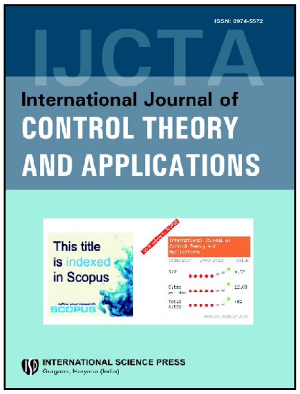


International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 9 • Number 48 • 2016

Area Efficient Implementation of One-Dimensional Median Filter using BEC CSLA

[1]Tulasiram K., [2]Y. Rajasree and [3]K. Anitha Sheela

[1] *Research Scholar, JNTU, Hyderabad, TS, India*
[2] *Professor, Department of ECE, St.Peter's Engineering College, Hyderabad,TS, India*
[3] *Professor, Department of ECE, JNTU , Hyderabad, TS, India*
Corresponding Author E-mail: ramukorrapati@gmail.com

Abstract: This paper presents a circuit implementation and new architecture of one dimensional median filter. Normally, digital adder affects the overall circuit performance. The proposed method low area carry select adder (CSLA) is mostly used in digital circuits and high speed applications. Due to the presence of two Ripple Carry adders (RCA) in the structure, regular square root CSLA absorbs more power and more area. In proposed method instead of using RCA, Binary to Excess-1 Converter (BEC) is used to reduce the area and power.

Keywords: BEC, RCA, CSLA, Excess-1

1. INTRODUCTION

Image enhancement and the noise filtering are the main application in the image processing. In the method of visual interpretation, these two tasks are very essential in image processing [1]. In previous years, linear filter has been used for noise removal and edge preserving. During this process, the data loss is main problem of linear filter. To minimize data loss problem consider the nonlinear filter, it can be performed edge preservation without any data loss. The inadequacy of image sensors causes the noisy images. Impulse noise is mostly affected by memory location hardware, camera sensors and errors during the data transmission. Impulse noise is commonly classified as two types such as random valued shot noise and salt and pepper noise. In random values shot noise, the arbitrary value can be assigned to the noisy pixels. If the salt and pepper noise affected to the image, minimum or maximum values can take by noisy pixels. So it is hard to abolish these kinds of noise using linear filters. To conquer this problem move to median filter [2]. With the help of median filter, without losing high frequency the impulse noise can be smoothened. By using median filter, it is possible to dismiss the impulse noise from the images, smoothen the transient signals and preserves the edge evidence [3].

Median filter can be separated into three methods such as sorting network architectures, array architectures and stack based architectures. In the sorting network architectures, before choose the sample of corresponding rank to perform ranging the samples. Due to the additional amount of compare-swap units, these architectures produced higher throughput. In array architectures, each element of the window is connected to the rank. The ranks are updated when the window moves to the succeeding position. In the stack based architectures with the aid of majority of elements, hamming comparators, and threshold logic, translate the filtering into the binary domain [4]. Depending on the number of samples, the hardware design divided into two types such as word level architecture and bit level architecture [5]. In the word level sorting, the input samples can be processed word by word in sequentially, and the receiving sample can be injected into the correct rank in two steps. The first step while moving the samples to the left, the previous sample is detached from the window. The arriving sample can be inserted in the right place after comparing the receiving sample with already organised samples, which performs is second step [6].In the bit level architecture, the samples are collected in parallel and incoming sample bits are sequentially processed [7]. Two clock cycles are needed for performing these two different architectures. These architectures require additional signal transitions in the circuit, large sample width and more dynamic power. To conquer this problem, the proposed model benefits in the new median filter with CSLA. In digital adders, to propagate a carry the speed of count is restricted by time. The implementation of a Rank generator modules 3 bit, 4 bit, 5 bit and 6 bit digital adders are required to generate the multiplexer output [8]. In proposed method instead of using digital adders, we have to use CSLA adder to produce the multiplexer output [9], [10]. Fast arithmetic function can be performed in CSLA, which is one of the quickest adders helps in various data processors. Minimizing the size and improvement in power consumption can be achieved by CSLA.

2. RELATED WORK

J. O. Cadenas*et al.* [11] proposed a median filter enterprise in parallel counters. In this paper, based on accumulative parallel counters (APC) to usual the positive integer window. The normal digital adder is utilize to diminish the area and power consumption. But not care about time delay.

R. D. Chen *et al.* [12] presented an area-efficient one dimensional median filter based on the sorting network. The window is sorting in descending order in word level filter. The new sample get into the block. But, the old sample not following the queue that causes the collision in the sample window.

Kamarujjaman *et al.*[13] proposed an effective attitude to dominance algorithm and its VLSI design for suppression of salt and pepper noise with higher density. This paper also using FPGA implementation, but not care about the area and power dissipation.

Mukherjee, M. *et al.* [14] presented a low complexity reconfigurable hardware architecture for adaptive median filter. In this paper, only concentrate on mean square error (MSE) and peak signal to noise ratio (PSNR). Here, the power and area is restricted one. So can’t decrease below from its restricted range.

A. Pereverzev *et al.* [15] developed the architecture of 1-D median filter which permitted to increasing the length of the aperture. The normal digital adder is utilize in the Verilog description. In this paper not possible to reduce the parameter such as power and area for low level.

3. PROPOSED METHOD

The proposed architecture consists of low power CSLA, MUX and logic gates. The efficiency of the architecture can be estimated by using whole power utilization, area utilization and maximum frequency. The working principle of the architecture is described below:

3.1. Filter architecture

This structure consists of three auxiliary modules such as rank calculation (Rank Cal), rank selection (Rank Sel), and median selection (Median Sel) and N identical cells, which is shown in the fig. 1. All the modules are linked to the X input register and the median can be kept in the Y output register. With help of growing edge of a total

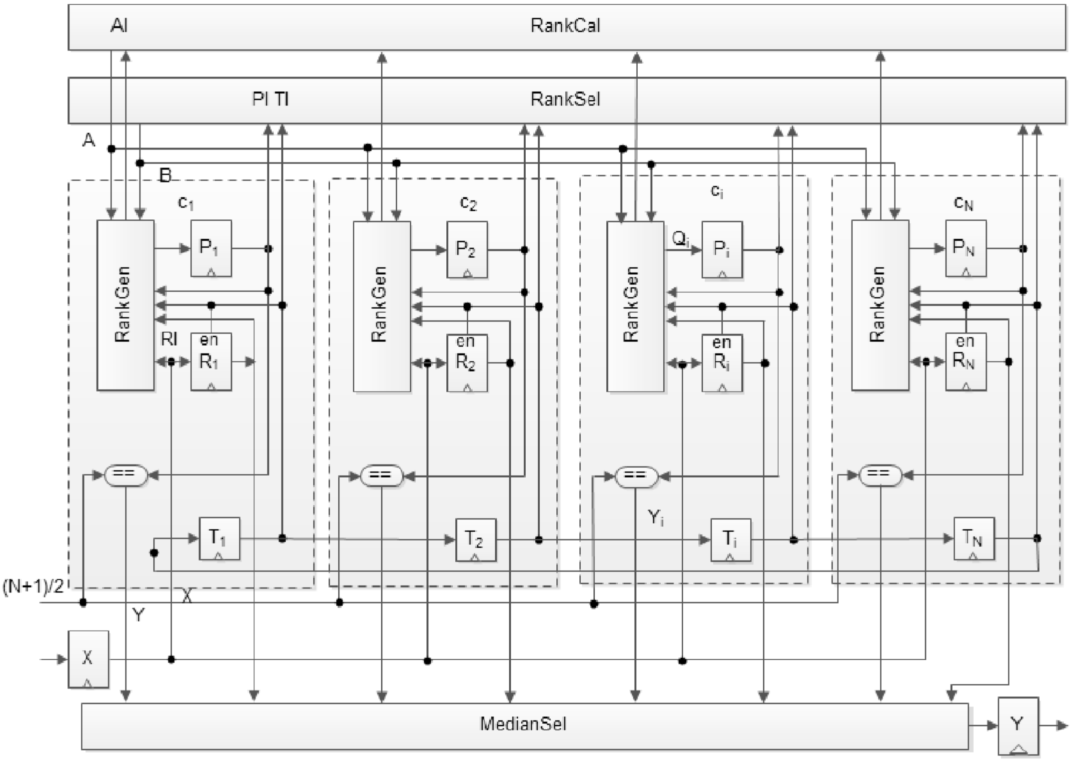


Figure 1: Low power one dimensional median filter architecture

clock, the register architecture can be synchronized. Each cell block ci consists of three registers such as rank register (Pi), token register (Ti), and data register (Ri). The sample cell ci stores in the Ri register, the rank of the sample keeps in Pi register, and Ti keeps the permit signal of Ri. The rank starts from 1 for a cell with a least sample value, and ends with N for a cell with the greatest example value in the N window size. The example value Ri of a cell ci whose rank Pi is equal to (N+1)/2, where odd number represent as N. In this architecture, based on FIFO method the input model enters into the block. Once the sample is queued, it won't be de-queued. If the sample keeps the token 1, it helps to resist the queuing of new input model and de-queuing of old sample at the same time. Once the token is utilized it will give to the succeeding clock cycle. Immobile of Ri, our architecture operates low power applications.

At the first stage, the received sample can be kept in the ci, exit cell mentioned as c_N and the shadow circle output mentioned as TN. the rank cell can be updated, whenever the input sample arrives through the window. It pivot on the token, the circuit performs differently.

3.1.1. Circuit behaviour

Initially, the two-stage pipelined filter executes the succeeding operation for every machine cycle t_i and input sample X: Initially, need to find the innovative rank cell of each cell. Then, X value insert to the cell that cell holds the token and the token will proceed to the following cell. At the next cycle $t_{i+1,}$ the new values will be compute and updated for all T_i, R_i, and P_i registers. The middle value will be evaluated by using second pipelined stage for the input sample enters into the aperture at the preceding cycle $t_{i-1.}$ The updated value at the upcoming cycle t_{i+1} will be determined in the output register Y. A window contains nine input window and five cells are given in Tab.1. All the registers such as Pi, Ri, and Ti for each and every cells Ci given in table. First input sample stored in first cell Ci, and the last cell is condescend to hold the taken (T_5=1) at t_0 clock cycle. The two output/input registers Y and X along with the sample and rank values (Pi and Ri) of every cells are rearrange to be zero.

At cycle t_1, the first sample enters into the window. At the time, the token has been passed to c_1 from c_5 (T_1 =1 and T_5=0). At cycle t_0, the P_5 value won't change from zero. Since c_1 holds the token, the new value of R_1 represented as 12 to cache the input sample at t_2 clock cycle. Since, the sample rate of other four cells is lesser than the sample value of 12, which helps to calculate P_1 value as 5. To indicate the token move to c_2 from c_1, the new values of T_2 and T_1 can be calculated as 1 and 0 respectively. At the following cycle t2, all the Pi, Ri, and Ti registers values will be updated. At cycle t_6, the cell c1 holds the token again (T_1=1), when the block fully busy with useable data. At cycle t_7, the median output Y will be enumerated as the value of R_4 (47) since the value p_4 is equivalent to 3, i.e., (5+1)/2. All the Pi, Ri, and Ti registers also refurbish at the cycle, when the Y is updated to be 47 at cycle t_7 for the input sample 66. The new median can be calculated in a cell whose rank is equivalent to (N+1)/2.

3.2. Rank updating

Table 1
Example illustrating the insertion of nine input samples into a window

Clk	Input Reg X	Cell Registers														Output Reg Y
		T1	T2	T3	T4	T5	R1	R2	R3	R4	R5	P1	P2	P3	P4	P6	
t0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
t1	12	1	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0
t2	59	0	1	0	0	0	12	0	0	0	0	5	0	0	0	4	0
t3	35	0	0	1	0	0	12	59	0	0	0	4	5	0	0	3	0
t4	47	0	0	0	1	0	12	59	35	0	0	3	5	4	0	2	0
t5	66	0	0	0	0	1	12	59	35	47	0	2	5	3	4	1	12
t6	52	1	0	0	0	0	12	59	35	47	66	1	4	2	3	5	35
t7	38	0	1	0	0	0	52	59	35	47	66	3	4	1	2	5	47
t8	18	0	0	1	0	0	52	38	35	47	66	4	2	1	3	5	52
t9	26	0	0	0	1	0	52	38	18	47	66	4	2	1	3	5	47

Table 1 shows the insertion of nine input samples into a window. In this case, two major operation can be considered such as cell with token and cell without token. If the sample contains the token, the sample value replaced with the input sample and rank can be recalculated. The window sample will be inserted, which sample don't occupy in the block cell.

3.2.1. Cell with the Token

For a c_i cell holds the token, by using input, X the R_i value can be replaced and also P_i has to be reformed. By comparing X with the sample values of other N-1 cells the new P_i value can be gained that don't contain the token. The fresh value of P_i will be K+1, when the number of cells (K) whose sample value is fewer or equivalent to X. For example, at clock cycle t_6 the novel rank value P_1 will be evaluated as 2+1 at the following cycle t_7.

3.2.2. Cell without the Token

In the method of cell without the token, there are five cases will be used such as decremented by 1, incremented by 1, rest of the three methods are unchanged.

Case I – (Decremented by 1)

In this case, check the condition Pi >Pj and Ri<= X. If the condition is fulfilled the reference value will be reduced by 1. At next clock cycle, the ref value is fewer or equivalent to Rican be declined by 1. i.e.., Pi has to be reduced by 1. For example, take a cycle t_3 at rank P_1 that value will be discount by 1 (from 4 to 3) at the following cycle t_4.

Case II – (Incremented by 1)

In this case, check the condition Pi <Pj and Ri> X. If the condition is gratified the current window will be raised by 1. At next clock cycle, the sample value is fewer or identical to Ri will be upturned by 1. i.e. ., Pi has to be hiked by 1. For example, take a cycle t_7 at rank P_4 that value will be inflation by 1 (from 2 to 3) at the succeeding clock cycle t_8.

Case III – (kept unchanged)

In this case, check the condition of Pi <Pj and Ri<= X. If this condition satisfied the reference value won't be changed. . At next clock cycle, the number of current block is fewer or equal to Ri at the current cycle can be identical i.e. ., Pi has to be kept unchanged. For example, take a cycle t_7 at rank P_3 that value unchanged at the following clock cycle $t_{8.}$

Case IV – (kept unchanged)

In this case, check the condition of Pi >Pj and Ri> X. If this condition satisfied the sample value won't be changed. . At next clock cycle, the sample value is less than or equal to Ri at the current cycle will be identical i.e.., Pi has to be kept unchanged. For example, take a cycle t_3 at rank P_2 that value will be unchanged at the succeeding clock cycle $t_{4.}$

Case V – (kept unchanged)

In this case, if Pi = Pj the sample will be unchanged. When the block is not yet fully busy with valid data, this case will be occurred. Initially, the rank of every cells resets to be zero. Once the window is fully busy, each cells set to be non-zero value and single rank. For example, take a cycle t_3 at rank P_4 that value will be zero at the following cycle $t_{4.}$

4. CIRCUIT IMPLEMENTATION

The rank generation module implementation in a cell c_i is shown in fig.2. The R_i and input X_i value can be performed "<=" operation, which gives the output F_i. If R_i greater than X , it will gives F_i=0 else F_i =1.

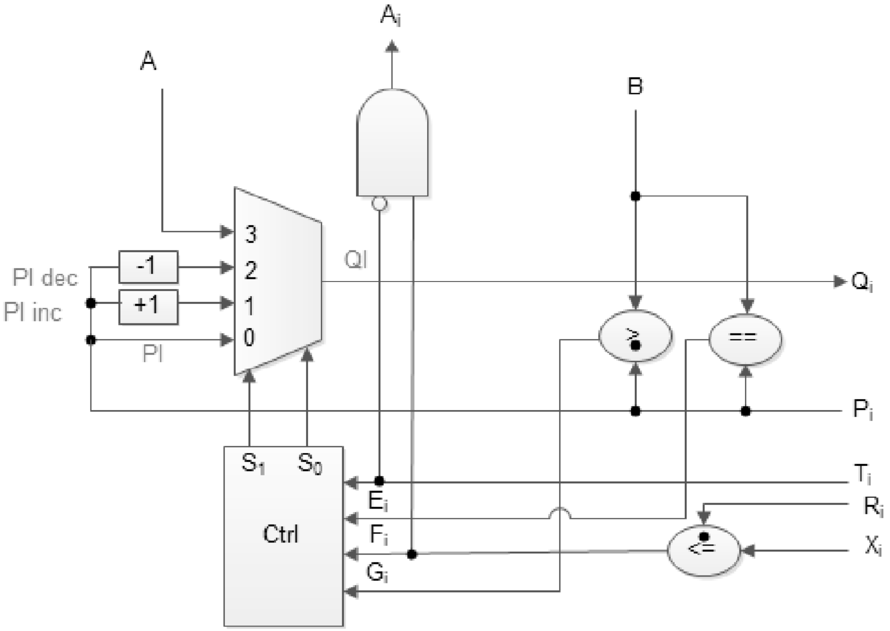


Figure 2: Implementation of rank gen module

If F_i =1 and T_i=0, the output of AND gate value should be A_i= 1. The output of AND gate gives a value A_i= 0, when R_i is fewer or equivalent to X and the cell c_i doesn't hold the token. Rank cal and A_i signal connecting together, which is charity to find the fresh rank cell that holds the token.

There are four sources is given to the input of 4:1 mux that delevers the one resourse signal Q_i, which is shown in fig.2. By using Ctrl module, two selection line (S_0, S_1) will be generated, which is apply to controlled the mux and deteremine the $F_{i,}$ $T_{i,}$ $G_i$andE$_i$ four signals.

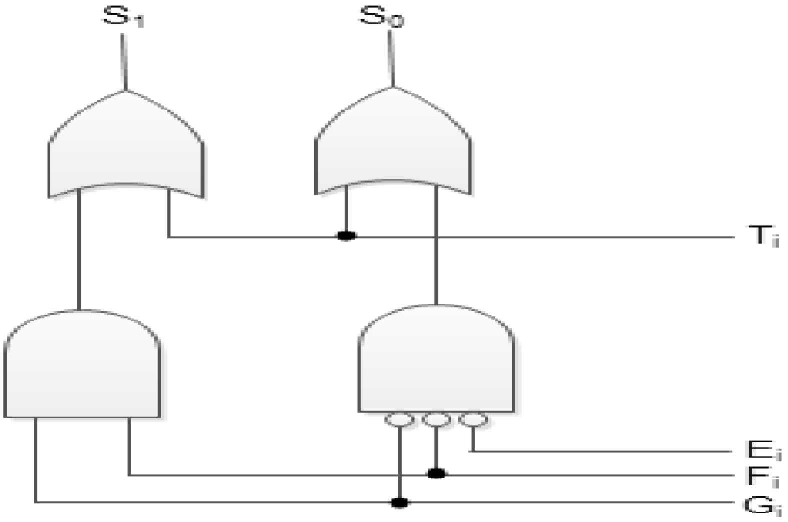


Figure 3: Implementation of the Ctrl module

The implementation of Ctrl module shown in fig.3. From the Rank cal module output A, the new rank will be reform if the cell c_i holds the token . if Ti=1, the S_0 and S_1value denoted as 11 else Ti should be 0.

To transferring the P_i rank to output B, the rank sel module can be used if c_i comprises the token when Ti=1. Fig. 4 shows the implementationof Rank Sel module using simple AND, OR gates. If the $T_i$mentioned as 1, this module gives the output B.

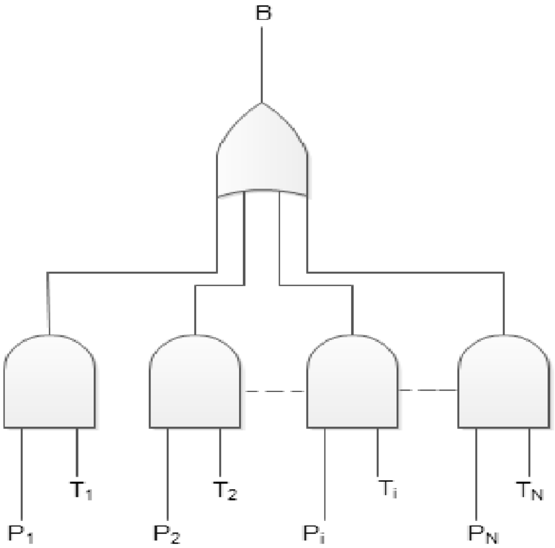


Figure 4: Implementation of Rank Sel module

The main proposal is instead of using normal adder CSLA adder can be used, which is given in fig.2. The major speed limitation in any adder is in the production of carries and many authors have considered the addition problem. The basic idea of the proposed work is using n-bit Binary to Excess-1 Converters (BEC) to improve the speed of addition. This logic can be implemented with Carry Select Adder to Achieve Low Power and Area Efficiency.

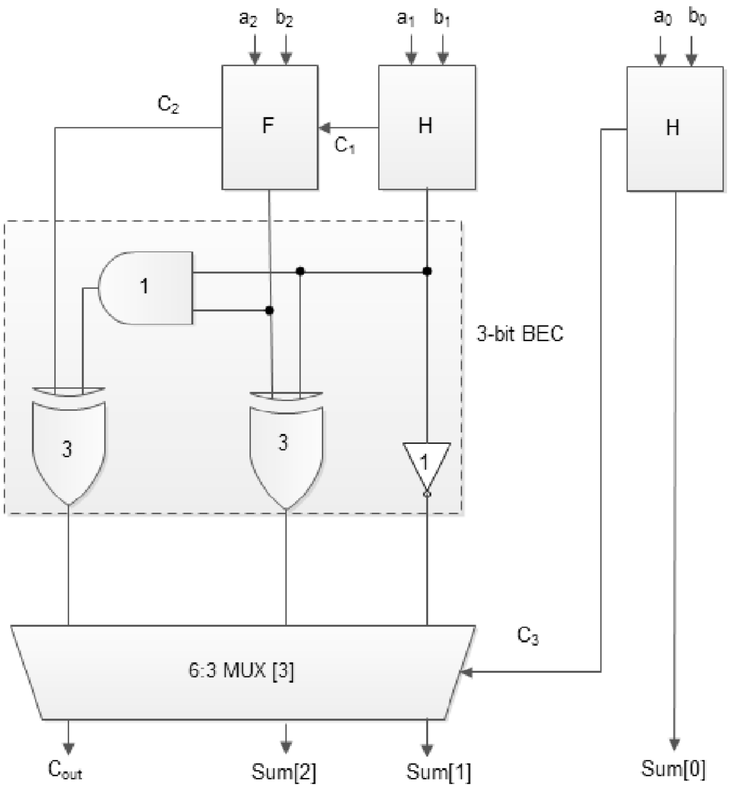


Figure 5: Low area carry select adder

The CSLA is used in many computational systems to reduce the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum. However, the CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input Cin = 0 and Cin = 1, then the final sum and carry are selected by the multiplexers (mux). The entire work performed by usage of Binary to Excess-1 Converter (BEC) instead of RCA with Cin = 1 in the regular CSLA to achieve lower power consumption The main advantage of this BEC logic comes from the lesser number of logic gates than the n -bit Ripple Carry Adder (RCA).

Table 2
Functional table of 3-bit BEC

B[2:0]	*X[2:0]*
000	001
001	010
010	011
011“‘	100
100	101

This carry select adder with BEC-1 is used in place of increment and decrement operations in rank generation module. For decrementing you need to add 2’s complement value of 1 to the count value.

4. EXPERIMENTAL SETUP

The proposed method simulated in Modelsim SE 10.1c using Verilog code and also the entire work is done by using I_7system with 8 GB RAM. Similarly, determination of area, power, and delay done by using cadence 180nm technology and RTL compiler.

5. RESULTS AND DISCUSSION

Table 2 and Table 3 gives the comparison of the existing and proposed method in terms of power, core area and delay for different 8 bit and 16 bit sample width. To find the energy for Lena, peppers and baboon image in 8-bit and three different audio in 16-bit.

Table 3
Experiments results for 8-bit

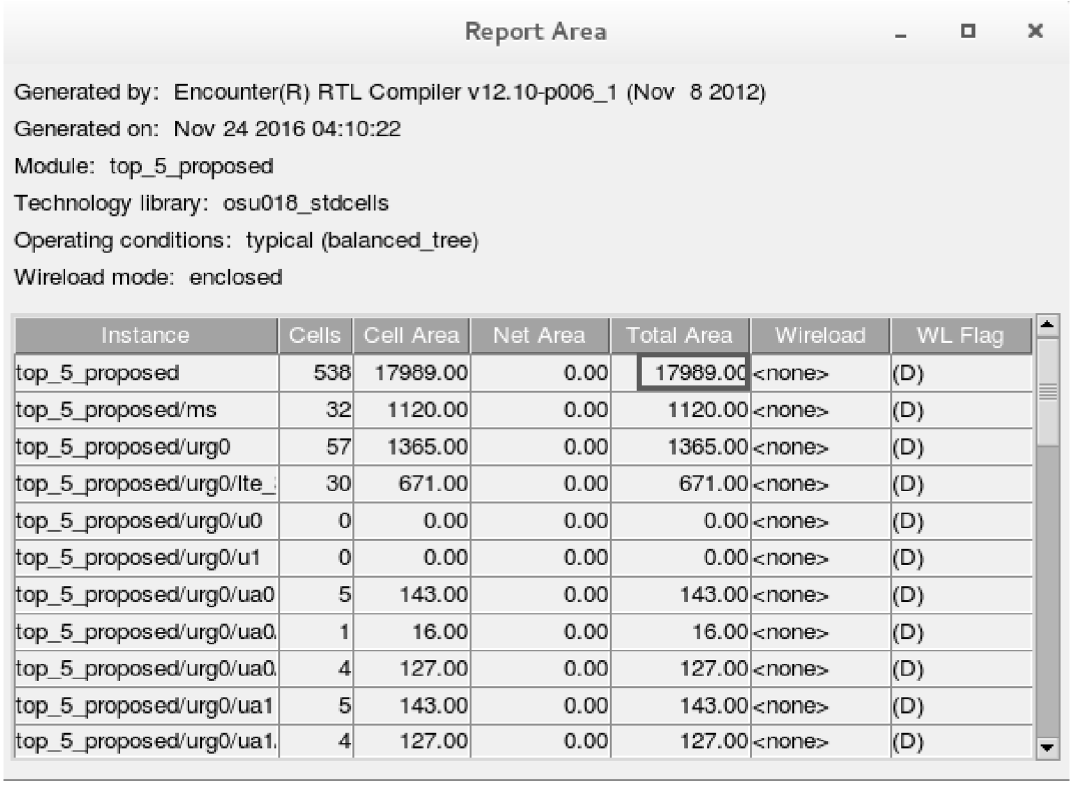
				8- bit sample width			*EPS(nJ)*			
Design	*Throughput (#median outputs/clock)*	*Latency (#clock cycles)*	*Window size*	*Core area (um^2)*	*Power (nW)*	*Delay (ps)*	*Lena*	*Peppers*	*baboon*	*Average*
Existing	1	w	5	39473.4	792366	3236.6	23.47	23.91	24.92	24.1
			9	79599.2	2073216	4297.7	61.42	62.58	65.21	63.07
Proposed	1	w	5	17989	1325922	1907	39.28	40.02	41.71	40.33
			9	34931	2694208	2728.9	79.81	81.33	84.75	81.96

Table 4
Experiments results for 16-bit

				16- bit sample width			*EPS(nJ)*			
Design	*Throughput (# median outputs/ clock)*	*Latency (#clock cycles)*	*Window size*	*Core area (um^2)*	*Power (nW)*	*Delay (ps)*	*Audio 1*	*Audio 2*	*Audio 3*	*Average*
Existing	1	w	5	59722.7	1402155	3157.9	51.16	47.41	72.69	57.08
			9	116579	3426665	4219	125.02	115.86	177.64	139.5
Proposed	1	w	5	28710	2076690	1998	75.77	70.22	107.66	84.55
			9	54272	3917452	3234	142.93	132.46	203.09	159.4

6. CONCLUSION

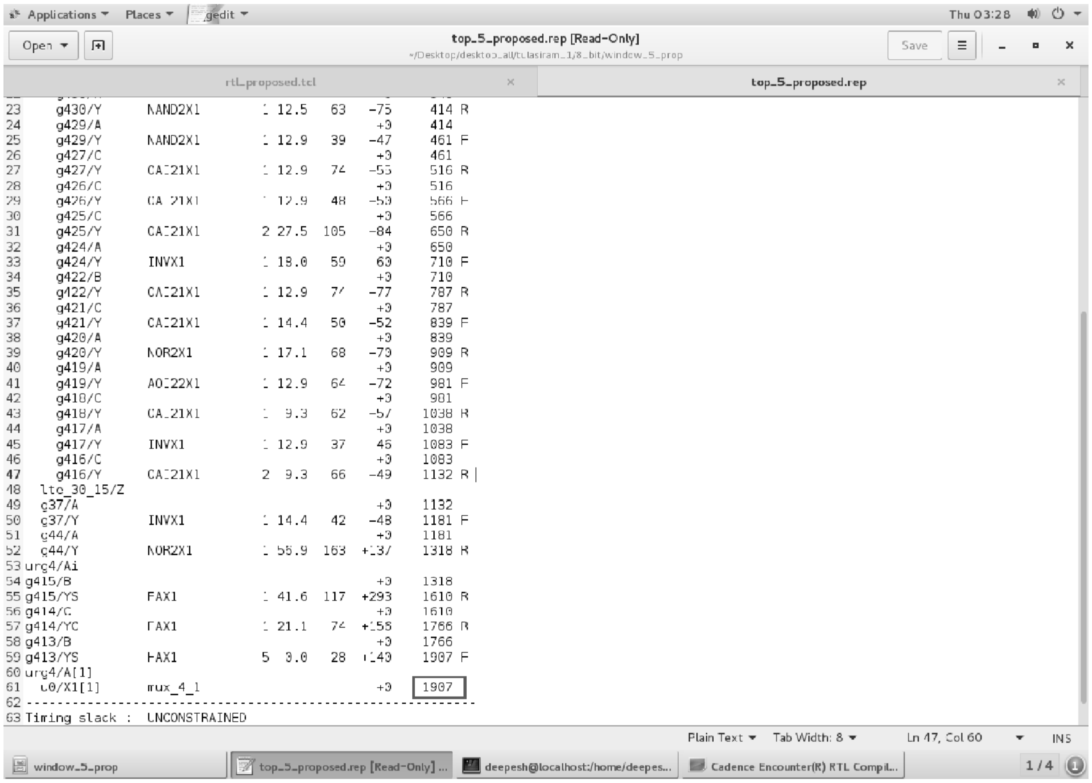
In this paper, one dimensional median filter architecture using CSLA method is presented, which benefits to reduce the area, power consumption and delay. BEC, which is utilize to increase the speed of addition operation. The CSLA adder obtained low power consumption and low area, when it operated in BEC instead of RCA. The following results are the main advantages in BEC such as low area, low power, less number of Full Adder (FA) structure, simple and high efficient in VLSI implementation.



Generated by: Encounter(R) RTL Compiler v12.10-p006_1 (Nov 8 2012)
Generated on: Nov 24 2016 04:10:22
Module: top_5_proposed
Technology library: osu018_stdcells
Operating conditions: typical (balanced_tree)
Wireload mode: enclosed

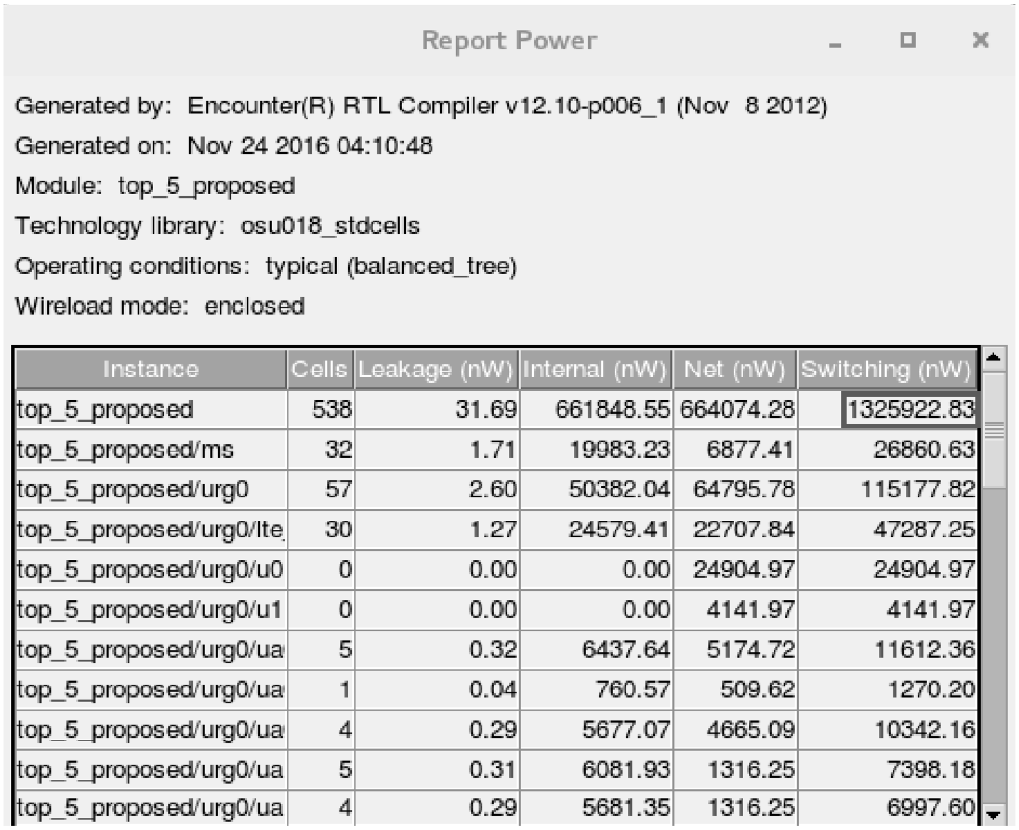
Instance	Cells	Cell Area	Net Area	Total Area	Wireload	WL Flag
top_5_proposed	538	17989.00	0.00	17989.00	<none>	(D)
top_5_proposed/ms	32	1120.00	0.00	1120.00	<none>	(D)
top_5_proposed/urg0	57	1365.00	0.00	1365.00	<none>	(D)
top_5_proposed/urg0/lte_	30	671.00	0.00	671.00	<none>	(D)
top_5_proposed/urg0/u0	0	0.00	0.00	0.00	<none>	(D)
top_5_proposed/urg0/u1	0	0.00	0.00	0.00	<none>	(D)
top_5_proposed/urg0/ua0	5	143.00	0.00	143.00	<none>	(D)
top_5_proposed/urg0/ua0.	1	16.00	0.00	16.00	<none>	(D)
top_5_proposed/urg0/ua0.	4	127.00	0.00	127.00	<none>	(D)
top_5_proposed/urg0/ua1	5	143.00	0.00	143.00	<none>	(D)
top_5_proposed/urg0/ua1.	4	127.00	0.00	127.00	<none>	(D)

Figure 6: Total area value for window 5 proposed



```
    g430/Y        NAND2X1        1 12.5   63   -75     414 R
    g429/A                                      +0     414
    g429/Y        NAND2X1        1 12.9   39   -47     461 F
    g427/C                                      +0     461
    g427/Y        OAI21X1        1 12.9   74   -55     516 R
    g426/C                                      +0     516
    g426/Y        OAI21X1        1 12.9   48   -50     566 F
    g425/C                                      +0     566
    g425/Y        OAI21X1        2 27.5  105   -84     650 R
    g424/A                                      +0     650
    g424/Y        INVX1          1 18.0   59    60     710 F
    g422/B                                      +0     710
    g422/Y        OAI21X1        1 12.9   74   -77     787 R
    g421/C                                      +0     787
    g421/Y        OAI21X1        1 14.4   50   -52     839 F
    g420/A                                      +0     839
    g420/Y        NOR2X1         1 17.1   68   -70     909 R
    g419/A                                      +0     909
    g419/Y        AOI22X1        1 12.9   64   -72     981 F
    g418/C                                      +0     981
    g418/Y        OAI21X1        1  9.3   62   -57    1038 R
    g417/A                                      +0    1038
    g417/Y        INVX1          1 12.9   37    45    1083 F
    g416/C                                      +0    1083
    g416/Y        OAI21X1        2  9.3   66   -49    1132 R
  lte_30_15/Z
  g37/A                                         +0    1132
  g37/Y           INVX1          1 14.4   42   -48    1181 F
  g44/A                                         +0    1181
  g44/Y           NOR2X1         1 55.9  163  +137    1318 R
urg4/Ai
g415/B                                          +0    1318
g415/YS           FAX1           1 41.6  117  +293    1610 R
g414/C                                          +0    1610
g414/YC           FAX1           1 21.1   74  +156    1766 R
g413/B                                          +0    1766
g413/YS           FAX1           5  3.0   28  +140    1907 F
urg4/A[1]
  u0/X1[1]        mux_4_1                       +0    1907
---------------------------------------------------------------
Timing slack :  UNCONSTRAINED
```

Figure 7: Total delay for window 5 proposed



Generated by: Encounter(R) RTL Compiler v12.10-p006_1 (Nov 8 2012)
Generated on: Nov 24 2016 04:10:48
Module: top_5_proposed
Technology library: osu018_stdcells
Operating conditions: typical (balanced_tree)
Wireload mode: enclosed

Instance	Cells	Leakage (nW)	Internal (nW)	Net (nW)	Switching (nW)
top_5_proposed	538	31.69	661848.55	664074.28	1325922.83
top_5_proposed/ms	32	1.71	19983.23	6877.41	26860.63
top_5_proposed/urg0	57	2.60	50382.04	64795.78	115177.82
top_5_proposed/urg0/lte	30	1.27	24579.41	22707.84	47287.25
top_5_proposed/urg0/u0	0	0.00	0.00	24904.97	24904.97
top_5_proposed/urg0/u1	0	0.00	0.00	4141.97	4141.97
top_5_proposed/urg0/ua	5	0.32	6437.64	5174.72	11612.36
top_5_proposed/urg0/ua	1	0.04	760.57	509.62	1270.20
top_5_proposed/urg0/ua	4	0.29	5677.07	4665.09	10342.16
top_5_proposed/urg0/ua	5	0.31	6081.93	1316.25	7398.18
top_5_proposed/urg0/ua	4	0.29	5681.35	1316.25	6997.60

Figure 8: Total power value for window 5 proposed

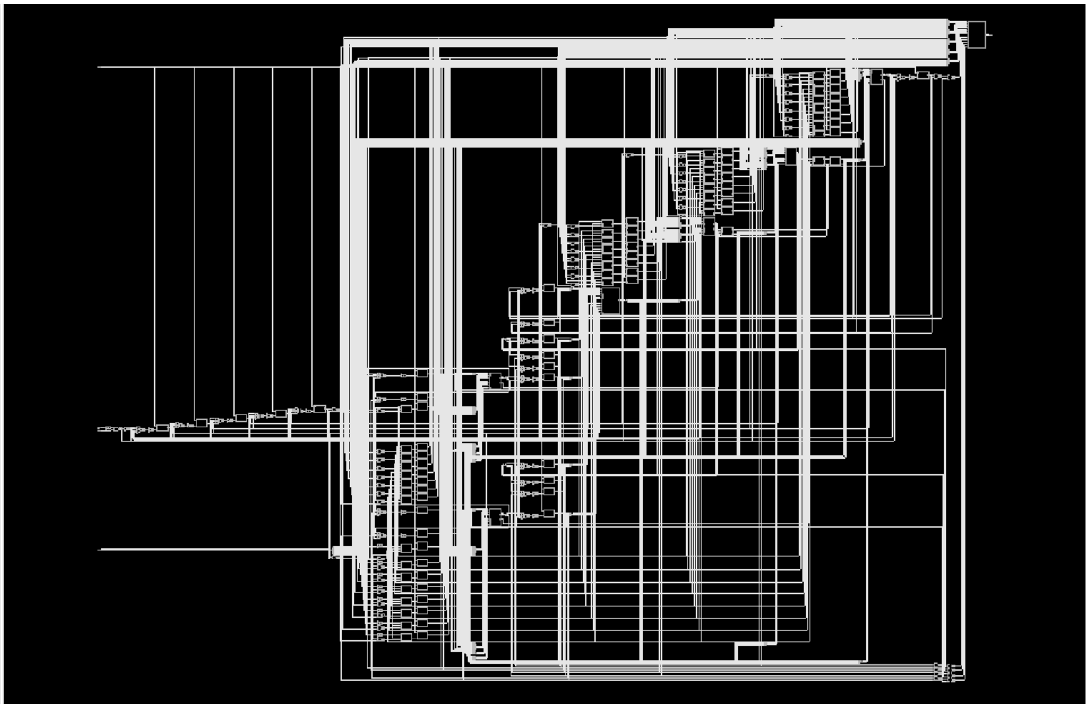


Figure 9: RTL schematic for window 5 proposed

REFERENCES

[1] Andreadis, Ioannis, and Gerasimos Louverdis., “Real-time adaptive image impulse noise suppression”, IEEE transactions on Instrumentation and Measurement, vol.53, no. 3, pp. 798-806, 2004.

[2] Vasicek, Zdenek, and Lukas Sekanina., “Novel hardware implementation of adaptive median filters”, Design and Diagnostics of Electronic Circuits and Systems, vol.5, no. 2, pp. 1-6, 2008.

[3] Teja, V. R., Ray, K. C., Chakrabarti, I. &Dhar, A.S.Janu., “High throughput VLSI architecture for one dimensional median filter”, In Signal Processing, Communications and Networking, vol.21, no. 6, pp. 339-344, 2008.

[4] Moshnyaga, V. G., & Hashimoto, K., “An efficient implementation of 1-D median filter”, IEEE International Midwest Symposium on Circuits and Systems, vol.12, no. 6, pp. 451-454, 2009.

[5] Prokin, Dragana, and Milan Prokin., “Low hardware complexity pipelined rank filter”, IEEE Transactions on Circuits and Systems II: Express Briefs, vol.57, no. 6, pp. 446-450, 2010.

[6] Fang, Q., Zhang, W., Pang, Z., Chen, D., & Wang, Z., “A proposed fast word level sequential scheme and parallel architecture for bit plane coding of EBCOT used in JPEG2000”, In Multimedia Technology (ICMT),vol.7, no. 2, pp. 1-4, 2010.

[7] Blad, A., & Gustafsson, O., “Bit-level optimized FIR filter architectures for high-speed decimation applications”, IEEE International Symposium on Circuits and Systems, vol.16, no. 12, pp. 1914-1917, 2008.

[8] Ramkumar, B., and Harish M. Kittur., “Low-power and area-efficient carry select adder”, IEEE transactions on very large scale integration (VLSI) systems, vol.20, no. 2, pp. 371-375, 2012.

[9] Anagha, U. P., &Pramod, P., “Power and area efficient carry select adder”, In Intelligent Computational Systems (RAICS), vol.10, no. 6, pp. 17-20, 2015.

[10] Sahu, R., &Subudhi, A. K., “ An area optimized Carry Select Adder”, In IEEE Power, Communication and Information Technology Conference (PCITC), vol.15, no. 5, pp. 589-594, 2015.

[11] Cadenas, J.O., Megson, G.M. and Sherratt, R.S., “Median filter architecture by accumulative parallel counters”, IEEE Transactions on Circuits and Systems II: Express Briefs, vol.62, no. 7, pp. 661-665, 2015.

[12] Chen, R.D., Chen, P.Y. and Yeh, C.H., “Design of an area-efficient one-dimensional median filter”, IEEE Transactions on Circuits and Systems II: Express Briefs, vol.60, no. 10, pp. 662-666, 2013.

[13] Mukherjee, M. and Maitra, M., “An efficient FPGA based de-noising architecture for removal of high density impulse noise in images”, IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), vol.12, no. 8, pp. 262-266, 2015.

[14] Mukherjee, M. and Maitra, M., 2015, “Reconfigurable architecture of adaptive median filter—An FPGA based approach for impulse noise suppression”, In Computer, Communication, Control and Information Technology (C3IT), vol.5, no. 9, pp. 1-6, 2015.

[15] Pereverzev, A., Prokofiev, Y. and Kaleev, D., “One-dimensional median filter with modular architecture hardware implementation”, In Internet Technologies and Applications (ITA), vol.18, no. 8, pp. 33-36, 2015.