Hardware Architecture Design of an H.264/AVC Video Codec

Tung-Chien Chen, Chung-Jr Lian, and Liang-Gee Chen
DSP/IC Design Lab, Graduate Institute of Electronics Engineering,
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
(e-mail: {djchen, cjlian, lgchen} @video.ee.ntu.edu.tw)

Abstract—H.264/AVC is the latest video coding standard. It significantly outperforms the previous video coding standards, but the extraordinary huge computation complexity and memory access requirement make the hardwired codec solution a tough job. This paper describes the design methodology for H.264/AVC video codec. The system architecture and scheduling will be addressed. The design consideration and optimization for its significant modules including bandwidth optimized motion compensation engine, reconfigurable intra predictor generator, low bandwidth parallel integer motion estimation will be mentioned. Due to the complex, sequential, and highly data-depended characteristics of all essential algorithms in H.264/AVC, not only the pipeline structure but also efficient memory hierarchy is required. The design case with a hybrid task pipelining scheme, a balanced schedule with block-level, MB-level, and frame-level pipelining, will be presented. By combining with many bandwidth reduction techniques and data reused schemes, very efficient architecture and implementation for plate-form based system is proved by the prototype chips.

I. Introduction

H.264/AVC is the new video coding standard. It can save 25%-45% and 50%-70% of bitrates when compared with MPEG-4 Advanced Simple Profile (ASP) [1] and MPEG-2 [2], respectively [3]. Although motion compensated transform coding is still adopted, many new features are used to achieve much better compression performance and subjective quality, such as quarter-pixel Motion Estimation (ME) with Multiple Reference Frames (MRF) and Variable Block Sizes (VBS), intra prediction, Context-based Adaptive Variable Length Coding (CAVLC), and in-loop deblocking filter [4][5][6]. The rate-distortion optimized mode decision [3] is also included in reference software [7] to improve rate-distortion efficiency.

There are many potential applications of H.264/AVC. Ongoing applications range from High Definition Digital Video Disc (HD-DVD) or BluRay for living room entertainment with large screens to Digital Video Broadcasting for Handheld terminals (DVB-H) with small screens. However, the H.264/AVC coding performance comes at the price of computation complexity. According to the instruction profiling with HDTV1024P (2048×1024, 30fps) specification, H.264/AVC decoding process requires 83 Giga-Instructions Per Second (GIPS) computation and 70 Giga-Bytes Per Second (GBPS) memory access. As for H.264/AVC encoder, up to 3600 GIPS and 5570 GBPS are required for HDTV720P (1280×720, 30fps) specification. For real-time applications, accelerating by the dedicated hardware is a must.

However, it is difficult to design an system architecture for the H.264/AVC codec. The design for the significant modules are also very challenging. Besides high computation complexity and memory access, the coding path is very long, which includes prediction, reconstruction, and entropy coding. The reference software adopts many sequential processing of each block in the MacroBlock (MB), which restricts the parallel processing. The block-level reconstruction loop caused by intra prediction induces the bubble cycles and decreases the hardware utilization and throughput. The coding tools involve with many data dependencies to enhance the coding performance, but the considerable storage space is the penalty. There are functionalities that have multiplex modes, and the re-configurable engine is essential to achieve resource sharing.

To address these difficulties, the hardware design methodology is described for H.264/AVC video coding system in this paper. There are three critical issues to be addressed. First, for H.264/AVC encoder, the traditional two-stage MB pipelines cannot be efficiently applied because of the long critical path and feedback loop. According to our analysis, five major functions are extracted and mapped into four stage MB pipelining structure with suitable task scheduling. Second, a hybrid task pipelining scheme is presented with a balanced schedule with block-level, MB-level, and frame-level pipelining to greatly reduce the internal memory size and bandwidth. Third, the design consideration and optimization for the significant modules, including bandwidth optimized Motion Compensation (MC) engine, reconfigurable intra predictor generator, low bandwidth parallel IME are involved. The design cases show that efficient implementation for H.264/AVC video coding system is achievable by combining these efficient architectures.

The rest of this paper is organized as follows. In Section II, the profiling and the design considerations are described. Then, the architecture optimization of H.264/AVC encoding system will be addressed in Section III. Those of H.264/AVC decoding parts are mentioned in Section IV. These architectures are proved by the prototype chips, which will be described in Section V. Finally, we will make a conclusion in Section VI.

II. Analysis and Design Space Exploration

A. Profiling

We use instruction profile to show the high computation complexity and memory access of H.264/AVC, and find the critical parts for hardware implementation. The iprof [8], a software analyzer at the instruction level, is used to profile an H.264/AVC encoder at a processor-based platform (SunBlade 2000 workstation, 1.015GHz Ultra Sparc II CPU, 8GB RAM, Solaris 8). To focus on the target specification, a software C model is developed by extracting all baseline profile compression tools from reference software [7]. Our focused design case is mainly targeted for SDTV (720×480, 30fps)/HDTV720P videos with four/one reference frame. The computational complexity and memory access for SDTV/HDTV720P are 2470/3600 GIPS and 3800/5570 GBPS. As for decoder with HDTV-1024P video formats. 83 GIPS and 70 GBPS of computation and memory access are required. The huge computational loads are far beyond the

0-7803-9451-8/06/$20.00

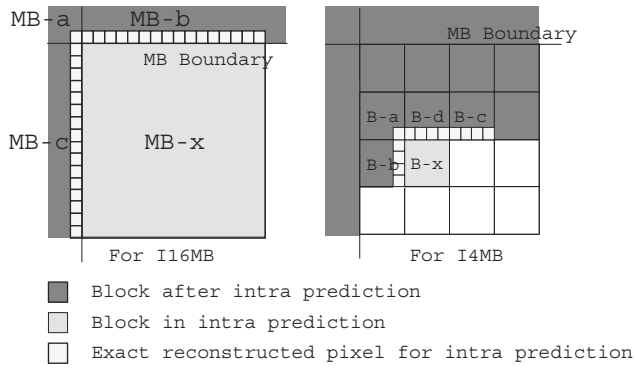

Fig. 1. (a) MB-level reconstruction loop; (b) block-level reconstruction loop. The intra prediction requires the reconstructed pixels of the left and top neighboring blocks, which induce the MB-level and block-level reconstruction loops.

capability of today's general purpose processors (GPPs). The dedicated hardware is essential for real-time applications.

B. Design Space Exploration

The considerations of hardware design are analyzed with the H.264/AVC compression algorithms. The major challenges are described as follows.

- **Computation Complexity and Bandwidth Requirement**: According to the profiling, H.264/AVC requires much more computation complexity than previous coding standards. this will greatly increase the hardware cost especially for the HDTV applications. The bandwidth requirement of H.264/AVC encoding system is also much higher than previous coding standard. The MRF-ME contributes the most traffic for loading reference pixels. Neighboring reconstructed pixels are required by intra prediction, and are also required by deblocking filter. Besides, Lagrangian mode decision and context-adaptive entropy coding have data dependencies between neighboring MB, and transmitting related information contribute considerable bandwidth as well. An efficient memory hierarchy combined with data sharing and Data Reuse (DR) scheme must be designed to reduce the system bandwidth.
- **Sequential Flow**: The H.264/AVC reference software adopts many sequential process to enhance the compression performance. It is hard to efficiently map the sequential algorithm to parallel hardware. For system architecture, we partition the sequential encoding process (prediction, reconstruction, and then entropy encoding) into several tasks and process them in MB-based pipelining structure, which improves the hardware utilization and the throughput. For module architecture, this problem is critical for ME since ME is the most computationally intensive part and requires the most degrees of parallelism. The inter Lagrangian mode decision takes MV costs into consideration. The MV of each block is generally medium predicted by left, top, and top-right neighboring blocks. The cost function can be computed only after prediction modes of neighboring blocks are determined, which also causes inevitable sequential processing. The modified hardware-oriented algorithms can be designed to enable parallel processing.
- **Coding Loops**: In traditional video coding standard, there is a frame-level reconstruction loop generating the reference frames for ME and MC. In H.264/AVC, the intra prediction requires the reconstructed pixels of the left and top neighboring blocks, which induce the MB-level and block-level reconstruction loops. For the MB-level reconstruction loop as shown in Fig. 1 (a), the reconstructed pixels of MB-a, MB-b, and MB-c are used to predict the pixels in MB-x for I16MB. Not until MB-a, MB-b, and MB-c are reconstructed can MB-x be predicted. Similarly, as Fig. 1 (b) shows, in order to support I4MB, not until 4×4-intra mode of B-a, B-b, B-c, and B-d are decided and reconstructed can B-x be processed. The reconstructed latency is harmful to hardware utilization and throughput if the intra prediction and reconstruction are not jointly considered and scheduled.
- **Data Dependency**: The new coding tools improve the compression performance with many data dependencies. The frame-level data dependencies contribute the considerable system bandwidth. The dependencies between neighboring MBs constrain the solution space of MB pipelining, and those between neighboring blocks limit the possibility of parallel processing.
- **Abundant Modes**: There are many algorithms of H.264/AVC that have multiplex modes. For example, there are 17 different modes for intra prediction while 259 kinds of partitions for inter prediction. Six kinds of 2-D transform, 4×4/2×2 DCT/IDCT/Hadamard transform, are involved in reconstruction loops. The reconfigurable processing engine, reusable prediction core, and appropriate pipeline system design are important to efficiently support all these functions.

C. Related Works

The conventional two-stage MB pipelining architecture [9][10] is widely adopted in prior video encoding hardware designs. Two MBs are processed simultaneously by prediction engine (ME only) and Block Engine (BE, including MC, reconstruction loop, and entropy coding) in pipeline manner. Several problems will be encountered if the two-stage MB pipelining is directly applied to H.264/AVC encoding. The prediction stage includes IME, FME, and intra prediction in H.264/AVC. The sequential prediction flow will make high operation frequency and low hardware utilization. Besides, because of MB-level and block-level reconstruction loops, it is impossible to completely separate the prediction and BE stages. The large bandwidth must also be reduced by efficient memory hierarchy and data reuse scheme.

Furthermore, because of the new funcationalities of H.264/AVC, the advanced module architectures are demanded for H.264/AVC encoder. IME is the most computationally intensive part in the encoder. Several IME architectures are proposed for VBSME [11] [12] [13][14] with different specifications. The main challenge for FME is to achieve parallel processing under the constraints of sequential Lagrangian mode decision. Besides, the functionalities of the VBS, MRF, 6-tap FIR, and Hadamard transform are involved. In [15], FME procedure is analyzed and decomposed into several loops. The fully pipelined architecture is designed with unfolding techniques and efficient scheduling. In [16], the forward/inverse multi-transform are designed to support 4×4 DCT, IDCT, and Hadamard transforms. The first deblocking filter propose the ad-

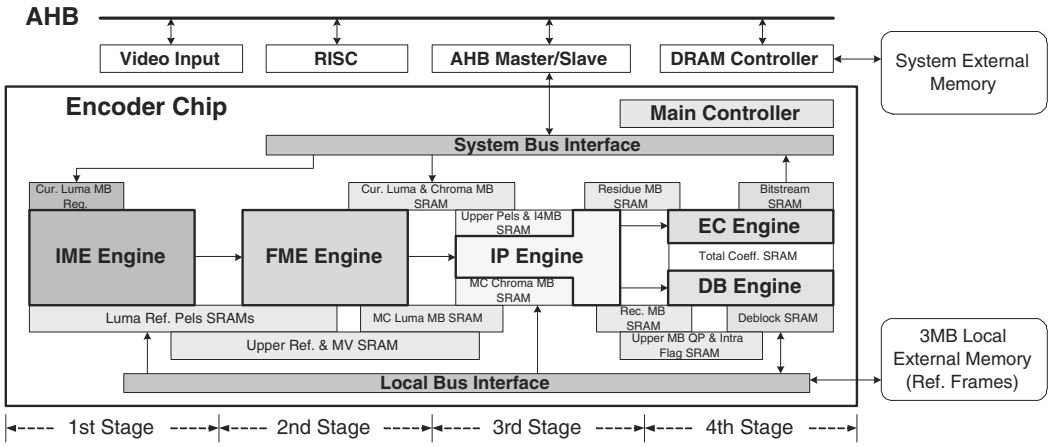

Fig. 2. Block diagram of the H.264/AVC encoding system. Five major tasks, including IME, FME, IP, EC, and DB, are partitioned from the sequential encoding procedure and processed MB by MB in pipeline structure.

vanced filtering schedule to save 50% on-chip bandwidth [17]. A column addressing technique used to favor the direction of vertical filtering is then proposed in [18] to increase the throughput. In [19], the architecture is further improved by interleavingly filtering the the horizontal and vertical edges. As for entropy encoder, the single buffer CAVLC architecture [20] is designed for SDTV specification. The dual-buffer architecture with block pipelining is proposed in [21] to double the hardware utilization and throughput.

III. H.264/AVC Encoding System

This section describes a new MB pipelining scheme for H.264/AVC encoder. The traditional two-stage MB pipelining [9][10], prediction (ME only) and BE, cannot be efficiently applied to H.264/AVC. In this encoding system, five major functions are extracted and mapped into four MB pipelining stages with suitable task scheduling [22]. Furthermore, the design consideration and optimization for the significant modules is described to enable the whole system. The efficient implementation for H.264/AVC encoding system is achieved by combining these techniques.

A. Four-Stage Macroblock Pipelining

The system architecture is shown in Fig. 2. Five major tasks, including IME, FME, Intra Prediction with reconstruction loop (IP), Entropy Coding (EC), and in-loop DeBlocking filter (DB), are partitioned from the sequential encoding procedure and processed MB by MB in pipeline structure.

Several issues are described as follows. The prediction, which is ME only in previous standards, includes IME, FME, and intra prediction in H.264/AVC. Because of the algorithms diversities and different computation complexity, it is difficult to implement IME, FME, and intra prediction by the same hardware. Putting IME, FME, and intra prediction in the same MB pipelining stage leads to very low utilization. Even if the resource sharing is achieved, the operating frequency becomes too high due to the sequentially processing. Therefore, FME is firstly pipelined MB by MB after IME to double the throughout. As for intra prediction, because of MB-level and block-level reconstruction loop, it cannot be separated with the reconstruction engine. Besides, the reconstruction process should be separated from ME and pipelined MB by MB to achieve highest hardware utilization. Therefore, engines of intra prediction together with forward/inverse transform/quantization should be located in the IP stage. In this way, the MB-level and block-level reconstruction loops can also be isolated in this pipelining stage. The EC encodes the MB header and residues after mode decision and prediction. The DB generates the standard-compliant reference frame after reconstruction. Since the EC/DB can be processed in parallel, they are placed at the 4th stage. The reference frame will be stored in external memory for the ME of the next frame, which constructs the frame-level reconstruction loop. Please note that, the luma MC is placed in FME stage for reuse of *Luma Ref. Pels SRAMs* and interpolation circuits. The compensated MB is transmitted to IP stage for generation of residues after intra/inter mode decision. On the other hand, chroma MC is implemented in IP stage since it is not required before intra/inter mode decision.

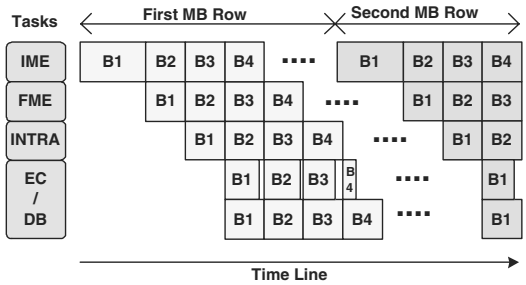

Fig. 3. The MB schedule of four stage MB pipelining. The horizontal arrow denotes the time line. One horizontal column indicates the MBs with different tasks that are processed in parallel.

MBs within one frame are coded in raster order with schedule in Fig. 3. The horizontal arrow denotes the time line. One horizontal column indicates the MBs with different tasks that are processed in parallel. As for reduction in system bandwidth, many on-chip memories are used for three purposes. First, in

order to find the best matched candidate, a huge amount of reference data are required for both IME and FME. Since pixels in neighboring candidate blocks are considerably overlapped, and so are the SWs of neighboring current MBs, the bandwidth of system bus can be greatly reduced by designing local buffers to store reusable data. Second, instead of transmitted by system bus, the raw data such as luma motion compensated MB, transformed and quantized residues, and reconstructed MB are shifted forward via shared memories. Third, because of data dependency, a MB is processed according to the upper and left MBs. The local memories are used to store the related data during the encoding process. For software implementation, the external bandwidth requirement is up to 5570 TBPS. As for hardware solution with local search window buffer embedded, the external bandwidth requirement is reduced to 700 MBPS. After all three techniques are applied, the final external bandwidth requirement is about 280 Mbytes/sec.

B. Low-Bandwidth Parallel Integer Motion Estimation

IME requires the most computational complexity and memory bandwidth in H.264/AVC. Besides, it is a kind of sequential flow due to the Lagrangian mode decision flow. However, a large degree of parallelism is required for the SDTV/HDTV specifications. In the following, techniques in algorithmic and architectural levels are used to enable parallel processing and to reduce the required hardware resources.

B.1 Hardware-Oriented Algorithm

The Motion Vector (MV) of each block is generally predicted by the medium values of MVs from the left, top, and top right neighboring blocks. The rate term of the Lagrangian cost function can be computed only after MVs of neighboring blocks are determined, which causes inevitable sequential processing. The blocks and subblocks in a MB cannot be processed in parallel. Moreover, when a MB is processed at the IME stage, its previous MB is still in the FME stage. The MB mode and the best MVs of the left MB cannot be obtained in the four-stage MB pipelining architecture. To solve these problems, the exact MVPs are replaced by modified MVPs, which is the medium of MVs from top-left, top, and top-right MBs. In addition, the modified MVP is applied for all of the 41 blocks in MB, as shown in Fig. 4. For example, the exact MV cost of the C22 4×4-block is the medium of the C12, C13, and C21 MVs. The MVPs of all 41 blocks are changed to the medium of MV0, MV1, and MV2 in order to facilitate the parallel processing and MB pipelining.

As for searching algorithm, FS is adopted to guarantee the highest compressing performance. The regular searching pattern is suitable for parallel processing. Besides, FS can effectively support VBS by reusing each 4×4-block SADs for larger blocks. Pixel truncation [23] of five-bit precision and subsampling [24] of half pixel rate are applied to reduce the hardware cost. Moreover, adaptive search range adjustment [25] is also applied to save computations.

B.2 Architectures Design of IME

In IME, in order to find the best matched candidate, a Search Window (SW) within one reference frame has to be searched.

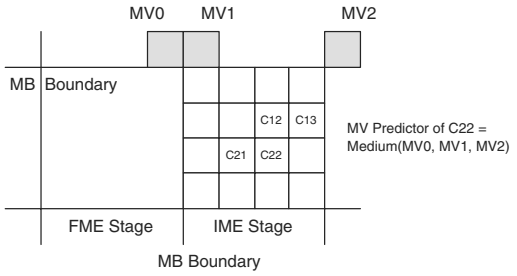

Fig. 4. Modified MVPs. In order to facilitate the parallel processing and MB pipelining, the MVPs of all 41 blocks are changed to the medium of MV0, MV1, and MV2.

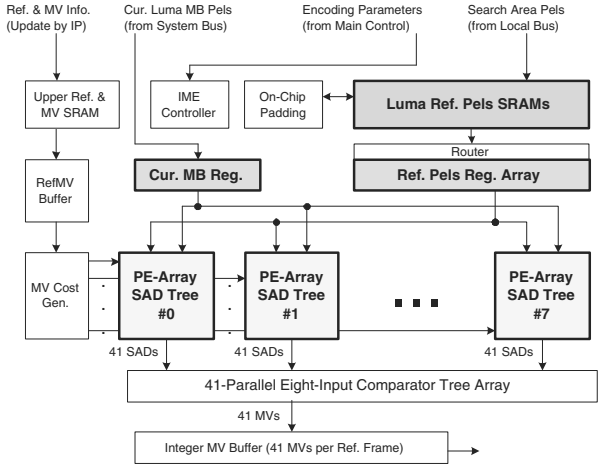

Fig. 5. Block diagram of the low bandwidth parallel IME engine. It mainly comprises eight *PE-Array 2-D SAD Tree*, and eight horizontally adjacent candidates are processed in parallel.

Figure 5 shows the low bandwidth parallel IME architecture, which mainly comprises eight *PE-Array 2-D SAD Tree*. The Current MB (CMB) is stored in *Cur. MB Reg*. The reference pixels are read from external memory and stored in *Luma Ref. Pels SRAMs*. Each PE array and its corresponding 2-D SAD tree compute the 41 SADs of VBS for one searching candidate each cycle. Therefore, eight horizontally adjacent candidates are processed in parallel.

Because SWs of neighboring current MBs are considerably overlapped, and so are the pixels of neighboring candidate blocks, a three-level memory hierarchy, including external memory, *Luma Ref. Pels SRAMs*, and *Ref. Pels Reg. Array*, is used to reduce bandwidth requirement by data reuse (DR). Three kinds of DR are implemented—MB-level DR, inter-candidate DR, and intra-candidate DR. The *Luma Ref. Pels SRAMs* are firstly embedded to achieve MB-level DR. when ME process is changed from one CMB to another CMB, there are overlapped area between neighboring SWs. Therefore, only a part of SW must be loaded from system memory, and the system bandwidth can be reduced [26].

The *Ref. Pels Reg. Array* acts as the cache between *PE-Array 2-D SAD Tree* and *Luma Ref. Pels SRAMs*. It is designed to achieve inter-candidate DR. A horizontal row of reference pixels, which is read from SRAMs, is stored and shifted

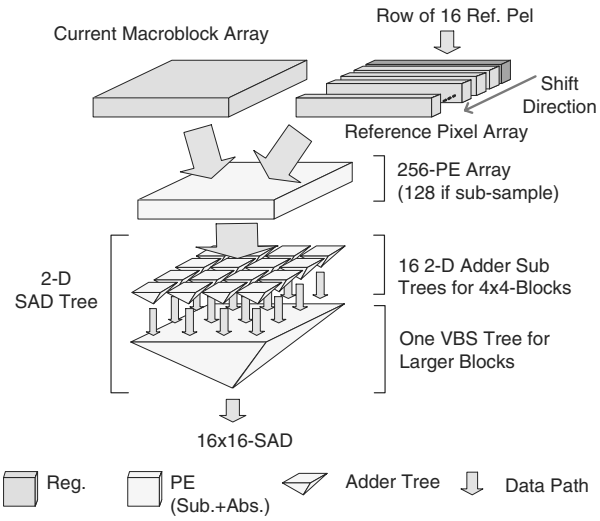

Fig. 6. *PE-array 2-D SAD Tree* architecture. The costs of sixteen 4×4-blocks are separately summed up by sixteen *2-D Sub-trees*, and then reused by one *VBS Tree* for larger blocks.

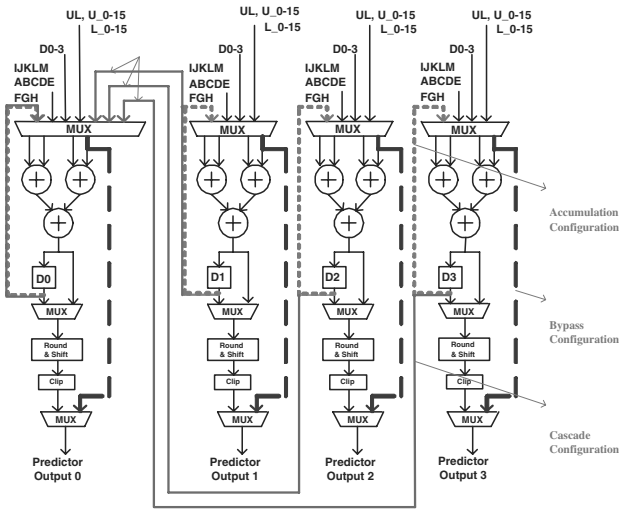

Fig. 7. Four-parallel reconfigurable intra predictor generator. Four different configurations are designed to support all intra prediction modes in H.264/AVC.

downward in *Ref. Pels Reg. Array*. When one candidate is processed, 256 reference pixels are required. When eight horizontally adjacent candidates are processed in parallel, not (256×8) but (256+16×7) reference pixels are required. Besides, when the ME process is changed to the next eight candidates, most data can be reused in *Ref. Pels Array*. The parallel architecture achieves inter-candidate DR in both horizontal and vertical directions and reduce the on-chip SRAM bandwidth.

Fig. 6 shows the architecture of *PE-array 2-D SAD Tree*. The costs of sixteen 4×4-blocks are separately summed up by sixteen *2-D Sub-trees*, and then reused by one *VBS Tree* for larger blocks. This is so-called intra-candidate DR. All 41 SADs for one candidate are simultaneously generated and compared with the 41 best costs. No intermediate data are buffered. Therefore, this architecture can support VBS without any partial SAD registers.

C. Reconfigurable Intra Predictor Generator

The intra prediction supports the most various prediction modes, which includes four I16MB modes, night I4MB modes, and four Chorma modes. For the RISC-based solution, the required operation frequency will become too high. For the dedicated hardware, 17 kinds of PEs for the 17 modes make the hardware cost high. Therefore, the reconfigurable circuit and resource sharing for all intra prediction modes are the efficient solutions.

The hardware architecture of the four-parallel reconfigurable intra predictor generator is shown in Fig. 7. Capital letters (A, B, C, ...) are the neighboring 4×4-block pixels. UL, L0-L15, and U0-U15 denote the bottom right pixel from the upper left MB, the 16 pixels of the right most column from the left MB, and the 16 pixels of the bottom row from the upper MB, respectively. Four different configurations are designed to support all intra prediction modes in H.264/AVC. Firstly, the I4MB/I16MB horizontal/vertical modes use the bypass data path to select the predictors extended from the block boundaries. Secondly, multiple PEs are cascaded to sum up the DC value for I4MB/I16MB/chroma DC mode. Thirdly, the normal configuration is used for I4MB directional modes 3–8. The four PEs select the corresponding pixels multiple times according to the weighted factors, and process independently. Finally, the recursive configuration is designed for I16MB plan prediction. The predictors are generated by adding the gradient values to the result of previous cycles.

IV. H.264/AVC Decoding System

In this section, a methodology to determine a suitable pipelining structure of a H.264/AVC decoder is presented. The target resolution is HDTV1024P 30fps videos. The design goals of this work are low area cost and low system bandwidth. In the following sections, the scheduling as well as the key modules of the decoding system will be elaborated.

A. Hybrid Task Pipelining

The overall system architecture is shown in Fig. 8 [27]. The sequence parameter set, picture parameter set, and slice headers are parsed by system processor. The MB-level information including MB headers and transformed/quantized residues are decoded by the *Parser Engine*. The predicted pixels are generated by the *Inter Pred. Engine* or *Intra Pred. Engine* according to the MB mode. The residues are recovered by the *IQ/IT Engine*. The MB is reconstructed by *Sum and Clipping Engine*. Finally, *Deblocking Engine* filters MB pixels and outputs them to the external memory. The buffers between the processing engines are required to separate pipelining stages.

Previous designs of video decoders are usually based on MB pipelining scheme [28]. Our system architecture is based on a hybrid task pipelining scheme including 4×4-block-level pipelining, MB-level pipelining, and frame-level pipelining. The reasons are stated as following. In H.264/AVC, 4×4-block is the smallest element of the prediction block mode. The transforms and entropy coding are also based on 4×4-blocks.

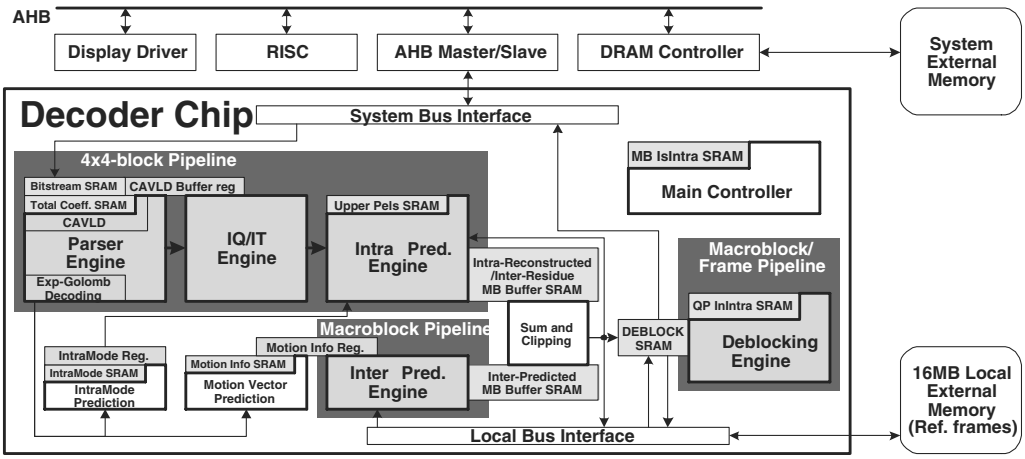

Fig. 8. Hybrid task pipelining system architecture.

Therefore, a 4×4-block pipelining scheme can be designed for CAVLD, inverse quantization/inverse transformation, and intra prediction with the benefit of less coding latency. It requires about $1/24$ of buffer size compared to the traditional MB pipelining architecture.

Inter prediction produces the predicted MB pixels from previously decoded reference frames. As with intra prediction, the basic processing element of inter prediction is also a 4×4-block. Due to the six-tap FIR filter for interpolation, 9×9 integer reference pixels are required for a current 4×4-block. If the blocks of prediction mode are larger than 4×4, overlapped reference frame pixels of these 4×4-blocks can be reused to reduce the system bandwidth. The inherent order of 4×4-blocks in the bitstream is the double-z-scan order. Reference frame DR will be less efficient if *Inter Pred. Engine* adopts the 4×4-block pipelining scheme and follows the double-z-scan order. Therefore, *Inter Pred. Engine* should be scheduled to MB-level pipelining with a customized scan order to exploit the reference frame DR. All reference pixels necessary to predict a MB are read from memory at once to reduce memory bandwidth.

Deblocking Engine is another special case that does not suit to the double-z scan order. *Deblocking Engine* filters the edges of each 4×4 block vertically then horizontally. Besides, one 4×4-block cannot be completely filtered until its neighboring blocks are reconstructed. This data dependency makes it impractical to fit the deblocking operation into a 4×4-block pipelining, since the buffer cannot be efficiently reduced and serious control overhead is required. If the decoder has to support FMO and ASO, where the MBs of a frame may not be coded in raster-scan order, the DB unit has to be scheduled to frame level pipelining because the filtering order of one frame can not either be violated in MB boundaries. Otherwise, the MB pipelining schedule is adopted.

B. Low-bandwidth Motion Compensation Engine

According to the analysis in decoder system, MC should be scheduled to MB-level pipelining with a customized scan order to exploit the reference frame DR. The 4×4-based MC is firstly adopted. All VBS are decomposed into several 4×4 element blocks, and processed by the MC engine with full hardware utilization. The straightforward memory access scheme processes every decomposed 4×4 element blocks independently, and always loads 9×9 pixels for interpolation as shown in Fig. 9(a). The bandwidth requirement of 4×4-based MC can be reduced by two bandwidth reduction techniques [29].

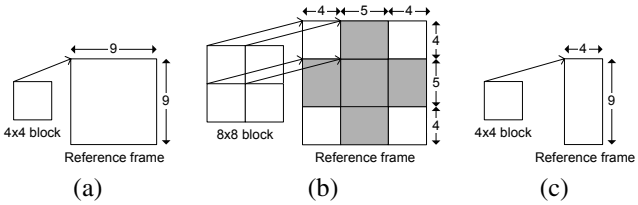

Fig. 9. (a) General case interpolation window; (b) Four interpolation windows for an 8×8 block (shaded region means reusable); (c) Interpolation window when MV pointing to horizontal integer pixels.

The first technique is *Interpolation Window Reuse* (IWR). As shown in Fig. 9(b), there are overlapped regions between interpolation windows for neighboring 4×4 element blocks when the block mode is larger than 4×4. The shaded regions can be reused. The second scheme is *Interpolation Window Classification* (IWC). The interpolation window is not always $(X+5)\times(Y+5)$ for a $X\times Y$ block. As shown in Fig. 9(c), a 4×4 block with integer MV in horizontal direction does not require horizontal filtering. A 4×9 interpolation window is read. In brief, the IWR and IWC scheme aim to precisely control the MC hardware to load a smaller and exact interpolation window. These techniques can provide about 60–80% bandwidth reduction for the 4×4-based MC.

Figure 10 shows the MC architecture. The efficient vertical scheduling in [15] is applied with *Down Shift Register Array* to support vertical IWR. Besides, an *Horizontal Reuse Memory* is designed for horizontal IWR. The IWC is implemented by *Control FSM* and *Address Generator*. The *Shift & Combine* circuit packs the required integer pixels inputted from external frame memory and *Horizontal Reuse Memory*. The *2-D IP Unit* performs the interpolation, and the compensated MB is buffered in the MC memory.

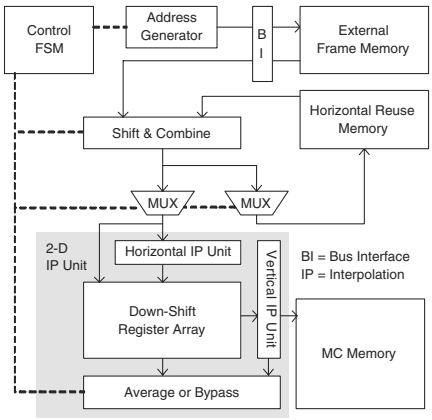

Fig. 10. Block diagram of MC hardware.

TABLE I
HARDWARE RESOURCE OF H.264/AVC ENCODER

Functional Block	Gate Counts	Memory (KB)
IME Module	305211	13.71
FME Module	401885	13.82
IP Module	121012	5.01
EC Module	29332	1.27
DB Module	20152	0.91
Others	45176	0.00
Total	922768	34.72

V. IMPLEMENTATION RESULTS

A. Implementation results of H.264/AVC SDTV/HDTV720P Encoder

The specification of this H.264/AVC encoder is baseline profile with level up to 3.1. The maximum computational capability is to real-time encode SDTV 30fps video with four reference frames or HDTV720P 30fps video with one reference frame. Table I shows the logic gate count profile synthesized at 120 MHz. The total logic gate count is about 922.8K. The prediction engine, including IME, FME, and IP stages, dominates 90% of logic area. As for on-chip SRAM requirement, 34.88 KB are required. The chip is fabricated with UMC 0.18 μm 1P6M CMOS process. Figure 11 shows the die micrograph. The core size is $7.68\times4.13\ mm^2$. The power consumption is 581 mW for D1 videos and 785 mW for HDTV720p videos at 1.8 V supply voltage with 81/108 MHz operating frequency. The detailed chip features are shown in Table II.

TABLE II
SPECIFICATION OF THE DEVELOPED H.264/AVC BASELINE PROFILE ENCODER CHIP

Technology	UMC 0.18 μm 1P6M CMOS
Pad/Core Voltage	3.3/1.8 V
Core Area	$7.68\times4.13\ mm^2$
Logic Gates	922.8 K (2-input NAND gate)
SRAM	34.72 KB
Operating Frequency	81/108 for D1/HDTV720P
Power Consumption	581/785 mW for D1/HDTV720P
Encoding Features	All Baseline Profile Compression Tools
Max. # of Ref. Frames	4/1 for D1/HDTV720P
Max. SR (Ref. 0)	H[-64,+63] V[-32,+31]
Max. SR (Ref. 1-3)	H[-32,+31] V[-16,+15]

TABLE III
GATE COUNT PROFILE OF THE H.264/AVC DECODER

Functional Block	Gate Counts	Memory (KB)
Main Control	22695	0.48
Parser Engine	21121	1.02
Inter Pred. Engine	69695	2.43
Intra Pred. Engine	28707	4.93
IQ/IT Engine	19792	0.00
Deblocking Engine	35437	1.12
Others	19980	0.00
Total	217428	9.98

TABLE IV
CHIP FEATURES OF H.264/AVC DECODER

Technology	TSMC 0.18 μm 1P6M CMOS
Pad/Core Voltage	3.3/1.8 V
Core Area	$2.19\times2.19\ mm^2$
Logic Gates	21.743 K (2-input NAND gate)
SRAM	9.98 KByte
Profile	Baseline
Operating Frequency	120/1.5 MHz for HDTV1024P 30fps/QCIF 15fps
Power Consumption	186.4/1.18 mW for HDTV1024P 30fps/QCIF 15fps

B. Implementation results of H.264/AVC HDTV1024P Decoder

The specification of this H.264/AVC decoder is baseline profile with level 4.1. It can real-time decode HDTV1024P 30fps video with operational frequency of 120 MHz. Table III shows the hardware resource requirement. The total logic gate count is 217K, and the on-chip SRAM requirement is 10 KB. The chip is fabricated with TSMC 0.18μm 1P6M CMOS technology. The core size is $2.19\times2.19\ mm^2$. The power consumption is 186.4 mW at 1.8V and 120MHz for decoding HDTV1024P 30fps videos, and is 1.18 mW at 1.8V and 1.5MHx for decoding QCIF (176×144) 15fps videos. The detailed chip features are shown in Table IV.

VI. CONCLUSION

In this paper, state-of-the-art hardware architectures for H.264/AVC video coding core have been presented. First, five major functional blocks are mapped into four-stage MB pipelining structure to highly increase the processing capability and hardware utilization. Besides, a hybrid task pipelining scheme, a balanced schedule with block-level, MB-level, and frame-level pipelining, is used for decoder to greatly reduce the internal memory size. Combined with many bandwidth reduction techniques and DR schemes, these two system architectures are all characterized by high throughput and low system bandwidth requirement. Moreover, efficient modules are designed to support the new H.264/AVC functionalities. A parallel IME architecture comprising eight PE arrays and adder trees is applied to dramatically reduce the memory bandwidth by three-level memory hierarchy and DR scheme. A reconfigurable intra predictor generator was designed to achieve resource sharing for all intra prediction modes. A bandwidth optimized MC engine highly exploits DR between interpolation windows of neighboring blocks. The hardwired coding system can efficiently support HDTV videos with real-time constraint, which is proved by the

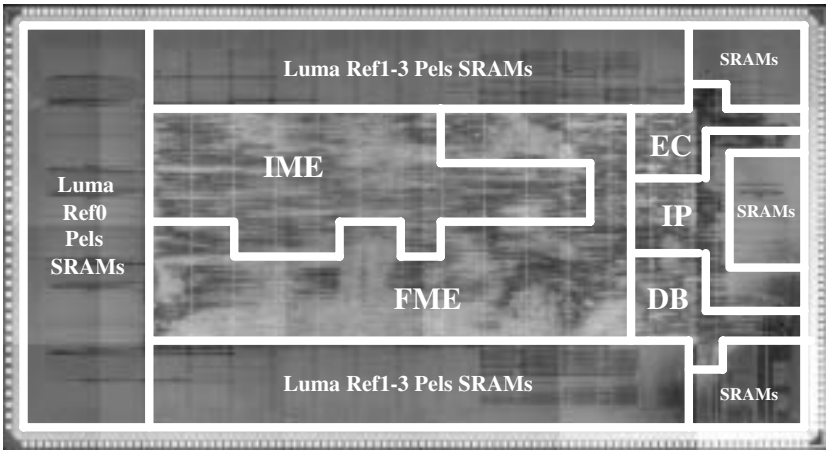

Fig. 11. Die micrograph of the H.264/AVC encoder [30].

prototype chips.

REFERENCES

[1] *Information Technology - Coding of Audio-Visual Objects - Part 2: Visual*, ISO/IEC 14496-2, 1999.

[2] *Information Technology - Generic Coding of Moving Pictures and Associated Audio Information: Video*, ISO/IEC 13818-2 and ITU-T Rec. H.262, 1996.

[3] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 688–703, July 2003.

[4] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[5] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Magazine on Circuits and Systems Magazine*, vol. 4, pp. 7–28, 2004.

[6] Atul Puri, Xuemin Chen, and Ajay Luthra, "Video coding using the H.264/MPEG-4 AVC compression standard," *Signal Processing: Image Communication*, vol. 19, pp. 793–849, Oct. 2004.

[7] *Joint Video Team Reference Software JM7.3*, http://bs.hhi.de/ suehring/tml/download/, Aug. 2003.

[8] ftp://ftp.lis.e-technik.tu muenchen.de/pub/iprof/, "Iprof ftp server," .

[9] M. Takahashi and et.al., "A 60-MHz 240-mW MPEG-4 videophone LSI with 16-Mb embedded DRAM," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1713–1721, Nov. 2000.

[10] H. Nakayama and et.al., "A MPEG-4 video LSI with an error-resilient codec core based on a fast motion estimation algorithm," in *Proceedings of IEEE International Solid-State Circuits Conference (ISSCC'02)*, Feb. 2005, vol. 2, pp. 296–512.

[11] Y.-W. Huang, T.-C. Wang, B.-Y. Hsieh, and L.-G. Chen, "Hardware architecture design for variable block size motion estimation in MPEG-4 AVC/JVT/ITU-T H.264," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'03)*, 2003, pp. 796–799.

[12] J.-H. Lee and N.-S. Lee, "Variable block size motion estimation algorithm and its hardware architecture for H.264," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'04)*, May 2004, vol. 3, pp. 740–743.

[13] Swee Yeow Yap and J. V. McCanny, "A VLSI architecture for variable block size video motion estimation," *IEEE Transactions on Circuit and System II*, vol. 51, pp. 384–389, 2004.

[14] Minho Kim, Ingu Hwang, and Soo-Ik Chae, "A fast vlsi architecture for full-search variable block size motion estimation in MPEG-4 AVC/H.264," in *Proc. of 2005 Asia and South Pacific Design Automation Conference*, Jan. 2005, vol. 1, pp. 631–634.

[15] T.-C Chen, Y.-W. Huang, and L.-G. Chen, "Fully utilized and reusable architecture for fractional motion estimation of H.264/AVC," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'04)*, 2004, pp. V9–V12.

[16] T.-C. Wang, Y.-W. Huang H.-C. Fang, and L.-G. Chen, "Parallel 4x4 2D transform and inverse transform architecture for MPEG-4 AVC/H.264," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'03)*, 2003, pp. 800–803.

[17] Y.-W. Huang, T.-C. Wang, B.-Y. Hsieh, T.-C. Wang, T.-H. Chang, and L.-G. Chen, "Architecture design for deblocking filter in H.264/JVT/AVC," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'03)*, 2003, pp. I693–I696.

[18] S.-Y Shih, C.-R. Chang, and Y.-L. Lin, "An AMBA-compliant deblocking filter IP for H.264/AVC," in *submission to IEEE International Symposium on Circuits and Systems (ISCAS'05)*, 2004.

[19] Tsu-Ming Liu, Wen-Ping Lee, and Chen-Yi Lee, "An area-efficient and high-throughput de-blocking filter for multi-standard video applications," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, 2005, pp. III–1044–1047.

[20] Y.-W. Huang, B.-Y. Hsieh, T.-C. Chen, , and L.-G. Chen, "Analysis, fast algorithm, and vlsi architecture design for H.264/AVC intra frame coder," *IEEE Transactions on Circuits and Systems for Video Technology*, 2004.

[21] T.-C. Chen, Y.-W. Huang, C.-Y. Tsai, , and L.-G. Chen, "Dual-block-pipelined VLSI architecture of entropy coding for H.264/AVC baseline profile," in *Proceedings of IEEE International Symposium on VLSI Design, Automation and Test (VLSI-TSA-DAT'05)*, 2005, pp. 271–274.

[22] T.-C Chen, Y.-W. Huang, and L.-G. Chen, "Analysis and design of macroblock pipelining for H.264/AVC VLSI architecture," in *Proceedings of 2004 International Symposium on Circuits and Systems (ISCAS'04)*, 2004, pp. II273–II276.

[23] Z.-L. He, C.-Y. Tsui, K.-K. Chan, and M.-L. Liou, "Low-power VLSI design for motion estimation using adaptive pixel truncation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 5, pp. 669–678, Aug. 2000.

[24] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 2, pp. 148–157, Apr. 1993.

[25] S. Saponara and L. Fanucci, "Data-adaptive motion estimation algorithm and VLSI architecture design for low-power video systems," *Proc. IEE on Computers and Digital Techniques*, vol. 151, pp. 51–59, 2004.

[26] J.-C. Tuan, T.-S. Chang, and C.-W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Transactions on CSVT*, vol. 12, pp. 61–72, Jan. 2002.

[27] T.-W. Chen, Y.-W. Huang, T.-C. Chen, Y.-H. Chen, C.-Y. Tsai, and L.-G. Chen, "Architecture design of H.264/AVC decoder with hybrid task pipelining for high definition videos," in *Proceedings of 2005 International Symposium on Circuits and Systems (ISCAS'05)*, 2005, pp. 2931–2934.

[28] H.-Y. Kang, K.-A. Jeong, J.-Y. Bae, Y.-S. Lee, and S.-H. Lee, "MPEG4 AVC/H.264 decoder with scalable bus architecture and dual memory controller," in *Proc. of Int. Symposium on Circuits and Systems (ISCAS'04)*, 2004.

[29] C.-Y. Tsai, T.-C. Chen, T.-W. Chen, , and L.-G. Chen, "Bandwidth optimized motion compensation hardware design for H.264/AVC HDTV decoder," in *Proceedings of 2005 International Midwest Symposium on Circuit and Systems (MWSCAS'05)*, 2005.

[30] Y.-W. Huang and etc., "A 1.3TOPS H.264/AVC Single-Chip Encoder for HDTV Applications," in *Proceedings of IEEE International Solid-State Circuits Conference (ISSCC'05)*, 2005, pp. 128–130.