An Efficient VLSI-EDDR Architecture for Motion Estimation in Testing Applications

1 G. Theivanathan,  2 Mr. P. Govindamoorthi  3 A. Vinothkumar
1 Assistant Professor, Dept. of ECE  P.S.R. Engineering College, Sivakasi  P.S.R. Engineering College, Sivakasi
Velammal Engineering College, Chennai  P.S.R. Engineering College, Sivakasi
Email: theiva.nathang@gmail.com, govindamoorthi@psr.edu.in

Abstract—Block based motion estimation is one of the critical tasks in today video compression standards such as H.26L, MPEG-1, -2 and -4 standards. With the advent of VLSI technology, a large collection of 4x4 processing elements can be assembled to achieve high-speed computation economically. While focusing on the testing of motion estimation (ME) in a video coding system, this work presents very large scale integration (VLSI) of an error detection and recovery (EDDR) architecture, based on the residue-and-quotient (RQ) code, to embed into ME for video coding testing applications. An error in processing elements (PEs), i.e. key components of a ME, can be detected and recovered effectively by using the proposed VLSI-EDDR architecture.

Keywords—Motion Estimation (ME), VLSI-Error Detection and Data Recovery (EDDR), Residue-and-Quotient (RQ) code, Processing Elements (PEs).

I. INTRODUCTION

In video coding, similarities between video frames can be exploited to achieve higher compression ratios. However, moving objects within a video scene diminish the compression efficiency of the straightforward approach that only considers pixels located at the same position in the video frames. A good example is the H.264 video standard, also known as MPEG-4 Part 10 Advanced Video Coding, which is widely regarded as the next generation video compression standard [1]. A flexible, reconfigurable and programmable motion estimation processor, such as the one proposed in this work, is well poised to address these challenges by allowing the core microarchitecture to be optimized alongside the estimation algorithm. This concept was briefly introduced in [2] and has been further developed and improved upon in this work to consider the advanced features of H.264: fractional-pixel search, variable partition sizes and rate-distortion optimization. A ME-motion estimation generally consists of PEs with a size of $4 \times 4$. However, accelerating the computation speed depends on a large PE array, especially in high-resolution devices with a large search range such as High Definition TV (HDTV) [3]. As a commercial chip, it is absolutely necessary for the ME to introduce design for testability (DFT). DFT focuses on increasing the ease of device testing, thus guaranteeing high reliability of a system. DFT methods rely on reconfiguration of a circuit under test (CUT) to improve testability[4]. Fault models are necessary for generating and evaluating a set of test vectors. Generally, an good fault model should satisfy two criteria: (1) It should accurately reflect the behavior of defects, and (2) it should be computationally efficient in terms of fault simulation and test pattern generation. Additionally, the visual quality and peak signal-to-noise ratio (PSNR) at a given bit rate are influenced if an error occurred in ME process. A testable design is thus increasingly important to ensure the reliability of numerous PEs in a ME. Although the advance of VLSI technologies facilitates the integration of a large number of PEs of a ME into a chip, the logic-per-pin ratio is subsequently increased, thus decreasing significantly the efficiency of logic testing on the chip. As a commercial chip, it is absolutely necessary for the ME to introduce design for testability (DFT) [5]–[7]. Thus, extended schemes of BIST referred to as built-in self-diagnosis [8] and built-in self-correction [9]–[10] have been developed recently.

The rest of this paper is organized as follows. Section II describes the SAD Tree. Section III describes the mathematical model of RQ code and the corresponding circuit design of the RQ code generator (RQCG). Section IV then introduces the proposed VLSI-EDDR architecture, fault model definition, and test method. Next, Section V evaluates the performance in simulation of the proposed EDDR architecture for ME testing applications. Conclusions are finally drawn in Section VI.

II. SAD TREE

Variable Block Size Motion Estimation (VBSME) has been adopted in the latest video coding standards, including H.263, MPEG-4, WMV9.0, and H.264/AVC. For instance, in H.264/AVC, an MB with a variable block size can be divided into seven kinds of blocks including 4x4, 4x8, 8x4, 8x8, 8x16, 16x8, and 16x16. Fixed Block Size Motion Estimation (FBSME), and they can be classified into two categories. One is an inter-level architecture, where each processing element (PE) is responsible for one SAD of a specific searching candidate and the other is an intra-level architecture, where each PE is responsible for the distortion of a specific current pixel. The concept of the proposed SAD Tree architecture is a 2-D intra-level architecture and consists of a 2-D PE array and one 2-D adder tree with propagation registers. Current pixels are stored in each PE, and reference pixels are stored in...
propagation registers for data reuse. In each cycle, current and reference pixels are inputted to PEs. Simultaneously, continuous reference pixels in a row are inputted into propagation registers to update reference pixels. In propagation registers, reference pixels are propagated in the vertical direction row by row. In SAD Tree architecture, all distortions of a searching candidate are generated in the same cycle, and by an adder tree, distortions are accumulated to derive the SAD in one cycle. In order to provide a high utilization and data reuse, the snake scan is adopted and reconfigurable data path propagation registers are developed in the proposed SAD Tree, which consists of five basic steps from A to E. The first step, A, fetches pixels in a row and the shift direction of propagation registers is downward. When calculating the last candidates in a column, one extra reference pixel is required to be inputted, that is, step B. When finishing the computation of one column, the reference pixels in the propagation registers are shifted left in step C. Because the reference data have already been stored in the propagation registers, the SAD can be directly calculated. The next two steps, D and E, are the same as steps A and B except that the shift direction is upward. After finishing the computation of one column in the search range, we execute step C and then go back to step A. This procedure will iterate until all searching candidates in the search range have been calculated the data reuse between two successive searching candidates can be maximized.

![Fig.1. SAD Tree](image)

Moreover, a more comprehensive fault model, i.e. the stuck-at (SA) model, must be adopted to cover actual failures in the interconnect data bus between PEs. The SA fault is a well-known structural fault model, which assumes that faults cause a line in the circuit to behave as if it were permanently at logic “0” [stuck-at 0 (SA0)] or logic “1” [stuck-at 1 (SA1)]. The SA fault in a ME architecture can incur errors in computing SAD values. A distorted computational error(e) and the magnitude of e are assumed here to be equal to SAD’-SAD where SAD’ denotes the computed SAD value with SA faults.

III. RESIDUE QUOTIENT (RQ) CODE GENERATION

Residue code is generally separable arithmetic codes by estimating a residue for data and appending it to data. To detect circuit errors Coding approaches such as parity code, Berger code, and residue code have been considered for design applications. For instance, assume that N denotes an integer, and N1 and N2 represent data words, and m refers to the modulus. A separate residue code of interest is one in which N is coded as a pair (N1,N2|m). Notably, |N|m is the residue of N modulo m Error detection logic for operations is typically derived using a separate residue code such that detection logic is simply and easily implemented. However, only a bit error can be detected based on the residue code. Additionally, an error cannot be recovered effectively by using the residue codes. Therefore, this work presents a quotient code, which is derived from the residue code, to assist the residue code in detecting multiple errors and recovering errors. The mathematical model of RQ code is simply described as follows. Assume that binary data is expressed as

\[ X = \{b_{n-1} b_{n-2} \ldots b_1 b_0\} = \sum_{i=0}^{n-1} b_i 2^i \]  

The RQ code for X modulo m is expressed as R = \[\lfloor X/m\rfloor\] and Q = \[\{X/m\}\] respectively. Notably, \[\lfloor . \rfloor\] denotes the largest integer not exceeding \(i\).

According to the above RQ code expression, the corresponding circuit design of the RQCG can be realized. In order to simplify the complexity of circuit design, the implementation of the module is generally dependent on the addition operation. Additionally, based on the concept of residue code, the following definitions shown can be applied to generate the RQ code for circuit design.

Definition 1:
\[ |N1+ N2|m = ||N1|m + |N2|m \]  

Definition 2: Let, N j = n1+n2+.....+ n j, then
\[ |N||m = |n1|m+|n2|m+.....+|nj|m|m \]

To accelerate the circuit design of RQCG, the binary data of equation (1) can generally be divided into two parts:

\[ X = \sum_{i=0}^{n-1} b_i 2^i \]  

\[ = \left( \sum_{i=0}^{k-1} b_i 2^i \right) + \left( \sum_{i=k}^{n-1} b_i 2^{i-k} \right) \]  

\[ = Y_0 + Y_1 2^k \]  

(4)

Significantly, the value of \(k\) is equal to \(\lfloor y/2 \rfloor\) and the data formation of \(Y_0\) and \(Y_1\) are a decimal system. If the modulus \(m\) = \(2^k-1\), then the residue code of X modulo m is given by,

\[ R = \{X|m\} \]  

\[ = |Y_0 + Y_1|m = |Z_0 + Z_1|m \]  

\[ = (Z_0 + Z_1)(5) \]

\[ Q = \{X|m\} \]
\[ Y = \frac{Y_0 + Y_1}{m} + Y_1 \]
\[ Z = \frac{Z_0 + Z_1}{m} + Z_1 + Y_1 + \beta \]

Where
\[
\alpha(\beta) = \begin{cases} 
0(1), & \text{if } Z_0 + Z_1 = m \\
1(0), & \text{if } Z_0 + Z_1 < m 
\end{cases}
\]

Since the value of \( Y_0 \) and \( Y_1 \) is generally greater than that of modulus \( m \), the equations in (5) and (6) must be simplified further to replace the complex module operation with a simple addition operation by using the parameters \( Z_0, Z_1, \alpha \) and \( \beta \) [10]. Based on (5) and (6), the corresponding circuit design of the RQCG is easily realized by using the simple adders (ADDS). Namely, the RQ code can be generated with a low complexity and little hardware cost.

**IV. PROPOSED VLSI-EDDR ARCHITECTURE**

Fig. 2 shows the conceptual view of the proposed EDDR scheme, which comprises two major circuit designs, i.e. error detection circuit (EDC) and data recovery circuit (DRC), to detect errors and recover the corresponding data in a specific CUT. The test code generator (TCG) in Fig. 2 utilizes the concepts of RQ code to generate the corresponding test codes for error detection and data recovery. In other words, the test codes from TCG and the primary output from CUT are delivered to EDC to determine whether the CUT has errors. DRC is in charge of recovering the corresponding data. This work adopts the systolic ME as a CUT to demonstrate the feasibility of the proposed EDDR architecture. A ME consists of many PEs incorporated in a 1-D or 2-D array for video encoding applications. A PE generally consists of two ADDs (i.e. an 8-b ADD and a 12-b ADD) and an accumulator (ACC). Next, the 8-b ADD (a pixel has 8-b data) is used to estimate the addition of the current pixel (Cur_pixel) and reference pixel (Ref_pixel). Additionally, a 12-bit ADD and an ACC are required to accumulate the results from the 8-bit ADD in order to determine the sum of absolute difference (SAD) value for video encoding applications. Notably, some registers and latches may exist in ME to complete the data shift and storage. Fig. 4 shows an example of the proposed EDDR circuit design for a specific PE of a ME. The fault model definition, RQCG-based TCG design, operations of error detection and data recovery are described carefully as follows.

**A. TCG Design**

TCG is the main block of the proposed EDCA design. Test Code Generation (TCG) design is based on the ability of the RQCG circuit to generate corresponding test codes in order to detect errors and recovers data. The specific PEi estimates the absolute difference between the Cur_pixel of the search area and the Ref_pixel of the current macroblock. The SAD value for the macroblock with size of \( N \times N \) can be evaluated:

\[
SAD = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |X_{ij} - Y_{ij}|
\]

\[
= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |(q_{xij} \cdot m + r_{xij}) - (q_{yij} \cdot m + r_{yij})|
\]

where \( r_{xij} \), \( q_{xij} \) and \( r_{yij} \), \( q_{yij} \) denote the corresponding RQ code of \( X_{ij} \) and \( Y_{ij} \) modulo \( m \). Importantly, \( X_{ij} \) and \( Y_{ij} \) represent the luminance pixel value of Cur_pixel and Ref_pixel, respectively. Based on the residue code, the definitions shown in (2) and (3) can be applied to facilitate generation of the RQ code ( \( R_T \) and \( Q_T \) ) form TCG. Namely, the circuit design of TCG can be easily achieved by using

\[
R_T = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij}) \quad m
\]
\[ Q_T = \left\lfloor \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij})}{m} \right\rfloor \quad (8) \]

**B. PROPOSED VLSI-EDDR ARCHITECTURE PROCESSES**

Fig. 3 clearly indicates that the operations of error detection in a specific PE, is achieved by using EDC, which is utilized to compare the outputs between TCG and RQCG in order to determine whether errors have occurred. If the values of \( R_{PEi} \neq R_T \) and/or \( Q_{PEi} \neq Q_T \), then the errors in a specific PE, can be detected. The EDC output is then used to generate a 0/1 signal to indicate that the tested PE, is error-free/errancy. This work presents a mathematical statement to verify the operations of error detection. Based on the definition of the fault model, the SAD value is influenced if either SA1 and/or SA0 errors have occurred in a specific. In other words, the SAD value is transformed to \( SAD' = SAD + e \) if an error occurred. Notably, the error signal is expressed as

\[ e = q_e \cdot m + r_e \quad (9) \]

However, \( R_{PEi} \) and \( Q_{PEi} \) are changed to (12) and (13) because of the error has occurred. Thus, the error in a specific PE, can be detected if and only if \( (7) \neq (10) \) and/or \( (8) \neq (11) \):

\[ R_{PEi} = \left\lfloor SAD' \right\rfloor m \]
\[ = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij}) + e \quad m \]
\[ = \left\lfloor r_{00} \right\rfloor m + \left\lfloor r_{01} \right\rfloor m + \cdots + \left\lfloor r_{(N-1)(N-1)} \right\rfloor m + \left\lfloor r_e \right\rfloor m \quad (10) \]

\[ Q_{PEi} = \left\lfloor SAD' \right\rfloor m \]
\[ = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij}) + e \quad m \]
\[ = q_{00} + q_{01} + \cdots + q_{(N-1)(N-1)} + q_e + \left( r_{00} + r_{01} + \cdots + r_{(N-1)(N-1)} + r_e \right) \quad (11) \]

During data recovery, the circuit DRC plays a significant role in recovering RQ code from TCG. The data can be recovered by implementing the mathematical model as

\[ SAD = m \times Q_T + R_T \]
\[ = (2^k - 1) \times Q_T + R_T \]
\[ = 2^k \times Q_T - Q_T + R_T \quad (12) \]

**C. SAMPLE CALCULATION**

A numerical example of the 16 pixels for a 4 \times 4 macroblock in a specific PE, of a ME is described as follows.

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
1 & 128 & 128 & 64 & 255 \\
2 & 128 & 64 & 255 & 64 \\
3 & 255 & 64 & 128 & 128 \\
\end{array}
\]

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
1 & 1 & 2 & 3 & 4 \\
2 & 2 & 3 & 4 & 5 \\
3 & 3 & 4 & 5 & 5 \\
\end{array}
\]

Fig. 4. Sample values

Fig. 4 presents an example of pixel values of the Cur_pixel and Ref_pixel. The SAD value of the 4 \times 4 macroblock is

\[
SAD = \sum_{i=0}^{3} \sum_{j=0}^{3} |X_{ij} - Y_{ij}| = |X_{00} - Y_{00}| + |X_{01} - Y_{01}| + \cdots + |X_{33} - Y_{33}| = (128 - 1) + (128 - 1) + \cdots + (128 - 5) = 2124 
\]

According to the description of RQ code in Section II, the modulo is assumed here to be \( m = 2^k - 1 = 63 \). Thus based above mentioned equation, the RQ code of the SAD values \( R_T = R_{PEi} = 2124/63 = 45 \) and \( Q_T = Q_{PEi} = 2124/63 = 33 \). Since the value of \( R_T(Q_T) \) is equal to \( R_{PEi}(Q_{PEi}) \), EDC is enabled and a signal “0” is generated to describe a situation in which the specific PE, is error free. Conversely if SA1 and SA0 errors in bits 1 and 12 of a specific PE, the pixel values of PE, 2124=10000101011002 is turned into 77=0000010011012 resulting transformation of the RQ code of \( R_{PEi} \) and \( Q_{PEi} \) into \( 77/63 = 14 \) and \( 77/63 = 1 \). Thus, an error signal “1” is generated from EDC and sent to MUX in order to select the recovery data from DRC.

**V. SIMULATION RESULTS:**

Fig. 5. SAD Tree
VI. CONCLUSION

In this work EDDR circuit is designed for detecting the errors and recovering the data of PEs in a ME. The proposed EDDR architecture is also implemented by using Verilog and synthesized by the ModelSim and Xilinx. The EDDR circuit is designed with reasonable area overhead and only a slight time penalty. The complexity of RQCG code generation is decreased so testing time is greatly reduced.

REFERENCES


