Journal of Real-Time Image Processing (2022) 19:539–550
https://doi.org/10.1007/s11554-022-01206-2

ORIGINAL RESEARCH PAPER

An efficient parallel-pipelined intra prediction architecture to support DCT/DST engine of HEVC encoder

Lakshmi Poola[1] · **P. Aparna**[1]

Received: 13 July 2021 / Accepted: 6 February 2022 / Published online: 21 February 2022

Abstract
The complexity of intra prediction in high-efficiency video coding (HEVC) is increased due to the addition of five variable sized prediction units (PUs) and 35 directional predictions. In this work, we propose an efficient parallel-pipelined architecture that can process 8 samples in parallel for every clock cycle. The functional units needed to predict the PU samples work in a pipelined fashion. With this balanced combination of parallel-pipelined structure, we are able to achieve higher throughput with limited hardware resources than existing literature works. The samples are processed row-wise, so that they can be directly transform coded, thus eliminating the need for an intermediate memory buffer of 8 K between the two modules. A compact reconfigurable reference buffer of size 0.8 KB is incorporated to reduce the read-write latency associated with reference samples' fetching. A dedicated module for arithmetic operations is used in the intra engine that ensures the reuse of multipliers to increase the hardware efficiency. The architecture so designed supports all the PU sizes and directional modes. The proposed design is tested and implemented on a field-programmable gate array (FPGA) platform operating at 150 MHz frequency to achieve 8 samples throughput with a hardware cost of 16.2 K Look-Up Tables (LUTs) and 5.7 K registers to support HD 4 K real-time video encoding applications.

Keywords HEVC · Intra prediction · FPGA · Hardware architecture · Pipeline · Parallel · Reconfigurable

1 Introduction

High-definition (HD) and ultra-high-definition (UHD) videos are currently adopted by security industries, video-based medical applications, broadcasting industries, etc. The pandemic outbreak in the years 2019–2021 has caused an explosion of video applications and a steep increase in video content. This enormous amount of video data which is further estimated to grow in the future [1] has emerged as the bottleneck for existing storage and transmission infrastructures. High Efficiency Video Coding (HEVC)/H.265 video compression standard is developed by expanding on the efficiencies of its predecessor standard H.264 [2]. HEVC essentially offers the same level of picture quality as H.264, but with 50% more bit rate saving which significantly reduces bandwidth and storage requirements [3]. This effectively lowers infrastructure costs and ultimately makes high-resolution surveillance systems, broadcasting, and high-quality video transmissions for medical applications more affordable. In turn, encoding and decoding HEVC streams require substantially more computational power [4]. Such increase in computational power requirement has led to the implementation of HEVC on reconfigurable hardware platforms like field-programmable gate arrays (FPGAs) to meet real-time media applications. The implementation of dedicated architecture on FPGA can achieve high-speed performance due to programmable and reconfigurable characteristics with multiple optimisations to accelerate video encoding in comparison to general-purpose processors.

HEVC uses an initial prediction, either intra frame and/or inter frame, followed by a transform, quantization, and entropy coding over the residual information [3, 5]. Flexible quad-tree-based block partitions and more directional predictions are introduced in HEVC intra prediction to achieve better compression on the encoder side. The encoder's input video frames are divided into square blocks called largest

✉ Lakshmi Poola
lakshmipoola@gmail.com

P. Aparna
p.aparnadinesh@nitk.edu.in

[1] Department of Electronics and Communications Engineering, National Institute of Technology Karnataka, Surathkal, Karnataka 575025, India

coding units (LCUs), which are further divided into coding units (CUs). CUs are further broken down into prediction units (PUs) and transform units (TUs). HEVC introduces five different sizes of PUs, namely blocks of sizes 4, 8, 16, 32, and 64, as well as TUs of sizes 4, 8, 16, and 32, as shown in Fig. 1. HEVC has 35 directional modes for each PU size. Several architectures have been proposed in the literature to implement intra prediction on FPGA platforms. However, most of these do not consider all the following requirements for hardware design: real-time constraint, high throughput, and low area footprint.

Li et al. [6] noticed that 4×4 PUs account for 66% of the total PUs on the decoder side. They proposed a 4×4 intra prediction engine supporting only 17 directional predictions using two parallel datapaths. In [7], a pipelined HEVC decoder to support 4K applications is proposed. Single cycle reference processing in intra prediction is used to optimise the architecture. In [8], a multiplexer-based intra prediction engine with parallel datapaths is proposed. They equipped one datapath for DC/planar mode and the other for directional modes to process 4K@24fps videos. In [9], a prediction engine with parallel datapaths to support DC, planar and angular modes is proposed. Moreover, the engine is PU block-based, processing 16 samples per clock cycle. Min et al. [10] proposed an adaptive scanning-based, fully pipelined intra prediction engine with a throughput of 4 samples per clock cycles. The adaptive scanning leads to both column-wise and row-wise predictions, due to which prediction samples need to be stored in an intermediate buffer before sending them to the transform engine. Dadan et al. [11] propose a flexible intra encoder engine with 32 parallel reconfigurable processing elements. In our previously proposed work [12], a mixed parallel-pipelined intra prediction engine is designed to support 4K@30fps video applications. A compact reusable 1KB reference buffer is incorporated in the design to speed up the prediction operations. Fan et al. in [13] used 32 parallel processing elements to support all PU sizes and angular modes. Reference buffers are reused to reduce the resource utilisation. In [14], two intra prediction engines to support DC and planar modes are proposed. One is a fully parallel engine, while the other is a parallel-pipelined architecture, with a throughput of 8 samples per clock cycle. However, engines are limited to only two angular modes. Palomino et al. [15] implement two parallel datapaths to increase the throughput and the memory is reused by introducing buffers. However, only 1K@13fps video applications are supported. In [16], five parallel datapath design optimised by taking advantage of the similarity in the prediction equations is introduced. However, the design is limited to 4, 8 size PUs only with reference buffer size of 1.8KB. Several hardware architectures take advantage of the parallel nature of angular predictions and use separate parallel datapaths to enhance the throughput. In [17], two parallel datapaths are designed, with one dedicated to 4×4 PUs and the second one for 8, 16, and 32 PUs. The 4×4 PU datapath will be of an advantage if the input frame has more PUs of size four. In [18], prediction computations are carried out using digital signal processing (DSP) unit in place of adders and shifters. Several architectures achieve higher throughput with more parallel elements. In [19], 32 reconfigurable processing elements (PEs) in parallel are used for intra prediction supporting all PU sizes and all angular modes.

Analysing the implementations in the literature and keeping the current HD/UHD applications in view, the following observations are drawn.

- It is important to design an intra prediction engine that can support all the PU sizes and all the directional modes while achieving high throughput.
- Adding more parallel processing elements may increase the throughput but that comes with an increased hardware cost and power requirements. Hence, it becomes

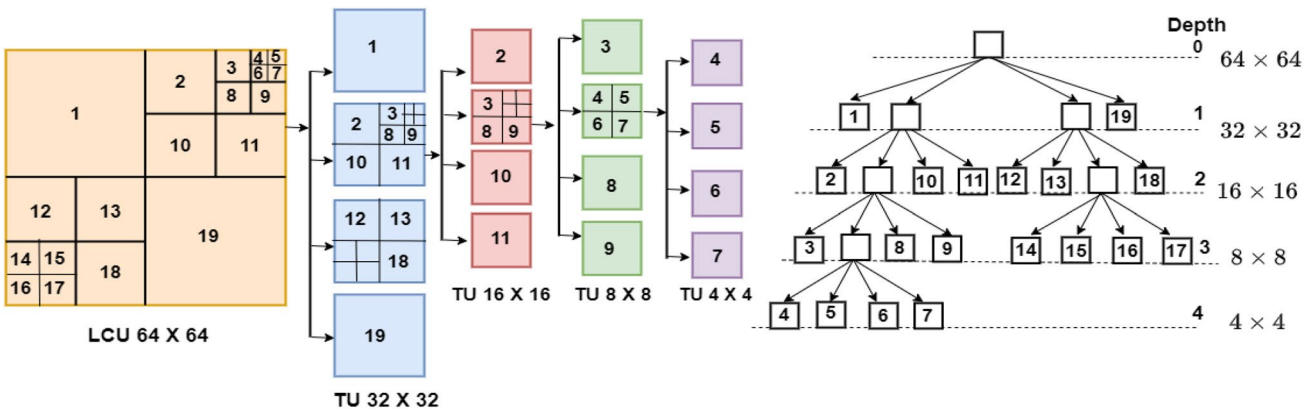

Fig. 1 Illustration of LCU and quad-tree partitioning of TUs introduced in HEVC [2]

very crucial to maintain a balance among the parallel/pipeline/sequential elements used in the design to achieve an efficient hardware engine.
– Most of the hardware architectures implemented in the literature requires an intermediate reference buffer between the intra prediction and the discrete cosine transform/discrete sine transform (DCT/DST) engines. This is because the intra prediction output is in the form of PU blocks, whereas DCT/DST engine requires TU blocks as inputs. Furthermore, the samples to the DCT/DST engine are scanned row-wise first and then column-wise from variable sized TUs.

To cater to all the above requirements, an efficient improved intra prediction architecture is proposed with the following key techniques.

1. The intra prediction engine is configured to generate 8 samples in parallel in every clock cycle with support to all 35 directional modes and all possible PU sizes. Furthermore, this novel intra prediction engine is implemented on the FPGA to achieve a high throughput to support 4 K videos. A reconfigurable compact reference buffer to hold the source references is included in the engine, which reduces the time required to fetch the references considerably.
2. A fully pipelined DCT/DST-based prediction engine is proposed. The samples are always operated and generated row-wise in the intra prediction module. With this, DCT/DST and intra prediction engines can operate in parallel ensuring the high throughput.
3. The hardware consuming interstage memory buffer between transform and intra prediction modules is eliminated by introducing row-wise processing of samples in our design.
4. Arithmetic operations in the directional angular predictions are one of the causes to increase the complexity of intra prediction engine. The prediction function is uniform in 33 directional modes. Hence, we have proposed a dedicated module to process multiplication and addition operations to ensure the reuse of multipliers present in the digital signal processing (DSP) slices of the FPGA, which in turn improves the efficiency of the hardware engine.

The rest of the paper is organized as follows. The semantics of intra prediction encoding procedure is described in Sect. 2. Next, the proposed hardware intra prediction architecture and its timing analysis are explained in Sect. 3. The detailed performance analysis of the proposed design is presented and discussed in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2 Intra prediction Encoding in HEVC

Intra prediction in HEVC greatly improves the compression efficiency and reduces the spatial data redundancy [3]. In intra module, the neighbouring samples, which are often coded ones, are used as references to predict the current PU block. The samples from the neighbouring left (L) and left-bottom (LB) column, top (T) and top-right row (TR), and the top-left or corner sample (C) of the current PU are used as an array for prediction and are shown in Fig. 2 along with the directional prediction angles. In the cases when some of the references are unavailable, the array is extended with the last available sample, and when no reference samples are available, the array is filled with the initial value of 128 (for 8-bit video). Each PU is subjected to 35 directional predictions. Modes 0 and 1 are Planar and Direct-Current (DC) predictions, respectively. Directional prediction modes (2–34) are grouped into the horizontal (2–18), and the vertical (19–34) predictions as demonstrated in Fig. 2.

For directional predictions (vertical or horizontal), the references are grouped as the main array and the side array. In vertical directions, the main array is composed of top-left, top, and top-right samples, with the left-column samples used as the side array. The top-left, left, and left-bottom column samples make up the main array for horizontal predictions, with the top-row samples as the side array. Only the main array is used for positive directional predictions, whereas for negative directional predictions, both main and side arrays are used.

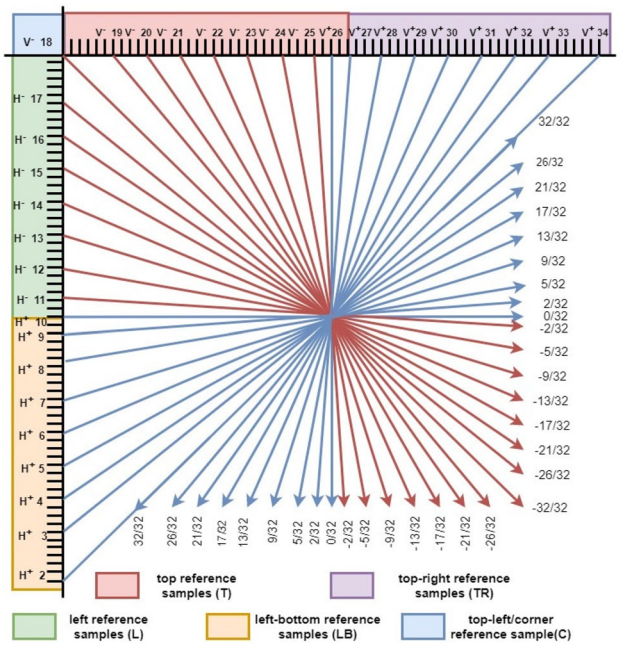

Fig. 2 Illustration of reference samples and intra directional modes with their indices used in HEVC [2]

In negative directional modes, the side array is extended to complete the reference array, and this is demonstrated in Fig. 3 for a 4×4 PU. Main reference array and the extended array samples are determined using the relations (1)–(2). *IAng* in (2) is the inverse angle and derived using the look-up table $[-4096, -1638, -910, -630, -482, -390, -315, -256]$, these values are associated with the corresponding directional angles (negative) as in the look-up table $[-2, -5, -9, -13, -17, -21, -26, -32]$. Reference arrays for horizontal modes are determined using relations (1)–(2) by replacing i with j

$$RefAng[i] = Ref[-1+i][-1] \quad (i \geqslant 0) \tag{1}$$

$$RefAng[i] = Ref[-1][-1 + ((i \times IAng + 128) \gg 8)] \quad (i < 0). \tag{2}$$

Block partition of the input video stream gives rise to contouring artifacts at the edges of the PU blocks. HEVC codec addresses this by applying two different methods of filtering/smoothing techniques on the reference samples [20]. Filtering is applied based on the PU sizes and the directional modes. For PUs of size 4, no filtering is applied, and for PUs of size 8, filtering is performed only on modes 2, 18, and 34. Whereas for PUs of size 16, all modes except 9, 11, 25 and 27 use filter. For PUs of size 32, all modes except 10 and 26 undergo filtering. In the first method, a three-tap filter $([1, 2, 1]/4)$ is applied by default on all the reference samples leaving out the outermost samples using relations (3)–(5)

$$Ref[-1][-1] = (Ref[-1][0] + 2 \times Ref[-1][-1] + Ref[0][-1] + 2) \gg 2 \tag{3}$$

$$Ref[-1][j] = (Ref[-1][j+1] + 2 \times Ref[-1][j] + Ref[-1][j-1] + 2) \gg 2 \tag{4}$$

$$Ref[i][-1] = (Ref[i+1][-1] + 2 \times Ref[i][-1] + Ref[i-1][-1] + 2) \gg 2, \tag{5}$$

where, $i, j = 0, 1, 2, ..., 2N_{Block} - 2$. In the second method, strong intra smoothing is applied, in which a linear interpolation of reference samples (including the corner samples) is generated.

2.1 Intra directional predictions

Planar and DC predictions are applied on smooth and gradually changing video content but for high-frequency and complex-textured content, directional predictions are recommended in HEVC [3, 5]. In the intra planar mode, each sample is predicted by finding the weighted average of the horizontal and vertical reference samples using the relation (6), where $PredH[i][j]$ and $PredV[i][j]$ are given by (7) and (8), respectively

$$PredB[i][j] = (PredH[i][j] + PredV[i][j] + N_{Block}) \gg (\log_{(} 2N_{Block}) + 1) \tag{6}$$

$$PredH[i][j] = (N_{Block} - 1 - i) \times Ref[-1][j] + (i+1) \times Ref[N_{Block}][-1] \tag{7}$$

$$PredV[i][j] = (N_{Block} - 1 - j) \times Ref[i][-1] + (j+1) \times Ref[-1][N_{Block}]. \tag{8}$$

The DC prediction is applied to the flat regions of the input video. In DC mode, the prediction is simply the average of row and column samples. For PUs of size 32×32, all the samples in the current PU are replaced by DC_{avg}, given by (9). For other size PUs, corner, top-row, and the left-column samples are filtered using relations (10), (11), and (12), respectively

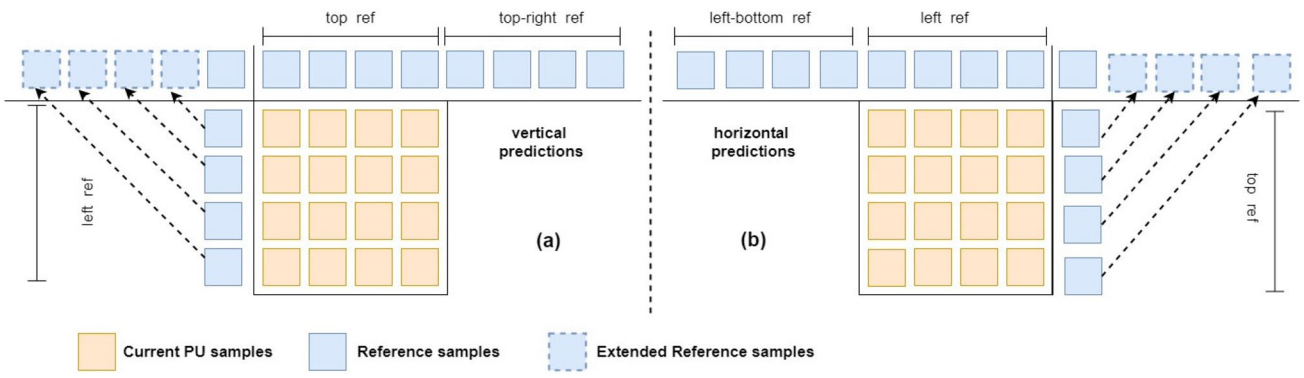

Fig. 3 Demonstration of reference samples' array mapping for negative vertical and horizontal modes in case of one 4×4 PU. Main array and the extended main array are shown. In **b**, all the symbols are rotated by 90° in the clockwise direction

$$DC_{Avg} = \left(\sum_{i=0}^{N_{Block}-1} Ref[i][-1] + \sum_{j=0}^{N_{Block}-1} Ref[-1][j] \right) \gg (\log_2 N_{Block} + 1) \quad (9)$$

$$PredB[0][0] = (PredB[-1][0] + 2 * DC_{Avg} + PredB[0][-1] + 2) \gg 2 \quad (10)$$

$$PredB[i][0] = (PredB[i][-1] + 3 * DC_{Avg} + 2) \gg 2 \quad (11)$$

$$PredB[0][j] = (PredB[-1][j] + 3 * DC_{Avg} + 2) \gg 2. \quad (12)$$

Intra directional predictions are performed by applying interpolation on the reference samples [3]. Two adjacent references are used to predict each sample. References are selected using the index—*ind* given by (13), where *Ang* is the direction associated with directional mode and $\gg$ is the bitwise shift operation. *fracPrt* is the coefficient, given by (14). For horizontal modes, *ind* and *fracPrt* are generated using (13) and (14) simply by replacing *j* with *i*

$$ind = ((j + 1) \times Ang) \gg 5 \quad (13)$$

$$fracPrt = ((j + 1) \times Ang)\&31. \quad (14)$$

For a given vertical angular mode, each sample in the PU block is predicted using the following relation. For horizontal predictions, *i* is replaced with *j* in (15)

$$PredBlock[i][j] = ((32 - fracPrt) \times RefAng[i + ind + 1] + fracPrt \times RefAng[i + ind + 2] + 16) \gg 5, \quad (15)$$

where, $i, j = 1, 2, ..., N_{Block} - 1$ for (6)–(12) and (15).

3 Proposed hardware architecture of the DCT/DST-based intra prediction engine

The key aspects of the proposed engine are to enable row-wise predictions and operate in a pipelined manner to achieve eight samples' throughput with an optimum hardware cost. This section gives the techniques involved to implement the architecture (Fig. 5).

3.1 Top-level architecture of the proposed efficient intra prediction engine

The top-level architecture of the proposed intra prediction engine is outlined in Fig. 4 which consists of the following stages.

1. **Reference Selection (RS):** The source references from the neighbouring left, left-bottom, left, top-left, top, and top-right positions of the current PU are fetched for prediction. The availability of the neighbouring references is updated using a 5-bit register.
2. **Reference Buffer (RB):** Using the selected references, reference arrays for all the directional modes and PU

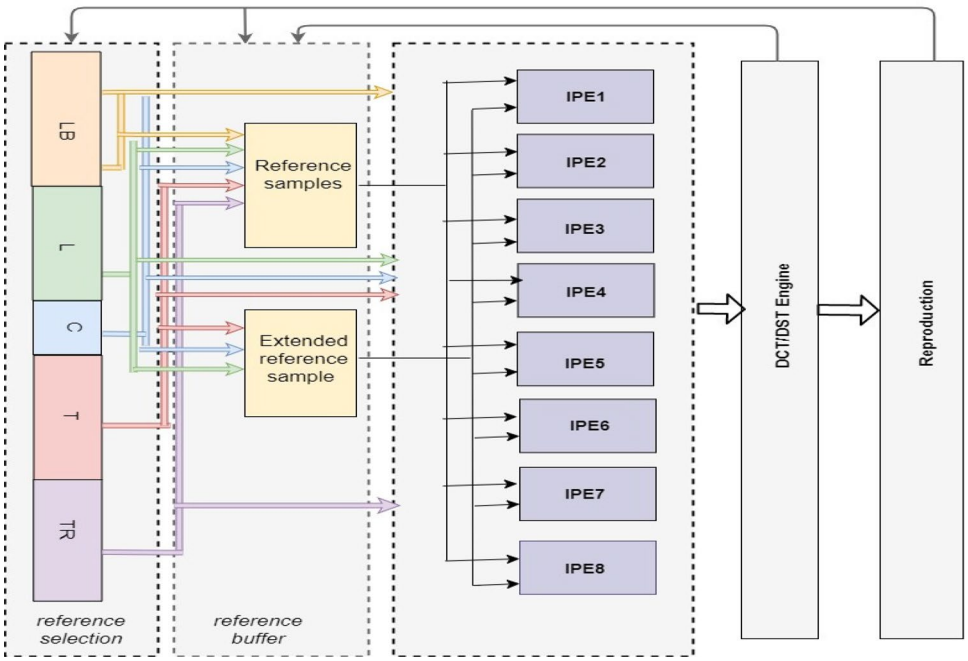

Fig. 4 Block diagram showing the top-level architecture of the proposed intra prediction engine

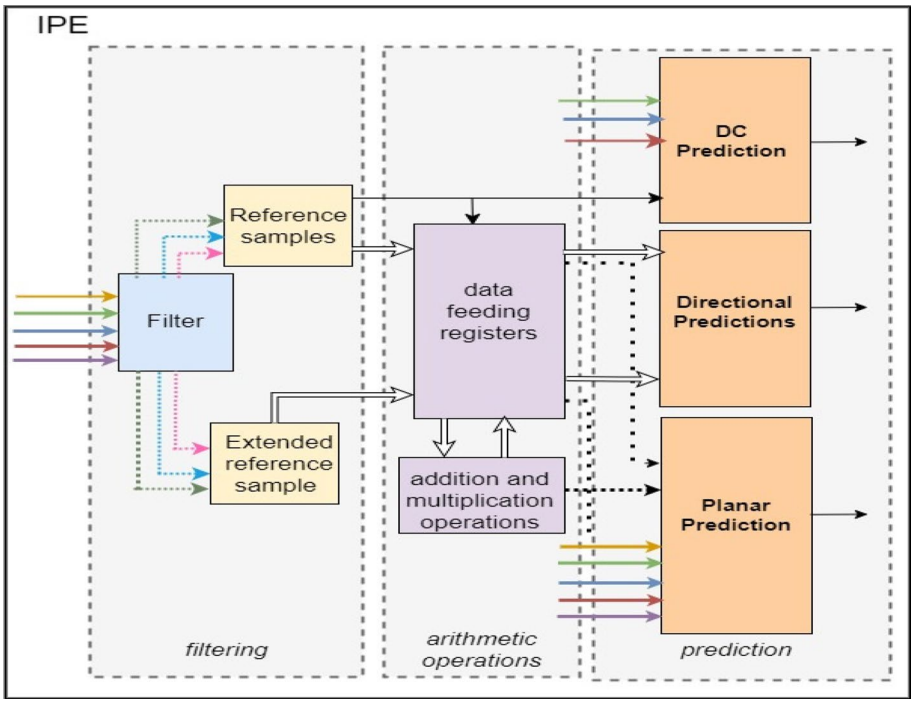

Fig. 5 Block diagram of the proposed Intra prediction Processing Element (IPE)

sizes are pre-determined and stored in registers. Reference samples and extended reference sample modules store the positive and negative directional mode references. For a given directional mode and PU size, the corresponding reference array is chosen and stored in the reconfigurable reference buffer for further processing. Predetermining of references saves the clock cycles required to fetch them and thus avoid latency associated with data loading.

3. **Intra prediction Processing Element (IPE):** Intra prediction engine consists of 8 parallel IPEs, which can process 8 samples in one clock cycle. Each IPE has a filtering stage, arithmetic operations unit, and prediction unit, explained below.

   Filtering (F): In this stage, reference samples undergo filtering based on the following conditions. The first decision is to apply three-tap or strong filtering based on the threshold. Then, based on the directional modes and PU sizes, filtering is applied to the references. The filtered samples move to the next stage. For 4×4 PUs, smoothing is not required. They are moved to the next stage through registers without applying any filtering.

   Arithmetic Operations (O): Data feeding registers and Addition–Multiplication (AM) modules are part of this stage. The data feeding module is an intermediate stage solely designed to exchange data between filtering and AM modules and between AM and prediction modules. Directional predictions for 2–34 modes are uniform operations as given by (15). The addition and multiplication functions in the prediction process are carried out in the AM module. The dedicated AM module reuses the multipliers efficiently and ensures the reuse of DSP slices. The FPGA platform used to test the prediction engine uses DSP48E1, with the proposed design uses only forty DSPs.

   Prediction Unit (P): The final stage consists of three predictors that process and produce eight samples in each clock cycle. Each predictor is explained in detail in Sect. 3.4.

The proposed engine efficiently processes eight samples in parallel using a pipelined design supporting all the directional modes and the variable sized PUs.

3.2 DCT/DST engine-based intra prediction unit

Intra prediction and DCT/DST modules are two fundamental and consecutive units that play a vital role in improving the compression performance of the HEVC encoder. According to the encoding standard, the input/output formats of the intra module and DCT/DST engines are different. In intra module process, the PU blocks both row-wise and column-wise based on the directional modes. The DCT/DST module processes row-wise samples from the variable sized TUs. Due to this inconformity in the data exchange between both the modules, an intermediate prediction register array becomes necessary to store the samples. The intermediate buffer is shown using dotted lines in Fig. 6. Generally, in horizontal predictions, column-wise samples are produced which cannot be directly processed

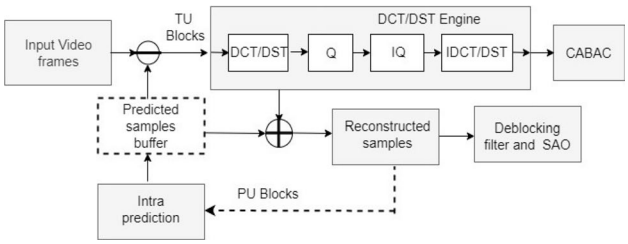

Fig. 6 Block diagram of HEVC encoder with the predicted samples buffer shown using the dotted lines

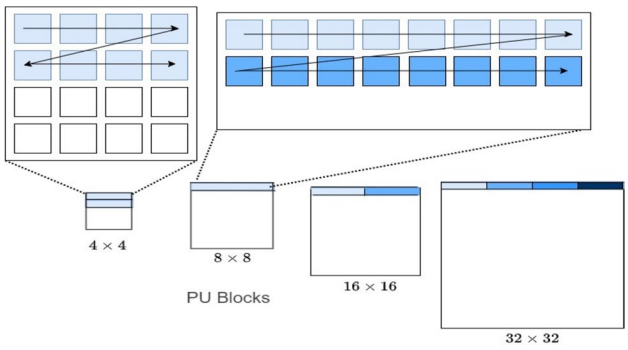

Fig. 7 Illustration of row-wise prediction of samples in intra prediction engine

by the DCT/DST engine and needs to wait till the entire block is predicted. This affects the overall throughput of the HEVC encoder. Apart from this, using the intermediate buffer is an unnecessary hardware overhead. The proposed intra prediction engine is designed to continuously provide unified row-wise predictions. In Fig. 7, row-wise scanning and selection of PU samples used for prediction are shown. The largest TU is 32 × 32, with an 8-bit depth video content, a buffer of size 8KB is required to store this TU/PU. The proposed method saves up to 128 clock cycles in the case of horizontal predictions for a 32 × 32 PU. The gain for an LCU is 512 clock cycles, and for a single 4K picture frame, the savings are close to 1.03 million clock cycles which is a substantial improvement. This method also allows DCT/DST and intra engines to run in parallel to keep the overall throughput high in the encoder. Furthermore, the hardware-intensive intermediate buffer is eliminated entirely in our design, resulting in a more efficient hardware engine.

3.3 Efficient reconfigurable reference buffers

Source references are located in five neighbouring positions of the current PU. The source references are selected in the first stage and are stored in buffers/registers to facilitate the prediction in the second stage. 32 × 32 is the biggest PU recommended in the standard. Hence, in our design, a reference buffer of size 776 bits is incorporated, i.e, the range of references for a 32 × 32 PU, is limited to −32 to +64 reference samples. For an 8 bit-depth video input, the maximum buffer size required is 776 bits.

The mechanism of reference buffer management is demonstrated in Fig. 8. The PU and the necessary references used for prediction are highlighted. The PUs are scanned in the Z-order as shown. Block C is the PU under prediction which is a sub-block of Block 2. Prediction is already completed in Block 1, Block A, and Block B and are available as shown by the highlighted samples. These samples are used to predict the current PU. Border samples from this predicted Block C act as the references for predicting Block D. These samples replace the older data in the reference buffer, as demonstrated in Fig. 8b. The same reference buffers are reused to store new data. Some samples get overlapped in the reference buffer, some get replaced, and unused older samples are discarded. Thus, using the reference buffer makes the management and storing of neighbouring samples simpler. Availability and validity of source references are easily determined using the reference buffer. The reference buffer

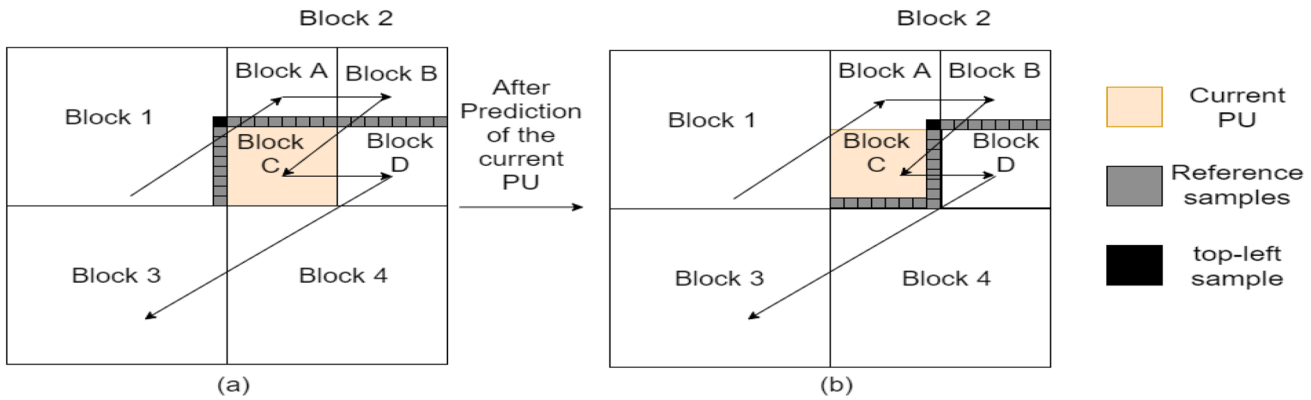

Fig. 8 Illustration of reference buffers. **a** Reference buffer before the current PU is predicted. **b** Updated reference buffer after the current PU is predicted

used in the design simplifies the fetching operation by saving the clock cycles required to fetch reference samples. For example, one 4×4 PU requires 17 references for prediction. With eight samples' processing engine, 3 clock cycles are required to fetch the references. Similarly, PUs of sizes 8, 16, and 32 require 33, 65, and 129 references for prediction which in turn needs 5, 9, and 17 cycles, respectively, to fetch the references. In our design, because of the reference buffers, reading and writing of the references take just one cycle resulting in a high-speed operation. Also, 776 bit buffer used in the proposed design is the smallest compared to other state-of-art designs available in the literature.

3.4 Planar, DC, and directional prediction architectures to support DCT/DST engine

The final stage in the proposed design is equipped with three prediction engines, viz., planar, DC, and the directional prediction (for predicting 33 directional modes) modules. The predicted bottom row and the right column samples of the current PU act as the source references for the next PU prediction and get updated in the reference buffer. All three predictors are designed to process and produce eight samples in each clock cycle.

3.4.1 DC prediction engine

DC prediction takes place in two steps: (1) Computation of DC_{avg}: The neighbouring top and left filtered references of the current PU are used to compute DC_{avg}, using relation (9). (2) Boundary sample filtering and Output Computation: The decision to filter boundary samples is made. If the PU size is 32, then DC_{avg} is the final output. For chroma PUs, DC_{avg} is the final output. For luma PUs of sizes 4, 8, 16, the final output is the filtered samples from the neighbouring top row and the left column.

A reusable adder module is used for DC prediction. This module comprises four three-input adders. The adder module computes the DC value and is also used for filtering the boundary samples. The DC mode has the least computational complexity.

3.4.2 Planar Prediction Engine

Block diagram of the planar predictor for a 4×4 PU is shown in Fig. 9. Sample by sample prediction is carried out by taking the weighted average of horizontal and vertical references. Each sample prediction involves multiplication followed by accumulation of the multiplied results. In the proposed planar predictor, accumulators are used to ensure row-wise predictions during both horizontal and vertical modes. N is the PU block size in Fig. 9. The planar prediction is computationally most expensive mode in the intra engine.

3.4.3 Directional Prediction Engine

Directional prediction takes place in three steps as follows: (1) Reference samples are selected in a unique way. For negative modes, side and main arrays are used, which are stored in extended reference samples and reference sample modules, respectively. For positive modes, only the main reference array is selected. Two references are required to predict one sample and are chosen from the array. (2) Index and weights are generated. The index gives the position of the source reference, using which its offset is also found. (3) Each sample is predicted using the index—*ind*, and weight—*fracPrt*.

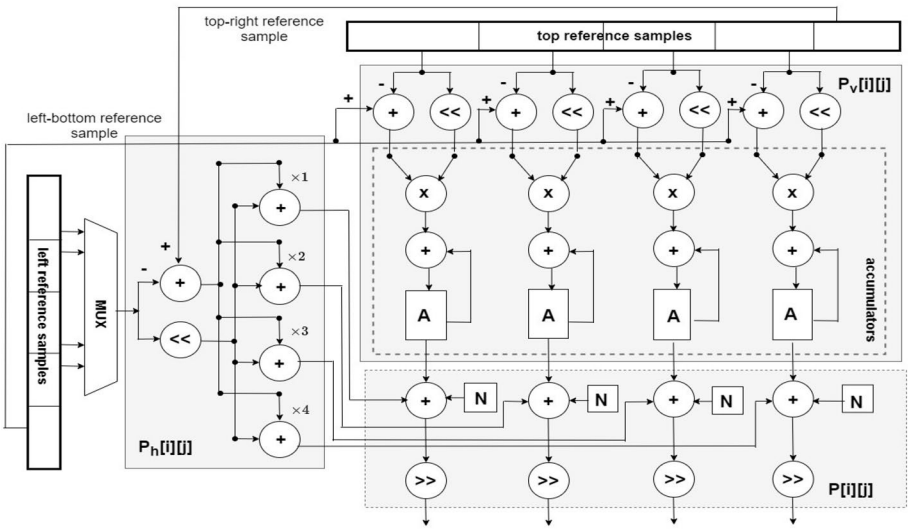

Fig. 9 Block diagram showing the Low-Level architecture of Planar Prediction Engine for a 4×4 PU

The engine ensures row-wise predictions by following a mechanism to select reference samples from the filter stage. For the vertical predictions, each prediction sample can determine the position of its input references given by *ind*, using (13), and its offset is determined. Using these two source references, the prediction is derived from (15). In the case of horizontal predictions, source reference and its offset are from neighbouring left column, which are not in the same row. Therefore, each prediction sample has to follow its own source selection logic. However, all predictions in the same column will always have the same set of references in the vertical direction. In this case, the reference registers are operated as the shifting window. In the case of each set of horizontal predictions, the projection of each prediction in the same row is maintained constant, and the main registers perform the right shift operation on the sample pattern to complete the row prediction. Index and weights for directional modes for all the PU sizes are fixed. Hence, in our design, index and weights for all the directional modes and PU sizes are precalculated and stored using registers to reduce the number of multipliers used in the design.

3.5 Timing Analysis of the Intra prediction Engine

Intra prediction is fully pipelined and the timing analysis for 4×4 PUs is demonstrated in Table 1. The same analysis holds good for larger size PUs. In Fig. 8, assume Block A, B, C, and D to be 4×4 PUs, and Block 1, 2, 3, and 4 as 8×8 PUs. Block A is the first PU under prediction, considered in the analysis. In *reference selection* (RS) stage, all the required references for all PU sizes are loaded into the reference registers in one clock cycle. Reference arrays for all prediction modes and for all PU sizes, *ind*, *fracPrt* are determined and saved in registers in this stage. In the next clock cycle, based on the prediction mode, corresponding reference buffers (RB) are updated.

The intra prediction engine consists of 8 IPEs working in parallel and perform filtering (F), arithmetic operations (O), and predictions (P) on eight samples in every clock cycle. Hence, one 4×4 PU is processed in 2 clock cycles. This is demonstrated in Table 1: Block A1 represents processing first and second rows (8 samples), while Block A2 represents third and fourth rows of the Block A. Thus, Block A (A1 and A2) completes filtering in clocks 3 and 4; arithmetic operations in clocks 4 and 5; and predictions in clocks 5 and 6. RS operation for Block B starts in clock 3; however, RB is stalled till Block A is predicted. Once PA1 is completed in clock 5, RB for Block B can resume followed by FB, OB, and PB. Similarly, as there is a dependency of Block D on the reference samples of Block C, RB operation for Block D is stalled till clock 11. Block 3 prediction follows in the similar fashion.

4 Experimental Results and Discussion

The proposed intra prediction engine is implemented on a 28nm Artix-7 FPGA board with a dual-core ARM Cortex-M1 processor operating at 150 MHz frequency to test the performance. The prediction engine is described using Verilog hardware description language (HDL). For analysis, simulation, synthesis, and implementation reports are generated using the Vivado design suite.

The behavioural simulation results for angular prediction mode 19 are shown in Fig. 10. The simulation shows eight row-wise predicted samples in each clock cycle.

The experimental results in Table 2 summarize the hardware cost in terms of gate count (LUTs), slice registers, and DSPs for each pipeline stage of the proposed engine. The intra top module in the table combines all the five stages into one module. The engine uses 16.2K LUTs, 5.7K registers and 40 DSPs, operating at 150MHz frequency. It supports all PU sizes and all directional modes processing an output of 64 samples/clock cycle. The engine processes 8 samples with each sample being represented with 8 bits, which gives $8 \times 8 = 64$ samples as output in each clock cycle.

Table 1 Timing analysis of intra prediction engine for 4×4 PUs

Clock	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Block A1	RS	RB	FA1	OA1	PA1	–	–	–	–	–	–	–	–	–		
Block A2	–	RS	RB	FA1	OA2	PA2	–	–					–	–	–	–
Block B1	–	–	RS	–	–	RB	FB1	OB1	PB1							
Block B2	–	–	–	RS	–	–	RB	FB2	OB2	PB2	–	–	–	–	–	–
Block C1	–	–	–	–	RS	–	–	RB	FC1	OC1	PC1	–	–	–	–	–
Block C2	–	–	–	–	–	RS	–	–	RB	FC2	OC2	PC2	–	–	–	–
Block D1	–	–	–	–	–	–	RS	–	–	–	–	RB	FD1	OD1	PD1	–
Block D2	–	–	–	–	–	–	–	RS	–	–	–	–	RB	FD2	OD2	PD2

RS—Reference Selection *RB*—Reference Buffer *F*—Filtering *O*—Arithmetic Operation *P*—Prediction

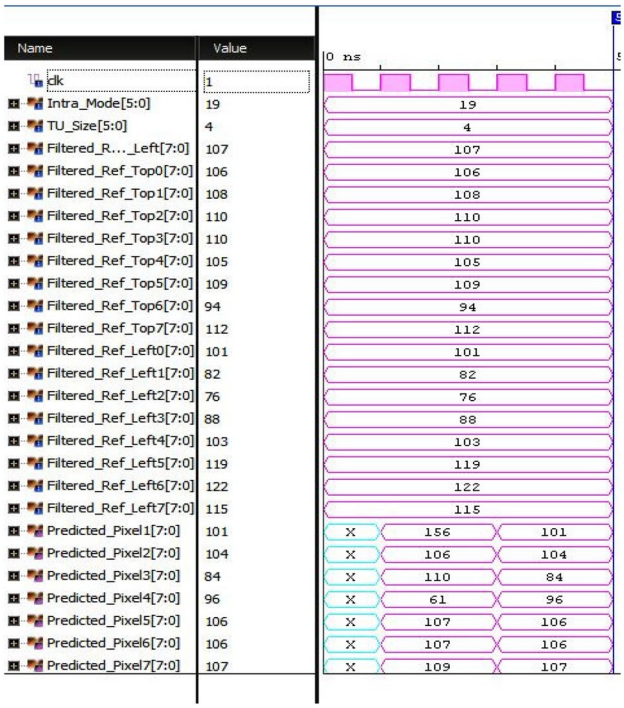

Fig. 10 Simulation results for mode 19 angular prediction

Table 2 Resource utilisation for the proposed intra prediction engine

Module	Sub Module	Gate Count	Register Count	DSP Count
Reference selection	–	259	129	–
Reference buffer	–	4268	1199	–
Filtering	–	2879	1047	–
Arithmetic operations	Data feeding reg	967	213	–
	AM	755	563	40
Prediction	DC	625	109	–
	Planar	242	66	–
	Directional	5401	1087	–
Intra top	–	802	1360	–
Total	–	16,216	5773	40

Table 3 compares the performance of the proposed design with other state-of-the-art counterparts in terms of technology, clock speed, supported PU block sizes, directional modes, hardware cost (in terms of LUTs and registers used by the design), hardware efficiency, supported video applications, and the reference buffer size used in the designs. Hardware efficiency is equal to (*Max Frequency* × *Througput* / *Hardware cost*).

In [7], an HEVC decoder is designed, and in Table 3, only the intra prediction module parameters are listed. Hardware cost is 63.3% higher, and the buffer memory size is 90% larger than our proposed architecture. Design in [8] includes five parallel datapaths to achieve higher throughput. However, the strategy is limited to 4K@24fps video applications, with a hardware resource utilisation of 77K LUTs; this design uses 90.4% more hardware. Design does not mention using register buffers and performance is tested on both Virtex-6 and 7 FPGA platforms.

Intra predictor in [9] processes 16 samples/cycle. Design is PU-based, which requires intermediate memory. The design predicts one 4×4 PU using 6 cycles, which gives an average throughput of 2.33 samples/cycle. The hardware has 85.5% more LUTs and 94% more registers than our proposed design. Fully pipelined intra predictor in [10] predicts 4 samples in parallel for each clock cycle. LUTs cost is slightly less and 13% better than our proposed design, but the reference buffer memory size is 86.6% larger for 50% less throughput. Data dependency in intra module is avoided with modified scanning techniques, which produces both row-wise and column-wise prediction samples, and hence, an intermediate memory buffer becomes necessary.

In [11], a flexible intra encoder engine is proposed; Table 3 lists only the intra prediction parameters. This engine has 17% better hardware efficiency but at the cost of 48% more hardware and 9.5 times larger buffer size and runs at 16% higher frequency. Also, because of more parallel elements in the design most of the hardware remains unused most of the time. The intra prediction engine in [12] stores the predicted samples in the buffer memory, which is a hardware overhead, and the reference buffer size is 20% larger than the proposed design. Predetermiming and storing *ind* and *weights* for all PUs and directional modes result in a higher slice register count in the proposed design, which does not affect the hardware efficiency or throughput.

Figures 11 and 12 show the graphical representation of the proposed work with other similar state-of-the-art works available in the literature. Throughput and hardware cost for each throughput are compared in Fig. 11, and [8] is omitted in the graph, because its throughput is very low for a very high hardware cost. The hardware costs of [7–9] are significantly higher, whereas the hardware costs of [11, 12] and the proposed work are nearly identical. However, as shown in Fig. 12, [11] achieves the same throughput with 9.5 times larger buffer, and [12] with 20% larger reference memory.

The novelty of the proposed hardware architecture is in the elimination of intermediate buffer memory by designing a row-wise intra prediction engine. This enables the intra- and DCT/DST modules to operate in parallel in the HEVC encoder to achieve high throughput. The proposed intra engine has a smaller reference buffer compared to the other works and achieves a throughput to support HD video applications at a low hardware cost.

Table 3 Comparison of the proposed work with other similar works

Parameter	[7] (2016)	[8] (2016)	[9] (2017)	[10] (2017)	[11] (2019)	[12] (2020)	This Work
Technology	28 nm	28 nm	28 nm	65 nm	28 nm	28 nm	28 nm
Max Frequency	150 MHz	219 MHz	119 MHz	110 MHz	175 MHz	110 MHz	150 MHz
PU size	all	all	all	all	all	all	all
Directional modes	all	all	all	all	all	all	all
Throughput (Samples/clock cycle)	2.6	0.2	16	4	16	8	8 samples of 8-bit depth
Hardware cost	43 K LUTs 22K registers	170 K LUTs 110 K registers	112 K LUTs 12 K registers	14 K LUTs 5.5 K registers	31.2 K LUTs 10 K registers	16 K LUTs 122 registers	16.2 K LUTs 5.7 K registers
Hardware Efficiency (8-bit video and K LUTs)	9.06	0.25	17	31.4	89.7	55	74.07
Video specification	4K@30fps	4K@24fps	4K@30fps	4K@30fps	2K@60fps	4K@30fps	4K@30fps
Ref Buffer size	8 KB	–	–	6 KB	7.6 KB	1 KB	0.8 KB

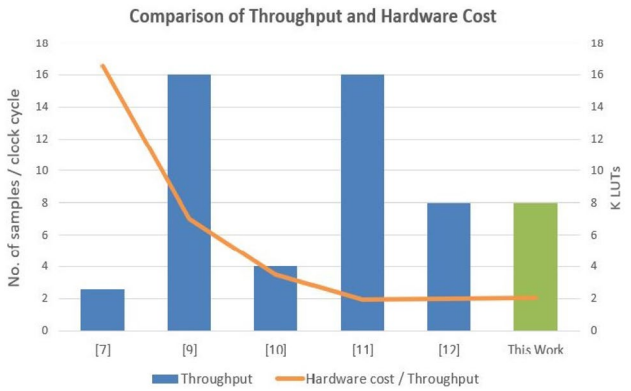

Fig. 11 Graphical representation of hardware cost and throughput of the proposed design with other similar works

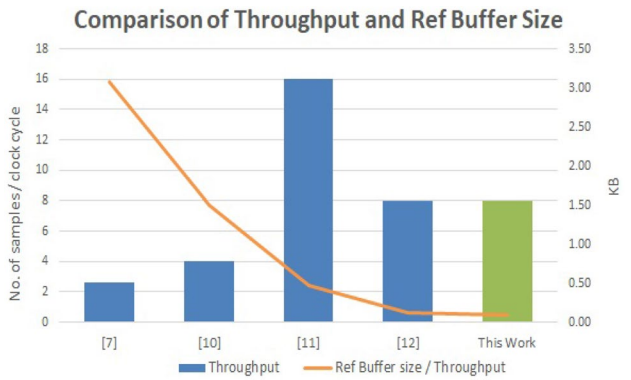

Fig. 12 Graphical representation of comparison of reference buffer memory size used in the proposed design with other similar works

5 Conclusion

In this work, we have proposed an improved efficient intra prediction architecture to support DCT/DST engine for the HEVC encoder. The engine is configured to always perform row-wise predictions and ensures that no extra intermediate buffer memory of 8K is required to store the predicted samples. For horizontal predictions, the architecture saves 128 clock cycles for a 32×32 PU, 512 clock cycles for an LCU and nearly 1.03 million clock cycles in the case of a single 4 K picture frame. This design supports all PU block sizes and 35 directional modes to process 64 samples (8 samples each with 8-bit depth) in parallel in each clock cycle. The proposed architecture is a combination of parallel and pipelined framework that achieves higher throughput using just 0.8 KB reference memory. A dedicated arithmetic unit ensures the reuse of multipliers to enhance the engine's hardware efficiency. Our design uses only 40 DSP slices. Using a 28 nm technology FPGA board operating at 150 MHz, a throughput of 64 samples is achieved with the hardware cost of 16.2 K LUTs and 5.7 K registers to support real time 4 K video encoding.

The proposed architecture is designed to operate in congruence with other modules of the HEVC encoder. As a part of the future work, we plan to investigate and use the proposed approach to design an efficient intra mode encoding engine with an optimal balance between complexity and performance and at the same time maintain a high throughput.

References

1. Cisco Systems, Inc, Cisco Annual Internet Report (2018–2023) White Paper, March, 2020, https://www.cisco.com/c/en/us/solut

ions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf (2020), Accessed on 29 April 2021
2. Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the High Efficiency Video Coding (HEVC) Standard. IEEE Trans. Circ. Syst. Video Technol. **22**(12), 1649–1668 (2012)
3. Sze, V., Budagavi, M., Sullivan, G.: J, High efficiency video coding (HEVC), 40. Springer, New York (2014)
4. Bossen, F., Bross, B., Suhring, K., Flynn, D.: HEVC Complex Implement. Anal. **22**, 1685–1696 (2012)
5. Wien, M.: High efficiency video coding, Coding Tools and specification, 40, 133–160. Springer, New York (2015)
6. Li, F., Shi, G., Wu, F.: An Efficient VLSI Architecture for 4× 4 Intra Prediction in the High Efficiency Video Coding (HEVC) standard, 18th IEEE International Conference on Image Processing, 373–376. Belgium, Brussels (2011)
7. Abeydeera, M., Karunaratne, M., Karunaratne, G., Silva, D.K., Pasqual, A.: 4K real-time HEVC decoder on an FPGA. IEEE Trans. Circ. Syst. Video Technol. **26**, 236–249 (2016)
8. Amish, F., Bourennane, E.-B.: Fully pipelined real time hardware solution for high efficiency video coding (HEVC) intra prediction. J. Syst. Architect. **64**, 133–147 (2016)
9. Choudhury, R., Rangababu, P.: Design and implementation of mixed parallel and dataflow architecture for intra-prediction hardware in HEVC ecoder. Int. Sympos. VLSI Des. Test **742–750**, Springer (2017)
10. Min, B., Xu, Z., Cheung, R.C.C.: A fully Pipelined Hardware Architecture for intra prediction of HEVC. IEEE Trans. Circ. Syst. Video Technol. **27**, 2702–2713 (2017)
11. Ding, D., Wang, S., Liu, Z., Yuan, Q., Real-Time, H.: 265/HEVC Intra Encoding with a Configurable Architecture on FPGA Platform. Chin. J. Electron. **28**, 1008–1017 (2019)
12. Poola, L., Aparna, P.: A Mixed Parallel and Pipelined Efficient Architecture for Intra Prediction Scheme in HEVC. IETE Tech. Rev., 1–13 (2020)
13. Fan, Y., Tang, G., Zeng, X.: A Compact 32-Pixel TU-Oriented and SRAM-Free Intra Prediction VLSI Architecture for HEVC Decoder. IEEE Access **7**, 149097–149104 (2019)
14. Poola, L., Aparna, P.: Efficient Architectures for Planar and DC modes of Intra Prediction in HEVC, 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN), 148–153, Noida, India (2020)
15. Palomino, D., Sampaio, F., Agostini, L., Bampi, S., Susin, A.: A memory aware and multiplierless VLSI architecture for the complete Intra Prediction of the HEVC emerging standard, 19th IEEE International Conference on Image Processing, 201–204. Orlando, FL, USA (2012)
16. Kalali, E., Adibelli, Y., Hamzaoglu, I.: In: A high performance and low energy intra prediction hardware for High Efficiency Video Coding, pp. 719–722. , Oslo, Norway (2012)
17. Abramowski, A., Pastuszak, G.: A double-path intra prediction architecture for the hardware H.265/HEVC encoder, IEEE 17th International Symposium on Design and Diagnostics of Electronic Circuits Systems, 27-32, Warsaw, Poland (2014)
18. Azgin, H., Mert, A.C., Kalali, E., Hamzaoglu, I.: An efficient FPGA implementation of HEVC intra prediction, 2018 IEEE International Conference on Consumer Electronics (ICCE), 1–5. Las Vegas, NV, USA (2018)
19. Zhou, N., Ding, D., Yu, L.: On hardware architecture and processing order of HEVC intra prediction module, IEEE International Conference on Picture Coding Symposium (PCS), 101–104. CA, USA, San Jose (2013)
20. Minezawa, A., Sugimoto, K., Sekiguchi, S.-I: An improved intra vertical and horizontal prediction. In: Joint Collaborative Team on Video Coding (JCT-VC), Document JCTVC-F172, Torino (2011)

Lakshmi Poola has received her B.E. degree in Electronics and Communication from UVCE, India, and MTech. in VLSI Design and Embedded Systems from NMAM Institute of Technology, India. She is currently working towards a Ph.D. in the National Institute of Technology Karnataka (NITK), Surathkal, India. Her research interests include VLSI design, algorithms, FPGA implementation of VLSI architectures for multimedia signal processing.

P. Aparna has received her B.E degree in Electronics and Communication Engineering from NMAM Institute of Technology, India, MTech Degree in Digital Electronics and Advanced Communication from NITK, India, and Ph.D. from NITK, India. Since 2008 she is serving the Department of Electronics and Communication Engineering, NITK, Surathkal, India, as an Assistant Professor. Her major research interests include Multimedia Signal Processing, Image/Video compression, Audio and Speech Processing, Biomedical Signal Processing, etc.