doi:10.3772/j.issn.1006-6748.2022.01.005

Reconfigurable design of deblocking filter for variable block sizes①

XIE Xiaoyan(谢晓燕)*, JI Shentao②*, ZHU Yun**, YANG Kun**, XIA Xinyuan**, WANG Shuxin*
(*School of Computing Science & Technology, Xi'an University of Posts and Telecommunications, Xi'an 710121, P. R. China)
(**School of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, P. R. China)

Abstract

Based on the flexible quadtree partition structure of coding tree units (CTUs), the deblocking filter (DBF) in high efficiency video coding (HEVC) consumes a lot of resources when implemented by hardware. It is difficult to achieve flexible switching between different sizes of coding blocks. Aiming at this problem, a reconfigurable implementation of DBF is proposed. Based on the dynamic programmable reconfigurable video array processor (DPRAP) with context switch reconfiguration mechanism, the runtime flexible switching of two coding block sizes is realized. The experimental results show that the highest work-frequency reaches 151.4 MHz. Compared with the dedicated hardware architecture scheme, the resource consumption can be reduced by 28.1% while realizing the dynamic switching between algorithms of two coding block sizes. Compared with the results of HM16.0, by using a complete I-frame for testing, the average peak signal-to-noise ratio (PSNR) of the reconfigurable implementation proposed in this paper has increased by 3.0508 dB, the coding quality has improved to a certain extent.

Key words: deblocking filter (DBF), high efficiency video coding (HEVC), array processor, dynamically reconfigurable

0 Introduction

In high efficiency video coding (HEVC), deblocking filter (DBF) accounts for 20% calculation of the video coding and plays a key role in reducing compression efficiency. There are three main types of DBF implementation hardware platforms, i.e. application-specific integrated circuit (ASIC), general-purpose processor (GPP), and reconfigurable architecture. GPP has high flexibility, but the computing performance is limited, and it is difficult to adapt to high-definition real-time encoding. ASIC can well solve the problem of the low computational efficiency of GPP. Therefore, many scholars have conducted a lot of research to deploy the DBF on ASIC. In Ref.[1], a hardware scheme combining DBF and sample adaptive offset (SAO) filter was proposed, which solved the data dependence between DBF vertical filtering and horizontal filtering, and obtained higher throughput. Refs[2-4] proposed a DBF based on a multi-stage pipeline hardware architecture. Although it significantly improves the efficiency of algorithm execution, it is at the cost of more hardware resource consumption and only supports a fixed coding block size, which can not reflect the advantages of low bit rate and high quality brought by multiple coding block modes in HEVC. In Refs[5,6], an architecture for DBF that supports four coding block sizes was proposed. It can reorganize the block structure of different application requirements but at the expense of more hardware resource overhead. In addition, flexible block switching can not be realized in the runtime encoding process.

When coding with the HEVC standard, the image needs to be divided into non-overlapping coding tree units (CTUs). A quad tree-based cyclic hierarchical structure is adopted inside the CTU. The size of the coding unit (CU) varies from 64×64 to 8×8. To deal with mixed coding blocks of different sizes in HEVC, this work has developed a dynamic programmable reconfigurable video array processor (DPRAP). It can flexibly configure the logic function of the processing element (PE) according to the requirements of different coding block sizes after CTU division during operation. It has both processing flexibility and high computational efficiency. Based on the DPRAP, this work

① Supported by the National Natural Science Foundation of China (No. 61834005, 61772417, 61802304, 61602377, 61874087, 61634004) and the Shaanxi Province Key R&D Plan (No. 2021GY-029, 2021KW-16).
② To whom correspondence should be addressed. E-mail: jishentao_1201@163.com.
Received on May 11, 2021

makes the rate-distortion optimization (RDO)[7] and the sum of absolute differences (SAD) in motion estimation[8] coming true. The experimental results show that the hardware resource consumption and algorithm execution time are reduced, and the coding quality is promoted to a certain extent.

According to the requirement of DBF in HEVC, the reconfigurable mechanism of the DPRAP can well support the flexible switching of coding block sizes from 8×8 to 16×16. Based on DPRAP, this paper puts forward a dynamically reconfigurable DBF scheme. The DBF with different configurations is initialized in the instruction storage of the reconfigurable array. And then, the flexible block sizes switched in runtime coding are completed with the reconfiguration based on context switching.

The remainder of this paper is organized as follows. Section 1 introduces the DBF algorithm in HEVC, block size selection of DBF reconfiguration, as well as the design of reconfiguration and parallelization. Section 2 introduces the realization of DBF reconfiguration and parallelization. Experiments are conducted in Section 3. Finally, Section 4 draws some concluding remarks.

1 Related works

1.1 DBF algorithm in HEVC

For the balance between image coding efficiency and coding quality, HEVC has added a new flexible coding tree division structure, recursively dividing the 64×64 CTU into CUs from 64×64 to 8×8. Among them, the prediction unit (PU) and the transformation unit (TU) are divided by the CU. The DBF algorithm processes the 8×8 block boundaries in all PUs and TUs. As shown on the left of Fig. 1, the input of the DBF algorithm is a complete CTU after the image reconstruction module. When the split flag is equal to 1, the coding block is recursively divided into 4 smaller coding blocks, and then the DBF processing is performed, shown on the right in Fig. 1. The DBF operations include two steps, the filtering decision, and the filtering operation. The filtering strength of the boundary can be obtained by the filtering decision, including strong filtering, weak filtering, and no filtering. The strong filtering corrects the boundary in a large and wide range. The weak filtering correction range is relatively small. The filtering operation is based on the filtering parameters and the filtering strength, correcting the corresponding pixels in the coding block.

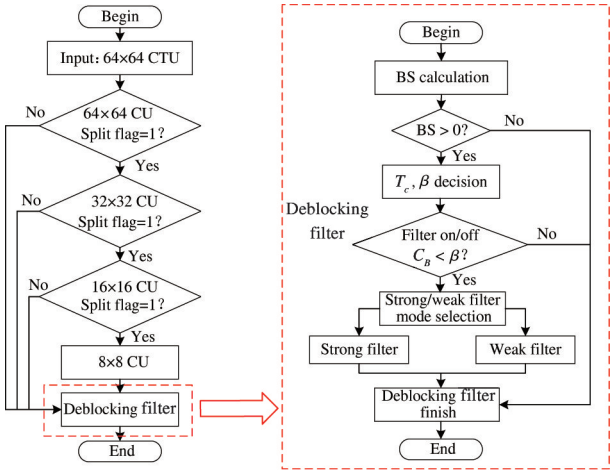

Fig. 1 Block diagram of DBF in HEVC

1.2 Blocking size selection for DBF reconfiguration

To process the four different sizes of coding blocks after CTU division in Fig. 1, Ref. [6] gives a hardware architecture for DBF, which supports 8×8 to 64×64 block sizes. Among them, the processing of 64×64 CU consumes fewer clock cycles but leads to a large increasing in power consumption. Although the power consumption is lower when processing 8×8 CU, it consumes more clock cycles. The CU sizes of 32×32 and 16×16 can meet the speed and power requirements at the same time. The scheme of Ref. [6] has a certain degree of flexibility, but it cannot achieve flexible block switching in runtime encoding. And the independent circuits design for four block sizes consumes more hardware resources. In this paper, a deployment scheme of runtime variable block sizes DBF is carried out, based on the reconfiguration mechanism of the DPRAP. It also considers the coding image quality and on-chip resource consumption.

Since the quadtree division of CTU takes a lot of coding time[9], it's important to reduce the types of CU size as much as possible. HM16.0 can be used to verify the effects of the DBF by the bitrate, the peak signal-to-noise ratio (PSNR), and the coding time under different coding block size combinations. The encoder _ lowdelay _ P _ mian configuration file is to change the size and depth of the large coding unit (LCU). By setting the three parameters, MaxCUWidth, MaxCUHeight, and MaxPartitonDepth, the 32×32, 16×16, 8×8 block sizes combination, 32×32, 16×16 block sizes combination, 16×16, 8×8 block sizes combination, 64×64, 32×32, 16×16, 8×8 block sizes combination, 64×64, 32×32, 16×16 block sizes combination, and 64×64, 32×32 block sizes combination are tested with the six standard

test sequences respectively. The PSNR, bitrate, and coding time are counted in Table 1.

It can be seen from Table 1 that the 64 × 64 and 32 × 32 block sizes combination shows no obvious advantages rather than the other block sizes combinations on PSNR and coding time, but its bitrate is the highest. So, this block combination method is excluded. Compared with the two methods of 32 × 32, 16 × 16 block combination and 16 × 16, 8 × 8 block combination, 32 × 32, 16 × 16, and 8 × 8 block sizes combination has increased the coding time by 42.96% and 50.62% respectively when the PSNR is equivalent to the other two sizes combination. But its bitrate only reduced by 3.67% and 0.73% respectively. So, this block combination method is excluded. Similarly, 64 × 64, 32 × 32, 16 × 16, and 8 × 8 block combination method can be excluded. Compared with the two methods of 64 × 64, 32 × 32, 16 × 16 block combination and 32 × 32, 16 × 16 block combination, the 16 × 16 and 8 × 8 block sizes combination has only reduced the bitrate by 2.94% and 3.85% respectively when the PSNR and coding time are equivalent to the other two sizes combination, and there is no significant difference. For further evaluation, statistics are made on the quadtree division results of four types of test sequences with different texture complexity randomly selected, as shown in Fig. 2. It is easy to see that the block sizes of 16 × 16 and 8 × 8 are chosen by HEVC more than 90%. In the other words, the 64 × 64 and 32 × 32 block sizes CUs are hardly used in coding. Therefore, just calculating DBF with 16 × 16 and 8 × 8 block sizes can meet the coding quality requirements.

Table 1 Performance parameter statistics of different block sizes combinations

Test sequences	Resolution	Parameters	32 × 32 16 × 16 8 × 8	32 × 32 16 × 16	16 × 16 8 × 8	64 × 64 32 × 32 16 × 16 8 × 8	64 × 64 32 × 32 16 × 16	64 × 64 32 × 32
Flower	352 × 288	Bitrate	1120.16	1159.36	1124.52	1098.92	1152.32	1239.92
		PSNR/dB	30.8495	30.5846	30.7816	30.9209	30.6090	30.2710
		Time/s	81.651	58.246	55.284	73.843	56.208	53.293
Stefan	352 × 288	Bitrate	880.96	918.80	890.00	877.24	938.24	1038.08
		PSNR/dB	31.8241	31.5889	31.7537	32.1118	31.9513	31.2930
		Time/s	91.529	64.140	58.495	84.522	68.818	59.254
News	352 × 288	Bitrate	295.40	305.80	299.12	293.48	315.76	341.40
		PSNR/dB	35.3611	35.1613	35.1946	35.3792	35.5702	34.7872
		Time/s	57.307	38.835	39.248	70.416	41.415	37.938
Average	352 × 288	Bitrate	765.51	794.65	771.21	756.55	802.11	873.13
		PSNR/dB	32.6782	32.4449	32.5766	32.8040	32.7102	32.1171
		Time/s	76.829	53.740	51.009	76.260	55.480	50.162

1.3 Reconfiguration and parallelization of DBF

To complete the DBF process of subsection 1.1, the runtime reconfigurable mechanism must be involved to ensure a high resource utilization rate, in addition to parallel array acceleration. So, the DBF is deployed on the DPRAP, a dynamic programmable reconfigurable array processor is developed, as shown in Fig. 3. It is composed of a global controller, process element group (PEG) with 4 × 4 PEs, data input memory (DIM), data output memory (DOM), and an H-Tree hierarchical configuration network (HCN). The global controller decides the operating mode and chooses the appropriate functions for the PEs by issuing the configuration information. The PEG fetches data from DIM, performs the corresponding operation according to configuration information, and outputs the results to DOM. During the execution of the DBF algorithm, the current operation modes and status of each PE are collected by the global controller via HCN. The next operation mode is judged according to the procedure. If they are different from current modes, the new configuration information

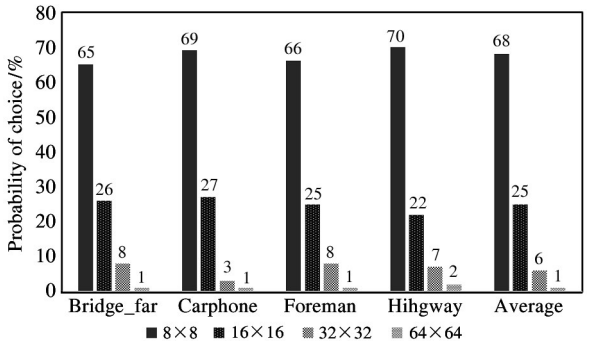

Fig. 2 Statistics of block division under different test sequences

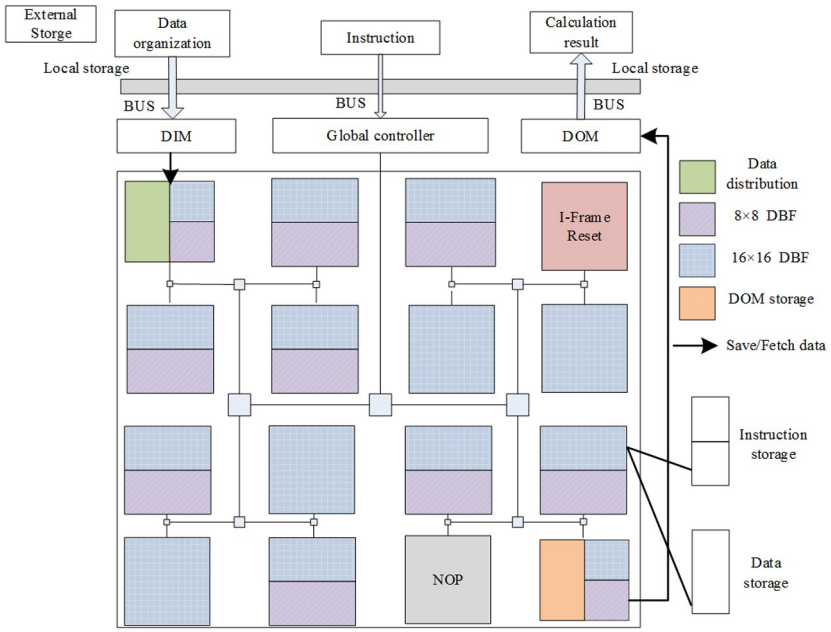

Fig. 3 Schematic diagram of the reconfigurable DBF algorithm network array structure

is issued. Otherwise, only data will be distributed. So as to realize the runtime reconfiguration of the PE function. According to the procedure of the DBF algorithm, the DPRAP is dynamically configured continuously. Different computing tasks are flexibly handled, the free switching of the 8 × 8 and 16 × 16 coding block sizes DBF algorithms is a compact implementation on the same PEG. Computing efficiency and resource utilization are all considered.

When mapping a reconfigurable DBF algorithm, firstly, initializing a frame of image data after image reconstruction in DIM and the state information of which coding block size is executed in the data storage of PE00, where the selection of the coding block size is determined by the intra-frame prediction algorithm according to the SAD value. Secondly, the algorithm instructions of the two configurations of 8 × 8 and 16 × 16 are initialized in different instruction stores of the same PE. When it is necessary to update the information of certain configuration storage, through the collection instruction issued by the global controller, the PE will feedback the address where the status information is placed to the global controller through the HCN, and the global controller sends corresponding configuration information to the processing element according to the collected status information, to realize the dynamic switching of two different coding block algorithms. When the configuration information of the 8 × 8 coding block is issued, the 8 × 8 coding block is first divided into two parts: the vertical boundary is in the unit of 8 × 4, and the horizontal boundary is in the unit of 4 × 8. Then, using the natural parallel characteristic of the reconfigurable array processor, the two coding blocks are processed at the same time, so as to realize the two-way parallelism of the deblocking filtering algorithm. The specific reconstruction process is shown in Fig. 4. Among them, when a block is processed, the status information stored in the #510 address in PE00 will be updated.

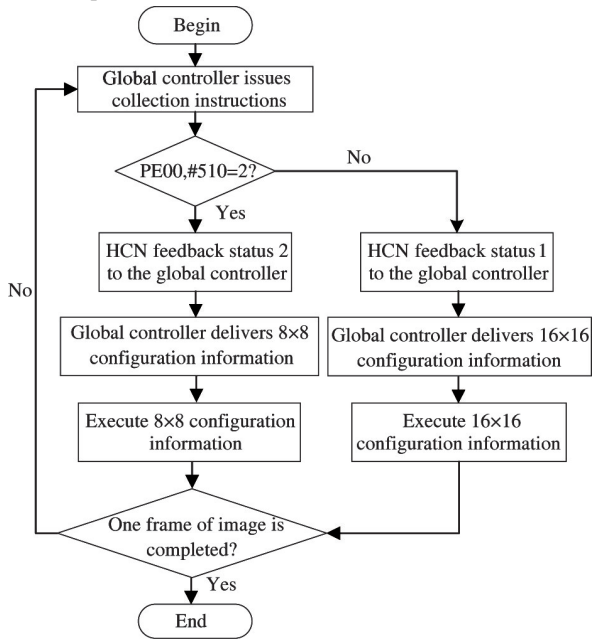

Fig. 4 DBF algorithm reconfigure process

2 The realization of DBF reconfiguration and parallelization

2.1 Parallelization of DBF

Taking the 8 × 8 size coding block DBF of the in-

tra-frame loop as an example, the process is implemented based on 4 × 4 PEs, as shown in Fig. 5. Firstly, PE00 of the DBF algorithm takes out an 8 × 8 coding block from DIM. Secondly, according to the order of horizontal filtering and then vertical filtering in the standard, PE00 divides the 8 × 8 coding block into upper and lower 4 × 8 size coding blocks in the horizontal direction. Using the natural parallel characteristic of the reconfigurable array processor, PE01 and PE10 make filtering decisions on the two parts of the coding block at the same time, PE02 and PE11 perform corresponding filtering operations. And PE12 integrates the two parts of the coding block after horizontal filtering into one coding block. So far, the horizontal filtering is completed. Then, vertical filtering decisions and filtering operations are performed like horizontal filtering. Finally, the filtered coding block is passed to the DOM through PE33 to restore the image.

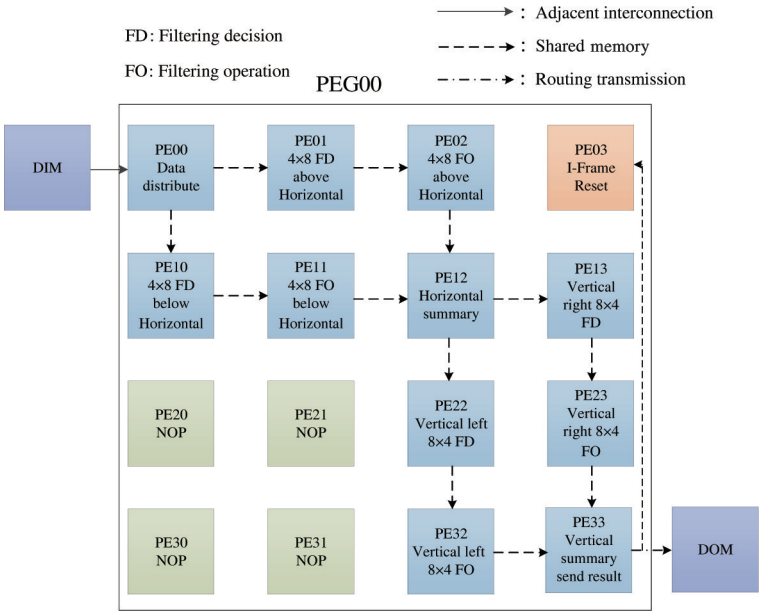

Fig. 5 The mapping process of 8 × 8 DBF algorithm

2.2 Reconfiguration deployment of DBF

The purpose of block size reconfigured in DBF runtime is to meet the requirements of HEVC, as analyzed in subsection 1.2. The reconfigure scheme of DBF based on different coding block sizes is shown in Fig. 6. Firstly, instructions of 16 × 16 and 8 × 8 coding blocks are stored in partitions. And then, the HCN issues the states to the array processor, by which the coding block size is specified, and makes the program counter (PC) pointed to the corresponding code. The PC1 matches the flag bit when executing the 16 × 16 code, and PC2 matches the flag bit when executing the 8 × 8 code. When the reconfigurable instruction is executed, the corresponding configuration information is read from the memory, and the PC value is modified. The first address of the configuration information is written into it. The PE performs the corresponding operation according to the configuration information, thereby achieving 16 × 16 and 8 × 8 flexible switching of the coding block sizes.

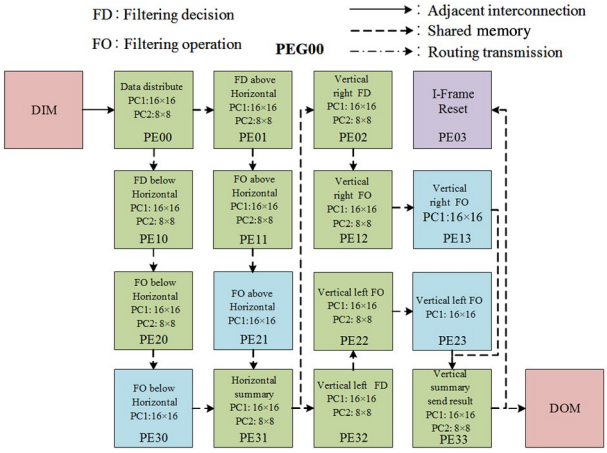

Fig. 6 The reconfigure and mapping process of DBF algorithm

Since the code storage capacity of each PE in the reconfigurable array processor is limited, PE02, PE11, PE20, and PE23 may cause overflow problems when storing 16 × 16 filtering operation codes. In the current PEG, PE12, PE13, PE21, and PE30 are idle PEs, and part of the 16 × 16 filtering operation codes can be stored here to solve the problem of code overflow caused by storage. The specific execution steps of the reconfigurable DBF algorithm are as follows.

Firstly, the instructions of the two coding block si-

zes of 8 × 8 and 16 × 16 are partitioned and stored. The algorithm instructions of the 16 × 16 coding block are initialized at addresses 0 – 254 of each PE instruction memory, the instructions of the 8 × 8 coding block are initialized in addresses 255 – 511 of each PE instruction memory. And store the PC flags '0' and '255' of the 16 × 16 and 8 × 8 coding blocks in the #500 and #501 addresses of PE00 in PEG00, respectively.

Secondly, the global controller issues a collection information instruction, and the status information of which coding block size is executed will be feedback to the global controller through HCN, and the global controller issues corresponding configuration information according to the collected status information.

Finally, when HCN sends the status information to 1, the PC value points to the flag PC1 corresponding to 16 × 16, and the global controller issues the data instruction of the 16 × 16 size coding block; When the status information sent by the HCN is 2, the PC value points to the mark PC2 corresponding to 8 × 8 size coding block, and the global controller executes the data sending instruction of the 8 × 8 coding block, thereby completing the reconfigure selection of different modes. The coding block after DBF will be transmitted to the DOM by PE33 through a routing transmission mechanism, then PE33 will send a handshake signal '222' to PE00, and PE00 will start to execute the next coding block after receiving the feedback information.

3 Experiment results and discussion

This paper is based on the dynamic reconfigurable array structure to verify the realization of DBF reconfigurable. The method is as follows. Firstly, store the test data into DIM, and then initialize the instructions into the instruction memory, and finally perform simulation verification on the reconfigurable array structure with the Questasim 10.1d. After the functional simulation of the DBF reconfigure scheme, it is synthesized through Xilinx's ISE14.7 development environment, and finally tested and verified based on the video array processor prototype system development board built on the BEEcube's BEE4. Through the establishment of the performance model, the HM16.0 standard test sequence is used to test on the hardware platform, and the results of five test sequences are given. PSNR and structural similarity index measurement (SSIM) are two commonly used image quality evaluation indicators. And analyzing the test image results of a complete I-frame, results are shown in Table 2. Compared with the results of HM16.0 testing a complete I-frame, the average PSNR value of the scheme proposed in this paper has improved by 3.0508 dB, and the average SSIM value is 0.9975. Fig. 7 shows the test results of salesman and bridge-far with a resolution of 176 × 144 on the reconfigurable video array processor. It can be seen from the figure that the test results have a good visualization effect.

Fig. 7 Test results of salesman and bridge-far

Table 2 Test performance analysis

Test sequences	HM16.0 PSNR/dB	This work PSNR/dB	△PSNR/dB	SSIM
Carphone	33.1804	35.3536	2.1732	0.9966
Highway	35.5497	38.0036	2.4539	0.9996
Foreman	33.0144	37.6829	4.6685	0.9953
Bridge-far	35.2603	38.8836	3.6233	0.9992
Container	34.2640	36.5990	2.3350	0.9969
Average	34.2538	37.3045	3.0507	0.9975

In this paper, the DBF based on the reconfigurable array processor supports 2-way parallel implementation. Fig. 8 shows the time comparison of the DBF in parallel and serial processing of different sizes of coding blocks. For the coding block of 8 × 8 size, the speed of filtering decision-making and filtering operations by using a 2-way parallel processing method is 28.0%, higher than that of the serial method, and the speed-up ratio is 1.39. For the coding block of 16 × 16 size, the speed of the 2-way parallel processing method is 29.3%, higher than that of the serial method, and the speed-up ratio is 1.41.

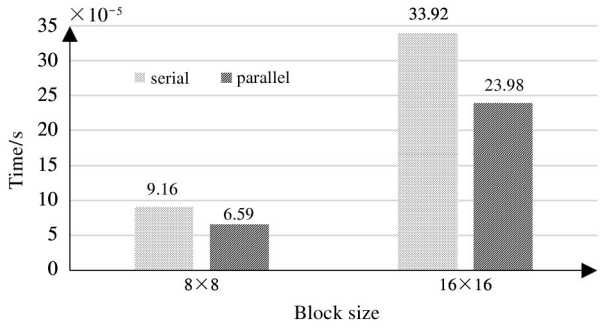

Fig. 8 Serial/parallel time comparison

Using Xilinx's ISE14.7 development environment to synthesize the design, it can be seen from Table 3

that under the same hardware technology, Ref. [6] proposed a parallel and pipelined HEVC deblocking filter hardware architecture. Although the throughput is high and processing a CTU only consumes 96 clock cycles, it is at the cost of more hardware resource consumption, and the highest frequency is reduced by 17.4% compared with the scheme proposed in this paper. The maximum frequency of Ref. [10] is better and only consumes 172 clock cycles, but its hardware resource consumption is 1.8 times that of this paper. Ref. [11] proposed a combined DBF for H.264 and HEVC with four parallel filters. Its hardware resource overhead is less, but under the same throughput as this paper, the maximum frequency has been reduced by 28.7%, and the clock cycle is 2.1 times that of this paper. Compared with Ref. [12], the proposed method has less hardware resource consumption and clock cycles, but the maximum frequency is lower and it cannot achieve flexible switching between different coding block algorithms. Ref. [13] and Ref. [14] respectively used different hardware processes, among which the hardware architecture proposed in Ref. [13] uses a high degree of parallelism to improve throughput, and its maximum frequency can reach 250 MHz, and the clock cycle is better than this paper, but the hardware resource consumption (LUTs and REG) is more than three times that of this paper. Ref. [14] proposed a parallel implementation method of DBF based on the 16 × 16 coding block. Although it can achieve higher computing performance at lower energy costs, compared with this paper, when the maximum frequency is close, the hardware resource consumption (LUTs and REG) is increased by 59%.

Table 3 Comparison of computing performance

References	Platform	LUTs	REG	LUT-FF	Maximum frequency/MHz	Clock cycles/CTU	Throughput/fps	Reconfigure
Ref. [9]	Virtex-6	35796	21134	12728	125	96	4000p@50	No
Ref. [10]	Virtex-6	62089	17146	10594	339.7	172	-	No
Ref. [11]	Virtex-6	5236	1547	-	108	7680	1080p@30	No
Ref. [12]	Virtex-6	1398	441	-	140	1027	2048p@60	No
Ref. [13]	XC7Z045 FFG900-2	5456	83070	-	250	2252	2160p@50	No
Ref. [14]	XC7Z045 FFG900-2	49040	16779	-	153.3	-	-	No
This work	Virtex-6	31730	9664	8656	151.4	3624	1080p@30	Yes

4 Conclusion

Aiming at the problem that the hardware implementation of DBF under the flexible quad-tree partition structure of the CTU consumes a lot of resources and it is difficult to achieve flexible switching between different sizes of the coding block, this paper processes a dynamically reconfigurable DBF implementation scheme by using DPRAP. Then, the scheme is verified by using Questasim 10.1d simulation software and the Xilinx Virtex-6 FPGA hardware platform. Based on the reconfiguration mechanism of context switching, this scheme realizes DBF algorithms flexible switching between the 8 × 8 and 16 × 16 coding blocks. The experimental results show that the maximum frequency of the reconfigurable implementation proposed in this paper can reach 151.4 MHz. Compared with the Ref. [9], the hardware resource consumption can be reduced by 28.1% while realizing flexible switching of coding block algorithms of different sizes. Compared with the results of HM16.0 testing a complete I-frame, the average PSNR has improved by 3.0508 dB, and its coding quality is improved.

References

[1] SHEN WW, FAN Y, BAI Y, et al. A combined deblocking filter and SAO hardware architecture for HEVC[J]. *IEEE Transactions on Multimedia*, 2016, 18(6): 1022-1033

[2] CHRISTOPHER P R, SATHASIVAM S. Five-stage pipelined dual-edge deblocking filter architecture for H.265 video codec[J]. *IEICE Electronics Express*, 2019, 16(22): 1-6

[3] KIM H M, KO J G, PARK S. An efficient architecture of in-loop filters for multicore scalable HEVC hardware decoders[J]. *IEEE Transactions on Multimedia*, 2017, 20(4): 810-824

[4] HSU P K, SHEN C A. The VLSI architecture of a highly efficient deblocking filter for HEVC systems[J]. *IEEE Transactions on Circuits and Systems for Video Technology*, 2016, 27(5): 1091-1103

[5] BALDEV S, ANUMANDLA K K, PEESAPATI R. Scalable wavefront parallel streaming deblocking filter hardware

for HEVC decoder[J]. *IEEE Transactions on Consumer Electronics*, 2019, 66(1): 41-50

[6] BALDEV S, RATHORE P K, PEESAPATI R, et al. A directional and scalable streaming deblocking filter hardware architecture for HEVC decoder[J]. *Microprocessors and Microsystems*, 2021, 84(12): 104029

[7] YANG K, JIANG L, XIE X Y, et al. Dynamic reconfigurable implementation of rate distortion optimization algorithm in HEVC[J]. *Computer Engineering and Science*, 2020, 42 (11):57-64

[8] JIANG L, WU X, CUI J X, et al. Dynamic reconfigurable implementation of SAD algorithm in HEVC motion estimation[J]. *Journal of Beijing University of Posts and Telecommunications*, 2018,41(4):37-43 (In Chinese)

[9] ZAKI F, MOHAMED A E, SAYED S G. CtuNet: a deep learning-based framework for fast CTU partitioning of H265/HEVC intra-coding[J]. *Ain Shams Engineering Journal*, 2021,12(2):1859-1866

[10] BALDEV S, SHUKLA K, GOGOI S, et al. Design and implementation of efficient streaming deblocking and SAO filter for HEVC decoder[J]. *IEEE Transactions on Consumer Electronics*, 2018, 64(1):127-135

[11] OZCAN E, ADIBELLI Y, HAMZAOGLU I. A high performance deblocking filter hardware for high efficiency video coding[J]. *IEEE Transactions on Consumer Electronics*, 2013, 59(3):714-720

[12] DINIZ C M, SHAFIQUE M, DALCIN F V, et al. A deblocking filter hardware architecture for the high efficiency video coding standard[C]// 2015 Design, Automation and Test in Europe Conference and Exhibition, Grenoble, France, 2015: 1509-1514

[13] AYADI L A, BOUBAKRI W, LOUKIL H, et al. A hardware-efficient parallel architecture for HEVC deblocking filter[C]// 2019 16th International Multi-Conference on Systems, Signals and Devices, Istanbul, Turkey, 2019: 669-673

[14] JIANG L, YANG Q, ZHU Y, et al. A parallel implementation of deblocking filter based on video array architecture for HEVC[C]// The 7th International Green and Sustainable Computing Conference, Hangzhou, China, 2016: 1-7

XIE Xiaoyan, born in 1972. She is a professor in Xi'an University of Posts and Telecommunications. She received her M. S. degree from Computer Science Department of Xidian University in 2002. She received her B. S. degree from Nanjing University of Science and Technology in 1995. Her research interests include the design of parallel computing architecture and multimedia algorithms for parallel processing.