



Universiteit Leiden

# Embedded Systems: Hardware Components (part II)

**Todor Stefanov**

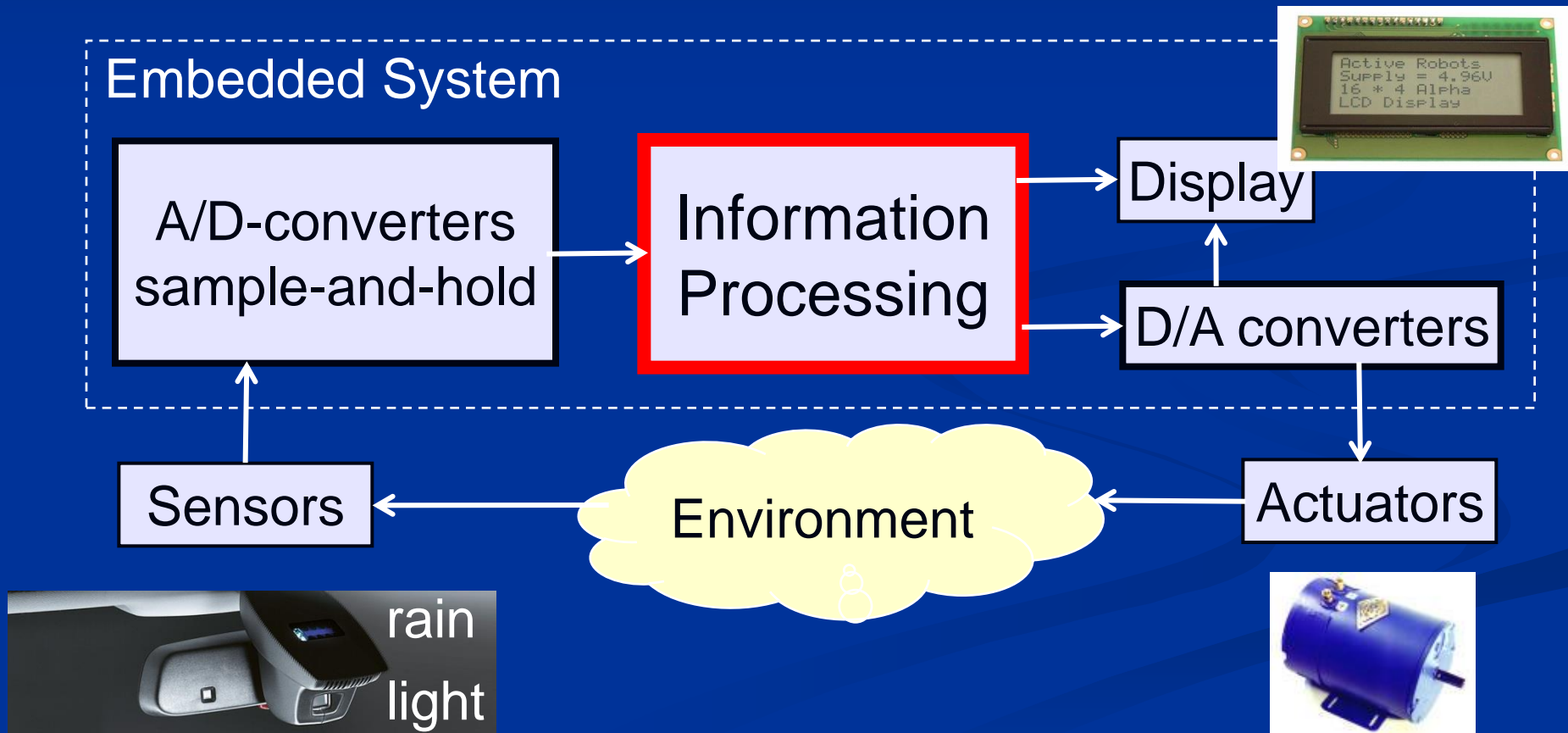
Leiden Embedded Research Center,  
Leiden Institute of Advanced Computer Science  
Leiden University, The Netherlands

# Outline

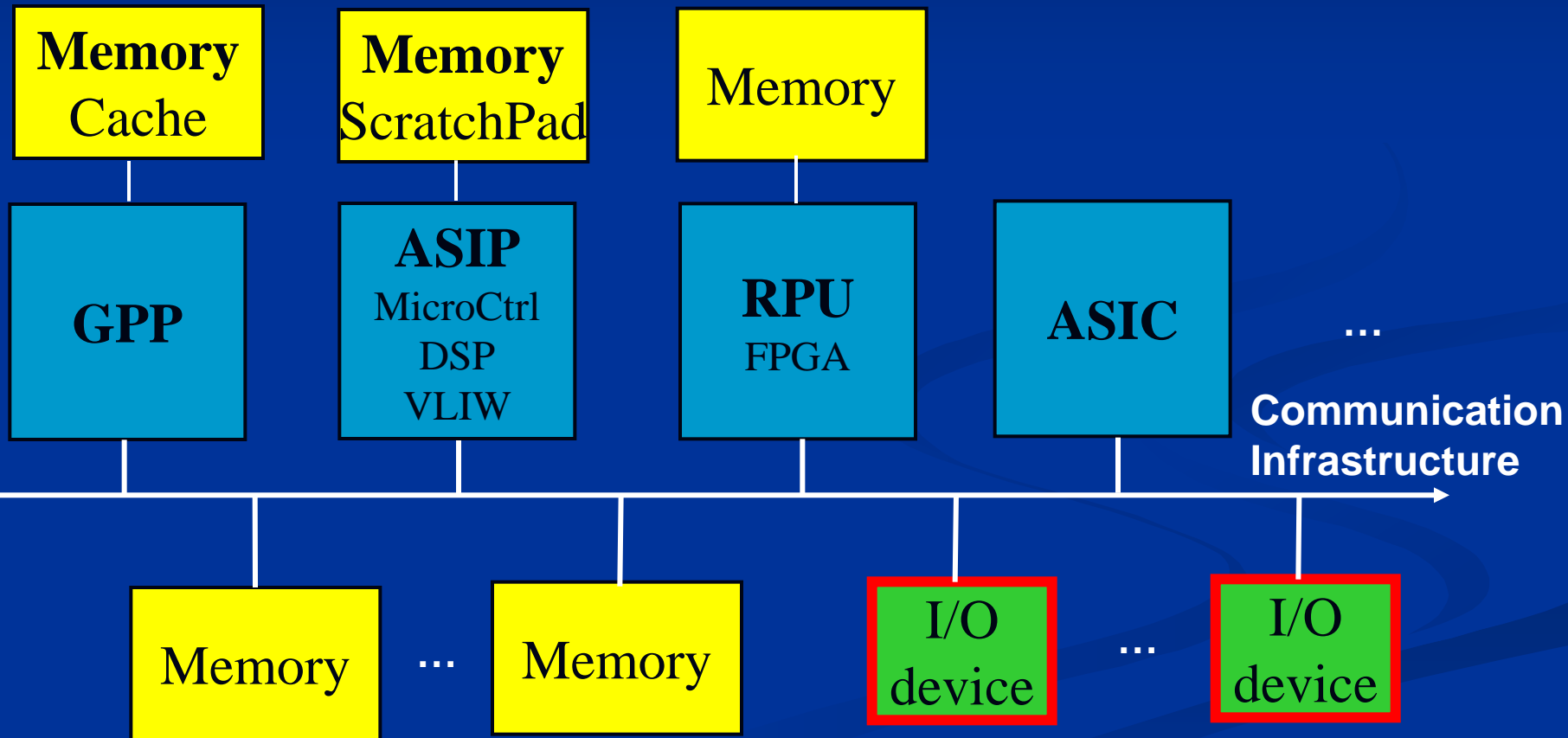
- Generic Embedded System component structure
- Sensors
- Analog-to-Digital (A/D-) converters
- Computation Components
  - General Purpose Processors (GPPS)
  - Application Specific Instruction Set Processors (ASIPs)
  - Reconfigurable Processing Units (RPUs)
  - Application Specific Integrated Circuits (ASICs)
- Memory
- Input/Output Devices
- Communication Infrastructure
- Digital-to-Analog (D/A) converters
- Actuators

# Embedded Systems Hardware

Embedded Systems hardware is frequently used in a loop (*“hardware in a loop”*):

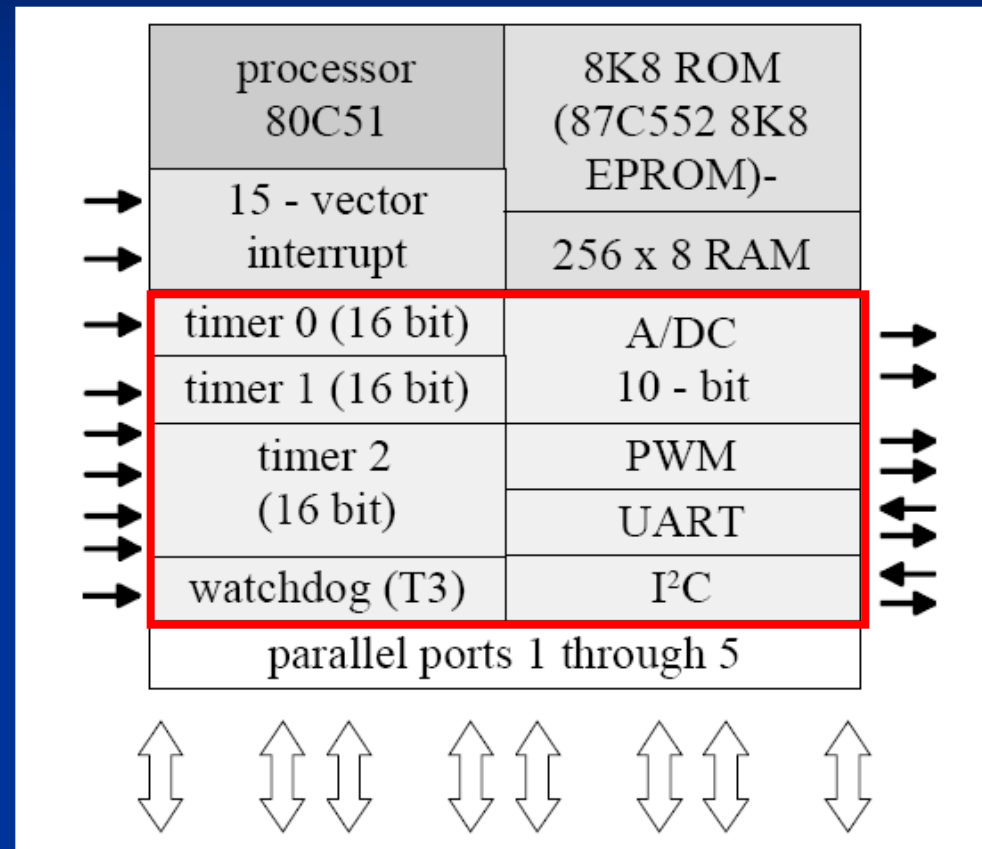


# Information Processing System: Input/Output Devices



# Examples of I/O Devices

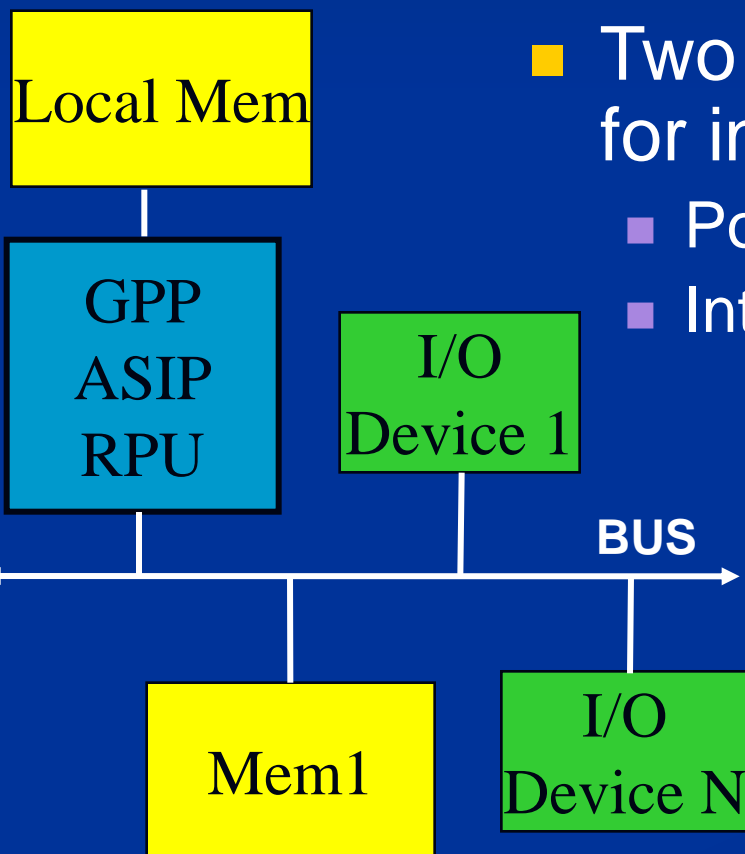
- Timers (counters)
- WatchDogs
- Pulse Width Modulators (PWM)
- Universal Asynchronous Receiver Transmitter (UART) – 2 serial lines (Tx and Rx)
- Inter Integrated Circuit (I<sup>2</sup>C) – 2 serial lines (data and clock)
- ...



**Philips 83C552:**  
**8 bit-8051 based microcontroller**

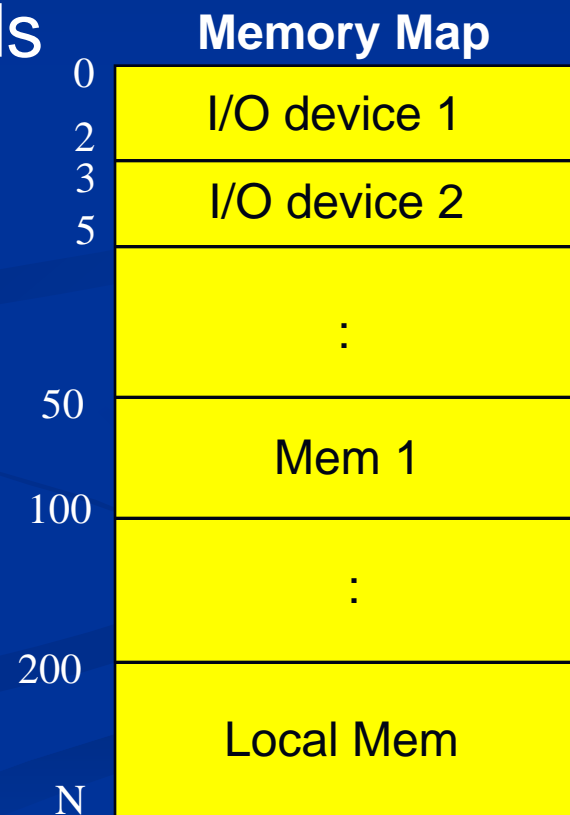
# How Computation Components (CCs) "talk" to I/O Devices?

- I/O Devices are typically mapped in the address space of CCs
- CCs communicate with I/Os by reading/writing from/to memory cells

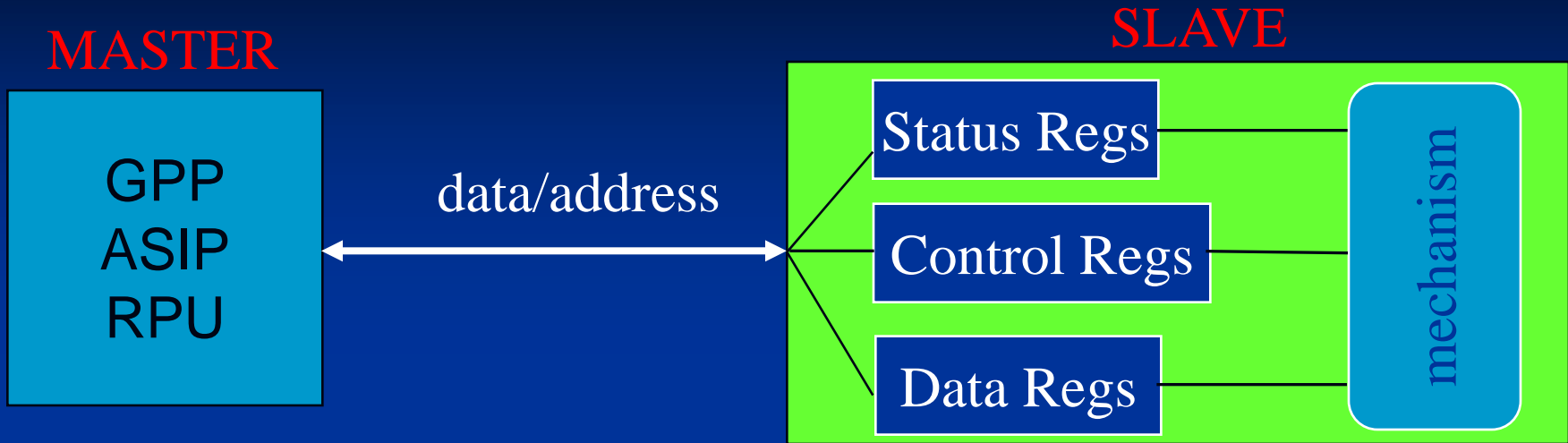


■ Two general methods for interaction

- Polling
- Interrupts



# Polling: Busy-Wait Interface



- *Continuous polling*

...

```
while (device not ready) { }
```

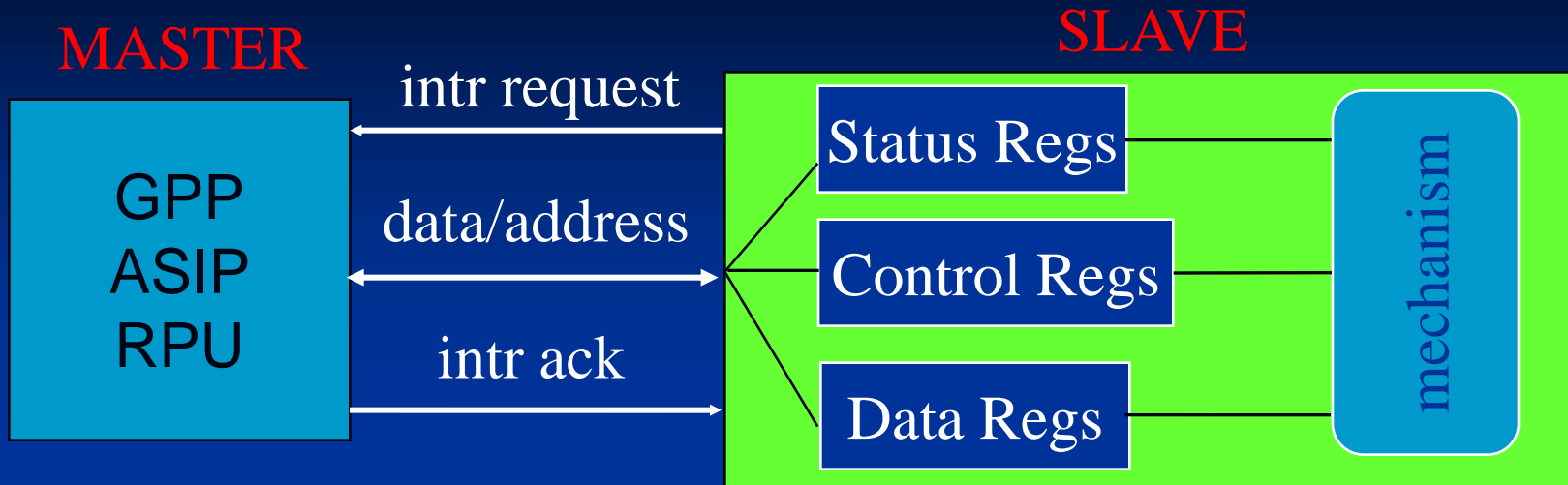
...

- *Periodic polling*

## Issues:

- MASTER is tied up in communication with I/O device – polling the status
- Polling and/or waiting for I/O device takes time from the MASTER
- Typically one MASTER in system, but many I/O devices
- Only really useful if devices are fast
  - No time consuming context switches

# Interrupt Interface



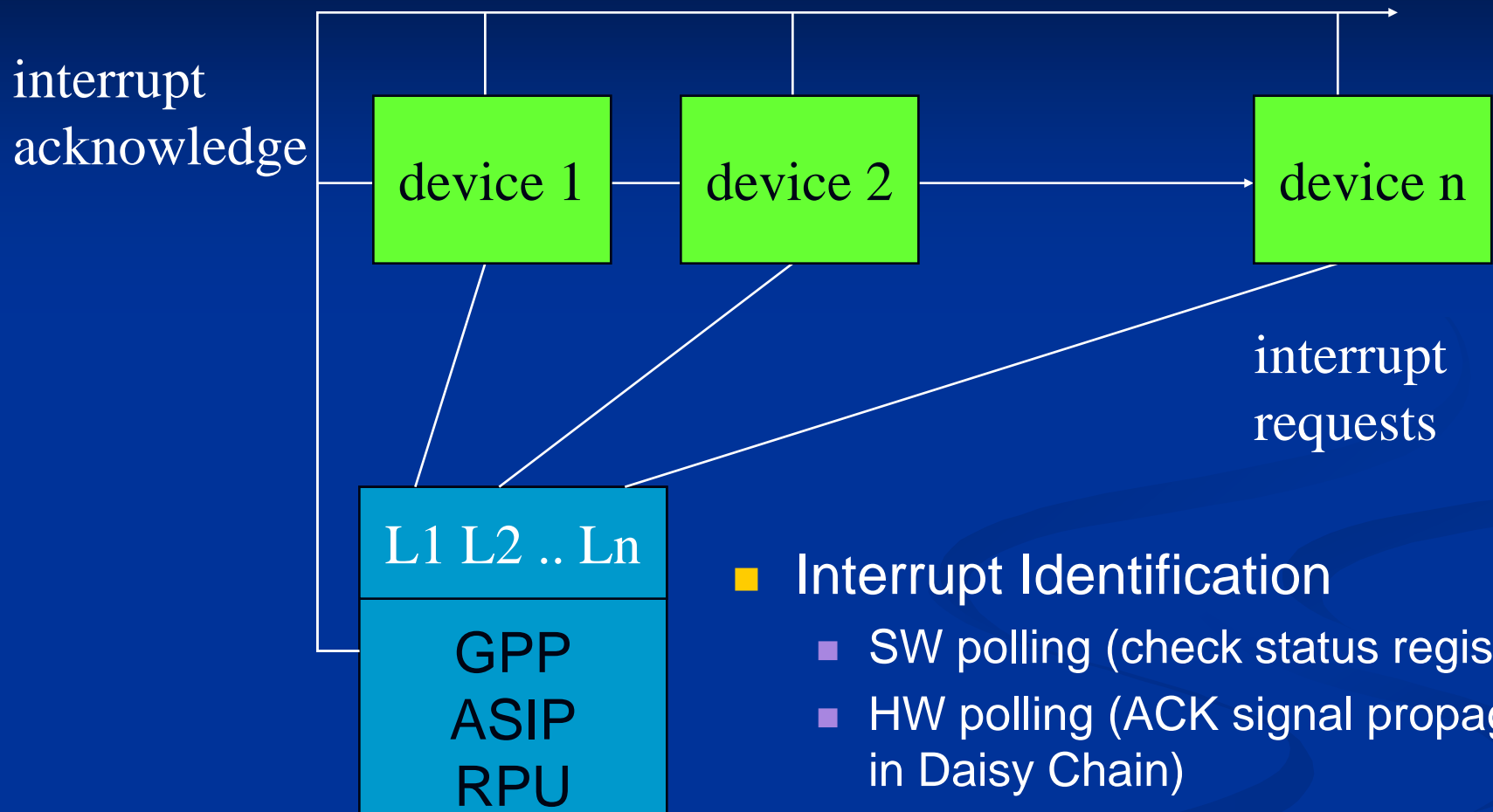
- When I/O device needs MASTER
  - interrupt signal is sent
- MASTER is “forced” to suspend its current task
  - Interrupts can be ignored (masked) when critical task is executed
- MASTER acknowledges interrupt and jumps to interrupt service routine
- When MASTER finished, control is returned to the interrupted task

## Key observations:

- HW polling of the interrupt signal
  - at beginning of each instruction
- MASTER is free to do something else until attention is needed
- Improves utilization of MASTER
- I/Os can proceed asynchronously
- Time consuming context switches!

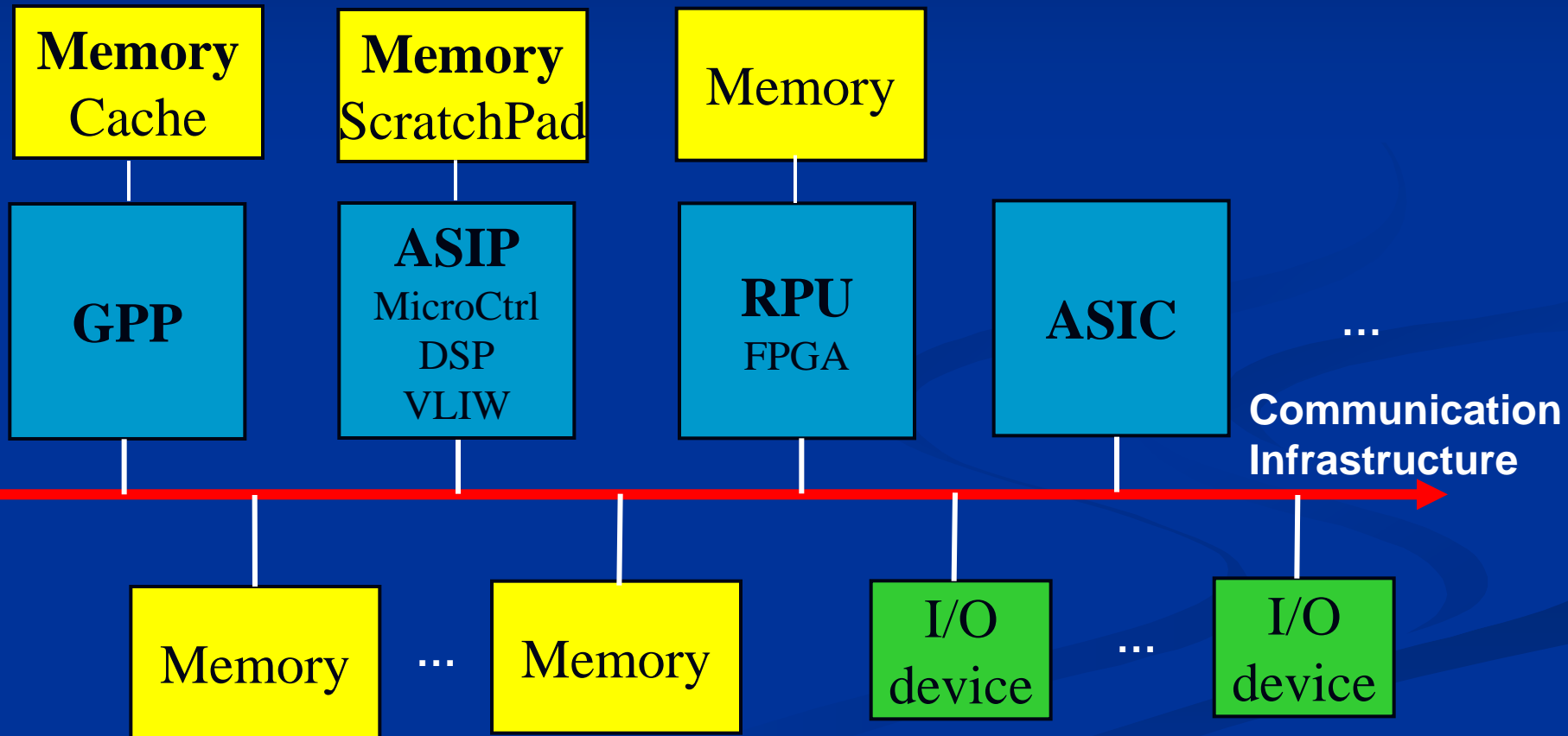


# Communication with many I/Os



- Interrupt Identification
  - SW polling (check status registers)
  - HW polling (ACK signal propagated in Daisy Chain)
- Interrupt Priority (in case multiple interrupts to be served)

# Information Processing System: Communication Infrastructure



# Types of Communication Infrastructure

- Single Bus
  - Single shared resource
  - May become the bottleneck in the system
- Multiple Busses/Crossbar Switches
  - Wire Switching communication infrastructure
- Networks (on Chip)
  - Packet Switching communication infrastructure

# Requirements for Comm. Infrastructure in ES

- Guaranteed Performance (for real-time ES)
  - bandwidth
  - latency
- Efficiency
  - cost (material, installation, maintenance)
  - low power
- Robustness
  - fault tolerance
  - maintainability, diagnoseability
  - security, safety

# Generic Bus Structure

- What is a BUS?

- Set of wires

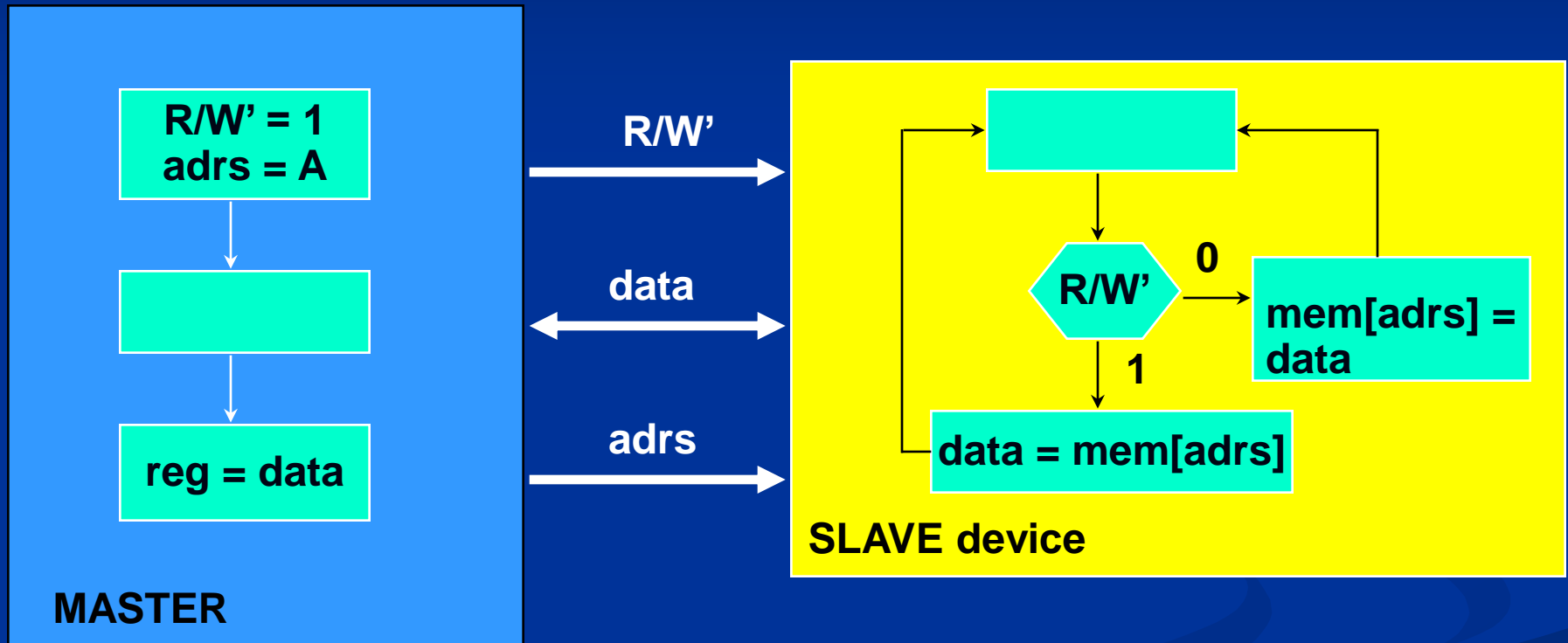
Address wires:  A horizontal double-headed arrow with a diagonal slash through it. The letter 'm' is centered above the arrow.

Data wires:  A horizontal double-headed arrow with a diagonal slash through it. The letter 'n' is centered above the arrow.

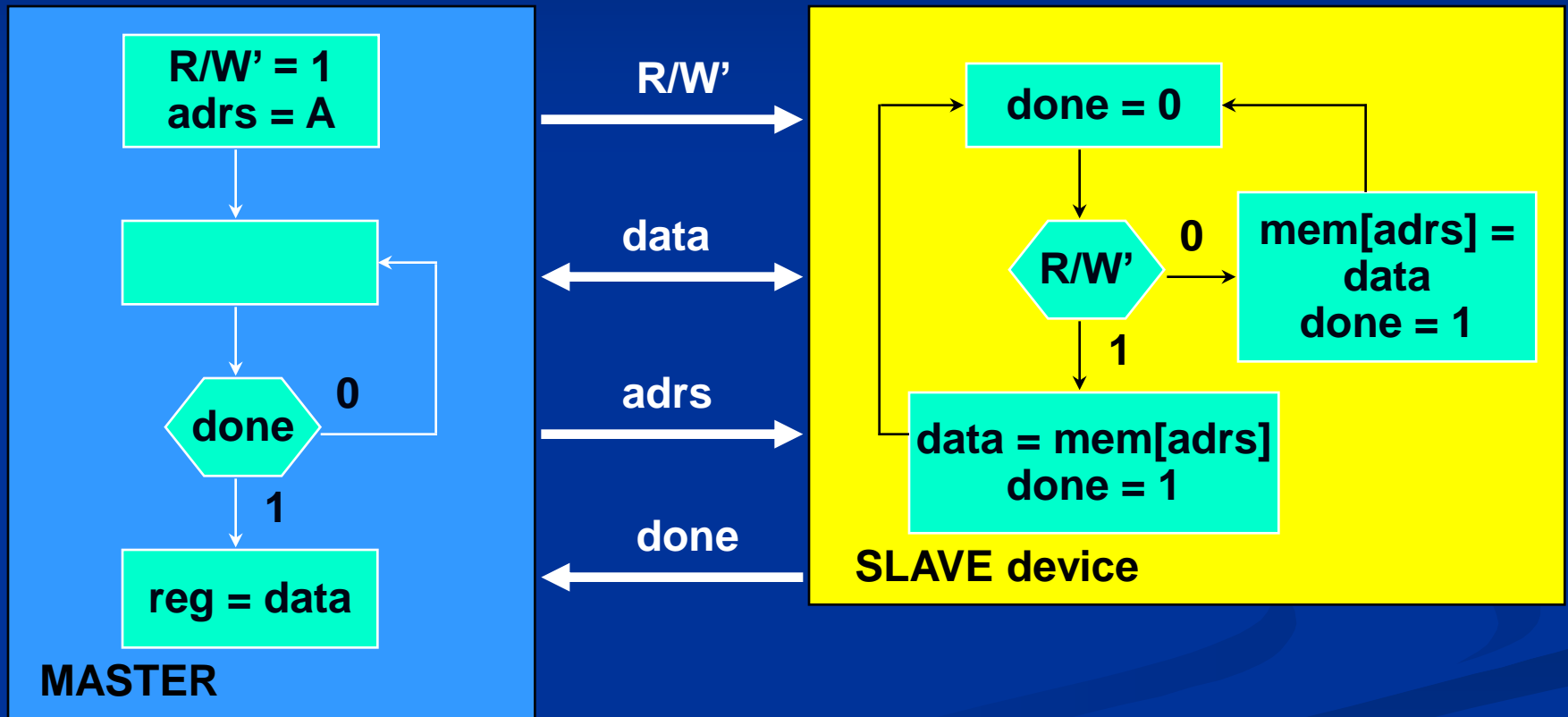
Control wires:  A horizontal double-headed arrow with a diagonal slash through it. The letter 'c' is centered above the arrow.

- Common protocol for communication with devices

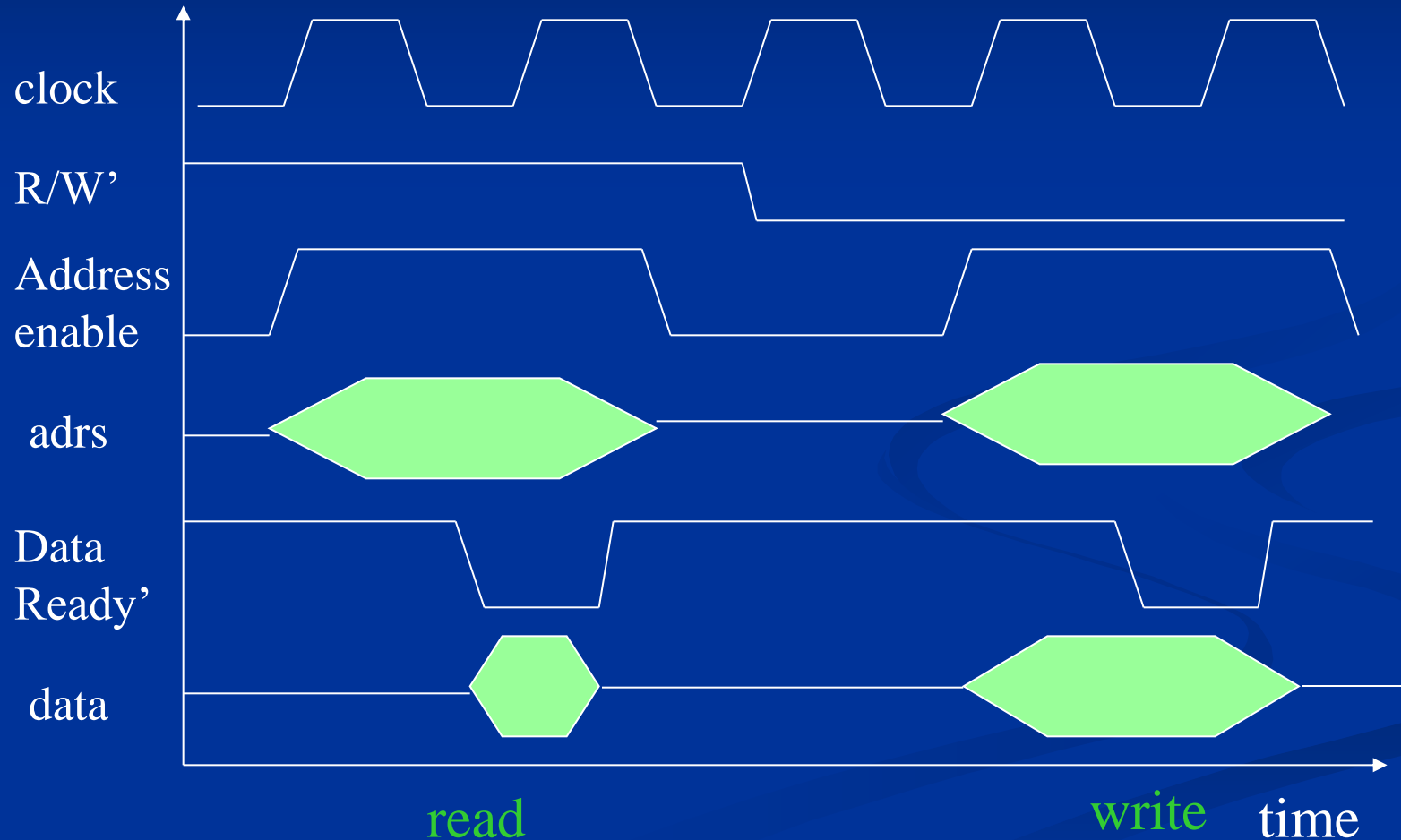
# Generic Fixed-delay Access Protocol



# Generic Variable-delay Access Protocol



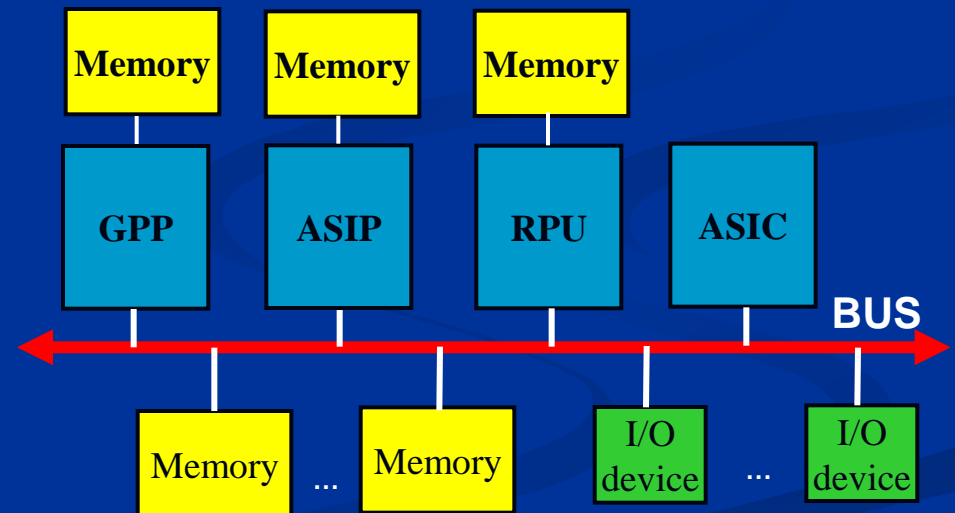
# Typical Bus Access Protocol



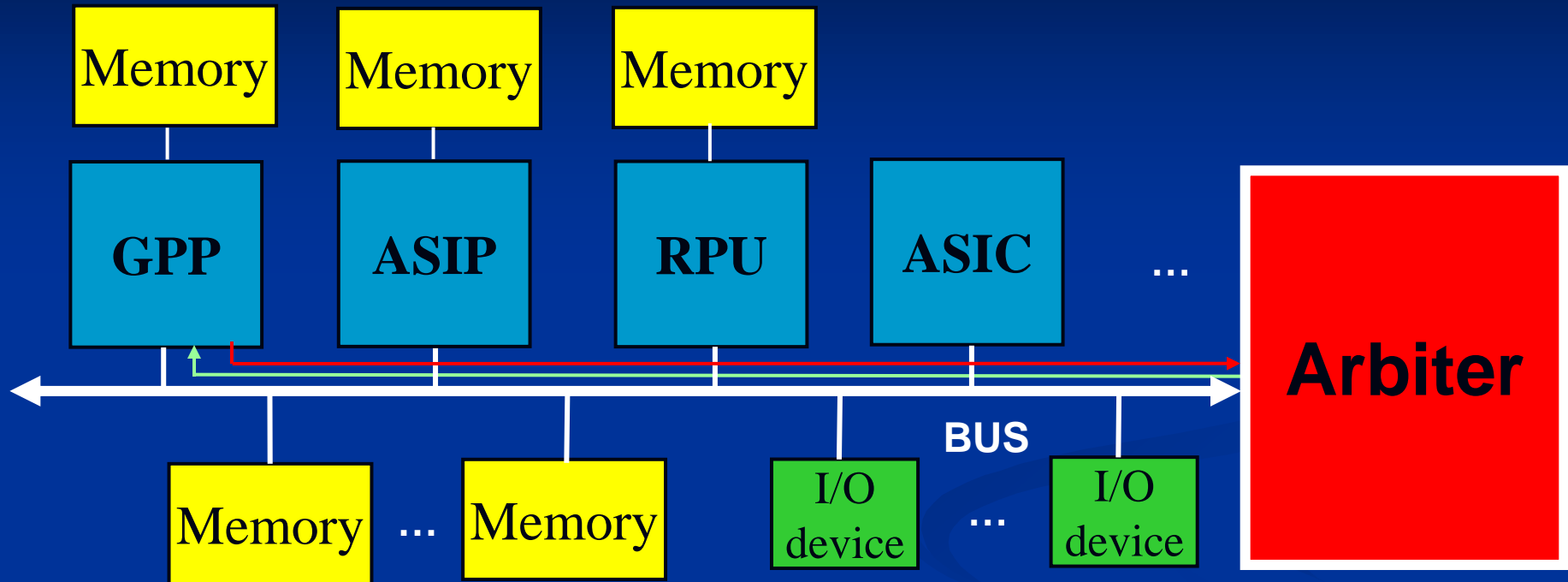


# What if Multiple Masters “want” the BUS simultaneously

- **Arbiter** is used
  - Controls access to the shared bus
  - Uses arbitration policy to select master to grant access to bus
- **Arbitration** policy
  - Centralized Arbitration
    - Round Robin policy
    - Priority policy
    - TDMA policy
  - Distributed Arbitration
    - Carrier Sense Multiple Access / Collision Detection (CSMA/CD)



# Centralized Arbitration



- Two important control signals per MASTER, i.e.,
  - *bus\_request* – from MASTER to ARBITER
  - *bus\_granted* – from ARBITER to MASTER
- Minimal change is required if new components are added to the system

# Arbitration Policies (1)

- Random
  - Randomly select master to grant bus access to
- Static priority
  - Masters are assigned with static priorities
  - Higher priority master request always serviced first
  - Can be pre-emptive or non-preemptive
  - May lead to starvation of low priority masters
- Round Robin (RR)
  - Masters allowed to access bus in a round-robin manner
  - No starvation – every master guaranteed bus access
  - Unpredictable, if masters have variable amount of data to communicate when the bus is granted

# Arbitration Policies (2)

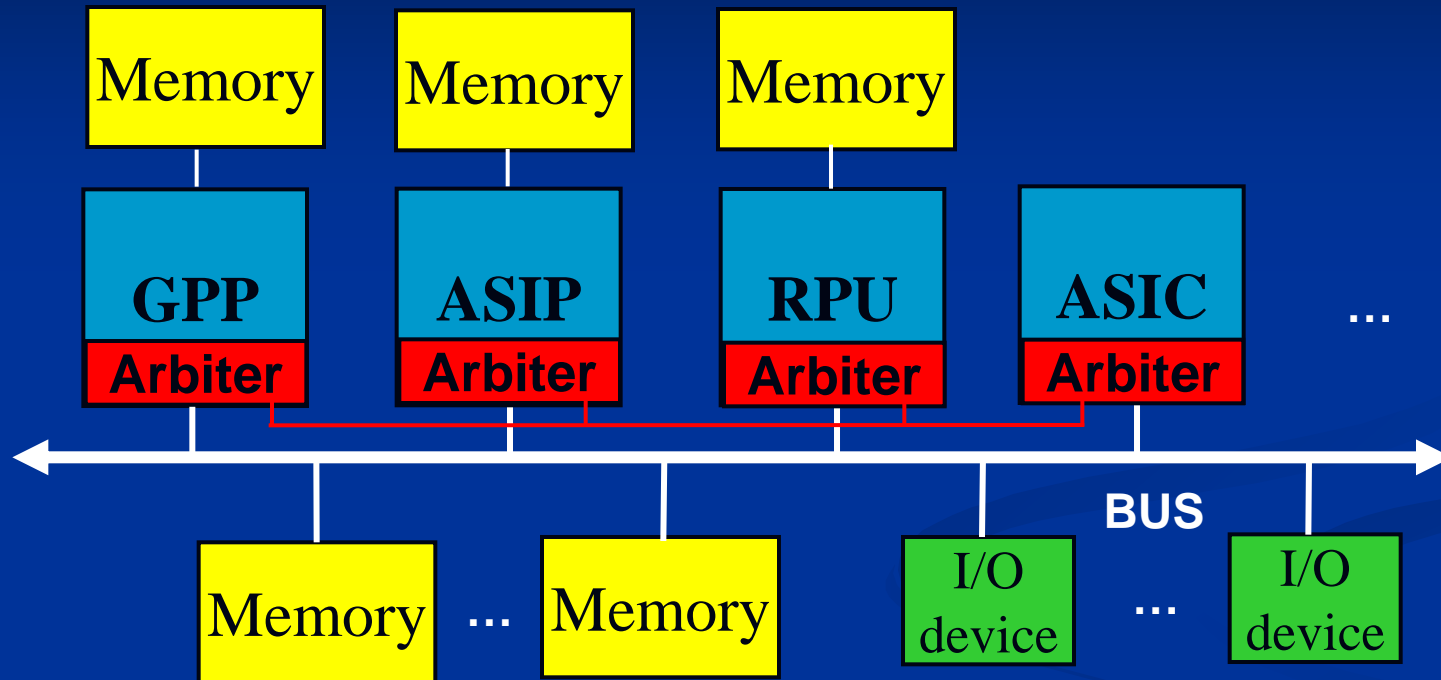
## ■ Time Division Multiple Access (TDMA)

- Assign slots (s) to masters (M) based on bandwidth requirements



- If a master does not have anything to read/write during its time slots, leads to low bus utilization
  - Choice of time slot length and number of slots are critical
  - Predictable behavior suitable for real-time Embedded Systems
- ## ■ TDMA/RR
- Two-level scheme
  - If master does not need to utilize its time slot, second level RR scheme grants access to another waiting master
  - Better bus utilization
  - Higher implementation cost for scheme (more logic, area)

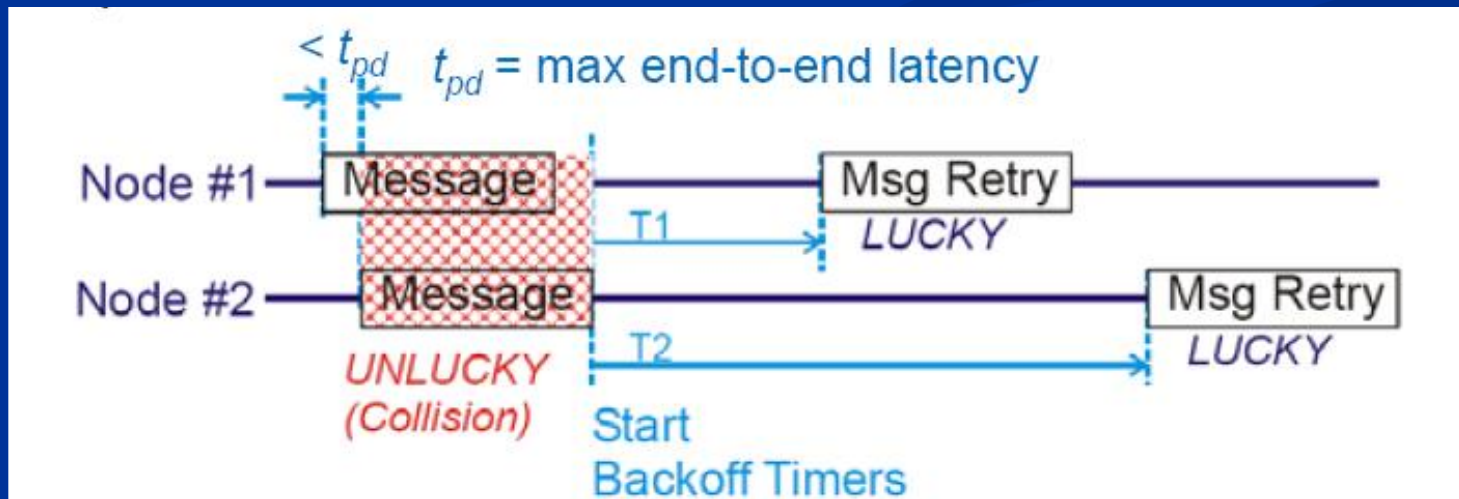
# Distributed Arbitration



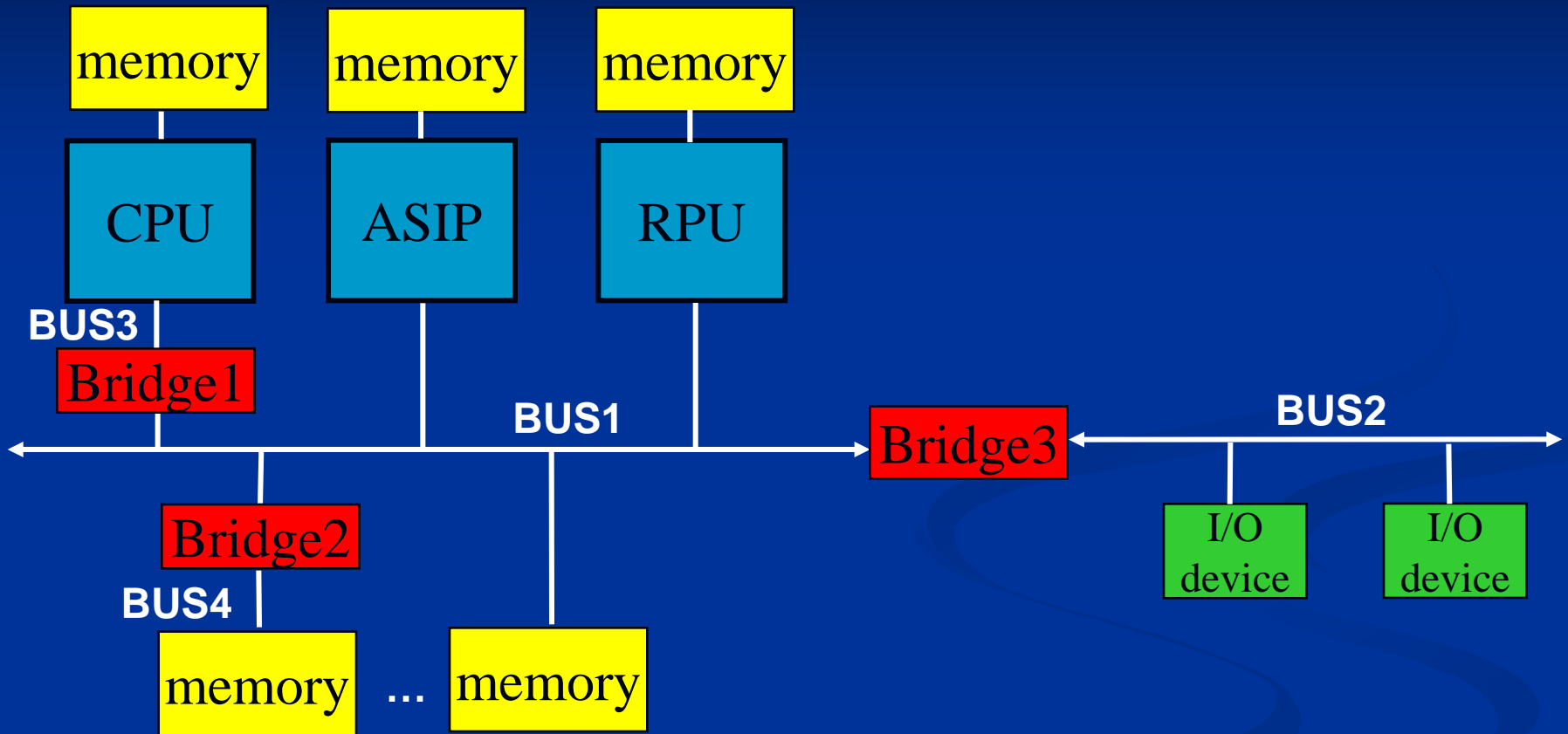
- Requires ***fewer signals*** compared to the centralized approach
- More hardware duplication, more logic/area, less scalable

# Distributed Arbitration Example: CSMA/CD

- Carrier Sense Multiple Access / Collision Detection
  - Try to avoid and detect collisions:
    - before starting to transmit, MASTER checks whether the bus is idle
    - if a collision is detected (several MASTERS started almost simultaneously), wait for some time (back-off timer)
  - Repeated collisions result in increasing back-off times



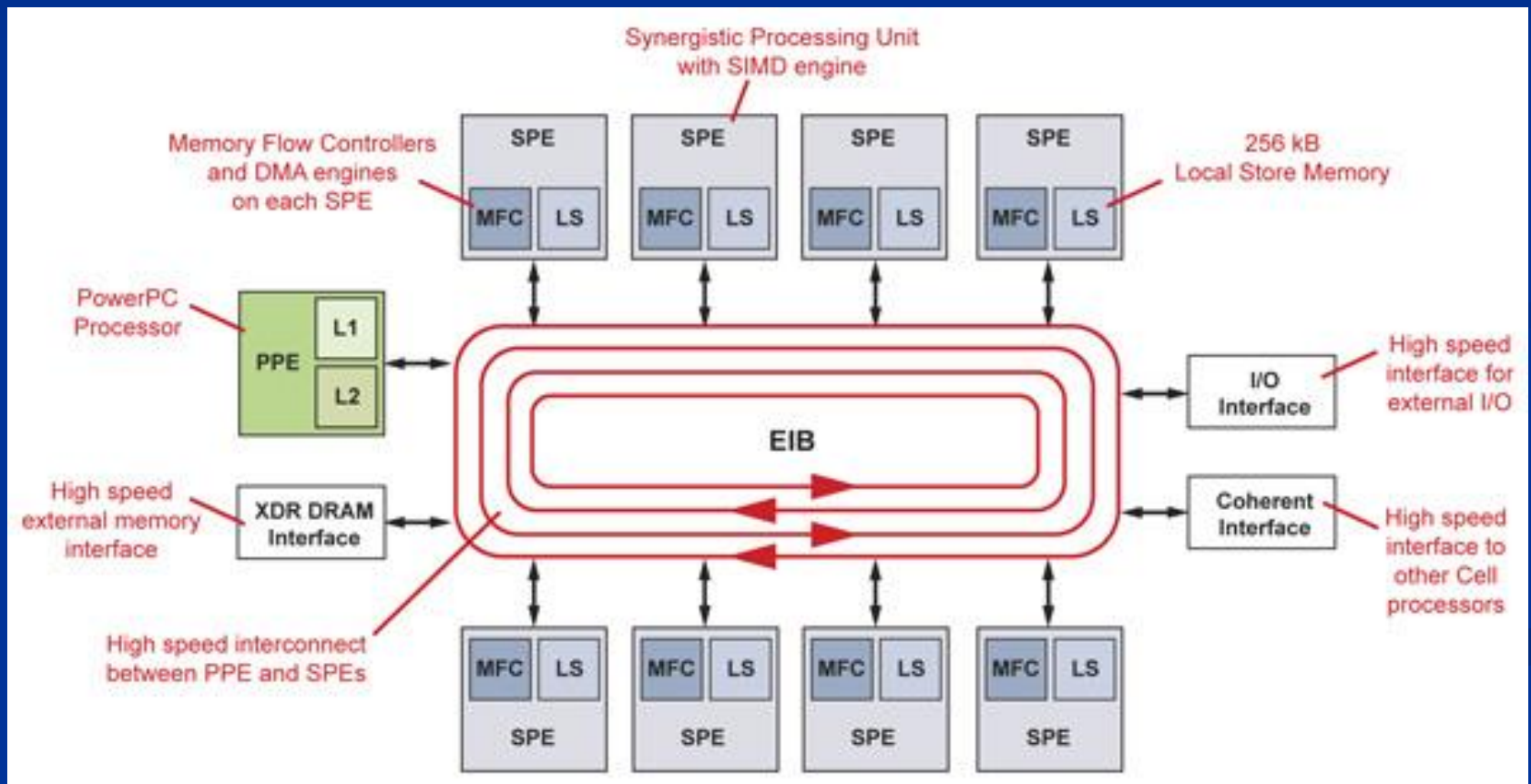
# How to Connect Different Busses and Interfaces?



- Use Bus Bridges
- They act as **slave** on one side and **master** on the other

# Multiple Busses

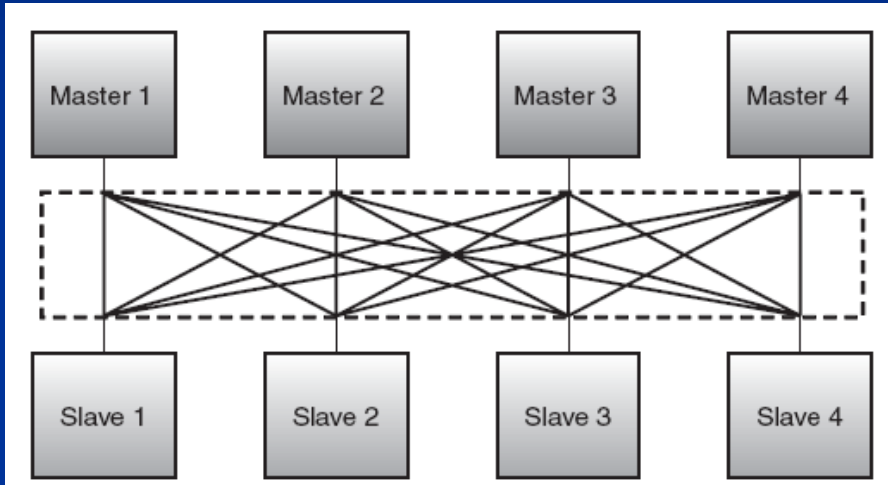
- IBM Cell ring bus communication architecture



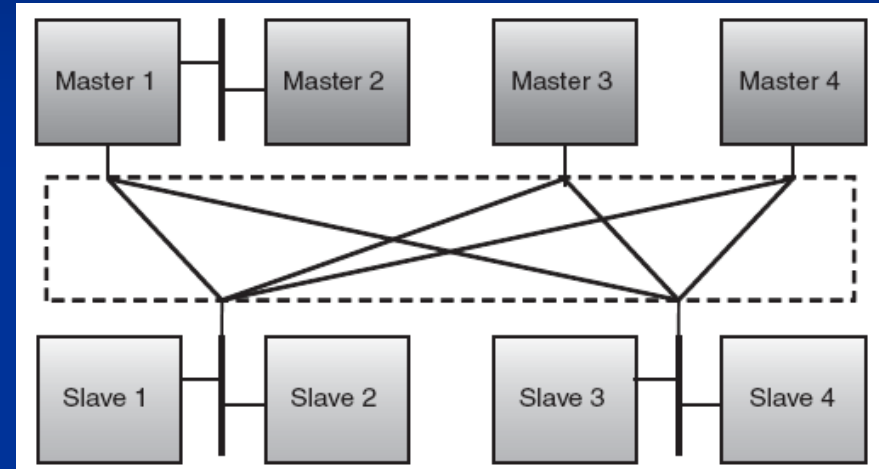


# Crossbar Switch

- Full-crossbar/matrix bus  
Point-to-Point



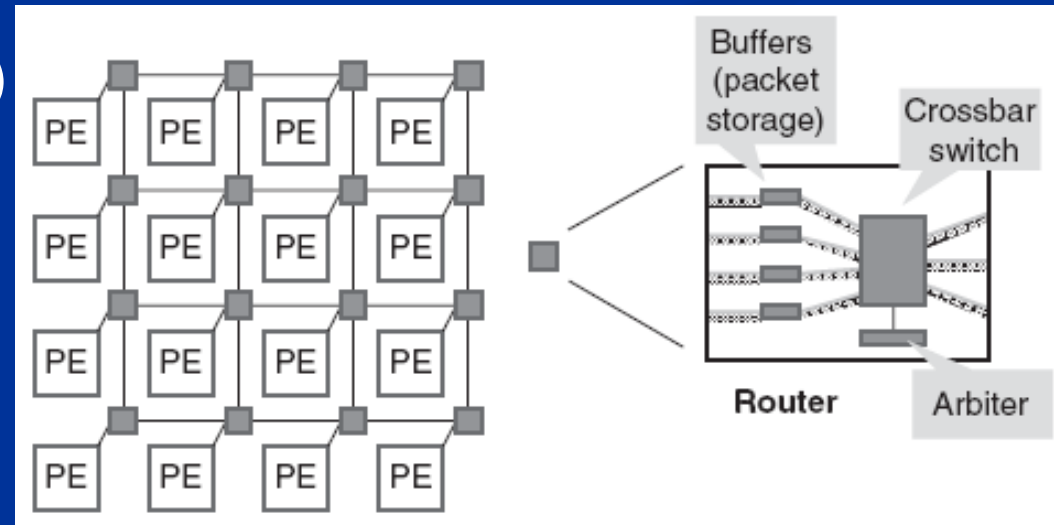
- Partial-crossbar/matrix bus



- Disadvantages:
  - lots of wires and multiplexers that take space!
  - Not very scalable

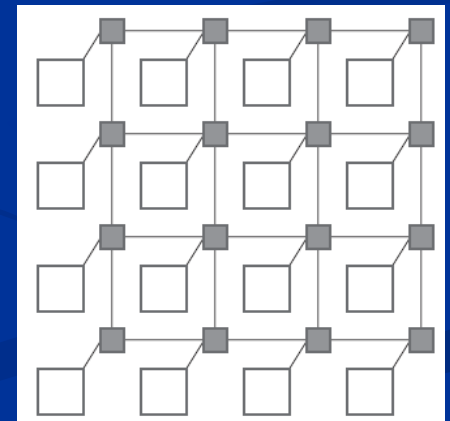
# Network-On-Chip: Introduction

- NoC is a packet switched on-chip communication network
  - It uses packets to route data from the source to the destination PE via the network fabric
- NoC consists of
  - network interfaces (NI)
  - switches (routers)
  - interconnection links (wires)



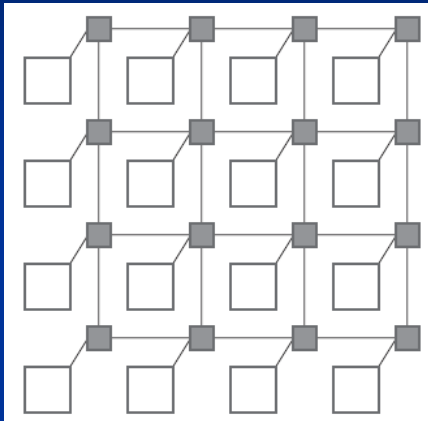
# Network-On-Chip: Introduction

- NoCs are an attempt to scale down the concepts of large-scale networks
  - apply them to the Embedded SoC domain
- NoC Properties
  - Reliable and predictable electrical and physical properties (packets never lost)
  - Regular geometry that is scalable
  - Higher bandwidth
  - Reusable components
    - Buffers, arbiters, routers, protocol stack

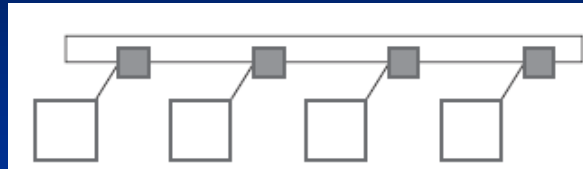


# Network-On-Chip: Topology

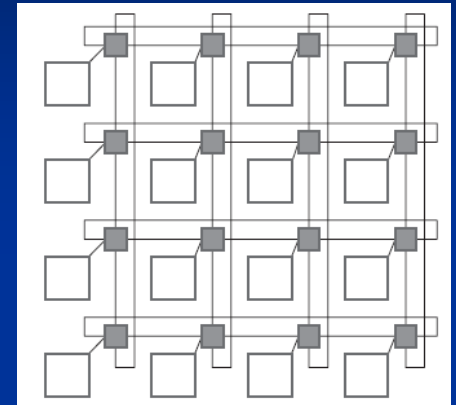
## 2-D Mesh



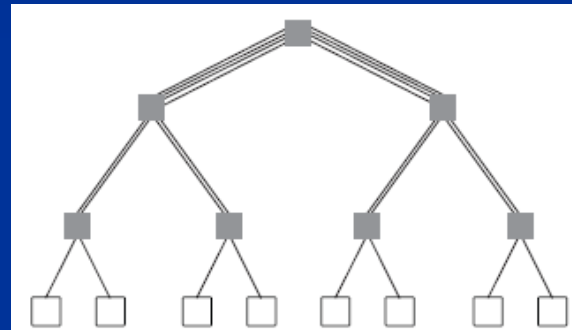
## 1-D Torus



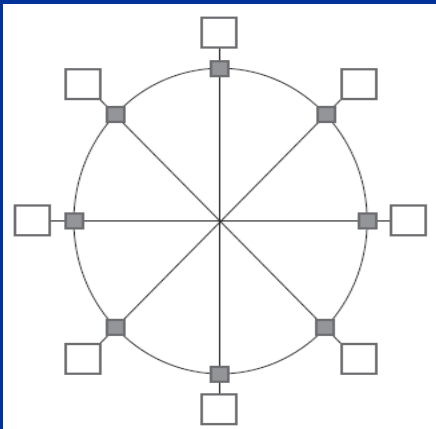
## 2-D Torus



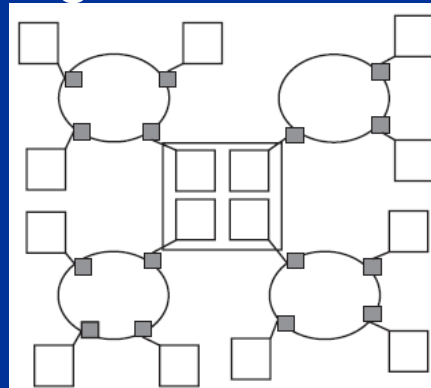
## Fat Tree



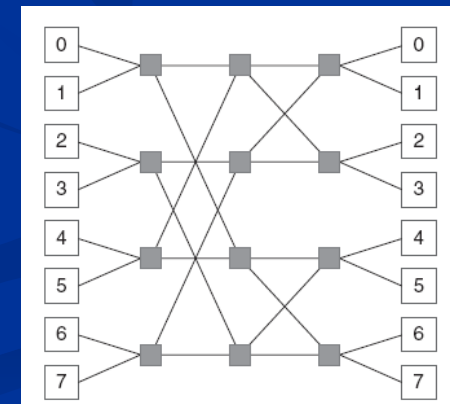
## Octagon



## Irregular or Ad-hoc



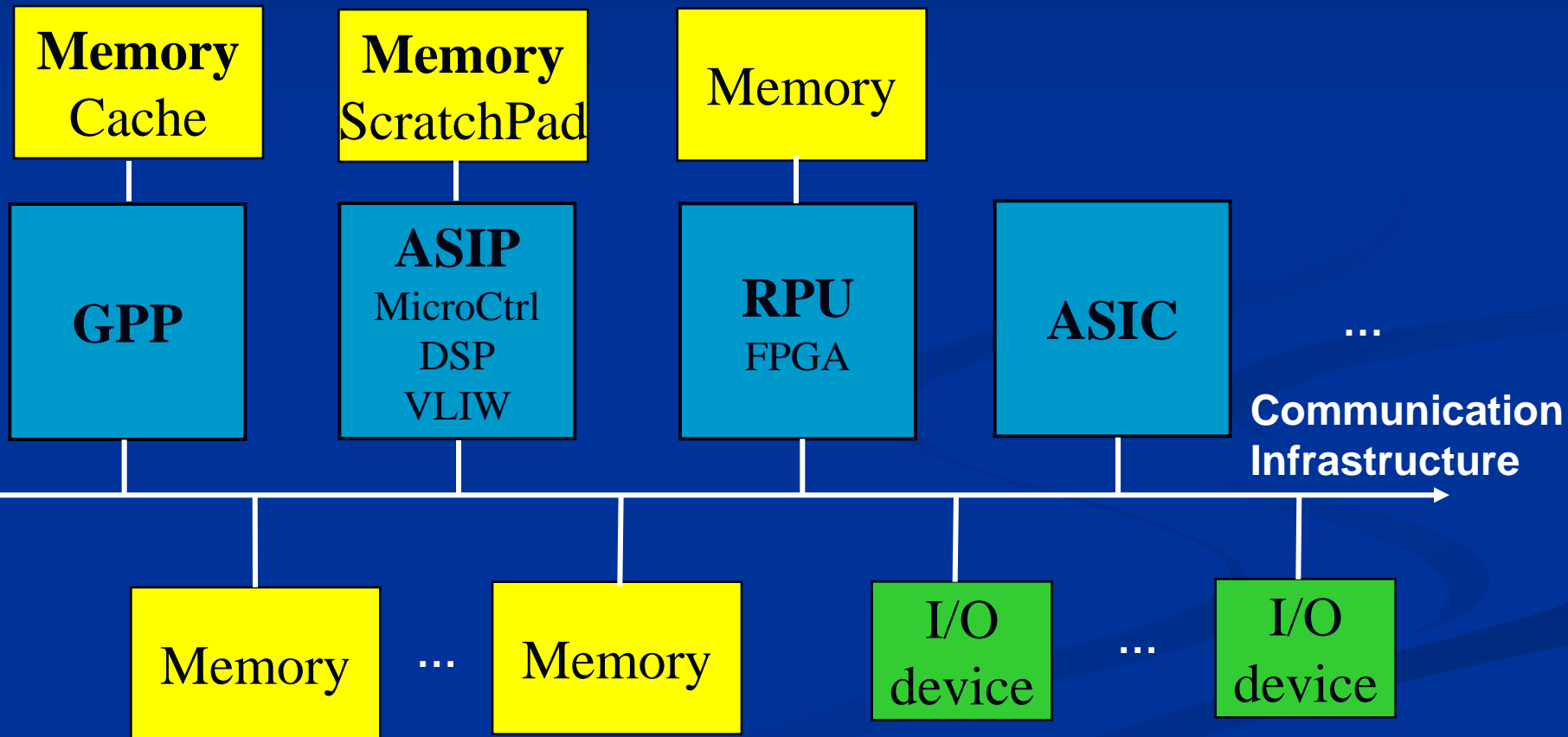
## Butterfly



# Network-On-Chip: Issues

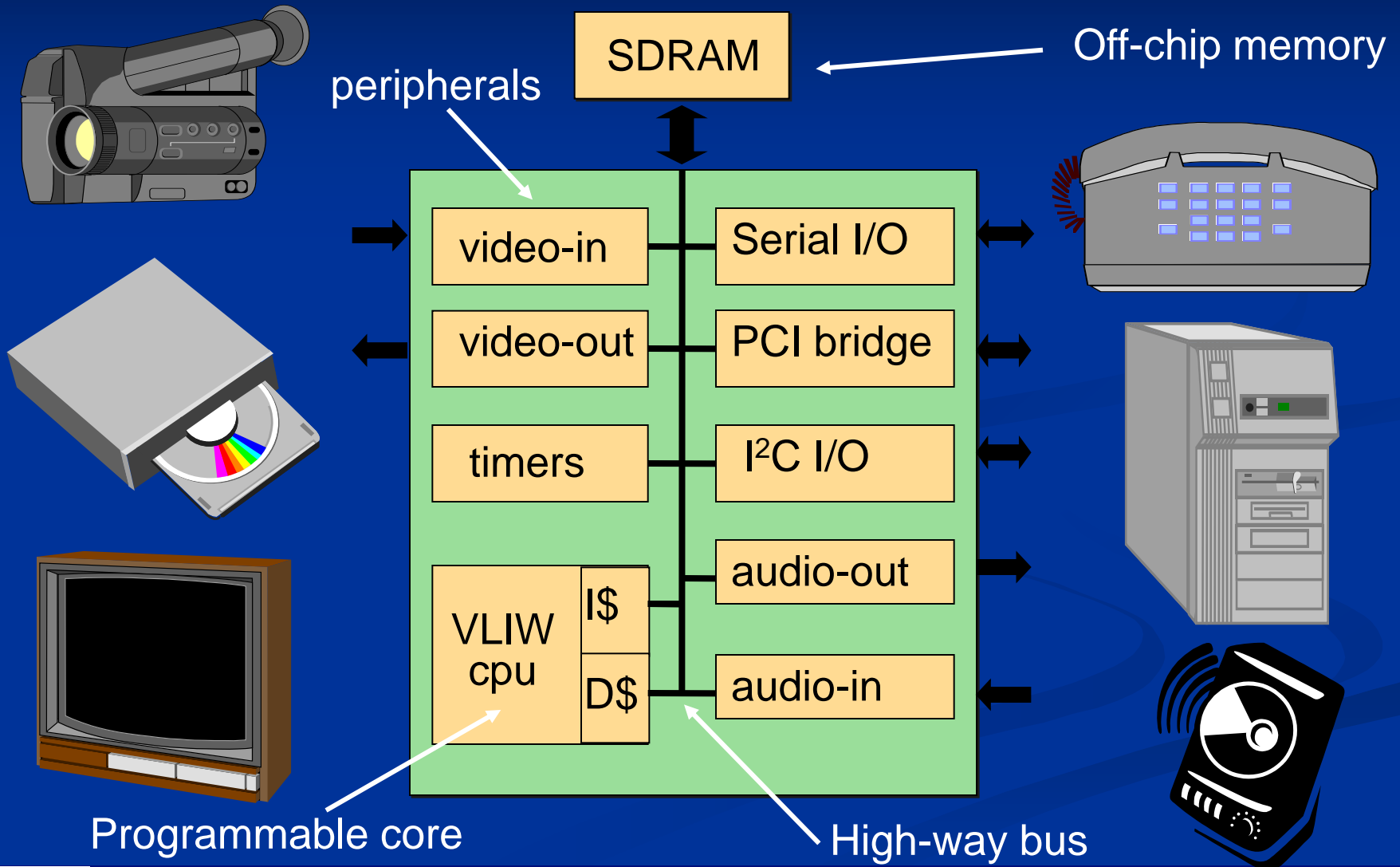
- NO wide adoption in Industry because of
  - High Power consumption
    - several times greater compared to buses
    - routers (their buffers inside) are power hungry
  - High Latency
    - additional delay to packetize/de-packetize data at NIs
    - delays at routers along the routing path
- Still active research is needed to solve the issues

# All this could be on a single chip!!! System-on-chip (SoC)

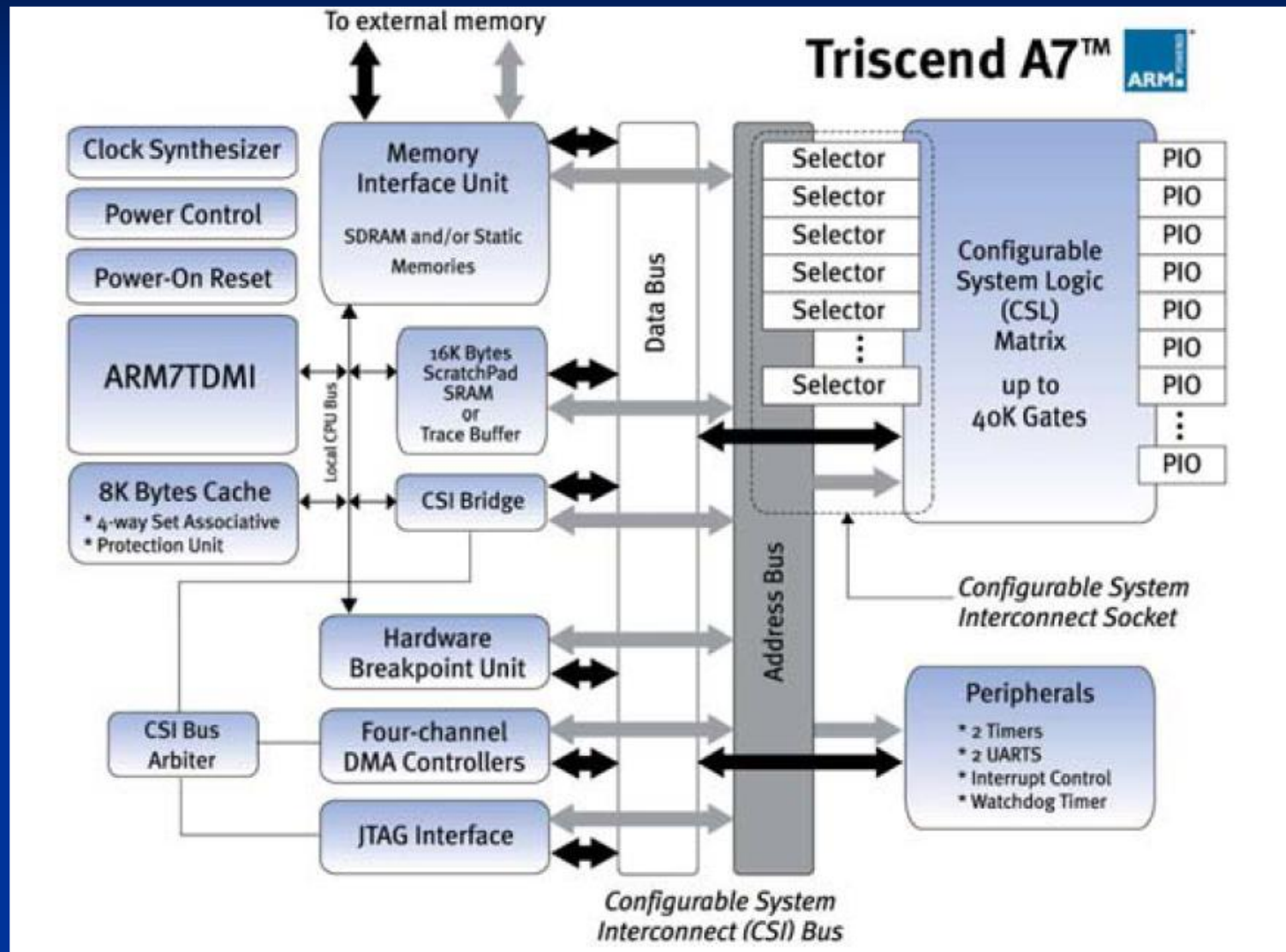


# Example of SoC (one processor with peripherals)

Philips Trimedia Chip

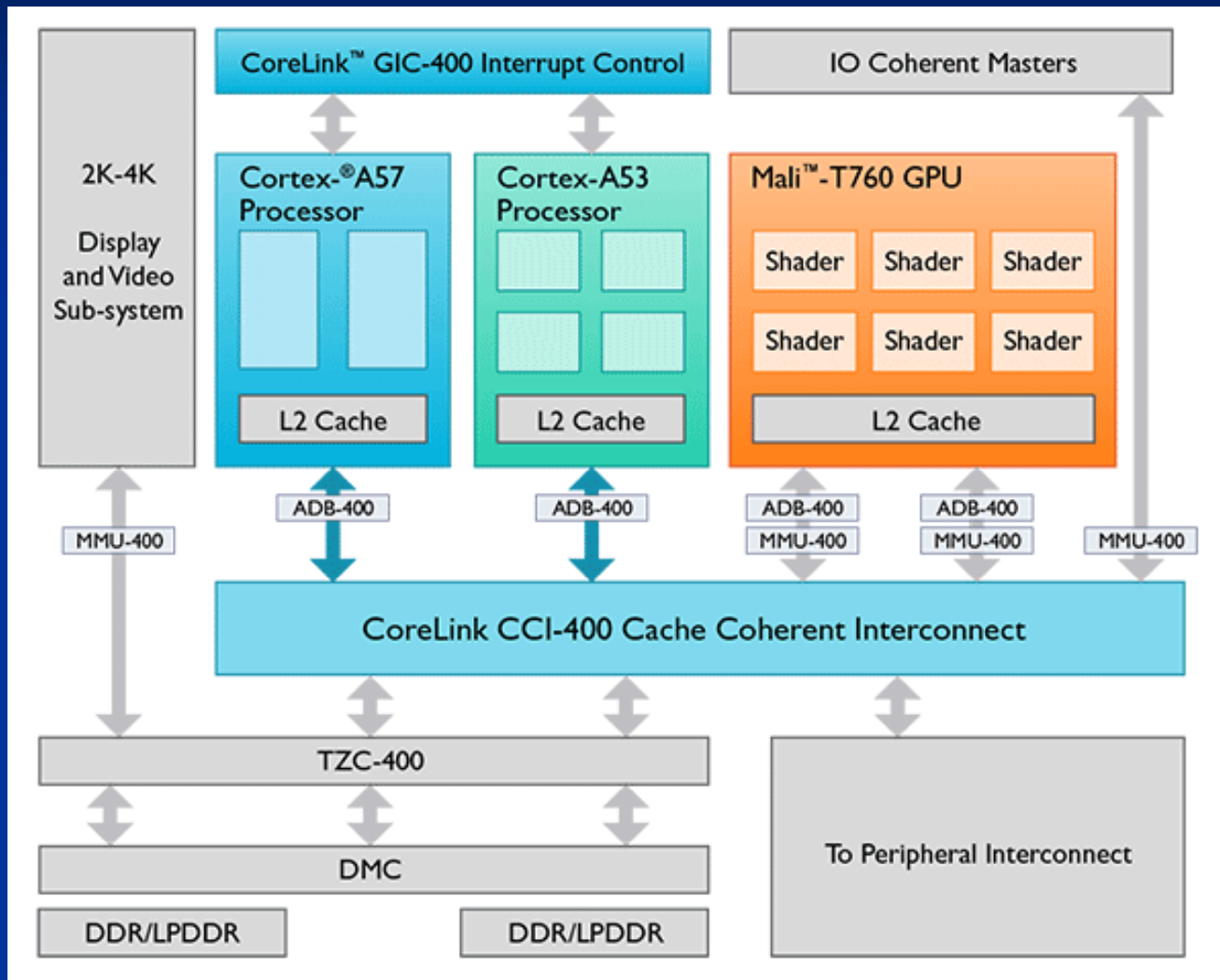


# Examples of Configurable SoC (one processor + RPU + peripherals)



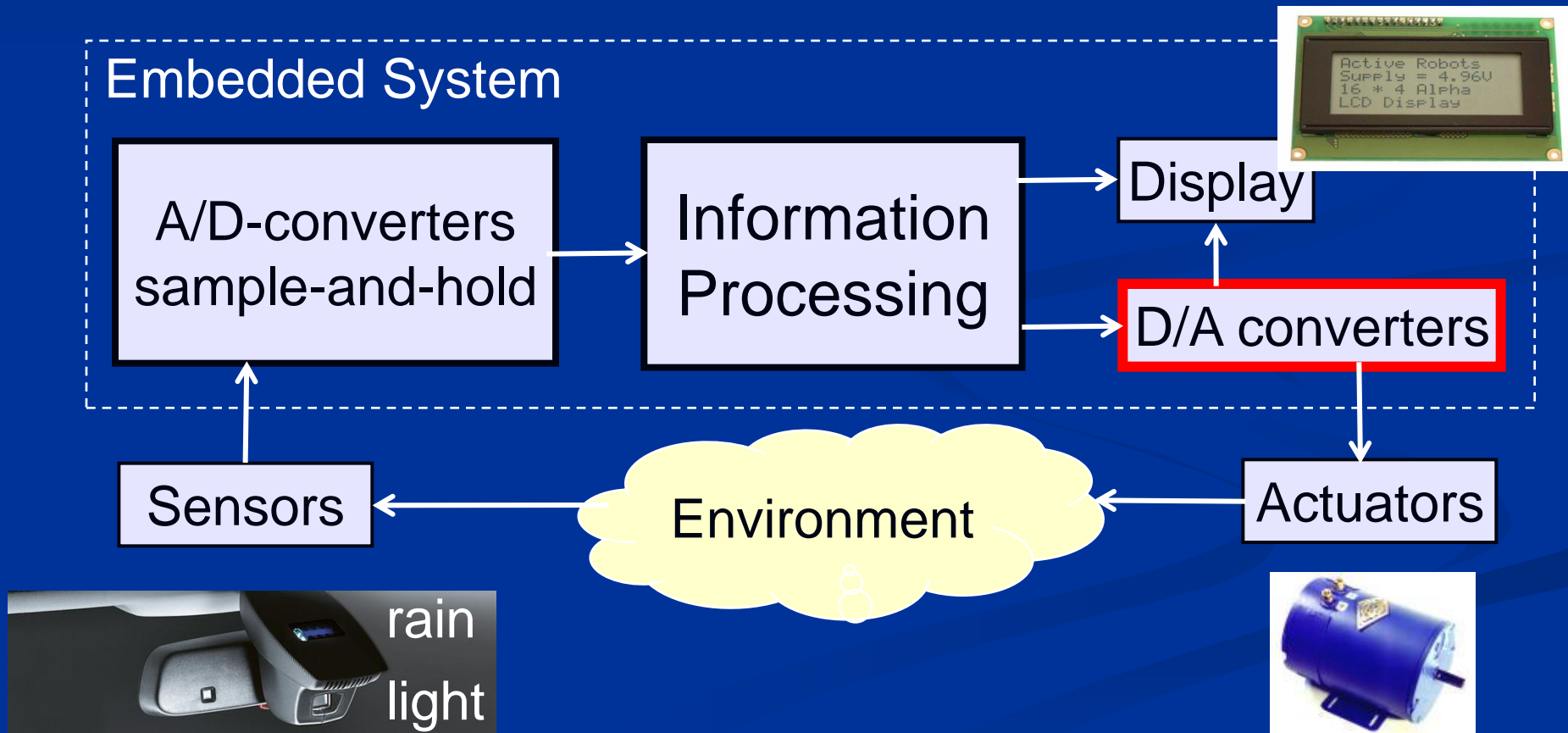


# Example of Multi-Processor SoC (ARM big.LITTLE mobile chip)



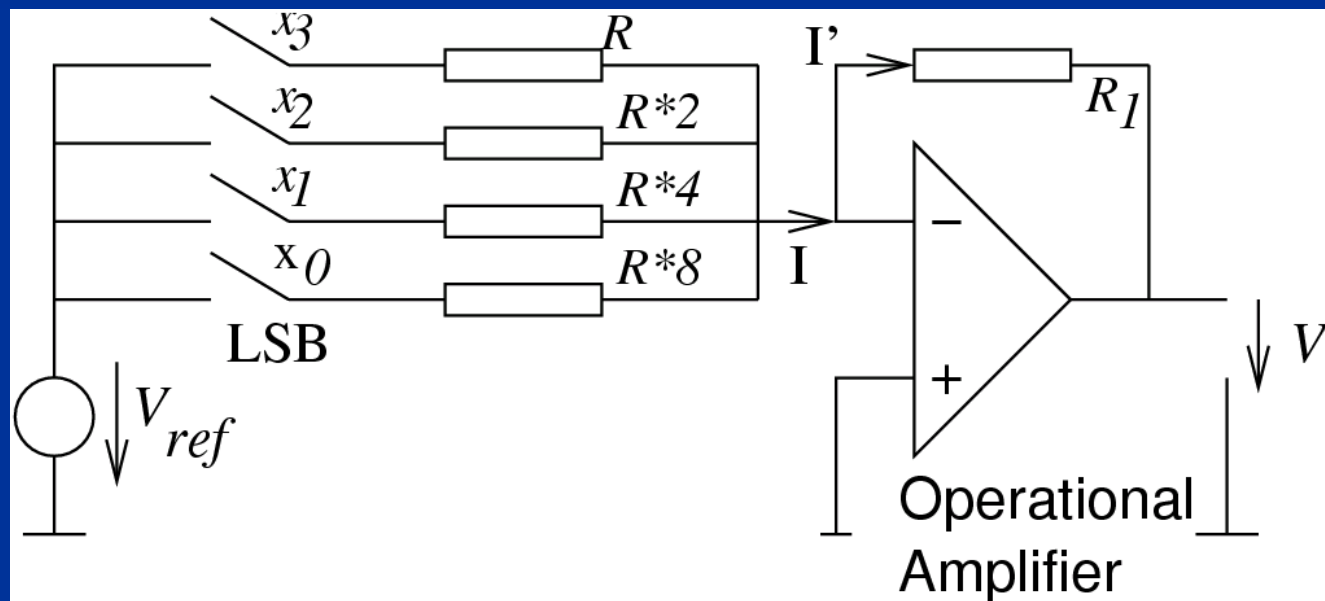
# Embedded Systems Hardware

Embedded Systems hardware is frequently used in a loop (*“hardware in a loop”*):

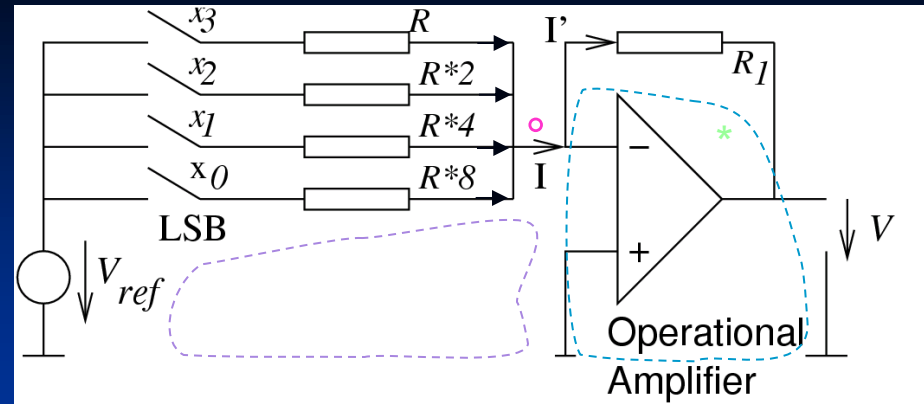


# Digital-to-Analog (D/A)-converters

Example of a simple 4-bit D/A-converter:



# Output voltage $\sim$ no. represented by $x$



Due to Kirchhoff's laws :



$$I = x_3 \times \frac{V_{ref}}{R} + x_2 \times \frac{V_{ref}}{2 \times R} + x_1 \times \frac{V_{ref}}{4 \times R} + x_0 \times \frac{V_{ref}}{8 \times R} = \frac{V_{ref}}{8 \times R} \times \sum_{i=0}^3 x_i \times 2^i$$

Loop rule\*:

$$V + R_1 \times I' = 0$$

Junction rule°:

$$I = I'$$

Hence:

$$V + R_1 \times I = 0$$

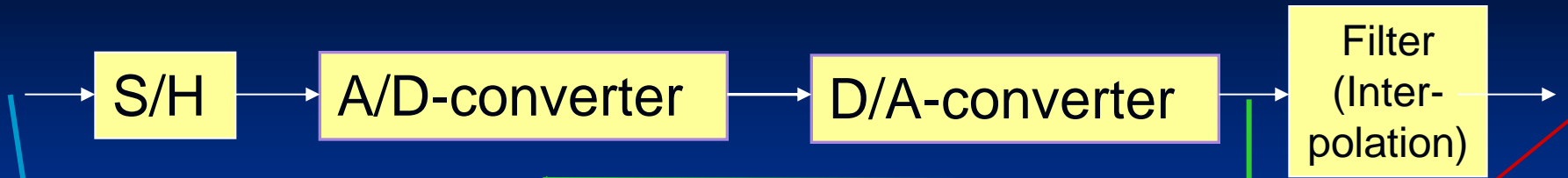
Finally:

$$-V = V_{ref} \times \frac{R_1}{8 \times R} \sum_{i=0}^3 x_i \times 2^i = V_{ref} \times \frac{R_1}{8 \times R} \times nat(x)$$

$I \sim nat(x)$ , where  $nat(x)$ : natural number represented by  $x$ ;

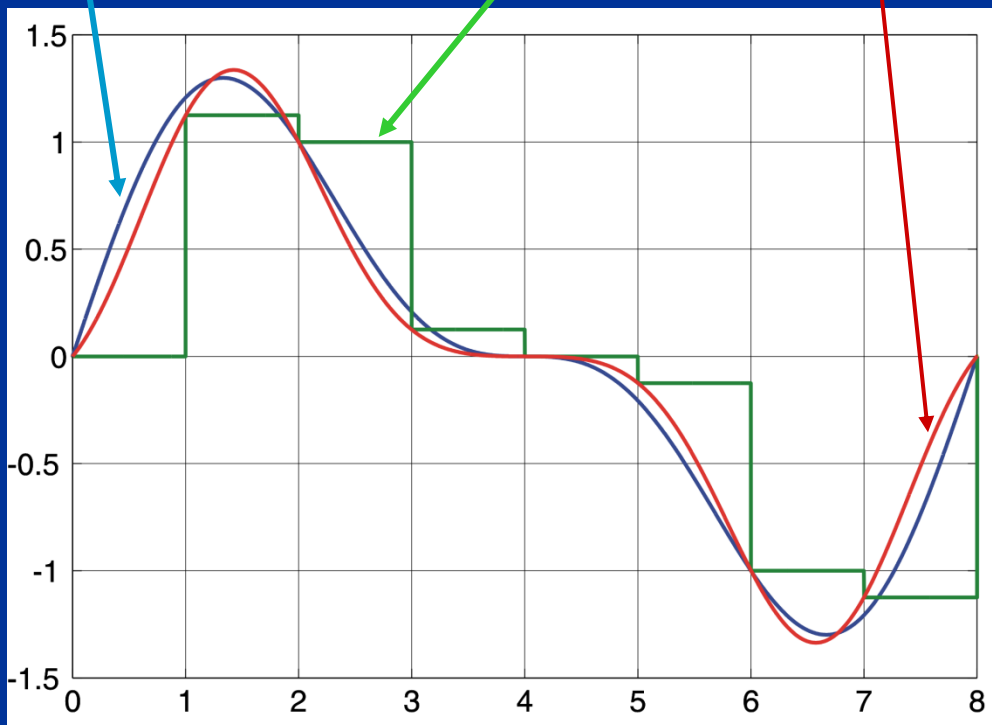
Op-amp turns current  $I \sim nat(x)$  into a voltage  $\sim nat(x)$

# Reconstruction Analog from Digital?



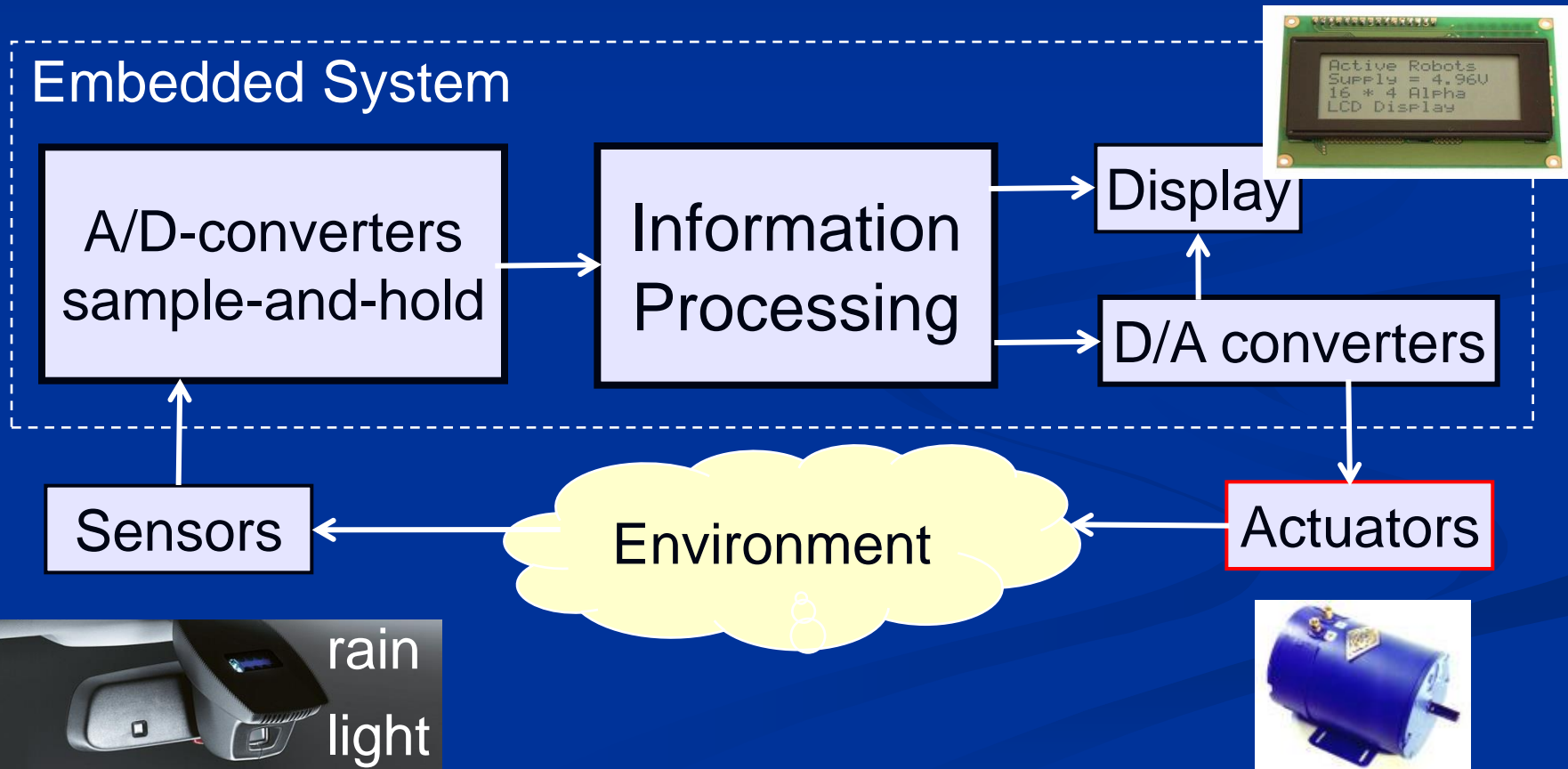
## ■ According to *Nyquist* theorem:

- Analog signal to sample-and-hold can be precisely reconstructed if
  - sampling frequency  $\geq$  double the highest frequency in the analog signal



# Embedded Systems Hardware

Embedded Systems hardware is frequently used in a loop (*“hardware in a loop”*):



# Actuators

- Convert an input command/signal to a physical stimulus
  - heat, light, sound, pressure, magnetism, or mechanical motion
- May require analog input signal
- Examples of physical stimulus by different actuators
  - Physical motion
    - Robotic arms, etc.
    - Pneumatic systems
  - Light
    - LEDs, displays, etc.
  - Rotation
    - DC and stepper motors
  - Sound
    - Loudspeakers, etc.
  - ...

