Introduction

Amir Mahdi Hosseini Monazzah

Room 332,
School of Computer Engineering,
Iran University of Science and Technology,
Tehran, Iran.

monazzah@iust.ac.ir

Fall 2024

Overview

1. What is an embedded system?
   - Definition
   - Embedded system example
   - Design principles of an embedded systems

Embedded system

- Two types of computing
  1. Desktop–produced millions/year
  2. Embedded–billions/year
- Non-embedded systems
  - PCs, servers, and notebooks
- The future of computing!
  - Automobiles, entertainment, communication, aviation, handheld devices, military and medical equipment.

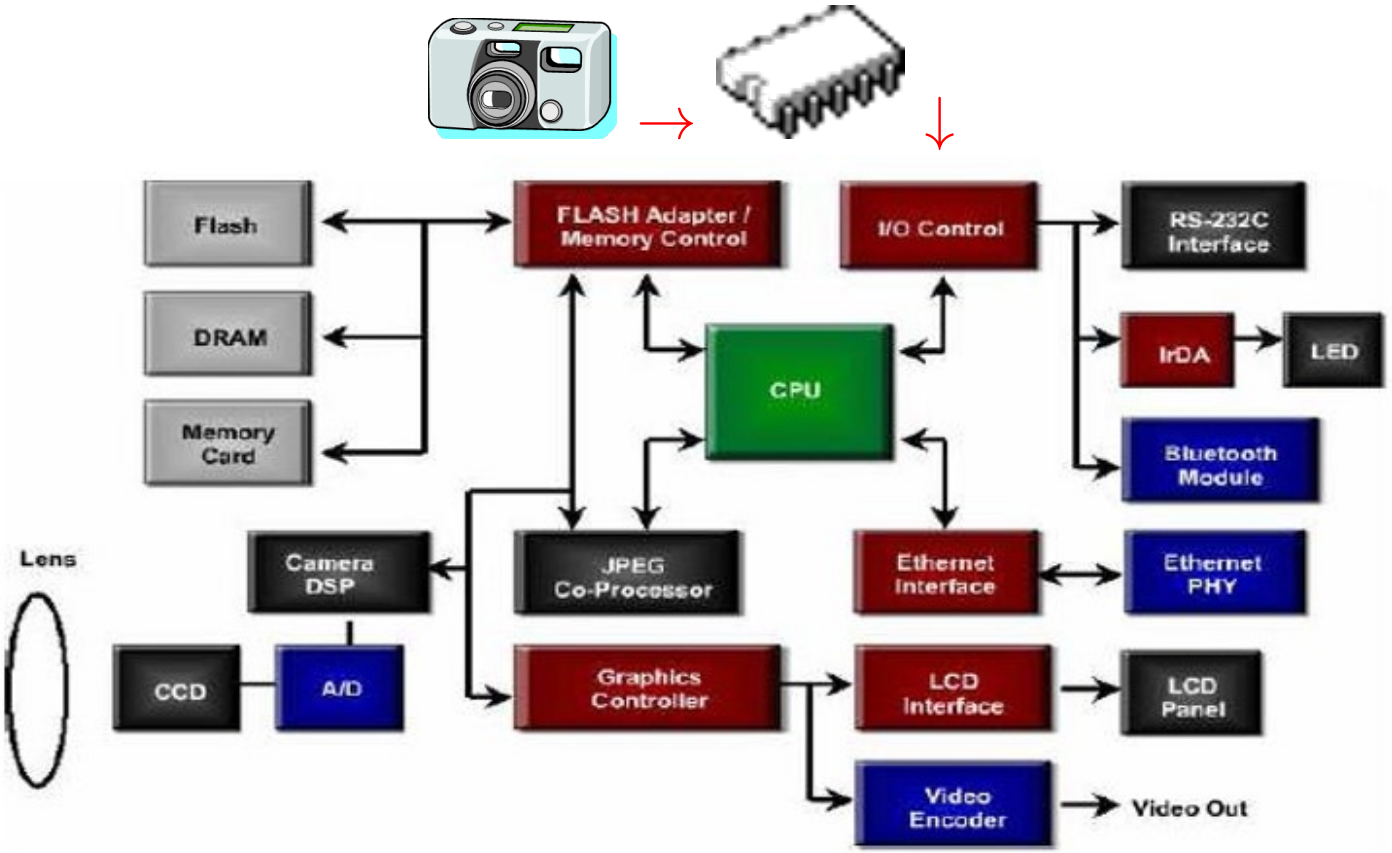
Embedded system (cont.)

Thus, how can we define embedded system?

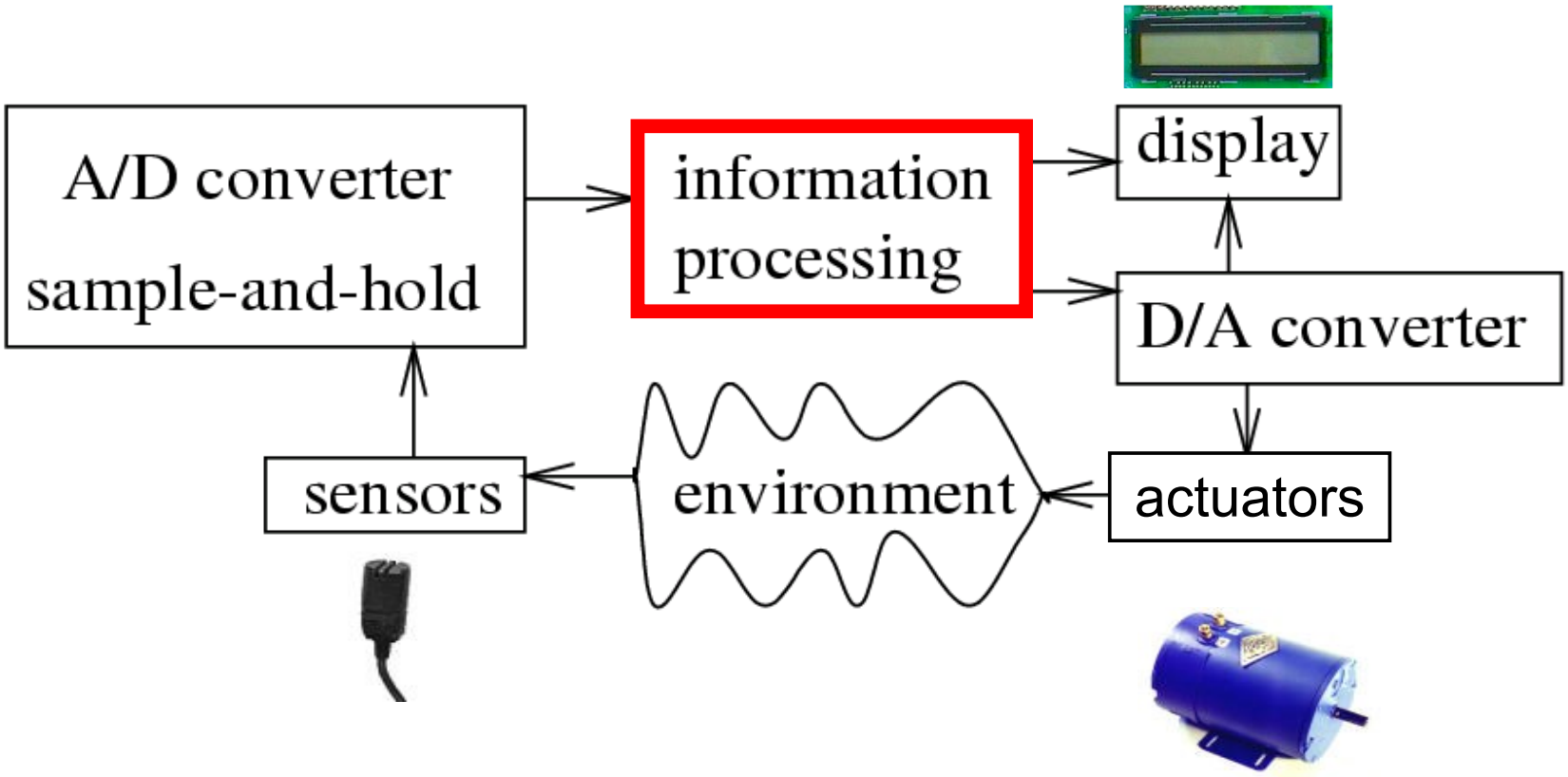
1. Information processing systems embedded into a larger product.
   - Computers camouflaged as non-computers
2. Main reason for buying is not information processing.
3. Devices other than desktop PCs, servers, and notebooks
   - Electricity running through
   - Perform something intelligent
4. Hardware/software which form a component of a larger system, but are concealed from user.
5. The future of computing!

An example of an embedded system

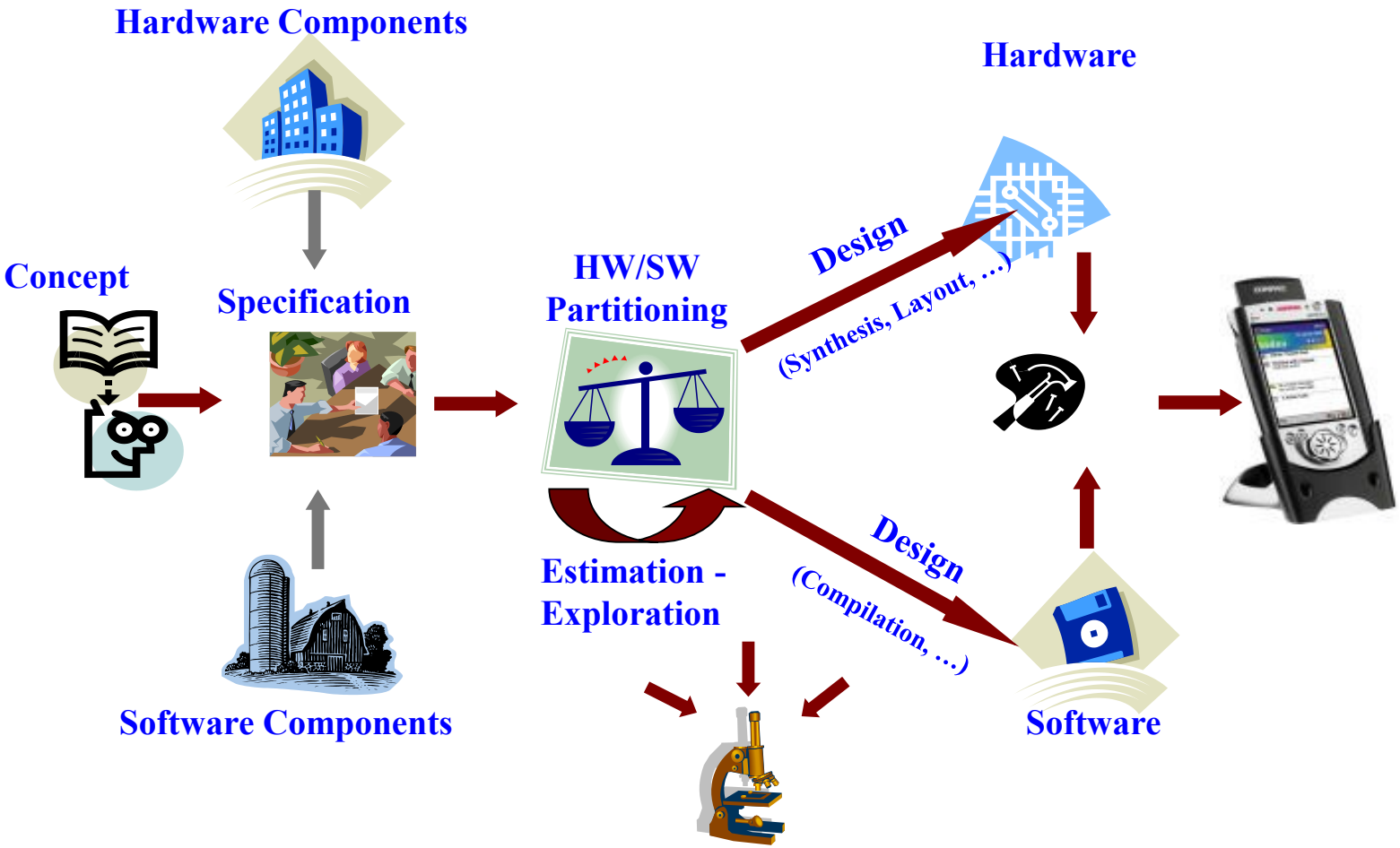
Digital camera block diagram



Simplified block diagram of an embedded system



Embedded system design flow



Components of an embedded systems

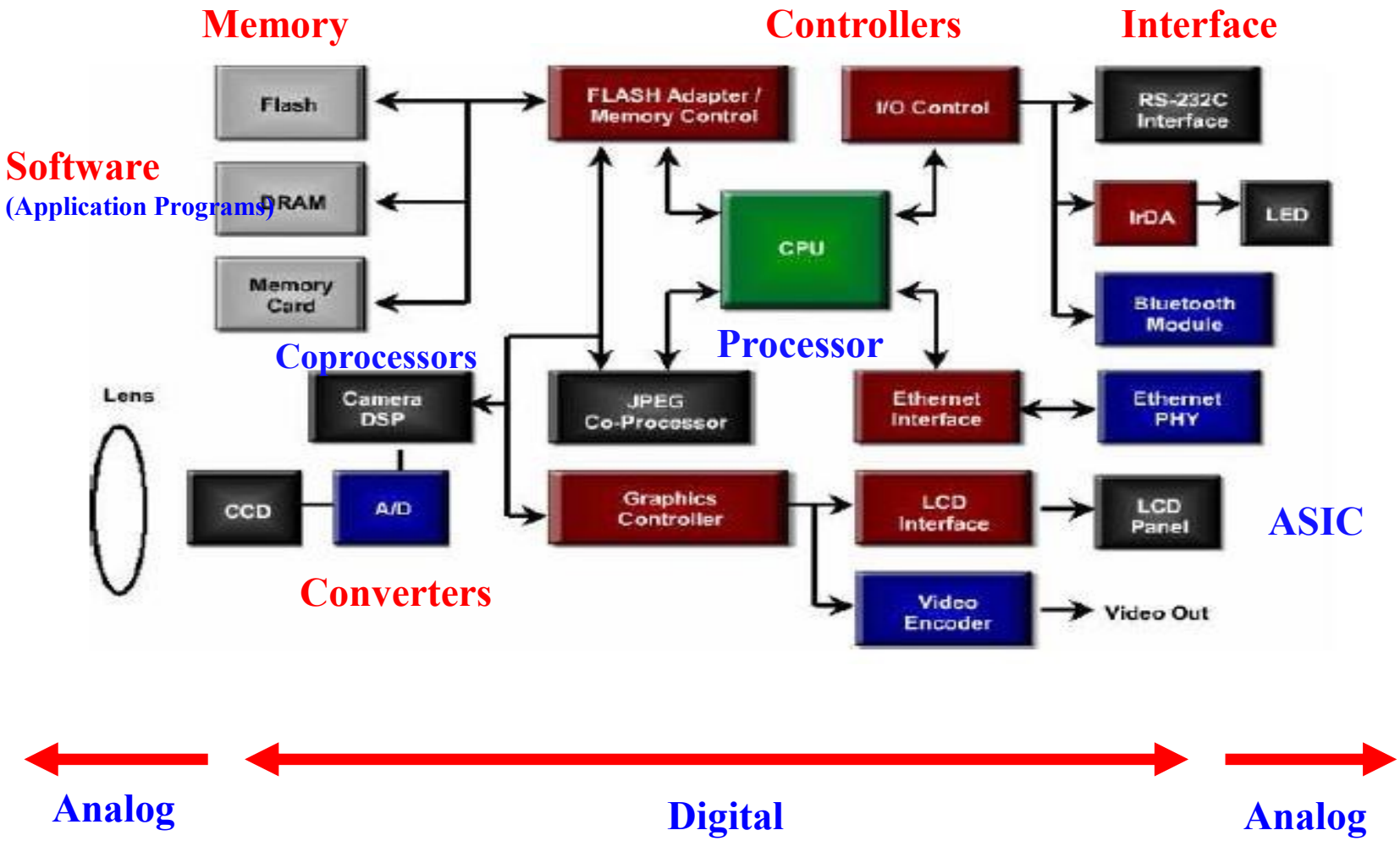
Hardware components

- Analog components
  - Sensors, actuators, controllers, ...
- Digital components
  - Processor, co-processors
  - Memories
  - Controllers, buses
  - Application Specific Integrated Circuits (ASIC)
- Converters
  - A2D
  - D2A

Software components

- Application programs
- Exception handlers

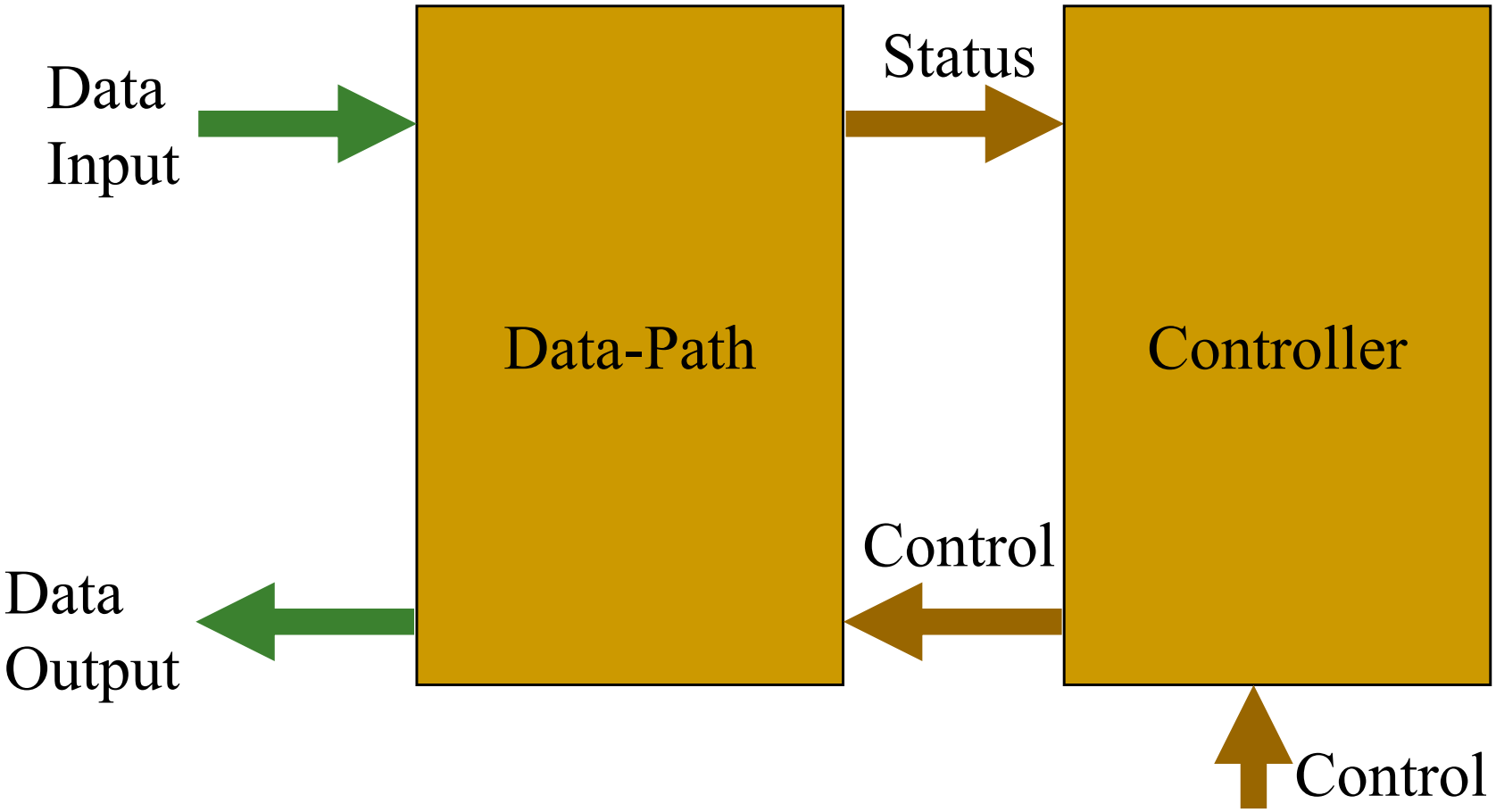
Digital camera hardware components: processor



Processors

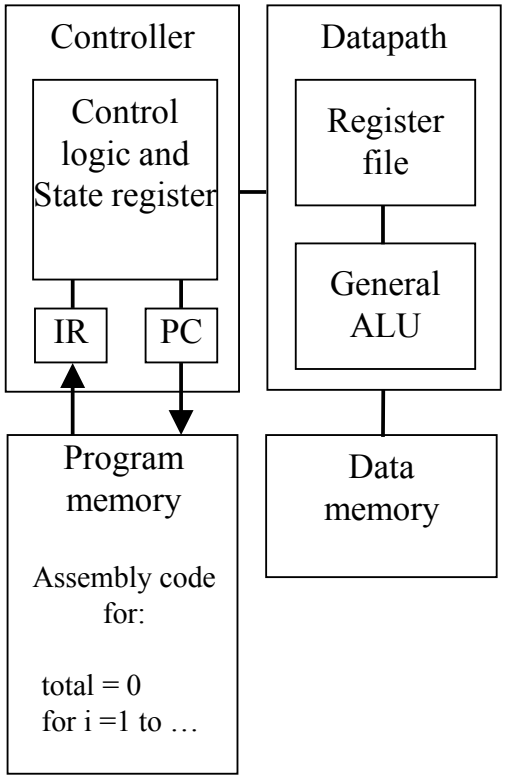
- What is a processor?
  - Artifact that computes (runs algorithms)
  - Controller and data-path
- General-Purpose (GP) processors
  - Variety of computation tasks
  - Functional flexibility and low cost at high volumes (maybe)
  - Slow and power hungry
- Single-Purpose (SP) processors (or ASIC)
  - One particular computation task
  - Fast and power efficient
  - Functional inflexibility and high cost at low volumes (maybe)

GP/SP processor architecture



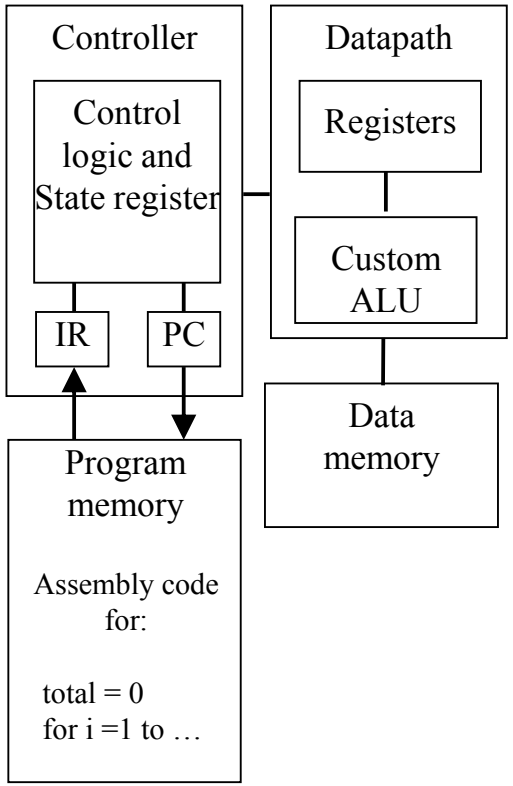
General-purpose processors

- Programmable device used in a variety of applications
  - Also known as "microprocessor"
- Features
  - Program memory
  - General datapath with large register file and general ALU
- User benefits
  - Low time-to-market
  - High flexibility
- Examples
  - Pentium, Athlon, PowerPC



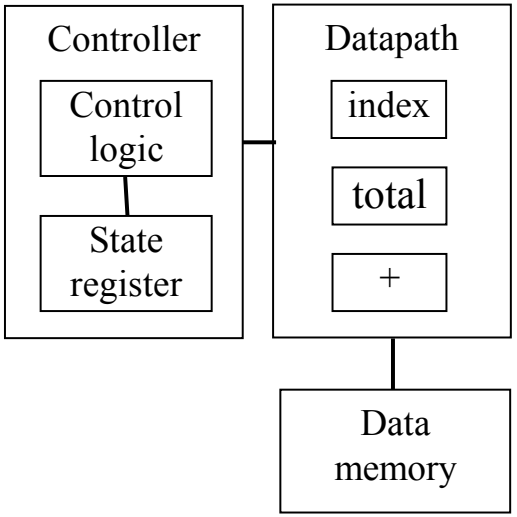
Application-specific IS processors (ASIPs)

- Programmable processor optimized for a particular class of applications having common characteristics
  - Compromise between general-purpose and ASIC (custom hardware)
- Features
  - Program memory
  - Optimized datapath
  - Special functional units
- User benefits
  - Some flexibility, good performance, size and power
- Examples
  - DSPs, video signal processors, network processors, ...



Application-Specific ICs (ASICs)

- Digital circuit designed to execute exactly one program
  - Coprocessor, hardware accelerator
- Features
  - Contains only the components needed to execute a single program
  - No program memory
- User benefits
  - Fast
  - Low power
  - Small size
- Examples
  - Digital voice recorder, high-efficiency bitcoin miner, ...



GP vs. SP processors

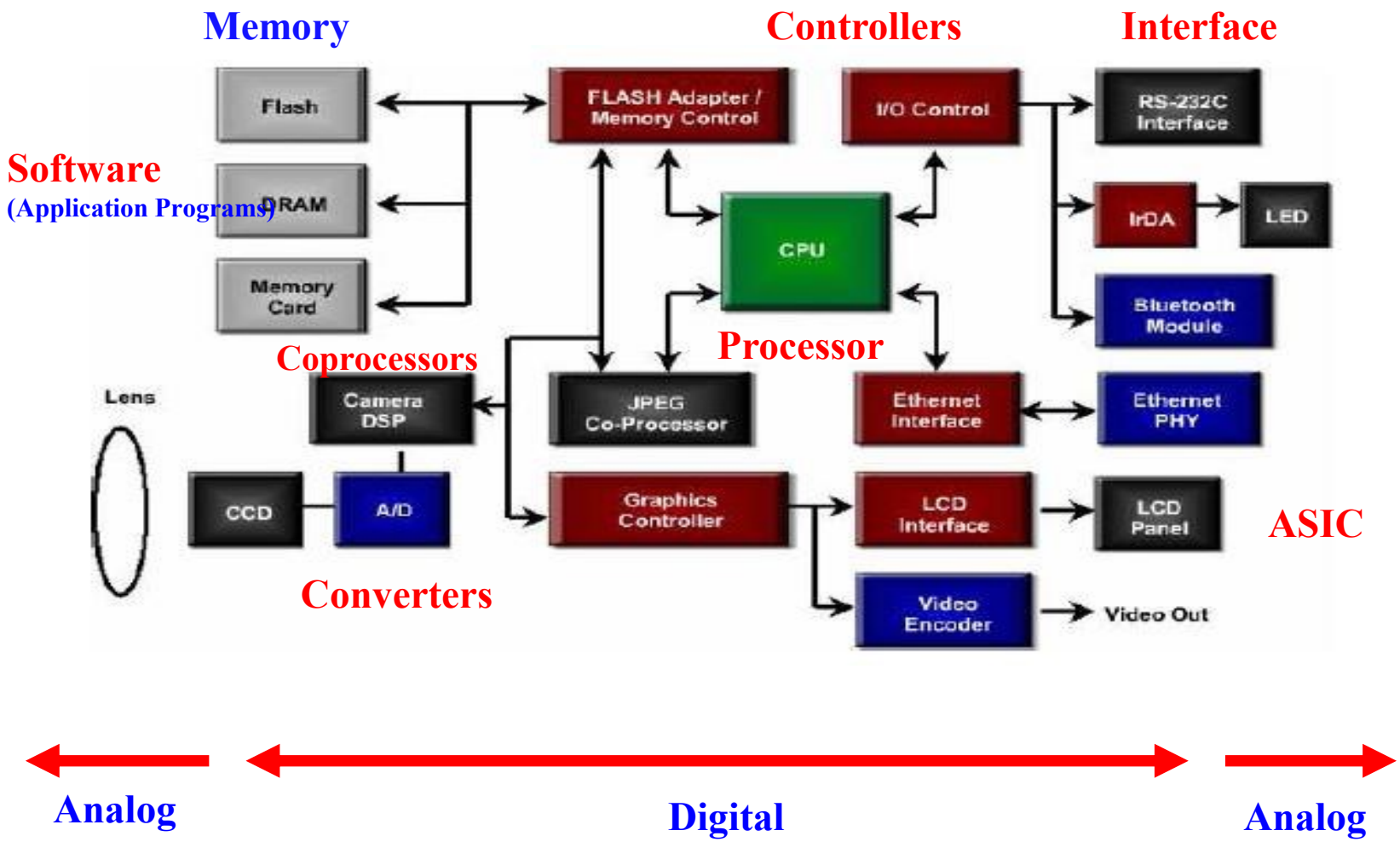
GP

- Programmable controller
  - Control logic is stored in memory
  - Fetch/decode overhead
- Highly general data-path
  - Typical bit-width (8, 16, 32, 64)
  - Complete set of arithmetic/logic units
  - Large set of registers
- High NRE/sale-volume

ASIC

- Hardwired controller
  - No need for program memory and cache
  - No fetch/decode overhead
- Highly tuned data-path
  - Custom bit-width
  - Custom arithmetic/logic units
  - Custom set of registers
- Low NRE/sale-volume

Digital camera hardware components: memory (from here: session 3)



Memory

- What is a memory?
  - Artifact that stores bits
  - Storage fabric and access logic
- Write-ability
  - Manner and speed a memory can be written
- Storage-permanence
  - Ability of memory to hold stored bits after they are written
- Many different types of memories
  - Flash, SRAM, DRAM, STT-MRAM, ...

Write-ability

Ranges of write-ability

- High end
  - Processor writes to memory simply and quickly
  - E.g., RAM
- Middle range
  - Processor writes to memory, but slower
  - E.g., FLASH, EEPROM

Write-ability

- Lower range
  - Special equipment, “programmer”, must be used to write to memory
  - E.g., EPROM, OTP ROM
- Low end
  - Bits stored only during fabrication
  - E.g., Mask-programmed ROM

Storage-permanence

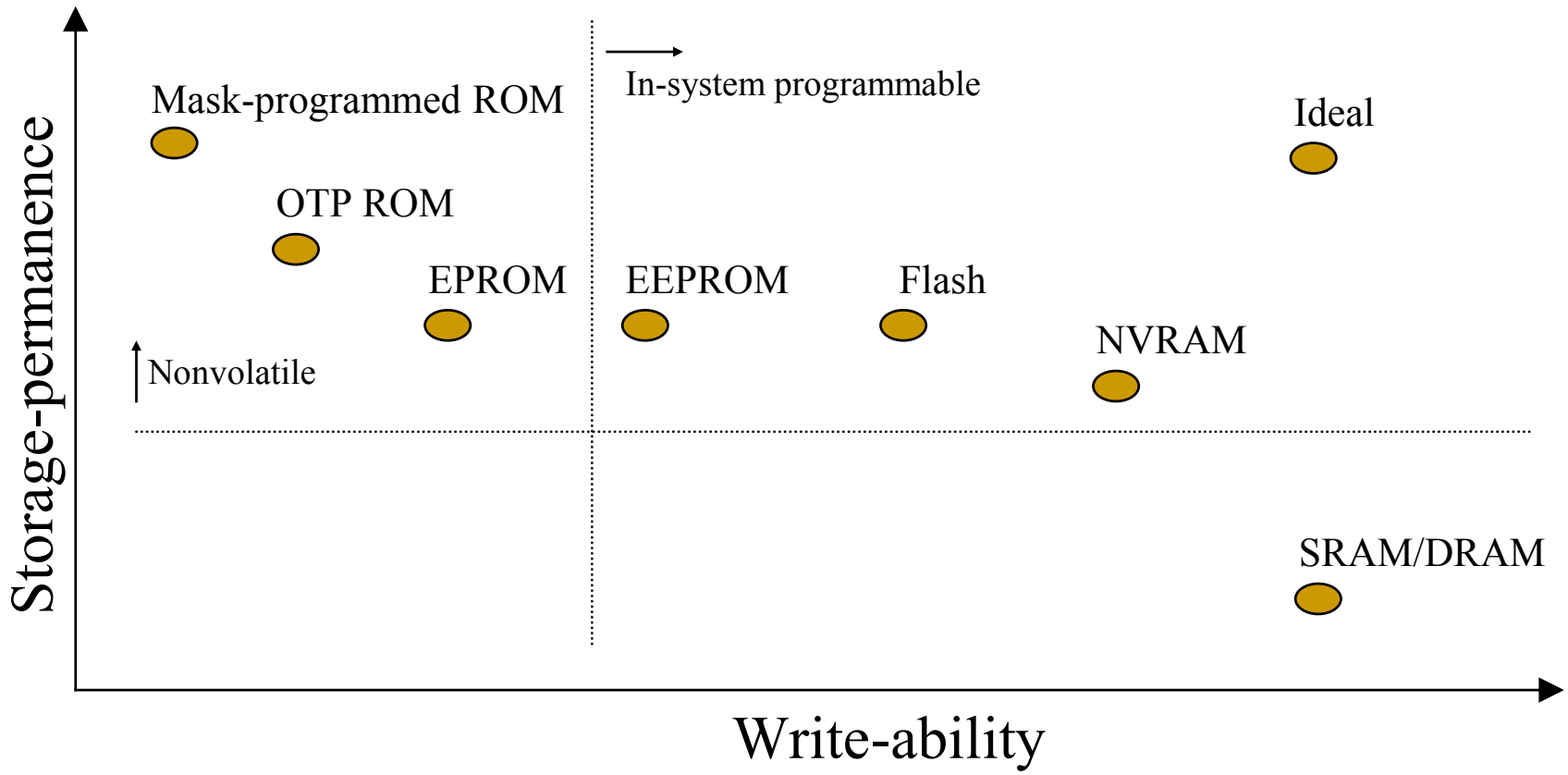
Ranges of storage permanence

- High end
  - Essentially never loses bits
  - E.g., mask-programmed ROM
- Middle range
  - Holds bits days/months/years after memory's power source turned off
  - E.g., NVRAM

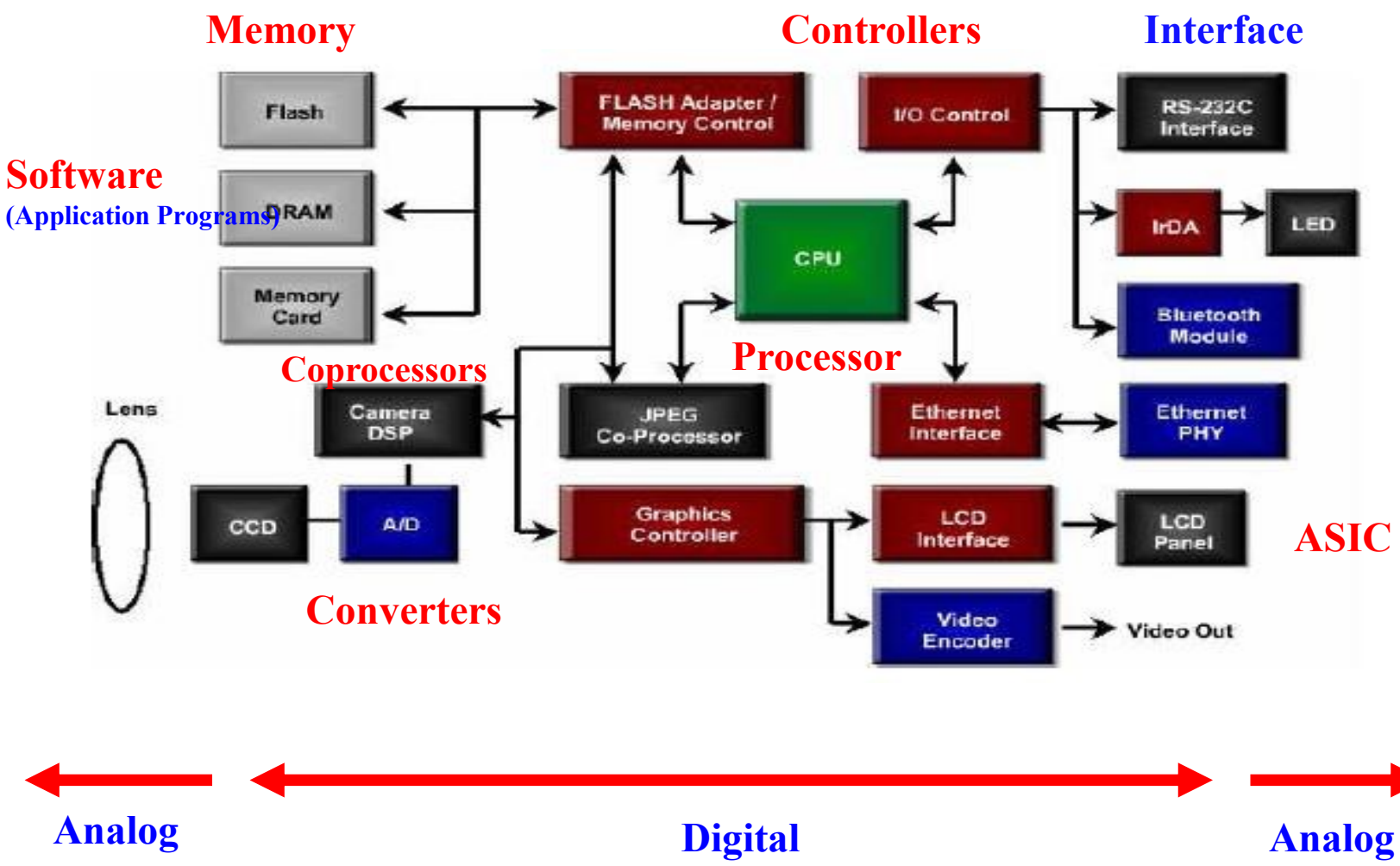
Storage-permanence

- Lower range
  - Holds bits as long as power supplied to memory
  - E.g., SRAM
- Low end
  - Begins to lose bits almost immediately after written
  - E.g., DRAM

Memory types



Digital camera hardware components: interface (up to here: session 2)



Communication

- What is a bus?
  - Artifact that transfers bits
  - Wires, air, or fiber and interface logic
- Associated with a bus, we have:
  - Connectivity scheme
    - Serial communication
    - Parallel communication
    - Wireless communication

Communication

- Protocol
  - Ports
  - Timing diagrams
  - Read and write cycles
- Arbitration scheme, error detection/correction, DMA, ...

Serial communication

- A single wire used for data transfer
- One or more additional wires used for control
  - But, some protocols may not use additional control wires
- Higher throughput for long distance communication
  - Often across processing node
- Lower cost in terms of wires (cable)
- E.g., USB, Ethernet, RS232, I2C, ...

Parallel communication

- Multiple buses (wires) used for data transfer
- One or more additional wires used for control
- Higher throughput for short distance communication
  - Data misalignment problem
  - Often used within a processing node
- Higher cost in terms of wires (cable)
- E.g., ISA, AMBA, PCI, ...

Wireless communication

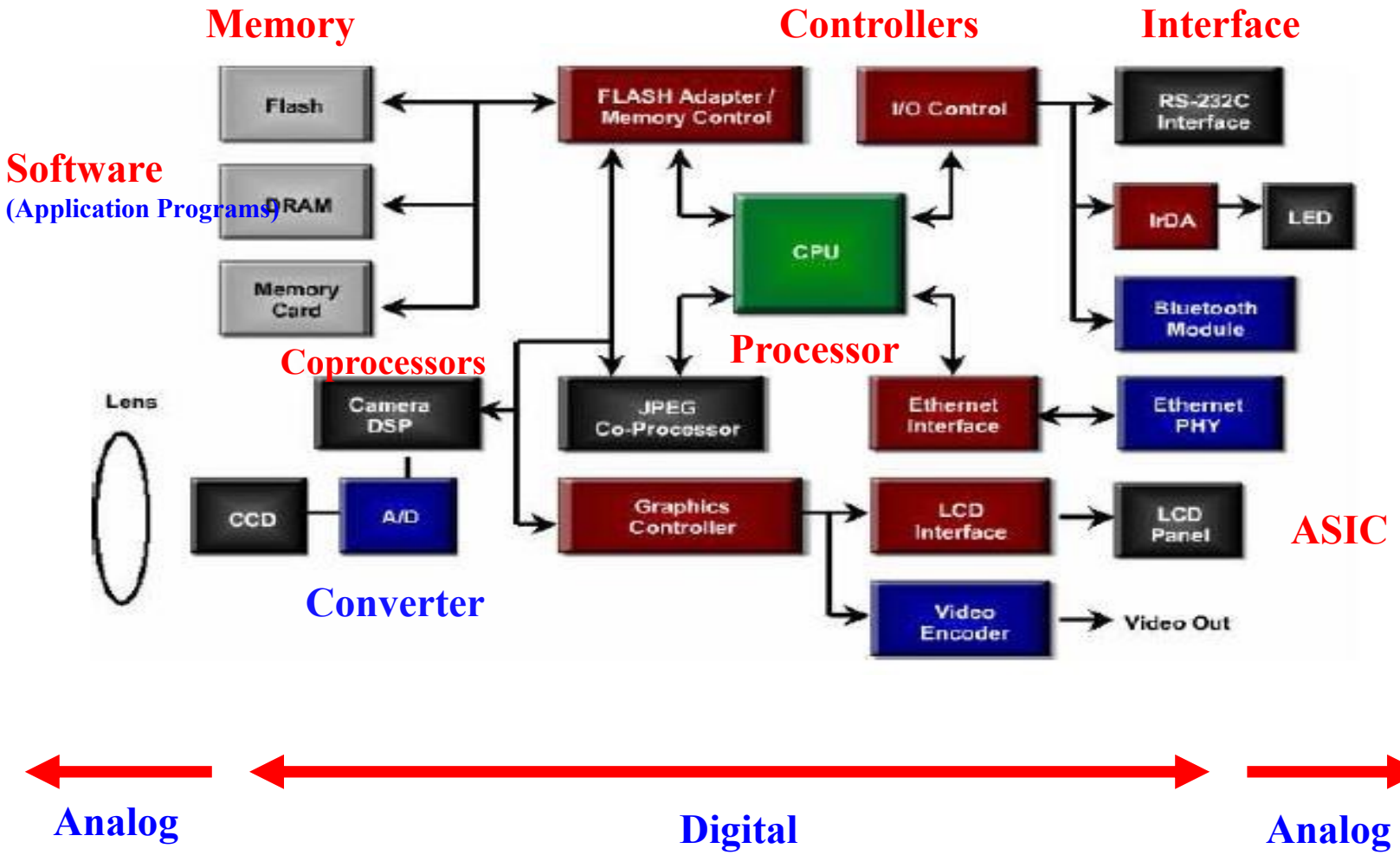
- InfraRed (IR)
  - Electromagnetic wave frequencies just below visible light spectrum
  - Diode emits infrared light to generate signal
  - Infrared transistor detects signal, conducts when exposed to infrared light
  - Cheap to build
  - Need line of sight, limited range

Wireless communication

- Radio Frequency (RF)
  - Electromagnetic wave frequencies in radio spectrum
  - Analog circuitry and antenna needed on both sides of transmission
  - Line of sight not needed, transmitter power determines range

Light comparison[9]			
Name	**Wavelength**	**Frequency (Hz)**	**Photon energy (eV)**
Gamma ray	less than 0.01 nm	more than 30 EHz	more than 124 keV
X-ray	0.01 nm – 10 nm	30 EHz – 30 PHz	124 keV – 124 eV
Ultraviolet	10 nm – 400 nm	30 PHz – 790 THz	124 eV – 3.3 eV
Visible	400 nm–700 nm	790 THz – 430 THz	3.3 eV – 1.7 eV
Infrared	700 nm – 1 mm	430 THz – 300 GHz	1.7 eV – 1.24 meV
Microwave	1 mm – 1 meter	300 GHz – 300 MHz	1.24 meV – 1.24 μeV
Radio	1 meter – 100,000 km	300 MHz – 3 Hz	1.24 μeV – 12.4 feV

Digital camera hardware components: converter (from here: session 4)

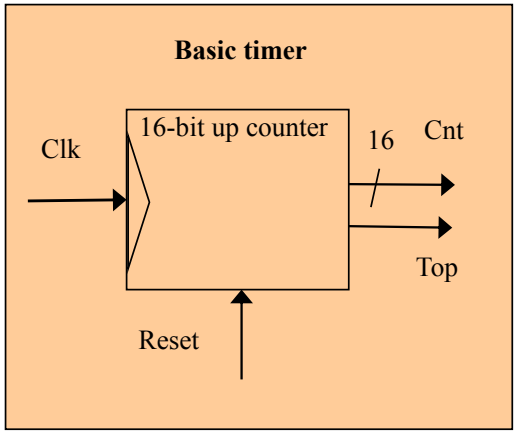


Peripherals

- Perform specific computation task
- Custom single-purpose processors
  - Designed by us for a unique task
- Standard single-purpose processors
  - "Off-the-shelf"
  - Pre-designed for a common task

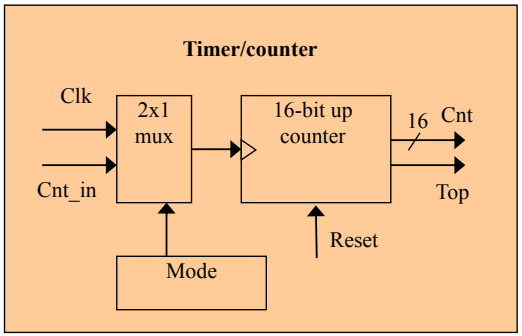
Timers

- Timers: measure time intervals
  - To generate timed output events
  - To measure input events
  - Top: max count reached
- Range and resolution



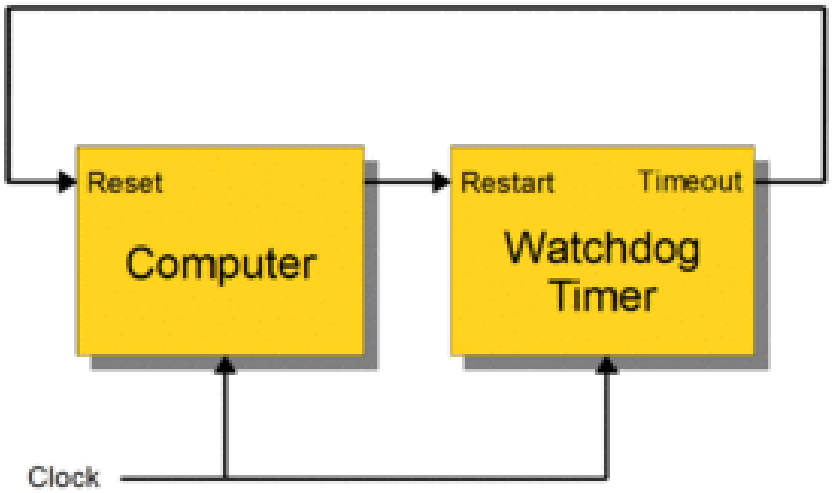
Counters

- Counter: like a timer, but counts pulses on a general input signal rather than clock
  - E.g., count cars passing over a sensor
  - Can often configure device as either a timer or counter



Watchdog timer

- Must reset timer every X time unit, else timer generates a signal
- Common use
  - Detect failure
  - Self-reset



UART

- UART: Universal Asynchronous Receiver Transmitter
  - Takes parallel data and transmits serially
  - Receives serial data and converts to parallel
- Parity: extra bit for simple error checking
- Start bit, stop bit
- Baud rate
  - Signal changes per second
  - Bit rate, sometimes different

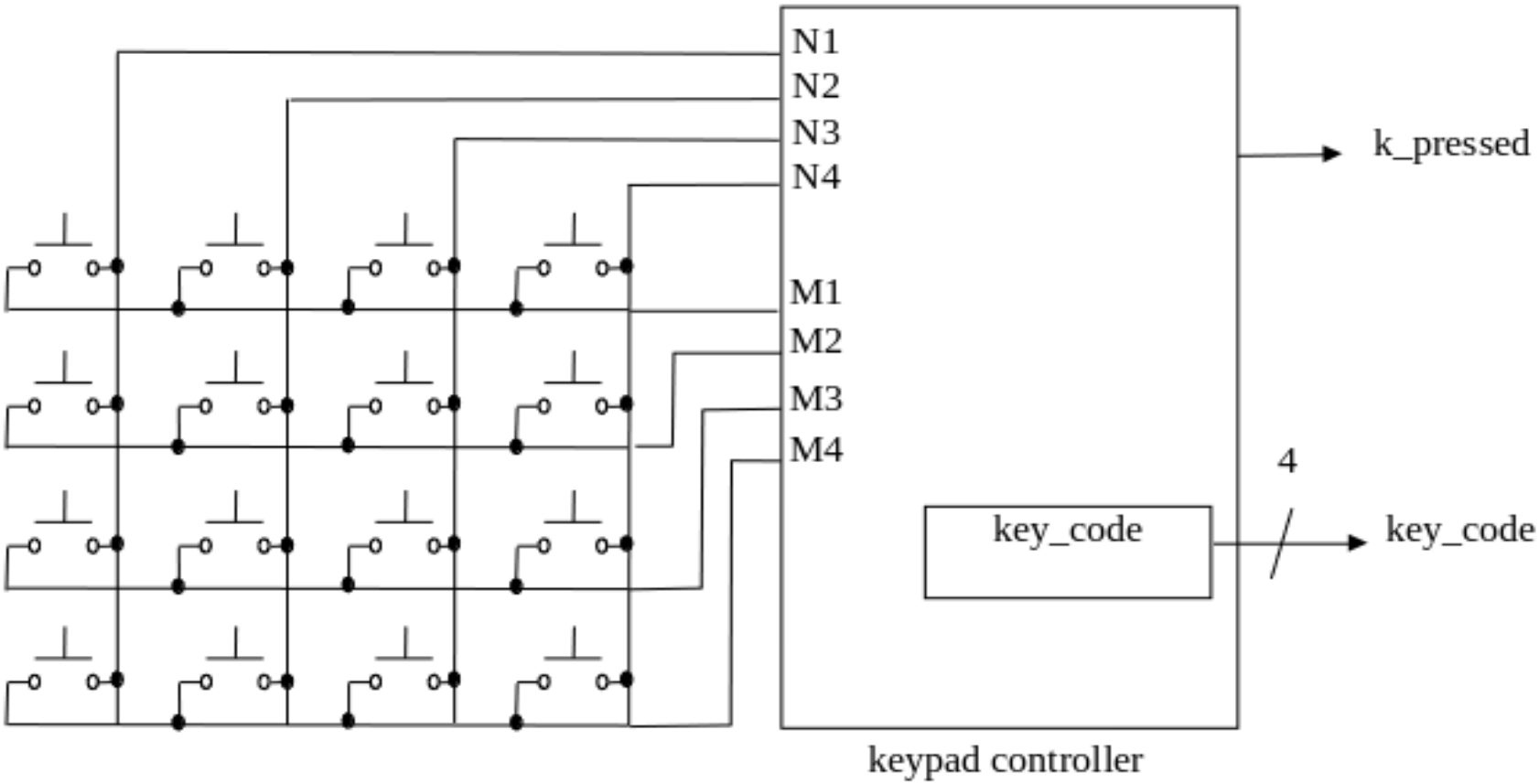
Pulse Width Modulator (PWM)

- Generates pulses with specific high/low times
- Duty cycle: % time high
  - Square wave: 50% duty cycle
- Common use: control average voltage to electric device
  - Simpler than DC-DC converter or digital-analog converter
  - DC motor speed, dimmer lights

LCD (up to here: session 3)

- Liquid Crystal Display
- N rows by M columns
- Controller build into the LCD module
- Simple microprocessor interface using ports
- Software controlled

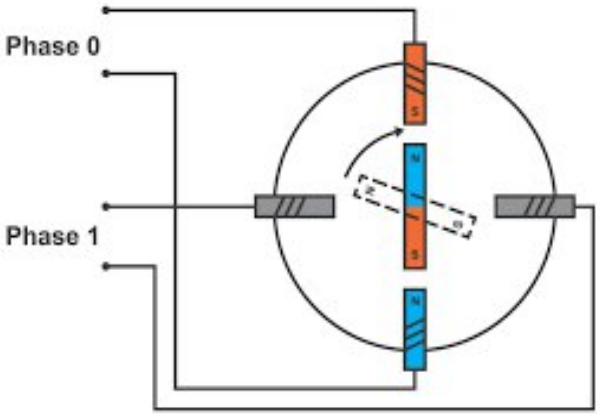
Keypad

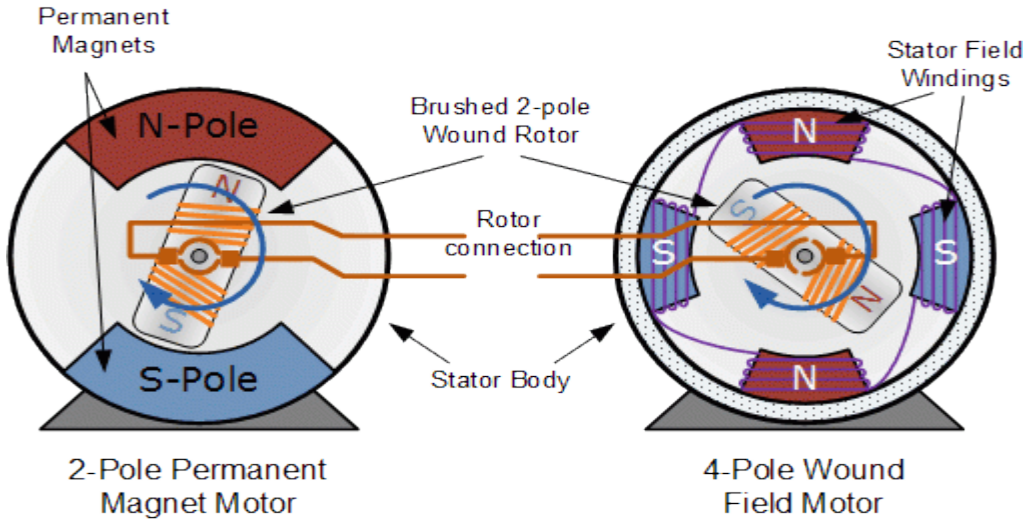

N1
N2
N3
N4
M1
M2
M3
M4
k_pressed
4
key_code
key_code
keypad controller

N=4, M=4

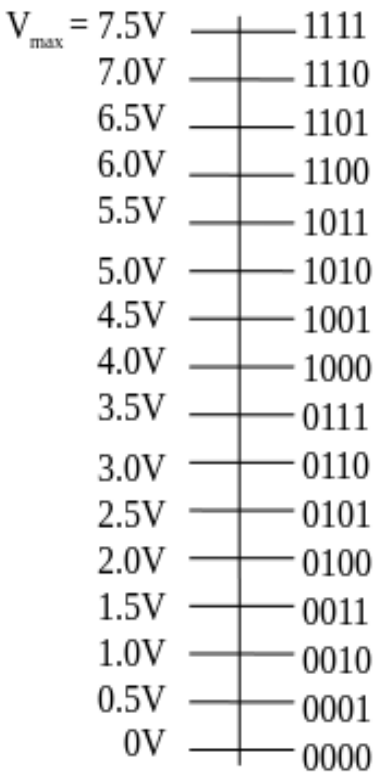
Stepper motor controller

- Stepper motor: rotates fixed number of degrees when given a "step" signal
  - In contrast, DC motor just rotates when power applied, coasts to stop
- Rotation achieved by applying specific voltage sequence to coils
- Controller greatly simplifies this

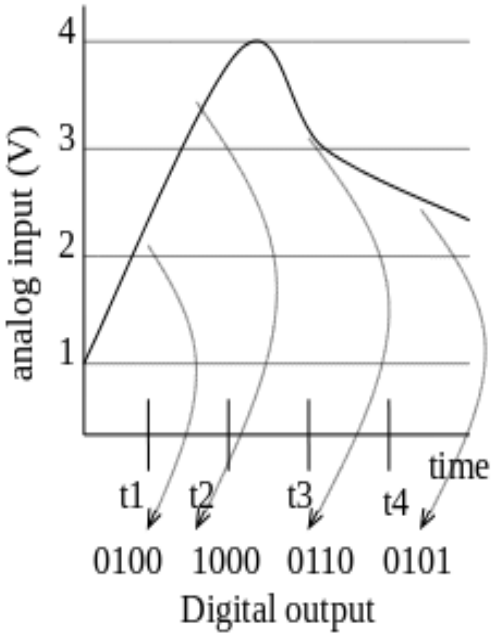




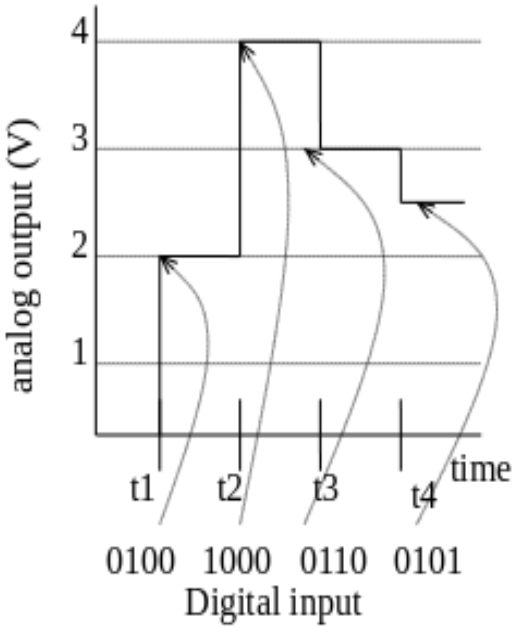
Analog-to-Digital Converter



Proportionality



Analog to Digital (A/D)



Digital to Analog (D/A)

Summary

- Hardware: Information processing
  - Processors
  - Memories
  - Communication
  - Peripherals

Importance of embedded systems

- 79% of all high-end processors are used in embedded systems
- They are part of almost everything that runs on electricity
- Crucial application in key industries
  - Automotive industry: 7% of EU's GNP

Characteristics of embedded systems

- Dependability
  - Reliability, Maintainability, Availability, Safety, Security
- Energy efficiency
- Performance
- Real-time constraints
  - For real-time systems, right answers arriving too late are wrong
- Weight efficient, cost efficient, code-size efficient
- Dedicated towards a certain application
  - Minimize resources, maximize robustness

Characteristics of embedded systems

- Dedicated user interface
  - No mouse, keyboard and screen
- Frequently connected to physical environment through sensors and actuators
- Hybrid systems (analog + digital parts)
- **Not every ES has all of the above characteristics**

A common misconception

- The study of embedded systems is simply a combination of some of the well-known areas such as:
  - Dependability
  - Real-time systems
  - Low power design
  - etc.

Interplay of design objectives

- Design objectives:
  - Fault tolerance (Dependability)
  - Energy efficiency
  - Real-time
  - Cost efficient
- The design objectives are at odds
  - Example: Fault tolerance requires some types of redundancy and redundancy leads to energy consumption.

Challenges for embedded systems

- Although the study of embedded systems is an interdisciplinary area of study, it has its own challenges:
  - Interplay of different design objectives
  - Special features of embedded systems
    - Weight efficient, cost efficient, code-size efficient, diskless systems
  - Challenges in system specification, design, and verification
    - We need to use special model, languages and tools

Reactive systems

- Typically ES are reactive systems

"A reactive system is in continual interaction with its environment and executes at a pace determined by that environment."

Reactive systems

- Reactive systems = Event-based systems
- The traditional paradigms of programming (i.e. model of computable functions) are inappropriate
  - Model of computable functions
    - Von Neumann paradigm
    - Sequential computing
- Suitable model for reactive systems:
  - Automata-based programming paradigm

Automata-based programming

- Automata-Based Programming is a programming paradigm whose defining characteristic is the use of finite state machines to describe program behavior
- The transition graphs of a state machines are used in all stages of software development
  - Specification
  - Implementation
  - Debugging
  - Documentation

Will be covered in chapter 3

What is a real-time system?

- Real-time system means that the system is subjected to real-time
  - Response should be guaranteed within a specified timing constraint
  - System should meet the specified deadline
- Types of real-time systems
  - Hard real-time system
    - Missing the deadline may have disastrous consequences
  - Soft real-time system
    - Missing the deadline have no disastrous consequences
    - Miss its deadline occasionally with some acceptably low probability

Is embedded and real-time synonymous?

- Most embedded systems are real-time
  - Assumption: Embedded non real-time system
    - Watering plants system, toys, video games, thermostat, e-reader and etc.
- Most real-time systems are embedded
  - Assumption: Real-time non-embedded system
    - Webkiosk and etc.

The End

(up to here: session 4)