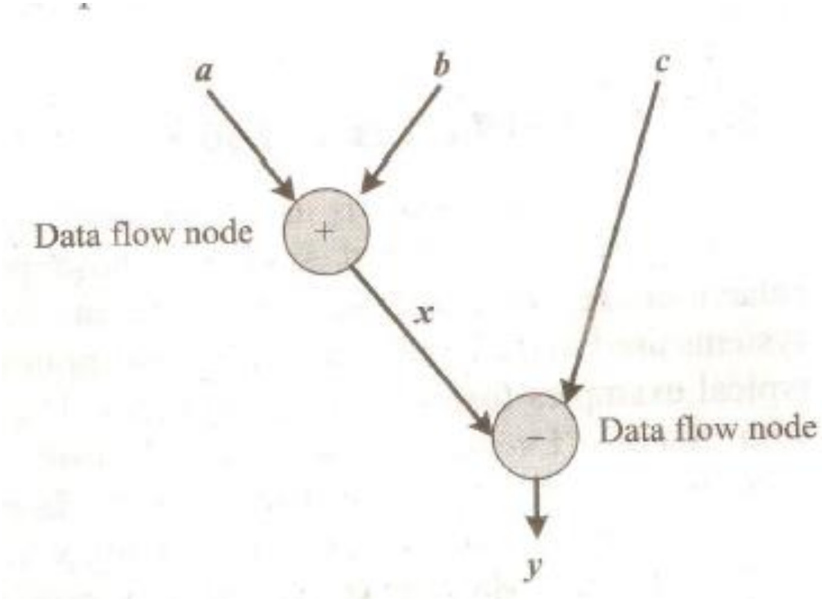
PROGRAM MODELS

- A model is a formal system consisting of objects and composition rules.
- In hardware software co-design, models are used for capturing and describing the system characteristics.
- Most often designers switch between a variety of models from the requirements specification to the implementation aspect of the system design.

DATA FLOW GRAPH (DFG) MODELS

- The DFG model translates the data processing requirements into a data flow graph.
- Operation on the data(process) is represented using a block (circle) and dataflow is represented using arrows.
- Embedded applications which are computational intensive and data driven are modelled using DFG



Control Data Flow Graph (CDFG) model

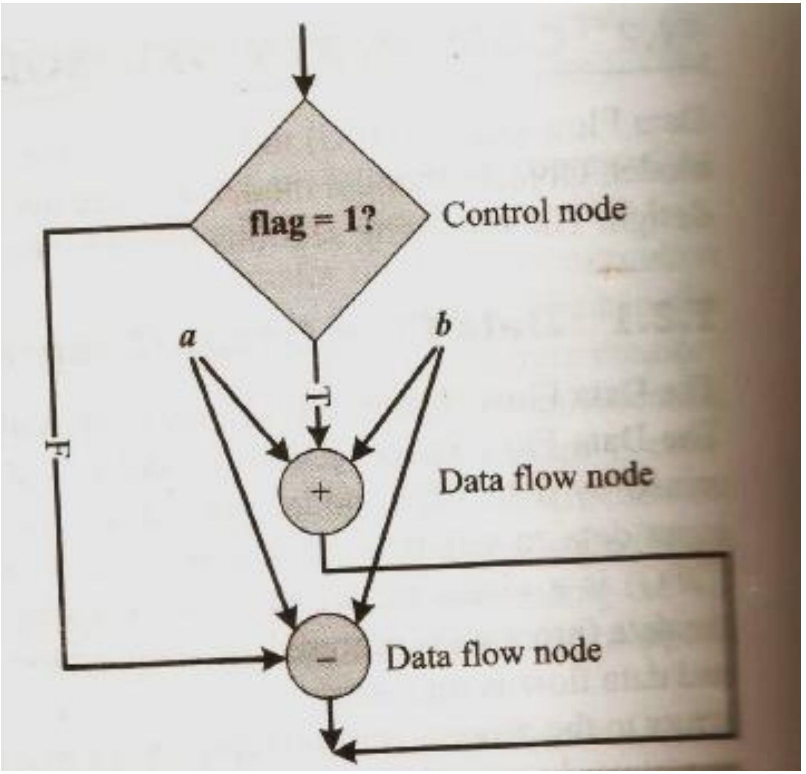
- The CDFG model is used for modelling applications involving conditional program execution.
- The CDFG models contain both data operations and

control operations.

- The control node is represented by a ‘Diamond’ block

which is the decision making element in a

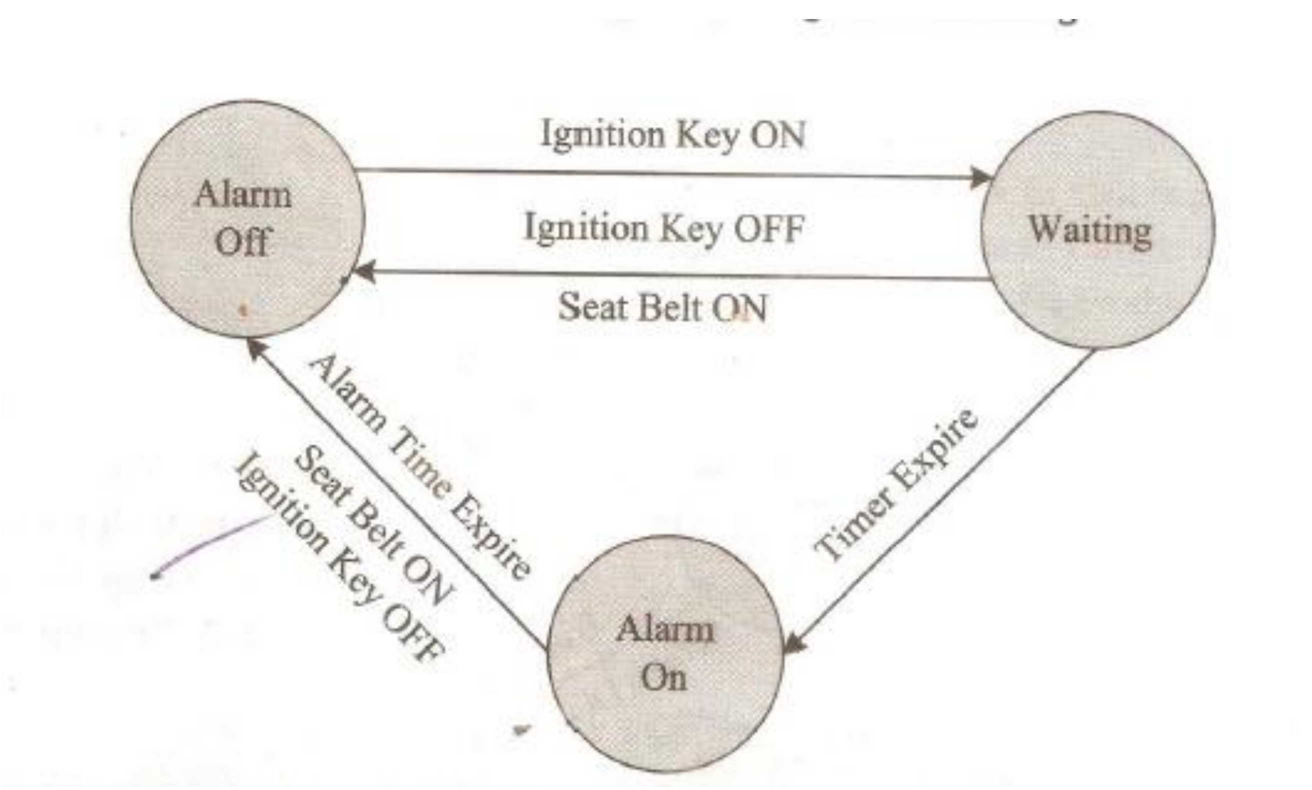
normal flow chart based design.



State Machine Model

- The state machine model is used for modelling reactive or event-driven embedded systems whose processing behaviour are dependent on state transitions.
- It describes the system behaviour with ‘States’, ‘Events’, ‘Actions’ and ‘Transitions’.
- *State* is a representation of a current situation
- An *event* is an input to the *state*
- The *event* acts as stimuli for state transition
- *Transition* is the movement from one state to another
- *Action* is an activity to be performed by the state machine.

FSM model for seat belt warning system



Real Time Operating System (RTOS)

Features of a Real Time operating system

- Clock and Timer support
  - Clock and timer services with adequate resolution are the most important issues in a RTOS
- Real Time priority levels
  - RTOS must support static priority levels.
- Fast task preemption
  - Time duration for which a higher priority task waits before it is allowed to execute is quantitatively expressed as *task preemption time*
- Predictable and fast interrupt latency
  - Interrupt latency is the occurrence of the interrupt and running of the corresponding subroutine
  - Interrupt latency must be less than a few micro seconds

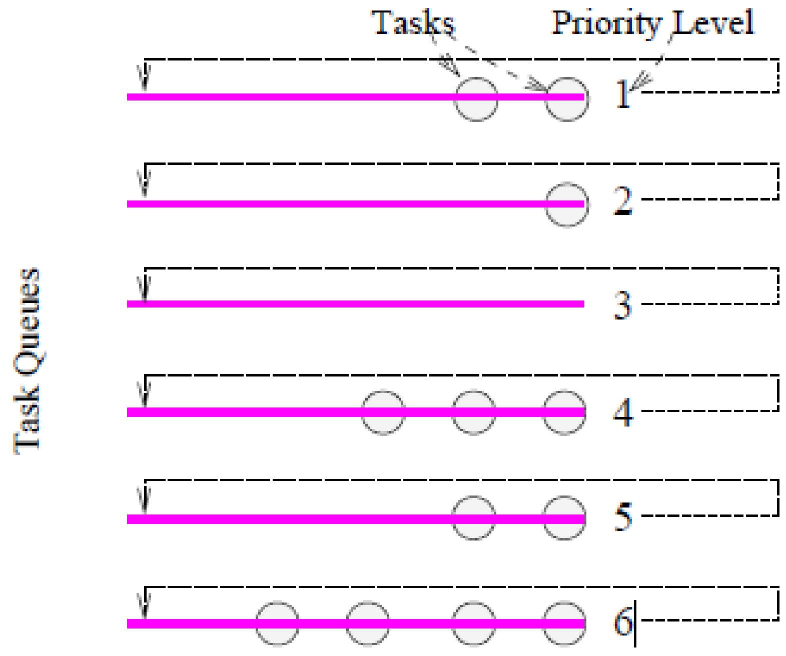
Unix

- UNIX was originally developed for mainframe computers.
- However, Unix and its variants have now permeated to desktop and even handheld computers.

Dynamic priority levels

- The scheduler arranges tasks in multilevel queues.
- At every preemption point, the scheduler scans the multilevel queue from the top(highest priority) and selects the tasks at the head of the first non empty queue.
- Each task is allowed to run for a fixed time

(Unix normally uses 1 second time slice)



Non preemptive kernel

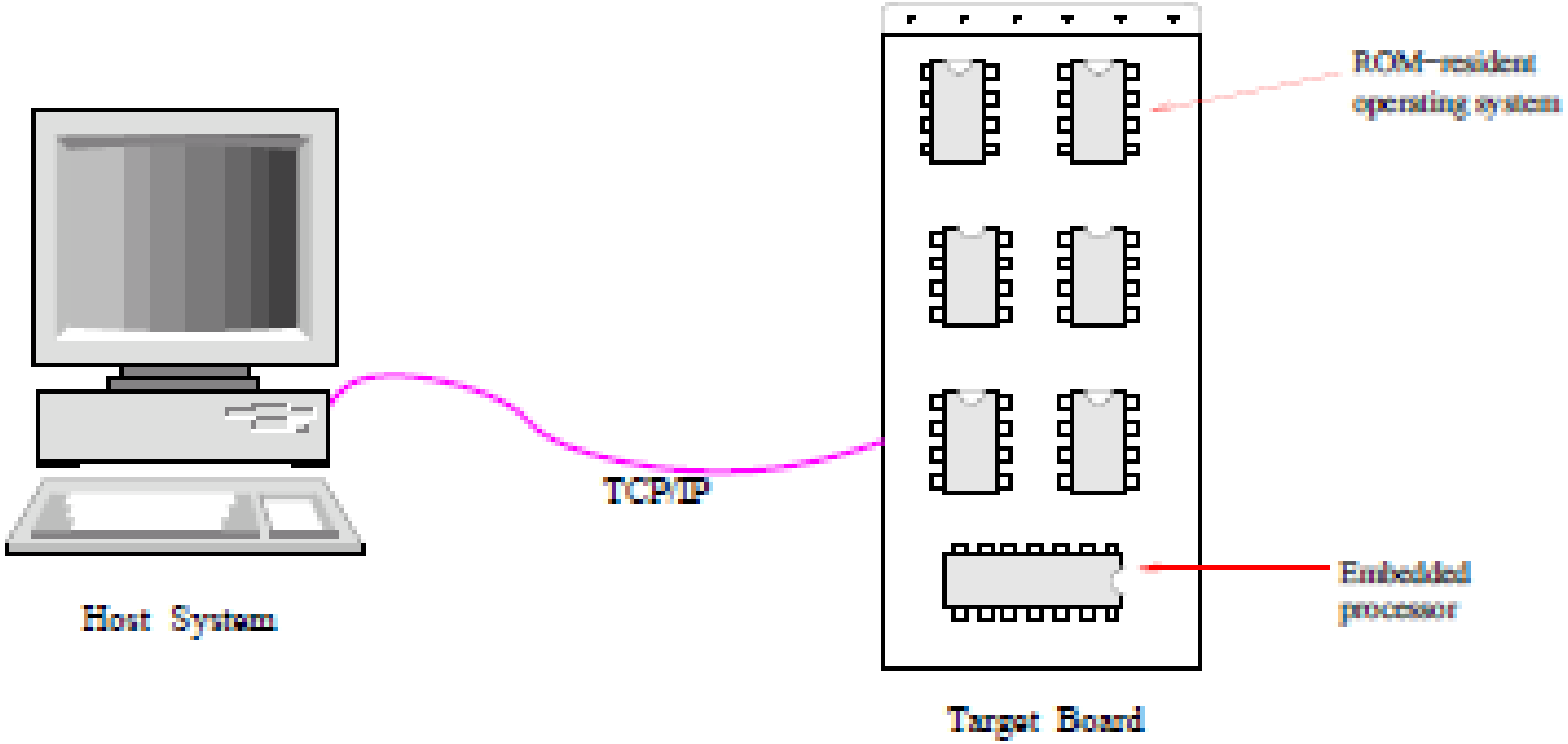
- In Unix, a process running in kernel mode cannot be pre-empted by other processes.

Unix based Real Time Operating System

- Different approaches undertaken to make Unix suitable for real time applications
  - Extensions to the traditional unix kernel
    - Additional capabilities such as real time timer support, a real time task scheduler were implemented.

- Host target approach
  - Real time application development is done on a host machine.
  - The host system is a Unix or Windows based system supporting the program development environment, including compilers, editors, library, cross-compilers, debuggers etc.
  - The host is connected to the target using a serial port or TCP/IP (Transmission control protocol/internet protocol) connection.
  - The real time program is developed on the host.

- It is then cross-compiled to generate code for the target processor.
- Subsequently, the executable module is downloaded to the target board.
- Tasks are executed on the target board and the execution is controlled at the host side using a cross-debugger.
- Once the program works successfully, it is fused on a ROM or flash memory and becomes ready to be deployed in applications.
- Example PSOS (portable software on silicon)


ROM−resident
operating system
TCP/IP
Embedded
processor
Host System
Target Board

Preemption point approach

- To improve the performance of non-preemptive kernels preemption points are introduced in system routines.
- At preemption points, the kernel can safely be preempted to make way for any waiting higher priority real time tasks to run without corrupting any kernel data structures.
- This approach is suitable for use in many categories of hard real time applications.
- Involves only minor changes to be made in the kernel code.
- Example: Windows CE

Self Host systems

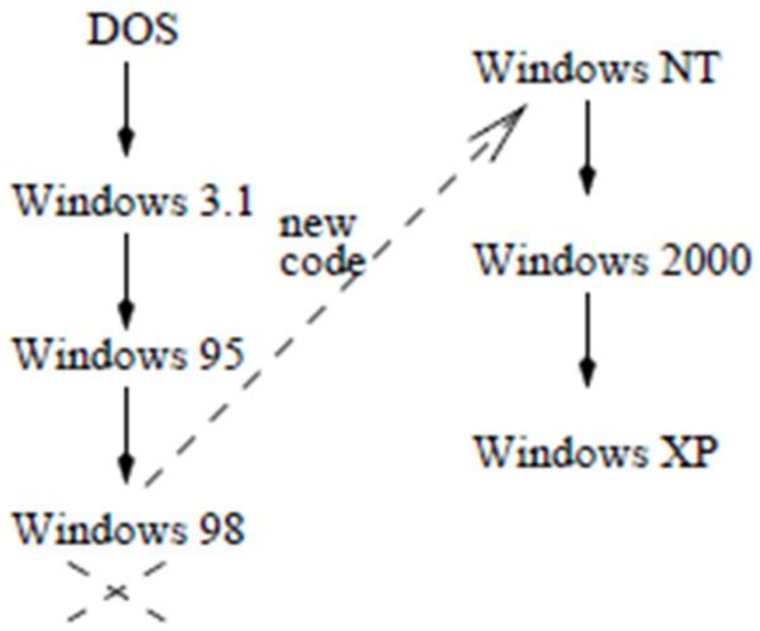
- A real-time application is developed on the same system on which the real-time application would finally run.
- The operating system modules that are not essential during task execution are excluded during deployment.
- The real time application is developed on the full fledged operating system.
- Once the application runs satisfactorily on the host, it is fused on a ROM or flash memory on the target board along with a possibly stripped down version of the operating system.

- Most of the self-host operating systems are based on micro-kernel architecture.
- The add-on modules can be easily excluded, whenever these are not required.

- In a micro-kernel architecture, only the core functionalities such as interrupt handling and process management are implemented as kernel routines.
- All other functionalities such as memory management, file management, device management etc are implemented as add on modules which operate in user mode.

Windows as a Real Time Operating System

- Microsoft developed DOS (Disk operating system) in the early Eighties.
- DOS was a very simple operating system that was single tasking
- DOS evolved to the Windows series operating systems in the late Eighties (graphical front end)
- The Windows code was completely rewritten in 1998 to develop the Windows NT system. (much more stable than the earlier DOS based systems)
- Computers based on Windows NT extensively used in homes, offices and industrial establishments.
- Used in prototype application development


DOS
Windows 3.1
Windows 95
Windows 98
new
code
Windows NT
Windows 2000
Windows XP

Important features of windows NT

- Windows NT support 32 priority levels.
- Each process belongs to one of the priority classes: idle, normal, high, real time.
- By default, the priority class at which a user task runs is normal
- Both normal and high priority classes are variable type (priorities of tasks in this class are recomputed periodically by the OS).
- Processes such as screen saver use priority class idle.

POSIX

- POSIX stands for Portable Operating System Interface.
- POSIX started as an open software initiative, but now has almost become standard for operating system.
- Open system advocates standard interfaces for similar products; so that users can easily integrate their application with the products supplied by any vendor.

- The most important goals of open systems are: interoperability and portability
- Interoperability means systems from multiple vendors can exchange information among each other.
- A system is portable if it can be moved from one environment to another without modifications.

- Open software:
  - An open system is a vendor neutral environment, which allows users to intermix hardware, software and networking solutions from different vendors.
  - Reduces cost of development and time to market a product.
  - Helps increase the availability of add-on software packages.
  - Enhances ease of programming
  - Facilitates easy integration of separately developed modules

- Open software standard can be classified into
- *Open source*
  - Provides portability at the source code level. Example ANSI, POSIX
  - To run an application on a new platform would require only compilation and linking.
- *Open object*
  - Provides portability of unlinked object modules across different platforms
  - To run an application on a new platform, relinking of the object modules would be required.
- *Open binary*
  - Provides complete software portability across hardware platforms based on a common binary language structure.
  - Can be portable at the executable code level.

History of POSIX

- Unix was originally developed by AT&T Bell Labs in the early seventies.
- UCB (University of California at Berkeley) was one of the earliest recipients of Unix source code.
- AT&T later developed Unix and came up with Unix V.
- UCB came up with its own version of Unix and named it as BSD (Berkeley Software Distribution)
- Many vendors implemented and extended Unix services in different ways: IBM with its AIX, HP with its HP-UX, Sun with its Solaris, Digital with its Ultrix, SCO with SCO-Unix

- The important parts of POSIX and the aspects they deal with
- POSIX.1 : system interfaces and system call parameters
- POSIX.2 : shells and utilities
- POSIX.3 : test methods for verifying conformance to POSIX
- POSIX.4 : real-time extensions

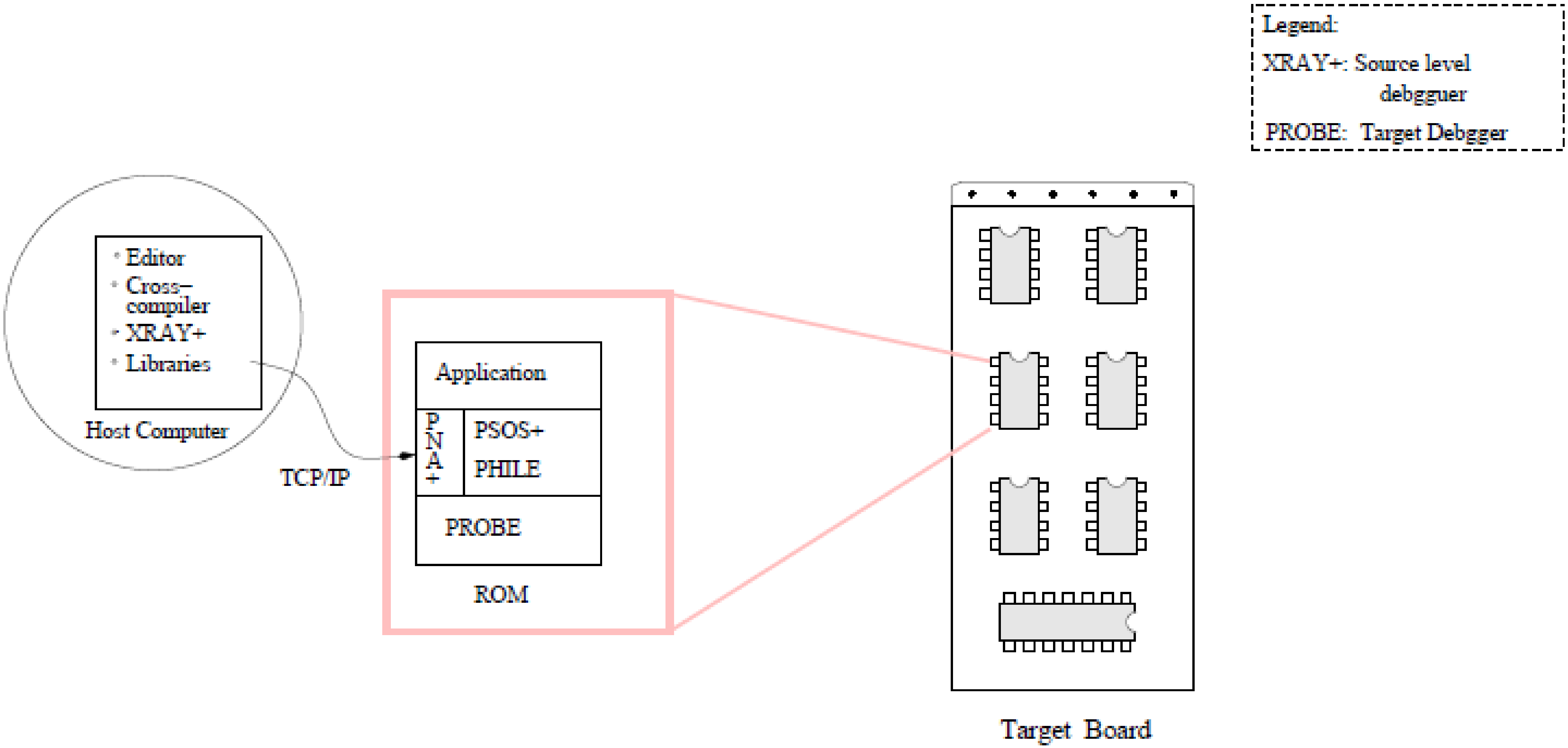
Real time POSIX standard

- Main requirements of POSIX-RT are:
- Execution scheduling
  - Must provide support for real-time (static) priorities
- Performance requirements on system calls
  - Worst case execution times are specified
- Priority levels
  - The number of priority levels supported should be at least 32
- Timers
  - Periodic and one shot timers should be supported
- Real Time files
  - It can preallocate storage for files and should be able to store file blocks on the disk

- Memory locking
  - mlockall() to lock all pages of a process
  - mlock() to lock a range of pages
  - mlockpage() to lock only the current page
  - The unlock services are munlockall(), munlock() and munlockpage()
- Multithreading support
  - POSIX-RT mandates threading support by an operating system

PSOS

- PSOS is available from Wind River Systems, a large player in the RTOS arena
- Used in several commercial embedded products
- An example application of PSOS is in the base stations of cell phone systems


Legend:
XRAY+: Source level debgguer
PROBE: Target Debgger
Editor
Cross–compiler
XRAY+
Libraries
Host Computer
TCP/IP
Application
PNA+
PSOS+
PHILE
PROBE
ROM
Target Board

- The host computer is typically a desktop.
- The target board contains the embedded processor, ROM, RAM etc.
- The host computer runs the editor, cross compiler, source-level debugger and library routines.
- PSOS+ and other optional modules such as PNA+, PHILE and PROBE are installed on a ROM on the target board.
- PNA+ is the network manager and it provides efficient downloading and debugging communication between the target and the host.
- PROBE+ is the target debugger and XRAY+ is the source-level debugger.

- Important features
    - PSOS supports 32 priority levels which can be assigned to tasks.
    - In the minimal configuration, the footprint of the target operating system is only 12KBytes.
    - For sharing critical resources among real-time tasks, it supports priority inheritance and priority ceiling protocols.
    - Supports segmented memory management as it is intended to be used in small and moderate sized embedded applications.

VRTX

- VRTX is a POSIX-RT compliant operating system from Mentor Graphics
- VRTX has been certified by the US FAA (Federal Aviation Agency) for use in mission and life critical applications such as avionics.
- Available in 2 multitasking kernels: VRTXsa and VRTXmc
- *VRTXsa*
  - Used for large and medium sized applications
  - Supports Virtual memory
  - Has a POSIX compliant library and supports priority inheritance.

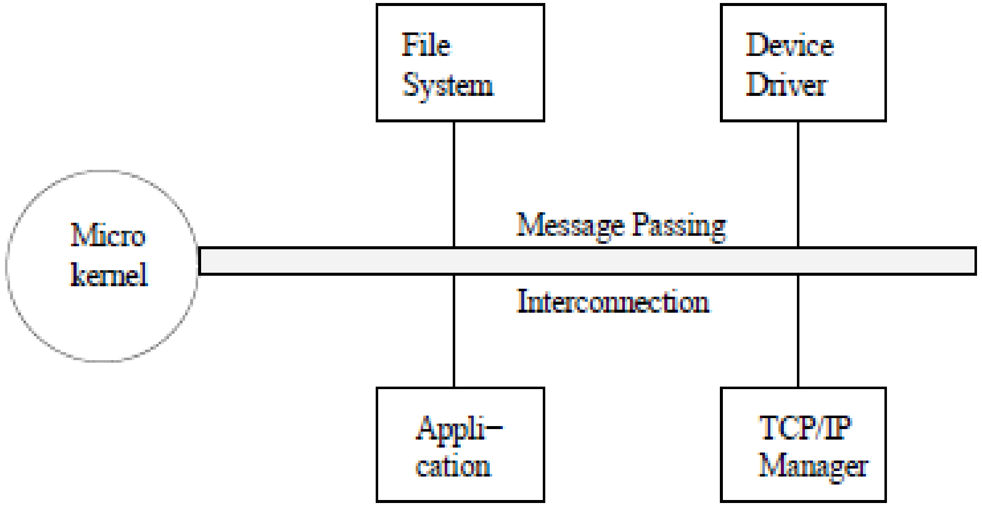
- *VRTXmc*
  - Optimized for power consumption and ROM and RAM sizes
  - Doesnot support virtual memory
  - Kernel typically requires only 4 to 8KBytes of ROM and 1KBytes of RAM
  - Targeted for use in embedded applications such as computer based toys, cell phones etc

VxWorks

- VxWorks is a product from Wind River Systems.
- Host can be either a Windows or a Unix machine.
- VxWorks comes with an integrated development environment (IDE) called Tornado which contains *VxSim* and *Windview*
- *VxSim* simulates a VxWorks target for use as a prototyping and testing environment in the absence of the actual target board.
- *WindView* provides debugging tools for the simulator environment.
- VxWorks was deployed in the Mars Pathfinder which was sent to Mars in 1997

QNX

- QNX is a product from QNX software system Ltd.
- Intended for use in mission critical applications in the areas such as medical instrumentation, Internet routers, telemetric devices, process control applications and air traffic control systems.
- Can be configured to a very small size

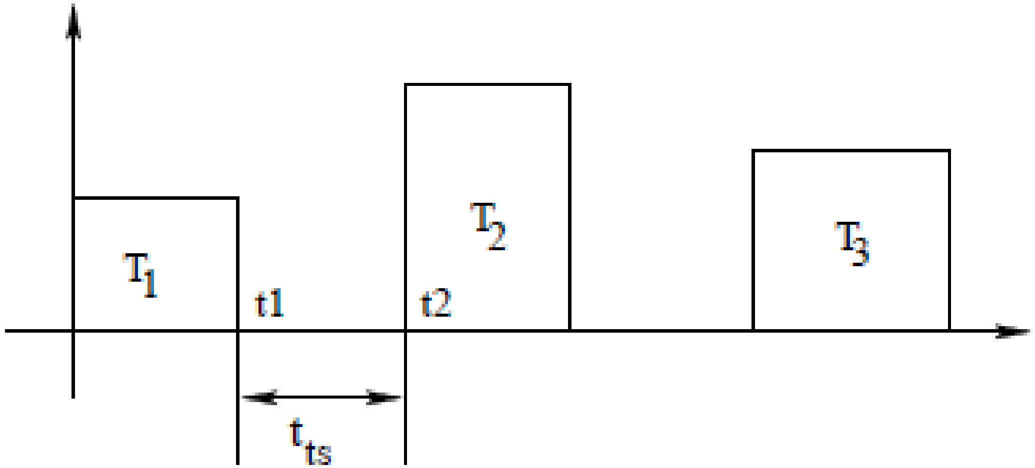


µC/OS-II

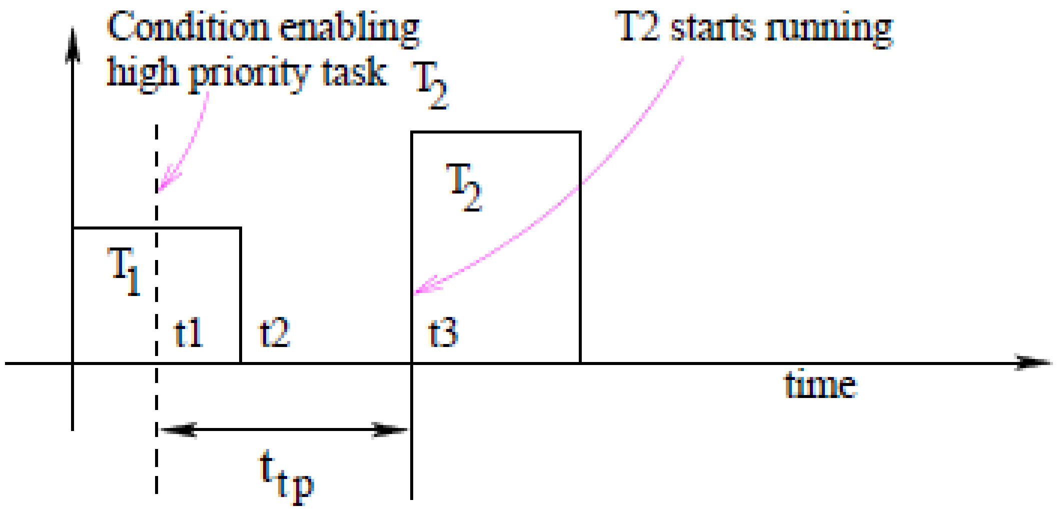
- µC/OS-II is available from Micrium Corporation.
- Written in ANSI X and contains small portion of assembly code.
- Important Features are
  - µC/OS-II was designed to let the programmers have the option of using just a few of the offered services or select the entire range of services
  - µC/OS-II has a fully preemptive kernel.
  - µC/OS-II allows upto 64 tasks to be created.
  - µC/OS-II uses a partitioned memory management scheme.
  - µC/OS-II has been certified by Federal Aviation Administration (FAA) for use in commercial aircrafts.

Benchmarking Real Time systems

- Rhealstone Metric
- Task Switching Time (t_{ts})
- The time it takes for one context switch among equal priority tasks.



- Task Preemption Time (t_{tp})
- Time it takes to start execution of a higher priority task (compared to the currently running task), after the condition enabling the task occurs.



- Interrupt Latency Time (t_{il})
- Time it takes to start the execution of the required ISR after an interrupt occurs.