An enhanced low-power high-speed adder for error-tolerant application

Zhu, Ning; Goh, Wang Ling; Yeo, Kiat Seng

2009

Zhu, N., Goh, W. L., & Yeo, K. S. (2009). An enhanced low-power high-speed adder for error-tolerant application. Proceedings of the 12th International Symposium on Integrated Circuits, (pp.69-72) Singapore.

https://hdl.handle.net/10356/93564

An Enhanced Low-Power High-Speed Adder For Error-Tolerant Application

Ning Zhu *, Wang Ling Goh, and Kiat Seng Yeo
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore
*e-mail: zhuning@ntu.edu.sg

***Abstract*—The occurrence of errors are inevitable in modern VLSI technology and to overcome all possible errors is an expensive task. It not only consumes a lot of power but degrades the speed performance. By adopting an emerging concept in VLSI design and test—Error- Tolerance (ET), we managed to develop a novel Error-Tolerant Adder which we named the Type II (ETAII). The circuit to some extent is able to ease the strict restriction on accuracy to achieve tremendous improvements in both the power consumption and speed performance. When compared to its conventional counterparts, the proposed ETAII is able to achieve more than 60% improvement in the Power-Delay Product (PDP). The proposed ETAII is an enhancement of our earlier design, the ETAI, which has problem adding small number inputs.**

***Index Terms*—Adders, error-tolerance, high speed integrated circuits, low power design**

I. Introduction

It is assumed that a usable circuit/system should function perfectly in conventional digital VLSI design, to always provide definite and accurate results. However, in our non-digital worldly requests, such ideal operations are seldom needed. “Analog computation” that yields “good enough” results instead of totally accurate results [1] may in fact be acceptable. In fact, for many digital systems, the data they process have already contained errors. In applications such as a communication system, the analog signal coming from outside world must first be sampled before we can convert it to digital data at the front end of the system. The digital data is then processed and transmitted in a noisy channel before being converted back to the analog signal at the back end. During this process, errors may occur everywhere. Furthermore, due to the advances in transistor size scaling, the previously insignificant factors such as noise and process variations are becoming important impacts in today’s digital IC design [2].

In this paper, a novel adder that we named the Error Tolerant Adder II (ETAII), based on the characteristic of digital VLSI design mentioned in paragraph is described. The concept of ETA was presented earlier as ETAI and had been accepted as a brief paper in IEEE Transaction on VLSI System. The ETAI can be illustrated via an example shown in Fig. 1.

We first split the input operands into two parts: an accurate part that includes several higher order bits and the inaccurate part that is made up of the remaining lower order bits. The length of each part need not necessary be equal. The addition process starts from the middle (joining point of the two parts) towards the two opposite directions simultaneously. In the example of Fig. 1, the two 16-bit input operands, A = “1011001110011010” (45978) and B = “0110100100010011” (26899), are divided equally into 8 bits each for the accurate and inaccurate parts.

The addition of the higher order bits (accurate part) of the input operands is performed from right to left (LSB to MSB) and normal addition method is applied. This is to preserve its correctness since the higher order bits play a more important role than the lower order bits. The lower order bits of the input operands (inaccurate part) require special addition mechanism. No carry signal will be generated or taken in at any bit position to eliminate the carry propagation path. To minimize the overall error due to the elimination of the carry chain, a special strategy is adapted, and can be described as follow: (i) check every bit position from left to right (MSB to LSB), (ii) if both input bits are “0” or different, normal one-bit addition is performed and the operation proceeds to next bit position, (iii) if both input bits are “1”, the checking process stopped and from this bit onwards, all sum bits to the right are set to “1”. The addition mechanism described can be easily understood from the example given in Fig. 1 that has a final result of “10001110010011111” (72863).

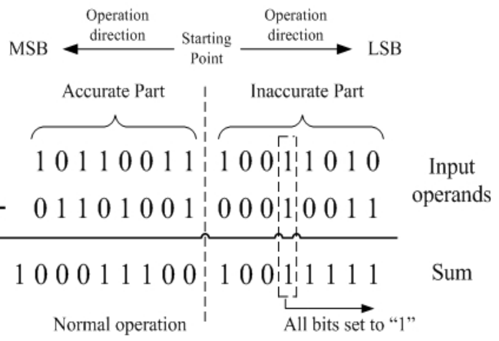


Figure 1. Error Tolerant Adder Type I (ETAI)

By eliminating the carry propagation path in the inaccurate part and simultaneously performing addition in two separate parts, the overall delay time is greatly reduced. So is the power consumption.

Of course, not all digital systems can employ the error-tolerant circuit. In some digital systems, such as a control system, the correctness of the output signals are extremely important and hence eliminating the use of error-tolerant circuit. On the other hand, since human behaviors/feelings are always error-tolerant, DSP systems such as the image processing and speech processing systems that process signals relating to human senses, i.e. sound, images, smells and touch, will be able to engage the error-tolerant circuits [3]-[5].

As mentioned in the start, ETAI has a problem of calculating small number inputs. Take for example 0000000000001111 + 0000000000001111, ETAI will yield a result of 0000000000001111 which is far from the correct value of 0000000000011110. ETA Type II (ETAII) proposed in this paper is therefore intended to solve this problem.

The rest of the paper is organized as follows. Section II proposes the addition arithmetic as well as the structure of the Error-Tolerant Adder II. Section III provides the hardware implementation. The experimental results are shown in Section IV. At last, in section V, brief statements concluding the presentation of this paper is given.

II. ERROR-TOLERANT ADDER TYPE II

Before detailing the ETAII, the definitions of the commonly used terminologies shown in this paper are given as follows:

- *Overall error (OE)–* $OE = |R_c - R_e|$, where R_e is the result obtained by the adder, and R_c denotes the correct result (all the results are represented as decimal numbers).
- *Accuracy (ACC)–* Accuracy of an adder indicates how "correct" the output is for a particular input. It is defined as: $ACC = (1 - \frac{OE}{R_c}) \times 100\%$. Its value ranges from 0% to 100%.
- *Minimum acceptable accuracy (MAA)–*Although some errors are allowed in ETA, the accuracy of an acceptable output should be "high enough". The result obtained whose accuracy is higher than the minimum acceptable accuracy (threshold) is called acceptable result.
- *Acceptance probability (AP)–*Acceptance probability is the probability that the accuracy of an adder is higher than the minimum acceptable accuracy. It can be expressed as $AP = P(ACC > MAA)$, with its value ranging from 0 to 1.

A. Proposed Addition Arithmetic for ETAII

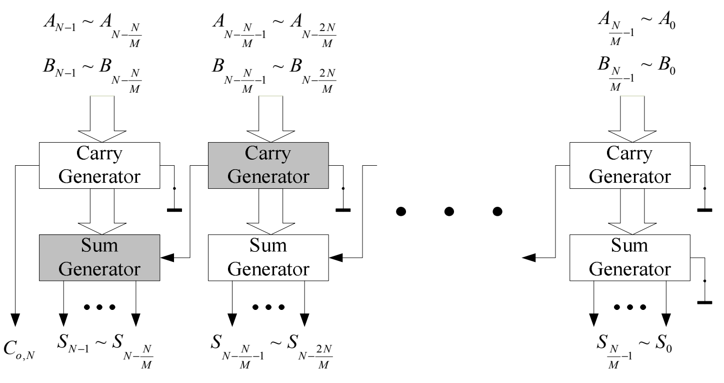
Different from ETAI, ETAII does not eliminate the entire or part of the carry propagation path. Instead, it splits the entire carry propagation path into a number of short paths and completes the carry propagations in these short paths concurrently. In this way, the speed performance of the adder can be significantly improved and with almost no degradation in its power consumption.

Consider the addition process of binary number, the carry signal at each bit position is determined by the previous input bits. For the worst case, this carry signal is generated at the LSB and propagates along the carry chain to the current bit position. If this worst case happens, tremendous amount of time and power will be consumed along the carry propagation path. But in fact, the worst case seldom happens. For most of the cases, this carry signal can be determined by just several input bits on the right of the current bit position. Assume that the input operands are perfectly random, the probabilities of getting correct carry signal on any bit when different number of bit positions to the right are involved in determining the carry signal can be derived. Some of the results are tabulated in Table I. From this table, it can be seen that when considering only one neighboring bit position, which occurs in 75% of the cases, one can get a correct carry signal to arrive at the correct sum result; if the number of bit positions to be taken into consideration is increased from 1 bit to 4 bits, the chances of obtaining a wrong carry signal is only about 3% of all the possible cases; if the number continues to increase to 8 bits or even 12 bits, the probability of getting incorrect carry signal is reduced to about 0.2% and 0.01%, respectively.

TABLE I. PROBABILITY OF GETTING CORRECT CARRY SIGNAL WHEN DIFFERENT NUMBER OF BIT POSITIONS ARE TAKEN INTO CONSIDERATION

Number of bit considered	**1 bit**	**2 bits**	**4 bits**	**8 bits**	**12 bits**
Probability of getting correct carry signal	0.75	0.875	0.96875	0.99805	0.99988

The architecture of ETAII is depicted in Fig. 2. For an N-bit adder, it is divided into M ($M \geq 2$) blocks. Each block contains N/M bits and is consists of two separate circuitries – Carry generator and Sum generator. As the name implies, the Carry generator creates the carry-out signal. It does not take in carry signal from previous block. The Sum generator however, takes the carry-in signal from previous block to generate its sum output bits. In this manner, the carry propagation only exists between two neighboring blocks instead of lying along the entire adder structure. In other words, the longest carry propagation path of ETAII is 2N/M-bit long. The worst-case delay path is shaded in grey in Fig. 2. So, the worst-case delay of ETAII is only 2/M times of the conventional adder. And since the carry propagation path has been curtailed, the dynamic power consumption of the adder caused by the spurious switching activities is reduced.



$A_{N-1} \sim A_{N-\frac{N}{M}}$
$B_{N-1} \sim B_{N-\frac{N}{M}}$
$A_{N-\frac{N}{M}-1} \sim A_{N-\frac{2N}{M}}$
$B_{N-\frac{N}{M}-1} \sim B_{N-\frac{2N}{M}}$
$A_{\frac{N}{M}-1} \sim A_0$
$B_{\frac{N}{M}-1} \sim B_0$
Carry Generator
Carry Generator
Carry Generator
Sum Generator
Sum Generator
Sum Generator
$C_{o,N}$
$S_{N-1} \sim S_{N-\frac{N}{M}}$
$S_{N-\frac{N}{M}-1} \sim S_{N-\frac{2N}{M}}$
$S_{\frac{N}{M}-1} \sim S_0$

Figure 2. Block diagram of Type II ETA (ETAII)

B. The dividing strategy

In design of ETAII, the dividing strategy refers to the choice of the number of blocks the adder is divided into. If fewer blocks are engaged and thus more bits are contained in one block, the probability of getting correct results becomes higher while the delay path also becomes longer; on the contrary, if the adder is divided in a "finer" manner, i.e. more blocks are engaged, the speed performance can be increased while the possibility that incorrect results can occur becomes higher. Take a 32-bit ETAII for instance, the simulation results of AP under different dividing strategies and MAA's are provided in Table II. Simulations were carried out using C program and 10000 random input patterns in the range of $0 \sim 2^{32}-1$ were tested.

TABLE II. ETAII UNDER DIFFERENT DIVIDING STRATEGIES AND MAA'S

MAA (%) \ No. of Bits	N = 1	N = 2	N = 4	N = 8	N = 16
100	0.0118	0.2204	0.8306	0.9961	1.0
99	0.5238	0.7927	0.9725	0.9985	1.0
98	0.5419	0.8075	0.9795	0.9985	1.0
97	0.5534	0.8119	0.9815	0.9995	1.0
96	0.5706	0.8236	0.9819	0.9995	1.0
95	0.5864	0.8352	0.9833	1.0	1.0
94	0.5894	0.8367	0.9833	1.0	1.0
93	0.5973	0.8386	0.9838	1.0	1.0
92	0.6062	0.8401	0.9838	1.0	1.0
91	0.6155	0.8425	0.9843	1.0	1.0
90	0.6213	0.8468	0.9843	1.0	1.0

C. Relationship between the Accuracy and the Range of Input Patterns

As mentioned in the introduction, when the inputs are small numbers, the accuracy of the ETAI is poor. This problem has been overcome by the proposed ETAII. To prove its merit, simulations were performed on ETAI and ETAII, and the results obtained are tabulated in Tables III and IV. In the simulation, the number of bits in a block for ETAII was set to 4 and all simulations were carried out using C program. 10000 random input patterns were tested for each range.

From Tables III and IV, it is obvious that the accuracy performance of the adder for small input operands is significantly improved using ETAII. It can be seen that there is no significant difference in the AP values between the different input ranges using ETAII. In other words, the unacceptable results are evenly distributed over the whole input range, from 0 to $2^{32}-1$, instead of "squeezing" in the range of small numbers. However, the accuracy of ETAII for large input operands is degraded (see last column of table IV) than ETAI (last column of table III).

TABLE III. AP OF ETAI WITH DIFFERENT MAA'S AND INPUT RANGES

MAA (%) \ Input Range	$0 \sim 2^{8}-1$	$0 \sim 2^{16}-1$	$0 \sim 2^{24}-1$	$0 \sim 2^{32}-1$
100	0.0988	0.0096	0.0032	0.0025
99	0.1364	0.1491	0.5650	1.0
98	0.1897	0.2055	0.7324	1.0
97	0.2239	0.2472	0.8375	1.0
96	0.2656	0.2808	0.9014	1.0
95	0.3002	0.3124	0.9385	1.0
94	0.3344	0.3409	0.9570	1.0
93	0.3568	0.3614	0.9679	1.0
92	0.3858	0.3899	0.9754	1.0
91	0.4040	0.4140	0.9812	1.0
90	0.4266	0.4318	0.9856	1.0

TABLE IV. AP OF ETAII WITH DIFFERENT MAA'S AND INPUT RANGES

MAA (%) \ Input Range	$0 \sim 2^{8}-1$	$0 \sim 2^{16}-1$	$0 \sim 2^{24}-1$	$0 \sim 2^{32}-1$
100	0.9691	0.9090	0.8590	0.8389
99	0.9691	0.9408	0.9393	0.9699
98	0.9691	0.9414	0.9417	0.9780
97	0.9691	0.9414	0.9422	0.9824
96	0.9691	0.9439	0.9453	0.9829
95	0.9691	0.9493	0.9517	0.9829
94	0.9691	0.9545	0.9557	0.9829
93	0.9691	0.9582	0.9598	0.9829
92	0.9691	0.9612	0.9624	0.9834
91	0.9691	0.9617	0.9636	0.9834
90	0.9691	0.9638	0.9663	0.9839

D. Modified ETAII

Although ETAII achieves good performances in both power and speed, the degraded accuracy for large input operands indicated in Section C may still restrict its use. A modified structure is hence implemented to further improve the accuracy performance of ETAII. In the new design, the higher order bits should be more accurate than the lower order bits as they play a more important role in representing a number. Therefore, for the higher order bit positions, more input bits should be considered when calculating the carry signals.

Fig. 3 depicts the modified structure of the 32-bit ETAII and for convenience; the modified ETAII is named ETAIIM. In this structure, the first three carry generators are cascaded together to generate the carry signals for the two highest blocks. In this way, the carry signal for the highest block is generated by the preceding 12 bits and the carry signal for the second block is generated by the preceding 8 bits and so on. The rest of the circuit is the same as that of ETAII (see Fig. 2).

The accuracy performance of ETAIIM is presented in Table V. It is clear that the AP simulation results of ETAIIM out-performed those of ETAII (see column 4 of Table II).

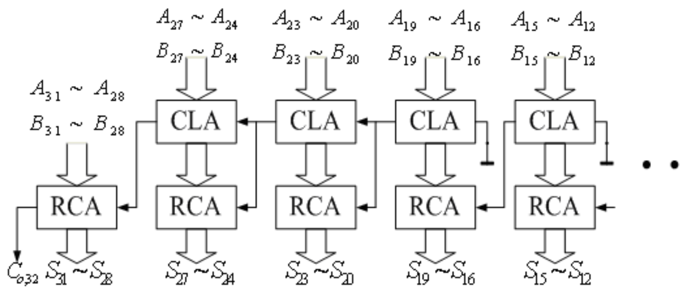


Figure 3. Structure of Modified ETAII (ETAIIM)

TABLE V. AP OF ETAIIM WITH DIFFERENT MAA'S

MAA (%)	AP	MAA (%)	AP
100	0.8854	**95**	1.0
99	0.9990	**94**	1.0
98	0.9995	**93**	1.0
97	1.0	**92**	1.0
96	1.0	**91**	1.0

III. HARDWARE IMPLEMENTATION

The hardware implementation of the ETAII uses the conventional designs. The "sum-generator" block in Fig. 2

and the "RCA" block shown in Fig. 3 are all built by cascading four conventional 28-transistor full adders, as shown in Fig. 4. The "carry generator" block of Fig. 2 and "CLA" block of Fig. 3 employed the carry generator of a 4-bit carry-look-ahead adder. Implementation of the carry generator circuit is illustrated in Fig. 5.

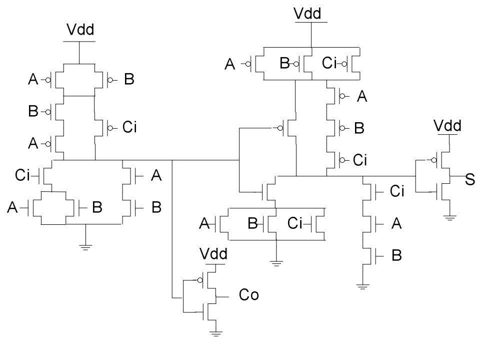


Figure 4. 28-transistor full adder

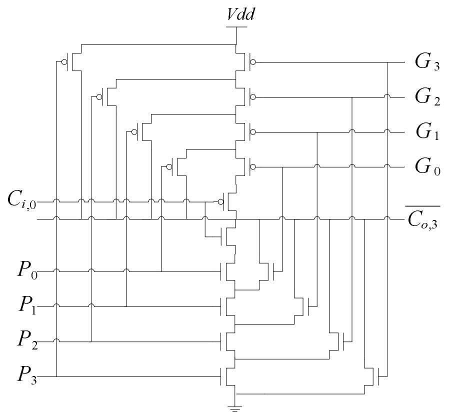


Figure 5. Implementation of 4-bit Carry Generator

IV. Simulation Results

To demonstrate the advantages of ETAII and ETAIIM, we simulated both along with four types of conventional adders and ETAI, i.e. the RCA (Ripple Carry Adder), CSK (Carry Skip Adder) [6], CSL (Carry Select Adder) [7], CLA (Carry Look Ahead Adder) [8] and ETAI (Error Tolerant Adder Type I), using HSPICE. All the circuits were implemented using Chartered Semiconductor Manufacturing Ltd's 0.18-µm CMOS process. The input frequency was set to 100 MHz and the simulation results are tabulated in Table VI.

HSPICE software was used to construct the ETAII models and the conventional adders. 100 sets of inputs were randomly created using the C program "random()" function. For each set of input, we ran the simulation for each adder and recorded the power consumption. With 100 sets of results, the average power consumption was determined. The worst case input was calculated and used to simulate the delay. The transistor count was derived directly from the HSPICE software.

Comparing the simulation results of our proposed ETAII (4-bit in one block) and ETAIIM with those of the conventional adders and ETAI, it is evident that both ETAII and ETAIIM can achieve a very small PDP that is almost the same as that of ETAI, which is of course far better than those realized using conventional adders. The transistor counts (areas) of ETAII and ETAIIM are a little larger than the ETAI; however it solved the small input problem in ETAI.

TABLE VI. Simulation Result for ETAII and ETAIIM versus Conventional Adders

Type	Power(mw)	Delay(ns)	PDP(pJ)	Transistor count
RCA	0.22	4.04	0.89	896
CSK	0.46	2.90	1.33	1728
CSL	0.60	3.06	1.84	2176
CLA	0.51	2.37	1.21	2208
ETAI	0.13	2.29	0.30	1006
ETAII	0.24	0.85	0.20	1372
ETAIIM	0.24	1.39	0.33	1372

V. Conclusion

In this paper, a Type II Error-Tolerant Adder (ETAII) that trades accuracy for significant power saving and performance improvement is proposed. Different from ETAI, it is able to carry out small number calculation well. Extensive comparisons with the conventional digital adders showed that the proposed ETAII/ETAIIM outperformed conventional adders in both the power consumption and speed performance. The potential applications of the ETA fall mainly in areas where there is no strict requirement on accuracy or where super-low power consumption and high speed performance are more important than the accuracy. One example of such applications is in the DSP application for portable devices such as cell phones and laptops.

Acknowledgment

The authors are grateful to the Temasek Lab@NTU for supporting the project.

References

[1] Melvin A. Breuer, "Let's think analog," in *Proc. of the IEEE Computer Society Annual Symposium on VLSI*, 2005.

[2] International Technology Roadmap for Semiconductors, latest edition available online at http://public.itrs.net/.

[3] Melvin A. Breuer and Haiyang Zhu, "Error-tolerance and multi-media," in *Proc. of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2006.

[4] K. J. Lee, T. Y. Hsieh and M. A. Breuer, "A novel testing methodology based on error-rate to support error-tolerance," in *Proc. of International Test Conference*, pp. 1136–1144, 2005.

[5] I. S. Chong and A. Ortega, "Hardware testing for error tolerant multimedia compression based on linear transforms," *Defect and Fault Tolerance in VLSI Systems Symp.*, 2005.

[6] M. Lehman and N. Burla, "Skip techniques for high-speed carry propagation in binary arithmetic units," *IRE Trans. on Electronic Computers*, vol. EC-10, pp. 691–698, December 1962.

[7] O. Bedrij, "Carry select adder," *IRE Trans. on Electronic Computers*, vol. EC-11, pp. 340–346, 1962.

[8] O. MacSorley, "High speed arithmetic in binary computers," *IRE proceedings vol*. 49, pp. 67–91, 1961.