ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN KỸ THUẬT ĐIỆN TỬ

Advanced Embedded System Design

Chapter 1: Embedded System Design Process

1. Embedded system features and issues
2. Embedded system design process
3. Embedded system analysis

1. Embedded system features and issues

- **Embedded system definition**
  - any device that includes a programmable computer but is not itself a general-purpose computer - **Wayne Wolf**
  - An embedded computer system includes a micro-computer with mechanical, chemical and electrical devices attached to it, programmed for a specific dedicated purpose, and packaged as a complete system - **Jonathan W. Valvano**
  - An embedded system is one that has computer-hardware with software embedded in it as one of its most important component - **Raj Kamal**




1.1. Embedded System Features

Common Features (Non-technical specification)

Cost

Applications

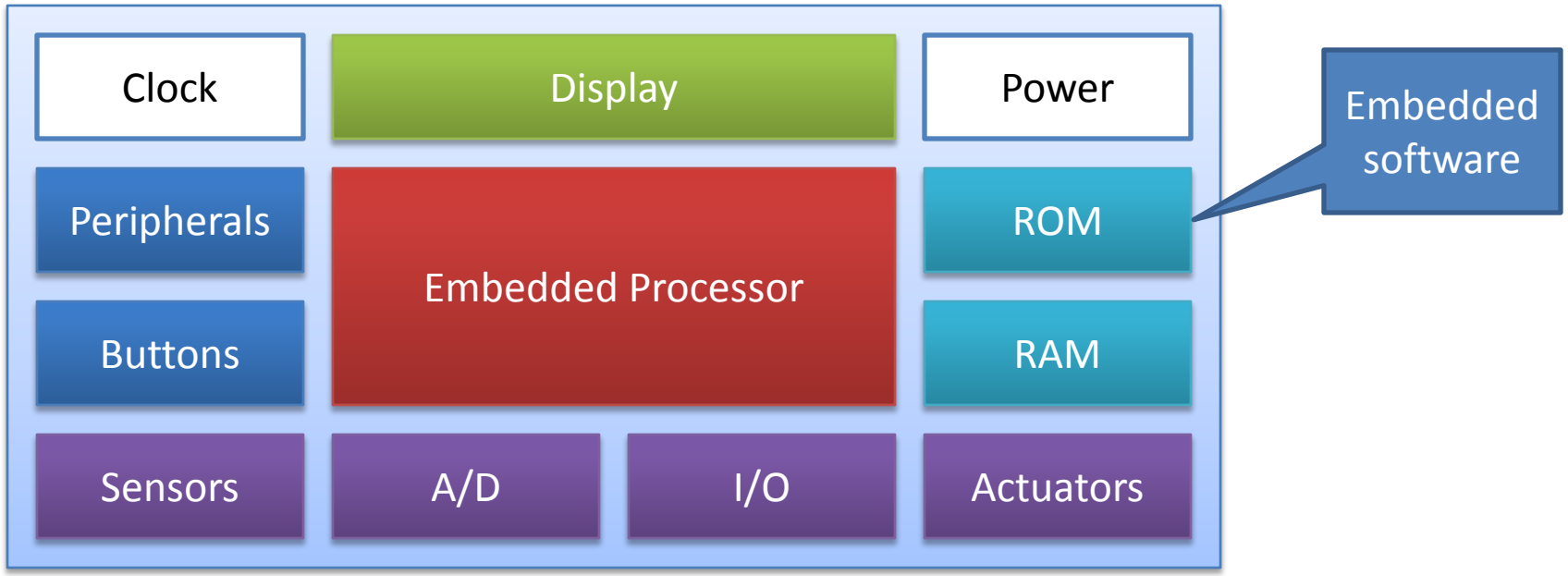
Speed

Size/Weight

Power/Energy

1.1. Embedded System Features

❑ HARDWARE FEATURES



- **Embedded processor** is a heart of the embedded system:
  - micro-processor: 8086, ARM7/9/11, PowerPC, Nios
  - micro-controller: 8051, ARM Cortex-M, PIC16F, 68HC11, LM4F120 , STM32F4

1.1. Embedded System Features

- **Memory**
  - ROM: embedded program
  - RAM: processing data
- **Peripheral interface**
  - GPIO, UART, I2C, SPI
  - ADC, PWM
  - USB, MicroSD
  - Ethernet, wifi
- **User interface**
  - Button / keypad / switch
  - LCD / 7-segment LED / LED indicator
- **Sensors**
  - temperature, humidity, light, ultrasound, pressure
  - Motion sensor, gyroscope
- **Actuators**
  - motor, solenoid, relay, …
- **Clock**
  - crystal
- **Power**
  - +5V / +3.3V / +2.5V
  - Battery, DC adapter, …

1.1. Embedded System Features

❑ SOFTWARE FEATURES

- **Operating system**
  - Non-OS
  - RTOS, LynxOS, RTLinux
  - Linux, Android, WinCE
- **Embedded programs**
  - Supported functions
  - User applications
  - Software update
- **User Interface**
  - Command line
  - Text display
  - Graphic user interface

1.2. Embedded System Issues

❑ Design Issues

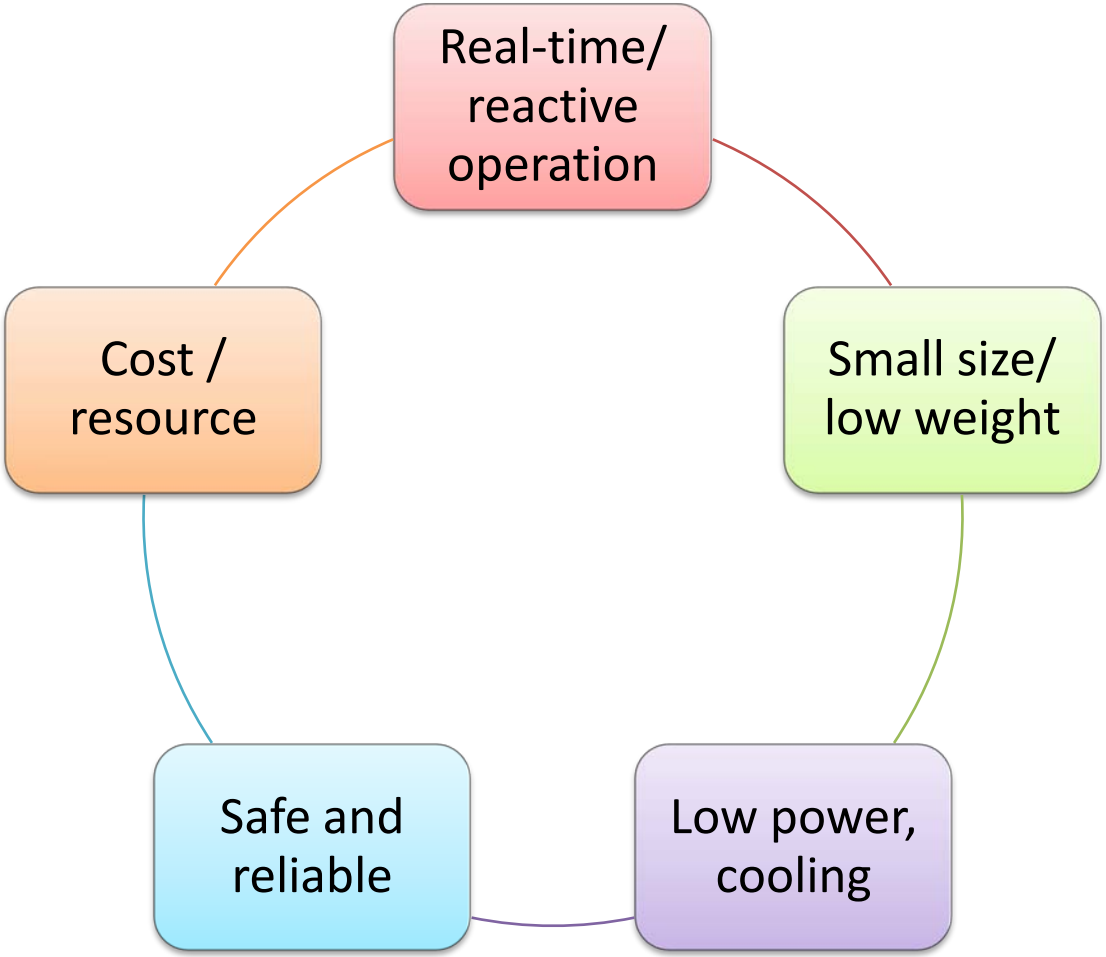
1. Constraints
   - cost may matter more than speed
   - long life cycle
   - Reliability/safety
   - Low-power
2. Functions
   - safety-critical applications
   - damage to life, economy can result
3. Real-time
   - Not only right output but at the right time
   - processing time is the most important one.

1.2. Embedded System Issues

4. Concurrency
   - System and environment run concurrently
   - multi-functional
   - interface with other systems
5. Reactive systems
   - Once started run forever
   - Continuous interaction with their environment
   - termination is a bad behavior

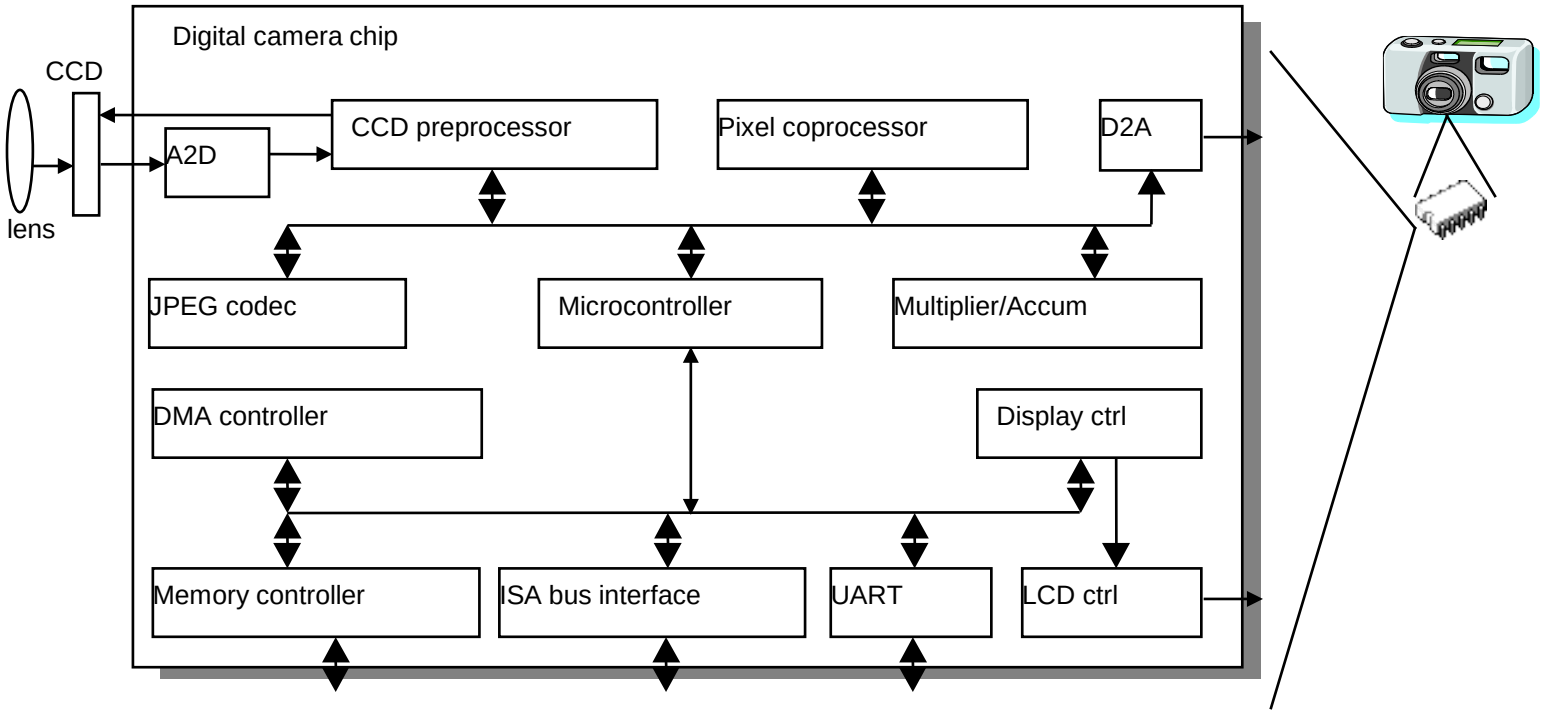
1.2. Embedded System Issues

- Basic requirements for an embedded system



An embedded system example - 1

A digital camera



- **Constraint:** Low cost, low power, small, fast
- **Function:** capture and store pictures / videos
- **Real-time:** only to a small extent
- **Concurrent:** single function
- **Reactive:** power on demand

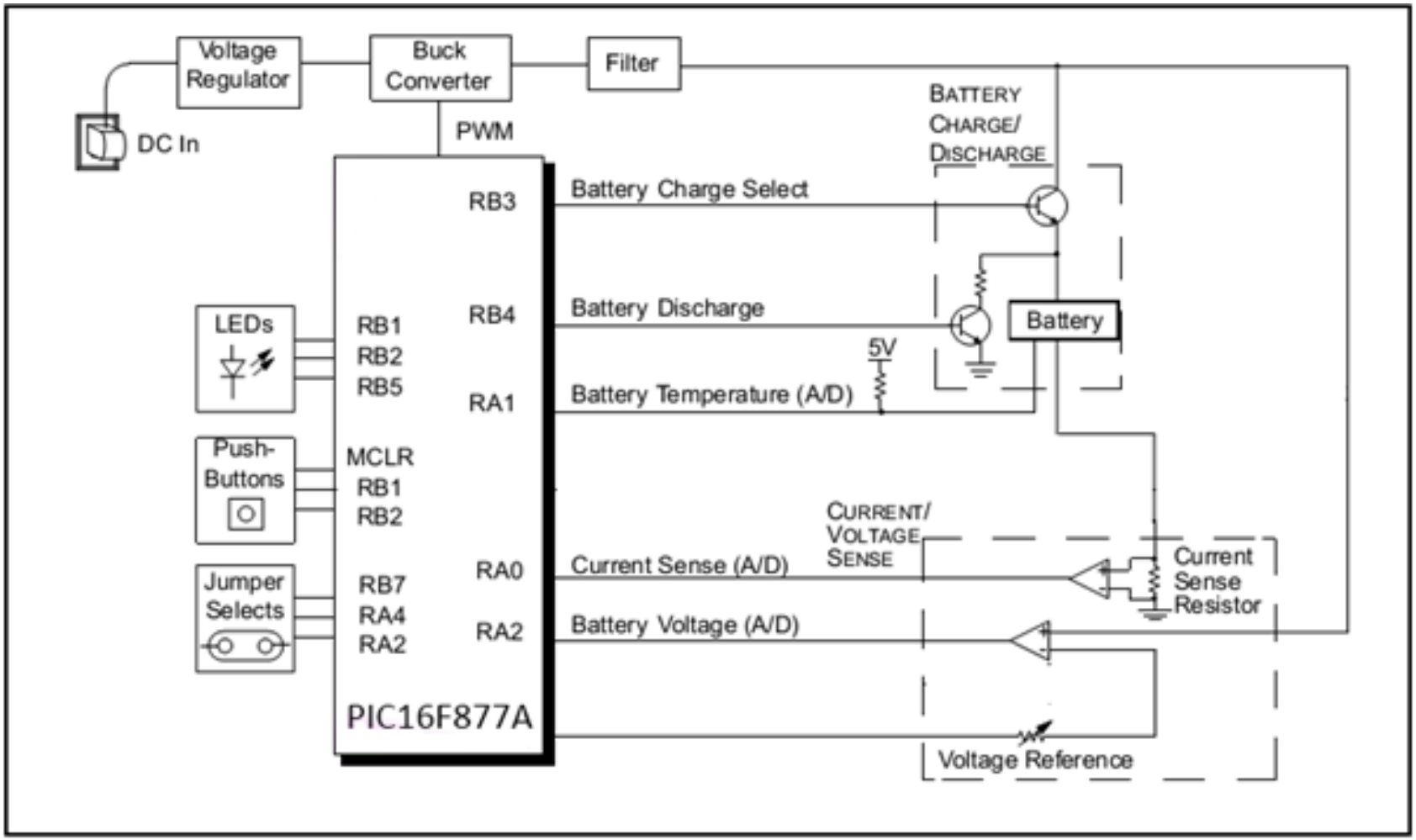
An embedded system example - 2

Battery charger

- Common features
  - **Cost**: Low Cost
  - **Applications/Functions**
    - Ability to charge NiCd, NiMH, Li-Ion batteries
    - User Selectable Charging Algorithms
    - High Charge Current Capability
  - **Speed:** Fast Charge Rate
  - **Size/weight**: Small size, portable
  - **Power/energy**: 12V DC adapter

An embedded system example - 2

- Battery Charger Design



An embedded system example - 2

Battery Charger

- Hardware features
  - **Processor:** PIC16F877A
  - **Memory:** 8k Flash ROM
  - **Peripheral interface:** GPIO, ADC, RS232
  - **User interface:** text LCD, LED, button, Jumper
  - **Sensor:** current sensor, voltage sensor
  - **Actuator:** 5A BJT
  - **Clock:** 16MHz
  - **Power:** 5VDC

An embedded system example - 2

Battery Charger

- Software features
  - **Operating system:** Non-OS
  - **Embedded programs**
    - Charging controlling
    - Charging current, voltage displaying
  - **User Interface**
    - Text LCD: displaying mode, current, voltage
    - LED: indicating status
    - Button: selecting charging mode
    - Jumper: selecting battery type

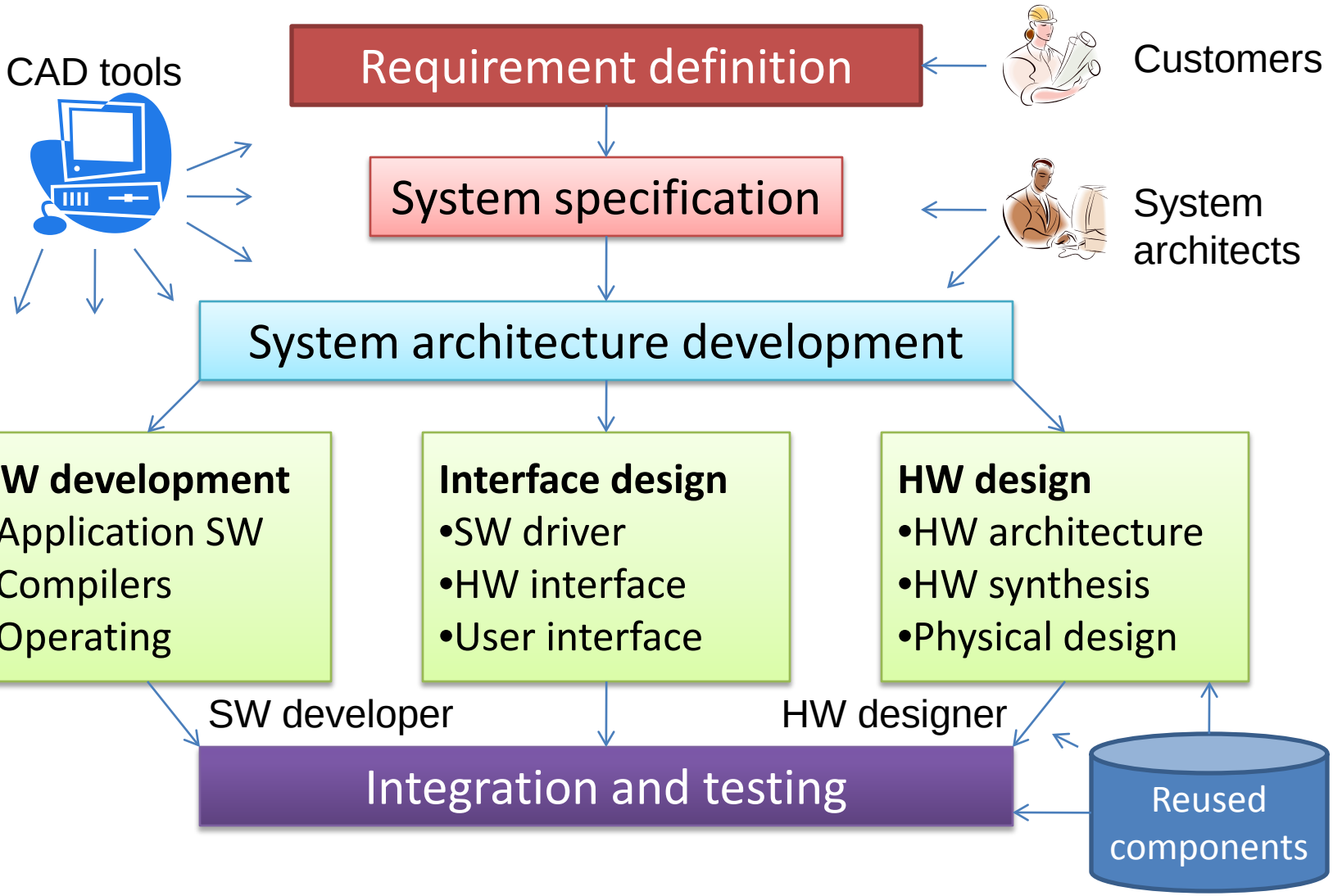
An embedded system example - 2

- Design Issues
  - **Constraints**
    - High charging current: 3A
    - Safety: fuse protection, insulating box
  - **Functions**
    - Charging process: bulk charge, absorption charge, float charge
  - **Real-time system**
    - Non-critical real-time system
    - Charging process based on predefined timer
  - **Concurrent system**
    - Charging control and current/voltage display
  - **Reactive system**
    - Non-continuous interaction

Quiz

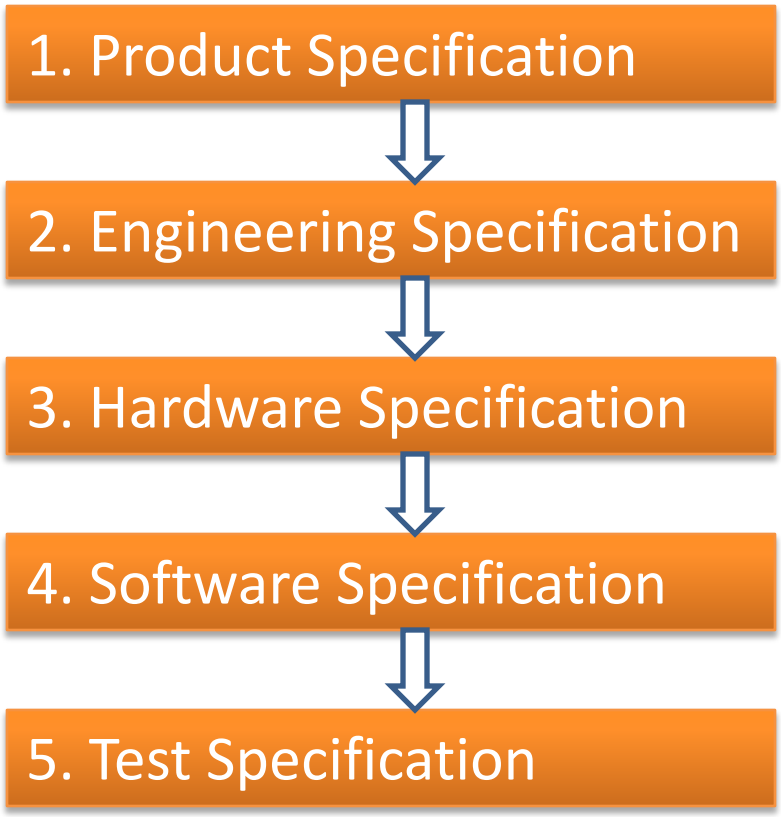
1. What is an embedded system?
2. What are embedded system features?
3. What are issues of embedded system design?
4. What are five basic requirements of embedded system?
5. What are common characteristics of an embedded system?
6. What are optimizing design metrics?
7. What is the most importance step of embedded system design process?
8. What are three key technologies for embedded systems?
9. Design features for the following design:
    1. Digital door lock
    2. Digital clock
    3. 3D LED cube

2. Embedded System Design Process



2.1. System Specification

Documents for System Specification



Describe how the product will be.

Describe what board, subsystem, and firmware will be used.

Describe how the board will be implemented and how it works.

Describe how the software will be implemented.

Describe how the system will be tested.

2.1. System Specification

1. Product Specification:

- What the system is to do
- What the user interface is
- What the real world I/O consists of
- What the external interface to other system is (if any)
- What are the constraints? (speed, stability, low power, cost)

- Example: Oven temperature control system
  - Functions: control the heater and the fan of an oven
  - I/O: temperature sensor, heater port, fan port
  - User interface: LCD display, keypad
  - External interface: UART

2.1. System Specification

2. Engineering Specification:

- What kind of hardware will be used
- What are the requirements for hardware and software

- Example: Oven temperature control system
  - 8051 microcontroller, LM35 sensor, LCD 16x2-B, ADC0809, RS232 IC
  - Requirements
    - PID control algorithm
    - real-time processing
    - Display current temperature value
    - able to set operating temperature value
    - Transfer data to computer through UART

2.1. System Specification

3. Hardware Specification:

- The requirements from engineering documents
- How the hardware implements the functionality
- The software interfaces to the hardware

- Example: Oven temperature control system
  - PIC Microcontroller 12MHz, sensor LM35, LCD 16x2-B, Keypad 16, ADC0809, FET IRF260 for heater/fan control
  - Microcontroller reads temperature value from LM35 through ADC, display this value to LCD, and then control heater and fan based on PID control algorithm

2.1. System Specification

4. Software Specification:

- The requirements from engineering specification
- Interface to other software
- How the software implements the requirements

- Example: Oven temperature control system
  - design a function:
    - void LCD_display(String str)
    - int PID_control(int temp, int data[])

2.1. Embedded System Design Process

5. Test Specification:

- The device, equipment, environment for testing
- Prototype
- Testing process

- Example: Oven temperature control system
  - Voltage meter, temperature meter
  - Prototype: bread board
  - Testing process:
    - calibrate temperature sensor
    - check LCD, keypad
    - check output port (heater, fan)
    - verify PID control algorithm

2.2. System architecture development

- Partitioning the system into three parts:
  - Interface design
  - Hardware design
  - Software development
- Methodology
  - Block diagram
  - Waveform
  - Function description
  - Coding guideline

2.2. System architecture development

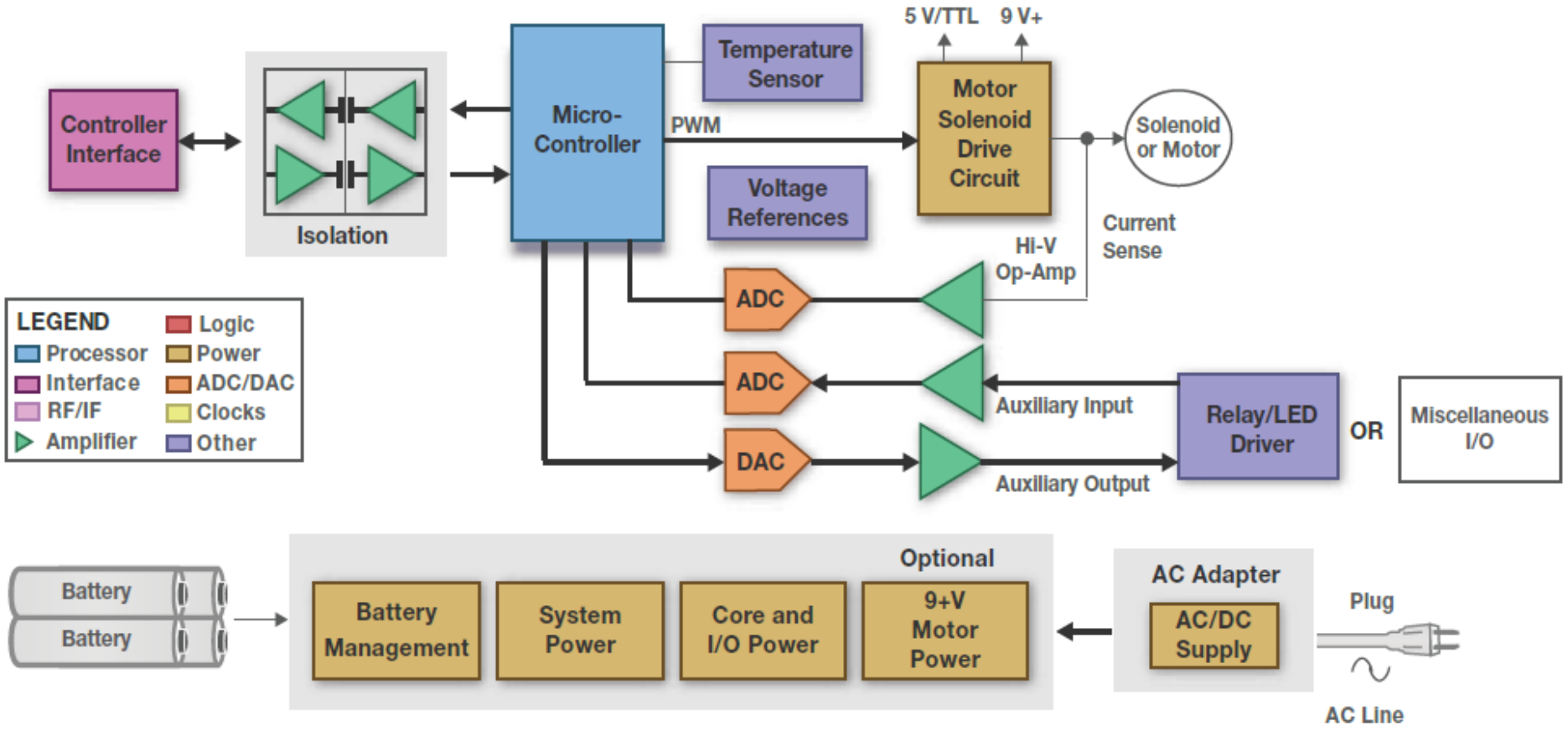
- Hardware block diagram
  - Use a rectangle for a hardware block

  - Use an arrow for a connection

  - Use a symbol for a special block

2.2. System architecture development

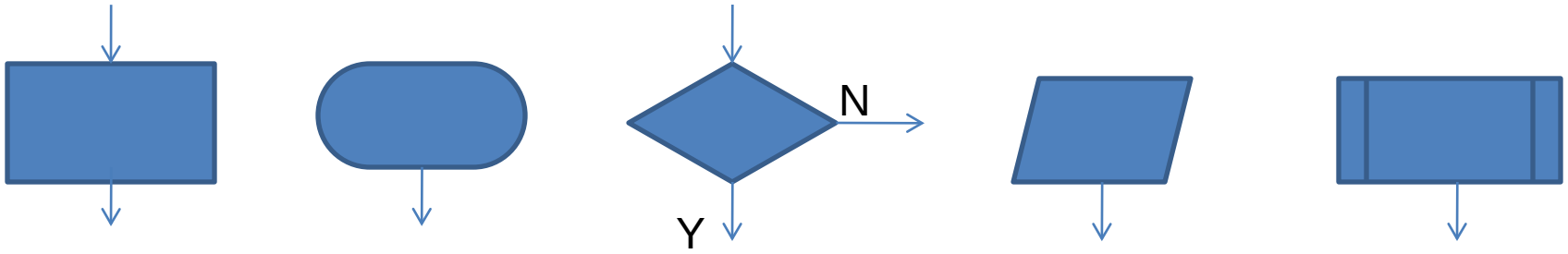
- Hardware block diagram - Example



MOTOR CONTROL

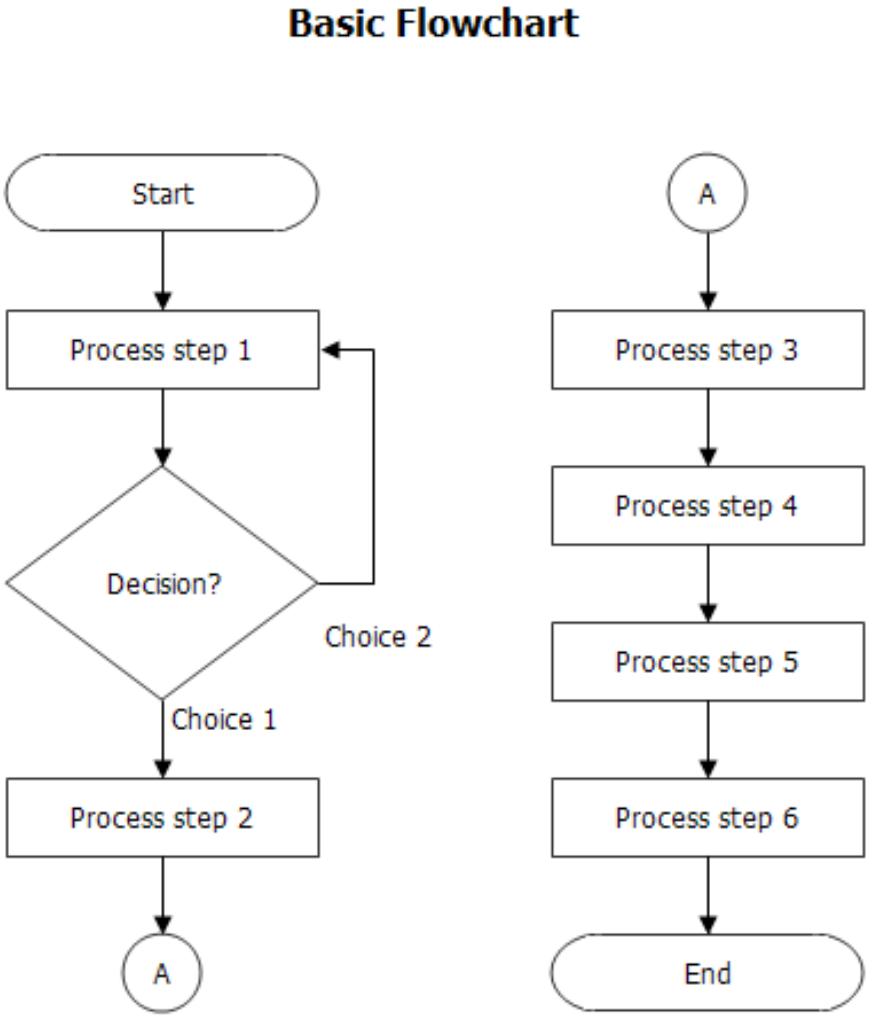
2.2. System architecture development

- Software diagram: flowchart, state diagram
  - Use a rectangle for a **process**
  - Use a rounded rectangle for a **terminator**
  - Use a diamond shape for a **decision**
  - Use a parallelogram for **data**
  - Use a rectangle with two vertical lines for **predefine process**



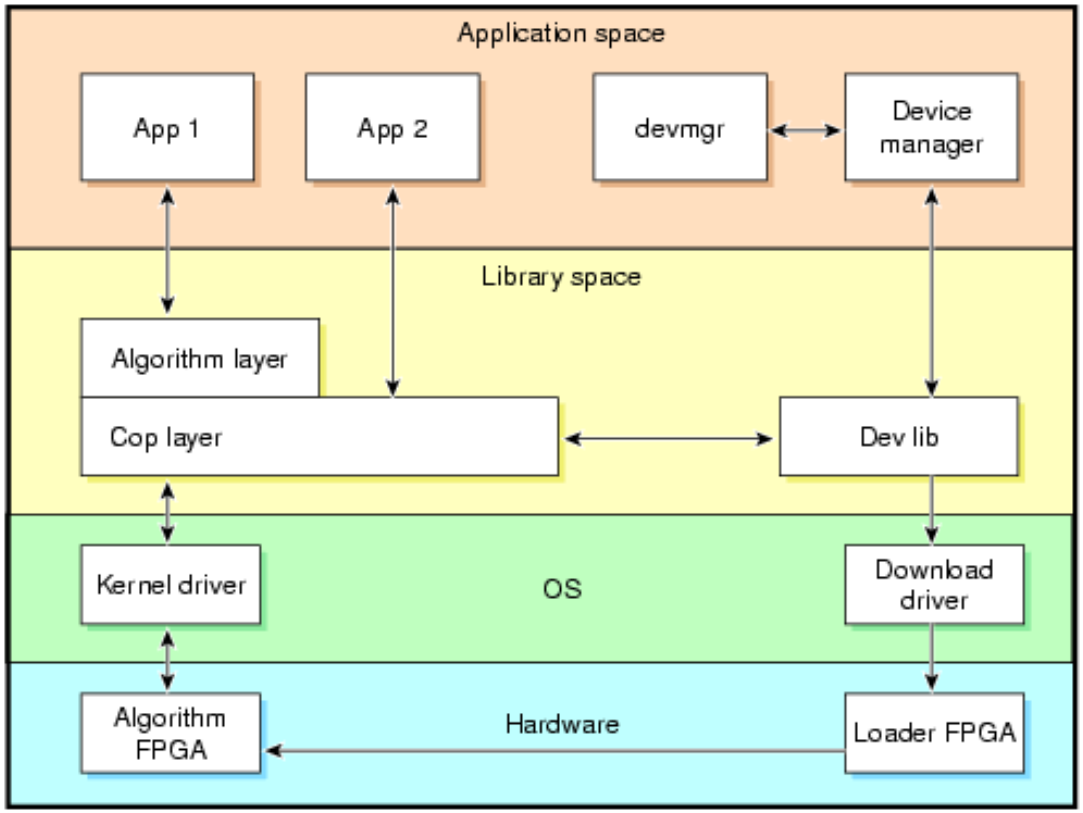
2.2. System architecture development

- Software block diagram - Example



2.2. System architecture development

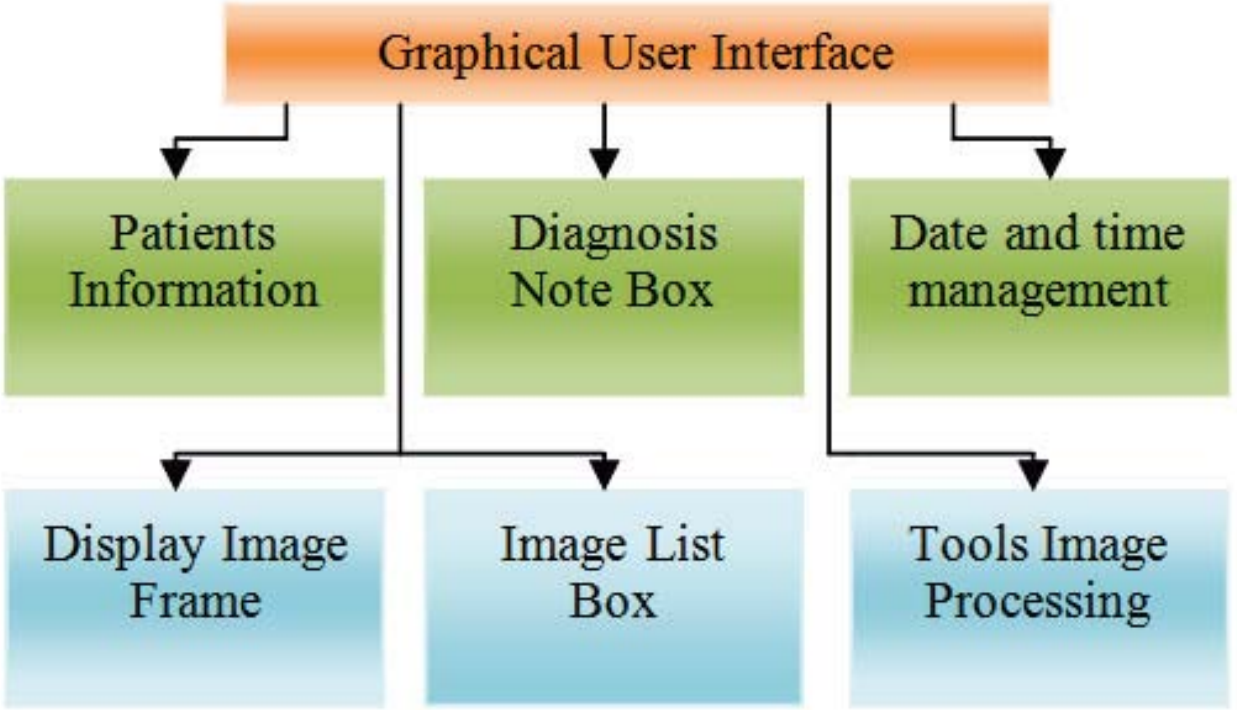
- Interface design
  - Show the connections between hardware and software
  - Show the connections with other systems
  - Show the interface with users



Examples

2.2. System architecture development

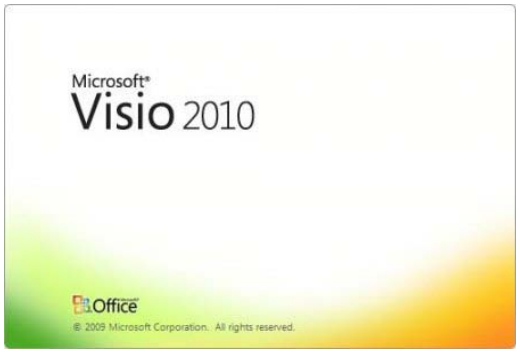
- Interface design – example :

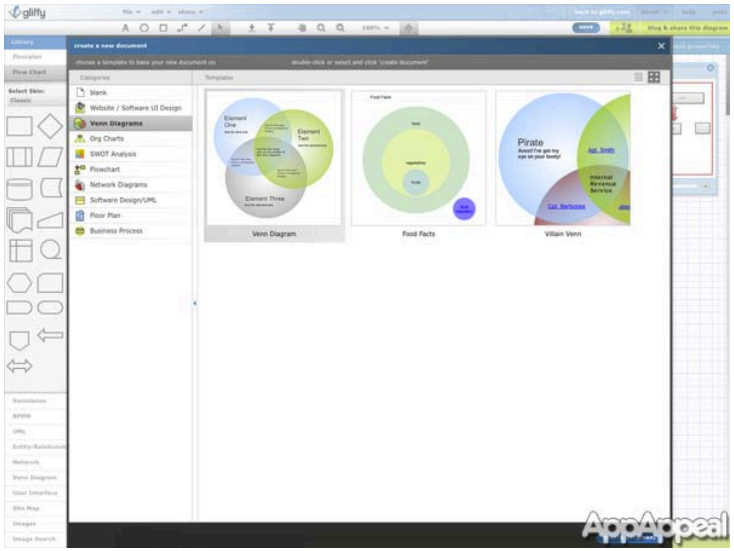


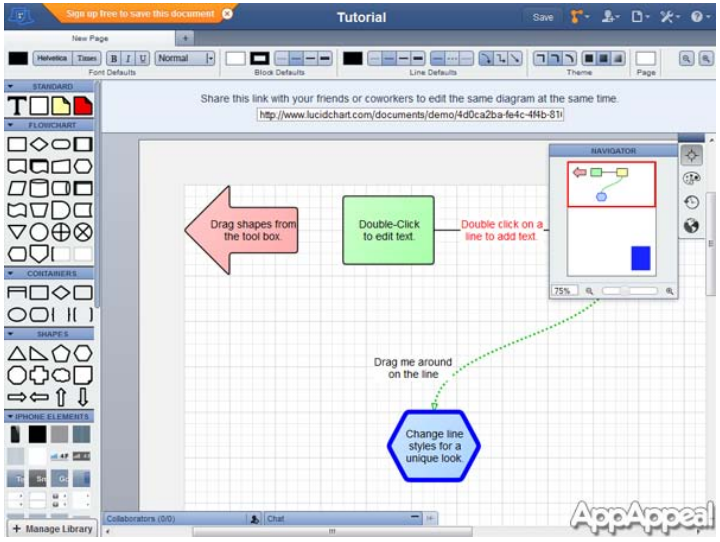
Interface design for a dental camera system

2.2. System architecture development

- Diagram software tools
  - Microsoft Visio
  - Gliffy
  - Lucid chart







2.3. Integration and Testing

- Integration process
  - Configuration
    - Define configurations, parameters, versions
  - Hardware integration
    - integrate microprocessor, peripherals, sensors, actuators, user interfaces
  - Software integration
    - integrate OS, drivers, functions,
  - System integration

2.3. Integration and Test

- Testing process
  - Hardware testing
    - CPU
    - Peripheral interface
    - Sensor
    - Actuator
    - User interface
  - Software testing
    - Operating system, drivers
    - Applications/functions
    - User interface
  - Hard/soft co-testing
    - Test cases

3. Embedded System Analysis

1. Technology selection
2. System optimization
3. Cost analysis

3.1. Technology Selection

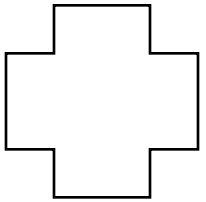
- Technology
  - A manner of accomplishing a task, especially using technical processes, methods, or knowledge
- Three key technologies for embedded systems
  - **Processor technology**: general-purpose, application-specific, single-purpose
  - **IC technology:** Full-custom, semi-custom, PLD
  - **Design technology:** Compilation/synthesis, libraries/IP, test/verification

3.1. Technology Selection

- Processor selection
  - number of IO pins required
  - interface required
  - memory requirements
  - number of interrupts required
  - real-time considerations
  - development environment
  - processing speed required

General-purpose processor

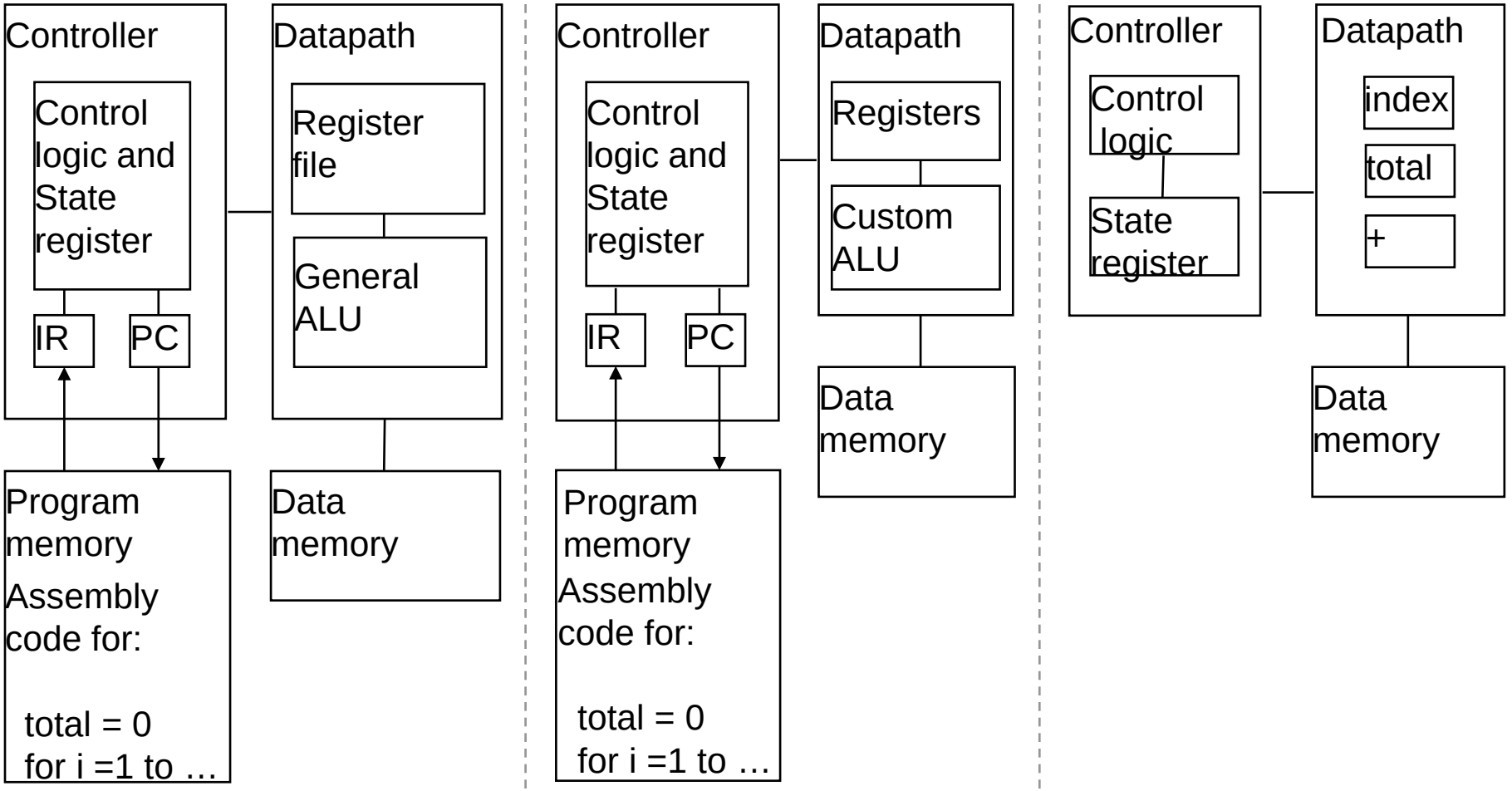
Application-specific processor



Single-purpose processor

Processor technology

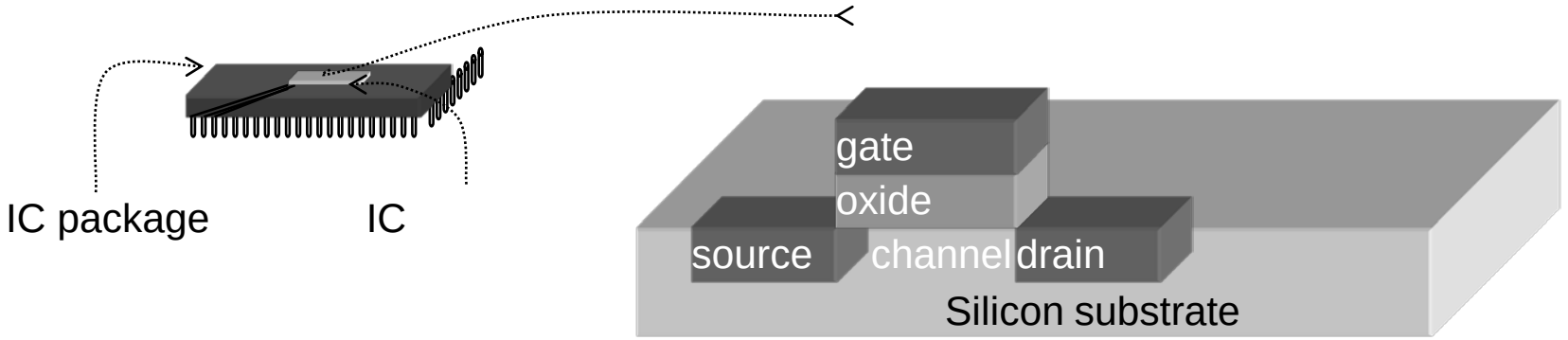
- "Processor" *not* equal to general-purpose processor



General-purpose ("software") **Application-specific** **Single-purpose** ("hardware")

IC technology

- The manner in which a digital (gate-level) implementation is mapped onto an IC
  - IC: Integrated circuit, or “chip”
  - IC technologies differ in their customization to a design
  - IC’s consist of numerous layers (perhaps 10 or more)
    - IC technologies differ with respect to who builds each layer and when



3.2. System Optimization

- Hardware / software partitioning
  - which functions should be performed in hardware, and which in software?
  - the more functions in software, the lower will be the product cost

3.2. System Optimization

Design challenge – optimizing design metrics

- Common metrics
  1. Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  2. NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
  3. Size: the physical space required by the system
  4. Performance: the execution time or throughput of the system
  5. Power: the amount of power consumed by the system
  6. Flexibility: the ability to change the functionality of the system without incurring heavy NRE cost

3.2. System Optimization

- Common metrics (continued)
  7. Time-to-prototype: the time needed to build a working version of the system
  8. Time-to-market: the time required to develop a system to the point that it can be released and sold to customers
  9. Maintainability: the ability to modify the system after its initial release
  10. Correctness: check the functionality throughout the process of designing the system; insert test circuitry to check that manufacturing was correct
  11. Safety: the probability that the system will not cause harm

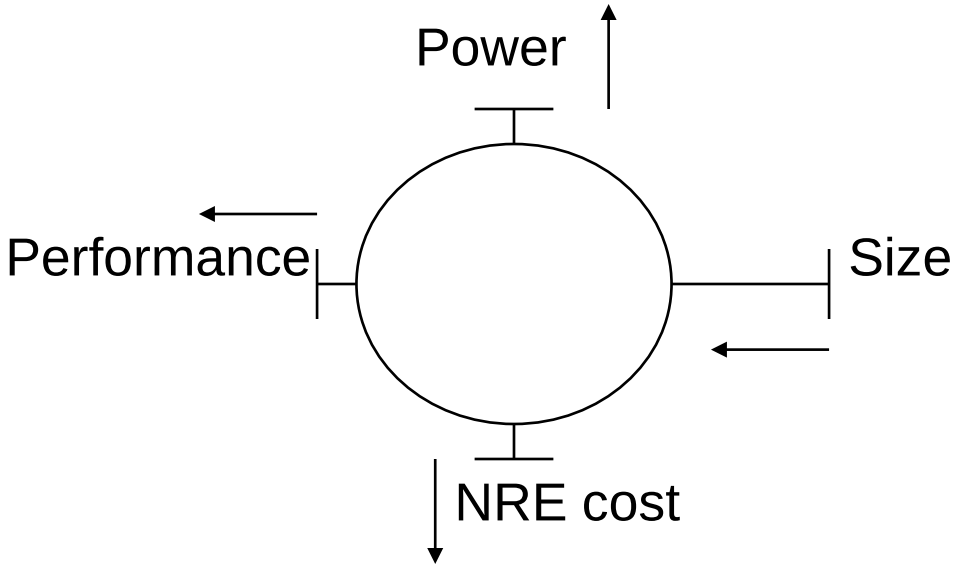
The performance design metric

- Widely-used measure of system, widely-abused
  - Clock frequency, instructions per second – not good measures
  - Digital camera example – a user cares about how fast it processes images, not clock speed or instructions per second
- Latency (response time)
  - Time between task start and end
  - e.g., Camera's A and B process images in 0.25 seconds
- Throughput
  - Tasks per second, e.g. Camera A processes 4 images per second
  - Throughput can be more than latency seems to imply due to concurrency, e.g. Camera B may process 8 images per second (by capturing a new image while previous image is being stored).
- *Speedup* of B over S = B's performance / A's performance
  - Throughput speedup = 8/4 = 2

3.2. System Optimization

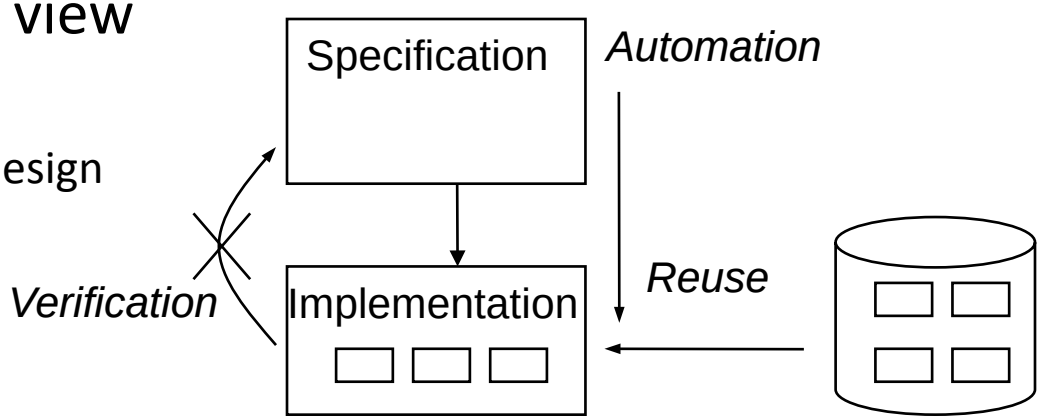
Design metric competition -- improving one may worsen others

- Expertise with both **software and hardware** is needed to optimize design metrics
  - Not just a hardware or software expert, as is common
  - A designer must be comfortable with various technologies in order to choose the best for a given application and constraints



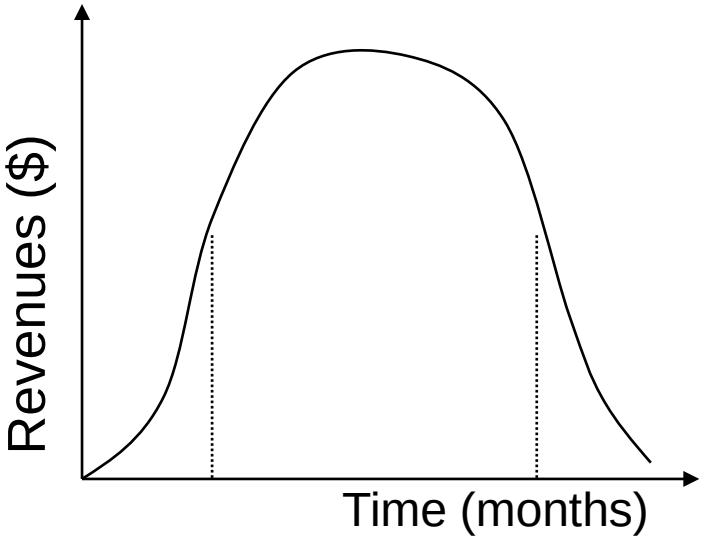
Improving productivity

- Design technologies developed to improve productivity
- We focus on technologies advancing hardware/software unified view
  - **Automation**
    - Program replaces manual design
    - Synthesis
  - **Reuse**
    - Predesigned components
    - Cores
    - General-purpose and single-purpose processors on single IC
  - **Verification**
    - Ensuring correctness/completeness of each design step
    - Hardware/software co-simulation



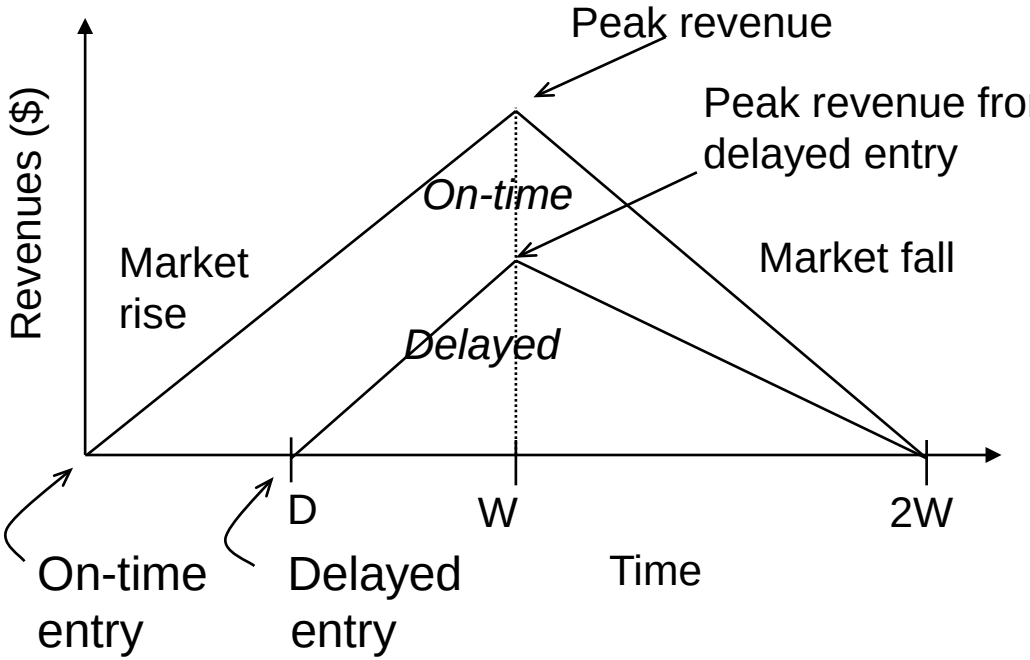
3.3. Cost Analysis

- Time required to develop a product to the point it can be sold to customers
- Market window
  - Period during which the product would have highest sales
- Average time-to-market constraint is about 8 months
- Delays can be costly



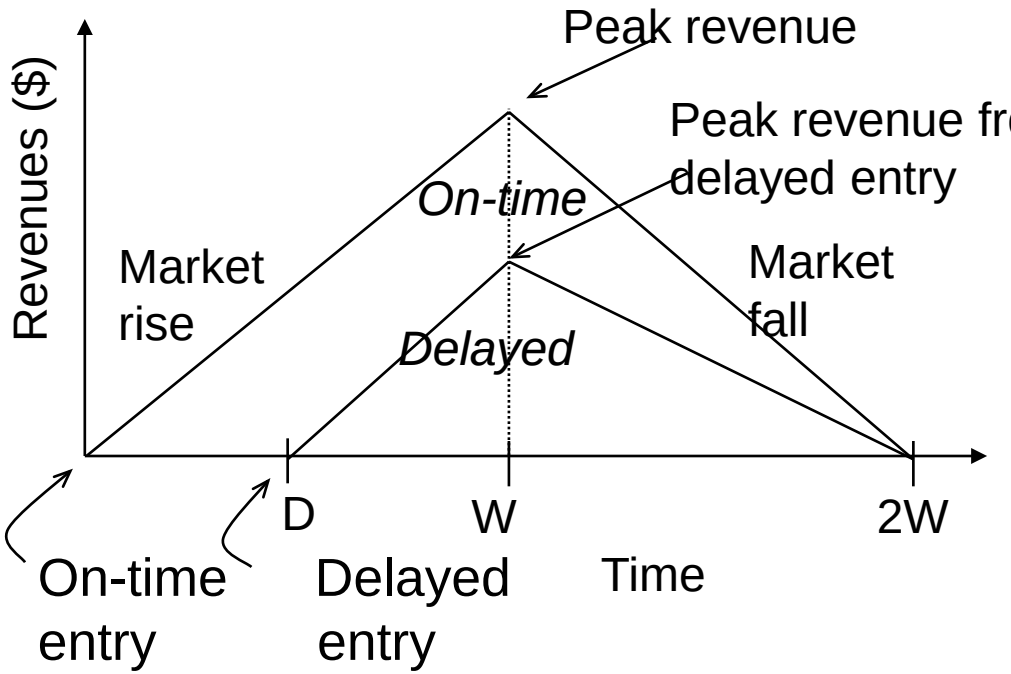
Losses due to delayed market entry

- Simplified revenue model
  - Product life = 2W, peak at W
  - Time of market entry defines a triangle, representing market penetration
  - Triangle area equals revenue
- Loss
  - The difference between the on-time and delayed triangle areas



Losses due to delayed market entry (cont.)

- Area = 1/2 * base * height
  - On-time = 1/2 * 2W * W
  - Delayed = 1/2 * (W-D+W)*(W-D)
- Percentage revenue loss = $(D(3W-D)/2W^2)*100\%$



Try some examples

- Lifetime 2W=52 wks, delay D=4 wks
- (4*(3*26 –4)/2*26^2) = 22%
- Lifetime 2W=52 wks, delay D=10 wks
- (10*(3*26 –10)/2*26^2) = 50%
- Delays are costly!

NRE and unit cost metrics

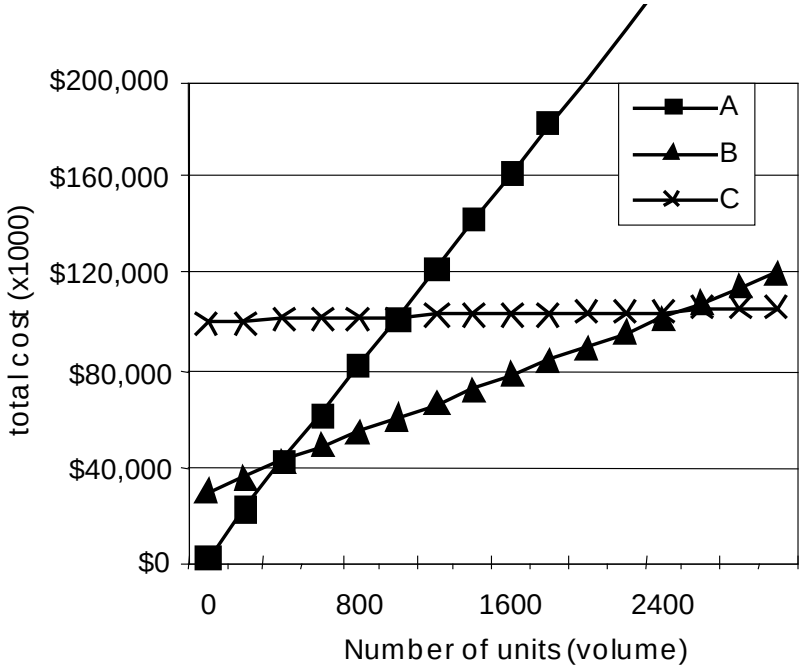
- Costs:
  - Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  - NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
  - *total cost = NRE cost + unit cost * # of units*
  - *per-product cost = total cost / # of units*
    = (NRE cost / # of units) + unit cost

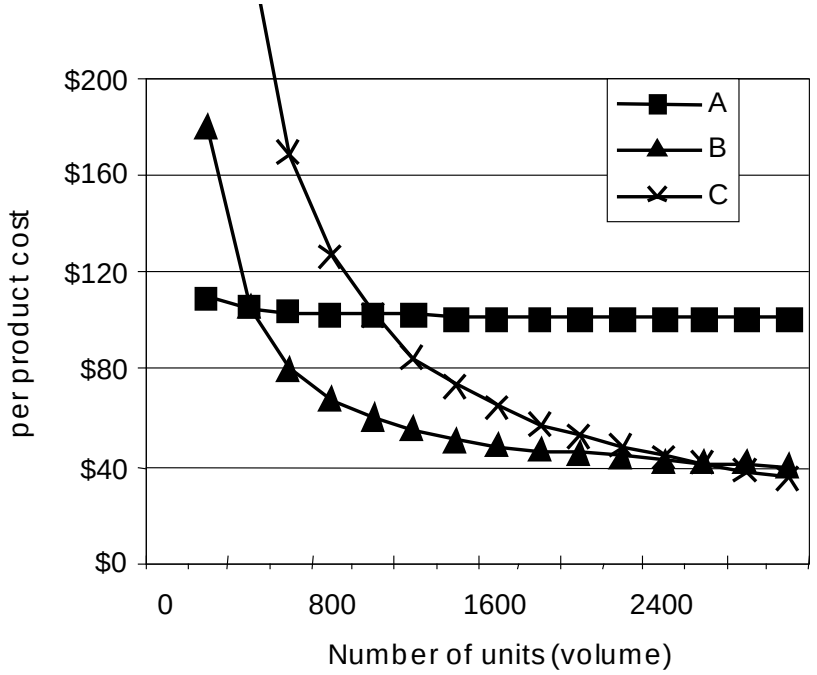
- Example
  - NRE=$2000, unit=$100
  - For 10 units
    - total cost = $2000 + 10*$100 = $3000
    - per-product cost = $2000/10 + $100 = $300

Amortizing NRE cost over the units results in an additional $200 per unit

NRE and unit cost metrics

- Compare technologies by costs -- best depends on quantity
  - Technology A: NRE=$2,000, unit=$100
  - Technology B: NRE=$30,000, unit=$30
  - Technology C: NRE=$100,000, unit=$2





- But, must also consider time-to-market

Class assignment

- Design a embedded system for washing machine

No.	Specification	Describe
1	Product specification	-Washing process: soak, wash, rinse, spin -Mode: manual, fuzzy -Constraints
2	Engineering specification	-Inputs: -Outputs: -Use interface:
3	Hardware specification	-Microcontroller: -Sensors: -Actuators:
4	Software specification	-Functions: -Control algorithm:
5	Test specification	-Platform: -Test process:

Class assignment

1. Consider the following embedded systems: a pager, a computer printer, and an automobile cruise controller. Create a table with each example as a column, and each row one of the following design metrics: unit cost, performance, size, and power. For each table entry, explain whether the constraint on the design metric is very tight. Indicate in the performance entry whether the system is highly reactive or not.
2. List three pairs of design metrics that may compete, providing an intuitive explanation of the reason behind the competition.
3. The design of a particular disk drive has an NRE cost of $100,000 and a unit cost of $20. How much will we have to add to the cost of the product to cover our NRE cost, assuming we sell: (a) 100 units, and (b) 10,000 units.
4. (a) Create a general equation for product cost as a function of unit cost, NRE cost, and number of units, assuming we distribute NRE cost equally among units. (b) Create a graph with the x-axis the number of units and the y-axis the product cost, and then plot the product cost function for an NRE of $50,000 and a unit cost of $5.

Class assignment

Consider the project car door mechanism

Write system specification for this project