Architectural Design of Effective Edge Preserving Median Filter with Error Correction System

Vanathi M, A. Kaleel Rahuman

Abstract—Designing of Median filter that can process 36 pixels at a time with edge preservation similar to a filter of size 9. Median sorting is done using Modified minimum exchange sorting method which attracts double the amount of inputs in order to reduce number of comparators used for median filtering. For the same reason i.e. double the amount of inputs switching loss is high in the circuit therefore data driven clock gating (DDCG) is applied for SRAM to form data driven FIFO. Considering space radiation that could excite memory state, Addition of DMR (Double Modular Redundancy) in FPIC would rectify the soft error that could possibly occur due to radiation in space. Therefore proposed method is capable of producing sharp image, controlling switching loss, minimizes area, and reduces soft errors.

Index Terms: FIFO,Median filter, DDCG,DMR

I. INTRODUCTION

Median filter an image processing filter which is best used for removing salt and pepper noise, other impulse noise. Specialty of median filter is that it could preserve edge better than any other image processing filter. Median filter processing involvesa window slides along the image, and the median intensity value of the pixels within the window becomes the output intensity of the pixel being processed. Like low pass filtering, median filtering smoothes the image and is thus useful in reducing noise. Unlike low pass filtering, median filtering can preserve discontinuities in a step function and can smooth a few pixels whose values differ significantly from their surroundings without affecting the other pixels. On comparing a low pass and a median filter, low pass filter removes the impulse noise could posses a value similar to the value of a pixel hence image obtained is not effective. Impulsive values due to noise, the result of using a median filtering will be to reduce noise. If the two low pass and median values are part of the signal, however, using the median filter will distort the signal.

Median filtering operation calculates the middle ranked value by comparing all the data in the window. Considering a window with order O, the median value will be obtained by sorting the pixels in the window in either ascending or descending order and the median value will be in the position O/2. For a window which had odd number of values obtaining the median value is in the position O/2, but an even number of pixels median value is obtained by (O/2 + 1) the position. On a study, median filter efficiency depends on order of matrix. Generally lower order matrixes are preferred so that small portion of image will be processed at a time but this could preserve edges i.e. it could produce a sharp image after removing noises. but having a higher order window would take maximum area in the image for processing on replacing the pixel value with the median value would generate a blunt image denoting high count of similar pixel for a comparably larger window size, because of this quality mostly all median filter implemented for real time FPGA would analyze the image in lower order matrix. In order to synthesis output soon clock latency must increase during the image processing.

Presenting an hardware architecture for 6*6 matrix with error correction system, order of the matrix can be increased to 9*9 and eventually number of Median filter which are embedded within the design and also the inputs to the filter would increase which will draw attention in designing a complex Multiplexer circuit as for the modified minimum sorting network, Multiplexers provide input to support the modification. For a 9*9 filter size, 81pixels will be sorted in a time span of 3*3 filter sizes. In a paper that discussed about Median filter image processing[15][14], the analyzed chart shows that above the filter of size 5*5, image would become blur or processed image would become smoother producing low quality image .image quality increases when the filter size is decreased, and the suggested filter size is 9 i.e. a 3X3 matrix. In order to overcome this constraint, proposed method deal with higher order filter that could process image similar to 3X3 matrix i.e. a lower order. Main idea is to reduce time and quickly produce quality image.

Errors have become a major concern in the design of deep submicron digital integrated circuits (ICs). A soft error occurring due to a high-energy particle hit manifests itself as a transient bit reversal in the logic value of a circuit node. The bit reversal arising can be stored in a memory cell in the circuit and can cause output errors many clock cycles after the soft error occurred. For large process geometry technologies, radiation-induced soft errors are rare and so design for reliability in the event of soft errors was primarily limited to mission critical ICs in space applications. Modern ICs, with smaller process geometries and lower supply voltages are, however, more susceptible to soft errors and reliability is becoming a concern, even in terrestrial

Revised Version Manuscript Received on 30 August, 2019.

Vanathi M, PG student ,VLSI system, PSNA college of engineering ,Dindigul ,Tamilnadu ,India.

Dr A. KhaleelRahuman, Associate professor , ECE department , PSNA college of engineering , Dindigul, Tamilnadu , India .

applications. Naturally, high volume terrestrial applications are more cost sensitive than space applications. Hence, new low-cost techniques to enhance circuit reliability in the event of soft errors are needed. Over the years, a range of techniques has been proposed to protect circuits against soft errors. Triple modular redundancy (TMR) uses three identical copies of the original module operating in parallel. An error is detected if the outputs of the modules differ. The error is corrected by voting, i.e., taking the most common output value as the correct result. In the case of soft errors, this approach is effective since, due to their low rate of occurrence, the probability of two soft errors striking the same module in the same clock cycle is negligible. The difficulty with TMR is that it more than triples the area and cost of the circuit. A well-known alternative is double modular redundancy (DMR) whereby the

Original module is duplicated. This reduces cost and provides error detection but, conventionally, error correction is not possible since voting cannot determine which of the two modules is in error. Recently, the DMR concept has been extended to provide error correction in certain cases by monitoring the

Distribution of the module output errors. For example, using one redundant module, Soft-NMR [4] can correct soft errors if they cause the module- in-error to output a value that is unlikely compared to the typical distribution of outputs. While low cost, the technique does not correct all single errors. In conventional modular redundancy, the redundant modules are identical copies of the original.

1.1. Coventional 6x6 Median Filtering:

The median filter is a nonlinear image processing technique used to remove impulsive noise from images. This spatial filtering operation applies a two-dimensional (2D) window mask to an image region and replaces its original center pixel value with the median value of the pixels contained within the window.

The Window is then moved to the next image region and the cycle is repeated until the entire image is processed. As a result, the ideal filtered image would have no impulsive noise and perfectly sharp edges. These are intrinsic properties of the median filter that cannot be achieved by traditional linear filtering techniques without resorting to time consuming data manipulations.

3	34	56	123	98	32	67	23	6	9
65	78	67	35	23	0	54	48	9	79
4	56	33	6	1	98	87	46	7	4
5	89	99	45	66	90	33	69	37	33
78	23	11	12	7	76	78	67	39	26
24	67	22	4	50	38	91	25	35	67
67	97	98	87	27	50	67	89	48	56
09	123	78	232	29	33	89	73	43	267
70	56	26	255	45	12	93	75	86	234
10	89	87	67	78	19	68	88	44	89

Table I Original Segmented Image

However, the median filter is not an ideal filtering operator, and edge preservation worsens as the total percentage of impulsive noise increases. In sliding window filtering, the 2D window is defined as an N-by-N square kernel (for odd N) whose center is positioned in the currently processed pixel. N being odd ensures a well-defined center to the window. The window size N can be increased in order to remove higher percentages of shot noise; however, this also causes more image blurring. Thus, a typical 3X3 pixels square window has been chosen in this paper. Table II, illustrates the described median substitution process with a 3X3 pixels square window. It should be noted that the analyzed "dead pixel" is removed from the original image. In order to process every image pixel, the 3X3 square window should be moved through the image. However, in FPGAs, this can also be achieved using a nine-pixel stream that sequentially passes through the median filter.

56	56	56	56	56	56	67	23	6	9
56	56	56	56	56	56	54	48	9	79
56	56	56	56	56	56	87	46	7	4
56	56	56	56	56	56	33	69	37	33
56	56	56	56	56	56	78	67	39	26
56	56	56	56	56	56	91	25	35	67
67	97	98	47	98	3	67	89	48	56
09	123	78	232	29	33	89	73	43	267
70	56	26	255	45	12	93	75	86	234
10	89	87	67	78	19	68	88	44	89

Table II Filtered Image For 6X6 Matrix

First sliding window has to be positioned according to the matrix order. Here 6X6 square matrix is chosen shown in Table I, .Original image. Median value is obtained by arranging all the pixel value in the window either in ascending or descending order and choosing the middle value in the order.

Obtained median value: 56

Replacing single median value i.e. 56 for entire 36 pixels Table II, value would reduce the sharpness of the image shown in Table II, filtered image.This drawback is overcome by breaking down this 6X6 matrix into four 3X3 matrix shown in Table III., so that sharpness of the image will be maintained, number of registers to be processed in a cycle will be cut down comparing to [3] usual Median Merge-Sort and the processing speed won't be affected.

1.2 Partitoned 6x6 Median Filtering:

When 3X3 matrix is used, for single clock cycle only 9 pixels are processed. But in proposed architecture for single clock pulse four 3X3 matrixes are processed for a single clock cycle providing 36 pixels can be processed in a single clock cycle.

Breaking 6X6 Matrix to four 3X3 Matrixes:

Original Image		Median sorting	
M1	M2	M1	M2
56	45	34	12
34	56	45	22
123	12	55	24
78	78	56	45
94	97	**M1 =7 8**	**M2=5 6**
89	255	56	78
114	24	94	89
55	89	114	97
45	22	123	52

Table III Median Sorting

Original image		Median sorting	
M3	M4	M3	M4
34	56	34	12
123	12	45	22
78	78	55	24
94	97	56	45
89	255	**M3 =78**	**M4=5 6**
114	24	89	78
55	89	94	89
45	22	114	97
56	45	123	255

Table IV Median Sorting

78	78	78	56	56	56	67	23	6	9
78	78	78	56	56	56	54	48	9	79
78	78	78	56	56	56	87	46	7	4
56	56	56	78	78	78	33	69	37	33
56	56	56	78	78	78	78	67	39	26
56	56	56	78	78	78	91	25	35	67
67	97	98	87	27	50	67	89	48	56
09	123	78	232	29	33	89	73	43	267
70	56	26	255	45	12	93	75	86	234
10	89	87	67	78	19	68	88	44	89

Table V Median Sorting Done For 36 Pixels

Hence by breaking the higher order matrix into N lower order, more number of pixels could be covered. Moreover the number of clock given for single analysis would be efficiently utilized. In [1] this architecture window of order 36.

II. PROPOSED SYSTEM

2. Clockgating Implementation:

2.1. Datadriven Clock Gating In Fifo:

In the proposed system 6X6 median filter is designed for the purpose of decreasing the image processing time and also to increase the quality of image by preserving edges in image. Agreeing to the fact that a conventional sorting for a filter size of 36 would provide a blur image and filter of size 9 would provide a sharp image 6X6 median filter is designed with edge preservation, since the novel sorting method increases the size of filter.

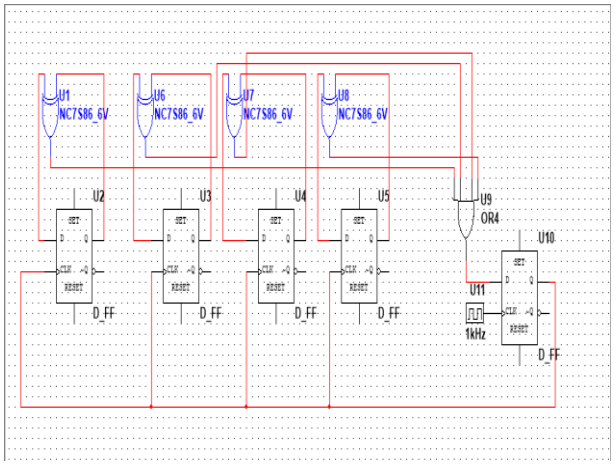

Fig 1 Gated FIFO

Area conservation is done by removing 6 comparator with 2 MUX and 2 FIFO. Inputs are doubled as per the table [3] for the purpose of comparator operation. Problem is that inputs have to be doubled and also there is repetition of input value especially the redundant 8'00000000 and 8'11111111. These values when called again and again would bring switching loss in FIFO. Thus to reduce switching loss FIFOs are clock gated

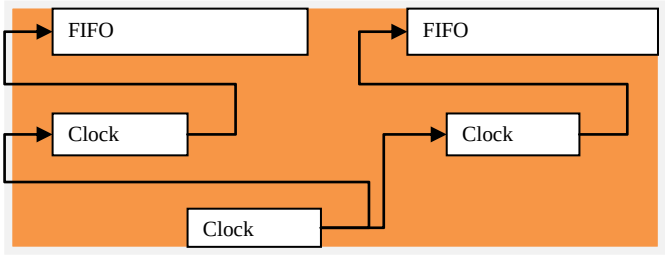

Fig 2 Division Of Global Clock For LSB And MSB

Data driven clock gating [3] is done in an SRAM to stop data overwriting on the same memory. Since the existing system has redundant inputs data driven clock gating is done over FIFO. Table I shows how clock gating is done to reduce switching loss. Inputs are provided through bus hence parallel in parallel out strategy is followed for FIFO. Actually size of SRAM is 8bit wide for the purpose of demonstration 4bits are considered moreover clock gating is done separately for LSB and MSB. Since LSB tend to be same for almost all the bits global clock is separately provided for both 4bits shown in Table 1.2.

2.2. Overall Architecture for 6x6 Effective Edge Preserving Median Filter:

The proposed architecture shown in Fig 3 has been divided into 4 parts first part has a MUX circuit , Second part 4 FIFO which has comparator units (CU) connected to registers and clocks , Third part has SISO(n) storage unit and Fourth part SISO(n)a which provides the initial Median value i.e. output .

Providing clock pulse is a separate part where clock pulses should be given to enable MUX circuit, FIFO circuit, SISO1 unit and finally SISO2 unit. Clock pulses to enable MUX circuit is E1, FIFO circuits is E2, SISO (n) is E3 and SISO (n) a is E

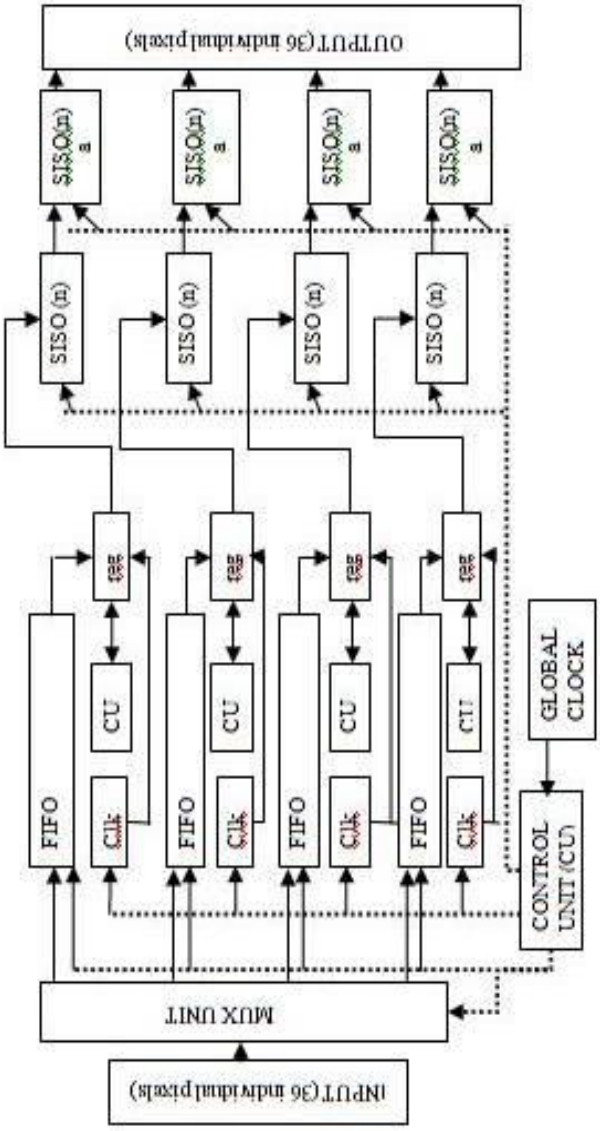

Fig 3 Overall Architecture Of Proposed Model

Enabling Signals:

- MUX => E1;
- FIFO => E2;
- SISO1 => E3;
- SISO1a => E4;

For the first enable signal MUX circuit is enabled, MUX unit has 8 [9:1] MUX, when input pixel values are provided to the MUX say 36 pixels are divided into set of 9 inputs per MUX and these 9 pixels are further divided into set of 6 and 3 with addition of 0s and 1s (detailed below) per MUX which provides 8 outputs. There are 4 select signals given to all 8 MUX to provide synchronizing output in all 4 FIFO.

For the second enable signal FIFO unit is enabled, output of the MUX is provided to the FIFO of 9* 8-bit register. All FIFO units have a comparator unit (CU) which does the Median sorting by Compare and Swap method [4]. When the sorting is completed the output bus is taken from the middle register.

When all 4 Median values are obtained and is available ready in the register, then third enable signal is activated to transfer all the sorted Median value to the SISO (n). All Median value has to be stored in the SISO (n) at same time which would realize analysis of 36 pixels and the value of n will range from 1 to 4 depending of number of partition in 36 pixels units if it is a 9*9 window then number of pixels will be 81, number of partition in 81 pixels i.e. there will be 6 Median output.

For the fourth enable signal, values that are present in SISO (n) is given to next 8-Bit register SISO (n) a. Output of SISO(n)a provides the 36 Median pixels. Once the analysis is over for first 36 pixels, window is slides for the next 36 pixels in the segmented image. There can be addition of say SISO (n) basis (n) c...etc depending on the necessities.

2.3 Design Of Median Filter:

Median filtering system design is provided in Fig 7 this part has a FIFO for accepting inputs which is given out of the MUX, 9 8-BitRegisters representing 9pixels, a comparator unit (CU) which take care of compare and swap operation for obtaining median value and All registers are connected to LACG [4] system shown in Fig 8 for gated clock input.

Comparing to the existing system [5] which has a separate power module for conserving power due to switching transition. During compare and swap operation redundancy clock pulses at which enabling signals are assigned depends on the operation , Since MUX circuit has 4 select line it takes 9 rising edge pulses , FIFO does the sorting operation which takes 8 rising edge pulse to obtain Median value , there can be delay while sorting due to the presence of jitter or skews . SISO1 takes one rising edge and SISO2 take another one rising edge. Thus making the circuit performs work within 19 rising edge pulses.

In Switching transition occurs when clock pulse has to be given to registers that doesn't need to be swapped or when compare and swap operation is completed for that register and the value stored in the register is at haul. Having just data driven clock gating method[6] which is implemented in existing system will increase the complexity of operation i.e. a signal will be transferred from the comparator unit (CU) if there is redundancy in switching transition for particular signal to power save module . Power save module will gate the clock pulse for this particular register. Comparator unit does the Compare and Swap operation, which is discussed in 1-D Median Filter Design [7], that is an effective Median sorting with just 8 clock cycles. FIFO is activated with enabling signal T1.

2.3.1 Design OfMUX UNIT:

Analysis for 36 pixels input need to be divided into 4 separate 9 pixels unit, by dividing into smaller order matrix would preserve the sharpness of the image more than sorting more than sorting a higher order matrix. Since we have to give 36 pixels as input for analysis, it will be divided into 9 separate pixels as input. Here we are designing MUX unit capable of processing 9 pixels for the proposed architecture [2].

INPUT ASSIGNMENT TABLE FOR MUX B [9:1]										
Sel[4:0]		**B0**	**B1**	**B2**	**B3**	**B4**	**B5**	**B6**	**B7**	**B8**
0000		0								
0001			0							
0010				q2						
0011					0					
0100						0				
0101							q5			
0110								0		
0111									0	
1000										q8

Table VI Input Assignment For MUX A

Median sorting adopted for this architecture follows sequential compare and swap technique i.e. the method at which sorting is done doesn't have any proper algorithm as Median filters doesn't behave to signal processing principles and it is a non leaner filter. Median analysis need not have to follow any linear methods for Median computation. Therefore the inputs are further divided into 6 and 3 pixels which sum up 9 input pixels.

Here MUX unit is designed for 9 input pixels, 2 (9:1) MUXs are used. As we have divided the inputs for 2 (9:1) MUX say

- Input to the MUX A :

A0, A1.A2, A3, A4, A5, A6, A7, A8

- Inputs to the MUX B:

B0, B1, B2, B3, B4, B5, B6, B7, B8

- Considering the inputs of the 9 input pixels :

Z0, Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8

- The input pixels value to the MUX A are assigned as:

A0 <= Z0; A1<= Z1; A2<= 1;
A3<= Z3; A4<= Z4; A5<= 1;
A6<= Z6; A7<= Z7; A8<=1;

INPUT ASSIGNMENT TABLE FOR MUX A [9:1]										
Sel[4]		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0000		p0								
0001			p1							
0010				1						
0011					p3					
0100						p4				
0101							1			
0110								p6		
0111									p7	
1000										1

Table VII Input Assignment For MUX B

- And the values to the MUX B is assigned as:

B0<= 0; B1<= 0; B2<= Z2;
B3<= 0; B4<= 0; B5<= Z5;
B6<= 0; B7<= 0; B8<= Z8;

- Output of MUX A:

pi…pi+8(p0,p1,p2,p3,p4,p5,p6,p7,p8)

- output of MUX B:

qi….qi+8 (q0,q1,q2,q3,q4,q5,q6,q7,q8)

Since only 9 pixels are fixed irrespective to size of window A2,A5and A8 are pre assigned with value 11111111 bits because during the shift operation in FIFO this 8bit is ANDed with values of output of MUX B . B0, B1, B3, B4, B6 and B7 are pre assigned with the 00000000 bits so that during FIFO shift operation this bit is NANDed with input of MUX A. Table VI and Table VII the type of output obtained from the MUX. Since there are 9 pixels the number of select line is 4. Select line sel [4:0] starts to provide output when select line Combination is 0000 till 1000, after reaching 1000 select line should start again from 0000.

This counting operation can be done using counter circuit which is not discussed in this architecture. Counter circuit design could be extended for this project as future work. Thus inputs for FIFOs are assigned through MUX units

2.3.2 Median Filter Architecture For 9 Pixels:

Since we have designed filter on the basis of sorting technique adapted from [5] which is has reduced switching transition. Comparator compares three inputs i.e. present input and past inputs. This architecture has 2 FIFO units to store inputs, a total of 13 comparators are used here comparators are mentioned as Comparator units (CU) as this controls the entire process, 18 latches are used to obtain the output from comparator 2 and 3, 9 SISOs are used to store values which would be taken out for computation i.e. for sorting method. These SISOs and latches are altogether addressed as Storing Unit. In the existing model from [8] has 19 (CU) for compare swap operation, as this method does compare swap operation sequentially and the number of registers used to store values obtained from comparator unit (CU) are more i.e. existing model used 15 registers , but in our proposed architecture we use 9 registers and 18 latches[5].

From the MUX unit 2 buses (BUS p and q) provide input to Median filter. Outputs from MUX A are pi and MUX B is qi. Architecture is based on how MUX are providing inputs.

During ENABLE SIGNAL E1and E2:

1. MUX undergoes first iteration (i=0)

When the select line of MUX combination is **'0000'**, p0 and q0 is fed to the circuit. From Table 1 and Table 2,

- **Value of p0 = Z0 and q0 = 0 (8'000000000).**

p0 is providing to **FIFO 1** and (q0, p0) is provided to **AND** gate. Output of **AND** gate will be 0 which is given to comparator 2 i.e. **CU2** which is stored in a latch.

2. MUX undergoes second iteration (i=1)

Select line combination is **'0001'**, Now p1 and q1 is fed to the circuit, the past value p0 is shifted to **FIFO 2** and the present value is given to **FIFO 1**.

- **Value of p1 = Z1 and q1 = 0 (8'00000000**).

p1 is fed to **FIFO 1** and (p1, q1) is **AND** gated. Output of **AND** gate will be 0 which is given to comparator 2 i.e. **CU 2** which is stored in a latch. Previous value in the latch will be shifted to next latch unit.

3. MUX undergoes third iteration (i=2)

Select line combination is **'0010'**, Now p2 and q2 is fed to the circuit, the past value p1 is shifted to **FIFO 2**, the present value p2 is given to **FIFO 1** and p0 is shifted out of **FIFO 1**.

➢ **Value of p2 = Z2 and q2 = 1 (8'11111111).**

p2 is fed to **FIFO 1** and (p2, q2) is **AND** gated. Output of **AND** gate will be q2 which is given to (**CU 2)** which is stored in a latch. Input to the (**CU 1**) is p1 and p0, where p0 is readily available when p2 is provided. At third iteration (i=2), (p0, p1, q2) is available for three value comparison.

Once again output of comparator 1 i.e. **h1** and **l1** is further fed into comparator 3 i.e. **CU3** for further comparison, and the output of **CU3** be h3 and l3. Now this value is fed to latch. Previous values i.e. when i=2, but when (i=2) i.e., the present iteration the values at the latches are moved to SISO (a, b, c) registers unit. Storing unit is designed in such a way that there are set of 6 latches in which 3 latches connected to comparator unit pipelined to another 3 latches. These latches are then pipelined to 3 SISO registers, these SISO registers are given enabling signals when (i=8) all the inputs (p0….p9 and q0….q9) are processed. These SISO has 2 output lines, one to transfer output to the latches and one to comparator unit (**CU**). When enable signal is provide to these SISO the values present in the SISO are taken out for sorting. By this way at third iteration desired output for sorting can be shifted out SISOs for sorting

*Output of : SISO a= a8,*SISO b= a7,SISO c= a6,SISO d= a5, SISO e= a4,SISO f= a3,SISO g= a2,SISO h= a1,SISO I = a0.

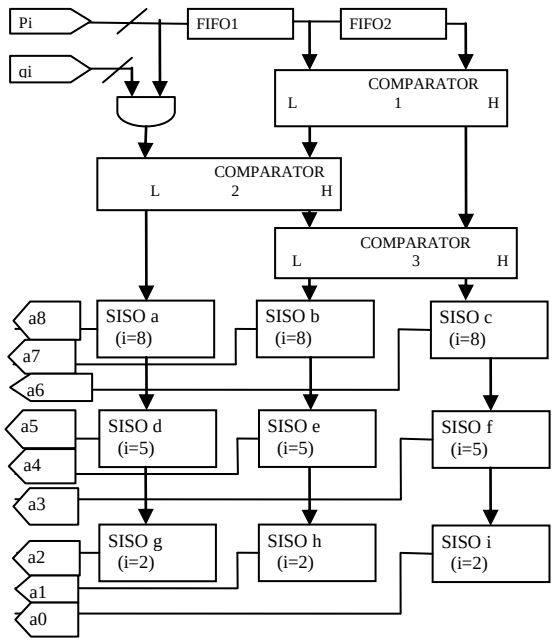

Fig 4 Design Of Median Filter

These Outputs of SISO are provided to the sorting unit shown in Fig 4

2.4Algorithm For Modified Minimum Exchange Sorting:

inputs=> pi ,qi,hj,lj;
Output=>ai, zi, y;
Stages i=0 to 9;
j= 1to 15;
Start
{
When
i: pi,qi
fifo1=> pi;
and => qi;

When
i+1: pi+1,qi+1;

fifo 1=> pi;
fifo 2 =>pi-1;
and=>qi+1;

when
i+2: pi+2,qi+2;

fifo 1=> pi-2;
fifo 2=> pi-1;
and=>qi+2;
}
end

Comparison operation:
if (pi >= pi-1)
{
hj=pi and lj=pi-1;
{
y= qi and pi;
if(y>lj)
hj+1=y and lj+1=lj;
if(hj+1>hj)
hj+2=ai;
lj+2=ai;
else(hj+1<hj)
hj+2=ai;
lj+2=ai;
else
hj+1=lj and lj+!=qi;
if(hj+1>hj)
hj+2=ai;
lj+2=ai;
else(hj+1<hj)
hj+2=ai;
lj+2=ai;
end if
end if
}
else
hj=pi-1 and lj=pi;
{
y= qi and pi;
if(y>lj)
hj+1=y and lj+1=lj;
if(hj+1>hj)
hj+2=ai;
lj+2=ai;

```
else(hj+1<hj)
hj+2=ai;
lj+2=ai;
else
hj+1=lj and lj+!=qi;
if(hj+1>hj)
hj+2=ai;
lj+2=ai;
else(hj+1<hj)
hj+2=ai;
lj+2=ai;
end if
end if
}
end if
}
//obtaining values stored in SISOs for sorting operation
if(i=2)
hj=a0
hj+2=a1
lj+2=a2
if (i=5)
hj=a3
hj+2=a4
lj+2=a5
if  (i=8)
hj=a6
hj+2=a7
lj+2=a8
else if
end if
```

At the output of SISOs clock gating is done, just to reduce the switching frequency due to repetitive registration of same Median value in the register. Hence at an event of recitative similar Median value synthesis from Median sorting network clocks are gated for that particular register. in this architecture clock gating can also be provided to Storing unit mentioned above , which uses clocks to obtain output after median sorting , therefore minimum switching transition can be obtained with this architecture. Concerning outputs, it will be 4 median values where each median values is duplicated into 9 values. These nine values are the replaceable pixels. Hence the output provides 36 similar values which will be replacing the original median value present in the window .Sorting network and its connection are shown in Fig.8.to know in which way this sorting is networked refer to [9].

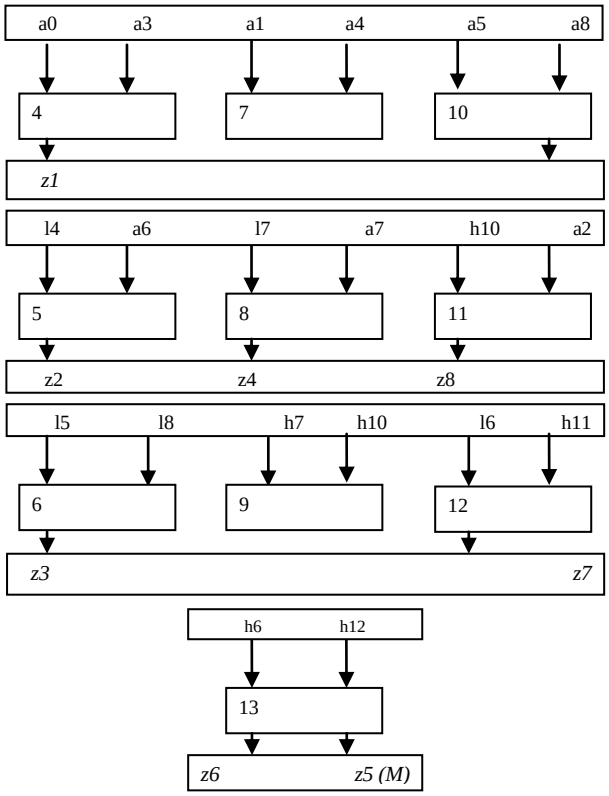

Fig 5 Sequential Sorting For 9 Pixels

2.5 Error Correction System:

A single event upset (SEU) is a change of state caused by one single ionizing particle (ions, electrons, photons) striking a sensitive node in a micro-electronic device, such as in a microprocessor, semiconductor memory, or power transistors. The state change is a result of the free charge created by ionization in or close to an important node of a logic element (e.g. memory "bit"). The error in device output or operation caused as a result of the strike is called an SEU or a soft error [13]. The SEU itself is not considered permanently damaging to the transistor's or circuits' functionality unlike the case of single event latch up (SEL), single event gate rupture (SEGR), or single event burnout (SEB). These are all examples of a general class of radiation effects in electronic devices called single unit upset (SUU). In digital and analog circuits, a single event may cause one or more voltages pulses (i.e. glitches) to propagate through the circuit, in which case it is referred to as a single-event transients (SET). Since the propagating pulse is not technically a change of "state" as in a memory SEU, one should differentiate between SET and SEU [11]. If a SET propagates through digital circuitry and results in an incorrect value being latched in a sequential logic unit, it is then considered an SEU.

2.5.1 DMR Method:

Due to the continuous scaling of digital systems and the increased demand on low power devices, design of effective soft error tolerance techniques is of high importance to cope with the increased susceptibility of systems to soft errors and

to enhance system reliability. In this work, we propose a double modular redundancy (DMR) technique that aims to achieve high reliability with reduced area overhead. Furthermore, we propose an improved application of DMR based on the use of C-element (DMR-CEL). The proposed technique is compared with Triple Modular Redundancy (TMR) technique and DMR-CEL. Simulations performed for LGSynth'91 benchmark circuits demonstrate that applying[11] the proposed DMR technique achieves improved reliability with significantly lower area overhead than TMR without voter protection. Furthermore, improved reliability with lower area overhead is achieved by the proposed DMR technique in comparison to DMR-CEL without C-element protection.

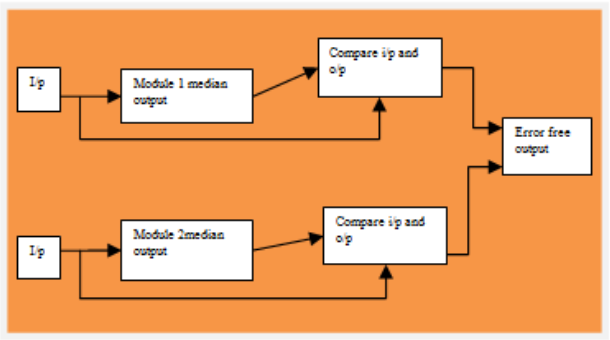

Fig 6 Dual Modular Redundancy For 9 Pixels

In the proposed work, all the input pixels are compared with output median value if the output is different from the nine pixels then it is an error in the circuit. Median value is EXNORed with input pixels , if input and output are same then output will be '1' ,if not '0' . All the median values are compared with the input median value if only the median value matches with input gray scale values output will be '1'. After comparison all the EXNORed values are ANDed, if the output is same as input AND gate will be '1', that will be the signal for one pixel value comparison. Comparison of all 9 pixel values is found finally every value is ORed; this one is given as enable signal to a flip-flop. Once the flip-flop is enabled the median value stored in memory will be copies to this stacked flip-flop. This strategy is followed for both modules, once the values are stored in flip-flop [10] output of the flip-flops are ORed to get the error free median value

.

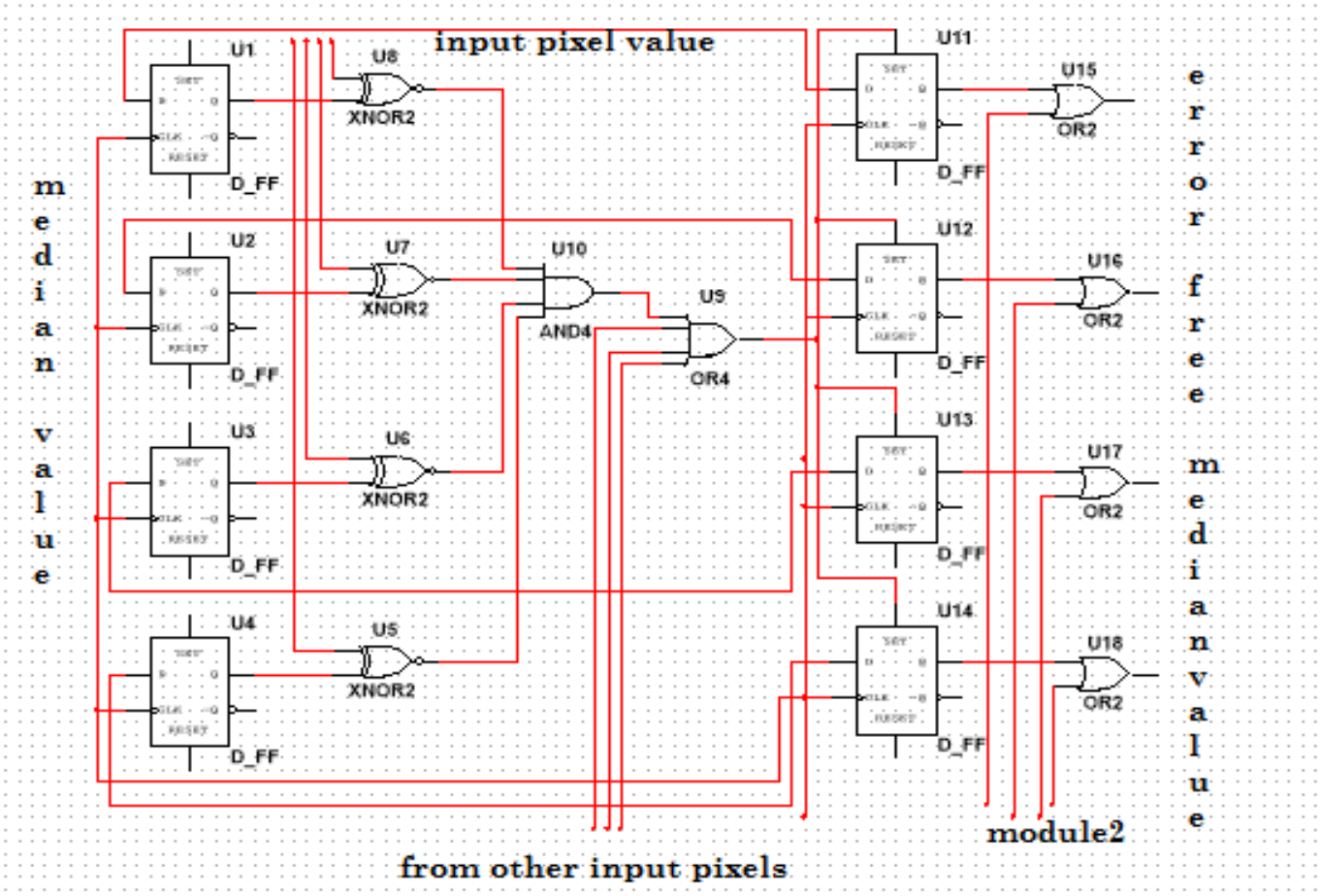

Fig 7 Circuits For DMR

III. RESULTS

3.1 Experiments And Analysis

System design and simulation are done using Xilinx software. From the table [2] shows that power consumption for filter size 9 is 4.43(mW) and for filter size of 36 is 6.53 (mW). From the analysis done in [7] comparing the delay of [1] and [2], delay through low power FIFO is less but number of register is high for filter size of 9 i.e. this method uses 18 registers and 9 control units to sort. Though the power consumption is less, number of registers, comparators and power modules are more. Thus collaborating [1] and [2] with modification in Minimum exchange sorting number of comparators are reduced and also registers. In [7] uses 19 comparators and 15 registers. Proposed system uses only 13 comparators with same number of registers used in [7]. This makes the power consumption reduced comparing to [6] and speed of operation is also increased. Presence of Gated FIFO further reduces redundancy in switching. DMR circuit checks for the odd pixels by comparing the present median value with the input pixels, if the Module 2 output is different from the input the median value is considered 0 but the other value is ORed with Module1 output and the errorless output value will be generated.

3.2 Simulation Result:

3.2.1 DESIGN OF MUX UNIT

Sel: 0000

Inputs selected: p0=00001111, q0=00000000

Output: x=00001111, y=00000000

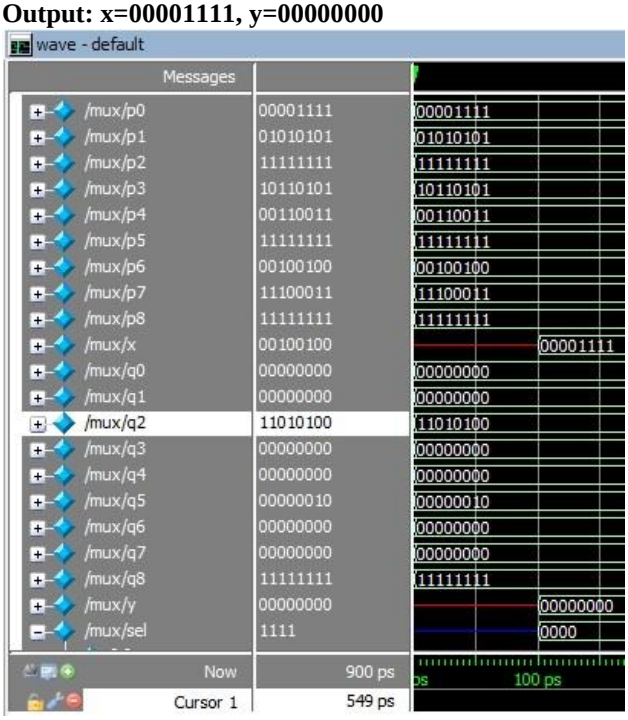

Fig 8 Simulated Output For MUX Unit

3.2.2 Median Sorting Simulation:

Through modified minimum exchange sorting network, Input pixels are sorted. Mean value which is the middle value in the sequence is chosen as Median value. Median value for 36 pixels is obtained in registers given below.

Z5:71 ; Za5:97 ; Zb5:73 ; Zc5:70

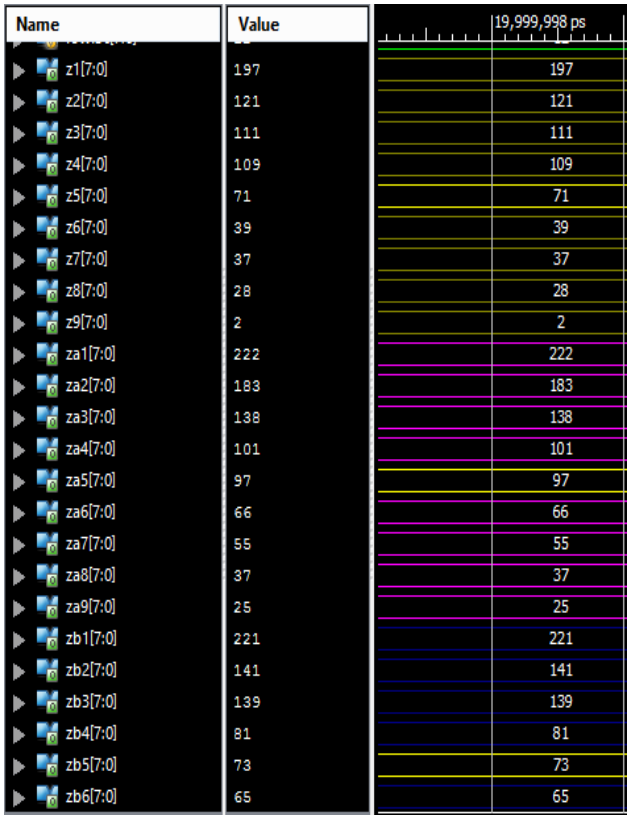

Fig 9 Simulated Output For Median Sorting

3.2.3 Simulation For Median Value Isolation From Sequence:

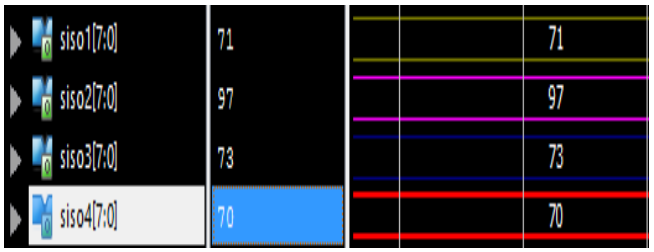

Fig 10 Median Value Isolated Through Shift Registers

SISO1:71;SISO2:97;SISO3:73;SISO4:70

3.2.4 Clockgated FIFO:

Input : 11110000
Output : 10101000
Clk: 0 clkg (clock gating for LSB): X (not yet initialized)
clkg1 (clcokgating for MSB): X (not yet initialized)
Rst: 1

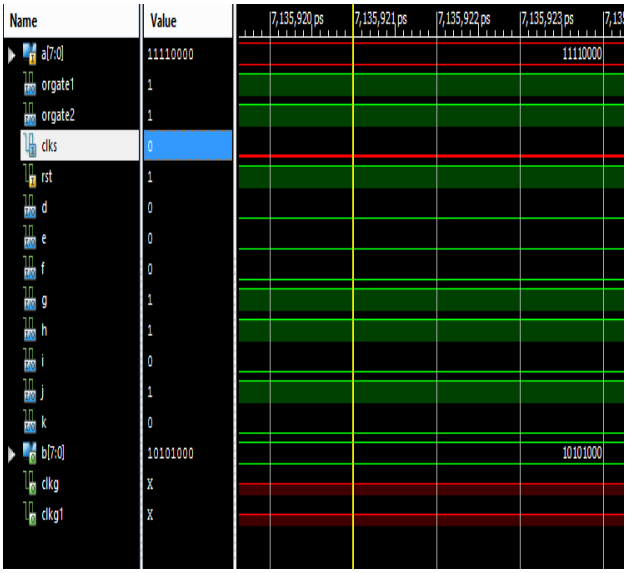

Fig 11 Initializing Clock Gating Circuit

Input to the FIFO is varying from the past input, since the clks (global clock) is zero clock gating is yet to be initialized.

3.2.5 Clock NOT GATED :

Input :11110000
Output:11110000
Clks(global clock) :1 clkg:1 clkg1:1
Rst:1

As the global clock pulse changes from 0 => 1 , clock gating is done both the clkg and clkg1 is set to 1 which by means provides input pulse to FIFO to change value inside the FIFO. Switching is done in the register .

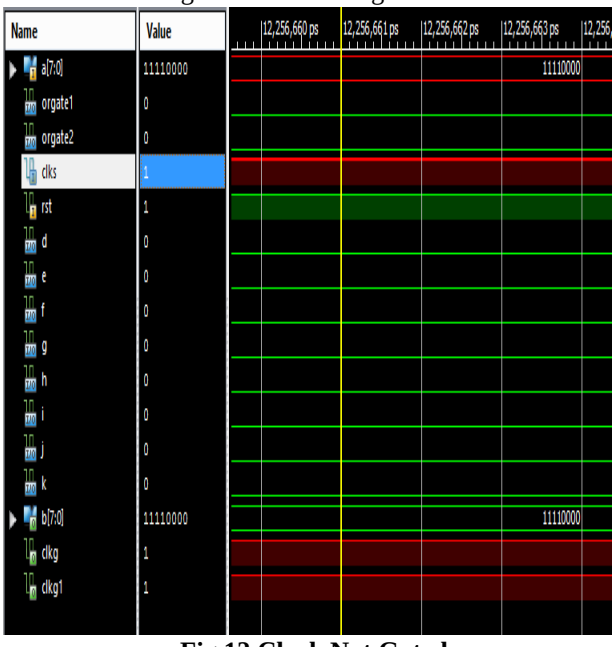

Fig 12 Clock Not Gated

3.2.6 CLOCK GATED :

Input: 11110000
Output: 11110000
Clks: 1 clkg: 0 clkg1:0
Rst: 1

Fig 13 Clock Gated

Here both the input and output are same, hence clkg and clkg1 are not 1 so, switching pulse is not given making it "gated clock". As both the input and the value inside the FIFO are same no switching is done.

3.2.7 DMR CIRCUIT:

P0, p1, p2 are the input pixel values of Module1 and p01, p11, p21 are the input pixels values of Module 2, median and median1 are the sorted median values for Module 1 and Module 2.

Module1: (P0, p1, p2): (11, 10, And 33)
Module2: (p01, p11, p21): (11, 10, And 33)
Median: 33
Median: 63
Clk: 1, rst (Module1):1, rst1 (Module2):1
Enable (Module1):1 Enable1 (Module2):0

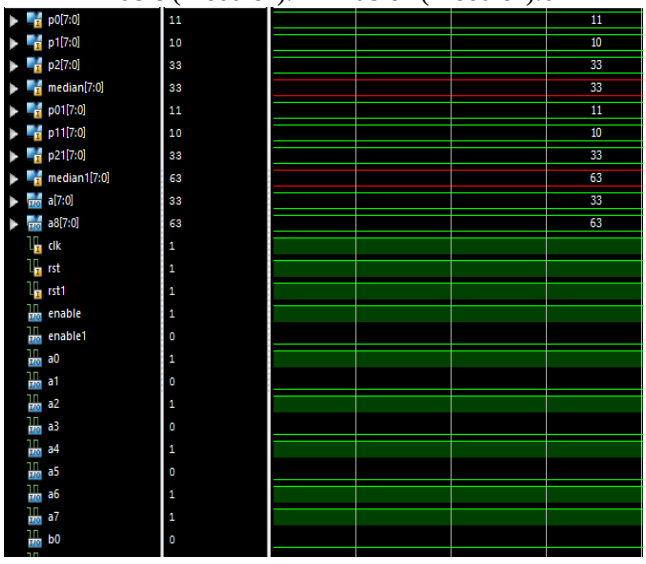

Fig 14 Initializing Inputs For Both Modules

Median value 63 isn't present in the input stream hence it can be an outcome of single event upset (SEU) which need to be rectified.

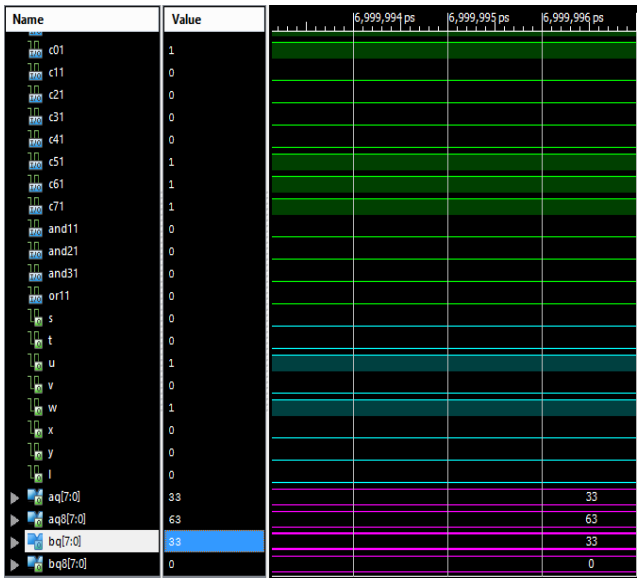

Fig 15 Error Rectified Output

Output : (s,t,u,v,w,x,y,l) : (00101000) => 33 , it is the correct median value . Error value 63 had been removed. This median value (33) is taken for pixel replacement

IV.CONCLUSION

Proposed architecture is capable of performing operation of 4 Median filter of size 9. Median filters tend to lose their edge preserving capability above filter size 25, in this architecture this drawback have been overcome. Median filter is efficient with filter size of 9, but the processing time will be high to obtain median value. Conventional filter size of 36 will provide blur image, this drawback is overcome by proposed architecture. Making the processing time less and also efficient. Gated FIFO reduced the switching loss. Presence of DMR circuit act as fault tolerant circuit to overcome single event upset (SEU).

4.1 Future Work:

- Image processing IC that could perform various image processing techniques could be built with reduced architecture .
- Physical implementation of this architecture should be done with EDA (Electronic Design Automation) for chip design.
- Further physical characteristics of standard cells and macros can be designed for proper design .

REFERENCE

1. Amarjit Roy, JoyeetaSingha, LalitManam, Rabul Hussain Laskar. "Combination of adaptive vector median filter and weighted mean filter for removal of high density impulse noise from colour images", *IET journals , IET Image Processing*, ISSN 1751-9659, 24th April 201
2. E. Nikahd, P. Behnam and R. Sameni, "High-Speed Hardware Implementation of Fixed and Runtime Variable Window Length 1-D Median Filters," in *IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 63, no. 5, pp. 478–482,May 2016.*
3. Error Detection Technique for a Median Filter Luis Alberto Aranda, Pedro Reviriego, Juan Antonio Maestro, IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. X, NO. X, XXXX 201X 1
4. G. Mikolajczak and J. Peksinski, "Estimation of the variance of noise in digital images using a median filter," 2016 39th International Conference on Telecommunications and Signal Processing (TSP), Vienna, 2016, pp.489–492
5. K. Benkrid, D. Crookes, and A. Benkrid, "Design and implementation of a novel algorithm for general purpose median filtering on FPGAs," IEEE International Symposium on Circuits and Systems (ISCAS), pp IV-425-IV-428 vol.4, 2002 D.
6. L. A. Aranda, P. Reviriego, and J. A. Maestro, "A Fault-Tolerant Implementation of the Median Filter", presented at the Radiation Effects on Components and Systems (RADECS) Conference, Bremen, Germany, N 2016.
7. Prokin and M. Prok in, "Low hardware complexity pipelined rankfilter," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 6,pp. 446–450, Jun. 2010.
8. Q. Yang et al., "Fusion of Median and Bilateral Filtering for Range Image Upsampling," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4841-4852, Dec. 2013.
9. ShmuelWimer, *Member, IEEE*, and AryeAlbahari, A Look-Ahead Clock Gating Based on Auto-Gated Flip-Flops.
10. ShmuelWimer, *Member, IEEE*, and Israel Koren, *Fellow, IEEE*, Design Flow for Flip-Flop Grouping in Data-Driven Clock Gating, IEEE Transactions On Very Large Scale Integration (Vlsi) Systems, Vol. 22, No. 4, April 2014.
11. Shih-Hsiang Lin, Pei-Yin Chen, *Member, IEEE*, and Chang-Hsing Lin, Hardware Design of an Energy-Efficient High-Throughput Median Filter, IEEE Transactions On Circuits And Systems —Ii: Express Briefs
12. T. S. Le, N. T. Do and K. Hamamoto, "Speed up temporal median filter and its application in background estimation,"*2016 IEEE InternationalConference on Computing & Communication Technologies, Research,Innovation, and Vision for the Future*, Hanoi, 2016, pp. 175–180.
13. V. G. Moshnyaga and K. Hashimoto, "An efficient implementation of 1-D median filter," in *Proc. IEEE Int. MWSCAS, 2009, pp. 451–454*
14. Voter Insertion Algorithms for FPGA Designs Using Triple Modular Redundancy Jonathan Johnson and Michael WirthlinNSF Center for High-Performance Reconfigurable Computing (CHREC) _Dept. Of Electrical and Computer EngineeringBrigham Young UniversityProvo, UT 84606, US
15. ZdenekVasicek, Lukas Sekanina," Novel Hardware Implementation of Adaptive Median Filters", IEEE Transactions, 2008.

AUTHORS PROFILE

Dr.A.KaleelRahuman has obtained his B.E degree in Electronics and Communication Engineering from Madurai Kamaraj University Madurai, in the year 2002 and completed his M.E., degree in VLSI Design from Anna University Chennai, in the year 2005. He has completed his Ph.D., degree in Information and Communication Engineering from Anna University Chennai, in the year 2015. He has published 21 papers in both National Conferences and International Conference. He has also published 20 papers in International Journals. His areas of interest include VLSI Design, Digital Signal Processing and Computer networks,. He has 17 years of teaching experience. Currently he is working as a Associate Professor in the Department of Electronics and Communication Engineering at PSNA College of Engineering and technology, Dindigul, Tamil Nadu, India.

Email: kaleelvlsi@psnacet.edu.in

Vanathi M, completed B.Tech in Electrical and Electronics Engineering in Amrita Vishwavidhyapeethamuniversity ,Coimbatore. (2012-2016). Completed M.Tech in VLSI systems in PSNA college of engineering, Dindigul (2017-2019).Published a paper on image processing in an internatitonal journal. Field of interest includes Digital systems, Physical design ,STA, Physical verification and RTL design . Presently working as an intern in Sion Semiconductors pvt.,ltd., Banglore,India.

Email:aaparivanathi@gmail.com.