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. 4, Issue 6, June 2016

Efficient Implementation of Modified Booth Algorithm in Radix-4 Form

Chaitra Pavate

M. Tech, Dept. of E&C, SDMCET, Dharwad, Karnataka, India

ABSTRACT: Multiplication is one of the important and basic operations involved in any Digital Signal Processing systems.It requires more hardware resources and processing time than addition and subtractions. Generally multipliers are slowest elements in the system. Booth algorithm is one of the many famous algorithms used for multiplication of two numbers. Modified Booth Algorithm is a slight advancement in the coding technique of Booth algorithm. Modified Booth algorithm is more efficient than the standard Booth algorithm in terms of speed, path delays and slices used. Thus the Radix-4 form is compared to Radix-2 form and the speed of the corresponding design is determined.

KEYWORDS: Modified Booth Algorithm; partial products; Radix-4;Radix-2; path delay; slices; LUTs.

I. INTRODUCTION

Multiplication is one of the basic functions used in digital signal processing(DSP). It requires more hardware resources and processing time than addition and subtractions. Multipliers are major components of high performance systems such as FIR filters, microprocessors etc. Generally multipliers are slowest elements in the system. Hence the multiplier performance determines the performance of the system. Furthermore, the area consumption is more. Hence optimising the speed and area is the major design issue. Speed and area are conflicting constraints, so that improving speed results in larger area. Thus fully parallel designs with different area, speed constraints are implemented.

Many of the high performance digital signal processing systems depend on hardware implementation to achieve high data throughput. Thus, multiplication based operations such as Multiply and Accumulate (MAC) are implemented on many DSP applications. These include convolution, Fast Fourier transform, filtering and in microprocessors in its arithmetic and logic units. Generally, a multiplier is a large block of computing system where the amount of circuitry involved is directly proportional to square of its resolution.

Any multiplier can be divided into 3 stages. Generation of partial product stage, addition of partial product stage and final addition stage. The speed of multiplication can be increased by reducing the number of partial products.

Further, Booth recoding highly reduces the number of partial products in multipliers. The benefit is the reduction of area in multipliers with medium to large operand widths. This has Radix-4 implementation. Radix-4 is suitable for high speed applications since hardware cost, power consumption and latency are reduced.

II. RELATED WORK

Many papers have shown the implementation of Booth multiplier. It uses generalized two bit pairing method. Each paired sequence is coded into a new digit based on Booth encoding table. Thus every set of data ,if even bits, is paired into groups of two and then recoded. If the data is in odd number of bits a '0' is appended at the end of the sequence and then coding is done. The table 1 shows Booth encoding table. It consists of four cases following a two bit pairing.Thismethodcanbeusedtomultiplytwo2'scomplementnumberswithoutthesignbitextension.It is observedthatwhenstringsof'1'bitsoccurinthemultiplicandthenumberofpartialproductscanbereduced bysubtraction. This requires high area consumption and thus utilizes larger execution time. The program is divided into a large number of internal fragments. These are called slices. Each slice is sub divided into Look Up Tables(LUTs). More the number of slices more are the LUTs and hence the path delay increases. Though the complexity level of Radix-2 encoding is lower, it does not provide that efficiency as that of higher radix elements.

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. 4, Issue 6, June 2016

TABLE 1
Booth Encoding Table

Yi	Yi-1	Operation	Comment	Zi
0	0	Shift	String of 0s	0
1	0	Sub and Shift	Beg of string of 1s	1
1	1	Shift	String of 0s	0
0	1	Add and Shift	End of string of 1s	1

Though the Booth algorithm is efficient due to pairing of bits and recoding, it lacks in some concerns such as

- There exist different number of add subtract operations and shift operations. Hence difficult to design parallel multiplier
- Thealgorithmdoes not work with isolated1s.

Hence due to the two main disadvantages the algorithm becomes less used and hence there comes the proposed algorithm.

III. PROPOSED ALGORITHM

This algorithm overcomes the disadvantages rose due to the bit pairing in Booth algorithm as a separate case.MBA process uses three number bits each time forrecoding.Recoding the multiplier with greater radix value increases the speed of Booth algorithm. If only a single bit is remaining zeros are appended. In every machine cycle the number of bits analyzed is higher. Thus machine cycles are reduced to obtain products. Number ofbitsinspectedinradixrisgivenbyn=1+$\log_2$r.

Description of the proposed algorithm:

Step 1: Attach a 0 to LSB right.
Step 2: If n is even extend the sign bit by 1 position.
Step 3: Find each partial product according to value of each vector.

TABLE 2
Modified Booth Encoding table

P(2i+1)	P(2i)	P(2i-1)	Recoded Digit	Operand *Multiplication
0	0	0	0	0*Multiplicand
0	0	1	+1	+1*Multiplicand
0	1	0	+1	+2*Multiplicand
0	1	1	+2	+2*Multiplicand
1	0	0	-2	-2*Multiplicand
1	0	1	-1	-1*Multiplicand
1	1	0	-1	-1*Multiplicand
1	1	1	0	0*Multiplicand

Multiplierdigits are reducedbyafactoroftwo in Radix 4. Thus 16 to 8 is the reduction of multiplier digit length.Carry is not propagated intosubsequentstages in Booth'srecoding. Heretheoriginalmultiplier is divided into group of three. Eachofthesenumbersfromtheset{2,10,-1,-1}Eachrecoderproducesa3-bitoutputwherethefirstbitrepresentsthenumber1andthesecondbitrepresentnumber2.Thethirdbit indicates which of the prior bits are negative. Bit pairing is as shown in Figure 1. Grouping starts from left hand side. It compares three bit at a time with an overlapping technique. We can notice that a '0' is appended to make every three bits paired.

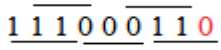

Figure 1. Bit pairing in MBA

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. 4, Issue 6, June 2016

The negative values are constructed by taking 2's complement. Multiplication is done by shifting one bit left.

IV. COMPARISON

Two separate units, one with the Radix-2 code and other with the Radix-4 code is run. Radix-2 is easily grouped by taking two bits at a time. The single bit which is unpaired is left free and is not coded. Thus being the greatest disadvantage as the codes may be in accurate. On the other hand there exists Radix-4 codes where each case is separately taken care of, including the left over ones. Thus the grouping and the coding technique is highly efficient. Each of these codes has separate utilization of area, speed and path delay.Radix-2 code in which the numbers are grouped two bits at a time consumes higher utilization. This utilization gives us the summary about amount of total space used for running the code. Space is determined in terms of slices and LUTs. Radix-4 utilizes less device space i.e less number of slices than the one used for Radix-4.

V. SIMULATION RESULTS

The simulation studies involve the behavior of Radix-2 codes and Radix-4 codes. The simulation for each of the algorithms is shown. We can observe that the simulation for Radix-2 contain seven number of partial products and that of Radix-4 contains only four. This reduces the overhead on the processor , thus improving the system performance. The design summary is the one which gives the complete report on system utilization. These may generally include number of slices the given program is divided into and number of LUTs a slice is divided into. Path delay is also compared between the two designs. Table 3 shows the design summary for Radix-2 and Table 4 shows the design summary for Radix-4. We can see that the number of LUTs and slices occupied in Radix-2 is much greater than Radix-4. IOBs also is higher(Input Output Bounds). The path delay for the Radix-2 code is much higher than that of Radix-4.

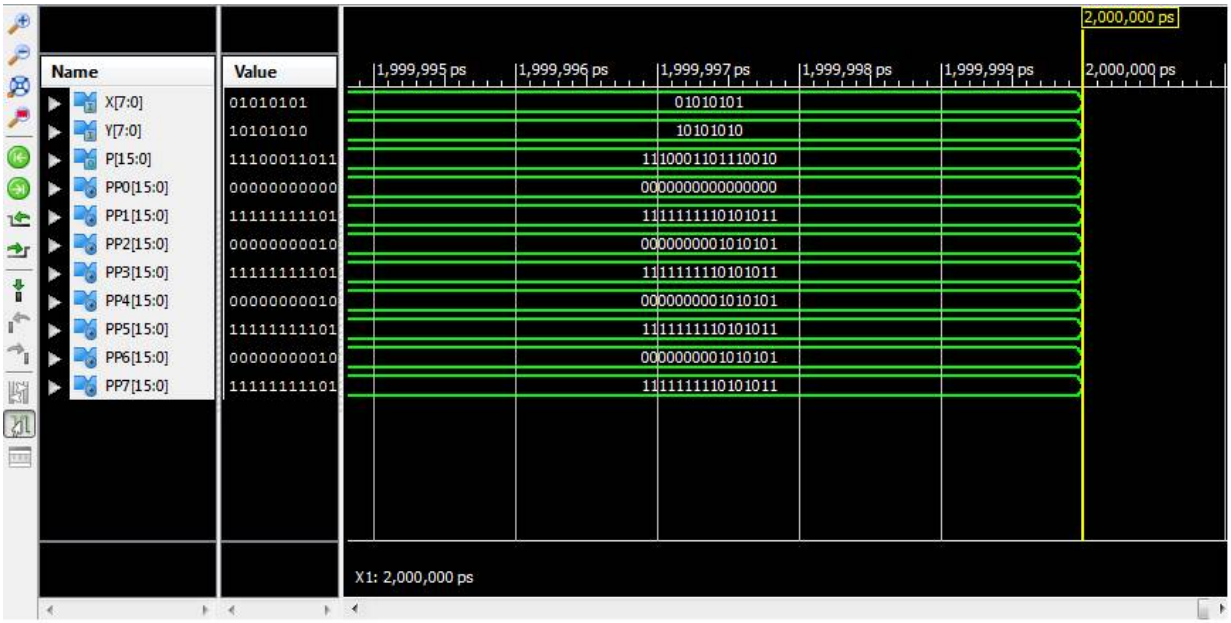

Figure 2. Simulation of Booth algorithm in Radix-2

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. 4, Issue 6, June 2016

Details	Used	Available	Utilization
No. of 4 input LUTs	242	1536	15%
No. of occupied Slices	127	768	16%
No. of bonded IOBs	32	124	25%
Path delay	49.177nS		

Table 3. Design Summary for Radix-2

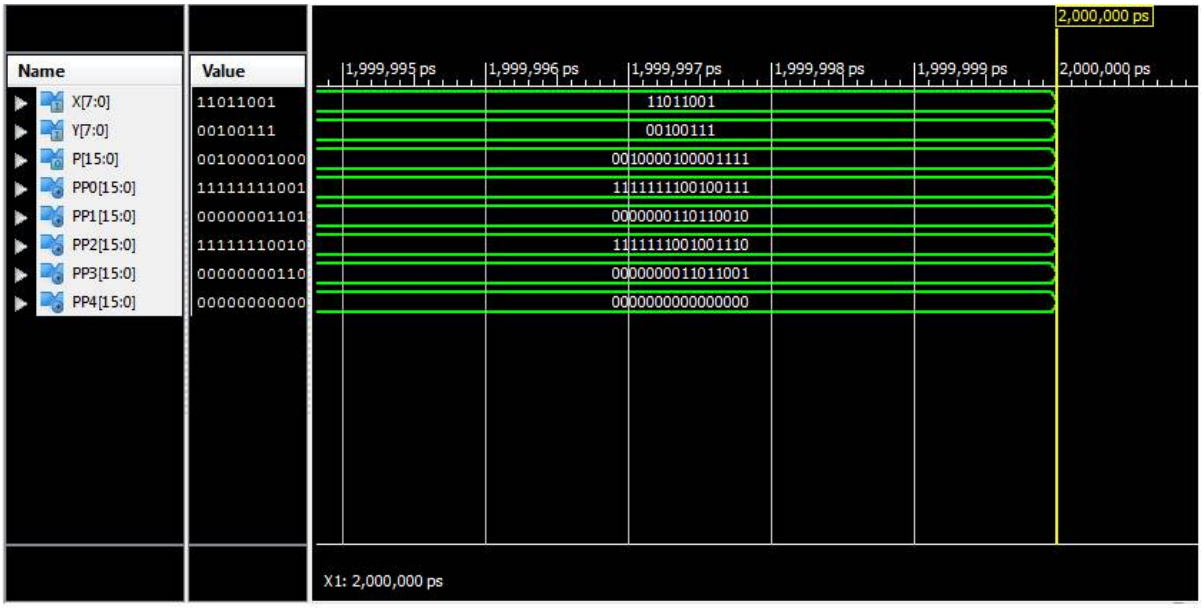

Figure 3. Simulation of Booth algorithm in Radix-4

Details	Used	Available	Utilization
No. of 4 input LUTs	210	1536	13%
No. of occupied Slices	113	768	14%
No. of bonded IOBs	32	124	25%
Path delay	35.801nS		

Table 4. Design Summary for Radix-4

VI. CONCLUSION AND FUTURE WORK

The results showed that the proposed algorithm performs better than the Radix-2 structure in terms of area consumed, path delays, number of LUTs and slices used. Thus Radix-4 is much efficient algorithm than Radix-2. As future work Radix-8 bit implementation can be made and can be compared with Radix-2 and Radix-4. Thus the device utilization for Radix-8 can be calculated.

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. 4, Issue 6, June 2016

REFERENCES

1. SukmeetKaur, Suman, Manpreet Singh Manna "Implementation of Modified Booth Algorithm in Radix-4 and its comparison with Booth Algorithm". Research India Publications, Vol 3, Number 6.
2. Kavita and JasabirKaur, "Design and Implementation of efficient Modified Booth multiplier using VHDL", ICETEM, 2013.
3. Vivek Joshi and Bonifus P L ," Implementation of FFt butterfly using SMB recoding techniques.".
4. M. Chitra and P. Maria"An efficient design of Sum-Modified Booth recoder for fused add-multiply operator",IJIET.
5. S. Jagdeesh, S. Venkatachary, " Design of Parallel Multiplier- Accumalor based on Radix-4 Modifies Booth Algorithm with SPST", IJERA, Vol 2, Issue5, 2014.
6. P.S.Tulsiram, D.Vaithiyanathan, "Implementation of Modified Booth recoded Wallace tree multiplier for fast arithmetic circuits ", IJARCSSE, Vol 4, Isuue 10,2014
7. Dr. C Venkatesh, " Optimising the power using fused add multiplier", IJCSMC, Vol 3, Issue 11,2014
8. Simran Kaur, " FPGA Implementation of efficient modified Booth Wallace multiplier", Thapar University, Patiala
9. Neeta Sharma, Ravi Sindal, "Modified Booth Muktiplier using Wallace structute and efficient carry select adder", IJCA, Vol 68, No.13,2013
10. Moumita Ghosh, " Design and implementation of different multipliers in VHDL",NIT,Rourkela
11. Dr. B. Gopi, G. Kohila "Optimisation of power in fused add multiply operator using Modified Booth recoder", IRJET, Vol 2, Isuue 2, 2015
12. Manju Mallayagari, Sahithi Reddy, "Optimising the Modified Booth Recoder for fused add multiply operator", SSRG_IJVSP, Vol 2, Issue 3,2015

BIOGRAPHY

Chaitra Mohan Pavate is a PG student in Digital Electronics, SDMCET, Dharwad, Karnataka, India. She received Bachelor of Engineering (BE) degree in 2013 from VTU, Belgaum, Karnataka, India. Her research interests are in the fields of DSP and Verilog.