

# VLSI Architecture Design for H.264/AVC Intra-frame Video Encoding

HUANG-CHIH KUO<sup>1</sup> YOUN-LONG LIN<sup>1,a)</sup>

Received: March 8, 2013, Released: August 5, 2013

**Abstract:** Intra-frame encoding is useful for many video applications such as security surveillance, digital cinema, and video conferencing because it supports random access to every video frame for easy editing and has low computational complexity that results in low hardware cost. H.264/AVC, which is the most popular video coding standard today, also defines novel intra-coding tools to achieve high compression performance at the expense of significantly increased computational complexity. We present a VLSI design for H.264/AVC intra-frame encoder. The paper summarizes several novel approaches to alleviate the performance bottleneck caused by the long data dependency loop among  $4 \times 4$  luma blocks, integrate a high-performance hardwired CABAC entropy encoder, and apply a clock-gating technique to reduce power consumption. Synthesized with a TSMC 130 nm CMOS cell library, our design requires 194.1 K gates at 108 MHz and consumes 19.8 mW to encode 1080p ( $1920 \times 1088$ ) video sequences at 30 frames per second (fps). It also delivers the same video quality as the H.264/AVC reference software. We suggest a figure of merit called *Design Efficiency* for fair comparison of different works. Experimental results show that the proposed design is more efficient than prior arts.

**Keywords:** video coding, H.264/AVC, VLSI, hardware architecture

## 1. Introduction

Intra-frame encoding compresses video pixels using only the neighboring pixels in the same frame. Therefore, every frame is processed, accessed, and transmitted individually. Intra-frame encoding is suitable for several video applications such as security surveillance, digital cinema, and video conferencing because (1) each frame can be processed independently, so as to simplify post-production and editing, (2) the loss of some frames due to unstable network environment will not affect the processing of other frames, and (3) its low computational complexity and memory bandwidth requirements result in low hardware cost. For example, both Apple and Panasonic employ H.264/AVC intra-frame encoders in their high-definition (HD) video products [3], [87].

Another application of intra-frame encoder is embedded frame compression to reduce bus traffic. For example, a video decoder writes the decoded display frames into an external memory, and a display controller then fetches these frames from the memory for the display device. As the video resolution increases, the amount of bus traffic increases, which usually degrades the system performance. We can employ compression algorithms to compress (decompress) video frames before (after) being stored (fetched) to (from) the external memory. Since the compression algorithm is integrated with a video decoder, it should have very low computational complexity to avoid degrading the system performance.

Motion JPEG [37] and Motion JPEG2000 [39] have been

widely used in the past for such applications as surveillance systems, digital cameras, and digital cinema production because of their low hardware cost. However, their compression efficiency can be considerably improved for high-definition applications.

H.264/MPEG-4 Advanced Video Coding (H.264/AVC) [41], [43], jointly developed by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG), is the most popular video coding standard today. Compared with previous video coding standards, H.264/AVC achieves at least 40% bit-rate saving [46] by adopting several advanced coding tools. These include variable-block-size motion estimation with quarter-pixel accuracy [107] and multiple reference frames [108], integer discrete cosine transform (DCT), in-loop deblocking filter (DF) [67], and context-based adaptive binary arithmetic coding (CABAC) [77].

H.264/AVC proposes two novel intra-frame coding tools, namely multiple-mode intra prediction and Lagrangian-based intra mode decision [109], [112], to utilize the spatial redundancy among frame pixels. It can achieve better compression performance than the previous intra-frame encoding standards MJPEG and MJPEG2000 [4], [80] at the expense of much higher computational complexity. Therefore, numerous previous studies propose algorithms and hardware architectures to improve the performance, decrease the complexity, and accelerate the encoding process of H.264/AVC intra-frame encoders.

We present an efficient H.264/AVC intra-frame encoder. We first analyze the H.264/AVC intra-frame encoding process to uncover its design challenge and propose three approaches to alleviate the performance bottleneck. We then propose several efficient

<sup>1</sup> Department of Computer Science, National Tsing Hua University, Hsin-Chu, Taiwan

<sup>a)</sup> ylin@cs.nthu.edu.tw

hardware architectures and on-chip memory subsystems. Finally, we employ a clock-gating scheme to reduce its power consumption. By integrating a high-performance CABAC encoding engine [22], our proposed intra-frame encoder can encode 1080p (1920 × 1088) video at 30 frames per second (fps) and can deliver the same video quality as that with the H.264/AVC reference software JM 11.0 [49]. That is we introduce no quality degradation.

The rest of this paper is organized as follows. In Section 2, we describe H.264/AVC Intra-frame encoding. In Section 3, we discuss challenges we faced when designing a high-performance H.264/AVC encoder. Section 4 survey related works. Then we set our design target in Section 5. After describing our three proposed technologies in Section 6, we present our architecture and system in Section 7. Section 8 shows some experimental results and compares the proposed encoder with several state of the art encoders. Finally, we draw conclusion and point to possible directions for future research in Section 9.

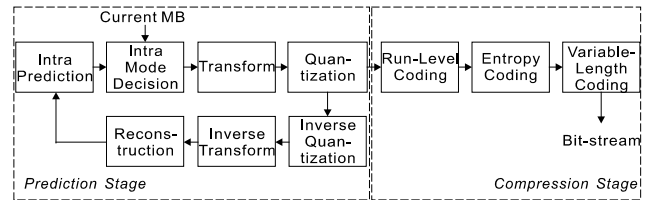
## 2. H.264/AVC Intra-frame Encoding

An H.264/AVC encoder contains an inter-frame encoding path and an intra-frame encoding path to exploit temporal and spatial redundancy, respectively, for compressing video data. Inter-frame encoding takes advantage of temporal redundancy among successive video frames. For every macroblock in the currently processing frame, it finds the best-matched macroblocks in either previous or later frames and then encodes the corresponding residuals. Intra-frame encoding, on the other hand, compresses video data by exploiting spatial redundancy. No data from neighboring frames is referenced. Instead, it encodes every macroblock using previously encoded-then-decoded neighboring macroblocks in the same frame.

**Figure 1** depicts the H.264/AVC intra-frame encoding process. We group all functional units into a prediction and a compression stage.

The intra prediction unit predicts a macroblock by referring to its neighboring reconstructed macroblocks. All prediction pixels are calculated using the reconstructed pixels of the neighboring 4 × 4 blocks or 16 × 16 macroblocks. For the luma component, a 16 × 16 block can be divided into sixteen 4 × 4 blocks and each encoded using a prediction mode, or treated as a single 16 × 16 block and encoded using a prediction mode. **Figure 2** lists all intra prediction modes. There are nine intra luma 4 × 4 modes for each of the sixteen 4 × 4 luma blocks and four intra luma 16 × 16 modes for a 16 × 16 luma block. The chroma component contains one 8 × 8 Cb block and one 8 × 8 Cr block. There are four intra chroma 8 × 8 modes for each of two 8 × 8 chroma blocks.

The intra mode decision unit needs all prediction pixels by all modes and the original pixels to perform mode selection. It computes the cost of each prediction mode according to distortion and bit-rate. Equations (1)–(4) below show how to obtain mode costs for a macroblock using the low-complexity mode decision algorithm proposed in the reference software JM 11.0. For the luma component, the decision unit calculates the distortion and bit-rate. For the chroma component, it simply compares the distortions of each chroma 8 × 8 mode.



**Fig. 1** H.264/AVC intra-frame encoding flow.

$$cost_{luma} = distortion + \begin{cases} bitrate, & \text{for the intra } 4 \times 4 \text{ mode} \\ 0, & \text{for } 4 \text{ intra } 16 \times 16 \text{ modes} \end{cases} \quad (1)$$

$$cost_{chroma} = distortion \quad (2)$$

$$bitrate = 4 \times \text{roundoff}(6 \times lamda + 0.4999) + \sum_{i=0}^{15} mpm_{costi} \quad (3)$$

$$mpm_{costi} = \begin{cases} 4 \times lamda, & \text{if best mode} \neq mpm \\ 0, & \text{if best mode} = mpm \end{cases} \quad (4)$$

Either the sum of absolute difference (SAD) or the sum of absolute transformed difference (SATD) can be used for estimating distortion. According to Reference [30], SATD gives better estimations than SAD does. To estimate bit-rate cost, the reference software JM 11.0 defines the estimated cost for the “intra4 × 4” mode, which indicates that the luma block is encoded using sixteen intra luma 4 × 4 modes, as shown in Eq. (3). The Lagrangian parameter  $\lambda$  is derived from the quantization parameter (QP) and is used to make trade-off between distortion and bit-rate. Moreover, JM 11.0 calculates the most probable mode cost (mpm\_cost), as shown in Eq. (4), for each 4 × 4 luma block. The most probable mode of a 4 × 4 luma block is decided by the best prediction modes of its upper- and left-neighboring 4 × 4 blocks. If the best prediction mode of the current 4 × 4 block is identical to its most probable mode, the intra mode decision unit sends a 1-bit flag instead of the mode number to the decoder. The intra mode decision unit selects the mode with the minimum cost. If its luma component is encoded using one of four intra luma 16 × 16 modes, we call the macroblock an “intra16 × 16” macroblock. Otherwise, we call it an “intra4 × 4” macroblock. The intra mode decision unit also generates the residuals in addition to the mode information.

The transform unit performs 4 × 4 integer Discrete Cosine Transform (DCT) and 2 × 2/4 × 4 Hadamard transform on spatial domain residuals to obtain frequency domain coefficients. The quantization unit then quantizes the coefficients to remove insignificant information and outputs the quantized coefficients to both the run-level coding unit and the inverse quantization unit. The inverse quantization unit and inverse transform unit then translate the coefficients back into residuals which are then output to the reconstruction unit. The reconstruction unit adds up the residuals and the prediction pixels of the chosen intra prediction mode to generate the reconstructed pixels. The run-level coding unit receives the quantized coefficients and outputs them to the entropy coding unit in a zig-zag scan order in which the low-frequency coefficients, which contain the significant data, appear before the high-frequency ones, which generally contain

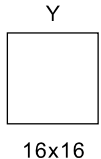

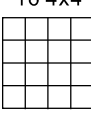




Component	Block Size	Prediction Modes	Abbreviation
 Y 16x16	1 16x16 	0:vertical 1:horizontal 2:DC 3:plane	L16_VER L16_HOR L16_DC L16_PLANE
	16 4x4 	0:vertical 1:horizontal 2:DC 3:diagonal down-left 4:diagonal down-right 5:vertical-right 6:horizontal-down 7:vertical-left 8:horizontal-up	L4_VER L4_HOR L4_DC L4_DDL L4_DDR L4_VR L4_HD L4_VL L4_HU
 Cb 8x8	1 8x8 	0:DC 1:horizontal 2:vertical 3:plane	CB8_DC CB8_HOR CB8_VER CB8_PLANE
 Cr 8x8	1 8x8 	0:DC 1:horizontal 2:vertical 3:plane	CR8_DC CR8_HOR CR8_VER CR8_PLANE

Fig. 2 Intra prediction modes.

only zero or very small values. The entropy coding unit performs lossless compression to encode the remaining quantized coefficients into a macroblock-level bitstream. H.264/AVC employs two types of entropy coding algorithms: context-based adaptive binary arithmetic coding (CABAC) and context-based adaptive variable length coding (CAVLC). According to our experiments, the CABAC algorithm is 13–19% better than the CAVLC algorithm with respect to bit-rate saving for 1080p video. The former achieves good compression ratio by adapting probability estimations based on local statistics and using arithmetic coding instead of variable-length coding. However, the CABAC algorithm involves a large number of bit-level operations, which makes it more complex than CAVLC. After entropy coding, the variable-length coding unit encodes configuration parameters into a bitstream header and packs them together with the macroblock-level bitstream into an H.264/AVC bitstream for system output. Moreover, the H.264/AVC standard defines a high profile to support super high definition video. The high profile introduces nine intra luma 8 × 8 modes and an integer 8 × 8 DCT to exploit the spatial redundancy. The prediction equations and encoding processes of intra luma 8 × 8 modes are very similar to those of intra luma 4 × 4 modes.

### 3. Design Challenge

The seven functional units in the prediction stage form a loop to encode sixteen 4 × 4 luma blocks. The intra prediction unit uses the reconstructed pixels to predict the subsequent 4×4 block. Therefore, we have to go through the encoding loop sixteen times. **Figure 3** shows the processing order and data dependency among all 4 × 4 luma blocks in the intra luma 4 × 4 prediction. For example, the arrows pointing to block 10 indicate that we predict block 10 by referring to blocks 4, 7, and 9.

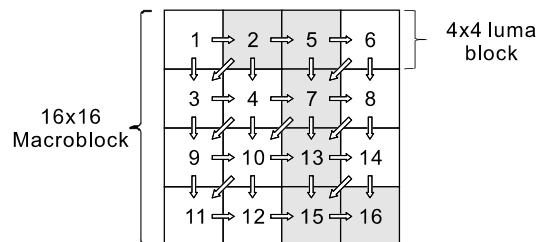
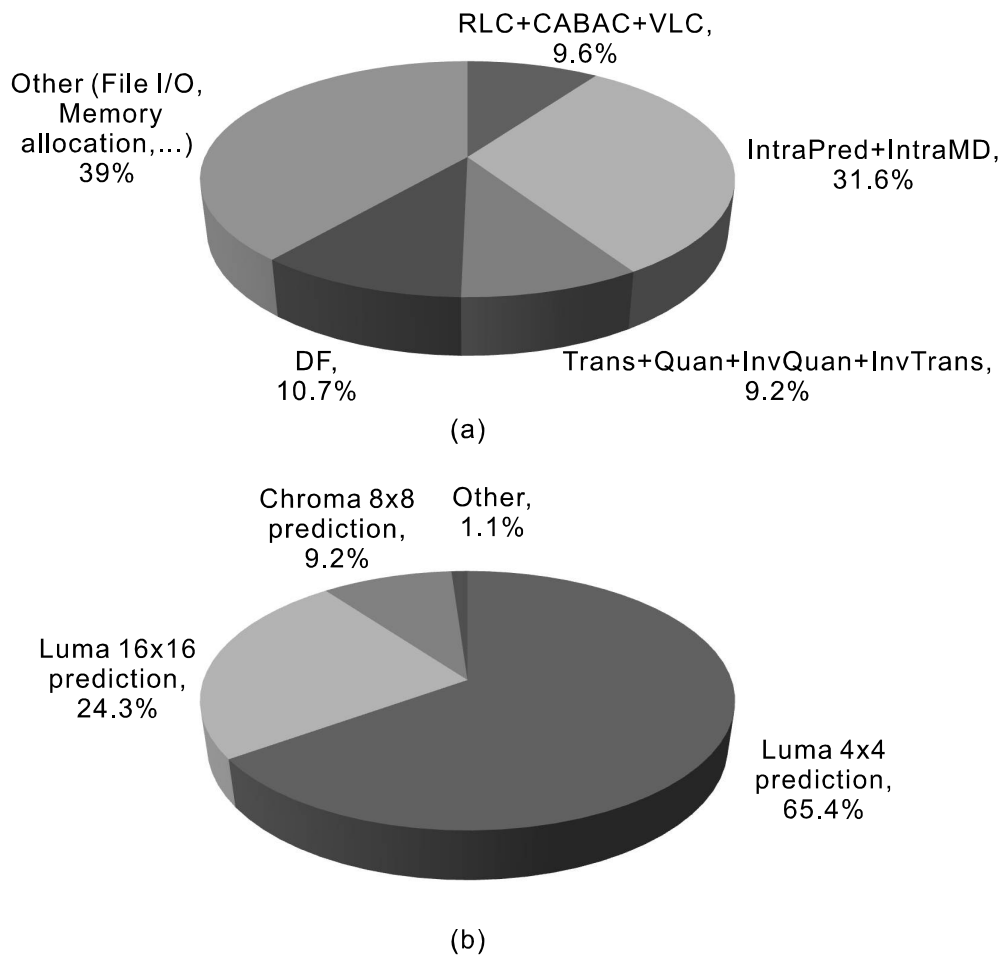


Fig. 3 Processing order and data dependency of 4 × 4 luma blocks during intra 4 × 4 prediction.

The data dependency loop dictates the performance of an intra-frame encoder. **Figure 4** (a) shows the profiling results of intra-frame encoding using JM 11.0. Intra prediction and intra mode decision together account for about 31% of total encoding time. Figure 4 (b) further breaks down the time spent on various components by intra prediction and intra mode decision. The results show that luma 4 × 4 prediction accounts for about 65% of encoding time for these two units, or equivalently 20% of total encoding time.

### 4. Related Works

Since 2005, numerous papers, patents, and books about H.264/AVC have been published because of its high compression performance. A lot of previous works have proposed high-performance hardware architectures for each functional modules such as the intra prediction unit [55], [84], [88], [90], [93], transform and quantization unit [19], [74], [85], and entropy coding unit [14], [27], [65] in an H.264/AVC encoder. All of these works achieve superior performance because they optimize the performance of each unit without considering the integration issues. For example, several previous works that propose the hardware architectures for the intra prediction unit [55], [84], [93] focus on



Platform: Intel Xeon E5530 16 core@2.4 GHz + 64GB DRAM  
 OS: CentOS release 4.6 Compiler: GCC 3.4.6  
 Encoder: JM 11.0 Video: Sunflower 1080p 100 frames (GOP: I, QP: 24)

Fig. 4 Profiling results of (a) intra-frame encoding and (b) intra prediction and intra mode decision units.

reducing the hardware resources and computation time for generating prediction pixels instead of alleviating the performance bottleneck caused by the data dependency loop among luma  $4 \times 4$  blocks. Therefore, it is not feasible to integrate these hardware modules into a high-performance H.264/AVC encoder.

Several previous works have proposed hardware architectures for H.264/AVC intra-frame encoders. Huang et al. [30] propose the first hardwired H.264/AVC intra-frame encoder. Instead of implementing the original Hadamard-based algorithm, they employ a DCT-based mode decision algorithm to save hardware resources and bus traffic. They also propose an interleaved I4MB/I16MB prediction schedule, which decomposes the intra luma  $16 \times 16$  prediction into the  $4 \times 4$  block level and inserts it right after the intra luma  $4 \times 4$  prediction of the same block to eliminate the bubble cycles caused by the data dependency loop. Their two-stage macroblock pipelined design can encode SD ( $720 \times 480$ ) video at 30 fps when running at 54 MHz. It costs 85 K gates using a TSMC 250 nm cell library. Cheng, Ku, and Chang [20] analyze the probability distribution of all intra luma  $16 \times 16$  modes for several CIF ( $352 \times 288$ ) video sequences. According to the analysis, they remove the plan mode prediction to reduce both the hardware cost and processing time of the intra prediction engine

at the expense of video quality degradation. They also propose an enhanced SATD cost function to compensate for the quality loss. The encoder can encode 720p ( $1280 \times 720$ ) video at 30 fps when running at 117 MHz. It costs 92.6 K gates using a UMC 180 nm cell library. It also observes on the average 0.06% bit-rate increase and 0.09 dB SNR drop in the luma component as compared to JM 8.6 [47]. The authors further expand their encoder to be a codec [50] by integrating a CAVLC decoder.

Li et al. [68] propose a modified three-step algorithm, which generates prediction pixels for only 7 out of 9 intra  $4 \times 4$  modes, to reduce the prediction time of a  $4 \times 4$  luma block. They also increase the throughput of intra prediction, intra mode decision, and transform engines by processing 8 pixels per cycle instead of the traditional 4 pixels per cycle [20], [30], [50]. Their encoder can encode 720p video at 30 fps when running at 61 MHz. It costs 72 K gates using a UMC 180 nm cell library. It also observes on the average 0.68% bit-rate increase as compared to JM 8.6. Lin et al. [69] propose an encoder that employs all fast algorithms and architectures they have previously proposed [20], [68]. The design can encode 1080p video at 30 fps when running at 140 MHz. It costs 94.7 K gates using a TSMC 130 nm cell library. It also observes on the average 1.03% bit-rate increase and 0.11 dB PSNR

drop as compared to JM 8.6.

Chang et al. [7] propose a quality-scalable intra-frame encoder. They propose three fast algorithms: a context-correlation search algorithm (CC-SA), a probability-context-correlation search algorithm (PCC-SA), and a quarter-macroblock search algorithm (QMB-SA), to speed up the encoding loop. In CC-SA and PCC-SA, they reduce the number of prediction modes to 5 and 4 for  $4 \times 4$  luma blocks, respectively. In QMB-SA, they only generate the prediction pixels and compute the cost of the left-top  $4 \times 4$  chroma block, instead of the whole  $8 \times 8$  chroma block, to decide the best chroma mode. The proposed encoder supports three quality levels, which are QS0 that predicts all intra prediction modes, QS1 that adopts CC-SA and QMB-SA, and QS2 that adopts PCC-SA and QMB-SA. The QS2 encoder can encode 720p video at 30 fps when running at 70 MHz. It costs about 169.6 K gates (including a deblocking filter) using a TSMC 130 nm cell library. It suffers from on the average 6.63% bit-rate increase and 0.06 dB PSNR drop as compared to JM 9.3 [48].

The authors of Ref. [7] further improve the encoder to support 1080p video. The improved encoder [12] can encode 1080p video at 30 fps when running at 152 MHz. It costs about 91 K gates when synthesized with a 90 nm cell library. It observes 7.5% bit-rate increase and 0.05 dB PSNR drop in the luma component as compared to JM 9.3.

Some previous works propose hardware architectures for the intra-frame encoding loop (i.e., functional blocks in the prediction stage). Suh, Park, and Cho [98] propose the first hardware architecture for the intra-frame encoding loop. They employ a  $4 \times 4$  pred\_pel calculator and a  $16 \times 16/8 \times 8$  pred\_pel calculator to generate luma  $4 \times 4$  and luma  $16 \times 16$ /chroma  $8 \times 8$  prediction pixels at the same time. Moreover, they propose a Hadamard coefficient pre-calculation method to reduce bubble cycles of the proposed transform engine. The design can encode SD video at 42 fps when running at 54 MHz. It costs 192 K gates using a Hynix 350 nm cell library.

Lin et al. [73] propose a hardware architecture for the intra-frame encoding loop with a transform-based intra prediction algorithm. The proposed algorithm first employs a conventional DCT to translate all luma and chroma blocks into coefficients and then decides the number of prediction modes for each block according to the sum of coefficients. By using the proposed algorithm, the largest number of prediction modes of a luma  $4 \times 4$ , a luma  $16 \times 16$  block, and a chroma  $8 \times 8$  block are 6, 2, and 2, respectively. The design can encode 16 SIF ( $1408 \times 960$ ) video at 30 fps when running at 99 MHz. It cost 89 K gates using a 180 nm cell library. Moreover, by adding a buffer to store transformed data, the design can encode 16 SIF video at 30 fps when running at 89 MHz at the expense of 32 K gates area overhead. It observes on the average 0.76% bit-rate increase and 0.08 dB PSNR drop as compared to JM 12.0.

Diniz et al. [24] propose a high throughput hardware architecture for the intra-frame encoding loop. To speed up the encoding process, they increase the throughput of all functional units in the prediction stage to 16-fold. The authors also employ a homogeneity mode decision algorithm to speed up the mode selection. The design can encode 1080p video at 61 fps when running at

150 MHz. It costs 201.8 K gates using a TSMC 180 nm library. It observes on the average 12% bit-rate increase and 0.28 dB PSNR drop as compared to JM 14.2.

He et al. [35] first propose a hardware architecture for the intra-frame encoding loop and then integrate it with a high-performance CABAC encoder to be an intra-frame encoder [117]. The proposed encoder aims to support super high-definition video. Therefore, the proposed design supports the intra prediction modes defined in the H.264/AVC high profile excluding the intra luma  $4 \times 4$  modes. To speed up the encoding process, the authors propose a coarse-to-fine mode decision algorithm to select a best macroblock mode, a best luma  $16 \times 16$  mode, and four best luma  $8 \times 8$  mode candidates, an interlaced block reordering to schedule the prediction orders of luma  $8 \times 8$  blocks, and a probability-based reconstruction to speed up the reconstruction process. The proposed designs for the intra-frame encoding loop and the CABAC can encode 4320p ( $7680 \times 4320$ ) video at 60 fps when running at 260 and 330 MHz, respectively. The encoder costs 678.8 K gates using a 65 nm library. It observes on the average 1.8% bit-rate increase as compared to JM 17.0. Most of the previous works that propose intra-frame encoders further propose hardware architectures for H.264/AVC full encoders [8], [9], [10], [11], [13], [28], [72] by integrating motion estimation and motion compensation modules into their intra-frame encoders. These works focus on reducing the huge memory bandwidth requirements caused by reference frame access and high computational complexity caused by motion estimation and pixel interpolation.

In summary, most related works employ heuristic algorithms to simplify the computation and save hardware cost at the expense of quality loss, bit-rate increase, or both. This is not what the H.264/AVC standardization effort is intended for. In this study, we aim to propose an efficient intra-frame encoder that fully preserves the prediction correctness.

## 5. Design Targets

Intra-frame encoders are widely adopted in many video applications such as digital cinema, digital photography, and surveillance. In this study, we aim to design an efficient H.264/AVC intra-frame encoder that is suitable for surveillance systems. Therefore, in the following paragraphs, we consider the desirable attributes of a surveillance camera in terms of resolution, frame rate, compression performance, and hardware cost.

### 5.1 Video Resolution

Many technologies of image processing such as facial recognition, license plate recognition, and object tracking are essential for a surveillance system. Therefore, pixels per foot (PPF) and horizontal field of view (HFOV) are two important parameters to observe in designing a surveillance camera [83], [101]. PPF indicates what level of detail can be seen in an image. Taking Fig. 5 that is originally illustrated in Reference [83] as an example, if the size of a human face is one square foot in a video frame, PPF should be larger than 40 to perform facial recognition. PPF is obtained by dividing the horizontal resolution of video by HFOV. HFOV indicates the size of a real-world scene that can be viewed



Fig. 5 Various PPF for a human face.

through the camera, which is determined by the distance from the camera and the focal length of the lens.

Although increasing the video resolution of a camera directly increases the PPF when using the same HFOV, it also increases the storage space and transmission bandwidth. Today, industry is moving toward a resolution of 1080p because of two reasons. First, using a camera that has a 16 : 9 aspect ratio is more efficient than using a camera that has a 4 : 3 aspect ratio. For example, the video size and the HFOV of a 1080p camera are 66% and 94%, respectively, compared to that of a 3.1 megapixels (2048 × 1536) camera. Second, the HFOV of a 1080p camera is large enough for most surveillance scenes when using a 40 PPF. Therefore, we target the resolution of the proposed encoder to 1080p in this study.

5.2 Frame Rate

The frame rate of a video encoder varies according to the target applications. For example, the lowest acceptable frame rate for viewing a movie is 24 fps and the standard frame rate for a TV program is 25 or 30 fps. If the frame rate is slower than 10, the human eyes perceive only a series of still images with no motion. The increase in the frame rate implies increases in both storage space and transmission bandwidth. According to industry guidelines [83], [101], using 30 fps is enough for most scenes, including high-stake areas such as casinos and banks.

5.3 Compression Performance

There are many configurations for an H.264/AVC intra-frame encoder to support various applications. For example, the H.264/AVC standard proposes two entropy algorithms, as mentioned earlier. Since the higher compression ratio implies less storage space and lower transmission bandwidth, we integrate a high-performance CABAC engine [22] into the proposed encoder in order to support H.264/AVC main profile at Level 4.1. The H.264/AVC standard defines several levels to support various video resolutions. Levels 4 and 4.1 support 1080p video at 30 fps with maximum output bit-rate of 20 Mbps and 50 Mbps, respectively. Video quality is also important for a surveillance system. We aim to implement the SATD-based mode decision and predict all modes defined in the main profile to deliver the same video quality as JM 11.0.

5.4 Hardware Cost

Area and power consumption are the most important figures of merit to measure the hardware cost. Nowadays, reducing power consumption becomes increasingly urgent as the computational complexity of consumer electronics products increases. Since power consumption is proportional to the working frequency, we aim to minimize the required frequency of the proposed encoder.

Table 1 Performance target of the proposed encoder.

Profile	Main
Level	4.1
Entropy Coding	CABAC
Cost Function	SATD
Prediction Modes	All in main profile

Moreover, we employ a module-level clock-gating scheme to further reduce dynamic power dissipation. Hardware area is directly relevant to the chip size and power consumption. However, there is a trade-off between hardware cost and compression performance. The fast algorithms that are proposed to speed up the encoding process usually reduce the hardware cost but video quality suffers. Therefore, we aim to find a good balance between hardware cost and compression performance. Table 1 summarizes the performance targets of our proposed encoder.

6. Performance Enhancement Approaches

We describe three approaches: a hardware utilization optimized schedule (HUOS), a resource sharing among skew modes (RSSM), and a referenced column first reconstruction (RCFR), to enhance the performance of our encoder. Each is presented in a subsection below.

6.1 Hardware Utilization Optimized Schedule

Figure 6 shows the proposed processing schedule for 16 4 × 4 luma blocks in our intra luma 4 × 4 prediction. Each 4 × 4 luma block is numbered according to the order shown in Fig. 3. The arrows indicate the data dependency. Compared with the processing order employed by the reference software JM, the proposed order theoretically saves (16 – 12)/16 = 25% of processing time to encode 4 × 4 luma blocks while preserving the prediction accuracy.

The proposed engines for the intra prediction and the intra mode decision units together take 7 cycles to generate prediction pixels for nine intra luma 4 × 4 modes. However, the proposed engines for the remaining five units, which are transform, quantization, inverse quantization, inverse transform, and reconstruction, need 12 cycles to process a 4 × 4 block. Therefore, the proposed engines for the intra prediction and the intra mode decision units must either use a buffer to store the residuals of the predicted 4 × 4 luma blocks or wait for the remaining five units. To alleviate this bottleneck, we decompose all computations of the intra luma 16 × 16 prediction into the 4 × 4 block level and predict each 4 × 4 block right after its luma 4 × 4 prediction to reduce the idle cycles. We also schedule the intra chroma 8 × 8 prediction into the reconstruction bubble cycles in the first, second, fifth, and seventh iterations to increase the hardware utilization. Although the functional units in the prediction stage process 4 × 4 blocks out of order, the proposed engine for the RLC unit will rearrange

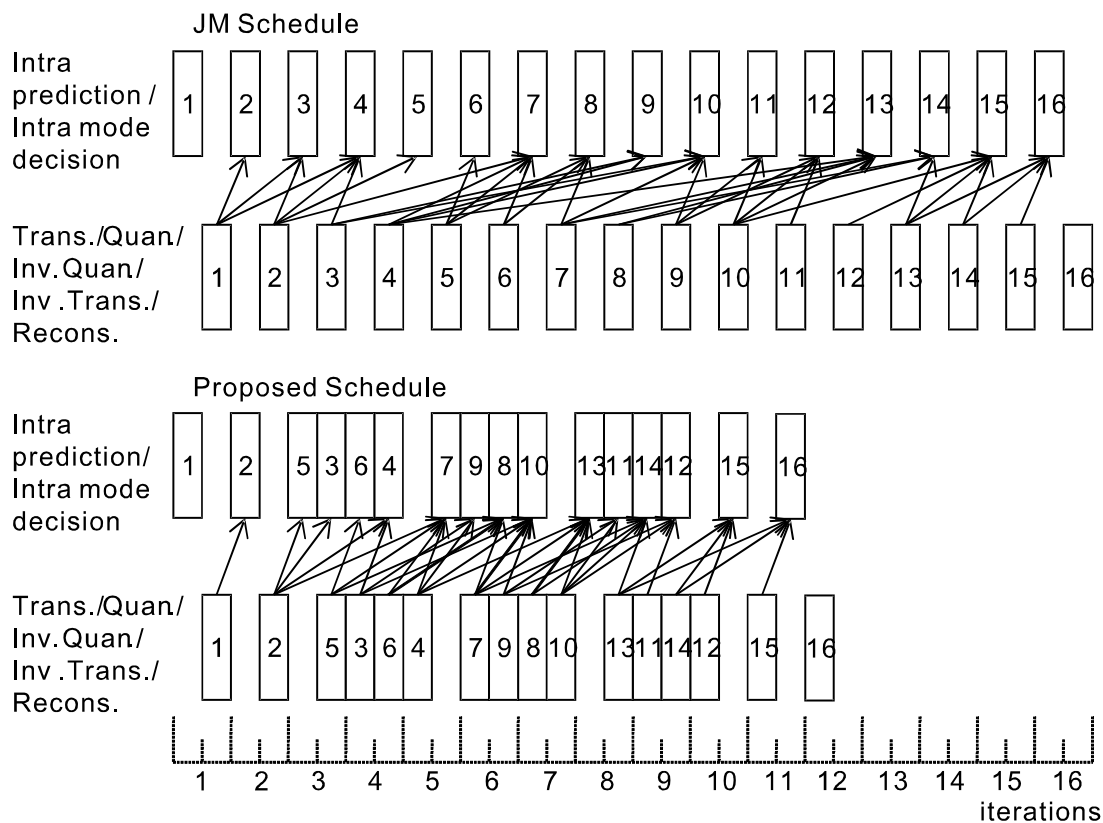


Fig. 6 Proposed processing schedule for  $4 \times 4$  blocks during the intra luma  $4 \times 4$  prediction.

coefficients back to order for the CABAC encoder such that our overall design is compliant with the H.264/AVC standard.

### 6.2 Resource Sharing Among Skew Modes

We define Pixel-Level Parallelism (PLP) as the number of pixels calculated at a time in the intra prediction unit. Increasing PLP directly decreases the total prediction time, but increases hardware cost. We increase the PLP for predicting intra luma  $4 \times 4$  modes to  $16 \text{ (pixels/mode)} \times 2 \text{ (modes)} = 32$  pixels. To eliminate the increased hardware area, we show a filter scheduling scheme for better resource sharing. Here, a filter is a basic hardware unit used for calculating prediction pixels.

We first categorize all intra luma  $4 \times 4$  modes into two groups: *bypass-mean* and *skew*. The *bypass-mean* category contains  $4 \times 4$  horizontal, vertical and DC modes. The *skew* category contains all the remaining modes. We use a four-pixel adder and several multiplexers to generate prediction pixels for the modes in the *bypass-mean* category. Modes in the *skew* category are the most complex. They need both three-tap and two-tap filtering operations to generate prediction pixels. However, there are great similarities among the filtering equations of different modes in the *skew* category. Table 2 shows computation equations required by each mode in the *skew* category. It is a modified form of the table originally proposed by Reference [30]. Each row represents a computation equation and the number of prediction pixels that use the equation in each mode. The computation equations labeled as “T3”, “T2”, and “Bypass” denote a three-tap filter, a two-tap filter, and a bypass operation, respectively. For example, four prediction pixels in the DDL mode and two in the VL mode share

the same computation equation, T3eq9.

If we predict one mode per cycle, it takes ten filters to generate all prediction pixels in nine cycles. On the other hand, the most parallel design takes 23 filters to predict nine modes in a cycle. To make a good trade-off between hardware resources and prediction time, we can share the filters that perform the same computation operation in the same cycle for two or more similar modes. For example, the HD and HU modes are highly similar according to Table 2. If we predict them in the same cycle, only 11 filters are required. Table 3 shows the number of filters needed for each mode combination if we aim to predict 2 modes per cycle. There are 15 combinations in total. According to Table 3, the proposed design takes only 12 filters to predict 6 modes in 3 cycles by adopting the second combination. Therefore, the proposed design takes 5 cycles to output (1) HD and HU modes, (2) DDR and VR modes, (3) DDL and VL modes, (4) horizontal and vertical modes, and (5) DC modes sequentially.

### 6.3 Referenced Column First Reconstruction

We can start predicting a  $4 \times 4$  luma block only after its referencing pixels have been reconstructed. For example, if the intra prediction unit processes sixteen  $4 \times 4$  luma blocks according to the original processing order, as shown in Fig. 6, it can start processing the even-numbered blocks after those in the right-most column of the left-hand side blocks (i.e., odd-numbered blocks) have been reconstructed. After the reconstruction of the right-most column of a  $4 \times 4$  luma block, we can start the prediction of the subsequent  $4 \times 4$  luma block. To generate reference pixels of a  $4 \times 4$  block’s right-most column first, before other columns,

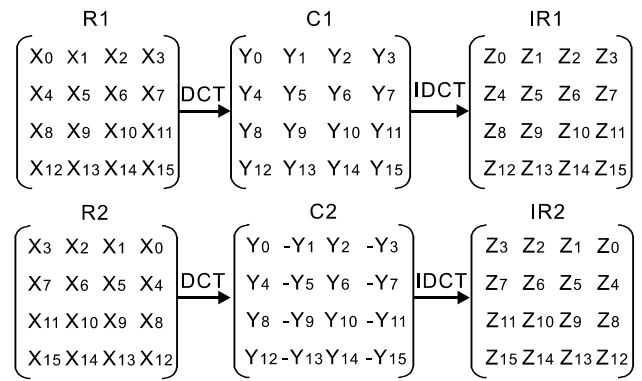
**Table 2** Operation table for modes in skew category.

Equation	DDL	DDR	VR	HD	VL	HU
T3eq0						2
T3eq1		1		1		2
T3eq2		2	1	2		1
T3eq4		3	1	2		
T3eq5		4	2	2		
T3eq6	1	3	2	1	1	
T3eq7	2	2	2	1	2	
T3eq8	3	1	1		2	
T3eq9	4				2	
T3eq10	3				1	
T3eq11	2					
T3eq12	1					
T2eq0				1		2
T2eq1				2		2
T2eq2				2		1
T2eq3				2		
T2eq4			2			
T2eq5			2		1	
T2eq6			2		2	
T2eq7			1		2	
T2eq8					2	
T2eq9					1	
Bypass						6

**Table 3** Number of filter needed for parallel prediction.

Combinations	Cycle 1	Cycle 2	Cycle 3	Needed Filters
1	HD, HU	VR, VL	DDR, DDL	15
2	HD, HU	VR, DDR	VL, DDL	12
3	HD, HU	VR, DDL	VL, DDR	15
4	HD, VR	HU, VL	DDR, DDL	16
5	HD, VR	HU, DDR	VL, DDL	15
6	HD, VR	HU, DDL	VL, DDR	15
7	HD, VL	HU, VR	DDR, DDL	19
8	HD, VL	HU, DDR	VR, DDL	19
9	HD, VL	HU, DDL	VR, DDR	19
10	HD, DDR	HU, VR	VL, DDL	15
11	HD, DDR	HU, VL	VR, DDL	16
12	HD, DDR	HU, DDL	VR, VL	15
13	HD, DDL	HU, VR	VL, DDR	16
14	HD, DDL	HU, VL	VR, DDR	16
15	HD, DDL	HU, VR	VL, DDR	16

one method is to design a special IDCT architecture that calculates reference pixels first. However, this method complicates the hardware design. We observe that the reference pixels can be calculated first by taking advantage of the symmetry between DCT and IDCT matrices. As illustrated in Fig. 7, if we perform DCT on matrix R1 to obtain output matrix C1, and subsequently perform IDCT on C1, the resulting matrix will be IR1. For another matrix R2, which is a reverse-column-presentation of R1, the final result after performing DCT and then IDCT is IR2. We



**Fig. 7** Consecutive DCT and IDCT operations on reverse-column-presentation leads to reverse-column-presentation results.

can see that IR2 is a reverse-column-presentation of IR1.

This result is useful for the encoder design. First, our proposed transform architecture outputs 8 inverse-transformed residuals for a 4 × 4 luma block per cycle. Second, since H.264/AVC’s quantization and inverse quantization units perform one-to-one multiplications, by sending the residuals to the proposed transform architecture in a reverse order, as shown in Fig. 7, the referenced column of a 4 × 4 luma block will be generated before other columns, and hence, can be reconstructed first. The proposed approach can eliminate the bubble cycles before processing six gray-colored 4 × 4 luma blocks, as illustrated in Fig. 6.

## 7. Proposed Architecture and System

We first depict the pipeline schedule of the proposed encoder. We then present the AMBA interface with which our design interacts with the system. Finally, we describe the detailed design of our encoder core.

### 7.1 Pipeline Schedule

We employ a two-stage pipelined architecture in our design. The first stage contains all functional units for prediction while the second stage contains compression units. Figure 8 shows the timing diagram of our design. All 4 × 4 luma blocks are numbered as that in Fig. 6. Figure 8(a) shows the first stage timing diagrams for predicting a macroblock. If the current macroblock is an intra16 × 16 macroblock, the proposed intra prediction and intra mode decision engines (IntraPred and IntraMD) spend 227 cycles to generate all prediction pixels and perform intra luma 4 × 4 mode decision. The proposed transform coding engine (TransQuan) that integrates the transform, quantization, inverse transform, and inverse quantization units and the proposed Recons engine then spend 134 (228 to 361 in Fig. 8(a)) and 72 (362 to 433 in Fig. 8(a)) cycles to encode the best intra luma 16 × 16 and intra chroma 8 × 8 modes, respectively. If the current macroblock is an intra4 × 4 macroblock, the proposed design spends 227 + 72 = 299 cycles to encode it.

Our IntraPred and IntraMD engines take 7 cycles (e.g., 1 to 7 in Fig. 8(a)) to generate all predictions of all 4 × 4 modes for a 4 × 4 block. They then take 3 cycles to calculate the costs of all intra luma 16 × 16 modes. After the IntraMD engine selects the best intra luma 4 × 4 mode, the proposed TransQuan engine takes 10 cycles (e.g., 25 to 34 in Fig. 8(a)) to process a 4 × 4 luma



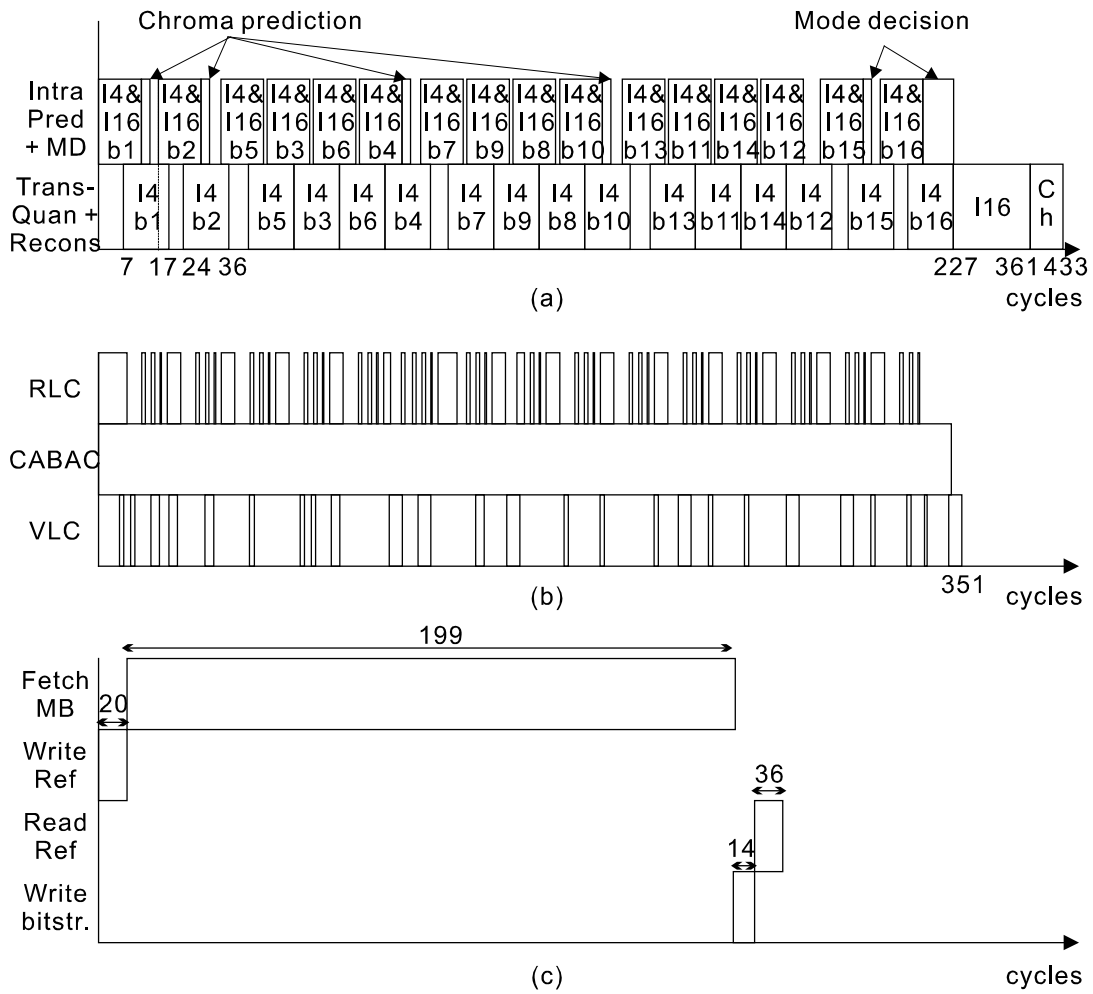


Fig. 8 Timing diagrams for (a) encoding a MB in the 1st stage, (b) encoding a MB in the 2nd stage, and (c) accessing the external memory.

block. The proposed Recons engine then spends two cycles (e.g., 35 to 36 in Fig. 8 (a)) to process a  $4 \times 4$  luma block. Adopting the proposed RCFR approach, the proposed IntraPred engine can start to predict the  $4 \times 4$  luma blocks 2, 5, 7, 13, 15, and 16 without waiting for the reconstruction of their previous blocks. For example, the IntraPred engine can start to predict block 2 at the 18th cycle as shown in Fig. 8 (a) while the Recons engine is still processing block 1.

There are three engines, run-level coding (RLC), context-adaptive binary arithmetic coding (CABAC), and variable-length coding (VLC), in the second pipelined stage. The processing time of these engines vary according to the quantization parameter (QP) and video content. Figure 8 (b) illustrates the second-stage timing diagram for our encoder to encode a macroblock in the 1080p video “rush\_hour” when QP is 16. Empirically, the average number of cycles taken by our encoder in the second pipelined stage to encode one macroblock of 1080p video is 157.1, 233.6, 358.8, and 465.4 when the QP is 28, 22, 16, and 10, respectively. We employ several memory access units (MAUs) to handle the data transfer between the encoder and the external memory. The MAUs prefetch the next macroblock, fetch and write the referenced data for the current macroblock, and access the encoded bitstream from and to the external memory. Figure 8 (c) gives the average number of cycles these MAUs needed for a macroblock.

### 7.2 AMBA Bus Interface

Figure 9 shows the top-level block diagram of our proposed intra-frame encoder. It consists of two main parts: an encoder core containing all units that perform the encoding functions and an AMBA bus interface for data transfer between the encoder core and the off-chip memory.

We implement several AMBA-compliant modules to deal with external-memory access and software/hardware communication. The AMBA interface contains a command receiver, an MAU arbiter, and three MAUs. The command receiver is connected to an AMBA slave. Users send to it configuration parameters including QP and slice types in a memory address format. The command receiver extracts from its input configuration parameters and sends them to the MainCtrl engine inside the encoder core. The MAU arbiter is connected to an AMBA master. It decides on which MAUs can access the external memory. There are three MAUs: a macroblock (MB) MAU, a top-row (TR) MAU, and a bitstream (BIT) MAU. The MB MAU reads in source macroblocks from the external memory and stores them in the local memory for the IntraMD engine. We access the local memory in a ping-pong fashion. When the IntraMD engine reads the pixels of the current macroblock, the MB MAU prefetches the subsequent macroblock concurrently. The MB MAU can support both interlaced and progressive video formats. Each of IntraPred, IntraMD, and CABAC

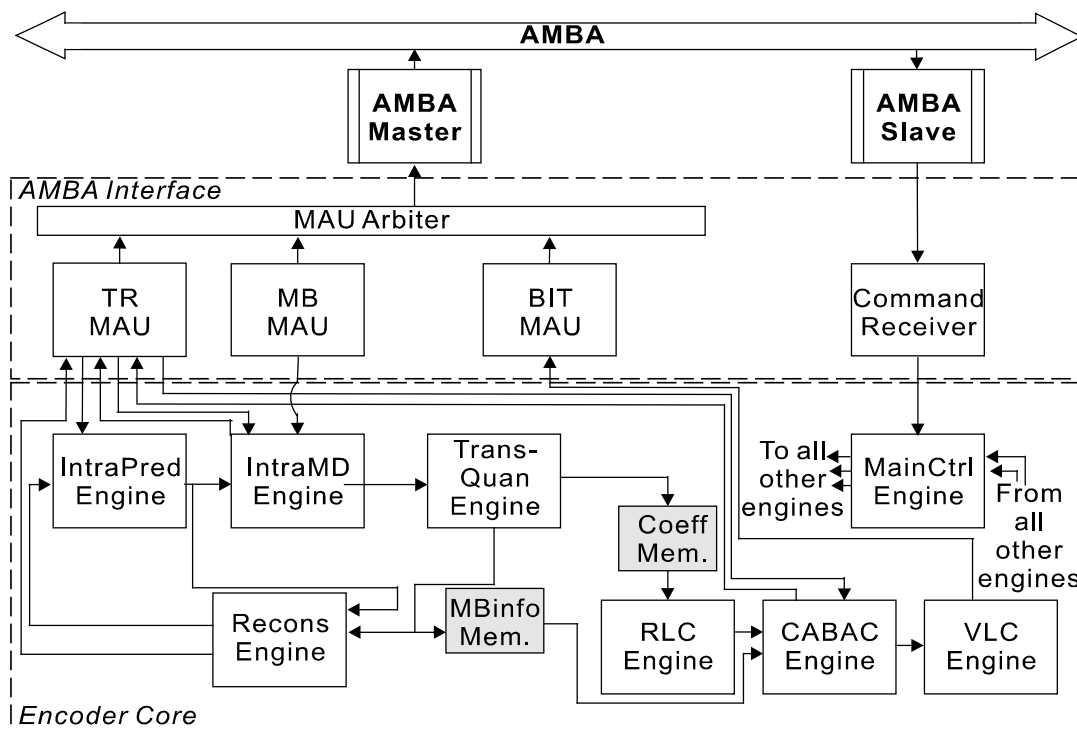


Fig. 9 Top level block diagram of the proposed encoder.

engines accesses to the upper neighboring macroblock for encoding the current one. The TR MAU reads and writes the referenced data for these engines. It provides a local buffer to store the referenced data needed by a macroblock.

The BIT MAU writes the compressed H.264/AVC bitstream to the external memory. Because the VLC engine generates different numbers of bits each time and outputs them aperiodically, the system performance will be degraded if we directly write its output to the external memory. Therefore, we employ a buffer queue to store the compressed bitstream temporarily and write the bits to the external memory when the queue length reaches 32 bytes.

### 7.3 Encoder Core Implementation

#### 7.3.1 IntraPred and IntraMD Engines

Figure 10 shows the block diagram of our proposed IntraPred engine. It consists of an I4 engine that generates the prediction pixels of intra luma  $4 \times 4$  modes, and an I16 engine that generates the prediction pixels of intra luma  $16 \times 16$  modes. All these engines contain an HV pixel generator and a DC pixel generator. The HV pixel generators generate prediction pixels for all vertical and horizontal modes by bypassing the reference pixels. The I4 DC and the I16 DC pixel generators use one and two four-pixel adder trees, respectively, to compute the luma  $4 \times 4$  and luma  $16 \times 16$  values.

The plane pixel generator calculates prediction pixels of three plane modes. We calculate 16 prediction pixels for every  $4 \times 4$  block using the plane parameters and the seed pixels. There are 16 seed pixels for the luma component and 8 seed pixels for the chroma components. Instead of buffering all 24 seed pixels, we use only 6 major seed pixels to calculate the remaining 18 seed pixels. After obtaining all seed pixels, the pixel calculator generates 16 prediction pixels in parallel. Moreover, we reuse the I16

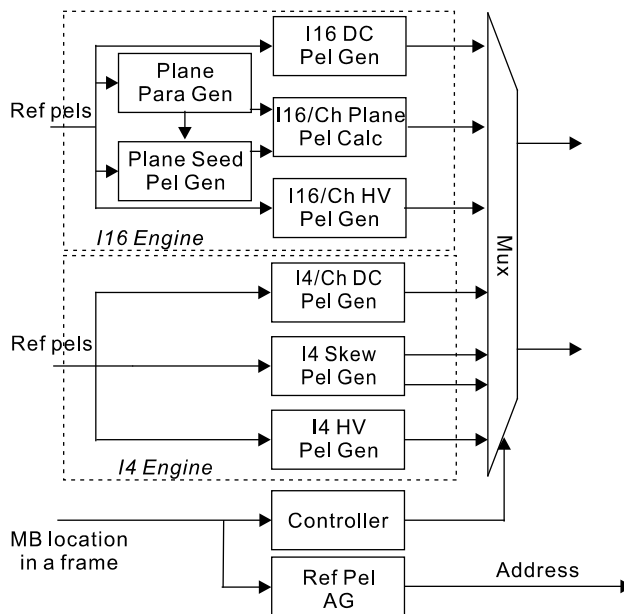


Fig. 10 Block diagram of the proposed IntraPred engine.

HV pixel generator, I16 plane pixel generator, and I4 DC pixel generator to generate the prediction pixels of the chroma modes.

Because the IntraPred engine generates prediction pixels for two modes for each of the  $4 \times 4$  blocks simultaneously, the IntraMD engine is equipped with two cost calculators, as shown in Fig. 11. Each cost calculator employs an HT engine to perform Hadamard transform. To calculate bit-rate cost, the  $\lambda$  generator uses QP to look-up a QPQUANT table to obtain  $\lambda$ . Moreover, Eq. (4) can be further simplified for hardware cost saving as  $\lambda$  is always an integer in JM 11.0. The proposed IntraMD engine also calculates the residuals of the best intra mode. Instead of using a large memory to store all prediction pixels, the proposed

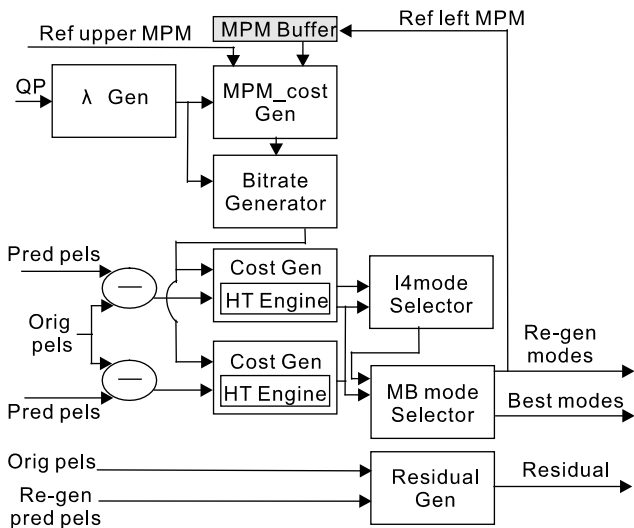


Fig. 11 Block diagram of the proposed IntraMD engine.

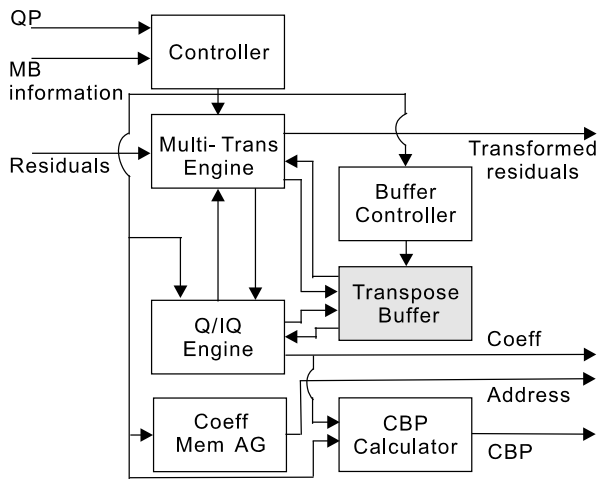


Fig. 12 Block diagram of the proposed TransQuan engine.

IntraPred engine will re-generate the prediction pixels of the best mode.

### 7.3.2 TransQuan and Recons Engines

Figure 12 shows the top-level block diagram of our proposed transform coding engine. Its main components are a 1-D multi-transform engine and a combined Q/IQ engine. Because of the 1-D multi-transform engine, we rely on a Transpose Buffer to carry out a 2-D transform.

The proposed multi-transform engine, as depicted in Fig. 13, has eight types of inputs and can carry out four types of operations. Because matrices for HT and IHT are the same, and those for DCT and IDCT are highly similar, we unify all transforms into a single datapath and use multiplexers to confirm the datapath for the needed type of transform.

The proposed Q/IQ engine, as depicted in Fig. 14, uses a table lookup method instead of division to calculate  $QP/6$  and  $QP\%6$ . Moreover, adders and multipliers are shared for quantization and inverse quantization operations. The TransQuan engine writes to two memories, *MBinfo* that holds macroblock parameters, and *Coeff* that holds transformed coefficients, for the CABAC engine.

The Recons engine contains only adders and multiplexers. It sums up the transformed residuals and the prediction pixels of the best mode to generate reconstructed pixels. For luma  $4 \times 4$  prediction, it writes the resulting reconstructed pixels to the IntraPred engine. On the other hand, for luma  $16 \times 16$  and chroma  $8 \times 8$  predictions, it writes the resulting reconstructed pixels to the top-row MAU.

### 7.3.3 RLC, CABAC, and VLC Engines

The proposed Run-Level Coding (RLC) engine reads coefficients and performs zig-zag scans for the CABAC engine. It also calculates parameters *sig\_coeff\_flag* by using every bit of a coefficient, *last\_sig\_coeff\_flag* by using *sig\_coeff\_flag* as the index to check for the location of the last non-zero coefficient,

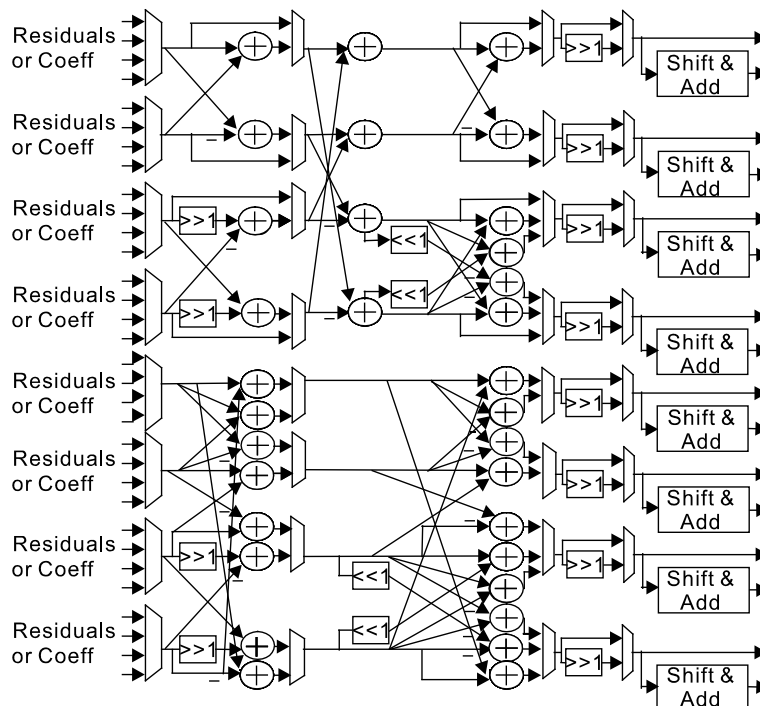


Fig. 13 Block diagram of the proposed multi-transform engine.

and *coded\_block\_flag* by using 16 *sig\_coeff\_flags*. We have integrated a high-performance CABAC engine [22] into the proposed encoder. It features a six-stage pipelined binary arithmetic engine that achieves high throughput by encoding up to 8 bins per cycle. The VLC engine is simple and implemented with several FSMs, combinational circuits, and adders.

**7.3.4 MainCtrl Engine**

The MainCtrl engine receives the encoder parameters from the command receiver and sends appropriate control signals to each engine in the encoder core. By running the encoder at 108 MHz, its power consumption for encoding 1080p video is 29.7 mW. Still, we observe that several engines have idle cycles during the encoding process. We apply a module-level clock-gating technique to further reduce the power consumption. For example, when the TransQuan engine is processing chroma blocks or the best intra luma 16 × 16 mode, the IntraMD engine is idle, and hence, can be turned off. Moreover, because the number of cycles with which the two stages execute are different, we turn off the engines in the faster stage when it finishes. The MainCtrl engine generates clock enable signals for all other engines of the encoder. Table 5 shows the power consumption before and after optimization of each engine for encoding 1080p video. Except for the MainCtrl engine, every engine achieved power reduction

after optimization. The MainCtrl engine consumes 0.0081 mW more power (from 0.2421 mW to 0.2502 mW) because of added circuits for clock-gating control. This optimization saves 33% of power at the expense of only 0.04% area overhead.

**7.3.5 Internal Buffers**

Table 4 shows the configuration of each internal buffer our encoder need. The column “S/D” denotes single-port or dual-port.

The *MB\_Info* memory holds parameters defined for an intra macroblock such as *mb\_type*, *rem\_intra\_pre\_mode*, and *pre\_intra\_pre\_mode\_flag*. The *Coeff* memory contains four banks, and each bank contains 28 entries that store eight coefficients. We first categorize these four banks into two groups. Because the TransQuan and the RLC engines belong to distinct pipelined stages, when the former writes the coefficients of the (N + 1)th macroblock into one of the two groups, the later reads the coefficients of the Nth macroblock from another one. Moreover, because the TransQuan engine generates up to eight coefficients at a time for a 4 × 4 block, we store the coefficients into the same memory location of two memory banks in a group. The *CMB* memory contains four banks, and each bank contains 24 entries that store eight source pixels. We also categorize the four banks into two groups. When the proposed encoder reads the current macroblock from one of the two groups, the propose MB MAU pre-fetches the successive macroblock to another one. Moreover, we store a 4 × 4 block into two banks because the TransQuan engine can process up to eight residuals at a time. The *Neighbor\_SE* memory, *Context* memory, and *CABAC ROM* serve as local buffers for the CABAC engine to access referenced neighboring data and context information.

**8. Experimental Results**

We have implemented the proposed architecture in Verilog [36] HDL and synthesized it targeted toward a TSMC 130 nm [102] CMOS cell library. Our design requires 193.4 K gates (162.3 K gates for the encoder core and 31.1 K gates for the AMBA interface) and consumes 19.8 mW of power as detailed in Table 5. Figure 15 shows its layout.

We use several 30-frame video sequences of different resolu-

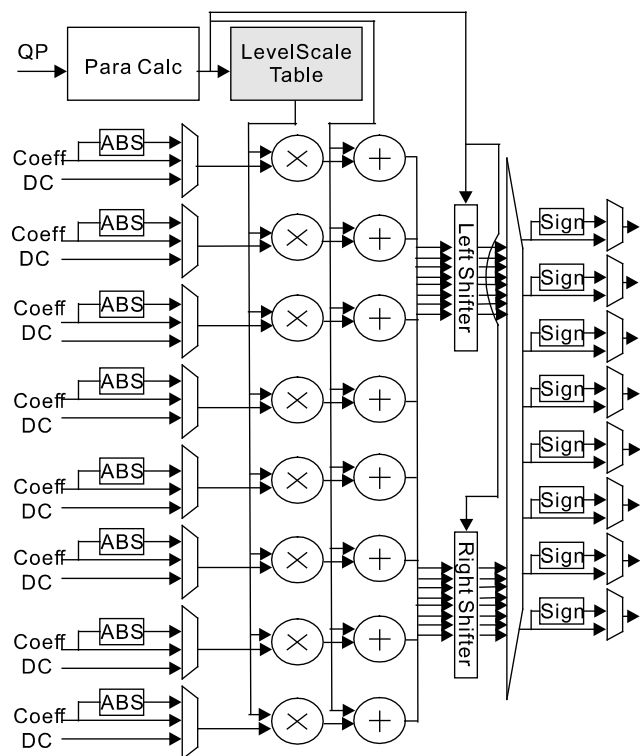


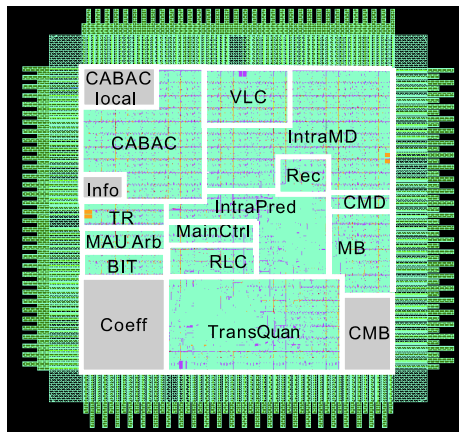
Fig. 14 Block diagram of the proposed Q/IQ engine.

Table 4 Configuration of the internal buffers.

Name	S/D	Bits x Entries x Banks	Total Bytes
MB_Info	S	12 x 20 x 1	30
Coeff	S	128 x 28 x 4	1792
CMB	S	64 x 24 x 4	768
Neighbor_SE	D	8 x 28 x 1	28
Context_up	D	7 x 100 x 2	175
Context_bot	D	7 x 100 x 2	175
CABAC_ROM	-	16 x 1600 x 1	3200

Table 5 Power consumption and implementation results of each engine.

	Engine	Power Consumption (mW)		Gate Count (K Gates)	
		Before Optimization	After Optimization		
Encoder	1st	IntraPred	1.11	0.86	19.3
		IntraMD	6.44	5.05	37.8
		TransQuan	6.14	5.95	44.3
		Recons	0.45	0.45	0.6
	2nd	MainCtr	0.24	0.25	1.5
		RLC	0.28	0.26	3.8
		CABAC	6.77	2.93	47.5
		VLC	0.66	0.29	7.5
Subtotal		22.09	16.04	162.3	
Amba	MB MAU	1.59	1.24	8.6	
	TR MAU	2.44	1.09	7.6	
	BIT MAU	3.07	1.18	13.0	
	CMD Rec	0.48	0.22	1.6	
	MAU Arb	0.03	0.03	0.3	
	Subtotal		7.61	3.76	31.1
Total		29.7	19.8	193.4	



Library	TSMC 130nm 1P8M CMOS
Core & I/O Voltage	1.2V/3.3V
Core Size (mm <sup>2</sup> )	1.4 x 1.4
Frequency (MHz) (for 1080p)	108

Fig. 15 Layout of the proposed encoder.

Table 6 Average processing cycles per MB for various QP values.

1080p sequences	QP28	QP22	QP16	QP10
Pedestrian_area	384.2	373.8	399.5	493.9
riverbed	314.8	308.4	400.4	566.6
Rush_hour	389.7	379.2	385.8	475.4
Station2	367.3	349.6	390.8	530.2
sunflower	371.0	352.2	358.3	452.8
tractor	367.4	350.1	390.8	528.2

Table 7 Average processing cycles and video quality gain over encoders with CAVLC.

Sequences	Level 4		Level 4.1	
	AvgCycle	$\Delta$ PSNR (dB)	AvgCycle	$\Delta$ PSNR (dB)
Pedestrian_area	352.94	1.8	372.9	1.55
riverbed	340.19	2.16	307.2	1.14
Rush_hour	353.72	1.1	375.6	1.13
Station2	346.16	1.37	348.7	1.16
sunflower	354.26	2.15	348.8	1.38
tractor	339.86	1.68	331.2	1.26
Average	347.86	1.71	347.4	1.27

tions to test the proposed intra-frame encoder. The evaluation environment includes a 32-bit bus model and a 16-MByte SDRAM model [81]. Table 6 lists the average number of cycles our encoder takes to encode a macroblock. The maximum number of cycles of the first pipelined stage is 433 whereas that of the second pipelined stage varies, since the processing time of the CABAC engine varies depending on the video content and the values of QP. Therefore, we further analyze the performance requirements needed to support the H.264/AVC Levels 4 and 4.1.

According to Sayood's work [91] and our experimental results, a CABAC unit on the average compresses 1.8 bins to 1 bit. Therefore, the throughput of a CABAC unit has to be  $1.8 \times 50 = 90$  MBins/s to support Level 4.1 encoding. Our CABAC engine can compress 1 bin per cycle in the worst case. Running at 108 MHz, its throughput will be at least 108 MBins/s. Therefore, our encoder can support the case of Level 4.1 (i.e., 30 fps 1080p video with output bit-rate 50 Mbps) when running at 108 MHz. Table 7 shows the average number of cycles the proposed encoder takes to encode a macroblock. It also shows the video quality gain

Table 8 Performance of the proposed encoder and previous works.

Work	Type	Max Resol	Frame Rate	Freq	Lib (nm)	Gate Count (K)	Local Mem (KBs)	Entropy Coder
[30]	Coder	SD	30	54	250	85	1.7K	CAVLC
[98]	Loop	SD	30	38	350	192.4	3.4	-
[20]	Coder	720p	30	117	180	92.6	2	CAVLC
[68]	Coder	720p	30	61	180	72	1.6	CAVLC
[7]	Coder	720p	30	68	130	170	4.4	CAVLC
[69]	Coder	1080p	30	138	130	94.7	1.8	CAVLC
[73]	Loop	16SIF	30	99	180	89	-	-
[73]	Loop	16SIF	30	89	180	120.7	-	-
[12]	Coder	1080p	30	152	90	91	2.9	CAVLC
[24]	Loop	1080p	61	150	180	201.8	5.5	-
[117]	Coder	4320p	60	257	65	678.8	27.1	CABAC
Proposed	Coder	1080p	30	108	130	193.4	6	CABAC

Table 9 Design efficiency of the proposed encoder and previous works.

Work	Mbs per frame	FPS	PMR	Freq (MHz)	Library (nm)	Total Gates (K)	DE
[73]	5280	30	0.65	99	180	93	11.1
[73]	5280	30	0.65	89	180	124.7	9.2
[24]	8160	61	1	150	180	256.8	12.9
Proposed180	8160	30	1	108	180	139.8	16.2
[69]	8160	30	0.77	138	130	102.7	13.2
Proposed130	8160	30	1	108	130	124.9	18.1
[12]	8160	30	0.59	152	90	62.2	15.2
Proposed90	8160	30	1	108	90	123.1	18.4

$\Delta$ PSNR of the proposed encoder for each test sequence. By integrating the CABAC encoder, under the same bit-rate constraint our design can deliver better video quality than that of the reference software JM 11.0 with CAVLC.

Table 8 shows the performance of the proposed intra-frame encoder compared to previous works including intra-frame encoders (coder) and intra-frame encoding loop designs (loop). The performance of Reference [7] and Reference [12] are obtained using the quality level QS2. The proposed design can encode 30 fps 1080p video when running at 108 MHz and consuming only 19.8 mW. It delivers the same video quality as that of JM 11.0 with CABAC and achieves 16% bit-rate saving as compared to JM 11.0 with CAVLC.

To further compare the performance of the proposed intra-frame encoder with the previous works that target to real-time encode full HD video, we suggest a metric "Design Efficiency (DE)" that is defined in Eq. (5) to evaluate the prediction engines. Moreover, we synthesize the proposed design targeted toward both a TSMC 180 nm and a UMC 90 nm [104] CMOS cell libraries to make a fair comparison. Since several previous works have proposed fast algorithms to improve the performance at the expense of video quality loss, we use the metric "Predicted Mode Ratio (PMR)" that is obtained by dividing the number of predicted modes per macroblock of each design by the number of prediction modes defined in the H.264/AVC main profile. The total gates is the sum of the function unit gates and the equivalent memory gates obtained by using an Artisan memory generator [5]. Table 9 shows the comparison results. The "Proposed180", "Proposed130", and "Proposed90" denote the proposed design synthesized, targeted toward a TSMC 180 nm, a TSMC 130 nm, and a UMC 90 nm CMOS cell libraries, respectively. We also add a 384-byte local memory that is used to

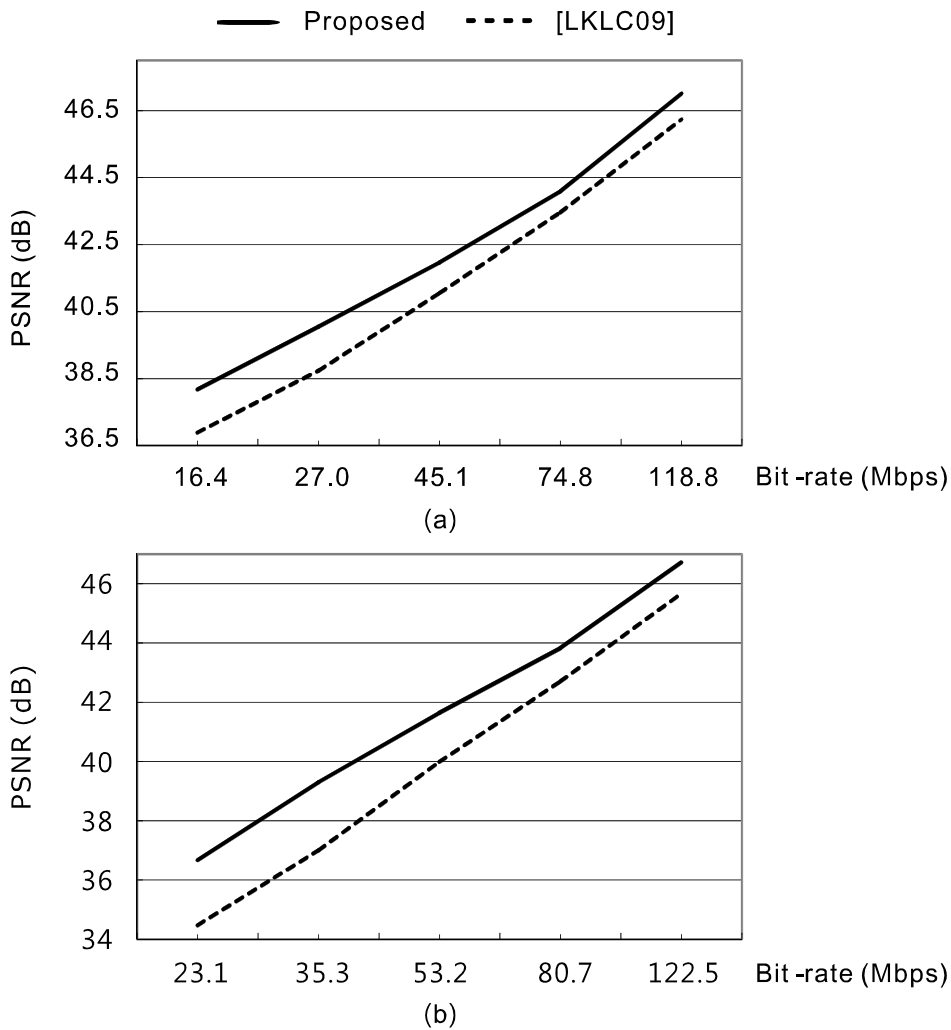


Fig. 16 Rate-distortion curves of the proposed encoder and Ref. [69] for (a) station 2 and (b) tractor.

store a source macroblock to Ref. [73] since it does not treat this memory as a local one whereas other works do. The proposed design outperforms all previous works by making better trade-off between hardware cost and compression performance. Figure 16 further shows the rate-distortion curves of our proposed encoder and Ref. [69] for the two test sequences reported in Ref. [69].

$$DE = \frac{MBs_{per\_frame} \times Frame_{rate} \times PMR}{Total_{gates} \times Frequency} \quad (5)$$

## 9. Conclusion and Future Work

Both video encoders and decoders have to access a lot of memory data to process a macroblock. The high memory bandwidth requirement usually degrades the system performance. According to our profiling, accessing display frames dramatically decreases the memory efficiency of a video decoder.

H.264/AVC intra-frame encoding delivers excellent coding performance at the expense of high computational complexity. In the encoding process, there exists a long data dependency loop among  $4 \times 4$  luma blocks that dominates the system performance. We summary three performance enhancement approaches to improve the encoding performance. The proposed HUOS approach speeds up the encoding process and improves the utilization of the hardware engines in our first pipelined stage to alleviate the per-

formance bottleneck caused by the data dependency loop without causing any quality degradation. The proposed RSSM approach increases the throughput of the intra prediction engine with minimized area overhead. The proposed RCFR approach eliminates the bubble cycles between prediction of two  $4 \times 4$  luma blocks.

By adopting the approaches, the encoder achieves better video quality and bit-rate saving compared with all previous works that target to encode full HD video. Moreover, we apply a module-level clock-gating technique to achieve very low power consumption and integrate an AMBA interface to make SOC integration easy. Experimental results show that our design can encode 30 fps 1080p video when running at 108 MHz and consumes only 19.8 mW. It delivers the same video quality as that of JM 11.0 with CABAC and achieves 16% bit-rate saving as compared to JM 11.0 with CAVLC.

As video resolution increases, the bus traffic of accessing reference and display frames increases. In the future, we plan to improve the proposed compression algorithm to compress reference frames since accessing reference frames also consumes a lot of bus cycles. The proposed algorithm can be modified to use the two-dimensional information to predict pixels in a block and compress each  $M \times N$  block independently. In addition to video coding, the compression algorithms can be employed in

other applications such as test-data compression, medical image compression, and audio coding. The proposed algorithm can be modified according to the characteristics of these data to support these applications. We also plan to further reduce the power consumption of the proposed encoder, since battery life is very important for mobile devices, and video encoders usually consume a lot of power. In addition to the module-level clock-gating method, several efficient methods such as register-level clock-gating, dynamic voltage and frequency scaling (DVFS), and power gating can be employed to further reduce power consumption. Moreover, the demand for ultra high resolution video increases as technology advances. A video codec will be required to support up to Quad Full High Definition (QFHD,  $3840 \times 2160$ ) resolutions in the near future. Therefore, a novel video coding standard, High Efficiency Video Coding (HEVC) [111], is under development by the Joint Collaborative Team on Video Coding (JCT-VT) of ITU-T VCEG and ISO/IEC MPEG now. The HEVC aims to achieve 50% more compression than H.264/AVC and supports up to 4320p ( $7680 \times 4320$ ) video resolution. The proposed design can be used as a reference model for designing high-performance hardware architectures for HEVC.

## References

- [1] Ausavarungrun, R., Chang, K.W., Subramanian, L., Loh, G.H. and Mutlu, O.: Staged memory scheduling: Achieving high performance and scalability in heterogeneous systems, *Proc. 39th Annual International Symposium on Computer Architecture*, Portland, OR, pp.416–427 (2012).
- [2] Altera Corporation: Stratix IV Device Handbook (2012) (online), available from (<http://www.altera.com/literature/lit-stratix-iv.jsp>).
- [3] Apple Inc.: About the iFrame video format (2012) (online), available from (<http://support.apple.com/kb/HT3905>).
- [4] Al, A., Rao, B.P., Kudva, S.S., Babu, S., Suman, D. and Rao, A.V.: Quality and complexity comparison of H.264 intra mode with JPEG2000 and JPEG, *Proc. IEEE International Conference on Image Processing*, Singapore, pp.525–528 (2004).
- [5] ARM Ltd.: Embedded Memory IP (2012) (online), available from (<http://www.arm.com/products/physical-ip/embedded-memory-ip/index.php>).
- [6] Bao, X., Zhou, D. and Goto, S.: A lossless frame recompression scheme for reducing DRAM power in video encoding, *Proc. IEEE International Symposium on Circuits and Systems*, Paris, France, pp.677–680 (2010).
- [7] Chang, C.H., Chen, J.W., Chang, H.C., Yang, Y.C., Wang, J.S. and Guo, J.I.: A quality scalable H.264/AVC baseline intra encoder for high definition video application, *Proc. IEEE Workshop on Signal Processing Systems*, Shanghai, China, pp.521–526 (2007).
- [8] Chen, T.C., Chien, S.Y., Huang, Y.W., Tsai, C.H., Chen, C.Y., Chen, T.W. and Chen, L.G.: Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.16, No.6, pp.673–688 (June 2006).
- [9] Chang, H.C., Chen, J.W., Su, C.L., Yang, Y.C., Li, Y., Chang, C.H., Chen, Z.M., Yang, W.S., Lin, C.C., Chen, C.W., Wang, J.S. and J. I.Guo: A 7 mW-to-183 mW dynamic quality-scalable h.264 video encoder chip, *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, San Francisco, CA, pp.280–281 (2007).
- [10] Chen, T.C., Chen, Y.H., Tsai, C.Y., Tsai, S.F., Chien, S.Y. and Chen, L.G.: 2.8 to 67.2 mW Low-Power and Power-Aware H.264 Encoder for Mobile Applications, *2007 Symposium on VLSI Circuits Digest of Technical Papers*, Kyoto, Japan, pp.222–223 (2007).
- [11] Chen, Y.H., Chen, T.C., Tsai, C.Y., Tsai, S.F. and Chen, L.G.: Algorithm and Architecture Design of Power-Oriented H.264/AVC Baseline Profile Encoder for Portable Devices, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.19, No.8, pp.1118–1128 (Aug. 2009).
- [12] Chen, J.W., Chang, H.C., Wang, J.S. and Guo, J.I.: A dynamic quality-adjustable H.264 intra coder, *IEEE Trans. Consumer Electronics*, Vol.57, No.3, pp.1203–1211 (Aug. 2011).
- [13] Chang, H.C., Chen, J.W., Wu, B.T., Su, C.L., Wang, J.S. and Guo, J.I.: A Dynamic Quality-Adjustable H.264 Video Encoder for Power-Aware Video Applications, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.19, No.12, pp.1739–1754 (Dec. 2009).
- [14] Chen, J., Chen, Y., Zhu, H., Sui, C., Wu, P. and Cao, X.: The hardware design and implementation for CAVLC and Exp-Golomb in H.264/AVC, *Proc. 12th International Conference on Advanced Communication Technology*, Yeongchang, Korea, pp.1610–1613 (2010).
- [15] Calderbank, A.R., Daubechies, I., Sweldens, W. and Yeo, B.L.: Wavelet transforms that map integers to integers, *Applied and computational harmonic analysis*, Vol.5, No.3, pp.332–369 (July 1998).
- [16] Chen, W.Y., Ding, L.F., Tsung, P.K. and Chen, L.G.: Architecture design of high performance embedded compression for high definition video coding, *Proc. IEEE International Conference on Multimedia and Expo*, Hannover, Germany, pp.825–828 (2008).
- [17] Chen, C.Y., Huang, C.T., Chen, Y.H. and Chen, L.G.: Level C+ data reuse scheme for motion estimation with corresponding coding orders, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.16, No.4, pp.553–558 (April 2006).
- [18] Chiu, L.C. and Chang, T.S.: A lossless embedded compression codec engine for HD video decoding, *Proc. International Symposium on VLSI Design, Automation, and Test*, Hsinchu, Taiwan, pp.1–4 (2012).
- [19] Chen, Y.H., Chang, T.Y. and Lu, C.W.: A low-cost and high-throughput architecture for H.264/AVC integer transform by using four computation streams, *Proc. 13th International Symposium on Integrated Circuits*, Singapore, Singapore, pp.380–383 (2011).
- [20] Cheng, C.C., Ku, C.W. and Chang, T.S.: A 1280 × 720 pixels 30 frames/s H.264/MPEG-4 AVC intra encoder, *Proc. IEEE International Symposium on Circuits and Systems*, Island of Kos, Greece, pp.5335–5338 (2006).
- [21] Chao, P. and Lin, Y.L.: A motion compensation system with a high efficiency reference frame pre-fetch scheme for QFHD H.264/AVC decoding, *Proc. IEEE International Symposium on Circuits and Systems*, Seattle, WA, pp.256–259 (2008).
- [22] Chen, J.W., Wu, L.C., Liu, P.S. and Lin, Y.L.: A high-throughput fully hardwired CABAC encoder for QFHD H.264/AVC main profile video, *IEEE Trans. Consumer Electronics*, Vol.56, No.4, pp.2529–2536 (Nov. 2010).
- [23] Dikbas, S. and Zhai, F.: Lossless image compression using adjustable fractional line-buffer, *Signal Process: Image Communication*, Vol.25, No.5, pp.345–351 (June 2010).
- [24] Diniz, C., Zatt, B., Thiele, C., Susin, A., Bampi, S., Sampaio, F., Palomino, D. and Agostini, L.: A high throughput H.264/AVC intra-frame encoding loop architecture for HD1080p, *Proc. IEEE International Symposium on Circuits and Systems*, Rio de Janeiro, Brazil, pp.579–582 (2011).
- [25] Gupte, A.D., Amrutur, B., Mehendale, M.M., Rao, A.V. and Budagavi, M.: Memory bandwidth and power reduction using lossy reference frame compression in video encoding, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.21, No.2, pp.225–230 (Feb. 2011).
- [26] Golomb, S.W.: Run-length encodings, *IEEE Trans. Inf. Theory*, Vol.IT-12, pp.399–401 (July 1966).
- [27] Han, C.S. and Lee, J.H.: Area efficient and high throughput CAVLC encoder for 1920×1080@30p H.264/AVC, *International Conference on Consumer Electronics Digest of Technical Papers*, Las Vegas, NV, pp.1–2 (2009).
- [28] Huang, Y.W., Chen, T.C., Tsai, C.H., Chen, C.Y., Chen, T.W., Chen, C.S., Shen, C.F., Ma, S.Y., Wang, T.C., Hsieh, B.Y., Fang, H.C. and Chen, L.G.: A 1.3TOPS H.264/AVC single-chip encoder for HDTV applications, *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, San Francisco, CA, pp.128–129 (2005).
- [29] Heithecker, S. and Ernst, R.: Traffic shaping for an FPGA based SDRAM controller with complex QoS requirements, *Proc. 20th European Signal Processing Conference*, Bucharest, Romania, pp.1054–1058 (2012).
- [30] Huang, Y.W., Hsieh, B.Y., Chen, T.C. and Chen, L.G.: Analysis, fast algorithm, and VLSI architecture design for H.264/AVC intra frame coder, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.15, No.3, pp.378–401 (Mar. 2005).
- [31] Horowitz, M., Joch, A., Kossentini, F. and Hallapuro, A.: H.264/AVC baseline profile decoder complexity analysis, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.13, No.7, pp.704–716 (July 2003).
- [32] Huffman, D.A.: A method for the construction of minimum-redundancy codes, *Proc. IRE*, Vol.40, No.9, pp.1098–1101 (Sep. 1952).
- [33] Hu, H., Sun, J. and Xu, J.: High efficiency synchronous DRAM controller for H.264 HDTV encoder, *Proc. 4th IEEE Conference on Industrial Electronics Applications*, Xian, China, pp.2132–2136 (2009).
- [34] Hu, H., Xu, J., Duan, Z. and Sun, J.: High efficiency synchronous

- DRAM controller for H. 264 HDTV encoder, *Proc. IEEE Workshop on Signal Processing Systems*, Shanghai, China, pp.373–376 (2007).
- [35] He, G., Zhou, D., Zhou, J. and Goto, S.: A 1991 Mpixels/s intra prediction architecture for super hi-vision H.264/AVC encoder, *Proc. 42nd Annual Design Automation Conference*, Anaheim, CA, pp.575–578 (2005).
- [36] IEEE standard for verilog hardware description language, IEEE Std 1364-2005 (2005).
- [37] Information technology — Digital compression and coding of continuous-tone still image — requirements and guidelines, ISO/IEC IS 10918-1 | ITU-T Recommendation T.81 (1992).
- [38] Information technology — Generic coding of moving pictures and associated audio information: Video, ISO/IEC 13818-2 (2000).
- [39] Information technology — JPEG 2000 image coding system: Motion JPEG 2000, ISO/IEC 15444-3 | ITU-T Recommendation T.802 (2002).
- [40] Information technology — Coding of audio-visual objects — Part 2: Visual, ISO/IEC 14496-2:2004 (2004).
- [41] Information technology — Coding of audio-visual objects — Part 10: Advanced Video Coding, ISO/IEC 14496-10:2012 (2012).
- [42] H.263: Video coding for low bit rate communication, ITU-T Recommendation H.263 (2005).
- [43] H.264: Advanced video coding for generic audiovisual services, ITU-T Recommendation H.264 (2012).
- [44] Ivanov, Y.V. and Moloney, D.: Reference frame compression using embedded reconstruction patterns for H.264/AVC decoders, *Proc. 3rd International Conference on Digital Telecommunications*, Bucharest, Romania, pp.168–173 (2008).
- [45] Jeong, H., Kim, J., Lee, K., Yoo, K. and Kim, J.: Lossless embedded compression algorithm with context-based error compensation for video application, *Proc. IEEE 16th International Symposium on Consumer Electronics*, Harrisburg, PA, pp.1–4 (2012).
- [46] Joch, A., Kossentini, F., Schwarz, H., Wiegand, T. and Sullivan, G.J.: Performance comparison of video coding standards using Lagrangian coder control, *Proc. IEEE International Conference on Image Processing*, Rochester, NY, pp.501–504 (2002).
- [47] Joint Video Team, H.264/AVC Joint Model reference software 8.6 (online), available from [http://iphome.hhi.de/suehring/tml/download/old\\_jm/](http://iphome.hhi.de/suehring/tml/download/old_jm/).
- [48] Joint Video Team, H.264/AVC Joint Model reference software 9.3 (online), available from [http://iphome.hhi.de/suehring/tml/download/old\\_jm/](http://iphome.hhi.de/suehring/tml/download/old_jm/).
- [49] Joint Video Team, H.264/AVC Joint Model reference software 11.0 (online), available from [http://iphome.hhi.de/suehring/tml/download/old\\_jm/](http://iphome.hhi.de/suehring/tml/download/old_jm/).
- [50] Ku, C.W., Cheng, C.C., Yu, G.S., Tsai, M.C. and Chang, T.S.: A high-definition H.264/AVC intra-frame codec IP for digital video and still camera applications, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.16, No.8, pp.917–928 (Aug. 2006).
- [51] Kim, Y., Han, D., Mutlu, O. and Harchol-Balter, M.: ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers, *Proc. IEEE 16th International Symposium on High Performance Computer Architecture*, Bangalore, India, pp.1–12 (2010).
- [52] Kim, J., Kim, J. and Kyung, C.M.: A lossless embedded compression algorithm for high definition video coding, *Proc. IEEE International Conference on Multimedia and Expo*, New York, NY, pp.193–196 (2009).
- [53] Kim, J. and Kyung, C.M.: A lossless embedded compression using significant bit truncation for HD video coding, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.20, No.7, pp.848–860 (June 2010).
- [54] Kim, H. and Park, I.C.: High-performance and low-power memory interface architecture for video processing applications, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.11, No.11, pp.1160–1170 (Nov. 2001).
- [55] Kao, Y.C., Kuo, H.C., Lin, Y.T., Hou, C.W., Li, Y.H., Huang, H.T. and Lin, Y.L.: A high-performance VLSI architecture for intra prediction and mode decision in H. 264/AVC video encoding, *Proc. IEEE Asia Pacific Conference on Circuits and Systems*, Singapore, pp.562–565 (2006).
- [56] Kim, H.S., Lee, J., Kim, H., Kang, S. and Park, W.C.: A lossless color image compression architecture using a parallel Golomb-Rice hardware codec, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.21, No.11, pp.1581–1587 (Nov. 2011).
- [57] Kulkarni, A.M. and Arunachalam, V.: FPGA implementation & comparison of current trends in memory scheduler for multimedia application, *Proc. International Conference and Workshop on Emerging Trends in Technology*, Mumbai, India, pp.1214–1218 (2011).
- [58] Kuo, H.C., Chen, J.W. and Lin, Y.L.: A high-performance low-power H.264/AVC video decoder accelerator for embedded systems, *Proc. IEEE/ACM/IFIP 7th Workshop on Embedded Systems for Real-time Multimedia*, Grenoble, France, pp.1–8 (2009).
- [59] Lee, S.C.: System-level optimization for efficient IP communication and memory access on a bus-enabled H.264/AVC decoder SoC, Master thesis, Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan (2008).
- [60] Lee, S.H., Chung, M.K., Park, S.M. and Kyung, C.M.: Lossless frame memory recompression for video codec preserving random accessibility of coding unit, *IEEE Trans. Consumer Electronics*, Vol.55, No.4, pp.2105–2113 (Nov. 2009).
- [61] Lee, K.B. and Chang, T.S.: SoC memory system design, *Essential Issues in SOC Design*, Lin, Y.L. (Ed.), 1st ed. ch. 4, pp.73–118, Springer (2006).
- [62] Lee, K.B., Lin, T.C. and Jen, C.W.: An efficient quality-aware memory controller for multimedia platform SOC, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.15, No.5, pp.620–633 (May 2005).
- [63] Lee, Y., Rhee, C.E. and Lee, H.J.: A new frame recompression algorithm integrated with H.264 video compression, *Proc. IEEE International Symposium on Circuits and Systems*, New Orleans, LA, pp.1621–1624 (2007).
- [64] Liew, T.B.Y., Lee, B.G. and Yoo, H.: A low complexity and lossless frame memory compression for display devices, *IEEE Trans. Consumer Electronics*, Vol.54, No.3, pp.1453–1458 (Aug. 2008).
- [65] Liu, Z. and Wang, D.: One-round renormalization based 2-bin/cycle H.264/AVC CABAC encoder, *Proc. 18th IEEE International Conference on Image Processing*, Brussels, Belgium, pp.369–372 (2011).
- [66] Li, Y., Jiang, Y. and Meng, H.: Adaptive pixel encoding: an effective algorithm for frame buffer compression, *Proc. 12th IEEE International Conference on Computer and Information Technology*, Chengdu, China, pp.5–11 (2012).
- [67] List, P., Joch, A., Lainema, J., Bjntegaard, G. and Karczewicz, M.: Adaptive deblocking filter, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.13, No.7, pp.614–619 (July 2003).
- [68] Li, D.W., Ku, C.W., Cheng, C.C., Lin, Y.K. and Chang, T.S.: A 61 MHz 72 K gates  $1280 \times 720$  30 fps H.264 intra encoder, *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Honolulu, HI, pp.(II) 801–804 (2007).
- [69] Lin, Y.K., Ku, C.W., Li, D.W. and Chang, T.S.: A 140-MHz 94 K gates HD1080p 30-frames/s intra-only profile H.264 encoder, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.19, pp.432–436 (Mar. 2009).
- [70] Lin, Y.K., Li, D.W., Lin, C.C., Kuo, T.Y., Wu, S.J., Tai, W.C., Chang, W.C. and Chang, T.S.: A 242 mW  $10 \text{ mm}^2$  1080 p H.264/AVC high-profile encoder chip, *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, San Francisco, CA, pp.314–315 (2008).
- [71] Liu, Z.Y., Song, Y., Shao, M., Li, S., Li, L.F., Ishiwata, S., Nakagawa, M., Goto, S. and Ikenaga, T.: A 1.41 W H.264/AVC real-time encoder SoC for HDTV1080p, *2007 Symposium on VLSI Circuits Digest of Technical Papers*, Kyoto, Japan, pp.12–13 (2007).
- [72] Liu, Z.Y., Song, Y., Shao, M., Li, S., Li, L.F., Ishiwata, S., Nakagawa, M., Goto, S. and Ikenaga, T.: HDTV1080p H.264/AVC Encoder Chip Design and Performance Analysis, *IEEE Journal of Solid-State Circuits*, Vol.44, No.2, pp.594–608 (Feb. 2009).
- [73] Lin, H.Y., Wu, K.H., Liu, B.D. and Yang, J.F.: An efficient VLSI architecture for transform-based intra prediction in H.264/AVC, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.20, No.6, pp.894–906 (June 2010).
- [74] Liu, Z., Zhou, J., Wang, D. and Ikenaga, T.: Register Length Analysis and VLSI Optimization of VBS Hadamard Transform in H.264/AVC, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.21, No.5, pp.601–610 (May 2011).
- [75] Lei, J., Zou, X., Wu, Z. and Fan, W.: Research of an image map encoding algorithm on frame buffer, *Proc. International Conference on ASIC*, Guilin, China, pp.894–897 (2007).
- [76] Ma, Z. and Segall, A.: Frame buffer compression for low-power video coding, *Proc. 18th IEEE International Conference on Image Processing*, Brussels, Belgium, pp.757–760 (2011).
- [77] Marpe, D., Schwarz, H. and Wiegand, T.: Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.13, No.7, pp.620–636 (July 2003).
- [78] Merhav, N., Seroussi, G. and Weinberger, M.J.: Coding for sources with two-sided geometric distributions and unknown parameters, *IEEE Trans. Inf. Theory*, Vol.46, No.1, pp.229–236 (Jan. 2000).
- [79] Merhav, N., Seroussi, G. and Weinberger, M.: Optimal prefix codes for sources with two-sided geometric distributions, *IEEE Trans. Inf. Theory*, Vol.46, No.1, pp.121–135 (Jan. 2000).
- [80] Marpe, D., George, V., Cycon, H.L. and Barthel, K.U.: Performance



- evaluation of Motion-JPEG2000 in comparison with H.264/AVC operated in intra coding mode, *Proc. SPIE Conference on Wavelet Applications in Industrial Processing*, Providence, RI, pp.129–137 (2003).
- [81] Micron Technology, Inc.: SDR SDRAM mt48lc16m16a2b4 (1999) (online), available from (<http://www.micron.com/parts/dram/sdram/mt48lc16m16a2b4-6a?pc={428A5CC9-2A78-447E-939B-6F3A40D538C6}>).
- [82] Morein, S.: ATI radeon hyperZ technology, *SIGGRAPH/EUROGRAPHICS Workshop On Graphics Hardware* (Aug. 2000).
- [83] Motorola Solutions, Inc.: Video surveillance trade-offs (2012) (online), available from ([http://www.motorola.com/web/Business/~/Documents/static%20files/VideoSurveillance\\_WP\\_3\\_keywords.pdf](http://www.motorola.com/web/Business/~/Documents/static%20files/VideoSurveillance_WP_3_keywords.pdf)).
- [84] Muralidha, P., Vasundhara, R., Rama Rao, C.B. and Murthy, N.S.: An efficient architecture for H.264 intra prediction mode decision algorithm, *Proc. 10th WSEAS international conference on electronics, hardware, wireless and optical communications, 10th WSEAS international conference on signal processing, robotics and automation, 3rd WSEAS international conference on nanotechnology, 2nd WSEAS international conference on Plasma-fusion-nuclear physics*, Cambridge, UK, pp.113–116 (2011).
- [85] Nadeem, M., Wong, S. and Kuzmanov, G.: An efficient realization of forward integer transform in H.264/AVC intra-frame encoder, *Proc. International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation*, Samos, Greece, pp.71–78 (2010).
- [86] Ostermann, J., Bormans, J., List, P., Marpe, D., Narroschke, M., Pereira, F., Stockhammer, T. and Wedi, T.: Video coding with H.264/AVC: tools, performance, and complexity, *IEEE Circuits and Systems Magazine*, Vol.4, No.1, pp.7–28 (Aug. 2004).
- [87] Panasonic Broadcast: H.264/AVC intra-only compression for P2 application (2007) (online), available from ([ftp://ftp.panasonic.com/pub/panasonic/drivers/PBTS/papers/WP/\\_AVC-Intra.pdf](ftp://ftp.panasonic.com/pub/panasonic/drivers/PBTS/papers/WP/_AVC-Intra.pdf)).
- [88] Ren, H., Fan, Y., Chen, X. and Zeng, X.: A 16-pixel parallel architecture with block-level/mode-level co-reordering approach for intra prediction in  $4k \times 2k$  H.264/AVC video encoder, *Proc. 17th Asia and South Pacific Design Automation Conference*, Sydney, Australia, pp.801–806 (2012).
- [89] Rice, R.F.: Some practical universal noiseless coding techniques, Jet Propulsion Laboratory, Pasadena, CA, Technical Report JPL-79-22 (Mar. 1979).
- [90] Roszkowski, M. and Pastuszak, G.: Intra prediction hardware module for high-profile H.264/AVC encoder, *Proc. Signal Processing Algorithms, Architectures, Arrangements, and Applications Conference*, Poznan, Poland, pp.62–67 (2010).
- [91] Sayood, K.: *Introduction to Data Compression*, Morgan Kaufmann Publishers (2006).
- [92] Shao, J. and Davis, B.T.: A burst scheduling access reordering mechanism, *Proc. IEEE 13th International Symposium on High Performance Computer Architecture*, Scottsdale, AZ, pp.285–294 (2007).
- [93] Shafique, M., Bauer, L. and Henkel, J.: A parallel approach for high performance hardware design of intra prediction in H.264/AVC video codec, *Proc. Design, Automation and Test in Europe Conference and Exhibition*, Nice, France, pp.1434–1439 (2009).
- [94] Son, C.H., Kim, J.W., Song, S.G., Park, S.M. and Kim, Y.M.: Low complexity embedded compression algorithm for reduction of memory size and bandwidth requirements in the JPEG2000 encoder, *IEEE Trans. Consumer Electronics*, Vol.56, No.4, pp.2421–2429 (Nov. 2010).
- [95] Song, T. and Shimamoto, T.: Reference frame data compression method for H.264/AVC, *IEICE Electronics Express*, Vol.4, No.3, pp.121–126 (Feb. 2007).
- [96] Silveria, D., Sanchez, G., Grellert, M., Possani, V. and Agostini, L.: Memory bandwidth reduction in video coding systems through context adaptive lossless reference frame compression, *Proc. 2012 VIII Southern Conference on Programmable Logic*, Bento Goncalves, Brazil, pp.1–6 (2012).
- [97] Song, L., Zhou, D., Jin, X., Goto, S. and Liu, P.: An adaptive bandwidth reduction scheme for video coding, *Proc. IEEE International Symposium on Circuits and Systems*, Paris, France, pp.401–404 (2010).
- [98] Suh, K., Park, S. and Cho, H.: An efficient hardware architecture of intra prediction and TQ/IQIT module for H.264 encoder, *ETRI Journal*, Vol.27, No.5, pp.511–524 (Oct. 2005).
- [99] Tsai, C.Y., Chen, T.C., Chen, T.W. and Chen, L.G.: Bandwidth optimized motion compensation hardware design for H.264/AVC HDTV decoder, *Proc. 48th Midwest Symposium on Circuits and Systems*, Cincinnati, OH, pp.1119–1202 (2005).
- [100] Teuhola, J.: A Compression Method for Clustered Bit-Vectors, *Information Processing Letters*, Vol.7, pp.308–311 (Oct. 1978).
- [101] Toshiba America Information Systems, Inc.: Designing an IP camera project (2011) (online), available from ([http://www.toshibasecurity.com/support/docs/Toshiba\\_Design\\_an\\_IPSystem\\_WhitePaper.pdf](http://www.toshibasecurity.com/support/docs/Toshiba_Design_an_IPSystem_WhitePaper.pdf)).
- [102] Taiwan Semiconductor Manufacturing Company Limited: 0.13  $\mu\text{m}$  Technology (2000) (online), available from (<http://www.tsmc.com/english/dedicatedFoundry/technology/0.13um.htm>).
- [103] Tuan, J.C., Chang, T.S. and Jen, C.W.: On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.12, No.1, pp.61–72 (Jan. 2002).
- [104] United Microelectronics Corporation: 90 Nanometer (2003) (online), available from ([http://www.umc.com/English/pdf/90nm\\_DM.pdf](http://www.umc.com/English/pdf/90nm_DM.pdf)).
- [105] Vo, D.T., Lertrattanapanich, S. and Kim, Y.T.: Low line memory visually lossless compression for color images using non-uniform quantizers, *IEEE Trans. Consumer Electronics*, Vol.57, No.1, pp.187–195 (Feb. 2011).
- [106] Wang, T.S. and Chiu, C.T.: Low power design of high performance memory access architecture for HDTV decoder, *Proc. IEEE International Conference on Multimedia and Expo*, Beijing, China, pp.699–702 (2007).
- [107] Wedi, T. and Musmann, H.G.: Motion- and aliasing-compensated prediction for hybrid video coding, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.13, No.7, pp.577–586 (July 2003).
- [108] Wiegand, T. and Girod, B.: *Multi-frame motion-compensated prediction for video transmission*, Springer (2001).
- [109] Wiegand, T. and Girod, B.: Lagrange multiplier selection in hybrid video coder control, *Proc. IEEE International Conference on Image Processing*, Thessaloniki, Greece, pp.542–545 (2001).
- [110] Wiegand, T. and Sullivan, G.J.: Video compression — from concepts to the H.264/AVC standard, *Proc. IEEE*, Vol.93, No.1, pp.18–31 (Jan. 2005).
- [111] Wiegand, T., Ohm, J.R., Sullivan, G.J., Han, W.J., Joshi, R., Tan, T.K. and Ugur, K.: Special section on the joint call for proposals on high efficiency video coding (HEVC) standardization, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.20, No.12, pp.1661–1666 (Dec. 2010).
- [112] Wiegand, T., Schwarz, H., Joch, A., Kossentini, F. and Sullivan, G.J.: Rate-constrained coder control and comparison of video coding standards, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.13, No.7, pp.688–703 (July 2003).
- [113] Yang, H.T., Chen, J.W., Kuo, H.C. and Lin, Y.L.: An effective dictionary-based display frame compressor, *Proc. IEEE/ACM/IFIP 7th Workshop on Embedded Systems for Real-time Multimedia*, Grenoble, France, pp.28–34 (2009).
- [114] Yoo, H., Jo, J.M. and Jeong, J.C.: A hierarchical lossless Image compression based on modified Hadamard Transform, *Proc. 10th Workshop on Image Processing and Understanding*, pp.516–520 (1998).
- [115] Yu, G.S. and Chang, T.S.: Optimal Data Mapping for Motion Compensation in H.264 Video Decoding, *Proc. IEEE Workshop on Signal Processing Systems*, Shanghai, China, pp.505–508 (2007).
- [116] Yuan, G.L., Bakhoda, A. and Aamodt, T.M.: Complexity effective memory access scheduling for many-core accelerator architectures, *Proc. 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, New York, NY, pp.34–44 (2009).
- [117] Zhou, D., He, G., Fei, W., Chen, Z., Zhou, J. and Goto, S.: A 4320p 60 fps H.264/AVC intra-frame encoder chip with 1.41 Gbins/s CABAC, *2012 Symposium on VLSI Circuits Digest of Technical Papers*, Honolulu, HI, pp.154–155 (2012).
- [118] Zhu, J., Hou, L. and Wu, W.: High performance synchronous DRAMs controller in H.264 HDTV decoder, *Proc. 7th International Conference on Solid-State Integrated Circuits Technology*, Beijing, China, pp.1621–1624 (2004).
- [119] Zhu, J., Liu, P. and Zhou, D.: An SDRAM controller optimized for high definition video coding application, *Proc. IEEE International Symposium on Circuits and Systems*, Seattle, WA, pp.3518–3521 (2008).

**Huang-Chih Kuo** received his B.S., M.S. and Ph.D., all in computer science, from National Tsing Hua University, Hsinchu, Taiwan, in 2004, 2006, and 2012 respectively. He is serving his one-year military obligation. His research interests include video coding algorithms and low-power VLSI architecture design.

**Youn-Long Lin** received his B.S. degree in electronics engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1982, and the Ph.D. in computer science from the University of Illinois at Urbana-Champaign, in 1987. He then joined National Tsing Hua University, Hsinchu, Taiwan, where he has served as Chairman of the Computer Science Department and Vice President of Research and Development. In 1998 he co-founded Global UniChip Corporation. Between 2001 and 2003, he worked for UniChip as its Chief Technical Officer and Executive Vice President. He is now a Chair Professor of computer science at National Tsing Hua University. He is also an Adjunct Professor with Peking University, Beijing, China, and a Guest Professor with Waseda University, Japan. His primary research interest is in computer-aided design (CAD) of VLSI circuits with emphasis on physical design automation and high-level synthesis. He co-authored the book “High Level Synthesis—Introduction to Chip and System Design” (Kluwer, 1992). His current research focus is on design technology for System-on-a-Chips (SOC) employing reusable silicon intellectual properties (IPs) and VLSI architecture for video coding.

(Invited by Editor-in-Chief: *Hiroyuki Tomiyama*)