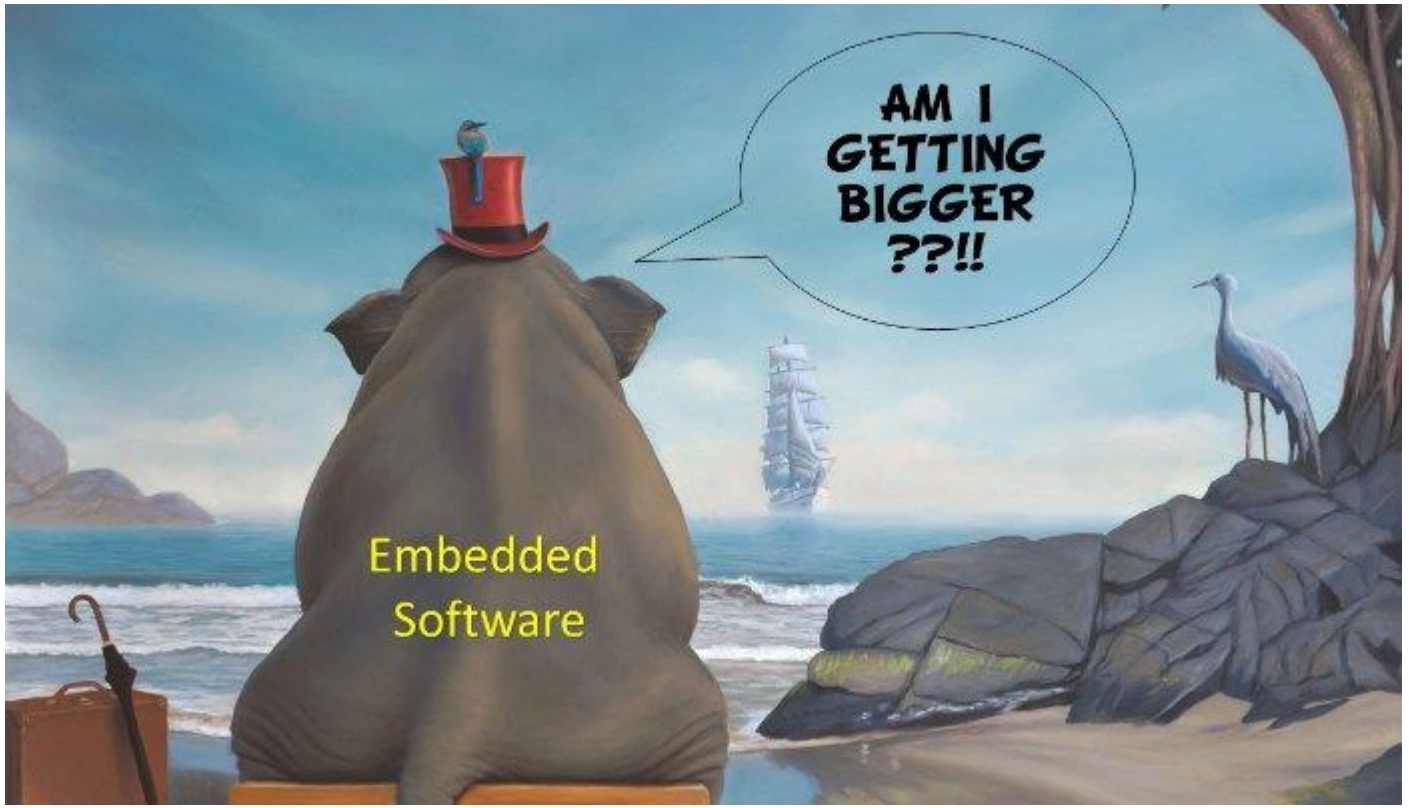
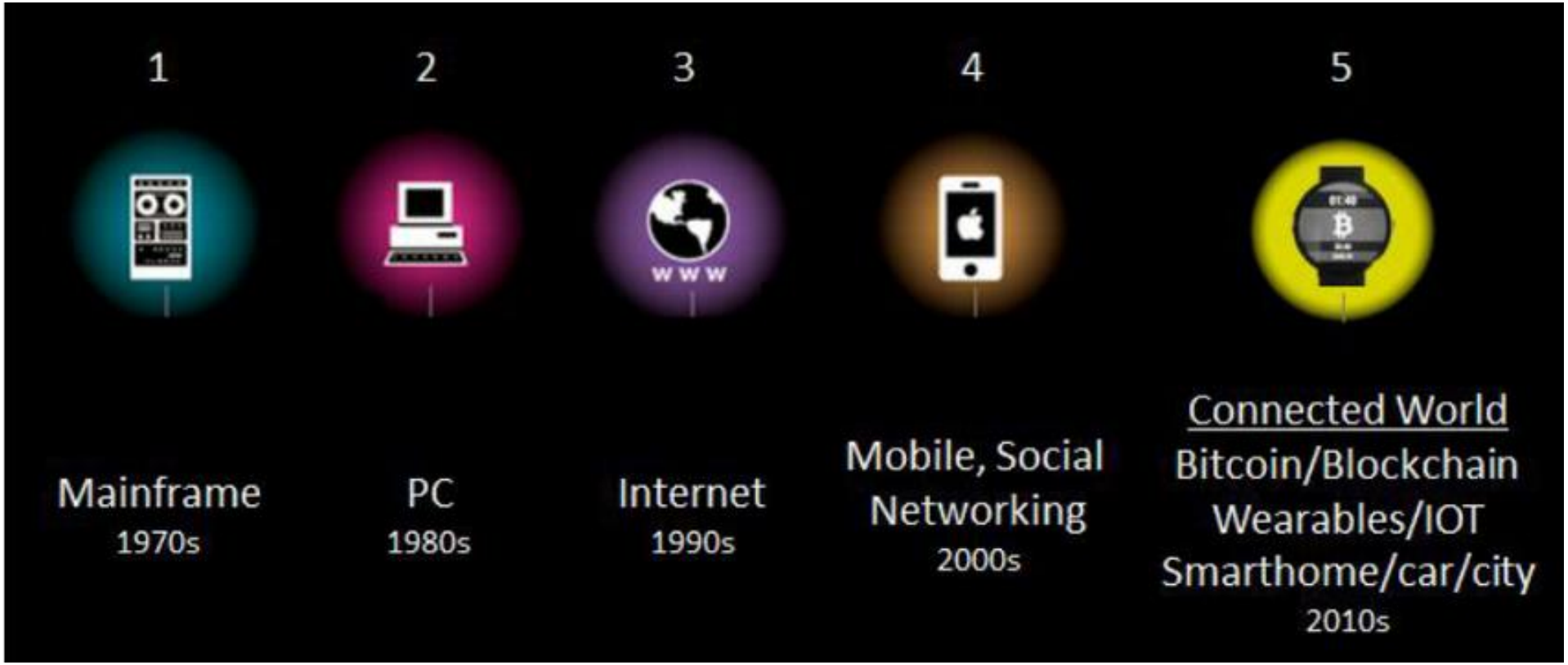
INTRODUCTION TO REAL TIME SYSTEMS



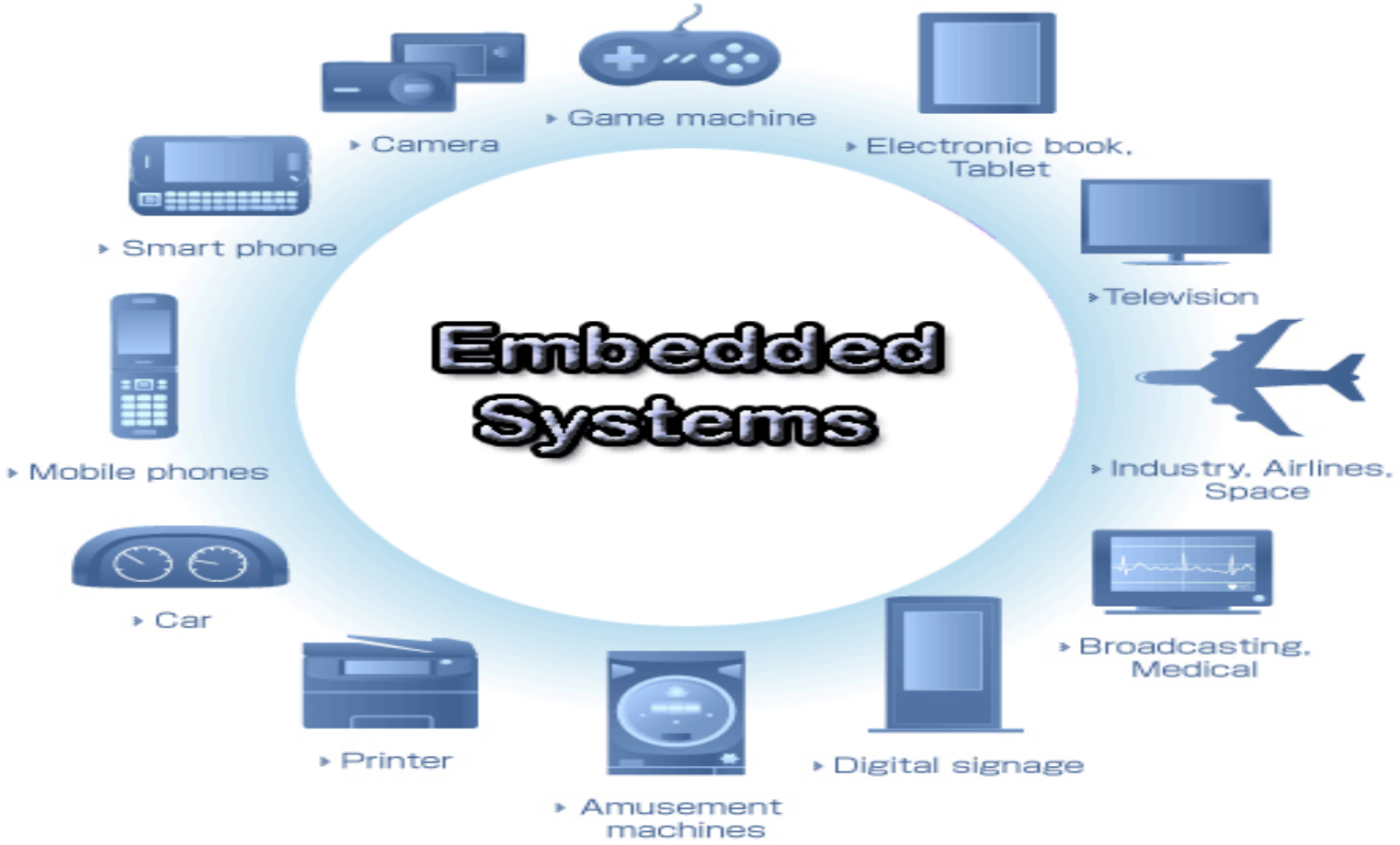
Technology Advancements



INTRODUCTION

- ✓Embedded Systems is an electronic/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware (software).
- ✓Increasingly being used in newer applications.
- ✓Usually real time in nature.

Where are the Real Time Systems?



WHY SURGE IN REAL TIME APPLICATION ??

- Trend of reducing cost of computers
  - ✓Processors
  - ✓Memory
- Flexibility due to internet
- Reducing power consumption
- Reducing size
- Increased:
  - ✓ Processing power
  - ✓Hardware and Software reliability

What Is Real Time?

- ✓Real-time is a quantitative notion of time measured using a physical clock.
  - ➢Example : After a certain event occurs (temperature exceeds 500°C) the corresponding action (coolant shower) must complete within 500ms.
- ✓ In contrast to REAL TIME, LOGICAL TIME DEALS WITH the qualitative notion of time
  - ➢Expressed using notions such as before, after, sometime, eventually etc
  - ➢Example: 'Issue of query book command' and 'display of results'.
- ✓A system is called a real time system, when we need quantitative expression of time, to describe the behavior of the system.

Real Time Systems Applications

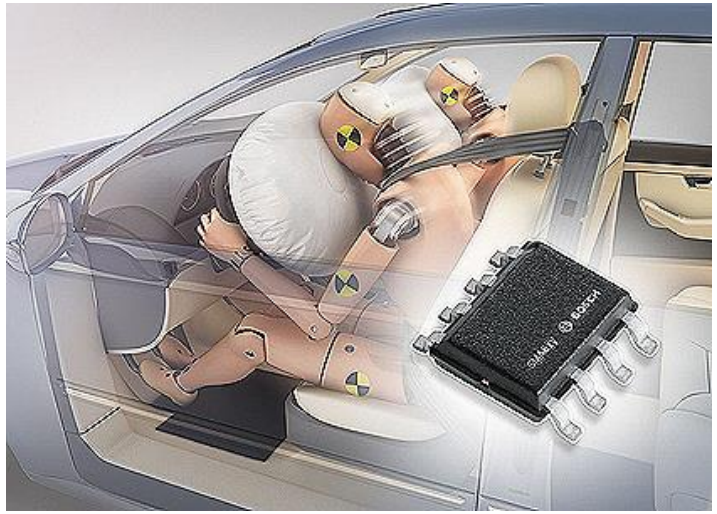
- Industrial
  - ✓Chemical Plant Control
  - ✓Automated Car Assembly plant
- Medical
  - ✓Robot used in recovery of displaced Radioactive material, medical equipments
- Peripheral Equipment
  - ✓Laser Printer, Digital Cameras, sensors

Real Time Systems Applications (contd..)

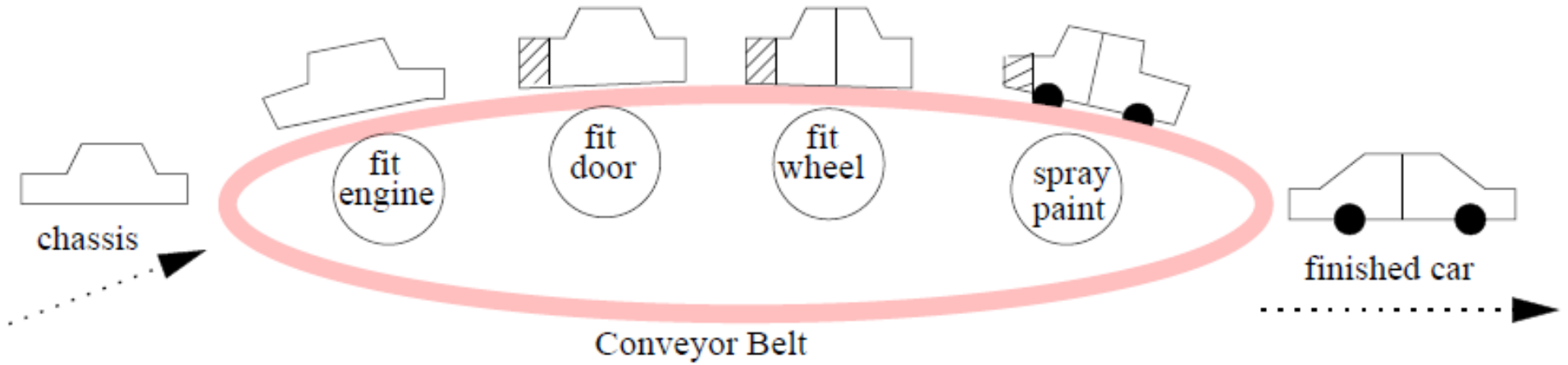
- Transportation
  - ✓Multi-Point Fuel Injection System (MPFI)
  - ✓Automated Car
- Telecommunication Applications
  - ✓Cellular System
- Aerospace
  - ✓Computer on board on aircraft
- Internet
- Multimedia Applications
  - ✓Video Conferencing
- Consumer Electronics
  - ✓Cell phones, digital cameras
- Defense Applications
  - ✓Missile Guidance System

EXAMPLE (AUTOMOTIVE APPLICATIONS)

- ✓Engine Control, Break System, Airbag deployment system
- ✓Windshield wiper, door locks, entertainment systems

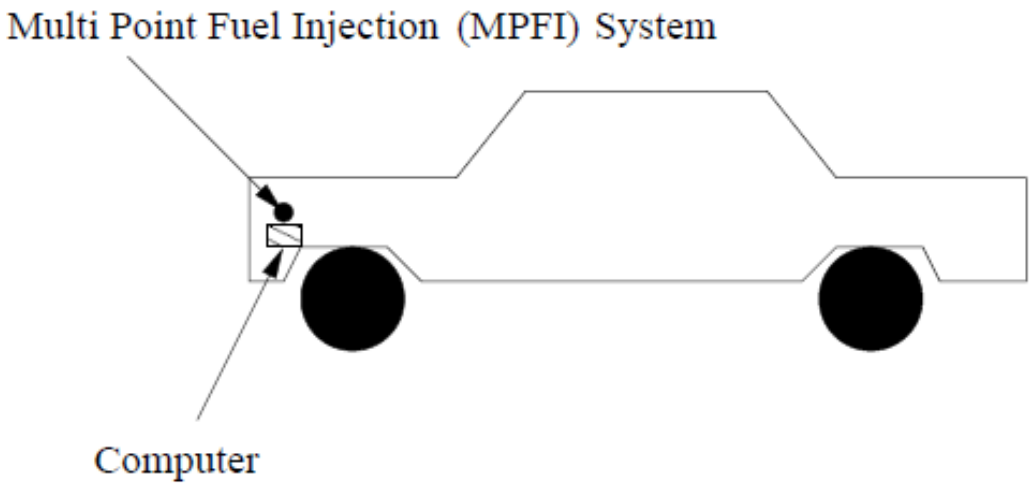


AUTOMATED CAR ASSEMBLY PLANT



MPFI (Multi-Point Fuel Injection)

- ✓ECU (Engine Control Unit) controls the timing and amount of fuel injected:
- ✓Receive signal from various sensor
- ✓Process the signals
- ✓Send control signals to actuators

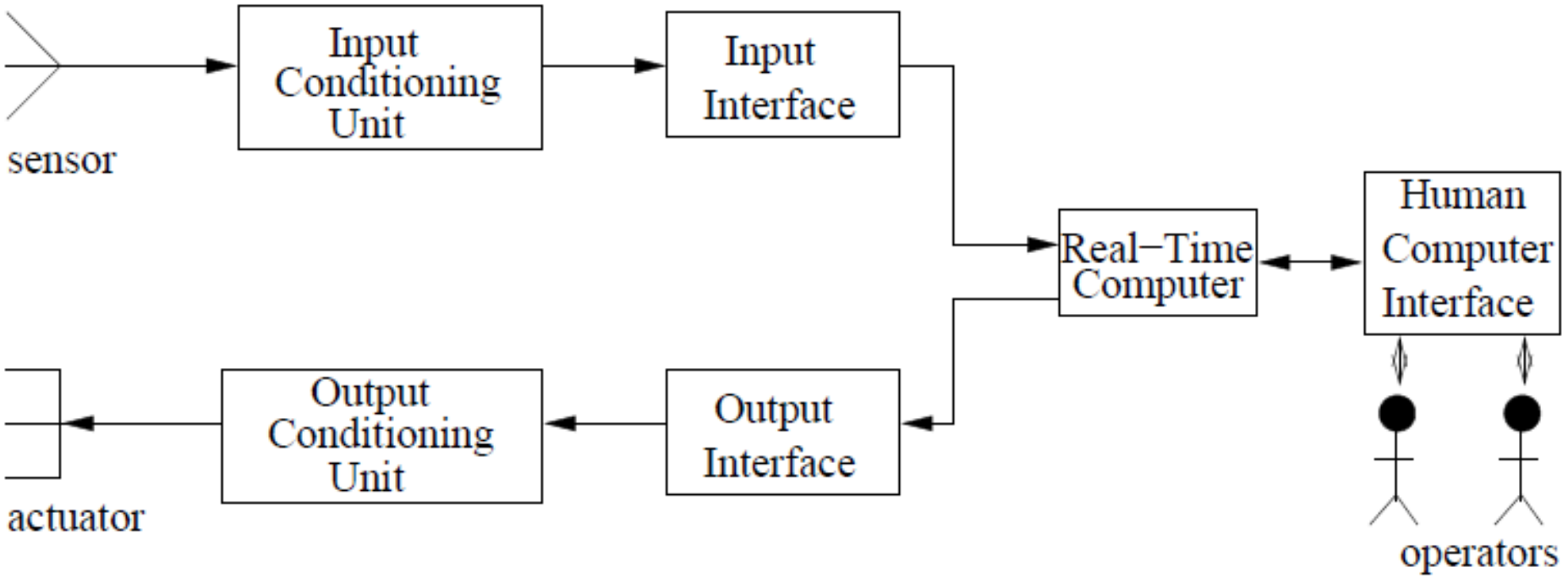


Missile Guidance System

- ✓Sense target and home onto it
- ✓Based on some thermal and electrical characteristics
- ✓Carry out track corrections based on target trajectory
- ✓Sampling and processing activities:

Typically need to be completed in a few micro or milli second

Basic Model of a Real Time System



SENSOR

- ✓A sensor converts some physical characteristic of its environment into electrical signal.
- ✓Example
  - ✓Temperature sensor
  - ✓Pressure sensor
  - ✓Photovoltaic cell

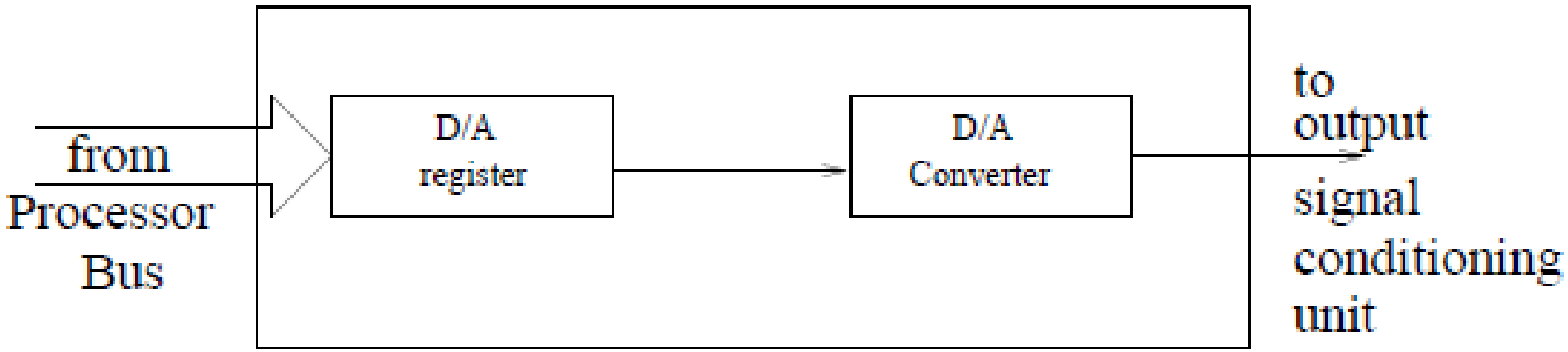
ACTUATOR

- ✓An actuator is a device that takes its inputs from the output interface of a computer and converts these electrical signals into some physical actions on its environment.
- ✓The physical actions may be in the form of motion, change of thermal, electrical, pneumatic or physical characteristics of its some objects.
- ✓Example
  - ✓Motors, Heaters

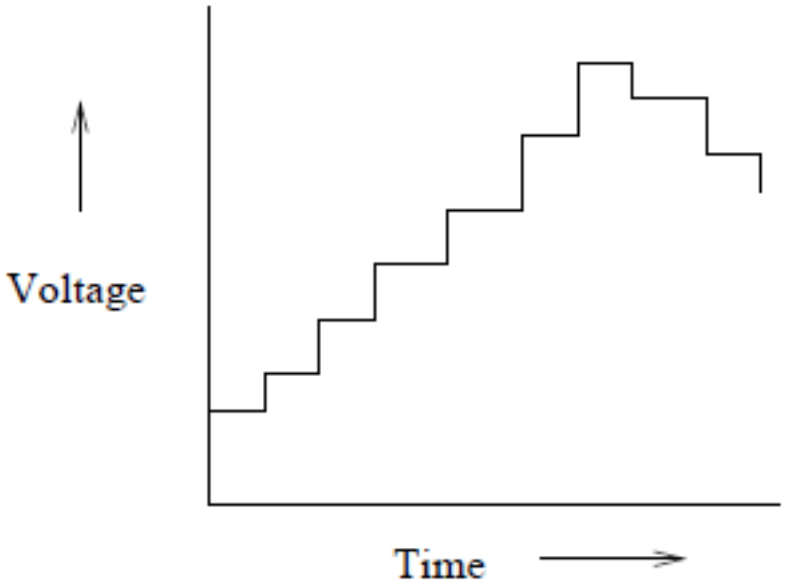
Signal Conditioning Units

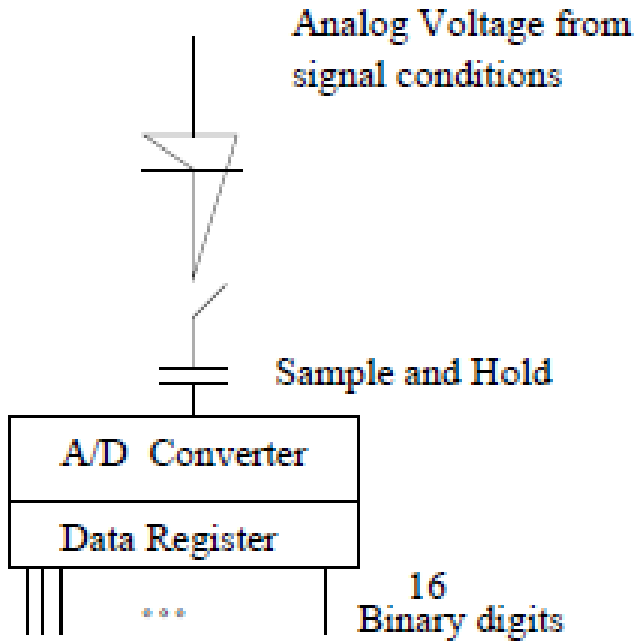
- ✓Voltage Amplification
- ✓Voltage level shifting
- ✓frequency range shifting and filtering
- ✓signal mode conversion

Output Interface



Analog to Digital Conversion

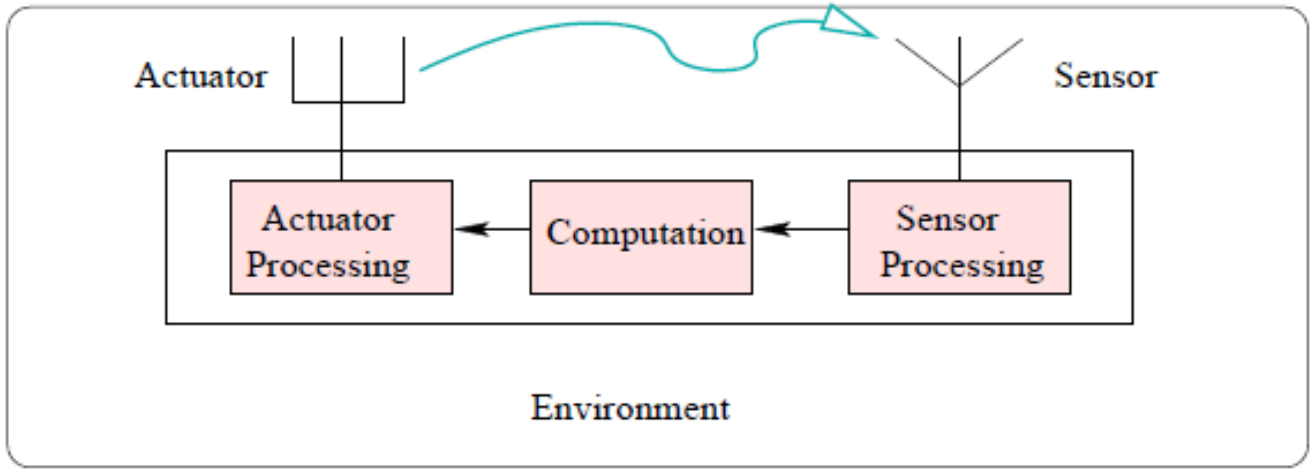




Characteristics of Real Time Systems

- Time Constraints
  - Task deadline specifies the time before which the task must complete and produce the results.
- New Correctness Criterion
  - Logical correctness and the time at which results are produced
- Embedded
- Safety criticality
  - A safety critical system is required to be highly reliable since any failure of the system can cause extensive damages.
- Concurrency
  - Needs to respond to several independent events within very short and strict time bounds

- Distributed and Feedback structure
  - In many real time systems, the different components of the system are naturally distributed across widely spread geographic locations.
  - At each data source, it makes good design sense to locally process the data before being passed on to a central processor.
  - Ex: A petroleum refinery plant



- Task Criticality
  - It is the measure of the cost of failure of a task.
  - The higher the criticality of a task, the more reliable it should be made.
  - In the event of a failure of a highly critical task, immediate failure detection and recovery are important.
- Custom Hardware
  - A real time system is often implemented on custom hardware that is specifically designed and developed for the purpose.
  - Ex: cellphone

- Reactive

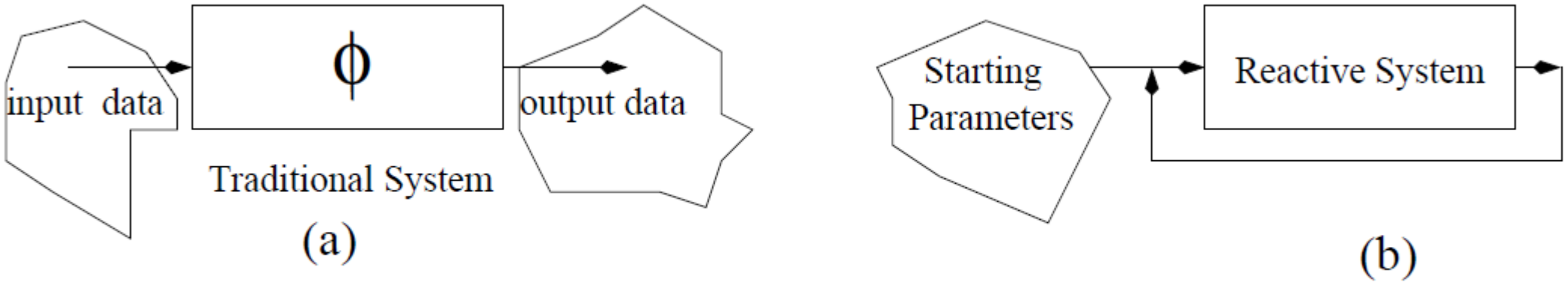


Figure 10: Traditional versus Reactive Systems

- Stability
  - Under overload conditions, real time systems need to continue to meet the deadlines of the most critical tasks, though the deadlines of non-critical tasks may not be met.
- Exception Handling
  - A failure should be detected and the system should continue to operate in a gracefully degraded mode rather than shutting off abruptly.

Safety and Reliability

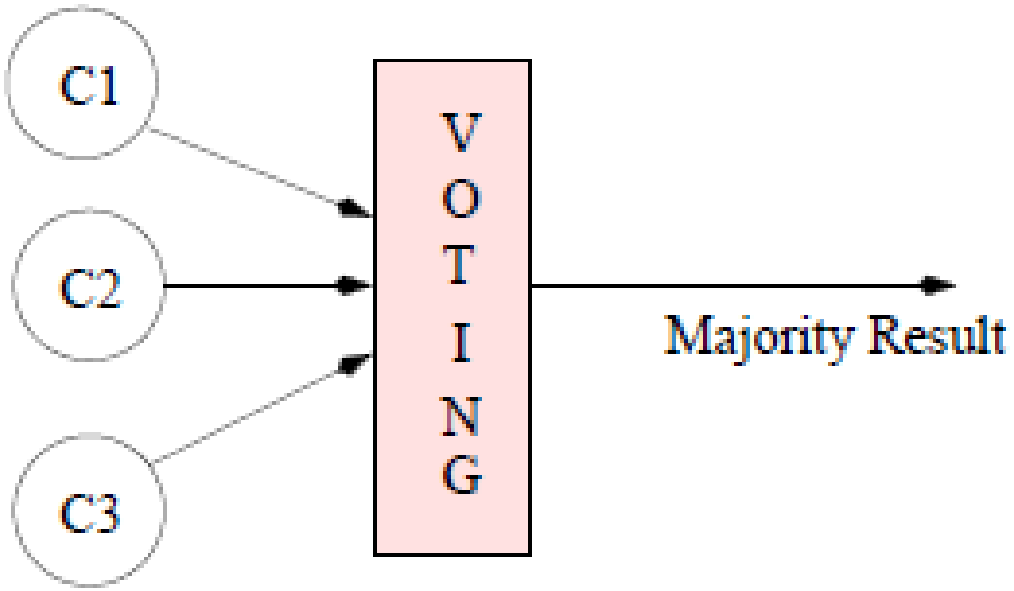
- In traditional systems, safety and reliability are normally considered to be independent issues.
  - Ex: a) A word processing software b) Hand gun
- In real time systems, safety and reliability are coupled together.
  - Ex: Traffic light controller that controls the flow of traffic at a road intersection.
- A fail-safe state of a system is one which if entered when the system fails, no damage would result.
- A safety critical system is one whose failure can cause severe damages.
  - Ex: Navigation system on board an aircraft.

How to achieve high reliability?

- Error Avoidance
  - Using well founded software engineering practices, using sound design methodologies etc
- Error detection and removal
  - Can be achieved to a large extent by conducting thorough reviews and testing.
- Fault tolerance
  - In situations where errors are present, the system should be able to tolerate the faults and compute the correct results.

- Two methods that are popularly used to achieve hardware fault tolerance
- Built in Self Test (BIST)
  - The system periodically performs self tests of its components.
  - Upon detection of a failure, the system automatically reconfigures itself by switching out the faulty component and switching in one of the redundant good components.

- Triple Modular Redundancy (TMR)
  - In TMR, three redundant copies of all critical components are made to run concurrently.
  - The system performs voting of the results produced by the redundant components to select the majority result.



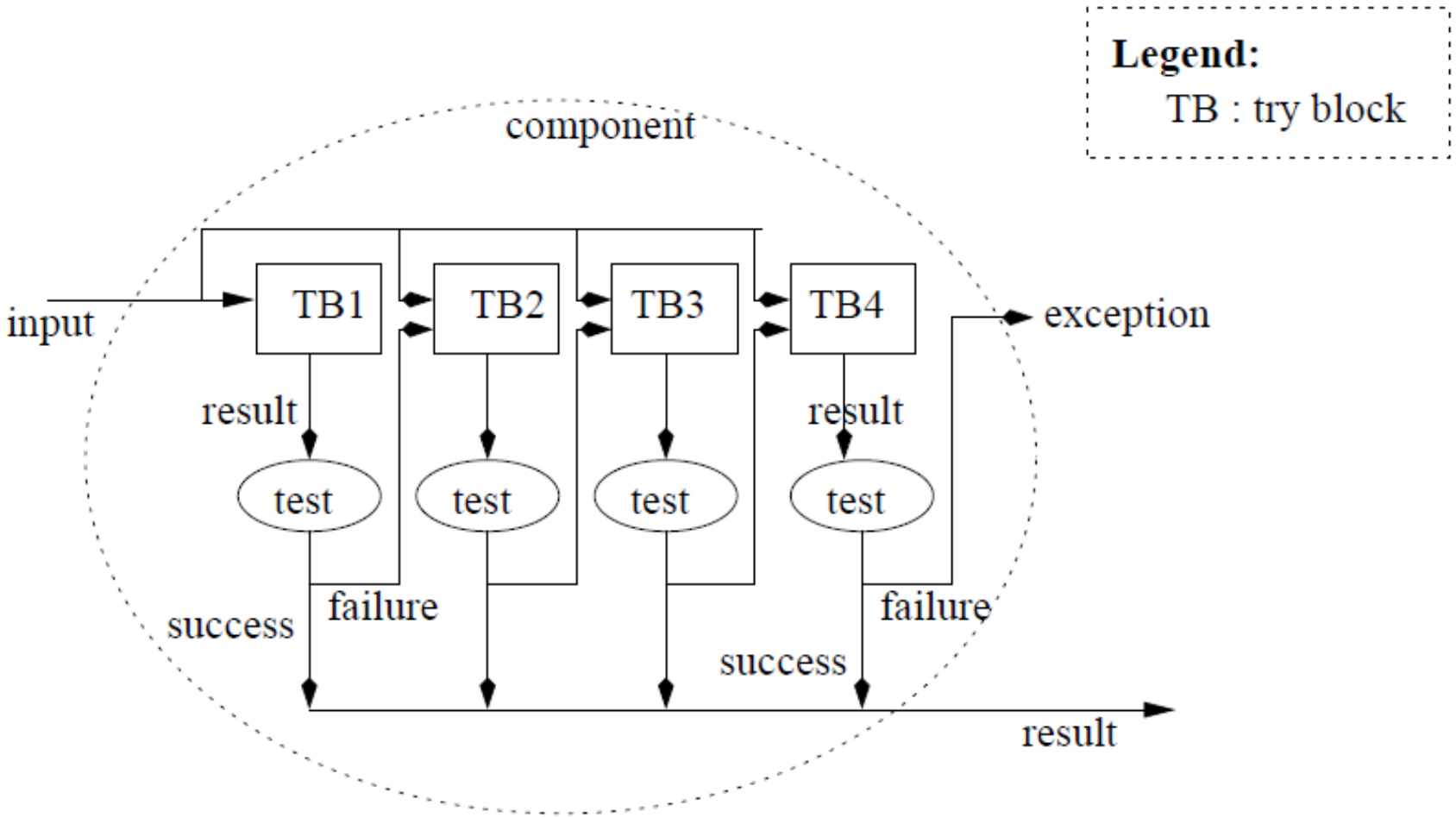
Software fault tolerance techniques

- Two methods are popularly used
  - N-version programming
  - Recovery block techniques

N- Version Programming

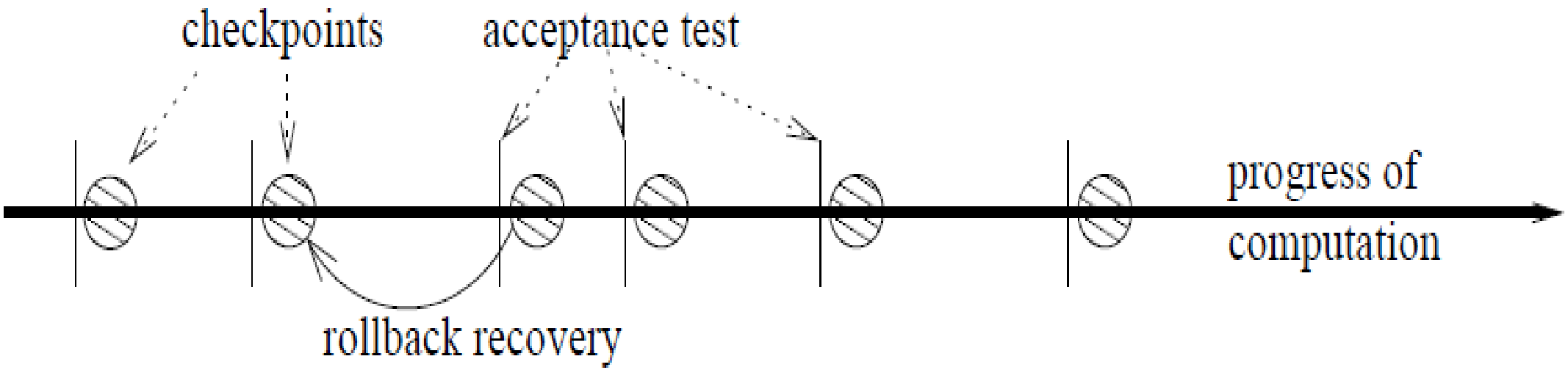
- In the N-version programming technique, independent teams develop N different versions of a software component (module).
- The results produced by the different versions of the module are subjected to voting at run time and the result on which majority of the components agree is accepted.
- The central idea behind this scheme is that independent teams would commit different types of mistakes, which would be eliminated when the results produced by them are subjected to voting.

Recovery blocks



- The redundant components are called try blocks.
- Different algorithms are used in different try blocks.
- Redundant copies run one after another.
- The results produced by a try block are subjected to an acceptance test.
- If the test fails, then the next try block is tried.
- This is repeated in a sequence until the result produced by a try block successfully passes the acceptance test.

Checkpointing and Rollback recovery

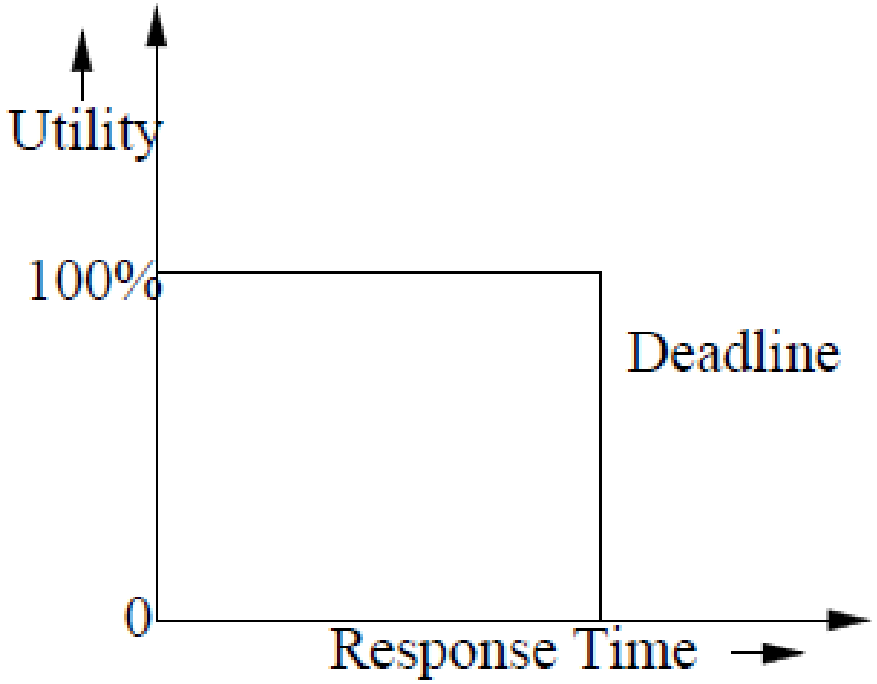


- As the computation proceeds, the system state is tested each time after some meaningful progress in computation is made.
- After a state check test succeeds, the state of the system is backed up on a stable storage.
- In case the next test does not succeed, the system can be made to roll-back to the last check pointed state.
- After a rollback from a check pointed state a fresh computation can be initiated.
- This technique is especially useful, if there is a chance that the system state may be corrupted as the computation proceeds, such as data corruption or processor failure.

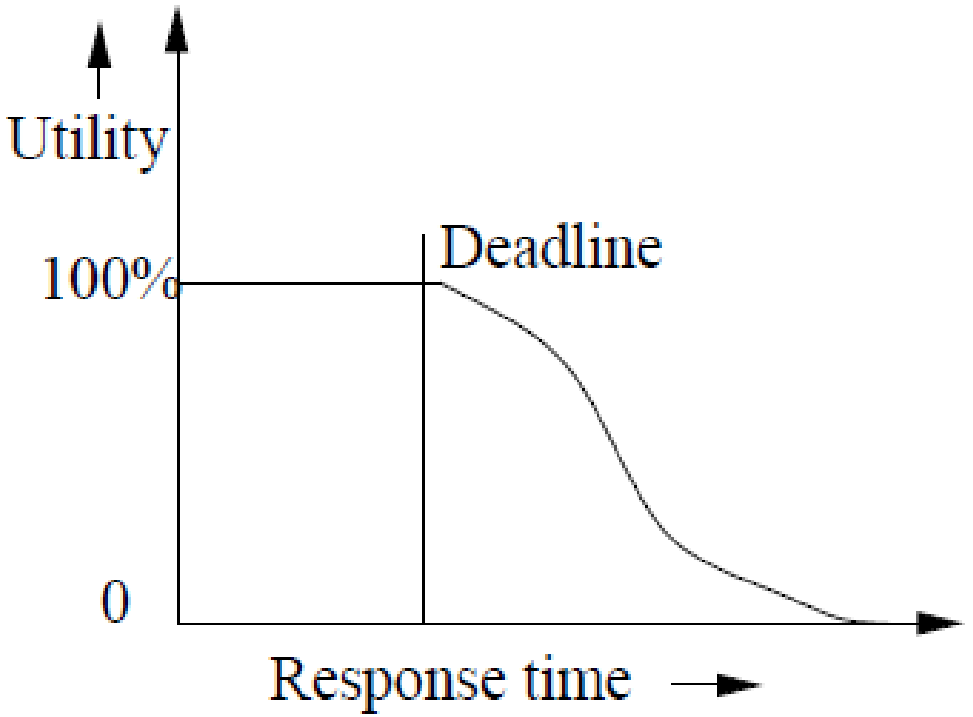
TYPES OF REAL TIME TASKS

- HARD REAL TIME TASKS
  - Constrained to produce its results within certain predefined time bounds.
  - Example: Robot, Anti missile system
  - There is no reward in completing a hard real time task much ahead of its deadline
  - Time bounds range from several micro seconds to few milli seconds

- FIRM REAL TIME TASKS
  - Task is associated with predefined deadline before which it is required to produce its results
  - Unlike a hard real time task, even when a firm real time task does not complete within its deadline, the system does not fail.
  - The late results are merely discarded.
  - Example: video conferencing, satellite based tracking of enemy movements
  - Time bounds range from few milli seconds to several hundreds of milli seconds


Utility
100%
Deadline
0
Response Time

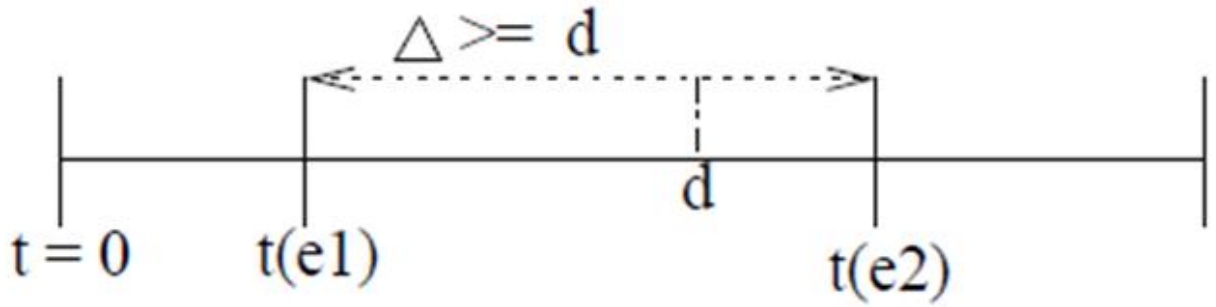
- SOFT REAL TIME TASKS
  - Timing constraints are expressed in terms of average response times required.
  - Example: web browsing.
  - Time bounds range from a fraction of a second to few seconds


Utility
100%
Deadline
0
Response time

CLASSIFICATION OF TIMING CONSTRAINTS

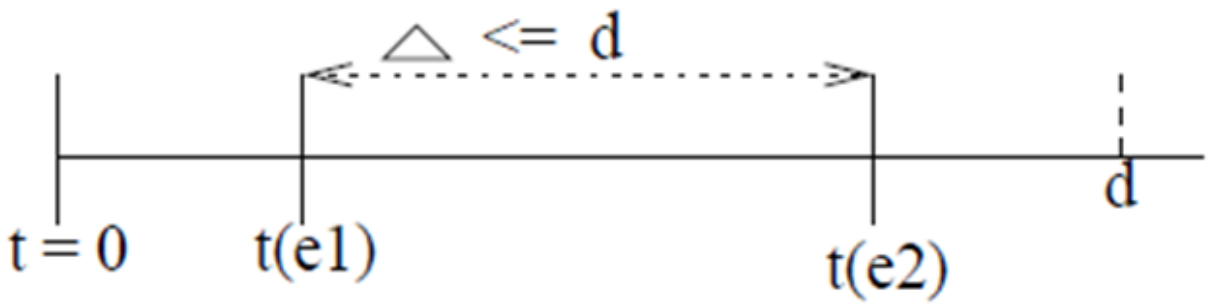
- Delay constraint
  - It captures the minimum time (delay) that must elapse between the occurrence of two arbitrary events e1 and e2
  - e2 must occur after atleast d time units have elapsed after the occurrence of e1.

$$t(e_2) - t(e_1) \geq d$$



- Deadline constraint
  - It captures the permissible maximum separation between any two arbitrary events e1 and e2
  - A deadline constraint implies that e2 must occur within d time units of e1's occurrence.

$$t(e_2) - t(e_1) \leq d$$

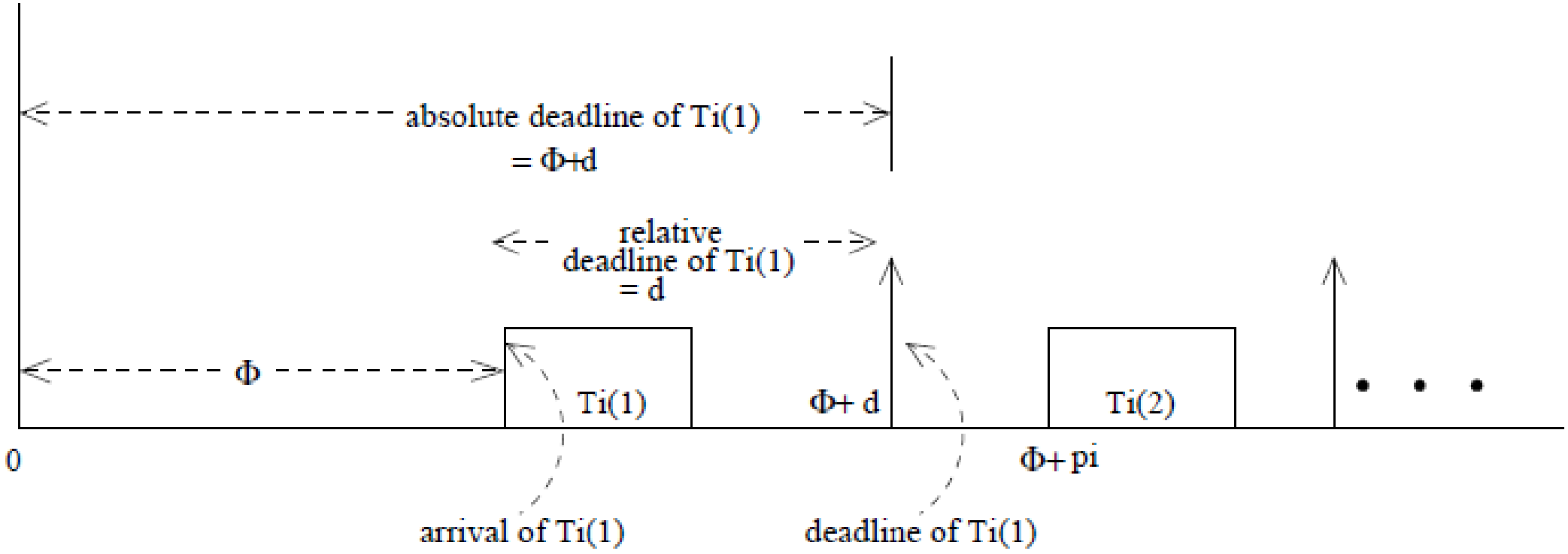


- Duration constraint
  - Specifies the time period over which the event acts
  - A duration constraint can either be a minimum type or maximum type
  - In minimum type, once the event starts the event must not end before a certain minimum duration.
  - In maximum type, once the event starts event must end before a certain maximum duration elapses.

REAL TIME TASK SCHEDULING TERMINOLOGIES

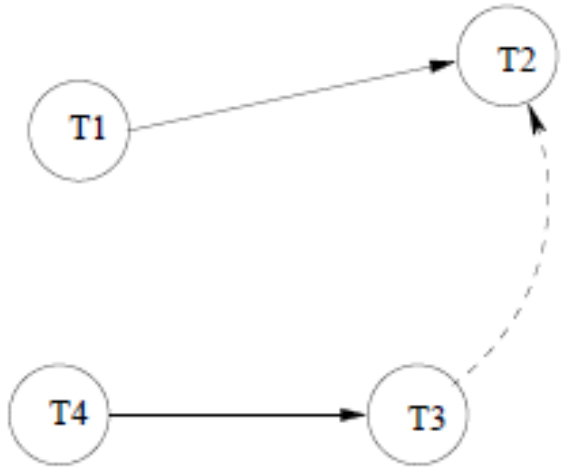
- TASK INSTANCE
  - A task is generated when some specific event occurs
  - Most real time tasks recur with certain fixed period
  - Each time a task recurs it is called instance of task
  - Each instance of a real time task is associated with deadlines by which it needs to complete and produce results.
  - The jth instance of a task T_i would be denoted as T_i(j)

Relative Deadline v/s Absolute Deadline



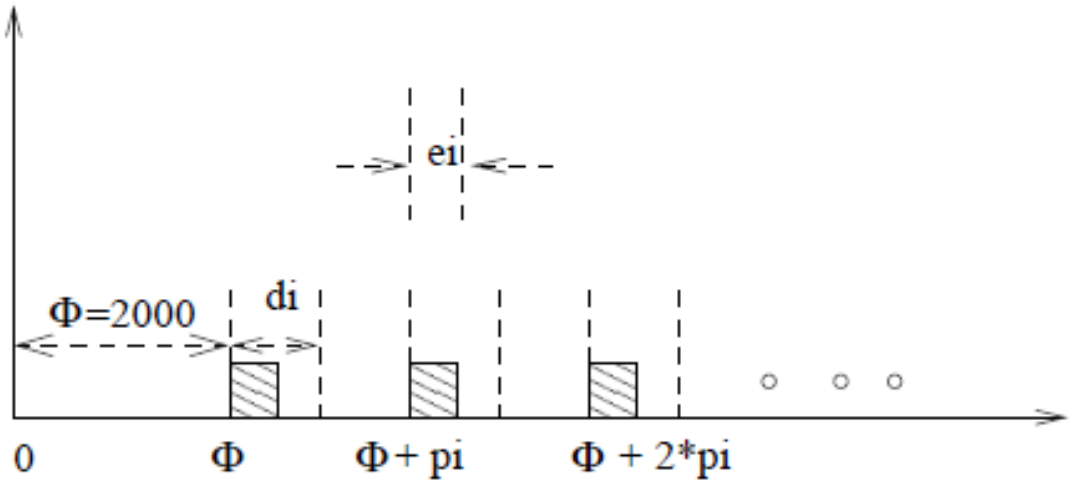
- Absolute Deadline
  - The interval of time between time 0 and the actual instant at which the deadline occurs as measured by some physical clock.
- Relative Deadline
  - Time interval between the arrival of a task and the corresponding deadline.
- The relative deadline of the task $T_i(1)$ is d, whereas its absolute deadline is Φ +d.

- Response Time
  - Time duration from the occurrence of the event generating the task to the time the task produces its results.
- Task Precedence
  - A task is said to precede another task, if the first task must complete before the second task start.



TYPES OF REAL TIME TASKS

- Periodic Task
  - A periodic task is one that repeats after a certain fixed interval
  - They are referred to as clock driven tasks
  - The fixed time interval after which a task repeats itself is called *period* of the task.
  - Ex: Chemical Plant



- If T_i is a periodic task, then the time from 0 till the occurrence of the first instance of T_i (i.e $T_i(1)$) is denoted by Φ_i and is called the phase of the task.
- Formally, a periodic task T_i can be represented by a 4 tuple (Φ_i, p_i, e_i, d_i)

where p_i is the period of task, e_i is the worst case execution time of the task, d_i is the relative deadline of the task.

Task representation

Consider track correction task typically found in a rocket control software. Track correction starts 2000ms after launch of rocket and it periodically recurs every 50ms. Each instance of task requires a processing time of 8ms and relative deadline of 50ms. Represent this using periodic task.

- T_i=(2000ms, 50ms, 8ms, 50ms)
- If p_i= d_i, T_i=(2000ms, 50ms, 8ms)
- When Φ_i=0, T_i=(50ms, 8ms)

- Sporadic Task
  - Recurs at random instants

$$T_i = (e_i, g_i, d_i)$$

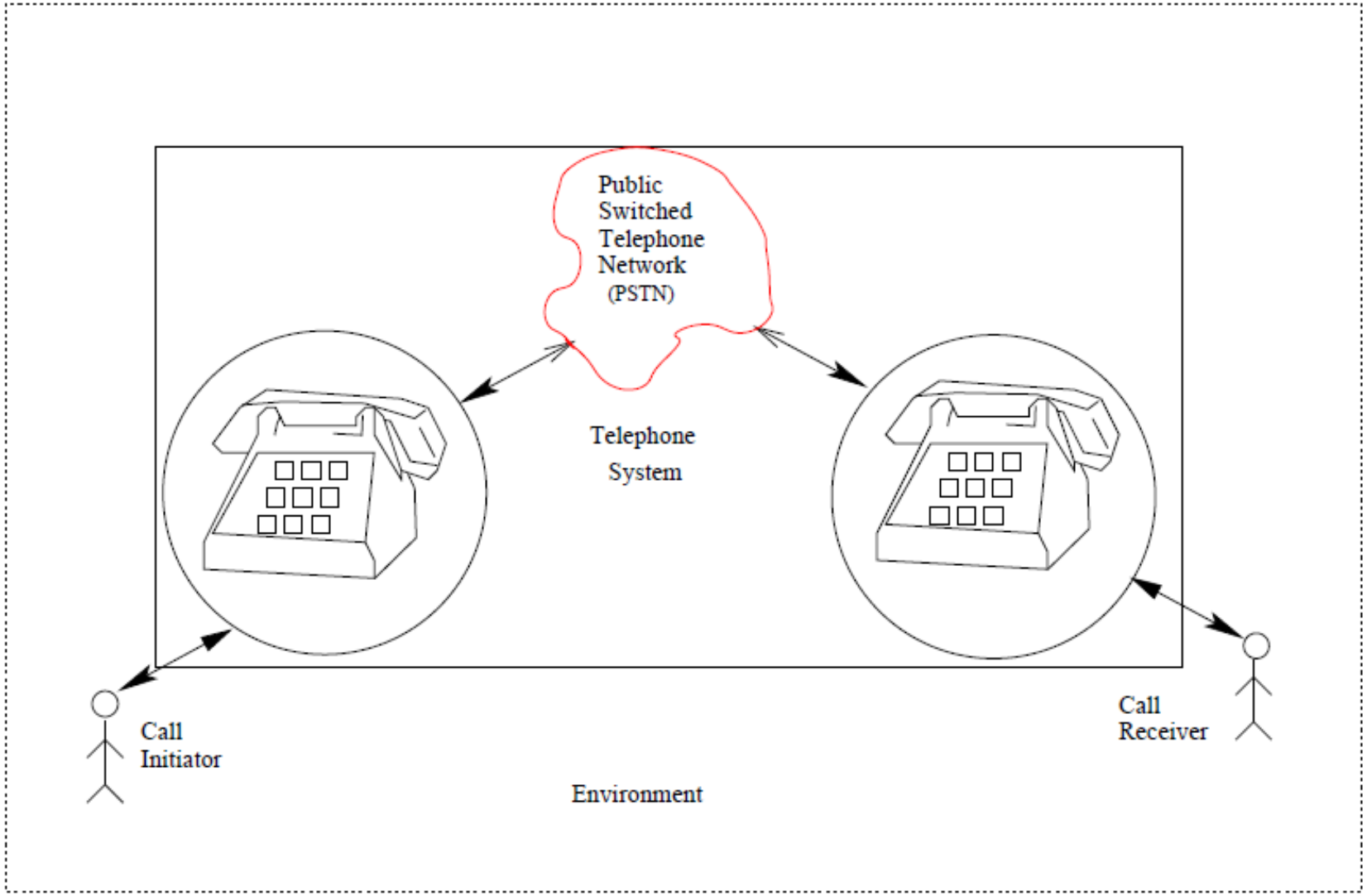
    - e_i is the worst case execution time of an instance of task
    - g_i is minimum separation between two consecutive instances of task
    - d_i is relative deadline
  - The minimum separation gi between two consecutive instances of task implies that once an instance of a sporadic task occurs, the next instance cannot occur before gi time units have elapsed.
  - Example: Task of robot to handle an obstacle, In a factory, the task that handles fire conditions.

- Aperiodic task
  - It can arise at random instants
  - The minimum separation gi between two consecutive instances can be 0.
  - The deadline for an aperiodic task is expressed either as an average value or is expressed statistically
  - Example: Data logging

EVENTS IN A REAL TIME SYSTEMS

- Stimulus events
  - Generated by the environment and act on the systems
  - Example periodic sensing of temperature of the reactor in a nuclear plant
- Response events
  - Produced by the system in response to some stimulus events
  - Example in a chemical plant as soon as the temperature exceeds 100 deg the system responds by switching off the heater

Schematic representation of a telephone system



- Stimulus-Stimulus (SS)
  - In this case, the deadline is defined between two stimuli.
  - This is a behavioral constraint, since the constraint is imposed on the second event which is a stimulus
  - Example: Once a user completes dialing a digit, he must dial the next digit within the next 5 seconds. Otherwise an idle tone is produced.

- Stimulus Response (SR)
  - In this case, the deadline is defined on the response event, measured from the occurrence of the corresponding stimulus event.
  - This is a performance constraint, since the constraint is imposed on a response event.
  - Ex: Once the receiver of the hand set is lifted, the dial tone must be produced by the system within 2 seconds, otherwise a beeping sound is produced until the handset is replaced.

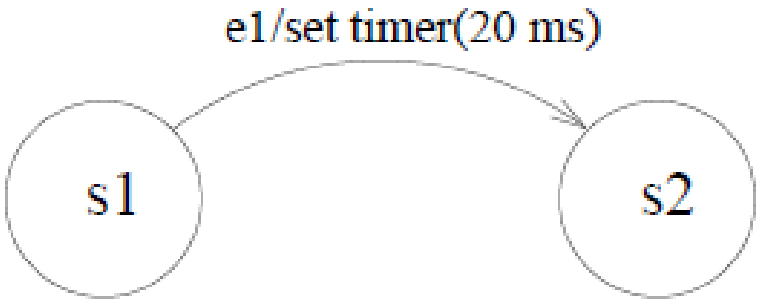
- Response-Stimulus (RS)
  - Here the deadline is on the production of response counted from the corresponding stimulus.
  - This is a behavioral constraint, since the constraint is imposed on the stimulus event.
  - Example: Once the dial tone appears, the first digit must be dialed within 30 seconds, otherwise the system enters an idle state and an idle tone is produced.

- Response-Response (RR)
  - Once the first response event occurs, the second response event must occur before a certain deadline.
  - This is a performance constraint, since the timing constraint has been defined on a response event.
  - Example: Once the ring tone is given to the callee, the corresponding ring back tone must be given to the caller within two seconds, otherwise the call is terminated

Modelling Timing Constraints

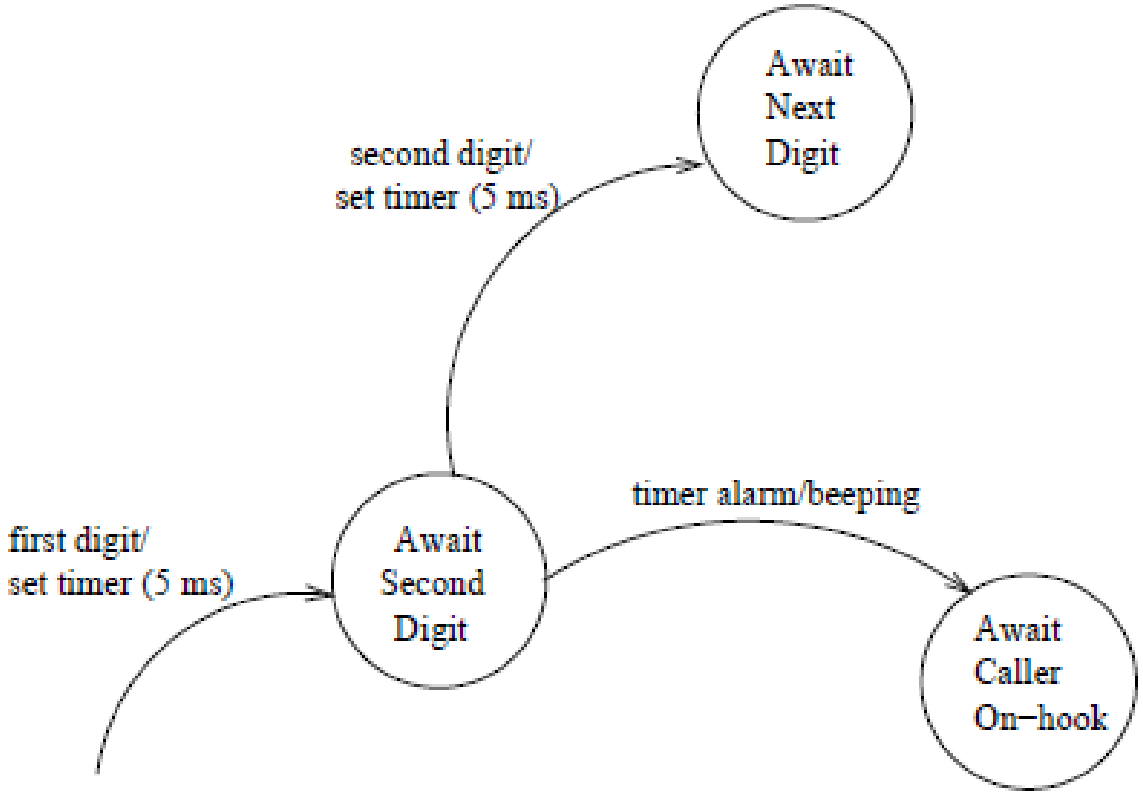
- Modelling time constraints is very important since once a model of the time constraints in a system is constructed, it can serve as a formal specification of the system
- Finite State Machines (FSMs) is a powerful tool which has long been used to model traditional systems.
- In an FSM model, at any point of time a system can be in any of the state.
- A state is represented by a circle.
- A state change is also called a state transition.
- A transition from one state to another is represented by drawing a directed arc from the source to the destination.

- Extended Finite State Machine (EFSM) is used to model time constraints.
- EFSM extends the traditional FSM by incorporating the action of setting a timer and the expiry event of a timer.

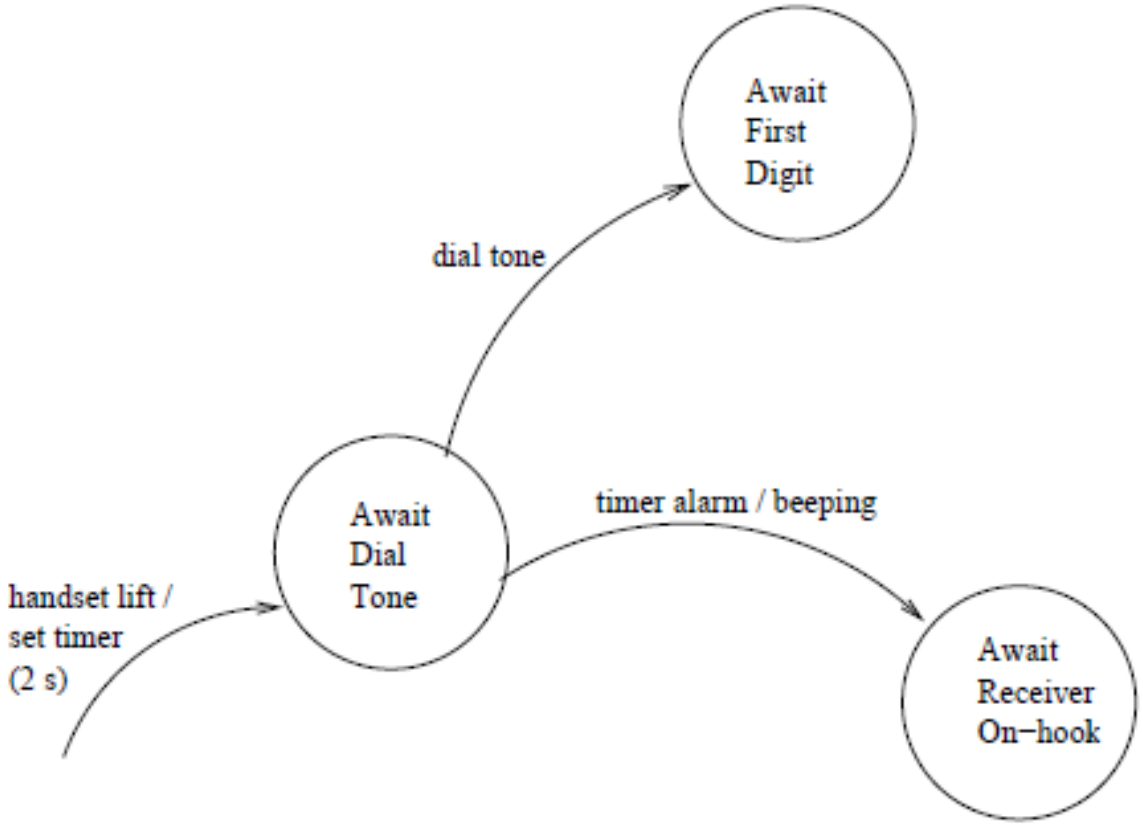


Stimulus Stimulus Constraints

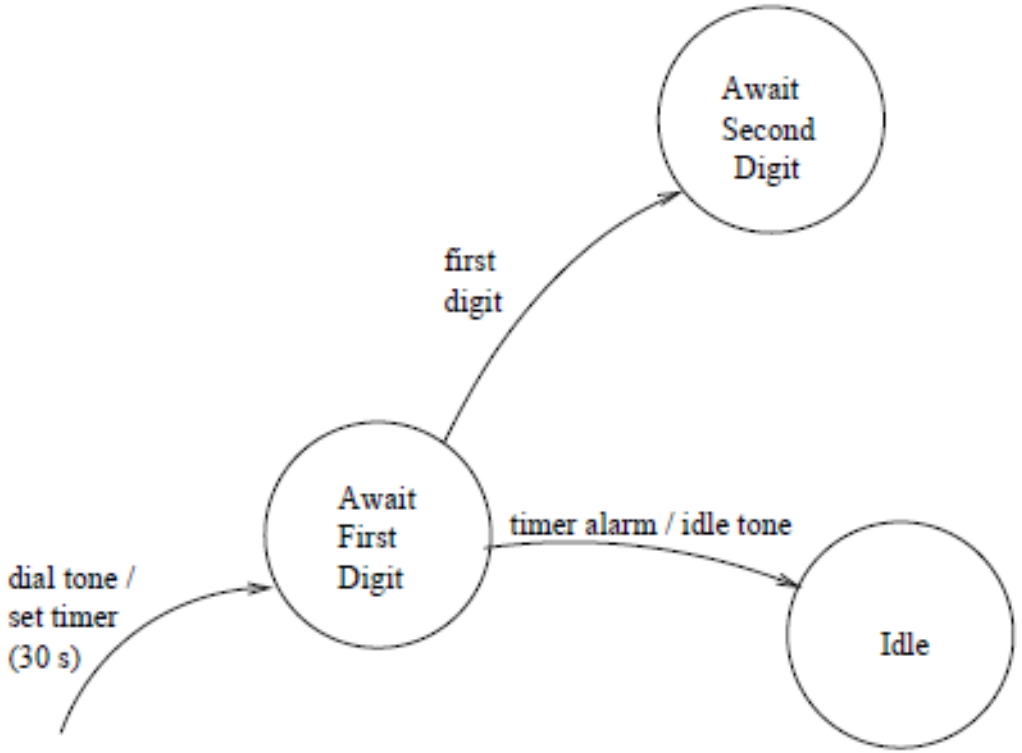
- Example: Once a user completes dialing a digit, he must dial the next digit within the next 5 seconds. Otherwise an idle tone is produced.



- Stimulus Response (SR)
- Ex: Once the receiver of the hand set is lifted, the dial tone must be produced by the system within 2 seconds, otherwise a beeping sound is produced until the handset is replaced.



- Response-Stimulus (RS)
- Example: Once the dial tone appears, the first digit must be dialed within 30 seconds, otherwise the system enters an idle state and an idle tone is produced.



- Response-Response (RR)
- Example: Once the ring tone is given to the callee, the corresponding ring back tone must be given to the caller within two seconds, otherwise the call is terminated