Tertiary-Tree 12-GHz 32-bit Adder in 65nm Technology

Amir Agah and S. Mehdi Fakhraie
School of ECE, University of Tehran
Tehran, Iran
Email: a.agah@ece.ut.ac.ir, Fakhraie@ut.ac.ir

Azita Emami-Neyestanak
Department of Electrical Engineering, Columbia University
New York, USA
Email: azita@ee.columbia.edu

***Abstract*— This paper presents a new 32-bit adder structure with 12 GHz low-power operation in 65nm technology. The Fast Conditional Sparse-Tree Logic (FCSL) is based on modifying the initial Sparse-Tree architecture [1] to enhance its speed using tertiary trees and applying a carry-select scheme in some of the more significant bits. This design has been compared with the Sparse-Tree adder and the Low-Voltage Swing adder in terms of speed and power. It has been shown that speed can be improved using FCSL architecture while keeping the power at a comparable level.**

I. Introduction

With the growth of demand for high-speed low-power processors, it is essential to improve the performance of adders as one of the most critical parts of the processors.

The widely-used Sparse-Tree adder architecture [1] is shown in Fig. 1. While it has delivered a high performance used in some commercial products [1] it has some limitations increasing the operating frequency: First, in this architecture, there are three stages of domino-static logic clocked with three signals which are delayed by two buffer stages. Such delays, which can vary depending on the technology, pose one major hurdle increasing the clock frequency. Furthermore, wiring of these clocks could be troublesome. The number of clock signals in our proposed architecture called Fast Conditional Sparse-Tree Logic (FCSL) has been reduced to two clocks saving one buffer delay enabling faster clocking.

Furthermore, in the Sparse-Tree structure [1], there are 4-bit Sum-Generate circuit blocks, shown in Figure 2. As shown in the figure, carry look-ahead structure is used in the 4-bit sum-generate circuit. Such structure has a rather long delay limiting the clock frequency of the 32-bit adder. In the proposed FCSL design, this problem has been solved by using 3-bit Sum-generate circuits instead of 4-bit ones. One advantage of using 3-bit sum-generate circuit is lowering the power dissipation of the circuit. This is further discussed in Section III.

In 32-bit Sparse-Tree adders, seven carry signals are generated in the carry-generate block, each of which is used in the 4-Bit conditional sum-generate circuit. In FCSL, however, ten caries are generated and then used in 3-bit conditional sum-generate circuits. The structure of FCSL is depicted in Fig. 3. The structure of the 3-bit conditional sum-generate circuit is completely static. No dynamic structure has been used in the 3-bit conditional sum-generate circuit to help this circuit work faster in cascade with the carry-generate circuit.

Using fast pull-up logic (FPL) [2] is another idea that lets increase the clock frequency of the adder. However, it comes at the price of increasing the power dissipation of the circuit (13% in our case) as discussed in Section III.

In the next section we will introduce the Sparse-Tree adder architecture. Section III is dedicated to the FCSL architecture, and Section IV includes the simulation results of the FCSL adder simulated in BPTM 65nm technology and also presents comparisons between the FCSL and other architectures.

II. SPARSE-TREE ADDER

Sparse-Tree adder architecture [1] is composed of two parts as depicted in Figs. 1 and 2: The carry-generate circuit, which is the critical path, and the sum-generate circuit. The carry generate circuit (Fig. 1) consists of six stages: an initial PG generator stage followed by five stages of carry merge logic, each merging two carries. Three stages of Domino-static logic

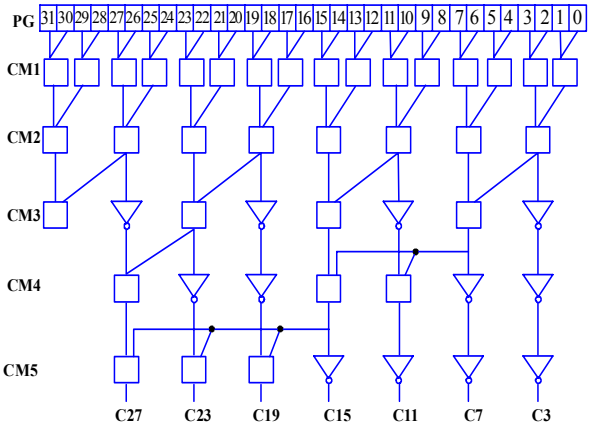


Fig 1. Carry generate structure of the traditional Sparse-Tree adder [1].

1-4244-0921-7/07 $25.00 © 2007 IEEE.

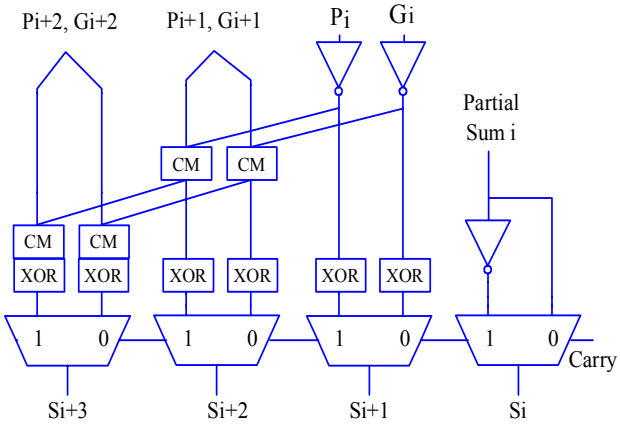


Fig 2. Sum-generate structure of the traditional Sparse-Tree adder, corrected from [1].

are used in this design requiring three clock signals. The main clock signal is applied to the PG generate stage. For the next domino stage, a buffer-delayed version of the clock signal is used, and for the last domino stage the clock is buffered for the second time.

The sum-generate circuit, coming after the carry-generate structure is shown in Fig. 2 and is a 4-bit conditional sum-generate circuit. The architecture is ripple-carry adder, which adds 4 bits for each of the two possibilities of the carry-in signal. Each of the output signals of the carry generate circuit is applied to one of these adder blocks to select which assumption is true.

III. FCSL STRUCTURE

FCSL uses the same general topology as the Sparse-Tree adder but with some modifications to increase its speed. This section explains these modifications in more detail.

A. Carry-Generate Structure in FCSL

The structure of the FCSL architecture is shown in Fig. 3. As seen in the figure, the carry-generate structure in FCSL has a depth of four stages.

In the first stage, Generate (Gi= Ai . Bi) and Propagate (Pi= Ai + Bi) signals are generated from the adder inputs *Ai#* and *Bi#* with dynamic logic using the clock signal Clk1 shown in Figure 4.

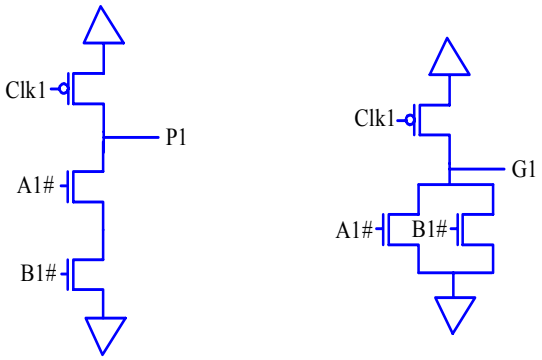


Fig 4. Dynamic P-generate and G-generate circuits in stage 1.

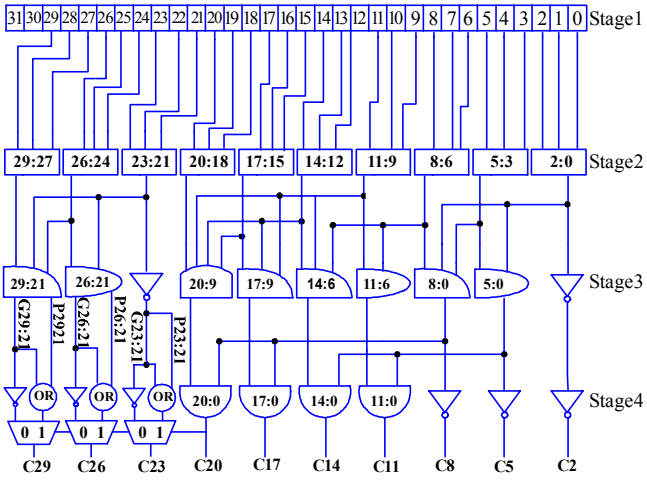


Fig 3. Carry-generate structure of FCSL.

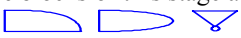
TABLE I.
POWER DISSIPATION AND PROPAGATION DELAY OF THE ADDER WITH AND WITHOUT FPL

	Power Dissipation	Propagation Delay
With FPL	1.925mW/GHz	84.6ps
Without FPL	1.7mW/GHz	91.46ps

In the second stage, which uses static logic, Three of the generated P and G signals are merged to generate 3-way group generate and group propagate signals. For instance, the first block generates *G0, G1, G2, P1* and *P2*. These are merged in the second stage building the *G2:0* signal as:

$$G2{:}0 = G2 + G1P2 + G0P1P2 \quad (1)$$

In order to improve the speed of the circuit, we have implemented the second stage with Fast Pull-Up Logic (FPL) [2] as shown in Fig. 5. FPL is a ratioed logic. We have used this logic family in the second stage of the carry generate circuit to decrease the pull-up time of the output, since it is obviously falls on the critical path of the carry-generate circuit. The advantages and disadvantages of this family of logic have been discussed in [2]. The improvement in the propagation delay of the stage by switching to FPL is demonstrated by simulations and the results are shown in Table I. Using the FPL stage, however, has a drawback of increasing the power dissipation because of being a ratioed logic. Power dissipation of the circuit before and after using FPL is found by simulation and a 13%increase is reported in Table I.

The key part of FCSL design is in the third stage. As shown in Fig. 3, this stage is composed of four different unit logic blocks. Although, the logic functions of these blocks are different, all of them serve as carry-merge blocks. Number of the sides of each block is an indication of the number of carries that a block in effect merges, for example  is a carry-merge circuit that merges three carries. It is interesting to note that the first three blocks of this stage are just the same as the last three blocks ().

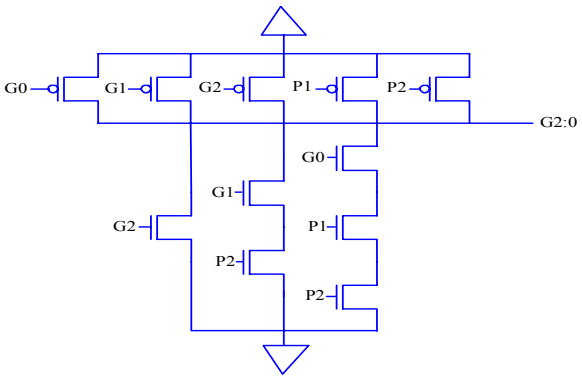


Fig 5. Circuit of a 3-bit FPL carry-merge.

TABLE II.
COMPARISON BETWEEN THE POWER DISSIPATION OF 3-BIT CONDITIONAL ADDER AND 4-BIT CONDITIONAL ADDER.

Design	Power Dissipation
Four 3-bit conditional adder	0.849mW
Three 4-bit conditional adder	1.37mw

One immediate advantage of the FCSL architecture, as obviates from Fig. 3, is the reduced maximum fanout of each block. As in the Sparse-Tree structure, one of the CM blocks is loaded with four gates. However, in FCSL the maximum load of each block is three. This reduction in the fanout is very useful in scalability of the circuit in the future generations of CMOS technologies.

In the first three blocks of this stage, all of the first three caries are generated independently, enabling generation of the first three carries in only three stages. This allows use of a conditional structure for the last three caries, i.e. the signals *G23:21, P23:21, G26:21, P26:21, G29:21* and *P29:21* are generated without having the results of the previous caries. In the next stage, the C23, C26 and C29 signals are generated for both assumptions of *C20:0 = 0* and *C20:0 = 1*. As a result, in this stage, two sets of C23, C26 and C29 signals are generated. For instance, *C23* equals *G23:21* in case *C20 = 0* and equals *G23:21 + P23:21* if *C20 = 1*.

Using such conditional architecture enables us to generate more carries because of using fewer number of logic stages in comparison with the Sparse-Tree Adder design. This allows our circuit to operate at higher frequencies.

B. Sum-Generate circuit in FCSL

We can see the structure of the 3-bit sum-generate circuit in Figure 6. All of the sub-circuits of the system have been designed using static logic. Using no dynamic structure in 3-bit conditional sum-generate circuit, helps it to make a cascade with the carry-generate circuit without needing an extra clock signal.

Using 3-bit conditional sum-generate circuit in FCSL also has a power advantage over the 4-bit version used in Sparse-Tree design. Avoiding the complexity of the last stage in the 4-bit structure is responsible for this power reduction. The power consumption of the two cases are compared through simulations and the results are summarized in Table II.

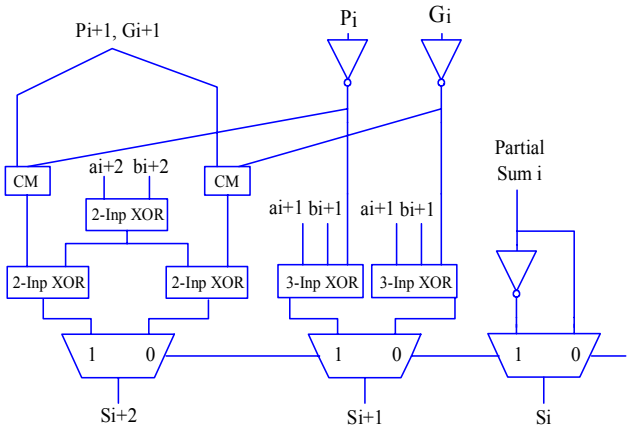


Fig 6. 3-bit conditional adder.

One interesting design choice made in sum-generate circuit is application of two 2-input XOR gates replacing 3-input XOR gates in the last stage as shown in Fig. 6. Using two cascaded XOR gates, *Ai* and *Bi* are XORed in the first round while the carry of the previous stage is being generated. This carry is XORed in the next step therefore reducing the critical delay of the path. Using 2-input XOR gates is only necessary for generating the third sum signal and for the second stage we have used a 3-input XOR gate with active load [3].

The FCSL design needs only two clock signals, one being the delayed version of the other by about 18.4 pSec. The use of one such delay stage in place of two was critical in improving the speed of the design. Details of the clock signal are shown in Fig. 7. As seen, the duration of the 1 cycle is chosen to be longer than the 0 duration, since the 1 clock cycle is where the main domino stages of the carry-generate circuit are in evaluation state and the zero cycle is just for the precharge.

In Figure 7 it is also shown how we generate the clock signal for the third stage of the adder.

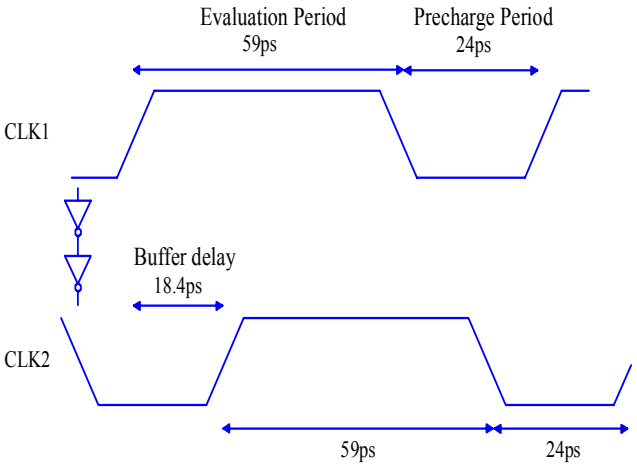


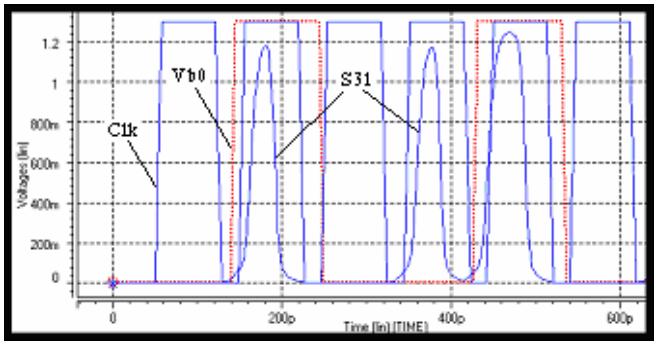
Fig 7. Details of the clock signals.

IV. SIMULATION RESULTS AND COMPARISONS

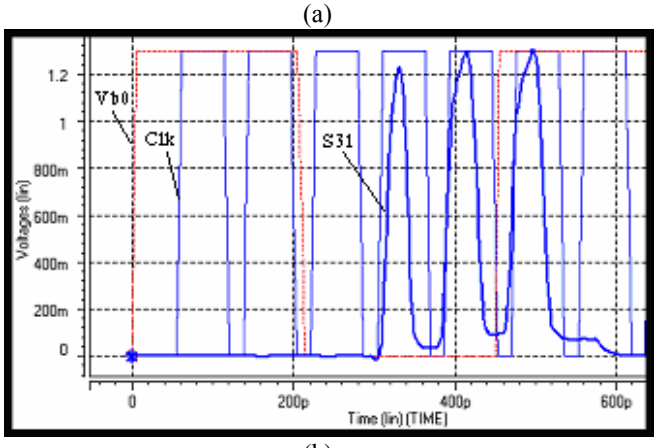
Simulations are performed to compare the maximum operating frequencies of adders with various architectures. The worst case waveforms are shown in Figure 8. The worst case is defined as the case where carry has to propagate through all the circuit stages from *A0* or *B0* to S31. We note that the highest previously reported frequency with this structure is 9 GHz [4]. We have simulated all of the compared circuits under similar conditions with BPTM [5] models. Our simulations show that for the Sparse-Tree adder, this fmax is 10.5 GHz. However, with FCSL at similar conditions, we were able to achieve 12 GHz.

In Table III, rise-time, fall-time and propagation delay for the normal Sparse-Tree adder and FCSL adder are presented. It is shown how the modifications in design have implied improvements in the delays of the FCSL.

Figure 8 (a) shows the worst case wave-forms of the Sparse-Tree adder where all *Vai* signals are set to 1 and all *Vbi* signals are set to 0 and a square wave is applied to *Vb0*. As a result, *S31* must oscillate between 1 and 0. The frequency cannot be increased any more than 10.5GHz with the constraint that the output reaches 90% of *Vdd*. Three waveforms are shown in the figure: The 12 GHz input clock, *Vb0* input and the *S31* signal as the output. Increasing the frequency causes *S31* to be unresolvable by the subsequent stage.



(a)



(b)

Fig 8. Wave-forms of *Vb0*, *S31* and clock signals. (a) Traditional Sparse-Tree adder. (b) FCSL architecture.

TABLE III.
RISE-TIME, FALL-TIME AND PROPAGATION DELAY OF FCSL AND TRADITIONAL SPARSE-TREE ADDER.

	FCSL	Sparse-Tree
Rise-time	17.7ps	22ps
Fall-time	15.35ps	14.2ps
Propagation delay	84.6ps	92.06ps

TABLE IV.
COMPARISONS BETWEEN DIFFERENT TYPES OF ADDERS.

Adder Type	Avg Power	Max Freq
FCSL	1.925 mW/GHz	12 GHz
Sparse-Tree	1.47 mW/GHz	10.5 GHz
Low-Voltage Swing	2.58 mW/GHz	10 GHz

In Figure 8 (b) the worst case waveforms of FCSL at 12 GHz are shown.

The FCSL and Sparse-Tree adders are also compared with the Low-Voltage Swing adder structure [6] regarding speed and power and the results are summarized in Table IV.

V. SUMMARY AND CONCLUSIONS

We have proposed a new structure called Fast Conditional Sparse-Tree Logic (FCSL) for a 32-bit adder. The adder is based on the Sparse-Tree adder structure but it comes with new improvements, which enable the design to operate at higher frequencies. We have compared this new structure with the conventional Sparse-Tree adder and also the Low-Voltage Swing adder to show the speed improvements of the proposed design. Simulations show that FCSL adder can operate at 12 GHz improving the speed over the conventional Sparse-Tree adder by 14%.

VI. REFERENCES

[1] S. Mathew, M. Anders, K. Krishnamurthy and S. Borkar, "A 4-GHz 130nm address generation unit with 32-bit sparse-tree adder core," *IEEE J. Solid-State Circuit*, vol. 38, pp. 689-695, May 2003.

[2] J. Kim, K. Lee, H.-J. Yoo, "A 372ps 64-bit Adder using Fast Pull-up Logic in 0.18-µm CMOS," *IEEE International Symposium on Circuit and Systems*, pp. 13-16, May 2006.

[3] K. Martin, *Digital Integrated Circuit Design.* New York: Oxford University Press, pp. 384, 2000.

[4] S. Wijeratne, et al, "A 9GHz 65nm Intel Pentium 4 processor integer execution core," *ISSCC Dig. Tech. Papers*, pp. 110-111, Feb 2006.

[5] Berkley Predictive Technology Model home page, 2005,[online:] www.eas.asu.edu/~ptm/

[6] F. Kashfi and S. M. Fakhraie, "Implementation of a high-speed low-power 32-bit adder in 70nm technology," *IEEE International symposium on Circuit and Systems,* pp. 9-12, May 2006.