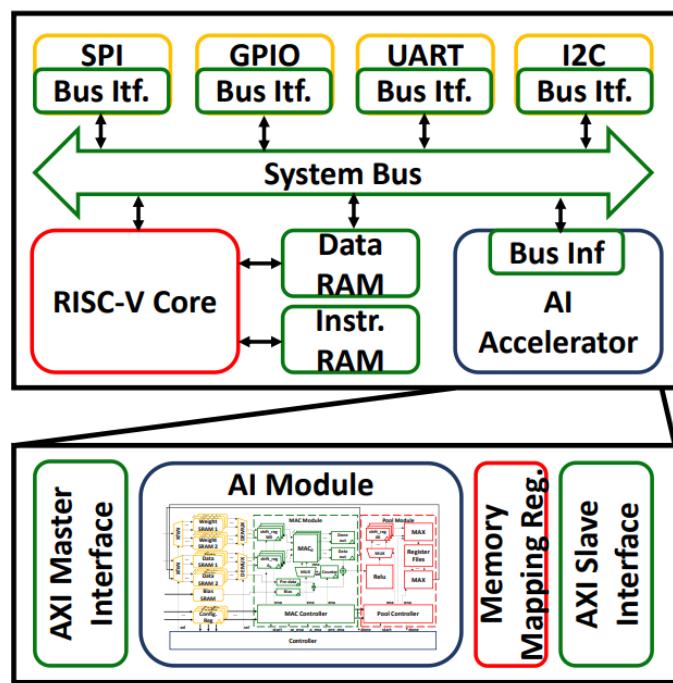


RISC-V AIoT System

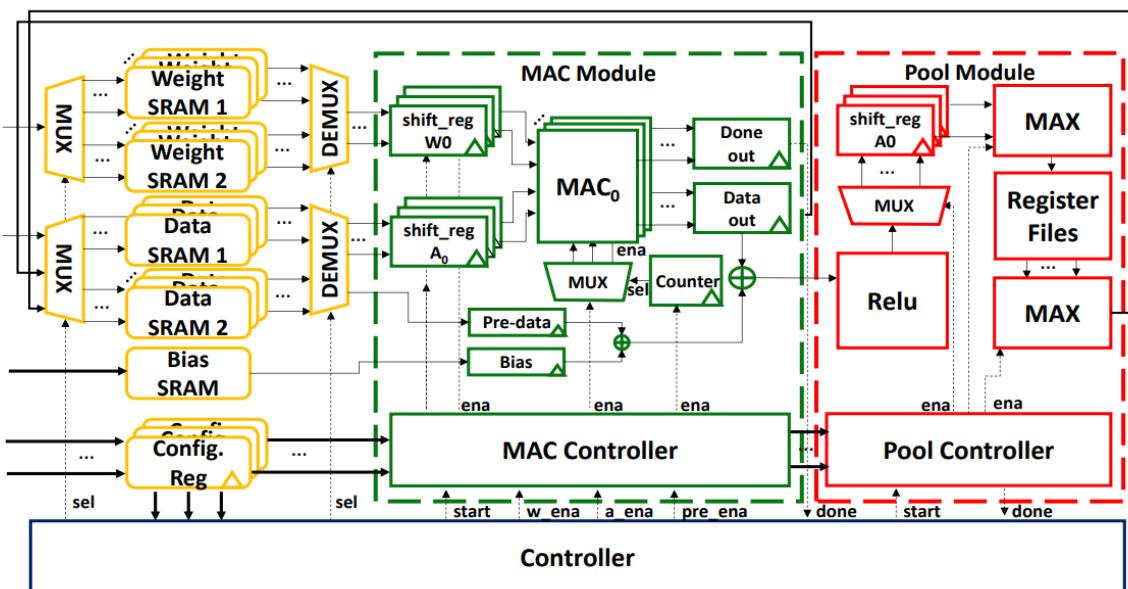
1. Khảo sát

1.1. Tiny Neuron Network System based on RISC-V Processor: A Decentralized Approach for IoT Applications

Kiến trúc đề xuất:



Kiến trúc khối AI

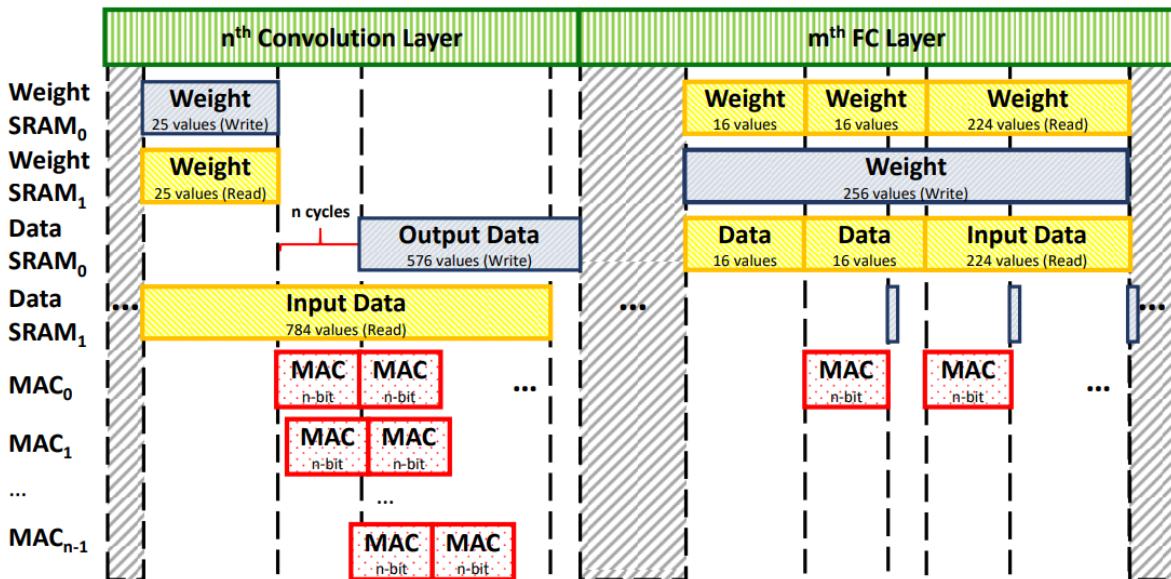


A. Khối MAC

Hiện thực hai mô hình MAC khác nhau: Variable-Bit-Precision (VBP) MAC và Stochastic (SC) MAC. Việc lựa chọn mô hình MAC nào được quyết định lúc thiết kế hệ thống.

B. Quản lý dữ liệu

Khối MAC hỗ trợ việc chuyển đổi giữa hai chế độ làm việc parallel và serial. Chế độ làm việc được lựa chọn phụ thuộc vào đặc tính của từng quá trình tính toán trong DNN.



Dữ liệu được gửi từ bộ nhớ hệ thống thông qua khối Direct Memory Access (DMA).

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Hệ thống được hiện thực bằng cách tích hợp khối AI lên SoC PULPino. SoC được hiện thực trên 2 công nghệ:

- FPGA hiện thực trên Arty A7 100T kit
- ASIC sử dụng công nghệ 65nm TSMC

Phần mềm được sử dụng để làm baseline so sánh là Pytorch framework.

Tập dữ liệu được sử dụng MNIST. Các DNN được sử dụng lần lượt là:

- ANN: là mạng 3 fully-connected layer gồm 1 input layer với 784 neuron, 1 hidden layer với 64 layer, và 1 output layer với 10 neuron
- CNN: là mạng gồm 2 convolution layer và 3 fully-connected layer với lần lượt 128, 84 và 10 neuron.

Kết quả thí nghiệm

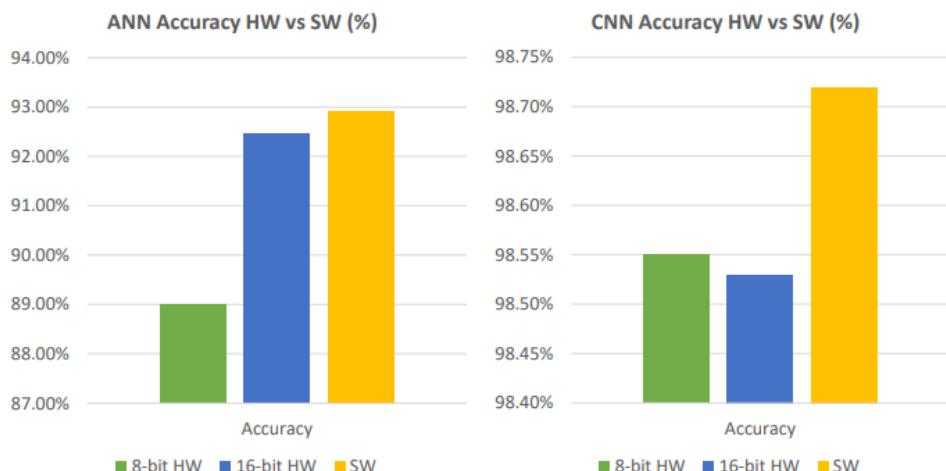
Thí nghiệm 1: So sánh kết quả hệ thống hiện thực với công nghệ ASIC ở tần số 25MHz với các cài đặt khác nhau từ mô hình của MAC cho tới độ chính xác (bit precision)

Configuration	Full SoC			AI IP		MAC Module		Bias SRAM	Weight SRAM	Data SRAM	Others
	Perf. (fps)	Power (mW)	Area (MGEs)	Power (mW)	Area (KGEs)	Power (mW)	Area (KGEs)	Area (KGEs)	Area (KGEs)	Area (KGEs)	Area (KGEs)
ANN&16-bit SC @25MHz	0.12 (28×28pix.)	47.305 (100%)	3.867	31.292 (66.2%)	1,424	8.51e-02 (0.2%)	9.89	41.55 (1024×16b) ¹	664.91 (1024×256b) ¹	664.91 (1024×256b) ¹	43.08
ANN&8-bit SC @25MHz	30 (28×28pix.)	31.835 (100%)	3.163	15.819 (49.7%)	720.18	5.93e-02 (0.2%)	5.40	20.77 (1024×8b) ¹	332.46 (1024×128b) ¹	332.46 (1024×128b) ¹	29.10
ANN&16-bit VBP @25MHz	488 (28×28pix.)	47.314 (100%)	3.87	31.303 (66.2%)	1,426	9.43e-02 (0.2%)	12.33	41.55 (1024×16b) ¹	664.91 (1024×256b) ¹	664.91 (1024×256b) ¹	43.06
ANN&8-bit VBP @25MHz	968 (28×28pix.)	31.822 (100%)	3.164	15.805 (49.7%)	720.46	4.31e-02 (0.1%)	5.69	20.77 (1024×8b) ¹	332.46 (1024×256b) ¹	332.46 (1024×256b) ¹	29.08
CNN&16-bit VBP @25MHz	250 (28×28pix.)	21.104 (100%)	3.037	5.088 (24.1%)	594.61	1.893 (9.0%)	141.41	18.78 (128×16b) ²	74.83 (512×16b) ²	297.64 (2048×16b) ²	61.91
CNN&8-bit VBP @25MHz	250 (28×28pix.)	19.750 (100%)	2.905	3.737 (18.9%)	461.74	0.762 (3.9%)	57.09	9.99 (128×8b) ²	39.66 (512×8b) ²	297.64 (2048×16b) ²	57.37

1: This memory is generated by using ARM Memory Compiler.

2: This memory is generated by using Synopsys Design Compiler.

Thí nghiệm 2: So sánh độ chính xác của mô hình ANN và CNN có độ chính xác là 8-bit và 16-bit với kết quả của mô hình phần mềm (Pytorch)



Thí nghiệm 3: So sánh kết quả hệ thống hiện thực với công nghệ FPGA ở các độ chính xác là 8-bit và 16-bit với các công trình trước đó. Các công trình này lần lượt là:

- M. Sankaradas, V. Jakkula, S. Cadambi, S. Chakradhar, I. Durdanovic, E. Cosatto, and H. P. Graf, “A massively parallel coprocessor for convolutional neural networks,” in 2009 20th IEEE International Conference on Application-specific Systems, Architectures and Processors, 2009, pp. 53–60.
- C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, “Optimizing fpga-based accelerator design for deep convolutional neural networks.” New York, NY, USA: Association for Computing Machinery, 2015.
- X. Zhang, A. Ramachandran, C. Zhuge, D. He, W. Zuo, Z. Cheng, K. Rupnow, and D. Chen, “Machine learning on fpgas to face the iot revolution,” in 2017 IEEE/ACM International Conference on ComputerAided Design (ICCAD), 2017, pp. 819–826

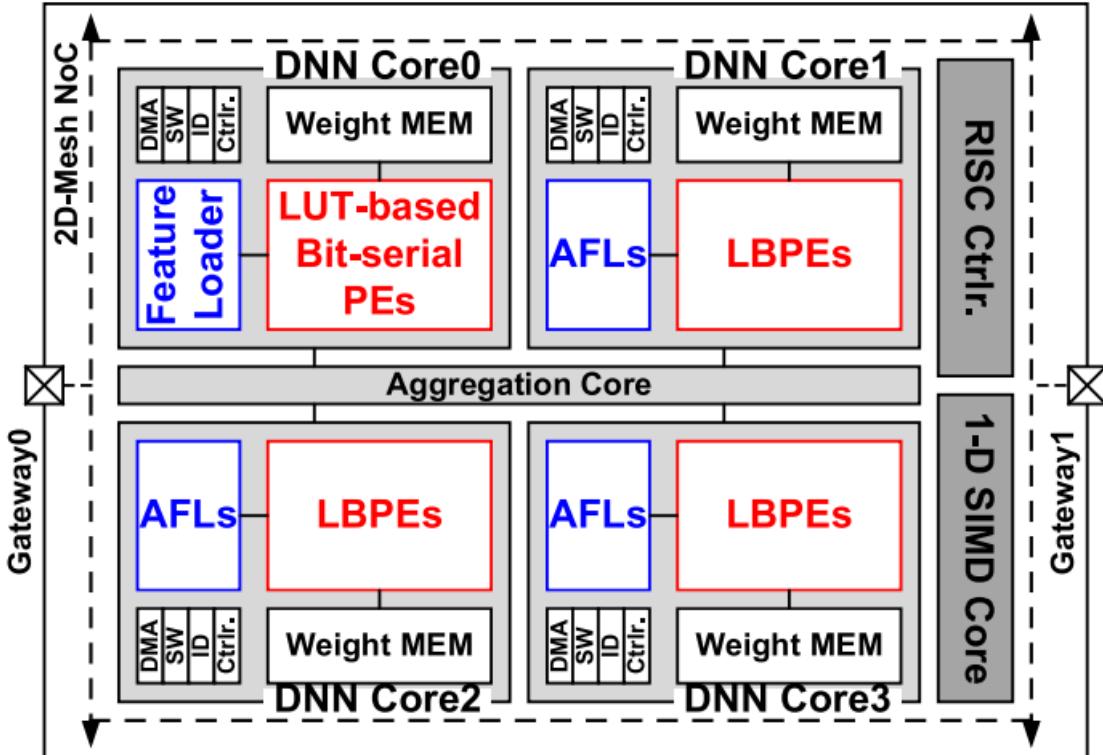
	Our Work				Zhang'17	Chen'15	Sanka.'09
	16-bit SoC	16-bit AI IP	8-bit SoC	8-bit AI IP	8-bit AI IP	32-bit AI IP	16-bit Coprocessor
FPGA Board	Arty A7 100T				Virtex-7 VC709	Virtex -7 VC707	Virtex-5
Frequency	50MHz	50MHz	50MHz	50MHz	100MHz	100MHz	230MHz
Slice LUTs	42792 (67.50%)	18826 (29.69%)	31818 (50.19%)	7842 (12.37%)	316250 (73%)	186251 (61.3%)	35263 (17%)
Slice registers	24218 (19.10%)	8701 (6.86%)	19765 (15.59%)	4248 (3.35%)	231165 (37%)	205704 (33.87%)	39501 (19%)
Slice	12749 (80.44%)	5418 (34.18%)	9931 (62.66%)	2413 (15.22%)	-	-	-
Block RAM	72.5 (53.70%)	2.5 (1.85%)	72.5 (53.70%)	2.5 (1.85%)	1508 (51%)	1024 (50%)	3 (1%)
DSP	0 (0%)	0 (0%)	0 (0%)	0 (0%)	3130 (87%)	2240 (80%)	107 (55%)

1.2. UNPU: An Energy-Efficient Deep Neural Network Accelerator With Fully Variable Weight Bit Precision (BASELINE)

Ý tưởng:

- Giới hạn về độ chính xác:
 - Các DNN khác nhau có các giá trị tối ưu về độ chính xác của weight khác nhau.
 - Các layer khác nhau ở trong một DNN có các giá trị tối ưu về độ chính xác của weight khác nhau.
 - Độ chính xác của weight cần cho tác vụ suy diễn của DNN phụ thuộc vào độ chính xác mong muốn của DNN
- Sự cần thiết của một phần cứng cho các tính toán ở các layer khác nhau (Convolution Layer, Recurrent Layer và Fully-Connected Layer)

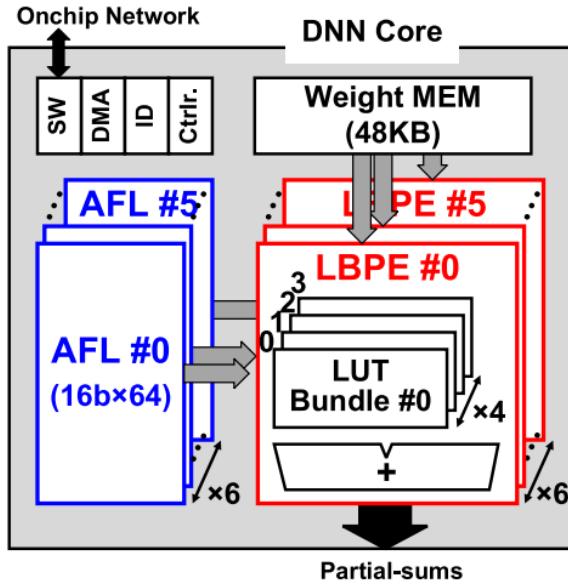
Kiến trúc hệ thống:



Hệ thống bao gồm: 2-D mesh type network-on-chip (NoC), 4 unified DNN cores, 1-D SIMD core, RISC controller, aggregation core, và two external gateways.

- 4 unified DNN core thực hiện tính toán độc lập tạo ra các tổng một phần (partial sum) và gửi kết quả tới aggregation core để thực hiện cộng tích lũy tạo ra kết quả cuối cùng
- 1-D SIMD core thực hiện các tính toán vectore như non-linear activation function, element-wise multiplication,...
- RISC controller thực thi lệnh cần cho việc trao đổi dữ liệu giữa các core với nhau thông qua NoC
- External gateway kết nối UNPU với bộ nhớ bên ngoài để tải các giá trị weight và input feature map từ bộ nhớ ngoài vào UNPU hoặc ghi kết quả từ UNPU ra bộ nhớ ngoài

A. Unified DNN core

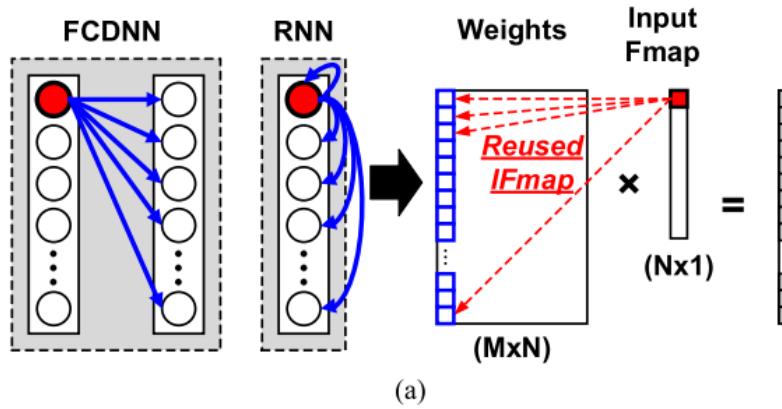


Unified DNN core bao gồm: switch, DMA controller, custom instruction decoder và controller, 6 cặp AFL và LBPE và weight MEM,

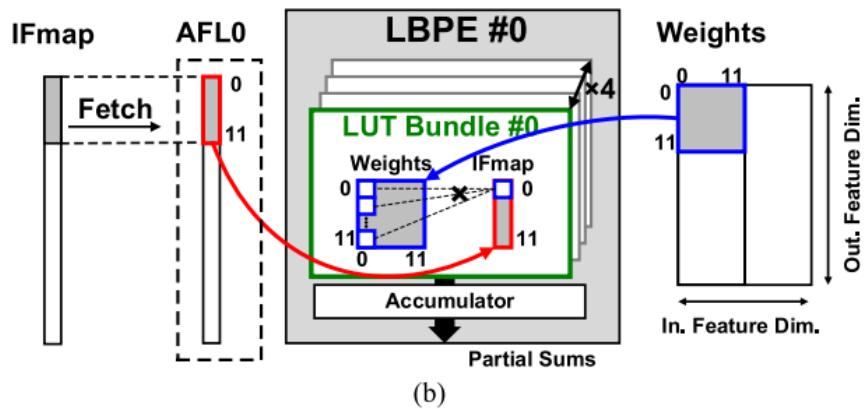
- Switch, DMA controller kết nối với NoC thực hiện giao tiếp giữa các core với nhau
- Custom instruction decoder và controller thực thi các lệnh tương ứng với custom instruction set
- Weight MEM lưu trữ các giá trị weight và gửi tới cho 6 LBPE bằng cách broadcast
- LBPE là thành phần xử lý thực thi các tác vụ nhân ma trận trong DNN. LBPE thực hiện tác vụ MAC bằng cách sử dụng Look-up Table (LUT) và hỗ trợ độ chính xác của weight từ 1 đến 16 bit theo phương pháp bit-serial. Mỗi LPBE bao gồm 4 LUT Bundle và mỗi LUT Bundle thực hiện tính toán riêng biệt và sau đó cộng lại để tạo ra kết quả một phần của LPBE.
- AFL lưu trữ các giá trị của input feature map tới cho LPBE sau khi đã căn chỉnh input feature map.

B. Dataflow

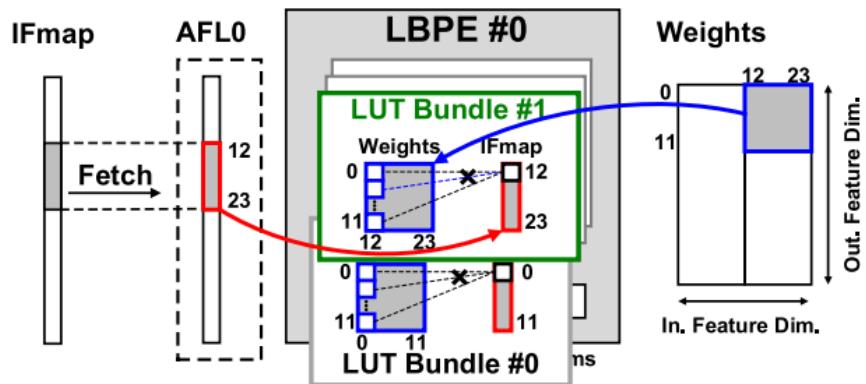
Recurrent Layers hoặc Fully-connected Layers



(a)

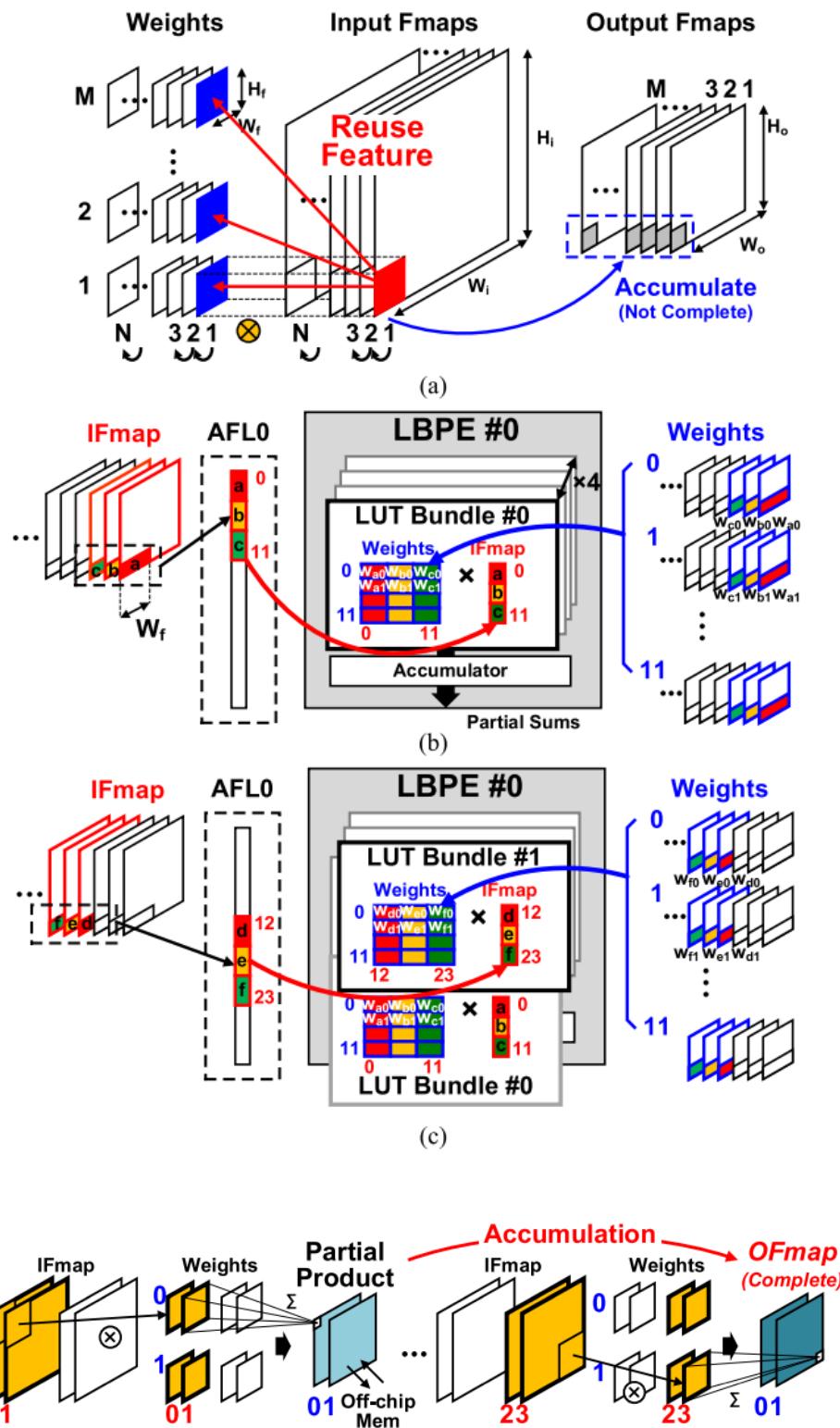


(b)



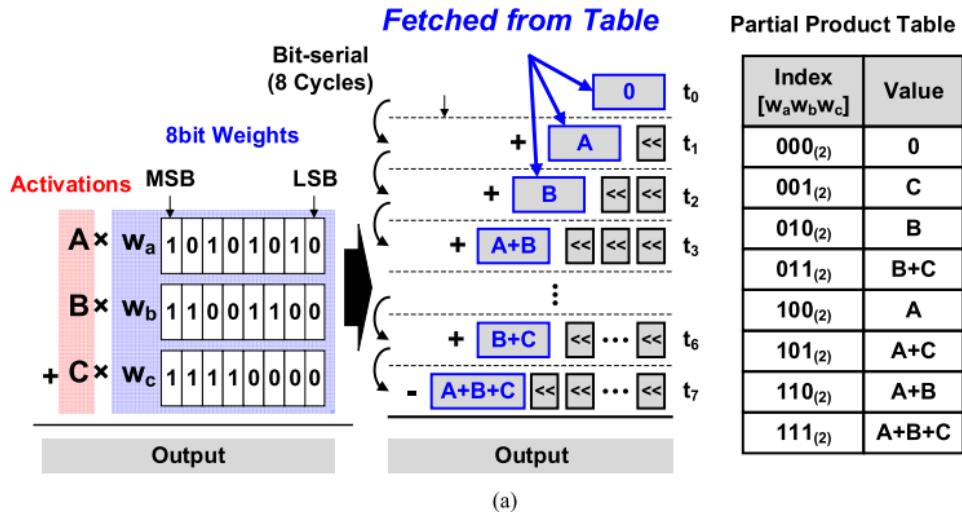
(c)

Convolution Layers

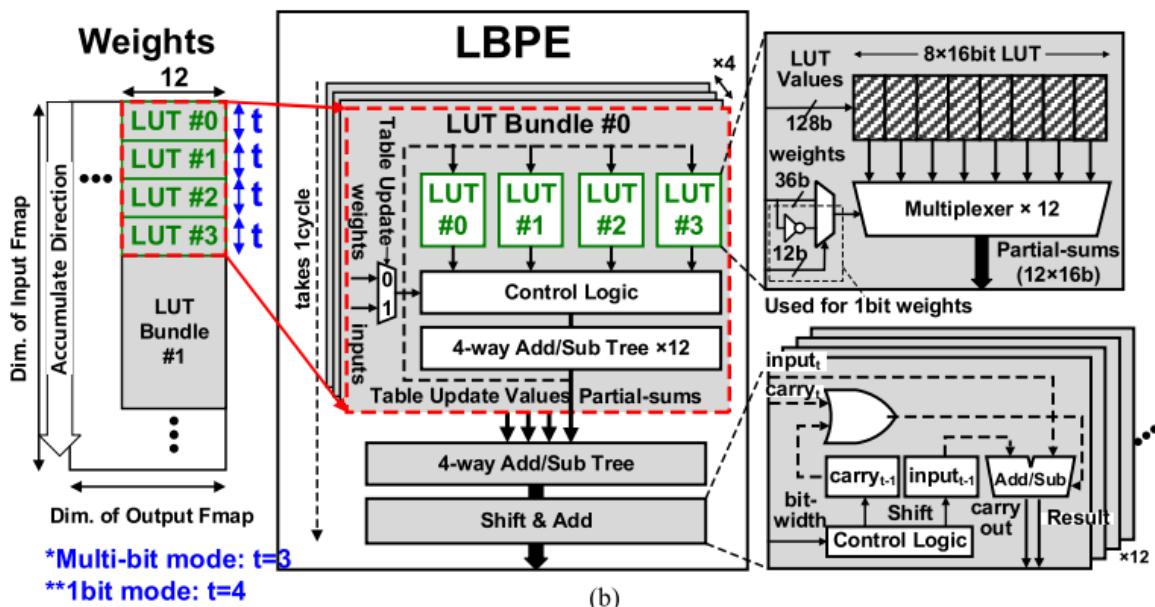


C. Lookup table-based Bit-serial Processing Element (LBPE)

Nguyên lý chính của LBPE dựa trên việc tái sử dụng feature map và tác vụ MAC dựa trên phương pháp bit-serial.



Kiến trúc của LBPE:

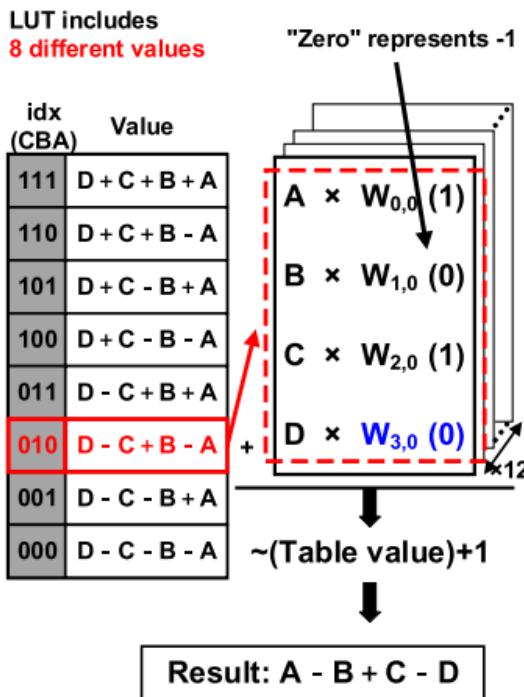


LBPE bao gồm: 4 LUT bundle, four-way add/sub tree và shift-and-add logic

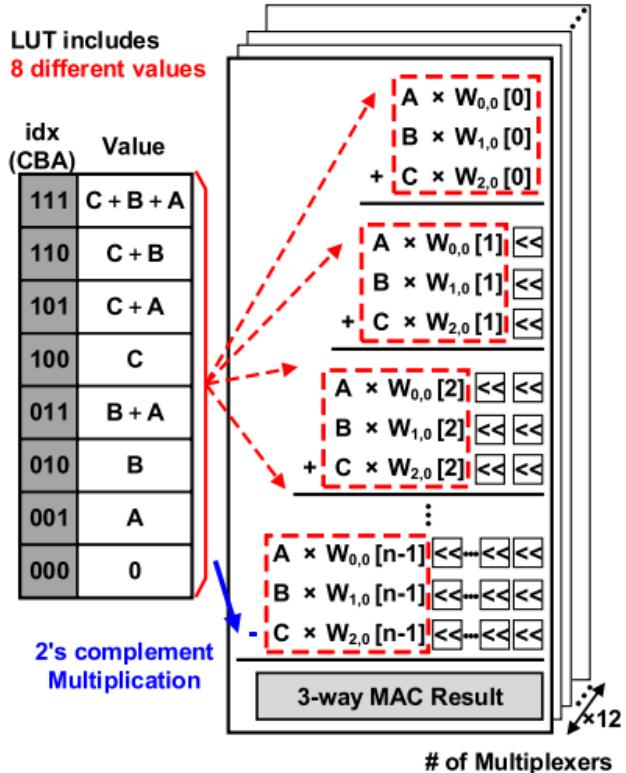
- Mỗi LUT bundle bao gồm 4 khối LUT, control logic và four-way add/sub trees.
- Four-way add/sub tree cộng tích lũy kết quả tính toán của các LUT bundle
- Shift-and-add logic thực hiện tác vụ MAC bằng cách dịch và cộng tích lũy các tổng bán phần.

Các chế độ tính toán của LBPE

Case1) 1-bit Weight

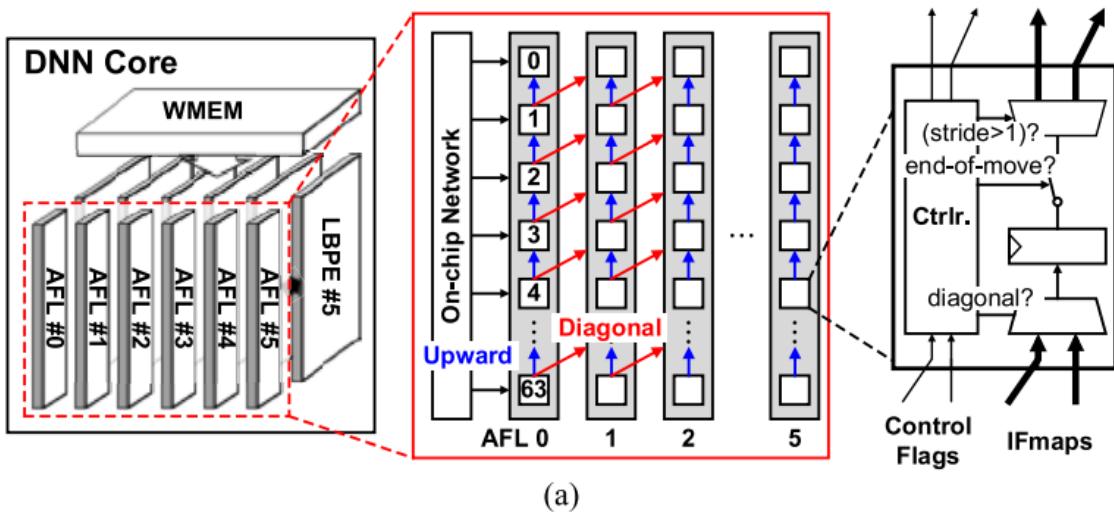


Case2) n-bit Weight



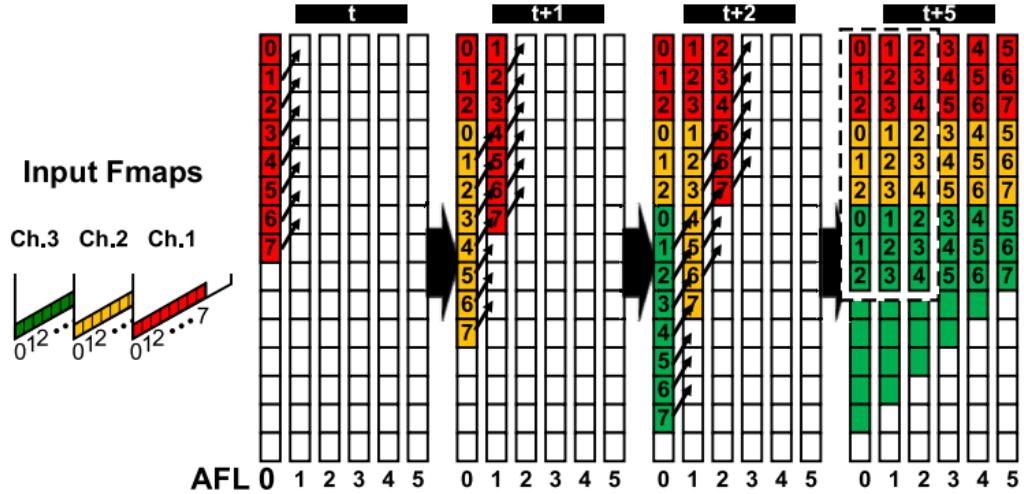
D. Aligned Feature map Loader (AFL)

Kiến trúc của AFL



Mỗi AFL bao gồm 64 thành phần con và mỗi thành phần bao gồm control logic và multiplexers để điều khiển chiều di chuyển của input activations

Ví dụ:



// Đánh giá //

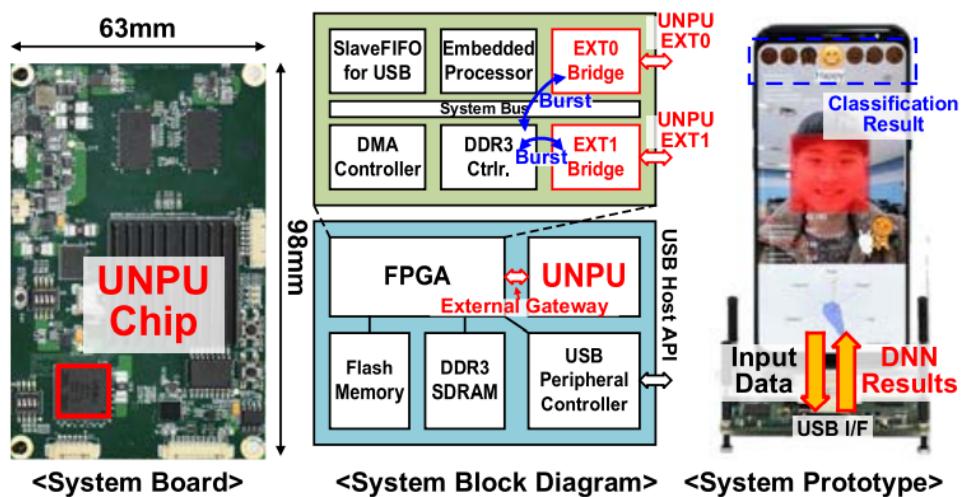
Chuẩn bị thí nghiệm

UNPU được hiện thực bằng ASIC sử dụng công nghệ 65-nm 1P8M CMOS. Diện tích của UNPU là $4 \times 4\text{mm}^2$ với mức tiêu hao năng lượng là 297mW ở 200MHz, 1.1V và 3.2mW ở 5MHz 0.63V.

Các DNN được chuẩn bị để benchmark:

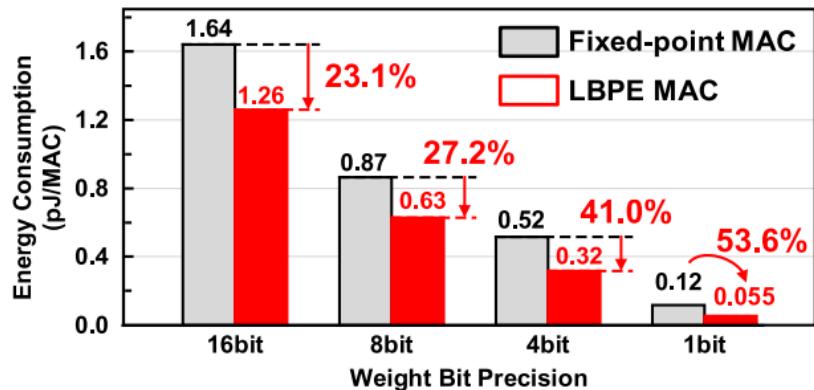
- VGG-16
- AlexNet

Hệ thống được dùng để thí nghiệm:

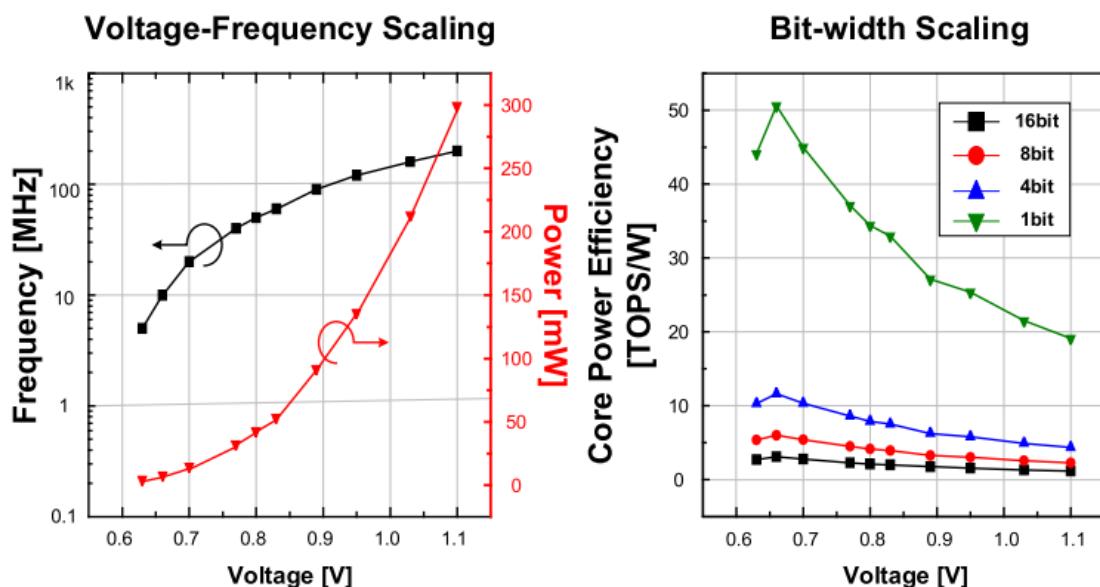


Kết quả thí nghiệm

Thí nghiệm 1: So sánh mức tiêu hao năng lượng giữa LBPE và conventional fixed-point MAC array.



Thí nghiệm 2: Kết quả của TOPS/W trong đồ thị liên hệ giữa hiệu điện thế - tần số và hiệu điện thế - giá trị bit-width của UNPU.



Thí nghiệm 3: Kết quả khi benchmark mạng VGG-16 và so sánh với:

- Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, “Eyeriss: An energyefficient reconfigurable accelerator for deep convolutional neural networks,” IEEE J. Solid-State Circuits, vol. 52, no. 1, pp. 127–138, Jan. 2017.
- B. Moons, R. Uyttterhoeven, W. Dehaene, and M. Verhelst, “Envision: A 0.26-to-10 TOPS/W subword-parallel dynamic-voltage-accuracy frequency-scalable convolutional neural network processor 28 nm FDSOI,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 246–257.

TABLE II
BENCHMARK ON VGG-16 (CONV)

8bit(F)x8bit(W)	Conv. Layers												
	1	2	3	4	5	6	7	8	9	10	11	12	13
<i>The number of Operations (10⁹)</i>	0.17	3.70	1.85	3.70	1.85	3.70	3.70	1.85	3.70	3.70	1.21	0.92	0.92
<i>Active PE (%)</i>	77	93	93	95	95	100	100	100	100	100	100	100	100
<i>External Memory Access (MB)</i>	6.31	10.3	4.16	5.25	3.86	6.20	6.20	5.98	12.0	12.0	5.40	4.68	4.68

TABLE III
VGG-16 BENCHMARK COMPARISON

	[16]	[19]	This Work	
<i>Frame Rate (fps)</i>	0.7	1.67	0.97	18.3
<i>Power Consumption (mW)</i>	236	26	6.4	297
<i>Power Efficiency (TOPS/W)</i>	0.092 (Avg.)	2.0 (Avg.)	4.71 ⁽¹⁾	1.91 ⁽²⁾
<i>Bit Precision (bit)</i>	16	N/A	8	8

(1) Core 10MHz, Link 15MHz, 0.7V

(2) Core 200MHz, Link 200MHz, 1.1V

Thí nghiệm 4: Kết quả khi benchmark mạng AlexNet và so sánh với:

- B. Moons, R. Uyttterhoeven, W. Dehaene, and M. Verhelst, “Envision: A 0.26-to-10 TOPS/W subword-parallel dynamic-voltage-accuracyfrequency-scalable convolutional neural network processor 28 nm FDSOI,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 246–257.
- D. Shin, J. Lee, J. Lee, and H.-J. Yoo, “DNPU: An 8.1 TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 240–241.
- S. Yin et al., “A 1.06-to-5.09 TOPS/W reconfigurable hybrid-neuralnetwork processor for deep learning applications,” in Proc. IEEE Symp. VLSI Circuits (VLSI-Circuits), Jun. 2017, pp. C26–C27.

TABLE IV
BENCHMARK ON ALEXNET

	Weight Precision	Conv. Layers				
		1	2	3	4	5
<i>The number of Operations (10^6)</i>	Common	211	448	299	224	149
<i>Active PE (%)</i>		77	93	100	96	96
<i>External Memory Access (MB)</i>	8bit	0.77	0.89	1.59	0.97	0.64
	1bit	0.74	0.45	0.35	0.23	0.18

TABLE V
ALEXNET BENCHMARK COMPARISON

	[19]	[20]	[22]	This Work		
<i>Frame Rate (fps)</i>	47	177	105	19.7	346	158
<i>Power Consumption (mW)</i>	44	63	290	6.1	290	7.7
<i>Power Efficiency (TOPS/W)</i>	1.42	4.2	1.27	4.3 ⁽¹⁾	1.59 ⁽²⁾	27.3 ⁽³⁾
<i>Bit Precision (bit)</i>	N/A	8	N/A	8 (Top-1 56.9%)	1 (Top-1 56.8%)	

(1) Core 10MHz, Link 15MHz, 0.7V

(2) Core 200MHz, Link 200MHz, 1.1V

(3) Core 10MHz, Link 30MHz, 0.75V

Thí nghiệm 5: So sánh hiệu năng giữa UNPU và các hardware accelerators được đề cập trong bài báo.

TABLE VI
PERFORMANCE COMPARISON TABLE

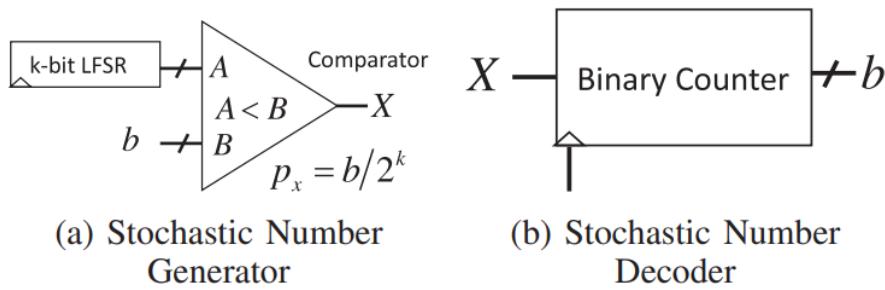
	DNPU [20]		ENVISION [19]		S.VLSI2017 [22]		BRein memory [23]		QUEST [25]		This Work	
<i>Purpose</i>	CLS, RLS, FCLs		CLS		CLS, RLS, FCLs		RLs, FCLs		CLS, RLS, FCLs		CLS, RLS, FCLs	
<i>Process [nm]</i>	65		28		65		65		40		40	
<i>Die Area [mm²]</i>	16		1.87		19.4		12		121.6		16	
<i>Supply Voltage [V]</i>	0.77 – 1.1		0.605 – 1.05		0.67 – 1.2		0.55 – 1.0		1.1		0.63 – 1.1	
<i>Maximum Frequency MHz</i>	200		200		200		400		330		200	
<i>PE Bit-precision [Bit]</i>	4, 8, 16		4, 8, 16		8, 16		1, 2		1 – 4		1 – 16	
<i>Peak Performance [GOPS]</i>	1200 (4b) 300 (16b)		408 (4b) 76 (16b)		409.6 (8b)		1,380 (1b)		7490 (1b) 1960 (4b)	3300	7372 (1b), 1382(4b), 691.2(8b), 345.6(16b)	
<i>Power Consumption [mW]</i>	50MHz 0.77V	34.6	50MHz 0.63V	7.5	10MHz 0.67V	4	100MHz 0.55V	50			5MHz 0.63V	3.2
	200MHz 1.1V	279	200MHz 1.05V	300	200MHz 1.2V	447	400MHz 1.0V	600			200MHz 1.1V	297
<i>Power Efficiency [TOPS/W]</i>	50MHz 0.77V	8.1 (4b)	50MHz 0.63V	10.0 (4b)	10MHz 0.67V	5.09 (8b)	100MHz 0.55V	6.0 (1bx1b)	2.27 (1b) 0.59 (4b)	3.08 (16b) 11.6 (4b) 50.6 (1b)		
	2.1 (16b)	100MHz 0.7V	0.53 (16b)	200 1.2V	1.06 (8b)	400MHz 1.0V	2.3 (1bx1b)					

1.3. An Efficient Hardware Implementation of Artificial Neural Network based on Stochastic Computing

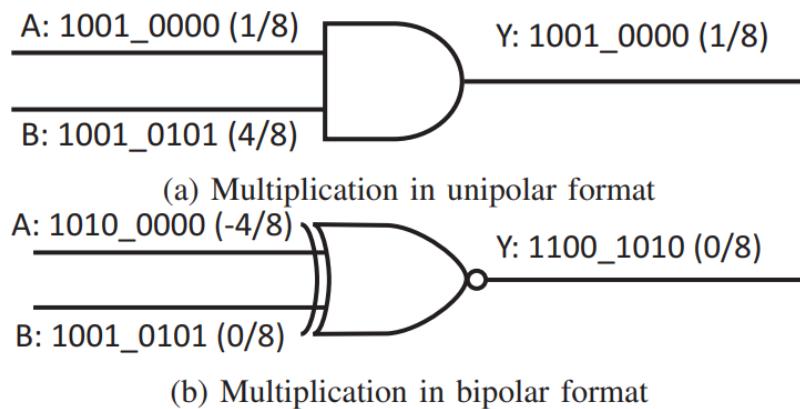
Ý tưởng: Stochastic Computing là phương thức tính toán dựa trên xác suất của các bit trong một chuỗi bit. Điểm mạnh của phương pháp này là các thành phần cần cho tính toán được giảm mạnh, do đó diện tích và năng lượng lượng giảm rất nhiều. Tuy nhiên, điểm yếu của phương pháp này là cần nhiều thời gian cho tính toán và độ chính xác tính toán cũng bị giảm xuống tương đối nhiều.

A. Thành phần tính toán của SC

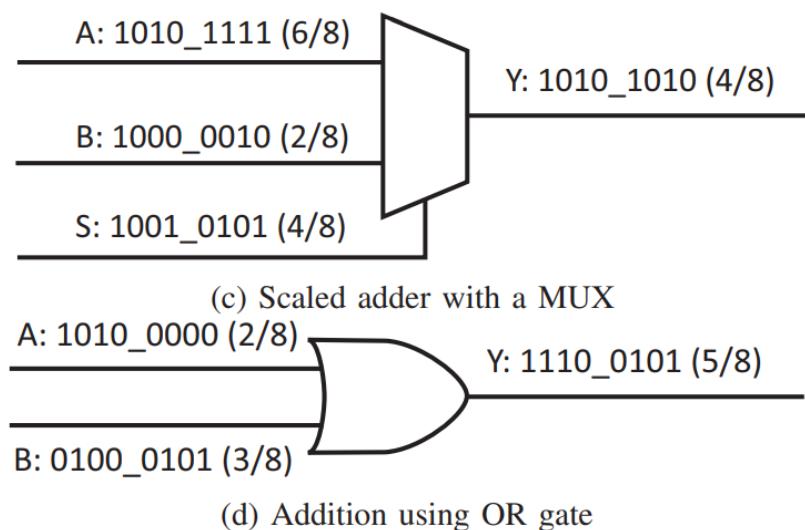
Stochastic Number converter:



Thành phần nhân:



Thành phần cộng:



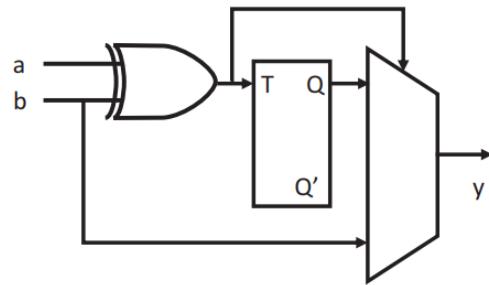
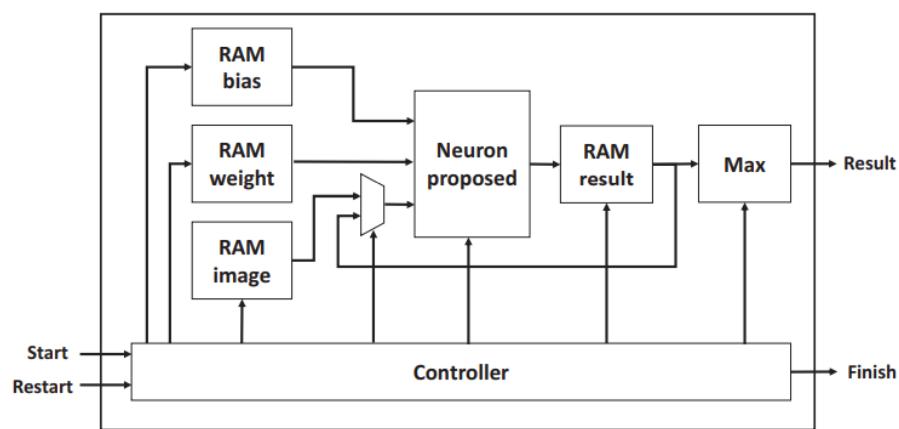


Figure 3: TFF-based stochastic adder.

B. Kiến trúc hệ thống



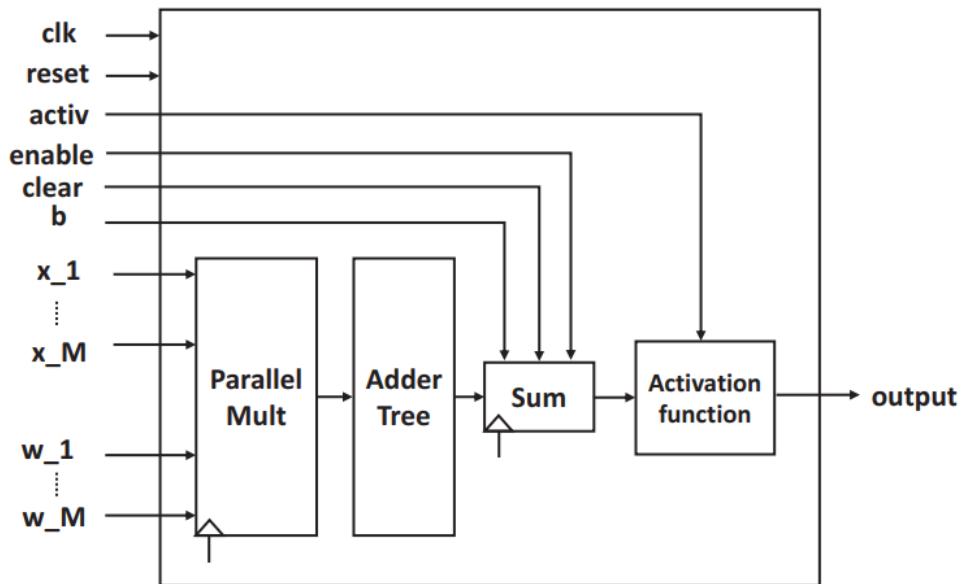
Hệ thống bao gồm:

- RAM image, RAM weight và RAM bias lưu trữ các giá trị input activation, weight và bias
- Khối Controller điều khiển toàn bộ hoạt động của hệ thống
- RAM result lưu kết quả tính toán output layer.
- Khối Max xác định kết quả phân lớp từ kết quả cuối cùng ở output layer
- Khối Neuron proposed thực hiện tác vụ tính toán chính cho DNN

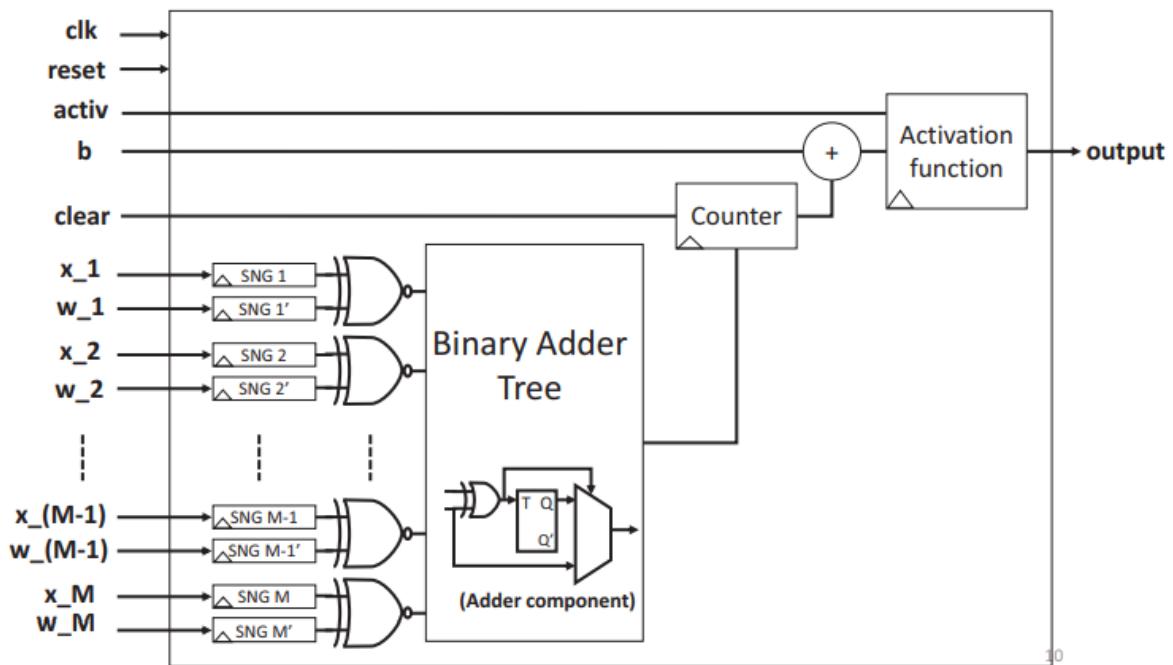
C. Neuron proposed module

Có hai mô hình được đề xuất:

- Binary neuron block



- SC neuron block



// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Hệ thống được hiện thực theo công nghệ FPGA trên board Xilinx Artix-7 FPGA

Ứng dụng dùng trong thí nghiệm: handwritten digit recognition application với dataset là MNIST.

Phần mềm dùng để làm baseline là Caffe.

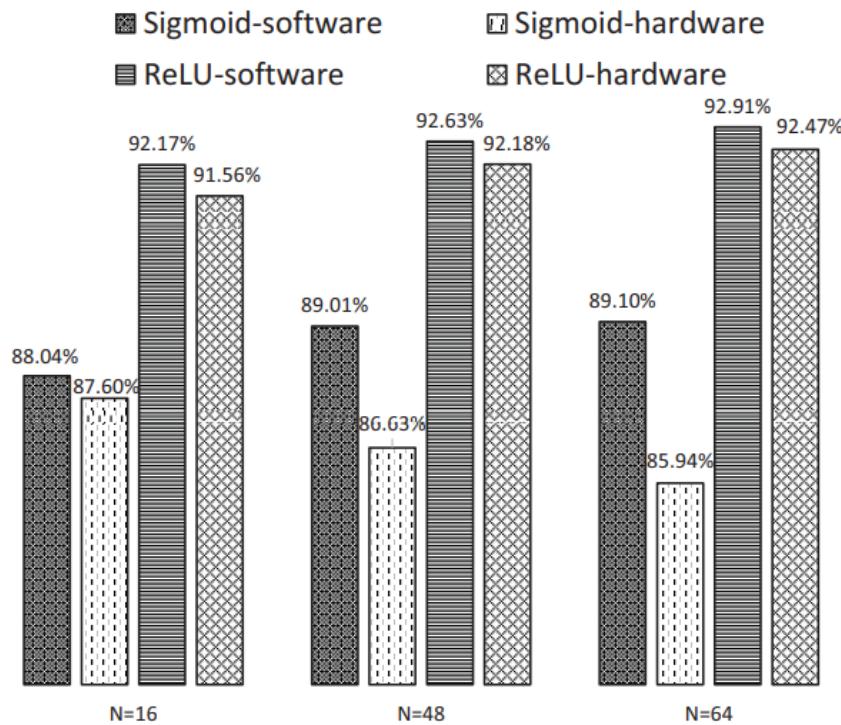
DNN được chuẩn bị để benchmark là ANN với 3 layer gồm 1 input layer 784 neuron, 1 hidden layer với N neuron và 1 output layer với 10 neuron. Activation function được dùng là Sigmoid và ReLU.

Kết quả thí nghiệm

Thí nghiệm 1: Kết quả độ chính xác của mô hình của hệ thống (fixed-point 10 bit) khi so sánh với phần mềm ở các giá trị N khác nhau.

Table II: Comparison of classification accuracies between the proposed design and the Caffe's implementation

	N	16	48	64
<i>Sigmoid</i>	Proposed design	87.60%	86.63%	85.94%
	Software simulation	88.04%	89.01%	89.10%
	Accuracy loss	0.44%	2.38%	3.16%
<i>ReLU</i>	Proposed design	91.56%	92.18%	92.47%
	Software simulation	92.17%	92.63%	92.91%
	Accuracy loss	0.61%	0.45%	0.44%



Thí nghiệm 2: So sánh MSE của bộ nhân và cộng SC của hệ thống (8-bit fixed point) với công trình sau:

V. T. Lee, A. Alaghi, J. P. Hayes, V. Sathe, and L. Ceze, "Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing," in Design, Automation Test in Europe Conference Exhibition (DATE), 2017, March 2017, pp. 13–18.

Table III: Comparison between the SC adder and SC multiplier's MSE with other works

	Design in [13]	Proposed design
SC format	8-bit unipolar	8-bit unipolar
Multiplier	2.57×10^{-4}	7.43×10^{-6}
Adder	1.91×10^{-6}	1.89×10^{-6}
		1.32×10^{-5}

Thí nghiệm 3: So sánh MSE và thời gian thực thi của hai cách hiện thực khối Neuron proposed.

Table IV: MSE and execution time comparison between binary radix neuron block and SC neuron block

	Bitwidth	Binary radix	SC
MSE	8	2.02×10^{-3}	7.05×10^{-2}
	10	1.47×10^{-4}	9.02×10^{-4}
Execution time (clock cycle)	8	2	257
	10	2	1025

Thí nghiệm 4: Kết quả khi hiện thực hệ thống bằng FPGA với các giá trị N = 48 và sử dụng độ chính xác là fixed point 10-bit. Và kết quả so sánh khối Proposed Neuron theo cả hai cách hiện thực.

Table V: Performance report of FPGA implementation

Performance metric	Result
Frequency	158 MHz
Clock cycle	4888
Execution time	$30.93 \mu\text{s}$
Total power	0.202 W

Table VI: Hardware resources utilization report of FPGA implementation

Cost	Design	FPGA resources	Utilization
LUTs	1659	133800	1.24%
Register	1020	267600	0.38%
Mux	0	100350	0%
Slice	578	33450	1.72%
DSP	0	740	0%
BRAM	0.5	365	0.14%
IO	8	400	2.0%

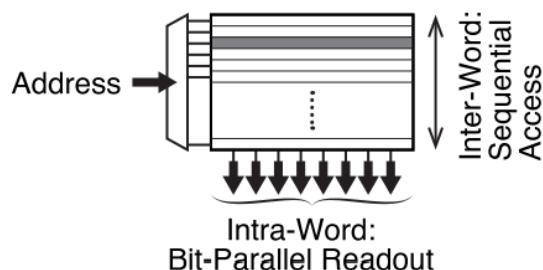
Table VII: FPGA implementation results of Binary neuron block and SC neuron block

	Binary neuron block	SC neuron block
Parallel inputs	16	16
Frequency	250 MHz	286 MHz
LUTs	1268	416
Register	23	299
IO	277	277
Power consumption	0.045W	0.039W

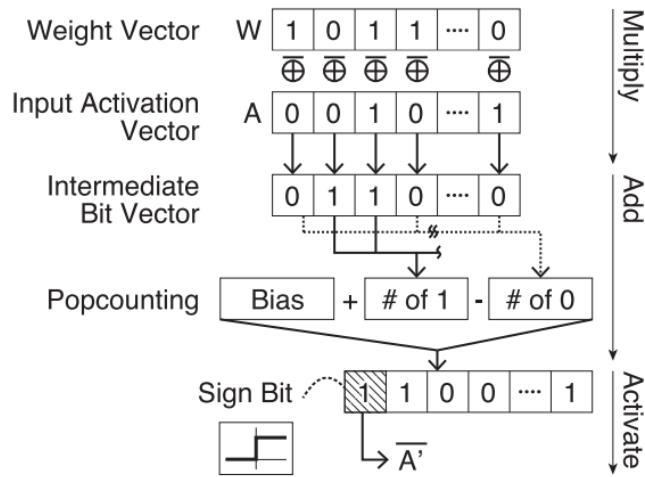
1.4. BREIN Memory: A Single-Chip Binary/Ternary Reconfigurable in-Memory Deep Neural Network Accelerator Achieving 1.4 TOPS at 0.6 W

Ý tưởng:

- Cách tiếp cận là single-chip in-memory, nghĩa là tất cả thông số của DNN được lưu trữ ở on-chip SRAMs và tất cả tính toán được thực thi bằng logic unit ở gần SRAMs.
- Key concepts:
 - Tích hợp các mạch xử lý hài hòa với bản chất của SRAM
 - Phương pháp nén tham số của DNN để đủ lưu trữ ở on-chip SRAM
 - Khai thác tối đa song song hóa intra-word và che giấu hành vi tuần tự của truy xuất inter-word

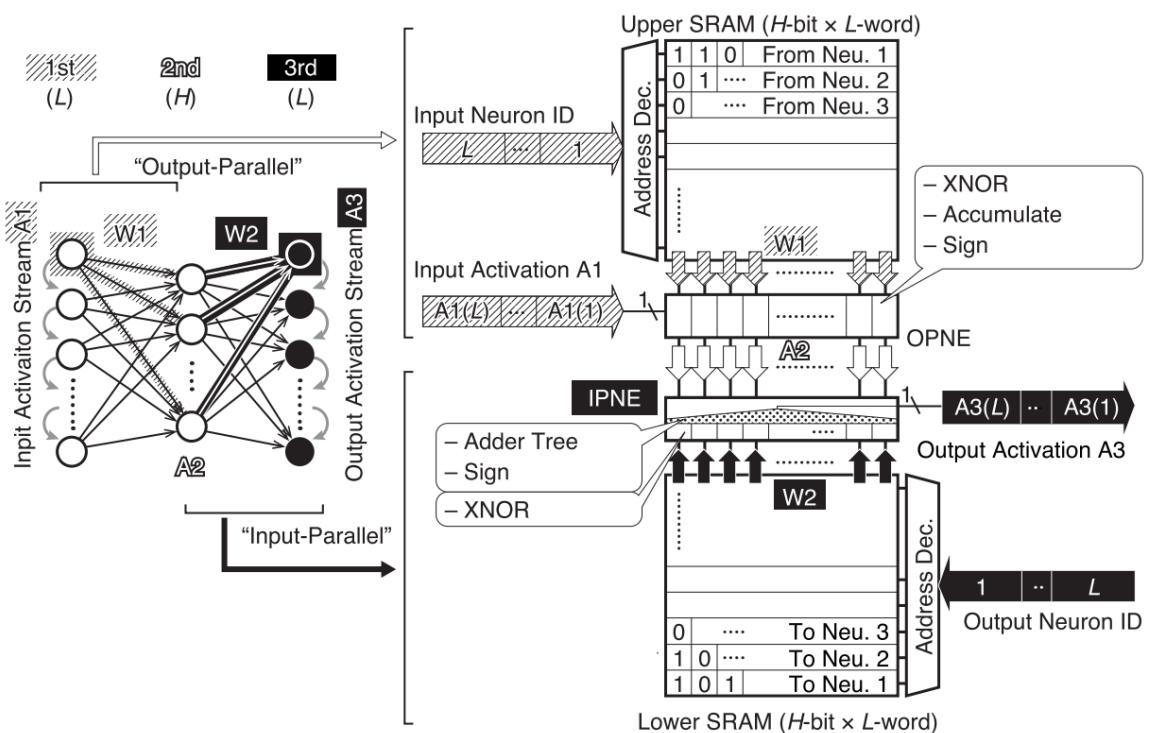


Quá trình tính toán trong một binary DNN



A. Processing-in-memory (PIM)

Khối Processing-in-memory (PIM) cho phép chứa 3-layer binary neural network với kích thước chiều rộng lần lượt là L-H-L.

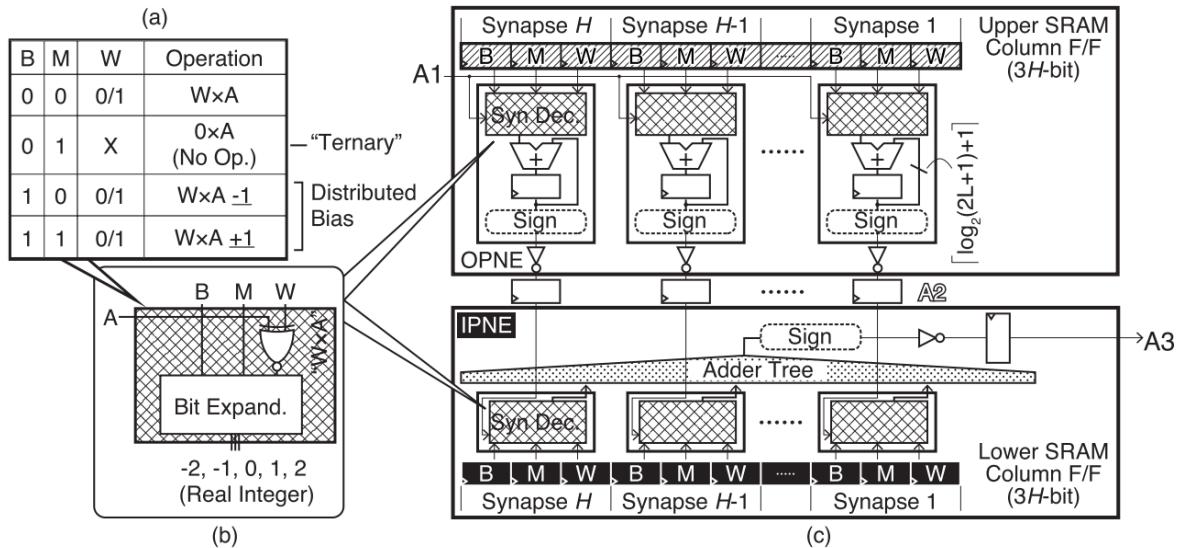


PIM bao gồm: 2 mảng H-bit L-word SRAM với output-parallel neural engine (OPNE) và input-parallel neural engine (IPNE) tương ứng.

- ONPE tuần tự nhận các giá trị input activation vào bit stream A1 và thực hiện phép XNOR với H-bit weight ở W1 (được đọc song song từ Upper SRAM). Sau đó cộng tích lũy để tạo ra kết quả H-bit output activation L chu kỳ.
- Kết quả H-bit output activation được lưu trữ ở thanh ghi A2 ở giữa ONPE và IPNE. Kết quả này trở thành input activation được đưa vào IPNE và thực hiện phép XNOR với H-bit weight ở W2 (được đọc

song song từ Lower SRAM). Sau đó kết quả được cộng tích lũy và thông qua adder tree để tổng hợp lại tất cả tạo ra kết quả cuối cùng là output activation của layer 3 và đưa ra ngoài ở bit stream A3.

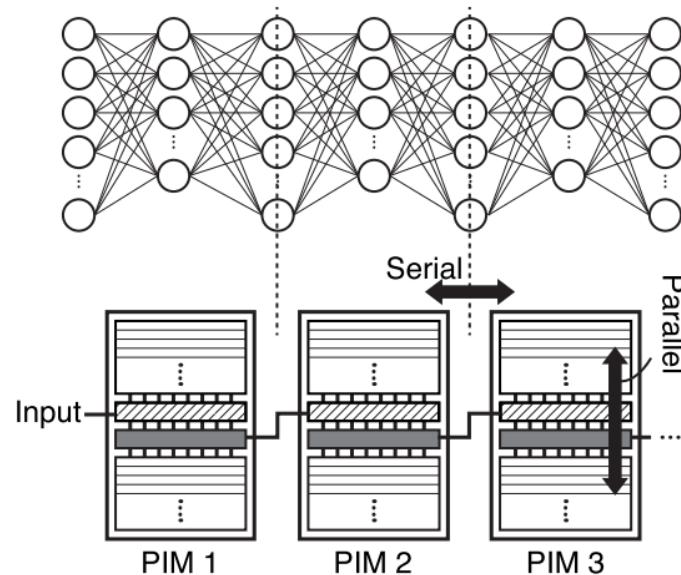
B. Synapse decoder (SynDec) với Ternary Neural Network và Bias Extension



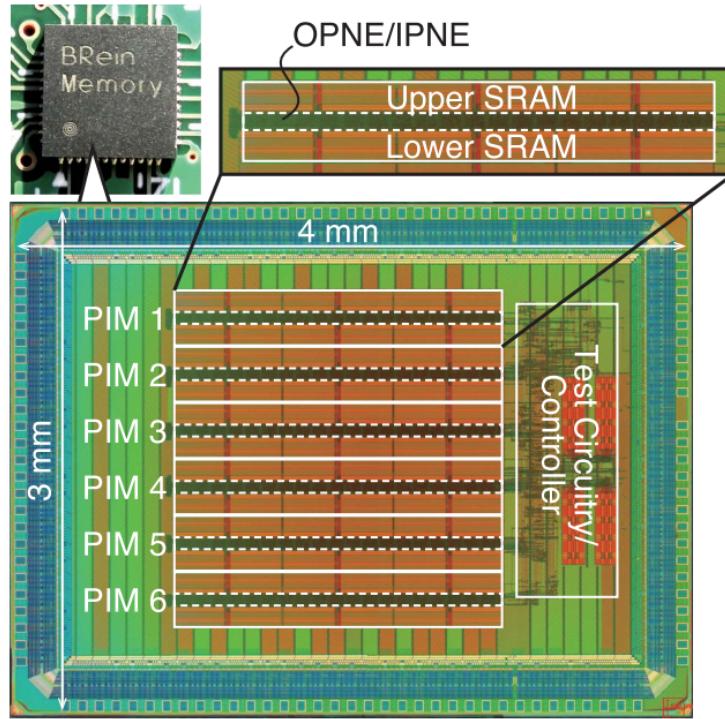
ONPE là tập hợp bởi mảng gồm H SynDec, theo đó là bộ cộng tích lũy. IPNE cũng bao gồm tập hợp SynDec tương tự, điểm khác nhau duy nhất là IPNE có thêm Adder Tree để tạo ra kết quả cuối cùng.

C. Mô hình hệ thống

Mô hình kết nối liên tiếp các PIM



BRain chip



// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

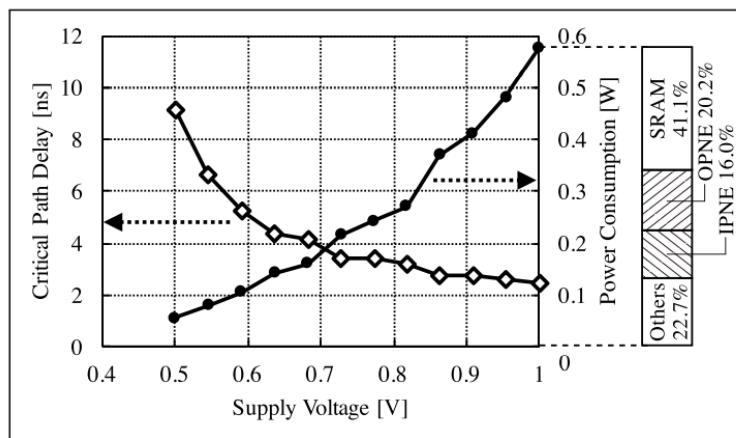
Hệ thống hiện thực theo ASIC trên công nghệ TSMC 65-nm với SRAM có kích thước L và H lần lượt là 484 và 144. Chip làm việc ở tần số 400MHz đạt 1.4 TOPs ở 0.6W năng lượng tiêu hao.

Tập dữ liệu được sử dụng được tạo ra từ MNIST bằng cách thay đổi kích thước ảnh thành 22×22 và đưa hình ảnh dạng trắng đen.

DNN được dùng để kiểm thử là Fully-connected Binary DNN để nhận diện chữ số viết tay. DNN này bao gồm 13 fully-connected layer có kích thước lần lượt là 484, 144, ..., 484, 144 và 10.

Kết quả thí nghiệm

Thí nghiệm 1: Đo đạt mức năng lượng tiêu hao và critical path delay (có thể hiểu đây là đại lượng này dùng để xác định tần số làm việc lớn nhất mà hệ thống có thể làm việc) ở các hiệu điện thế cung cấp khác nhau.



Thí nghiệm 2: So sánh hệ thống với CPU, GPU và FPGA thực thi trên cùng một binary DNN được miêu tả ở trên.

TABLE I
COMPARISON ON THE SAME 13-LAYER BINARY DNN (MEASURED)

	Process Tech.	Clock Freq. [GHz]	Exec. Time [sec]	Power [W]	Energy Consum. [J]	Effective Perf. [GOPS]	Energy Efficiency [GOPS/W]	Ratio			
								Clock Freq.	Exec. Time	Power	Energy Consum.
CPU	14nm	2.2	124	155	19.2k	12.4	0.08	5.5	102	266	27.1k
GPU	28nm	1.0	13.8	152	2,098	111.3	0.73	2.5	11	261	2,966
FPGA	28nm	0.2	53	11	583	29.0	2.64	0.5	44	19	825
This Work	65nm	0.4	1.22	0.6	0.73	1264.4	2172.42	1	1	1	1

Measured using 1 million transactions of the binary DNN handwritten digit recognition. We assumed that 1 MAC is composed of 1 ADD and 1 MUL; thus, 1 MAC is equivalent to 2 OPs.

Thí nghiệm 3: So sánh thông số hệ thống với các công trình trước đó:

- Y. H. Chen, T. Krishna, J. Emer, and V. Sze, “14.5 Eyeriss: An energyefficient reconfigurable accelerator for deep convolutional neural networks,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Jan. 2016, pp. 262–263
- D. Shin, J. Lee, J. Lee, and H.-J. Yoo, “14.2 DNPU: An 8.1TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 240–241.
- B. Moons, R. Uyttterhoeven, W. Dehaene, and M. Verhelst, “14.5 envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracyfrequency-scalable convolutional neural network processor in 28 nm FDSOI,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 246–247

TABLE II
COMPARISON AND CHIP SUMMARY

	ISSCC 2016 [5]	ISSCC 2017 [8]	ISSCC 2017 [9]	This Work
Technology	65nm LP	65nm	28nm FD-SOI	65nm GP
Target	CNN	CNN + RNN	CNN	Versatile DNN ¹
W/A Precision [bits]	16	4–16 (Conv), 4–7 (FC)	4–16	Bin/Ter
Operating Frequency [MHz]	200	50–200	200 (Typ.)	100–400
Voltage [V]	1.0	0.77–1.1	0.6–1.1	0.55–1.0
Power [W]	0.3	0.03–0.28	0.075–0.3	0.05–0.6
Core Area [mm ²]	12.3	7.4	0.95	3.9
Peak Performance ² [TOPS]	0.07	1.25	0.41	1.38
Energy Efficiency [TOPS/W]	0.2	1.0–8.1	0.26–10 [0.05–1.86] ³	2.3–6.0
Area Efficiency [TOPS/mm ²]	0.006	0.045–0.169	0.108–0.431 [0.02–0.08] ³	0.089–0.365
Dataset (Network) for Evaluation	(AlexNet)	(AlexNet)	Hierarchical Face Recognition (AlexNet, VGG-16)	Resized MNIST (Binary/Ternary MLP)

¹ Although our architecture is not specially optimized for CNNs, it can still house CNNs by mapping convolutional layers as fully connected layers, as mentioned in Section VI-C.

² Note that the definition of OP differs chip-wise. We estimated the performance by assuming that 1 MAC for 1 synapse is composed of 2 operations (addition and multiplication).

³ The efficiency assuming the system is integrated on a 65nm technology, that is calculated using the scaling law (Power, Area \propto (Gate Length)²).

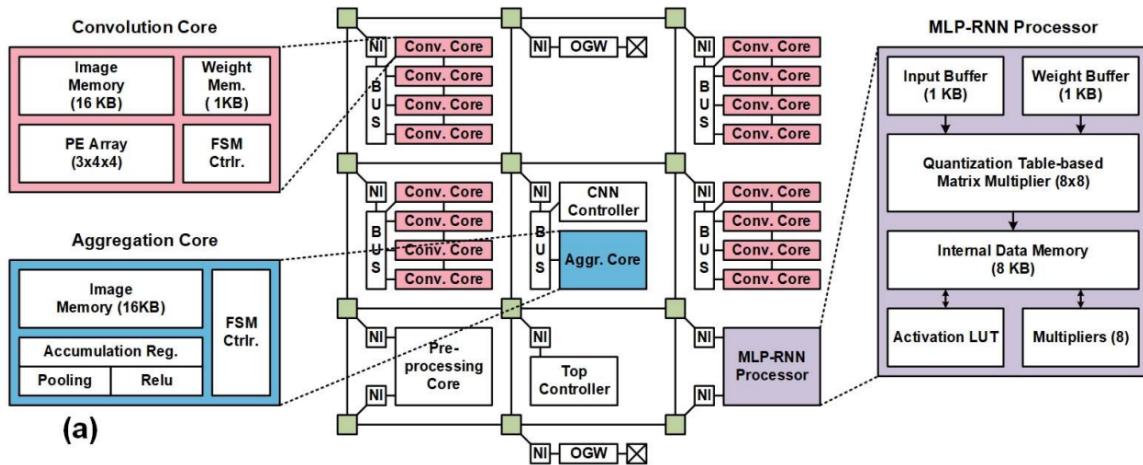
1.5. DNPU: An Energy-Efficient Deep-Learning Processor with Heterogeneous MultiCore Architecture

Ý tưởng:

- Đặc điểm bất đồng nhất

- CNN: thực hiện một lượng lớn tác vụ MAC trên một lượng nhỏ các tham số (input activation và weight kernel)
- ANN và RNN: thực hiện một lượng nhỏ tác vụ MAC trên một lượng lớn các tham số (input activation và weight kernel)
- Đa dạng về các cài đặt: Trên cùng một loại layer của DNN, các cài đặt rất đa dạng (stride, kernel, số input channel, output channel,...). Do đó, cần có một phương pháp để chia sẻ workload cho phù hợp.
- Số lượng tham số và tính toán cực kỳ lớn

Hệ thống vận dụng kiến trúc bất đồng nhất. Kiến trúc như sau:



Hệ thống bao gồm: CNN processor, MLP-RNN processor và top RISC controller.

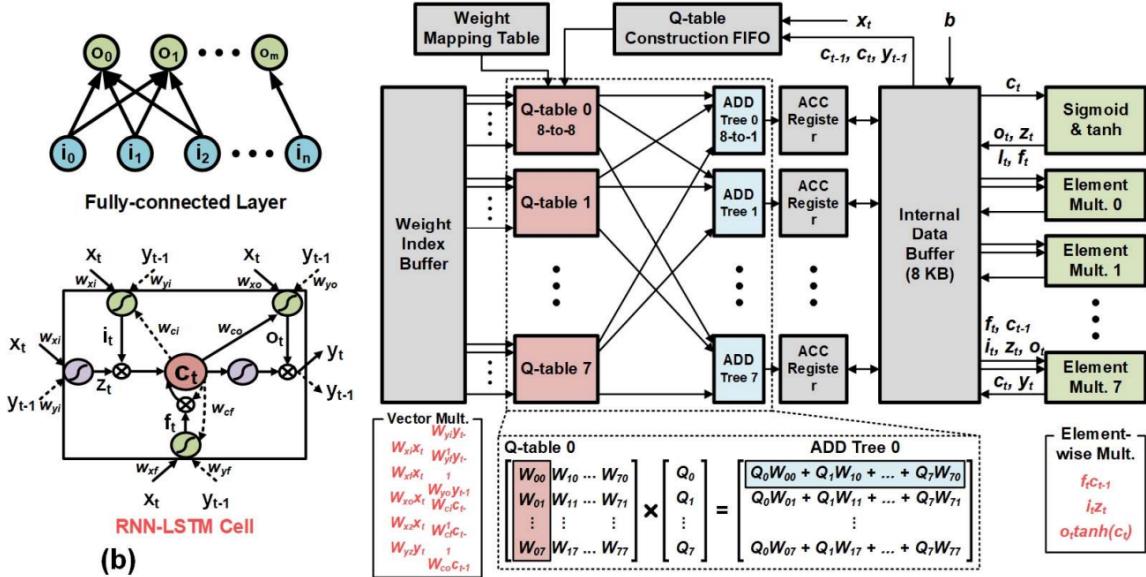
- CNN processor được thiết kế để khai thác tối đa việc tái sử dụng trong các tác vụ ở convolution layer và có số lượng lớn của các processing element để thực thi tác vụ MAC.
- MLP-RNN processor có số lượng processing element ít hơn, tuy nhiên nó được tối ưu hóa để giảm số lượng truy cập vào bộ nhớ ở bên ngoài chip để lấy các giá trị tham số.

A. CNN processor

CNN processor là tổng hợp của 16 convolution core và 1 aggregation core.

Mỗi 4 convolution core được kết nối serial với nhau và convolution core cuối cùng của chuỗi thì kết nối với aggregation core. Các tổng bán phần từ các tác vụ tính toán ở các convolution core sẽ được gửi tới các core tiếp theo và cộng tích lũy lại.

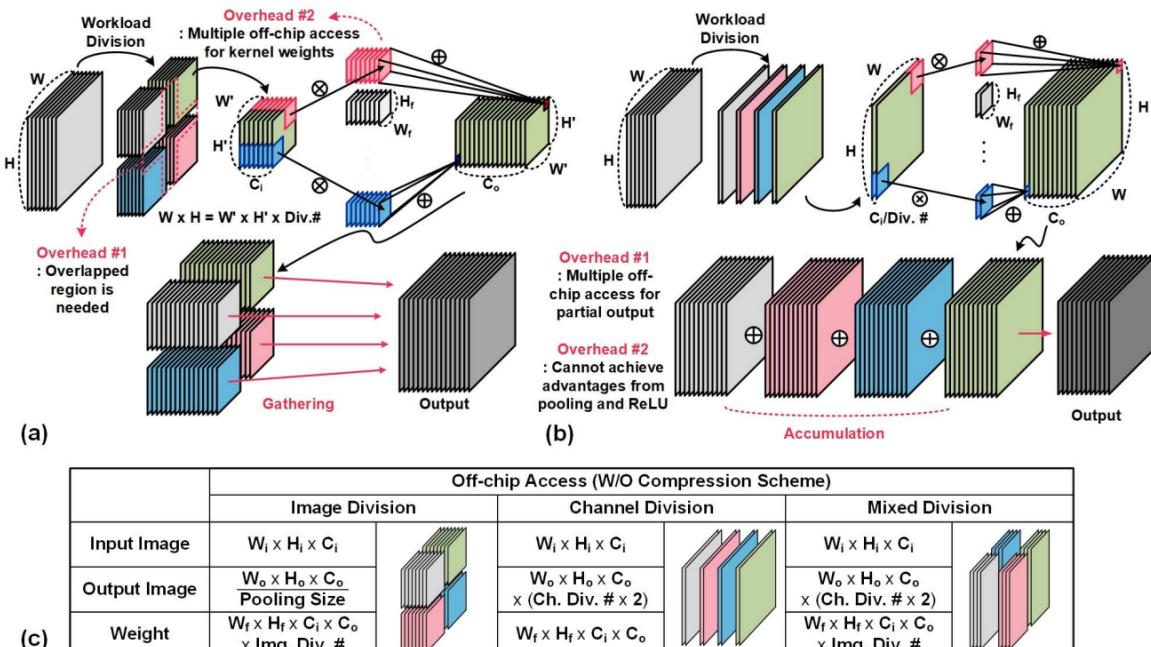
B. MLP-RNN processor



MLP-RNN processor thực hiện tác vụ nhân ma trận bằng việc sử dụng 8 16-entry quantization tables (Q-tables), 8 16b fixed-point multipliers (nhiệm vụ của bộ nhân này là cập nhật giá trị của Q-tables và thực hiện các tác vụ element-wise) và các lookup table phục vụ cho activation function (sigmoid, tanh, ...)

C. Mixed Workload Division

Các vấn đề liên quan:

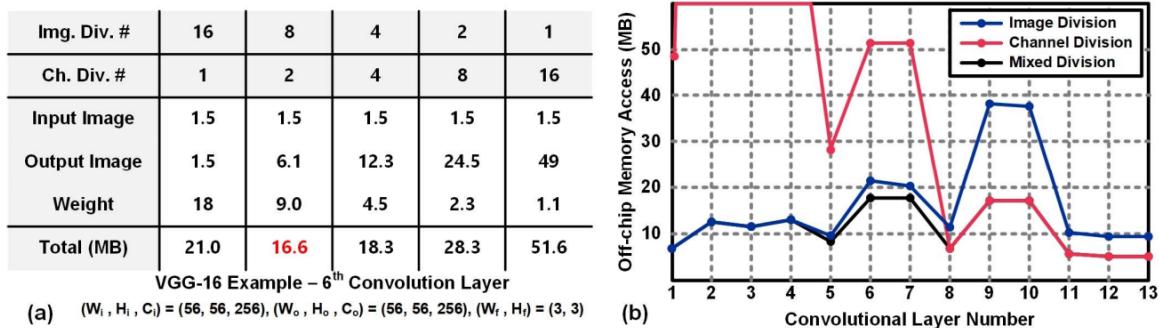


- Phương pháp Image Division: Các nhóm input feature map được phân chia để xử lý độc lập nhau. Tuy nhiên giá trị weight kernel phải được tải đi tải lại nhiều lần cho mỗi nhóm input feature map, dẫn đến số lượng truy cập vào bộ nhớ để lấy kernel weight gia tăng.

Phương pháp này phù hợp trong trường hợp kernel weight có kích thước tương đối nhỏ so với input feature map.

- Phương pháp Channel Division: Các channel của input feature map được phân chia để xử lý độc lập. Các giá trị weight kernel không cần phải tải lên nhiều lần, tuy nhiên các giá trị tính toán trung gian (các tổng bán phần) cần được tổng hợp lại để tạo ra kết quả cuối cùng, do đó cần phải truy xuất vào bộ nhớ ngoài để lưu trữ và cộng tích lũy.

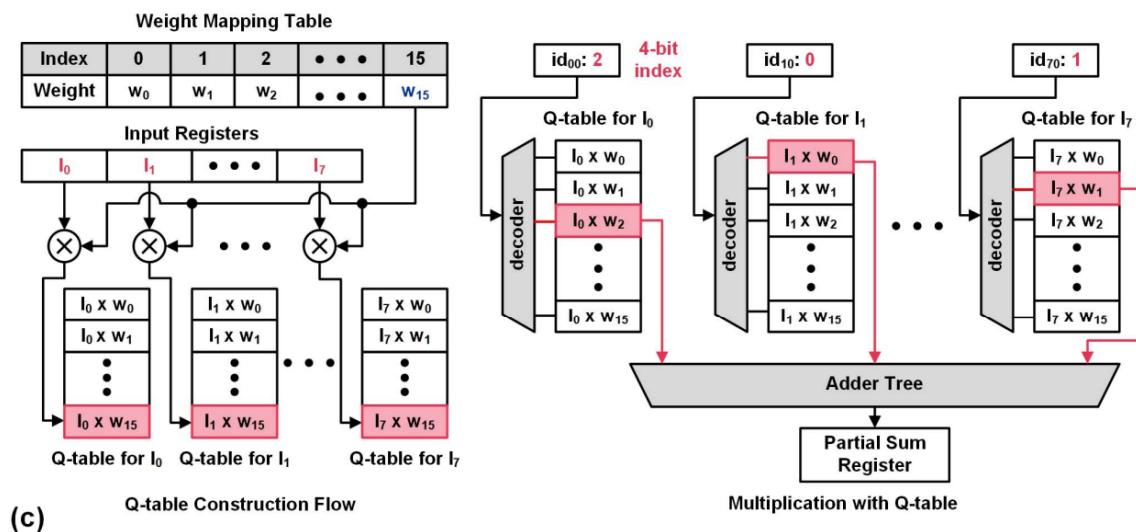
Phương pháp này phù hợp trong trường hợp input feature map có kích thước tương đối nhỏ so với kernel weight.



- Phương pháp Mixed Division: Kết hợp giữa hai phương pháp trên lại, vừa phân chia input feature map thành các nhóm, vừa phân chia các channel trong nhóm này.

D. Quantization Table-based Multiplier

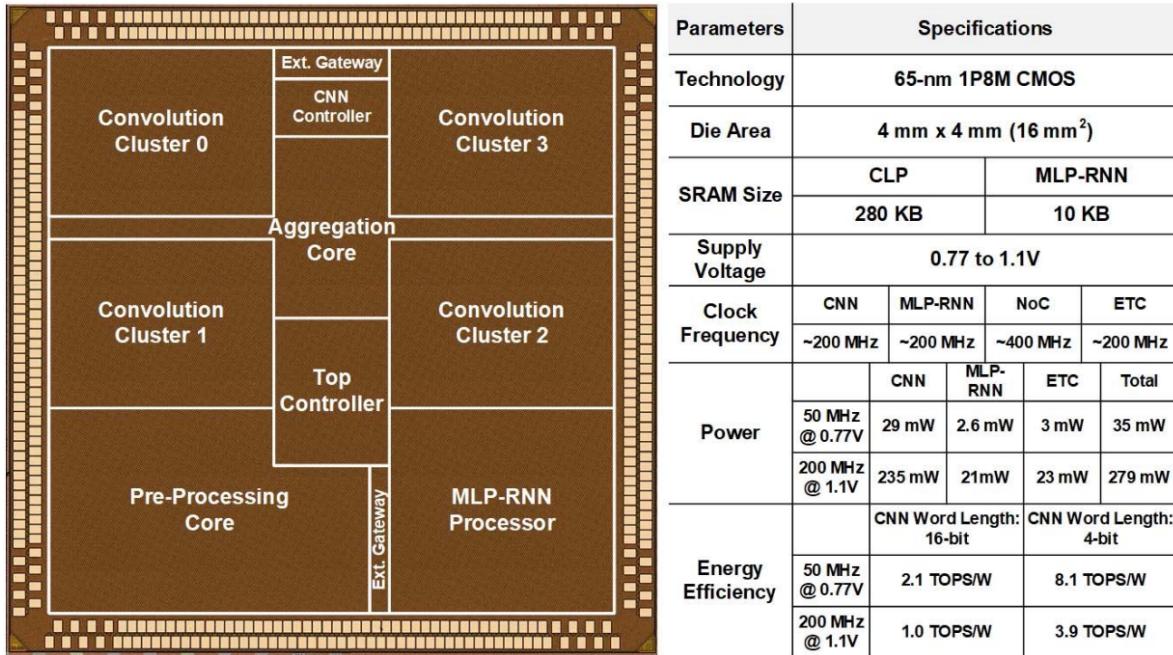
Ý tưởng: Khi sử dụng việc làm tròn weight, phép nhân có thể được thay thế bằng table lookup.



// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

DNPU được hiện thực bằng ASIC với công nghệ 65-nm one-poly eight-metal CMOS.



DNN được dùng trong so sánh:

- ImageNet
- Flickr
- VGG-16
- RNN gồm 3 layer trong đó có 1 LSTM layer

Kết quả thí nghiệm

Thí nghiệm 1: Kết quả khi làm tròn weight ở các layer khác nhau và so sánh kết quả khi tổng hợp trên ASIC của việc dùng Q-table và 16-bit fixed point.

Table (a) compares FC Layer Weight Quantization and LSTM Layer Weight Quantization across three bit widths (32bits, 4bits, 2bits) for two tests: ImageNet Classification Test and Flickr 8K Image Captioning Test. The table shows Top-1 Error and Top-5 Error for FC, and Perplexity and BLEU scores for LSTM.

FC Layer Weight Quantization			LSTM Layer Weight Quantization		
	Top-1 Error	Top-5 Error		Perplexity (Lower is better)	BLEU (Higher is better)
32bits	42.78%	19.73%	32bits	15.680	55.7 / 37.4 / 24 / 15.7
4bits	42.79%	19.73%	4bits	15.829	56.7 / 38 / 24.4 / 15.7
2bits	44.77%	22.33%	2bits	19.298	58.4 / 38.6 / 24 / 14.8

Table (b) compares 16-bit Fixed-point and Q-table for Area, Power, and Latency.

	16-bit Fixed-point	Q-table
Area	1890	1380 (per 1 Mult.)
Power	0.32mW	0.068mW (per 1 Mult.)
Latency (Unconstrained)	7.1ns	0.49ns

Simulation results on 65nm @ 200MHz, 1.2V

Thí nghiệm 2: So sánh hiệu năng hệ thống với các công trình trước đó:

- B. Moons and M. Verhelst, “A 0.3–2.6 TOPS/W precision-scalable processor for realtime large-scale ConvNets,” IEEE Symposium on VLSI Circuits (VLSI-Circuits), 2016; <https://ieeexplore.ieee.org/document/7573525/>.
- B. Moons et al., “Envision: A 0.26-to-10 TOPS/W subword-parallel dynamic-voltageaccuracy-frequency-scalable Convolutional Neural Network processor in 28nm FDSOI,” IEEE International

Solid-State Circuits Conference, 2017; <https://ieeexplore.ieee.org/document/7870353/>.

- G. Desoli et al., “A 2.9 TOPS/W deep convolutional neural network SoC in FD-SOI 28nm for intelligent embedded systems,” IEEE International Solid-State Circuits Conference (ISSCC), 2017; <https://ieeexplore.ieee.org/document/7870349/>.
- S. Han et al., “EIE: efficient inference engine on compressed deep neural network,” ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), 2016; <https://ieeexplore.ieee.org/document/7551397/>.
- N. P. Jouppi et al., “In-Datacenter Performance Analysis of a Tensor Processing Unit,” ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA), 2017; <https://ieeexplore.ieee.org/document/8192463/>.

Table 1. Performance comparison with previous DNN SoCs.

	Tech. (nm)	Clock Freq. (MHz)	Supply Voltage (V)	CNN	MLP, RNN	Peak Perform. (GOPS)	Power (mW)	Energy Efficiency (TOPS/W)
SoVC 2016 [6]	40	12 – 204	0.55 – 1.1	o	x	102	-	2.6
							@ 12 MHz, 4-bit	
ISSCC 2017 [7]	28	~ 200	0.65 – 1.1	o	x	76	7.6	10
							@ 50 MHz, 4-bit	
ISSCC 2017 [8]	28	200 – 1175	0.575 – 1.1	o	x	676	39	2.93
							@ 200 MHz, 8-bit	
ISCA 2016 [9]	28	1200	-	o	o	400	2360	0.17
							@ 1200 MHz, 4-bit	
ISCA 2017 [10]	28	700	-	o	o	92000	40000	2.3
							@ 700 MHz, 8-bit	
Proposed	65	50 – 200	0.77 – 1.1	o	o	1214	34.6	8.1
							@ 50 MHz, 4-bit	

Thí nghiệm 3: Kết quả hiệu năng hệ thống khi benchmark với VGG-16 và RNN được trình bày ở phần trước đó.

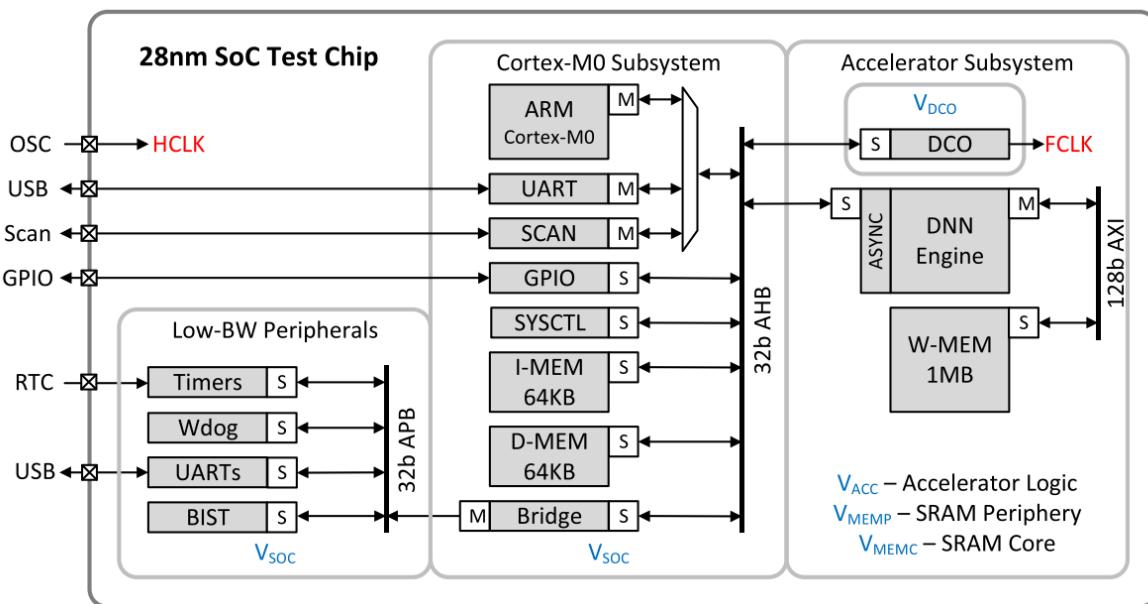
Table 2. Performance analysis on VGG-16 and RNN.

	Convolutional Layer								MLP			RNN			
	1	2	3	4	...	10	11	12	13	1	2	3	Input	LSTM	Out.
# of Input Channels	3	64	64	128	...	512	512	512	512	25,0 88	4096	4096	1000	512 LSTM cells	512
# of Output Channels	64	64	128	128		512	512	512	512	4096	4096	1000	512	512 LSTM cells	2538
Image Height & Width	224	224	112	112		28	14	14	14	1	1	1	1		1
# of Filter Weights (M)	0.002	0.04	0.07	0.14		2.2	2.2	2.2	2.2	98	16	4	0.5	2	1.2
# of MAC (Gops)	0.081	1.7	0.86	1.7		1.7	0.43	0.43	0.43	0.096	0.016	0.004	0.0005	0.002	0.0012
Throughput (Inference/s)	1022	78	165	80		79	320	320	313	16	98	390	3080	704	1239
Power (mW)	22	31	33	32		31	31	31	31	2.9	2.9	2.9	3.0	2.9	3.0
Efficiency (TOPS/W)	7.5	8.7	8.6	8.6		8.8	8.9	8.9	8.7	1.0	1.1	1.0	1.0	0.9	1.0

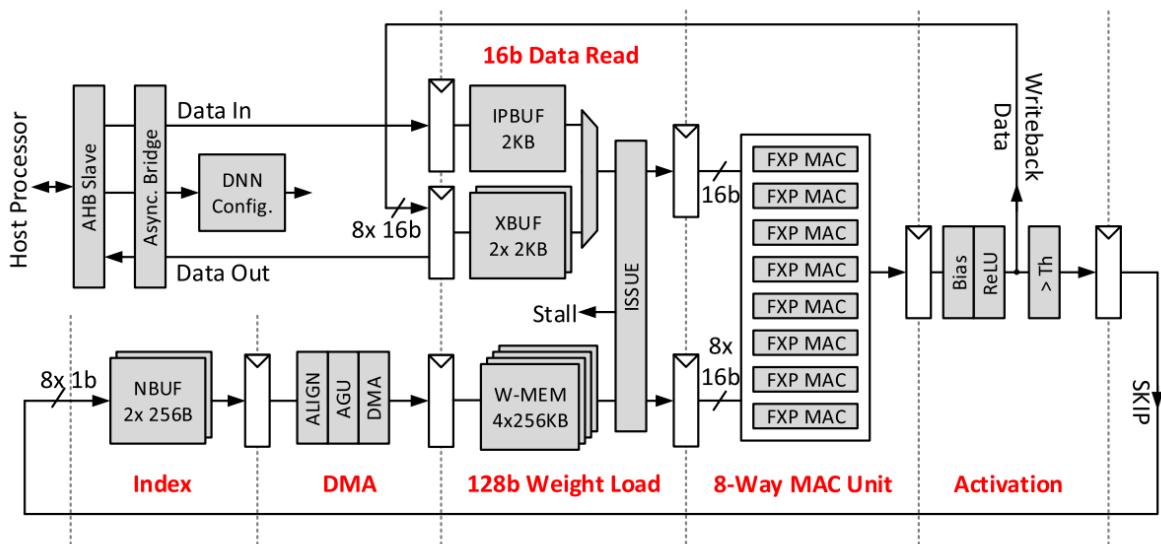
*Operating Conditions: Convolution Processor (0.77V, 50MHz), MLP-RNN Processor (0.77V, 25MHz)

1.6. DNN Engine: A 28-nm Timing-Error Tolerant Sparse Deep Neural Network Processor for IoT Applications

Kiến trúc tổng quát



A. DNN Engine



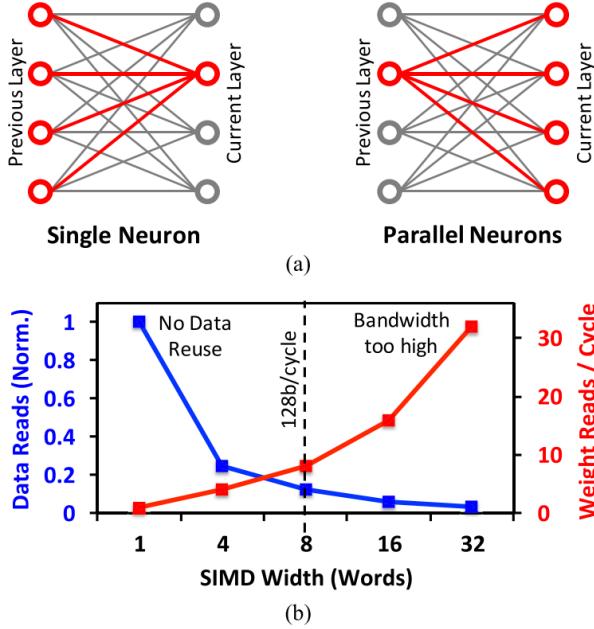
DNN engine là một 5-stage SIMD-style sparse programmable matrix-vector machine cho việc xử lý DNN. Kiến trúc của engine dựa trên 8-way SIMD MAC data path.

- Mỗi lane của data path xử lý một đơn vị neuron cho tới khi hoàn thành tính toán. Mỗi lane đọc dữ liệu input activation từ IPBUF khi xử lý input layer hoặc từ XBUF SRAM khi xử lý hidden layer.
- Một vector bao gồm 8 weight được tải lên từ W-MEM SRAM.
- Sau khi tính toán xong ở neuron, giá trị bias được thêm vào ở thanh ghi tích lũy, kích hoạt activation function và ghi 8 output activation trở lại XBUF.

B. Data reuse

Có hai cách để sắp xếp tính toán cho FC layer vào kiến trúc N-way parallel SIMD.

- Tính toán một neuron ở một layer hiện thời ở một thời điểm
- Tính toán N neuron song song với nhau và đọc một giá trị input activation từ SRAM ở mỗi chu kỳ



C. Sparsity

Ý tưởng: Các giá trị bubbles (0 hoặc gần bằng 0) được loại bỏ bằng cách tự động trượt qua các toán tử zero ở activation stage trước khi writeback. Nghĩa là sau khi đi qua hàm ReLU được áp dụng vào tổng cộng tích lũy, một bộ so sánh được dùng để so sánh giá trị với giá trị ngưỡng (ở mỗi layer sẽ có một ngưỡng khác nhau). Các giá trị nhỏ hơn ngưỡng sẽ tạo ra tín hiệu điều khiển SKIP và ghi vào XBUF. Ở layer tiếp theo, việc tính toán của neuron bị đánh dấu SKIP bị bỏ qua.

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Hệ thống hiện thực dưới dạng SoC tích hợp DNN Engine được hiện thực theo ASIC với công nghệ 28nm TSMC CMOS

TABLE I
TEST CHIP SUMMARY

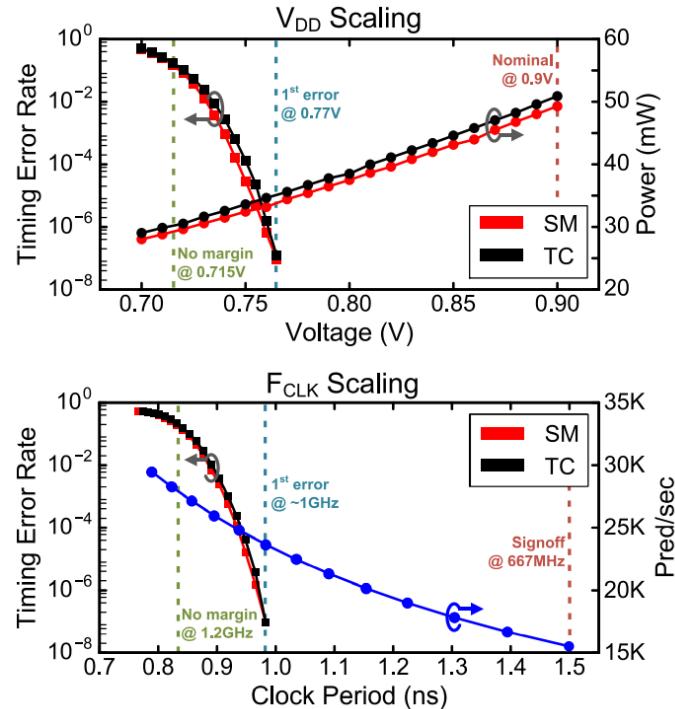
Process Tech.	TSMC 28nm HPC 1P10M
Die Size	2.4mm x 2.4mm
Total SRAM	128KB (SoC), 1MB (WMEM), 6.5KB (DNN)
Total Flip-Flops	8460 (896/8460 RZFF)
ML Model	Fully-Connected Deep Neural Network
Data Types	8/16-bit Weights, 16-bit Act., 32bit Acc.
Model Support	Layers: 0-8, Nodes: 8-1024
Error Tolerance	$>10^{-1}$ @ 98.36% Accuracy
V _{DD}	0.9V (nom.) / 0.6 – 1.1 V (operational)
F _{MAX}	667MHz @ 0.9V (WC sign-off) 1.2GHz @ 0.9V (w/Razor DFS)
Total Power	33.7mW @ 667MHz/0.9V/8b (WC sign-off) 20.3mW @ 667MHz/0.715V/8b (DVS) 63.5mW @ 1.2GHz/0.9V/8b (DFS)
Leakage	3.03mW @ 0.9V

Tập dữ liệu được sử dụng là MNIST.

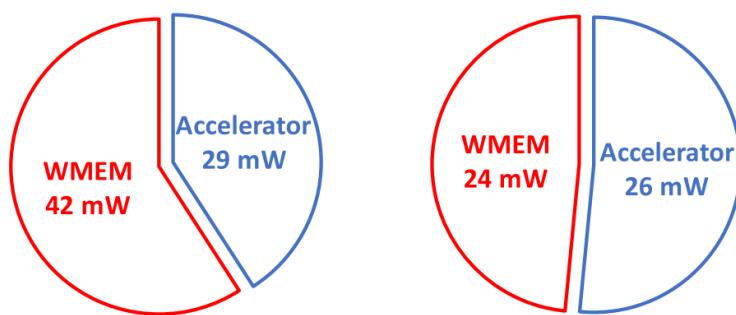
DNN dùng là ANN gồm 5 loại với từ 256 node/layer xuống 16 với 3 hidden layer

Kết quả thí nghiệm

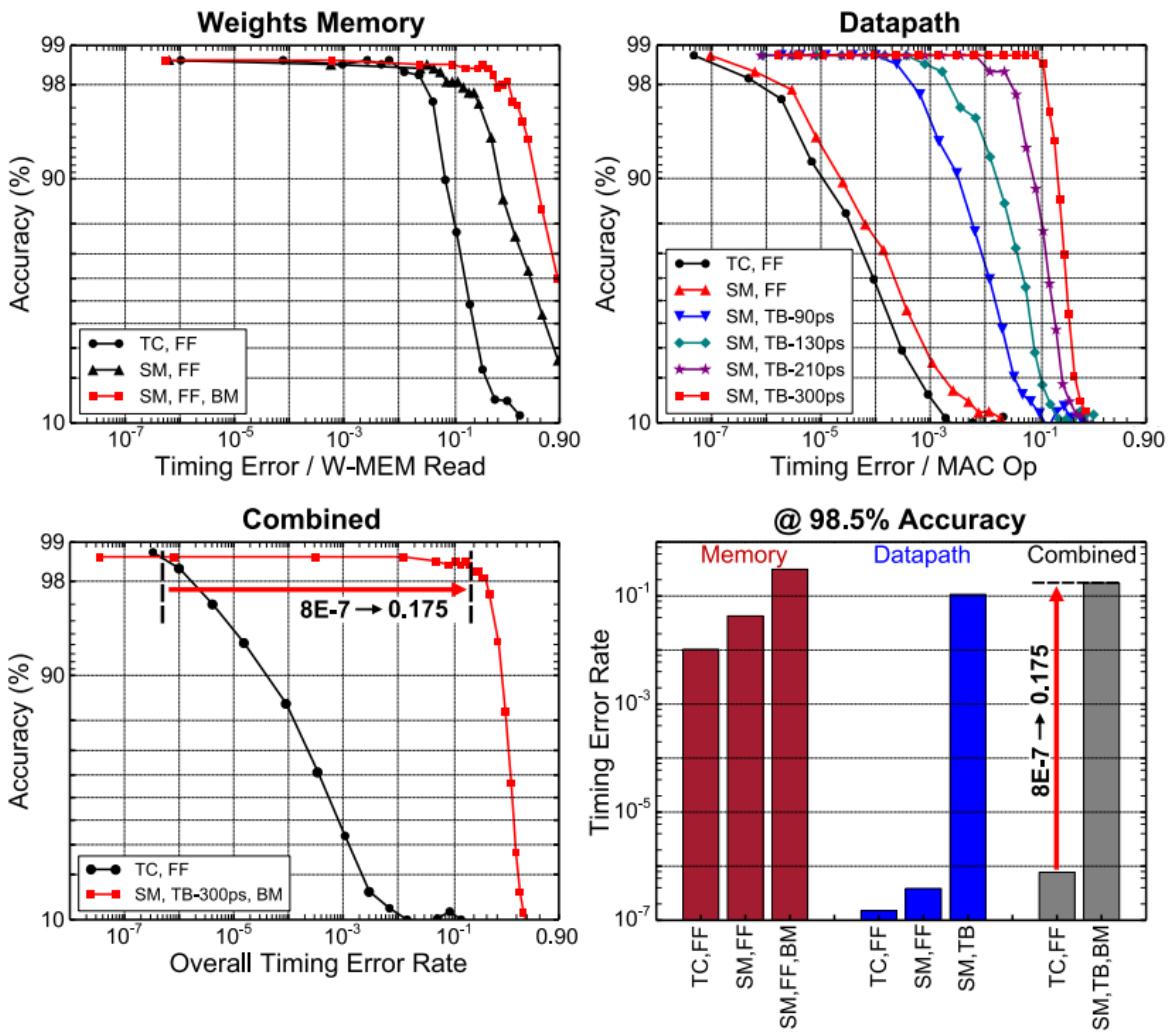
Thí nghiệm 1: Kết quả đo đạc error rate, năng lượng tiêu thụ đối với các giá trị hiệu điện thế cung cấp ở tần số 667 MHz (hình trên) và error rate, băng thông đối với các giá trị tần số làm việc ở nguồn 0.9V (hình dưới) ở độ chính xác 16 bit.



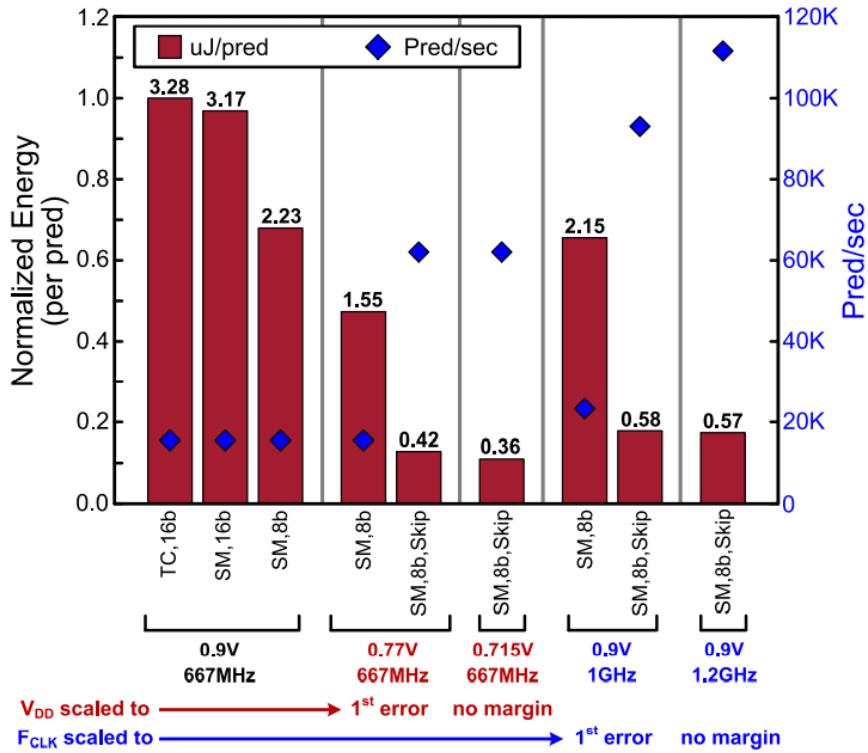
Thí nghiệm 2: Kết quả về năng lượng tiêu thụ ở WMEM và DNN Engine ở các độ chính xác là 16-bit và 8-bit ở hiệu điện thế 0.9V, 1 GHz.



Thí nghiệm 3: Kết quả về độ chính xác của mô hình ANN và timing violation rates ở WMEM, data path của MAC và tổ hợp của chúng.

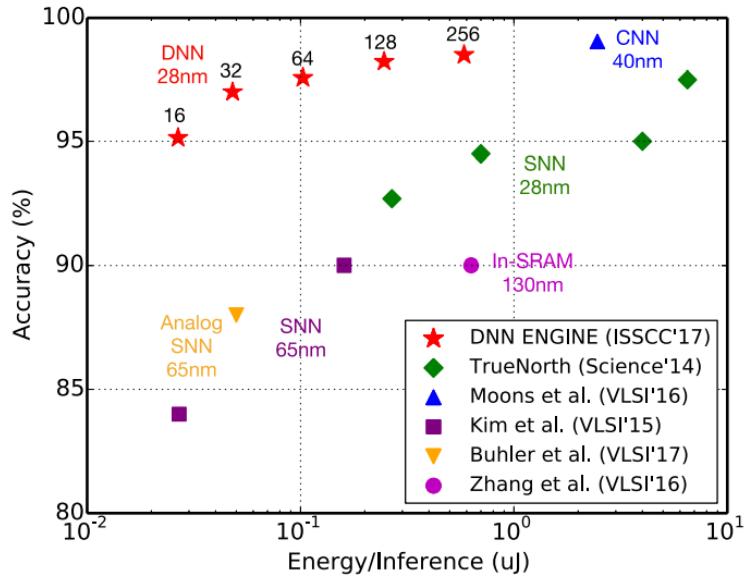


Thí nghiệm 4: Kết quả đo lường năng lượng và băng thông ở các cài đặt khác nhau ở độ chính xác của mô hình (MNIST) từ 98.36% trở lên. Các cài đặt bao gồm độ chính xác 16 hoặc 8 bit, TC hoặc SC format, và việc dùng sparsity optimization (skip).



Thí nghiệm 5: Kết quả so sánh về độ chính xác của mô hình và mức năng lượng tiêu hao giữa DNN Engine với các công trình trước đó:

- J. K. Kim, P. Knag, T. Chen, and Z. Zhang, “A 640 M pixel/s 3.65 mW sparse event-driven neuromorphic object recognition processor with on-chip learning,” in Proc. IEEE Symp. VLSI Circuits, Jun. 2015, pp. C50–C51.
- F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang, and M. P. Flynn, “A 3.43 TOPS/W 48.9 pJ/pixel 50.1 nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40 nm CMOS,” in Proc. Symp. VLSI Circuits, Kyoto, Japan, 2017, pp. C30–C31.
- P. Merolla, “A million spiking-neuron integrated circuit with a scalable communication network and interface,” Science, vol. 345, no. 6197, pp. 668–673, 2014.
- B. Moons and M. Verhelst, “A 0.3–2.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets,” in Proc. IEEE Symp. VLSI Circuits, Jun. 2016, pp. 1–2.
- J. Zhang, Z. Wang, and N. Verma, “In-memory computation of a machine-learning classifier in a standard 6T SRAM array,” IEEE J. Solid-State Circuits, vol. 52, no. 4, pp. 915–924, Apr. 2017.



Thí nghiệm 6: So sánh yếu tố timing error tolerant với các công trình trước đó

- J. Tschanz et al., “A 45 nm resilient and adaptive microprocessor core for dynamic variation tolerance,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, San Francisco, CA, USA, Feb. 2010, pp. 282–283.
- D. Bull, S. Das, K. Shivshankar, G. Dasika, K. Flautner, and D. Blaauw, “A power-efficient 32b ARM ISA processor using timing-error detection and correction for transient-error tolerance and adaptation to PVT variation,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2010, pp. 284–285.
- E. P. Kim, D. J. Baker, S. Narayanan, D. L. Jones, and N. R. Shanbhag, “Low power and error resilient PN code acquisition filter via statistical error compensation,” in Proc. IEEE Custom Integr. Circuits Conf., Sep. 2011, pp. 1–4.
- S. Das, G. S. Dasika, K. Shivashankar, and D. Bull, “A 1 GHz hardware loop-accelerator with razor-based dynamic adaptation for energyefficient operation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 8, pp. 2290–2298, Aug. 2014.
- P. N. Whatmough, S. Das, and D. M. Bull, “A low-power 1 GHz razor FIR accelerator with time-borrow tracking pipeline and approximate error correction in 65 nm CMOS,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2013, pp. 428–429.
- P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, “A 28 nm SoC with a 1.2 GHz 568 nJ/prediction sparse deep-neural-network engine with >0.1 timing error rate tolerance for IoT applications,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2017, pp. 242–243. (Công trình này chính là bài báo hội nghị trước đó)

TABLE II
COMPARISON OF TIMING ERROR TOLERANT ARCHITECTURES

	Tschanz ISSCC'10 [19]	Bull ISSCC'10 [20]	Kim CICC'11 [26]	Das CICC'13 [22]	Whatmough ISSCC'13 [24]	This work ISSCC'17 [12]
Tech. Area	45nm 13.64mm ²	65nm 4mm ²	180nm 0.24mm ²	65nm 0.24mm ²	65nm 0.53mm ²	28nm 5.76mm ²
Pipeline	8-stage 32-bit LEON-3 CPU	8-stage 32-bit ARM CPU	256-tap correlator	Sobel Edge Detection	16-tap FIR filter	FC DNN
Error - Detection	Double Sampling + Latch	Transition Detector + FF	Algorithmic (FF)	Transition Detector + Latch	Transition-Detector + Latch	Double-Sampling + Latch
Error - Correction	Exact- Replay	Exact- Replay	Approximate- Mean/Median	Exact- Replay	Approximate- Interpolation + TB ^c	Approximate- Algo. + TB ^c
F _{MAX}	1.45 GHz	1 GHz	50 MHz	667 MHz	1 GHz	667 MHz
F _{MAX} Gain	41% ^a	50% ^b				50% ^a / 80% ^b
Energy Gain	22% ^a	52% ^b	87% ^b	34%	37% ^a	30% ^a / 40% ^b

^a First error point, with margin remaining.

^b Zero-margin point, beyond which accuracy degrades.

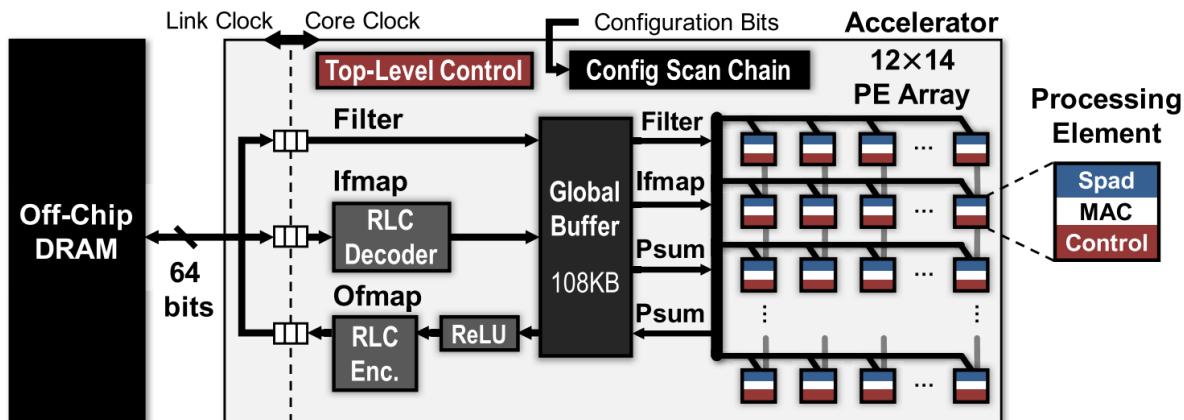
^cTB – time borrowing.

1.7. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks

Ý tưởng: Eyeriss dựa vào hai cách tiếp cận chính để cải thiện về hiệu suất dùng năng lượng:

- Giảm lượng dịch chuyển dữ liệu
- Khai thác tính phân phối, thống kê của dữ liệu

Kiến trúc tổng quát



Hệ thống bao gồm hai miền clock: miền core clock phục vụ và miền link clock. Hai miền chạy độc lập với nhau và giao tiếp thông qua một interface FIFO bất đồng bộ

- Miền core clock phục vụ cho việc xử lý bao gồm mảng spatial của 168 Processing Element (12 x 14), 108 kB GLB, RLC CODEC và khối ReLU. Để nhận dữ liệu cần cho tính toán, PE giao tiếp với các PE kế cận, hoặc GLB thông qua NoC, hoặc truy xuất vào bộ nhớ local của nó là spad.
- Miền link clock phục vụ cho giao tiếp với DRAM bên ngoài chip.

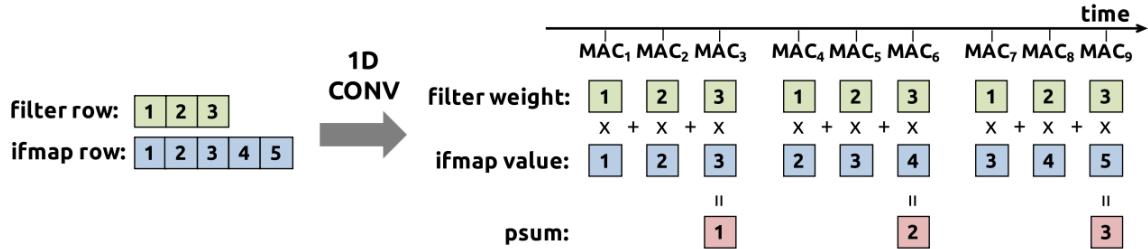
A. Row Stationary Dataflow

RS Dataflow sử dụng một cách tiếp cận hệ thống để tối ưu hóa quá trình này:

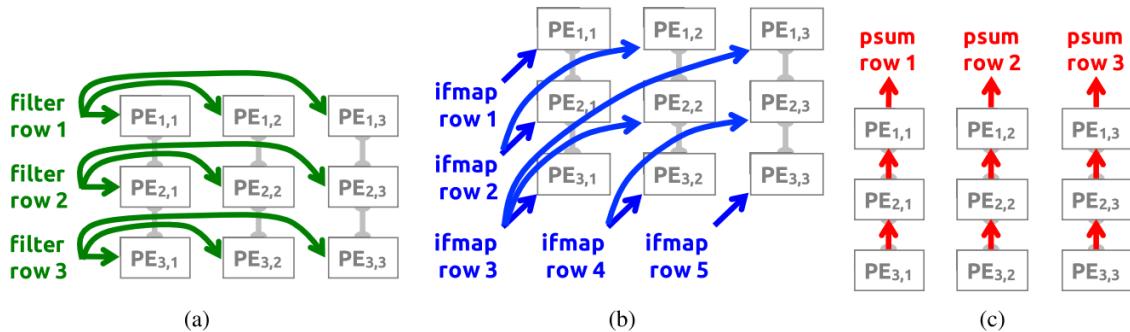
- 1-D Convolution Primitive: RS Dataflow đầu tiên chia quá trình tính toán thành các 1-D Convolution Primitive để các thành phần này có thể chạy song song với nhau. Mỗi tính toán primitive ở trên 1

hàng của kernel weight và 1 hàng của input feature map và tạo ra 1 hàng các tổng bán phần. Các tổng này sau đó được cộng tích lũy giữa các PE để tạo thành output feature map cuối cùng.

Bằng cách ánh xạ mỗi primitive cho 1 PE, quá trình tính toán của từng cặp hàng ổn định ở trong PE. (row pair stays stationary)



- 2-D Convolution PE Set: Một 2-D Convolution Set là tập hợp của nhiều 1-D Convolution primitive và việc tính toán của nó:
 - Chia sẻ cùng hàng của kernel hoặc input feature map giữa các 1-D Convoluton primitive.
 - Cộng tích lũy giá trị tổng bán phần ở nhiều primitive với nhau



Trong một Set, mỗi hàng của kernel được tái sử dụng theo chiều ngang, mỗi hàng của input feature map được tái sử dụng theo đường chéo và mỗi hàng của tổng bán phần được cộng tích lũy theo chiều dọc

- PE Set Mapping: Kích thước của PE set là hàm số của hình dạng của một layer và độc lập với kích thước vật lý của PE array. Một PE set có thể ánh xạ vào bất kỳ nhóm nào của PE array mà có cùng kích thước. Tuy nhiên, không thể áp dụng cho:
 - PE set có nhiều hơn 168 thành phần
 - PE set có ít hơn 168 thành phần, nhưng có kích thước chiều dài lớn hơn 14 hoặc chiều rộng lớn hơn 12.
- Dimensions Beyond 2-D in PE Array: Xử lý nhiều phép 2-D convolution là bắt buộc để hoàn thành bởi vì sự có mặt của 3 yếu tố là batch size, số lượng channel và số lượng kernel weight. 2-D convolution sử dụng:
 - Các input feature map khác nhau sử dụng cùng một kernel weight (kernel reuse)
 - Các kernel weight khác nhau sử dụng cùng input feature map (input feature map reuse)

- Các kernel và input feature map từ các channel khác nhau cộng tích lũy kết quả tổng bán phần với nhau (partial sum accumulate).

Việc tái sử dụng kernel có thể đạt được bằng cách cho đi ngang qua nhiều input feature map ở cùng một PE set một cách tuần tự.

Việc tái sử dụng input feature map và tích lũy tổng bán phần có thể được khai thác bằng cách tận dụng local spad hoặc tính chất song song hóa độc lập của PE array.

- Multiple 2-D Convolutions in a PE Set: Mỗi PE set được chạy nhiều tác vụ 2-D convolution ở kernel và channel khác nhau.
- Multiple PE Sets in the PE Array: PE array có thể chạy nhiều hơn một PE set nếu PE set đủ nhỏ.

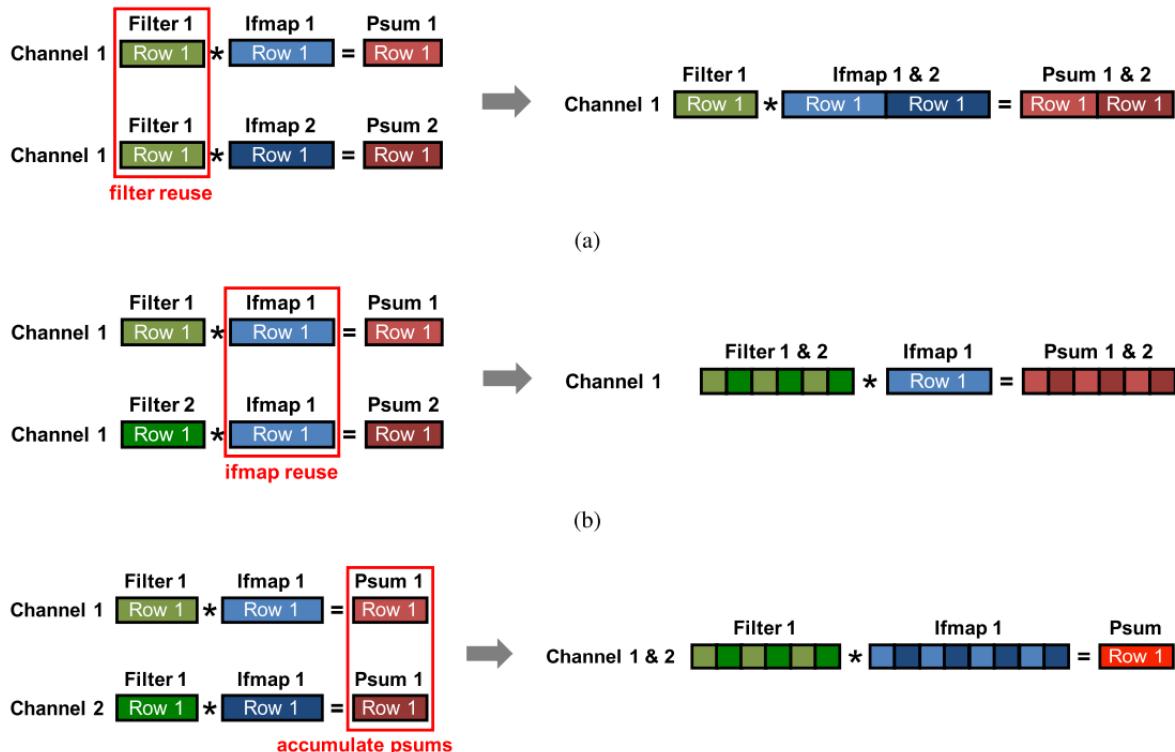


TABLE II
MAPPING PARAMETERS OF THE RS DATAFLOW

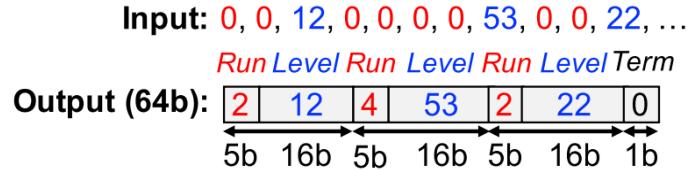
Parameter	Description
m	number of ofmap channels stored in the global buffer
n	number of ifmaps used in a processing pass
e	width of the PE set (strip-mined if necessary)
p	number of filters processed by a PE set
q	number of channels processed by a PE set
r	number of PE sets that process different channels in the PE array
t	number of PE sets that process different filters in the PE array

B. Data Statistics

Mục đích:

- Giảm số lượng truy cập vào DRAM bằng cách sử dụng compression
- Bỏ qua các tính toán không cần thiết.

Ý tưởng: Hàm ReLU mang tới nhiều các giá trị 0 vào feature map bằng cách điều chỉnh các âm thành các giá trị 0. Eyeriss sử dụng RLC để khai thác tính chất này. Trừ feature map ở tầng đầu tiên (input của hệ thống), tất cả input feature map được lưu dưới RLC ở trong DRAM.



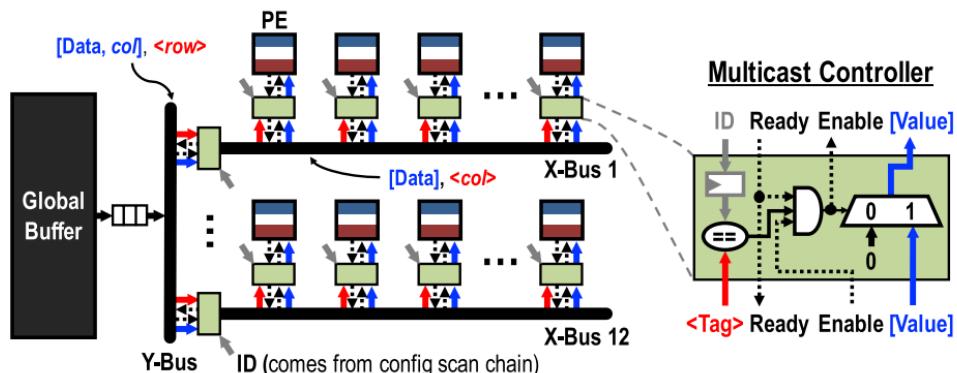
C. Global Buffer

GLB có kích thước 108 kB có thể giao tiếp với DRAM thông qua interface bất đồng bộ và với PE array thông qua NoC. GLB lưu trữ tất cả ba loại dữ liệu: input feature map, kernel, và partial sum/output feature map.

D. Network-on-Chip

NoC quản lý việc phân phối dữ liệu giữa GLB và PE array cũng như giữa các PE với nhau. NoC bao gồm 3 loại network khác nhau:

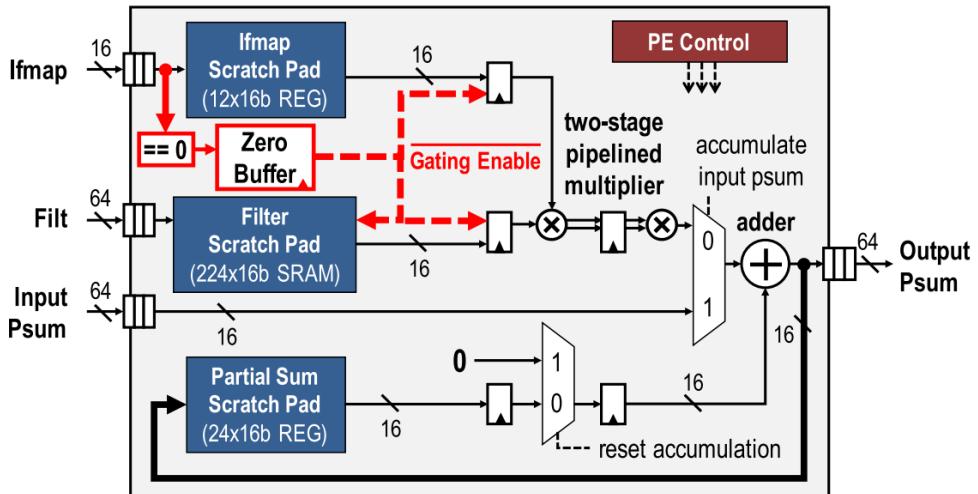
- Global Input Network: GIN được tối ưu cho việc single-cycle multicast từ GLB tới một nhóm các PEs mà nhận cùng giá trị kernel weight, input feature map hoặc là partial sum.



- Global Output Network: GON được dùng để đọc giá trị partial sum được tạo ra từ PE array trở lại GLB. GON có cùng kiến trúc với GIN, chỉ khác là chiều dữ liệu ngược lại.
- Local Network: Giữa từng cặp của PE mà thuộc hai hàng liên tiếp của cùng một cột có một data path 64 bit được hiện thực để đưa partial sum từ PE dưới lên PE trên trực tiếp.

E. Processing Element

Kiến trúc của PE



PE bao gồm:

- FIFO được dùng để được dùng ở I/O để cân bằng tải giữa NoC và tính toán.
- Số lượng kernel và channel mà một PE xử lý ở một thời điểm được cài đặt tĩnh ở PE Control.
- Data path được pipeline gồm 3 stage: 1 stage cho việc truy cập spad và 2 stage cho tính toán.
- Spad được phân tách độc lập cho 3 loại dữ liệu để cung cấp đủ băng thông truy cập.
- Data gating logic được hiện thực để khai thác giá trị 0 trong input feature map.

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Eyeriss được hiện thực bằng ASIC theo công nghệ TSMC 65nm và được tích hợp vào Caffe framework.

TABLE IV
CHIP SPECIFICATIONS

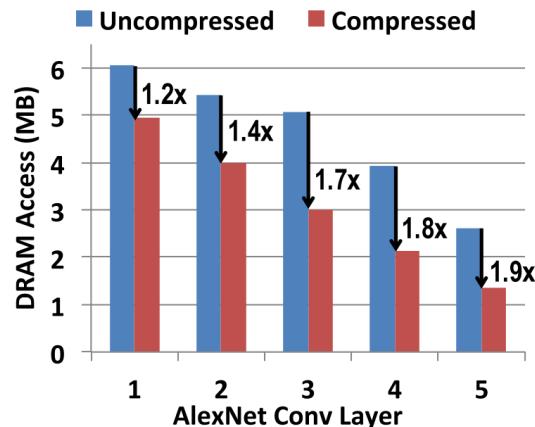
Technology	TSMC 65nm LP 1P9M
Chip Size	4.0 mm × 4.0 mm
Core Area	3.5 mm × 3.5 mm
Gate Count (logic only)	1176k (2-input NAND)
On-Chip SRAM	181.5K bytes
Number of PEs	168
Global Buffer	108.0K bytes (SRAM)
Scratch Pads (per PE)	filter weights: 448 bytes (SRAM) feature maps: 24 bytes (Registers) partial sums: 48 bytes (Registers)
Supply Voltage	core: 0.82–1.17 V I/O: 1.8 V
Clock Rate	core: 100–250 MHz link: up to 90 MHz
Peak Throughput	16.8–42.0 GMACS
Arithmetic Precision	16-bit fixed-point
Natively Supported CNN Shapes	filter height (R): 1–12 filter width (S): 1–32 num. of filters (M): 1–1024 num. of channels (C): 1–1024 vertical stride: 1, 2, 4 horizontal stride: 1–12

DNN được dùng:

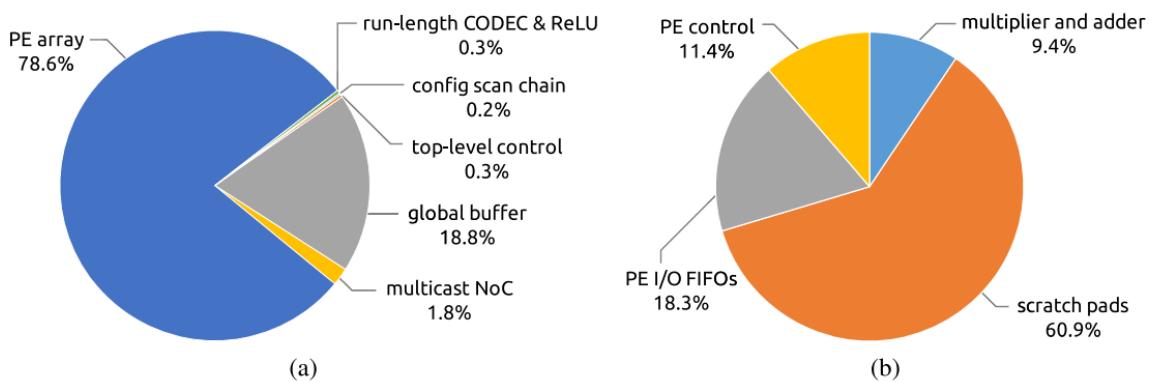
- AlexNet
- VGG-16

Kết quả thí nghiệm

Thí nghiệm 1: Kết quả so sánh số lượng truy cập DRAM (đọc và ghi) trước và sau khi dùng RLC ở 5 convolution layer của AlexNet



Thí nghiệm 2: Kết quả về diện tích chiếm dụng của từng thành phần trong lõi Eyeriss và PE.

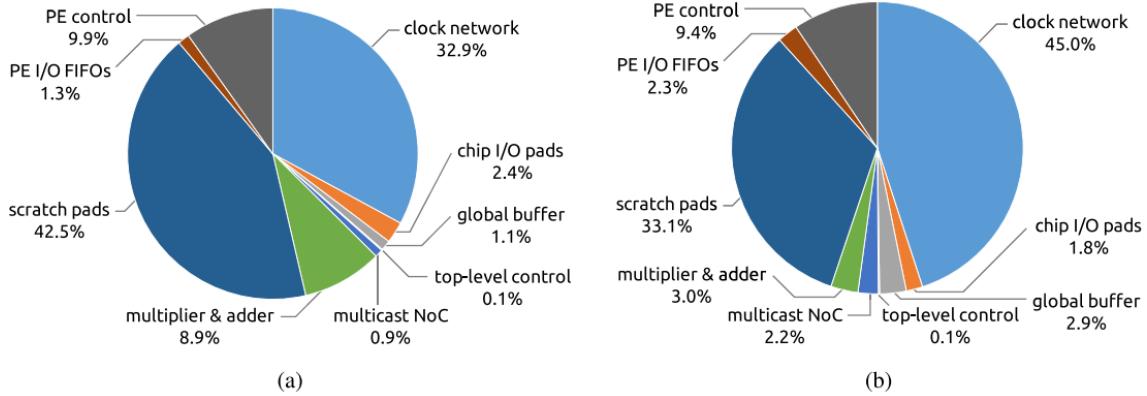


Thí nghiệm 3: Kết quả đo lường hiệu năng của 5 convolution layer thuộc AlexNet ở 1V.

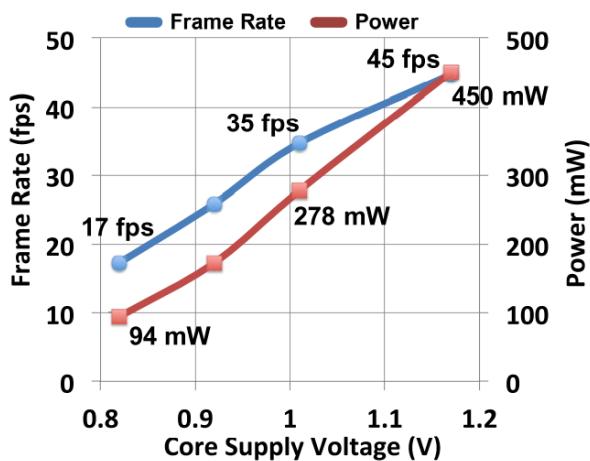
TABLE V
PERFORMANCE BREAKDOWN OF THE FIVE CONV LAYERS IN AlexNet AT 1 V. BATCH SIZE (N) IS 4.
THE CORE AND LINK CLOCKS RUN AT 200 AND 60 MHz, RESPECTIVELY

Layer	Power (mW)	Total Latency (ms)	Processing Latency (ms)	Num. of MACs	Num. of Active PEs	Zeros in Ifmaps (%)	Global Buff. Accesses	DRAM Accesses
CONV1	332	20.9	16.5	0.42G	154 (92%)	0.01%	18.5 MB	5.0 MB
CONV2	288	41.9	39.2	0.90G	135 (80%)	38.7%	77.6 MB	4.0 MB
CONV3	266	23.6	21.8	0.60G	156 (93%)	72.5%	50.2 MB	3.0 MB
CONV4	235	18.4	16.0	0.45G	156 (93%)	79.3%	37.4 MB	2.1 MB
CONV5	236	10.5	10.0	0.30G	156 (93%)	77.6%	24.9 MB	1.3 MB
Total	278	115.3	103.5	2.66G	148 (88%)	57.53%	208.5 MB	15.4 MB

Thí nghiệm 4: Kết quả về phân tích năng lượng của chip khi chạy ở layer CONV1 và CONV5 thuộc AlexNet.



Thí nghiệm 5: Kết quả phân tích ảnh hưởng giữa hiệu điện thế nguồn và hiệu năng khi chạy AlexNet.



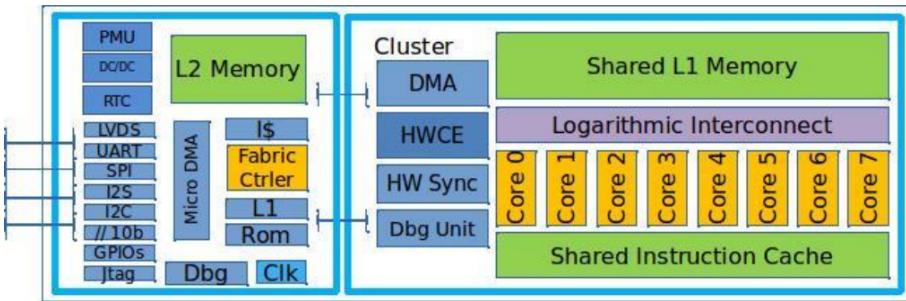
Thí nghiệm 6: Kết quả đo lường hiệu năng của 13 convolution layer thuộc VGG-16 ở 1V.

TABLE VI
PERFORMANCE BREAKDOWN OF THE 13 CONV LAYERS IN VGG-16 AT 1 V. BATCH SIZE (N) IS 3.
THE CORE AND LINK CLOCKS RUN AT 200 AND 60 MHz, RESPECTIVELY

Layer	Power (mW)	Total Latency (ms)	Processing Latency (ms)	Num. of MACs	Num. of Active PEs	Zeros in Ifmaps (%)	Global Buff. Accesses	DRAM Accesses
CONV1-1	247	76.2	38.0	0.26G	156 (93%)	1.6%	112.6 MB	15.4 MB
CONV1-2	218	910.3	810.6	5.55G	156 (93%)	47.7%	2402.8 MB	54.0 MB
CONV2-1	242	470.3	405.3	2.77G	156 (93%)	24.8%	1201.4 MB	33.4 MB
CONV2-2	231	894.3	810.8	5.55G	156 (93%)	38.7%	2402.8 MB	48.5 MB
CONV3-1	254	241.1	204.0	2.77G	156 (93%)	39.7%	607.4 MB	20.2 MB
CONV3-2	235	460.9	408.1	5.55G	156 (93%)	58.1%	1214.8 MB	32.2 MB
CONV3-3	233	457.7	408.1	5.55G	156 (93%)	58.7%	1214.8 MB	30.8 MB
CONV4-1	278	135.8	105.1	2.77G	168 (100%)	64.3%	321.8 MB	17.8 MB
CONV4-2	261	254.8	210.0	5.55G	168 (100%)	74.7%	643.7 MB	28.6 MB
CONV4-3	240	246.3	210.0	5.55G	168 (100%)	85.4%	643.7 MB	22.8 MB
CONV5-1	258	54.3	48.3	1.39G	168 (100%)	79.4%	90.0 MB	6.3 MB
CONV5-2	236	53.7	48.5	1.39G	168 (100%)	87.4%	90.0 MB	5.7 MB
CONV5-3	230	53.7	48.5	1.39G	168 (100%)	88.5%	90.0 MB	5.6 MB
Total	236	4309.5	3755.2	46.04G	158 (94%)	58.6%	11035.8 MB	321.1 MB

1.8. GAP-8: A RISC-V SoC for AI at the Edge of the IoT

Kiến trúc tổng quát:



Hệ thống bao gồm 2 miền điện áp và tần số khác nhau điều khiển bởi Fabric Controller (FC) và Cluster.

A. Fabric Controller

Fabric Controller là một MCU nâng cao với nền tảng là lõi RISC-V kết hợp thêm instruction cache và bộ nhớ truy xuất nhanh. SoC cung cấp đủ các ngoại vi cần thiết như QSPI, I2C, I2S, CAM, UART,...

B. Parallel Computing Cluster

Cluster, nằm ở miền điện thế và tần số riêng biệt, được bật lên bởi Fabric Controller khi có tải nặng về tính toán xuất hiện.

Cluster gồm:

- 8 lõi RISC-V gồm 4-stage pipeline, và dùng tập ISA RISC-V I, M và C.
- HW Sync đóng vai trò quản lý event, phân phối đa luồng và đồng bộ hóa
- Hardware Convolution Engine (HWCE) thực hiện các tải tính toán khi thực hiện tác vụ ở Convolution layer. HWCE được sử dụng lại ở bài báo sau:

F. Conti and L. Benini, "A ultra-low-energy convolution engine for fast brain-inspired vision in multicore clusters," 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015, pp. 683-688.

C. Power Management

SoC có thêm bộ chuyển đổi DC/DC để kết nối trực tiếp với pin bên ngoài.

- Khi hệ thống ở trạng thái hoạt động, hệ thống cần cấp nguồn từ 1.0V - 1.2V
- Khi hệ thống ở trạng thái ngủ, bộ chuyển đổi DC/DC được tắt thay vào đó là bộ chuyển đổi linear drop-out được dùng để duy trì cho clock phục vụ cho việc wake up hệ thống.

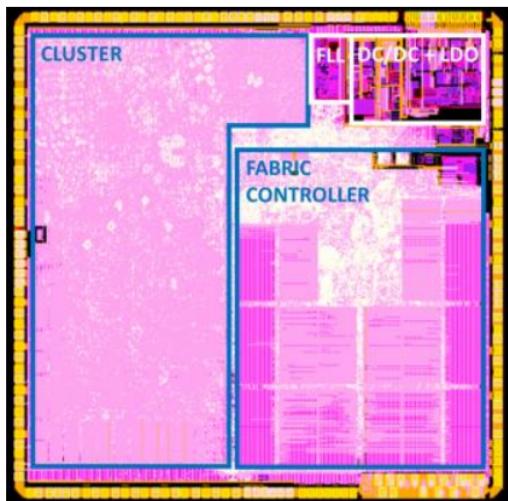
D. Programming

Hỗ trợ GCC 7.1

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Hệ thống hiện thực theo ASIC với công nghệ TSMC 55nm.



Technology	CMOS 55nm LP
Die Area	10 mm ²
Embedded Memory	4608 Kbit (4096kbit state-retentive)
Logic Gates	2 Mgates
Power Management	1 DC/DC, 1 LDO, 2 FLLs, embedded power switches
VDD range	0.8V – 1.2V
Frequency Range	32kHz – 250 MHz
Power Range	3.6 µW – 75mW

Ứng dụng được dùng: 1D FFT1024 Radix4, 2D FFT 256 x 256 Radix4; Byte 5x5 Conv; Short 5x5 Conv; Binary 5x5 Conv; Short MaxPool2x2; Short MatMult 32x32; Short 2048 to 1 Fully Connected; CannyEdge; AES-CTR 128b; 64 Mel Coefficients; HoG, 8x8 Cells, 2x2Blocks,9 Bins

DNN được dùng là CNN

- CIFAR10
- MNIST
- TextReco

Kết quả thí nghiệm

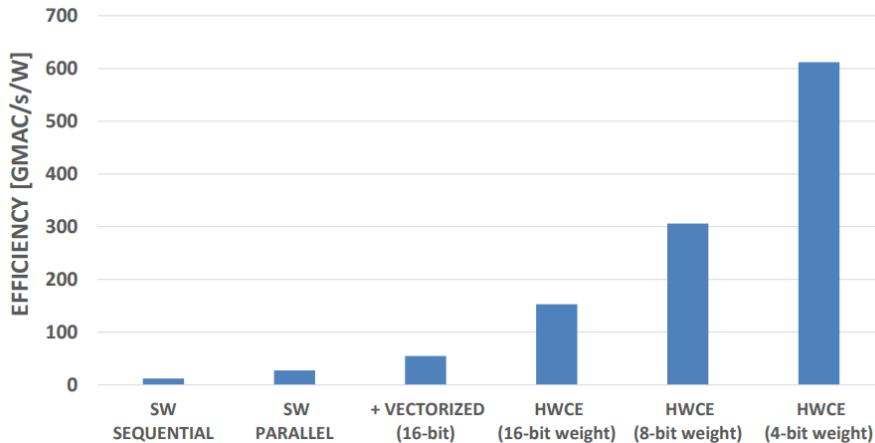
Thí nghiệm 1: Kết quả hiệu năng của hệ thống và các thành phần của hệ thống ở tần số 100MHz.

TABLE I. GAP-8 PEAK PERFORMANCE @ 100 MHz.

	32x32-bit GMAC/s	16x16-bit GMAC/s	8x8-bit GMAC/s	16x4-bit GMAC/s
SoC	0.1	0.2	0.4	-
Cluster	0.8	1.6	3.2	-
HWCE	-	2.8	5.6*	11.2*
Total	0.9	4.6	9.2	11.2

*HWCE supports also 16x8 and 8x4, and 4x4 configurations.

Thí nghiệm 2: So sánh kết quả hiệu năng khi chạy các ứng dụng CNN giữa HWCE ở các độ chính xác khác nhau và phiên bản Software (Sequential và Parallel hiện thực ở các lõi Cluster)



Thí nghiệm 3: Kết quả vì các chế độ làm việc của hệ thống với điện thế nguồn, mức năng lượng tiêu thụ và tần số.

TABLE II. GAP-8 POWER MODES AND CONSUMPTION.

Power Mode	VDD [V]	Nom. Freq. [MHz]	Power @ Nom. Freq.
Deep Sleep	0.8	0.32	3.6 μ W
Retentive Deep Sleep	0.8	0.32	30 μ W
FC ON	1.0	150	27 mW
FC ON, Cluster ON	1.0	150, 90	37 mW
FC ON	1.2	250	39 mW
FC ON, Cluster ON	1.2	250, 170	75 mW

Thí nghiệm 4: Kết quả về số cycle cần để chạy ứng dụng benchmark ở các số lõi khác nhau.

TABLE III. APPLICATION PERFORMANCE [KCYCLES].

Application	Cores				Speed-up vs. 1-core
	1	2	4	8	
1D FFT1024 Radix4	28.2	14.3	7.8	4.7	6×
2D FFT 256 x 256 Radix4	78.9	41.9	22.6	13.3	5.9×
Byte 5x5 Conv	18.5	9.3	4.7	2.2	8.4×
Short 5x5 Conv	37.8	18.9	9.5	4.6	8.2×
Binary 5x5 Conv	20.8	10.5	5.3	2.8	7.4×
Short MaxPool2x2	8.2	4.2	2.1	1.1	7.5×
Short MatMult 32x32	41.9	20.9	14.0	5.2	8×
Short 2048 to 1 Fully Connected	3112.0	1616.0	847.0	495.0	6.3×
CannyEdge	99.5	50.9	26.2	12.7	7.8×
AES-CTR 128b	15.3	7.7	4.0	2.1	7.3×
64 Mel Coefficients	542.7	299.4	176.7	101.3	5.3×
HoG, 8x8 Cells, 2x2Blocks,9 Bins	65.0	35.0	18.0	9.0	7.2×

Thí nghiệm 5: Kết quả về số cycle cần để chạy DNN benchmark ở các số lõi khác nhau.

TABLE IV. FULL CNN PERFORMANCE [CYCLES].

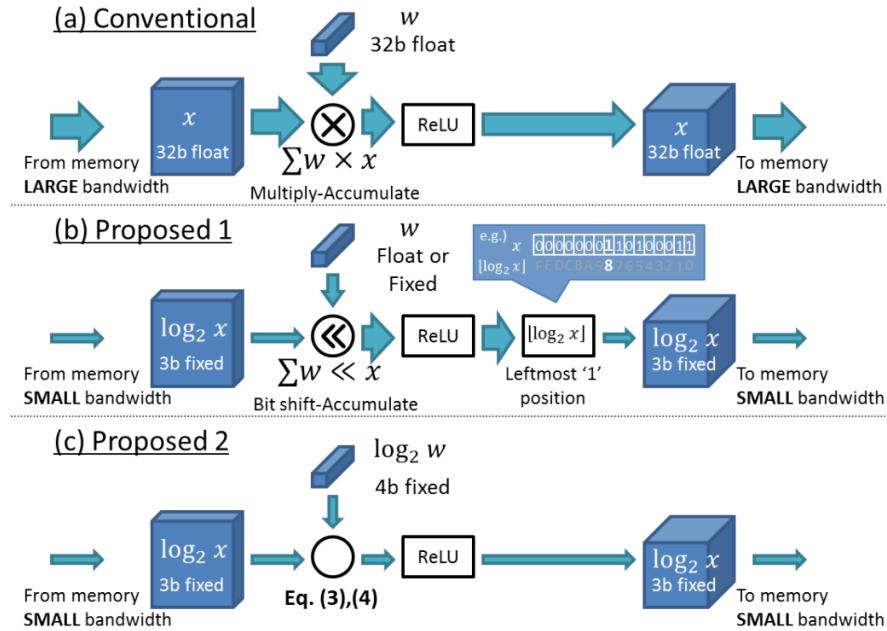
Topology	Cores					Speed-up vs. 1-core	Energy gain vs. 1-core
	1	2	4	8	+ HWCE		
CIFAR10	711 K	415 K	254 K	178 K	65 K	10.9×	4.9×
MNIST	7.6 M	4 M	2.4 M	1.5 M	0.8 M	9.3×	4.3×
TextReco	98 M	51 M	28 M	17 M	8.3 M	11.7×	5.3×

1.9. Convolutional Neural Networks using Logarithmic Data Representation

Ý tưởng: Chuyển việc biểu diễn số nguyên thông thường thành dạng logarithm. Khi đó việc tính toán phép nhân sẽ thay bằng việc cộng hai số mũ và số bit cần để biểu diễn một số giảm xuống. Tuy nhiên, độ chính xác số học cũng bị giảm xuống. Động lực nghiên cứu:

- Việc huấn luyện DNN với các giá trị weight suy biến dẫn tới kết quả weight cuối cùng phân bố không đồng đều xung quanh giá trị 0.
- Tương tự với giá trị activation cũng gần 0
- Biểu diễn Logarithmic có thể mã hóa dữ liệu với một khoảng lớn giá trị với số bit biểu diễn ít hơn so với fix-point
- Biểu diễn dữ liệu ở miền log được mã hóa trong phần cứng số một cách tự nhiên

Tổng quan phương pháp



A. Proposed Method 1

Ý tưởng: Chỉ biến đổi 1 toán tử dưới dạng biểu diễn log, làm tròn kết quả bằng phương pháp tuyến tính và nhân nó với toán tử còn lại.

$$\begin{aligned} w^T x &\simeq \sum_{i=1}^n w_i \times 2^{\tilde{x}_i} \\ &= \sum_{i=1}^n \text{Bitshift}(w_i, \tilde{x}_i) \end{aligned}$$

với $\tilde{x}_i = \text{Quantize}(\log_2(x_i))$

B. Proposed Method 2

Ý tưởng: Biến đổi mở rộng phương pháp 1 cho cả 2 toán tử. Do đó, phép nhân của hai toán tử trở thành phép cộng ở miền log.

$$\begin{aligned} w^T x &\simeq \sum_{i=1}^n 2^{\tilde{w}_i + \tilde{x}_i} \\ &= \sum_{i=1}^n \text{Bitshift}(1, \tilde{w}_i + \tilde{x}_i) \end{aligned}$$

với $\tilde{w}_i = \text{Quantize}(\log_2(w_i))$ và $\tilde{x}_i = \text{Quantize}(\log_2(x_i))$

C. Accumulation in log domain

Ý tưởng: Việc cộng tích lũy kết quả được thực hiện ở miền log dựa vào ước lượng sau $\log_2(1+x) \simeq x$ với $0 \leq x < 1$. Cụ thể với $s_n = w_1 x_1 + \dots + w_n x_n$, $\tilde{s}_n = \log_2(s_n)$, và

$\tilde{p}_i = \tilde{w}_i + \tilde{x}_i$ thì với $n = 2$,

$$\begin{aligned}\tilde{s}_2 &= \log_2 \left(\sum_{i=1}^2 \text{Bitshift}(1, \tilde{p}_i) \right) \\ &\simeq \max(\tilde{p}_1, \tilde{p}_2) + \text{Bitshift}(1, -|\tilde{p}_1 - \tilde{p}_2|)\end{aligned}$$

với với giá trị bất kỳ của n ,

$$\tilde{s}_n \simeq \max(\tilde{s}_{n-1}, \tilde{p}_n) + \text{Bitshift}(1, -|\lfloor (\tilde{s}_{n-1}) \rfloor - \tilde{p}_n|)$$

D. Logarithmic Representation Training Algorithm

Algorithm 1 Training a CNN with base-2 logarithmic representation. C is the softmax loss for each minibatch. LogQuant(x) quantizes x in base-2 log-domain. The optimization step Update(W_k, g_{W_k}) updates the weights W_k based on backpropagated gradients g_{W_k} . We use the SGD with momentum and Adam rule.

Require: a minibatch of inputs and targets (a_0, a^*) , previous weights W .

Ensure: updated weights W^{t+1}

{1. Computing the parameters' gradient:}

{1.1. Forward propagation:}

for $k = 1$ to L **do**

$W_k^q \leftarrow \text{LogQuant}(W_k)$

$a_k \leftarrow \text{ReLU}(a_{k-1}^q W_k^b)$

$a_k^q \leftarrow \text{LogQuant}(a_k)$

end for

{1.2. Backward propagation:}

Compute $g_{a_L} = \frac{\partial C}{\partial a_L}$ knowing a_L and a^*

for $k = L$ to 1 **do**

$g_{a_k}^q \leftarrow \text{LogQuant}(g_{a_k})$

$g_{a_{k-1}} \leftarrow g_{a_k}^q W_k^q$

$g_{W_k} \leftarrow g_{a_k}^{q\top} a_{k-1}^q$

end for

{2. Accumulating the parameters' gradient:}

for $k = 1$ to L **do**

$W_k^{t+1} \leftarrow \text{Update}(W_k, g_{W_k})$

end for

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Task được dùng để phân loại là classification task của ILSVRC-2012 sử dụng Chainer.

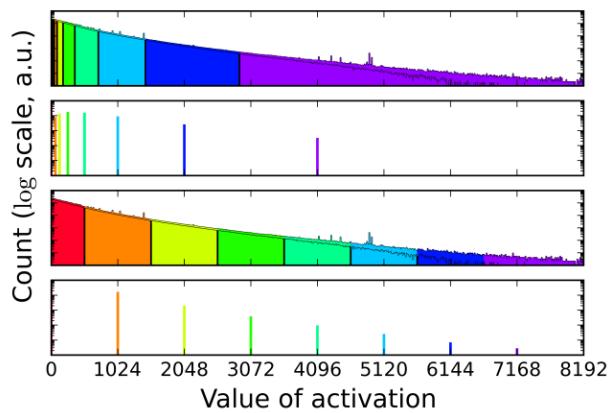
DNN được tùy chỉnh lại:

- CNN
 - AlexNet
 - VGG16

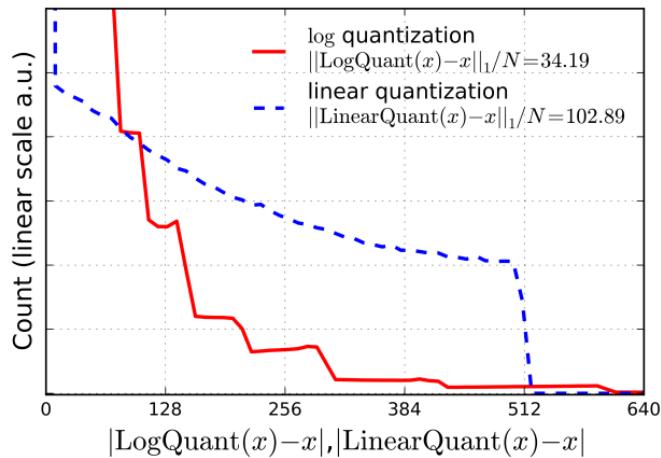
- ANN
 - CIFAR10

Kết quả thí nghiệm

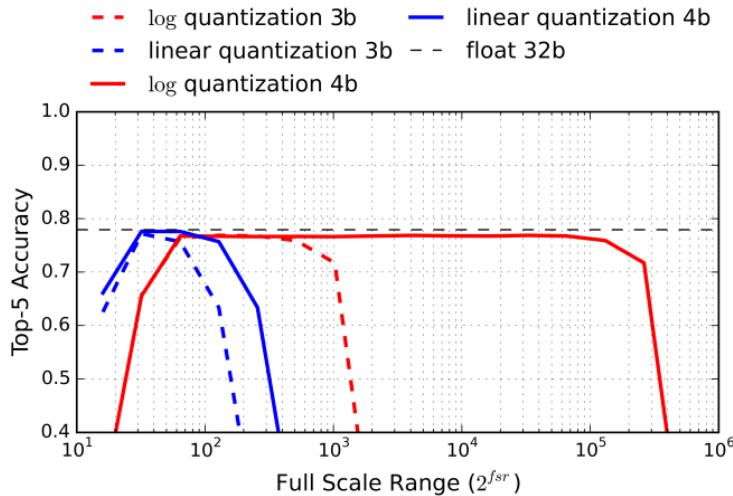
Thí nghiệm 1: Kết quả ảnh hưởng của bộ làm tròn đối với activation ở conv2_2 layer trong mô hình VGG16. Thứ tự từ trên xuống lần lượt là trước khi log-quantization, sau khi log-quantization, trước khi linear-quantization và sau khi linear-quantization



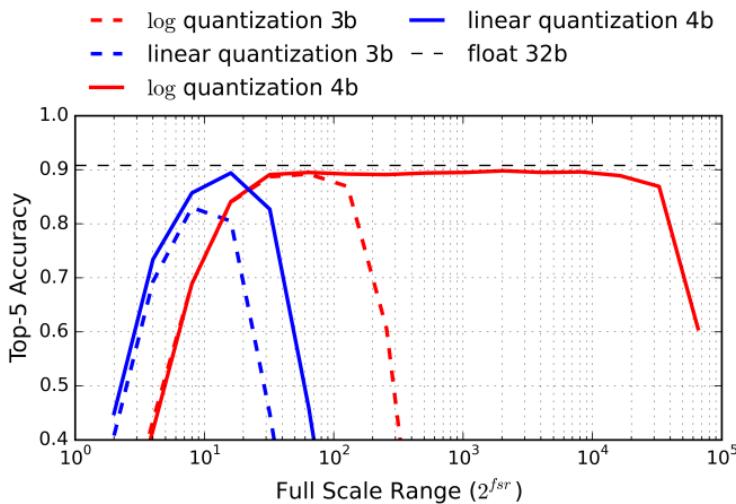
Thí nghiệm 2: Kết quả so sánh về phân phối quantization error giữa log-quantization và linear-quantization.



Thí nghiệm 3: Kết quả khi chạy trên mô hình AlexNet với việc dùng floating-point 32 bit; log-quantization 3 bit, 4 bit và linear quantization 3 bit, 4 bit.



Thí nghiệm 4: Kết quả khi chạy trên mô hình VGG16 với việc dùng floating-point 32 bit; log-quantization 3 bit, 4 bit và linear quantization 3 bit, 4 bit.



Thí nghiệm 5: Kết quả độ chính xác phân lớp với giá trị FSR tối ưu với việc dùng floating-point 32 bit; log-quantization 3 bit, 4 bit và linear quantization 3 bit, 4 bit.

Table 3. Top-5 accuracies with quantized activations at optimal FSRs

Model	AlexNet	VGG16
Float 32b	78.3%	89.8%
Log. 3b	76.9% (fsr = 7)	89.2% (fsr = 6)
Log. 4b	76.9% (fsr = 15)	89.8% (fsr = 11)
Linear 3b	77.1% (fsr = 5)	83.0% (fsr = 3)
Linear 4b	77.6% (fsr = 5)	89.4% (fsr = 4)

Thí nghiệm 6: Kết quả độ chính xác phân lớp khi thực hiện làm tròn đối với weight ở fully-connected layer với việc dùng floating-point 32 bit; log-quantization 4 bit và linear quantization 4 bit.

Table 4. Top-5 accuracy after applying quantization to weights of FC layers

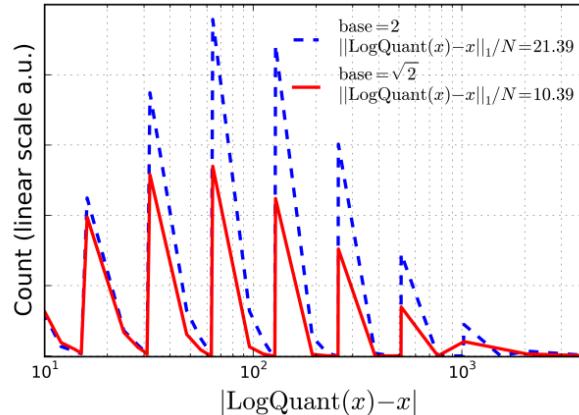
Model	Float 32b	Log. 4b	Linear 4b
AlexNet	76.9%	76.8%	76.4%
VGG16	89.8%	89.5%	89.7%

Thí nghiệm 7: Kết quả độ chính xác phân lớp khi thực hiện làm tròn đối với weight ở convolution layer với việc dùng floating-point 32 bit; linear quantization 5 bit, log-quantization cơ số 2 với cơ số $\sqrt{2}$ ở 5 bit.

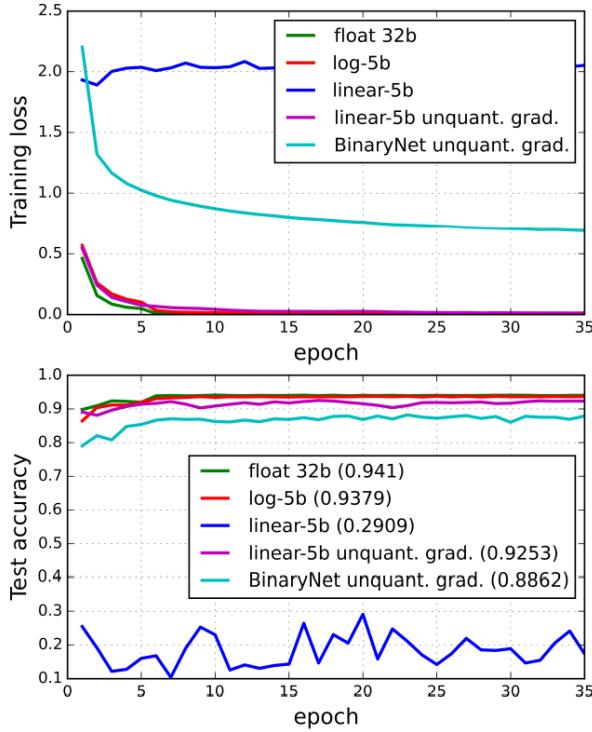
Table 5. Top-5 accuracy after applying quantization to weights of convolutional layers

Model	Float 32b	Linear 5b	Base-2 Log 5b	Base- $\sqrt{2}$ Log 5b
AlexNet	76.8%	73.6%	70.6%	75.1%
VGG16	89.5%	85.1%	83.4%	89.0%

Thí nghiệm 8: Kết quả phân phối quantization error cho log-quantization cơ số 2 với cơ số $\sqrt{2}$ ở 5 bit.

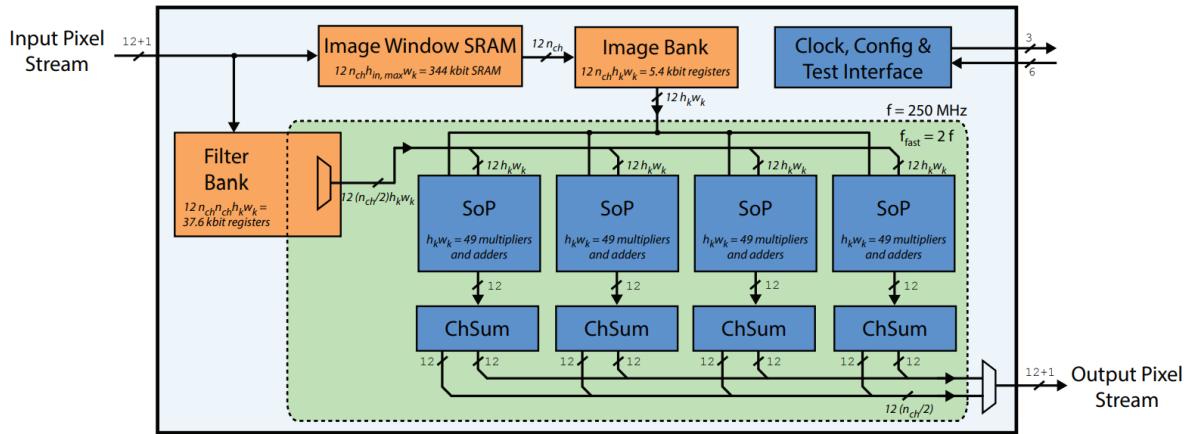


Thí nghiệm 9: Kết quả khi huấn luyện mô hình ANN của tập dữ liệu CIFAR10 ở floating-point 32 bit, log-quantization 5 bit, linear 5 bit và BinaryNet.



1.10. Origami: A 803 GOps/W Convolutional Network Accelerator

Kiến trúc Origami



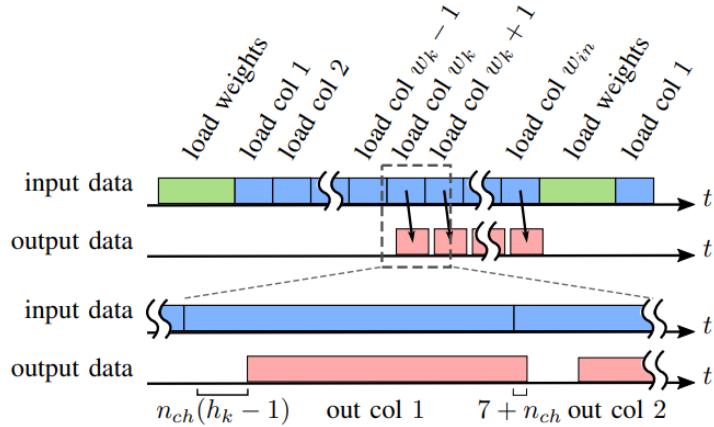
Origami bao gồm:

- Image Window SRAM và Image Bank đóng vai trò lưu trữ dữ liệu mới được đưa vào và cung cấp cho mỗi SoP một image patch cần thiết ở mỗi chu kỳ tính toán.
- Filter Bank: Filter Bank lưu tất cả giá trị kernel weight. Ở configuration mode, các giá trị này được thêm vào. Ở function mode, các giá trị chỉ cho phép đọc (read-only)
- Sum-of-Products Units: SoP unit thực hiện tính toán tích giữa image feature map patch và kernel weight.
- Channel Summer Units: Mỗi đơn vị ChSum thực hiện cộng tích lũy các kết quả được thực hiện bởi SoP cho ra kết quả cuối cùng và đưa ra ngoài.

Origami sử dụng hoạt động ở hai miền tần số khác nhau.

- Một miền tần số thấp phục vụ cho I/O, SRAM và thành phần không yêu cầu chạy nhanh như image và filter bank.
- Một miền tần số cao phục vụ cho việc tính toán của SoP và ChSum. Các khối này cũng chạy pipeline với nhau: 1 stage cho SoP và 1 stage cho ChSum.

A. Dataflow

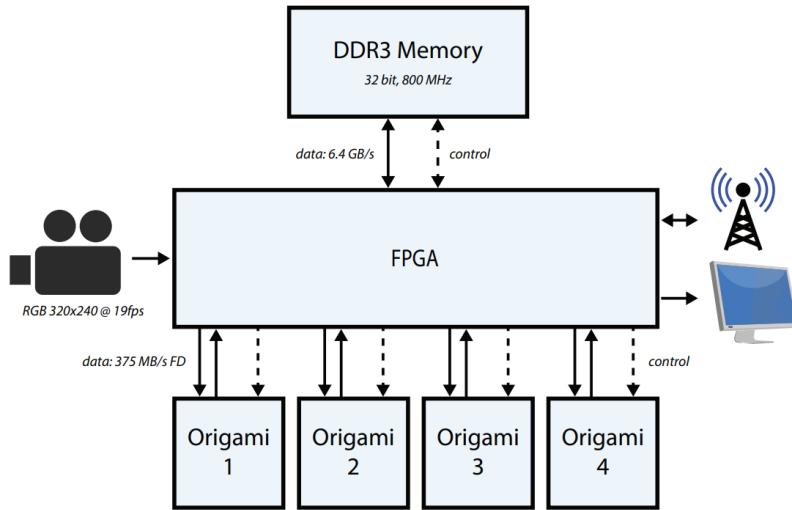


Quá trình tính toán như sau:

- Dữ liệu đầu vào được đưa Image Window SRAM với kích thước được cài đặt sẵn và cách một khoảng bằng stripe, và đảm bảo vị trí của spatial window của dữ liệu hình ảnh đầu vào. (Đơn giản là load đoạn input feature map tiếp theo cho kernel)
- Sau đó, dữ liệu được tải vào Image Bank, mà ở đó một phần input feature map có kích thước nhỏ hơn kernel được giữ lại ở register và di chuyển xuống trên giá trị stripe trước khi sang cột tiếp theo. Image Bank cung cấp đầu vào cho các đơn vị SoP để thực hiện các tính toán bên trong cho từng kernel.
- Mỗi đơn vị SoP thực hiện tính toán với cùng channel của input feature map ở các kernel khác nhau, do đó cho ra kết quả là tổng bán phần của các output feature map ở các channel khác nhau.
- Hệ thống lặp lại tính toán trên ở tất cả channel của input feature map trong khi các tổng bán phần được tính lũy ở đơn vị ChSum để tạo ra kết quả cuối cùng. Sau đó, kết quả này được đưa ra khỏi Origami.

B. General Architecture

Đề xuất hệ thống tích hợp Origami



- FPGA nên được cấu hình để bao gồm lõi phục vụ việc tiền xử lý như điều chỉnh kích thước ảnh, chuyển đổi giữa không gian màu, và chuẩn hóa độ tương phản cục bộ.
- Để lưu trữ dữ liệu ở bộ nhớ sau khi tiền xử lý, đồng thời để đọc và ghi dữ liệu khi tính toán convolution layer (sử dụng Origami) và fully-connected layer, cần có DMA controller và memory controller.

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Origami được hiện thực theo ASIC với công nghệ UMC 65 nm.

TABLE V
MEASURED SILICON KEY FIGURES.

Physical Characteristics	
Technology	UMC 65 nm, 8 Metal Layers
Core/Pad Voltage	1.2 V / 1.8 V
Package	QFN-56
# Pads	55 (i: 14, o: 13, clk/test: 8, pwr: 20)
Core Area	3.09 mm ²
Circuit Complexity ^a	912 kGE (1.31 mm ²)
Logic (std. cells)	697 kGE (1.00 mm ²)
On-chip SRAM	344 kbit
Performance & Efficiency @ 1.2 V	
Max. Clock Frequency	core: 500 MHz, i/o: 250 MHz
Power ^a @ 500 MHz	449 mW (core) + 205 mW (pads)
Peak Throughput	196 GOp/s
Effective Throughput	145 GOp/s
Core Power-Efficiency	437 GOp/s/W
Performance & Efficiency @ 0.8 V	
Max. Clock Frequency	core: 189 MHz, i/o: 95 MHz
Power ^b @ 189 MHz	93 mW (core) + 144 mW (pads)
Peak Throughput	74 GOp/s
Effective Throughput	55 GOp/s
Core Power-Efficiency	803 GOp/s/W

^a Including the SRAM blocks

^b The power usage was measured running with real data and at max. load

Origami chạy ở hai miền clock với một clock có tần số gấp đôi miền còn lại:

- Miền clock chậm 250 MHz cho I/O và SRAM với các thành phần không cần phải chạy quá nhanh khác
- Miền clock nhanh 500 MHz cho SoP và ChSum.

Dataset là MNIST, CIFAR-10 và standford bg

DNN dùng để benchmark là:

- ConvNet đơn giản chạy trên MNIST
- VGG-16 chạy trên CIFAR-10

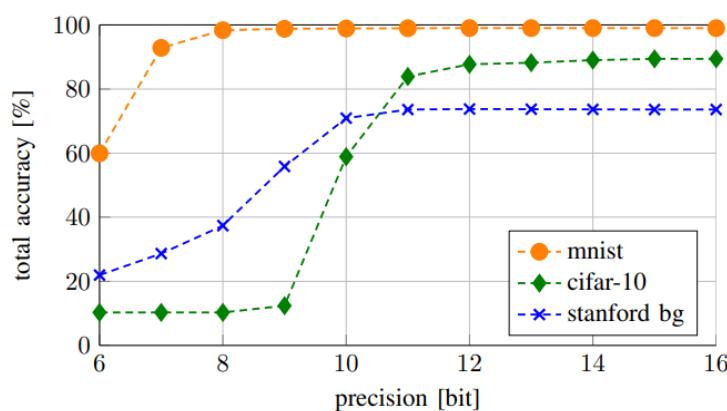
Kết quả thí nghiệm

Thí nghiệm 1: Kết quả tổng băng thông với ConvNet chạy trên hệ thống và độ hiệu quả của từng stage.

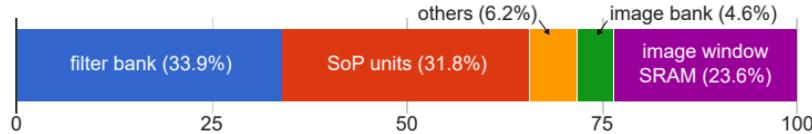
TABLE III
THROUGHPUT AND EFFICIENCY FOR THE INDIVIDUAL STAGES OF OUR
REFERENCE CONVOLUTIONAL NETWORK FOR 320×240 INPUT IMAGES.

stage # channels	Stage 1 (3→16)	Stage 2 (16→64)	Stage 3 (64→256)
η_{blocks}	0.38	1.00	1.00
$\eta_{filterLoad}$	0.99	0.98	0.91
η_{border}	0.96	0.91	0.82
η	0.36	0.89	0.75
throughput	71 GOp/s	174 GOp/s	147 GOp/s
# operations	0.35 GOp	1.68 GOp	5.43 GOp
run time	4.93 ms	9.65 ms	36.94 ms
Average throughput: 145 GOp/s → 19.4 frame/s @ 320×240			

Thí nghiệm 2: So sánh sự ảnh hưởng của fixed-point precision đối với độ chính xác dự đoán của mô hình ở các tập dataset khác nhau



Thí nghiệm 3: Kết quả phân tích về diện tích của Origami sau khi đã hoàn thành.

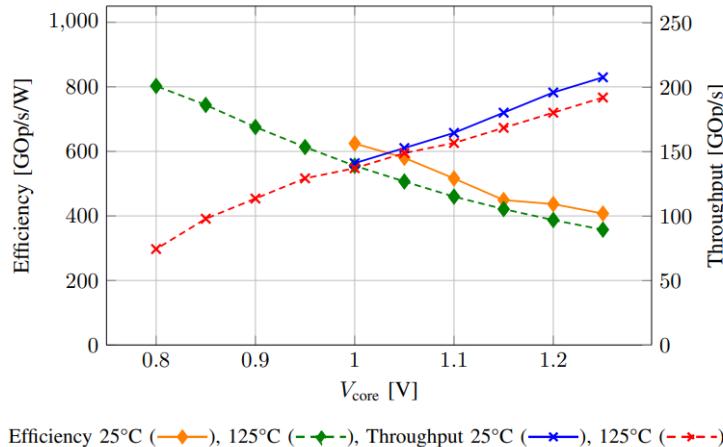


Thí nghiệm 4: Kết quả mô phỏng năng lượng sau khi đã layout ở các giá trị cài đặt thông số khác nhau của Origami (kích thước kernel, số channel)

TABLE IV
VARIOUS FILTER SIZES IN HARD- AND SOFTWARE, n_{ch} , AND THEIR
EFFECT ON POWER AND AREA EFFICIENCY

filter dim. hw	filter dim. sw	n_{ch}	rel. power eff.	rel. area eff.
7×7	7×7	8	100.0%	100.0%
5×5	5×5	8	92.6%	111.2%
3×3	3×3	8	89.1%	148.0%
3×3	3×3	16	131.9%	51.9%
7×7	3×3	8	32.8%	18.4%

Thí nghiệm 5: Kết quả đo lường mức độ hiệu quả năng lượng và băng thông ở các giá trị nguồn độc lập ở nhiệt độ $25^{\circ}C$ và $125^{\circ}C$



Thí nghiệm 6: Kết quả so sánh của Origami post-layout và post-synthesis và các công trình trước đó.

- L. Cavigelli, M. Magno, and L. Benini, “Accelerating Real-Time Embedded Scene Labeling with Convolutional Networks,” in Proc. ACM/IEEE Des. Autom. Conf., 2015.
- S. Chintala, “convnet-benchmarks,” 2016. [Online]. Available: <https://github.com/soumith/convnet-benchmarks>
- C. Farabet, C. Poulet, J. Y. Han et al., “CNP: An FPGA-based processor for Convolutional Networks,” in Proc. IEEE Int. Conf. F. Program. Log. Appl., 2009, pp. 32–37.
- C. Farabet, B. Martini, B. Corda et al., “NeuFlow: A Runtime Reconfigurable Dataflow Processor for Vision,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Work., jun 2011, pp. 109–116.

- V. Gokhale, J. Jin, A. Dundar et al., “A 240 G-ops/s Mobile Coprocessor for Deep Neural Networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2014, pp. 682–687.
- C. Zhang, P. Li, G. Sun et al., “Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks,” in Proc. ACM Int. Symp. Field-Programmable Gate Arrays, 2015, pp. 161–170.
- W. Qadeer, R. Hameed, O. Shacham et al., “Convolution Engine : Balancing Efficiency & Flexibility in Specialized Computing,” in Proc. ACM Int. Symp. Comput. Archit., 2013, pp. 24–35.
- Z. Du, R. Fasthuber, T. Chen et al., “ShiDianNao: Shifting Vision Processing Closer to the Sensor,” in Proc. ACM/IEEE Int. Symp. Comput. Archit., 2015.
- P. H. Pham, D. Jelaca, C. Farabet et al., “NeuFlow: Dataflow vision processing system-on-a-chip,” in Proc. Midwest Symp. Circuits Syst., 2012, pp. 1044–1047.
- F. Conti and L. Benini, “A Ultra-Low-Energy Convolution Engine for Fast Brain-Inspired Vision in Multicore Clusters,” in Proc. IEEE Des. Autom. Test Eur. Conf., 2015.

TABLE VI
SUMMARY OF RELATED WORK FOR A WIDE RANGE OF PLATFORMS (CPU, GPU, FPGA, ASIC).

publication	type	platform	theor. ^a GOp/s	peak ^a GOp/s	act. ^a GOp/s	power ^b W	power eff. GOp/s/W	prec.	V _{core} V	area ^h MGE	area eff. ^g GOp/s/MGE
Cavigelli et al. [27]	CPU	Xeon E5-1620v2	118		35 ^d	230	0.15	float32			
Cavigelli et al. [27]	GPU	GTX780	3977	3030	1908 ^d	sd:200	14 ^f	float32			
cuDNN R3 [33]	GPU	Titan X	6600	6343		d:250	25.6 ^f	float32			
Cavigelli et al. [27]	SoC	Tegra K1	365	95	84 ^d	s:11	8.6	float32			
CNP [37]	FPGA	Virtex4	40	40	37	s:10	3.7	fixed16			
NeuFlow [24]	FPGA	Virtex6 VLX240T	160	160	147	s:10	14.7	fixed16			
nn-X [38]	FPGA	Zynq XC7Z045	227	227	200	s:8 d+m:4	25	fixed16			
Zhang et al. [40]	FPGA/HLS	Virtex7 VX485T	62	62		s:18.6	3.3	float32			
ConvEngine [44]	synth.	45nm	410			c:1.0	409	fixed10	0.9		
ShiDianNao [42]	layout	TSMC 65nm	194			c:0.32	606	fixed16		d ^g :4.86	39.9
NeuFlow [24]	layout	IBM 45nm SOI	1280	1280	1164	d:5	230	fixed16	1.0	d:38.46	33.3
NeuFlow [25]	layout	IBM 45nm SOI	320	320	294	c:0.6	490	fixed16	1.0	d:19.23	16.6
HWCE [41]	layout	ST 28nm FDSOI	37	37		c:0.18	206	fixed16	0.8		
HWCE [41]	layout	ST 28nm FDSOI	1	1		c:0.73m	1375	fixed16	0.4		
this work	silicon	umc 65nm	196	196	145 ^d	c:0.51 ^e	437	fixed12	1.2 c:0.91 d:2.16	90.7	
this work	silicon	umc 65nm	74	74	55 ^d	c:0.093 ^e	803	fixed12	0.8 c:0.91 d:2.16	34.3	

Thí nghiệm 7: Kết quả dự đoán về điện thế nguồn, năng lượng và độ hiệu quả tính toán của Origami và các công trình khác khi được hiện thực trên công nghệ 28nm. Các công trình bao gồm:

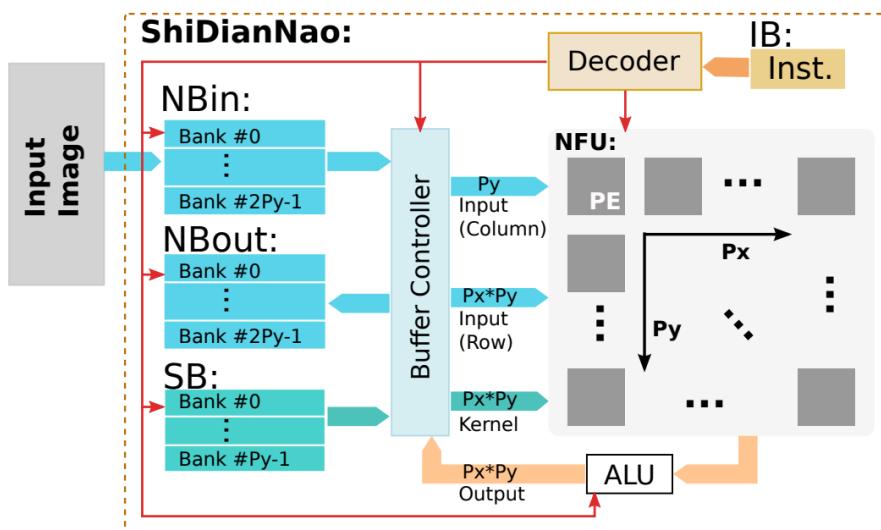
- W. Qadeer, R. Hameed, O. Shacham et al., “Convolution Engine : Balancing Efficiency & Flexibility in Specialized Computing,” in Proc. ACM Int. Symp. Comput. Archit., 2013, pp. 24–35.
- Z. Du, R. Fasthuber, T. Chen et al., “ShiDianNao: Shifting Vision Processing Closer to the Sensor,” in Proc. ACM/IEEE Int. Symp. Comput. Archit., 2015.
- C. Farabet, B. Martini, B. Corda et al., “NeuFlow: A Runtime Reconfigurable Dataflow Processor for Vision,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Work., jun 2011, pp. 109–116.
- F. Conti and L. Benini, “A Ultra-Low-Energy Convolution Engine for Fast Brain-Inspired Vision in Multicore Clusters,” in Proc. IEEE Des. Autom. Test Eur. Conf., 2015.

TABLE VII
PROJECTED POWER AND POWER-EFFICIENCY WHEN SCALED TO 28 NM

publication	V_{core} [V]	power [mW]	power eff. [GOp/s/W]
ConvEngine [44]	0.72	398	1030
ShiDianNao [42]	0.8	61.3	3167
NeuFlow [24]	0.8	239	1339
HWCE [41]	0.8	180	260
HWCE [41]	0.4	0.73	1375
this work	0.8	86.1	2276
this work	0.53	7.81	9475

1.11. ShiDianNao: Shifting Vision Processing Closer to the Sensor

Kiến trúc hệ thống

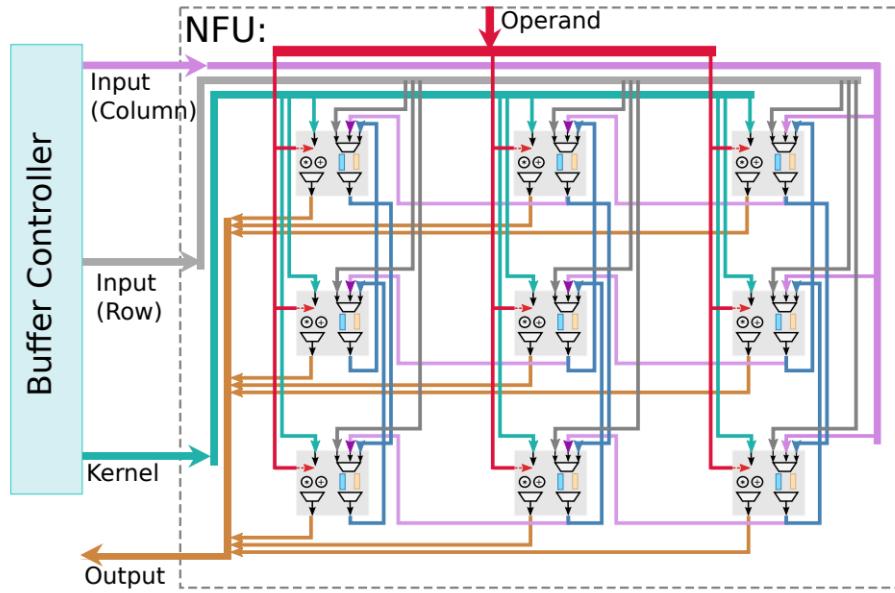


Hệ thống bao gồm:

- NBin và NBout lần lượt là hai buffer cho input và output activation
- SB là buffer cho weight
- Neural Functional Unit (NFU) cùng với Arithmetic Logic Unit (ALU) để thực hiện tính toán ở độ chính xác 16 bit fixed-point cho output activation
- Buffer và decoder (IB) cho các lệnh thực thi

A. Neural Functional Unit

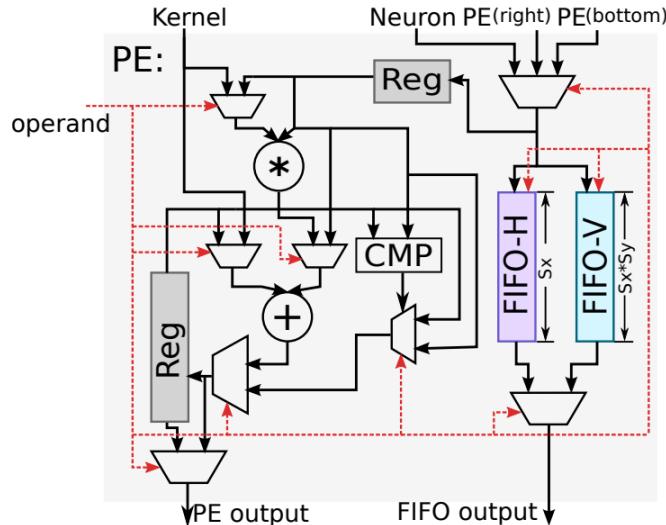
Kiến trúc của NFU:



NFU có thể đồng thời đọc weight và input activation từ NBin/NBout và SB, sau đó phân phối nó tới các PE khác nhau. Sau khi thực hiện tính toán, NFU thu thập kết quả từ các PE và gửi chúng tới NBout/NBin hoặc ALU.

B. Processing Element

Kiến trúc của PE



Mỗi PE bao gồm:

- 3 input:
 - 1 input cho việc nhận tín hiệu điều khiển
 - 1 input cho việc đọc các giá trị weight từ SB
 - 1 input cho việc đọc các giá trị input activation từ NBin/NBout, từ PE ở lân cận ngay bên phải hoặc bên dưới phụ thuộc vào tín hiệu điều khiển.

- 2 output:
 - 1 output cho việc ghi kết quả tính toán vào NBout/NBin
 - 1 output cho việc truyền giá trị input activation được lưu trữ nội bộ tới các PE lân cận

Ở mỗi chu kỳ, mỗi PE có thể thực hiện:

- Phép nhân và cộng cho convolution, fully-connected hoặc normalization layer
- Chỉ phép cộng cho average pooling layer
- Chỉ phép so sánh cho max pooling layer

Để phục vụ việc truyền nhận dữ liệu giữa các PE trong NFU, hệ thống sử dụng 2 FIFO (horizontal FIFO-H và vertical FIFO-V) ở trong mỗi PE để tạm thời lưu trữ giá trị input nhận được.

- FIFO-H buffer cho dữ liệu từ NBin/NBout và từ PE lân cận bên phải
- FIFO-V buffer cho dữ liệu từ NBin/NBout và từ PE lân cận bên trên

C. Arithmetic Logic Unit (ALU)

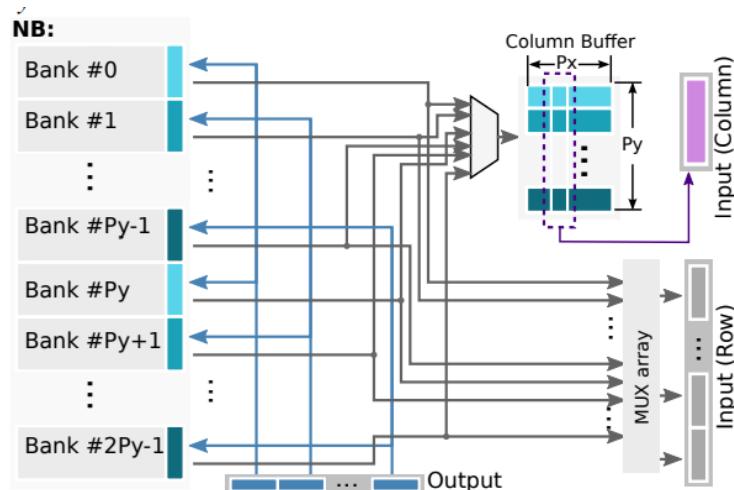
ALU hiện thực các phép toán 16 bit fixed-point bao gồm phép chia và các non-linear activation function (sigmoid, tanh, ...)

D. Storage

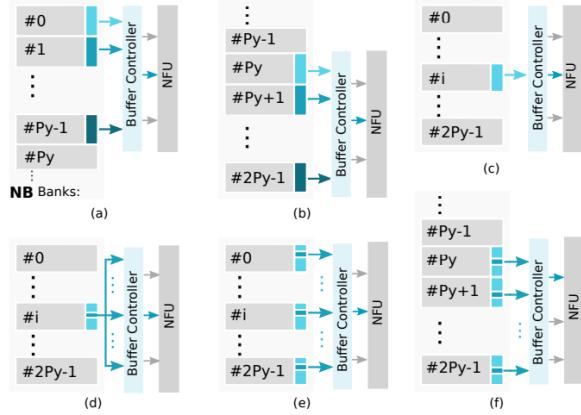
On-chip SRAM được dùng để lưu trữ tất cả dữ liệu và các lệnh cho CNN. On-chip SRAM sau đó được phân chia thành các buffer riêng biệt (NBin, NBout và SB) cho các loại dữ liệu khác nhau (input activation, output activation và kernel weight)

E. Control

Kiến trúc của Buffer Controller



Readmodes của NB Controller



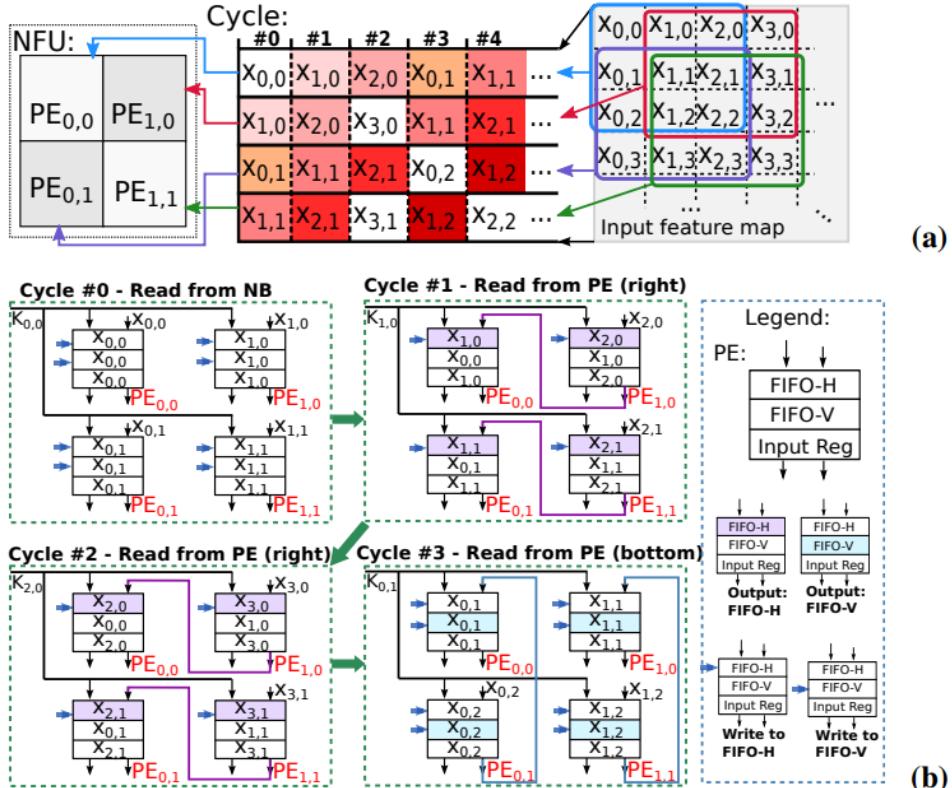
Bao gồm 6 loại:

- Đọc cùng lúc nhiều bank (từ #0 tới #Py - 1)
- Đọc cùng lúc nhiều bank (từ #Py tới #2Py - 1)
- Đọc một bank
- Đọc một neuron riêng lẻ
- Đọc các neuron cách nhau với giá trị cho trước
- Đọc một neuron cho từng bank (từ #0 tới #Py - 1 hoặc từ #Py tới #2Py - 1)

F. Data mapping

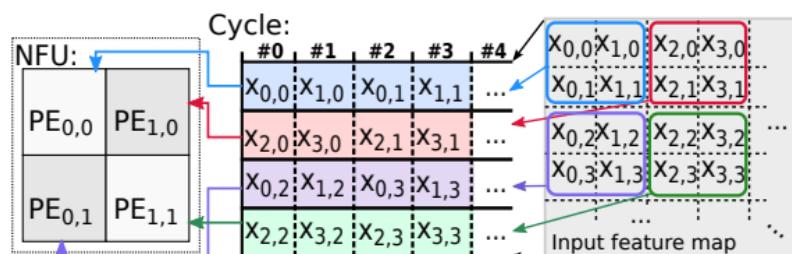
Convolution Layer

Ý tưởng: Ở layer này, hệ thống liên tục thực hiện tính toán cho một output feature map và không chuyển sang output feature map tiếp theo cho tới khi tính xong feature map hiện tại.



Pooling Layer

Ý tưởng: Khi ở layer này, việc tính toán tương tự như Convolution layer, hệ thống liên tục thực hiện tính toán cho một output feature map và không chuyển sang output feature map tiếp theo cho tới khi tính xong feature map hiện tại. Với mỗi tính toán cho từng output feature map, các PE được gắn cho một đơn vị output feature map và chỉ thực hiện phép toán trên đó tới khi hoàn thành.



Fully-Connected Layer

Ý tưởng: Khi ở layer này, các PE được gắn cho một đơn vị output feature map và chỉ thực hiện phép toán trên đó tới khi hoàn thành mà không chuyển sang đơn vị khác.

// Đánh giá hệ thống //

Chuẩn bị thí nghiệm

Hệ thống được hiện thực bằng ASIC theo công nghệ TSMC 65 nm.

Các hệ thống khác được so sánh:

- CPU: Intel Xeon (E7-8830, 2.13 GHz, 1 TB bộ nhớ)

- GPU: NVIDIA K20M, 5 GB GDDR5, 3.52 TFlops
- Accelerator: kết quả trước của bài báo này DianNao

T. Chen, Z. Du, N. Sun, J. Wang, and C. Wu, "DianNao: a small-footprint high-throughput accelerator for ubiquitous machinelearning," in Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Salt Lake City, UT, USA, 2014, pp. 269–284.

DNN được dùng:

Table 1: CNNs.

CNN	Largest Layer Size (KB)	Synapses Size (KB)	Total Storage (KB)	Accuracy (%)
CNP [46]	15.19	28.17	56.38	97.00
MPCNN [43]	30.63	42.77	88.89	96.77
Face Recogn. [33]	21.33	4.50	30.05	96.20
LeNet-5 [35]	9.19	118.30	136.11	99.05
Simple conv. [53]	2.44	24.17	30.12	99.60
CFF [17]	7.00	1.72	18.49	—
NEO [44]	4.50	3.63	16.03	96.92
ConvNN [9]	45.00	4.35	87.53	96.73
Gabor [30]	2.00	0.82	5.36	87.50
Face align. [11]	15.63	29.27	56.39	—

Kết quả thí nghiệm

Thí nghiệm 1: Kết quả về đặc tính vật lý (diện tích, công suất và năng lượng) và cài đặt các tham số cho hệ thống.

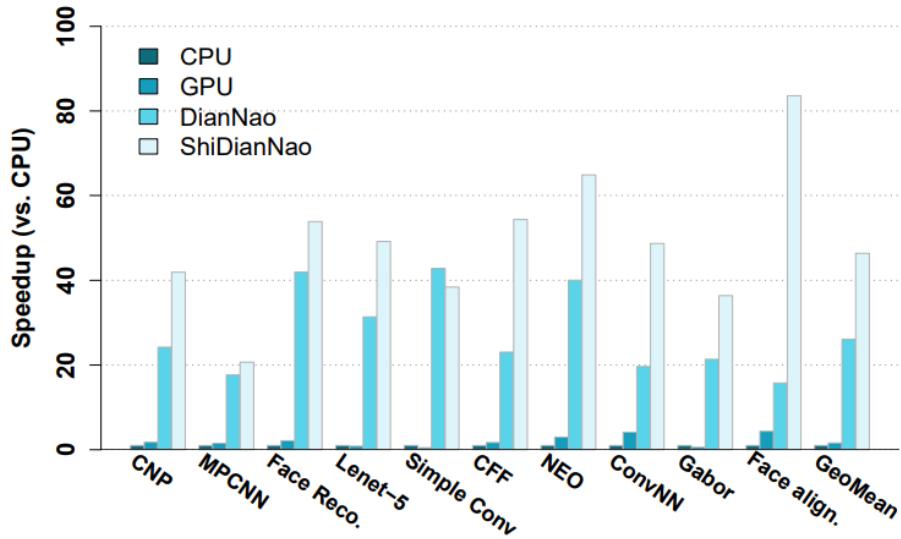
Table 3: Parameter settings of ShiDianNao and DianNao.

	ShiDianNao	DianNao
Data width	16-bit	16-bit
# multipliers	64	64
NBin SRAM size	64 KB	1 KB
NBout SRAM size	64 KB	1 KB
SB SRAM size	128 KB	16 KB
Inst. SRAM size	32 KB	8 KB

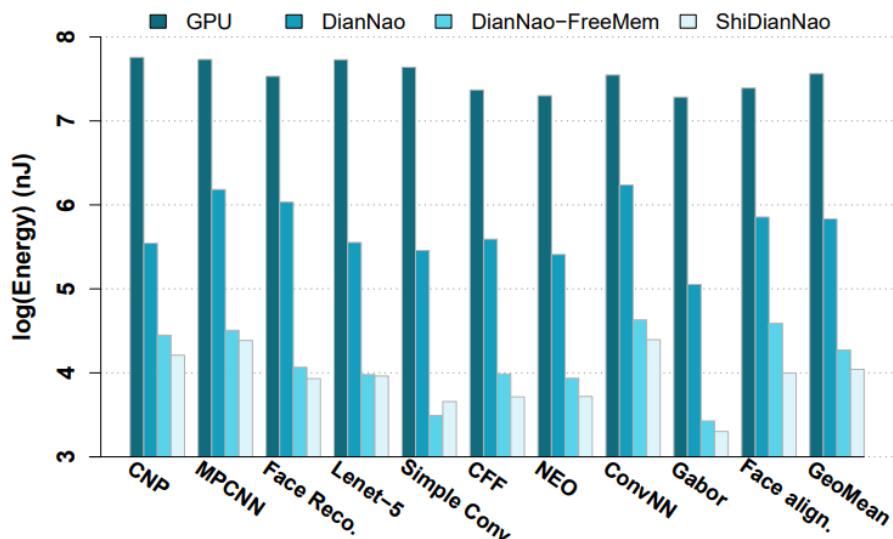
Table 4: Hardware characteristics of ShiDianNao at 1GHz, where power and energy are averaged over 10 benchmarks.

Accelerator	Area (mm^2)	Power (mW)	Energy (nJ)
Total	4.86 (100%)	320.10 (100%)	6048.70 (100%)
NFU	0.66 (13.58%)	268.82 (83.98%)	5281.09 (87.29%)
NBin	1.12 (23.05%)	35.53 (11.10%)	475.01 (7.85%)
NBout	1.12 (23.05%)	6.60 (2.06%)	86.61 (1.43%)
SB	1.65 (33.95%)	6.77 (2.11%)	94.08 (1.56%)
IB	0.31 (6.38%)	2.38 (0.74%)	35.84 (0.59%)

Thí nghiệm 2: Kết quả so sánh về tốc độ tính toán của hệ thống với CPU, GPU và DianNao khi thực hiện benchmark với các DNN ở phần trước.



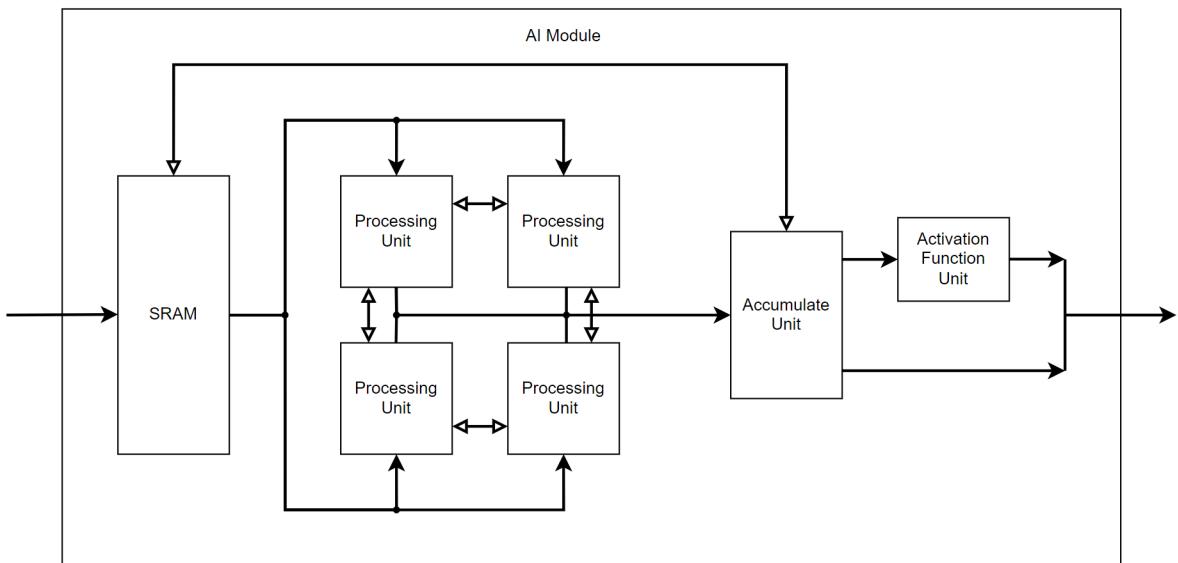
Thí nghiệm 3: Kết quả so sánh về năng lượng tiêu thụ của hệ thống với CPU, GPU và DianNao khi thực hiện benchmark với các DNN ở phần trước.



2. Kiến trúc hệ thống

Ý tưởng: Hệ thống lấy ý tưởng từ 1.2 UNPU với việc tinh chỉnh cho hệ thống gọn nhẹ hơn để đảm bảo việc năng lượng tiêu thụ < 150mW. Hệ thống xây dựng dataflow rõ ràng hơn ở convolution layer và thêm các phần cứng hỗ trợ cho việc tái sử dụng input activation và weight thông qua chia sẻ giữa các core với nhau.Thêm nữa, khảo sát việc sử dụng các loại MAC khác nhau ở hệ thống để xem xét mức độ hiệu quả.

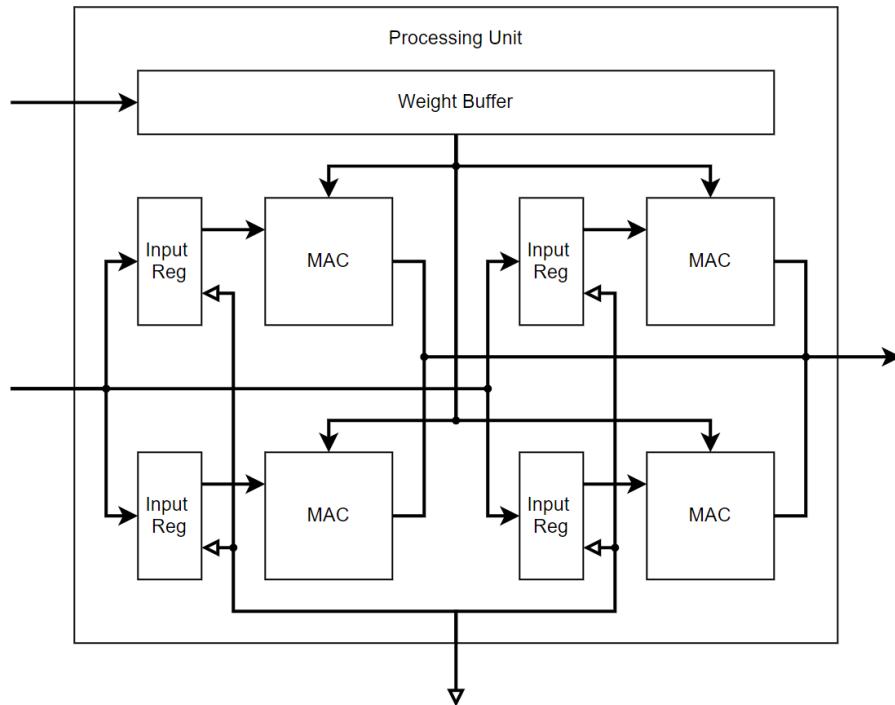
Kiến trúc đề xuất



Hệ thống bao gồm 4 thành phần chính:

- SRAM đóng vai trò lưu trữ các giá trị weight, input feature map (input activation) và bias. SRAM phân phối các giá trị này tới cho dãy các Processing Unit và Accumulate Unit.
- Processing Unit đóng vai trò thực hiện tính toán các phép toán cộng và nhân trong mạng Deep Learning. Đầu vào của Processing Unit là các weight và input feature map (input activation) từ SRAM. Các giá trị sau khi tính toán của Processing Unit được gửi tới cho Accumulate Unit để thực hiện cộng tích lũy các giá trị. Ngoài ra, giữa các Processing Unit còn có thể giao tiếp với nhau để chia sẻ giá trị input feature map phục vụ Convolutional layer.
- Accumulate Unit đóng vai trò cộng tích lũy giá trị một phần khi tính toán cho output feature map (output activation) và giá trị bias tương ứng.

2.1. Processing Unit



Processing Unit bao gồm:

- Weight Buffer: lưu trữ các giá trị weight từ SRAM và phân phối tới dãy các MAC.
- Input Reg: lưu trữ các giá trị của input feature map (input activation) và phân phối tới MAC tương ứng. Mỗi Input Reg được liên kết với một MAC. Ngoài ra, các Input Reg này có liên kết với các Input Reg ở Processing Unit phục vụ mục đích chia sẻ dữ liệu input feature map (input activation) đối với Convolutional layer.
- MAC: thực hiện tính toán phép nhân và cộng tích lũy ($a_0 \times b_0 + a_1 \times b_1 + a_2 \times b_2 + \dots$) từ các giá trị weight và input feature map (input neuron).

2.2. MAC

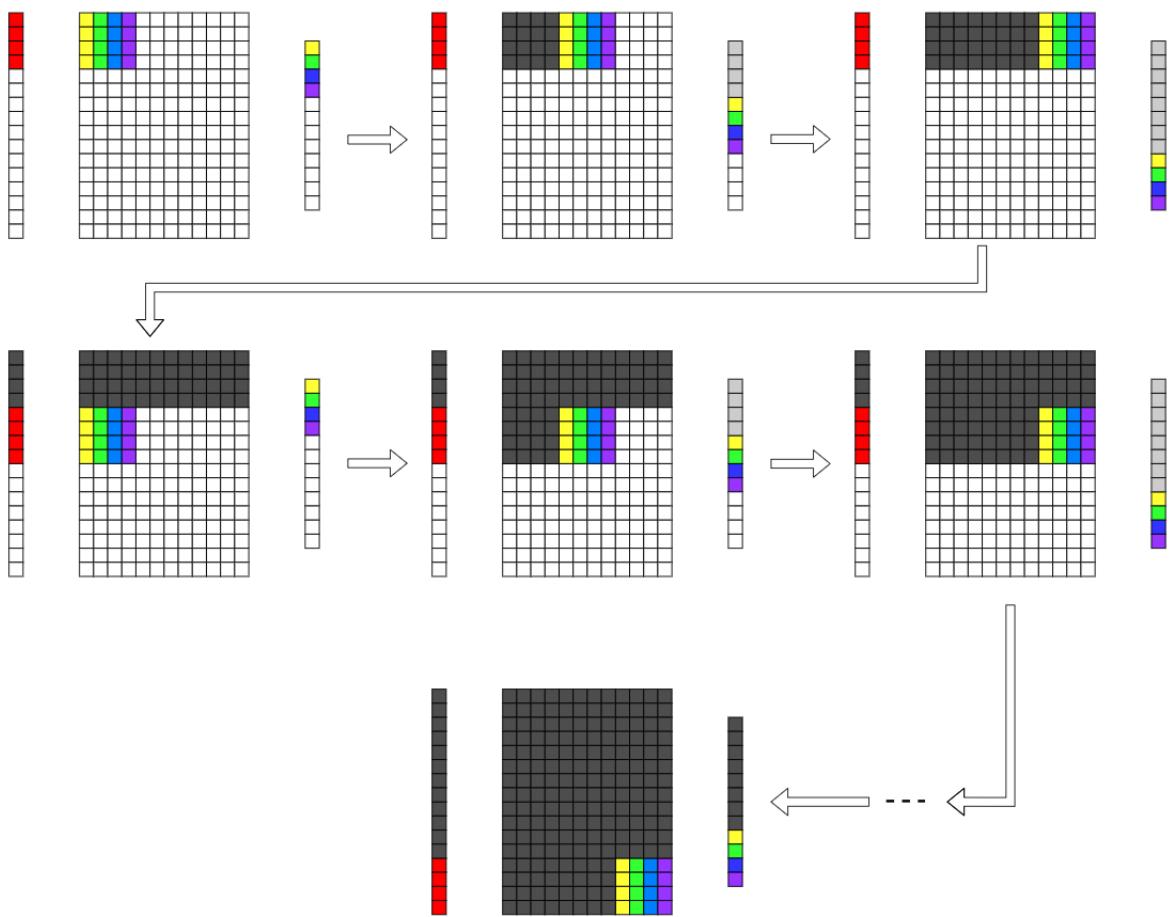
MAC có thể dùng một trong ba loại sau:

- Variable-Bit-Precision MAC (tham khảo ở 1.2.C)
- Stochastic MAC (tham khảo ở 1.3.A)
- Log-quantization MAC (tham khảo ở 1.9.A, 1.9.B và 1.9.C)

2.3. Data flow

A. Fully-Connected Layer

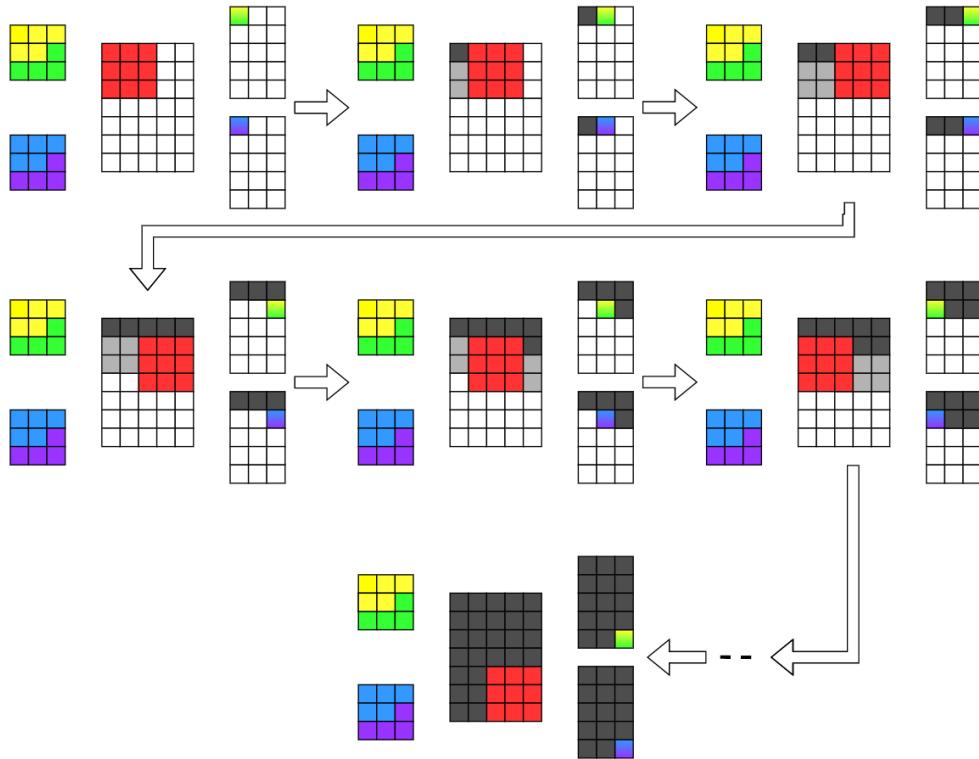
Ở Fully-Connected Layer, các tác vụ bộ nhớ áp đảo hoàn toàn so với các tác vụ tính toán. Nguyên nhân của việc này là do số lượng weight cực kỳ lớn. Mô hình tính toán được đề xuất như sau:



Mục đích của flow này để khai thác tối đa việc tái sử dụng input feature map.

B. Convolution Layer

Ở Convolution Layer, các tác vụ tính toán áp đảo hoàn toàn so với các tác vụ bộ nhớ. Mô hình tính toán được đề xuất như sau:



Mục đích của flow này để khai thác tối đa việc tái sử dụng input feature map.

3. Kế hoạch thực hiện

Tổng thời gian của một luận văn gồm 15 tuần, có 3 phần chính:

- Hiện thực các khối của module dạng RTL và đảm bảo testbench tương ứng: 4 tuần
- Nhúng module vào SoC và chạy testbench để đảm bảo về khả năng giao tiếp và hoạt động: 2 tuần
- Chỉnh sửa RISC-V compiler và các Deep Learning framework: 3 tuần
- Benchmark hệ thống: 2 tuần

Dự trù thời gian: 4 tuần.