Chapter

6

Memory

Memory Basics

This chapter discusses memory from both a physical and logical point of view. First, we'll examine what memory is, where it fits into the PC architecture, and how it works. Then we'll look at the various types of memory, speeds, and packaging of the chips and memory modules you can buy and install.

This chapter also covers the logical layout of memory, defining the various areas of memory and their uses from the system's point of view. Because the logical layout and uses are within the "mind" of the processor, memory mapping and logical layout remain perhaps the most difficult subjects to grasp in the PC universe. This chapter contains useful information that removes the mysteries associated with memory and enables you to get the most out of your system.

Memory is the workspace for the processor. It is a temporary storage area where the programs and data being operated on by the processor must reside. Memory storage is considered temporary because the data and programs remain there only as long as the computer has electrical power or is not reset. Before the computer is shut down or reset, any data that has been changed should be saved to a more permanent storage device (usually a hard disk) so it can be reloaded into memory in the future.

Memory often is called *RAM*, for *random access memory*. Main memory is called RAM because you can randomly (as opposed to sequentially) access any location in memory. This designation is somewhat misleading and often misinterpreted. Read-only memory (ROM), for example, is also randomly accessible, yet is usually differentiated from the system RAM because it maintains data without power and can't normally be written to. Although a hard disk can be used as virtual random access memory, we don't consider that RAM either.

Over the years, the definition of RAM has changed from a simple acronym to become something that means the primary memory workspace the processor uses to run programs, which usually is constructed of a type of chip called dynamic RAM (DRAM). One of the characteristics of DRAM chips (and therefore most types of RAM in general) is that they store data dynamically, which really has two meanings. One meaning is that the information can be written to RAM repeatedly at any time. The other has to do with the fact that DRAM requires the data to be refreshed (essentially rewritten) every few milliseconds or so; faster RAM requires refreshing more often than slower RAM. A type of RAM called static RAM (SRAM) does not require the periodic refreshing. An important characteristic of RAM in general is that data is stored only as long as the memory has electrical power.

Note

Both DRAM and SRAM memory maintain their contents only as long as power is present. However, a different type of memory known as Flash memory does not. Flash memory can retain its contents without power, and it is most commonly used today in digital camera and player media and USB Flash drives. As far as the PC is concerned, a Flash memory device emulates a disk drive (not RAM) and is accessed by a drive letter, just as with any other disk or optical drive.

When we talk about a computer's memory, we usually mean the RAM or physical memory in the system, which is mainly the memory chips or modules the processor uses to store primary active programs and data. This often is confused with the term *storage*, which should be used when referring to things such as disk and tape drives (although they can be used as a form of RAM called virtual memory).

RAM can refer to both the physical chips that make up the memory in the system and the logical mapping and layout of that memory. *Logical mapping* and *layout* refer to how the memory addresses are mapped to actual chips and what address locations contain which types of system information.

People new to computers often confuse main memory (RAM) with disk storage because both have capacities that are expressed in similar megabyte or gigabyte terms. The best analogy to explain the relationship between memory and disk storage I've found is to think of an office with a desk and a file cabinet.

In this popular analogy, the file cabinet represents the system's hard disk, where both programs and data are stored for long-term safekeeping. The desk represents the system's main memory, which allows the person working at the desk (acting as the processor) direct access to any files placed on it. Files represent the programs and documents you can "load" into the memory. For you to work on a particular file, it must first be retrieved from the cabinet and placed on the desk. If the desk is large enough, you might be able to have several files open on it at one time; likewise, if your system has more memory, you can run more or larger programs and work on more or larger documents.

Adding hard disk space to a system is similar to putting a bigger file cabinet in the office—more files can be permanently stored. And adding more memory to a system is like getting a bigger desk—you can work on more programs and data at the same time.

One difference between this analogy and the way things really work in a computer is that when a file is loaded into memory, it is a copy of the file that is actually loaded; the original still resides on the hard disk. Because of the temporary nature of memory, any files that have been changed after being loaded into memory must then be saved back to the hard disk before the system is powered off (which erases the memory). If the changed file in memory is not saved, the original copy of the file on the hard disk remains unaltered. This is like saying that any changes made to files left on the desktop are discarded when the office is closed, although the original files are still preserved in the cabinet.

Memory temporarily stores programs when they are running, along with the data being used by those programs. RAM chips are sometimes termed *volatile storage* because when you turn off your computer or an electrical outage occurs, whatever is stored in RAM is lost unless you saved it to your hard drive.

Because of the volatile nature of RAM, many computer users make it a habit to save their work frequently—a habit I recommend. Many software applications perform periodic saves automatically in order to minimize the potential for data loss.

Physically, the *main memory* in a system is a collection of chips or modules containing chips that are usually plugged into the motherboard. These chips or modules vary in their electrical and physical designs and must be compatible with the system into which they are being installed to function properly. This chapter discusses the various types of chips and modules that can be installed in different systems.

How much you spend on memory for your PC depends mostly on the amount and type of modules you purchase. It is typical to spend the same amount on memory as you do for the motherboard in a system, which is usually anywhere from $50 to $150.

Before the big memory price crash in mid-1996, memory had maintained a fairly consistent price for many years of about $40 per megabyte. A typical configuration back then of 16MB cost more than $600. In fact, memory was so expensive at that time that it was worth more than its weight in gold. These high prices caught the attention of criminals and memory module manufacturers were robbed at gunpoint in several large heists. These robberies were partially induced by the fact that memory was so valuable, the demand was high, and stolen chips or modules were virtually impossible to trace. After the rash of armed robberies and other thefts, memory module manufacturers began posting armed guards and implementing beefed-up security procedures.

By the end of 1996, memory prices had cooled considerably to about $4 per megabyte—a tenfold price drop in less than a year. Prices continued to fall after the major crash until they were at or below 50 cents per megabyte in 1997. All seemed well, until events in 1998 conspired to create a spike in memory prices, increasing them by four times their previous levels. The main culprit was Intel, who had driven the industry to support Rambus DRAM (RDRAM) and then failed to deliver the supporting chipsets on time. The industry was caught in a bind by shifting production to a type of memory for which there were no chipsets or motherboards to plug into, which then created a shortage of the existing (and popular) SDRAM memory. An earthquake in Taiwan (where much of the world's RAM is manufactured) during that year served as the icing on the cake, disrupting production and furthering the spike in prices.

Since then, things have cooled considerably, and memory prices have dropped to all-time lows, with actual prices of under 6 cents per megabyte. In particular, 2001 was a disastrous year in the semiconductor industry, prompted by the dot-com crash as well as worldwide events, and sales dropped well below that of previous years. This conspired to bring memory prices down further than they had ever been and even forced some companies to merge or go out of business.

Memory is less expensive now than ever, but its useful life has also become shorter. New types and speeds of memory are being adopted more quickly than before, and any new systems you purchase now most likely will not accept the same memory as your existing ones. In an upgrade or a repair situation, that means you often have to change the memory if you change the motherboard. The chance that you can reuse the memory in an existing motherboard when upgrading to a new one is slim.

Because of this, you should understand all the various types of memory on the market today, so you can best determine which types are required by which systems, and thus more easily plan for future upgrades and repairs.

To better understand physical memory in a system, you should understand what types of memory are found in a typical PC and what the role of each type is. Three main types of physical memory are used in modern PCs. (Remember, I'm talking about the type of memory chip, not the type of module that memory is stored on.)

- **ROM**—Read-only memory
- **DRAM**—Dynamic random access memory
- **SRAM**—Static RAM

The only type of memory you normally need to purchase and install in a system is DRAM. The other types are built in to the motherboard (ROM), processor (SRAM), and other components such as the video card, hard drives, and so on.

ROM

Read-only memory, or ROM, is a type of memory that can permanently or semipermanently store data. It is called read-only because it is either impossible or difficult to write to. ROM also is often referred to as *nonvolatile memory* because any data stored in ROM remains there, even if the power is turned off. As such, ROM is an ideal place to put the PC's startup instructions—that is, the software that boots the system.

Note that ROM and RAM are not opposites, as some people seem to believe. Both are simply types of memory. In fact, ROM could be classified as technically a subset of the system's RAM. In other words, a portion of the system's random access memory address space is mapped into one or more ROM chips. This is necessary to contain the software that enables the PC to boot up; otherwise, the processor would have no program in memory to execute when it was powered on.

◀◀ For more information on ROM, see "Motherboard ROM BIOS," p. 446.

The main ROM BIOS is contained in a ROM chip on the motherboard, but there are also adapter cards with ROMs on them as well. ROMs on adapter cards contain auxiliary BIOS routines and drivers needed by the particular card, especially for those cards that must be active early in the boot process, such as video cards. Cards that don't need drivers active at boot time typically don't have a ROM because those drivers can be loaded from the hard disk later in the boot process.

Most systems today use a type of ROM called *electrically erasable programmable ROM (EEPROM)*, which is a form of flash memory. Flash is a truly nonvolatile memory that is rewritable, enabling users to easily update the ROM or firmware in their motherboards or any other components (video cards, SCSI cards, peripherals, and so on).

◀◀ For more information on BIOS upgrades, see "Upgrading the BIOS," p. 459.

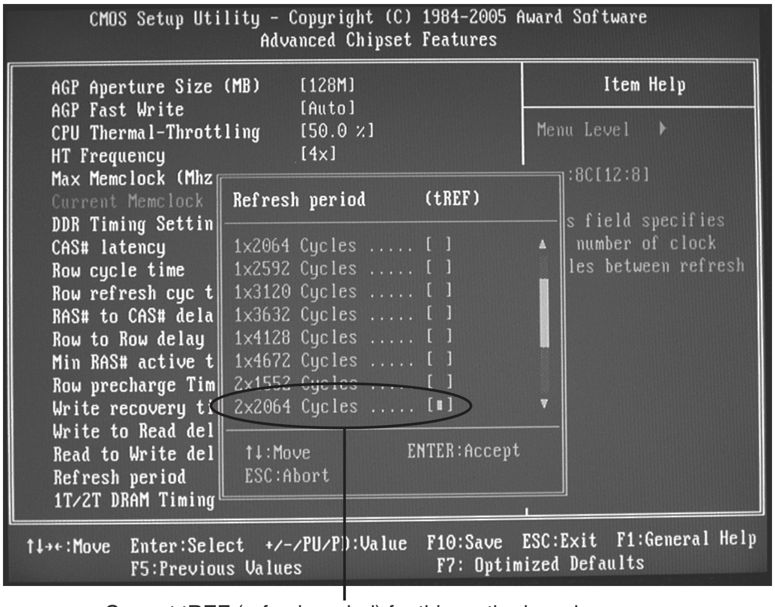
DRAM

Dynamic RAM (DRAM) is the type of memory chip used for most of the main memory in a modern PC. The main advantages of DRAM are that it is very dense, meaning you can pack a lot of bits into a very small chip, and it is inexpensive, which makes purchasing large amounts of memory affordable.

The memory cells in a DRAM chip are tiny capacitors that retain a charge to indicate a bit. The problem with DRAM is that it is dynamic—that is, its contents can be changed. With every keystroke or every mouse swipe, the contents of RAM change. And the entire contents of RAM can be wiped out by a system crash. Also, because of the design, it must be constantly refreshed; otherwise, the electrical charges in the individual memory capacitors will drain and the data will be lost. Refresh occurs when the system memory controller takes a tiny break and accesses all the rows of data in the memory chips. Most systems have a memory controller (normally built into the North Bridge portion of the motherboard chipset or located within the CPU in the case of the AMD Athlon 64 and Opteron processors), which is set for an industry-standard refresh time of 15ms (milliseconds). This means that every 15ms, all the rows in the memory are automatically read to refresh the data.

◀◀ See "Chipsets," p. 273.

Refreshing the memory unfortunately takes processor time away from other tasks because each refresh cycle takes several CPU cycles to complete. In older systems, the refresh cycling could take up to 10% or more of the total CPU time, but with modern systems running in the multi-gigahertz range, refresh overhead is now on the order of a fraction of a percent or less of the total CPU time. Some systems allow you to alter the refresh timing parameters via the CMOS Setup. The time between refresh cycles is known as *tREF* and is expressed not in milliseconds, but in clock cycles (see Figure 6.1).



```
CMOS Setup Utility - Copyright (C) 1984-2005 Award Software
                 Advanced Chipset Features

AGP Aperture Size (MB)    [128M]                  Item Help
AGP Fast Write            [Auto]
CPU Thermal-Throttling    [50.0 %]             Menu Level  ▶
HT Frequency              [4x]
Max Memclock (Mhz                               :8C[12:8]
Current Memclock  Refresh period     (tREF)
DDR Timing Settin                              s field specifies
CAS# latency      1x2064 Cycles ..... [ ]       number of clock
Row cycle time    1x2592 Cycles ..... [ ]      les between refresh
Row refresh cyc t 1x3120 Cycles ..... [ ]
RAS# to CAS# dela 1x3632 Cycles ..... [ ]
Row to Row delay  1x4128 Cycles ..... [ ]
Min RAS# active t 1x4672 Cycles ..... [ ]
Row precharge Tim 2x1552 Cycles ..... [ ]
Write recovery ti 2x2064 Cycles ..... [■]
Write to Read del
Read to Write del  ↑↓:Move       ENTER:Accept
Refresh period     ESC:Abort
1T/2T DRAM Timing

↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
           F5:Previous Values              F7: Optimized Defaults
```

Figure 6.1 The refresh period dialog box and other advanced memory timings can be adjusted manually through the system CMOS setup program.

It's important to be aware that increasing the time between refresh cycles (tREF) to speed up your system can allow some of the memory cells to begin draining prematurely, which can cause random soft memory errors to appear.

A *soft error* is a data error that is not caused by a defective chip. To avoid soft errors, it is usually safer to stick with the recommended or default refresh timing. Because refresh consumes less than 1% of modern system overall bandwidth, altering the refresh rate has little effect on performance. It is almost always best to use default or automatic settings for any memory timings in the BIOS Setup. Many modern systems don't allow changes to memory timings and are permanently set to automatic settings. On an automatic setting, the motherboard reads the timing parameters out of the serial presence detect (SPD) ROM found on the memory module and sets the cycling speeds to match.

DRAMs use only one transistor and capacitor pair per bit, which makes them very dense, offering more memory capacity per chip than other types of memory. DRAM chips are currently being prepared for production with densities up to 2Gb. This means that DRAM chips are available with over one billion transistors! Compare this to a Core 2 Duo, which has 291 million transistors, and it makes the processor look wimpy by comparison. The difference is that in a memory chip, the transistors and capacitors are all consistently arranged in a (normally square) grid of simple repetitive structures, unlike the processor, which is a much more complex circuit of different structures and elements interconnected in a highly irregular fashion.

The transistor for each DRAM bit cell reads the charge state of the adjacent capacitor. If the capacitor is charged, the cell is read to contain a 1; no charge indicates a 0. The charge in the tiny capacitors is constantly draining, which is why the memory must be refreshed constantly. Even a momentary power interruption, or anything that interferes with the refresh cycles, can cause a DRAM memory cell to lose the charge and thus the data. If this happens in a running system, it can lead to blue screens, global protection faults, corrupted files, and any number of system crashes.

DRAM is used in PC systems because it is inexpensive and the chips can be densely packed, so a lot of memory capacity can fit in a small space. Unfortunately, DRAM is also relatively slow, typically much slower than the processor. For this reason, many types of DRAM architectures have been developed to improve performance. These architectures are covered later in the chapter.

Cache Memory: SRAM

Another distinctly different type of memory exists that is significantly faster than most types of DRAM. SRAM stands for *static RAM*, which is so named because it does not need the periodic refresh rates like DRAM. Because of how SRAMs are designed, not only are refresh rates unnecessary, but SRAM is much faster than DRAM and much more capable of keeping pace with modern processors.

SRAM memory is available in access times of 2ns or less, so it can keep pace with processors running 500MHz or faster. This is because of the SRAM design, which calls for a cluster of six transistors for each bit of storage. The use of transistors but no capacitors means that refresh rates are not necessary because there are no capacitors to lose their charges over time. As long as there is power, SRAM remembers what is stored. With these attributes, why don't we use SRAM for all system memory? The answers are simple.

Compared to DRAM, SRAM is much faster but also much lower in density and much more expensive (see Table 6.1). The lower density means that SRAM chips are physically larger and store fewer bits overall. The high number of transistors and the clustered design mean that SRAM chips are both physically larger and much more expensive to produce than DRAM chips. For example, a DRAM module might contain 64MB of RAM or more, whereas SRAM modules of the same approximate physical size would have room for only 2MB or so of data and would cost the same as the 64MB DRAM module. Basically, SRAM is up to 30 times larger physically and up to 30 times more expensive than DRAM. The high cost and physical constraints have prevented SRAM from being used as the main memory for PC systems.

Table 6.1 Comparing DRAM and SRAM

Type	Speed	Density	Cost
DRAM	Slow	High	Low
SRAM	Fast	Low	High

Even though SRAM is too expensive for PC use as main memory, PC designers have found a way to use SRAM to dramatically improve PC performance. Rather than spend the money for all RAM to be SRAM memory, which can run fast enough to match the CPU, designing in a small amount of high-speed SRAM memory, called *cache memory*, is much more cost-effective. The cache runs at speeds close to or even equal to the processor and is the memory from which the processor usually directly reads from and writes to. During read operations, the data in the high-speed cache memory is resupplied from the lower-speed main memory or DRAM in advance. To convert access time in nanoseconds to MHz, use the following formula:

$$1 / \text{nanoseconds} \times 1000 = \text{MHz}$$

Likewise, to convert from MHz to nanoseconds, use the following inverse formula:

$$1 / \text{MHz} \times 1000 = \text{nanoseconds}$$

Today we have memory that runs faster than 1GHz (1 nanosecond), but up until the late 1990s, DRAM was limited to about 60ns (16MHz) in speed. Up until processors were running at speeds of 16MHz, the available DRAM could fully keep pace with the processor and motherboard, meaning that there was no need for cache. However, as soon as processors crossed the 16MHz barrier, the available DRAM could no longer keep pace, and SRAM cache began to enter PC system designs. This occurred way back in 1986 and 1987 with the debut of systems with the 386 processor running at speeds of 16MHz to 20MHz or faster. These were among the first PC systems to employ what's called *cache memory*, a high-speed buffer made up of SRAM that directly feeds the processor. Because the cache can run at the speed of the processor, it acts as a buffer between the processor and the slower DRAM in the system. The cache controller anticipates the processor's memory needs and preloads the high-speed cache memory with data. Then, as the processor calls for a memory address, the data can be retrieved from the high-speed cache rather than the much lower-speed main memory.

Cache effectiveness can be expressed by a hit ratio. This is the ratio of cache hits to total memory accesses. A *hit* occurs when the data the processor needs has been preloaded into the cache from the main memory, meaning the processor can read it from the cache. A cache *miss* is when the cache controller did not anticipate the need for a specific address and the desired data was not preloaded into the cache. In that case the processor must retrieve the data from the slower main memory, instead of the faster cache. Anytime the processor reads data from main memory, the processor must wait longer because the main memory cycles at a much slower rate than the processor. As an example, if the processor with integral on-die cache is running at 3.6GHz (3,600MHz) on an 800MHz bus, both the processor and the integral cache would be cycling at 0.28ns, while the main memory would most likely be cycling almost five times more slowly at 800MHz (1.25ns) DDR2. So, every time the 3.6GHz processor reads from main memory, it would effectively slow down to only 800MHz! The slowdown is accomplished by having the processor execute what are called *wait states*, which are cycles in which nothing is done; the processor essentially cools its heels while waiting for the slower main memory to return the desired data. Obviously, you don't want your processors slowing down, so cache function and design become more important as system speeds increase.

To minimize the processor being forced to read data from the slow main memory, two or three stages of cache usually exist in a modern system, called Level 1 (L1), Level 2 (L2), and Level 3 (L3). The L1 cache is also called *integral* or *internal cache* because it has always been built directly into the processor as part of the processor die (the raw chip). Because of this, L1 cache always runs at the full speed of the processor core and is the fastest cache in any system. All 486 and higher processors incorporate integral L1 cache, making them significantly faster than their predecessors. L2 cache was originally called *external cache* because it was external to the processor chip when it first appeared. Originally, this meant it was installed on the motherboard, as was the case with all 386, 486, and first generation Pentium systems. In those systems, the L2 cache runs at motherboard and CPU bus speed because it is installed on the motherboard and is connected to the CPU bus. You typically find the L2 cache physically adjacent to the processor socket in Pentium and earlier systems.

◀◀ See "How Cache Works," p. 73.

In the interest of improved performance, later processor designs from Intel and AMD included the L2 cache as a part of the processor. In all processors since late 1999 (and some earlier models), the L2 cache is directly incorporated as a part of the processor die just like the L1 cache. In chips with on-die L2, the cache runs at the full core speed of the processor and is much more efficient. By contrast, most processors from 1999 and earlier with integrated L2 had the L2 cache in separate chips that were external to the main processor core. The L2 cache in many of these older processors ran at only half or one-third the processor core speed. Cache speed is very important, so systems having L2 cache on the motherboard were the slowest. Including L2 inside the processor made it faster, and including it directly on the processor die (rather than as chips external to the die) is the fastest yet. Any chip that has on-die full core speed L2 cache has a distinct performance advantage over any chip that doesn't.

A third-level or L3 cache has been present in some high-end workstation and server processors since 2001. The first desktop PC processor with L3 cache was the Pentium 4 Extreme Edition, a high-end chip introduced in late 2003 with 2MB of on-die L3 cache. Although it seemed at the time that this would be a forerunner of widespread L3 cache in desktop processors, later versions of the Pentium 4 Extreme Edition (as well as its successor, the Pentium Extreme Edition) dropped the L3 cache. Instead, larger L2 cache sizes are used to improve performance. Although L3 cache may eventually become a standard feature in processors, the current trend is to use the extra die space for multiple cores with larger L2 caches instead.

The key to understanding both cache and main memory is to see where they fit in the overall system architecture.

See Chapter 4, "Motherboards and Buses," for diagrams showing recent systems with different types of cache memory. Table 6.2 illustrates the need for and function of cache memory in modern systems.

Cache designs originally were *asynchronous*, meaning they ran at a clock speed that was not identical or in sync with the processor bus. Starting with the 430FX chipset released in early 1995, a new type of synchronous cache design was supported. It required that the chips now run in sync or at the same identical clock timing as the processor bus, further improving speed and performance. Also added at that time was a feature called *pipeline burst mode*, which reduces overall cache latency (wait states) by allowing single-cycle accesses for multiple transfers after the first one. Because both synchronous and pipeline burst capability came at the same time in new modules, specifying one usually implies the other. Synchronous pipeline burst cache allowed for about a 20% improvement in overall system performance, which was a significant jump.

The cache controller for a modern system is contained in either the North Bridge of the chipset, as with Pentium and lesser systems, or within the processor, as with the Pentium II, Athlon, and newer systems. The capabilities of the cache controller dictate the cache's performance and capabilities. One important thing to note for older systems is that most external cache controllers have a limitation on the amount of memory that can be cached. Often, this limit can be quite low, as with the 430TX chipset–based Pentium systems from the late ’90s. Most original Pentium-class chipsets from that time, such as the 430FX/VX/TX, can cache data only within the first 64MB of system RAM. If you add more memory than that to those systems, you will see a noticeable slowdown in system performance because all data outside the first 64MB is never cached and is always accessed with all the wait states required by the slower DRAM. Depending on what software you use and where data is stored in memory, this can be significant. For example, 32-bit operating systems such as Windows load from the top down, so if you had 96MB of RAM, the operating system and applications would load directly into the upper 32MB, which is beyond the 64MB cacheability limit and therefore not cached. This results in a dramatic slowdown in overall system use. Removing memory to bring the system total down to the cacheable limit of 64MB would actually speed up the system in this case. In short, it is unwise to install more main RAM memory than your system (CPU or chipset) can cache. Fortunately, this limitation does not affect modern systems with Pentium III and newer processors because they can cache all addressable memory.

Chipsets made for the Pentium Pro and later processors do not control the L2 cache because the cache was moved into the processor instead. So, with the Pentium Pro and beyond, the processor sets the cacheability limits. The Pentium Pro and some of the earlier Pentium IIs can only cache memory within the first 512MB of address space. The later Pentium IIs can cache up to 4GB, whereas the Pentium III and later can cache any and all addressable memory up to 64GB, well beyond the maximum RAM support of any chipsets.

Table 6.2 shows the evolutionary changes in cache and main memory in processors since the first-generation Pentium. Note especially how the L2 cache moved from being external to on-chip and then on-die, while also getting larger and larger. In addition, there have been significant increases in CPU speed as well as cache, memory, and bus speeds.

RAM Types and Performance

The speed and performance issue with memory is confusing to some because memory speed is sometimes expressed in nanoseconds (ns) and processor speed has always been expressed in megahertz (MHz) or gigahertz (GHz). Newer and faster types of memory usually have speeds expressed in MHz, thus adding to the confusion. Fortunately, you can easily translate MHz/GHz to ns, and vice versa.

A *nanosecond* is defined as one billionth of a second—a very short time indeed. To put some perspective on that, the speed of light is 186,282 miles (299,792 kilometers) per second in a vacuum. In one billionth of a second, a beam of light travels a mere 11.80 inches or 29.98 centimeters—less than the length of a typical ruler!

Chip and system speeds have often been expressed in megahertz (MHz), which is millions of cycles per second, or gigahertz (GHz), which is billions of cycles per second. Today’s processors run in the 2GHz–4GHz range with most performance improvements coming from changes in CPU design (such as multiple cores) rather than pure clock speed increases.

Because it is confusing to speak in these different terms for speeds, I thought it would be interesting to see how they compare. Earlier in this chapter I listed formulas you could use to mathematically convert these values. Table 6.3 shows the relationship between common nanosecond (ns) and megahertz (MHz) speeds associated with PCs from yesterday to today and tomorrow.

Table 6.2 Evolutionary Changes in L1 (Internal) Cache, L2 (External) Cache, and Main Memory in Modern Processors

CPU Type Extreme	Pentium	Pentium Pro	Pentium II	AMD K6-2	AMD K6-3
CPU speed	233MHz	200MHz	450MHz	550MHz	450MHz
L1 cache speed	4.3ns (233MHz)	5.0ns (200MHz)	2.2ns (450MHz)	1.8ns (550MHz)	2.2ns (450MHz)
L1 cache size	16K	32K	32K	64K	64K
L2 cache type	external	on-chip	on-chip	external	on-die
CPU/L2 speed ratio	—	1/1	1/2	—	1/1
L2 cache speed	15ns (66MHz)	5ns (200MHz)	4.4ns (225MHz)	10ns (100MHz)	2.2ns (450MHz)
L2 cache size	—	256K	512K	—	256K
CPU bus bandwidth	533MBps	533MBps	800MBps	800MBps	800MBps
Memory bus speed	60ns (16MHz)	60ns (16MHz)	10ns (100MHz)	10ns (100MHz)	10ns (100MHz)

1. *4M total L2 cache / 2M per core*
2. *8M total L2 cache / 4M per core*

Table 6.3 The Relationship Between Megahertz (MHz) and Cycle Times in Nanoseconds (ns)

Clock Speed	Cycle Time	Clock Speed	Cycle Time	Clock Speed	Cycle Time	Clock Speed	Cycle Time
4.77MHz	210ns	150MHz	6.7ns	550MHz	1.82ns	1,000MHz	1.00ns
6MHz	167ns	166MHz	6.0ns	566MHz	1.77ns	1,100MHz	0.91ns
8MHz	125ns	180MHz	5.6ns	600MHz	1.67ns	1,133MHz	0.88ns
10MHz	100ns	200MHz	5.0ns	633MHz	1.58ns	1,200MHz	0.83ns
12MHz	83ns	225MHz	4.4ns	650MHz	1.54ns	1,300MHz	0.77ns
16MHz	63ns	233MHz	4.3ns	666MHz	1.50ns	1,400MHz	0.71ns
20MHz	50ns	250MHz	4.0ns	700MHz	1.43ns	1,500MHz	0.67ns
25MHz	40ns	266MHz	3.8ns	733MHz	1.36ns	1,600MHz	0.63ns
33MHz	30ns	300MHz	3.3ns	750MHz	1.33ns	1,700MHz	0.59ns
40MHz	25ns	333MHz	3.0ns	766MHz	1.31ns	1,800MHz	0.56ns
50MHz	20ns	350MHz	2.9ns	800MHz	1.25ns	1,900MHz	0.53ns
60MHz	17ns	366MHz	2.7ns	833MHz	1.20ns	2,000MHz	0.50ns
66MHz	15ns	400MHz	2.5ns	850MHz	1.18ns	2,100MHz	0.48ns
75MHz	13ns	433MHz	2.3ns	866MHz	1.15ns	2,200MHz	0.45ns
80MHz	13ns	450MHz	2.2ns	900MHz	1.11ns	2,300MHz	0.43ns
100MHz	10ns	466MHz	2.1ns	933MHz	1.07ns	2,400MHz	0.42ns
120MHz	8.3ns	500MHz	2.0ns	950MHz	1.05ns	2,500MHz	0.40ns
133MHz	7.5ns	533MHz	1.88ns	966MHz	1.04ns	2,600MHz	0.38ns

Pentium III	Athlon	Athlon XP	Pentium 4	Athlon 64 X2	Core 2 Duo	Core 2
1.4GHz	1.4GHz	2.2GHz	3.8GHz	3GHz	2.66GHz	2.93GHz
0.71ns (1.4GHz)	0.71ns (1.4GHz)	0.45ns (2.2GHz)	0.26ns (3.8GHz)	0.33ns (3GHz)	0.37ns (2.66GHz)	0.34ns (2.93GHz)
32K	128K	128K	20K	256K	64K	64K
on-die	on-die	on-die	on-die	on-die	On-die	on-die
1/1	1/1	1/1	1/1	1/1	1/1	1/1
0.71ns (1.4GHz)	0.71ns (1.4GHz)	0.45ns (2.2GHz)	0.26ns (3.8GHz)	0.33ns (3GHz)	0.37ns (2.66GHz)	0.34ns (2.93GHz)
512K	256K	512K	2M	2M	4M[1]	8M[2]
1,066MBps	2,133MBps	3,200MBps	6,400MBps	4,000MBps	8,533MBps	8,533MBps
7.5ns (133MHz)	3.8ns (266MHz)	2.5ns (400MHz)	1.25ns (800MHz)	2.5ns (400MHz)	0.94ns (1066MHz)	0.94ns (1066MHz)

Table 6.3 Continued

Clock Speed	Cycle Time	Clock Speed	Cycle Time	Clock Speed	Cycle Time	Clock Speed	Cycle Time
2,700MHz	0.37ns	3,300MHz	0.303ns	3,900MHz	0.256ns	4,500MHz	0.222ns
2,800MHz	0.36ns	3,400MHz	0.294ns	4,000MHz	0.250ns	4,600MHz	0.217ns
2,900MHz	0.34ns	3,500MHz	0.286ns	4,100MHz	0.244ns	4,700MHz	0.213ns
3,000MHz	0.333ns	3,600MHz	0.278ns	4,200MHz	0.238ns	4,800MHz	0.208ns
3,100MHz	0.323ns	3,700MHz	0.270ns	4,300MHz	0.233ns	4,900MHz	0.204ns
3,200MHz	0.313ns	3,800MHz	0.263ns	4,400MHz	0.227ns	5,000MHz	0.200ns

As you can see from Table 6.3, as clock speeds increase, cycle time decreases proportionately.

Over the development life of the PC, memory has had a difficult time keeping up with the processor, requiring several levels of high-speed cache memory to intercept processor requests for the slower main memory. More recently, however, systems using DDR, DDR2, and DDR3 SDRAM have memory bus performance equaling that of the processor bus. When the speed of the memory bus equals the speed of the processor bus, main memory performance is optimum for that system.

For example, using the information in Table 6.3, you can see that the 60ns DRAM memory used in the original Pentium and Pentium II PCs up until 1998 works out to be an extremely slow 16.7MHz! This slow 16.7MHz memory was installed in systems running processors up to 300MHz or faster on a processor bus speed of 66MHz, resulting in a large mismatch between processor bus and main memory performance. However, starting in 1998 the industry shifted to faster SDRAM memory, which was able to match the 66MHz speed of the processor bus at the time. From that point forward, memory has largely evolved in step with the processor bus, with newer and faster types coming out to match any increases in processor bus speeds.

By the year 2000 the dominant processor bus and memory speeds had increased to 100MHz and even 133MHz (called PC100 and PC133 SDRAM, respectively). Starting in early 2001, double data rate (DDR) SDRAM memory of 200MHz and 266MHz become popular. In 2002 DDR memory increased to 333MHz, and in 2003 the speeds increased further to 400MHz. During 2004 we saw the introduction of DDR2, first at 400MHz and then at 533MHz. DDR2 memory continued to match processor bus speed increases in PCs during 2005 and 2006, rising to 667MHz and 800MHz during that time. By 2007 DDR2 memory was available at speeds of up to 1,066MHz, and DDR3 came on the market at 1,066MHz and faster. Table 6.4 lists the primary types and performance levels of PC memory.

Table 6.4 Types and Performance Levels of Memory Used in PCs

Memory Type	Years Popular	Module Type	Voltage	Max. Clock Speed	Max. Throughput Single-Channel	Max. Throughput Dual-Channel
Fast Page Mode (FPM) DRAM	1987–1995	30/72-pin SIMM	5V	22MHz	177MBps	N/A
Extended Data Out (EDO) DRAM	1995–1998	72-pin SIMM	5V	33MHz	266MBps	N/A
Single Data Rate (SDR) SDRAM	1998–2002	168-pin DIMM	3.3V	133MHz	1,066MBps	N/A
Rambus DRAM (RDRAM)	2000–2002	184-pin RIMM	2.5V	1,066MTps	2,133MBps	4,266MBps
Double Data Rate (DDR) SDRAM	2002–2005	184-pin DIMM	2.5V	400MTps	3,200MBps	6,400MBps
DDR2 SDRAM	2005–2008	240-pin DDR2 DIMM	1.8V	1,066MTps	8,533MBps	17,066MBps
DDR3 SDRAM	2008+	240-pin DDR3 DIMM	1.5V	1,600MTps	12,800MBps	25,600MBps

MHz = Megacycles per second

MTps = Megatransfers per second

MBps = Megabytes per second

SIMM = Single inline memory module

DIMM = Dual inline memory module

The following sections look at these memory types in more detail.

Fast Page Mode DRAM

Standard DRAM is accessed through a technique called *paging*. Normal memory access requires that a row and column address be selected, which takes time. Paging enables faster access to all the data within a given row of memory by keeping the row address the same and changing only the column. Memory that uses this technique is called *Page Mode* or *Fast Page Mode* memory. Other variations on Page Mode were called *Static Column* or *Nibble Mode* memory.

Paged memory is a simple scheme for improving memory performance that divides memory into pages ranging from 512 bytes to a few kilobytes long. The paging circuitry then enables memory locations in a page to be accessed with fewer wait states. If the desired memory location is outside the current page, one or more wait states are added while the system selects the new page.

To improve further on memory access speeds, systems have evolved to enable faster access to DRAM. One important change was the implementation of burst mode access in the 486 and later processors. Burst mode cycling takes advantage of the consecutive nature of most memory accesses. After setting up the row and column addresses for a given access, using burst mode, you can then access the next three adjacent addresses with no additional latency or wait states. A burst access usually is limited to four total accesses. To describe this, we often refer to the timing in the number of cycles for each access. A typical burst mode access of standard DRAM is expressed as x-y-y-y; x is the time for the first access (latency plus cycle time), and y represents the number of cycles required for each consecutive access.

Standard 60ns-rated DRAM normally runs 5-3-3-3 burst mode timing. This means the first access takes a total of five cycles (on a 66MHz system bus, this is about 75ns total, or 5×15ns cycles), and the consecutive cycles take three cycles each (3×15ns = 45ns). As you can see, the actual system timing is somewhat less than the memory is technically rated for. Without the bursting technique, memory access would be 5-5-5-5 because the full latency is necessary for each memory transfer. The 45ns cycle time during burst transfers equals about a 22.2MHz effective clock rate; on a system with a 64-bit (8-byte) wide memory bus, this would result in a maximum throughput of 177MBps (22.2MHz × 8 bytes = 177MBps).

DRAM memory that supports paging and this bursting technique is called *Fast Page Mode (FPM)* memory. The term comes from the capability of memory accesses to data on the same page to be done with less latency. Most 386, 486, and Pentium systems from 1987 through 1995 used FPM memory, which came in either 30-pin or 72-pin SIMM form.

Another technique for speeding up FPM memory is called *interleaving*. In this design, two separate banks of memory are used together, alternating access from one to the other as even and odd bytes. While one is being accessed, the other is being precharged, when the row and column addresses are being selected. Then, by the time the first bank in the pair is finished returning data, the second bank in the pair is finished with the latency part of the cycle and is now ready to return data. While the second bank is returning data, the first bank is being precharged, selecting the row and column address of the next access. This overlapping of accesses in two banks reduces the effect of the latency or precharge cycles and allows for faster overall data retrieval. The only problem is that to use interleaving, you must install identical pairs of banks together, doubling the number of modules required. This was popular on 32-bit wide memory systems for 486 processors but fell out of favor on Pentium systems because of their 64-bit wide memory widths. To perform interleaving on a Pentium machine using FPM memory, you would need to install memory 128 bits at a time, meaning four 72-pin SIMMs at a time.

Extended Data Out RAM (EDO)

In 1995, a newer type of DRAM called *extended data out (EDO)* RAM became available for Pentium systems. EDO, a modified form of FPM memory, is sometimes referred to as *Hyper Page mode*. EDO was invented and patented by Micron Technology, although Micron licensed production to many other memory manufacturers.

EDO memory consists of specially manufactured chips that allow a timing overlap between successive accesses. The name *extended data out* refers specifically to the fact that unlike FPM, the data output drivers on the chip are not turned off when the memory controller removes the column address to begin the next cycle. This enables the next cycle to overlap the previous one, saving approximately 10ns per cycle.

The effect of EDO is that cycle times are improved by enabling the memory controller to begin a new column address instruction while it is reading data at the current address. This is almost identical to what was achieved in older systems by interleaving banks of memory, but unlike interleaving, with EDO you didn't need to install two identical banks of memory in the system at a time.

EDO RAM allows for burst mode cycling of 5-2-2-2, compared to the 5-3-3-3 of standard fast page mode memory. To do four memory transfers, then, EDO would require 11 total system cycles, compared to 14 total cycles for FPM. This is a 22% improvement in overall cycling time. The resulting two-cycle (30ns) cycle time during burst transfers equals a 33.3MHz effective clock rate, compared to 45ns/22MHz for FPM. On a system with a 64-bit (8-byte) wide memory bus, this would result in a maximum throughput of 266MBps (33.3MHz × 8 bytes = 266MBps). Due to the processor cache, EDO typically increased overall system benchmark speed by only 5% or less. Even though the overall system improvement was small, the important thing about EDO was that it used the same basic DRAM chip design as FPM, meaning that there was practically no additional cost over FPM. In fact, in its heyday EDO cost less than FPM and yet offered higher performance.

EDO RAM generally came in 72-pin SIMM form. Figure 6.4 (later in this chapter) shows the physical characteristics of these SIMMs.

To actually use EDO memory, your motherboard chipset had to support it. Most motherboard chipsets introduced on the market from 1995 (Intel 430FX) through 1997 (Intel 430TX) offered support for EDO, making EDO the most popular form of memory in PCs from 1995 through 1998. Because EDO memory chips cost the same to manufacture as standard chips, combined with Intel's support of EDO in motherboard chipsets, the PC market jumped on the EDO bandwagon full force.

◀◀ See "Fifth-generation (P5 Pentium Class) Chipsets," p. 284, and "Sixth-generation (P6 Pentium Pro/II/III Class) Chipsets," p. 286.

EDO RAM was used in systems with CPU bus speeds of up to 66MHz, which fit perfectly with the PC market up through 1998. However, starting in 1998, with the advent of 100MHz and faster system bus speeds, the market for EDO rapidly declined, and faster SDRAM architecture became the standard.

One variation of EDO that never caught on was called burst EDO (BEDO). BEDO added burst capabilities for even speedier data transfers than standard EDO. Unfortunately, the technology was owned by Micron and not a free industry standard, so only one chipset (Intel 440FX Natoma) ever supported it. BEDO was quickly overshadowed by industry-standard SDRAM, which came into favor among PC system chipset and system designers over proprietary designs. As such, BEDO never really saw the light of production, and to my knowledge no systems ever used it.

SDRAM

SDRAM is short for *synchronous DRAM*, a type of DRAM that runs in synchronization with the memory bus. SDRAM delivers information in very high-speed bursts using a high-speed clocked interface. SDRAM removes most of the latency involved in asynchronous DRAM because the signals are already in synchronization with the motherboard clock.

As with any type of memory on the market, motherboard chipset support is required before it can be usable in systems. Starting in 1996 with the 430VX and 430TX, most of Intel's chipsets began to support industry-standard SDRAM, and in 1998 the introduction of the 440BX chipset caused SDRAM to eclipse EDO as the most popular type on the market.

SDRAM performance is dramatically improved over that of FPM or EDO RAM. However, because SDRAM is still a type of DRAM, the initial latency is the same, but burst mode cycle times are much faster than with FPM or EDO. SDRAM timing for a burst access would be 5-1-1-1, meaning that four memory reads would complete in only eight system bus cycles, compared to 11 cycles for EDO and 14 cycles for FPM. This makes SDRAM almost 20% faster than EDO.

Besides being capable of working in fewer cycles, SDRAM is also capable of supporting up to 133MHz (7.5ns) system bus cycling. Most PC systems sold from 1998 through 2002 included SDRAM memory.

SDRAM is sold in DIMM form and is normally rated by clock speed (MHz) rather than cycling time (ns), which was confusing during the initial change from FPM and EDO DRAM. Figure 6.5 (later in this chapter) shows the physical characteristics of DIMMs.

To meet the stringent timing demands of its chipsets, Intel created specifications for SDRAM called PC66, PC100, and PC133. For example, you would think 10ns would be considered the proper rating for 100MHz operation, but the PC100 specification promoted by Intel calls for faster 8ns memory to ensure all timing parameters could be met with sufficient margin for error.

In May 1999, the Joint Electron Device Engineering Council (JEDEC) created a specification called PC133. It achieved this 33MHz speed increase by taking the PC100 specification and tightening up the timing and capacitance parameters. The faster PC133 quickly caught on for any systems running a 133MHz processor bus. The original chips used in PC133 modules were rated for exactly 7.5ns or 133MHz; later ones were rated at 7.0ns, which is technically 143MHz. These faster chips were still used on PC133 modules, but they allowed for improvements in column address strobe latency (abbreviated as CAS or CL), which somewhat improves overall memory cycling time.

Note

JEDEC is the semiconductor engineering standardization body of the Electronic Industries Alliance (EIA), a trade association that represents all areas of the electronics industry. JEDEC was originally created in 1960 and governs the standardization of all types of semiconductor devices, integrated circuits, and modules. JEDEC has about 300 member companies, including memory, chipset, and processor manufacturers as well as practically any company involved in manufacturing computer equipment using industry-standard components.

The idea behind JEDEC is simple: to create open standards that can be freely adopted throughout the industry. For example, if one company were to create a proprietary memory technology, other companies who wanted to manufacture components compliant with that memory would have to pay license fees, assuming the company that owned the technology was interested in licensing at all! Parts would be more proprietary in nature, causing problems with interchangeability or sourcing reasonably priced replacements. In addition, those companies licensing the technology would have no control over the evolution of the technology or any future changes made by the owner company.

JEDEC prevents this type of scenario for things such as memory by getting all the memory manufacturers to work together to create shared industry standards covering memory chips and modules. JEDEC-approved standards for memory can then be freely shared by all the member companies, and no one single company has control over a given standard, or any of the companies producing compliant components. FPM, SDRAM, DDR, DDR2, and DDR3 are all examples of JEDEC memory standards used in PCs, whereas memory such as EDO and RDRAM are proprietary examples. You can find out more about JEDEC standards for memory and other semiconductor technology at www.jedec.org.

Table 6.5 shows the timing, rated chip speeds, and standard module speeds for various SDRAM DIMMs.

Table 6.5 SDRAM Timing, Actual Speed, and Rated Speed

Timing	Rated Chip Speed	Standard Module Speed
15ns	66MHz	PC66
10ns	100MHz	PC66
8ns	125MHz	PC100
7.5ns	133MHz	PC133
7.0ns	143MHz	PC133

SDRAM normally came in 168-pin DIMMs, running at several different speeds. Table 6.6 shows the standard single data rate SDRAM module speeds and resulting throughputs.

Table 6.6 JEDEC Standard SDRAM Module (168-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)
PC66	10ns	66	1	66	8	533
PC100	8ns	100	1	100	8	800
PC133	7ns	133	1	133	8	1,066

MTps = Megatransfers per second

MBps = Megabytes per second

ns = Nanoseconds (billionths of a second)

DIMM = Dual inline memory module

▶▶ See "SIMMs, DIMMs, and RIMMS," p. 533.

Some module manufacturers sold modules they claimed were "PC150" or "PC166," even though those speeds did not exist as official JEDEC or Intel standards, and no chipsets or processors officially supported those speeds. These modules actually used hand-picked 133MHz rated chips that could run overclocked at 150MHz or 166MHz speeds. In essence, PC150 or PC166 memory was PC133 memory that was tested to run at overclocked speeds not supported by the original chip manufacturer. This overclockable memory was sold at a premium to enthusiasts who wanted to overclock their motherboard chipsets, thereby increasing the speed of the processor and memory bus.

Caution

In general, PC133 memory is considered to be backward compatible with PC100 memory. However, some chipsets or motherboards had more specific requirements for specific types of 100MHz or 133MHz chips and module designs. If you need to upgrade an older system that requires PC100 memory, you should not purchase PC133 memory unless the memory is specifically identified by the memory vendor as being compatible with the system. You can use the online memory-configuration tools provided by most major memory vendors to ensure that you get the right memory for your system.

DDR SDRAM

Double data rate (DDR) SDRAM memory is a JEDEC standard that is an evolutionary upgrade in which data is transferred twice as quickly as standard SDRAM. Instead of doubling the actual clock rate, DDR memory achieves the doubling in performance by transferring twice per transfer cycle: once at the leading (falling) edge and once at the trailing (rising) edge of the cycle (see Figure 6.2). This effectively doubles the transfer rate, even though the same overall clock and timing signals are used.

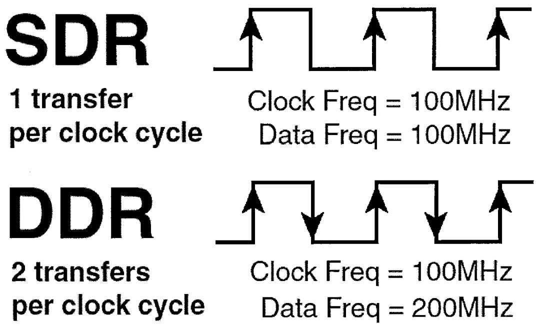


Figure 6.2 SDR (single data rate) versus DDR (double data rate) cycling.

DDR SDRAM first came to market in the year 2000 and was initially used on high-end graphics cards since there weren't any motherboard chipsets to support it at the time. DDR finally became popular in 2002 with the advent of mainstream supporting motherboards and chipsets. From 2002 through 2005, DDR was the most popular type of memory in mainstream PCs. DDR SDRAM uses a DIMM module design with 184 pins. Figure 6.6 (later in this chapter) shows the 184-pin DDR DIMM.

DDR DIMMs come in a variety of speed or throughput ratings and normally run on 2.5 volts. They are basically an extension of the standard SDRAM DIMMs redesigned to support double clocking, where data is sent on each clock transition (twice per cycle) rather than once per cycle as with standard SDRAM. To eliminate confusion with DDR, regular SDRAM is often called *single data rate (SDR)*. Table 6.7 compares the various types of industry-standard DDR SDRAM modules. As you can see, the raw chips are designated by their speed in megatransfers per second, whereas the modules are designated by their approximate throughput in megabytes per second.

Table 6.7 JEDEC Standard DDR Module (184-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)	Dual-Channel Transfer Rate (MBps)
PC1600	DDR200	100	2	200	8	1,600	3,200
PC2100	DDR266	133	2	266	8	2,133	4,266
PC2700	DDR333	166	2	333	8	2,667	5,333
PC3200	DDR400	200	2	400	8	3,200	6,400

MTps = Megatransfers per second
MBps = Megabytes per second
DIMM = Dual inline memory module
DDR = Double data rate

The major memory chip and module manufacturers normally produce parts that conform to the official JEDEC standard speed ratings. However, to support overclocking, several memory module manufacturers purchase unmarked and untested chips from the memory chip manufacturers, then independently test and sort them by how fast they run. These are then packaged into modules with unofficial designations and performance figures that exceed the standard ratings. Table 6.8 shows the popular unofficial speed ratings I've seen on the market. Note that since the speeds of these modules are beyond the standard default motherboard and chipset speeds, you won't see any advantage to using these unless you are overclocking your system to match.

Table 6.8 Overclocked (non-JEDEC) DDR Module (184-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)	Dual-Channel Transfer Rate (MBps)
PC3500	DDR433	216	2	433	8	3,466	6,933
PC3700	DDR466	216	2	466	8	3,733	7,466
PC4000	DDR500	250	2	500	8	4,000	8,000
PC4200	DDR533	266	2	533	8	4,266	8,533
PC4400	DDR550	275	2	550	8	4,400	8,800
PC4800	DDR600	300	2	600	8	4,800	9,600

MTps = Megatransfers per second

MBps = Megabytes per second

DIMM = Dual inline memory module

DDR = Double data rate

The bandwidths listed in these tables are per module. Most chipsets that support DDR also support dual-channel operation—a technique in which two matching DIMMs are installed to function as a single bank, with double the bandwidth of a single module. For example, if a chipset supports standard PC3200 modules, the bandwidth for a single module would be 3,200MBps. However, in dual-channel mode, the total bandwidth would double to 6,400MBps. Dual-channel operation optimizes PC design by ensuring that the CPU bus and memory bus both run at exactly the same speeds (meaning throughput, not MHz) so that data can move synchronously between the buses without delays.

DDR2 SDRAM

DDR2 is simply a faster version of DDR memory: It achieves higher throughput by using differential pairs of signal wires to allow faster signaling without noise and interference problems. DDR2 is still double data rate, just as with DDR, but the modified signaling method enables higher clock speeds to be achieved with more immunity to noise and cross-talk between the signals. The additional signals required for differential pairs add to the pin count—DDR2 DIMMs have 240 pins, which is more than the 184 pins of DDR. The original DDR specification officially topped out at 400MHz (although faster unofficial overclocked modules were produced), whereas DDR2 starts at 400MHz and goes up to an official maximum of 1,066MHz. Table 6.9 shows the various official JEDEC-approved DDR2 module types and bandwidth specifications.

Table 6.9 JEDEC Standard DDR2 Module (240-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)	Dual-Channel Transfer Rate (MBps)
PC2-3200	DDR2-400	200	2	400	8	3,200	6,400
PC2-4200	DDR2-533	266	2	533	8	4,266	8,533
PC2-5300	DDR2-667	333	2	667	8	5,333	10,667
PC2-6400	DDR2-800	400	2	800	8	6,400	12,800
PC2-8500	DDR2-1066	533	2	1066	8	8,533	17,066

MTps = Megatransfers per second

MBps = Megabytes per second

DIMM = Dual inline memory module

DDR = Double data rate

The fastest official JEDEC-approved standard is DDR2-1066, which are chips that run at an effective speed of 1,066MHz (really megatransfers per second), resulting in modules designated PC2-8500 having a bandwidth of 8,533MBps. However, just as with DDR, many of the module manufacturers produce even faster modules designed for overclocked systems. These are sold as modules with unofficial designations and performance figures that exceed the standard ratings. Table 6.10 shows the popular unofficial speed ratings I've seen on the market. Note that since the speeds of these modules are beyond the standard default motherboard and chipset speeds, you won't see any advantage to using these unless you are overclocking your system to match.

Table 6.10 Overclocked (non-JEDEC) DDR2 Module (240-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)	Dual-Channel Transfer Rate (MBps)
PC2-6000	DDR2-750	375	2	750	8	6,000	12,000
PC2-7200	DDR2-900	450	2	900	8	7,200	14,400
PC2-8000	DDR2-1000	500	2	1000	8	8,000	16,000
PC2-8800	DDR2-1100	550	2	1100	8	8,800	17,600
PC2-8888	DDR2-1111	556	2	1111	8	8,888	17,777
PC2-9136	DDR2-1142	571	2	1142	8	9,136	18,272
PC2-9200	DDR2-1150	575	2	1150	8	9,200	18,400
PC2-9600	DDR2-1200	600	2	1200	8	9,600	19,200
PC2-10000	DDR2-1250	625	2	1250	8	10,000	20,000

MTps = Megatransfers per second

MBps = Megabytes per second

DIMM = Dual inline memory module

DDR = Double data rate

In addition to providing greater speeds and bandwidth, DDR2 has other advantages. It uses lower voltage than conventional DDR (1.8V versus 2.5V), so power consumption and heat generation are reduced. Because of the greater number of pins required on DDR2 chips, the chips typically use fine-pitch ball grid array (FBGA) packaging rather than the thin small outline package (TSOP) chip packaging used by most DDR and conventional SDRAM chips. FPGA chips are connected to the substrate (meaning the memory module in most cases) via tightly spaced solder balls on the base of the chip.

DDR2 DIMMs resemble conventional DDR DIMMs but have more pins and slightly different notches to prevent confusion or improper application. For example, the different physical notches prevent you from plugging a DDR2 module into a conventional DDR (or SDR) socket. DDR2 memory module designs incorporate 240 pins, significantly more than conventional DDR or standard SDRAM DIMMs.

JEDEC began working on the DDR2 specification in April 1998, and published the standard in September 2003. DDR2 chip and module production actually began in mid-2003 (mainly samples and prototypes), and the first chipsets, motherboards, and systems supporting DDR2 appeared for Intel processor–based systems in mid-2004. At that time variations of DDR2 such as G-DDR2 (Graphics DDR2) began appearing in graphics cards as well. Mainstream motherboard chipset support for DDR2 on Intel processor–based systems appeared in 2005. Notable for its lack of DDR2 support through 2005 was AMD, whose Athlon 64 and Opteron processor families included integrated DDR memory controllers. AMD processor–based systems first supported DDR2 in mid-2006, with the release of socket AM2 motherboards and processors to match. (AMD's Socket F, also known as 1207 FX, also supports DDR2 memory.)

It is interesting to note that AMD was almost two years behind Intel in the transition from DDR to DDR2. This is because AMD included the memory controller in its Athlon 64 and newer processors, rather than incorporating the memory controller in the chipset North Bridge, as with more traditional Intel designs. Although there are advantages to integrating the memory controller in the CPU, a major disadvantage is the inability to quickly adopt new memory architectures, because doing so requires that both the processor and processor socket be redesigned. With the memory controller in the chipset, Intel can more quickly adopt newer and faster memory architectures without having to redesign existing processors. These transitional differences will likely become apparent again in the expected transition from DDR2 to DDR3 in 2008.

DDR3

DDR3 isis the latest JEDEC memory standard, which will enable higher levels of performance along with lower power consumption and higher reliability than DDR2. JEDEC began working on the DDR3 specification in June of 2002, and the first DDR3 memory modules and supporting chipsets (Intel 3xx series) were released for Intel-based systems in mid-2007. AMD is expected to introduce new processors and sockets supporting DDR3 memory sometime during 2008. Initially DDR3 will be more expensive than DDR2, and will be used in systems requiring extremely high-performance memory. DDR3 is expected to achieve mainstream status in the 2008 to 2009 timeframe.

DDR3 modules use advanced signal designs, including self-driver calibration and data synchronization, along with an optional onboard thermal sensor. DDR3 memory runs on only 1.5V, which is nearly 20% less than the 1.8V used by DDR2 memory. The lower voltage combined with higher efficiency is expected to reduce overall power consumption by up to 30% compared to DDR2.

DDR3 modules are expected to initially become popular for systems where the processor and/or memory bus runs at 1,333MHz, which is faster than the 1,066MHz maximum supported by DDR2. For higher-speed memory in standard (non-overclocked) systems, DDR3 modules rated PC3-10600 and PC3-12800 will allow for throughputs of 10,667MBps and 12,800MBps, respectively. When combined in dual-channel operation, a pair of PC3-12800 modules will result in a total throughput of an incredible 25,600MBps. Table 6.11 shows the various official JEDEC-approved DDR3 module types and bandwidth specifications.

Table 6.11 JEDEC Standard DDR3 Module (240-pin DIMM) Speeds and Transfer Rates

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)	Dual-Channel Transfer Rate (MBps)
PC3-6400	DDR3-800	400	2	800	8	6,400	12,800
PC3-8500	DDR3-1066	533	2	1066	8	8,533	17,066
PC3-10600	DDR3-1333	667	2	1333	8	10,667	21,333
PC3-12800	DDR3-1600	800	2	1600	8	12,800	25,600

MTps = Megatransfers per second

MBps = Megabytes per second

DIMM = Dual inline memory module

DDR = Double data rate

240-pin DDR3 modules are similar in pin count, size, and shape to DDR2 modules; however, DDR3 modules are incompatible with DDR2 circuits, and are designed with different keying to make them physically noninterchangeable.

RDRAM

Rambus DRAM (RDRAM) was a proprietary (non-JEDEC) memory technology found mainly in certain Intel-based Pentium III and 4 systems from 2000 through 2002. Intel had signed a contract with Rambus in 1996 ensuring it would both adopt and support RDRAM memory into 2001. Believing that any memory it endorsed would automatically become the most popular in the industry, Intel also invested heavily in Rambus at the time. Since RDRAM was a proprietary standard owned by Rambus, using or producing it would require licensing from Rambus, something that was not very popular with other memory and chipset manufacturers. Still, the technology was licensed and Intel originally promised that supporting chipsets and motherboards would be available in 1998.

Unfortunately there were problems in getting the supporting chipsets to market, with delays of many months resulting in memory manufacturers stockpiling RDRAM chips with no systems to support them, while conventional SDRAM and DDR meanwhile came into short supply. The delays resulted in an industry-wide debacle that caused Intel to rethink and eventually abandon its investment in the technology. After 2001, Intel continued to support RDRAM in existing systems; however, new chipsets and motherboards rapidly shifted to DDR SDRAM. AMD wisely never invested in the RDRAM technology, and as a result no AMD-based systems were ever designed to use RDRAM.

Although RDRAM standards had been proposed that would support faster processors through 2006, without Intel's commitment to future chipset development and support, very few RDRAM-based systems were sold after 2002. Due to the lack of industry support from chipset and motherboard manufacturers, RDRAM was only used in PCs for a short time, and will most likely not play a big part in any future PCs.

With RDRAM, Rambus developed what is essentially a chip-to-chip memory bus, with specialized devices that communicate at very high rates of speed. What might be interesting to some is that this technology was first developed for game systems and first made popular by the Nintendo 64 game system, and it subsequently was used in the Sony Playstation 2.

Conventional memory systems that use SDRAM are known as *wide-channel systems*. They have memory channels as wide as the processor's data bus, which for the Pentium and up is 64 bits, or even wider in dual-channel modes. The dual inline memory module (DIMM) is a 64-bit wide device, meaning data can be transferred to it 64 bits (or 8 bytes) at a time.

RDRAM modules, on the other hand, are narrow-channel devices. They transfer data only 16 bits (2 bytes) at a time (plus 2 optional parity bits), but at faster speeds. This was a shift away from a more parallel to a more serial design for memory and is similar to what has been happening with other evolving buses in the PC.

Each individual chip is serially connected to the next on a package called a *Rambus inline memory module (RIMM)*, which looks similar to a DIMM module but which is not interchangeable. All memory transfers are done between the memory controller and a single device, not between devices. A single Rambus channel typically has three RIMM sockets and can support up up to 32 individual RDRAM devices (the RDRAM chips) and more if buffers are used. However, most motherboards implement only two modules per channel (four sockets in a dual-channel design) to avoid problems with signal noise.

The RDRAM memory bus is a continuous path through each device and module on the bus, with each module having input and output pins on opposite ends. Therefore, any RIMM sockets not containing a RIMM must then be filled with a continuity module to ensure that the path is completed. The signals that reach the end of the bus are terminated on the motherboard.

16-bit single channel RIMMs originally ran at 800MHz, so the overall throughput is 800×2, or 1.6GB per second for a single channel—the same as PC1600 DDR SDRAM. Pentium 4 systems typically used two banks simultaneously, creating a dual-channel design capable of 3.2GBps, which matched the bus speed of the original Pentium 4 processors. The RDRAM design features less latency between transfers because they all run synchronously in a looped system and in only one direction.

Newer RIMM versions ran at 1,066MHz in addition to the original 800MHz rate, but very few chipsets or motherboards were released to support the higher speed.

Each RDRAM chip on a RIMM1600 essentially operates as a standalone device sitting on the 16-bit data channel. Internally, each RDRAM chip has a core that operates on a 128-bit wide bus split into eight 16-bit banks running at 100MHz. In other words, every 10ns (100MHz), each RDRAM chip can transfer 16 bytes to and from the core. This internally wide yet externally narrow high-speed interface is the key to RDRAM.

Other improvements to the design include separating control and data signals on the bus. Independent control and address buses are split into two groups of pins for row and column commands, while data is transferred across the 2-byte wide data bus. The actual memory bus clock runs at 400MHz; however, data is transferred on both the falling and rising edges of the clock signal, or twice per clock pulse. The falling edge is called an *even cycle*, and the rising edge is called an *odd cycle*. Complete memory bus synchronization is achieved by sending packets of data beginning on an even cycle interval. The overall wait before a memory transfer can begin (latency) is only one cycle, or 2.5ns maximum.

Figure 6.2 (shown earlier) depicts the relationship between clock and data cycles; you can see the DDR clock and data cycles used by RDRAM and DDR SDRAM. An RDRAM data packet always begins on an even (falling) transition for synchronization purposes.

The architecture also supports multiple, simultaneous interleaved transactions in multiple separate time domains. Therefore, before a transfer has even completed, another can begin.

Another important feature of RDRAM is that it is designed for low power consumption. The RIMMs themselves as well as the RDRAM devices run on only 2.5 volts and use low-voltage signal swings from 1.0V to 1.8V, a swing of only 0.8V total. RDRAMs also have four power-down modes and can automatically transition into standby mode at the end of a transaction, which offers further power savings.

A RIMM is similar in size and physical form to a DIMM, but they are not interchangeable. RIMMs are available in module sizes up to 1GB or more and can be added to a system one at a time because each individual RIMM technically represents multiple banks to a system. Note, however, that they have to be added in pairs if your motherboard implements dual-channel RDRAM and you are using 16-bit wide RIMMs.

RIMMs are available in four primary speed grades and usually run in a dual-channel environment, so they have to be installed in pairs, with each one of the pairs in a different set of sockets. Each set of RIMM sockets on such boards is a channel. The 32-bit version incorporates multiple channels within a single device and, as such, is designed to be installed individually, eliminating the requirement for matched pairs. Table 6.12 compares the various types of RDRAM modules. Note that the once-common names for RIMM modules, such as PC800, have been replaced by names that reflect the actual bandwidth of the modules to avoid confusion with DDR memory.

Table 6.12 RDRAM Module Types and Bandwidth

Module Standard	Chip Type	Clock Speed (MHz)	Cycles per Clock	Bus Speed (MTps)	Bus Width (Bytes)	Transfer Rate (MBps)
RIMM1200	PC600	300	2	600	2	1,200
RIMM1400	PC700	350	2	700	2	1,400
RIMM1600	PC800	400	2	800	2	1,600
RIMM2100	PC1066	533	2	1,066	2	2,133

MTps = Megatransfers per second

MBps = Megabytes per second

RIMM = Rambus inline memory module

When Intel initially threw its weight behind the Rambus memory, it seemed destined to be a sure thing for success. Unfortunately, technical delays in the chipsets caused the supporting motherboards to be significantly delayed, and with few systems to support the RIMMs, most memory manufacturers went back to making SDRAM or shifted to DDR SDRAM instead. This caused the remaining available RIMMs being manufactured to be originally priced three or more times that of a comparatively sized DIMM. More recently, the cost for RDRAM RIMMs has come down to approximately that of DDR SDRAM, but by the time that happened, Intel had shifted all future chipset development to support only DDR and DDR2 memory.

As I've stated many times, one of the main considerations for memory is that the throughput of the memory bus should match the throughput of the processor bus, and in that area RDRAM RIMMs were originally more suited to the initial Pentium 4 processor systems. However, with the increases in speed of the Pentium 4 processor bus along with the advent of chipsets supporting dual-channel DDR memory, DDR, DDR2, and DDR3 became the best choices for the CPU bus speeds of both Intel and AMD processors. In short, the advent of newer chipsets has rendered DDR memory types as the best choices for modern systems, offering the maximum memory performance possible.

Note

Unfortunately for the memory chip manufacturers, Rambus has claimed patents that cover both standard and DDR SDRAM designs. So, regardless of whether these companies manufacture SDRAM, DDR, or RDRAM, it is the contention of Rambus that these memory manufacturers must pay the company royalties. Several court cases are ongoing with companies challenging these patents, and a lot is riding on the outcome. Most of the cases that have gone to trial have so far ruled against Rambus, essentially invalidating its patents and claims on DDR and SDRAM. Many appeals are pending, and it will likely be a long time before the patent issues are resolved.

With support for RDRAM memory essentially gone in 2003, RDRAM quickly disappeared from the PC marketplace. Because RDRAM is in such limited supply, if you have existing systems with RDRAM memory, it is generally not cost effective to upgrade them by adding more memory.

Memory Modules

The CPU and motherboard architecture (chipset) dictates a particular computer's physical memory capacity and the types and forms of memory that can be installed. Over the years, two main changes have occurred in computer memory—it has gradually become faster and wider. The CPU and the memory controller circuitry indicate the speed and width requirements. The memory controller in a modern PC resides in the motherboard chipset. Even though a system might physically support a given amount of memory, the type of software you run could dictate whether all the memory can be used.

The 8088 and 8086 CPUs, with 20 address lines, can use as much as 1MB (1,024KB) of RAM. The 286 and 386SX CPUs have 24 address lines and can keep track of as much as 16MB of memory. The 386DX, 486, Pentium, and Pentium-MMX CPUs have a full set of 32 address lines, so they can keep track of 4GB of memory; the Pentium Pro, Pentium II/III, and 4, as well as the AMD Athlon and Duron, have 36 address lines and can manage an impressive 64GB. The Itanium processor, on the other hand, has 44-bit addressing, which allows for up to 16TB (terabytes) of physical RAM!

◀◀ See "Processor Specifications," p. 43.

When the 286 and higher chips emulate the 8088 chip (as they do when running 16-bit software, such as DOS or Windows 3.x), they implement a hardware operating mode called *real mode*. Real mode is the only mode available on the 8086 and 8088 chips used in PC and XT systems. In real mode, all Intel processors—even the mighty Pentium family—are restricted to using only 1MB of memory, just as their 8086 and 8088 ancestors were, and the system design reserves 384KB of that amount. Only in protected mode can the 286 or better chips use their maximum potentials for memory addressing.

◀◀ See "Processor Modes," p. 50.

P5 class systems can address as much as 4GB of memory, and P6/P7 class systems can address up to 64GB. To put these memory-addressing capabilities into perspective, 64GB (65,536MB) of memory would cost more than $10,000! Even if you could afford all this memory, some of the largest memory modules available for desktop PCs today are 1GB DIMMs. Installing 64GB of RAM would require 64 1GB DIMMs, and most systems today support up to only four DIMM sockets.

Although memory sizes are increasing and some current desktop motherboards support 2GB modules, the real limitations on memory sizing in any system are the chipset and the number of sockets on the motherboard. Most desktop motherboards incorporate from two to four memory sockets, which allows a maximum of 4GB–8GB if all the sockets are filled. These limitations are from the chipset, not the processor or RAM modules. Some processors can address 64GB, but no chipset on the market will allow that!

Note

See the "Chipsets" section in Chapter 4 for the maximum cacheable limits on all the Intel and other motherboard chipsets.

SIMMs, DIMMs, and RIMMs

Originally, systems had memory installed via individual chips. They are often referred to as dual inline package (DIP) chips because of their designs. The original IBM XT and AT had 36 sockets on the motherboard for these individual chips; then more of them were installed on the memory cards plugged into the bus slots. I remember spending hours populating boards with these chips, which was a tedious job.

Besides being a time-consuming and labor-intensive way to deal with memory, DIP chips had one notorious problem—they crept out of their sockets over time as the system went through thermal cycles. Every day, when you powered the system on and off, the system heated and cooled, and the chips gradually walked their way out of the sockets—a phenomenon called *chip creep*. Eventually, good contact was lost and memory errors resulted. Fortunately, reseating all the chips back in their sockets usually rectified the problem, but that method was labor intensive if you had a lot of systems to support.

The alternative to this at the time was to have the memory soldered into either the motherboard or an expansion card. This prevented the chips from creeping and made the connections more permanent, but it caused another problem. If a chip did go bad, you had to attempt desoldering the old one and resoldering a new one or resort to scrapping the motherboard or memory card on which the chip was installed. This was expensive and made memory troubleshooting difficult.

A chip was needed that was both soldered and removable, and that is exactly what was found in the module called a SIMM. For memory storage, most modern systems have adopted the single inline memory module (SIMM) or the more recent DIMM and RIMM module designs as an alternative to individual memory chips. These small boards plug into special connectors on a motherboard or memory card. The individual memory chips are soldered to the module, so removing and replacing them is impossible. Instead, you must replace the entire module if any part of it fails. The module is treated as though it were one large memory chip.

Two main types of SIMMs, three main types of DIMMs, and one type of RIMM have been commonly used in desktop systems. The various types are often described by their pin count, memory row width, or memory type.

SIMMs, for example, are available in two main physical types—30-pin (8 bits plus an option for 1 additional parity bit) and 72-pin (32 bits plus an option for 4 additional parity bits)—with various capacities and other specifications. The 30-pin SIMMs are physically smaller than the 72-pin versions, and either version can have chips on one or both sides. SIMMs were widely used from the late 1980s to the late 1990s but have become obsolete.

DIMMs are also available in three main types. DIMMs usually hold standard SDRAM or DDR SDRAM chips and are distinguished by different physical characteristics. Standard DIMMs have 168 pins, one notch on either side, and two notches along the contact area. DDR DIMMs, on the other hand, have 184 pins, two notches on each side, and only one offset notch along the contact area. DDR2 DIMMs have 240 pins, two notches on each side, and one in the center of the contact area. All DIMMs are either 64-bits (non-ECC/parity) or 72-bits (parity or error-correcting code [ECC]) wide (data paths). The main physical difference between SIMMs and DIMMs is that DIMMs have different signal pins on each side of the module. That is why they are called dual inline memory modules, and why with only 1" of additional length, they have many more pins than a SIMM.

Note

There is confusion among users and even in the industry regarding the terms *single-sided* and *double-sided* with respect to memory modules. In truth, the single- or double-sided designation actually has nothing to do with whether chips are physically located on one or both sides of the module, and it has nothing to do with whether the module is a SIMM or DIMM (meaning whether the connection pins are single- or double-inline). Instead the terms single-sided and double-sided are used to indicate whether the module has one or two banks of memory chips installed. A double-banked DIMM module has two complete 64-bit wide banks of chips logically stacked so that the module is twice as deep (has twice as many 64-bit rows). In most (but not all) cases, this requires chips to be on both sides of the module; therefore, the term double-sided has often been used to indicate that a module has two banks, even though the term is technically incorrect. Single-banked modules (incorrectly referred to as single-sided) can have chips physically mounted on both sides of the module, and double-banked modules (incorrectly referred to as double-sided) can have chips physically mounted on only one side. I recommend using the terms *single-banked* and *double-banked* instead because they are much more accurate and easily understood.

RIMMs also have different signal pins on each side. Three different physical types of RIMMs are available: a 16/18-bit version with 184 pins, a 32/36-bit version with 232 pins, and a 64/72-bit version with 326 pins. Each of these plugs into the same sized connector, but the notches in the connectors and RIMMs are different to prevent a mismatch. A given board will accept only one type. By far the most common type is the 16/18-bit version. The 32-bit version was introduced in late 2002, and the 64-bit version was introduced in 2004.

The standard 16/18-bit RIMM has 184 pins, one notch on either side, and two notches centrally located in the contact area. The 16-bit versions are used for non-ECC applications, whereas the 18-bit versions incorporate the additional bits necessary for ECC.

Figures 6.3 through 6.9 show a typical 30-pin (8-bit) SIMM, 72-pin (32-bit) SIMM, 168-pin SDRAM DIMM, 184-pin DDR SDRAM (64-bit) DIMM, 240-pin DDR2 DIMM, 240-pin DDR3 DIMM, and 184-pin RIMM, respectively. The pins are numbered from left to right and are connected through to both sides of the module on the SIMMs. The pins on the DIMM are different on each side, but on a SIMM, each side is the same as the other and the connections carry through. Note that all dimensions are in both inches and millimeters (in parentheses), and modules are generally available in error-correcting code (ECC) versions with 1 extra ECC (or parity) bit for every 8 data bits (multiples of 9 in data width) or versions that do not include ECC support (multiples of 8 in data width).

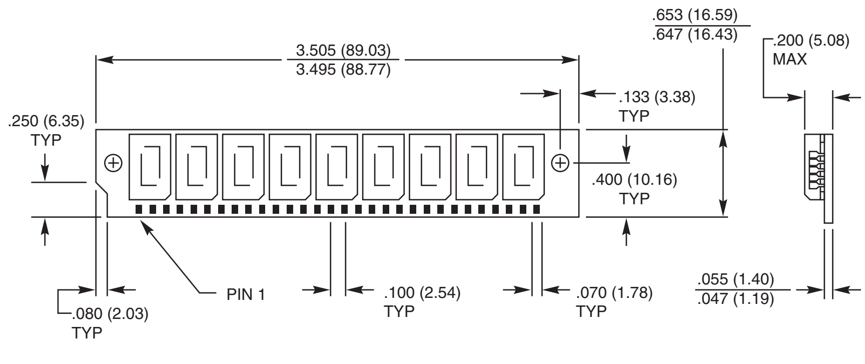


Figure 6.3 A typical 30-pin SIMM.

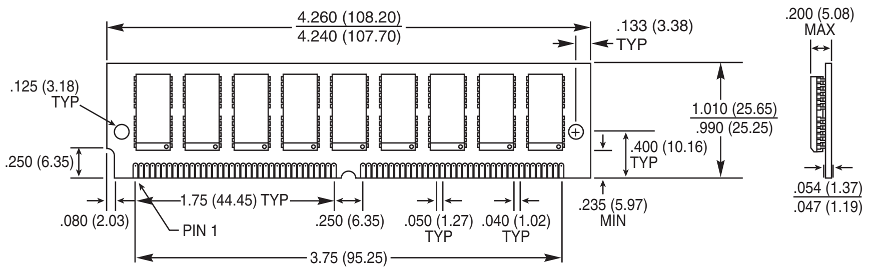


Figure 6.4 A typical 72-pin SIMM.

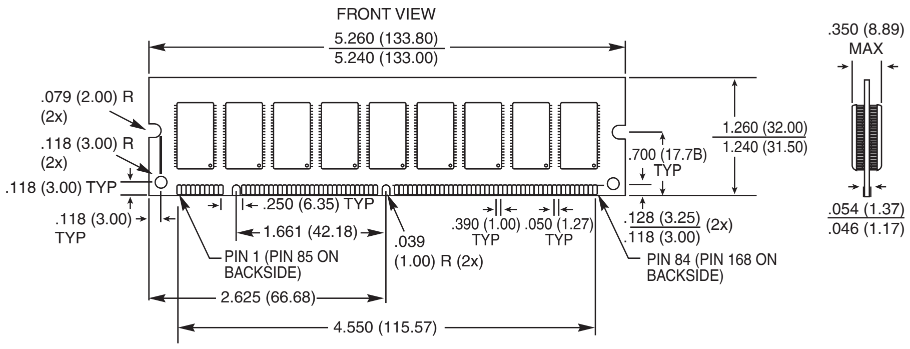


Figure 6.5 A typical 168-pin SDRAM DIMM.

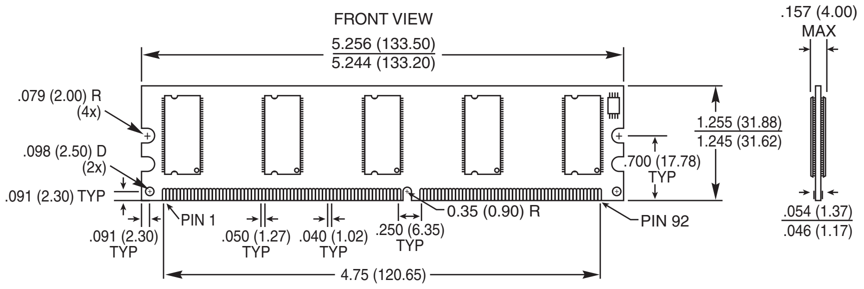


Figure 6.6 A typical 184-pin DDR DIMM.

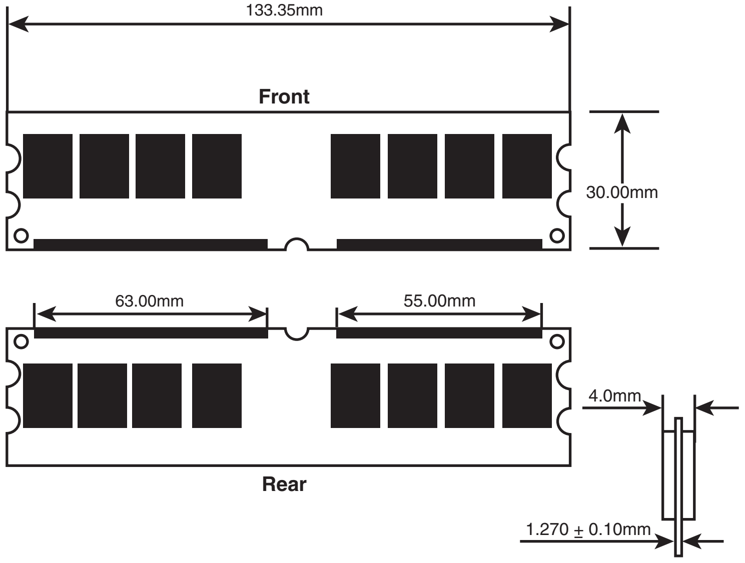


Figure 6.7 A typical 240-pin DDR2 DIMM.

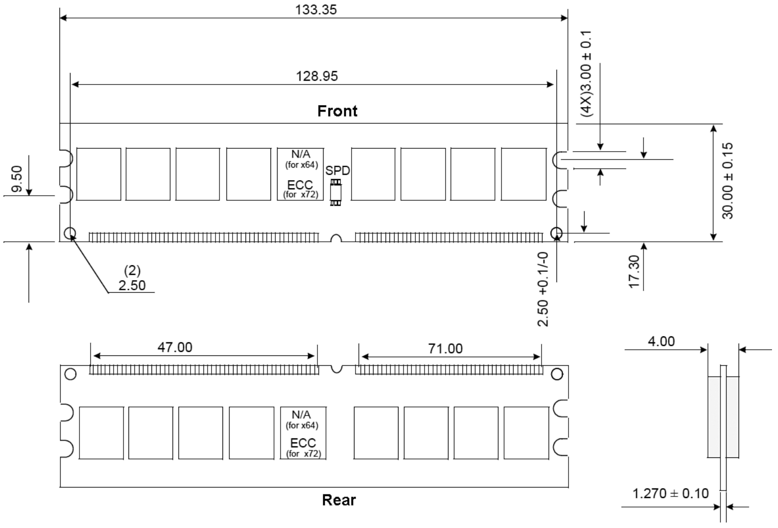


Figure 6.8 A typical 240-pin DDR3 DIMM.

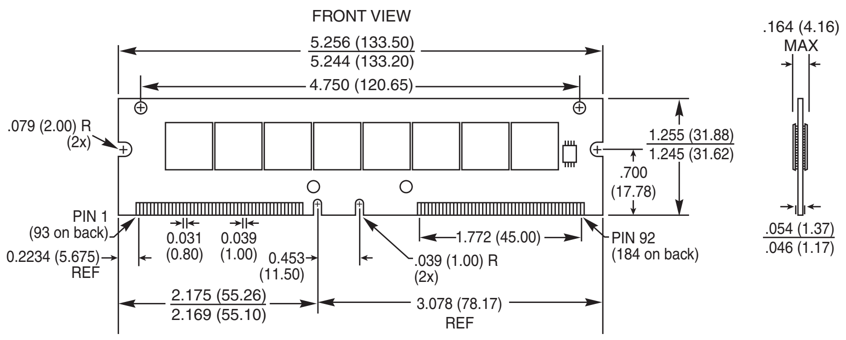


Figure 6.9 A typical 184-pin RIMM.

All these memory modules are fairly compact considering the amount of memory they hold and are available in several capacities and speeds. Table 6.13 lists the various capacities available for SIMMs, DIMMs, and RIMMs.

Table 6.13 SIMM, DIMM, and RIMM Capacities

Capacity	Standard	Parity/ECC
	30-Pin SIMM	
256KB	256KB×8	256KB×9
1MB	1MB×8	1MB×9
4MB	4MB×8	4MB×9
16MB	16MB×8	16MB×9
	72-Pin SIMM	
1MB	256KB×32	256KB×36
2MB	512KB×32	512KB×36
4MB	1MB×32	1MB×36
8MB	2MB×32	2MB×36
16MB	4MB×32	4MB×36
32MB	8MB×32	8MB×36
64MB	16MB×32	16MB×36
128MB	32MB×32	32MB×36
	168/184-Pin DIMM/DDR DIMM	
8MB	1MB×64	1MB×72
16MB	2MB×64	2MB×72
32MB	4MB×64	4MB×72
64MB	8MB×64	8MB×72
128MB	16MB×64	16MB×72
256MB	32MB×64	32MB×72
512MB	64MB×64	64MB×72
1,024MB	128MB×64	128MB×72

Table 6.13 Continued

Capacity	Standard	Parity/ECC
2,048MB	256MB×64	256MB×72
	240-Pin DDR2 DIMM	
256MB	32MB×64	32MB×72
512MB	64MB×64	64MB×72
1,024MB	128MB×64	128MB×72
2,048MB	256MB×64	256MB×72
	240-Pin DDR3 DIMM	
256MB	32MB×64	32MB×72
512MB	64MB×64	64MB×72
1,024MB	128MB×64	128MB×72
2,048MB	256MB×64	256MB×72
	184-Pin RIMM	
64MB	32MB×16	32MB×18
128MB	64MB×16	64MB×18
256MB	128MB×16	128MB×18
512MB	256MB×16	256MB×18
1,024MB	512MB×16	512MB×18

Memory modules of each type and capacity are available in various speed ratings. Consult your motherboard documentation for the correct memory speed and type for your system. It is usually best for the memory speed (also called throughput or bandwidth) to match the speed of the processor data bus (also called the front side bus, or FSB).

If a system requires a specific speed memory module, you can almost always substitute faster speeds if the one specified is not available. Generally, no problems occur in mixing module speeds, as long as you use modules equal to or faster than what the system requires. Because there's little price difference between the various speed versions, I often buy faster modules than are necessary for a particular application. This might make them more usable in a future system that could require the faster speed.

Because SDRAM and newer modules have an onboard serial presence detect (SPD) ROM that reports their speed and timing parameters to the system, most systems run the memory controller and memory bus at the speed matching the slowest module installed.

Note

A bank is the smallest amount of memory needed to form a single row of memory addressable by the processor. It is the minimum amount of physical memory that is read or written by the processor at one time and usually corresponds to the data bus width of the processor. If a processor has a 64-bit data bus, a bank of memory also is 64 bits wide. If the memory is interleaved or runs dual-channel, a virtual bank is formed that is twice the absolute data bus width of the processor.

You can't always replace a module with a higher-capacity unit and expect it to work. Systems might have specific design limitations for the maximum capacity of module they can take. A larger-capacity module works only if the motherboard is designed to accept it in the first place. Consult your system documentation to determine the correct capacity and speed to use.

Registered Modules

SDRAM through DDR3 modules are available in unbuffered and registered versions. Most PC motherboards are designed to use unbuffered modules, which allow the memory controller signals to pass directly to the memory chips on the module with no interference. This is not only the cheapest design, but also the fastest and most efficient. The only drawback is that the motherboard designer must place limits on how many modules (meaning module sockets) can be installed on the board, and possibly also limit how many chips can be on a module. So-called double-sided modules that really have multiple banks of chips onboard might be restricted on some systems in certain combinations.

Systems designed to accept extremely large amounts of RAM (such as servers) often require registered modules. A registered module uses an architecture that has register chips on the module that act as an interface between the actual RAM chips and the chipset. The registers temporarily hold data passing to and from the memory chips and enable many more RAM chips to be driven or otherwise placed on the module than the chipset could normally support. This allows for motherboard designs that can support many modules and enables each module to have a larger number of chips. In general, registered modules are required by server or workstation motherboards designed to support more than four sockets. One anomaly is the initial version of the AMD Athlon 64 FX processor, which also uses registered memory because its Socket 940 design was based on the AMD Opteron workstation and server processor. Subsequent Socket 939, AM2, and Socket F versions of the Athlon FX no longer require registered memory.

To provide the space needed for the buffer chips, a registered DIMM is often taller than a standard DIMM. Figure 6.10 compares a typical registered DIMM to a typical unbuffered DIMM.

Tip

If you are installing registered DIMMs in a slimline case, clearance between the top of the DIMM and the case might be a problem. Some vendors sell low-profile registered DIMMs that are about the same height as an unbuffered DIMM. Use this type of DIMM if your system does not have enough head room for standard registered DIMMs. Some vendors sell only this type of DIMM for particular systems.

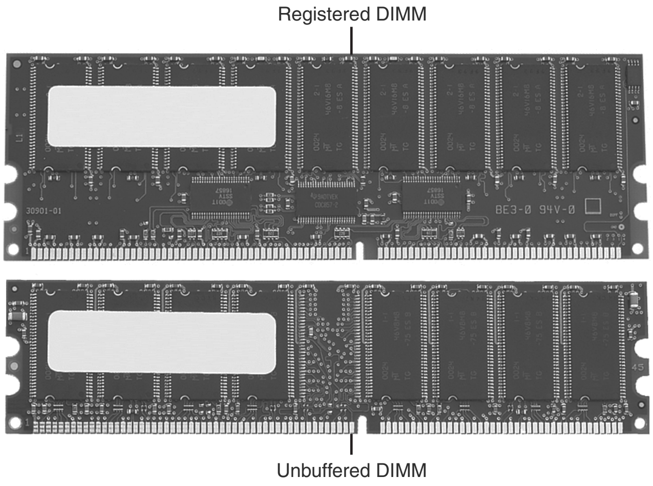


Figure 6.10 A typical registered DIMM is taller than a typical unbuffered DIMM to provide room for buffer chips.

The important thing to note is that you can use only the type of module your motherboard (or chipset) is designed to support. For most, that is standard unbuffered modules or, in some cases, registered modules.

SIMM Pinouts

Table 6.14 shows the interface connector pinouts for standard 72-pin SIMMs. They also include a special presence detect table that shows the configuration of the presence detect pins on various 72-pin SIMMs. The motherboard uses the presence detect pins to determine exactly what size and speed SIMM is installed. Industry-standard 30-pin SIMMs do not have a presence detect feature, but IBM did add this capability to its modified 30-pin configuration. Note that all SIMMs have the same pins on both sides of the module.

Table 6.14 Standard 72-Pin SIMM Pinout

Pin	SIMM Signal Name	Pin	SIMM Signal Name	Pin	SIMM Signal Name	Pin	SIMM Signal Name
1	Ground						
2	Data Bit 0	21	Data Bit 20	40	Column Address Strobe 0	56	Data Bit 27
3	Data Bit 16	22	Data Bit 5			57	Data Bit 12
4	Data Bit 1	23	Data Bit 21	41	Column Address Strobe 2	58	Data Bit 28
5	Data Bit 17	24	Data Bit 6	42	Column Address Strobe 3	59	+5 Vdc
6	Data Bit 2	25	Data Bit 22			60	Data Bit 29
7	Data Bit 18	26	Data Bit 7	43	Column Address Strobe 1	61	Data Bit 13
8	Data Bit 3	27	Data Bit 23			62	Data Bit 30
9	Data Bit 19	28	Address Bit 7	44	Row Address Strobe 0	63	Data Bit 14
10	+5 Vdc	29	Address Bit 11	45	Row Address Strobe 1	64	Data Bit 31
11	Presence Detect 5	30	+5 Vdc	46	Reserved	65	Data Bit 15
12	Address Bit 0	31	Address Bit 8	47	Write Enable	66	EDO
13	Address Bit 1	32	Address Bit 9	48	ECC Optimized	67	Presence Detect 1
14	Address Bit 2	33	Address Bit 12	49	Data Bit 8	68	Presence Detect 2
15	Address Bit 3	34	Address Bit 13	50	Data Bit 24	69	Presence Detect 3
16	Address Bit 4	35	Parity Data Bit 2	51	Data Bit 9	70	Presence Detect 4
17	Address Bit 5	36	Parity Data Bit 0	52	Data Bit 25	71	Reserved
18	Address Bit 6	37	Parity Data Bit 1	53	Data Bit 10	72	Ground
19	Address Bit 10	38	Parity Data Bit 3	54	Data Bit 26		
20	Data Bit 4	39	Ground	55	Data Bit 11		

Notice that the 72-pin SIMMs use a set of four or five pins to indicate the type of SIMM to the motherboard. These presence detect pins are either grounded or not connected to indicate the type of SIMM to the motherboard. Presence detect outputs must be tied to the ground through a 0-ohm resistor or jumper on the SIMM—to generate a high logic level when the pin is open or a low logic level when the motherboard grounds the pin. This produces signals the memory interface logic can decode. If the motherboard uses presence detect signals, a power-on self test (POST) procedure can determine the size and speed of the installed SIMMs and adjust control and addressing signals automatically. This enables autodetection of the memory size and speed.

Note

In many ways, the presence detect pin function is similar to the industry-standard DX coding used on modern 35mm film rolls to indicate the ASA (speed) rating of the film to the camera. When you drop the film into the camera, electrical contacts can read the film's speed rating via an industry-standard configuration.

Presence detect performs the same function for 72-pin SIMMs that the serial presence detect (SPD) chip does for DIMMs.

Table 6.15 shows the Joint Electronic Devices Engineering Council (JEDEC) industry-standard presence detect configuration listing for the 72-pin SIMM family. JEDEC is an organization of U.S. semiconductor manufacturers and users that sets semiconductor standards.

Table 6.15 Presence Detect Pin Configurations for 72-Pin SIMMs

Size	Speed	Pin 67	Pin 68	Pin 69	Pin 70	Pin 11
1MB	100ns	Gnd	—	Gnd	Gnd	—
1MB	80ns	Gnd	—	—	Gnd	—
1MB	70ns	Gnd	—	Gnd	—	—
1MB	60ns	Gnd	—	—	—	—
2MB	100ns	—	Gnd	Gnd	Gnd	—
2MB	80ns	—	Gnd	—	Gnd	—
2MB	70ns	—	Gnd	Gnd	—	—
2MB	60ns	—	Gnd	—	—	—
4MB	100ns	Gnd	Gnd	Gnd	Gnd	—
4MB	80ns	Gnd	Gnd	—	Gnd	—
4MB	70ns	Gnd	Gnd	Gnd	—	—
4MB	60ns	Gnd	Gnd	—	—	—
8MB	100ns	—	—	Gnd	Gnd	—
8MB	80ns	—	—	—	Gnd	—
8MB	70ns	—	—	Gnd	—	—
8MB	60ns	—	—	—	—	—
16MB	80ns	Gnd	—	—	Gnd	Gnd
16MB	70ns	Gnd	—	Gnd	—	Gnd
16MB	60ns	Gnd	—	—	—	Gnd
16MB	50ns	Gnd	—	Gnd	Gnd	Gnd
32MB	80ns	—	Gnd	—	Gnd	Gnd
32MB	70ns	—	Gnd	Gnd	—	Gnd
32MB	60ns	—	Gnd	—	—	Gnd
32MB	50ns	—	Gnd	Gnd	Gnd	Gnd

— = No connection (open)
Gnd = Ground
Pin 67 = Presence detect 1
Pin 68 = Presence detect 2
Pin 69 = Presence detect 3
Pin 70 = Presence detect 4
Pin 11 = Presence detect 5

Unfortunately, unlike the film industry, not everybody in the computer industry follows established standards. As such, presence detect signaling is not a standard throughout the PC industry. Different system manufacturers sometimes use different configurations for what is expected on these four pins. Compaq, IBM (mainly PS/2 systems), and Hewlett-Packard are notorious for this type of behavior. Many of the systems from these vendors require special SIMMs that are basically the same as standard 72-pin SIMMs, except for special presence detect requirements. Table 6.16 shows how IBM defines these pins.

Table 6.16 Presence Detect Pins for IBM 72-Pin SIMMs

67	68	69	70	SIMM Type	IBM Part Number
—	—	—	—	Not a valid SIMM	n/a
Gnd	—	—	—	1MB 120ns	n/a
—	Gnd	—	—	2MB 120ns	n/a
Gnd	Gnd	—	—	2MB 70ns	92F0102
—	—	Gnd	—	8MB 70ns	64F3606
Gnd	—	Gnd	—	Reserved	n/a
—	Gnd	Gnd	—	2MB 80ns	92F0103
Gnd	Gnd	Gnd	—	8MB 80ns	64F3607
—	—	—	Gnd	Reserved	n/a
Gnd	—	—	Gnd	1MB 85ns	90X8624
—	Gnd	—	Gnd	2MB 85ns	92F0104
Gnd	Gnd	—	Gnd	4MB 70ns	92F0105
—	—	Gnd	Gnd	4MB 85ns	79F1003 (square notch) L40-SX
Gnd	—	Gnd	Gnd	1MB 100ns	n/a
Gnd	—	Gnd	Gnd	8MB 80ns	79F1004 (square notch) L40-SX
—	Gnd	Gnd	Gnd	2MB 100ns	n/a
Gnd	Gnd	Gnd	Gnd	4MB 80ns	87F9980
Gnd	Gnd	Gnd	Gnd	2MB 85ns	79F1003 (square notch) L40SX

— = No connection (open)

Gnd = Ground

Pin 67 = Presence detect 1

Pin 68 = Presence detect 2

Pin 69 = Presence detect 3

Pin 70 = Presence detect 4

Because these pins can have custom variations, you often must specify IBM, Compaq, HP, or generic SIMMs when you order memory for systems using 72-pin SIMMs. Although very few (if any) of these systems are still in service, keep this information in mind if you are moving 72-pin modules from one system to another or are installing salvaged memory into a system. Also, be sure you match the metal used on the module connectors and sockets. SIMM pins can be tin or gold plated, and the plating on the module pins must match that on the socket pins; otherwise, corrosion will result.

Caution

To have the most reliable system when using SIMM modules, you must install modules with gold-plated contacts into gold-plated sockets and modules with tin-plated contacts into tin-plated sockets only. If you mix gold contacts with tin sockets, or vice versa, you are likely to experience memory failures from 6 months to 1 year after initial installation because a type of corrosion known as *fretting* will take place. This has been a major problem with 72-pin SIMM-based systems because some memory and motherboard vendors opted for tin sockets and connectors while others opted for gold. According to connector manufacturer AMP's "Golden Rules: Guidelines for the Use of Gold on Connector Contacts" (available at http://www.tycoelectronics.com/documentation/whitepapers/pdf/aurulrep.pdf) and "The Tin Commandments: Guidelines for the Use of Tin on Connector Contacts" (available at http://www.tycoelectronics.com/documentation/whitepapers/pdf/sncomrep.pdf), you should match connector metals.

If you are maintaining systems with mixed tin/gold contacts in which fretting has already occurred, use a wet contact cleaner. After cleaning, to improve electrical contacts and help prevent corrosion, you should use a liquid contact enhancer and lubricant called Stabilant 22 from D.W. Electrochemicals when installing SIMMs or DIMMs. The company's website (http://www.stabilant.com) has detailed application notes on this subject that provide more technical details.

DIMM Pinouts

Table 6.17 shows the pinout configuration of a 168-pin standard unbuffered SDRAM DIMM. Note again that the pins on each side of the DIMM are different. All pins should be gold plated.

Table 6.17 168-Pin SDRAM DIMM Pinouts

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	GND	43	GND	85	GND	127	GND
2	Data Bit 0	44	Do Not Use	86	Data Bit 32	128	Clock Enable 0
3	Data Bit 1	45	Chip Select 2#	87	Data Bit 33	129	Chip Select 3#
4	Data Bit 2	46	I/O Mask 2	88	Data Bit 34	130	I/O Mask 6
5	Data Bit 3	47	I/O Mask 3	89	Data Bit 35	131	I/O Mask 7
6	+3.3V	48	Do Not Use	90	+3.3V	132	Reserved
7	Data Bit 4	49	+3.3V	91	Data Bit 36	133	+3.3V
8	Data Bit 5	50	NC	92	Data Bit 37	134	NC
9	Data Bit 6	51	NC	93	Data Bit 38	135	NC
10	Data Bit 7	52	Parity Bit 2	94	Data Bit 39	136	Parity Bit 6
11	Data Bit 8	53	Parity Bit 3	95	Data Bit 40	137	Parity Bit 7
12	GND	54	GND	96	GND	138	GND
13	Data Bit 9	55	Data Bit 16	97	Data Bit 41	139	Data Bit 48
14	Data Bit 10	56	Data Bit 17	98	Data Bit 42	140	Data Bit 49
15	Data Bit 11	57	Data Bit 18	99	Data Bit 43	141	Data Bit 50
16	Data Bit 12	58	Data Bit 19	100	Data Bit 44	142	Data Bit 51
17	Data Bit 13	59	+3.3V	101	Data Bit 45	143	+3.3V
18	+3.3V	60	Data Bit 20	102	+3.3V	144	Data Bit 52
19	Data Bit 14	61	NC	103	Data Bit 46	145	NC
20	Data Bit 15	62	NC	104	Data Bit 47	146	NC

Table 6.17 Continued

=Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
21	Parity Bit 0	63	Clock Enable 1	105	Parity Bit 4	147	NC
22	Parity Bit 1	64	GND	106	Parity Bit 5	148	GND
23	GND	65	Data Bit 21	107	GND	149	Data Bit 53
24	NC	66	Data Bit 22	108	NC	150	Data Bit 54
25	NC	67	Data Bit 23	109	NC	151	Data Bit 55
26	+3.3V	68	GND	110	+3.3V	152	GND
27	WE#	69	Data Bit 24	111	CAS#	153	Data Bit 56
28	I/O Mask 0	70	Data Bit 25	112	I/O Mask 4	154	Data Bit 57
29	I/O Mask 1	71	Data Bit 26	113	I/O Mask 5	155	Data Bit 58
30	Chip Select 0#	72	Data Bit 27	114	Chip Select 1#	156	Data Bit 59
31	Do Not Use	73	+3.3V	115	RAS#	157	+3.3V
32	GND	74	Data Bit 28	116	GND	158	Data Bit 60
33	Address Bit 0	75	Data Bit 29	117	Address Bit 1	159	Data Bit 61
34	Address Bit 2	76	Data Bit 30	118	Address Bit 3	160	Data Bit 62
35	Address Bit 4	77	Data Bit 31	119	Address Bit 5	161	Data Bit 63
36	Address Bit 6	78	GND	120	Address Bit 7	162	GND
37	Address Bit 8	79	Clock 2	121	Address Bit 9	163	Clock 3
38	Address Bit 10	80	NC	122	Bank Address 0	164	NC
39	Bank Address 1	81	SPD Write Protect	123	Address Bit 11	165	SPD Address 0
40	+3.3V	82	SPD Data	124	+3.3V	166	SPD Address 1
41	+3.3V	83	SPD Clock	125	Clock 1	167	SPD Address 2
42	Clock 0	84	+3.3V	126	Reserved	168	+3.3V

Gnd = Ground
SPD = Serial presence detect
NC = No connection

The DIMM uses a completely different type of presence detect than a SIMM, called *serial presence detect (SPD)*. It consists of a small EEPROM or flash memory chip on the DIMM that contains specially formatted data indicating the DIMM's features. This serial data can be read via the serial data pins on the DIMM, and it enables the motherboard to autoconfigure to the exact type of DIMM installed.

DIMMs can come in several varieties, including unbuffered and buffered as well as 3.3V and 5V. Buffered DIMMs have additional buffer chips on them to interface to the motherboard. Unfortunately, these buffer chips slow down the DIMM and are not effective at higher speeds. For this reason, most PC systems (those that do not use registered DIMMs) use unbuffered DIMMs. The voltage is simple—DIMM designs for PCs are almost universally 3.3V. If you install a 5V DIMM in a 3.3V socket, it would be damaged, but fortunately keying in the socket and on the DIMM prevents that.

Modern PC systems use only unbuffered 3.3V DIMMs. Apple and other non-PC systems can use the buffered 5V versions. Fortunately, the key notches along the connector edge of a DIMM are spaced differently for buffered/unbuffered and 3.3V/5V DIMMs, as shown in Figure 6.11. This prevents inserting a DIMM of the wrong type into a given socket.

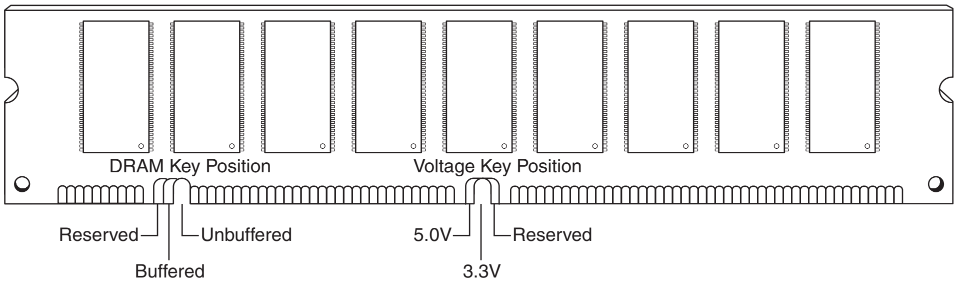


Figure 6.11 168-pin DRAM DIMM notch key definitions.

DDR DIMM Pinouts

Table 6.18 shows the pinout configuration of a 184-pin DDR SDRAM DIMM. Note again that the pins on each side of the DIMM are different. All pins should be gold plated.

Table 6.18 184-Pin DDR DIMM Pinouts

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	Reference +1.25V	47	Data Strobe 8	93	GND	139	GND
2	Data Bit 0	48	Address Bit 0	94	Data Bit 4	140	Data Strobe 17
3	GND	49	Parity Bit 2	95	Data Bit 5	141	Address Bit 10
4	Data Bit 1	50	GND	96	I/O +2.5V	142	Parity Bit 6
5	Data Strobe 0	51	Parity Bit 3	97	Data Strobe 9	143	I/O +2.5V
6	Data Bit 2	52	Bank Address 1	98	Data Bit 6	144	Parity Bit 7
7	+2.5 V	53	Data Bit 32	99	Data Bit 7	145	GND
8	Data Bit 3	54	I/O +2.5 V	100	GND	146	Data Bit 36
9	NC	55	Data Bit 33	101	NC	147	Data Bit 37
10	NC	56	Data Strobe 4	102	NC	148	+2.5V
11	GND	57	Data Bit 34	103	Address Bit 13	149	Data Strobe 13
12	Data Bit 8	58	GND	104	I/O +2.5V	150	Data Bit 38
13	Data Bit 9	59	Bank Address 0	105	Data Bit 12	151	Data Bit 39
14	Data Strobe 1	60	Data Bit 35	106	Data Bit 13	152	GND
15	I/O +2.5V	61	Data Bit 40	107	Data Strobe 10	153	Data Bit 44
16	Clock 1	62	I/O +2.5V	108	+2.5V	154	RAS#
17	Clock 1#	63	WE#	109	Data Bit 14	155	Data Bit 45
18	GND	64	Data Bit 41	110	Data Bit 15	156	I/O +2.5V
19	Data Bit 10	65	CAS#	111	Clock Enable 1	157	S0#
20	Data Bit 11	66	GND	112	I/O +2.5V	158	S1#
21	Clock Enable 0	67	Data Strobe 5	113	Bank Address 2	159	Data Strobe 14
22	I/O +2.5V	68	Data Bit 42	114	Data Bit 20	160	GND
23	Data Bit 16	69	Data Bit 43	115	Address Bit 12	161	Data Bit 46
24	Data Bit 17	70	+2.5V	116	GND	162	Data Bit 47
25	Data Strobe 2	71	S2#	117	Data Bit 21	163	S3#
26	GND	72	Data Bit 48	118	Address Bit 11	164	I/O +2.5V

Table 6.18 Continued

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
27	Address Bit 9	73	Data Bit 49	119	Data Strobe 11	165	Data Bit 52
28	Data Bit 18	74	GND	120	+2.5V	166	Data Bit 53
29	Address Bit 7	75	Clock 2#	121	Data Bit 22	167	FETEN
30	I/O +2.5V	76	Clock 2	122	Address Bit 8	168	+2.5V
31	Data Bit 19	77	I/O +2.5V	123	Data Bit 23	169	Data Strobe 15
32	Address Bit 5	78	Data Strobe 6	124	GND	170	Data Bit 54
33	Data Bit 24	79	Data Bit 50	125	Address Bit 6	171	Data Bit 55
34	GND	80	Data Bit 51	126	Data Bit 28	172	I/O +2.5V
35	Data Bit 25	81	GND	127	Data Bit 29	173	NC
36	Data Strobe 3	82	+2.5VID	128	I/O +2.5V	174	Data Bit 60
37	Address Bit 4	83	Data Bit 56	129	Data Strobe 12	175	Data Bit 61
38	+2.5V	84	Data Bit 57	130	Address Bit 3	176	GND
39	Data Bit 26	85	+2.5V	131	Data Bit 30	177	Data Strobe 16
40	Data Bit 27	86	Data Strobe 7	132	GND	178	Data Bit 62
41	Address Bit 2	87	Data Bit 58	133	Data Bit 31	179	Data Bit 63
42	GND	88	Data Bit 59	134	Parity Bit 4	180	I/O +2.5V
43	Address Bit 1	89	GND	135	Parity Bit 5	181	SPD Address 0
44	Parity Bit 0	90	SPD Write Protect	136	I/O +2.5V	182	SPD Address 1
45	Parity Bit 1	91	SPD Data	137	Clock 0	183	SPD Address 2
46	+2.5V	92	SPD Clock	138	Clock 0#	184	SPD +2.5V

Gnd = Ground
SPD = Serial presence detect
NC = No connection

DDR DIMMs use a single key notch to indicate voltage, as shown in Figure 6.12.

The 184-pin DDR DIMMs use two notches on each side to enable compatibility with both low- and high-profile latched sockets. Note that the key position is offset with respect to the center of the DIMM to prevent inserting it backward in the socket. The key notch is positioned to the left, centered, or to the right of the area between pins 52 and 53. This is used to indicate the I/O voltage for the DDR DIMM and to prevent installing the wrong type into a socket that might damage the DIMM.

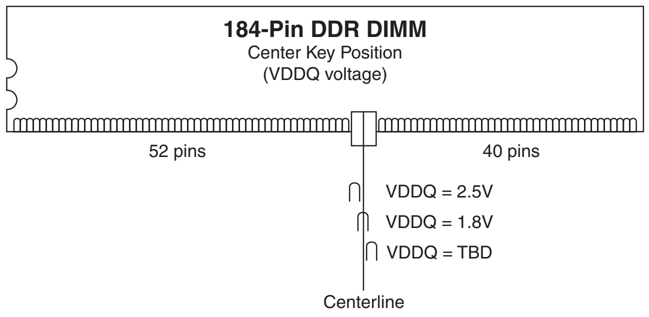


Figure 6.12 184-pin DDR SDRAM DIMM keying.

DDR2 DIMM Pinouts

Table 6.19 shows the pinout configuration of a 240-pin DDR2 SDRAM DIMM. Pins 1–120 are on the front side, and pins 121–240 are on the back. All pins should be gold plated.

Table 6.19 240-Pin DDR2 DIMM Pinouts

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	VREF	61	A4	121	VSS	181	VDDQ
2	VSS	62	VDDQ	122	DQ4	182	A3
3	DQ0	63	A2	123	DQ5	183	A1
4	DQ1	64	VDD	124	VSS	184	VDD
5	VSS	65	VSS	125	DM0	185	CK0
6	-DQS0	66	VSS	126	NC	186	-CK0
7	DQS0	67	VDD	127	VSS	187	VDD
8	VSS	68	NC	128	DQ6	188	A0
9	DQ2	69	VDD	129	DQ7	189	VDD
10	DQ3	70	A10/-AP	130	VSS	190	BA1
11	VSS	71	BA0	131	DQ12	191	VDDQ
12	DQ8	72	VDDQ	132	DQ13	192	-RAS
13	DQ9	73	-WE	133	VSS	193	-CS0
14	VSS	74	-CAS	134	DM1	194	VDDQ
15	-DQS1	75	VDDQ	135	NC	195	ODT0
16	DQS1	76	-CS1	136	VSS	196	A13
17	VSS	77	ODT1	137	CK1	197	VDD
18	NC	78	VDDQ	138	-CK1	198	VSS
19	NC	79	SS	139	VSS	199	DQ36
20	VSS	80	DQ32	140	DQ14	200	DQ37
21	DQ10	81	DQ33	141	DQ15	201	VSS
22	DQ11	82	VSS	142	VSS	202	DM4
23	VSS	83	-DQS4	143	DQ20	203	NC
24	DQ16	84	DQS4	144	DQ21	204	VSS
25	DQ17	85	VSS	145	VSS	205	DQ38
26	VSS	86	DQ34	146	DM2	206	DQ39
27	-DQS2	87	DQ35	147	NC	207	VSS
28	DQS2	88	VSS	148	VSS	208	DQ44
29	VSS	89	DQ40	149	DQ22	209	DQ45
30	DQ18	90	DQ41	150	DQ23	210	VSS
31	DQ19	91	VSS	151	VSS	211	DM5
32	VSS	92	-DQS5	152	DQ28	212	NC
33	DQ24	93	DQS5	153	DQ29	213	VSS
34	DQ25	94	VSS	154	VSS	214	DQ46

Table 6.19 Continued

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
35	VSS	95	DQ42	155	DM3	215	DQ47
36	-DQS3	96	DQ43	156	NC	216	VSS
37	DQS3	97	VSS	157	VSS	217	DQ52
38	VSS	98	DQ48	158	DQ30	218	DQ53
39	DQ26	99	DQ49	159	DQ31	219	VSS
40	DQ27	100	VSS	160	VSS	220	CK2
41	VSS	101	SA2	161	NC	221	-CK2
42	NC	102	NC	162	NC	222	VSS
43	NC	103	VSS	163	VSS	223	DM6
44	VSS	104	-DQS6	164	NC	224	NC
45	NC	105	DQS6	165	NC	225	VSS
46	NC	106	VSS	166	VSS	226	DQ54
47	VSS	107	DQ50	167	NC	227	DQ55
48	NC	108	DQ51	168	NC	228	VSS
49	NC	109	VSS	169	VSS	229	DQ60
50	VSS	110	DQ56	170	VDDQ	230	DQ61
51	VDDQ	111	DQ57	171	CKE1	231	VSS
52	CKE0	112	VSS	172	VDD	232	DM7
53	VDD	113	-DQS7	173	NC	233	NC
54	NC	114	DQS7	174	NC	234	VSS
55	NC	115	VSS	175	VDDQ	235	DQ62
56	VDDQ	116	DQ58	176	A12	236	DQ63
57	A11	117	DQ59	177	A9	237	VSS
58	A7	118	VSS	178	VDD	238	VDDSPD
59	VDD	119	SDA	179	A8	239	SA0
60	A5	120	SCL	180	A6	240	SA1

The 240-pin DDR2 DIMMs use two notches on each side to enable compatibility with both low- and high-profile latched sockets. The connector key is offset with respect to the center of the DIMM to prevent inserting it backward in the socket. The key notch is positioned in the center of the area between pins 64 and 65 on the front (184/185 on the back), and there is no voltage keying because all DDR2 DIMMs run on 1.8V.

DDR3 DIMM Pinouts

Table 6.20 shows the pinout configuration of a 240-pin DDR3 SDRAM DIMM. Pins 1–120 are on the front side, and pins 121–240 are on the back. All pins should be gold plated.

Table 6.20 240-Pin DDR3 DIMM Pinouts

Pin	Signal Front	Pin	Signal Back	Pin	Signal Front	Pin	Signal Back	Pin	Signal Front	Pin	Signal Back
1	VREFDQ	121	VSS	42	DQS8	162	DQS17	82	DQ33	202	VSS
2	VSS	122	DQ4	43	DQS8	163	VSS	83	VSS	203	DM4, DQS13
3	DQ0	123	DQ5	44	VSS	164	CB6	84	DQS4	204	DQS13
4	DQ1	124	VSS	45	CB2	165	CB7	85	DQS4	205	VSS
5	VSS	125	DM0,DQS9	46	CB3	166	VSS	86	VSS	206	DQ38
6	DQS0	126	NC,DQS9	47	VSS	167	TEST	87	DQ34	207	DQ39
7	DQS0	127	VSS	48	NC	168	Reset	88	DQ35	208	VSS
8	VSS	128	DQ6	Key	Key	Key	Key	89	VSS	209	DQ44
9	DQ2	129	DQ7	49	NC	169	CKE1	90	DQ40	210	DQ45
10	DQ3	130	VSS	50	CKE0	170	VDD	91	DQ41	211	VSS
11	VSS	131	DQ12	51	VDD	171	A15	92	VSS	212	DM5, DQS14
12	DQ8	132	DQ13	52	BA2	172	A14	93	DQS5	213	DQS14
13	DQ9	133	VSS	53	ERR-OUT(NC)	173	VDD	94	DQS5	214	VSS
14	VSS	134	DM1, DQS10	54	VDD	174	A12	95	VSS	215	DQ46
15	DQS1	135	NC,DQS10	55	A11	175	A9	96	DQ42	216	DQ47
16	DQS1	136	VSS	56	A7	176	VDD	97	DQ43	217	VSS
17	VSS	137	DQ14	57	VDD	177	A8	98	VSS	218	DQ52
18	DQ10	138	DQ15	58	A5	178	A6	99	DQ48	219	DQ53
19	DQ11	139	VSS	59	A4	179	VDD	100	DQ49	220	VSS
20	VSS	140	DQ20	60	VDD	180	A3	101	VSS	221	DM6, DQS15
21	DQ16	141	DQ21	61	A2	181	A1	102	DQS6	222	DQS15
22	DQ17	142	VSS	62	VDD	182	VDD	103	DQS6	223	VSS
23	VSS	143	DQS11	63	CK1/NC	183	VDD	104	VSS	224	DQ54
24	DQS2	144	DQS11	64	CK1/NC	184	CK0	105	DQ50	225	DQ55
25	DQS2	145	VSS	65	VDD	185	CK0	106	DQ51	226	VSS
26	VSS	146	DQ22	66	VDD	186	VDD	107	VSS	227	DQ60
27	DQ18	147	DQ23	67	VREFCA	187	NF	108	DQ56	228	DQ61
28	DQ19	148	VSS	68	Par_In(NC)	188	A0	109	DQ57	229	VSS
29	VSS	149	DQ28	69	VDD	189	VDD	110	VSS	230	DM7, DQS16
30	DQ24	150	DQ29	70	A10	190	BA1/BA0	111	DQS7	231	DQS16
31	DQ25	151	VSS	71	BA0/BA1	191	VDD	112	DQS7	232	VSS
32	VSS	152	DM3, DQS12	72	VDD	192	RAS	113	VSS	233	DQ62

Table 6.20 Continued

Pin	Signal Front	Pin	Signal Back	Pin	Signal Front	Pin	Signal Back	Pin	Signal Front	Pin	Signal Back
33	DQS3	153	DQS12	73	WE	193	S0	114	DQ58	234	DQ63
34	DQS3	154	VSS	74	CAS	194	VDD	115	DQ59	235	VSS
35	VSS	155	DQ30	75	VDD	195	ODT0	116	VSS	236	VDDSPD
36	DQ26	156	DQ31	76	S1	196	A13	117	SA0	237	SA1
37	DQ27	157	VSS	77	ODT1	197	VDD	118	SCL	238	SDA
38	VSS	158	CB4	78	VDD	198	NF	119	VSS	239	VSS
39	CB0	159	CB5	79	RFUSPD	199	VSS	120	VTT	240	VTT
40	CB1	160	VSS	80	VSS	200	DQ36				
41	VSS	161	DM8,DQS17	81	DQ32	201	DQ37				

NC = No Connect

NF = No Function

NU = Not Usable

RFU = Reserved Future Use

The 240-pin DDR3 DIMMs use two notches on each side to enable compatibility with both low- and high-profile latched sockets. The connector key is offset with respect to the center of the DIMM to prevent inserting it backward in the socket. The key notch is positioned in the center of the area between pins 48 and 49 on the front (168/169 on the back), and there is no voltage keying because all DDR3 DIMMs run on 1.5V.

RIMM Pinouts

RIMM modules and sockets are gold plated and designed for 25 insertion/removal cycles. Each RIMM has 184 pins, split into two groups of 92 pins on opposite ends and sides of the module. The pinout of the RIMM is shown in Table 6.21.

Table 6.21 RIMM Pinout

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A1	GND	B1	GND	A47	NC	B47	NC
A2	LData Bit A8	B2	LData Bit A7	A48	NC	B48	NC
A3	GND	B3	GND	A49	NC	B49	NC
A4	LData Bit A6	B4	LData Bit A5	A50	NC	B50	NC
A5	GND	B5	GND	A51	VREF	B51	VREF
A6	LData Bit A4	B6	LData Bit A3	A52	GND	B52	GND
A7	GND	B7	GND	A53	SPD Clock	B53	SPD Address 0
A8	LData Bit A2	B8	LData Bit A1	A54	+2.5V	B54	+2.5V
A9	GND	B9	GND	A55	SDA	B55	SPD Address 1
A10	LData Bit A0	B10	Interface Clock+	A56	SVDD	B56	SVDD
A11	GND	B11	GND	A57	SPD Write Protect	B57	SPD Address 2
A12	LCTMN	B12	Interface Clock-	A58	+2.5V	B58	+2.5V
A13	GND	B13	GND	A59	RSCK	B59	RCMD

Table 6.21 Continued

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A14	LCTM	B14	NC	A60	GND	B60	GND
A15	GND	B15	GND	A61	Rdata Bit B7	B61	RData Bit B8
A16	NC	B16	LROW2	A62	GND	B62	GND
A17	GND	B17	GND	A63	Rdata Bit B5	B63	RData Bit B6
A18	LROW1	B18	LROW0	A64	GND	B64	GND
A19	GND	B19	GND	A65	Rdata Bit B3	B65	RData Bit B4
A20	LCOL4	B20	LCOL3	A66	GND	B66	GND
A21	GND	B21	GND	A67	Rdata Bit B1	B67	RData Bit B2
A22	LCOL2	B22	LCOL1	A68	GND	B68	GND
A23	GND	B23	GND	A69	RCOL0	B69	RData Bit B0
A24	LCOL0	B24	LData Bit B0	A70	GND	B70	GND
A25	GND	B25	GND	A71	RCOL2	B71	RCOL1
A26	LData Bit B1	B26	LData Bit B2	A72	GND	B72	GND
A27	GND	B27	GND	A73	RCOL4	B73	RCOL3
A28	LData Bit B3	B28	LData Bit B4	A74	GND	B74	GND
A29	GND	B29	GND	A75	RROW1	B75	RROW0
A30	LData Bit B5	B30	LData Bit B6	A76	GND	B76	GND
A31	GND	B31	GND	A77	NC	B77	RROW2
A32	LData Bit B7	B32	LData Bit B8	A78	GND	B78	GND
A33	GND	B33	GND	A79	RCTM	B79	NC
A34	LSCK	B34	LCMD	A80	GND	B80	GND
A35	VCMOS	B35	VCMOS	A81	RCTMN	B81	RCFMN
A36	SOUT	B36	SIN	A82	GND	B82	GND
A37	VCMOS	B37	VCMOS	A83	Rdata Bit A0	B83	RCFM
A38	NC	B38	NC	A84	GND	B84	GND
A39	GND	B39	GND	A85	Rdata Bit A2	B85	RData Bit A1
A40	NC	B40	NC	A86	GND	B86	GND
A41	+2.5V	B41	+2.5V	A87	Rdata Bit A4	B87	RData Bit A3
A42	+2.5V	B42	+2.5V	A88	GND	B88	GND
A43	NC	B43	NC	A89	Rdata Bit A6	B89	RData Bit A5
A44	NC	B44	NC	A90	GND	B90	GND
A45	NC	B45	NC	A91	Rdata Bit A8	B91	RData Bit A7
A46	NC	B46	NC	A92	GND	B92	GND

The 16/18-bit RIMMs are keyed with two notches in the center. This prevents a backward insertion and prevents the wrong type (voltage) RIMM from being used in a system. Currently, all RIMMs run on 2.5V, but proposed 64-bit versions will run on only 1.8V. To allow for changes in the RIMMs, three keying options are possible in the design (see Figure 6.13). The left key (indicated as “DATUM A” in Figure 6.13) is fixed in position, but the center key can be in three different positions spaced 1mm or

2mm to the right, indicating different types of RIMMs. The current default is option A, as shown in Figure 6.13 and Table 6.22, which corresponds to 2.5V operation.

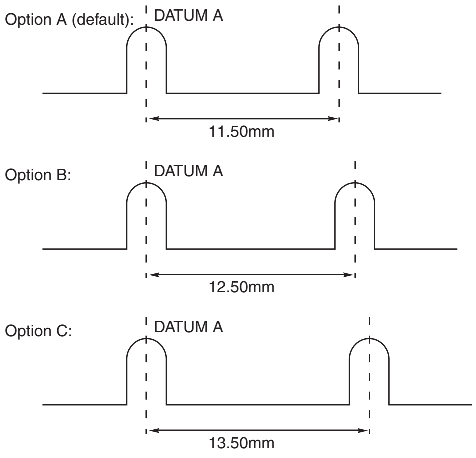


Figure 6.13 RIMM keying options.

Table 6.22 Possible Keying Options for RIMMs

Option	Notch Separation	Description
A	11.5mm	2.5V RIMM
B	12.5mm	Reserved
C	13.5mm	Reserved

RIMMs incorporate an SPD device, which is essentially a flash ROM onboard. This ROM contains information about the RIMM's size and type, including detailed timing information for the memory controller. The memory controller automatically reads the data from the SPD ROM to configure the system to match the RIMMs installed.

Figure 6.14 shows a typical PC RIMM installation. The RDRAM controller and clock generator are typically in the motherboard chipset North Bridge component. As you can see, the Rambus memory channel flows from the memory controller through each of up to three RIMM modules in series. Each module contains 4, 8, 16, or more RDRAM devices (chips), also wired in series, with an onboard SPD ROM for system configuration. Any RIMM sockets without a RIMM installed must have a continuity module, shown in the last socket in Figure 6.13. This enables the memory bus to remain continuous from the controller through each module (and, therefore, each RDRAM device on the module) until the bus finally terminates on the motherboard. Note how the bus loops from one module to another. For timing purposes, the first RIMM socket must be 6" or less from the memory controller, and the entire length of the bus must not be more than it would take for a signal to go from one end to another in four data clocks, or about 5ns.

Interestingly, Rambus does not manufacture the RDRAM devices (the chips) or the RIMMs; that is left to other companies. Rambus is merely a design company, and it has no chip fabs or manufacturing facilities of its own. It licenses its technology to other companies who then manufacture the devices and modules.

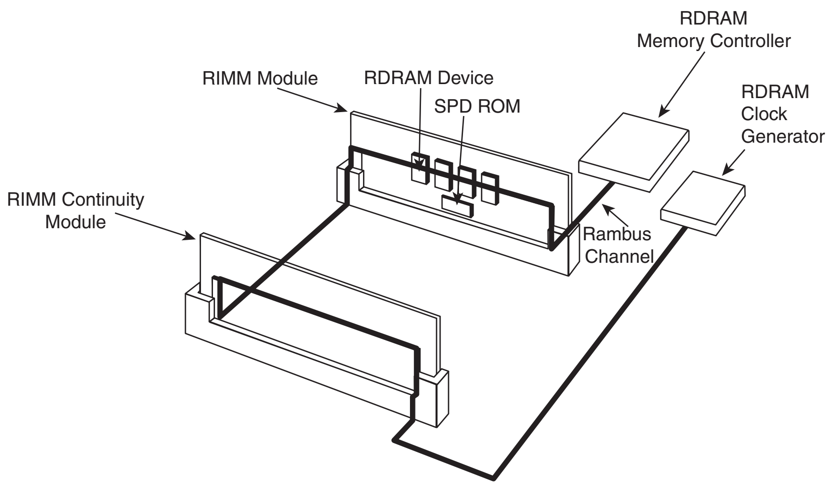


Figure 6.14 Typical RDRAM bus layout showing a RIMM and one continuity module.

Determining a Memory Module's Size and Features

Most memory modules are labeled with a sticker indicating the module's type, speed rating, and manufacturer. If you are attempting to determine whether existing memory can be used in a new computer, or if you need to replace memory in an existing computer, this information can be essential. Figure 6.15 illustrates the markings on typical 512MB and 1GB DDR memory modules from Crucial Technologies.

However, if you have memory modules that are not labeled, you can still determine the module type, speed, and capacity if the memory chips on the module are clearly labeled. For example, assume you have a memory module with chips labeled as follows:

MT46V64M8TG-75

By using an Internet search engine such as Google and entering the number from one of the memory chips, you can usually find the data sheet for the memory chips. Consider the following example: Say you have a registered memory module and want to look up the part number for the memory chips (usually eight or more chips) rather than the buffer chips on the module (usually from one to three, depending on the module design). In this example, the part number turns out to be a Micron memory chip that decodes like this:

MT = Micron Technologies (the memory chip maker)

46 = DDR SDRAM

V = 2.5V DC

64M8 = 8 million rows × 8 (equals 64) × 8 banks (often written as 64 Meg × 8)

TG = 66-pin TSOP chip package

–75 = 7.5ns @ CL2 latency (DDR 266)

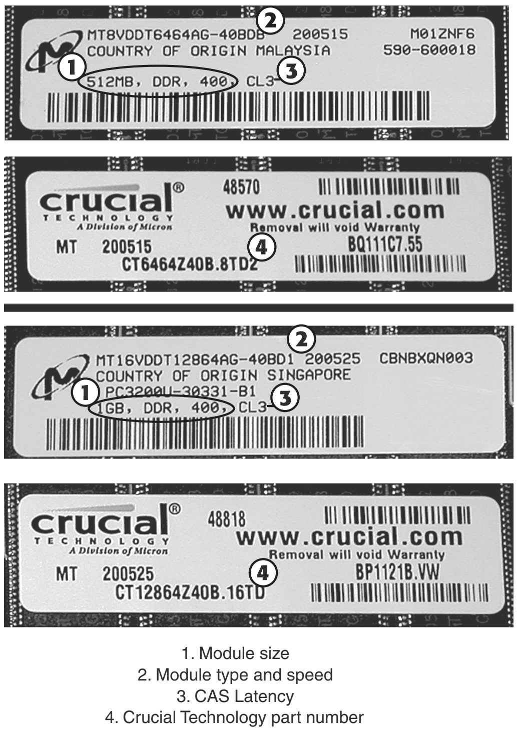

MT8VDDT6464AG-40BDB 200515 M01ZNF6
COUNTRY OF ORIGIN MALAYSIA 590-600018
512MB, DDR, 400, CL3
crucial TECHNOLOGY A Division of Micron
48570
www.crucial.com
Removal will void Warranty
MT 200515 BQ111C7.55
CT6464Z40B.8TD2
MT16VDDT12864AG-40BD1 200525 CBNBXQN003
COUNTRY OF ORIGIN SINGAPORE
PC3200U-30331-B1
1GB, DDR, 400, CL3
crucial TECHNOLOGY A Division of Micron
48818
www.crucial.com
Removal will void Warranty
MT 200525 BP1121B.VW
CT12864Z40B.16TD
1. Module size
2. Module type and speed
3. CAS Latency
4. Crucial Technology part number

Figure 6.15 Markings on 512MB (top) and 1GB (bottom) DDR memory modules from Crucial Technology.

The full datasheet for this example is located at http://download.micron.com/pdf/datasheets/dram/ddr/512MBDDRx4x8x16.pdf.

From this information, you can determine that the module has the following characteristics:

- The module runs at DDR266 speeds using standard 2.5V DC voltage.
- The module has a latency of CL2, so it can be used on any system that requires CL2 or slower latency (such as CL2.5 or CL3).
- Each chip has a capacity of 512Mb ($64 \times 8 = 512$).
- Each chip contains 8 bits. Because it takes 8 bits to make 1 byte, the capacity of the module can be calculated by grouping the memory chips on the module into groups of eight. If each chip contains 512Mb, a group of eight means that the module has a size of 512MB (512Mb $\times$ 8 = 512MB). A dual-bank module has two groups of eight chips for a capacity of 1GB (512Mb $\times$ 8 = 1024MB, or 1GB).

If the module has nine instead of eight memory chips (or 18 instead of 16), the additional chips are used for parity checking and support ECC error correction on servers with this feature.

To determine the size of the module in MB or GB and to determine whether the module supports ECC, count the memory chips on the module and compare them to Table 6.23. Note that the size of each memory chip in Mb is the same as the size in MB if the memory chips use an 8-bit design.

Table 6.23 Module Capacity Using 512Mb (64Mbit × 8) Chips

Number of Chips	Number of Bits in Each Bank	Module Size	Supports ECC?	Single or Dual-bank
8	64	512MB	No	Single
9	72	512MB	Yes	Single
16	64	1GB	No	Dual
18	72	1GB	Yes	Dual

The additional chip used by each group of eight chips provides parity checking, which is used by the ECC function on most server motherboards to correct single-bit errors.

A registered module contains 9 or 18 memory chips for ECC plus additional memory buffer chips. These chips are usually smaller in size and located near the center of the module, as shown previously in Figure 6.9.

Note

Some modules use 16-bit wide memory chips. In such cases, only four chips are needed for single-bank memory (five with parity/ECC support) and eight are needed for double-bank memory (10 with parity/ECC support). These memory chips use a design listed as capacity × 16, like this: 256Mb × 16.

You can also see this information if you look up the manufacturer, the memory type, and the organization in a search engine. For example, a web search for Micron "64 Meg x 8" DDR DIMM locates a design list list for Micron's 64MB–512MB modules at http://www.micron.com/support/designsupport/tools/ddrtoolbox/designfiles. The table lists the DIMM organization, SDRAM density, and other information for listed modules.

As you can see, with a little detective work, you can determine the size, speed, and type of a memory module—even if the module isn't marked, as long as the markings on the memory chips themselves are legible.

Tip

If you are unable to decipher a chip part number, you can use a program such as HWiNFO or SiSoftware Sandra to identify your memory module, as well as many other facts about your computer, including chipset, processor, empty memory sockets, and much more. You can download shareware versions of HWiNFO from www.hwinfo.com and SiSoftware Sandra from www.sisoftware.net.

Memory Banks

Memory chips (DIPs, SIMMs, SIPPs, and DIMMs) are organized in banks on motherboards and memory cards. You should know the memory bank layout and position on the motherboard and memory cards.

You need to know the bank layout when adding memory to the system. In addition, memory diagnostics report error locations by byte and bit addresses, and you must use these numbers to locate which bank in your system contains the problem.

The banks usually correspond to the data bus capacity of the system's microprocessor. Table 6.24 shows the widths of individual banks based on the type of PC.

Table 6.24 Memory Bank Widths on Various Systems

Processor	Data Bus	Memory Bank Width	Memory Bank Width (Parity/ECC)	8/9-bit SIMMs per Bank	32/36-bit SIMMs per Bank	64/72-bit DIMMs per Bank
8088	8-bit	8 bits	9 bits	1	—	—
8086	16-bit	16 bits	18 bits	2	—	—
286	16-bit	16 bits	18 bits	2	—	—
386SX, SL, SLC	16-bit	16 bits	18 bits	2	—	—
486SLC, SLC2	16-bit	16 bits	18 bits	2	—	—
386DX	32-bit	32 bits	36 bits	4	1	—
486SX, DX, DX2, DX4, 5x86	32-bit	32 bits	36 bits	4	1	—
Pentium, Athlon and newer running Single-Channel mode	64-bit	64 bits	72 bits	—	—	1
Pentium, Athlon and newer running Dual-Channel mode	64-bit	128 bits	144 bits	—	—	2

Dual-channel mode requires matched pairs of memory inserted into the memory sockets designated for dual-channel mode. If a single module or two different-size modules are used, or the dual-channel sockets are not used, the system runs in single-channel mode.

The number of bits for each bank can be made up of single chips, SIMMs, or DIMMs. Modern systems don't use individual chips; instead, they use only SIMMs or DIMMs. If the system has a 16-bit processor, such as a 386SX, it probably uses 30-pin SIMMs and has two SIMMs per bank. All the SIMMs in a single bank must be the same size and type.

A 486 system requires four 30-pin SIMMs or one 72-pin SIMM to make up a bank. A single 72-pin SIMM is 32 bits wide, or 36 bits wide if it supports parity. You can often tell whether a SIMM supports parity by counting its chips. To make a 32-bit SIMM, you could use 32 individual 1-bit wide chips, or you could use eight individual 4-bit wide chips to make up the data bits. If the system uses parity, 4 extra bits are required (36 bits total), so you would see one more 4-bit wide or four individual 1-bit wide chips added to the bank for the parity bits.

As you might imagine, 30-pin SIMMs are less than ideal for 32-bit or 64-bit systems (that is, 486 or Pentium) because you must use them in increments of four or eight per bank. Consequently, only a few 32-bit systems were ever built using 30-pin SIMMs, and no 64-bit systems have ever used 30-pin SIMMs. If a 32-bit system (such as any PC with a 386DX or 486 processor) uses 72-pin SIMMs, each SIMM represents a separate bank and the SIMMs can be added or removed on an individual basis rather than in groups of four, as would be required with 30-pin SIMMs. This makes memory configuration much easier and more flexible. In 64-bit systems that use SIMMs, two 72-pin SIMMs are required per bank.

DIMMs are ideal for Pentium and higher systems because the 64-bit width of the DIMM exactly matches the 64-bit width of the Pentium processor data bus. Therefore, each DIMM represents an individual bank, and they can be added or removed one at a time. Many recent systems have been designed to use matched pairs of memory modules for faster performance. So-called "dual-channel" designs treat a matched pair of modules as a single 128-bit device (or a 144-bit device if parity or ECC memory is used). In those cases, although a single module can be used, modules must be installed in pairs to achieve best performance.

The physical orientation and numbering of the SIMMs or DIMMs used on a motherboard is arbitrary and determined by the board's designers, so documentation covering your system or card comes in handy. You can determine the layout of a motherboard or an adapter card through testing, but that takes time and might be difficult, particularly after you have a problem with a system.

Caution

If your system supports dual-channel memory, be sure you use the correct memory sockets to enable dual-channel operation. Check the documentation to ensure that you use the correct pair of sockets. Most dual-channel systems will still run if the memory is not installed in a way that permits dual-channel operation, but performance is lower than if the memory were installed properly. Some systems provide dual-channel support if an odd number of modules are installed, as long as the total capacity of two modules installed in one channel equals the size of the single module in the other channel and all modules are the same speed and latency. Again, check your documentation for details.

Memory Module Speed

When you replace a failed memory module or install a new module as an upgrade, you typically must install a module of the same type and speed as the others in the system. You can substitute a module with a different (faster) speed but only if the replacement module's speed is equal to or faster than that of the other modules in the system.

Some people have had problems when "mixing" modules of different speeds. With the wide variety of motherboards, chipsets, and memory types, few ironclad rules exist. When in doubt as to which speed module to install in your system, consult the motherboard documentation for more information.

Substituting faster memory of the same type doesn't result in improved performance if the system still operates the memory at the same speed. Systems that use DIMMs or RIMMs can read the speed and timing features of the module from a special SPD ROM installed on the module and then set chipset (memory controller) timing accordingly. In these systems, you might see an increase in performance by installing faster modules, to the limit of what the chipset will support.

To place more emphasis on timing and reliability, there are Intel and JEDEC standards governing memory types that require certain levels of performance. These standards certify that memory modules perform within Intel's timing and performance guidelines.

The same common symptoms result when the system memory has failed or is simply not fast enough for the system's timing. The usual symptoms are frequent parity check errors or a system that does not operate at all. The POST might report errors, too. If you're unsure of which chips to buy for your system, contact the system manufacturer or a reputable chip supplier.

▶▶ See "Parity Checking," p. 560.

Parity and ECC

Part of the nature of memory is that it inevitably fails. These failures are usually classified as two basic types: hard fails and soft errors.

The best understood are hard fails, in which the chip is working and then, because of some flaw, physical damage, or other event, becomes damaged and experiences a permanent failure. Fixing this type of failure normally requires replacing some part of the memory hardware, such as the chip, SIMM, or DIMM. Hard error rates are known as HERs.

The other, more insidious type of failure is the soft error, which is a nonpermanent failure that might never recur or could occur only at infrequent intervals. (Soft fails are effectively "fixed" by powering the system off and back on.) Soft error rates are known as SERs.

More than 20 years ago, Intel made a discovery about soft errors that shook the memory industry. It found that alpha particles were causing an unacceptably high rate of soft errors or single event upsets (SEUs, as they are sometimes called) in the 16KB DRAMs that were available at the time. Because alpha particles are low-energy particles that can be stopped by something as thin and light as a sheet of paper, it became clear that for alpha particles to cause a DRAM soft error, they would have to be coming from within the semiconductor material. Testing showed trace elements of thorium and uranium in the plastic and ceramic chip packaging materials used at the time. This discovery forced all the memory manufacturers to evaluate their manufacturing processes to produce materials free from contamination.

Today, memory manufacturers have all but totally eliminated the alpha-particle source of soft errors. Many people believed that was justification for the industry trend to drop parity checking. The argument is that, for example, a 16MB memory subsystem built with 4MB technology would experience a soft error caused by alpha particles only about once every 16 years! The real problem with this thinking is that it is seriously flawed, and many system manufacturers and vendors were coddled into removing parity and other memory fault-tolerant techniques from their systems even though soft errors continue to be an ongoing problem. More recent discoveries prove that alpha particles are now only a small fraction of the cause of DRAM soft errors.

As it turns out, the biggest cause of soft errors today are cosmic rays. IBM researchers began investigating the potential of terrestrial cosmic rays in causing soft errors similar to alpha particles. The difference is that cosmic rays are very high-energy particles and can't be stopped by sheets of paper or other more powerful types of shielding. The leader in this line of investigation was Dr. J.F. Ziegler of the IBM Watson Research Center in Yorktown Heights, New York. He has produced landmark research into understanding cosmic rays and their influence on soft errors in memory.

One example of the magnitude of the cosmic ray soft-error phenomenon demonstrated that with a certain sample of non-IBM DRAMs, the SER at sea level was measured at 5950 FIT (failures in time, which is measured at 1 billion hours) per chip. This was measured under real-life conditions with the benefit of millions of device hours of testing. In an average system, this would result in a soft error occurring every 6 months or less. In power-user or server systems with a larger amount of memory, it could mean one or more errors per month! When the exact same test setup and DRAMs were moved to an underground vault shielded by more than 50 feet of rock, thus eliminating all cosmic rays, absolutely no soft errors were recorded. This not only demonstrates how troublesome cosmic rays can be, but it also proves that the packaging contamination and alpha-particle problem has indeed been solved.

Cosmic-ray-induced errors are even more of a problem in SRAMs than DRAMS because the amount of charge required to flip a bit in an SRAM cell is less than is required to flip a DRAM cell capacitor. Cosmic rays are also more of a problem for higher-density memory. As chip density increases, it becomes easier for a stray particle to flip a bit. It has been predicted by some that the soft error rate of a 64MB DRAM will be double that of a 16MB chip, and a 256MB DRAM will have a rate four times higher. As memory sizes continue to increase, it's likely that soft error rates will also increase.

Unfortunately, the PC industry has largely failed to recognize this cause of memory errors. Electrostatic discharge, power surges, or unstable software can much more easily explain away the random and intermittent nature of a soft error, especially right after a new release of an operating system or major application.

Studies have shown that the soft error rate for ECC systems is on the order of 30 times greater than the hard error rate. This is not surprising to those familiar with the full effects of cosmic-ray-generated soft errors. The number of errors experienced varies with the density and amount of memory present. Studies show that soft errors can occur from once a month or less to several times a week or more!

Although cosmic rays and other radiation events are the biggest cause of soft errors, soft errors can also be caused by the following:

- **Power glitches or noise on the line**—This can be caused by a defective power supply in the system or by defective power at the outlet.
- **Incorrect type or speed rating**—The memory must be the correct type for the chipset and match the system access speed.
- **RF (radio frequency) interference**—Caused by radio transmitters in close proximity to the system, which can generate electrical signals in system wiring and circuits. Keep in mind that the increased use of wireless networks, keyboards, and mouse devices can lead to a greater risk of RF interference.
- **Static discharges**—Causes momentary power spikes, which alter data.
- **Timing glitches**—Data doesn't arrive at the proper place at the proper time, causing errors. Often caused by improper settings in the BIOS Setup, by memory that is rated slower than the system requires, or by overclocked processors and other system components.
- **Heat buildup**—High-speed memory modules run hotter than older modules. RDRAM RIMM modules were the first memory to include integrated heat spreaders, and many high-performance DDR and DDR2 memory modules now include heat spreaders to help fight heat buildup.

Most of these problems don't cause chips to permanently fail (although bad power or static can damage chips permanently), but they can cause momentary problems with data.

How can you deal with these errors? Just ignoring them is certainly not the best way to deal with them, but unfortunately that is what many system manufacturers and vendors are doing today. The best way to deal with this problem is to increase the system's fault tolerance. This means implementing ways of detecting and possibly correcting errors in PC systems. Three basic levels and techniques are used for fault tolerance in modern PCs:

- Nonparity
- Parity
- ECC

Nonparity systems have no fault tolerance at all. The only reason they are used is because they have the lowest inherent cost. No additional memory is necessary, as is the case with parity or ECC techniques. Because a parity-type data byte has 9 bits versus 8 for nonparity, memory cost is approximately 12.5% higher. Also, the nonparity memory controller is simplified because it does not need the logic gates to calculate parity or ECC check bits. Portable systems that place a premium on minimizing power might benefit from the reduction in memory power resulting from fewer DRAM chips. Finally, the memory system data bus is narrower, which reduces the amount of data buffers. The statistical probability of memory failures in a modern office desktop computer is now estimated at about one error every few months. Errors will be more or less frequent depending on how much memory you have.

This error rate might be tolerable for low-end systems that are not used for mission-critical applications. In this case, the extreme market sensitivity to price probably can't justify the extra cost of parity or ECC memory, and such errors then must be tolerated.

At any rate, having no fault tolerance in a system is simply gambling that memory errors are unlikely. You further gamble that if they do occur, memory errors will result in an inherent cost less than the additional hardware necessary for error detection. However, the risk is that these memory errors can lead to serious problems. A memory error in a calculation could cause the wrong value to go into a bank check. In a server, a memory error could force a system to hang and bring down all LAN-resident client systems with subsequent loss of productivity. Finally, with a nonparity or non-ECC memory system, tracing the problem is difficult, which is not the case with parity or ECC. These techniques at least isolate a memory source as the culprit, thus reducing both the time and cost of resolving the problem.

Parity Checking

One standard IBM set for the industry is that the memory chips in a bank of nine each handle 1 bit of data: 8 bits per character plus 1 extra bit called the parity bit. The parity bit enables memory-control circuitry to keep tabs on the other 8 bits—a built-in cross-check for the integrity of each byte in the system. If the circuitry detects an error, the computer stops and displays a message informing you of the malfunction. If you are running a GUI operating system, such as Windows or OS/2, a parity error generally manifests itself as a locked system. When you reboot, the BIOS should detect the error and display the appropriate error message.

SIMMs and DIMMs are available both with and without parity bits. Originally, all PC systems used parity-checked memory to ensure accuracy. Starting in 1994, a disturbing trend developed in the PC-compatible marketplace. Most vendors began shipping systems without parity checking or any other means of detecting or correcting errors! These systems can use cheaper nonparity SIMMs, which saves about 10%–15% on memory costs for a system. Parity memory results in increased initial system cost, primarily because of the additional memory bits involved. Parity can't correct system errors, but because parity can detect errors, it can make the user aware of memory errors when they happen. This has two basic benefits:

- Parity guards against the consequences of faulty calculations based on incorrect data.
- Parity pinpoints the source of errors, which helps with problem resolution, thus improving system serviceability.

PC systems can easily be designed to function using either parity or nonparity memory. The cost of implementing parity as an option on a motherboard is virtually nothing; the only cost is in actually purchasing the parity SIMMs or DIMMs. This enables a system manufacturer to offer its system purchasers the choice of parity if the purchasers feel the additional cost is justified for their particular applications.

Unfortunately, several of the big names began selling systems without parity to reduce their prices, and they did not make it well known that the lower cost meant parity memory was no longer included as standard. This began happening mostly in 1994 and 1995, and it has continued until recently, with few people understanding the full implications. After one or two major vendors did this, most of the others were forced to follow to remain price-competitive.

Because nobody wanted to announce this information, it remained sort of a dirty little secret within the industry. Originally, when this happened you could still specify parity memory when you ordered a system, even though the default configurations no longer included it. There was a 10%–15% surcharge on the memory, but those who wanted reliable, trustworthy systems could at least get them, provided they knew to ask, of course. Then a major bomb hit the industry, in the form of the Intel Triton 430FX Pentium chipset, which was the first major chipset on the market that did not support parity checking at all! It also became the most popular chipset of its time and was found in practically all Pentium motherboards sold in the 1995 timeframe. This set a disturbing trend for the next few years. All but one of Intel's Pentium processor chipsets after the 430FX did not support parity-checked memory; the only one that did was the 430HX Triton II.

Since then, Intel and other chipset manufacturers have put support for parity and ECC memory in most of their chipsets (especially so in their higher-end models). The low-end chipsets, however, typically do lack support for either parity or ECC. If more reliability is important to you, make sure the systems you purchase have this support. In Chapter 4, you can learn which recent chipsets support parity and ECC memory and which ones do not.

Let's look at how parity checking works, and then examine in more detail the successor to parity checking, called ECC, which not only can detect but also correct memory errors on-the-fly.

How Parity Checking Works

IBM originally established the odd parity standard for error checking. The following explanation might help you understand what is meant by odd parity. As the 8 individual bits in a byte are stored in memory, a parity generator/checker, which is either part of the CPU or located in a special chip on the motherboard, evaluates the data bits by adding up the number of 1s in the byte. If an even number of 1s is found, the parity generator/checker creates a 1 and stores it as the ninth bit (parity bit) in the parity memory chip. That makes the sum for all 9 bits (including the parity bit) an odd number. If the original sum of the 8 data bits is an odd number, the parity bit created would be a 0, keeping the sum for all 9 bits an odd number. The basic rule is that the value of the parity bit is always chosen so that the sum of all 9 bits (8 data bits plus 1 parity bit) is stored as an odd number. If the system used even parity, the example would be the same except the parity bit would be created to ensure an even sum. It doesn't matter whether even or odd parity is used; the system uses one or the other, and it is completely transparent to the memory chips involved. Remember that the 8 data bits in a byte are numbered 0 1 2 3 4 5 6 7. The following examples might make it easier to understand:

```
Data bit number:     0 1 2 3 4 5 6 7     Parity bit
Data bit value:      1 0 1 1 0 0 1 1     0
```

In this example, because the total number of data bits with a value of 1 is an odd number (5), the parity bit must have a value of 0 to ensure an odd sum for all 9 bits.

Here is another example:

```
Data bit number:     0 1 2 3 4 5 6 7     Parity bit
Data bit value:      1 1 1 1 0 0 1 1     1
```

In this example, because the total number of data bits with a value of 1 is an even number (6), the parity bit must have a value of 1 to create an odd sum for all 9 bits.

When the system reads memory back from storage, it checks the parity information. If a (9-bit) byte has an even number of bits, that byte must have an error. The system can't tell which bit has changed or whether only a single bit has changed. If 3 bits changed, for example, the byte still flags a parity-check error; if 2 bits changed, however, the bad byte could pass unnoticed. Because multiple bit errors (in a single byte) are rare, this scheme gives you a reasonable and inexpensive ongoing indication that memory is good or bad.

The following examples show parity-check messages for three types of older systems:

For the IBM PC:	`PARITY CHECK x`	
For the IBM XT:	`PARITY CHECK x`	`yyyyy (z)`
For the IBM AT and late model XT:	`PARITY CHECK x`	`yyyyy`

where `x` is 1 or 2:

1 = Error occurred on the motherboard

2 = Error occurred in an expansion slot

In this example, `yyyyy` represents a number from 00000 through FFFFF that indicates, in hexadecimal notation, the byte in which the error has occurred.

Where (z) is (S) or (E):

(S) = Parity error occurred in the system unit

(E) = Parity error occurred in an optional expansion chassis

Note

An expansion chassis was an option IBM sold for the original PC and XT systems to add more expansion slots.

When a parity-check error is detected, the motherboard parity-checking circuits generate a nonmaskable interrupt (NMI), which halts processing and diverts the system's attention to the error. The NMI causes a routine in the ROM to be executed. On some older IBM systems, the ROM parity-check routine halts the CPU. In such a case, the system locks up, and you must perform a hardware reset or a power-off/power-on cycle to restart the system. Unfortunately, all unsaved work is lost in the process.

Most systems do not halt the CPU when a parity error is detected; instead, they offer you the choice of rebooting the system or continuing as though nothing happened. Additionally, these systems might display the parity error message in a different format from IBM, although the information presented is basically the same. For example, most systems with a Phoenix BIOS display one of these messages:

```
Memory parity interrupt at xxxx:xxxx
Type (S)hut off NMI, Type (R)eboot, other keys to continue
```

or

```
I/O card parity interrupt at xxxx:xxxx
Type (S)hut off NMI, Type (R)eboot, other keys to continue
```

The first of these two messages indicates a motherboard parity error (Parity Check 1), and the second indicates an expansion-slot parity error (Parity Check 2). Notice that the address given in the form xxxx:xxxx for the memory error is in a segment:offset form rather than a straight linear address, such as with IBM's error messages. The segment:offset address form still gives you the location of the error to a resolution of a single byte.

You have three ways to proceed after viewing this error message:

- You can press S, which shuts off parity checking and resumes system operation at the point where the parity check first occurred.

- You can press R to force the system to reboot, losing any unsaved work.
- You can press any other key to cause the system to resume operation with parity checking still enabled.

If the problem occurs, it is likely to cause another parity-check interruption. It's usually prudent to press S, which disables the parity checking so you can then save your work. In this case, it's best to save your work to a floppy disk or USB flash drive to prevent the possible corruption of the hard disk. You should also avoid overwriting any previous (still good) versions of whatever file you are saving because you could be saving a bad file caused by the memory corruption. Because parity checking is now disabled, your save operations will not be interrupted. Then, you should power the system off, restart it, and run whatever memory diagnostics software you have to try to track down the error. In some cases, the POST finds the error on the next restart, but you usually need to run a more sophisticated diagnostics program—perhaps in a continuous mode—to locate the error.

Systems with an AMI BIOS display the parity error messages in the following forms:

```
ON BOARD PARITY ERROR ADDR (HEX) = (xxxxx)
```

or

```
OFF BOARD PARITY ERROR ADDR (HEX) = (xxxxx)
```

These messages indicate that an error in memory has occurred during the POST, and the failure is located at the address indicated. The first one indicates that the error occurred on the motherboard, and the second message indicates an error in an expansion slot adapter card. The AMI BIOS can also display memory errors in the following manners:

```
Memory Parity Error at xxxxx
```

or

```
I/O Card Parity Error at xxxxx
```

These messages indicate that an error in memory has occurred at the indicated address during normal operation. The first one indicates a motherboard memory error, and the second indicates an expansion slot adapter memory error.

Although many systems enable you to continue processing after a parity error and even allow disabling further parity checking, continuing to use your system after a parity error is detected can be dangerous. The idea behind letting you continue using either method is to give you time to save any unsaved work before you diagnose and service the computer, but be careful how you do this.

Note that these messages can vary depending not only on the ROM BIOS but also on your operating system. Protected mode operating systems, such as most versions of Windows, trap these errors and run their own handler program that displays a message different from what the ROM would have displayed. The message might be associated with a blue screen or might be a trap error, but it usually indicates that it is memory or parity related. For example, Windows 98 displays a message indicating "Memory parity error detected. System halted." when such an error has occurred.

Caution

When you are notified of a memory parity error, remember the parity check is telling you that memory has been corrupted. Do you want to save potentially corrupted data over the good file from the last time you saved? Definitely not! Be sure you save your work with a different filename. In addition, after a parity error, save only to a floppy disk or USB flash drive if possible and avoid writing to the hard disk; there is a slight chance that the hard drive could become corrupt if you save the contents of corrupted memory.

After saving your work, determine the cause of the parity error and repair the system. You might be tempted to use an option to shut off further parity checking and simply continue using the system as though nothing were wrong. Doing so is like unscrewing the oil pressure warning indicator bulb on a car with an oil leak so the oil pressure light won't bother you anymore!

Several years ago, when memory was more expensive, a few companies marketed SIMMs with bogus parity chips. Instead of actually having the extra memory chips needed to store the parity bits, these "logic parity" or parity "emulation" SIMMs used an onboard parity generator chip. This chip ignored any parity the system was trying to store on the SIMM, but when data was retrieved, it always ensured that the correct parity was returned, thus making the system believe all was well even though there might have been a problem.

These bogus parity modules were used because memory was much more expensive and a company could offer a "parity" SIMM for only a few dollars more with the fake chip. Unfortunately, identifying them can be difficult. The bogus parity generator doesn't look like a memory chip and has different markings from the other memory chips on the SIMM. Most of them had a "GSM" logo, which indicated the original manufacturer of the parity logic device, not necessarily the SIMM itself.

One way to positively identify these bogus fake parity SIMMs is by using a hardware SIMM test machine, such as those by Tanisys (www.tanisys.com), CST (www.simmtester.com), or Innoventions (www.memorytest.com). I haven't seen DIMMs or RIMMs with fake parity/ECC bits, and memory prices have come down far enough that it probably isn't worth the trouble anymore.

Error-correcting Code (ECC)

ECC goes a big step beyond simple parity-error detection. Instead of just detecting an error, ECC allows a single bit error to be corrected, which means the system can continue without interruption and without corrupting data. ECC, as implemented in most PCs, can only detect, not correct, double-bit errors. Because studies have indicated that approximately 98% of memory errors are the single-bit variety, the most commonly used type of ECC is one in which the attendant memory controller detects and corrects single-bit errors in an accessed data word (double-bit errors can be detected but not corrected). This type of ECC is known as *single-bit error-correction double-bit error detection (SEC-DED)* and requires an additional 7 check bits over 32 bits in a 4-byte system and an additional 8 check bits over 64 bits in an 8-byte system. ECC in a 4-byte (32-bit, such as a 486) system obviously costs more than nonparity or parity, but in an 8-byte wide bus (64-bit, such as Pentium/Athlon) system, ECC and parity costs are equal because the same number of extra bits (8) is required for either parity or ECC. Because of this, you can purchase parity SIMMs (36-bit), DIMMs (72-bit), or RIMMs (18-bit) for 32-bit systems and use them in an ECC mode if the chipset supports ECC functionality. If the system uses SIMMs, two 36-bit (parity) SIMMs are added for each bank (for a total of 72 bits), and ECC is done at the bank level. If the system uses DIMMs, a single parity/ECC 72-bit DIMM is used as a bank and provides the additional bits. RIMMs are installed in singles or pairs, depending on the chipset and motherboard. They must be 18-bit versions if parity/ECC is desired.

ECC entails the memory controller calculating the check bits on a memory-write operation, performing a compare between the read and calculated check bits on a read operation, and, if necessary, correcting bad bits. The additional ECC logic in the memory controller is not very significant in this age of inexpensive, high-performance VLSI logic, but ECC actually affects memory performance on writes. This is because the operation must be timed to wait for the calculation of check bits and, when the system waits for corrected data, reads. On a partial-word write, the entire word must first be read, the affected byte(s) rewritten, and then new check bits calculated. This turns partial-word write operations into slower read-modify writes. Fortunately, this performance hit is very small, on the order of a few percent at maximum, so the tradeoff for increased reliability is a good one.

Most memory errors are of a single-bit nature, which ECC can correct. Incorporating this fault-tolerant technique provides high system reliability and attendant availability. An ECC-based system is a good choice for servers, workstations, or mission-critical applications in which the cost of a potential memory error outweighs the additional memory and system cost to correct it, along with ensuring that it does not detract from system reliability. If you value your data and use your system for important (to you) tasks, you'll want ECC memory. No self-respecting manager would build or run a network server, even a lower-end one, without ECC memory.

By designing a system that allows for the choice of ECC, parity, or nonparity, you allow the users to choose the level of fault tolerance desired, as well as how much they want to gamble with their data.

Installing RAM Upgrades

Adding memory to a system is one of the most useful upgrades you can perform and also one of the least expensive—especially when you consider the increased performance of Windows and Linux when you give them access to more memory. In some cases, doubling the memory can practically double the speed of a computer.

The following sections discuss adding memory, including selecting memory chips, installing memory chips, and testing the installation.

Upgrade Options and Strategies

Adding memory can be an inexpensive solution; at this writing, the cost of memory has fallen to about 12 cents per megabyte or less. A small dose can give your computer's performance a big boost.

How do you add memory to your PC? You have two options, listed in order of convenience and cost:

- Adding memory in vacant slots on your motherboard
- Replacing your current motherboard's memory with higher-capacity memory

If you decide to upgrade to a more powerful computer system or motherboard, you usually can't salvage the memory from your previous system. Most of the time it is best to plan on equipping a new board with the optimum type of memory that it supports.

Be sure to carefully weigh your future needs for computing speed and a multitasking operating system against the amount of money you spend to upgrade current equipment.

To determine at what point you should add memory, you can use the Performance Monitor (Perfmon.msc) built into Windows 2000 and Windows XP. You can launch it remotely or from the server's own console. To check memory usage, select Memory as the Performance object and enable the following counters:

- **Pages/Sec**—This counter measures the number of times per second that the system uses virtual (swapfile) memory rather than physical memory. A value above 20 indicates a potential problem. Check the virtual memory settings; if the counter remains above 20, install more memory.
- **Committed Bytes and Available Bytes**—Committed Bytes tracks virtual memory in use; Available Bytes tracks physical memory available. Add more memory if you run short of available bytes.
- **Cache Bytes**—Measures the amount of RAM used for file system cache. Add more RAM if this amount exceeds 4MB.

To gather this same information in Windows Vista, open the Reliability and Performance Monitor, select the Performance Monitor, and use the Add tool to select these counters from the Memory category.

Before you add RAM to a system (or replace defective RAM chips), you must determine the memory modules required for your system. Your system documentation has this information.

If you need to replace a defective memory module or add more memory to your system, there are several ways to determine the correct module for your system:

- **Inspect the modules installed in your system**. Each module has markings that indicate its capacity and speed. RAM capacity and speed were discussed in detail earlier in this chapter. You can write down the markings on the memory module and use them to determine the type of memory you need. Check with a local store or an online memory vendor for help.
- **Look up your system using the online memory-configuration utility provided by your preferred memory vendor**. Originally, these configuration utilities were primarily for users of name-brand systems. However, most vendors have now added major motherboard brands and models to their databases. Therefore, if you know your system or motherboard brand and model, you can find the memory that is recommended.
- **Download and run analysis software provided by the memory module maker or from a third party**. SiSoftware Sandra and similar programs use the SPD chip on each module to determine this information.
- **Consult your system documentation**. I list this option last for a reason. If you have installed BIOS upgrades, you might be able to use larger and faster memory than your documentation lists as supported by your system. You should check the latest tech notes and documentation available online for your system and check the BIOS version installed in your system to determine which memory-related features it has. A BIOS upgrade might enable your system to use faster memory.

Adding the wrong modules to a system can make it as unreliable as leaving a defective module installed and trying to use the system in that condition.

Note

Before upgrading an older Pentium (P5 class) system beyond 64MB of RAM, be sure your chipset supports caching more than 64MB. Adding RAM beyond the amount supported by your L2 cache slows performance rather than increases it. See the section "Cache Memory: SRAM," earlier in this chapter, and the discussion of chipsets in Chapter 4 for a more complete explanation of this common system limitation. This limitation was mostly with Pentium (P5) class chipsets. Pentium II and later processors, including the AMD Athlon,Duron, and Sempron famiies have the L2 cache controller integrated in the processor (not the chipset), which supports caching up to 4GB and beyond on most newer models. With the higher price of older SIMM-type memory modules per megabyte compared to SDRAM and DDR-SDRAM, you might find it more cost-effective to replace your motherboard, processor, and memory with new components than to add memory to an older system that uses SIMMs.

Selecting and Installing Memory

Installing extra memory on your motherboard is an easy way to add memory to your computer. Most systems have at least one vacant memory socket where you can install extra memory at a later time and speed up your computer.

If your system requires dual-channel memory, as some high-performance systems do, you must use two identical memory modules (same size, speed, and type).

Purchasing Memory

When purchasing memory, there are some issues you need to consider. Some are related to the manufacturing and distribution of memory, whereas others depend on the type of memory you are purchasing. This section covers some of the issues you should consider when purchasing memory.

Suppliers

Many companies sell memory, but only a few companies actually make memory. Additionally, only a few companies make memory chips, but many more companies make memory modules such as SIMMs, DIMMs, and RIMMs. Most of the companies that make the actual RAM chips also make modules containing their own chips. Other companies, however, strictly make modules; these companies purchase memory chips from several chip makers and then produce modules with these chips. Finally, some companies don't make either the chips or modules. Instead, they purchase modules made by other companies and relabel them.

I refer to memory modules made by the chip manufacturers as *first-party modules*, whereas those made by module (but not chip) manufacturers I call *second-party modules*. Finally, those that are simply relabeled first- or second-party modules under a different name are called *third-party modules*. I always prefer to purchase first- or second-party modules if I can because they are better documented. In essence they have a better pedigree and their quality is generally more assured. Not to mention that purchasing from the first or second party eliminates one or more middlemen in the distribution process as well.

First-party manufacturers (where the same company makes the chips and the modules) include Micron (www.crucial.com), Infineon (formerly Siemens), Samsung, Mitsubishi, Toshiba, NEC, and others. Second-party companies that make the modules (but not the chips) include Kingston, Viking, PNY, Simple Tech, Smart, Mushkin, and OCZ Technologies. At the third-party level you are not purchasing from a manufacturer but from a reseller or remarketer instead.

Most of the large manufacturers don't sell small quantities of memory to individuals, but some have set up factory outlet stores where individuals can purchase as little as a single module. One of the largest memory manufacturers in the world, Micron, sells direct to the consumer at www.crucial.com. Because you are buying direct, the pricing at these outlets is often highly competitive with second- and third-party suppliers.

Considerations in Purchasing or Reusing SIMMs

When purchasing SIMMs, the main things to consider are as follows:

- Do you need FPM (Fast Page Mode) or EDO (extended data out) versions?
- Do you need ECC or non-ECC?
- What speed grade do you need?
- Can you use memory from other systems or from parts stores as an alternative to purchasing (very expensive) new memory?

Most Pentium systems after 1995 used EDO SIMMs that were non-ECC and rated for 60ns access time. If your system is older than that, you might need regular FPM versions. The FPM and EDO types are interchangeable in many systems, but some older systems do not accept the EDO type. If your system is designed for high-reliability using ECC, you might need (or want) ECC versions; otherwise, standard non-ECC types are typically used. You can mix the two, but in that case the system defaults to non-ECC mode.

Unfortunately, FPM and EDO SIMMs are obsolete by today's standards, so they are much more expensive than newer, better, and faster types of memory. This can make adding memory to older systems cost prohibitive.

Tip

Instead of buying new SIMM memory for older systems, check with computer repair shops or other users who might have a collection of old parts.

Considerations in Purchasing DIMMs

When purchasing DIMMs, the main things to consider are as follows:

- Do you need SDR, DDR, DDR2, or DDR3 versions?
- Do you need ECC or non-ECC?
- Do you need registered or standard (unbuffered) versions?
- What speed grade do you need?
- Do you need a specific column address strobe (CAS) latency?

Currently, DIMMs come in SDR (SDRAM), DDR, DDR2, and DDR3 versions. They are not interchangeable because they use completely different signaling and have different notches to prevent a mismatch. High-reliability systems such as servers can use ECC versions, although most desktop systems use the less-expensive non-ECC types. Most systems use standard unbuffered DIMMs, but file server or workstation motherboards designed to support very large amounts of memory might require registered DIMMs (which also include ECC support). Registered DIMMs contain their own memory registers, enabling the module to hold more memory than a standard DIMM. DIMMs come in a variety of speeds, with the rule that you can always substitute a faster one for a slower one, but not vice versa. As an example, if your system requires PC2700 DDR DIMMs, you can install faster PC3200 DDR DIMMs but not slower PC2100 versions.

Another speed-related issue is the column address strobe (CAS) latency. Sometimes this specification is abbreviated CAS or CL and is expressed in a number of cycles, with lower numbers indicating higher speeds (fewer cycles). The lower CAS latency shaves a cycle off a burst mode read, which marginally improves memory performance. Single data rate DIMMs are available in CL3 or CL2 versions.. DDR DIMMs are available in CL2.5 or CL2 versions. DDR2 DIMMs are available in CL 3, 4 or 5. DDR3 DIMMs are available in CL 7, 8, and 9. With all memory types, the lowest CL number is the fastest (and usually the most expensive) memory type. You can mix DIMMs with different CAS latency ratings, but the system usually defaults to cycling at the slower speeds of the lowest common denominator.

Considerations in Purchasing RIMMs

When purchasing RIMMs, the main things to consider are as follows:

- Do you need 184-pin (16/18-bit) or 232-pin (32/36-bit) versions?
- Do you need ECC or non-ECC?
- What speed grade do you need?

RIMMs are available in 184-pin and 232-pin versions, and although they appear to be the same size, they are not interchangeable. Differences exist in the notches that prevent a mismatch. High-reliability systems might want or need ECC versions, which have extra ECC bits. As with other memory types, you can mix ECC and non-ECC types, but systems can't use the ECC capability.

Replacing Modules with Higher-capacity Versions

If all the memory module slots on your motherboard are occupied, your best option is to remove an existing bank of memory and replace it with higher-capacity modules. For example, if you have a motherboard that supports two DIMM modules (each representing one bank on a processor with a 64-bit data bus) and runs in single-channel mode, you could remove one of them and replace it with a higher-capacity version. For example, if you have two 256MB modules, giving you a total of 512MB, you could remove one of the 256MB modules and replace it with a 512MB unit, in which case you'd then have a total of 768MB of RAM.

However, just because higher-capacity modules are available that are the correct pin count to plug into your motherboard, don't automatically assume the higher-capacity memory will work. Your system's chipset and BIOS set limits on the capacity of the memory you can use. Check your system or motherboard documentation to see which size modules work with it before purchasing the new RAM. You should make sure you have the latest BIOS for your motherboard when installing new memory.

If your system supports dual-channel memory, you must use matched pairs of DDR or DDR2 modules (depending on which type your system supports) and install them in the correct location on the motherboard to achieve the superior memory performance that dual-channel access offers. You should consult your motherboard manual for details.

Installing DIMM or RIMM Modules

This section discusses installing memory—specifically, SIMM or DIMM modules. It also covers the problems you are most likely to encounter and how to avoid them. You also get information on configuring your system to use new memory.

When you install or remove memory, you are most likely to encounter the following problems:

- Electrostatic discharge
- Improperly seated modules
- Incorrect memory configuration settings in the BIOS Setup

To prevent electrostatic discharge (ESD) when you install sensitive memory chips or boards, you shouldn't wear synthetic-fiber clothing or leather-soled shoes because these promote the generation of static charges. Remove any static charge you are carrying by touching the system chassis before you begin, or better yet, wear a good commercial grounding strap on your wrist. You can order one from any electronics parts store. A grounding strap consists of a conductive wristband grounded at the other end through a 1-meg ohm resistor by a wire clipped to the system chassis. Be sure the system you are working on is unplugged.

Caution

Be sure to use a properly designed commercial grounding strap; do not make one yourself. Commercial units have a 1-meg ohm resistor that serves as protection if you accidentally touch live power. The resistor ensures that you do not become the path of least resistance to the ground and therefore become electrocuted. An improperly designed strap can cause the power to conduct through you to the ground, possibly killing you.

Follow this procedure to upgrade DIMM or RIMM memory on a typical desktop PC:

1. Shut down the system and unplug it. As an alternative to unplugging it, you can turn off the power supply using the on/off switch on the rear of some power supplies. Wait about 10 seconds for any remaining current to drain from the motherboard.

2. Open the system. See the system or case instructions for details.
3. Connect a static guard wrist strap to your wrist and then to a metal portion of the system chassis, such as the frame. Make sure the metal plate on the inside of the wrist strap is tight against the skin of your wrist.
4. Some motherboards feature an LED that glows as long as the motherboard is receiving power. Wait until the LED dims before removing or installing memory.
5. Move obstructions inside the case, such as cables or wires, out of the way of the memory modules and empty sockets. If you must remove a cable or wire, note its location and orientation so you can replace it later.
6. If you need to remove an existing DIMM or RIMM, flip down the ejector tab at each end of the module and lift the module straight up out of the socket. Note the keying on the module.
7. Note the specific locations needed if you are inserting modules to operate in dual-channel mode. The sockets used for dual-channel memory might use a different-colored plastic to distinguish them from other sockets, but ultimately you should consult the documentation for your motherboard or system to determine the proper orientation.
8. To insert a DIMM or RIMM module into a socket, ensure that the ejector tabs are flipped down on the socket you plan to use. DIMMs and RIMMs are keyed by notches along the bottom connector edges that are offset from the center so they can be inserted in only one direction, as shown in Figure 6.16.

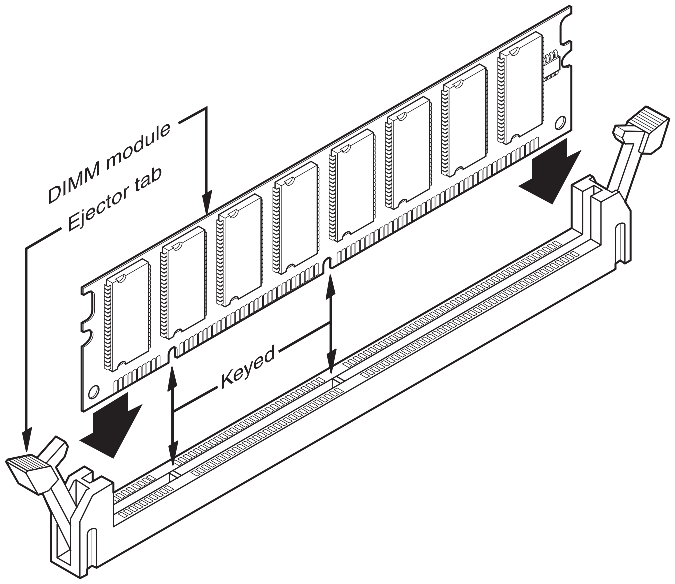


Figure 6.16 DIMM keys match the protrusions in the DIMM sockets. DDR DIMM and RIMM keys are similar but not exactly the same.

9. Push down on the DIMM or RIMM until the ejector tabs lock into place in the notch on the side of the module. It's important that you not force the module into the socket. If the module does not slip easily into the slot and then snap into place, it is probably not oriented or aligned correctly. Forcing the module could break it or the socket. When installing RIMMs, you need to fill any empty RIMM sockets with continuity modules. Refer to Figure 6.14 for details.

10. Replace any cables or wires you disconnected.
11. Close the system, reconnect the power cable, and turn on the PC.

After adding the memory and putting the system back together, you might have to run the BIOS Setup and resave with the new amount of memory being reported. Most newer systems automatically detect the new amount of memory and reconfigure the BIOS Setup settings for you. Most newer systems also don't require setting any jumpers or switches on the motherboard to configure them for your new memory.

After configuring your system to work properly with the additional memory, you might want to run a memory-diagnostics program to ensure that the new memory works properly. Some are run automatically for you. At least two and sometimes three memory-diagnostic programs are available for all systems. In order of accuracy, these programs are as follows:

- POST (power-on self test)
- Disk-based advanced diagnostics software

The POST is used every time you power up the system.

Many additional diagnostics programs are available from aftermarket utility software companies.

Installing SIMM Modules

SIMM memory is oriented by a notch on one side of the module that is not present on the other side, as shown in Figure 6.17. The socket has a protrusion that must fit into this notched area on one side of the module. This protrusion makes installing a SIMM backward impossible unless you break the connector or the module. Figure 6.18 details the notch and locking clip.

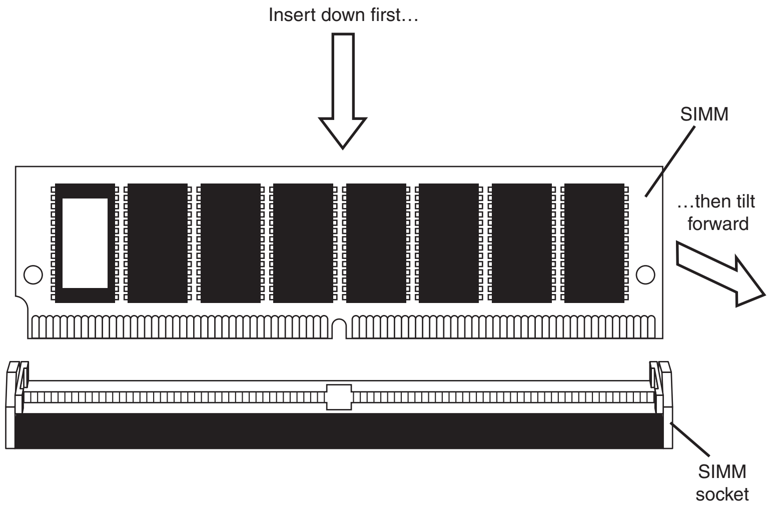


Figure 6.17 The notch on this SIMM is shown on the left side. Insert the SIMM at a 45° angle and then tilt it forward until the locking clips snap into place.

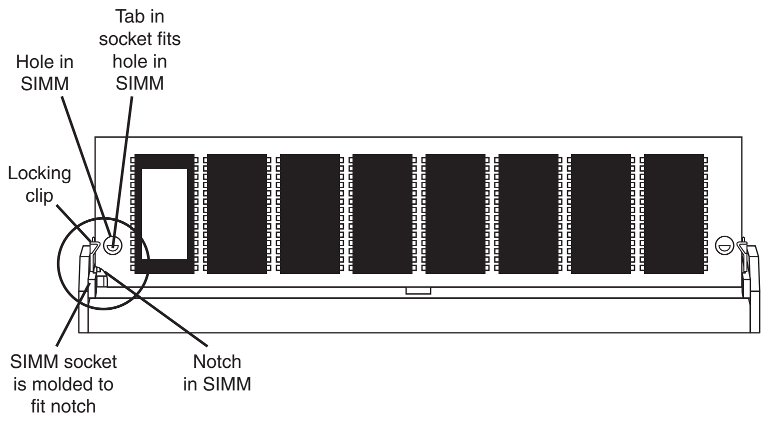

Tab in socket fits hole in SIMM
Hole in SIMM
Locking clip
SIMM socket is molded to fit notch
Notch in SIMM

Figure 6.18 This figure shows the SIMM inserted in the socket with the notch aligned, the locking clip locked, and the hole in the SIMM aligned with the tab in the socket.

Troubleshooting Memory

Memory problems can be difficult to troubleshoot. For one thing, computer memory is still mysterious to people because it is a kind of "virtual" thing that can be hard to grasp. The other difficulty is that memory problems can be intermittent and often look like problems with other areas of the system, even software. This section shows simple troubleshooting steps you can perform if you suspect you are having a memory problem.

To troubleshoot memory, you first need some memory-diagnostics testing programs. You already have several and might not know it. Every motherboard BIOS has a memory diagnostic in the POST that runs when you first turn on the system. In most cases, you also receive a memory diagnostic on a utility disk that came with your system. Many commercial diagnostics programs are on the market, and almost all of them include memory tests.

When the POST runs, it not only tests memory, but also counts it. The count is compared to the amount counted the last time BIOS Setup was run; if it is different, an error message is issued. As the POST runs, it writes a pattern of data to all the memory locations in the system and reads that pattern back to verify that the memory works. If any failure is detected, you see or hear a message. Audio messages (beeping) are used for critical or "fatal" errors that occur in areas important for the system's operation. If the system can access enough memory to at least allow video to function, you see error messages instead of hearing beep codes.

See the disc accompanying this book for detailed listings of the BIOS beep and other error codes, which are specific to the type of BIOS you have. These BIOS codes are found in the Technical Reference section of the disc in printable PDF format for your convenience. For example, most Intel motherboards use the Phoenix BIOS. Several beep codes are used in that BIOS to indicate fatal memory errors.

If your system makes it through the POST with no memory error indications, there might not be a hardware memory problem, or the POST might not be able to detect the problem. Intermittent memory errors are often not detected during the POST, and other subtle hardware defects can be hard for the POST to catch. The POST is designed to run quickly, so the testing is not nearly as thorough as it could be. That is why you often have to boot from a standalone diagnostic disk and run a true hardware diagnostic to do more extensive memory testing. These types of tests can be run continuously and be left running for days if necessary to hunt down an elusive intermittent defect.

Fortunately several excellent memory test programs are available for free download. The ones I recommend include

- **Microsoft Windows Memory Diagnostic**—http://oca.microsoft.com/en/windiag.asp
- **DocMemory Diagnostic**—http://www.simmtester.com/page/products/doc/docinfo.asp
- **Memtest86**—http://www.memtest86.com

Not only are all these free, but they are available in a bootable CD format, which means you don't have to install any software on the system you are testing. The bootable format is actually required in a way since Windows and other OSs prevent the direct access to memory and other hardware required for testing. These programs use algorithms that write different types of patterns to all of the memory in the system, testing every bit to ensure it reads and writes properly. They also turn off the processor cache in order to ensure direct testing of the modules and not the cache. Some, such as Windows Memory Diagnostic, will even indicate the module that is failing should an error be encountered.

Still, even these programs do only pass/fail type testing; that is, all they can do is write patterns to memory and read them back. They can't determine how close the memory is to failing—only whether it worked. For the highest level of testing, the best thing to have is a dedicated memory test machine, usually called a *module tester*. These devices enable you to insert a module and test it thoroughly at a variety of speeds, voltages, and timings to let you know for certain whether the memory is good or bad. Versions of these testers are available to handle all types of memory modules. I have defective modules, for example, that work in some systems (slower ones) but not others. What I mean is that the same memory test program fails the module in one machine but passes it in another. In the module tester, it is always identified as bad right down to the individual bit, and it even tells me the actual speed of the device, not just its rating. Companies that offer memory module testers include Tanisys (www.tanisys.com), CST (www.simmtester.com), and Innoventions (www.memorytest.com). They can be expensive, but for a professional in the PC repair business, using one of these module testers can save time and money in the long run.

After your operating system is running, memory errors can still occur, typically identified by error messages you might receive. Here are the most common:

- **Parity errors**—Indicates that the parity-checking circuitry on the motherboard has detected a change in memory since the data was originally stored. (See the "How Parity Checking Works" section earlier in this chapter.)
- **General or global protection faults**—A general-purpose error indicating that a program has been corrupted in memory, usually resulting in immediate termination of the application. This can also be caused by buggy or faulty programs.
- **Fatal exception errors**—Error codes returned by a program when an illegal instruction has been encountered, invalid data or code has been accessed, or the privilege level of an operation is invalid.
- **Divide error**—A general-purpose error indicating that a division by 0 was attempted or the result of an operation does not fit in the destination register.

If you are encountering these errors, they could be caused by defective or improperly configured memory, but they can also be caused by software bugs (especially drivers), bad power supplies, static discharges, close proximity radio transmitters, timing problems, and more.

If you suspect the problems are caused by memory, there are ways to test the memory to determine whether that is the problem. Most of this testing involves running one or more memory test programs.

I am amazed that most people make a critical mistake when they run memory test software. The biggest problem I see is that people run memory tests with the system caches enabled. This effectively invalidates memory testing because most systems have what is called a *write-back cache*. This means that data written to main memory is first written to the cache. Because a memory test program first writes data and then immediately reads it back, the data is read back from the cache, not the main memory. It makes the memory test program run very quickly, but all you tested was the cache. The bottom line is that if you test memory with the cache enabled, you aren't really writing to the SIMM/DIMMs, but only to the cache. Before you run any memory test programs, be sure your cache is disabled. The system will run very slowly when you do this, and the memory test will take much longer to complete, but you will be testing your actual RAM, not the cache.

The following steps enable you to effectively test and troubleshoot your system RAM. Figure 6.19 provides a boiled-down procedure to help you step through the process quickly.

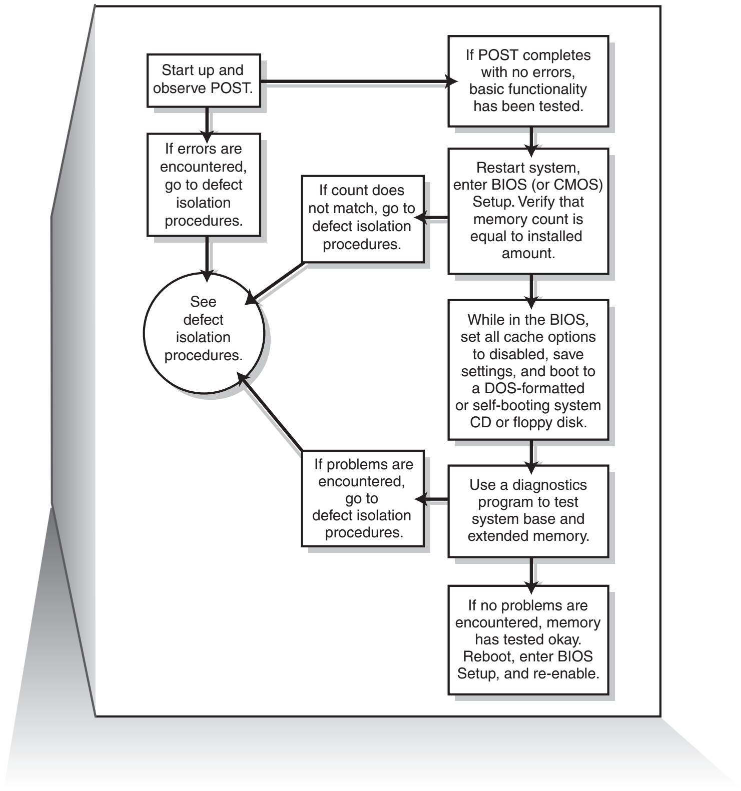
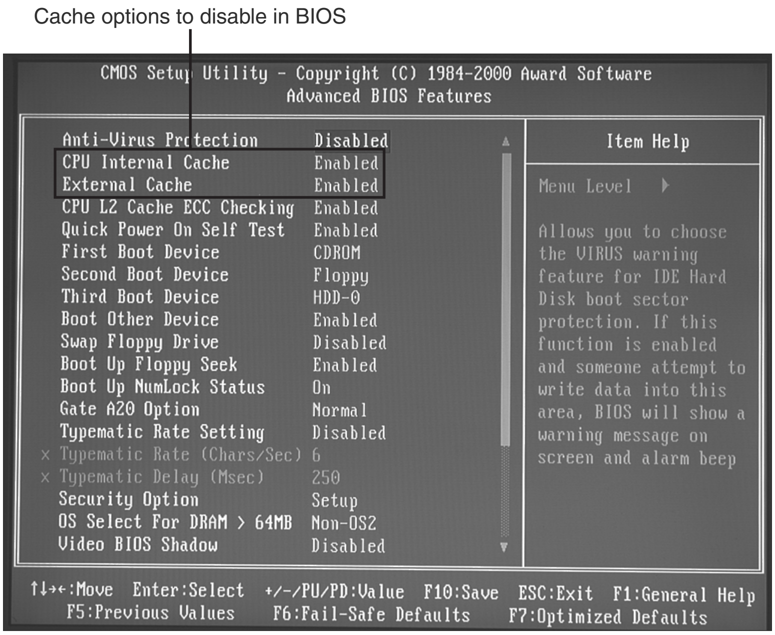

Start up and observe POST.
If POST completes with no errors, basic functionality has been tested.
If errors are encountered, go to defect isolation procedures.
Restart system, enter BIOS (or CMOS) Setup. Verify that memory count is equal to installed amount.
If count does not match, go to defect isolation procedures.
See defect isolation procedures.
While in the BIOS, set all cache options to disabled, save settings, and boot to a DOS-formatted or self-booting system CD or floppy disk.
If problems are encountered, go to defect isolation procedures.
Use a diagnostics program to test system base and extended memory.
If no problems are encountered, memory has tested okay. Reboot, enter BIOS Setup, and re-enable.

Figure 6.19 Testing and troubleshooting memory.

First, let's cover the memory-testing and troubleshooting procedures.

1. Power up the system and observe the POST. If the POST completes with no errors, basic memory functionality has been tested. If errors are encountered, go to the defect isolation procedures.
2. Restart the system and then enter your BIOS (or CMOS) Setup. In most systems, this is done by pressing the Del or F2 key during the POST but before the boot process begins (see your system or motherboard documentation for details). Once in BIOS Setup, verify that the memory count is equal to the amount that has been installed. If the count does not match what has been installed, go to the defect isolation procedures.
3. Find the BIOS Setup options for cache and then set all cache options to disabled. Figure 6.20 shows a typical Advanced BIOS Features menu with the cache options highlighted. Save the settings and reboot to a DOS-formatted or self-booting CD or floppy disk containing the diagnostics program of your choice. If your system came with a diagnostics disk, you can use that, or you can use one of the many commercial PC diagnostics programs on the market, such as PC-Technician by Windsor Technologies, Norton System Works by Symantec, or Doc Memory from SIMMTester.



```
CMOS Setup Utility - Copyright (C) 1984-2000 Award Software
                    Advanced BIOS Features

  Anti-Virus Protection      Disabled        Item Help
  CPU Internal Cache         Enabled
  External Cache             Enabled         Menu Level  ▶
  CPU L2 Cache ECC Checking  Enabled
  Quick Power On Self Test   Enabled         Allows you to choose
  First Boot Device          CDROM           the VIRUS warning
  Second Boot Device         Floppy          feature for IDE Hard
  Third Boot Device          HDD-0           Disk boot sector
  Boot Other Device          Enabled         protection. If this
  Swap Floppy Drive          Disabled        function is enabled
  Boot Up Floppy Seek        Enabled         and someone attempt to
  Boot Up NumLock Status     On              write data into this
  Gate A20 Option            Normal          area, BIOS will show a
  Typematic Rate Setting     Disabled        warning message on
x Typematic Rate (Chars/Sec) 6               screen and alarm beep
x Typematic Delay (Msec)     250
  Security Option            Setup
  OS Select For DRAM > 64MB  Non-OS2
  Video BIOS Shadow          Disabled

↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
   F5:Previous Values    F6:Fail-Safe Defaults    F7:Optimized Defaults
```

Figure 6.20 The CPU Internal (L1) and External (L2) caches must be disabled in the system BIOS setup before you test system memory; otherwise, your test results will be inaccurate.

4. Follow the instructions that came with your diagnostic program to have it test the system base and extended memory. Most programs have a mode that enables them to loop the test—that is, to run it continuously, which is great for finding intermittent problems. If the program encounters a memory error, proceed to the defect isolation procedures.
5. If no errors are encountered in the POST or in the more comprehensive memory diagnostic, your memory has tested okay in hardware. Be sure at this point to reboot the system, enter the

BIOS Setup, and re-enable the cache. The system will run very slowly until the cache is turned back on.

6. If you are having memory problems yet the memory still tests okay, you might have a problem undetectable by simple pass/fail testing, or your problems could be caused by software or one of many other defects or problems in your system. You might want to bring the memory to a SIMM/DIMM tester for a more accurate analysis. Most PC repair shops have such a tester. I would also check the software (especially drivers, which might need updating), power supply, and system environment for problems such as static, radio transmitters, and so forth.

Memory Defect Isolation Procedures

To use these steps, I am assuming you have identified an actual memory problem that is being reported by the POST or disk-based memory diagnostics. If this is the case, see the following steps and Figure 6.21 for the steps to identify or isolate which SIMM or DIMM in the system is causing the problem.

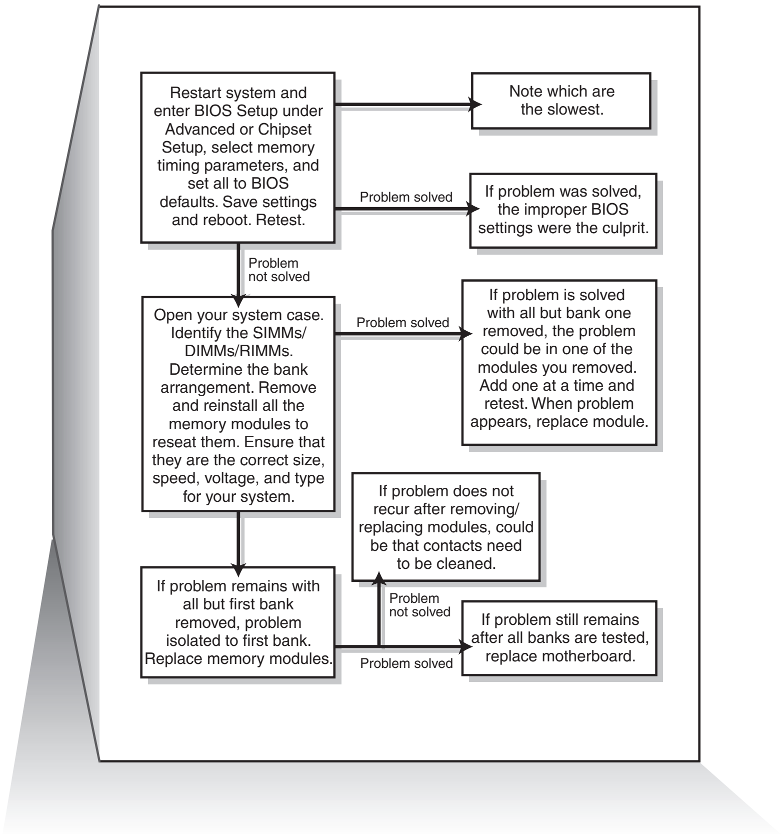
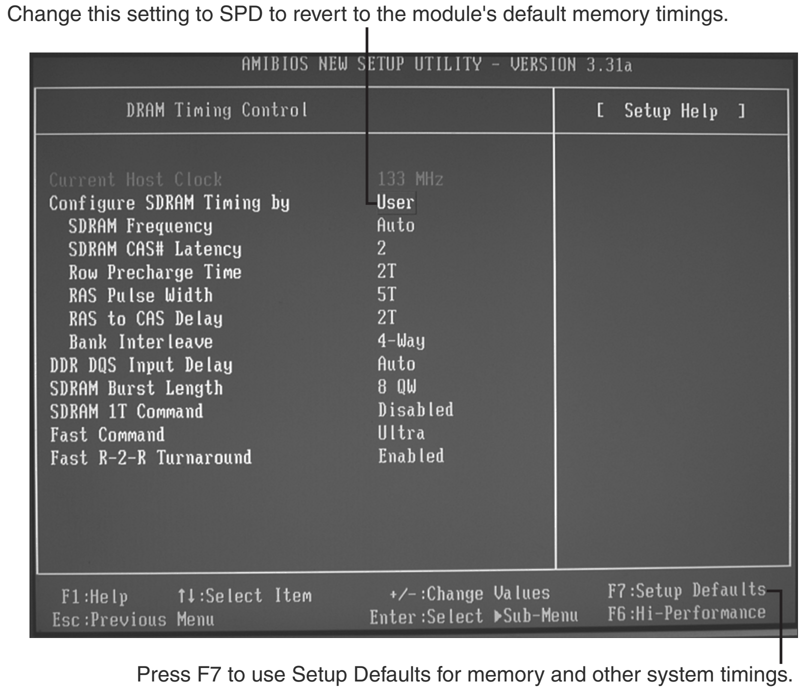

Restart system and enter BIOS Setup under Advanced or Chipset Setup, select memory timing parameters, and set all to BIOS defaults. Save settings and reboot. Retest.
Note which are the slowest.
Problem solved
If problem was solved, the improper BIOS settings were the culprit.
Problem not solved
Open your system case. Identify the SIMMs/DIMMs/RIMMs. Determine the bank arrangement. Remove and reinstall all the memory modules to reseat them. Ensure that they are the correct size, speed, voltage, and type for your system.
Problem solved
If problem is solved with all but bank one removed, the problem could be in one of the modules you removed. Add one at a time and retest. When problem appears, replace module.
If problem does not recur after removing/replacing modules, could be that contacts need to be cleaned.
Problem not solved
If problem remains with all but first bank removed, problem isolated to first bank. Replace memory modules.
Problem solved
If problem still remains after all banks are tested, replace motherboard.

Figure 6.21 Follow these steps if you are still encountering memory errors after completing the steps in Figure 6.19.

1. Restart the system and enter the BIOS Setup. Under a menu usually called Advanced or Chipset Setup might be memory timing parameters. Select BIOS or Setup defaults, which are usually the slowest settings. If the memory timings have been manually set, as shown in Figure 6.20, reset the memory configuration to By SPD.
2. Save the settings, reboot, and retest using the testing and troubleshooting procedures listed earlier. If the problem has been solved, improper BIOS settings were the problem. If the problem remains, you likely do have defective memory, so continue to the next step.
3. Open the system for physical access to the SIMM/DIMM/RIMMs on the motherboard. Identify the bank arrangement in the system. Using the manual or the legend silk-screened on the motherboard, identify which modules correspond to which banks. Remember that if you are testing a dual-channel system, you must be sure you remove both Channel A and Channel B modules in the same logical bank.
4. Remove all the memory except the first bank and then retest using the troubleshooting and testing procedures listed earlier (see Figure 6.22). If the problem remains with all but the first bank removed, the problem has been isolated to the first bank, which must be replaced.



Change this setting to SPD to revert to the module's default memory timings.

```
                  AMIBIOS NEW SETUP UTILITY - VERSION 3.31a

        DRAM Timing Control                         [  Setup Help  ]

  Current Host Clock                 133 MHz
  Configure SDRAM Timing by          User
    SDRAM Frequency                  Auto
    SDRAM CAS# Latency               2
    Row Precharge Time               2T
    RAS Pulse Width                  5T
    RAS to CAS Delay                 2T
    Bank Interleave                  4-Way
  DDR DQS Input Delay                Auto
  SDRAM Burst Length                 8 QW
  SDRAM 1T Command                   Disabled
  Fast Command                       Ultra
  Fast R-2-R Turnaround              Enabled

  F1:Help      ↑↓:Select Item       +/-:Change Values       F7:Setup Defaults
  Esc:Previous Menu                 Enter:Select ▶Sub-Menu  F6:Hi-Performance
```

Press F7 to use Setup Defaults for memory and other system timings.

Figure 6.22 This system is using user-defined memory timings, which could cause the memory to be unstable.

5. Replace the memory in the first bank (preferably with known good spare modules, but you can also swap in others that you have removed) and then retest. If the problem still remains after testing all the memory banks (and finding them all to be working properly), it is likely the motherboard itself is bad (probably one of the memory sockets). Replace the motherboard and retest.

6. At this point, the first (or previous) bank has tested good, so the problem must be in the remaining modules that have been temporarily removed. Install the next bank of memory and retest. If the problem resurfaces now, the memory in that bank is defective. Continue testing each bank until you find the defective module.
7. Repeat the preceding step until all remaining banks of memory are installed and have been tested. If the problem has not resurfaced after you have removed and reinstalled all the memory, the problem was likely intermittent or caused by poor conduction on the memory contacts. Often simply removing and replacing memory can resolve problems because of the self-cleaning action between the module and the socket during removal and reinstallation.

The System Logical Memory Layout

The original PC had a total of 1MB of addressable memory, and the top 384KB of that was reserved for use by the system. Placing this reserved space at the top (between 640KB and 1,024KB, instead of at the bottom, between 0KB and 640KB) led to what is often called the *conventional memory barrier*. The constant pressures on system and peripheral manufacturers to maintain compatibility by never breaking from the original memory scheme of the first PC has resulted in a system memory structure that is (to put it kindly) a mess. Almost two decades after the first PC was introduced, even the newest Core 2 Extreme or Athlon 64 x2–based systems are limited in many important ways by the memory map of the first PCs.

What we end up with is a PC with a split personality. There are two primary modes of operation that are very different from each other. The original PC used an Intel 8088 processor that could run only 16-bit instructions or code, which ran in what was called the *real mode* of the processor. These early processors had only enough address lines to access up to 1MB of memory, and the last 384KB of that was reserved for use by the video card as video RAM, other adapters (for on-card ROM BIOS or RAM buffers), and finally the motherboard ROM BIOS.

The 286 processor brought more address lines, enough to allow up to 16MB of RAM to be used, and a new mode called protected mode that you had to be in to use it. Unfortunately, all the operating system's software at the time was designed to run only within the first 1MB, so extender programs were added and other tricks performed to eventually allow DOS and Windows 3.x access up to the first 16MB. One area of confusion was that RAM was now noncontiguous; that is, the operating system could use the first 640KB and the last 15MB, but not the 384KB of system reserved area that sat in between.

When Intel released the first 32-bit processor in 1985 (the 386DX), the memory architecture of the system changed dramatically. There were now enough address lines for the processor to use 4GB of memory, but this was accessible only in a new 32-bit protected mode in which only 32-bit instructions or code could run. This mode was designed for newer, more advanced operating systems, such as Windows 9x, NT, 2000, XP, Vista OS/2, Linux, UNIX, and so forth. With the 386 came a whole new memory architecture in which this new 32-bit software could run. Unfortunately, it took 10 years for the mainstream user to upgrade to 32-bit operating systems and applications, which shows how stubborn we are! From a software instruction perspective, all the 32-bit processors since the 386 are really just faster versions of the same. Other than the more recent additions of MMX and SSE (or AMD's 3DNow) instructions to the processor, for all intents and purposes, a Pentium 4 or Athlon is just a "turbo" 386. Even more advanced server-oriented, 64-bit instruction set processors such as the Intel Itanium and AMD Opteron processors fit into this category because they can also run 32-bit software.

The real problem now is that the 32-bit processors have two distinctly different modes, with different memory architectures in each. For backward compatibility, you could still run the 32-bit processors in real mode, but only 16-bit software could run in that mode, and such software could access only the first 1MB or 16MB, depending on how it was written. For example, 16-bit drivers could load into and access only the first 1MB. Also, it is worth noting that the system ROM BIOS, including the POST, BIOS configuration, boot code, and all the internal drivers, is virtually all 16-bit software. This is because all Intel-compatible PC processors begin operation in 16-bit real mode when they are powered on. When a 32-bit operating system loads, it is that operating system code that instructs the processor to switch into 32-bit protected mode.

When an operating system such as Windows is loaded, the processor is switched into 32-bit protected mode early in the loading sequence. Then, 32-bit drivers for all the hardware can be loaded, and then the rest of the operating system can load. In 32-bit protected mode, the operating systems and applications can access all the memory in the system, up to the maximum limit of the processor (64GB for most of the Pentium II and later chips). Similarly, on a 64-bit operating system, the system switches into 64-bit protected mode early in the boot process and loads 64-bit drivers, followed by the remainder of the operating system.

Unfortunately, one problem with protected mode is just that: It is protected. The name comes from the fact that only driver programs are allowed to talk directly to the hardware in this mode; programs loaded by the operating system, such as by clicking an icon in Windows, are not allowed to access memory or other hardware directly. This protection is provided so that a single program can't crash the machine by doing something illegal. You might have seen the error message in Windows indicating this, and that the program will be shut down.

Diagnostics software by nature must talk to the hardware directly. This means that little intensive diagnostics testing can be done while a protected mode operating system such as Windows 9x, NT, 2000, XP, Vista, Linux, and so forth is running. For system testing, you usually have to boot from a DOS or self-booting floppy or CD or interrupt the loading of Windows (press the F8 key when Starting Windows appears during the boot process) and select Command Prompt Only, which boots you into DOS. In Windows 9x (but not Me), you can execute a shutdown and select Restart the Computer in MS-DOS mode. Many of the higher-end hardware diagnostics programs include their own special limited 16-bit operating systems so they can more easily access memory areas even DOS would use. With Windows 2000 and XP, you can format a floppy disk with MS-DOS startup files by selecting that option from the Format menu in My Computer. In Windows Vista, use the Format menu in Computer.

For example, when you boot from a Windows 9x startup disk, you are running 16-bit DOS, and if you want to access your CD-ROM drive to install Windows, you must load a 16-bit CD-ROM driver program from that disk as well. In this mode, you can do things such as partition and format your hard disk, install Windows, and test the system completely. OEM versions of Windows 98 and all versions of Windows Me and newer come on bootable CDs, so if your system supports booting from a CD-ROM drive, be sure to set that option in the BIOS Setup to eliminate the need for a formatted floppy disk.

The point of all this is that although you might not be running DOS very much these days, at least for system configuration and installation, as well as for high-level hardware diagnostics testing, data recovery, and so forth, you might still have to boot to a 16-bit OS occasionally. When you are in that mode, the system's architecture changes, less memory is accessible, and some of the software you are running (16-bit drivers and most application code) must fight over the first 1MB or even 640KB for space.

System memory areas, including the 384KB at the top of the first megabyte, which is used for video, adapter BIOS, and motherboard BIOS, as well as the remaining extended memory, are all part of the PC hardware design. They exist whether you are running 16-bit or 32/64-bit software; however, the limitations on their use in 16-bit (real) mode are much more severe. Because most people run 32-bit operating systems, such as Windows 9x, 2000, XP, Vista, Linux, and so on (or 64-bit operating systems such as Windows XP, Vista, or Linux), these operating systems automatically manage the use of RAM, meaning you don't have to interact with and manage this memory yourself as you often did with the 16-bit operating systems.

If you are using DOS or 16-bit Windows operating systems or are using legacy hardware that uses the ISA, EISA, MCA, or VL-Bus architectures, you need to have a detailed understanding of the logical memory map to avoid conflicts between devices and to optimize the performance of your system.

To learn more about the logical memory map and how to optimize its use, see Chapter 6 of *Upgrading and Repairing PCs, 17th Edition*, available in electronic form on the disc packaged with this book.