International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue14
November 2017

Implementation of DADDA Multiplier based Carry save Arithmetic (CSA)

Nagalaxmi Bairapangu & Dr. S. K. Sinha
[1]Asst. Professor Department Of ECE [2]Professor Department Of ECE
Kommuri Pratap Reddy Institute Of Technology (KPRIT), Telangana, Hyderabad, India
nagalaxmibaira@gmail.com

Abstract: *Equipment increasing speed has been demonstrated a to a great degree promising usage methodology for the digital signal processing (DSP) space. As opposed to embracing a solid application-particular integrated circuit configuration approach, in this concise, we introduce a novel quickening agent architecture involving adaptable computational units that help the execution of an expansive arrangement of operation layouts found in DSP kernels. We separate from past takes a shot at adaptable quickening agents by empowering calculations to be forcefully performed with carry-save (CS) arranged data. Propelled arithmetic plan ideas, i.e., recoding techniques, are used empowering CS enhancements to be performed in a bigger extension than in past methodologies. Broad test assessments demonstrate that the proposed quickening agent architecture conveys normal increases of up to 61.91% in zone postpone item and 54.43% in vitality utilization contrasted and the condition of-workmanship adaptable data ways.*

keywords: Digital signal processing (DSP), Data flow graph (DFG), Register Transfer logic (RTL), Template(T), Spurious power suppression technique (SPST), Carry save arithmetic (CSA).

I. INTRODUCTION

Present day implanted frameworks target top of the line application spaces requiring proficient executions of computationally serious digital signal processing (DSP) capacities. The joining of heterogeneity through particular equipment quickening agents enhances execution and decreases vitality utilization [1]. In spite of the fact that application-particular integrated circuits (ASICs) shape the perfect speeding up arrangement regarding execution and power, their firmness prompts expanded silicon multifaceted nature, as numerous instantiated ASICs are expected to quicken different kernels. Numerous specialists have proposed the utilization of space particular coarse-grained reconfigurable quickening agents keeping in mind the end goal to expand ASICs' adaptability without altogether trading off their execution. Superior adaptable data ways have been proposed to proficiently outline or affixed operations found in the underlying data-flow graph (DFG) of a piece. The layouts of complex

affixed operations are either separated specifically from the portion's DFG or determined in a predefined behavioral format library. Outline choices on the quickening agent's data way very effect its proficiency. Existing chips away at coarse-grained reconfigurable data ways for the most part abuse architecture level advancements, e.g., expanded direction level parallelism (ILP) . The space particular architecture age calculations of [5] and [9] fluctuate the sort and number of calculation units accomplishing a redid configuration structure. The adaptable architectures were proposed misusing ILP and operation binding. As of late forceful operation binding is embraced to empower the calculation of whole sub articulations utilizing numerous ALUs with heterogeneous arithmetic highlights. The previously mentioned of reconfigurable architectures prohibit arithmetic enhancements amid the building amalgamation and think of them as just at the inner circuit structure of primitive segments, e.g., adders, amid the logic blend. In any case, inquire about exercises have demonstrated that the arithmetic advancements at higher deliberation levels than the basic circuit one fundamentally affect on the data way execution. In [10], timing-driven advancements in view of carry save (CS) arithmetic were performed at the post-Register Transfer Level (RTL) plan organizes. In [11], regular sub articulation disposal in CS calculations is utilized to advance straight DSP circuits. Vermaet al. [12] created change techniques on the application's DFG to augment the utilization of CS arithmetic earlier the real data path amalgamation. The previously mentioned CS improvement approaches target unbendable data path, i.e., ASIC, executions. As of late, an adaptable architecture consolidating the ILP and pipelining techniques with the CS-mindful operation affixing has been proposed. In any case, the entire previously mentioned arrangements highlight an inborn restriction, i.e., CS advancement is limited to consolidating just increases/subtractions. A CS to parallel transformation is embedded before every operation that contrasts from expansion/subtraction, e.g., duplication, along these lines, designating different CS to double changes that vigorously debases execution because of tedious carry proliferations. In this concise, we propose a high performance building plan for the combination of adaptable equipment DSP quickening agents by consolidating advancement techniques from both the architecture and arithmetic levels of deliberation. We present a adaptable data path architecture that endeavors CS advanced formats of affixed operations. The proposed architecture involves adaptable computational units (FCUs), which empower the execution of a substantial arrangement of operation layouts found in DSP kernels. The proposed quickening agent architecture conveys normal picks up in territory defer item and in vitality utilization contrasted

with condition of-craftsmanship adaptable data ways, maintaining productivity toward scaled advancements.

II.CARRY-SAVE ARITHMETIC: MOTIVATIONAL OBSERVATIONS AND LIMITATIONS:

CS portrayal has been generally used to configuration quick arithmetic circuits because of its characteristic leeway of disposing of the expansive carry-engendering chains. CS arithmetic enhancements improve the application's DFG and uncover numerous info added substance operations (i.e., fastened options in the underlying DFG), which can be mapped onto CS compressors. The objective is to boost the range that a CS calculation is performed inside the DFG. Be that as it may, at whatever point an augmentation hub is interleaved in the DFG, either a CS to parallel change is summoned or the DFG is changed utilizing the distributive property. Consequently, the previously mentioned CS advancement approaches have constrained effect on DFGs commanded by increases, e.g., separating DSP applications. In this short, we handle the previously mentioned constraint by misusing the CS to altered Booth (MB) recoding each time duplication should be performed inside a CS-advanced datapath. In this way, the calculations all through the augmentations are handled utilizing CS arithmetic and the operations in the focused on data path are completed without utilizing any moderate carry-engender snake for CS to parallel change, along these lines enhancing execution.

III. PROPOSED FLEXIBLE ACCELERATOR:

The proposed adaptable quickening agent architecture is appeared in Fig 1. Each FCU works straightforwardly on CS operands and produces data in the same form1 for coordinate reuse of moderate outcomes. Each FCU works on 16-bit operands. Such a bit-length is satisfactory for the most DSP data ways, yet the building idea of the FCU can be direct adjusted for littler or bigger piece lengths. The quantity of FCUs is resolved at configuration time in view of the ILP and zone requirements forced by the creator. The CS to Bin module is a swell carry viper and believers the CS frame to the two's supplement one. The register bank comprises of scratch registers and is utilized for putting away middle of the road results and sharing operands among the FCUs. Diverse DSP kernels (i.e., distinctive register assignment and data correspondence designs per piece) can be mapped onto the proposed architecture utilizing post-RTL data way interconnection sharing techniques. The control unit drives the general architecture (i.e., communication flanked by the data port and the register bank, setup terminology of the FCUs and strength of mind signals for the multiplexers) in every clock cycle.

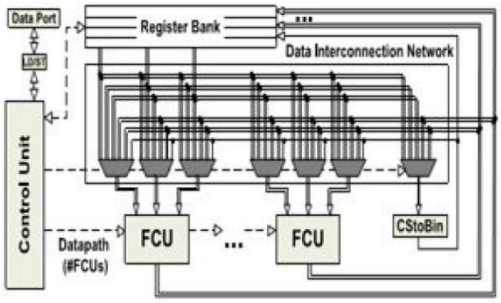


Fig.1: Flexible accelerator architecture.

A. Structure of the Proposed Flexible Computational Unit

The arrangement of the FCU (Fig. 2) has been planned to authorize better flexible process fastening in view of a library of operation formats. Each FCU can be designed to any of the T1– T5 operation layouts appeared in Fig. 3. The proposed FCU empowers intra format operation binding by intertwining the augmentations performed earlier/after the increase &performs any fractional operation layout of the accompanying complex operations:

$$W^* = A \times (X^* + Y^*) + K^* \quad (1)$$
$$W^* = A \times K^* + (X^* + Y^*) \quad (2)$$

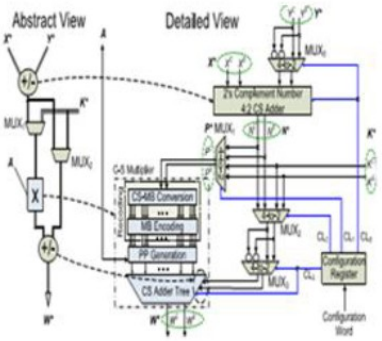


Fig.2: shows a segment of the internal structure of the FCU i.e. operational template of the data path specified in equation (1) and (2)

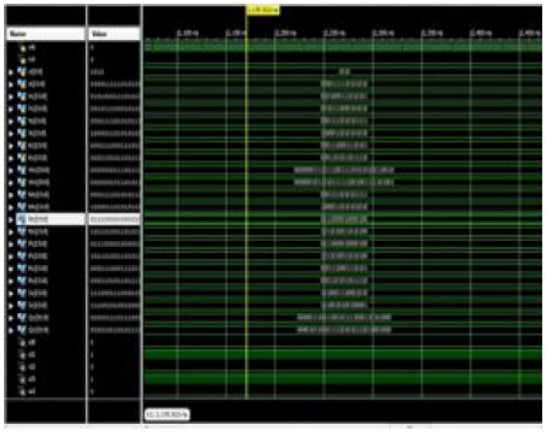
The accompanying connection holds for all CS data: X* = { XC, XS} = XC + XS. The operand A will be a two's supplement number. The option execution pathsin each FCU are indicated after appropriately setting the control signals of the multiplexers MUX1 and MUX2 (Fig. 2). The multiplexer MUX0 yields Y* when CL0 = 0 (i.e., X* + Y* is completed) or Y* when X* − Y* is required and CL0 = 1. The two's supplement 4:2 CS snake delivers the N* = X* + Y* when the information carry rises to 0 or the N* = X* − Y* when the information carry rises to 1. The MUX1 decides whether N* (1) or K* (2) is duplicated with A. The MUX2 determines if K* (1) or N* (2) is included with the augmentation item. The multiplexer MUX3 acknowledges the yield of MUX2 and its 1's supplement and yields the previous one when an expansion with the augmentation item is required (i.e., CL3 = 0) or the later one when a subtraction is completed (i.e., CL3 = 1). The 1-bit expert for the subtraction is included the CS viper tree. The multiplier contains a CS-to-MB module, which embraces an as of late proposed technique to recode the 17-bit P* in its individual MB digits with insignificant carry proliferation. The multiplier's item comprises of 17 bits. The multiplier incorporates a remuneration strategy for decreasing the blunder forced at the item's precision by the truncation technique. Notwithstanding, since all the FCU inputs comprise of 16 bits and gave that there are no overflows, the 16 most noteworthy bits of the 17-bit W* (i.e., the yield of the Carry-Save Adder

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue14
November 2017

(CSA) tree, and along these lines, of the FCU) are embedded in the suitable FCU when asked.

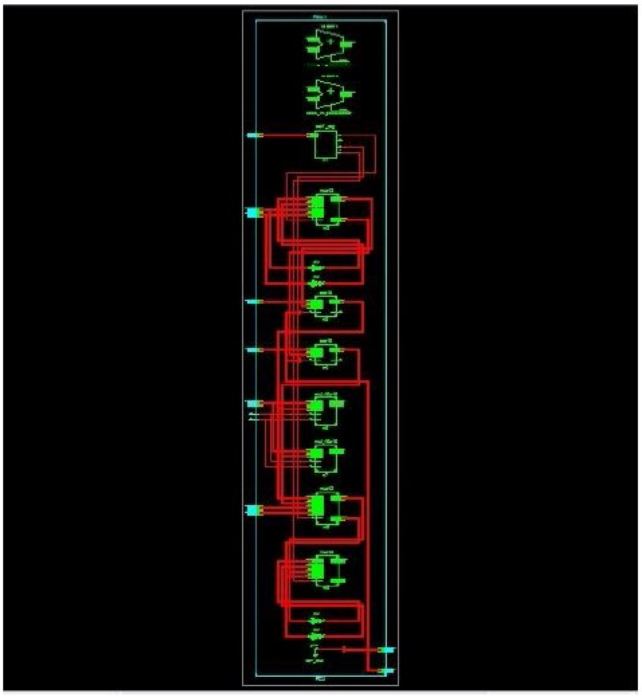
VI. RESULTS

Simulation Results:



Synthesis Results:

RTL Schematic:



Design Summary:

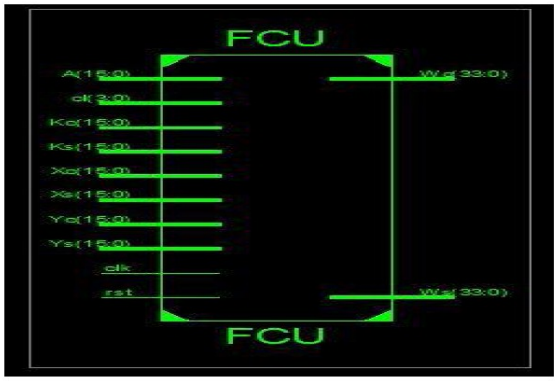
Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	824	8672	9%
Number of Slice Flip Flops	128	17344	0%
Number of 4 input LUTs	1472	17344	8%
Number of bonded IOBs	185	190	97%
Number of GCLKs	1	24	4%

Timing Report:

```
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc18
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc19
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc20
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc21
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc22
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc23
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc24
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc25
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc26
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc27
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc28
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc29
MUXCY:CI->O           1   0.051   0.000  Madd_Ws_addsub0000_cyc30
XORCY:CI->O           1   0.699   0.357  Madd_Ws_addsub0000_xor<3
OBUF:I->O             3.169          Ws_31_OBUF (Ws<31>)
----------------------------------------
Total                    18.934ns (12.483ns logic, 6.451ns route)
                                  (65.9% logic, 34.1% route)
```

Technology schematic

Top schematic



VII. CONCLUSION

In this to the point, we introduced a elastic accelerator structural design that exploits the integration of CS arithmetic optimizations to permit fast chaining of additive and multiplicative operations. The proposed flexible accelerator architecture is able to function on both conventional two's complement and CS formatted data operands, thus enabling high degrees of computational density to be achieved. Theoretical and experimental analyses have shown that the proposed solution forms an efficient design tradeoff point delivering optimized latency/area and energy implementations.

REFERENCES

[1] T. Kim and J. Um, "A practical approach to the synthesis of arithmetic circuits using carrysave-adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 19, no. 5, pp. 615– 624, May 2000.

[2] B. Mei, S. Vernalde, D. Verkest, H. D. Man, and R. Lauwereins, "ADRES: An architecture with firmly coupled VLIW processor and coarse-grained reconfigurable matrix," in Proc. thirteenth Int. Conf. Field Program. Rationale Appl., vol. 2778. 2003, pp. 61– 70.

[3] T. Kim and J. Um, "A practical approach to the synthesis of arithmetic circuits using carry-saveadders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 19, no. 5, pp. 615–624, May 2000.

[4] A. Hosangadi, F. Fallah, and R. Kastner, "Optimizing high speed arithmetic circuits using threeterm extraction," in Proc. Design, Autom. Test Eur. (DATE), vol. 1. Mar. 2006, pp. 1–6.

[5] A. K. Verma, P. Brisk, and P. Ienne, "Data-flow transformations to maximize the use of carry-save representation in arithmetic circuits," IEEE Trans.Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 10, pp. 1761–1774, Oct. 2008.

[6] A. K. Verma, P. Brisk, and P. Ienne, "Dataflow transformations to maximize the use of carry-save representation in arithmetic circuits," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 10, pp. 1761–1774, Oct. 2008.

[7] S. Xydis, G. Economakos, D. Soudris, and K. Pekmestzi, "High performance and area efficient flexible DSP datapath synthesis," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 3,pp. 429–442, Mar. 2011.

[8] P. M. Heysters, G. J. M. Smit, and E. Molenkamp, "A flexible and energy-efficient coarse-grained reconfigurable architecture

for mobile systems," J. Supercomput., vol. 26, no. 3, pp. 283–308, 2003.

[9] A. Hosangadi, F. Fallah, and R. Kastner, "Optimizing high speed arithmetic circuits using three-term extraction," in Proc. Design, Autom. Test Eur. (DATE), vol. 1. Mar. 2006, pp. 1–6.

[10] K. Compton and S. Hauck, "Automatic design of reconfigurable domainspecific flexible cores," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 16, no. 5, pp. 493–503, May 2008.