Unary Positional Computing

McKenzie van der Hagen, Marc Riedel

University of Minnesota | Department of Electrical and Computer Engineering

1 Introduction

Numbers in digital logic computations are commonly represented in one of two ways: Binary or Stochastic. In 2016, a Deterministic approach to Stochastic computing was introduced as another alternative representation [1]. The benefits and downfalls of each of these approaches is summarized below.

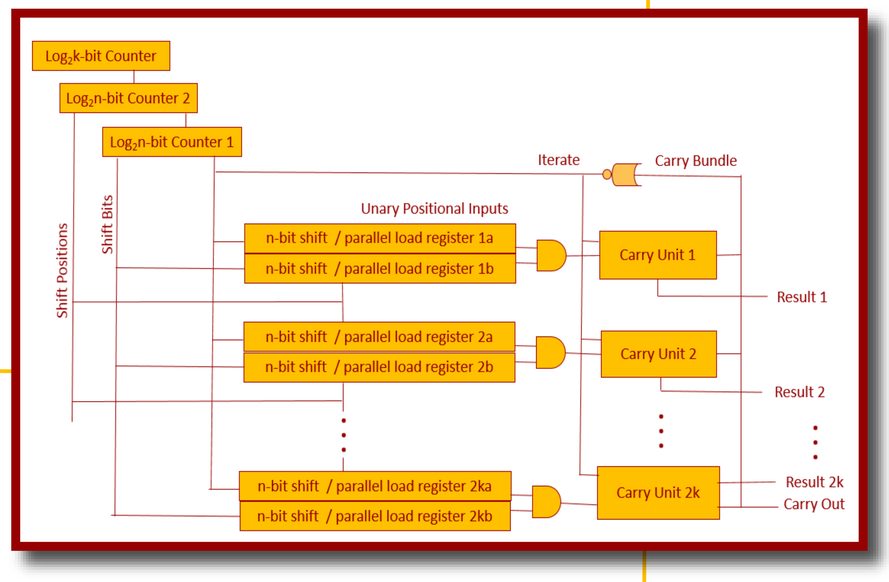
	Compact	Fault Tolerant	Simple Generation	Exactly Accurate	Simple Hardware
Binary	X		X	X	
Stochastic		X			X
Deterministic		X	X	X	X

In response to these results, this research builds upon the achievements of the Deterministic approach and addresses the remaining issue of long latencies associated with a non-compact representation. It does so by introducing a hybrid representation that includes both positional and uniform aspects. After developing a representation, hardware to perform simple arithmetic operations was implemented and analyzed.

2 Representation

The Unary Positional system uses k uniform bit streams of length n to represent numbers in base-n over the range $[0, n^k]$. The position of bits within a stream is insignificant, but the positions of each stream are weighted by increasing powers of n. An example is presented below with $n = 8$ and $k = 3$.

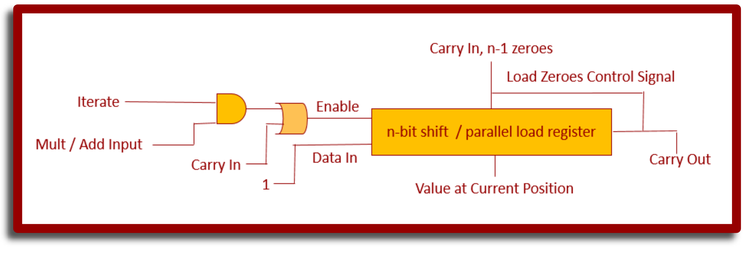
$$\underset{(5 \times 8^2) = 320}{00011111} + \underset{(2 \times 8^1) = 16}{00000011} + \underset{(4 \times 8^0) = 4}{00001111} = 340$$



Stochastic multiplication is performed with a single AND gate. The Deterministic approach produced exactly accurate results with this same hardware by rotating the input bit streams in such a way that each bit of one input lined up with each bit of the other input exactly once. This strategy is combined with the previously discussed carryover circuitry to perform 2-input Unary Positional multiplication as shown by the complete circuit diagram above. The stacked registers in the middle house the multiplier and multiplicand. Initially, the first position of the multiplier is multiplied with each position of the multiplicand. When this completes, the entire multiplicand is shifted one position, and the next portion of the multiplier is loaded in. This continues until all positions have been completed. Counters are used to account for this positional weighting, and to appropriately rotate the bits within each position. Addition is performed in a similar way, but instead of sending the inputs through an AND gate, they are simply concatenated together and fed into the carryover circuit.

3 Computational Hardware

With a representation in place, basic arithmetic operations including addition and multiplication were explored. Accounting for carryover is the most prominent challenge in performing either of these computations in any positional representation. Intuitively, carryover represents full groups of n bits that can be represented more compactly by using the weighting of the next higher position. Carryover in the Unary Positional representation is handled with a shift register and a collection of control signals as shown below. One of these Carry Units is used at each position of the computation.



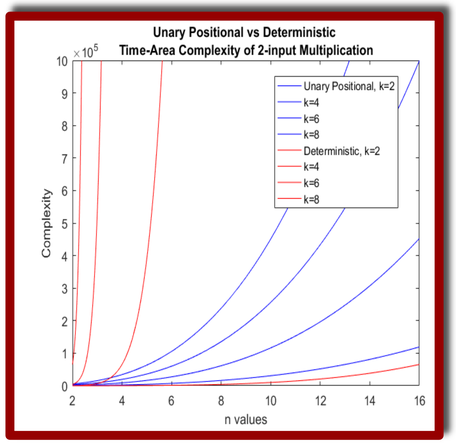
Resulting bits from multiplication, addition, or carry operations act as the enable to the shift register. In this way, 1 bits are stacked in to the register, while 0 bits are discarded. The last bit is thus recognized as a carryover bit, prompting the register at the current position to reset, the register at the next position to shift in a one, and registers at all other positions to pause.

[1] Jenson D, Riedel M. A Deterministic Approach to Stochastic Computation. *ICAAD '16*. Texas, USA. 2016 Nov.

4 Performance Evaluation

The representation and computational hardware described were implemented in Verilog and simulated to ensure correctness and evaluate performance. Evaluations were made in five main areas of interest including: compactness, cost of conversion/generation, fault tolerance, and time-space complexity. Comparisons made with respect to Deterministic and Binary computations were most thoroughly investigated. Results are presented below. In these results, n represents the base number or bit stream length, and k represents the number of positions in the Unary Positional number.

Time – Area Complexity



	Positional Binary	Deterministic		Una-Posi
Complexity of 2-input Multiplication	$[(\log_2 n)k]^2$	Time:	n^{2k}	kn^2
		Area:	1	$2\log_2 n + \log_2 k + 2k + 2k(6n + 6n + 3) + n$

Compactness & Generation / Conversion

Representation	Stochastic	Deterministic	Una-Posi	Binary
Bits to represent n^k distinct numbers	n^{2k}	n^k	nk	$(\log_2 n)k$
Cost of Generation / Conversion for i inputs	$12(\log_2 n)ki^2 + 3(\log_2 n)ki$	$9(\log_2 n)ki$ $15(\log_2 n)ki - 6(\log_2 n)k$	6nik	n/a

Fault Tolerance

Representation	Bits to represent n^k distinct numbers	Most significant impact of a single bit flip	Maximum unique representations of a single number
Deterministic	n^k	1	n^k
Una-Posi	nk	n^{k-1}	n
Binary	$(\log_2 n)k$	$(1/2)n^k$	1

As expected, Unary Positional computing, a hybrid between a fully positional and fully uniform system, bears complexities that fall between traditional binary and the Deterministic approach to Stochastic computing for all examined performance metrics.

5 Conclusion

Referring back to the evaluation criteria in the Introduction, Unary Positional computation has produced promising results. The representation boasts simple generation, high fault tolerance, and exact accuracy.

As desired, the Unary Positional representation did indeed boast exponentially shorter latencies than the Deterministic approach. Although this goal was achieved at a cost of more complex hardware, the overall time-space complexity is still highly desirable over the Deterministic approach. Moving forward, additional research is needed to identify what applications would benefit from this hybrid approach.