Reconvergent Path-aware Simulation of Bit-stream Processing

Sercan Aygun
University of Louisiana at Lafayette
Lafayette, LA, USA
sercan.aygun@louisiana.edu

M. Hassan Najafi
University of Louisiana at Lafayette
Lafayette, LA, USA
najafi@louisiana.edu

Mohsen Imani
University of California Irvine
Irvine, CA, USA
m.imani@uci.edu

Ece Olcay Gunes
Istanbul Technical University
Istanbul, Turkey
gunesec@itu.edu.tr

ABSTRACT

Few studies have explored the complex circuit simulation of stochastic and unary computing systems, which are referred to under the umbrella term of *bit-stream processing*. The computer simulation of multi-level cascaded circuits with reconvergent paths has not been largely examined in the context of bit-stream processing systems. This study addresses this gap and proposes a *contingency table*-based reconvergent path-aware simulation method for fast and efficient simulation of multi-level circuits. The proposed method exhibits significantly better runtime and accuracy.

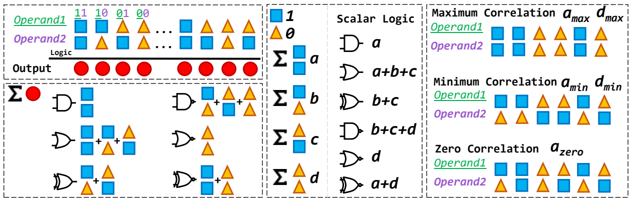
ACM Reference Format:
Sercan Aygun, M. Hassan Najafi, Mohsen Imani, and Ece Olcay Gunes. 2023. Reconvergent Path-aware Simulation of Bit-stream Processing . In *Proceedings of the Great Lakes Symposium on VLSI 2023 (GLSVLSI '23), June 5–7, 2023, Knoxville, TN, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3583781.3590323

1 INTRODUCTION

In recent years, bit-stream processing systems have emerged as an attractive alternative paradigm for achieving efficient and robust outcomes in a variety of applications [1]. The primitive data representation is a bit-stream, which accumulates values of logic-1s across a bit-stream of size N [5]. Bit-stream computing approach offers a new perspective on lightweight hardware design for machine learning but requires a testable environment for design space exploration [3]. To this end, prior works offered some frameworks for hardware and software simulation of bit-stream processing systems. A new tool, `CT`, short for "Contingency Table," imitates actual bit-streams but without generating long bit-level sequences. Instead, it works with scalar numbers [4]. In this work, we follow the design principles of `CT` for runtime- and memory-efficient simulation of bit-stream processing systems. However, we ❶ efficiently and rapidly simulate multi-level circuits for bit-stream processing, and ❷ consider reconvergent paths.

2 GLANCE AT SCALAR PROCESSING

Given the randomized and iterative nature of bit-stream size, N, developing a fast and efficient software method for bit-stream processing is challenging in terms of runtime and memory. Figure 1 depicts the processing of two bit-stream operands, where each bit overlaps at every position. Each bit undergoes a logic operation, and following N cycles, the output bit-stream is produced. Each bit in the output, 🔴, can then be accumulated. The results derived from different logic operations can be expressed via the cumulative counts of overlapping elements: 🟦🟦, 🟦▲, ▲🟦, and ▲▲. The total numbers of symbol overlaps are denoted as a, b, c, and d. Consider the output bit-stream of the `AND` operation, which is based on the total occurrence of 🟦 🟦 between the operands. Hence, the total number of 1s at the output accumulation, denoted by $\sum$🔴, is equal to a. By establishing relationship between symbols, input scalars, and N, the bit-stream generation and logic processing can be eliminated, as scalar logic exists, as shown in Figure 1. The crucial argument for establishing a direct relation is founded upon *correlation*. The design process can be effectively initiated through: *maximum*, *minimum*, and *zero* correlation. Figure 1 elucidates the cases with maximally, minimally correlated, or uncorrelated operands while taking into account the overlappings of identical symbols: 🟦 🟦 and ▲ ▲. The implementation of maximum correlation guarantees that a reaches its maximum value, whereas minimum correlation ensures its minimum. Therefore, symbol a assumes critical significance. The symbol a and the zero correlation case are intrinsically linked through the utilization of cross-correlation optimization. The formulas are presented in Figure 1.



$a_{max} = min(Op1, Op2)$	$b = Op1 - a$
$a_{min} = max(0, Op1 + Op2 - N)$	$c = Op2 - a$
$a_{zero} = \lfloor (Op1 \times Op2)/N \rceil$	$d = N - (a + b + c)$

Op: scalar operand

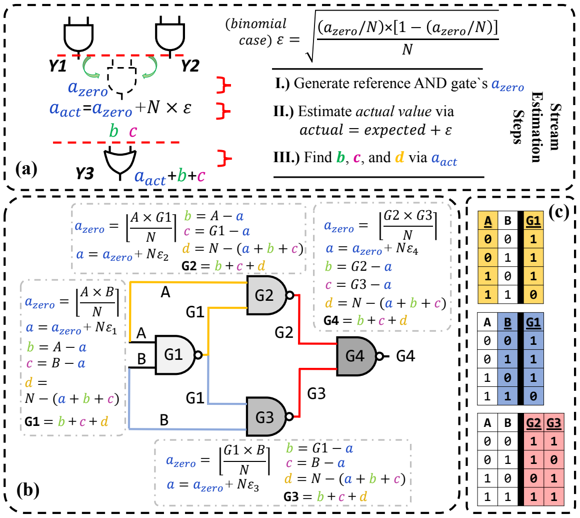
Figure 1: Scalar processing and formulas.

3 EXPLOITING RECONVERGENT PATHS

In-stream correlation manipulation is a growing trend in bit-stream processing [2]. Stream estimation has the potential to revolutionize bit-stream simulation tools, especially by bringing the reconvergence concept to digital bit-stream processing. For emulation of multi-level circuits, let us consider a 2-input (**X1** and **X2**) `AND` gate. The expected value at the gate's output, denoted by P_Y, is calculated as $\mathbb{E}[\boldsymbol{X1} \times \boldsymbol{X2}] = \mathbb{E}[\boldsymbol{X1}] \times \mathbb{E}[\boldsymbol{X2}] = P_{X1} \times P_{X2} = P_Y$, where the inputs are independent random variables. The difference between the exact value (P_Y) and the obtained value ($\hat{P_Y}$) reflects the random fluctuations error [1]. The squared error provides a measure of the random fluctuation, which can be quantified by

GLSVLSI '23, June 5–7, 2023, Knoxville, TN, USA

ACM ISBN 979-8-4007-0125-2/23/06.
https://doi.org/10.1145/3583781.3590323


(a)
Y1
Y2
Y3
a_{zero}
$a_{act}=a_{zero}+N\times\varepsilon$
b c
$a_{act}+b+c$
(binomial case) $\varepsilon=\sqrt{\frac{(a_{zero}/N)\times[1-(a_{zero}/N)]}{N}}$
I.) Generate reference AND gate`s a_{zero}
II.) Estimate *actual value* via $actual = expected + \varepsilon$
III.) Find b, c, and d via a_{act}
Stream Estimation Steps
(b)
$a_{zero}=\left\lfloor\frac{A\times B}{N}\right\rceil$
$a=a_{zero}+N\varepsilon_1$
$b=A-a$
$c=B-a$
$d=N-(a+b+c)$
G1 $= b+c+d$
$a_{zero}=\left\lfloor\frac{A\times G1}{N}\right\rceil$
$a=a_{zero}+N\varepsilon_2$
$b=A-a$
$c=G1-a$
$d=N-(a+b+c)$
G2 $= b+c+d$
$a_{zero}=\left\lfloor\frac{G2\times G3}{N}\right\rceil$
$a=a_{zero}+N\varepsilon_4$
$b=G2-a$
$c=G3-a$
$d=N-(a+b+c)$
G4 $= b+c+d$
$a_{zero}=\left\lfloor\frac{G1\times B}{N}\right\rceil$
$a=a_{zero}+N\varepsilon_3$
$b=G1-a$
$c=B-a$
$d=N-(a+b+c)$
G3 $= b+c+d$
A
B
G1
G2
G3
G4
(c)

A	B	G1
0	0	1
0	1	1
1	0	1
1	1	0

A	B	G1
0	0	1
0	1	1
1	0	1
1	1	0

A	B	G2	G3
0	0	1	1
0	1	1	0
1	0	0	1
1	1	1	1

Figure 2: Stream estimation (a), cascaded circuit simulation (b), and truth tables for reconvergent path analysis (c).

$\sigma^2 = P_Y \times (1-P_Y)/N$ for Bernoulli random variables. As σ is a function of P_Y and N, the type of circuit is unimportant, and the AND gate can serve as a reference gate due to its importance in scalar processing. Figure 2(a) illustrates the steps of multi-level circuit simulation using scalar processing and the proposed stream estimation steps. Zero correlation is used to obtain an ***expected*** value for the AND gate. This is essential since ϵ necessitates it for standard deviation calculation in the binomial case. That is, $\epsilon = \sigma = \sqrt{P_Y(1-P_Y)/N}$, where $P_Y = P_{Y1} \times P_{Y2} = \frac{a_{zero}}{N}$. Nevertheless, the expected value deviates from the ***actual*** value. To tackle this, we use the deviation calculation related to the random source through ϵ, and the expected value is estimated (via a_{zero}), giving rise to a_{act}. Figure 2(b) illustrates the stream estimation and also is an exemplar circuit (four-NANDs XOR) for reconvergence. The circuit paths that exhibit reconvergence are highlighted using different colors. Truth tables in Figure 2(c) are used to analyze the outputs of gates. We observe that in the yellow-colored loop, A and $G1$ never have a '00' state, resulting in signal correlation for $G2$. This holds true for gates $G3$ and $G4$. The outputs of the gates do not possess a '00' state in the colored loops, and there is a correlation; it is called reconvergence. We must set the corresponding d symbol (00) to a minimum, thus setting a to a minimum as well: $a_{min} = max(0, Op1 + Op2 - N)$. As a result, the formulas for $G2$, $G3$, and $G4$ in Figure 2(b) are initialized using a_{min} instead of a_{zero}.

4 SIMULATION RESULTS

Next, we simulate (i) Four-NANDs XOR and (ii) 2-bit ripple carry adder known in the literature for reconvergent paths [6]. The latter topology is more complex with four cascading levels and ten gates to analyze; **S**um and **C**arry outputs (**S0**, **S1**, **C1**, **C2**) are checked. Each topology is evaluated with two simulation approaches: *Sim1* for actual bit-streams and *Sim2* for scalar processing. We utilize binomial random sources. Over 10,000 random iterations, each *Sim* accepts random inputs either as bit-streams or directly as scalars. The checkpoints in each topology (**G1**, **G2**, **G3**, **G4** in (i) & **S0**, **S1**, **C1**, **C2** in (ii)) are compared using mean absolute error (MAE) between *Sim1* and *Sim2*; hence, the performance of simulator is measured with respect to real bit-streams. *Sim2* does not use the reconvergence concept and follows the always-set-a_{zero} approach in Figure 2(b). Nevertheless, one more simulation, *Sim3*, considers the reconvergent paths and initiates scalar processing symbols considering signal correlations. Figure 3 compares the MAE performance between *Sim2* and *Sim3*. The reconvergence-aware simulation gives better results, especially for the longer bit-stream size. Figure 3 also compares the runtime performance of *Sim1* and *Sim3*.

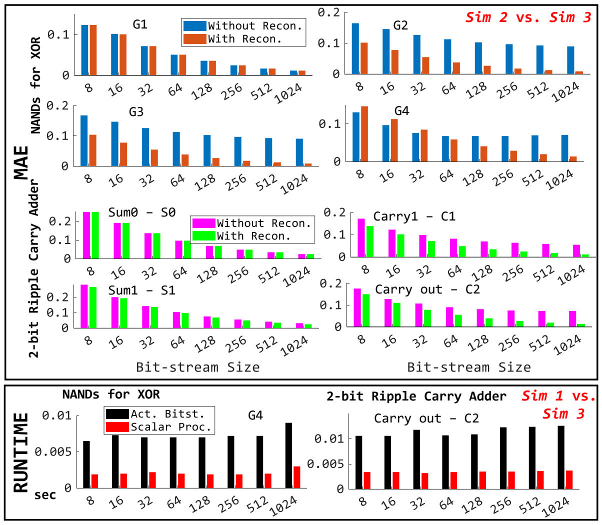


Figure 3: MAE and Runtime performance.

5 CONCLUSIONS

This study introduces two novel concepts for the simulation of bit-stream processing systems: stream estimation at mid-levels of cascaded logic circuits and reconvergence awareness for signal correlations. The proposed simulation model, which utilizes efficient estimation, results in a significantly faster runtime.

ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation (NSF) grants #2127780 and #2019511, Semiconductor Research Corporation (SRC), Office of Naval Research, grant #N00014-21-1-2225 and #N00014-22-1-2067, the Air Force Office of Scientific Research under award #FA9550-22-1-0253, the Louisiana Board of Regents Support Fund #LEQSF(2020-23)-RD-A-26, and generous gifts from Cisco, Xilinx, and Nvidia.

REFERENCES

[1] Armin Alaghi et al. 2018. The promise and challenge of stochastic computing. *IEEE TCAD* 37, 8 (2018), 1515–1531.
[2] Sina Asadi et al. 2021. CORLD: In-stream correlation manipulation for low-discrepancy stochastic computing. In *2021 ICCAD*. 1–9.
[3] Sercan Aygun. 2022. *Stochastic bitstream-based vision and learning machines*. Ph. D. Dissertation. Istanbul Technical University, Istanbul, Turkey.
[4] Sercan Aygun and Ece Olcay Gunes. 2022. Utilization of contingency tables in stochastic computing. *IEEE Transactions on Circuits and Systems II* 69, 6 (2022).
[5] S. Rasoul Faraji et al. 2019. Energy-efficient convolutional neural networks with deterministic bit-stream processing. In *2019 DATE*. 1757–1762.
[6] Torras Flaquer et al. 2010. Handling reconvergent paths using conditional probabilities in combinatorial logic netlist reliability estimation. In *2010 ICECS*.