# The PC Boot Process - Windows XP

**Power supply switched on**

The power supply performs a self-test. When all voltages and current levels are acceptable, the supply indicates that the power is stable and sends the Power Good signal to the processor. The time from switch-on to Power Good is usually between .1 and .5 seconds.

**The microprocessor timer chip receives the Power Good signal**

With the arrival of the Power Good signal the timer chip stops sending reset signals to the processor allowing the CPU to begin operations.

**The CPU starts executing the ROM BIOS code**

The CPU loads the ROM BIOS starting at ROM memory address FFFF:0000 which is only 16 bytes from the top of ROM memory. As such it contains only a JMP (jump) instruction that points to the actual address of the ROM BIOS code.

**The ROM BIOS performs a basic test of central hardware to verify basic functionality**

Any errors that occur at this point in the boot process will be reported by means of 'beep-codes' because the video subsystem has not yet been initialized.

**The BIOS searches for adapters that may need to load their own ROM BIOS routines**

Video adapters provide the most common source of adapter ROM BIOS. The start-up BIOS routines scan memory addresses C000:0000 through C780:0000 to find video ROM. An error loading any adapter ROM generates an error such as:

XXXX ROM Error

where XXXX represents the segment address of the failed module.

**The ROM BIOS checks to see if this is a 'cold-start' or a 'warm-start'**

To determine whether this is a warm-start or a cold start the ROM BIOS startup routines check the value of two bytes located at memory location 0000:0472. Any value other than 1234h indicates that this is a cold-start.

**If this is a cold-start the ROM BIOS executes a full POST (Power On Self Test). If this is a warm-start the memory test portion of the POST is switched off**

The POST can be broken down into three components:
The Video Test initializes the video adapter, tests the video card and video memory, and displays configuration information or any errors.
The BIOS Identification displays the BIOS version, manufacturer, and date.
The Memory Test tests the memory chips and displays a running sum of installed memory.

Errors the occur during the POST can be classified as either 'fatal' or 'non-fatal'. A non-fatal error will typically display an error message on screen and allow the system to continue the boot process. A fatal error, on the other hand, stops the process of booting the computer and is generally signaled by a series of beep-codes.

**The BIOS locates and reads the configuration information stored in CMOS**

CMOS (which stands for Complementary Metal-Oxide Semiconductor) is a small area of memory (64 bytes) which is maintained by the current of a small battery attached to the motherboard. Most importantly for the ROM BIOS startup routines CMOS indicates the order in which drives should be examined for an operating systems - floppy first, CD-Rom first, or fixed disk first.

**Fixed Disk**

If the first bootable disk is a fixed disk the BIOS examines the very first sector of the disk for a **Master Boot Record (MBR)**. For a floppy the BIOS looks for a Boot Record in the very first sector.

On a fixed disk the Master Boot Record occupies the very first sector at cylinder 0, head 0, sector 1. It is 512 bytes in size. If this sector is found it is loaded into memory at address 0000:7C00 and tested for a valid signature. A valid signature would be the value 55AAh in the last

| | | | |
|---|---|---|---|
| | | | two bytes. Lacking an MBR or a valid signature the boot process halts with an error message which might read:<br>NO ROM BASIC - SYSTEM HALTED<br>A Master Boot Record is made up of two parts - **the partition table** which describes the layout of the fixed disk and **the partition loader code** which includes instructions for continuing the boot process. |
| | ↓ | | |
| **MBR** | With a valid MBR loaded into memory the BIOS transfers control of the boot process to the partition loader code that takes up most of the 512 bytes of the MBR. | → | The process of installing multiple operating systems on a single PC usually involves replacing the original partition loader code with a Boot Loader program that allows the user to select the specific fixed disk to load in the next step of the process |
| | ↓ | | |
| **Partition Table** | The partition loader (or Boot Loader) examines the partition table for a **partition marked as active**. The partition loader then searches the very first sector of that partition for a **Boot Record**. | → | The Boot Record is also 512 bytes and contains a table that describes the characteristics of the partition (number of bytes per sectors, number of sectors per cluster, etc.) and also the jump code that locates the first of the operating system files (IO.SYS in DOS). |
| **Operating System** | ↓ | | |
| **Boot Record** | The active partition's boot record is checked for a valid boot signature and if found the boot sector code is executed as a program. | → | The loading of Windows XP is controlled by the file NTLDR which is a hidden, system file that resides in the root directory of the system partition. NTLDR will load XP in four stages:<br>1) Initial Boot Loader Phase<br>2) Operating System selection<br>3) Hardware Detection<br>4) Configuration Selection |
| | ↓ | | |
| **NTLDR Initial Phase** | During the initial phase **NTLDR** switches the processor from real-mode to protected mode which places the processor in 32-bit memory mode and turns memory paging on. It then loads the appropriate mini-file system drivers to allow NTLDR to load files from a partition formatted with any of the files systems supported by XP. | → | Windows XP supports partitions formatted with either the FAT-16, FAT-32, or NTFS file system. |

| | | | |
|---|---|---|---|
| | ↓ | | |
| **NTLDR**<br>**OS Selection**<br>**BOOT.INI** | If the file **BOOT.INI** is located in the root directory, NTLDR will read it's contents into memory. If BOOT.INI contains entries for more than one operating system NTLDR will stop the boot sequence at this point, display a menu of choices, and wait for a specified period of time for the user to make a selection. | → | If the file BOOT.INI is not found in the root directory NTLDR will continue the boot sequence and attempt to load XP from the first partition of the first disk, typically C:\. |
| | ↓ | | |
| **F8** | Assuming that the operating system being loaded is Windows NT, 2000, or XP **pressing F8** at this stage of the boot sequence to display various boot options including "**Safe Mode**" and "**Last Known Good Configuration**" | → | After each successful boot sequence XP makes a copy of the current combination of driver and system settings and stores it as the Last Known Good Configuration. This collection of settings can be used to boot the system subsequently if the installation of some new device has caused a boot failure. |
| | ↓ | | |
| **NTLDR**<br>**Hardware**<br>**Detection** | If the selected operating system is XP, NTLDR will continue the boot process by locating and loading the DOS based **NTDETECT.COM** program to perform hardware detection. | → | NTDETECT.COM collects a list of currently installed hardware components and returns this list for later inclusion in the registry under the HKEY_LOCAL_MACHINE\ HARDWARE key. |
| | ↓ | | |
| **NTLDR**<br>**Configuration**<br>**Selection** | If this computer has more than one defined **Hardware Profile** the NTLDR program will stop at this point and display the Hardware Profiles/Configuration Recovery menu. | → | Lacking more than one Hardware Profile NTLDR will skip this step and not display this menu. |
| | ↓ | | |
| **Kernel Load** | After selecting a hardware configuration (if necessary) NTLDR begins loading the XP kernel (**NTOSKRNL.EXE**). | → | During the loading of the kernel (but before it is initialized) NTLDR remains in control of the computer. The screen is cleared and a series of white rectangles progress across the bottom of the screen. NTLDR also loads the **Hardware Abstraction Layer (HAL.DLL)** at this time which will insulate the kernel from hardware. Both files are located in the \system32 directory. |
| | ↓ | | |

| | | | |
|---|---|---|---|
| **NTLDR Boot Device Drivers** | NTLDR now loads **device drivers** that are marked as boot devices. With the loading of these drivers NTLDR relinquishes control of the computer. | → | Every driver has a registry subkey entry under HKEY_LOCAL_MACHINE \SYSTEM\Services. Any driver that has a Start value of SERVICE_BOOT_START is considered a device to start at boot up. A period is printed to the screen for each loaded file (unless the /SOS switch is used in which case file names are printed. |
| | ↓ | | |
| **Kernel Initialization** | NTOSKRNL goes through two phases in its boot process - **phase 0 and phase 1**. Phase 0 initializes just enough of the microkernel and Executive subsystems so that basic **services** required for the completion of initialization become available.. At this point, the system display a graphical screen with a status bar indicating load status.<br>↓ | → | XP disables interrupts during phase 0 and enables them before phase 1. The HAL is called to prepare the interrupt controller; the Memory Manager, Object Manager, Security Reference Monitor, and Process Manager are initialized.<br>Phase 1 begins when the HAL is called to prepare the system to accept interrupts from devices. If more than one processor is present the additional processors are initialized at this point. All Executive subsystems are reinitialized in the following order:<br><br>1) Object Manager<br>2) Executive<br>3) Microkernel<br>4) Security Reference Monitor<br>5) Memory Manager<br>6) Cache Manager<br>7) LPCS<br>8) I/O Manager<br>9) Process Manager |
| | ↓ | | |
| **I/O Manager** | The initialization of I/O Manager begins the process of loading all the **systems driver files**. Picking up where NTLDR left off, it first finishes the loading of boot devices. Next it assembles a prioritized list of drivers and attempts to load each in turn. | → | The failure of a driver to load may prompt NT to reboot and try to start the system using the values stored in the Last Known Good Configuration. |
| | ↓ | | |
| **SMSS** | The last task for phase 1 initialization of the kernel is to launch the **Session Manager Subsystem (SMSS)**. SMSS is responsible for creating the user-mode environment that provides the visible interface to NT. | → | SMSS runs in user-mode but unlike other user-mode applications SMSS is considered a trusted part of the operating system and is also a native application (it uses only core Executive functions). These two features allow SMSS to start the graphics subsystem and login processes. |

| | | | |
|---|---|---|---|
| | ↓ | | |
| **win32k.sys** | SMSS loads the **win32k.sys** device driver which implements the Win32 graphics subsystem. | → | Shortly after win32k.sys starts it switches the screen into graphics mode. The Services Subsystem now starts all services mark as Auto Start. Once all devices and services are started the boot is deemed successful and this configuration is saved as the **Last Known Good Configuration**. |
| | ↓ | | |
| **Logon** | The XP boot process is not considered complete until a user has successfully logged onto the system. The process is begun by the **WINLOGON.EXE** file which is loaded as a service by the kernel and continued by **the Local Security Authority (LSASS.EXE)** which displays the logon dialog box. | → | This dialog box appears at approximately the time that the Services Subsystem starts the network service. |