IEEE Asian Solid-State Circuits Conference
November 3-5, 2008 / Fukuoka, Japan

A Low-Power 0.7-V H.264 720p Video Decoder

D.F. Finchelstein, V. Sze, M.E. Sinangil, Y. Koken, A.P. Chandrakasan
Massachusetts Institute of Technology

Abstract

The H.264/AVC video coding standard can deliver high compression efficiency at a cost of large complexity and power. The increasing popularity of video capture and playback on portable devices requires that the energy of the video codec be kept to a minimum. This paper proposes several architecture optimizations such as increased parallelism, multiple voltage/frequency domains, and custom voltage-scalable SRAMs that enable low voltage operation and reduce the power of a high-definition decoder. An H.264/AVC Baseline Level 3.1 decoder ASIC was fabricated in 65-nm CMOS and verified. It operates down to 0.7-V and has a measured power of 1.8mW when decoding a high definition 720p video at 30 frames per second, which is over an order of magnitude lower than previously published results.

1 Introduction

The H.264/AVC video coding standard can deliver high compression efficiency; however, it comes at a cost of high complexity and power which is critical for battery-operated devices. State-of-the-art ASIC H.264 decoders have used techniques such as pipelining, parallelism, and caching to increase throughput and reduce power consumption [1, 2, 3, 4]. In this work, we use additional architecture optimizations to enable low voltage operation, and thus reduce the power consumption of high-definition decoding. We propose that further parallelism, multiple voltage/frequency domains, and custom voltage-scalable SRAMs are necessary in order to maintain the throughput necessary for high definition decoding at low voltages. To validate these ideas, a 1.8 mW H.264/AVC Baseline Level 3.1 video decoder was implemented in 65-nm CMOS and decodes 720p@30fps with supply voltages down to 0.7-V.

Voltage scaling is a well-known technique that reduces energy consumption by a quadratic factor, at the cost of increased circuit delay. This decreased speed is a challenge for real-time applications such as video decoding where on average a new frame must be computed every 33 ms. We will describe a video decoder that allows low voltage operation by reducing the total number of cycles required per frame without increasing the levels of logic on the critical path between registers. This allows for a longer clock period, which leads to a lower supply voltage and lower energy.

2 Decoder Pipeline Architecture

The H.264 codec is described in [5] and builds upon previous MPEG standards. A block diagram of our decoder hardware is shown in Figure 1. At the system level of the decoder, FIFOs of varying depths connect the major processing units: Entropy Decoder (ED), Inverse Transform (IT), Motion Compensation (MC), Spatial Prediction (INTRA), Deblocking Filter (DB), Memory Controller (MEM) and Frame Buffer (FB). The pipelined architecture allows the decoder to process several 4x4 blocks simultaneously, requiring fewer cycles to decode each frame.

The number of cycles required to process each 4x4 block varies for each unit as shown in Table 1. The table describes the pipeline performance for decoding P-frames (temporal prediction). Most of the optimization efforts were focused on P-frame performance, since they occur more frequently than I-frames (spatial prediction) in highly compressed videos. The presence of FIFOs between each unit distributes the pipeline control and allows the units to operate out of lockstep. This adapts for the variable latency of each unit and increases the throughput of the decoder by reducing the number of stalls, as described in [6].

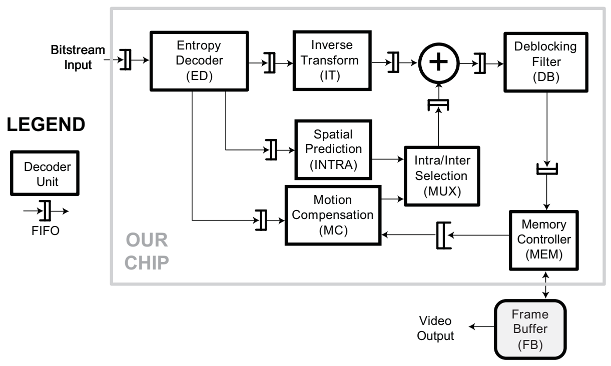


Figure 1: H.264 decoder architecture.

Table 1: Cycles per 4x4 luma block for each unit in P-frame pipeline of Figure 1, assuming no stalling. [ ] is performance after Section 3 optimizations.

Pipeline Unit	*Min Cycles*	*Max Cycles*	*Average Cycles*
ED	0	39	4.3
IT	0	4	1.6
MC	4 [2]	9 [4.5]	4.6 [2.3]
DB	8 [2]	12 [6]	8.9 [2.9]
MEM-read	4	27	13

The luma and chroma components have separate pipelines that share minimal hardware and are mostly decoupled from each other. The two pipelines do have dependencies on each other, which sometimes prevents them from running at the same time. For example, both pipelines use the same ED at the start, since this operation is inherently serial and produces input coefficients and motion vectors to both pipelines.

3 Parallelism

Parallelism can be used within each unit to reduce the number of cycles required to process each 4x4 block. This is particularly applicable to the motion compensation (MC) and deblocking (DB) units, which were found to be key bottlenecks in the system pipeline.

The cost of parallelism is primarily an increase in area. The increase in logic area due to the parallelism in the MC and DB units described in the following sections is about 10%. When compared to the entire decoder area (including on-chip memory) the area overhead is less than 3%.

3.1 Motion Compensation Parallelism

The MC unit uses interpolation and filtering to predict the current 4x4 block from past frames to exploit temporal redundancy. The luma interpolator predicts the current 4x4 block from a 9x9 block of pixels from the previous frame, while the chroma interpolator uses bilinear filters to predict a 2x2 block from a 3x3 block.

To improve the throughput of MC, a second identical interpolator was added in parallel as shown in Figure 2. This parallel structure can double the throughput of the MC stage if during each cycle, both motion vectors are available and two new columns of 9 pixels from the frame buffer are available at the inputs of MC0 and MC1. The chroma interpolator has 4 parallel bilinear filters and can predict a 2x2 block every cycle.

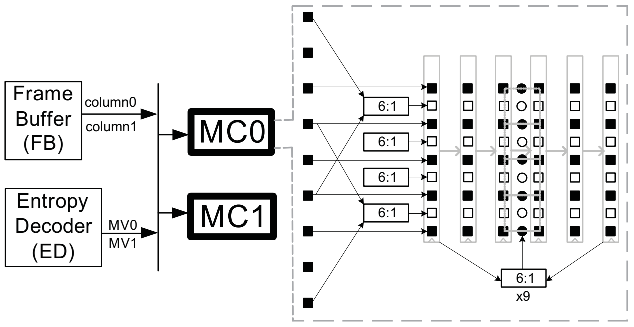


Figure 2: The MC unit is duplicated into MC0 and MC1. The datapath of each MC is 54 8-bit registers, thirteen 6:1 FIR filters, and four 4:1 bilinear filters.

3.2 Deblocking Filter Parallelism

There are 4 luma and 2 chroma deblocking filters running in parallel. The luma architecture is shown in Figure 3. The 4 luma filters share the same boundary strength calculation, as do the 2 chroma filters. The luma and chroma filters can all run at the same time, assuming the input data and configuration parameters are available. A luma macroblock (16 4x4 blocks) has 128 edges that need to be filtered, so with 4 luma filters the macroblock takes 32 clock cycles to complete. Unlike previous implementations [7], filtering on 4x4 blocks begins before the entire macroblock is reconstructed.

A single-port on-chip cache with 4x4x8-bit data-width is used to store pixels from the top and left macroblocks. The edge filtering order of the macroblock was optimized to minimize the number of stalls from read/write conflicts to the cache and accounts for the 4x4 block arrival order and the left-right-top-bottom edge order specified in H.264 [5]. Furthermore, as the on-chip cache has a 2-cycle read latency, a few extra stall cycles are needed for these memory accesses when prefetching them is not possible. Taking into account this overhead, the average number of cycles required by DB for a luma 4x4 block is about 2.9.

Each of the two chroma components of a macroblock has 32 pixel edges to be filtered. Using 2 filters per cycle results in slightly above 32 clock cycles per macroblock. When accounting for stalls, the number of cycles per macroblock is about the same as luma. Overall, the number of cycles required by DB is around 46 cycles per macroblock, which is less than other deblocking filter implementations [7].

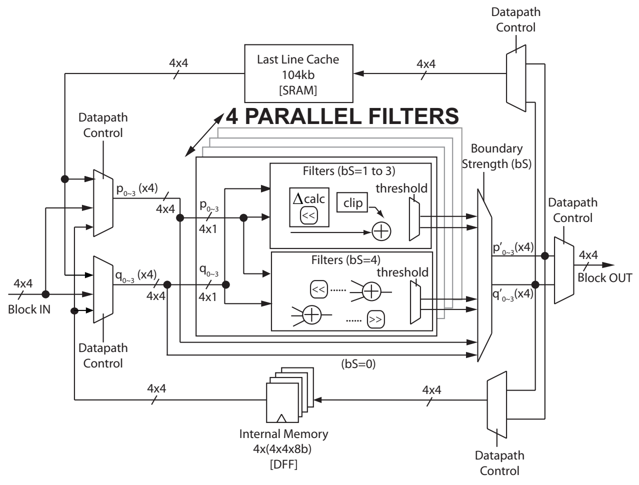


Figure 3: Deblocking filter architecture for luma filtering.

4 Multiple Voltage/Frequency Domains

Several factors limit the amount of parallelism that can be exposed in certain units. For instance, the number of pads on an ASIC is limited, which restricts the degree of memory controller parallelism for off-chip memory accesses. This explains the large cycle count for MEM in Table 1. In a single-frequency design, MEM causes many stalls in the rest of the system. This forces the entire decoder to run at a high frequency in order to maintain the required performance.

We propose partitioning the decoder into 2 domains (memory controller and the core decoder), so that we can independently tailor the frequency and voltage of each domain. The two domains are completely independent, and are separated by asynchronous FIFOs and voltage level-shifters, as shown in Figure 4. Additionally, Table 1 shows that there could be further benefit to also placing the ED unit on a separate domain.

Having two separate domains allows us to dynamically adjust frequencies for the different workloads independently. This is important since workloads for these two domains will vary widely and differently depending on whether the decoder is working on I-frames or P-frames. For example, the MEM domain requires a higher frequency for P-frames versus

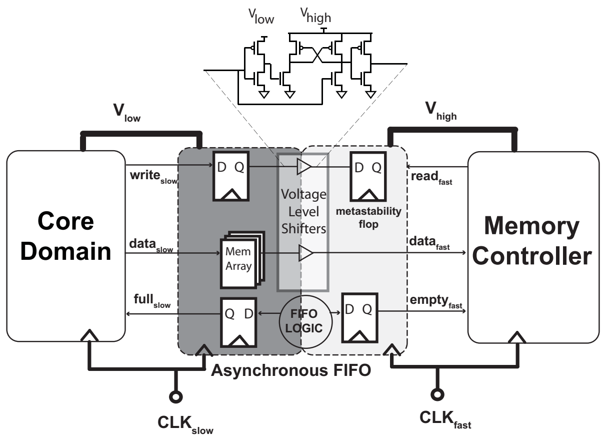


Figure 4: Independent voltage/frequency domains are separated by asynchronous FIFOs and level-converters.

I-frames. Conversely, the core domain requires a higher frequency during I-frames since more residuals are present and they are processed by the ED unit.

5 Memory Optimization

Since the frame buffer (FB) is large and located off-chip, it is important to minimize the off-chip memory bandwidth in order to reduce power. We leveraged two key techniques to reduce this memory bandwidth. During motion compensation, some of the redundant reads are recognized and avoided. This happens when there is an overlap in the vertical or horizontal direction and the neighboring 4x4 blocks (within the same macroblock) have motion vectors with identical integer components[4].

Second, on-chip caches were used to store data that is likely to be reused, as shown in Figure 5. This caching scheme reduces total off-chip memory bandwidth by almost 13% relative to the case where no caches are used. Custom low-voltage SRAMs ([8]) were designed to operate at the desired core voltage and frequency. The SRAM architecture is shown in Figure 6. The 8T bit-cell (BC) is optimized for low-voltage writability and target performance. Row-wise supply node MCHd is actively pulled-down during write accesses to improve degraded write margin at low voltages. The cell performs buffered reads through the 2 extra transistors on the right, in order to avoid read upsets at low voltages. A pseudo-differential sensing scheme compatible with the 8T cell was implemented, as shown on the right side of Figure 6. The latch-based sense-amplifier uses inputs RDBL and a global reference voltage (snsRef) to produce DATAOUT, which is then buffered to the memory interface's output. Compared to a conventional SRAM, this custom design can operate at a much lower supply voltage and trade-off performance for energy.

6 Power Measurements

The H.264/AVC Baseline Level 3.1 decoder, shown in Figure 8 was implemented in 65-nm CMOS and the power was measured when performing real-time decoding of several 720p@30fps video streams. The frame buffer was implemented using 32-bit-wide Cypress ZBT SRAMs. A Virtex II FPGA was used to interface the ASIC to the display. Figure 7 shows a comparison of our ASIC with other decoders, while Table 2 lists the performance of our decoder at 720p. Our 720p decoder also has lower power and frequency relative to D1 of [1]. The power of the I/O pads was not included in the measurement comparisons. The reduction in power over the other reported decoders can be attributed to a combination of using the low-power techniques described in this paper and a more advanced process.

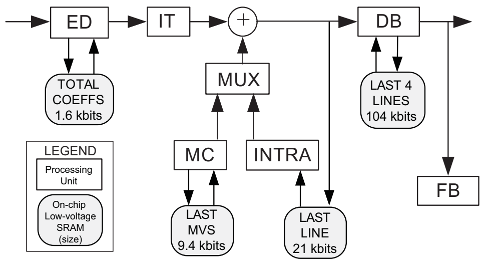


Figure 5: On-chip caches reduce off-chip memory bandwidth.

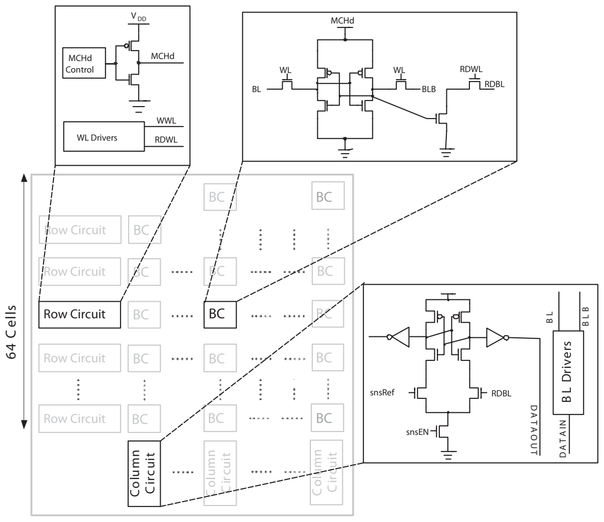


Figure 6: Low-voltage SRAM architecture.

Table 2: Measured performance numbers for 720p@30fps.

Video	mobcal	shields	parkrun
# of Frames	300	300	144
Bitrate (Mbps)	5.4	7.0	26
Off-chip BW (Gbps)	1.2	1.1	1.2
Core Freq (MHz)	14	14	25
Mem Ctrl Freq (MHz)	54	50	50
Core Vdd [V]	0.7	0.7	0.8
Mem Ctrl Vdd [V]	0.85	0.84	0.84
Power (mW)	**1.8**	**1.8**	**3.2**

6.1 Power Breakdown

This section shows the simulated power breakdown during P-frame decoding. The power of P-frames is dominated by

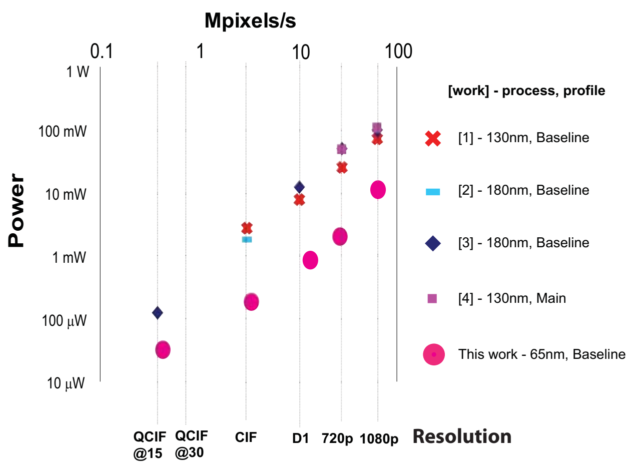
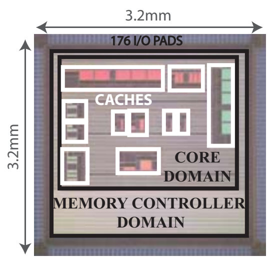


Figure 7: Comparison with other H.264 decoders [1, 2, 3, 4]



DECODER STATISTICS

Total Area :	10mm^2
Technology :	65nm
I/O Pads :	172
Area Utilization :	41%
Standard Cells :	134k
On-chip SRAM :	17kB

Figure 8: Die photo showing the different domains.

MC (42%) and DB (26%), as seen on the left chart of Figure 9. About 75% of the MC power, or 32% of total power, is consumed by the MEM read logic, as illustrated by the pie chart on the right of the same figure. The memory controller is the largest power consumer since it must run at a higher voltage and frequency than the core domain to achieve the large MC read bandwidth (3 luma pixels are read for every decoded pixel).

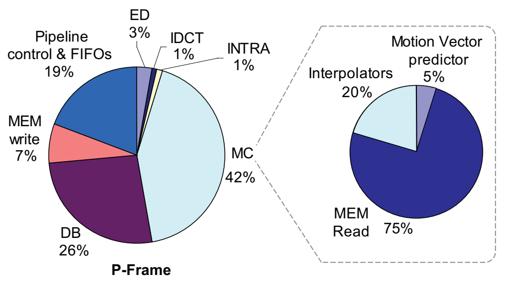


Figure 9: Post-layout simulated ASIC power breakdown during P-frame decoding.

7 Summary

We implemented a full video decoder system that demonstrates high definition decoding at low-voltage operation. Several techniques were leveraged to make this low-power decoder possible. The decoder units were pipelined and isolated by FIFOs to increase concurrency. The MC interpolator and DB filters were replicated for increased performance. The decoder was partitioned into independent voltage/frequency islands to enable lower voltage/frequency operation for some of the processing blocks. Finally, voltage-scalable on-chip caches helped reduce both on-chip and off-chip memory power.

8 Acknowledgements

The authors would like to thank Nokia, Texas Instruments (TI), and NSERC for their support of our research, TI for providing chip fabrication, and the many researchers from Nokia, TI and MIT who have provided valuable feedback.

References

[1] C.-D. Chien, C.-C. Lin, Y.-H. Shih, H.-C. Chen, C.-J. Huang, C.-Y. Yu, C.-L. Chen, C.-H. Cheng, and J.-I. Guo, “A 252kgate/71mW Multi-Standard Multi-Channel Video Decoder for High Definition Video Applications,” in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2007.

[2] S. Na, W. Hwangbo, J. Kim, S. Lee, and C.-M. Kyung, “1.8mW, Hybrid-Pipelined H.264/AVC Decoder For Mobile Devices,” *IEEE Asian Solid State Circuits Conference*, November 2007.

[3] T.-M. Liu, T.-A. Lin, S.-Z. Wang, W.-P. Lee, K.-C. Hou, J.-Y. Yang, and C.-Y. Lee, “A 125uW, Fully Scalable MPEG-2 and H.264/AVC Video Decoder for Mobile Applications,” *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 161–169, January 2007.

[4] C.-C. Lin, J.-W. Chen, H.-C. Chang, Y.-C. Yang, Y.-H. O. Yang, M.-C. Tsai, J.-I. Guo, and J.-S. Wang, “A 160K Gates/4.5 KB SRAM H.264 Video Decoder for HDTV Applications,” *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 170–182, January 2007.

[5] “Recommendation ITU-T H.264: Advanced Video Coding for Generic Audiovisual Services,” ITU-T, Tech. Rep., 2003.

[6] E. Fleming, C.-C. Lin, N. Dave, Arvind, G. Raghavan, and J. Hicks, “H.264 Decoder: A Case Study in Multiple Design Points,” *Proceedings of Formal Methods and Models for Co-Design, (MEMOCODE)*, pp. 165–174, June 2008.

[7] K. Xu and C.-S. Choy, “A Five-Stage Pipeline, 204 Cycles/MB, Single-Port SRAM-Based Deblocking Filter for H.264/AVC,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 3, pp. 363–374, March 2008.

[8] N. Verma and A. Chandrakasan, “A 65nm 8T Sub-Vt SRAM Employing Sense-Amplifier Redundancy,” in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, February 2007.