Low Power Motion Estimation with Probabilistic Computing

Charvi Dhoot[‡∏], Vincent J. Mooney[§#&∏], Lap Pui Chau[§∏] and Shubhajit Roy Chowdhury[‡]
[‡]International Institute of Information Technology, Hyderabad, India
[§]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[#]School of Computer Engineering, Nanyang Technological University, Singapore
[&]School of Electrical and Computer Engineering, Georgia Institute of Technology, Georgia, USA
[∏]Institute of Sustainable and Applied Infodynamics, Nanyang Technological University, Singapore
charvi.dhoot@research.iiit.ac.in, mooney@ece.gatech.edu, elpchau@ntu.edu.sg, src.vlsi@iiit.ac.in

***Abstract*— As Moore's law approaches the low nanometer range, predictions have been made that computing via future technology nodes may no longer be correct due to, for example, the presence of noise sources such as thermal noise. Probabilistic computing, a term coined for computing via devices that yield a probabilistically correct output, has earlier been shown as a means for realizing energy efficient computing in signal processing applications. In this paper, we explore the application of probabilistic computing for low power motion estimation. It is observed that motion estimation itself exhibits high resilience to erroneous computation. Simulations show that energy savings up to 44% can be achieved in motion estimation using full search block matching with probabilistic computing with a minor impact (under 0.5 dB) on the required quality of the output. A scheme of error correction based on the motion vector distribution is proposed that further increases the possible energy savings with full search block matching up to 57%.**

Keywords- *PCMOS architecture, probabilistic computing, error resilient design, low power design, motion estimation*

I. INTRODUCTION

Motion estimation has been a landmark innovation in video compression. It has been able to provide a tremendous breakthrough in video applications such as online conferencing, live streaming, etc., due to its ability to encode videos at very low bitrates [1]. However, motion estimation is also known to be the most computationally expensive unit of a video encoder, accounting between 66% and 94% of the encoder computational complexity [2]. This along with the growth in the popularity of handheld devices and other mobile systems using video applications places an additional demand for low power solutions to motion estimation.

It is believed that the upcoming CMOS technology will bring about a revolution in the domain of ultra-low power computing. The operability of these technologies at extremely low voltages has brought about the notion of voltage scaling as a solution for energy reduction. However, it is also predicted that with the reduction in the voltage levels, the noise immunity of these devices will decrease enough to result in variability in their output. Hence, computing through this technology may, in the future, be "probabilistically" correct. This calls for design of algorithms that are able to gain these energy savings while having the resilience with respect to the probabilistic nature of computations [3].

Probabilistic computing has been shown as an effective solution for image processing applications where unperceived degradations in the visual quality are allowed [4]. Our approach will differ in the case of motion estimation because degradation in the performance of motion estimation leads to an increase in the number of bits required for encoding. In this paper, we show techniques to lower energy consumption even with a constraint on the allowed degradation in quality.

The non-deterministic behavior of modern silicon technologies can be modeled due to various phenomena. We consider a thermal noise based model, a phenomenon which some predict will affect the performance of the future technology nodes [5]. It should be noted, however, that our approach in applying probabilistic computing is generic enough to encompass other sources of errors which can be modeled by a Gaussian distribution [6,8].

The organization of this paper is as follows. Section 2 briefly discusses some important prior work in probabilistic computing and low power motion estimation. Section 3 provides a background on motion estimation required to understand the subsequent sections. Section 4 discusses the proposed methodologies, these being the application of probabilistic computing to motion estimation and an error correction scheme based on an observed phenomenon in video sequences. Section 5 presents experimental results based on these methodologies, and the final section concludes the paper.

II. PRIOR WORK

Prior work in the domain of probabilistic computing has established that energy savings can grow exponentially with probability of error [8,9]. Furthermore, methods for modeling thermal noise based probabilistic primitives such as logic gates, adders, etc., have also been explored and established [6]-[9]. George et al. showed that probabilistic computing combined with an intelligent design of primitives may lead to a significant amount of energy savings while also minimizing the error introduced in the application [4].

Low power motion estimation is an extensively studied domain of research. Variants of efficient algorithms and architectural implementations for these algorithms have been proposed [10]-[12]. Much of this work is based on algorithmic modifications and does not consider voltage scaling as a solution for low power motion estimation. Voltage scaling for low power motion estimation with errors occurring because of delay based variations (due to, e.g., scaled supply voltage or process variations) has earlier been used by Varatkar et al. [13]. They propose an error correction scheme to achieve energy savings. Our approach uses a different noise model but mainly differs in terms of the error correction scheme proposed. The error correction scheme proposed in this paper is based on a property of real video sequences and is more suitable for full

search block matching algorithm compared to the one in [13], proposed originally for three step search algorithm.

III. BACKGROUND

Motion estimation exploits the fact that there typically exists a high degree of commonality between two adjacent frames in a video sequence. It describes the movement of objects in a frame with respect to the previous frame through motion vectors. The encoding of the frame is done through these motion vectors. At the decoder, these vectors are used to reconstruct the frame. The reconstruction process is known as motion compensation. There are various approaches followed to determine the required motion vectors. The most popularly employed approach is block-matching.

A. Motion Estimation via Block Matching

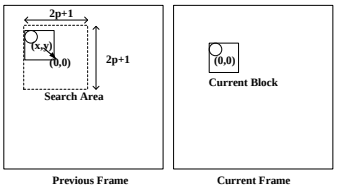


Fig. 1. Block-Matching

In a block-matching approach, the current frame in the video sequence is divided into non-overlapping macro-blocks of size N × N pixels (e.g., N=16). For each macro-block in the current frame, a displacement motion vector (MV) is calculated with respect to the best match for this block in the previous frame. The search for the matching block is constrained within a search area of size (2p+1)×(2p+1) pixels, where the parameter p varies from 7 to 31 chosen according to the type of video sequences to encode. An algorithm is used to determine the number and place of search points in the search area for an efficient search.

The criterion for arriving at the best match out of the candidate macro-blocks is sum of absolute differences (SAD). SAD is calculated by summing up the absolute difference between pixel luminance values of the current block 'a' and the corresponding pixel luminance values of the candidate block 'b'.

$$SAD = \sum_{i,j}^{N \times N} |a(i,j) - b(i,j)|$$

B. Motion Estimation in Video Encoding

The quality of motion estimation is critical to video encoding. Video encoding through motion estimation is done through motion vectors as opposed to Discrete Cosine Transform (DCT) and quantization that encode pixel values of the frame. Thus, compression through motion estimation is able to achieve much lower bitrates. The video encoding procedure is illustrated in Fig. 2. The encoded frames can be characterized as either intra or inter-coded frames. Intra coded frame or I-frames are reference frames and are encoded via DCT and quantization only. Inter coded frames or P-frames are encoded via motion estimation as well as DCT and quantization. DCT and quantization are required in P-frames to encode the difference frame, i.e., the difference between the original frame and the motion compensated frame. In this paper, we avoid usage of other inter coded frames such as B-frames to keep the discussion simple. The order in which the intra and inter-coded frames are arranged is specified by the Group of Pictures (GOP) structure. Each GOP begins with an I-frame. The number of frames between two I-frames is the GOP size. We use a GOP size of 15.

Quantization leads to lossy compression. The amount of quantization to apply for encoding is governed by the Quantization Parameter (QP). The value for QP varies from 1 to 31. When a lower bitrate is desired, QP is increased, which increases the compression loss. In the event that the performance of motion estimation degrades, the video encoder either increases QP to lower the quality but maintain a fixed bitrate, or, in applications where the quality cannot be compromised, it increases the bits required to encode the frame at a fixed QP. In most wireless and storage applications the bitrate is critical. Hence, in this paper, we consider low power motion estimation at a fixed bitrate only.

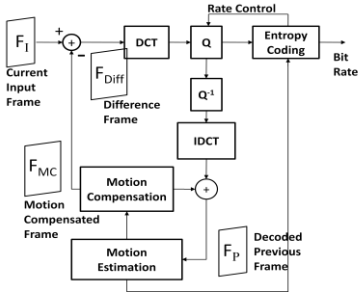


Fig. 2. Video Encoder

C. Full Search Block Matching Algorithm (FSBMA) and Systolic Array Architecture

In the full search block matching algorithm, all pixel locations in the search area are considered as candidate locations. This algorithm is preferred in video codecs such as H.264 as it yields the most optimal results in terms of the visual quality. However, it is also computationally very expensive. Due to its computational complexity, many low power architectural implementation variants for this algorithm are proposed [10]-[12]. We consider this algorithm to show the possible energy savings with probabilistic computing with minor degradation in the required quality of the output.

Systolic array architectures are widely used to model the architecture used for calculating the motion vector when the block matching algorithm used is FSBMA [14,15]. The popularity of systolic array architectures is because of the nature of computations required and the regularity in memory access for the computations required in FSBMA. The systolic array architecture for block size N=16 is illustrated in Fig. 3 [14]. Blocks AD and A contribute to calculating the SAD. Block M is a comparator that determines the minimum SAD. The functionality of the blocks AD, A and M is illustrated in Fig. 4. The systolic array architecture described thus makes up the datapath architecture for motion estimation.

IV. PROPOSED METHODOLOGY

The computational intensity of motion estimation is due to the fact that in a given video sequence the number of frames to process is very large. In a full architectural implementation for motion estimation for FSBMA, the datapath architecture accounts for 75% of the involved processing [11]. In this

section we propose a method to model a Probabilistic CMOS (PCMOS) based datapath architecture for low power motion estimation. This section also describes an error correction scheme that can further enhance the possible energy savings possible with PCMOS.

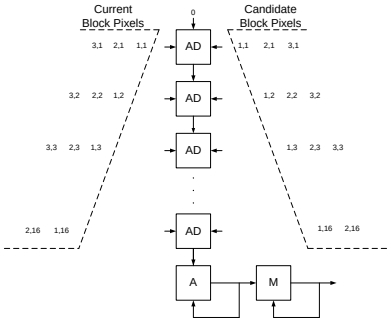


Fig. 3. Systolic Array Architecture

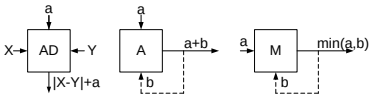


Fig. 4. Block level details for Systolic Array Architecture

A. *Modeling the PCMOS Architecture for Motion Estimation*

The systolic array architecture described in Fig. 3 for motion estimation consists of adders of bit width ranging from 8 to 16 bits. These are in turn modeled as ripple carry adders. Also, the architecture has to be modeled as a sequential circuit running on a clock to facilitate memory accesses and computations at the right timings. Thus, blocks AD, A and M require registers. At the gate level implementation, the basic building blocks of the systolic array architecture are full adders, D-flip flops, inverters and Exor gates. Inverters are used in blocks AD and M for subtraction, and Exor gates are used in block AD for absolute value computation. These gates are modeled as probabilistic gates with the exception of inverters and Exor gates which account for a small percentage (less than 10%) of the total energy consumed in the systolic array architecture described in Fig. 3.

The modeling of probabilistic gates is done through the methodology described in [6]. The methodology involves coupling a Gaussian noise source at the output of each gate [8]. The noise source is modeled as a voltage controlled voltage source with gain equal to the noise RMS. The choice of noise RMS is such that no errors are observed at the output of each probabilistic gate at the nominal supply voltage for the technology node used. Errors begin to show up at the output of the probabilistic gate when its supply voltage is scaled down. By modeling the gates in such a manner we have tried to emulate what may be observed in a future technology node, say at 16nm.

The modeling described in [6,7] shows that the error rates observed in Hspice based simulations for larger circuits such as ripple carry adders built through probabilistic gates such as full adders, can be emulated through C-based simulations. This is useful in estimating degradation in the output quality of motion estimation. Hspice simulations can be used to characterize the probability of error for a probabilistic gate at the scaled supply voltage values. The calculations made through the architecture can then be modeled in C as per the computations at the gate level in the circuit, and false bit flipping can be introduced in the C-based calculations using the probability of error for the specific gate. Thus, C based simulations can be used to determine the error tolerance of motion estimation, thereby aiding in making the choice of the appropriate operating supply voltage for the circuit. The energy consumption for the PCMOS architecture at the appropriate supply voltage is calculated using Hspice. A detailed discussion follows in Section 5.

B. *Error Correction Scheme based on Motion Vector Statistics*

In real video sequences, it is often observed that much of the scene remains stationary (background, fixed objects, etc). Also, the displacement of most of the objects in the video is usually very small over a set of adjacent frames. Based on this notion, we can postulate that a large percentage of motion vectors will be equal to (0, 0), and an even larger percentage will be concentrated within (±r, ±r), where r is a small value.

A preliminary analysis with some example video sequences showed that in a search area of size 23×23, a motion vector was found to lie within an area of 5×5 almost 50% of the time, with this value being as high as 97% in the case of videos with slow motion. This observation has been used in the past for proposing center biased searches [16,17]. We use it to propose the following scheme for correction of errors introduced by the PCMOS architecture. This scheme for error correction requires that there be two systolic arrays working at different operating voltages (resulting in increased area, see Section 5.A), one of which always works at 1.2 V while the decision on the voltage selection for the other unit is described in Section 5. The scheme works by dividing the search area into two regions as shown in Fig. 5. The candidate blocks in Fig. 5 in region 1, which is smaller and more probable to have the best matching block, are evaluated using the error-free architecture operating at 1.2 V. Region 2 in Fig. 5, outside of region 1, is evaluated using an architecture operating at a lower voltage value. The decision on the size of region 1 is made through QP that can be taken as feedback from the quantization block in the video codec. We know that QP in a video encoder, coding at a fixed bitrate, increases when the quality of motion estimation degrades. This property can be used to vary the size of region 1 as per the needs of the video sequence being encoded. The size of region 1 is controlled by the range parameter 'r'. In the steps for the error correction scheme shown below, QP_{TH}, η_1 and η_2 are parameters that facilitate the decision on what value of 'r' to choose based on the value of QP. Parameter r_1 is the maximum value that 'r' can assume. The approach we use is as follows:

Step 1: Select an initial range parameter 'r'.

Step 2: For each macro-block in the current frame,

Step 2.1: Divide the search area into two regions.

Step 2.2: Determine the best matching block in region 1 using the full search algorithm and the architecture maintained at 1.2 V.

Step 2.3: Determine the best matching block in region 2 using the full search algorithm and the architecture running at a scaled supply voltage. Calculate SAD for the winner candidate in this region with an arithmetic unit

maintained at 1.2 V; note that this arithmetic unit is already present in the proposed architecture for the scheme for calculations required in Step 2.2.

Step 2.4: Compare the SAD for the best matching blocks in Step 2.2 and Step 2.3, and select the one with the smaller SAD as the final best matching macro block. Store the motion vector corresponding to this block.

Step3: After every 5th frame, if $QP > \eta_1 \times QP_{TH}$ and $r < r_1$, then r = r+1,
else if $QP < \eta_2 \times QP_{TH}$ and $r > 0$, then r = r-1. Continue from Step 2.

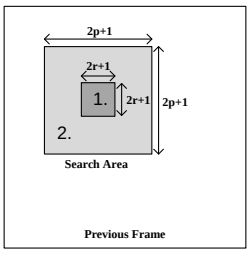


Fig. 5. Search Area Division

In the proposed scheme, QP_{TH}, η_1, η_2 and r_1 are determined empirically. The range parameter 'r' is changed after every 5th frame for the results of the previous change in 'r' to reflect in the quality of motion estimation. The error correction in this scheme is in terms of selection of the correct motion vector which is more likely to be present in region 1. Note that most video sequences have more than 50% of their motion vectors lying within a small region around the current block position. Thus, the increase in energy consumption with the error correction scheme proposed above is negligibly small.

V. RESULTS

In this section, we discuss the energy savings possible in the datapath architecture through the methodologies proposed in Section 4. Our base case for comparing the achieved energy savings is the error-free systolic array architecture described in Fig. 3. We model this architecture in Hspice in the Synopsys 90nm generic library, and operate it at the nominal supply voltage for this technology, which is 1.2 V for error-free operation. Case 1 and Case 2 correspond to uniformly voltage scaled PCMOS architecture and the proposed error correction scheme using uniformly voltage scaled PCMOS respectively.

We first describe the experimental setup for modeling the PCMOS datapath architecture and estimating the energy consumption through the architecture. Next, the experimental setup for motion estimation and the performance metrics used are explained. Also, the method used for estimating the error in the performance of motion estimation is described. Then, an analysis with some example video sequences is carried out to validate the thinking behind the error correction scheme proposed. The possible energy savings at a fixed quality for the two cases is presented. The output quality at fixed energy savings is also compared for the two schemes.

A. *Experimental Setup for PCMOS based Datapath Architecture and Energy Estimation*

The Hspice simulations of the datapath architecture were carried out using a 90nm generic library from Synopsys. The basic components in the datapath architecture were identified as full adders, D-flip flops, inverters and Exor gates. The datapath architecture was modeled at the transistor level in Hspice. A clock frequency of 125 MHz was used. This was chosen in order to process a video with frame size 352x288 at 20 fps and block size N=16 by the systolic array architecture used. The chosen frame rate and frame size are the ones required by most videophones today.

The transistor level circuit used for full adders is a 24 transistor mirror adder circuit [18], and the circuit used for D-flip flops is described in [19]. The circuit used for the D-flip flop is a dynamic edge triggered flip flop which holds data only for a short period of time. This design for the flip flop suits the requirements of the architecture used as the outputs from the registers are processed every clock cycle. Full adders and D-flip flops account for 90% of the total energy consumed. So, noise based PCMOS models based on the method described in [6] were developed only for full adders and D-flip flops. The RMS value for noise source was chosen for the components in a manner that no errors were observed at the nominal supply voltage of 1.2 V. The noise RMS value chosen was 0.2 V. This predicts a future node, e.g., 16 nm.

To calculate the probability of error for the full adder and flip flop, we simulate the PCMOS based netlists of these gates at the scaled voltage values. The range of voltage values used was 1.2, 1.15, 1.10 down till 0.5 V, the step size being 0.05 V. The lowest voltage used was 0.5V which conformed to the delay requirements of the architecture to process the video at the specified frame rate. The operability of the current technology nodes, e.g., 90nm for voltage values lower than 0.5 V has been shown in [20]. At the scaled supply voltage, the blocks AD, A and M in the architecture become erroneous. However, block M is a comparator and needs to make correct decisions. Hence, block M was maintained at the supply voltage of 1.2 V. The output of block A was level shifted to the voltage value of 1.2 V using level shifters. The circuit used for level shifting is described in [21].

The energy consumption through the architecture was calculated using Hspice. The inputs to the architecture were test vectors from the video sequences used to show the energy savings. The video sequences used are described in the following sections. Case 1 requires the components AD and A of the architecture to run only at one of the voltage levels specified, and block M to run at 1.2 V. Hence, energy consumption was calculated by simulating the entire systolic array architecture circuit by scaling down the voltage supply uniformly for all components of blocks AD and A to the required voltage level and using level shifters before block M in Hspice. Case 2 requires two parallel architectures maintained at different voltage levels, 1.2 V and a lower voltage value. If the average power consumed by the two architectures circuits for Case 2 at the different supply voltages is P1 and P2, and the percentage of calculations made through the two architectures, determined by the range parameter 'r', are C1 and C2, then the energy consumption for Case 2 was estimated as $(P1 \times C1 + P2 \times C2) \times T$, where T is time period of clock cycle required per calculation. The values of C1 and C2 were determined over 100 frames of the video sequence for a good estimate. Note that static power is less than 1%.

Note that for Case 2 additional control logic will be required to multiplex the data from memory between the two systolic arrays, i.e., pixel data of the search area pixels in region 1 (Fig. 5) and current block data to be given to systolic array-1 working at 1.2 V, and pixel data in region 2 (Fig. 5) to be given to systolic array-2 working at a lower voltage. The control logic will have to facilitate one additional block memory access required for the calculation of the final motion vector (Step 2.3 in Section 4.B). Also, the area of the datapath architecture will be approximately doubled because of the presence of two systolic arrays as shown in Fig. 6. In the energy savings quoted in this paper, we account only for the energy savings in the datapath architecture as it makes up for most (about 75%) of the involved processing in case of full search algorithm [11].

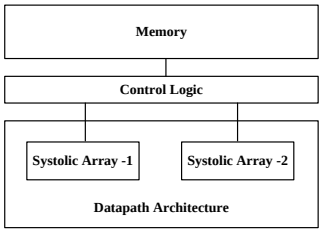


Fig. 6. Motion Estimation Architecture Details for Case 2

B. Experimental Setup for Motion Estimation

The input video sequences to motion estimation are CIF video sequences of size 352x288 at 20 fps. Four video sequences with motion characteristics varying from slow to fast have been considered to show the results. The parameter p for the search area is chosen to be 11, appropriate for the video sequences considered. The simulations are carried out in the MPEG-2 Test Model 5 codec, at main profile and main level [22]. The GOP size used is 15. The performance of the two cases is evaluated by fixing the average quality degradation ΔPSNR of the motion compensated frame within 0.5 dB from the base case. PSNR is calculated as

$$PSNR = 10 \times \log\left(\frac{255^2}{(1/(H \times W))\sum_{i,j}^{H,W}(F_I(i,j) - F_{MC}(i,j))^2}\right)$$

where H and W are the dimensions of the frame. $F_I(i,j)$ and $F_{MC}(i,j)$ are the pixel luminance values for the input and the motion compensated frames. To estimate the performance of motion estimation for the PCMOS architecture, we perform C-based simulations. The 8-bit pixel inputs from the current and candidate blocks are converted into binary values. The calculations are then performed as they would occur in any hardware according to the manner in which logic gates are connected in the circuit. False bit flipping is introduced for full adders and D-flip flops as per the probability of errors characterized earlier through Hspice simulations [6,7]. The final output from the circuit is converted back to its decimal value to be used in the motion estimation code. For Case 2, QP is made available by the MPEG-2 codec.

C. Motion Vector Distribution

The motion vector distribution for some standard video sequences characterized by different motion types was analyzed. Table 1 shows the percentage of motion vectors for these video sequences found within (±r, ±r), where r is varied from 0 to 2. The analysis in Table 1 shows that a significant percentage of motion vectors exist within a range of (±2, ±2). Even sports video sequences such as 'Stefan', which are characterized by fast motion, exhibit this characteristic. This analysis validates the idea behind the error correction scheme proposed in Section 4.B.

Table 1. Motion Vector Distribution

MV	(0,0)	(±1,±1)	(±2,±2)
Susie	59.13%	87.69%	91.49%
Mobile Calendar	26.7%	96.54%	97.55%
Flower Garden	7.5%	54.08%	79.02%
Stefan	36.32%	48.12%	54.79%

D. Experimental Results

Table 2 shows the energy savings possible in the datapath architecture with Case 1 and Case 2 when degradation is constrained to be within 0.5 dB. The circuit supply voltage is varied for Case 1 for different video sequences to see the best possible energy savings. However, for implementation, an on the fly decision on the circuit supply voltage for different video sequences may not be practical. For Case 2, we fix the supply voltage for the two parallel Systolic Arrays (SA1 and SA2) at 1.2 V and 0.55 V. The quality degradation constraint of 0.5 dB is achieved in Case 2 by varying the range parameter according to QP. In a video codec, QP is different for each macro block in a frame. We use the average QP for a frame, which is derived by averaging over the QP values for the macro-blocks in the frame [22]. We multiply the average QP for the frame with the number of macro-blocks in the frame, in this case 396. The scaling of average QP by number of macro-blocks provides more accurate results and is not required to be calculated separately as this value can be taken from the MPEG codec. We set QP_{TH}, η_1 and η_2 as QP_{mean}, 1.0200 and 1.0065 respectively, where QP_{mean} is the mean QP for a frame in the codec when operated for the error-free architecture, scaled by the number of macro blocks in a frame (in this case 396). The upper limit on 'r', i.e., 'r_1' is set as 4.

Fig. 7 illustrates the improvement in average PSNR and visual quality of the motion compensated frames when the energy consumption through Case 1 and Case 2 is approximately same. The architecture in Case 1 is maintained at a supply voltage of 0.65 V for the comparison, and the two parallel architectures for Case 2 are maintained at 1.2 V and 0.55 V. Both cases account for same energy savings of approximately 57% in Fig. 7.

6. CONCLUSION

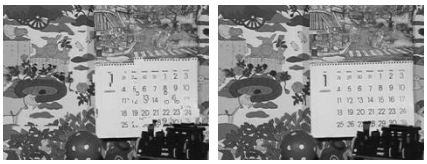
It is shown that probabilistic computing provides an effective solution to low power motion estimation. Furthermore, a novel scheme proposed in this paper is able to enhance the energy savings to as high as 57% energy and is able to improve the PSNR by about 1.2X over probabilistic computing alone. The paper thus shows that algorithmic solutions such as the error correction scheme proposed in this paper can be used to further exploit the achievable energy savings via probabilistic computing.

Table 2. Energy savings with PSNR loss less than 0.5 dB

Video Sequences	Base Case: No energy savings PSNR (dB)	Case 1			Case 2			
		Avg. PSNR (dB)	Energy Savings	Circuit Supply Voltage (V)	Avg. PSNR (dB)	Energy Savings	Average range parameter 'r'	Circuit Supply Voltage (V) (SA1, SA2)
Susie	35.74	35.35	34%	0.95	35.23	56%	2.73	1.2, 0.55
Mobile Calendar	23.82	23.44	44%	0.85	23.36	57%	2.18	1.2, 0.55
Flower Garden	25.69	25.37	44%	0.85	25.22	57%	2.42	1.2, 0.55
Stefan	25.64	25.27	44%	0.85	25.12	52%	4.00	1.2, 0.55

PSNR = 30.08 dB PSNR = 35.23 dB

(a)



PSNR = 20.92 dB PSNR = 23.36 dB

(b)

PSNR = 23.13 dB PSNR = 25.22 dB

(c)

Fig. 7. Motion compensated frames for three different video sequences (a) Susie (b) Mobile Calendar and (c) Flower Garden for Case 1 (left) and Case 2 (right) at approximately same energy savings of 57%

REFERENCES

[1] "H.264. http://iphome.hhi.de/suehring/tml/"

[2] P. Kuhn, *Complexity Analysis and VLSI Architectures for MPEG-4 Motion estimation*, Boston, MA: Kluwer, 1999.

[3] K. V. Palem, "Energy Aware Algorithm Design via Probabilistic Computing: From Algorithms and Models to Moore's law and Novel (Semiconductor) Devices," *Proceedings of the International conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES'03)*, pp. 113-116, Oct. 2003.

[4] J. George, B. Marr, B. E. S. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," *Proceedings of the 2006 International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, pp. 158-168, Oct. 2006.

.[5] L. B. Kish, "End of Moore's law: Thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 3-4, pp. 158-168, 2002.

[6] A. Singh, A. Basu, K.V. Ling, and V. Mooney, "Modeling Multi-output Filtering Effects in PCMOS," *Proceedings of the VLSI Design and Test Conference (VLSIDAT 2011)*, April 2011.

[7] A. Bhanu, M. S. K. Lau, K. V. Ling, V. J. Mooney III, and A. Singh, "A More Precise Model of Noise Based PCMOS Errors," *Fifth IEEE International Symposium on Electronic Design, Test & Applications (DELTA '10)*, pp. 99-102, Jan. 2010.

[8] Suresh Cheemalavagu, Pinar Korkmaz, Krishna V. Palem, Bilge E. S. Akgul, and Lakshmi N. Chakrapani, "A Probabilistic CMOS switch and its Realization by Exploiting Noise," *Proceedings of IFIP-VLSI SoC* , Oct. 2005.

[9] S. Cheemalavagu, P. Korkmaz, and K. V. Palem, "Ultra low-energy computing via probabilistic algorithms and devices: CMOS device rimitives and the energy-probability relationship," *Proceedings of the 2004 International Conference on Solid State Devices and Materials*, pp. 402–403, Sep. 2004.

[10] M. A. Elgamel, A. M. Shams, and M. A. Bayoumi, "A comparative analysis of low power motion estimation VLSI architectures," *in Proceedings of IEEE Workshop Signal Processing*, pp. 149-158, 2000.

[11] R. S. Richmond II and D. S. Ha, "A low-power motion estimation block for low bit-rate wireless video," *Proceedings of IEEE Workshop Signal Processing*, pp. 60-63, Aug. 2001.

[12] S. S. Lin, P. C. Tseng, and L. G. Chen, "Low-power parallel tree architecture for full search block-matching motion estimation," *Proceedings of the 2004 International Symposium on Circuits and Systems (ISCAS '04)*, vol. 2, pp. II - 313-316, May 2004.

[13] G. V. Varatkar and N. R. Shanbhag, "Error-Resilient Motion Estimation Architecture," *IEEE Transactions on VLSI Systems*, vol. 16, no. 10, pp. 1399-1412, Oct. 2010.

[14] T. Komarek and P. Pirsch, "Array architectures for block matching algorithms," *IEEE Transactions on Circuits and System*, vol. 36, no. 10, pp. 1301-1308, Oct. 1989

[15] Liang Lu, J.V. Mc Canny, and S. Sezer, "Systolic Array Based Architecture for Variable Block-Size Motion Estimation," *Adaptive Hardware and Systems (AHS)*, pp. 160-165, Aug. 2007.

[16] R. Li, B. Zeng, and M. L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438-442, Aug. 1994.

[17] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, "A Novel Unrestricted Center-Biased Search for Block Motion Estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 369-377, Aug. 1998.

[18] J. M. Rabaey, A. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 3rd ed. Prentice Hall, 2003.

[19] http://en.wikipedia.org/w/index.php?title=File:TSPC_FF_R.png *License:* Creative Commons Attribution 3.0, *Contributors*: Jon Guerber.

[20] R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and Threshold Voltage Scaling for Low-Power CMOS," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 3, pp. 395–400, Mar. 1999.

[21] Kyoung-Hoi Koo, Jin-Ho Seo, Myeong-Lyong Ko, and Jae-Whui Kim, "A New Level-up Shifter for High Speed and Wide Range Interface in Ultra Deep Sub-Micron", *IEEE International Symposium on Circuits and Systems, 2005, (ISCAS '05)*, vol. 2, pp. 1063- 1065, May 2005.

[22] "MPEG-2 Test Model-5. http://www.mpeg.org/MPEG/MSSG/tm5