The BIOS Optimization Guide

rev. 5.7

BIOS Features Setup (Part 1)

Virus Warning / Anti-Virus Protection

Options : Enabled, Disabled, ChipAway

When Virus Warning is enabled, the BIOS will flash a warning message whenever there's an attempt to access the boot sector or the partition table. You should leave this feature enabled if possible. Note that this only protects the boot sector and the partition table, not the entire hard disk.

However, this feature will cause problems with the installation of certain software. One good example is the installation routine of Win95/98. When enabled, this feature will cause Win95/98's installation routine to fail. Disable it before installing such software.

Also, many disk diagnostic utilities that access the boot sector can trigger the error message as well. You should first disable this option before using such utilities.

Finally, this feature is useless for hard disks that run on external controllers with their own BIOS. Boot sector viruses will bypass the system BIOS and write directly to such hard disks. Such controllers include SCSI controllers and UltraDMA 66 controllers.

Some motherboards will have their own rule-based anti-virus code (ChipAway) incorporated into the BIOS. Enabling it will provide additional anti-virus protection for the system as it will be able to detect boot viruses before they have a chance to infect the boot sector of the hard disk. Again, this is useless if the hard disk is on a separate controller with its own BIOS.

CPU Level 1 Cache

Options : Enabled, Disabled

This BIOS setting can be used to enable or disable the CPU's L1 (primary) cache. Naturally, the default setting is Enabled.

This feature is useful for overclockers who want to pinpoint the cause of their unsuccessful overclocking. I.e. if a CPU cannot reach 500MHz with the L1 cache enabled and vice versa; then the L1 cache is what's stopping the CPU from reaching 500MHz stably.

However, disabling the L1 cache in order to increase the overclockability of the CPU is a very bad idea, especially in highly pipelined designs like Intel's P6 family of processors (Pentium Pro, Celeron, Pentium II, Pentium !!!).

CPU Level 2 Cache

Options : Enabled, Disabled

This BIOS setting can be used to enable or disable the CPU's L2 (secondary) cache. Naturally, the default setting is Enabled.

This feature is useful for overclockers who want to pinpoint the cause of their unsuccessful overclocking. I.e. if a CPU cannot reach 500MHz with the L2 cache enabled and vice versa; then the L2 cache is what's stopping the CPU from reaching 500MHz stably.

Users may choose to disable L2 cache in order to overclock higher but the trade-off isn't really worth it.

CPU L2 Cache ECC Checking

Options : Enabled, Disabled

This feature enables or disables the L2 cache's ECC checking function (if available). Enabling this feature is recommended because it will detect and correct single-bit errors in data stored in the L2 cache. It will also detect double-bit errors but not correct them. Still, ECC checking stabilizes the system, especially at overclocked speeds when errors are most likely to creep in.

There are those who advocate disabling ECC checking because it reduces performance. The performance difference is negligible, if at all. However, the stability and reliability achieved via ECC checking is real and substantial. It may even enable you to overclock higher than is possible with ECC checking disabled. So, enable it for added stability and reliability.

Processor Number Feature

Options : Enabled, Disabled

This feature is only valid if you install a Pentium !!! processor. It will most probably not appear unless you have the Pentium !!! processor installed. This feature enables you to control whether the Pentium !!!'s serial number can be read by external programs. Enable this if your secure transactions require you to use such a feature. But for most people, I think you should disable this feature to safeguard your privacy.

Quick Power On Self Test

Options : Enabled, Disabled

When enabled, this feature will shorten some tests and skip others that are performed during the booting up process. Thus, the system boots up much quicker.

Enable it for faster booting but disable it after making any change to the system to detect any errors that may slip through the Quick Power On Self Test. After a few error-free test runs, you can reenable this option for faster booting without impairing system stability.

Boot Sequence

Options : A, C, SCSI/EXT
C, A, SCSI/EXT
C, CD-ROM, A
CD-ROM, C, A
D, A, SCSI/EXT (*only when you have at least 2 IDE hard disks*)
E, A, SCSI/EXT (*only when you have at least 3 IDE hard disks*)
F, A, SCSI (*only when you have 4 IDE hard disks*)
SCSI/EXT, A, C
SCSI/EXT, C, A
A, SCSI/EXT, C
LS/ZIP,C

This feature enables you to set the sequence in which the BIOS will search for an operating system. To ensure the shortest booting time possible, select the hard disk that contains your operating system as the first choice. Normally, that would be drive C but if you are using a SCSI hard disk, then select SCSI.

Special : Some motherboards (i.e. ABIT BE6 and BP6) have an extra onboard IDE controller. The BIOS options in these motherboards replaces the **SCSI** option with an EXT option. This allows the computer to boot from an IDE hard disk on the 3rd or 4th IDE ports (courtesy of the extra onboard IDE controller) or from a SCSI hard disk. If you

want to boot from an IDE hard disk running off the 1st or 2nd IDE ports, do not set the Boot Sequence to start with **EXT**. Note that this function has to work in conjunction with the **Boot Sequence EXT Means** function.

Boot Sequence EXT Means

Options : IDE, SCSI

This function is only valid if the **Boot Sequence** function above has **EXT** settings and this function has to cooperate with the **Boot Sequence** function. This function allows you to set whether the system boots from an IDE hard disk that's connected to any of the ***extra*** two IDE ports found on some motherboards (i.e. ABIT BE6 and BP6) or a SCSI hard disk.

To boot from an IDE hard disk that's connected to the 3rd or 4th IDE port, courtesy of the extra onboard IDE controller), you'll first have to set the **Boot Sequence** (above) function to start with EXT first. For example, the **EXT, C, A** setting. Then, you will have to set this function, **Boot Sequence EXT Means** to **IDE**.

In order to boot from a SCSI hard disk, set the **Boot Sequence** (above) function to start with EXT first. For example, the **EXT, C, A** setting. Then, you will have to set this function, **Boot Sequence EXT Means** to **SCSI**.

BIOS Features Setup (Part 2)

First Boot Device

Options : Floppy, LS/ZIP, HDD-0, SCSI, CDROM, HDD-1, HDD-2, HDD-3, LAN, Disabled

This feature allows you to set the **first** device from which the BIOS will attempt to load the operating system (OS) from. Note that if the BIOS is able to load the OS from the device set using this feature, it naturally won't load another OS, if you have another on a different device.

For example, if you set Floppy as the **First Boot Device**, the BIOS would boot the DOS 3.3 OS which you have placed in the floppy disk but won't bother loading Win2k even though it may be residing on your hard disk drive C. As such, this is useful for troubleshooting purposes and for installing an OS off a CD.

The default setting is **Floppy**. But unless you boot often from the floppy drive or need to install an OS from a CD, it's better to set your hard disk (usually HDD-0) as the First Boot Device. That will shorten the booting process.

Second Boot Device

Options : Floppy, LS/ZIP, HDD-0, SCSI, CDROM, HDD-1, HDD-2, HDD-3, LAN, Disabled

This feature allows you to set the **second** device from which the BIOS will attempt to load the operating system (OS) from. Note that if the BIOS is able to load the OS from the device set as the First_Boot_Device, any setting toggled by this feature will have no effect. Only if the BIOS fails to find an OS on the First_Boot_Device, will it then attempt to find and load one on the **Second Boot Device**.

For example, if you set Floppy as the First Boot Device but left the floppy disk out of the drive, the BIOS will then load Win2k which you have installed on your hard disk drive C (set as Second Boot Device).

The default setting is **HDD-0**, which is first detected hard disk, usually the one attached to the Primary Master IDE channel. Unless you have a removable drive set as the First Boot Device, this feature has very little use. HDD-0 is a perfectly fine choice although you can set an different device to serve as an alternative boot drive.

Third Boot Device

Options : Floppy, LS/ZIP, HDD-0, SCSI, CDROM, HDD-1, HDD-2, HDD-3, LAN, Disabled

This feature allows you to set the **third** device from which the BIOS will attempt to load the operating system (OS) from. Note that if the BIOS is able to load the OS from the device set as the First_Boot_Device or the Second_Boot_Device, any setting toggled by this feature will have no effect. Only if the BIOS fails to find an OS on the First_Boot_Device and Second_Boot_Device, will it then attempt to find and load one on the **Third Boot Device**.

For example, if you set Floppy as the First Boot Device and the LS-120 drive as the Second Boot Device but left both drives empty, the BIOS will then load Win2k which you have installed on your hard disk drive C (set as Third Boot Device).

The default setting is **LS/ZIP.** Unless you have a removable drives set as the First and Second Boot Devices, this feature has very little use. LS/ZIP is a perfectly fine choice although you can set an different device to serve as an alternative boot drive.

Boot Other Device

Options : Enabled, Disabled

This feature determines whether the BIOS will attempt to load an OS from the Second or Third Boot Device if it fails to load one from the First_Boot_Device.

The default is **Enabled** and it's recommended that you leave it as such. Otherwise, if the BIOS cannot find an OS in the First Boot Device, it will then halt the booting process with the error message "No Operating System Found" even though there's an OS on the Second or Third Boot Device.

Swap Floppy Drive

Options : Enabled, Disabled

This feature is useful if you want to swap the logical arrangement of the floppy drives. Instead of opening up the casing to do it physically, you can just set this feature to Enabled. Then, the first drive will be mapped as drive B: and the second drive, mapped as drive A:; which is the opposite of the usual convention.

This feature is also useful if both the floppy drives in your system are of different formats and you want to boot from the second drive. That's because the BIOS will only boot from floppy drive A:.

Boot Up Floppy Seek

Options : Enabled, Disabled

This feature controls whether the BIOS checks for a floppy drive while booting up. If it cannot detect one (either due to improper configuration or physical inavailability), it will flash an error message. It will also detect if the floppy drive has 40 or 80 tracks but since all floppy drives in use today have 80 tracks, this check is redundant. This feature should be set as **Disabled** for a faster booting process.

Boot Up NumLock Status

Options : On, Off

This feature controls the functionality of the Numeric Keyboard at boot up. If set to On, the Numeric Keyboard will function in the numeric mode (for typing out numbers) but if set to Off, it will function in the cursor control mode (for controlling the cursor). The setting of this feature is entirely up to your preference.

BIOS Features Setup (Part 3)

Gate A20 Option

Options : Normal, Fast

This feature determines how Gate A20 is used to address memory above 1MB. When this option is set to **Fast**, the motherboard chipset controls the operation of Gate A20. But when set to **Normal**, a pin in the keyboard controller controls Gate A20. Setting Gate A20 to Fast improves memory access speed and thus, overall system speed, especially with OS/2 and Windows.

This is because OS/2 and Windows enter and leave protected mode via the BIOS a lot so Gate A20 needs to switch often from enabled to disabled and back again. Setting this feature to **Fast** improves memory access performance above 1MB because the chipset is much faster in switching Gate A20 than the keyboard controller. It is recommended that you set it to **Fast** for faster memory accesses.

IDE HDD Block Mode

Options : Enabled, Disabled

The IDE HDD Block Mode feature speeds up hard disk access by transferring data from multiple sectors at once instead of using the old single sector transfer mode. When you enable it, the BIOS will automatically detect if your hard disk supports block transfers and configure the proper block transfer settings for it. Up to **64KB** of data can be transferred per interrupt with IDE HDD Block Mode enabled. Since virtually all hard disks now support block transfers, there is normally no reason why IDE HDD Block Mode should not be enabled.

However, if you are running WinNT, beware. According to Chris Bope, Windows NT does not support IDE HDD Block Mode and enabling IDE HDD Block Mode can cause corrupted data. Ryu Connor confirmed this by sending me a link to a Microsoft article about Enhanced IDE operation under WinNT 4.0. According to this article, IDE HDD Block Mode (and 32-bit Disk Access) had been observed to cause data corruption in some cases. Microsoft recommends that WinNT 4.0 users **disable** IDE HDD Block Mode.

On the other hand, Lord Mike asked someone in the know and he was told that the data corruption issue was taken very seriously at Microsoft and that it had been corrected through Service Pack 2. Although he couldn't get an official statement from Microsoft, it's probably safe enough to enable IDE HDD Block Mode if you are running WinNT, just as long as you upgrade to Service Pack 2.

If you disable IDE HDD Block Mode, only **512 bytes** of data can transferred per interrupt. Needless to say, that degrades performance quite a bit. So, only disable IDE HDD Block Mode if you are running WinNT. Other than that, you should enable it for optimal performance.

For more detailed information on IDE HDD Block Mode, check out our Speed Demonz' guide on IDE Block Mode!

32-bit Disk Access

Options : Enabled, Disabled

32-bit Disk Access is a misnomer because it doesn't really allow 32-bit access to the hard disk. What it actually does is set the IDE controller to combine two 16-bit reads from the hard disk into a single 32-bit double word transfer to the processor. This makes more efficient use of the PCI bus as fewer transactions are needed for the transfer of a particular amount of data.

However, according to a Microsoft article about Enhanced IDE operation under WinNT 4.0, 32-bit disk access can cause data corruption under WinNT in some cases. Microsoft recommends that WinNT 4.0 users disable 32-bit Disk Access.

On the other hand, Lord Mike asked someone in the know and he was told that the data corruption issue was taken very seriously at Microsoft and that it had been corrected through Service Pack 2. Although he couldn't get an official statement from Microsoft, it's probably safe enough to enable IDE HDD Block Mode if you are running WinNT, just as long as you upgrade to Service Pack 2.

If disabled, data transfers from the IDE controller to the processor will then occur only in 16-bits. This degrades performance, of course, so you should enable it if possible. Disable it only if you face the possibility of data corruption.

You can also find more information on the WinNT issue above in our Speed Demonz' guide on IDE Block Mode!

Typematic Rate Setting

Options : Enabled, Disabled

This feature enables you to control the keystroke repeat rate when you depress a key continuously. When enabled, you can manually adjust the settings using the two typematic controls (Typematic Rate and Typematic Rate Delay). If disabled, the BIOS will use the default setting.

Typematic Rate (Chars/Sec)

Options : 6, 8, 10, 12, 15, 20, 24, 30

This is the rate at which the keyboard will repeat the keystroke if you press it continuously. This setting will only work if Typematic Rate Setting is Enabled.

Typematic Rate Delay (Msec)

Options : 250, 500, 750, 1000

This is the delay, in milliseconds, before the keyboard automatically repeats the keystroke that you have pressed continuously. This setting will only work if Typematic Rate Setting is Enabled.

Security Setup

Options : System, Setup

This option will only work once you have created a password through **PASSWORD SETTING** out in the main BIOS screen.

Setting this option to **System** will set the BIOS to ask for the password each time the system boots up.

If you choose **Setup**, then the password is only required for access into the BIOS setup menus. This option is useful for system administrators or computer resellers who just want to keep novice users from messing around with the BIOS. **:)**

BIOS Features Setup (Part 4)

PCI / VGA Palette Snoop

Options : Enabled, Disabled

This option is only useful if you use an MPEG card or an add-on card that makes use of the graphics card's Feature Connector. It corrects incorrect color reproduction by "snooping" into the graphics card's frame buffer memory and modifying (synchronizing) the information delivered from the graphics card's Feature Connector to the MPEG or add-on card. It will also solve the problem of display inversion to a black screen after using the MPEG card.

Assign IRQ For VGA

Options : Enabled, Disabled

Many high-end graphics accelerator cards now require an IRQ to function properly. Disabling this feature with such cards will cause improper operation and/or poor performance. Thus, it's best to make sure you enable this feature if you are having problems with your graphics accelerator card.

However, some low-end cards don't need an IRQ to run normally. Check your graphics card's documentation (manual). If it states that the card does not require an IRQ, then you can disable this feature to release an IRQ for other uses. When in doubt, it's best to leave it enabled unless you really need the IRQ.

MPS Version Control For OS

Options : 1.1, 1.4

This option is only valid for multiprocessor motherboards as it specifies the version of the Multiprocessor Specification (MPS) that the motherboard will use. The MPS is a specification by which PC manufacturers design and build Intel architecture systems with two or more processors.

MPS version 1.4 added extended configuration tables to improve support for multiple PCI bus configurations and improve future expandability. It is also required for a secondary PCI bus to work without the need for a bridge. Newer versions of server operating systems will generally support MPS 1.4 and as such, you should change the BIOS Setup from the default of 1.1 to 1.4 if your operating system supports version 1.4. Leave it as 1.1 only if you are running older server OSes.

Eugene Tan informed me that the setting for WinNT should be 1.4.

OS Select For DRAM > 64MB

Options : OS/2, Non-OS/2

When the system memory is more than 64MB in size, OS/2 differs from other operating systems (OS) in the way it manages the RAM. So, for systems running IBM's OS/2 operating system, select **OS/2** and for systems running other operating systems, select **Non-OS/2**.

HDD S.M.A.R.T. Capability

Options : Enabled, Disabled

This option enables/disables support for the hard disk's S.M.A.R.T. capability. The S.M.A.R.T. (Self Monitoring Analysis And Reporting) technology is supported by all current hard disks and it allows the early prediction and warning of impending hard disk disasters. You should enable it so that S.M.A.R.T. aware utilities can monitor the hard disk's condition. Enabling it also allows the monitoring of the hard disk's condition over a network. There's no performance advantage in disabling it even if you don't intend to use the S.M.A.R.T. technology.

However, there's a possibility that enabling S.M.A.R.T. may cause spontaneous reboots in networked computers. Johnathan P. Dinan mentioned such an experience with S.M.A.R.T. enabled. S.M.A.R.T. may be sending packets of data through the network even though there's nothing monitoring those data packets. This may have caused the spontaneous reboots that he had experienced (Comment #103). So, try disabling **HDD S.M.A.R.T. Capability** if you experience reboots or crashes while you are on a network.

Report No FDD For Win95

Options : Enabled, Disabled

If you are using Windows 95/98 without a floppy disk drive, select **Enabled** to release IRQ6. This is required to pass Windows 95/98's SCT test. You should also disable the Onboard FDC Controller in the **Integrated Peripherals** screen when there's no floppy drive in the system. If you set this feature to **Disabled**, the BIOS will not report the missing floppy drive to Win95/98.

Delay IDE Initial (Sec)

Options : 0, 1, 2, 3, ..., 15

The booting process of new BIOSes is much faster these days. Thus, some IDE devices may not be able to spin up fast enough for the BIOS to detect them during the booting up process. This setting is used to delay the initialization of such IDE devices during the booting up process.

Leave it at **0** if possible for faster system booting. But if one or more of your IDE devices fail to initialize while booting, increase the value of this setting until they all initialize properly.

BIOS Features Setup (Part 5)

Video BIOS Shadowing

Options : Enabled, Disabled

When this feature is enabled, the Video BIOS is copied to the system RAM for quicker access. Shadowing improves the BIOS' performance because the BIOS can now be read by the CPU through the 64-bit DRAM bus as opposed to the 8-bit XT bus. This seems quite attractive since that's at least a 100x increase in transfer rate and the only price is the loss of some system RAM which is used to mirror the ROM contents.

However, modern operating systems bypass the BIOS completely and access the graphics card's hardware directly. So, no BIOS calls are made and no benefit from BIOS shadowing is realized. In light of this, there's no use in wasting RAM just to shadow the Video BIOS when it's not used at all.

Ryu Connor confirmed this by sending me a link to a Microsoft article about Shadowing BIOS under WinNT 4.0. According to this article, shadowing the BIOS (irrespective of what BIOS it is) does not bring about any performance enhancements because it's not used by WinNT. It will only waste memory. Although the article did not say anything about Win9x, it's the same for Win9x as it's based on the same Win32 architecture.

Not only that, some manuals also allude to the possibility of system instability when certain games access the RAM region that has already been used to shadow the Video BIOS. However, this is no longer an issue as the shadowed RAM region has been moved far from the reach of programs.

What could be an issue is if only 32KB of the video BIOS is shadowed. Newer video BIOSes are larger than 32KB in size but if only 32KB is shadowed and the rest is left in their original locations, then stability issues may arise when the BIOS is accessed. So, if you intend to shadow the video BIOS, you'll need to ensure that the entire video BIOS is shadowed. In many cases, only the C000-C7FF region is shadowed by default. To correct that, you'll need to :-

- enable video BIOS shadowing (for the C000-C7FF region) **and**
- enable the shadowing of the remaining portions, i.e. C800-CBFF, until the entire video BIOS is shadowed.

That tip was generously contributed by X.
Finally, most graphics cards now come with Flash ROM (EEPROM) which is much faster than the old ROM and even faster than DRAM. Thus, there's no longer a need for video BIOS shadowing and there may even be a performance advantage in not shadowing! In addition, you shouldn't shadow the video BIOS if your graphics card comes with a Flash ROM because you wouldn't be able to update its contents if shadowing is enabled.
On the other hand, there may still be a use for this feature. Some DOS games still make use of the video BIOS because they don't directly access the graphics processor (although more graphical ones do). So, if you play lots of old DOS games,

you can try enabling Video BIOS Shadowing for better performance. This tip is courtesy of Ivan Warren.
For an excellent overview of video BIOSes and their shadowing, check out William Patrick McNamara's explanation :-

The whole issue is historical in nature. Way back when having a VGA video card was a big thing, graphics cards were pretty dumb and fairly simple as well. They amounted to a chunk of memory that represented the pixels on the screen. To change a pixel, you changed the memory representing it. Things like changing color palettes, screen resolutions, etc were done by writing to a set of registers on the video card. However, everything was done by the processor. Since interfacing with hardware varies with the hardware, talking to the video card depended on the card you had installed. To help solve the problem, the video card included a BIOS chip on it. Quite simply the video BIOS was
an extension to the system BIOS. It was simply a documented set of function calls a programmer could use to interface with the video chipset. So why did BIOS shadowing come about? The memory used to store the BIOS on a video card is usually some sort of EPROM (Electrically Programmable Read Only Memory). A very fast EPROM has an access time of 130-150ns, which is about the same as the memory in an 8086 based computer. Also, the bus width is 8bits. As computers got faster (x386, x486, etc) and games got more graphical, calling the BIOS got to be more of a bottleneck. To help alleviate the problem, the video BIOS was moved to the faster 16bit system memory to speed things up. Actually most graphical DOS games rarely call the BIOS anyway. Most interact with the chipset directly if possible.
A quick summary: In the "old days", video BIOS didn't really have much to do with running the video card. It simply provided a set of function calls to make a developers life easier.
"And now for something completely different....."
New video cards, ones that have accelerated functions, fall into a different category. They actually have a processor built on the card. In the same way that the system BIOS tells you processor how to start your computer, your video BIOS tells you video processor how to display images. The reason, new card have flash ROMs on them, is so that the manufacturers can fix any bugs that exist in the code. Any operating system that uses the accelerated features of a video card, communicates directly with the processor on the card, giving it a set of commands. This is the job of the video driver. The idea is, the driver presents the operating system with a document set of function calls. When on of these calls is made, the driver sends the appropriate command to the video processor. The video processor the carries out the commands as it programming (video BIOS) dictates.
As far as shadowing the video BIOS goes, it doesn't matter. Windows, Linux, or any other OS that uses the accelerated functions never directly communicates with the video BIOS. Good 'ole DOS however still does, and the same functions that existed in the original VGA cards exist in the new 3D cards. Depending on how the video interface on DOS programs is written, they may benefit from having the video BIOS shadowed.
Quick Summary #2: In today's accelerated video cards, the main job of the video BIOS is to provide a program for the video processor (RIVA TNT2, Voodoo3, etc) to run so that it can do its job. Interface between the video card and software is done through a command set provided by the driver

and really has nothing to do with the video BIOS. The original BIOS function are still available to maintain backwards VGA compatibility.
More on this can be found from his e-mail (Comment #91). Check it out for more information.
For a final confirmation on why you should not shadow the video BIOS, check out Steve Hauser's account of his bad experience with video BIOS shadowing :-
A few years back (probably '96 or so) I had a Matrox Millenium card and the BIOS I had at the time defaulted to shadowing enabled for the VGA BIOS... *WELL* the Millenium had a larger than 32KB BIOS. So, when I ran a BIOS flash, the first bit just copied into the shadow in system RAM, while the rest hit the video card itself.
Needless to say, with the first 32KB block missing, the BIOS of the card was completely corrupted and no longer functioned. Already you can see how shadowing *CAN* get you in real trouble with carelessly written flash software (that doesn't check for it first). Now, I can't attest to any speed increases/decreases it may have caused but here's the really pertinent part, what happened with the card after it no longer had a BIOS....
It still worked! (mostly)... ALL 'DOS' video modes were gone - total blank screen. But you can hear the computer beep and then boot normally. Once the Windows GUI (with proper drivers) loaded, it operated 100% normally. All video acceleration modes worked fine... *EXCEPT* anything related to DOS (even a DOS window within Windows itself) was 100% devoid of text. This includes the 'built-in' VGA (640x480x16 colours) safe mode which also didn't work at all (since it doesn't use drivers).
So, apparently you are 100% correct in assuming that modern video cards do not use the 'DOS addressable' BIOS for anything except driverless VGA/EGA/text modes... Now, that's not to say 'BIOS updates' are useless, as the actual BIOS of the card includes far more than the little table DOS can see. It can include micro-code with patches for problems (just like how motherboard BIOS updates can fix certain processor problems).
I've given you at least one case now where enabling BIOS shadowing can cause SERIOUS and permanent harm to the video card itself... After the failed 'shadowed' flash, the card was never again able to render DOS video modes or text; and further BIOS updates would not work since they 'failed to detect current BIOS revision'.

Shadowing Address Ranges (xxxxx-xxxxx Shadow)

Options : Enabled, Disabled
This option allows you to decide if the memory block of an addon card in the address range of xxxxx-xxxxx will be shadowed or not. Leave it as disabled if you don't have an addon card using that memory range. Also, like Video BIOS Shadowing, there's no benefit in enabling this option if you run Win95/98 and have the proper drivers for your addon card.
Ryu Connor confirmed this by sending me a link to a Microsoft article about Shadowing BIOS under WinNT 4.0. According to this article, shadowing the BIOS (irrespective of what BIOS it is) does not bring about any performance enhancements because it's not used by WinNT. It will only waste memory. Although the article did not say anything about Win9x, it's the same for Win9x as it's based on the same Win32 architecture.
In addition, Ivan Warren warns that if you are using an add-on card which is using some CXXX-EFFF area for I/O, then shadowing would probably prevent the card from working because the memory R/W requests might not be passed to the ISA bus.

Chipset Features Setup (Part 1)

SDRAM CAS Latency Time

Options : 2, 3

This controls the time delay (in clock cycles - CLKs) that passes before the SDRAM starts to carry out a read command after receiving it. This also determines the number of CLKs for the completion of the first part of a burst transfer. Thus, the lower the latency, the faster the transaction. However, some SDRAM cannot handle the lower latency and may become unstable and lose data.

So, set the SDRAM CAS Latency Time to **2** for optimal performance if possible but increase it to **3** if your system becomes unstable.

SDRAM Cycle Time Tras/Trc

Options : 5/6, 6/8

This feature toggles the minimum number of clock cycles required for the Tras and the Trc of the SDRAM.

Tras refers to the SDRAM's **Row Active Time**, which is the length of time in which the row is open for data transfers. It is also known as **Minimum RAS Pulse Width**.

Trc, on the other hand, refers to the SDRAM's **Row Cycle Time**, which determines the length of time for the entire row-open, row-refresh cycle to complete.

The default setting is **6/8** which is more stable and slower than **5/6**. However, **5/6** cycles the SDRAM faster but may not leave the row open long enough for data transactions to complete. This is especially true at SDRAM clockspeeds above 100MHz. Therefore, you can try **5/6** for better SDRAM performance but should increase it to **6/8** if your system becomes unstable.

SDRAM RAS-to-CAS Delay

Options : 2, 3

This option allows you to insert a delay between the RAS (**Row Address Strobe**) and CAS (**Column Address Strobe**) signals. This occurs when the SDRAM is written to, read from or refreshed. Naturally, reducing the delay improves the performance of the SDRAM while increasing it reduces performance.

So, reduce the delay from the default value of **3** to **2** for better SDRAM performance. However, if you are facing system stability issues after reducing the delay, reset the value back to **3**.

SDRAM RAS Precharge Time

Options : 2, 3

This option sets the number of cycles required for the RAS to accumulate its charge before the SDRAM refreshes. Reducing the precharge time to **2** improves SDRAM performance but if the precharge time of **2** is insufficient for the installed SDRAM, the SDRAM may not be refreshed properly and it may fail to retain data.

So, for better SDRAM performance, set the **SDRAM RAS Precharge Time** to **2** but increase it to **3** if you face system stability issues after reducing the precharge time.

SDRAM Cycle Length

Options : 2, 3

This feature is similar to SDRAM CAS Latency Time.

It controls the time delay (in clock cycles - CLKs) that passes before the SDRAM starts to carry out a read command after receiving it. This also determines the number of CLKs for the completion of the first part of a burst transfer. Thus, the lower the cycle length, the faster the transaction. However, some SDRAM cannot handle the lower cycle length and may become unstable.

So, set the SDRAM Cycle Length to **2** for optimal performance if possible but increase it to **3** if your system becomes unstable.

SDRAM Leadoff Command

Options : 3, 4

This option allows you to adjust the leadoff time needed before the data stored in the SDRAM can be accessed. In most cases, it is the access time for the first data element in a burst. For optimal performance, set the value to **3** for faster SDRAM access times but increase it to **4** if you are facing system stability issues.

SDRAM Bank Interleave

Options : 2-Bank, 4-Bank, Disabled

This feature enables you to set the interleave mode of the SDRAM interface. Interleaving allows banks of SDRAM to alternate their refresh and access cycles. One bank will undergo its refresh cycle while another is being accessed. This improves performance of the SDRAM by masking the refresh time of each bank. A closer examination of interleaving will reveal that since the refresh cycles of all the SDRAM banks are staggered, this produces a kind of pipelining effect.

If there are 4 banks in the system, the CPU can ideally send one data request to each of the SDRAM banks in consecutive clock cycles. This means in the first clock cycle, the CPU will send an address to Bank 0 and then send the next address to

Bank 1 in the second clock cycle before sending the third and fourth addresses to Banks 2 and 3 in the third and fourth clock cycles respectively. The sequence would be something like this :-

1. CPU sends address #0 to Bank 0
2. CPU sends address #1 to Bank 1 and receives data #0 from Bank 0
3. CPU sends address #2 to Bank 2 and receives data #1 from Bank 1
4. CPU sends address #3 to Bank 3 and receives data #2 from Bank 2
5. CPU receives data #3 from Bank 3

As a result, the data from all four requests will arrive consecutively from the SDRAM without any delay in between. But if interleaving was not enabled, the same 4-address transaction would be roughly like this :-

1. SDRAM refreshes
2. CPU sends address #0 to SDRAM
3. CPU receives data #0 from SDRAM
4. SDRAM refreshes
5. CPU sends address #1 to SDRAM
6. CPU receives data #1 from SDRAM
7. SDRAM refreshes
8. CPU sends address #2 to SDRAM
9. CPU receives data #2 from SDRAM
10. SDRAM refreshes
11. CPU sends address #3 to SDRAM
12. CPU receives data #3 from SDRAM

As you can see, with interleaving, the first bank starts transferring data to the CPU in the same cycle that the second bank receives an address from the CPU. Without interleaving, the CPU would send the address to the SDRAM, receive the data requested and then wait for the SDRAM to refresh before initiating the second data transaction. That wastes a lot of clock cycles. That's why the SDRAM's bandwidth increases with interleaving enabled.
However, bank interleaving only works if the addresses requested consecutively are not in the same bank. If they are, then the data transactions behave as if the banks were not interleaved. The CPU will have to wait till the first data transaction clears and that SDRAM bank refreshes before it can send another address to that bank.
Each SDRAM DIMM consists of either 2 banks or 4 banks. 2-bank SDRAM DIMMs use 16Mbit SDRAM chips and are usually 32MB or less in size. 4-bank SDRAM DIMMs, on the other hand, usually use 64Mbit SDRAM chips though the SDRAM density may be up to 256Mbit per chip. All SDRAM DIMMs of at least 64MB in size or greater are 4-banked in nature.

If you are using a single 2-bank SDRAM DIMM, set this feature to 2-Bank. But if you are using two 2-bank SDRAM DIMMs, you can use the 4-Bank option as well. With 4-bank SDRAM DIMMs, you can use either interleave options.
Naturally, 4-bank interleave is better than 2-bank interleave so if possible, set it to 4-Bank. Use 2-Bank only if you are using a single 2-bank SDRAM DIMM. Note, however, that Award (now part of Phoenix Technologies) recommends that SDRAM bank interleaving be disabled if 16Mbit SDRAM DIMMs are used.

Chipset Features Setup (Part 2)

SDRAM Precharge Control

Options : Enabled, Disabled

This feature determines whether the processor or the SDRAM itself controls the precharging of the SDRAM. If this option is **disabled**, all CPU cycles to the SDRAM will result in an All Banks Precharge Command on the SDRAM interface, which improves stability but reduces performance.

If this feature is enabled, precharging is left to the SDRAM itself. This reduces the number of times the SDRAM is precharged since multiple CPU cycles to the SDRAM can occur before the SDRAM needs to be refreshed. So, enable it for optimal performance unless you are facing system stability issues with this option enabled.

DRAM Data Integrity Mode

Options : ECC, Non-ECC

This BIOS setting is used to configure your RAM's data integrity mode. ECC stands for **Error Checking and Correction** and it should only be used if you are using special 72-bit ECC RAM. This will enable the system to detect and correct single-bit errors. It will also detect double-bit errors though it will not correct them. This provides increased data integrity and system stability at the expense of a little speed.

If you own ECC RAM, enable it (set **ECC**) to benefit from the increased data integrity. After all, you have already spent so much for the expensive ECC RAM so why not use it? **;)** If you are not using ECC RAM, choose **Non-ECC** instead.

Read-Around-Write

Options : Enabled, Disabled

This BIOS feature allows the processor to execute read commands out of order, as if they are independent from the write commands. So, if a read command points to a memory address whose latest write (content) is in the cache (waiting to be copied into memory), the read command will be satisfied by the cache contents instead.

This negates the need for the read command to go all the way to the DRAM and improves the efficiency of the memory subsystem. Therefore, it is recommended that you enable this feature.

System BIOS Cacheable

Options : Enabled, Disabled

This feature is only valid when the system BIOS is shadowed. It enables or disables the caching of the system BIOS ROM at **F0000h-FFFFFh** via the L2 cache. This greatly speeds up accesses to the system BIOS. However, this does **not** translate into better system performance because the OS does not need to access the system BIOS much.

As such, it would be a waste of L2 cache bandwidth to cache the system BIOS instead of data that are more critical to the system's performance. In addition, if any program writes into this memory area, it will result in a system crash. So, it is recommended that you **disable** System BIOS Cacheable for optimal system performance.

Video BIOS Cacheable

Options : Enabled, Disabled

This feature is only valid when the video BIOS is shadowed. It enables or disables the caching of the video BIOS ROM at **C0000h-C7FFFh** via the L2 cache. This greatly speeds up accesses to the video BIOS. However, this does **not** translate into better system performance because the OS bypasses the BIOS using the graphics driver to access the video card's hardware directly.

As such, it would be a waste of L2 cache bandwidth to cache the video BIOS instead of data that are more critical to the system's performance. In addition, if any program writes into this memory area, it will result in a system crash. So, it is recommended that you **disable** Video BIOS Cacheable for optimal system performance.

Video RAM Cacheable

Options : Enabled, Disabled

This feature enables or disables the caching of the video RAM at **A0000h-AFFFFh** via the L2 cache. This is supposed to speed up accesses to the video RAM. However, this does **not** translate into better system performance.

Many graphics cards now have a RAM bandwidth of **5.3GB/s** (128bit x 166MHz DDR) and that number is climbing constantly. Meanwhile, SDRAM's bandwidth is still stuck around **0.8GB/s** (64bit x 100MHz) or at most **1.06GB/s** (64bit x 133MHz) if you are using a PC133 system.

Now, although a Pentium !!! 650 may have a L2 cache bandwidth of about 20.8GB/s (256bit x 650MHz), it makes more sense to cache the really slow system SDRAM instead of the graphics card's RAM.

Also note that caching the video RAM doesn't make much sense even with the Pentium !!!'s high L2 cache bandwidth. This is because the video RAM communicates

with the L2 cache via the AGP bus which has a maximum bandwidth of only **1.06GB/s** using the AGP4X protocol. Actually, that bandwidth is halved in the case of the L2 cache caching the graphics card's RAM because data has to pass in two directions.

In addition, if any program writes into this memory area, it will result in a system crash. So, there's very little benefit in caching the video card's RAM. It would be much better to use the processor's L2 cache to cache the system SDRAM instead. It is recommended that you **disable** Video RAM Cacheable for optimal system performance. For more detailed information, check out the Video RAM Caching guide.

Memory Hole At 15M-16M

Options : Enabled, Disabled

Some special ISA cards require this area of memory for them to work properly. Enabling this function reserves the memory area for the card's use. It will also prevent the system from accessing memory above 16MB.

This means that if you enable this function, your OS can only use up to 15MB of RAM, irrespective of how much RAM your system actually has. So, always disable this function unless your ISA card absolutely requires this memory area to work properly.

Chipset Features Setup (Part 3)

8-bit I/O Recovery Time

Options : NA, 8, 1, 2, 3, 4, 5, 6, 7

The PCI bus is much faster than the ISA bus. So, for ISA cards to work properly with I/O cycles from the PCI bus, the I/O bus recovery mechanism adds additional bus clock cycles between each consecutive PCI-originated I/O cycles to the ISA bus.

By default, the bus recovery mechanism adds a minimum of 3.5 clock cycles between each consecutive 8-bit I/O cycle to the ISA bus. The options above enable you to add **even more** clock cycles between each consecutive 8-bit I/O cycle to the ISA bus. Choosing NA sets the number of delay cycles at the minimum 3.5 clock cycles.

So, set the 8-bit I/O Recovery Time at **NA** if possible for optimal ISA bus performance. Increase the I/O Recovery Time only if you are having problems with your 8-bit ISA cards. Note that this function has no meaning if you are **not** using any ISA cards.

16-bit I/O Recovery Time

Options : NA, 4, 1, 2, 3

The PCI bus is much faster than the ISA bus. So, for ISA cards to work properly with I/O cycles from the PCI bus, the I/O bus recovery mechanism adds additional bus clock cycles between each consecutive PCI-originated I/O cycles to the ISA bus.

By default, the bus recovery mechanism adds a minimum of 3.5 clock cycles between each consecutive 16-bit I/O cycle to the ISA bus. The options above enable you to add **even more** clock cycles between each consecutive 16-bit I/O cycle to the ISA bus. Choosing NA sets the number of delay cycles at the minimum 3.5 clock cycles.

So, set the 16-bit I/O Recovery Time at **NA** if possible for optimal ISA bus performance. Increase the I/O Recovery Time only if you are having problems with your 16-bit ISA cards. Note that this function has no meaning if you are **not** using any ISA cards.

Passive Release

Options : Enabled, Disabled

If Passive Release is enabled, CPU-to-PCI bus accesses are allowed during passive release of the PCI bus. Therefore, the processor can access the PCI bus while the ISA bus is being accessed.

Otherwise, the arbiter only accepts another PCI master access to local DRAM. In other words, only another PCI bus master can access the PCI bus, not the processor. This function is used to meet the latency of the ISA bus master, which is much longer than that of the PCI bus master.

Enable Passive Release for optimal performance. Disable it only if you are facing problems with your ISA cards.

Delayed Transaction

Options : Enabled, Disabled

This feature is used to meet the latency of PCI cycles to and from the ISA bus. The ISA bus is much, much slower than the PCI bus. Thus, PCI cycles to and from the ISA bus take a longer time to complete and this slows the PCI bus down.

However, enabling **Delayed Transaction** enables the chipset's embedded 32-bit posted write buffer to support delayed transaction cycles. This means that transactions to and from the ISA bus are buffered and the PCI bus can be freed to perform other transactions while the ISA transaction is underway.

This option should be **enabled** for better performance and to meet PCI 2.1 specifications. Disable it only if your PCI cards cannot work properly or if you are using an ISA card that is not PCI 2.1 compliant.

PCI 2.1 Compliance

Options : Enabled, Disabled

This is the same thing as **Delayed Transaction** above.

This feature is used to meet the latency of PCI cycles to and from the ISA bus. The ISA bus is much, much slower than the PCI bus. Thus, PCI cycles to and from the ISA bus take a longer time to complete and this slows the PCI bus down.

However, enabling **Delayed Transaction** enables the chipset's embedded 32-bit posted write buffer to support delayed transaction cycles. This means that transactions to and from the ISA bus are buffered and the PCI bus can be freed to perform other transactions while the ISA transaction is underway.

This option should be **enabled** for better performance and to meet PCI 2.1 specifications. Disable it only if your PCI cards cannot work properly or if you are using an ISA card that is not PCI 2.1 compliant.

AGP Aperture Size (MB)

Options : 4, 8, 16, 32, 64, 128, 256

This option selects the size of the AGP aperture. The aperture is a portion of the PCI memory address range dedicated as graphics memory address space. Host cycles that hit the aperture range are forwarded to the AGP without need for translation. This size also determines the maximum amount of system RAM that can be allocated to the graphics card for texture storage.

AGP Aperture size is set by the formula : **maximum usable AGP memory size x 2 plus 12MB**. That means that usable AGP memory size is *less than half* of the AGP aperture size. That's because the system needs AGP memory (uncached) plus an equal amount of write combined memory area and an additional 12MB for virtual addressing. This is address space, not physical memory used. The physical memory is allocated and released as needed only when Direct3D makes a "create non-local surface" call.

Win95 (with VGARTD.VXD) and Win98 use a "waterfall effect". Surfaces are created first in local memory. When that memory is full, surface creation spills over into AGP memory and then system memory. So, memory usage is automatically optimized for each application. AGP and system memory are not used unless absolutely necessary.

The size of the aperture does **not** correspond to performance so increasing it to gargantuan proportions will not improve performance. Many graphics card, however, will require a larger than 8MB AGP aperture size to work properly so you will need to set a minimum of 16MB for the AGP aperture size. Even then, you should set the aperture size at a higher setting so that it will be large enough to accommodate any texture storage requirements that your games/applications may have.

At the moment, the rule of the thumb is an AGP aperture size of about 64MB to 128MB. Increasing the AGP aperture size beyond 128MB wouldn't really hurt performance but it would still be best to keep the aperture size to about 64MB-128MB so that the GART table won't be too large. As the amount of onboard RAM increases and texture compression becomes commonplace, there's less of a need for the AGP aperture size to increase beyond 64MB. So, it's recommended that you set the AGP Aperture Size as **64MB** or at most, **128MB**.

Chipset Features Setup (Part 4)

AGP 2X Mode

Options : Enabled, Disabled

This BIOS feature enables or disables the AGP2X transfer protocol. The standard AGP1X only makes use of the rising edge of the AGP signal for data transfer. At 66MHz, this translates into a bandwidth of 264MB/s. Enabling **AGP 2X Mode** doubles that bandwidth by transferring data on **both** the rising and falling edges of the signal. Therefore, while the clockspeed of the AGP bus still remains as 66MHz, the effective bandwidth of the bus is doubled. This is the same method by which UltraDMA 33 derives its performance boost.

However, both the motherboard chipset and the graphics card must support AGP2X transfers before you can use the AGP2X transfer protocol. If your graphics card support AGP2X transfers, **enable** AGP 2X Mode for a higher AGP transfer rate. Disable it only if you are facing stability issues (especially with Super Socket 7 motherboards) or if you intend to overclock the AGP bus beyond 75MHz and can't just disable sidebanding.

AGP Master 1WS Read

Options : Enabled, Disabled

By default, the AGP busmastering device waits for at least 2 wait states or AGP clock cycles before it starts a read transaction. This BIOS option allows you to reduce the delay to only 1 wait state or clock cycle. For better AGP read performance, **enable** this option but disable it if you experience weird graphical anomalies like wireframe effects and pixel artifacts after enabling this option.

AGP Master 1WS Write

Options : Enabled, Disabled

By default, the AGP busmastering device waits for at least 2 wait states or AGP clock cycles before it starts a write transaction. This BIOS option allows you to reduce the delay to only 1 wait state or clock cycle. For better AGP write performance, **enable** this option but disable it if you experience weird graphical anomalies like wireframe effects and pixel artifacts after enabling this option.

USWC Write Posting

Options : Enabled, Disabled

USWC or **Uncacheable Speculative Write Combination** improves performance for Pentium Pro systems (and possibly other P6 processors as well) with graphic cards that have a linear framebuffer (all new ones do). By combining smaller data writes into 64-bit writes, it reduces the number of transactions required for a particular amount of data to be transferred into the linear framebuffer of the graphics card.

However, it may cause issues like graphic corruption, crashes, booting problems, etc... if the graphics card does not support such a feature.

In addition, tests using FastVid (in the old article - The Phoenix Project) have shown that such a setting can possibly ***decrease*** performance, instead of increasing it! This was observed with the Intel 440BX-based motherboard.

So, if you are using a Pentium Pro processor or a motherboard based on older chipsets, enable it for faster graphics performance. If you own a newer motherboard, you can try enabling it but make sure you run some tests to determine if this feature really improves performance or not. It's quite possible that it may not anything at all or even decrease performance.

Spread Spectrum

Options : Enabled, Disabled, 0.25%, 0.5%, Smart Clock

When the motherboard's clock generator pulses, the extreme values (spikes) of the pulses creates EMI (Electromagnetic Interference). The **Spead Spectrum** function reduces the EMI generated by modulating the pulses so that the spikes of the pulses are reduced to flatter curves. It does so by varying the frequency so that it doesn't use any particular frequency for more than a moment. This reduces interference problems with other electronics in the area.

However, while enabling Spread Spectrum decreases EMI, system stability and performance may be slightly compromised. This may be especially true with timing-critical devices like clock-sensitive SCSI devices.

Some BIOSes offer a Smart Clock option. Instead of modulating the frequency of the pulses over time, Smart Clock turns off the AGP, PCI and SDRAM clock signals when not in use. Thus, EMI can be reduced without compromising system stability. As a bonus, using Smart Clock can also help reduce power consumption.

If you do not have any EMI problem, leave the setting at **Disabled** for optimal system stability and performance. But if you are plagued by EMI, use the **Smart Clock** setting if possible and settle for **Enabled** or one of the two other values if Smart Clock is not available. The percentage values denote the amount of jitter (variation) that the BIOS performs on the clock frequency. So, a lower value (0.25%) is comparatively better for system stability while a higher value (0.5%) is better for EMI reduction.

Auto Detect DIMM/PCI Clk

Options : Enabled, Disabled

This function is similar to the **Smart Clock** option of the Spread Spectrum function. The BIOS monitors the AGP, PCI and SDRAM's activity. If there are no cards in those slots, the BIOS turns off the appropriate AGP, PCI or SDRAM clock signals. And when there's no activity in occupied AGP / PCI / SDRAM slots, the BIOS turns off those clock signals as well.

This way, EMI (Electromagnetic Interference) can be reduced without compromising system stability. This also allows the computer to reduce power consumption because only components that are running will use power.

Still, if you do not have any EMI problem, leave the setting at **Disabled** for optimal system stability and performance. Enable it only if you are plagued by EMI or if you want to save more power.

Chipset Features Setup (Part 5)

Flash BIOS Protection

Options : Enabled, Disabled

This function protects the BIOS from accidental corruption by unauthorized users or computer viruses. When enabled, the BIOS' data cannot be changed when attempting to update the BIOS with a Flash utility. To successfully update the BIOS, you'll need to disable this Flash BIOS Protection function.

You should **enable** this function at all times. The only time when you need to disable it is when you want to update the BIOS. After updating the BIOS, you should immediately re-enable it to protect it against viruses.

Hardware Reset Protect

Options : Enabled, Disabled

This function is useful for file servers and routers, etc., which need to be running 24 hours a day. When enabled, the system's hardware reset button will not function. This prevents the possibility of any accidental resets. When set as **Disabled**, the reset button will function as normal.

It is recommended that you leave it as **Disabled** unless you are running a server and you have kids who just love to press that little red button running around. **;)**

DRAM Read Latch Delay

Options : Enabled, Disabled

This is a BIOS function that introduces a small delay before the system reads data from a DRAM module. This feature was added to facilitate the use of some special SDRAM modules that have unusual timings. You need not enable this feature unless you experience strange system crashes that you suspect is due to memory instability.

So, it's recommended that you leave it as **Disabled** unless you are experiencing some system stability issues. In that case, you can **enable** this BIOS function to see if your DRAM module is one of those with unusual timings and to correct that problem.

DRAM Interleave Time

Options : 0ms, 0.5ms

This BIOS function controls the timing for reading the next bank of data when DRAM Interleave or SDRAM Bank Interleave is enabled. Naturally, the lower the time you use, the faster the DRAM modules can interleave and consequently, the better the performance.

So, it is recommended that you set the time as low as possible for better DRAM performance. Increase the DRAM interleave time only if you face system stability problems.

Byte Merge

Options : Enabled, Disabled

Byte merging holds 8-bit or 16-bit writes from the CPU to the PCI bus in a buffer where it is accumulated and merged into 32-bit writes. The chipset then writes the data in the buffer to the PCI bus when it can. As you can see, merging 8-bit or 16-bit writes reduces the number of PCI transactions, thus freeing up both bandwidth and CPU time.

So, it's recommended that you **enable** this feature for better PCI performance.

PCI Pipeline / PCI Pipelining

Options : Enabled, Disabled

This BIOS function combines PCI or CPU pipelining with byte merging. Byte merging is then used to enhance performance of the graphics card. This function controls the byte-merge feature for framebuffer cycles. When **Enabled**, the controller checks the eight **CPU Byte Enable** signals to determine if data bytes read from the PCI bus by the CPU can be merged.

So, it's recommended that you **enable** this feature for better performance with your PCI graphics card. Other PCI devices may benefit from this feature as well.

Fast R-W Turn Around

Options : Enabled, Disabled

This BIOS option reduces the delay that occurs when the CPU first reads from the RAM and then writes to it. There is normally an extra delay associated with this switch from reading to writing.

If you enable this option, the delay will be reduced and switching from read to write will be faster. However, if your RAM modules cannot handle the faster turnaround, data may be lost and your system may become unstable. With that in mind, **enable**

this option for better RAM performance unless you face stability problems after enabling it.

CPU to PCI Write Buffer

Options : Enabled, Disabled

This controls the CPU write buffer to the PCI bus. If this buffer is disabled, the CPU writes directly to the PCI bus. Although this may seem like the faster and thus, the better method, this isn't true. Because the CPU bus is faster than the PCI bus, any CPU writes to the PCI bus has to wait until the PCI bus is ready to receive data. This prevents the CPU from doing anything else until it has completed sending the data to the PCI bus.

Enabling the buffer enables the CPU to immediately write up to 4 words of data to the buffer so that it can continue on another task without waiting for those 4 words of data to reach the PCI bus. The data in the write buffer will be written to the PCI bus when the next PCI bus read cycle starts. The difference here is that it does so without stalling the CPU for the entire CPU to PCI transaction.

Therefore, it's recommended that you **enable** the CPU to PCI write buffer.

Chipset Features Setup (Part 6)

PCI Dynamic Bursting

Options : Enabled, Disabled

This BIOS option controls the PCI write buffer. If this is enabled, then every write transaction on the PCI bus goes straight to the write buffer. Burst transactions are then sent on their way as soon as there are enough to send in a single burst.

If this option is disabled, the data will go to the write buffer and burst-transferred later (when the PCI bus is free or when the buffer is full) if the write transaction is a burst transaction. If the write transaction is not a burst transaction, then the write buffer is flushed and the data is written to the PCI bus immediately.

It is recommended that you enable **PCI Dynamic Bursting** for better PCI performance.

PCI Master 0 WS Write

Options : Enabled, Disabled

This function determines whether there's a delay before any writes to the PCI bus. If this is enabled, then writes to the PCI bus are executed immediately (with zero wait states), as soon as the PCI bus is ready to receive data. But if it is disabled, then every write transaction to the PCI bus is delayed by **one** wait state.

Normally, it's recommended that you **enable** this for faster PCI performance. However, disabling it may be useful when overclocking the PCI bus results in instability. The delay will generally improve the overclockability of the PCI bus.

PCI Delay Transaction

Options : Enabled, Disabled

This feature is similar to the Delayed Transaction BIOS option. It is used to meet the latency of PCI cycles to and from the ISA bus. The ISA bus is much, much slower than the PCI bus. Thus, PCI cycles to and from the ISA bus take a longer time to complete and this slows the PCI bus down.

However, enabling **Delayed Transaction** enables the chipset's embedded 32-bit posted write buffer to support delayed transaction cycles. This means that transactions to and from the ISA bus are buffered and the PCI bus can be freed to perform other transactions while the ISA transaction is underway.

This option should be **enabled** for better performance and to meet PCI 2.1 specifications. Disable it only if your PCI cards cannot work properly or if you are using an ISA card that is not PCI 2.1 compliant.

PCI#2 Access #1 Retry

Options : Enabled, Disabled

This BIOS feature is linked to the CPU to PCI Write Buffer. Normally, the CPU to PCI Write Buffer is enabled. All writes to the PCI bus are, as such, immediately written into the buffer, instead of the PCI bus. This frees up the CPU from waiting till the PCI bus is free. The data are then written to the PCI bus when the next PCI bus cycle starts.

There's a possibility that the buffer write to the PCI bus may fail. When that happens, this BIOS option determines if the buffer write should be reattempted or sent back for arbitration. If this BIOS option is enabled, then the buffer will attempt to write to the PCI bus until successful. If disabled, the buffer will flush its contents and register the transaction as failed. The CPU will have to write again to the write buffer.

It is recommended that you **enable** this feature unless you have many slow PCI devices in your system. In that case, **disabling** this feature will prevent the generation of too many retries which may severely tax the PCI bus.

Master Priority Rotation

Options : 1 PCI, 2 PCI, 3 PCI

This feature controls the CPU's access to the PCI bus.

If you choose **1 PCI**, the CPU will always be granted access right after the current PCI bus master transaction completes, irrespective of how many other PCI bus masters are on the queue. This affords the quickest CPU access to the PCI bus but means poorer performance for the PCI bus devices.

If you choose **2 PCI**, the CPU will be granted access after the current and the next PCI transaction completes. In other words, the CPU is guaranteed access after two PCI bus master transactions, irrespective of how many other PCI bus masters are also on the queue. This means the CPU has to wait a little longer than with the **1 PCI** option but PCI devices will have quicker access to the PCI bus.

If you choose **3 PCI**, the CPU will only be granted access to the PCI bus after the current PCI bus master transaction and the following two PCI bus master transactions on the queue have been completed. So, the CPU has to wait for three PCI bus masters to complete their transactions on the PCI bus before it can gain access to the PCI bus itself. This means poorer CPU-to-PCI performance but PCI bus master devices will enjoy better performance.

But irrespective of your choice, the CPU is guaranteed access to the PCI bus after a maximum of 3 PCI master grants. It doesn't matter if there are numerous PCI bus

masters on the queue or when the CPU requested access to the PCI bus. It will always be granted access after one PCI bus master transaction (**1 PCI**), two transactions (**2 PCI**) or three transactions (**3 PCI**).

AGP 4X Mode

Options : Enabled, Disabled

This feature is only found on motherboards that support AGP4X. However, it's usually set to **Disabled** by default because not everyone will be using an AGP4X card with the motherboard. For users of AGP1X or 2X cards, this BIOS option needs to be disabled for the cards to work properly. In order to prevent complications, manufacturers prefer to just disable AGP4X mode.

However, this means users of AGP4X cards will lose out on the greater bandwidth afforded by the AGP4X mode. While AGP4X mode's actual transfer rate isn't significantly higher than that of AGP2X, it's still a waste not to use the mode when it's available.

So, if you own an AGP4X card, it's recommended that you **enable** AGP4X mode for better AGP performance. Leave it as disabled only if you have a graphics card that can only support AGP1X or AGP2X transfer modes.

Chipset Features Setup (Part 7)

AGP Driving Control

Options : Auto, Manual

This BIOS function allows you to adjust the control of the AGP driving force. It is usually set to **Auto** by default, thereby allowing the chipset to assume control and automatically adjust the AGP driving force to suit the installed AGP card.

However, for troubleshooting or overclocking purposes, you can set the AGP Driving Control to manual so that you can select the **AGP Driving Value** you want.

AGP Driving Value

Options : 00 to FF (Hex numbers)

This option is slaved to the **AGP Driving Control** BIOS function. If you set the AGP Driving Control to **Auto**, then the value you set here won't have any effect. In order for this BIOS option to work, you need to set the AGP Driving Control to Manual.

The **AGP Driving Value** determines the signal strength of the AGP bus. The higher the value, the stronger the signal. The range of Hex values (00 to FF) translates into 0 to 255 in decimal values. By default, the AGP Driving Value is set to **DA** (*218*) but if you are using an AGP card based on the NVIDIA GeForce2 line of GPUs, then it's recommended that you set the AGP Driving Value to the higher value of **EA** (*234*).

Due to the nature of this BIOS option, it's possible to use it as an aid in overclocking the AGP bus. The AGP bus is sensitive to overclocking, especially in AGP4X mode and with sidebanding enabled. As such, a higher AGP Driving Value may be just what you need to overclock the AGP higher than normally possible. By raising the signal strength of the AGP bus, you can improve its stability at overclocked speeds.

But be *very, very* circumspect when you increase the AGP Driving Value on an overclocked AGP bus as your AGP card may be **irreversibly damaged** in the process!

BTW, contrary to some reports, increasing the AGP Driving Value **won't** improve the performance of the AGP bus. It is **not** a performance enhancing option so you shouldn't increase the value unless you need to.

Integrated Peripherals (Part 1)

Onboard IDE-1 Controller

Options : Enabled, Disabled

This option enables you to activate/inactivate the first IDE channel of the motherboard's onboard IDE controller. You should leave this enabled if you are using this onboard IDE channel. Disabling it will prevent the IDE devices attached to this channel from functioning at all.

If you are not attaching any IDE devices to this port (or if you are using a SCSI / external IDE card instead), you can disable this IDE channel to free an IRQ for other use.

Onboard IDE-2 Controller

Options : Enabled, Disabled

This option enables you to activate/inactivate the second IDE channel of the motherboard's onboard IDE controller. You should leave this enabled if you are using this onboard IDE channel. Disabling it will prevent the IDE devices attached to this channel from functioning at all.

If you are not attaching any IDE devices to this port (or if you are using a SCSI / external IDE card instead), you can disable this IDE channel to free an IRQ for other use.

Master/Slave Drive PIO Mode

Options : 0, 1, 2, 3, 4, Auto

This feature is usually found under the **Onboard IDE-1 Controller** or **Onboard IDE-2 Controller** option. It's linked to one of the IDE channels so if you disable one, the corresponding Master/Slave Drive PIO Mode option for that IDE channel either disappears or is grayed out.

This feature allows you to set the PIO (Programmed Input/Output) mode for the two IDE devices (Master and Slave drives) attached to that particular IDE channel. Normally, you should leave it as **Auto** and let the BIOS auto-detect the IDE drive's PIO mode. You should only set it manually for the following reasons :-

- if the BIOS cannot detect the correct PIO mode
- if you want to try to run the IDE device with a higher PIO mode than it was designed for

- if you have overclocked the PCI bus and one or more of your IDE devices cannot function properly (you can correct the problem by using a slower PIO mode)

Note that overclocking the PIO transfer rate can cause **loss of data**.
Below is a table of the different PIO transfer rates and their corresponding maximum throughputs.

PIO Data Transfer Mode	Maximum Throughput (MB/s)
PIO Mode 0	3.3
PIO Mode 1	5.2
PIO Mode 2	8.3
PIO Mode 3	11.1
PIO Mode 4	16.6

Master/Slave Drive UltraDMA

Options : Auto, Disabled
This feature is usually found under the **Onboard IDE-1 Controller** or **Onboard IDE-2 Controller** option. It's linked to one of the IDE channels so if you disable one, the corresponding Master/Slave Drive Ultra DMA option for that IDE channel either disappears or is greyed out.
This feature allows you to enable or disable UltraDMA support (if available) for the two IDE devices (Master and Slave drives) attached to that particular IDE channel. Normally, you should leave it as **Auto** and let the BIOS auto-detect if the drive supports UltraDMA. If it does, the proper UltraDMA transfer mode will be enabled for that drive, allowing it to burst data at up to 100MB/s. You should only disable it for troubleshooting purposes.
Note that setting this to **Auto** does not enable UltraDMA or any of the slower DMA mode for IDE devices that do not support UltraDMA. Also, in order for any of those DMA modes to work (including the UltraDMA modes), you will have to enable DMA transfer via the OS. In Win9x, that can be done by ticking the **DMA** checkbox in the properties sheet of that IDE drive.
Below is a table of the different DMA transfer rates and their corresponding maximum throughputs.

DMA Transfer Mode	Maximum Throughput (MB/s)
DMA Mode 0	4.16
DMA Mode 1	13.3
DMA Mode 2	16.6
UltraDMA 33	33.3
UltraDMA 66	66.7
UltraDMA 100	100.0

Ultra DMA-66/100 IDE Controller

Options : Enabled, Disabled
This option allows you to enable or disable the **extra** onboard UltraDMA 66/100 controller (if available). This does **not** include the built-in IDE controller of the Intel ICH1 and ICH2 or VIA chipsets which already support UltraDMA 66/100. This function is only for the **extra** IDE controller (from HighPoint or Promise) that has been included onboard the motherboard, in addition to the built-in IDE controller of the chipset.

If you have one or more IDE devices attached to this UltraDMA 66/100 controller, you should enable this function in order to be able to use those IDE devices. You should only disable it for the following reasons :-

- if you don't have any IDE device attached to the additional UltraDMA 66/100 controller
- your motherboard doesn't have an extra UltraDMA 66/100 controller onboard
- for troubleshooting purposes

Note that disabling this function may cut down booting time. This is because the IDE controller's BIOS won't be loaded and thus there won't be a need to wait for it to query for IDE devices on its IDE channels. So, if you don't use it, it might be best to disable it.

USB Controller

Options : Enabled, Disabled

This function is similar to **Assign IRQ For USB**. It enables or disables IRQ allocation for the USB (Universal Serial Bus). Enable this if you are using a USB device. If you disable this while using a USB device, you may have problems running that device. However, if you don't use any USB devices, set the option to **Disabled**. It will free up an IRQ for other devices to use.

Integrated Peripherals (Part 2)

USB Keyboard Support

Options : Enabled, Disabled

This function enables or disables support for a USB keyboard. Enable it if you are using a USB keyboard. Otherwise, disable it.

USB Keyboard Support Via

Options : OS, BIOS

This option determines whether the USB keyboard is supported via the operating system or the BIOS. Support via OS offers better functionality but at the expense of zero functionality in DOS. So, if you use real mode DOS, set the option to BIOS so that you can use the USB keyboard in DOS without the need to install a driver.

Init Display First

Options : AGP, PCI

If you are using more than one graphics card, this function enables you to select whether to use the AGP graphics card or the PCI graphics card as the primary graphics card. This is useful for users who install more than one graphics card but only use a single monitor. This will enable them to select whether to boot the system using the AGP graphics card or the PCI graphics card.

If you are only using one graphics card, then the BIOS will detect it as such and boot it up as normal, irrespective of what you set the option as. However, there may be a slight reduction in initialization time if you set this function to its proper setting. That means if you only use an AGP graphics card, then setting **Init Display First** to **AGP** may reduce boot-up time a little.

KBC Input Clock Select

Options : 8MHz, 12MHz, 16MHz

This function allows you to adjust the keyboard clock for better response or to fix a keyboard problem. You should set it to **16MHz** for a better response time. But if the keyboard becomes erratic or fails to initialize, try a lower clockspeed to fix that.

Power On Function

Options : Button Only, Keyboard 98, Hot Key, Mouse Left, Mouse Right

This function allows you to set the method by which your system can be turned on. Normally, it should be set as **Button Only** so that your system will only start up if you use the button/switch on the casing. Other alternative options including starting up the system using the keyboard (if it supports the Keyboard 98 standard), a keyboard hot key (for other standard keyboards) or the mouse.

Note that only PS/2 mice support this function and then, not all of them. Some PS/2 mice cannot support this function due to some compatibility problem. Mice using the COM port and the USB connection will also not work with this function.

The Keyboard 98 option will only work if you have installed Windows 98 and you have the appropriate keyboard. Then you can use the keyboard's wake-up key to start up the system.

Older keyboards that don't have the special wake-up key can use the **Hot Key** option instead. There are twelve hot keys available : Ctrl-F1 to Ctrl-F12. Select the hot key you want and you will be able to start up the computer using that hot key. However, if your keyboard is too old, this function may not work.

There is no performance advantage in choosing any one of the options above so choose one that you are comfortable with.

Onboard FDD Controller

Options : Enabled, Disabled

This function allows you to enable or disable the onboard floppy drive controller. If you are using a floppy drive connected to the onboard controller, then leave it at the default setting of **Enabled**. But if you are using an add-on FDD controller or if you are not using any floppy drive at all, set it to **Disabled** to save an IRQ.

Onboard Serial Port 1/2

Options : Disabled, 3F8h/IRQ4, 2F8h/IRQ3, 3E8h/IRQ4, 2E8h/IRQ3, 3F8h/IRQ10, 2F8h/IRQ11, 3E8h/IRQ10, 2E8h/IRQ11, Auto

This feature allows you to disable the onboard serial port or to manually select the I/O address and IRQ for it. Normally, you should leave it as **Auto** so that the BIOS can select the best settings for it but if you need a particular IRQ that's been taken up by this serial port, you can manually select an alternative IRQ for it. If you are not using this serial port, you can also disable it to save an IRQ.

Onboard IR Function

Options : IrDA (HPSIR) mode, ASK IR (Amplitude Shift Keyed IR) mode, Disabled

This feature is usually found under the Onboard Serial Port 2 option. It's linked to the 2nd serial port so if you disable the 2nd serial port, this feature will disappear from the screen or appear grayed out.

There are two different IR (Infra-Red) modes. Choose the one appropriate for the connection to the external device. Note that this feature requires an IR connector to be plugged into the IR header provided on the motherboard.

Duplex Select

Options : Full-Duplex, Half-Duplex

This feature is usually found under the Onboard Serial Port 2 option. It's linked to the 2nd serial port so if you disable the 2nd serial port, this feature will disappear from the screen or appear grayed out.

This feature allows you to determine the transmission mode of the IR port. Selecting **Full-Duplex** will permit simultaneous two-way transmission, like a conversation over the phone. However, selecting **Half-Duplex** permits transmission in one direction at any one time only. Thus, the **Full-Duplex** mode is faster and much more desirable. However, consult your IR peripheral's manual to determine if **Full-Duplex** is supported or not.

RxD, TxD Active

Options : High, Low

This feature is usually found under the Onboard Serial Port 2 option. It's linked to the 2nd serial port so if you disable the 2nd serial port, this feature will disappear from the screen or appear grayed out.

This feature enables you to set the IR reception/transmission polarity as **High** or **Low**. You'll need to consult your IR peripheral's documentation to determine the correct polarity.

Integrated Peripherals (Part 3)

Onboard Parallel Port

Options : 3BCh/IRQ7, 278h/IRQ5, 378h/IRQ7, Disabled

This function allows you to select the I/O address and IRQ for the onboard parallel port. The default I/O address of 378h and IRQ of 7 should work well in most cases so unless you have a problem, you should just leave it at the default settings. Only select an alternative I/O address or IRQ if you are facing configuration problems with the parallel port.

Parallel Port Mode

Options : ECP, EPP, ECP+EPP, Normal (SPP)

This feature is usually found under the Onboard Parallel Port option. It's linked to the parallel port so if you disable the parallel port, this feature will disappear from the screen or appear greyed out.

There are four options. The default value is **Normal (SPP)** which will work with all parallel port devices but is very slow. Two faster bidirectional modes are available, namely the **ECP** (Enhanced Com Port) and **EPP** (Enhanced Parallel Port) modes. **ECP** uses the DMA protocol to achieve data transfer rates of up to 2.5Mbits/s and provides symmetric bidirectional communication. On the other hand, **EPP** uses existing parallel port signals to provide asymmetric bidirectional communication.

Generally, because of its FIFOs and the DMA channel it uses, **ECP** is good for large data transfers (useful for scanners and printers). On the other hand, **EPP** is better with links that switch directions frequently (like parallel port drives). This tip was obtained from Jan Axelson's Parallel Port FAQ so check it out if you require more information on parallel ports. However, the manufacturer of your parallel port peripheral may have designated a preferred parallel port mode. In that case, it's best to follow their recommendations.

For those who don't know what mode to select but at least know that their parallel port device supports bi-directional transfers, the BIOS offers the **ECP+EPP** mode. If you select this mode, then the parallel port device will be able to use either one of those modes. However, this should be considered as a last resort as you may be needlessly tying up an IRQ for nothing (if your device does not use ECP at all) or your BIOS may not select the best parallel port mode for the device. If possible, set the parallel port to the transfer mode that best suits your parallel port device.

ECP Mode Use DMA

Options : Channel 1, Channel 3

This feature is usually found under the Parallel Port Mode option. It's linked to that option so if you did not enable either the **ECP** or **ECP+EPP** mode, this feature will disappear from the screen or appear grayed out.

You can use this feature to select the DMA channel of your preference. Normally, the default value of DMA Channel 3 will work just fine. You should only select the alternative value of Channel 1 if there's a conflict with another device.

EPP Mode Select

Options : EPP 1.7, EPP 1.9

This feature is usually found under the Parallel Port Mode option. It's linked to that option so if you did not enable either the **EPP** or **ECP+EPP** mode, this feature will disappear from the screen or appear grayed out.

You can use this feature to choose which version of EPP to use. I couldn't find anything on the difference between version 1.7 and 1.9 but an educated guess would be that version 1.9 would be faster/better than version 1.7. So, for want of better information, I can only recommend that you select **EPP 1.9** if possible but step down to **EPP 1.7** if your device appears to be having some problem with the connection.

PNP/PCI Configuration

PNP OS Installed

Options : Yes, No

If all your operating systems support Plug & Play (PnP), select **Yes** so that they can take over the management of device resources. If you are using a non-PnP-aware OS or not all of the operating systems you are using support PnP, select **No** to let the BIOS handle it instead.

Note that Windows 2000 **will** work with ACPI even with **PnP OS Installed** set to **Enabled**. Just make sure you disable **Advanced Power Management** (**APM**). This information was contributed by Alex. For more information, check out his e-mail. Alex also provided a link to a Microsoft article on how to setup ACPI support for Windows 98 users (*Unfortunately, this link is broken. If you find this article, do let me know so that I can mirror it.*). However, Microsoft recommends that you disable PnP OS Installed, just to be safe. Here's the link to the Microsoft article on IRQ sharing in Windows 2000 provided by Ryu Connor.

For Linux users, Jonathan has the following advice -

Although Linux is not really PnP compatible, most distributions use a piece of software called ISAPNPTOOLS to setup ISA cards. If you have PnP OS set to No, the BIOS will attempt to configure ISA cards itself. This does not make them work with Linux, though, you still need to use something like ISAPNPTOOLS. However, having both the BIOS and ISAPNPTOOLS attempting to configure ISA cards can lead to problems where the two don't agree.

The solution? Set PnP OS to Yes, and let ISAPNPTOOLS take care of ISA cards in Linux, as BIOS configuration of ISA cards doesn't work for Linux anyway (with the current stable and development kernels). Most times, it probably won't make a difference, but someone somewhere will have problems, and Linux will always work with PnP OS set to Yes.

Please refer to Comments #80 and #82 for more information on Linux and PnP.

Force Update ESCD / Reset Configuration Data

Options : Enabled, Disabled

ESCD (Extended System Configuration Data) is a feature of the Plug & Play BIOS that stores the IRQ, DMA, I/O and memory configurations of all the ISA, PCI and AGP cards in the system (PnP or otherwise). Normally, you should leave the setting as **Disabled**.

But if you have installed a new add-on card and the consequent system reconfiguration causes a serious conflict of resources (the OS may not boot as a result), then you should enable it so that the BIOS will reset and reconfigure the

settings for all PnP cards in the system during bootup. The BIOS will automatically reset the setting to **Disabled** the next time you boot.

Resource Controlled By

Options : Auto, Manual

The BIOS has the capability to automatically configure all of the boot and Plug & Play compatible devices. Normally, you should set it as **Auto**, so that the BIOS can automatically assign the IRQs and DMA channels. All the IRQ and DMA assignment fields should disappear as a result.

But if you are facing problems assigning the resources automatically via the BIOS, you can select **Manual** to reveal the IRQ and DMA assignment fields. Then you can assign each IRQ or DMA channel to either *Legacy ISA* or *PCI/ISA PnP* devices.

Legacy ISA devices are compliant with the original PC AT bus specification and require a specific interrupt / DMA channel to function properly. *PCI/ISA PnP* devices, on the other hand, adhere to the Plug & Play standard and can use any interrupt / DMA channel.

Assign IRQ For VGA

Options : Enabled, Disabled

Many high-end graphics accelerator cards now require an IRQ to function properly. Disabling this feature with such cards will cause improper operation and/or poor performance. Thus, it's best to make sure you enable this feature if you are having problems with your graphics accelerator card.

However, some low-end cards don't need an IRQ to run normally. Check your graphics card's documentation (manual). If it states that the card does not require an IRQ, then you can disable this feature to release an IRQ for other uses. When in doubt, it's best to leave it enabled unless you really need the IRQ.

Assign IRQ For USB

Options : Enabled, Disabled

This function is similar to **USB Controller**. It enables or disables IRQ allocation for the USB (Universal Serial Bus). Enable this if you are using a USB device. If you disable this while using a USB device, you may have problems running that device. However, if you don't use any USB devices, set the option to **Disabled**. It will free up an IRQ for other devices to use.

PCI IRQ Activated By

Options : Edge, Level

This is a rarely seen BIOS feature that allows you to set the method by which the IRQs for your PCI cards are activated / triggered. ISA and old PCI cards are **Edge** triggered (using a single voltage) while newer PCI and AGP cards are **Level** triggered (using multiple voltage levels).

When PCI devices were just introduced, the setting that everyone was asked to use was **Edge** because no PCI device back then supported IRQ sharing. However, now that almost every PCI device supports IRQ sharing and IRQs are usually in shortage, it's best to set it as **Level** so that your PCI devices can share IRQs. So, set it to **Level** unless you are using old edge-triggered PCI cards.

PIRQ_0 Use IRQ No. ~ PIRQ_3 Use IRQ No.

Options : Auto, 3, 4, 5, 7, 9, 10, 11, 12, 14, 15

This feature allows you to manually set the IRQ for a particular device installed on the AGP and PCI bus. This is especially useful when you are transferring a hard disk from one computer to another; and you don't want to reinstall your OS to redetect the IRQ settings. So, by specifying the IRQ for the devices to fit the original settings, you can circumvent a lot of configuration problems after installing the hard disk in a new system.

Notes :-

- If you specify a particular IRQ here, you can't specify the same IRQ for the ISA bus. If you do, you will cause a hardware conflict.
- Each PCI slot is capable of activating up to 4 interrupts - INT A, INT B, INT C and INT D
- The AGP slot is capable of activating up to 2 interrupts - INT A and INT B
- Normally, each slot is allocated INT A. The other interrupts are there as reserves in case the PCI/AGP device requires more than one IRQ or if the IRQ requested has been used up.
- The AGP slot and PCI slot #1 share the same IRQs
- PCI slot #4 and #5 share the same IRQs
- USB uses PIRQ_4

Below is a table showing the relations between PIRQ and INT :-

Signals	AGP Slot PCI Slot 1	PCI Slot 2	PCI Slot 3	PCI Slot 4 PCI Slot 5
PIRQ_0	INT A	INT D	INT C	INT B
PIRQ_1	INT B	INT A	INT D	INT C
PIRQ_2	INT C	INT B	INT A	INT D
PIRQ_3	INT D	INT C	INT B	INT A

You will notice that the interrupts are staggered so that conflicts do not happen easily. Still, because the AGP slot and PCI slot 1 share the same set of IRQs, it's best to only use either one of those two slots unless you don't have other slots to use. The same goes for PCI slot 4 and 5.

Normally, you should just leave it as AUTO. But if you need to assign a particular IRQ to a device on the AGP or PCI bus, here's how you can make use of this BIOS feature. First of all, check out which slot the device is located in. Then, check the table above to determine which is its primary PIRQ. For example, if you have a PCI network card in PCI slot 3, the table shows that its primary PIRQ is PIRQ_2 because all slots are first allocated INT A if possible.

After that, select the IRQ you want to use for that slot by assigning it to the appropriate PIRQ. If the network card (in the example above) requires IRQ 7, then set PIRQ_2 to use IRQ 7. The BIOS will then allocate IRQ 7 to PCI slot 3. It's that easy! **:)**

Just remember that the BIOS will try to allocate the PIRQ linked to INT A for each slot. So, the AGP slot's and PCI slot 1's primary PIRQ is PIRQ_0 while PCI slot 2's primary PIRQ is PIRQ_1 and so on. It's just a matter of linking the IRQ you want to the correct PIRQ for that slot.

The BIOS Optimization Guide

rev. 5.7

Revision History

Comments?

If you have a comment or two about this article, please post them **here**.

Thanks for your time and I hope you enjoyed the article! **:)**

Adrian Wong
Adrian's Rojak Pot
http://www.rojakpot.com/
http://www.adriansrojakpot.com/