

Using FPGAs for HDTV design

DESIGNERS HAVE BEGUN TO APPLY DYNAMIC IMAGE-PROCESSING ALGORITHMS IN FPGAs TO CONVERT AND MAP DIGITAL-VIDEO SIGNALS ONTO DISPLAY PANELS. MULTIPLE VIDEO-PROCESSING TECHNIQUES AND BUILDING BLOCKS EXIST TO HANDLE, PROCESS, AND DISPLAY CLEAN, SMOOTH PICTURES ON FLAT-PANEL HDTVs.

Today's rapid proliferation of large-screen HDTVs (high-definition televisions) requires the use of highly complex video-processing algorithms to achieve high resolution, which in turn necessitates faster data rates to address the many artifacts that viewers would not typically notice on smaller screens. To overcome these challenges, designers are now beginning to apply dynamic image-processing algorithms in FPGAs to convert and map digital-video signals onto display panels. Designers are using many approaches for handling, processing, and displaying clean, smooth images on flat-panel HDTVs.

DESIGN CHALLENGES

The consumer market for HDTVs presents a significant challenge for product designers. To compete for the best-selling brand names, designers must consider both cost and video-performance quality. Even many ASSPs (application-specific standard products) target video-display applications; simply using only a standard IC makes it difficult to differentiate one product from another. Using available low-cost FPGAs to design a proprietary video-enhancement algorithm improves the probability of product success. FPGAs are also more effective than ASICs in shortening the design cycle. Most design groups today design an HDTV system either with a stand-alone FPGA or by coupling an FPGA with an ASSP as a co-processor. Most FPGAs include hard-coded DSP blocks and internal memories, which form the basic elements for video and image processing.

BASIC BUILDING BLOCKS

From an architectural point of view, an HDTV usually receives a digital-TV signal from either a terrestrial or a cable/satellite set-top box

(Figure 1). The front-end tuner demodulates the RF signal to baseband for video decoding. Typically, the decoder is either MPEG-2 or MPEG-4/H.264. HDTVs must also receive other video-signal sources, such as external-component and composite signals. The internal microcontroller multiplexes and selects all of these video-signal streams for further video processing and enhancement before the video processor maps the video pixels onto the flat-panel display.

One of the components of a video processor is a deinterlacer, which converts interlaced video to progressive video using a variety of algorithms. Television standards, such as PAL (phase-alternation line) and NTSC (National Television System Committee), commonly use interlaced video, but LCDs require progressive video, which is often more useful for subsequent image-processing functions. The basic algorithm for progressive video is the bob-or-weave algorithm. Weave deinterlacing creates an output frame by filling all of the missing lines in the current input field with lines from the previous in-

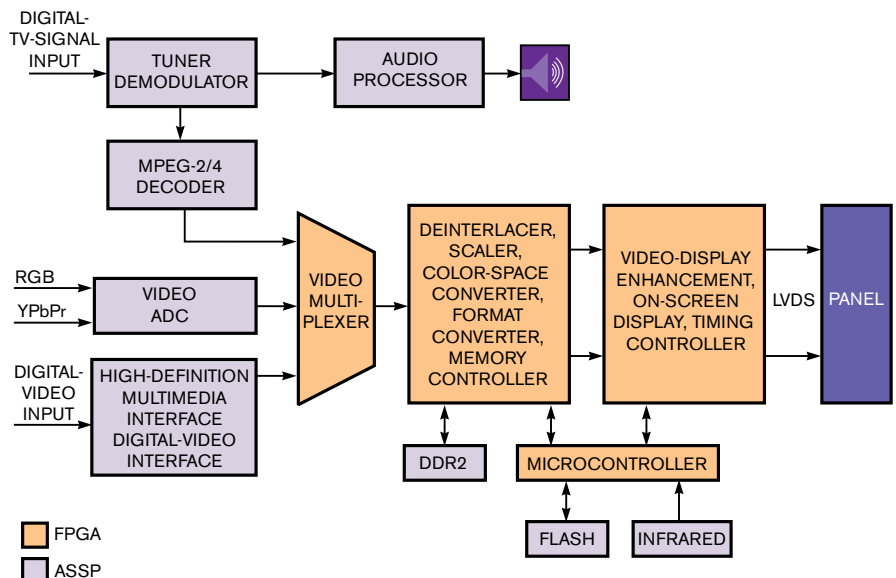


Figure 1 An HDTV usually receives a digital-TV signal from either a terrestrial or a cable/satellite set-top box.

put field. This option provides adequate results for the still parts of an image but unpleasant artifacts for the motion parts. Weave deinterlacing also requires frame-buffer storage in either on- or off-chip memory, depending on the device, to allow the weaving together of lines from different fields. For this reason, the weave deinterlacer requires a built-in triple-buffering function. Bob deinterlacing vertically scales up input fields by a factor of two. The two types of scaling for bob deinterlacing are scan-line duplication and scan-line interpolation. Scan-line duplication simply scales by twice repeating each scan line in Input Field 0 to create the output frame and discarding Input Field 1. Scan-line interpolation re-creates the lines missing from Input Field 0 by performing an unweighted mean of the lines above and below them and discarding Input Field 1. At the bottom of Field 0, only one line is available. The interpolator merely replicates this line as in scan-line duplication.

Another enhancement algorithm, motion-adaptive deinterlacing, involves a progressive-scan-video sequence consisting of a 3-D array of data in the horizontal, vertical, and temporal dimensions (Figure 2). In the absence of motion, the deinterlacer identically reconstructs the interlaced sequences. In a fast-motion-video sequence, the motion detector uses the deinterlacer to separate stationary or moving areas within the same video frame. The motion-detection output uses this deinterlacer as a trigger to select either a spatial interpolation or a temporal interpolation to generate a progressive video.

Another building block in video processing is a scaler for converting video signals between arbitrary resolutions. The processor usually uses the scaler to convert low-resolution interlaced SD (standard-definition) signals to the high-resolution, noninterlaced signals that HDTV uses. A scaler's basic function is similar to that of a line doubler with extra video-signal processing and optimizing. The system designer can also use various tools and IP (intellectual property) to implement a scaler.

The human eye is less sensitive to color than to luminance. You can improve video-transmission bandwidth by storing more data affecting luminance than data affecting color. A viewer notices no perceptible loss when viewing a smaller TV panel sampling the color detail at a lower rate. Video systems achieve this feature through the use of color-difference components. They divide the signal into a Y' (luma) component and two color-difference components (chroma) in a video-camera. Larger panels require further chroma resampling to enhance the video quality of the display.

Chroma resampling deviates from color science in that the luma and the chroma components form naturally as a weighted sum of gamma-corrected R'G'B' (red/green/blue)

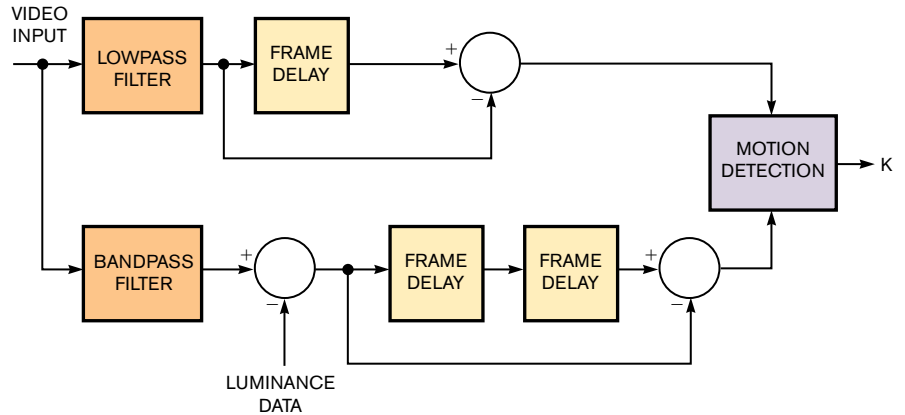


Figure 2 A basic video motion detector uses bandpass and lowpass filters.

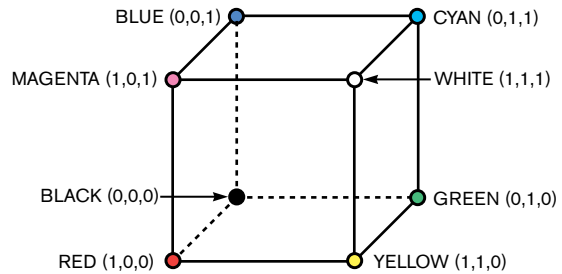


Figure 3 A color-space converter provides a flexible and efficient means of converting image data from one color space to another and provides a method for precisely specifying the display of color using a 3-D-coordinate system.

components instead of linear-RGB components. As a result, luminance and color detail are not independent of one another; some “bleeding” of luminance and color information occurs between the luma and the chroma components. The error is greatest for highly saturated colors and can be somewhat noticeable between the magenta and the green bars of a color-bars test pattern. This approximation allows a designer to easily implement color resampling. Using this fact, video transmitted in the Y’CbCr (luminance/chroma-blue/chroma-red) color space often subsamples the color components, Cb and Cr, to save data bandwidth. The video-sampling format is Y’CbCr of 4:4:4, 4:2:2, or 4:2:0. The 4:2:2 and 4:2:0 formats are subsamples of the 4:4:4 format. How the chroma resampler performs this subsampling depends on the type of application, whether in the professional-broadcasting arena or in the consumer market. These sampling formats are parts of the MPEG-1, MPEG-2, MPEG-4, and other standards. You might wonder why standards don’t use the full 4:4:4 sampling. The following video-resolution detail provides the answer: 720×486 resolution=349,920 pixels/frame; 349,920 pixels×10 bits/sample×3 samples/pixel=10,497,600 bits/frame; 10,497,600 bits/frame×29.97 frames/sec=314,613,072 bps; and 314,613,072 bps×3600 sec≈141.58 Gbytes/hour. For a

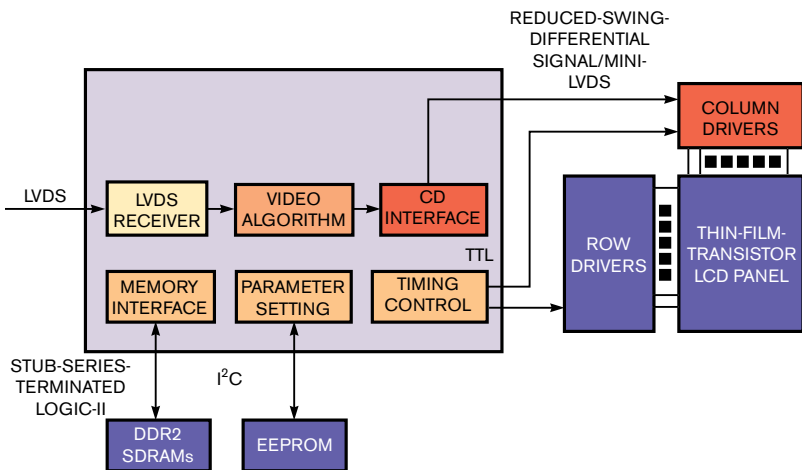


Figure 4 A timing controller is a key element of an HDTV-LCD module.

1920×1080-pixel HDTV, that figure would reach 840 Gbytes/hour. Therefore, using 4:4:4 sampling for a consumer HDTV is neither feasible nor cost-effective. A typical consumer HDTV processes a 4:2:0 video format.

A nonlinear relationship always exists between a pixel value and its displayed intensity on an HDTV monitor. This nonlinear relationship is roughly a power function: $L = V_{\text{GAMMA}}^{\gamma}$, where L is the displayed intensity and V_{GAMMA} is the pixel value with gamma correction, a nonlinear operation that designers use to code and decode luminance values in a video-image-processing system. The gamma-correction function allows designers to modify video streams for physical properties in display devices. For example, CRTs and LCD monitors display a brightness that has a nonlinear response to the voltage of a video signal. To account for this response, designers program the gamma-correction function with a look-up table that models the nonlinear function and then use that function to transform the video data and produce the best image on the display.

FIR AND MEDIAN FILTERING

One of the most common video-enhancement blocks is the FIR (finite-impulse-response) filter. A FIR filter multiplies and sums a sequence of received-video-data impulses, creating a

2-D convolution process. A 2-D FIR filter can perform 2-D convolution using matrices of 3×3, 5×5, or 7×7 coefficients. A 2-D FIR filter's key provides sharpening, smoothing, and edge detection of a video image. By designing the proper coefficients and applying the correct matrix, you can produce a crystal-clear video output. However, the electrical system can introduce video noise into a video stream during transmission in any channel. A median filter provides a simple and effective noise-filtering process. The median value of all the pixels in a population—that is, a selected neighborhood block—determines each video pixel. The median value of a population is that value in which one-half of the population has smaller values than the median and the other half has larger values than the median value.

You can combine an OSD (on-screen display) and an alpha-blending mixer to provide a method to layer streams of video onto a display screen with text and other graphics. One of the most common applications is the EPG (electronic-programming guide) for an HDTV display. A typical valid range of alpha coefficients is [0, 1], where 1 represents full translucence and 0 represents full opaqueness. For N -bit RGBA (red/green/blue-alpha) values, the range is [0, 2 N -1]. This range interprets (2 N -1) as 1 and all other values as the alpha value

divided by 2N. For example, 8-bit alpha value ($255 \Rightarrow 1$), ($254 \Rightarrow 254 \div 256$), ($253 \Rightarrow 253 \div 256$), and so on. Implementing OSD in an FPGA is fairly straightforward. You use a local microcontroller either within or outside the FPGA to generate the graphics or text characters, and you use a video-line buffer to mix the generated text.

A color-space converter provides a flexible and efficient means of converting image data from one color space to another and provides a method for precisely specifying the display of color using a 3-D-coordinate system (Figure 3). Different color spaces are appropriate for different display devices, such as R'G'B' for computer monitors or Y'CbCr for HDTV. Color-space conversion is often necessary when transferring data between devices that use different color-space models. For example, to transfer a TV image to a computer monitor, you must convert the image from the Y'CbCr color space to the R'G'B' color space. Conversely, transferring an image from a computer display to a TV may require a transformation from the R'G'B' color space to Y'CbCr. Video displays for SDTV and HDTV require different conversions, such as to or from the Y'IQ (perceived-luminance/color-luminance-information) model for NTSC systems or the Y'UV (luminance-bandwidth-chrominance) color model for PAL systems.

You achieve conversions between color spaces by providing an array of nine constant coefficients and three constant summands that relate the color spaces. Given the constant coefficients A0, A1, A2, B0, B1, B2, C0, C1, and C2 and the constant summands S0, S1, and S2, you calculate the output values on channels 0, 1, and 2 (d_{OUT0} , d_{OUT1} , and d_{OUT2}): $d_{OUT0} = (A0 \times d_{IN0}) + (B0 \times d_{IN1}) + (C0 \times d_{IN2}) + S0$; $d_{OUT1} = (A1 \times d_{IN0}) + (B1 \times d_{IN1}) + (C1 \times d_{IN2}) + S1$; and $d_{OUT2} = (A2 \times d_{IN0}) + (B2 \times d_{IN1}) + (C2 \times d_{IN2}) + S2$.

CONTROLLER AND INTERFACE

A timing controller is a key element of an HDTV-LCD module (Figure 4). It is the last building block you imple-

MORE AT EDN.COM ▶

+ Go to www.edn.com/ms4263 and click on Feedback Loop to post a comment on this article.

ment after the video processor has performed all video processing and signal enhancement and before interfacing to the display panel. The controller must correctly map all video color pixels on the display with precise timing without

incurring video-quality degradation. As in the past, designers currently implement these controllers using fully customized ASIC devices or ASSPs. Neither of these types of devices provides the full scalability of an FPGA, which allows you to scale a design for a small display to a large display. FPGAs also provide a built-in LVDS (low-voltage-differential-signal)- or RSDS (reduced-swing-differential-signal)-interface format that the timing controller typically uses to directly drive the HDTV-display panel. You can program FPGA-based timing controllers to support nonstandard resolutions and different LCD-panel configurations from various LCD manufacturers.

Designers have over the last two years made significant improvements in the quality of image processing for HDTV flat-panel displays. In addition to a variety of standard available ASSPs, designers are now using low-cost FPGAs with built-in features, such as DSP blocks, memories, microcontrollers, and differential interfaces, to leverage the rapid design cycle. FPGAs can serve as stand-alone units or as complements to an ASSP-system design to provide superior video enhancement for the digital-TV-display market. **EDN**

AUTHOR'S BIOGRAPHY



Tam Do is a senior technical-marketing manager at Altera Corp, where he is responsible for defining and developing technical systems and marketing plans for the professional-broadcast and consumer-electronics arenas. Do holds a bachelor's degree in electrical engineering from University of Nevada—Reno, and his personal interests include ham radio. You can reach him at tamdo@altera.com or at his ham-radio call sign, N7EQR.

Do holds a bachelor's degree in electrical engineering from University of Nevada—Reno, and his personal interests include ham radio. You can reach him at tamdo@altera.com or at his ham-radio call sign, N7EQR.