Introduction and Course Outline
Real-Time Systems Design (CS 6414)

Sumanta Pyne

Assistant Professor
Computer Science and Engineering Department
National Institute of Technology Rourkela
pynes@nitrkl.ac.in

December 23, 2020

What is a Real-time System ?

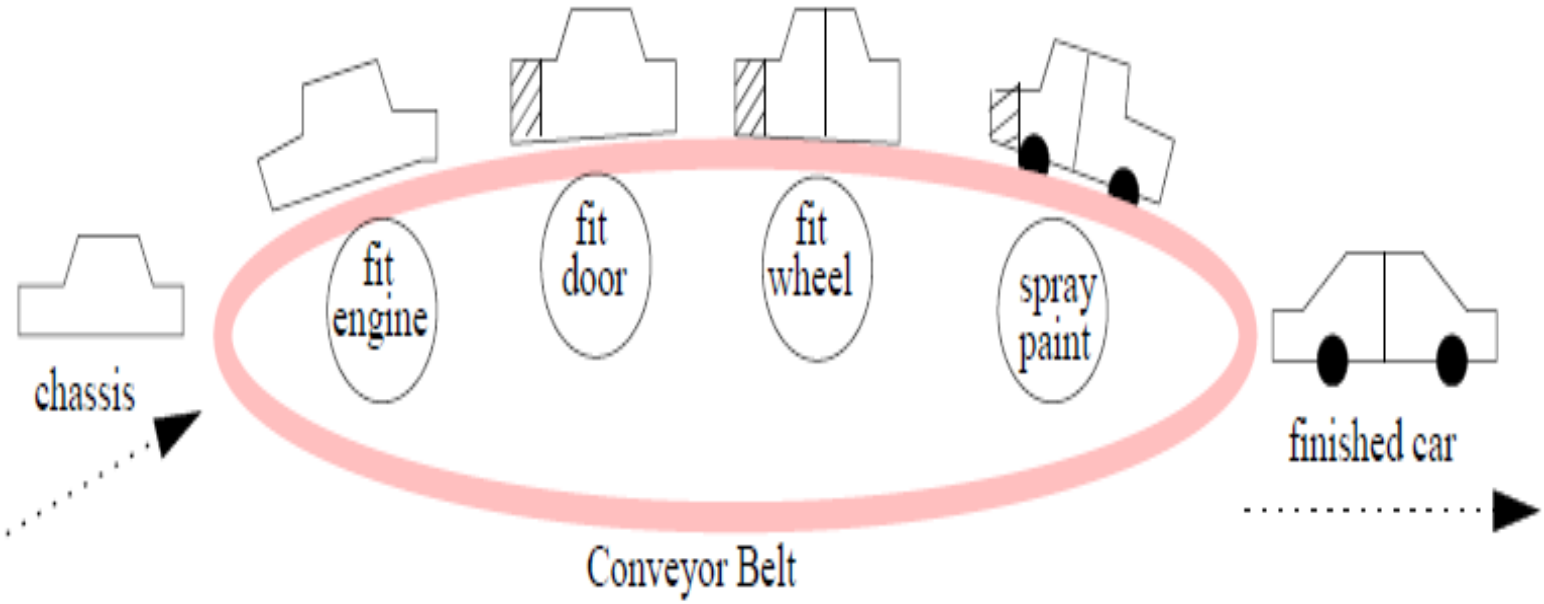
- A computer system that must
  - process information, or
  - complete task
- within a specified interval
- to avoid risk system failure
- Examples
  - aircraft
  - nuclear power plant
  - airline reservations

Industrial Appications

- Chemical Plant Control
  - Periodic monitoring of reaction chambers
  - temperature, pressure and chemical concentration
  - decide corrective actions necessary at that instant
  - time bound ranges from micro seconds to milli seconds

Industrial Appications

- Automated Car Assembly Plant



  - Each workstation performs a specific work
  - fitting engine, fitting door, fitting wheets and spray paint
  - time constraint - a workstation complete before moving to next
  - time bounds hundreds of few milli seconds

Industrial Appications

- Supervisory Control and Data Acquisition (SCADA)
  - collect raw data from sensors scattered
  - data processed and stored in real-time database
  - load balancing in electrical energy distribution and gas pipeline network
  - time bounds hundreds of few milli seconds

Medical Appications

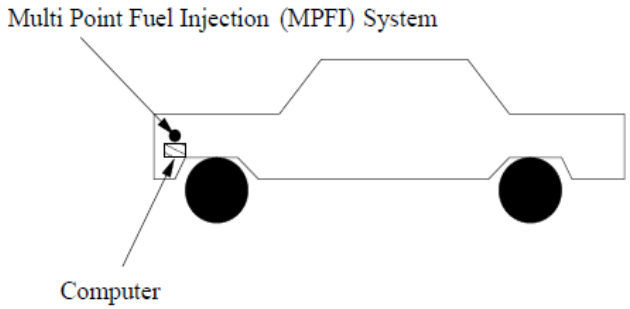
- Robot used in recovery of displaces radioactive material
  - Radioactive materials like Cobalt and Radmium are used to treat cancer
  - During treatment these materials get dislocated
  - Human beings cannot come near a radioactive material
  - A robot recovers it using a real-time path planning task
  - time constraint in order of a few milli seconds
- Cardiac pacmaker
- Blood glucose level monitoring system

Peripheral Equipments

- Laser Printer
  - Laser printers have embedded microprocessor
  - Microprocessor receives print commands
  - Determines how dots can be composed to achieve desired shapes
  - Manages to print by issuing dot commands to laser engine
  - Time contraints in order of a few milli seconds

Automotive and Transportation

- Multi-Point Fuel Injection (MPFI) System



  - MPFI is an automotive engine control system
  - Controls rate of fuel injection
  - Has a computer and sensors to monitor
    - ambient, coolant and exhaust temperatures
    - engine rpm and vehicle road speed
    - crankshaft and camshaft positions
    - emission gas contents
  - Determines an accurate fuel injection for given speed and acceleration
  - Minimize fuel consumption and pollution
- Traffic Light Controller

Telecommunication Applications

- A Cellular System
  - Call hand-off occur when mobile phone moves away from a base station (BS)
  - As mobile moves away its received signal strength (RSS) falls at BS
  - BS monitors the RSS
  - When RSS falls below a threshold value BS intiates hand-off
  - Passes on-going call details to BS of cell where mobile has moved
  - Hand-off delay should avoid temporary disruption in service
  - Achieved within a few milliseconds

Aerospace

- Auto pilot - Computer On-board an Aircraft
  - Controls aircraft - navigation, take-off and landing
  - Periodic sampling of velocity and acceleration
  - Computes current position (X,Y,Z) and compares with track data
  - Computes deviation and take corrective actions
  - Sampling and processing to be done in few microseconds

Internet and Multimedia Applications

- Video Conferencing
  - Camera produces video signals
  - Microphone produces audio signals
  - Signals sampled at a prespecified frame rate
  - Compressed and sent as packets over a network
  - At receiver-end, packets are ordered, decompressed and then played
  - Process and play received frames at predetermined constant rate
  - For display rate of 30 frames/minute
    - once a frame play-out is complete
    - next frame must be played within 2 seconds

Consumer Electronics

- Cell Phones
  - Multiple tasks simultaneously
    - Convert voice to digital signals
    - Convert electrical signals to voice signals
    - Respond to base station signals within specified time bounds
  - Base station cell phone specifies frequency for an on-going communication
  - Cell phone must comply within a few milliseconds

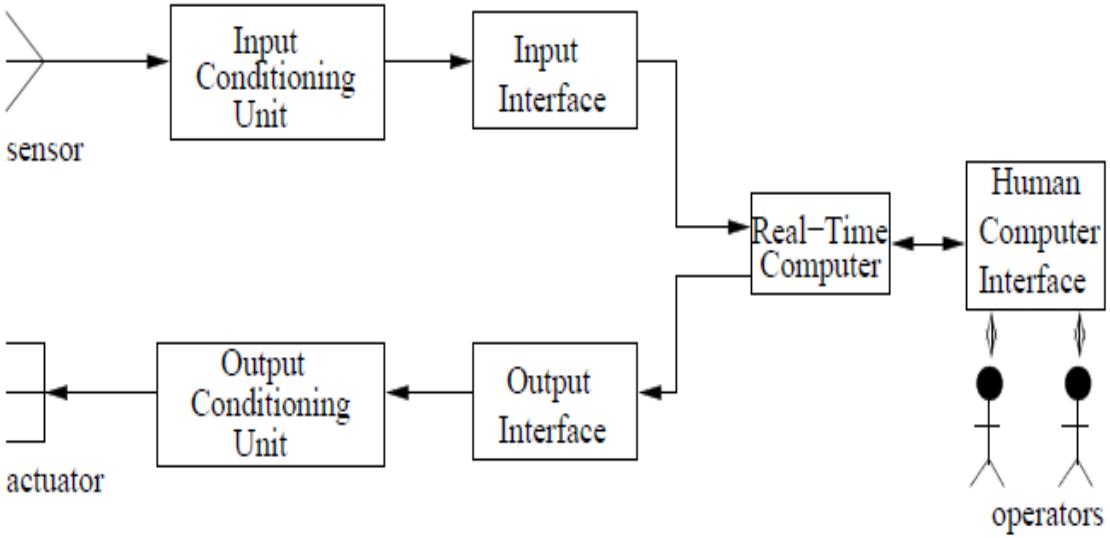
Defense Applications

- Guided Missile
  - Guidance by a computer on the missi
  - Compute deviation from the required trajectory
  - Effects track changes of missile to guide onto target
  - Target sensing and track correction depends on
    - speed of missile
    - type of target
  - Completed within a few hundreds of microseconds

Miscellaneous Applications

- Railway Reservation System
  - A central repository maintains up-to-date booking status
  - Ticket booking counters are distributed across different locations
  - Customers queue at different counters request reservation
  - Takes few seconds to
    - confirm reservation and print ticket, or
    - display seat unavailability message
  - before the average human response time (about 20 seconds) expires
  - Customers do not notice any delay

A Basic Model of a Real-Time System



- Underlying hardware in a real-time system
  - Sensor
    - Converts physical characteristics of environment to electrical signals
    - Light sensor – Photo-voltaic cell converts light to electrical energy
    - Temperature sensors – thermocouple, varistor
    - Pressure sensor – piezoelectricity
  - Actuator
    - Takes electrical signals from computer, converts to physical actions
    - Physical actions – motion, change of thermal, electrical, pneumatic, etc
    - Motor, heater, hydraulic and pneumatic actuators

A Basic Model of a Real-Time System

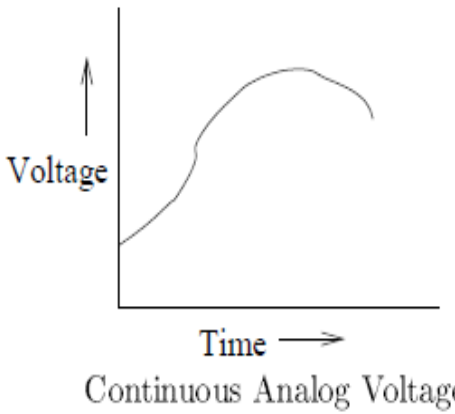
- Signal Conditioning Units
- Computer signals are conditioned before used by actuators
  1. Voltage Amplification
     - Match full scale voltages of sensor output and computer interface input
     - Sensors produce milli volts, computer input interface require volts
  2. Voltage Level Shifting
     - Align voltage level generated by sensor, make it computer acceptable
     - Sensor output -0.5 – +0.5V, computer interface input 0 – 1V
     - requires level shifting
  3. Frequency Range Shifting and Filtering
     - Reduce noise components in signal
     - Signal shifted from noise bands to filter out noise
  4. Signal Mode Conversion
     - Direct current to alternate current and vice-versa
     - Analog signal to constant amplitude pulse train
     - Constant amplitude pulse train - input for transformer coupled circuit

A Basic Model of a Real-Time System

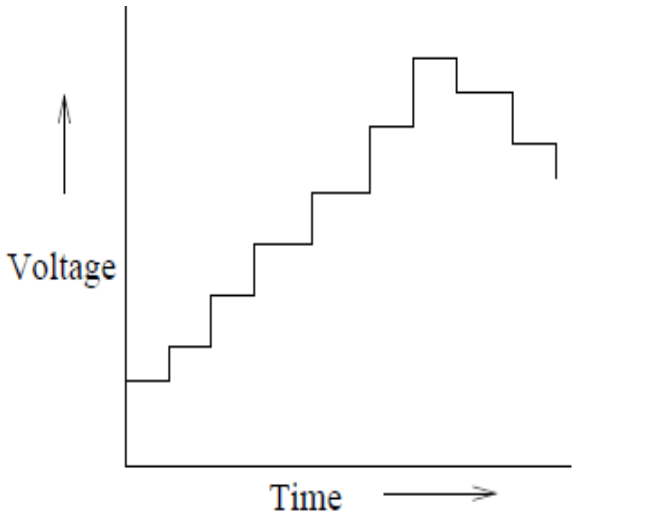
- Interface Unit
  - Output interface - CPU commands into analog, passes to actuator
  - CPU selects and writes in data register of output interface
  - Takes care of buffering and handshake control aspects
  - Analog to digital conversion in input interface
  - Digital to analog conversion in output interface

A Basic Model of a Real-Time System

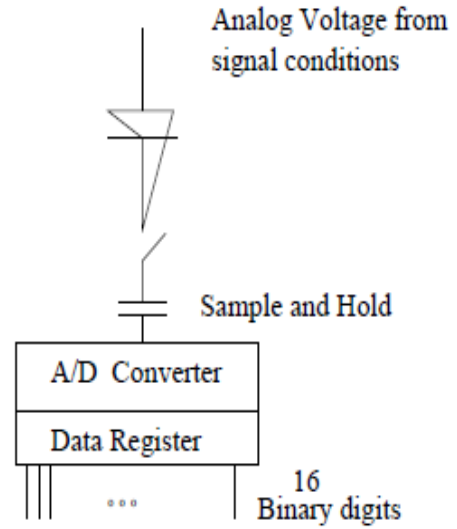
- Analog to Digital Conversion
  - Sample analog signal at regular intervals
  - Sampling using capacitor circuitry to store voltage levels
  - Stored voltage is discretized
  - Convert stored value to binary number using ADC



Continuous Analog Voltage



Analog Voltage Converted to Discrete Form



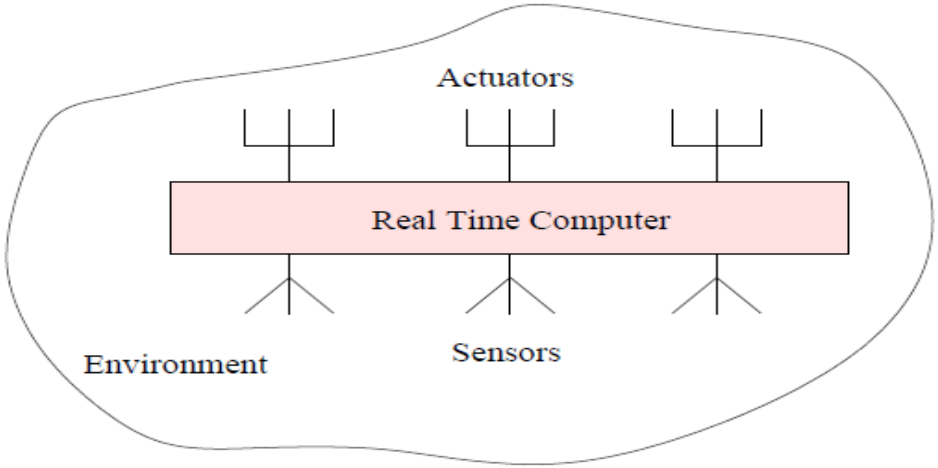
Conversion of an Analog Signal to a 16 bit Binary Number

- Digital to Analog Conversion
  - Complementary of analog to digital conversion

Characteristics of Real-Time Systems

- Time contraints
  - Task deadline - specifies time before, task must complete and produce
  - RTOS ensures all task meet time constraints
- New Correctness Criterion
  - Not only logical correctness of results
  - Time at which results are produced is important
  - Logically correct result produced after deadline is incorrect

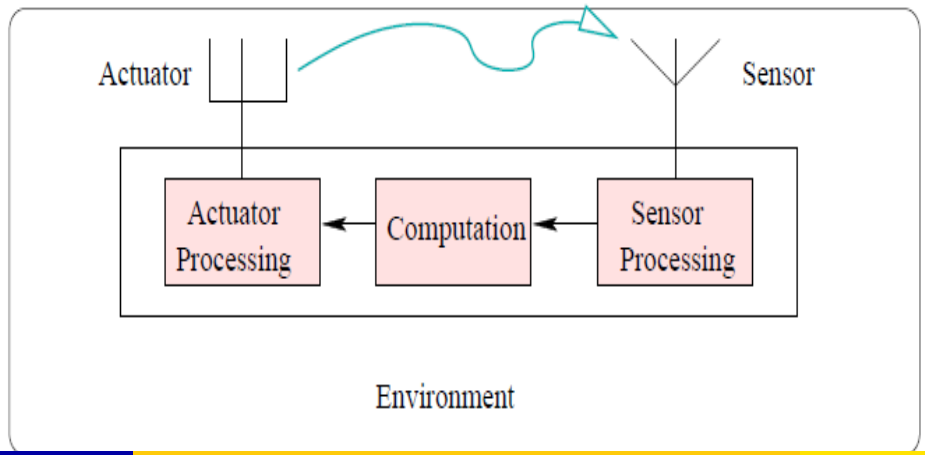
Characteristics of Real-Time Systems



- Embedded
  - Embedded computer system physically "embedded" in its environment
  - Control environment
  - Sensor of real-time computer collect data from environment
  - Computer processes data, pass information to actuators
  - Actuators carry necessary work on environment
  - An example - Multi-Point Fuel Injection (MPFI) system
- Safety-Criticality
  - A safe system does not cause any damage even wheen it fails
  - A reliable system can operate for long durations without any failure
  - A safety-critical system needs high reliability as failure cause damage

Characteristics of Real-Time Systems

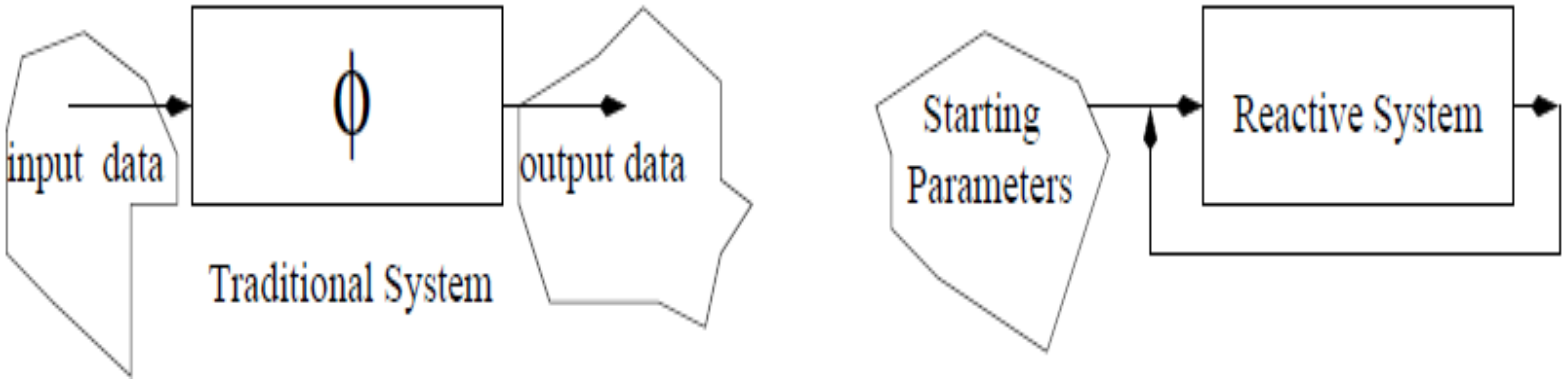
- Concurrency
  - Real-time system needs to respond within short and strict time bounds
  - Sensors may generate data asynchronously at different rates
  - Real-time system must process data concurrently to avoid malfunction
  - Non-deterministic - system behaviour depends on timing of inputs
  - Non-deterministic computation - different outputs for same input set
- Distributed and Feedback Structure
  - Real-time system components spread across geographic locations
  - Events handled locally to them prevent overloading of network
  - Sensors and actuators at different locations of event generation
  - Feedback - sensed data processed to determine corrective actions
  - Example - petroleum refinery plant



Characteristics of Real-Time Systems

- Task Criticality
  - Measure of cost of failure of a task
  - Determined by examining how critical are results produced by task
  - Criticalities of tasks considered for designing fault-tolerance
  - Higher criticality more reliablility to be added
  - Fast detection and recovery for failure of highly critical task
  - Task priority and task criticality are different concepts
- Custom Hardware
  - Real-time system implemented on specific hardware
  - Cell phones use power-efficient processors supporting its operations
  - MPFI engines require require 16- or 32-bit processors of 40 MHz

Characteristics of Real-Time Systems



- Reactive
  - Reactive system maintains ongoing computer-environment interaction
  - Real-time systems are often reactive
  - Ordinary systems compute functions on input to generate output
  - Reactive systems do not produce any output but enter into on-going interaction with environment
  - In each interaction step, results carry out action on environment
  - Reaction of environment is sampled and fed back to system
  - Computations of real-time system is non-terminating

Characteristics of Real-Time Systems

- Stability
  - Under overloading real-time systems meet critical task deadline
  - Non-critical tasks may not meet deadline
- Exception Handling
  - Real-time systems work round-the-clock without human interaction
  - Taking corrective actions on a failure is difficult
  - No immediate corrective action ensure failure with no catastrophe
  - Failure must be detected but system operation must continue

Safety and Reliability

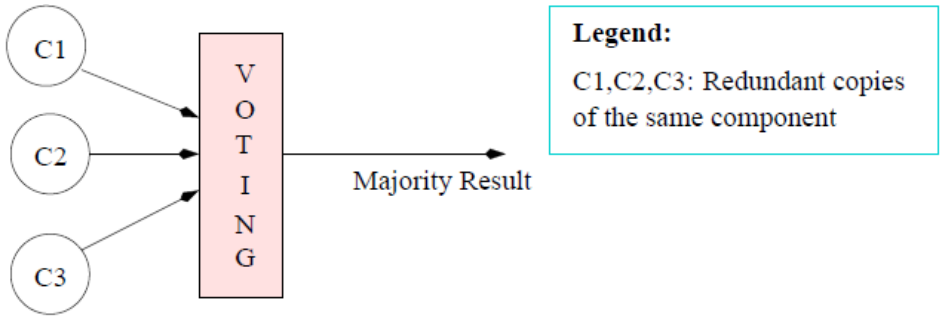
- In traditional systems safety and reliability are independent issues
- Safe and unreliable systems
  - Failure does not cause significant damage or financial loss
  - Word processing software
- Unsafe and reliable
  - Hand gun rarely fails - reliable
  - Unsafe if gun fails - misfire or explode cause damage
- In real-time systems safety and reliability coupled together
- A **fail-safe** state of a system - state at system fails with no damage
  - Fail-safe state of word processing program saves document on disk
- A **safety-critical** system - failure can cause severe damage
  - Navigation system on-board an aircraft has no fail-safe states
  - On-board computer fails, no fail-safe for engine switched-off
  - Absence of fail-safe state $\implies$ safety ensured by higher reliability
  - Any failure of aircraft controlling computer is not acceptable
  - Standard reliability of aircraft computer - 1 failure per 10^9 flying hours

How to Achieve High Reliability?

- Error Avoidance
  - Minimize every possibity of error occurrence during development
    - using well-founded software engineering practices
    - using sound design methodologies
    - adopting suitable CASE tools
- Error Detection and Removal
  - Conducting thorough reviews and testing
- Fault-Tolerance
  - Entirely error-free practical software system is virtually impossible
  - Few errors persists after reviews and testing
  - Errors cause failures
  - System should tolerate faults work correctly even when error occurs
  - Fault-tolerance achieved by incorporating redundancy

Hardware Fault-Tolerance Techniques

- Built in Self Test (BIST)
  - System periodically performs self tests of its components
  - Reconfigures itself on failure detection
  - Switching-out faulty component, switching-in alternate

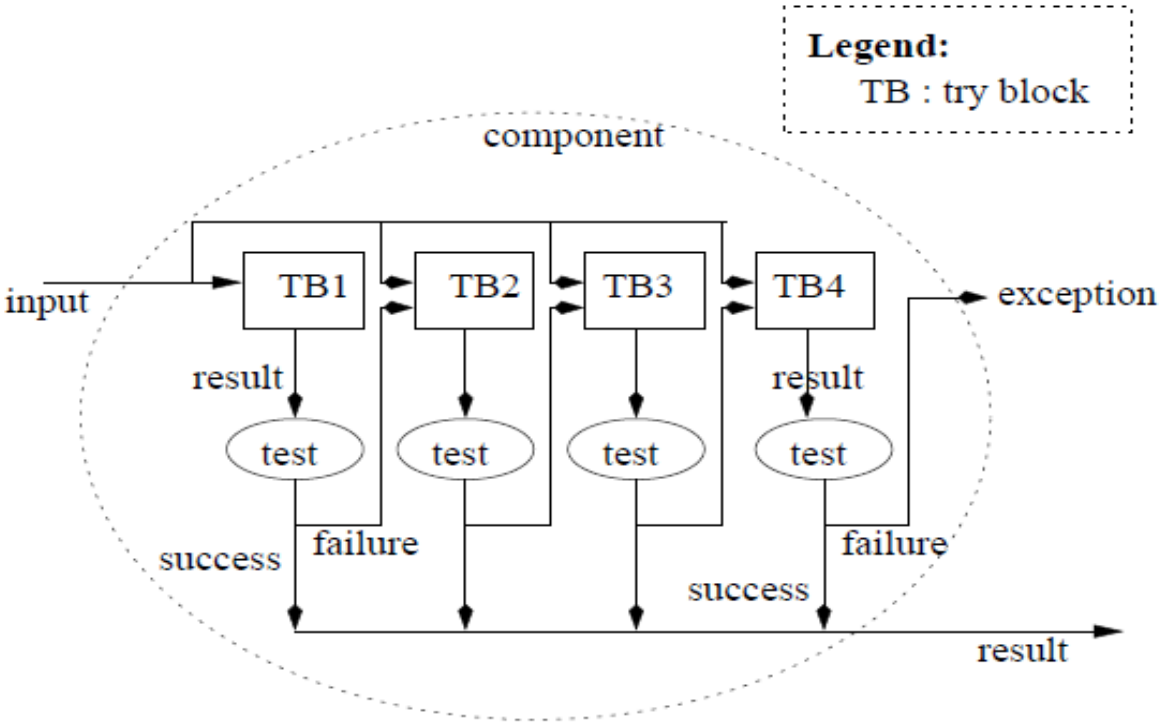


- Triple Modular Redundancy (TMR)
  - Performs voting of results produced by redundant components
  - Select majority result
  - Can tolerate occurrence of single failure any time
  - Assumption - at any time only one component out of three fails
  - Majority erroneous if two or more fail simultaneously
  - Majority likely to fail requires greater amount of redundancies
  - At least 2n+1 components to tolerate simultaneous failure of n

Software Fault-Tolerance Techniques

- N-Version Programming
  - Adaptation of TMR technique for hardware fault-tolerance
  - Independent teams develop N different versions of software component
  - N depends on degree of fault-tolerance required
  - Redundant modules are run concurrently on redundant hardware
  - Result of different versions subjected to voting at run time
  - Result agreed by majority is accepted
  - Independent teams commit different mistakes, eliminated by voting
  - Statistical correlation of failures
    - different versions of a component show similar failure patterns
    - modules developed independently contain identical errors
    - programmers commit error in part of a problem difficult for all teams
    - identical errors - most complex and least understood

Recovery Blocks

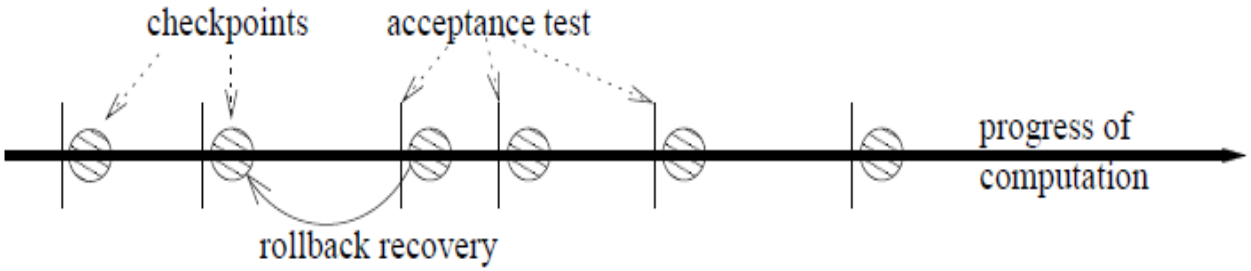


A Software Fault-Tolerance Scheme Using Recovery Blocks

Software Fault-Tolerance Techniques

- Recovery Blocks
  - Redundant components are called **try blocks**
  - Different algorithms are used in different try blocks
  - Each try block computes same result from other try blocks
  - Try blocks are run one after another
  - Results from try block go for acceptance test
  - If test fails, then next try block is tried
  - Repeated until a try block result successfully passes acceptance test
  - Statistical correlation of failures
    - different try blocks fail for identical reasons
  - Can be used if task deadlines are larger than task computation times
    - since different try blocks are executed one after another on failure
    - difficulty real-time tasks with very short slack time
    - as try blocks tried one after another may miss deadlines
    - in such cases later try blocks contain only skeletal code
    - try blocks with skeletal codes produce approximate results in less time

Software Fault-Tolerance Techniques



Checkpointing and Rollback Recovery

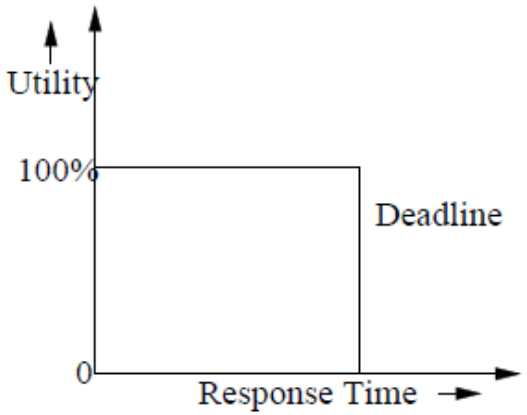
- Checkpointing and Rollback Recovery
  - System state is tested each time after progress in computation
  - After a state-check test succeeds, system state backed up on storage
  - If test does not succeed, system roll-backs to last checkpointed state
  - After a rollback, a fresh computation is initiated
  - Useful during system state corruption
    - data corruption or
    - process failure

Types of Real-Time Tasks

- A real-time task is either hard, soft, or firm
- Depending on consequences of a task missing it deadline
- Hard Real-Time Tasks
  - Constrained to produce result within predefined time bounds
  - Fails when any hard real-time task does produce result before deadline
  - A robot has hard real-time tasks
    - Cyclically communicates, senses-detects obstacles, plan paths, moves
    - Sudden obstacle must be detected fast to prevent collision
    - Detection task has to completed before time bound
    - Otherwise collision occurs - robot failure
    - Detecting obstacles and reacting to it are hard real-time tasks
  - An anti-missile system has hard real-time tasks
    - First detect all incoming missiles
    - Properly position the anti-missile gun
    - Then fire to destroy incoming missile
    - All tasks are hard-real time
    - Fails if any task fails to complete before corresponding deadlines

Types of Real-Time Tasks

- Hard real-time tasks
  - Safety-critical - defense, medical, industrial, robotics
  - Computer games may have hard real-time tasks not safety-critical
  - Time bound ranges from several microseconds to a few milliseconds
  - Not necessary to complete in shortest time possible
  - Must complete within specified time bound



Utility of Result of a Firm Real-Time Task with Time

- Firm Real-Time Tasks
  - Produce results before predefined deadline
  - If not completed before deadline, no system failure
  - Late results are discarded
  - Utility of results become zero after deadline

Firm Real-Time Tasks

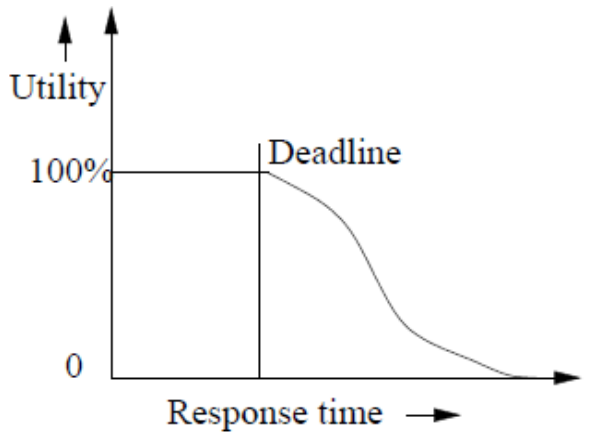
- Video conferencing
  - Video frames with accompanying audio sent over network as packets
  - Some frames may be delayed due congestion at different network nodes
  - Result varying queueing delays packets experience in different routes
  - While playing a frame preceding frame discarded on arrival
  - Frame delayed beyond time bound discarded at receiving-end
- Satellite-based tracking of enemy movements
  - Frames of satellite picture of enemy territory sent to ground station
  - Each frame is processed to locate objects of interest wrt previous frame
  - Determines movements of enemy
  - Ground computer is overloaded new frame received before older ones
  - Older images are of not much use and hence discarded

Soft Real-Time Tasks

- Time constraints are expressed in terms of average response times
- Web browsing is a soft real-time task
  - An URL click fetches and displays web page in 2 seconds on average
  - Several minutes to display is not failure but degrade performance
- A request for a seat in railway reservation system
  - Response for a request should occur in 20 seconds on average
  - Response in form of printed ticket or unavailability message
  - At least 95% requests ticket processed and printed in < 20 sec.

Impact of failure of soft-real time task to meet its deadline

- Railway reservation task
- If ticket is printed in 20 seconds $\Longrightarrow$ system working fine
- Missed deadlines do not result system failure
- Utility of results fall continuously with time after expiry of deadline
- Utility 100% if result produced before deadline, otherwise utility falls
- Soft real-time tasks in practical applications time bound range
  - fraction of second to few seconds



Utility of the Results Produced by a Soft Real-Time Task As A Function of Time

Non-Real-Time Tasks

- Not associated with any time bounds
- Presently most of interactive computations are soft real-time tasks
- Earlier non-interactive computers, non-real-time tasks
- Batch processing jobs, e-mail, back ground tasks
- Non-real-time task time bounds in orders few minutes, hours, days
- Soft-real-time task time bounds are at most order of few seconds

Timing Constraints

- Correctness of real-time tasks depend on both
  - logical correctness, and
  - satisfaction of timing constraints
- Timing constraints apply to certain events in a system
- Event of activation of a motor
- Characterize events in a system to understand timing behaviour
- Events in a Real-Time System
  - An event may be generated by system or its environment
  - Stimulus Events
    - Generated by environment and act on system
    - Produced asynchronously (aperiodically)
    - User pressing button of telephone set, generates stimulus event
    - Also generated periodically
    - Periodic sensing of temperature of nuclear reactor

Events in a Real-Time System

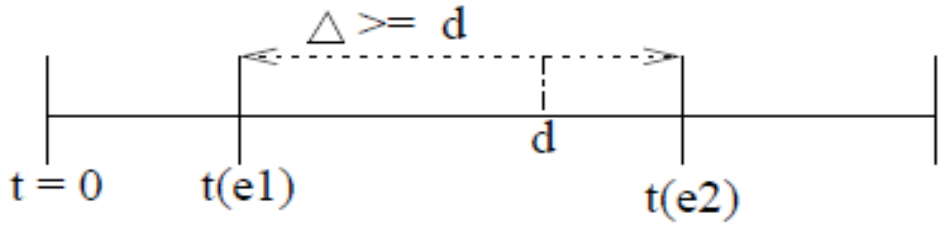
- Response Events
  - Produced by system in response to stimulus events
  - Act on environment
  - Chemical plant temperature $> 100°C$, switch off heater
  - Event of temperature exceeding $> 100°C$ is stimulus
  - Switching off heater is response
  - Response events - periodic or aperiodic
- Event - instantaneous or have certain duration
- Button press event - duration of button pressed
- Duration is combination of start and end events
- Button press event - start button press and stop button press

Classification of Timing Constraints

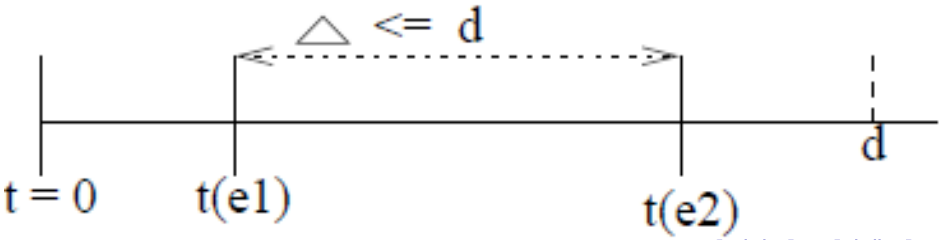
- Timing constraints in real-time system - performance and behavioral
- Performance constraints imposed on response of system
- Behavioral constraints imposed on stimuli generated by environment
- Behavioral constraints ensure environment of a system is well behaved
- Performance constraints - computer system performs satisfactorily
- Types of performance and behavioral constraints
  - Delay Constraint
  - Deadline Constraint
  - Duration Constraint

Performance and Behavioral Constraints

- Delay Constraint
  - Captures minimum time must elapse between occurence of two events
  - Events e1 and e2
  - Delay violation - After e1 occurs, if e2 occurs earlier than min delay
  - A delay constraint on e2: t(e2) - t(e1) $\geq$ d
  - t(e2) and t(e1) are time stamps, d minimum delay specified from e2

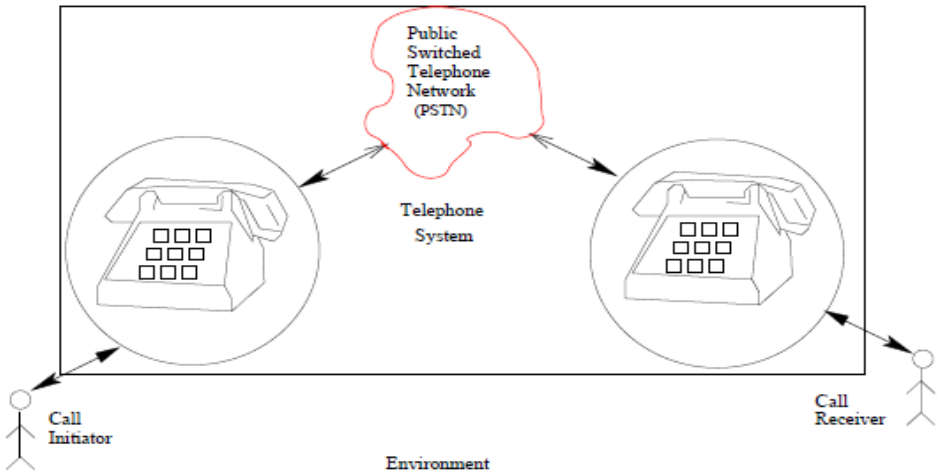


- Deadline Constraint
  - Captures permissible maximum separation between e1 and e2
  - e2 must follow e1 within permissible maximum separation time
  - e2 must occur within d time units of e1's occurrence
  - Deadline constraint: t(e2) - t(e1) $\leq$ d



Performance and Behavioral Constraints

- Deadline and delay constrains classified based on
  - constraint imposed on stimulus or response event
- Duration Constraint
  - Specifies the time period over which the event acts
  - Can be of minimum or maximum type
  - Minimum type - once event starts, must not end before min duration
  - Max type - once event starts, must end before max duration elapses
- Examples of Different Types of Timing Constraints



  - PSTN as computer system, users as environment

Examples of Different Types of Timing Constraints

- Deadline constrains
  - Stimulus-Stimulus(SS):
    - Deadline is defined between two stimuli
    - Behavioral constraint imposed on second event which is stimulus
    - Once a user completes dialing a digit, must dial next in 5 sec.
    - Otherwise an idle tone is produced
  - Stimulus-Response(SR):
    - Deadline is defined on response, measured from occurence of stimulus
    - Performance constraint imposed on a response event
    - Once receiver of handset is lifted, dial tone produced within 2 sec.
    - Otherwise beeping sound until handset replaced
    - Lifting receiver handset is stimulus, production of dial tone is response

Deadline constrains

- Response-Stimulus(RS):
  - Deadline on production of response counted from stimulus
  - Behavioral constraint imposed on a stimulus event
  - Once dial tone appears, first digit must be dialled within 30 seconds
  - Otherwise system in idle state, idle to is produced
- Response-Response(RR):
  - Deadline is defined on response of two events
  - Once first response event occurs, second must occur before deadline
  - Performance constraint - timing constraint defined on response event
  - Once ring tone given to callee, ring back tone to caller within 2 sec.
  - Otherwise call is terminated
  - Ring back tone and corresponding ring tone are two response events

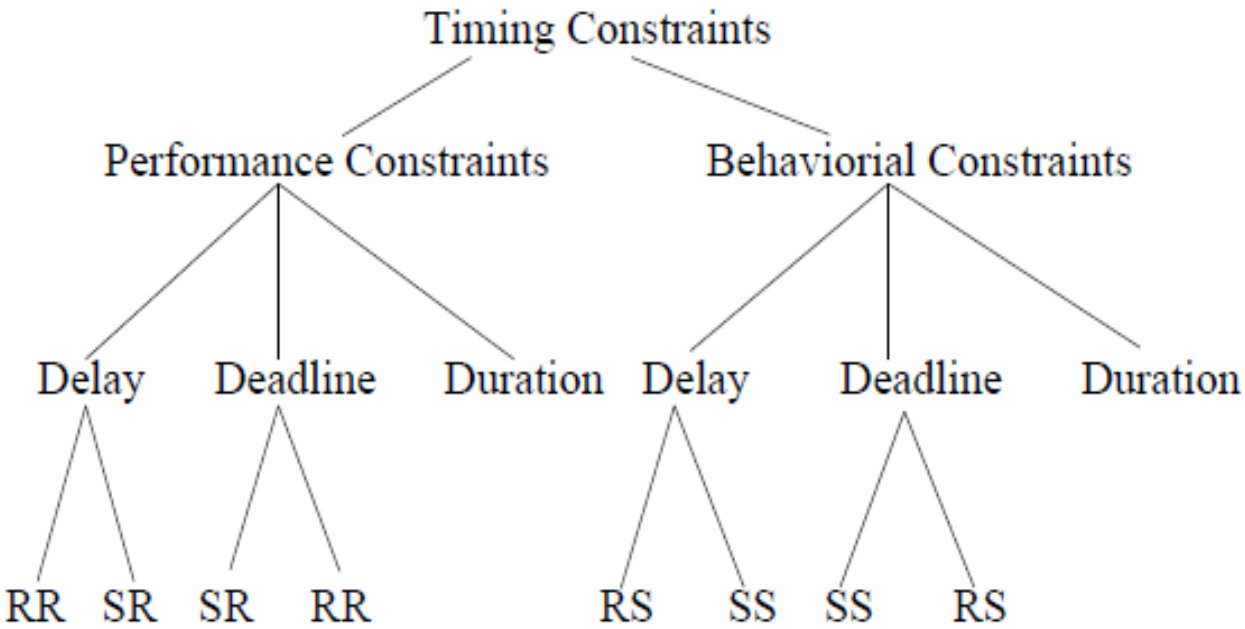
Delay Constraint

- Different types of delay constraints in various problems
- SS type delay constraint is in telephone system
- SS type delay constraint is behavioral
- Once a digit dialled, next digit dialled after at least 1 second
- Otherwise, beeping sound until call initiator replaces handset
- Delay constraint on event of dialing next digit after digit is dialled
- Both events are stimulus

Duration Constraint

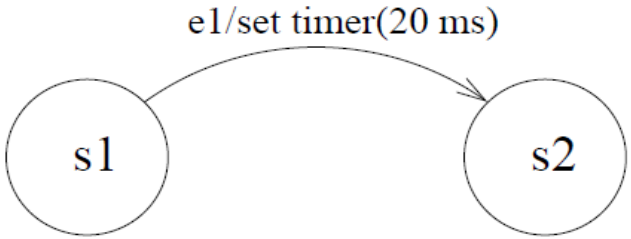
- Specifies time interval over which an event acts
- If handset button pressed $<$ 15 seconds, connect to local operator
- If button pressed 15-30 seconds, connect to international operator
- If button pressed $>$ 30 seconds, produce ring to on release

Classification of Time Constraints



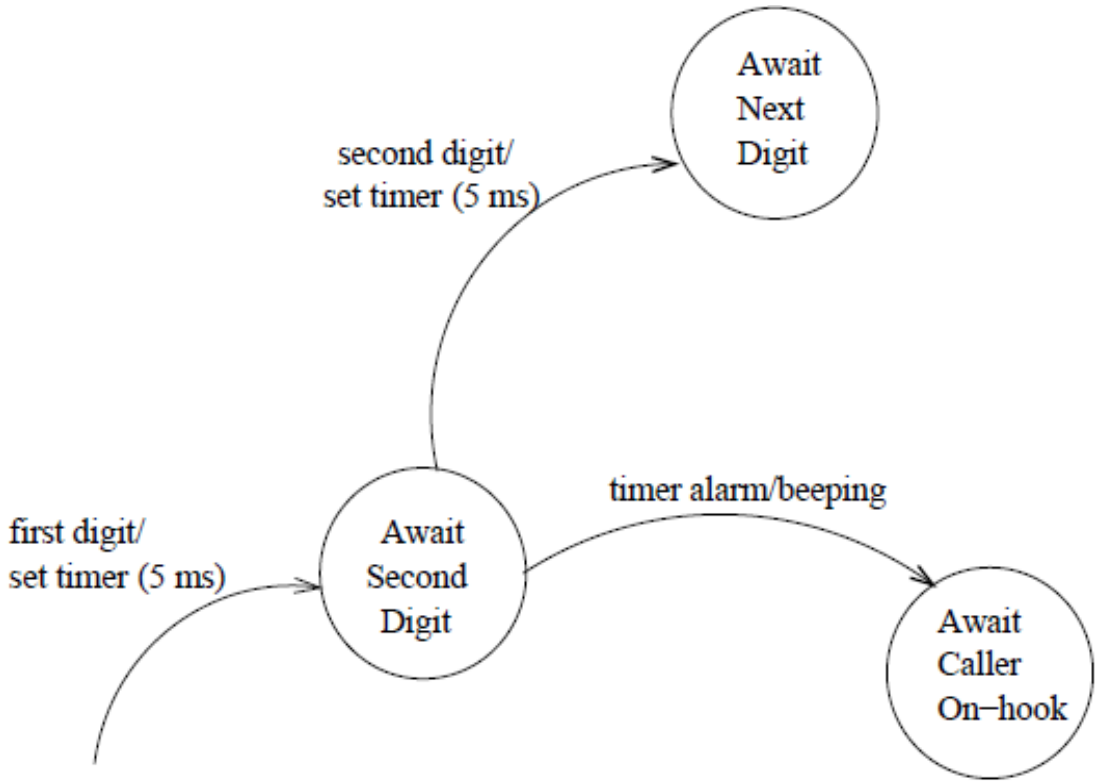
Modelling Timing Constraints

- Formal specification of a system
- If modelled correctly, automatic generate code
- To verify and understand a real-time system
- Based on Finite State Machines (FSM)
- States of an elevator are move up, move down and stationary
- Extended Finite State Machine (EFSM) to model time constrains
- EFSM - FSM with action of setting and expiry events of a timer



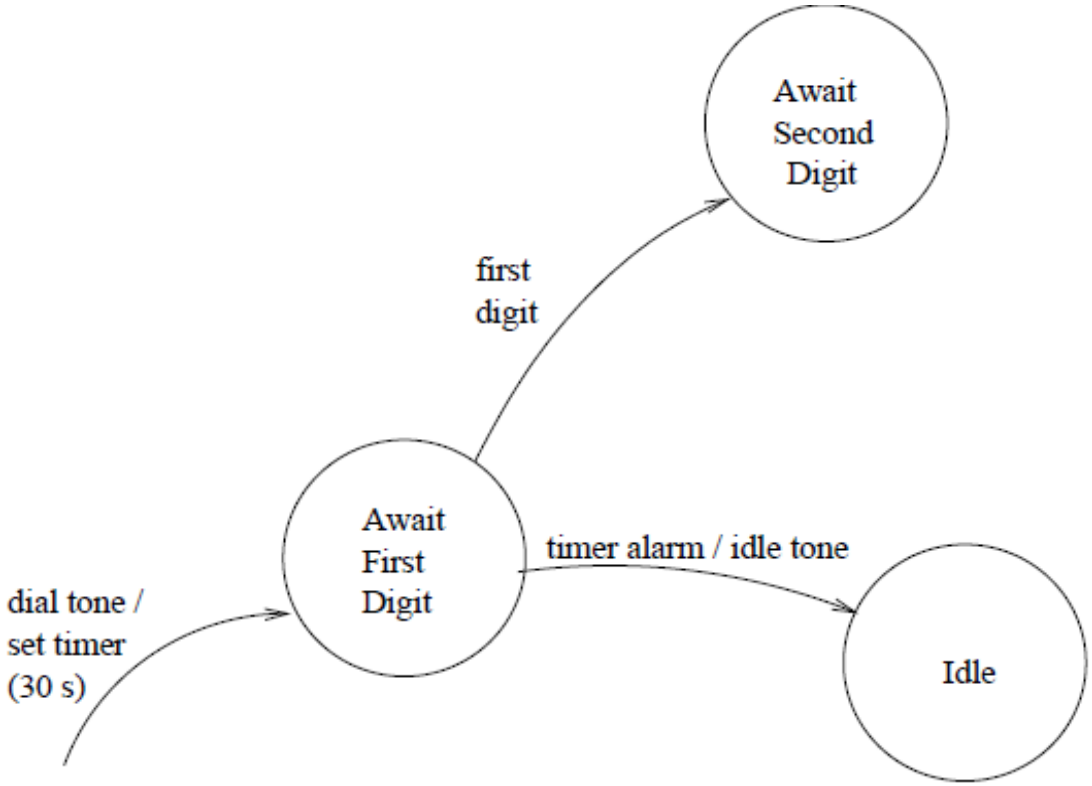
- At current state s1, event e1 occurs timer expire, transit next state s2

Stimulus-Stimulus (SS) deadline constraints



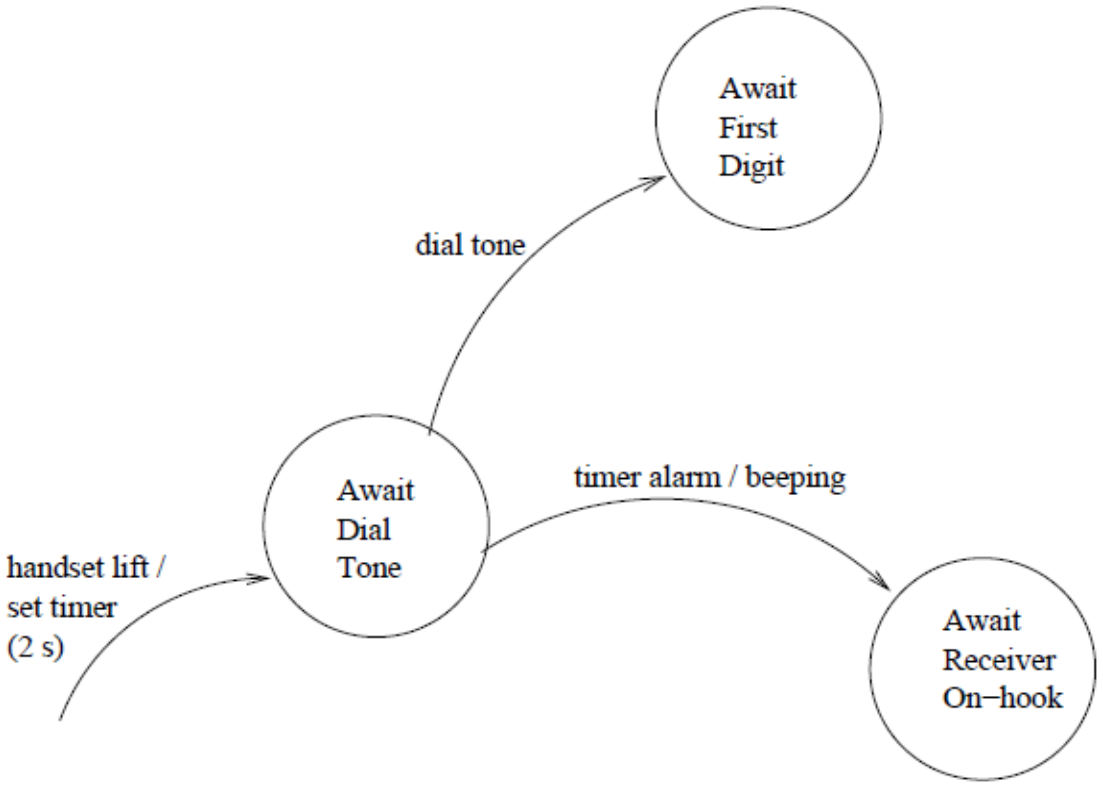
- Once first digit dialled, next digit to be dialled within next 5 msec

Response-Stimulus (RS) deadline constraints



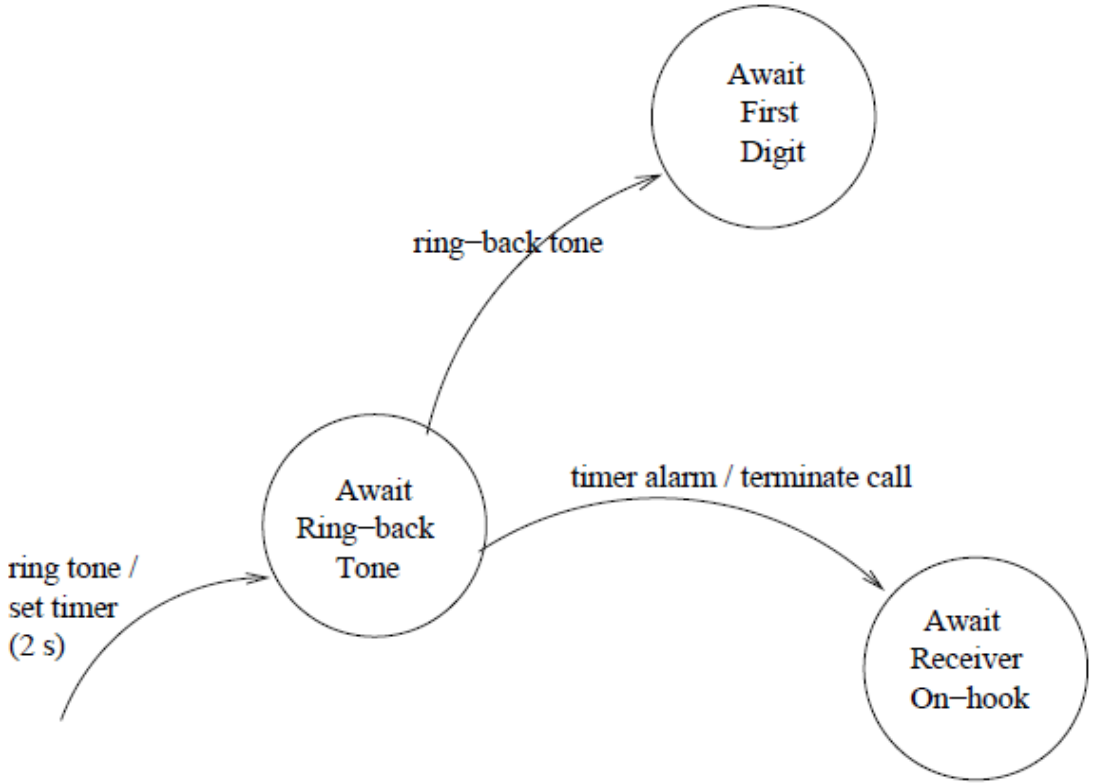
- Once dial tone appears, first digit must be dialled within 30 seconds
- Otherwise system enters idle state and idle tone produced

Stimulus-Response (SR) deadline constraints



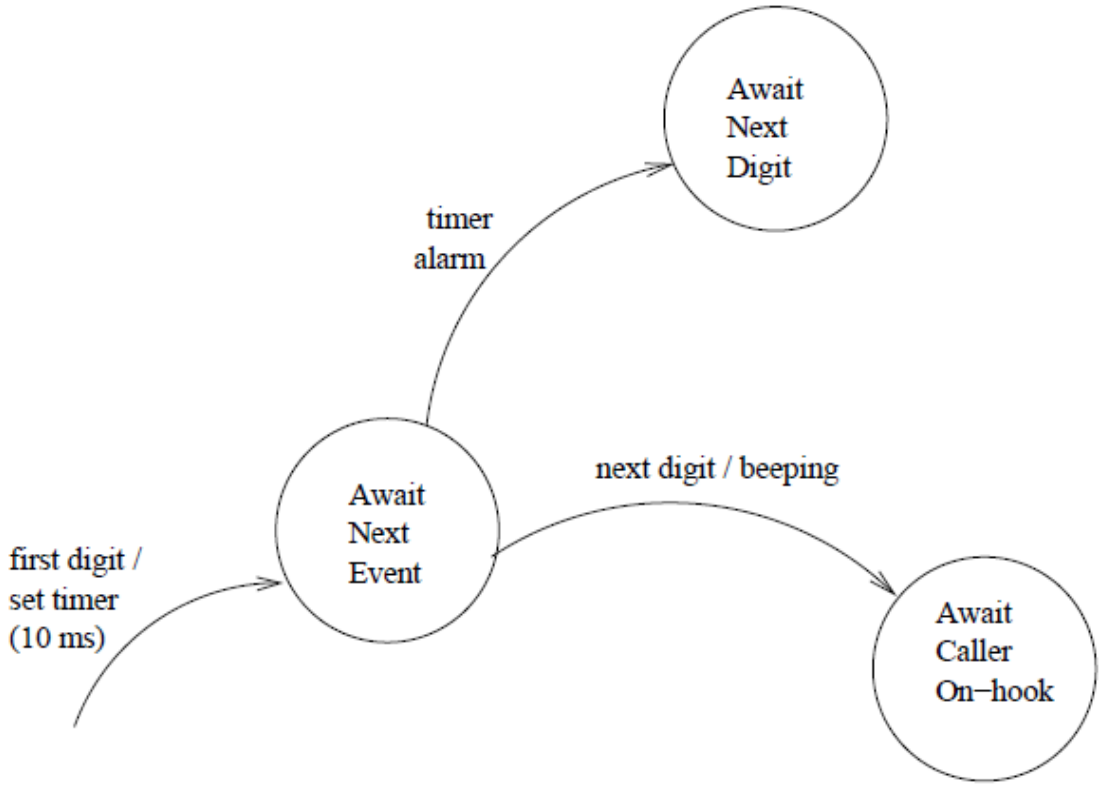
- Once receiver lifted, dial tone must be produced within 20 seconds
- Otherwise beeping sound produced until handset replaced

Response-Response (RR) deadline constraints



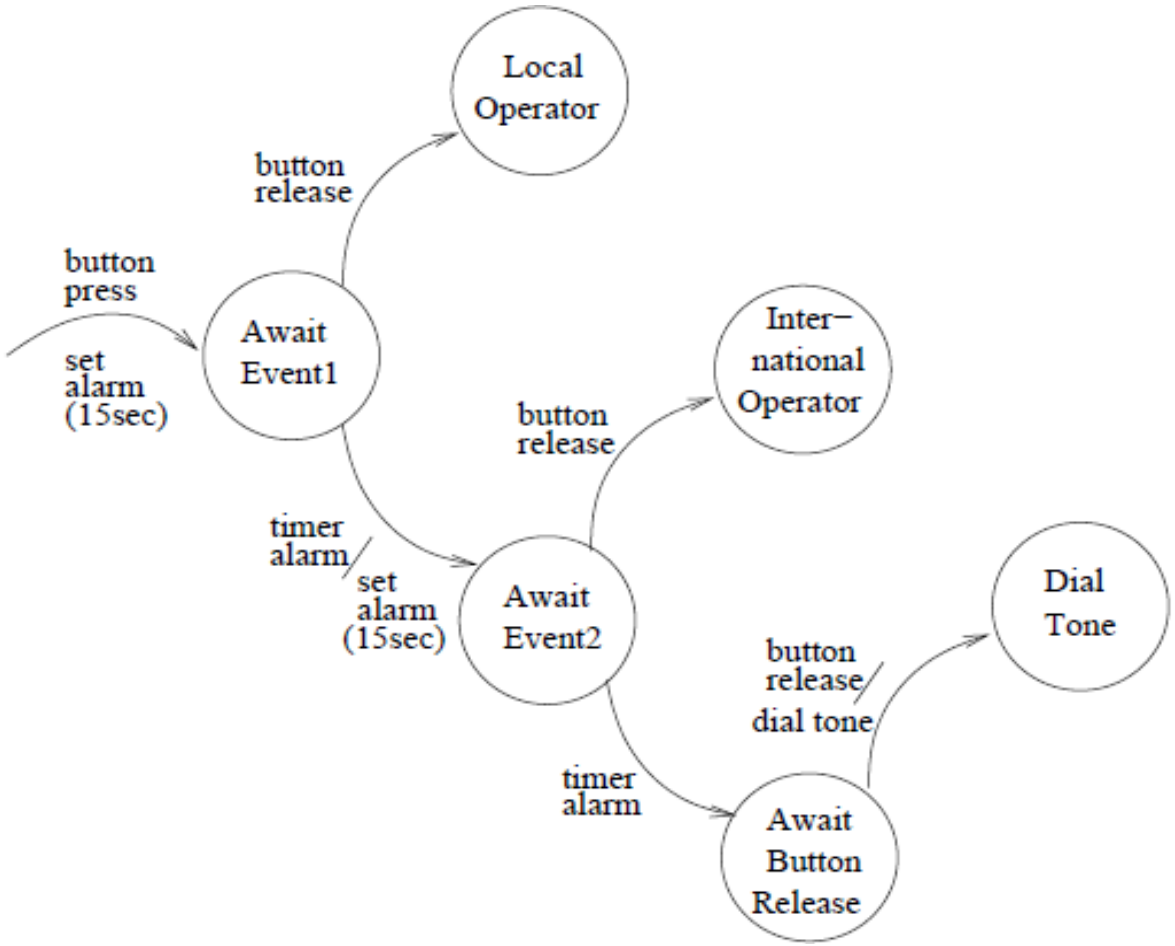
- Once ring tone given to callee, ring back tone to caller in 2 sec
- Otherwise call terminated

Stimulus-Stimulus (SS) delay constraints



- After a digit dialled, next digit must be dialled after 10 msec

Model of a Durational Constraint



- If handset button pressed $<$ 15 seconds, connect to local operator
- If button pressed 15-30 seconds, connect to international operator
- If button pressed $>$ 30 seconds, produce ring to on release

Exercises

- Represent a wash-machine having following specification by means of an EFSM diagram. The wash-machine waits for the **start** switch to be pressed. After the user presses the **start** switch, the machine fills the wash tub with either hot or cold water depending upon the setting of the **HotWash** switch. The water filling continues until the high level is sensed. The machine starts the agitation motor and continues agitating the wash tub until either the present timer expires or the user presses the **stop** switch. After the agitation stops, the machine starts the hot air blower and continues blowing hot air into the drying chamber until either the user presses the **stop** switch or the preset timer expires.

Exercises

- Represent the timing constrains in a collision avoidance task in air surveillance system as an EFSM diagram. The collision avoidance task consists of the following activities
  - The first subtask named radar signal processor processes the radar signal on a signal processor to generate the track record in terms of the target's location and velocity within 100 msec of receipt of the signal.
  - The track record is transmitted to the data processor within 1 msec after the track record is determined.
  - A subtask on the data processor correlates the received track record with the track records of other targets that come close to detect potential collision that might occur within the next 500 msec.
  - If a collision is anticipated, then the corrective action is determined within 10 msec by another subtask running on the data processor.

Thank you