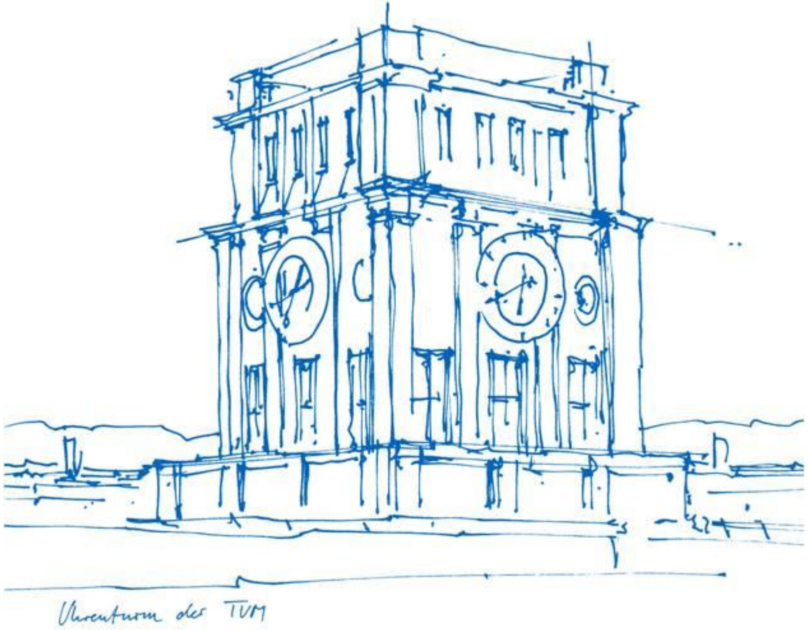
OrthrusPE: Runtime Reconfigurable Processing Elements for Binary Neural Networks

Nael Fasfous

Technical University of Munich

Department of Electrical and Computer Engineering

Chair of Integrated Systems



Outline

- Optimization of Convolutional Neural Networks
- Challenges of Binary Neural Networks
- Motivation for Runtime Reconfigurable BNN Processing Elements
- OrthrusPE: Dual Modes
- Experimentation and Results

Optimization of Convolutional Neural Networks

Structural

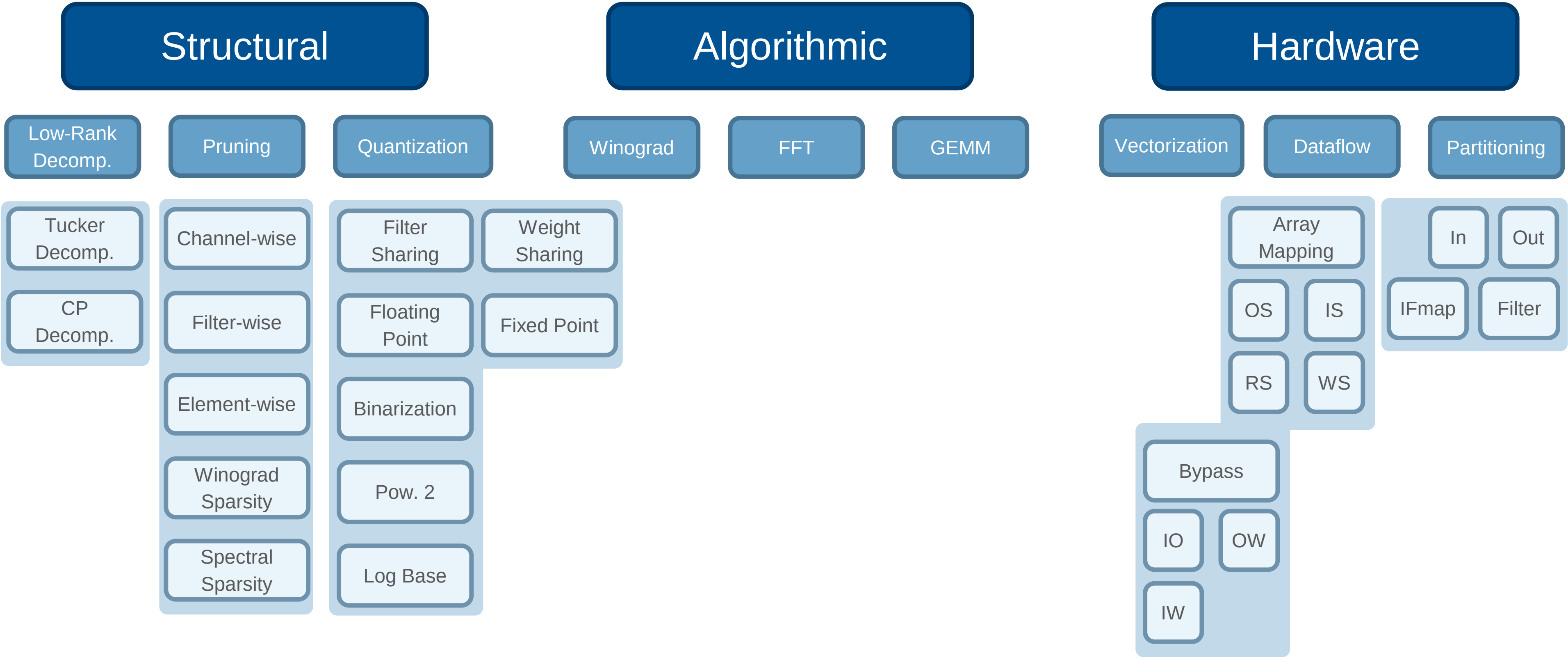
Algorithmic

Hardware

Optimization of Convolutional Neural Networks

- Structural
  - Low-Rank Decomp.
  - Pruning
  - Quantization
- Algorithmic
  - Winograd
  - FFT
  - GEMM
- Hardware
  - Vectorization
  - Dataflow
  - Partitioning

Optimization of Convolutional Neural Networks



Overview of Binary Neural Networks

- Binary Weights
- Binary Activations
- Binary Weights AND Activations

Overview of Binary Neural Networks

- Binary Weights
- Binary Activations
- Binary Weights AND Activations → Replace Multiplications by XNOR ops
  Replace Accumulations by Popcount ops

Overview of Binary Neural Networks

- Binary Weights
- Binary Activations
- Binary Weights AND Activations

→ Replace Multiplications by XNOR ops
Replace Accumulations by Popcount ops

→ Reduce Memory requirements (1/16 of FP-16)

Overview of Binary Neural Networks

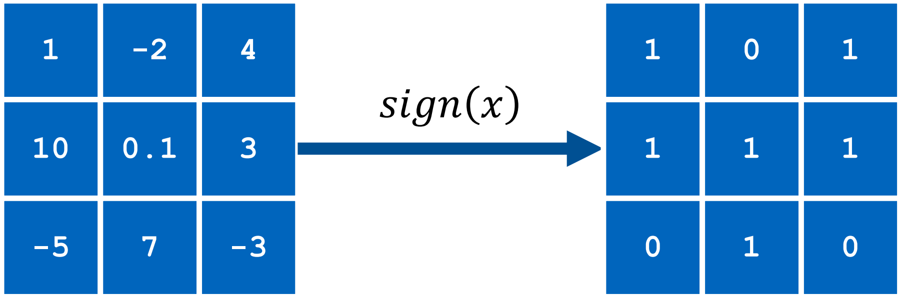
- Binary Weights
- Binary Activations
- Binary Weights AND Activations
  - Replace Multiplications by XNOR ops
    Replace Accumulations by Popcount ops
  - Reduce Memory requirements (1/16 of FP-16)
  - Severe Information Loss

Overview of Binary Neural Networks

- Binary Weights
- Binary Activations
- Binary Weights AND Activations
  - → Replace Multiplications by XNOR ops
    Replace Accumulations by Popcount ops
  - → Reduce Memory requirements (1/16 of FP-16)
  - → Severe Information Loss
  - → MNIST, CIFAR-10, SVHN
    ImageNet

Overview of Binary Neural Networks

Naïve Binarization



Severe information loss:
10 and 0.1 have the same effect on the network.

$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Overview of Binary Neural Networks

Approximation through binary bases

$$\begin{bmatrix} 1 & -2 & 4 \\ 10 & 0.1 & 3 \\ -5 & 7 & -3 \end{bmatrix} \xrightarrow{sign(x-5)} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

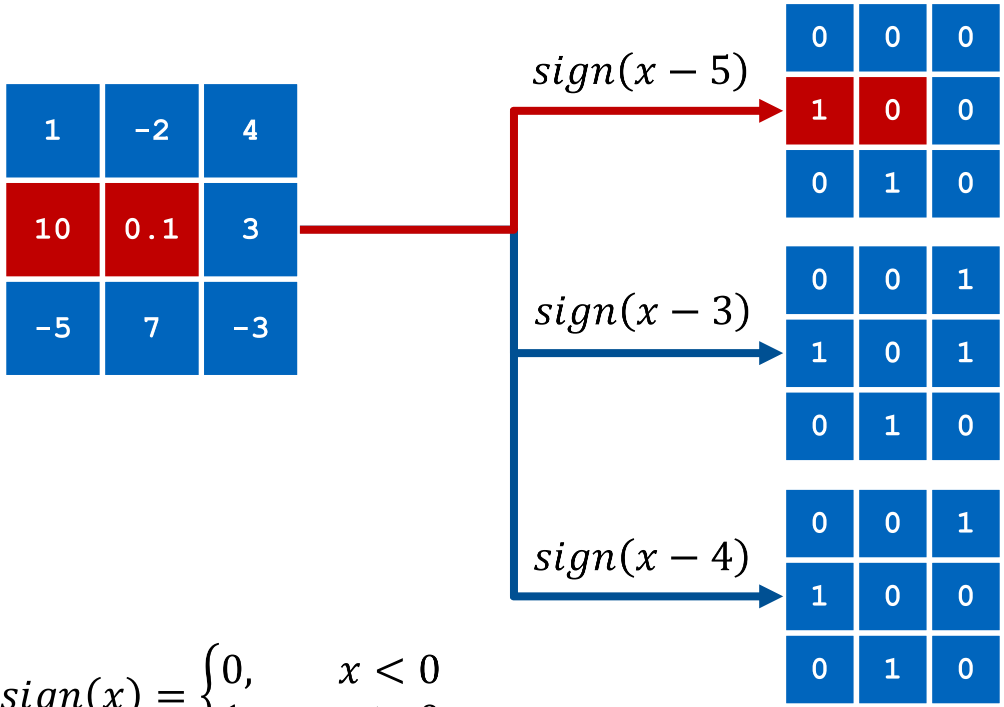
$$\begin{bmatrix} 1 & -2 & 4 \\ 10 & 0.1 & 3 \\ -5 & 7 & -3 \end{bmatrix} \xrightarrow{sign(x-3)} \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -2 & 4 \\ 10 & 0.1 & 3 \\ -5 & 7 & -3 \end{bmatrix} \xrightarrow{sign(x-4)} \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Overview of Binary Neural Networks

Approximation through binary bases



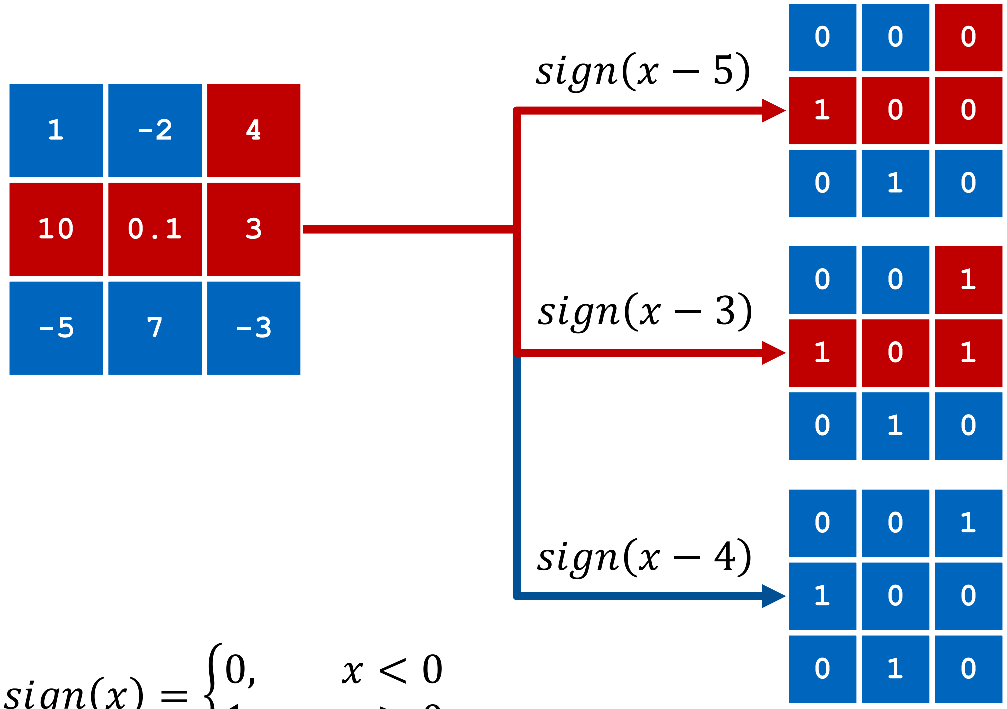
$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Captured Information:

0.1 is less positive than **10**

Overview of Binary Neural Networks

Approximation through binary bases



$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

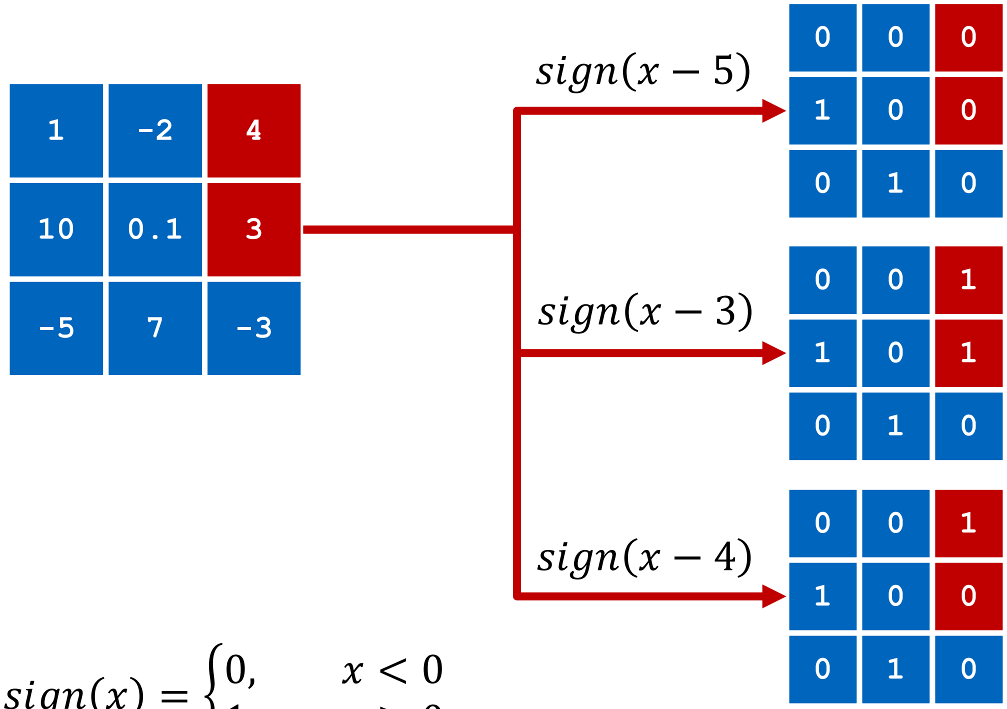
Captured Information:

0.1 is less positive than **10**

4 and 3 lie between **10 and 0.1**

Overview of Binary Neural Networks

Approximation through binary bases



$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Captured Information:

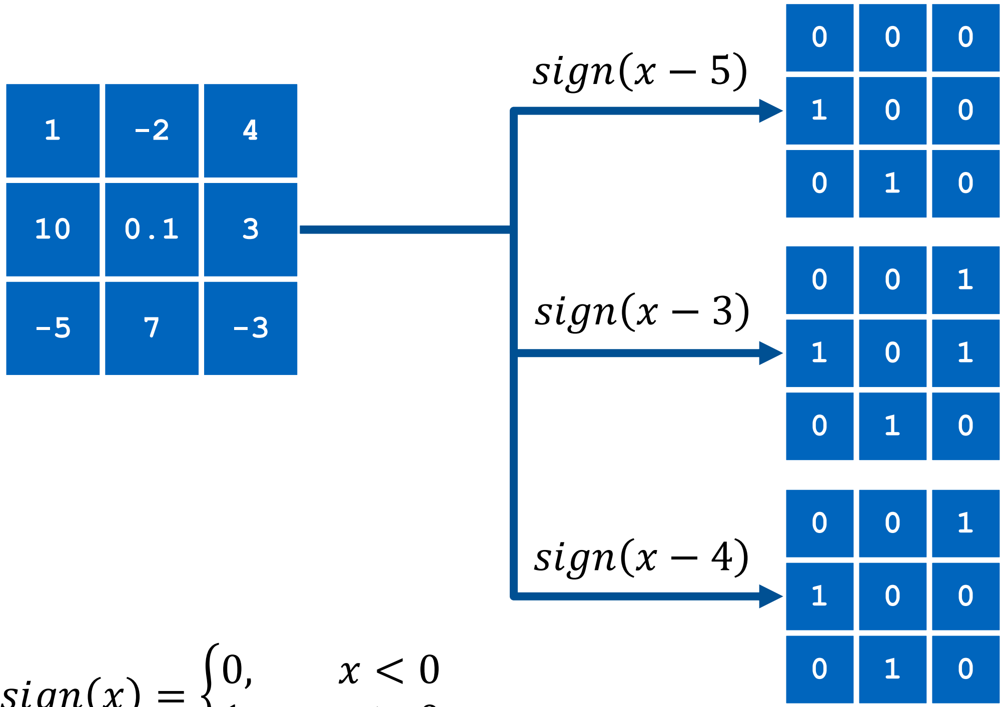
0.1 is less positive than **10**

4 and 3 lie between **10 and 0.1**

4 and 3 are single unit away from each other

Overview of Binary Neural Networks

Approximation through binary bases



$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Captured Information:

0.1 is less positive than **10**

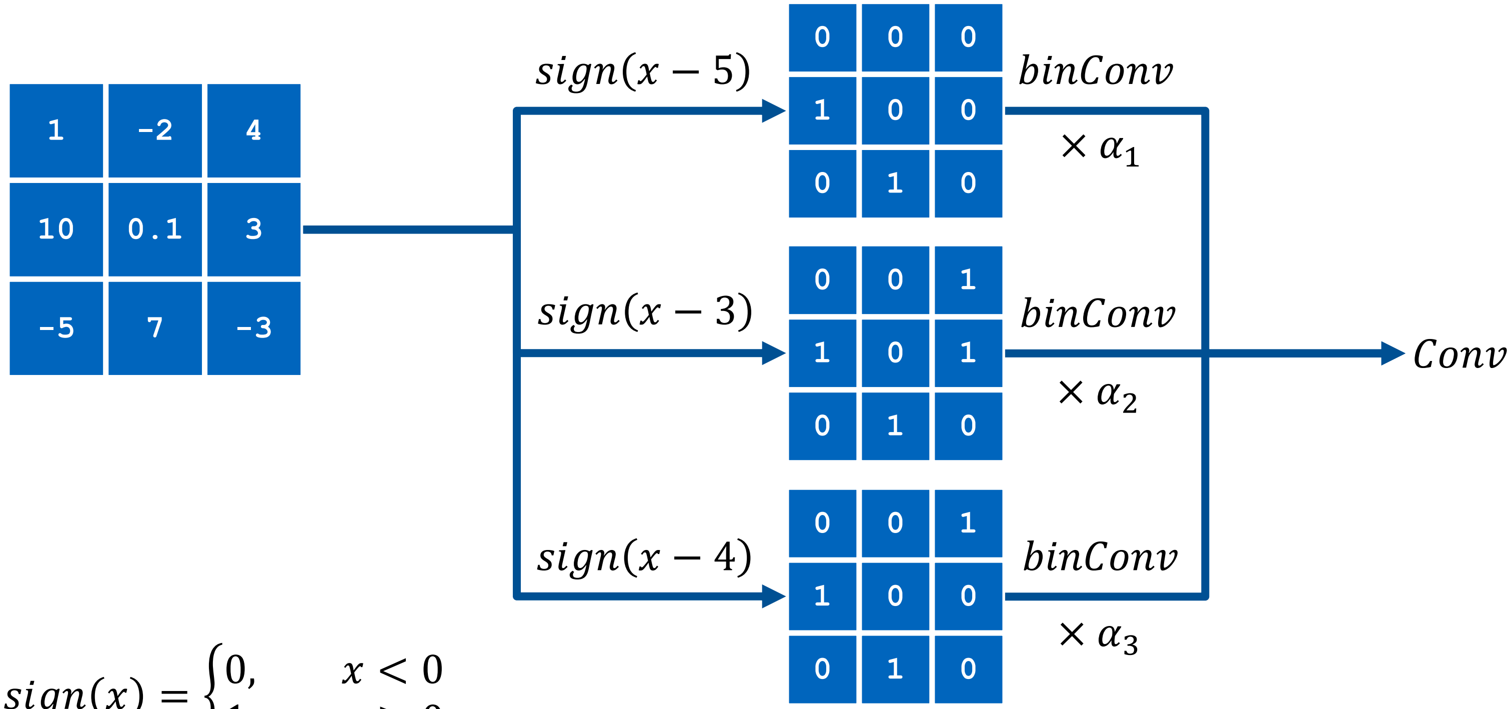
4 and 3 lie between **10 and 0.1**

4 and 3 are single unit away from each other

Information can be extracted for negative numbers, e.g. $sign(x+3)$

Overview of Binary Neural Networks

Approximation through binary bases

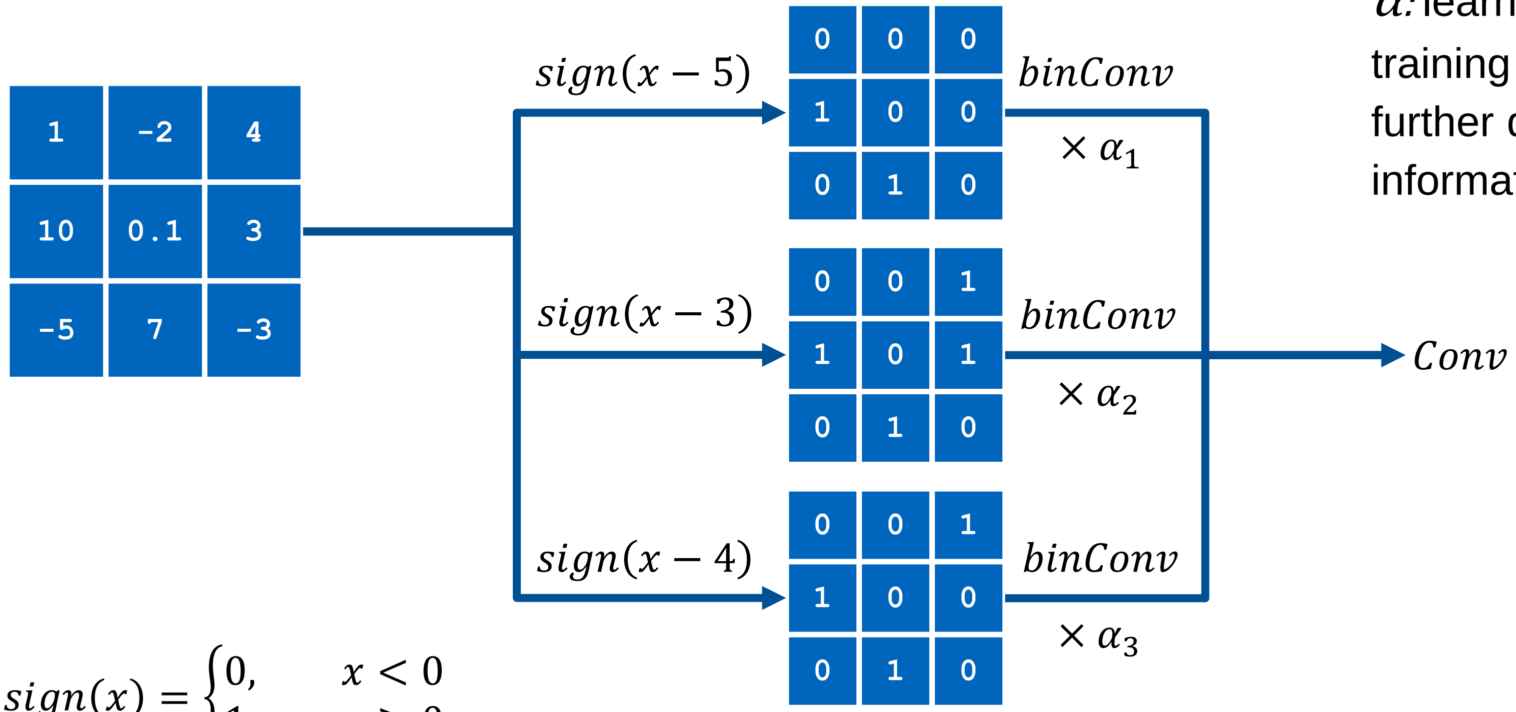


$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

[1] X. Lin et al., “Towards accurate binary convolutional neural network,” NIPS, 2017.

Overview of Binary Neural Networks

Approximation through binary bases



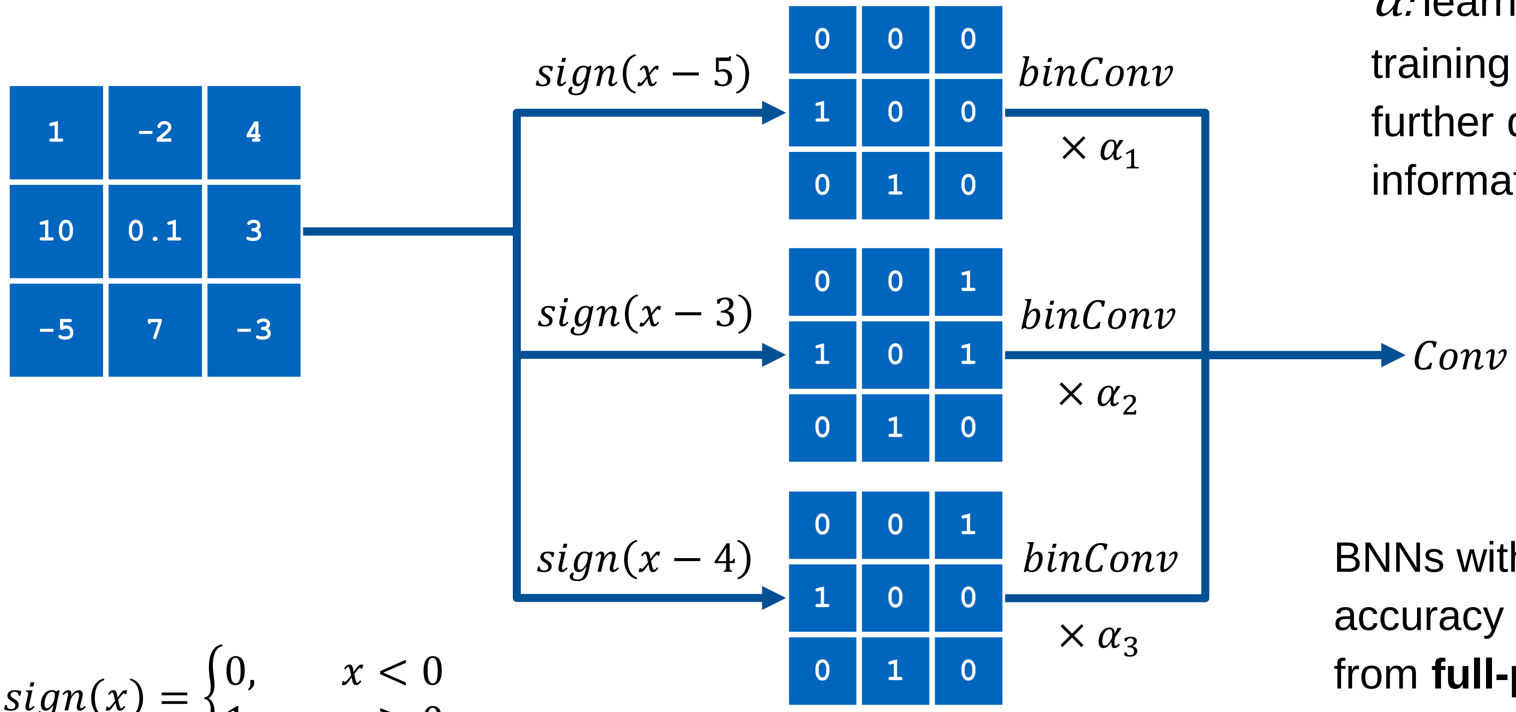
α: learned during training to induce further diversity in information.

$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

[1] X. Lin et al., “Towards accurate binary convolutional neural network,” NIPS, 2017.

Overview of Binary Neural Networks

Approximation through binary bases



α: learned during training to induce further diversity in information.

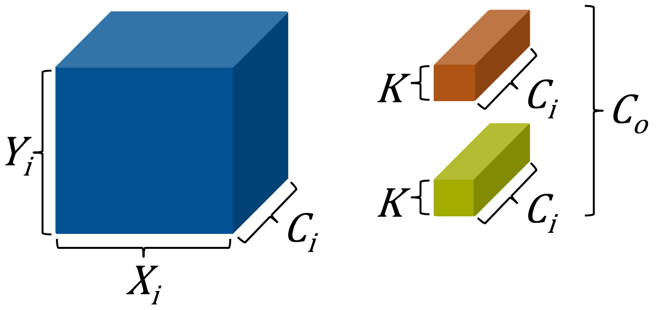
BNNs with only **4-5%** accuracy degradation from **full-precision** CNN on **ImageNet** [1].

$$sign(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

[1] X. Lin et al., “Towards accurate binary convolutional neural network,” NIPS, 2017.

How Binary are Binary Neural Networks?

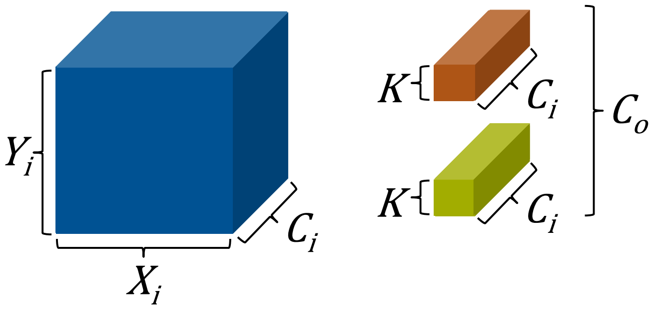
$A^{l-1} \in \mathbb{R}^{X_i \times Y_i \times C_i}$ $W^l \in \mathbb{R}^{K \times K \times C_i \times C_o}$



$$A^l = Conv(W^l, A^{l-1})$$

How Binary are Binary Neural Networks?

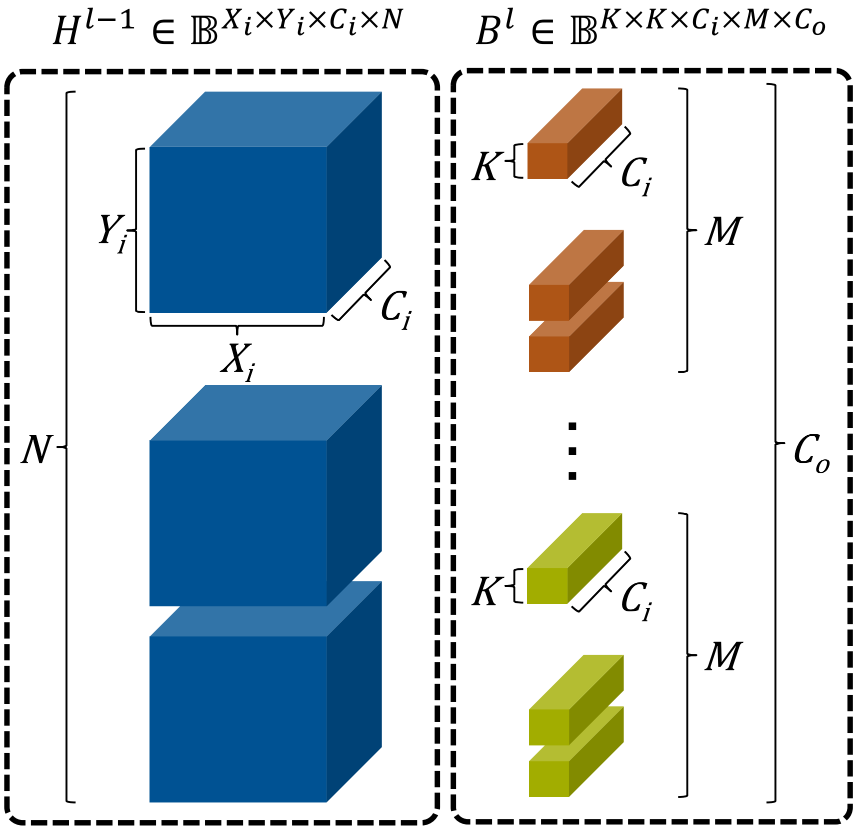
$H^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i \times N}$ $\quad$ $B^l \in \mathbb{B}^{K \times K \times C_i \times M \times C_o}$



$$A^l = Conv(W^l, A^{l-1})$$

$$\mathbb{B} = \{0,1\}$$

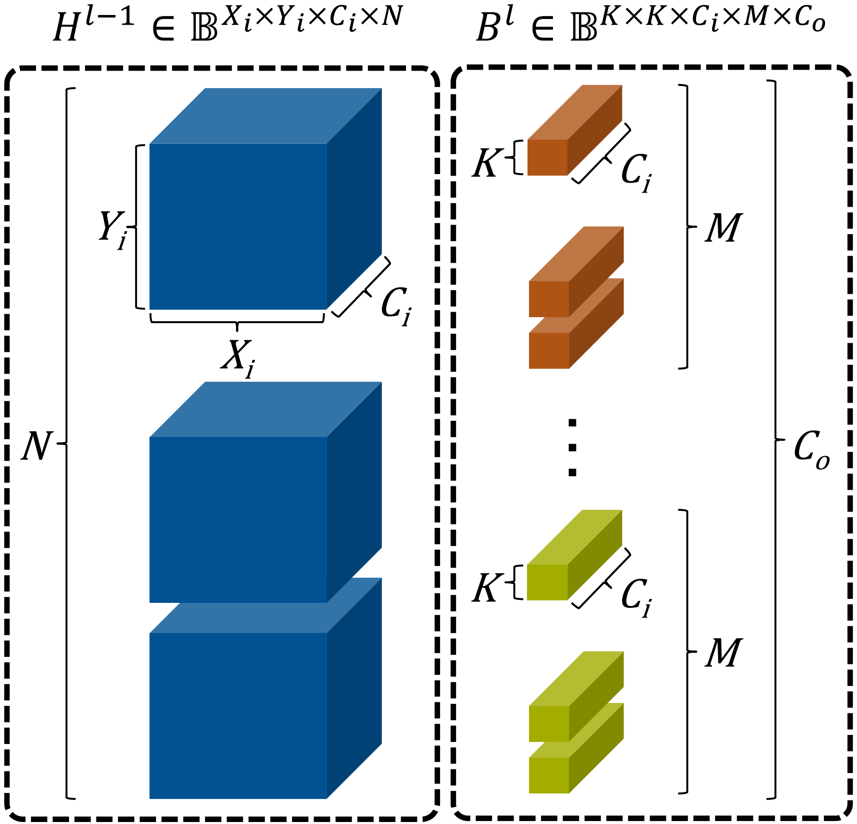
How Binary are Binary Neural Networks?



$\mathbb{B} = \{0,1\}$

$$A^l = Conv(W^l, A^{l-1})$$

How Binary are Binary Neural Networks?

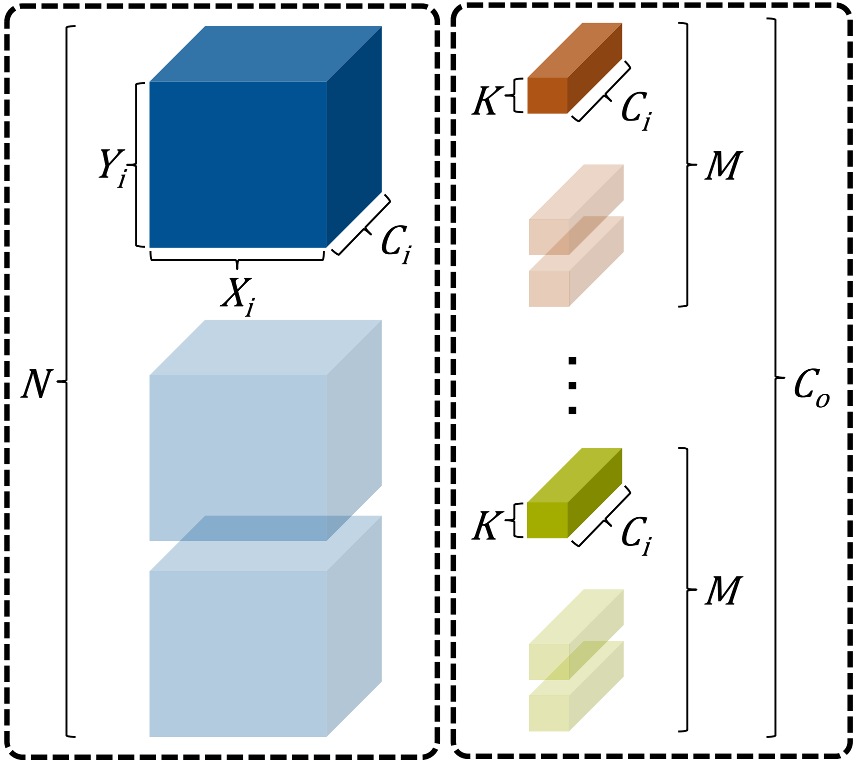


$$A^{l} = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^{l}, H_n^{l-1})$$

$\mathbb{B} = \{0,1\}$

How Binary are Binary Neural Networks?

$H_n^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i}$ $\quad$ $B_m^l \in \mathbb{B}^{K \times K \times C_i \times C_o}$

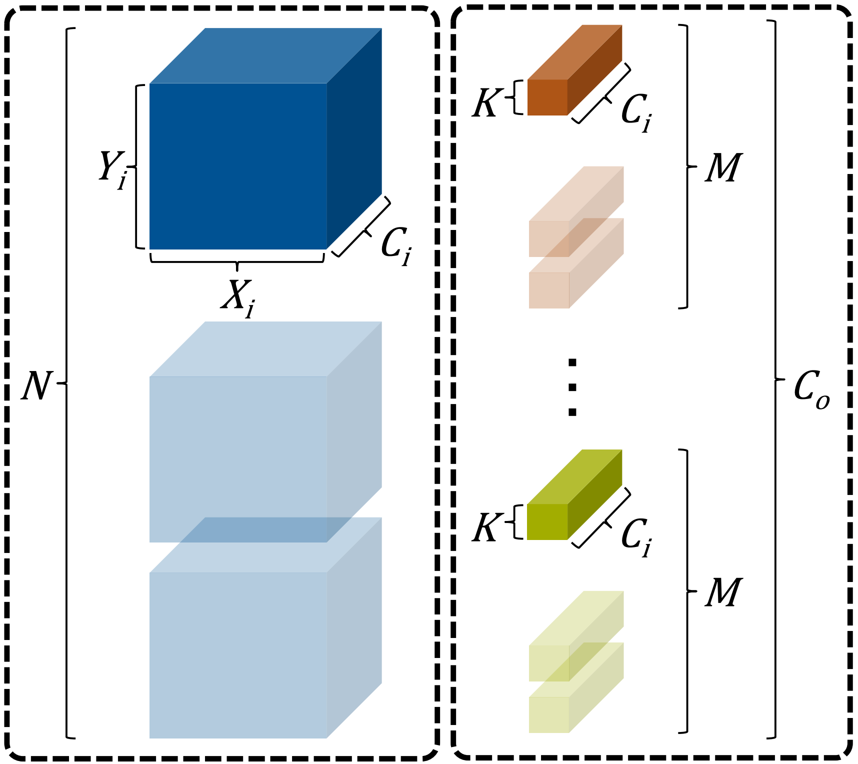


$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

$\mathbb{B} = \{0,1\}$

How Binary are Binary Neural Networks?

$H_n^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i}$ $\quad$ $B_m^l \in \mathbb{B}^{K \times K \times C_i \times C_o}$



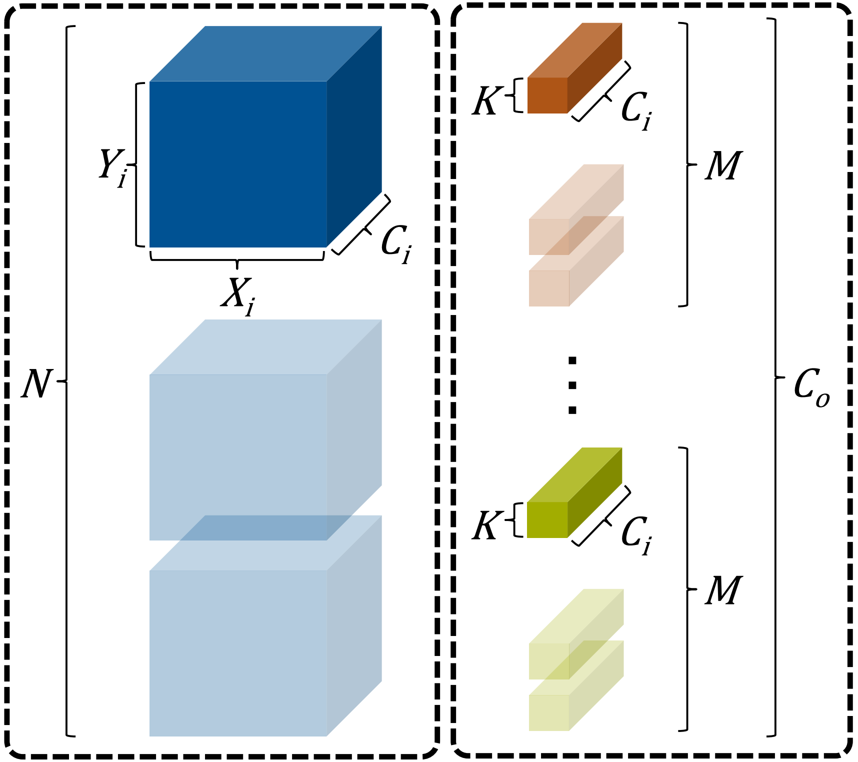
$\mathbb{B} = \{0,1\}$

$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

$$a_{m,n} = \sum_{c_i=1}^{C_i} (p_{m,n,c_i})$$

How Binary are Binary Neural Networks?

$H_n^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i}$ $\quad$ $B_m^l \in \mathbb{B}^{K \times K \times C_i \times C_o}$



$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

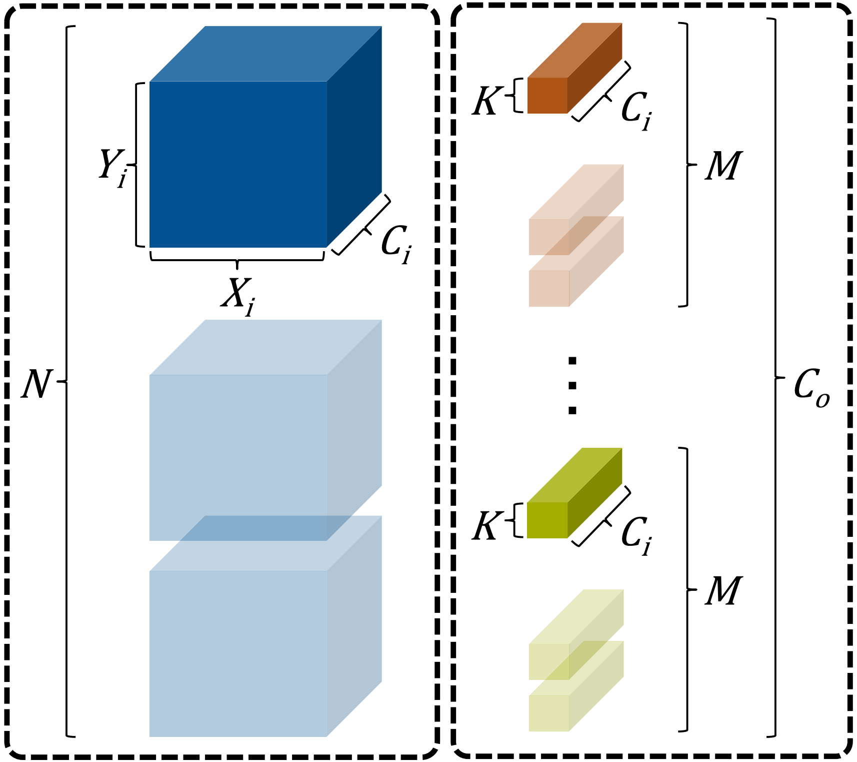
$$a_{m,n} = \sum_{c_i=1}^{C_i} (p_{m,n,c_i})$$

$$p_{m,n,c_i} = \sum_{k_x=1}^{K} \sum_{k_y=1}^{K} xnor\left(b_{k_x,k_y}, h_{x_i+k_x, y_i+k_y}\right)$$

$\mathbb{B} = \{0,1\}$

How Binary are Binary Neural Networks?

$H_n^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i}$ $B_m^l \in \mathbb{B}^{K \times K \times C_i \times C_o}$



$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

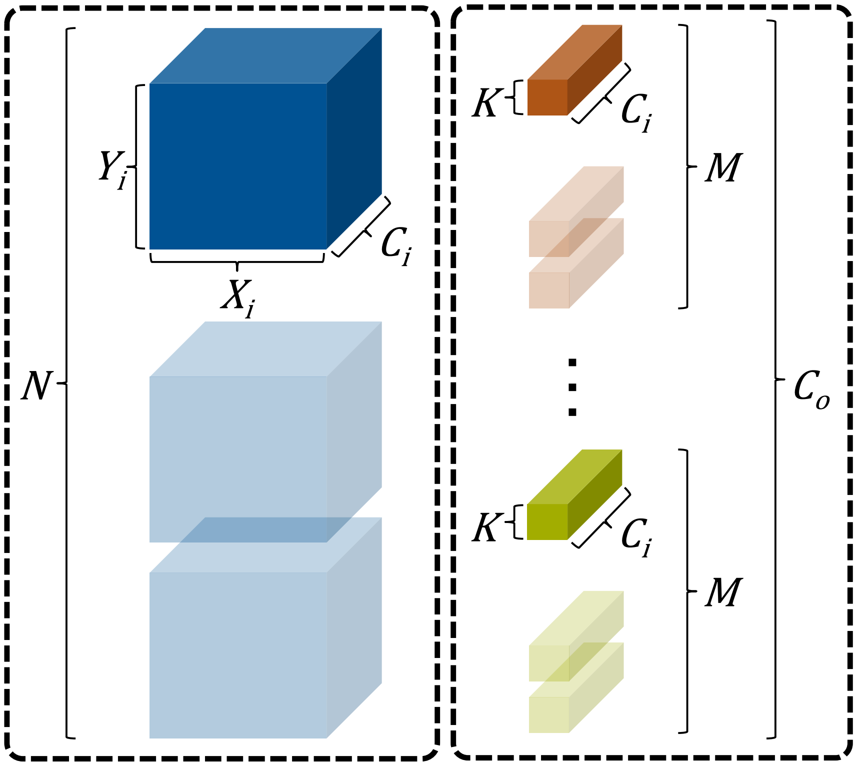
$$a_{m,n} = \sum_{c_i=1}^{C_i} (p_{m,n,c_i})$$

$$p_{m,n,c_i} = popcnt\left(xnor\left(b_{k_x,k_y}, h_{x_i+k_x, y_i+k_y}\right)\right)$$

$\mathbb{B} = \{0,1\}$

How Binary are Binary Neural Networks?

$H_n^{l-1} \in \mathbb{B}^{X_i \times Y_i \times C_i}$ $\quad B_m^l \in \mathbb{B}^{K \times K \times C_i \times C_o}$



$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

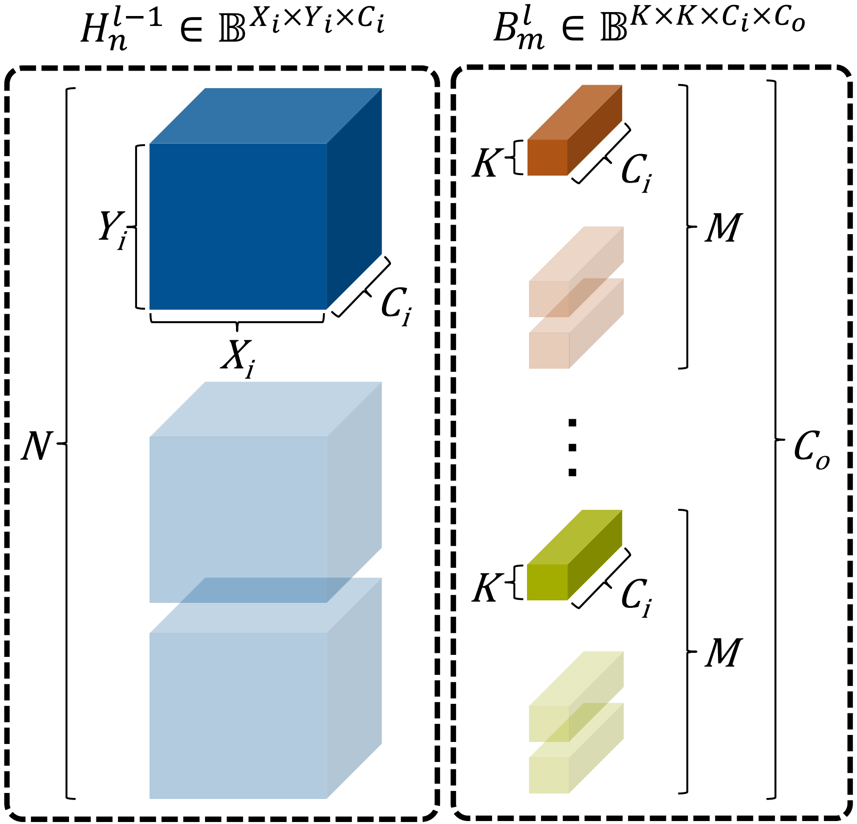
$$a_{m,n} = \sum_{c_i=1}^{C_i} (p_{m,n,c_i})$$

$$p_{m,n,c_i} = popcnt\left(xnor\left(b_{k_x,k_y}, h_{x_i+k_x,\, y_i+k_y}\right)\right)$$

$\mathbb{B} = \{0,1\}$

h^{l-1} $\quad b^l$

How Binary are Binary Neural Networks?



$$A^l = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_m \beta_n BinConv(B_m^l, H_n^{l-1})$$

$$a_{m,n} = \sum_{c_i=1}^{C_i} (p_{m,n,c_i})$$

$$\overbrace{p_{m,n,c_i}}^{\texttt{Fixed-point}} = popcnt\ (\underbrace{xnor\left(b_{k_x,\,k_y}, h_{x_i+k_x,\,y_i+k_y}\right)}_{\texttt{Binary}})$$

$\mathbb{B} = \{0,1\}$

More FP in Binary Neural Networks

- Binary Weight and Activation Bases (Scale/Shift)
- First Layer Remains Non-Binarized
- Batch Normalization

More FP in Binary Neural Networks

- Binary Weight and Activation Bases (Scale/Shift)
- First Layer Remains Non-Binarized
- Batch Normalization

Fixed-Point Ops

Binary Ops

Motivation for Reconfigurable Processing Elements

OrthrusPE: Runtime Reconfigurable PEs for BNNs

Dual Modes

- **Fixed-precision mode:** First layer, Batch-norm, Scale and Shift Operations
- **Binary mode:** SIMD Binary Hadamard Products and Popcounts
- Achieved with high resource reuse

OrthrusPE: Binary Mode

Efficient SIMD Binary Hadamard Product Execution

Input: $h^{l\text{-}1} \subset H_n^{\,l\text{-}1}$

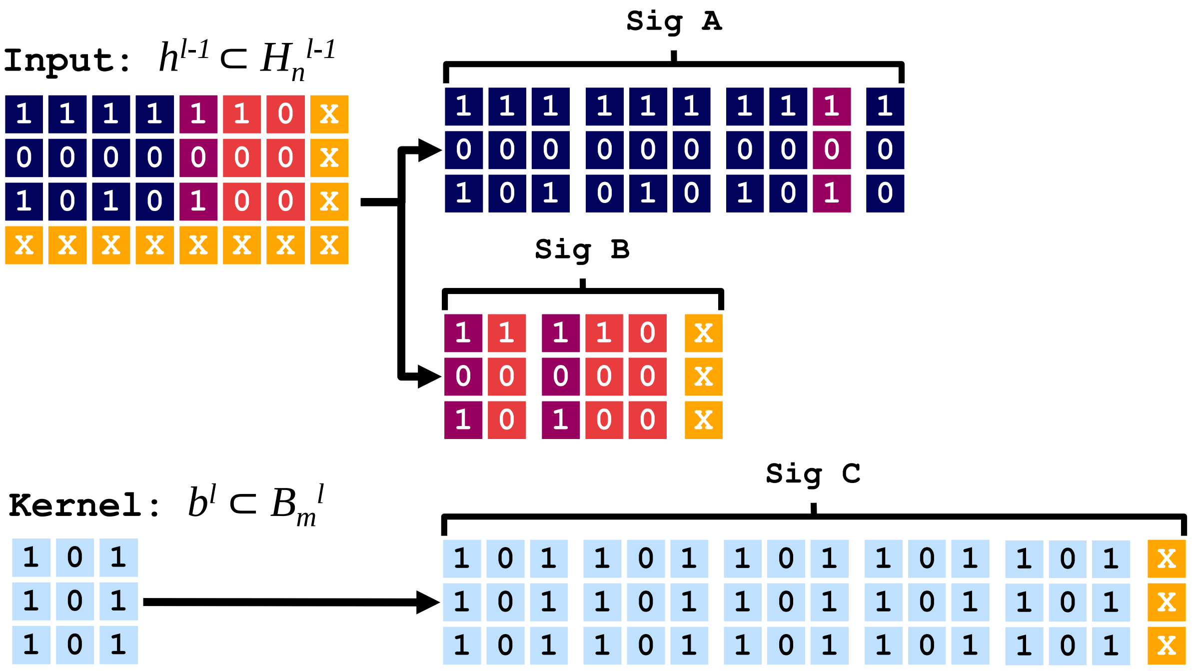
1	1	1	1	1	1	0	X
0	0	0	0	0	0	0	X
1	0	1	0	1	0	0	X
X	X	X	X	X	X	X	X

Kernel: $b^{l} \subset B_m^{\,l}$

1	0	1
1	0	1
1	0	1

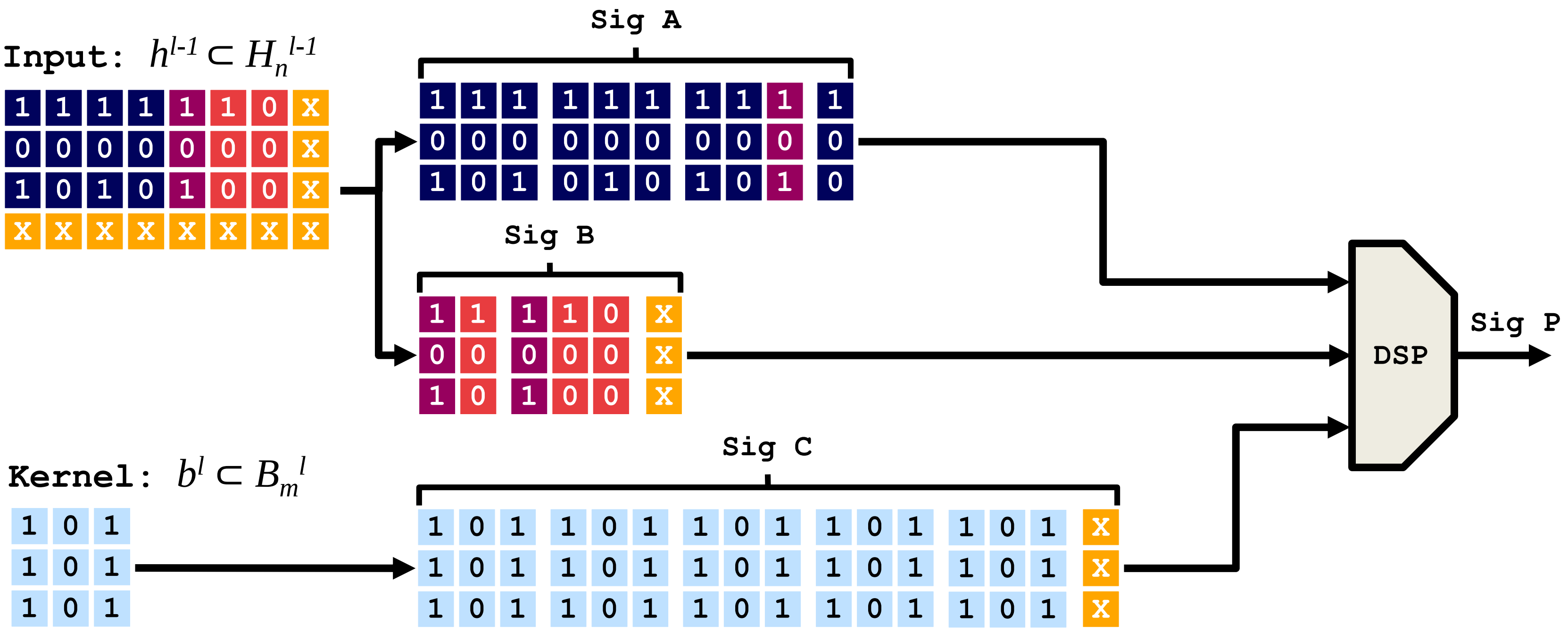
OrthrusPE: Binary Mode

Efficient SIMD Binary Hadamard Product Execution



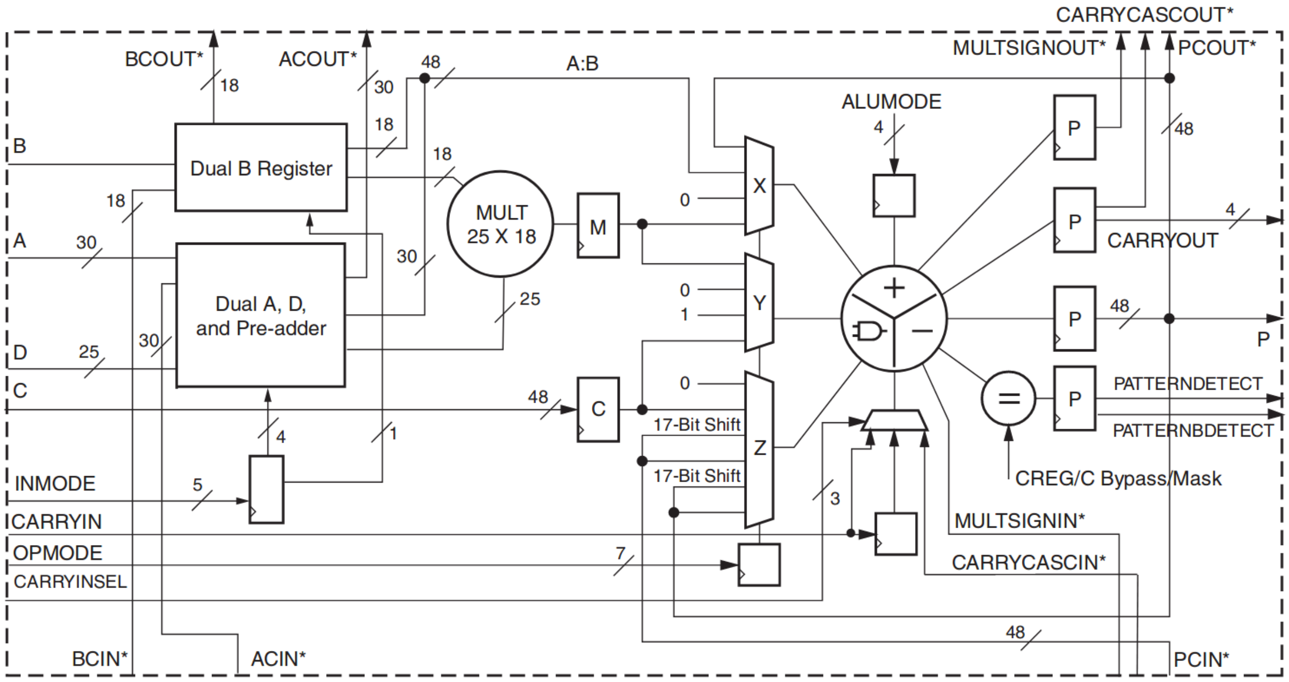
OrthrusPE: Binary Mode

Efficient SIMD Binary Hadamard Product Execution



OrthrusPE: Dual Modes

Runtime Reconfigurability



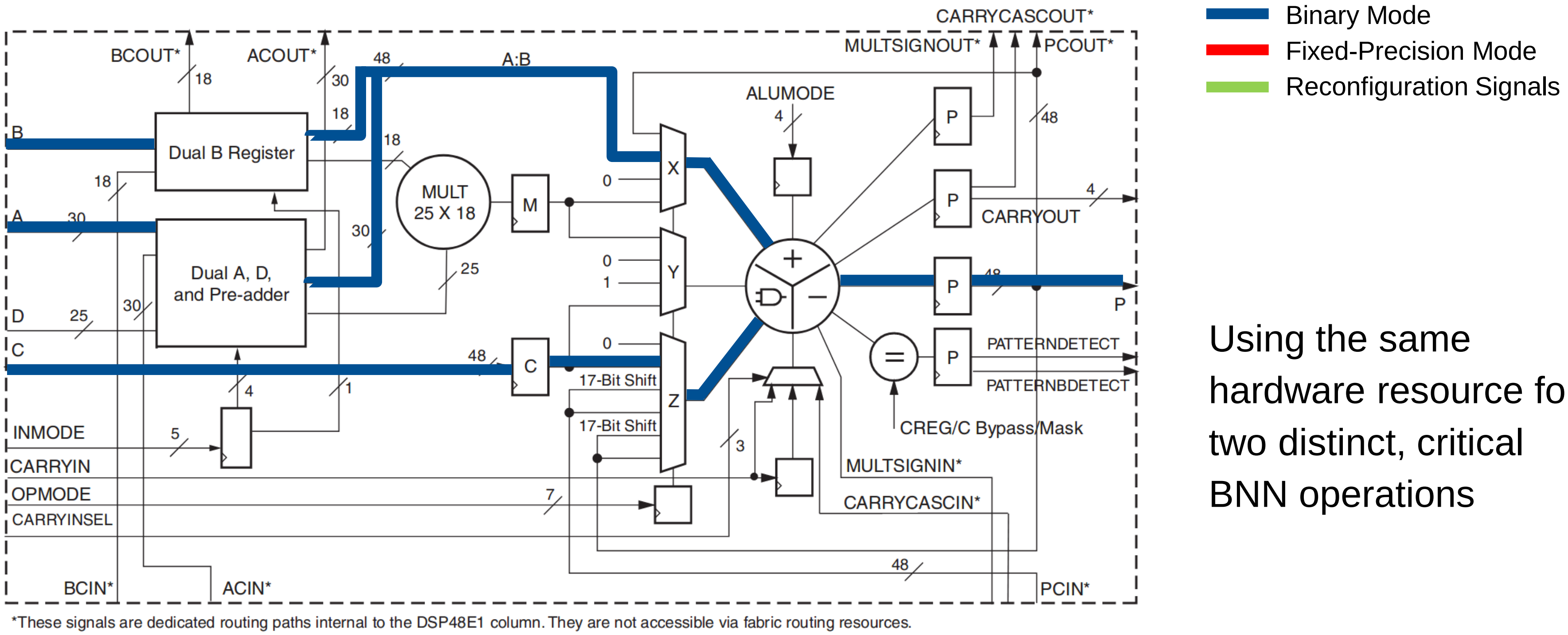
*These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.

Binary Mode
Fixed-Precision Mode
Reconfiguration Signals

Using the same hardware resource for two distinct, critical BNN operations

OrthrusPE: Dual Modes

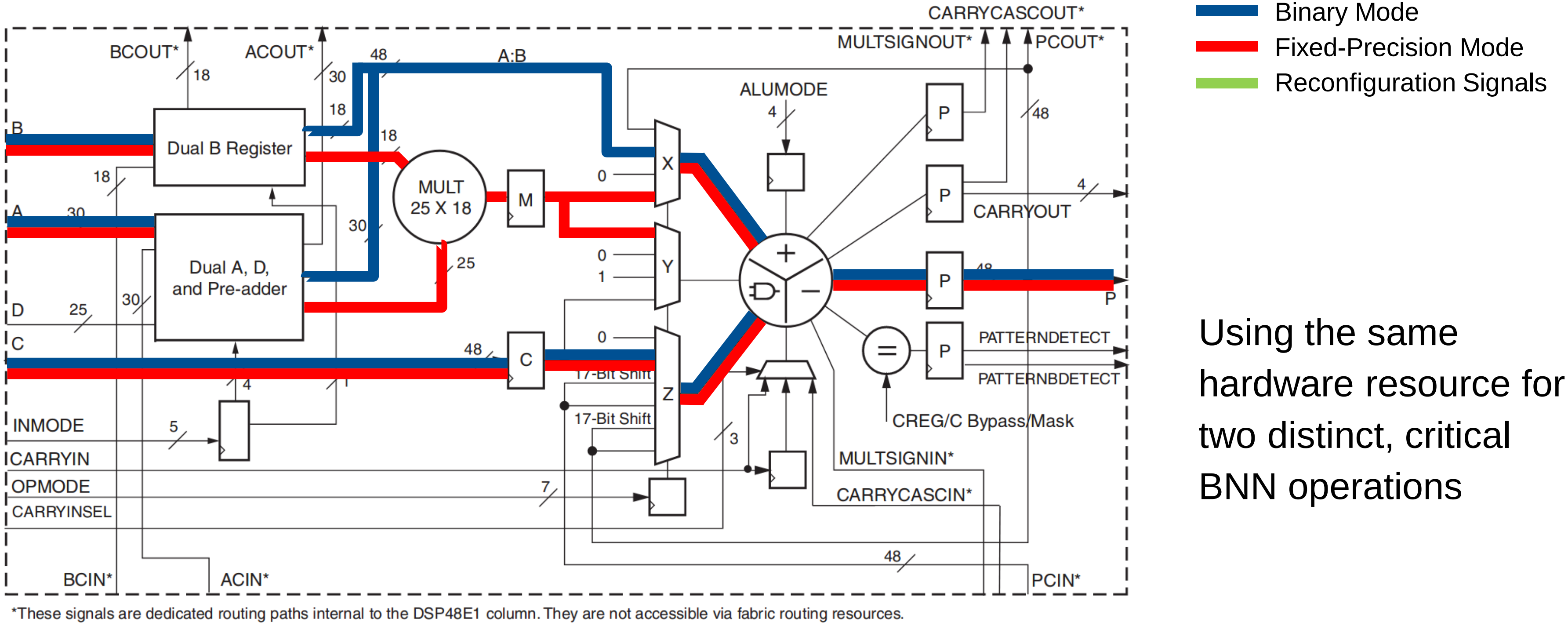
Runtime Reconfigurability



Using the same hardware resource for two distinct, critical BNN operations

OrthrusPE: Dual Modes

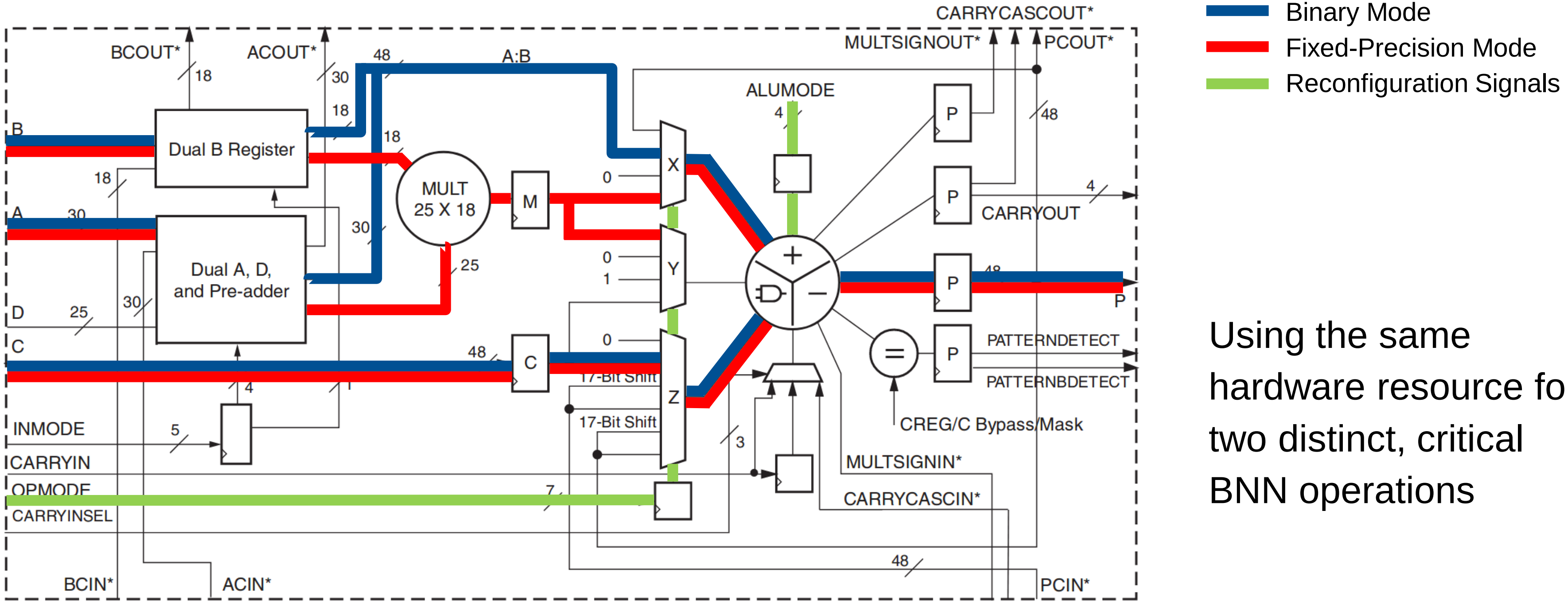
Runtime Reconfigurability



Using the same hardware resource for two distinct, critical BNN operations

OrthrusPE: Dual Modes

Runtime Reconfigurability



*These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.

Using the same hardware resource for two distinct, critical BNN operations

Experimentation and Evaluation

Synthesized Four Throughput-Equivalent Configurations:

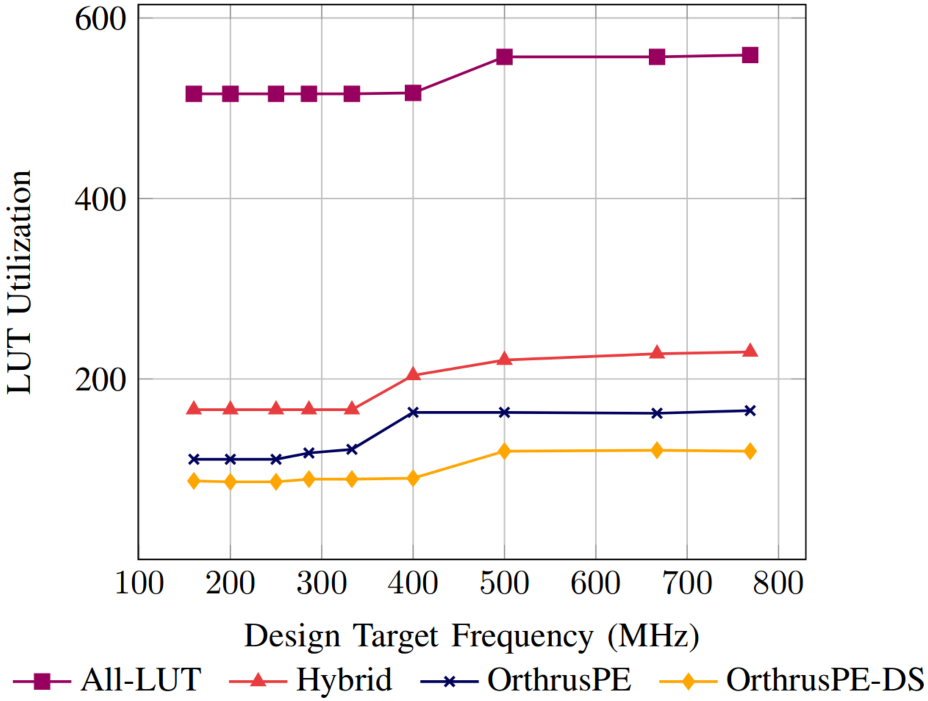
- OrthrusPE
- OrthrusPE-DS (Dual-Static): SIMD Binary Hadamard Products on Static DSP
- Hybrid (Common): Binary operations on LUTs, FP operations on DSP
- All-LUT: Execution restricted to LUTs

Experimentation and Evaluation

Resource Utilization

- OrthrusPE and OrthrusPE-DS are more resource efficient across all target accelerator frequencies.

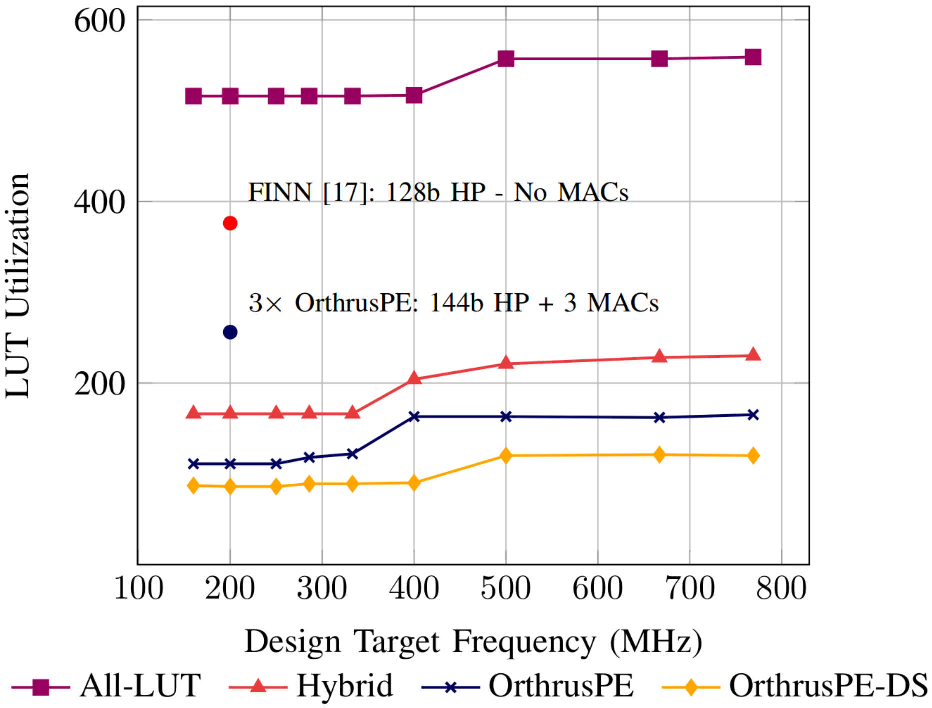
Implementation	F=770MHz			F=160MHz		
	LUTs	**FF**	**DSP**	**LUTs**	**FF**	**DSP**
All-LUT	559	160	0	516	160	0
Hybrid (Common)	230	253	1	166	253	1
OrthrusPE	**165**	210	1	**111**	210	1
OrthrusPE-DS	**120**	229	2	**87**	229	2



Experimentation and Evaluation

Resource Utilization

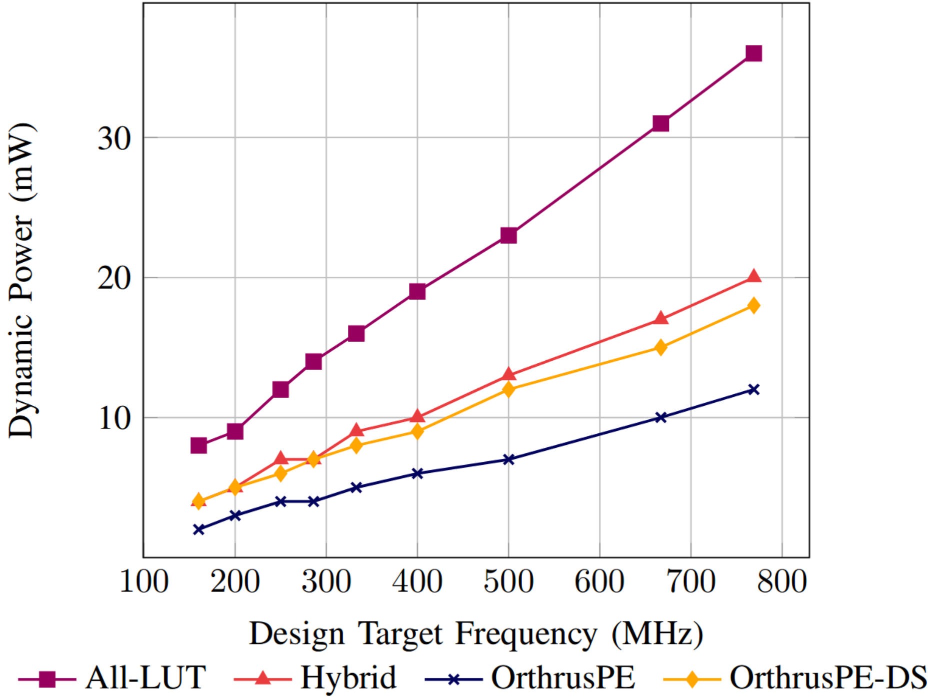
- OrthrusPE's closest FINN configuration @200MHz
  - 16 Extra Bit Accumulations
  - 3 MACs (through reconfigurability)
  - 32% fewer LUTs



Experimentation and Evaluation

Dynamic Power Estimation

- OrthrusPE more efficient across all frequencies
- Results scale as accelerators use 100-1000s of PEs



Conclusion and Future Work

- Accurate BNNs cannot be achieved without fixed-point operations and reliance on DSP blocks.

- OrthrusPE improves the efficiency of computation by executing both fixed-point and binary ops on FPGA hard blocks.

- Accurate BNNs solve many of the computation and memory challenges for deep neural network workloads on edge devices.

Thank you for your attention