Processors Aim At Desktop Video

Special Hardware Brought To Bear on Compression Problem

by Curtis P. Feigel

Desktop digital video offers exciting potential for business and consumers. Video-on-demand, niche programming, interactive television, high-rate data service, video telephony, digital ad insertion, and satellite news gathering are just some of the concepts being bandied about. The demands that these video applications place on digital systems, however, are quite high, beyond the capability of most general-purpose processors. To meet these demands, many vendors—including IIT, C-Cube, LSI Logic, and AT&T—have designed special-purpose video processors. Some can only decode and display images, while others can also encode.

These video processors have many features not found in general-purpose CPUs. Most have special function units that handle the unique calculations required by the popular video algorithms. Some of these units operate autonomously, so the chips can execute many operations in parallel and meet the high processing rates required for video. Some chips even include multiple programmable processors to increase the number of calculations per cycle.

In this article, we give an overview of the issues involved in processing video data and the standards that apply in this area. Then, using IIT's Vision Controller Processor (VCP) as an example, we examine the internal workings of a video processor.

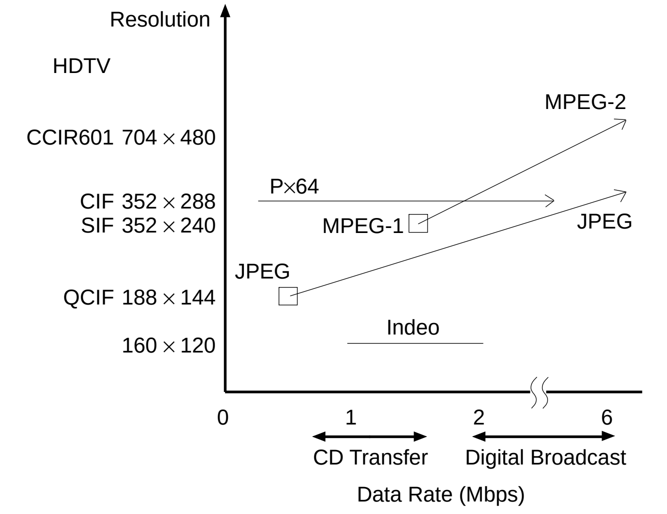

Figure 1. Comparing standards shows that MPEG-2, while providing the highest compression ratio, is specified only for SIF resolution and above. P×64 and JPEG are appropriate in low-data-rate applications. (Source: IIT)

Tight Squeeze Is a Challenge

Bringing digitized multimedia to desktop PCs is challenging in a number of ways, but mostly because its dense stream of moving images and audio requires high bandwidth. Reproducing digitized full-motion video on a 640 ×480 VGA screen requires a data rate in the hundreds of megabits per second (640 ×480 pixels/frame ×16 bits/pixel ×30 frames/s = 140 Mbps). It's possible to transmit at this rate across a computer's internal bus, but physical limitations are tougher to overcome outside the box.

Video's particular demands also make it more difficult than other applications to implement over LANs. While relatively insensitive to infrequent timing errors, the video data stream constantly requires bandwidth. Common Ethernet and Token-Ring LANs offer numerous opportunities for delays, especially with many network nodes competing for bandwidth. Transmission delays affect the apparent quality of image motion and can cause loss of synchronization between image and sound.

Newer buses and networking schemes, such as QuickRing *(see **071603.PDF**)* and the Fast Ethernet proposals, provide dedicated transport for *isochronous* transmissions, guaranteeing a certain amount of bandwidth for time-sensitive data. Some of the first desktop video-conferencing systems avoid the complications of LAN by employing multiple dedicated ISDN or Switched-56 digital telephone lines, even for short-haul transmissions.

Another obstacle is the storage requirement imposed by digital video—a minute of uncompressed digital video would occupy about a gigabyte. Hard-drive prices may have dropped dramatically in the last year, but we're unlikely to see affordable hundred-gigabyte drives any time soon. Even CD-ROMs, considered ample for most other digital applications, are small and slow media for digital video.

To solve both the bandwidth and storage issues requires making heroic efforts to compress the video material. Full-motion video recording, storage, and playback are impractical on a personal computer without compression ratios of 100:1 or more (see Figure 1). Standard lossless compression techniques simply remove redundancies in the image data but generally offer ratios of only 4:1 or less. Beyond this, digital video must rely on algorithms that selectively remove image information. The concept of *lossy* compression may

Video and Compression Glossary

Artifact—A portion of an image that has visibly degraded as a side effect of the compression process.

CCIR 601—A broadcast-television resolution standard that specifies an active image area of 480 lines of 704 pixels for NTSC and 576 lines of 704 pixels for PAL.

CCITT—A French acronym for the International Consultative Committee for Telegraphy and Telephony.

CIF—The Common Intermediate Format that specifies a color image using 288 lines of 352 pixels at 30 fps (frames per second).

CIF 240—A variant of CIF that specifies a color image using 240 lines of 352 pixels at 30 fps.

Codec—Software or hardware that both compresses ("co") and decompresses ("dec").

Genlock—The act of aligning the frames of an image being displayed or recorded with external events.

H.221—A CCITT standard for video telephony that specifies framing and video/audio multiplexing protocols.

H.261—A CCITT standard for video telephony that specifies a video codec using intraframe and interframe compression, is based on discrete cosine transformation (DCT), and transmits over ISDN lines.

H.320—A CCITT standard under development that combines P×64 with high-resolution still-frame and VCR-like capabilities (such as control of fast forward and reverse). A revised version includes multipoint capability for video conferencing.

Huffman Encoding—A lossless data-compression algorithm that uses a statistical method to convert data symbols into variable-length bit strings. The most frequently occurring data symbols are converted into the shortest bit strings.

Interframe—Occurring among several image frames.

Intraframe—Occurring within a single image frame.

ISO—The International Standards Organization.

Isochronous—Having a known delay and delivery rate.

Indeo—A compression standard developed by Intel that uses vector quantization for intraframe compression. It is intended primarily to allow general-purpose processors to perform real-time decompression of video, but it can take advantage of special-purpose hardware, if present.

JPEG—A compression standard for still images of continuous-tone color, named for the Joint Photographic Experts Group of the ISO. The standard allows compression ratios of up to 100:1, but ratios of about 20:1 are generally preferred to maintain image quality.

MPEG-1—A compression standard originally intended for delivery of video on compact disc, named for the Motion Picture Experts Group of the ISO. Its low transfer rate of 1.4 Mbps led to the specification of the SIF image format. Decoded images are expanded to fill a TV screen, resulting in image quality similar to VHS tape. It allows compression ratios up to 50:1.

MPEG-2—A standard for compression of CCIR 601 images into a 4–8 Mbps data stream, consisting of three elements: video, audio, and system. It outlines compression techniques and defines a syntax for compressed video. It also defines a syntax for compressed audio and a mechanism for combining and synchronizing the video and audio elements into a single data stream. The algorithm is specified to decode MPEG-1 images, is extensible to HDTV resolution, and allows compression ratios up to 160:1.

NTSC—A standard for broadcast video used predominantly in the US and named for the National Television Standards Committee. The standard specifies two interlaced image fields making up a 525-line frame displayed at 30 fps.

PAL—The Phase-Alternate Line standard for broadcast video used predominantly in Europe. The standard specifies two interlaced image fields making up a 625-line frame displayed at 25 fps.

P×64—A committee of the CCITT that has promulgated several standards covering facets of video encompassing telephony, conferencing, processing, compression, and transmission, including the H.261 and H.221 standards.

QCIF—The Quarter Common Intermediate Format that specifies a color image using 144 lines of 176 pixels at 30 fps.

SIF—The Source Input Format resolution that specifies an image using 240 lines of 352 pixels at 30 fps for NTSC, 288 lines of 352 pixels for PAL.

seem foreign, but compressing images at these rates inevitably causes a loss of information resulting in *artifacts* in the image.

Real-time encoding of broadcast-quality video at these compression levels requires enormous processing power. C-Cube, for example, rates its eight-processor, $10,000 CLM4600 at 2,000 MOPS (million operations per second) for motion estimation while IIT rates its forthcoming $350 VCP device at 4,000 MOPS.

Compression Requires Multiple Steps

To squeeze video images to an acceptable size, modern standards specify a series of different compression methods (see Figure 2). While lossy, these techniques take advantage of certain characteristics of human perception to minimize visual degradation.

For example, the eye is less sensitive to change in color than to change in intensity, so loss of color information is less noticeable. The typical technique is to separate an image's color and intensity information by translating from the common RGB color space to YUV format (where Y is an intensity value, and U and V are color values). Later, the color information can be compressed at a higher, more lossy rate with only minor

© 1994 MicroDesign Resources

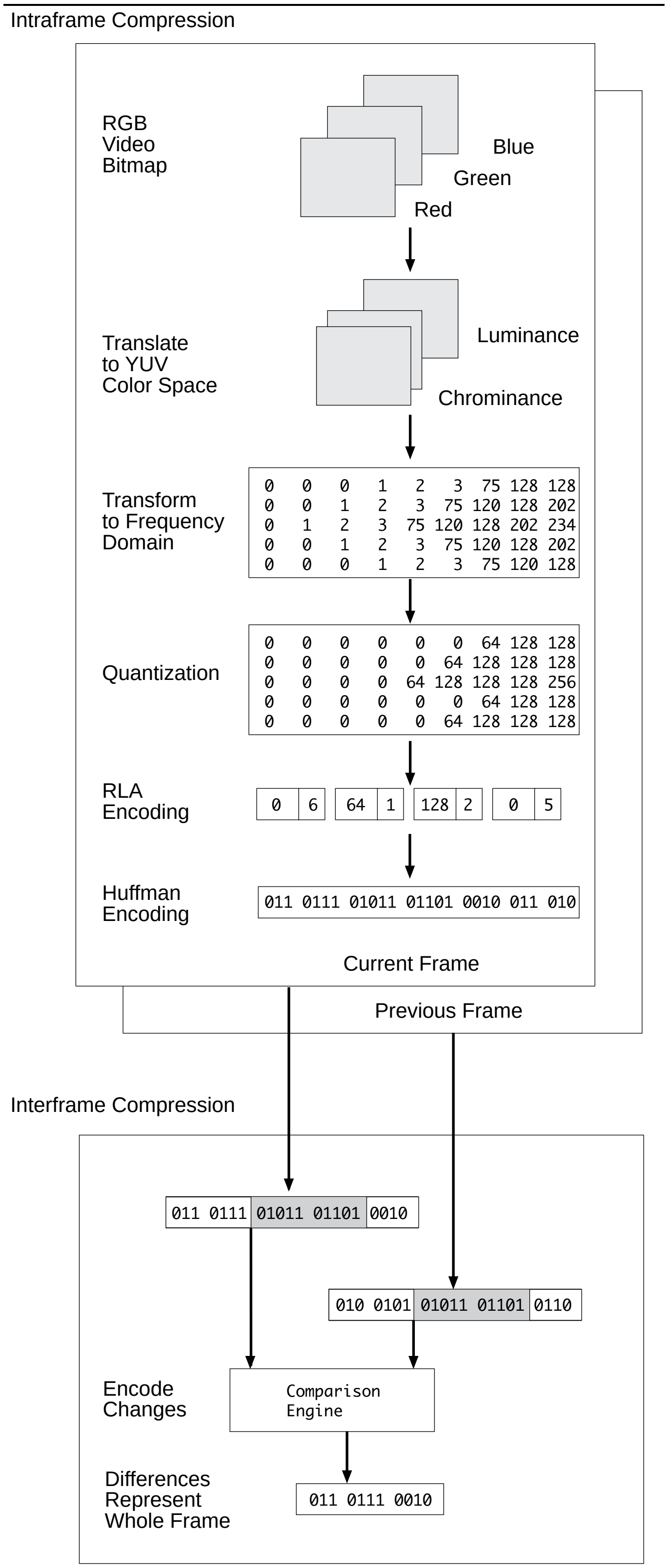

Figure 2. The compression algorithm is actually a series of techniques applied as a pipeline. Intraframe compression removes redundancies and detail within a single frame while interframe compression removes redundancies between adjacent frames. (The process is simplified here for clarity.)

perceived degradation in the final image.

Likewise, fine detail is less obvious to the eye than coarse detail. To take advantage of this characteristic, the compression scheme transforms the image data from the spatial domain (a bit map) to the frequency domain. Finer details are represented as higher frequencies that, again, can be compressed at a higher rate. There are currently four basic methods that reach the level of compression required: DCT (discrete cosine transform), VQ (vector quantization), fractal compression, and DWT (discrete wavelet transform). DCT-based algorithms are the most prevalent in open standards. These transforms operate on blocks of YUV pixels to produce arrays of coefficients, mathematically isolating the frequency components of the intensity and color values.

Once the image is translated and transformed, it can be compressed via quantization. This reduces the number of values used to represent the image, making the data set more coarse. Quantization can be applied in different degrees to the color and intensity information and also differently depending on the frequency component of the coefficients. This is where loss of image information occurs, but the technique is flexible enough that the losses can be adjusted to whatever level is visually acceptable and to meet the needed compression ratio.

The result of the quantization process is an array of numbers in which many adjacent values are the same. This situation lends itself to further compression via run-length encoding. Even higher compression ratios are achieved when this part of the algorithm is applied across the array in zig-zag fashion. The final products of this stage of the algorithm are RLA (run-length/amplitude) tokens, which contain a run-length count followed by the amplitude of the next value.

Further compression can be applied using Huffman coding. This converts RLA tokens into variable-length bit-strings with the most frequently occurring tokens assigned the shortest string.

These techniques are all types of *intraframe* compression; that is, they apply to a single image frame. Even greater compression is achieved with two *interframe* techniques that take advantage of the redundancy and regularity in a sequence of images. The first, called predictive coding, compares each image with the previous one and encodes the differences. The second, called interpolative coding, represents intervening images as the difference between two non-adjacent images in the sequence. The MPEG standard calls such compressed images “P frames” (for predictive compression) and “B frames” (for bidirectional compression).

Real-Time Compression Needs Hardware

Video compression can be done in software. This is an inexpensive solution for the end user, and the capability can be integrated into existing systems and applications. But performance is limited because the volume of data heavily loads the system, taxing both the compression operation and the system's ability to run complex video applications. General-purpose architectures are not optimized for compression applications.

Because compression is much more compute-intensive than decompression, most software compression schemes work best with some form of hardware assistance. To capture and compress in real time using Intel's Indeo algorithm, for example, requires that company's $700 Smart Video Recorder.

There are several types of specialized hardware that can greatly enhance the compression speed:

- A processor for matrix operations to transform from the spatial domain to the frequency domain
- A processor and lookup table for quantization
- A Huffman-coding engine
- Comparators and vector generators for interframe compression

Almost all current and emerging standards involve processes that could benefit from such hardware.

IIT's Vision Control Processor

One example of such specialized hardware is IIT's VCP *(see **0715MSB.PDF**)*. Billed as a single-chip codec and multimedia communications processor, IIT's VCP is a superset of the company's previous VP (video signal processor) and VC (video controller) chip set, as shown in Figure 3. This device does not use function blocks that are hardwired and dedicated to one stage of the compression algorithm. Rather, the VCP uses a programmable algorithm-executing processor called the VP+ in combination with the VC, a general-purpose RISC processor, and a few function blocks for hardware assist where sensible.

The fact that it uses programmable processors means the chip can implement any of the standard compression algorithms—including JPEG, MPEG-1, MPEG-2, H.261, and Indeo—as well as future and custom algorithms. The VCP also includes numerous interfaces and function blocks for pre- and post-processing of video to simplify the device's integration into a complete system.

The VC is a RISC core based on the MIPS X design licensed from Stanford. Created by a team led by John Hennessy and Mark Horowitz, the MIPS X has its roots in the original Stanford MIPS architecture from which commercial MIPS processors also grew.

The VC can be programmed via a C compiler and supervises the codec and other on-chip functions. This pipelined processor uses 32-bit instructions, but has neither data nor instruction caches. Instead, the chip has a small amount of on-chip SRAM for frequently used data. The VC's program, data, and stack memory are provided by external SRAM; on-board ROM holds basic setup and initialization code. The 32-bit-wide SRAM interface provides byte enables and four address decodes to directly control external SRAM and memory-mapped I/O devices.

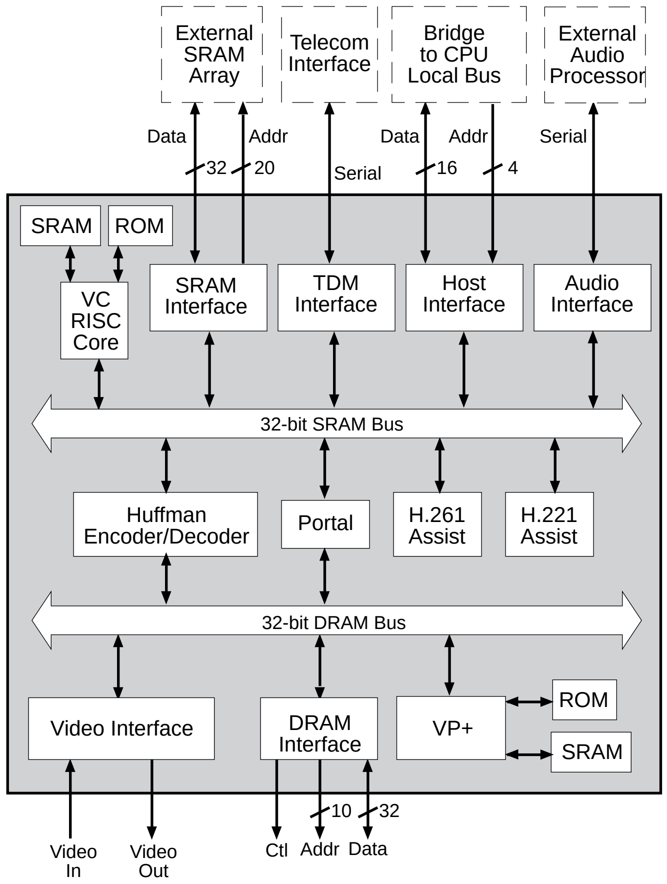

Figure 3. IIT's VCP incorporates two processors. The VP+ contains special parallel hardware for processing high-bandwidth pixel data. The VC is a RISC-type core based on the MIPS X architecture and is used as a supervisor. Special function blocks and interfaces simplify the device's integration into a system.

Compressing and decompressing images is done by the VP+ engine (see Figure 4). It performs all stages of the algorithm, processing images by converting pixels into RLA tokens, or vice versa. Whereas the VC operates on word, half-word, and byte data, the VP+ can directly address individual bits. It is microprogrammable and can operate in SIMD-fashion (single instruction, multiple data), performing the same operation in parallel on six 8 ×8-pixel blocks simultaneously.

Within the VP+, a small 16-bit processor runs microcode that sets up various function blocks for the processing algorithm. Microcode for the standard algorithms is included in on-chip ROM. The core can perform scalar operations while image and I/O operations are executing. A small block of SRAM is provided for application-specific microcode.

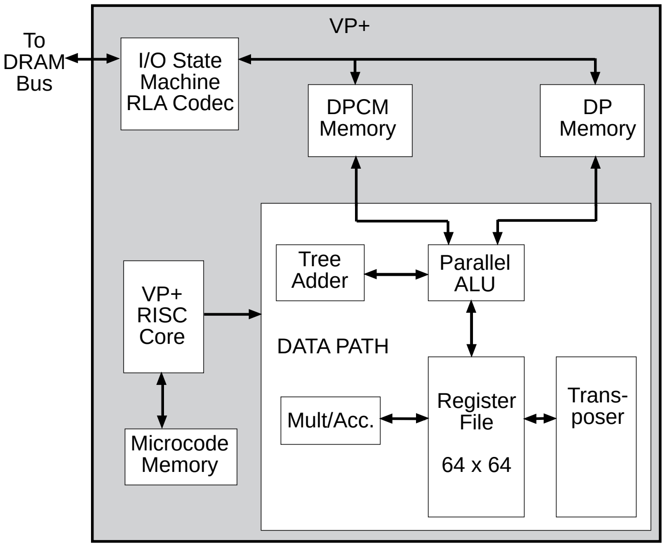

Figure 4. This detail of the VCP shows the internal arrangement of the VP+, which completely executes the compression/ decompression algorithm. The processor has SIMD capability and can operate on multiple blocks of pixels at a time.

Image data enters the VP+ through an I/O state machine. This is a DMA slave port that controls access to the DP and DPCM memory blocks. Image data must be written to one of these memory blocks to be processed by the VP+; results are stored here and can be read by external circuitry. These memory blocks are triple-ported to allow overlapping computation and I/O. The I/O state machine can automatically perform RLA encoding and decoding when reading and writing the DP memory.

Operations on pixel data are performed within the data path block. Its 64 ×64-bit register file has multiple ports to handle up to two reads and one write per cycle. The transposer allows 16-bit quantities to be transposed within each 64-bit word. The parallel ALU can perform sixteen 8-bit or eight 16-bit operations at a time. The ALU is capable of completing a sequence of subtract, compare, absolute value, and sum operations in a single cycle, and can even add and subtract the same quantities simultaneously (which is useful in the DCT algorithm). IIT estimates that performing these operations on a general-purpose CPU could take up to 64 cycles.

To perform motion estimation for interframe compression, the VP+'s tree adder and shifter can compare two 8 ×8 blocks of pixels every four clocks. The VP+ can use the chip's DRAM interface, directly controlling the memory as a frame buffer for storing uncompressed and reference images. The multiply/accumulate block is similar to that found in DSP chips and contains four 16 ×16-bit multipliers with a 24-bit accumulator. This block is used during DCT, quantization, and mean square error calculations.

Improvements Encompass New Standards

IIT put earlier versions of the VC and VP+ processors in separate chips. The newly combined system is a superset of the previous one, but compatibility has taken a back seat to improvements. Besides running at 66 MHz (double the old clock speed), the VCP now provides special functions and video pre- and post-processing features such as its Huffman encoder and decoder, H.261 BCH assist, and H.221 assist.

The Huffman encoder and decoder includes tables for MPEG, JPEG, and P×64 formats. The encoding tables are preprogrammed, but the decoding tables can be redefined by software. This section can process data at up to 15 Mbps.

The BCH assist block is treated as a memory-mapped peripheral. Compressed image data written to this block will be frame-aligned and will have cyclic-coded error detection and correction applied per the H.261 specification. Other parts of the standard must be implemented by the VCP's processors.

The H.221 assist block is designed to implement that standard's specifications for multiplexing audio and video. Because data frames may not fall on word or byte boundaries, a situation that the VC processor is not suited to handle, this block contains hardware that is table-driven to manipulate data at the bit level.

To simplify its integration into a system, the VCP's video interface now handles uncompressed images in a variety of video formats. Its input and output buses each incorporate an image-format controller that can handle interlaced scanning and can be *genlocked* to an external video source. The interface has separate buffers for incoming and outgoing images and each bus may be synchronized separately.

The VCP also incorporates video pre- and post-processing features. Multitap decimation and interpolation filters allow conversion between CCIR 601 resolution and SIF, CIF, and QCIF resolutions. The VCP's temporal filter can improve picture quality in applications with low-speed data streams. Other functions include color conversion, image scaling, and graphics overlay.

A compressed image's bit stream may be transmitted or received over the host-bus interface, or in serial via the new TDM (time-division multiplexed) interface. The TDM bus implements a number of high-speed serial protocols and can interface to a variety of communications chips for LAN and WAN connections via ISDN, T1, PABX, and other protocols. An image's bit stream can be transmitted across multiple channels to achieve the necessary bandwidth; under control of the VC processor, the SRAM DMA controller and H.221 hardware can compensate for delays that may occur on individual channels.

While the VCP contains no internal provision for processing audio, this may be done with an external DSP. Audio can be passed directly to the VCP via a bidirectional serial bus for inclusion into the compressed video data stream.

Video Comes to Your Doorstep (Almost)

Computer and communications technologies are just now becoming powerful enough to move digital video out of the realm of mere curiosities. Compared with its bandwidth demands, desktop video is still limited by shortcomings in system memory, processing power, and I/O, not to mention storage. Concepts such as desktop video conferencing require several technological innovations before they become appealing, and prices must continue to fall.

One major benefit of hardware-based compression is realized when the stream of video data is separated from the general-purpose data path. Using a general-purpose computer's local bus to interconnect various video processors and peripherals is limiting. Supporting the bandwidth required by video is costly, but unless its speed is high, using the same bus to carry video data and to link the main processor with its memory will cause conflicts. Encapsulating the video processing into a single unit, as in the VCP, achieves this separation, eliminating large numbers of PCB traces and reducing the physical size of the device.

Interest in digital video is apparent, but only recently has there been widespread concern about a common digital video standard. With the work of the MPEG-2 committee nearing completion, many companies using proprietary compression schemes are moving to support this standard.

The market for high-quality, real-time compression is small—only originators of broadcast video need this capability. These applications demand performance that is achievable only through special-purpose video processors that deliver high performance with little regard for cost.

High-volume video applications include decode-only products, such as digital televisions, and products—including video telephones—that require compression as well, although not with the high image quality of broadcast video. Because these consumer products must deliver adequate performance at a low cost, it is likely that they will also use specialized video chips instead of costly general-purpose processors.

Thus, there will be a growing market for video processors as the world moves to digital video standards. Current chips have not reached the price level to make digital video pervasive, but they demonstrate the potential of this technology. Ongoing advances in IC manufacturing will ultimately overwhelm these price barriers, bringing digital video to your door. ♦