Design of Power Efficient One Dimensional Median Filter for Real Time Noise Reduction Applications

[1]M. D. Manigandan ,[2] S. Deepa, [3] B. Sathyabhama
[1]PG student,[2]Professor, [3]Assistant Professor,Department of Electronics and Communication Engineering
Panimalar Engineering College,Chennai ,India

***Abstract*—Tremendous growth of multimedia technologies mandates the need for high quality images. Thus fast and efficient noise removal technique is the need of the hour. Nonlinear filters have shown their supremacy in removing the outliers from a signal affected by non-Gaussian noise, such as clicks, scratches, salt-and pepper impulses etc., Median filter is the most sought after non linear filter known for its excellent noise reduction capability. A modified selective one dimensional median filter design is proposed in this work which is aimed at reducing the power consumption. Proposed design is a word-level filter; the samples are stored in descending order in the window of size N. When a sample enters the window, the oldest sample is removed, and the new sample is inserted in an appropriate position to preserve the sorting of samples. The throughput of the design is increased by performing the deletion and insertion of samples in the same, so that the median output is generated at each cycle. The simulation results has shown reduction in power and delay thereby improving the throughput.**

Keywords: Median filter, Rank generation, Rank calculation, sorting, Adaptive Median filter

I. INTRODUCTION

The performance of imaging sensors is affected by a variety of factors, such as environmental conditions during image acquisition and by the quality of sensing elements themselves. For example sensor temperature, light levels are major factors affect the amount of noise in the resulting image. Images are corrupted during transmission principally due to interference in the channel used for transmission. With the advent of internet technology and the stupendous growth of multimedia technologies, the design of power efficient real time image and video processing hardware has become a great challenge to designers and researchers.

Median filter belongs to the class of edge preserving smoothing filters which are non-linear filters. These filters smooth the data while keeping the small sharp details. Median filtering is a simple and very effective noise removal filtering process. Its performance is particularly good.

Median filter is a non-linear digital filter (order statistics filter) widely used in signal and image processing for smoothing of signals, suppression of impulse noise and edge preservation. In median filtering a window of size $N=2M+1$ is sliced along entire signal sequence and the centre value of each window is replaced by the median of the values in the window. For example, in 1-Dimensional median filtering the window $Wi = \{xi-M, \ldots xi, \ldots xi+M\}$ is centered on the i^{th} input value; the filter output is y_i=median(Wi). The main problem of the median filter is its high computational cost: the temporal complexity of sorting N values even with the most efficient sorting algorithms is $O(N \cdot log\ N)$. Therefore, the real-time median filters are traditionally implemented in hardware. Designing power efficient hardware has always been in focus of research community. A low power design is mainly appreciated in all hardware architecture to obtain efficient output. The low power design problems can be broadly classified into two: analysis and optimization. Analysis problems are concerned about the accurate estimation of the power or energy dissipation at different phases of the design process. Optimization is a process of generating the best design given an optimization goal, without violating design specification. Analyze techniques also serve for design optimization but major criteria to be considered are the impact of circuit delay which affects throughput and performance of circuit. Other factors are design cycle time, reliability, testability, quality, reusability. There is great interest in understanding how to continue increasing performance without increasing the power dissipation.

Depending on the number of samples processed at each machine cycle, there are two architectures for hardware design, namely, bit level and word level. In the bit-level architecture, the samples are processed in parallel, and the bits of the incoming samples are sequentially processed [1],[2],[3],[5],[8]. On the contrary, the word-level samples are sequentially processed word by word, and the bits of the sample are processed in parallel [4], [6],[7],[10],[11],[12]. Cadenas et al. [1] proposed a novel quantum like representation allowing the median to be computed in an accelerated manner compared to the best-known method.

The median output can be generated in *W/B* clock cycles, where *B* is a parameter denoting the number of sample bits to be processed at a time. Chen et al. [2] have proposed low-power 2's complement multipliers by minimizing the switching activities of partial products using the radix-4 Booth algorithm. Prokin & Prokin [9] adopt a bit-level pipelined architecture to generate the median output in *W* clock cycles, where *W* is the sample width. Fahmy et al. used a histogram based method, and the efficiency can be shown only for larger windows. Moshnyaga & Hashimoto [7] have proposed a new architecture and circuit implementation of 1-D median filter using non recursive sorting. The work in [7] adopts the sorting network architecture, where the median of a set of samples is computed by sorting the input samples and then selecting the middle value. In their method, the input samples are sequentially processed word by word, and the incoming sample is inserted into the window in two clock cycles. In the first cycle, the oldest sample is removed from the window by moving some of the stored samples to the left. In the second cycle, the incoming sample is compared with all the samples in the window and then inserted in the right place by moving some of them to the right. This architecture has a latency of two clock cycles, and it generates only one median output for every two cycles.

In this paper, an efficient design of median-filter hardware, capable of delivering the median value in real-time under stringent requirements on power is proposed. A new word level one dimensional filter architecture is proposed, in which the new median output, with a filtering latency of only one clock cycle, is generated at each machine cycle for each incoming sample. The improvement is achieved by performing the deletion and insertion of samples in one clock cycle, and a new control circuit is proposed to govern these two operations. Section 2 presents the methodology used along with the architecture of the proposed1-D median filter and the description of the various functional blocks used in the design. The simulation results are presented in section 3 and conclusion is presented in section 4.

II. METHODOLOGY

In median filter architecture input samples are sequentially processed word by word and the incoming samples is inserted in to the correct rank in two steps. In the first step the oldest sample is removed from the window by moving some of the stored samples to the left. In the second step, the incoming sample is compared with the already sorted samples and then inserted in the right place by moving some of them to the right .In this median filter the rank of each sample can be generated and calculated by ripple carry adder. The main drawback of a standard median filter (SMF) is that it is effective only for low noise densities [8]. At high noise densities, SMFs often exhibit blurring for large window sizes .This will affect the entire image clarity and data loss. For some application large sample width may be required ,in these case more signal transitions is needed so dynamic power consumption is high. Delay is also a drawback of some architecture. Adaptive Median is a **"**decision-based**"** filter that first identifies possible noisy pixels and then replaces them using the median filter or its variants, while leaving all other pixels unchanged. New median filter architecture is designed to reduce the power consumption. Here instead of sorting samples physically in the window sorted samples are kept immobile, only rank of each samples are updated at each cycle .Since, the architecture is implemented in two stage pipeline the median output is the sample with median rank. Median filter architecture could be used the resources with the less power consumption. The proposed architecture is aimed at minimizing the power consumption and thus making it suitable for all hardware related sources and applications.

A. Median Filter Architecture

Figure 1 shows the architecture of the proposed one dimensional median filter. This architecture gives an overview of low-power median filter with window size N. It consists of a circular array of N identical cells and three auxiliary modules: i) Median selection ii) Rank calculation
iii) Rank selection. All the cells are connected to a global input register X, through which they receive the incoming sample, and the median is stored in the output register Y.

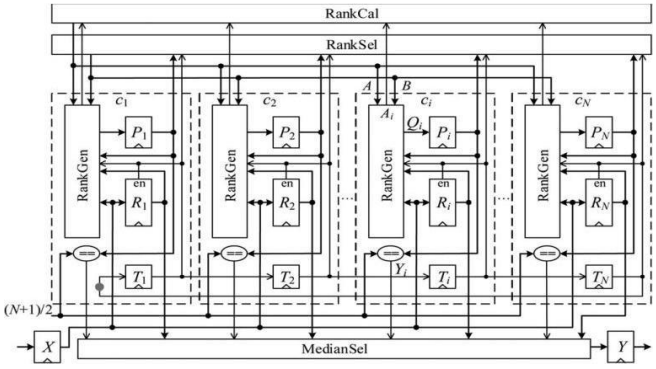

Figure 1. Architecture of one dimensional median filter

The architecture is implemented as a two-stage pipeline, where the registers in all the cells serve as the internal pipeline registers. All the registers in the architecture are synchronized by the rising edge of a global clock. Each cell block C_i is composed of a rank generation module, a comparator module **"=="** and three registers rank register P_i, a data register (R_i), and a 1-bit token register (T_i). Register R_i stores the value of the sample in cell c_i, register P_i keeps the rank of this sample, and the enable signal (en) of R_i is stored in register T_i.All the samples in the window are ranked according to their values, regardless of their physical locations in the window. In our design, a cell with a greater sample value will be associated with a greater rank. However, for two cells ci and cj, whose sample values are equal, c_i will be given a greater rank if R_i is newer than R_j i.e., the sample in c_j enters the window earlier than the sample in c_i. The rank is hence unique for each cell. For a window with size N, the rank starts from 1 for a cell with the least sample value, and ends with N for a cell with the greatest sample value. The median of the window can then be obtained from the sample value R_i of a cell c_i whose rank

P_i is equal to (N + 1)/2, assuming N is an odd number. In the architecture, the input sample enters the window in a FIFO manner. After it is queued, it will not be moved and is simply de-queued in place. A token, which is represented as logic state 1 in the token register of some cell, is used to control the de-queuing of old sample and queuing of new input sample at the same time. After the token is used, it will be passed to the next cell at a new cycle. All the token registers

form a token ring with exactly one token. It serves as a state machine in the circuit, where the location of the token defines the state of the machine. Since the data in each register R_i is immobile, the proposed architecture has the potential for low-power applications. At the initial state of the architecture, to make the first incoming sample be stored in the first cell C_i, the token is assumed to exist in the last cell C_n, shown as the shadowed circle at the output of register T_n.Whenever an input sample enters the window at a new cycle, the rank of each cell has to be updated. It may have to be recalculated, or may be the old rank decremented by 1, incremented by 1, or kept unchanged. The new rank of each cell, which is denoted by signal Qi in the cell, is generated by the RankGen module. Each RankGen module receives signal A from the RankCal module and signal B

from the RankSel module. Signal A is the rank of a cell C_i that contains the token and signal B is the old rank of C_i. Moreover, signal Yi is the output of a comparator module **"=="** which compares the value of rank Pi with a constant value (N + 1)/2 so that VY_i= 1 if P_i is equal to $(N+1)/2$, else Y_i= 0. This signal is used to indicate if the corresponding cell C_i contains the median in Ri. Similar to the RankSel module, the MedianSel module transfers the value of Ri to the output register Y if Yi= 1; i.e., if the median is stored in R_i.At each machine cycle, the architecture performs the following operations: find the cell with the oldest sample (called the *oldest cell*), find the cell for the input sample to insert into the window (called the *insertion cell*), and update the sample in each cell. The oldest cell is the one whose age is equal to *N*. Since only one input sample is allowed to enter the window at each cycle, there is only one oldest cell in the window. When the input sample is inserted in the insertion cell c_j , the original sample in c_j and those in the cells between c_j and the oldest cell c_k will all be shifted toward c_k for one cell. The original sample in c_k is discarded to provide a new space for storing the sample shifted in from its adjoining cell. The insertion cell is chosen in such a way that the sorting of samples in the window is still preserved if the input sample is inserted there. Since the original sample in the insertion cell may be shifted left or right depending on the position of the oldest cell, a left-insertion (LI) [right-insertion (RI)] cell is defined as an insertion cell whose original sample will be shifted left (right) when the input sample is inserted there. To keep the sorting of samples when the input sample is inserted, a cell c_j will be the LI cell if it is the last cell in the window from left to right whose sample is larger than *X*, i.e., $R_j > X \geq R_{j+1}$, for $j = 1$ *to* $N-1$. If c_j is the rightmost cell c_N in the window, it will also be the LI cell when $R_N > X$. On the other hand, if c_j is the first cell in

the window whose sample is smaller than or equal to *X*, i.e., $\boldsymbol{Rj-1 > X \geq} Rj$, for $j=2$ to *N*, it will be the RI cell. If c_j is the leftmost cell c_1 in the window, it will also be the RI cell when $X \geq R_1$. In general, the LI and RI cells are two adjoining cells in the window with the LI cell on the left and the RI cell on the right. However, when the rightmost cell is the LI cell, there will be no RI cell, and when the leftmost cell is the RI cell, there will be no LI cell, either.

Cycle	Input Sample	c_1	c_2	c_3	c_4	c_5	c_6	c_7
1	25	0	0	0	0	0	0	0
2	53	25	0	0	0	0	0	0
3	44	53	25	0	0	0	0	0
4	32	53	44	25	0	0	0	0
5	40	53	44	32	25	0	0	0
6	22	53	44	40	32	25	0	0
7	55	53	44	40	32	25	20	0
8	36	55	53	44	40	32	25	20
9	29	55	53	44	40	36	32	20
10	39	55	44	40	36	32	29	20

Figure 2. Example Illustrating the insertion of 10 Samples

If the oldest cell c_k is on the left (right) of the LI (RI) cell cj , the input sample is inserted in cj , and the original sample in cj and those in the cells between c_j and c_k will all be shifted one cell-space left (right). The last of the shifted samples in cell ck+1 ($\boldsymbol{ck}$−1) is then shifted into the oldest cell c_k for the left (right) shift operation. If the oldest cell ck happens to be the LI or RI cell, the input sample is directly inserted in ck without shifting any sample between cells.

An example illustrating the insertion of 10 input samples into a window with seven cells is given in Figure 2. Initially, when the sample 25 enters the window at cycle $t0$, it is inserted in the RI cell c_1. In the figure the cells coloured in green indicate the RI cells and cells coloured in saffron indicate the LI cells. After the passing of seven clock cycles, the window is fully occupied with valid data at cycle $t7$, and cell c_6 is the oldest cell, coloured in blue. At this time, since c_6 is on the right of the RI cell c_3, the input sample 36 is inserted in c_5 and the samples in cells c_6, and c_7 are all shifted one cell-space right. When an input sample is inserted into the window along with some necessary shift operations, each cell in the window may keep its original sample, receive the input sample, or receive the sample from its adjoining cell on the left or right.

B. Rank Generation

Signal F_i is the output of a comparator module **"<=,"** which compares the value of R_i with that of the input sample *X* so that F_i = 1if R_i is less than or equal to X ($R_i \leq X$), else F_i = 0 (R_i >X). Signal A_i is the output of a logic AND gate so that A_i = 1 if T_i = 0 and F_i = 1; i.e., if cell c_i, does not contain the token and Ri is less than or equal to X, else A_i = 0. The A_i signal of each cell is connected to the RankCal module,

which calculates the new rank of a cell that contains the token. The new rank is calculated as K + 1, where K is the number of cells, which does not contain the token and whose sample value is less than or equal to X; The RankCal module can be implemented by a multi-input adder that adds all the Ai signals and then increments the sum by 1. Figure 3 shows the RankCal module. If cell c_i contains the token, the output A of the RankCal module will be its new rank at the next cycle. On the contrary, if cell c_i does not contain the token, its new rank will be determined by the other signals. Signal G_i is the output of a comparator module **">,"** which compares the value of rank P_i with that of signal B so that G_i = 1 if P_i is greater than B, else Gi = 0. Since the value of B is the rank Pj of another cell cj that contains the token, the meaning of G_i can also be described as G_i = 1 if P_i is greater than P_j (P_i > P_j), else G_i = 0(P_i <= P_j). Signal E_i is the output of a compactor module **"==,"** which also compares the value of Pi with that of B so that E_i = 1 if P_i is equal to B, else Ei = 0. Likewise, the meaning of Ei can be described as E_i = 1 if Pi is equal to P_j (P_i = P_j), else E_i = 0. Combining these two signals E_i and G_i, for the three relations between P_i and P_j (P_i > P_j , P_i = Pj , and P_i< P_j), the corresponding value of E_iG_i will be 00,10,11 respectively.

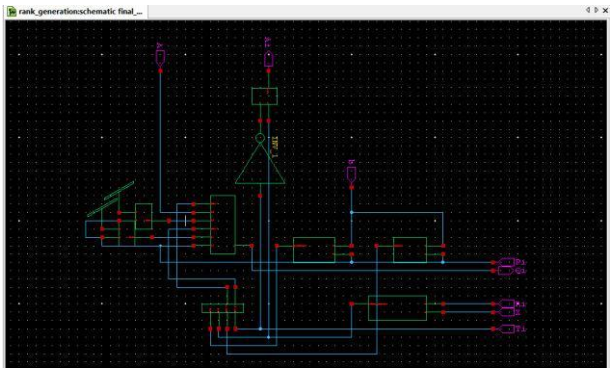

Figure 3. Rank Generation Module

C. Rank Selection

The RankSel module is responsible for transferring the rank P_i of a cell c_i to its output B if c_i contains the token; i.e. when T_i=1. It shows a simple implementation of this module using AND/OR gates. It can also be implemented by tristate buffers, where B is the output of a global data bus that collects the output signals of all the tristate buffers. Since there exists exactly one cell that contains the token at any time; i.e., there exists exactly one Ti signal whose value is equal to 1, the value of B will always be valid. The Median Selection module can also be implemented in a similar way. It transfers the value of R_i to the output register Y if R_i is the median; i.e., when Y_i = 1. However, if it is implemented by tristate buffers, the median will be valid only when there exists at least (N + 1)/2 samples in the window; otherwise, it will remain in a high impedance state. For a cell c_i, since there are four possible sources to update its rank, a 4-to-1 multiplexer is used to select one of these sources for signal Q_i. Then, the value of rank Pi will be updated by the value of Qi at each cycle. The multiplexer is controlled by two selection signals S1 and S0; and these two signals, which are generated by the Ctrl module, are determined by four signals T_i, E_i, F_i, and G_i. If cell c_i contains the token (T_i = 1), its new rank is obtained from the output A of the RankCal module; i.e., the value of S1S0 should be 11 when T_i = 1. If cell c_i does not contain the token (T_i = 0). In Case 1 when P_i > P_j (E_iG_i = 01) and $R_i \leq X$ (F_i = 1), the rank of c_i will be decremented by 1; i.e., the value of S_1S_0 should be 10 when $E_iF_iG_i$ = 011.

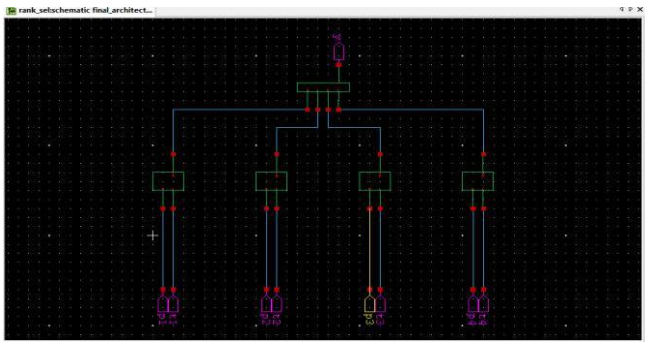

Figure 4. Rank Selection Module

D. Control Module

Figure 5 shows the control module. For a cell c_i, since there are four possible sources to update its rank, a 4-to-1 multiplexer is used to select one of these sources for signal Q_i. Then, the value of rank P_i will be updated by the value of Q_i at each cycle. The multiplexer is controlled by two selection signals S_1 and S_0; and these two signals, which are generated by the Ctrl module, are determined by four signals T_i, E_i, F_i, and G_i. If cell c_i contains the token (T_i = 1), its new rank is obtained from the output A of the RankCal module; i.e., the value of S_1S_0 should be 11 when T_i = 1. If cell ci does not contain the token (Ti = 0). In Case 1 when P_i > P_j (E_iG_i = 01) and $R_i \leq X$ (F_i = 1), the rank of c_i will be decremented by 1; i.e., the value of S1S0 should be 10 when $E_iF_iG_i$ = 011. Similarly in Case 2, when P_i < P_j (E_iG_i = 00) and R_i > X (F_i = 0), the rank of c_i will be incremented by 1; i.e., the value of S_1S_0 should be 01 when $E_iF_iG_i$ = 000. Finally, when P_i < P_j (E_iG_i = 00) and R_i <= X (F_i = 1) in Case 3, P_i > Pj (EiGi = 01) and R_i >X (F_i = 0) in Case 4, and P_i = P_j (E_iG_i = 10) in Case 5, the rank of ci will be kept unchanged; i.e., when $E_iF_iG_i$ = 010,001, the value of S_1S_0 should be 00.

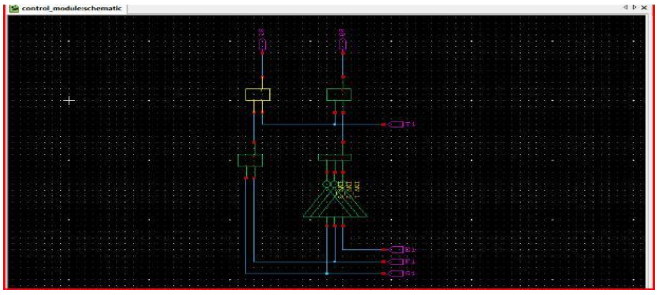

Figure 5. Control Module

III. EXPERIMENTAL RESULTS

The low-power architecture of the proposed 1-D median filter design was implemented TANNER EDA tool and synthesized using Synopsys Design Vision with the 180nm cell library. Figure 6 shows the four bit low power architecture and Figure 7 shows architecture of the single cell. Figure 8 shows the simulation results. Figure 9 (a) shows the power results for the existing technique and Figure 9(b) shows the power results for the proposed modified architecture. The power consumption is reduced. The output power is reduced when compared to the previous existing method.

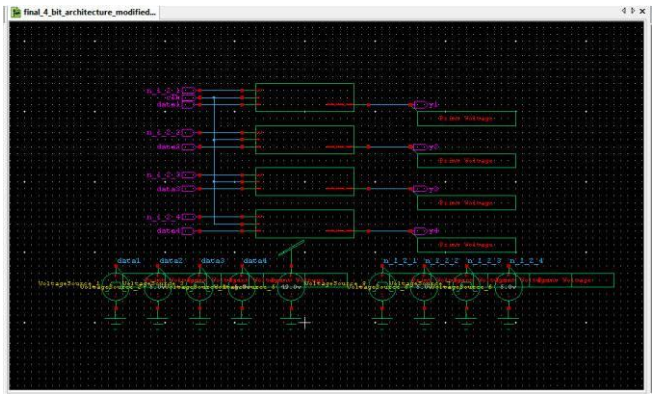

Figure 6. Four Bit Low Power Architecture

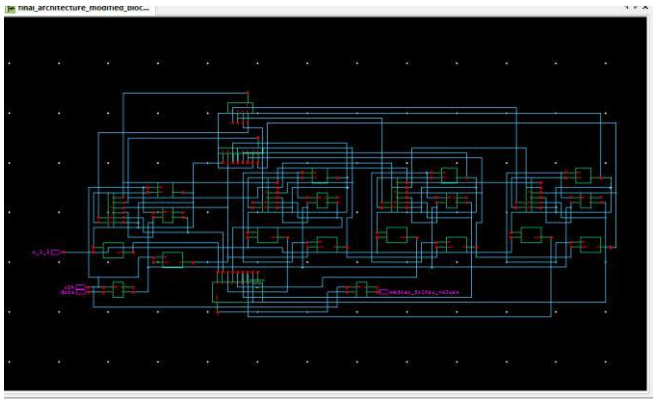

Figure 7. Architecture of the Single Cell

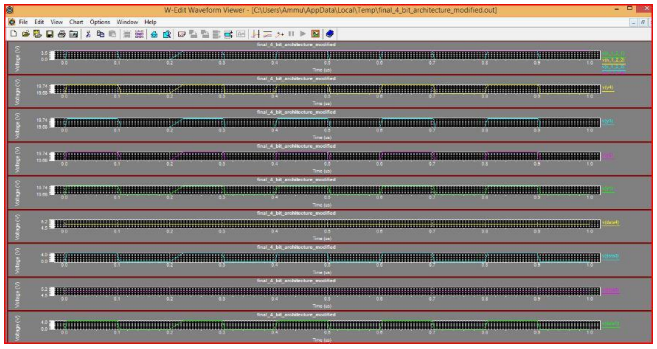

Figure 8. Simulation Results

```
Power Results
vdd gnd from time 0 to 100
Average power consumed -> 5.833027e-005 watts
Max power 5.839030e+003 at time 1e-007
Min power 5.827269e+003 at time 3.05e-007

* END NON-GRAPHICAL DATA
*
* Parsing                          0.03 seconds
* Setup                            4.84 seconds
* DC operating point              41.59 seconds
* Transient Analysis              15.43 seconds
* Overhead                         2.94 seconds
* ---------------------------------------------
* Total                           64.82 seconds

* Simulation completed  with  1 Warning

* End of T-Spice output file
```

(a)

```
Power Results
vdd gnd from time 0 to 100
Average power consumed -> 5.295103e-005 watts
Max power 5.302994e+003 at time 5e-009
Min power 5.287528e+003 at time 1e-006

* END NON-GRAPHICAL DATA
*
* Parsing                          0.08 seconds
* Setup                            5.19 seconds
* DC operating point               7.59 seconds
* Transient Analysis              14.89 seconds
* Overhead                         3.02 seconds
* ---------------------------------------------
* Total                           30.77 seconds

* Simulation completed

* End of T-Spice output file
```

(b)

Figure 9. Power Results(a,b)

IV. CONCLUSION

In this paper a power efficient 1-D word level median filter design is proposed. The filters receive an input sample at each clock cycle and generate a median output at the same cycle. When an input sample enters the window, the sorting of existing samples is reused in generating the new median output. The hardware complexity of the architecture is directly proportional to the number of samples in the window. The simulation results have shown that the proposed architecture is performs better in terms of power consumption and also the reduction in delay enhances the speed of the hardware.

REFERENCES

[1] J. Cadenas, G. M. Megson, R. S. Sherratt, and P. Huerta, "Fast median calculation method," Electron. Lett., vol. 48, no. 10, pp. 558– 560, May 2012.

[2] C.-T. Chen, L.-G. Chen, and J.-H. Hsiao, "VLSI implementation of a selective median filter," IEEE Trans. Consum. Electron., vol. 42, no. 1, pp. 33–42, Feb. 1996.

[3] C. Choo and P. Verma, "A real-time bit-serial rank filter implementation using Xilinx FPGA," in Proc. SPIE, vol. 6811, Real-Time Image Processing, 2008, pp. 68 110F-1–68 110F-8.

[4] S. A. Fahmy, P. Y. K. Cheung, and W. Luk, "High-throughput onedimensional median and weighted median filters on FPGA," IET Comput.Digit. Technol., vol. 3, no. 4, pp. 384–394, Jul. 2009.

[5] M. Karaman, L. Onural, and A. Atalar, "Design and implementation of a general-purpose median filter unit in CMOS VLSI," IEEE J. Solid-State Circuits, vol. 25, no. 2, pp. 505–513, Apr. 1990.
[6] C.-Y. Lee, P.-W. Hsieh, and J.-M. Tsai, "High-speed median filter designs using shiftable content-addressable memory," IEEE Trans. Circuits Syst.Video Technol., vol. 4, no. 6, pp. 544–549, Dec. 1994.
[7] V. G. Moshnyaga and K. Hashimoto, "An efficient implementation of 1-D median filter," in Proc. 52nd IEEE Int. MWSCAS, 2009, pp. 451–454.
[8] D. Prokin and M. Prokin, "Low hardware complexity pipelined rank filter," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 57, no. 6, pp. 446–450, Jun. 2010.
[9] D. S. Richards, "VLSI median filters," IEEE Trans. Acoust., Speech,Signal Process., vol. 38, no. 1, pp. 145–153, Jan. 1990.
[10] V. V. R. Teja, K. C. Ray, I. Chakrabarti, and A. S. Dhar, "High throughput VLSI architecture for one dimensional median filter," in Proc. ICSCN, 2008, pp. 339–344.
[11] Z.Vasicek and L.Sekanina, "Novel hardware implementation of adaptive median filters," in Proc. 11th IEEE Workshop DDECS, 2008, pp. 1–6.

S.Deepa is currently working as Professor in the Department of Electronics and Communication Engineering in Panimalar Engineering College, Chennai. She received her Bachelor of Engineering degree in Electronics and Communication Engineering from Madras University in the year 1998. She completed her Masters in Engineering in Applied Electronics in the year 2005 and received doctorate in image processing in the year 2014, both from Sathyabama University, Chennai. She has a total teaching experience of about 14 years and her research interest includes Image Processing, Computer Vision, Pattern Recognition, Low power VLSI design and architecture.

B.Sathyabhama is currently working as assistant professor in the Department of Electronics and Communication Engineering in Panimalar Engineering College, Chennai. She received her Bachelor of Engineering degree in Electronics and Communication Engineering from DMI college of engineering in the year of 2010. She completed her Masters in Engineering in VLSI design in the year of 2012 in Easwari engineering college, Chennai under Anna university of Chennai. Her current research interests include field-programmable gate array (FPGA) architectures, computer-aided design (CAD) tools for FPGAs, logic synthesis, and low power VLSI design.

M.D.Manigandan completed his Bachelor of Bachelor of Engineering degree in Electronics and Communication Engineering from Panimalar Engineering College in the year of 2015. Currently he is pursuing his Masters in Communication Engineering in Panimalar Engineering College, affiliated to Anna University Chennai. His current research interests include Medical Image Processing, field-programmable gate array (FPGA) architectures and low power VLSI design.