IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

VOL. E98-A NO. 4
APRIL 2015

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

IEICE TRANS. FUNDAMENTALS, VOL.E98–A, NO.4 APRIL 2015

PAPER

Multistage Function Speculation Adders*

Yinan SUN[†], ***Nonmember***, **Yongpan LIU**[†a)], ***Member***, **Zhibo WANG**[†], ***and*** **Huazhong YANG**[†], ***Nonmembers***

SUMMARY Function speculation design with error recovery mechanisms is quite promising due to its high performance and low area overhead. Previous work has focused on two-stage function speculation and thus lacks a systematic way to address the challenge of the multistage function speculation approach. This paper proposes a multistage function speculation with adaptive predictors and applies it in a novel adder. We deduced the analytical performance and area models for the design and validated them in our experiments. Based on those models, a general methodology is presented to guide design optimization. Both analytical proofs and experimental results on the fabricated chips show that the proposed adder's delay and area have a logarithmic and linear relationship with its bit number, respectively. Compared with the DesignWare IP, the proposed adder provides the same performance with 6–17% area reduction under different bit lengths.

key words: *multistage function speculation, variable latency adder, design methodology*

1. Introduction

Binary adders are one of the most frequently used arithmetic units, and are widely used in other advanced design units, such as multipliers and dividers. Previous work [2] has given the delay–area relationship in traditional adders, which indicates that no adders with traditional architectures can be implemented with sub-logarithmic delay. However, almost correct variable latency adders [3] can provide much better performance and energy efficient solutions. This is due to the fact that the worst-case design styles in traditional adders will lead to larger design margins, while variable latency adders permit the average-case design style and better performance. When a certain level of calculation error is acceptable — as in application areas such as data mining and machine learning — the variable latency adders can provide even better performance and smaller area.

Among the emerging variable latency adders, an interesting approach is function speculation, where only part of input vectors causing timing violations are considered in the approximate function. This speculation approach is more area-efficient and easy to find the speculation function by covering only parts of the problematic input vectors. Since speculative architectures may sometimes predict wrong logic values, error detection and recovery mechanisms are needed for correct operations, as the previous works did in time-domain speculation [4], [5] or at system level [6]. The penalty for the approximation is that the circuit needs more clock cycles to recover from an incorrect prediction. It is obvious that the gain is positive as long as the incorrect predict ratio is below a certain threshold.

Previous works have studied how to use function speculation in circuit design. In an asynchronous design, Nowick [7] presented a low-latency adder with speculative completion. In synchronous circuits, Lu [8] proposed a method to replace a complete logic function with a simplified approximation circuit that mimics the original one. These techniques can effectively increase a processor's clock frequency. Chen et al. [9] proposed a variable-latency adder to overcome the negative bias temperature instability by automatically shifting data capturing clock edge on critical timing paths. Verma et al. [3] proposed a very fast adder under an extreme low function speculation error ratio (0.01%) with error detection units. Baneres et al. [10] proposed a practical approach to synthesize two-stage variable latency circuits. Liu et al. [1] and Barrio et al. [11], [12] proposed a multistage speculation adder, deduced and validated the adder's throughput model of multi-cycle behavior. Cilardo et al. [13] noticed the effort of the long carry chain in two's complement addition in Euclid's greatest common divisor algorithm and encryption algorithms.

Although these works have provided various approaches for specific designs using function speculation, a systematical way to design multistage speculation adders still lacks. The main challenges include theoretical analysis and design optimization for multistage function speculation circuit.

In this work, we systematically solve design challenges in multistage function speculation adders under various input patterns. Our contributions are listed as below:

1. We propose a multistage function speculation structure with adaptive predictors, and apply them in a novel adder. The adder could be configured with different predictors under different inputs. We deduce the analytical throughput, and area models for the design and validate them in our experiments.
2. Based on the analytical models, we show the necessary condition to acquire the optimal performance for the multistage function speculation adder under various distributions. By giving out the average-case cycle expression of the optimal design under independent uniform distribution, we show that there are two optimal regions in the design space and verify those observa-

Manuscript received October 2, 2014.
Manuscript revised November 28, 2014.
[†]The authors are with the Department of Electronic Engineering, Tsinghua University, 100084, Beijing, P.R. China.
[*]This paper was partial presented at ISQED 2010 [1].
a) E-mail: ypliu@tsinghua.edu.cn
DOI: 10.1587/transfun.E98.A.954

tions by experiments.

3. We prove that the proposed adder's delay and area has a logarithmic and linear relationship with its bit number, respectively, which is validated by the experimental results on the fabricated chips. Compared with the DesignWare IP, the proposed adder provides the same performance with 6–17% area reductions under different bit lengths.

The rest of the paper is organized as follows. Section 2 describes the motivation of our work. We present the architecture of the proposed multistage function speculation adder in Sect. 3. Sections 4, 5 and Sect. 6 describe the performance analysis and design methodology of the proposed adder for independent uniform input distribution and application specific distribution. Section 7 presents experimental results. We conclude the paper in Sect. 8.

2. Motivation

This section provides the motivation of the proposed variable latency adder. We first discuss the delay characterization of a ripple carry adder to show the necessity of the average-case design style. Furthermore, the advantages and challenges in adders with multistage function speculation are detailed.

2.1 Delay Analysis on Ripple Carry Adders (RCA)

Given a uniform input distribution, the worst-case delay of an 8-bit RCA is almost one magnitude longer than that of the best case in Fig. 1. However, the critical path is not activated in most cases except for very small portions of input patterns, i.e. the activated ratio of the critical path is less than 1%. Therefore, the average-case design style is quite promising to boost the adder's performance.

2.2 Multistage Function Speculation

This paper mainly focuses on the adder architecture with function speculations to realize the average-case performance. Figures 2 and 3 show the original circuit and a circuit with two-stage function speculation. We denote the critical delay of the fixed latency circuit in Fig. 2 as τ. The inserted corrector is used to connect the function speculation circuit to the combinational block, to check the speculation value, and to correct the output in an extra clock cycle if predicted wrong. Hence, The critical delay of the circuit is the maximum of the register-register delays of the these path groups, and is denoted as $\tau' = \max\{\tau_{fs}+\tau_{mux}+\lambda\tau, (1-\lambda)\tau+\tau_{xor}, \tau_{mux}+\lambda\tau\}$. Generally, the value of τ_{fs}, τ_{mux} and τ_{xor} are relatively small compared with the delay τ of combinational circuits, which consist of tens or hundreds of logic gates. Therefore, the critical delay τ' of the circuit with two-stage function speculation can be reduced as much as half of the original delay τ in theory. The latency of the circuit is one clock cycle τ' when no speculation errors are detected. Once a speculation error is detected by the corrector, the latency of the circuit becomes two clock cycles $2\tau'$. The throttle signal will be asserted until the correct values appear on the output. If the speculation error ratio is below a certain threshold, the variable latency circuits can obtain significant performance improvements compared with the original circuit. We call this architecture as two-stage function speculation. Baneres, et al., [10] illustrated the variable latency design by two-stage function speculation in combinational circuits. However, their design methodology is only applicable for two-stage function speculation, which limits further reduction of the cycle time.

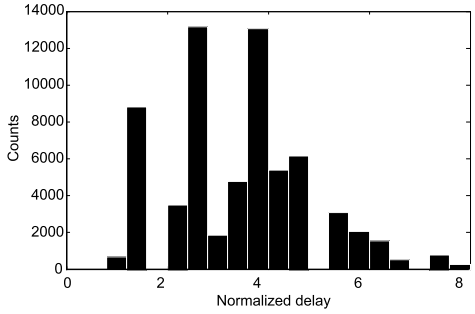


Fig. 1 Delay histogram of a 8-bit ripple carry adder.

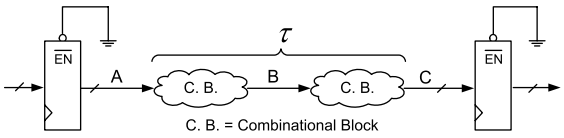


Fig. 2 Original fixed latency architecture.

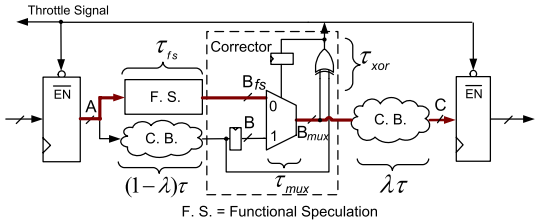


Fig. 3 Two-stage function speculation architecture [10].

The architecture of a circuit with multistage function speculation is shown in Fig. 4, where we insert $(m-1)$ function speculation points into the critical path. The critical delay of the circuit with $(m-1)$ stages function speculation is equal to $\tau' = \max\{\tau_{fs,j-1} + \tau_{mux} + \lambda_j\tau + \tau_{xor} + \tau_{or}, \lambda_{j-1}\tau + \tau_{xor}, \tau_{mux} + \lambda_j\tau\}$ in which $j = 2, 3, ..., m$ and $\sum_{i=1}^{m} \lambda_i = 1$. As the first part donates the critical delay, we get the critical delay as $\tau' = \max(\tau_{fs,j-1} + \lambda_j\tau + \tau_{cor})$, where $\tau_{cor} = \tau_{mux} + \tau_{xor} + \tau_{or}$ is defined as the additional delay of the corrector. The critical path of the 2nd stage is marked in Fig. 4. As we can see, the theory lower bound of the circuit's optimal latency is τ/m, when the speculation delay $\tau_{fs,j-1}$ and the additional delay τ_{cor} can be omitted compared with the original circuit delay τ and no speculation errors occurred. However, the circuit's latency with error recovery is μ times τ/m. The value of average cycle μ de-

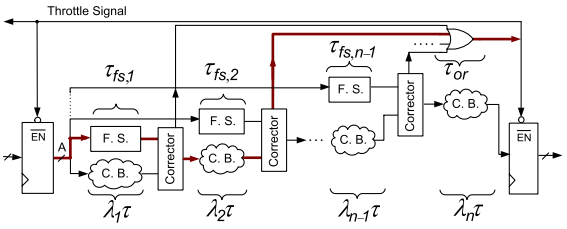


Fig. 4 Multistage function speculation architecture.

pends on the distribution of speculation errors ranging from 2 to m.

To summarize this section, multistage speculation adder provides significant performance improvements. However, systematical design analysis and optimization for speculation adder is still unsolved. We try to attack these design challenges in this paper.

3. Multistage Function Speculation Adder

This section discusses the structure of the variable latency adder with multistage function speculation. We first illustrate the adder architecture. Second, we show how to design adaptive predictors under different input patterns.

3.1 Adder Architecture

The proposed multistage function speculation adder (MFSA) is based on a RCA. First, we introduce the RCA briefly. After that, we describe the multistage function speculation adder including m sub adders. Finally, the work flow will be explained.

An N-bit RCA consists of N full adders, which is shown in Fig. 5(a). In a 1-bit full adder, A_i and B_i are the adder's inputs, C_{i-1} is the carry input, S_i is the sum output, and C_i is the carry output. By defining the carry generate signal $G_i = A_iB_i$ and carry propagate signal $P_i = A_i \oplus B_i$, S_i and C_i can be written as follows:

$$S_i = P_i \oplus C_{i-1} \tag{1}$$

$$C_i = G_i + P_iC_{i-1} \tag{2}$$

An N-bit RCA is constructed by cascading N 1-bit full adders in series. Expanding the carry output C_N by P_i and G_i, we get the following equation:

$$\begin{aligned} C_N = {} & G_N + P_NG_{N-1} + \ldots P_NP_{N-1}\ldots P_2G_1 \\ & + P_NP_{N-1}\ldots P_1C_0 \end{aligned} \tag{3}$$

Based on the above RCA, the m-stage function speculation adder is shown in Fig. 5(b). Each block is composed of a modified RCA (*PG*, *CS* and *Predictor*) and a *Corrector*. The length of stage j is n^j. We define $N^j = \sum_{i=1}^{j} n^i$ as the position of the j-stage predictor, and $N^m = N$ is the total length of the adder. The modified RCA includes three parts: the PG unit for propagate and generate signals, the CS unit for carry and sum signals, and the predictor. The PG and CS units are the same as those in the RCA. The predictor will be discussed in the next subsection. Supposing a block is the j-th stage in the m-stage function speculation adder, we denote the inputs of the PG unit as $A^j = (A_1^j, A_2^j, \ldots, A_n^j)$ and $B^j = (B_1^j, B_2^j, \ldots, B_n^j)$. The carry output and input of the CS unit are C_o^j and C_{in}^j, respectively. The output of the predictor is $C_{o,fs}^j$ and the sum output is $S^j = (S_1^j, S_2^j, \ldots, S_n^j)$.

Table 1 Design parameters.

Parameters	Description
N	Adder's bit length
m	Adder's stages
n^i	Width of i-th stage adder
k^i	Width of i-th stage predictor
d^i	Type of i-th stage predictor

The work flow is illustrated as follows. In Fig. 5(b), the predictor in each stage speculates a carry output $C_{o,fs}^j$ (the last stage does not have a predictor) and this signal will be given to its next stage as a carry input via a multiplex. The original carry output C_o^j will also be given out to the next stage via the same multiplex. The select signal of the multiplex is decided by the flip-flop, which stores the comparison result to indicate if the speculation is correct, or not.

In most cases, the adder can finish its calculation in one cycle when the speculation value in each stage is correct. In other cases, the predictor produces some erroneous values. The errors can be detected by the XOR gate, which compares the speculated carry output $C_{o,fs}^j$ with the exact carry output C_o^j. In such cases, the adder has to spend 2 to m clock cycles to complete the calculation. Under the worst case, m clock cycles are needed to generate the correct results at the output, since there are $(m-1)$ predictors to be verified and corrected. As Fig. 4 demonstrates, the clock cycle is limited by the longest stage delay, which is much shorter than the original carry chain delay. Hence, the average delay is improved.

3.2 Adaptive Predictor Design

Obviously, the predictors determine the error rate of prediction, which has significant impacts on the performance. This subsection discusses the predictor design. As the input patterns of adder are various depending on application scenarios, a fixed predictor may not work well in all cases. For example, two opposite-sign addends with a positive sum will activate a long carry chain in two's complement adder. Therefore, the error rate of the predictors in previous speculation adders [1], [11], will be quite high. If this input pattern happens frequently, the average performance will be bad. Based on the observation, the predictor should be designed with priori knowledge of the system and can adapt to the inputs.

The predictor is modified from the carry chain of an RCA. As shown in Fig. 5(c), we rewrite the logic expression of the output $C_{o,fs}$ of k-bit carry chain as follows:

$$\begin{aligned} C_{o,fs} = {} & G_{n-1} + P_{n-1}(G_{n-2} + P_{n-2}(G_{n-3} + \ldots \\ & + P_{n-k+1}C_{n-k+1,fs})\ldots) \end{aligned} \tag{4}$$

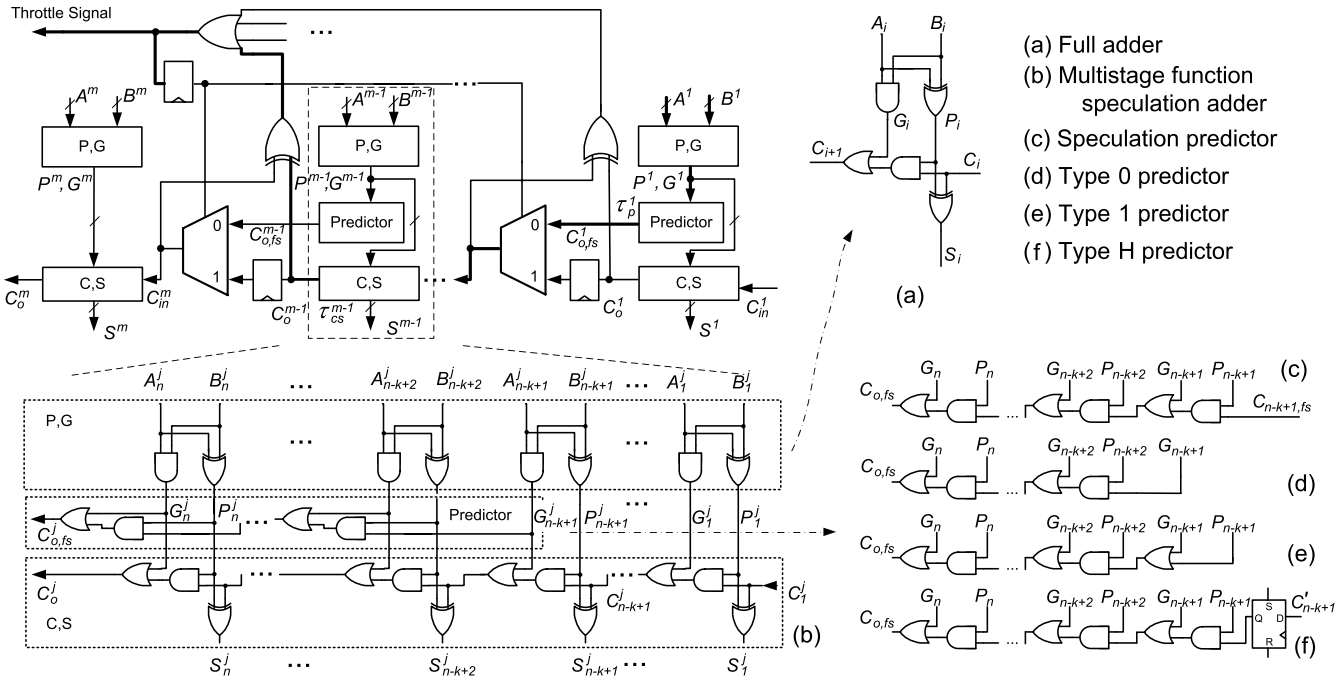


Fig. 5 Multistage function speculation adder diagram.

Since C_n is partially determined by C_{n-k+1}, we can use a speculated signal, $C_{n-k+1,fs}$, to approximate C_n. The input $C_{n-k+1,fs}$ could be assumed to 0, 1, or its latest history value. Three kinds of predictors are shown in Figs. 5(d), (e) and (f). They are named as *Type 0*, *Type 1* and *Type H*. The flexible *Type H* predictor uses a D flip-flop with asynchronous reset (R) and set (S) pins. Enabling reset or set signals will force the predictor to be *Type 0* or *Type 1*.

The speculation input $C_{n-k+1,fs}$ would affect the predictor's final carry out $C_{o,fs}$, if and only if the following two equations are satisfied.

$$G_i = 0, P_i = 1 \quad i = n-k+1, ..., n-1 \tag{5}$$

In the following section, we will prove that the probability of the condition Eq. (5) decreases exponentially with the increment of k under a uniform input distribution. Therefore, it is possible to speculate the final carry output with a low error rate. However, the exponentially decreasing law is not satisfied in some other distributions. As a result, choosing proper type of predictors becomes critical to adapt the distribution of the inputs in the design phase. In addition, the *Type H* predictors can be configured on the fly when the input distribution changing.

4. Performance Analysis

To analyze and optimize a multistage function speculation adder, this section gives out its analytical models. In an m-stage function speculation circuit, the average delay can be described as follows:

$$\tau_{ave} = \sum_{\mu=1}^{m} P(C=\mu)(\mu\tau_{cyc}) = C_{ave}\tau_{cyc} \tag{6}$$

where $P(C=\mu)$ is the probability of the adder to complete the calculation in μ cycles. τ_{cyc} is the length of the adder's clock cycle, and C_{ave} is the average cycle to complete one operation.

The clock cycle τ_{cyc} is decided by the critical path delays in the multistage function speculation adder which is marked in Fig. 5(b). We can obtain the expression of τ_{cyc} as follows:

$$\begin{aligned} \tau_{cyc} &= \max(\tau_{cs}^{i} + \tau_{p}^{i-1} + \tau_{cor}) \quad i = 2, 3, \ldots, m \\ \tau_{cor} &= \tau_{mux} + \tau_{xor} + \tau_{or} \end{aligned} \tag{7}$$

where τ_{cs}^{i} is the critical delay of the i-th CS unit; τ_{p}^{i} is the critical delay of the i-th predictor; τ_{cor} is the delay introduced by the corrector; τ_{mux}, τ_{xor} and τ_{or} are the delay of logic gates, respectively.

τ_{cyc} can be expressed as follows:

$$\begin{aligned} \tau_{cyc} &= \max(\tau_{cs}^{i} + \tau_{p}^{i-1} + \tau_{cor}) \\ &= \max(n^{i}\tau_{cs0} + k^{i}\tau_{p0} + \tau_{d}(d^{i}) + \tau_{cor}) \end{aligned} \tag{8}$$

where n^i and k^i is the bit length of the i-th stage and predictor. τ_{cs0} and τ_{p0} are the 1-bit unit delays of CS unit and predictor. τ_d is an additive delay depending on the type of the predictor, and $\tau_d = 0$ for the *Type 0* predictor.

4.1 Average Cycle

The event E_{α}^{i} is defined as stage i generates a conflict, and its probability is $P(E_{\alpha}^{i}) = \alpha_i$. Let us focus on the i-th stage, its conflict can be transferred to the following $(m-i)$ stages. Event E_{β}^{i} is defined as a conflict transferd from the i-th stage to the $(i+1)$-th stage, and its probability is $P(E_{\beta}^{i}) = \beta_i$. The

event $E(C_i \le \mu)$ denotes this conflict is eliminated within μ cycles, and can be expressed as follows:

$$E(C_i \le \mu) = \Omega - E(C_i > \mu) \tag{9}$$
$$E(C_i > \mu) = {E_\alpha}^i \cap {E_\beta}^{i+1} \cap {E_\beta}^{i+2} \ldots \cap {E_\beta}^{i+\mu-1} \tag{10}$$

where Ω is the complete set, and C_i is the cycle number for the i-th stage to eliminate its conflict. $(i + \mu)$ is always less than m because there are at most $(m-i)$ stages for the conflict of the i-th stage to be transferred.

The probability for the adder to eliminate all conflicts within μ cycles is

$$P(C \le \mu) = P\left(\bigcap_{i=1}^{m-\mu} (\Omega - E(C_i > \mu))\right) \tag{11}$$

Based on Eq. (11), Eq. (6) can be rewritten as:

$$\begin{aligned} C_{ave} &= \sum_{\mu=1}^{m} P(C = \mu)\mu \\ &= \sum_{\mu=1}^{m} (P(C \le \mu) - P(C \le \mu - 1))\mu \\ &= m - \sum_{\mu=1}^{m} P(C \le \mu) \\ &= m - \sum_{\mu=1}^{m} P\left(\bigcap_{i=1}^{m-\mu} (\Omega - E(C_i > \mu))\right) \end{aligned} \tag{12}$$

In general, $P(E_\alpha^i)$ and $P(E_\beta^i)$ are determined by the design and the input distribution $f(x, y)$. The distribution is detailed in the Appendix.

$$P(E_\alpha^i) = \int\int f(x, y) I^i(x, y) dxdy \tag{13}$$
$$P(E_\beta^i) = \int\int f(x, y) J^i(x, y) dxdy \tag{14}$$

If the input (x, y) causes a conflict in stage i, $I^i(x, y) = 1$, otherwise, $I^i(x, y) = 0$. Similarly, if the inputs cause a conflict to be transfered to the next stage, $J^i(x, y) = 1$, otherwise, $J^i(x, y) = 0$.

Based on Eq. (5), the conflict conditions for stage j are listed as follows:

$$C_{n^j-k^j+1,fs} \ne C_{n^j-k^j+1} \tag{15}$$

and,

$$\begin{aligned} P_i &= 1, n^j \ge i \ge n^j - k^j + 1 \\ G_i &= 0, n^j \ge i \ge n^j - k^j + 1 \end{aligned} \tag{16}$$

If we express the condition in a graphical way, Fig. 6 illustrates the behavior pattern of an i-th stage predictor. The example predictor is located at the bit of $N^i = N - 1$ with a length of $k^i = 2$. The coordinate (a, b) represents the adder's inputs a and b. In Fig. 6(a), the color represents the actual carry value $C_{(N-1)-k+1}$ at the position of the predictor input. The colored region indicates that the conflict condition in Eq. (16) is satisfied at those inputs in Fig. 6(b). Figure 6(c) is generated according to the conflict condition Eq. (15) and (16). It is composed by a set of mis-prediction triangles. For the *Type 0* predictor, if the input (a, b) is at the green region, the conflict condition is satisfied. The vertices of the triangle

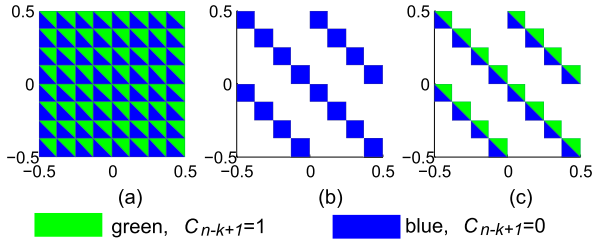


Fig. 6 Prediction conflict and transfer pattern.

are (u, v), $(u + t, v - t)$, and $(u + t, v)$, where

$$\begin{aligned} u &= -2^{-1} + 2^{-(N-N^i)} r + 2^{-k^i} s \\ v &= -2^{-1} + 2^{-(N-N^i)} r - 2^{-k^i} s \\ t &= 2^{-(N-N^i+k^i)} \\ r &= 0, 1, ..., 2^{N-N^i} - 1, \quad s = 0, 1, ..., 2^{k^i} - 1 \end{aligned} \tag{17}$$

Similarly, a mis-prediction occurs when the inputs are in the blue region for the *Type 1* predictor. Its vertices are (u, v), $(u, v - t)$, $(u + t, v - t)$. The *Type H* predictor takes the history information into account, and a mis-prediction occurs when: (1) current input locates at the blue region in Fig. 6(b), and (2) current input and previous one in Fig. 6(a) have different values (colors).

The transfer pattern of $({E_\beta}^i \cap {E_\beta}^{i+1} \ldots \cap {E_\beta}^{i+j})$ consists of rectangles, of which vertices are (u, v), $(u, v-t)$, $(u+t, v-t)$ and $(u + t, v)$, where

$$\begin{aligned} u &= -2^{-1} + 2^{-(N-N^{i+j})} r + 2^{-(N^{i+j}-N^i)} s \\ v &= -2^{-1} + 2^{-(N-N^{i+j})} r - 2^{-(N^{i+j}-N^i)} s \\ t &= 2^{-(N-N^{i+j})} \\ r &= 0, 1, ..., 2^{N-N^{i+j}} - 1, \quad s = 0, 1, ..., 2^{N^{i+j}-N^i} - 1 \end{aligned} \tag{18}$$

4.2 Area

As Fig. 4 has shown, the proposed m-stage adder with k-bit predictors consists of $N = m * n$ full adders, $(m - 1)$ correctors and $(m - 1)$ predictors. Due to the adder's regular structure, the area of the circuit is proportional to its cell number, which can be expressed by the following equation:

$$A_{msad} = NA_{fa} + (m - 1)A_{cor} + \sum_{i=1}^{m} A_{pre}^i \tag{19}$$

where A_{msad} is the area of the multistage function speculation adder, A_{fa} is the area of a full adder, and A_{cor} is the area of a corrector. A_{pre} is the area of a predictor, which depends on its type:

$$A_{pre} = \begin{cases} (2k - 2)A_{GATE2} & \text{Type 0} \\ (2k - 1)A_{GATE2} & \text{Type 1} \\ 2kA_{GATE2} + A_{SRDFF} & \text{Type H} \end{cases} \tag{20}$$

where A_{GATE2} is the average area of 2-input standard cell NAND2 and NOR2, and A_{SRDFF} is the area of a D-type flip-flop with asynchronous set and reset signals.

Table 2 Typical area parameters for a 128-bit 16-stage with 4-bit predictor multistage function speculation adder.

Unit name	Unit number	Area per unit
Full adder A_{fa}	128	$37.7\,\mu m^2$/bit
Predictor(0) A_{pre}	15	$28.2\,\mu m^2$/stage
Predictor(1) A_{pre}	15	$32.9\,\mu m^2$/stage
Predictor(H) A_{pre}	15	$76.5\,\mu m^2$/stage
Corrector A_{cor}	15	$65.9\,\mu m^2$/stage

The typical unit area is shown in Table 2 and the values are extracted based on the SMIC 0.13 μm process. The total area of a 128-bit 16-stage function speculation adder with 4-bit *Type 0* predictor is $6237\,\mu m^2$. The predictor and the corrector occupy 22.6% of the total area.

5. Designs for Independent Uniform Distribution Inputs

Based on the models in Sect. 3, we illustrate a design methodology for multistage function speculation adders under different configurations. We analyze the throughput and area trends with variable bit length of the adder under the independent uniform distribution.

Given an N-bit multistage function speculation adder, this section decides the stage number m, the bit number n^i of each stage, and the bit number k^i for each predictor to maximize the performance. In order to gain tractability, we remove predictor area as a factor and suppose that K, the total number of predictor bits, is a constant as follows:

$$K = \sum_{i=1}^{m-1} k^i \tag{21}$$

5.1 Average Cycle

First, we deduce the average delay τ_{ave} under independent uniform distribution (*i.u.d.*) inputs. In this distribution, E_α^i and E_β^i are independent. Therefore, Eqs. (11) and (12) could be written as:

$$\begin{aligned} &P(E(C_i > \mu)) \\ &\quad = P\left(E_\alpha{}^i \cap E_\beta{}^{i+1} \cap E_\beta{}^{i+2} \ldots \cap E_\beta{}^{i+\mu-1}\right) \\ &\quad = P\left(E_\alpha{}^i\right) P\left(E_\beta{}^{i+1}\right) P\left(E_\beta{}^{i+2}\right) \ldots P\left(E_\beta{}^{i+\mu-1}\right) \\ &\quad = \alpha_i \beta_{i+1} \beta_{i+2} \ldots \beta_{i+\mu-1} = \alpha_i \prod_{j=i+1}^{i+\mu-1} \beta_j \end{aligned} \tag{22}$$

$$\begin{aligned} C_{ave} &= m - \sum_{\mu=1}^{m} P\left(\bigcap_{i=1}^{m-\mu} (\Omega - E(C_i > \mu))\right) \\ &\le m - \sum_{\mu=1}^{m-1} \prod_{i=1}^{m-\mu} \left(1 - \alpha_i \prod_{j=i+1}^{i+\mu-1} \beta_j\right) \end{aligned} \tag{23}$$

The result gives an upper bound for C_{ave}, as the event $(\Omega - E(C_i > \mu))$ is correlated. The bound is tight enough as the experiments have shown in Sect. 7.

The conflict and transfer probabilities, α_i and β_i under *i.u.d.* inputs are:

$$\alpha_i = P(E_\alpha^i) = \int\int I^i(x,y)dxdy = 2^{-1} * 2^{-k^i} \tag{24}$$

$$\beta_i = P(E_\beta^i) = \int\int J^i(x,y)dxdy = 2^{-n^i} \tag{25}$$

The types of predictor do not affect the probabilities, as carry $C_{n^i-k^i+1}$ is independent and the probability $P(C_{n^i-k^i+1} = 0) = P(C_{n^i-k^i+1} = 1) = 1/2$. Thus, we choose the *Type 0* predictor for its lower delay.

5.2 Design Optimization

To optimize the performance of the multistage, we give a theorem for optimal performance design under *i.u.d.* inputs.

Theorem 1: For an m-stage N-bit multistage function speculation adder with total K-bit predictor under uniform input distribution, the bit number of each stage and predictor should be N/m and $K/(m-1)$, respectively. This condition is necessary to obtain optimal performance under the first order approximation.

Proof 1: As $\beta_i < \alpha_i$ $(k \le n)$, we can obtain the first order approximation of Eqs. (6) and (12) as follows:

$$\begin{aligned} \tau_{ave} &= C_{ave}\tau_{cyc} \\ &= \left(m - \sum_{\mu=1}^{m-1} \prod_{i=1}^{m-\mu} (1 - \alpha_i \prod_{j=i+1}^{i+\mu-1} \beta_j)\right)\tau_{cyc} \\ &\approx \left(1 + \sum_{i=1}^{m-1} \alpha_i\right)\tau_{cyc} \end{aligned} \tag{26}$$

The optimal performance will be acquired when both τ_{cyc} and $(1+\sum_{i=1}^{m-1}\alpha_i)$ reach their minimal values simultaneously. Since $N = \sum_{i=1}^{m} n_i$ and $K = \sum_{i=1}^{m-1} k_i$, the minimal of τ_{cyc} is obtained:

$$\begin{aligned} \tau_{cyc} &= \max\left(n^i\tau_{cs0} + k^i\tau_{p0} + \tau_{cor}\right) \\ &\ge N\tau_{cs0}/m + K\tau_{p0}/(m-1) + \tau_{cor} \end{aligned} \tag{27}$$

when $n^i = N/m$ and $k^i = K/(m-1)$.

Next, we will show that this condition will also minimize $(1 + \sum_{i=1}^{m-1}\alpha_i)$.

$$\left(1 + \sum_{i=1}^{m-1} \alpha_i\right) \ge 1 + (m-1)\sqrt[m-1]{\prod_{i=1}^{m-1} \alpha_i} \tag{28}$$

The equal relationship is satisfied when $k_i = K/(m-1)$ and $\alpha_i = 2^{-K/(m-1)+1}$ under the uniform input distribution. Therefore, the optimal average-case delay can be given:

$$\begin{aligned} \tau_{ave} &= \max\left(n_i\tau_{cs0} + \tau_{cor} + k_i\tau_{p0}\right)\left(1 + \sum_{i=1}^{m-1} \alpha_i\right) \\ &\ge \left(\frac{N}{m}\tau_{cs0} + \frac{K}{m-1}\tau_{p0} + \tau_{cor}\right) \cdot \\ &\quad \left(1 + (m-1)\sqrt[m-1]{\prod_{i=1}^{m-1} \alpha_i}\right) \\ &\ge \left(\frac{N}{m}\tau_{cs0} + \frac{K}{m-1}\tau_{p0} + \tau_{cor}\right) \cdot \\ &\quad \left(1 + (m-1)2^{-K/(m-1)+1}\right) \end{aligned} \tag{29}$$

τ_{ave} reaches the minimum when $n^i = N/m$ and $k^i = K/(m-1)$. ■

5.3 Throughput and Area Trends

We will give a theorem on the delay and area trend of the multistage function speculation adder with the adder's bit number N.

Theorem 2: The average-case delay of the optimal performance N-bit multistage function speculation adder is proportional to $O(\log_2 N)$ and the adder's area is proportional to $O(N)$.

Proof 2: Give $n = k = \log_2 N$, then m is $\frac{N}{n} = \frac{N}{\log_2 N}$. According to Eq. (29), the optimized the average-case delay $\tau_{ave,opt}$ is

$$\begin{aligned}\tau_{ave,opt} &= \left(\frac{N}{m}\tau_{cs0} + k\tau_{p0} + \tau_{cor}\right)\left(1 + (m-1)\cdot 2^{-(k+1)}\right)\\ &= \left(n\tau_{cs0} + k\tau_{p0} + \tau_{cor}\right)\left(1 + \frac{N}{n}2^{-(k+1)}\right) \qquad (30)\\ &= \left((\tau_{cs0} + \tau_{p0})\log_2 N + \tau_{cor}\right)\left(1 + \frac{N}{\log_2 N}\cdot\frac{1}{N+1}\right)\\ &\to (\tau_{cs0} + \tau_{p0})\log_2 N + \tau_{cor}\\ &\sim O(\log_2 N)\end{aligned}$$

The area in Eq. (19) can be expressed as

$$\begin{aligned}A_{msad} &= NA_{fa} + (m-1)A_{cor} + \sum_{i=1}^{m} A^i_{pre} \qquad (31)\\ &= NA_{fa} + \left(\frac{N}{\log_2 N} - 1\right)A_{cor} + \frac{N}{\log_2 N}(2\log_2 N - 1)A_{gate2}\\ &\to (A_{fa} + 2A_{gate2})N\\ &\sim O(N)\end{aligned}$$

■

6. Designs for Specific Distribution Inputs

The keys to the optimization problem with specific distribution inputs are to characterize the input distribution and to quantify the performance for a given adder configuration.

In the optimization problem, the design parameters are the width n^i of each stage, and the width k^i and the type d^i of each predictor. The goal is to minimize either the average delay τ_{ave} or the area A_{msad}, while satisfying the area and the throughput constraints.

6.1 Average Cycle

As E^i_α and E^i_β are correlated under a specific input distribution, we cannot deduce C_{ave} as a sum of product of α_i and β_i as Eqs. (22) and (23). To reduce the complexity and keep the accuracy, we inherit the upper bound approximation in Eq. (23). Thus, Eq. (12) can be deduced as:

$$\begin{aligned}&P(\Omega - E(C_i > \mu)) \qquad (32)\\ &= 1 - P\left({E_\alpha}^i \cap {E_\beta}^{i+1} \cap {E_\beta}^{i+2} \ldots \cap {E_\beta}^{i+\mu-1}\right)\\ &\approx 1 - P\left({E_\alpha}^i\right)P\left({E_\beta}^{i+1} \cap {E_\beta}^{i+2} \ldots \cap {E_\beta}^{i+\mu-1}\right)\\ &= 1 - \alpha_i\beta_{i+1,i+\mu-1}\\ C_{ave} &= m - \sum_{\mu=1}^{m} P\left(\bigcap_{i=1}^{m-\mu}(\Omega - E(C_i > \mu))\right) \qquad (33)\\ &\le m - \sum_{\mu=1}^{m}\prod_{i=1}^{m-\mu} P(\Omega - E(C_i > \mu))\\ &\approx m - \sum_{\mu=1}^{m-1}\prod_{i=1}^{m-\mu}\left(1 - \alpha_i\beta_{i+1,i+\mu-1}\right)\end{aligned}$$

The conflict and transfer probability α and β are sensitive to the distribution. We take the exponential distribution detailed in the Appendix as an example. Suppose a predictor is located at the bit N^i and the predictor length is k^i, the conflict rate $\alpha(0)$, $\alpha(1)$ and $\alpha(H)$ (for *Type 0*, *1* and *H* predictor), can be calculated by integrating the probability density $f(x, y)$ in the regions $I(x, y)$ where a conflict can occur.

For *Type 0* predictor, the conflict probability is

$$\begin{aligned}\alpha_i(0) &= \int_{-1/2}^{1/2}\int_{-1/2}^{1/2} I_0(x,y)f(x,y)dxdy \qquad (34)\\ &= \sum_{u,v}\int_{v-1/2^{n+k}}^{v}\int_{u+v-y}^{u+1/2^{n+k}} \frac{\lambda}{1-e^{-\lambda}}e^{-\lambda[\![x+y]\!]}dxdy\\ &= \left(1 - \frac{1-e^{-\lambda/2^{n+k}}}{\lambda/2^{n+k}}\right)\frac{1}{1-e^{-\lambda/2^n}}\end{aligned}$$

where n and k denote $(N-N^i)$ and k^i for short. Similarly, For *Type 1* and *Type H* predictors, $\alpha_i(1)$ and $\alpha_i(H)$ are deduced as

$$\begin{aligned}\alpha_i(1) &= \int_{-1/2}^{1/2}\int_{-1/2}^{1/2} I_1(x,y)f(x,y)dxdy \qquad (35)\\ &= \sum_{u,v}\int_{v-1/2^{n+k}}^{v}\int_{u}^{u+v-y} \frac{\lambda}{1-e^{-\lambda}}e^{-\lambda[\![x+y]\!]}dxdy\\ &= \left(\frac{e^{\lambda/2^{n+k}} - 1}{\lambda/2^{n+k}} - 1\right)\frac{e^{-\lambda/2^n}}{1-e^{-\lambda/2^n}}\\ \alpha_i(H) &= \left(1 - \frac{1-e^{-\lambda/2^{n+k}}}{\lambda/2^{n+k}}\right)\left(\frac{e^{\lambda/2^{n+k}} - 1}{\lambda/2^{n+k}} - 1\right) \qquad (36)\\ &\quad\cdot\left(\frac{e^{-\lambda/2^n} + e^{-\lambda/2^{n+k}}}{(1-e^{-\lambda/2^n})\left(1-e^{-\lambda/2^{n+k}}\right)}\right)\end{aligned}$$

Conflict transfer probability $\beta_{i,\mu} = P({E_\beta}^{i+1} \cap {E_\beta}^{i+2} \ldots \cap {E_\beta}^{i+\mu-1})$ is deduced as

$$\begin{aligned}\beta_{i+1,i+\mu-1} &= \int_{-1/2}^{1/2}\int_{-1/2}^{1/2} J(x,y)f(x,y)dxdy \qquad (37)\\ &= \sum_{u,v}\int_{v-1/2^{n+k}}^{v}\int_{u}^{u+1/2^{n+k}} \frac{\lambda}{1-e^{-\lambda}}e^{-\lambda[\![x+y]\!]}dxdy\end{aligned}$$

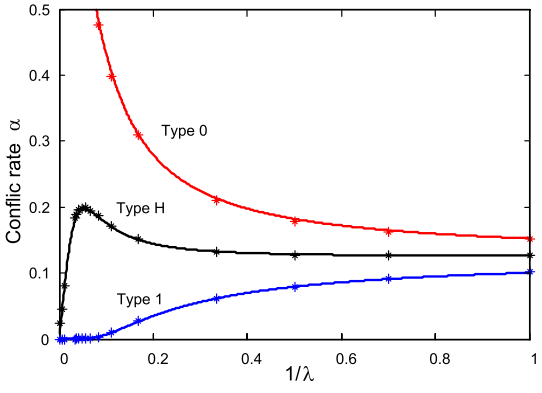


Fig. 7 Conflict rates of *Type 0*, *1* and *H* predictor.

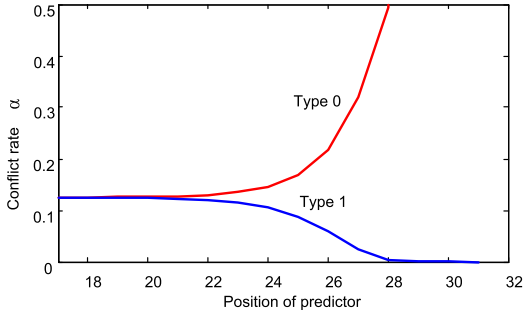


Fig. 8 Conflict rate vs. predictor's position.

$$= \frac{e^{-\lambda/2^{n+k}} - 1 + \lambda/2^{n+k} + \left(e^{\lambda/2^{n+k}} - 1 - \lambda/2^{n+k}\right) e^{-\lambda/2^{n}}}{\lambda/2^{n+k}\left(1 - e^{-\lambda/2^{n}}\right)}$$

where $n = N - N^{i+\mu-1}$, $k = N^{i+\mu-1} - N^{i+1}$.

As Eqs. (34)–(37) illustrate, the probabilities are functions of the distribution parameter λ. When λ approaches 0, the distribution tends to be uniform. That is, $\alpha_i(0)$, $\alpha_i(1)$ and $\alpha_i(H)$ approach $1/2^{k^i+1}$, and $\beta_{i+1,i+\mu-1}$ approaches $1/2^{N^{i+\mu-1}-N^i}$. This result is consistent with the previous results under *i.u.d.* inputs. When λ approaches infinity, the limits of $\alpha(0)$, $\alpha(1)$, $\alpha(H)$ and β are 1, 0, 0 and 1.

6.2 Discuss on Predictor's Type

Predictor selection is an important part of the optimization problem. We discuss the behavior of predictors as a basis. Figure 7 shows the relationship between the distribution parameter λ and the conflict rates $\alpha(0)$, $\alpha(1)$ and $\alpha(H)$ of *Type 0*, *1* and *H* predictor. The asterisks are obtained by circuit simulations. In this example, the predictor is located at the $(N-2)$ bit with the width of 2. We see that the *Type 1* predictor has a lower conflict rate. However, the performance gap between *Type 0* and *1* predictor widens gradually as $1/\lambda$ decreases.

The predictors' behaviors are different at different position. As Fig. 8 shown, we compared the conflict rate under different predictor's positions. Both widths of the *Type 0* and *1* are 2, and the distribution parameter $1/\lambda$ is 1/200. The *Type 1* predictor is much better than the zero-input one as the position is higher than the 22th bit.

However, the average delay is determined not only by the average cycle, but also by the delay of the critical path in the circuit. The *Type 1* predictor has a better conflict performance, but it leads to a larger delay according to Eq. (8).

Algorithm 1 design methods for specific applications

```
Input: characteristic of adder's input, performance requirement
Analyze the distribution characteristic of the inputs
for c = c1 to c2 do
    Divide the adder by n0 = log2 λ + c
    Design the lower side of the adder with methodology in Sec.5
    Design the higher side of the adder by brute force
    if Obtained a better performance then
        Record the design parameters
    end if
end for
Output: the design parameters(the types of predictors and the length of
stages and predictors)
```

6.3 Design Optimization

To determine the optimal type of the predictor and the optimal length of each stage and predictor, a nonlinear integer optimization is used. For adders with 32 or less bit length, a brute force method is acceptable, if an appropriate pruning method removes most infeasible design areas. A heuristic algorithm is proposed for the adders with larger bit length.

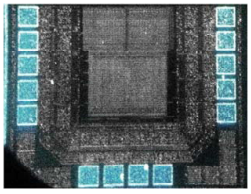
A proper partition is important to reduce the complexity of the heuristic algorithm. As we have got a closed analytical optimization result for *i.u.d.* inputs, extending the methodology to a part of adder under specific distribution inputs is attractive. We noted that the conflict rate α and the conflict transfer probability β of the three types of the predictors, converge to the result as that of independent uniform distribution, when $2^{N-N^i} \gg \lambda$, $2^{N-N^i+k^i} \gg \lambda$. This trend is shown in Fig. 8, and can be deduced from Eqs. (34) and (35).

Therefore, an appropriate division of the adder can reduce the complexity of the optimization. We determine the critical position $n_0 = log_2\lambda + c$, where c is a design margin. It is guaranteed that the difference of the performance is less than 2% among the three types of predictors if we takes $c = 4$. For the lower side of the critical position, where the distribution are similar to *i.u.d.*, we used the design method illustrated Sect. 5. However, for the higher side, it is required to consider the impact on the distribution parameters to optimize the design. As n_0 is less than 20 bit in most applications, the brute force method is acceptable. We try the critical position in a range (c_1 to c_2) in order not to miss the optimal configuration, since the parameters of prediction stages are integers.

The full design flow is illustrated in Algorithm 1.

7. Experiments

In this section, we explain our experimental setup and verify the proposed models. Second, we analyze and optimize the adder performance. Third, we show the average delay



Minimal clock cycle

Struct.	T_D /ns	T_M /ns
RCA	95.88	106.6
Type 0	31.49	38.2
Type 1	34.08	40.8
Type H	36.71	41.6

Fig. 9 Test chip.

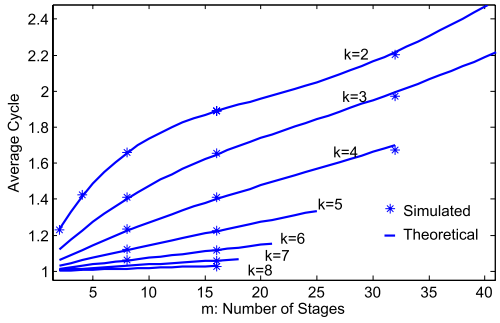


Fig. 10 Average cycles under different k and m configurations.

and area relationship of the proposed adder using different stage and predictor configurations. Finally, we compare the proposed adder with other existing adders.

7.1 Experimental Setup

We use Verilog and DesignWare library to implement the multistage function speculation adder and the reference adder. The circuits are synthesized using a 0.5 V standard-cell library for SMIC 0.13 μm CMOS technology. We generate specified distributed 5000 inputs and use gate level simulator to catch the speculation errors for performance model analysis at gate level. Timing are evaluated by Synopsys PrimeTime^{TM} The bit number of the multistage function speculation adder ranges from 64 to 1024.

A test chip is fabricated as shown in Fig. 9, which includes a 32-bit proposed adder and an RCA for reference. A built-in on-chip serial-parallel adapter is used to feed the adder and catch its output.

7.2 Performance Model Validation

Figure 10 shows average cycles under different k and m configurations. The solid line is plotted based on Eq. (12). The star points are obtained by simulating the specific adder under 5000 random input data. As we can see, the values predicted by our performance models Eq. (23) fit well with the experimental results. It verifies that our assumption that each stage generates a conflict and transfers the conflict to the next stage independently is reasonable. Furthermore, the average cycle number decreases when m decreases and k is held constant. This drop can be explained by noting that when k is held constant, the error probability of each stage remains unchanged — this means the lower stage number, m, will lead to a smaller average cycle number. However, the cycle number increases when m is held constant and

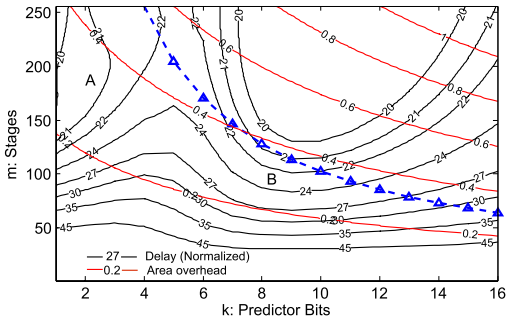


Fig. 11 Analytical performance and area curves under different k and m configurations.

k decreases. When k decreases, the speculation errors rise thus the average cycle number does so as well.

7.3 Performance Analysis and Optimization for *i.u.d.* Inputs

Based on Eqs. (29) and (19), we can plot the delay and area of a 1024-bit multistage function speculation adder under different m parameters in Fig. 11. We assume that both 1-bit full adder and multiplex critical path delay are 1 unit. The area overhead is normalized with a 1024-bit ripple carry adder. The black and red line separately represents the equal delay and area overhead under different k and m configurations. When we restrict $k \leq n$, the valid design region is below the dotted line with triangle marks in Fig. 11.

To our surprise, there are two regions: A and B for the optimal performance design candidates. As Fig. 11 has shown, there are two design regions A and B for optimal performance candidates of a 1024-bit adder. In region A, a small predictor bits and low stage numbers are used: the clock frequency can be quite high. Region B tends to use more predictor bits for higher precisions with less stages. We synthesized those two designs and profiled their execution cycle number in Fig. 12. As we can see, it will cost 2–3 cycles for design A to complete one operation in a 1024-bit adder, while design B spent 1.001–1.01 cycle number for the same operation with much lower clock frequency. Based on the Design Compiler's results, the energy per operation for design A and B is 8.01 pJ and 4.58 pJ at 0.5 V supply voltage, respectively. Although both designs provide the same performance, design B is 1.75x more power efficient than design A due to the lower frequency and less operation cycles. Therefore, region B should be more preferable for low power applications. As a comparison, the energy consumption per operation of a 1024-bit Designware adder with same performance is 6.03 pJ, which is 27.7% higher compared with our proposed adder in Region B.

7.4 Performance Analysis and Optimization for Specific Distribution Inputs

The optimization result for a 64-bit adder design is shown in Table 3. The table summarize the number of stages m, the width of the m-th stage n^m, the predictor length and type

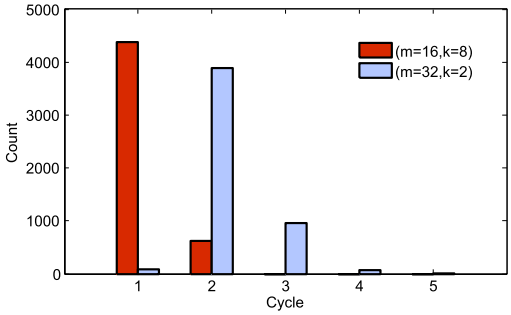


Fig. 12 Histogram of execution cycle number in two regions.

Table 3 64 bit adder design.

λ	m	n^m	(k^{m-1}, d^{m-1})	$\tau_{ave}(\tau'_{ave})$	gain
i.u.d.	8	8	(5,0)	−(15.53)	-
1	8	8	(5,0)	15.54(15.54)	1
10^2	8	8	(5,0)	15.58(15.58)	1
10^4	8	8	(4,1)	15.32(21.24)	1.39
10^6	7	11	(3,1)	14.90(31.84)	2.13

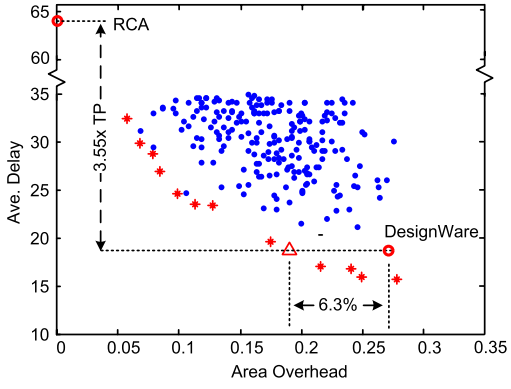


Fig. 13 Average delay and area overhead of the proposed adder.

of the $(m-1)$-th stage (k^{m-1}, d^{m-1}), and the average delay τ_{ave} of the optimized design under different input distribution parameter λ. An optimized design for *i.u.d.* is used as a baseline. When parameter λ increases, the *Type 1* is used instead. The length of predictor is shortened as its high prediction accuracy, which increases the performance. As a comparison, we apply the specific input to the baseline adder which is optimized for *i.u.d.* inputs, and its delay τ'_{ave} is shown in the brackets. Due to the high conflict rate of the *Type 0* predictors, the average delay gets worse. Compared with the non-adaptive baseline design, the last column shows the gain by using the proposed method, which is defined as τ'_{ave}/τ_{ave}.

7.5 Throughput and Area Tradeoffs

Figure 13 shows the normalized average delay and area overhead for a 64-bit proposed adder with *i.u.d.* inputs. The dots indicate the adders with random configurations, and the asterisks indicate the adders following the methodology in Sect. 5.2. The sub-optimal configurations are always dominated by those configurations in the Pareto frontier. It validated that our design methodology can guide the multistage function speculation adder design effectively. The configuration marked by a triangle indicates the adder with the design setting in Sect. 5.3. For this design, we have a 3.55x throughput performance gain compared with the traditional RCA, and a 6.3% area reduction compared with Designware with the same throughput performance.

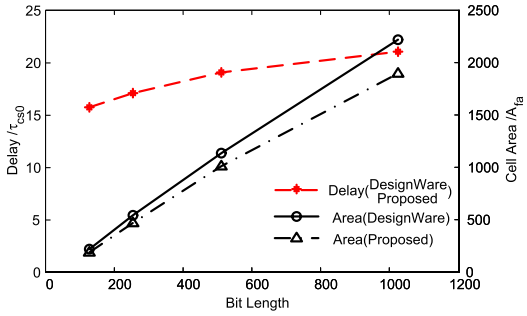


Fig. 14 Delay and area of the proposed and the DesignWare adder.

7.6 Measured Results of the Test Chip

We fabricated an adder to verify the proposed design methodology. The configuration of the adder is $N = 32$, $m = 4$, $k = 5$. The adder operates at 0.5 V supply voltage. We measured the minimal clock cycle of our chip. Experimental result is shown in Fig. 9, where T_D is the designed minimal clock cycle based on Eq. (7) and the delay information from the cell library. T_M is the measured minimal clock cycle from the test chip. *Type 0*, *1* and *H* indicate the type of the predictors in the adder respectively. As we can see, the relative trend of predictor delay is obeyed. The difference between the design and the measurement is due to the significant process variation under the low supply voltage.

7.7 Comparison with Other Adders

Figure 14 shows the delay and area of adders with different bit numbers from 128 to 1024. The delay and area are normalized by the carry delay τ_{cs0} and the area A_{fa} of a full adder. The plotted delay is for both the multi stage variable latency speculative adder and the adder generated by Designware; i.e., the proposed adder and the reference adder are compared for area at equal delay. As we can see, the delay for both adders obeys a $\log_2 N$ trend with the adder's bit number N. It should be noted that the delay for traditional adders is obtained by synthesis tools for optimal performance, which includes the register delay. While the delay for adders with function speculation is defined as the average-case delay. It means that the original clock delay of multistage function speculation adder would be multiplied by a constant (> 1) due to possible multiple cycle executions for fair comparison. The area values of the proposed architecture are 6–17% smaller than those of the fastest DesignWare adders ranging from 128 to 1024. The $O(\log N)$ delay and $O(N)$ area trends with the bit number N declared in Theorom 2 are also validated in Fig. 14. Besides the

above advantage, when a certain level of calculation error is allowed the proposed adder can provide much better performance and smaller area. These almost correct units are quite promising for algorithm level fault-tolerant calculations, such as data mining and machine learning.

8. Conclusions

Multistage function speculation design is quite promising to offer superior average-case performance. In this paper, we propose performance and area models for multistage function speculation adders, based on which, a general methodology is presented to guide design optimization. Both analytical and experimental results validate our models and the design methodology. Experiments showed that the proposed adder's delay and area has a logarithmic and linear relationship with its bit number, respectively. Compared with the DesignWare IP, the proposed adder provides the same performance with 6–17% area reductions under different bit number configurations.

Acknowledgement

This work was supported in part by the NSFC under grant 61271269 and High-Tech Research and Development (863) Program under contract 2013AA01320 and Huawei Shannon Lab and the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions under contract YETP0102.

References

[1] Y. Liu, Y. Sun, Y. Zhu, and H. Yang, "Design methodology of variable latency adders with multistage function speculation," 2010 11th International Symposium on Quality Electronic Design (ISQED), pp.824–830, 2010.
[2] I. Koren, Computer Arithmetic Algorithms, Prentice-Hall, New Jersey, 1993.
[3] A. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," Proc. Design, Automation and Test in Europe Conf, pp.1250–1255, 2008.
[4] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, and K. Flautner, "Razor: A low-power pipeline based on circuit-level timing speculation," Proc. Int. Symp. Microarchitecture, pp.7–18, 2003.
[5] W. Shan, H. Gu, B. Li, X. Wu, H. Jin, Y. Guo, and P. Cao, "An improved timing monitor for deep dynamic voltage scaling system," IEICE Electron. Express, vol.10, no.6, pp.1–7, 2013.
[6] L. Leem, H. Cho, J. Bau, Q. Jacobson, and S. Mitra, "Ersa: Error resilient system architecture for probabilistic applications," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010, pp.1560–1565, 2010.
[7] S. Nowick, "Design of a low-latency asynchronous adder using speculative completion," IEE Proc. Computers and Digital Techniques, vol.143, no.5, pp.301–308, 1996.
[8] S. Lu, "Speeding up processing with approximation circuits," IEEE Trans. Comput., vol.37, no.3, pp.67–73, 2004.
[9] Y. Chen, H. Li, J. Li, and C. Koh, "Variable-latency adder (VL-adder): New arithmetic circuit design practice to overcome NBTI," Proc. Int. Symp. Low Power Electronics and Design, pp.195–200, 2007.
[10] D. Baneres, J. Cortadella, and M. Kishinevsky, "Variable-latency design by function speculation," Proc. Design, Automation and Test in Europe Conf, pp.1704–1709, 2009.
[11] A. Del Barrio, R. Hermida, S. Memik, J. Mendias, and M. Molina, "Multispeculative addition applied to datapath synthesis," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.31, no.12, pp.1817–1830, 2012.
[12] A.A. Del Barrio, R. Hermida, S.O. Memik, J.M. Mendías, and M.C. Molina, "Improving circuit performance with multispeculative additive trees in high-level synthesis," Microelectronics Journal, vol.45, no.11, pp.1470–1479, 2014.
[13] A. Cilardo, "A new speculative addition architecture suitable for two's complement operations," Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE'09, pp.664–669, 2009.

Appendix

This section is to show how to characterize the distribution. The performance of the proposed adder relies on the distribution of the input. The inputs are normalized to $[-1/2, 1/2)$ by a factor of 2^N with a resolution of $1/2^N$. We use probability density function(pdf) to describe the input pattern of the proposed adder. If the input (a_n, b_n) is not correlated with the previous input (a_i, b_i), $i < n$, the pdf $f(a_n, b_n)$ is used. Otherwise, a first order Markov model $f(a_n, b_n, a_{n-1}, b_{n-1})$ is used as the behavior of the adder is determined by the current and the very last inputs only.

Independent uniform distribution is one of the simplest models, in which the inputs are in the uniform distribution and have no correlations. In this case, $f(a, b) = 1$ for any (a, b). In some applications, the input distribution differs. For example, Poisson process is used as a model of radioactive delay and other physical phenomena. The interval time, which is the time between each pair of consecutive events, has an exponential distribution. Equation (A· 1) shows the distribution of an exponential distribution, where λ is the distribution parameter.

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases} \quad (\text{A}\cdot 1)$$

When we consider the binary wrap overflow, the input's pdf of the adder which calculates the interval time is

$$f(a, b) = \frac{\lambda}{1 - e^{-\lambda}} e^{-\lambda [\![a+b]\!]} \quad (\text{A}\cdot 2)$$

where $[\![x]\!]$ is defined as $[\![x]\!] = x - \lceil x \rceil$.

Figure A· 1 is a scatter plot of the adder's input, where $\lambda = 30$.

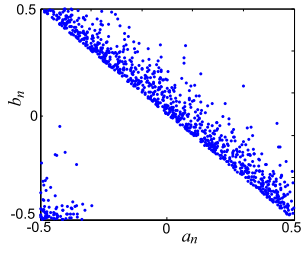
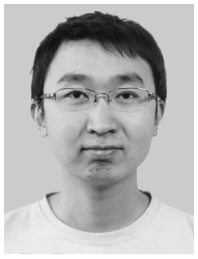
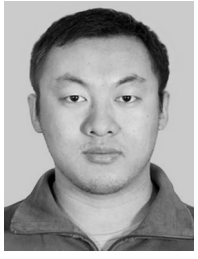


Fig. A· 1 Scatter plot of the adder's inputs.



Yinan Sun was born in 1986. He received his B.S. from Electronic Engineering Department, Tsinghua University in 2008 and now is a Ph.D. candidate in Electronic Engineering Department, Tsinghua University. His research interests are low power circuit and architecture design of embedded systems. He is now working in the project of a self-powered WSN sensor platform design.

Yongpan Liu was born in 1977. He received his B.S., M.S. and Ph.D. degrees from Electronic Engineering Department, Tsinghua University in 1999, 2002 and 2007. He worked as a research fellow in Tsinghua University from 2002 to 2004. He is now an associate professor in Dept. of Electronic Engineering Tsinghua University. His main research interests include low power VLSI design, emerging device based circuits and systems and design automation. His research is supported by NSFC, 863, 973 Program and Industry Companies such as Intel, Rohm, IBM and so on. He has published over 50 peer-reviewed conference and journal papers and led over 6 SoC design projects for sensor applications and has received the ISLPED2012, ISLPED2013 Design Contest Award and several Best Paper Candidates. He is an IEICE and IEEE member and has been invited to serve on several conference technical program committees (ASP-DAC, ISLPED, ICCD, A-SSCC).



Zhibo Wang was born in 1990. He received his B.S. from Electronic Engineering Department, Tsinghua University in 2013 and now is a Ph.D. candidate in Electronic Engineering Department, Tsinghua University. His research interests are near-threshold circuit and non-volatile computing.

Huazhong Yang was born in 1967. He received B.S. degree in microelectronics in 1989, M.S. and Ph.D. degrees in electronic engineering in 1993 and 1998, respectively, all from Tsinghua University, Beijing. In 1993, he joined the Department of Electronic Engineering, Tsinghua University, Beijing, where he is a full professor since 1998. Dr. Yang is a specially-appointed professor of the Cheung Kong Scholars Program. His current interest includes wireless sensor networks, data converters, parallel circuit simulation algorithms, nonvolatile processors and energy-harvesting circuits. Dr. Yang has authored and co-authored over 300 technical papers and 70 granted patents.