

---

# Realizing Ternary Logic in FPGAs for SWL DSP Systems

TAYAB DIN MEMON\*, IRFAN AHMED HALEPOTO\*, AND AHMED AL-OTABI\*\*

RECEIVED ON 27.08.2012 ACCEPTED ON 20.03.2013

## ABSTRACT

Recently SWL (Short Word Length) DSP (Digital Signal Processing) applications has been proposed to overcome multiplier complexity that is evident in most of the digital applications. These SWL applications have been processed through sigma-delta modulation as a key element. For such applications, adder design plays vital role and can impact upon the chip area and its performance. In this paper, a ternary approach for adder tree has been proposed instead of binary that can accommodate more data with less chip-area at the cost of extra pin. The proposed ternary adder tree has been designed and developed in Quartus-II using three different design strategies namely T-gate (Ternary gate), LUT (Look Up Table) and algebraic equations. Through rigorous simulation it was found that T-gate technique results in superior performance, an average of 23.5 and 33% improvement compared to the same adder structure based on Boolean Algebraic Equation and LUT, respectively. The proposed adder design would benefit the efficient implementation of SWL applications.

**Key Words:** Ternary Adder, Single-Bit DSP System, Sigma Delta Modulation, VHDL.

## 1. INTRODUCTION

Multiplier complexity is very common issue in the digital signal processing that starts from convolution of two inputs to the fast Fourier transform. Following to the multiplier complexity is adder tree that sums up the partial products and provide the final sum output. An efficient design algorithm is needed for adder tree implementation in hardware to minimize the latency. Various adder tree approaches have been reported including: fully parallel, fully serial, serial-in parallel out format etc. Despite the low chip area in serial format, parallel processing offers higher performance [1].

General purpose digital algorithms implemented by using sigma delta modulated SWL techniques are promising alternatives to the contemporary multi-bit systems, and have the obvious advantages of reduced hardware complexity, simple arithmetic, low power consumption and robustness. Recently, a classical LMS (Least Mean Square) like single-bit approach has been proposed in [2-4]. In LMS-like approach, authors has simulated LMS-like block adaptive filter using binary, ternary and 2-bit word coefficients and compared to its counterpart multi-bit adaptive LMS algorithm. It has been shown that in all

---

\* Assistant Professor, Department of Electronic Engineering, Mehran University of Engineering & Technology, Jamshoro.

\*\* School of Electrical and Computer Engineering, RMIT Melbourne, Victoria, Australia.

cases, especially, single-bit ternary and 2-bit word coefficients, LMS like algorithm converge and has improved SNR (Signal to Noise Ratio) performance than its counterpart multi-bit system to earlier stages. In all these experimental setups, input, output and coefficients remain in single-bit nature.

It is already proved that, short word length DSP systems has tendency of high noise shaping due to fast clock rate as compared to the Nyquist rate [2]. Hardware complexity is reduced by the SWL systems with simple multiplexers and a LUT, which is highly efficient for current processors especially FPGAs (Field Programmable Gate Arrays). The shortcoming of the fast clock rate is the large number of coefficients (i.e. samples) as compared to multi-bit system. Therefore, optimization is needed in adder circuits to deal with such type of systems to compete with area and performance tradeoff.

Memon, et. al. [5], it is reported that SWL based systems exhibit lower area-performance results compared to their multi-bit counterpart. In this work, authors has shown that SBTF (Single Bit Ternary FIR-Like Filters) offers superior performance than its multi-bit counterpart in both modes i.e., pipelined and non-pipelined by a ratio of on average 32% except in the most extreme case of 18-bit (i.e. 8196) coefficients. For both the design approaches, dynamic power dissipation (i.e., switching) was computed using build-in power analysis tool in Quartus-II with two approaches i.e.  $F_{MAX}$  approach and Nyquist rate approach [5]. In both approaches, it was shown that, SBTF dissipates lower power than multi-bit approach except extreme case of coefficients i.e. 8192. Superior performance of multi-bit systems at the extreme case of coefficients, indicated that higher clock rates of

SWL approaches impacts upon chip area that counts as latency overall. In result, as coefficients number goes high, performance deteriorates. It is also reported that these systems are more efficient in the performance domain while system parameters remain in single-bit domain i.e., input and output. It can be either binary or ternary format as reported in [2, 6].

In this paper, we have taken more optimized approach to mainly focus on improvement of area-performance tradeoffs for ternary FIR (Finite Impulse Response) filter [3,7-8] by considering ternary base instead of binary. In this work, we have addressed directly implementation of a ternary adder tree using a balanced ternary logic. Using three different approaches i.e. T-gate, LUT, and algebraic expression, area-performance tradeoffs has been reported. It is concluded that T-gate techniques offers better performance as compared to other contemporary techniques.

The remainder of this paper proceeds as follows. Three different adder modules are discussed in Section-2, followed by design and implementation of balanced ternary adder tree in Section 3. FPGA based simulation and results are discussed in Section 4. Finally, we summarize and conclude the paper and point to future work in Section 5.

## 2. BALANCED TERNARY ADDER DESIGN

The term trit is defined as the input taken from the set of  $\{+1, 0, -1\}$ . BTFA (Balanced Ternary Full Adders) add two trit inputs and have a carry-in trit; after doing the summation, it generates a sum trit along with a carry trit. The half adder follows the same trend but it does not have carry-in trits. Varying voltage trits are defined in VHDL

(Verilog Hardware Description Language) by using 2's complement encoding for each single ternary trit. The simplest logic used for this purpose is defined as: 11→-1, 00→0 and 01→+1. This encoding has been used for simulation purpose in all three adder design techniques (discussed in subsequent sections).

We may have used other available techniques for example CSD (Canonical Signed Digit) encoding technique that has ternary inputs similar to the proposed technique. However, it is proved by authors that canonical signed digit and two's complement approach has no difference in single-bit domain [9]. Therefore, we have choice of using either of the techniques for current application. Hence, we have adopted 2's complement approach than CSD encoding technique. The possible truth table for ternary logic is given in Table 1.

It can be observed that (Table 1) BTFA produces two trit outputs i.e., summation and a carry-out. Unlike BTFA, half-adder ternary circuit has no carry-in. The inputs and the outputs are all in the two ternary bit form (i.e. trits), which represents the two's complement of the ternary trit.

### 2.1 Ternary Adder Using LUT

In first place, we have designed and synthesized the LUT approach for adder tree using full adder and half adder and it produces definite output for each input. The benefit of adopting such approach was that the output for any combination is well known beforehand. The truth table presented in Table 1 was used for the LUT approach. The output is thus received from the corresponding I/O in the form of the sum and the carry out.

### 2.2 Ternary Adder Using Boolean Algebra Equations

This method exploits the Boolean algebra equations to express the ternary adder. In this technique, the truth table presented in Table 1 is mapped in VHDL and its functions are explained mathematically using Boolean equations.

### 2.3 Ternary Adder Using T-gate

Here, we have employed T-gate for the design of ternary adder tree. For the proposed ternary adder

TABLE 1. TRUTH TABLE FOR THE TERNARY FULL ADDER [9]

A	B	Carry in	Carry Out	Sum
1	1	1	1	0
1	1	0	1	-1
1	1	-1	0	1
1	0	1	1	-1
1	0	0	0	1
1	0	-1	0	0
1	-1	1	0	1
1	-1	0	0	0
1	-1	-1	0	-1
-1	-1	-1	-1	0
-1	-1	0	-1	1
-1	-1	1	0	-1
-1	0	-1	-1	1
-1	0	0	0	-1
-1	0	1	0	0
-1	1	-1	0	-1
-1	1	0	0	0
-1	1	1	0	1
0	0	-1	0	-1
0	0	0	0	0
0	0	1	0	1
0	1	-1	0	0
0	-1	-1	-1	1
0	-1	0	0	-1
0	-1	1	0	0
0	1	1	1	-1

design, we needed fourteen T-gates to develop a single ternary adder module as shown in Fig. 1 [9]. In functional sense, T-gate is similar to multiplexer that has that has control input available in between three logic values. Here again, Table 1 was used to build the ternary adder module.

### 3. THE BALANCED TERNARY ADDER TREE

The adder tree circuits are used in circuits for adding up several operands and put the final output in a single unit form. Following is given as an example of balanced adder tree design using proposed method.

#### 3.1 Example Balanced Ternary Adder Tree

Balanced ternary adder tree was designed, developed and synthesized in Quartus-II as an example using 16 inputs as depicted in Fig. 2 using VHDL programming. As discussed earlier, for 16 inputs representation, we will require thirty-two binary bits. The number of total adder stages was calculated using  $\log_2 N$  method. We have

considered input (i.e. N) as 16, so by using log formula, the total number of adder stages would become 4 ( $\log_2 16$ ). The final output stage would be considered as a summation stage. However, the adder blocks would become half at each successive stage of the adder tree. Hence, number of inputs are 16, so in first stage number of adder modules will be eight, and will reduce to four at second, and 2 at third stage and final stage has only one adder block (Fig. 2 and Table 2). From Table 2, it can be noted that every ternary number is represented by two digits as explained earlier according to two's complement format. Hence, dot is used to separate each number into the given Table 2.

For the first stage, that possesses eight adder blocks, we have used all half-adder circuits, which can accommodate two trit inputs and finally produce a two trit output. Unlike to first stage, in the next step, four adder blocks are required, in which each block has a half and a full adder. In this case, all adder blocks requires fours inputs and produce two outputs. Here, the summation output would become input to the half adder and carry output would become input to the full adder for the next stage, along with the carry out of the half adder in the same stage. The adder tree stage discussed here takes two trit (2-bit) inputs and produces three-trit outputs. The third trit-bit output (i.e. carry-out) has been ignored while sum bits are preserved for the next stage. By ignoring the carry-out bit, it will not affect the final output due to the balanced ternary nature of the design. The maximum and minimum of the adder blocks are from the set of  $\{+1, -1\}$ . Thus the output sum can be represented by using the two bits rather than three and last bit always be '0' that may be neglected. By neglecting this last bit, will minimize the chip area and improves the performance of the system.

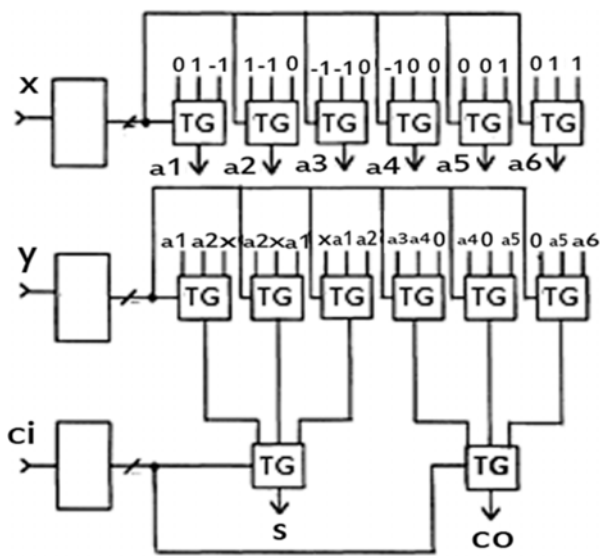


FIG. 1. A FULL TERNARY ADDER USING T-GATES [9]

In the third stage, just two adder blocks are needed so we have used one full adder and one half adder block. Half adder has been used in this stage, because the total number of inputs can be handled easily. Number of inputs given to the third stage are four, while the output produced by this stage are three, that is one ahead of the last stage. Here, we didn't ignore the third input; because the maximum output at this stage can be  $\pm 8$ , so three bits are required to handle it which is represented in the balanced ternary by 101 and 101 pattern respectively, and thus it requires only three outputs.

In last stage, predictable output result of -8 (signed number) that is depicted in Table 2 (in base 3 i.e. 00110001) is obtained from the four trit outputs obtained through single adder block. Like previous stages, here we have build final block by using two full adders and one half adder.

TABLE 2. THE SIMULATION EXAMPLE OF THE TERNARY ADDER TREE WITH 16-INPUTS

Inputs	Stage-1	Stage-2	Stage-3	Stage-4
11	00.11	11.00	00.11.11	00.1100.01
00				
11	11.01			
11				
01	00.01	00.11		
00				
11	11.01			
11				
11	00.11	00.11	00.11.11	
00				
11	00.00			
01				
00	00.11	11.00		
11				
11	11.01			
11				

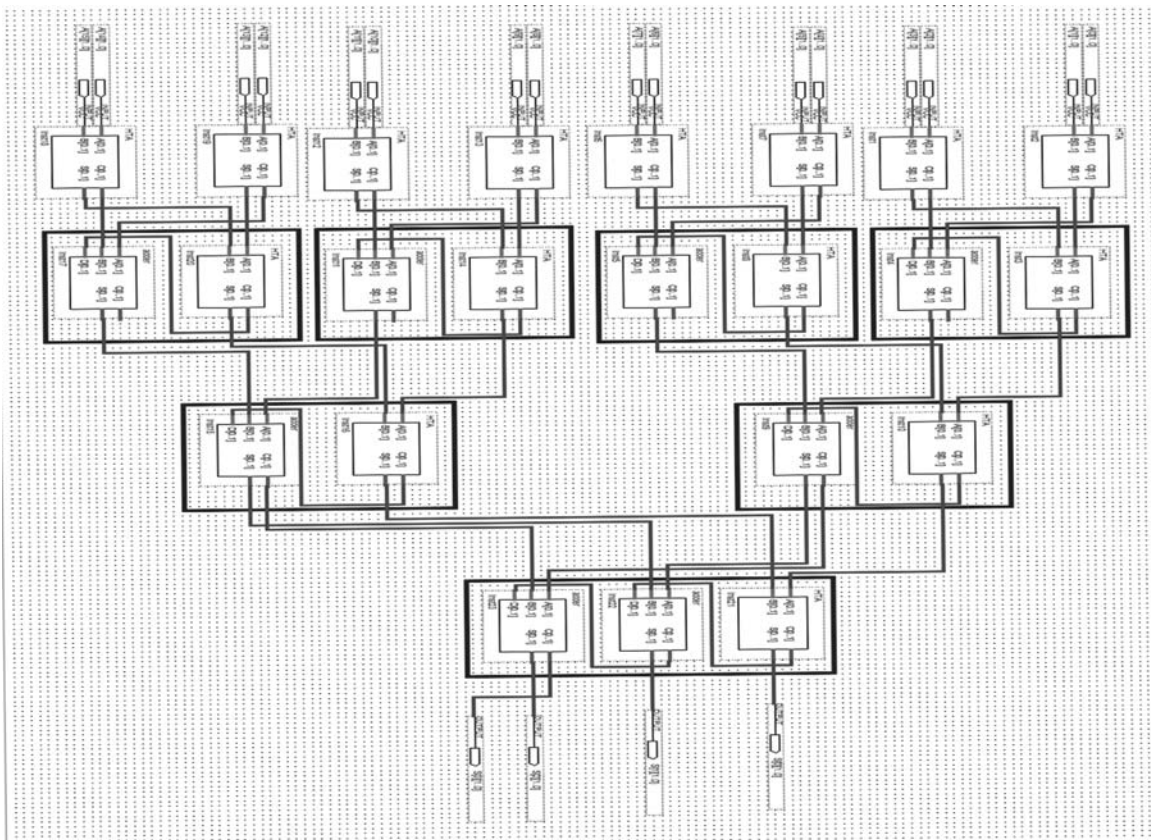


FIG. 2. TERNARY ADDER CIRCUIT FOR 16 INPUTS

#### 4. SIMULATION RESULTS AND DISCUSSION

In this work, we have obtained experimental results with inputs in power of 2 i.e.  $2^N$ , and maximum  $N$  number was 7 (i.e. 128). Overall experimental setup was kept same except  $N$  number was changed at each stage that changed the overall adder levels. For the convenience of understanding, all the design approaches were simulated and synthesized in pipelined and the non-pipelined configurations. In Table 3 all the three techniques results obtained in pipelined and non-pipelined modes are presented. The maximum clock

frequency given in Table 3 was obtained by inserting extra constraint into the software as worst case results may be obtained.

It can be seen that MUX (Multiplexer) offers better results as compared to other two techniques with respect to area- performance. Further to this, it can be realized that MUX occupies 30% less LUTs than LUT ternary adder approach. In same way, Algebraic equation approach occupies 10% lower LUTs than ternary adder approach. Thus, MUX consumes 20% lower chip area than Algebraic equation approach with Cyclone-III

TABLE 3. THE BALANCED TERNARY ADDER TREE RESULTS USING THE THREE METHODS

Device	Number of Inputs	Adder Approach	Non Pipelined			Pipelined		
			LUTs	Registers	F <sub>MAX</sub> (MHz)	LUTs	Registers	F <sub>MAX</sub> (MHz)
Cyclone-III EP3C120F484C7	16	MUX	125	40	136	120	100	*250
		Boolean Equation	170	40	101	151	100	245.95
		LUT	17	40	99	167	100	220.90
	64	MUX	630	138	74	583	426	241.25
		Boolean Equation	797	138	55.5	738	426	184.60
		LUT	958	138	44.5	856	426	164.77
	128	MUX	1267	268	61.03	1225	864	203
		Boolean Equation	1673	268	48.75	1542	864	163
		LUT	2025	268	38.5	1774	864	146.56
Stratix-III EP3SL340F1517C4	16	MUX	79	49	237	86	100	*360
		Boolean Equation	83	44	176	84	100	*360
		LUT	163	46	166	84	100	*360
	64	MUX	382	190	155	383	429	*360
		Boolean Equation	402	152	120.3	389	432	*360
		LUT	787	145	92.2	384	427	*360
	128	MUX	791	387	121.1	787	869	*360
		Boolean Equation	865	317	94	790	868	*360
		LUT	1745	304	69	794	866	*360

device. Hence, ternary and Algebraic equation approach offers poor performance than MUX that is on average 33% and 24%. It can be seen that (Table 3) MUX maximum clock frequency reaches to 203 MHz, compared to ternary and Algebraic equation approach at similar conditions is 146 and 163 MHz.

In pipelined mode, performance of the design increases at the cost of higher chip area because extra registers added at each level reduces latency and boosts the performance.

As per expectation, maximum clock frequency ( $F_{MAX}$ ) obtained by using Stratix-III in Quartus-II, remained higher as compared to Cyclone-III. Moreover, it can be seen that with Stratix-III, the number has reached to the maximum  $F_{MAX}$  and it can't go beyond it due to its I/O limitations. All these results were obtained using maximum frequency constraint applied through software settings.

However, in this work, we have typically synthesized inputs in the maximum range of 128s; however, this number may be higher for general purpose DSP applications. Therefore, maximum clock frequency obtained with Stratix-III, may deteriorate. Apparently, it is expected that performance results for all the experimental results would increase mostly linearly and trend will follow as give in Table 3.

## 5. CONCLUSIONS

In this paper, we have proposed a new design for a balanced ternary adder tree. This is comprised of half and full ternary adders so that direct implementation could be supported when a balanced ternary logic system is put into effect, with the logic set  $\{-1, 0, 1+\}$ .

Three different design methods were used to synthesize the proposed design in Quartus-II and area-performance tradeoffs for the all three design techniques i.e. T-gate, LUT, and algebraic expression have been determined.

At this stage, we only considered the design and synthesize of three approaches in FPGA. In next stage we aim to synthesize the application designs using these techniques. This will require more emphasis on the single-bit ternary FIR filter's structural design and simulation.

## ACKNOWLEDGEMENTS

Authors are highly thankful to Prof. Zahir Mohammad Hussain, RMIT Melbourne, Australia, who helped us throughout this work and provided technical support at all stages. Authors are very thankful to Mehran University of Engineering & Technology, Jamshoro, Sindh, Pakistan, for providing all necessary facilities for this research.

## REFERENCES

- [1] Meher, P.K., Chandrasekaran, S., and Amira, A., "FPGA Realization of FIR Filter by Efficient and Flexible Systolization Using Distributed Arithmetic", IEEE Transaction on Signal Processing, Volume 56, pp. 3009-3017, 2008.
- [2] Sadik, A.Z., and Hussain, Z.M., "Short Word-Length LMS Filtering ", ISSPA, pp. 1-4, Sharjah, UAE, 2007.
- [3] Memon, T.D., Beckett, P., and Sadik, A.Z., "Efficient Implementation of Ternary SDM Filters using State-of-the-Art FPGA", Mehran University Research Journal of Engineering and Technology, Volume 30, pp. 207-212, Jamshoro, Pakistan, 2011.

- [4] Sadik, A.Z., and Hussain, Z.M., "Adaptive LMS Ternary Filtering", IEEE Region 10 TENCON, Melbourne, Australia, 2005.
- [5] Memon, T.D., Beckett, P., and Sadik, A.Z., "Power-Area-Performance Characteristics of FPGA Based Sigma-Delta Modulated FIR Filters", Journal of Signal Processing Systems, 2012.
- [6] Thompson, A.C., "Techniques in Single-Bit Digital Filtering", Ph.D. Thesis, RMIT University, 2004.
- [7] Thompson, A.C., Hussain, Z.M., and O'Shea, P., "Efficient Digital Single-Bit Resonator", IEEE Electronic Letters, Volume 40, pp. 1396-1397, 2004.
- [8] Al-Otabi, A., "VHDL Implementation for a Balanced Ternary Adder Tree in FPGA", Ph.D. Thesis, RMIT University, 2011.
- [9] Srivastava, A., and Venkatapathy, K., "Design and Implementation of a Low Power Ternary Full Adder", VLSI Design, Volume 4, pp. 75-81, 1996.
- [10] Memon, T.D., and Beckett, P., "The Impact of Alternative Encoding Techniques on the FPGA Implementation of Sigma-Delta Modulated Ternary FIR Filters ", Australian Journal of Electrical and Electronics Volume 10, No. 1, Australia, March, 2013.