

Real-Time Systems
Professor: Durga Prasad Mohapatra
Department of Computer Science and Engineering
National Institute of Technology Rourkela
Lecture 41: Windows as RTOS

Good morning to all of you. So last class we have discussed about what is the architecture of Unix, whether Unix is suitable for real-time applications. So, today, we will take up about whether Windows can be used for real-time application systems. So, let us say about Windows as a real-time operating system.

(Refer Slide Time: 00:36)

CONCEPTS COVERED

- Microsoft Windows as RTOS
- Evolution of Windows
- Windows NT & XP
- Deferred Procedure Calls (DPC)
- Windows NT vs UNIX

KEYWORDS

- Microsoft Windows
- Windows NT
- Windows NT Priority Levels
- Interrupt Processing
- Deferred Procedure Call (DPC)

The slide content is presented in two sections. The top section, titled 'CONCEPTS COVERED', lists five topics: Microsoft Windows as RTOS, Evolution of Windows, Windows NT & XP, Deferred Procedure Calls (DPC), and Windows NT vs UNIX. The bottom section, titled 'KEYWORDS', lists five terms: Microsoft Windows, Windows NT, Windows NT Priority Levels, Interrupt Processing, and Deferred Procedure Call (DPC). Both sections include a small video inset of the professor and logos for NITRR and NPTEL at the bottom right.

We will discuss about this Microsoft Windows as real-time operating system, the evolution of Windows. We will see, the two other versions of Windows like Windows NT and XP,

Windows XP. We will see something about DPC, that is, Deferred Procedure Calls, and then you will compare what we have discussed Unix with Windows NT. So, so these are the key words we will use in our talk today. Let us see the beginning.

(Refer Slide Time: 01:04)

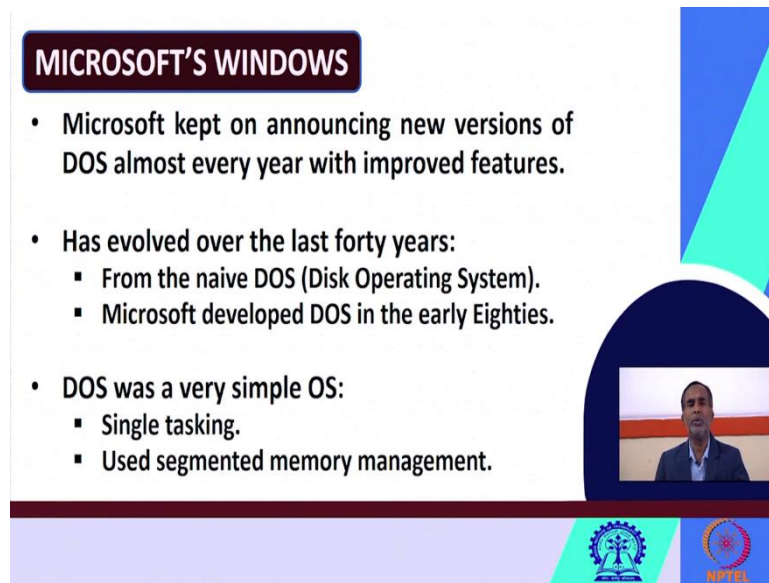
MICROSOFT WINDOWS AS RTOS

- Sometimes it may be required to use Windows as RTOS.
- **Windows NT series:**
 - More stable than the Win98 series.
- POSIX support.
- Supports Multithreading.

So, sometimes, it might be required to use Windows as the real-time operating system because nowadays Windows is very popular in almost all the, all of the desktops we are using Windows. So, sometimes, it might be required to use a window as a real-time operating system. You know, that Windows having different about versions, and currently, we are using the Windows NT series 5. The Windows NT series is more stable than the earlier Windows 98 series, I will give the evolution.

So, Windows is it support POSIX. What is POSIX, we will discuss in the next class? Windows is POSIX compliance, and it also support so multi-threading. So, these are the some of the aspects of why we will consider windows for real-time applications.

(Refer Slide Time: 01:50)



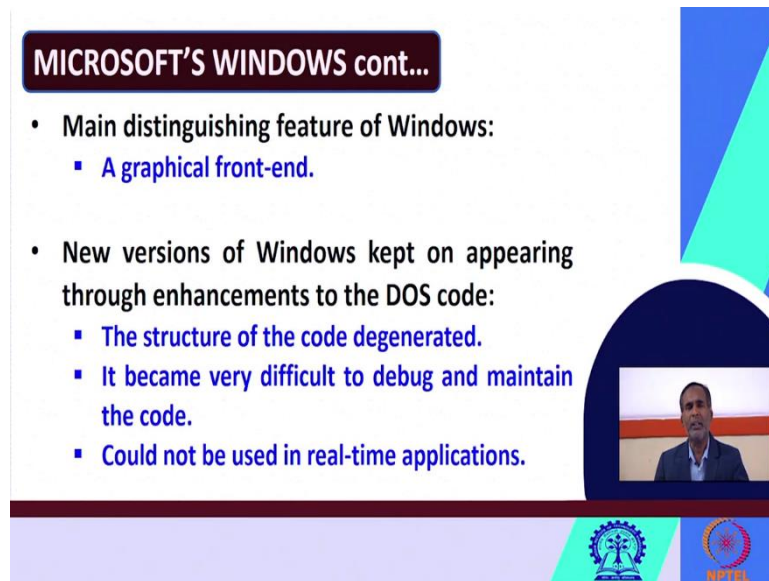
MICROSOFT'S WINDOWS

- Microsoft kept on announcing new versions of DOS almost every year with improved features.
- Has evolved over the last forty years:
 - From the naive DOS (Disk Operating System).
 - Microsoft developed DOS in the early Eighties.
- DOS was a very simple OS:
 - Single tasking.
 - Used segmented memory management.

You know that Microsoft it kept on announcing new versions of the DOS, almost every year with improved features. So, this Microsoft Windows has evolved over the last 40 years, and from the disk operating system till now. So, this Microsoft developed this DOS when, in the early 80s. So, this disk operating system was a very simple operating system. It was based on single tasking.

Also, it used segmented memory management. I hope you know what is paging segmentation, segmented memory management, single taskings you have already known in the basic operating system paper. So, that is why we said that DOS is a very simple operating system.

(Refer Slide Time: 02:35)



MICROSOFT'S WINDOWS cont...

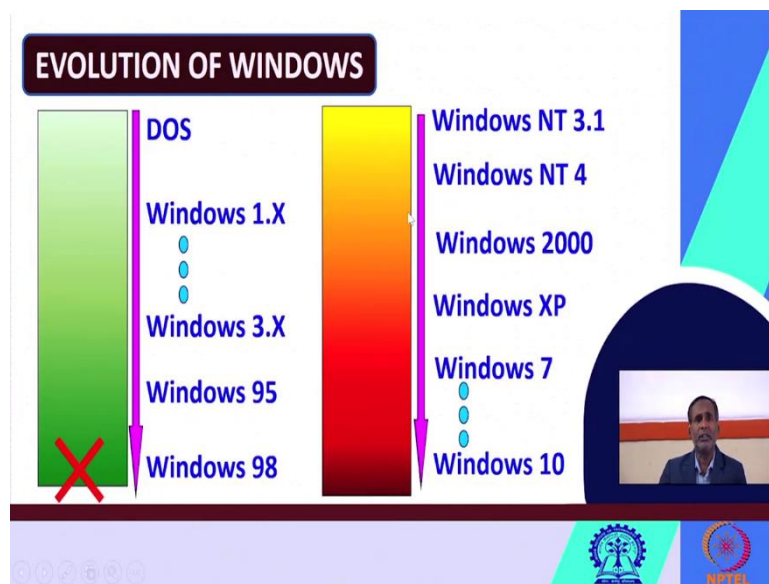
- Main distinguishing feature of Windows:
 - A graphical front-end.
- New versions of Windows kept on appearing through enhancements to the DOS code:
 - The structure of the code degenerated.
 - It became very difficult to debug and maintain the code.
 - Could not be used in real-time applications.

The slide features a dark blue header with the title in white. The main content is on a light blue background with a dark blue sidebar on the right containing a video inset of a man speaking. The footer has a light blue background with logos for IIT Bombay and NPTEL.

The main distinguishing feature of Windows is what? A graphical front end, which we are not finding in case of Unix. So, the most important or distinguishing feature of Windows, is that, it has a graphical front end, it has a better GUI, graphical user interface. The new versions of Windows they kept on appearing through enhancements to the DOS code. So, whatever I have already told you, first it was developed DOS, then enhancements are made to the DOS code.

So, these new versions of Windows right now we will see they kept on appearing through enhancements to the DOS code. So, in case of this DOS code when it was developed the structure of the code that degenerated. And since the structure of the code was degenerated, hence it became very difficult to debug and maintain the code. So, it could not be used in real-time applications. So, the traditional disk operating system, it could not be used in the real-time applications.

(Refer Slide Time: 03:28)



Now let us see, about how these windows have been evolved during the last 40 years. You know that initially they have developed this disk operating system, then gradually they have developed Windows 1.X, 2.X and other versions, and then, they have developed Windows 3.X. Then in 95 they have developed Windows 95 with some advancements with some enhancements then in 98 they have developed the version Windows 98.



And then this was the DOS series. This DOS series was almost stopped. We will see, why it was stopped. Then, they have not further extended the windows 98, rather they have started with another version called as Windows NT. So, they have started with the Windows NT 3.1 then Windows NT 4 then Windows 2000, then that was in year 2000 then Windows XP, Windows 7 like that and almost around now we are using Windows 10.

So, you will say this DOS line of products they are stopped in after Windows 98. Then then they have not made any enhancements, they have not made any extensions to Windows 98. Rather they have developed another series of products that started with the Windows NT 3.1 then Windows NT 4 like that and now we are using Windows 10. This is how this Windows have enabled using the last 40 years.


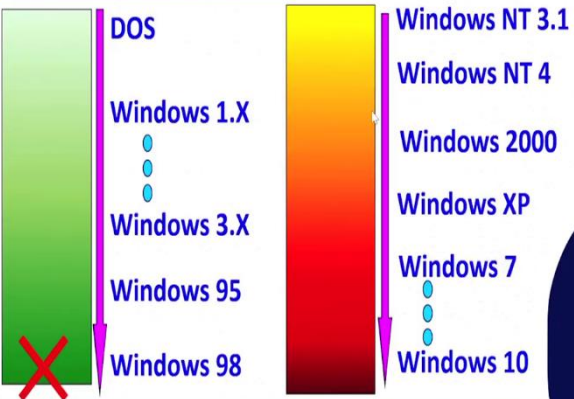
(Refer Slide Time: 04:51)

MICROSOFT'S WINDOWS

- Windows code was completely rewritten:
 - Windows NT system was developed.
 - More layered than microkernel design.
- Windows NT system:
 - Much more stable (does not crash) than the earlier DOS-based systems.
- The later versions of Windows:
 - Descendants of Windows NT.
 - The DOS-based systems were scrapped.



EVOLUTION OF WINDOWS



Now let us say, look a little bit about this Microsoft's Windows. This Windows code was completely rewritten because you see that this DOS series now it was almost scrapped. So, what do they have started, they have started writing code, they have completely rewritten the code for Windows NT, so that is what I am saying here.

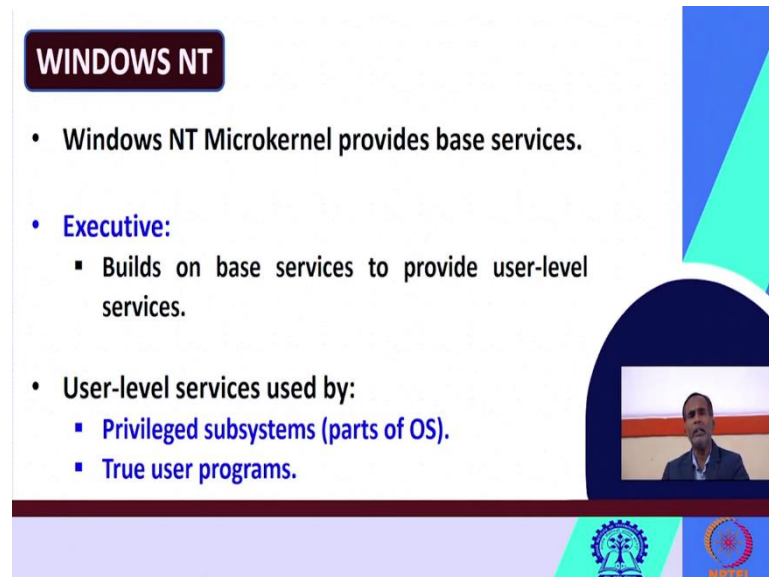
Windows code was completely rewritten this Windows system was developed. So, they have scrapped this Windows 98 and they have started with Windows NT version. Windows code was completely rewritten, this Windows NT system was developed and this Windows NT system was more layered than a microkernel design. Basically, Windows NT it uses this microkernel architecture that we have seen in the last class, and here, this Windows NT it is more layered than even the microkernel design.

This Windows NT, it is more layered than the microkernel design. This Windows NT system it is more stable than the earlier DOS-based systems. What do you mean by more stable? It does not crash, it does not have frequently crashed, Windows NT system like this DOS-based systems they crash. So, these Windows NT system, it does not frequently crash in comparison to DOS-based system that's why we say that this Windows NT system is more stable than the DOS-based systems.

So, the latter versions of Windows. So, what we will see, the current versions of the Windows or the later version of the Windows after the DOS, so they are descendants of Windows NT not descendants of DOS. You can see this previous, so the later versions of Windows, they are not descendants of DOS, they are the descendants of Windows NT. So, Windows 10 you can see, this is not a descendant of DOS, rather it is the descendent of Windows NT.

So, and the DOS-based systems are completely scrapped. So, all the products from this, all the products in the DOS series, they were scrapped and then they started rewriting the code. And whatever we are using today, they are all descendants of Windows NT.

(Refer Slide Time: 07:00)



WINDOWS NT

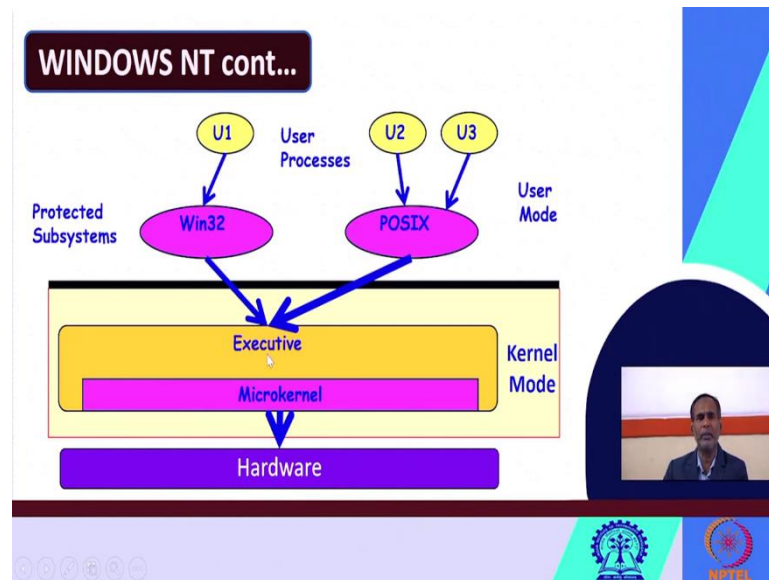
- Windows NT Microkernel provides base services.
- **Executive:**
 - Builds on base services to provide user-level services.
- **User-level services used by:**
 - Privileged subsystems (parts of OS).
 - True user programs.

The slide features a blue and green geometric design on the right side. A small video inset in the bottom right corner shows a man speaking. At the bottom, there are logos for a university and NPTL.

Now, let us see a little bit more or defer about Windows NT before going to see whether it can be used for real-time applications or not. This Windows NT microkernel it provides the best services. The Windows NT microkernel, it provides you what I have already told you the monolithic architecture and microkernel architecture in the last class.

So, Windows NT microkernel it provides what, Windows NT microkernel, it provides all the best services. And in the Windows NT I will show you the architecture there is one component called as executive. Let us first see the architecture.

(Refer Slide Time: 07:32)



So, this is this windows architecture you can see. There are two modes you can see, the kernel mode and the user mode. In the user mode, all the user processes will be there. And besides that, here Win 32 and POSIX are there. So, this the user mode, you can say these are the protected subsystems.

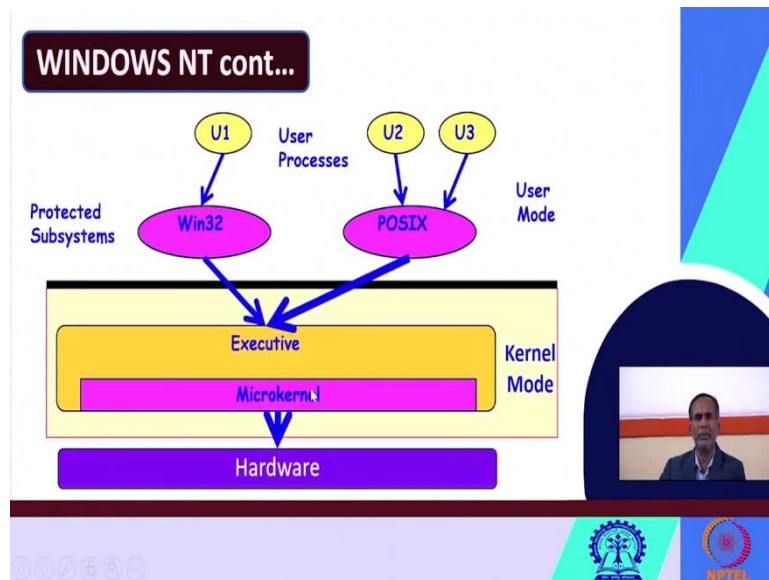


The user processes in the user mode, they can interact with the kernel mode through Win 32 and POSIX. You can see in the kernel mode there are two important components, executive and microkernel. So, microkernel is not here mixed with executive, it is separated from the executive.

So, this user processes in the user mode, they contact, they communicate with this kernel mode or they interact with the kernel mode that is the executive and microkernel through what through Win 32 and POSIX. And in turn, microkernel, it interacts with the hardware. So now let us see, two important terms we are observing here. So, POSIX you will discuss in the next class. This executive and microkernel let us see. So, what this Window executive does.

(Refer Slide Time: 08:39)

WINDOWS NT

- Windows NT Microkernel provides base services.
- **Executive:**
 - Builds on base services to provide user-level services.
- User-level services used by:
 - Privileged subsystems (parts of OS).
 - True user programs.

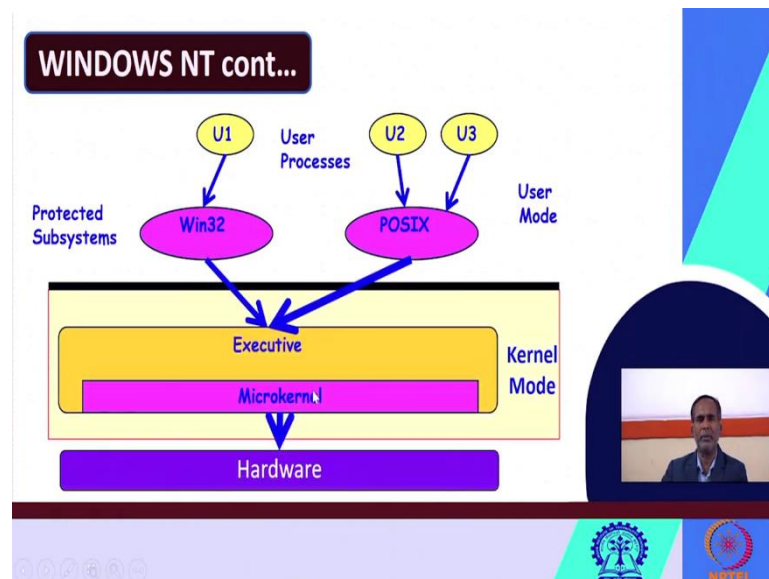


Windows executive it builds on the base services to provide the user level services. So, Windows executive it builds on what, it builds on the base services to provide the user level services that I have already shown here. So, it builds this executive, it builds on the what, the base services. The base services may be provided by the microkernel and why it builds on the base services because to provide the services to the what user level.

I have already told you that Window NT microkernel provides the base services. And then this you can see here that the microkernel provides the best services, the microkernel architecture we have seen already earlier, then the executive it is built on the base services in order to provide the user level services.

So, the user level services that they are used by what, the user level services are used by the privilege subsystems. The user level services are used by the privileged subsystems, which are the parts of the operating system, these are the true user programs. So, I have already explained to you this Windows NT architecture. So, this already we have discussed.

(Refer Slide Time: 09:47)



NT EXECUTIVE

- Provides higher level services than the microkernel.
- Runs in kernel mode:
 - But separates from the microkernel itself.
 - Makes it easier to maintain.
- Built of independent modules:
 - All preemptible and pageable.

Then let us see more about this Windows NT Executive. What does it do actually? I have already told you, it provides high level services than the microkernel. So, I have already told you microkernel provides the base services. So, Windows NT Executive it provides higher level services than the microkernel, and it runs in the kernel mode, we have already seen in the figure. This executive where does it run?

It runs in the kernel mode. So, NT Executive it runs in the kernel mode, but it separates from the microkernel itself. It is not mixed with the microkernel. The Executive it separate itself from the microkernel. Why? So, it runs in the kernel mode, it separates from the microkernel itself. Why? In order to make it easier to maintain.

So, since it is separated from the microkernel, it is easier to maintain the executive. It is built-up independent modules and all the modules are preemptable and pageable. So, this Executive it is built up some independent modules which all are preemptable pageable. I hope, you already you have known about what is preemption, what is paging, what is segmentation, you have already known earlier.

(Refer Slide Time: 11:07)

WINDOWS NT

- 32 levels of priority divided into
 - Real-time class.
 - Dynamic class.
 - Idle class
- A task cannot move between these classes.
- The priority level of a task changes:
 - When you iconify it.
- Processes such as screen saver use **priority class idle**.

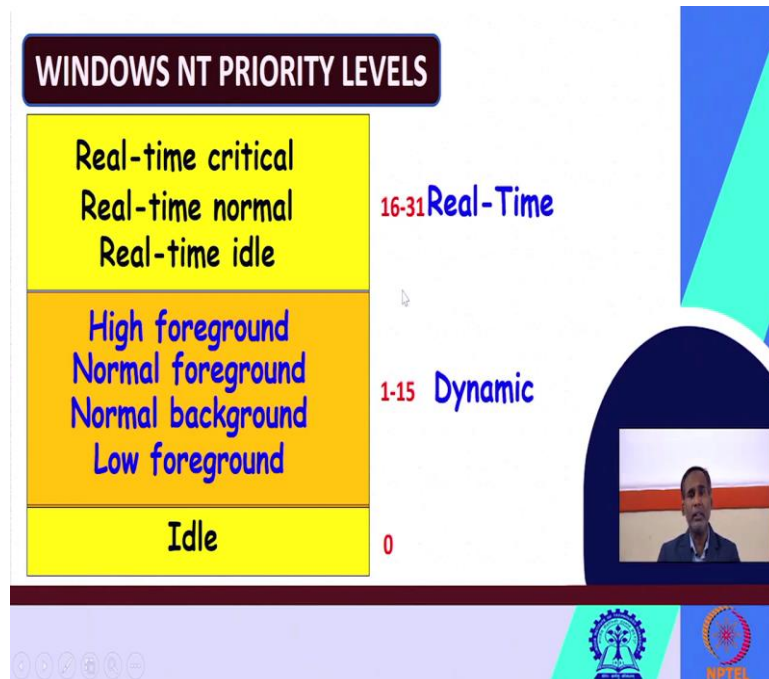
The slide features a dark blue header with the text 'WINDOWS NT' in white. Below the header is a list of bullet points. The first bullet point is '32 levels of priority divided into', followed by three sub-bullets: 'Real-time class.', 'Dynamic class.', and 'Idle class'. The second main bullet point is 'A task cannot move between these classes.'. The third main bullet point is 'The priority level of a task changes:', followed by a sub-bullet 'When you iconify it.'. The fourth main bullet point is 'Processes such as screen saver use priority class idle.'. The slide also includes a small video inset of a man speaking, and logos for IIT Bombay and NPTEL at the bottom.

In Windows NT there are 32 levels of priority. If we will see, we have only discussed about the priority levels while discussing about this Unix. So, similarly in Windows NT there are you will see, 32 levels of priority, and these 32 levels of priority they are divided into three important classes, real-time class, dynamic class and idle class.

So, a task cannot migrate, cannot move between these classes. So, any task it cannot move between these classes. The priority level of a task changes when, when you iconify it. So, you can, or the operating system can change the priority level of a task only when you will iconify it. Because you have already seen in case of Unix, so this is dynamically changing priority. Once a priority is given to the task, the operating system can change it dynamically.

So, here, a task cannot move between these classes, that means, changing the priority is difficult. The priority level of a task from the changes only when you iconify it. So, we will take, we will see one example of ideal class that I will say here. First let us see how the 32 levels of the priorities are divided.

(Refer Slide Time: 12:19)



So, the 32 there are you can see, some major subclasses you can see. Like normal, first is ideal normal, high and real-time. Critical means, may maybe it is more than this high. So, these are the different sub classes into which the priority levels are divided. As I have already told you there are 32 price levels it starts from 0 to 31.



So, this idle class is assigned with priority 0 then the task in between this is the second, the middle level is the dynamic. The task present in the dynamic class they are given priority in between 1 to 15 that mans, this bottom one is 1 then this top one is 15.

And the tasks, which are belonging to the real-time class they are assigned priorities from 16 to 31. So, in this way, the 32 bits sorry the 32 priority levels are divided among these classes that is normal, high and real-time. Let us take one example of one task, which may belong to the ideal class.

(Refer Slide Time: 13:26)



WINDOWS NT

- 32 levels of priority divided into
 - Real-time class.
 - Dynamic class.
 - Idle class
- A task cannot move between these classes.
- The priority level of a task changes:
 - When you iconify it.
- Processes such as screen saver use **priority class idle**.



WINDOWS NT PRIORITY LEVELS

Real-time critical Real-time normal Real-time idle	16-31	Real-Time
High foreground Normal foreground Normal background Low foreground	1-15	Dynamic
Idle	0	



So, the processes or the task such as your screensaver, they do not have, you know already about screensaver, they do not do much of the things. They will just set the screen. So, processes so just screen saver they use priority class idle. So, one example of this idle class will have taken like screens saver, this process, screensaver this belongs to the idle class, and hence its priority is 0. So, in this way, the 32 priority levels they are divided into these sub classes in Windows NT. Okay.

(Refer Slide Time: 14:00)

IMPORTANT FEATURES OF WINDOWS NT

- Supports multi-threading
- Timers and finer clock resolutions (1ms)
- Availability of RT priority levels

The slide features a dark blue header with the title in white. The content is on a white background. A small video inset of a man is in the bottom right. Logos for a university and NIPTEL are at the bottom.

WINDOWS NT PRIORITY LEVELS

Real-time critical Real-time normal Real-time idle	16-31 Real-Time
High foreground Normal foreground Normal background Low foreground	1-15 Dynamic
Idle	0

The slide shows a vertical stack of three colored boxes (yellow, orange, yellow) representing priority levels. To the right, numerical ranges and labels are shown. A small video inset of a man is in the bottom right. Logos for a university and NIPTEL are at the bottom.

Now, let us quickly look at some of the important features of Windows NT. I have already told you, it supports multi-threading then it also provides timers. What is timers etc different types of timers, we have already seen in case of Unix, it provides timers. It supports finer clock resolutions. So, in Unix we have seen there the clock resolution is of the order of 10 milliseconds.

But here, in Windows, it supports finer clock resolution of the order for just 1 millisecond so which is better than that of the windows, and availability at the priority levels. So, here, real-time priority levels are there. I mean, just it is closer to the static, but in Unix, there are dynamic

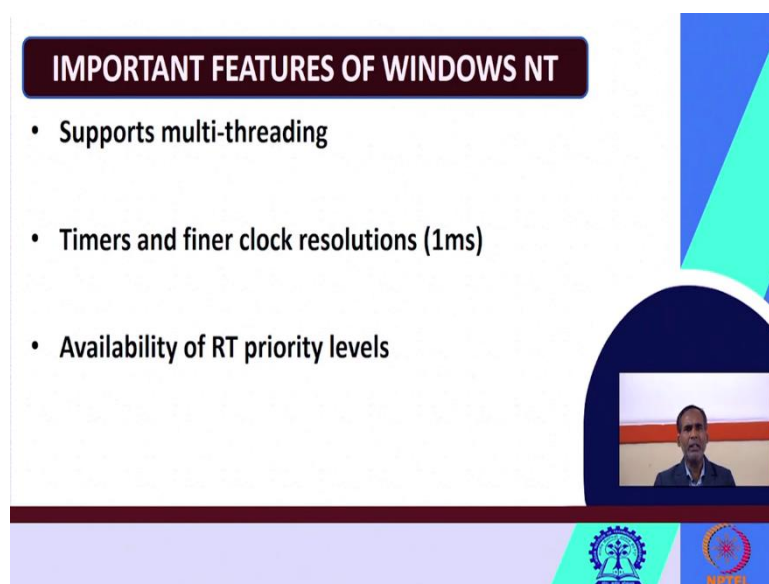
changing priority levels, but here, availability of real-time priority levels are there, which I have already shown in here.

So, these are the top-level priority levels these are applicable for the real-time classes. So, here, the real-time priority levels are available. These are some of the important features Windows NT, due to which this Windows NT may be suitable to some extent to be used in real-time applications.

(Refer Slide Time: 15:06)

IMPORTANT FEATURES OF WINDOWS NT

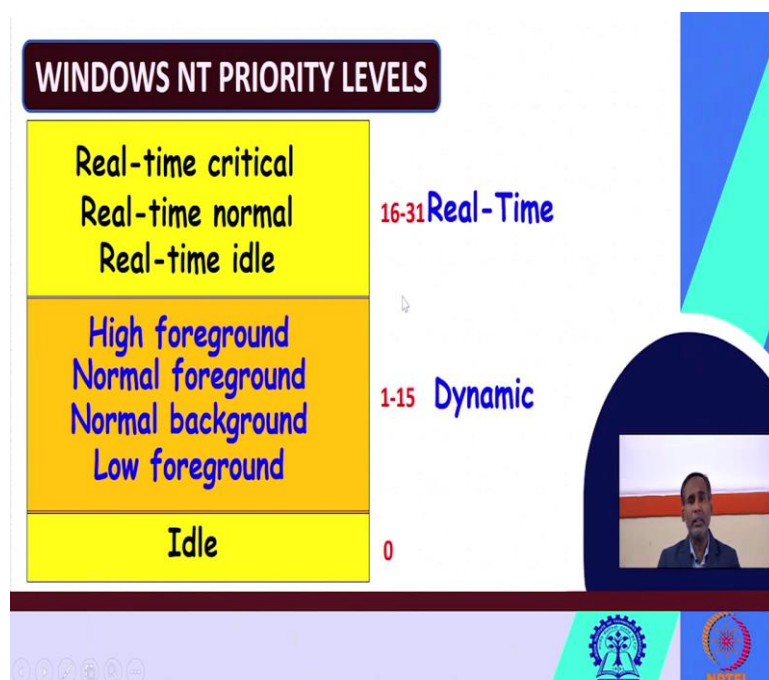
- Supports multi-threading
- Timers and finer clock resolutions (1ms)
- Availability of RT priority levels



This slide features a dark blue header with the title 'IMPORTANT FEATURES OF WINDOWS NT' in white. Below the header, three bullet points are listed in black text. The slide includes a video inset of a man in a suit on the right side and logos for IIT Bombay and NPTEL at the bottom.

WINDOWS NT PRIORITY LEVELS

Real-time critical Real-time normal Real-time idle	16-31 Real-Time
High foreground Normal foreground Normal background Low foreground	1-15 Dynamic
Idle	0





This slide features a dark blue header with the title 'WINDOWS NT PRIORITY LEVELS' in white. The content is organized into three horizontal sections. The top section, labeled 'Real-Time' (16-31), lists 'Real-time critical', 'Real-time normal', and 'Real-time idle'. The middle section, labeled 'Dynamic' (1-15), lists 'High foreground', 'Normal foreground', 'Normal background', and 'Low foreground'. The bottom section, labeled 'Idle' (0), lists 'Idle'. The slide includes a video inset of a man in a suit on the right side and logos for IIT Bombay and NPTEL at the bottom.

Now, let us see how these priority levels, they are divided, how do they work? I have already told you NT supports 32 priority levels. They are divided into three classes, I have shown, ideal, dynamic and real-time. Each process belongs to one of the following classes as shown in the figure. I have already told you each of these process, either may belong to each of these any of the class.

(Refer Slide Time: 15:30)



PRIORITY LEVELS IN WINDOWS NT

- Supports 32 priority levels.
- Each process belongs to one of the following classes as shown in the figure.
- By default, the priority class at which a user task runs is **normal**.
- Normal & High priority classes are variable type,
 - i.e. priorities of tasks in this class are recomputed periodically by the OS.
- Uses priority-driven preemptive scheduling
- Threads of RT priorities have precedence over other threads.

WINDOWS NT PRIORITY LEVELS

Real-time critical Real-time normal Real-time idle	16-31 Real-Time
High foreground Normal foreground Normal background Low foreground	1-15 Dynamic
Idle	0

Maybe idle, may be normal, may be high or may be real-time by default the priority class at which a user task runs is normal. By default, the priority class at which any user task runs is normal. The normal and high priority classes they are variable type. Normal or that means the

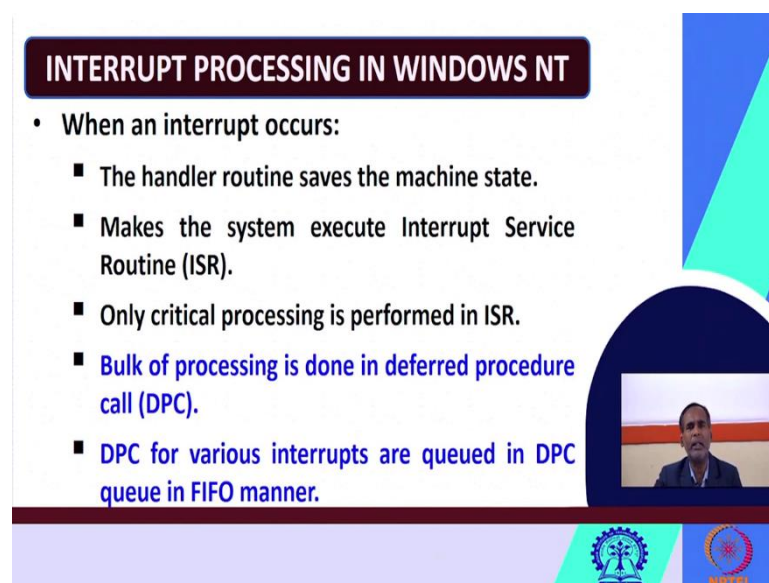
task belonging to normal and high priority classes they are variable type, that means, the priorities of the tasks in these classes that means in normal class and high priority class they are repeated, they are recomputed periodically by the operating system

We have already seen, how in Unix they are recomputed periodically. So, here, only normal and high priority classes are variable type, and these priorities of the class, tasks in this class are recomputed periodically by the operating system, but this is not applicable to real-time. There are, these our top level what real-time.

So, once you assign priority to the tasks belonging to the real-time class, they cannot be changed, they are fixed, they are static that is why it is suitable for what your real-time applications. But the normal and high priority classes the tax in normal and high priority classes they are or variable type, their priority they are recomputed periodically by the operating system. This Windows NT uses priority driven pre-emptive scheduling. I hope, you have already known about preemptive scheduling and priority driven preemptive scheduling.

That means, whenever any high priority task is running and what a low priority tax is coming, so it may be preempted so it uses priority driven preemptive scheduling. The threads of real-time priorities have precedence over the other threads. So, the threads having real-time priorities, they have more precedence than or over the other threads. So, these are some of the basic concepts about the priority levels in Windows NT.

(Refer Slide Time: 17:28)



INTERRUPT PROCESSING IN WINDOWS NT

- When an interrupt occurs:
 - The handler routine saves the machine state.
 - Makes the system execute Interrupt Service Routine (ISR).
 - Only critical processing is performed in ISR.
 - Bulk of processing is done in deferred procedure call (DPC).
 - DPC for various interrupts are queued in DPC queue in FIFO manner.

The slide features a dark blue header with the title in white. The main content is a bulleted list with blue square markers. A small video inset in the bottom right shows a man speaking. The footer contains the logos of IIT Bombay and NPTEL.

Now let us see, how interrupt processing is handled in Windows NT. So, actually when an interrupt occurs in Windows NT, the handler routine saves the machine state. So, whenever any interrupt occurs, the handler routine it saves the state of the machine or the machines state, then it makes the system execute interrupt service routine.

So, after it saves the machine state then it makes the system to execute the ISR. ISR, you know, Interrupt Service Routine. So, and in ISR what will be executed? What will be processed? Only the critical processing is performed in ISR. Only the activities which are very much critical only the critical processing activities they are performed in ISR or the Interrupt Service Routine all other activities, that means, the bulk of the processing activity the rest of the processing activities, they are drawn in deferred procedure call.

So, only the critical processing they are performed in ISR. All other are the rest of the activities the backup for the processing activities is done in DPC that is the Deferred Procedure Call. The DPC of various interrupts are queued in DPC queue in FIFO manner. So, the deferred procedure call for various interrupts where they are placed, the deferred procedure calls for various interrupts they are queued in a queue called as DPC queue in FIFO, first in first out manner.

(Refer Slide Time: 19:01)

WINDOWS XP

- Layered architecture:
 - **Executive runs in protected mode:**
 - Provides the basic system services.
 - On top of the executive, several services operate in **user mode**.
- Modular structure allows:
 - **Additional services to be added without affecting the executive.**

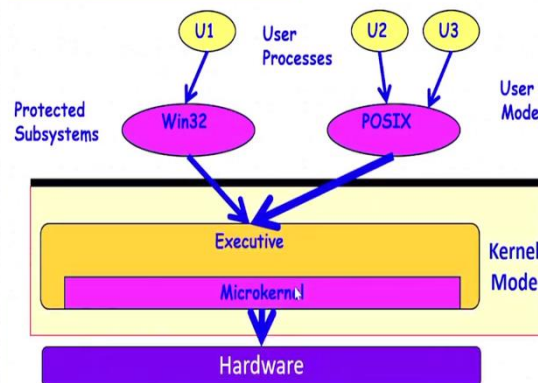
The slide features a dark blue header with the title 'WINDOWS XP' in white. The main content is a bulleted list with blue and black text. A small video inset in the bottom right shows a man speaking. The slide is decorated with blue and green geometric shapes and logos for IIT Bombay and NPTEL at the bottom.

INTERRUPT PROCESSING IN WINDOWS NT

- When an interrupt occurs:
 - The handler routine saves the machine state.
 - Makes the system execute Interrupt Service Routine (ISR).
 - Only critical processing is performed in ISR.
 - Bulk of processing is done in deferred procedure call (DPC).
 - DPC for various interrupts are queued in DPC queue in FIFO manner.



WINDOWS NT cont...



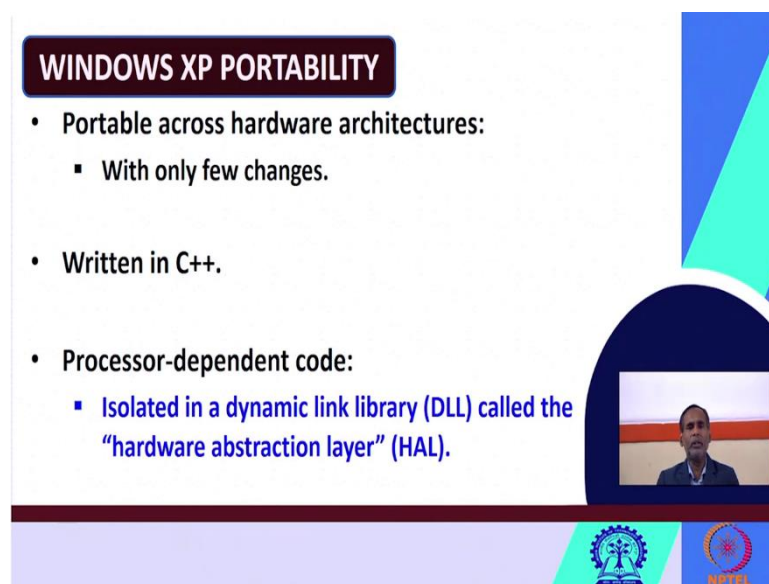
This now let us look quickly at this Windows XP. So, we have seen a little bit about the Windows NT, how the priority levels are arranged, how the inter-processing is handled like this. Now next we will see the next variant that is Windows XP. So, Windows XP has a layered architecture. So, in Windows XP, the executive runs in protected mode. So, that is the one of the difference, you can mark.

Please see, here, in this normal Windows when Windows NT we have seen this executive it runs in kernel mode, but in Windows XP in Windows XP, the executive it runs in the protected mode. Why? Because to provide the basic system services. So, basically, Windows XP has a layered architecture. Here the executive runs in protected mode. It provides that means the executive, the Windows XP Executive it provides the basic system services.

On top of the Executive several services operate in user mode. On top of this executive, several other services they operate in user mode. I have already told you this has a layered architecture, Windows XP has a modular structure this modular structure allows other additional services to be added without affecting the Executive.

So, since the Windows XP is modular in nature, this modular structure it allows the other additional services to be added or to be included without affecting the executive. This is the advantage of the modular structure that is provided in Windows XP.

(Refer Slide Time: 20:39)



WINDOWS XP PORTABILITY

- Portable across hardware architectures:
 - With only few changes.
- Written in C++.
- Processor-dependent code:
 - Isolated in a dynamic link library (DLL) called the "hardware abstraction layer" (HAL).

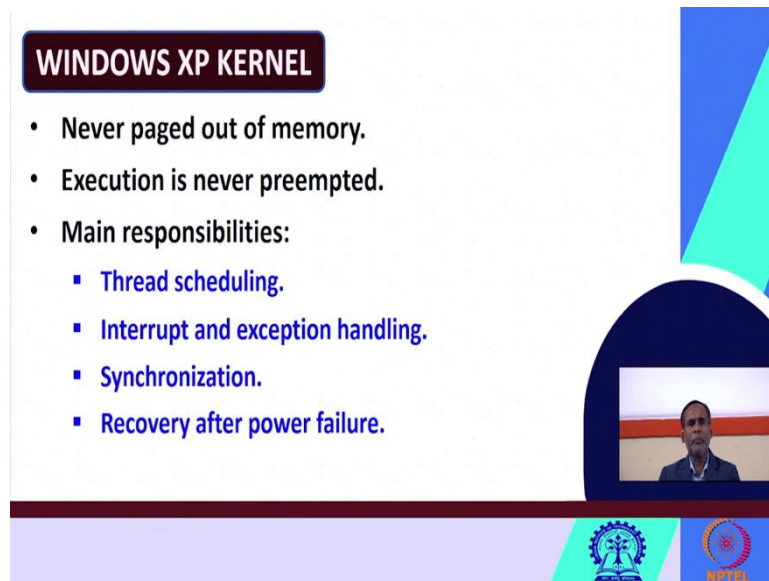
The slide features a dark blue header with the title in white. The main content is on a white background with a dark blue footer. A small video inset in the bottom right shows a man speaking. Logos for a university and APTEL are in the footer.

Now, let us see a little bit about this Windows XP portability. So, Windows XP is portable, across the different hardware architectures with only few changes. That means, with just making few changes, only making few changes the Windows XP it can be portable across different hardware architectures.

So, this was written in C++. This code Windows XP has processor-dependent code. Windows XP has processor-dependent code and this is isolated in a dynamic linking library. I hope, you must have studied about two DLL Dynamic Linking Library. So, this Windows XP has a processor-dependent code, and this is isolated in a dynamic linking library or DLL.

And this processor-dependent code or this library, this code which is isolated in a DLL is called the hardware abstraction layer or HAL. I am repeating again, this is very important, that Windows XP has a processor-independent code, which is isolated in a dynamic linking library called the hardware abstraction layer or HAL.

(Refer Slide Time: 21:55)



WINDOWS XP KERNEL

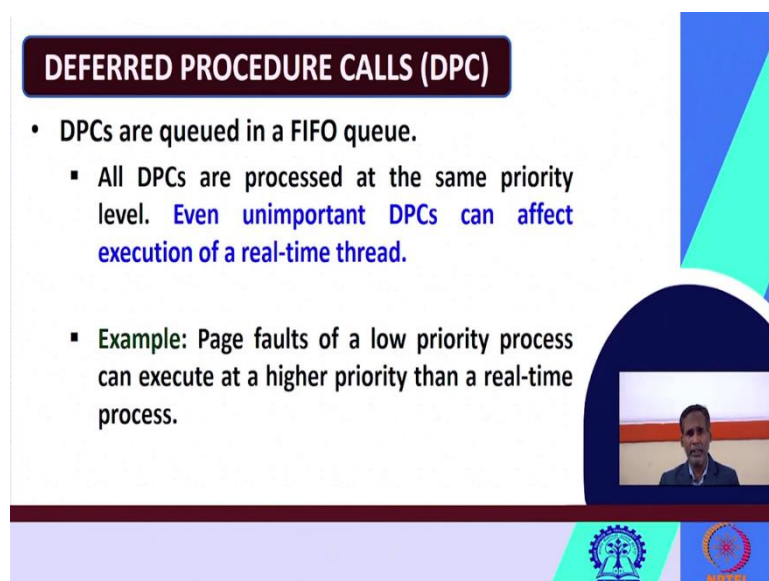
- Never paged out of memory.
- Execution is never preempted.
- Main responsibilities:
 - Thread scheduling.
 - Interrupt and exception handling.
 - Synchronization.
 - Recovery after power failure.

The slide features a dark blue header with the title 'WINDOWS XP KERNEL' in white. Below the title is a bulleted list of responsibilities. A video inset in the bottom right shows a man speaking. The footer contains logos for IIT Bombay and NPTEL.

Let us quickly look at this kernel of Windows XP. Windows XP kernel, it never paged out of memory. That is the one of the major advantage of this, so Windows XP kernel never paged out of memory. The execution is never preempted. So here, the execution is never preempted. The major responsibilities of Windows XP kernel are the followings.

It is responsible for thread scheduling, it performs interrupt and exception handling, it also is responsible for synchronization and it also takes care of recovery after a power failure occurs. So, these are some of the major responsibilities of Windows XP kernel.

(Refer Slide Time: 22:36)



DEFERRED PROCEDURE CALLS (DPC)

- DPCs are queued in a FIFO queue.
 - All DPCs are processed at the same priority level. **Even unimportant DPCs can affect execution of a real-time thread.**
 - **Example: Page faults of a low priority process can execute at a higher priority than a real-time process.**

The slide features a dark blue header with the title 'DEFERRED PROCEDURE CALLS (DPC)' in white. Below the title is a bulleted list of characteristics and an example. A video inset in the bottom right shows a man speaking. The footer contains logos for IIT Bombay and NPTEL.

Now, let us see, more about this DPC. I have already told you DPC while discussing this Windows NT. Now, let us see something more about this DPC. I have already told you the deferred procedure calls, they are queued in a queue called as a DPC queue and in a manner that FIFO manner.

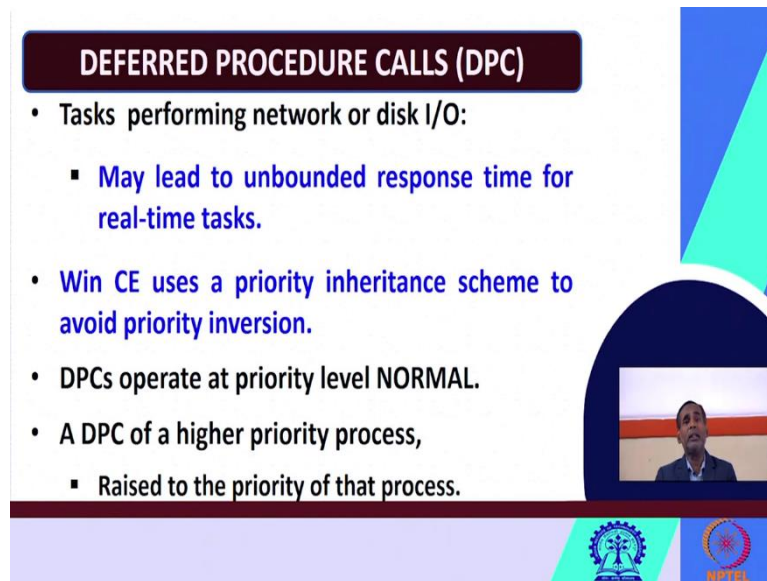
All the DPCs are processed at the same priority level. All the Deferred Procedure Calls they are processed at the same priority level. Even unimportant DPCs, which are not very much important even the unimportant deferred procedure calls they can affect the execution of a real-time thread. So, if a real time thread it arrives later and there is a very on important, not so important DPC it is running it can affect the execution of a real-time thread.

Even since, why, because all the DPCs they are processed at the same priority level. So, even if some unimportant DPCs, they are running and a real-time task is coming these unimportant DPCs they can affect the execution of real-time threat. Let us take a small example in order to illustrate this. You have all known, page faults, paging segmentation you have known page faults you have known.

The page faults have a low priority process, it can execute at a higher priority than the real-time process. So, in this either Windows NT or this Windows XP what you can say the page faults of a low priority process which is what you can say it is not so much important, it is unimportant. The page faults of a low priority process it can execute at a higher priority than a real-time process.

We require for us the real-time tasks are very much important. They should be given first priority. Since the unimportant DPCs can affect the execution of real-time threat due to this the following happens. The page faults of a low priority process, it can, they can execute at a higher priority than even than a real-time process.

(Refer Slide Time: 24:35)



DEFERRED PROCEDURE CALLS (DPC)

- Tasks performing network or disk I/O:
 - May lead to unbounded response time for real-time tasks.
- Win CE uses a priority inheritance scheme to avoid priority inversion.
- DPCs operate at priority level NORMAL.
- A DPC of a higher priority process,
 - Raised to the priority of that process.

The slide features a dark blue header with the title in white. The main content is on a light blue background with a list of bullet points. A video inset in the bottom right shows a man speaking. Logos for IIT Bombay and IITEL are at the bottom.

The tasks performing network or disk I/O operations they may lead to unbounded response time for the real-time tasks. So, the tasks, which are performing just network operations or disk I/O operations, they may lead to unbounded response time for this real-time task. So, this is a problem in Windows, so that this will hamper about or this will obstacle, this will obstruct to be used as a real-time operating system.

We will see another variant of Windows that is Windows CE. Somehow it overcomes this problem. It uses a priority inheritance scheme to avoid this type of priority inversion. So, when we will discuss Win CE in some of the next classes we may discuss this feature. The DPCs operate at a priority level normal. I have already told you the different priority levels. Idle, normal, high, and real-time.

So, the DPCs that means, the deferred procedure calls they operate at priority level normal. Now, a DPC have a high priority process. So, how to do it? How to assign or how to rate the priority? So, when a DPC of a high priority process it comes then the priority of that deferred procedure call is raised to the priority of that process.

So, when a DPC have a deferred procedure call of a higher priority process comes then the priority of that deferred procedure call is raised to the priority of that process, that means, that higher priority process.

(Refer Slide Time: 26:08)

MAJOR SHORTCOMINGS OF WINDOWS NT

- **Unpredictable interrupt handling:**
 - Can cause unbounded priority inversion.
- **Weak resource access control:**
 - Does not support even the basic priority inheritance scheme.
- **Poor support for distributed applications.**

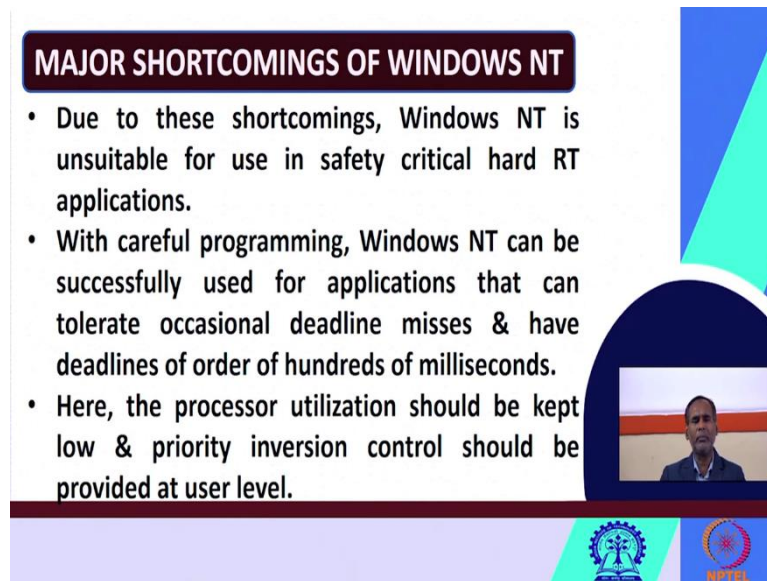
The slide features a video inset of a man speaking in the bottom right corner. At the bottom, there are navigation icons and logos for IIT Bombay and NPTEL.

Now, let us quickly look at the major shortcomings of Windows NT, which obstructs from using or which obstructs from the point that that to be used as a real-time operating system. Here, there is unpredictable interrupt handling. So, the interrupt handling is unpredictable. You cannot predict that whether the interrupts are properly handled, correctly handle or not, there is unpredictable interrupt handling, which may cause unbounded priority inversion and there is no mechanism to handle this unbounded priority inversion.

I hope, you have already known about these problems, unbounded priority inversion etc in this earlier classes. The second drawback is that, weak resource access control. There is no, how to share the resources such as the basic principles, the basic schemes such as PIP, PCP, HLP, etc. So, those things are not supported here. Even it does not support the basic priority inheritance scheme so like PIP.

So, in case of Windows NT, there is a weak resource access control. Even, it does not support even the basic priority inversion schemes such as PIP. The other shortcomings that it has very poor support for distributed applications. So, we know that Windows for desktop applications, it has very poor support for distributed applications. So, these are some of the major shortcomings of Windows NT.

(Refer Slide Time: 27:35)



MAJOR SHORTCOMINGS OF WINDOWS NT

- Due to these shortcomings, Windows NT is unsuitable for use in safety critical hard RT applications.
- With careful programming, Windows NT can be successfully used for applications that can tolerate occasional deadline misses & have deadlines of order of hundreds of milliseconds.
- Here, the processor utilization should be kept low & priority inversion control should be provided at user level.

The slide features a dark blue header with the title in white. The main content is on a white background with a dark blue border. A video inset on the right shows a man speaking. The bottom of the slide has a light blue footer with two logos: a circular one on the left and a red one on the right.

Due to these shortcomings this Windows NT is not suitable for use in safety critical hard real-time applications. However, with careful programming, with proper programming, the Windows NT can be successfully used for applications that can tolerate some occasional deadline misses.

So, when some few deadline misses can be tolerated for those cases you can use, NT can be used as a real-time applications. And the tasks which have deadlines of orders of the order of hundreds of milliseconds in those cases. If it is just 1 millisecond or 2 milliseconds you should not go for Windows NT, but if the task they are having deadlines of the order of some hundreds of milliseconds, then in that case, we can tolerate some of the deadlines and the Windows NT can be successfully used.

So, I am just concluding that due to the above shortcomings Windows NT, is not suitable for use in safety critical hard real-time applications. However, with careful programming, this Windows NT can be successfully used for applications, which can tolerate occasional or few deadline misses. And the tasks that have deadlines of the order of few hundreds of milliseconds, but the processor utilization should be kept low.

So, while we are trying to apply Windows NT or use Windows NT for these applications as a real-time operating system in these cases the processor utilization must be kept low and the priority inversion control must be provided at user level. So, for this, when I am saying, I have, I am saying that it cannot be used as a real-time system for safety critical hard real-time applications. If you will carefully program it may be used for applications. So here few deadline

misses can be tolerated, and the deadlines of the tasks are of the order of the hundreds of milliseconds, but in this case, the processor utilization must be kept very low, and the priority inversion control must be provided at the user level.

So, in this way, in some cases, the Windows NT can be used for real-time applications, but not certainly it cannot be used for the safety critical hard real-time applications only for the application where few deadline misses can be tolerated and the tasks are having deadlines of the order of some hundreds of milliseconds, in those cases, you can use Windows NT as a real time operating system.

(Refer Slide Time: 30:09)

Real-Time Feature	Win NT	Unix V
DPCs	Yes	No
Real-Time Priorities	Yes	No
Timer Precision	1msec	10msec
Aynchronous I/O	Yes	No
Memory locking	Yes	No

Now, let us quickly look at this comparison. We will compare the Windows NT versus the Unix. Already we have discussed Unix in the last class. So, let us the first comparison is, so we will compare with respect to some real-time features. So, first feature is DPC Deferred Procedure Calls. We have seen Unix does not support DPC, whereas, Windows NT supports DPC. Regarding real-time priorities we have seen Windows NT is having real-time priorities but in XP Unix V there is no real-time priorities, the priorities are dynamically changing.

The timer precision in case of Windows NT it is much finer just it is of 1 millisecond, but in Unix it is of the order of 10 millisecond. Regarding the I/O operations, Windows NT, what supports asynchronous I/O operation, but Unix V it does not support asynchronous I/O mainly it supports a synchronous I/O.

Regarding memory locking Windows NT supports the memory locking whereas, Unix V does not support memory locking. So, these are the points how we can or these are the features against which we can compare Windows NT and Unix V.

(Refer Slide Time: 31:19)

CONCLUSION

- Discussed Windows as a Real-Time operating system.
- Presented evolution of Windows operating system.
- Discussed on Windows NT & XP.
- Explained the shortcomings of Windows NT.
- Compared Windows NT vs UNIX.

REFERENCES

1. Rajib Mall, Real-Time Systems: Theory and Practice, 1st Edition, 2007, Pearson Education
2. C. M. Krishna & K. G. Shin, Real-Time Systems, 2017, Tata McGraw Hill Education

So, today we have discussed windows as a real-time operating system. We have seen what is the what suitability that real-time that Windows can be used as a real-time operating system. We have presented the evolution of Windows operating system. We have discussed on two variants of windows that is Windows NT and Windows XP.

We have explained the shortcomings of Windows NT to be used as a real-time operating system. Finally, we have compared Windows NT versus Unix. So, but you can see that most

popularly even if Unix has some of the drawbacks, but it is a little bit cheaper. So nowadays many people have started using Unix as a real-time operating system. We have taken these things from these two please have a look at this or even add internal sources to get a much more idea on the windows and the suitability of Windows real-time operating systems. Thank you very much.