

# 國立交通大學

資訊工程所

碩士論文

視訊品質控制之軟硬體協同設計與純粹硬體方法



**Video Rate Control for Co-Design and Pure Hardware Approach**

研究生：陳京何

指導教授：蔡淳仁 教授

中華民國九十三年六月

視訊品質控制之軟硬體協同設計與純粹硬體方法

**Video Rate Control for Co-Design and Pure Hardware Approach**

研 究 生：陳京何

Student : Ching-Ho Chen

指導教授：蔡淳仁

Advisor : Chun-Jen Tsai

國 立 交 通 大 學

資 訊 工 程 系

碩 士 論 文



Submitted to Department of Computer Science and Information Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of Master

in

Computer Science and Information Engineering

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月



# 視訊品質控制之軟硬體協同設計與純粹硬體方法

學生：陳京何

指導教授：蔡淳仁

國立交通大學資訊工程所碩士班

## 摘要

大部分軟體為主的視訊編碼器會在動作預測之後，對頻寬及失真作分析來確定量化器的大小，由動作預測單元得到的絕對誤差總和被使用來估計頻寬控制模式中的編碼複雜度。對於 SoC 平台上的視訊編碼器來說，如果要達到較佳的頻寬控制效能，由於需要計算複雜的數學模式，比較適合由微處理器來計算，而區塊編碼迴圈則是由 VLSI 加速器來執行。為了降低微處理器與加速電路溝通上的負荷，因此本論文設計出一套能在編碼迴圈外獨立執行的頻寬控制模式的方法。這個方法除了控制視訊編碼的資料流量之外，並可作場景變換偵測。另外，為了更進一步降低溝通上的負擔及簡化系統匯流排的設計，在此亦提出了查表方式的頻寬控制模式，用查表的方式取代複雜的碼率失真數學模型的計算，此演算法十分適合直接做在加速電路之中。總結而言，在此論文中，我們提出了一套適用於 SoC 平台上的視訊編碼器的迴圈外頻寬控制演算法，以及一個查表方式的頻寬控制演算法。由實驗結果可知，這些方法的效果極佳，很適合用在實際的系統中。

# **Video Rate Control for Co-Design and Pure Hardware Approach**

Student : Ching-Ho Chen

Advisor : Chun-Jen Tsai

Department of Computer Science and Information Engineering  
Nation Chiao Tung University

## **ABSTRACT**

Most software-based video encoders perform rate-distortion model analysis to determine the quantizer step size after motion estimation (ME). Typically, Sum of Absolute Differences (SAD) from the motion estimator is used as the complexity estimates for rate-control model. However, for video encoders on SoC platforms, the calculation of rate-control model is typically done on the microcontroller (MCU) core while the macroblock encoding loop (ME, transform, quantization, and entropy coding) is executed by ASIC accelerators. In order to reduce the communication overhead between the MCU and the ASIC accelerator, this thesis proposes a rate control algorithm that can be executed outside the encoding loop solely by the MCU is very useful. In addition, this thesis also proposes a lookup table approach that is suitable for pure hardware implementation. In this approach, the sophisticated R-D model is replaced with a lookup table and low complexity VLSI architecture can be used for rate-control for ASIC accelerators. Experimental results show that these algorithms are very promising for video codec hardware/software co-design and pure hardware implementation in practical systems.

## Acknowledgement

I am glad to finalize this thesis in favor of many people. First, my advisor, Chun-Jen Tsai, gives me much motivation and ideas, and encourages me to think in different viewpoints. Then, I also thanks for helps and comments from my seniors, juniors, and classmates. Of course, I want to thanks my family for giving me strong economic and moral support, especially for my dear mother. Finally, it is my pleasure to finish this thesis in my laboratory, MMES Lab of CSIE in NCTU.



# Content

<b>1. Introduction.....</b>	<b>1</b>
1.1. Introduction to Rate Control .....	1
1.1.1. Buffer Issue .....	1
1.1.2. Quality Variation .....	1
1.1.3. Coding and Transmission Delay .....	2
1.1.4. Granularity of Rate Control .....	2
1.1.5. Limitation of SoC Implementations.....	2
1.2. Video Encoder Architecture .....	3
1.2.1. Introduction to Video Coding modules.....	4
1.2.2. Rate Control Illustration .....	5
1.3. Research Motivation and Background.....	6
1.4. Problem Analysis .....	8
1.4.1. Communication Overhead of Frame-Level Rate Control.....	8
1.4.2. Communication Overhead of Macroblock-Level Rate Control.....	8
<b>2. Previous Work.....</b>	<b>10</b>
2.1. Number of Encoding Passes for RC.....	10
2.2. Encoding Delay and RC.....	10
2.3. Rate-Distortion Optimization and RC .....	10
2.4. Rate-Distortion Model for RC .....	11
2.5. Frame Dependency .....	11
2.6. Variable Frame Rate RC .....	11
2.7. RC for SoC Platforms .....	11
<b>3. Proposed RC Using Estimated Complexity .....</b>	<b>12</b>
3.1. Rate control using a first order RD model.....	12
3.1.1. Frame Level Rate Control.....	12
3.1.2. Macroblock Level Rate Control.....	13
3.2. Out-of-Loop Video Rate Control for Software/Hardware co-Design.....	17
3.2.1. Frame Complexity Estimation and Frame-Level Rate Control.....	17
3.2.2. Adaptive RD Model.....	18
3.2.3. Other Adaptive RD Model Methods.....	19
3.2.4. Macroblock Level Complexity Estimation and Rate Control.....	20
3.3. Encoding of I-frames .....	23

3.3.1.	Scene Change Detection .....	23
3.3.2.	Complexity Estimation for I frame .....	24
<b>4.</b>	<b>Pure Hardware Rate Control.....</b>	<b>24</b>
4.1.	Consideration in Hardware Rate Control.....	25
4.2.	Hardware Rate Control Approach.....	25
4.2.1.	Process Flow of the H/W Rate Control Method .....	26
4.2.2.	Modeling Table Indexing .....	27
4.2.3.	Update Modeling Table.....	28
4.3.	Operations of the H/W Rate Control .....	28
4.4.	Hardware architecture of rate control .....	30
<b>5.</b>	<b>Experiments and Analysis .....</b>	<b>32</b>
5.1.	Testing Environment and Video Encoder Configuration.....	32
5.1.1.	Environment.....	32
5.1.2.	Encoder Configuration.....	32
5.2.	Frame Level Rate Control Comparison .....	33
5.2.1.	PSNR.....	34
5.2.2.	Frame Bits Variation .....	36
5.2.3.	Buffer Status .....	38
5.3.	Macroblock Level Rate Control Comparison.....	40
5.3.1.	PSNR.....	40
5.3.2.	Frame Bits Variation .....	43
5.3.3.	Buffer Status .....	45
5.4.	Comparisons Among Sequences.....	47
5.4.1.	PSNR and Output Bitrate.....	47
5.4.2.	Mean and Standard Deviation for PSNR Variation .....	48
5.4.3.	Mean and Standard Deviation for Bits Variation .....	50
<b>6.</b>	<b>Conclusion and Future Work .....</b>	<b>52</b>
6.1.	Discussions .....	52
6.2.	Future Work .....	52
6.2.1.	Design Better Measure for Frame-Level or MB-Level Complexity .....	52
6.2.2.	Use More Sophisticated R-D Model.....	53
<b>7.</b>	<b>Reference .....</b>	<b>53</b>



## Figure List

Fig 1. Video Encoder Architecture .....	3
Fig 2. Rate Control Flow .....	5
Fig 3. Data Flow Between MCU and ASIC Accelerator .....	7
Fig 4. Dual-Core SoC Platform Architecture .....	7
Fig 5. MB Level Rate Control R-D Modeling .....	14
Fig 6. Communication Between MCU and Accelerator .....	16
Fig 7. R-D Modeling for One Pack .....	21
Fig 8. Modeling Table in Hardware Rate Control .....	27
Fig 9. System Architecture of H/W Rate Control .....	31
Fig 10. Block Diagram of H/W Rate Control .....	31
Fig 11. PSNR plots for COASTGUARD sequence .....	34
Fig 12. PSNR plots for FOREMAN sequence .....	35
Fig 13. PSNR plots for STEFAN sequence .....	36
Fig 14. Frame bits plots for COASTGUARD sequence .....	36
Fig 15. Frame bits plots for FOREMAN sequence .....	37
Fig 16. Frame bits plots for STEFAN sequence .....	38
Fig 17. Buffer status plots for COASTGUARD sequence .....	38
Fig 18. Buffer status plots for FOREMAN sequence .....	39
Fig 19. Buffer status plots for STEFAN sequence .....	40
Fig 20. PSNR plots for COASTGUARD sequence .....	41
Fig 21. PSNR plots for FOREMAN sequence .....	41
Fig 22. PSNR plots for STEFAN sequence .....	42
Fig 23. Frame bits plots for COASTGUARD sequence .....	43
Fig 24. Frame bits plots for FOREMAN sequence .....	44
Fig 25. Frame bits plots for STEFAN sequence .....	45
Fig 26. Buffer status plots for COASTGUARD sequence .....	45
Fig 27. Buffer status plots for FOREMAN sequence .....	46
Fig 28. Buffer status plots for STEFAN sequence .....	47

## Table List

Tab 1.	Encoder Configuration .....	33
Tab 2.	PSNR & Output Bitrate for Frame Level Rate Control .....	47
Tab 3.	PSNR & Output Bitrate for Macroblock Level Rate Control.....	48
Tab 4.	Mean & Standard Deviation for PSNR Variation for Frame Level Rate Control .....	49
Tab 5.	Mean & Standard Deviation for PSNR Variation for Macroblock Level Rate Control .....	49
Tab 6.	Mean & Standard Deviation for Frame Bits Variation for Frame Level Rate Control .....	50
Tab 7.	Mean & Standard Deviation for Frame Bits Variation for Macroblock Level Rate Control .....	51



# 1. Introduction

## 1.1. Introduction to Rate Control

Recently, digital video applications based on international standards (for example, H.263, MPEG-4 SP[1], and H.264 [2]) have become popular for embedded devices such as mobile phones, digital cameras, and PDAs. SoC platforms are commonly used to implement the core components of these devices. The most flexible and cost efficient way to realize a video codec on these platforms are to adopt hardware/software co-design.

When applying digital video in different service environments, there are different transport/storage characteristics which should be taken into account in the encoding process. There are two types of video bitstreams, the first one is constant bitrate (CBR) bitstreams and the second one is variable bitrate (VBR) bitstreams. CBR traffics are preferred in most communication systems. However, compressed digital video is, by nature, composed of VBR data. This is due to the fact that complexity of video content varies across time, and it must be regularized by a rate control mechanism in order to achieve CBR. Unfortunately, enforcing CBR constraint reduces the quality of compressed video drastically, if not done properly. The encoder module that controls the data bandwidth of the video bitstream is called the rate-control (RC) module.

While developing a suitable rate control mechanism, there are many important factors that have to be taken into account, such as, buffer usage and timing constraints, etc. These factors affect RC greatly and are discussed in the following subsections.

### 1.1.1. Buffer Issue

From rate-control point of view, the encoder must make sure that buffer overflow/underflow does not appear by controlling encoder buffer fullness during encoding process. As long as the decoder buffer is larger than or equal to the encoder buffer, there will be no buffer overflow/underflow problem. However, when unreliable transports between the encoders and the decoders are involved, the situation becomes much more complicated and is beyond the scope of this thesis.

### 1.1.2. Quality Variation

For block-based motion compensated video codec, the coded visual quality varies with the degree of motion. When the degree of motion in two consecutive frames is large and irregular, the performance of block-based prediction becomes worse. The error residuals in this case will be high and more bits are required to

encode the video data.

In video encoding, the RC module controls the output bitrate by changing the quantization parameter, QP. Obviously, if constant quantization parameter (QP) is used for the whole sequence, the output bitrate will be VBR. If the RC module strictly enforces the CBR constraint, the video quality at high motion segments will be sacrificed. The fluctuation of visual quality in this case can be very large and are not tolerable by most viewers.

The tradeoff between visual quality and output bitrate is the key to coding performance. In order to achieve good overall visual quality under a rate constraint, a good RC mechanism is needed.

### **1.1.3. Coding and Transmission Delay**

For real-time applications such as videophone, videoconferencing, interactive system, etc., long delay for each compressed video frame is not permissible. In order to reduce the delay, both encoding delay and transmission delay must be reduced. Therefore, for such applications, the rate control module cannot apply sophisticated multiple-pass analysis or complicated prediction pattern such as B-frames.

### **1.1.4. Granularity of Rate Control**

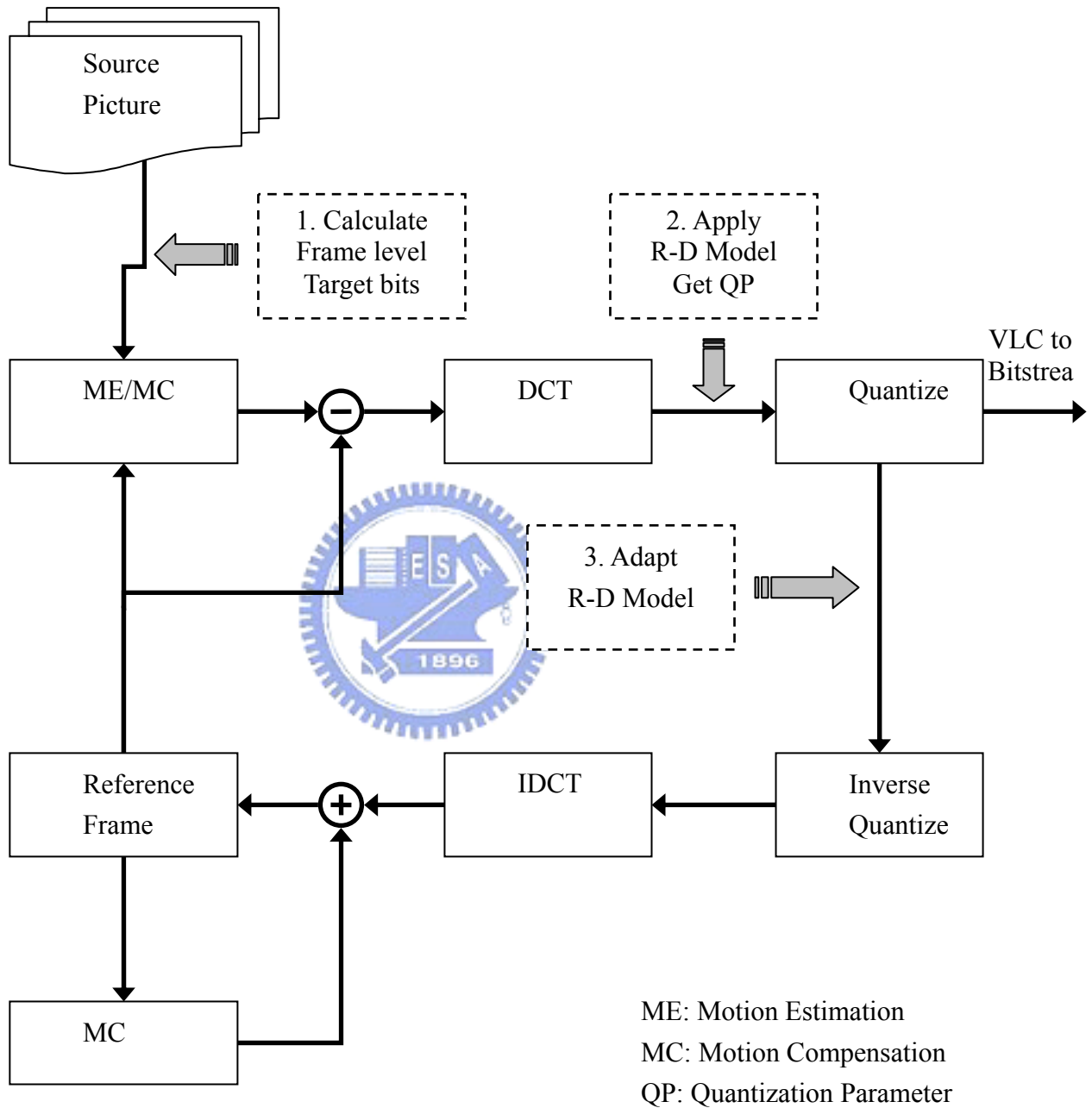
The granularity of rate control can be set at frame level or macroblock (MB) level. A frame-level rate control computes a single quantization parameter for each frame and applies buffer control to shape the overall output bitrate to match the target bitrate. A MB-level rate control method does not only set the quantization parameter for each frame but also adaptively changes the quantization parameter for each macroblock in a frame. Theoretically, the performance of MB-level rate controls should be better than frame-level rate controls. However, for MB-level rate control, it is not easy to fulfill rate constraint while maintaining constant quality. In addition, due to syntax limitation of MPEG-4 standard [1], the quantization parameter difference of two consecutive macroblocks must be less than or equal to 2. These difficulties make MB-level rate control more challenging.

### **1.1.5. Limitation of SoC Implementations**

For a software-based rate-control, good coding efficiency can be achieved by employing sophisticated rate-distortion model for good quantization parameter selection. This is not suitable for a pure hardware implementation because it requires a powerful Arithmetic Logic Unit (ALU). On the other hand, a hardware/software co-design with rate-control algorithm embedded in the encoding loop would cause too much communication overhead between the MCU and the VLSI accelerator core. This overhead typically includes interrupts handling and excessive data transfer

overhead.

## 1.2. Video Encoder Architecture



**Fig 1. Video Encoder Architecture**

Fig 1 is the flow diagram for video encoder to generate the output bitstream. The blocks with solid lines are the essential processing elements and blocks with dashed lines are rate control points.

### 1.2.1. Introduction to Video Coding modules

In this section, we will give a brief introduction to the essential processing modules of a video encoder. The input video data to an encoder is usually in  $YC_B C_R$  format. In this representation, the luminance information ( $Y$ -channel, a.k.a. luma) is separated from the chrominance information ( $C_B$ - and  $C_R$ - channels, a.k.a. chroma).

#### 1.2.1.1. Motion Estimation/Motion Compensation

Each video frame is encoded using either inter-coding or intra-coding modes. Inter-coding modes (P-frame or B-frame) employs motion prediction to reduce the temporally correlated data while intra-coding mode (I-frame) operates directly on the pixel data of the current frame. For the first source frame, the only choice is intra-coding because there are no previous frames to support inter-coding. For subsequent frames, all three coding types (I, P, B frame) can be used. These coding modes also exist at the macroblock-level. However, some restrictions apply. For I-frames, only I-MBs are allowed. For P-frames, both I- and P-MBs are permissible. Finally, for B-frames, all coding modes are possible. The key to the performance of an video encoder is the mode decision algorithm (which can be part of the RC module) and the motion estimation algorithm.

#### 1.2.1.2. DCT

DCT is used to transform pixel data (I-MBs) or error residual data (P- and B-MBs) into frequency domain.  $8 \times 8$  block size is used for the transform. That is, each MB is split into six  $8 \times 8$  blocks for the transform.

#### 1.2.1.3. Quantization

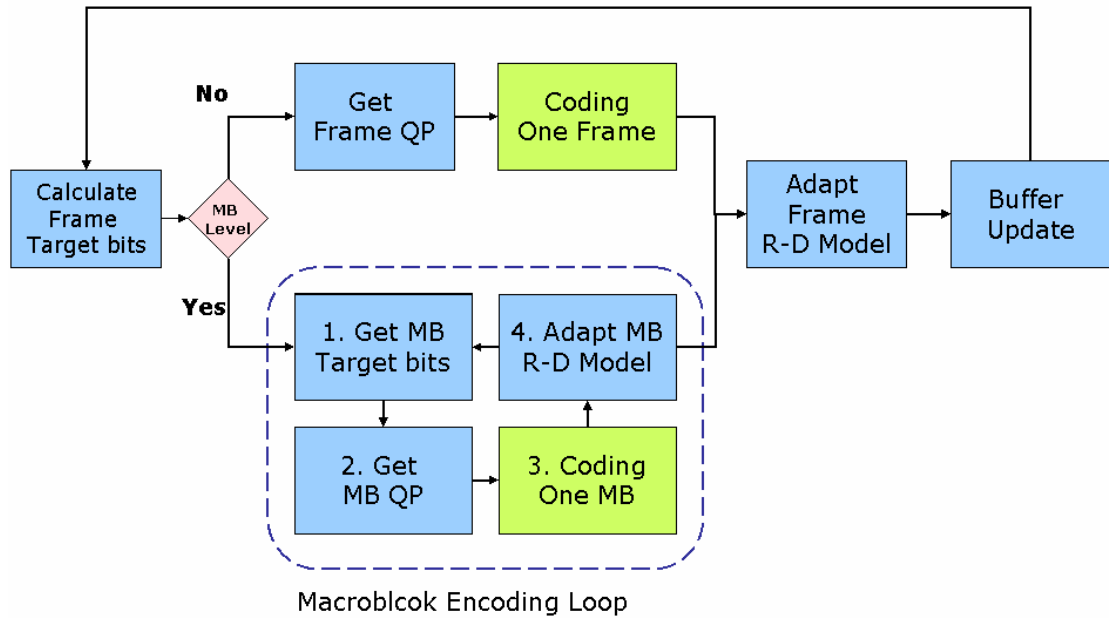
The quantization process reduces the entropy of the source data. This is the key technique to lossy coding. A properly-designed non-uniform quantizer can produce much better results than a uniform quantizer. However, for real-time video applications, uniform quantizer is often used. The RC module determines the quantization step size (QP) as a tradeoff between quality and distortion.

#### 1.2.1.4. IDCT and Inverse Quantization

In order to do predictive coding for the next frame, reconstructed frame should be stored in the reference frame buffer. Hence, there is a video decoder (minus the entropy decoder) embedded inside the encoder. Inverse quantization and IDCT are used to reconstruct spatial data from DCT coefficients which are different from the original coefficients due to quantization effect.

## 1.2.2. Rate Control Illustration

In Fig 1, the blocks which are in dashed lines are an example of rate control points in a video encoder. As mentioned in previous section, there are two types of rate control, namely, frame-level RC and macroblock-level RC.



**Fig 2. Rate Control Flow**

Fig 2 shows the flow of rate control. The top path of Fig 2 is for frame-level control, so the rate-distortion curve is modeled using entire frame; while the bottom path is macroblock-level control, which uses MBs as modeling units and adapts quantization parameters for each MBs. The main tasks for a typical rate control algorithm are described in the following sections.

### 1.2.2.1. Calculate Frame Level Target Bits

The essential goal of rate control is to control the output bitrate, so the first task of RC is to compute the bit budget for current frame. Bit budget (a.k.a. bit-allocation) is obtained according to the status of buffer and possible pre-analysis for current frame content. In general, pre-analysis is employed to exploit the frame complexity and some critical information using the technique like image processing, probability modeling, and etc., and this may be one essential factor to improve the performance of rate control.

It is important for RC algorithm to estimate the complexity of a frame/macroblock before the encoding loop starts so that the RC algorithm can controlled the quantization stepsize (QP) to meet the bit budget constraint. If QP is fixed, the more complex a frame/macroblock is the more bits it will take to encode. Since there is a strong correlation between the compressed data size and the

sum-of-absolute-difference (SAD) of motion-compensated frame and current frame, consequently, SAD is often used to represent the degree of complexity.

Good bit allocation algorithm could improve the overall visual quality, even if the same rate-distortion model is used. Therefore, this is a key differentiator among different video encoder implementations.

#### **1.2.2.2. Apply R-D model and Get QP**

Given target number of bits (i.e. data rate), QP can be calculated from the rate-distortion model at macroblock-level or frame-level. Many algorithms approximate R-D curves with polynomial functions, logarithmic functions, or in a transformed domain (e.g. Rho-domain). Selecting a QP that fulfills the bit-budget constraint while maintaining consistent visual quality across macroblocks and video frames is one of the biggest challenges.

#### **1.2.2.3. Update R-D model**

Because R-D model is content-dependent, a RC algorithm must adapt model parameters progressively during the encoding of a video sequence. After QP is determined and used to encode current frame, the actual bits used to code current frame could be obtained, and this information could help correct the R-D model parameters for next encoding iteration (for next frame or next macroblock).

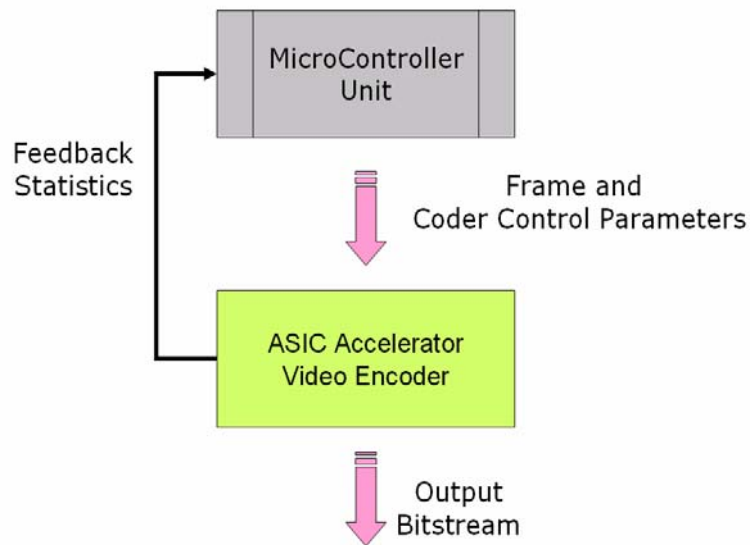
When scene change happens in a video sequence, the model parameters may need to be reset. A common practice is to encode the first frame at the scene change position as an I-frame and restart the RC modeling process from that point.

### **1.3. Research Motivation and Background**

If hardware/software co-design approach is used to implement video encoder, a Microcontroller Unit (MCU) will take care of control flow, set the data path, and handles computation of complex mathematical model. An ASIC accelerator will be used to handle massive data processing (such as DCT/IDCT and motion estimation) to speed up the encoder. Fig 3 illustrates the data flow between MCU and ASIC accelerator.

The shared data between the MCU and the ASIC may includes frame data and coder control parameters (such as the quantization parameters, QP's). Typically, QP's which will be calculated by the MCU and passed to the ASIC accelerator. However, the communication overhead may be expensive since it requires some information from the ASIC accelerator (in particular, the SAD values).

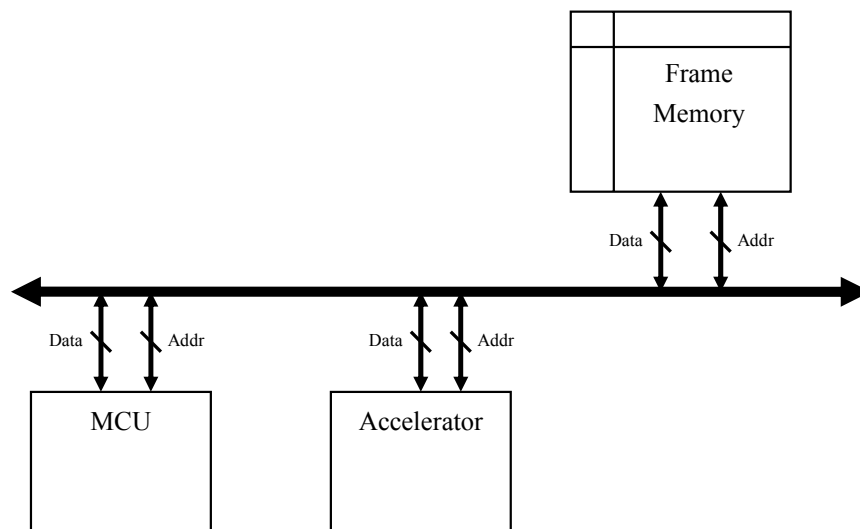




**Fig 3. Data Flow Between MCU and ASIC Accelerator**

Typically, encoder components in solid lines in Fig 1 are often integrated into the ASIC; while the components in dashed lines will be executed by the MCU. Fig 4 is a typical layout of the two cores and the shared memory device.

This thesis tries to provide solutions to two different RC implementations. First, if hardware/software co-design approach is used, a out-of-loop RC algorithm is proposed to reduce the communication overhead between the MCU and the ASIC. Secondly, if pure-hardware RC is desired, a table-look up approach is proposed to simplify the hardware architecture while maintaining accurate modeling of complicated RD curves.



**Fig 4. Dual-Core SoC Platform Architecture**

## 1.4. Problem Analysis

Some analysis on the communication overhead for a hardware/software co-design codec framework is presented in this section. We assume that the ASIC is designed as a BUS master device and interrupt-driven communication between the MCU and the ASIC is used for the system.

### 1.4.1. Communication Overhead of Frame-Level Rate Control

Assuming a 50MHz ARM core and a 100MHz ASIC accelerator is used to encode a 30-Hz CIF resolution video sequence. Also assume that it takes 60 clock cycles to handles an interrupt and transferring SAD to MCU, and 30 clock cycles for the accelerator to get QP from the MCU. The communication overhead can be computed as follows.

$$\text{Accelerator Waiting cycle} = \text{framerate} *$$

$$\text{(clock cycles for interrupt and transfer SAD + clock cycles for accelerator to get QP)}$$

$$= 30 * (60 + 30) = 2700$$

$$\text{Accelerator usage of waiting MCU} = \frac{\text{Accelerator waiting cycles}}{\text{Accelerator total cycles}} = \frac{2700}{100 * 10^6} = 2.7 * 10^{-3}\%$$

$$\text{MCU interrupt cycles} = \text{framerate} * \text{clock cycle for interrupt}$$

$$= 30 * 60 = 1800$$

$$\text{MCU usage of interrupt} = \frac{\text{MCU interrupt cycles}}{\text{MCU total cycles}} = \frac{1800}{50 * 10^6} = 3.6 * 10^{-5} = 3.6 * 10^{-3}\%$$

### 1.4.2. Communication Overhead of Macroblock-Level Rate Control

$$\text{Accelerator Waiting cycle} = \text{framerate} * (\text{macroblocks per frame}) *$$

$$\text{(clock cycles for interrupt and transfer SAD + clock cycles for accelerator to get QP)}$$

$$= 30 * (22 * 18) * (60 + 30) = 1069200$$

$$\text{Accelerator usage of waiting MCU} = \frac{\text{Accelerator waiting cycles}}{\text{Accelerator total cycles}} = \frac{1069200}{100 * 10^6} = 1.0692\%$$

$$\text{MCU interrupt cycles} = \text{framerate} * (\text{macroblocks per frame}) * \text{clock cycle for interrupt}$$

$$= 30 * (22 * 18) * 60 = 712800$$

$$\text{MCU usage of interrupt} = \frac{\text{MCU interrupt cycles}}{\text{MCU total cycles}} = \frac{712800}{50 * 10^6} = 1.4256 * 10^{-2} = 1.4256\%$$

According to the analyses above, while using macroblock-level rate control, the accelerator usage for waiting MCU is about 1.0692%, and MCU usage of interrupt is almost up to 1.42%. However, this is under the condition that MCU does not

execute any operations for rate control algorithm. The actual waiting cycles will be larger in real cases.



## 2. Previous Work

Many rate control algorithms have been proposed for various scenarios. Depending on the applications, the policies should be targeted at fulfilling some constraints, such as buffer size, encoding delay, etc., mentioned in previous sections. Some of these published rate control algorithms are discussed in this chapter. The sections are classified based on the characteristics of the algorithms.

### 2.1. Number of Encoding Passes for RC

Some rate control methods encode each image block several times with different quantization parameters before it decides on the best parameter [4][6][10]. By multiple-pass encoding, the calculated rate-distortion models are more accurate and improves coding efficiency. On the other hand, the coding time and complexity increase a lot. Therefore, these methods are usually used for off-line encoding.

### 2.2. Encoding Delay and RC

In real-time communication systems, the end-to-end delay for transmitting video data needs to be very small, particularly in two-way interactive applications such as videophone or videoconferencing. Rate control designed for low-delay environment should take into account of delays for processing, buffering, and transmission [11][12]. To reduce the encoding delay, these techniques keep the complexity as modest as possible by using simple R-D model and low computational buffer control. To reduce transmission delay per frame (or over a small time window), the RC algorithms should avoid generating large frames at highly complex video scenes. In some applications, VBR rate control with low delay ability could be use for real-time encoding for the purpose of digital storage [21].

### 2.3. Rate-Distortion Optimization and RC

The goal of rate-distortion optimization (RDO) is to achieve best quality-distortion tradeoff by selecting proper encoding modes and motion vectors. Although RDO can be used independently to the RC module, optimal results only achieved when both algorithms are designed jointly. Some rate control mechanisms employ Lagrangian optimization techniques [4][5][17] to exploit the better control parameters. Typically, these methods measure the rate-distortion characteristics and perform pre-analysis on future frames. Some methods apply dynamic programming to get the optimal control point by using previous/future frame information [3]. In

general, more memory and computation are required for this kind of approaches.

## **2.4. Rate-Distortion Model for RC**

In order to estimate correct rate and distortion control points, different types of R-D models are proposed. Among these models, some are based on the modified version of the classical R-D functions which uses logarithmic expressions [11]. Other types of models use power functions [6] and polynomial functions [8]. These R-D models try to estimate the correct behaviors under some criteria such as bitrate, delay, etc. Traditionally, the distortion is specified in QP domain. Recently, a new R-D function in different domain is proposed for rate-distortion analysis [18]. By transforming the R-D function to a different domain, it can be shown that better quantization parameters can be obtained to satisfy given rate constraint.

## **2.5. Frame Dependency**

Video coding removes large redundancy by using predictive coding, in which a given frame is dependent on previous and/or future frames. Good rate control algorithms should take prediction types into account to achieve better results [10][11][12]. For example, increasing the bits for a P frame would decrease the distortion in that P frame and, equivalently, increase its PSNR. This would likely increase the PSNR of the B frames which predicted from this P frame as well. Equivalently, I frames also could have the same effect on other frames.

## **2.6. Variable Frame Rate RC**

To enhance coding efficiency, some rate control algorithms using variable frame rate to achieve good balance between spatial and temporal quality [17][20]. However, variable frame rate would introduce flickering and motion jerkiness, which degrade video smoothness. Some methods are also presented to reduce flickering effects by finding good tradeoff in both spatial and temporal quality.

## **2.7. RC for SoC Platforms**

On SoC platforms, both available computational power and memory bandwidth are limited. Some rate control methods are proposed for SoC platforms [19] with lower requirements on memory bandwidth. In order to improve the performance on SoC platforms, this thesis proposes an algorithm for hardware/software co-design implementation and another algorithm for pure VLSI implementation. The experiments show promising performance compared to other more complex rate control methods for more powerful platforms.

### 3. Proposed RC Using Estimated Complexity

The proposed method is based on the first order Rate-Distortion (RD) model [15], in which

$$Q \cdot R^\lambda = X, \text{ where } Q \text{ is QP, } R \text{ is bit, } X \text{ is complexity and } \lambda=1, \quad (1)$$

to estimate the quantization step size used in next encoding. A typical rate-control algorithm uses the SAD of current frame (or previous macroblocks) computed by the motion estimator as a complexity measure in order to estimate the quantization step sizes. Even though more complex RD model has been proposed described in previous chapter, it is not suitable for hardware implementation. To facilitate hardware/software co-design implementation, the quantization step size must be estimated before entering the macroblock encoding loop. The rationale here is to reduce the amount of interaction between the MCU core and the dedicated multimedia ASIC. In this section, an algorithm is proposed to estimate the complexity of current frame using statistic information.

#### 3.1. Rate control using a first order RD model

The expression of proposed rate control could be reformatted as following first order RD model based on Eq (1),

$$R = \alpha \cdot \frac{C}{Q} \quad (2)$$

,where R is the frame size (in bits), C is the complexity of the frame, and Q is quantization step size (QP) which controls distortion. The scaling factor  $\alpha$  is a parameter that adjusts the RD model to fit different types of video contents. It is important to note that  $\alpha$  is not time-invariant. It can vary from one frame to the next. However, in this section, we assume that  $\alpha$  is a constant throughout the video sequence.

##### 3.1.1. Frame Level Rate Control

If R, C, and Q for the previous frame and R and C for the target frame are available, two equations could be established:

$$\alpha \cdot C_{target} = R_{target} \cdot Q_{target} , \quad (3)$$

$$\alpha \cdot C_{prev} = R_{prev} \cdot Q_{prev} .$$

Hence, the following equation can be used to compute the current quantization step size  $Q_{target}$  (namely, QP):

$$Q_{target} = \frac{C_{target} \cdot R_{prev} \cdot Q_{prev}}{C_{prev} \cdot R_{target}} \quad (4)$$

Typically,  $R$  stands for the size of the frame (in bits),  $C$  is estimated from SAD for motion-compensated P frames and the sum of pixel deviation for intra-coded I frames, and  $Q$  is measured by the quantization step size QP. Large QP implies high distortion. Even though this is a very simple RD model, it is shown later that the model performs very well against more sophisticated models.

In order to avoid bitrate fluctuation, the target size of the frame,  $R_{target}$ , is computed using a sliding window of  $N$  frames based on the target bitrate  $B$  and the target frame rate  $F$ . In addition, efficient buffer utilization is preferred, so it is necessary to maintain the buffer utilization at some specific percent of buffer usage,  $\omega$ , (i.e.  $0 < \omega < 1$ ). The average target frame size can be calculated as  $B/F$ , and the initial video decoder buffer size is  $V_0 = (N \times B/F) \times (1 - \omega)$ . If previously coded frame is  $K$  bits, and the available video decoder buffer size is  $V$  then the updated available decoder buffer size is  $V' = (V - K) + B/F$ . For the next frame, the target frame size is  $P_{target} = V' + B/F - V_0$  for a P-frame.

Since the average I frame size is about three or four times bigger than a P frame ([16], Annex-L), the target frame size for I frame is  $k \times P_{target}$  ( $k = 4 \sim 9$ , depending on the characteristics of the video sequence). The rate control algorithm must compute QP using Eq (4). Before the encoder enters the macroblock coding loop, the only term that is missing from Eq (4) is the target frame complexity estimate  $C_{target}$ .

### 3.1.2. Macroblock Level Rate Control

If the MB level rate control scheme is adopted, each QP of MB should be calculated. By considering two sets of MBs, non-coded and coded MBs, the two equations are established:

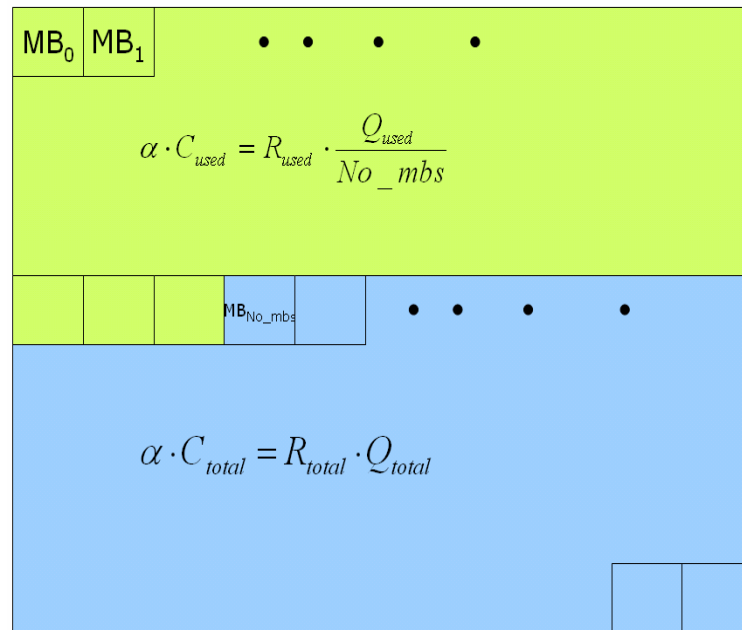
$$\alpha \cdot C_{total} = R_{total} \cdot Q_{total},$$

$$\alpha \cdot C_{used} = R_{used} \cdot \frac{Q_{used}}{No\_mbs},$$
(5)

in which  $C_{total}$  is total complexity (SAD for inter MB and deviation for intra MB) in remaining MBs,  $C_{used}$  is the complexity of coded MBs,  $R_{total}$  is the number of bits which could be used to encode remaining MBs in current frame,  $R_{used}$  is the number of bits used to encode previous MBs,  $Q_{total}$  is the average QP of remaining MBs,  $Q_{used}$  is sum of QPs of previous MBs,  $No\_mbs$  is the number of encoded MBs, and  $Q_{used}/No\_mbs$  is average QP of coded MBs.

In Eq (5), the first equation shows that under ideal condition, the remaining MBs could be encoded with  $R_{total}$  bits, average QP of MBs is  $Q_{total}$ , and total complexity is  $C_{total}$ . Here, it could be understood that in ideal case, if  $Q_{total}$  is used to encode remaining MBs,  $R_{total}$  bits and  $C_{total}$  complexity is conformed. Consequently,  $Q_{total}$  could be used to encode next one MB.

On the contrary, the second equation shows that under actual condition, the coded  $No\_mbs$  MBs require that total bits is  $R_{used}$ , average QP is  $Q_{used}$ , and complexity is  $C_{used}$ . The following figure shows two sets of MBs, and their modeling equations.



**Fig 5. MB Level Rate Control R-D Modeling**



In terms of previous two equations, the QP of next MB could be obtained

$$Q_{mb} = Q_{total} = \frac{C_{total} \cdot R_{used}}{C_{used} \cdot R_{total}} \cdot \frac{Q_{used}}{No\_mbs}. \quad (6)$$

After encoding current MB, the variables in Eq (6) should be updated to maintain the correctness of coding status.

### 3.1.2.1. Initialization

The initialization of frame level information is the same as what is described in previous section. After applying the frame level mechanism to get QP and available bits for current frame, the QP could be used to encode the first MB. Initially,  $C_{total}$  and  $R_{total}$  are set to frame complexity  $C_{target}$  and target frame bits  $R_{target}$  in Eq (4), respectively. Also  $C_{used}$ ,  $Q_{used}$ ,  $R_{used}$ , and  $No\_mbs$  are set to zeros.

### 3.1.2.2. Get QP for current MB

Use Eq (6) to acquire the QP for current MB and encode current MB. Because of the limitation in standard bitstream syntax such that the different of QPs in current and previous MBs could not exceed +/-2 and some MB modes (i.e. NOT\_CODED and INTER4V) could not change its QP, the QP for current MB should be well-managed to take care of these conditions.

### 3.1.2.3. Update MB level information

After encoding current MB, update the variables in Eq (6) by using following equations:

$$\begin{aligned} C_{total} &= C_{total} - MB\_SAD, \\ R_{total} &= R_{total} - MB\_Bits, \\ C_{used} &= C_{used} + MB\_SAD, \\ R_{used} &= R_{used} + MB\_Bits, \\ Q_{used} &= Q_{used} + Q_{mb}, \\ No\_mbs &= No\_mbs + 1, \end{aligned} \quad (7)$$

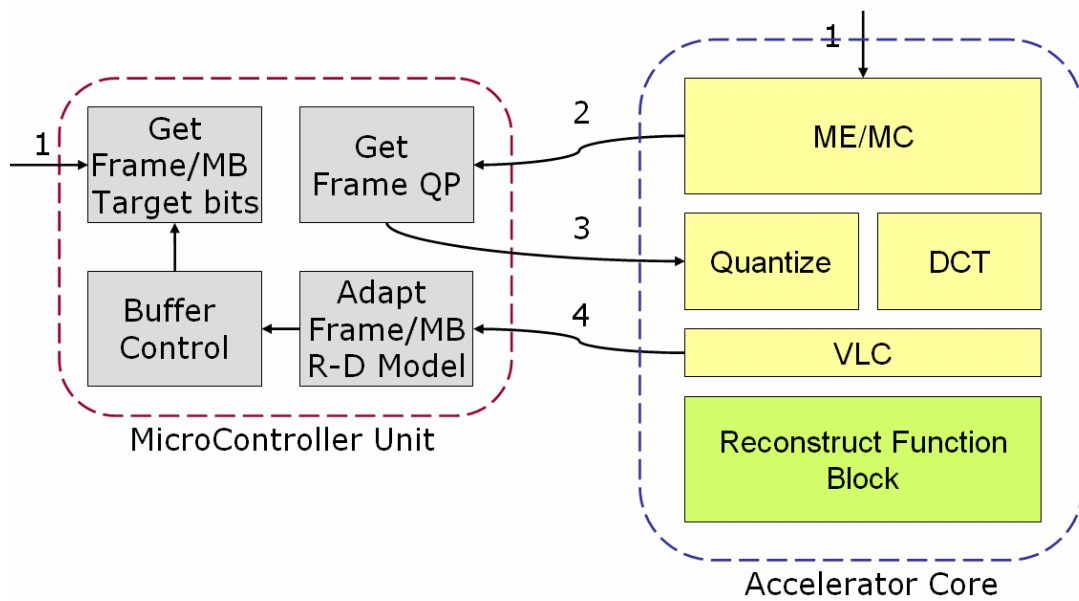
in which  $MB\_Bits$  is the actual number of bits for coding this MB,  $MB\_SAD$  is

the complexity measured in SAD for inter MB and deviation for intra MB. After updating variables, if there is any MB to be encoded, go back to section 3.1.2.2.

### 3.1.2.4. Update frame level information

After all MBs in current frame are encoded, use the same way as frame level in section 3.1.1 to update frame parameters and buffer status for estimating the QP of next frame. Then, continue to encode next frame by repeating steps above.

Most software-based encoder computes the complexity estimates after motion estimation for P frame since SAD are closely related to the complexity of a motion compensated frame.



**Fig 6. Communication Between MCU and Accelerator**

Fig 6 explains the communications between MCU and accelerator. The numbers marked in figure shows the execution order, and the same number means that they could be processed in parallel. In step 2, the SAD for current frame/MB is transferred to MCU and applies R-D model to get QP. After MCU gets QP for frame/MB, QP is transferred back to accelerator in step 3. Then, accelerator continues to quantize blocks and do variable length coding. In step 4, accelerator sends the coding bits and QP for coded frame/MB to MCU to adapt the Frame/MB R-D model. If frame level rate control is adopted, this needs one time to execute 4 steps above and might not be very critical. If macroblock level rate control algorithm is employed, these steps will be repeated as many times as total number of MBs in one frame. The performance would degrade strictly while MCU is busy in jobs.

However, in order to simplify hardware design for a VLSI codec implementation, the proposed algorithm estimates the complexity without resorting to the motion

estimator.

## 3.2. Out-of-Loop Video Rate Control for Software/Hardware co-Design

Motion estimation exploits the correlated information between the reference frame and the target frame and greatly reduces the redundancies of the source data. The coding cost (complexity) of a target frame is measured by the total energy of the motion-compensated error residuals. However, motion estimation requires a lot of computational complexity and memory bandwidth that it is often done in ASIC. On the other hand, rate control requires many computations based on a RD mathematical model that the task is more suitable for a microcontroller (MCU). Therefore, it would be useful if the coding complexity can be estimated outside the motion estimator, either by the MCU or the video preprocessor that is common to most real-time video capturing systems.

### 3.2.1. Frame Complexity Estimation and Frame-Level Rate Control

In this section a low complexity algorithm for the estimation of  $C_{\text{target}}$  is proposed to facilitate SoC based video codec implementations. The estimation is based on the statistics of image pixel differences of the reference frame and the target frame. The algorithm is listed as follows:

- Compute the frame difference between the current frame and the reference frame, scanline by scanline. If necessary, this process can be done on the subsampled frames to reduce memory bandwidth.
- For each  $N \times M$  region,  $R$ , of the difference image, compute its deviation and mean:

$$\text{mean}(R) = \frac{1}{NM} \sum_{i \in R} \text{luma\_diff}_i,$$

$$\text{deviation}(R) = \sum_{i \in R} \text{abs}(\text{luma\_diff}_i - \text{mean}),$$

where  $\text{luma\_diff}_i$  is the component value of the luma channel difference image at pixel  $i$ . Again, if memory access is of concern,  $N$  can be set to 1 so that the procedure is done on a scanline basis.

- Compute the estimated complexity  $\chi$  for the target frame:

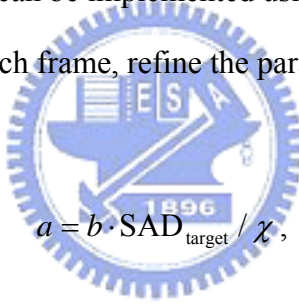
$$\chi = \sum_{R \in \text{frame}} \text{mean}(R) \cdot \text{deviation}(R) \quad (8)$$

- At this point,  $\chi$  describes the complexity of the frame, but it is not in the same scale (unit) as SAD. We must adjust the scale of  $\chi$  to match the scale of SAD for the RC algorithm to work. A simple linear scaling factor ( $a/b$ ) is applied here to do the job:

$$\hat{C}_{\text{target}} = a \cdot \chi / b$$

Note that in order to simplify hardware implementations, the parameter  $a$  is an integer and  $b$  is an integer power of two ( $b$  can be fixed to, say,  $2^{16}$ ). That is, the division can be implemented using right shift.

- After encoding of each frame, refine the parameter  $a$  by the following equation:



$$a = b \cdot \text{SAD}_{\text{target}} / \chi,$$

where  $\text{SAD}_{\text{target}}$  is the true motion compensated SAD value of the target frame. Note that scene change detection can be achieved using  $\chi$  as well. If  $\chi$  is larger than certain threshold, then the target frame should be encoded as an I-frame. In the following section, experiments have been conducted to determine this threshold.

### 3.2.2. Adaptive RD Model

In the first order RD model (Eq. (2)), the parameter,  $\alpha$ , is assumed to be stationary, namely, it stays the same throughout the whole sequence. However, in most cases,  $\alpha$  is content dependent and varies along the video sequence. In order to adopt an adaptive first order RD model, a new parameter,  $\beta$ , is introduced as follows,

$$Q_{\text{target}} = \beta \cdot \frac{C_{\text{target}} \cdot R_{\text{prev}} \cdot Q_{\text{prev}}}{C_{\text{prev}} \cdot R_{\text{target}}} \quad (9)$$

In Eq (9),  $\beta$  is equivalent to  $\alpha_{\text{target}}/\alpha_{\text{prev}}$ , where  $\alpha_{\text{target}}$  and  $\alpha_{\text{prev}}$  are the RD model parameters for the target frame and the previous frame, respectively. After the encoding of one frame,  $\beta$  can be recalculated using Eq (10):

$$\beta = \frac{C_{\text{prev}} \cdot R_{\text{target}} \cdot Q_{\text{target}}}{C_{\text{target}} \cdot R_{\text{prev}} \cdot Q_{\text{prev}}} \quad (10)$$

The parameter  $\beta$  can be computed using a moving average window, that is, by averaging variable of  $\beta$  for each frame in the window.

The adaptive RD model (Eq.(9)) is obtained using the relation of two consequent frames. It is possible to use other adaptive RD models here. If a fixed RD model is used, then  $\alpha$  can be computed at any frame by the following equation:

$$\alpha = \frac{R_{\text{target}} \cdot Q_{\text{target}}}{C_{\text{target}}}.$$

The difference between the fixed RD model and the adaptive model is that in general for low frame rate cases the adaptive model has lower bit variation, while in high frame rate cases the fixed RD model has lower bit variation. This is because that in low frame rate cases, the motion between frames dominates the complexity and the adaptive model utilizes the relation between frames to have a better prediction of QP. With the frame rate becoming higher, the amount of motion decreases, and sometimes the “noise” (unpredictable part of the content) dominates the complexity.

### 3.2.3. Other Adaptive RD Model Methods

Based on the same RD model, one of some factors affecting the visual quality is how to adapt the RD model, how well this is managed is corresponding to how efficient the encoder is. Moreover, each adaptive strategy will result in different characteristic, such as bitrate variation, PSNR variation, decoder buffer usage, and etc. According to the situation and limitation, some adaptive RD model could be design for specific domain and application. Here in order to reduce the complication in MCU, the adaptive RD model method is as dedicated as possible. In addition to the adaptive method in the proposed rate control, moving average window, some other adaptive methods could be under consideration.

Historical Weighted:

$$\beta' = \beta_{history} \cdot (1 - weight) + \beta_{now} \cdot weight$$

where  $\beta_{now}$  is the parameter calculated for current frame,  $\beta_{history}$  is the history for  $\beta$ , and  $weight$  is used to adjust the proportion between  $\beta_{history}$  and  $\beta_{now}$ .

#### Outlier Removal:

- Respectively take the average of R, C, Q, and get the model parameter,  $\beta$ , in a moving average window
- Use the model parameter,  $\beta$ , to compute standard error as threshold, and remove the data point which error is over the threshold
- Recalculate the model parameter  $\beta$  after removing outlier point

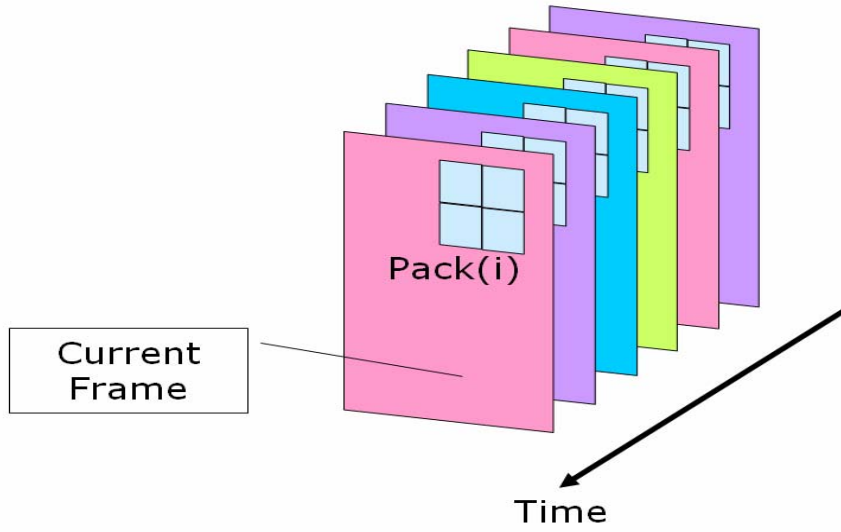
Above these are some direct points of view to solve the adaptation problem of RD model, some RD models using other aspects to treat this trade-off matter, like variable encoding frame rate [17], rho-doman [18], and etc. Although in most advanced adaptable RD models could have good performance and coding efficiency, they are often used in software implementation and not introduced in hardware-based system due to complicated calculation.

### **3.2.4. Macroblock Level Complexity Estimation and Rate Control**

If the macroblock level rate control is employed, the MB level approach in section 3.1.2 could not be used directly even though the macroblock complexity could be estimated by introducing the algorithm in section 3.2.1. Because each QP of MB should be acquired before entering MB encoding loop, there is no information about number of bits used for previous coded MBs. For example, if QP of the 10th MB would be computed using Eq. (6), because the number of bits used to encode previous 9 MBs could not be obtained before entering encoding loop, the formula cannot be applied. Consequently, a novel algorithm is proposed to take care of this issue.

#### **3.2.4.1. Rate-Distortion Modeling**

Because each MB QP is needed before the encoding is started, the only information which could be obtained is the results of previous encoded frames, and the rate-distortion relation of MB can use the way like frame level to model. In order to exploit correlation between rate and distortion for each MB, the specific modeling unit, pack, which includes U MBs (i.e.  $1 \leq U \leq$  total number of MBs in frame), is chosen.



**Fig 7. R-D Modeling for One Pack**

Fig 7 above shows one 4-MBs pack, and the co-located packs in previous frames are used in R-D modeling. Certainly, the pack are not limited in the shape of square, it could be arranged in rectangle or any other shape of MBs according to the image content.

Based on the first order R-D model, each pack in current frame could apply following formula to get its QP:

$$Q_{cur\_pack} = \frac{C_{cur\_pack} \cdot R_{pack}}{C_{pack} \cdot R_{cur\_pack}} \cdot \frac{Q_{pack}}{No\_mbs}, \quad (11)$$

where  $C_{cur\_pack}$  is the complexity for current pack,  $C_{pack}$  is the complexity of previous frame measured by SAD,  $R_{cur\_pack}$  and  $R_{pack}$  are the target bits for current pack and coded bits for previous pack, respectively.  $Q_{pack}$  is the sum of QPs for previous frame and  $No\_mbs$  is the number of MBs in pack. Equally, the only term missed in Eq. (11) is the  $C_{cur\_pack}$ , and this could be estimated using proposed algorithm. In this way, all QPs of MBs in current pack could be given as the same value,  $Q_{cur\_pack}$ . Obviously, choosing adequate size of pack is a critical consideration for accurate and stable modeling.

### 3.2.4.2. Macroblock Complexity Estimation

In order to get QP for each macroblock using Eq. (11), the complexity of each pack could be estimated using following procedure which is similar to estimation for frame complexity:

- Compute the frame difference between the current frame and the reference frame. If necessary, this process can be done on the subsampled frames to reduce memory bandwidth.
- For each  $N \times M$  region,  $R$ , of the difference image, compute its deviation and mean:

$$mean(R) = \frac{1}{NM} \sum_{i \in R} luma\_diff_i,$$

$$deviation(R) = \sum_{i \in R} abs(luma\_diff_i - mean),$$

where  $luma\_diff_i$  is the component value of the luma channel difference image at pixel  $i$ . Because the complexity of each pack comes from each region in each pack, the size of  $N \times M$  should choose suitable value.

- Compute the estimated complexity  $\chi_{pack}(i)$  for the each pack:

$$\chi_{pack}(i) = \sum_{R \in pack(i)} mean(R) \cdot deviation(R) \quad (12)$$

- $\chi_{pack}(i)$  describes the complexity of the  $i$ -th pack, but it is not in the same scale (unit) as SAD. We must adjust the scale of  $\chi_{pack}(i)$  to match the scale of SAD for the RC algorithm to work. The same linear scaling factor is applied here to do this:

$$\hat{C}_{pack}(i) = a_{pack}(i) \cdot \chi_{pack}(i) / b$$

- After encoding of each frame, refine the parameter  $a_{pack}(i)$  by the following equation:

$$a_{pack}(i) = b \cdot SAD_{pack}(i) / \chi_{pack}(i),$$

where  $SAD_{pack}(i)$  is the true motion compensated SAD value of the  $i$ -th pack. Note that scene change detection can be achieved using sum of all



$\chi_{\text{pack}}(i)$  as well.

The bus overhead savings of this approach, compared with conventional rate-control, are quite large. First of all, there is no lock-step synchronization (at macroblock level in the worse cases) between the MCU and the accelerator core. Lock-step synchronization between two cores is typically achieved via either interrupts or semaphores which are very inefficient for high speed SoC implementation due to loss of parallelism. Secondly, the proposed approach does not require passing of the SAD data (per macroblock) from the accelerator to the MCU over internal bus after each one macroblock is encoded which also saves memory bandwidth and time for bus arbitration.

Because parameters of the R-D model and complexity estimation need to be updated, one interrupt and data transfer for entire frame information are required, and this don't need much time compared to interrupt in encoding loop. Even though only one interrupt per frame is needed, the communication between MCU and the accelerator core is still one cost. If this time could be removed and no communication is need, all the work could be accomplished in the accelerator core. Therefore, one novel pure hardware rate control is proposed in next chapter and the issues described in section 1.3 should also be considered.

### **3.3. Encoding of I-frames**

#### **3.3.1. Scene Change Detection**

When scene change happens, the current frame can not be predicted effectively from the previous frame. In this case, forcing the current frame to be encoded as a P-frame requires more bits. It is better to encode a scene change frame as an I-frame since the coding efficiency could be higher. Furthermore, I frames are independently decodable and therefore are more robust than P-frames. The technique proposed in section 3.2.1 for P-frame coding complexity estimation can be applied to scene change detection as well. If the complexity of a P-frame is too high, it is better off to encode it as an I-frame. In other words, if  $\chi$  in Eq. (8) is larger than certain threshold, the target frame should be treated as a scene change frame.

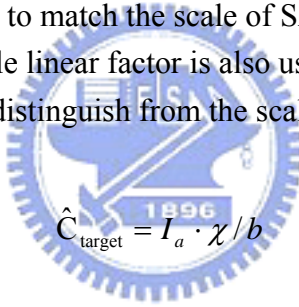
In addition, in order to avoid too many I-frames for low frame rate sequences and for long scene transitions, a constraint is set to enforce a minimal distances between I-frames. The minimal I-frame distance is determined based on the magnitude of the value  $\chi$ .

### 3.3.2. Complexity Estimation for I frame

Before encoding the scene change frame as an I-frame, the quantization step size for I-frame should be determined as well. However, quantization step size can not be computed using the same RD model as that for the P-frame. Because the P frame is motion-compensated while the I-frame is intra-coded, they are different intrinsically coding methods. A different RD model for I-frames is used here. It is also based on the first order RD model (Eq. (2)). By using equation Eq. (4), the quantization step size for I-frame is computed with  $C_{target}$  estimated as follows.

To computing the  $C_{target}$  for an I-frame, the statistics (mean and deviation) are obtained from the luma intensity directly. The algorithm is listed as follows:

- For each  $N \times M$  region,  $R$ , of the target frame, compute its mean and deviation using luma intensity.
- Compute the estimated complexity  $\chi$  for the target frame as the same as Eq. (8)
- Adjust the scale of  $\chi$  to match the scale of SAD for the RC algorithm to work. Here a simple linear factor is also used as before and it introduced another factor,  $I_a$  to distinguish from the scale factor for P-frame:


$$\hat{C}_{target} = I_a \cdot \chi / b$$

Again, a power of two is selected for  $b$ . The initial value for  $I_a$  and  $b$  can be determined by using a two-pass encoding of the very first frame.

- After encoding the target frame as one  $I$  frame,  $I_a$  should be refined according to true encoded frame size.

$$I_a' = I_a \cdot R_{target} / R_{true}$$

where  $R_{true}$  is the true frame size encoded using  $D_{target}$ .

## 4. Pure Hardware Rate Control

Due to the preferred flexibility of rate control approach and to simplify design in the accelerator core, the rate control algorithm is implemented in the way of

software/hardware co-design, which is emphasized in section 3. By using this kind of co-design rate control, the performance of entire system could be promoted and use less time to encode video sequence. Because the time and memory bandwidth used to communicate between MCU and the accelerator core is primary bottleneck, it must be managed more elegant to remedy. In order to reduce the cost of system bus bandwidth and interrupt overhead to the microcontroller for the bit rate update and assignment to each MB, it's suitable to implement a hardware rate control mechanism instead of utilizing the MCU. A hardware rate control implementation inside the MB encoding loop of the accelerator ASIC greatly reduces the number of data transfer and interrupt overhead between the MCU and the ASIC [19].

#### **4.1. Consideration in Hardware Rate Control**

The reason why rate control is preferred being implemented using software is the complicated mathematic model will do harm to hardware design and result in inflexibility and larger cost. In order to implement rate control in hardware, the complex mathematic analysis and lots of memory space must be reduced, and reformatted to be adopted in logic design. In addition, in view point of hardware, the difference of the clock cycles between addition and multiplication are large, so the more multiplication or division operations, the slower the accelerator core is. Hence when designing a rate control algorithm suitable hardware implementation, it is important to introduce simple operations to model rate distortion behavior.

Within this limitation, the R-D model which only uses simple mathematic operations could not work well and the probability to get imprecise QP is large, so a lookup-table approach which needn't model R-D function is proposed in following section.

#### **4.2. Hardware Rate Control Approach**

In general, rate control mechanism exploits correlation between rate and distortion and uses the R-D model to get adequate quantization parameter. After encoding one frame or macroblock, the R-D model should be refined and parameters are re-calculated for next time estimation. Because the parameters in R-D model are sequence dependent and vary with the image content, the accurate number of bits, quantization parameter, and relative information can be obtained only after actual encoding operation.

In order to avoid using complicated mathematic operation to estimate QP and modifying model parameters after encoding either each frame or macroblock, it is not suitable to utilize the R-D model for hardware. By using a different way to model the relation between rate and distortion, a lookup-table mechanism is employed to

record the accurate QP under the situation of current SAD and target number of bits for current frame or macroblock. With this method, there's no need to worry about using a wrong R-D model. Therefore, the proposed method is more adaptive to variable video features. The method utilizes previous accurate data point to acquire one QP for encoding. When one frame or macroblock is to be encoded, the target number of bits and SAD value for current MB are utilized to index the table and acquire the QP. A large amount of interrupt service time for each MB executed in the system core is saved by implementing the h/w rate control with the lookup up table and other logic into the accelerator. Although the SAD and target number of bits are used in this approach, because the lookup table is also implemented in the accelerator core, the SAD could be pass directly from the motion estimator without interrupt to MCU.

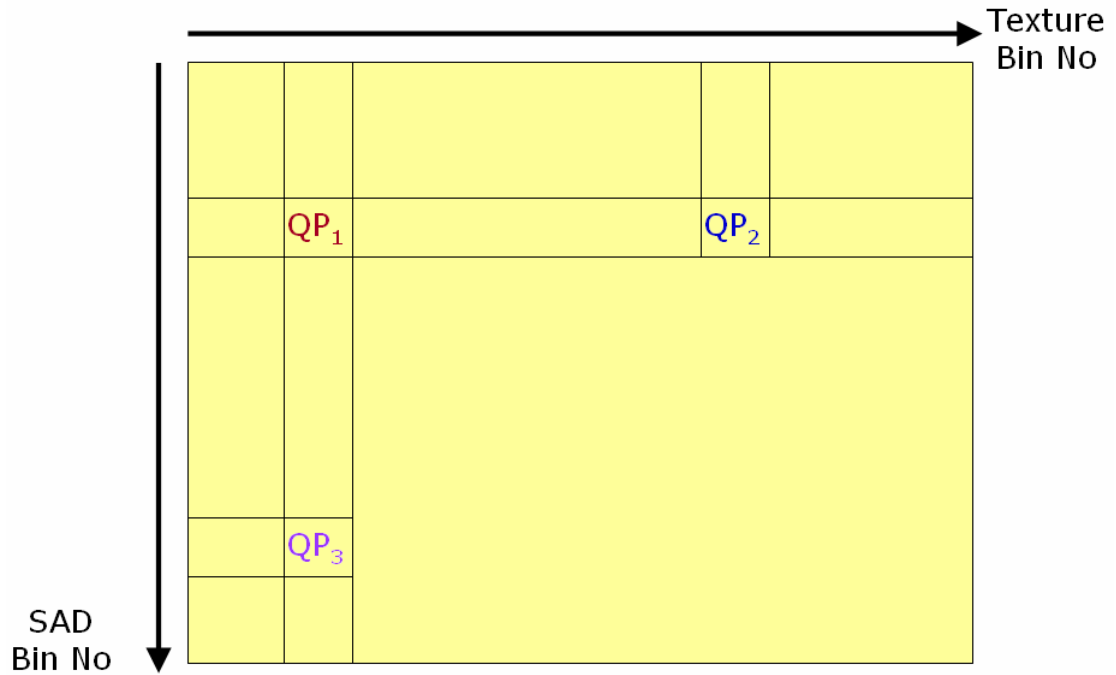
#### **4.2.1. Process Flow of the H/W Rate Control Method**

There are three main processes for the table look-up method. The detail is described as follows.

##### **4.2.1.1. Modeling Table Initialization**

What a rate control mechanism does is to select a quantization parameter according to the target bits of current frame or macroblock and complexity which shows how difficult to encode the residual and is measured as SAD or MSE. Quantization only affects the number of bits used to code quantized DCT coefficients. Therefore, the number of bits which codes texture and SAD are used to index the lookup table.

Before indexing the lookup table to acquire a MB's QP, one of the major issues is how to acquire the initial content of the modeling table. Here the initial modeling table is obtained from the result of encoding some sequence and this modeling table is saved into a storage device such as ROM. When encoding is started, the modeling table stored in the device is loaded into the memory for indexing. Experiments show that different contents of initial modeling table would not affect the coding result strictly.



**Fig 8. Modeling Table in Hardware Rate Control**

Fig 8 shows a modeling table to be looked up in hardware rate control, in which the complexity can be measured by SAD or any other cost functions. The number of texture bits and SAD are used to index this table and both of them are clustered into groups of bins. Naturally, in Fig 8  $QP_1$  is larger than  $QP_2$ , and smaller than  $QP_3$ . This means that at the same SAD value, the more texture bits are available, the smaller the QP is. On the other hand, while having the same texture bits, the larger SAD it has, the larger QP it is. By using this characteristic, the initial modeling table could be interpolated for the entries which are not generated after encoding entire sequence.

#### 4.2.2. Modeling Table Indexing

Once the lookup table (LUT) initialization is done, the rate control can be started. In the beginning of encoding a frame, the rate control computes the frame target bits based on the buffer status. Furthermore, a MB target bit can be calculated in the MB-level. On the other hand, the SAD value is acquired from the motion estimator. The SAD and the target bit for the current MB are used for indexing the LUT. By weighting the QP read out from the LUT with the current average QP, the new QP value for the current MB can be assigned to the quantizer. The weighting mechanism here is to reduce the effect by the inappropriate QP from the table with large variation due to different video sequence features.

### 4.2.3. Update Modeling Table

After texture encoding of MB is done, the actual encoding bits on current MB can be acquired. Such as the updating actions in general rate control algorithm, after encoding one frame or MB, the accurate QP should be saved (updated) into the modeling table indexed by the actual bit number and corresponding SAD values for later bit rate estimation and indexing. By continuously modifying modeling table, the validity of the table can be maintained even though the video sequence varies with image content, and this is somewhat similar with the refinement of R-D model parameters which adapts R-D model to be used with the change of sequence characteristics. In addition to updating the modeling table, updating of the buffer control for next frame encoding is required.

### 4.3. Operations of the H/W Rate Control

The additional arithmetic operations required for the h/w rate control are listed as follows.

- Load the initial modeling table from the on board ROM and save into SRAM in accelerator core. Here we can know that the size of modeling table affects the performance directly because the larger the table is, the more elegant the complexity and texture bits could be described.
- Calculate the target number of texture bits,  $B_{text\_bit}$ , for current frame which is given as follows:

$$B_{text\_bit} = B_{frame} - B_{OH\_bit} ,$$

where  $B_{frame}$  is the total number of target bits for current frame calculated by the buffer update, and  $B_{OH\_bit}$  is the number of overhead bits excluding of texture bits of previous frame. Here, the most up-to-date number of overhead bits is used as a prediction of current frame. Because  $B_{OH\_bit}$  is a predicted value,  $B_{text\_bit}$  should be refined during MB coding process.

- Calculate the average number of overhead bits,  $B_{mb\_OH\_bit}$ :

$$B_{mb\_OH\_bit} = B_{OH\_bit} / MBsInFrame$$

where  $MBsInFrame$  is total number of MBs in one frame.

- Calculate the target number of texture bits for indexing the table:

$$B_{mb\_text\_bit} = B_{text\_bit} / (MBsInFrame - No\_MBs),$$

where No\_MB is the number of MBs being encoded.

- Use  $B_{mb\_text\_bit}$  and SAD to index the table:

$$Bin\_No_{text} = B_{mb\_text\_bit} / Bin\_Width_{text},$$

$$Bin\_No_{SAD} = SAD / Bin\_Width_{SAD},$$

$$MB\_QP = Index(Bin\_No_{text}, Bin\_No_{SAD}),$$

where  $Bin\_Width_{text}$  and  $Bin\_Width_{SAD}$  are select as power of 2.

- Refine QP of MB according current status:

Because of time-variant video content, MB\_QP might not be accurate. Therefore, different weighting with average QP of those previous encoded MBs is adapted based on different  $B_{mb\_text\_bit}$ .

If  $B_{mb\_text\_bit}$  is larger than TEXT\_THRESHOLD which is a constant, it implies that more texture bits are available. Therefore, MB\_QP can be refined as follows:

$$MB\_QP = (MB\_QP \times 2 + AverageQP) / 4.$$

where AverageQP is the average of QP in previous encoded MBs.

If  $B_{mb\_text\_bit}$  is smaller than zero which implies that all the texture bits are exhausted, MB\_QP is set to the largest value.

Otherwise, MB\_QP is assigned as follows:

$$MB\_QP = (MB\_QP + AverageQP) / 2.$$

- After encoding, update the modeling table. Because of the bitstream syntax limitation, the QP might be adjusted, and the modeling table should be updated by using the actual texture bits and QP to index. In order to maintain the historical QP, the QP in the entry indexed is refined with the

value of  $(QP + MB\_QP)/2$ .

- The following variables are required to be updated for further processing.

$$B_{text\_bit} = B_{text\_bit} - Text\_bit + B_{mb\_OH\_bit} - OH\_bit ,$$

$$No\_MBs = No\_MBs + 1 ,$$

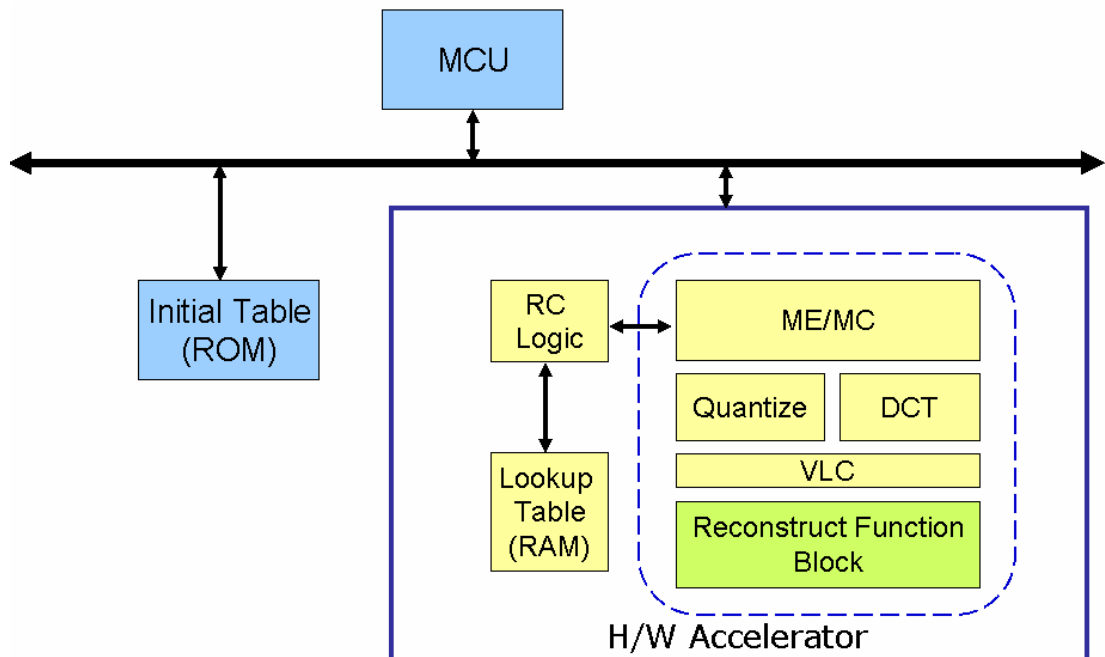
$$TotalQP = TotalQP + MB\_QP ,$$

$$AverageQP = TotalQP / No\_MBs ,$$

where TotalQP is the sum of QPs for all encoded MBs. Text\_bit and OH\_bit are the number of texture and overhead bits for current MB. In order to simplify the operation, the division operation is done and substituted by shifting only when No\_MBs is the power of 2.

- If there is any MB to be encoded in current frame, repeat the procedures above. After encoding one entire frame, do the buffer control described in section 3.1.1, and continue to encode next frame until finish the sequence.

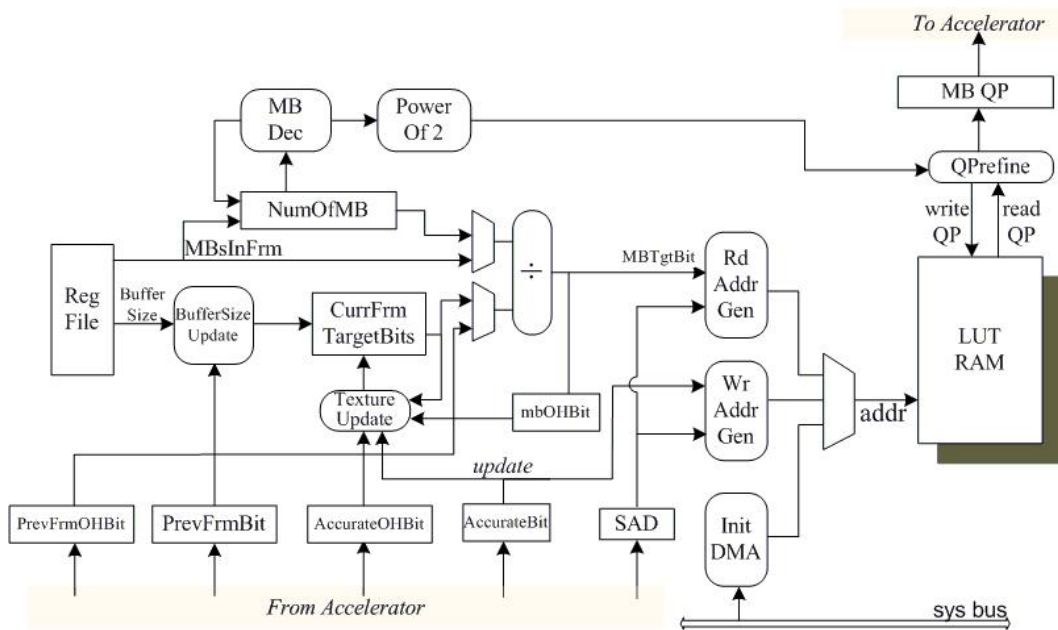
#### 4.4. Hardware architecture of rate control





**Fig 9. System Architecture of H/W Rate Control**

Hardware implementation of the rate control is proposed in this section. Fig 9 shows the system architecture of the rate control with the coding accelerator and the system microcontroller. The initial table is stored in the ROM. As encoding is started, the initiator logic within RC logic block is transferring the table into the internal memory. Also, RC logic block controls all the rate control arithmetic operation as mentioned in 4.3 with inter-communications between h/w rate control and other encoding accelerator.



**Fig 10. Block Diagram of H/W Rate Control**

The overall block diagram for the hardware rate control method is shown Fig 10. Four parameters, the number of overhead bits for previous frame, SAD value, the accurate number of coded texture bits of current MB, and the number of whole frame coded bits, are required from the h/w encoding accelerator. For each frame encoding, the BufferSizeUpdate controls the buffer fluctuation and calculates the target number of bits for a frame. The CurrFrmTargetBits register stores the number of bits available for the rest of MBs in the current frame and will be updated (decremented) by the accurate number of texture encoded bits after each MB encoding. Also, the number of MBs will be decremented by 1 to present how many MBs in the frame are not coded yet. Two divisions are required for calculating the target MB bits in MB-level and average number of overhead bits in frame-level. However, they can share one divider. The maximum of  $B_{mb\_text\_bit}$  and the SAD are 2048 and 4096; while  $Bin\_Width_{text}$  and  $Bin\_Width_{SAD}$  are 64 and 128 respectively. Hence, the LUT

has  $2^5 \times 2^5$  entries with each having 5-bits QP size for MPEG-4 system.

## 5. Experiments and Analysis

In this section, experiments are performed to examine the performances of rate control mechanisms mentioned in previous sections in different view of points. In general, they are separated into two sets, frame level and macroblock rate control, and use variation of PSNR, frame bits, and buffer status to compare the differences. Frame level rate control includes the method, first order R-D model w/SAD (SAD), which uses true SAD as frame complexity measure mentioned in section 3.1.1 and the other one, first order R-D model w/ESAD (ESAD), which uses estimated SAD to solve the problem on SoC platform described in section 3.2.1.

Macroblock rate control includes first order macroblock level R-D model w/SAD (SAD\_MBL), which uses true SAD as complexity for each MB noted in section 3.1.2, first order macroblock level R-D model w/ESAD (ESAD\_MBL), which uses estimated SAD to solve issue of SoC platform explained in section 3.2.4, and Table Based rate control (TBRC), which uses a lookup table to do rate and distortion modeling mentioned in section 4.2. In addition, some well-known rate control mechanisms, e.g., MPEG-4 Annex L rate control [16], Test Model 5 [7], and Rho-domain rate control [18], are also compared in this section.

### 5.1. Testing Environment and Video Encoder Configuration

#### 5.1.1. Environment

In order to have fair experiments, simulations are performed under the same condition, Pentium-4 compatible machine and Windows XP operation system. Source codes of algorithms are compiled with Visual C++ 6.0 using the same optimization setting.

#### 5.1.2. Encoder Configuration

The video encoder is standard compliant, and each rate control mechanisms are implemented on the same software based encoder. In the cause of getting correct simulation results among approaches, it is required to have the same configuration parameters of video encoder. In addition, because some algorithms do not model rate and distortion for I-frame, use the IP...P coding pattern with the same initial quantization parameter to encode sequence. The following is list of parameters used in all rate control approaches.

<b>Coding Pattern</b>	IPP...P
<b>Frame Skipping</b>	Not Used
<b>Profile</b>	MPEG-4 Simple Profile
<b>Data Partition</b>	Not Used
<b>RVLC</b>	Not Used
<b>Slice Mode</b>	Not Used
<b>Quantization Type</b>	H.263
<b>Half Pixel</b>	Used
<b>Inter4v</b>	Used
<b>Motion Estimation</b>	PMV Search
<b>Search Range</b>	32

**Tab 1. Encoder Configuration**

However, the parameters of rate control are not listed in Tab 1 above, and they would be presented in following experiment description.

## 5.2. Frame Level Rate Control Comparison

Because of the different characteristics between frame level and macroblock level rate control, they are tested separately to get fair result. Here three frame level rate control algorithms, SAD, ESAD, and Annex-L are used to compare the performance, and result of SAD\_MBL is used as referenced rate control. Three CIF resolution sequences, COASTGUARD, STEFAN, and FORMAN, are used to test and three kinds of plots, PSNR, bits variation, and buffer status are shown. In order to have correct comparison, the quantization parameter of first frame is set to the same (ex. 10 for COASTGUARD at target rate 256kBps, 16 for FORMAN at target rate 256kBps and 9 for STEFAN at target rate 768kBps). According to Annex N [1], the decoder buffer is set to the size of interval of 0.5 second, which equals to 0.5 times of target bitrate.

Following plots will show the variation of PSNR, number of frame bits, and buffer status frame by frame.

## 5.2.1. PSNR

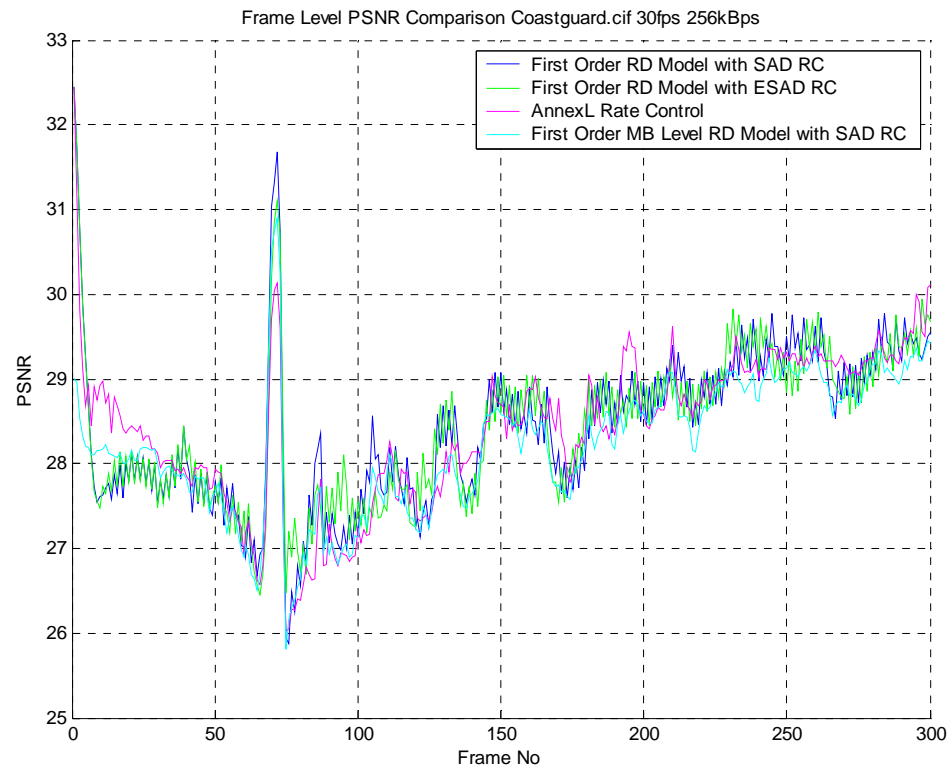
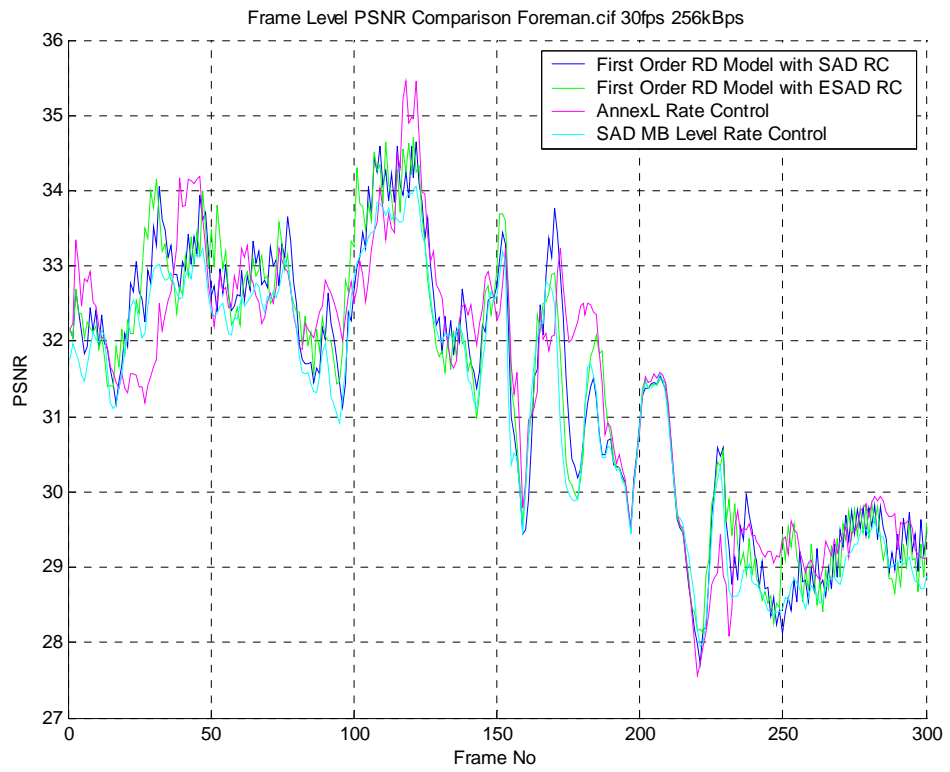
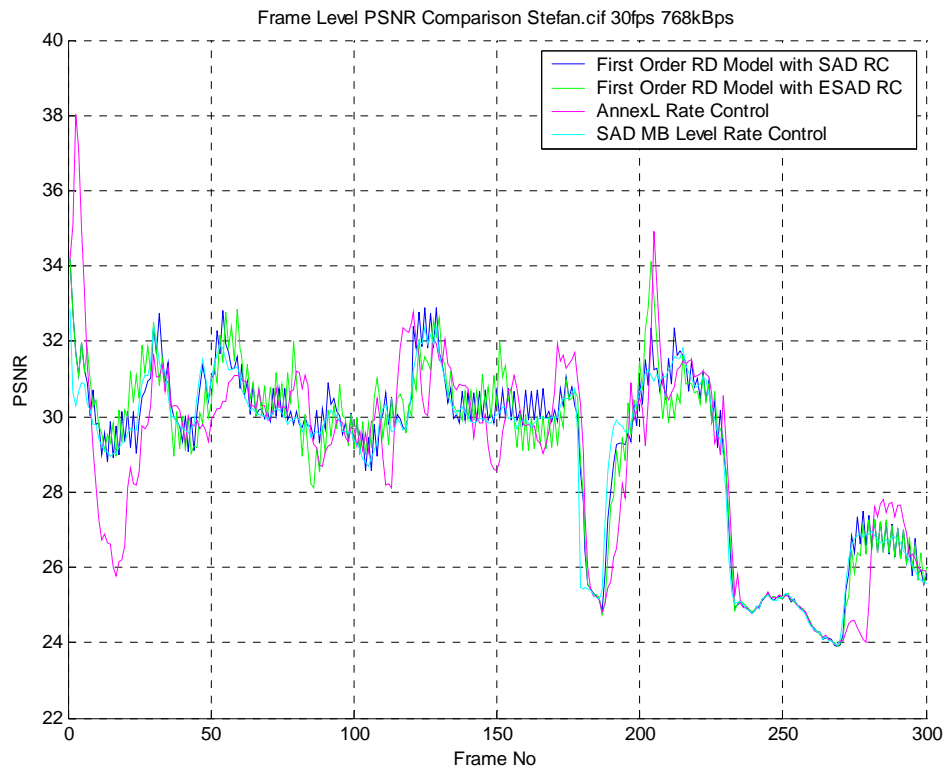


Fig 11. PSNR plots for COASTGUARD sequence





**Fig 12. PSNR plots for FOREMAN sequence**



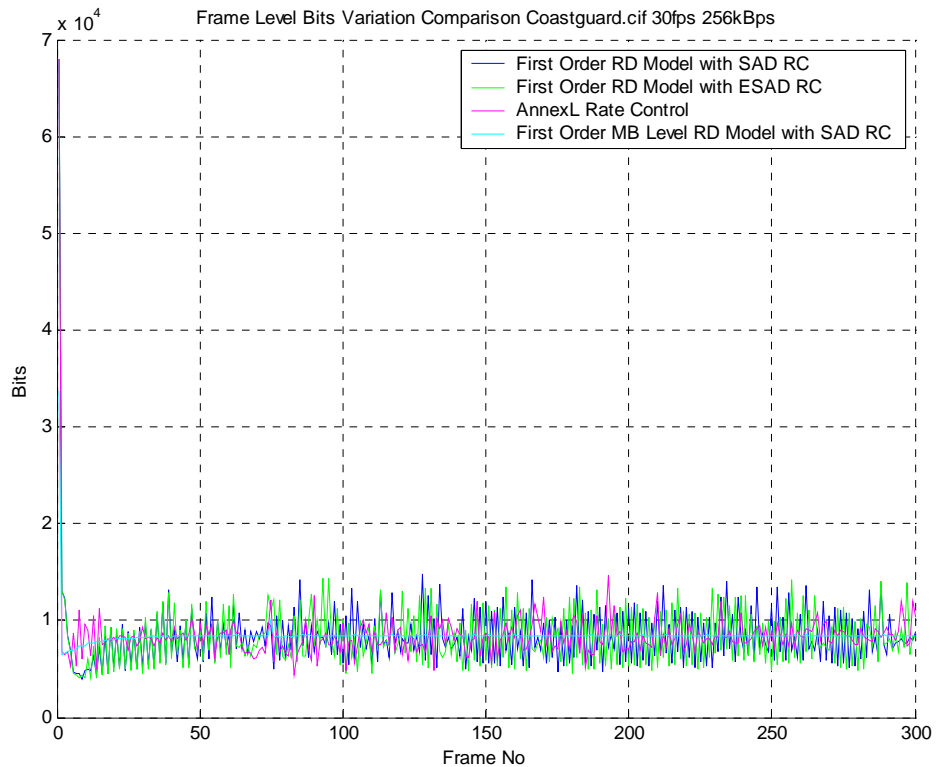
**Fig 13. PSNR plots for STEFAN sequence**

In Fig 11 above, a phenomenon could be observed that when the frame number is about 70, PSNR becomes very good, and after frame number is larger than 74, the PSNR go back original range. It could be guessed that when frame number is about 70, the sequence is almost the same as previous frame, so the motion estimator could find good motion vectors to reduce error residual and the plot of PSNR will generate one pulse.

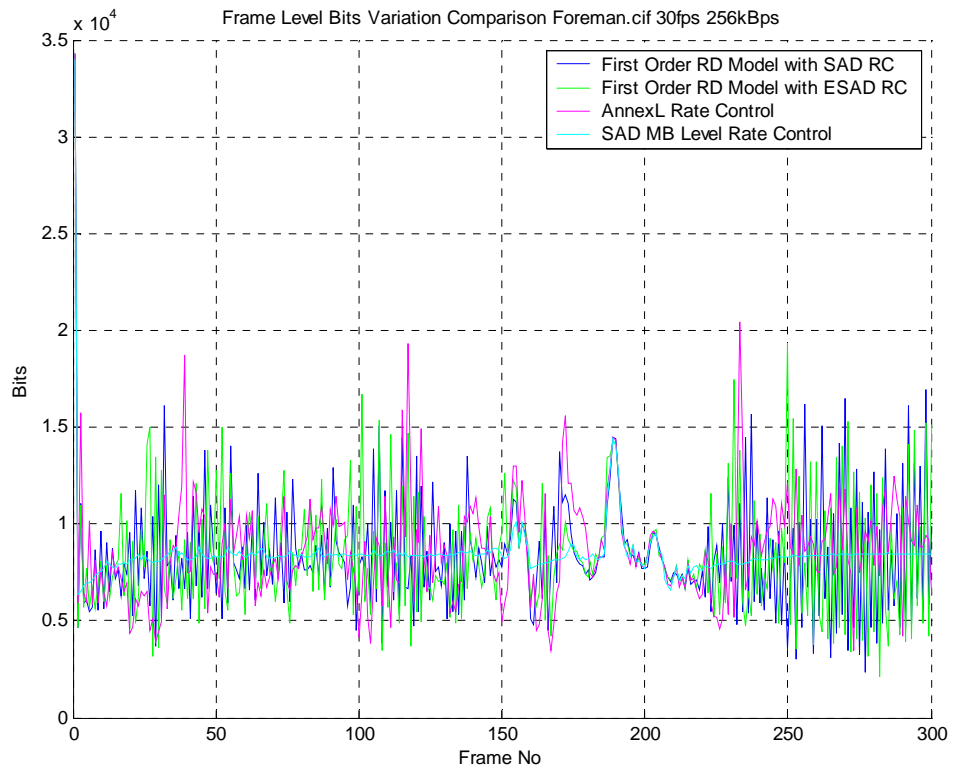
In figure above, proposed algorithm, ESAD, has competitive performance to MPEG-4 Annex L rate control which employs quadratic R-D model to get adequate quantization parameter. While the sequence has larger motion, the four methods have almost the same value of PSNR because of the obvious scene change in sequence.

On the other hand, the result of SAD\_MBL has smaller amplitude and steadier than other frame level methods because it uses smaller block size and can do delicate adaptation for quantization parameter.

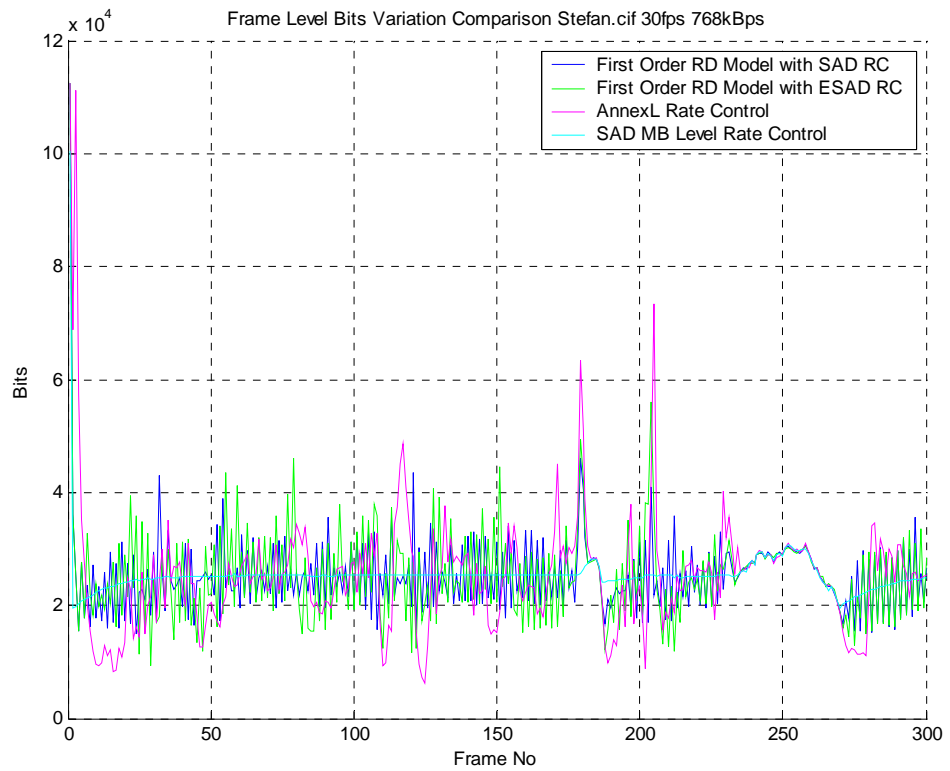
### 5.2.2. Frame Bits Variation



**Fig 14. Frame bits plots for COASTGUARD sequence**



**Fig 15. Frame bits plots for FOREMAN sequence**



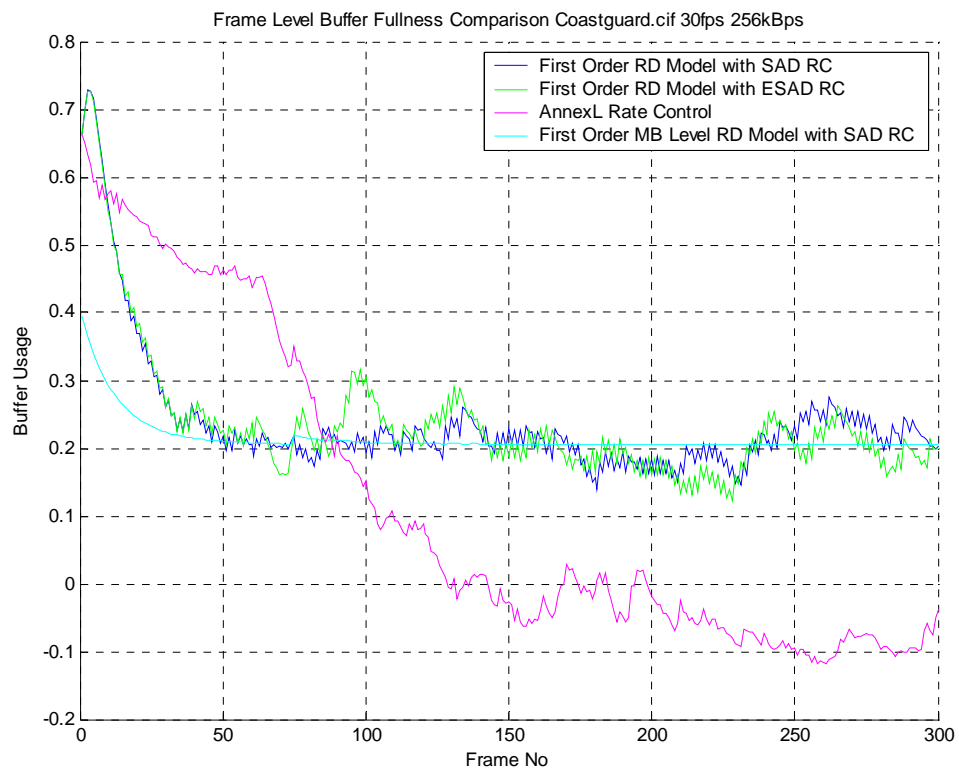
**Fig 16. Frame bits plots for STEFAN sequence**

In Fig 14, for three frame level rate control, the differences among them are limited and Annex L has lower variation of frame bit which uses sliding window to remove outlier data points. In end of FORMAN sequence, frame bits of SAD and ESAD has apparent variation because of repeatedly change between two quantization parameters.

In Fig 16, Annex L has sudden vibration, and generally, three methods has similar result.

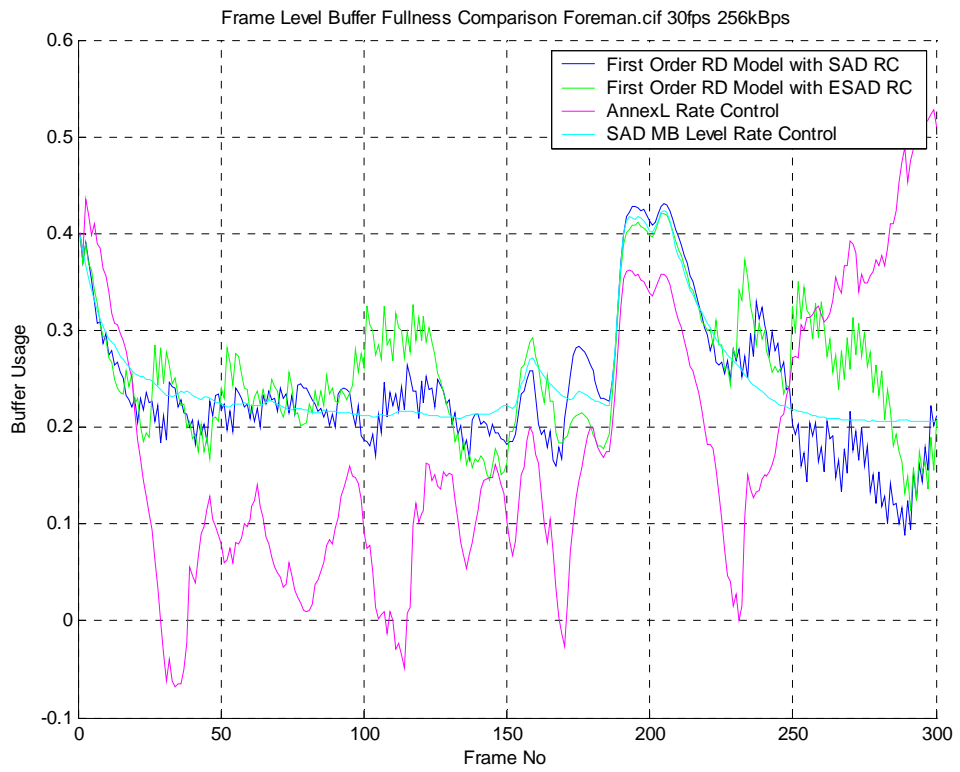
In addition, SAD\_MBL has promising quality and almost the same number of frame bits. Even though SAD\_MBL could match target rate very well, there still has some parts which vary violently and should be scene change segment in which motion estimator could not estimate adequate motion vector to lower down error residual

### 5.2.3. Buffer Status

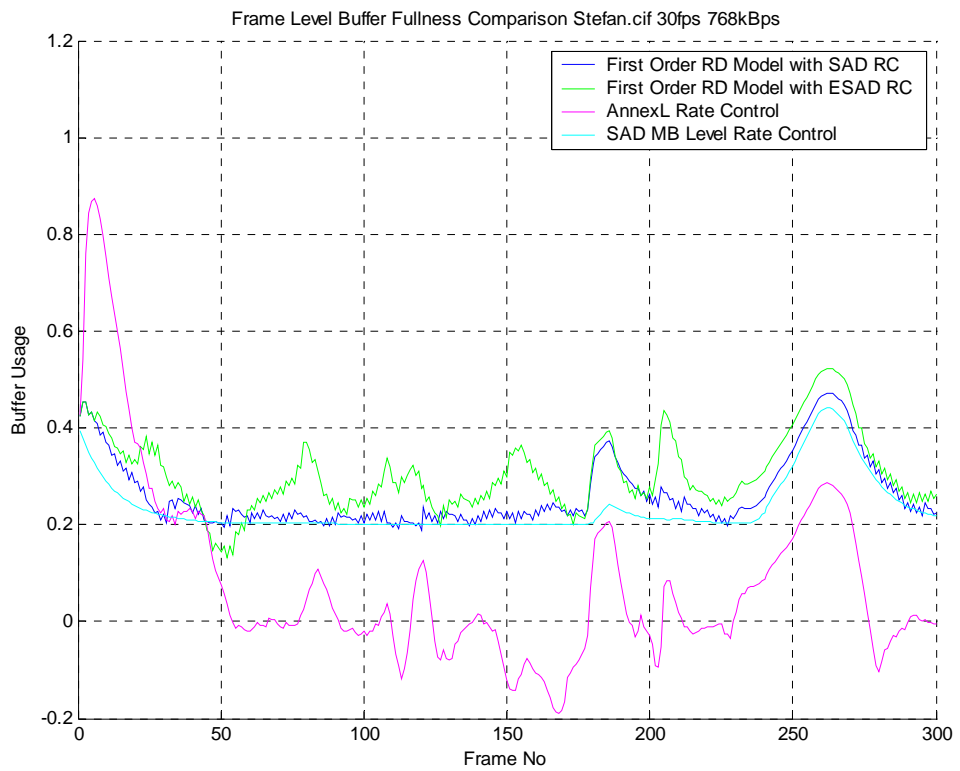


**Fig 17. Buffer status plots for COASTGUARD sequence**





**Fig 18. Buffer status plots for FOREMAN sequence**



**Fig 19. Buffer status plots for STEFAN sequence**

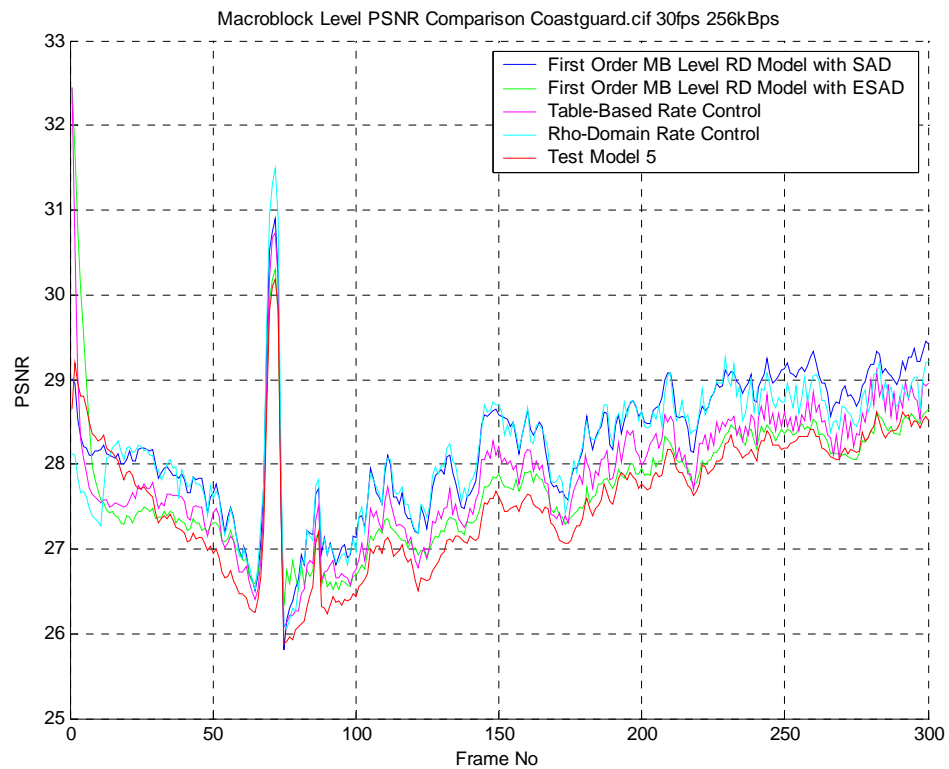
In Fig 17, Fig 18 and Fig 19, it shows ESAD and SAD has stable bits in decoder buffer, and because Annex L rate control uses the idea which limits the number of buffer bits within some range, its buffer status has more different tendency than SAD and ESAD. ESAD has similar variation as SAD, and at scene change point, it has larger variation.

Equivalently, SAD\_MBL has the steadiest buffer status like the plot of frame bits variation because of its good approximation of frame bits.

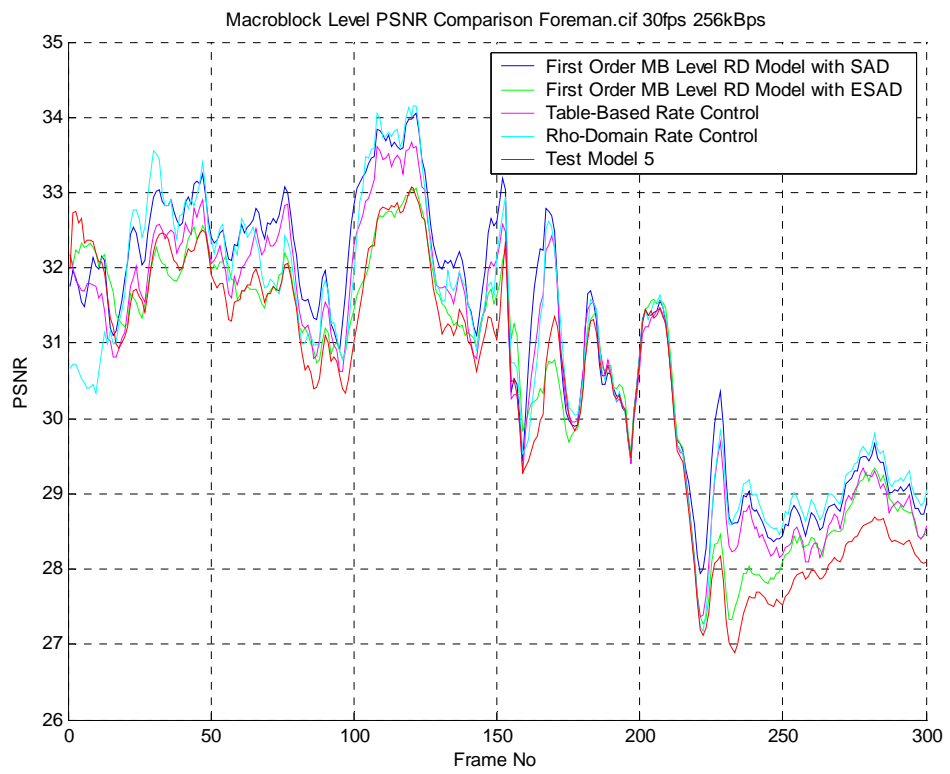
### 5.3. Macroblock Level Rate Control Comparison

Five macroblock level rate control algorithms, which include SAD\_MBL, ESAD\_MBL, and TBRC, are used to test their characteristics and performances. ESAD\_MBL is used to solve the issue on SoC platform mentioned in previous section, TBRC is the approach which is suitable for hardware implementation, and SAD\_MBL is used again to give comparison between frame level and macroblock level rate control. Another strategies, TM5 [7] and Rho-domain rate control described in section [18], are also taken into account.

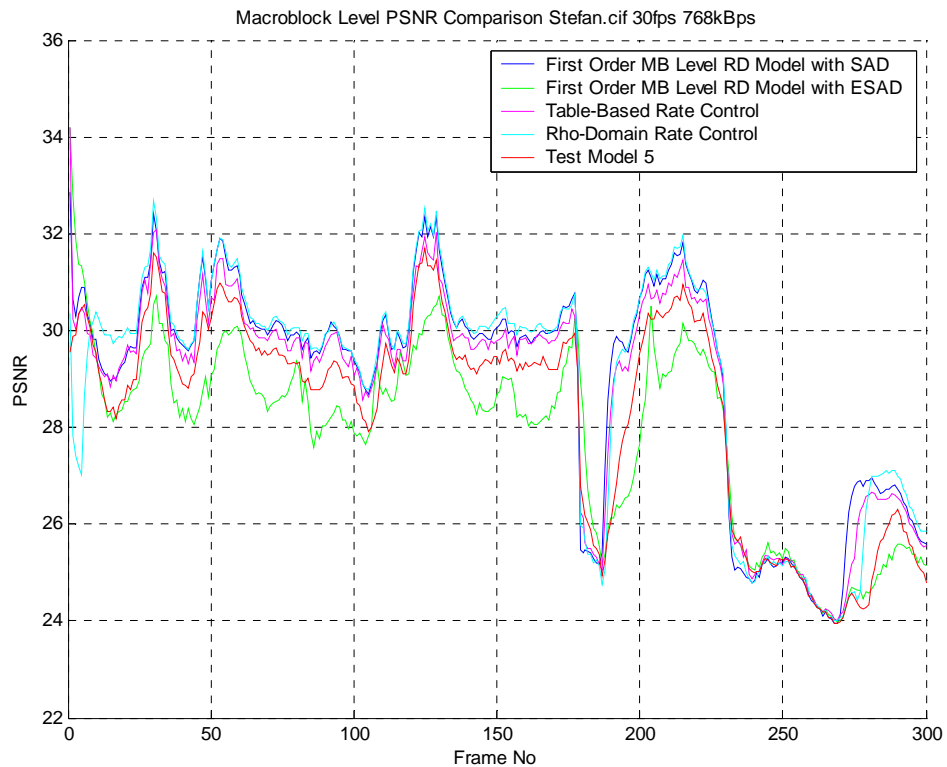
#### 5.3.1. PSNR



**Fig 20. PSNR plots for COASTGUARD sequence**



**Fig 21. PSNR plots for FOREMAN sequence**



**Fig 22. PSNR plots for STEFAN sequence**

Generally speaking, the two rate control methods, ESAD\_MBL and TM5, are not very well. TM5 is designed for MPEG-2 video coding which is adequate for high bitrate encoding and there is no syntax limitation in adaptation of quantization parameter, so this might introduce degradation in quality. ESAD\_MBL, which is used to remedy the interrupt service time, bases on the modeling units in previous frame to obtain QP for current frame. The main issue is that the actual information like QP, SAD, and number of coding bits could not be acquired, so the previous frame is used to estimate QP and results in low quality.

The three mechanisms, SAD\_MBL, Table-based rate control, and Rho-domain rate control, has similar quality. Rho-domain RC applies impressive mapping between QP and percentage of quantization DCT coefficients and has good performance. SAD\_MBL, which uses delicate conception, is close to result of Rho-domain RC. Although TBRC has little degradation on PSNR, it still has result close to Rho-domain RC. In addition, TBRC is adequate for hardware implementation and could be designed with efficient circuit.

### 5.3.2. Frame Bits Variation

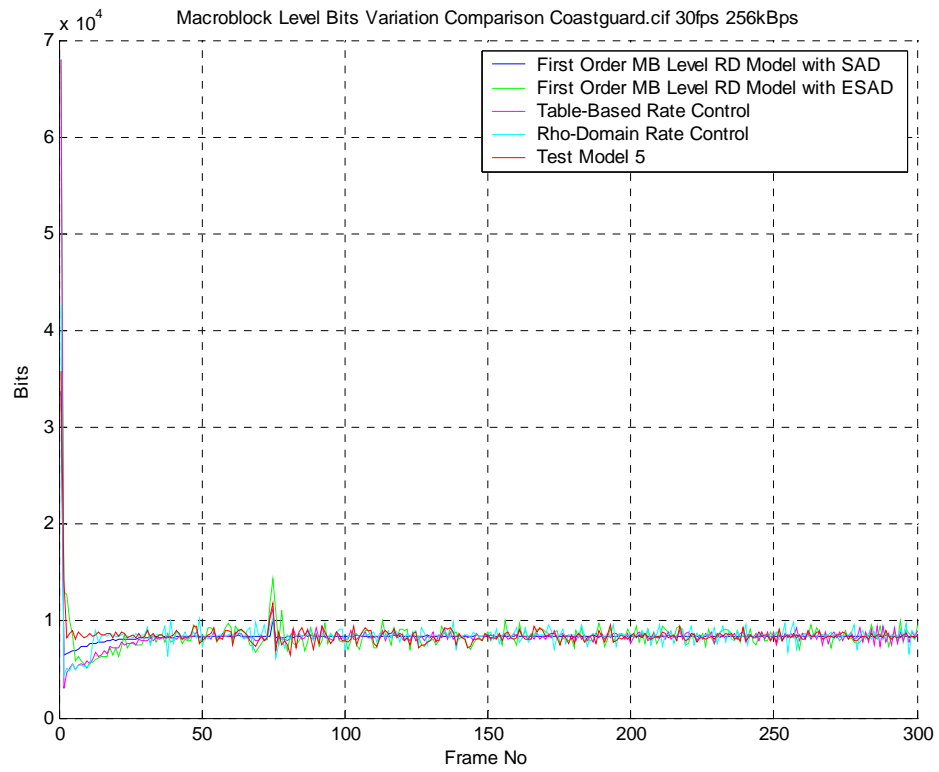
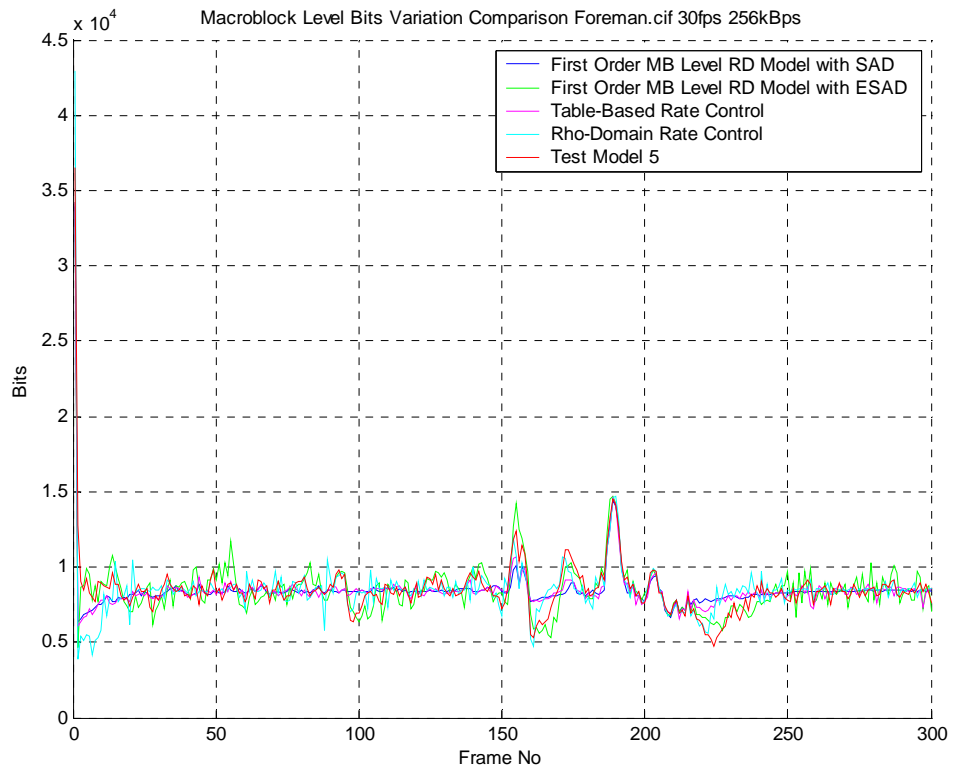
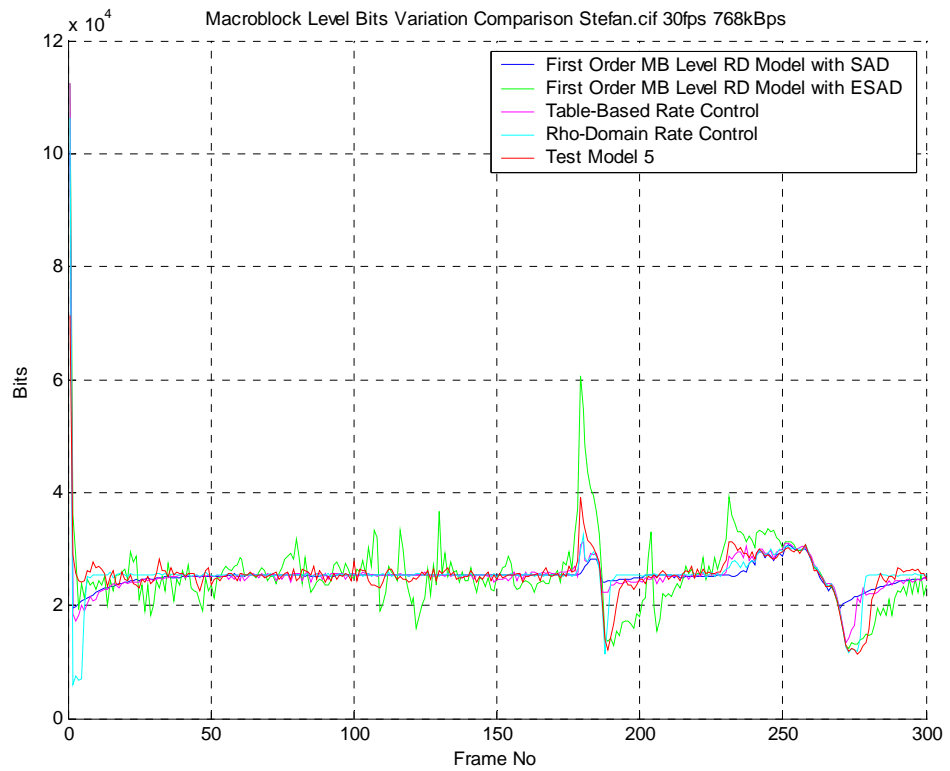


Fig 23. Frame bits plots for COASTGUARD sequence





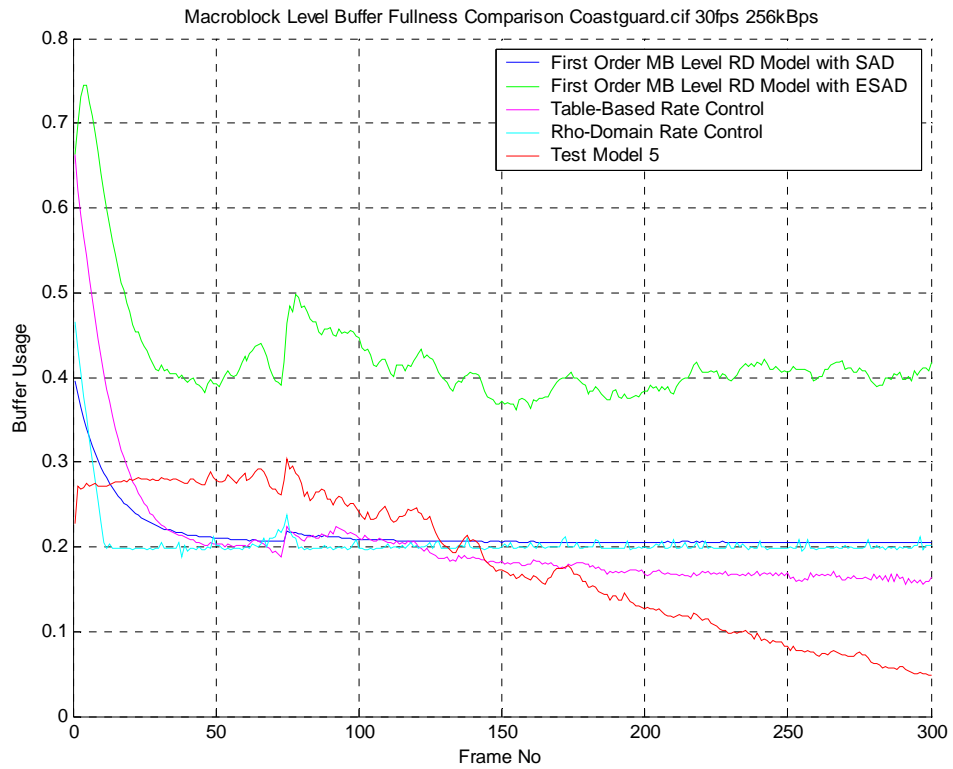
**Fig 24. Frame bits plots for FOREMAN sequence**



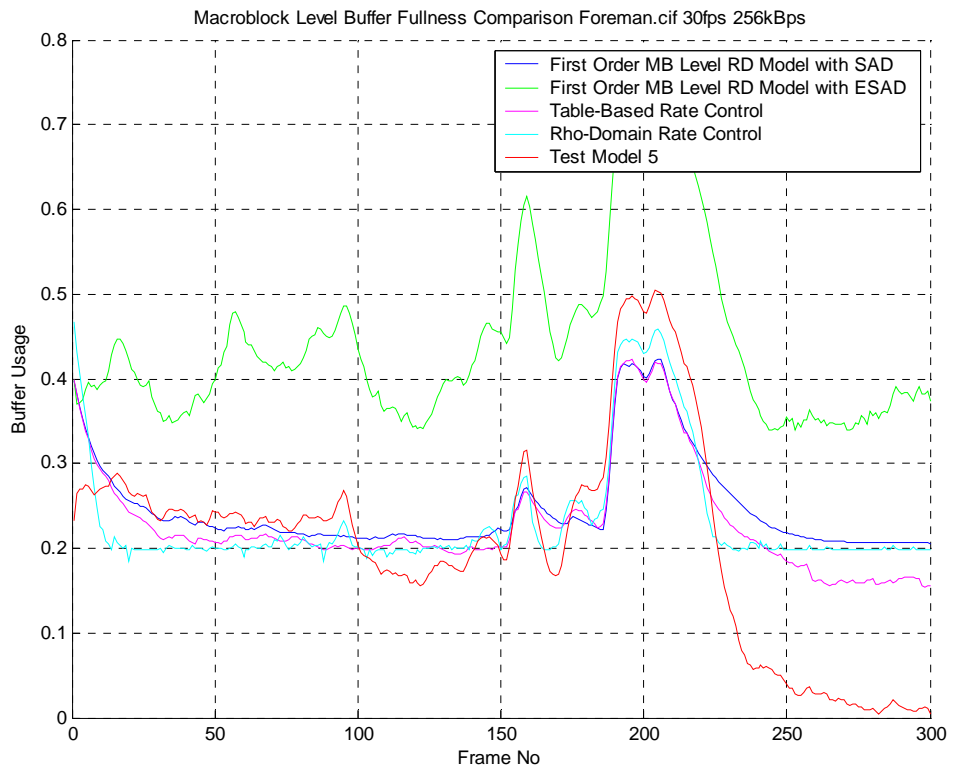
**Fig 25. Frame bits plots for STEFAN sequence**

In the point of view in frame bits variation, ESAD\_MBL has apparent amplitude, which implies the capability of QP prediction is not good and could not control number of frame bits very well. The variation of SAD\_MBL, TBRC, Rho-domain RC, and TM5 are the frame bits similar to the some fixed value, and this implies they could control the output bitrate very well to match the target rate.

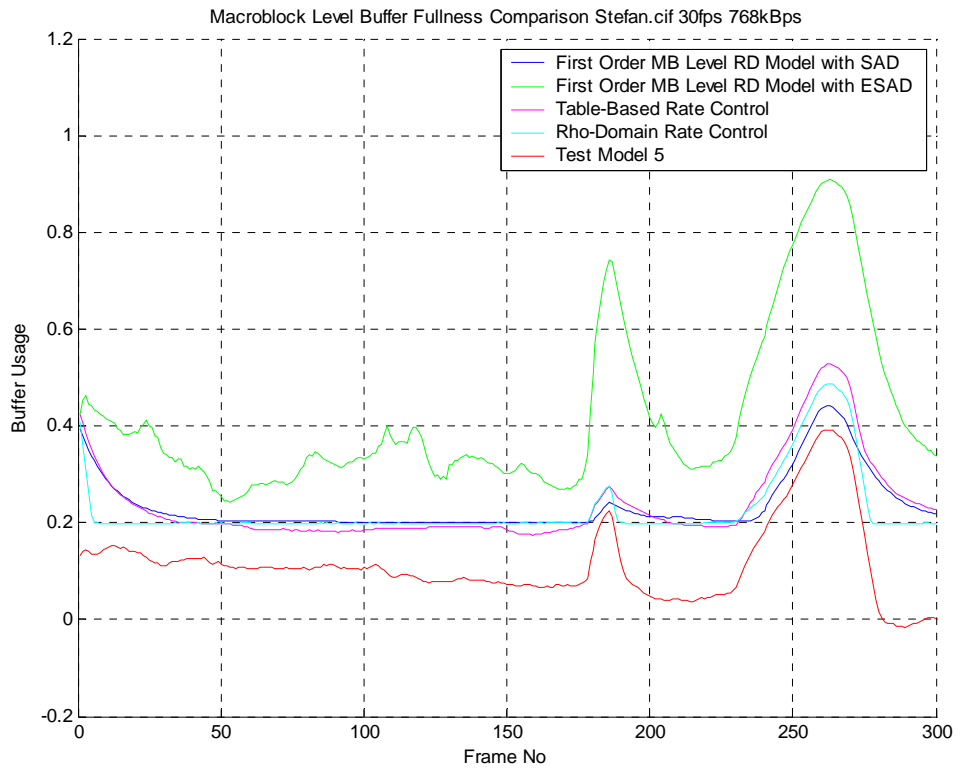
### 5.3.3. Buffer Status



**Fig 26. Buffer status plots for COASTGUARD sequence**



**Fig 27. Buffer status plots for FOREMAN sequence**





**Fig 28. Buffer status plots for STEFAN sequence**

In Fig 26, there is large gap between TM5 and other approaches. This results from the large error of first I-frame and the different design for calculating target frame bit. Other methods use similar updating policies for frame bits, so the difference is small. Such as the plots in frame bits variation, ESAD\_MBL has large amplitude compared to other approaches, and SAD\_MBL has the most stable variation which is close to some fixed value in FORMAN and COASTGUARD sequences and almost one straight line in STEFAN sequence. Because SAD\_MBL distributes the as equivalent number of bits as possible among all macroblocks, it can attain the target number of frame bits more correctly.

## 5.4. Comparisons Among Sequences

Every algorithm described in previous section is used to test its performance. All sequences are CIF resolution and encoded at 30 frames per second.

### 5.4.1. PSNR and Output Bitrate

Following tables show the PSNR and output bitrate using different sequences using different target bitrates.

Sequence Target	SAD	ESAD	Annex L
Foreman 256k	256 (kbps) 31.39 (dB)	255 31.41	264 31.55
Stefan 768k	768 29.12	767 29.12	769 29.04
Coastguard 256k	256 28.45	255 28.46	257 28.44
Table 256k	255 32.73	256 32.74	261 32.76
Mobile 256k	512 24.93	512 24.93	521 24.90
News 256k	258 35.12	259 35.21	263 34.93

**Tab 2. PSNR & Output Bitrate for Frame Level Rate Control**

Sequence Target	SAD_MBL	ESAD_MBL	TBRC	RhoRC	TM5
-----------------	---------	----------	------	-------	-----

Foreman 256k	255 (kbps) 31.18 (dB)	256 30.67	255 30.89	257 31.06	255 30.42
Stefan 768k	767 29.03	768 28.06	767 28.84	769 29.06	761 28.36
Coastguard 256k	255 28.25	256 27.77	255 27.96	258 28.21	255 27.57
Table 256k	256 32.07	256 32.20	255 32.09	257 32.22	255 30.62
Mobile 256k	511 24.74	512 24.63	511 24.63	513 24.82	510 24.35
News 256k	258 34.42	259 34.24	258 34.40	260 34.38	259 32.89

**Tab 3. PSNR & Output Bitrate for Macroblock Level Rate Control**

According to Tab 2 and Tab 3, use the approach, first order R-D model w/SAD (SAD), as the referenced rate control. It could be observed that on average the PSNR of frame level rate control will be about 0.2 to 0.7 dB higher than macroblock level rate control. This results from the intrinsic differences between these two differences of rate control mechanisms in quantization and R-D modeling unit.

In frame level rate control algorithms, proposed first order R-D model w/ESAD (ESAD) gives well PSNR and output bitrate compared to first order R-D model w/SAD (SAD) and Annex L rate control. Generally speaking, there is no obvious difference among these three approaches.

In macroblock level rate control algorithms, proposed first order macroblock level R-D model w/ESAD (ESAD\_MBL) do not give promising result because of the difficulty in estimating actual number of coding bits. In addition, by means of fine analysis in specific domain, the Rho-domain rate control gives the better performance in most sequences. Table based rate control could get acceptable result in employing a lookup table to get quantization parameters, and in this way it could be implemented into hardware circuit. The result of first order macroblock level R-D model w/SAD (SAD\_MBL) is very close to Rho-domain RC because of its excellent ability of controlling frame bits to match target rate.

#### **5.4.2. Mean and Standard Deviation for PSNR Variation**

Following tables show the average and standard deviation for PSNR using different sequences at different target bitrates. In the entries of the following tables, the first value is average of PSNR, and the second one is standard deviation of PSNR is.

Sequence Target	SAD	ESAD	Annex L
Foreman 256k	31.39 (mean) 1.82 (std dev)	31.41 1.82	31.54 1.67
Stefan 768k	29.11 2.52	29.11 2.51	29.03 2.35
Coastguard 256k	28.44 0.93	28.46 0.95	28.44 0.92
Table 256k	32.73 2.38	32.73 2.46	32.75 2.38
Mobile 256k	24.92 0.7	24.93 0.71	24.89 0.51
News 256k	35.12 0.77	35.21 0.88	34.92 0.49

**Tab 4. Mean & Standard Deviation for PSNR Variation for Frame Level Rate**

**Control**

Sequence Target	SAD_MBL	ESAD_MBL	TBRC	RhoRC	TM5
Foreman 256k	31.18 (mean) 1.72 (std dev)	30.66 1.59	30.89 1.63	31.06 1.67	30.41 1.72
Stefan 768k	29.03 2.35	28.05 2.00	28.84 2.27	29.05 2.38	28.36 2.13
Coastguard 256k	28.2535 0.792541	27.7654 0.809037	27.9552 0.824705	28.2107 0.779721	27.5713 0.752996
Table 256k	32.0697 2.41904	32.2006 2.54736	32.089 2.55068	32.2198 2.58835	30.6201 1.61258
Mobile 256k	24.7361 0.59	24.6339 0.62	24.6322 0.51	24.8188 0.6	24.3458 0.55
News 256k	34.4155 0.848839	34.2426 0.771372	34.3993 0.82167	34.3803 1.20322	32.8881 0.437876

**Tab 5. Mean & Standard Deviation for PSNR Variation for Macroblock Level Rate Control**


To maintain quality as stable as possible is an important thing rate control needs to do. Hence, the average and standard deviation is shown here to examine stability of quality for each rate control algorithms.

In Tab 4 of frame level rate control, there is no very apparent difference among these three approaches. Annex L rate control has smallest standard deviation than other two methods for almost all sequences. Proposed first order R-D model w/ESAD (ESAD) has a little larger deviation than first order R-D model because the action of updating parameter for true SAD from motion estimator, but the performance is still acceptable.

In Tab 5 of macroblock level rate control, first order macroblock level R-D model (SAD\_MBL) has the average PSNR very close to Rho-domain RC, and so is the standard deviation. The result of table based rate control is a little lower than Rho-domain RC, but it is still acceptable due to the great reduction for hardware implementation. Test Model 5 has not good testing result, and even the worst one among all rate control algorithms.

### 5.4.3. Mean and Standard Deviation for Bits Variation

Following tables show the average and standard deviation for bits using different sequences at different target bitrates. In the entries of the following tables, the first value is average of frame bits, and the second one is standard deviation of frame bits is.



Sequence Target	SAD	ESAD	Annex L
Foreman	8540 (mean)	8524	8816
256k	3380 (std dev)	3497	2889
Stefan	25611	25575	25659
768k	9274	9461	11002
Coastguard	8535	8526	8585
256k	4358	4488	3820
Table	8530	8537	8725
256k	6193	5802	5801
Mobile	17069	17076	17387
256k	7823	7888	3881
News	8621	8634	8791
256k	5330	5358	4616

**Tab 6. Mean & Standard Deviation for Frame Bits Variation for Frame Level Rate Control**

Sequence Target	SAD_MBL	ESAD_MBL	TBRC	RhoRC	TM5
-----------------	---------	----------	------	-------	-----

Foreman 256k	8532 (mean) 1677 (std dev)	8534 2158	8531 1738	8597 2257	8508 2042
Stefan 768k	25598 4901	25606 7057	25598 5654	25662 5359	25394 4304
Coastguard 256k	8532.69 1466.67	8542.37 3679.13	8530.48 3543.09	8602.03 2096.9	8527.52 1698.53
Table 256k	8534.32 2048.13	8549.57 5535.53	8531.55 4450.33	8597.81 2521.69	8520.77 2351.26
Mobile 256k	17065.3 2932.95	17082.6 5177.85	17064 3037.37	17130.5 4222.8	17009.4 4026.14
News 256k	8617.28 2082.71	8639.52 4575.12	8615.95 4698.79	8681.23 3334.08	8662.45 2943.32

**Tab 7. Mean & Standard Deviation for Frame Bits Variation for Macroblock Level Rate Control**

Low standard deviation of frame bits is also required because large variation of frame will make the overall system unstable, even crash.

In the table of frame level rate control Tab 6, Annex L has lower standard deviation than other two methods. Standard deviation of proposed first order R-D model w/ESAD (ESAD) is similar to first order R-D model w/SAD (SAD), and a little lower than Annex L.

In the table of macroblock level rate control Tab 7, as the same reason mentioned before, first order R-D macroblock level model w/SAD (SAD\_MBL) has the lowest standard deviation of frame bits except to STEFAN sequence because of its concentration on controlling frame bits. Table based rate control performs very well in some sequences like FOREMAN and MOBILE, and in others, its performance is similar to Rho-domain RC. The proposed algorithm of first order R-D macroblock level model w/ESAD (ESAD\_MBL) is still not as stable as expectation, and some issues are needed to be solved.

## 6. Conclusion and Future Work

This thesis provides two different solutions to rate control implementation on SoC platforms. For hardware/software co-design approaches, out-of-loop rate control algorithm is proposed to reduce communication overhead between processor cores. In this scheme a simple approximation to SAD without motion estimation is derived and applied to frame-level and macroblock-level rate controls. Further, a novel table-lookup based rate control which is adequate for hardware implementation is proposed for pure hardware video encoder implementation.

### 6.1. Discussions

Several rate control algorithms are discussed in this paper. Two first order R-D model based algorithms which use true SAD as complexity measure, including frame level (SAD) and macroblock level (SAD\_MBL), are used as start points. Then, two out-of-loop rate control algorithms, for frame-level (ESAD) and macroblock-level (ESAD\_MBL) RC, are designed for co-design approach for video encoders. ESAD method has competitive performance compared to SAD method and MPEG-4 Annex L rate control. ESAD\_MBL method does not performance well compared to SAD\_MBL method, Rho-domain RC, and TM5 because of absence of actual coding bits and complexity for each MB.

Moreover, a new rate control (TBRC) is proposed to provide the ability to implement rate control algorithm into hardware circuit directly. There are many advantages to this method. Simply put, it has low complexity and is able to model sophisticated R-D curves that are common in real video sequences. Theoretically, it can do a better job than low-order curve fitting R-D modeling methods that are commonly used in video encoders.

### 6.2. Future Work

#### 6.2.1. Design Better Measure for Frame-Level or MB-Level Complexity

In order to estimate true complexity (sum of absolute difference used here), in section 3.2.1 a simple measure, which computes mean and deviation within specific region, is used. Obviously, if better measure could be used to predict true complexity, the performance will be promoted undoubtedly. For example, optical flow-based techniques [22] could be applied, and it may provide better estimation of true complexity.

### 6.2.2. Use More Sophisticated R-D Model

Proposed algorithms apply first order R-D model to solve issues on SoC platform. Even though it is better to use simple R-D model in MCU, if a sophisticated R-D model is used, the performance can be improved further.

## 7. Reference

- [1] ISO/IEC/JTC1/SC 29/WG 11 N4350, INFORMATION TECHNOLOGY – CODING OF AUDIO-VISUAL OBJECTS – Part 2: Visual.
- [2] ITU-T Rec. H.264 / ISO/IEC 11496-10, “Advanced Video Coding”, Final Committee Draft, Document JVT-E022, September 2002
- [3] K. Ramchandran, A. Ortega, and M. Vetterli, “Optimal Trellis-based Buffered Compression and Fast Approximations,” *IEEE Trans. Image Processing*, Vol. 3, pp. 26-40, Jan. 1994.
- [4] K. Ramchandran, A. Ortega, and M. Vetterli, “Bit allocation for dependent quantization with application to multiresolution and MPEG video coders,” *IEEE Trans. Image Processing*, Vol. 3, pp. 533-545, Sep. 1994.
- [5] J. Choi and D. Park, “A stable feedback control of the buffer state using the controlled lagrange multiplier method,” *IEEE Trans. Image Processing*, Vol. 3, pp. 546-558, Sep. 1994.
- [6] Wei Ding and Bede Liu, “Rate Control of MPEG Video Coding and Recording by Rate-Quantization Modeling,” *IEEE Trans. On Circuit and System for Video Tech.*, Vol. 6, no. 1, pp. 12-20, Feb. 1996.
- [7] ISO/IEC/JTC1/SC29/WG11, “Test Model 5,” JTC1/SC29/WG11 Coding of Moving Pictures and Associated Audio, MPEG 1994
- [8] Tihao Chiang and Ya-Qin Zhang, “A New Rate Control Scheme Using Quadratic Rate Distortion Model,” *IEEE Trans. On Circuit and System for Video Tech.*, Vol.7, pp.246-250, Feb. 1997.
- [9] H. M. Hang and J. J. Chen, “Source model for transform video coder and its application-Part I: Fundamental theory,” *IEEE Trans. On Circuit and System for Video Tech.*, Vol.7, pp.287-298, Apr. 1997.
- [10] Liang-Jin Lin and Antonio Ortega, “Bit-Rate Control Using Piecewise Approximated Rate-Distortion Characteristics,” *IEEE Trans. On Circuit and System for Video Tech.*, Vol.8, no.4, pp.446-459, Aug. 1998.
- [11] Jordi Ribas-Corbera and Shawmin Lei, “Rate Control in DCT Video Coding for Low-Delay Communications,” *IEEE Trans. On Circuit and System for Video Tech.*, Vol.9, no.1, pp.172-185, Feb. 1999.
- [12] Jordi Ribas-Corbera and Shawmin Lei, “A Frame-Layer Bit Allocation for H.263+,”

- IEEE Trans. On Circuit and System for Video Tech., Vol.10, no.7, pp.1154-1158, Oct. 2000.
- [13] J. Ribas-Corbera and D. L. Neuhoff, "Optimizing block size in motion-compensated video coding," J. Electron. Imaging, Vol.7, pp. 155-165, Jan. 1998
- [14] A. Gersho and R. M. Gray, Vector Quantization and Signal Compression. Norwell, MA: Kluwer Academic, 1992, Ch. 6.
- [15] E. Murata et al, "Study of relationship between quantizer step size and coded bit amount for video coding," IEICE Spring Conf., D-190, 1994.
- [16] ISO/IEC JTC 1/SC 29/WG 11, *14496-2:2002 Information Technology – Coding of Audio-Visual Objects – Part 2: Visual 3<sup>rd</sup> Ed., N5546*, Pattaya, Thailand, March 2003.
- [17] Hwangjun song and C. C. Jay Kuo, "Rate Control for Low-Bit-Rate Video via Variable-Encoding Frame Rates," IEEE Trans. On Circuit and System for Video Tech., Vol.11, no.4, pp.512-521, Aug. 2001.
- [18] Zhihai He and Sanjit K. Mitra, "Optimum Bit Allocation and Accurate Rate Control for Video Coding via  $\rho$ -Domain Source Modeling," IEEE Trans. On Circuit and System for Video Tech., Vol.12, no.10, pp.840-849, Oct. 2002.
- [19] H-C Fang, T-C Wang, Y-W Chang and L-G Chen, "Hardware Oriented Rate Control Algorithm and Implementation for Realtime Video Coding," ICASSP '03, Vol. 2, pp. 6-10 April 2003.
- [20] Byung Cheol Song and Kang Wook Chun, "A Virtual Frame Rate Control Algorithm for Efficient MPEG-2 Video Encoding," IEEE Trans. On Consumer Electronics, Vol.49, no.2, pp.460-465, May. 2003.
- [21] Ashish Jagmohan and Krishna Ratakonda, "MPEG-4 One-Pass VBR Rate Control for Digital Storage," IEEE Trans. On Circuit and System for Video Tech., Vol.13, no.5, pp.447-452, May. 2003.
- [22] Berthold K. P. Horn and Brian G. Schunk, "Determining Optical Flow," Artificial Intelligence, Vol. 17, no. 1-3, pp. 185-204, 1981.