International Journal of Engineering Research in Electronic and Communication Engineering (IJERECE) Vol 3, Issue 4, April 2016

# Design of Flexible Accelerator Based on Recoding Techniques

[1] Sambhu P.G, [2] Sreejesh Kumar R
[1] PG Student [VLSI and ES] [2] Assistant professor,
Department of ECE, TKM Institute of Technology, Kollam
[1]sambhu7799@gmail.com, [2] sjkumar.pillai@gmail.com

***Abstract*— DSP accelerators are hardware modules attached to a processor core externally to enhance the performance and functionality of computationally intensive DSP functions. Domain specific hardware architectures forms ideal acceleration in terms of performance and power, but their inflexible data paths lead to increased silicon complexity. In flexible DSP accelerator functional computational unit (FCU) are incorporated to improve performance, reduce energy consumption and to provide flexible data paths. The architecture exploits carry save (CS) arithmetic to enable fast chaining of additive and multiplicative operations. However, the carry save optimization approaches have limited impact on data flow graph (DFG) dominated by multiplications. These limitations are tackled by exploiting CS to modified booth recoding, which consists of three algorithms, signed-bit Full Adders (FAs) and Half Adders (HAs) as building blocks. To validate the design, code can be developed using VHDL and synthesis & simulation will be done in Xilinx ISE Suite 13.2 & Model Sim SE 6.3f respectively.**

***Index Terms*—carry Save (CS), Flexible Computational Unit (FCU), and Flexible data path**

## I. INTRODUCTION

In digital signal processing, information signal is used for mathematical manipulation of an. Over the past few years portable multimedia and DSP systems had become more popular, which requires more enhanced processing ability, very low power consumption, and have short design cycle. DSPs are used in accelerators from mid-1980's. The word "accelerator" indicates that the DSP can "accelerates" the performance of the computer itself, which of course is already used for processing digital signals. DSP accelerators run their own plug-ins, specially written, which only function with that specific hardware. Accelerator uses include diagnostics, machine protection and feed forward/feedback control.

Multimedia as well as DSP accelerators are very highly multiplication intensive so that the performance and power consumption of these systems are dominated by multipliers. Multipliers are key components of many high performance systems. FIR filters, processors, digital signal processors etc... are some of them. Multiplier is generally the slowest element in the system, a systems performance is generally determined by the performance of the multiplier. Furthermore, it is generally the most area consuming element. The speed of multiply operation has got great importance in general purpose processors as well as in digital signal processing.

In this paper a sum to modified booth recoding technique which will be optimizing both the architectural and arithmetic level of flexible computational unit (FCU). To achieve high performance, data path is implemented by flexible unit. The flexibility is achieved by using template, a group of chained units. Template generation is necessary for architecture generation as well for its compilation. The FCUs flexible datapath that will be exploiting carry save arithmetic, which can execute large set of templates.

## II. RELATED WORKS

Generally Digital signal processing (DSP) and multimedia applications spend most of their time in executing a small number of code segments with defined characteristics, called kernels. In order to efficiently map the operations found in the initial data flow graph (DFG) of a kernel a wide variety of architectures are available. It is possible to extract the complex chained operations directly from the kernel's DFG [7]. The design decisions on the accelerator's datapath will highly impact its efficiency. The existing works on coarse grained reconfigurable datapath are explained in various papers. A domain specific architecture generation algorithm, which vary the type and number of computation units achieving a customized design structure [6] and [3]. Flexible architectures were then proposed, which exploits instruction-level parallelism (ILP) and operation chaining [5]. An aggressive operation chaining is adopted to enable the computation of entire sub expressions using multiple ALUs [4]. The arithmetic optimizations are

excluded in the aforementioned reconfigurable architectures and it is considered only at the internal circuit structures of primitive components, during the logic synthesis eg. Adders [2]. There will be a significant impact on the datapath and its performance, when the arithmetic optimizations are done at the higher abstraction levels. Based on Carry-Save (CS) arithmetic the timing-driven optimizations were performed at the post-Register Transfer Level (RTL) design stage is presented in various papers. For optimizing the linear DSP circuits, it is possible to use common sub expression elimination in CS computations. By developing the transformation techniques on the applications DFG, we can maximize the use of CS arithmetic. The above stated CS optimization approaches target inflexible data path, i.e., ASIC, implementations. A flexible architecture that based on ILP and pipelining techniques with the CS-aware operation chaining was then introduced. All the above solutions feature an inherent limitation, the optimization of CS is bounded to merging only additions/subtractions. For the operations iffer from addition / subtraction, in each operation it is needed to add a CS to binary conversion.

## III. PROPOSED SYSTEM

DSP accelerators performance can be increased by the optimization of both in architectural and arithmetic level. Flexible data path architecture that exploits carry-save optimized templates which comprises flexible computational unit. In arithmetic level the carry- save arithmetic eliminates the intermediate carry propagate adder which is used to convert carry save to binary format.

The Fig.1. shows the block diagram of flexible computational unit used in DSP Accelerator. The flexible computational unit mainly consists of a compressor, mux's, a multiplier unit and finally an Adder. The detailed view of the basic building block is shown above: The twos complement 4:2 CS adder produces two outputs based on the input carry signal value. if the value of carry is 0 then the N*= X*+Y * otherwise N* = X*-Y * . The MUX 1 determines whether A is multiplied by N* (1) or K* (2). The MUX 2 specifies among K*(1) and N*(2) which is to be added with the multiplication product. The multiplexer MUX 3 accepts the output of MUX 2 and its ones complement and makes an outputs based on the equation 4.1 if the signal CL3 is 0 or it produces an output based on the equation 4.2 if the CL3 is 1. The multiplier comprises a CS-to-MB module, which exploits a recent recoding technique, the 17-bit P* in its respective Modified Booth digits with minimal carry propagation. The multipliers product consists of 17 bits. The compensating method truncation technique is used for reducing the error imposed at the products accuracy of the multiplier. Since all the FCU inputs consist of 16 bits and it does not produce overflows, the 16 most significant bits of the 17-bit W* are inserted in the appropriate FCU when requested.

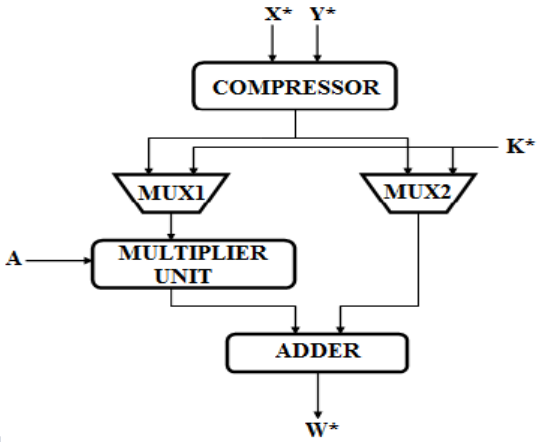


*Fig.1. Block Diagram of FCU*

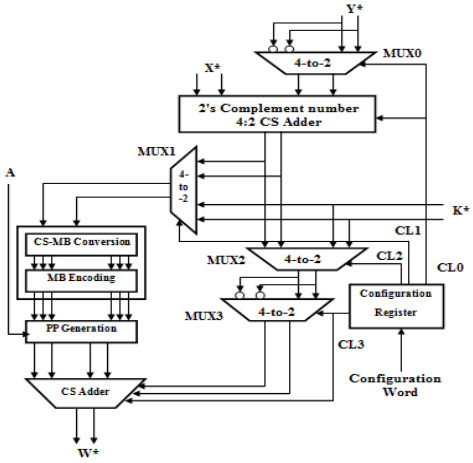


*Fig.2. Detailed view of FCU*

The FCU enables template operation chaining by fused add-multiply operator;

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

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

For increasing the performance of the system we can optimize the add- multiply operator. The existing recoding schemes are on the basis of complex manipulations in bit-level. These recoding schemes are implemented by dedicated circuits in gate-level in this case. In order to get more efficient implementation of the fused Add-Multiply (FAM) unit, direct recoding of the sum of two numbers in its

modified booth (MB) form leads to a more efficient. The efficient design of FAM operators can be done by the direct shaping of modified booth form of sum of the two numbers to sum to modified form(S-MB). The S-MB algorithm is simple in structure and can be modified in order to apply either in signed or unsigned numbers, which can comprise odd or even number of bits. Three alternative schemes of the proposed S-MB approach. That is by using various building blocks such as conventional, signed-bit Full Adders (FAs) and Half Adders (HAs).

### *A. Optimization of Add Multiply Operator*

Optimization of add multiply operator will increase the performance of the system. Sum of two numbers can be directly recoded in Modified booth (MB) which leads to more efficient implementation of the fused Add-Multiply (FAM). The efficient design of FAM operators targets on optimization of recoding scheme for direct shaping of the MB form of the sum of two numbers .The sum to modified booth(S-MB) algorithm is structured is simple and can be easily modified, it can be applied in either signed (in 2s complement representation) or unsigned numbers. Conventional, signed-bit Full Adders(FAs) and Half Adders

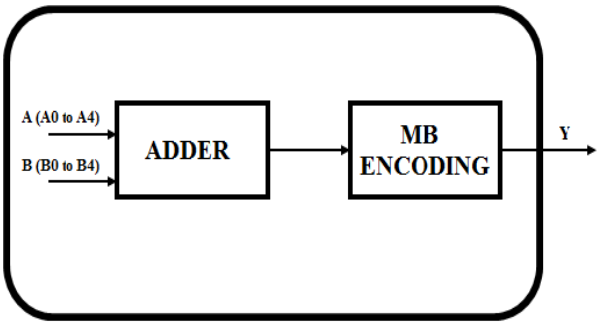


*Fig.3. FAM Unit Design*

(HAs) are the building blocks of S-MB.MB encoding unit into a single data path block (Fig. 2) by recoding directly and fusing the adder depends on the optimized design of Add Multiply operator. Only one adder contain in the fused Add- Multiply (FAM), as a result critical path delay of the recoding process is reduced. The sum to is embedded Adder and encoder block contain the modified booth recoder and it is structured with half adders and full adders .By fusing the area of the FAM design can be reduced. Using S-MB Recoder technique S-MB Recoder block can be implemented.

Conditional sum adder works on some condition. The carry select adder is the one comes under the category of conditional sum adder. Sum and carry is calculated by assuming input carry as 1 and 0 based on the input carry comes. By using the multiplexer sum and carry can be calculated, when actual carry input arrives. The conventional carry select adder consists of n- bit adder for least significant bits (LSBs) and for the most significant bits (MSBs) two n-bit adders. In MSB adders one adder takes carry input as one for performing addition and another carry input as zero.

In digital logic, an adder called carry-look-ahead adder (CLA) is used to improve speed that is by reducing the amount of time to determine carry bits. This adder calculates the carry bits before the sum, so that for large value bits, this adder can reduce the wait time.

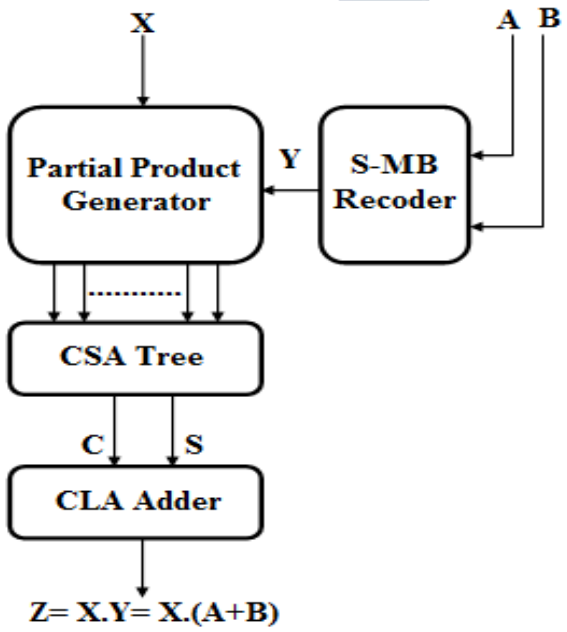


*Fig.4. Fused Design*

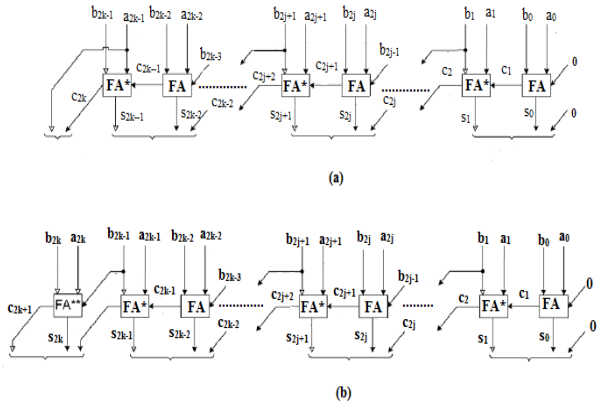

***Fig.5. Block diagram of S-MB 1(a) Even (b) Odd number of bits***

The signed Half Adders and Full Adders is used to design the three new different schemes of the Sum to Modified Booth recoding technique (Fig. 4). These schemes can be applied in both signed and unsigned numbers which can be done on both odd or even number of bits. Consider that both inputs A and B are in 2s complement form and consist of 2k bits in case of even and 2k+1 bits in case of odd bit width.

The basic building consists of different types of half adders and full adders; [ 2 ]

- HAbasic, HA*,HA**.
- FAbasic, FA*,FA**.

By using this basic building block sum to modified booth recoding can be classified into three types

1) S-MB1
2) S-MB2
3) S-MB3

***S-MB1 Recoding:***

The recoding scheme is referred to as S-MB1 and it is obtained for both odd or even number of bit width of input numbers. The encoding of the modified booth digits $Y_j$ mb; 0<j<k-1 is based on both digits $S_{2j+1}$ and $S_{2j}$ are extracted from the j recoding cell. A conventional full adder with inputs $a_{2j}$, $b_{2j}$, $b_{2j+1}$ and produce the carry $c_{2j+1}$.

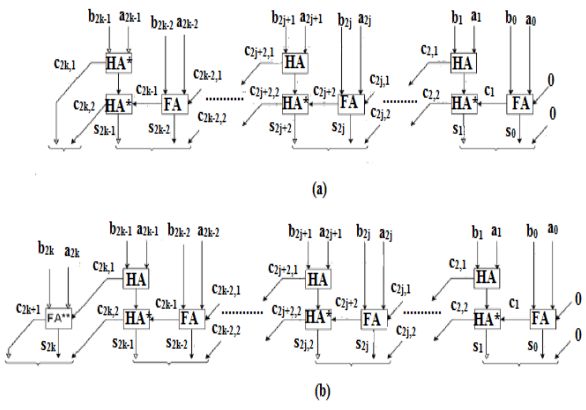

***Fig.6. Block diagram of S-MB 2 (a) Even (b) Odd number of bits***

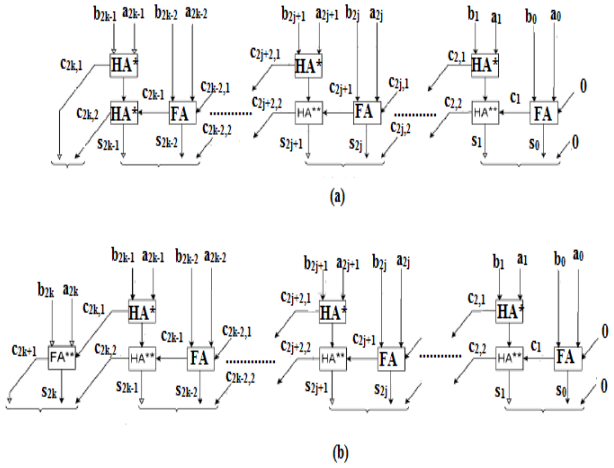

***Fig.7. Block diagram of S-MB 3(a) Even (b) Odd number of bits***

***S-MB2 Recoding****:*

S-MB2 is described for even and odd bit-width of input numbers. Consider the initial values $c_{0,1}$=0 and $c_{0,2}$=0. As in the S-MB1 recoding scheme, we use a conventional FA to produce the carry $c_{2j+1}$ and the sum $s_{2j}$. The inputs of the FA are $a_{2j}$, $b_{2j}$ and $c_{2j,1}$. The bit $c_{2j,1}$ is the output carry of a conventional HA which is part of the (j-1)recoding cell and has the bits $a_{2j-1}$,$b_{2j-1}$ , as inputs. The HA* is used in order to produce the negatively signed sum and its outputs.

*S-MB3 Recoding:*

S-MB3 in which we use a conventional FA to produce the carry $c_{2j+1}$ and the sum $s_{2j}$. The bit $c_{2j,1}$ is now the output carry of a HA* which belongs to the (j-1) recoding cell and has the bits $a_{2j-1}$, $b_{2j-1}$ as inputs. The negatively signed bit $s_{2j+1}$ is produced by a HA** in which we drive $c_{2j+1}$ and the output sum (negatively signed) of the HA* of the recoding cell with the bits, as inputs $a_{2j+1}$, $b_{2j+1}$. The carry and sum outputs of the HA** are obtained**.**

## IV. EXPERIMENTAL RESULTS

### *A. Simulation of Flexible Computational Unit (FCU)*

In Flexible Computational Unit FCU, first the inputs are applied into the 4:2 Wallace compressor. The sum and carry form of both X and Y having 16 bits are the inputs. These are then applied to the S-MB Recoder. The recoded output and the another input A are applied into the partial product generation unit and the partial products generated are added together in the Wallace CSA and the resultant sum and carry obtained here are then applied into the CLA and final

output W is obtained. Simulation of FCU1 done in Xilinx ISE Design suite 13.2 and obtained the output shown in Fig.8 Similarly FCU 2 (Fig.9) and FCU 3(Fig.10) output can be obtained.

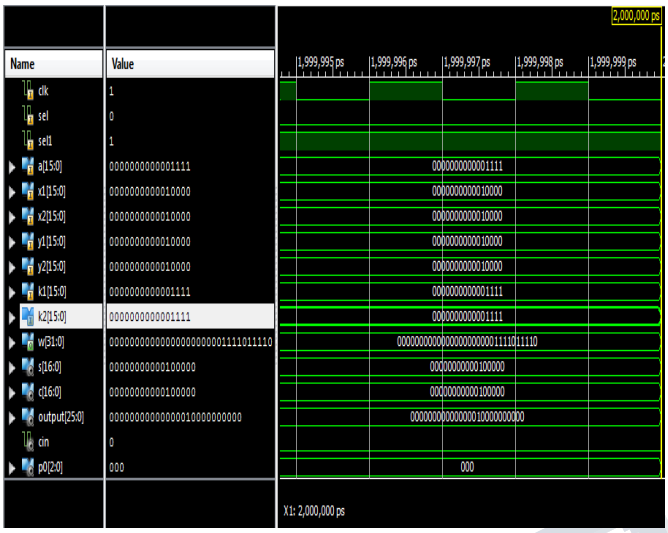


*Fig.8. Simulation results of FCU 1*

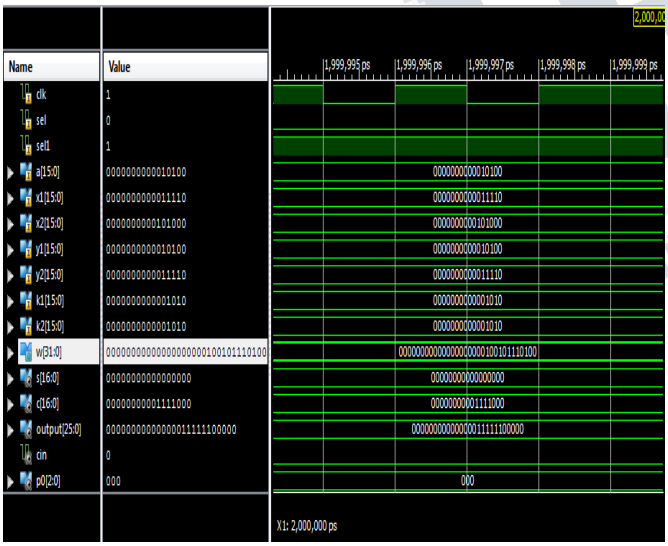


*Fig.9. Simulation results of FCU 2*

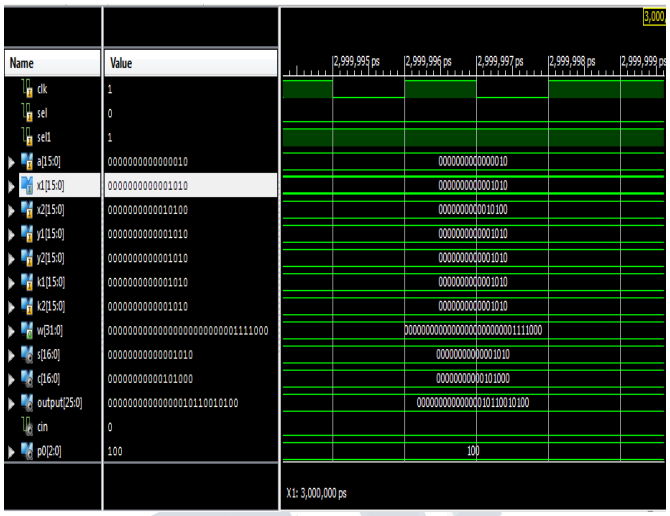


*Fig.10. Simulation results of FCU 3*

## V. CONCLUSION

In this work, optimization of both the arithmetic circuits and architectural level is considered. The flexible accelerator is utilizing the Carry Save (CS) optimization in order to eliminate the large carry propagation chains. This optimization is done by using fused add-multiply operator which mainly consisting of three types of sum to modified booth recoding techniques which can be done in both odd and even form. The fused add multiply algorithm generate the partial products with less computation time. Due to the fusion of add multiply there is no need of conversion from carry save to binary, by this we can improve the performance of an DSP Accelerator. The newly developed Functional Computation Unit (FCU) consists of adders, which will reduce the performance. To increase the performance of FCU the adders are replaced by Common Boolean Logic.

## REFERENCES

[1]K. Tsoumanis, S. Xydis, C. Efstathiou, N. Moschopoulos, and K. Pekmestzi, "Flexible DSP Accelerator Architecture Exploiting Carry-Save Arithmetic", IEEE Trans.Circuits Syst. I, Reg. Papers, volume: 61, no.4, January 2015.

[2] K. Tsoumanis, S. Xydis, C. Efstathiou, N. Moschopoulos, and K. Pekmestzi,"An optimized modified booth recoder for efficient design of the add-multiply operator", IEEE Trans. Circuits Syst. I, Reg. Papers, volume: 61, no.4, April 2014.

[3] M. Stojilovic, D. Novo, L. Saranovac, P. Brisk, and P. Ienne, "Selective flexibility: Creating domain-specific reconfigurable arrays", IEEE Trans.Comput.-Aided Design

Integr. Circuits Syst., volume:32, no.5,May 2013.

[4] G. Ansaloni, P. Bonzini, and L. Pozzi, "EGRA: A coarse grained reconfigurable architectural template", IEEE Trans. Very Large Scale Integr. (VLSI) Syst., volume: 19, no. 6, June. 2011.

[5] 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., volume: 19, no. 3, March. 2011.

[6] K. Compton and S. Hauck,"Automatic design of reconfigurable domain specific flexible cores", IEEE Trans. Very Large Scale Integr. (VLSI) Syst.,volume: 16, no.5,May 2008.

[7] M. D. Galanis, G. Theodoridis, S. Tragoudas, and C. E. Goutis, "A high performance data path for synthesizing DSP kernels", IEEETrans. Comput.- Aided Design Integr. Circuits Syst., volume: 25, no. 6, June. 2006.