# Accelerating Unary Bit-Stream Processing Using Residue Numbers

Kamyar Givaki[1], Reza Hojabr[1], M. Hassan Najafi[2], Ahmad Khonsari[1,3], M. H. Gholamrezayi[4], Saeid Gorgin[5], Dara Rahmati[4]
[1]School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran
[2]School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA, USA
[3]School of Computer Sciences, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[4] Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran
[5] Iranian Research Organization for Science and Technology (IROST), Tehran, Iran
{givakik, r.hojabr}@ut.ac.ir, najafi@louisiana.edu, ak@ipm.ir, m.gholamrezayi@mail.sbu.ac.ir, gorgin@irost.ir, dara.rahmati@ieee.org

***Abstract*— Inaccuracy of computations is an important challenge with the Stochastic Computing (SC) paradigm. Recently, *deterministic* approaches to SC are proposed to produce completely accurate results with SC circuits. Instead of random bit-streams, the computations are performed on structured deterministic bit-streams. However, current deterministic methods take a large number of clock cycles to produce correct result. This long processing time directly translates to a very high energy consumption. In this work, we propose a design methodology based on the Residue Number Systems (RNS) to mitigate the long processing time of the deterministic methods. Compared to the state-of-the-art deterministic methods of SC our proposed approach delivers 760× and 170× improvements in terms of processing time and energy consumption, respectively.**

***Index Terms*—Stochastic computing, unary processing, deterministic bit-stream computing, residue number system, accelerator.**

## I. Introduction

Stochastic computing [1]–[3], an unconventional computing paradigm processing bit-streams, has been recently gained considerable attention in the hardware design and computer architecture communities. Higher noise tolerance and lower hardware cost compared to the conventional binary-radix-based designs are the most appealing advantages of SC. Inaccurate computation and long processing time, however, are the two major drawbacks of the conventional SC-based designs. Recently, some *deterministic* approaches to SC have been proposed to perform deterministic and completely accurate computations with SC circuits [4], [5].

The deterministic methods proposed in [4] and [5] offer completely accurate computation by processing *unary* bit-streams (bit-streams with first all 1's followed by all 0's). As depicted in Fig. 1, unary bit-streams can be generated by comparing an increasing/decreasing number from an up/down counter and a constant number based on the target value. To produce statistically independent bit-streams (which is a requirement for many SC-based operations), three approaches are proposed: rotation of bit-streams [4], clock dividing bit-streams [4], and using bit-streams with relatively prime lengths [5]. These methods guarantee high accuracy computation by processing bit-streams for a large number of clock cycles (i.e., the product of the length of the input bit-streams). For instance, to multiply two $n$-bit precision data represented using two bit-streams of length $2^n$, the deterministic methods takes $2^{2n}$ cycles to produce correct result. Although the deterministic methods outperform the conventional SC in term of computation accuracy, the long processing time and consequently, high energy consumption (power × time) limit their application. This work seeks a novel approach to mitigate the high processing time of the current deterministic methods of processing bit-streams. We integrate the deterministic methods with the Residue Number Systems (RNS) [6] to improve the performance of deterministic bit-stream computing.

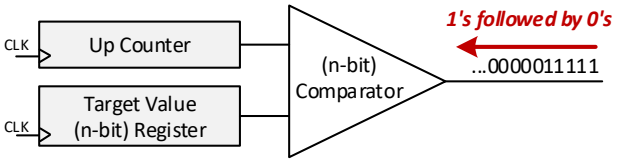


Fig. 1. Structure of a unary bit-stream generator.

## II. Motivation and Background

### A. Deterministic SC

Deterministic methods of processing bit-streams [4], [5] outperform the conventional SC due to their capability in producing completely accurate results. Random fluctuations in generating bit-streams and correlation between bit-streams are the main sources of inaccuracy in the conventional SC. Relying on deterministically generated unary bit-streams, the deterministic methods proposed in [4] and [5] remove the randomness requirement of processing bit-streams. By properly structuring bit-streams, these methods produce completely accurate results when the operation runs for the product of the length of the bit-streams. Three methods are proposed based on unary bit-streams: rotation [4], clock division [4], and relatively prime bit-stream lengths [5]. Fig. 2 exemplifies these three methods. All these approaches follow the same rule: every bit of one input bit-stream meets every bit of the other input bit-stream exactly once.

### B. Residue Number System

RNS is a number representation system in which a number is represented by its residues to the members of a set of relatively prime integers (called *moduli set*). Formal definition of RNS can be written as follows: considering a set of $L$ relatively prime numbers $< m_1, m_2, ..., m_L >$, an integer value $x$ is represented by $< x_1, x_2, ..., x_L >$ where $x_i = x \ mod \ m_i$. An important advantage of using the RNS is that complex and costly operations can be split into several independent and simple operations running in parallel [6], [7]. For example, a 2-input multiplication ($x \times y$) can be performed in three steps:

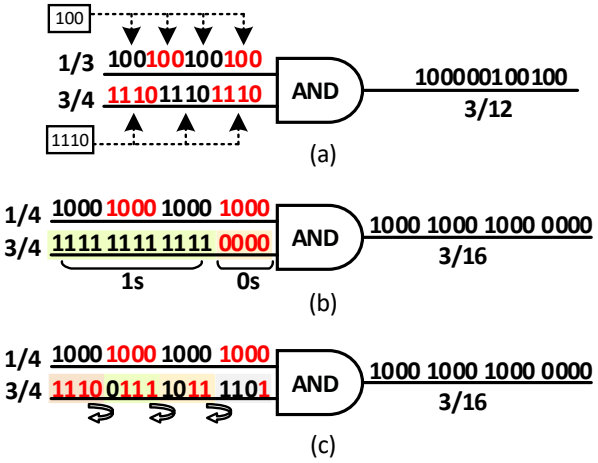


Fig. 2. Examples of the multiplication operation using the three deterministic methods: (a) relatively prime stream length, (b) clock division and (c) rotation.

- Convert the two integer operands, $x$ and $y$, to their RNS representation.
- Multiply each pair of $(x_i, y_i)$
- Convert the multiplication result back to the weighted binary.

This computation flow can be also extended to other arithmetic operations such as addition and subtraction. Considering the fact that the residues are significantly smaller than the original numbers, they require a lower bit-width to be represented.

An RNS system with moduli set $< m_1, m_2, ..., m_L >$ represents $M = m_1 \times m_2 \times ... \times m_L$ different numbers. $M$ is called the *dynamic range* of the system. To represent an $n$-bit binary number, the dynamic range of the selected moduli set should be greater than $2^n$ to guarantee that all numbers in the range of [0, $2^n - 1$] can be covered by the selected moduli set. For example, $< 8, 7, 5 >$ is a moduli set with a dynamic range equal to $280 = 8 \times 7 \times 5$ that can be used to represent 8-bit precision weighted binary numbers.

*Example 1:* considering the moduli set $< 8, 7, 5 >$ and two operands $x = 17$ and $y = 13$, multiplication of these two operands can be performed as follows. $x = 17$ is represented by $< 2, 3, 1 >$, where 2, 3, and 1 are the residues when dividing 17 by the modulus 8, 7 and 5, respectively. Also, $y = 13$ is represented by $< 3, 6, 5 >$. A drawback of using the RNS can be seen in this example. Representing an 8-bit binary number using the RNS format requires 9 bits (each modulus needs 3 bits). As shown in Fig. 3, the multiplication and addition operations are each divided into three simpler operations running in parallel. In multiplication of modulus $m_2 = 7$, $3 \times 6 = 18$ is greater than 7 and hence the result must be changed to 4. So, the operation in each modulus must be performed using modular operators such that the result of the operation is within [0, $m_i$]. Designing such a circuit is difficult and one of the challenges with the RNS. Chinese Reminder Theorem (CRT) [8] is a solution to convert a number in the RNS format to its corresponding weighted binary representation.

## III. Our Proposed Method

In this section, we propose a novel approach to tackle the long processing time problem of the deterministic bit-stream computing. We combine the deterministic methods of processing bit-streams with the RNS to reduce the number of clock cycles required to perform accurate computation. The total processing time of the bit-stream-based methods is found by multiplying the number of clock cycles needed to complete an operation and the critical path latency of the circuit. In deterministic bit-stream computing, the number of clock cycles required to perform an operation follows an exponential function of the bit-width of the operands. Using high-bit-width binary numbers as the operands of the deterministic methods lead to very long processing time and hence very high energy consumption. Exploiting the inherent parallel nature of the RNS, we split the high-bit-width operands of the computation into operands (residues) with lower bit-widths.

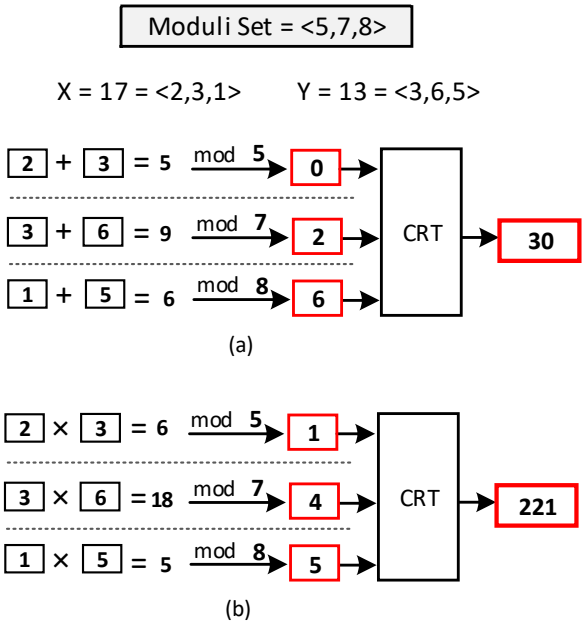


Fig. 3. (a) A two-input addition and (b) a two-input multiplication using RNS.

*Example 2:* Given two 8-bit precision inputs in the weighted binary format, multiplication using the deterministic methods needs $2^{16}$ clock cycles to produce completely accurate result [4]. Our proposed method that combines the deterministic methods with the RNS with moduli set $M =< 8, 7, 5 >$ requires a significantly smaller number of clock cycles (e.g., $2^6$ cycles). Table I compares our proposed method and prior deterministic methods in term of the number of clock cycles for five different cases. Our RNS-based method needs a CRT module to convert the result back to the weighted binary format. This module imposes an area and power overhead to the proposed method compared to the methods of [4] and [5]. The energy saving due to an exponential reduction in the number of clock cycles, however, compensates the overhead and results in a significantly lower total energy consumption.

### A. Hardware Architecture

Fig. 4 demonstrates the high-level block diagram of the proposed architecture to perform an $n$-bit precision multiplication. The moduli set is $< m_1, m_2, ..., m_L >$. An input buffer is used to store two distinct numbers, $A$ and $B$, in the RNS format. These

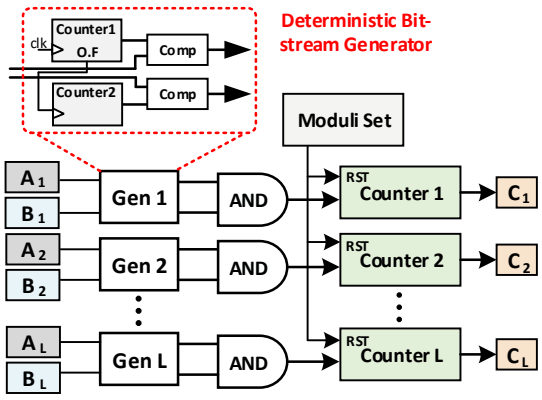


Fig. 4. Integrating deterministic SC multiplication with RNS

TABLE I
CLOCK CYCLE COMPARISON OF THE PROPOSED METHOD AND PRIOR DETERMINISTIC METHODS

| Binary Precision | Moduli Set | Prior Deterministic [4], [5] | Proposed Method | Clock Reduction |
|---|---|---|---|---|
| 8 bits | 8,7,5 | $2^{16}$ | $2^{6}$ | $2^{10}$ |
| 16 bits | 16,15,13,11,7 | $2^{30}$ | $2^{8}$ | $2^{22}$ |
| 16 bits | 64,63,61 | $2^{32}$ | $2^{12}$ | $2^{20}$ |
| 24 bits | 32,31,29,27,25 | $2^{48}$ | $2^{10}$ | $2^{38}$ |
| 24 bits | 64,63,61,59,55 | $2^{48}$ | $2^{12}$ | $2^{36}$ |

numbers are injected as the inputs of the circuit. $L$ bit-stream generators (*Gen 1*,...,*Gen L*) are used to generate deterministic bit-streams based on one of the three deterministic methods proposed in [4] and [5]. Using a proper moduli set, the needed bit-width in our proposed method is slightly more than that of other deterministic methods. Our experimental results show that this difference does not exceed three bits in most cases. As shown in Fig. 4, $L$ parallel AND gates are employed to perform the multiplication operations. Each AND gate is followed by a counter to convert the produced bit-stream to binary representation by counting the number of 1s in the bit-stream. At the same time, to calculate the residues, the reset signal of each counter is connected to the corresponding modulus. Due to using simple AND gates, the design and implementation of the modular multiplier in our approach is simpler than implementing the conventional binary modular multipliers. To convert the final results to the weighted binary format, a CRT module is used. The hardware overhead of the CRT module is relatively high. However, in case of using several multiplications in large applications (such as image processing), the RNS-to-binary conversion can be confined to the last stages of computation. This inefficiency can then be compensated by the performance improvement offered by the proposed RNS-based method.

## IV. Experimental Results

We implemented the proposed architecture in RTL VHDL and synthesized by the Synopsys Design Compiler using a 45 nm technology library. Synthesis results in terms of power consumption at maximum working frequency, critical path latency, and energy consumption for the case of implementing a 2-input multiplier with the proposed design and with the prior deterministic bit-stream processing are reported in Table II. For the prior deterministic method we implemented the clock division method of [4]. For the proposed design, we measured the processing time and the energy consumption of the bit-stream-based part and the CRT module, separately, and then accumulated the numbers to find the total processing time (*PT*) and total energy consumption. For the bit-stream-based part, the processing time and energy consumption are functions of the number of clock cycles.

TABLE II
THE SYNTHESIS RESULTS IN 45NM TECHNOLOGY: POWER ($\mu W$ ), CRITICAL PATH LATENCY ($nS$), ENERGY($pJ$), AND PROCESSING TIME ($nS$).

| | Arch. | Power ($\mu W$) | CP ($nS$) | # of cycles | Energy ($nJ$) | PT ($nS$) |
|---|---|---|---|---|---|---|
| 8-bit | 8,7,5 | 165 | 1.29 | $2^{6}$ | 108.19 | 186 |
| | Deter. Clk Div [4] | 131 | 2.16 | $2^{16}$ | 1.86E+4 | 1.42E+5 |
| 16-bit | 16,15,13,11,7 | 208 | 1.49 | $2^{8}$ | 261.81 | 532 |
| | Deter. Clk Div [4] | 232 | 3.44 | $2^{32}$ | 3.43E+9 | 1.48E+10 |
| 16-bit | 64,63,61 | 307 | 1.73 | $2^{12}$ | 3203.17 | 7839 |
| | Deter. Clk Div [4] | 232 | 3.44 | $2^{32}$ | 3.43E+9 | 1.48E+10 |

As reported in Table II, the proposed design outperforms its counterpart in terms of energy consumption and processing time. The proposed RNS-based design offers 172× saving in energy and 763× speedup compared to the clock division method of [4]. Table II also shows that the energy and processing time reduction for a specific bit-width depends on the selected moduli set. Due to using some additional gates and the CRT module, the power consumption of the proposed design is higher than that of the prior deterministic design. However, since the required number of clock cycles with our method is much less than the conventional deterministic method, our work outperforms its counterpart in terms of energy consumption and processing time.

## V. Conclusion

In this work, we proposed a novel approach to improve the processing time and hence the energy consumption of the current methods of deterministic bit-stream processing. Integrating the RNS with the deterministic methods, 763× speedup is achieved. The proposed method also reduces the energy consumption by a factor of 172×.

## References


[1] B. Gaines, "Stochastic computing systems," in *Advances in Information Systems Science*. Springer US, 1969, pp. 37–172.

[2] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Transactions on Embedded computing systems (TECS)*, vol. 12, no. 2s, p. 92, 2013.

[3] W. Qian, X. Li, M. Riedel, K. Bazargan, and D. Lilja, "An Architecture for Fault-Tolerant Computation with Stochastic Logic," *Computers, IEEE Trans. on*, vol. 60, no. 1, pp. 93–105, Jan 2011.

[4] D. Jenson and M. Riedel, "A deterministic approach to stochastic computation," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2016, pp. 1–8.

[5] M. H. Najafi, S. Jamali-Zavareh, D. J. Lilja, M. D. Riedel, K. Bazargan, and R. Harjani, "Time-Encoded Values for Highly Efficient Stochastic Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 5, pp. 1644–1657, May 2017.

[6] H. L. Garner, "The residue number system," in *Papers presented at the western joint computer conference*. ACM, 1959, pp. 146–153.

[7] B. Parhami, *Computer arithmetic*, vol. 20, no. 00.

[8] C.-H. Chang, A. S. Molahosseini, A. A. E. Zarandi, and T. F. Tay, "Residue number systems: A new paradigm to datapath optimization for low-power and high-performance digital signal processing applications," *IEEE circuits and systems magazine*, vol. 15, no. 4, pp. 26–44, 2015.