BSC: Block-based Stochastic Computing to Enable Accurate and Efficient TinyML

Yuhong Song[1] Edwin Hsing-Mean Sha[1,*] Qingfeng Zhuge[1] Rui Xu[1]
Yongzhuo Zhang[1] Bingzhe Li[2] Lei Yang[3]

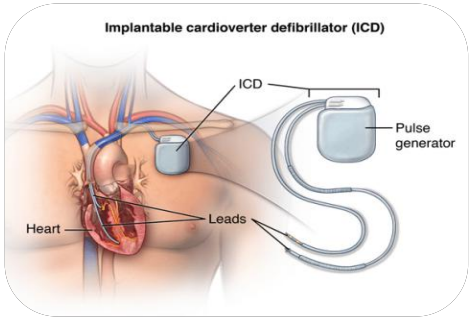
[1] East China Normal University [2] Oklahoma State University
[3] University of New Mexico

* Edwin Hsing-Mean Sha is the corresponding author (edwinsha@cs.ecnu.edu.cn)

THE UNIVERSITY OF NEW MEXICO

Background

Application Needs: There are increasing demands on the deployment of deep neural networks (DNNs) on tiny devices.



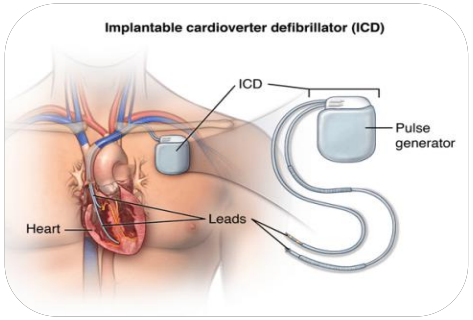
Medical Treatment

Surveillance Cameras

Smart Driving

Background

Application Needs: There are increasing demands on the deployment of deep neural networks (DNNs) on tiny devices.



Medical Treatment

Surveillance Cameras

Smart Driving

Hardware Constraint: the energy supply of tiny devices is extremely limited, which requires low-power execution for DNNs.

Background

Stochastic Computing (SC) stands out to reduce power consumption by simplifying computing circuits because of the special data expression.

Data representation

$$\underbrace{11010100}_{4}$$

Unipolar: $\frac{4}{8} = 0.5$

(range: [0, 1])

$$\underbrace{11010100}_{4}$$

Bipolar: $2 \times \frac{4}{8} - 1 = 0.0$

(range: [-1, 1])

Background

Stochastic Computing (SC) stands out to reduce power consumption by simplifying computing circuits because of the special data expression.

Data representation

$$\underbrace{11010100}_{}$$

Unipolar: $\frac{4}{8} = 0.5$

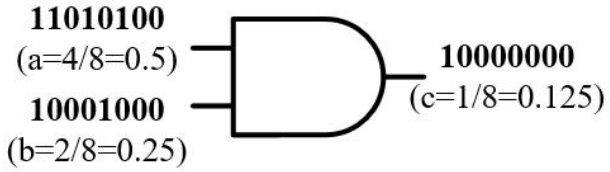
(range: [0, 1])

$$\underbrace{11010100}_{}$$

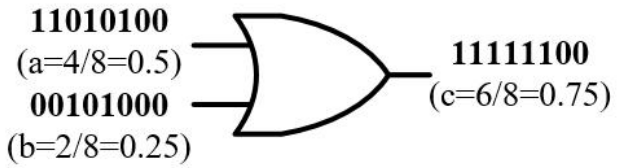
Bipolar: $2 \times \frac{4}{8} - 1 = 0.0$

(range: [-1, 1])

Arithmetic Circuit



Unipolar AND Multiplier



Unipolar OR Adder

Background

Stochastic Computing (SC) stands out to reduce power consumption by simplifying computing circuits because of the special data expression.

Data representation

11010100

Unipolar: $\frac{4}{8} = 0.5$

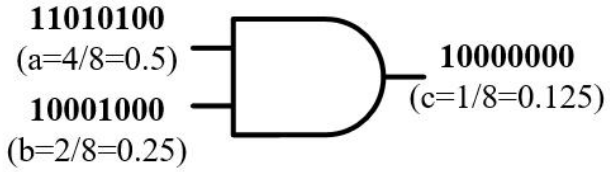
(range: [0, 1])

11010100

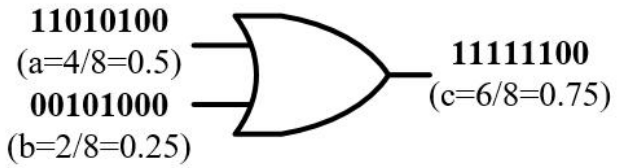
Bipolar: $2 \times \frac{4}{8} - 1 = 0.0$

(range: [-1, 1])

Arithmetic Circuit



Unipolar AND Multiplier



Unipolar OR Adder

SC greatly decreases the hardware cost in terms of area and power.

However, it suffers huge accuracy loss.

Motivation1

Low Data Precision leads to inaccurate data representation

bit length	*4 bits*	*16 bits*	*64 bits*	*256 bits*	*1024 bits*
	↓	↓	↓	↓	↓
data precision	0.125	0.0625	0.015625	0.00390625	0.0009765625

deterministic SC[1] ⟶ $2^{m \times \log_2 n}$-bit bitstreams to process m n-bits inputs

[1] W. Qian, X. Li, M. D. Riedel, K. Bazargan, et al., "An architecture for fault-tolerant computation with stochastic logic," IEEE transactions on computers (TC), vol. 60, no. 1, pp. 93–105, 2010.

Motivation1

Low Data Precision leads to inaccurate data representation

bit length	*4 bits*	*16 bits*	*64 bits*	*256 bits*	*1024 bits*
	↓	↓	↓	↓	↓
data precision	0.125	0.0625	0.015625	0.00390625	0.0009765625

deterministic SC[1] ⟶ $2^{m \times \log_2 n}$-bit bitstreams to process m n-bits inputs

High Data Precision Requires Long Computing Latency

[1] W. Qian, X. Li, M. D. Riedel, K. Bazargan, et al., "An architecture for fault-tolerant computation with stochastic logic," IEEE transactions on computers (TC), vol. 60, no. 1, pp. 93–105, 2010.

Motivation1

Low Data Precision leads to inaccurate data representation

bit length	*4 bits*	*16 bits*	*64 bits*	*256 bits*	*1024 bits*
	↓	↓	↓	↓	↓
data precision	0.125	0.0625	0.015625	0.00390625	0.0009765625

deterministic SC[1] ⟶ $2^{m \times \log_2 n}$-bit bitstreams to process m n-bits inputs

High Data Precision Requires Long Computing Latency

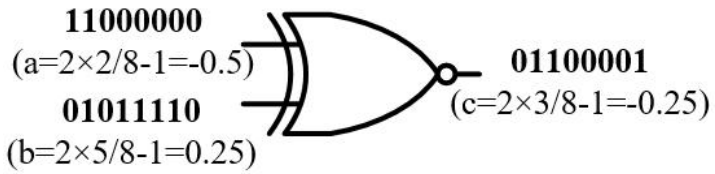
To shorten latency, we propose a block-based architecture, namely BSC, which divides inputs into blocks and executes blocks in parallel.

[1] W. Qian, X. Li, M. D. Riedel, K. Bazargan, et al., "An architecture for fault-tolerant computation with stochastic logic," IEEE transactions on computers (TC), vol. 60, no. 1, pp. 93–105, 2010.

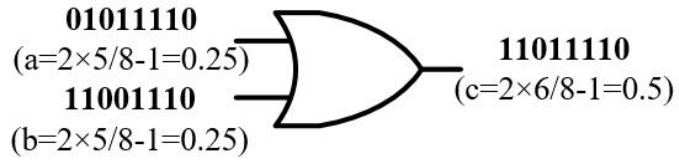
Motivation2

Arithmetic circuits affect the accuracy of computation

- Basic XNOR Multiplier and OR-tree Adder are Trapped in **Correlation Problem**.



Bipolar XNOR Multiplier



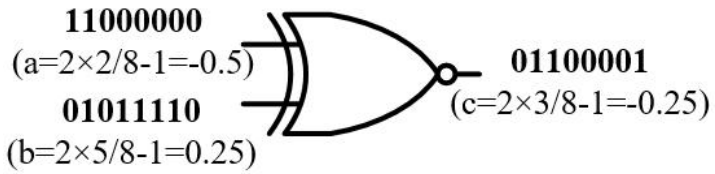
Bipolar OR Adder[2]

[2] P. Li, D. J. Lilja, W. Qian, K. Bazargan, et al., "Computation on stochastic bit streams digital image processing case studies," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 22, no. 3, pp. 449–462, 2013.

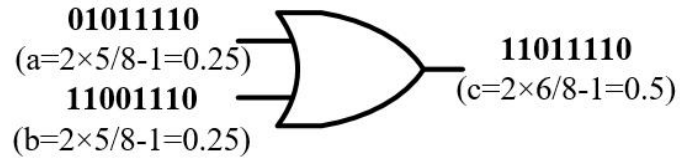
Motivation2

Arithmetic circuits affect the accuracy of computation

- Basic XNOR Multiplier and OR-tree Adder are Trapped in **Correlation Problem**.

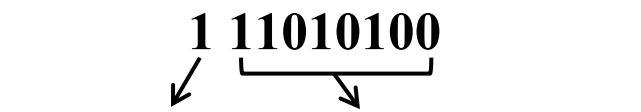


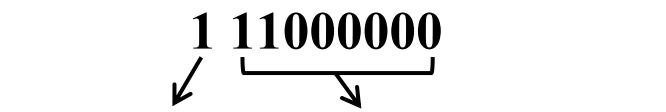
Bipolar XNOR Multiplier



Bipolar OR Adder[2]

'Sign-magnitude'[3] format is proposed and utilizes AND as the multiplier.





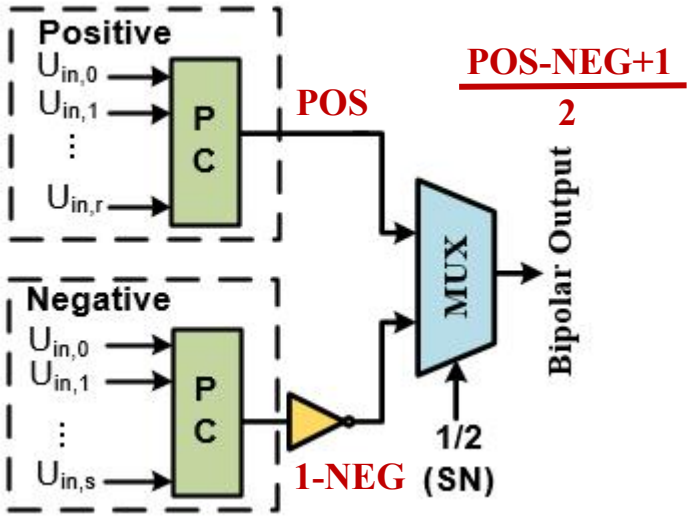
[2] P. Li, D. J. Lilja, W. Qian, K. Bazargan, et al., "Computation on stochastic bit streams digital image processing case studies," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 22, no. 3, pp. 449–462, 2013.

[3] A. Zhakatayev, S. Lee, H. Sim, and J. Lee, "Sign-magnitude sc: getting 10x accuracy for free in stochastic computing for deep neural networks," in 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). IEEE, 2018, pp. 1–6.

Motivation2

Arithmetic circuits affect the accuracy of computation

- Basic XNOR Multiplier and OR-tree Adder are Trapped in **Correlation Problem**.
- Separated Adder Suffers from Severe **Overflow Problem**.



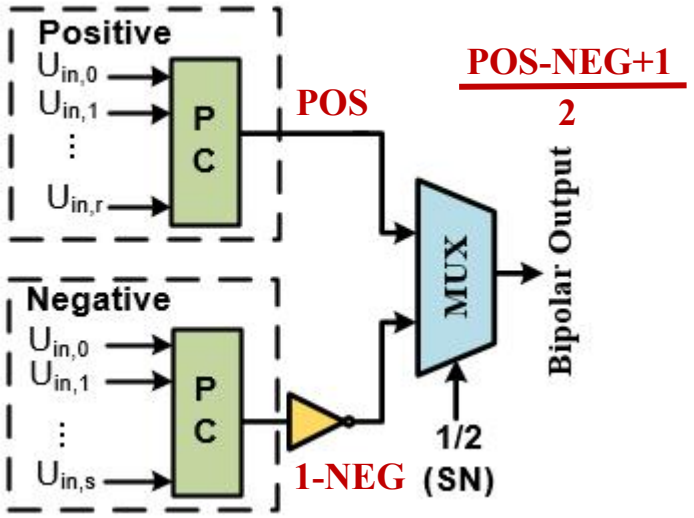
Seperated Adder[4]

[4] B. Li, Y. Qin, B. Yuan, and D. J. Lilja, "Neural network classifiers using stochastic computing with a hardware-oriented approximate activation function," in 2017 IEEE International Conference on Computer Design (ICCD), 2017.

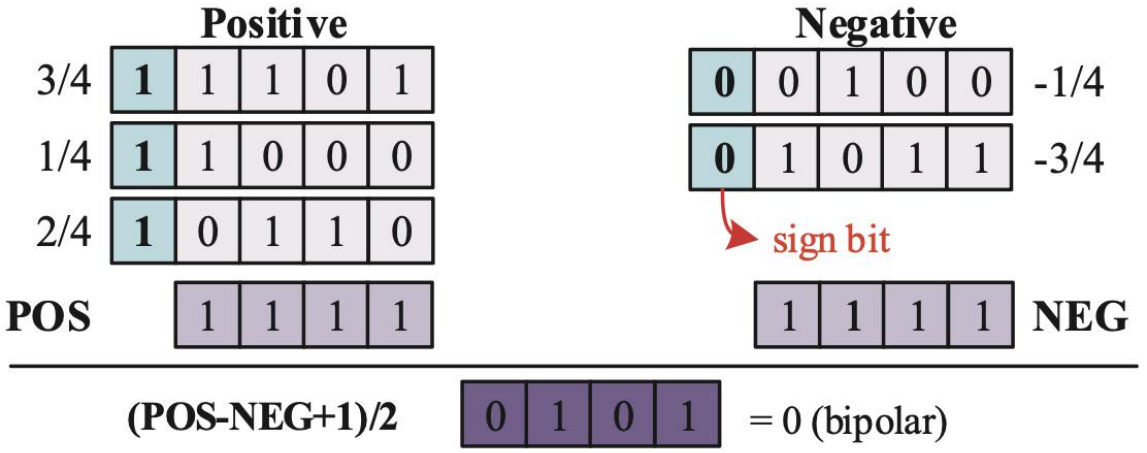
Motivation2

Arithmetic circuits affect the accuracy of computation

- Basic XNOR Multiplier and OR-tree Adder are Trapped in **Correlation Problem**.
- Separated Adder Suffers from Severe **Overflow Problem**.



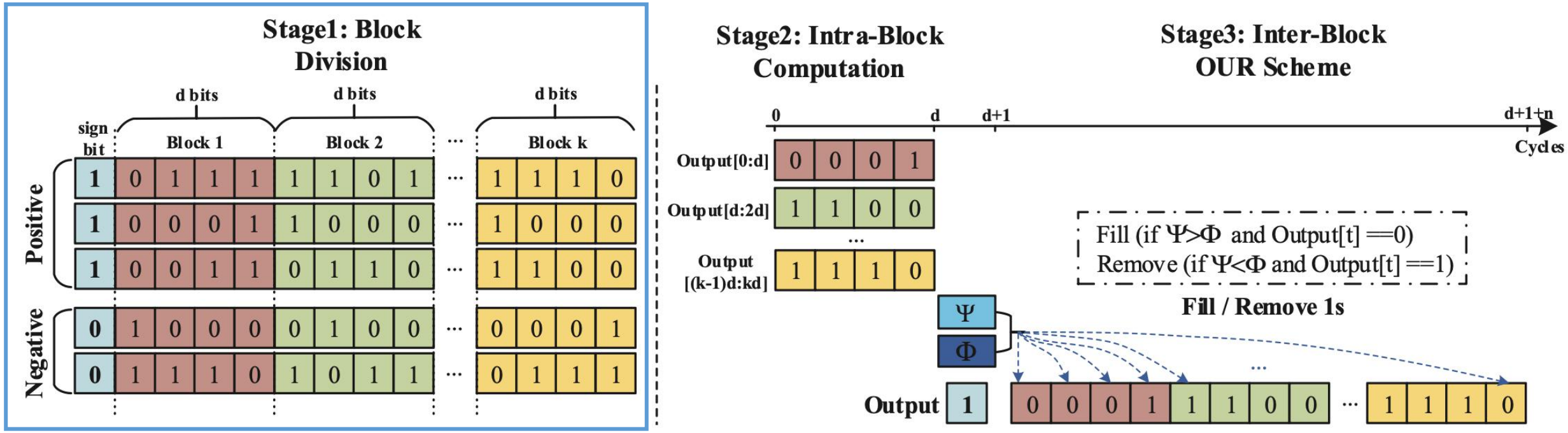
Seperated Adder[4]



An example of seperated adder for overflow problem

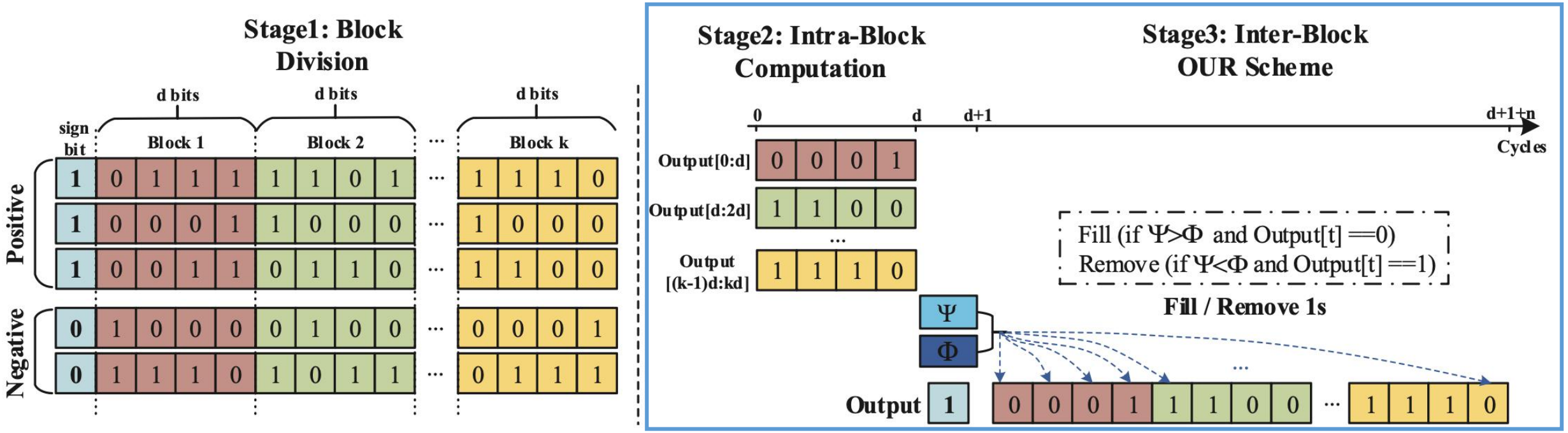
[4] B. Li, Y. Qin, B. Yuan, and D. J. Lilja, "Neural network classifiers using stochastic computing with a hardware-oriented approximate activation function," in 2017 IEEE International Conference on Computer Design (ICCD), 2017.

BSC framework



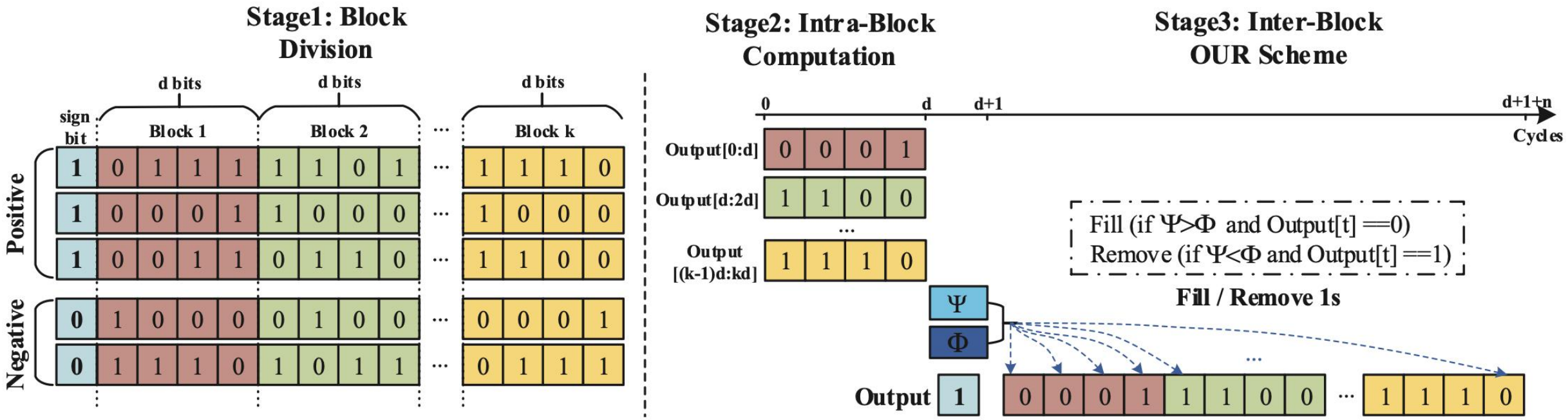
- Inputs are divided into **blocks** and execute them in parallel to reduce latency.

BSC framework



- Inputs are divided into **blocks** and execute them in parallel to reduce latency.
- **Optimized arithmetic circuit** inside blocks and **output revision scheme** between blocks are proposed to improve inference accuracy.

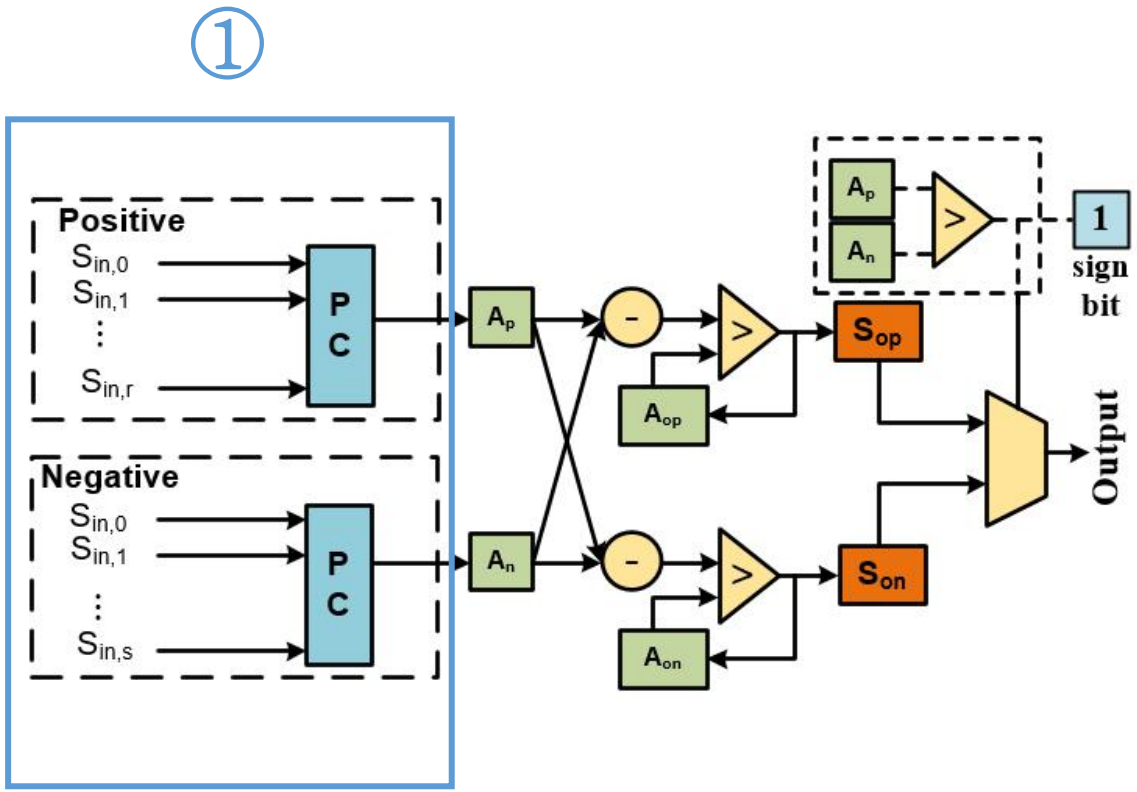
BSC framework



- Inputs are divided into **blocks** and execute them in parallel to reduce latency.
- **Optimized arithmetic circuit** inside blocks and **output revision scheme** between blocks are proposed to improve inference accuracy.
- **A global optimization approach** is devised to determine the number of blocks for better accuracy, latency, and power trade-off

Intra-Block: Accumulator-based adder

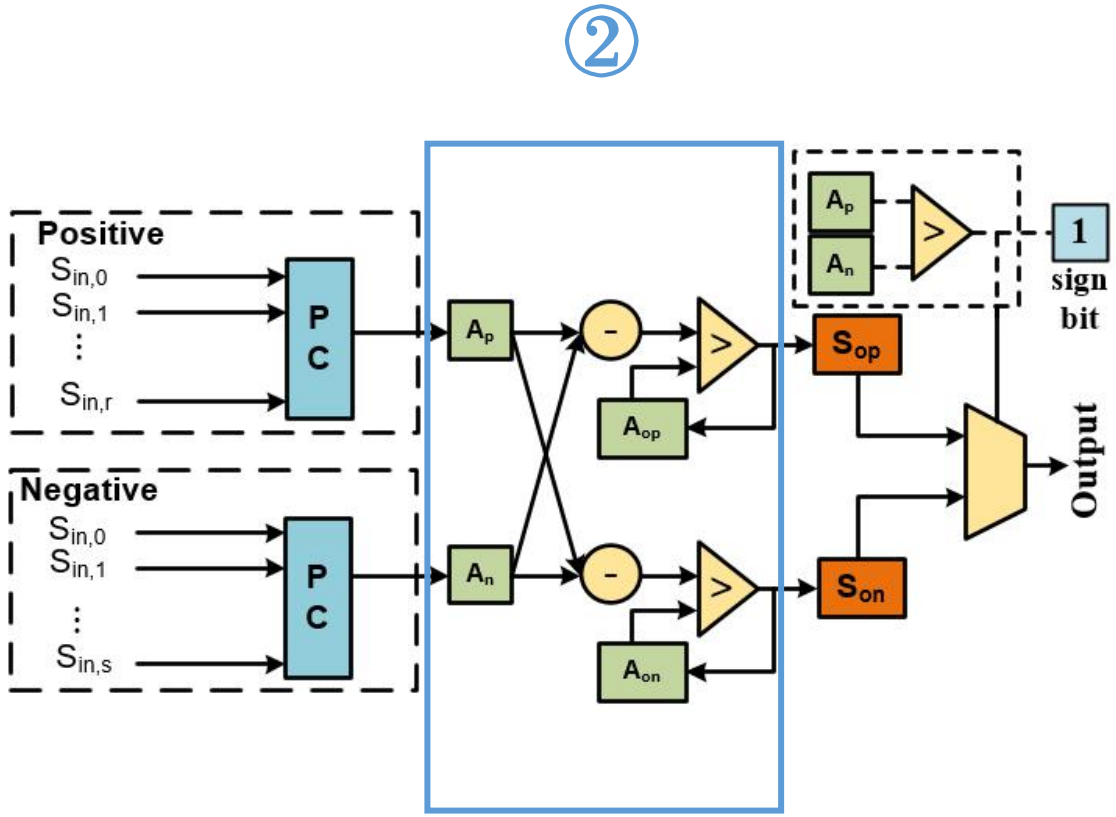
Accumulator-based adder can alleviate correlation problem in XNOR+OR circuit and overflow problems in seperated adder.



① Input in parallel

Intra-Block: Accumulator-based adder

Accumulator-based adder can alleviate correlation problem in XNOR+OR circuit and overflow problems in seperated adder.

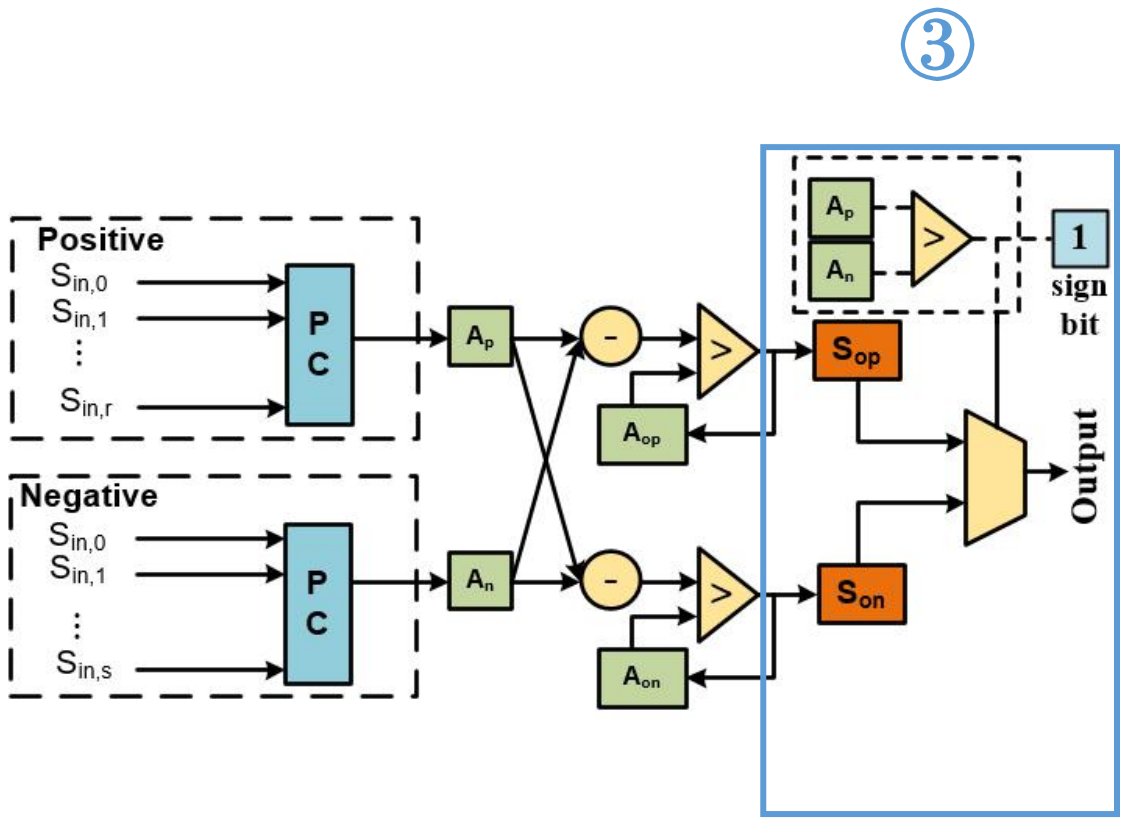


① Input in parallel

② Accumulate '1's

Intra-Block: Accumulator-based adder

Accumulator-based adder can alleviate correlation problem in XNOR+OR circuit and overflow problems in seperated adder.



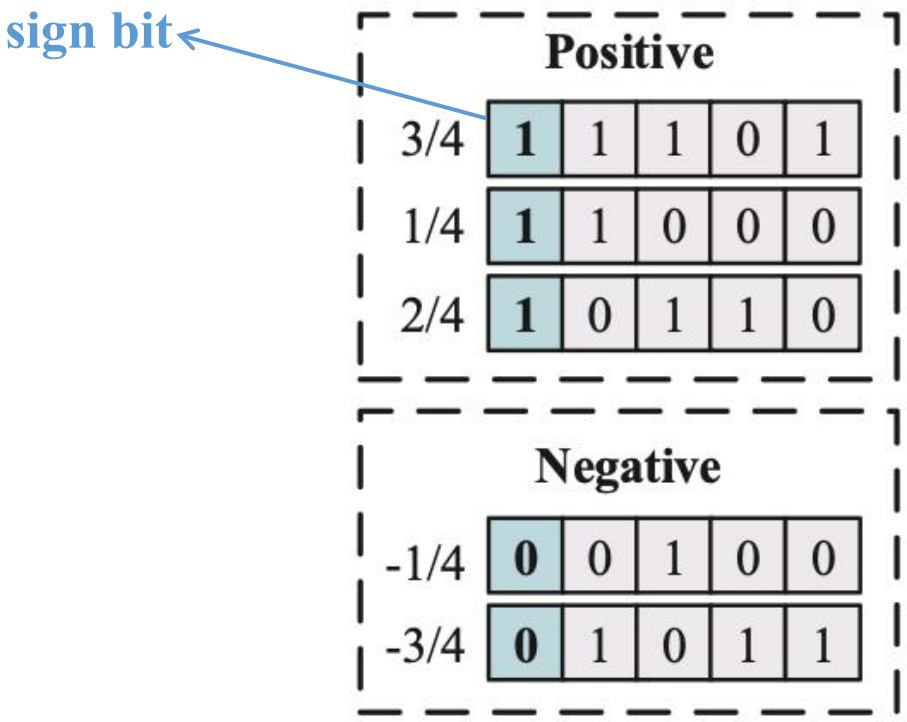
① Input in parallel

② Accumulate '1's

③ Determine the output

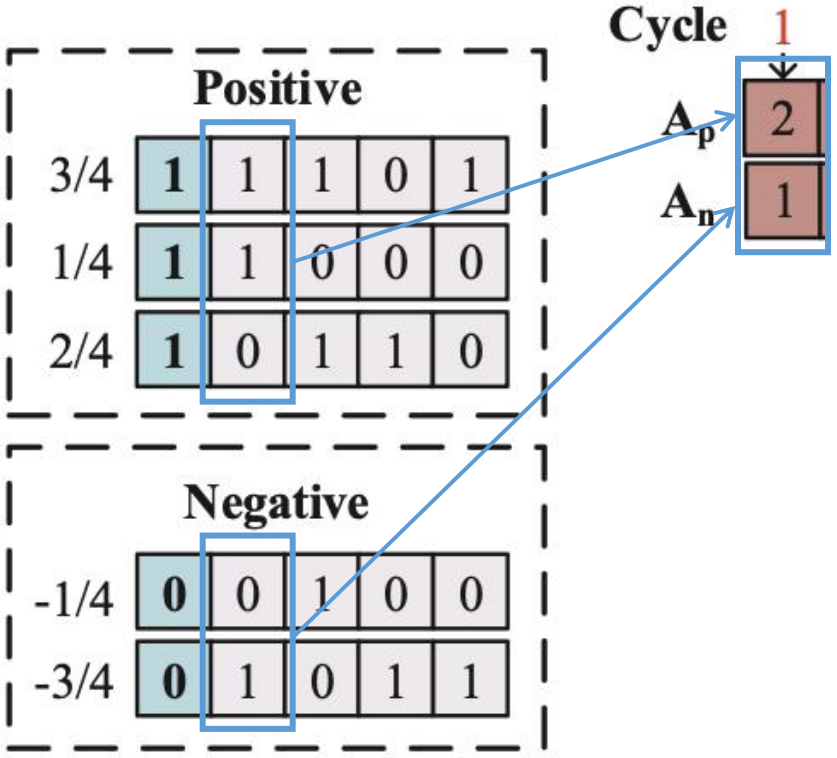
Intra-Block: Accumulator-based adder

① Input



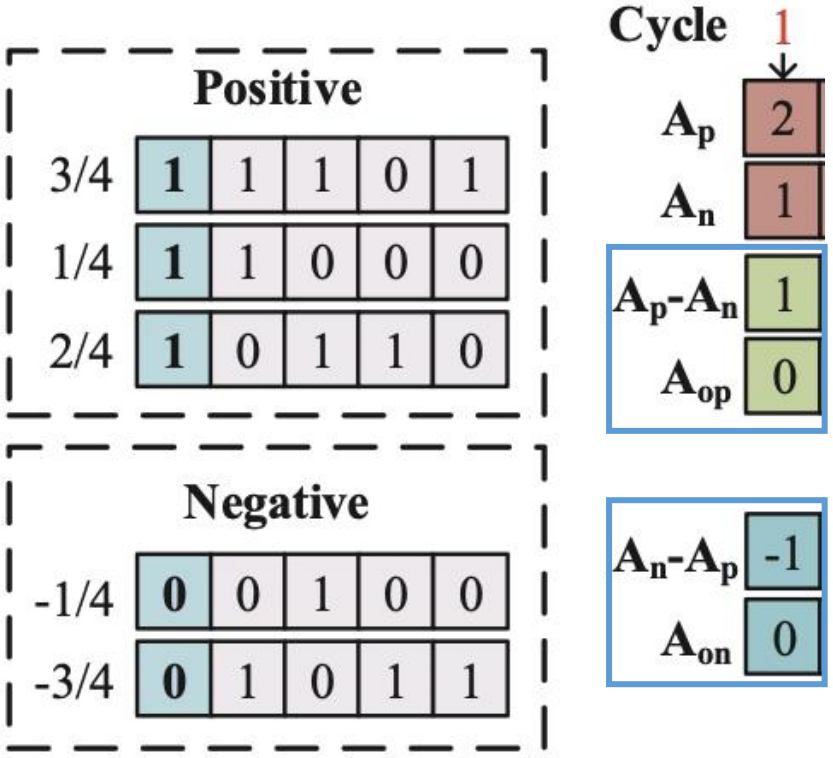
Intra-Block: Accumulator-based adder

② Accumulate '1's



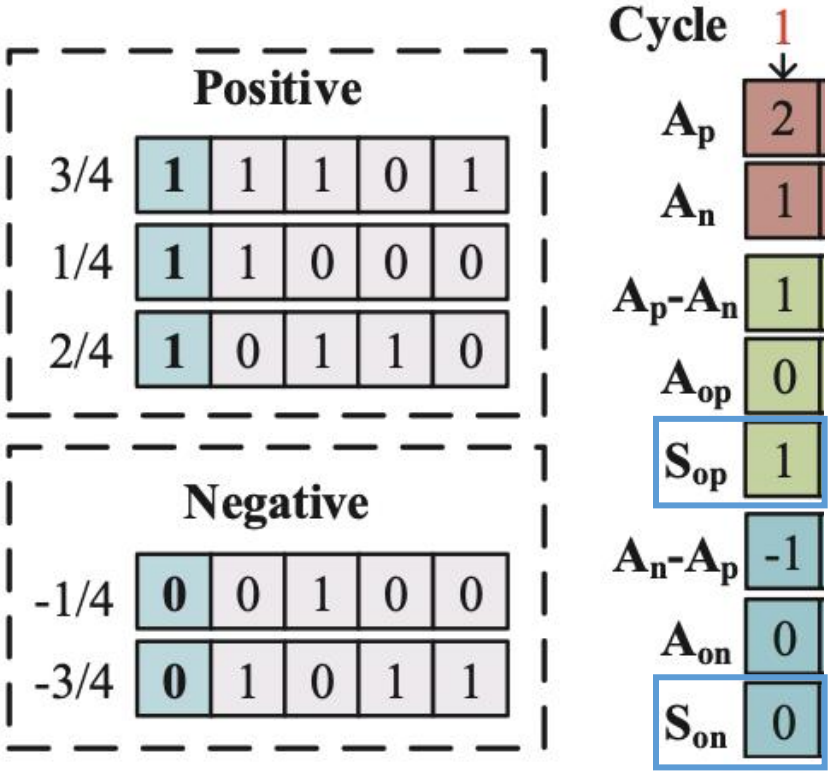
Intra-Block: Accumulator-based adder

② Accumulate '1's



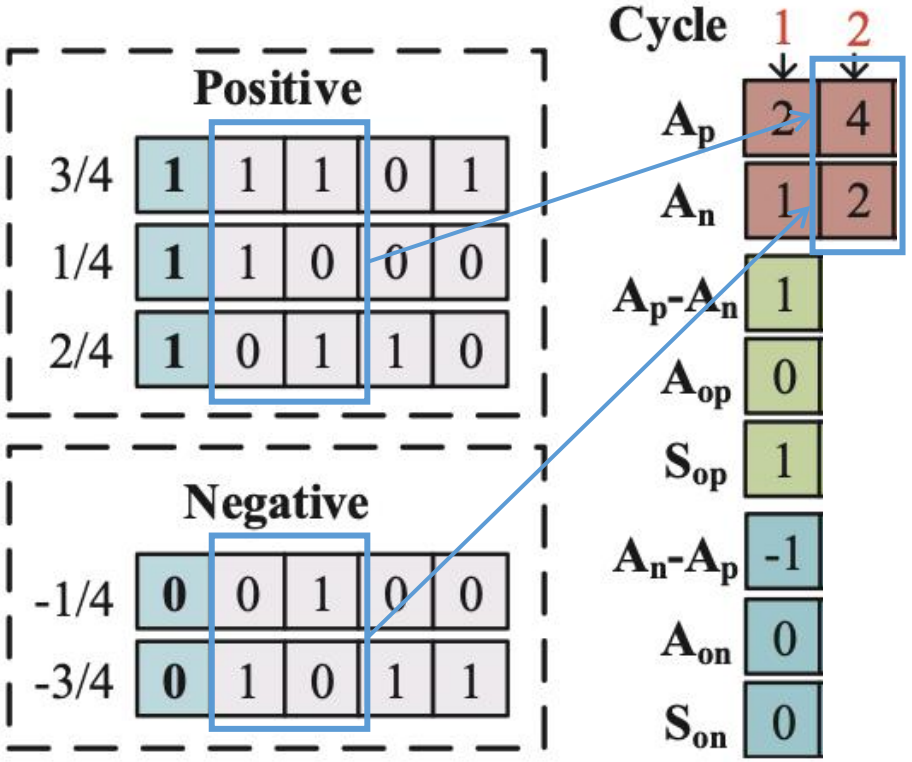
Intra-Block: Accumulator-based adder

② Accumulate '1's



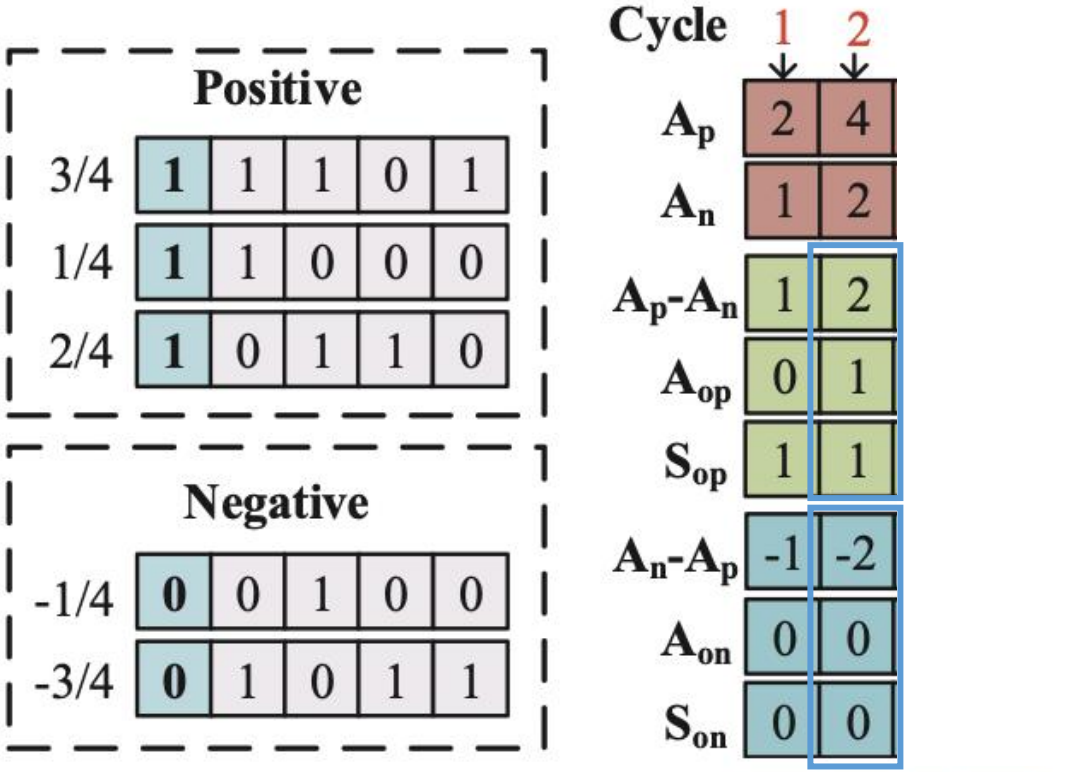
Intra-Block: Accumulator-based adder

② Accumulate '1's



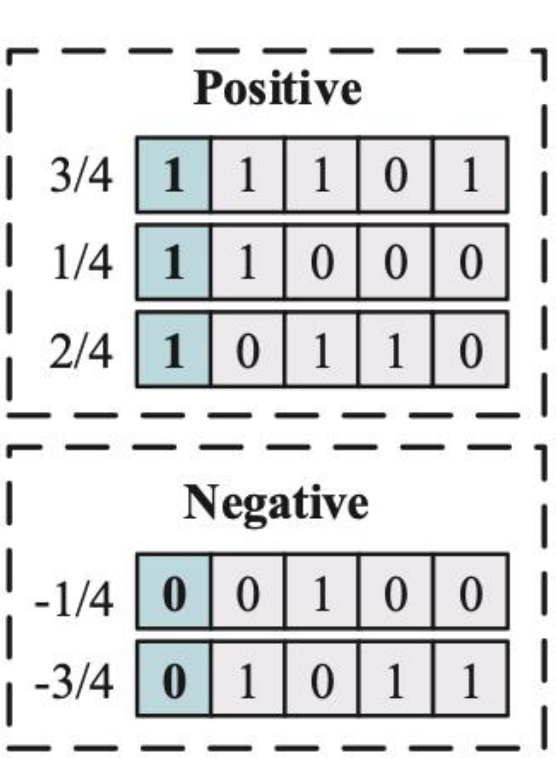
Intra-Block: Accumulator-based adder

② Accumulate '1's



Intra-Block: Accumulator-based adder

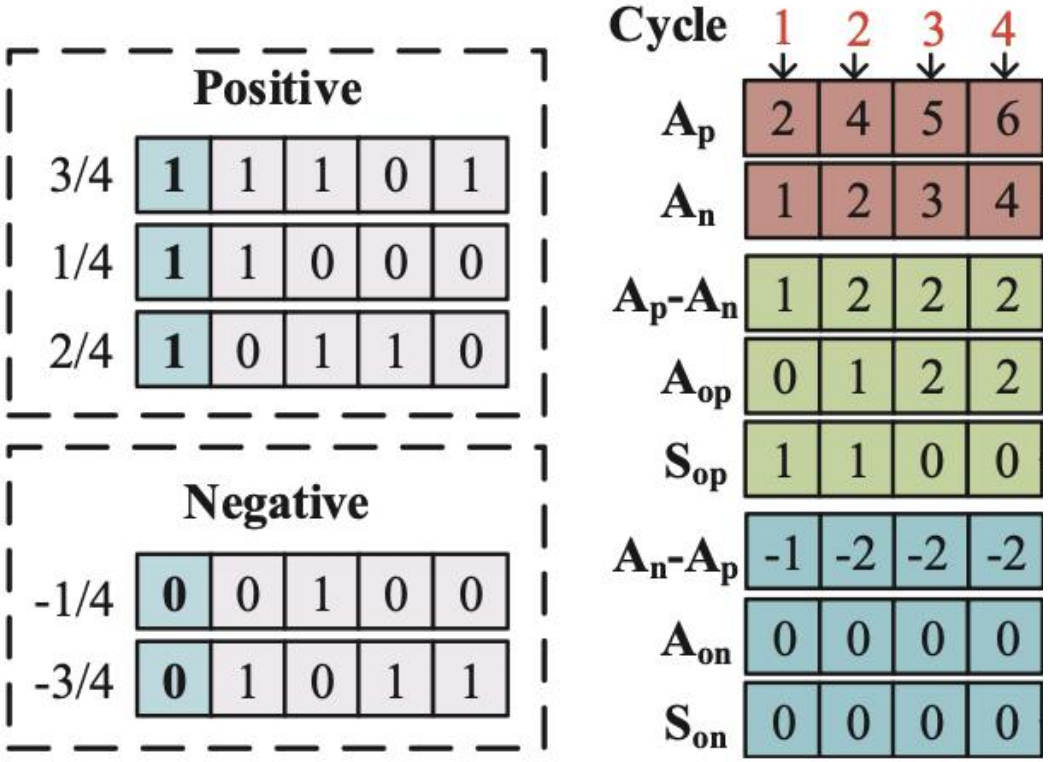
② Accumulate '1's



Cycle	1	2	3
A_p	2	4	5
A_n	1	2	3
A_p-A_n	1	2	2
A_{op}	0	1	2
S_{op}	1	1	0
A_n-A_p	-1	-2	-2
A_{on}	0	0	0
S_{on}	0	0	0

Intra-Block: Accumulator-based adder

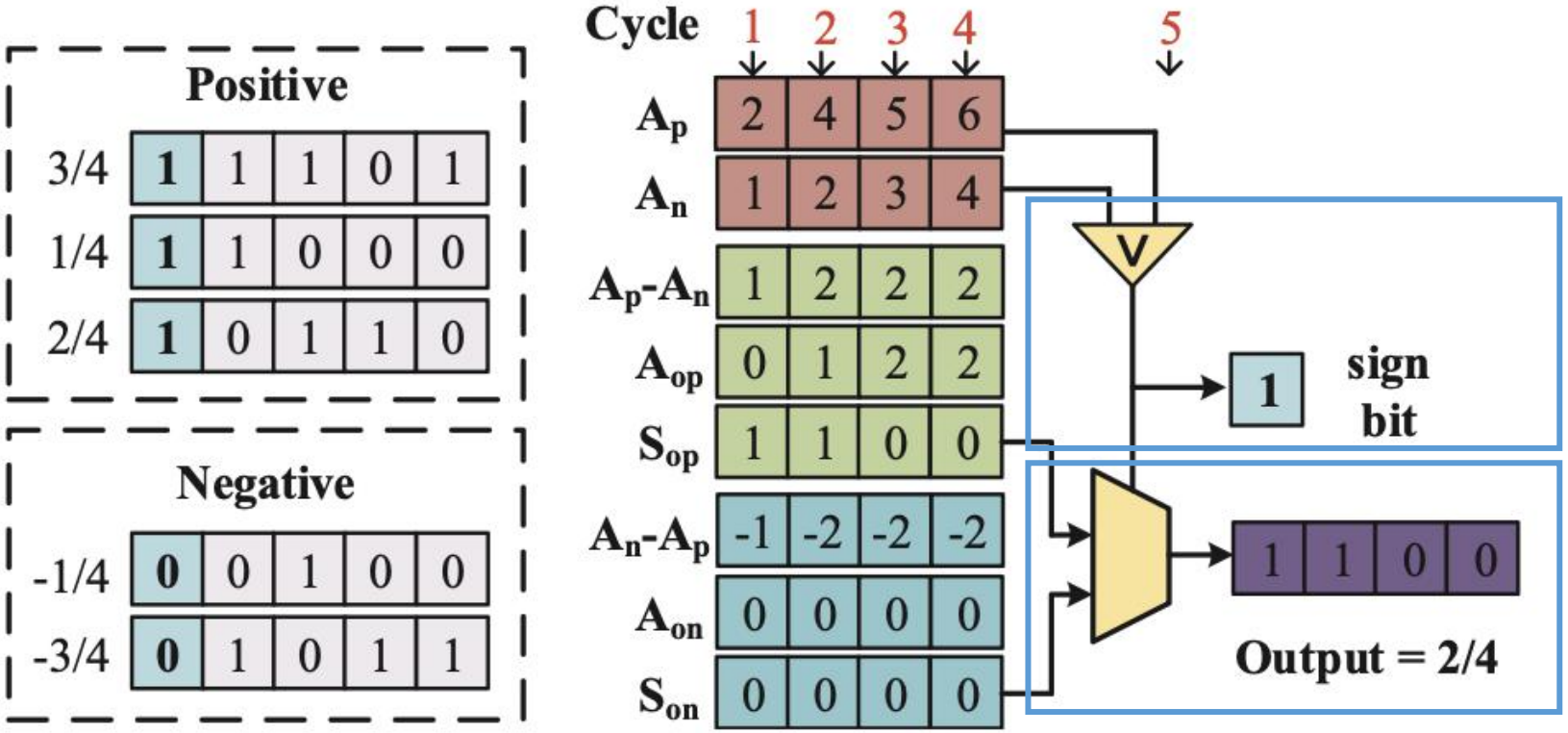
② Accumulate '1's



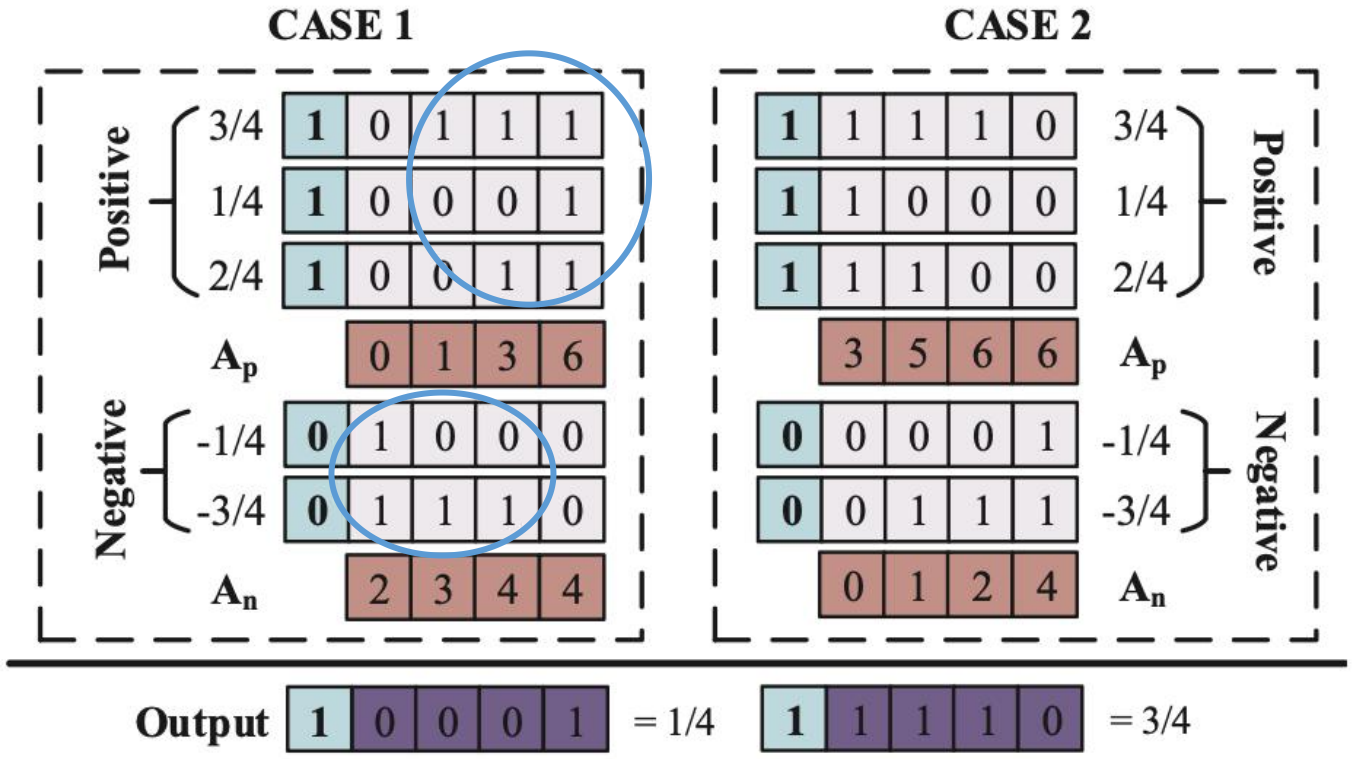
Cycle	1	2	3	4
A_p	2	4	5	6
A_n	1	2	3	4
A_p-A_n	1	2	2	2
A_{op}	0	1	2	2
S_{op}	1	1	0	0
A_n-A_p	-1	-2	-2	-2
A_{on}	0	0	0	0
S_{on}	0	0	0	0

Intra-Block: Accumulator-based adder

③ Output the correct result

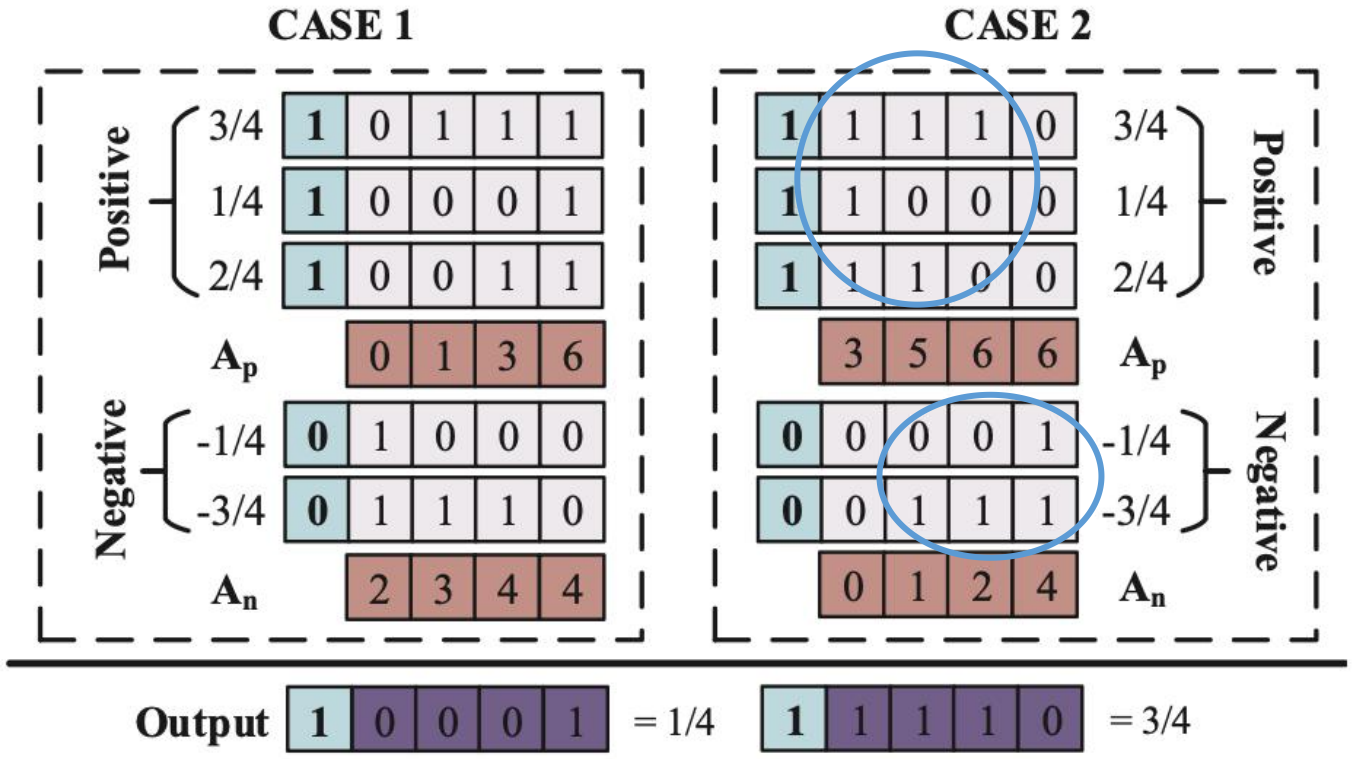


Intra-Block: Accumulator-based adder



Lost '1's in result

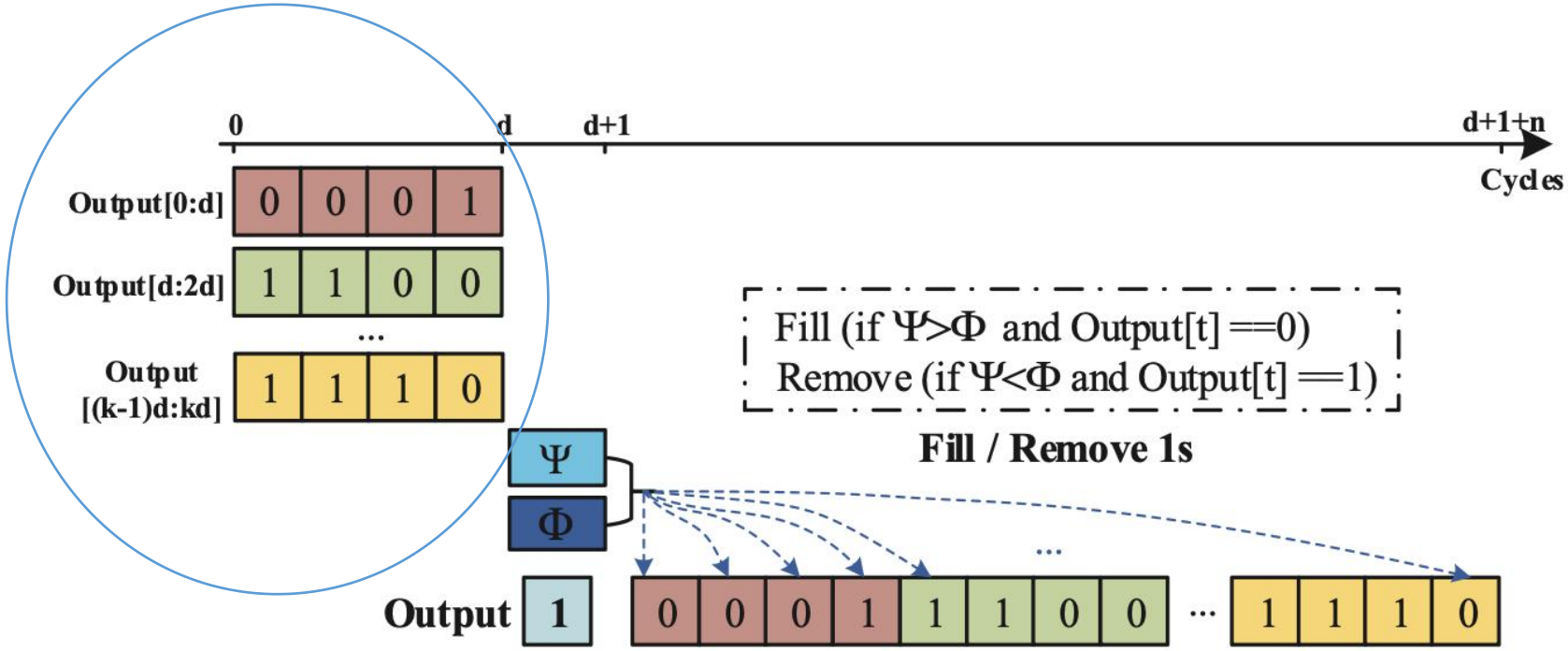
Intra-Block: Accumulator-based adder



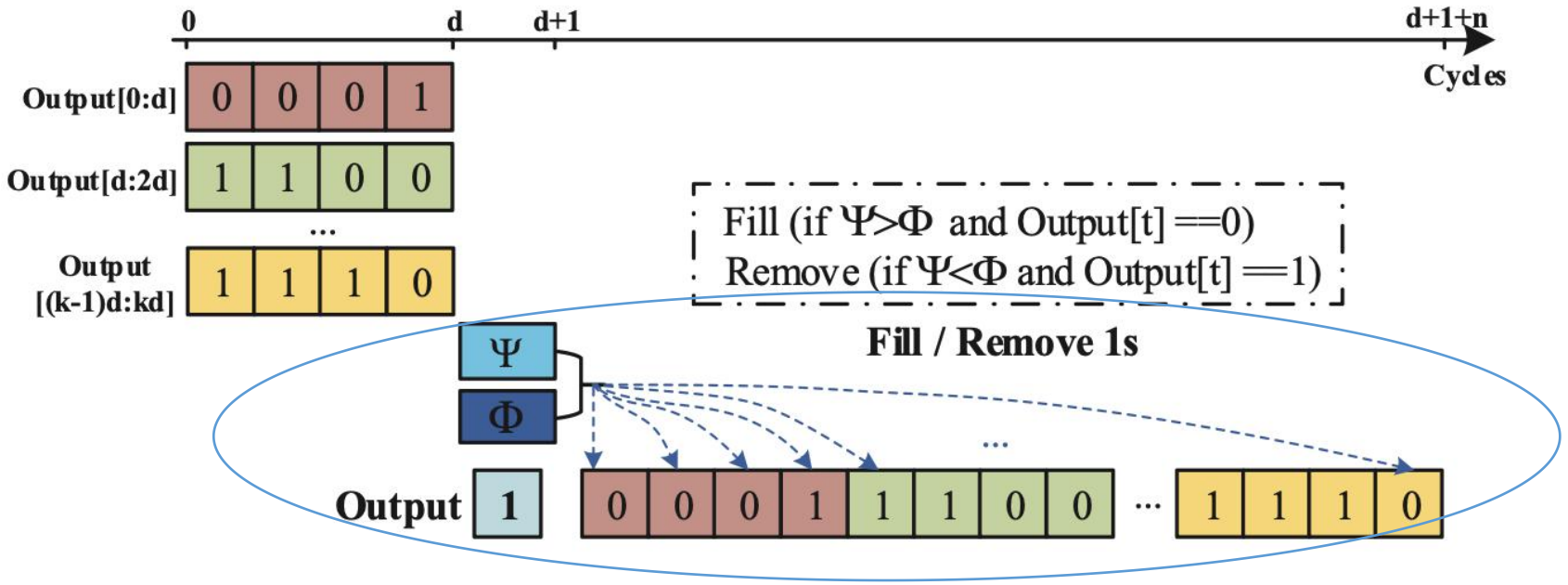
Redundant '1's in result

Inter-Blocks: Output Revision (OUR) Scheme

Output for each block using accumulator-based adder



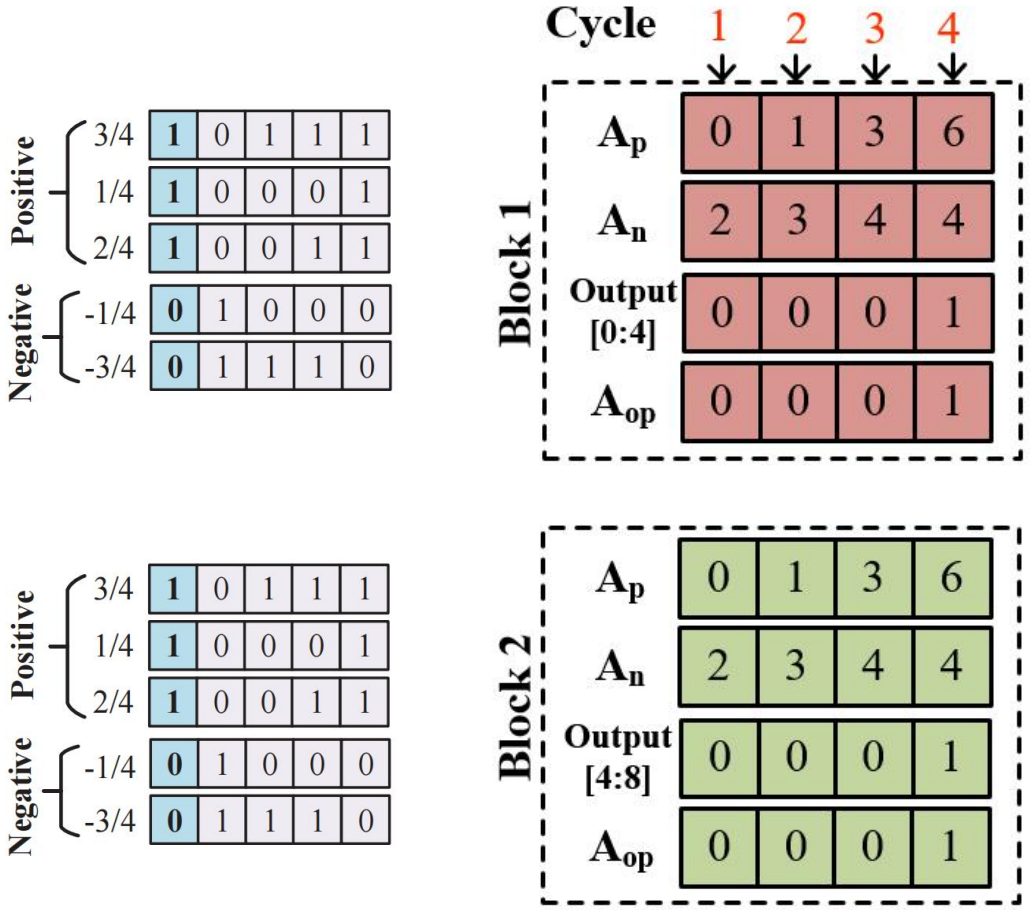
Inter-Blocks: Output Revision (OUR) Scheme



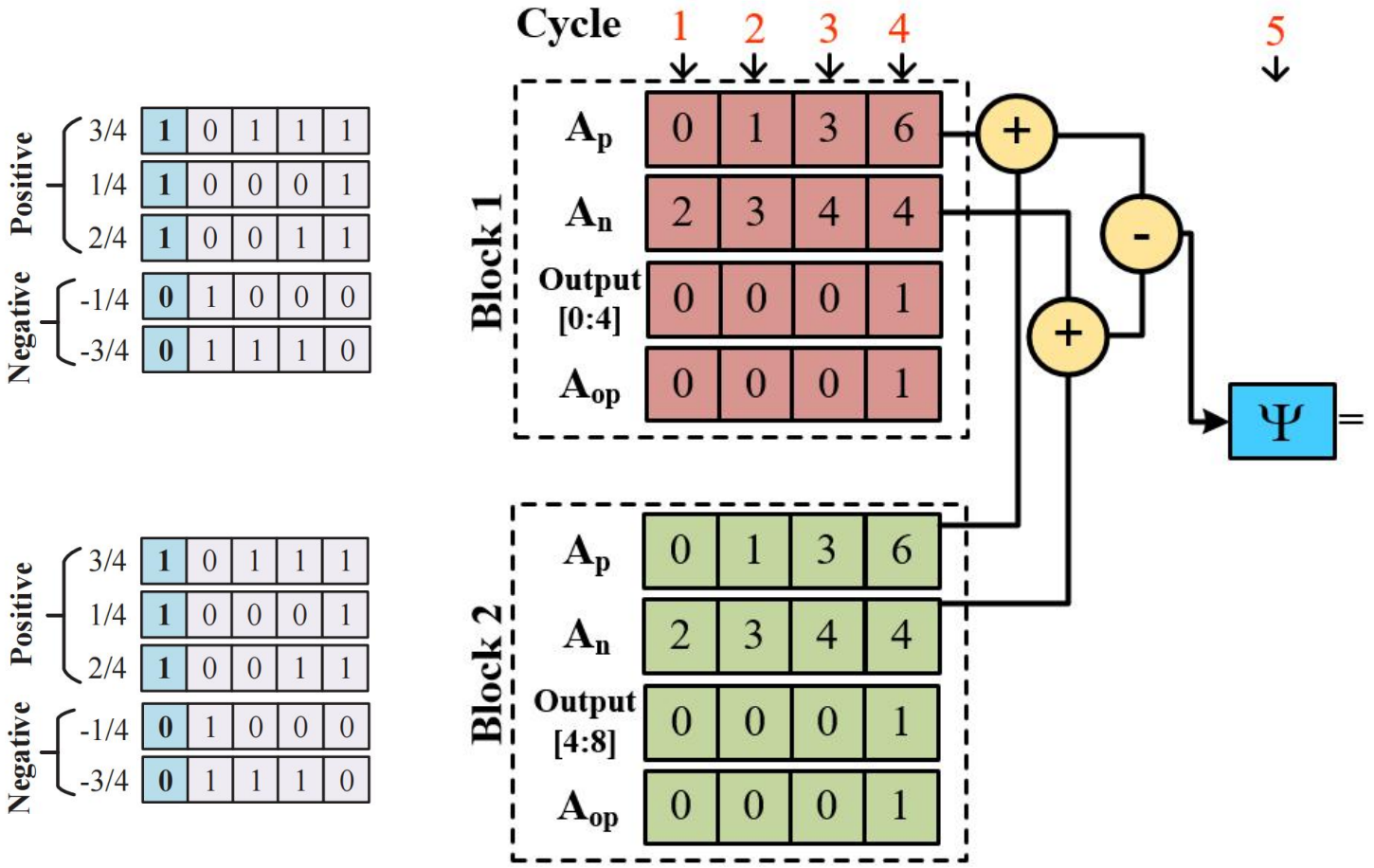
Compare the accurate number of 1s and the real number of 1s in outputs

Output revision scheme induces inaccuracy among blocks.

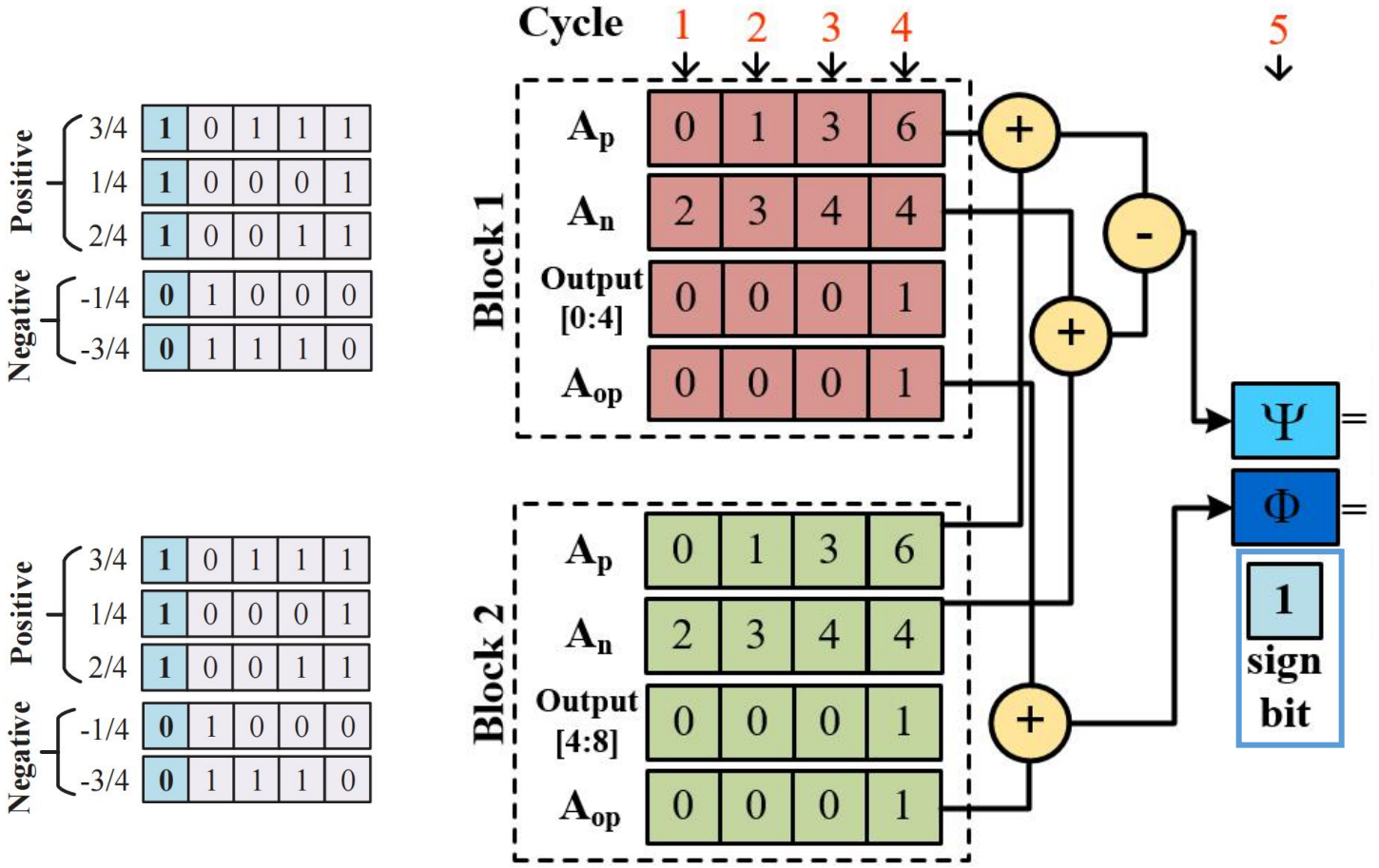
Inter-Blocks: Output Revision (OUR) Scheme



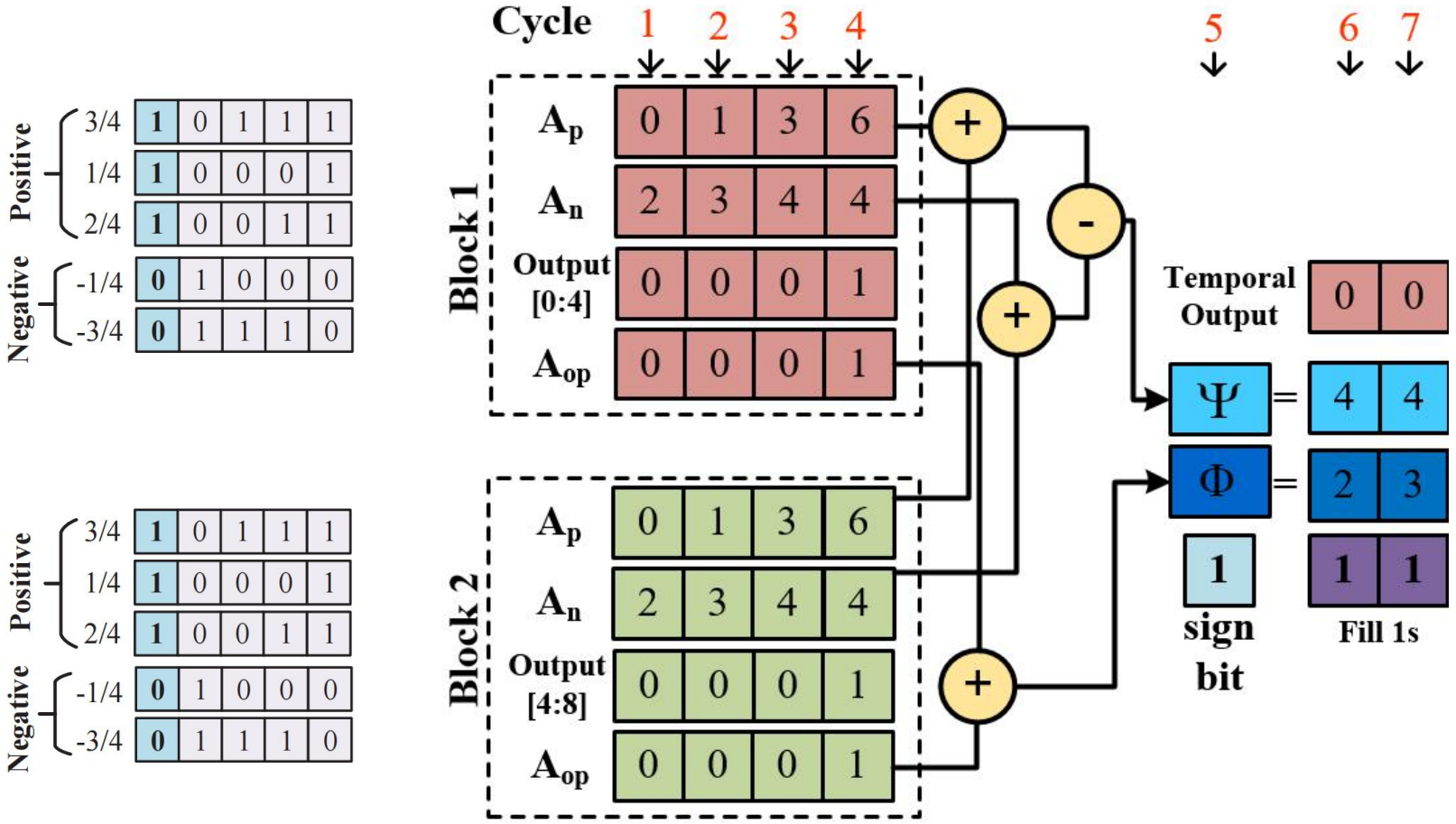
Inter-Blocks: Output Revision (OUR) Scheme



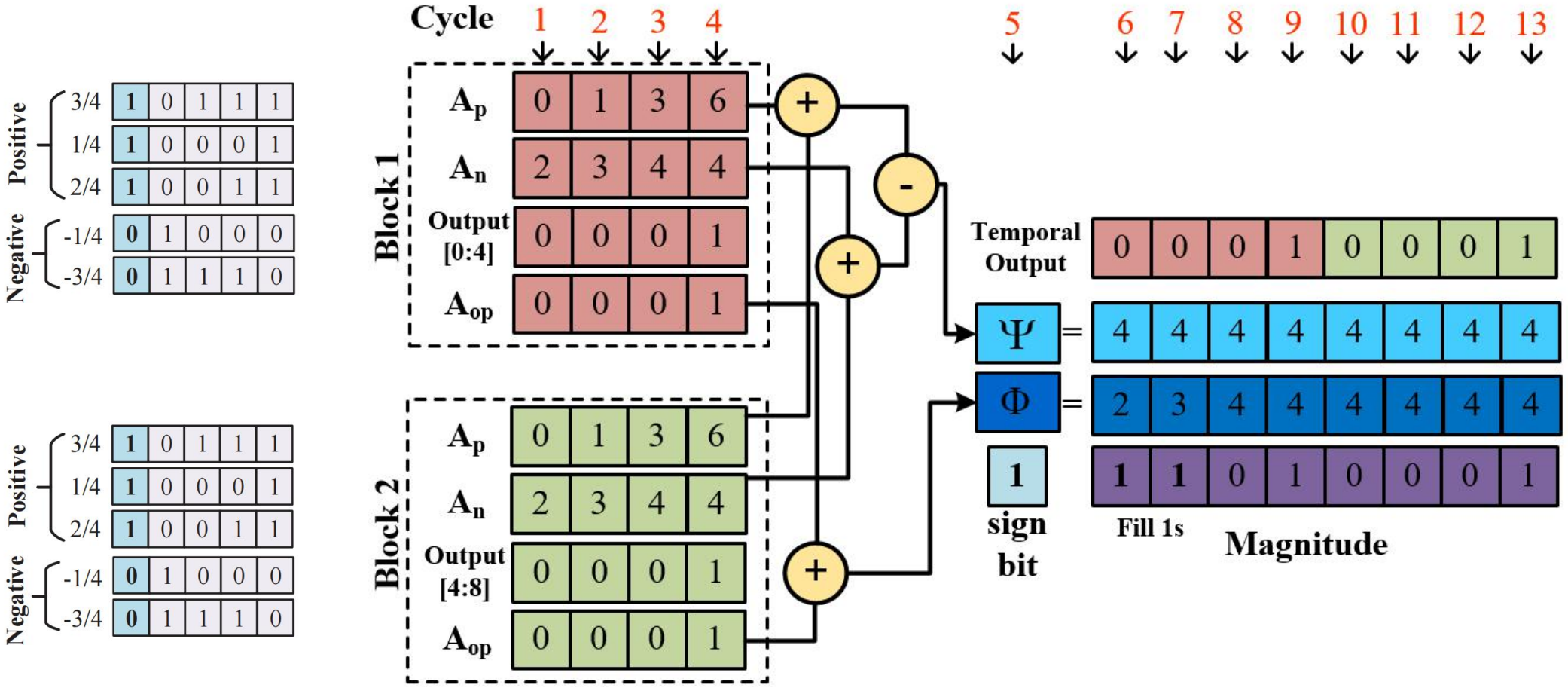
Inter-Blocks: Output Revision (OUR) Scheme



Inter-Blocks: Output Revision (OUR) Scheme



Inter-Blocks: Output Revision (OUR) Scheme



Heuristic Strategy for Block Division

- **On the accuracy side, the number of blocks impacts the accuracy of proposed intra-block adder.**

$$Probability = \begin{cases} P(p \geq q) & p \geq q \\ 1 - P(p \geq q) & p < q \end{cases}$$

Heuristic Strategy for Block Division

- **On the accuracy side, the number of blocks impacts the accuracy of proposed intra-block adder.**

$$Probability = \begin{cases} P(p \geq q) & p \geq q \\ 1 - P(p \geq q) & p < q \end{cases}$$

$$P(p \geq q) = \sum_{i=0}^{d} [C_d^i p^i (1-p)^{d-i} \sum_{j=0}^{i} C_d^j q^j (1-q)^{d-j}]$$

Heuristic Strategy for Block Division

- On the accuracy side, the number of blocks impacts the accuracy of proposed intra-block adder.

$$Probability = \begin{cases} P(p \geq q) & p \geq q \\ 1 - P(p \geq q) & p < q \end{cases}$$

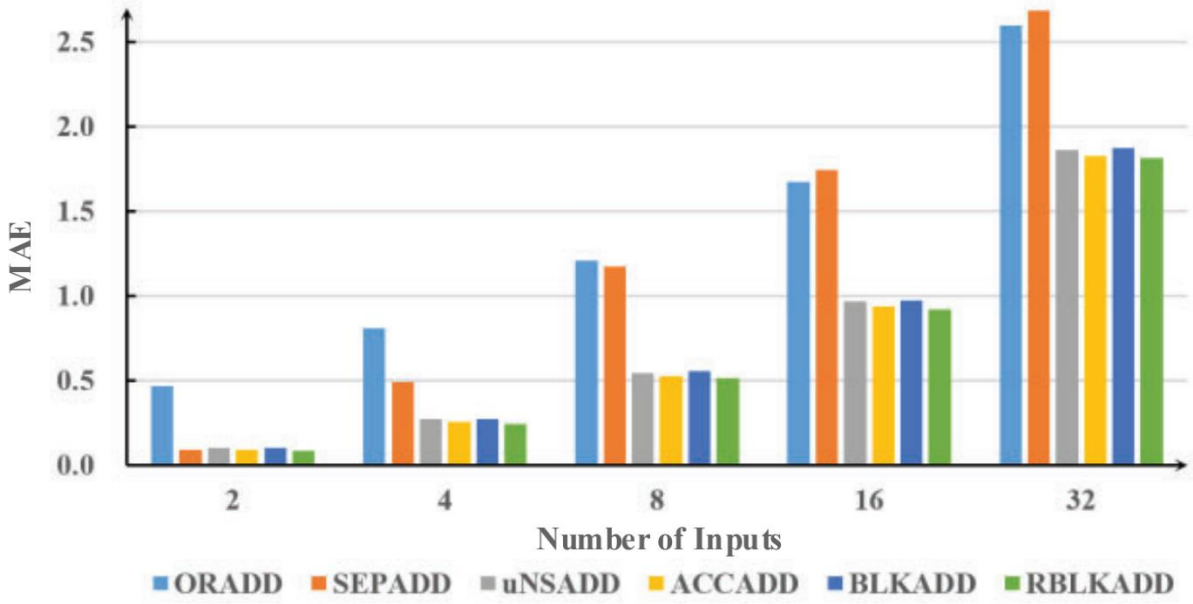
$$P(p \geq q) = \sum_{i=0}^{d} [C_d^i p^i (1-p)^{d-i} \sum_{j=0}^{i} C_d^j q^j (1-q)^{d-j}]$$

- The hardware power consumption become higher because of the parallel execution. We set the hardware power consumption of BSC must be lower than floating point.

Experiments

MAE Evaluation of Adders

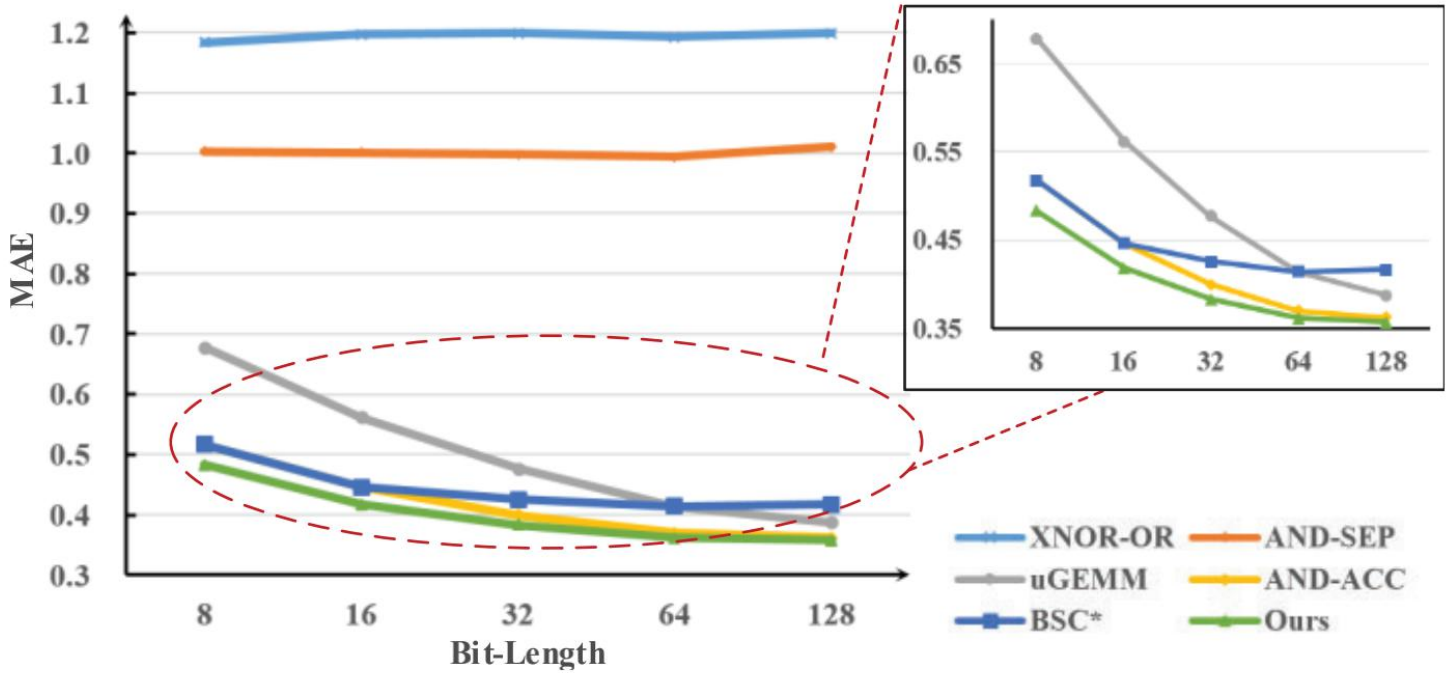
- With the number of inputs increasing, mean absolute error (MAE) shows an upward trend among adders because of the overflow problem.
- Our proposed RBLKADD is the best because of intra-block ACCADD and inter-block OUR scheme.



Experiments

MAE Evaluation of General Matrix Multiplication (GEMM)

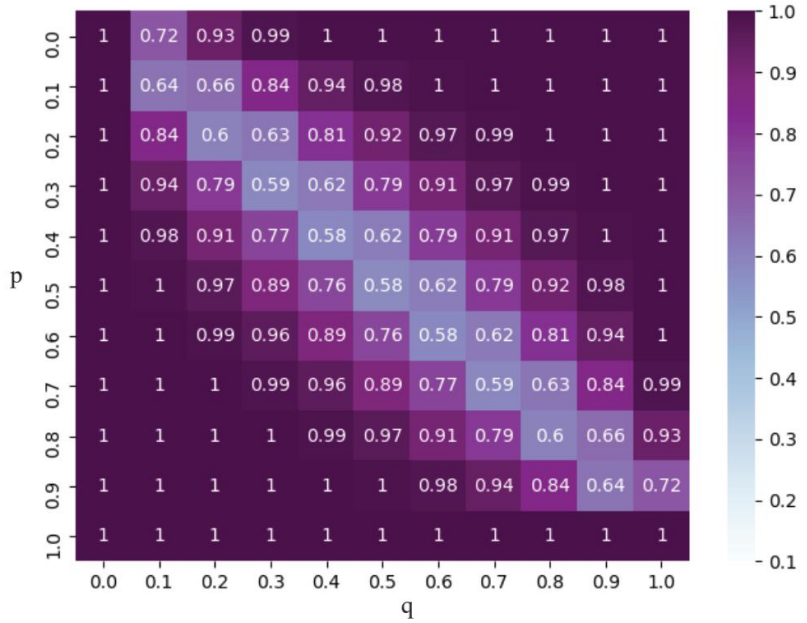
- XNOR-OR and AND-SEP circuits suffer from severe correlation and overflow problems.
- Our proposed BSC obtains the lowest MAE.



Experiments

The Number of Block Exploration

- Through exhaustive search, the average correct probability becomes greater than θ=90% when the number of block is 12.
- Guarantee the addition accuracy inside every block, reduce the cost of OUR scheme and keep output relatively uniform.



Experiments

The Number of Block Exploration

- Compared with BSC*, our BSC can improve the accuracy without any additional cycles.
- BSC achieves over 6 × power saving than FP circuit.

Design		Cycles	Stalls	MAE	Power (W)
1/64	Ours	130	64	0.314	2.17
	BSC*			0.325	2.17
2/32	Ours	98	32	0.362	3.35
	BSC*			0.390	3.22
4/16	Ours	82	16	0.305	5.74
	BSC*			0.361	5.35
8/8	Ours	74	8	0.346	10.86
	BSC*			0.442	9.96
16/4	Ours	70	4	0.368	21.02
	BSC*			0.519	19.09
32/2	Ours	68	2	0.332	41.31
	BSC*			0.533	37.31
64/1	Ours	67	1	0.352	81.02
	BSC*			0.562	72.90

Experiments

Evaluation of Latency

- BSC produces a little higher cycles and pipeline stalls than first 3 circuits.
- BSC achieves 3.6 × , 3.1 × and 1.2 × accuracy improvement than them.
- BSC saves lots of cycles because of parallel computing than AND-ACC.

Design	MAE	Cycles	Stalls
XNOR-OR	1.097	65	0
AND-SEP	0.937	65	0
uGEMM	0.362	65	0
AND-ACC	0.315	130	64
BSC*	0.361	82	16
Ours	0.305	82	16

Experiments

MLP Implementation

- 3-layers MLP with 32 and 64 hidden neurons.
- Methods are evaluated on the MNIST dataset.
- There is only 0.7% accuracy gap compared with FP.

Design	FP	XNOR-OR	AND-SEP	uGEMM	BSC*	Ours
Accuracy	96.1%	10.0%	21.7%	85.1%	93.4%	95.4%
Acc.loss	-	86.1%	74.4%	11.0%	2.7%	0.7%

Conclusion

- In order to reduce the computing **latency**, BSC divides inputs into **blocks**, then they are executed in parallel.
- Aim at improving the **accuracy** of SC arithmetic circuits, we propose a novel **intra-block accumulator-based adder** and **inter-block output revision scheme**.
- Finally, we propose **a heuristic strategy** to determine the number of block, which takes **accuracy, latency and power consumption** into consideration.
- Results show that our method achieves over **10% higher accuracy** than existing methods, and saves over **6× power consumption**.

BSC: Block-based Stochastic Computing to Enable Accurate and Efficient TinyML

Thank you!
Questions?

If any questions, please contact us!
Edwin Hsing-Mean Sha edwinsha@cs.ecnu.edu.cn
Yuhong Song yhsong@stu.ecnu.edu.cn

THE UNIVERSITY OF NEW MEXICO