Embedded Design Concepts and Definitions

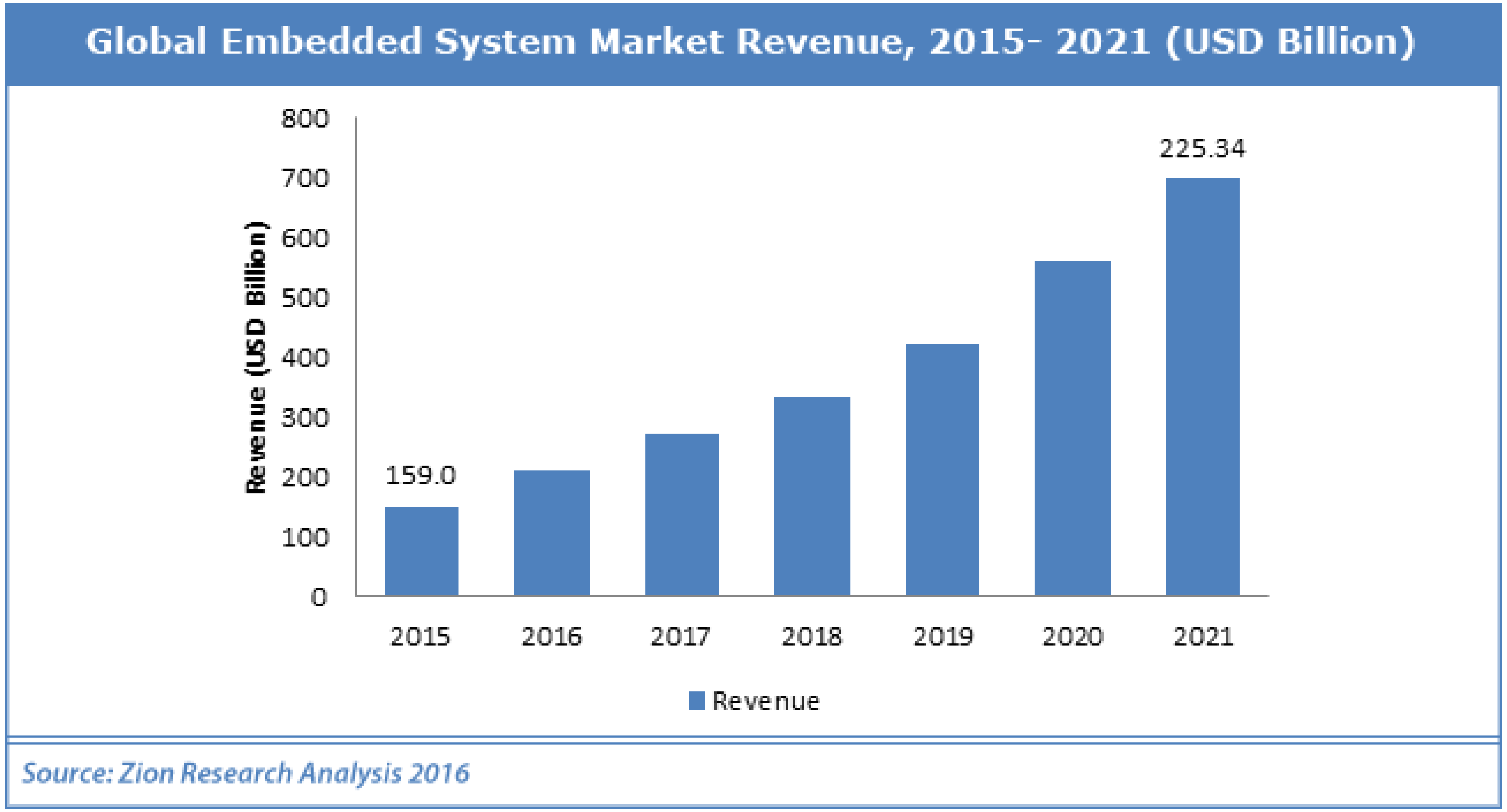
ET2223 Microprocessors, Microcontrollers, and Embedded Systems

Partially based on
Embedded Systems Design: A Unified Hardware/Software Introduction by Vahid/Givargis

Embedded systems

- An application specific electronic sub-system which is completely encapsulated by the main system it belongs to.
    - The main systems can range from household appliances, home automation, consumer electronics, ATMs, network routers, automobiles, aircrafts, etc.
- Designed for some specific tasks
- Feature tightly integrated combinations of hardware and software
    - Small foot prints (in memory)
    - Highly optimized code
- Subjected to real time performance constraints that must be met

Embedded systems



Embedded systems basics

- Embedded systems are designed for a specific task.
  - Although they use computer techniques, they cannot be used as a general purpose computer using a variety of different programmes for different task. In this way their function can be focussed onto what they need to do, and they can accordingly be made cheaper and more efficiently.
- The software for embedded systems is referred to as firmware.
  - Rather than being stored on a disc, where many programmes can be stored, the single programme for an embedded system is normally stored on chip and it is referred to as firmware.

Embedded systems hardware

- When using an embedded system there is a choice between the use of a microcontroller or a microprocessor.
  - ***Microcontroller based systems:***
    - A microcontroller is essentially a CPU, central processor unit, or processor with integrated memory or peripheral devices. As fewer external components are needed, embedded system using microcontrollers tend to be more widely used
  - ***Microprocessor based systems:***
    - Microprocessors contain a CPU but use external chips for memory and peripheral interfaces. As they require more devices on the board, but they allow more expansion and selection of exact peripherals, etc, this approach tends to be used for the larger embedded systems.

Embedded systems software

- One of the key elements of any embedded system is the software that is used to run the microcontroller.
- There is a variety of ways that this can be written:
  - ***Machine code:***
    - Machine code is the most basic code that is used for the processor unit. The code is normally in hex code and provides the basic instructions for each operation of the processor. This form of code is rarely used for embedded systems these days.
  - ***Programming language:***
    - Writing machine code is very laborious and time consuming. It is difficult to understand and debug. To overcome this, high level programming languages are often used. Languages including C, C++, etc are commonly used.

Characteristics of embedded systems

- Must be dependable:
  - Reliability:
    - R(t) = probability of system working correctly provided that is was working at t=0
  - Maintainability:
    - M(d) = probability of system working correctly d time units after error occurred.
  - Availability:
    - probability of system working at time t
  - Safety:
    - no harm to be caused
  - Security:
    - confidential and authentic communication

Characteristics of embedded systems

- Must be efficient:
    - Energy efficient
    - Code-size efficient (especially for systems on a chip)
    - Run-time efficient
    - Weight efficient
    - Cost efficient
- Dedicated towards a certain application:
    - Knowledge about behaviour at design time can be used to minimize resources and to maximize robustness.
- Dedicated user interface (no mouse, keyboard and screen).

Characteristics of embedded systems

- Many ES must meet real-time constraints:
  - A real-time system must react to stimuli from the controlled object (or the operator) within the time interval dictated by the environment.
  - For real-time systems, right answers arriving too late (or even too early) are wrong.
  - All other time-constraints are called soft.
  - A guaranteed system response has to be explained without statistical arguments.

“A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe” [Kopetz, 1997].

Characteristics of embedded systems

- Frequently connected to physical environment through sensors and actuators.
- Hybrid systems (analog + digital parts).
- Typically, ES are reactive systems:
  - Behaviour depends on input and current state.

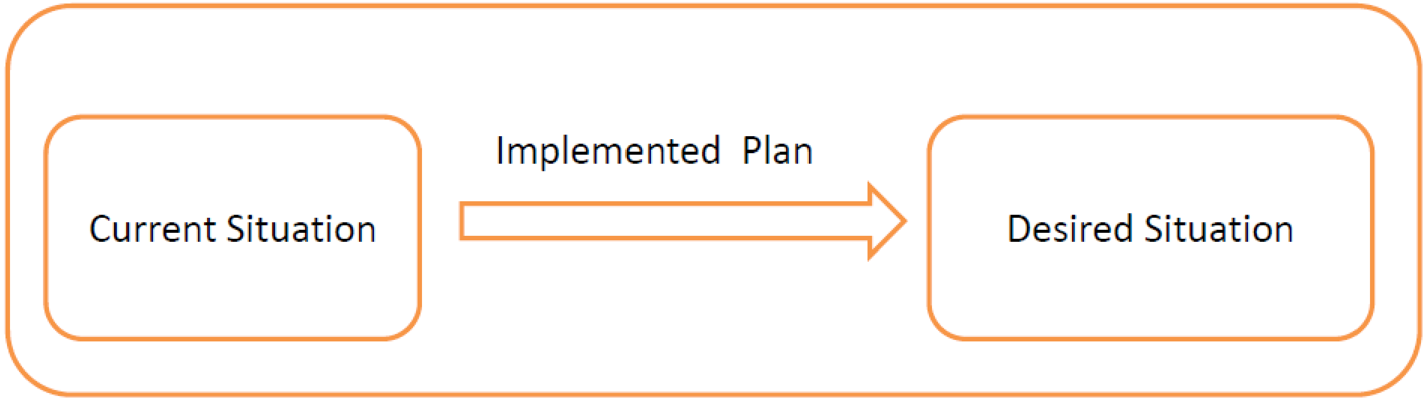
“A reactive system is one which is in continual interaction with is environment and executes at a pace determined by that environment” [Bergé, 1995]

What is Design?

What is Design?

- Village women walk to water source
- Roads to water source are bad
- Large distance away from home
- Carry clothes which become heavier
- Also carry water back
- For safety go in groups

- Women cycle to water source
- Carry clothes on cycle
- Also a washing machine
- Pedal cycle to power washing
- Less effort, faster washing
- Less effort, faster to water source

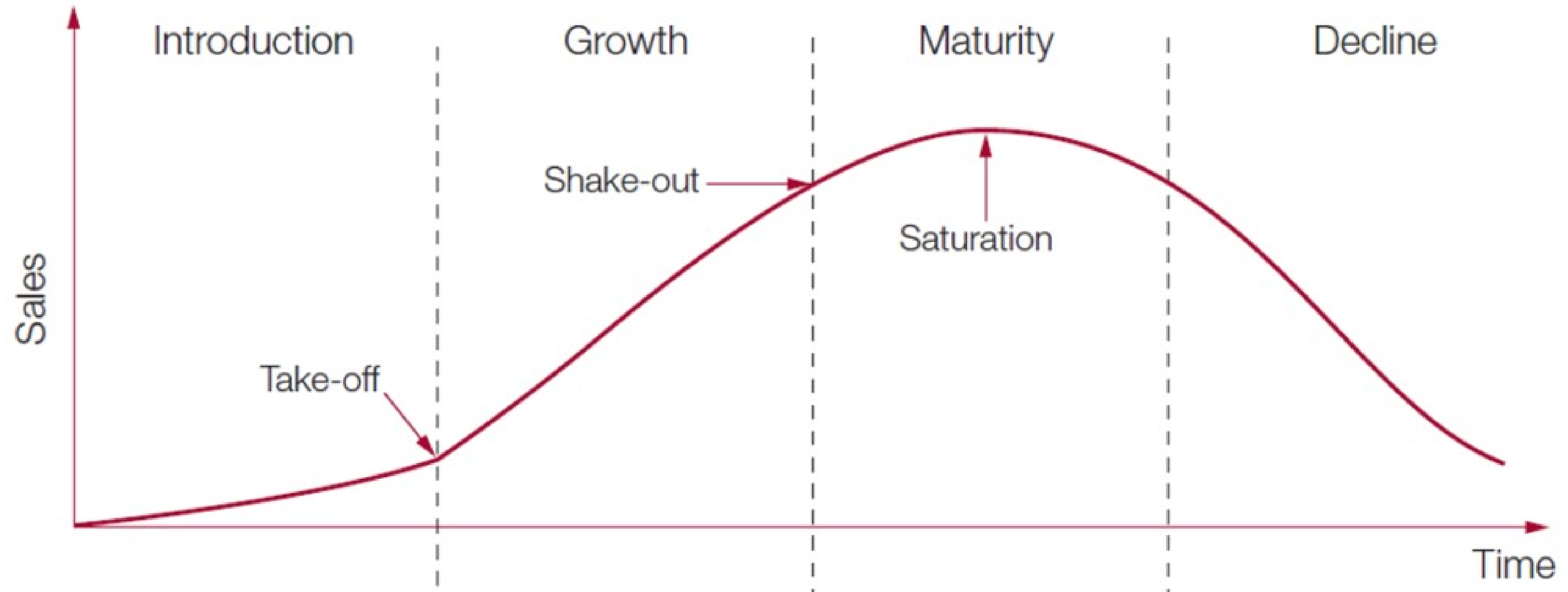


What is product design?

- A creative activity – involves bringing into being something new and useful that has not existed before (Reswick, 1965).
- Process of devising and laying down the plans needed for manufacturing a product.
- From:
  - Need: Not fully defined, not fully structured
- To:
  - Plan: Well-defined, well-structured

Why is design important?

The 4 Life Cycle Stages and their Marketing Implications



Introduction	Growth	Maturity	Decline
• Low sales	• Increasing sales	• Peak sales	• Falling sales
• High cost per customer	• Cost per customer falls	• Cost per customer lowest	• Cost per customer low
• Financial losses	• Profits rise	• Profits high	• Profits fall
• Innovative customers	• Increasing No. of customers	• Mass market	• Customer base contracts
• Few (if any) competitors	• More competitors	• Stable number of competitors	• Number of competitors fall

Why is design important?

- Innovation is needed for continues success of any venture
- Product design is an essential part of the industrial innovation process which is important for both society and business
- Product design is an early stage of product development, where it is inexpensive to make changes, but consequences of changes is substantial

System design

- Meaning as noun: a design
  - A **plan** for change from existing undesired to a desired situation
  - An engineering drawing, CAD model, flow chart etc.
- Meaning as verb: the act of designing
  - **Processes** through which designs are developed
  - Both **goal** and **plan**
- Designs can be for:
  - technical systems (power plant), educational systems (Montessori Method), aesthetic systems (logo designs, advertisements), legal systems, social, religious or cultural systems, theories, Models, etc.

System design

- How to develop ‘good’ designs?
  - Initially only **goals** are known better
  - But, finally both goals and **plans** are known and more clearly
  - Co-evolution: **both** goals and plans evolve together, one influencing the other

- Designing does **NOT** guarantee that designs will work. Some designing may be better than others in achieving goals.
  - Multiple goals: some goals are more **important** than others
  - Multiple plans: some plans are **better** than others

Design process

- System identification
  - understand the process and identify the relationship between input and output
- Requirement definition
  - determining the needs or conditions to meet for a new or altered product, taking into account the possibly conflicting (technical, functional and non-functional) requirements of the various stakeholders, such as beneficiaries, legal parties or users.
- System specification
  - describing the requested behaviour of the system.
- Functional design
  - defining which subprocesses (or components) are needed in the system.
- Detailed design
  - resulting in a concrete structure of all modules from which the system must be made.
- Implementation
  - building and testing prototypes of the eventual system.

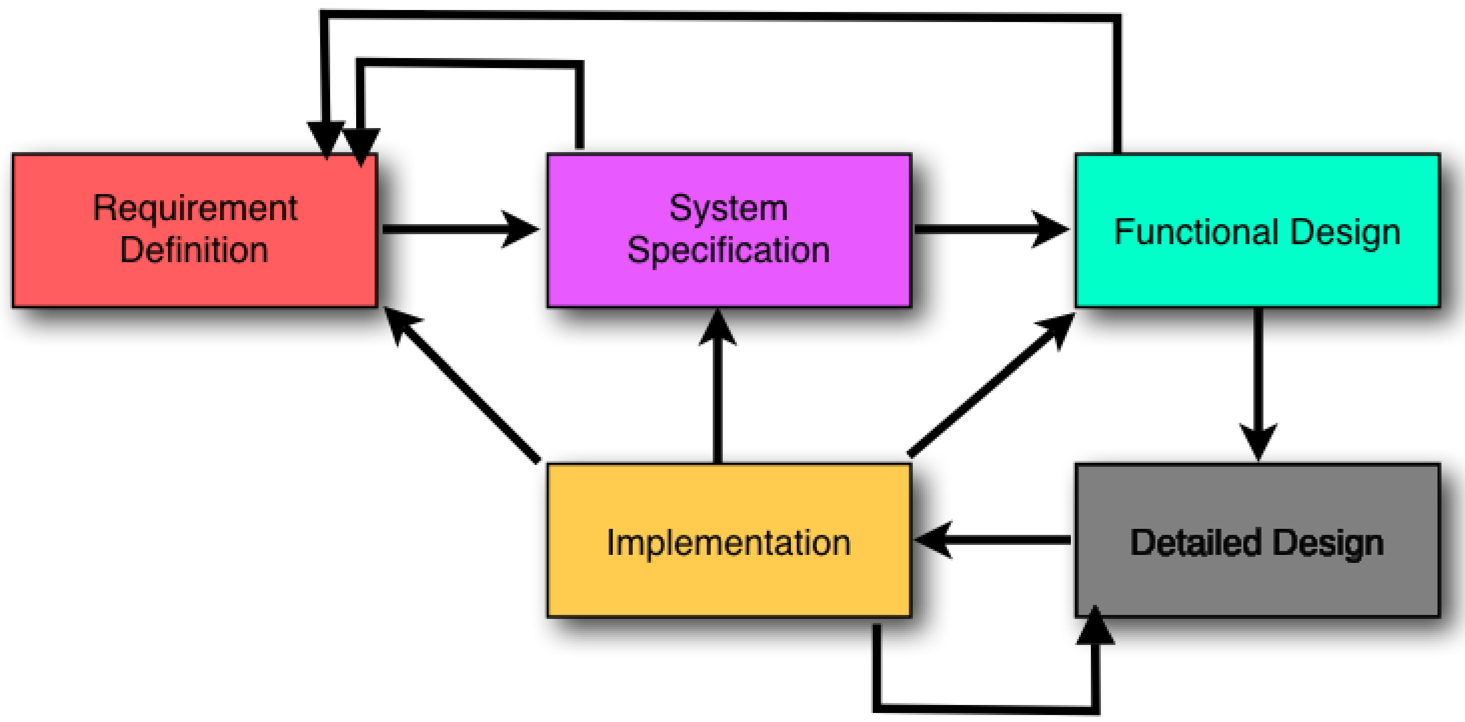
Traditional design process models

- Waterfall model
  - move to a phase only when its preceding phase is completed and perfected. Phases of development in the waterfall model are kept very separated.
- V model
  - has the same strict serial structure as the waterfall model, but it suggests that, before going to a more detailed design level, one should already test all the system features and properties that can be tested at the current level of design abstraction.
- Incremental model
  - allows multiple iterations in some of the design phases, resulting in a multi-waterfall process.
- Spiral model
  - similar to the incremental model, with more emphases placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model).

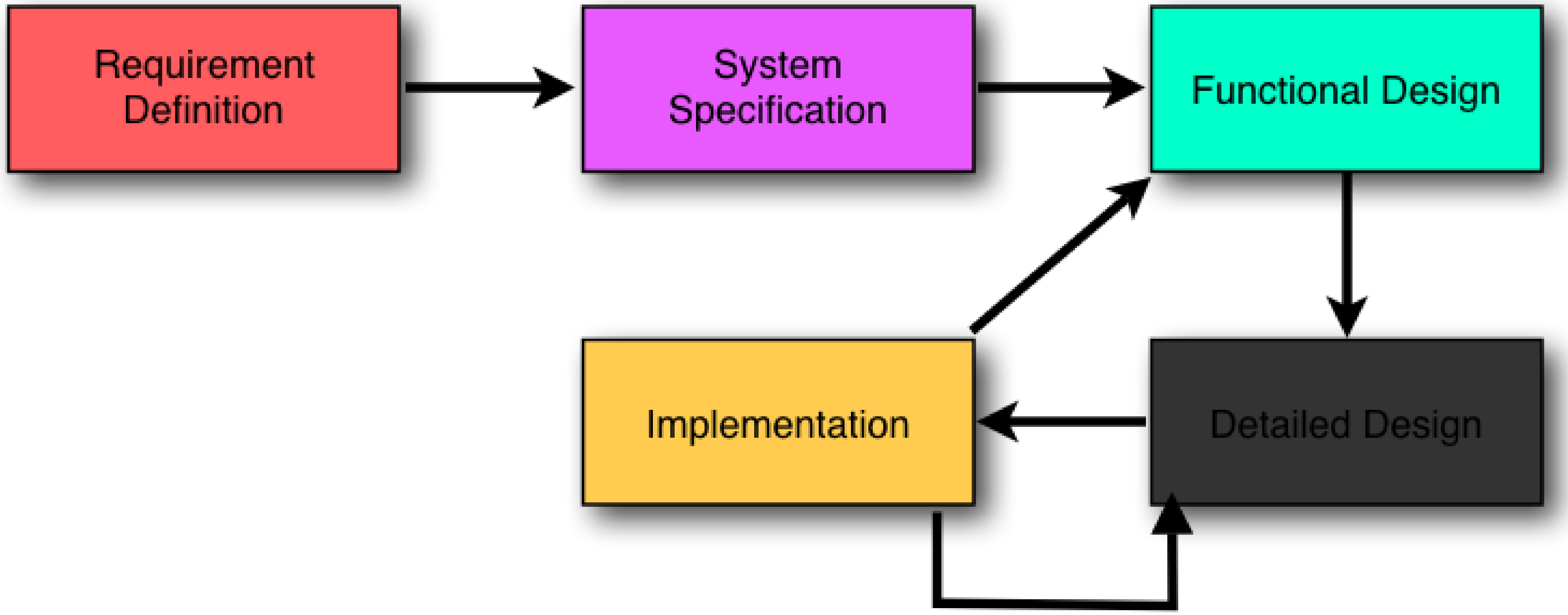
Design contexts

- In practice, the design process of a particular system is not only determined by what phases to go through, in what order, but also by the particularities of the context in which the system is to be designed:
  - From Scratch Environment
  - Adapting Environment
  - Competition Environment
  - Research Environment

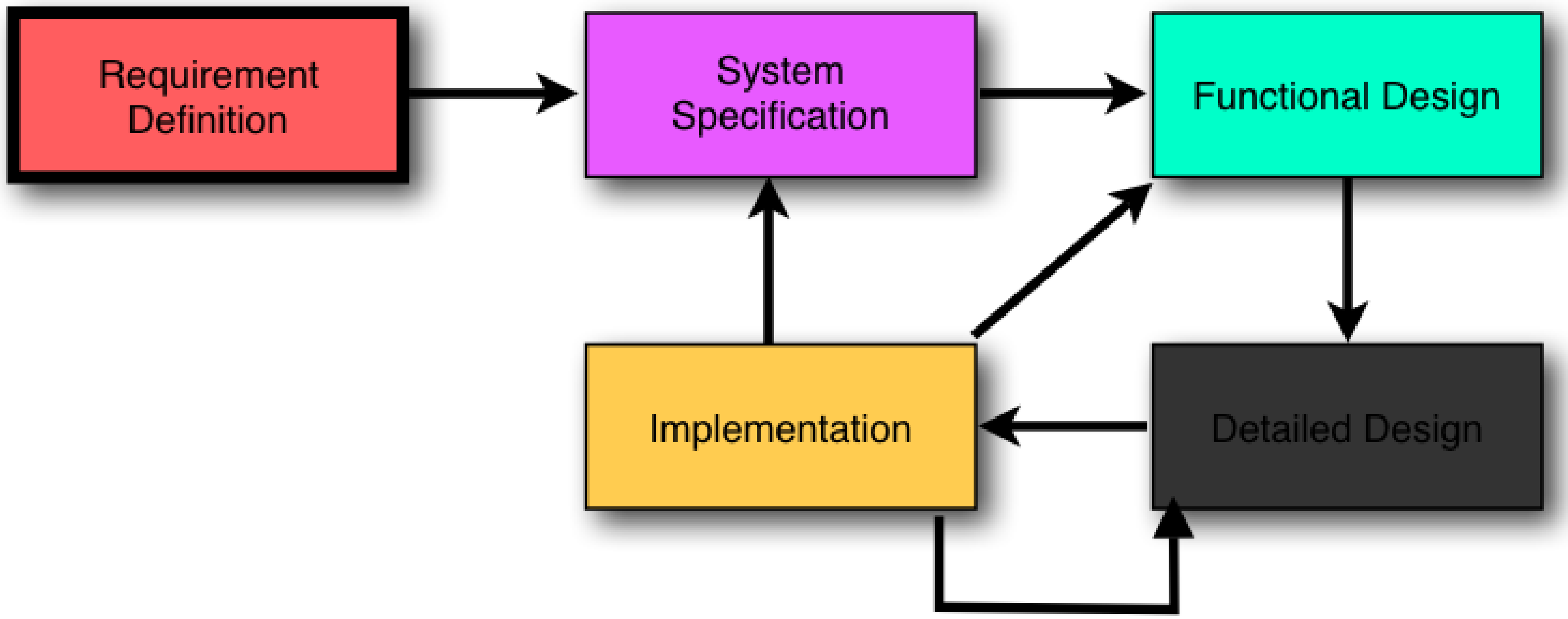
Design process in a from scratch environment



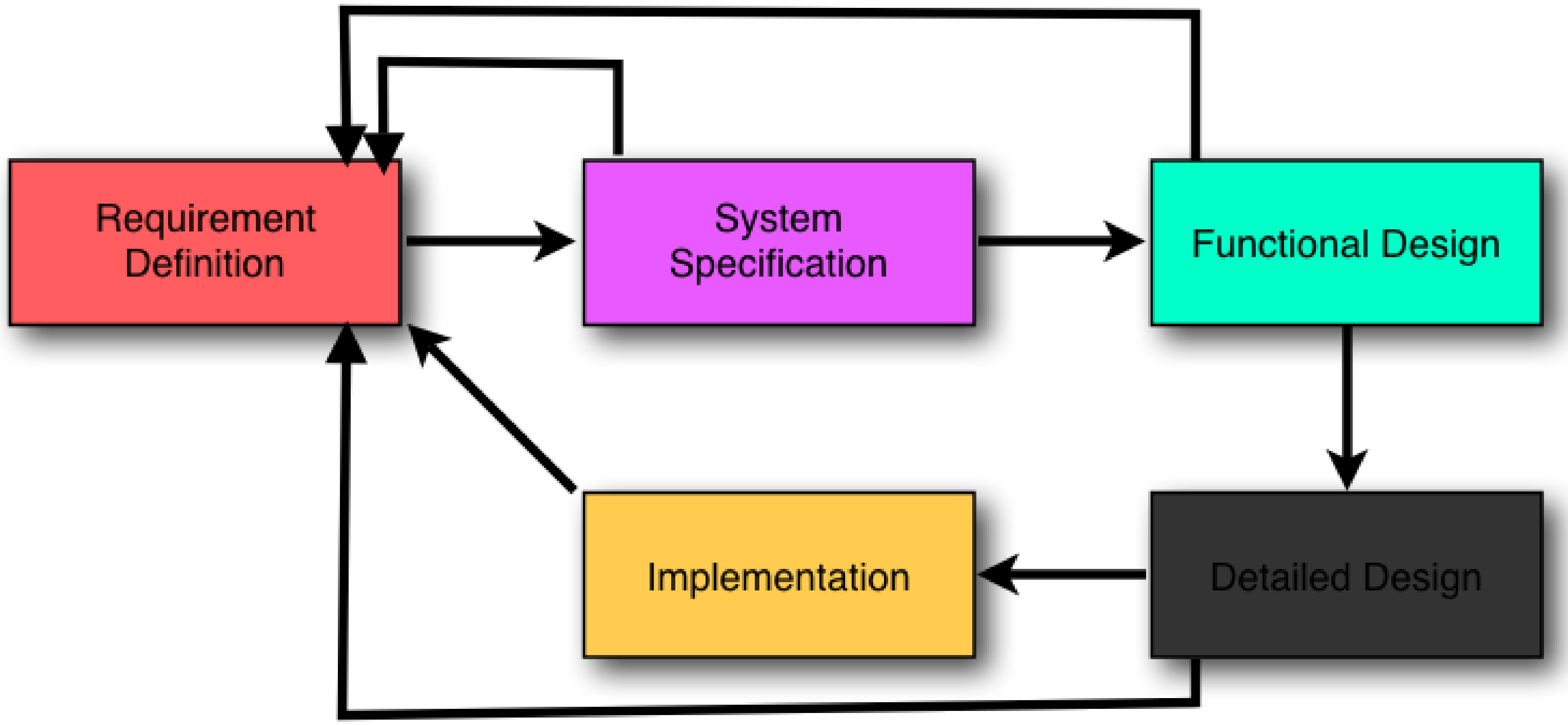
Design process in an adapting environment



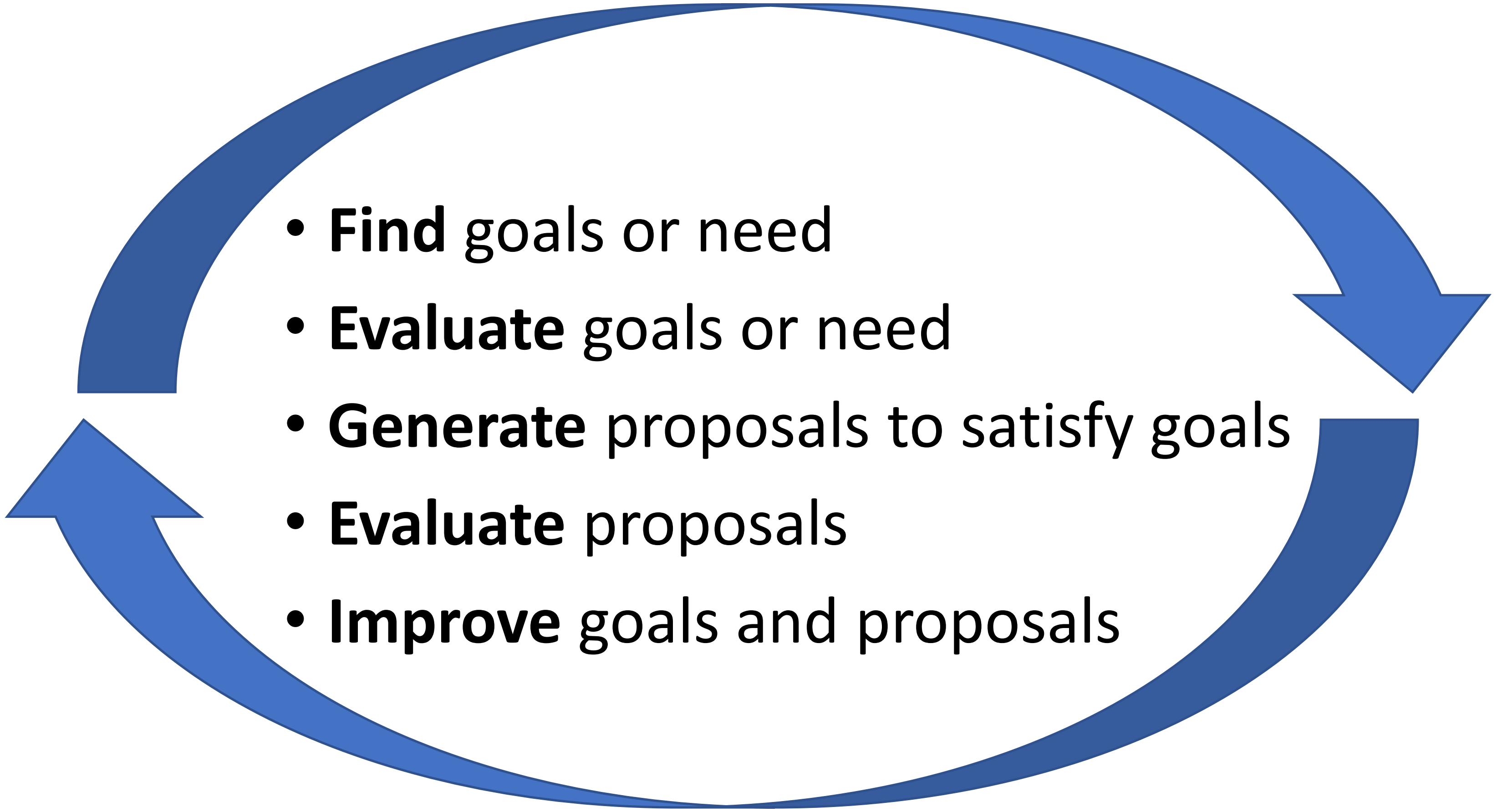
Design process in a competition environment



Design process in a research environment



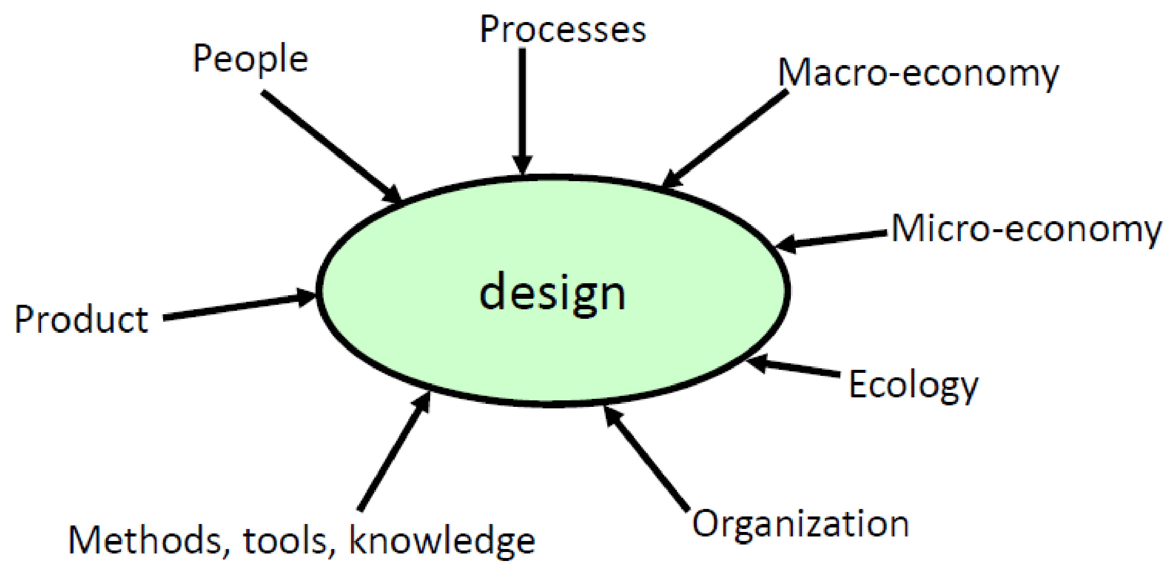
Design Thinking Process



- **Find** goals or need
- **Evaluate** goals or need
- **Generate** proposals to satisfy goals
- **Evaluate** proposals
- **Improve** goals and proposals

Facets of Design

- Designing is planning for changing existing, undesired situations into preferred ones
- Influenced by people, product, process, tools, organization, economy and ecology
- Multi-disciplinary: uses knowledge from human, natural, engineering, ecological, etc. sciences
- Develops necessary knowledge when knowledge is not available for designing



Design Research

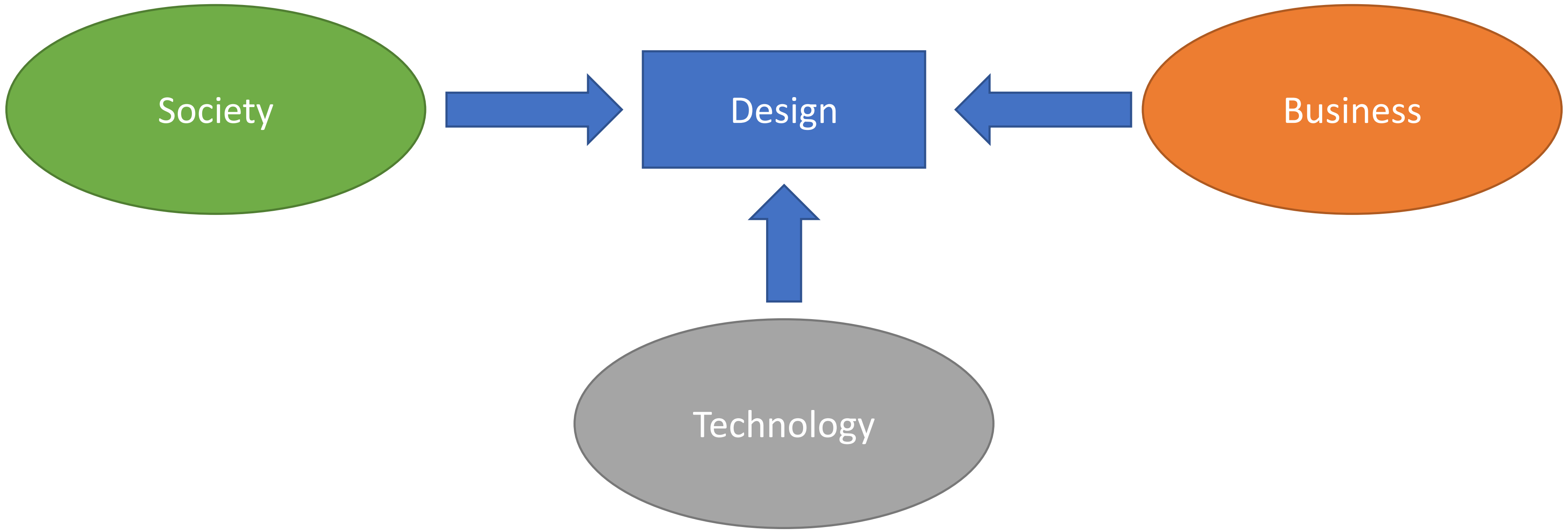
- Systematic study of design phenomena
- Develop knowledge about the design
  - Purposeful: Describes/explains/predicts design system behaviour
  - New: Not before
  - Generic: Applies to multiple things, cases, people…
  - Valid: Has some sense of truth

Design Research

- Develops knowledge in the form of
  - **Theories/models:** Theory of Technical Systems, Integrated Model of designing
  - **Guidelines:** Design for Manufacture and Assembly (Boothroyd-Dewhurst)
  - **Methods:** Weighted Objectives method for comparative evaluation
  - **Tools:** Sketchpad – a tool for sketching using GUI (Sutherland, 1963)
  - **Standards:** IDEF0 standards for representing processes
  - **Materials:** Ferromagnetic-composite material for light, conducting aircraft body
  - **Processes:** CNC processes for computer aided machining
  - **Technologies:** Graphical User Interfaces (GUI); micro-pressure-sensors…
- • To help develop successful products by making designing
  - **More effective:** better products – novelty, quality, reliability…
  - **More efficient:** less resources – less time to market, iterations, cost…

Society, Business, Technology

- Design draws knowledge from Society, Business and Technology
- Develops or integrates technology to provide value to society to fulfil its needs



Design for Society: Value

- Need domain knowledge of user/problem
- Processes of knowledge: how to find the needs of society
  - Focus groups
  - Innovation situation
  - Questionnaire
  - Immersion

- Products must perform (function) and be:
  - Safe
  - Reliable
  - Economic
  - Sustainable
  - Ergonomic
  - Aesthetic

Design for Business: Profit

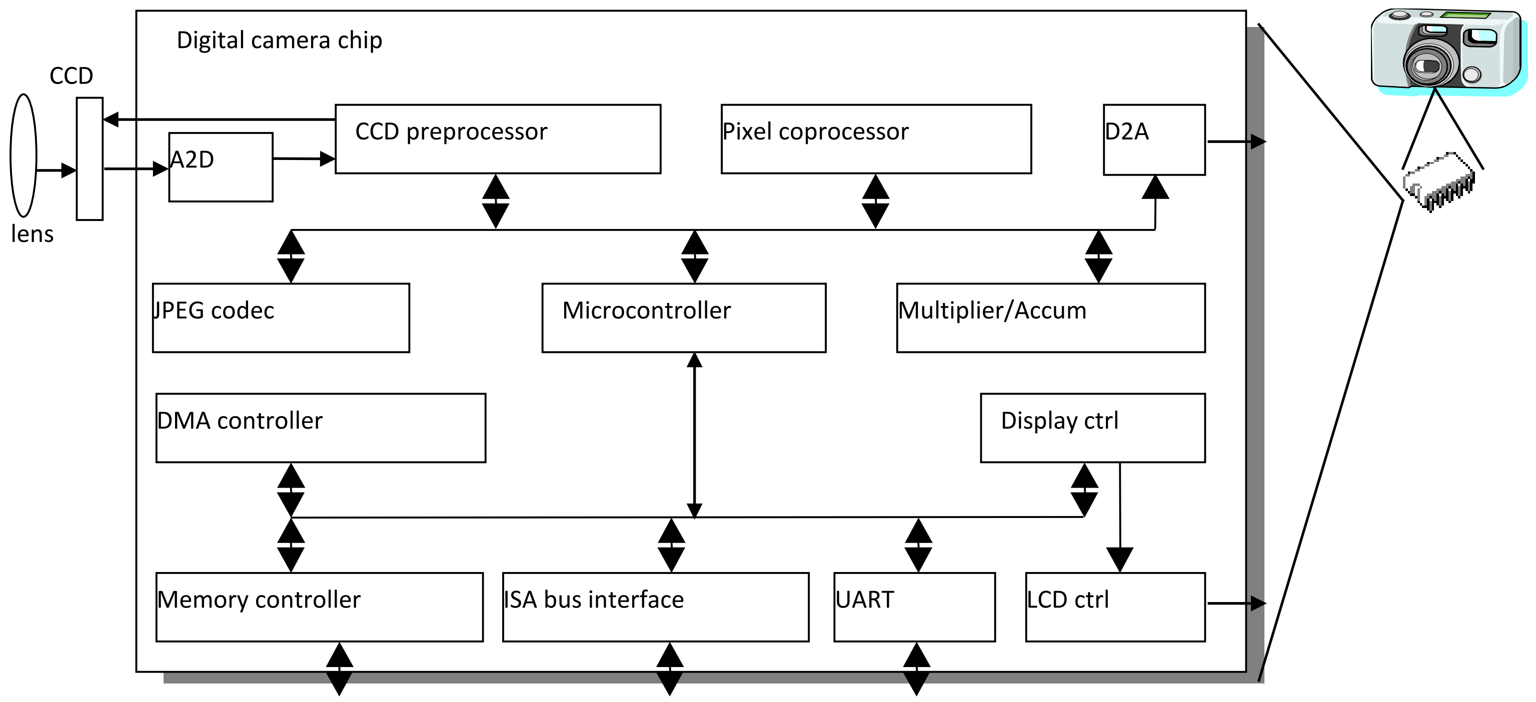
- Need domain knowledge of costs of the materials, manufacturing, etc.
- If it is not affordable users will not buy, if it is not profitable the business will fail
- Process of knowledge: cost modelling
  - Life cycle costing
  - Concept costing
  - Cost to the environment

Design for Technology: Feasibility

- Need domain knowledge of various technologies, principles from sciences
- Process knowledge: how to create ideas
  - Brainstorming
  - Stimuli from nature: shrug, tail, sneeze

Embedded systems design

An embedded system example: 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

Design challenge – optimizing design metrics

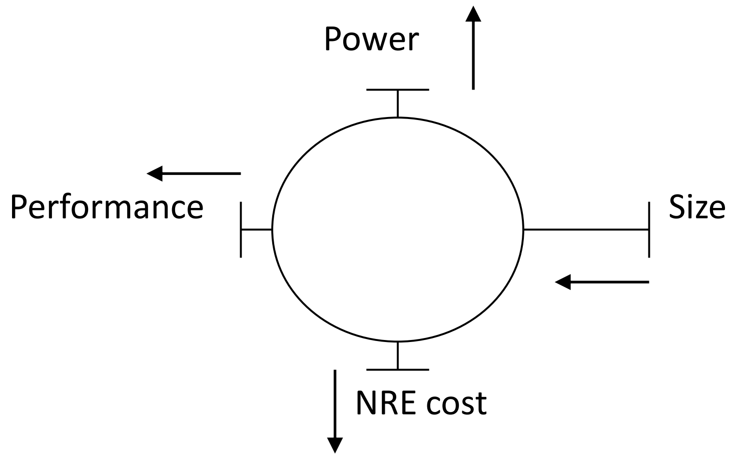
- Obvious design goal:
    - Construct an implementation with desired functionality
- Key design challenge:
    - Simultaneously optimize numerous design metrics
- Design metric
    - A measurable feature of a system's implementation
    - Optimizing design metrics is a key challenge

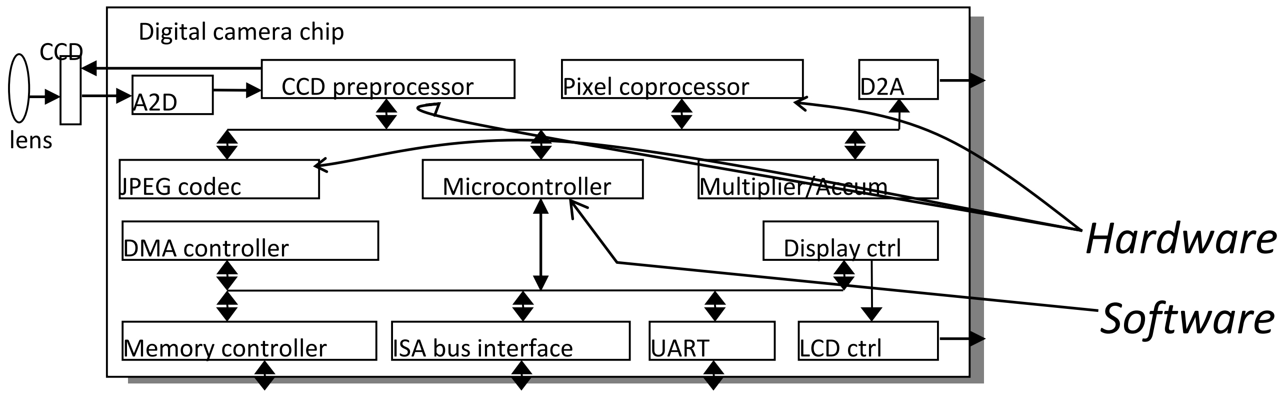
Design challenge – common metrics

- Unit cost:
  - manufacturing each copy of the system, excluding NRE cost
- NRE (Non-Recurring Engineering) cost:
  - one-time cost of designing the system
- 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
- Flexibility:
  - the ability to change the functionality of the system without incurring heavy NRE cost
- Maintainability:
  - the ability to modify the system after its initial release
- 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
- Correctness, safety, many more

Design metric competition

- Improving one may worsen others
- Expertise with both **software and hardware** is needed to optimize design metrics
- A designer must be comfortable with various technologies in order to choose the best for a given application and constraints





The performance design metric

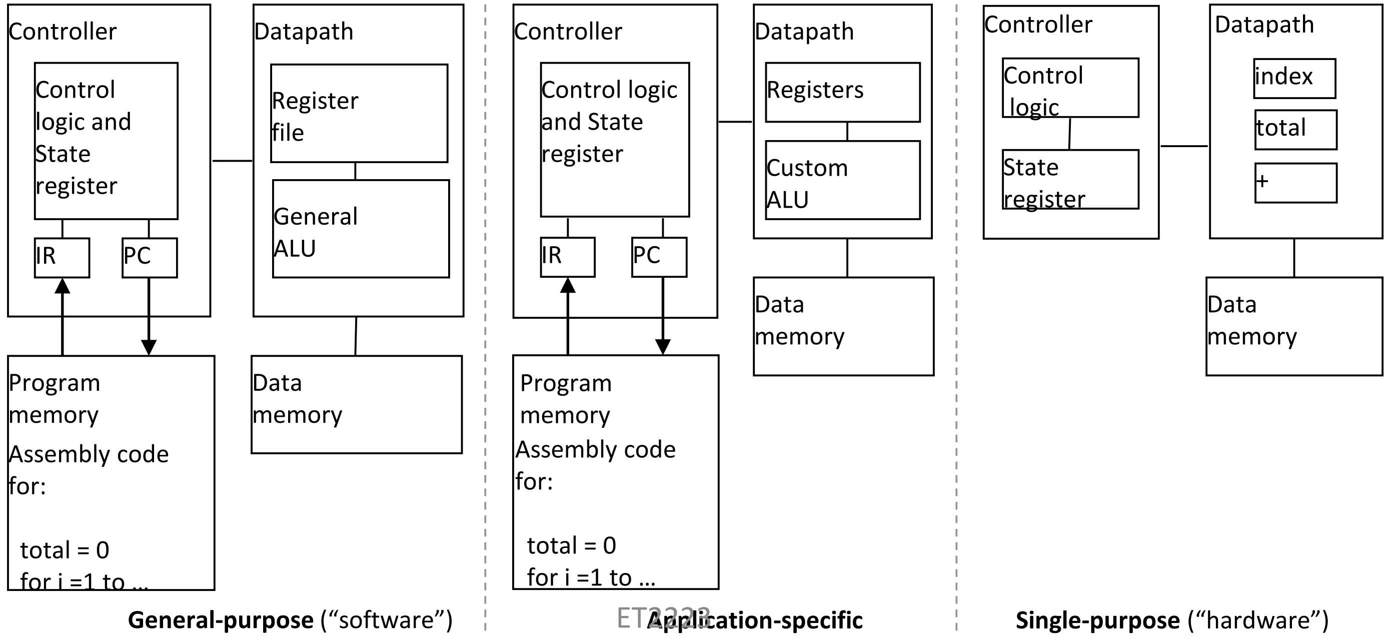
- Widely-used measure of system, widely-abused
  - Clock frequency, instructions per second – not good measures
  - Digital camera example – a user cares about how fast it processes images, not clock speed or instructions per second
- Latency (response time)
  - Time between task start and end
  - e.g., 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

Three key embedded system technologies

- Processor technology
- IC technology
- Design technology

Processor technology

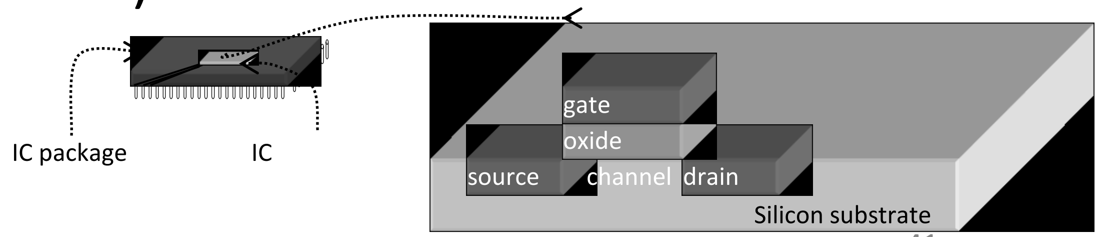
- The architecture of the computation engine used to implement a system's desired functionality
- Processor does not have to be programmable
  - "Processor" not equal to general-purpose processor



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

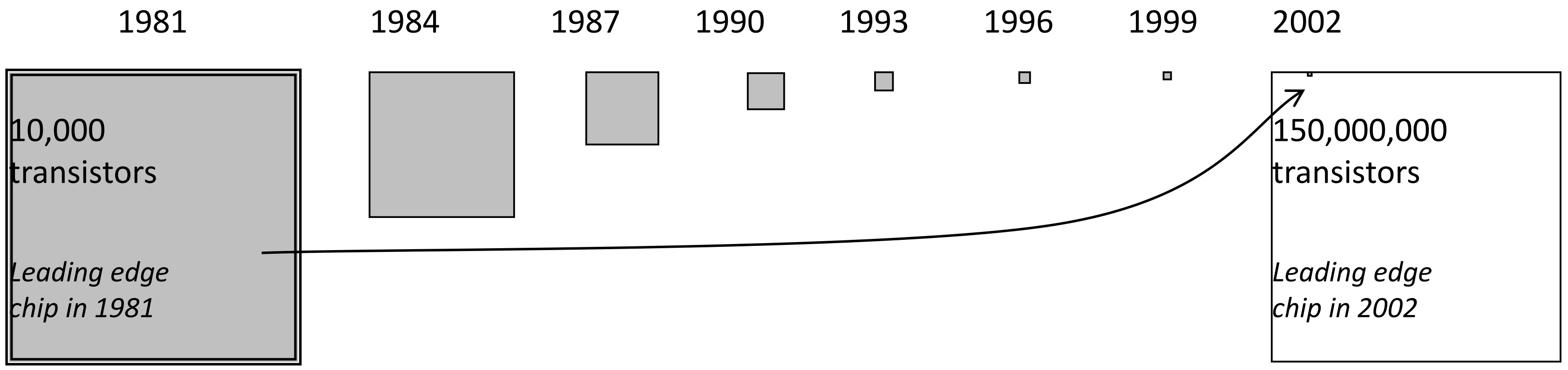
IC technology

- 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
- Three types of IC technologies
  - Full-custom/VLSI
  - Semi-custom ASIC (gate array and standard cell)
  - PLD (Programmable Logic Device)



Graphical illustration of Moore's law

- Something that doubles frequently grows more quickly than most people realize!
    - A 2002 chip can hold about 15,000 1981 chips inside itself



Typical embedded software components

Embedded Application Code

Device Drivers

A Real-Time Operating System (RTOS)

Hardware abstraction layer(s)

System initialization routines

Summary

- Embedded systems are everywhere
- Key challenge: optimization of design metrics
  - Design metrics compete with one another
- A unified view of hardware and software is necessary to improve productivity
- Three key technologies
  - Processor: general-purpose, application-specific, single-purpose
  - IC: Full-custom, semi-custom, PLD
  - Design: Compilation/synthesis, libraries/IP, test/verification