

IMPLEMENTING AN MPEG-2-TO-H.264 TRANSCODER ON THE DM642

You can realize a real-time transcoder capable of transcoding an MPEG-2 D-2 bit stream to an H.264 bit stream on a 600-MHz TMS320DM642 processor.

By Sriram Sethuraman et al.



The technology behind H.264, a joint video coding standard of the ITU-T and the ISO, significantly reduces the bit rate compared with earlier standards-based technologies, like MPEG-2 and MPEG-4. As a result, it extends the envelope of video transmission and storage applications.

Since digital television content today is transmitted predominantly in the MPEG-2 format, such new applications will need to transcode data from format to H.264. The increased compression efficiency of H.264, however, comes at a significant computational cost. In addition, the transcoder poses significant challenges in terms of memory bandwidth and code size, not to mention the challenges in achieving good transcoded video quality.

You can develop a transcoder on Texas Instruments' TMS320DM642™ digital media processor, which is one of the very few programmable digital signal processors available that has sufficient processing power to realize a single-chip solution. By carefully exploiting the algorithm- and instruction-level parallelism inherent in the transcoder; the capabilities of the processor, such as its powerful instruction set and versatile EDMA controller; and such tool chain capabilities as code sectioning and code overlay, you can meet the challenges of high computational complexity with a low-complexity solution.

OVERVIEW OF H.264

H.264 builds on the motion-compensated transform coding paradigm of the earlier video coding standards. Instead of constraining the technology to use building blocks employed in earlier standards (such as 8x8 IDCT), the entire technology has been engineered from the ground up with no requirement for backward compatibility. Some of the salient coding tools that lead to the compression efficiency gain of H.264 are improved spatial intraprediction, enhanced temporal prediction (through quarter-sample motion compensation, variable-block size motion compensation, multi-hypothesis motion compensation, and weighted prediction tools), efficient context-based entropy coding (through variable-length coding or binary arithmetic coding tools), and in-loop content and coding-mode adaptive deblocking filtering.

H.264 specifies three profiles, based on the potential applications. The Baseline Profile targets low-end devices for stored playback and streaming over reliable channels. The Main Profile targets entertainment-quality

applications, like broadcasting, DVD recording and playback, high-definition television, and digital cinema. The Extended Profile targets error-resilient streaming for robust video-on-demand delivery over wired and wireless networks.

Experiments have shown that H.264 reduces the bit rate 35 to 50 percent compared with MPEG-4 Advanced Simple Profile coding and 40 to 65 percent compared with MPEG-2, at a similar visual quality. As Figure 1 shows, the H.264 Baseline Profile cuts the peak signal-to-noise ratio (PSNR) by more than 50 percent

compared over a wide range of bit rates compared with MPEG-4 Simple Profile.

MPEG-2-TO-H.264 TRANSCODING

The compression efficiency advantage of H.264 is spurring its adoption in personal video recorders (PVRs), to increase the hours of stored content, and in network edge servers that stream content over various last-mile solutions. In the PVR application, since the same device records and plays back, there are no interoperability issues. Hence this application is very likely to be one of the first to take off.

A related application is home gateways that transmit multimedia content wirelessly from a central home server to multiple TVs within the home. In such a case, the higher compression efficiency translates into good entertainment-quality video even though the network is wireless.

Similarly, new video-on-demand service providers that have a lower-bit-rate last-mile pipe but are still looking to provide entertainment-quality content will find the new standard attractive.

Because MPEG-2 is the most prevalent format for transmitting digital television content, virtually all these applications require transcoding from that format.

The substantial difference between MPEG-2 and H.264 necessitates a solution in which complete decoding of the MPEG-2 bit stream content is followed by H.264 re-encoding, rather than compressed-domain or minimal-drift-based low-complexity transcoding. Since we are discussing enter-

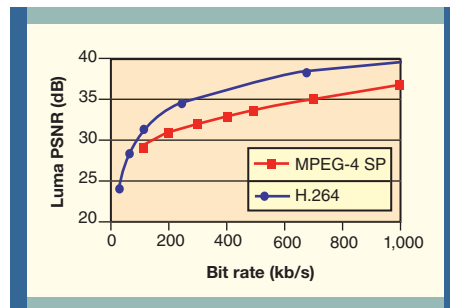


Figure 1: The H.264 video coding standard reduces the bit rate by half at a given quality (as measured using PSNR) over the entire bit rate range compared with MPEG-4. The H.264 encoder used I and P slices, a single reference frame, segmented motion estimation, an in-loop filter, and context-adaptive binary arithmetic coding. (The graph was generated using the Joint Model H.264 reference encoder and the MoMuSys VM implementation of an MPEG-4 SP encoder.)

tainment video, the H.264 profile considered here is the Main Profile.

Complete re-encoding of the decoded MPEG-2 content without using any information from MPEG-2 can require more than an order of magnitude increase in the computational requirements compared with MPEG-2 decoding—in essence, typically requiring multiple high-end programmable processors.

LOW-COMPLEXITY APPROACH

A low-complexity single-processor approach to MPEG-2-to-H.264 transcoding is possible, however (Figure 2). This approach leverages coding information (like motion vectors, coding modes, quantizers, and bit allocation profile) from the MPEG-2 bit stream to reduce the complexity of re-encoding in the H.264 format. It employs H.264 coding tools, such as spatial intraprediction, segmented motion compensation on macroblocks, context-adaptive variable-length coding/ context-adaptive binary arithmetic coding (CAVLC/ CABAC), and quarter-sample motion compensation to reduce the bit rate of the MPEG-2 stream by 25 percent without any significant loss in visual quality. Furthermore, adapting the approach to work within a transcoding framework (in which the resolution at the output of the transcoder is lower than at the input) boosts the

stored). In a transcoder, the optional processing module in Figure 2 would correspond to the down-scaling of the MPEG-2 decoded output to the desired output resolution.

As we said, the improved compression efficiency of H.264 comes at the expense of increased computational complexity. For instance, the tools mentioned above adapt spatially to the video content,

implying that they require a good deal of contextual information and conditional processing.

In addition, the standard imposes certain control flow restrictions. For instance, in an intra-4x4 macroblock (MB), each 4x4 sub-block must be completely reconstructed before the next 4x4 sub-block can be intrapredicted. Another example

is the macroblock pair (two vertically adjacent MBs), which are coded one after the other in macroblock adaptive field/frame coding (MB-AFF), rather than the strict raster-scan coding in MPEG-2. The decoding and re-encoding result in high memory bandwidth requirements and increased code size.

Given the challenges, the fact that multistandard support is becoming almost unavoidable on PVRs, and the inflexibility and long lead times associated with ASIC designs, programmable processors are the preferred target for implementing the transcoder.

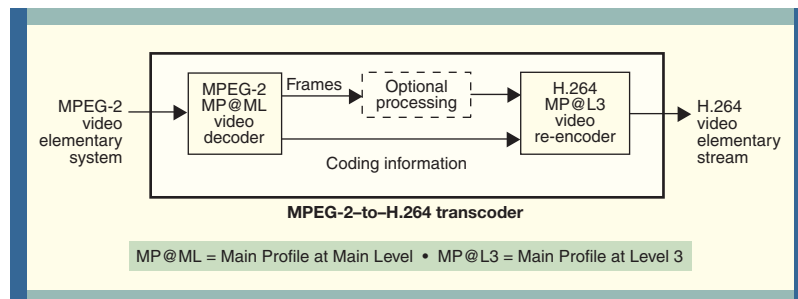


Figure 2: A low-complexity single-processor MPEG-2-to-H.264 transcoder uses coding information from the MPEG-2 bit stream to reduce the complexity of re-encoding in the H.264 format and employs H.264 coding tools to reduce the bit rate. Using transcoding lowers the resolution at the transcoder's output, thus increasing the compression. An optional processing module can be down-scaling to reduce the coding artifacts before transcoding.

The approach employs H.264 coding tools to reduce the bit rate of the MPEG-2 stream by 25 percent without any significant loss in visual quality.

compression advantage further. For instance, a 4-Mb/s MPEG-2 bit stream with D-1 resolution can be transcaled to a 1-Mb/s H.264 bit stream with half D-1 resolution (in a way similar to the long play and extended play modes in a VCR, which trade off quality to increase the hours of content that can be

Before discussing the design steps, let's take a look at the DM642 digital media processor. It consists of a TMS320C64x™ generation DSP integrated with a 64-bit-wide external memory interface, a two-level memory hierarchy (16 kB each of L1 program and data caches and 256 kB of L2 memory that can

be configured as internal SRAM and L2 cache), an enhanced DMA controller with 64 programmable channels, and digital video and audio I/O ports. The C64x™ DSP employs a very long instruction word (VLIW) architecture with eight functional units (across two datapaths) that each can execute an instruction in the same clock cycle, and access from one datapath to the other is supported. Each datapath has an L (logical), D (data), M (multiplier), and S (shift) unit and thirty-two 32-bit registers. Some of these functional units further support SIMD instructions that are customized for packed-byte or half-word operations, which are very useful in handling video data. The D units are capable of loading or storing two 64-bit aligned data or one 64-bit unaligned data word in one clock cycle, capabilities that, during motion compensation, are likewise very useful. All instructions can be conditionally executed using five conditional registers. The processor includes an Ethernet MAC and a PCI interface. It is capable of running at clock speeds of up to 720 MHz,

and the C64x DSP has been tested at up to 1 GHz.

DESIGN METHODOLOGY

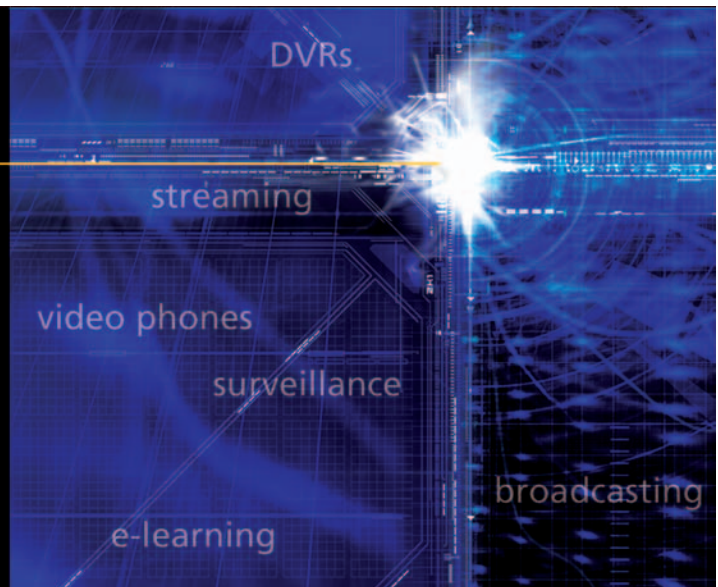
The major design steps in implementing the transcoder on the DM642 media processor are:

1. Decide on the granularity of interaction between the decoder and the re-encoder.
2. Estimate the MCPS and code size requirements of the various modules after optimizing the key leaf modules.
3. Group the various processing stages suitably to design a pipeline. The pipeline determines the code and data buffer requirements within the I-SRAM.
4. Based on the processing requirements dictated by the pipeline, develop a strategy to transfer data and, if necessary, code using DMA.
5. Implement the pipeline design and adjust the code

Home of H.264

CODECS FOR DM64x

Highest Picture Quality
Lowest Bit Rates



W&W
Communications

information@wwcoms.com
Phone: 408.481.0264

www.wwcoms.com

and buffer placements in I-SRAM.

The granularity of interaction determines the rate of code switching between the MPEG-2 decoder code and the H.264 re-encoder code and influences I-cache misses and the amount of data buffering needed. For instance, a simple transcoder design might decode one macroblock and immediately re-encode it. In that case, most of the data is reused, but the code is used only once. Possible granularities range from groups of macroblocks, to a frame, to a group of frames. Depending on the design, the interaction can be strictly synchronous (implying that one granule of decoding is followed by its re-encoding) or asynchronous. Five factors govern this choice: the use of MB-AFF coding in H.264, the need to display the decoded frame while transcoding is in progress, the level of I-cache misses, the EDMA bandwidth requirements to fetch the data again, and the internal memory requirements to buffer the data.

OPTIMIZING KEY LEAF MODULES

The C64x DSP VLIW architecture allows you to exploit both algorithm-level parallelism (ALP) existing in the transcoder and instruction-level parallelism (ILP), the parallelism that exists in a code segment in which dissimilar operations are required that have no interdependencies.

Typically, all video processing algorithms have a high degree of ALP, since the same operation tends to be performed on multiple pixels. The DM642 DSP's single-instruction, multiple-data (SIMD) instructions, such as DOTPU4, SUBABS4, AVGU4, and SPACKU4, come in very handy at performing multiple operations on multiple bytes in a single instruction. The two datapaths also allow two parallel lines of execution. Since all functional units support arithmetic and logical operations, such opera-

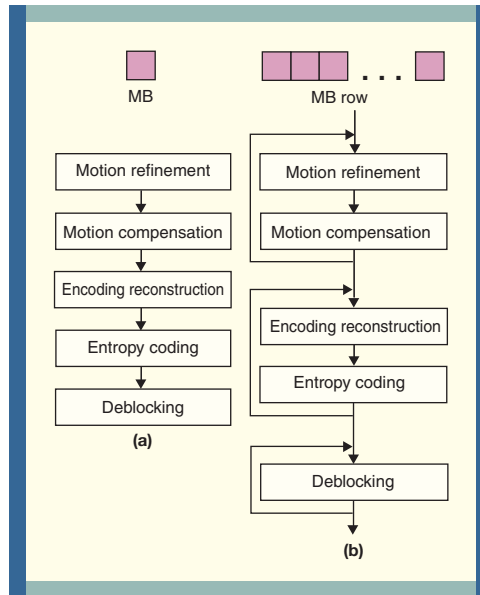


Figure 3: When all stages of the re-encoder are together at a macroblock level (a), the data memory requirements are lower, but the code size for the various processing stages exceeds the L1-P cache size by several factors and hence I-cache misses will be very high. By grouping the various stages together at the level of a row of macroblocks (b), you can tailor the code for each module on which you loop for the L1-P cache size. However, the output of each stage must be buffered, and the L1-D cache misses increase with the additional data buffering requirements

tions can be performed in parallel on multiple pixels in the same clock cycle.

A high degree of ILP is common typically in complicated control code in which multiple expressions need to be evaluated to finally make a decision. ILP can be used to fill any remaining functional units in an execution packet after taking advantage of ALP.

Here are a few more pointers on optimizing key leaf modules on the DM642:

- Analyze whether a module is I/O or process-limited. That serves to identify the best possible cycle count for key loops.
- Identify appropriate SIMD instructions to increase the amount of work performed by each functional unit.
- Balance the load between datapaths A and B cleanly to reduce cross-path accesses.
- Explore unrolling possibilities in the case of shallow loops. However, you need to carefully balance code size, register pressure, and the speedup achieved.

- Maximize the utilization by packing as many slots as possible. Explore other ways of implementing a desired computation using the available functional units.
- Minimize stalls due to branches and delay slots of the chosen instructions.
- Use conditional execution when possible to replace branches.
- Ensure that data accesses do not result in memory bank stalls.

You need to group the various processing stages such that three criteria are met: I-cache thrashing is

minimized, suitable pipelining can be established to minimize the cycles spent by the DSP waiting for data by using the EDMA capabilities, and the data buffering made necessary by the grouping can be accommodated within the available I-SRAM.

PIPELINE DESIGN

The processing stages in the transcoder can be classified as decoding and re-encoding. The decoder processing stages are syntax parsing, motion compensation, and texture decoding. The re-encoder processing stages are motion refinement, encoding loop, entropy coding, and deblocking. In the case of a transcaler, the downscaling becomes another processing stage. Furthermore, additional processing stages related to format conversion may be needed if a video display is required.

- For instances when EDMA completes asynchronously and can only be done in an interrupt service routine, optimize the ISR handler to reduce cycle overhead for EDMA setup.
- Balance the transfer load across the four priority queues in the EDMA controller according to how critical the data is.
- Take care of cache coherency issues by ensuring that the same region of SDRAM isn't simultaneously accessed via L2 cache and EDMA.

Aligning the data buffers properly and reusing scratch buffers is essential to minimize L1-D misses. Similarly, since the code size of the decoder and re-encoder can be extremely large (even larger

As the clock rates of the DM642 DSP keep increasing, the transcoder can move toward achieving full D-I transcoding on a single chip.

Figure 3 shows the impact of the grouping of processing stages on instruction and data cache performance. When all stages of the re-encoder are grouped together at a macroblock level (Figure 3a), the total code size is very high and the I-cache thrashes heavily. Grouping the various stages together at the level of a row of macroblocks (Figure 3b) alleviates I-cache thrashing; however, the intermediate outputs between the grouped stages need to be buffered, thereby increasing the internal memory requirements and L1 D-cache misses. You should consider reusing scratch memory space in I-SRAM to alleviate that situation.

DMA CONSIDERATIONS

Once you've designed the pipeline to satisfy the three criteria, the various data transfers from and to external memory become apparent. These become candidates for DMA transfer. While scheduling EDMA transfers:

- Use QDMA for short bursts of transfer, since the setup overhead for QDMA is low.
- Consider chaining EDMA transfers to automate the triggering of the next transfer without the intervention of the DSP.

than the available I-SRAM), it's essential to consider strategies for minimizing L1-P misses. Proper code sectioning is critical to L1-P performance. The Code Composer Studio™ integrated development environment tool chain allows a different load-time address and run-time address for each function. This capability can be used to overlay mutually exclusive pieces of code in I-SRAM at run time. Take care, though, to ensure that the appropriate code is in I-SRAM before the corresponding function call is made.

Using the design methodology described here, you can realize a real-time transcaler capable of transcoding an MPEG-2 bit stream with D-1 resolution to an H.264 bit stream with half D-1 resolution on a 600-MHz DM642 media processor. As the clock rates of the DM642 DSP keep increasing, the transcoder can move toward achieving full D-1 transcoding on a single chip. ♦

Sriram Sethuraman (sriram.sethuraman@ittiam.com) is the technologist—senior member of technical staff at Ittiam Systems (Pvt.) Ltd. in Bangalore, India. The other authors are *Arvind Raman*, senior engineer; *Kismat Singh*, senior engineer; *Manisha Agrawal Mohan*, engineer; *Neelakanth Shigihalli*, senior engineer; and *B. S. Supreeth*, engineer.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		Applications	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265