Journal of Materials and Electronic Devices 1 (2019) 7-10

JOURNAL OF MATERIAL SCIENCE AND ELECTRONIC DEVICES
FYTRONIX JMED http://dergi-fytronix.com/

Low power adaptive Motion Estimator for real time applications H.264/AVC Codec

Salah Dhahri[1,2], Abdelkrim Zitouni[3]
[1]Laboratory of Electronic and Micro-Electronic (LAB-IT06) Faculty of Sciences of Monastir, 5019, Tunisia
[2]Research Unit of Valorization and Optimization of Exploitation of Resources, Faculty of Science and Technology of Sidi Bouzid, 9100, Tunisia
[3]Department of Physics, College of Science and Humanities - Jubail, Imam Abdulrahman Bin Faisal University, P.O. Box: 1982, Jubail Industrial City 31961, Saudi Arabia

In the past few decades, we have been running towards the Digital Age. Telephony networks, In-ternet, satellites, third-generation (3G) wireless networks, these advanced transmission technolo-gies and network infrastructure ensure digital information can flood into every corner around us. So, many application of digital image processing are enhanced by motion estimation algorithms. In a video Codec such as H264AVC, motion estimation (ME) comprises one of the most important compression methods for video communications. Since this task is computationally intensive to result in large power consumption, a low-power design is essential for portable or mobile systems. In this paper we propose a ME architecture that is adaptive to the TSS (three-step search) and the GS (Gradient search) algorithms to support the slow and rapid H264AVC video sequences. This architecture is based on parallel processing technique. The proposed design is also suitable for low-cost implementation in modern multimedia applications.

Keywords: H.264, Fast Adaptive Motion Estimation, Gradient Search, Three step Search, Low Power, Area.

Submission Date:10 January 2019
Acceptance Date: 01 March 2019

Corresponding author: azitouni@iau.edu.sa

1. Introduction

It is widely believed that video coding techniques have always been advancing the multimedia technology during the last years. Several applications knows more and more exponential develop-ment and a wide public use, such as video conferencing, DVD players, video security, digital cam-eras, and so on. all these multimedia applications indicate a push through of video coding tech-niques.
A several standards has been developed for different multimedia applications, such as H.261 [1], H.263 [2], MPEG-1 [3], MPEG-2 [4] and MPEG-4 [5] , and H264[6,7]. These standards are to meet the requirements of different types of applications in terms of com-plexity, picture quality, bit rate, latency, etc. The new technology also contributed to the develop-ment of forms of different areas from time to time, for example, education, meteorology, medicine, etc.
To make, these standards use in addition a fixed technique compression such us a DCT, a system video coding by motion estimation. H.264/AVC adopts the motion estimation block. The Opera-tion of motion estimation is crucial an encoder video. Indeed, firstly the quality of a motion esti-mator significantly affects the compression performance, and secondly it is the operation which necessitates the addition of computing power. the motion estimation reduces the temporal redundancy between the successive pictures of a video sequence to obtain a high data compression ratio. According to studies that were made in [8], the motion estimation algorithm takes 74.29 % of the computations, and 77.49 % of the memory accesses for the whole encoder. However, a large number of VLSI architectures for motion estimation have been proposed during the last two decades to accelerate the ME process and reduce energy consumption while maintain-ing video quality such us the FSS (Four Step search) algorithm [9], the DS (Diamond Search) algo-rithm [10], the GS (Gradient Search) algorithm [11], the TSS (Three Step Search) algorithm [12], hexagon-based search (HEXS) [13], etc.

A desirable solution, presented in thise paper, is an adaptive motion estimation algorithm (AMEA) and its VLSI implementation. The proposed architecture is capable of running the ME in GS who is desirable for delivering high quality video service or TSS who is desirable for reducing the execution time with a compromise in the compression quality. It based on a stoppable clock for power saving, which is integrated in the PE array and the unit control of both algorithim. An important aspect of this architecture is that it is able to reuse the results of the previous calculated SAD theSADs(sum of absolute differences) in order to compute the actual SAD, deduce the motion vector from neighboring macroblocks, and compare all the SADs in a parallel way and with mini-mal resources.The highest level of this VLSI architecture contains 9PEs array to compute the SADs, comparators to determine the minimum SAD and a distributed control unit for economize the power consumption. This paper is organized as follows. In Section 2, related work is reviewed. The proposed architecture is discussed in Section 3. A distributed control for power saving is pro-posed in Section 4. The experimental results are explained and discussed in Section 5. Finally, Sec-tion 6 concludes the paper.

2. Experimental details

According to various proposals, we can adopt various methods to implement a best compromise between the execution time and complexity of the motion estimator. Nonetheless various algorithms succeeded in accelerating of the ME process and reducing the power consump-tion, but do not give well defined optimal solutions. In [14], Ng Ka Ho presented a new motion estimation algorithm called search patterns switching (SPS) algorithm, which uses the error descent rate as motion content classifier. An optimum threshold for the SPS algorithm is found. Perfor-mance evaluations prove that SPS algorithm is very fast and robust for only some video sequences. However, the SPS algorithm does not usually give the highly performance quality and uses a min-imum of search points. In [15], Jinku Choi proposed a reconfigurable approach to motion estima-tion algorithm that supports two block-matching algorithms (TSS and GS). This algorithm deter-mines motion type and then selects the well adapted algorithm in order to improve video quality. However, the developed architecture duplicates the control and the data path of each algorithm and it is not based on the circuit sharing method. Anand Paul et al.[16]This paper introduces a block‑ based motion estimation algorithm based on projection with Adaptive Window Size Selec-tion (AWSS) along with its architecture. This projection method is combined with AWSS in which appropriate search window for each block is determined on the basis of motion vectors and predic-tion errors obtained for the previous block. However, this method is very fast several times , but does not usually give the highly performance.

3. Results and discussion

In view of the increasing demand for computation power of video coding applications, it is in-creasingly common to use the power of algorithms to speed up processing time and respect real-time constraints. So, the main objective is to propose algorithms able to compete with existing standard algorithms. for this, an adaptive motion estimation is presented in this article, in order to obtain higher frequencies accompanied by the least possible surface cost and to ensure an accepta-ble image quality. the proposed architecture is illustrated in the fig1.

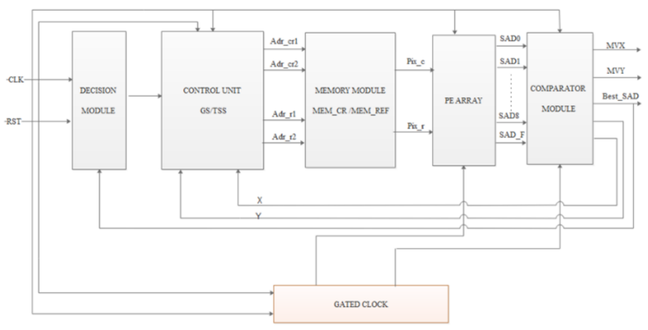

Fig.1: *Proposed architecture of adaptive ME*

3.1 Motivation

The design of our motion estimator architecture is based on two observations.

• Adaptive algorithm: Traditional motion estimation hardware architectures generally use only an algorithm. To improve coding efficiency, a desirable solution for portable devices is an adaptive motion estimation engine that is able to operate the estimator in two (slow and fast) search modes.

• Faster Search Algorithms: Advances in mobile communication technologies have enabled mo-bile devices to run complex multimedia applications such as video processing. In order to speed up the ME computation we pipelined the data flow treatment by adopting an adaptive and parallel architecture composed by nine processing element (PE). Each PE calculates the SAD of one of nine search points.

3.2. PEunit

The processing element unit allows the computation of SAD in the adaptive motion estimator ap-proach. The PEs, in the proposed architecture,are working in parallel. The proposed pipelined pro-cessing architecture consists of 9 PEs which evaluate matching for the nine candidate locations at time and operates on a set of nine motion vectors at an in parallel.
The PE has two inputs one from the courant MB and one from the reference block. These pixels are broad casted to the PE after access to the memories. The distribution of pixels is shown in table1.

3.3. Comparator array

Based on a sharing method, the comparator permits the determination of the MV and its coordi-nates, when the best SAD is extracted with minimum value from several positions of the gradient or three steep search algorithm. The position information is again sent to the control unit to select the next step search for the current algorithm with another distance, or to select the appropriate algorithm.

4. Distributed control for power saving

The power consumed in the proposed architecture is related to parameters that affect consump-tion such as frequency.
Energy consumption in ME can reach 70% of the overall H.264 consumption. For this, it is necessary to integrate a modul of control that reduces the power consumption. Stopable clock is among the techniques that are used to reduce power consumption. The proposed model in this pa-per is based on an FSM which makes it possible to operate the running of the various other mod-ules. Therefore, we integrate FSM into the PE and Comparator networks.

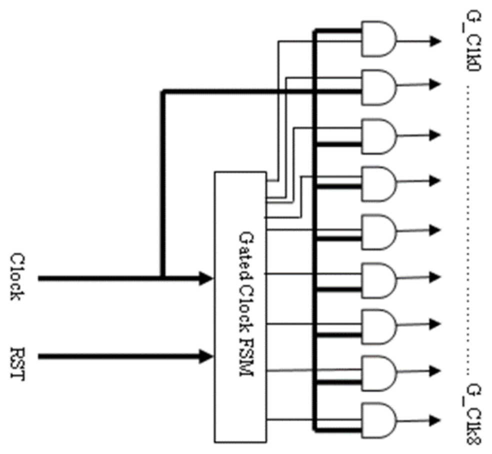

Fig.1: *Gated Clock*

So,we have developed a hardware architecture based on an FSM model, which is subsequently in-tegrated into the adaptive architecture of the estimator. The principle operate of our proposal is as follows:

- The data flow of pixels is pipelined, and there are no pixels that should be processed. Therefore, disabling these PEs is necessary in order to minimize the consumed power .
The clock of each PE will be activated at the instant when it starts to make its calculation during the first few clock cycles or during the last cycles. So the 9 PE not run simultaneously all the time. Let us use the TSS algorithm for example, the PE0 operates at t = 0, and the other PEs are disa-bled. At t= t +4 is the PE1 that works, at t = t +8 is the PE2 that works. After 56 cycles (i.e. PE2) is the EP3 works at t = t +64 and PE4 is operating at t = t+68. At t = t +72 is the PE5 that works. After 56 cycles (i.e. PE5) at t = t+128 is the PE6 that works. At t = 132 t is the PE7 works and the PE8 works for t = t +136. At t= t+256, the PE0 finishes its calcula-tion (disabled), then this is the PE1 which ends the calculation at t = t+260, and so on until PE8 ends its calculation. The SAD0 is calculated and PE0 is disabled after 256 cycles, and so on. Thus, our idea is to design a module which allows the stopping of the each PE clock when this PE is disabled.
The flowchart in Fig 3 shows the different steps to obtain a module capable of stopping the clock in PE during the time required for the operation of the estimator based on the table 1.
- The comparator will only be active after the calculation of the 9 SAD.

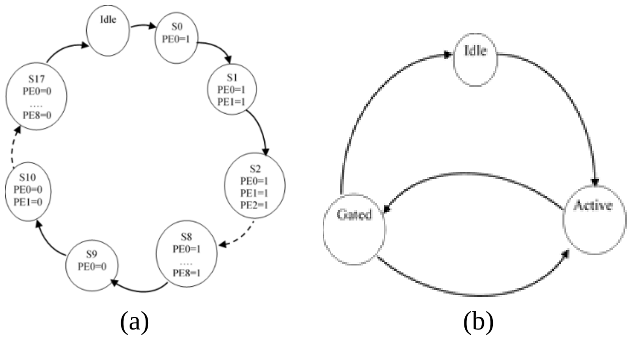

Figure 3: *(a) PE estimator FSM , (b) Comparator FSM Gated Clock*

Table 1: Data Flow of TSS algorithm

Mr	PE0	PE1	PE2	PE3	PE4	PE5	PE6	PE7	PE8
r(-4,-4)	c(0,0),r(-4,-4)								
r(-3,-4)	c(1,0),r(-3,-4)								
r(-2,-4)	c(2,0),r(-2,-4)								
r(-1,-4)	c(3,0),r(-1,-4)								
r(0,-4)	c(4,0),r(0,-4)	c(0,0),r(0,-4)							
r(1,-4)	c(5,0),r(1,-4)	c(1,0),r(1,-4)							
r(2,-4)	c(6,0),r(2,-4)	c(2,0),r(2,-4)							
r(3,-4)	c(7,0),r(3,-4)	c(3,0),r(3,-4)							
r(4,-4)	c(8,0),r(4,-4)	c(4,0),r(4,-4)	c(0,0),r(4,-4)						
r(5,-4)	c(9,0),r(5,-4)	c(5,0),r(5,-4)	c(1,0),r(5,-4)						
•	•	•	•	•	•	•	•		•
r(6,-1)	c(9,3),r(6,-1)	c(5,3),r(5,-1)	c(1,3),r(5,-1)						
r(11,-1)	c(14,3),r(11,-1)	c(10,3),r(10,-1)	c(6,3),r(10,-1)						
r(12,-1)	c(15,3),r(12,-1)	c(11,3),r(11,-1)	c(7,3),r(11,-1)						
r(-4,0)	c(0,4),r(-4,0)	c(12,3),r(12,-1)	c(8,3),r(12,-1)	c(0,0),r(-4,0)					
r(-3,0)	c(1,4),r(-3,0)	c(13,3),r(13,-1)	c(9,3),r(13,-1)	c(1,0),r(-3,0)					
r(-2,0)	c(2,4),r(-2,0)	c(14,3),r(14,-1)	c(10,3),r(14,-1)	c(2,0),r(-2,0)					
r(-1,0)	c(3,4),r(-1,0)	c(15,3),r(15,-1)	c(11,3),r(15,-1)	c(3,0),r(-1,0)					
r(0,0)	c(4,4),r(0,0)	c(0,4),r(0,0)	c(12,3),r(16,-1)	c(4,0),r(0,0)	c(0,0),r(0,0)				
r(1,0)	c(5,4),r(1,0)	c(1,4),r(1,0)	c(13,3),r(17,-1)	c(5,0),r(1,0)	c(1,0),r(1,0)				
r(2,0)	c(6,4),r(2,0)	c(2,4),r(2,0)	c(14,3),r(18,-1)	c(6,0),r(2,0)	c(2,0),r(2,0)				
r(3,0)	c(7,4),r(3,0)	c(3,4),r(3,0)	c(15,3),r(19,-1)	c(7,0),r(3,0)	c(3,0),r(3,0)				
r(4,0)	c(8,4),r(4,0)	c(4,4),r(4,0)	c(0,4),r(4,0)	c(8,0),r(4,0)	c(4,0),r(4,0)	c(0,0),r(4,0)			
r(5,0)	c(9,4),r(5,0)	c(5,4),r(5,0)	c(1,4),r(5,0)	c(9,0),r(5,0)	c(5,0),r(5,0)	c(1,0),r(5,0)			
•	•	•	•	•	•	•	•	•	•
r(10,3)	c(14,7),r(10,3)	c(10,7),r(10,3)	c(6,7),r(10,3)	c(14,3),r(10,3)	c(10,3),r(10,3)	c(6,3),r(10,3)			
r(11,3)	c(15,7),r(11,3)	c(11,7),r(11,3)	c(7,7),r(11,3)	c(15,3),r(11,3)	c(11,3),r(11,3)	c(7,3),r(11,3)			
r(-4,4)	c(0,8),r(-4,4)	c(12,7),r(12,3)	c(8,7),r(12,3)	c(0,4),r(-4,4)	c(12,3),r(12,3)	c(8,3),r(12,3)	c(0,0),r(-4,4)		
r(-3,4)	c(1,8),r(-3,4)	c(13,7),r(13,3)	c(9,7),r(13,3)	c(1,4),r(-3,4)	c(13,3),r(13,3)	c(9,3),r(13,3)	c(1,0),r(-3,4)		
r(-2,4)	c(2,8),r(-2,4)	c(14,7),r(14,3)	c(10,7),r(14,3)	c(2,4),r(-2,4)	c(14,3),r(14,3)	c(10,3),r(14,3)	c(2,0),r(-2,4)		
r(-1,4)	c(3,8),r(-1,4)	c(15,7),r(15,3)	c(11,7),r(15,3)	c(3,4),r(-1,4)	c(15,3),r(15,3)	c(11,3),r(15,3)	c(3,0),r(-1,4)		
r(0,4)	c(4,8),r(0,4)	c(0,8),r(0,4)	c(12,7),r(16,3)	c(4,4),r(0,4)	c(0,4),r(0,4)	c(12,3),r(16,3)	c(4,0),r(0,4)	c(0,0),r(0,4)	
r(1,4)	c(5,8),r(1,4)	c(1,8),r(1,4)	c(13,7),r(17,3)	c(5,4),r(1,4)	c(1,4),r(1,4)	c(13,3),r(17,3)	c(5,0),r(0,5)	c(1,0),r(1,4)	
r(2,4)	c(6,8),r(2,4)	c(2,8),r(2,4)	c(14,7),r(18,3)	c(6,4),r(2,4)	c(2,4),r(2,4)	c(14,3),r(18,3)	c(6,0),r(0,6)	c(2,0),r(2,4)	
r(3,4)	c(7,8),r(3,4)	c(3,8),r(3,4)	c(15,7),r(19,3)	c(7,4),r(3,4)	c(3,4),r(3,4)	c(15,3),r(19,3)	c(7,0),r(0,7)	c(3,0),r(3,4)	
r(4,4)	c(8,8),r(4,4)	c(4,8),r(4,4)	c(0,8),r(4,4)	c(8,4),r(4,4)	c(4,4),r(4,4)	c(0,4),r(4,4)	c(8,0),r(0,8)	c(4,0),r(4,4)	c(0,0),r(4,4)

PEsinaslipping mode

5. Implementation and results analysis

In this paper we propose an adaptive mtion estimation ME architecture that integrat the TSS and GS. the proposed design is based on stoppable clock architecture. To evaluate the effectiveness and to estimate the power dissipation of the proposed adaptive ME, we have modeled its architec-ture by VHDL. Experimental result of the proposed architectureis shown in Table 2. From the re-sults in table II, the proposed vlsi architecture present a less power consumption and the occupied area then the others architecture. It also presents a high clock speed which is 158 Mhz.

Table 2. Performance Results For Motion Estimator

	Algo	PEs	Freq(MHz)	Area(kgates)	Total Power (mw)
[18]	FSS/GS	9	100	11.98	4.65
[19]	TSS/HEXS	16	200	14.2	59
[17]	FS/TSS	21	17.6	34.3	1.98
Our	TSS/GS	9	158.5	10.5	3.95

Conclusions

In order to reduce power dissipation and the optimization of the time of execution in a video Codec, we have presented in this paper hardware architecture of an adaptive algorithm which inte-grates the GS an TSS using 9 PEs. This architecture is based on a stoppable clock technique that allows to shutting down each PE when it is not under running. In this study, we discussed the chal-lenges of designing an adaptive architecture such as execution time, reuse of modules, and memory access management. So the experimental results lead us to conclude that the goals of the experi-ment were achieved. Indeed, the proposed estimator works with less computing time, less area oc-cupied by FPGA and less power dissipated compared to other architectures existing in the litera-ture.

References:

[1] ITU-T SG15, "Video codec for audiovisual services at px64 kbit/s", ITU-T Recommendation H.261 Version 2, Mar. 1993.

[2] ITU-T SG16, "Video coding for low bitrate communication", ITU-T Recommendation H.263 Version 2, Feb. 1998.

[3] JTC1/SC29/WG11, "Information technology – coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s – part 2: Video", ISO/IEC 11172-2 (MPEG-1 Video), 1993.

[4] ITU-T and ISO/IEC JTC 1, "Generic coding of moving pictures and associated audio information – Part 2: Video", ITU-T Recommendation H.262 - ISO/IEC 13818-2 (MPEG-2), Nov. 1994

[5] JTC1/SC29/WG11, "Information technology – coding of audio-visual objects – Part 2: Visual", ISO/IEC 14469-2 (MPEG-4 Visual) Version 1, April 1999.

[6] ISO/IEC 14496-10 and ITU-T Rec. H.264, "Advanced Video Coding", 2003.

[7] JVT, "Advanced video coding for generic audiovisual services", Rapport technique, ISO/IEC 14496-10 and ITU-T Rec. H.264, 2003.

[8] Chen, T.C., Chien, S.Y., Huang, Y.W., Tsai, C.H., Chen, C.Y., Chen, T.W., Chen, L.G. (2006). Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder. IEEE Transactions on Circuits and Systems for Video Technology, 16(6), 673– 688.

[9] S Dhahri, A Zitouni, K Torki ,"A stoppable clock based four step search block-matching motion estimation architecture", Computer Technology and Application (CTA, 2011

[10] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast blockmatching motion estimation", IEEE Trans. Image Processing, vol. 9, pp. 287–290, 2000.

[11] S Dhahri, L Kabbai, A Zitouni, K Torki," A Low Power VLSI ASIC Design of a GS Motion Estimator ", IJCSES, 2011.

[12] S Dhahri, A Zitouni, R Tourki ," Implementation of a fast and low power 3SS algorithm for H. 264 video coding", Decision and Information Technologies (CoDIT), 2013.

[13] Zhu, C., Lin, X., Chau, L.P.," Hexagon-based search pattern for fast block motion estimation", IEEE Transactions on Circuits and Systems for Video Technology, 12(5), 349–355,2002.

[14] Ka-Ho Ng, Lai-Man Po, Ka-Man Wong, Chi-Wang Ting, and Kwok-Wai Cheung, "A Search Patterns Switching Algorithm for Block Motion Estimation", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 19, No. 5, May 2009

[15] Jinku CHOI and all., "Implementation of Motion Estimation IP Core for MPEG Encoder", International Technical Conference on Circuits/Systems, Computers and Communications , Tokushima, July 10-12,2001.

[16] Anand Paul, K. Bharanitharan and JiaJi Wu,"Algorithm and Architecture for Adaptive Motion Estimation in Video Processing", IETE Technical Review, Vol. 30, No.1,pp 24-30, Sep 2013

[17] Yang Song · Ali Akoglu"An Adaptive Motion Estimation Architecture for H.264/AVC" J. Sign Process Syst (2013) 73:161–179.

[18] S. Dhahri, A. Zitouni and K. Torki," An adaptive motion estimator design for high performances H.264/AVC codec", IJCSES International Journal of Computer Sciences and Engineering Systems, Vol. 6, No. 2, 2013.

[19] Vanne, J., Aho, E., Kuusilinna, K., Hamalainen, T.D., "A configurable motion estimation architecture for block-matching algorithms", IEEE Transactions on Circuits and Systems for Video Technology, 19(4), 466–476, 2009.