International
Bestseller

Second Edition
Now Includes:

New chapters
on MPEG 1
and MPEG 2

Video
Conferencing
Standards

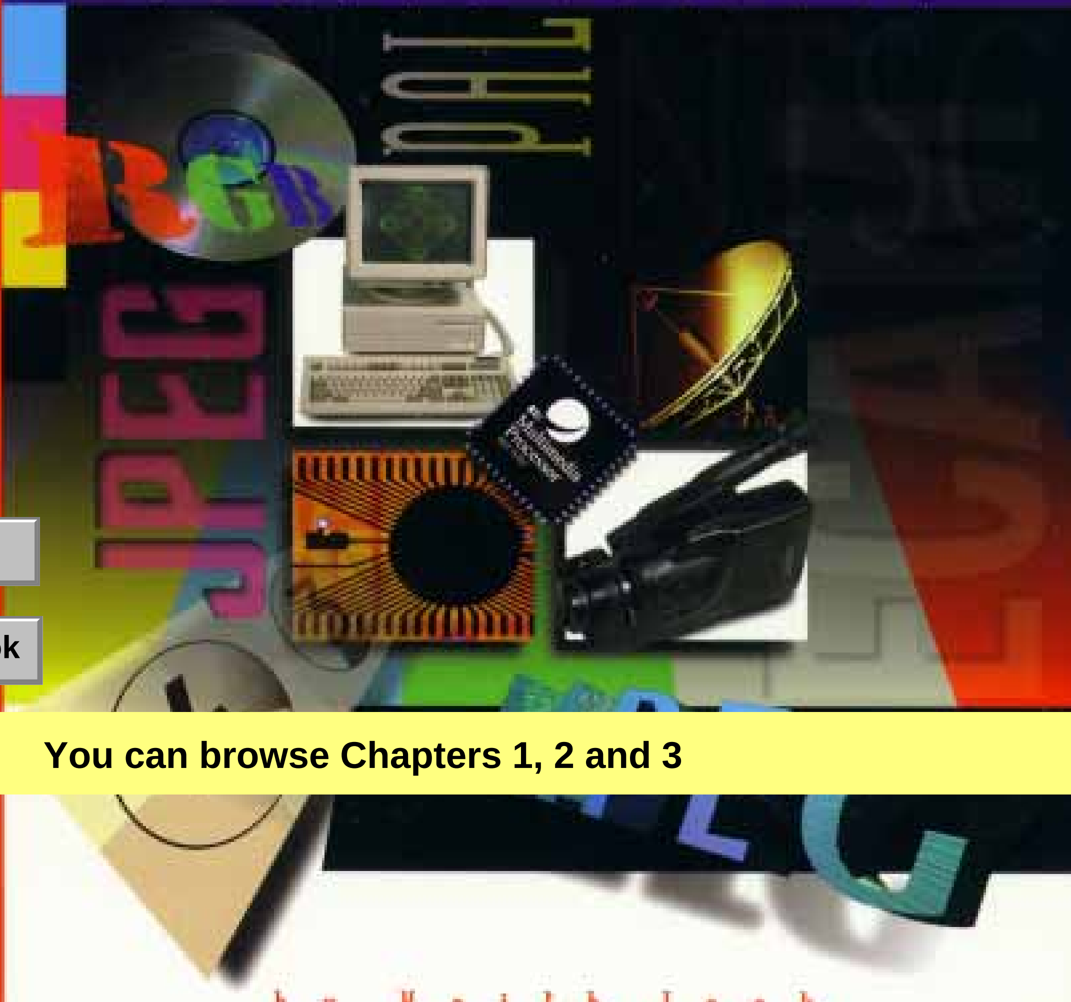
Windows and

HARRIS

Video Demystified

A Handbook
for the
Digital Engineer

SECOND EDITION



by Keith Jack

Video Demystified

A Handbook
for the
Digital Engineer

Second Edition

by Keith Jack

HighText
An imprint of LLH Technology Publishing

Eagle Rock, Virginia

HARRIS
SEMICONDUCTOR

Visit the LLH Web Site

Visit the Harris Web Site

About the Author

Keith Jack has been the architect for numerous multimedia ICs for the PC and consumer markets, including the world's first single-chip NTSC/PAL digital encoder/decoder (Brooktree Corporation). Currently with Harris Semiconductor, he is now working on next-generation multimedia products.

Visit the Author's Web Site

Library of Congress Cataloging-in-Publication Data
Jack, Keith, 1955–
Video demystified: a handbook for the digital engineer / Keith Jack. -2nd ed.
p. cm.
Includes bibliographical references and index.
ISBN 1-878707-36-1 casebound
ISBN 1-878707-23-X paperbound
1. Digital television. 2. Microcomputers. 3. Video recordings--Data processing. I. Title.
TK6678.J33 1996
621.388--dc20 96-076446
CIP

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

Cover design: Sergio Villarreal
Developmental editing: Carol Lewis
Production: Lynn Edwards

ISBN: 1-878707-23-X (paperbound)
ISBN: 1-878707-36-1 (casebound)
Library of Congress catalog card number: 96-076446

Foreword

The foreword to the first edition of this book started with “Video on computers is a hot topic.” If it was hot then, it is now white hot. The effect that multimedia products have had on the growth rate and pervasiveness of personal computer sales has been spectacular.

To effectively service the multimedia market, companies such as Harris Semiconductor need engineers who understand the analog and digital domains of audio and video, as well as the current and emerging compression standards. It is still a case of helping to “demystify video.” Standard texts are still scarce, and the subject material highly dynamic.

Keith Jack is with Harris Semiconductor, helping to define our new multimedia products. In this second edition of *Video Demystified*, he has expanded the subject material considerably by covering MPEG and video conferencing, arguably two of the most recent important advances in multimedia technology.

The first edition became the standard reference for the video engineer. This second edition will be an essential addition for all multimedia and professional video engineers. We anticipate that the reader will find this new text immensely useful.

Geoff Phillips
Vice President, Multimedia Products
Harris Semiconductor

Acknowledgments

The development of an extensive work such as this requires a great deal of research and the cooperation and assistance of many individuals. It's impossible to list everyone who contributed, but I would like to acknowledge all those who provided technical assistance and advice, as well as moral support, and David Richards for the overscan test pattern.

Contents

Chapter 11 • *MPEG 2 503*

Chapter 1

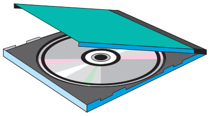
Introduction

A popular buzzword in the computer world is "convergence"—the intersection of various technologies that, until recently, were unrelated (see Figure 1.1). Video is a key element in the convergence phenomenon. Whether you call it multimedia, "PCTV," desktop video, digital video, or some other term, one thing is certain: we are now entering an era of digital video communications that will permeate both homes and businesses. Although technical hurdles still remain, most agree that the potential for this technology has few limits.

Implementing video on computers is not without its problems, and digital engineers working on these problems often have had little previous exposure to or knowledge of video. This book is a guide for those engineers charged with the task of understanding and implementing video features into next-generation computers. It concentrates both on system issues, such as getting video into and out of a computer environment, and video issues, such as video standards and new processing technologies.

This new edition of *Video Demystified* is accompanied by a CD-ROM, readable on both PCs and Macintosh computers. It contains numerous files to assist in testing and evaluating video systems. These files include still images at various resolutions, QuickTime movies, source code for MPEG, H.261, and H.263 encoders/decoders, and several software tools. These are described in Appendix F.

In addition, supplemental reference information is also found on the CD-ROM, keyed to the appropriate book chapter. When supplemental information is available for a particular chapter, this icon will alert you.

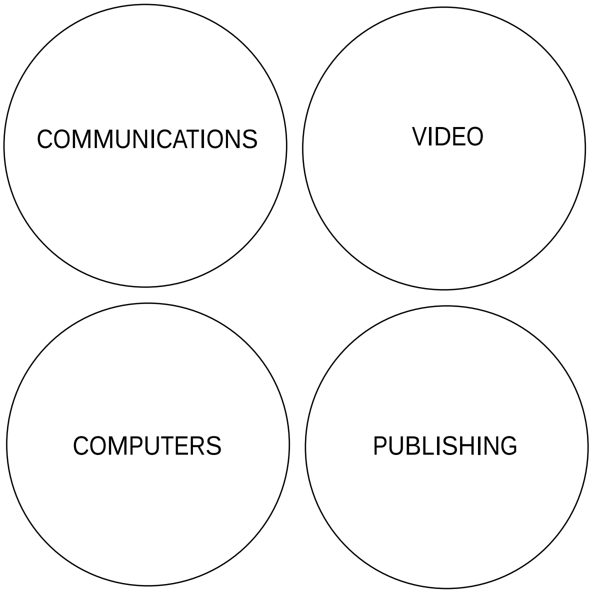


These files, in a PDF format, are found on the CD-ROM in a folder called "Goodies." They can be viewed using Adobe™Acrobat™ Reader. Files for the Reader (both Macintosh and Windows versions) can be found on the CD-ROM in a folder called "ACRO_V2."

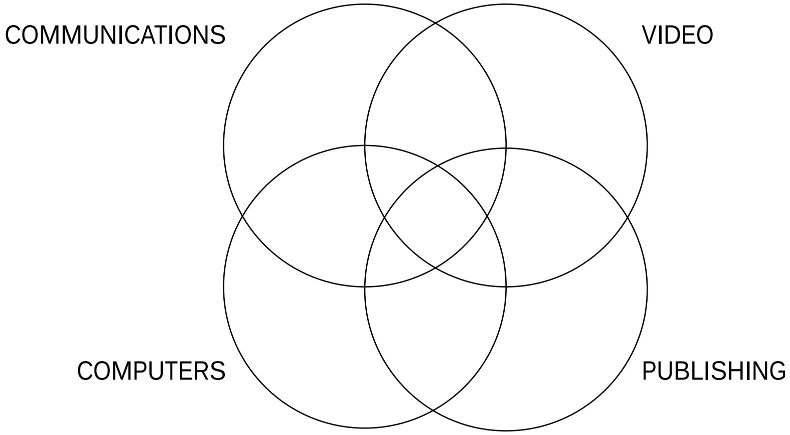
Emphasis is placed on the unique requirements of the computer environment. The book can be used by system design engineers who need or want to learn video, VLSI design engineers who want to build video products, or anyone who wants to evaluate or simply know more about such systems.

Chapter 2 assumes the reader has a working knowledge of the design of computer graphics systems; although not necessary for the other chapters, it is helpful.

A glossary of video terms has been included for the user's reference. If you encounter an unfamiliar term, it likely will be defined in the glossary.



(A) Yesterday



(B) Tomorrow

Figure 1.1. Technology Overlap of the Communications, Video, Computer, and Publishing Markets.

Some of the better-known applications for merging video and computers are listed below. At this point, everyone is still waiting for the broad-based application that will make video-in-a-computer a "must have" for users, similar to the spreadsheet and word processor.

Each of these applications requires real-time video decompression and possibly real-time compression. Some applications can use software compression and/or decompression, while others will require dedicated hardware for that function.

- *Video help windows.* Rather than the conventional text-only on-line help available from within applications, video help windows contain digitized audio and video, allowing moving images to be displayed. A user can see what to do, and see what the expected results will be, before actually executing a command.
- *Video notes within a document.* Video and audio can now be transferred within e-mail (electronic mail) or a document to another user. A "video note" can be included to clarify a point or just to say "Hi!" to your co-worker.
- *Video teleconferencing.* Video teleconferencing over a computer network or telephone lines allows users to interact more personally and be able to use visual aids to make a point more clearly. This application requires that a small video camera, speakers, and microphone be mounted within or near the display.
- *CD-ROM and network servers.* Digitized and compressed video and audio may be stored on CD-ROM or on a network server and later may be retrieved by any user and modified for inclusion within a presentation or document. With compression, it is possible to play back a compressed video over a network. However, performance is dependent on the number of users and network bandwidth. In some cases, it's acceptable to do a non–real-time download of the compressed video, and perform the decompression locally for viewing. CD-ROMs containing digital audio and video clips—similar to "clip art" files in desktop publishing—are now available. They include various types of audio and video for inclusion in presentations. Once the CD-ROM is purchased, the owner may have the right to use its contents whenever needed.
- *Video viewing.* This application allows one or more channels of television to be viewed, each with its own window. Users such as stock brokers would use this application to monitor business news while simultaneously using the computer to perform transactions and analysis. No compression or decompression is really required, just the ability to display live video in a window.
- *Video editing.* Some users want the ability to edit video on-line, essentially replacing multiple tape decks, mixers, special-effects boxes, and so on. This application requires up to three video windows, depending on the level of editing sophistication, and the highest video quality.
- *Video tutorials/interactive teaching.* Interactive video allows the user to learn at his or her own pace. Video tutorials allow on-line explanations, optionally in an interactive mode, of almost any topic from grammar to jet engines.

In addition to computers, audio/video compression and decompression technology has prompted new features and products in the consumer market:

- *Digital Video via Satellite (DBS)*. Several ventures now transmit compressed digital audio and video, offering up to 150 channels or more.
- *Digital Video via Cable*. Work is progressing on upgrading cable systems to handle up to 500 channels and various interactive requirements.
- *Digital Video via Phone*. Phone systems are being upgraded to handle digital video and interactivity, allowing them to compete with cable companies.
- *Digital VCRs*. VCRs that store digital audio and video on tape will be available in 1996.
- *Videophones*. Videophones are now available to the consumer.
- *Video CDs*. An entire movie can now be stored digitally on two CDs. With DVD, a 2-hour movie requires only a single CD.

An example of the impact of merging video and computers can be seen in the video editing environment, as shown in Figures 1.2 through 1.4. The analog solution, shown in Figure 1.2, uses two VCRs and an editing controller, which performs mixing, special effects, text overlays, etc. The output of the edit controller is stored using another VCR. Note that the two VCRs must be genlocked together. (Genlocking means getting the video signals to line up correctly—refer to Chapter 5 for more information on this topic.)

As computers can now generate video, they can be used to provide one of the video sources, as shown in Figure 1.3. Note that the video output of the computer must be able to be genlocked to the VCR. If not, a time base corrector for each video input source (which includes genlock capability) is required.

Advances in software now allow editing and special effects to be performed using the computer, offering the ideal video editing environment as shown in Figure 1.4. In this case, each video source is individually digitized, real-time compressed, and stored to disk. Any number of video sources may be supported, since they are processed one at a time and need not be genlocked together. Editing is done by decompressing only the frames of each video

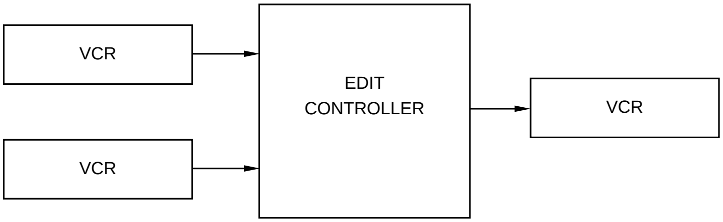


Figure 1.2. Analog Video Editing.

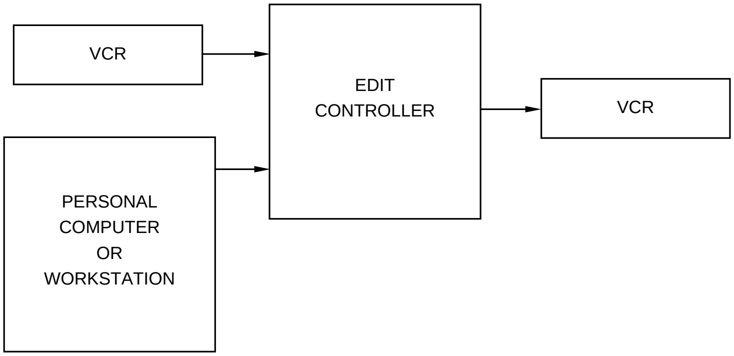


Figure 1.3. Video Editing with a Computer Providing a Video Source.

source required for the edit, performing the processing, compressing the result, and storing it back to the disk.

In addition to the standard consumer analog composite and Y/C (S-video) video formats, video editing products for the professional market should support several other video interfaces: digital composite video, digital component video, and the various analog YUV video formats. These formats are discussed in detail in later chapters.

The remainder of the book is organized as follows:

Chapter 2 discusses various *architectures for incorporating video* into a computer system, along with the advantages and disadvantages of each. Also reviewed is why scaling, interlaced/noninterlaced conversion, and field rate conversion are required. A review of commercially available VLSI solutions for adding video capabilities to computers is presented (CD-ROM).

Chapter 3 reviews the common *color spaces*, how they are mathematically related, and when a specific color space is used. Color spaces reviewed include RGB, YUV, YIQ, YCbCr, YDbDr, HSI, HSV, HSL, and CMYK. Considerations for converting from a non-RGB to a RGB color space and gamma correction are discussed. A review of commercially available VLSI solutions for color space conversion is presented (CD-ROM).

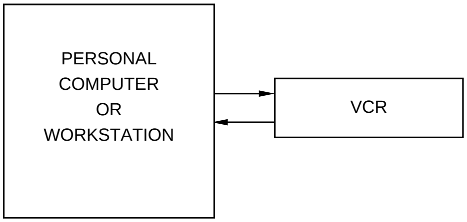


Figure 1.4. Video Editing Using the Computer.

Chapter 4 is an *overview of analog video* signals, serving as a crash course. The NTSC, PAL and SECAM composite analog video signal formats are reviewed. In addition, SuperNTSC™ (a trademark of Faroudja Labs), the PALplus standard, and color bars are discussed.

Chapter 5 covers digital techniques used for the *encoding of NTSC and PAL* color video signals. Also reviewed are various video test signals, video encoding parameters, “clean encoding” options, timecoding, and closed captioning. Commercially available NTSC/PAL encoders are also reviewed (CD-ROM).

Chapter 6 discusses using digital techniques for the *decoding of NTSC and PAL* color video signals. Also reviewed are various luma/chroma (Y/C) separation techniques and their trade-offs. The brightness, contrast, saturation, and hue adjustments are presented, along with techniques for improving apparent resolution. Commercially available NTSC/PAL decoders are reviewed (CD-ROM).

Chapter 7 reviews the background development and applications for *composite digital video*, sometimes incorrectly referred to as D-2 or D-3 (which are really tape formats). This chapter reviews the digital encoding and decoding process, digital filtering considerations for bandwidth-limiting the video signals, video timing, handling of ancillary data (such as digital audio), and the parallel/serial interface and timing. It also discusses the SMPTE 224M and 259M standards. A review of commercially available VLSI solutions for digital composite video is presented (CD-ROM).

Chapter 8 discusses *4:2:2 digital component video*, sometimes incorrectly referred to as D-1 (which is really a tape format), providing background information on how the sample rates for digitizing the analog video signals were chosen and the international efforts behind it. It also reviews the encoding and decoding process, digital filtering considerations for bandwidth-limiting the video signals, video timing, handling of ancillary data (such as digital audio, line numbering, etc.), and the parallel/serial interface and timing. It also discusses the 4:4:4 format, and the 10-bit environment. The 4:4:4:4 (4 × 4) digital component video standard is also covered. A review of commercially available VLSI solutions for digital component video is presented (CD-ROM).

Chapter 9 covers several *video processing* requirements such as real-time scaling, deinterlacing, noninterlaced-to-interlaced conversion, field rate conversion (i.e., 72 Hz to 60 Hz), alpha mixing, and chroma keying.

Chapter 10 is a “crash course” on the *MPEG 1* standard and also compares MPEG with JPEG, motion JPEG, and H.261 compression standards.

Chapter 11 is a “crash course” on the *MPEG 2* standard.

Chapter 12 discusses *Video Teleconferencing*, over ISDN, covering H.320; H.261 (video); G.711, G.722, G.728 (audio); and H.221, H.230, H.231, H.233, H.242, and H.243 (control).

Chapter 13 covers *Video Teleconferencing* over normal phone lines, discussing H.263 (video).

Chapter 14 reviews the analog and digital “high-definition production standards” (SMPTE 240M, 260M, and ITU-R BT.709)

Chapter 2

Video and the Computer Environment

In the early days of personal computers, televisions were used as the fundamental display device. It would have been relatively easy at that time to mix graphics and video within the computer. Many years later, here we are at the point (once again) of wanting to merge graphics and video. Unfortunately, however, the task is now much more difficult!

As personal computers became more sophisticated, higher resolution displays were developed. Now, 1280 × 1024 noninterlaced displays have become common, with 1600 × 1280 and higher resolutions now upon us. Refresh rates have increased to 75 Hz noninterlaced or higher.

However, in the video world, most video sources use an *interlaced* display format; each frame is scanned out as two fields that are separated temporally and offset spatially in the vertical direction. Although there are some variations, NTSC (the North American and Japanese video standard in common use) color composite video signals have a refresh rate of 60 Hz interlaced (actually 59.94 Hz), whereas PAL and SECAM (used in Europe and elsewhere) color composite video signals have a refresh rate of 50 Hz interlaced.

Digital component video and digital composite video were developed as alternate methods of transmitting, processing, and storing video digitally to preserve as much video quality as possible. The basic video timing parameters, however, remain much the same as their analog counterparts. Video and still image compression techniques have become feasible, allowing the digital storage of video and still images within a computer or on CD-ROM. Figure 2.1 illustrates one possible system-level block diagram showing many of the audio and video input and output capabilities now available to a system designer.

To incorporate video into a personal computer now requires several processing steps, with trade-offs on video quality, cost, and functionality. With the development of the graphical user interface (GUI), users expect video to be treated as any other source—displayed in a window that can be any size and positioned anywhere on the display. In most cases some type of video scaling is required. On the output side, any portion of the display could be selected to be output to video, so scaling on this side is also required. Computer users want their displays to be noninterlaced to reduce fatigue due to refresh flicker, requiring interlaced-to-noninterlaced conversion (deinterlacing) on the video input side and noninterlaced-to-interlaced conversion on the video output side. The refresh rate difference between NTSC/PAL/SECAM video (50 or 60 Hz interlaced) and the computer display (60–80 Hz noninterlaced) must also be considered.

This chapter reviews some of the common architectures and problems for video input and output. (Of course, there are as many possible architectures as there are designers!)

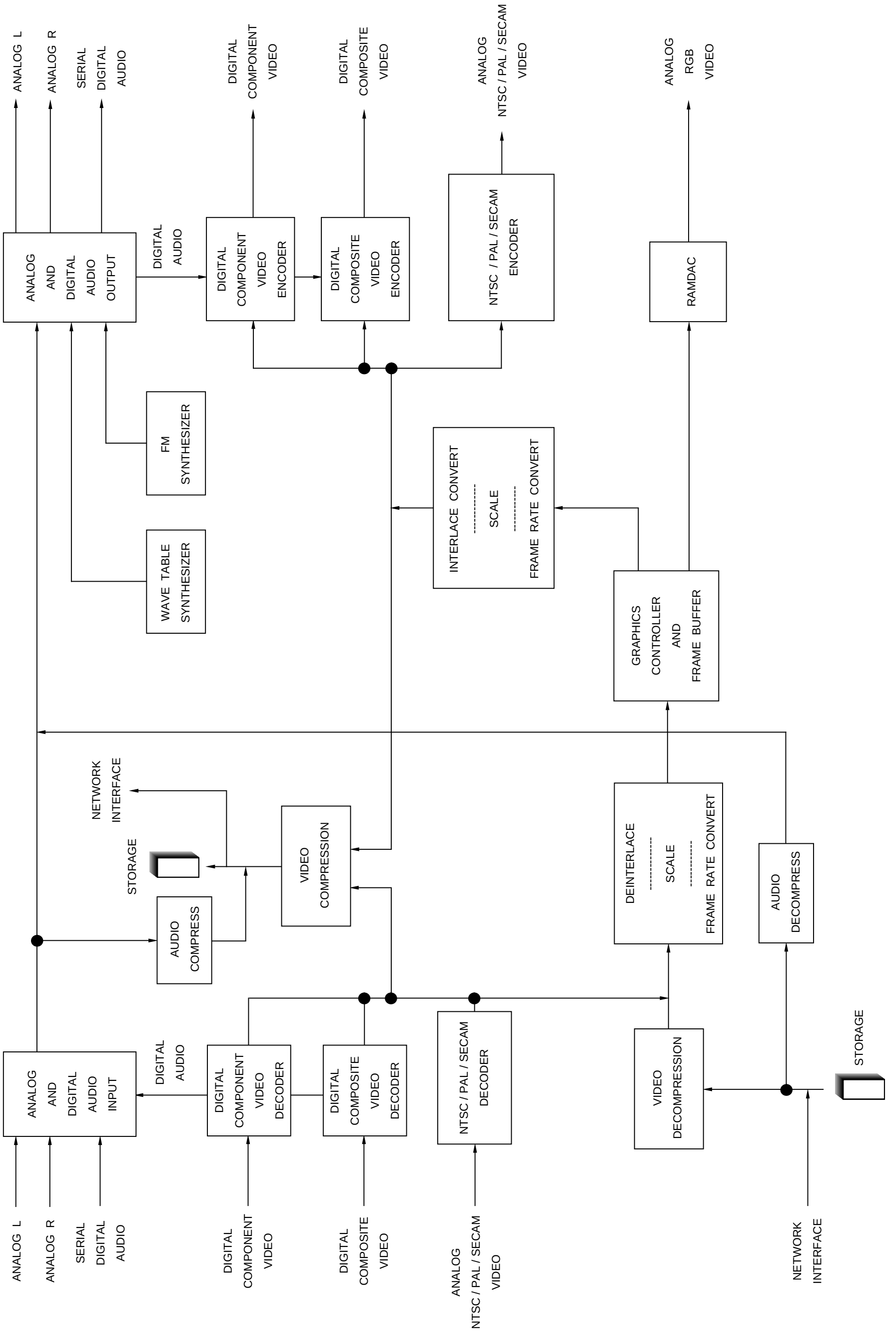


Figure 2.1. Example of Computer Architecture Illustrating Several Possible Video Input and Output Formats.

Displaying Real-Time Video

There are several ways to display live video in the computer environment, each with its own trade-offs regarding functionality, quality, and cost. The video source for each of these example architectures could be a NTSC, PAL, or SECAM decoder (decoding video from an analog video tape recorder, laserdisk player, camera, or other source), a MPEG decoder (for decompressing real-time video from a CD-ROM or disk drive), a teleconferencing decoder (for decompressing teleconferencing video from a network or phone line), or a digital component video or digital composite video decoder (for decoding digital video from a digital video tape recorder). As work progresses on video compression and decompression solutions, there undoubtedly will be other future sources for digital video.

In many consumer NTSC/PAL/SECAM decoders, additional circuitry is included to sharpen the image to make it more pleasing to view. However, this processing reduces video compression ratios and should not be used if the video is to be compressed. In the computer environment, the ideal place for such video-viewing-enhancement circuitry is within the RAMDAC, which drives the computer display. MPEG, JPEG, and video teleconferencing decoders currently do not include circuitry to support brightness, contrast, hue, and saturation adjustments. If each video source does not have these user adjustments supported within the RAMDAC or elsewhere, there is no way to compensate for nonideal video sources.

Simple Graphics/Video Overlay

An example architecture for implementing simple overlaying of graphics onto video is shown in Figure 2.2. This implementation requires that the timing of the video and computer graphics data be the same; if the incom-

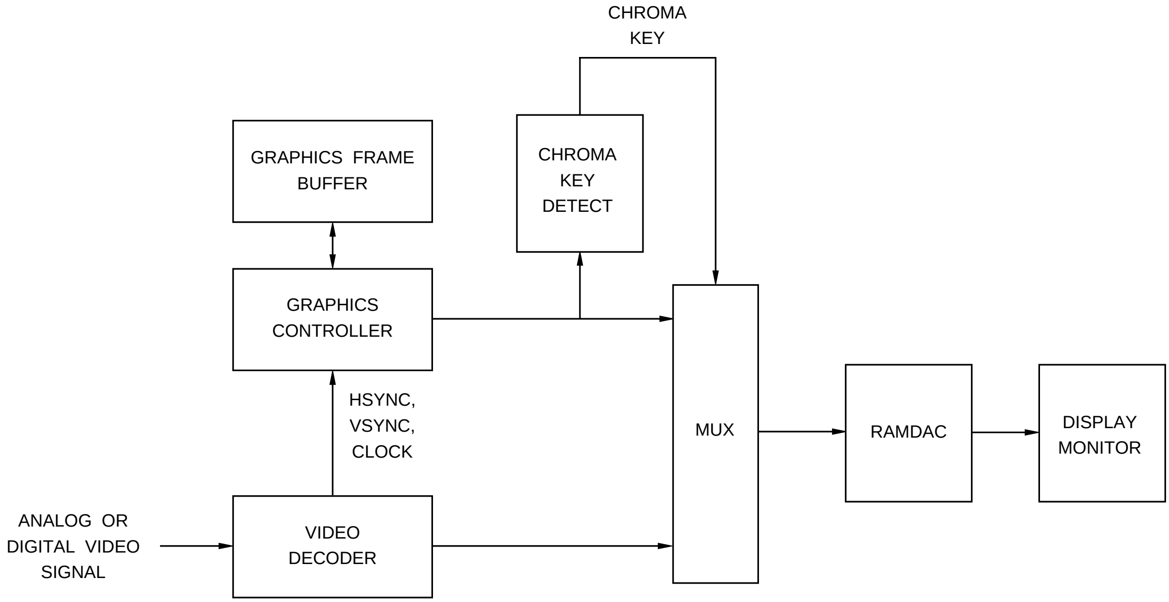


Figure 2.2. Example Architecture of Overlaying Graphics Onto Video (Digital Mixing).

ing video signal is 50 Hz or 60 Hz interlaced, the graphics controller must be able to operate at 50 Hz or 60 Hz interlaced. The display is also limited to a resolution of approximately 640 × 480 (NTSC) or 768 × 576 (PAL and SECAM); resolutions for other video formats will be different. Note that some computer display monitors may not support refresh rates this low. Timing can be synchronized by recovering the video timing signals (horizontal sync, vertical sync, and pixel clock) from the incoming video source and using them to control the video timing generated by the graphics controller. Note that the graphics controller must have the ability to accept external video timing signals—some low-end graphics controllers do not support this requirement.

A chroma key signal is used to switch between the video or graphics data. This signal is generated by detecting on a pixel-by-pixel basis a specific color (usually black) or pseudo-color index (usually the one for black) in the graphics frame buffer. While the specified chroma key is present, the incoming video is displayed; graphics information is displayed when the chroma key is not present. The RAMDAC processes the pixel data, performing gamma correction and generating analog RGB video signals to drive the computer display. If the computer system is using the pseudo-color pixel data format, the RAMDAC must also convert the pseudo-color indexes to 24-bit RGB digital data.

Software initially fills the graphics frame buffer with the reserved chroma key color. Subsequent writing to the frame buffer by the graphics processor or MPU changes the pixel values from the chroma key color to what is written. For pixels no longer containing the chroma key color, graphics data is now displayed, overlaid on top of the video data. This method works best with applications that allow a reserved color that will not be used by another application. A major benefit of this implementation is that no changes to the operating system are required.

Rather than reserving a color, an additional bit plane in the frame buffer may be used to generate the chroma key signal directly (“1” = display video, “0” = display graphics) on a pixel-by-pixel basis. Due to the additional bit plane, there may be software compatibility problems, which is why this is rarely done.

Note that the ability to adjust the gamma, brightness, contrast, hue, and saturation of the video independently of the graphics is required. Otherwise, any adjustments for video by the user will also affect the graphics.

Higher-end systems may use alpha mixing (discussed in Chapter 9) to merge the graphics and video information together. This requires the generation of an 8-bit alpha signal, either by the chroma key detection circuitry or from additional dedicated bit planes in the frame buffer. The multiplexer would be replaced with an alpha mixer, enabling soft switching between graphics and video. If the resulting mixed graphics and video signals are to be converted to NTSC/PAL/SECAM video signals, alpha mixing is very desirable for limiting the bandwidth of the resulting video signal.

A problem occurs if a pseudo-color graphics system is used. Either the true-color video data from the video decoder must be converted in real-time from true-color to pseudo-color (the color map must match that of the graphics system), or the pseudo-color graphics data must be converted to true-color before mixing with the video data and a true-color RAMDAC used. Because of this, low-cost systems may mix the graphics and video data in the analog domain, as shown in Figure 2.3.

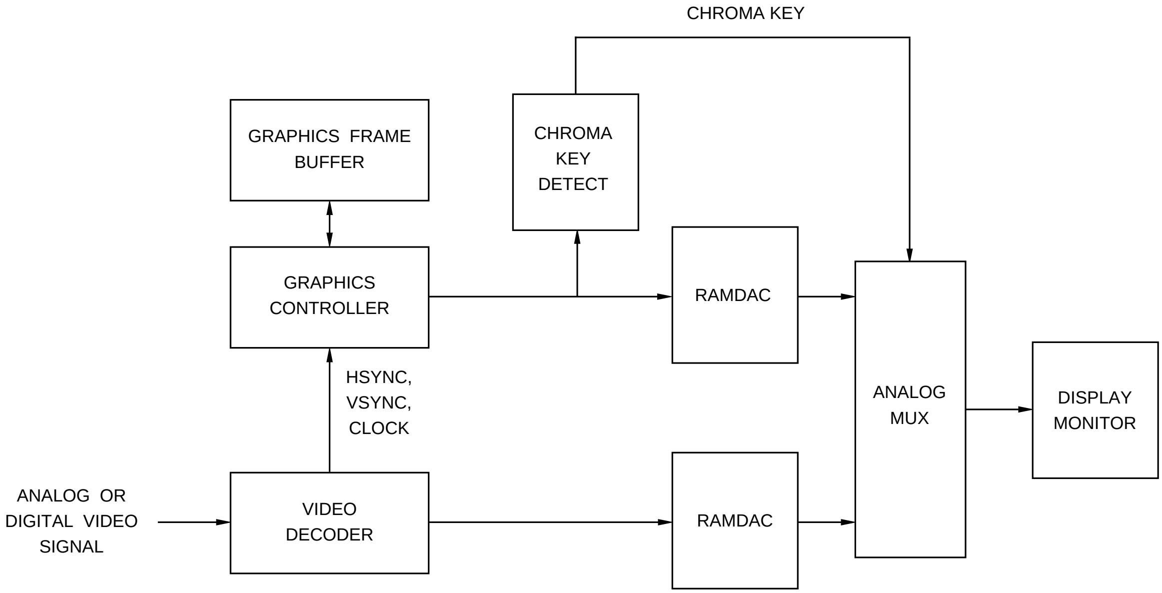


Figure 2.3. Example Architecture of Overlaying Graphics Onto Video (Analog Mixing).

Separate Graphics/Video Frame Buffers

This implementation (Figure 2.4) is more suitable to the standard computer environment, as the video is scaled to a window and displayed on a standard high-resolution monitor. The video frame buffer is typically the same size as the graphics frame buffer, and display data is output from both the video and graphics frame buffers synchronously at the same rate (i.e., about 135 Mpixels per second for a 1280 × 1024 display). This architecture is suited to the current graphical user interfaces (GUI) many systems now use. Switching between the video or graphics data may be done by a chroma key or alpha signal, as discussed in the previous example, or by a window-priority encoder that defines where the video window is located on the display (video is displayed inside the window and graphics is displayed outside the window). This architecture has the advantage that the graphics and video frame buffers may be independently modified.

The video frame buffer must be large enough to hold the largest video window size, whether it is a small window or the entire display. If a high-resolution display is used, the user may want the option of scaling up the video to have a more reasonable size window or even to use the entire display. Note that scaling video up by more than two times is usually not desirable due to the introduction of artifacts—a result of starting with such a bandwidth-limited video signal.

As the video source is probably interlaced, and the graphics display is noninterlaced, the video source should be converted from an interlaced format (25- or 30-Hz frame rate) to a noninterlaced format (50- or 60-Hz frame rate).

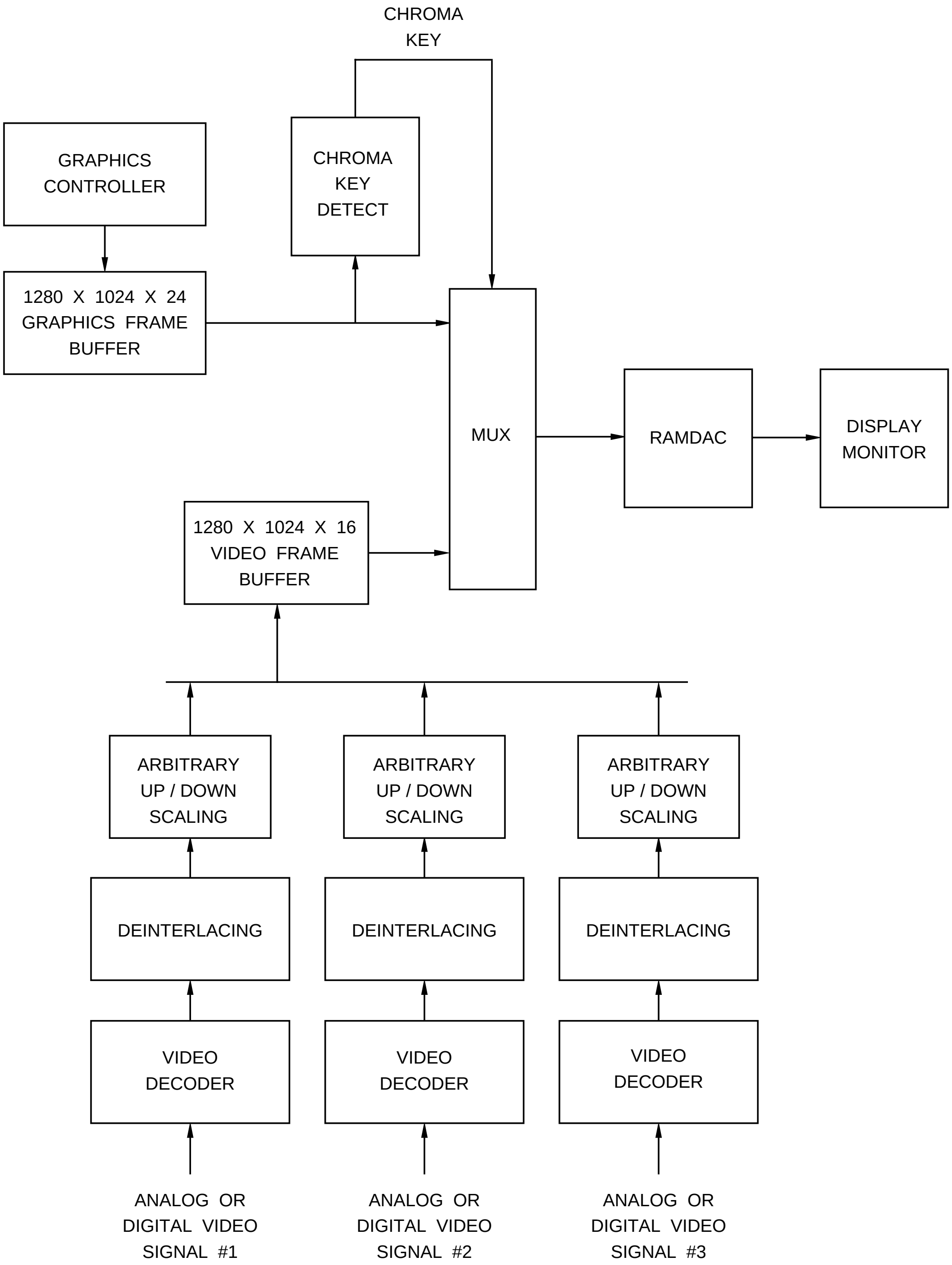


Figure 2.4. Example Architecture of Separate Graphics / Video Frame Buffers (Video Frame Buffer Size = Graphics Frame Buffer Size).

There are several ways of performing the deinterlacing. The designer must trade off video quality versus cost of implementation. Today's GUI environment also often requires scaling the video to an arbitrary-sized window; this also means a trade-off of video quality versus cost.

The video frame buffer is usually double-buffered to implement simple frame-rate conversion and eliminate "tearing" of the video picture, since the frame rate of the video source is not synchronized to the graphics display. After deinterlacing, the video will have a frame rate of 60 Hz (NTSC) or 50 Hz (PAL and SECAM) versus the 70–80 Hz frame rate of the graphics system. Ideally, the video source is frame-rate converted to generate a new frame rate that matches the computer display frame rate. In reality, frame-rate conversion is complex and expensive to implement, so most solutions simply store the video and redisplay the video frames at the computer display rate. The video frame therefore must be stored in a frame buffer to allow it to be displayed for multiple computer display frames.

Tearing occurs when a video picture is not from a single video frame, but rather is a portion of two separate frames. It is due to the updating of the video frame buffers not being synchronized to the graphics display. Switching between the video frame buffers must be done during the display vertical retrace interval only after the video frame buffer that is to be used to drive the display has been completely updated with new video information.

Alpha mixing (discussed in Chapter 9) may be used in some high-end systems to merge the graphics and video information together. An 8-bit alpha signal is generated, either by the chroma key detection circuitry or from additional dedicated bit planes in the frame buffer. An alpha mixer replaces the multiplexer, permitting soft switching between graphics and video. If the resulting mixed graphics and video signals are to be converted to NTSC/PAL/SECAM video signals, alpha mixing is very desirable for limiting the bandwidth of the resulting video signal.

If a pseudo-color graphics system is used, the pseudo-color graphics data must be converted to true-color before mixing with the video data, and a true-color RAMDAC used. Alternately, the true-color video data may be converted in real-time from true-color to pseudo-color (the color map must match that of the graphics system), although this is rarely done since it is difficult to do and affects the video color quality.

Two problems with this architecture are the cost of additional memory to implement the video frame buffer and the increase in bandwidth into the video frame buffer required when more than one video source is used. Table 2.1 illustrates some bandwidth requirements of various video sources. Due to the overhead of accessing the video frame buffer memory (such as the possible requirement of read–modify–write accesses rather than simple write-only accesses) these bandwidth numbers may need to be increased by two times in a real system. Additional bandwidth is required if the video source is scaled up; scaling the video source up to the full display size may require a bandwidth into the frame buffer of up to 300–400 Mbytes per second (assuming a 1280 × 1024 display resolution). Supporting more than one or two live video windows is difficult!

An "ideal" system would support multiple video streams by performing a "graceful degradation" in capabilities. For example, if the hardware can't support the number of full-featured video windows that the user wants, the windows could be made smaller or of reduced video quality. Note that the ability to adjust the gamma, brightness, contrast, hue, and satura-

Resolution		Bandwidth	
Total Resolution	**Active Resolution**	**MBytes / sec (burst)**	**MBytes / sec (average)**
ITU-R BT.601 (30 Frames per Second)			
QSIF	176 × 120	1.68	1.27
SIF	352 × 240	6.74	5.07
858 × 525	720[1] × 480	27.0	20.74
ITU-R BT.601 (25 Frames per Second)			
QSIF	176 × 144	1.69	1.27
SIF	352 × 288	6.74	5.07
864 × 625	720[1] × 576	27.0	20.74
Square Pixel (30 Frames per Second)			
QSIF	160 × 120	1.53	1.15
SIF	320 × 240	6.13	4.61
780 × 525	640 × 480	24.55	18.43
Square Pixel (25 Frames per Second)			
QSIF	192 × 144	1.84	1.38
SIF	384 × 288	7.36	5.53
944 × 625	768 × 576	29.5	22.12

Table 2.1. Bandwidths of Various Video Signals. Average column indicates entire frame time is used to transmit active video. 16-bit 4:2:2 YCbCr format assumed. Multiply these numbers by 1.5 if 24-bit RGB or YCbCr data is used. [1]704 true active pixels.

tion of the video independently of the graphics is required—otherwise, any adjustments for video by the user will also affect the graphics.

One variation on this architecture that solves the bandwidth problem into the video frame buffer allows each video input source to have its own video frame buffer. This approach, however, increases memory cost in direct proportion to the number of video sources to be supported.

A second variation, shown in Figure 2.5, is to scale the video down to a specific size (for example 320 × 240) prior to storing in the video frame buffer. This limits the amount (and cost) of the video frame buffer. The video is then scaled up or down to the desired resolution for

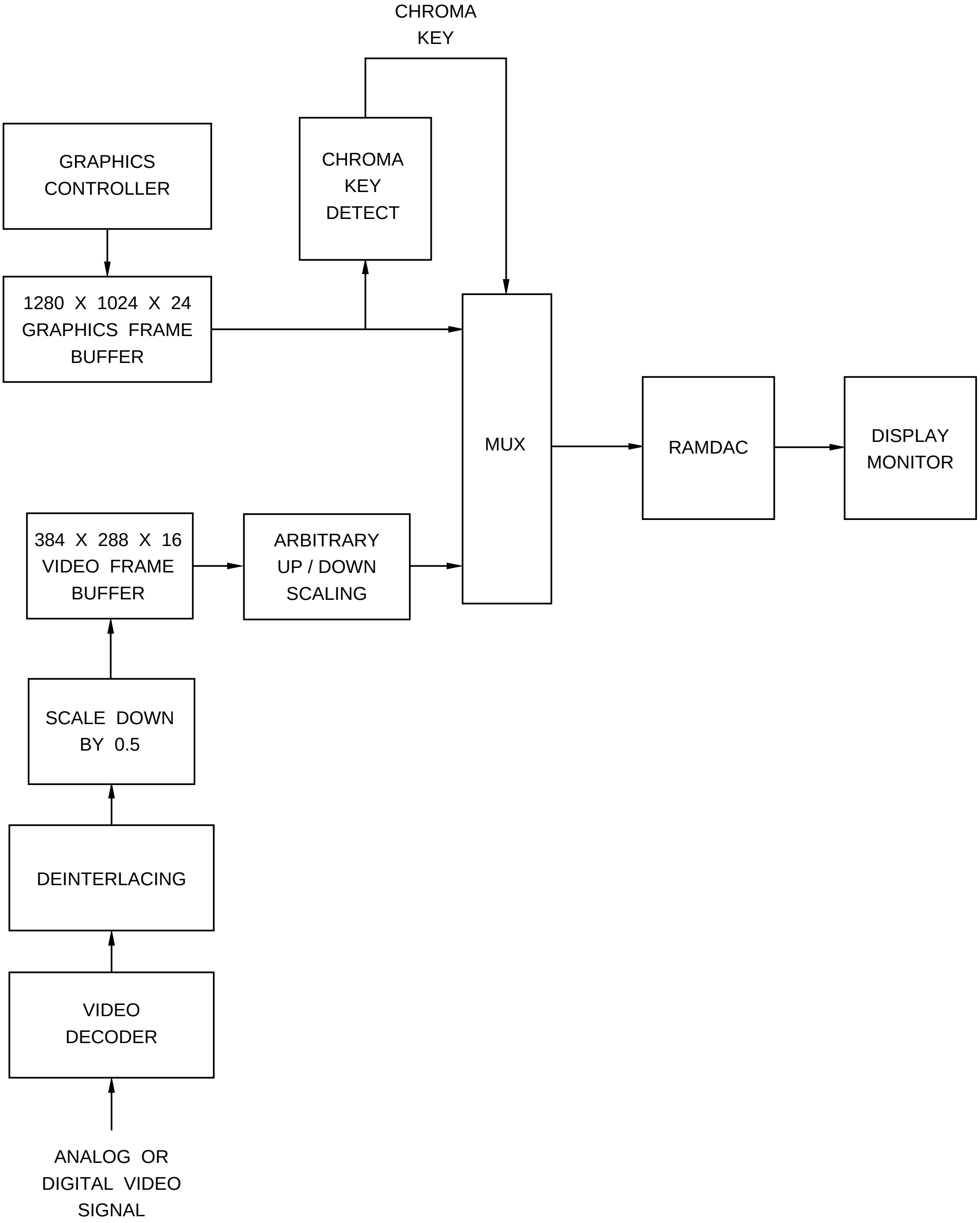


Figure 2.5. Example Architecture of Separate Graphics / Video Frame Buffers (Minimum Video Frame Buffer Size).

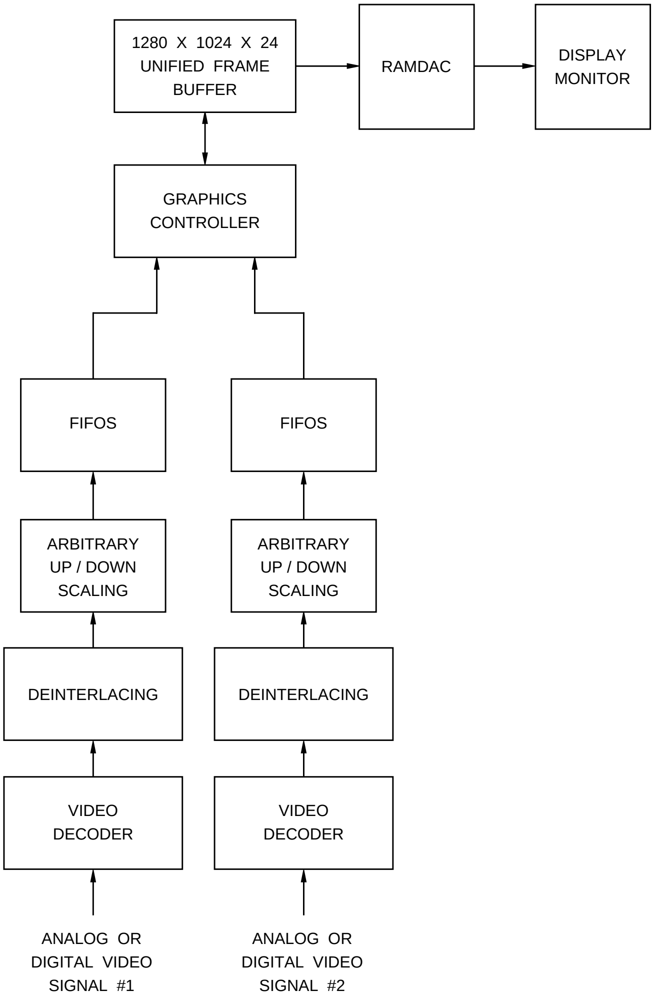


Figure 2.6. Example Architecture of a Unified Graphics / Video Frame Buffer.

mixing with the graphics and displayed. This architecture does limit the usefulness of the system for some applications, such as serious video editing, due to the limited video resolution available.

Single Unified Frame Buffer

This implementation (Figure 2.6) is also suitable to the standard computer environment, as both the video and graphics are rendered into a frame buffer. The video data is treated as any other data type that may be written into the frame buffer. Since video is usually true-color, a true-color graphics frame buffer is normally used. One advantage of this architecture is that the video is decoupled from the graphics/display subsystem. As a result, the display resolution may be changed, or multiple frame buffers and display monitors added, without affecting the video functionality. In addition, the amount of frame-buffer memory remains constant, regardless of the number of video sources supported. A disadvantage is that once the video is mixed with another source (video or graphics), the original image is no longer available.

In this system, the decoded video source is deinterlaced and scaled as discussed in the previous example. Since access to the frame buffer is not guaranteed, FIFOs are used to temporarily store the video data until access to the frame buffer is possible. Anywhere from 16 pixels to an entire scan line of pixels are loaded into the FIFOs. When access to the frame buffer is available, the video data is read out of the FIFO at the maximum data rate the frame buffer can accept it. Note that, once written into the frame buffer, video and graphics data may not be differentiated, unless additional data is used to "tag" the video data. Each video source must have its own method of adjusting the gamma, brightness, contrast, hue, and saturation, so as not to affect the graphics data.

Frame buffer access arbitration is a major concern. Real-time video requires a high-bandwidth interface into the frame buffer, as shown in Table 2.1, while other processes, such as the graphics controller, also require access. A standard deinterlaced video source may require up to 50–100 Mbytes per second bandwidth into the frame buffer, due to bus latency or read–modify–write operations being required. Additional bandwidth is required if the video source is scaled up; scaling the video source up to the full display size may require a bandwidth into the frame buffer of up to 300–400 Mbytes per second (assuming a 1280 × 1024 display resolution). A potential problem is that the video source being displayed may occasionally become "jerky" as a result of not being able to update the frame buffer with video information, due to another process accessing the frame buffer for an extended period of time. In this instance, it is important that any audio data accompanying the video not be interrupted.

If the data in the frame buffer is also to be converted to NTSC, PAL, or SECAM video signals for recording onto a video tape recorder, the video and graphics data should be alpha-mixed (discussed in Chapter 9) to eliminate hard switching at the graphics and video boundaries. To perform alpha mixing, data must be read from the frame buffer, mixed with the video data, and the result written back to the frame buffer. This technique can also be used to perform gradual fades and dissolves under software control.

Ideally, this system would be able to support multiple video streams by performing the "graceful degradation" discussed previously. For example, if the hardware can't support the number of video windows that the user wants, the windows could be made smaller or of reduced video quality to enable the display of the desired number of windows.

Virtual Frame Buffer

This implementation, shown in Figure 2.7, uses a "segmented" frame buffer memory. Each segment may be any size and may contain graphics, video, or any other data type. The graphics and video data are individually read out of the frame buffer and loaded into the RAMDAC, where they are mixed and displayed.

The decoded video source is deinterlaced and scaled as previously discussed. Since access to the memory is not guaranteed, FIFOs are again used to store the video data until access to the memory is possible. When memory access is available, the video data is read out of the FIFO at the maximum data rate possible. Note that, once written into the memory, there *is* still a way of differentiating video and graphics data, since each has its own memory allocation.

The major limitation to this approach is the bus bandwidth required to handle all of the

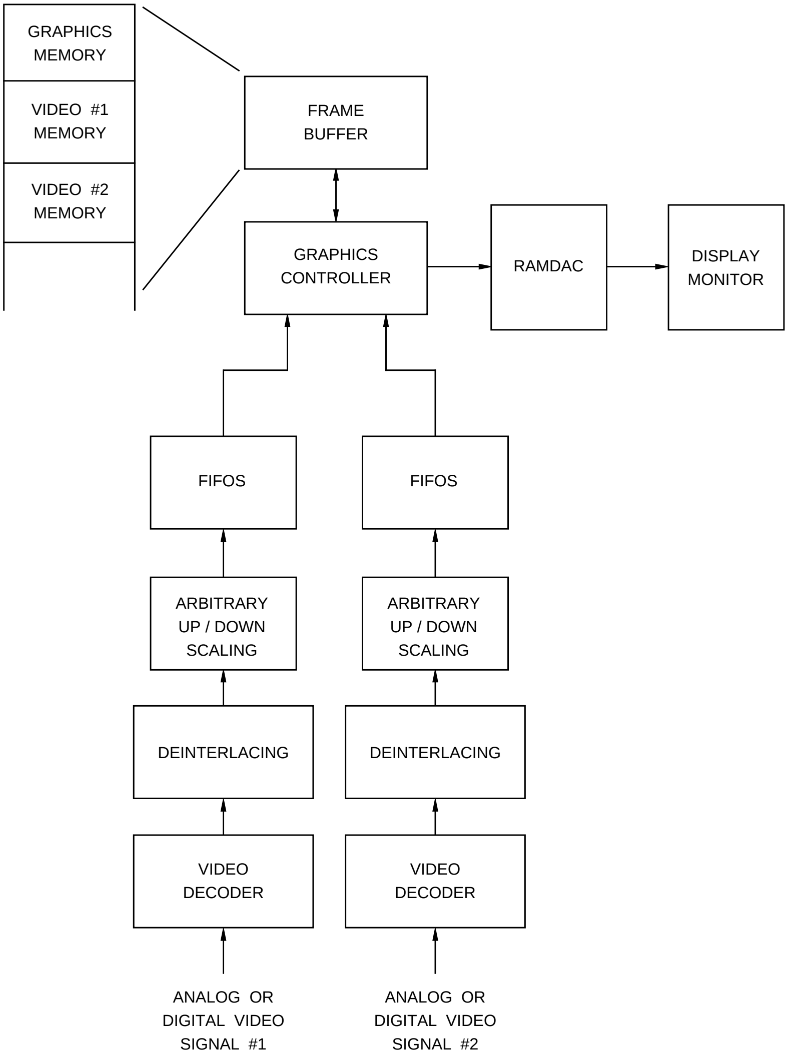


Figure 2.7. Example Architecture of a Virtual Graphics / Video Frame Buffer.

sources. For a 1280 × 1024 display, just the RAMDAC requires a bus bandwidth of up to 400 MBytes per second for the true-color graphics and 10–400 Mbytes per second for each video window (depending on the video window size). Each video input source also requires 10–400 Mbytes per second to store the video (again depending on the video window size). Don’t forget the bandwidth required for drawing graphics information into the memory!

Again, if the data in the memory is to be converted also to NTSC, PAL, or SECAM video signals for recording onto a video tape recorder, the video and graphics data should be alpha-mixed at the encoder to eliminate

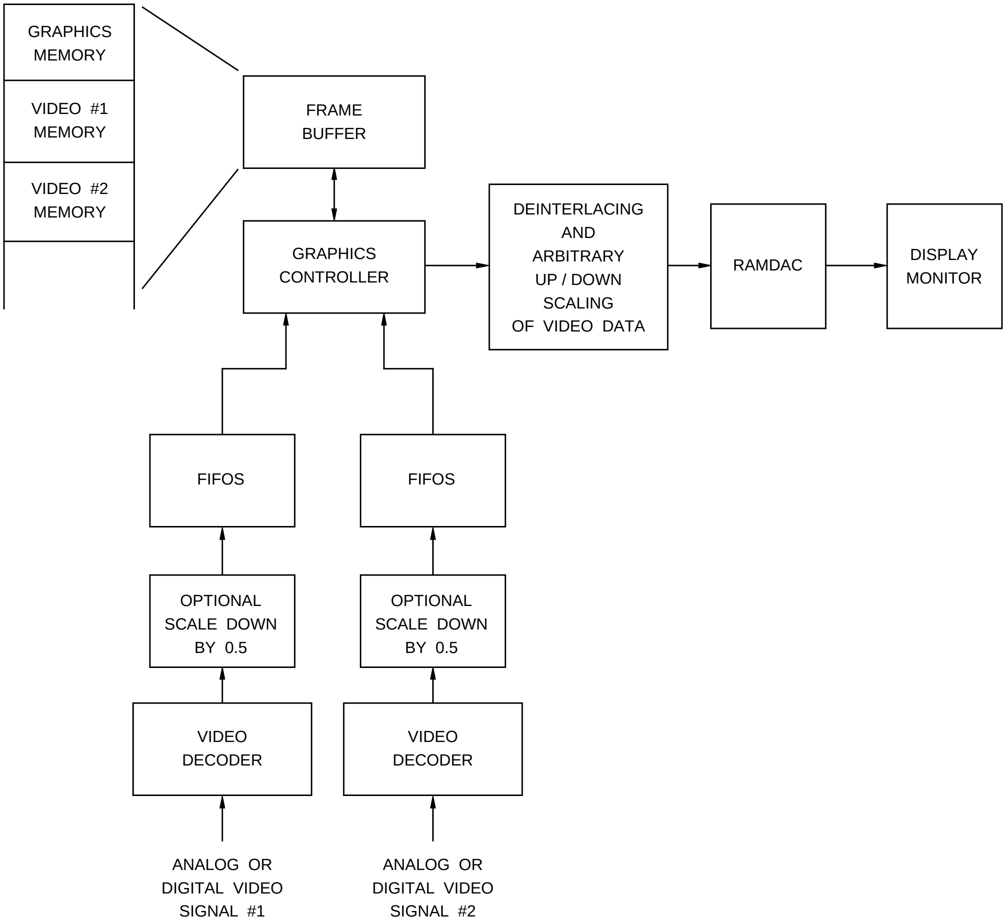


Figure 2.8. Alternate Example Architecture of a Virtual Graphics / Video Frame Buffer.

hard switching at the graphics and video boundaries.

A second variation, shown in Figure 2.8, is to store the video at a specific size (for example 640 × 480 or 320 × 240) prior to storing in the frame buffer. This limits the amount (and cost) of the frame buffer and eases bus bandwidth requirements. The video is then deinterlaced and scaled up or down to the desired resolution for mixing with the graphics and displayed.

Fast Pixel Bus

This implementation, shown in Figure 2.9, uses a high-bandwidth bus to transfer graphics and video data to the RAMDAC. The video sources are deinterlaced and scaled before being transferred to the RAMDAC and mixed with the graphics data.

The major limitation to this approach is the bus bandwidth required to handle all of the sources. For a 1280 × 1024 display, the RAM-

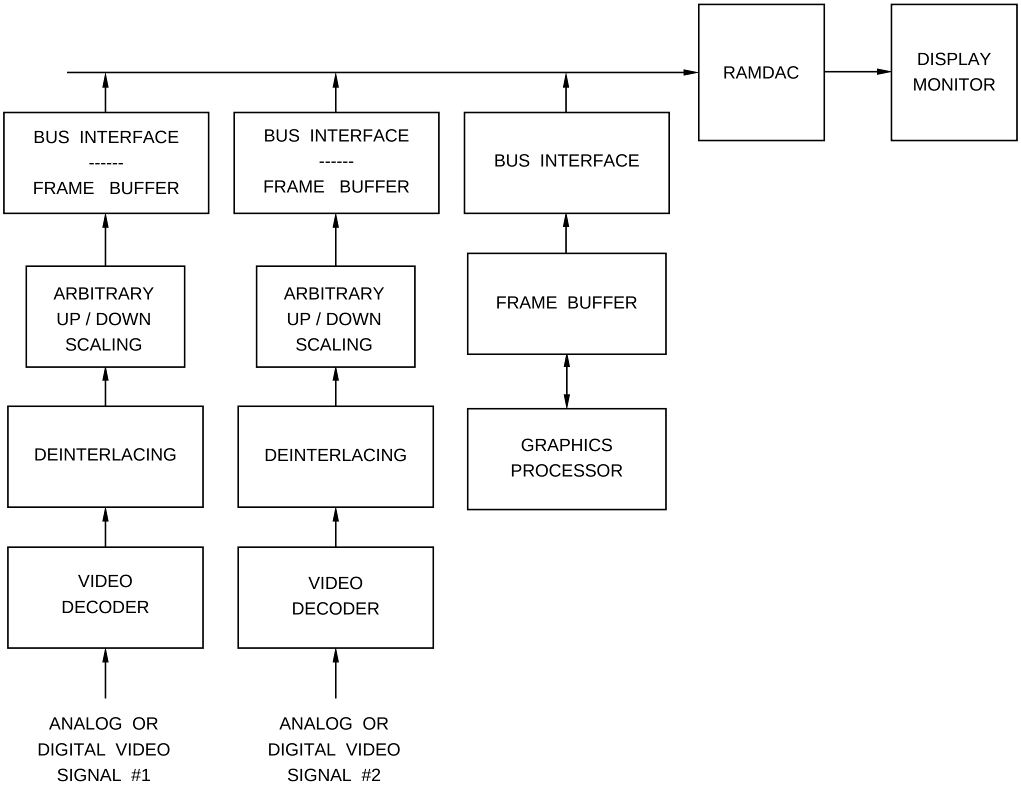


Figure 2.9. Example Architecture of a Graphics / Video System Using a Fast Pixel Bus.

DAC requires a bus bandwidth of up to 400 Mbytes per second for the true-color graphics and 10–400 Mbytes per second for each video window (depending on the video window size).

Alternately, doing the deinterlacing and scaling of the video within the RAMDAC allows additional processing. These processing functions may include determining which overlays (if any) are associated with the video and optionally disabling the cursor from being encoded into the NTSC, PAL, or SECAM video signal. A constant bandwidth for a video source is also maintained if the processing is done within the RAMDAC, rather than having the bandwidth be dependent on the scaling factors involved.

Generating Real-Time Video

Several possible implementations exist for generating "non-RGB" video in the computer environment, each with its own functionality, quality, and cost trade-offs. The video signals generated can be NTSC, PAL, or SECAM (for recording video onto a video tape recorder), a teleconferencing encoder (compressing teleconferencing video for transmission over a network), or a digital component video or digital composite video encoder (for recording digital video onto a digital tape recorder). The discussions in this chapter focus on generating analog NTSC / PAL / SECAM composite color video.

Figure 2.10 illustrates adding a video encoder to the video/graphics system shown in Figure 2.2. This example assumes that the graphics timing controller and display are operating in an interlaced mode, with a display resolution of 640 × 480, 59.94 fields per second (for generating NTSC video) or 768 × 576, 50 fields per second (for generating PAL or SECAM video). Since these refresh rates and resolutions are what the video encoder requires, no scaling, interlace conversion (converting from noninterlaced to interlaced), or frame rate conversion is required. Some newer computer display monitors may not support refresh rates this low and, even if they do, the flicker of white and highly saturated colors is worsened by the lack of movement in areas containing computer graphics information.

Figures 2.11 and 2.12 illustrate adding a video encoder to the video/graphics system shown in Figures 2.4 and 2.6, respectively. Display resolutions may range from 640 × 480 to 1280 × 1024 or higher. If the computer display resolution is higher than the video resolution, the video data may be scaled down to the proper resolution or only a portion of the entire display screen may be output to the video encoder. Scaling should be done on noninterlaced data if possible to minimize artifacts.

Since computer display resolutions of 640 × 480 or higher are usually noninterlaced, conversion to interlaced data after scaling typically must be performed. Some newer computer display monitors may not support the low refresh rates used by interlaced video, and if

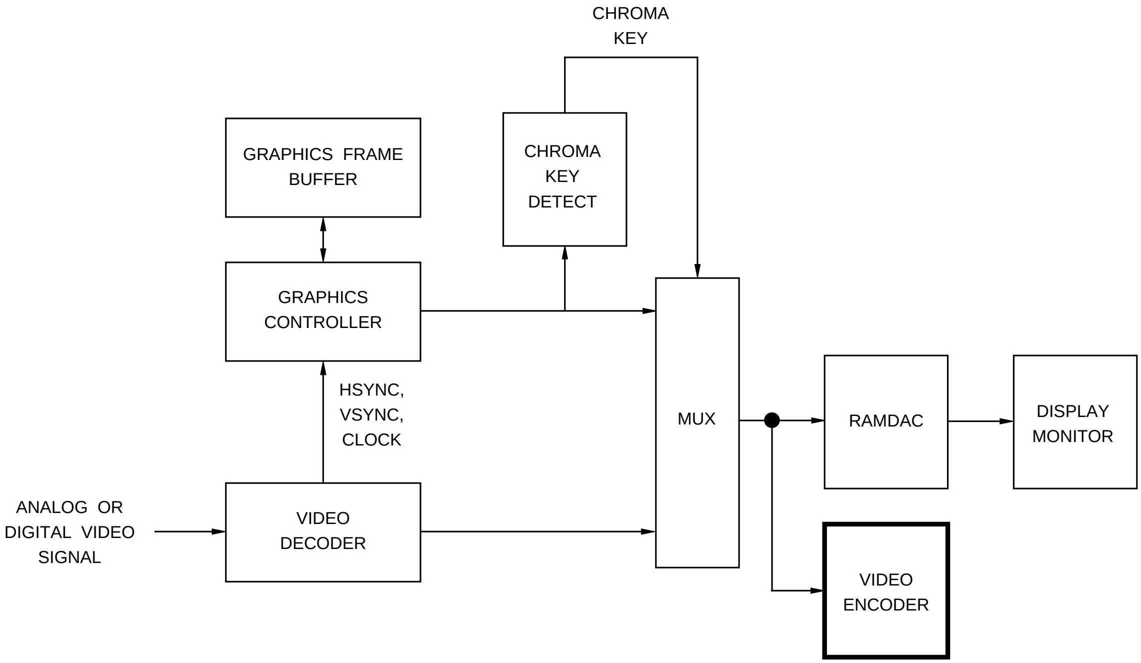


Figure 2.10. Adding a Video Encoder to the Video/Graphics System Shown in Figure 2.2.

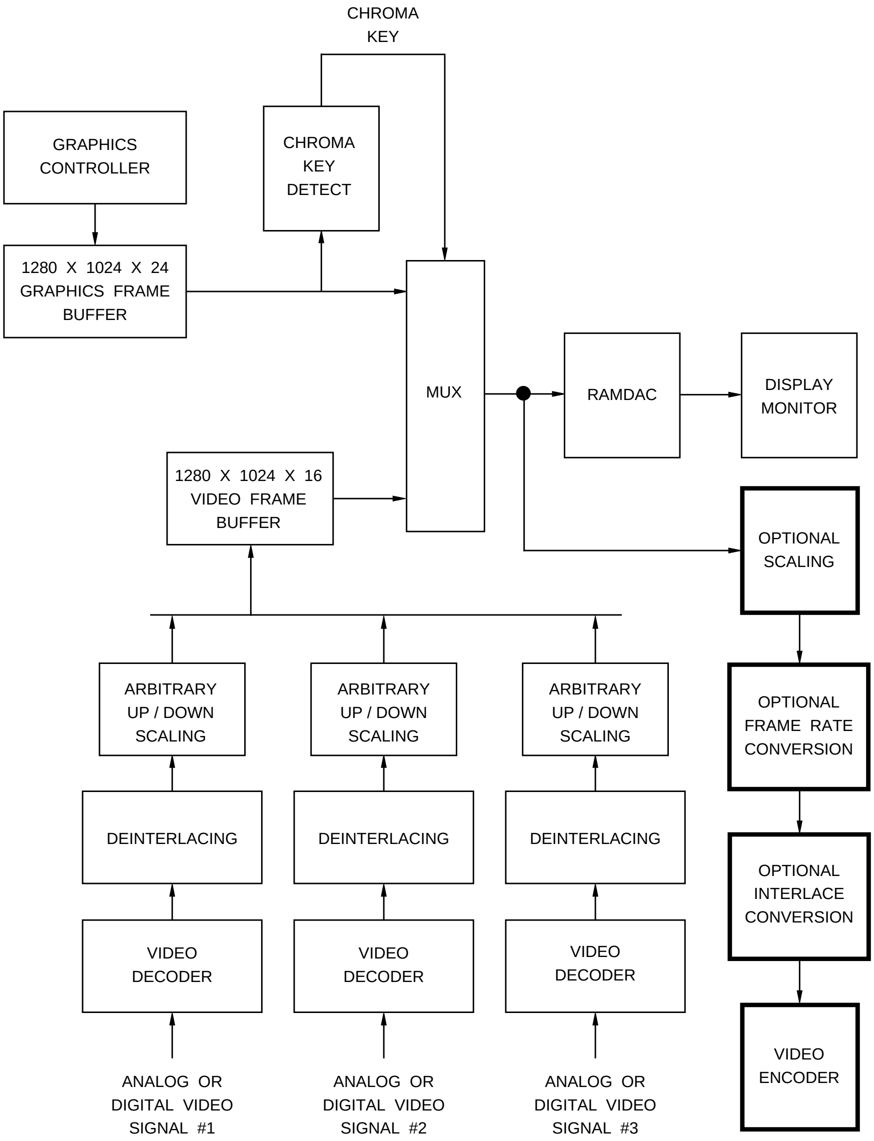


Figure 2.11. Adding a Video Encoder to the Video / Graphics System Shown in Figure 2.4.

they do, the flicker of white and highly saturated colors is worsened by the lack of movement in areas containing computer graphics information.

Computer display refresh rates higher than 50 or 60 Hz (i.e, 72 Hz) also require some type of frame-rate conversion to 50 or 60 Hz interlaced. Frame-rate conversion may be eliminated by using shadow video and graphics frame buffers. Containing the same information as the primary video and graphics frame buffers, these duplicate buffers employ a separate timing controller that allows data to be output at 59.94 Hz (NTSC)

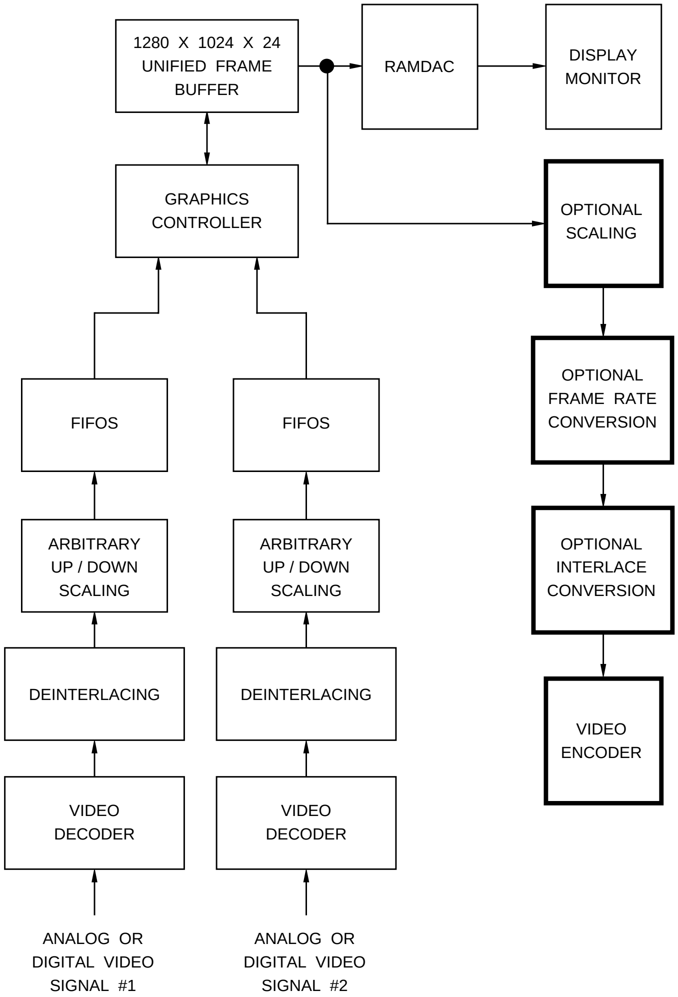


Figure 2.12. Adding a Video Encoder to the Video/Graphics System Shown in Figure 2.6.

or 50 Hz (PAL and SECAM), either interlaced or noninterlaced. Interlaced operation would eliminate the noninterlaced to interlaced conversion, but consideration of the graphics information still is required (for example, no one-pixel-wide lines or small fonts) for acceptable video quality. Alternately, using triple-port frame buffers (allowing support of two independent serial output streams) also will eliminate frame-rate conversion. One serial output port drives the RAMDAC at the display refresh rate, and the other serial output port drives the optional scaling and video encoder circuitry at the appropriate NTSC or PAL/SECAM refresh rates.

A problem arises if the RAMDAC incorporates various types of color processing, such as supporting multiple pixel formats on a pixel-by-pixel basis. For example, the RAMDAC may have the ability to input 8-bit pseudo-color, 16-bit YCbCr, or 24-bit RGB data on a pixel-by-pixel basis or provide overlay support for cursors and menus. Either the circuitry performing the color processing inside the RAMDAC must be duplicated somewhere in the video encoder path or the RAMDAC must output a digital RGB version of its analog outputs to drive the video encoder path.

Color Space Issues

Most computers use the RGB color space (either 15-, 16-, or 24-bit RGB) or 8-bit pseudocolor. Although newer video encoders and decoders support both RGB and YCbCr as color space options, the video processing required (such as interlace conversion and scaling) is usually much more efficient using the YCbCr color space. As some video quality is lost with each color space conversion (for example, converting from RGB to YCbCr), care should be taken to minimize the number of color space conversions used in the video encoding and decoding paths.

For example, along the video encoding path, the RGB data from the frame buffer should be converted to YCbCr before any scaling and interlace conversion operations (the input to the scaler should support RGB-to-YCbCr conversion). The input to the video encoder then should be configured to be the YCbCr color space.

Color spaces are discussed in detail in the next chapter. To facilitate the transfer of video data between VLSI devices that support multiple color spaces, de facto standards have been developed, as shown in Tables 2.2 and 2.3.

Square Pixel Issues

Most modern displays for computers use a 1:1 aspect ratio (square pixels). Most consumer video displays currently use rectangular pixels. Without taking these differences into account, computer-generated circles will become ellipses when processed by a video encoder. Similarly, circles as displayed on a television will appear as ellipses when displayed on the computer.

To compensate for the pixel differences, NTSC/PAL/SECAM video encoders and decoders are able to operate at special pixel clock rates and horizontal resolutions. Square-pixel NTSC video encoders and decoders operate at 12.2727 MHz (640 active pixels per line), whereas square-pixel PAL and SECAM video encoders and decoders operate at 14.75 MHz (768 active pixels per line). NTSC/PAL/SECAM encoders and decoders are also available that operate at 13.5 MHz (720 active pixels per line), supporting the conventional rectangular video aspect ratio.

When processing a video stream, it is important to know its format. For example, if a NTSC decoder supports square pixels, and a MPEG encoder expects rectangular pixels, scaling between the NTSC decoder and the MPEG encoder must take place. When the video is stored in memory for processing, if the format ratio is not what is expected, major processing errors will occur.

Audio Issues

Just as there are many issues involved in getting video into and out of a computer environment, there are just as many issues regarding the audio. As seen in Figure 2.1, some of the possible audio sources are analog audio from a microphone or video tape recorder or digital audio from a digital video decoder, MPEG decoder, CD player, or digitized audio file on the disk or a CD-ROM.

Just as there are many video standards, there are even more audio standards. The audio may be sampled at one of several rates (8 kHz, 11.025 kHz, 22.05 kHz, 32 kHz, 44.1 kHz,

24-bit RGB (8, 8, 8)	16-bit RGB (5, 6, 5)	15-bit RGB (5, 5, 5)	24-bit 4:4:4 YCbCr	16-bit 4:2:2 YCbCr	12-bit 4:1:1 YCbCr
R7	–	–	Cr7	–	–
R6	–	–	Cr6	–	–
R5	–	–	Cr5	–	–
R4	–	–	Cr4	–	–
R3	–	–	Cr3	–	–
R2	–	–	Cr2	–	–
R1	–	–	Cr1	–	–
R0	–	–	Cr0	–	–
G7	R7	–	Y7	Y7	Y7
G6	R6	R7	Y6	Y6	Y6
G5	R5	R6	Y5	Y5	Y5
G4	R4	R5	Y4	Y4	Y4
G3	R3	R4	Y3	Y3	Y3
G2	G7	R3	Y2	Y2	Y2
G1	G6	G7	Y1	Y1	Y1
G0	G5	G6	Y0	Y0	Y0
B7	G4	G5	Cb7	[Cb7], Cr7	[Cb7], Cb5, Cb3, Cb1
B6	G3	G4	Cb6	[Cb6], Cr6	[Cb6], Cb4, Cb2, Cb0
B5	G2	G3	Cb5	[Cb5], Cr5	[Cr7], Cr5, Cr3, Cr1
B4	B7	B7	Cb4	[Cb4], Cr4	[Cr6], Cr4, Cr2, Cr0
B3	B6	B6	Cb3	[Cb3], Cr3	–
B2	B5	B5	Cb2	[Cb2], Cr2	–
B1	B4	B4	Cb1	[Cb1], Cr1	–
B0	B3	B3	Cb0	[Cb0], Cr0	–

Timing control signals:

graphics:
- horizontal sync (HSYNC*)
- vertical sync (VSYNC*)
- composite blank (BLANK*)

ITU-R BT.601:
- horizontal blanking (H)
- vertical blanking (V)
- even/odd field (F)

Philips:
- horizontal sync (HS)
- vertical sync (VS)
- horizontal blank (HREF)

Table 2.2. Pixel Format Standards for Transferring RGB and YCbCr Video Data Over a 16-bit or 24-bit Bus. For 4:2:2 and 4:1:1 YCbCr data, the first active pixel data per scan line is indicated in brackets [ ].

Pixel Bus	24-bit RGB (8, 8, 8)	16-bit RGB (5, 6, 5)	15-bit RGB (5, 5, 5)	24-bit 4:4:4 YCbCr	16-bit 4:2:2 YCbCr	12-bit 4:1:1 YCbCr
P7D	–	R7	–	–	Y7	Y7
P6D	–	R6	R7	–	Y6	Y6
P5D	–	R5	R6	–	Y5	Y5
P4D	–	R4	R5	–	Y4	Y4
P3D	–	R3	R4	–	Y3	Y3
P2D	–	G7	R3	–	Y2	Y2
P1D	–	G6	G7	–	Y1	Y1
P0D	–	G5	G6	–	Y0	Y0
P7C	R7	G4	G5	Cr7	Cb7, Cr7	Cb5, Cb1
P6C	R6	G3	G4	Cr6	Cb6, Cr6	Cb4, Cb0
P5C	R5	G2	G3	Cr5	Cb5, Cr5	Cr5, Cr1
P4C	R4	B7	B7	Cr4	Cb4, Cr4	Cr4, Cr0
P3C	R3	B6	B6	Cr3	Cb3, Cr3	–
P2C	R2	B5	B5	Cr2	Cb2, Cr2	–
P1C	R1	B4	B4	Cr1	Cb1, Cr1	–
P0C	R0	B3	B3	Cr0	Cb0, Cr0	–
P7B	G7	R7	–	Y7	Y7	Y7
P6B	G6	R6	R7	Y6	Y6	Y6
P5B	G5	R5	R6	Y5	Y5	Y5
P4B	G4	R4	R5	Y4	Y4	Y4
P3B	G3	R3	R4	Y3	Y3	Y3
P2B	G2	G7	R3	Y2	Y2	Y2
P1B	G1	G6	G7	Y1	Y1	Y1
P0B	G0	G5	G6	Y0	Y0	Y0
P7A	B7	G4	G5	Cb7	[Cb7], Cr7	[Cb7], Cb3
P6A	B6	G3	G4	Cb6	[Cb6], Cr6	[Cb6], Cb2
P5A	B5	G2	G3	Cb5	[Cb5], Cr5	[Cr7], Cr3
P4A	B4	B7	B7	Cb4	[Cb4], Cr4	[Cr6], Cr2
P3A	B3	B6	B6	Cb3	[Cb3], Cr3	–
P2A	B2	B5	B5	Cb2	[Cb2], Cr2	–
P1A	B1	B4	B4	Cb1	[Cb1], Cr1	–
P0A	B0	B3	B3	Cb0	[Cb0], Cr0	–

Table 2.3. Pixel Format Standards for Transferring RGB and YCbCr Video Data Over a 32-bit Bus. For 4:2:2 and 4:1:1 YCbCr data, the first active pixel data per scan line is indicated in brackets [ ]. For all formats except 24-bit RGB and 24-bit YCbCr data, PxA and PxB data contain pixel n data, PxC and PxD contain pixel n + 1 data. Refer to Table 2.2 for timing control signals.

or 48 kHz), be 8-bit, 12-bit, or 16-bit samples, mono or stereo, linear PCM (pulse code modulation) or ADPCM (adaptive differential pulse code modulation). Problems arise when mixing two or more digital audio signals that use different sampling rates or formats. Both audio streams must be converted to a common sample rate and format before digital mixing is done. With the CD, DAT, DCC, MD, and laserdisk machines now supporting digital audio as an option, having the ability to input and output digital audio is desirable. Although the bandwidths required for audio are much smaller than those for video (Table 2.4 illustrates the audio bandwidth requirements for various sample rates and resolutions), they must be taken into account along with the video bandwidths when determining the maximum system bandwidth required.

When generating audio for recording with computer-generated video, system-dependent sounds should not be recorded with the video. Two solutions are to turn off the system audio during recording or to generate separate system audio signals that drive the internal speakers within the computer. The serial digital audio interface should support both the coaxial and optical formats, as both are standard in the consumer industry. Audio sample rates are 44.1 kHz for CD; DAT sample rates are 32 kHz, 44.1 kHz, or 48 kHz. A generic serial audio interface enabling interfacing to MIDI sound

Sample Rate	Mono or Stereo	Resolution	Bandwidth (kbytes per second per audio channel)
8.0 kHz	mono	8-bit μ-law PCM	8.0
	mono	8-bit A-law PCM	8.0
	mono	4-bit ADPCM	4.0
11.025 kHz	m/s	8-bit linear PCM	11.1
	m/s	4-bit ADPCM	5.6
22.05 kHz	m/s	8-bit linear PCM	22.1
	m/s	4-bit ADPCM	11.1
44.1 kHz	m/s	16-bit linear PCM	88.2
	m/s	4-bit ADPCM	22.1
48.0 kHz	m/s	16-bit linear PCM	96.0
	m/s	4-bit ADPCM	24.0

Table 2.4. Bandwidths of Common Audio Signal Formats.

synthesizers and DSP processors should be supported.

What is a RAMDAC?

With all this talk about RAMDACs, perhaps we should review the basic function of a RAMDAC for those not familiar with its function.

Each pixel on a color display monitor is composed of three phosphor "dots," one red, one green, and one blue. Using three primary colors, such as RGB, allows almost any color to be represented on the display. If a pixel is supposed to be red, the red phosphor is excited, whereas the green and blue ones are not. The amount of excitation, from 0% to 100%, determines the brightness of the phosphor.

Today, the RAMDAC function is usually incorporated within the graphics controller.

True-Color RAMDAC

Figure 2.13 illustrates a simplified block diagram of a true-color RAMDAC. True-color means the red, green, and blue phosphors of each pixel on the display are individually represented by their own red, green, and blue data in the frame buffer (although another color space may be used, RGB is the most common for computer graphics). For each pixel to be displayed, 24 bits of data (8 bits each of red, green, and blue) are transferred in real time from the frame buffer memory to the RAMDAC. Thus, for a 1280 × 1024 noninterlaced display, the 24 bits of data must be transferred every 7.4 ns.

The red, green, and blue pixel data address three "lookup table" RAMs, one each for the red, green, and blue data. The primary purpose of the lookup table RAMs in this

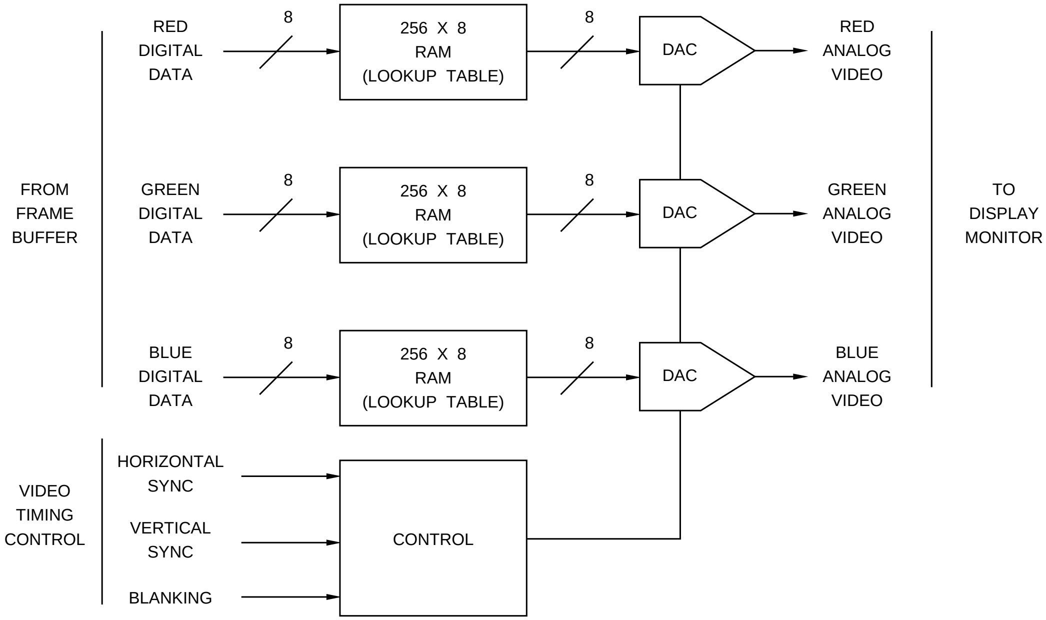


Figure 2.13. Simplified Block Diagram of a True-Color RAMDAC.

instance is to provide gamma correction (discussed in Chapter 3) for the display monitor. The lookup table RAMs are loaded by the CPU with the desired data.

The output of the lookup table RAMs drive three digital-to-analog converters (DACs) to convert the digital data to red, green, and blue analog signals used to drive the display monitor. Video timing information, such as horizontal sync, vertical sync, and blanking are optionally added to the analog signals, or sent to the monitor separately.

Pseudo-Color RAMDAC

Figure 2.14 illustrates a simplified block diagram of a pseudo-color RAMDAC. Pseudo-color means the red, green, and blue phosphors of each pixel on the display are *not* individually represented by their own red, green, and blue data in the frame buffer. Instead, the frame buffer contains an 8-bit "index" value for each pixel. The primary benefit of this implementation is the much lower cost of frame buffer memory (one-third that of a true-color system).

This 8-bit index is used to address a lookup table that "looks up" what red, green, and blue data is to be generated for a given index value. The only correlation between the 8-bit index value and the color is whatever the CPU assigns. In other words, the generated colors are artificial. For each pixel to be displayed, only 8 bits of data must be transferred in real time from the frame buffer memory to the RAMDAC. Thus, for a 1280 × 1024 noninterlaced display, 8 bits of data are transferred every 7.4 ns.

The 8 bits of pixel data simultaneously address three "lookup table" RAMs, one each for the red, green, and blue data. The primary purpose of the lookup table RAMs in this instance is to generate (look up) the red, green, and blue data and to provide gamma correction for the display (discussed in Chapter 3). The lookup table RAMs are loaded by the CPU with the appropriate data.

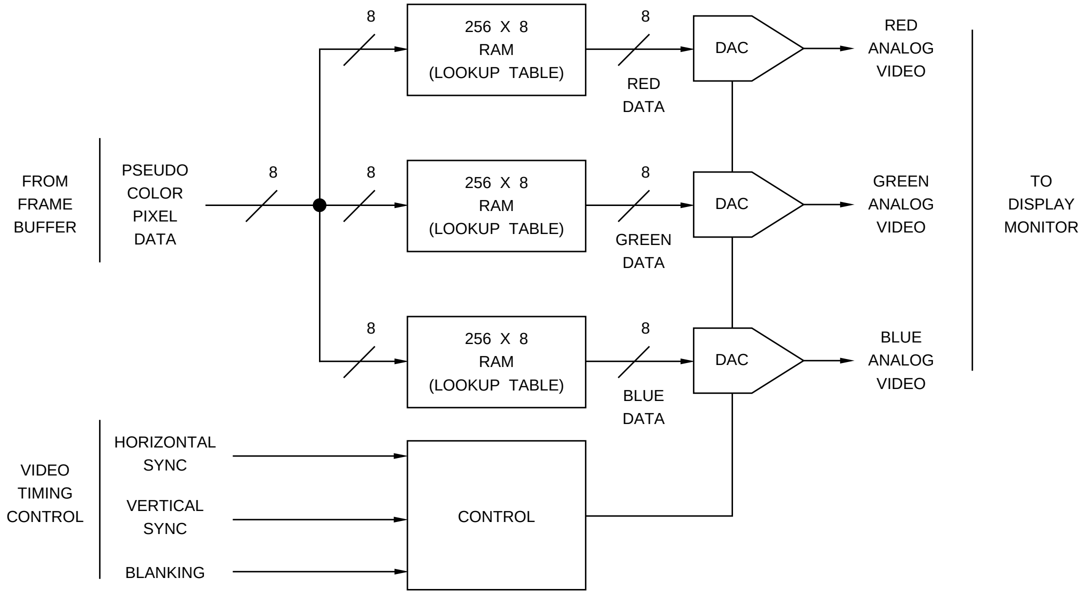


Figure 2.14. Simplified Block Diagram of a Pseudo-Color RAMDAC.

Again, the output of the lookup table RAMs drive three digital-to-analog converters (DACs) to convert the digital data to red, green, and blue analog signals used to drive the display monitor. Video timing information, such as horizontal sync, vertical sync, and blanking are optionally added to the analog signals, or sent to the monitor separately.

Hardware Assist of Software

To reduce the cost of adding multimedia to price-sensitive personal computers, as many functions as possible are implemented in software. However, some video processing functions now are handled in hardware to assist the video decompression software.

Implementing color space conversion (typically from YCbCr to RGB) and video scaling in hardware enables larger video windows, or a higher video refresh rate, to be displayed at minimum additional cost. Typically, these hardware functions are added to graphics controllers that also support graphics drawing acceleration.

In the future, even more hardware probably will be embedded in existing devices, such as graphics controllers and processors, to accelerate software video decompression.

Native Signal Processing

Intel has developed Native Signal Processing (NSP) to facilitate the incorporation of multimedia functions into a standard PC.

NSP promotes “balanced partitioning” of CPU resources, processing requirements, and system cost. Complex signal processing functions are implemented in software running on the Pentium® processor. Simple functions are implemented in silicon. The partitioning approach results in lower-cost components and add-in cards.

Some of the requirements for NSP hardware support are direct interfacing to the PCI or Pentium® bus, direct memory access control, direct access to other PCI functions, and handling the latency, bandwidth, and buffering requirements.

Common PC Video Architectures

To ensure compatibility between systems and provide a common platform, several standards have been defined for adding video to personal computers.

VESA Advanced Feature Connector

The VESA Advanced Feature Connector (VAFC) allows a graphics controller on the motherboard to have access to the digital video data from an add-in video card (Figure 2.15a). It also allows an add-in video board to have access to the digital graphics data from a graphics controller on the motherboard (Figure 2.15b). In this case, the video output of the motherboard is not used to drive the monitor. Either scheme allows mixing of the graphics and video data digitally, allowing many capabilities to be easily added.

The advantage of the VAFC is that add-in cards may be used, at the user’s convenience, to add multimedia capabilities. This keeps the cost of the basic personal computer to a minimum, while still allowing users to add multimedia capability at any time.

The VAFC has a 32-bit bidirectional data bus and several timing control signals, as shown in Tables 2.5 and 2.6. Data may be transferred at a 37.5 MHz maximum clock rate, resulting in a 150 Mbyte/sec maximum bandwidth. The supported data formats are:

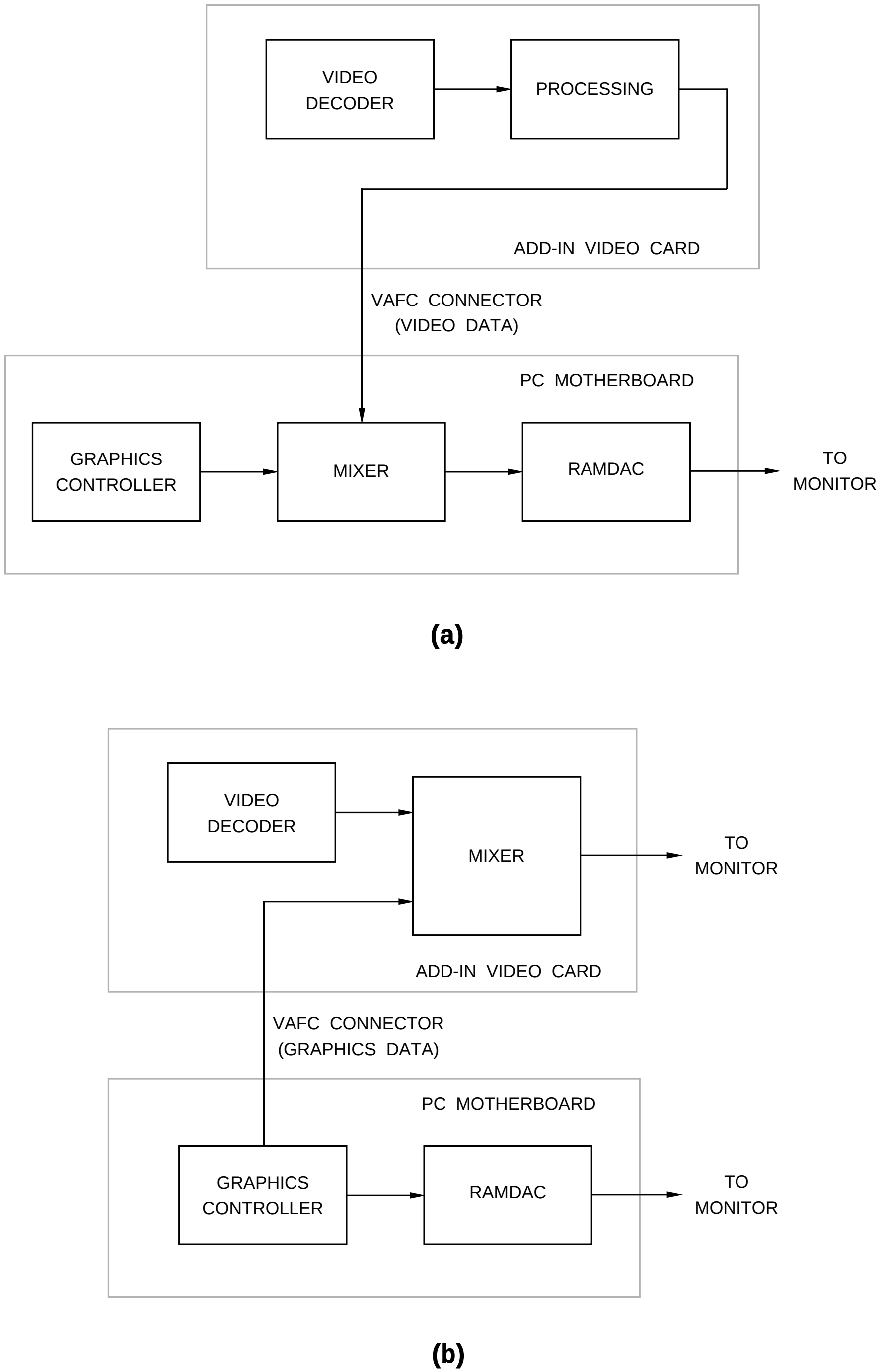


Figure 2.15. VESA Advanced Feature Connector Applications.

Name	Video Mode	I/O	Sync	Description
P0–P31	in	I	VCLK	Data from video system
	out	O	DCLK	Data from graphics system
DCLK	in/out	O		Graphics pixel clock
VCLK	in/out	I		Video pixel clock
BLANK*	in/out	O	DCLK	Graphics blanking
HSYNC	in/out	O	DCLK	Graphics horizontal sync
VSYNC	in/out	O	DCLK	Graphics vertical sync
EVIDEO*	in	I		Low = Video data present
	out	I		High = Graphics data present
EGEN*	in/out	I		Enable GENCLK
GRDY	in	O	VCLK	Graphics ready to latch data
VRDY	in	I	VCLK	Valid video data
FSTAT	in	O	VCLK	FIFO status
OFFSET (0, 1)	in	I	VCLK	Pixel offset
GENCLK	in/out	I		Genlock clock
RSRV (0–2)	tbd	tbd	tbd	

Video Mode: “In” means data coming **into** the graphics system from the video system.
“Out” means data is going **out** of the graphics system into the video system.

I/O: Pin directions are defined as seen at the graphics system.

Sync: Indicates if the pin is synchronous to DCLK, VCLK, or neither.

Table 2.5. VAFC Signals.

Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	RSRV0	21	GND	41	GND	61	P6
2	RSRV1	22	P7	42	GND	62	GND
3	GENCLK	23	P8	43	GND	63	P9
4	OFFSET0	24	GND	44	GND	64	P10
5	OFFSET1	25	P11	45	GND	65	GND
6	FSTAT	26	P12	46	GND	66	P13
7	VRDY	27	GND	47	GND	67	P14
8	GRDY	28	P15	48	GND	68	GND
9	BLANK*	29	P16	49	GND	69	P17
10	VSYNC	30	GND	50	GND	70	P18
11	HSYNC	31	P19	51	GND	71	GND
12	EGEN*	32	P20	52	GND	72	P21
13	VCLK	33	GND	53	GND	73	P22
14	RSRV2	34	P23	54	GND	74	GND
15	DCLK	35	P24	55	GND	75	P25
16	EVIDEO*	36	GND	56	GND	76	P26
17	P0	37	P27	57	P1	77	GND
18	GND	38	P28	58	P2	78	P29
19	P3	39	GND	59	GND	79	P30
20	P4	40	P31	60	P5	80	GND

Table 2.6. VAFC Connector Pin Assignments.

RGB data:

32-bit including alpha (8, 8, 8, 8)
24-bit (8, 8, 8)
16-bit (5, 6, 5)
16-bit including alpha (1, 5, 5, 5)
15-bit (5, 5, 5)
8-bit (3, 3, 2)

YCbCr data:

24-bit 4:4:4
16-bit 4:2:2
12-bit 4:1:1

Pseudo-color data:

8-bit

VESA Media Channel

The VESA Media Channel (VMChannel) is a multiple-master, multiple-drop, clock-synchronous interface designed for video streams. It enables the bidirectional, real-time flow of uncompressed video between devices, as shown in Figure 2.16, regardless of the resolution or refresh rate. As with the VAFC, VMChannel allows add-in cards to be added to a personal computer at the user's convenience to add multimedia capabilities.

VMChannel requires a single "bus controller," which is anticipated to be within the graphics controller. Otherwise, it is a peer-to-peer multi-master bus. Video sources are granted ownership of the bus via various token-passing schemes. "Broadcasting" is also supported, allowing video data to be sent to any number of devices at the same time.

The VMChannel has a 32-bit bidirectional data bus and several control signals, as shown in Tables 2.7 and 2.8. Data may be transferred at a 33-MHz maximum clock rate, resulting in a 132-Mbyte/sec maximum bandwidth. The supported data formats are (* represents baseline requirements):

RGB data:

32-bit including alpha (8, 8, 8, 8)*
24-bit (8, 8, 8)
16-bit (5, 6, 5)*
8-bit (3, 3, 2)*
8-bit gray-scale*

YCbCr data:

4:2:2
4:2:0
4:1:1

Pseudo-color data:

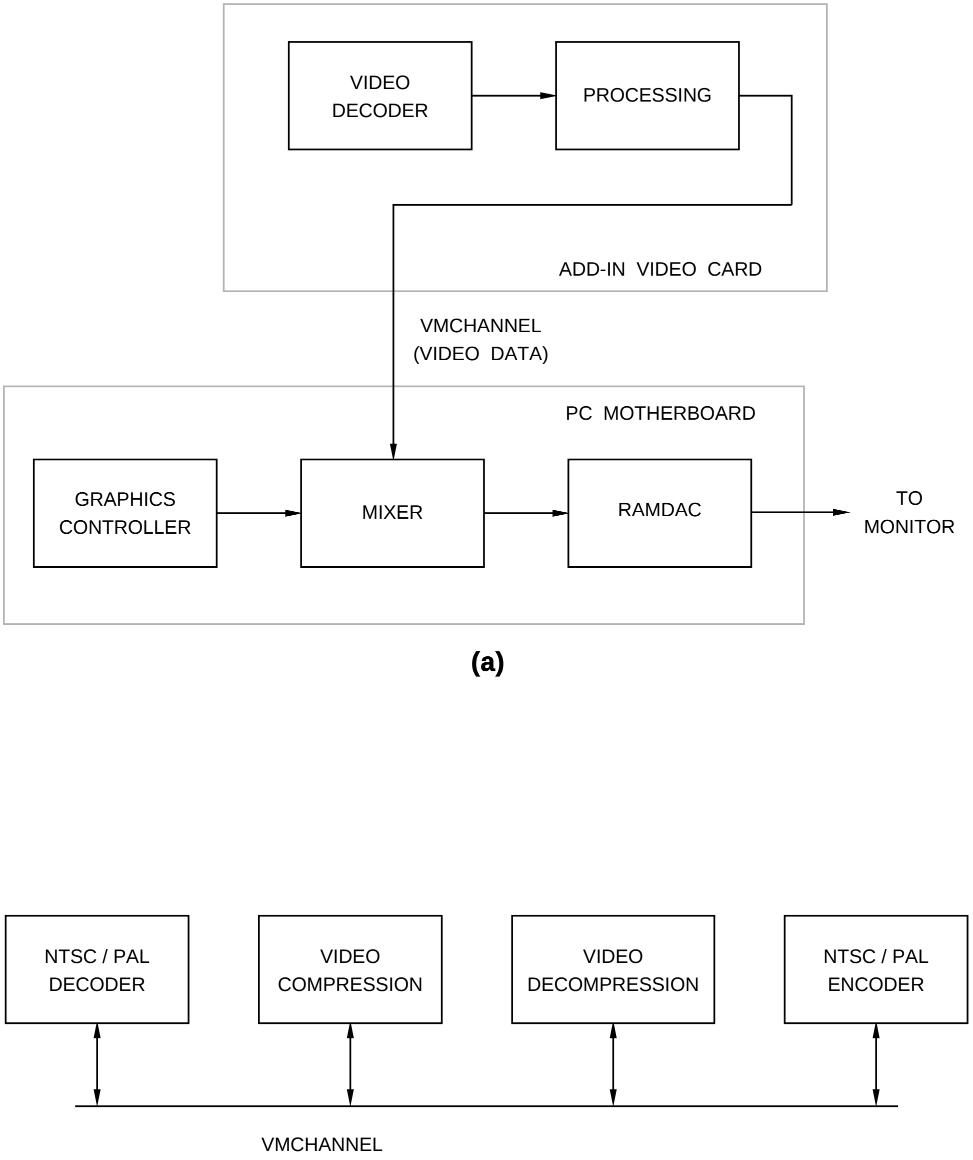
8-bit

The SA* and SB* signals, used during the configuration phase, are also used by devices to request an interrupt. This allows devices currently excluded from the token loop to get attention and reinsert themselves into the token loop.

The BS0* and BS1* signals allow dynamic bus resizing of 8, 16, or 32 bits.

PCI Bus

Many companies are pursuing using the 32-bit high bandwidth PCI bus (up to 132-MB/second bandwidth, usually implemented as a secondary PCI bus) for transferring real-time digital audio and video data between devices. This also eliminates the need for both dedicated audio/video and host processor inter-



(a)

(b)

Figure 2.16. VMChannel Applications.

faces on devices. However, there is some debate as to whether the protocol overhead of the 32-bit PCI bus limits the ability to transfer real-time full-resolution video.

Work is progressing on implementing the 64-bit version (up to 264 MB/second bandwidth), along with reducing the protocol overhead, and increasing the clock to 66 MHz.

Name	I/O	Description
DATA (0–31)	in/out	
CLK	in	Bus clock provided by bus controller
CONTROL	in/out	Indicates a control rather than data transfer
BS0*, BS1*	in/out	Bus size
SNRDY*	in/out	Slave not ready
SA*, SB*	in/out	Serial in/out
RESET*	in	Reset signal
MASK (0, 1)	in/out	Pixel mask
EVST (0, 1)	out	Event status

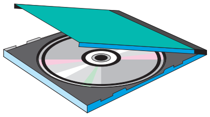
I/O: Pin directions are defined for “non-bus controller” devices.

Table 2.7. VMChannel Signals.

Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	SA*	18	DATA10	35	EVST0	52	DATA11
2	EVST1	19	GND	36	GND	53	DATA12
3	BS0*	20	DATA13	37	BS1*	54	GND
4	GND	21	DATA14	38	SNRDY*	55	DATA15
5	CONTROL	22	GND	39	GND	56	DATA16
6	RESET*	23	DATA17	40	GND	57	GND
7	CLK	24	DATA18	41	GND	58	DATA19
8		25	GND	42	GND	59	DATA20
9	MASK0	26	DATA21	43	MASK1	60	GND
10	GND	27	DATA22	44	DATA0	61	DATA23
11	DATA1	28	GND	45	GND	62	DATA24
12	DATA2	29	DATA25	46	DATA3	63	GND
13	GND	30	DATA26	47	DATA4	64	DATA27
14	DATA5	31	GND	48	GND	65	DATA28
15	DATA6	32	DATA29	49	DATA7	66	GND
16	GND	33	DATA30	50	DATA8	67	DATA31
17	DATA9	34	GND	51	GND	68	SB*

Table 2.8. VMChannel Connector Pin Assignments.

VLSI Solutions



See C2_VLSI

References

1. Alliance Semiconductor Corporation, ProMotion-3210™ Databook, May 1994.
2. AuraVision Corporation, VxP201 Product Brief, March 1994.
3. AuraVision Corporation, VxP202 Product Brief, August 1994.
4. AuraVision Corporation, VxP500 Product Brief, March 1994.
4. AuraVision Corporation, VxP501 Product Brief, October 1994.
6. Brooktree Corporation, Bt819 Datasheet, July 6, 1994.
7. Brooktree Corporation, Bt885 Datasheet, L885001 Rev. D.
8. Brooktree Corporation, Bt895 Datasheet, L895001 Rev. D.
9. Brooktree Corporation, BtV MediaStream Family Product Guide and Overview, BtVPG001, October 1994.
10. Chips and Technologies 69003/69004 Datasheet, January 1992.
11. Chips and Technologies 82C9001 Datasheet, April 1992.
12. Cirrus Logic, CL-PX2070 Datasheet, July 1993.
13. Cirrus Logic, CL-PX2080 Datasheet, September 1993.
14. Cirrus Logic, CL-PX4070 Datasheet, April 1994.
15. Cirrus Logic, CL-PX4080 Datasheet, April 1994.
16. S3, Trio64V+ Product Overview, June 1995.
17. S3, Scenic/MX1 Product Overview, June 1995.
18. S3, Scenic/MX2 Product Overview, June 1995.
19. S3, Sonic/AD Product Overview, June 1995.
20. Trident Microsystems, Inc., TVP9512 Datasheet.
21. Tseng Labs, VIPeR Datasheet.
22. Tseng Labs, VIPeR f/x Datasheet, March 1995.
23. Tseng Labs, W32 Datasheet.
24. Tseng Labs, W32i Datasheet.
25. Tseng Labs, W32p Datasheet.
26. VESA Advanced Feature Connector (VAFC) Standard, November 1, 1993.
27. VESA Advanced Feature Connector (VAFC) Software Interface Standard, March 30, 1994.
28. VESA Media Channel (VMChannel) Standard, December 1, 1993.
29. VESA VM-Channel (VMC) Software Interface Standard, Baseline Implementation, September 23, 1994.
30. Weitek Corporation, Power 9100 Technical Overview, November 1993.
31. Weitek Corporation, Video Power Technical Overview, October 1993.

Color Spaces

A color space is a mathematical representation of a set of colors. Three fundamental color models are RGB (used in color computer graphics and color television); YIQ, YUV, or YCbCr (used in broadcast and television systems); and CMYK (used in color printing). However, none of these color spaces are directly related to the intuitive notions of hue, saturation, and brightness. This has resulted in the development of other models, such as HSI and HSV, to simplify programming, processing, and end-user manipulation.

All of the color spaces in common use can be derived from the RGB information supplied by devices like cameras and scanners.

RGB Color Space

The red, green, and blue (RGB) color space is widely used throughout computer graphics and imaging. Red, green, and blue are three primary additive colors (individual components are added together to form a desired color) and are represented by a three-dimensional, Cartesian coordinate system (Figure 3.1). The indicated diagonal of the cube, with equal amounts of each primary component, represents various gray levels. Table 3.1 contains the RGB values for 100% amplitude, 100% saturated color bars, a common video test signal.

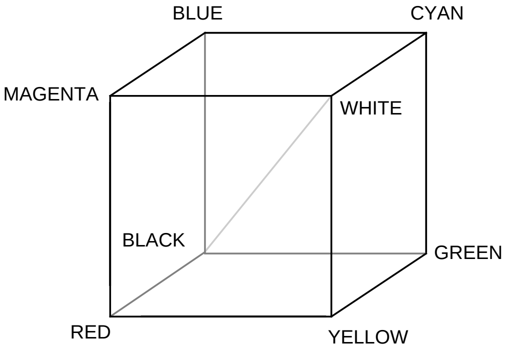


Figure 3.1. The RGB Color Cube.

	Nominal Range	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
R	0 to 255	255	255	0	0	255	255	0	0
G	0 to 255	255	255	255	255	0	0	0	0
B	0 to 255	255	0	255	0	255	0	255	0

Table 3.1. 100% Amplitude, 100% Saturated RGB Color Bars.

The RGB color space is the most prevalent choice for graphics frame buffers because color CRTs use red, green, and blue phosphors to create the desired color. Therefore, the choice of the RGB color space for a graphics frame buffer simplifies the architecture and design of the system. Also, a system that is designed using the RGB color space can take advantage of a large number of existing software routines, since this color space has been around for a number of years.

However, RGB is not very efficient when dealing with "real-world" images. All three RGB components need to be of equal bandwidth to generate any color within the RGB color cube. The result of this is a frame buffer that has the same pixel depth and display resolution for each RGB component. Also, processing an image in the RGB color space is not the most efficient method. For example, to modify the intensity or color of a given pixel, the three RGB values must be read from the frame buffer, the intensity or color calculated, the desired modifications performed, and the new RGB values calculated and written back to the frame buffer. If the system had access to an image stored directly in the intensity and color format, some processing steps would be faster. For these and other reasons, many broadcast, video, and imaging standards use luminance and color difference video signals. These may exist as YUV, YIQ, or YCbCr color spaces. Although all are related, there are some differences.

YUV Color Space

The YUV color space is the basic color space used by the PAL (Phase Alternation Line), NTSC (National Television System Committee), and SECAM (Sequentiel Couleur Avec Mémoire or Sequential Color with Memory) composite color video standards. The black-and-white system used only luma (Y) information; color information (U and V) was added in a such a way that a black-and-white receiver would still display a normal black-and-white picture. Color receivers decoded the additional color information to display a color picture.

The basic equations to convert between gamma-corrected RGB (R´G´B´) and YUV are:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$\begin{aligned} U &= -0.147R' - 0.289G' + 0.436B' \\ &= 0.492\,(B' - Y) \end{aligned}$$

$$\begin{aligned} V &= 0.615R' - 0.515G' - 0.100B' \\ &= 0.877(R' - Y) \end{aligned}$$

(*Note*: In video applications, the terms luma and luminance are commonly interchanged; they both refer to the black-and-white information in the video signal. Chroma and chrominance, which both refer to the color information, are also commonly interchanged. Luma and chroma are the more technically correct terms.)

$$R' = Y + 1.140V$$

$$G' = Y - 0.394U - 0.581V$$

$$B' = Y + 2.032U$$

(The prime symbol indicates gamma-corrected RGB, explained at the end of this chapter.)

For digital RGB values with a range of 0 to 255, Y has a range of 0 to 255, U a range of 0 to ±112, and V a range of 0 to ±157. These equations are usually scaled to simplify the implementation in an actual NTSC or PAL digital encoder or decoder.

If the full range of (B´ – Y) and (R´ – Y) had been used, the modulated color information levels would have exceeded what the (then current) black-and-white television transmitters and receivers were capable of supporting. Experimentation determined that modulated subcarrier excursions of 20% of the luma (Y) signal excursion could be permitted above white and below black. The scaling factors were then selected so that the maximum level of 75% amplitude, 100% saturation yellow and cyan color bars would be at the white level (100 IRE).

YIQ Color Space

The YIQ color space is derived from the YUV color space and is used optionally by the NTSC composite color video standard. (The “I” stands for “in-phase” and the “Q” for “quadrature,” which is the modulation method used to transmit the color information.) The basic equations to convert between gamma-corrected RGB (R´G´B´) and YIQ are:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$\begin{aligned} I &= 0.596R' - 0.275G' - 0.321B' \\ &= V\cos 33^\circ - U\sin 33^\circ \\ &= 0.736(R' - Y) - 0.268(B' - Y) \end{aligned}$$

$$\begin{aligned} Q &= 0.212R' - 0.523G' + 0.311B' \\ &= V\sin 33^\circ + U\cos 33^\circ \\ &= 0.478(R' - Y) + 0.413(B' - Y) \end{aligned}$$

or, using matrix notation:

$$\begin{bmatrix} I \\ Q \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \cos(33) & \sin(33) \\ -\sin(33) & \cos(33) \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix}$$

$$R' = Y + 0.956I + 0.620Q$$

$$G' = Y - 0.272I - 0.647Q$$

$$B' = Y - 1.108I + 1.705Q$$

For digital RGB values with a range of 0 to 255, Y has a range of 0 to 255, I has a range of 0 to ±152, and Q has a range of 0 to ±134. I and Q are obtained by rotating the U and V axes 33°. These equations are usually scaled to simplify the implementation in an actual NTSC digital encoder or decoder.

YDbDr Color Space

The YDbDr color space is used by the SECAM composite color video standard. The black-and-white system used only luma (Y) information; color information (Db and Dr) was added in a such a way that a black-and-white receiver would still display a normal black-and-white picture; color receivers decode the additional color information to display a color picture.

The basic equations to convert between gamma-corrected RGB and YDbDr are:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$\begin{aligned} Db &= 1.505(B' - Y) \\ &= -0.450R' - 0.883G' + 1.333B' \end{aligned}$$

$$\begin{aligned} Dr &= -1.902(R' - Y) \\ &= -1.333R' + 1.116G' + 0.217B' \end{aligned}$$

$$R' = Y - 0.526Dr$$
$$G' = Y - 0.129Db + 0.268Dr$$
$$B' = Y + 0.665Db$$

For digital RGB values with a range of 0 to 255, Y has a range of 0 to 255, and Db and Dr have a range of 0 to ± 340. These equations are usually scaled to simplify the implementation in an actual SECAM digital encoder or decoder.

YCbCr Color Space

The YCbCr color space was developed as part of Recommendation ITU-R BT.601 (formerly CCIR 601) during the development of a worldwide digital component video standard (discussed in Chapter 8). YCbCr is a scaled and offset version of the YUV color space. Y is defined to have a nominal range of 16 to 235; Cb and Cr are defined to have a range of 16 to 240, with 128 equal to zero. There are several YCbCr sampling formats, such as 4:4:4, 4:2:2, 4:1:1, and 4:2:0 that are also described.

The basic equations to convert between digital gamma-corrected RGB (R´G´B´) signals with a 16 to 235 nominal range and YCbCr are:

$$Y = (77/256)R' + (150/256)G' + (29/256)B'$$

$$Cb = -(44/256)R' - (87/256)G' + (131/256)B' + 128$$

$$Cr = (131/256)R' - (110/256)G' - (21/256)B' + 128$$

$$R' = Y + 1.371(Cr - 128)$$

$$G' = Y - 0.698(Cr - 128) - 0.336(Cb - 128)$$

$$B' = Y + 1.732(Cb - 128)$$

When performing YCbCr to RGB conversion, the resulting gamma-corrected RGB (R´G´B´) values have a nominal range of 16–235, with possible occasional excursions into the 0–15 and 236–255 values. This is due to Y and CbCr occasionally going outside the 16–235 and 16–240 ranges, respectively, due to video processing. Table 3.2 lists the YCbCr values for 75% amplitude, 100% saturated color bars, a common video test signal.

Computer Systems Considerations

If the gamma-corrected RGB (R´G´B´) data has a range of 0 to 255, as is commonly found in computer systems, the following equations may be more convenient to use:

	Nominal Range	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
Y	16 to 235	180	162	131	112	84	65	35	16
Cb	16 to 240 (128 = zero)	128	44	156	72	184	100	212	128
Cr	16 to 240 (128 = zero)	128	142	44	58	198	212	114	128

Table 3.2. 75% Amplitude, 100% Saturated YCbCr Color Bars.

$$Y = 0.257R' + 0.504G' + 0.098B' + 16$$

$$Cb = -0.148R' - 0.291G' + 0.439B' + 128$$

$$Cr = 0.439R' - 0.368G' - 0.071B' + 128$$

$$R' = 1.164(Y - 16) + 1.596(Cr - 128)$$

$$G' = 1.164(Y - 16) - 0.813(Cr - 128) - 0.392(Cb - 128)$$

$$B' = 1.164(Y - 16) + 2.017(Cb - 128)$$

Note that for the YCbCr-to-RGB equations, the RGB values must be saturated at the 0 and 255 levels due to occasional excursions outside the nominal YCbCr ranges.

4:4:4 YCbCr Format

Figure 3.2 illustrates the positioning of YCbCr samples for the 4:4:4 format. Each sample has a Y, a Cb, and a Cr value. Each sample is typically 8 bits (consumer applications) or 10 bits (editing applications) per component. Each sample therefore requires 24 bits (or 30 bits for editing applications).

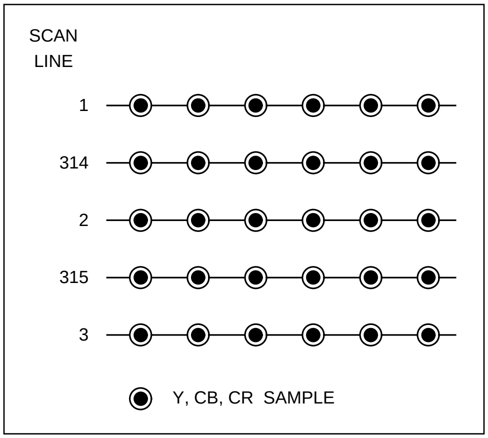


Figure 3.2. 4:4:4 Orthogonal Sampling. The position of sampling sites on the scan lines of a 625-line interlaced picture.

4:2:2 YCbCr Format

Figure 3.3 illustrates the positioning of YCbCr samples for the 4:2:2 format. For every two horizontal Y samples, there is one Cb and Cr sample. Each sample is typically 8 bits (consumer applications) or 10 bits (editing applications) per component. In a frame buffer, each sample requires 16 bits (or 20 bits for editing applications), usually formatted as shown in Figure 3.4. During display, Y samples that have no Cb and Cr data use interpolated Cb and Cr data from the previous and next samples that do.

4:1:1 YCbCr Format

Figure 3.5 illustrates the positioning of YCbCr samples for the 4:1:1 format, used in consumer video applications (it has been mostly replaced by the 4:2:2 format). For every four horizontal Y samples, there is one Cb and Cr value. Each component is typically 8 bits. In a frame buffer, each sample requires 12 bits, usually format-

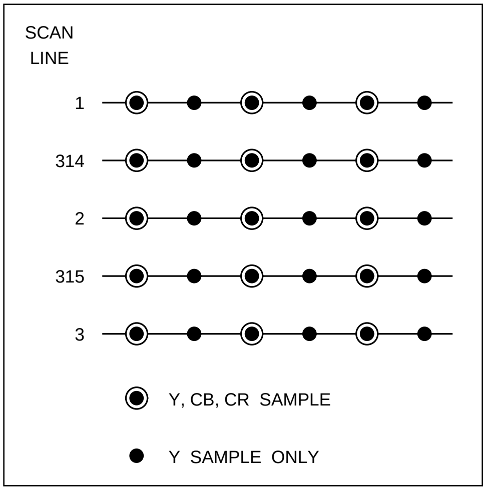


Figure 3.3. 4:2:2 Orthogonal Sampling. The position of sampling sites on the scan lines of a 625-line interlaced picture.

PIXEL 0	PIXEL 1	PIXEL 2	PIXEL 3	PIXEL 4	PIXEL 5
Y7 - 0	Y7 - 1	Y7 - 2	Y7 - 3	Y7 - 4	Y7 - 5
Y6 - 0	Y6 - 1	Y6 - 2	Y6 - 3	Y6 - 4	Y6 - 5
Y5 - 0	Y5 - 1	Y5 - 2	Y5 - 3	Y5 - 4	Y5 - 5
Y4 - 0	Y4 - 1	Y4 - 2	Y4 - 3	Y4 - 4	Y4 - 5
Y3 - 0	Y3 - 1	Y3 - 2	Y3 - 3	Y3 - 4	Y3 - 5
Y2 - 0	Y2 - 1	Y2 - 2	Y2 - 3	Y2 - 4	Y2 - 5
Y1 - 0	Y1 - 1	Y1 - 2	Y1 - 3	Y1 - 4	Y1 - 5
Y0 - 0	Y0 - 1	Y0 - 2	Y0 - 3	Y0 - 4	Y0 - 5
CB7 - 0	CR7 - 0	CB7 - 2	CR7 - 2	CB7 - 4	CR7 - 4
CB6 - 0	CR6 - 0	CB6 - 2	CR6 - 2	CB6 - 4	CR6 - 4
CB5 - 0	CR5 - 0	CB5 - 2	CR5 - 2	CB5 - 4	CR5 - 4
CB4 - 0	CR4 - 0	CB4 - 2	CR4 - 2	CB4 - 4	CR4 - 4
CB3 - 0	CR3 - 0	CB3 - 2	CR3 - 2	CB3 - 4	CR3 - 4
CB2 - 0	CR2 - 0	CB2 - 2	CR2 - 2	CB2 - 4	CR2 - 4
CB1 - 0	CR1 - 0	CB1 - 2	CR1 - 2	CB1 - 4	CR1 - 4
CB0 - 0	CR0 - 0	CB0 - 2	CR0 - 2	CB0 - 4	CR0 - 4

16 BITS PER PIXEL

- 0 = PIXEL 0 DATA
- 1 = PIXEL 1 DATA
- 2 = PIXEL 2 DATA
- 3 = PIXEL 3 DATA
- 4 = PIXEL 4 DATA

Figure 3.4. 4:2:2 Frame Buffer Formatting.

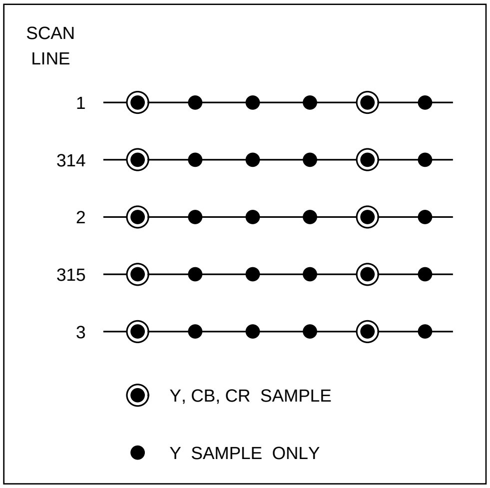


Figure 3.5. 4:1:1 Orthogonal Sampling. The position of sampling sites on the scan lines of a 625-line interlaced picture.

ted as shown in Figure 3.6. During display, Y samples with no Cb and Cr data use interpolated Cb and Cr data from the previous and next pixels that do.

4:2:0 YCbCr Format

Figure 3.7 illustrates the positioning of YCbCr samples for the 4:2:0 format used by the H.261 and H.263 video teleconferencing standards and the MPEG 1 video compression standard (the MPEG 2 4:2:0 format is slightly different). Rather than the horizontal-only 4:1 reduction of Cb and Cr used by 4:1:1, 4:2:0 implements a 2:1 reduction of Cb and Cr in both the vertical and horizontal directions.

PhotoYCC Color Space

PhotoYCC (a trademark of Eastman Kodak Company) was developed by Kodak to encode Photo CD image data. The goal was to develop a display-device-independent color space. For maximum video display efficiency, the color space is based upon Recommendation ITU-R BT.601 (formerly CCIR 601) and Recommendation ITU-R BT.709 (formerly CCIR 709).

The encoding process (RGB to PhotoYCC) assumes the illumination is CIE Standard Illuminant D_{65} and that the spectral sensitivities of the image capture system are proportional to the color-matching functions of the Recommendation ITU-R BT.709 reference primaries. The RGB values, unlike those for a computer graphics system, may be negative. The color gamut of PhotoYCC includes colors outside the Recommendation ITU-R BT.709 display phosphor limits.

RGB to PhotoYCC

Linear RGB data (normalized to have values of 0 to 1) is nonlinearly transformed to PhotoYCC as follows:

PIXEL 0	PIXEL 1	PIXEL 2	PIXEL 3	PIXEL 4	PIXEL 5
Y7 - 0	Y7 - 1	Y7 - 2	Y7 - 3	Y7 - 4	Y7 - 5
Y6 - 0	Y6 - 1	Y6 - 2	Y6 - 3	Y6 - 4	Y6 - 5
Y5 - 0	Y5 - 1	Y5 - 2	Y5 - 3	Y5 - 4	Y5 - 5
Y4 - 0	Y4 - 1	Y4 - 2	Y4 - 3	Y4 - 4	Y4 - 5
Y3 - 0	Y3 - 1	Y3 - 2	Y3 - 3	Y3 - 4	Y3 - 5
Y2 - 0	Y2 - 1	Y2 - 2	Y2 - 3	Y2 - 4	Y2 - 5
Y1 - 0	Y1 - 1	Y1 - 2	Y1 - 3	Y1 - 4	Y1 - 5
Y0 - 0	Y0 - 1	Y0 - 2	Y0 - 3	Y0 - 4	Y0 - 5
CB7 - 0	CB5 - 0	CB3 - 0	CB1 - 0	CB7 - 4	CB5 - 4
CB6 - 0	CB4 - 0	CB2 - 0	CB0 - 0	CB6 - 4	CB4 - 4
CR7 - 0	CR5 - 0	CR3 - 0	CR1 - 0	CR7 - 4	CR5 - 4
CR6 - 0	CR4 - 0	CR2 - 0	CR0 - 0	CR6 - 4	CR4 - 4

12 BITS PER PIXEL

- 0 = PIXEL 0 DATA
- 1 = PIXEL 1 DATA
- 2 = PIXEL 2 DATA
- 3 = PIXEL 3 DATA
- 4 = PIXEL 4 DATA

Figure 3.6. 4:1:1 Frame Buffer Formatting.

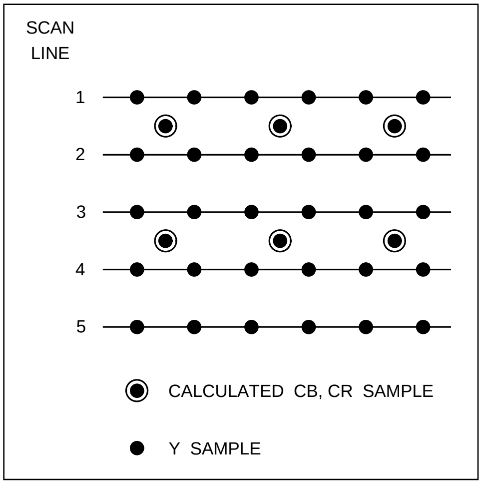


Figure 3.7. 4:2:0 Coded Picture Sampling for H.261, H.263, and MPEG 1. The position of sampling sites on the scan lines of a non-interlaced picture.

for $R, G, B \geq 0.018$

$$R' = 1.099\ R^{0.45} - 0.099$$
$$G' = 1.099\ G^{0.45} - 0.099$$
$$B' = 1.099\ B^{0.45} - 0.099$$

for $R, G, B \leq -0.018$

$$R' = -1.099\ |R|^{0.45} - 0.099$$
$$G' = -1.099\ |G|^{0.45} - 0.099$$
$$B' = -1.099\ |B|^{0.45} - 0.099$$

for $-0.018 < R, G, B < 0.018$

$$R' = 4.5\ R$$
$$G' = 4.5\ G$$
$$B' = 4.5\ B$$

From R′, G′, and B′, a luminance (luma) and two chrominance signals (chroma1 and chroma2) are generated:

$$\text{luma} = 0.299R' + 0.587G' + 0.114B' = Y$$
$$\text{chroma1} = -0.299R' - 0.587G' + 0.866B' = B' - Y$$
$$\text{chroma2} = 0.701R' - 0.587G' - 0.114B' = R' - Y$$

These are quantized and limited to the 8-bit range of 0 to 255:

$$\text{luma} = (255\ /\ 1.402)\ \text{luma}$$
$$\text{chroma1} = (111.40\ \text{chroma1}) + 156$$
$$\text{chroma2} = (135.64\ \text{chroma2}) + 137$$

As an example, a 20% gray value (R, G, and B = 0.2) would be recorded on the Photo CD disc using the following values:

luma = 79
chroma1 = 156
chroma2 = 137

PhotoYCC to RGB

Since PhotoYCC attempts to preserve the dynamic range of film, decoding PhotoYCC images requires the selection of a color space and range appropriate for the output device. Thus, the decoding equations are not always the exact inverse of the encoding equations. The following equations are suitable for generating RGB values for driving a display, and assume a unity relationship between the luminance in the encoded image and the displayed image.

$L' = 1.3584$ (luma)
$C1 = 2.2179$ (chroma1 – 156)
$C2 = 1.8215$ (chroma2 – 137)

$R_{display} = L' + C2$
$G_{display} = L' - 0.194(C1) - 0.509(C2)$
$B_{display} = L' + C1$

The RGB values should be limited to a range of 0 to 255. The equations above assume the display uses phosphor chromaticities that are the same as the Recommendation ITU-R BT.709 reference primaries, and that the video signal luminance (V) and the display luminance (L) have the relationship:

for $0.0812 \le V < 1.0$

$$L = ((V + 0.099) / 1.099)^{2.2}$$

for $0 \le V < 0.0812$

$$L = V / 4.5$$

HSI, HLS, and HSV Color Spaces

The HSI (hue, saturation, intensity) and HSV (hue, saturation, value) color spaces were developed to be more “intuitive” in manipulating color and were designed to approximate the way humans perceive and interpret color. They were developed when colors had to be specified manually, and are rarely used now that users can select colors visually or specify Pantone colors. These color spaces are primarily discussed for “historic” interest. HLS (hue, lightness, saturation) is similar to HSI; the term lightness is used rather than intensity.

The difference between HSI and HSV is the computation of the brightness component (I or V), which determines the distribution and dynamic range of both the brightness (I or V) and saturation (S). The HSI color space is best for traditional image processing functions such as convolution, equalization, histograms, and so on, which operate by manipulation of the brightness values since I is equally dependent on R, G, and B. The HSV color space is preferred for manipulation of hue and saturation (to shift colors or adjust the amount of color) since it yields a greater dynamic range of saturation.

Figure 3.8 illustrates the single hexcone HSV color model. The top of the hexcone corresponds to V = 1, or the maximum intensity colors. The point at the base of the hexcone is black and here V = 0. Complementary colors are 180° opposite one another as measured by H, the angle around the vertical axis (V), with red at 0°. The value of S is a ratio, ranging from 0 on the center line vertical axis (V) to 1 on the sides of the hexcone. Any value of S between 0 and 1 may be associated with the point V = 0. The point S = 0, V = 1 is white. Intermediate values of V for S = 0 are the grays. Note that when S = 0, the value of H is irrelevant. From an artist’s viewpoint, any color with V = 1, S = 1 is a pure pigment (whose color is defined by H). Adding white corresponds to decreasing S (without changing V); adding black corresponds to decreasing V (without changing S). Tones are created by decreasing both S and V. Table 3.3 lists the 75% amplitude, 100% saturated HSV color bars.

Figure 3.9 illustrates the double hexcone HSI color model. The top of the hexcone corresponds to I = 1, or white. The point at the base of the hexcone is black and here I = 0. Comple-

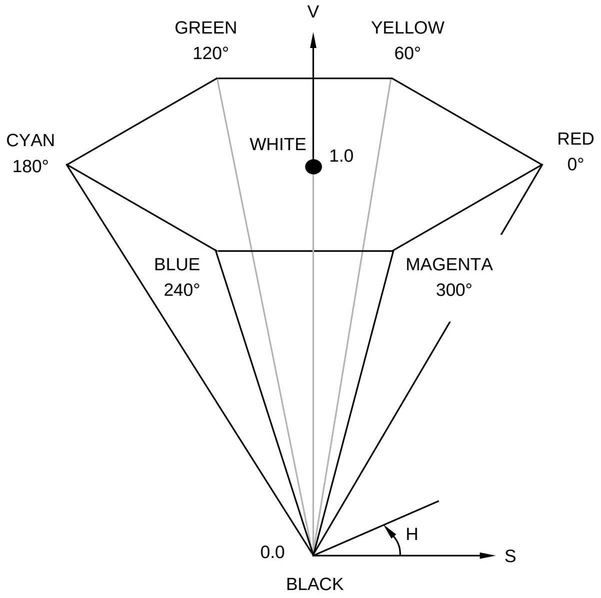


Figure 3.8. Single Hexcone HSV Color Model.

mentary colors are 180° opposite one another as measured by H, the angle around the vertical axis (I), with red at 0° (for consistency with the HSV model, we have changed from the Tektronix convention of blue at 0°). The value of S ranges from 0 on the vertical axis (I) to 1 on the surfaces of the hexcone. The grays all have S = 0, but maximum saturation of hues is at S = 1, I = 0.5. Table 3.4 lists the 75% amplitude, 100% saturated HSI color bars.

There are several ways of converting between HSI or HSV and RGB. Although based on the same principles, they are implemented slightly differently.

	Nominal Range	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
H	0° to 360°	x	60°	180°	120°	300°	0°	240°	x
S	0 to 1	0	1	1	1	1	1	1	0
V	0 to 1	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0

Table 3.3. 75% Amplitude, 100% Saturated HSV Color Bars.

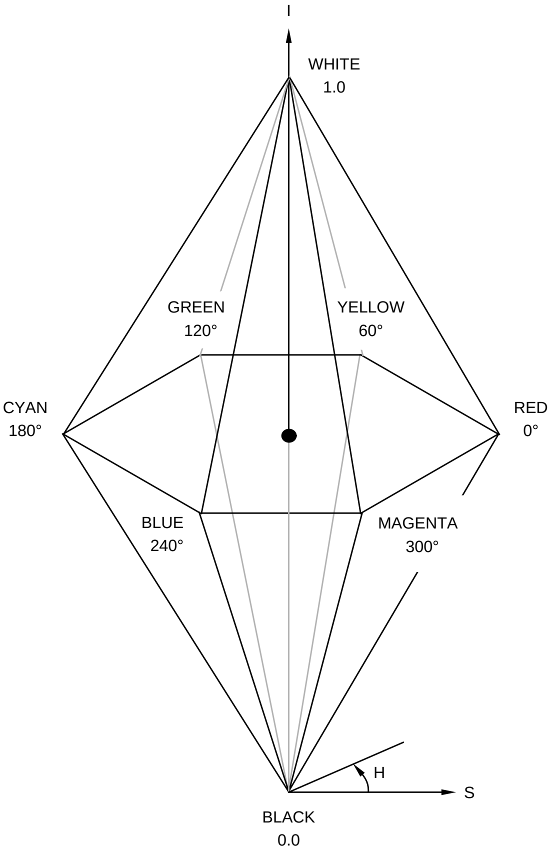


Figure 3.9. Double Hexcone HSI Color Model. For consistency with the HSV model, we have changed from the Tektronix convention of blue at 0° and depict the model as a double hexcone rather than as a double cone.

	Nominal Range	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
H	0° to 360°	x	60°	180°	120°	300°	0°	240°	x
S	0 to 1	0	1	1	1	1	1	1	0
I	0 to 1	0.75	0.375	0.375	0.375	0.375	0.375	0.375	0

Table 3.4. 75% Amplitude, 100% Saturated HSI Color Bars. For consistency with the HSV model, we have changed from the Tektronix convention of blue at 0°.

HSV-to-RGB and RGB-to-HSV Conversion

This conversion is similar to that described in the section entitled “HSI-to-RGB and RGB-to-HSI conversion.”

RGB-to-HSV Conversion

Setup equations (RGB range of 0 to 1):

$M = \max(R,G,B)$
$m = \min(R,G,B)$
$r = (M - R) / (M - m)$
$g = (M - G) / (M - m)$
$b = (M - B) / (M - m)$

Value calculation (value range of 0 to 1):

$V = \max(R,G,B)$

Saturation calculation (saturation range of 0 to 1):

if $M = 0$ then $S = 0$ and $H = 180°$
if $M \neq 0$ then $S = (M - m) / M$

Hue calculation (hue range of 0 to 360):

if $R = M$ then $H = 60(b - g)$
if $G = M$ then $H = 60(2 + r - b)$
if $B = M$ then $H = 60(4 + g - r)$
if $H \geq 360$ then $H = H - 360$
if $H < 0$ then $H = H + 360$

HSV-to-RGB Conversion

Setup equations:

if $S = 0$ then $H = 180°$, $R = V$, $G = V$,
and $B = V$
otherwise

if $H = 360$ then $H = 0$
$h = H / 60$
i = largest integer of h
$f = h - i$
$p = V * (1 - S)$
$q = V * (1 - (S * f))$
$t = V * (1 - (S * (1 - f)))$

RGB calculations (RGB range of 0 to 1)

if $i = 0$ then $(R,G,B) = (V,t,p)$
if $i = 1$ then $(R,G,B) = (q,V,p)$
if $i = 2$ then $(R,G,B) = (p,V,t)$
if $i = 3$ then $(R,G,B) = (p,q,V)$
if $i = 4$ then $(R,G,B) = (t,p,V)$
if $i = 5$ then $(R,G,B) = (V,p,q)$

HSI-to-RGB and RGB-to-HSI Conversion (Method 1)

In this implementation, intensity is calculated as the average of the largest and smallest primary values. Saturation is the ratio between the difference and sum of the two primary values, with the difference corresponding to color content and the sum corresponding to color plus white. Two sets of equations for calculating hue are provided: one set with red at 0° (to be compatible with the HSV convention) and one set with blue at 0° (Tektronix model).

RGB-to-HSI Conversion

Setup equations (RGB range of 0 to 1):

$M = \max(R,G,B)$
$m = \min(R,G,B)$
$r = (M - R) / (M - m)$
$g = (M - G) / (M - m)$
$b = (M - B) / (M - m)$

Intensity calculation (intensity range of 0 to 1):

$I = (M + m) / 2$

Saturation calculation (saturation range of 0 to 1):

if $M = m$ then $S = 0$ and $H = 180°$
if $I \leq 0.5$ then $S = (M - m) / (M + m)$
if $I > 0.5$ then $S = (M - m) / (2 - M - m)$

Hue calculation (hue range of 0 to 360):

```
red = 0°
if $R = M$ then $H = 60(b - g)$
if $G = M$ then $H = 60(2 + r - b)$
if $B = M$ then $H = 60(4 + g - r)$

if $H \geq 360$ then $H = H - 360$
if $H < 0$ then $H = H + 360$

blue = 0°
if $R = M$ then $H = 60(2 + b - g)$
if $G = M$ then $H = 60(4 + r - b)$
if $B = M$ then $H = 60(6 + g - r)$

if $H \geq 360$ then $H = H - 360$
if $H < 0$ then $H = H + 360$
```

HSI-to-RGB Conversion

Two sets of equations for calculating RGB values are provided: one set with red at 0° (to be compatible with the HSV convention) and one set with blue at 0° (Tektronix model).

Setup equations:

```
if $I \leq 0.5$ then $M = I(1 + S)$
if $I > 0.5$ then $M = I + S - IS$
$m = 2I - M$
if $S = 0$ then $R = G = B = I$ and $H = 180°$
```

Equations for calculating R (range of 0 to 1):

```
red = 0°
if $H < 60$ then $R = M$
if $H < 120$ then $R = m + ((M - m) / ((120 - H) / 60))$
if $H < 240$ then $R = m$
if $H < 300$ then $R = m + ((M - m) / ((H - 240) / 60))$
otherwise $R = M$

blue = 0°
if $H < 60$ then $R = m + ((M - m) / (H / 60))$
if $H < 180$ then $R = M$
if $H < 240$ then $R = m + ((M - m) / ((240 - H) / 60))$
otherwise $R = m$
```

Equations for calculating G (range of 0 to 1):

```
red = 0°
if $H < 60$ then $G = m + ((M - m) / (H / 60))$
if $H < 180$ then $G = M$
if $H < 240$ then $G = m + ((M - m) / ((240 - H) / 60))$
otherwise $G = m$

blue = 0°
if $H < 120$ then $G = m$
if $H < 180$ then $G = m + ((M - m) / ((H - 120) / 60))$
if $H < 300$ then $G = M$
otherwise $G = m + ((M - m) / ((360 - H) / 60))$
```

Equations for calculating B (range of 0 to 1):

```
red = 0°
if $H < 120$ then $B = m$
if $H < 180$ then $B = m + ((M - m) / ((H - 120) / 60))$if $H < 300$ then $B = M$
otherwise $B = m + ((M - m) / ((360 - H) / 60))$

blue = 0°
if $H < 60$ then $B = M$
if $H < 120$ then $B = m + ((M - m) / ((120 - H) / 60))$
if $H < 240$ then $B = m$
if $H < 300$ then $B = m + ((M - m) / ((H - 240) / 60))$
otherwise $B = M$
```

HSI-to-RGB and RGB-to-HSI Conversion (Method 2)

This implementation is used by Data Translation in their RGB/HSI converters. Intensity is calculated as the average of R, G, and B. Saturation (S) is obtained by subtracting the lowest value of (R/I), (G/I), or (B/I) from 1. The RGB values have a normalized range of 0 to 1.

RGB-to-HSI Conversion

Setup equations

if $G = B$ then $F = 0$
if $G \neq B$ then $F = ((2R - G - B) / (G - B)) / \mathrm{SQRT}(3)$

if $F \neq 0$ then $A = \arctan(F)$
if $F = 0$ and $R > (G \text{ or } B)$ then $A = +90$
otherwise $A = -90$

if $G \geq B$ then $X = 0$
if $G < B$ then $X = 180$

Intensity calculation (range of 0 to 1)

$$I = (R + G + B) / 3$$

Saturation calculation (range of 0 to 1):

if $I = 0$ then $S = 0$
if $I \neq 0$ then $S = 1 - (\min(R,G,B) / I)$

Hue calculation (range of 0 to 360):

$$H = 90 - A + X$$

HSI-to-RGB Conversion

L equals the min(R,G,B) value. M is the color 120° counterclockwise from L, and N is the color 120° counterclockwise from M. The SEL_0 and SEL_1 signals are used to map L, M, and N to R, G, and B, as shown below.

	SEL_1	**SEL_0**	**K**
$0° < H \leq 120°$	0	0	$(\cos H) / (\cos (60° - H))$
$120° < H \leq 240°$	0	1	$(\cos (H - 120°)) / (\cos (60° - H + 120°))$
$240° < H \leq 360°$	1	1	$(\cos (H - 240°)) / (\cos (60° - H + 240°))$

	SEL_1 = 0 **SEL_0 = 0**	**SEL_1 = 0** **SEL_0 = 1**	**SEL_1 = 1** **SEL_0 = 1**
$L = I - IS$	blue	red	green
$M = I + ISK$	red	green	blue
$N = 3I - (L + M)$	green	blue	red

CMYK Color Space

The CMYK (cyan, magenta, yellow, black) color space commonly is used in color printers, due to the subtractive properties of inks. Cyan, magenta, and yellow are the complements of red, green, and blue, respectively, as shown in Figure 3.10, and are subtractive primaries because their effect is to subtract some color from white light. Color is specified by what is removed (or subtracted) from white light. When a surface is coated with cyan ink, no red light is reflected. Cyan subtracts red from the reflected white light (which is the sum of red, green, and blue). Therefore, in terms of additive primaries, cyan is blue plus green. Similarly, magenta absorbs green so it is red plus blue, while yellow absorbs blue so it is red plus green. A surface coated with cyan and yellow ink absorbs red and blue, leaving only green to be reflected from white light. A cyan, yellow, and magenta surface absorbs red, green, and blue, and there is black. However, to maintain black color purity, a separate black ink is used rather than printing cyan, magenta, and yellow to generate black. As an interesting side note, white cannot be generated unless a white paper is used (i.e., white cannot be generated on blue paper).

RGB-to-CMYK Considerations

Lookup tables are typically used to nonlinear process the RGB data prior to converting it to the CMY color space. This takes into account the inks and paper used and is different from the RGB gamma correction used when generating video. Some systems have additional bit planes for black data, resulting in a 32-bit frame buffer (assuming 8 bits each for red, green, blue, and black). In this case, a lookup table for the black data is also useful.

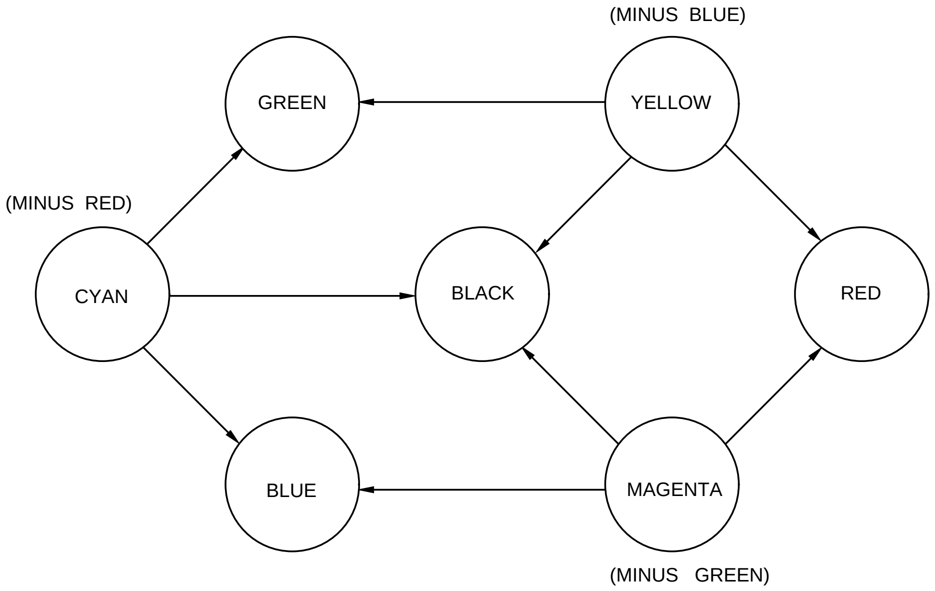


Figure 3.10. Subtractive Primaries (Cyan, Magenta, Yellow) and Their Mixtures.

Since ideally CMY is the complement of RGB, the following linear equations (known also as masking equations) were used initially to convert between RGB and CMY:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} C \\ M \\ Y \end{bmatrix}$$

However, more accurate transformations account for the dependency of the inks and paper used. Slight adjustments, such as mixing the CMY data, are usually necessary. Yellow ink typically provides a relatively pure yellow (it absorbs most of the blue light and reflects practically all the red and green light). Magenta ink typically does a good job of absorbing green light, but absorbs too much of the blue light (visually, this makes it too reddish). The extra redness in the magenta ink may be compensated for by a reduction of yellow in areas that contain magenta. Cyan ink absorbs most of the red light (as it should) but also much of the green light (which it should reflect, making cyan more bluish than it should be). The extra blue in cyan ink may be compensated for by a reduction of magenta in areas that contain cyan. All of these simple color adjustments, as well as the linear conversion from RGB to CMY, may be done using a 3 × 3 matrix multiplication:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} m1 & m2 & m3 \\ m4 & m5 & m6 \\ m7 & m8 & m9 \end{bmatrix} \begin{bmatrix} 1 - R \\ 1 - G \\ 1 - B \end{bmatrix}$$

The coefficients m1, m5, and m9 are values near unity, and the other coefficients are small in comparison. The RGB lookup tables may be used to perform the normalized (1 – R), (1 – G), and (1 – B) conversion, as well as nonlinear correction. In cases where the inks and paper are known and relatively constant (such as a color laser printer), the coefficients may be determined empirically and fixed coefficient multipliers used. Note that saturation circuitry is required on the outputs of the 3 × 3 matrix multiplier. The circuitry must saturate to the maximum digital value (i.e., 255 in an 8-bit system) when an overflow condition occurs and saturate to the minimum digital value (i.e., 0) when an underflow condition occurs. Without the saturation circuitry, overflow or underflow errors may be generated due to the finite precision of the digital logic.

More sophisticated nonlinear RGB-to-CMY conversion techniques are used in high-end systems. These are typically based on trilinear interpolation (using lookup tables) or the Neugebauer equations (which were originally designed to perform the CMYK-to-RGB conversion).

Under Color Removal

Under color removal (UCR) removes some amount (typically 30%) of magenta under cyan and some amount (typically 50%) of yellow under magenta. This may be used to solve problems encountered due to printing an ink when one or more other layers of ink are still wet.

Black Generation

Ideally, cyan, magenta, and yellow are all that are required to generate any color. Equal amounts of cyan, magenta, and yellow should create the equivalent amount of black. In reality, printing inks do not mix perfectly, and dark brown shades are generated instead. Therefore, real black ink is substituted for the mixed-black color to obtain a truer color print. Black generation is the process of calculating the amount of black to be used.

There are several methods of generating black information. The applications software may generate black along with CMY data. In this instance, black need not be calculated; however, UCR and gray component replacement (GCR) may still need to be performed. One common method of generating black is to set the black component equal to the minimum value of (C, M, Y). For example, if

(C, M, Y) = (0.25, 0.5, 0.75)

the black component (K) = 0.25.

In many cases, it is desirable to have a specified minimum value of (C,M,Y) before the generation of any black information. For example, K may increase (either linearly or nonlinearly) from 0 at min(C,M,Y) ≤ 0.5 to 1 at min(C,M,Y) = 1. Adjustability to handle extra black, less black, or no black is also desirable. This can be handled by a programmable lookup table.

Gray Component Replacement

Gray component replacement (GCR) is the process of reducing the amount of cyan, magenta, and yellow components to compensate for the amount of black that is added by black generation. For example, if

(C, M, Y) = (0.25, 0.5, 0.75)

and

black (K) = 0.25

0.25 (the K value) is subtracted from each of the C, M, and Y values, resulting in:

(C, M, Y) = (0, 0.25, 0.5)

black (K) = 0.25

The amount removed from C, M, and Y may be exactly the same amount as the black generation, zero (so no color is removed from the cyan, magenta, and yellow components), or some fraction of the black component. Programmable lookup tables for the cyan, magenta, and yellow colors may be used to implement a nonlinear function on the color data.

Desktop Color Publishing Considerations

Users of color desktop publishing have discovered that the printed colors rarely match what is displayed on the CRT. This is because several factors affect the color of images on the CRT, such as the level of ambient light and its color temperature, what type of phosphors the CRT uses and their age, the brightness and contrast settings of the display, and what inks and paper the color printer uses. All of these factors should be taken into account when accurate RGB-to-CMYK or CMYK-to-RGB conversion is required. Some high-end systems use an intermediate color space to simplify performing the adjustments. Very experienced users know what color will be printed versus the displayed color; however, this is unacceptable for the mass market.

Since the CMYK printable color gamut is less than the displayable RGB color gamut, some systems use the CMYK color space in the frame buffer and perform CMYK-to-RGB conversion to drive the display. This helps prevent the problem of trying to specify an unprintable color.

CIE 1931 Chromaticity Diagram

The color gamut perceived by a person with normal vision (the 1931 CIE Standard Observer) is shown in Figure 3.11. Color perception was measured by viewing combinations of the three standard CIE (International Commission on Illumination or Commission

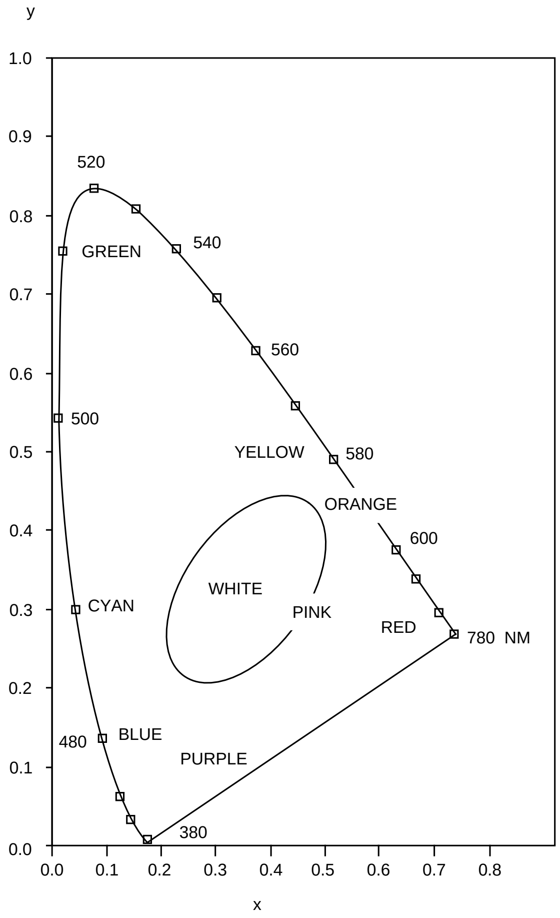


Figure 3.11. CIE 1931 Chromaticity Diagram Showing Various Color Regions.

Internationale de I'Eclairage) primary colors: red with a 700-nm wavelength, green at 546.1 nm, and blue at 435.8 nm. These primary colors, and the other spectrally pure colors resulting from mixing of the primary colors, are located along the outer boundary of the diagram. Colors within the boundary are perceived as becoming more pastel as the center of the diagram (white) is approached.

Each point on the diagram, representing a unique color, may be identified by two coordinates, *x* and *y*. Typically, a camera or display specifies three of these (*x*, *y*) coordinates to define the three primary colors it uses; the triangle formed by the three (*x*, *y*) coordinates encloses the gamut of colors that the camera can capture or the display can reproduce. This is shown in Figure 3.12, which compares the

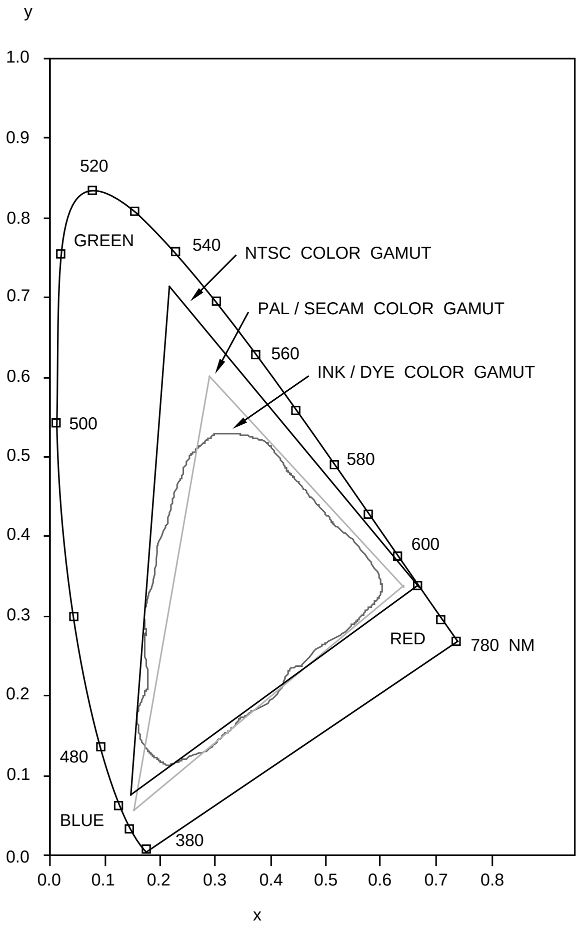


Figure 3.12. CIE 1931 Chromaticity Diagram Showing the Color Gamuts of Phosphors Used in NTSC, PAL, and SECAM Systems and of Typical Inks and Dyes.

color gamuts of NTSC, PAL/SECAM, and typical inks and dyes.

In addition, a camera or display usually specifies the (x, y) coordinate of the white color used, since pure white is not usually captured or reproduced. White is defined as the color captured or produced when all three primary signals are equal, and it has a subtle shade of color to it. The standard “white” in video is Illuminant D_{65} (at $x = 0.313$ and $y = 0.329$). Note that luminance, or brightness information, is not included in the standard CIE 1931 chromaticity diagram, but is an axis that is orthogonal to the (x, y) plane. The lighter a color is, the more restricted the chromaticity range is, much like the HSI and HSV color spaces.

It is interesting to note that over the years color accuracy of television receivers has declined while brightness has increased. Early color televisions didn't sell well because they were dim compared to their black-and-white counterparts. Television manufacturers have changed the display phosphors from the NTSC/PAL standard colors to phosphors that produce brighter images at the expense of color accuracy. Over the years, brightness has become even more important since television is as likely to be viewed in the afternoon sun as in a darkened room.

Non-RGB Color Space Considerations

When processing information in a non-RGB color space (such as YIQ, YUV, YCbCr, etc.), care must be taken that combinations of values are not created that result in the generation of invalid RGB colors. The term invalid refers to RGB components outside specified normalized RGB limits of (1, 1, 1). For example, if the luma (Y) component is at its maximum (white) or minimum (black) level, then any nonzero values of the color difference components (IQ, UV, CbCr, etc.) will give invalid values of RGB. As a specific example, given that RGB has a normalized value of (1, 1, 1), the resulting YCbCr value is (235, 128, 128). If Cb and Cr are manipulated to generate a YCbCr value of (235, 64, 73), the corresponding RGB normalized value becomes (0.6, 1.29, 0.56)—note that the green value exceeds the normalized value of 1. From this illustration it is obvious that there are many combinations of Y, Cb, and Cr that result in invalid RGB values; these YCbCr values must be processed so as to generate valid RGB values. Figure 3.13 shows the RGB normalized limits transformed into the YCbCr color space.

Best results are obtained using a constant luma and constant hue approach—Y is not altered and the color difference values (Cb and Cr in this example) are limited to the maximum valid values having the same hue as the invalid color prior to limiting. The constant hue principle corresponds to moving invalid CbCr combinations directly towards the CbCr origin (128, 128), until they lie on the surface of the valid YCbCr color block.

Color Space Conversion Considerations

ROM lookup tables can be used to implement multiplication factors; however, a minimum of 4 bits of fractional color data (professional applications may require 8 to 10 bits of fractional data) should be maintained up to the final result, which is then rounded to the desired precision.

When converting to the RGB color space from a non-RGB color space, care must be taken to include saturation logic to ensure overflow and underflow conditions do not occur due to the finite precision of digital circuitry. RGB values less than 0 must be set to 0, and values greater than 255 must be set to 255 (assuming 8-bit data).

Gamma Correction

Many display devices use the cathode-ray picture tube (CRT). The transfer function of the CRT produces intensity that is proportional to some power (usually about 2.5 and referred to as gamma) of the signal voltage. As a result, high-intensity ranges are expanded, and low-intensity ranges are compressed (see Figure 3.14). This is an advantage in combatting noise introduced by the transmission process, as the

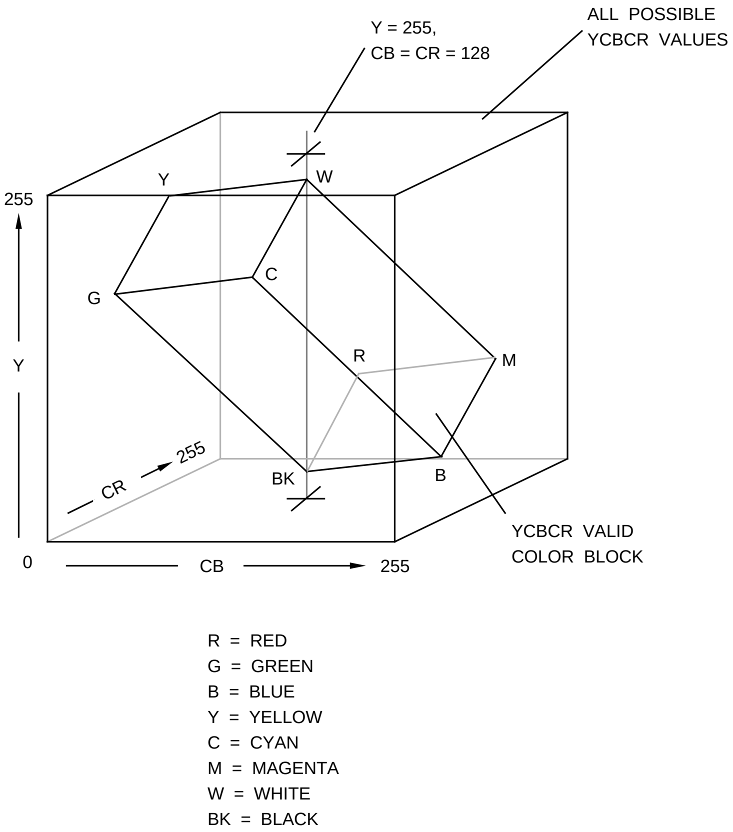


Figure 3.13. RGB Limits Transformed into 3-D YCbCr Space.

eye is approximately equally sensitive to equally relative intensity changes. By “gamma correcting” the video signals before use, the intensity output of the CRT is roughly linear (the gray line in Figure 3.14), and transmission-induced noise is reduced.

Gamma factors of 2.2 are typical in the consumer video environment. Until recently, systems assumed a simple transformation (values are normalized to have a range of 0 to 1):

$$R_{display} = R_{received}{}^{2.2}$$
$$G_{display} = G_{received}{}^{2.2}$$
$$B_{display} = B_{received}{}^{2.2}$$

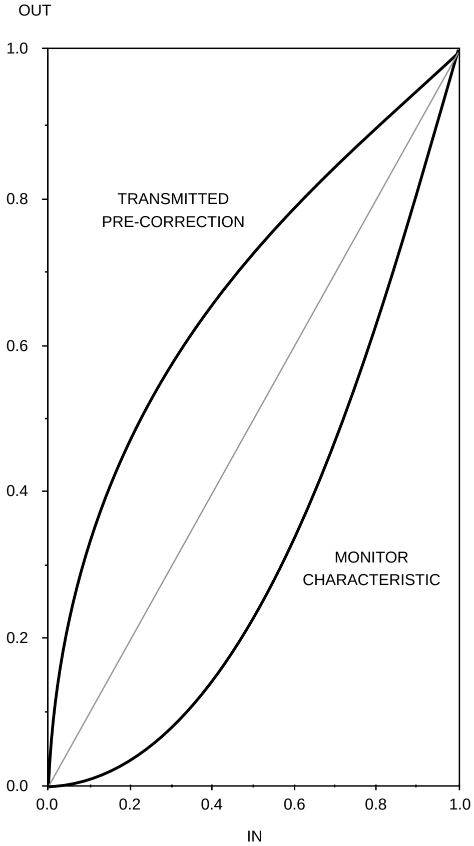


Figure 3.14. The Effect of Gamma Correction.

To compensate for the nonlinear processing at the display, linear RGB data was "gamma-corrected" prior to use (values are normalized to have a range of 0 to 1):

$R_{transmit} = R^{0.45}$
$G_{transmit} = G^{0.45}$
$B_{transmit} = B^{0.45}$

Although NTSC video signals typically assume the receiver has a gamma of 2.2, a gamma of 2.35 to 2.55 is more realistic. However, this difference is often ignored, which improves the viewing in dimly lit environments, such as the home. More accurate viewing in brightly lit environments, such as on the office computer, may be accomplished by applying another gamma factor of 1.1 to 1.2.

To minimize noise in the darker areas of the image, modern video systems limit the gain of the curve in the black region. Assuming values are normalized to have a range of 0 to 1, gamma correction would be applied to linear RGB data prior to use as follows:

for R, G, B < 0.018

$R_{transmit} = 4.5\ R$

$G_{transmit} = 4.5\ G$

$B_{transmit} = 4.5\ B$

for R, G, B ≥ 0.018

$R_{transmit} = 1.099\ R^{0.45} - 0.099$

$G_{transmit} = 1.099\ G^{0.45} - 0.099$

$B_{transmit} = 1.099\ B^{0.45} - 0.099$

This limits the gain close to black and stretches the remainder of the curve to maintain function and tangent continuity.

Although the PAL standards specify a value of 2.8, a value of 2.2 is actually used.

The receiver is assumed to perform the inverse function (values are normalized to have a range of 0 to 1):

for $(R, G, B)_{received} < 0.0812$

$R_{display} = R_{received}\ /\ 4.5$

$G_{display} = G_{received}\ /\ 4.5$

$B_{display} = B_{received}\ /\ 4.5$

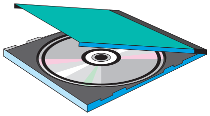
for $(R, G, B)_{received} \geq 0.0812$

$R_{display} = ((R_{received} + 0.099)\ /\ 1.099)^{2.2}$

$G_{display} = ((G_{received} + 0.099)\ /\ 1.099)^{2.2}$

$B_{display} = ((B_{received} + 0.099)\ /\ 1.099)^{2.2}$

VLSI Solutions



See C3_VLSI

References

1. Alexander, George A., *Color Reproduction: Theory and Practice, Old and New. The Seybold Report on Desktop Publishing*, September 20, 1989.
2. Benson, K. Blair, *Television Engineering Handbook*. McGraw-Hill, Inc., 1986.
3. Clarke, C.K.P., 1986, *Colour Encoding and Decoding Techniques for Line-Locked Sampled PAL and NTSC Television Signals*, BBC Research Department Report BBC RD1986/2.
4. Data Translation, DT7910/DT7911 datasheet, 1989.
5. Devereux, V. G., 1987, *Limiting of YUV digital video signals*, BBC Research Department Report BBC RD1987 22.
6. EIA Standard EIA-189-A, July 1976, *Encoded Color Bar Signal*.
7. Faroudja, Yves Charles, *NTSC and Beyond. IEEE Transactions on Consumer Electronics*, Vol. 34, No. 1, February 1988.
8. GEC Plessey Semiconductors, *VP510 datasheet*, January 1993.
9. ITU-R Report BT.470, 1994, *Characteristics of Television Systems*.
10. Philips Semiconductors, *Desktop Video Data Handbook*, 1995.
11. Poynton, Charles A., *A Technical Introduction to Digital Video*, John Wiley & Sons, 1996.
12. Raytheon Semiconductor, 1994 Databook.
13. SGS-Thomson Microelectronics, STV3300 datasheet, September 1989.
14. *Specification of Television Standards for 625-Line System-I Transmissions*, 1971. Independent Television Authority (ITA) and British Broadcasting Corporation (BBC).