國立交通大學

電子工程學系電子研究所

碩　士　論　文

次微瓦適 H.264/AVC 之極高編碼效能動態估測研究

On sub-*m*W R-D Optimized Motion Estimation

for Portable H.264/AVC

學生：史彥芃

指導教授：張添烜　教授

中華民國九十六年七月

次微瓦適 H.264/AVC 之極高編碼效能動態估測研究

On sub-*m*W R-D Optimized Motion Estimation for Portable H.264/AVC

研 究 生：史彥芑　　Student：Yen-Chi Shih

指導教授：張添烜　　Advisor：Dr. Tian-Sheuan Chang

國 立 交 通 大 學

電子工程學系電子研究所

碩 士 論 文

A Thesis

Submitted to Department of Electronics Engineering

Institute of Electronics

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electronics Engineering

July 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

次微瓦適 H.264/AVC 之極高編碼效能動態估測研究

研究生: 史彥芃　　指導教授: 張添烜 博士

國立交通大學

電子工程學系 電子研究所碩士班

摘　要

本論文旨在提出一次微瓦適編碼標準H.264/AVC之極高編碼效能動態估測研究。硬體設計以Artisan/TSMC 0.13μm (TSMC-CL013G-FSG) 製程實現核心面積 0.69mm^{2}於工作電壓 1.2V 及時脈頻率 20MHz 時提供平均功耗僅 0.6 微瓦 (動態功耗 0.2 微瓦) 之CIF 30-f/s 即時影像編碼能力。

複雜的編碼預測算術，H.264/AVC 大幅的增進習知影像編碼標準之編碼率-失真表現。肇因於動態估測為主要運算複雜度及計憶體存取頻寬，伴隨著編碼能力的提升，動態估測設計成為了影像編碼系統之實現瓶頸。近年之研究已提出為數眾多之即時、低成本動態估設計實現，然而此間之設計主要基於並非充分之設計指標及設計準則，過度以硬體設計成本為考量因此亦稱之為"硬體導向"設計。不適當的設計準則不但限制了設計易度且大幅犧牲系統效能表現。鑒於習知設計之謬誤，本論文致力提出於一次微瓦設計、具極高編碼效能之動態運算設計方法，該方法主要包含三種最先進之低能設計技巧：

一、預先的巨區塊複製檢測：基於似然率檢定能預先於動態估測前判斷目前編碼巨區塊是否當為區塊複製預測，可有效的移除編碼運算冗餘進而節省操作功耗。

二、適應性之搜尋範圍預測：使用適應性搜尋範圍之搜尋中心偏移區塊比對機制能大幅的增進習知技術之編碼算術效能及動態補償之有效性。

三、內部記憶體最佳化設計：發明一基於製程之高抽象層級設計方法最小化記憶體存取電流修正習知設計之謬誤，大幅的降低資料緩衝處理之存取能耗。

此外，低切換率之差絕對值邏輯設計能降低過去設計 50% 之邏輯面積與能量損耗，有效的減輕全算術平行化之硬體設計成本。為了進一步降低內部記憶體定址功耗，吾人以數學證明一最短距離編碼方法能不需額外設計成本而有效降低定址之邏輯切換率。基於位移向量編碼量主要影響影像編碼之位元數目，本論文亦提出一疊加器架構之位移成本算術方法，有效而低成本的進一步提升最多 15% 之編碼效能。

藉由本文之設計方法，相較習知技術，吾人之設計已帶來極重要之功耗與編碼率-失真表現效益，在僅 198.8 微瓦之動態功耗與相同編碼失真條件下，最多將提升編碼效能達 50% 以上。簡言之，本文所提出之設計準則、演算法、及架構相較於習知設計能大幅的增進設計指標與系統效能，包含適於先進編碼實現之編碼效率、邏輯面積與功率損耗。

On sub-*m*W R-D Optimized Motion Estimation for Portable H.264/AVC

Student: Yen-Chi Shih Advisor: Dr. Tian-Sheuan Chang

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

Abstract

This thesis presents an exceptional motion estimator design for portable rate-distortion optimized H.264. The proposed design targets in processing capability of real-time CIF 30-f/s video with core area 0.69*m*m^2 and 0.6*m*W (dynamic 0.2*m*W) power dissipation when worked at 1.2V and 20MHz under Artisan/TSMC 0.13μm process (TSMC-CL013G-FSG).

Due to advanced prediction in complex arithmetic, H.264/AVC achieves significant rate distortion (R-D) improvement than prior standards. While the motion estimation predominates computation and memory access complexity, as increasing the capability, it becomes the most crucial component in the video codec. In recent, numerous studies were promoted to realize the cost-efficiency, real-time motion estimation. However, most of works are structured on the insufficient metrics and criteria, and so called *hardware-oriented*. Such improper limitations result in inherent restriction in design flexibility and significantly degrade on system performance. This therefore motivates us to demonstrate an elegant design methodology of achieving a sub-*m*W low-cost/high-performance motion computing. The proposed methodology includes three upmost power-efficiency techniques:

1. Early detecting on macroblock-skipping: novel likelihood ratio test (LRT) method effectively detects whether the MB should be SKIP coding prior to motion estimation, which efficiently exploits the computational redundancy and thus saves the power,
2. Adaptive prediction in search boundaries: biased block-matching scheme in use of dynamic boundary significantly improves arithmetic efficiency as well as motion compensated fidelity of prior arts, and
3. Perspective in optimizing memory structure: minimizing the access current using high level technology-dependent analysis rectifies assumed design fallacy, hence greatly suppressing power consumption in reference data buffering.

In addition, the low switching AD (absolute difference) logic, using half the area and the power, is presented to successively eliminate full-parallelism design cost. To further minimize

the power of address switching, a shortest distance bus coding is mathematically proven cost-free and effectiveness. Since bit-stream size is largely affected by motion vector difference (MVD) in low rate coding, an accumulator-structured logic is then exhibited for motion vector cost arithmetic, which sufficiently further improves at most 15% of coding efficiency.

Applied with the concluded methodology, our work supports dramatically power and rate distortion benefits over pervious studies, which has been summarized in average of 198.8 μW of dynamic power dissipation and more than half the bitrate improvement of equal frame PSNR compared to prior full searches. In brief, the present criteria, algorithms, and structures significantly improve design metrics and provide essential superiority than traditional designs, in terms of coding efficiency, silicon area and power consumption for advanced codec implementation.

誌　謝

理想如引路燈塔。沒有理想, 沒有可靠方向; 沒有方向, 又何來人生。

— *Leo Tolstoy*

承蒙指導教授張添烜博士的耐心與殷切指導得以讓本論文順利完成。誠摯的感謝恩師黃永達教授與李宇旼博士, 老師們的學者風範及治學態度淺移默化學生對於探究學問的嚴謹與執著。感謝陳永昌教授與李鎮宜教授, 於百忙之中蒞臨口試指導, 畢使本論文更臻完善; 以及感謝助理李清音小姐在口試期間的熱心協助與提醒。特別感激母親曾詒翔女生不辭辛勞與耐心包容, 母親的諄諄教誨與默默支持, 是學生追求理想的主要動力, 感謝母親; 同時感謝身邊親友建議與鼓勵, 亦令學生獲益匪淺。

最後, 僅將此論文獻給最敬愛的父親　史君先生, 所有的榮耀與貢獻皆歸於您。

史彥芪
謹識于　新竹
九十六年七月

Contents

List of Tables

List of Figures

Chapter 1

Introduction

H.264 Advanced Video Coding (AVC) is an emerging next generation video coding standard which was approved by Joint Video Team (JVT) of ISO/IEC and ITU-T in 2003 [1]. The purpose of the AVC aims to cover all different bit-rate applications, for instance, video telecommunication, video photography, and broadcast-quality digital television.The framework of AVC is based on the hybrid motion-compensated structure, which had been proven the most successful class of video compression over past two decades.

Based on the hybrid coding scheme, several improved key techniques in AVC significantly increase the compression efficiency for video coding. Some notable key advance over the pervious standards are *enhanced SKIP mode*, *variable block size prediction* as well as *rate-distortion optimization* (RDO). SKIP coding is a simplest temporal redundancy reduction technique, that has been used in the first digital video coding standard ITU-T Rec. H.120 [2] since 80's. The enhanced SKIP in AVC replenishes depicted frame according to predicted displacement instead of original. SKIP coding with displaced motion is capable of further reducing the frame difference between depicted picture and imaging plan, and thus improves the R-D quality. Often, segmenting and predicting an area of coded block individually can result in a reduction in the amount of information that needs to be sent as a DFD. In AVC, variable block size predicting segments coded block into maximum 41 subdivisions. This flexibility provides diverse degree of DFD on motion-compensated prediction, and reasonably improves the coding efficiency.

For all permitted coding modes, R-D optimization in AVC using Lagrangian method makes the best mode choice and parameter settings that substantially leads rate-distortion improvement. In practical, RDO can code with the same frame distortion as RD optimization off using 80% or less than 80% bit-stream size whenever coder encodes an motion picture. With involved modern coding techniques, JVT H.264/AVC has much better R-D quality and therefore suits various consumer devices on video coding. It is therefore these high-efficient coding features that greatly emerge implementation obstructions, such as hardware complexity, compression quality and power dissipation. The summary of oper-

Table 1.1: Instruction profiling of an H.264/AVC baseline profile encoder at CIF (352×288) 30fps, 5 reference frames, search range $[-16 : 15]$, QP=20

Functions	Arithmetic		Controlling		Memory Access		
	MIPS	%	MIPS	%	MIPS	Mbyte/Sec	%
Integer-Pel ME	95,491.9	78.3	21,915.1	55.4	116,830.8	365,380.7	77.5
Fractional-Pel ME	21,396.6	17.6	14,093.2	35.6	30,084.9	85,045.7	18.0
Interpolation	558.0	0.5	586.6	1.5	729.7	1,067.6	0.2
Mode Decision	674.6	0.6	431.4	1.1	880.7	2,642.6	0.6
Intra Prediction	538.0	0.4	288.2	0.7	585.8	2,141.8	0.5
VLC	35.4	0.0	36.8	0.1	44.2	154.9	0.0
T&Q	3,223.9	7.6	2,178.6	5.5	4,269.0	14,753.4	3.1
Deblocking	29.5	0.0	47.4	0.1	44.2	112.6	0.0
Total	121,948.1	100.0	39,577.3	100.0	153,469.3	471,299.3	100.0

MIPS: Million Instructions per Second, data from Chen *et al.,* TCSVT'06 [3]

ating MIPS (Million Instructions Per Second) of baseline profile AVC at CIF 30fps video coding is illustrated in Table 1.1 [3]. Clearly, most computation-hungry functions are due to (variable block size) motion estimation.

Motion estimation effectively exploits temporal redundancy in video compression by estimating of texture movement relative to the depicted frame. In a video codec, the motion estimation, significantly impacting on coded video quality and coded bit-stream size, demands over 70% of complexity and memory access requirements; therefore becomes a most crucial component for potable video coding application.

For past decades, numerous algorithms and corresponded hardware designs have been proposed to realize the real-time motion estimation for portable application. Unfortunately, parts of design criterion and limitations inherently restrict design flexibility and thus lead the moderate degree of system performance degeneration. These motivate us to develop a novel qualified methodology of sub-mW low-cost and high performance ME approach.

1.1 Motivation

Due to high complex decision process and high arithmetic tasks, AVC encounters the high computing and high data traffic challenges. To meet the stringent requirements on low power and high performance for the consumer electronics market, there is a clear need for optimized AVC VLSI implementation.

In recent years, the demands for extremely low power and high-resolution mobile video coder are rapidly growing. Examples for such commercial devices are handset cellphone,

digital compare (DC) and digital video camcorder (DV). Designs capable of real-time encoding for portable application are needed to support the creation of such digital video contents. An analysis of an H.264 baseline profile performed on CIF video coding already shown in Table 1.1 that H.264/AVC video encoding requires giga-operations per second (GOPS) and thousand-mega-byte scale memory access per second. To support the encoding on such resource demanding, the computation complexity is extremely high.

The motion estimation is the focus of our work to achieve a high performance H.264/AVC design, since it is the most crucial task in H.264. Because the complexity of excellent full-search algorithm in motion estimation is extremely high, numerous hardware design studies have been proposed to reduce the computational complexity of full-search block-matching algorithm (BMA). However, most to these implementations are structured on mal-assumptions and design criterion. The improper limitations result in inherent restriction in design flexibility and lead the significant degree in the system performance degeneration.

Besides, multimedia potable devices that rely on batteries for energy support becomes more and more popular these days. Cellphones equipped with digital camera capable of transmitting real-time video are a major trend for cellphone development. Other applications like digital video camcorder (DV) trends to record higher resolution movies. Unfortunately, the advance in battery technology does not compel with the growth of power-hungry fancy video applications. In order to maintain acceptable operating hours with the same battery capacity, low power design for AVC becomes a most important issue. Such design obstructions and requirements lead to our motivation of exploring cost-efficient low power/high R-D performance algorithms and the corresponding VLSI architectures.

1.2 Overview of Video Coding

Motion video data consists essentially of a time-ordered sequence of pictures, and cameras typically generate approximately 24, 25 or 30 frames per second. This results in a large amount of data that demands the use of compression. For example, assume that the video sequence is transmitted at the frame-rate 15 pictures/s and each picture has a low "QCIF" (quarter-common-intermediate-format) resolution (*i.e.*, 176×144 samples) for that each sample is digitally represented with 8 bits. For color pictures, three color component samples are necessary to represent a sufficient color space for each pixel. In order to transmit even this relatively low-fidelity sequence of pictures, the raw source data rate is still more than 9 Mbits/s. However, the low-cost transmission channels often operate at much lower data rates so that the data rate of the video signal needs to be further compressed. In order to eliminate the data redundancy and to facilitate transmission or

storage, many video coding standards have been regulated in recent year, such as ISO/IEC MPEG-1/2/4, CCITT H.261/ITU-T H.263 and the ITU-T/ISO JVT H.264/AVC.

The statistical analysis indicates that video scenes have strong correlation both between successive frames and within the picture themselves. The strong correlation between consecutive frames is called the *temporal correlation* (also called the *temporal redundancy*) and the strong correlation within the picture is called the *spatial correlation* (also called the *spatial redundancy*). Based on the two correlation classifications, the video compression is developed separately for the individual processing domain in common video coding technology. In the video signal, the data amount reduction achieved mainly by eliminating the spatial correlation, temporal correlation and the inter-symbol redundancy. In blow subsections, we briefly describe the fundamental concepts of the spatial, temporal and inter-symbol correlation in video coding.

Spatial Correlation

The spatial correlation dedicates the relationship between the adjacent samples (image pixels) in same frame. Figure 1.1 demonstrates an enlarged video frame example for spatial relationship. Obviously, the sample characteristic is usually similar to the neighboring pixels. In other words, the pixels at many locations can be predicted from the surrounding. The total entropy for data transmission can be greatly reduced via some useful prediction methods. One well-known technique is DPCM (differential pulse code modulation), which is wide used in many communication applications.

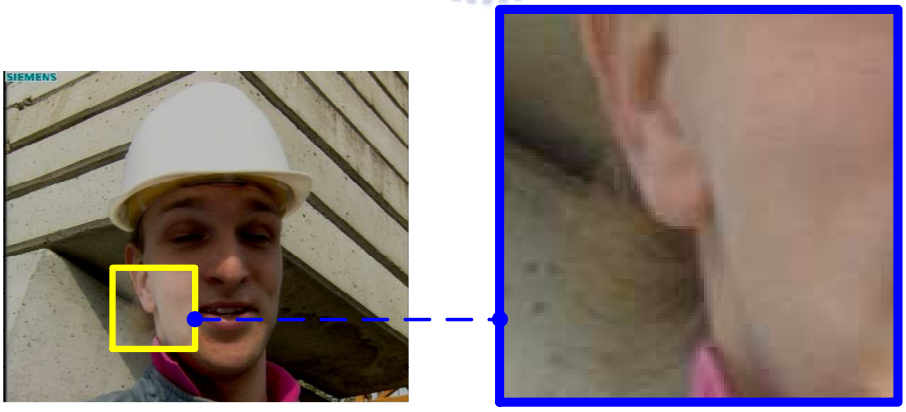


Figure 1.1: Illustrations of the spatial correlation (foreman-CIF)

Temporal Correlation

The temporal redundancy dedicates the high relationship between the successive frames. In order to generate the moving pictures, the sampling for the real world scene samples not only the spatial elements but also the scene over a period of time, and we define this

period as the frame rate. Higher frame rate, more smooth object moving can be seen. For the human visual system, the frame rates should be higher than at last 15 frames per second to result in the continuous and smooth moving scene. Consequently, the difference between the successive frames is very small in case of the higher frame rate presents.

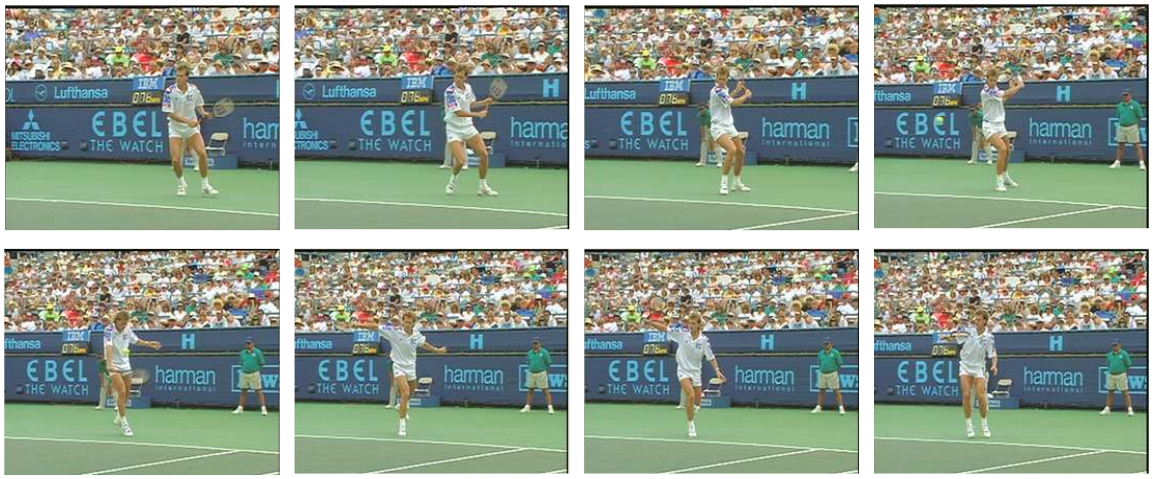


Figure 1.2: Illustrations of the temporal correlation (stefan-CIF)

Figure 1.2 shows the snapshots of the CIF test sequence "stefan". Each snapshot separates 5 frames. It obviously shows that texture difference between any consecutive picture is relatively small.

Statistical Correlation

An information-carrying signal always contains redundancy, which means that it exists a more efficient codeword in information representing. For example, characters within a text message occur with varying frequencies: in English, the letters E, T and A occur more often than the letters Q, Z and X. This makes it possible to *translate* message by representing frequently occurring characters with shorter codewords and infrequently occurring characters with longer codewords. This coding method using variable length codeword such as Huffman Code and Golobm Code is thus called variable length coding (VLC). Beside VLC, another technique, arithmetic coding, further exploits statistical correlation and much closely approaches entropy bound. In other word, it means arithmetic coding presents more compact information compression. Both VLC and arithmetic, frequently refer as entropy coding in video coding terminology.

1.2.1 Video Coding Standards

Video compression techniques have played an important role in multimedia communication field. After several decades' development, ISO and ITU organizations have regulated

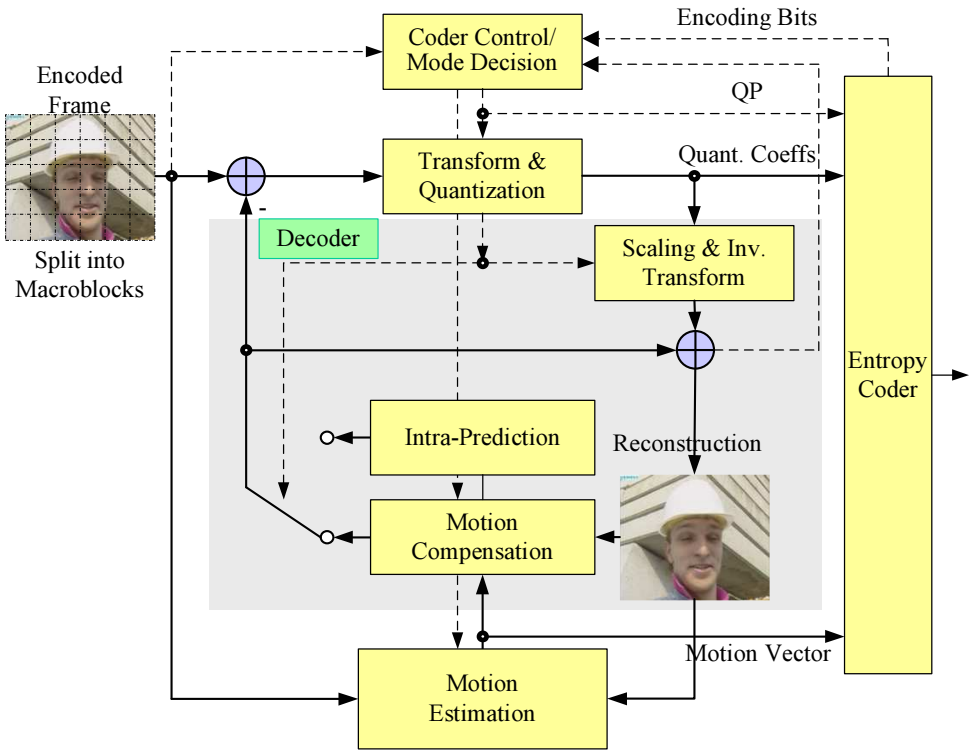


Figure 1.3: Advanced video coder block diagram and its coding flow

MPEG series and H.26× series video coding standards to aim at the video compression. On typical application demands, it has different video coding standards to satisfy the specific purposes. For instance, H.261 and H.263 are presented by ITU-T (the International Telecommunications Union — Telecommunications Standardization Sector, then called the CCITT) for low-rate video phone or video conferencing application, and MPEG-1/2/4 are presented by ISO (international Organization for Standardization) for high bit-rate video entertainment application. MPEG-4 Part-10 H.264/AVC (Advanced Video Coding) is introduced by the Joint Video Team (JVT: The Joint Team of ISO/ICE and ITU-T) which aims to achieve different bit-rate applications and suitable for different Internet transmission environments.

1.2.2 Hybrid Video Coding Structure

All video coding standards aforementationed are based on the hybrid coding architecture. The naming of *hybrid* is due to the picture reconstruction as a combination of motion-handling and picture-coding techniques, and the term codec is used to refer to both the coder and decoder of a video compression system. Figure 1.3 shows a coder using hybrid structure. The design and operation of such hybrid scheme should involve the optimization of a number of decisions, including

1. Properly replaces each coded block (*e.g.*, MB) of the picture with completely INTRA-frame content,

2. If not replacing an MB with new INTRA content
   (a) Properly segments each MB into sub-areas,
   (b) Performs sub-area motion estimation; *i.e.*, searches the spatial shifting displacement to use for INTER-picture predictive coding,
   (c) Properly selects INTER-picture coding method (SKIP or prediction mode) for coed block according to the R-D quality of given coding method.
   (d) Performs DFD coding if prediction mode is applied; *i.e.*, coding the INTER residuals as refinement of the INTER prediction,
3. Performs SKIP coding if SKIP is applied, *i.e.*, repeats depicted frame as the replacement content on indicated block, and
4. If replacing an area with new INTRA content, coding the INTRA residuals as the replacement content.

An INTRA-frame coding is similarly using an image-coding syntax as still image coding JPEG. Every segmented block using JPEG coding are transformed by a discrete cosine transform (DCT), and the DCT coefficients are then quantized and transmitted using entropy coding, such as VLC or arithmetic coding. This coding scheme is referred as INTRA-frame coding, since the picture is coded without referring to other pictures in the video sequence. However, improved compression performance can be attained by taking advantage of the large amount of temporal redundancy in video content. Such techniques that exploit the relationship of temporal correlation are referred to as INTER-frame coding. Usually, much of the depicted scene essentially repeats in picture after picture without any significant change. It should be obvious then that the video can be represented more efficiently by coding only the changes in the video content, rather than coding each entire picture repeatedly.

A simplest way for INTER-frame coding is CR method, which just repeats predictive picture on indicated area, and also refers to SKIP mode. However, INTER coding without residuals compensation has a significant shortcoming: inability of refining an approximation of picture content. Often the content of an area of a prior picture can be a good approximation of the new picture, needing only a minor alteration to become a better resemblance. Hence, adding the method of *motion compensation*, in which a refining *frame difference* approximation can be attended, results in a further improvement of compression performance.

Most changes in video content are typically due to the motion of objects in the depicted scene relative to the imaging plane. A small amount of motion may accordingly result in a large difference in the values of the pixels in a picture area. In general, displacing an area of the predicted picture by a few pixels in spatial location can reflect a significant reduction on the amount of information that needs to be sent as DFD or a frame difference approximation. This use of spatial displacement to form a reconstruction is known as

motion compensation (MC), and the encoder's search for the best spatial displacement is thus known as motion estimation (ME). The coding of the resulting difference, or the residuals, for the frame refinement of the MCP is referred as displaced frame difference (DFD) coding.

Beside inter/intra prediction, the transform coding transforms the residuals into frequency domain due to frequency perceptivity of human visual system. The transformed coefficients followed with quantization further compact insignificant information. An Entropy coder, who eliminates statistical redundancy, is then adopted to compress prediction residuals into streaming and to count required bit rate after prediction and transform coding. In AVC, coder optimizes multi-block size prediction and hence significantly improves R-D quality. Therefore, decision best prediction based on the coded information, control/mode decision functional block becomes most important key in a coder. An optimized AVC can largely increase bit-rate efficiency and can mitigate channel overhead.

In summary, a video coder using hybrid structure which is combined with INTRA and INTER frame coding techniques, efficiently eliminates spatial and temporal redundancy and thus compresses the video sequence with slight objective and subjective quality degradation.

1.3 Problem Briefs and Thesis Organization

Intensive arithmetic in motion estimation predominates computation and memory access complexity in a video codec. As increasing the processing capability, it becomes the most crucial obstruction to realize a real-time portable coder. To demonstrate an exceptional motion estimation for portable H.264/AVC applications, the thesis organization of the proposed design methodology is concluded as follows.

Chapter 2 To effectively eliminate complexity, after an induction to advanced video coding, chapter 2 reveals a macroblock-skipping method in use of likelihood ratio test, which ingeniously exploits the computation dependence and releases the complexity burden. Starting from the mathematics on statistic characteristics of Lagrangian optimization, a false rate constrained macroblock-skipping detection is proposed at maximizing the probability of detection by a graph-based approach. Based on the exploration, the proposed detection method moderately detects whether the current macroblock should be SKIP coding or not prior to motion estimation, which efficiently eliminates the computational redundancy and thus saves the power.

Chapter 3 Motion estimation with exhaustive search provides a more superior degree of rate-distortion performance; however, it demands huge amount of system complexity and power dissipation. Motion estimation is usually found consuming over half the power in a

video codec. To release the power demand, most implementation are strictly exploited the characteristics of hardware acts, for instance, the modularity and the regularity. These techniques primarily concerns about design metrics of data-reuse efficiency, memory bandwidth, and arithmetic utilization, thus so-called *hardware-oriented algorithms*.

However, inapposite tradeoffs on these metrics leads to power-insufficient assessment as well as poor prediction fidelity. To resolve metric insufficiency problem at high level of design abstraction as well as the prediction reliability issue existing in pervious studies, a robust fast scheme with dynamic boundary decision is then presented to significantly improve the R-D quality, where the computational burden and corresponded power dissipation are both substantially preserved. In addition, two wide-used power-efficiency techniques, date depth truncation, and prediction simplification, well be examined in the chapter 3.

Chapter 4 Based on system/algorithm development in chapter 3, the focus of chapter 4 will advance in design abstraction of power-efficiency architectures. To deal with most arithmetic intensive SAD computation, this chapter first theoretically investigates the array processor architecture by well-known graphic-based design methodology. Then a specialized arithmetic is derived which significantly eliminates switching power and area cost of absolute difference processing element (PE). In section 4.3, we further exhibits a novel memory structuring methodology using high-level technology-dependent power analysis, which minimize the access power due to internal memory buffering.To further minimize the power dissipation from address switching, we mathematically develop the shortest distance bus coding scheme in section 4.4. An ultra cost-efficiency logic structure is presented to facility motion cost computation, which effectively improves MCP fidelity and rate-distortion metric in low rate motion estimation in the end of this chapter.

Chapter 5 Chapter 5 details design specification and implementation flow, including design characteristic, I/O, timing, and architecture specifications as well as implementation procedures and physical chip specifications.

Chapter 6 Experimental assessments in rate-distortion and power measurement proceeds to be illustrated in this chapter. Two compared objects, JM full search and traditional hardware-oriented full search, were selected to assess the system performance in R-D and power. By simulation, it has shown that low-power design based on our studies is capable of using *half the coding bitrate* with better reconstructed quality and saves more than *90% of dynamic power* over prior most advanced studies.

Chapter 7 At the end of the thesis, concluding remarks are addressed in chapter 7.

Chapter 2

Macroblock-skipping Detection

To effectively eliminate codec complexity prior to arithmetic intensive processes, this chapter explores an low-cost macroblock (MB) skipping detection method using likelihood ratio test (LRT). Starting from the mathematics on statistic characteristics of Lagrangian rate-distortion optimization (RDO), a false rate constrained MB-skipping detection is proposed at maximizing the probability of detection by a graph-based approach.

Based on the graphic exploration, the proposed method efficiently eliminates the computational redundancy without sacrificing rate-distortion performance. Experiments conclude that the 17%–87% probability of detection is archived at false rate 1% relative to motion activity.

This chapter is organized into six parts. Section 2.1 briefs the motivations, and section 2.2 introduces related works on macroblock-skipping detection. A novel graph-based LRT approach is then presented in section 2.3, followed with its logic implementation. Performance evaluation is addressed in section 2.5. Finally, section 2.6 concludes this chapter.

2.1 Background and Motivation

Significant rate-distortion (R-D) improvement is achieved at the expense of advanced prediction scheme in ITU H.264/AVC [4]. In portable application, image data are most encoded as SKIP coding due to limited channel capacity. Since decision the optimal prediction of each macroblock requires a series of transformations and reconstructions, the codec fairly wastes computational resource and power for encoding these macroblocks. Hence, AVC has the ability to precisely detect whether an MB should be skipped or not considerably decreasing computation redundancy and calculating power.

To detect macroblock-skipping in advance, there are some methods that have been proposed. The method according to the magnitude of motion vector (MV) and sum-of-absolutes-differences (SADs) was revealed in [5,6]. Two early detection methods of

estimation of Lagrangian cost prior to motion estimation was stated in [7,8]. Another similar cost estimation method was proposed based on the maximum a posteriori probability (MAP) hypothesis testing [9]. However, these detection methods suffered from either restricted detection sufficiency or the infeasibility of probabilistic characterization. Since coding statistics may vary diversely, generalizing a probabilistic model degrades the fidelity of implementation. Besides, the restricted sufficiency lowers probability of detection and leads ordinary complexity reduction.

In fact, SKIP displacement utilizes spatial motion correlation, and only depends on neighboring vectors. It implies that the residuals can be obtained simultaneously with vector derivation, and a coder may decide the macroblock skipped or not using SKIP coding residuals. By this concept, we first analyze some properties and criteria of macroblock-skipping detection based on the rate-distortion optimization (RDO) framework. To maximize the detection probability subject to an acceptable false rate, a likelihood ratio test (LRT) is then proposed by a graph approach. An adaptive threshold is also presented to constrain the false rate in different encoding scenarios according to the "CODE-detected" characteristic.

The proposed method eludes the difficulty of probabilistic parameterization. While the direct test on SKIP coding increases the probability of detection, the computation burden can thus be efficiently saved. A coder with proposed algorithm conditionally eliminates the encoding complexity in terms of motion activity, without excessively degenerating R-D performance.

For hardware implementation, we have mapped the proposed algorithm to n low-cost accumulator-based structure, which is fabricated by TSMC-CL013G process with 1K gates/7.5μW. Our low-complexity implement is easily incooperated into a coder to efficiently reduce computational power during frame encoding.

2.2 Problem Formulation and SKIP Detections

Variable size prediction has greatly advanced in compression efficiency. The approved predictions of H.264 INTER P-frame are listed as follows,

$$S_p = \left\{ \begin{array}{c} \text{INTRAs, SKIP, } 16\times16 \\ 16\times8,\ 8\times16,\ \text{P}8\times8 \end{array} \right\} \tag{2.1}$$

and the Lagrangian rate-distortion cost in test model JM (Joint Model) [10] is described as Eq. (2.2)[1].

$$J(\mathbf{s},\mathbf{c},\text{MODE}|\lambda) = \text{SSD}(\mathbf{s},\mathbf{c},\text{MODE}) + \lambda \cdot R(\mathbf{s},\mathbf{c},\text{MODE}) \tag{2.2}$$

[1]Formally named Lagrangian cost, refer to 3.2.3 (pp. 31)

where SSD is predicted distortion using sum-of-squared difference measurement between the encoded MB $\mathbf{s}$ and its reconstruction $\mathbf{c}$ associated with the prediction MODE. The Lagrange multiplier λ, related with quantization parameter (QP) considering all MBs, is chosen for mode decision of MB prediction. Function R maps the bits required of present mode. Mode-selection algorithm using Eq. (2.3) optimizes rate-distortion by assuming all macrobloks encoding independently, therefore referred to *constrained RDO* [11].

$$\text{MODE}^* = \arg\min_{\text{MODE}} J(\mathbf{s}, \mathbf{c}, \text{MODE}|\lambda) \tag{2.3}$$

In portable application, such as videophone, video conferencing, the encoded bitrate is strictly constrained to meet channel capacity. This results in coding occurrence as SKIP selected much higher than others. Since the rate associated with a skipped coding is effectively zero, obviously, that SKIP is chosen implies

$$\text{SSD}(\mathbf{s}, \mathbf{p}, \text{SKIP}) \leq \text{SSD}(\mathbf{s}, \mathbf{c}, \text{CODE}) \tag{2.4}$$

where $\mathbf{p}$ indicates the predictions of replaced MB, and CODE, presents available mode opposed to SKIP, whose residuals remain to refine frame approximation. For convenience, we rewrite SSD of SKIP coding as Eq. (2.5),

$$\text{SSD}(\mathbf{s}, \mathbf{p}, \text{SKIP}) = \|\mathbf{r}_Y\|^2 + \|\mathbf{r}_U\|^2 + \|\mathbf{r}_V\|^2 \tag{2.5}$$

and SSD for other predictions as follows.

$$\text{SSD}(\mathbf{s}, \mathbf{c}, \text{CODE}) = \|\mathbf{r}_Y - \mathbf{r}'_Y\|^2 + \|\mathbf{r}_U - \mathbf{r}'_U\|^2 + \|\mathbf{r}_V - \mathbf{r}'_V\|^2 \tag{2.6}$$

SKIP coding residuals, $\mathbf{r} = \mathbf{s} - \mathbf{p}$, present the differences between the MB and it prediction, suffixes Y, U, and V denote luminance and chrominances respectively, and the operator $\|\cdot\|$, a generalized norm of an" N-by-M dimensional vector, is defined as

$$\|\mathbf{A}\| = \sqrt{tr(\mathbf{A}^t\mathbf{A})} \tag{2.7}$$

where $\mathbf{A}$ is an N-by-M dimensional vector, $tr(\cdot)$ indicates trace operation, and t denotes transpose. Eq. (2.6) presents the squared error of the coded-MB reconstruction, where vector $\mathbf{r}'$ presents the reconstruction associated with residual $\mathbf{r}$. In fact, a zero-coefficient block is followed by a zero reconstruction. Accordingly, a further case for macroblock-skipping is that the estimated motion equals to the predicted of SKIP, and transformed predictions are all-zero when R-D cost of 16×16 mode is smaller than other modes excluding SKIP, *i.e.*, CODE. In [5,6] as well as the fast mode decision algorithm in reference JM [12], these methods are presented based on the sufficient condition. However, constrained efficiency due to strict detection condition degrades the necessity. Similarly, the algorithms in [13,14] are intrinsically identical. In addition, requirement of motion estimation eliminates the feasibility as well.

In conventional video coder, since transformation such as Discrete-Cosine-Transform (DCT) is considerably energy conserved, we may assume that the mean-squared-error (MSE) after reconstruction of an all-zero coefficient block is smaller than the non-zero, as following

$$E\{\|\mathbf{X}\|^2|\mathbf{T}(\mathbf{X})=\mathbf{0}\} \leq E\{\|\mathbf{X}-\mathbf{X}'\|^2|\mathbf{T}(\mathbf{X})\neq\mathbf{0}\} \tag{2.8}$$

where $\mathbf{T}$ presents the combined process of transformation and quantization in a coder. Table 2.1 compares the simulated reconstruction MSE relationship of H.264 baseline coder between all-zero coefficient block (All-zero) and non-all-zero (Non-zero) in different quantization step sizes (Q_{step}). The stimuli are uncorrelated 4×4 Gaussian random vectors with zero mean, standard deviation (σ) 15. While the energy of residual is relatively inneglectable as smaller step size, the reconstructed MSE of all-zero coefficients reaches saturation when QP=30.

Table 2.1: MSE simulation for residuals reconstruction (uncorrelated Gaussian with zero-mean, standard deviation 15)

	QP	30	36	42
	Q_{step}	20	40	80
MSE	All-zero	711.3	1844.7	3064.0
	Non-zero	741.3	2734.9	5843.8
Relative (%)		95.94%	67.45%	52.43%

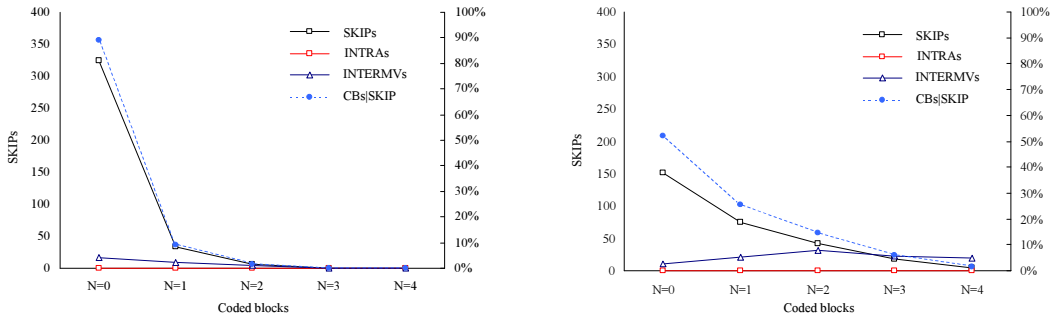
By the assumption of Eq. (2.8), the MSE relationship between the SKIP and CODE can be easily derived as

$$E\{\|\mathbf{s}-\mathbf{p}\|^2|\mathbf{T}(\mathbf{r}_n)=\mathbf{0},\forall n\} \leq E\{\|\mathbf{s}-\mathbf{c}\|^2|\text{CODE}\} \tag{2.9}$$

which implies that the expected R-D cost holds the inequality

$$E\{J(\mathbf{s},\mathbf{p},\text{SKIP}|\lambda)|\mathbf{T}(\mathbf{r}_n)=\mathbf{0},\forall n\} \leq E\{J(\mathbf{s},\mathbf{c},\text{MODE}|\lambda)|\text{CODE}\} \tag{2.10}$$

The inequality Eq. (2.10) has been applied in many fast inter mode decision algorithms as SKIP prediction in referenced JM. Two experiments of sufficiency and necessity of all-zero block test are shown as Figure 2.1. Figure 2.1(a) shows the average predictions in a coded frame of the sequence "hall_monitor", where "SKIP" indicates the average number of SKIP present in a frame, "INTRA"presents prediction using {INTRA}, and "INTERMVs" is the rest of modes. The "CBs|SKIP" presents the necessity of SKIP detection in term of conditional probability of CBs given that encoded MB is skipped. Note that a block is called CB (coded block) if its partitions are non all-zero coefficients. The necessity test of another sequence foreman is shown as Figure 2.1(b). Although the



(a) Test sequence: hall_monitor (b) Test sequence: foreman

Figure 2.1: Sufficiency and necessity tests — MB-skipping using coded-blocks ($QP = 36$, CIF@30fps, Baseline+RDO)

necessity of the all-zero block testing in foreman is relatively obscure due to the variety of residuals energy, the all-zero block or, more general, coded block testing is effective enough for SKIP detection.

Nevertheless, The all-zero block testing has less worth of use due to the requirement and dependence of transformation and quantization. To avoid transform coding, *i.e.*, integrated DCT and quantization, all-zero block detection algorithms such as [15,16], may be used to early detect whether the residuals are all-zero quantized or not. These indirect detection methods, however, restrict the necessity of detecting. A severe degeneration of detection probability may be introduced and the burden of complexity won't be remarkably released.

2.3 Proposed Detection Method

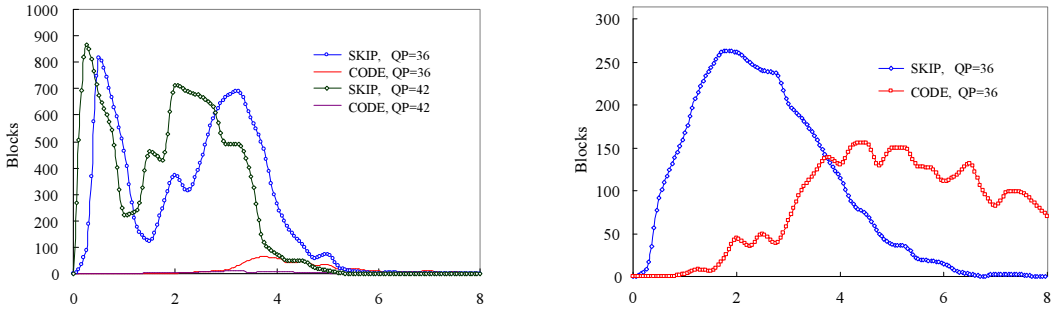
2.3.1 Theoretical Analyses

While an indirect approach restricts the necessity on detection, the direct method, such as hypothesis testing, may considerably extend the probability of detection.

In a LRT problem, given a set of observations, a decision has to be made regarding the source of the observations. A general form of an LRT is as followed

$$\Lambda(\mathbf{z}) \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{2.11}$$

where $\Lambda(\mathbf{z})$ names likelihood ratio function, related with the observations $\mathbf{z}$ from the observation space $\{\mathbf{Z}\}$. The test consists of comparing the ratio with a threshold and is therefore referred as a likelihood ratio test. In general, we would like to make false rate, P_F, as small as possible and probability of detection, P_D, as large as possible, where P_F



(a) Test sequence: mother_daughter (b) Test sequence: stefan

Figure 2.2: Hypothesis occurrence of SKIP and CODE against $\max(\mathrm{SAD}_{4\times4})/Q_{step}$ (CIF@30fps, Baseline+RDO)

and P_D are defined as Eq. (2.12) and Eq. (2.13), respectively.

$$P_F = P_r\{\hat{\mathrm{MODE}} = \mathrm{SKIP} \wedge \mathrm{MODE}^* \in \{\mathrm{CODE}\}\} \tag{2.12}$$

$$P_D = P_r\{\hat{\mathrm{MODE}} = \mathrm{SKIP} | \mathrm{MODE}^* = \mathrm{SKIP}\} \tag{2.13}$$

However, these are usually conflicting objectives. Instead of the opposition, we are therefore targeting on minimizing computational redundancy subject to an acceptable degree of encoding quality drop. A false rate α of P_F is thus predetermined, while simultaneously maximize the probability of detection P_D, and can be read as follows

$$\max\ P_D \text{ subject to } P_F = \alpha \tag{2.14}$$

The optimized LRT based on the Eq. (2.14), maximizing P_D subjtect to a given value P_F, can be derived straightforward as

$$\Lambda(\mathbf{z}) = \frac{f_{\mathbf{z}|\mathrm{SKIP}}}{f_{\mathbf{z}|\mathrm{CODE}}} \underset{\mathrm{CODE}}{\overset{\mathrm{SKIP}}{\gtrless}} \lambda \cdot P_{\mathrm{CODE}} \tag{2.15}$$

where the threshold of the test $\lambda \cdot P_{\mathrm{CODE}}$ is chosen to satisfy the constraint of P_F. We thus have

$$\alpha = \int_{\mathbf{Z}_{\mathrm{SKIP}}} f_{\mathbf{Z}|\mathrm{CODE}} d\mathbf{Z} = \int_{\Lambda_{\mathrm{SKIP}}} f_{\Lambda|\mathrm{CODE}} d\Lambda \tag{2.16}$$

and Λ_{SKIP}, namely decision region of LRT, locates SKIP present.

2.3.2 Proposed Algorithm

Image statistics may vary significantly even in same sequence. Hence, characterizing the probability to resolve the optimization problem as Eq. (2.16) is definitely mathematical. Instead of working with the difficulty in modeling of density functions, a graph approach operated in "receiver characteristic" of the *likelihood ratio* candidates is then revealed. This approach is well-known as *the Most Powerful* (MP) test [17].

Figure 2.2 exhibits the hypothesis occurrence between SKIP and CODE within 100 frames encoding of CIF sequences "mother_daughter" and "stefan". The histograms demonstrate the encoded blocks distribution for hypotheses in normalized SAD value of block size 4×4 partitioned residuals. For both cases, it is clearly hard to characterize the hypothesis occurrence mathematically such as using curve fitting.

To bypass working with the difficulty in a mathematical manner, a graph approach is then proposed based on receiver operating characteristic (ROC) in this thesis. To resolve likelihood ratio testing of Eq. (2.11), there are some reasonable possibilities for the *observations*, $\mathbf{z}$, and *likelihood ratio*, Λ. We specify the ratio function Eq. (2.17) as candidate for the following statements:

$$\Lambda(\mathbf{r}) = \max_k\{\sum_{ij} |r_{Y,n}^{ij}|\} \tag{2.17}$$

1. Due to chroma subsampling, the luma residuals largely determine the Lagrangian R-D cost.
2. The displacing of SKIP is predicted alone with the neighboring MBs, *i.e.*, independent of motion estimation of current MB.
3. All-zero coefficients can be prior detected by measuring SAD of the SKIP residuals.
4. Block partitioning and $\max_k$ SAD improve the flexibility on detection necessity.
5. The cost in SAD form is relative simple, and is easily incorporated with hardware implementation.

In Eq. (2.17), $n \in \{0, 1, \ldots, 256/N^2 - 1\}$, and the partitioning size $N \in \{4, 8, 16\}$. $\max_k$ takes the k^{th} ranked SAD in descending order. The resolution of motion vector is evaluated as well, since the quantity of the prediction error $\mathbf{r}$ largely depends on the accuracy of the replaced displacement.

By the ROCs experiments demonstrated in section 2.3.4, The proposed SKIP detection LRT becomes

$$\max\{\sum_{ij} |r_{Y,n}^{ij}|\} \underset{\text{SKIP}}{\overset{\text{CODE}}{\gtrless}} \frac{\kappa}{2} \cdot Q_{step} \tag{2.18}$$

with each partitioned block size 8×8. The decision parameter, κ, adapts according to a period of encoding statistics.

2.3.3 The Decision Threshold

The appropriate decision threshold can be dynamically approached according to a period of encoding statistics. Distinct from the general hypothesis problem, the true hypothesis can be evaluated while "CODE" is detected. A CODE is an opposite hypothesis of the SKIP indicated all predictions excluded SKIP. As the SKIP is detected, coder promptly terminates the encoding process and replaces a MB by SKIP coding; otherwise, current MB detection is failed, and the optimal prediction is made according to the Lagrangian

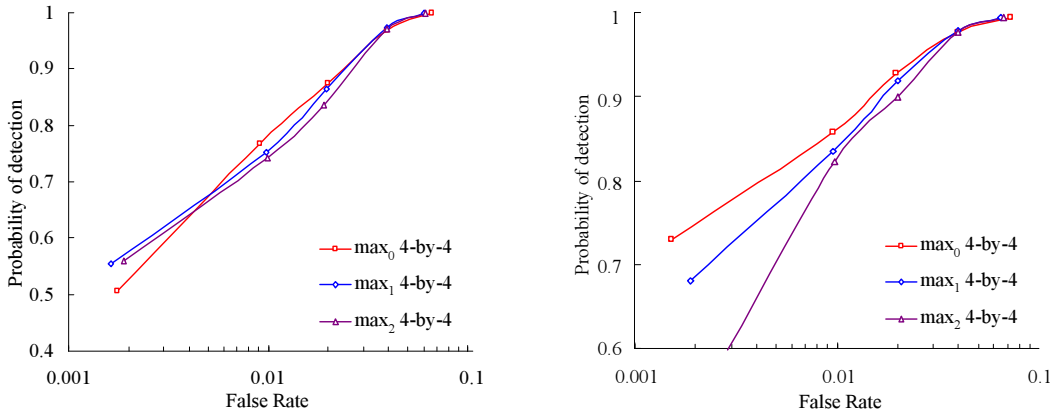
cost calculation. In the case that a series of CODE presents are detected but true hypothesis corresponded or called *miss* during a period of time, the likelihood ratio test is reasonably pessimistic; the threshold should be best increased. Reversely, considering the ratio value is spatial correlated, once the ratio for a CODE MB located in the *alarm region*, between η and η plus *alarm metric*, it reasonably implies threshold usually optimistic. Thus the decision region should be best restricted conservatively, in order to constrain the false rate within an acceptable value. According to the discuss, the algorithm is summarized in the following pseudo-code:

```
//Inter-Frame Initialization
set trials; set TRIALS; set kappa; set alarm_metric;
//Inter-Frame Coding
while(Inter-frame) {
  threshold = kappa*Qstep;
  SKIP_DETECTION();
  MB_ENCODING();
  if(SKIP detected){
    if(SKIP present){
      if(ratio > threshold+alarm_metric){
        trials++;
        if(trials == TRIALS){
          set TRIALS=0;
          kappa++;
        } //full trials
      } // beyond the alarm
    } //SKIP present
  } //SKIP detected
  else
  {
    set TRIALS=0;
    kappa--;
  }
  GET_NEXT_MB();
} //finish Inter-Frame Coding
```

The decision parameter is usually optimistic and has to be restricted in case of the difference between `ratio` and `threshold` is lower than the `alarm_metric`. Otherwise, increase the decision parameter whenever the accumulative number of *passed* SKIP testing "`trials`" reaches the trial bound "`TRIALS`". *Alarm metric* observes the necessity and sufficiency of decision threshold, and accordantly adapts threshold related to "CODE-detected statistics".

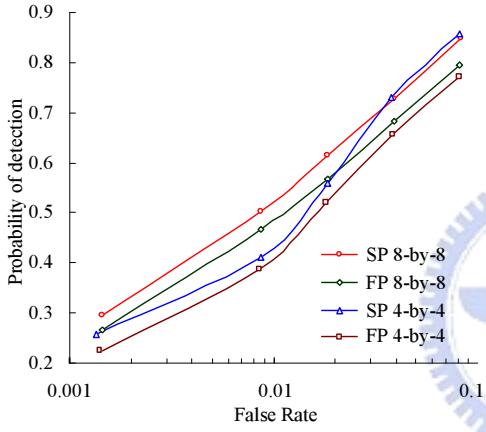
Based on the self-correct mechanism, the proposed LRT thus provides a efficient and effective MB-skipping detection regardless of probabilistic model characterization. The partial graph-based investigations for the flexible LRT parameters are demonstrated in the following.

2.3.4 Determination in Flexibility

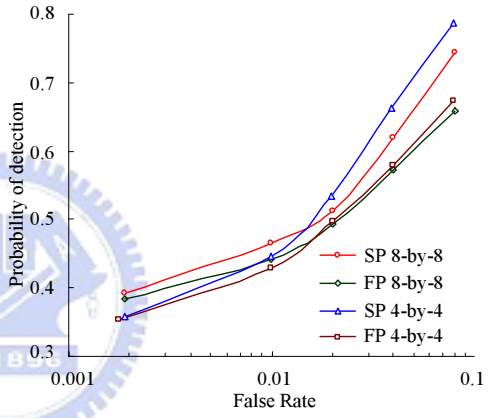


(a) $\max_k$-mother_daughter

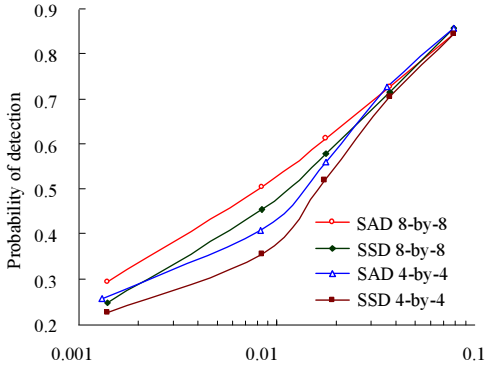
(b) $\max_k$-akiyo



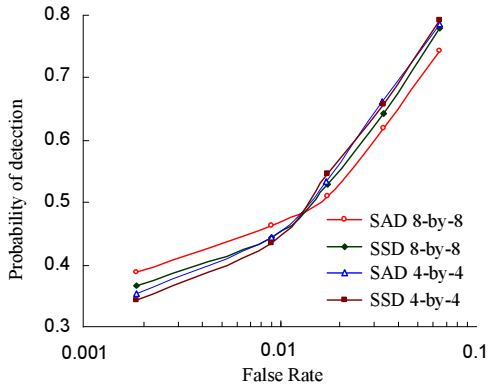
(c) MV resolution-foreman



(d) MV resolution-stefan



(e) MV resolution-foreman



(f) MV resolution-stefan

Figure 2.3: Flexibility analyses in $\max_k$, block size, pixel-resolution, and SAD versus SSD (CIF@30fps, 300 frames, Baseline+RDO)

The likelihood ratio assessment in terms of $\max_k$, block size, displacement resolution, and distortion form were investigated based on the ROC graph approach.

$\max_k$ Figure 2.3(a) and 2.3(b) show the partial ROC comparisons for $\max_k$, with each partitioned size 4×4 and sub-pel SAD. In most cases, the maximum SAD value offers better operating characteristics than other partitioning, especially in static sequences. The experiments succinctly reflect that SAD measuring in 4×4 size is sufficient to detect SKIP coding.

Block size and vector resolution The valid candidates of block size and pixel resolution are $\{4 \times 4, 8 \times 8, 16 \times 16\}$ and full-pel or quarter-pel, respectively. Notice that comparisons excludes block size 16×16 due to poor detection efficiency. The ROCs were compared in Figurer 2.3(c) and 2.3(d). It's straightforward that prediction error largely determined with displacement accuracy; the resolution in fraction is more efficient than integer. In block size comparisons, however, the effect is relatively obscure.

SAD versus SSD Conflicting of that improved performance is followed with the expense in complex computation, the detection performance may be degenerated with more complex squared calculation, as shown in Figure 2.3(e) and 2.3(f).

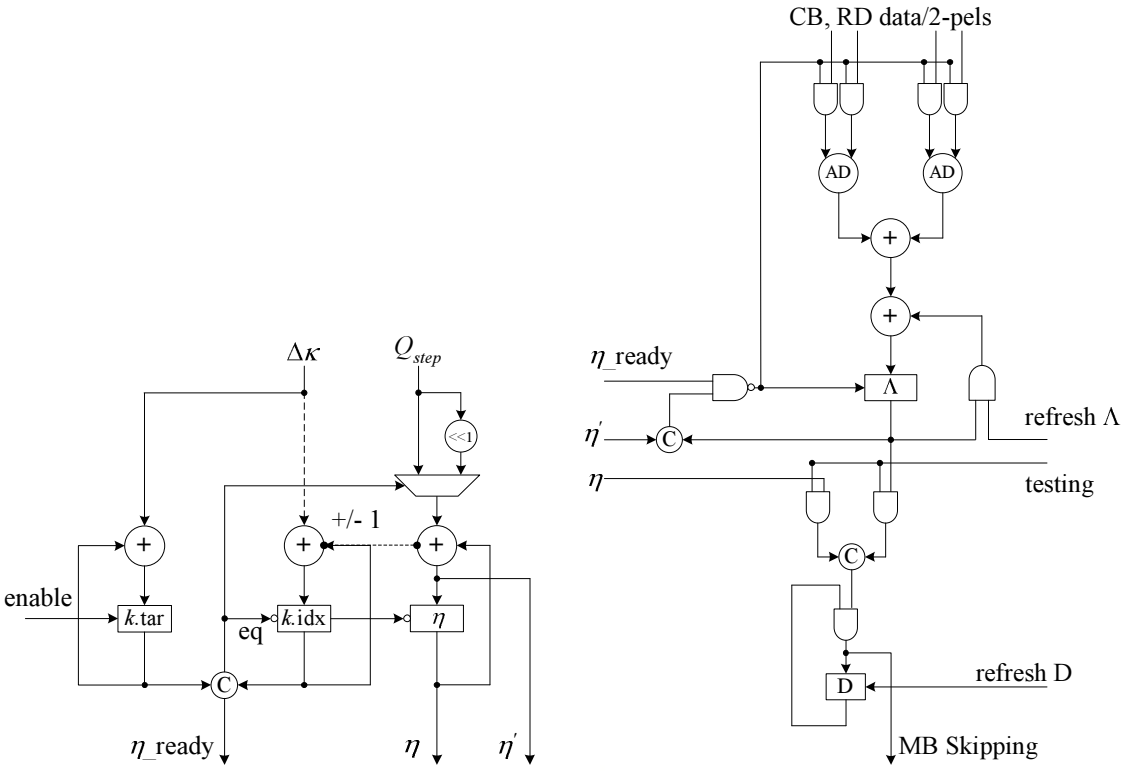
2.4 Logic Implementation

To translate the ratio testing Eq. (2.18) into logic structure is quite straightforward. For low-cost implementation, we instead utilize an accumulator-based structure to avoid the multiplication in decision threshold, since κ always increment/decrement unless inter-frame initialization.

A simple logic based on comparator for decision threshold computing is as Figure 2.4(a). For 1$^{\text{st}}$ encoded MB in P-frame, $\Delta\kappa$ is set to the initial value, and η accumulates until κ equals to the target value. The η' is defined as η plus with the *alarm metric*, setting to Q_{step}, for self-correctness of ratio testing. Figure 2.4(b) shows the proposed equivalent logic structure. The cumulative η is summed with ADs of each two pixels in current block and reference data, total four sub-blocks detected with each sub-block 8×8 pixels. Once current Λ exceeds η', testing is failed, all logic computing terminated until next inter-frame MB. Notice that, for hardware simplification, the design logic shown in Figure 2.4 is integer-resolution; hence degenerated detection probability slightly.

2.5 R-D performance and Detection Characteristics

This section presents the graph approach of LRT based on Eq. (2.18) in the proposed macroblock-skipping detection method. In the experiments, the performance of the proposed MB-skipping method was compared with baseline encoder JM8.6 and RDO mode enabled. 3 Static CIF sequences: akiyo, mother_daughter, and hall_monitor and 3 active



(a) ACC-based threshold computing logic (b) Proposed low-cost LRT logic

Figure 2.4: Logic implementation on macroblock-skipping detection

CIF sequences: foreman, mobile, and stefan were tested in flexibility, rate-distortion performance, and detection characteristics with one reference frame, one slice per frame, and P-slices only at encoding rate 30fps. All predictions are applied.

The P-frame R-D performance evaluation factor ΔBitrate and ΔPSNR in whole thesis are defined as

$$\Delta\text{Bitrate}(\%) = \frac{\text{P\_Bits}_{\text{proposed}} - \text{P\_Bit}_{\text{JM}}}{\text{P\_Bits}_{\text{JM}}} \times 100\% \tag{2.19}$$

$$\Delta\text{PSNR}(\text{dB}) = \text{P\_PSNR}_{\text{proposed}} - \text{P\_PSNR}_{\text{JM}} \tag{2.20}$$

Table 2.2 summarizes R-D performance and detection characteristics in integer resolution of SKIP displacing for different encoding scenarios, where the testing false rate, α, is constrained to 1%. To further compare in sub-pel resolution, we summarize R-D performance and detection characteristics in Table 2.3 of detail QPs as 36 and 42. Figure 2.5 plots the corresponded rate-distortion curves of test sequence hall_monitor and foreman.

By experiments, the proposed macroblock-skipping method conditionally eliminates the computational redundancy of inter-frame coding, while the rate-distortion performance is substantially preserved.

Table 2.2: Detection characteristics and rate-distortion performance in integer resolution, compared with JM 8.6 (CIF@30fps, Baseline+RDO, QP={36, 42})

Sequence(CIF)	ΔBitrate(%)	ΔPSNR(dB)	P_D(%)	P_F(%)	MB-skipping(%)
akiyo	−1.40%	−0.020	86.88%	0.99%	81.77%
mother_daughter	−0.75%	−0.047	86.99%	1.07%	78.25%
hall_monitor	−4.16%	−0.070	81.07%	1.84%	74.62%
foreman	−0.22%	0.013	54.67%	0.88%	35.88%
mobile	−0.42%	−0.010	16.52%	1.26%	9.52%
stefan	−0.28%	−0.004	44.77%	1.23%	22.25%

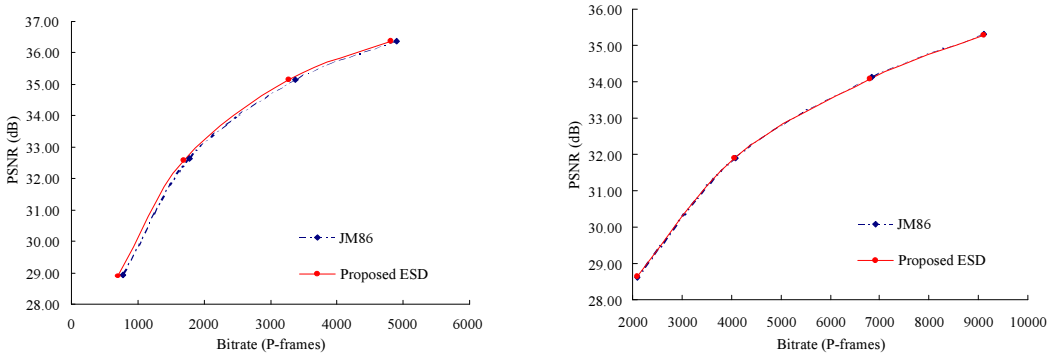
Table 2.3: Detection characteristics and rate-distortion performance evaluation in fractional resolution, compared with JM 8.6 (CIF@30fps, Baseline+RDO)

Sequence (CIF)	QP	ΔBitrate (%)	ΔPSNR (dB)	Probability of detection (%)	False rate (%)
akiyo	36	-0.63%	-0.019	82.97%	1.00%
	42	-2.17%	-0.021	90.78%	0.98%
M&D	36	-0.53%	-0.023	75.03%	1.20%
	42	-0.96%	-0.071	98.95%	0.93%
hall_monitor	36	-4.48%	-0.068	70.36%	1.96%
	42	-3.84%	-0.072	91.78%	1.72%
foreman	36	-0.14%	0.010	50.57%	0.91%
	42	-0.29%	0.016	58.76%	0.84%
mobile	36	-0.10%	-0.012	14.83%	0.99%
	42	-0.73%	-0.008	18.20%	1.52%
stefan	36	-0.15%	0.004	44.32%	1.07%
	42	-0.41%	-0.012	45.21%	1.38%

2.6 Conclusion

To exploit the coding characteristic in H.264 low-rate application, based on the Lagrangian RDO framework, this chapter suggests a low-complexity LRT-based algorithm and corresponded architecture for MB-skipping detection.

Theoretical analyses state some MB-skipping detection criteria and corresponded methods according to R-D cost properties and the assumption of encoding statistics. To maximize the probability of detection subject to a acceptable false rate, a LRT-based MB-skipping detection technique is therefore presented by a graph investigation



(a) Test sequence: hall_monitor (b) Test sequence: foreman

Figure 2.5: Corresponded rate-distortion curves

approach regardless of the infeasibility of probability parameterization. The self-correct decision threshold is achieved at the constrained rate of false detection related to the "CODE-detected" characteristics. Simulation results indicate that the proposed MB-skipping detection has the probability of detection 16%–86% at the false rate 1%, and the corresponded skipping rates are 10%–81%, where the logic count of the corresponded architecture is about 1Kgates.

Chapter 3

Low Power Algorithms

Motion estimation demands on prime power and visual quality, which becomes a most crucial component to realize a video codec. In recent years, a verity of related studies on low power algorithms have been proposed to cross the obstructing of handset device implementation. Design in higher levels of abstraction is clearly to provide the more degree of freedom both in terms of design intent and constraints. Yet most of these design suffered from the motion-compensated prediction (MCP) reliability due to the excess attention on *hardware-oriented* considerations, and hence essentially affects system performance.

To elegantly resolve the reliability problem, this chapter mainly deals with the abstraction on motion estimation algorithms having excellency in rate-distortion performance. An advanced estimation strategy using dynamic boundary decision is proven significantly improving the R-D quality. In addition, the associated computational burden and power dissipation are both substantially preserved. These key advantages have made the proposed algorithm successively brigading the gap between requirements of low power and high coding performance in a video codec.

This chapter is comprised of four sections. Section 3.1 introduces the power measurement used in digital CMOS circuits. Section 3.2 describes the block-matching motion algorithms and the relative deign issues as an example implemented in H.264 reference software. Section 3.3 demonstrates a high quality motion search algorithm using dynamic boundary decision. Lastly, in section 3.4, the design parameters for well-known bitwidth-truncated technique as well as block size of predictions are both analyzed to further reduce the power dissipation and implementation overhead.

3.1 The Power Dissipation

It was not until recent years that power consumption was only a secondary metric of concern in comparison to area and speed. This thinking has begun to change and power becomes most crucial factor than area and speed followed with phenomenal growth of

portable electronics. To clearly state the power consumption, this section briefs an overview of the sources of power in digital CMOS circuit.

The power consumption in digital CMOS circuits is typically formulated as follows [18]

$$P_{avg} = I_{standby}V_{dd} + I_{leakage}V_{dd} + I_{sc}V_{dd} + P_{sw.cap.} \tag{3.1}$$

where

- the *standby current* $I_{standby}$ is the DC current drawn continuously form the power supply (V_{dd}) to ground,
- the *leakage current* $I_{leakage}$ is primarily determined by the fabrication technology, caused by
  - the reverse bias current in the parasitic diodes formed between source and drain diffusions and the bulk region in a MOS transistor, and
  - the subthreshold current that arises from the inversion that exists at the gate voltages below the threshold voltage,
- *short-circuit current* I_{sc} is due to the DC path between the supply rails during output transitions, and
- the last term, $P_{sw.cap.}$, refers to the capacitive switching power dissipation, caused by the charging and discharging of parasitic capacitances in the circuit.

The Static and Dynamic Power In general, power consumed in a digital circuit can be more simply classified into *static power dissipation* and *dynamic power dissipation*.

- The term *static power dissipation* refers to the sum of the standby and leakage power dissipations. Leakage currents in digital CMOS circuits can be reduced with the proper choice of device technology. Standby currents play an important role in design styles like pseudo-nMOS and nMOS pass transistor logic and in memory cores.
- The term *dynamic power dissipation* refers to the sum of the short-circuit and capacitive switching power. The dynamic power usually accounts over 70% of the overall power in a non-specialized ASIC (application-specific IC).

Each dissipated term has further briefed in the following.

3.1.1 Power in CMOS Logic

The Standby Current

The standby current and hence standby power consumption occurs when both the nMOS and the pMOS transistors are continuously on. This could happen, for example, in a

pseudo-nMOS inverter, when the drain of an nMOS transistor is driving the gate of another nMOS transistor in a pass-transistor logic, or when the tri-stated input of a CMOS gate leaks away to a value between power supply and ground. The standby power is equal to the product of V_{dd} and the DC current drawn from power supply to ground.

The Leakage Current

The leakage current is decomposed into two components that are shown in the following equation

$$I_{leakage} = I_{diode} + I_{subthreshold} \tag{3.2}$$

The term I_{diode} refers to the currents following through the reverse biased diodes that are formed between the diffusion regions and the substrate. This term is proportional to the area of the source or drain diffusion and the leakage current density and is typically in the order of $1pA$ for a $1micron$ technology.

The term $I_{subthreshold}$ refers to the currents arising due to the fact that transistors that are "off" conduct some non-zero current.

Leakage current becomes a larger and larger problem as geometries shrink and threshold voltages drop. The leakage current in a .13μm process with a threshold voltage of 0.7V is about 10–20pA per transistor. In that same process, if the threshold voltage is lowered to 0.2–0.3V, then leakage current skyrockets to 10–20nA per transistor. For a 1M transistor chip, leakage power can increase from 15μW to 15mW due to a lower threshold. In the practical measurement, the leakage power in our chip implementation is about 400μW with core area 0.69μm^2 in a 0.13μm technology.

Leakage current depends on the V_{dd} (or how close it is with respect to threshold voltage), threshold voltage itself (V_t), the transistor aspect ratio (W/L) and the temperature. Leakage power used to be only 5% for technologies 0.18μm and above. As the voltage scales down with technology, this has increased exponentially and has become a problem in nano-meter technologies. For instance, leakage power occupies major proportional in average of 68% in this work. Increasing die area affects the leakage power adversely, as this causes a number of transistors to be increased.

The Short Current

The short current is caused by direct supply-to-ground paths that are created due to transents in signal values cause. For instance, when the input of a CMOS inverter changes from logic 1 to 0, there is a period of time when both the nMOS and pMOS transistors are conducting, leading to a short-circuit current being drawn from the supply. Assuming symmetric rise and fall delays and threshold voltages, the short-circuit power dissipation

of a CMOS inverter is approximately cubic relation to supply voltage V_{dd} as following

$$P_{sc} = K(V_{dd} - 2V_T)^3 \frac{\tau N}{T_{clk}} \tag{3.3}$$

where K is a constant the depends on the transistor sizes and the technology, V_T is the magnitude of the threshold voltage of the nMOS and pMOS transistors, τ is the input rise/fall time, N is the average number of transition at the inverter's output, and T_{clk} presents the clock period in sequential logics. Short-circuit current can be controlled by minimizing the transition times on nets, or to a small portion of the total power by appropriate sizing of transistors and reducing the input rise/fall times to all the gate in the circuit. It is also reduced by scaling the supply voltage, and by reducing the switching activity at the gate output.

The Capacitive Switching Power

The average capacitive switching power ($P_{sw.cap.}$) over the digital CMOS circuit can be expressed as

$$P_{sw.cap.} = \sum_i C_{L_i} \cdot V_{dd}^2 \cdot f_{clk} \cdot \alpha_{i,0\to1} \tag{3.4}$$

C_{L_i} indicates equivalent loading capacitance that is to be charged on the node i, V_{dd} is the power supply voltage, f_{clk} is the clock frequency, and $\alpha_{i,0\to1}$ is named as the *switching activity* being the average number of output logic 0 transiting to 1 on the node i. The switching activity is defined mathematically as

$$\alpha_{i,0\to1} = \lim_{N\to\infty} \frac{\text{number of switching}_{i,0\to1}}{N} \tag{3.5}$$

where N represents the total number of clock cycles. For example, a clock signal, said *clk*, always transits one period during a clock cycle, which simply means that $\alpha_{clk,0\to1} = 1$, while a gated-clock signal, said *gclk*, often switches less than once in a period of cycle, *i.e.,* $\alpha_{gclk,0\to1} \leq 1$, hence suppressing the "switching capacitance" $(C_{L_i} \cdot \alpha_{gclk,0\to1})$. Another common measurement, product of the clock frequency and the switching activity $(f_{clk} \cdot \alpha_{i,0\to1})$ is the expected number of transitions per clock cycle, and also referred to as the transition density [19].

Technology scaling has resulted in smaller transistors and hence smaller transistor capacitances. Unfortunately, interconnect capacitance does not scale with the process and has become the dominant component of capacitance. With technology scaling, designer is capable of packing more modules in a single chip. This has increased the number of interconnects and thus the overall power dissipated.

In most static CMOS technologies, the capacitive switching power accounts for a predominant part of the total power. As a result, most power estimation and optimization methods targeting technologies focused on reducing the component of a circuit's

power consumption. Since dependence of power on voltage is quadratic, voltage scaling can efficiently eliminate switching power dissipation. One can simply apply voltage scaling by using more advanced manufacturing technology or a specialized cell library, *i.e.,* multi-threshold or low power cells. Yet the primary focus of our work spots the design abstraction on the higher algorithmic and architectural levels, we do not interest in such technology-dependent low power manners. Surely, some well-known architectural techniques were already adopted in the work to further reduce the switching power, such as clock gating, operand isolation, and bus coding, *etc.*. These will be stated in the rest sections of this thesis.

3.2 Motion Estimation in H.264/AVC

Motion estimation have attended recently in research and industry, because of the reasons:

1. Motion estimation is computational most demanding of a video encoder (about 60%–80% of the total computational load). Is the bottleneck to performance of an encoder.
2. The motion estimation has a high impact on the visual performance of an encoder.
3. The video codec does not specify a standardized method to motion estimation, thus being open competition.

Section 1.2.2 has briefly described the hybrid video coding structure as well as the motion-compensated prediction (MCP) technique. Motion-compensated prediction using motion estimation effectively exploits the temporal redundancy of video sequences and is therefore a primary part of video compression standards, such as in MPEG series and in H.26× series. In block-based video coding, the frame elementary segment, macroblock, in the imaging plane can be represented as predictive contents in the indicated area with the associated difference approximation. Often the predictive content needing only a minor alteration can be a satisfactory fiction to the new picture. Using the refining difference approximation usually results in the further achievement of higher compression performance than using INTRA alone. In video decoding, the new picture is restored by repeating the frame contents of the prior pictures compensated with the refining difference approximation. The proceeding on exploiting the repeating contents is referred to *motion estimation*, and the approximation refining using predicted difference is then referred to *motion compensation*.

Motion estimation is also applied to other applications than video encoding like image stabilization, computer vision, motion segmentation, and video analysis. However, the requirements for these applications differ essentially for video encoding algorithm: motion vectors have to reflect the real motion within the image sequence, otherwise the results will not be desired. In video encoding the situation is relatively tolerant. Motion

vectors are used to compensate the object motion of imaging plane and only the residuals (remaining difference) have to be encoded and transmitted. Therefore, the motion vectors may be selected in the manner of minimizing the total required bit-stream size for given acceptable visual quality. In the thesis, the investigation on motion estimation algorithms are restricted within video coding for portable application. Yet similar concepts are applicable to other areas as well.

3.2.1 Block-matching Algorithm

The relative studies (not specialized in video coding field) can be classified into two major categories, the time-domain algorithms and the frequency-domain ones [20,21]. The time-domain methods include feature-matching, recursive, and gradient-based algorithms, *etc.*. The frequency-domain algorithms comprises phase-correlation and transform-based (*e.g.* Fourier Transform, DCT, wavelet) algorithms.

Most of the these schemes are generally based on block-matching, matching of (all/some) pels of the current block with a candidate block in the search area. Block-matching algorithm (BMA) is the one of the most popular techniques for motion estimation. The basic concept of BMA is to represent the block in current frame using a block in the reference and a motion vector indicating their displacements. The block in reference picture that is a best match is the one that minimize the matching cost. A search range is set to confine the search procedure within an area that is more probably to have a good match. Various derivative strategies are proposed to further lower the searching time or searching cost.

Matching Criterion

The motion vector is the displacement of the current macroblock and the block that minimize the matching cost. Various matching cost has been adopted. These cost function vary in terms of implementation complexity and inefficiency. The following are commonly used cost functions in implementation [21]:

MSE (Mean Squared-Error): The mean square error produces outstanding results. The MSE finds a block that minimizes the squared-error of two blocks, which is close to human visual perception. In addition, the square term appears in both MSE and PSNR equation. Thus, MSE generally produces excellent PSNR results. However, the MSE also suffers from high computational complexity due to the square term of cost function.

MAE (Mean Absolute Error): The mean absolute error is similar the MSE function but replace the square operation with absolute operation. It has much less computation complexity comparing to MSE, and make it a popular cost function, especially for VLSI design. One problem for the MAE function is that small and

large error are treated equally, while the MSE emphasize on large errors by squaring the errors.

SAD (Sum of Absolute Difference): The SAD is very similar to the MAD function. The SAD does not divide the summation of errors by the total number of pixels in the block, which is a constant through out the encoding process. This makes SAD more practical then MAD for implementation. The SAD criteria is most popular in hardware implementation of block-matching algorithms. The block-based nature leads to regularity and parallelism which are suitable for hardware-based realization.

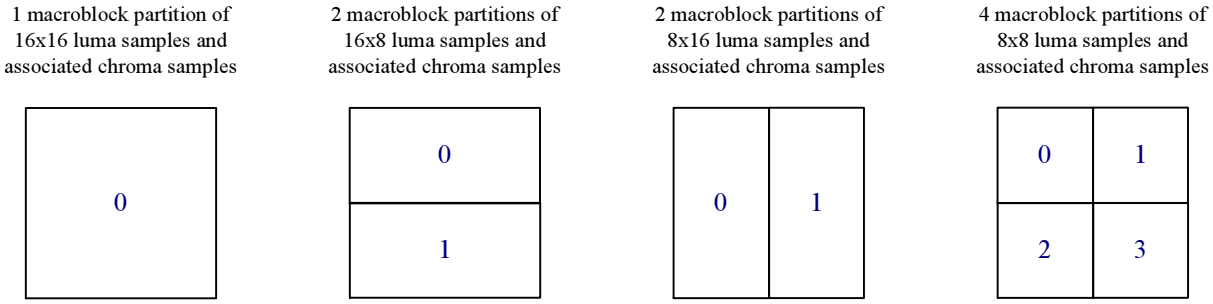
MME (Minimized Maximum Error): The MME function inspects the maximum error of a block and select the block that has minimum maximum error. The advantage is that a 8-bits comparators is required to get the maximum error instead of a 16 bit accumulator to save the differences of MAE.

TRANS (Transformation of Block): The block is first transformed to a block with lower bits resolution. One bits is used to represent the block after transform. Then, apply the technique similar to SAD on the reduced resolution block to find the motion vector. Computations are saved in the process of finding motion vector, but the transformation requires extra computation overhead.

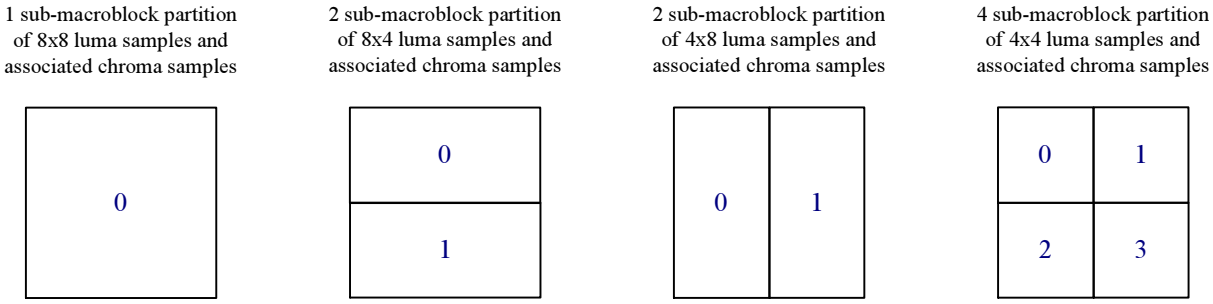
3.2.2 Motion Estimation in H.264/AVC

This organization simply describes several advanced functionalities of the motion estimation for H.264 codec and software realization in JVT Joint Model (JM), including

Variable Block Size Motion Estimation Accurate prediction efficiently reduces the degree of displaced-difference between the current macroblock and the predictive region. During block-based motion compensation, it is not possible of macroblock to contain more than one moving object with exactly one direction. Traditional video coding using MB size of compensated-prediction is hence not enough good fitting to these objects in the moving sequence. For this reason, multi-block motion estimation is therefore adopted in H.264/AVC for better compression performance. The luminance components of each macroblock are most split up in four ways, as shown in Figure 3.1(a). The motion compensation can be one macroblock partitioning, two half thick partitions (each 8×16 pixels), two half tall partitions (each 16×8 pixels), or four quarter squares partitioning (each 8×8 pixels). The 8×8 partition can be divided into further four manners, as shown in Figure 3.1(b). These subdivisions are one 8×8 partitioned sub-block, two 4×8 sub-blocks, two 8×4 sub-blocks, and four equal size partitions with each size 4×4 pixels. The associated chroma block sizes are given in Table 3.1. The approval macroblock segmentation leads to a large number of combinations. Each block in the coded area is compensated by indi-



(a) **Macroblock partitions**



(b) **Sub-macroblock partitions**

Figure 3.1: Partitioning of a MB for motion compensation

Table 3.1: Chroma block sizes associated with luminance partitions

Block size	Luma	16×16	16×8	8×16	8×8	8×4	4×8	4×4
(full pixel)	Chroma	8×8	8×4	4×8	4×4	4×2	2×4	2×2

vidual motion vector. This method of partitioning macroblock into motion compensated sub-blocks is known as *tree-structured* motion estimation.

Enhanced Fractional Resolution Prior ITU-T H.263 standard is capable of supporting the motion-compensated prediction in half-pel accuracy for both luma and chroma MVs [22]. The H.264 motion compensation adopts the chroma MVs at $\frac{1}{8}$ pixel resolution are derived form transmitted luma MVs of $\frac{1}{4}$ pixel resolution. The fractional-pel motion compensation extends the search points form integer resolution to half-pel or even quarter-pel resolution, which further eliminates the displaced-frame difference (DFD) and improves the coding efficiency. The pixel values of fractional pixels are interpolated and the matching cost are calculated on the fractional search points using these interpolated pixels. Since the computation is too expensive to explore exhaustive fractional search, the block matching on fractional-pel is usually separated from integer motion estimation; *i.e.* the fractional motion estimation is performed on the fractional search points around the integer position just found.

Multiple Reference Frame Motion Estimation Multiple reference frame motion estimation is a feature adopted by H.264. In the pervious video coding standards, the motion estimation allows to find out the best matching block in the previous or/and future one reference frame. This manner is better for the smoothly moving sequence. But for the high motion sequence, the best matching block may occur in the reference frame which is not the adjacent one. For the reason, H.264 takes multiple reference frames into consideration for motion estimation. Instead of only search previous one reference frame, H.264 is capable of searching most five previous frames for finding the best fitting. Therefore, adopting multiple reference frames increase the access frequency according to a linear model, 25% complexity increased for each add frame. By using multiple reference frame motion estimation, a negligible gain (less than 2%) in bit rate is observed for low and medium motion sequence, but more significant saving can be achieved for active motion sequences (up to 10%) [23].

3.2.3 R-D Optimization using Lagrangian Multiplier

Video codec did not standardize the encoding process; they bestowed the flexibility of optimizing the encoder. Similar to ITU-T H.263 TML, Test Model Long Term, the Joint Video Team of MPEG and ITU-T also drafted the open source test model to establish reference platform, namely JM (Joint Model) [10]. For objective competition, JVT uses the rate-distortion metric to discriminate system performance, which only concentrates on *distortion* versus *bit-stream size*. In practice, most actual comparisons use quantitative measurements as SSD (sum of squared-difference) or its logarithmic equivalent, known as PSNR (peak signal-to-noise ratio) as distortion model. The equivalency between PSNR and SSD is given as

$$\mathrm{PSNR}_{\mathcal{A}} = 10 \log_{10} \frac{P^2 |\mathcal{A}|}{\mathrm{SSD}_{\mathcal{A}}} \text{ decibels} \tag{3.6}$$

where $\mathcal{A}$ denotes data arguments size, and P presents maximum pixel value, usually 255. Therefore, the Lagrangian cost which minimizes reconstructed SSD given in a acceptable bit-rate number, was firstly addressed in [24] to achieving optimum rate-distortion performance.

In an ideal R-D optimization problem, the modes of operation that are assigned to the image regions have differing rate-distortion characteristics, and the goal of an encoder is to optimize its overall fidelity: *Minimize distortion subject to a constraint rate among K available coding modes*. Let $\mathbf{X} = (X_1, X_2, \ldots, X_N)$ denote a group of N macroblocks, For a vector of N macroblock coding mode allocations $\mathbf{M} = (M_1, M_2, \ldots, M_N)$ and a rate constraint R_c, this constrained problem reads as follows

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} D(\mathbf{X}, \mathbf{M}) \tag{3.7}$$

subject to

$$R(\mathbf{X}, \mathbf{M}) \leq R_c \tag{3.8}$$

where $\mathbf{M}^*$ is a vector of optimal mode allocations, $D(\mathbf{X}, \mathbf{M})$ is a distortion metric calculated for the group of coded and decoded macroblocks $\mathbf{X}$, and $R(\mathbf{X}, \mathbf{M})$ is the coded rate of the group of macroblocks $\mathbf{X}$. The unconstrained optimization task in Eq. (3.7) can be elegantly combined with rate constraint of Eq. (3.8) by introducing Lagrange multiplier, λ, as follows [25,26]

$$M^* = \arg\min_{\mathbf{M}} \sum_{i=1}^{N} J(X_i) \tag{3.9}$$

where $J(X_i)$ is the rate-distortion cost for MB i and is given as

$$J(X_i) = D(X_i, \mathbf{M}) + \lambda R(X_i, \mathbf{M}) \tag{3.10}$$

and Lagrange multiplier is chosen considering all N macroblocks, such that the rate constraint holds.

This is typically a complex problem due to inter-dependence between macroblocks. The complexity can be reduced from exponential to linear in the number of macroblocks by assuming coding of macroblocks irrelavent [26], in general leading to a suboptimal solution

$$\mathbf{M}^* = \{\arg\min_{M_i} J(X_i)\} \tag{3.11}$$

where now

$$J(X_i) = D(X_i, M_i) + \lambda R(X_i, M_i) \tag{3.12}$$

Lagrangian optimization using Eq. (3.12) separably optimizes rate-distortion into one macroblock and the functional J is minimized for a particular value of the Lagrange multiplier λ. It's therefore referred constrained RDO, where the Lagrangian cost used in H.264 JM was already stated in section 2.2.

Lagrangian Motion Estimation

Video coding using Lagrangian optimization has proven significant benefits on R-D performance if it's judiciously applied. The analogous sense is ideally extended into the variable block size motion estimation, where the Lagrangian optimization can be simply preformed to search for an MV that minimizes the cost function prior to residual coding, as the following

$$J_{\text{motion}} = D_{\text{DFD}} + \lambda_{\text{motion}} R_{\text{motion}} \tag{3.13}$$

in which the distortion D_{DFD}, representing the prediction error measured as SSD or SAD, is weighted against the number of bits R_{motion} associated with the MVs using a Lagrange multiplier λ_{motion}.

In JVT test codec JM, the matching criteria of integer-pel motion estimation determines the best fitting displacement in the R-D optimization sense, as Lagrangian functional

$$\mathbf{m}_n^* = \arg\min_{\mathbf{m}} J_{\text{motion}}(\mathbf{m}|n) \tag{3.14}$$

and

$$J_{\text{motion}}(\mathbf{m}|n) = \sum_M \sum_N |x_{ij} - y_{ij}^{\mathbf{m}}| + \lambda \cdot R_m(4\mathbf{m} - \mathbf{p}_n) \tag{3.15}$$

where $J_{\text{motion}}(\mathbf{m}|n)$ uses SAD measuring the prediction distortion, which presents R-D cost for n^{th} partition motion estimation with $\mathbf{m}$ displacing (integer), the number M, N are block size of current partition, x_{ij} and y_{ij} donates predicted content in current macroblock and reference data of prior frame respectively, the term $R_m(4\mathbf{m} - \mathbf{p}_n)$ maps codeword length in quarter-pel of MVD (motion vector difference), $4\mathbf{m} - \mathbf{p}_n$, and λ is related with quantization parameter QP, predetermined as

$$\lambda = \sqrt{0.85} \times 2^{\frac{QP-12}{6}} \tag{3.16}$$

The block-matching algorithm applies Lagrange multiplier as Eq. (3.14)–Eq. (3.16) efficiently improving the motion-compensated prediction quality. However, cost measuring involved inter-predictor dependence depredates the regularity and modularity for hardware implementation. The arithmetic requirement of multiplication increases the cost as well. To be effective, we instead represent an accumulation-form approximation associated with its logic structure, which will be stated in section 4.5.

3.3 Full Search Algorithm using Dynamic Boundary

Specialized refinement on the abstraction of system or algorithm provides most superior benefits on realization performance. As the motion estimation is heavy power demand in a video codec, prudently specializing the matching algorithm yields multiple times or more power saving than that on RTL (register-transition level) or on physical (*e.g.* synthesis or placement and routing). Hence for powerful motion estimation design, this section illustrates a block-matching algorithm using dynamic boundary decision strategy, which significantly improves R-D quality as well as the associated arithmetic efficiency.

Full search block-matching algorithm within the limited search range, maximum displacement around the search window, exhaustively matches displaced candidates. Since displaced frame difference (DFD) usually monotonically decreases as search range increasing, to constrain DFD to an acceptable amount, search range should be effectively large relative to its frame size. For instance, the value is typically regulated at least 16 pixels in CIF (352×288) and 32 pixels for SDTV (standard definition television, frame size 720×480) format.

Exhaustive search within wide range of search window provides a more superior degree in rate-distortion performance. However, ponderous computation resource is thus wasted in estimating the large number of unnecessary candidate blocks, since object motions usually surround the predicted displacing. To diminish the redundancy without sacrificing R-D metric, a dynamic search range scheme is presented to efficiently exploit the temporal correlations by boundary prediction. It has experientially proven significant improving on motion-compensated quality and on power-efficiency than ever prior works. The presenting advantages have made successively brigading the gap between requirements of low power and high coding performance in a video codec.

3.3.1 Operations and Reference Buffer Bandwidth

In practical, despite of synchronized clock network, power dissipation of motion estimation mainly comes from two ways: the arithmetic operations and the memory buffering. Measuring power in upper design abstraction can be hence roughly characterized by *the number of operations* and *reference buffer bandwidth* with dedicated implementation architecture.

Arithmetic Operation Rate Let the picture frame be segmented in $N \times N$ macroblocks. If the displacement is determined for each macroblock, $N_l \times N_p/N^2$ displacement vectors have to be computed for a frame, where N_l and N_p are the number of pixel on frame width and on frame height, respectively. For simplification, we assume that the SAD criteria is applied for full search matching, and thus each MB has $(2\mathrm{SR}+1)^2$ candidate blocks to be examined, where SR denotes one side search range. All N^2 subtractions, absolute operations, and additions of the absolute values are performed in total $3N^2 \cdot (2\mathrm{SR}+1)^2$ arithmetic operations during motion search for each MB. Eq. (3.17) shows that the approximate estimating of computation rate corresponds to the frame size $N_l \times N_p$, and the frame rate f_F. Notice that computation rate Op is irrelevant to its block size N.

$$\mathrm{Op} \sim 3 \cdot (2\mathrm{SR}+1)^2 \cdot N_l \cdot N_p \cdot f_F \tag{3.17}$$

Reference Buffer Bandwidth Precisely, memory access bandwidth is insufficient to accurately estimation power dissipation, since memory power dissipation is largely technology-dependent.[1] However, this metric is somehow intuitively providing effective system exploring prior to architecture implementation.

In the following, we assume that 2-D full parallelism arithmetic is applied for MB-pipelined memory access.[2] For rapid power analysis, we only regard the bandwidth when

[1] An experimental illustration will be stated in 4.3 (pp. 58)

[2] Refer to 4.1 (pp. 45) for detail description

Table 3.2: Arithmetic Operation and memory access bandwidth comparisons of full search BMA ($N = 16$, $n = 8$)

	N_l	N_p	SR	f_F	Arithmetic Op. (GOPS)	$\mathrm{BW_{mem}}$ (Gbit/s)
QCIF/15	176	144	16	15	1.24	0.30
QCIF/30	176	144	16	30	2.48	7.23
CIF/15	352	288	16	15	4.97	14.45
CIF/30	352	288	16	30	9.94	28.90
SDTV	720	480	32	30	131.41	539.14
HDTV/720P	1080	720	64	30	1164.60	7800.14

buffer reads, because the power dissipated in internal memory predominates during performance of candidate matching (data loading). For each MB, as search area consists of $(2\mathrm{SR} + N)^2$ pixels in a full search scheme, $(2\mathrm{SR} + N)^2$ pixel words from reference buffer are necessary to compute. Let n be the word width for each arithmetic data with total $N \cdot (2\mathrm{SR} + N) \cdot (2\mathrm{SR} + 1)$ words broadcasting from reference buffer. Then buffering bandwidth in bit per second can be roughly characterized as

$$\mathrm{BW_{mem}} = n \cdot N \cdot (2\mathrm{SR} + N) \cdot (2\mathrm{SR} + 1) \cdot (N_l \cdot N_p / N^2) \cdot f_F \tag{3.18}$$

If a video sequence with $N_l \cdot N_p = 352 \times 288$ pixels per frame and the frame rate $f_F = 30$ Hz is encoded, the full search BMA of maximum displacement SR = 16 and 16×16 block size approximately produces 10 Giga Operations Per Second (GOPS) of operation rate, and results in 28.9 Gbit/s bandwidth requirement of each pixel 8 bit depth. Table 3.2 compares operation rate versus reference buffer bandwidth ($\mathrm{BW_{mem}}$) in different video formats and search ranges. To be clear, Figure 3.2 sketches they growth in log scale.

Motion estimation using full search scheme is extremely computation intensive. The complexity of a block-matching algorithm is increased in hyper-exponential since search range and frame size reproduce coherently. To release computations, fast algorithms using search point subsample were instinctively presented for a long while, such as well-known three-step search, four-step search, or diamond search, *etc.* [21]. These fast search methods usually suffer from the trapped problem due to matching aliasing as well as regional-dependence search strategy. Besides, they are hard to implement through VLSI design because of its poor data regularity and poor support on variable block size motion estimation (VBSME). Data regularity reflects memory access efficiency; poor data regularity leads poor data reuse efficiency, hence increasing memory I/O bandwidth. Since matching order may differ from each block partitioning, the searching strategy of these fast algorithm is in general required to repeat N times individually, if each MB contains N levels of partitioning. Such inability of intra-candidate data reuse fairly wastes the

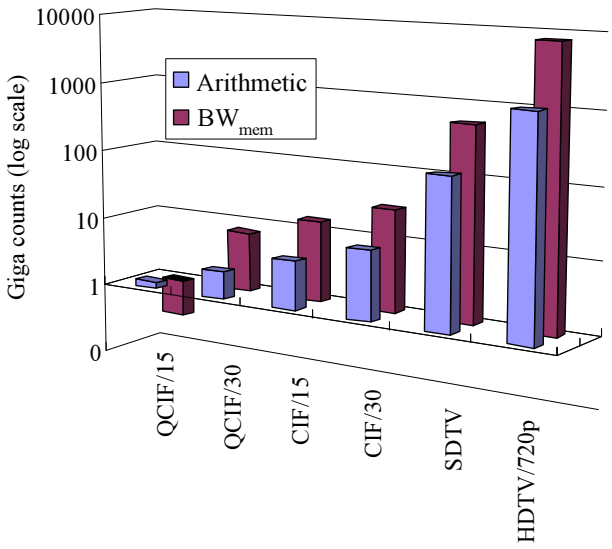


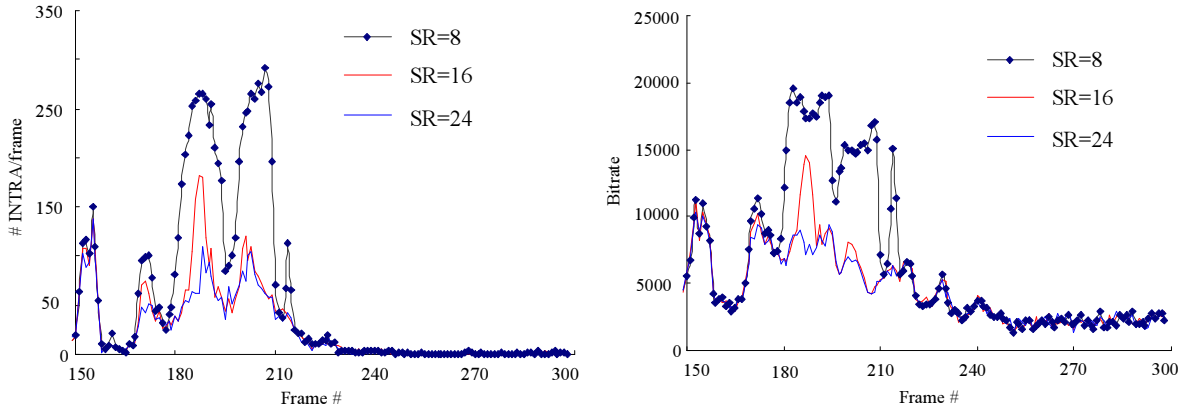
Figure 3.2: Arithmetic Operation versus memory access bandwidth in full search BMA ($N = 16$, $n = 8$)

computation and power. These inherent defects both on algorithm and on hardware issue make traditional fast algorithms less worth realizing in a hardware manner.

Accordingly, instead of the matching subsample scheme, numerous methods based on the *hardware-oriented* algorithms, such as [27–34], have been widely investigated to facilitate low power implementation in recent years. These techniques primarily concentrated on the characteristics of hardware acts, *e.g.*, design metrics of data-reuse efficiency, memory I/O bandwidth, and arithmetic utilization, thus so-called *hardware-oriented*. The design metrics largely affect the performance attributes of system realization such as coding efficiency, complexity, and implemented cost. Indeed, excessively taking account on these metrics may explicitly diminish system performance, even without apparently attaining satisfactory power reduction. For instance, to maximum reference pixels reuse region, search center often used original in most designs. Such restricted reference region however significantly degrade compensated-prediction fidelity, hence presenting poor system R-D performance without when an active scene is encoded. This will be further demonstrated in section 3.3.2.

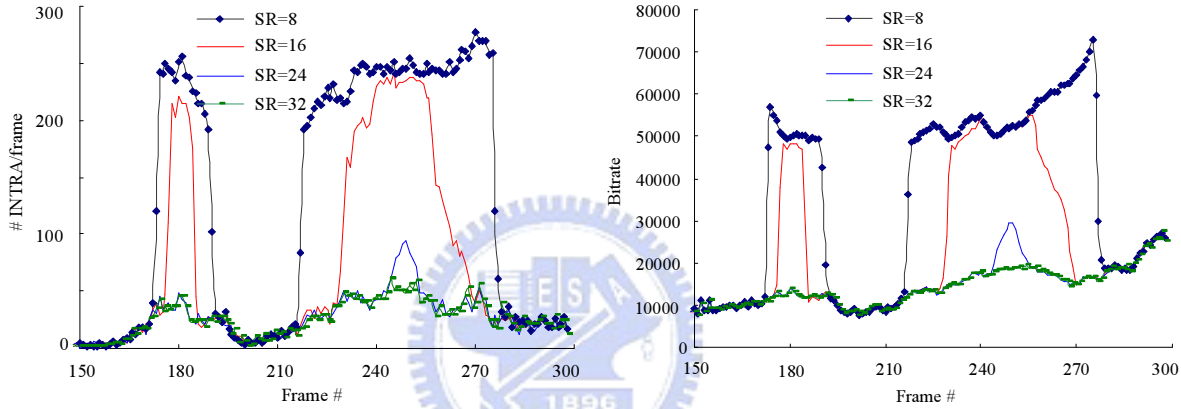
3.3.2 Proposed Boundary Prediction Method

To resolve the metric insufficiency as well as the prediction fidelity issue in pervious works, this section will present a robust fast search algorithm using dynamic boundary prediction. The boundary prediction adopts adaptive search center and area not only effectively improving the R-D performance but significantly eliminating computational complexity, which involves two main design inspections, 1) *the matching center* and 2) *the matching boundary*.



(a) Frame INTRAs (b) Frame bitrate

Figure 3.3: Frame INTRAs versus frame coding rate (**foreman**, CIF@30fps, QP = 36)



(a) Frame INTRAs (b) Frame bitrate

Figure 3.4: Frame INTRAs versus frame coding rate (**stefan**, CIF@30fps, QP = 36)

The Matching Center

Almost all fast algorithms for VLSI implementation use vector $(0, 0)$ as search center since it maximizes the data reuse efficiency in reference region. Most studies believe that data-reuse efficiency and I/O bandwidth requirement are two sufficient measurements in power dissipation. However, this design assumptions is somehow fallacious, especially for adaptive boundary search.[3] Moreover, because scene may changes rapidly in extremely active motion sequence, using constant search center structures will greatly degrade the reliability of motion-compensated prediction. To be demonstrated, two active CIF test sequences, foreman and stefan, are discovered as in Figure 3.3 and Figure 3.4. Figure 3.3 shows the relationship between frame INTRAs and frame bitrate of encoding CIF test sequence "foreman" within frame #150 to #299 and motion search using (0,0) as center with different search ranges. Figure 3.3(a) depicts the number of INTRA coding of each frame, and Figure 3.3(b) shows corresponded bitrate distribution during coding these P-

[3]Refer to section 4.3 (pp. 58) and 6.2 (pp. 58) for detail discussion of hardware design metrics

frames. Figure 3.4 illustrates another case of test sequence "stefan". As the analyses, the relationship between INTRAs and bitrate in these cases are strongly correlated, in particular within those extremely active frames. The reason is quite simple, the efficiency of video coding largely determined on fidelity of motion compensation, or simply, effectiveness of matching area. To avoid the defects of fixed matching area on power and R-D performance, a judicious algorithm clearly uses an adaptive matching area scheme rather than consideration of reference region reuse.

The Matching Boundary

The fast algorithms using dynamic search range have been proposed in [35], [36] and [37] based on the fact of the motion vectors of adjacent blocks usually correlated in time or in space. The best matching displacement typically lies inside the boundaries that adapts by neighboring correlation, similar to the motion predicting. Therefore adapting the search range by neighboring correlation, the fast block-matching algorithm will efficiently eliminate the unnecessary candidate blocks to be examined and thus greatly saves power.

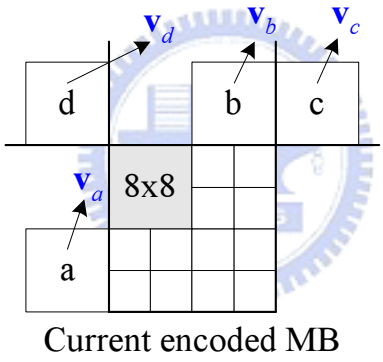


Figure 3.5: Illustration of search boundary prediction using neighboring vectors

Since motion of objects is usually regular in direction or in magnitude for either temporal or spatial domain, this motion correlations facilitate us to exploit computational redundancy in motion search. One simple method to restrict the computation load for full search BMA is to accurately predict the effective search window which contains all possible candidates associated with the displacement relative to the current block. In the proposed method, the search boundaries are predicted adaptively by using neighboring correlations. Four key displacements for determining the boundaries of search window are calculated by the following:

$$\begin{aligned} p_{x,max} &= \max\{v_{x,a}, v_{x,b}, v_{x,c}, v_{x,d}\} \\ p_{x,min} &= \min\{v_{x,a}, v_{x,b}, v_{x,c}, v_{x,d}\} \\ p_{y,max} &= \max\{v_{y,a}, v_{y,b}, v_{y,c}, v_{y,d}\} \\ p_{y,min} &= \min\{v_{y,a}, v_{y,b}, v_{y,c}, v_{y,d}\} \end{aligned} \tag{3.19}$$

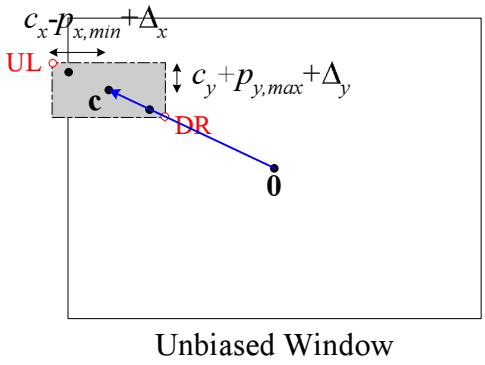


Figure 3.6: Illustration of proposed BMA using dynamic boundary prediction

where blocks a, b, c, and d are quasi-coded neighboring blocks with each size 8×8 relative to the location of current MB. The geometric relationship between current MB and these neighboring blocks are shown as Figure 3.5.

By analysis, we find that the boundaries directly predicted from these vectors are too restricted to involve most best fitting displacements inside the window. Therefore, two search increments, Δ_x and Δ_y are introduced to appropriately adjust predicted boundaries. The two increments are derived as the function of the displacement of macroblock predictor and vector differences between $\mathbf{p}_{max}$ and $\mathbf{p}_{min}$. The search boundaries of the proposed fast algorithm then become

$$\begin{aligned} \mathbf{v}_{\mathrm{UL}} &= (p_{x,min} - \Delta_x, p_{y,max} + \Delta_x) \\ \mathbf{v}_{\mathrm{DR}} &= (p_{x,max} + \Delta_x, p_{y,min} - \Delta_y) \end{aligned} \tag{3.20}$$

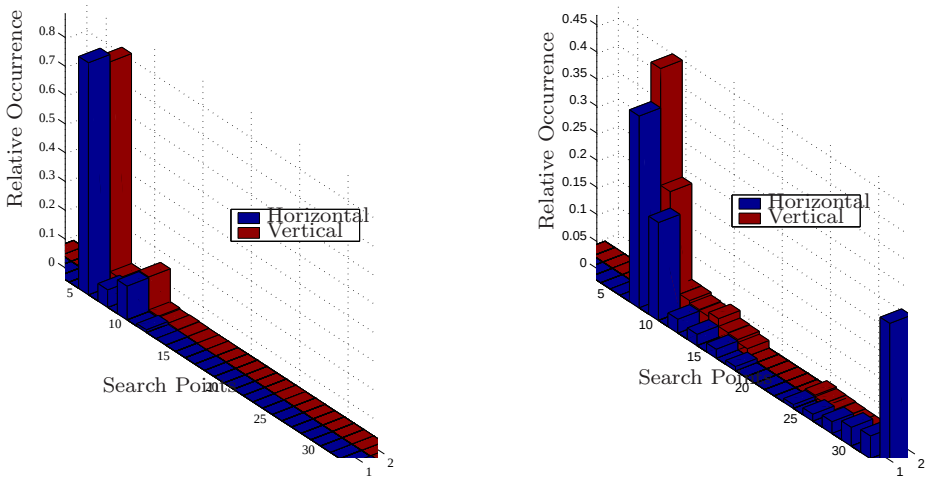
Figure 3.6 depicts the search window of the proposed dynamic boundary BMA.

For those boundary macrobloks, however, prediction formula stated as Eq. (3.20) has not enough spatial information to deal with. Hence, a constant search range of 4 or 8 or 16 is then conditionally assigned according to the number of INTRA of pervious frame slice to accurately establish spatial predictors.

3.3.3 Simulation and Comparison

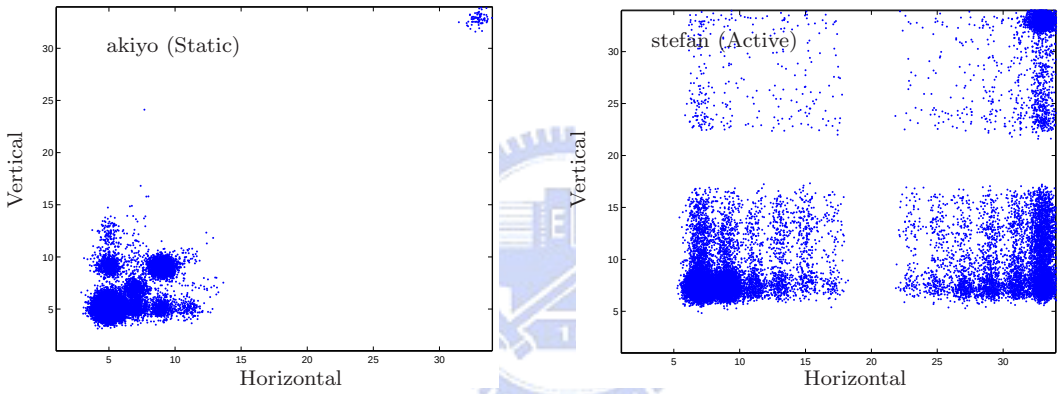
Figure 3.7(a) and 3.7(b) shows two examples of boundary distance histogram of coding static CIF sequence "akiyo" and active sequence "stefan", respectively. Note that the *boundary distance* is the number of required search point in horizontal or in hermetical. Even in extremely active sequence, the major portion of relative occurrence is located within distance of 5 to 10 pixels, clearly significantly reducing computational redundancy by estimating unnecessary candidate blocks. Figure 3.8(a) and 3.8(b) exhibit two correlations between the boundary distance of x and of y; totally $60 \times 396 = 23760$ MBs were tested for each sequence.

Since the expected value of operation rate as Eq. (3.17) is proportional to average



(a) Test sequence: akiyo (b) Test sequence: stefan

Figure 3.7: Relative Occurrence versus boundary distance, CIF@30fps, QP={36, 42}



(a) Test sequence: akiyo (b) Test sequence: stefan

Figure 3.8: Boundary distance distribution, CIF@30fps, QP={36, 42}

number of search point (SP) as follows

$$\mathrm{SP} = (p_{x,max} - p_{x,min} + 2\Delta_x + 1) \times (p_{y,max} - p_{y,min} + 2\Delta_y + 1) \tag{3.21}$$

and, memory bandwidth $\mathrm{BW_{mem}}$ as Eq. (3.18) can be roughly represented as search area (SA), the pixel count within the area to be searched, as Eq. (3.22).

$$\mathrm{SA} = (p_{x,max} - p_{x,min} + 2\Delta_x + N) \times (p_{y,max} - p_{y,min} + 2\Delta_y + N) \tag{3.22}$$

In the performance comparisons, we use SP and SA two metrics to simply represent Op and $\mathrm{BW_{mem}}$ for complexity and power measurement. Table 3.2 compares the proposed algorithm with traditional full search and JM86 codec using fixed search range of 16 pixels in rate-distortion performance and complexity reduction, respectively. Simulation shows that the proposed boundary prediction technique significantly improves coding efficiency and eliminates the required size of search area and search point. Our fast

Table 3.3: Rate-distortion performance and complexity reduction, compared to full search of SR = 16 (CIF@30fps, QP = {30, 32, 36, 42})

			ΔBitrate	ΔPSNR	Average	BW_{mem}	Arithmetic Op.
			(%)	(dB)	Search point	Reduction (%)	
Proposed	akiyo	CIF	−1.43%	0.032	27.31	81.35%	97.49%
	M&D	CIF	−0.34%	0.042	32.13	82.16%	97.05%
	foreman (static)	CIF	−3.25%	0.031	52.19	74.33%	95.21%
	foreman (active)	CIF	−35.30%	−0.128	166.88	56.45%	84.68%
	mobile	CIF	−1.49%	−0.015	46.98	73.90%	95.69%
	stefan	CIF	−52.60%	0.120	173.27	59.12%	84.09%
	Average		−15.73%	0.014	83.12	71.22%	92.37%
JM86	akiyo	CIF	−2.32%	0.116			
	M&D	CIF	−2.02%	0.110			
	foreman (static)	CIF	−12.03%	0.129			
	foreman (active)	CIF	−40.57%	−0.142			
	mobile	CIF	−3.05%	0.080			
	stefan	CIF	−57.03%	0.221			
	Average		−19.50%	0.086	1089	0%	0%

† Proposed: simplified predictions ($16 \times 16 \sim 8 \times 8$), TBs = 3bits, without SKIP detection

full search scheme averagely improves 15.7% of bitrate effeminacy (maximum 52.6%), while the required operation rate and memory bandwidth achieves 92.4% and 71.2% of reduction, respectively.[4]

3.3.4 Summary of Proposed Method

The characteristics of proposed boundary prediction algorithm can be concluded as

1. Dynamic search boundary

$$v_{x,max} = p_{x,max} + \Delta_x, \quad v_{x,min} = p_{x,min} - \Delta_x$$

$$v_{y,max} = p_{y,max} + \Delta_x, \quad v_{y,min} = p_{y,min} - \Delta_x$$

2. Adaptive search increment Δ_i: search increment Δ_i is adaptively adjusted according to the locality of window center $\mathbf{c}$ and predicted displacements $\mathbf{p}_{max}, \mathbf{p}_{min}$, *i.e.*,

$$\Delta_x = f_{\Delta_x}(c_x, p_{max,x} - p_{min,x}), \quad \Delta_y = f_{\Delta_x}(c_y, p_{max,y} - p_{min,y})$$

which effectually compensates prediction sufficiency of neighboring side information.

[4] BW_{mem} using SA is pessimistic to real case

3. Full search scheme with adaptive search boundary provides the significant superiority in motion-compensated quality and maximizing arithmetic efficiency.
4. Full search within rectangular search window improves memory bandwidth usage, since it inherently exploit data-reuse in inter-candidate; beside, exhaustive search within a search range avoids matching aliasing due to search candidate subsampling.
5. By simulation, the proposed algorithm improves maximum 52.6% the coding bitrate efficiency, and averagely eliminates 92.4% the arithmetic operations and 71.2% of memory bandwidth compared to traditional full search scheme.

3.4 Bit truncation Technique and Predictions

This section determines the resolution parameter and block size of motion prediction for proposed block-matching algorithm.

Bit Truncation Technique

For portable video applications, there are several design spaces that need to be traded off: bit rate, system performance, circuit complexity, and power consumption. If we implement an arithmetic algorithm using full data resolution, it usually has better responded performance (*e.g.* superior picture quality or smaller bit-rate). For example, as the accurate DCT/quantization is applied in the motion-compensated prediction (MCP) loop, which gives precisely transformed coefficients, coder can reconstruct more accurately the frame predictions and thus reduces the bit-stream size or achieves better picture quality. However, an exact arithmetic needs more computation, which means more demand of implementation complexity and power. Since the relation is typically nonlinear between system performance and implemented accuracy, it could benefit to trade off the arithmetic resolution for low power purposes without an apparent impact on the coding quality.

Among all functional block in an AVC codec, motion estimation is most computational intensive. For MCP, the image frame is segmented into the macroblock and the motion vector is then estimated for each macroblock. To find the motion vector, each predicted block in the current frame is matched with reference data in the previous frame within a search area. Some criterion used to identify the motion displacement have been stated in section 3.2. In our study, we assume that SAD-based criteria is applied as Eq. (3.14) for distortion measuring. Since arithmetic parallelism is used to suppress switching power, the core design of motion estimation usually requires a multiple of block size N squared absolute difference (AD) logics. For instance, it requires 1×256 (1×16^2) AD arithmetic elements in a typical 2-dimensional array structure. Such a large number of AD may therefore take up more than 50% of the total power of the entire motion estimation. In proposed implementation, the AD array averagely consumes 54% of total power dissipa-

Table 3.4: Comparison of power reduction/PE (CIF@30fps, 30frames, QP = {36, 42})

Reduced-bit	0 bit (μW)	2 bits(%)	4 bits(%)
akyio	4.80μW	52.59%	79.72%
foreman	7.40μW	43.08%	68.25%
mobile	7.21μW	41.56%	64.20%
stefan	7.85μW	36.23%	57.82%
Average (%)	—	43.37%	67.50%

Table 3.5: Comparison of rate-distortion (CIF@30fps, 150frames, QP = {30, 32, 36, 42})

Truncated-bit		2 bits		3 bits		4 bits		7 bits	
		ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)
akiyo	CIF	0.23%	0.014	1.00%	0.027	2.74%	0.022	37.47%	−0.367
M&D	CIF	0.36%	0.002	1.29%	−0.040	4.11%	−0.083	40.89%	−0.405
foreman	CIF	0.45%	−0.007	2.15%	−0.019	5.35%	−0.056	70.65%	−0.309
mobile	CIF	0.15%	−0.003	0.51%	−0.004	1.87%	−0.020	49.98%	−0.289
stefan	CIF	0.17%	−0.003	0.74%	−0.014	1.91%	−0.028	20.49%	−0.189
Average		0.27%	0.001	1.14%	−0.010	3.20%	−0.033	43.90%	−0.312

tion, even though the specialized AD was used rather than direct design. The specialized AD arithmetic for video motion estimation well be stated in section 4.2.

In order to restrict the power consumption due to full AD arithmetic parallelism, one simple and effective method is using the bit-truncation technique, in which the bit precision is adjusted such that the bit-width, the dynamic power, and the hardware cost of the ME modules are minimized with the performance meeting the specification. Bit-truncation technique restrict the dynamic power by the manner of reducing the switching activity in a CMOS circuit. In fact, digital value on the arithmetic circuit is non-uniformly distributed. In arithmetic, the Most Significant Bit (MSB) have much lower transition probability than the Least Significant Bit (LSB). Because only a few LSBs actually transit on the circuit in each clock period, a large number of the MSBs maintain the data. Such switching characteristic that the transition probability of LSB and MSB distributes asymmetrically usually refers to as the "dual-bit model" [38].

Power and R-D analyses Some papers [39,40] have presented motion estimation design in use of data width truncation. However, as we concern the rate-distortion as primary design metric, these studies did not clarify the relationship between R-D drop against pixel depth. To simply demonstrate it, we examined power reduction as well

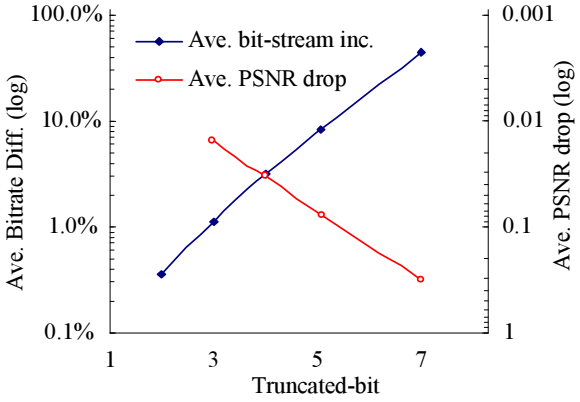


Figure 3.9: Log plot of Table 3.5

Table 3.6: MODE comparison between full-JM 8.6 and basic mode (QP=30, 32, 36, 42)

	Compared to full prediction	
	ΔBitrate (%)	ΔPSNR (dB)
akiyo	0.598%	−0.016
foreman	1.351%	−0.030
mobile	1.182%	−0.059
stefan	2.880%	−0.062
Average	1.503%	−0.042

as bitrate and reconstructed-frame PSNR drop in different data resolutions. The comparisons are exhibited in Table 3.4, Table 3.5 and Figure 3.4, respectively. Based on the simulations, we therefore select 3 bits of pixel width to truncate, which averagely increases 1.14% the bitrate and degrades frame quality nearly −0.01 dB.

Variable Size Prediction

By statics analysis, variable size prediction is greatly decided as macroblock modes, *e.g.* SKIP or 16×16, in low rate application. Although extending of prediction mode may efferently increase rate-distortion quality, it increases implementation complexity as well. To eliminate hardware cost and unnecessary arithmetic computation, we further analyze rate-distortion performance in different block prediction modes. Two prediction cases were simply examined of 1) $16 \times 16 \sim 8 \times 8$: namely basic prediction, 2) $16 \times 16 \sim 4 \times 4$: namely full prediction, with same simulation environment as data bit truncation. The rate-distortion difference between basic prediction and full prediction is summarized in Table 3.6. Simulation shows that basic prediction acceptably drops 1.5% coding efficiency and 0.04dB of frame quality in average. As stated performance comparison, we judiciously select more simpler basic mode as variable size prediction candidates.

Chapter 4

The Architectural

After system/algorithm development as pervious chapter, we proceed of advance in design abstraction of architectures for power-efficiency motion estimation in this chapter.

To deal with most arithmetic intensive SAD computation, section 4.1 theoretically investigates the array processor architecture through the well-known graphic design methodology. In section 4.2, orientating absolute difference arithmetic to motion estimation, a specialized processing element (PE) is then derived which significantly eliminates implementation cost of using the half the switching power and the area to conventional design. Since internal memory access usually consumes upmost operational power, section 4.3 exhibits the memory structuring methodology to minimize the access power due to reference pixel buffering in an original high-abstraction technology-dependent manner. After the optimization, we also mathematically prove a shortest distance coding scheme in section 4.4 to further reduce the power dissipation of address switching. In addition, an ultra cost-efficiency logic structure is presented to facilitate motion cost computation. The invention effectively improves motion prediction fidelity and rate-distortion metric which profits low rate motion estimation in the end of this chapter.

4.1 Design of Array Processor

In block-matching algorithm of video motion estimation, the SAD distortion measuring is always the most intensive. However, due to the arithmetic modularity and regularity, it can be separately implemented apart from the whole system of using array structures.

This section will explore the SAD measuring structural through the well-known systematic graphical representation. Consisting of the graphic methodology retrospection, structural investigation, and array architecture design, a superior study of power-efficiency array structure was performed for video motion estimation.

4.1.1 Graphic-based Design Methodology

Mapping DSP (digital signal processing) algorithm in to array processors through graphical representation was firstly introduced by Kung [41]. Graphical representation efficiently exploits the data flow properties and inherent parallelism of the DSP algorithm into array structures. It provides formal and powerful design methodology and insight into space and time tradeoffs.

More importantly, these representations judiciously bridge the gap between algorithmic description and specialized structure implementation in a technology-independent manner. The absolute measures of the performance metrics, area, speed, power, and roundoff noise are in general manufacturing technology dependent, on which the system is implemented. Structural design using graphical representation can appropriately select specific architecture without particularly dealing with physical technologies.

Canonical Graphical Representations

Three canonical mapping steps in the graphic approach consist of [41,42]:

1st step: Mapping algorithm to DG.
Exploit the data flow properties and inherent parallelism in algorithm by using dependence graph (DG). A DG is a directed graph that shows the dependence of the computations in a space-time algorithm. DG greatly affects the final array design; further modifications on the DG are often desirable in order to achieve a better array structure. In general, dependence graph is formally used for systolic array design, where the graph representations in lower abstraction can be derived from a single DG by exploiting the parallelism in different ways.

2nd step: Mapping DG to SFG.
Based on different mappings of the DG onto array structure, a number of signal-flow-graphs (SFGs) can be derived from the DG. A signal-flow-graph is graphical representation in a lower abstraction than DG, which is presented in nodes and directed edges. The nodes represent computations or tasks. A directed edge (j, k) denotes a branch originating form node j and terminating at node k. Since designate one processing element of each node in a DG leads to very inefficient, map the DG first to an intermediate SFG form to reduce the total PEs (processing elements) in array structure, and thus to improve the instinctive utilization. SFGs provide an abstract flowgraph representation of linear networks and have been extensively used in digital filter structure design and analysis of finite word-length effects [43]. In general, SFGs are only applicable to linear networks; they cannot be used to describe, for instance, multirate DSP system.

3rd step: Array processor design.

Realize the specialized space-time SFG into array processor in terms of SIMD (single instruction, multiple data), systolic, wavefront, or MIMD (multiple-instruction, multiple-data) array structure.

Notice that 1) the signal-flow-graph is exploited from DG and is closer to hardware level design, 2) in practice, DG is presented for systolic array design, and 3) the purpose of SFG is sued to analyze structural properties and to explore architectural alternatives using high-level transformation.

4.1.2 Array Design using Graphical Approach

Systolic Structure

Systolic structures (also referred to as systolic arrays) represent a network of PEs that rhythmically compute and pass data through the system. That PEs regularly pump data in and out features array structure modularity and regularity, which are important properties and advantages for VLSI design. The regular data flow driven by host device and sequentially pumped out is analogous the flow of blood through the heart, thus the name "systolic".

Systolic array are designed by using linear mapping techniques on regular dependence graphs. The edges in DGs represent precedence constraints. A dependence graph is said to be regular if the presence of an edge in a contain direction at any node represents presence of an edge in the same direction at all other nodes. The systolic mapping transforms a space representation in DG to a space-time representation where each node is mapped to a certain processing element and is scheduled to a certain time instance. The time instance corresponded in a space-time representation is assigned without any arithmetic computation. Since the block matching algorithm used for motion estimation maintains space-time data flow modularity and regularity, this regular dependence property makes BMA easily implemented as systolic array structure.

Before the derivation, we define two elementary vectors for N-dimensional DG systolic design.

Projection vector (also called iteration vector), $\mathbf{p} = (p_1, p_2, \ldots, p_n)$

The design methodology maps an N-dimensional DG to a lower dimensional systolic architecture. Projection vector $\mathbf{p}$ maps DG from index space onto a lower dimensional lattice of points, known as the processor space. Two nodes that are displaced by $\mathbf{p}$ or multiples of $\mathbf{p}$ are executed by the same processor.

The scheduling vector, $\mathbf{s} = (s_1, s_2, \ldots, s_n)$

Any node with N-dimensional index $\mathbf{I} = (i, j, k, \ldots)$ would be executed at time, $\langle \mathbf{s}, \mathbf{I} \rangle$, where $\langle , \rangle$ denotes inner product in $\mathbb{R}^M$ space. The scheduling vector specifies the sequence of the operations in all the processing elements, and represents a

mapping form the N-dimensional index space of the DG onto 1-dimension schedule (time) space. A linear schedule is based on a set of parallel and uniformly spaced *equitemporal hyperplanes* in the DG.

Derivation of Array Processors

As the canonical representations retrospected previously, we will now focus on representing the space-time block matching algorithm into the dependence graph. The SAD distortion term in proposed block-matching algorithm can be rewritten as follows[1]

$$\mathrm{SAD}_{\mathbf{D}=(D_x,D_y)} = \sum_{M}\sum_{N} |x_{ij} - y_{i+(SR_{0,y}-D_y),j+(SR_{0,x}+D_x)}| \tag{4.1}$$

where integer vector $\mathbf{D}$, the motion vector difference (MVD), indicates the locality of current candidate block ($M \times N$ pixels) relative to integer macroblock predictor $\mathbf{P}_{16\times16}$, and its two elements x and y lie inside

$$\begin{aligned} v_{x,min} - P_{16\times16,x} \le D_x \le v_{x,max} - P_{16\times16,y} \\ v_{y,min} - p_{16\times16,y} \le D_y \le v_{y,max} - P_{16\times16,y} \end{aligned} \tag{4.2}$$

note that $\mathbf{P}_{16\times16} = (P_{16\times16,x}, P_{16\times16,y})$ and assuming $v_{x,min}$, $v_{x,max}$ are two integer search boundaries in horizontal; $v_{y,min}$, $v_{y,max}$ are another two vertical boundaries. $SR_{0,x}$ and $SR_{0,y}$ present one side search ranges of $\mathbf{D}$ displacing, defined as

$$\begin{aligned} SR_{0,x} = v_{x,min} - P_{16\times16,x} \\ SR_{0,y} = v_{y,max} - P_{16\times16,y} \end{aligned} \tag{4.3}$$

Figure 4.1 geometrically expresses the block-matching algorithm in Eq. (4.1).

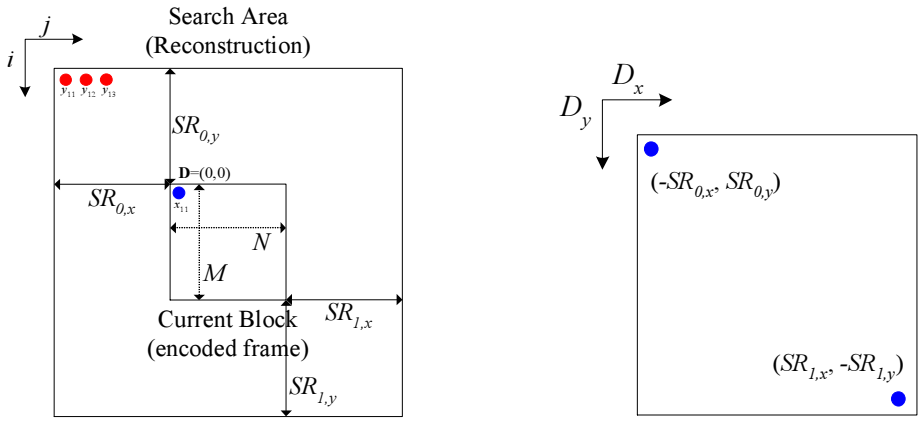
For the given MV displacing, the larger block size SAD can be combined with smaller block size SADs. For instance, two 8×16 blocks or 16×8 blocks sum the SAD value of 16×16 block, and each block SAD is the combination of the elementary SADs, thus referred to as intra-candidate data-reuse. By intra-candidate data-reuse, we can simplify the array design for block matching into its elementary size. The present derivation follows N and M in SAD formula Eq. (4.1) both 4 ($N = 4, M = 4$).

We now introduce two *shifted matching index* k and l

$$\begin{aligned} k = SR_{0,y} - D_y + 1, \text{ where } k = 1, 2, \ldots, SR_{0,y} + SR_{1,y} + 1 \\ l = SR_{0,x} + D_x + 1, \text{ where } l = 1, 2, \ldots, SR_{0,x} + SR_{1,x} + 1 \end{aligned} \tag{4.4}$$

It should be recognized that indexes k an l are always positive integers. Clearly, the block-matching of Eq. (4.1) can be represented into a four-dimensional index space over i,

[1]The proposed BMA has been described in 3.2 (pp. 27)



(a) Search area (b) Search window

Figure 4.1: Motion Search using BMA: *search area* versus *search window*

j, k, and l. In general, the BMA is decomposed into two parts each over two-dimensional index space. The first is spawn by the *local indexes* i and j and consists of the additional of the sum $\text{SAD}_{\mathbf{D}}$. In the rest, which is defined over global indexes k and l, it performs the displacing for minimum cost searching. Thus the explanation of the mapping procedure can be easily supplied by a graphical description. The addition of $\text{SAD}_{\mathbf{D}}$ starts with the index i and is continued over the index j given in fixed displacing indexes k and l. The associativity of additions permits several different DGs in its space dimension. A simple example of three-dimensional DG in the i, j, and k space is presented in Figure 4.2.

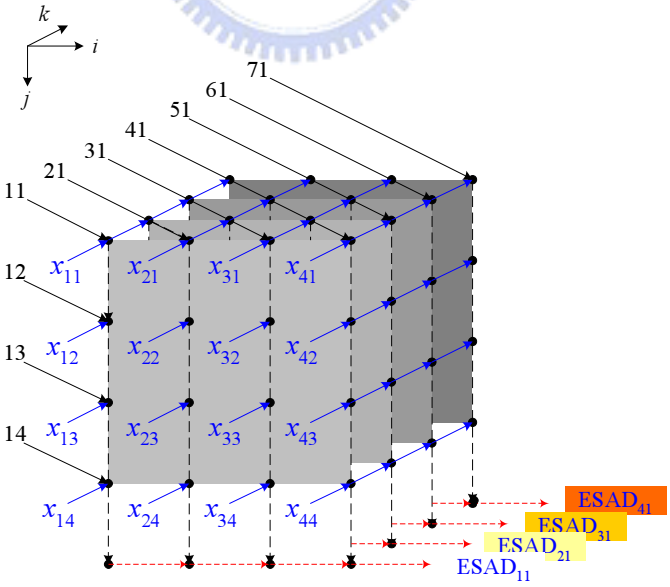
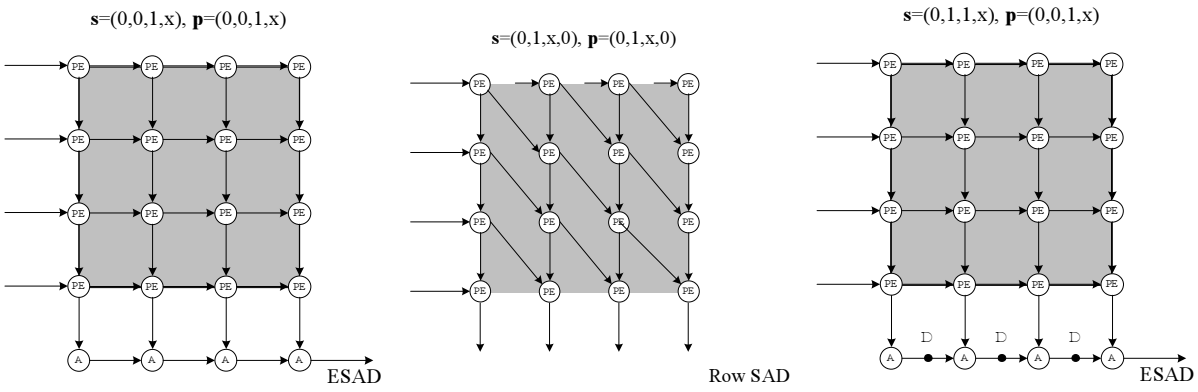


Figure 4.2: A simple DG of computations and data dependencies for 4×4 block-size BMA (i, j rotated)

Figure 4.2 illustrates that total four candidates are matched in y dimension, and each candidate is 4×4 pixels. Each slice, indexed by k, correspond to the computation



(a) SFG-(a) (b) SFG-(b) (c) SFG-(c)

Figure 4.3: Mapped signal flow graph with different **s**, **p**

of elementary SAD (ESAD) associated with displacing (D_x, D_y). The associativity of project vector (**p**) and scheduling (**s**) corresponds several different SFGs in its projection dimension. In the following, we constrain the array structure from the four-indexed dimension DG projected to the two-dimension SFGs, and three derived structures are then compared for low power motion estimation.

Although graphical representation provides formal and powerful array design methodology, it surly exists some inherent limitations. The translated structures can be derived in a specific problem by selecting different projection and scheduling vectors, but not any combination of $(\mathbf{p}, \mathbf{s})$ is validated. For instance, vectors must satisfy some feasibility constraints, such as $\langle \mathbf{p}, \mathbf{s} \rangle \neq 0$, obviously. Moreover, given two any effective vectors, the translated structure is not even practical for a specific application. The following **p** and **s** assignments are typically applied to the power-efficiency two-dimensional systolic structures. Notice that the indexes presented in a four-dimensional vector are sequentially i, j, k, and l.

SFG-(a) : $\mathbf{s} = (0, 0, 1, \times)$, $\mathbf{p} = (0, 0, 1, \times)$

The mapping systolic SFG of the projection and scheduling vectors is depicted in Figure 4.3(a), where "×" denotes *projection irrelevant index*, on which treats the precedence in the space as invisible part in DG, and PE (processing element) operates the absolute different arithmetic. All PEs in SFG-(a) process candidate matching of identical displacing in a time slice, and the maximum PE utilization can achieve to 100%. The representative structure designs have been presented by Chen *et al.,*. [3,31].

SFG-(b) : $\mathbf{s} = (0, 1, \times, 0)$, $\mathbf{p} = (0, 1, \times, 0)$

The mapping systolic SFG of the projection and scheduling vectors is depicted in Figure 4.3(b). All PEs in SFG-(b) parallel row processing horizontally of four successive candidates in a time slice; also, the maximum PE utilization can achieve to 100%. The representative structures designs have been presented by Lin *et al.,*.

[33,44,45].

SFG-(c) : $\mathbf{s} = (0, 1, 1, \times)$, $\mathbf{p} = (0, 0, 1, \times)$

The mapping systolic SFG of the projection and scheduling vectors is depicted in Figure 4.3(c).All PEs in SFG-(c) parallel row processing of corresponded locality during consecutive four candidates in a time slice. However, that PE utilization can't reach 100% disadvantages its practicability. The representative structures designs have been presented by Huang *et al.,*. [3,46].

The PE utilization somehow reflects the proportion of effective computation cycles to the requirement. The higher utilization usually leads to higher data-reuse efficiency as well as lower PE power dissipation. Since structures SFG-(a) or SFG-(b) in use of some specific search patterns may achieve 100% of PE utilization, they were more favorite than SFG-(c) in pervious studies of low power motion estimation. However, we still apply SFG-(c) as array structure rather than preceding two designs, due to the following statements:

1. Either structure (a) or (b) with snake-like search strategy will reduce the effective computing cycles, and reaches 100% of PE utilization. Chen *et al.* [47] proposed an original two-dimensional random access memory scheme dedicated to such search pattern. Since there are minimum 16 times of memory assess in an effective calculating cycle of said method, it apparently burdens memory access; thus wastes power.[2]

2. A well-known computation reduction technique used in software realization, PDE (partial distortion elimination) is intrinsically applied into structure (b) to terminate the unnecessary calculating by comparing the minimum SAD and partial accumulation. To support the intra-candidate data-reuse, the SADs of elementary block are reused for the upper-layer blocks, however the PDE doesn't suit for utilizing the intra-candidate data-reuse. Besides, to achieve efficient power reduction, the array structure has to implementation with large degrees of parallelism; therefore increases the overhead on duplications of motion cost and MV decision unit.

3. The PE utilization of structure (c) is lower than that of the other structures, but the low power techniques, such as operation isolation, can be adopted to the data input of PE to eliminate the improper arithmetic calculations. As the applying of operation isolating, the power consumption of PE becomes obscure to the PE utilization. Furthermore, the structure of SFG-(c) has the advantage on simpler logic control and implementation, more suitable to systolic array architecture, and we have proven that utilizing with SFG-(c) are efficient in all concerned design spaces, power, area, and timing.

[2]Will discuss in 4.3 (pp. 58)

4.1.3 The Proposed Array Processor

The proposed systolic array processor (SAP) based on the SFG-(c) structure is shown in Figure 4.4. The signal "data_valid" increases effective PE utilization to 100% by PE and

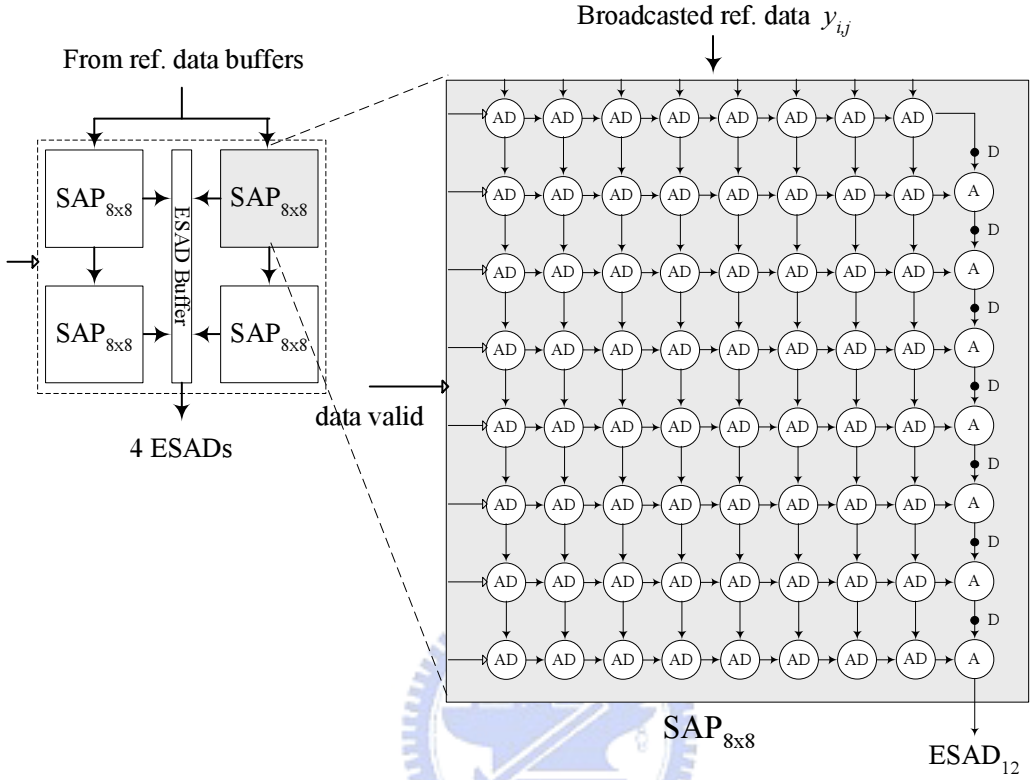


Figure 4.4: Variable size systolic array processor with ESAD 8×8

adder isolating, and enables data registers "D" and "ESAD Buffer" to eliminate unnecessary logic switching and thus to reduce the dynamic power. The ESAD buffer is consisted of register chains, which sequentially propagates the valid ESAD values generated from contiguous four elementary systolic array processors. Notice that we have restricted the ESAD size as 8×8 pixels.[3]

4.2 The Absolute Difference Logic

Section 4.1 has demonstrated systolic array processor design using graphic-based methodology. To increase data-reuse efficiency, a large number of processing elements of absolute difference were fully paralleled, and hence the PEs demand major proportion of area and power. The arithmetic design of absolute subtracted value becomes essential key component for low power motion estimation. Several equivalence designs were proposed based on the succinct logic rather than direct arithmetic [48–51], but they less took into consideration characteristic features of motion-compensated video coding. To exploit such

[3]Prediction block reduction has been stated in 3.4 (pp. 42)

features, this section illustrates a specialized absolute-difference arithmetic as well as its logic implementation.

4.2.1 Arithmetic Equivalence

Equivalency is derived for calculating of n-bits absolute subtracted value as following. Let integers X, Y are both within $[0, 2^n - 1]$

$$0 \leq X,\ Y \leq 2^n - 1, \text{ where } X, Y \in \mathbb{I} \tag{4.5}$$

and we then define the complementary operation, $\overline{X} = 2^n - 1 - X$, sum $S = X + \overline{Y}$, and carry signal C_o , as 4.6

$$C_o = \begin{cases} 1 & \text{if } S \geq 2^n \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

If the integer X is large than Y, then we will induce that

$$\begin{aligned} |X - Y| = X - Y &\geq 1 \\ &\Longleftrightarrow X - (2^n - 1 - \overline{Y} \geq 1 \\ &\Longleftrightarrow X + \overline{Y} \geq 2^n \\ &\Longleftrightarrow C_o = 1 \end{aligned} \tag{4.7}$$

In the case, the computing of S carries ($C_o = 1$), and meanwhile the absolute subtracted value $|X - Y|$ can be expressed as

$$\begin{aligned} |X - Y| &= X - Y \\ &= X + \overline{Y} + 1 - 2^n \\ &= (S + C_o) \bmod 2^n \end{aligned} \tag{4.8}$$

where **mod** represents modulo operation (remainder in division). Therefore for $X \leq Y$, if and only if computing of S not carries ($C_o = 0$); meanwhile the absolute subtracted value $|X - Y|$ becomes

$$\begin{aligned} |X - Y| &= Y - X \\ &= 2^n - 1 - \overline{Y} - X \\ &= 2^n - 1 - S \\ &= \overline{S} \end{aligned} \tag{4.9}$$

As we substitute the logic equivalency of Eq. (4.6) into Eq. (4.9), then the absolute difference arithmetic can be represented by

$$|X - Y| = \begin{cases} (S + C_o) \bmod 2^n & C_o = 1 \\ \overline{(S + C_o) \bmod 2^n} & C_0 = 0 \end{cases} \tag{4.10}$$

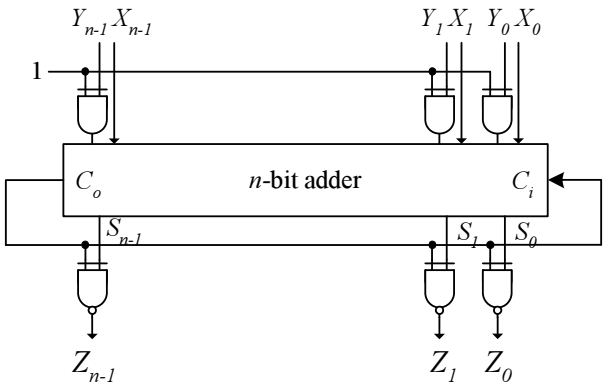


Figure 4.5: An equivalency arithmetic design of absolute difference value

The pervious art [48] alternatively performed " $(X + \overline{Y} + 1)$ mod 2^n" or "$(Y + \overline{X} + 1)$ mod 2^n" by the identity of Eq. (4.6). Operand inverting accords to the state of generate carry, and thus the conditional increment for 2's complement can be substituted by a constant for SIMD or MIMD vector-based calculation. However, demand of two n-bits comparators and two sets of n-bitwise XOR gates prior to sum arithmetic substantially increases the area cost.

Figure 4.5 shows another arithmetic design corresponded to the identity Eq. (4.10), which uses less area and power than [48]. Notice that $Z = |X - Y|$.

4.2.2 Orientated Arithmetic

Timing Loop Based on the design in Figure 4.5, however, a *timing path* exists between adder carry output C_o and carry input C_i. For logic design, timing path should be best avoided for qualified static timing analysis (STA) and for qualified automatic synthesis, regardless of that the path is actually true. In this case, since carry C_o only depends on its input operands, it is clearly irrelevant to carry in C_i, and therefore feedback from C_i to C_o is not a true path.

To escape around, the US patent [49] presented a latch-based architecture, pre-storing carry in latch during the front half clock cycle. Two other similar approaches were disclosed in patents [50] and [51], which combined carry C_o with sum S by post-multiplexer processing.

Such prior arts efficiently calculates absolute subtracted value by applying the logic equivalence of Eq. (4.6) either in computing time or in gate area. However, they did not take into consideration of characteristics of video motion estimation: 1) extensive parallelism and 2) less sensitivity on arithmetic accuracy for block matching algorithm. Furthermore, since these architectures are all operand-inverted, it comes gate area and signal switching overhead by high PEs parallelism, and arithmetic equivalency may restrict design flexibility.

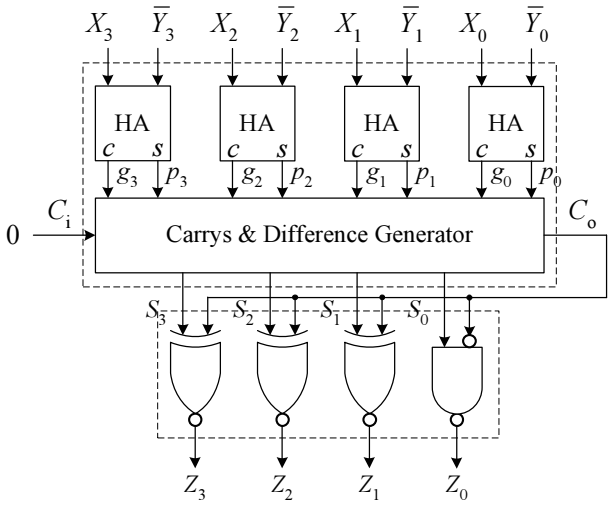


Figure 4.6: An embodiment of proposed arithmetic design with 4 bit-length and 1 bit carry reservation, $(n, p) = (4, 1)$

To improve the architecture suited for block matching algorithm, the proposed logic design has two superior modifications:

- distributive: remove operand-inverted and pre-process inverts during reference buffer access, and thus eliminate inverted switching redundancy.
- non-full propagated carry: break the full carry path into r-bits reservation, and thus avoid unnecessary logic transitions.

The truth table for calculating of absolute subtracted value Z is shown in Table 4.1, where r denotes the number of carry reservation bits.

Table 4.1: The truth table of carry logic

C_o	$S' = \{S_{r-1}, S_{r-2}, \ldots, S_0\}$	$Z' = \{Z_{r-1}, Z_{r-2}, \ldots, Z_0\}$
0	S'	$\overline{S'}$
1	$S' \neq 2^r - 1$	$S' + 1$
	$S' = 2^r - 1$	S'

Figure 4.6 depicts an embodiment design using CLA (carry look ahead) structure with design parameters $(n, r) = (4, 1)$, where acronym HA denotes "half adder", and *carry generate* and *carry propagate* logics g_i and p_i in the adder are

$$\begin{aligned} g_i &= X_i \bullet \overline{Y_i}, \quad \text{generate} \\ p_i &= X_i \oplus \overline{Y_i}, \quad \text{propagate} \end{aligned} \tag{4.11}$$

Logic S presents sum of operands X and $\overline{Y}$ (complementary), given as

$$\begin{aligned} S_0 &= g_0 \\ S_1 &= p_1 \oplus C_0 \\ S_2 &= p_2 \oplus C_1 \\ S_3 &= p_3 \oplus C_2 \end{aligned} \tag{4.12}$$

and lock-ahead carry C_i

$$\begin{aligned} C_0 &= g_0 \\ C_1 &= g_1 + p_1 \bullet g_0 \\ C_2 &= g_2 + p_2 \bullet g_1 + p_2 \bullet p_1 \bullet g_0 \\ C_3 &= g_3 + p_3 \bullet g_2 + p_3 \bullet p_2 \bullet g_1 + p_3 \bullet p_2 \bullet p_1 \bullet g_0 \end{aligned} \tag{4.13}$$

Simply, the adder out C_o is looked ahead by carry C_3, and the absolute subtracted value Z follows logics that has been described in Table 4.1.

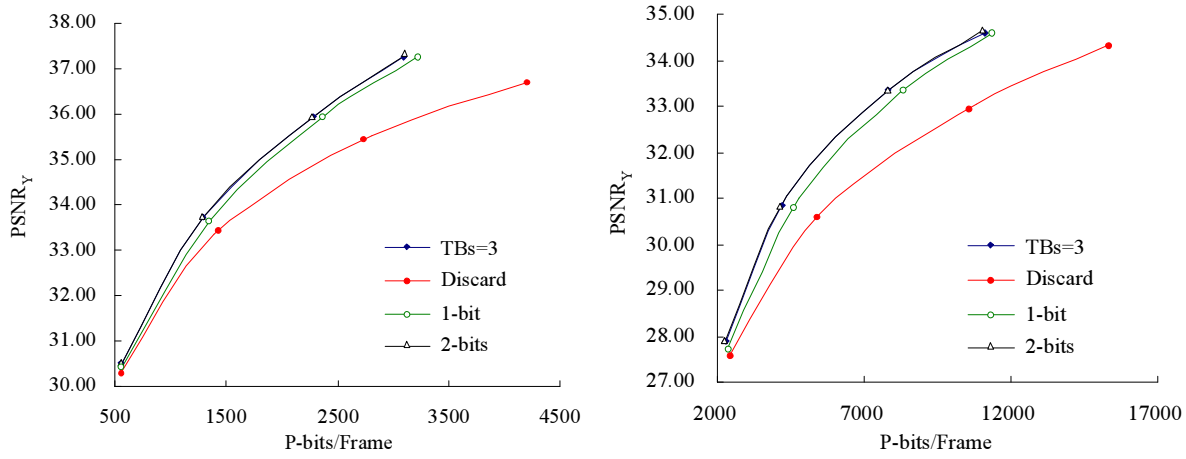
4.2.3 Determination of the Reservation

Due to carry's reservation, the arithmetic approximation may impact the accuracy of motion displacement, and may alter MCP performance, dependently on r, the number of bit of carry reservation.

Table 4.2: Coding rate and distortion comparisons of $r = 0, 1, 2$ with respect to truncation bits = 3 (TBS = 3, $n = 5$), CIF@30fps 300frames

		Discard ($r = 0$)		1-bit ($r = 1$)		2-bits ($r = 2$)	
		ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)
akiyo	CIF	22.845%	−0.258	2.916%	0.020	0.187%	−0.019
m&d	CIF	15.425%	−0.409	2.539%	−0.048	0.401%	0.013
foreman	CIF	27.177%	−0.320	5.527%	−0.062	0.365%	0.007
mobile	CIF	15.026%	−0.056	4.115%	0.015	0.586%	0.013
stefan	CIF	12.646%	−0.227	4.268%	−0.058	0.951%	−0.005

Table 4.2, and Figure 4.7 demonstrate the rate-distortion alteration in terms of r from 0 (discard case) to 2 bits at CIF@30pfs, 300 frames, and QP = {30, 32, 36, 42} tests. Since truncation technique of 3 bits is used, the analyses show that 2-bit reservation has almost performed identical prediction capability to that of full bit propagation ($n = 5$).



(a) Test sequence: mother_daughter

(b) Test sequence: foreman

Figure 4.7: Rate-distortion performance comparisons in $r = 0, 1, 2$ with respect to truncation bits = 3 (TBS = 3, $n = 5$), CIF@30fps 300frames

4.2.4 The Comparisons

We have compared the proposed design with prior arts in logic area and power consumption using design parameters $(n, r) = (5, 2)$ for different QP = {36, 42}. Table 4.3 compares these implement metrics with that of traditions [48,50], at synthesis conditions of 50MHz clock frequency and logic critical path $T_{\mathrm{AD}} = 2ns$.

Table 4.3: The area and average power comparisons (Artisan/TSMC-CL013G, clock rate = 50MHz, $T_{\mathrm{AD}} = 2ns$, and QP = {36, 42})

TBs = 3	Cell Area/PE (μm^2)	Power/PE (μW)
Vassiliadis *et al.* [48]	1149.1	14.89
Kanoh's [50]	972.6	12.65
Proposed	607.6	7.27

Table 4.4 compares power and area reduced percentage with prior arts for different QPs and CIF test sequences. By experiments, the proposed design significantly saves up 51%, 47% and 42%, 37% on the power and the area over [48] and [50], respectively; meanwhile the coding quality in use of proposed absolute difference arithmetic is substantially persevered, as stated in Table 4.2 and Figure 4.7.

Table 4.4: The area and average power comparisons with different CIF test sequences (Artisan/TSMC-CL013G, clock rate = 50MHz, $T_{\mathrm{AD}} = 2ns$, and QP = {36, 42})

TBs = 3	Average Power (μW)			Power Reduction (%)		Area Reduction (%)	
	Proposed	[48]	[50]	wrt [48]	wrt [50]	wrt [48]	wrt [50]
akiyo	4.10	8.19	6.52	49.94%	37.14%		
foreman	6.45	12.95	10.45	50.20%	38.29%		
mobile	10.42	21.25	18.90	50.96%	44.86%		
stefan	8.10	17.16	14.71	52.82%	44.97%		
AVG	7.27	14.89	12.65	51.19%	42.54%	47.12%	37.53%

4.3 Optimization of Reference Buffer Structure

Fast search algorithm and data-reuse are two wide-used techniques of exploiting computational redundancy in motion estimation. In relative works, most studies focused on fast search algorithms due to their characteristically intuitive and simple to realize, but they are rarely hardware-oriented. For hardware realization, memory bandwidth as well as data-reuse efficiency becomes two upmost issues in low power motion estimation [27,31–34,52,53]. The primary thought is intuitive: the more degree of data-reuse efficiency (or less memory bandwidth), the less complexity or arithmetic, and thus the less the power dissipation. Based on the prior studies, the abstraction of data-reuse was categorized into three different levels in proceedings of block matching, reference region, inter-candidate, and intra-candidate [31]. These levels of abstraction have their own derivative implementations. For instance, [31,33] used snake-like search pattern with PDE (partial distortion elimination) to adopt inter-candidate data-reuse, and most designs paralleled processing of variable size block matching by reusing the subdivision SADs.

4.3.1 The Data-Reuse in Motion Estimation

On Reference Region

For the reference region, since it of adjacent MB or the frame area overlaps vertically or horizontally, we may only replenish non-overlapped reference region and bypass (reuse) the reduplication during search area pre-buffering. According to the locality of candidate matching, motion estimation typically uses local memory structure pre-loading reference pixels from the search area. Consider that memory access demands relatively intensive power density; In [53], Tuan *et al.* analyzed four fundamental data-reuse schemes on different levels of replenishing. One metric for evaluating memory access efficiency is

called *redundancy access* (RA) factor, addressing as follows

$$\mathrm{RA}_{\mathrm{Tuan}} = \frac{\text{total pixels replenished in local memory}}{\text{total pixels in one frame}} \tag{4.14}$$

The factor $\mathrm{RA}_{\mathrm{Tuan}}$ in Eq. (4.14) represents the average count per pixel during reference region pre-buffering, and apparently, RA = 1 for the complete data-reuse case.

Although unfolding the reference region can achieve the higher RA, most designs instead restrict the reference region to the search area by trade-off between the hardware complexity and data-reuse efficiency. It's obviously by displacing the search area with a constant value that maximizing RA can be attained for sequential coding, such as using original as search center. Because of its regularity and simplicity of unbiased displacing, most hardware designs are implemented on this structure. However, the fixed search range may substantially fail in motion-compensated prediction (MCP) since motion estimation misses the predictor while object spreads extremely; as the result, coder alternatively predicts as INTRA instead, and thus affects the rate-distortion efficiency. Such rate degeneracy has already stated in section 3.3. Hence, to maintain the prediction quality, designer has to expand the search range, but essentially increases the hardware complexity as well as returning memory bandwidth. In brief, distinct from conventional studies, an R-D qualified low power coder should not consider utilizing the data-reuse scheme on reference region.

On Inter-Candidate

Since the reference data between adjacent candidates greatly overlap, reusing the overlapped pixels for matching arithmetic can eliminate the memory unnecessary accesses and improve PE utilization [27]. It is hence so-called *inter-candidate* data-reuse. Therefore, during the block matching, the data-reuse scheme reasonably saves the memory and arithmetic power dissipation on the inter-candidate abstraction. In quantitative analysis, Chen *et al.* [31] similarly presented an RA metric for evaluating data-reuse efficiency by the proportion of reference pixels loaded from the local memory to the required minimum pixels, described as follows

$$\mathrm{RA}_{\mathrm{Chen}} = \frac{\text{total pixels loaded from the local memory in task}}{\text{minimum requirement}} \tag{4.15}$$

The minimum requirement is the required minimum pixels for searching all candidates. For instance, minimum requirement is 256 pixels for one candidate, and then 272 (256+16) pixels for two successive candidates. If $\mathrm{RA}_{\mathrm{Chen}}$ factor equals to one, it simply means that the total pixels loaded from the local memory for all candidates matching is exactly to it minimum requirement, and memory access has therefore the lowest redundancy. The factor $\mathrm{RA}_{\mathrm{Chen}}$ was presented to evaluate the degree of bandwidth redundancy within reference buffering. Because of heavy power demand on memory access, Chen [31] then

proposed a low bandwidth redundancy technique and corresponded low-latency ME architecture concentrated on inter-candidate data-reuse. Nevertheless, the basis of memory access power is rather intricate, incomprehensively over weighting data-reuse efficiency and memory bandwidth may lead a significant degeneration on power, area, and even system performance. This declaration will be demonstrated in the rest of discussion.

The basis formed the buffering power are several interactive factors, involving memory type, width, depth, data characteristics, as well as memory control mechanism. More precisely, both considering of data-reuse efficiency and memory bandwidth are insufficient to evaluate the power dissipation from reference data buffering. Of course, data-reuse efficiency and memory bandwidth requirement can be reflected by such factors. For a qualified low power design with embedded memory, we should elaborate these interactions simultaneously and should further trade off them between the impacts on power, area, and system performance.

To minimize the bandwidth redundancy, [31] proposed a clever memory structuring, namely "2-D random access", which supports two-dimensional reference buffering by using the "ladder-shaped" interleaving pattern. With the data interleaving, both horizontally and vertically adjacent pixels can be processed in parallel from the reference buffer. In the arithmetic sense, this memory structuring greatly exploits memory redundancy access. Such pixel arrangement substantially eliminates waiting cycles during data access, meanwhile improving the efficiency of inter-candidate data-reuse and of PE utility. Intuitively by [27,31,53] prior assumptions, 2-D random access is therefore capable of significantly reducing memory access power followed with memory bandwidth. However, at least 16 memories will be activated meanwhile during the reading cycle. In other words, a great number of total memory access times substantially consumes heavy processing current during the interval of candidate matching. Besides, considering reference region data-reuse as mentioned previously, the enlarged size of search range (H[−32,+31], V[−16,+15]) contributes prime area on reference buffer (200kgates/64kbits), and predominate over 80% the power of whole dissipation [47]. While the effective bandwidth and data-reuse efficiency is therefore greatly improved, clearly, the design proposed in [31] increase the burden on processing current, even deteriorating the power demand.

4.3.2 Current Minimization in High Design Abstraction

In fact, design with over considering high design abstraction metrics, such as data-reuse, memory bandwidth, or PE utilization is deficient in practical experience. These high abstraction metrics is difficult to truly reflect the design behavior toward the physical. Thus, we need a more specialized perspective to link the abstractions between the algorithmic and the physical.

This section presents a novel methodology for reference buffer optimization using high-

level power analysis. Equivalently, that minimize power are that minimize total current consumed. The corresponded frame structure will minimize the average access current flowed into embedded memories. Assume that the matching algorithm is based on the dynamic boundaries scheme stated in section 3.3, in which both the maximum search range SR_x, and SR_y are limited to 16 pixels. The optimization parameters are dedicated to memory depth L ($L = 1, 2, 4, 8, 16$ pels) and the number of arithmetic array processor M ($M = 1, 2$). The following derivation is presented in the interval of coding one macroblock. The optimized parameters $(L, M)^*$ follows the relationship

$$(L, M)^* = \arg\min_{(L,M)}(E\{i|L, M\}), \text{ for all valid } L, M \tag{4.16}$$

where the average current i within one macroblock coding cycles can be approximated as

$$i = \frac{1}{N_{\mathrm{MB}}}(N_{\mathrm{W}} i_{\mathrm{W}} + N_{\mathrm{R}} i_{\mathrm{R}} + N_{\mathrm{S}} i_{\mathrm{S}}) \tag{4.17}$$

number N_{MB} presents required cycles for coding an macroblock, W indicates memory in "write" state, R indicates memory in "read" state, and S presents that memory stands by for further access. Random variables (RVs) i_{W}, i_{R}, and i_{S} present average currents within one cycle of their corresponded states.

Single array processor ($M = 1$):

Assuming single array processor is applied; then total state of memory in "write" follows

$$N_{\mathrm{W}} = (\mathrm{SP}_y + 15) \times \lceil \frac{\mathrm{SP}_x + 15}{4} \rceil \tag{4.18}$$

where $\lceil \cdot \rceil$ denotes ceiling operation, and SP_y and SP_y are two RVs of total search points in dimension x and y (rectangular window), total stats of memory in "read" can be simplified as

$$N_{\mathrm{R}} = (\mathrm{SP}_y + 15)\{\frac{16(L + \mathrm{SP}_x - 1) + (\mathrm{SP}_x - 1)(L - 1)(L + 16)}{L^2}\} \tag{4.19}$$

and, total stat of memory in "stand-by" is therefore written to be

$$N_{\mathrm{S}} = \frac{48 N_{\mathrm{MB}}}{L} - N_{\mathrm{R}} - N_{\mathrm{W}} \tag{4.20}$$

Dual array processors ($M = 2$):

If the array processor is paralleled by two, then total state of memory in "write" follows

$$N_{\mathrm{W}} = (\mathrm{SP}_y + 15) \times \lceil \frac{\mathrm{SP}_x + 15}{4} \rceil \tag{4.21}$$

total stats of memory in "read" can be simplified as

$$N_{\mathrm{R}} = (\mathrm{SP}_y + 15)\{\frac{32 + (\mathrm{SP}_x - 1)(L + 16)}{2L} \tag{4.22}$$

and, total stat of memory in "stand-by" is therefore written to be

$$N_{\mathrm{S}} = \frac{48N_{\mathrm{MB}}}{L} - N_{\mathrm{R}} - N_{\mathrm{W}} \tag{4.23}$$

To be proceeding, we reasonably assume that RVs SP_x, SP_y, i_{W}, i_{R}, and i_{S} are all independent, and the average access current can be therefore simplified as

$$i_{\mathrm{avg}}N_{\mathrm{MB}} = E\{N_{\mathrm{W}}\}i_{\mathrm{W,avg}} + E\{N_{\mathrm{R}}\}i_{\mathrm{R,avg}} + E\{N_{\mathrm{S}}\}i_{\mathrm{S,avg}} \tag{4.24}$$

where the current terms $i_{\mathrm{W,avg}}$, $i_{\mathrm{R,avg}}$, and $i_{\mathrm{S,avg}}$ present the average current for memory access of their corresponded state, which are assumed independent of other factors, *e.g.*, data and addressing statistics. The average current is listed in the electric characteristics provided by library vendor, or preestimated by experiments.

Table 4.5: Average current list of Artisan n-bit×48words Mux-2 Register file working on 20MHz/30MHz (TSMC-CL013G process)

L	M	$i_{\mathrm{W,avg}}(\mu\mathrm{A})$	$i_{\mathrm{R,avg}}(\mu\mathrm{A})$	$i_{\mathrm{S,avg}}(\mu\mathrm{A})$
$L = 2/n = 10$	$M = 2$ (20MHz)	114	112	5
	$M = 1$ (30MHz)	171	167	5
$L = 4/n = 20$	$M = 2$ (20MHz)	173	178	5
	$M = 1$ (30MHz)	267	259	5
$L = 8/n = 40$	$M = 2$ (20MHz)	307	298	10
	$M = 1$ (30MHz)	460	446	10

By these high-level synthesis data, the optimization of Eq. (4.16) can be thus promptly derived. Table 4.5 lists average access currents of register file using TSMC-CL013G process by Artisan Inc. of different data width and word depth, operating at frequency 20MHz and 30MHz. By these experimental statistics, Table 4.6 enumerates average current (in filed 1, 6), power dissipation in different access states (in field 2–4, 7–9), and power reduction compared with non-stand-by design (in field 5, 10), under vertical search point $\mathrm{SP}_y = 7$, memory width $L = 4$ and $L = 8$ as well as processor count $M = 1$ and $M = 2$, for different horizontal search points SP_x.

From the table, the optimized structure parameters $(L, M)^*$ are selected to $(4, 2)$ (20bit data width, dual arithmetic processors) as the proposed structure design, which is exacted averagely saving 28% — 41% of memory access power. Since bus switching in Table 4.5 is designated to transit randomly, the actual memory access power well be more optimistic than that in Table 4.6.

Table 4.6: High-level Memory Access Power analysis with different architectures (CIF 30fps)

M	SP_x	$L = 8$					$L = 4$				
		i^1_{avg} (μA)	$Power^2_T$ (μW)	$Power^3_{R,W}$ (μW)	$Power^4_S$ (%)	$Reduction^5$ (%)	i^6_{avg} (μA)	$Power^7_T$ (μW)	$Power^8_{R,W}$ (μW)	$Power^9_S$ (%)	$Reduction^{10}$ (%)
$M = 1/30$MHz	5						113.5	136.2	76.2	44.0%	96.3%
	9	171.2	205.4	145.4	29.2%	93.6%	158.7	190.5	130.5	31.5%	94.9%
	13						203.9	244.7	184.7	24.5%	93.4%
	17	268.6	322.3	262.3	18.6%	90.0%	249.1	299.0	239.0	20.1%	92.0%
	21						294.3	353.2	293.2	17.0%	90.5%
	25	366.0	439.2	379.2	13.7%	86.3%	339.5	407.5	347.5	14.7%	89.1%
	29						384.7	461.7	401.7	13.0%	87.6%
	33	463.4	556.0	496.0	10.8%	82.7%	430.0	515.9	455.9	11.6%	86.2%
$M = 2/20$MHz	5						93.3	112.0	52.0	53.6%	97.0%
	9	128.6	154.3	94.3	38.9%	95.2%	118.2	141.9	81.9	42.3%	96.2%
	13						143.2	171.8	111.8	34.9%	95.4%
	17	183.4	220.1	160.1	27.3%	93.1%	168.1	201.7	141.7	29.7%	94.6%
	21						193.1	231.7	171.7	25.9%	93.8%
	25	238.2	285.8	225.8	21.0%	91.1%	218.0	261.6	201.6	22.9%	93.0%
	29						242.9	291.5	231.5	20.6%	92.2%
	33	292.9	351.5	291.5	17.1%	89.1%	267.9	321.5	261.5	18.7%	91.4%

Processor's Trade-off Instead of single processor, dual core design was adopted in proposed motion estimator, since the computation content can be separated by two independent parts without consuming extra apparently power, except duplication leakage and few control logics. Meanwhile, the increased area cost by logic duplication is relatively acceptable.

4.3.3 Actual Power Estimation

SYNOPSYS PrimePower is used to estimate the power dissipation on postlayout gate-level, at memory structure $L = 2$, and $M = 2$ with operating clock rate 20MHz. We test several CIF sequences in terms of moving activity of coding rate 30 fps, 30 frames totally, under quantization parameter, QP, 36 and 42. Clearly, as we apply the technology-dependent memory structuring methodology, the power dissipation of internal memory is successfully suppressed to 12% with respected to total core power and only 7.8% in dynamic power.

Table 4.7: Power consumption analysis for the proposed memory architecture (CIF 30fps, QP = {36,42})

		Static Scene			Active Scene			
Sequence		akiyo	M&D	foreman	foreman	mobile	stefan	
Frames		1–30	1–30	221–250	191–220	1–30	221–250	
Motion		local	local	global	extreme	global	extreme	**Average**
Core	Total (μW)	483.0	493.7	615.7	680.6	601.9	856.2	621.8
	Dyn. (μW)	60.0	70.7	192.7	257.6	178.9	433.2	198.8
Ref. buffer	Total (μW)	62.7	63.6	78.8	88.1	73.2	104.2	78.4
	Dyn. (μW)	2.7	3.6	18.8	28.1	13.2	44.2	18.4
	wrt. Total (core)	13.0%	12.9%	12.8%	12.9%	12.2%	12.2%	12.6%
	wrt. Dyn. (core)	4.5%	5.0%	8.7%	10.9%	7.4%	10.2%	7.8%
Comp. Red.	ME skipped	70.3%	71.0%	33.1%	27.0%	12.9%	26.3%	40.1%
	BW_{mem}	81.1%	80.4%	76.7%	57.8%	77.7%	59.1%	72.1%
	Op. rate	97.6%	97.3%	95.3%	84.5%	96.0%	85.9%	92.8%
R–D	ΔPSNR (dB)	0.076	0.065	0.054	-0.145	0.011	0.155	0.019
	ΔRate (%)	-1.76%	-0.15%	-8.86%	-32.52%	-2.58%	-52.21%	-16.35%

4.4 The Shortest Distance Code

For a fixed size search area, section 4.3 has proposed an optimized local memory structure in power dissipation using high-level technology-dependent power analysis. This section

will present a novel bus coding method - *shortest distance coding* (SDC), which minimizes memory switching redundancy due to sequentially addressing.

For a given memory macro generated by a memory compiler, the power dissipation information is specified in the logic library [54]. The synthesis information contained in the library includes PVT (process, voltage, and temperature) environments, geometry, capacitance, timing and power. The timing model is used for static timing analysis (STA) which is usually classified into control signaling and non-control signaling, and non-control signaling will be further divided into data signaling and addressing modes. However, regardless of data switching, only the addressing power is stated in the (switching) power model for non-control signaling [55].

Refer to Hsien's study [56]. It examined the impact on power dissipation between address switching and data switching during memory access. Two cases was analyzed in Hsien's study in

1. fixing the data bus as constant value and assigning the address bus signals randomly, and
2. fixed address and varying data bus values.

Experiment shows that the switching power on address bus variation has more crucial than that of data bus variation, and the amount of addressing power largely depends on the number of bits in which successive addresses differ (Hamming distance). Besides, since interconnect capacitance gradually predominates the source of capacitance loading as technology scaling down, the address bus broadcasted to each memory macro may largely demands capacitive switching Power.

That is to say once a digital circuit utilizes an on chip memory regarding with low power issue, it should incorporate the consideration of address switching into structural optimization. In this section, the low-power addressing issue will be stated by minimizing circuit switching activity using bus coding technique.

4.4.1 Gray Coding

In a systolic array processor, pipelined processing elements regularly produce block-matching arithmetic each cycle by successively feeding reference data from local memory. In typical, the buffer addressing varies sequentially most of the time. Hence the bus coding technique such as "Gray code" can efficiently reduce the switching power due to memory sequentially addressing [57,58].

A Gray code sequence is a set of numbers, represented by a combination of 1s and 0s, in which contiguous members have only one bit difference. A formal definition of a Gray code sequence is as follows [59]

Definition 1. Let $N_1 = 0, 1$ and $N_k = n_0, n_1, \ldots, n_{2^k-2}, n_{2^k-1}$. We form N_{k+1} by first preceding all members of the sequence N_k by 0, then repeating N_k with the order reversed and all members preceded by 1. In other words,

$$N_{k+1} = 0n_0, 0n_1, \ldots, 0n_{2^k-2}, 0n_{2^k-1}, 1n_{2^k-1}, 1n_{2^k-2}, \ldots, 1n_1, 1n_0.$$

For example, if $N_2 = 00, 01, 11, 10$, $N_3 = 000, 001, 011, 010, 110, 111, 101, 100$. Clearly the foregoing construction ensures that consecutive members of a Gray code sequence differ in exactly 1 bit.

Definition 2. Let N_k be an ordered k digit binary sequence with first element $n_{0,i}$ all 0s. N_k is said to be the shortest distance code, if the cyclic sum, d_{CS}, of Hamming distance for contiguous members are shortest, *i.e.*, the member in k digit binary sequence.

For example, if $N_2 = 00, 01, 11, 10$ in which cyclically consecutive members differ in exactly 1 bit, thus the summed distance for N_2 are shortest ($d_{\mathrm{CS}} = 4 = |N_2|$). Clearly, Gray code is the shortest distance code for a complete binary sequence case. For a sequential complete binary code with k digits, the cyclic sum distance for contiguous members are as following

$$d_{\mathrm{CS}} = 2^k\{1 + \frac{1}{2} + \ldots + \frac{1}{2^{n-1}}\} = 2^{k+1}(1 - \frac{1}{2^n}) \tag{4.25}$$

Hence, switching activity reduction by Gray code addressing of bus width, k, can be expressed as

$$\frac{2^{k-1} - 1}{2^k - 1} \tag{4.26}$$

If $k = 8$, the bus reduces 49.8% of address switching (127/255).

A generalized shortest distance coding for arbitrary member count will be presented below by the modification of Gray code.

4.4.2 The Shortest Distance Code

As word depth requirement of a memory is not always complete (power of 2), in incomplete case, Gray code can no longer be applied as bus coding. To derive the generalized shortest distance code, we should stat some facts on binary sequence.

By the definition, We can detach Gray code into forward segment with capital 0 and backward segment with capital 1. If a part of grouped elements are eject from the first of backward segment, as shown in Figure 4.8, the flip number between the noncontiguous members obviously exceeds 1 bit.

If a truncated sequence is not shortest (cyclic sum distance), we must have a new sequence after some permutations such that the sequence has shortest distance. For this permuted shortest distance sequence, the number of adjacent members in which only one bit differs is as large as possible. In fact, it can be shown that if a full members shortest

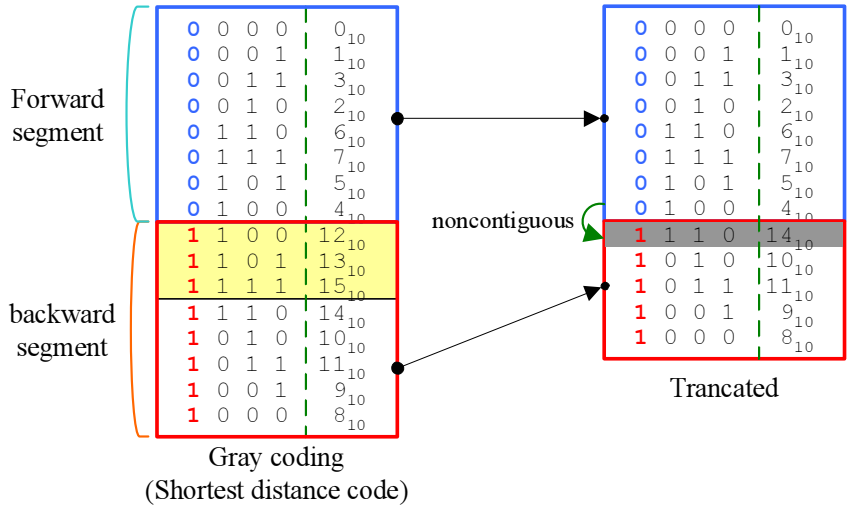


Figure 4.8: Forward segment and backward segment

distance sequence code with first member 0_{10} followed with 1_{10}, then the sequence will be uniquely Gray code by definition 1. In other words, when the sequence $\{0_{10}, 1_{10}, \ldots\}$ is shortest distance, then the sequence implies Grad coded. Since the members excluded first in backward segment are already in shortest order. If there exists the shortest distance sequence after 1 time of permutation, the first member can be relocated to some place such that the new distance is less than the original. However, it can be impossible, as described as follows.

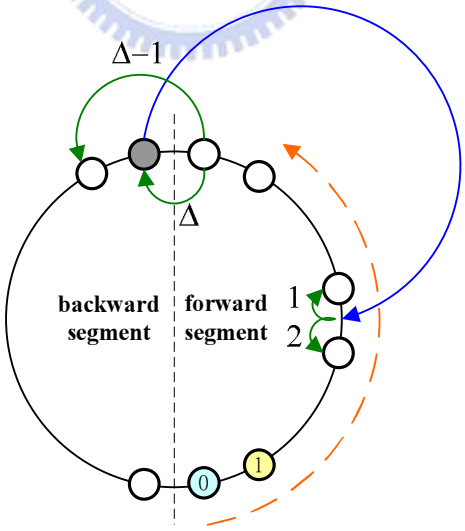


Figure 4.9: An illustration of permutated codeword distance

Let Δ be the flip number on the incision of contracted sequence as stated in Figure 4.9. In fact, since any adjacent member in Gray sequence always differs in 1 bits, the flip number after 1 time of permutation clearly will be at least $\Delta - 1$, and the two additional flip distances on relocated incision must be at least 1 and 2 bits, respectively, as depicted

in Figure 4.9. Thus, the cyclic sum distance of this relocated sequence becomes

$$d_{\mathrm{CS},1} \geq n - 2 + \Delta - 1 + 1 + 2 = n + \Delta \tag{4.27}$$

and the distance before permutation

$$d_{\mathrm{CS},0} = n - 1 + \Delta \tag{4.28}$$

which implies the cyclic sum distance after 1 time codeword permutation greater than that without permutation, $d_{\mathrm{CS},0}$. Without loss generality, such relationship between Eq. (4.27) and Eq. (4.28) can be easily applied to other case of time permutation similarly with above discussion.

Based on aforementioned induction, we have proven that the binary sequence as the form as shown in Figure 4.8 has shortest (cyclic sum) distance. To further utilize the shortest distance code in the physical memory usage, we then propose an equivalent coding algorithm in section 4.4.3. Before we starting, two parameters, addressing length l_a and paging length l_p, are defined as

$$\begin{aligned} l_{\mathrm{a}} &= \log_2 \gcd(d_{\mathrm{mem}}, 2^k) \\ l_p &= k - l_a \end{aligned} \tag{4.29}$$

where gcd denotes greatest common divisor, d_{mem} is the depth or word number of the memory, and k is total address bits.

4.4.3 Shortest distance code Structuring

Figure 4.10 depicts the proposed shortest distance coding (SDC) structure of containing 12 codeword members. The structure consists of two main elements, paging field and addressing field. The paging field is the ordered sequence with l_p bit most significant part of k-bit Gray sequence excluding ejected members, and the addressing field comprise the remaining l_a bit least significant part.

For the case of our design, the depth of embedded memory is 48 ($d_{\mathrm{mem}} = 48, k = 6$). Thus from Eq. (4.29), l_a and l_p are corresponded to 4 and 2, respectively. Paging field contains $\{00, 01, 10\}$, and the addressing field $\{a_{n-1}, a_{n-2}, \ldots, a_0\}$ can be specified by

$$\begin{aligned} a_{n-1} &= b_n \oplus b_{n-1}, \quad && n = l_a \\ a_i &= b_{i+1} \oplus b_i, && i = 0, 1, \ldots, n-1 \end{aligned} \tag{4.30}$$

where $\{a_i\}$, $i = n-1, n-2, \ldots, 0$ is addressing series binary representation. Moreover, for the address decoding in memory, not all truncated cases are always available to data access; such as the truncated case shown in Figure 4.8 , non-continuous integers 12_{10}, 13_{10}, 15_{10} are ejected from Gray code. Therefore, address decoding circuit in memory design should be ability to deal with such blocking requirement; *i.e.*, if necessary, word decoding

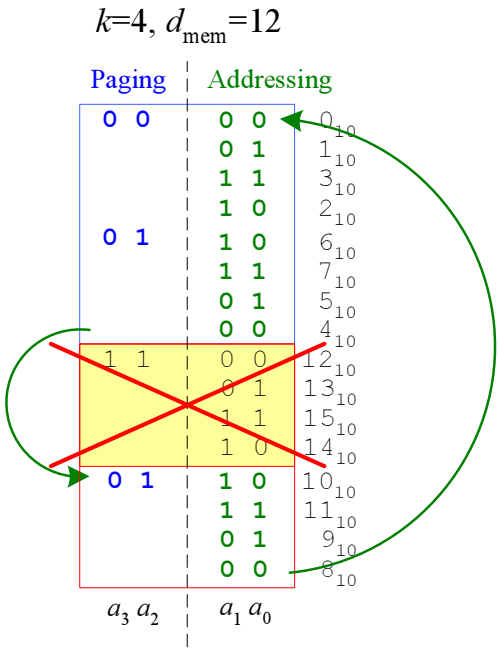


Figure 4.10: An illustration of SDC structure with $k = 4$, $d_{\text{mem}} = 12$

must be customized to accommodate non-continuous addressing, and the proposed SDC technique can be successfully adopted to suppress the power dissipation from address switching.

4.5 Motion Vector Cost

In low rate video coding, motion estimation with Lagrangian rate-distortion optimization will promote apparent coding superiority over in use of SAD alone. The RDO (rate-distortion optimization) adopted in JVT test codec JM has been introduced in 2.2 and 3.2.3. In JM, totally 41 MVs associated with their predictors are specified of variable size from a macroblock to smallest 4×4 block. The motion cost in each block is computed by optimizing the prediction distortion and corresponded candidate offset relative to its displacing predictor. However, such cost measuring is only suited for software realization due to inter-block predictor dependence as well as its arithmetic complexity. As a result, few designs have involved motion vector cost calculation though hardware realization.

According to MV cost arithmetic, prior motion estimations can be divide into three classifications,

- SAD only: most cases of implementation, typically performs ordinary compensated prediction quality,
- By multiplier: Yalcin *et al.* [60] presented a calculating architecture by full parallelism of multiplier, followed with inneglectable complexity overhead, and

Table 4.8: Coding rate and distortion comparisons of motion vector cost (CIF@30fps, QP = {30, 32, 36, 42})

		Full-prediction		16×16 only		W/O MV-cost	
		ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)	ΔBitrate (%)	ΔPSNR (dB)
akiyo	CIF	−0.113%	−0.018	0.899%	−0.015	4.376%	0.004
M&D	CIF	−0.021%	0.013	1.151%	−0.030	6.995%	−0.140
foreman	CIF	0.006%	0.000	1.721%	−0.032	16.121%	−0.031
mobile	CIF	0.048%	−0.002	0.265%	−0.006	4.379%	−0.018
stefan	CIF	1.621%	−0.003	2.226%	−0.014	7.944%	−0.047
Average		0.308%	−0.002	1.252%	−0.019	5.963%	−0.036

- Using LUT: Zhang *et al.* [61] used ROM to store MV costs pre-calculated in an LUT (look up table) manner, also followed with inneglectable complexity overhead.

Both of these designs are suffered from either sacrificed rate-distortion performance or overpaid implementation cost. To be effective, it therefore promotes us presenting a novel cost-efficiency algorithm and its inventive arithmetic logic.

4.5.1 H/W-oriented Block-matching Method

We will begin at some INTER coding properties in low rate video coding:

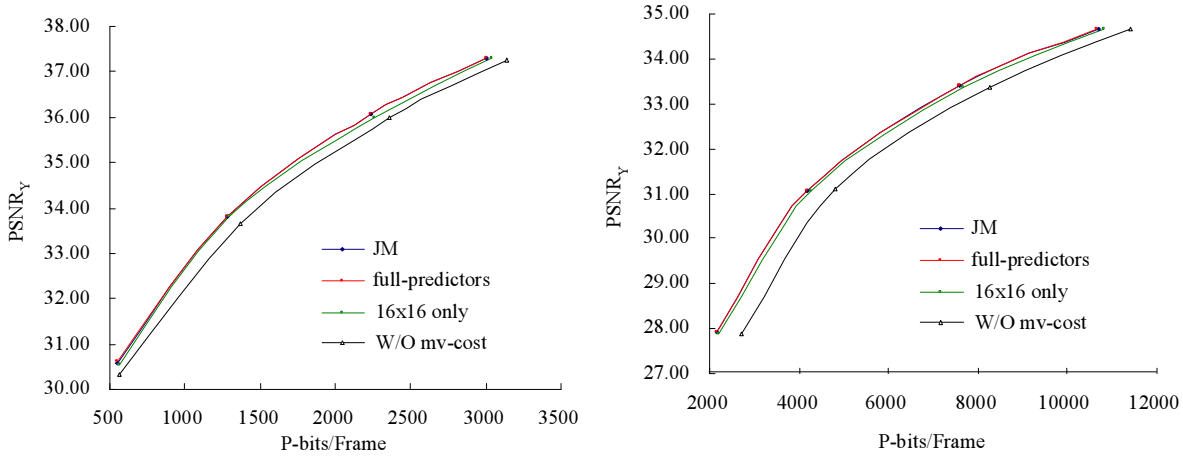
- Macro modes (*i.e.*, SKIP, MCP 16×16) are predominant, and
- Different block predictors are mutually correlated.

Given theses apparent facts, a cost-efficiency matching algorithm may be described as Eq. (4.31) by assuming that predictors are all same with $\mathbf{p}_{16\times16}$

$$J_{motion}(\mathbf{m}|n) = \sum_{M}\sum_{N} |x_{ij} - y_{ij}^{\mathbf{m}}| + \lambda \cdot R_m(4\mathbf{m} - \mathbf{p}_{16\times16}) \tag{4.31}$$

We have estimated several simplified prediction strategies compared with JM on R-D performance, consisted of full-predictions, 16×16 only, and W/O mv-cost. The R-D analyses are illustrated in Table 4.8 and Figure 4.11, where the simulation parameters are specified as follows

- JM: defaults of JM,
- full-predictions: sequentially block matching using individual block predictor,
- 16×16 only: parallel matching, using $\mathbf{p}_{16\times16}$ as all predictors, and
- W/O MV-cost: without using motion vector as cost.



(a) Test sequence: mother_daughter

(b) Test sequence: foreman

Figure 4.11: Coding rate and distortion comparisons of motion vector cost (CIF@30fps, QP = {30, 32, 36, 42})

Figure 4.11(a) and Figure 4.11(b) depict R-D curves for active and static test sequences "mother_daughter" and "foreman", and Table 4.8 summaries average bit-rate difference and PSNR drop compared to JM using quantization parameters {30, 32, 36, 42}. Based on the simulation results, it shows that the motion cost as Eq. (4.31) has already provided the satisfactory R-D performance, while the average bit-rate compared to JM is slightly increased to 1.25% and PSNR drop is less than 0.02dB.

4.5.2 Arithmetic Implementation

function R_m in simplified motion cost Eq. (4.31) can be further expressed as

$$R_m(4\mathbf{m} - \mathbf{p}_{16\times16}) = R(4m_x - p_{x,16\times16}) + R(4m_y - p_{y,16\times16}) \tag{4.32}$$

where R counts codeword bits of the displacement difference between the candidate block and the MB predictor. In H.264, the motion difference is coded as Exp-Golomb variable length code (VLC), which is constructed as follows:

$$[M \text{ leading zeros}][1][\text{INFO}] \tag{4.33}$$

where INFO is an M-bit field carrying information, Notice that the first codeword has no leading zero or trailing INFO; codeword 1 and 2 have a signal-bit INFO field; codeword 3-6 have a 2-bit INFO field; and so on, and M is the number of leading zeros in a codeword, followed the relation

$$M = \lfloor \log_2(\text{code\_num} + 1) \rfloor \tag{4.34}$$

The length of each codeword is clearly $2M + 1$ bits. Table 4.9 lists first 9 codewords in Exp-Golomb VLC.

Table 4.9: Exp-Golomb code number and codeword structures

$d = 4m - p$ (MVD)	code_num	codeword
0	0	1
1	1	010
−1	2	011
2	3	00100
−2	4	00101
3	5	00110
−3	6	00111
4	7	0001000
−4	8	0001001
⋮	⋮	⋮

From the relation of Eq. (4.34) and Table 4.9, the codeword length $(2M + 1)$ can be represented in terms of motion difference d as Eq. (4.35)

$$R(d) = \begin{cases} 1 & d = 0 \\ 2\lfloor \log_2(2|d|) \rfloor + 1 & \text{otherwise} \end{cases} \tag{4.35}$$

and quarter-pel difference component, $d = 4D - \delta$, where D presents motion difference in integer resolution, and δ presents remanding quarter-pel displacement, which is valid 0–3. For a low switching approach, we approximate Eq. (4.35) to the a simpler form as Eq. (4.36), which ignores the *initial increment* "λ" to further reduce arithmetic cost and logic switching. Notice that the logarithm arithmetic can be easily realized by using a "priority encoder".

$$R(D, \delta) = \begin{cases} 0 & D = 0, \delta = 0 \\ 2\lfloor \log_2(2|4D - \delta|) \rfloor & \text{otherwise} \end{cases} \tag{4.36}$$

In sequentially matching, the MV cost of current displacing is obviously summed by the pervious cost and the increment associated to the displacing state, noted $(\Delta J_R(\mathbf{D}_n, \boldsymbol{\delta}))$, and equivalently notice as

$$\Delta J_R(\mathbf{D}_n, \boldsymbol{\delta}) = J_R(\mathbf{D}_n, \boldsymbol{\delta}) - J_R(\mathbf{D}_{n-1}, \boldsymbol{\delta}) \tag{4.37}$$

where the MV cost for current displacing

$$J_R(\mathbf{D}_n, \boldsymbol{\delta}) = \lambda\{R(D_{n,x}, \delta_x) + R(D_{n,y}, \delta_y)\} \tag{4.38}$$

Substituting the aforementioned relations into Eq. (4.37), we now obtain

$$\Delta J_R(\mathbf{D}_n, \boldsymbol{\delta}) = \lambda \cdot \Delta R(D_{n,x}, \delta_x) + \lambda \cdot \Delta R(D_{n,y}, \delta_y) \tag{4.39}$$

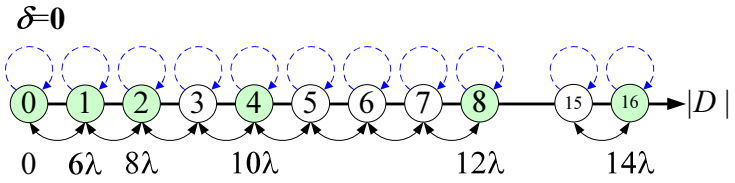


Figure 4.12: Example of motion vector cost calculating, $\boldsymbol{\delta} = \mathbf{0}$

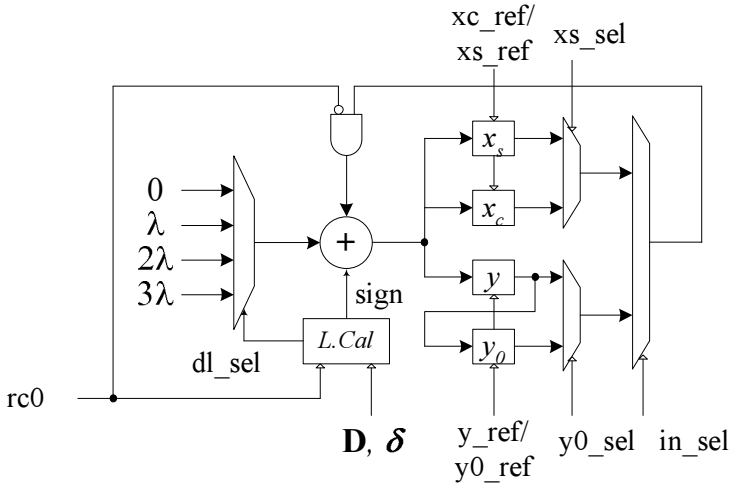


Figure 4.13: Proposed MV cost arithmetic structure

It is obvious from the Eq. (4.39) that the architecture design of displacing cost in Eq. (4.31) can be extremely simplified to use an accumulator, for all block size n.

Figure 4.12 exposes a simple arithmetical example of sequential search in x and in y given at absolute integer difference $|D|$ within $[0, 16]$ and fractional displacement $\boldsymbol{\delta} = \mathbf{0}$.

The proposed accumulator-based arithmetic structure is illustrated at Figure 18. Registers x_c, x_s, and y store the corresponded cost of current motion component in 1^{st} x, 2^{nd} x, and y, respectively (dual processors). Register y_0 stores (initial) boundary cost in y direction, and y_0 is activated when vertical matching at locality of current x is accomplished. The cost value of next state is summed by current cost plus an increment value which is precalculated by $L.Cal$ logic. The $L.Cal$ first calculates Eq. (4.37), which counts the corresponded difference of codeword length between current and next displacement, and then generates the accordant signals (dl_sel and sign) that properly controls the cost value computation.

Chapter 5

Design Specification/Implementation

This chapter describes design specification and implementation flow, including design characteristic, I/O, timing, and architecture specifications as well as implementation procedures and physical chip specifications.

5.1 Design Specification

Design Target

The primary target of this work aims at sub-mW of excellent R-D quality integer-pel motion estimator, supporting to H.264 CIF@30fps video coding capability.

Design Features

The features of proposed implementation are summarized as follows

- Processing capability of maximum H.264/MPEG-4 CIF@30fps Integer-pel motion estimation,
- Significant R-D quality improvement with ultra low power consumption,
- Incorporated with macroblock-skipping detection, further eliminating unnecessary encoding computation,
- Biased matching boundary prediction, significantly improving computational efficiency as well as MCP fidelity,
- Optimized embedded-memory structure, effectively reducing the power dissipation of reference buffering (12 120-byte register files with each 48-word $\times$ 20-bit width),
- Low cost/switching activity processing element (PE), using half the area and power than pervious arts,
- low complexity/switching activity motion vector cost (MV cost) arithmetic, based on low-cost accumulator structure,

- Advanced synthesis flow using Physical Compiler (RTL-to-gate-placed topographical technology), and
- High Density of core area 0.69mm^2 (97Kgates), fabricated by TSMC-CL013G-FSG process.

Supporting Functions

Two principal supporting functions are

- Early macroblock-skipping detection
- Variable block size motion estimation (Combinations of each size of 16 or 8 pixels)
  - Simplified block predictions (16×16, 16×8, 8×16, 8×8, total 9 MVs)

5.2 I/O Specification

Design I/O is specified as Table 5.1, and the symbol view is depicted as Figure 5.2, respectively.

Table 5.1: I/O Description

I/O	Pin	Width	Description	Definition
Input pins	clk	1	Clock signal	Clock input
	rst_	1	Reset signal	Active low
	me_enable	1	Chip enable	
	core_init	1	Core operating enable	
	crtl_in	14	Ctrl data	ctrl_in[0]=LSB
	data_in	32	Encoding data	data_in[0]=LSB
Output pins	skip	1	macroblock-skipping	
	alarm	1	ESD alarm	
	mv_valid	1	MVs valid	
	mv_out	14	(D_x, D_y) output	mv_out[0]=LSB
	me_finish	1	motion search finished	

5.3 Timing Specification

I/O Timing

The I/O timing of proposed motion computing scheme is composed of three main operational functions, including

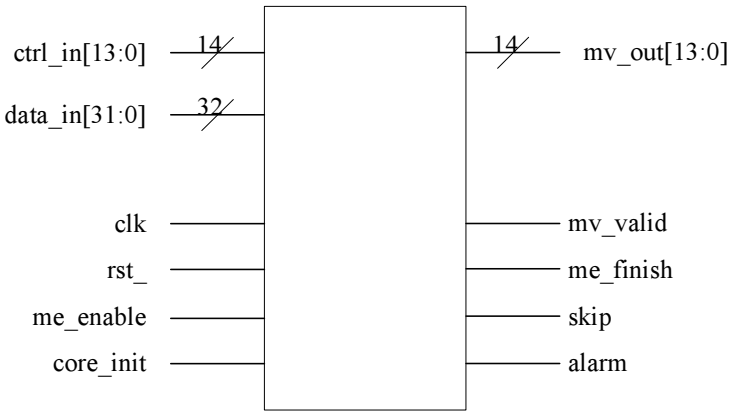


Figure 5.1: The symbol View

1. **State I**, macroblock-skipping detection: The I/O timing during detecting macroblock-skipping is specified as Figure 5.2.
2. **State II**, candidate block matching: The I/O timing during candidate block matching is specified as Figure 5.3, where N presents the required cycle for buffer writing (`data_in`), $N = (\mathrm{SP}_y + 15) \times \lceil \frac{\mathrm{SP}_x+15}{4} \rceil$
3. **State III**, motion difference output:
   (a) Valid after than candidate block matching, sequentially exporting total 9 integer-pel motion difference vectors when `mv_valid` activates.
   (b) Return to IDLE state after rising of `me_finish`, finishing motion exportation.
   (c) The sequence of motion difference vector exporting
   - $\mathbf{D}_{16\times16}$,
     $\mathbf{D}_{16\times8,11}, \mathbf{D}_{16\times8,12}$,
     $\mathbf{D}_{8\times16,11}, \mathbf{D}_{8\times16,21}$,
     $\mathbf{D}_{8\times8,11}, \mathbf{D}_{8\times8,12}, \mathbf{D}_{8\times8,21}, \mathbf{D}_{8\times8,22}$

System Timing

The system operational cycles is as following

Macroblock cycle:	$\mathrm{MB}_{\mathrm{cycle}} = 1683$[1] (cycles/MB)
Total MB-skipping detection cycle:	$\mathrm{ESD}_{\mathrm{cycle}} = 16 \times 16/2 = 128$
Maximum reference buffer load cycle:	$\mathrm{W}_{\mathrm{cycle,max}} = 48 \times 12 = 576$
Maximum RD buffer read cycles:	$\mathrm{R}_{\mathrm{cycle,max}} = 48 \times 16 = 816$
MV out cycles:	$\mathrm{MVO}_{\mathrm{cycle}} = 9$
$\mathrm{Cycle-required}_{\mathrm{max}}$:	$\mathrm{ESD}_{\mathrm{cycle}} + \mathrm{W}_{\mathrm{cycle,max}} + \mathrm{R}_{\mathrm{cycle,max}} + \mathrm{MVO}_{\mathrm{cycle}}$
Redundant cycles:	$\mathrm{MB}_{\mathrm{cycle}} - \mathrm{Cycle\text{-}required}_{\mathrm{max}} = 154$

† Timing details are illustrated in Figure 5.4

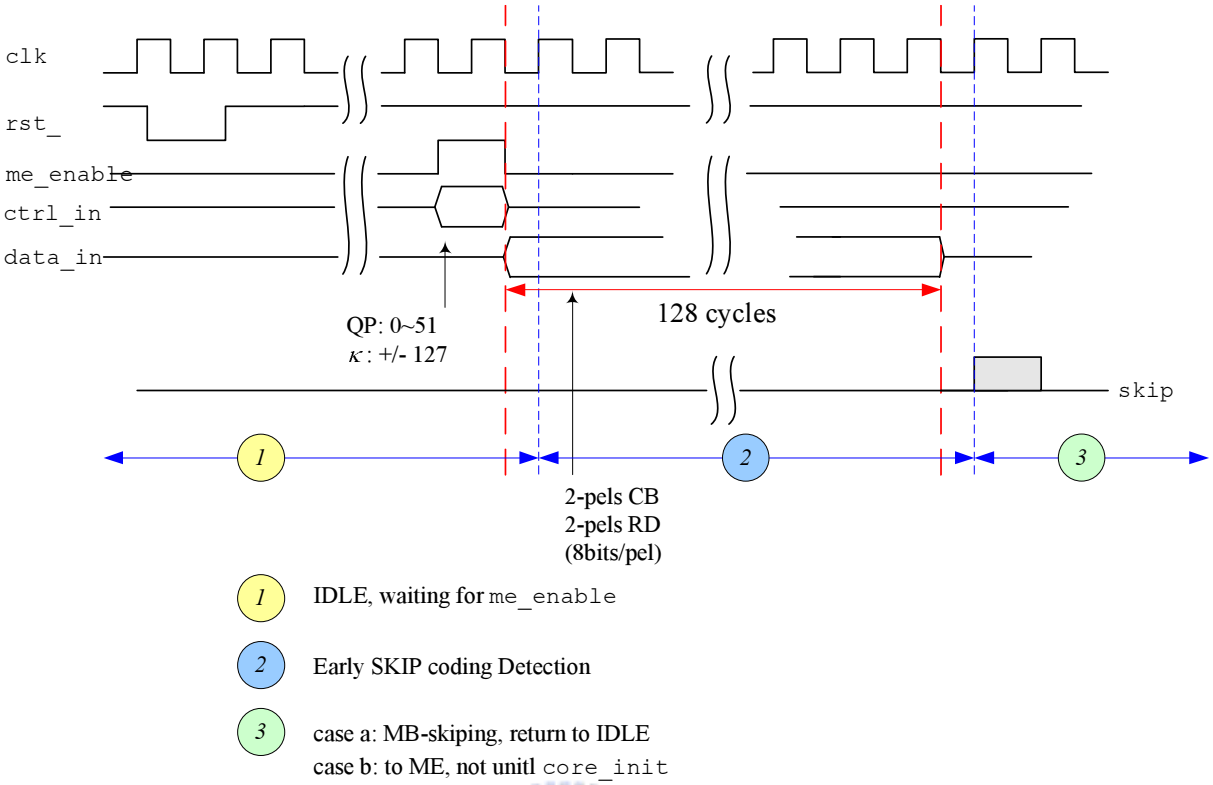


Figure 5.2: State #1: Data transferring timing for the case of macroblock-skipping detection

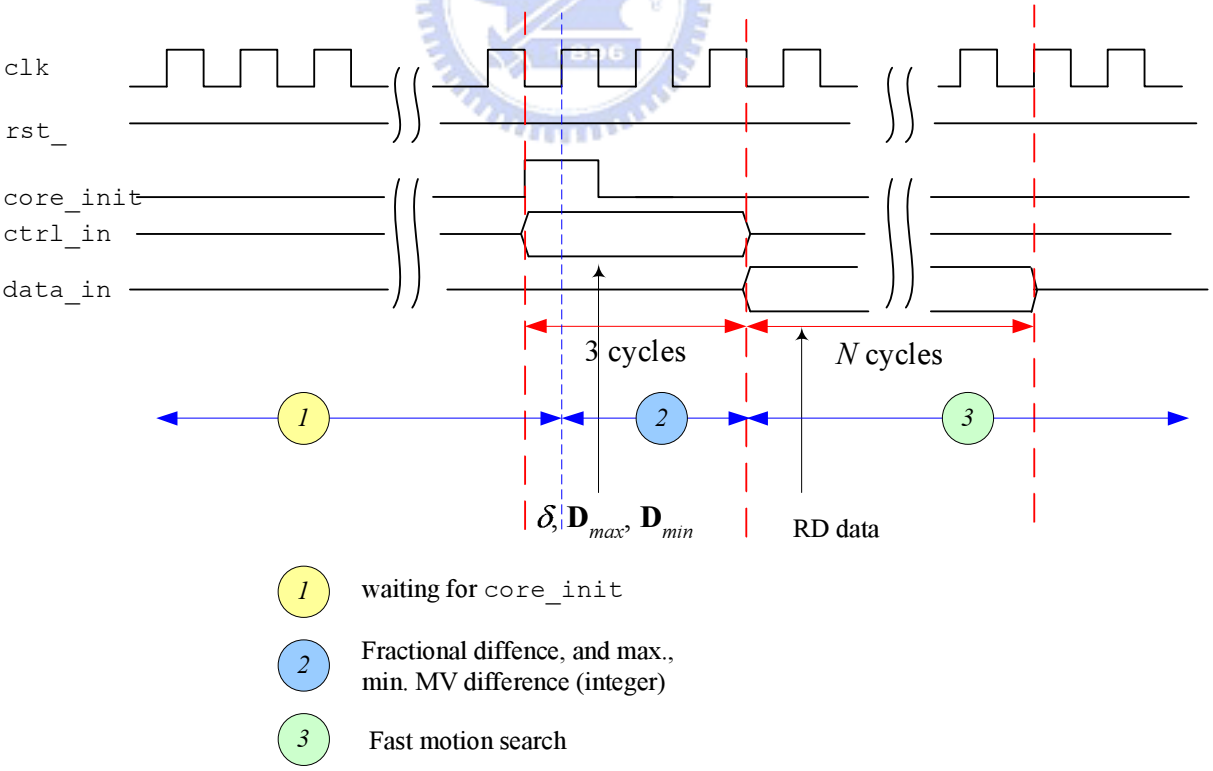


Figure 5.3: State #2: Data transferring timing for the case of candidate block matching

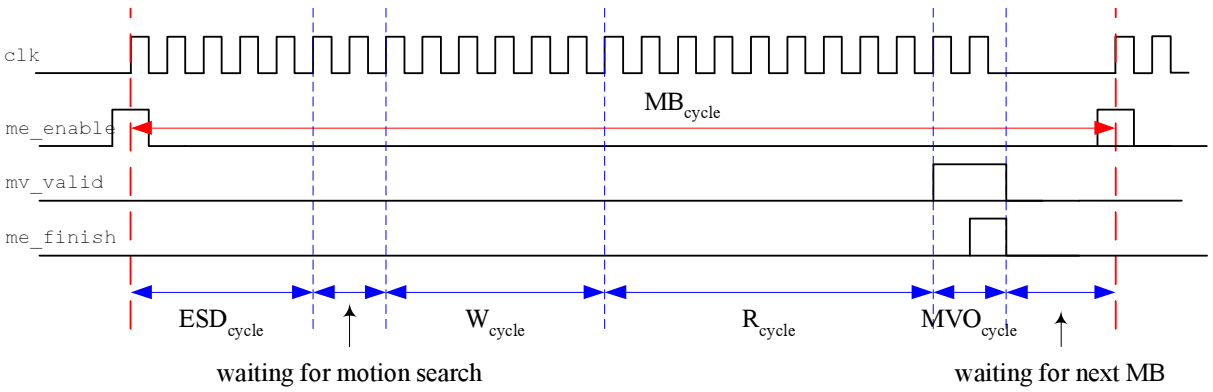


Figure 5.4: A simplified full timing example, including candidate block matching state

5.4 Architecture and Block Diagram Specification

The low power architectural design have been demonstrated in chapter 4. Proposed motion estimator adopts 12 120-byte register files with each 48-word × 20-bit width as reference buffer, and dual arithmetic array processors are applied to eliminate access frequency of reference buffering. Before candidate matching, frame data are imported to ESD, early SKIP detection, for prior decision whether current macroblock should be SKIP coding or not. The macroblock is also replenished parallel to ESD decision. While detecting of SKIP not presents, reference data are then sequentially fed into SAP processors for measuring of SAD distortion. The Lagrangian cost are generated from the propagated SAD through parallel adder tree and summed with current displacing cost, and minimum search units decides the "best" block fitting by minimizing Lagrangian cost. After all valid candidates are completely matched, it orderly exports 9 motion differences, $(D_x, D_y)_n$, $n = 0, 1, 2, \ldots, 8$, and the time motion estimation of current macroblock is accomplished. The functional diagram architecture and corresponded specification are listed in Figure 5.5 and Table 5.2. The corresponded cell gate count of each module is as well summarized in Table 5.3 (after P&R).

5.5 The Implementation/Verfication Flow

This section briefs the proposed design flow, including four separations: 1) design prototyping, 2) physical synthesis, 3) back-end flows, and 4) power analysis.

Design prototyping The implementation/verification flow that specifically applies advanced physical synthesis for low-power/more-robust timing closure demands is depicted in Figure 5.6. The proposed flow starts at high abstraction description model, containing entry of C-Language model of JM codec, MATLAB analysis/varification interface, and RTL design/verification flow. To prior obtain a prototyping floorplan, we first make a

Table 5.2: Functional diagram specification

Module Name	Functionality	Description
ESD	Early SKIP coding detection, core operating-skipping or not	Refer to chap. 2 (pp. 10)
CTRL	Control unit	
CB_BUF	Current block buffer	Register $8 \times 16 \times 10 = 160$bytes
RD_BUF	Reference region buffer	Register file $12 \times 20 \times 48 = 1440$bytes Refer to 4.3 (pp. 58)
SAP_array	Systolic ESAD computation & scheduling propagated ESADs to parallel adder tree	Refer to 4.1 (pp. 45)
MVCOST	Vector cost generation, low-cost acc.-based structure	Refer to 4.5 (pp. 69)
ADDER_TREE	Lagrangian cost calculation	
M_UNIT_array	Minimum cost search units	9 units
MV_out	Exporting motion differences	9 MVs, $(D_x, D_y)_n$, $n = 0, 1, 2, \ldots, 8$

Table 5.3: Cell area count after P&R (Artisan/TSMC-CL013G-FSG standard cell)

Module Name	Cell area (μm^2)	Gate count (NAND)	Relativity (%)
Total	493714.3	96955	
ESD		1010	1.04%
CTRL		1174	1.21%
CB_BUF		8170	8.43%
RD_BUF		32935	33.97%
SAP_array (CORE)		23470	24.21%
SAP_array (SUB)		23434	24.17%
MVCOST		1498	1.55%
ADDER_TREE		2432	2.51%
M_UNIT_array		2013	2.08%
MV_out		267	0.28%
clock buffers		5565	5.74%

trail logic synthesis followed with chip floorplaning using Cadance SOC encounter APR (auto place & route) tool. A prototyping power estimation is also investigated using

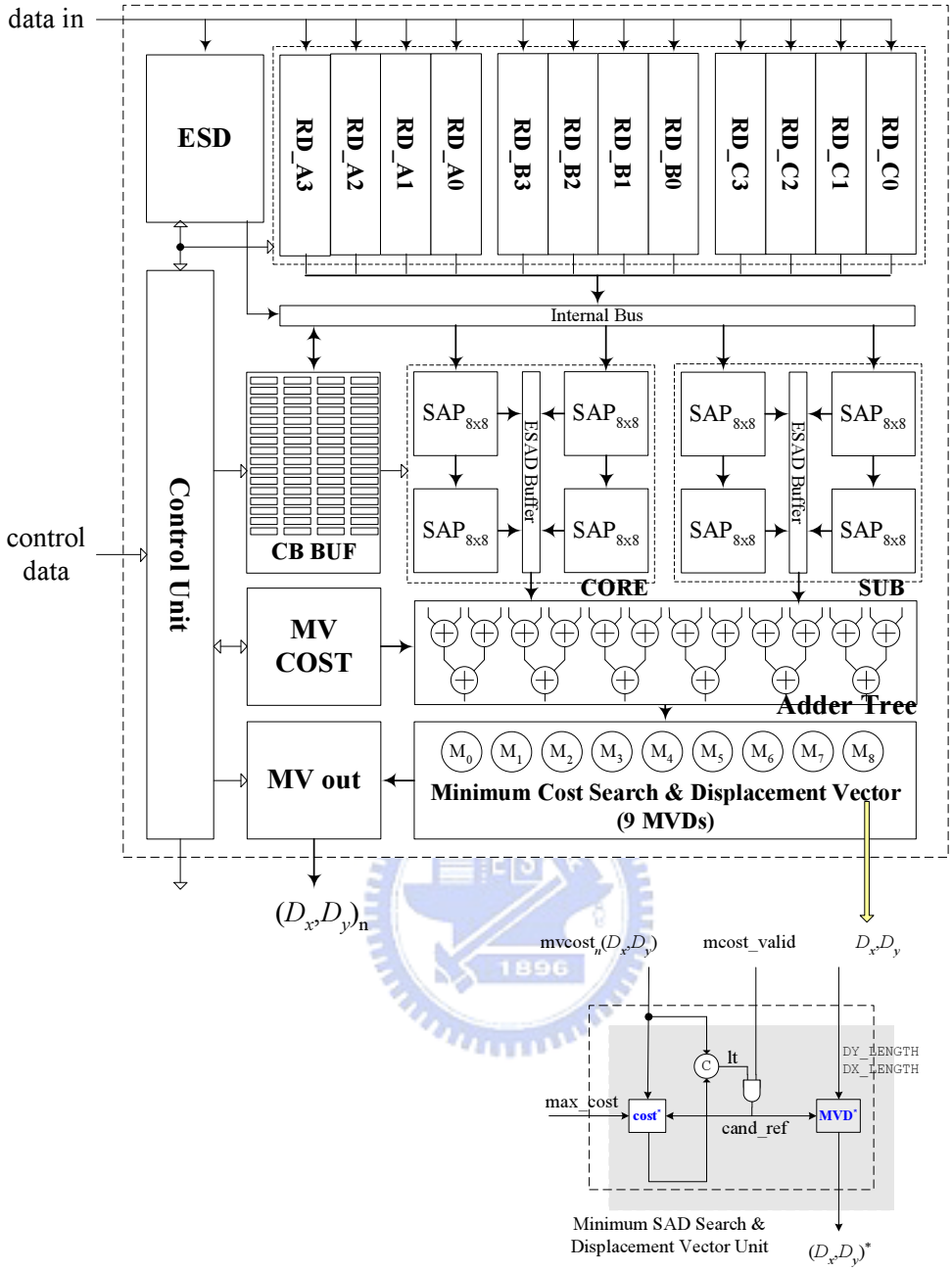


Figure 5.5: The architecture/block diagram of the proposed motion computing scheme

SYNOPSYS PrimePower, and the design iteration may thus be preformed between logic gate and system level for refining/optimizing algorithms/architectures.

Physical synthesis After passing the trial STA (static timing analysis), SYNOPSYS physical compiler is used to logically implement the RTL architecture descriptions with pre-generated prototype chip floorplan (RTL-to-gate-placed synthesis methodology [62]). Physical compiler performs advanced typographical technology, which automatically syntheses logic cell with regarding physical characteristics of placement and routing. Based on the our design experience, physical synthesis is capable of providing more moderate

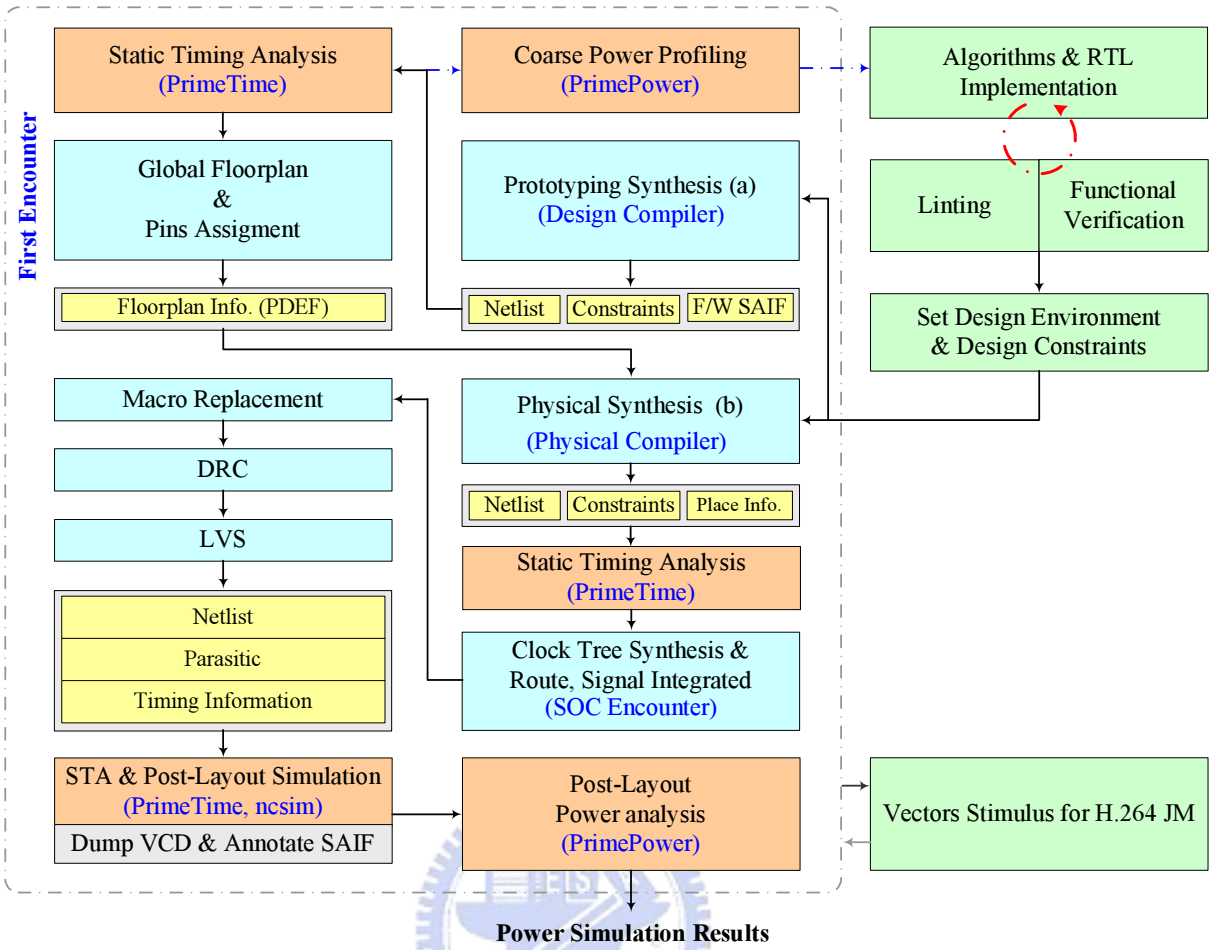


Figure 5.6: A simplified design and analysis flow for proposed Implementations and Verifications

degree of logic synthesis advantages of not only ingenious gate placement but also significantly diminishing switching power due to clock tree buffering.

Back-end implementations and verifications The gate-placed implementation is followd with clock tree synthesis and APR using SOC encounter. The layout verification contains typically DRC/LVS as well as sign-off STA and power estimation.

Power Analysis using PrimePower We use PrimePower to estimate the power information since post-layout gate-level power analysis is greatly efficient and accurate enough. To further improve simulation efficiency, we instead use SAIF (switching activity interface format) technique rather than VCD, which bypasses the simulation time consuming without sacrificing analysis accuracy [63]. Totally 12 test sequences are examined under that QP = {36, 42}, with each sequence consisted of 30 P-frames. In average, the simulation time of using SAIF technique of each sequence is about 1 day, where dramatic simulation burden can be eliminated than transistor-level methodology, *e.g.*, Nanosim.

5.6 Chip Specification

The chip is fabricated by Artisan/TSMC-CL013G-FSG cell libraries, containing standard cell and memory macro, which use TSMC general 0.13μm single-poly eight-metal CMOS technology (uni-threshold voltage, V_t). The area size of implemented chip is listed in Table 5.4.

Core area Lists as Table 5.4.

Table 5.4: Core aspects of implemented chip

Width (μm)	Height (μm)	Area (mm^2)
840.185	819.180	0.688

† Area parameters do not include ring size of $32n$m per side

Clock gating Since the power dissipation from clock tree is always an inneglectable issue in the design of low-power digital VLSI circuit, to efficiently reduce clock tree power, we apply clock gating technique in the proposed design. Total 2806-bit D-F/Fs are clock gated of latch-based gating cells which are automatically transformed using SNYOPSYS Power Compiler [64]. The clock gating ratio is estimated to 99.05%, summarized as follows.

```
                Clock Gating Summary
-----------------------------------------------------------
|    Number of Clock gating elements    |        197       |
|                                       |                  |
|    Number of Gated registers          |   2806 (99.05%)  |
|                                       |                  |
|                                       |                  |
|    Number of Ungated registers        |    27 (0.95%)    |
|                                       |                  |
|    Total number of registers          |        2833      |
-----------------------------------------------------------
```

Average Power Consumption Table 5.5 summaries the average power associated dynamic and leakage power information. The average power dissipation is obtained from total 12 test sequences post-layout simulation under that QP = {36, 42}. Detail test sequence specifications remain in chapter 6.

Table 5.5: Average power consumption (post-layout gate-level using PrimePower)

Total Power	Dynamic Power	Leakage Power
621.8μW	198.8μW	423.0μW
100.0%	32.0%	68.0%

Operating condition: Typical
Operating frequency: 20.00MHz

Chip Layout Figure 5.7 shows an OPUS layout view of implemented chip, where each functional modular has been described in section 5.4 (pp. 78).

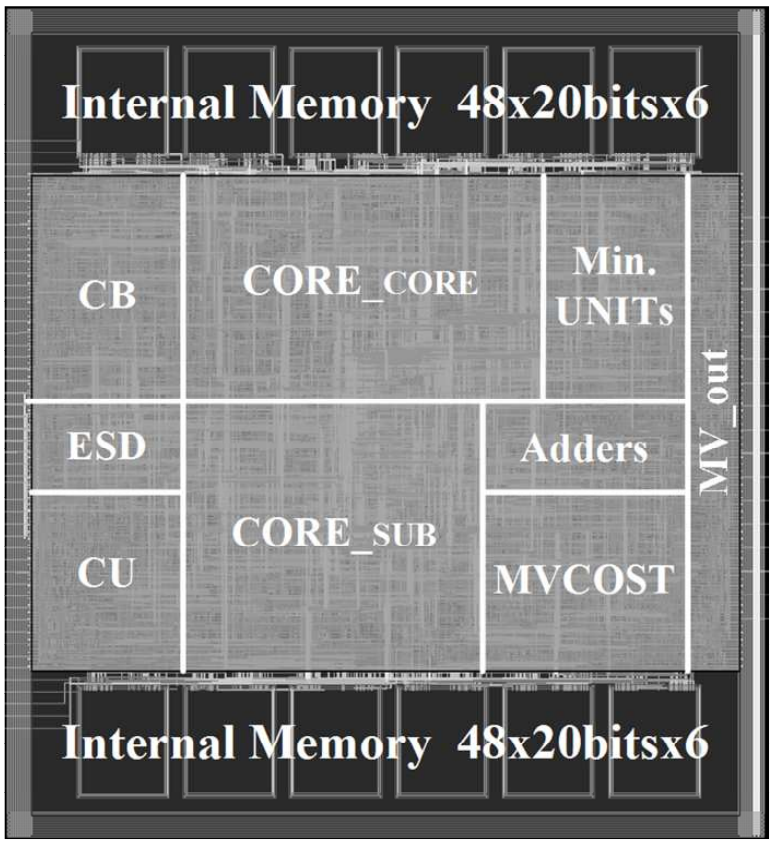


Figure 5.7: OPUS Layout and Floorplaning

Chapter 6

Performance Assessments

This chapter presents experimental assessments in rate-distortion and power measurement. Two compared objects, JM full search and traditional hardware-oriented full search, were selected to assess the system performance in R-D and power. By simulation, it has shown that the proposed design is capable of using half the coding bitrate with better reconstructed quality and saves more than 90% of dynamic power over prior most advanced studies. All assessment environments and comparisons will be stated in the following.

6.1 Test Plan/Assessment Environment

Test system was established on reference baseline JM8.6 [12], RDO mode enabled, one reference frame, one slice per frame, and P-slices only at encoding rate 30fps. We have equivalently modeled the proposed algorithms/architectures and compared traditional full-search scheme in test model for system-accurate assessment. Total 6 encoding scenarios in 5 test sequences were tested in rate-distortion performance, power consumption, and macroblock-skipping characteristics.

In measuring of power, we used SYNOPSES PrimePower for post-layout gate-level power estimation, as already stated in chapter 5. Instead of typical VCD flow, we adopted stimulus-based SAIF technique as power model because of its great time-efficiency with the equality of estimation accuracy of VCD.

The PVT environment in power estimation is assumed in typical condition, *i.e.* working voltage 1.2 V and 25°C temperature. Each test sequence contains 30 P-frame stimulus data, *i.e.*, $30 \times 396 = 11880$ macroblocks per sequence under quantization parameter 36 and 42.

General/Customized Assessment Environment

The comparisons are made against two common block-matching implementations, software full search as well as prior hardware-oriented full search. The general and specialized

encoding environments between the proposed implementation, JM, and prior full-search are listed in Table 6.1 and Table 6.2, respectively.

Table 6.1: General encoding environment

JM	JM 8.6 baseline profile
LevelIDC	3.0
R-D optimization	Enable
Hadamard transform	Disable
Reference frame	1 reference, 1 slice/frame
VOP structure	IPPP...
Frame size	352×288 pixels (CIF)
Frame rate	30fps

Table 6.2: Specialized encoding environment

	Proposed	JM	Full search
Prediction mode	Basic $16 \times 16 \sim 8 \times 8$	Full prediction $16 \times 16 \sim 4 \times 4$	
Predictor	Shared $\mathbf{P}_{16\times16}$	Full Individual	None N/A
Max. search range	16 pixels	16 pixels	16 pixels
Search boundary	Adaptive	Fixed	Fixed
ME center	Dynamic	MV-predictor	$(0, 0)$
MV-cost	Proposed	JM	None
Data-resolution (ESD)	8bits/pel	N/A	
Data-resolution (ME)	5bits/pel	8bits/pel	

Specifically, as described in section 3.4, we applies basic prediction mode ($16 \times 16 \sim 8 \times 8$) and data truncation 3 bits in block matching. In order to detect SKIP coding accurately, we use full data resolution (without data truncation) during partitioned block SAD measuring.

Test Sequences

In performance assessment, all test sequences start at INTRA coding followed with proceeding 30 INTER coding frames for both R-D analysis and power estimation (VOP

(a) akiyo (b) mother_daughter (c) foreman-static

(d) foreman-active (e) mobile (f) Stefan

Figure 6.1: Frame snapshot in the simulation interval

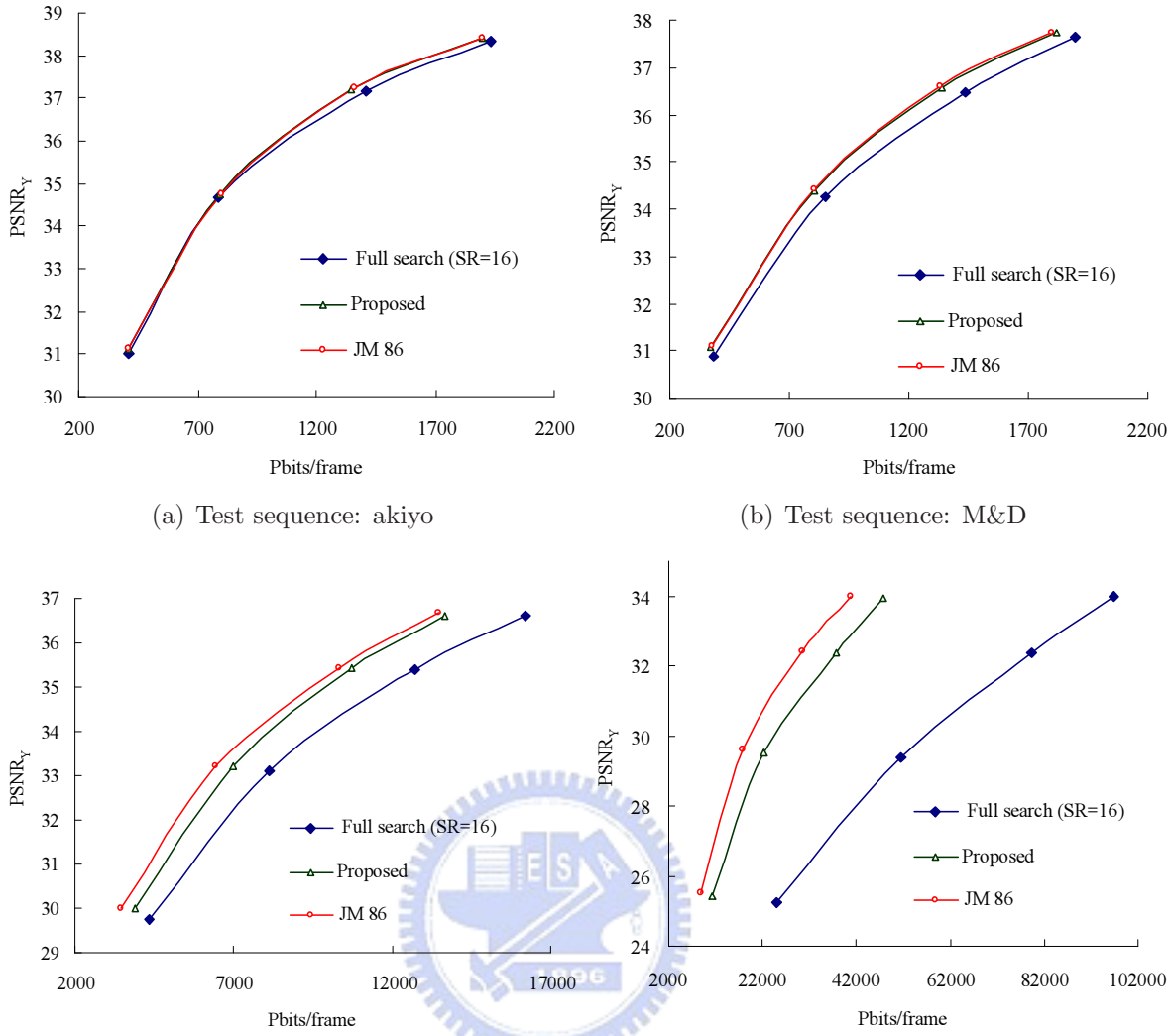
Table 6.3: Test sequences characteristics

	Static			Active		
Sequence	akiyo	M&D	foreman	foreman	mobile	stefan
Motion	local	local	global	extreme	global	extreme
Intra frame	0	0	220	190	0	220
Inter frame	1−30	1−30	221−250	191−220	1−30	221−250
R-D analysis	1−30	1−30	221−250	191−220	1−30	221−250
Power estimation	1−30	1−30	221−250	191−220	1−30	221−250

† QP: for R-D analysis – $\{30, 32, 36, 42\}$, for power estimation – $\{36, 42\}$

structure IPPP...). Total 6 scenarios have been tested, including 3 static CIF test sequences: akiyo (#0 ∼ #30), mother_daughter (#0 ∼ #30), foreman (#220 ∼ #250), and 3 active CIF test sequences: foreman (#190 ∼ #220), mobile (#0 ∼ #30), and stefan (#220 ∼ #250).

Figure 6.1 illustrates snapshots for each scenario. The scene characteristics of these scenarios are summarized in Table 6.3.



(a) Test sequence: akiyo

(b) Test sequence: M&D

(c) Test sequence: foreman - active

(d) Test sequence: stefan

Figure 6.2: Rate-distortion curves of this work, JM, and prior full search

6.2 Rate-Distortion Assessment

Under the simulation environments aforementioned, we have verified rate-distortion distribution of the proposed design against to prior full search and JM.

Table 6.4 compares the rate-distortion and design metrics "BW_{mem}" and "Arithmetic Op." improvements of the proposed algorithm relative to prior hardware-oriented full search. In maximum, the dynamic search boundary algorithm increases 53% of bitrate efficiency at 0.13dB PSNR gain (stefan, QP = 36); even the complexity in "BW_{mem}" and "Arithmetic Op." are greatly reduced to 58.2% and 84.9%, respectively. Generally, the proposed design has ability to eliminate 15% bitrate, 71.9% bandwidth requirement, and 92.5% arithmetic operations. Table 6.5 further compares JM and full search of search range 32 pixels in rate-distortion performance to full-search with SR=16. Notice that "Full search, SR = 32" only increases search range to each side of 32 pixels to prior full

search. The corresponded rate-distortion curve are shown in Figure 6.2.

For static scenarios, this work provides moderate R-D performance since data truncation/prediction simplification degenerate slightly the total quality. For active encoding scenarios, especially in extremely moving, as the predicted search boundary adapts according to neighboring block displacing, it provides significant R-D advantage of maximum 50% bitrate reduction compared with prior full search scheme.

6.3 Power Dissipation Assessment

By post-layout power analysis, three most power demand modules, reference buffer, SAP arrays, and clock buffer, and two power measuring metrics $\mathrm{BW_{mem}}$, arithmetic operation are compared in 6 test senecios for each senecio coding with QP = {36, 42}. For clearly understanding, table 6.6 exhibits summarized characteristics between core design and these primary modules, design metrics as well as corresponded R-D performance in different test sequences. Table 6.7 further compares thesis experimental characteristics in different quantization parameters.

From the tables, the leakage power dominates total power dissipation (68%, 423.0μW of 621.8μW), where the clock tree switching accounts for 41% of core dynamic power, which occupies another 32% of power consumption. Clearly, the power dissipation of internal memory (register file) is successfully suppressed to 12% with respected to total core power and only 7.8% in dynamic power. As a result, dual processors of SAD arithmetic then becomes prime source of total power dissipation, which consumes 0.35mW.

The $\mathrm{BW_{mem}}$ and Arithmetic Operation The metrics $\mathrm{BW_{mem}}$ and arithmetic operation was commonly introduced in section 3.3 to diagnose power in high level of design abstraction.[1] However, as we observer physical power characteristics in fields foreman (active) and stefan, $\mathrm{BW_{mem}}$ and Arithmetic Operation are obviously insufficient for power analysis, since both of them present similar degree of reduction rate but associated with quite different power characteristic, surly with same macroblock-skipped ratio.

Comparison with Prior Designs To compare with the pervious hardware implementation, Table **??** particularly lists design parameters, search algorithms, implemented features, as well as average power dissipation.

Although Huang *et al.* (ISCAS'05) [46] used full search algorithm, which was implemented as similar 2-D systolic array as core architecture of proposed design, it demanded more than 1000 times of the power dispassion over our implementation. Huang's another design (TCSVT'04) [65] used the global elimination fast algorithm with global search

[1] Refer to section 3.3 (pp. 33) for clear definitions of $\mathrm{BW_{mem}}$ and Arithmetic Operation

Table 6.4: Rate-distortion performance and complexity reduction, compared to full search of SR = 16 (CIF@30fps, QP = {30, 32, 36, 42})

	Sequence	Format	QP	ΔBitrate (%)	ΔPSNR (dB)	Average Search point	BW_{mem} Reduction (%)	Arithmetic Op. Reduction (%)
Proposed, Basic Mode, TBs = 3bits, SKIP detection applying	akyio	CIF	30	−1.04%	0.002	27.4	80.7%	97.5%
			32	−1.46%	0.011	28.6	80.5%	97.4%
			36	0.62%	0.062	26.9	80.9%	97.5%
			42	−4.14%	0.089	24.4	81.3%	97.8%
		Average		−1.51%	0.041	26.8	80.9%	97.5%
	m&d	CIF	30	−1.15%	0.014	33.3	79.6%	96.9%
			32	−0.72%	0.032	32.0	79.9%	97.1%
			36	−2.77%	0.075	30.9	80.1%	97.2%
			42	2.47%	0.054	27.5	80.7%	97.5%
		Average		−0.54%	0.044	30.9	80.1%	97.2%
	foreman static	CIF	30	1.60%	−0.007	47.3	77.2%	95.7%
			32	1.14%	0.004	47.4	77.2%	95.6%
			36	−2.40%	0.051	47.4	77.1%	95.6%
			42	−15.32%	0.058	53.9	76.2%	95.1%
		Average		−0.54%	0.044	30.9	80.1%	97.2%
	foreman active	CIF	30	−34.81%	−0.180	169.5	58.2%	84.4%
			32	−35.34%	−0.243	168.9	58.4%	84.5%
			36	−34.31%	−0.224	175.6	57.4%	83.9%
			42	−30.73%	−0.065	162.7	58.1%	85.1%
		Average		−33.80%	−0.178	169.2	58.1%	84.5%
	Mobile	CIF	30	−0.48%	−0.049	44.2	77.7%	95.9%
			32	−0.74%	−0.024	44.8	77.6%	95.9%
			36	−1.33%	−0.003	44.0	77.7%	96.0%
			42	−3.82%	0.024	43.7	77.7%	96.0%
		Average		−1.59%	−0.013	44.2	77.7%	95.9%
	Stefan	CIF	30	−46.55%	0.012	185.2	56.5%	83.0%
			32	−50.62%	0.040	178.7	57.0%	83.6%
			36	−53.56%	0.131	164.6	58.2%	84.9%
			42	−50.86%	0.178	142.6	60.0%	86.9%
		Average		−50.40%	0.090	167.8	57.9%	84.6%
		Average		−15.26%	0.002	81.3	71.9%	92.5%

† Proposed: simplified predictions ($16 \times 16 \sim 8 \times 8$), TBs = 3bits, SKIP detection applying.

pattern. Both of Huang's designs are related high computation complexity and power

Table 6.5: Rate-distortion performance and complexity reduction, compared to full search of SR = 16 (CIF@30fps, QP = {30, 32, 36, 42})

	Sequence	Format	ΔBitrate (%)	ΔPSNR (dB)	Average Search point	BW_{mem} Reduction (%)	Arithmetic Op. Reduction (%)
Proposed	akiyo	CIF	−1.51%	0.041	26.8	80.9%	97.5%
	M&D	CIF	−0.54%	0.044	30.9	80.1%	97.2%
	foreman (static)	CIF	−3.75%	0.026	49.0	76.9%	95.5%
	foreman (active)	CIF	−33.80%	−0.178	169.2	58.0%	84.5%
	mobile	CIF	−1.59%	−0.013	44.2	77.7%	95.9%
	stefan	CIF	−50.40%	0.090	167.8	57.9%	84.6%
	Average		−15.26%	0.002	81.3	71.9%	92.5%
JM86	akiyo	CIF	−2.32%	0.116			
	M&D	CIF	−2.02%	0.110			
	foreman (static)	CIF	−12.03%	0.129			
	foreman (active)	CIF	−40.57%	−0.142			
	mobile	CIF	−3.05%	0.080			
	stefan	CIF	−57.03%	0.221			
	Average		−19.50%	0.086	1089	0%	0%
Full search, SR = 32	akiyo	CIF	−1.4%	−0.005			
	M&D	CIF	0.8%	−0.003			
	foreman (static)	CIF	−0.2%	0.001			
	foreman (active)	CIF	−26.1%	−0.067			
	mobile	CIF	1.4%	−0.006			
	stefan	CIF	−56.3%	0.130			
	Average		−13.6%	0.008	4225	−277.8%	−288.9%

† Proposed: simplified predictions ($16 \times 16 \sim 8 \times 8$), TBs = 3bits, SKIP detection applying.
† Full search, SR = 32: increasing search range to each side of 32 pixels.

consumption. Miyama *et al.* (JSSC'04) [32] used 1-D tree architecture under fast gradient decent algorithm. Because gradient decent algorithm does not suited for hardware implementation, the design performs poor power, huge area, and ordinary R-D efficiency. For Lin *et al.* (ISCAS'04) [33] and Chen *et al.* (TCSVT'07) [31] designs, they adopted the parallel 1-D tree systolic architecture and 2-D tree systolic architecture, respectively, as already clearly discussed in section 4.1. For all aforementioned designs, suffering of mal-assumptions and criteria demands large hardware cost and hence wastes power. Additionally, that all of these designs use $(0, 0)$ as search center conditionally degrades the rate-distortion performance as well as increases memory access bandwidth.

6.4 Chapter Summary

Applied with impressive dynamic boundary prediction scheme as well as the proposed low-power/high-performance methodology, our work supports dramatic power and rate-distortion superiority over prior studies, which has been summarized in average of 621.8μW of power dissipation with dynamic 198.8μW and more than half the bitrate improvement of equal frame PSNR compared to prior hardware-oriented full search, as from Figure 6.2 to Table 6.8.

Table 6.6: Power & rate-distortion assessments for the proposed memory architecture (CIF 30fps, QP = {36,42})

		Static Scene			Active Scene			
Sequence		akiyo	M&D	foreman	foreman	mobile	stefan	
Frames		1–30	1–30	221–250	191–220	1–30	221–250	
Motion		local	local	global	extreme	global	extreme	**Average**
Core	Total (μW)	483.0	493.7	615.7	680.6	601.9	856.2	621.8
	Dyn. (μW)	60.0	70.7	192.7	257.6	178.9	433.2	198.8
Ref. buffer	Total (μW)	62.7	63.6	78.8	88.1	73.2	104.2	78.4
	Dyn. (μW)	2.7	3.6	18.8	28.1	13.2	44.2	18.4
	wrt. Total (core)	13.0%	12.9%	12.8%	12.9%	12.2%	12.2%	12.6%
	wrt. Dyn. (core)	4.5%	5.0%	8.7%	10.9%	7.4%	10.2%	7.8%
SAP	Total (μW)	280.0	285.0	310.9	378.0	340.7	502.4	349.5
	wrt. Total (core)	58.0%	57.7%	51.4%	55.5%	56.6%	58.7%	56.3%
Clock	Dyn. (μW)	33.25	36.97	73.71	94.94	63.41	112.60	69.1
	wrt. Dyn. (core)	55.6%	52.4%	39.5%	36.9%	35.5%	26.1%	41.0%
Comp. Red.	ME skipped	70.3%	71.0%	33.1%	27.0%	12.9%	26.3%	40.1%
	BW_{mem}	81.1%	80.4%	76.7%	57.8%	77.7%	59.1%	72.1%
	Op. rate	97.6%	97.3%	95.3%	84.5%	96.0%	85.9%	92.8%
R–D	ΔPSNR (dB)	0.076	0.065	0.054	-0.145	0.011	0.155	0.029
	ΔRate (%)	-1.76%	-0.15%	-8.86%	-32.52%	-2.58%	-52.21%	-17.35%

Table 6.7: Power & rate-distortion assessments of the proposed implementation (CIF 30fps, QP = {36,42})

	Sequence		QP	Core power			Ref. buffer		SAP array		Clock tree		SKIP	BW_{mem}		Arithmetic Op.		Rate-Distortion	
				Total (μW)	Dyn. (μW)	D/T (%)	Total (μW)	wrt. Core	Total (μW)	wrt. Core	Total (μW)	wrt. Core	ratio ratio (%)	Avg. SA	Saving (%)	Avg. SP	Saving (%)	ΔRate (%)	ΔPSNR (dB)
Static Scene	Akiyo	CIF	36	487.8	64.8	13.3%	63.1	12.9%	282.0	57.8%	35.1	54.2%	65.3%	441.0	80.9%	26.9	97.5%	0.62%	0.062
			42	478.2	55.2	11.5%	62.3	13.0%	278.0	58.1%	31.4	56.9%	75.3%	429.9	81.3%	24.4	97.8%	-4.14%	0.089
		Avg.		483.0	60.0	12.4%	62.7	13.0%	280.0	58.0%	33.2	55.6%	70.3%	435.5	81.1%	25.6	97.6%	-1.76%	0.076
	M&D	CIF	36	500.2	77.2	15.4%	64.2	12.8%	288.1	57.6%	39.2	50.8%	68.2%	457.9	80.1%	30.9	97.2%	-2.77%	0.075
			42	487.2	64.2	13.2%	63.0	12.9%	281.8	57.8%	34.7	54.0%	73.7%	445.3	80.7%	27.5	97.5%	2.47%	0.054
		Avg.		493.7	70.7	14.3%	63.6	12.9%	285.0	57.7%	37.0	52.4%	71.0%	451.6	80.4%	29.2	97.3%	-0.15%	0.065
	Foreman 221–250	CIF	36	537.8	114.8	21.3%	67.1	12.5%	317.0	58.9%	48.7	42.4%	47.1%	526.5	77.1%	47.4	95.6%	-2.40%	0.051
			42	693.5	270.5	39.0%	90.5	13.0%	304.8	44.0%	98.7	36.5%	19.1%	549.2	76.2%	53.9	95.1%	-15.32%	0.058
		Avg.		615.7	192.7	30.2%	78.8	12.8%	310.9	51.4%	73.7	39.5%	33.1%	537.9	76.7%	50.6	95.3%	-8.86%	0.054
Active Scene	Foreman 191–220	CIF	36	693.5	270.5	39.0%	90.5	13.0%	384.4	55.4%	98.7	36.5%	19.1%	980.9	57.4%	175.6	83.9%	-34.31%	−0.224
			42	667.7	244.7	36.6%	85.7	12.8%	371.6	55.7%	91.2	37.2%	34.8%	964.9	58.1%	162.7	85.1%	-30.73%	−0.065
		Avg.		680.6	257.6	37.8%	88.1	12.9%	378.0	55.5%	94.9	36.9%	27.0%	972.9	57.8%	169.1	84.5%	-32.52%	−0.145
	Mobile	CIF	36	601.4	178.4	29.7%	73.3	12.2%	340.3	56.6%	63.6	35.6%	9.1%	513.6	77.7%	44.0	96.0%	-1.33%	−0.003
			42	602.3	179.3	29.8%	73.1	12.1%	341.1	56.6%	63.2	35.3%	16.6%	513.8	77.7%	43.7	96.0%	-3.82%	0.024
		Avg.		601.9	178.9	29.7%	73.2	12.2%	340.7	56.6%	63.4	35.5%	12.9%	513.7	77.7%	43.9	96.0%	-2.58%	0.011
	Stefan	CIF	36	896.4	473.4	52.8%	110.1	12.3%	527.5	58.8%	120.0	25.3%	21.0%	963.3	58.2%	164.6	84.9%	-53.56%	0.131
			42	815.9	392.9	48.2%	98.4	12.1%	477.2	58.5%	105.2	26.8%	31.6%	921.5	60.0%	142.6	86.9%	-50.86%	0.178
		Avg.		856.2	433.2	50.5%	104.2	12.2%	502.4	58.7%	112.6	26.1%	26.3%	942.4	59.1%	153.6	85.9%	-52.21%	0.155
Average				621.8	198.8	32.0%	78.4	12.6%	349.5	56.3%	69.1	41.0%	40.1%	642.3	72.1%	78.7	92.8%	-16.35%	0.019

Table 6.8: Power consumption comparisons of previous arts

	NTU, Huang *et al.* ISCAS'03 [46]	NTU, Huang *et al.* TCSVT'04 [65]	Miyama *et al.* JSSC'04 [32]	NTU, Lin *et al.* ISCAS'04 [33,45]	NTU, Chen *et al.* TCSVT'07 [31,37,47]	This work
Processing Capability	720×480@30fps	CIF@30fps	CIF@30fps	CIF@30fps	CIF@30fps	CIF@30fps
Resolution	Integer-pel	Integer-pel	Half-pel	Integer-pel	Integer-pel	Integer-pel
Block Size	16×16	16×16	16×16, 8×8	16×16	$16 \times 16 \sim 4 \times 4$	$16 \times 16 \sim 8 \times 8$
Algortihm	Full search	Global Elimination	Gradient Decent	Fast full search (4SS)	Fast full search (4SS-based)	Fast full search
Search Center	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$	Adaptive
Technology	0.35μm	0.35μm	0.18μm	0.18μm	0.18μm	0.13μm
Voltage	3.3V	3.3V	1.2V	1.8V	1.3V	1.2V
Core Area	$25.56\mu\text{m}^2$	$27.8\mu\text{m}^2$	$13.65\mu\text{m}^2$	$2.2\mu\text{m}^2$	$3.61\mu\text{m}^2$	$0.69\mu\text{m}^2$
Internal Memory	24.6Kbits	24.1Kbits	40Kbits	24.1Kbits	64Kbits	11.5Kbits
Eqv. Gates (2-NAND)	106Kgates	115Kgates	1000Kgates	137Kgates	300Kgates	96Kgates
Clock Rate	66.67MHz	27.8MHz	13.5MHz	48.67MHz	13.5MHz	20MHz
Power Consumption	$737.32m$W	$189m$W	$12m$W	$8.46m$W	$4.38m$W	$0.62m$W
Dynamic Power			**N/A**			$0.20m$W

Chapter 7

Conclusions

To reveal an impressive design methodology for portable video codec, this thesis thoroughly treats design tradeoffs in motion estimation between rate-distortion, complexity, and power. Three most advanced power-efficiency techniques have been explored to achieve an excellent low-power, cost-efficiency, and high rate-distortion requirements for advanced motion estimation task.

A false rate constrained MB-skipping detection in use of likelihood ratio test has successively exploited the computation dependence and effectively eliminated the complexity burden. The proposed detection method moderately detects whether the current macroblock should be SKIP coding or not prior to motion estimation, where the 17%–87% probability of detection is archived at false rate 1% relative to the motion activity.

Because of advanced prediction on intensive mathematics, motion estimation dominates computation and memory access complexity. To effectively release the power demand, prior studies was strictly exploited the characteristics of hardware acts in excessively compromising rate-distortion performance. In addition, measuring of power in high abstraction regardless dependance of architecture is generally insufficient. To illustrate, a biased block-matching scheme in use of dynamic boundaries is then preformed, which has been proven significantly improving arithmetic efficiency as well as motion-compensated fidelity. Simulation assessment shows that our method can save maximum 52.6% the coding bitrate efficiency, and release more than 90% arithmetic amount.

In reference buffer optimization, we theocratically give a perspective in optimizing memory structure using technology-dependent high-level power analysis methodology. By applying the optimization methodology, the power dissipation of internal memory (register file) is successfully suppressed to 12% with respected to total core power and only 7.8% in dynamic power.

Moreover, three novel key techniques was first presented in this thesis are 1) low switching AD (absolute difference) logic (section 4.2), 2) shortest distance bus coding scheme (section 4.4), and 3) low-cost accumulator-structured MV cost computation logic

(section 4.5). These ingenious arithmetic simplifications or designs effectively advantages ours work as well.

As an excellent design methodology was applied in the implementation, the proposed design supports dramatically power and rate-distortion superiority over prior designs, which has been summarized in average of $621.8\mu W$ of power dissipation with dynamic $198.8\mu W$ and more than half the bitrate improvement of equal frame PSNR compared to prior hardware-oriented full search. In brief, the present work not only elegantly resolves the prior design gap between rate-distortion fidelity and power demanding, but also further regulates the design guideline for qualified low power video motion estimation, and has greatly encouraged the advancement to realized a real-time, low-cost, as well as high-performance video codec.

Reference

[1] *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification*, ITU-T Recommendation H.264 and ISO/IEC 14496-10 AVC, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Mar. 2003.

[2] *Codec for Videoconferencing Using Primary Digital Group Transmission*, ITU-T Recommendation H.120, ITU-T (formerly CCITT), version 1, 1984; version 2, 1988.

[3] T.-C. Chen, S.-Y. Chien, Y.-W. Huang, C.-H. Tsai, C.-Y. Chen, T.-W. Chen, and L.-G. Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 6, pp. 673–688, Jun. 2006.

[4] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[5] R.-L. Lin, "Method for determining to skip macroblocks in encoding video," U.S. Patent 6,192,148, Feb. 20, 2001.

[6] B. A. Hall *et al.*, "Macroblock coding technique with biasing towards skip macroblock coding," U.S. Patent 6,993,078, Jan. 31, 2006.

[7] C. S. Kannangara *et al.*, "Low-complexity skip prediction for H.264 through Lagrangian cost estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 2, pp. 202–208, Feb. 2006.

[8] Y. V. Ivanov and C. J. Bleakley, "Skip prediction and early termination for fast mode decision in H.264/AVC," in *Proc. ICDT'06*, 2006.

[9] Y. Zhao, M. Bystrom, and I. Richardson, "A MAP framework for efficient skip/code mode decision in H.264," in *Proc. IEEE ICIP'06*, Oct. 2006, pp. 45–48.

[10] JM web site. [Online]. Available: http://iphome.hhi.de/suehring/tml/

[11] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[12] JM reference software version 8.6. [Online]. Available: http://iphome.hhi.de/suehring/tml/

[13] J.-F. Yang, S.-C. Chang, and C.-Y. Chen, "Computation reduction for motion search in low rate video coders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 948–951, Oct. 2002.

[14] Y. Cheng, K. Dai, Z. Wang, and J. Lu, "Motion search method based on zero-block detection in H.264/AVC," in *Proc. Int. Conf. on Computer Supported Cooperative Work in Design (CACWD'04)*, vol. 2, May 26–28, 2004, pp. 739–743.

[15] Y. H. Moon, G. Y. Kim, and J. H. Kim, "An improved early detection algorithm for all-zero blocks in H.264 video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 8, pp. 1053–1057, Aug. 2005.

[16] H. Wang, S. Kwong, and C.-W. Kok, "Effectively detecting all-zero DCT blocks for H.264 optimization," in *Proc. IEEE ICIP'06*, Oct. 2006, pp. 1329–1332.

[17] H. L. V. Trees, *Detection, Estimation, and Modulation Theory.* New York, US: John Wiley & Sons, 2001.

[18] A. Raghunathan, N. K. Jha, and S. Dey, *High-Level Power Analysis and Optimization.* Norwell, US: Kluwer Academic Publishers, 1998.

[19] F. N. Najm, "Transition density: a new measure of activity in digital circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 12, no. 2, pp. 310–323, Feb. 1993.

[20] A. M. Tekalp, *Digital Video Processing.* New Jersey, US: Prentice Hall, 1995.

[21] P. Kuhn, *Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation.* Norwell, US: Kluwer Academic Publishers, 1999.

[22] *Video Coding for Low Bitrate Communication*, DRAFT ITU-T Recommendation H.263, ITU-T, May 2, 1996.

[23] A. Puri, X. Chen, and A. Luthra, "Video coding using the h.264/mpeg-4 avc compression standard," *Signal Processing: Image Communication*, vol. 19, no. 9, pp. 793–849, Oct. 2004.

[24] T. Wiegand, M. Lightstone, D. Mukherjee, T. G. Campbell, and S. K. Mitra, "Rate-distortion optimized mode selection for very low bit rate video coding and the emerging h.263 standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 2, pp. 182–190, Apr. 1996.

[25] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization: Methods and Applications*. New York, US: John Wiley & Sons, 1983.

[26] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, no. 9, pp. 1445–1453, Sep. 1988.

[27] C.-Y. Chen, S.-Y. Chien, Y.-W. Huang, T.-C. Chen, T.-C. Wang, and L.-G. Chen, "Analysis and architecture design of variable block-size motion estimation for H.264/AVC," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 2, pp. 578–593, Feb. 2006.

[28] L. Fanucci, L. Bertini, and S. Saponara, "Programmable and low power VLSI architecture for full search motion estimation in multimedia communications," in *Proc. ICME'00*, vol. 3, Jul. 30–Aug. 2, 2000, pp. 1395–1398.

[29] M. Sayed, I. Amer, and W. Badawy, "Towards an H.264/AVC full encoder on chip: An efficient real-time VBSME ASIC chip," in *Proc. IEEE ISCAS'06*, vol. 2, May 21–24, 2003, pp. 2613–2616.

[30] C.-M. Ou, C.-F. Le, and W.-J. Hwang, "An efficient vlsi architecture for H.264 variable block size motion estimation," *IEEE Trans. Consum. Electron.*, vol. 51, no. 4, pp. 1291–1299, Nov. 2005.

[31] T.-C. Chen, Y.-H. Chen, S.-F. Tsai, S.-Y. Chien, and L.-G. Chen, "Fast algorithm and architecture design of low-power integer motion estimation for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 5, pp. 568–577, May 2007.

[32] M. Miyama, J. Miyakoshi, Y. Kuroda, K. Imamura, H. Hashimoto, and M. Yoshimoto, "A sub-mw MPEG-4 motion estimation processor core for mobile video application," *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1562–1570, Sep. 2004.

[33] S.-S. Lin, P.-C. Tseng, and L.-G. Chen, "Low-power parallel tree architecture for full search block-matching motion estimation," in *Proc. IEEE ISCAS'04*, vol. 2, May 23–26, 2004, pp. 313–316.

[34] C.-Y. Chen, Y.-W. Huang, C.-L. Lee, and L.-G. Chen, "One-pass computation-aware motion estimation with adaptive search strategy," *IEEE Trans. Multimedia*, vol. 8, no. 4, pp. 698–706, Aug. 2006.

[35] T. Yamada, M. Ikekawa, and I. Kuroda, "Fast and accurate motion estimation algorithm by adaptive search range and shape selection," in *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP'05)*, vol. 2, Mar. 18–23, 2005, pp. 897–900.

[36] S. Li, Y. Jiang, T. Ikenaga, and S. Goto, "Content-based motion estimation with extended temporal-spatial analysis," *IEICE Trans. on Info. and Syst.*, vol. E88-D, no. 7, pp. 1561–1568, Jul. 2005.

[37] Y.-H. Chen, T.-C. Chen, and L.-G. Chen, "Hardware oriented content-adaptive fast algorithm for variable block-size integer motion estimation in H.264," in *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP'05)*, Dec. 13–16, 2005, pp. 341–344.

[38] P. E. Landman and J. M.Rabaey, "Power estimation for high level synthesis," in *Proc. IEEE EDAC'93*, Feb. 22-25, 1993, pp. 361–366.

[39] Z.-L. He, C.-Y. Tsui, K.-K. Chan, and M. L. Liou, "Low-poewr VLSI design for motion estimation using adaptive pixel truncation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 5, pp. 669–678, Aug. 2000.

[40] V. G. Moshnyaga, "Reducing switching activity of subtraction via variable truncation of the most-significant bits," *J. VLSI Signal Process. Syst.*, vol. 33, no. 1, pp. 75–82, 2003.

[41] S. Y. Kung, *VLSI Array Processors*. New Jersey, US: Prentice Hall, 1988.

[42] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. New York, US: John Wiley & Sons, 1999.

[43] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-time Signal Processing*, 2nd ed. New Jersey, US: Prentice Hall, 1999.

[44] C.-P. Lin, P.-C. Tseng, Y.-T. Chiu, S.-S. Lin, C.-C. Cheng, H.-C. Fang, W.-M. Chao, and L.-G. Chen, "A 5mw MPEG4 SP encoder with 2D bandwidth-sharing motion estimation for mobile applications," in *Proc. IEEE ISSCC'06*, vol. 2, Feb. 6-9, 2006, pp. 1626–1635.

[45] S.-S. Lin, "Low-power motion estimation processors for mobile video application," Master's thesis, National Taiwan University, Taipei, Taiwan, 2004.

[46] Y.-W. Huang, T.-C. Wang, B.-Y. Hsieh, and L.-G. Chen, "Hardware architecture design for variable block size motion estimation in MPEG-4 AVC/JVT/ITU-T H.264," in *Proc. IEEE ISCAS'03*, vol. 2, May 25–28, 2003, pp. 796–799.

[47] T.-C. Chen, Y.-H. Chen, S.-F. Tsai, and L.-G. Chen, "Architecture design of low power integer motion estimation for H.264/AVC," in *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP'06)*, vol. 3, 2006, pp. 900–903.

[48] S. Vassiliadis *et al.*, "The sum-absolute-difference motion estimation accelerator," in *Proc. IEEE Euromicro Conf.*, vol. 2, Vasteras, Sweden, Aug. 25-27, 1998, pp. 559–566.

[49] D. F. McManigal, "Absolute difference generator for use in display system," U.S. Patent 4,218,751, Aug. 19, 1980.

[50] T. Kanoh, "Absolute value calculating circuit having a single adder," U.S. Patent 4,953,115, Aug. 28, 1990.

[51] J. M. Dodson and C. T. Cheng, "Low-power area-efficient absolute value arithmetic unit," U.S. Patent 5,251,164, Oct. 5, 1993.

[52] T. Komarek and P. Pirsch, "Array architectures for block matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 36, no. 10, pp. 1301–1308, Oct. 1989.

[53] J.-C. Tuan, T.-S. Chang, and C.-W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 1, pp. 61–72, Jan. 2002.

[54] H. Bhatnagar, *Advanced ASIC Chip Synthesis: Using Synopsys Design Complier Physical Compiler and PrimeTime*, 2nd ed. Norwell, US: Kluwer Academic Publishers, 2002.

[55] *Artisan Standard Library Register File Generator User Manual*, Artisan Components, 2003Q4, release 2.2.

[56] W.-T. Hsien *et al.*, "An efficient power modeling approach for embedded memory using LIB format," in *Proc. IEEE VLSI-TSA-DAT*, Apr. 27-29, 2005, pp. 55–58.

[57] C.-L. Su, C.-Y. Tsui, and A. M. Despain, "Saving power in the control path of embedded processors," *IEEE Des. Test. Comput.*, vol. 11, no. 4, pp. 24–31, 1994 Winter.

[58] C.-H. Lin, C.-M. Chen, and C.-W. Jen, "Low power design for mpeg-2 video decoder," in *Proc. IEEE TCE'96*, vol. 42, no. 3, 1996, pp. 513–520.

[59] J. P. Hayes, *Computer Architecture and Organization*, 3rd ed. New York, US: McGraw-Hill, 1998.

[60] S. Yalcin, H. F. Ates, and I. Hamzaoglu, "A high performance hardware architecture for an SAD reuse based hierarchical motion estimation algorithm for H.264 video coding," in *Proc. Int. Conf. on Field Programmable Logic and Appl. (FPL'06)*, Aug. 24–26, 2005, pp. 509–514.

[61] L. Zhang and W. Gao, "Improved FFSBM algorithm and its VLSI architecture for variable block size motion estimation of H.264," in *Proc. Int. Symp. on Intell. Signal Process. and Commun. Syst. (ISPACS'05)*, Hong Kong, Dec. 13–16, 2005, pp. 445–448.

[62] *Physical Compiler User Guide*, SYNOPSYS, Inc., Jun. 2003, version U-2003.06.

[63] *PrimePower Manual*, SYNOPSYS, Inc., Dec. 2004, version W-2004.12.

[64] *Power Compiler User Guide*, SYNOPSYS, Inc., Jan. 2005, release W-2004.12.

[65] Y.-W. Huang, S.-Y. Chien, B.-Y. Heish, and L.-G. Chen, "Global elimination algorithm and architecture design for fast block matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 898–907, Jun. 2004.