An Accurate and Compact Hyperbolic Tangent and Sigmoid Computation Based Stochastic Logic

***Van-Tinh Nguyen*[1], *Tieu-Khanh Luong*[2], *Emanuel Popovici*[2], *Quang-Kien Trinh*[3], *Renyuan Zhang*[1], *Yasuhiko Nakashima*[1]**

[1]NARA Institue of Science and Technology, [2]University College Cork, [3]Le Quy Don Technical University

International Midwest Symposium on Circuits and Systems

Outline

- ❑ **Motivation & Background**
- ❑ **Existing Methods**
- ❑ **Stochastic Logic Implementation Based Bernstein Polynomial**
- ❑ **Proposed Architecture**
- ❑ **Experimental Results & Comparisons**
- ❑ **Conclusions**

Outline

- ❑ **Motivation & Background**
- ❑ **Existing Methods**
- ❑ **Stochastic Logic Implementation Based Bernstein Polynomial**
- ❑ **Proposed Architecture**
- ❑ **Experimental Results & Comparisons**
- ❑ **Conclusions**

Motivation

Hyperbolic tangent and Sigmoid function are widely used, especially signal processing based application

- **High accuracy**
- **Reasonable hardware cost**
- **Fit to the stochastic end-to-end system**

Background

Stochastic computing

Generate a random sequence to represent a number based on fraction of the number 1's in bitstream

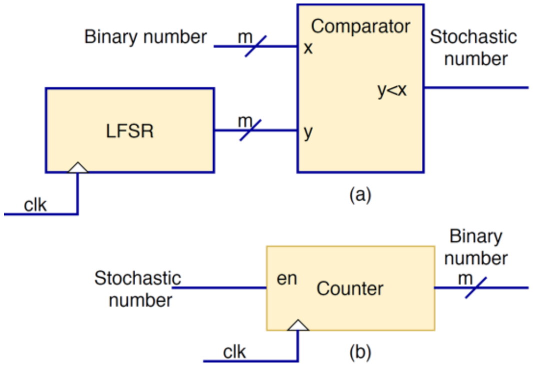
- **Unipolar format**

$$x = p(X = 1) = p(X)$$

- **Bipolar format**

$$x = p(X = 1) - 1 = 2p(X) - 1$$

Format conversion can be done the two format



(a) Stochastic number generator

(b) Converting stochastic number to binary number

Outline

- ✓ Motivation & Background
- ❑ Existing Methods
- ❑ Stochastic Logic Implementation Based Bernstein Polynomial
- ❑ Proposed Architecture
- ❑ Experimental Results & Comparisons
- ❑ Conclusions

Existing Methods

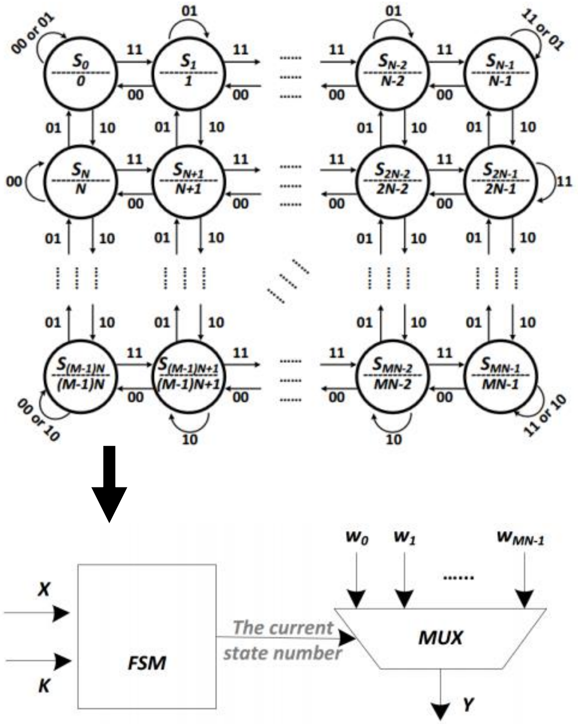
FSM based method

Pros

- Synthesize sophisticated funtions

Cons

- High hardware cost
- Low accuracy



Existing Methods

Series expansion and JK-FFs method

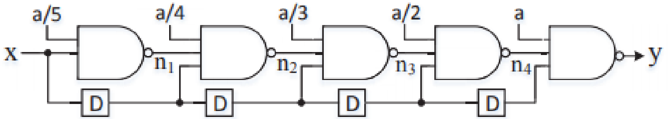
$$\tanh(ax) = \frac{e^{ax} - e^{-ax}}{e^{ax} + e^{-ax}} = \frac{1 - e^{-2ax}}{1 + e^{-2ax}}$$

Pros

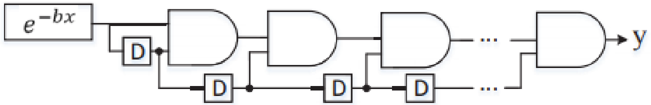
- End-to-end stochastic
- Reasonable hardware cost

Cons

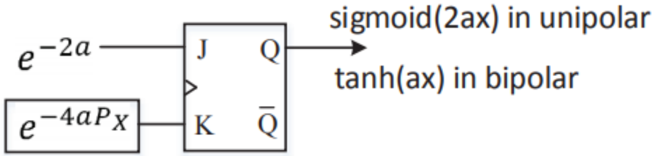
- Low accuracy



Stochastic implementation of e^{-ax} using Maclaurin polynomial



Stochastic implementation of e^{-ax} $(a > 1)$ using e^{-bx} cascaded



Stochastic implementation of $\tanh(ax)$ and $sigmoid(2ax)(a > 1)$ with bipolar input using format conversion

Existing Methods

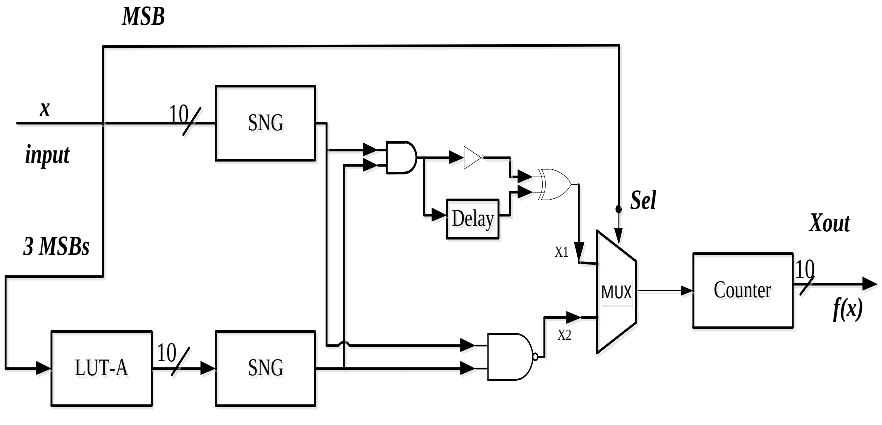
Piecewise linear approximation:

Pros

- Good accuracy
- Reasonable hardware cost

Cons

- Input format conversion from binary to stochastic



The architecture of stochastic implementation of the function e^{-2x}.

Outline

- ✓ **Motivation & Background**
- ✓ **Existing Methods**
- ❑ **Stochastic Logic Implementation Based Bernstein Polynomial**
- ❑ **Proposed Architecture**
- ❑ **Experimental Results & Comparisons**
- ❑ **Conclusions**

SC based Bernstein Polynomials

- **Polynomial functions $f(x)$ are done by using multiplications and additions**
- **It fails to compute the polynomial functions which computation range is outside [0, 1], e.g $1.2x - 1.2x^2$**
- **For any polynomial functions, transforming a power-form polynomial to a Bernstein polynomial:**

$$B(x) = \sum_{i=0}^{n} b_i B_{i,n}(x)$$

Where $B_{i,n}(x) = \binom{n}{i} x^i (1-x)^{n-i}$.

Outline

- ✓ Motivation & Background
- ✓ Existing Methods
- ✓ Stochastic Logic Implementation Based Bernstein Polynomial
- ❑ Proposed Architecture
- ❑ Experimental Results & Comparisons
- ❑ Conclusions

Proposed Architecture

Implementing hyperbolic tangent and sigmoid function in the bipolar format

- $tanh(ax) = \frac{1 - e^{-2ax}}{1 + e^{-2ax}} \quad a > 0$
- $sigmoid(2ax) = \frac{1}{1 + e^{-2ax}}$

$\Rightarrow tanh(ax) = 2\frac{1}{1+ e^{-2ax}} - 1$

$= 2sigmoid(2ax) - 1$ [1]

Proposed Architecture

Bipolar format: $\boldsymbol{x = 2P_x - 1}$

- $\boldsymbol{x \in [-1, 1]}$
- $\boldsymbol{P_x \in [0, 1]}$

$\Rightarrow$ **Format conversion between bipolar and unipolar format**

$\boldsymbol{sigmoid(2ax) \in [0, 1]}$ **=> output is unipolar format**

Applying format conversion to equation (1) to same

bitstream of $\boldsymbol{sigmoid(2ax) \Rightarrow tanh(ax)}$

Proposed Architecture

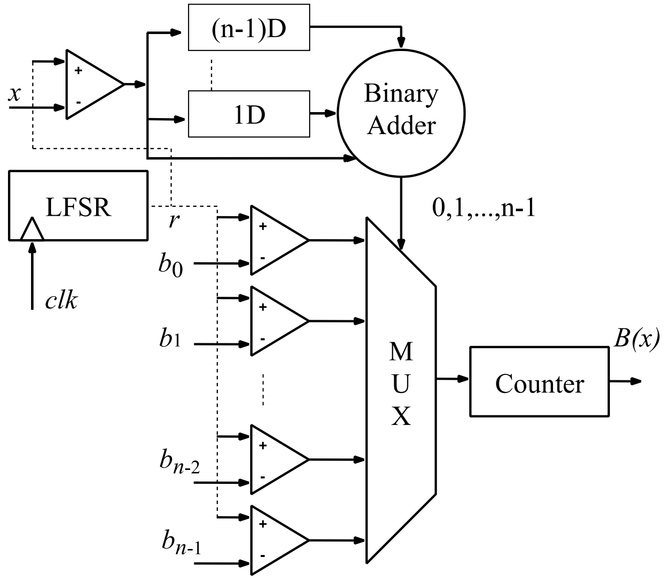
Impmentation of $sigmoid(2ax)$

- $$sigmoid(2ax) = \frac{1}{1+e^{-2ax}}$$

$$= \frac{1}{1+e^{-2a(2P_x-1)}} = \frac{1}{1+e^{-4aP_x}e^{-2a}}$$

$$= \frac{e^{-2a}}{e^{-2a}+e^{-4aP_x}}$$

- **The approximation can now be made by using Bernstein computations.**



Stochastic implementation of $Sigmoid(2ax)$ based Bernstein computation

Outline

- ✓ Motivation & Background
- ✓ Existing Methods
- ✓ Stochastic Logic Implementation Based Bernstein Polynomial
- ✓ Proposed Architecture
- ❑ Experimental Results & Comparisons
- ❑ Conclusions

Simulations & Experiments Set-up

<u>Design</u>

- $\boldsymbol{Sigmoid(2x), Sigmoid(4x), tanh(x), tanh(2x)}$
- **5th order Bernstein polynomial**

<u>Simulation</u>

- **Matlab**
- **10 bit LFSR**
- **Mean Absolute Error (MAE) results of approximated and target functions**
- **Monte Carlo experiments**

Implementation

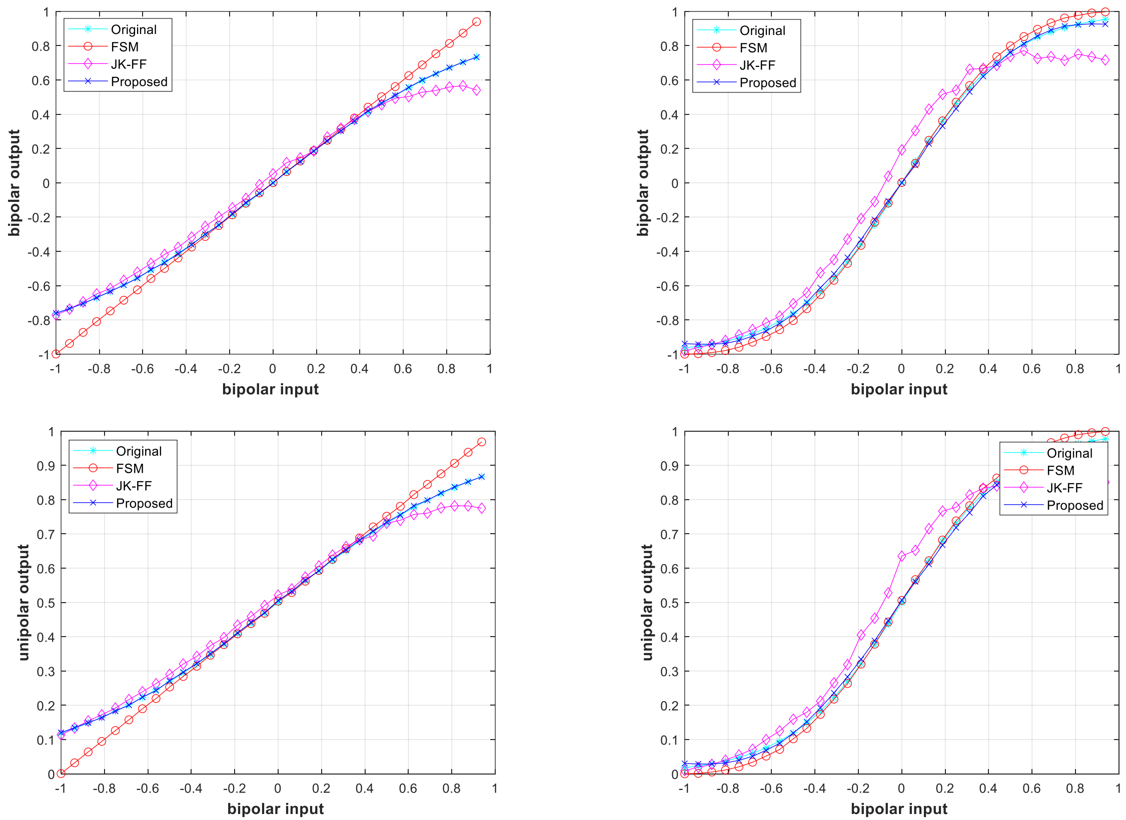
- **Synopsys DC using TSMC 180 nm**

Results & Comparison

Matlab simulation results

Function	Tanh(x) and sigmoid(2x)			
Method	Proposed		FSM [4]	JK-FF [6]
	n=3	n=5	2 states	-
MAE	0.003	0.001	0.06	0.02
Function	Tanh(2x) and sigmoid(4x)			
Method	Proposed		FSM [4]	JK-FF [6]
	n=3	n=5	2 states	-
MAE	0.007	0.003	0.03	0.05

Results & Comparison



Simulation results compared different approaches with target functions: $\mathbf{tanh}(\boldsymbol{x})$, $\mathbf{tanh}(\mathbf{2}\boldsymbol{x})$, $\mathbf{sigmoid}(\mathbf{2}\boldsymbol{x})$, $\mathbf{sigmoid}(\mathbf{4}\boldsymbol{x})$ **respectively**

Results & Comparison

Synopsys DC simulation results

Function	Tanh(x) and sigmoid(2x)			
Method	Proposed		FSM [4]	JK-FF [6]
	n=3	n=5	2 states	-
Area ($(\mu m)^2$)	1554	1777	1345	10121
Latency (ns)	2.25	2.33	2.38	3.42
Power (mW)	0.07	0.08	0.06	0.4

Results & Comparison

Synopsys DC simulation results

Function	Tanh(2x) and sigmoid(4x)			
Method	Proposed		FSM [4]	JK-FF [6]
	n=3	n=5	2 states	-
Area ($(\mu m)^2$)	1777	2106	1551	10476
Latency (ns)	2.33	3.3	3.07	3.07
Power (mW)	0.11	0.11	0.08	0.08

Outline

- ✓ Motivation & Background
- ✓ Existing Methods
- ✓ Stochastic Logic Implementation Based Bernstein Polynomial
- ✓ Proposed Architecture
- ✓ Experimental Results & Comparisons
- ❑ Conclusions

Conclusions

$sigmoid(2ax)$ **and** $tanh(ax)$ **based Bernstein polynomial in bipolar format:**

- **Comparable hardware complexity**
- **Improvement of accuracy**

Reference

1. B. R. Gaines, “Stochastic computing,” in Proceedings of AFIPS spring joint computer conference, pp. 149–156, ACM, 1967.
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. K. K. Parhi, “Analysis of stochastic logic circuits in unipolar, bipolar and hybrid formats,” 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, 2017, pp. 1-4.
4. B. D. Brown and H. C. Card, “Stochastic neural computation. I. computational elements,” IEEE Transactions on Computers, vol. 50, no. 9, pp. 891–905, 2001.
5. P. Li, D. J. Lilja, K. Bazargan and M. Riedel, "The synthesis of complex arithmetic computation on stochastic bit streams using sequential logic," in IEEE/ACM (ICCAD), San Jose, CA, USA, pp. 480-487, Dec. 2012.
6. Y. Liu and K. K. Parhi, ”Computing hyperbolic tangent and sigmoid functions using stochastic logic,” 2016 50th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, 2016, pp. 1580- 1585.
7. K. T. Luong, V. Nguyen, A. Nguyen and E. Popovici, "Efficient Architectures and Implementation of Arithmetic Functions Approximation Based Stochastic Computing," 2019 IEEE 30th International Conference on Application-specific Systems, Architectures and Processors (ASAP), New York, 2019, pp. 281-287.
8. W. Qian, X. Li, M. D. Riedel, K. Bazargan and D. J. Lilja, ”An Architecture for Fault-Tolerant Computation with Stochastic Logic,” in IEEE Transactions on Computers, vol. 60, no. 1, pp. 93-105, Jan. 2011

THANK YOU FOR YOUR ATTENTION