AN RVC DATAFLOW DESCRIPTION OF THE AVC CONSTRAINED BASELINE PROFILE DECODER

J. Gorin[2] - *M. Raulet*[1] - *Y-L. Cheng*[4] - *H-Y. Lin*[4] - *N. Siret*[1] - *K. Sugimoto*[3] - *G.G. Lee*[4]

[1]IETR, INSA Rennes F-35043, France
[2]ARTEMIS, Institut Telecom SudParis, UMR 8145, Evry,France
[3]Mitsubishi Electric Corporation, Japan
[4]National Cheng Kung Univ, Taiwan

ABSTRACT

Video codec applications become more and more complex to design. To ease the description of such applications, MPEG creates a Framework called Reconfigurable Video Coding (RVC). All existing codecs in MPEG have a similar structure, they are based on a hybrid decoding structure and some of their part can be reused on other design. In RVC, the dataflow is expressed using a network of components also called Functional Units (FUs) interconnected by FIFOs. An FU, programmed in CAL Language, includes the processing and the internal states. This paper puts the focus on a parallel dataflow description of the most complex MPEG RVC decoder available called MPEG4-AVC Constrained Baseline Profile (CBP) decoder.

I. INTRODUCTION

The **Moving Picture Experts Group** (MPEG) produced several video coding standards such as MPEG-1, MPEG-2 and MPEG-4 (*Simple Profile*, *Advanced Video Coding* and its extension *Scalable Video Coding*). The past monolithic (usually in C/C++) specification of standards lacks of genericness, flexibility and reusability. To overcome this limitation, MPEG is standardizing a novel Framework named reconfigurable video coding (RVC)[1] capable of building various existing coding standards and even future standards. It defines several elementary functional units (FU) that implement coding tools. By carefully connecting proper FUs, a decoder can be built.

So far, the only way to experiment video decoding with the MPEG RVC approach was to use the MPEG-4 Simple Profile description provided by the MPEG RVC experts. Since RVC Framework has grown and its associated tool matured, a more relevant application with realistic sizes and complexity that can prove and promote the benefit of the RVC approach is mandatory.

This position paper describes the practical approach to design the MPEG-4 *Advanced Video Coding* (*AVC*), which is the current most complex MPEG RVC video coding standard. After reviewing in section II the RVC Framework and some basic properties of the RVC CAL actor language, we present in section III the MPEG-4 AVC decoder design and its translation into a dataflow model. This paper finally closes with the result and the issue opened by this decoder modeling.

II. RVC FRAMEWORK

The MPEG RVC Framework is currently under development by MPEG committee as part of MPEG-B and MPEG-C standards [1]. It aims at providing an abstract model of existing or completely new MPEG standards at system-level.

An abstract decoder, shown in figure 1, is built as a block diagram — expressed with the Functional unit Network Language (FNL) [2]— in which blocks define processing entities called Functional Units (FUs) [3] and connections represent data flows between FUs. The MPEG-B part 5 Bitstream Syntax Description Language (BSDL) [4] describes the syntax of the bitstream of an RVC decoder.

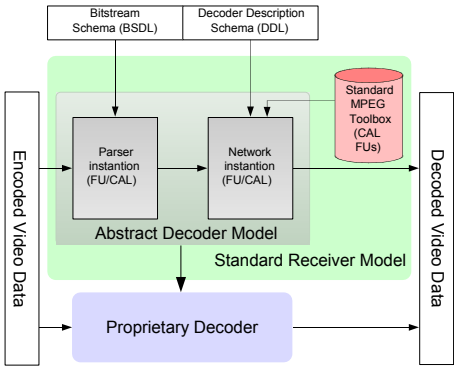

Fig. 1. Graphical representation of the conceptual process of deriving a RVC abstract decoder model.

RVC provides both a normative standard library of FUs and a set of decoder descriptions expressed as networks of FUs. Such a representation is modular and helps the reconfiguration of a decoder by modifying the topology of the network. RVC mainly focuses on reusability by allowing

different decoder descriptions to instantiate common FUs across standards.

II-A. RVC Decoder Description

An RVC decoder is described as a block diagram with FNL [2], an XML dialect that describes the structural network of interconnected actors from the Standard MPEG Toolbox. A network can be hierarchical — this means that a network may be a part of a more general network — and FNL can be used to pass parameters to actors.

The only case study performed by MPEG RVC experts so far is the RVC CAL specification of MPEG-4 Simple Profile decoder [5], [6]. Figure 2 shows the network representation of the macroblock-based MPEG-4 Simple Profile decoder description. All of these blocks are atomics actors except for the parser which is an FNL hierarchical network of actors.

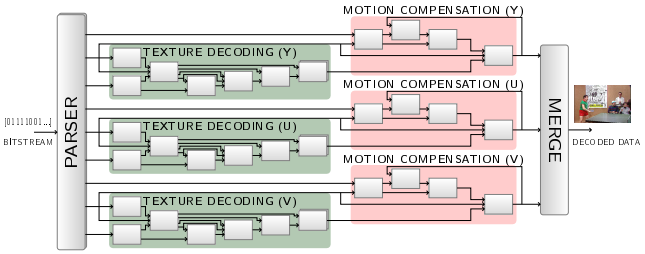

Fig. 2. MPEG-4 Simple Profile decoder description.

II-B. FU Specification

The language for the reference software is RVC CAL ; a subset of CAL Actor Language[7]. RVC CAL specifies the model of computation and the normative I/O behaviors of the FUs composing the RVC standard library. A CAL actor (Fig. 3) is a computational entity with interfaces (input and output ports), internal states and parameters. An actor is strongly encapsulated ; it can neither access nor modify the state of any other actor and may only interact with others by sending data (called tokens) along channels. During an *action* execution (also referred as *firing* an action), actor can consume input tokens, produce output tokens and change its internal state. *Action guards, action scheduling* with *Finite State Machine* (*FSM*) or *action priorities* are control structure that constrained the selection of the action to *fire*.

An action is fireable if it respects the following conditions : (1) there are the necessary tokens on input ports ; (2) guard clauses, if present, evaluate to true ; (3) the current state enables the action to fire according to the *Finite State Machine* (*FSM*) ; no higher-priority action respects (1), (2) and (3).

Functional Unit (*FU*) corresponds to the RVC CAL actors contained in the Standard MPEG Toolbox (Fig. 1). Two kinds of *FU* are distinguished : algorithmic video coding tool (IDCT, inverse quant) and a data management tool (Parser, multiplexer, demultiplexer). *FUs* containing algorithmic video coding tool are usually reusable between decoder. In opposite, data management tools have the role to adapt these FUs to a specific structure and are usually specific to a decoder.

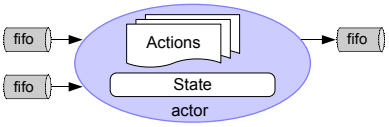

Fig. 3. Illustration of a CAL actor.

II-C. RVC tools

Once the RVC CAL model is defined, the user can simulate the CAL dataflow model by using the OpenDF simulator [1][8]. Two tools that generates low-level code from an abstract CAL model are also available. CAL2HDL[6] converts a CAL model into an HDL representation, allowing to test dataflow execution in a hardware platform. CAL2C converts CAL representation to C code[5]. The C code can then be compiled on any processor, from embedded DSPs and ARMs to general-purpose processors. For an automatic generation of the parser, a BSDL to CAL translator is also under development as part of the OpenDF effort [4].

MPEG4 SP decoder	Speed kMB/S
CAL simulator	0.015
Cal2C	3
Cal2HDL	290

Table I. MPEG4 Simple Profile decoder speed.

Table I shows the performance of the synthesized MPEG-4 SP decoder *Intel Core 2 Duo* @ 2.5Ghz. The C-software synthesized decoder tested on a QCIF video format is real time (30 frames/s) contrary to the simulated CAL dataflow program, which decode 0.15 frames/s for the same format. The automatically synthesized hardware description generated by Cal2HDL performs a real-time decoding on a Full HD video format (1080p30) on *Xilinx Virtex2 FPGA* @ 100 Mhz.

III. MPEG-4 AVC DECODER

MPEG-4 Advanced Video Coding (AVC), or also know as H.264[9], is a state-of-the-art video compression standard. Compared to previous coding standards, it is able to deliver higher video quality for a given compression ratio, and 30% better compression ratio compare to MPEG-4 SP for the same video quality. Because of its complexity, many

1. Open dataflow sourceforge project : http://opendf.sourceforge.net/.

applications including Blu-ray, iPod video, HDTV broadcasts, and various computer applications use variations of MPEG-4 AVC codec (also called *profiles*). A popular uses of MPEG-4 AVC is the encoding of high definition video contents. Due to high resolutions processing required, HD video is the application that requires the highest performance for decoding. Common formats used for HD include 720p (1280x720) and 1080p (1920x1080) resolutions, with frame rates between 24 and 60 frames per second.

III-A. Overview

All MPEG codec are based on the same structure, the hybrid decoding structure including intra-prediction and motion compensation. Therefore, MPEG4SP and MPEG-4 AVC are hybrid decoders. Figure 4 shows the main functional blocks composing MPEG-4 AVC.

The MPEG-4 AVC decoder divides each frame of a video sequence into *macroblocks*, which are blocks of 16x16 pixels. These macroblocks are decoded using *intra-prediction* or *inter-prediction* in *raster scan* order (left to right, then top to bottom). Intra prediction uses neighboring macroblocks information to predict pixel value into the current macroblock. The inter prediction uses previous decoded picture taken into *picture buffering* to generate the current macroblock. The deblocking filter reduces the distortion between neighbouring macroblocks.

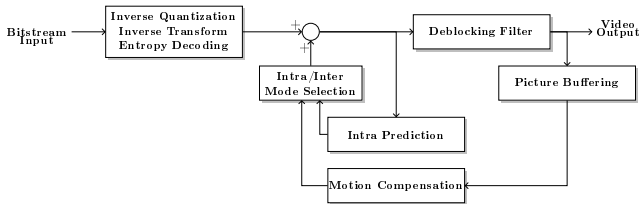

Fig. 4. The main functional blocks of MPEG-4 AVC/H.264.

III-B. Implementation

The decoder introduced in this section corresponds to the *Constrained Baseline Profile* (*CBP*). This profile is primarily fitted to lowest-cost applications and corresponds to a subset of features that are in common between the *Baseline*, *Main*, and *High Profiles*.

The description of this decoder expresses the maximum of parallelism and mimics the MPEG4 SP. This description is composed of different hierarchical level. Fig. 5 shows a view of the highest hierarchy of the MPEG-4 AVC decoder — note that for readability, one input represents a group of input for similar information on each actor. The main functional block includes a parser, one luma and two chroma decoders.

The parser analyses the syntax of the bitstream with a given formal grammar. This grammar, written by hand, will later be given to the parser by a BSDL[4] description. As the execution of a parser strongly depends on the context of the bitstream, the parser incorporates a Finite State Machine so that it can sequentially extract the information from bitstream. This information passes through an entropy decoder and is then encapsulated in several kinds of tokens (residue coefficients, motion vectors…). These tokens are finally sent to the selected input port of the luma/chroma decoding actor.

Because decoding a luma/chroma component does not need to share information with the other luma/chroma component, we choose to encapsulate each luma/chroma decoding in a single actor. This means that each decoding actor can run independently and at the same time in a separate thread. The entire decoding component actor has the same structure.

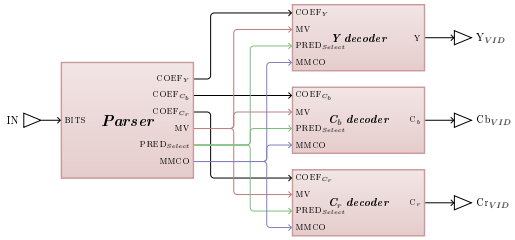

Fig. 5. Top view of MPEG-4 Advanced Video Coding decoder description.

Luma/chroma decoding actors (Fig. 6) decode a picture and store the decoded picture for later use in inter-prediction process. Each component owns the memory needed to store pictures, encapsulates into the *Decoded Picture Buffer* (*DPB*) actor. *DPB* actor also contains the *Deblocking Filter* and is a buffering solution to regulate and reorganize the resulting video flow according to the *Memory Management Control Operations* (*MMCO*) input.

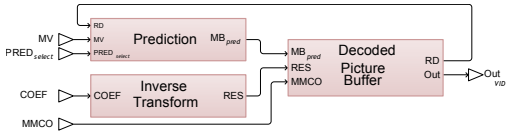

Fig. 6. Structure of decoding actors.

The *Decoded Picture Buffer* creates each frame by adding prediction data, provided by the actor *prediction*, and residual data, provided by the actor *Inverse Transform*. The *Prediction* actor (Fig. 7) encompasses *inter/intra prediction* modes and a multiplexer that sends prediction results to the output port. The $PRED_{select}$ input port has the role to stoke the right actors contingent on a prediction mode. The target of this structure is to offer a quasi-static working of the global actor and, by adding or removing prediction modes, to easily switch between configurations of the decoder. For

instance, adding *B inter-prediction mode* into this structure switches the decoder into the *main profile* configuration[10].

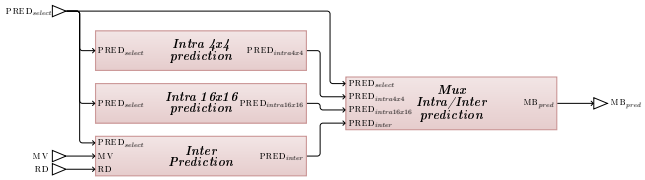

Fig. 7. Structure of prediction actor.

III-C. Results

A comparison of the CAL description (Tab. II) shows that the MPEG-4 AVC CAL decoder is twice more complex than the MPEG-4 Simple Profile CAL description. Some parts of the model have already been redesign in order to improve pipelining and parallelism between actors. A simulation of the MPEG-4 AVC CAL model on a *Intel Core 2 Duo @* 2.5Ghz is more than 2.5 slower than the RVC MPEG-4 Simple Profile description.

Comparing to the MPEG-4 Simple Profile CAL model, the MPEG-4 AVC decoder has been modeled to use more CAL possibility (for instance processing of several tokens in one firing) while staying fully RVC conformant. Thanks to this increasing complexity, MPEG-4 AVC CAL model is the most reliable way to test the accordance and the efficiency of the current RVC tools. The next challenge that RVC experts have to deal with is to get the CAL synthesizing tools compliant with this description.

	Actors	Parser size SLOC	Decoder size SLOC	Performance MB/S
MPEG-4 SP	27	960	2900	15
MPEG-4 AVC	45	1980	3900	6

Table II. OpenDF benchmark of MPEG-4 Simple Profile and MPEG-4 Advanced Video Coding RVC description.

IV. CONCLUSION AND PERSPECTIVE

MPEG RVC is presented as the new solution to reduce the problem induces by new decoder complexities. So far, the only RVC description available was a MPEG-4 Simple Profile description. This paper gives a new way to test the efficiency and the reliability of the CAL tools, especially synthesizing tools, by modeling the most complex decoder provided by MPEG RVC.

Moreover, the MPEG-4 *Scalable Video Coding* decoder is an extension of the MPEG- 4 AVC decoder. It reuses most of its components [11] and can decode video content with several layers i.e. one AVC compatible base-layer and several optional enhancement layers. In addition to improve the AVC decoder, RVC plans include the implementation of the SVC decoder as an overlay of AVC decoder. Our goal is to illustrate the practicality of the RVC approach by reusing functional units from the Standard MPEG Toolbox to design new MPEG standards.

V. REFERENCES

[1] S. Bhattacharyya, J. Eker, J.W. Janneck, C. Lucarz, M. Mattavelli, and M. Raulet, "Overview of the MPEG Reconfigurable Video Coding Framework," *Journal of Signal Processing Systems*, 2009.

[2] ISO/IEC FDIS 23001-4 : 2009, "Information technology - MPEG systems technologies - Part 4 : Codec Configuration Representation," 2008.

[3] ISO/IEC FDIS 23002-4 : 2009, "Information technology - MPEG video technologies - Part 4 : Video Tool Library," 2008.

[4] M. Raulet, J. Piat, C. Lucarz, and M. Mattavelli, "Validation of bitstream syntax and synthesis of parsers in the MPEG Reconfigurable Video Coding framework," in *Signal Processing Systems (SiPS)*, 2008, pp. 293–298.

[5] G. Roquier, M. Wipliez, M. Raulet, J.W. Janneck, I.D. Miller, and D.B. Parlour, "Automatic software synthesis of dataflow program : An MPEG-4 simple profile decoder case study," in *Signal Processing Systems (SiPS). IEEE Workshop on*, 2008, pp. 281–286.

[6] J.W. Janneck, I.D. Miller, D.B. Parlour, G. Roquier, M. Wipliez, and M. Raulet, "Synthesizing hardware from dataflow programs : an MPEG-4 Simple Profile decoder case study," in *Signal Processing Systems (SiPS)*, 2008.

[7] J. Eker and J. Janneck, "CAL Language Report," Tech. Rep. ERL Technical Memo UCB/ERL M03/48, University of California at Berkeley, 2003.

[8] S. Bhattacharyya, G. Brebner, J. Eker, J. Janneck, M. Mattavelli, C. von Platen, and M. Raulet, "OpenDF - A Dataflow Toolset for Reconfigurable Hardware and Multicore Systems," in *First Swedish Workshop on Multi-Core Computing (MCC)*, 2008, pp. 43–49.

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

[10] T. Wiegand, G.J. Sullivan, and A. Luthra, "Draft ITU-T recommendation and final draft international standard of joint video specification," Tech. Rep. ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, 2003.

[11] M. Pelcat, M. Blestel, and M. Raulet, "From AVC Decoder to SVC : Minor impact on dataflow graph description," in *Proceedings of Picture Coding Symposium (PCS)*, 2007.