REAL TIME OPERATING SYSTEM PROGRAMMING-I: μC/OS-II and VxWorks

Lesson-1: RTOSes

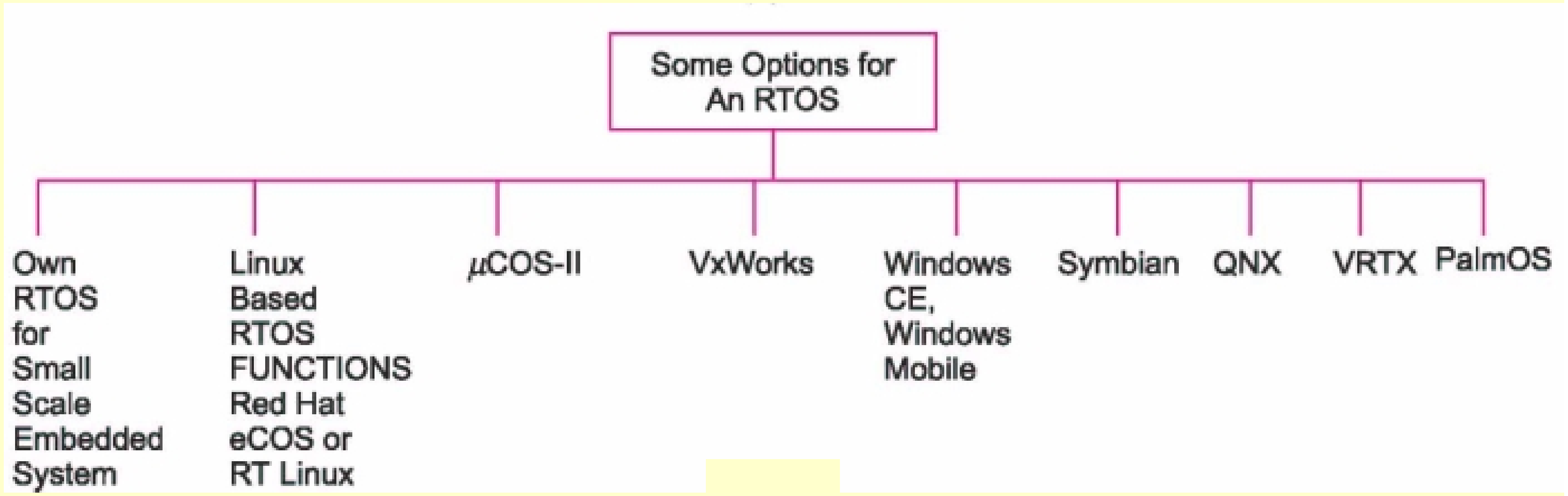
1. Kernel of an RTOS

Kernel of an RTOS

- Used for real-time programming features to meet hard and soft real time constraints,
- Provides for preemption points at kernel, user controlled dynamic priority changes, fixed memory blocks, asynchronous IOs, user processes in kernel space and other functions for a system.

2. Common options for RTOS

Common options available for selecting an RTOS


Some Options for An RTOS
Own RTOS for Small Scale Embedded System
Linux Based RTOS FUNCTIONS Red Hat eCOS or RT Linux
µCOS-II
VxWorks
Windows CE, Windows Mobile
Symbian
QNX
VRTX
PalmOS

Linux 2.6.24

3. Complex multitasking embedded system design requirements

Complex multitasking embedded system design requirements

- Integrated Development Environment,
- Multiple task functions in Embedded C or Embedded C++,
- Real time clock— hardware and software timers,
- Scheduler,
- Device drivers and device manager,

Complex multitasking embedded system design requirements…

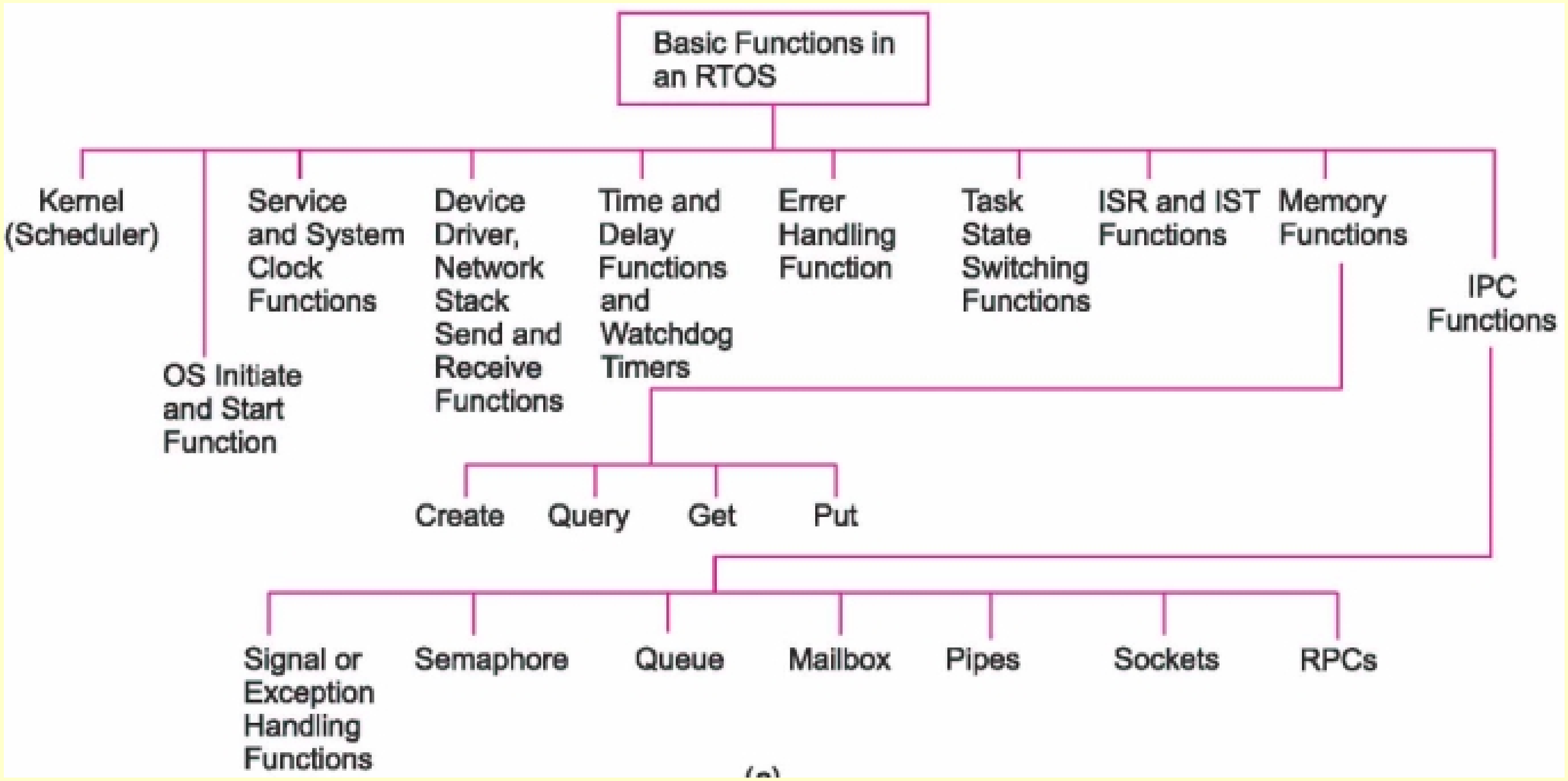
- Functions for inter inter-process communications using the signals, event flag group, semaphore- handling functions, functions for the queues, mailboxes, pipe, and sockets,

Complex multitasking embedded system design requirements…

- Additional functions for example, TCP/IP or USB port, other networking functions,
- Error handling functions and Exception handling functions, and
- Testing and system debugging software for testing RTOS as well as developed embedded application

4. Basic functions expected from kernel of an RTOS

Basic functions expected from kernel of an RTOS



Basic Functions in an RTOS
Kernel (Scheduler)
OS Initiate and Start Function
Service and System Clock Functions
Device Driver, Network Stack Send and Receive Functions
Time and Delay Functions and Watchdog Timers
Errer Handling Function
Task State Switching Functions
ISR and IST Functions
Memory Functions
IPC Functions
Create
Query
Get
Put
Signal or Exception Handling Functions
Semaphore
Queue
Mailbox
Pipes
Sockets
RPCs

5. RTOS features in general

RTOS features in general

- Basic kernel functions and scheduling: Preemptive or Preemptive plus time slicing
- Support to Limited Number of tasks and threads
- Task priorities and Inter Service Threads priorities definitions
- Priority Inheritance feature or option of priority ceiling feature

RTOS features in general…

- Task synchronization and IPC functions
- Support to task and threads running in kernel space
- IDE consisting of editor, platform builder, GUI and graphics software, compiler, debugging and host target support tools

RTOS features in general…

- Device Imaging tool and device drivers
- Clock, time and timer functions,
- Support to POSIX,
- Asynchronous IOs,
- Fixed memory blocks allocation and deallocation system,
- Support to different file systems and flash memory systems

RTOS features in general…

- TCP/IP protocols, network and buses protocols,
- Development environment with Java
- Componentization (reusable modules for different functions), which leads to small footprint (small of size of RTOS codes placed in ROM image)
- Support to number of processor architectures, such as INTEL, ARM, Philips, …

6. Development Approaches

Host and Target Based Development Approach

- . A host machine (Computer) for example, a PC uses a general purpose OS, for example, Windows or Unix for system development. The target connects by a network protocol for example TCP/IP during the development phase. The developed codes and the target RTOS functions first connect a target. The target with downloaded codes finally disconnects and contains a small size footprint of RTOS. For example, the target does not download host machine resident compiler, cross compiler, editor for programs, simulation and debugging programs, and MMU support

Self Host Based Development Approach

- same system with full RTOS is used for development on which the application will be running. This also does not require cross compilation. When application codes are ready, the required RTOS function codes and application codes are downloaded into the ROM of the target board

7. Types of RTOSes

In-House Developed RTOSes

- Codes written for the specific need based, and application or product
- Customizes the in-house design needs.

In-House Developed RTOSes …

- Generally either a small level application developer uses the in-house RTOS or a big research and development company uses the codes built by in-house group of engineers and system integrators

Broad based Commercial RTOSes

- Provides an advantage of availability of off the self thoroughly tested and debugged RTOS functions.
- Provides several development tools.
- Testing and debugging tools

Broad based Commercial RTOSes…

- Support to many processor architectures─ ARM as well as x86, MIPS and SuperH.
- Support to GUIs
- Support to many devices, graphics, network connectivity protocols and file systems

Broad based Commercial RTOSes …

- Support to device software optimization (DSO)
- Provides error and exceptional handling functions can be ported directly as these are already well tested by thousands of users

Broad based Commercial RTOSes …

- Simplifies the coding process greatly for a developer
- Helps in building a product fast
- Aids in building robust and bug-free software by thorough testing and simulation before locating the codes into the hardware

Broad based Commercial RTOSes …

- Saves large amount of RTOS, tools and in-house documentation development time.
- Saving of time results in little time to market an innovative and new product.
- Saves the maintenance costs.
- Saves the costs of keeping in-hose engineers.

General Purposes OSes with RTOS

- Embedded Linux or Windows XP is general purpose OS. They are not componentized. Footprint (the code that goes as ROM image) is not reducible.
- The tasks are not assignable priorities.
- However, they offer powerful GUIs, rich multimedia interfaces and have low cost.

General Purposes OSes with RTOS …

- The general purpose OSes can be used in combination with the RTOS functions.
- For example, Linux 2.6.24 and RTLinux sre real time kernels over the Linux kernel.
- Other example is ‘Windows XP Embedded’ for x86 architecture

Special Focus RTOSes

- Used with specific processors like ARM or 8051 or DSP,
- OSEK for automotives or
- Symbian OS for Mobile phones

Summary

We learnt

- RTOS used for real-time programming features to meet hard and soft real time constraints,
- Provides for preemption points at kernel, user controlled dynamic priority changes, fixed memory blocks, asynchronous IOs, user processes in kernel space and other functions for a system

We learnt

- RTOSes basic functions are OS initiate and start, scheduling, error handling, system clock and service, time delay and task, memory and IPC management functions.

We learnt

- Host-target and self hos development approaches
- In-house, broad-based, general purpose with RTOS functions and special focus RTOSes

End of Lesson 1 of Chapter 9