iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2 **Issue:** I **Month of publication:** January 2014

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

Built in self Auto Detection/Correction Architecture Through Motion Estimation Arrays

P.Sunil Kumar[#1], Patibandla Swapna [#2,]
Hyderabad(A.P),India

***Abstract:** This work develops a built-in self-detection/correction (BISDC) architecture for motion estimation computing arrays (MECAs). Based on the error detection/correction concepts of biresidue codes, any single error in each processing element in an MECA can be effectively detected and corrected online using the proposed BISD and built-in self-correction circuits. Performance analysis and evaluation demonstrate that the proposed BISDC architecture performs well in error detection and correction with minor area overhead. The objective of DFT is to increase the ease with which a device can be tested to guarantee high system reliability. Among these techniques, BIST has an obvious advantage in that expensive test equipment is not needed and tests are low cost. Moreover, BIST can generate test simulations and analyze test responses without outside support, making tests and diagnoses of digital systems quick and effective. However, as the circuit complexity and density increases, the BIST approach must detect the presence of faults and specify their locations for subsequent repair. The extended techniques of BIST are built-in self-diagnosis and built-in self-re- pair (BISR).*

1. INTRODUCTION

As the multimedia and wireless technologies become mature, more and more sophisticated portable multimedia applications, such as video cellular phone and hand-held digital video camcorders, are becoming available. Real-time video compression is required to reduce the bandwidth, either for transmission or for storage of video data. However, it consumes a lot of power. Although advances in VLSI technology allow integration of a large number of processing elements (PEs) in an MECA into an SOC, this increases the logic-per-pin ratio, thereby significantly decreasing the efficiency of chip logic testing. For a commercial chip, a video coding system must introduce design for testability (DFT), especially in an MECA. The objective of DFT is to increase the ease with which a device can be tested to guarantee high system reliability. Many DFT approaches have been developed. These approaches can be divided into three categories: ad hoc (problem oriented), structured, and built-in self-test (BIST). Among these techniques, BIST has an obvious advantage in that expensive test equipment is not needed and tests are low cost. Several motion estimation algorithms and VLSI (very large scale integration) architectures have been proposed for practical video coding applications. These algorithms are almost based on the spatial domain. Therefore, some characteristics of video information in the transform domain cannot be used to further reduce the computation complexity domain cannot be used to further reduce the computation complexity. Although BIST and BISR are utilized in many studies, most studies focused on memory testing. And also because of advances in VLSI technology the number of processing elements integrated on to the silicon chip is increased rapidly. So the no of pins for chip also increases more and more because of that the testability of the chip is become more complex. In order to increase the testability we are introducing the DFT here. The objective of DFT (Design for testability) is to increase the ease with which a device can be tested to guarantee high system reliability.

The use of residue codes to detect error is a useful approach in computer Arithmetic. Arithmetic codes have been found to be useful in the detection and correction of errors in arithmetic operations as well as communication. Arithmetic Code is a selected set of code words or code numbers each of, digit say, n digits (in some radix r representation). For each integer N from the information set Zm= {0, 1...m-1}, there is a corresponding code word. Errors can be detected or corrected in arithmetic operations using these code words. Based on the preceding, arithmetic codes fall into these three major classes:

1. AN code which are nonsystematic and therefore non separate.
2. Separate codes with one or more residue checks, for example the (N, N|A, N| B) biresidue code.
3. And the systematic sub codes, which are also called systematic non separate codes.

The AN codes were introduced first by Diamond and their detection- and error-correction properties were discussed by Brown and Peterson. However, their effectiveness (fault coverage), and applicability to all arithmetic-related operations, such as shift and cycle, have not been established. The separate codes using a single residue check, such as (N, NA) code, can only provide error detection for all arithmetic- related operations, but not correction, and therefore are of limited value. In order to obtain error correction by use of separate codes, two or more residue checks are required, and that has led to the introduction of biresidue and multi residue codes.

2.1 Motion Estimation Computing Arrays

Generally, motion estimation computing array (MECA) performs up to 50% of computations in the entire video coding system (VCS).In VCS, Video data needs to be compressed before storage and transmission, complex algorithms are required to eliminate the redundancy, extracting the redundant information.Motion Estimation (ME) is the process of creating motion vectors to track the motion of objects within video footage. It is an essential part of many compression standards and is a crucial component of the H.264 video compression standard .In particular; ME can consist of over 40% of the total computation.

The name motion picture comes from the fact that a video, once encoded, is nothing but a sequence of still pictures that are shown at a reasonably high frequency. That gives the viewer the illusion that it is in fact a continuous animation. Each frame is shown for one small fraction of a second, more precisely 1/ *k* seconds, where *k* is the number of frames per second. Coming back to the definition of a scene, where the frames are captured without interruption, one can expect consecutive frames to be quite similar to one another, as very little time is allowed until the next frame is to be captured. With all this in mind we can finally conclude that each scene is composed of at least $3 \times k$ frames (since a scene is at least 3 seconds long). In the NTSC case, for example, that means that a movie is composed of a sequence of various segments (scenes) each of which has at least 90 frames similar to one another.

Macro blocks are then organized in “groups of blocks” (GOBs) which are grouped in pictures (or in layers and then pictures). Pictures are further grouped in scenes, as described above, and we can consider scenes grouped as movies. Motion estimation is often performed in the macro block domain. For simplicity’ sake we’ll refer to the macro blocks as blocks, but we shall remember that most often the macro block domain is the one in use for motion estimation.

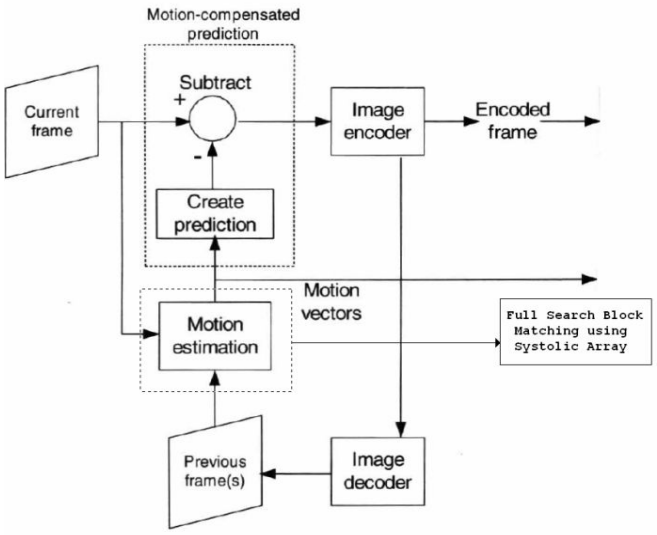


Fig 2.1: Video Encoding System

For motion estimation the idea is that one block B of a current frame C is sought for in a previous (or future) frame R. If a block of pixels which is similar enough to block B is found in R, then instead of transmitting the whole block just a “motion vector” is transmitted. Ideally, a given macro blocks would be sought for in the whole reference frame; however, due to the computational complexity of the motion estimation stage the search is usually limited to a pre-defined region around the macro blocks. Most often such region includes 15 or 7 pixels to all four directions in a given reference frame. The search region

is often denoted by the interval [-P, P] meaning that it includes P pixels in all directions.

The distortion measure is Sum of Absolute Difference for its simplicity, in which the candidate block with minimum amount of distortion is considered as the best-match. To achieve a best trade-off between the computational complexity of FSBM and degraded PSNR of motion compensated frame using faster algorithms, recently some researchers have investigated reduction of computational complexities of *FSBM*. All these algorithms are not optimal in the sense that instead of exhaustive search, only some fixed positions are searched, based on the predictions of motion. Any error in motion prediction may lead to wrong motion vectors, resulting in poor peak signal-to-noise ratio *(PSNR)* of the motion-compensated frame.

2.2 Motion estimation

Motion estimation is the process of determining motion vectors that describe the transformation from one 2D image to another; usually from adjacent frames in a video sequence. It is an ill-posed problem as the motion is in three dimensions but the images are a projection of the 3D scene onto a 2D plane. The motion vectors may relate to the whole image (global motion estimation) or specific parts, such as rectangular blocks, arbitrary shaped patches or even per pixel. The motion vectors may be represented by a translational model or many other models that can approximate the motion of a real video camera, such as rotation and translation in all three dimensions and zoom. Closely related to motion estimation is optical flow, where the vectors correspond to the perceived movement of pixels. In motion estimation an exact 1:1 correspondence of pixel positions is not a requirement. Applying the motion vectors to an image to synthesize the transformation to the next image is called motion compensation. The combination of motion estimation and motion compensation is a key part of video compression as used by MPEG 1, 2 and 4 as well as many other video codes. The methods for finding motion vectors can be categorized into pixel based methods ("direct") and feature based methods ("indirect"). A famous debate resulted in two papers from the opposing factions being produced to try to establish a conclusion.

Direct Methods:

- Block-matching algorithm
- Phase correlation and frequency domain methods
- Pixel recursive algorithms
- MAP/MRF type "Bayesian" estimators
- Optical flow

Evaluation Metrics:

In direct methods several evaluation metrics can be used.

- Mean squared error (MSE)
- Sum of absolute differences (SAD)
- Mean absolute difference (MAD)
- Sum of squared errors (SSE)
- Sum of absolute transformed differences (SATD)

Indirect Methods:

Indirect methods use features, such as Harriscorners, and match corresponding features between frames, usually with a statistical function applied over a local or global area. The purpose of the statistical function is to remove matches that do not correspond to the actual motion. Statistical functions that have been successfully used include RANSAC.

2.3 Block Matching Motion Estimation

Several different algorithms derived from various theories, including object-oriented tracking, exist to perform motion estimation. Among them, one of the most popular algorithms is the Block Matching Motion Estimation (BME) algorithm. BME treats a frame as being composed of many individual sub-frame blocks, known as macro Blocks. Motion vectors are then used to encode the motion of the macro Blocks through frames of video via a frame by frame matching process.

When a frame is brought into the encoder for compression, it is referred to as the current frame. It is the goal of the BME unit to describe the motion of the macro Blocks within the current frame relative to a set of reference frames. The reference frames may be previous or future frames relative to the current frame. Each reference frame is also divided into a set of sub-frame blocks, which are equal to the size of the macro Blocks. These blocks are referred to as reference Blocks. The BME algorithm will scan several candidate reference Blocks within a reference

frame to find the best match to a macro Block. Once the best reference Block is found a motion vector is then calculated to record the spatial displacement of the macro Block relative to the matching reference Block, as shown in Figure.

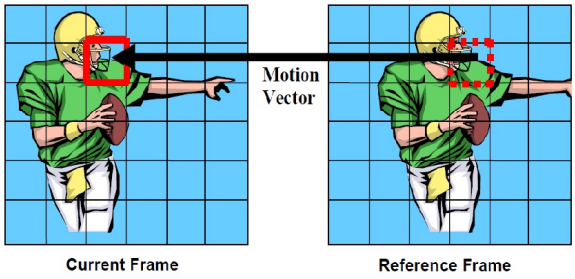


Fig:2.2 matching reference Block

Each subsequent reference Block is offset by either one pixel row or one pixel column from its predecessor while the entire search window area is covered by the overlapping candidate reference Blocks. Note that the original 16x16 macro Block is positioned at the centre of the search window. In order to compare it to every candidate reference Block within the search window, the macro Block has a maximum displacement of 24 pixels to the left, 23 pixels to the right, 16 pixels up, and 16 pixels down from its original position – resulting in a horizontal search range of [-24, +23] and a vertical search range of [-16, +16].

3. ERROR DETECTION/CORRECTION CODES

3.1 Introduction

In information theory and coding theory with applications in computer science and telecommunication, error detection and correction or error control are techniques that enable reliable delivery of digital data over unreliable communication channels. Many communication channels are subject to channel noise, and thus errors may be introduced during transmission from the source to a receiver. Error detection techniques allow detecting such errors, while error correction enables reconstruction of the original data. The general idea for achieving error detection and correction is to add some redundancy (i.e., some extra data) to a message, which receivers can use to check consistency of the delivered message, and to recover data determined to be erroneous. Error-detection and correction schemes can be either systematic or non-systematic: In a systematic scheme, the transmitter sends the original data, and attaches a fixed number of *check bits* (or *parity data*), which are derived from the data bits by some deterministic algorithm. If only error detection is required, a receiver can simply apply the same algorithm to the received data bits and compare its output with the received check bits; if the values do not match, an error has occurred at some point during the transmission.

In a system that uses a non-systematic code, the original message is transformed into an encoded message that has at least as many bits as the original message. Good error control performance requires the scheme to be selected based on the characteristics of the communication channel. Common channel models include memory-less models where errors occur randomly and with a certain probability, and dynamic models where errors occur primarily in bursts. Consequently, error-detecting and correcting codes can be generally distinguished between *random-error-detecting/correcting* and *burst-error-detecting/correcting*. Some codes can also be suitable for a mixture of random errors and burst errors. If the channel capacity cannot be determined, or is highly varying, an error-detection scheme may be combined with a system for retransmissions of erroneous data. This is known as automatic repeat request (ARQ), and is most notably used in the Internet. An alternate approach for error control is hybrid automatic repeat request (HARQ), which is a combination of ARQ and error-correction coding.

3.2 Implementation

Error correction may generally be realized in two different ways:

- *Automatic repeat request (ARQ)* (sometimes also referred to as *backward error correction*): This is an error control technique whereby an error detection scheme is combined with requests for retransmission of erroneous data. Every block of data received is checked using the error detection code used, and if the check fails, retransmission of the data is requested – this may be done repeatedly, until the data can be verified.
- *Forward error correction (FEC)*: The sender encodes the data using an *error-correcting code (ECC)* prior to transmission. The additional information (redundancy) added by the code is used by the receiver to recover the

original data. In general, the reconstructed data is what is deemed the "most likely" original data.

ARQ and FEC may be combined, such that minor errors are corrected without retransmission, and major errors are corrected via a request for retransmission: this is called *hybrid automatic repeat-request (HARQ).*

3.3 Bi-Residue Codes

The biresidue codes separate residue coding using two residue detectors with respect to two suitable moduli. For Residue code a separate code with two residue checks is called bi-residue code. We study [N, $|N|_{A1}$, $|N|_{A2}$]. A_1, A2 are called the check bases. We can choose $A_1=2^c-1$, $A2=2^d-1$, for positive integers c and d.

Theorem:

Consider a biresidue code with k-bit data, that is, N \in 2^k-1, $A_1=2^c-1$,$A_2=2^d-1$ for c!=d. The code can correct any single error in the adder or any error in one of the checker if and only if k=LCM(c, d).The theorem tells us in order to single-bit error correction, the bits of encoder/decoder and two check bases has the least common multiple relations. For example, if the input data bit width is 24, we can choose (c=8, d=6) or (c=12, d=8). If we want the data bit width is 32, we have to choose (c=32, d=2 or 4 or ...). Obviously, this is not a good choose.

3.4 Syndrome and error correction:

The syndrome of a 3-turple (x,y,z) is given by $S(x,y,z)=(s_1,s_2)=(|x-y|_{A1},|x-z|_{A2})$. It is assumed that the three components of the code are processed in separate units, called the adder, check_1, and check_2, respectively, and therefore at most one of the components of the result will be erroneous at any given time. We have the following rules:

- $s_1=0$, $s_2=0$: no error.
- $s_1!=0$, $s_2!=0$: error in the adder.
- $s_1!=0$, $s_2=0$: error in check_1.
- $s_1=0$, $s_2!=0$: error in check_2.
- The following table is a detailed example of syndromes to detect and correct the error for the adder. Here, k=6, c=2, d=3. That is, $A_1=3$, $A_2=7$. Input data is in the range of (0-63).

Error bit – i	0≥1 Error	1≥0Error
0	(1,1)	(2,6)
1	(2,2)	(1,5)
2	(1,4)	(2,3)
3	(2,1)	(1,6)
4	(1,2)	(2,5)
5	(2,4)	(1,3)

4. BUILT-IN SELF TEST

4.1 Introduction to Testing

The introduction of integrated circuits (ICs), commonly referred to as microchips or simply chips, was accompanied by the need to test these devices. Small-scale integration (SSI) devices, with tens of transistors in the early 1960s, and medium-scale integration (MSI) devices, with hundreds of transistors in the late 1960s, were relatively simple to test.

4.2 Importance of Testing:

In the Moore's law the scale of ICs has doubled every 18 months. A simple example of this trend is the progression from SSI to VLSI devices. In the 1980s, the term "VLSI" was used for chips having more than 100,000 transistors and has continued to be used over time to refer to chips with millions and now hundreds of millions of transistors. VLSI devices with many millions of transistors are commonly used in today's computers and electronic appliances. This is a direct result of the steadily decreasing dimensions, referred to as feature size, of the transistors and interconnecting wires from tens of microns to tens of nanometers, with current submicron

technologies based on VLSI Test Principles and Architectures on a feature size of less than 100 nanometers (100 nm).
The reduction in feature size has also resulted in increased operating frequencies and clock speeds; for example, in 1971, the first microprocessor run at a clock frequency of 108 KHz, while current commercially available microprocessors commonly run at several giga hertz. The reduction in feature size increases the probability that a manufacturing defect in the IC will result in a faulty chip.

It is also necessary to test components at various stages during the manufacturing process. For example, in order to produce an electronic system, we must produce ICs, use these ICs to assemble printed circuit boards (PCBs), and then use the PCBs to assemble the system. Testing is used not only to find the fault-free devices, PCBs, and systems but also to improve production yield at the various stages of manufacturing by analyzing the cause of defects when faults are encountered.

5. RESULTS

5.1 Area Results:

The area is estimated based on the number of cells. Considering 16 PEs in an MECA, the area overhead introduced is given by

$$\text{Area Overhead} = \frac{\text{Area BISD} + \text{Area BISC} - \text{AreaTCG}}{\text{Area}_{MECA}}$$

The MECA was selected to act as the CUT to demonstrate the effectiveness of the proposed BISD and built-in self correction (BISC) procedures. The area overhead, timing penalty, and throughput are also utilized to demonstrate the good performance of the proposed BISDC architecture.

Area overhead Estimation

Components	BISD		BISC		MECA	
	TCG	DAS	TCG	SAC	1 PE	16 PE's
AREA	3362	2446	3362	13049	41992	671879

Area Overhead of BISD	0.86%					
Area Overhead of BISC	2.44%					
Area Overhead of BISDC	2.80%					

6. APPLICATIONS

1. Motion Estimation used in MPEG-4 Multimedia Applications and H.264/AVC video coding standards.
2. MECA used in Video Teleconferencing & Video Telephones Digital video codec's.
3. In industries require on-chip, on-board, or in-system self-test to improve the reliability of the entire system, as well as the ability to perform remote diagnosis.

7. FUTURE SCOPE

The input to the MECA is taken in binary format. By Adding the Image to Bit Converter input to MECA is directly in the form of frames, timing required for Motion Estimation will be reduced. The input to the MECA is 8-bit data. It also can be extended to higher volume of data. But the Calculation time required is also high.

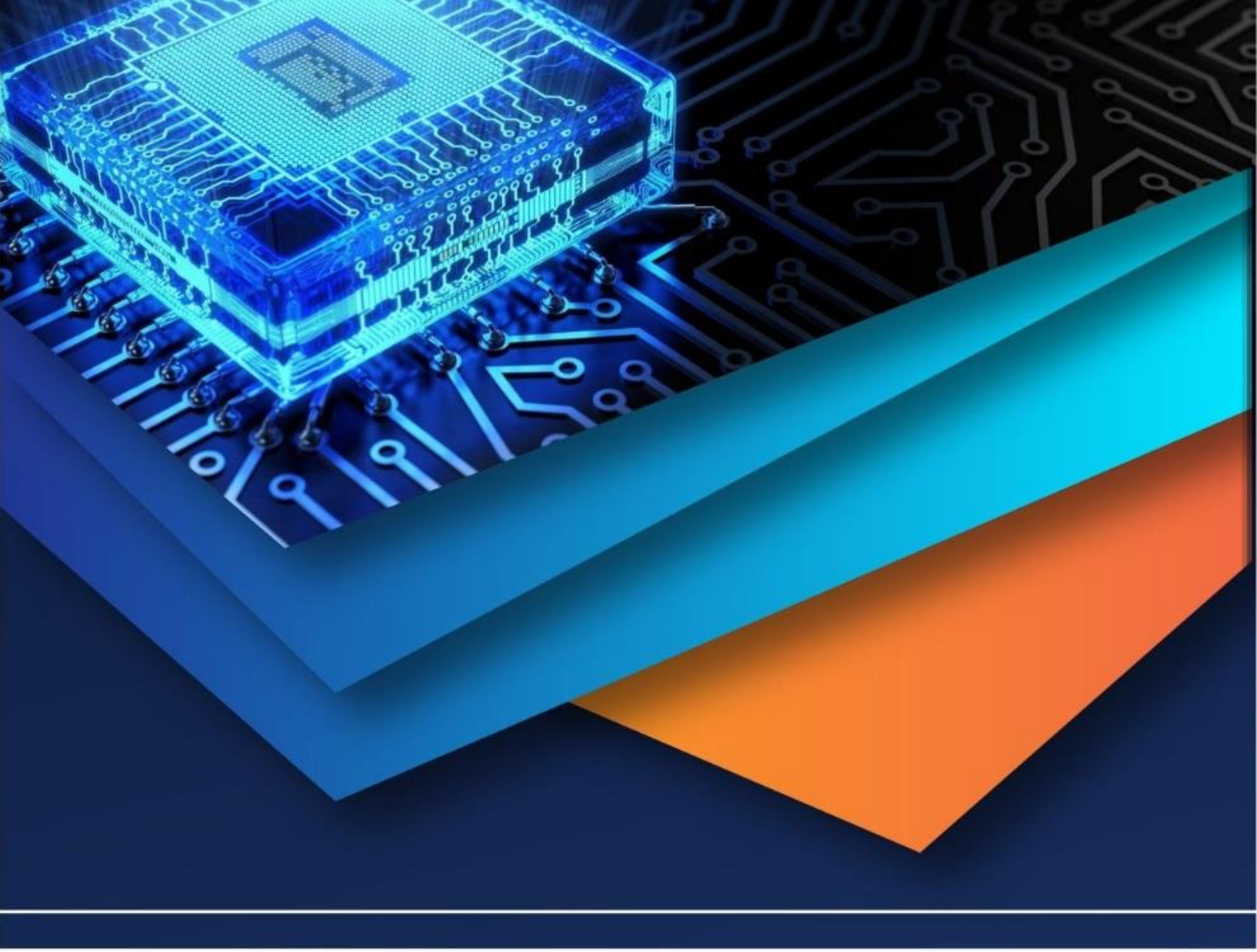
BIBLIOGRAPHY

1. "Built- in self-detection/correction architecture for Motion Estimation Computing Arrays", Chun-lung Hsu, chang-Hsin Cheng, and Yu Liu.VOL.18, No2, February 2010.
2. High Throughput and Efficient VLSI Architecture of Integer Motion Estimation for H.264/AVC, Meihua GU, Ningmei YU, Lei ZHU, Wenhua JIA. (1315-1316).
3. "Biresidue Error- Correcting Codes for Computer Arithmetic", Thammavarapu R.N Rao, Member, IEEE.
4. "VLSI Test Principles and Architectures", Laung-Terng.
5. "Flash memory testing and built-in self-diagnosis with march-like test algorithms," J. C. Yeh, K. L. Cheng, Y. F. Chou, and C. W. Wu.
6. P.Gallagher, V. Chickermane, S. Gregor, and T. S. Pierre, "A building block BIST methodology for SOC designs: A case study," in Proc. Int.Test Conf., Oct. 2001, pp. 111–120.

7. "Design of Programmable Logic-BIST Structures Using Verilog for Digital VLSI Circuits", Ramesh Bhakthavatchalu, Archana K R.

Mr.P.Sunil kumar, Post Graduated in ECE-M.Tech From JNTUH, 2011, and graduated in ECE(B.TECH) From JNTU Hyderabad, 2007. He is working presently as Asst.Professor in Department of ECE in HOLY MARY INSTITUTE OF TECHNOLOGY & SCIENCE (HITS), R.R.Dist, A.P, INDIA.He Is Has 3+ Years Experience, His Research Interests Include mobile ad-hoc networksand mobile communication.

Mrs.P.Swapna, B.TECH In KLCE-Vijayawada, ANU,(ECE), M.Tech(ECE-DE&CS) in SGIET- Markapuram - JNTUK, Her research interests include VLSI and Cloud Enumerating.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY