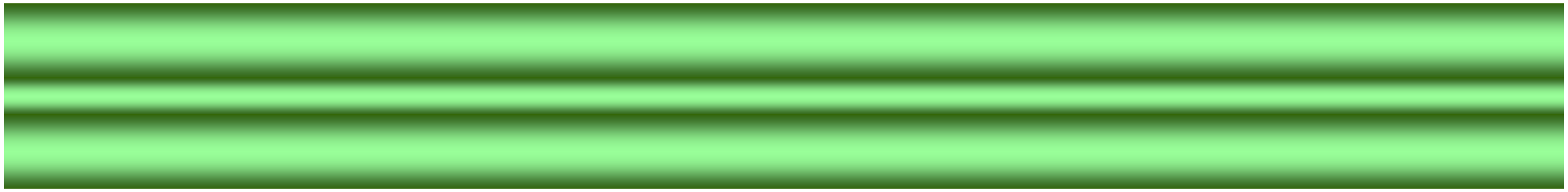
EE414 Embedded Systems

Ch 1. Introduction to Embedded Systems
Part 1/2



Byung Kook Kim
School of Electrical Engineering
Korea Advanced Institute of Science and Technology

Flow of Lectures

- **Logic/Digital Design** (Prerequisite)
- Introduction to **Programming in C**
- Computer System Organizations
- **Embedded Systems**
- VLSI/ASIC Design
- Control Systems
- Digital Signal Processing
- Real-Time Systems

Overview

Introduction

- 1.1 Embedded Systems Overview
  - What are they?
- 1.2 Design challenge – optimizing design metrics

Technologies

- 1.3 Processor technologies
- 1.4 IC technologies
- 1.5 Design technologies

- 1.6 Trade-offs

1.1 Embedded Systems Overview

- *Computing systems are everywhere.*

- Most of us think of "desktop" computers

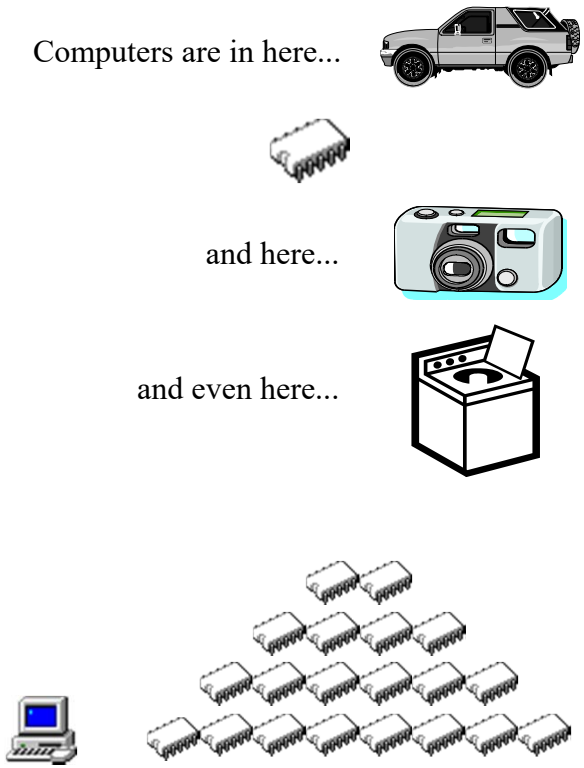
  - PC's
  - Laptops
  - Mainframes [1]
  - Servers [2]

- But there's another type of computing system
  - Far more common...

1. http://goseerobert.com/2008/02/08/ibm-tapping-teens-for-mainframe-careers/
2. http://en.wikipedia.org/wiki/Server_(computing)

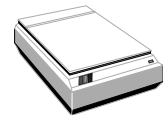
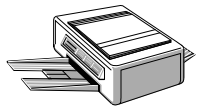
Embedded systems overview

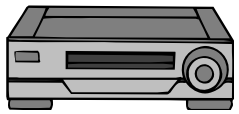
- **Embedded [computing] systems**
  - Computing systems embedded within electronic devices
  - Hard to define. Nearly any computing system other than a desktop computer
  - Billions of units produced yearly, versus millions of desktop units
  - Perhaps 50 per household and per automobile

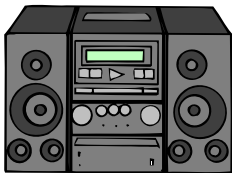


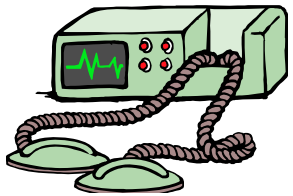
A "short list" of embedded systems

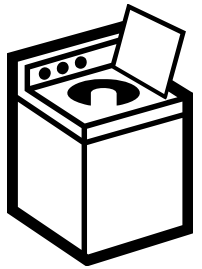
Anti-lock brakes
Auto-focus cameras
Automatic teller machines
Automatic toll systems
Automatic transmission
Avionic systems
Battery chargers
Camcorders
Cell phones
Cell-phone base stations
Cordless phones
Cruise control
Curbside check-in systems
Digital cameras
Disk drives
Electronic card readers
Electronic instruments
Electronic toys/games
Factory control
Fax machines
Fingerprint identifiers
Home security systems
Life-support systems
Medical testing systems
Modems
MPEG decoders
Network cards
Network switches/routers
On-board navigation
Pagers
Photocopiers
Point-of-sale systems
Portable video games
Printers
Satellite phones
Scanners
Smart ovens/dishwashers
Speech recognizers
Stereo systems
Teleconferencing systems
Televisions
Temperature controllers
Theft tracking systems
TV set-top boxes
VCR's, DVD players
Video game consoles
Video phones
Washers and dryers





And the list goes on and on…

Some common characteristics of embedded systems

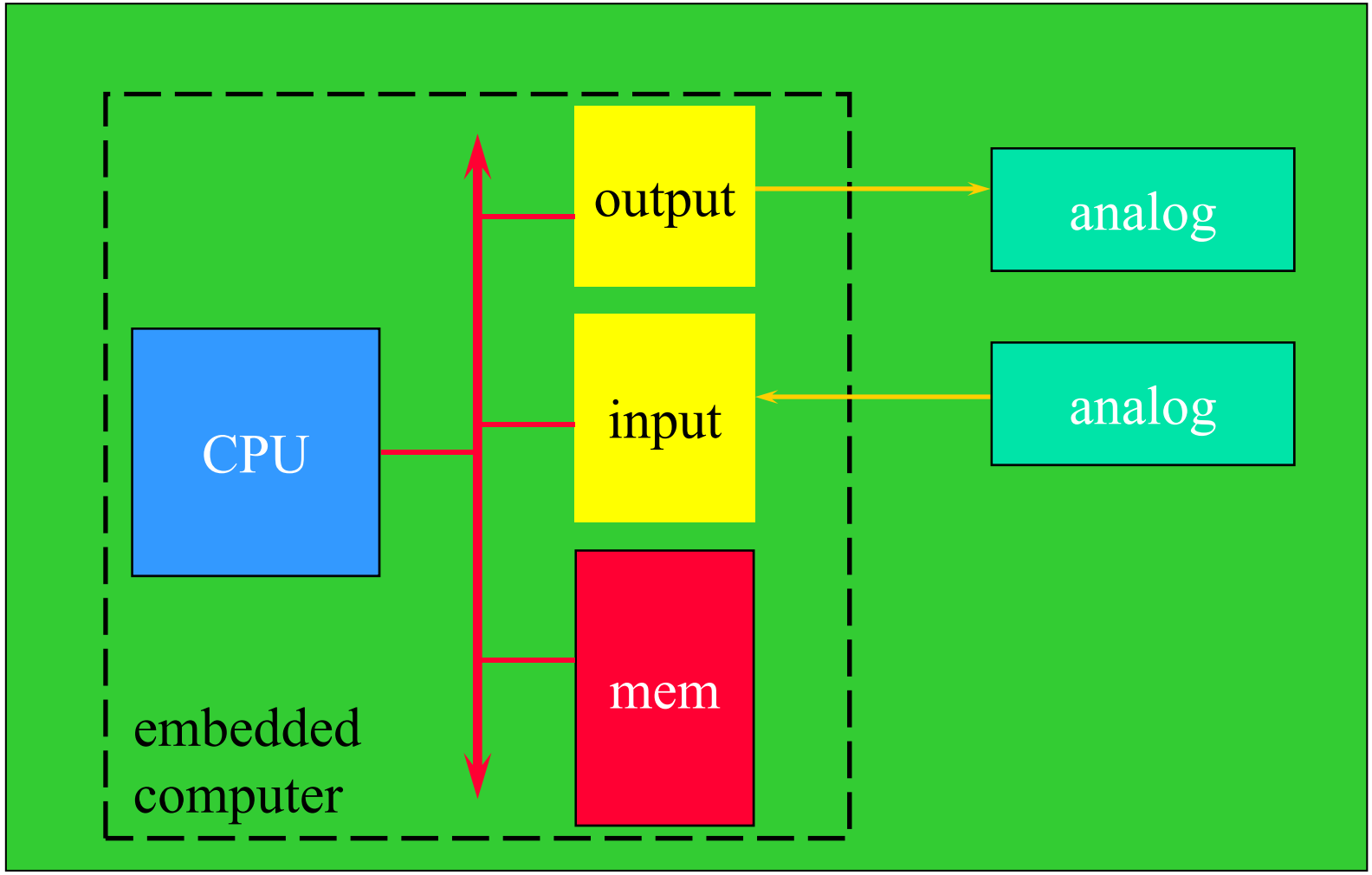
Common characteristics

- Single-functioned
  - Executes a single program, repeatedly.
- Tightly-constrained
  - Low cost, low power, small, fast, etc.
- Reactive and real-time
  - Continually reacts to changes in the system's environment.
  - Must compute certain results in real-time without delay.

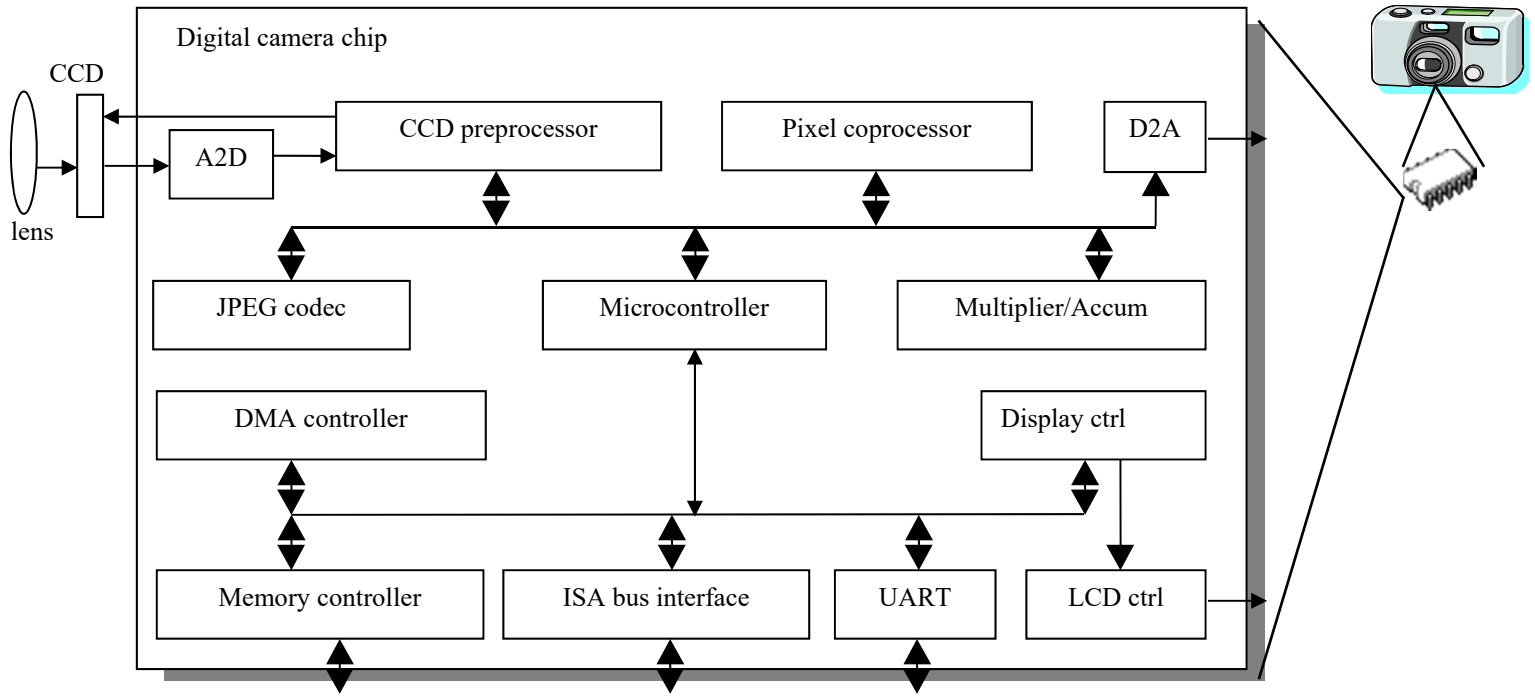
Definition of Embedded Systems

- Embedded system (임베디드 시스템, 내장형 시스템)
  - Electronic systems including hardware and software to control systems designed for specific purposes
    - (특정 목적을 수행하기 위해 설계된 기기를 제어하기 위한 hardware와 software가 내장된 전자 응용 시스템)
  - Remarks
    - Microprocessor or microcontroller based system
    - Functionality implementation with effective combination of hardware and software.
    - PC is a general-purpose system.

Embedding a computer



An embedded system example 1 -- a digital camera



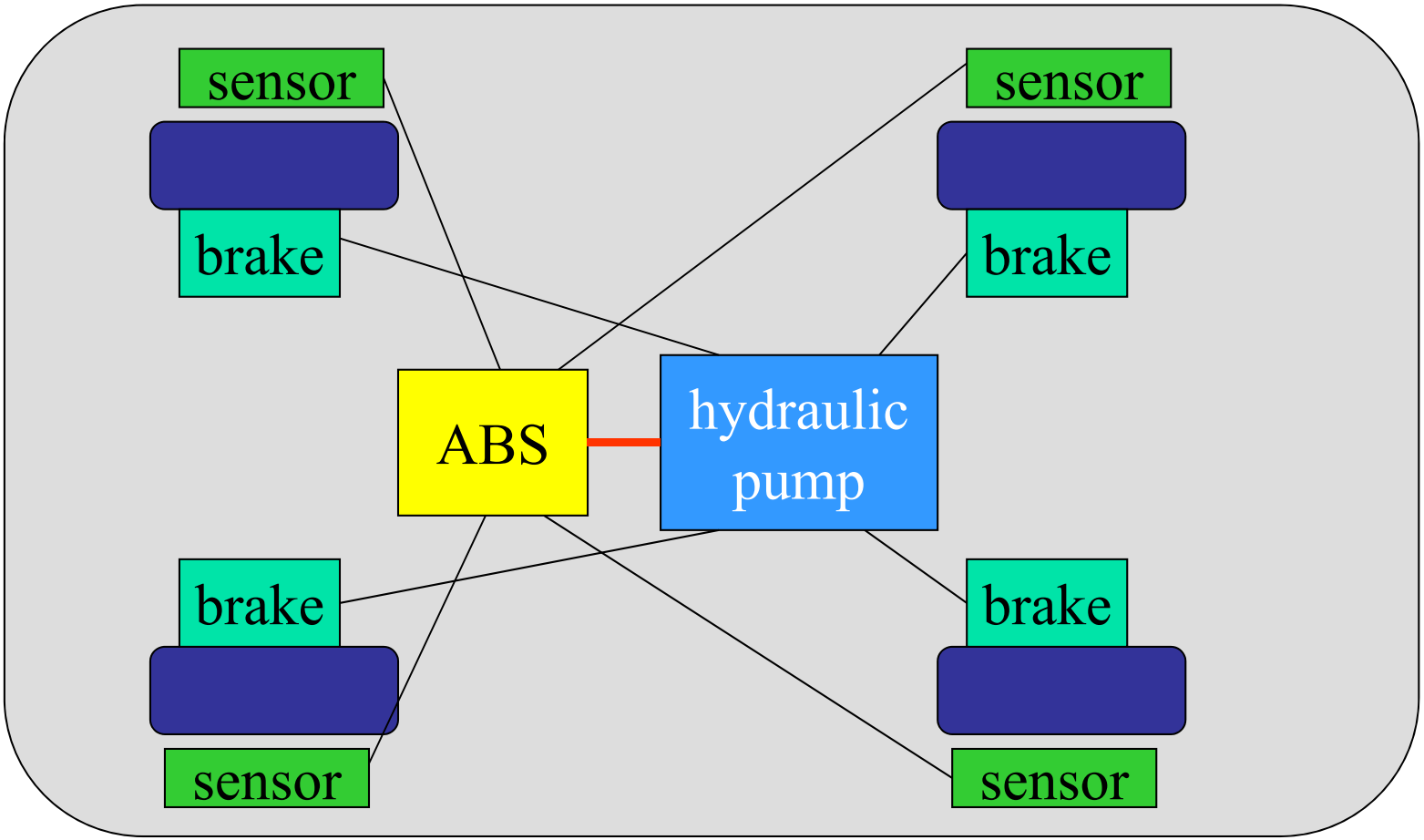
- Single-functioned -- always a digital camera
- Tightly-constrained -- Low cost, low power, small, fast
- Reactive and real-time -- only to a small extent (Shutter response)

Ex 2. Automotive embedded systems

- Today's high-end automobile may have 100 microprocessors [2]:
  - 4-bit microcontroller checks seat belt;
  - microcontrollers run dashboard devices;
  - 16/32-bit microprocessor controls engine.

- BMW 850i
  - Anti-lock brake system (ABS): pumps brakes to reduce skidding.
  - Automatic stability control (ASC+T): controls engine to improve stability.
  - ABS and ASC+T communicate.
    - ABS was introduced first---needed to interface to existing ABS module.

BMW 850i - ABS


sensor
sensor
brake
brake
ABS
hydraulic pump
brake
brake
sensor
sensor

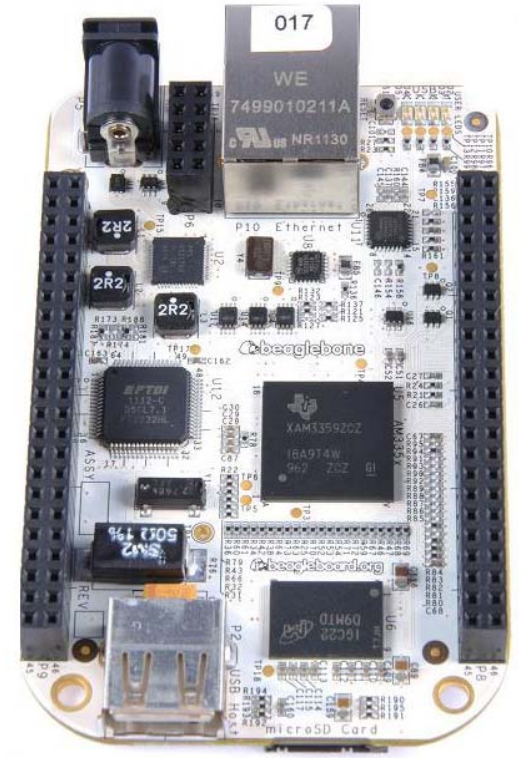
Labs in Embedded Systems

- A simple design and implementation of an embedded system
- Establish capability of embedded system design for various fields in the future.
- Especially, a project which requires no additional hardware is formulated:

Design of a Metronome.

- It will be designed and implemented during this semester.

Embedded Board for Labs



1.2 Design Challenge – Optimizing Design Metrics

- Obvious design goal:
  - Construct an implementation with desired functionality.
- Key design challenge:
  - Simultaneously optimize numerous design metrics.
- **Design metric**
  - A measurable feature of a system's implementation
  - Optimizing design metrics is a key challenge

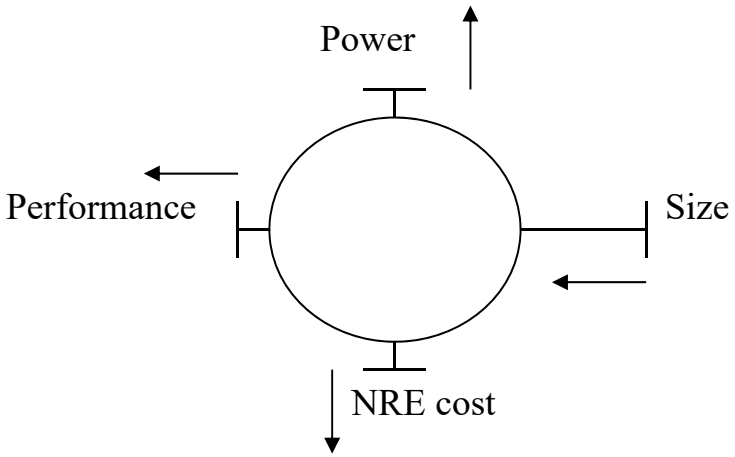
Design challenge – optimizing design metrics

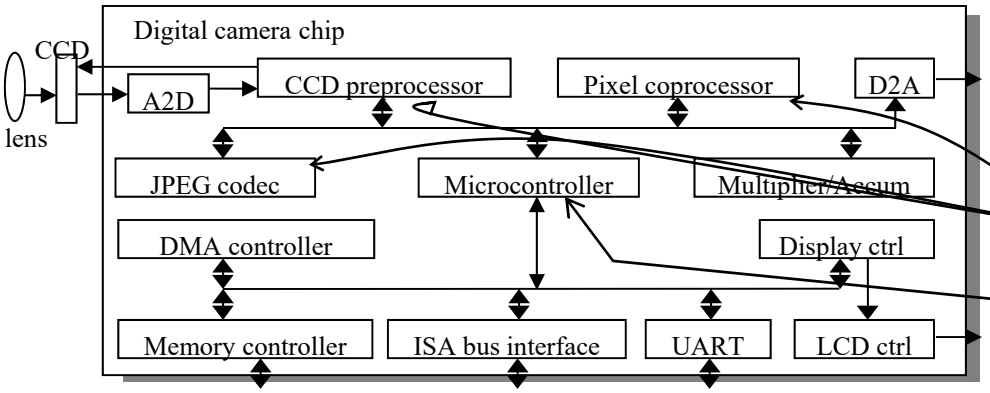
- Common design metrics
  - Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost
  - NRE cost (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
  - Size: the physical space required by the system
  - Performance: the execution time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

Design challenge – optimizing design metrics

- Common design metrics (continued)
  - Time-to-prototype: the time needed to build a working version of the system
  - Time-to-market: the time required to develop a system to the point that it can be released and sold to customers
  - Maintainability: the ability to modify the system after its initial release
  - Correctness: Functional correctness
  - Safety: Harmless
  - And many more…
- Metrics typically compete each other.

Design metric competition – improving one may worsen others

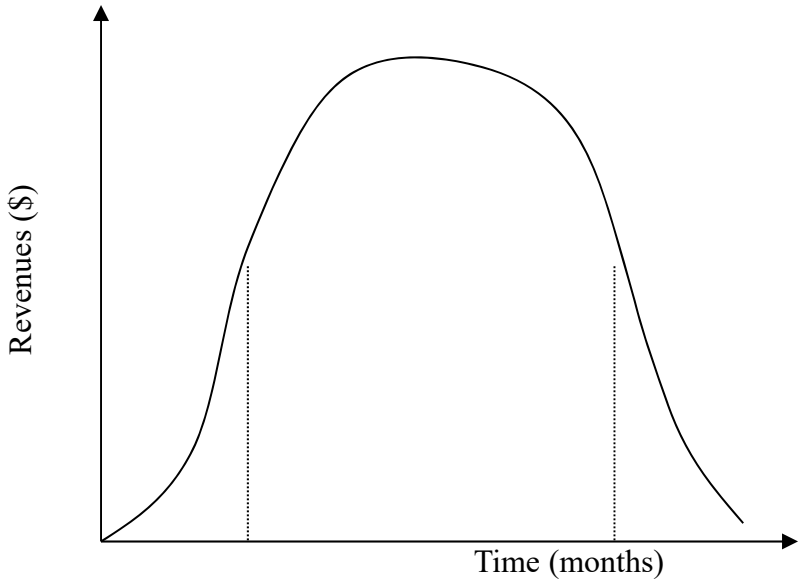




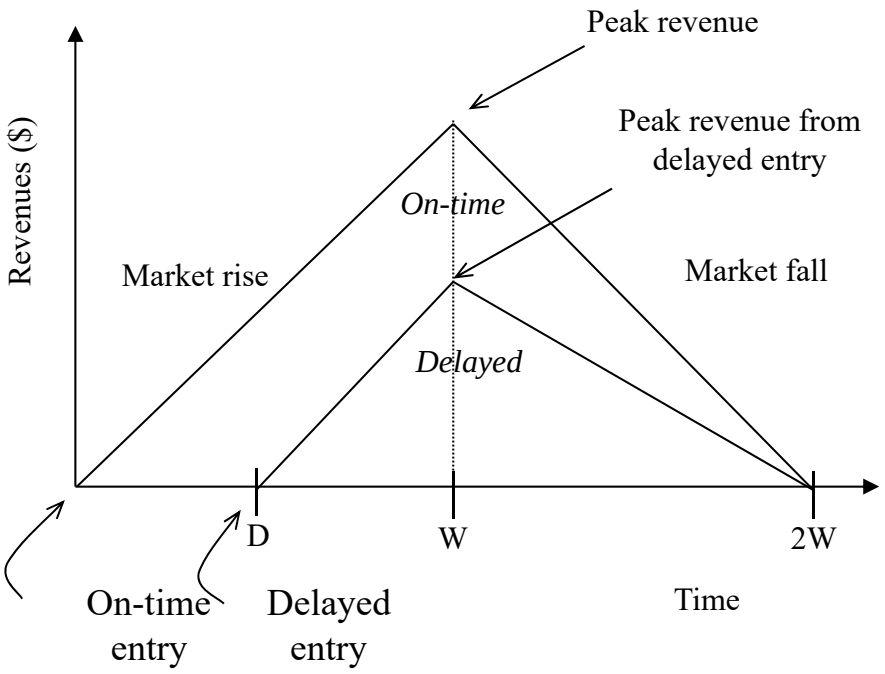
- Expertise with both **software and hardware** is needed to optimize design metrics
  - Not just a hardware or software expert, as is common
  - A designer must be comfortable with various technologies in order to choose the best for a given application and constraints.

A. Time-to-market:
a demanding design metric

- 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.

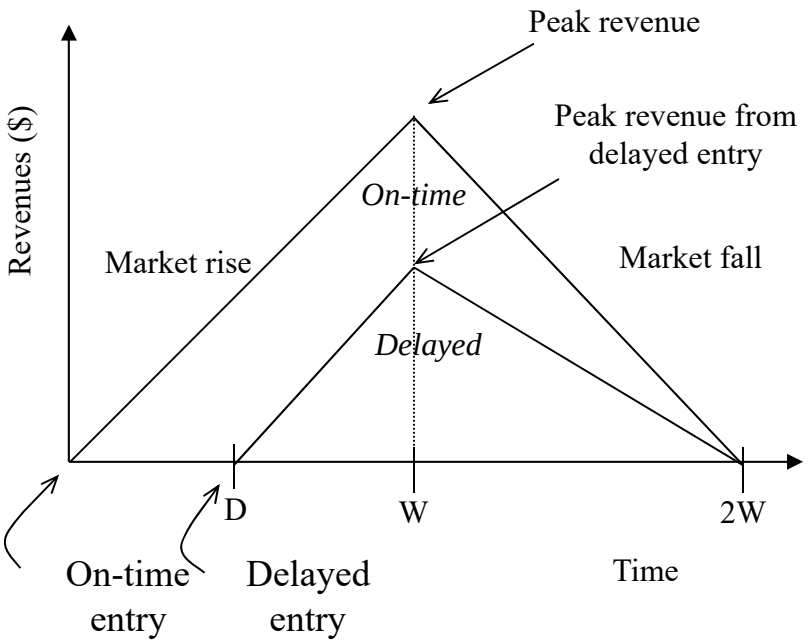


Losses due to delayed market entry



- 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 = a = 1/2 * 2W * W
  - Delayed = d = 1/2 * (2W-D)*(W-D)
- Percentage revenue loss
  - L = (a – d)/a = (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!**

B. 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

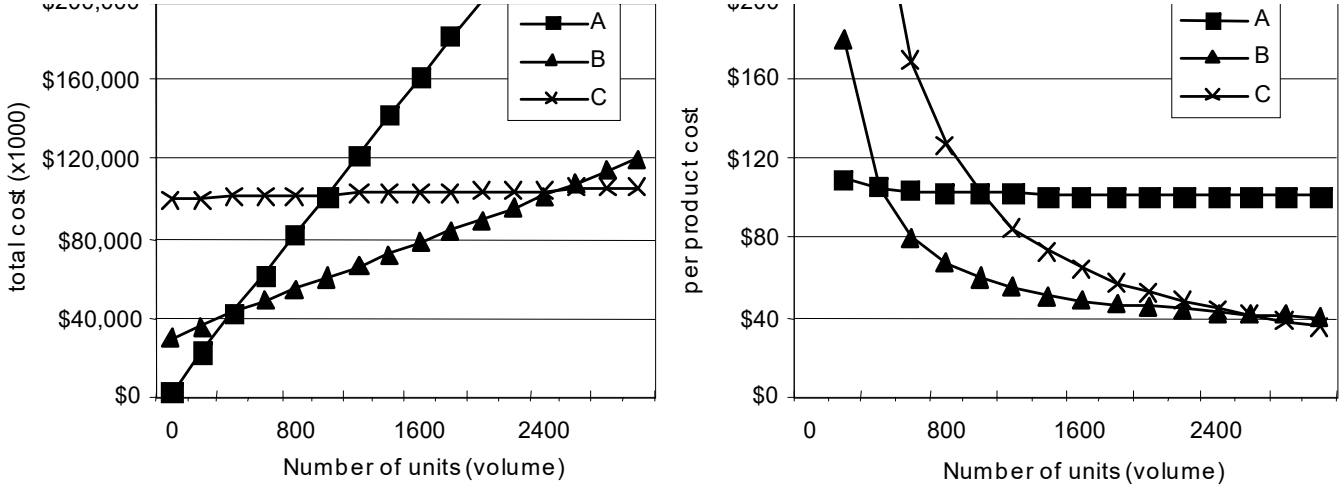
  - *total cost = NRE cost + unit cost * # of units*
  - *per-product cost = total cost / # of units*
    = (NRE cost / # of units) + unit cost

- 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



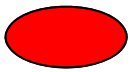
- But, must also consider time-to-market

C. The performance design metric

- Performance
  - How long the system takes to execute to execute desired tasks.
- 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 A = B's performance / A's performance = 8/4 = 2.

Summary

- Embedded systems are all around us.
  - Many systems have complex embedded hardware and software.
- Embedded systems pose many design challenges: design time, deadlines, power, etc.
  - Optimizing design metrics.



References

- [1] Frank Vahid, "Embedded system design: A unified hardware/software introduction", John Wiley & Sons, 2002.

- [2] Wayne Wolf, "Computers and Components: Principles of Embedded Computing System Design," Morgan Kaufman, 2001.