

---

# Chapter 6

## Platform Optimization Techniques and Directions

---

This chapter discusses how contemporary platform architectures are changing to track evolving uses of the PC platform and to continually improve system performance. The discussion relies upon and ties together the platform standards, initiatives, and technology components presented in Chapters 4 and 5.

About half of the chapter is devoted to a tutorial on platform optimization, including a survey of techniques and selected examples. Going beyond the obvious fundamentals of increasing operating rates and shifting to faster technology, the survey is a compilation of basic optimization concepts applicable to contemporary platform designs. The survey is followed by an examination of how the optimization basics are applied in contemporary 3D graphics platforms.

The second half of the chapter discusses the directions in which the PC platform is evolving and the factors presently driving the changes. Attention is especially given to the dynamic developments in platform content and connectivity. The sections on platform directions include a study of contemporary design issues, examines a planned progression of changes to the platform architecture, and looks at forecasts for the near and long-term future.

### ROAD MAP OF CHAPTER 6

Section	Audience
Survey Of Platform Performance Optimizations	Students and those practitioners, managers and faculty who want to study outside their area of specialization

ROAD MAP OF CHAPTER 6

Section	Audience
Platform Directions	Students and those unfamiliar with the PC industry’s vision for future platforms and connected computing
<i>The following more detailed subsections:</i>  Performance Optimization Specifics in a Contemporary 3D Graphics Platform	All
Geometry, Rendering, and Display—The 3D Graphics Pipeline	All

SURVEY OF PLATFORM  
PERFORMANCE  
OPTIMIZATIONS

IMPROVING DATA MOVEMENT AND MANIPULATION IN THE PLATFORM

The essence of computing is data movement and manipulation. Bit patterns are moved (routed between source and destination resources) and operated on at a low level to implement functions at far-removed levels of abstraction. A universe of data movement and manipulation solutions exists for every computing problem. System performance optimization is the pursuit of those system solutions that enable getting more work done in less time for a given level of investment.

The system computing environment includes the operating system, applications, and hardware architecture. Changes in the behavior of each of these components, as well as the manner in which they interact, can have a dramatic impact on performance. Here we will emphasize how platform hardware architecture affects system performance.

Different system hardware-architecture solutions have variations in topology, system resources, and methods of operation. These system resources include the processor, a hierarchically organized memory subsystem, various I/O peripherals, buses and other interconnects, bus bridges, and resource controllers. Within each resource there are possible variations in organization and technology.

Performance optimization generally focuses on improving the speed and efficiency with which the system carries out data movements and data manipulations to accomplish high-level functions. The possible solution variations at the hardware-architecture and resource level offer different speeds and efficiencies for data movements and manipulations, leading to different levels of performance.

In the absence of radically different system-level algorithms, efficiency improvements generally require increased sophistication in the management of expensive system resources. This includes minimizing the number of required data transfers between resources and minimizing unnecessary resource use while maximizing productive resource use. Speed improvements generally involve increasing the rate of required data transfers and resource operations. Improvements to either speed or efficiency may require additional system resources. The additional resources may take the form of new types of system resources or merely augmentation (e.g., increasing bit widths) of existing resources.

Cost and other system constraints are frequently at odds with improving system speed and efficiency. In particular, the processor is often called upon to perform operations that, from strictly a performance perspective, might best be carried out by a dedicated data pump or specialized coprocessor. For a particular target market, however, the increased system cost for the additional system resources may compromise sales in an unacceptable manner.

### *Platform Focus Areas for Performance Optimization*

Platform focus areas for optimizing performance include meeting the data requirements of applications, the appropriate use and configuration of system resources, cautious management of data movement or bus traffic between system resources, and specification of the overall system architecture and its operation. Meeting data requirements of applications primarily involves the ability to meet maximum latency delays and deliver minimum sustained throughput rates. System resources that should be examined closely for optimal operation include the CPU; buses, ports, and sideband signals; data-staging storage devices (including buffers, caches, and queues); bridges/controllers; peripherals; and memory. The specification of the system architecture and operation is done carefully to achieve performance that is well apportioned and tuned. In this section we will overview optimization issues for required data movements, for overall system design, and specific optimizations for buses, data-staging storage devices, and bridges.

### Application Processing

Application processing executes data transfers among the CPU, memory arrays, and disk files, in conjunction with the maintenance and manipulation of data structures and data streams, searching, formatting, and conditional control. Application developers can minimize unnecessary data movements through good programming discipline, algorithm selection and optimization, and the use of optimizing compilers.

*Data movements and Data manipulations are examined in more detail in conjunction with Figure 6.3 on page 486 & Figure 6.4 on page 487, later in this chapter.*

*processor recognition**CPUID instruction*

Application developers should also use *processor recognition* techniques to identify particular processors. The primary tool of processor recognition is the *CPUID instruction*. This instruction returns processor information including the vendor, model number, revision, features, and name. Dynamically determined knowledge of the processor being used permits the application to exploit processor-specific optimizations without penalizing or preventing application execution on other processors. Whenever available, the use of specialty instruction sets for floating-point, multimedia, and 3D graphics delivers increased performance on corresponding specialty workloads, while reducing system bus traffic.

## REPORTS ON CD-ROM



## Software Customization for Performance Optimization

If a x86 processor meets Microsoft's *Designed for Windows* certification requirements and is compatible with the physical and electrical features of a motherboard, the processor and motherboard should be functional for generic Microsoft Windows software without mandatory customization. However, customization for a specific processor is necessary to realize *any* benefits of instruction set extensions. Customization is also necessary to optimize performance for each processor's unique microarchitecture.

We have included on the CD-ROM the K6 documents listed below, which are useful to customizing BIOS, system, and application software. Comparable documents will exist for other processors. The first two provide information particular to BIOS design. The remaining documents (two on processor recognition, two on code optimization, and two on instruction set extensions) are applicable to all types of software development.

- *AMD K86 Family BIOS and Software Tools*, Developers Guide, Document 21062E/0-June 1997.
- *AMD K86 Family BIOS Design*, Document 21329, December 1997
- *AMD Processor Recognition*, Publication #20734, January 1998.
- *Processor Recognition Code Sample*, Publication #21035.
- *AMD-K6 3D Processor Code Optimization*, Publication #21924, February 1998.
- *AMD-K6 MMX Enhanced Processor x86 Code Optimization*, Publication #21828, August 1997.
- *AMD-K6 MMX Enhanced Processor Multimedia Technology*, Publication #20726C/0-June 1997.
- *AMD-3D Technology Manual*, Publication #21928, February 1998.

## OVERALL SYSTEM ARCHITECTURE PERFORMANCE OPTIMIZATION

To optimize platform performance, attention needs to be given to the platform's hardware resources, topology, resource management policies, and performance features and management policies internal to each resource. This section presents a compilation of general and specific performance optimization guiding basics and brainstorming ideas for each of these areas. The format of the presentation is to introduce a set of performance optimization basics in a definition box. Below each box is a discussion that expands on how the particular boxed set of rules is applicable in platforms.

The rules given are for *performance* optimization. In PC platforms however, often something else is being optimized. For example, the USB and IEEE 1394 buses discussed in Chapter 4 are directly contrary to many of the performance optimization basics that we will present for platform-level buses. These buses seek to combine devices on the same bus, reduce bus widths, and increase competition for the bus, all of which negatively impact performance. What is happening is that these buses provide good performance, but are not optimized solely for performance. They are optimized for attaching large numbers of peripherals, requiring minimal system resources per peripheral, and having extreme ease of configuration.

*Optimizing for performance is not always the primary goal, but it is usually an important factor in any optimization. The rules given in this chapter are thus useful for other optimizations in that you can anticipate how changes motivated by other concerns will likely affect performance.*

### PLATFORM PERFORMANCE OPTIMIZATION BASICS

#### Stipulations and Caveats

1. The rules attempt to optimize for *performance* only
2. You must use judgement, common sense, and an engaged mind to successfully apply the rules
3. You must determine when the application of a rule has reached a point of diminishing return
4. You must determine if a rule does not apply due to situational design constraints
5. You must be alert for undesirable side effects
6. Generally, you must trade off cost, power, and size against performance
7. You must weigh the impact of a change in terms of its costs and benefits
8. There will be situations in which the best choice is contrary to one or more of the rules

General Platform-Level Performance Optimization Principles

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>a</sup>
<p>General Platform Principles</p> <ol style="list-style-type: none"><li>1. Seek to increase operating rates.</li><li>2. Seek to increase resource operating-widths.</li><li>3. Seek to increase the efficient use of high-demand resources.</li><li>4. Seek to reduce the number of subsystems that process each task.</li><li>5. Seek to keep expensive resources continuously productive.</li></ol>

<sup>a</sup>. See Stipulations and Caveats on page 467.

The first optimization principle is fundamental. Intuitively, overall platform performance is proportional to the useful work per operating cycle times the operating rate (in cycles per second). It's obvious that work performed per unit-time goes up if you can make the platform go faster.

The second optimization principle is also fundamental, known in different form as the time-space trade-off. Loosely, this states that more bits operating in parallel will get work done faster than fewer bits operating sequentially. Thus overall platform performance, being proportional to work per unit-time, goes up if you can increase the useful work per operating cycle. For some tasks simply replicating operating resources can do this. At the lowest level this includes increasing resource bit widths.

The third general optimization principle suggests that the platform architect should, for each platform process that uses a system resource, seek solutions that decrease unnecessary use. Such solutions are instrumental in improving the performance of other processes that compete for access to the resource under consideration, and thereby generally improve the performance of the system as a whole.

As applied to buses, this means to attempt to eliminate or reduce the number of bus transactions and transfers required. Discovery of such higher efficiency bus-using processes tends to reduce process stalls and bus latencies due to the bus being busy when needed by a second process. It also reduces the need to purchase additional bus bandwidth.

Applying this principle may require the platform architect to increase widths, use data staging, divert specialty traffic to dedicated buses, use command-driven dedicated accelerators or bus-masters on isolated sub-segments, improve bus protocols, or use data compression. Such techniques are discussed throughout this section.

## DEFINITIONS

## Stalls and Latencies

In the platform context of the present discussion, a *stall* is a platform event in which an executing task is forced to be idle unproductively, while some crucial enabling condition is unrealizable or some essential resource is unavailable to the task.

At the platform level, *latency* describes the duration of a stall. In contemporary synchronous designs, platform-level tasks proceed with execution only at time intervals defined by the operating cycle of the required function-units. In turn, function-unit cycles are integral multiples of the platform's clock cycle. Depending on the platform's dynamic workload and the design of the platform, the latency of a stall may be momentary, no more than one function-unit cycle, or may be much larger. Thus, *stalls* are incidents of unproductive idling, while *latencies* are a measure of how much time is lost.

*Latency* more broadly refers to the duration of any delay. At the device or device-interface level, latency describes electrical or mechanical delays that are measured in small fractions of a second, which act to limit operating rates.

The fourth general optimization principle suggests that the platform architect should, for each subsystem, seek solutions that keep task processing isolated within each subsystem and minimize the involvement of the CPU and other subsystems. Autonomous bus-masters, DMA, and special-purpose accelerators are examples of techniques to keep task processing localized and have the added benefit that they off-load the CPU. Dedicated signal paths, such as the AGP between the North-Bridge and the graphics accelerator, and a "video capture" port that couples streaming video directly into the video accelerator, are examples of techniques that keep data off of the general system buses and localized within a single subsystem. Such methods improve system bus bandwidth and processor availability, reducing the need to pay for greater bus bandwidth and processing power, and help to increase overall system concurrency and multi-tasking performance.

The fifth general optimization principle suggests that the platform architect should, for each subsystem, seek solutions that improve the utilization of expensive system resources. Ideally, the major system resources should be kept continually busy. Thus, the platform architect should look for ways to increase concurrency throughout the system, primarily by orchestrating overlapping processes in multiple independent subsystems. This is enabled by selective isolation of the various platform subsystems,

explicit pipelining, and by out-of-order processing of younger short tasks that can independently overlap with, and thereby effectively hide, older long latency tasks. The platform architect should first look for solutions wherein modest additions of inexpensive resources may enable expensive resources to be more fully exploited.

*Specific Performance Optimizations Made at the Platform Level*

Basics for Management of Platform Interconnect Resources

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>b</sup>
<p>Bus Optimization at the Platform Level</p> <ol style="list-style-type: none"><li>1. Seek to reduce unnecessary bus transactions and transfers.</li><li>2. Seek to normally isolate buses capable of concurrent operation.</li><li>3. Seek to increase the number of continuously productive buses.</li><li>4. Seek to reduce simultaneous competition for each bus.</li><li>5. Seek to minimize the use of Programmed I/O (PIO).</li><li>6. Seek to command normally isolated bus-mastering accelerators.</li><li>7. Seek to transfer compressed data.</li><li>8. Seek to keep streaming data out of general-purpose subsystems.</li><li>9. Seek to maximize block sizes and minimize per-block latencies.</li></ol>

<sup>b</sup>. See Stipulations and Caveats on page 467.

The above platform-level optimizations focus on how effectively expensive system resources and subsystems are interconnected with each other. Accordingly, actions taken according to these suggestions often impact the organization and operation of multiple subsystems.

Buses are the resources used to interconnect system components, either via time-share access by multiple system components, or via dedicated access between only two components. The platform architect should seek to eliminate or reduce unnecessary bus transactions and transfers for each bus throughout the platform.

Bus-bridges can be used to couple together two diverse buses or sub-segments of the same bus. While frequently coupling is a necessity, it should not otherwise be done, in order to maximize the isolation of the various subsystems. Isolated subsystems can then process different tasks concurrently, maximizing overall system work.

It is also essential that contention for a bus by multiple requesters is not unduly limiting access to a crucial system resource, resulting in stalls



to the processing of a task needing the resource. If necessary, the platform architect can increase the number of buses capable of concurrent operation, or decrease the number of devices that share each bus. Such measures generally mean the addition of point-to-point busing directly between devices that need coupling, and possibly the addition of physical ports to selected resources.

#### DEFINITION

##### Contention

In the present system optimization context, *contention* means the simultaneous competition for a system resource, in particular, a bus. Such competition results in stalls and increased latencies.

*Arbitration* is the process by which a competing process wins access to a resource that is subject to contention. Processes *request* the resource from an *Arbitrator* that will *grant* access. Arbitration algorithms must take into account factors including maximum allowable latencies, process priorities, and fairness of access.

In a bus's electrical performance context, *contention* means multiple I/O drivers fighting to drive a node to different signal levels. If not controlled this can cause unreliable operation, current spikes, high power dissipation, and premature device failure.

For many tasks, (such as compression, large block transfers, and graphics rendering), dedicated data pumps, controllers, and accelerators should be employed wherever possible. Anytime the processor can be relieved from carrying out a task by a dedicated controller, the available processing power for other tasks is increased. Generally, the system bus traffic is also reduced, increasing the available bus bandwidth for other tasks.

The transfer of compressed data clearly permits available bus bandwidth to be used more effectively, but it requires special processing by both source and destination. Such special processing needs to be integrated into the overall system operation to prevent process stalls. Compression also add latency that would not otherwise exist. The additional latency must be evaluated to determine if the use of compression is viable. Ideally, compression and decompression should be performed in peripheral subsystem dedicated hardware, and not by the processor.

Streaming data types need to be treated specially. Unless processor based DSP is required, or the streamed data is to be otherwise acted on by the processor in the near future, streaming data are preferably coupled via dedicated data paths directly into destination subsystems and kept out of caches. Large data transfers, including streaming video, must be broken into block transfers, if the data are being transferred over general-purpose

buses. Otherwise, large transfer may cause excessive latencies for other devices on the bus.

System-Mediated Large Byte-Count Transfers

System software normally decomposes large data transfers into multiple smaller “block” transfers. This is necessary for a number of reasons, including maximum available main memory buffer sizes and maximum allowable latencies for other tasks that need timely access to system services and resources. However, when a file transfer is decomposed into multiple block transfers, there is invariably some time penalty incurred on each of the blocks. This *per-block latency* may be partly due to overhead delays, arising from device and interface physical limitations and interface protocol requirements, which must be incurred for each block. The per-block latency may also be due to other system events that are allowed to execute before control is returned to the ongoing transfer; or due to sloppily, carelessly, or naively written system software.

Three expressions useful in evaluating the effect of per-block latency are given below. *Effective Data Rate* provides the real throughput at which the large byte-count data transfer is being accomplished. *Realizable Data Rate Fraction* provides the ratio of effective data rate to raw data rate and thus makes explicit what fraction of the raw data rate is being realized. *Rate Degradation* reexpresses the Realizable Data Rate Fraction in terms of a percentage loss.

*per-block latency*

*Effective Data Rate*

*Realizable Data Rate Fraction*

*Rate Degradation*

The above expressions find application in the analysis of networks and wherever else large transfers must be made via smaller block transfers.<sup>75 76</sup>

DEFINITION
<p>Effective Data Rate</p> $\text{Effective data rate} = \frac{\text{Block size}}{\left(\frac{\text{Block size}}{\text{Raw data rate}}\right) + \text{Per-block latency}}$ <p>(sizes are in bytes, rates are in bytes per second, latency is in seconds)</p>

<sup>75</sup> “The ABCs of Capacity Planning,” *LAN Magazine*, Dec. 1996, Miller Freeman Inc.  
<sup>76</sup> “Memory Technology Still Needs to ‘Catch Up,’” *Electronic News*, Feb. 3, 1997, Cahners Publishing Company.

## DEFINITION

## Realizable Data Rate Fraction

$$\text{Realizable Data Rate Fraction} = \left( \frac{\text{Effective Data Rate}}{\text{Raw Data Rate}} \right)$$

$$= \left( 1 + \text{Effective raw data rate} \cdot \frac{\text{Per-block latency}}{\text{Block size}} \right)^{-1}$$

(sizes are in bytes, rates are in bytes per second, latency is in seconds)

## DEFINITION

## Rate Degradation

$$\text{Rate Degradation} = (1 - \text{Realizable Data Rate Fraction}) \times 100$$

$$= \frac{100}{1 + \frac{\text{Block size}}{\text{Realizable Data Rate} \cdot \text{Per-block latency}}}$$

(Rate Degradation is a percentage, sizes are in bytes, rates are in bytes per second, latency is in seconds)

Additional insight can be obtained by normalizing the raw data rate by the ratio of block size to per-block latency (block size / per-block latency) and plotting either the realizable data rate fraction or rate degradation as a function of the normalized raw data rate. Figure 6.1 is a plot of rate degradation. This reveals the usefulness of the (block size / per-block latency) ratio as a transfer rate performance metric.

The (block size/per-block latency) metric acts as a plot “breakpoint,” in that at a normalized raw data rate equal to one, the rate degradation reaches 50%. For raw data rates an order of magnitude or less than the breakpoint, rate degradation is often negligible. For raw data rates within an order of magnitude either above or below the breakpoint, rate degradation increases rapidly with raw data rate and can be quite large. For raw data rates an order of magnitude or greater than the breakpoint, rate degradation is very large.

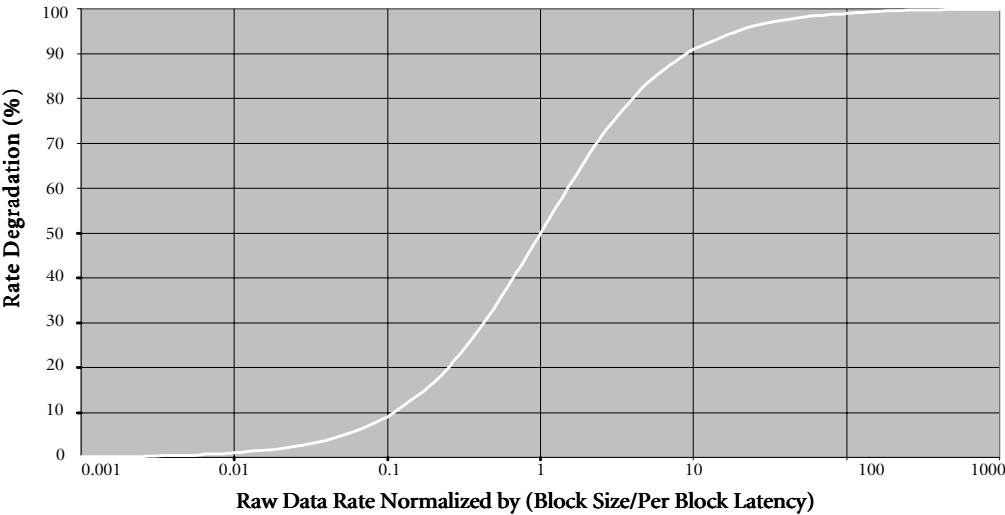


Figure 6.1 RATE DEGRADATION

Design trade-offs must be made to ensure that the effective data rate is sufficient to avoid any user-perceived discontinuities or system malfunctions, while ensuring that other system tasks are not left waiting for excessive periods during block transfers. Per-block latencies, block sizes, and the resulting rate degradation should be tracked closely for all transfers. There is little point in paying for a higher raw data rate device interface if the higher rate is not effectively realizable.

## Platform Data-Staging Basics

### PLATFORM PERFORMANCE OPTIMIZATION BASICS<sup>c</sup>

#### Platform Data-Staging

1. Seek to exploit spatial and temporal locality of reference.
2. Seek to avoid stalls.
3. Seek to reduce latencies.
4. Seek to reduce data traffic between subsystems.
5. Seek to smoothly couple subsystems operating at different rates.
6. Seek to identify optimal block size.

<sup>c</sup> See Stipulations and Caveats on page 467.


The foregoing platform-level optimization suggests that the platform architect apply data-staging techniques throughout the platform architecture, in particular within bus-bridges, and not just in the CPU or CPU-cache subsystem. Buffers and caches may be strategically located within various system resources and managed to fetch or retain data in anticipation of its future use. Such data storage devices generally exploit spatial and temporal locality of reference (for non-streaming data types) to avoid stalls, reduce latencies, and reduce data traffic between subsystems.

Data storage devices, and in particular queues (FIFOs), may also be employed to couple diverse subsystems operating at different or dynamically varying rates, while avoiding stalls in each. Queues permit a source subsystem to deliver data or commands to the queue at a rate that is faster in the short term than the destination subsystem is capable of accepting them. They also permit the destination subsystem to take data or commands from the queue at a rate that is faster in the short term than the source-subsystem is capable of delivering them.

Clearly, if we were dealing with only two subsystems, their long-term throughput is necessarily identical. Queue depth to avoid stalls to either subsystem is easily determined based on the extent to which short-term rate mismatch can occur. For the queues used in platform bridges, there are many subsystems selectively coupling with each other in a manner that is a function of the dynamic workload. Hence, the selection of queue depth is less straightforward.

Whenever data storage locations are transparent within the memory hierarchy, associative techniques are generally incorporated to ensure data coherency. This is true of the data staging techniques, whether implemented by buffers, caches, or queues.

REPORT ON CD-ROM



To learn more about Platform Data Staging Basics, see *Implementation of Write Allocate in the K86 Processors*, Document 21326, June 1997, on the companion CD-ROM.

Basics for Increased Concurrency of Platform Resources

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>d</sup>
<p>Concurrency Optimization</p> <ol style="list-style-type: none"><li>1. Seek to overlap the processing of multiple tasks.</li><li>2. Seek to explicitly pipeline task processing.</li><li>3. Seek to speculatively prefetch sequential data and command streams.</li><li>4. Seek to decompose operations into split transactions and enable increased pipelining or out-of-order processing.</li><li>5. Seek to add independent concurrently operating resources.</li></ol>

<sup>d</sup>. See Stipulations and Caveats on page 467.

The above listed basics are platform-level principles that suggest the platform architect should, for each subsystem, seek to increase concurrency to keep expensive or scarce resources from going idle. This can be done by seeking ways to increase opportunistic overlapped processing of independent or loosely coupled tasks, or through explicit pipelining of tightly coupled tasks. Such techniques may be applied at a variety of levels, including the application or platform-level and the device level. Later in this chapter, a 3D graphics pipeline is used to illustrate how pipelining is used at the platform level.

To further maximize concurrency and the utilization of resources, attention must be paid to minimizing stalls and latencies that interrupt processing overlap and pipelining. This requires that when a task is to be executed, the resources it needs are available, its input data is available, and its output data is accepted when generated. Methods to do this include the addition of parallel resources, data-staging techniques, and techniques that permit out-of-order processing and otherwise dynamically reschedule platform tasks.

Split transactions

*Split transactions* permit command and operation controllers to dynamically break operations into separate start, operation, and termina-

tion phases to increase overall system concurrency. Variations of this technique have been applied to instruction, memory, and bus controllers where multiple independent slave function units are capable of overlapped operation processing. In each of these situations, a common controller may dispatch queued nonconflicting split-transaction operations to multiple independent function-units. The controller need not wait for an older operation to finish before dispatching a younger operation to a free and available function-unit. Provided there are enough available function units, the split-transaction technique permits long duration operations to execute without stalling subsequent short duration operations.

### *Specific Performance Optimizations Made at the Component Level*

#### Data Storage Optimization Basics

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>e</sup>
<p style="text-align: center;">Data Storage</p> <ol style="list-style-type: none"> <li>1. Seek to increase the operating rate.</li> <li>2. Seek to increase the hit rate.</li> <li>3. Seek to improve low-level organization and management.</li> <li>4. Seek to reduce the retrieval latency.</li> </ol>

<sup>e</sup>. See Stipulations and Caveats on page 467.

The above set of basics is directed at optimizations at the component level to data storage devices intended for data-staging applications. There are a number of storage device types that correspond to standard organization and technology forms. These standard storage device types, in increasing order of performance from a system perspective, include off-chip DRAM, multiple types of specialty off-chip DRAM, off-chip SRAM, on-chip DRAM, on-chip SRAM, and on-chip registers. Using better circuit designs and semiconductor processing technology within a storage device type can increase operating rates. When such measures do not offer sufficient improvement, then the platform architect should consider an intrinsically faster device type.

Increases in buffer hit rate are generally accomplished through larger buffer sizes, and forms of increased associativity. Better organization and management can improve one or both of operating rate and hit rate. Techniques include using a better replacement algorithm, increasing the line width, the use of sectorized lines, and improving the fill method. Latencies are generally reduced at the device level by incorporating faster buffer

*Chapter 5 discusses, in more detail, platform memory controller operation, including low-level memory organization and management techniques for optimized performance.*

technology, and at the platform level by making sure that the buffer is managed in a way that it has the desired data when needed.

Bus Optimization Basics

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>f</sup>
<p>Bus Optimization at the Component Level</p> <ol style="list-style-type: none"><li>1. Seek to use a bus protocol matched to the data transfer requirements.</li><li>2. Seek to increase the speed of operation.</li><li>3. Seek to increase the effective bandwidth.</li><li>4. Seek to pipeline transfers.</li><li>5. Seek to use split-transaction transfers.</li><li>6. Seek to improve transfer efficiency.</li></ol>

<sup>f</sup> See Stipulations and Caveats on page 467.

The above set of basics is directed at optimizations at the component level to buses, ports, and other interconnects. There are a number of bus standards and types that correspond to established organization and technology forms. Bus optimization begins with selecting a bus protocol matched to the required data transfer characteristics. Bus operating rates can be increased by increasing the bus clock to the technology limits of a given bus standard or type. This includes improvements in bus I/O driver circuit design and processing technology. When such measures do not offer sufficient improvement, then the platform architect should consider an intrinsically faster bus type.

More generally, bus optimizations also include increases in the bus width, changes to the electrical signaling protocol beyond mere circuit and process enhancements, and increasing bus concurrency by pipeline and split-transaction methods. Bus transfer efficiencies are improved by reducing the non-data overhead associated with each bus transfer. Changes are focused on getting more data items per transfer by reducing arbitration, address, and handshake cycles relative to data cycles. A key method of accomplishing better transfer efficiency is the use of burst or block transfers.

*The Chapter 5 section on Rambus technology discusses many interconnect optimization techniques in more detail.*



Bridge/Controller Optimization Basics

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>9</sup>
<p>Bridge/Controller</p> <ol style="list-style-type: none"><li>1. Seek to isolate the various buses and ports whenever possible.</li><li>2. Seek to couple two subsystems without delay when required.</li><li>3. Seek to smoothly couple subsystems operating at different rates.</li><li>4. Seek to reduce stalls in coupled subsystems.</li><li>5. Seek to virtualize a multi-ported main memory.</li></ol>

<sup>9</sup> See Stipulations and Caveats on page 467.

The North-Bridge has a number of system optimizations that are inherent in the controller functions it performs. For example, it normally isolates the different buses and the main memory connected to it. This maximizes the opportunity for concurrent system operations to occur. As required, it performs *crossing transfers* by selectively coupling any two of the buses connected to it, or couples one of the buses with the main memory. During these coupling operations the North-Bridge acts as a bus controller, implementing the protocols of the various buses it couples. Thus, the bus transaction is translated from the protocol of the master's bus to that of the slave.

*crossing transfers*

Bridges may selectively couple buses for crossing transfers based on conditions established via programmable configuration. Generally a variety of different configurations can be set up for different buses, crossing directions and address regions. Some of the more common techniques for enabling crossing transfers include positive, negative, and subtractive decode.


DEFINITION
<p>Bridge/Controller Crossing Transfer Decode Schemes</p> <p><i>Positive Decode</i> immediately enables a crossing transfer when the address is inside one or more predefined positive-decode address regions.</p> <p><i>Negative Decode</i> immediately enables crossing transfers except when the address is inside one or more predefined negative-decode address regions.</p> <p><i>Subtractive Decode</i> enables a crossing transfer after a predefined latency, if no slave has responded to the address.</p>

*Selective crossing transfers are discussed in more detail in U.S. Patent 5,627,976, entitled “Crossing Transfers for Maximizing the Effective Bandwidth in a Dual-bus Architecture.”*

These techniques were particularly useful in South-Bridges during the early ascension of the PCI Bus. Programmable configuration of crossing transfer conditions gave transparent flexibility in choice of bus type for peripheral cards. For example, during the configuration process, the bridge was programmed based on whether the graphics adapter was an ISA-Bus-based device, or whether it resided on the PCI Bus. If a program wrote to reserved video memory locations implemented on the adapter, the bridge could correctly and immediately couple or isolate the PCI and ISA Buses, entirely based on the programmed configuration.

The North-Bridge also functions as a memory controller for the main memory, implementing memory access protocols and refresh as required for the installed memory technology. In conjunction with both its memory control and bus coupling functions, the North-Bridge possesses a number of command and data buffers. These buffers are used for various data-staging techniques including posted writes, write combining, caching, and prefetching. These techniques reduce bus traffic, reduce stalls and latencies, enable different buses to operate at different rates, and support the illusion that, for much of the time, the main memory has dedicated ports for each bus.

REPORTS ON CD-ROM



To learn more about bridge operation see the following data sheets on the companion CD-ROM:

- *AMD-640 System Controller Data Sheet, 21090C/0.*
- *AMD-645 Peripheral Bus Controller, 21095B/0.*

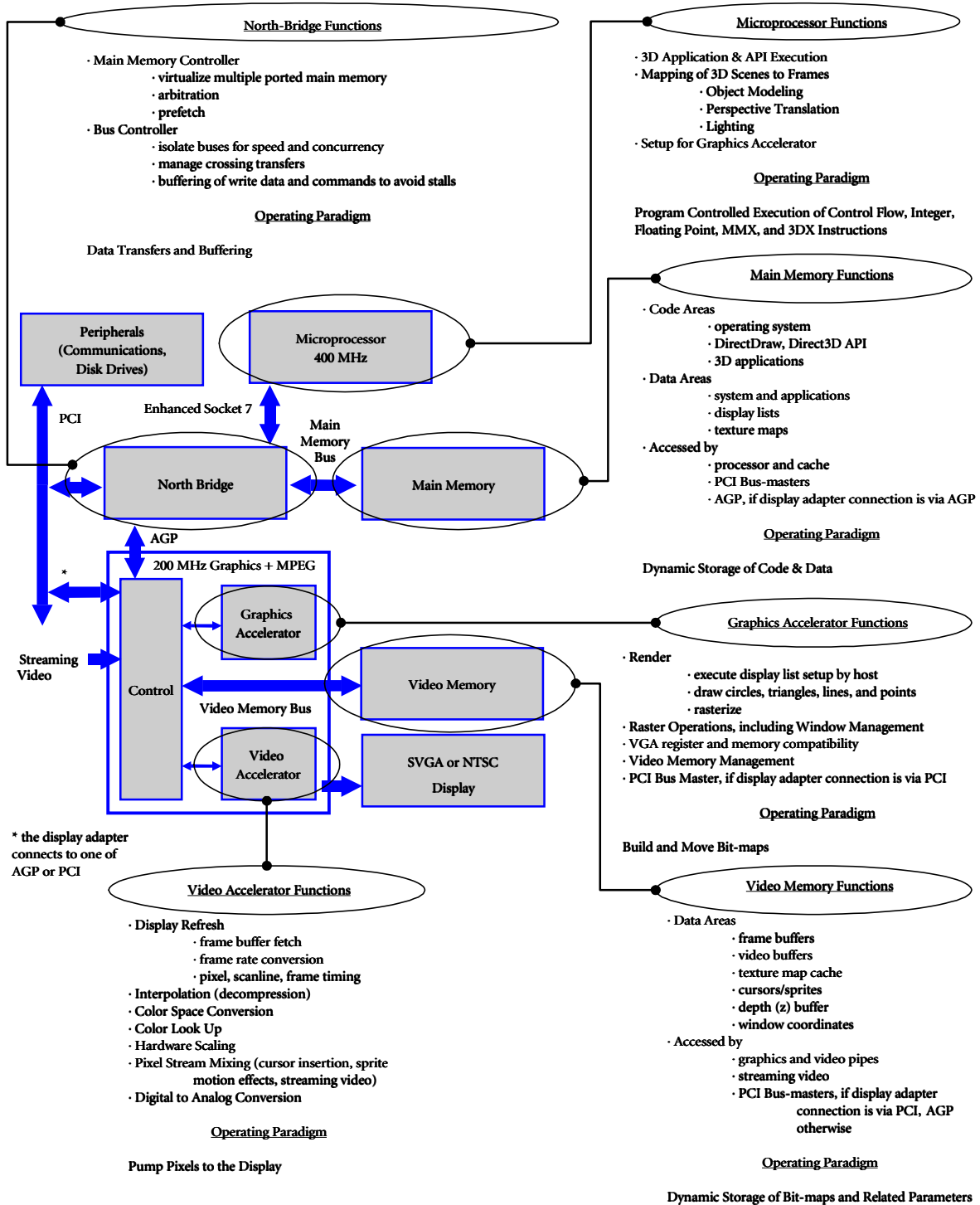
PERFORMANCE OPTIMIZATION SPECIFICS IN A CONTEMPORARY 3D GRAPHICS PLATFORM

We will next examine more closely the operation of 3D graphics in contemporary platforms. This will give the reader better insight into overall platform operation and how the foregoing optimization rules are applied from both low-level and high-level perspectives. To do this, we start by examining Figure 6.2. This is a rearranged and abstracted representation of the “3D Graphics PC Platform” previously presented in Chapter 1. Figure 6.2 has also been annotated with each component’s principal operating paradigm and functions. These characterize the role of each component from a component-localized and hardware-centric perspective. Understanding Figure 6.2 is a prerequisite to our later study of the 3D graphics pipeline from an application perspective.

## *Component Operating Paradigms, Functions for 3D Graphics, and Optimization Features*

### Microprocessor

The microprocessor's operating paradigm is that of program-controlled instruction execution of control, integer, floating-point, and other specialized instructions. For 3D graphics, the microprocessor acts to execute 3D applications and at least one Application Programming Interface (API). This execution includes geometry calculations and setup for the graphics accelerator. The geometry and setup calculations compose 3D scenes for each frame to be rendered and reduce the scenes to command and parameter data for the graphics accelerator. The composition involves floating-point calculations for object modeling, perspective translation, and lighting. High-level microprocessor-specific optimization features include a high-speed execution unit core having MMX and 3D instructions for accelerating geometry calculations, data staging between the core and the external processor bus using on-chip first- and possibly second-level caches, and an enhanced external processor bus interface running at 100 MHz.



**Figure 6.2** ROLES OF PLATFORM COMPONENTS IN 3D GRAPHICS

### Main Memory

The main memory's operating paradigm is that of dynamic storage of code and data. For 3D graphics, main memory provides access to code and data areas by the processor and cache subsystem, PCI Bus-master peripherals, and the Advanced Graphics Port (AGP). The code areas contain the operating system, DirectDraw and Direct3D API, and 3D applications. The data areas include system and applications data, texture maps, and display lists. Optimization features for memories include larger total sizes, high-speed interfaces, high-speed core memory arrays, efficient control protocols, and multiple banks capable of concurrent operation.

*Memory performance issues are discussed in more detail in Chapter 5.*


### North-Bridge

The North-Bridge's operating paradigm is that of selective data transfer and buffering. The North-Bridge includes bus and memory control functions. As a bus controller it manages crossing transfers between diverse bus protocols. As a memory controller it manages the DRAM main memory. Optimization features include isolating buses for speed and concurrency when not bus-crossing, buffering bus-crossing data transfers to avoid stalls, creating the illusion that the main memory is multi-ported, and implementing a dedicated port for transfers between main memory and the graphics accelerator and video memory.

### Graphics Accelerator

The graphic accelerator's operating paradigm is to build and move bit maps. Its functions include managing video memory accesses by multiple requestors, rendering images into the video memory, and providing VGA compatibility. The very existence of the graphics accelerator is a platform-level optimization in that the accelerator performs rendering operations that otherwise would be executed on the processor. Other optimization features include providing acceleration specifically for 3D rendering and raster operations, performing window management, acting as a PCI Bus-master, executing display lists in memory setup by the host, and implementing AGP for transfers between main memory and the graphics accelerator and video memory.

ARTICLE ON CD-ROM

	<p>To learn more about Graphics Accelerator, see the article “Competition Heats Up in 3D Accelerators,” by Yong Yao, <i>Microprocessor Report</i>, Vol. 10, No. 3, March 5, 1996, on the companion CD-ROM.</p>
---	--

[Video Accelerator](#)

The video accelerator’s operating paradigm is to pump pixels to the display. Its functions include frame buffer fetch; establishing pixel, scanline, and frame timing; digital to analog conversion; and display refresh. Optimization features include pixel stream mixing for cursor insertion, sprite motion effects, and windowing of streaming video; interpolation; color space conversion; color lookup; and hardware scaling.

[Video Memory](#)

Front and back (dual) frame buffers effectively increase bandwidth by removing previous strict limitations on frame buffer writes.

Dual-ported video DRAM permit random access for graphics accelerator access while permitting simultaneous sequential access for the video accelerator, effectively increasing the bandwidth available to the random-access port.

The video memory’s operating paradigm is the dynamic storage of bitmaps and related parameters. For 3D graphics, video memory provides access to data areas by the graphics and video accelerators, streaming video (or video capture) port, PCI Bus-master peripherals, and AGP. The data areas include front and back buffers, streaming video buffers, texture map caches, cursor/sprite definitions, depth (z) buffer, and window coordinates. General optimization features for video memories are the same as for main memories above. Specialty DRAMs are more frequently used in video memories than in main memories.

## DEFINITION

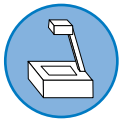
## Processes Accessing Video Memory

Our discussion of video memory access has been greatly simplified. The following is a list, in order of decreasing priority, of processes accessing video memory for a contemporary 3D graphics accelerator.<sup>a</sup>

- Primary stream fetch
- RAM refresh
- RAMDAC read/write
- Secondary stream fetch
- Hardware cursor fetch
- Local peripheral bus (used for streaming video or video capture writes)
- Read DMA
- CPU accesses
- Graphics accelerator accesses

<sup>a</sup> *ViRGE/VX Integrated 3D Accelerator*, S3 Incorporated, June 1996, pp. 7-16.

## TECHNICAL PRESENTATION ON CD-ROM



“Platform Components for 3D Graphics” is a Technical Presentation included on the CD-ROM that is a companion to the foregoing discussion of Figure 6.2 on page 482.

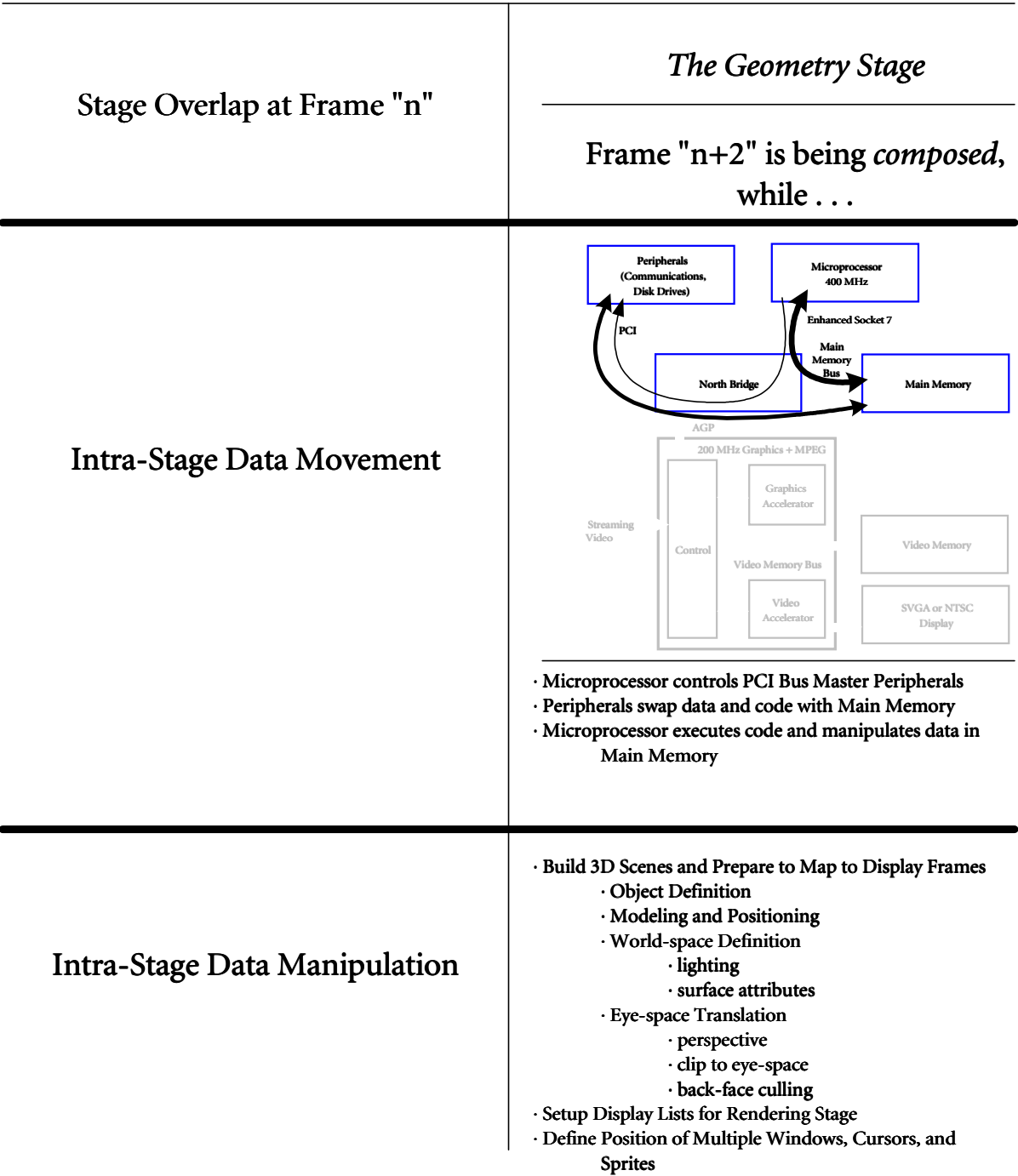
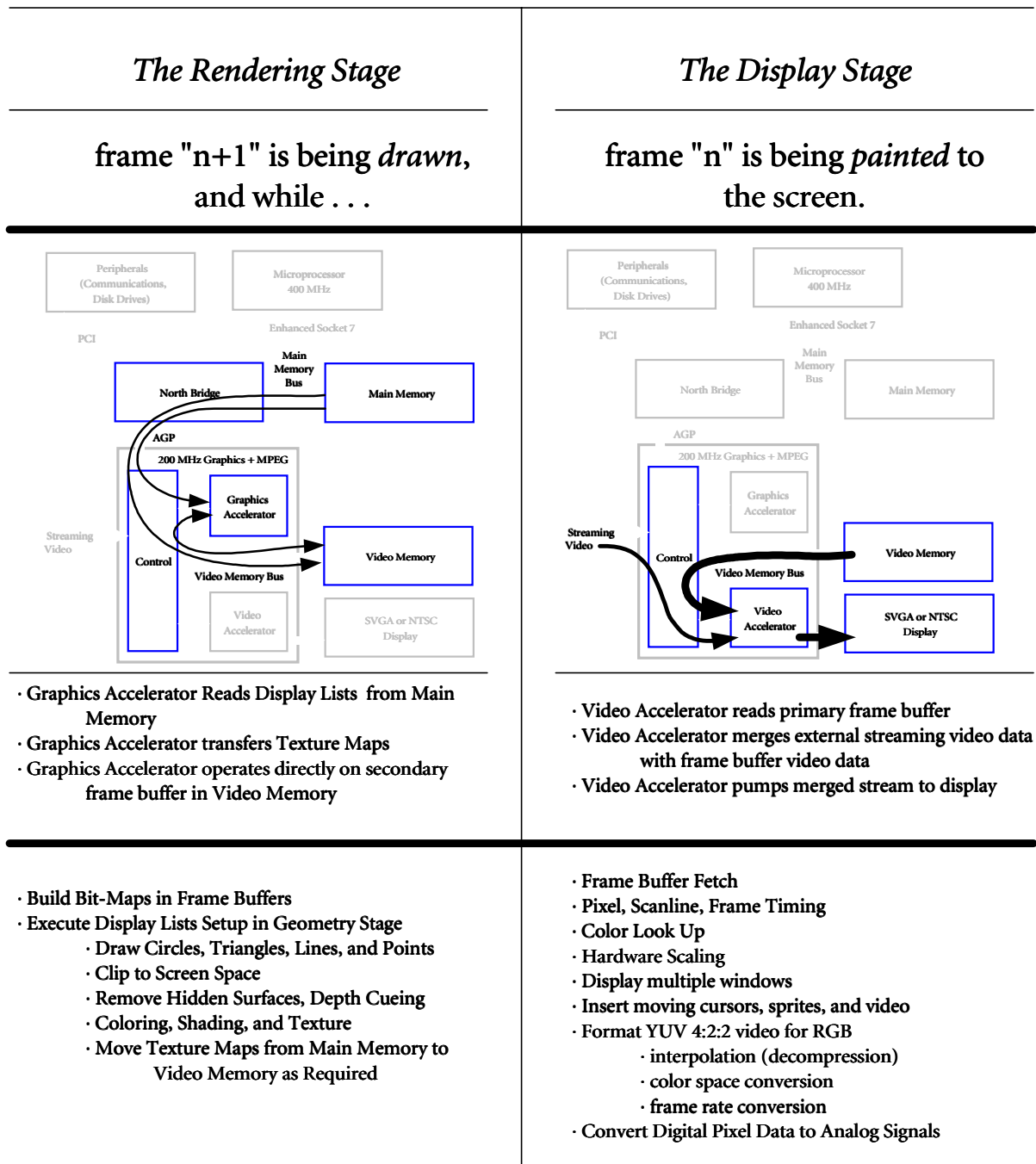


Figure 6.3 3D GRAPHICS PIPELINE STAGES (THE GEOMETRY STAGE)





**Figure 6.4** 3D GRAPHICS PIPELINE STAGE (THE RENDERING STAGE AND THE DISPLAY STAGE)

Elsewhere, when more abstract discussions focus purely on 3D geometry and rendering, the Display Stage may be implicitly included in the Rendering Stage. Here, our focus is on the relatively lower-level platform architecture and hardware behavior. In this context the Display Stage deserves to be treated separately.

setup

The Geometry Stage may also compose descriptions for display control features (such as Window positions and scaling, cursors, and sprites) that are processed only by the Display Stage.

Geometry, Rendering, and Display—The 3D Graphics Pipeline

Figure 6.3 and Figure 6.4 offer an application perspective of the 3D graphics pipeline. They apply the platform organization used in Figure 6.2 on page 482 as a template to examine the dynamic behavior of data movements and manipulations for each stage of the 3D graphics pipeline. Figure 6.3 and Figure 6.4 are intended to be viewed collectively as one large graphic that simultaneously shows the activities of three stages that make up the 3D graphics pipeline. These stages are, from left to right and in order of processing, the Geometry, Rendering, and Display Stages. We will next overview the particular data manipulations and movements that characterize each stage.

The Geometry (Modeling) Stage

DEFINITION
<p>Geometry Stage</p> <p>The Geometry Stage first models, then composes descriptions, representative of a real or virtual world scene for each frame to be displayed. The descriptions take the form of sequences of commands, parameters, and other data, that can be readily processed by the Rendering Stage. The preparation of these descriptions is called <i>setup</i>.</p>

The data movements of the Geometry Stage include the execution of programs by the microprocessor and its associated manipulation of data in main memory. Peripherals swap data and code with main memory as required by the executing program. These swaps are managed by system software that also executes on the microprocessor. Ideally, the microprocessor programs the PCI Bus-master peripherals that later carry out in isolation all necessary swaps at the behest of the processor.

The Geometry Stage data manipulations build 3D scenes in preparation for mapping these scenes to display frames. This involves object definition, including modeling and positioning; definition of the world-space (the interrelationship of objects to the unseen surrounding environment) through lighting and surface attributes, and eye-space translation, including application of perspective, clipping to eye-space, and back-face culling to simplify later processing by eliminating information obviously unnecessary for rendering. Subsequent to the foregoing geometry calculations, displays lists are set up for the Rendering Stage. In the 3D architecture represented, the microprocessor builds the display lists, in memory for subse-

quent execution by a bus-mastering graphics accelerator that fetches and executes the stored display list. In the 3D architecture represented, the microprocessor builds the display lists in memory for subsequent execution by a bus-mastering graphics accelerator that fetches and executes the stored display list.

### The Rendering Stage

DEFINITION
<p>Rendering Stage</p> <p>The Rendering Stage <i>draws</i> bit maps in the frame buffer based on the per frame descriptions set up by the Geometry Stage. The bit maps take the form of pixel data that collectively represent an image to be displayed for the current frame.</p>

The bus-mastering graphics accelerator predominantly controls the primary data movements of the Rendering Stage. These movements include the fetching of display lists from main memory, data manipulations on the video memory as dictated by the display lists, and the transfer of any required texture maps between main memory and video memory via the AGP.


Rendering Stage data manipulations, called for by the display lists previously set up in the Geometry Stage, are executed by the graphics accelerator. These data manipulations build bit maps in the frame buffer by drawing circles, triangles, lines, and points; clipping to the screen space; using depth cueing and otherwise removing hidden surfaces; incorporating coloring, shading, and texture; and moving, manipulating, and applying texture maps to define surface features.

*Rendering may be more directly controlled by the microprocessor. A more classic alternative is for the microprocessor to use PIO in the Rendering Stage to send the commands and data (functionally equivalent to the display list) directly to the graphics accelerator, which functions as a slave controller to the microprocessor.*

DEFINITION
<p>3D Rendering Models</p> <p><i>Polygonal Rendering</i> is the 3D rendering model underlying all the examples in this book. That is, the real world scenes are decomposed into a representation consisting of polygons (or simply triangles) and individual pixels. The polygonal model is popular in PC platforms because it offers rendering speeds supporting good user interactivity, at quality levels approaching arcade games, at relatively low cost. Other rendering techniques and algorithms exist.</p> <p><i>Talisman</i> is a Microsoft rendering initiative that is a “region-based” variation on the polygonal model. It attempts to boost pixel-fill rates and reduce processing demands by limiting rendering to only those scene areas that change from frame to frame. It also attempts to reduce bus bandwidth demands by transferring compressed texture maps.</p> <p><i>Ray tracing</i> is a rendering technique that offers superior quality, but has much greater computational requirements that do not support interactive use. Ray tracing techniques are usually used to render the Photorealistic images required by the motion picture industry.<sup>a</sup></p>

<sup>a</sup> “3D Vendors Prepare for Rough Seas in ‘98,” *Microprocessor Report*, MicroDesign Resources, Dec. 29, 1997.

REPORT ON CD-ROM

	To learn more about The Rendering Stage, see the article “Talisman Redefines 3D Rendering,” by Peter N. Glaskowsky, <i>Microprocessor Report</i> , Vol. 10, No. 11, August 6, 1996, on the companion CD-ROM.
---	--

The Display Stage

DEFINITION
<p>Display Stage</p> <p>The Display Stage <i>paints</i> pixels to the display generated as a function of the bit maps drawn in the frame buffer by the Rendering Stage.</p>

The video accelerator acts as a data pump, or more specifically a pixel pump, creating and controlling pixel data streams for the continual refresh

of the display. These pixel data streams define the primary data movements of the Display Stage. These movements include fetches of pixel data from selected frame buffer locations, pumping pixel data through a pixel processing pipeline (described below), merging pixel data streams from various sources, and pumping the merged pixel stream to appropriate interface circuitry for either a CRT, LCD, or other display technology.

The video accelerator manipulates the pixel data in a pixel-processing pipeline. The pipeline is programmatically configured by display control commands sent by the microprocessor. Many such commands are performed during the initialization of the display subsystem. However, others may be dynamically performed in conjunction with the rendering setup of the Geometry Stage. Common commands establish the pixel, scanline, and frame timing; perform any required frame rate conversions; and establish window boundaries and the location of cursors and sprites. Subsequent to the foregoing configuration, the video accelerator's pixel-processing pipeline performs any required interpolation, color space conversion, scaling, and color lookup; insertion of moving cursors, sprites, and streaming video; and conversion of pixel data to analog signals.

*The pixels painted by the Display Stage are generally a function of display control features (discussed above) set up by the Geometry Stage.*

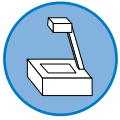
### Stage Overlap in the 3D Graphics Pipeline

The three stages reviewed above are in execution simultaneously. As annotated in Figure 6.3 on page 486 and Figure 6.4 on page 487, consider the stage processing overlap at a time when the user perceives a frame “n” being displayed. This happens when frame “n” is being read from the front frame buffer, manipulated by the pixel-processing pipeline, and painted to the screen, all in the Display Stage. Simultaneous with the painting of frame “n,” the Rendering Stage is executing display lists from main memory and drawing frame “n+1” to the back frame buffer. Also simultaneous with the painting of frame “n,” the Geometry Stage is composing and performing rendering setup for frame “n+2.”

*For the moment in time represented by Figure 6.3 on page 486 and Figure 6.4 on page 487, frame “n” previously passed through the Geometry and Rendering Stages and frame “n+1” previously passed through the Geometry Stage.*

The overlap can also be visualized as follows. Imagine that each of the three data movement drawings (of the collective illustration formed by Figure 6.3 on page 486 and Figure 6.4 on page 487) is a separate transparency. By overlaying and aligning the transparencies of the three stages, the composite view gives the collective data movements on going simultaneously during the display of frame “n.”


TECHNICAL PRESENTATION ON CD-ROM

	<p>“3D Pipeline” is a Technical Presentation included on the CD-ROM that illustrates that all three stages are overlapped in each frame.</p>
---	--

*If the microprocessor is used to send rendering commands and data directly to the graphics accelerator, increased competition for the microprocessor and PCI Bus will occur between the Geometry and Rendering Stages.*

Note that the main memory and North-Bridge are used in both the Geometry and Rendering Stages. Also note that the video memory is used in both the Rendering and Display Stages. Thus, though our abstract representation has ignored the possibility of the platform executing other tasks, there will be competition for these resources even between stages of the 3D graphics pipeline. Use of well-planned arbitration protocols and data staging is especially important in order to effectively manage stalls and reduce latencies that occur due to such resource competition.

ARTICLE ON CD-ROM

	<p>To learn more about Stage Overlap in the 3D Graphics Pipeline, see the article “PC Graphics Reach New Level: 3D,” by Yong Yao, <i>Microprocessor Report</i>, Vol. 10, No. 1, January 22, 1996, on the companion CD-ROM.</p>
--	--

SUMMARY OF CONTEMPORARY PLATFORM OPTIMIZATIONS

Included below is the combined list of the optimizations discussed in this major section. These basic principles can be seen in contemporary platform designs, which have gone beyond merely increasing operating rates and shifting to faster technology.

**PLATFORM PERFORMANCE OPTIMIZATION BASICS<sup>h</sup>**

## Compilation of all Optimization Basics

1. Seek to increase operating rates.
2. Seek to increase resource operating-widths.
3. Seek to increase the efficient use of high-demand resources.
4. Seek to reduce the number of subsystems that process each task.
5. Seek to keep expensive resources continuously productive.
6. Seek to reduce unnecessary bus transactions and transfers.
7. Seek to normally isolate concurrent operation buses capable of.
8. Seek to increase the number of continuously productive buses.
9. Seek to reduce simultaneous competition for each bus.
10. Seek to minimize the use of Programmed I/O (PIO).
11. Seek to command normally isolated bus-mastering accelerators.
12. Seek to transfer compressed data.
13. Seek to keep streaming data out of general-purpose subsystems.
14. Seek to maximize block sizes and minimize per-block latencies.
15. Seek to exploit spatial and temporal locality of reference.
16. Seek to avoid stalls.
17. Seek to reduce latencies.
18. Seek to reduce data traffic between subsystems.
19. Seek to smoothly couple subsystems operating at different rates.
20. Seek to identify the optimal block size.
21. Seek to overlap the processing of multiple tasks.
22. Seek to explicitly pipeline task processing.
23. Seek to speculatively prefetch sequential data streams.
24. Seek to decompose operations into split transactions and enable increased pipelining or out-of-order processing.
25. Seek to add independent concurrently operating resources.
26. Seek to optimize data storage devices
  - a. Seek to increase the operating rate.
  - b. Seek to increase the hit rate.
  - c. Seek to improve low-level organization and management.
  - d. Seek to reduce the retrieval latency.

<sup>h</sup>. See Stipulations and Caveats on page 467.

PLATFORM PERFORMANCE OPTIMIZATION BASICS <sup>i</sup> (CONT.)
Compilation of all Optimization Basics
27. Seek to optimize buses. <ul style="list-style-type: none"><li>a. Seek to increase the speed of operation.</li><li>b. Seek to increase the effective bandwidth.</li><li>c. Seek to pipeline transfers.</li><li>d. Seek to use split-transaction transfers.</li><li>e. Seek to improve transfer efficiency.</li></ul>
28. Seek to use a bus protocol matched to the data transfer requirements.
29. Seek to optimize Bus-bridges and Controllers <ul style="list-style-type: none"><li>a. seek to isolate the various buses and ports whenever possible.</li><li>b. Seek to couple two subsystems without delay when required.</li><li>c. Seek to smoothly couple subsystems operating at different rates.</li><li>d. Seek to reduce stalls in coupled subsystems.</li><li>e. Seek to virtualize a multi-ported main memory.</li></ul>

<sup>i</sup> See Stipulations and Caveats on page 467.

Contemporary platforms use sophisticated organization and management of multiple autonomous subsystems that operate concurrently, but at independent rates, coupled loosely via bridges that perform data staging to avoid stalls, reduce latencies, and virtualize ports to high-demand resources. The autonomous subsystems make use of bus-mastering data pumps, dedicated accelerators, and peripherals; data staging; efficient protocols; and explicit pipelining. Performance has been increased and costs reduced by keeping platform resources continuously and efficiently productive.

Platform optimization is a dynamic problem. Platform architecture is a moving target. Designs must be continually adapted to changes in application requirements and new uses of the platform. The next section examines the directions the PC platform is taking to track these changes.



## ARTICLES ON CD-ROM



To learn more about Platform Optimizations, see the following articles on the companion CD-ROM:

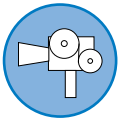
- “A Performance Analysis of Pentium Processor Systems,” by Michael Bekerman and Avi Mendelson, *IEEE Micro*, Vol. 15, No. 5, October 1995.
- “Limited Bandwidth to Affect Processor Design,” by Doug Burger, James R. Goodman and Alain Kägi, *IEEE Micro*, November/December 1997.

## OVERVIEW OF CONTEMPORARY ISSUES IMPACTING PC PLATFORMS

## PLATFORM DIRECTIONS

The most important aspects of contemporary platform development pertains to *connectivity* and *content*. These are associated with new uses of the PC, changes in the base of PC users, and new views of PC platforms.

## VIDEO ON CD-ROM



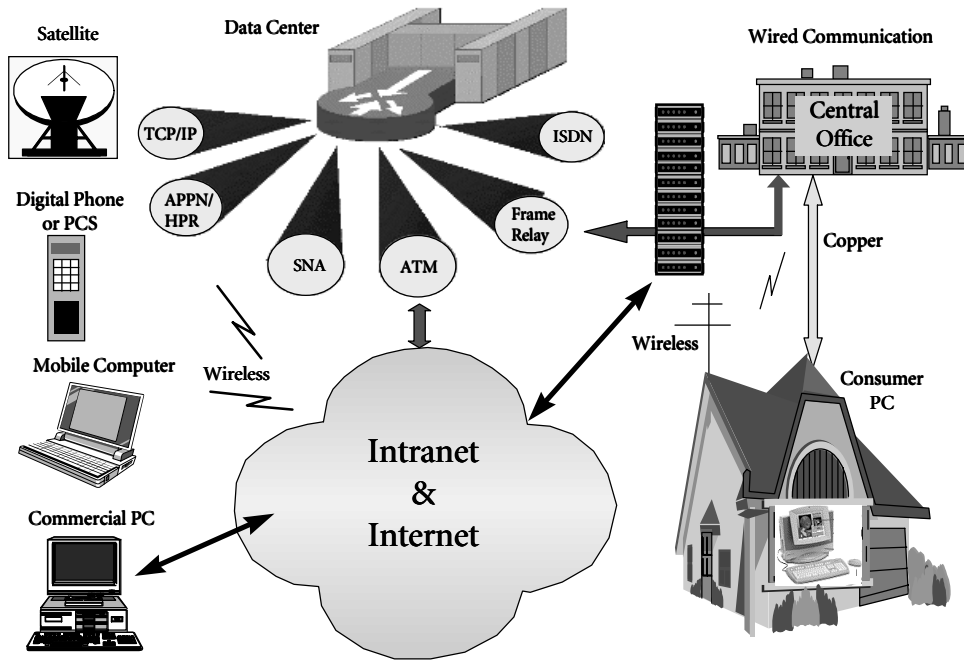
Atiq Raza, AMD CTO and Executive VP, replies to the question: “How are the changes in connectivity and content impacting PC systems?”

## DEFINITION

## Connectivity

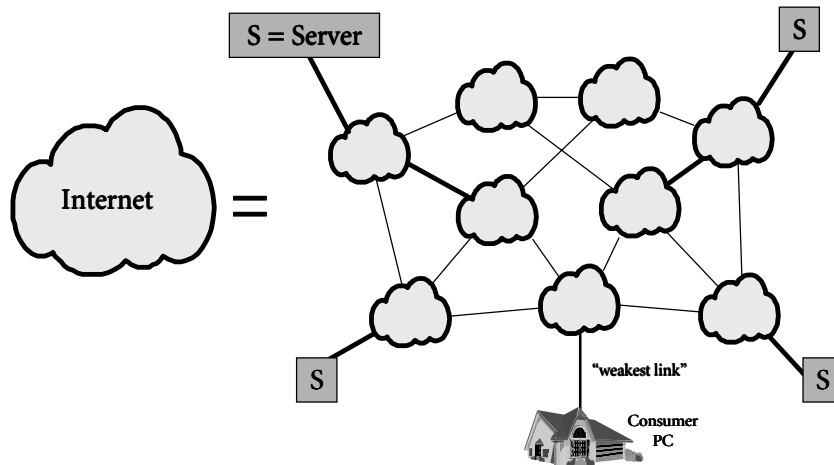
*Connectivity* the connection of a PC platform to computing, storage, or I/O resources. The resources may be local or remote. The resources may be accessed via connection with workgroup servers, the corporate data center, a personal satellite link, the TelCo (Telephone Company) central office, the Internet or an intranet. The connection may be via modem or network and may be wired or wireless.

*Connectivity* in new platform installations is changing rapidly, becoming more commonplace and more diverse. Connectivity is predominantly via network for businesses of all sizes and via modem for consumers and small businesses. Connections are predominantly wired, with emerging use of wireless. Figure 6.5 illustrates some of the many methods of connectivity available today.



**Figure 6.5** METHODS OF CONNECTIVITY

Adapted with permission of Advanced Micro Devices Inc., from *Competing with Intel*, Copyright 1997.



**Figure 6.6** VARIATIONS IN INTERNET BANDWIDTH

Adapted with permission of Advanced Micro Devices Inc., from *Corporate Strategy*, Copyright 1997.

**DEFINITION****Data Concentration**

Data Concentration is the implementation of multi-user access to (hard disk) storage servers via workgroup, intranet, and Internet networking technologies, with an emphasis on reducing client storage. Data concentration can reduce overall system storage requirements by reducing unnecessary multiple copies, facilitates on-demand access of critical information to anyone with the appropriate access permissions, and enables project data to be backed up more easily by data-center personnel.

Ideally, data concentration could manage hard disk storage on all machines coupled to the network from an overall system perspective. Such an ideal is similar to recent proposals for *thin clients*, *Network Computers*, and *NetPCs*, all of which shift storage requirements, in various degrees, to networked storage servers. In conventional practice, data is kept on servers rather than on users' machines whenever applicable and practical, by management initiative and on an ad hoc basis decided by content creators and users, under permissions decided by server administrators and management.

Commercial data concentration is the most significant factor driving connectivity growth. Large entities are rapidly building and expanding private intranets to provide better organization, control, and increased utilization of geographically distributed computing resources. In conjunction with this activity, commercial bandwidth is growing an order of magnitude at a time and faster than PC processing growth. This increase in connectivity, and the disparity in growth between bandwidth and processing power, are evolving the PC platform and are redefining PC platform market segments.

The Internet also provides data concentration, but in the public infrastructure. It does so, in that publicly accessible databases and archives reduce the need for users to acquire and maintain individual copies of popular information. This public infrastructure is built upon the storage servers of Internet Service Providers (ISPs) providing subscribers with Web site hosting services, Usenet and other news services, and storage and forwarding of e-mail.

However, unlike the order-of-magnitude growth in the commercial sector, the growth in Internet access bandwidth for consumers is slower. This growth is being paced by the connection between the Internet Service Provider and the end user, which typically is the slowest part of the link. High-speed links to the Internet are presently beyond the reach of most

households’ budget. Figure 6.6 on page 496 graphically depicts the variations in bandwidth that exist between the nodes that comprise the Internet.

DEFINITION
<p>Content</p> <p>In the same way that the program <i>content</i> on a television channel describes the nature of the program being broadcast, in the PC world <i>content</i> connotes the type of subject matter that the user chooses to access or manipulate on the platform. In the broadest sense, content is whatever subject matter the user chooses to have delivered, or to interact with, using the PC platform. More narrowly, the term is most often applied to professional-quality entertainment, news, education, games, or an information/database retrieval facility or service.</p>

PC platforms for users are coming to be thought of, not as computing devices, but as user-operated tools (or information appliances) for *content access*. For example, many consumers view PCs as Internet-access devices. In this emerging paradigm, enabled by evolving sophistication that better hides underlying complexity, applications are now often described in terms of the *content* they provide, generate, or manipulate for the user.

PC content is continuing to change rapidly. New multimedia-based content forms are relying on intensive use of 2D and 3D graphics, 3D-quality audio, quality video, and voice recognition and voice synthesis. The source of multimedia data today is now expanded beyond the CD-ROM to include DVD-ROMs and the Internet. These multimedia technologies are being used commercially for the visualization and presentation of business data. In games they are providing consumers with realism for simulated worlds previously available only in arcades or in expensive commercial applications. The new multimedia-based content is providing significantly enhanced user interfaces and experiences for all applications compared to earlier PCs.

Internet and intranet content is characterized primarily by the Web-browser navigation paradigm. The use of the Internet for data centralization and its complement—information retrieval (including searching, browsing, and surfing)—is now well established among computer-literate consumers. In the commercial PC market Internet technology is also increasingly used for the dissemination of both internal and external information.

The emergence of these new content types is driving a relentless and rapid flow of enhancements at both the processor and the platform level. Rich graphical content is particularly demanding on all aspects of platform performance and is driving new standards and designs for all PC

platform components and interconnects. Microprocessor enhancements include higher frequencies, increased instruction-level parallelism, and superscalar units for emerging specific content. Platform enhancements include new means to access, process, and display content and infrastructure support for processor enhancements.

The rapid growth of Internet and intranet content and the requisite connectivity has led to fast growth in the storage server platform segment. While the bulk of servers have historically come from non-PC platforms, Internet Service Providers and data centers are viewing PC platforms from a new perspective as high-performance/low-cost Windows NT-based multi-processor PCs increasingly penetrate into the Internet and intranet server market.

Server growth is particularly strong in the commercial segment. In particular, intranet-connected servers are being used to accomplish data (storage) concentration. While growth in the Internet also clearly impacts server sales, the effect is somewhat muted by the tendency of ISPs to concentrate a greater number of user ports per server, increasing latencies and throughput, than would generally be acceptable in a commercial intranet environment. Internet growth is also occurring at a slower rate than intranet growth, being limited by the lack of cheap high-speed connectivity. The most likely candidates for providing Internet connectivity, the Telephone Companies, have been slow in adapting to the rapidly developing needs of Internet users. Inexpensive Internet connections are tediously slow. High-speed Internet connections have high setup costs and metered access with significant usage rates. In either case, the user's finite on line budget—of time or money—significantly limits their Internet activity.

*Recent changes in content and connectivity have led to fast growth in storage servers.*

## CONTEMPORARY PLATFORM STRATEGIC ISSUES

A number of key issues have a significant influence on PC platform design. Today, platform features and techniques are of necessity being enhanced in order to realize the performance potential of present and forthcoming high-performance microprocessors. Due to the rapid pace of increasing performance, these changes can no longer be done in isolation. As a result, co-design of processors and PC platforms has become standard. Internet, multimedia, networking, and communications technologies are continuing their ascension. The perception that PC users are rapidly adopting and demanding these technologies is further redefining the PC platform as features are added to track shifts in how PCs are being used. This section examines these issues and their impact on the PC platform.

*This section overviews the contemporary platform environment and the design and business issues that exist for processor and platform design. Future Platforms are discussed at the end of this chapter.*

### *Directions in Optimization of Contemporary Systems*

A number of features and techniques are being applied at the processor and platform level to optimize systems performance in the near term. These optimizations, driven primarily by the new connectivity and content, are rapidly continuing, at the microprocessor, chipset, and motherboard levels.

At the processor level, optimizations include faster clock rates, faster bus rates, improved architectural performance, and changes to the platform memory hierarchy. In addition to its traditional computation and control roles, the processor in contemporary platforms is called upon for signal processing for playback decoding of digital multimedia; local encoding for authoring of digital multimedia and imaging; and for geometry computations and rendering setup to support the 3D graphics display subsystem.

At the chipset and motherboard levels, there is a seemingly insatiable demand for bus and memory bandwidth. The memory and I/O subsystems must be optimized to sustain the higher processor performance and higher bandwidth data transfers required for access, processing, and display of rich 3D graphics. Support must also be included for emerging networking, communications, multimedia, graphics, and I/O standards and associated newer infrastructure chips. The platform must also reflect associated changes that are occurring in the interface between hardware and software. Platforms must support these new standards and interfaces in order for them to be successful.

General platform optimizations are being performed within the Socket 7 infrastructure. These changes will extend Socket 7's viability by providing increased performance and new standards and features. AMD refers to the enhanced Socket 7 as "Super 7." For uniprocessor systems, Super 7 promises to be highly competitive on a performance-to-cost basis with platforms based on the P6 processor bus.

Platforms based on the new Super 7 chipsets will extend frequency and bus bandwidths, permit larger main memory and caches, and allow for a backside cache. They also will offer I/O system improvements to display, disk, and communications subsystems. These platforms will support the most important new standards such as ACPI, AGP, 1394, USB, and DVD-ROM. Super 7 chipsets will offer improved PCI system bus efficiency by incorporating the latest interleaved system bus-mastering and faster arbitration protocols. Sophisticated internal bridge architectures, including such techniques as bus transaction buffering, posted write buffers, and bus prefetch, will allow for multiple internal concurrent subsystems.

3D content requires Socket 7 infrastructure enhancements in the buses coupling the processor, North-Bridge, the memory subsystem, and the graphics subsystem. The processor bus for Super 7 is 100 MHz, compared to 66 MHz for the earlier Socket 7 processor bus. This increase is planned in conjunction with increasing the main memory bus bandwidth to 100 MHz. These optimizations are needed to reduce the penalty for off-chip accesses, which occur on cache-line write-backs to main memory, cache-line fills from main memory, and all non-cacheable accesses to main memory and other addressable platform components. Slow off-chip accesses, while relatively infrequent, act to limit the benefit realizable from processor core-frequency speedups.

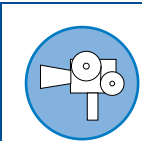
In addition to extending the main memory bus bandwidth to 100 MHz, the new Super 7 platforms will support larger and faster caches, ultimately including an on-chip backside cache. In each memory subsystem within the platform, a variety of competing memory technologies and organizations are being considered. These include EDO, SDRAM, Rambus, and SLDRAM. These memory technologies all differ in terms of their costs, ease of system design, latency and bandwidth of their interface, and the efficiency of their transfer protocols.

Super 7 platforms are also making specific optimizations in their graphic subsystems to support the display of contemporary 3D graphic applications. Primary among these optimizations is the inclusion of AGP. Motivated by the demands of 3D content, this new point-to-point bus directly couples the graphics accelerator to the North-Bridge. Contemporary Super 7 3D graphics accelerators will be AGP compatible, have texture mapping hardware assist, and also incorporate other hardware acceleration for 3D rendering.

*The increase in the processor bus speed happens to be closely inter-related to many other system issues, and requires consideration and management of a host of complex factors. See “Co-Design of Processors and Platforms” later in this chapter.*

*Advanced memory technologies and organizations are discussed in Chapter 5. The last section in Chapter 5 is devoted to a detailed discussion of Rambus.*

#### VIDEO ON CD-ROM



Atiq Raza, AMD CTO and Executive VP, replies to the question: “What are the difficulties in achieving optimal graphics performance?”

Advances in a PC platform’s processor performance, 3D graphics performance, and supporting architecture must be concurrent and well coordinated. Increased speed processors need platform architectures that can sustain the processor’s performance capabilities. The performance and capabilities of the entire system are readily discernible when using new content having rich 3D graphics. In the negative, rich 3D content will visually highlight the inadequacy of older platforms and slower processors. Frames may be noticeably dropped, audio synchronization with the display may be lost, and interactivity may be sporadic. In the positive, rich

3D content enables the user to visually appreciate the benefits of having a well-proportioned and-tuned advanced architecture platform with a fast processor. The presence of textures and other advanced 3D attributes, enabled by high-performance 3D graphics accelerators, are clearly visible and are impressive.

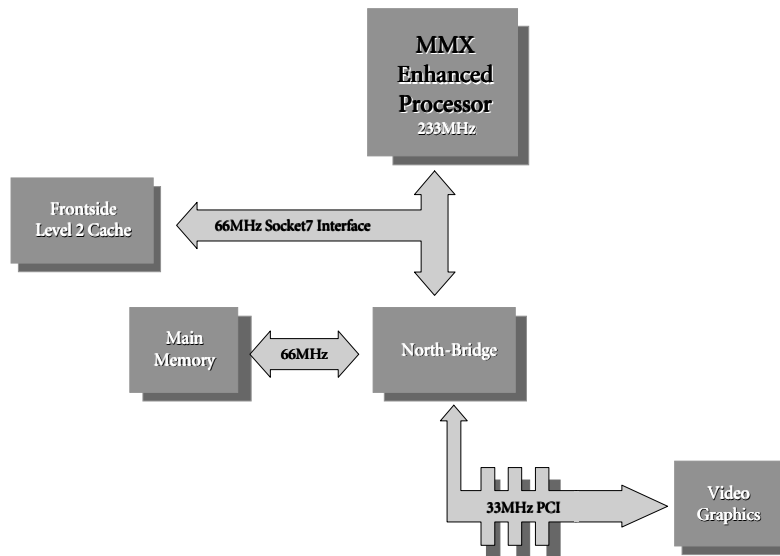
### Illustrative Progression of Platform Architecture

We next survey a succession of platform developments planned to occur over 1998. Compared to earlier systems, these contemporary systems offer improvements in microarchitectural performance and memory hierarchy efficiencies. Another important aspect, notable in each platform shown, is the constant upward march of core frequencies.

Figures 6.7-6.9 illustrate the rapid evolution that is occurring in platform architecture. Figure 6.7 shows a highly abstract view of a typical high-performance system in early '97. This system uses a frontside cache operating at the processor bus rate of 66 MHz. All bus traffic between the graphics accelerator and main memory makes use of the 33-MHz PCI system bus.

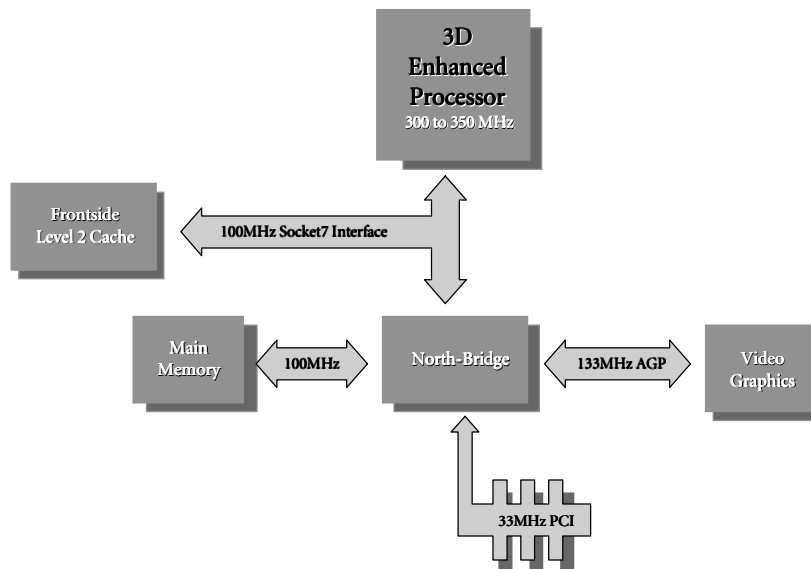
A Super 7 system planned for mid '98 is shown in Figure 6.8. This system also features 100-MHz processor and memory buses and the addition of instruction set extensions for 3D graphics. This system also augments the earlier Socket 7 system by the addition of a 133-MHz Advanced Graphics Port (AGP). The increased memory bus speed decreases latencies and improves throughputs for transfers via AGP to the graphics accelerator, or via the processor bus to the frontside secondary cache and to the processor core, including the on-chip primary cache. The increased processor bus reduces penalties for all off-chip accesses (cacheable and non-cacheable), reduces the latencies for primary and secondary cache-line fills, and increases the effective throughput of the entire memory system.





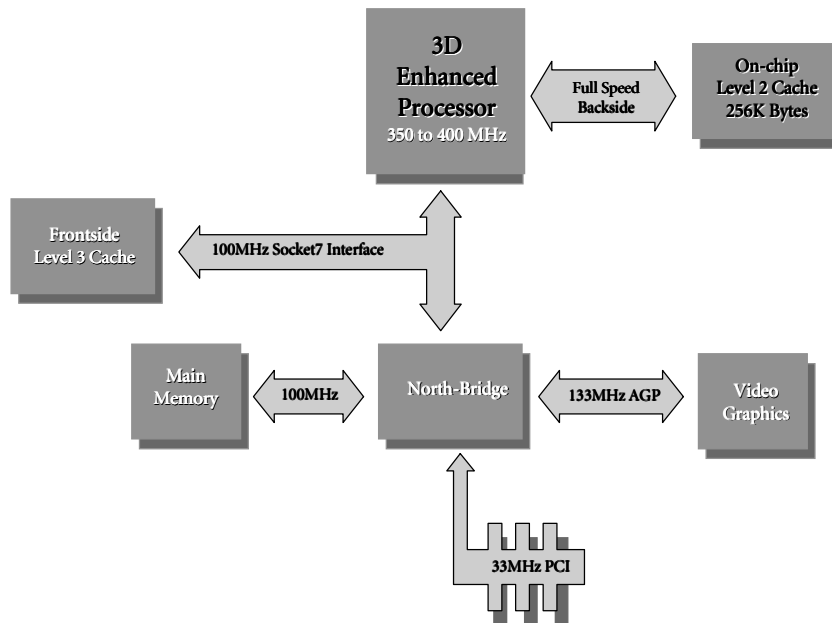
**Figure 6.7** 66MHz SOCKET 7 BUS PLATFORM

Adapted with permission of Advanced Micro Devices Inc., from *A New World Order—Alternative, Microsoft Windows Platforms*, Copyright 1997.



**Figure 6.8** A 100MHz SUPER 7 BUS PLATFORM

Adapted with permission of Advanced Micro Devices Inc., from *A New World Order—Alternative, Microsoft Windows Platforms*, Copyright 1997.



**Figure 6.9** A 100MHz SUPER 7 BUS PLATFORM WITH BACKSIDE CACHE

Adapted with permission of Advanced Micro Devices Inc., from *A New World Order—Alternative, Microsoft Windows Platforms*, Copyright 1997.

Figure 6.9 illustrates a Super 7 platform with backside cache, planned for the second half of '98. This platform augments the frontside cache with an on-chip 256KB backside cache that runs at the processor's core frequency. This significantly reduces latencies for primary cache-line fills and improves the apparent main memory performance perceived by the processor for cacheable accesses. The backside cache filters out many accesses that would otherwise result in traffic on the processor bus port to the North-Bridge, increasing the available bandwidth to main memory for other system components.

### *Co-Design of Processors and Platforms*

The relation between processor design and PC platform development is drawing closer. Platforms define the system context (or environment) in which the processor functions. Faster processors require greater care and sophistication in the design and implementation of the platform. This is requiring a closer coupling between processor and platform development in each succeeding generation of the PC platform.

Both processor and systems technology are currently undergoing rapid evolution. Each technology is being buffeted by fast ongoing changes in speed, density, performance, level of integration, new features, and new

industry standards. To avoid being caught with a new product design that is obsolete at introduction, product cycles are shortening in the face of this rapid technology march. Since higher profits are generally realized in the early stages of the product cycle than later, it is very important to orchestrate an early stage offering.

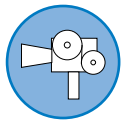
Early stage offerings are major feats to accomplish. In order to succeed, processor vendors must ensure that there will be an available supply of chipsets and motherboard reference designs with performance and features that enable and complement each new processor model. The design of the chipsets and motherboards must be carried out to ensure that the combination of processor and platform delivers a competitive offering that addresses platform vendors' wants and needs with respect to performance, features cost, constraints, volume, ease of design, and infrastructure support for new processor features.

Processor and platform co-design describes the close working relationship that has of necessity developed between processor design and the design of the surrounding parts of the system, especially the motherboard and its associated chipset. Co-design includes extensive technical and business exchanges between processor and platform designers.

Through co-design exchanges, complex hardware and software interactions between the processor and the surrounding system are cooperatively understood and mastered. As a result, the risk of latent compatibility problems is greatly reduced and better ease of system design generally results. Also, processor and platform designers have better knowledge of both parties' design constraints, capabilities, and volume expectations, and can act accordingly. Co-design also enables coordination of infrastructure support for new advanced technology features and standards, permitting their rapid introduction.

Both the processor and platform vendors benefit from co-design. They both share in the timely delivery of aggressively competitive systems with improved profitability. End users enjoy increased application performance and value, and thereby benefit from co-design as well.

#### VIDEO ON CD-ROM

	<p>Atiq Raza, AMD CTO and Executive VP, replies to the question: "Why is co-design so critical to PC systems development?"</p>
---	--

### *Continual Redefinition of the PC Platform*

There is a reinforcing circle of interaction that includes advances in processor and platform technology, development of new applications, and

The market segments are used as an abstraction of buyer needs and behavior. Illustrated by the evolution of the PC platform, the original PC platform was redefined over the course of a decade into the now familiar desktop, laptop, and workstation segments. In the last several years these segments have been further redefined by distinctions between enterprise/workgroup/home-office, server/client, modem/networked/isolated, Internet/intranet access, and others.

user demand for new applications and hardware. New applications are enabled by and exploit advanced technology that is introduced into limited editions of platform models within a chosen market segment. Further, new types of applications will likely emerge that make novel eclectic use of these redefined platforms with their enhanced technologies. If successful, the new types of applications generate significant end-user enthusiasm and demand. As more users purchase these applications, platform purchases indirectly result. The purchases come from users new to the platform and from existing platform users that need to upgrade in order to meet the advanced technology requirements of the new applications in an acceptable fashion.

If the new type of application develops a sufficient user base, the user base may come to be viewed as its own emerging platform segment. In this manner, PC platform market segments are being continually redefined, as processor and platform designers attempt to track shifts in the way PCs are being used. The redefinition takes the form of additional novel platform segments and growth-motivated division of previously existing segments. Each new segment has optimized platform architectures tailored to its specific needs.

Supported by demand as well as driven by aggressive competition, successful advanced technology is cost-reduced and made more widely available, while new advanced technology is designed into future platform releases. This again encourages new applications, and the process runs full circle. Thus we can expect further expansions in new technology, new types of applications, and new platform segments, as the process continues in a positive growth spiral.

A major thrust of platform redefinition ongoing today is driven by diverse new applications exploiting the convergence of connectivity and content technologies. Applications are using the Internet, Web, and networking and other communications technologies to enable and enhance aspects of productivity, message passing, socialization, and access to remote and distributed databases. Applications rely on multimedia technologies to enhance their ease of use and to create positive memorable experiences that meet or exceed those from other competitive media technologies. Applications intended to entertain are attempting to turn the PC into a game platform, or a home entertainment center with hybrid TV and PC capabilities.

## NEXT-GENERATION PLATFORMS

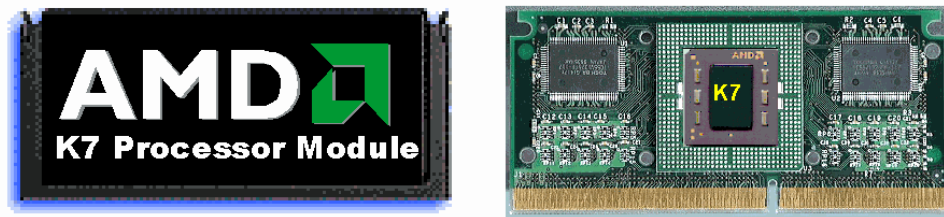
Bridging the gap between the previously discussed K6 platforms and the platforms beyond the millennium, lies K7-based platforms. K7 processors will continue to make microarchitectural enhancements and increase clock rates. Expected core clock rates for the initial shipments of K7 are at 500 MHz. In addition, K7 will require some significant changes in PC

platform designs. In particular, K7 platforms will use a variant of Digital's EV6 multiprocessor bus and the platform's processor and cache will be integrated into a processor module.

Figure 6.10 shows what the K7 Processor Module is expected to look like. This module integrates the highest-level cache on a common module card with the processor. Careful card design and component selection permits the highest-level cache to run at frequencies much higher than the 100-MHz Super 7 bus. The module is designed and manufactured by the processor vendor (AMD in this case) to ensure that the cache and processor combination is reliable. AMD plans for the K7 Processor Module to be mechanically compatible with the physical infrastructure for Intel's Slot 1.

*Other logic can be added to the module beside the cache. By integrating any logic that is processor specific, it becomes much easier for a system manufacturer to incorporate subsequent processor module generations.*

## AMD-K7™ Processor



- Driven by customer requirements
- Clock speeds in excess of 500 MHz
- Advanced bus interface, “Alpha” EV6 bus protocol
- Plan of record: slot “A” mechanically identical to Intel’s slot 1
- Enabling alternative platforms for 1999 and Beyond

**Figure 6.10** AMD-K7 PROCESSOR

Copyright 1997/1998, Advanced Micro Devices Inc., used with permission.

The K7processor module bus is expected to have a 64-bit data bus and a packetized command bus. The bus will likely use high-performance signaling techniques including clock forwarding, double-data rate transfers, and point-to-point connections. It is expected that the K7 bus will be designed for coupling to either a memory controller or a multiprocessor

*As new technology puts otherwise perfect digital “masters” of films in the hands of the public, expect the entertainment industry to ensure that embedded copy-righted content protection becomes standard.*

*Figure 6.5 on page 496 and Figure 6.6 on page 496, earlier in the chapter, illustrated the link to the home and its relationship to the Internet connectivity cloud.*

controller. At a presumed initial “fractional” speed of 250 MHz, the K7 bus will support a bandwidth to main memory of 2 GB/s.

In the near term, K6- and K7-based processor and platform research and development objectives will be centered on further growing the user base of PC platforms and to track perceived shifts in use among existing users. Key focus areas of these efforts are to reduce user frustration and customer support costs associated with platform configuration and system crashes, enable new applications that deliver an “enhanced user experience,” and to further the transition of the consumer PC platform into a simple-to-use entertainment appliance for “content access.”

K6- and K7-based entertainment-oriented PCs will attempt to increasingly shield the user from PC internal complexities. These devices will support rich graphical content through improved multimedia features, including 3D graphics, 3D audio, and streaming video. The graphics subsystem and a requisite DVD-ROM will support the display of “Hollywood” content at somewhat better than VCR quality. Such PCs will be equipped with improved communications, presently based on 56K modems or ISDN adapters, but shifting in the next few years to XDSL schemes. The communications functions of the platform will support voice, telephony, and conferencing. These technologies and Internet connectivity will enable the significant (but still limited) use of on-demand broadcast services and secure commerce.

The likely eventual pervasiveness of DVD-ROM will significantly raise users’ standard for multimedia quality on the PC with its use of MPEG-2 quality video and AC-3 audio. Internet entertainment content will be competing against these raised expectations. The delivery of competitive quality Internet multimedia content, and its ability to supplant TV and other entertainment forms, will be paced by the availability of affordable high-speed user access rates. User access communications rates are determined by the link between the user’s Internet Service Provider and the user’s home. This is the slowest link in the Internet connectivity “cloud.”

## A VISION OF PLATFORMS BEYOND THE MILLENNIUM

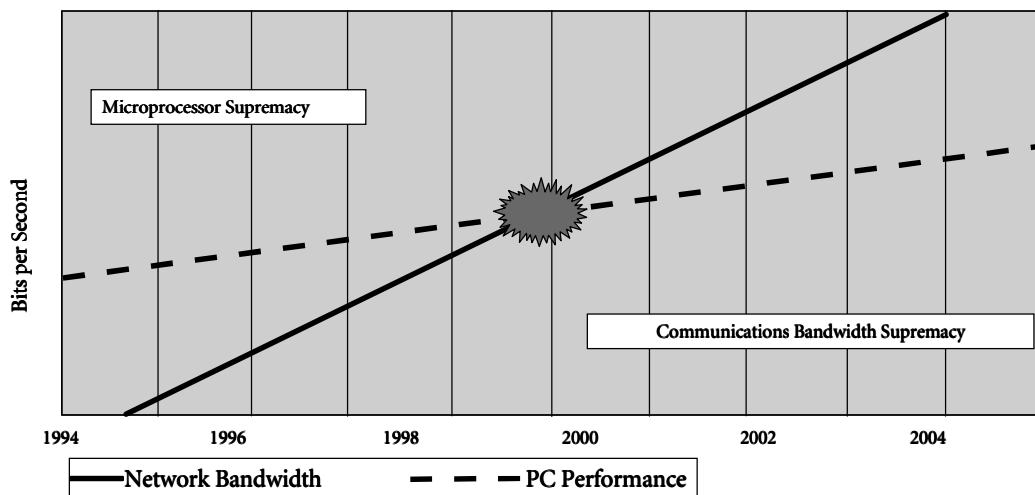
The remainder of this section examines how current trends and predictions may impact the PC platform and society, once the underlying technologies are mainstream and well beyond today’s demonstrations, field trials, and hobbyist experimentation. We have organized the following discussion loosely along the lines of Future Content, Connectivity, Hybrid Content/Connectivity, Platforms, and Platform Industry Directions.

## Future Content

Enabled by the authoring capability of future platforms, there will be an explosion in content for entertainment, productivity, and education. Much of the new content will make intensive use of multimedia, visualization techniques, and simulated environments. PC entertainment applications will be overwhelming competition for other free-time activities including TV, as we now know it. The trend to suppress the underlying technology will continue. Many applications will become invisible to the user, being indirectly selected based on what the user will perceive as a selection from different variations in content.

## Future Connectivity

A hope and promise of the computer industry is that in the next five to ten years, communications bandwidth will be dramatically more available than it is presently. In particular, it is expected that “fast” (1-10 Mbit/s) communications rates will be available at affordable rates to the masses. Figure 6.11 graphically depicts the relative growth of network bandwidth versus the growth of PC processing power. Industry visionary George Gilder has suggested that this shift in dominance, as represented by the crossover point shown in the figure, calls for completely rethinking systems architectures. In particular, he suggests that the focus of networked platforms be placed on what can be achieved via fast connectivity over what can be achieved via local processing.



**Figure 6.11** COMMUNICATIONS BANDWIDTH ASCENSION

Adapted with permission of Advanced Micro Devices Inc., from *Competing with Intel*, Copyright 1997, based on George Gilder's Telecosm articles as published in *Forbes* ASAP and available on the George Gilder Web Site.

*Connectivity for the last mile to the user in today's undeveloped countries will come about predominantly via Wireless Local Loop (WLL). WLL encompasses various forms of land-based wireless repeater technology.*

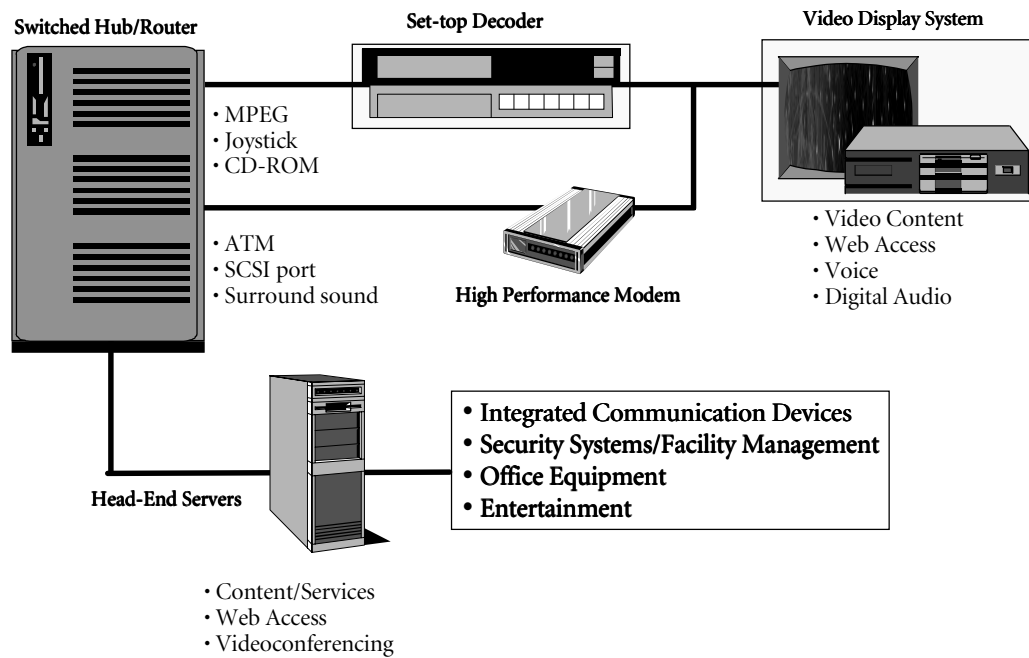
Another industry visionary, Nicholas Negroponte, has emphasized the ramifications of worldwide connectivity even at slow communications rates, see “The Third Shall Be First,” by Nicholas Negroponte, *Wired*, January 1998, p. 96. Negroponte predicts that worldwide school connectivity for education purposes will occur in the next few years, enabled by \$2,700 Geostationary-Earth-Orbiting satellites’ (GEOs) satellite links, and eagerly underwritten by loans from the World Bank. Longer term, Negroponte implies Low-Earth-Orbiting satellites (LEOs) will provide more pervasive connectivity until the eventual time when unlimited local telephony will eventually be available “in every civilized place” at low fixed rates.

Negroponte suggests that even at slow rates by U.S. standards, per-school access to Internet-connected world libraries would significantly impact the education programs of the poorest third-world countries. Negroponte implies that developing countries appreciate the significance of Internet connectivity better than in some developed nations, are therefore better poised for the digital world, and that “the ‘Third World’ five years from now may not be where you think it is.”

### *Future Hybrids of Content and Connectivity*

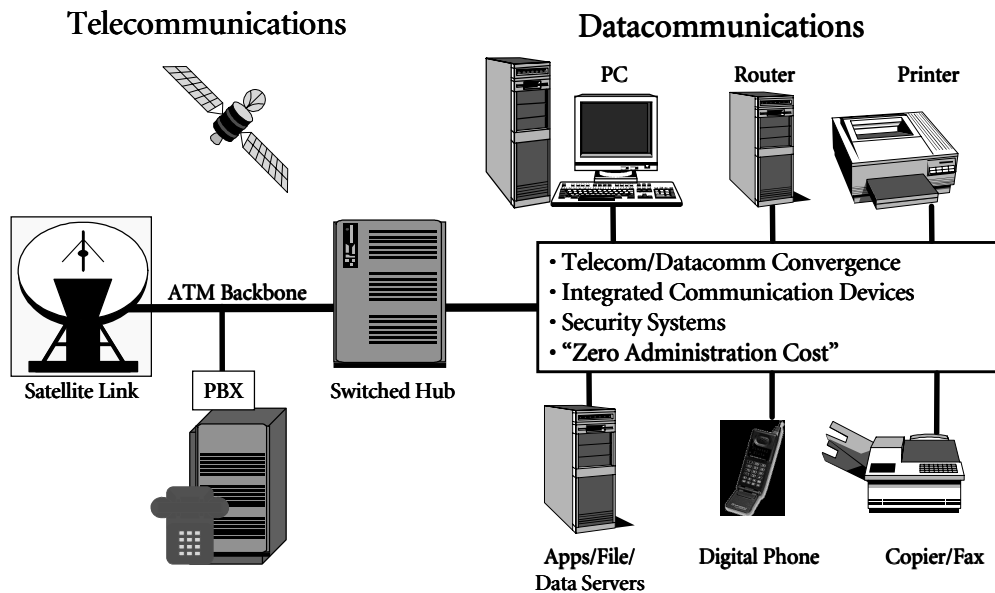
This wide availability of fast networking will encourage pervasive use of a number of Internet/intranet web-oriented applications that have developed a following, but are likely considered as presently at the field-trial-stage or as hobbyist by society at large. These applications include Web access for broadcasting of information and entertainment, secure commerce, socialization, video conferencing, collaborative computing, interactive gaming, casting of votes, and remote security monitoring and facility management. Figure 6.12 and Figure 6.13 illustrate this future view of fast-network-enabled Internet and intranet-connected computation.





**Figure 6.12** INTERNET-CONNECTED COMPUTATION

Adapted with permission of Advanced Micro Devices Inc., from *Competing with Intel*, Copyright 1997.



**Figure 6.13** INTRANET-CONNECTED COMPUTATION

Adapted with permission of Advanced Micro Devices Inc., from *Competing with Intel*, Copyright 1997.

Ubiquitous fast networking will also finally make practical several Internet-centric visions based on the pervasive integration or coordination of remote platforms and facilities over arbitrary geographic domains. This covers a broad spectrum of applications. On the simple side, minimalist network platforms (thin clients, Network Computers, information appliances, and NetPCs) can exploit storage and processor concentration on intranets to reduce the hardware, software, setup, and maintenance costs of the overall network. On the elaborate side, worldwide-connected computation could regularly treat remote platforms, facilities, and workgroups as real-time subsystems of a single geographically distributed multiprocessing platform.

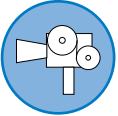
### Future Platforms

Content and connectivity will continue to strongly influence the definition of PC platforms. Continued expansion, redefinition, and blurring of market segments will occur as PC platforms are evolved to track changes in use. The Internet, networking, multimedia, Personal Communications Services (PCS), and Personal Digital Assistant (PDA) technologies will fuel continued new applications and uses. The complexity underlying PC


platforms will be buried as they infiltrate and replace the hardware, infrastructure, and business models behind present-day TV, telephones, and fax machines.

Accompanying the growth of widely available fast communications rates will be increased growth in the infrastructure of the Internet and of institutional intranets. This infrastructure will be in the form of switches, routers, and new classes of powerful remote compute and storage *head-end* servers for specialized services and content. Servers will integrate portions of the net-fabric by coming standard with integral gigabit-Ethernet routers and switches.

#### VIDEO ON CD-ROM

	<p>Atiq Raza, AMD CTO and Executive VP, replies to the question: "How will the use of servers grow in the future?"</p>
---	--

#### ARTICLE ON CD-ROM

	<p>To learn more about Future Platforms, see the article "Multimedia Storage Servers: A Tutorial," by D. James Gemmell, Harrick M. Vin, Dilip D. Kandlur, P. Venkat Rangan and Lawrence A. Rowe, <i>IEEE Computer</i>, Vol. 28, No. 5, May 1995, on the companion CD-ROM.</p>
--	---

Upon the achievement of affordable high-speed access to the Internet, the Entertainment PC will require further improvements in bus and interface bandwidths and system partitioning. As an example, Internet broadcast content will ultimately consist of multiple sub-streams including video, audio, control, and secondary content. Secondary content sub-streams include captioning, alternate languages, supplementary information and URL links. Platform architectures will be optimized for coupling each sub-stream directly to the appropriate platform subsystems, bypassing the processor and main memory whenever possible.

### *Platform Industry Directions for the Future*

Many aspects of contemporary industry direction will continue. Users will continue to buy content and connectivity first, software second, and hardware last. Accordingly, new platform technology will be continually introduced or made more widely available to raise platform baseline features, facilitating application development, and thereby enabling new applications. The success of compelling new applications requiring the new platform technology will in turn drive platform sales and market growth.

The PC industry as a whole will expand its market by capturing market share from other forms of content delivery. PC platforms will be refined until collectively the PC platform is the preferred content delivery instrument for all forms of consumer content. This will result in nothing less than the eventual assimilation of all but the most basic consumer electronics.

While many PC platforms will be capable of functioning as a unified consumer content delivery device, the more likely scenario is that specialty market segments and platforms will be developed for the various device classes. Specialty platforms of the new diverse PC market segments will capture the new installation and replacement market for “single function” devices such as TVs, VCRs, video games, high-fidelity sound systems, and telephones. These devices (as we know them today) will be absorbed into PCs in the same way that the typewriter, adding machine, and data terminal already have been absorbed. They will otherwise exist only as antiques, or will be produced in a format where they will be considered and priced like an incidental tool, gadget, or toy.

The PC industry (including processor, chipset, motherboard, and system vendors) has transcended its namesake role and is poised to begin the new millennium in the business of providing Platforms for Content and Connectivity. This new class of electronics devices will be appliances for content access and tools for content authoring that are implicitly connected to the intranet or an intranet. As the PC industry has evolved in an environment of rapid-pace product development and rollout, it is poised to rapidly take over the markets of today’s major consumer electronics.

## CHAPTER SUMMARY

In this final chapter, we examined the nature of platform evolution. First we surveyed a number of basic guidelines for optimizing platforms. We next illustrated how many of these optimization basics are used within a contemporary 3D-graphics platform. The 3D graphics study was made from both component and application perspectives. The component perspective characterized the role of each component, describing its principal operating paradigm and functions in the platform. The application perspective gave insight into how the components are orchestrated and optimized at the platform level. This was done through a study of how the individual platform components are collectively and dynamically managed to implement a high-performance multi-stage 3D Graphics Pipeline.

We also presented how the changes in content and connectivity are rapidly evolving the platform, as optimizations are continually being required to track the changes in application requirements and new uses of the platform. New standards are emerging for networking, communications, multimedia, graphics, as well as for processors, memory, and peripheral interconnect. Platform designs and infrastructure are being optimized and redefined in a continual effort to support the new standards, enable higher speed connectivity, sustain higher processor performance, provide higher bandwidth data transfers, and display rich graphical content.