

Real-Time Systems
Prof. Dr. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Module No. # 01

Lecture No. # 28

Open Source and Commercial RTOS (Contd.).

Good morning. So, let us get started with what we were doing last time. So, far we have seen the basic concepts in real-time systems and scheduling requirements of real-time operating systems, and then, we saw few open source real-time operating systems. So, today, we will continue with that discussion and we will look at few more important open source and commercial real-time operating systems. So, let us proceed with that.

(Refer Slide Time: 00:44)

VRTX (Mentor Graphics)

- Originally 4KB in size:
 - Did not support many required functionalities.
 - Could not be used as a full RTOS.
- Now comes in 2 versions:
 - **VRTX SA and VRTX MC**

03/08/10

4

Open Source and Commercial RTOS

(cont...)

(Lecture 28)

Dr. RAJIB MALL
Professor
Department Of Computer Science & Engineering
IIT Kharagpur.

03/08/10 1

(Refer Slide Time: 01:10)

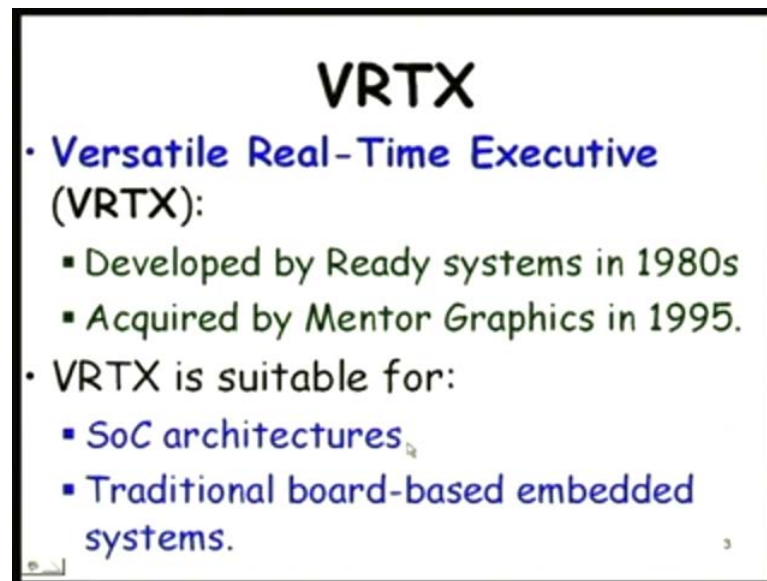
Review

- In the last class, we discussed a few free real-time operating systems:
 - RTAI
 - RTLinux
 - eCOS
 - microCOS

2

So, last class, we discussed few free real-time operating systems; we basically looked at the RTAI developed in a university situation in Italy, the RT Linux, e COS and the micro COS. Today, we will look at few more important ones; **those the** ones that we look today is used widely in many applications.

(Refer Slide Time:01:25)



VRTX

- **Versatile Real-Time Executive (VRTX):**
 - Developed by Ready systems in 1980s
 - Acquired by Mentor Graphics in 1995.
- VRTX is suitable for:
 - SoC architectures
 - Traditional board-based embedded systems.

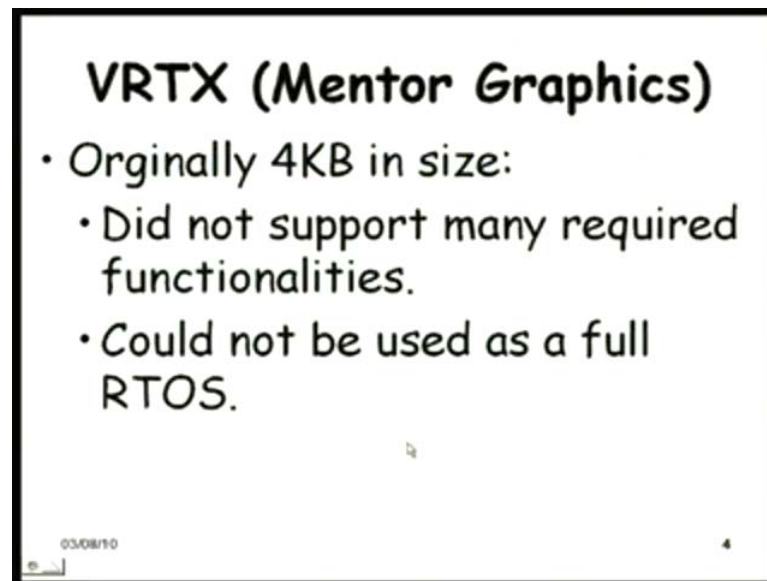
3

We first look at the VRTX, VRTX stands for versatile real-time executive; it is developed nearly 30 years back by ready systems. There are basically two or three persons in a small company, hunter and Ready Company 1980's, one of the very few real-time operating systems which could be used in embedded applications 1980's and this was acquired by the mentor graphics 1995.

As I was saying that, it is suitable for embedded applications, it is widely used for system on chip architectures. See, many of these embedded applications are actually system and chip. So, a single chip takes care of the entire system basically to reduce power consumption and to reduce cost and so on, just a single chip takes care of the entire systems requirements. And VRTX is widely used on the SoC architectures and also the traditional board- based embedded systems, which we had discussed some extent earlier, it is used in both of these very widely.

The original VRTX developed in 1980's had **very few features** very little features and after it was acquired by mentor graphics, features have improved and now it could be used with many sophisticated applications.

(Refer Slide Time: 03:09)



Originally only 4 kilobyte in size, very very rudimentary operating system and did not support many of the required functionalities and it could not be used as a full real-time operating systems; many aspects, the programmer have to write himself or herself. But now as I was saying that, it has become more sophisticated; it can be used as a full real-time operating system now; we will see that **posix r t** compliant and comes in two versions one is the VRTX SA and the other is the VRTX MC.

The VRTX MC is used for very small applications, MC stands for microcontroller; very small applications, where power cost, etcetera are prime importance, but for medium and large applications, VRTX SA is used.

(Refer Slide Time: 04:13)

VRTX (Mentor Graphics)

- Originally 4KB in size:
 - Did not support many required functionalities.
 - Could not be used as a full RTOS.
- Now comes in 2 versions:
 - **VRTX SA and VRTX MC**

03/08/10 4

The VRTX MC are the microcontroller version; it is optimized for low power consumption ROM and RAM sizes. The kernel again is very small only a few kilobytes intended for very simple mobile phones. Now a days of course, the cell phones are becoming more and more sophisticated requiring full fledged operating system, where you can do web browsing and so on email, etcetera, it is not like that, it is very simple operating system on very simple cell phones, VRTX has been used.

(Refer Slide Time: 04:52)

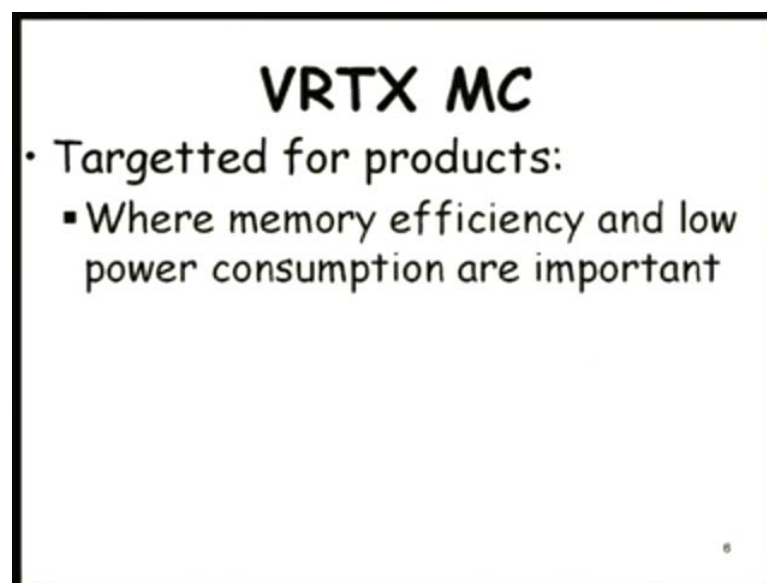
VRTX MC (Micro Controller)

- Optimized for low power consumption and ROM and RAM sizes:
 - Kernel occupies only a few KBs
 - Intended for embedded applications such as a cell phone.
- **First commercial RTOS certified by Federal Aviation Authority (FAA)**
 - For safety-critical applications such as avionics.

03/08/10 5

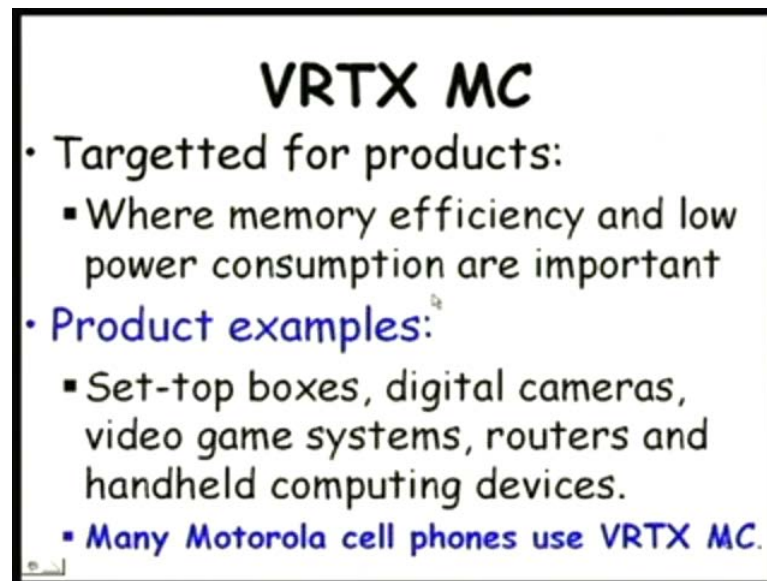
Another very important aspect of the VRTX is that, it was the first commercial real-time operating system to be certified by the Federal Aviation Authority for safety-critical applications like avionics. To get the federal aviation authorities approval is not easy, you have to show that, **is** not only it has been thoroughly tested, all paths covered, etcetera, but also you have to give a guarantee or you have to demonstrate how many failures can occur over millions of hour of operation. Typically five or six failures can be tolerated in a million hour of operation where extremely safety critical.

(Refer Slide Time: 05:40)



As we are saying that it is targeted for products, where memory and low power consumption, cost, etcetera **low** are very important.

(Refer Slide Time: 05:51)

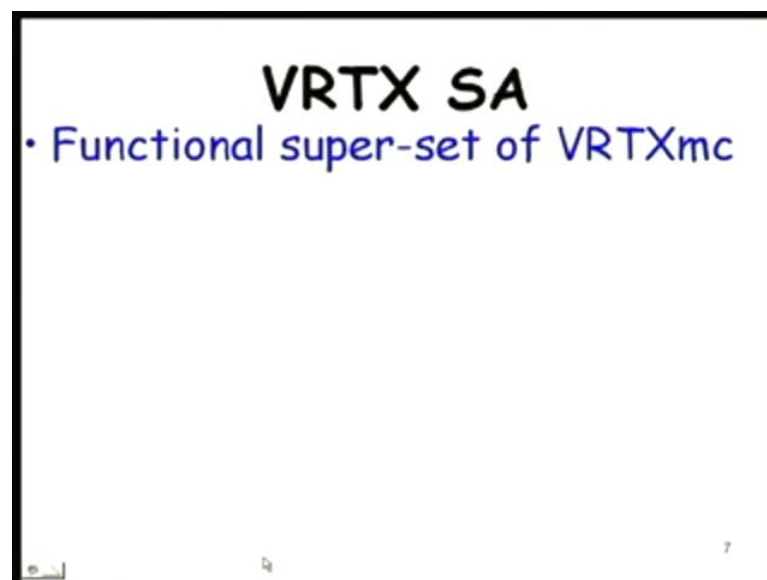


VRTX MC

- Targetted for products:
 - Where memory efficiency and low power consumption are important
- Product examples:
 - Set-top boxes, digital cameras, video game systems, routers and handheld computing devices.
 - Many Motorola cell phones use VRTX MC.

Examples of the products are set-top boxes, simple digital cameras, simple video games, routers mobile phones, handheld computing devices, and etcetera. Many Motorola cell phone versions use VRTX MC.

(Refer Slide Time: 06:12)

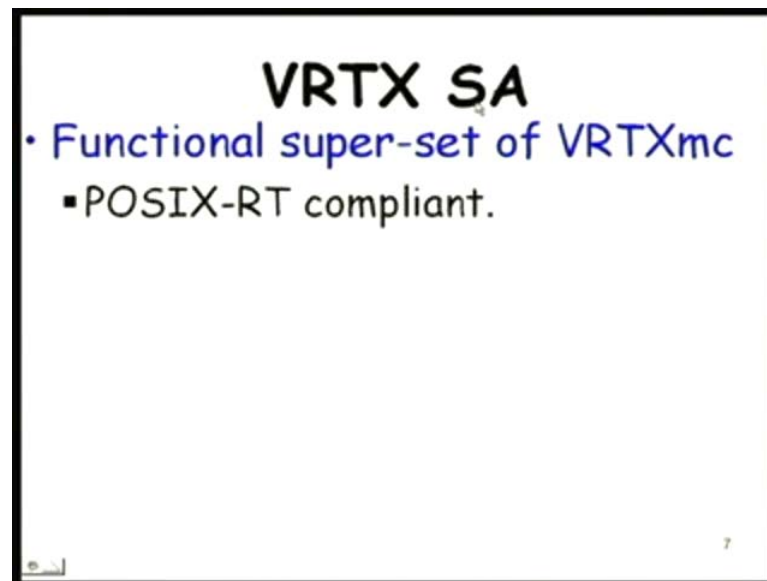


VRTX SA

- Functional super-set of VRTXmc

There is also another flavor of VRTX, the SA version.

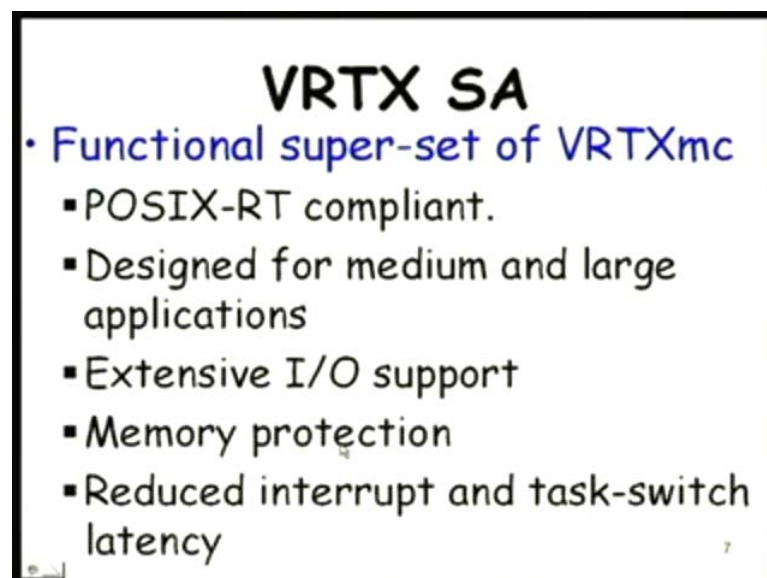
(Refer Slide Time: 06:19)



SA stands for scalable architecture, it is a POSIX compliant. So, all the features are there in this, we should infer just by looking at the POSIX- RT complaint means all the required features for a real-time operating systems are there.

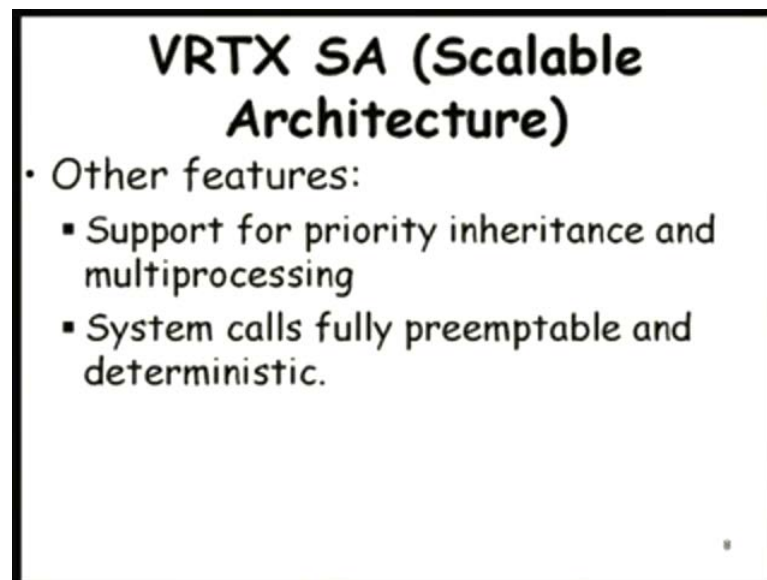
But VRTX MC does not support all the features; designed for medium and large applications, good I/O support, and memory protection.

(Refer Slide Time: 06:49)



So, the user programs or the application programs and the kernel operate in different memory space, it very important for large product development. On the other hand, VRTX MC, it works only for small applications, it would be difficult to develop for larger applications, reduced interrupt and task switching latency.

(Refer Slide Time: 07:15)



The other important features: support for priority inheritance and multiprocessing. We had discussed resource sharing in a non trivial application, tasks need to share resources and therefore, priority inheritance **become** becomes important for resource sharing among real-time tasks. So, I hope all these terms are making sense and system calls are fully preemptable and deterministic.

We also know the implications of a non-preemptable system call and the jitter that can introduce due to a non-preemptable system call, here system calls are preemptable and deterministic.

(Refer Slide Time: 08:07)

VRTX SA (Scalable Architecture)


- Other features:
 - Support for priority inheritance and multiprocessing
 - System calls fully preemptable and deterministic.
- Example use:
 - Hubble space telescope

Many systems have been developed on the VRTX SA; one prominent example is the Hubble space telescope.

(Refer Slide Time: 08:17)

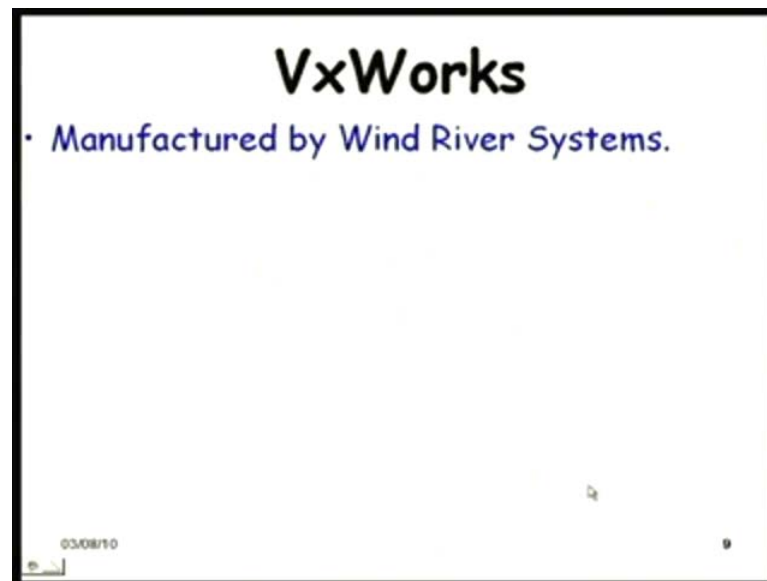
VRTX SA (Scalable Architecture)

- Other features:
 - Support for priority inheritance and multiprocessing
 - System calls fully preemptable and deterministic.
- Example use:
 - Hubble space telescope



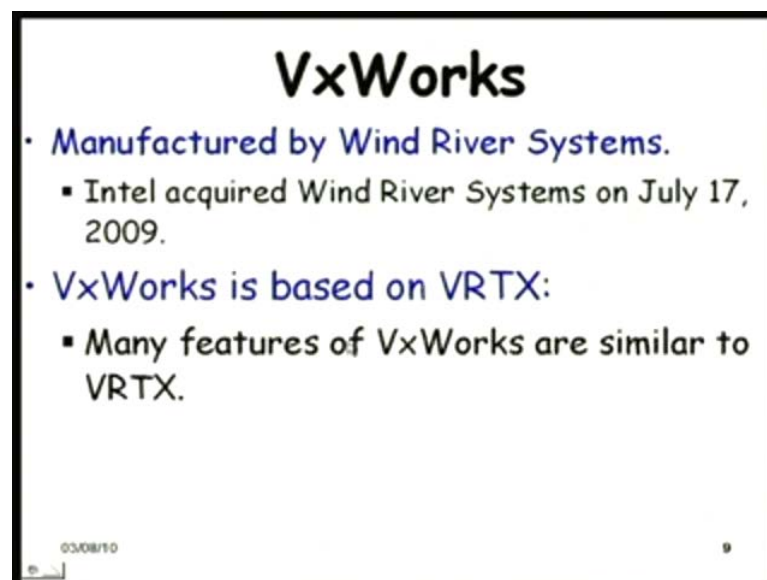
So, here the telescope is sent to the space and it sends images from there.

(Refer Slide Time: 08:28)



Look at another very prominent real-time operating system called as VxWorks manufactured by Wind River systems. Actually now the Wind River systems, which was a large player in this real-time operating system area has been acquired by Intel in 2009.

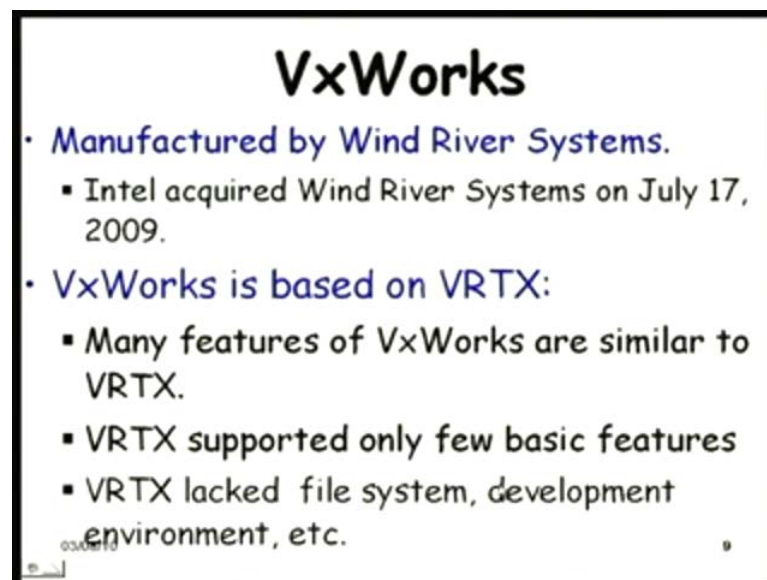
(Refer Slide Time: 09:00)



And the VxWorks, the development history is that, the VRTX operating system was licensed by Wind River systems. So, they said, we will develop an operating system based on VRTX, so they could get the source code and so on. And naturally, many features of VxWorks are similar to VRTX; we had just seen the VRTX.

And the Wind River systems, they had noticed that VRTX is extremely small operating system, lagged most of the facilities even to be called as operating system. So, many said that, VRTX does not work as an operating system and this is supposed to be what you can say joke or something. So, this is a VRTX operating system which works or VxWorks. So, that is how they have named it VxWorks just to and later of course, the VRTX license expired, it was for a certain period and therefore, the source code etcetera is entirely different, but features are similar. So, deliberately they had developed features based on different source codes so that there would not be any licensing problem later.

(Refer Slide Time: 10:42)



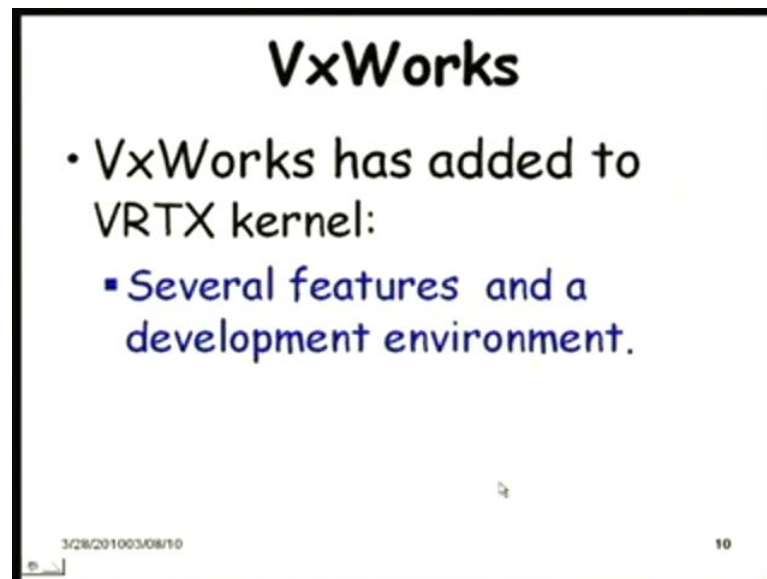
VxWorks

- **Manufactured by Wind River Systems.**
 - Intel acquired Wind River Systems on July 17, 2009.
- **VxWorks is based on VRTX:**
 - Many features of VxWorks are similar to VRTX.
 - VRTX supported only few basic features
 - VRTX lacked file system, development environment, etc.

© 03408710

The VRTX as we are saying that, supported very basic features originally only 4 k b code and lagged for example, file systems, development environment and so on and VxWorks provides all this.

(Refer Slide Time: 10:43)

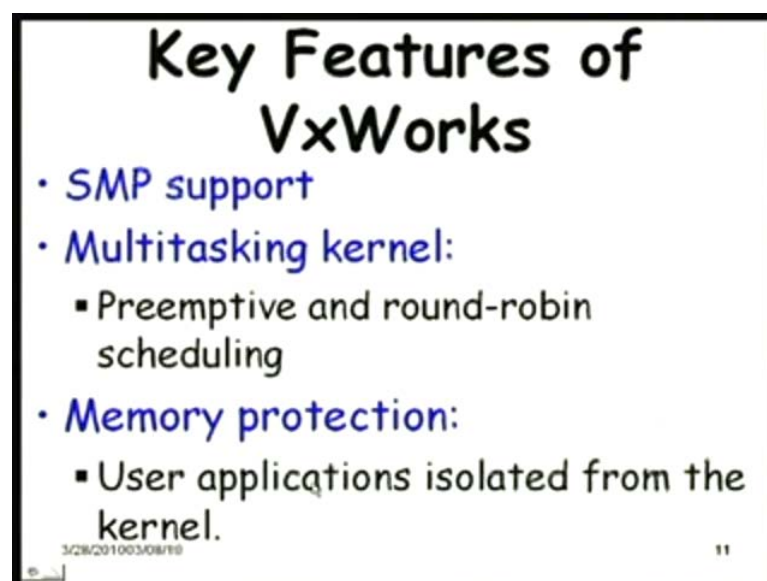


VxWorks

- VxWorks has added to VRTX kernel:
 - Several features and a development environment.

3/28/2010 3:08:10 10

(Refer Slide Time: 11:36)



Key Features of VxWorks

- SMP support
- Multitasking kernel:
 - Preemptive and round-robin scheduling
- Memory protection:
 - User applications isolated from the kernel.

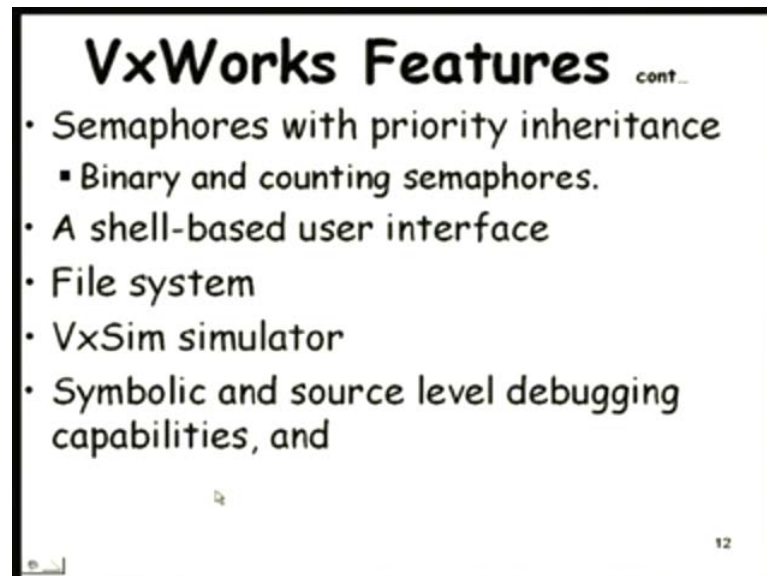
3/28/2010 3:08:10 11

It has added several features and development environment; even it can work on multiprocessors, symmetric multiprocessors, multitasking kernel and preemptive round-robin scheduling. So, you can implement **r m a, e d f** etcetera.

Memory protection between the **application tasks and kernel...** The user applications are isolated from the kernel; so, development of the applications becomes easy. We were discussing on the other day, if protection is not there, each time your application

develops as a bug, it will crash the system and becomes very difficult to debug and correct the bug.

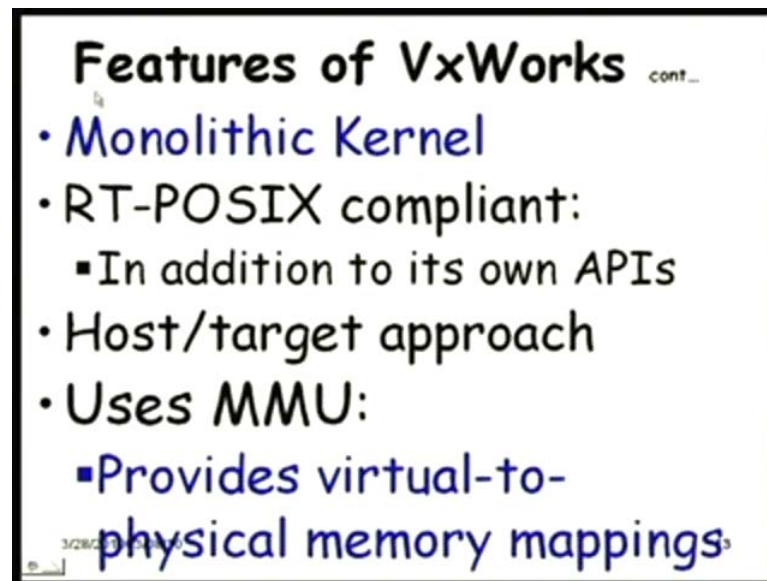
(Refer Slide Time: 11:42)



The features that are expected for a good real-time operating systems POSIX compliant are all available. Semaphores with priority inheritance - both binary and counting semaphores are there; shell based user interface, file system support and also there is a VxSim Simulator. We will just discuss the role of the simulator the VxWorks, I think in the next slide.

Symbolic and source level debugging is supported; and not only that not only the simulator, but also you can monitor the performance of various system calls and application tasks and so on.

(Refer Slide Time: 12:26)



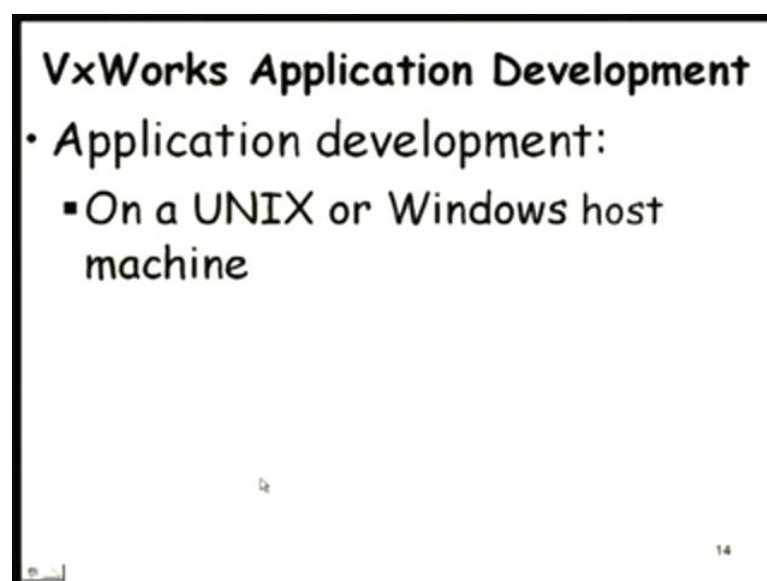
Features of VxWorks cont...

- **Monolithic Kernel**
- RT-POSIX compliant:
 - In addition to its own APIs
- Host/target approach
- Uses MMU:
 - Provides virtual-to-physical memory mappings

3/28/2013 12:26:10 PM

But it was a monolithic kernel operating system; it is real-time POSIX compliant and in addition, it has also several additional APIs and it is a host target approach. We know what the host target approach is. Develop the application on the host and **download the target** download the application on to the target board and provides protection of the application tasks from the kernel through MMU and also the **virtual memory** virtual to physical memory mappings achieved through MMU.

(Refer Slide Time: 13:10)



VxWorks Application Development

- Application development:
 - On a UNIX or Windows host machine

14

The application development can occur either on a UNIX or a windows host. You typically while developing the application use a cross compiler that comes with the operating system; you can run the application on various target architectures and also initially run the application on that host itself through the simulator VxSim.

So, first you simulate the running on the host using VxSim and once it works or you are satisfied, you download it to the target architecture and again run and check for performance and so on and if it works satisfactorily on the target architecture, then you fuse the application on a flash or a RAM.

(Refer Slide Time: 14:24)






And it has also a **development environment** integrated development environment consisting of the operating system application building tools like cross compilers, debuggers. Support for host to target communication would like to download the application to the target and also check the performance, need to debug also in the target. So, that is taken care by the communication. You can run, monitor the run and debug on the target board and of course, the idea also consist of the simulator that helps you run it on the host.

(Refer Slide Time: 15:01)

Products Using VxWorks

- Mars exploration Rovers **Spirit and Opportunity**.
 - Used in several other spacecraft as well, e.g. the **Deep Impact** mission.
- Boeing 787 airliner.
- The **Linksys** wireless router
- Siemens Medical Solutions:
 - Use VxWorks to control real time events of MRI scanners.
- KUKA and ABB industrial robots

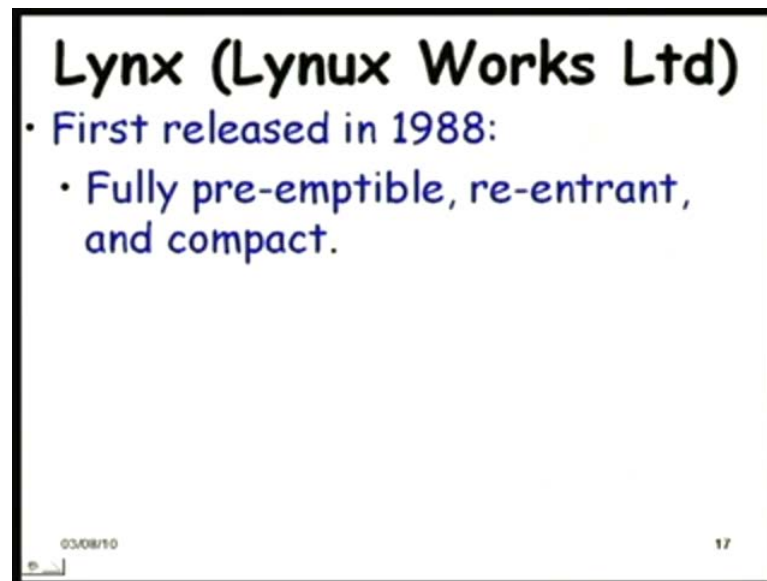


As I was saying that extremely popular operating system, I think we had discussed sometime back that the mars rover Mars exploration Rover, the spirit and opportunity they used the VxWorks operating system; not only that, several other space craft used VxWorks.

For example, the deep impact machine. So, this is the Mars Rover, the Boeing 787 airliner, Linksys wireless router; the Siemens medical solutions also used VxWorks to control real-time events of the MRI scanners and also widely used in industrial robots. So, a robot here is working on some samples here. So, this robo's or many robo's of this type are powered by the VxWorks operating system.

So, right now we are looking only at the features being supported, the kind of application whether it is for a small embedded application or a medium or large application the kind of features, whether POSIX compliant, etcetera. So, depending on the application, appropriate operating system needs to be selected.

(Refer Slide Time: 16:43)



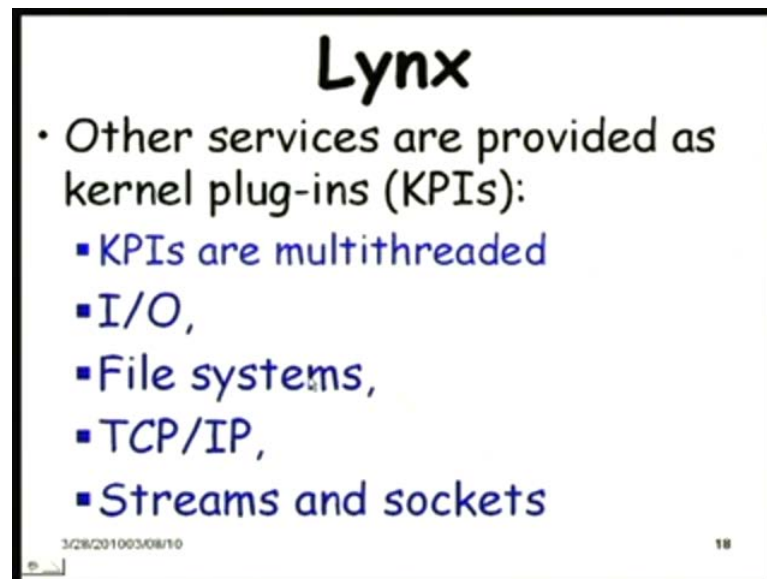
The Lynx also is very popular; first released in 1988, fully preemptable, re-entrant and compact operating system. Initially, the name of the company was Lynx itself and **it is** as the name says, it is a Lynux based operating system. Lynux is open source and definitely they have used the Lynux code and developed based on that and possibly they developed many other software under their company and that is why they have changed it from Lynx to Lynux.

And over the years, they have moved from a monolithic architecture to a microkernel design. Initially, they started as a monolithic architecture, but later due to the advantages of the microkernel design especially in real-time applications, they have moved to the microkernel design.

I hope everybody knows and remember. So, what is the advantage of a microkernel design for a real-time application? Compared to a monolithic kernel and also its disadvantages and also for traditional applications, the advantages and disadvantages become monolithic versus microkernel architecture.

The microkernel is only 28 kilo byte in size; provides the basic services, basic scheduling interrupt dispatch and synchronization and of course, very basic memory management.

(Refer Slide Time: 19:02)

A presentation slide titled "Lynx" in a large, bold, black font. Below the title is a bulleted list. The first bullet point is "Other services are provided as kernel plug-ins (KPIs):". Indented under this are five sub-bullet points: "KPIs are multithreaded", "I/O,", "File systems,", "TCP/IP,", and "Streams and sockets". The text for these sub-bullets is in a blue font. At the bottom left of the slide, there is a small date "3/28/2010 3:08/10" and a small logo. At the bottom right, the number "18" is displayed.

Lynx

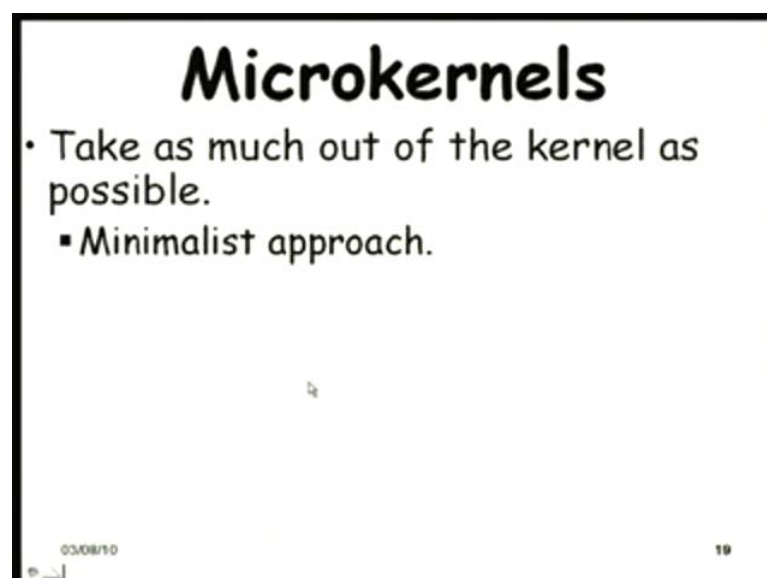
- Other services are provided as kernel plug-ins (KPIs):
 - KPIs are multithreaded
 - I/O,
 - File systems,
 - TCP/IP,
 - Streams and sockets

3/28/2010 3:08/10 18

And all other services are provided as kernel plug-ins and the kernel plug-ins are multithreaded. So, multiple requests, they can handle at a time.

The other kernels, the kernel plug-ins can be the I/O, the file system, the TCP/ IP streams and sockets all work as the kernel plug-ins and this operate in the user space; all these file systems, stream, socket, I/O etcetera at the user space.

(Refer Slide Time: 19:43)

A presentation slide titled "Microkernels" in a large, bold, black font. Below the title is a bulleted list. The first bullet point is "Take as much out of the kernel as possible.". Indented under this is one sub-bullet point: "Minimalist approach.". At the bottom left of the slide, there is a small date "03/08/10" and a small logo. At the bottom right, the number "19" is displayed.

Microkernels

- Take as much out of the kernel as possible.
 - Minimalist approach.

03/08/10 19

The microkernel I think, we had also seen earlier, that the idea is to minimize the kernel size and we had also distinguished between what is a kernel and what is a user space. So, does anybody remember, what is the difference of a kernel with respect to the user space applications with respect? See, there are different aspects; we can compare a kernel with respect to a user space application, but what about the memory management, what do you think?

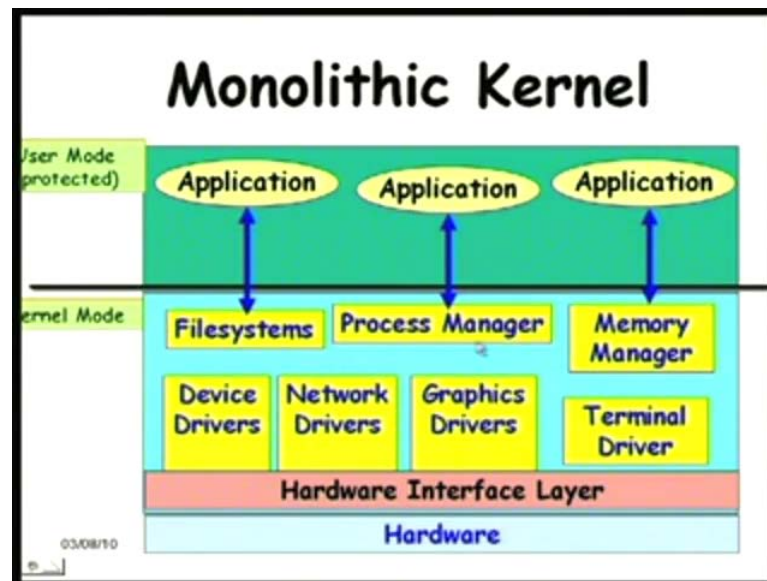
What's the difference between a kernel and the...

(())

Yeah. So, the typically the application the user applications or the even the k p i's, they operate in a virtual memory environment, where they are operate through n m u and then the page swapping. Missiles, etcetera, faults page, faults etcetera occur, but a kernel code is resident.

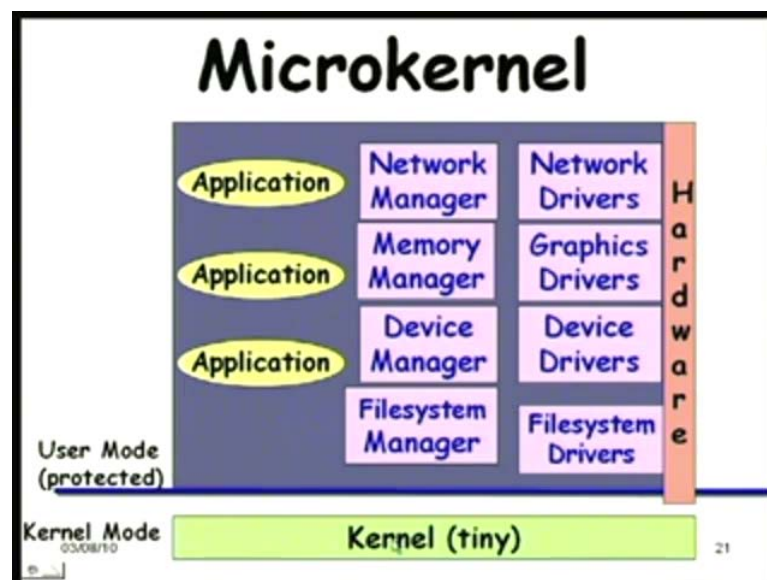
So, let us proceed, say the kernel in a microkernel, the idea is to minimize the kernel or the memory resident part modular and small typically few kilobytes to hundreds of kilobytes and becomes easier to develop a small kernel, easier to port to a different hardware easier to maintain and extend. And we had seen that, different operating system vendors use different facilities for the kernel to support, but typically, they all support some basic process management, basic memory management and inter process communication.

(Refer Slide Time: 21:23)



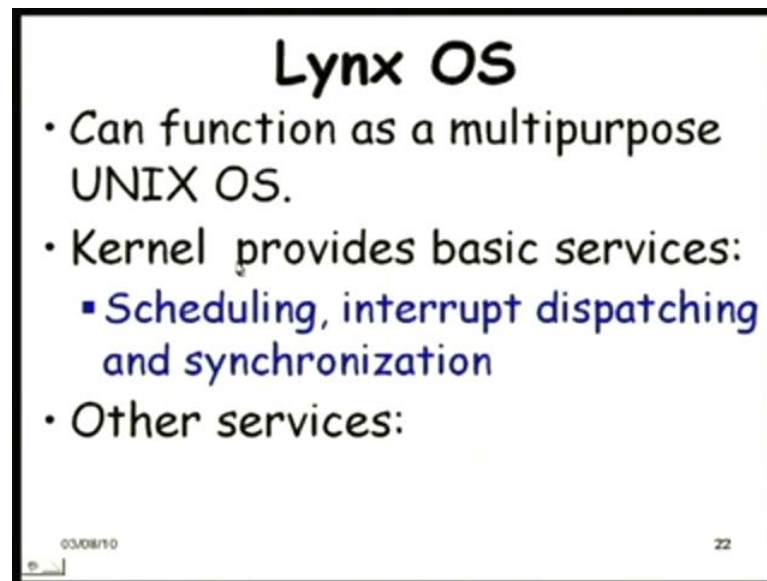
So, this is a monolithic kernel, where all the operating system facilities or the services operate in the kernel mode file system, process manager, memory manager drivers and so on and the applications in the user space.

(Refer Slide Time: 21:45)



On the other hand, in the microkernel approach, the applications and many of the operating system services, they operate in the user space, whereas the microkernel operates in the kernel mode. We will not discuss, please try to recollect the relative advantages of the microkernel and monolithic kernel approach.

(Refer Slide Time: 22:10)



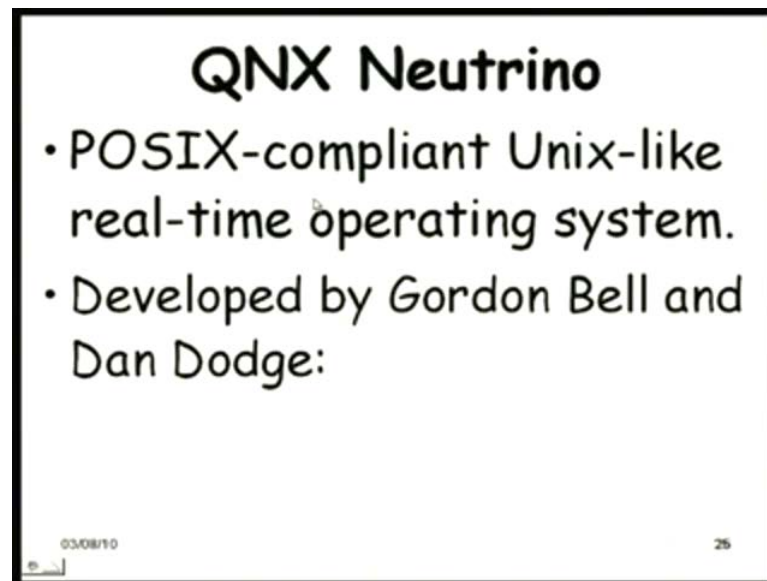
The Lynx can also function as a multipurpose UNIX operating system; you can run soft real-time applications, even non real-time applications, compilers, etcetera on this.

(Refer Slide Time: 22:31)

The kernel provides the basic services and other services are the kernel plug-ins, and unlike the VxWorks and VRTX, Lynx is a self - host system. You know a self-host system is one, where **you are** the real-time application runs on the same host on which you develop, whereas VRTX and VxWorks you develop it on a host, and then, download it to an embedded application. Here the applications, compliers, debuggers, and etcetera, they run through the MMUs providing virtual memory, that is, large memory address space as well as **protection from the kernel, I am sorry** the kernel is protected from the user space.

If you look at more detailed features, it supports a single scheduling policy which is a fixed **a fixed** priority preemptive scheduler with 256 priority levels and the clock frequency is also fixed at 100 hertz, which is timer resolution of hundred milliseconds. So, if your application requires the timer setting to be less than a few milliseconds, obviously, it will have difficulty.

(Refer Slide Time: 24:16)



This is another popular operating system, the QNX Neutrino is the kernel, the QNX Neutrino just read them together; the QNX Neutrino, it is again a UNIX-like operating system. POSIX **rt** compliant, it was actually developed as a student project, two computer science students in the University of Waterloo 1980s.

(Refer Slide Time: 24:20)

Gordon bell and Dan dodge, they developed **a student** in 1980s, and then, both these persons, they setup their own company - the QNX corporation - and they started developing initially of course, it had very **very** basic features as a student project, but they kept on developing this and it became a full-fledged and a good real-time operating system.

(Refer Slide Time: 25:17)



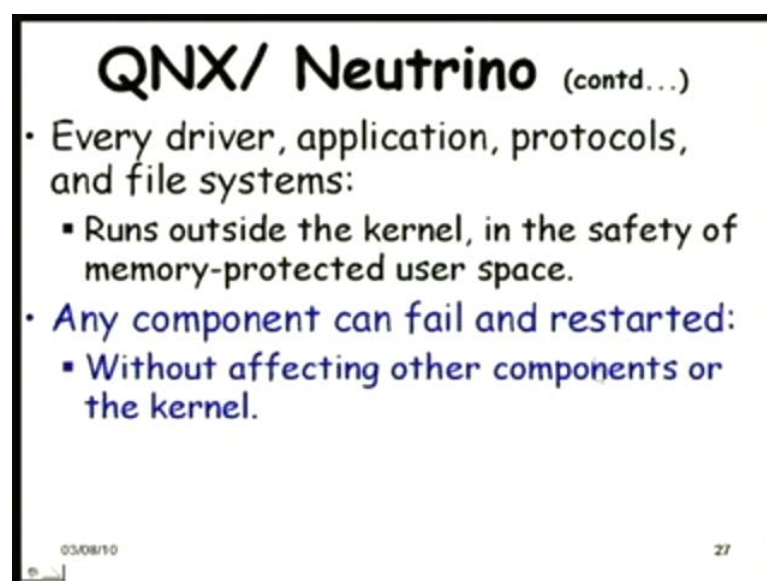
QNX Neutrino

- **Microkernel design:**
 - Kernel provides thread and time services.
 - Allows developers to easily turn off any functionality they do not require.

26

It is a microkernel design, the kernel provides the thread and time services and since it is a microkernel design, you can easily allow or turnoff any functionality that you do not require; you can add new functionalities, remove functionalities not required and so on. And as with any microkernel operating system, here the drivers, applications, protocols, file system, etcetera run outside the kernel and memory protected user space through MNU, making it simple to develop and debug etcetera.

(Refer Slide Time: 26:01)



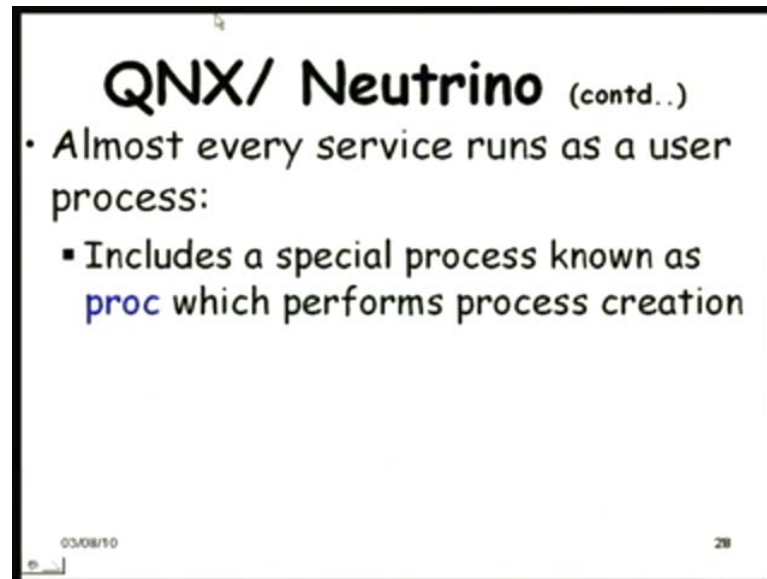
QNX/ Neutrino (contd...)

- Every driver, application, protocols, and file systems:
 - Runs outside the kernel, in the safety of memory-protected user space.
- Any component can fail and restarted:
 - Without affecting other components or the kernel.

03/08/10 27

Not only that, even if a component fails, it can be restarted without affecting the components of the kernel or the system crashing. In a monolithic operating system, if any problem occurs in any of these applications while running, it would result in a system crash; here it does not, you just restart it. And also good application portability feature of any microkernel operating system POSIX-RT complaint.

(Refer Slide Time: 26:37)



QNX/ Neutrino (contd..)

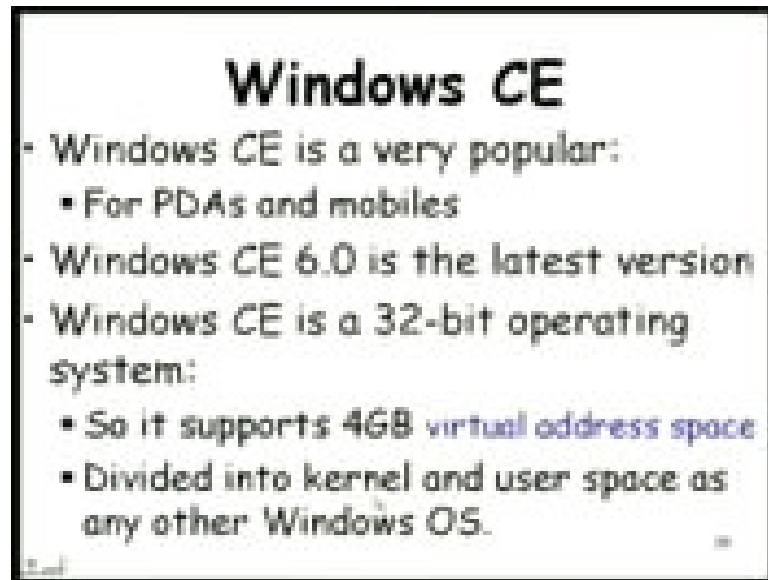
- Almost every service runs as a user process:
 - Includes a special process known as **proc** which performs process creation

03/08/10 28

Here **we will here** most of the services run as user process; here the microkernel is taken to an extreme end like, we had seen microkernels, where they provide many facilities in the microkernel itself; here even the process creation is done through proc which also runs in the user process.

Memory management also as a user process of course, with the help of some basic microkernel routines and if you are looking for a distributed application development, this can be used; QNX is naturally extended to a distributed operating system because of the strict microkernel architecture.

(Refer Slide Time: 27:35)

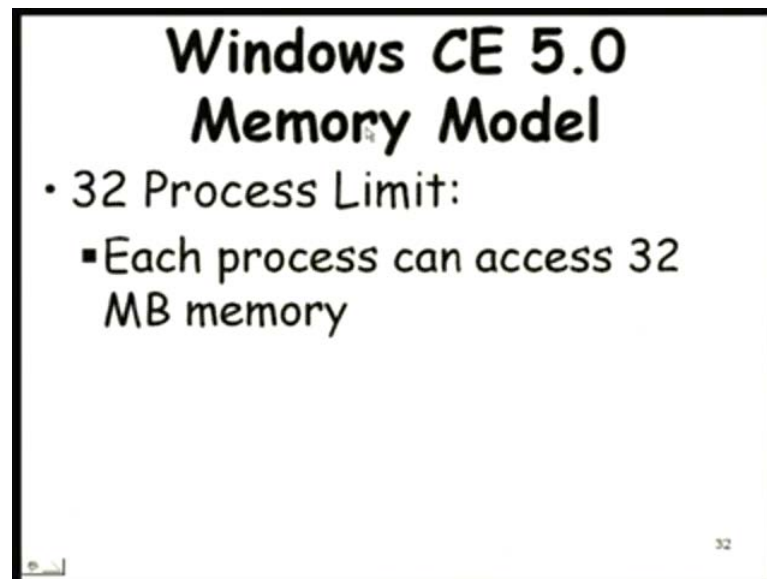


Now, let us look at another very popular operating system the windows CE, it is very popular for handhelds mobile, the portable digital assistants and mobiles. And the development started 20, 25 years back and now you have the windows CE 6.0, which is the latest version, is developed over several versions and divisions say 32 bit operating system and naturally support a 4 gigabyte virtual address space.

Again it divides into the kernel and user space just like any other windows desktop operating system (Refer Slide Time: 27:35) and it is reconfigurable, you can configure it to be a very small and feature rich; the smallest configuration is called as a Minkern only 350 k b size, but of course, most of the facilities would have been turned off no graphics, window support etcetera.

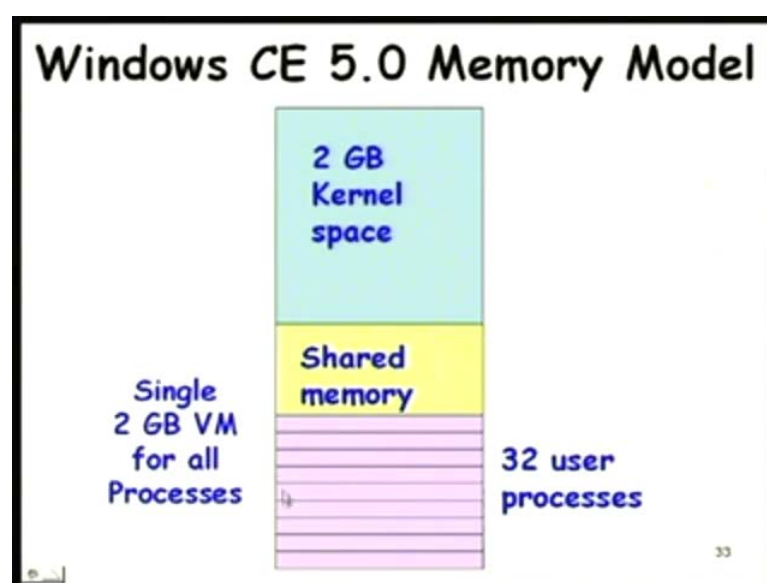
Supports multiple processes, multithreaded DLLs - dynamic link libraries - virtual memory management comes with support for large number of I/O device drivers as any Microsoft operating system. So, these are some of the important features of the windows CE.

(Refer Slide Time: 29:10)



The memory model of windows 5 had one limitation, that is, 32 process limit. So, the **maximum number of process that...** This is of course, carried over from the legacy of CE 1.0, 2.0 etcetera, you can at best have 32 processes and each process can have a 32 m b memory. So, just see here, the entire virtual space is not available, the memory is divided into slots and they use those slots. The shared memory is **half of the user space** upper half of the user space.

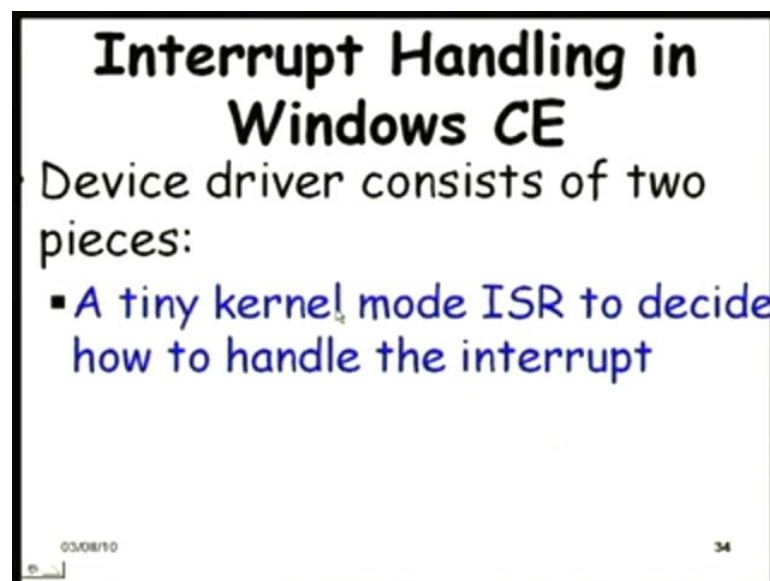
(Refer Slide Time: 30:00)



So, this is how looked like, 2 gigabyte kernel space and 2 gigabyte virtual memory for all processes, but in the virtual memory, they do not reside just anywhere, just like a traditional operating system, they have slots here for the 32 processes, each one is 32 m b and there is also shared memory; this portion is here marked for the shared memory.

So, the 2 g b user space is divided into the shared memory and the slots for the user processes; naturally a severe restriction for operating system of course, this has been removed in the CE 6.0.

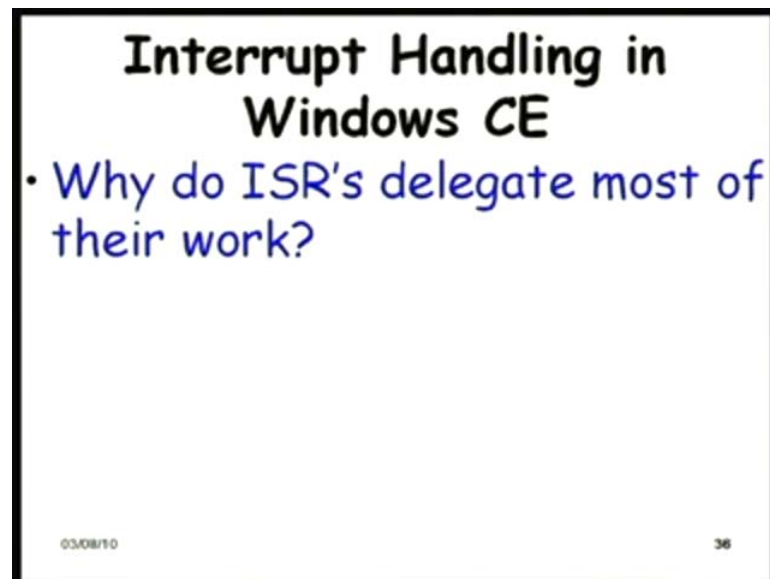
(Refer Slide Time: 30:47)



One thing that needs to be discussed is about the interrupt handling in the windows CE. The device driver as we are saying with any operating system, which can give a fast response to interrupts must have two parts: one is a tiny kernel mode ISR service routine, which basically decides how to handle the interrupts and **it can** it can queue a interrupt service thread or a deferred procedure call, which does the bulk of the interrupt handling work, but the basic work deciding how to handle the interrupt etcetera is done by the tiny kernel mode.

So, same thing is supported here which is of course, very desirable for a real-time operating system. When a interrupt occurs, the kernel saves the state of the currently executing user mode code as soon as the instruction completes, and then, the kernel invokes the ISR to handle the interrupt.

(Refer Slide Time: 32:06)



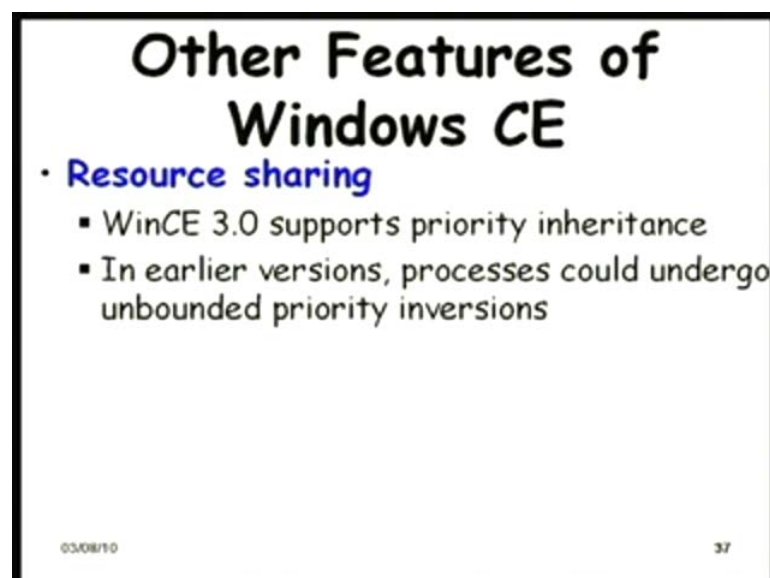
Interrupt Handling in Windows CE

- Why do ISR's delegate most of their work?

03/08/10 36

But most of the responsibilities of the ISR is delegated to a deferred procedure call. The main reason for that is that, the ISR's typically run in the kernel mode; they have a very small stack and the local variables are limited. And also since it is kernel mode, most of the interrupts are masked by making **it** most of the processing in the deferred procedure call, you can reduce the time, the interrupt is must.

(Refer Slide Time: 33:00)



Other Features of Windows CE

- **Resource sharing**
 - WinCE 3.0 supports priority inheritance
 - In earlier versions, processes could undergo unbounded priority inversions

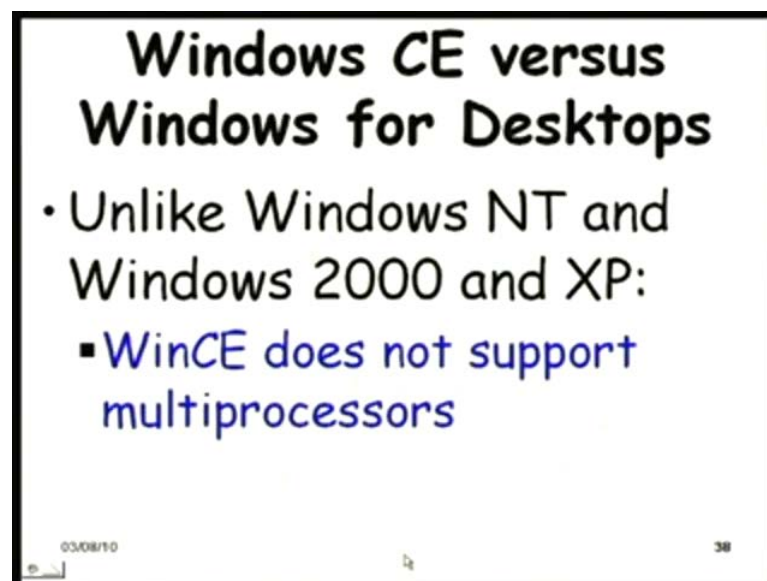
03/08/10 37

Other features of windows CE that we need to discuss is, supports priority inheritance for resource sharing among real-time tasks, but before windows CE 3.0 and even the

windows NT and so on, it could undergo unbounded priority inversions. While discussing Windows NT, we had discussed about this aspect that, resource sharing even though the Windows NT had many desirable features **for a real-time task** for real-time application, for example, real-time priority levels as well as dynamic priority levels and so on, it had difficulty with resource sharing; unbounded priority inversions could occur.

Fast context switch and each process has its own page table sorry, the Windows 2000, which has each process its own page table, but here, they share a single page table.

(Refer Slide Time: 34:10)



And of course, does not support multiprocessors, but again we cannot just say that it does not support multiprocessor, because you are just talking of some versions of the WinCE, maybe in the later versions they support **multiprocessors**, the multi core processors.

And as we are I think, we have just mentioned this one, that Windows 5.0, Windows CE 5.0 and earlier to that only 32 simultaneously executing processes were allowed and this is too small number of processes, especially for larger applications considering that the kernel itself takes up many processes. The kernel itself, the device drivers, the file system, etcetera they all are counted as processes.

(Refer Slide Time: 35:30)



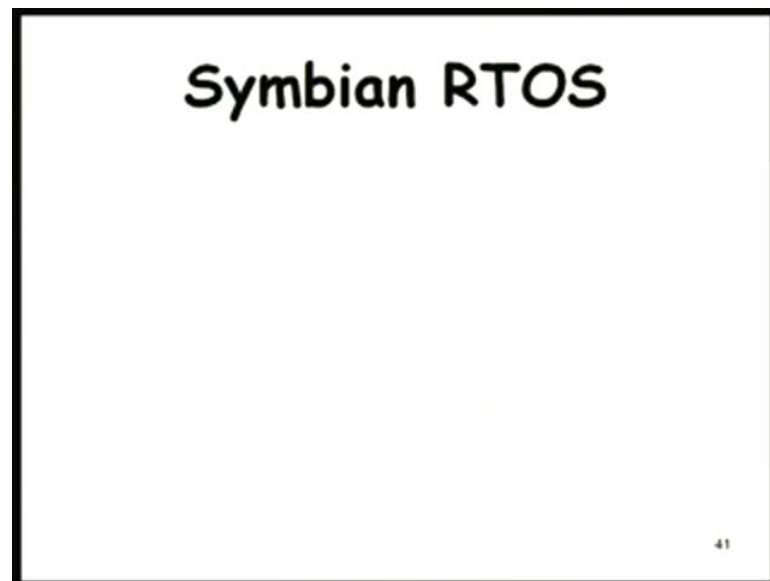
So, this 32 gets reduced and as we are saying that, in 5.0 and before that, the processes had 32 m b address space limit compared to 2 g b for the desktop versions. And the Windows CE.NET, easy to develop application, supports Bluetooth, IPV6, etcetera is part of the operating system. IP version 6 supports Kerberos security, because mobile applications security is important; the secure socket layer, Kerberos, etcetera are supported.

(()) kerberos security

So, this is the security service and it is actually a several hours of discussion on Kerberos and even secure socket layer, we can have several hours of discussion on security. So, maybe we will not spend time now; if possible towards the end, we will see if we can discuss little bit or an hour or so on the security services in operating system, but at least you can just read this up, some basic aspects of Kerberos and secured socket layer, how this provide security and how they help with authentication and crypto attacks and so on.

So, you can build and test applications on Windows 2000 and XP desktops using emulation for Win CE.NET helps in application development on a larger platform **win c**
e sorry Win 2000 or Win XP and then make it run on win CE.

(Refer Slide Time: 37:10)

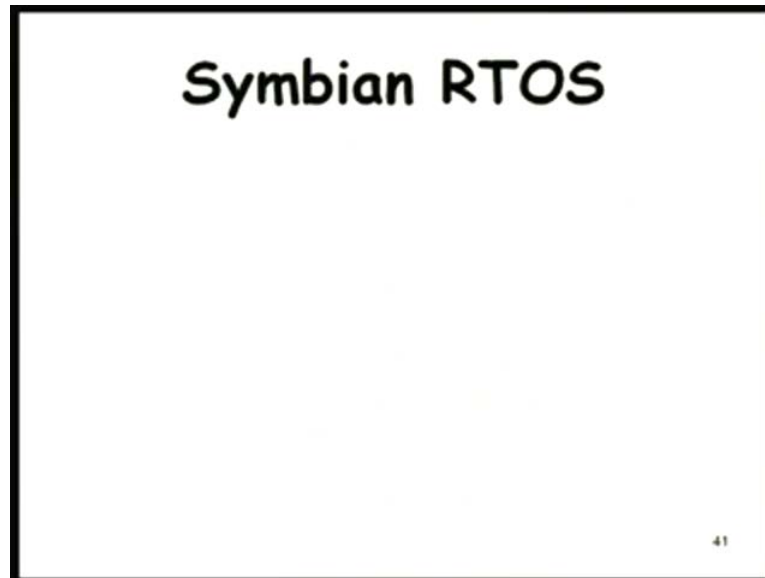


Another popular operating system which **is** becomes very increasingly or it is suddenly gained large popularity, the Symbian operating system specially designed for mobile phones and it become very popular once it has acquired by Nokia in 2008 and not only that, Nokia made it an open source from February 2010.

So, now, anybody can develop applications, can extend the kernel and so on and large number of **operating system sorry** the mobile phones, almost half of mobile phones, you see people holding, they run the Symbian operating system, **it is the** it is not a exaggeration to say that it is the world's most popular mobile phone operating system.

Next to the Symbian, I think **is** the Windows CE which has only some 10 percent or so or 12 percent users as a mobile phone operating system.

(Refer Slide Time: 38:20)



So, Microkernel design and like other operating systems, it is multitasking, has memory protection and one important features is that, the CPU is switched into a low power mode whenever an application is not directly dealing with an event. So, it keeps track of which application is dealing with what event and if none of the applications is dealing with an event, it immediately switches onto a low power mode, very important feature required in any mobile phone.

(Refer Slide Time: 39:04)



And another one which developed just couple of years back, one or two years back is the android operating system. Again a mobile phone operating system specifically targeted towards the mobile phone based on the Linux kernel. See, here, the Linux kernel, all the services are there on that specific libraries and specific applications, for example, security and so on are built on this. It is just couple of years old, but its usage is growing very rapidly, free operating system again and possibly because of the popularity that the android operating system, the interest that it was developing, it was generating may be the Symbian they made it open source just by looking at the android operating system; the way it was increasing in popularity just a couple of years old, but already has captured some two to four percent of the mobile phone market.

So, these are two operating systems, possibly they are going to become more and more important and popular in **in** not only the mobile phone space, but other real-time applications as well, because other applications also they need capability to connect with other devices and so on the future applications and lot of literature available on these two the Symbian and the android operating system. So, I would request you to prepare a **write up** survey the material available on the net and prepare report which is basically a bonus problem spend some time with a reports available understand their features their implementation compare their features and advantages and disadvantages and please submit it before the end semester.

(Refer Slide Time: 41:16)

How to Select a Computer System?

- Suppose you are asked to select a computer system for a specific application:

03/08/1044

So, **we will** we have briefly surveyed the features that are available in some of the popular real-time operating systems; there are many more real-time operating systems that are available, some are free open source, some are priced, but it is very difficult to include all of them, there are several dozens of them. So, we have just looked at some eight ten operating systems, which are highly popular, used extensively in many of the embedded and real-time applications.

Now, let us try to answer one question. Suppose you are asked to select a computer system for a specific application, let us say your college teacher in college or may be a engineer in a company, and then, your principal or your manger in your company comes and say see we need to buy a computer to do an application, let us say we need to implement a web service; we need to host the web pages of our organization of our company. So, can you select a computer for this application? And of course, the cost etcetera would be given to you, **that** see we have budget of a lakh or something one lakh and then the application that we were trying run is a hosting a web page of the organization, it should be a web server basically.

So, naturally you will find that there are dozens of manufacturers who can quote for your product and satisfying your cost, then how do you select the computer out of all those which satisfy the cost etcetera criterion, how will you select an application which will provide the best service and the best service is let us say provide the fastest **wave application sorry** web server for let us say a fixed number of users, let us say the manager or the principle might say that see we will get generate only about 300 bits per second or something 300 bits per minute.

So, what do you do to select?

Yes.

Processor speed which will not (()).

So, how do you measure processor speed?

Million instructions per second.

Mips.

Mips.

So, he says that we should look at the Mips rating of the processors priority, and then, select. What about any other have different ideas, how do we select the computer for this? Because it might happen someday, your boss somebody is going to come and say that, see you are the responsible person who will select the computer for a specific application.

Actually for web service that hits my request to retrieve the information.

Yes.

For that purpose a d m a computer all depending always if it is not an always a processor is used always.

True I mean, see processor rating is one where it says that how many instructions it can complete per second that is what he says millions of instruction processed per second and of course, that would also include the I/O instructions say in some, where I/O is there. So, not clear actually what you have to saying, is there anyone else who would like to tell anything.

Memory sir.

Yes.

Memory.

Memory yes, how does memory help?

Suppose in put an upper bound and how much memory can be used a (()) memory?

No **no** he said that we should look at the memory. So, what should we look at memory would? You **use** insist that we should have 2 g b memory, **4 g b memory or...**

Speed of the memory as well as the size of the memory.

I mean what you will try to maximize both is it? **is it.**

(()) between a flash memory and the permanent memory.

So, he says that not only the main memory, he will also look cache. So, then I mean how do we look at it? You just keep on comparing, let us say one computer has more main memory less cache memory and another one has more cache memory less main memory. So, how do you compare all this?

Depending upon how much data do we have?

Say data you know web service for this thing, an organization it might have some let us say 10000 pages h t m l 10000 HTML pages. So, that is what the data would be.

Sir, we have to look for what is the basic requirement of our application.

What what do you mean basic requirement of the application?

The minimum space for (()) web server.

Space you mean hard disk or memory or memory.

No memory is virtual memory, it does not really...

Cache.

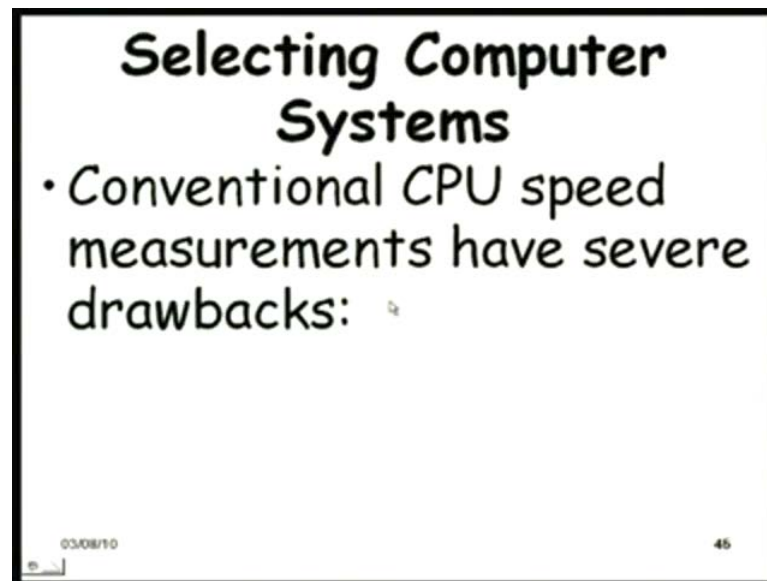
Cache.

With multi code it is having any two double caches?

No, extremely confusing. See, all these are all virtual memory and then the cache you know, it does not really fit everything.

Anyway, so, let us look at, see all of you are giving very divergent answers. So, let us see what you do?

(Refer Slide Time: 47:00)



As he said to start with that, we can use Mips, flags, etcetera, but these are highly misleading and indicate almost nothing that is why I think somebody said that, these are more misleading or misleading information about processing processor speed; somebody gave that acronym Mips is a misleading information about processor speed why is it misleading?

But sir, highly Mips rating does not guarantee that CPU performance will be more.

High Mips rating does not guarantee that c p u performance will more why is that?

Average mips will be more.

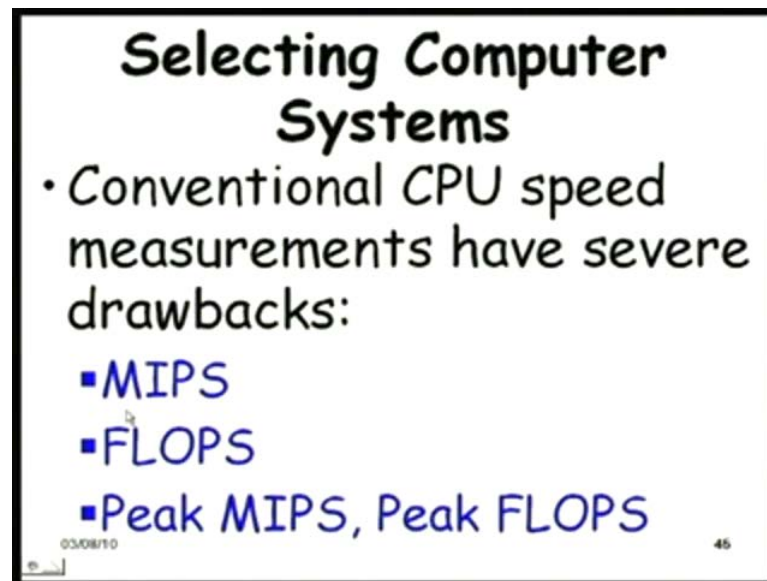
This mips is average you know millions of instructions. So, you run the instructions, and then, find out how many millions of the instructions execute per second that is the average basically per second means average.

Then ultimately through put should be mips alone may not guarantee the throughput.

Why?

We does not say that specify the instruction (()) that is getting (()).

(Refer Slide Time: 48:10)



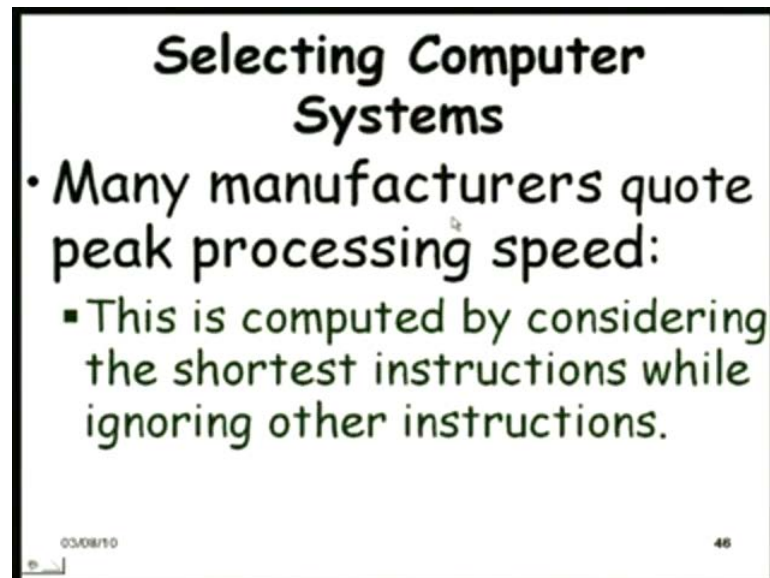
So, the Mips as he says that, in Mips it does not say which instructions are considered to compute the Mips rating and typically the vendors use the shortest instructions. See, all different instructions take different number of cycles, some the I/O instructions, etcetera may be taking hundreds or thousands of cycles and some instructions maybe taking ten's or few cycles and they deliberately use the shortest instructions; they select those instructions. So, basically determine Mips by running some program, that is, how we determine the Mips and they select those instructions are though that kind of program, where you have some instructions , shortest instructions are executed.

Naturally Mips and flops are highly misleading; they do not tell about **the** even the average performance respect to all the instructions only the peak performance. So, if your application used only those shortest instructions, then what would be your performance, that is, what it basically boils down and everybody notice that and later the vendor started coating the performance in terms of peak Mips and peak flops.

They agreed that see the Mips is misleading. So, if not much use and peak Mips and peak flops is possibly a more meaningful thing, but then even if you know the peak Mips and peak flops, your application is not going to use only those shortest instructions isn't it? So, just by knowing the peak Mips and peak flops does not help might choose a computer which has a high peak Mips and high peak flops rating. But finally, when you

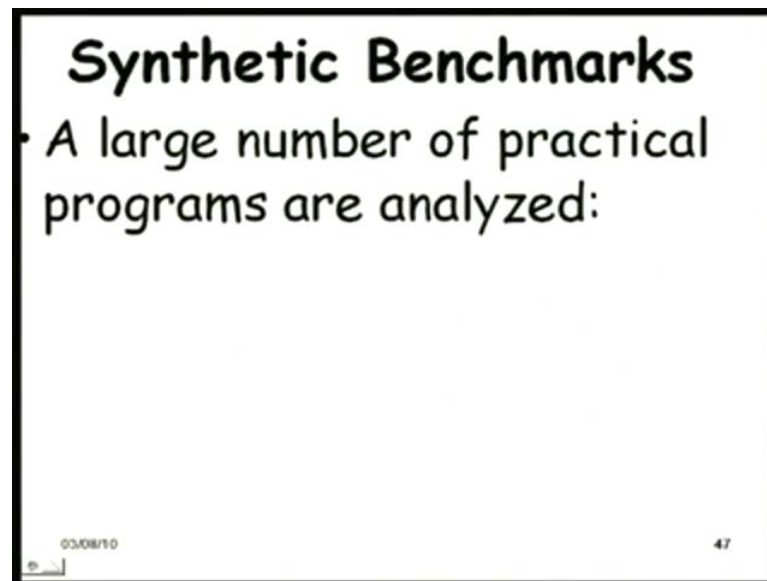
run your application, you find that it is running extremely slow and even much worst than some computer which had a very low peak Mips and peak flops rating.

(Refer Slide Time: 50:10)



So, that is what this slide says that many manufactures quote the peak processing speed by considering the shortest instructions and ignore the other instructions, which take large time very misleading and later the synthetic bench marks were developed to overcome the problem of the Mips rating. Because Mips does not say anything at all, if you **if you** choose based on that, you will be highly disappointed that your application will even run slower than the slowest Mips computer. If you have no choice of the several and you choose the Mips rating, the one that has highest Mips rating you will be disappointed that possibly the one which have the slowest or the smallest Mips rating your application could have also run faster than that you could observe that it is running faster.

(Refer Slide Time: 51:10)



So, the synthetic bench marks are developed in nineteen eighties towards the end of eighties many synthetic benchmarks existed the idea behind synthetic benchmarks is that in a typical application when you run a typical means, you run different applications and then average out that is what is a typical or a average application. You run different types of application and then keep track of what kind of instructions is executed with what frequency.

So, basically you find out the average frequency of execution of different instructions when you run an application and then a synthetic program is written it is a synthetic program, because it does not do any meaningful thing it just has that distribution of the instructions.

For example, ALU instructions ten percent I/O instructions, ten percent conditional instructions twenty percent and so on. So, a synthetic program when you run it, you find that the instructions that are executed by the synthetic program, the average distribution of the instructions is the same as a typical application. You run different types of application, networking application, there data processing application using some d b m s then user interface or browser and so on, various types of application you run and find the average distribution and then develop the synthetic program which has similar distribution. So that if you run the synthetic program on different computers, then you

will have some idea that how the average program will run. So, the vendors, they started quoting on how many of those synthetic programs, it can run per second.

So, the typical synthetic benchmarks were the whetstone and Dhrystone, LINPAC etcetera these were popular benchmark programs synthetic benchmark programs and the vendor started quoting in terms of the benchmark programs saying that how many whetstones per second how many Dhrystones per second etcetera.

But again this had a problem the problem is that the code for this is source publicized the source code is publicized the whetstone, Dhrystone, Linpack, etcetera is publicized and the vendors soon they optimize their compilers for these source code.

For example, if there is a instruction here which has no always evaluates to true and if condition instruction always evaluates to true, then they remove the if condition, some simplifications they did and of course, later the spec benchmarks are developed and our focus is on benchmarking real-time applications, real-time computer systems and real-time operating system.

(Refer Slide Time: 54:10)



So, we will see how what needs to be taken into account to benchmark the real-time operating system performance in the real-time computer system performance. So, we will stop today. Tomorrow we will continue from this point and try to see how to benchmark a real-time computer system and real-time operating system. Thank you.