7.2 A Highly Parallel and Scalable CABAC Decoder for Next Generation Video Coding

Vivienne Sze, Anantha P. Chandrakasan

Massachusetts Institute of Technology, Cambridge, MA

Future video decoders will need to support high resolutions such as Quad Full HD (QFHD, 4096×2160) and fast frame rates (e.g. 120fps). Many of these decoders will also reside in portable devices. The next-generation standard called High-Efficiency Video Coding (HEVC), which is being developed as a successor to H.264/AVC, not only seeks to improve the coding efficiency but also to account for implementation complexity and leverage parallelism to meet future power and performance demands [1]. Parallel processing increases the throughput for higher performance, which can be traded-off for lower power with voltage scaling. This paper presents a silicon prototype for a pre-standard algorithm developed for HEVC ("H.265") called Massively Parallel CABAC (MP-CABAC) that addresses a key video decoder bottleneck. The test chip has over an order-of-magnitude higher throughput than state-of-the-art H.264/AVC CABAC engines [2-4], while maintaining high coding efficiency. Architecture and joint algorithm-architecture optimizations, which modify the MP-CABAC algorithm, are used to reduce critical path delay and memory size.

Context-based Adaptive Binary Arithmetic Coding (CABAC) is a well known bottleneck in existing H.264/AVC decoders. Although CABAC provides high coding efficiency, its tight feedback loops (Fig. 7.2.1) make it difficult to parallelize and limit the overall decoder throughput. The feedback loops are tied to the binary symbols (bins); thus, the throughput and performance of the CABAC engine are measured in bins/cycle and bins/second, respectively. Speculative computation is often used to increase the throughput at the cost of increased power consumption [2-4]. Unlike the rest of the video decoder which can use macroblock-line (wavefront) parallelism, CABAC can only be parallelized across frames [5]; consequently, buffering is required between CABAC and the rest of the decoder which increases external memory bandwidth.

Massively Parallel CABAC (MP-CABAC), previously developed by the authors [6], is currently under consideration for HEVC, and has been adopted into the standard body's JM-KTA working software [7]. It enables parallel processing, while maintaining the high coding efficiency of CABAC, by using a combination of two forms of parallelism shown in Fig. 7.2.1: interleaved entropy slices (IES) and syntax element partitions (SEP). IES enables several slices to be processed in parallel, allowing the entire decoder to achieve wavefront parallel processing without increasing external memory bandwidth [8]. SEP enables different syntax elements (e.g. motion vectors, coefficients, etc.) to be processed in parallel with low area cost [9]. Figure 7.2.1 shows the MP-CABAC data structure, where each frame is composed of several IES, and each IES is composed of five SEP. The MP-CABAC test chip presented in this paper supports up to 16 IES per frame with 80 arithmetic decoders running in parallel.

IES are processed in parallel by several slice engines as shown in Fig. 7.2.2. IES FIFOs are used between slice engines to synchronize IES, which is required due to top block dependencies. The properties of the neighboring blocks (A and B) are used for context selection and are stored in the IES FIFOs and last line buffer. Figure 7.2.2 shows a joint algorithm-architecture optimization in the context selection logic that reduces the last line buffer size by 50%. To enable scalability, the number of slice engines is configurable; a multiplexer connects the output of the last enabled slice engine to the last line buffer. To reduce power, the clocks to the disabled slice engines are turned off using hierarchical clock gating. Over a 9× increase in throughput is achieved with 16 IES per frame using the architecture in Fig. 7.2.2.

SEP are processed in parallel by several arithmetic decoders (AD) within the slice engine as shown in Fig. 7.2.2. Syntax elements are assigned to 5 different partitions based on their workload (i.e. number of bins). The FSM of the context modeler (CM) and de-binarizer (DB) is divided into smaller FSMs for each SEP. The register-based context memory is also divided into smaller memories for each SEP. Thus, the context memory and the FSM are not replicated, which keeps the area cost low. The slice engine contains 5 different partition engines, each with a small FSM, context memory and AD. Dependencies between SEP are managed using SEP FIFOs, allowing SEP of different macroblocks to be processed concurrently. During the stall cycles, the partition engine clock is disabled with hierarchical clock gating to reduce power. Using this slice engine architecture, up to five bins can be decoded in parallel, with an average throughput increase of 2.4×.

Figure 7.2.3 shows the architecture of the partition engine. CM selects the context (i.e. state and most probable symbol (MPS)) based on the syntax element being processed. AD uses this context and encoded bits from the bitstream controller to decode a bin. The bin is fed back to CM to update the context memory and to DB to compute the syntax element. Several techniques are used to reduce critical path delay. First, the engine is pipelined by inserting a register between CM and AD for a 40% reduction. Next, the critical path in AD is reduced using 3 optimizations: (1) Leading-Zero (LZ) detection is done using a look-up table (LUT) in parallel with least-probable symbol interval (rLPS) LUT to speed up renormalization. (2) Early range shifting enables renormalization of rLPS to occur in parallel with the range and offset subtractions. (3) Offset renormalization is moved to the beginning of the next cycle so that it occurs in parallel with the rLPS look up. These architectural optimizations reduce the critical path delay of AD by 11%.

Finally, a joint algorithm-architecture optimization, highlighted as (4) in Fig. 7.2.3 and shown in detail in Fig. 7.2.4, further speeds up AD. Subinterval reordering changes the order of the least and most probable symbol subintervals (rLPS and rMPS). Placing rLPS at the bottom of the range enables the offset comparison to occur in parallel with the subtraction for rMPS, which reduces the critical path by an additional 11% without affecting coding efficiency.

The MP-CABAC test chip, shown in Fig. 7.2.7, was implemented in 65nm CMOS. Figure 7.2.5 shows a summary of the chip features and an example trace and distribution of the number of decoded bins-per-cycle for a video sequence. Figure 7.2.6 shows the trade-off between measured power, performance and coding efficiency across a wide operating range. Scaling the number of IES per frame from 1 to 16 increases the performance range by an order of magnitude, and reduces the minimum energy per bin by 3× to 10.5pJ/bin with less than 5% coding penalty. An average throughput of 24.11bins/cycle is achieved across several HD video sequences, which is 10.6× higher than the state-of-the-art H.264/AVC CABAC implementations [2-4]. The MP-CABAC approach can be combined with techniques used in [2-4] for an additional throughput increase of 1.3 to 2.3×. The MP-CABAC test chip decodes the max H.264/AVC bit-rate (300Mb/s) with an 18MHz clock at 0.7V, consuming 12.3pJ/bin. At 1.0V, it has a performance of 3026Mbins/s for a bit-rate of 2.3Gb/s, enough for real-time QFHD at 186fps, or equivalently 7.8 views of QFHD at 24fps for multiview video coding (MVC). MP-CABAC helps enable a fully parallel video decoder.

Acknowledgements:
Thanks to Texas Instruments for algorithm support, chip fabrication and funding.

References:
[1] "Joint Call for Proposals on Video Compression Technology", ITU-T SG16/Q6, 39th VCEG Meeting: Kyoto, 17-22 Jan. 2010, Doc. VCEG-AM91
[2] T.-D. Chuang, et al., "A 59.5mW Scalable/Multi-View Video Decoder Chip for Quad/3D Full HDTV and Video Streaming Applications," *ISSCC Dig. Tech. Papers*, Feb. 2010.
[3] P. Zhang, et al., "Variable-bin-rate CABAC engine for H.264/AVC high definition real-time decoding," *IEEE Trans. on VLSI Systems*, March 2009.
[4] Y.-C. Yang, et al., "High-Throughput H.264/AVC High-Profile CABAC Decoder for HDTV Applications," *IEEE Trans. CSVT*, Sept. 2009.
[5] S. Nomura et al., "A 9.7mW AAC-Decoding, 620mW H.264 720p 60fps Decoding, 8-Core Media Processor with Embedded Forward-Body-Biasing and Power Gating Circuit in 65nm CMOS Technology," *ISSCC Dig. Tech. Papers*, Feb. 2008.
[6] V. Sze et al., "Massively Parallel CABAC", ITU-T SG16/Q6, 38th VCEG Meeting: London / Geneva, 1-8 July 2009, Doc. VCEG-AL21
[7] KTA Reference Software, 2.7 [Online]. Available: http://iphome.hhi.de/suehring/tml/download/KTA/
[8] D. F. Finchelstein, et al., "Multi-Core Processing and Efficient On-Chip Caching for H.264 and Future Video Decoders," *IEEE Trans. CSVT*, Nov. 2009.
[9] V. Sze, A.P. Chandrakasan, "A High Throughput CABAC Algorithm Using Syntax Element Partitioning," *IEEE Int. Conf. on Image Processing*, Nov. 2009.

 978-1-61284-302-5/11/$26.00 ©2011 IEEE

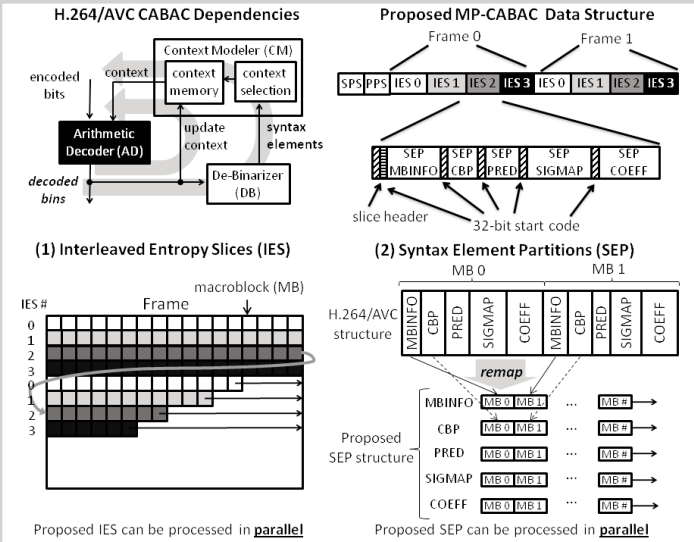

Figure 7.2.1: MP-CABAC data structure: Two forms of parallelism for highly parallel decoding.

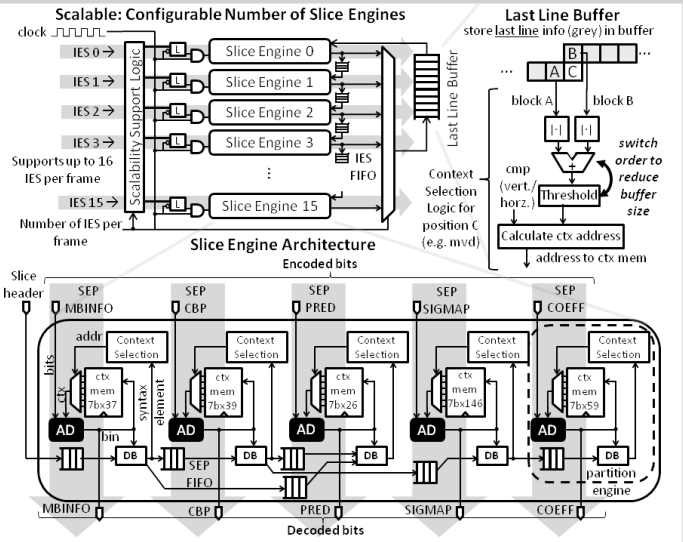

Figure 7.2.2: Scalable architecture of MP-CABAC for parallel processing of IES and SEP. Last line buffer supports up to 4k×2k (QFHD) resolutions.

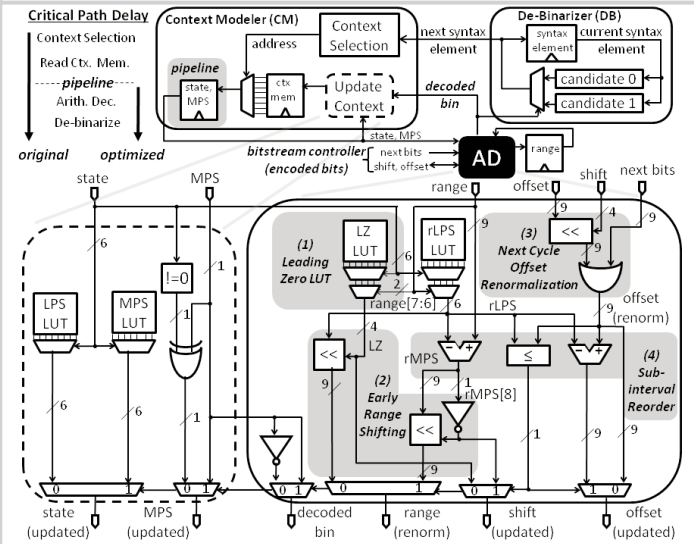

Figure 7.2.3: Partition engine speed up: insert pipeline register after context memory and four AD optimizations (highlighted).

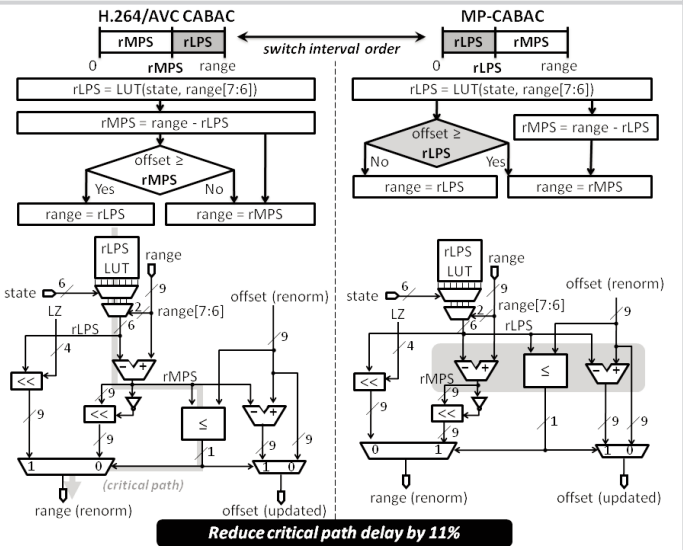

Figure 7.2.4: Subinterval Reordering: Joint algorithm and architecture optimization to reduce critical-path delay in the arithmetic decoder.

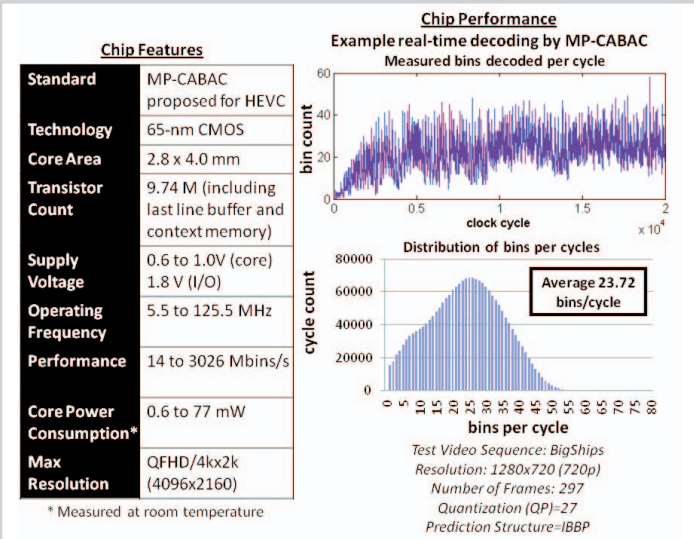

Figure 7.2.5: Features of the MP-CABAC test chip and plot showing the real-time performance for the BigShips video sequence.

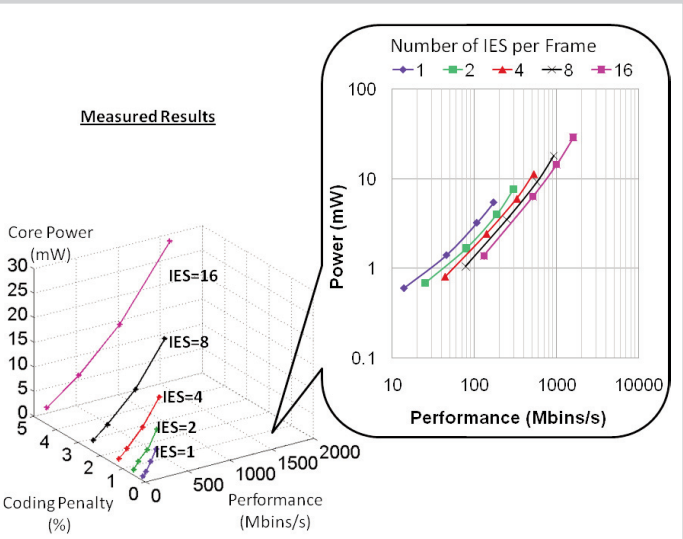

Figure 7.2.6: Measured results showing trade-off between power-performance-coding efficiency of the MP-CABAC test chip.

Figure 7.2.7: Die photo of MP-CABAC test chip. 16 slice engines are highlighted.

978-1-61284-302-5/11/$26.00 ©2011 IEEE