IEIE Transactions on Smart Processing and Computing, vol. 3, no. 5, October 2014
http://dx.doi.org/10.5573/IEIESPC.2014.3.5.251

IEIE Transactions on Smart Processing and Computing

Efficient Token Flow Design for the MPEG RMC Framework

Li Cui[1,2], Sowon Kim[3], Hyungyu Kim[4], and Euee S. Jang[4]

[1] Department of Computer & Software, Hanyang University / Seoul, South Korea Lcui2000@gmail.com
[2] Department of Computer Sicence & Technology, College of Science & Engineering, Yanbian University / Yanji City, China cuili@ybu.edu.cn
[3] Department of Computer & Software, Hanyang University / Seoul, South Korea {mogyak, khg.dmlab}@gmail.com
[4] Division of Computer Science & Engineering, Hanyang University / Seoul, South Korea esjang@hanyang.ac.kr

***** Corresponding Author: Euee S. Jang

Received January 15, 2014; Revised March 17, 2014; Accepted July 28, 2014; Published October 31, 2014

** Regular Paper*

***Abstract*:** This paper proposes an efficient token flow design methodology for a decoder in the MPEG Reconfigurable Media Coding (RMC) framework. The MPEG RMC framework facilitates a decoder to be configured with a set of modules called functional units (FUs) that are connected by tokens. Such a modular design philosophy of the MPEG RMC framework enables the reusability and reconfigurability of FUs. One drawback of the MPEG RMC framework is that the decoder performance can be affected by increasing the token transmissions between FUs. The proposed method improves the design of the FU network in the RMC framework toward real-time decoder implementation. In the proposed method, the merging of FU, the separation of token flow, and the merging of token transactions are applied to minimize the token traffic between FUs. The experimental results of the MPEG-4 SP decoder show that the proposed method reduces the total decoding time by up to 77 percent compared to the design of the RMC simulation model.

***Keywords*:** Reconfigurable media coding, Token transaction, Functional unit, FU network design

1. Introduction

Recently, standard video codecs have become increasingly complex to meet the increasing compression efficiency requirements of emerging video applications. The increasing complexity of video codecs results in a greater implementation burden on the developers. Although a large amount of the functionality of existing codecs is shared by new codecs, many of the coding tools of standardized codecs must be redeveloped from scratch. The MPEG reconfigurable media coding (RMC) framework is one of the first standards to tackle this problem by describing the codecs using a set of modular tools called functional units (FUs) [1]. The modular design of video codecs in the MPEG RMC framework makes possible a video coding standard, which can reuse many similar functions from other video coding standards.

Continuous performance improvement is essential for making the RMC framework feasible for practical implementation. Several approaches for the RMC framework have been made to achieve better codec performance (e.g., reduced decoder complexity, flexible implementation, etc.). Previous research on performance enhancement is mainly hardware-oriented. For example, a luminance block splitting method that splits an 8×8 luminance block into 4×4 blocks on FPGA enhances the decoding speed of the MPEG-4 SP decoder by up to 50.5 percent [2]. Another FPGA-based method to reconfigure two Inverse Quantization (IQ) FUs in MPEG-2 and H.263 for MPEG-4 has been described [3]. In addition, Yviquel et al. [4] described two schedule strategies, including round-robin and data-driven/demand-driven for multicore processing, and claimed that their performance simulated on MPEG-4 SP and MPEG-4 AVC decoders could achieve decoding speeds of up to three times faster on four cores than on a single core. Although a few approaches based on specific architectures have been described, the general methodologies without architectural constraints have rarely been considered for the fundamental enhancement of the performance of the RMC codec at the modular design level.

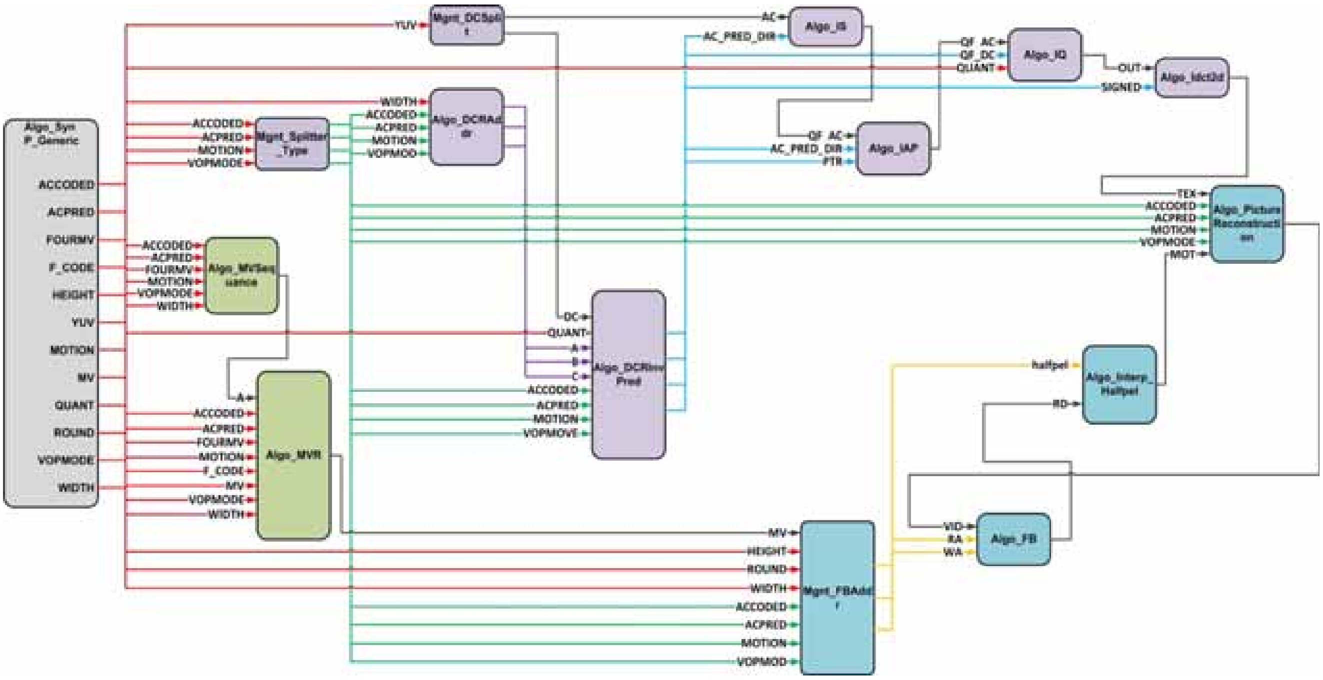

Fig. 1. FU network of MPEG-4 SP decoder.

In the RMC framework, a video decoder is composed of FUs, whose connections construct a dataflow network called the FU network. During the decoding process, the FUs consume input tokens and produce output tokens. The token traffic (or the number of token transactions in the FU network) directly affects the decoding performance by its complexity overhead. Despite the design space exploration based on minimizing the buffer size to optimize the decoding process in [5], the token traffic for efficient FU network implementation has not been studied. Although there have been some studies on the development of tools for automatic analysis of dataflow network and its model of computation in [6-9], fundamental FU network design methodologies to reduce the decoder complexity have not been considered seriously in the MPEG RMC framework. A general methodology for FU network design is needed to minimize the complexity overhead caused by the token traffic and to enhance the decoding performance in the RMC framework.

This paper proposes three approaches (i.e., FU merge, token separation, and token merge) to reduce the complexity of the RMC decoder. In the proposed approaches, unnecessary token transactions within the FU network are removed to enhance the computational efficiency of the decoder. This paper is organized as follows. Section 2 introduces the features and issues of MPEG RMC. Section 3 discusses four design issues to enhance the efficiency of FU network design and a practical case is analyzed based on those issues. Section 4 introduces the proposed method through several examples. Section 5 shows the experimental results of the MPEG-4 SP decoder using the proposed method and the paper is concluded in Section 6.

2. MPEG RMC

In MPEG RMC, a video decoder is separated into two parts: one for FUs and the other for configuration information to compose the FUs named as decoder description. Decoder description includes parser information for parsing the encoded media data, such as video bitstreams and FU network description (FND) for describing how FUs are connected.

For example, the RMC-based MPEG-4 SP decoder is designed using the FUs described in the media tool library (MTL) [10]. The graphical representation of FND in Fig. 1 shows the dataflow network design of the RMC-based MPEG-4 SP decoder. As shown in Fig. 1, the FU network design is constituted by 15 FUs: one FU for the parsing process, 8 FUs for the texture decoding process, 2 FUs for the motion vector decoding (MVD) process, and 4 FUs for the motion compensation process. MPEG RMC has five significant features.

Platform-independence: The MPEG RMC framework is not designed to support a specific target platform. Such platform-independent specification enhances the flexibility in implementing the decoders on various platforms.

Reusability: The reusability of a FU can enhance its usability and result in a decrease in the decoder development cost.

Reconfigurability: Through the reconfigurability in RMC, it is possible to reconfigure a decoder to support additional profiles by adding and removing FUs.

Testability: The conventional coding standards in MPEG are specified at the codec level. On the other hand, RMC provides modularized specifications, which enables the easy debugging of codec implementation.

Design-time vs. Run-time: The FUs described in MTL can be used in two distinctive approaches: design-time approach and run-time approach. In the design-time approach, the FUs are used as a reference code in decoder implementation during the design time. Therefore, the decoder complexity can be optimized in a flexible way during implementation. In the run-time approach, FU implementations are configured into a decoder from a decoder description in run time; hence, FUs in MTL are used directly and the efficiency of decoder development can be enhanced.

Some issues remain to be considered except for the standardized methodologies in MPEG RMC.

- The granularity of FU means the degree to which a decoder is subdivided into a number of FUs. The excessively low granularity of FU can result in a large number of tokens transmitted in the FU network. Excessively high granularity of the FU can affect the reusability and reconfigurability of FU, such as a conventional decoder assumed as an FU. Therefore, an optimization methodology to design FUs with proper granularity should be considered toward practical applications.
- The on-the-fly configuration is to configure a decoder on the fly from a given decoder description. Although an on-the-fly configuration can provide the rapid deployment of reconfigured FUs and flexible support for various coding methods, several issues need to be addressed before it is adopted for RMC. The design of FU needs to consider proper granularity and good applicability to support the immediate adaption of different coding methods. The framework needs to provide an efficient mechanism toward a range of operations (e.g., token transaction, FU call, etc.) due to the difficult optimization of decoder complexity in run time.

3. Design Issues for an Efficient Network

As described in the previous section, the token traffic is related directly to the complexity of a decoder in the RMC framework. The complexity of a video decoder can be formulated with the number of computations in the decoder (C_D), which is

$$C_D = F\left(C_{FU}, C_{TT}, C_{FO}\right) \tag{1}$$

where C_{FU} is the number of computations required for the execution of FUs, C_{TT} is the number of computations for the token transactions (i.e., output tokens from FUs), and C_{FO} is the number of computations required within the framework as an overhead. The reduction of the C_{FU} and C_{TT} directly influences the performance of the decoder based on an RMC framework implementation, whereas the reduction of C_{FO} can be effective only in a subset of RMC framework implementations.

3.1 Issues

An analysis of the current FU network design in the MPEG RMC standard showed that unnecessary token transactions can be reduced to achieve a further reduction of C_{FU} and C_{TT}, which can be described in the following four scenarios: (i) inefficiently designed FU granularity, (ii) unnecessary FU calls, (iii) inefficient token definition, and (iv) a lack of support for block data in token transactions.

Inefficiently designed FU granularity: A video decoder in MPEG RMC is generally more complex than that of the conventional decoder as follows:

$$C_{D\_conventional} \le C_{D\_RMC}\left(N_{FU}\right) = \sum_{i=0}^{N_{FU}-1} \left(c_{FU_i} + c_{TT_i}\right) \tag{2}$$

where C_{D_RMC} is the complexity of a video decoder in MPEG RMC and $C_{D\_conventional}$ is the complexity of the conventional decoder. In Eq. (2), N represents the number of FUs, C_{FUi} denotes the number of computations required by the ith FU, and C_{TTi} denotes the number of computations needed to process the output tokens from ith FU. Under the conditions shown in Eq. (2), the optimal N can be determined as follows:

$$\mathrm{N}_{\mathrm{optimal}} = \arg\min_{\mathrm{j}} \mathrm{C}_{\mathrm{D\_RMC}}\left(\mathrm{j}\right) \tag{3}$$

where j denotes the number of FUs with proper FU granularity. Because the decision of the number of FUs is affected by several factors (e.g., the number of cores, memory size, the speed of CPU, etc.), finding a global $N_{optimal}$ in Eq. (3) is an NP-hard combinatorial problem [11]. Some exploratory studies on the determination of $N_{optimal}$ have been reported [12, 13], which remains as a challenging topic. Heuristic approaches as the most common way to solve NP-hard problems are often used to determine the local $N_{optimal}$.

Unnecessary FU calls: C_{FU} can be defined as follows:

$$C_{FU} = \sum_{i=0}^{N_{FU}-1} c_{FU_i} \tag{4}$$

When called, an FU processes the necessary tokens for the decoding process. During the decoding process, there may be different decoding modes (e.g., intra and inter modes), and the number of computations C_{FU} can be defined based on the decoding mode as follows:

$$C_{FU} = C_{FU^{intra}} + C_{FU^{inter}} = \sum_{i=0}^{N_{FU_1}-1} c_{FU_i^{intra}} + \sum_{i=0}^{N_{FU_2}-1} c_{FU_i^{inter}} \tag{5}$$

In general, the FUs called in the intra mode are considered to be a subset of FUs in the inter mode. Therefore, N_{FU1} is smaller than N_{FU2}. Although the decoding process can work even when $N_{FU1}=N_{FU2}$, to reduce C_{FU}, N_{FU1} should be minimized by excluding the unnecessary FUs that do not contribute to the intra-decoding process. Similar cases also exist in decoder implementations based on the MPEG MTL; for example, the MPEG-4 SP decoder is designed to call FUs related to the motion vector difference (MVD) decoding process, even in the intra-decoding process.

Inefficient token definition: C_{TT} can be formulated as a function of the amount of total tokens (A_T) and the number of token transactions (N_{TT}), because C_{TT} is proportional to A_T and N_{TT}, which is illustrated as follows:

$$C_{TT} = G\left(A_T, N_{TT}\right) \quad (6)$$

The reduction of A_T and N_{TT} will result in a reduction of C_{TT}. N_{TT} can be reduced by removing the unnecessary token transactions or by reducing A_T. A_T can be reduced by avoiding the generation of redundant or unnecessary tokens from the FU network. Therefore, having an efficient token definition can reduce both A_T and N_{TT}. For example, in the MVD decoding process in the MPEG-4 SP decoder based on MPEG MTL, A_T and N_{TT} can be reduced by improving the efficiency of the token definition.

A lack of support for block data in token transactions: The number of tokens per token transaction (A_T /N_{TT}) can affect the decoder complexity in the RMC framework. During the video decoding process, block data combined with a different number (e.g., 8×8, 16×16, etc.) of elements is used as much as individual elements. The token throughput (A_T/N_{TT}) can be increased by sending the block data immediately instead of splitting the block data into elements. Therefore, the block-basis token transaction can reduce the NTT and consequently increase the efficiency of the FU network design.

3.2 Case Study

According to the design in Fig. 1, the MPEG-4 SP decoder was implemented on codec configuration simulation model (CC-SiM) that is a configurable codec framework based on C/C++ [14]. The implementation (hereafter called the reference implementation) includes 12 FUs in MTL as described in Fig. 1. One granular FU is implemented, which is consistent with three FUs (Mgnt_FBAddr, Algo_FB, and Algo_InterpHalfpelBilinearRoundingControl) in MTL.

The token flow in the reference implementation was analyzed based on the issues explained in the previous section. The MPEG-4 conformance bitstreams was taken as an input that are encoded by CIF sequences and the details of token analysis are enumerated as follows.

Inefficiently designed FU granularity: The MVD process is separated into two functionalities (Algo_MVSequence_LeftAndTopAndTopRight, hereafter called Algo_MVSequence, and Algo_MVR_MedianOfThree- LeftAndTopAndTopRight, hereafter called Algo_MVR) that are specific to the MPEP-4 Part2 standard. Despite the fact that the design rarely affects the decoding performance in the design-time configuration, it results in an increase of the unnecessary token transmissions in run-time configuration. Therefore, the granularity of FUs should be determined carefully in run-time configuration.

Unnecessary FU calls: In the intra mode, although a number of input tokens (1190 for Algo_MVSequence and 1191 for Algo_MVR) are transmitted per frame, there is no operation in those FUs because of no output tokens generated. Unlike in a design-time configuration, calling FUs unnecessarily can generate unnecessary token transmissions, which can increase the decoding time in run-time configuration.

Inefficient token definition: In some cases, some token definition is not efficient for run-time configuration even when they sufficiently satisfy the requirements of design-time configuration. For example, MOTION, FOURMV, and VOPMODE tokens are defined to represent the macroblock (MB) type or VOP type and all of them are transmitted per MB, even though they can be reduced to an average of one token transmitted per MB. Efficient token definition for run-time configuration is necessary to reduce the unnecessary token transmissions, which is critical toward the real-time decoder performance.

A lack of support for block data in token transactions: 7 block tokens are used as input tokens for the texture decoding process and each of them is transmitted 64 times on element basis. Although element-basis transmission for block token can rarely affect the decoding performance in design-time configuration, block-basis transmission for block token can increase token throughput to decrease the decoding time in run-time configuration.

4. Proposed Method

As suggested by the four problems mentioned in the third section, this paper proposes three approaches and explain them through several demonstrated examples in this section, including FU merge/split, token separation, and token merge approaches.

4.1 The FU Merge/Split Approach

One method to minimize C_D based on Eq. (3) is to split or merge FUs. Splitting FUs in a decoder has been a basic principle in MPEG RMC and the methodology is summarized in [15]. FUs can be merged using the granular FU concept, which was proposed in MPEG [16, 17]. The granular FU design provides a method to merge the existing FUs into FUs with higher granularity in MPEG RMC. In addition, it is important to merge the algorithmic FUs with the proper granularity by maintaining their own independent functionality. For example, Algo_MV Sequence and Algo_MVR FUs, which are the partial implementation of the MVD process in the reference implementation, were merged. These FUs can be replaced by one granular FU (Granular_MVR) and Fig. 2 presents the design of Granular_MVR. A similar attempt in [7] was made to speed up the MPEG-4 SP decoder by reconfiguring the FU network with merging four algorithmic FUs (i.e., Algo_IS, Algo_IAP, Algo_IQ, Algo_Idct2D) into an FU at the cost of abandoning the reusability of the individual FUs.

4.2 The Token Separation Approach

Depending on the decoding mode, the FUs may be called unnecessarily without any operation despite

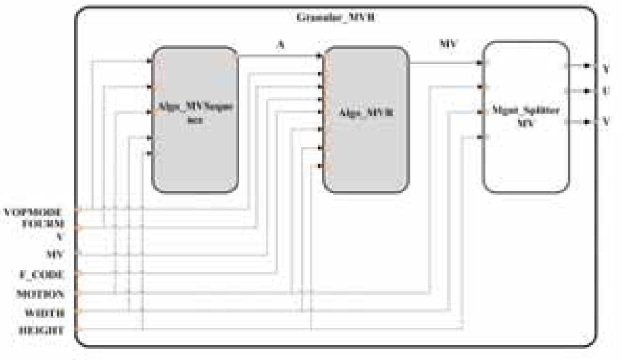

Fig. 2. Inner network diagram of Granular MVR.

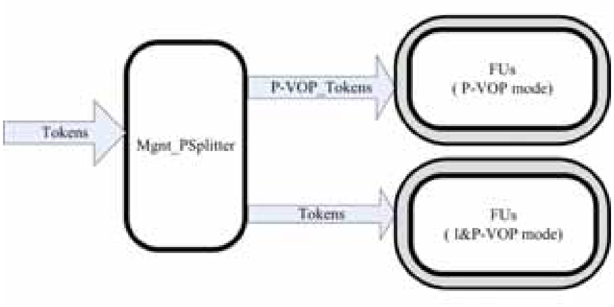

Fig. 3. Design of the token separation approach.

consuming input tokens. To avoid unnecessary FU calls, FUs can be called separately based on the decoding mode. This can be achieved by transmitting the input tokens separately to the corresponding FUs. Therefore, to avoid Algo_MVSeque- nce and Algo_MVR FUs to be called unnecessarily in the intra mode, a data management FU (Mgnt_PSplitter) was designed to perform the proposed method, as shown in Fig. 3.

4.3 The Token Merge Approach

FU design can be improved by merging the tokens in the following two cases. In the first case, the same information carried in several tokens can result in redundant or unnecessary tokens in the FU network. This can be avoided by merging the redundant or unnecessary tokens to reduce A_T and N_{TT}. Therefore, the VOPMODE, MOTION and FOURMV tokens listed in Table 1 can be merged into a MBTYPE token for per MB and VOPTYPE token for per frame, as listed in Table 1.

The second case is the element-basis transmission of block data, which can be improved by merging and transmitting tokens on a block basis, as discussed in the previous section. Therefore, to improve the texture decoding process in the reference implementation, 7 block tokens were designed to be transmitted on a block basis.

5. Experimental Results

The proposed approaches were applied to the FU network of the MPEG-4 SP decoder in MPEG MTL. The FU network was modified according to the proposed approaches described in Section 4. Table 2 lists the changes in the FU network design. The proposed approaches were applied in two stages to measure the complexity reduction of each stage separately. This is because the block token merge approach was predicted to significantly affect the decoder complexity.

Table 1. New Token Definition for MV Decoding FUs Based on Token Merge Approach.

VOPTYPE		MBTYPE	
Code	Meaning	Code	Meaning
0	P	5	not coded
0	P	0	Inter
0	P	1	Inter+q
0	P	2	Inter4V
0	P	3	Intra
0	P	4	Intra+q
1	I	3	Intra
1	I	4	Intra+q

Table 2. Changes in the FU network of the MPEG-4 SP decoder by the proposed approaches.

Approach	Detail	Stage	N_{FU}
	Before applying the approaches below	Reference implementation	12
Block Token Merge	Block and AC tokens transmitted on block basis	BM implementation	12
FU Merge	Three FUs (Algo_MVSequence, Algo_MVR, Mgnt_SplitterMV) merged into one granular FU (Algo_MVR)	MVDR implementation	10
Token Separation	a) Four tokens (WIDTH, HEIGHT, VOPMODE, MOTION) separated. b) FOURMV token modified		
Token Merge	VOPMODE, MOTION, and FOURMV tokens merged into MBTYPE and VOPTYPE tokens		

To evaluate the performance of the proposed design, the reference implementation, which is described in Section 3.2, was used as the basis of the experiment. Based on the reference implementation, two approaches described in Table 2 were implemented: hereafter called the block merge (BM) implementation and the MVD redesign (MVDR) implementation. Tests verifying the performance of the proposed design were performed on eight test sequences retrieved from [18], which consist of the I-only sequences (jvc000, san000, and san001) and the IPPP sequences (hit001, hit004, hit008, hit011, and jvc001). The tests were conducted on a PC with an Intel

Table 3. A_T and N_{TT} comparison between two stages for the different type of sequences.

Sequence				Reference		BM		MVDR	
GOP Structure	Name	Image Size (pixel)	Coded frames	A_T[a]	N_{TT}[b]	A_T[a]	N_{TT}[b]	A_T[a]	N_{TT}[b]
I only	jvc000	176×144	100	2524.3	2524.3	2524.3	256.3	2443.3	175.3
	san000	352×288	50	2524.1	2524.1	2524.1	256.1	2443.1	175.1
	san001	352×288	100	2524.1	2524.1	2524.1	256.1	2443.1	175.1
Average				2524.2	2524.2	2524.2	256.2	2443.1	175.1
Average Reduction (%)						0	89.85	3.21	93.06
IPPP	hit001	352×288	100	2523.9	2523.9	2523.9	255.9	2442.9	174.9
	hit004	352×288	100	2524.0	2524.0	2524.0	256.0	2443.0	175.0
	hit008	352×288	100	2524.0	2524.0	2524.0	256.0	2443.0	175.0
	hit011	352×288	100	2523.8	2523.8	2523.8	255.8	2442.8	174.8
	jvc001	352×288	150	2523.8	2523.8	2523.8	255.8	2442.8	174.8
Average				2523.9	2523.9	2523.9	255.9	2442.9	174.9
Average Reduction (%)						0	89.86	3.21	93.07

* [a] number of tokens per macroblock

* [b] number of token transactions per macroblock

Table 4. T_{FU} and T_D comparison between two stages for the different type of sequence.

Sequence				Reference		BM		MVDR	
GOP Structure	Name	Image Size (pixel)	Coded frames	T_{FU}[a]	T_D[a]	T_{FU}[a]	T_D[a]	T_{FU}[a]	T_D[a]
I only	jvc000	176×144	100	0.28	2.26	0.18	0.56	0.16	0.49
	san000	352×288	50	0.27	2.11	0.16	0.54	0.17	0.48
	san001	352×288	100	0.27	2.12	0.16	0.55	0.16	0.48
Average				0.27	2.16	0.17	0.55	0.16	0.49
Average Reduction (%)						37.04	74.54	40.74	77.31
IPPP	hit001	352×288	100	0.22	1.23	0.15	0.51	0.16	0.45
	hit004	352×288	100	0.21	1.07	0.16	0.50	0.15	0.44
	hit008	352×288	100	0.22	1.06	0.16	0.49	0.16	0.44
	hit011	352×288	100	0.20	0.97	0.15	0.49	0.15	0.44
	jvc001	352×288	150	0.21	1.02	0.15	0.50	0.15	0.44
Average				0.21	1.07	0.15	0.50	0.15	0.44
Average Reduction (%)						28.57	53.27	28.57	58.88

*[a] milliseconds per macroblock

Core2 Quad CPU Q6600 2.4GHz and 4GB RAM running Microsoft Windows 7.

The complexity of the decoder (C_D) can be measured by the decoding time (T_D). Similarly, the complexity of the FU (C_{FU}) can also be measured using the time consumed for the execution of FU (T_{FU}). Accordingly, A_T, N_{TT}, T_{FU}, and T_D of the aforementioned decoder implementations were measured. Considering the different spatial and temporal resolution of the test sequences, the results measured from the decoder were normalized to the number of macroblocks. The reduced A_T, N_{TT}, T_{FU}, and T_D compared to the results from the reference implementation are also reported to evaluate effectiveness of the proposed approaches.

Table 3 lists the number of A_T and N_{TT} measured from the experimental implementations. A_T and N_{TT} of the BM implementation and the MVDR implementation were compared with those of the reference implementation. From the table, N_{TT} is reduced by 90 percent, whereas A_T is not changed at the BM stage. N_{TT} is further reduced by 3 percent and A_T is reduced by 3 percent at the MVDR stage. Overall, N_{TT} can be reduced significantly at the BM stage more than at the MVDR stage. On the other hand, A_T can be reduced at the MVDR stage; hence, N_{TT} can be reduced further.

Table 4 lists the measured decoding time of the test sequences in the single-threaded environment to evaluate the overall complexity of the decoder. The table shows that T_{FU} is reduced by 37 percent for the I-only sequences and 29 percent for the IPPP sequences at the BM stage. The number of FU calls is believed to decrease due to the merged token transactions, eventually reducing T_{FU}, as described in Eq. (5). In addition, the reduction of N_{TT} can also result in a decrease in T_{TT}, as described in Eq. (6). Therefore, T_D is reduced by 75 percent for the I-only sequences and 53 percent for the IPPP sequences based on the reduction of N_{TT}.

Moreover, the results at the MVDR stage show that T_{FU} for the IPPP sequences is reduced by 4 percent, whereas T_{FU} for the I-only sequences is unchanged. This is presumably because N_{FU} is reduced by 2, as described in Table 2, and the reduced FUs can affect only the P-frame decoding process. On the other hand, it appears that a further reduction of N_{TT} by 3 percent cannot affect T_{FU} by the MVDR approach, whereas a significant reduction of N_{TT} results in a reduction of T_{FU} at the BM stage. T_D was also reduced by 3 percent for the I-only sequences and 6 percent for the IPPP sequences based on the further reduction of N_{TT}. This suggests that a reduction of N_{TT} may result in a more efficient reduction of T_{TT} for the IPPP sequences than for the I-only sequences at the MVDR stage. Furthermore, N_{TT} can be reduced more efficiently at the MVDR stage than at the BM stage based on the accumulated results.

These results show that the proposed approaches are effective in reducing the unnecessary token transactions in the FU network. Although the most important contribution is achieved by the block merge approach, these results confirm that the proposed approaches are important for further improving the efficiency through measurements of A_T and N_{TT}.

In addition, the initial goal of MPEG RMC was to improve the efficiency of implementing decoders, which is not limited in a parallel or non-parallel computing environment. But it is still possible to expect the results of the proposed approaches by exploring parallelism. On the other hand, the main objective of this paper was to demonstrate a fundamental complexity reduction of the RMC-based decoder, which is the main focus by MPEG RMC. Future studies will examine the support for parallelism.

6. Conclusion

This paper presented the design issues and proposed a methodology for efficient token flow design to improve the decoding performance of a decoder based on the MPEG RMC framework. The three proposed approaches were implemented on the MPEG RMC-based MPEG-4 SP decoder. Comparative analysis of the experimental results using a reference implementation confirmed that those improvements in the FU network design can be achieved using the proposed methodology.

References

[1] ISO/IEC 23001-4:2011, Information technology: MPEG systems technologies—Part 4:Codec configuration representation, 2011. Article (CrossRef Link)

[2] H. Amer, A.A.H. Ab-Rahman, I. Amer, C. Lucarz, and M. Mattavelli, "Methodology and technique to improve throughout of FPGA-based CAL dataflow programs: case study of the RVC MPEG-4 SP intra decoder," Signal Processing Systems (SiPS), pp.186-191, 2011. Article (CrossRef Link)

[3] M. Hentati, Y. Aoudni, JF Nezan, M. Abid, and O. Deforges, "FPGA dynamic reconfiguration using the RVC technology: inverse quantization case study," Conference on Design and Achitectures for Signal and Image Processing (DASIP), pp.1-7, 2011. Article (CrossRef Link)

[4] H. Yviquel, E. Casseau, M. Wipliez, M. Raulet, "Efficient multicore scheduling of dataflow process networks," Signal Processing Systems (SiPS), pp.198-203, 2011.Article (CrossRef Link)

[5] S. Casale-Brunet, A. Elguindy, E. Bezati, R. Thavot, G. Roquier, M. Mattavelli, J. Janneck, "Methods to explore design space for MPEG RMC codec specifications," Signal Processing: Image Communication, pp. 1278-1294, 2013. Article (CrossRef Link)

[6] R. Gu, J. W. Janneck, S. S. Bhattacharyya, M. Raulet, M. Wipliez, W. Plishker, "Exploring the Concurrency of an MPEG RVC Decoder Based on Dataflow Program Analysis," IEEE Transactions on Circuilts and Systems for Video Technology, pp.1646-1657, 2009. Article (CrossRef Link)

[7] M. Wipliez, M. Raulet, "Classification of Dataflow Actors with Satisfiability and Abstract Interpretation," International Journal of Embedded and Real-Time Communication Systems, 2012. Article (CrossRef Link)

[8] J. Ersfolk, G. Roquier, W. Lund, M. Mattavelli, J. Lilius, "Static and Quasi-static Composition of Stream Processing Applications from Dynamic Dataflow Programs," International Conference Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2013. Article (CrossRef Link)

[9] J. Gorin, H. Yviquel, F. Préteux, "Just-in-Time Adaptive Deocder Engine: A Universal Video Decoder based on MPEG RVC," ACM Multimedia, Scottsdale, USA, 2011. Article (CrossRef Link)

[10] ISO/IEC 23002-4:2014. Information technology: MPEG video technologies Part 4: Video Tool Libray, 2014. Article (CrossRef Link)

[11] C. Lucarz, G. Roquier, M. Mattavelli, "High Level Design Space Exploration of RVC Codec Specifications for Multi-core Heterogeneous Platforms," Conference on Design and Achitectures for Signal and Image Processing (DASIP), Edinburgh, 2010. Article (CrossRef Link)

[12] S. C. Brunet, E. Bezati, C. Alberti, M. Mattavelli, E. Amaldi, J. W. Janneck, "Partitioning and Optimization of High Level Stream Applications for Multi Clock Domain Architectures," IEEE Workshop on Signal Processing Systems(SiPS), Taipei, 2013. Article (CrossRef Link)

[13] J. Boutellier, C. Lucarz, S. Lafond, V. M. Gomez, M. Mattavelli, "Quasi-Static Scheduling of CAL Actor Networks for Reconfigurable Video coding," Jounral of Signal Processing Systems,pp. 191-202, 2011. Article (CrossRef Link)

[14] C. Tulvan, and S. Lee, "Installation and usage of software frameworks for GTL," ISO/IEC JTC1/SC29/WG11 N12767, Geneva, Switzerland, May, 2012. Article (CrossRef Link)

[15] H. Kim, S. Kim, M. Park, H. Shin, B. Choi, C. Yie, and E. S. Jang, "MPEG-4 ASP Decoder Description and a Case Study of the Decoder Design Process in the RVC framework," ISO/IEC JTC1/SC29/WG11 M16450, Maui, USA, April 2009. Article (CrossRef Link)

[16] S. Lee, J. Lee, S. Lee, and E. S. Jang, "Proposed to use the virtual FU concept in RVC/RGC," ISO/IEC JTC1/SC29/WG11 M22735, Geneva, Switzerland, Nov. 2011. Article (CrossRef Link)

[17] C. Tulvan, S. Lee, M. Raulet, E. S. Jang, and M. Preda, "Text of ISO/IEC 23002-4:2010/PDAM 3 Graphics Tool Library for Reconfigurable Multimedia Coding Framework," ISO/IEC JTC1/SC29/WG11 N12589, San Jose, USA, Feb. 2012. Article (CrossRef Link)

[18] ISO/IEC, Video conformance Bitstreams, Publicly Available Standards: ISO/IEC 14496-4:2004 Conformance testing, 2012. Article (CrossRef Link)

Li Cui received her B.S. degree from Yanbian University, Yanji, China in 2000 and the M.S. degree from Honam University, Korea in 2007. She is currently working toward a Ph.D. at Hanyang University, Seoul, Korea. She is also a lecturer in the Department of Computer Science and Technology, College of Science and Engineering, Yanbian University, Yanji, China. Her current research interests include MPEG Reconfigurable Media Coding, MPEG Reconfigurable Graphics Coding, and computer graphics.

Sowon Kim received his B.S. degree and M.S. degree from Hanyang University, Seoul, Korea in 2008 and 2012. He is currently working toward a Ph.D. at Hanyang University, Korea. His current research interests include multimedia framework design and dataflow optimization for MPEG Reconfigurable Media Coding (RMC).

Hyungyu Kim received his Ph. D. degree from Hanyang University, Seoul, Republic of Korea, in 2014. He is currently a post-doctorate researcher in the Division of Computer Science and Engineering of Hanyang University. He is interested in the MPEG Reconfigurable Media Coding framework: his main research topics are the bitstream syntax description and the bitstream parser generation methodology. He has received a Certificates of Appreciation from International Standards Organization and International Electro-technical Commission in 2011 for his contribution to the development of ISO/IEC 23001-4 international standard.

Euee S. Jang received his B.S. degree from Jeonbuk National University, Korea and a Ph. D. from SUNY at Buffalo, NY, USA. He is a Professor in the Division of Computer Science & Engineering, Hanyang University, Seoul, Korea. His research interests include image/video coding, reconfigurable video coding, and computer graphics objects. He has authored more than 150 MPEG contribution papers, more than 30 journal or conference papers, 35 pending or accepted patents, and two book chapters. Dr. Jang has received three ISO/IEC Certificates of Appreciation for contributions to MPEG-4 development. He received the Presidential Award from the Korean Government for his contribution to MPEG standardization.