A Variable Latency Ling Adder Based on Brent-Kung Parallel-Prefix Topology

Pixia Guo, Guangjun Xie, Xiaoyue Chen, Jie Han, Yongqiang Zhang

***Abstract*—For emerging applications, such as deep learning, design complexity can be reduced to lower hardware costs by releasing the strict requirement of computing accuracy. In this paper, a variable latency Ling adder (VLLA) with reduced propagation delay is proposed to produce exact results in most cases through a speculative operation. When an error is detected, the error detection and correction modules are activated to generate exact results. All circuits are synthesized in the Synopsys Design Compiler with a TSMC 65 nm standard cell library and evaluated for the error probability in MATLAB. Experimental results show that the proposed VLLA achieves reductions in power and smallest average delay by up to 17% and 23%, respectively, compared to the original exact Ling adder (LA). In addition, the proposed VLLA achieves reductions of 8%, 18%, 11.4%, and 7.6% in power, smallest average delay, area average-delay product, and power average-delay product, compared to an existing variable latency adder (VLA).**

Keywords—Approximate computing, parallel-prefix adder, Ling adder, error correction

I. INTRODUCTION

Adders are basic building blocks for arithmetic operations [1]. As the scope of arithmetic applications continues to expand, high operating speeds are demanded. As a type of fast parallel prefix adder [2], the Brent-Kung adder has a simpler structure than adders designed using other topologies [3].

Recently, parallel prefix Ling adders based on Sklansky and Kogge-Stone topologies have been designed to offer reduced delays and fanout requirements, compared to conventional parallel prefix adders [4, 5, 6]. In [7], the variable latency adder (VLA) uses speculation to reduce the average delay and achieve a low error rate through efficient error detection and correction techniques. A novel Brent-Kung prefix-processing topology has been considered to improve performance over a traditional one [8].

To maintain high speed and effectively reduce energy consumption, a 32-bit variable latency Ling adder (VLLA) is proposed and analyzed in this work. It consists of error detection and correction modules, and a speculative Ling adder (LA). The speculative LA is truncated in specific rows to lower hardware costs according to the distribution patterns of inputs, while the error detection and correction modules are designed to recover the induced errors for different input patterns. Synthesized results demonstrate that the proposed VLLA achieves significant reductions in power and delay compared to the exact LA and existing VLA. The main contributions of this work are summarized as follows: (1) A speculative LA is proposed, by deleting some intermediate rows of the prefix-processing unit of an exact LA. (2) Error detection and correction modules are used to maintain computing accuracy.

The rest of the paper is organized as follows. Section II reviews the parallel prefix adder and Brent-Kung topology. Section III describes the circuit structure of the proposed VLLA. Error probability analysis and hardware overheads are presented in Section IV. Section V concludes this paper.

II. BACKGROUND

A parallel prefix adder can be divided into three units, including pre-processing, prefix-processing, and post-processing [9]. Given an n-bit augend $A=a_{n-1}a_{n-2}\ldots a_0$ and an n-bit addend $B=b_{n-1}b_{n-2}\ldots b_0$, it generates an i^{th} carry c_i and an n-bit sum $S=s_{n-1}s_{n-2}\ldots s_0$.

In the pre-processing unit, the generate g_i and propagate p_i are given as:

$$\begin{aligned} g_i &= a_i b_i \\ p_i &= a_i \oplus b_i \end{aligned}, \tag{1}$$

where $0\le i\le n-1$, and $\oplus$ denotes the XOR operation.

In the prefix-processing unit, the carry c_{i+1} is computed using the generate g_i and propagate p_i as:

$$c_{i+1} = g_i + p_i g_{i-1} + \cdots + p_i p_{i-1} \cdots p_1 g_0 . \tag{2}$$

The parallel prefix adders use an associative operator $\square$ to associate the pairs of generate and propagate, defined as:

$$(g_{i+1}, p_{i+1}) \square (g_i, p_i) = (g_{i+1} + p_{i+1} g_i, p_{i+1} p_i) . \tag{3}$$

Thus, the carry c_{i+1} can be represented as:

$$(c_{i+1}, \sim) = (g_i, p_i) \square (g_{i-1}, p_{i-1}) \square \cdots \square (g_0, p_0) . \tag{4}$$

In the post-processing unit, the sum s_i is given by:

$$s_i = p_i \oplus c_i . \tag{5}$$

As shown in Fig. 1, the Brent-Kung parallel prefix processing unit is one of the connection topologies through black nodes ● and white nodes ○. The black node ● represents the associative operator $\square$ above, while the white node ○ is a buffering component.

III. THE PROPOSED VARIABLE LATENCY LING ADDER

As shown in Fig. 2, a 32-bit VLLA can be divided into five units, including pre-processing, prefix-processing, post-

P. Guo, G. Xie, X. Chen and Y. Zhang are with the School of Microelectronics, Hefei University of Technology, Hefei 230009, China (e-mail: 1739954382@qq.com; gjxie8005@hfut.edu.cn; 1617090911@qq.com; ahzhangyq@hfut.edu.cn)

J. Han is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada (e-mail: jhan8@ualberta.ca)

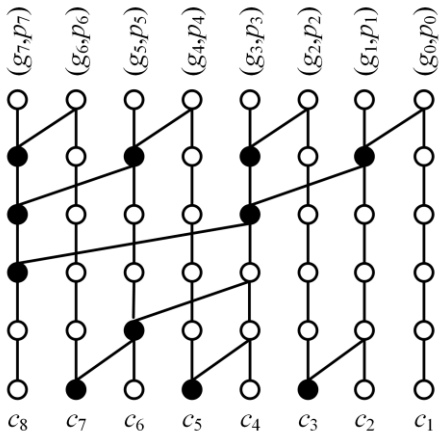


Figure 1. An 8-bit Brent-Kung prefix-processing unit.

processing, error detection, and error correction. If the error detection signal E_s is false, the VLLA needs one clock cycle to produce the exact results, or, it produces the exact results through the error correction module in the next clock cycle. Thus, the average adder delay T_{avg} is

$$T_{avg} = 2P_{Es} \cdot T_{clk} + (1 - P_{Es}) \cdot T_{clk}, \tag{6}$$

where T_{clk} is the clock period and P_{Es} is the probability that the error detection signal E_s is true.

A. Pre-Processing

Given an augend $A=a_{31}a_{30}\ldots a_0$ and an addend $B=b_{31}b_{30}\ldots b_0$, the proposed 32-bit VLLA produces a carry c_{32} and a 32-bit sum $S=s_{31}s_{30}\ldots s_0$. In the pre-processing stage, the half-sum d_i, generate g_i, and propagate p_i are generated [10], as defined as:

$$\begin{aligned} d_i &= a_i \oplus b_i \\ g_i &= a_i b_i \\ p_i &= a_i + b_i \end{aligned}, \tag{7}$$

where $0 \le i \le 31$.

The proposed VLLA produces the intermediate generate G_i and propagate P_i to reduce the number of fanouts in the parallel prefix processing unit. The intermediate generate G_i and propagate P_i are defined as:

$$\begin{aligned} G_i &= g_i + g_{i-1} \\ P_i &= p_i p_{i-1} \end{aligned}. \tag{8}$$

B. Prefix-Processing

In contrast to the general parallel prefix adders, a LA generates the Ling carry H_i instead of generating the carry c_{i+1} in the prefix-processing unit [10], as:

$$H_i = c_{i+1} + c_i . \tag{9}$$

According to (3), (7), (8), and (9), the Ling carry H_i can be rewritten as:

$$\begin{aligned} (H_{2k}, \sim) &= (G_{2k}, P_{2k-1}) \square (G_{2k-2}, P_{2k-3}) \square \cdots \square (G_0, P_{-1}) \\ (H_{2k+1}, \sim) &= (G_{2k+1}, P_{2k}) \square (G_{2k-1}, P_{2k-2}) \square \cdots \square (G_1, P_0) \end{aligned}, \tag{10}$$

where $0 \le k < 16$, and it is defined that $P_{-1}=0$.

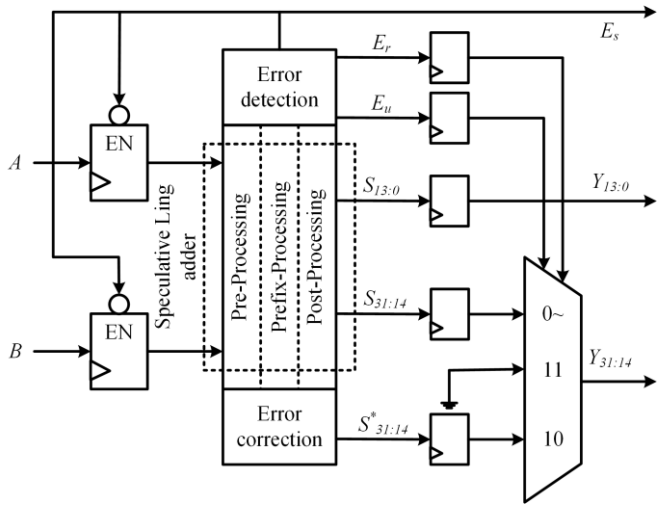


Figure 2. The proposed 32-bit variable latency Ling adder.

The prefix-processing unit of an exact LA with Brent-Kung topology has a total of 8 rows in Fig. 3(a). The proposed prefix-processing unit breaks the carry chain by deleting the middle three rows of the prefix-processing unit in the exact LA, surrounded by a dashed box, as shown in Fig. 3(b). If only one row is removed, the smallest T_{avg} is too large; if five rows are removed, the error correction module will become more complicated, requiring excessive area and power. We observe that the lower 14 bits are exact and the higher 18 bits are speculative in Fig 3(b). The Ling carries H^* of the higher 18 bits can be obtained by deleting some specific intermediate generate and propagate pairs from (10). For example, the expressions for the 14th and 15th Ling carries are

$$\begin{aligned} (H^*_{14}, \sim) &= (G_{14}, P_{13}) \square (G_{12}, P_{11}) \square (G_{10}, P_9) \square (G_8, P_7) \\ (H^*_{15}, \sim) &= (G_{15}, P_{14}) \square (G_{13}, P_{12}) \square (G_{11}, P_{10}) \square (G_9, P_8) \end{aligned}. \tag{11}$$

C. Post-Processing

For the prefix-processing unit, the carry c_{32} and 32-bit sum S are generated. Due to (7) and (9), the sum S for each bit can be further rewritten as:

$$s_i = d_i \oplus (p_{i-1} H_{i-1}), \tag{12}$$

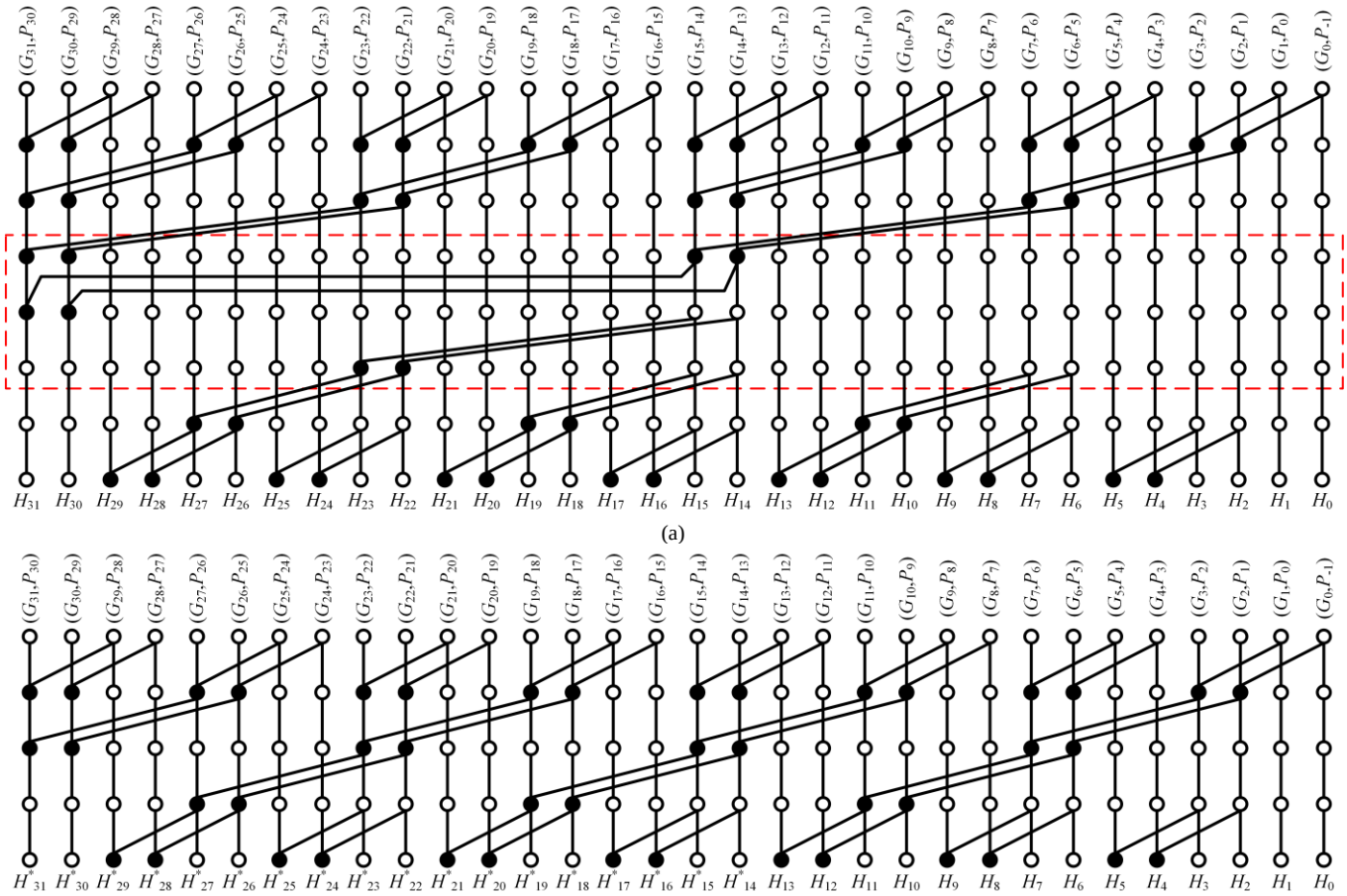
and the carry c_{32} can be written as:

$$c_{32} = p_{31} H_{31} . \tag{13}$$

D. Error Detection

To simplify the expressions of error detection signals, several definitions are proposed as in (14) and (15).

$$\begin{aligned} &G_{[2k:2(k-j)]} = \\ &\begin{cases} G_{2k} & for: j = 0 \\ G_{[2k:2(k-m)]} + P_{[2k-1:2(k-m)-1]} G_{[2(k-m-1):2(k-j)]} & otherwise \end{cases} \\ &P_{[2k:2(k-j)]} = \\ &\begin{cases} P_{2k} & for: j = 0 \\ P_{[2k:2(k-m)]} P_{[2(k-m-1):2(k-j)]} & otherwise \end{cases} \end{aligned}. \tag{14}$$



(a)

(b)

Figure 3. Prefix-processing unit. (a) LA. (b) The proposed 32-bit VLLA.

$$G_{[2k+1:2(k-j)+1]} = \begin{cases} G_{2k+1} & for: j=0 \\ G_{[2k+1:2(k-m)+1]} + P_{[2k:2(k-m)]} G_{[2(k-m)-1:2(k-j)+1]} & otherwise \end{cases}, \quad (15)$$

$$P_{[2k+1:2(k-j)+1]} = \begin{cases} P_{2k+1} & for: j=0 \\ P_{[2k+1:2(k-m)+1]} P_{[2(k-m)-1:2(k-j)+1]} & otherwise \end{cases}$$

where $0 \le m < j \le k < 16$.

1) Unsigned Operands

It is acceptable to approximate a real case to conform to a uniform distribution if the inputs of an adder are unsigned numbers. The maximum carry chain length for 32-bit inputs is mostly less than 7 [7]. Hence, the smallest carry chain length for the proposed VLLA is set to 7 in Fig. 3(b), which is longer than the maximum one for inputs in most cases. Thus, in most cases, the proposed VLLA produces exact results. However, the proposed VLLA is likely to generate errors, if the maximum carry chain length exceeds 7. For these cases, the error detection signal for each sum s_i can be obtained from (10), (11), and (12) as:

$$e_{2k} = \begin{cases} 0 & for: 0 \le k < 7 \\ p_{2k} H_{2k} & for: 7 \le k < 16 \end{cases}$$
$$e_{2k+1} = \begin{cases} 0 & for: 0 \le k < 7 \\ p_{2k+1} H_{2k+1} & for: 7 \le k < 16 \end{cases}. \quad (16)$$

Thus, the error detection signal E_u for the proposed speculative LA can be expressed as:

$$E_u = \sum_{i=14}^{31} p_i H_i \,. \quad (17)$$

According to (14), (15), and (17), the error detection signal E_u can be further simplified to:

$$E_u = p_{31} P_{[30:24]} G_{[23:17]} + p_{23} P_{[22:16]} G_{[15:9]} + p_{15} P_{[14:8]} G_{[7:1]} + p_{30} P_{[29:23]} G_{[22:16]} + p_{22} P_{[21:15]} G_{[14:8]} + p_{14} P_{[13:7]} G_{[6:0]} \,. \quad (18)$$

2) Signed Operands

If the inputs are signed numbers, it is more accurate to approximate a real case to conform to a Gaussian distribution. The maximum carry chain length for 32-bit inputs is concentrated not only less than 7 but also larger than 25 [7].

If the carry chain length is less than 7, the proposed VLLA produces exact results. The maximum carry chain length exceeds 25 if two inputs with opposite signs are added and the generated result is a positive number. For example, if adding the two numbers 29 and -26, the carry chain length is 29. In this case, the proposed VLLA has a large error rate and average adder delay T_{avg}. Moreover, the half-sums $d_3 \sim d_{31}$ are all 1 and the sums $s_3 \sim s_{31}$ are all 0. As a result, the error detection signal E_u can be optimized to E_r, as:

$$E_r = d_{31} d_{30} \cdots d_{14} E_u \,. \quad (19)$$

TABLE I. THE ERROR PROBABILITY VALUES OF ADDERS

Adders	Error probability (%)		
	r=100%	*r=50%, σ=256*	*r=50%, σ=30000*
VLLA	0.77	0.40	9.21
VLA	0.42	0.19	0.25

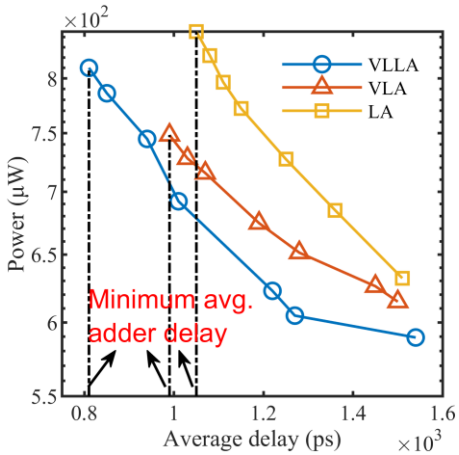


Figure 4. The power of the proposed VLLA, VLA, and LA, as a function of timing constraint.

According to (18), (19) can be rewritten to

$$E_r = d_{31}d_{30}\cdots d_{14}p_{13}G_{[13:1]}\,. \tag{20}$$

However, if the carry chain length is larger than 7 and less than 25, the proposed VLLA produces errors. These errors can be detected through the error detection signal E_s as:

$$E_s = \overline{d_{31}d_{30}\cdots d_{14}}E_u\,. \tag{21}$$

E. Error Correction

In the error correction unit, if E_u=1 and E_r=1, the sums are directly grounded. If E_u=1 and E_r=0, the three rows of the prefix-processing unit removed in Fig. 3(b) are complemented in the error correction unit to obtain the exact results.

IV. ERROR ANALYSIS AND HARDWARE OVERHEADS

VLA, LA, and the proposed VLLA are designed in Verilog HDL language and synthesized in the Synopsys Design Compiler with a TSMC 65 nm standard cell library. The analysis of the error probability (P_{Es}) for the proposed VLLA and existing VLA is implemented through MATLAB.

A. Error Analysis

The error probability is analyzed using Monte Carlo simulations, with a 2% relative error and a 99% confidence level. To obtain the input distributions of the real workloads in 2's complement, software programs running on a physical machine are traced in [11]. Three mathematical models are used to approximate the inputs in the real workloads as follows: (1) The inputs are taken from a uniform distribution; (2) Half of the inputs are taken from a uniform distribution and half from a Gaussian distribution with a standard deviation σ=256; (3) Half of the inputs are taken from a uniform distribution and half from a Gaussian distribution with σ=30000. The first distribution means that the inputs represent unsigned numbers in the workloads. The second distribution means that more operations are between small signed numbers, which is a frequent occurrence in practical workloads [11]. The third distribution means that the inputs represent signed numbers in the workloads.

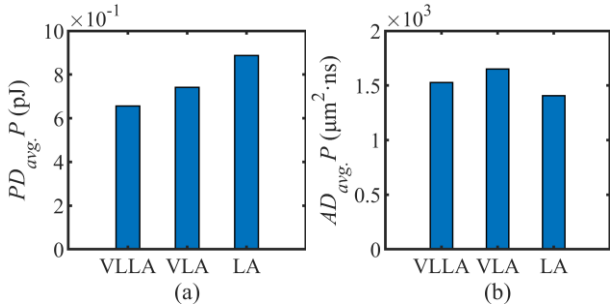


Figure 5. The circuit measurements of the proposed VLLA, VLA, and LA. (a) Power Average-Delay product. (b) Area Average-Delay product.

The error probabilities of the proposed and existing designs are compared in TABLE I, where r represents the percentage of the inputs coming from a uniform distribution. For example, r=100% means that the inputs are taken from a uniform distribution in the range [0, 2^{32}-1]. It can be found that the error probabilities are less than 10% for both the proposed VLLA and existing VLA, for three distributions [7]. It can also be found that the error probability is the smallest compared to the other two distributions if the inputs come from the second distribution. It shows that the proposed VLLA can cope well with cases where more inputs are small signed numbers in the workloads.

B. Hardware Overheads

The inputs following the second mathematical distribution are used to evaluate the performance of the proposed VLLA. Thus, the average adder delay T_{avg} is computed using the error probability P_{Es} with the second distribution, according to (6).

In Fig. 4, the hardware costs are compared between the VLLA, VLA, and LA. The power dissipation of three circuits is measured at different T_{avg}, by adjusting the clock period T_{clk}. It can be found that the smallest T_{avg} for the proposed VLLA is the smallest at 0.81 ns, the next is the existing VLA, and then the exact LA. The smallest T_{avg} of the proposed VLLA is 18% and 23%, respectively, less than those of VLA and LA. If the average adder delay T_{avg} is selected as a fixed value, it can be found that the power of the proposed VLLA is the lowest, the next is VLA, and then LA. The power of the proposed VLLA is 8% and 17%, respectively, less than those of VLA and LA.

To further compare the performance of the three circuits at the smallest T_{avg}. In Fig. 5, the area average-delay product ($AD_{avg.}P$) and power average-delay product ($PD_{avg.}P$) are shown.

The $AD_{avg.}P$ is given as:

$$AD_{avg.}P = area \cdot T_{avg}\,, \tag{22}$$

where T_{avg} is the average adder delay.

The $PD_{avg.}P$ is defined as:

$$PD_{avg.}P = power \cdot T_{avg}\,. \tag{23}$$

As shown in Fig. 5, the $PD_{avg.}P$ of the proposed VLLA is reduced by 11.4% and 26% compared to those of VLA and LA, while the $AD_{avg.}P$ is reduced by 7.6% and 8% in comparison.

V. CONCLUSION

In this paper, a VLLA based on the Brent-Kung parallel prefix topology is proposed. Its delay and power are reduced by using a speculative LA and error detection and correction modules. Simulation results show that the proposed adder reduces the smallest T_{avg} and power by 23% and 17%, compared to the original exact LA. The proposed VLLA achieves reductions of 8%, 18%,11.4%, and 7.6% in power, the smallest T_{avg}, $PD_{avg.}P$, and $AD_{avg.}P$, respectively, compared to an existing VLA.

REFERENCES

[1] K. Shilpa, M. Shwetha, B. Geetha, D. Lohitha, Navya, and N. Pramod, "Performance analysis of parallel prefix adder for datapath VLSI design," in Proceedings of the 2018 Second International Conference on Inventive Communication and Computational Technologies, Coimbatore, India, 2018, pp. 1552-1555.

[2] G. Thakur, H. Sohal, and S. Jain, "Design and analysis of high-speed parallel prefix adder for digital circuit design applications," in International Conference on Computational Performance Evaluation (Compe-2020), Shillong, India, 2020, pp. 95-100.

[3] M. Veena, P. Akkamanchi, and V. Uttarkar, "Low power high speed brent kung adder using spst," in IEEE 3rd Global Conference for Advancement in Technology (GCAT), 2022, pp. 1-6.

[4] G. Dimitrakopoulos, and D. Nikolos, "High-speed parallel-prefix VLSI ling adders," *IEEE Trans. Comput.*, vol. 54, no. 2, pp. 225-231, Feb. 2005.

[5] T. Ene, and J. Stine, "Point-targeted sparseness and ling transforms on parallel prefix adder trees," in IEEE 29th Symposium on Computer Arithmetic (ARITH), Lyon, France, 2022, pp. 68-75.

[6] C. Simsek, and K. Turk, "Hardware optimization for belief propagation polar code decoder with early stopping criteria using high-speed parallel-prefix ling adder." pp. 182-185.

[7] D. Esposito, D. De Caro, and A. G. M. Strollo, "Variable latency speculative parallel prefix adders for unsigned and signed operands," *IEEE Trans. Circuits Syst. I-Regul. Pap.*, vol. 63, no. 8, pp. 1200-1209, Aug. 2016.

[8] G. Thakur, H. Sohal, and S. Jain, "FPGA-based parallel prefix speculative adder for fast computation application," in Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 2020, pp. 206-210.

[9] I. Brzozowski, "Software tool aiding analysis and design of low-power parallel prefix adders," in 28th International Conference on Mixed Design of Integrated Circuits and System, Lodz, Poland, 2021, pp. 141-146.

[10] H. Ling, "High-speed binary adder," *IBM J. Res. Dev.,* vol. 25, no. 3, pp. 156-166, May 1981.

[11] A. Cilardo, "A new speculative addition architecture suitable for two's complement operations," in Design, Automation & Test in Europe Conference & Exhibition, Nice, France, 2009, pp. 664-669.