HIGH SPEED AND POWER EFFICIENT FPGA IMPLEMENTATION OF IMPROVED FIRST ORDER MOMENT ALGORITHM OF CYCLIC CONVOLUTION IN VLSI SYSTEMS

Mrs. Anwar Jahan, Ms. Imrose Basha, Mr. Md Altaf ur Rahman

1,2 ,3 Assistant Professor,
1,2,3 Department of ECE,
1,2,3 Global Institute of Engineering & Technology, Moinabad, Rangareddy Dist., Telangana State.

ABSTRACT:

The fundamental computing substructure is composed simplest of a easy manipulate module, several round right-shift registers, and accumulator devices, requiring no multipliers and large memory. Comparisons are made in terms of region-postpone technology, region-time era and electricity consumption with current reminiscence-based architectures to illustrate the performance and effectiveness of the proposed systems. Using the same metrics, the contrast outcomes show a significant improvement of the proposed designs over the previous first-order second-based shape

Index Terms—Fast convolutions, First order moment,Cyclic convolution.

INTRODUCTION:

Cyclic convolution is a fundamental method in signal and image processing that can be formed from linear convolution, finite impulse response (FIR) filters, and other sinusoidal transforms. Numerous quick techniques and structures have been presented for various applicationsto enhance its computing performance.

The Fast Fourier transform (FFT), which can compute convolution with minimal computer cost, is a widely popular early approach. Even when using convolution of actual sequences, the FFT cannot always produce extremely precise computational results since it requires trigonometric functions and complex arithmetic. A high speed parallel architecture in for cyclic convolution as well as quick algorithms in and combining with multidimensional methods have been created. The polynomial transform offers another effective way for quick convolutions, especially when the convolution length is a power of two, to further reduce the processingcost. These methods, meanwhile, involve intricate mathematical modifications, and the most of them still have restrictions on the length of the convolution.

A high speed parallel architecture for cyclic convolution as well as quick methods in and combining with multidimensional techniques have been created. The polynomial transform offers another effective way for quick convolutions, especially when the convolution length is a power of two, to further reduce the processing cost. These methods, meanwhile, involve intricate mathematical modifications, and the most of them still have restrictions on the length of the convolution.

The multiplier-free solutions are created to achieve effective hardware realization, thus enhancing overall performance. The cyclic convolution in can be implemented with the advantages of low hardware cost, low input/output (I/O) cost, and high computation speed by using parallel adders and the common sub expression sharing technique. These adder-based architectures typically rely too heavily on certain convolution kernels, which restricts their potential uses. There are direct memory-based designs and distributed arithmetic (DA)-based designs because the convolution kernel can be preprocessed and put into read-only memory (ROM), which is possible because it is typically fixed.

In this proposed system novel parallel and multiplier-free structures for cyclic convolution based on the improved first-order moment algorithm. The proposed structures notonly implement convolution in parallel with much less additions and latency, but also are more suitable and flexible for hardware implementation with the decomposition scheme and simple preprocessing.

Moreover, due to much smaller ROM size relative to the memory-based designs, the proposed structures may have lower hardware cost and power consumption. Since there is no limitation on the convolution length and word width,they can achieve even better performance via appropriately being combined with the fast algorithms for long-length cyclic convolution.

The cyclic convolution between two length-sequences can be formulated as

$$y(k) = h(k) * x(k) = \sum_{n=0}^{N-1} h(n)x(k-n)_N,$$

$$k = 0, 1, \ldots, N-1,$$

where $(.)_N$ represents modulo operation.

Generally, the sequence {h(n)} is set as the fixed convolution kernel, and the other one {h(n)} is the input sequence whose value constantly changes at any time.

IMPROVED FAST ALGORITHM FORFIRST-ORDER MOMENTS

To express word width, like 8, 16 or more, if we re-express it as L=M*t, then each element {h(n)} in can be decomposed into M parts. As shown in Fig. 1,

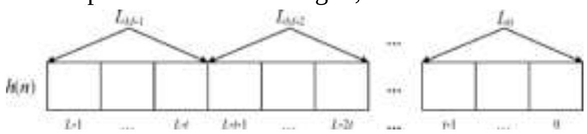


Fig.(1). The binary form h(n)of represented L by bits Where, L_{M-1} and L_0 respectively represents the more

b) Storing the original index n of $h_m(n)$ as the value of , where each consists of a control sequence

$$\{q_m(n'), n' = 0, 1, \ldots, N-1\}$$

c) Assigning the number of elements in S_i^m to $o_m(i)$for (i= 0,1,2,3….. 2^t-1),. Then the other control sequence $\{p_m(r)\}$,r = 0,1,2,3….. N+2^t-1)is assigned with a Boolean value "1" and $o_m(i)$ Boolean values "0" in Group . The number of "1"and "0" in $p_m(r)$ respectively corresponds to the convolution length N and the range of value 2^t.

significant bits and the less significant bits of h(n) , where each $h_m(n)$ can be group into a new sequence $\{h_m(n)\}$ for m= 0,1,2,3,……M-1.

Next, separately implementing the same statistics on each sequence $\{h_m(n)\}$, as we did on {h(n)} before, we can get 2^t subsets S_i^m = (i= 0,1,2,3….. 2^t-1), which are

$$S_i^m = \{n | h_m(n) = i, n \in \{0, 1, 2, \ldots, N-1\}\},$$
$$i = 0, 1, \ldots, 2^t - 1.$$

Based on above equation, another new sequence is definedas

$$a_i^m(k) = \begin{cases} \sum\limits_{n \in S_i^m} x(k-n)_N & \text{if } S_i^m \neq \Phi \\ 0 & \text{otherwise} \end{cases},$$
$$i = 0, 1, \ldots, 2^t - 1.$$

Via substituting h(n) for $h_m(n)$, and bringing the relevant

$$\begin{aligned} y_m(k) &= \sum_{i=0}^{2^t-1} \sum_{n \in S_i^m} i \cdot x(k-n)_N \\ &= \sum_{i=1}^{2^t-1} i \cdot a_i^m(k), k = 0, 1, \ldots, N-1. \end{aligned}$$

The final result y(k) can be derived by shifting and adding these sub-results, as shown below.

$$y(k) = y_0(k) + 2^t \cdot y_1(k) + \cdots + 2^{(M-1) \cdot t} \cdot y_{M-1}(k)$$

1. PROPOSED PARALLEL STRUCTURE FOR SUBPART

While computing process is going on, as the input sequence {x(n)} changes repeatedly, we cannot get $\{a_i^m\}$ on time. So the convolution kernel {h(n)} is given in advance, through dividing it into M subparts, and separately performing statistical analysis {h(n)} on for m=0,1,2,…..M-1, their corresponding statistical results compose the control signals for subsequent calculations.

To obtain the fixed statistical results following rules to befollowed:

a) Reordering the subsequence $\{h_m(n)$,n = 0,1,2,…….N-1} from small to large. The reordered sequence can be represented as

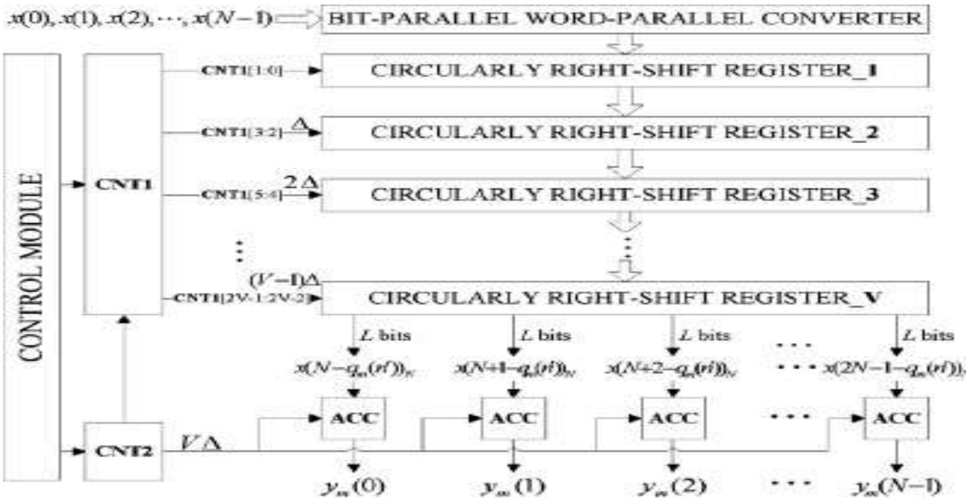


Fig.(2).The proposed parallel structure for subpart Y$_m$.The general schematic of the parallel structure.

which mainly consists of a control module, a group of circularly right-shift registers and N accumulation units. The outputs of accumulation units correspond to the sub- result It's worth noting that the steps of gathering relevant input items and accumulating them are combined in the first row, which not only further shortens the computation time, but also reduces the number of adders and registers.

2. TIME-EFFICIENT STRUCTURE

The below structure designed to reduce the computationtime for the cyclic convolution. It consists of basic computable substructures, Parallel adders, bit to word parallel converter. The working process is the composite detail inside the bit-parallel phrase parallel converter is dispatched into M groups of circularly proper shift registers at each cycle. Then, underneath the manage of signals output from their respective manipulate modules, accumulation modules composed of *M.N* accumulation devices, paintings independently till finishing all of the accumulations. As illustrated in below diagram, the expression <<*m.t* denotes transferring bits to the left side. After moving and adding all the Y_m with N parallel adders, the very last end result is obtained.

$$\{h_m(n'), n' = 0, 1, \ldots, N-1\}$$

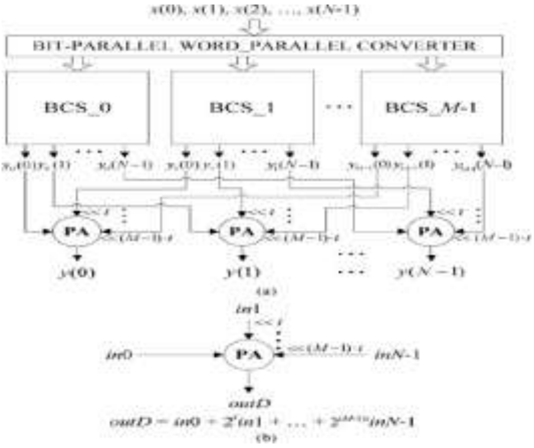

x(0), x(1), x(2), …, x(N-1)
BIT-PARALLEL WORD_PARALLEL CONVERTER
BCS_0
BCS_1
BCS_M-1
PA
y(0)
y(1)
y(N − 1)
(a)
in1
<< t
in0
<<(M−1)·t
inN-1
outD
outD = in0 + 2^t in1 + … + 2^(M-1)t inN-1
(b)

Fig.(3).The general schematic of the time-efficientstructure & the function of a parallel adder.

3. IMPLEMENTATION RESULTS:Schematic Diagram:

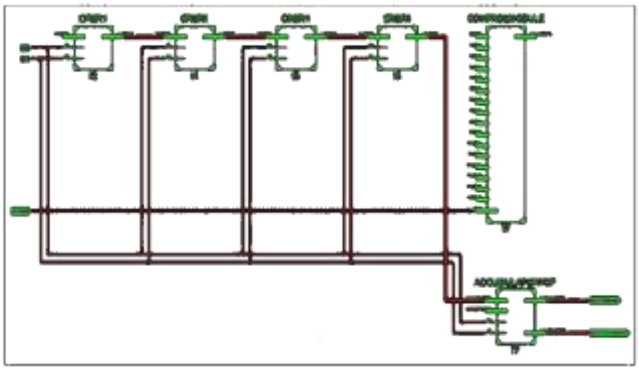
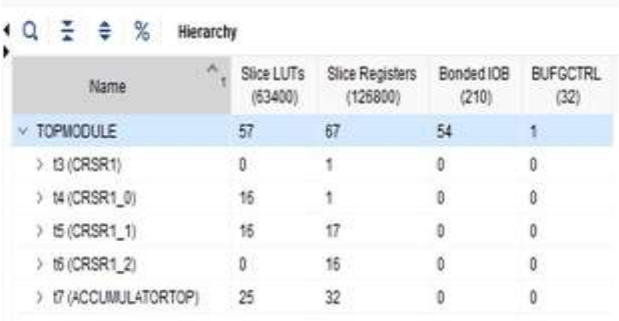


Fig.(4). Schematic diagram of Cyclic Convolution

Hierarchy of the Module:



Hierarchy

Name	Slice LUTs (63400)	Slice Registers (126800)	Bonded IOB (210)	BUFGCTRL (32)
TOPMODULE	57	67	54	1
t3 (CRSR1)	0	1	0	0
t4 (CRSR1_0)	16	1	0	0
t5 (CRSR1_1)	16	17	0	0
t6 (CRSR1_2)	0	16	0	0
t7 (ACCUMULATORTOP)	25	32	0	0

Fig.(5). Declaration Summary Report

4. SIMULATION RESULTS:

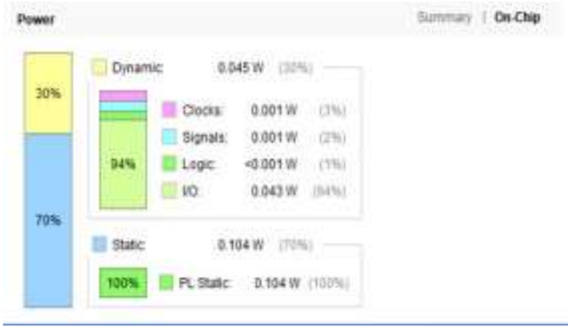
This simulation result is the most important that can show the result of the proposed scheme.The shown graph can tells us from which point the required convloution is taking place

Fig.(6). Simulation results

Before power optimization:

Dynamic & Static power:

The below diagram describes about the total dynamic & static power utilizing on the chip before implememtation.



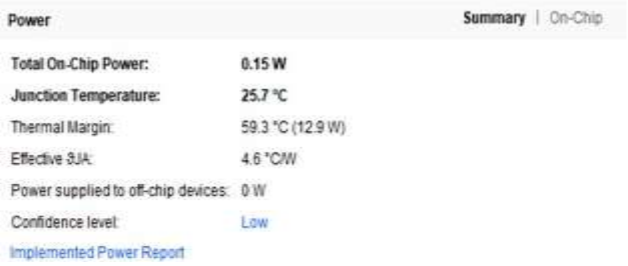
```
Power                                   Summary | On-Chip

30%   Dynamic       0.045 W  (30%)
      Clocks:       0.001 W  (3%)
      Signals:      0.001 W  (2%)
94%   Logic:       <0.001 W  (1%)
      I/O:          0.043 W  (94%)
70%   Static        0.104 W  (70%)
100%  PL Static:    0.104 W  (100%)
```

Fig.(7). Dynamic & Static power beforeimplementation

Hardware Utilizatation:

The below report shows the total hardware utilizationsummary on the chip before implementation



```
Power                                   Summary | On-Chip

Total On-Chip Power:                0.15 W
Junction Temperature:               25.7 °C
Thermal Margin:                     59.3 °C (12.9 W)
Effective ϑJA:                      4.6 °C/W
Power supplied to off-chip devices: 0 W
Confidence level:                   Low
Implemented Power Report
```

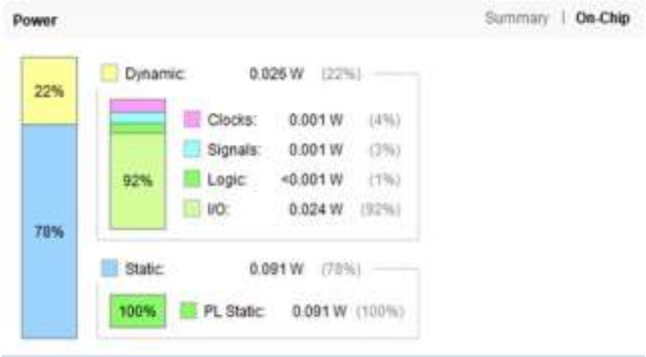
Fig.(8). Hardware utilization summary

After power optimization:

Dynamic & Static power:

Power is optimized up to half of the previous proposedstructure and the total time it is executed in 10ns earlierwhen we compare with the previous.

The below diagram describes about the total dynamic &static power utilizing on the chip after implememtation.



Power

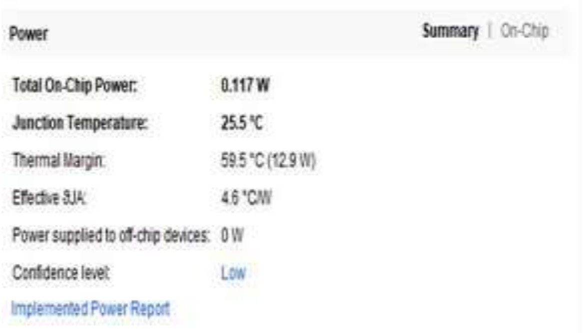
Summary | On-Chip

Dynamic:	0.026 W	(22%)
Clocks:	0.001 W	(4%)
Signals:	0.001 W	(3%)
Logic:	<0.001 W	(1%)
I/O:	0.024 W	(92%)
Static:	0.091 W	(78%)
PL Static:	0.091 W	(100%)

Fig.(9). Dynamic & Static power after implementation

Hardware Utilizatation:

The below report shows the total hardware utilizationsummary on the chip after implementation



Power

Summary | On-Chip

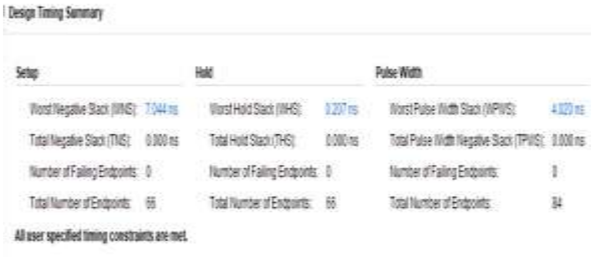
Total On-Chip Power:	0.117 W
Junction Temperature:	25.5 °C
Thermal Margin:	59.5 °C (12.9 W)
Effective ϑJA:	4.6 °C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Implemented Power Report

Fig.(10). Hardware utilization summary

Time Utilized:

The below report shows that total time utilized afterimplementation



Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	7.044 ns	Worst Hold Slack (WHS):	0.207 ns	Worst Pulse Width Slack (WPWS):	4.020 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	65	Total Number of Endpoints:	65	Total Number of Endpoints:	34

All user specified timing constraints are met.

Fig.(11). Total design Timing summary utilized.

Total Power Optimized:

The below bar diagram shows the total power optimizedafter implementation. This shows how much power is consumed while executing the process on the chip

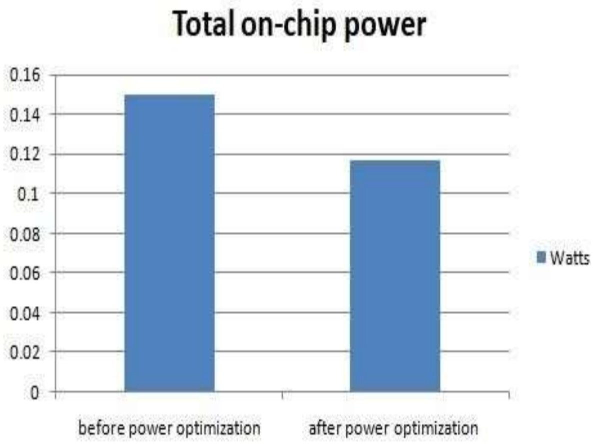


Fig.(12). Graphical representation of total poweroptimized

5. CONCLUSION

The proposed systems nonetheless own superiority in terms of region-time product. Similarly, in assessment with our preceding first-order moments-based totally shape, the simulation consequences tells that the proposedsystems gain brilliant enhancements, whilst the word width of convolution kernel is barely larger. Based on these blessings, they may be implemented to other computations, together with Discrete Fourier Transform (DFT), virtual clear out and correlation, via transforming all them into the shape of cyclic convolution

6. REFERENCES

***[1]** R. C. Agarwal and C. S. Burrus, "Number theoretic transforms to implement fast digital convolution," Proc.IEEE, vol. 63, no. 4, pp. 550–560, Apr. 1975.*

***[2]** R. C. Agarwal and C. S. Burrus, "Fast convolutionusing Fermat number transforms with applications to digital filtering," IEEE Trans. Acoust., Speech, SignalProcess., vol. ASSP-22, no. 2, pp. 87–99, Apr. 1974.*

***[3]** R. C. Agarwal and C. S. Burrus, "Fast one-dimensionaldigital convolution by multidimensional techniques," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP- 22, no. 2, pp. 1–10, Feb. 1974.*

***[4]** R. C. Agarwal and J. W. Cooley, "New algorithms fordigital convolution," IEEE Trans. Acoust., Speech, SignalProcess., vol. ASSP-25, no. 5, pp. 392–410, Oct. 1977.*

***[5]** J. Zhang and S. G. Li, "High speed parallel architecture for cyclic convolution based on FNT," inProc. IEEE Comput. Soc. Ann. Symp. VLSI, 2009, pp.199–204.*

***[6]** H. J. Nussbaumer, "Fast polynomial transform algorithms for digital convolution," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP–28, no. 2, pp. 205– 215, Apr. 1980.*