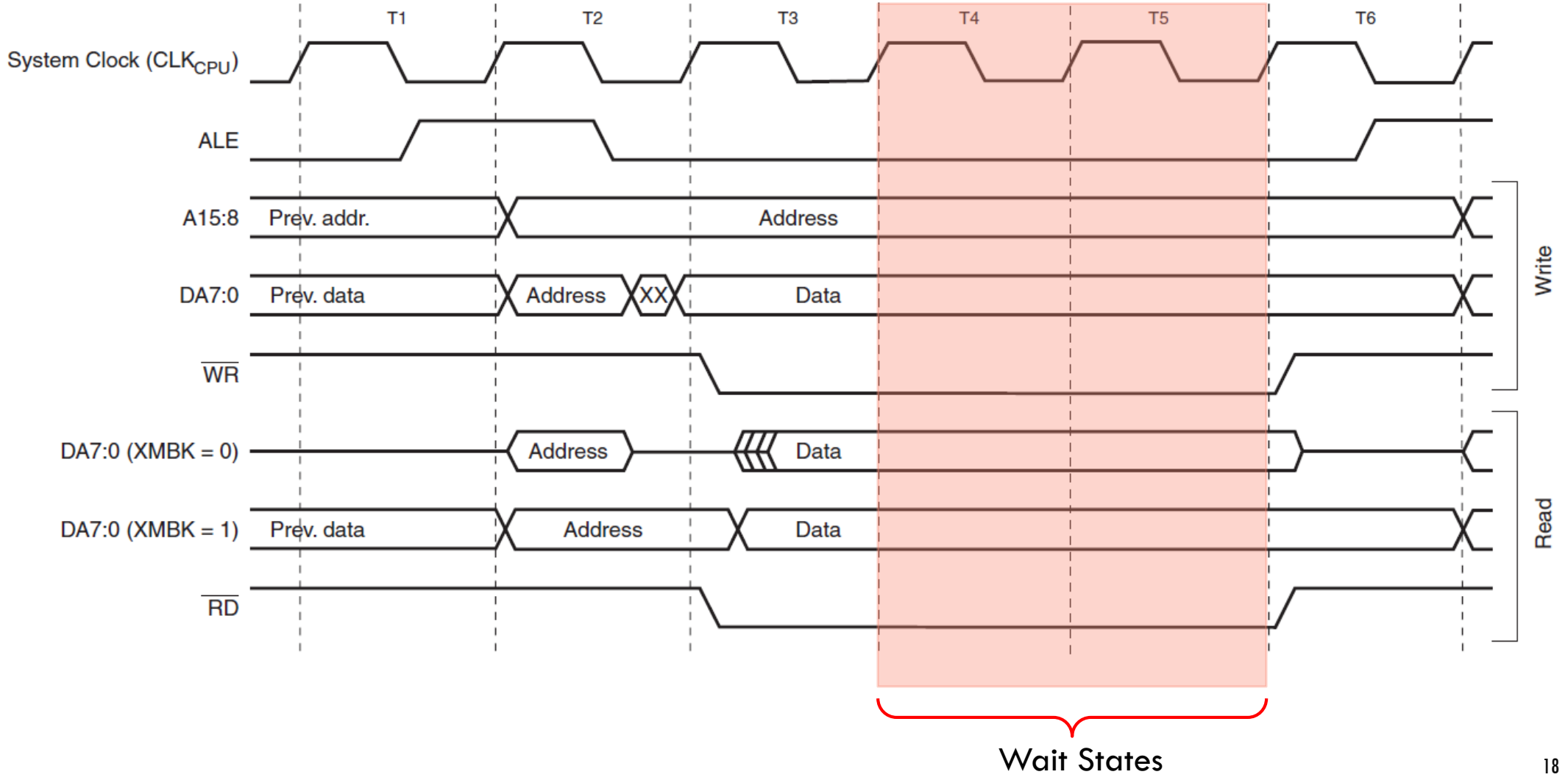
Problem: The processor is normally much faster than the peripheral devices connected to the bus interface

- Peripheral devices may not be able to respond to processor requests within the next clock state
- Data on the bus may be invalid when they are read by the processor

Solution: *Wait states*

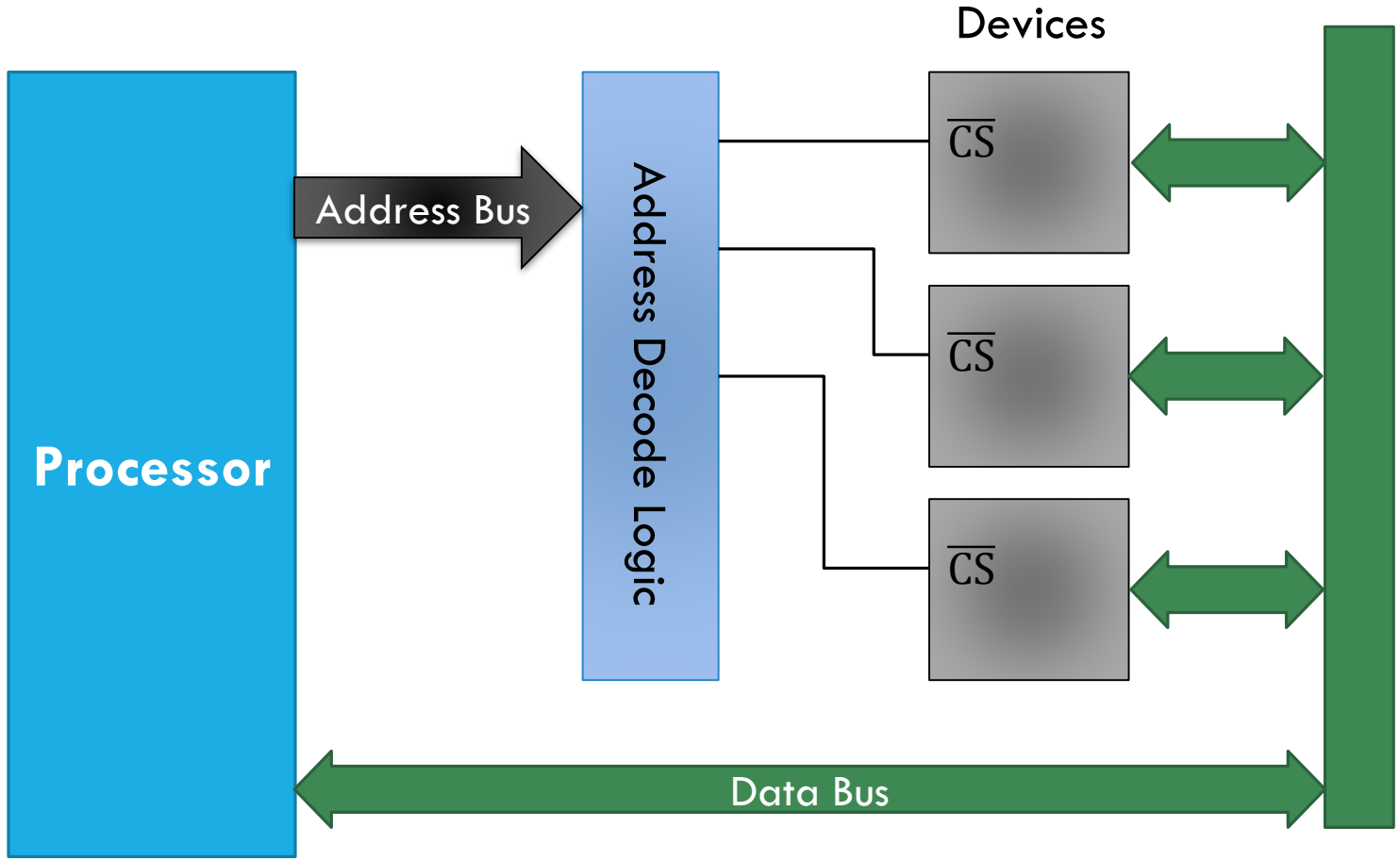
- Synchronous processors: Injects one or more extra clock cycles into the bus cycle.
- Asynchronous processors: Delayed assertion of the DTACK signal.

Processor/Bus cycles ATmega162 – 2 wait states



Address Decoding

- Address space is divided among several devices.
- Address Decoding Logic is configured according to the address space mapping.
- Address Decoding logic enables device(s) based on the address requested by the processor.



Example: Memory Map & Address Decoding

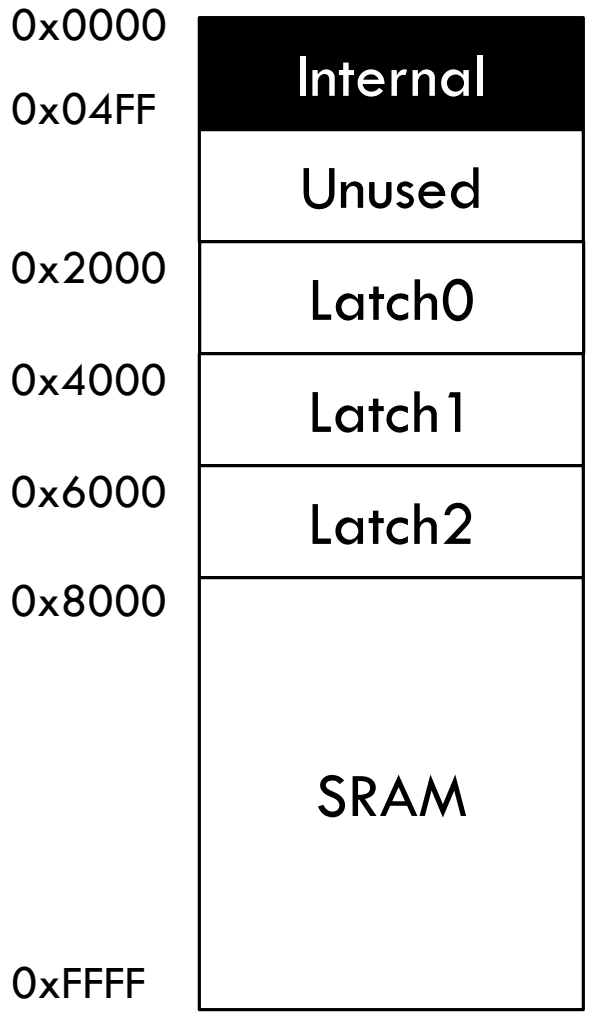
Design an AVR-based computer that includes four devices in its address space:

- One SRAM 32K, for data storage
- Digital outputs for driving 24 LEDs using three 8-bit latches

Make a memory map and address decoder for the system (use partial decoding)

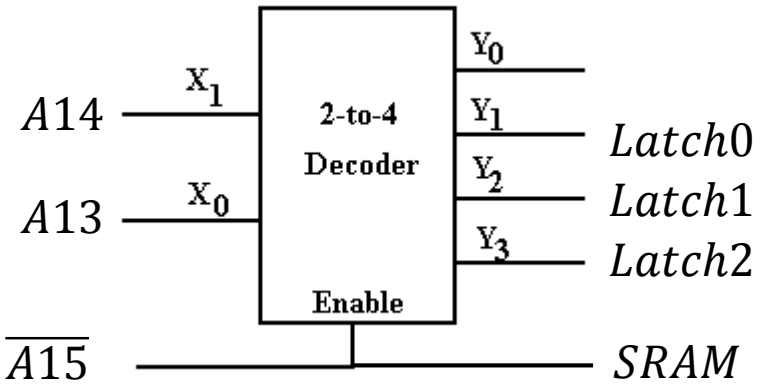
Selected Memory Map

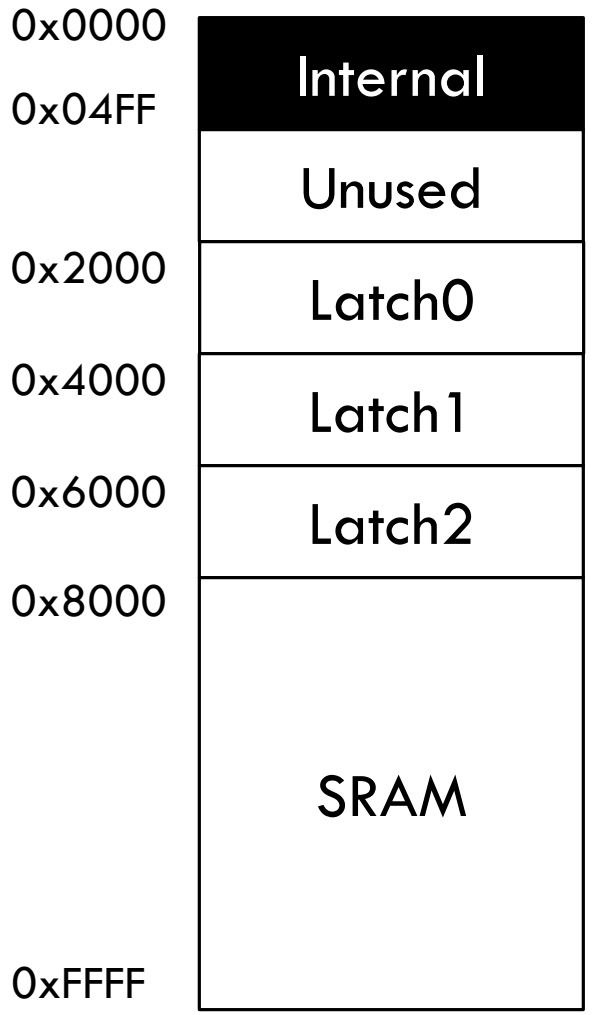
Device	Address Range	A15 … A0
Internal/Unused	0x0000 – 0x1FFF	0000 0000 0000 0000 0001 1111 1111 1111
Latch0	0x2000 – 0x3FFF	0010 0000 0000 0000 0011 1111 1111 1111
Latch1	0x4000 – 0x5FFF	0100 0000 0000 0000 0101 1111 1111 1111
Latch2	0x6000 – 0x7FFF	0110 0000 0000 0000 0111 1111 1111 1111
SRAM	0x8000 – 0xFFFF	1000 0000 0000 0000 1111 1111 1111 1111



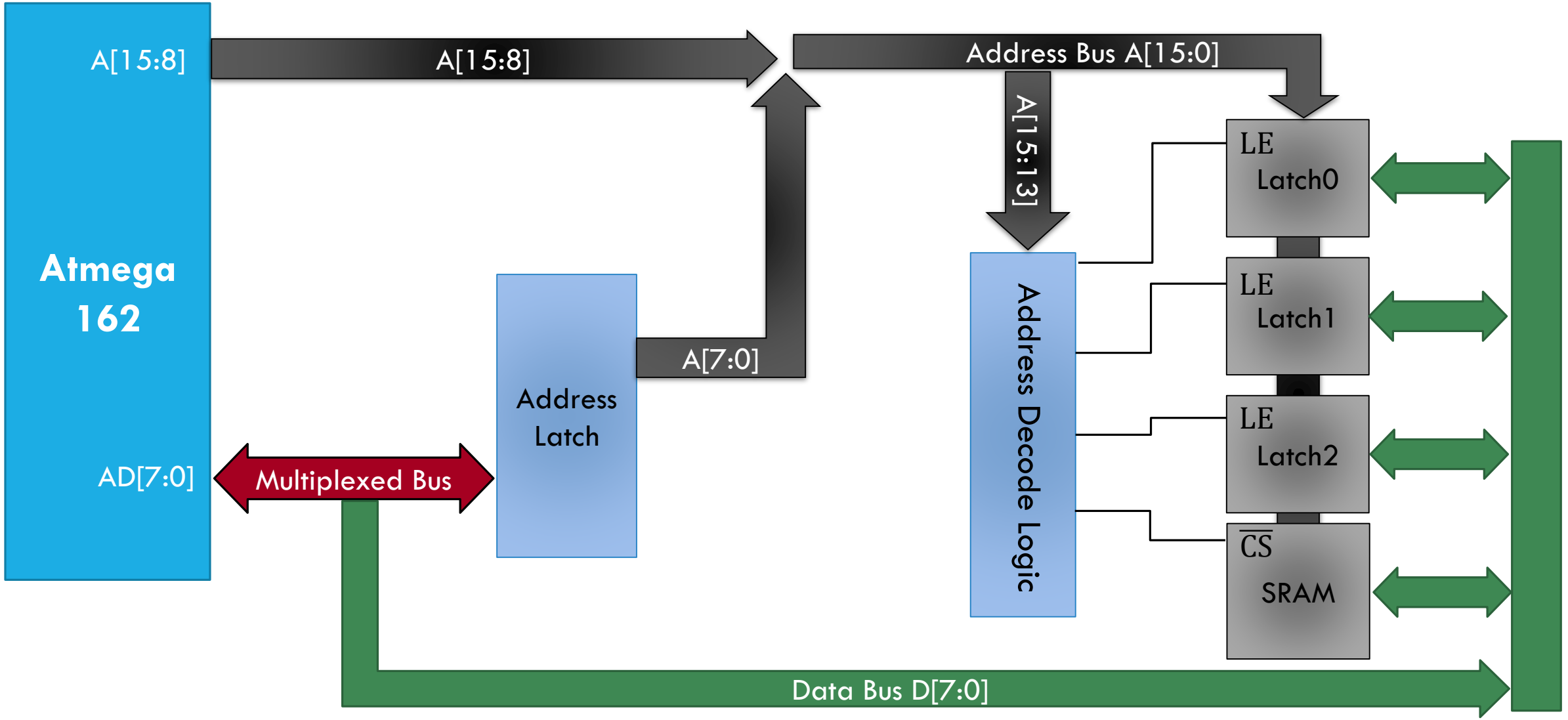
Address Decoding of selected Memory Map

Device	Address Range	A15 … A0
Internal/Unused	0x0000 – 0x1FFF	000x xxxx xxxx xxxx
Latch0	0x2000 – 0x3FFF	001x xxxx xxxx xxxx
Latch1	0x4000 – 0x5FFF	010x xxxx xxxx xxxx
Latch2	0x6000 – 0x7FFF	011x xxxx xxxx xxxx
SRAM	0x8000 – 0xFFFF	1xxx xxxx xxxx xxxx

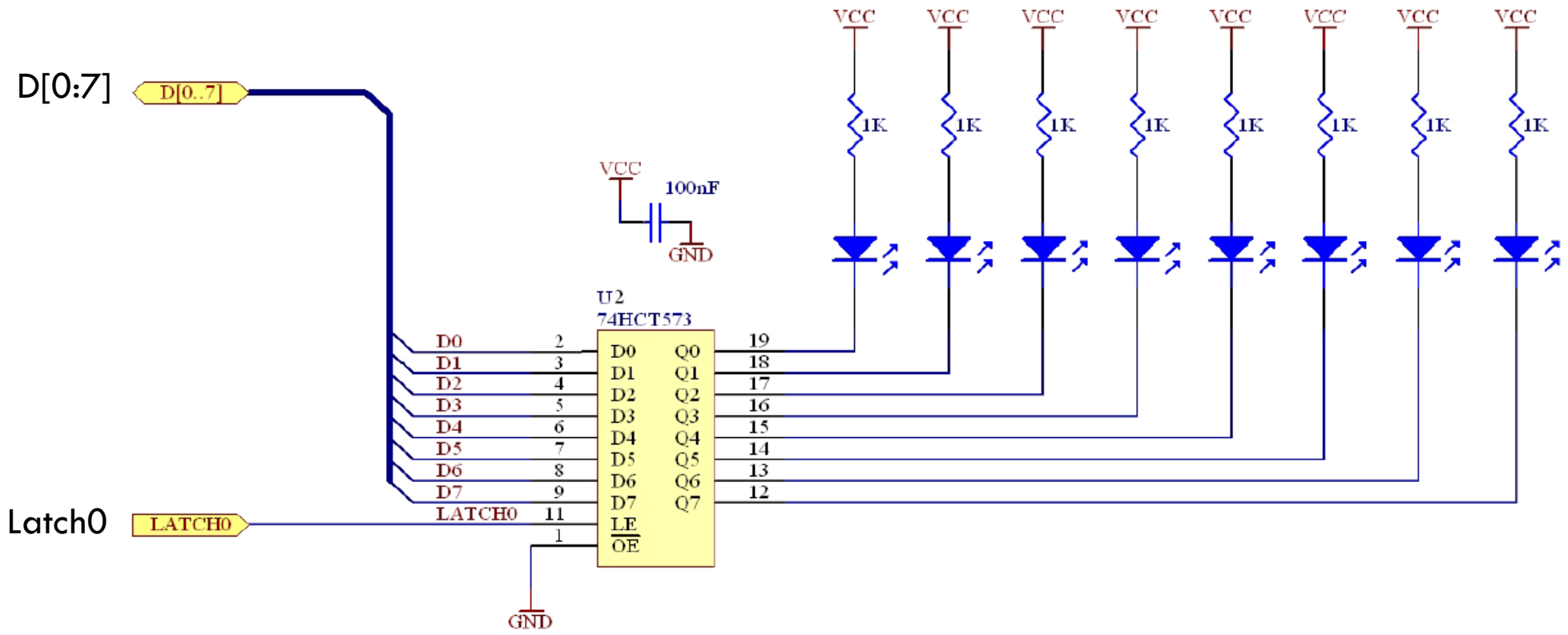




Bus Multiplexing & Address Decoding

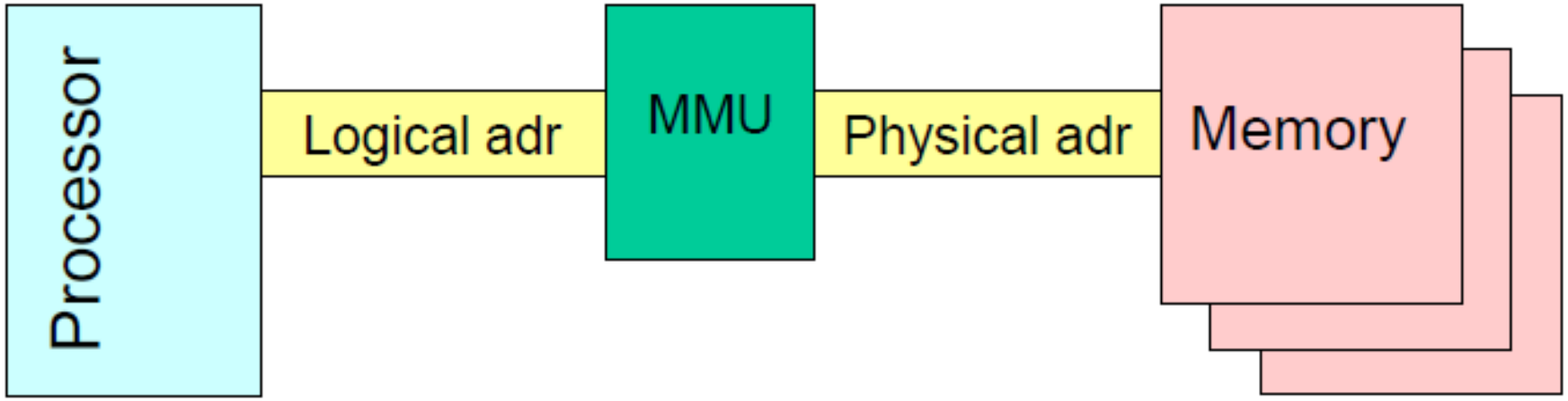


Connecting the LED Latches



Memory Management

- Translation between logical and physical memory space.
- Memory Management Unit (MMU) handles the address translation (and other jobs).



Memory Management

- Physical memory > logical memory
  - Banked memory → Dividing the physical memory into N partitions (banks) where the size of each partition is equal to (or lesser than) the processor's logical address space
- Physical memory < Logical memory
  - Virtual memory → Exploits the entire logical memory space of the processor. On-demand loading of data blocks to the physical memory from secondary storage (paging).
- Protection of memory regions
  - Monitor the address bus and intercept in case of unauthorized access to critical memory regions (OS, I/O space, interrupt tables etc.) (MPU)
- Isolation of tasks/threads
  - Prevent unauthorized access between the memory spaces of threads in a multitasking system

Example: Banked memory

In some cases it will be necessary to expand the physical memory beyond the logical memory space of the processor, e.g. 512Kb memory on a 16 bit address buss.

- **Solution:** Map a data latch into the processor's address room and use it to keep the 4 MSB of the 19 bit physical address
  →16 memory banks of 32 Kb = 512 Kb available to the application
  → Needs software control