Embedded Systems

Dr. Sajesh Kumar U

Module 1 : Introduction to Embedded Systems(06 Hours)

- Complex Systems and Microprocessors

Embedding Computers, Characteristics of Embedded computing Applications, Application of Microprocessors, The Physics of Software, Challenges in Embedded Computing System, Characteristics and quality attributes of an embedded system, Performance in Embedded Computing

- The Embedded System Design Process

Requirements, Specification , Architecture Design, Designing Hardware and Software Components ,System Integration.

- Formalisms for System Design

Structural Description, Behavioral Description, An embedded system design example.

- Embedded product development cycle (EDLC)

Different phases of EDLC, EDLC models

Module 2 : Embedded system interfacing and peripherals (06 Hours)

- Communication devices

Serial Communication Standards and Devices - UART, HDLC and SPI. Serial Bus Protocols -I2C Bus, CAN Bus and USB Bus. Parallel communication standards ISA, PCI and PCI-X Bus.

- Memory

Memory devices and systems – ROM-Flash, EEPROM,RAM-SRAM, DRAM, Cache memory, memory mapping and addresses, memory management unit– DMA .

- I/O Device

Interrupts--Interrupt sources, recognizing an interrupt, ISR – Device drivers for handling ISR, Shared data problem, Interrupt latency.

Module 3 : ARM Processor fundamentals (07 Hours)

❑ ARM Processor architecture

The Acorn RISC Machine, Architectural inheritance, The ARM programmer's model, ARM development tools.

❑ ARM Assembly Language Programming

Data processing instructions, Data transfer instructions, Control flow instructions, writing simple assembly language programs.

❑ ARM Organization and Implementation

Three stage pipeline ARM organization, Five stage pipeline ARM organization, ARM instruction execution, ARM implementation, The ARM coprocessor interface.

Module 4: ARM Programming (10 Hours)

- Architectural Support for High-Level Languages

Abstraction in software design, Data types, Floating-point data types, The ARM floating-point architecture, Expressions, Conditional statements, Loops, Functions and procedures, Use of memory, Run-time environment.

- The Thumb Instruction Set

The Thumb bit in the CPSR, The Thumb programmer's model, Thumb branch instructions, Thumb software interrupt instruction, Thumb data processing instructions, Thumb single register data transfer instructions, Thumb multiple register data transfer instructions, Thumb breakpoint instruction, Thumb implementation, Thumb applications.

- Architectural Support for System Development

The ARM memory interface, The Advanced Microcontroller Bus Architecture (AMBA).

- Programming

Assembly and C language programming applications of embedded systems.

Module 5: Real Time Operating Systems (07 Hours)

- Operating system basics

Kernel, types of operating systems.

- Real time operating systems

Tasks, process, threads, multiprocessing and multi-tasking, task scheduling, types, threads and process scheduling, task communication, task synchronization, device drivers, choosing an RTOS.

Text Books

- Raj kamal, Embedded Systems Architecture, Programming and Design, TMH, 2003
- K.V. Shibu, Introduction to Embedded Systems, 2e, McGraw Hill Education India, 2016.
- Wayne Wolf, Computers as Components: Principles of Embedded Computing System Design, Morgan Kaufman Publishers - Elsevier 3ed, 2008
- Steve Furber, ARM system-on-chip architecture, Addison Wesley, Second Edition, 2000

Reference Books

- David E. Simon, An Embedded Software Primer, Pearson Education Asia, First Indian Reprint 2000.
- Steve Heath, Embedded Systems Design, Newnes – Elsevier 2ed, 2002
- Andrew N. Sloss, Dominic Symes, Chris Wright, ARM System Developer's Guide
- Designing and Optimizing System Software, Morgan Kaufmann Publishers 2004
- Frank Vahid and Tony Givargis, Embedded Systems Design – A Unified Hardware/ Software Introduction, John Wiley, 2002.
- Tammy Noergaard, Embedded Systems Architecture, A Comprehensive Guide for Engineers and Programmers, Newnes – Elsevier 2ed, 2012
- Iyer - Embedded Real time Systems, 1e, McGraw Hill Education New Delhi, 2003
- Lyla B. Das, Embedded Systems: An Integrated Approach, 1/e , Lyla B. Das, Embedded Systems, 2012

COMPLEX SYSTEMS AND MICROPROCESSORS

CHARACTERISTICS, APPLICATIONS

CHALLENGES, QUALITY ATTRIBUTES

EMBEDDED SYSTEM DESIGN PROCESS

FORMALISMS FOR SYSTEM DESIGN

EDLC

MULTI PROCESSING AND MULTI TASKING

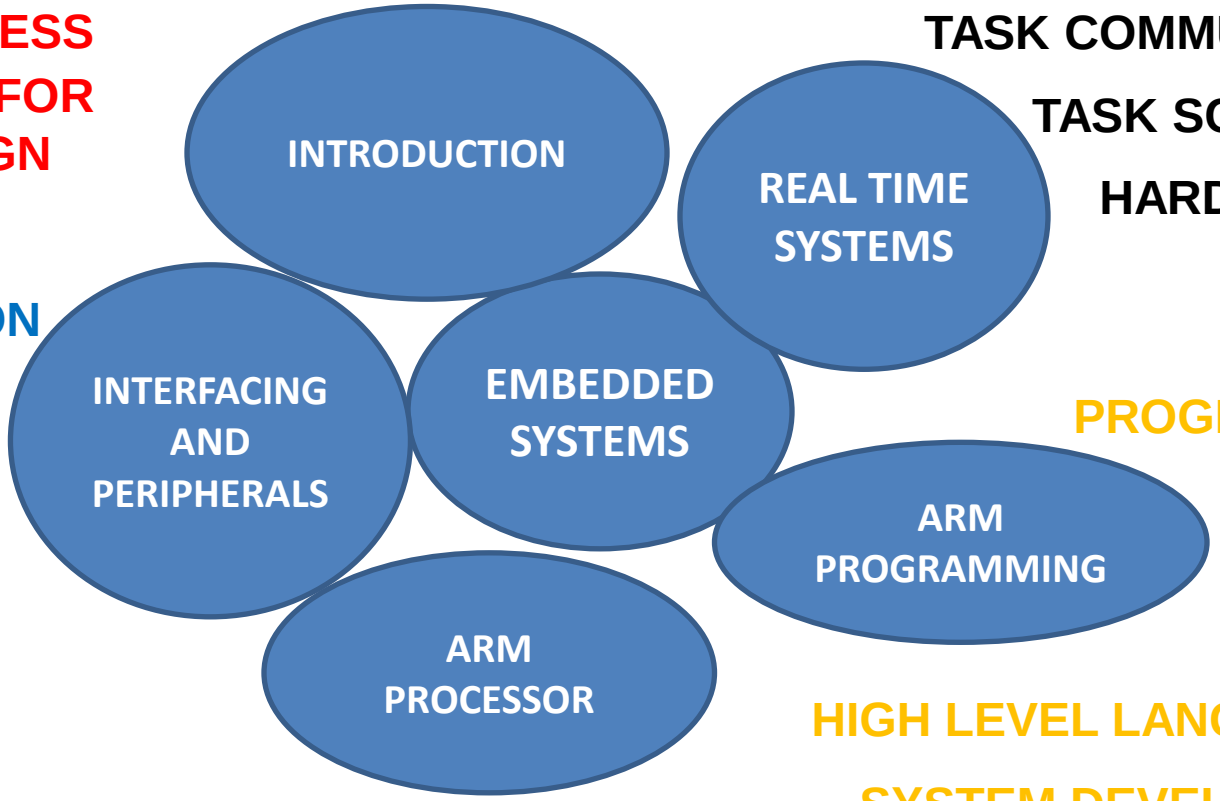
TIMING CONSTRAINTS

TASK SYNCHRONIZATION

TASK COMMUNICATION

TASK SCHEDULING

HARD & SOFT



COMMUNICATION DEVICES

MEMORY DEVICES

INPUT/OUTPUT DEVICES

INTERFACING

PROGRAMMING

HIGH LEVEL LANGUAGES

SYSTEM DEVELOPMENT

THUMB INSTRUCTION SET

ARM PROCESSOR ARCHTECTURE

ASSEMBLY LANGUAGE

ARM ORGANIZATION AND IMPLEMENTATION

Embedded System-Definition

- Any device that includes a programmable computer but is not itself intended to be a general purpose computer
- An embedded system is an electronic/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware(software).

EMBEDDED SYSTEMS

EMBEDDED SYSTEMS ARE ELECTRONIC SYSTEMS THAT EXECUTE LIMITED NUMBER OF FIXED TASKS

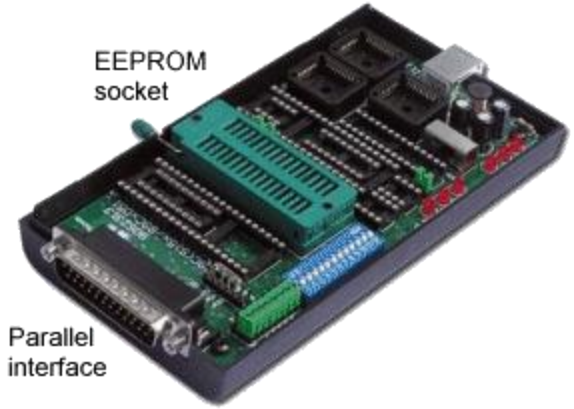
BECAUSE THE TASKS DO NOT CHANGE DURING THE LIFE SPAN OF AN EMBEDDED SYSTEM, IT IS NOT GENERAL PROGRAMMABLE IN THE WAY THAT A PERSONAL COMPUTER IS

EMBEDDED SYSTEMS ARE ELECTRONIC DEVICES THAT INCOPERATE MICROPROCESSOR WITHIN THEIR IMPLEMENTATIONS

EMBEDDED SYSTEMS CONTD…..

DO NOT HAVE DISK DRIVE SO SOFTWARES ARE OFTEN STORED IN A ROM

THIS MEANS THAT MODIFYING THE SOFTWARE REQUIRES EITHER REPLACING OR REPROGRAMMING THE ROM



An EEPROM programmer

HOW EMBEDDED SYSTEMS ARE CHARACTERIZED

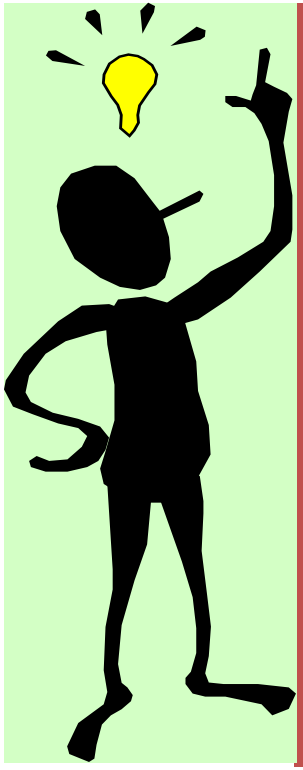
- **APPLICATION SPECIFIC: THE APPLICATION OF A EMBEDDED SYSTEM IS FIXED IN ADVANCE**
- **REACTIVE: REACTS ON EVENTS COMING FROM ITS ENVIRONMENT**
- **EFFICIENT: SMALL SIZE, LOW POWER, LOW COST, RELIABILITY**
- **Operates in harsh environments**
- **Distributed**

GENERAL CHARACTERISTICS

- **HW-SW SYSTEMS**

SOFTWARE IS USED FOR MORE FEATURES AND FLEXIBILITY

HARDWARE (PROCESSORS, ASICS, MEMORY ETC. ARE USED FOR PERFORMANCE AND SECURITY



Most of the functionality of embedded systems will be implemented in software!

Characteristics of Embedded Systems

- Complex Algorithms---Automobile Engine, Aircraft Controller
- User Interface---Moving map in a GPS controller
- Real Time---Deadline is important. System may break or gives unhappy feeling
- Multi rate—Synchronization of audio and video which runs at different rates
- Manufacturing Cost---Processor, memory, I/O devices
- Power and Energy—Larger power supply, Battery life, Heat consumption

Embedded Systems and General Purpose Computers

- A system which is a combination of a generic hardware and a General Purpose Operating System for executing a variety of applications
- Contains a General Purpose Operating System (GPOS)
- A system which is a combination of special purpose hardware and embedded OS for executing a specific set of applications
- May or may not contain an operating system for functioning

Embedded Systems and General Purpose Computers

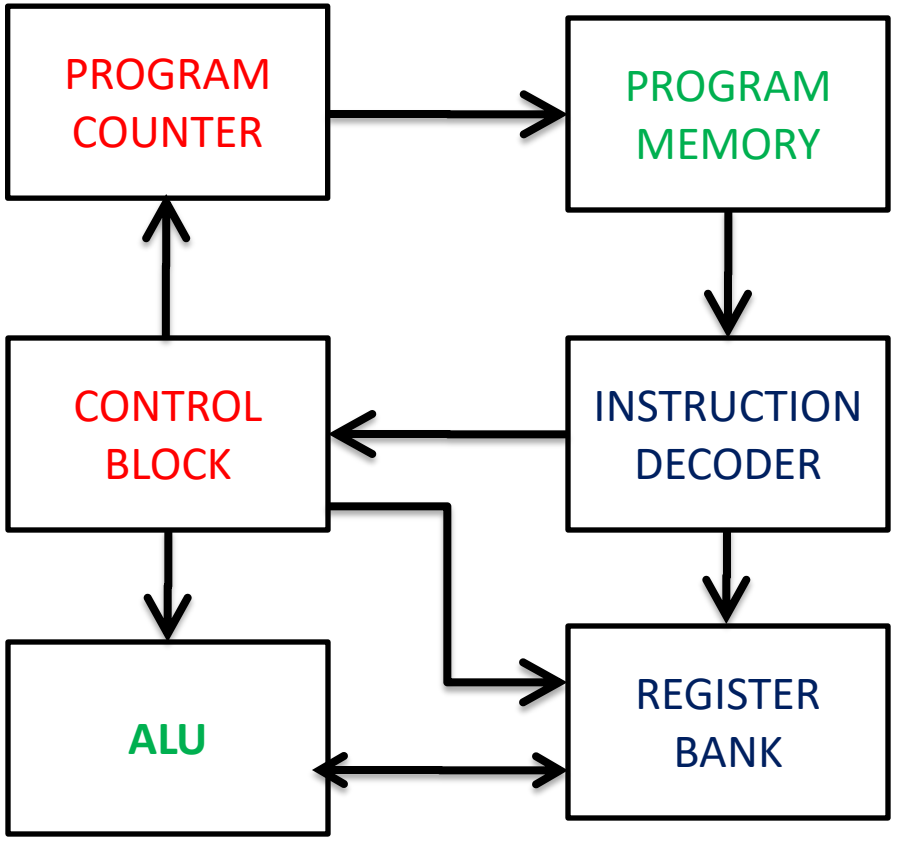
- Applications are alterable (programmable) by the user (It is possible for the end user to re-install the operating system, and also add or remove user applications)
- Performance is the key deciding factor in the selection of the system. Always, 'Faster is Better
- The firmware of the embedded system is pre-programmed and it is non-alterable by the end-user (There may be exceptions for systems supporting OS kernel image flashing through special hardware settings)
- Application-specific requirements (like performance, power requirements, memory usage, etc.) are the key deciding factors

Embedded Systems and General Purpose Computers

- Less/not at all tailored towards reduced operating power requirements, options for different levels of power management.
- Response requirements are not time-critical
- Need not be deterministic in execution behaviour
- Highly tailored to take advantage of the power saving modes supported by the hardware and the operating system
- For certain category of embedded systems like mission critical systems, the response time requirement is highly Critical
- Execution behaviour is deterministic for certain types of embedded systems like ‘Hard Real Time’ systems

DESIGN OF A MICROCONTROLLER

- PROGRAM COUNTER
- PROGRAM MEMORY
- INSTRUCTION DECODER
- CONTROL BLOCK
- ALU
- REGISTER BANK



Why Microprocessor?

- Very efficient way to implement digital systems.
- Easier to design products with various features and can be extended to provide new features.
- Predefined instruction set processor can implement design faster than custom IC design
- Parallel processing, multiprocessing increases number of instructions executed per cycle
- A special purpose logic designed for a specific function cannot be used for other functions
- Program design is separated from hardware design
- Real time, Cost and Power are important

WHY MULTIPROCESSOR

- More than one processor
- Fastest processors are costly.
- Splitting the application so that it can be performed on several processors will reduce the price
- Real time performance
- Low power
- Low cost

Objective Type Questions

(1) Embedded systems are
(a)General purpose (b) Special purpose

(2) Embedded system is
(a) An electronic system (b) A pure mechanical system
(c) An electro-mechanical system (d) (a) or (c)

(3) Which of the following is not true about embedded systems?
(a) Built around specialized hardware (b) Always contain an operating system (c) Execution behaviour may be deterministic (d) All of these (e) None of these

Objective Type Questions

(4) Which of the following is not an example of a 'Small-scale Embedded System'?
(a) Electronic Barbie doll (b) Simple calculator
(c) Cell phone (d) Electronic toy car

(5) The first recognized modern embedded system is
(a) Apple Computer (b) Apollo Guidance Computer (AGC) (c) Calculator (d) Radio Navigation System

Objective Type Questions

(6) The first mass produced embedded system is

(a) Minuteman-I (b) Minuteman-II (c) Autonetics D-17 (d) Apollo Guidance Computer (AGC)

(7) Which of the following is are) an intended purpose(s) of embedded systems?

(a) Data collection (b) Data processing (c) Data communication (d) All of these (e) None of these

(8) Which of the following is an (are) example(s) of embedded system for data communication?

(a) USB Mass storage device (b) Network router (c) Digital camera (d) Music player (e) All of these (f) None of these

Objective Type Questions

(9) A digital multimeter is an example of an embedded system for

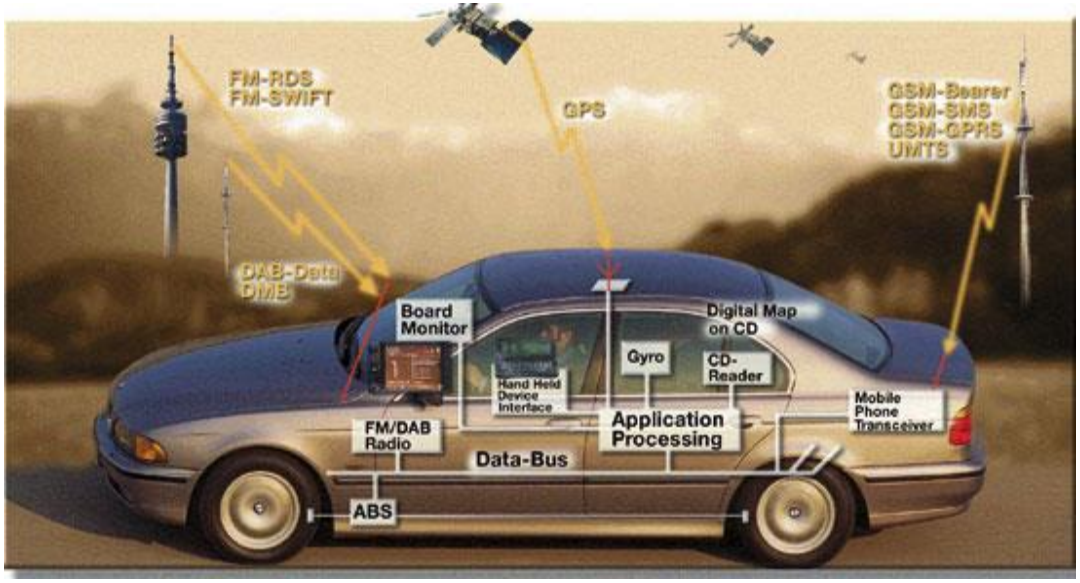
(a) Data communication (b) Monitoring (c) Control (d) All of these (e) None of these

(10) Which of the following is an (are) example(s) of an embedded system for signal processing?

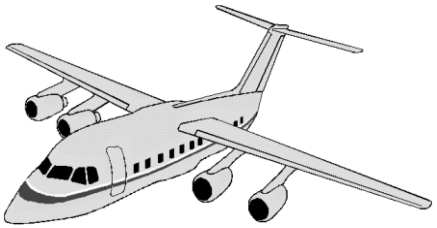
(a) Apple iPOD (media player device) (b) SanDisk USB mass storage device (c) Both (a) and (b) (d) None of these

APPLICATION AREAS

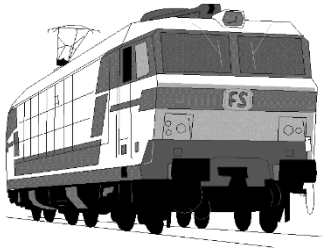
Automotive electronics



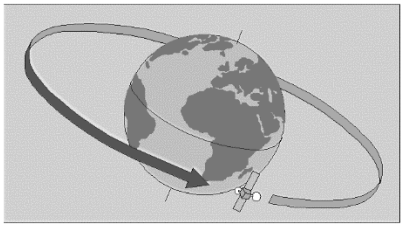
Aircraft electronics



Trains

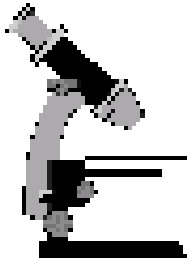


Telecommunication

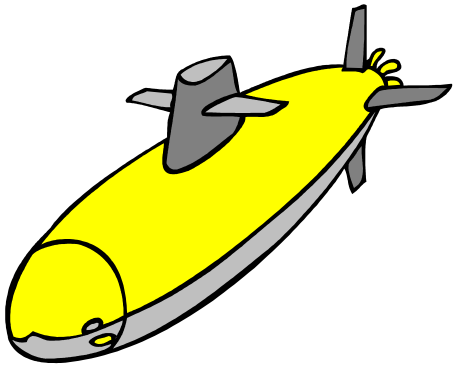


APPLICATION AREAS

- Medical systems



- Military applications



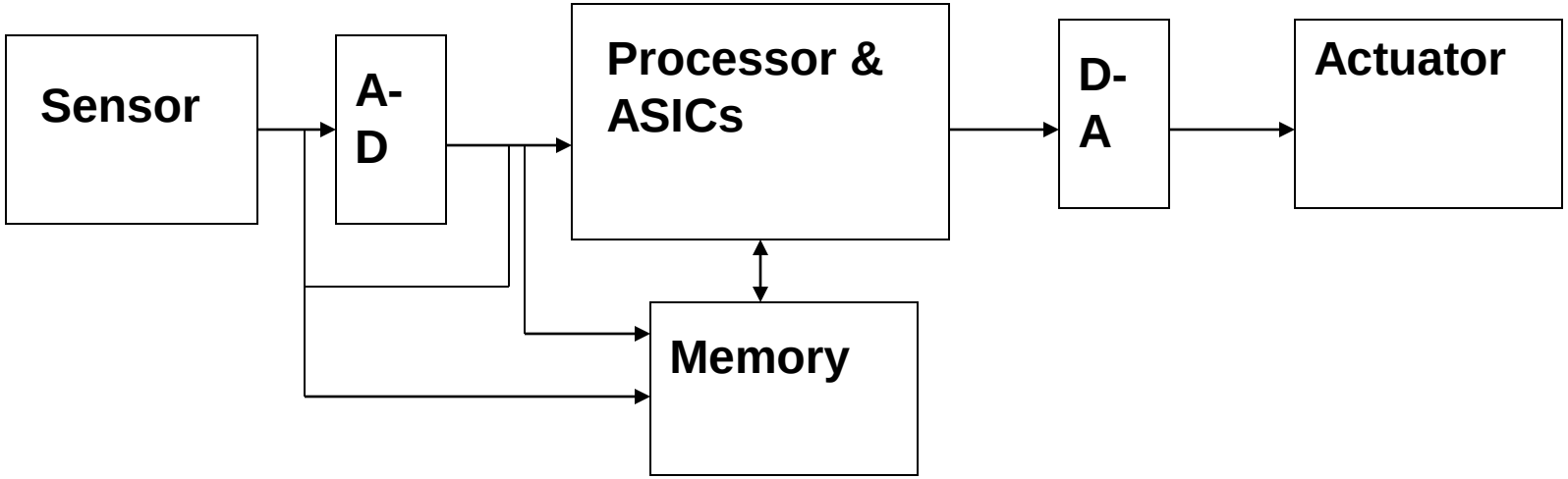
- Authentication

- Consumer electronics

- Fabrication equipment

- Smart buildings

ESSENTIAL COMPONENTS OF AN EMBEDDED SYSTEM



Microprocessor
Sensors
Converters (A-D and D-A)
Actuators
Memory (On-chip and Off chip)
Communication path with the interacting environment

Quality Attributes

Quality Attributes are non functional requirements

Operational Quality Attributes

- Response
- Throughput
- Reliability
- Maintainability
- Security
- Safety

Quality Attributes

Non-Operational Quality Attributes

- Testability, Debug-ability
- Evolvability
- Portability
- Time to Prototype and Market
- Per unit and Total Cost

CHALLENGES OF EMBEDDED SYSTEM DESIGN

- ***How much hardware do we need?***
- ***How do we meet deadlines?***
- ***How do we minimize power consumption?***
- ***How do we design for upgradability?***
- ***Does it really work?***

1. Complex testing:

2. Limited observability and controllability

3. Restricted development environments

CHALLENGES OF EMBEDDED SYSTEM DESIGN

- **Clock rate reduction : Power dissipation proportional to clock frequency**
- **Voltage reduction : Power reduces as square of voltage**

RISC vs CISC

- Lesser number of instructions
- Instruction pipelining and increased execution speed
- Orthogonal instruction set (Allows each instruction to operate on any register and use any addressing mode)

- Greater number of Instruction
- Generally no instruction pipelining feature
- Non-orthogonal instruction set (All instructions are not allowed to operate on any register and use any addressing mode. It is instruction-specific)

RISC vs CISC

- Operations are performed on registers only, the only memory operations are load and store
- A large number of registers are available
- Programmer needs to write more code to execute a task since the instructions are simpler ones

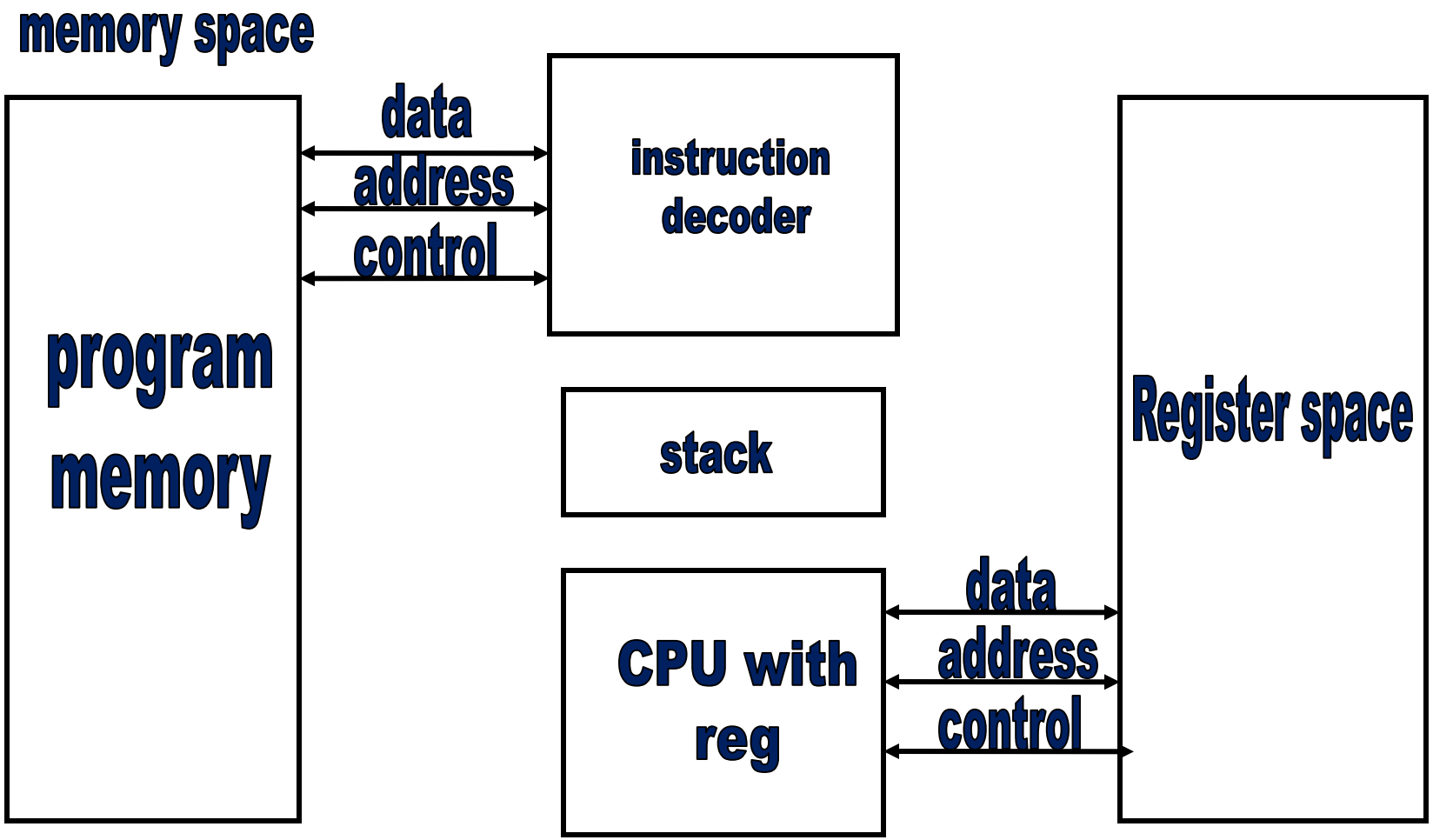
- Operations are performed on registers or memory depending on the instruction
- Limited number of general purpose registers
- Instructions are like macros in C language. A programmer can achieve the desired functionality with a single instruction which in turn provides the effect of using more simpler single instructions in RISC

RISC vs CISC

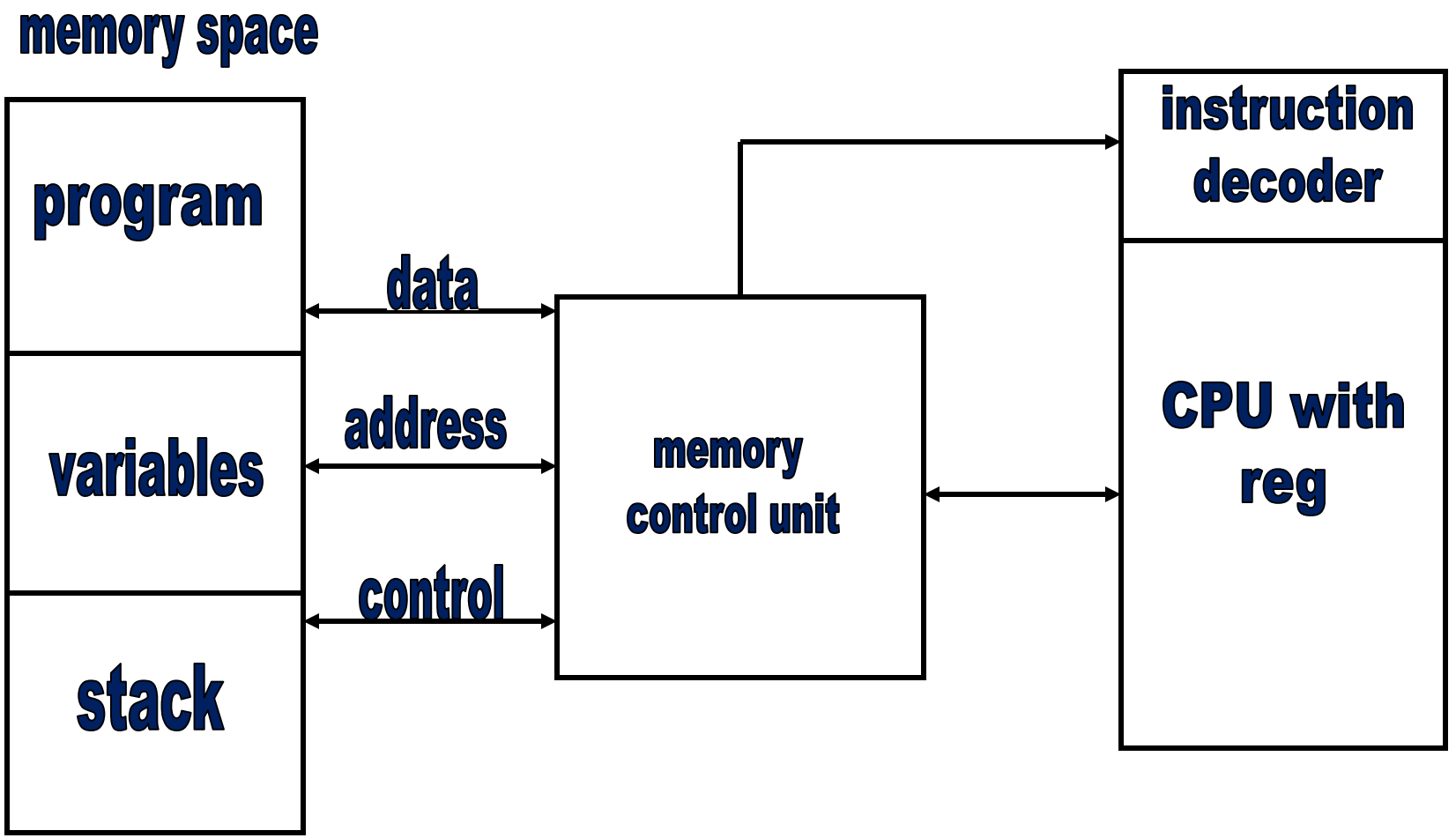
- Single, fixed length instructions
- Less silicon usage and pin count
- With Harvard Architecture

- Variable length instructions
- More silicon usage since more additional decoder logic is required to implement the complex instruction decoding
- Can be Harvard or Von-Neumann Architecture

HARVARD ARCHITECTURE



VON NEUMANN ARCHITECTURE



Harvard architecture vs Von Neumann Architecture

- Separate buses for instruction and data fetching
- Easier to pipeline, so high performance can be achieved
- Comparatively high cost

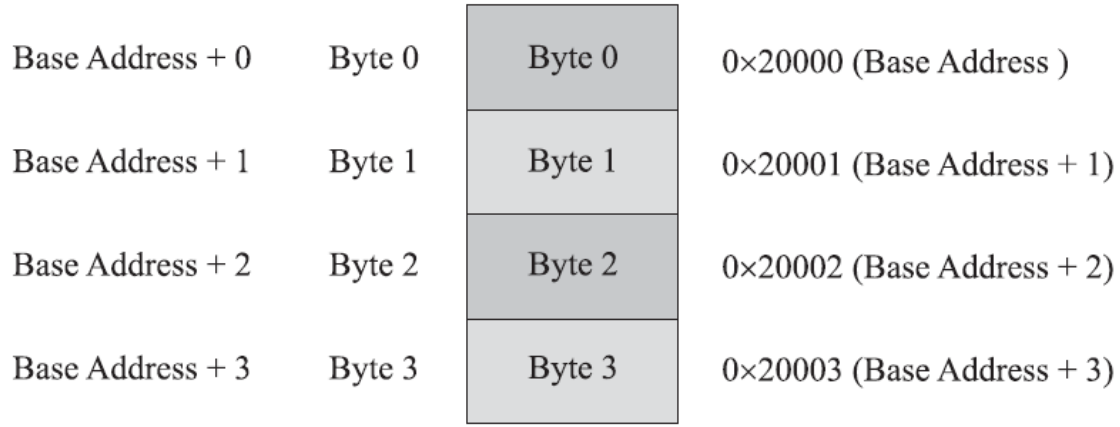
- Single shared bus for instruction and data fetching
- Low performance compared to Harvard architecture
- Cheaper

Harvard architecture vs Von Neumann Architecture

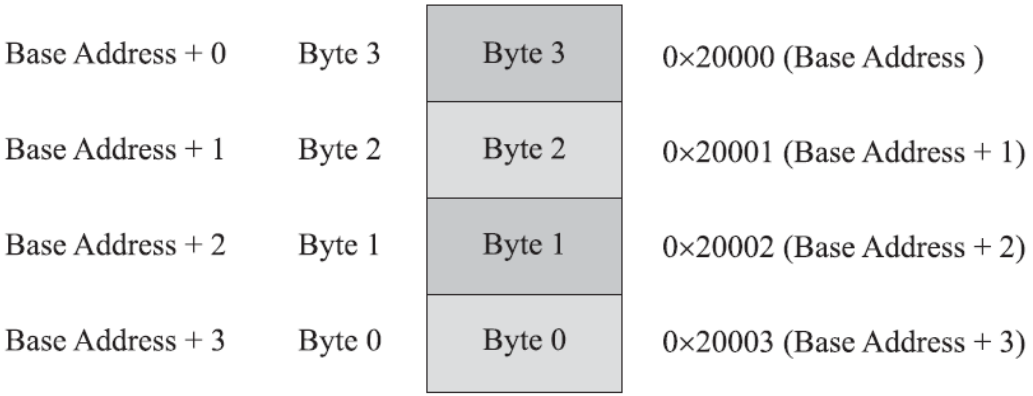
- No memory alignment problems
- Since data memory and program memory are stored physically in different locations, no chances for accidental corruption of program memory

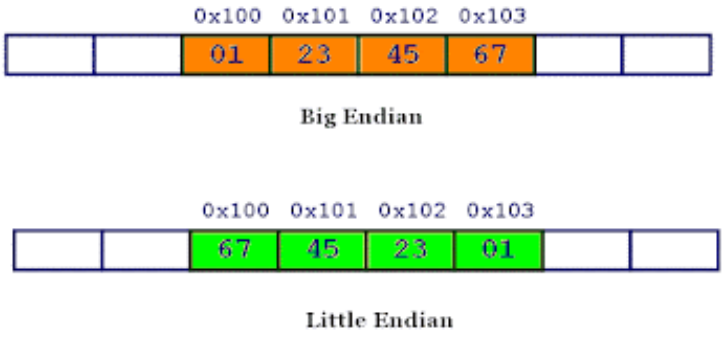
- Allows self modifying codes
- Since data memory and program memory are stored physically in the same chip, chances for accidental corruption of program memory

Big-Endian vs. Little-Endian Processors



Data **0x01234567** is stored as





(1) Embedded systems are application and domain specific. State True or False

(a)True (b) False

(2) Which of the following is true about Embedded Systems?

(a) Reactive and Real Time (b) Distributed (c) Operates in harsh environment (d) All of these (e) None of these

(3) Which of the following is a distributed embedded system?

(a) Cell phone (b) Notebook Computer (c) SCADA system (d) All of these (e) None of these

(4) Quality attributes of an embedded system are

(a) Functional requirements (b) Non-functional requirements

(c) Both (d) None of these

(5) Response is a measure of

(a) Quickness of the system (b) How fast the system tracks changes in Input (c) Both (d) None of these

(6) Throughput of an embedded system is a measure of

(a) The efficiency of the system (b) The output over a stated period of time (c) Both (d) None of these

(7) Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR) defines the reliability of an embedded system. State True or False

(a) True (b) False

(8) Which of the following are the three measures of information security in embedded systems?
(a) Confidentiality, secrecy, integrity
(b) Confidentiality, integrity, availability
(c) Confidentiality, transparency, availability
(d) Integrity, transparency, availability

(9) You are working on a mission critical embedded system development project for a client and the client and your company has signed a Non Disclosure Agreement (NDA) on the disclosure of the project-related information. You share the details of the project you are working with your friend. Which aspect of Information security you are violating here?
(a)Integrity (b) Confidentiality (c) Availability (d) None of these

(10) Which of the following is an example of ‘gradual’ safety threat from an embedded system?
(a) Product blast due to overheating of the battery
(b) UV emission from the embedded product
(c) Both of these
(d) None of these
(11) Which of the following is (are) an operational quality attribute?
(a) Testability (b) Safety (c) Debug-ability (d) Portability
(e) All of these
(12) Which of the following is (are) non-operational quality attribute?
(a) Reliability (b) Safety (c) Maintainability (d) Portability
(e) All of these (f) None of these

(13) The Mean Time Between Failure (MTBF) of an embedded product is 4 months and the Mean Time To Repair (MTTR) of the product is 2 weeks. What is the availability of the product?

(a) 100% (b) 50% (c) 89% (d) 10%

(14) In the Information security context, Confidentiality deals with the protection of data and application from unauthorized disclosure. State True or False

(a)True (b) False

(15) What are the two different aspects of debug-ability in the embedded system development context?

(a) Hardware & Firmware debug-ability (b) Firmware & Software debug-ability (c) None of these

(16) For an embedded system, the quality attribute 'Evolvability' refers to

(a) The upgradability of the product (b) The modifiability of the product (c) Both of these (d) None of these

(17) Portability is a measure of ‘system independence’. State True or False

(a) True (b) False

(18) For a commercial embedded product the unit cost is high during

(a) Product launching (b) Product maturity

(c) Product growth (d) Product discontinuing

(19) For a commercial embedded product the sales volume is high during

(a) Product launching (b) Product maturity

(c) Product growth (d) Product discontinuing

(20) An integer variable with value 255 is stored in memory location at 0x8000. The processor word length is 8 bits and the processor is a big endian processor. The size of integer is considered as 4 bytes in the system. What is the value held by the memory location 0x8000?

(a)0xFF (b) 0x00 (c) 0x01 (d) None of these

(21) The instruction set of RISC processor is

(a) Simple and lesser in number (b) Complex and lesser in number

(c) Simple and larger in number (d) Complex and larger in number

(22) Which of the following is true about CISC processors?

(a) The instruction set is non-orthogonal

(b) The number of general purpose registers is limited

(c) Instructions are like macros in C language

(d) Variable length Instructions

(e) All of these

(f) None of these

23. Which of the following processor architecture supports easier instruction pipelining?
(a) Harvard (b) Von Neumann
(c) Both of them (d) None of these

24. Microprocessors/controllers based on the Harvard architecture will have separate data bus and instruction bus. This allows the data transfer and program fetching to occur simultaneously on both buses. State True or False
(a) True (b) False

DESIGN METRICS

Power dissipation
Performance
Process deadlines
User interfaces
Size
Engineering cost
Manufacturing cost
Flexibility
Prototype
Development time
Time to market
System and user safety
Maintenance

STEPS IN A DESIGN PROCESS

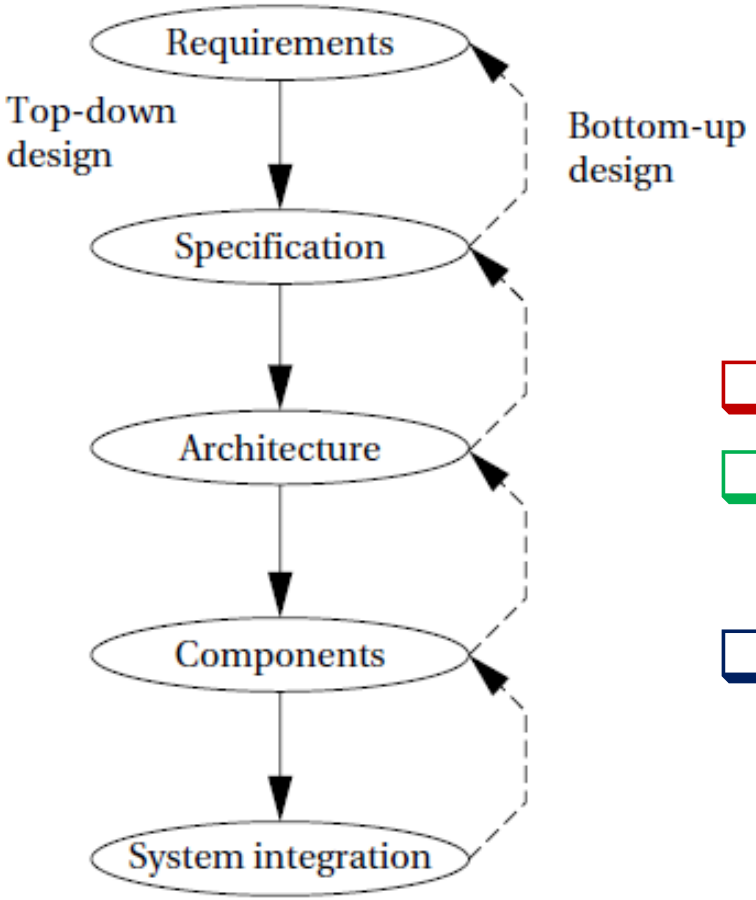
1. **BOTTOM UP DESIGN –BUILDS BY STARTING FROM THE COMPONENTS**

2. **TOP DOWN DESIGN – STARTS FROM THE ABSTRACTION OF THE PROCESS AND THEN DETAILS ARE CREATED**

FIVE LEVEL OF ABSTRACTIONS

1. REQUIREMENTS
2. SPECIFICATIONS
3. ARCHITECTURE
4. COMPONENTS
5. SYSTEM INTEGRATION

Embedded System Design Process



- **Design steps**
- **Design Methodology**

- Performance
- Breaking the process into managing steps
- Communication between team members

Requirements

- **Functional**
- **Non Functional**
- Name
- Purpose
- Inputs
- Outputs
- Functions
- Performance
- Manufacturing cost
- Power
- Physical Size and Weight

SYSTEM DESIGN EXAMPLE REQUIREMENTS

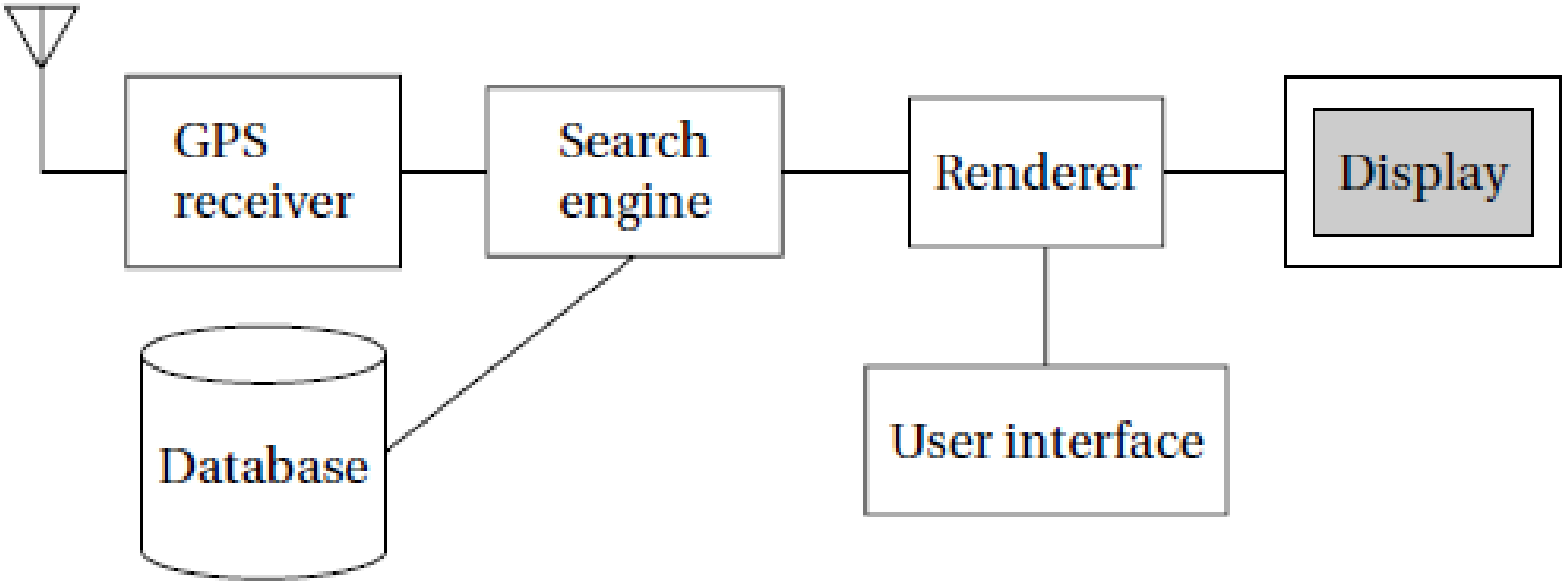
Name	GPS moving map
Purpose	Consumer-grade moving map for driving use
Inputs	Power button, two control buttons
Outputs	Back-lit LCD display 400 × 600
Functions	Uses 5-receiver GPS system; three user-selectable resolutions; always displays current latitude and longitude
Performance	Updates screen within 0.25 seconds upon movement
Manufacturing cost	$30
Power	100 mW
Physical size and weight	No more than 2" × 6, " 12 ounces

SPECIFICATION

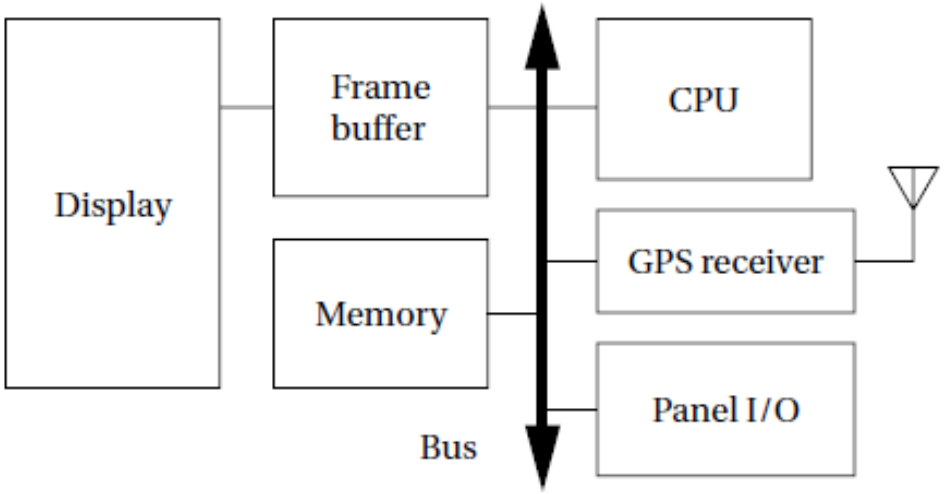
A specification of the GPS system would include several components:

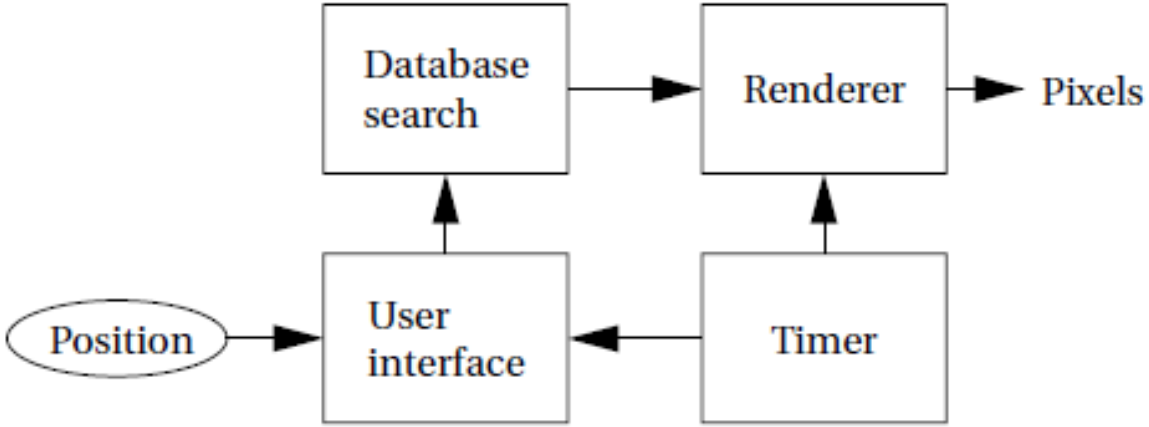
- Data received from the GPS satellite constellation.
- Map data.
- User interface.
- Operations that must be performed to satisfy customer requests.
- Background actions required to keep the system running, such as operating the GPS receiver.

ARCHITECTURE



HARDWARE & SOFTWARE





EXAMPLES

1. AUTOMATIC CHOCOLATE VENDING MACHINE
2. SMART CARD
3. DIGITAL CAMERA
4. MODEL TRAIN CONTROLLER

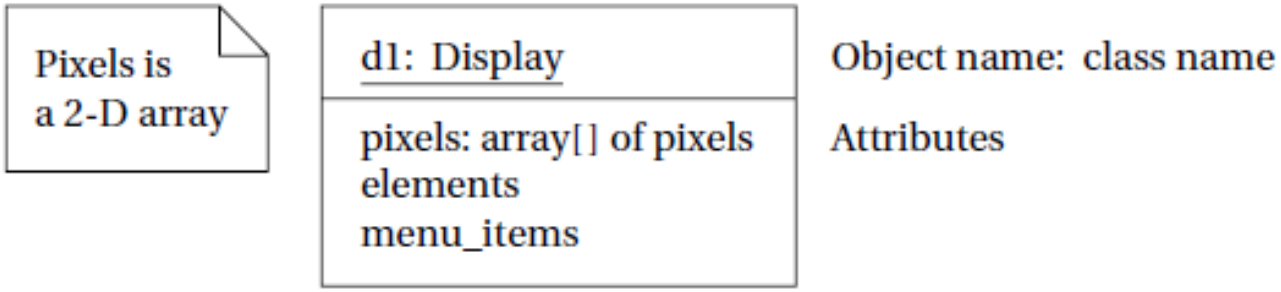
FORMALISMS OF SYSTEM DESIGN

UNIFIED MODELING LANGUAGE (UML) – Object oriented modeling language

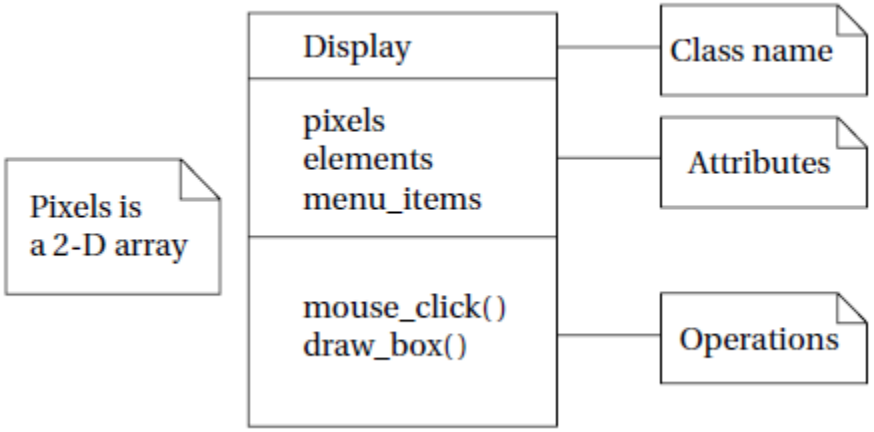
- Object-oriented specification allows a system to be described in a way that closely models real-world objects and their interactions

- Object-oriented specification provides a basic set of primitives that can be used to describe systems with particular attributes, irrespective of the relationships of those systems' components to real-world objects.

Structural Description



OBJECT



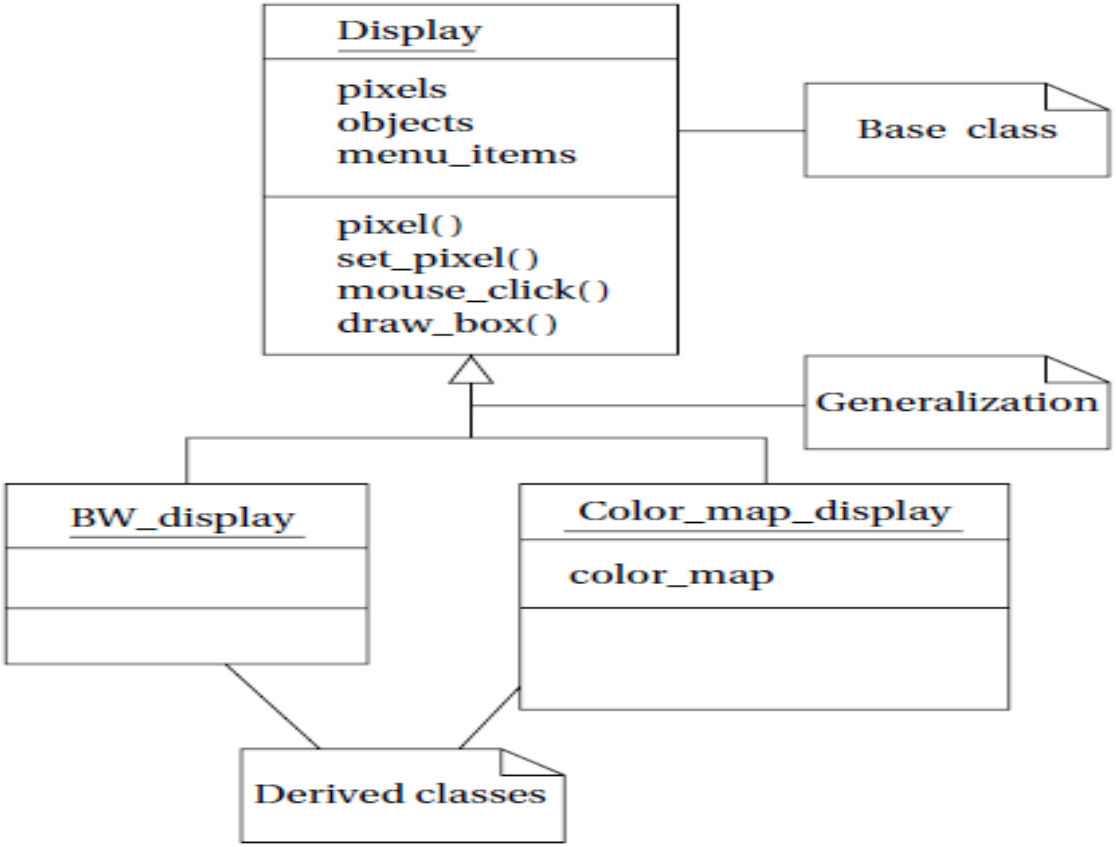
CLASS

A class defines both the ***interface*** for a particular type of object and that object's ***implementation***

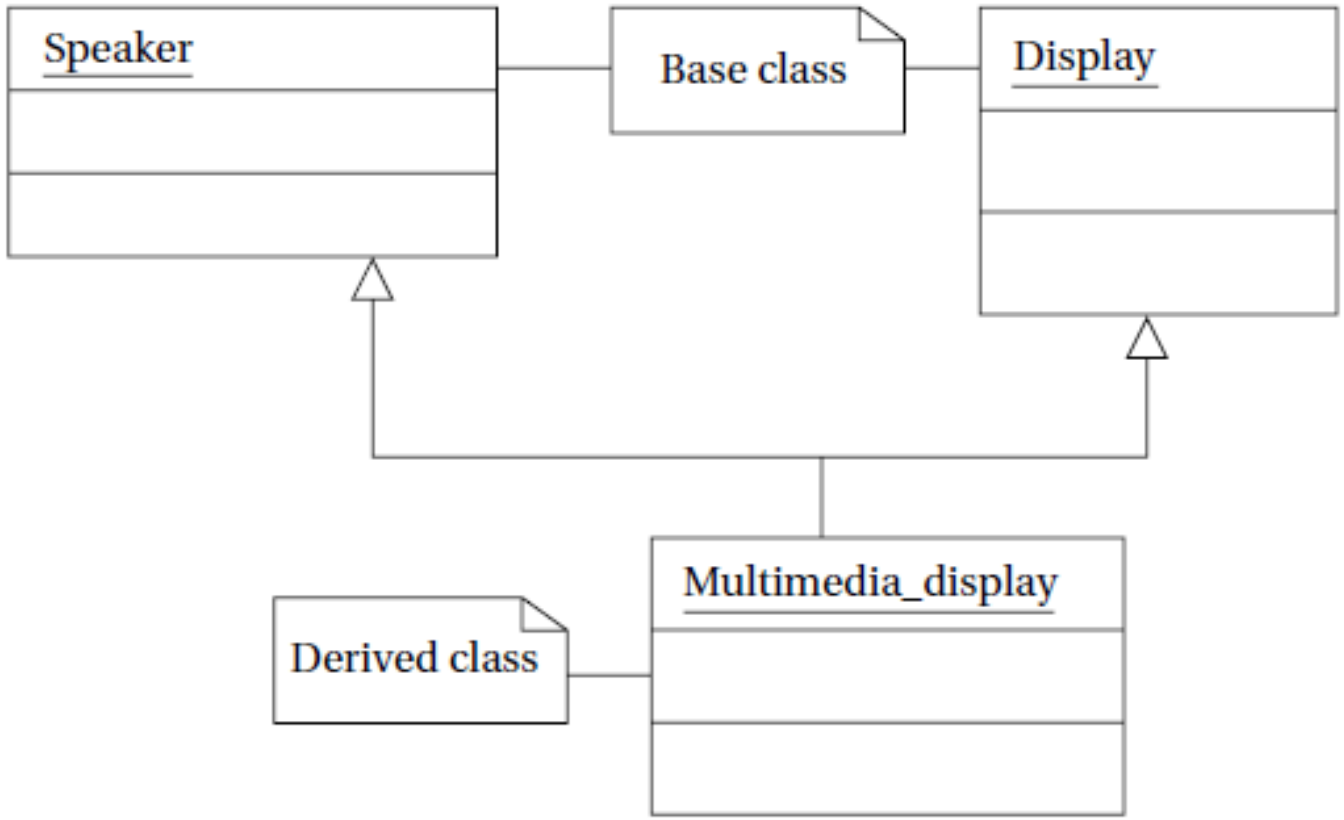
Relationship-Classes and Objects

- ***Association*** occurs between objects that communicate with each other but have no ownership relationship between them.
- ***Aggregation*** describes a complex object made of smaller objects.
- ***Composition*** is a type of aggregation in which the owner does not allow access to the component objects.
- ***Generalization*** allows us to define one class in terms of another.

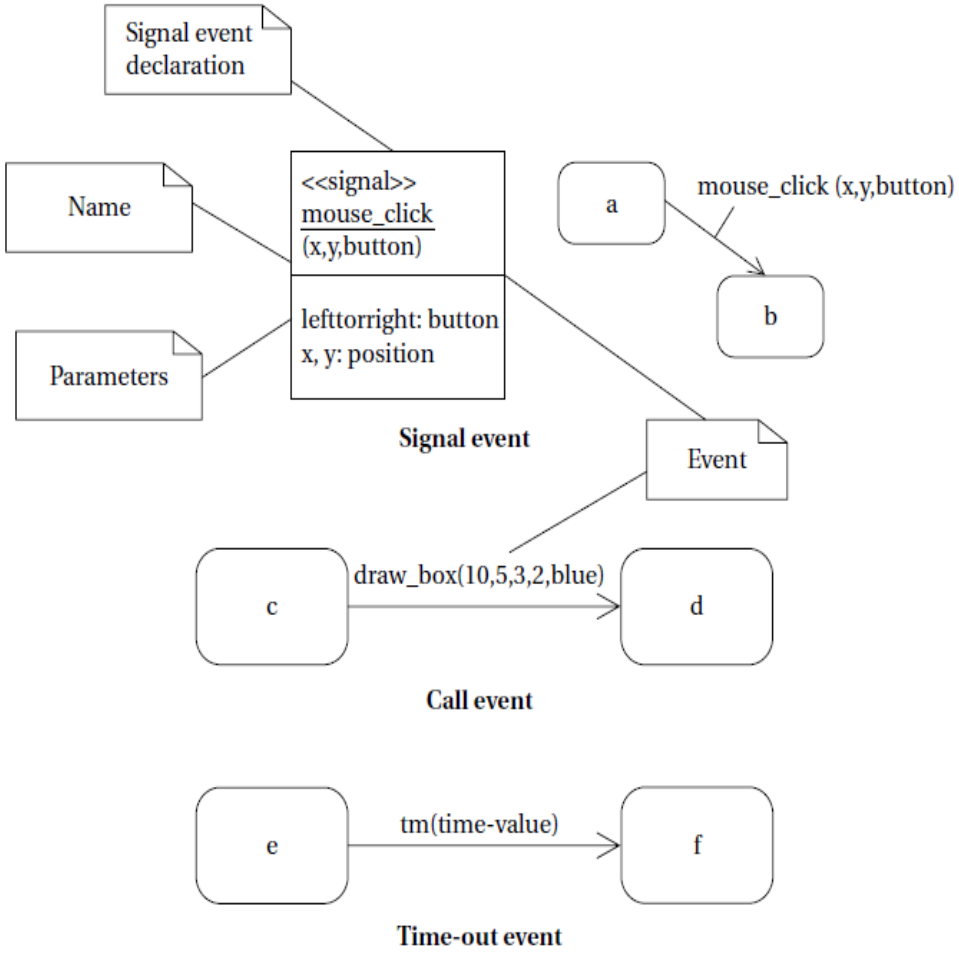
Derived Classes

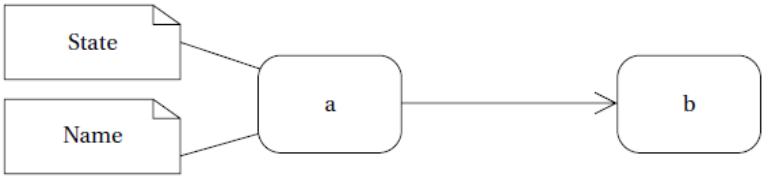


MULTIPLE INHERITANCE FROM BASE CLASS

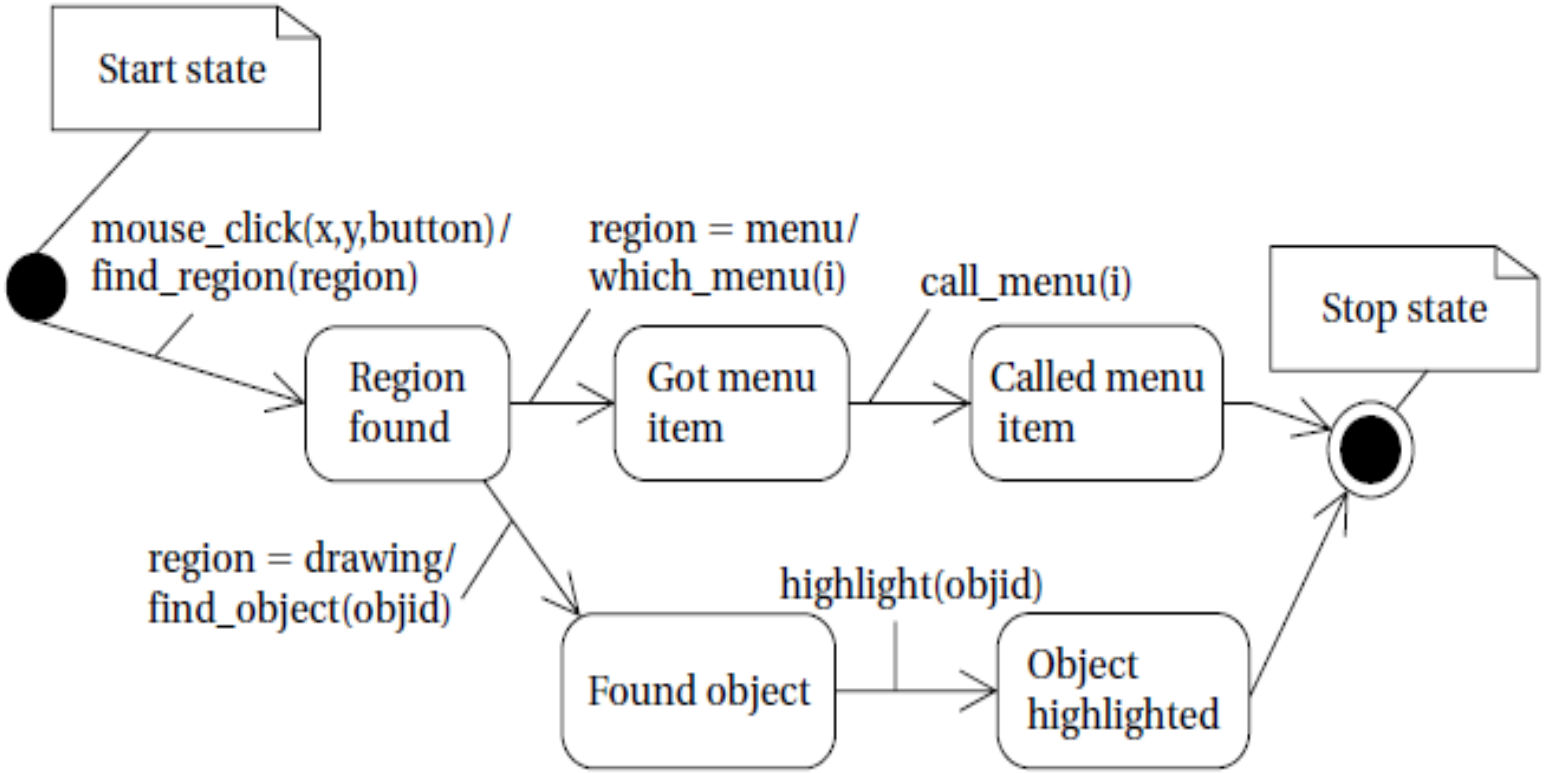


BEHAVIORAL DESCRIPTION – STATE MACHINE

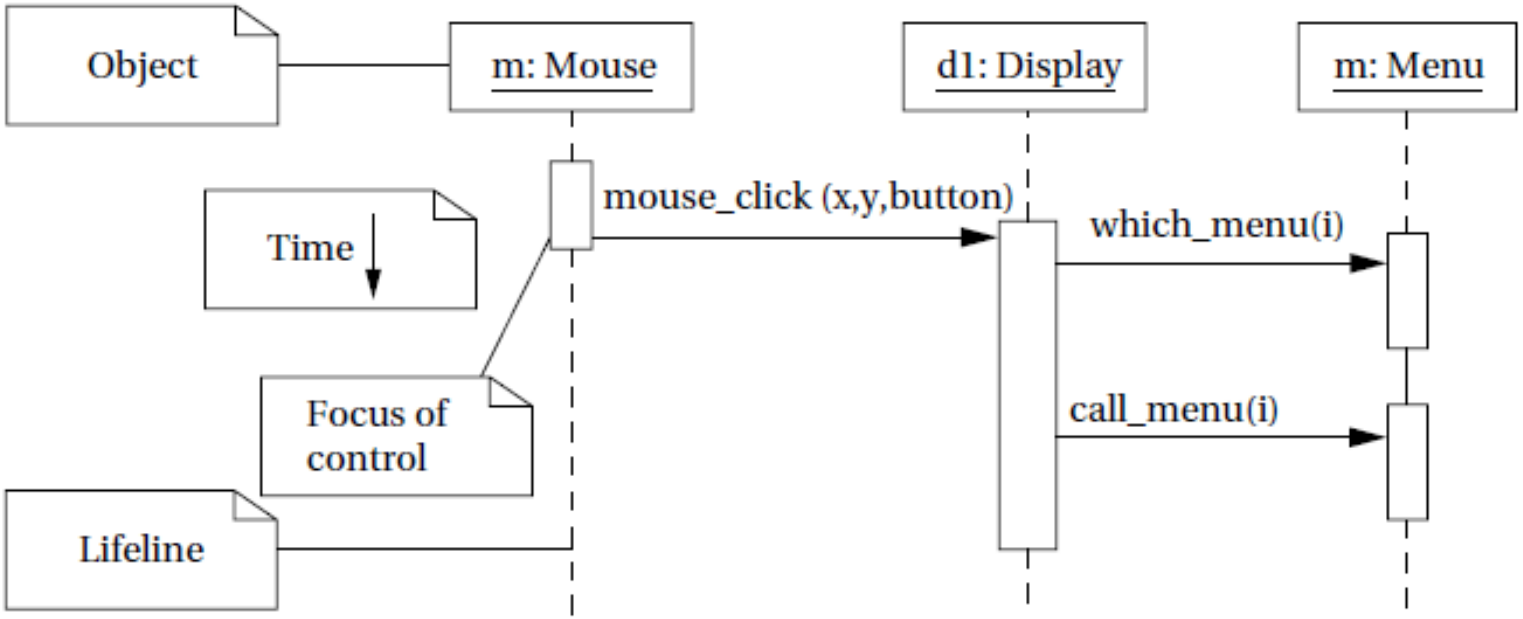




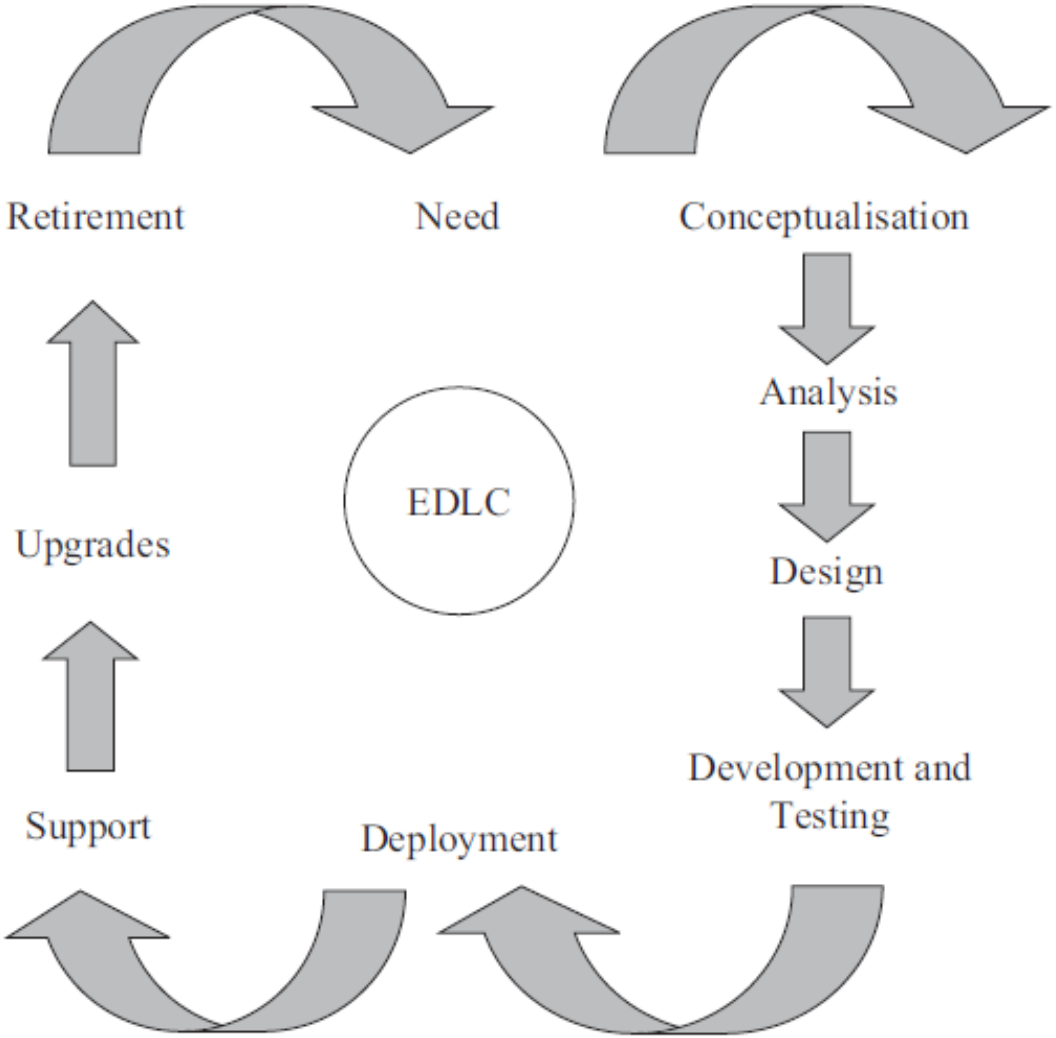
BEHAVIORAL DESCRIPTION – STATE MACHINE



SEQUENCE DIAGRAM

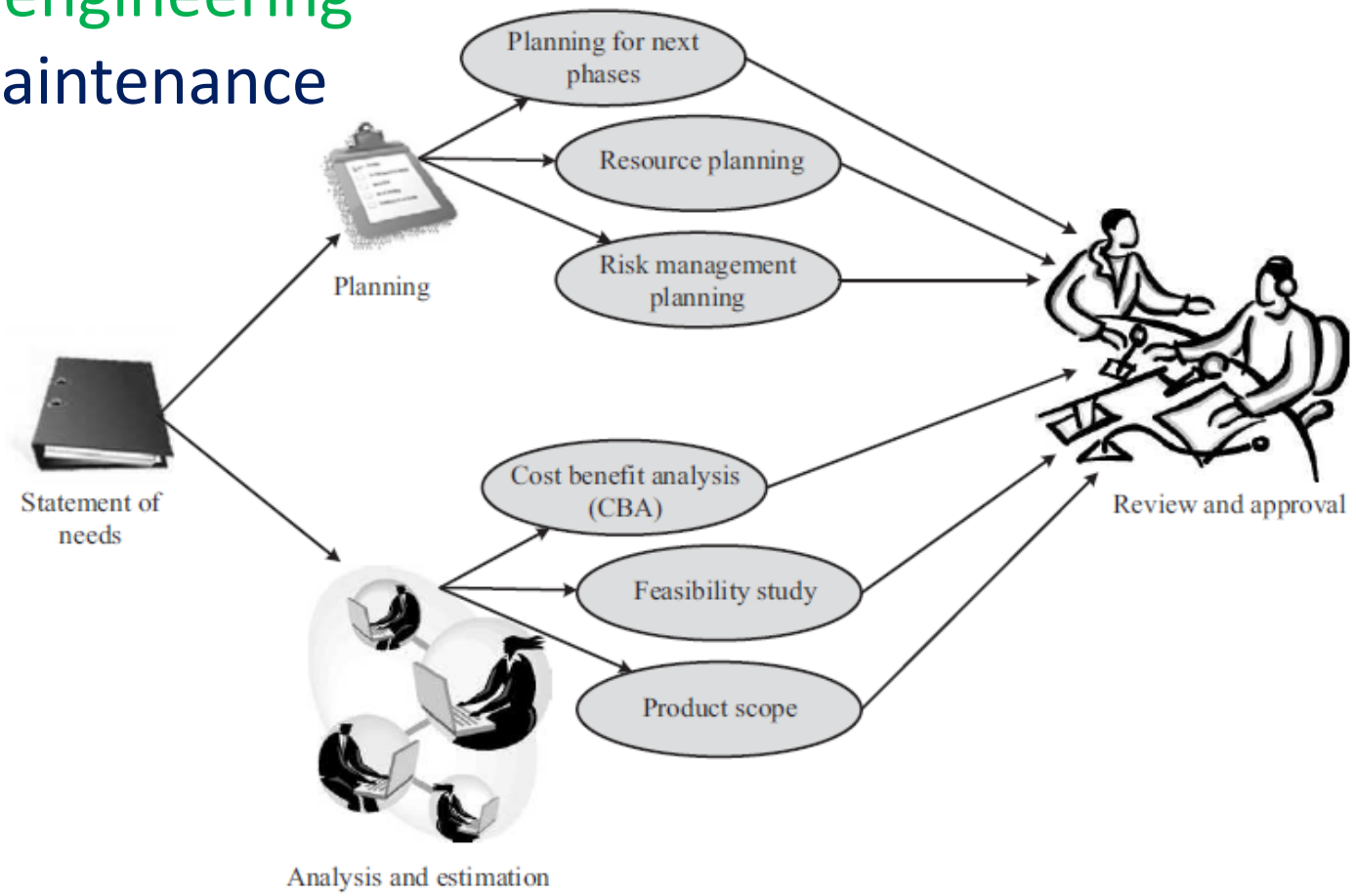


Embedded Development Life Cycle

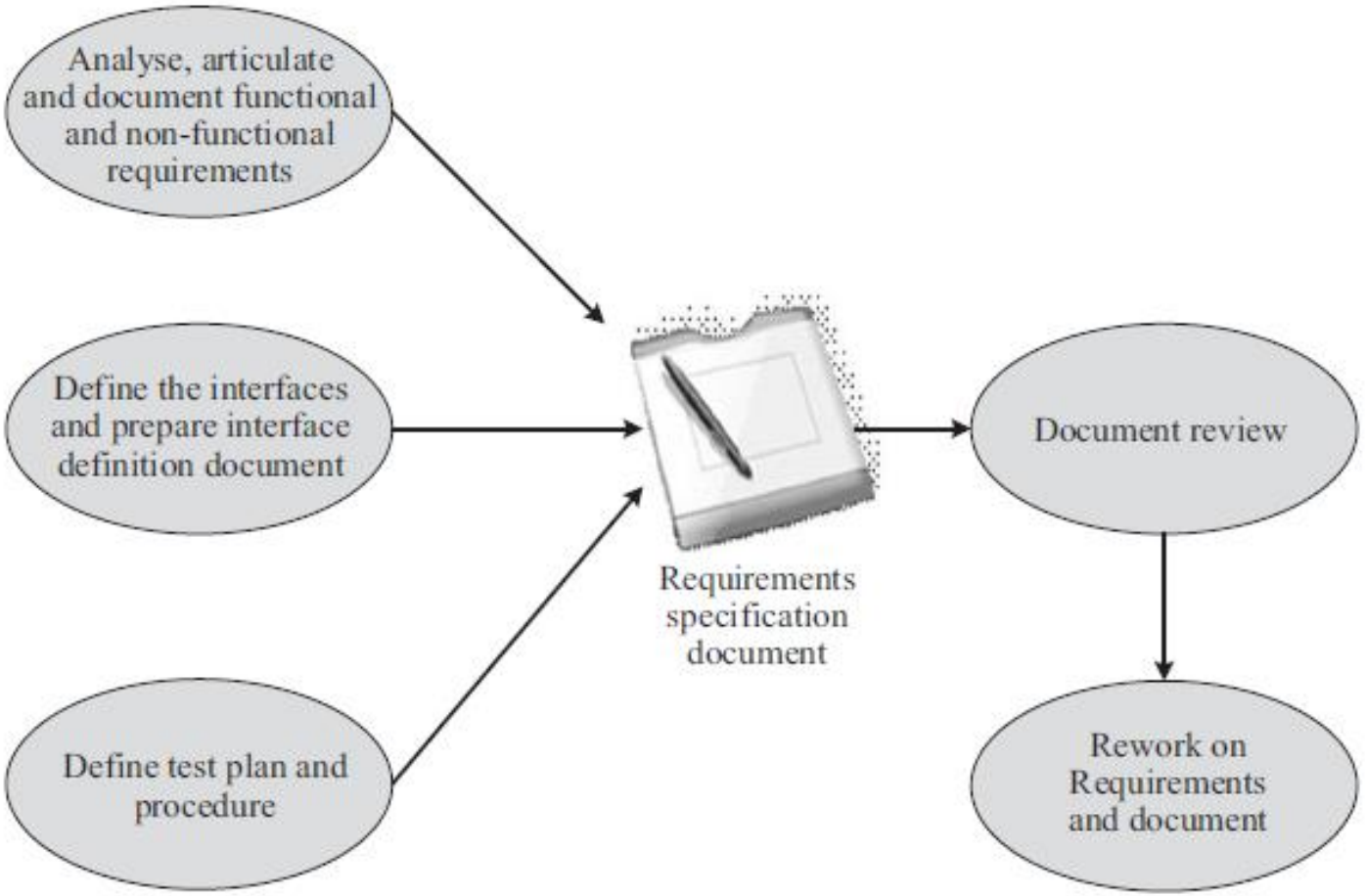


Need& Conceptualization

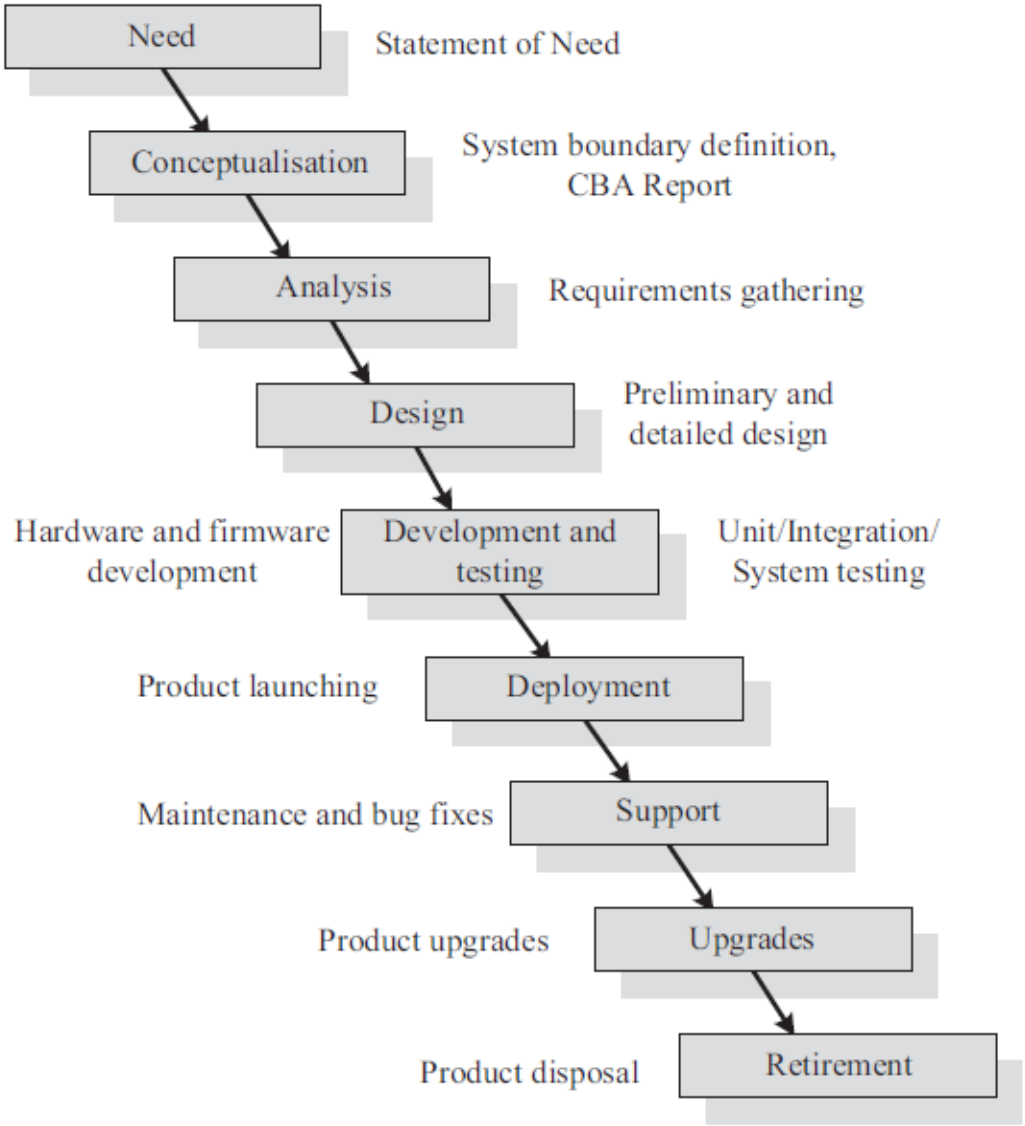
- New product Development
- Product reengineering
- Product maintenance



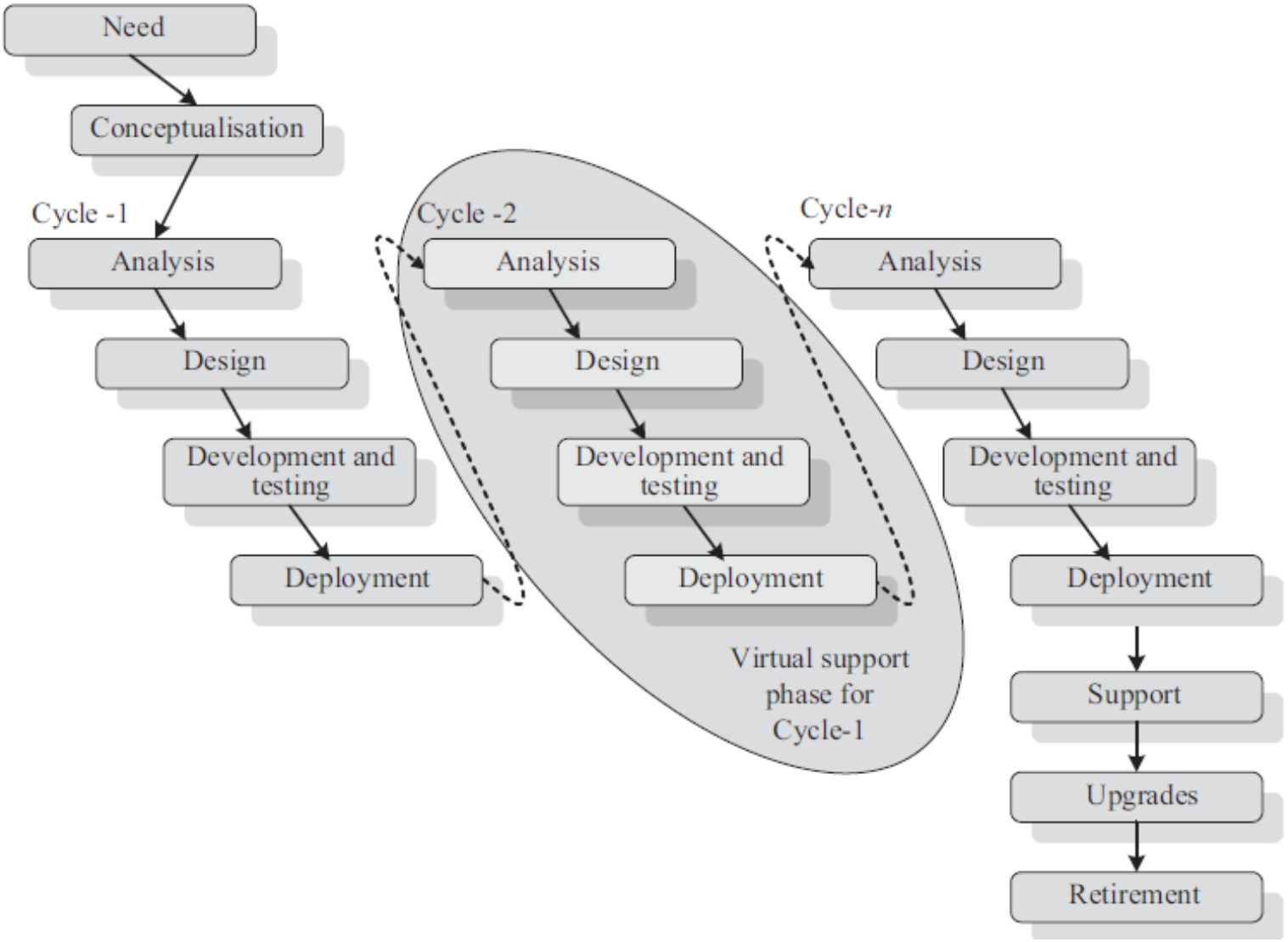
Requirement Analysis



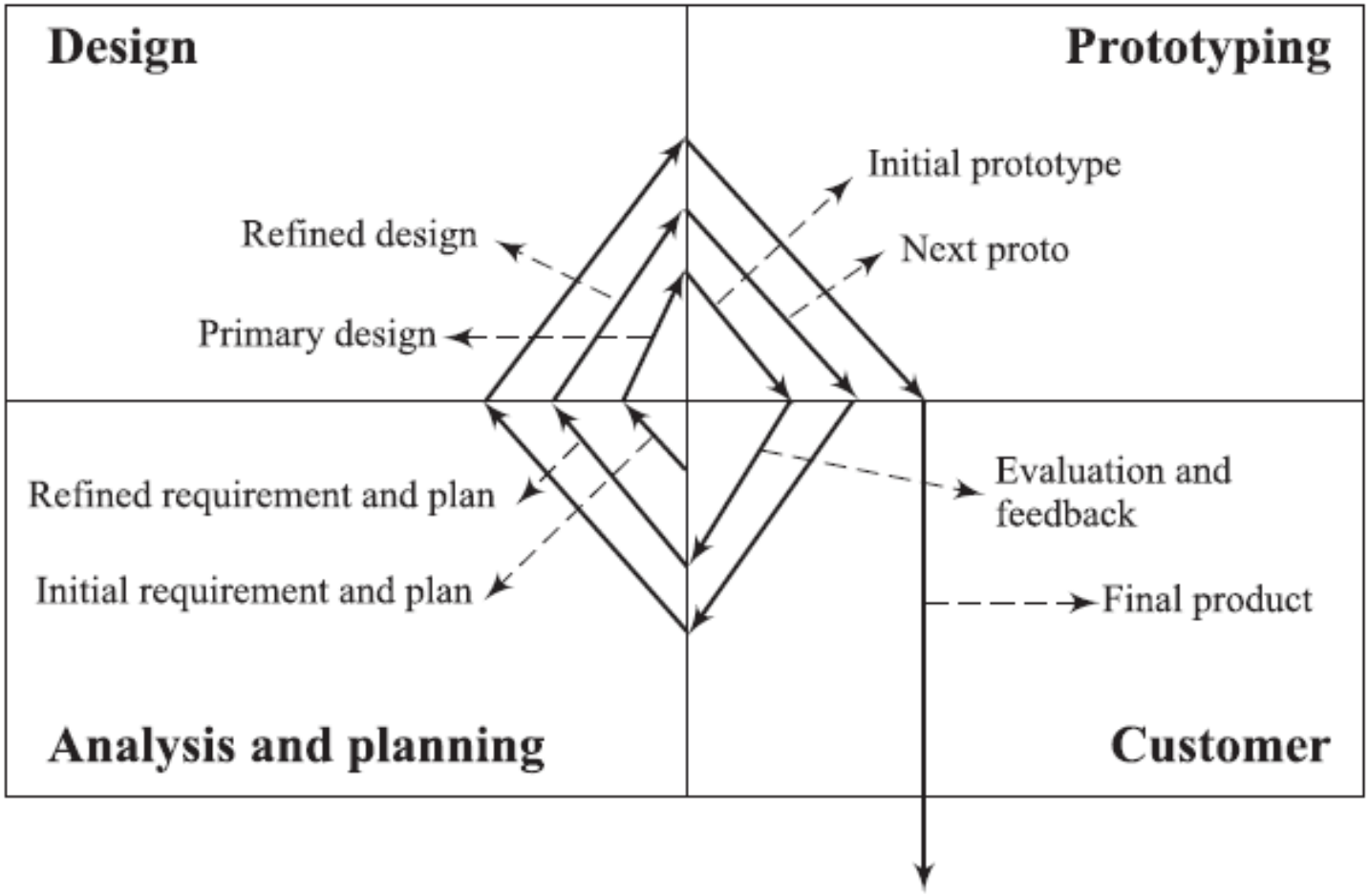
Linear/Water fall Model



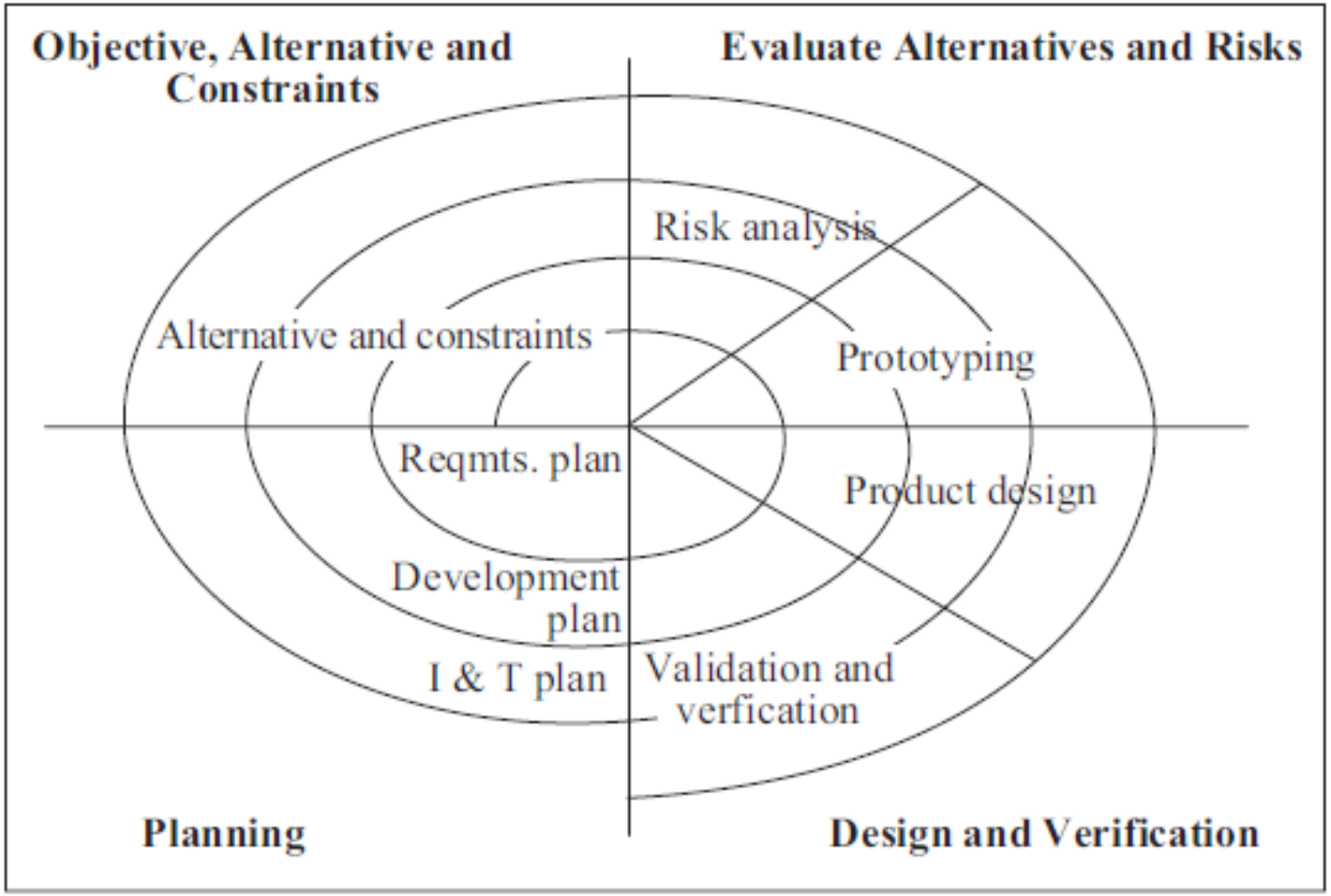
Iterative/Incremental/Fountain Model



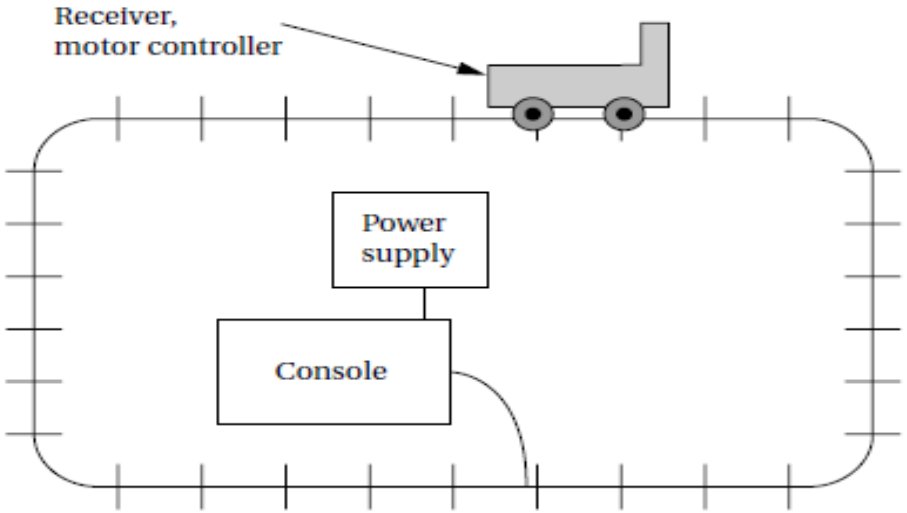
Prototyping/Evolutionary Model



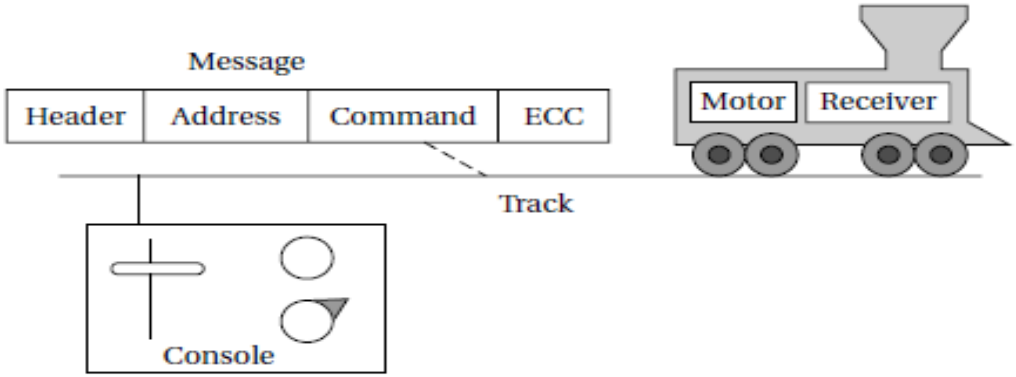
Spiral Model



Embedded System Design Example
Model Train Controller



System setup

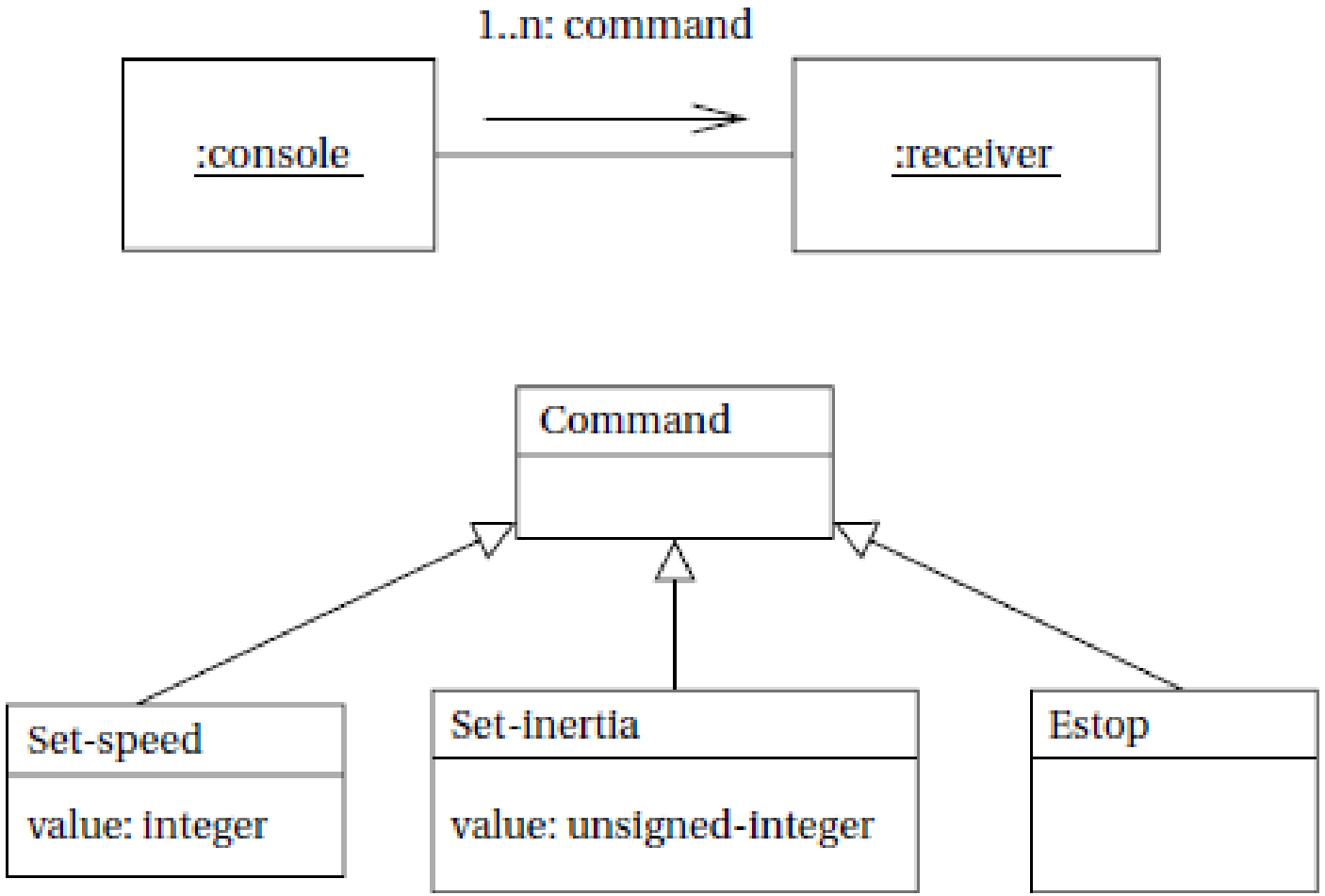


Model Train Controller

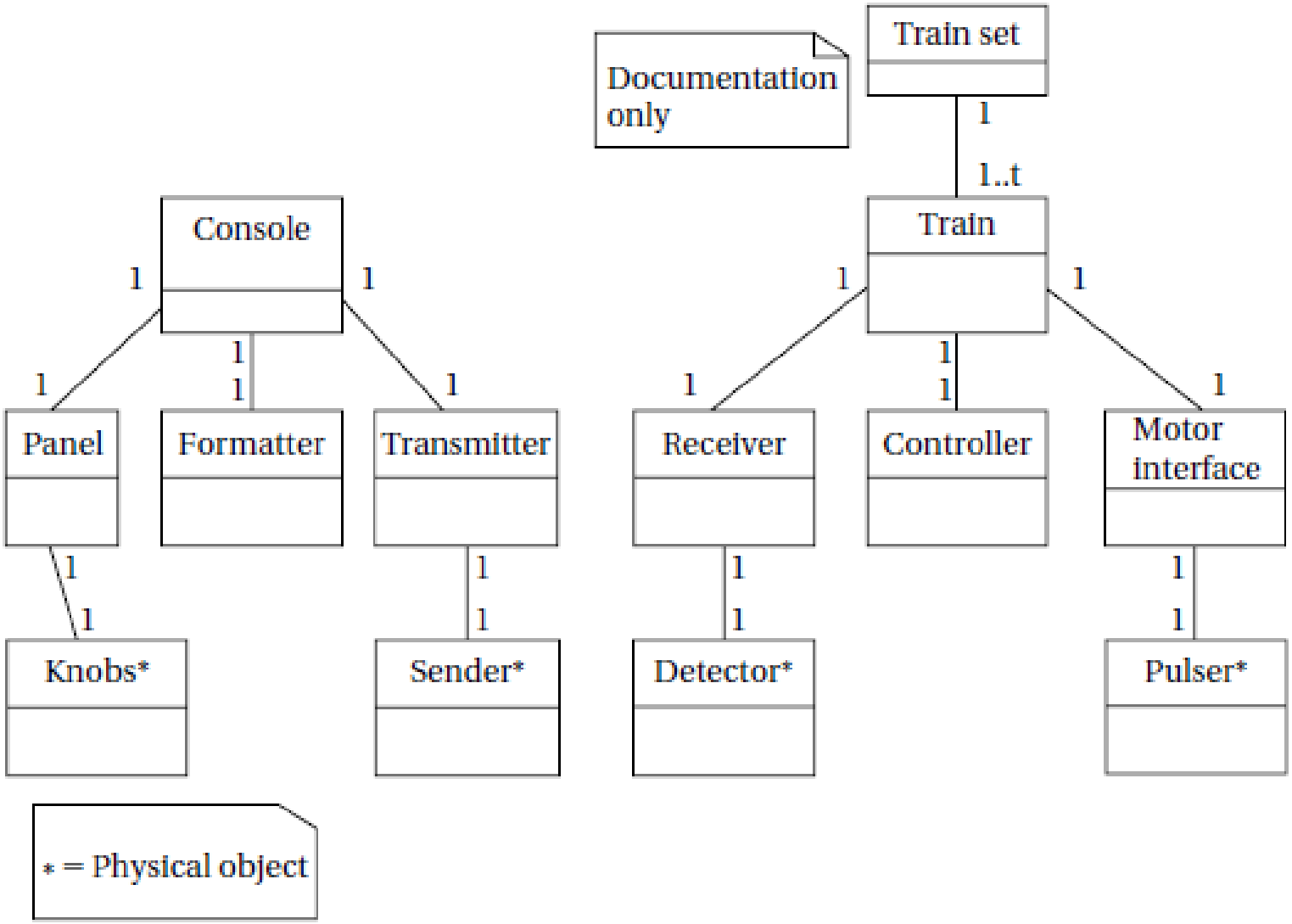
Requirements

- Name: Model Train Controller
- Purpose: Controls speed of up to 8 model trains
- Input: Throttle, Inertia settings, emergency stop, train number
- Output: Train control signals
- Functions: Set engine speed based on inertia settings, respond to emergency stop
- Performance: Can update train speed at least 10 times per second
- Power: 10W
- Cost: Rs/-5000
- Size & Weight: Console should be of small size to be carried on hands, Weight less than 200g.

Class Diagram of Controller and Commands



Class diagram of Train Controller



Detailed Specification

Knobs*
train-knob: integer speed-knob: integer inertia-knob: unsigned-integer emergency-stop: boolean
set-knobs( )

Pulser*
pulse-width: unsigned-integer direction: boolean

Sender*
send-bit( )

Detector*
<integer> read-bit( ): integer

Detailed Specification

Panel
panel-active( ): boolean train-number( ): integer speed( ): integer inertia( ): integer estop( ): boolean new-settings( )

Motor-interface
speed: integer

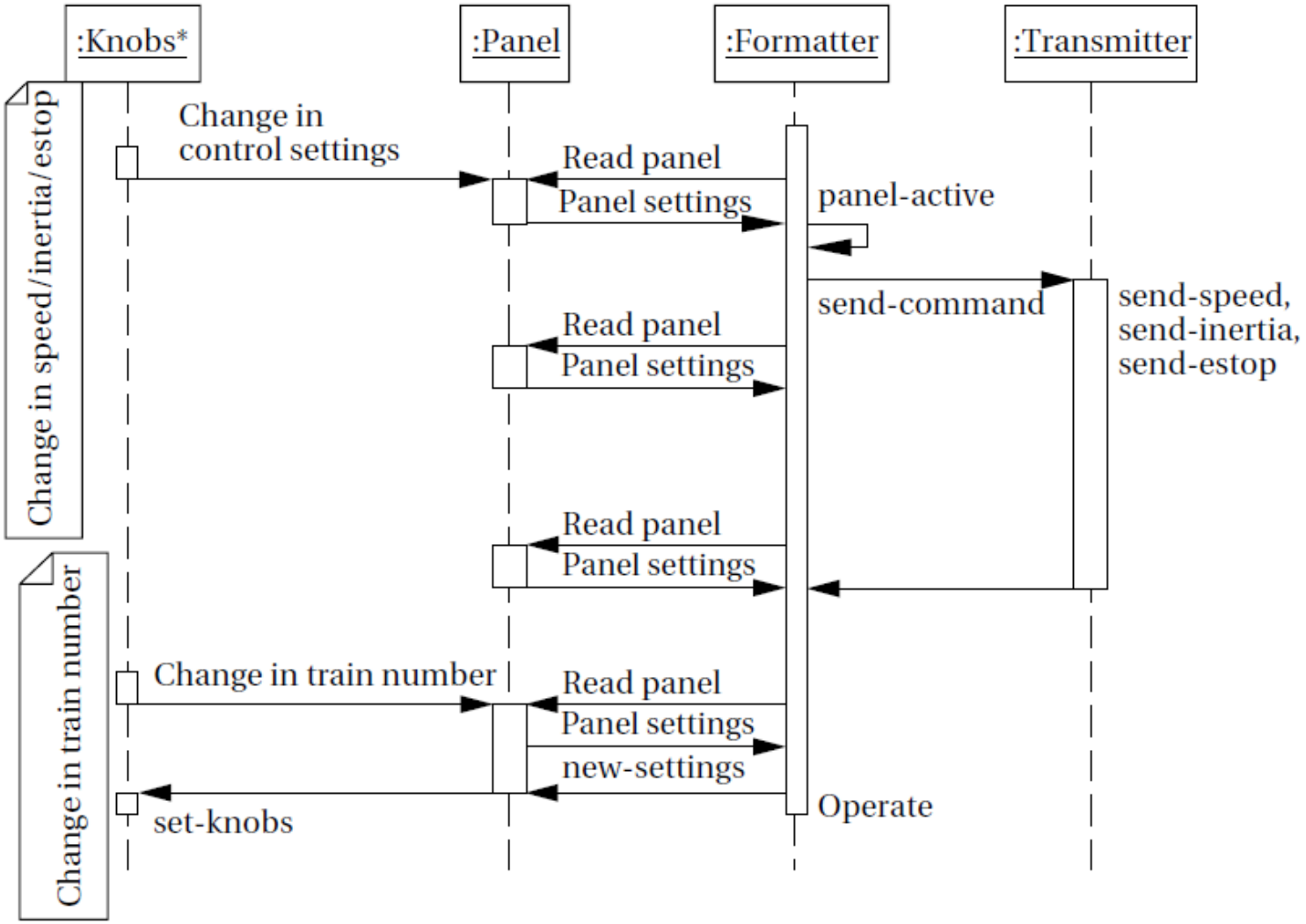
Transmitter
send-speed(adrs: integer, speed: integer) send-inertia(adrs: integer, val: integer) send-estop(adrs: integer)

Receiver
current: command new: boolean
read-cmd( ) new-cmd( ): boolean rcv-type(msg-type: command) rcv-speed(val: integer) rcv-inertia(val: integer)

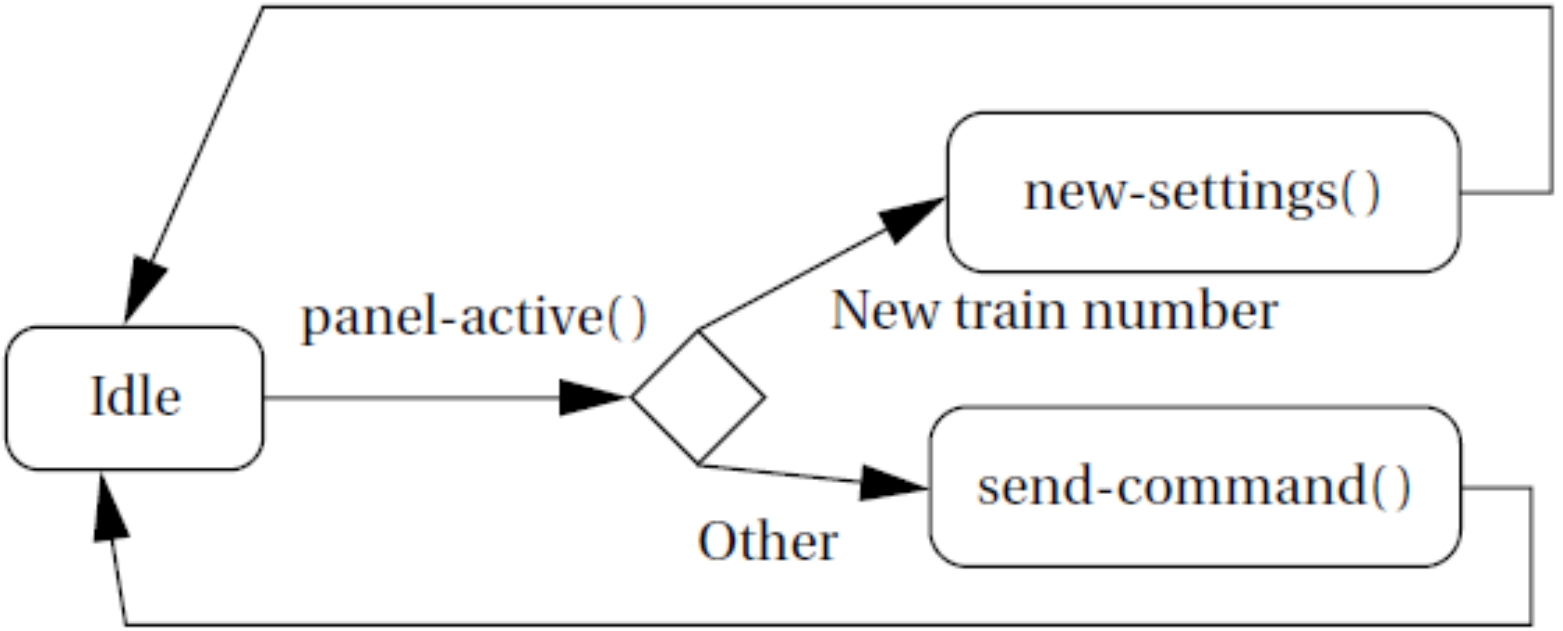
Detailed Specification

Formatter
current-train: integer current-speed[ntrains]: integer current-inertia[ntrains]: unsigned-integer current-estop[ntrains]: boolean
send-command( ) panel-active( ): boolean operate( )

Sequence Diagram



State Diagram-Formatter



State Diagram-Formatter