ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

Hardware reduction of DSP kernel Data Path using Carry Save Arithmetic operation in Fused Add Multiple unit

G.Rajkumar[1], Vaidyanathan[2]

Assistant Professor, Department of Electronics and Communication Engineering, SKP Engineering College

Tiruvannamalai, Tamilnadu, India[1, 2]

ABSTRACT: A high-performance data-path to implement DSP kernels is introduced in this paper. The proposed architecture of data-path is realized by FAMA (Fused Add-Multiply-Add (FAMA) unit). We differentiate from previous works on flexible computational unit using addition and subtraction process it takes that extra hardware. A mapping methodology, for datapath composed with the proposed flexible unit.it exploits the features of proposed units and enables fast computations. Comparing them with the datapath which use existing schemes, the proposed technique yields considerable reductions in terms of area and power consumption in datapath.

KEYWORDS: Arithmetic optimizations, carry-save (CS) form, datapath synthesis, flexible accelerator, operation chaining, FAMA unit.

I. INTRODUCTION

Digital Signal Processing (DSP) and multimedia applications, usually spend most of their time executing a small number of code segments with well-defined characteristics are called kernel. To accelerate the execution of such kernels, various high-performance data-paths have been proposed. The incorporation of heterogeneity through specialized hardware accelerators improves performance and reduces energy consumption. Although the specific application integrated circuit (ASICs) form the ideal acceleration solution in terms of performance, area and power, their inflexibility to increased silicon complicity ,as multiple instantiated ASICs are needed to accelerate various kernels.

The High-performance flexible datapath [8], [9], [10], [11], [12] been proposed to efficiently map primitive or chained operations found in the initial data-flow graph (DFG) of a kernel. The templates of complex chained operations are either extracted directly from the kernel's DFG or specified in a predefined behavioral template library. Design decisions on the accelerator's datapath highly impact its efficiency. In this paper, motivated by the above observations, we propose flexible and high performance architecture able to execute efficiently large set DSP operation templates. High performance is achieved by exploiting the CS arithmetic format which eliminates the time consuming carry-propagation Flexibility is enabled based on small scale configurability, the proposed architecture inserting by a limited number of multiplexors. The reconfigurable architectures exclude arithmetic optimizations during the architectural synthesis and consider them only at the internal circuit structure of primitive components, e.g., adders, during the logic synthesis.

In timing-driven optimizations based on carry-save (CS) arithmetic were performed at the post- register Transfer Level (RTL) design stage CS optimization is bounded to merging only additions/subtractions. In common sub expression elimination in CS computations is used to optimize linear DSP circuits. The proposed architecture comprises Fused Add- Multiply-Add (FAMA) units which enable the execution of a large set of operation templates found in DSP kernels.

In this brief, we propose a high-performance architectural scheme for the synthesis of flexible hardware DSP accelerators by combining optimization techniques from both the architecture and arithmetic levels of abstraction. We introduce a flexible datapath architecture that exploits CS optimized templates of chained operations. The proposed architecture comprises Fused Add-Multiply-Add (FAMA) unit, which enable the execution of a large set of operation

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

templates found in DSP kernels. The proposed accelerator architecture delivers 1.10% in area-delay and 1% in energy consumption compared to datapath.

Carry save arithmetic

Carry save arithmetic has been used to design fast arithmetic circuits due to its inherent advantage of eliminating the large carry-propagation chains. CS arithmetic optimizations the application's DFG and reveal multiple input additive operations (i.e., chained additions in the initial DFG), which can be mapped onto CS compressors. The goal is to maximize the range that a CS computation is performed within the DFG. However, whenever a multiplication node is interleaved in the DFG, either a CS to binary conversion is invoked or the DFG is transformed using the distributive property.

Thus, the aforementioned CS optimization approaches have limited impact on DFGs dominated by multiplications, e.g., filtering DSP applications. In multiplication process is used to modified booth concept to reduce the partial product in multiplier to achieve area and power conceptions. The multiplications are processed using CS arithmetic and the operations in the targeted datapath are carried out without using any intermediate carry-propagate adder for CS to binary conversion, thus improving performance.

II. LITERATURE SURVEY

Modified Booth techniques.

In 2014 APRIL Kostas Tsoumanis Sotiris Xydis, Constantinos Efstathiou, Nikos Moschopoulos, and Kiamal Pekmestzi[1] are says that modified booth technique . In Mb recoding techniques multiplication process to reduces by half the number of partial products in multiplication comparing to any other radix-4 representation.

Carry save arithmetic techniques.

In Oct 2008 Ajay K. Verma, Philip Brisk, and Paolo Ienne [2] are says that carry save arithmetic technique The carry save arithmetic operation to reduce delay and doing fast addition process.

FCU (Flexible computational unit)

In April 2015 Kostas Tsoumanis, Sotiris Xydis, Georgios Zervakis, and Kiamal Pekmestzi [5] are says to FCU unit. In this unit is used to operate add/sub and multiplication are doing to achieve quickly.

EXISTING FLEXIBLE ACCELERATOR

Flexible computational units (FCUs)

The existing flexible accelerator architecture is shown in Fig.1. Each FCU operates directly on CS operands and produces data in the same form for direct reuse of intermediate results. Each FCU operates on 16-bit operands. Such a bit-length is adequate for the most DSP data-paths, but the architectural concept of the FCU can be straightforwardly adapted for smaller or larger bit-lengths. The number of FCUs is determined at design time based on the ILP and area constraints imposed by the designer. Which enable the execution of a large set of operation templates found in DSP kernels.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

EXISTING THE FLEXIBLE ACCLELERATOR ARCHITECTURE

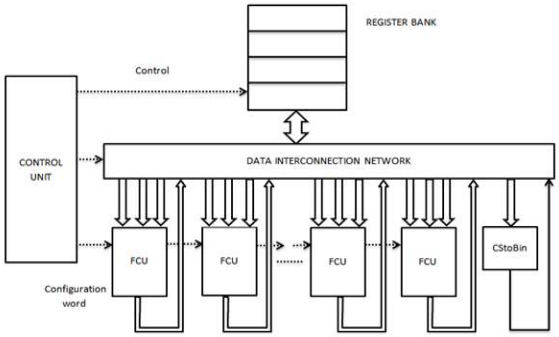


Fig 1.Existing Abstract form of Flexible datapath

STRUCTURE OF THE FLEXIBLE COMPUTATIONAL UNIT

The structure of the FCU has been designed to enable high-performance flexible operation chaining based on a library of operation templates.

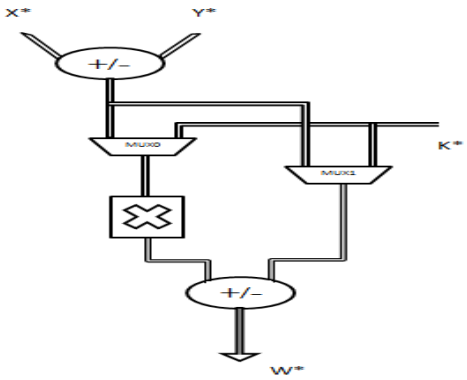


Fig 2 Abstract view of FCU

Each FCU can be configured to operation templates shown in Fig. 2. The proposed FCU enables intra template operation chaining by fusing the additions. The FCU is able to operate on either CS or two's complement formatted operands, since a CS operand comprises two 2's complement binary numbers. FCU template library performed before/after the multiplication and performs partial operation template of the following complex operations.

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

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

The following relation holds for all CS data: $X^* = \{X^C, X^S\} = X^C + X^S$. The operand A is a two's complement number.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

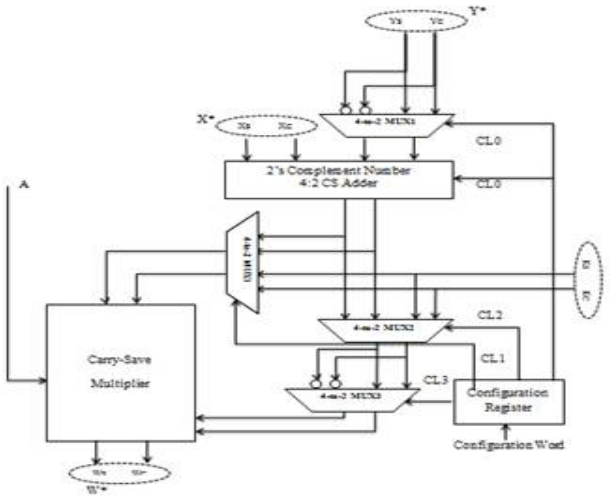


Fig 3. FCU

The alternative execution paths in each FCU are specified after properly setting the control signals of the multiplexers MUX1 and MUX2 (Fig. 3).

The multiplexer MUX0 outputs Y * when CL0 = 0 (i.e., X* + Y * is carried out) or Y * when X* − Y* is required and CL0 = 1. The two's complement 4:2 CS adder produces the N* = X* +Y * when the input carry equals 0 or the N* = X* −Y * when the input carry equals 1. The MUX1 determines if N* (1) or K* (2) is multiplied with A. The MUX2 specifies if K* (1) or N* (2) is added with the multiplication product. The multiplexer MUX3 accepts the output of MUX2 and its 1's complement and outputs the former one when an addition with the multiplication product is required (i.e., CL3 = 0) or the later one when a subtraction is carried out (i.e., CL3 = 1). The 1-bit ace for the subtraction is added in the CS adder tree.

The multiplier comprises a CS-to-MB module, which adopts a recently proposed technique to recode the 17-bit *P** in its respective MB digits with minimal carry propagation. The multiplier's product consists of 17 bits. The multiplier includes a compensation method for reducing the error imposed at the product's accuracy by the truncation technique.

However, since all the FCU inputs consist of 16 bits and provided that there are no overflows, the 16 most significant bits of the 17-bit W* (i.e., the output of the Carry-Save Adder (CSA) tree, and thus, of the FCU) are inserted in the appropriate FCU when requested.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

THE FLEXIBLE ACCLELERATOR ARCHITECTURE

The proposed flexible accelerator architecture is shown in Figure 1. It consist of FAMA, register bank, control unit, CStoBin module.

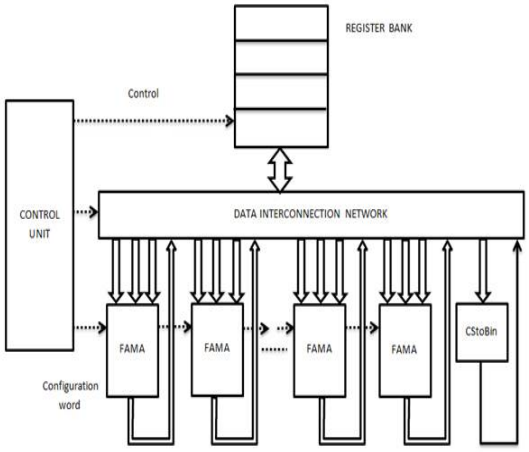


Fig 4 Abstract form of flexible datapath

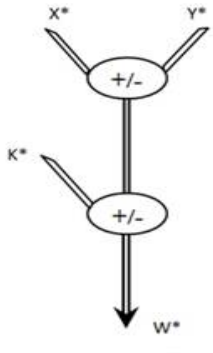


Fig 5 Abstract view of FAMA

Structure of the Fused Add-Multiply-Add (FAMA) unit

The FAMA architecture are shown in Fig 6. The FAMA consist of two mux, two 4:2 CS adder, Carry Save Multiplier, one configuration word.

FAMA operates can directly on Carry save operands and produces data in the same form 1 for direct reuse of intermediate results.

Each FAMA operates on 16-bit operands. Such a bit-length is adequate for the most DSP datapath, but the architectural concept of the FAMA can be straightforwardly adapted for smaller or larger bit-lengths.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

The number of FAMAs is determined at design time based on the ILP and area constraints imposed by the designer. The CStoBin module is a carry propagate adder and converts the CS form to the two's complement one.

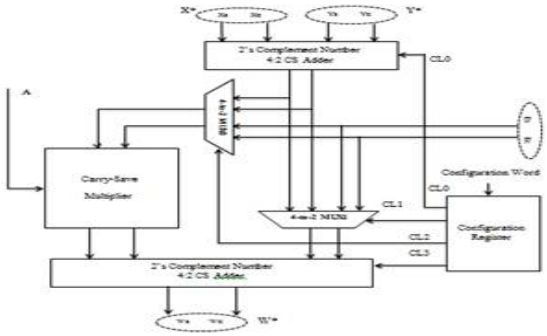


Fig 6 Detailed view of FAMA

The register bank consists of scratch registers and is used for storing intermediate results and sharing operands among the FAMAs. Different DSP kernels (i.e., different register allocation and data communication patterns per kernel) can be mapped onto the proposed architecture using post-RTL datapath interconnection sharing techniques. The control unit drives the control in overall architecture (i.e., communication between the data port and the register bank, configuration words of the FAMAs and selection signals for the multiplexers.

The structure of the FAMA has been designed to enable high-performance flexible operation changing based on a library of the operation templates. In each FAMA can be configured to any of the operation templates T1-T5. The proposed FAMA enables intra template operation chaining by fusing the additions and subtraction performed before /after the multiplication.

$$Z^* = N^* \times A + K^* \quad (4.a)$$
$$N^* = X^* + Y^* \quad (4.b)$$
$$X^* = \{X^c, X^s\} = X^c + X^s \quad (4.c)$$
$$Y^* = \{Y^c, Y^s\} = Y^c + Y^s \quad (4.d)$$

The star * is denoted the redundant representation
composed of two numbers both in 2s complement form (defined as 2's complement CS form/representation).
Each of the Z^*, N^*, Y^*, K^* is formed like X^*. The quantities Z^C, Z^S,N^C, N^S, X^C, X^S, Y^C, Y^S, K^C, K^S and A are all 2's complement conventional binary numbers.

The primary inputs of the units X*,Y*,K* can be either in 2's complement conventional binary or in 2's complement CS form.

The overall architecture of flexible FAMA unit is depicted Fig 3. It operates on data of 16-bit word length, since 16-bits enable sufficient accuracy for the majority of
modern DSP applications [6]. It consists of 1) a 2's complement numbers 4:2 CS adder (CSA), for the addition of the input data ($X¤$, $Y¤$), 2) the recoding scheme, 3) a tree based adder for the addition of the partial products, 4) the final CS accumulation unit implemented by a 4:2 CSA and 5) a configuration register.

The upper 4:2 CSA computes the N*=X* +Y* which can be used either in the pre- or the post multiplication addition. Input A has to be in 2's complement binary numeric representation. At next, the 2's complement CS formed data, N* or K*, are driven to the recoding unit. The recoding unit performs the conversion from CS format to SD format and subsequently the conversion of each signed digit to the Sign- Magnitude (SM) digit representation. The recoding unit is presented. For now, we support signed digits, Dj, ranging between [-1, 1], but this scheme can easily be extended (i.e in a Modified's Booth) for even more efficient implementations of the multiplier.

The Sign-Magnitude (SM) [7] digits (sgn|Dj|, $|D_j|$)

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

are passed through the partial product generation component together with the input *A*. The Signed Partial Products (SPP_js) are generated based on the following logic function.

$$SPP_j = (A \oplus sgn(D_j)).|D_j|.2^j \qquad (5)$$

Next, we decide to use a tree based multiplier trading with a more canonical multiplication scheme, because tree multipliers deliver high performance gains. For 16-bit data, we need 3 levels of 4:2 CS adders. In case of Modified Booth encoding 2 levels of 4:2 adders are sufficient. The main principles can be adapted in other multiplication schemes, i.e array multipliers, in a straightforward manner. Because the paper focuses on the overall architecture and the mapping opportunities it reveals, we neglect further details about the core multiplier's circuit. The final CS accumulation unit is a 4:2 CSA with two fixed inputs (the multipliers output bits) and two configurable inputs. All the inputs of the accumulation unit are in 2's complement CS format. The configured inputs are either the N* derived from the initial 4:2 CSA or an independent CS input number (K*).

The configuration register controls the operation mode of the unit and the signs of the input numbers, by driving with proper control logic bits (CL_i) the multiplexors and the sign selection inside each 4:2 CSA module. It is loaded in a cycle by cycle basis.

The FAMA's template library is illustrated, considering primary inputs in the 2's complement CS form. FAMA delivers high speed computations exploiting the Carry-Save arithmetic operation without surcharging the data path's controller due to its small configuration word.

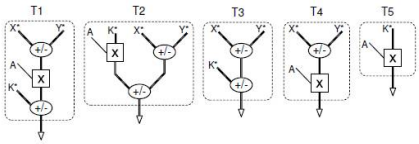


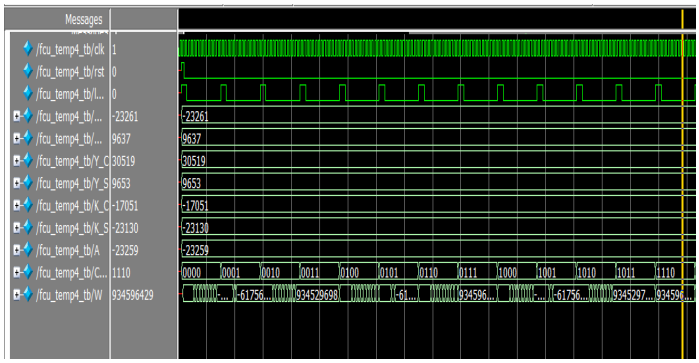
Fig 4. FAMA's Template Library.

In order to efficiently map DSP kernels onto the proposed FCU-based accelerator, the semiautomatic synthesis methodology presented in [7] has been adapted. At first, a CS-aware transformation is performed onto the original DFG, merging nodes of multiple chained additions/subtractions to 4:2 compressors.

The scheduled FAMA operations are bound onto FAMA instances and proper configuration bits are generated. After completing register allocation, a FSM is generated in order to implement the control unit of the overall architecture.

EXPERIMENTAL RESULT

Simulation Result of Proposed FCU



ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

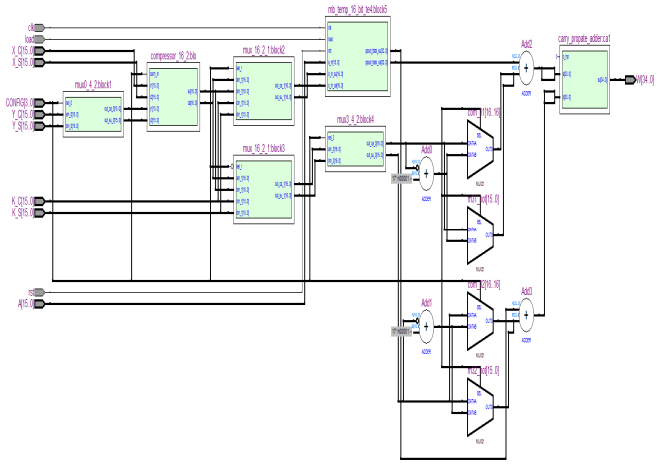


Fig 5. RTL view of Proposed FCU

Table 1 Comparison Result of area, power, time in FCU and FAMA in Quartus II

EDA tools	**EXISTING FCU**		
	Area	***Power***	***time***
QuarusII(9.0) cyclone II-family	1,125(logic element out of 33567)	128.29	6.701ns

EDA tools	**PROPOSED FCU**		
	Area	***Power***	***time***
QuarusII(9.0) cyclone II-family	1,104(logic element out of 33567)	128.09	6.577ns

V. CONCLUSION

The proposed a flexible accelerator architecture that exploits the incorporation of CS arithmetic optimizations to enable fast chaining of additive and multiplicative operations.

We differentiate from previous works on flexible computational unit using addition and subtraction it takes in extra hardware. Proposed Architecture forms in FAMA an efficient design of DSP Acceleration and improving the area, energy consumption and reduce the hardware. It exploits the incorporated features of the proposed units and enables fast computations, high operation densities and advanced data reusability

REFERENCES

1. Kostas Tsoumanis, Sotiris Xydis, Constantinos Efstathiou, Nikos Moschopoulos, and Kiamal Pekmestzi “Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Operator”, IEEE transactions on circuits and system in vol. 61, no. 4, April 2014 1133

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

2. Ajay K. Verma, Philip Brisk, and Paolo Ienne " Data-Flow Transformations to Maximize the Use of Carry-Save Representation in Arithmetic Circuits", IEEE transactions on computer-aided design of integrated circuits and systems, vol. 27, no. 10, October 2008.
3. Sotiris Xydis, Isidoros Sideris, George Economakos and Kiamal Pekmestzi "A Flexible Architecture For Dsp Applications Combining High Performance Arithmetic With Small Scale Configurability".
4.16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 25-29, 2008, copyright by EURASIP
5. Kostas Tsoumanis, Sotirios Xydis, Georgios Zervakis, and Kiamal Pekmestzi "Flexible DSP Accelerator Architecture Exploiting Carry-Save Arithmetic ", IEEE Transactions On Very Large Scale Integration (Vlsi) Systems April 2015.
6. G. Constantinides, P. Cheung, and W. Luk, *SynthesisAnd Optimization Of DSP Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 2004.
7. K. Hwang, "Computer Arithmetic," in *John Wiley andSons*, 1979.
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. M. D. Galanis, G. Theodoridis, S. Tragoudas, and C. E. Goutis, "A high-performance data path for synthesizing DSP kernels," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 6, pp. 1154–1162, Jun. 2006.
10. S. Xydis, G. Economakos, and K. Pekmestzi, "Designing coarse-grain reconfigurable architectures by inlining flexibility into custom arithmetic data-paths," *Integr., VLSI J.*, vol. 42, no. 4, pp. 486–503, Sep. 2009.
11. 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.
12. R. Kastner, A. Kaplan, S. O. Memik, and E. Bozorgzadeh, "Instruction generation for hybrid reconfigurable systems," *ACM Trans. Design Autom. Electron. Syst.*, vol. 7, no. 4, pp. 605–627, Oct. 2002.