An Optimized Hardware Design for the VVC Intra Reference Sample Smoothing Filter

Sara Vitória Henssler
Universidade Federal do Pampa (UNIPAMPA)
Bagé, RS, Brasil
sarahenssler.aluno@unipampa.edu.br

Marcel Corrêa
Instituto Federal Sul-rio-grandense (IFSul)
Bagé, RS, Brasil
marcelcorrea@ifsul.edu.br

Abstract

The H.266 Versatile Video Coding (VVC) standard introduces a set of mandatory pre-processing smoothing filters applied to reference samples prior to angular intra prediction. Although these filters lie on the critical path of the prediction process and directly affect latency and hardware complexity, existing VVC hardware designs either omit their implementation or assume pre-filtered samples. This work presents the first dedicated hardware architecture for the VVC reference sample smoothing filter at the decoder side, supporting all block sizes from 8x8 to 64x64. A bit-level reformulation of the 3-tap FIR filter enables an optimized datapath that reduces arithmetic complexity, eliminating unnecessary computations while preserving compliance with the specification. Synthesized on an Intel Cyclone V FPGA, the design occupies only 595 ALMs and 553 registers (1% of the device) and achieves a maximum frequency of 181.79 MHz. Throughput analysis shows that real-time operation is sustained for 8K video at 60 fps with a clock requirement of only 31.10 MHz, enabling underclocking and power reduction. The results demonstrate that the proposed architecture provides a small, high-throughput, and fully compliant solution suitable for integration into a complete VVC decoder.

Keywords

Hardware design, Video coding, VVC

1 Introduction

Today, people increasingly depend on digital video services for work, study, and leisure activities, resulting in a significant rise in video consumption through digital platforms. In this context, efficient video encoding becomes essential since transmitting or storing uncompressed video is impractical. For example, a 10-minute uncompressed recording in Full High Definition (FHD) 1080p resolution (1920×1080 pixels) at 30 frames per second would require approximately 55 gigabytes (GB) of storage, while the same duration recorded in Ultra-High Definition (UHD) 4K resolution (3840×2160 pixels) at 30 frames per second would require roughly 224 GB. Considering that many smartphones, tablets, and laptops commonly offer storage capacities of 128 GB, 256 GB, or 512 GB, a single 10-minute 1080p recording would occupy about 43% of a 128 GB device. More critically, a 10-minute 4K recording would exceed the total capacity of a 128 GB device and consume approximately 87% of a 256 GB device. These figures clearly demonstrate that uncompressed video storage is impractical in consumer electronics and would also impose transmission bandwidth requirements far beyond what is typically available in consumer networks.

To address this challenge, dedicated video compression tools are employed. Video codecs (coder/decoder) perform both encoding and decoding, and may be implemented in either hardware or software. In real-time applications such as digital cameras, live streaming devices, surveillance systems, and video conferencing platforms, encoding must be performed under strict latency and power constraints. Although many commercial systems rely on software-based codecs or hybrid architectures combining software with hardware acceleration, purely software-based solutions may still struggle to meet throughput, energy efficiency, and determinism requirements, particularly in embedded or edge devices with limited computational resources. Dedicated hardware architectures offer advantages such as massive parallelism, predictable latency, and improved energy efficiency. Furthermore, as compression standards evolve toward higher coding efficiency, the algorithmic complexity increases disproportionately, making architectural optimization essential.

The H.266 Versatile Video Coding (VVC) standard [1] was collaboratively developed by the Joint Video Experts Team (JVET) of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). Launched in 2020 as the successor to the H.265 High-Efficiency Video Coding (HEVC) standard [2], VVC introduced substantial improvements in compression efficiency, achieving approximately 50% better compression than HEVC [3] and around 75% better than the widely supported H.264 Advanced Video Coding (AVC) standard [4, 5]. In addition, VVC was designed to support a wide range of digital media applications, including UHD 4K and 8K video, immersive content such as 360-degree video and virtual reality, and high-dynamic-range (HDR) imagery. These characteristics make VVC well suited for modern streaming services, broadcast systems, video conferencing, and emerging interactive media.

As of 2026, VVC remains a relatively new standard, with widespread dedicated hardware support still in the early stages of adoption. Currently, support for VVC decoding is available on selected platforms, such as the Intel Lunar Lake processors with integrated hardware media acceleration [6], the Hantro VC9000D decoder IP core from Verisilicon [7], the D320 decoder IP core from Allegro [8], and MediaTek Pentonic SoCs targeting television applications [9, 10]. At the time of writing, regarding encoding, the Allegro E320 encoder IP core [11] appears to be the only commercially available hardware encoder supporting VVC.

The closest related works are [12, 13], which propose low-power and high-performance hardware architectures for angular intra prediction in VVC. However, these studies focus primarily on the prediction computation itself and do not address the pre-processing

smoothing filters applied to reference samples. In the VVC standard, these smoothing filters constitute a mandatory step of the intra prediction process and are applied prior to prediction across multiple angular modes and block sizes. As they lie on the critical path of intra-frame prediction, their implementation has a direct impact on latency, hardware complexity, and memory access patterns. Despite this relevance, dedicated hardware architectures for these smoothing filters have received little to no attention in the literature, leaving an important gap between the algorithmic specification of VVC and its efficient realization in hardware.

Therefore, this paper presents a dedicated hardware architecture for the pre-processing smoothing filters used in intra-frame prediction, designed to meet the timing constraints of the decoder side. The proposed design was developed in VHDL and synthesized using Quartus Prime, enabling implementation on FPGA platforms and demonstrating its practical feasibility for hardware deployment.

This paper is organized as follows: Section 2 provides background on intra-frame prediction in the VVC standard. Section 3 describes the proposed architecture and design methodology, while Section 4 presents and discusses the experimental results. Finally, Section 5 concludes the paper.

2 Video Coding Basics

Most modern video encoders rely on a common set of signal- and data-processing stages: (i) block partitioning, (ii) intra- and inter-frame prediction, (iii) transform coding (T module), (iv) quantization (Q module), and (v) entropy coding, as illustrated in Figure 1. In addition, encoders include a reconstruction path, commonly referred to as the decoding loop and depicted by the dashed line in Figure 1, which contains the inverse quantization (IQ module) and inverse transform (IT module). This structure ensures that the encoder uses only reference samples that are also available at the decoder, allowing both sides to perform identical prediction processes. An optional in-loop filtering stage may be applied at the end of the reconstruction path to improve the visual quality of the reconstructed frames.

Block partitioning consists of dividing each video frame into smaller units that can be processed independently for compression purposes. This strategy increases adaptability to local spatial and temporal characteristics of the content, enabling more efficient

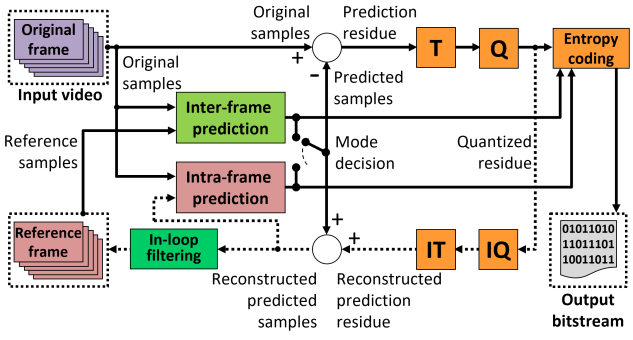

Figure 1: Diagram of a typical hybrid block-based video encoder [14].

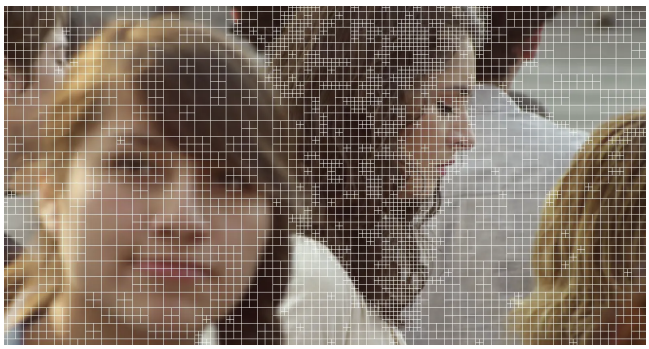

Figure 2: Example of block partitioning of a frame.

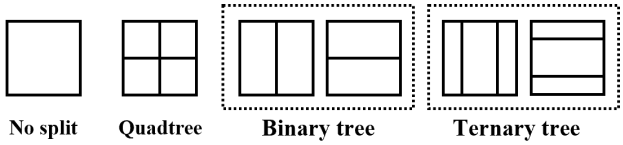

Figure 3: The six partitions allowed in the VVC multi-type tree.

coding decisions. Figure 2 illustrates this concept with a practical example, in which larger blocks are used in more homogeneous regions (e.g., a blurred face), while smaller blocks are employed to better represent areas with rich texture and detail (e.g., a focused face). In the VVC standard, block partitioning is based on Coding Tree Units (CTUs) of up to 128×128 pixels, which can be recursively subdivided into Coding Units (CUs) through a flexible multi-type tree (MTT) framework. This framework supports quadtree, binary tree, and ternary tree splits, allowing finer granularity and improved adaptation to local image characteristics compared to previous standards, as illustrated in Figure 3.

After partitioning, the encoder performs prediction for each block using already available information. In intra-frame prediction —the focus of this work— blocks are predicted from previously reconstructed samples within the same frame, exploiting spatial correlations. In inter-frame prediction, blocks are predicted from reference frames coded earlier in time, exploiting temporal redundancy. In both cases, the prediction signal is subtracted from the original block to generate a residual. The effectiveness of the prediction stage directly influences in reducing this residual and, consequently, the overall compression efficiency of the encoder [15]. In VVC, intra prediction supports an expanded set of directional and non-directional modes, while inter prediction employs advanced motion models and higher-precision motion compensation, significantly improving prediction accuracy.

The residual signal generated after prediction is then processed by a transform stage, whose goal is to concentrate the signal energy into a reduced number of low-frequency coefficients, facilitating subsequent compression. In modern video coding standards, this stage may involve additional partitioning of the residual into transform blocks that better match local signal characteristics [15]. VVC supports a broad set of transform sizes and types, including multiple

variants of the Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST), enabling improved energy compaction across a wide range of content types and block configurations [3].

Next, the transform coefficients are quantized, reducing their numerical precision to remove perceptually less relevant information. The amount of irreversible distortion introduced at this stage is controlled by the Quantization Parameter (QP): higher QP values result in stronger quantization and higher compression ratios, while lower QP values preserve more signal detail [15]. Quantization is applied independently to each transform block, and QP values may vary across blocks through the use of signaling mechanisms that enable fine-grained rate control.

Finally, the quantized transform coefficients, along with associated coding parameters—such as prediction modes, motion information, block partitioning data, and other syntax elements—are passed to the entropy-coding stage. This stage applies lossless compression techniques to reduce statistical redundancy by assigning shorter codes to more probable symbols. In VVC, entropy coding is performed using Context-Adaptive Binary Arithmetic Coding (CABAC), which provides high compression efficiency by adapting symbol probabilities to local coding contexts.

2.1 Intra-picture Prediction

2.1.1 ***Prediction Step.*** The basic processing units produced by the block-partitioning stage are referred to as Coding Units (CUs), which are used for prediction and subsequent coding steps. In VVC intra-picture coding, CUs can assume a wide range of block sizes, enabled by the flexible multi-type tree partitioning structure, allowing square and rectangular blocks with dimensions ranging from small blocks used for detailed textures to larger blocks suitable for smooth regions [3, 14].

To reduce spatial redundancy, VVC defines a comprehensive set of 67 intra-picture prediction modes. These modes generate a prediction for the current CU by filtering previously reconstructed reference samples located along the top and left boundaries of the block, exploiting the strong spatial correlation typically present in natural images [3, 14]. The available modes are classified into non-directional and directional categories.

The non-directional modes include the Planar and DC modes. The Planar mode (Mode 0) is particularly effective in smooth regions, as it generates a gradual surface by combining horizontal and vertical interpolations, thereby reducing visible blocking artifacts. The DC mode (Mode 1) targets flat areas by predicting all samples within the CU using a single constant value computed as the average of the available reference samples.

The remaining 65 modes correspond to Angular prediction modes. These modes predict pixel values by projecting reference samples into the CU along predefined angular directions, effectively modeling edges and textures with arbitrary orientations. The increased number of angular directions allows VVC to more accurately represent complex spatial structures, leading to a substantial reduction in prediction error and, consequently, lower residual energy in textured regions [3].

During encoding, the optimal intra prediction mode is selected through Rate–Distortion Optimization (RDO) [16], aiming to maximize the similarity between the predicted block and the original signal while minimizing the associated bit cost. In this process, the same block is repeatedly evaluated using multiple intra prediction modes, with each candidate generating a distinct prediction, residual, and associated cost. As a result, RDO entails a computationally exhaustive search over a large set of alternatives, significantly increasing encoder complexity. A more accurate prediction directly translates into a residual signal with lower energy, which can be more efficiently processed by the subsequent transform and quantization stages, ultimately improving compression efficiency. Consequently, efficient hardware designs for intra prediction and its associated pre-processing steps are particularly valuable to enable practical, real-time VVC encoding under strict performance and energy constraints.

2.1.2 ***Pre- and Post-processing Tools.*** VVC employs various auxiliary techniques to enhance intra prediction, distinctly categorized into pre- and post-processing stages. The pre-processing, which is the main focus of this work, aims to reduce artifacts in the references samples that are used in the prediction itself, while the post-processing aims to reduce artifacts in the predicted samples themselves.

In the pre-processing stage, the reference samples used for predicting a given block are derived from its spatially neighboring blocks, specifically those located above and to the left, which have already been reconstructed. For a block of width W and height H, the VVC intra prediction process defines two extended reference sample arrays: a top reference array of length $2W$ and a left reference array of length $2H$, in addition to a single shared sample located at the top-left corner. This reference sample configuration is illustrated in Figure 4.

The ITU-T H.266 Recommendation [1] specifies the behavior of the reference sample pre-processing smoothing filter, which is applied identically in both the encoder and the decoder. The reference samples are organized into two arrays, denoted as $Left_{-1\,..\,2H-1}$ and $Top_{-1\,..\,2W-1}$, where the top-left corner sample is shared and represented as both $Left_{-1}$ and Top_{-1} for notational simplicity.

The top-left sample is filtered according to Eq. 1. The samples in the Top array with indices from 0 to $2W-2$ are filtered as defined

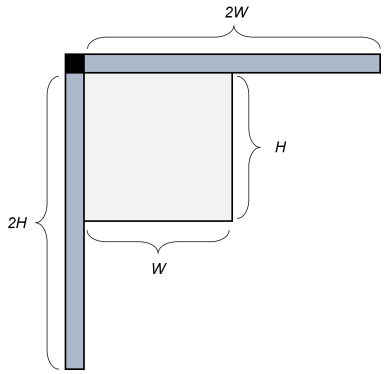

Figure 4: Example of a block to be predicted of size $W \times H$. This block requires a reference array on top of length $2W$, a reference array to the left of length $2H$, and a single top-left sample.

in Eq. 2, while the samples in the $Left$ array with indices from 0 to $2H-2$ are filtered as defined in Eq. 3. The last sample of each array is intentionally left unfiltered, since the smoothing operation requires the availability of two adjacent neighboring samples.

$$Top_{-1} = (Left_0 + 2 \times Top_{-1} + Top_0 + 2) >> 2 \quad (1)$$

$$FilteredTop_i = (Top_{i-1} + 2 \times Top_i + Top_{i+1} + 2) >> 2 \quad (2)$$

$$FilteredLeft_i = (Left_{i-1} + 2 \times Left_i + Left_{i+1} + 2) >> 2 \quad (3)$$

Finally, it is important to note that the reference sample smoothing filter is not always applied. Among the requirements for filtering, the following conditions must be satisfied:

- The block dimensions must be at least 8×8 pixels, as the VVC standard does not apply the smoothing filter to smaller block sizes;
- The required reference samples must be valid and available. Consequently, the filter is not applied when the current block is located at frame boundaries, where neighboring reconstructed samples are unavailable.
- The prediction mode selected for the current block, as well as the prediction modes of the relevant neighboring blocks, must correspond to angular intra prediction modes.

3 Proposed Hardware Design

The proposed top-level architecture performs reference sample smoothing for VVC intra prediction across all supported block sizes, from the smallest 8×8 up to 64×64. The design targets high throughput to enable UHD 8K decoding by exploiting parallelism: for the smallest 8×8 block, which includes 33 reference samples (top and left reference arrays plus the top-left corner sample), the architecture instantiates 33 parallel Reference Sample Smoothing Units (see 3.1), enabling all reference samples of the block to be filtered in a single clock cycle.

To support larger blocks, the architecture reuses the same hardware over multiple cycles. Input reference samples are buffered and delivered to the smoothing units in successive groups, allowing the full reference set of the block to be processed sequentially. The total number of cycles required for a block is proportional to its dimensions, while the parallelism of 33 samples per cycle ensures that no additional cycle is needed for the extra top-left corner sample. This operation is coordinated by a dedicated control unit, which schedules the delivery of samples and manages input and output buffers.

The following subsections describe the individual components of this architecture in detail.

3.1 Reference Sample Smoothing Unit

The reference sample smoothing process employed in VVC intra prediction applies a simple 3-tap finite impulse response (FIR) filter using the sample to be filtered and its two adjacent reference samples. For each output sample, the filtered value is computed as $y = (A + 2B + C + 2)/4$ (see 2.1.2), where A and C denote adjacent reference samples, and B corresponds to the reference sample being smoothed. The addition of the constant 2 prior to the division ensures correct rounding, as defined by the specification [1]. While the filter expression is simple, a direct hardware implementation may incur unnecessary arithmetic cost if the bit-level behavior of the division and rounding operations is not carefully exploited.

3.1.1 ***Naive Hardware Implementation.*** A straightforward hardware implementation of the smoothing filter follows directly from its mathematical definition. In this baseline design, the samples A, B and C are first summed using two adders, with B being left-shifted by one bit to implement the multiplication by two. A third adder is then used to ensure rounding, and the final division by four is implemented via a right shift of two bits. This approach results in a combinational datapath comprising three adders and simple shift operations, with intermediate signals requiring extended bit widths to prevent overflow.

While this implementation is functionally correct and easy to verify, it does not exploit the fact that the least significant bits discarded by the right shift operations do not contribute to the final output value. Consequently, some of the arithmetic operations performed in the baseline design compute information that is never used, leading to redundant hardware and increased cost.

3.1.2 ***Optimized Implementation.*** To reduce arithmetic complexity, an optimized implementation was developed by reformulating the filter as a sequence of two divide-by-two operations:

$$y = \left\lfloor \frac{\left\lfloor \frac{A+C}{2} \right\rfloor + B + 1}{2} \right\rfloor \quad (4)$$

This decomposition is algebraically equivalent to the original expression and preserves exact rounding behavior. Moreover, it allows the computation to be structured as two consecutive stages, each discarding one least significant bit through a right shift, keeping the adders smaller. The circuit was designed as follows:

In the first stage, the sum $A + C$ is computed only for bits 7 down to 1, since the least significant bit is removed by the subsequent division. The discarded least significant bit (LSB) is nevertheless used to generate a carry into the retained portion of the adder. Specifically, the carry-in to bit position 1 is obtained directly as the logical **AND** of the least significant bits of A and C, which exactly matches the carry behavior of binary addition in half adders. As a result, the first stage produces $\lfloor (A + C)/2 \rfloor$ without explicitly computing the full sum.

The second stage combines the result of the first stage with sample B and performs the final rounding. Again, only the bits that survive the subsequent division are added explicitly (7 down to 1). Rounding is implemented by forcing the carry-in of this adder to logic '1'. Under this condition, the LSB carry-out generation is performed by a full-adder logic, but with a fixed carry-in. The carry-out logic of a full-adder is shown below, with the carry-in replaced by '1'. Through boolean algebra, this simplifies to a logical **OR** between the operands' LSBs, allowing further reduction of logic.

$$C_{\text{out}} = xy + 1(x \oplus y) \quad (5)$$

Figure 5 illustrates both implementations while detailing the cost of each. This approach produces a bit-exact implementation of the original filter while requiring only two 7-bit adders.

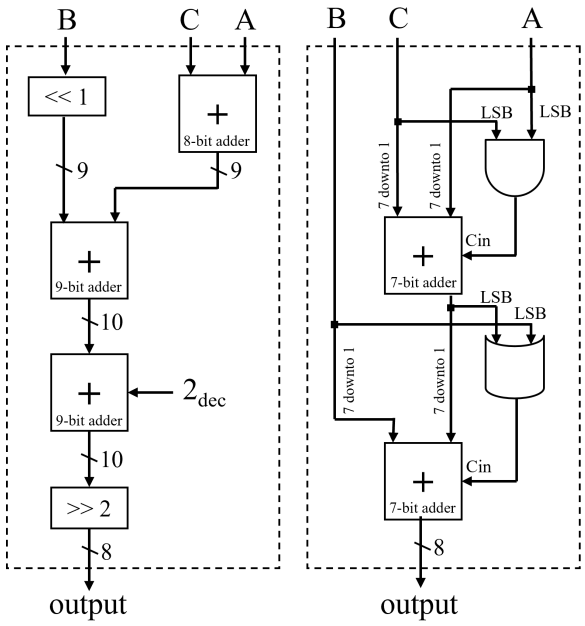

Figure 5: RTL diagram of the Naive and Optimized filter implementation.

3.2 Input and Output Buffers

To feed the combinational filter array efficiently, the architecture incorporates dedicated input and output buffers. The input buffer stores the maximum number of extended reference samples required for a single filtering step, that is, 35 samples. While an 8×8 block only requires filtering 33 of these samples (the extra two being unused for this smallest block), these additional samples are essential for larger blocks: when the 33 parallel filters process successive segments of a bigger reference array, the extra two samples ensure that the final elements of the array can be correctly filtered.

Each sample is represented as an 8-bit value, resulting in 264 bits per buffer and a total of 528 1-bit registers for the module.

The input buffer captures samples synchronously from higher-level modules, while the output buffer sends filtered values for subsequent external modules. Both buffers are coordinated by the control unit, which asserts load and transfer signals based on the block dimensions and the progress of the cycle counter.

3.3 Control Unit

The control unit manages the sequential operation of the module when blocks contain more than 33 reference samples. It uses a cycle counter to track processing progress and a look-up table (LUT) to determine the number of cycles required per block based on the block width (W) and height (H). The LUT implements the formula $\lceil (2W + 2H - 1)/33 \rceil$ precomputed for all valid block dimensions stored in a small read-only memory (ROM), eliminating runtime arithmetic.

Once a new block is started, indicated by a *start* signal, the counter is reset to zero. It increments on each clock while the block is being processed, and asserts a *done* signal when the final cycle is completed, indicating that the last filtered samples of the block are available at the output.

Under certain conditions, some reference samples of a block may not require filtering (see 2.1.2). To handle this, the control unit monitors an input signal that indicates when the smoothing filter should be skipped. This ensures that, regardless of the block size, skipped blocks consume only a single cycle.

This very lightweight LUT- and counter-based control ensures deterministic operation, simplifies logic, and allows the combinational filter array to achieve maximum throughput without stalling.

4 Synthesis Results

The proposed reference sample filter architecture was implemented in VHDL and synthesized using Quartus Prime v25.1 targeting the Cyclone V FPGA (device 5CGXFC7C7F23C8). The synthesis results provide insights into the hardware resource utilization, timing performance, and scalability of the design for real-time VVC intra prediction.

4.1 Hardware Resource Utilization

The complete architecture, including the combinational filter array, input/output buffers, and control unit, occupies 595 adaptive logic modules (ALMs), corresponding to approximately 1% of the target device. A total of 553 1-bit registers are required to implement the input and output buffers, the cycle counter, and associated control signals. These results demonstrate that the smoothing filter, despite being applied to every intra block and lying on the critical path of prediction, can be implemented with negligible hardware overhead. In practical encoder and decoder architectures, where logic resources must be shared among several complex modules, such compactness is highly desireable and can facilitate integration with existing intra prediction modules.

4.2 Timing Performance

The timing analysis under the slow 1100 mV 0° C model indicates a maximum operating frequency of 181.79 MHz. This high performance is largely attributed to the highly optimized datapath described in Section 3.1.2. By decomposing the 3-tap FIR filter into two divide-by-two stages and eliminating unnecessary computations associated to LSBs, the design significantly reduces carry propagation and shortens the critical path. Compared to a naive implementation that performs full-width additions followed by a single division, the optimized structure requires fewer arithmetic operations and smaller adders, resulting in a faster and more hardware-efficient circuit. The synthesis results validate that this bit-level reformulation translates into measurable timing improvements.

4.3 Throughput Analysis

To evaluate throughput, a worst-case scenario was considered in which all intra blocks are of size 8×8 and all require filtering. In this case, each block can be processed in a single clock cycle by the 33-unit parallel filter array. Table 1 shows the required clock frequency for UHD 4K and 8K resolutions at 30 and 60 fps.

Even for 8K at 60 fps, the required frequency is only 31.10 MHz, which is nearly six times lower than the maximum frequency achieved by synthesis. This large performance margin indicates that

the architecture can be underclocked to reduce power consumption or deployed on lower-cost FPGA families without compromising real-time operation. It also suggests that the design is suitable for ASIC implementations, where even higher clock frequencies and lower power consumption can be achieved.

Table 1: Required clock frequency for worst-case 8×8 block scenario.

Resolution	Frame Rate (FPS)	Frequency (MHz)
4K UHD	30	3.89
4K UHD	60	7.78
8K UHD	30	15.55
8K UHD	60	31.10

4.4 Comparison with Existing Works

To the best of our knowledge, no prior work has presented a dedicated hardware architecture for the VVC reference sample smoothing filter. Existing studies on VVC intra prediction focus primarily on the angular prediction stage and assume that reference samples have already been pre-processed. The results presented here therefore fill an important gap by demonstrating that this mandatory step can be implemented with high throughput and negligible hardware cost. The proposed architecture complements existing intra prediction designs and can be directly integrated into complete VVC decoder pipelines, contributing to the development of efficient hardware support for the VVC standard.

5 Conclusion

The reference sample smoothing filter is a mandatory component of the VVC intra prediction process and lies directly on its critical path, yet it has received little attention in prior hardware design research. This work presented a dedicated and fully compliant hardware architecture capable of performing reference sample smoothing across all VVC block sizes, targeting the decoder. By reformulating the 3-tap FIR filter into two divide-by-two stages and eliminating unnecessary LSB computations, the proposed design achieves substantial reductions in arithmetic complexity while preserving bit-exact behavior.

The proposed architecture is small, requiring only 595 ALMs and 553 registers on a Cyclone V FPGA, and achieves a maximum operating frequency of 181.79 MHz. Throughput analysis shows that even under worst-case conditions, real-time processing for 8K video at 60 fps is sustained with a clock requirement of only 31.10 MHz, enabling significant underclocking. These results demonstrate that the proposed solution offers a high-throughput, low-cost, and easily integrable hardware block suitable for decoder implementations.

Future work will focus on further optimizing arithmetic-intensive datapaths within the codec by exploring alternative adder architectures, such as carry-lookahead and carry-select adders. These structures offer opportunities to shorten the carry-propagation chain at the expense of additional area, enabling a systematic exploration of performance–area trade-offs.

Acknowledgments

This research was funded by the *Instituto Federal Sul-rio-grandense* (IFSul) and the *Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul* (FAPERGS) – Grant number 25/2551-0000903-0.

References

[1] ITU-T. Versatile Video Coding. https://www.itu.int/rec/t-rec-h.266, Sep. 2023. Recommendation ITU-T H.266 (V3) (09/2023).
[2] ITU-T. High Efficiency Video Coding. https://www.itu.int/rec/t-rec-h.265, Jul. 2024. Recommendation ITU-T H.265 (V10) (07/2024).
[3] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (vvc) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3736–3764, 2021. doi: 10.1109/TCSVT.2021.3101953.
[4] ITU-T. Advanced video coding for generic audiovisual services. https://www.itu.int/rec/t-rec-h.264, Aug. 2024. Recommendation ITU-T H.264 (V15) (08/2024).
[5] Benjamin Bross, Jianle Chen, Jens-Rainer Ohm, Gary J. Sullivan, and Ye-Kui Wang. Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc). *Proceedings of the IEEE*, 109(9): 1463–1493, 2021. doi: 10.1109/JPROC.2020.3043399.
[6] Intel Corporation. Intel® core™ ultra processors series 2, 2024. URL https://www.intel.com/content/www/us/en/ark/products/series/241071/intel-core-ultra-processors-series-2.html#@Mobile. [Online; accessed July 2025].
[7] VeriSilicon Holdings Co., Ltd. Verisilicon unveils hantro vc9000d multi-format video decoder, 2024. URL https://verisilicon.com/en/PressRelease/HantroVC9000D. [Online; accessed July 2025].
[8] Allegro DVT. Al-d320 decoder ip. https://www.allegrodvt.com/products/al-d320-decoder-ip/, n.d. [Online; accessed November 2025].
[9] MediaTek. Pentonic 2000. https://www.mediatek.com/products/pentonic/2000, 2021. [Online; accessed November 2025].
[10] MediaTek. Pentonic 1000. https://www.mediatek.com/products/pentonic/1000, 2022. [Online; accessed November 2025].
[11] Allegro DVT. E320 vvc encoder video ip, 2024. URL https://www.allegrodvt.com/products/e320-vvc-encoder-video-ip/. [Online; accessed July 2025].
[12] V. Borges, M. Perleberg, M. Porto, and L. Agostini. Efficient architecture for vvc angular intra prediction based on a hardware-friendly heuristic. In *2023 IEEE 14th Latin America Symposium on Circuits and Systems (LASCAS)*, pages 1–4, Quito, Ecuador, 2023. doi: 10.1109/LASCAS56464.2023.10108393.
[13] V. Borges, M. Perleberg, M. Porto, and L. Agostini. High-throughput hardware design for the complete vvc angular intra prediction. In *2024 31st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 1–5, Nancy, France, 2024. doi: 10.1109/ICECS61496.2024.10848892.
[14] Marcel Corrêa, Mário Saldanha, Alex Borges, Guilherme Corrêa, Daniel Palomino, Marcelo Porto, Bruno Zatt, and Luciano Agostini. Av1 and vvc video codecs: Overview on complexity reduction and hardware design. *IEEE Open Journal of Circuits and Systems*, 2:564–576, 2021. doi: 10.1109/OJCAS.2021.3107254.
[15] I.E. Richardson. *Coding Video: A Practical Guide to HEVC and Beyond.* Wiley, 2024. ISBN 9781118711781.
[16] G.J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, 1998. doi: 10.1109/79.733497.