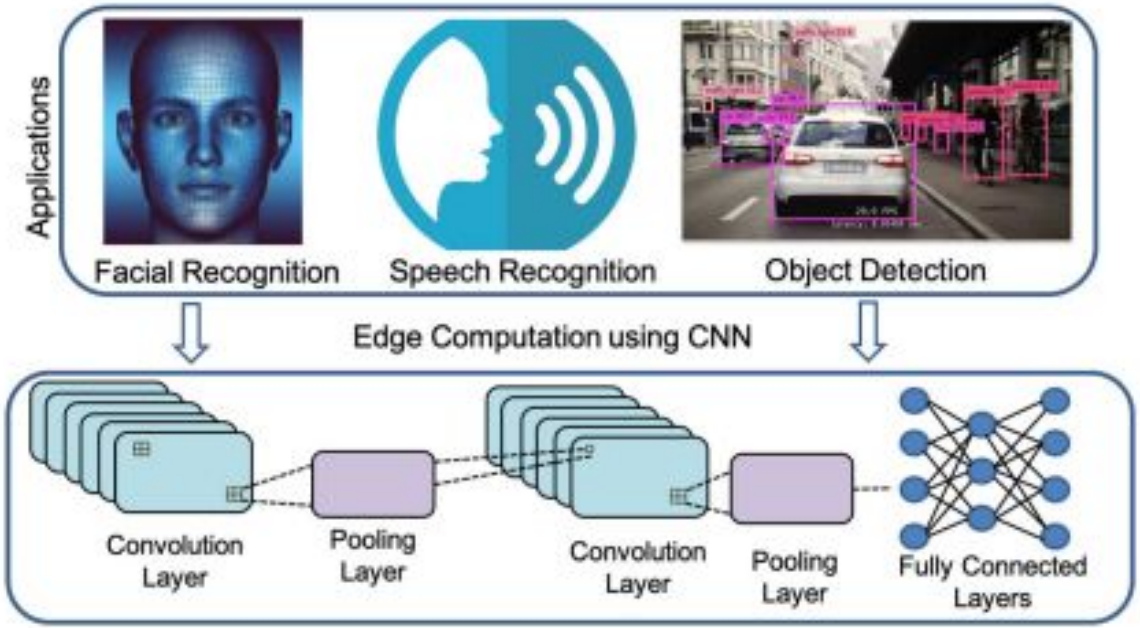
EE 617: VLSI Design

An SRAM-Based Hybrid Computation-in-Memory Macro Using Current-Reused Differential CCO

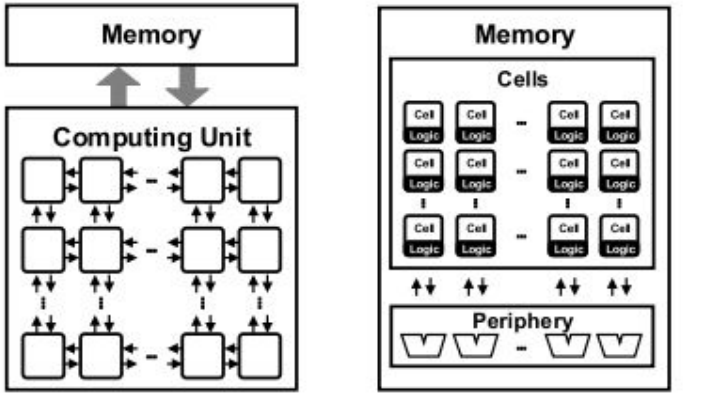
Course Instructor: Prof. Joycee M. Mekie
TA: Kailash Prasad

Submitted By: Prateek Sharma (22250026)
Daniel Giftson E (20110051)

Introduction



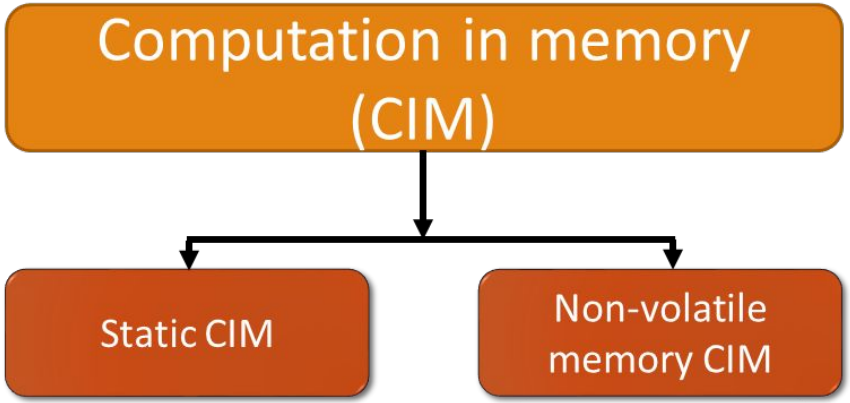
Introduction



Von Neumann Arch. Processing-in-Memory Arch.

- Traditional CNN engines, based on Von Neumann Architecture consumes more power and less energy efficiency due to the data movement between the processor and memory, known as **memory wall problem.**
- To overcome this problem, **Computation In Memory (CIM)** Architecture provides a promising solution by processing data in memory without data movement.
- Achieves high throughput and energy efficiency by enabling parallel data processing.

Introduction



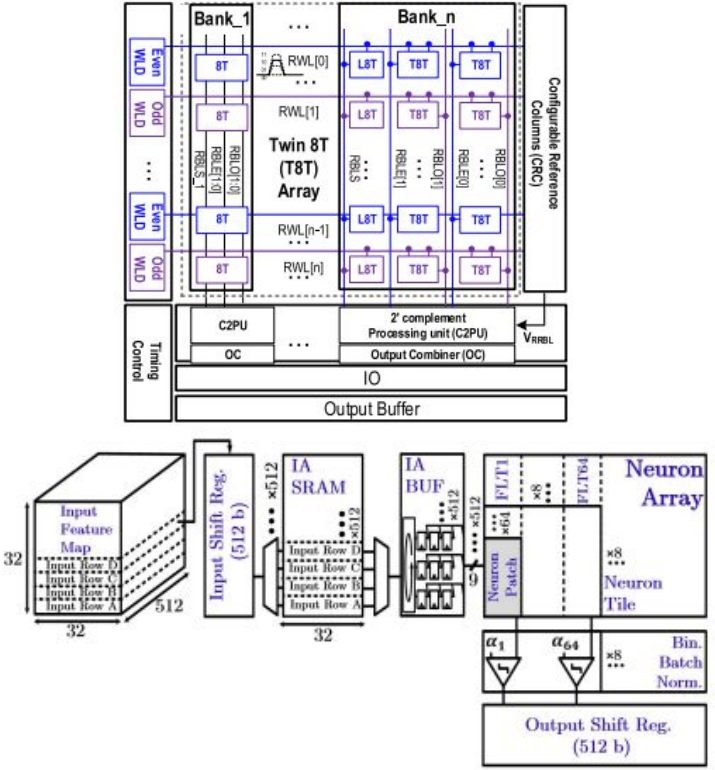
- Even though NVM-CIM can keep the data even when powered off, the **cost of the write operation** is much higher in non-volatile memory.
- Although SRAM-CIM is volatile in nature, it provides faster write speeds and lower write energy than NVM-CIM.
- It also shows good-compatibility with state-of-the-art CMOS Logic Technology, thus making it easy to scale down latency and increase energy efficiency.

Challenges

- Even though NVM-CIM can keep the data even when powered off, the **cost of the write operation** is much higher in non-volatile memory.
- Although SRAM-CIM is volatile in nature, it provides faster write speeds and lower write energy than NVM-CIM.
- It also shows good-compatibility with state-of-the-art CMOS Logic Technology, thus making it easy to scale down latency and increase energy efficiency.

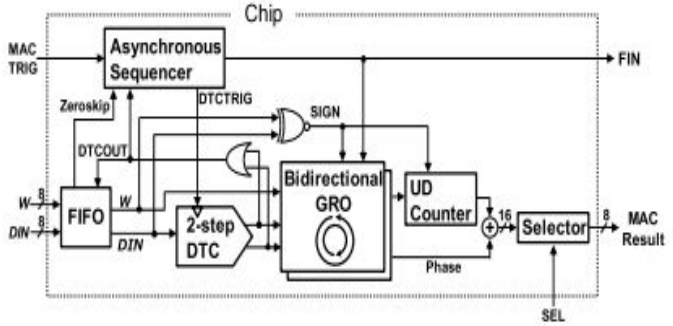
Analog-Domain Computing

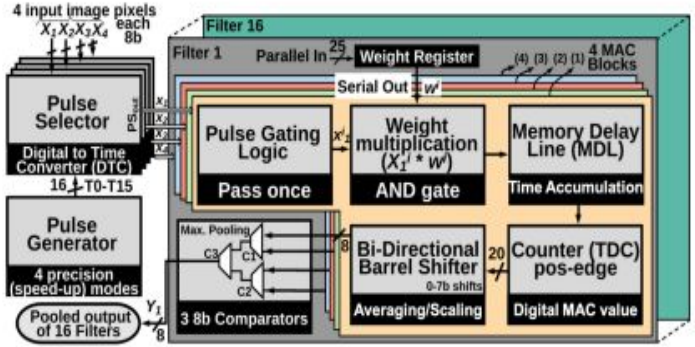
- In general, analog-domain computing is a key design technique to achieve **high energy efficiency**
- Some of the initial research works in analog-computing are as follows:
  - Current domain computing based on twin 8T SRAM cell arrays to prevent write disturbances and to improve accuracy.
  - A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute which achieves a high computational SNR by using charge-domain computation based on metal-oxide-metal (MOM) capacitors.
- **Advantages:** Performs energy-efficient MAC Operations
- **Disadvantages:** MAC results must be digitized by ADCs which has high power and area overhead.



Analog-Domain Computing

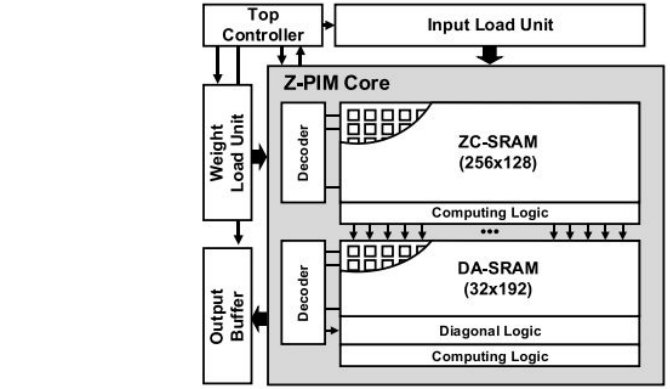
- To reduce the overhead of ADCs, some of the time and phase-domain computing techniques were proposed:
  - An 8 bit 12.4 TOPS/W phase-domain MAC circuit for energy-constrained deep learning accelerators in which partial MAC values are continuously accumulated in a gated-ring oscillator(GRO) and the phase generated is sampled once by a readout logic at the end.
  - A 12.08-TOPS/W all-digital time-domain CNN engine using bidirectional memory delay lines for energy efficient edge computing in which a time-domain MAC computing circuit is proposed to implement a CNN engine without CCO-based ADCs.
- **Advantages:** Produce high-precision digital outputs while minimizing area and power consumption.
- **Disadvantages:** Low throughput because the inputs are serially fed to the MAC Operation circuit.

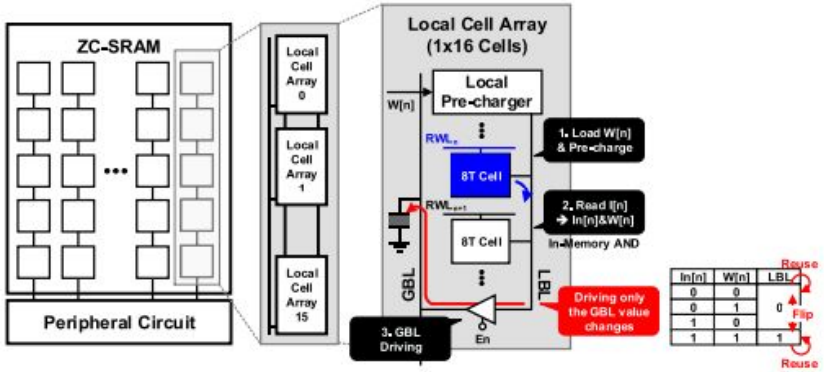




Digital-Domain Computing

- Digital-domain computing techniques are proposed to achieve **high accuracy**.
- One of the works done as part of digital-domain computing was “A Sparsity-Aware Processing-in-Memory Architecture With Fully Variable Weight Bit-Precision for Energy-Efficient Deep Neural Networks”.
- This implemented a zero-skipping convolutional SRAM to perform energy-efficient in-memory operations and **a charge reuse scheme** to decrease energy consumption.
- **Advantages:** High accuracy MAC Operations
- **Disadvantages:** Lesser energy efficiency compared to its analog counterpart.

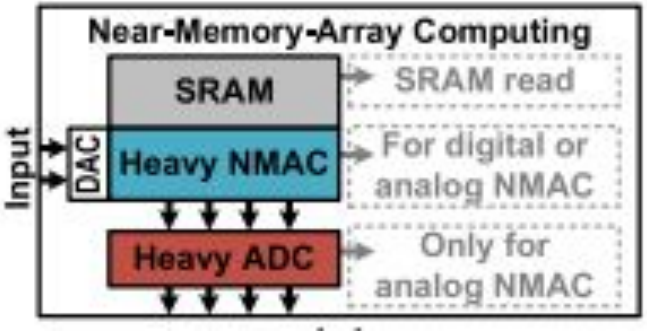




Summary of Computing techniques

- These computing techniques have trade-offs among **computation energy, dynamic range, read accuracy, and area efficiency.**
- Therefore, this paper proposes to employ a phase- and digital-domain computing to minimize the power overhead of ADCs and improve the robustness of IMAC.
  - The low throughput problem in phase-domain computing is solved by the proposed **PNMAC** that performs parallel summation and accumulation.
  - The energy efficiency problem in digital-domain computing is solved by the proposed **DIMAC** which does bit-serial digital computing that reduces complexity.

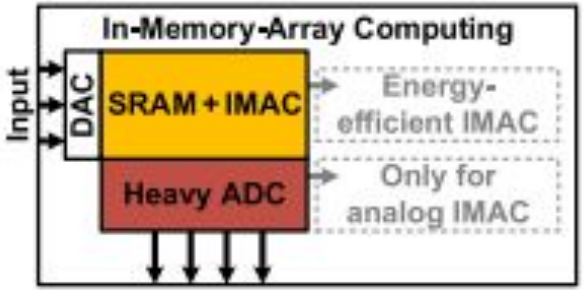
Near-Memory-Array Computing



Advantages: Multi-row READ access in SRAM and NMAC proves to be highly robust to PVT variations.

Disadvantages: Heavy ADCs required to convert the analog MAC results to digital.

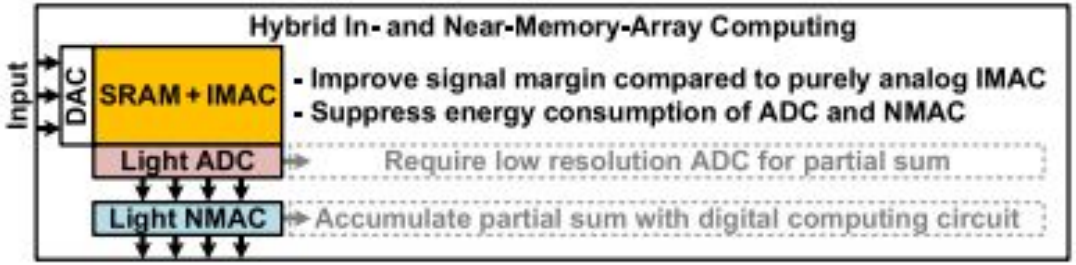
In-Memory-Array Computing



Advantages: Fully parallel bit-wise multiplication thus increasing the energy efficiency and throughput.

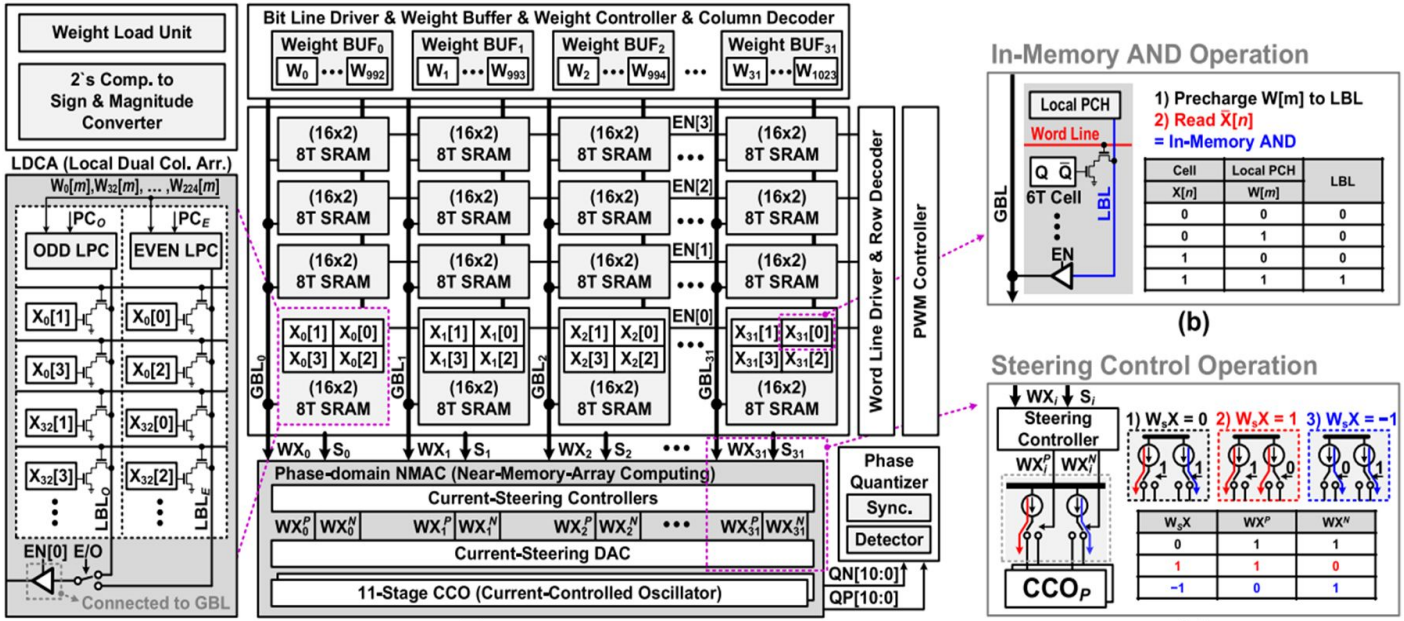
Disadvantages: Limited output precision due to the limited analog signal margin for multi-bit MAC operations

Hybrid In-and Near-Memory-Array Computing



- Partial MAC operations are performed in digital-domain and the remaining MAC operations are performed in analog-domain.
- Allows the ADC to digitize the partial MAC with a sufficient signal margin that ensures robust ADC output accuracy.
- Since still the ADC consumes a large portion of the total power and since the ADC is required for each partial MAC operation conversion, they propose a hybrid CIM macro that accumulates partial MAC results in **phase domain.**
- Therefore **ultra-low power consumption** and **wide-dynamic range** analog-to-digital conversion.

Proposed SRAM Based Hybrid Architecture

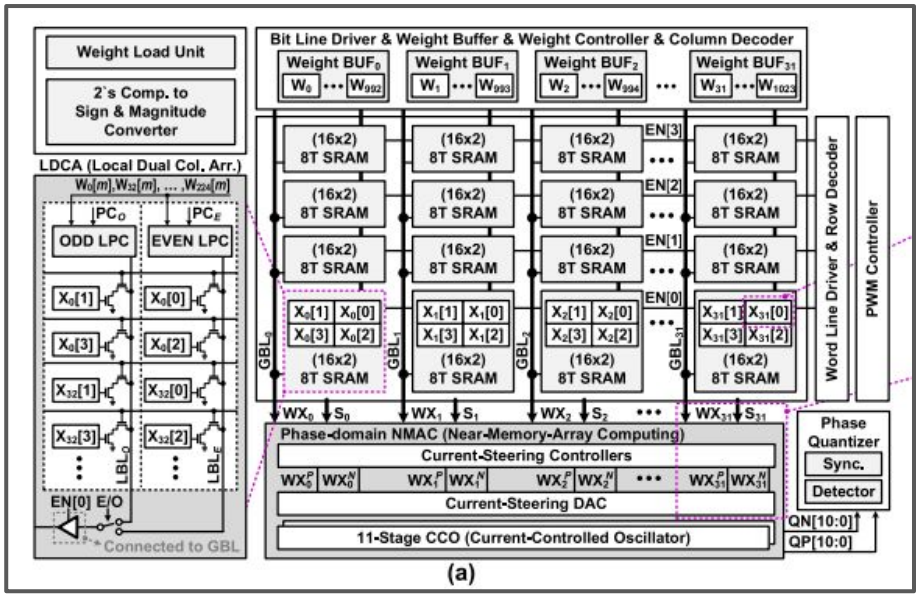


The proposed SRAM-based Hybrid CIM consists of two main parts:

DIMAC: performs energy-efficient in-memory bit-wise multiplications & generates pulse-width-modulated (PWM) signals

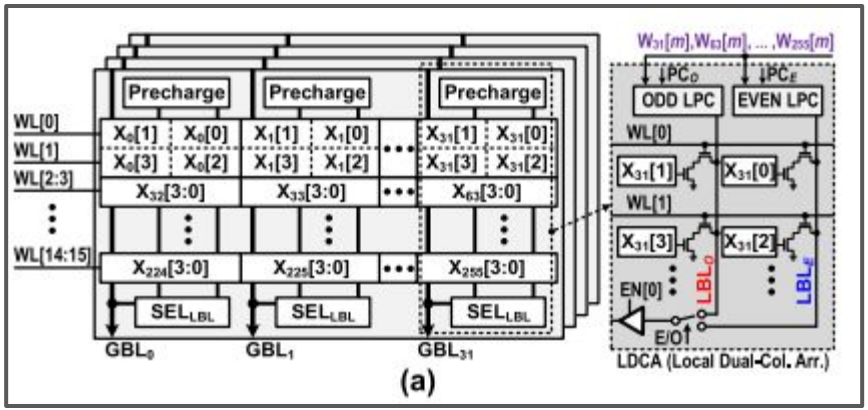
PNMAC: parallely accumulates the partial MAC results continuously & convert the accumulated results into digital data once at the end with ultra-low power consumption.

DIMAC Architecture



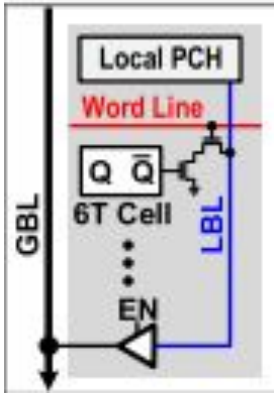
- Each two-column pair of the 64x64 SRAM are divided into 4 LDCAs (local dual-column array).
- Zero-skipping convolutional SRAM is used which is composed of 8T SRAM cells and hierarchical BL structure.
- Bit-wise multiplication or the AND operation of W[m] x X[n] is computed through the local precharger (LPC) and 8T SRAM cell.
- Each column has a global BL (GBL) which reads out the multiplication result from the local BLs (LBLs) through tri-state buffer enabled by EN[k].
- The weights are selected as an external source and represented in S&M format by S, W[2:0]
- Weight data set, W_{32r+i} where r = 0,1,2….31 is loaded to the weight buffer, BUF_i , by the Weight Load unit and further bit-wise serially loaded to their respective LDCA groups.

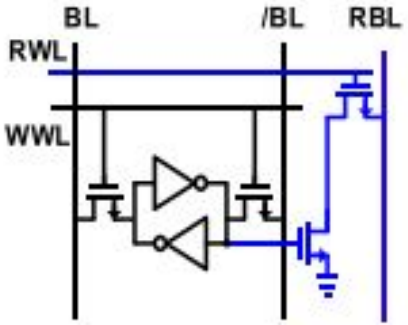
Circuitry and Working of LDCA



- Generates PWM voltage signals with the multiplication values, W[m] x X[3:0]
- Capacity of 32b (16 x 2)
- First LDCA can write and store inputs from X_0, X_{32},…..X_{224}
- The input, X[3:0] is written to the macro and stored in two columns by even and odd-bit positions.
- One of the even-bit or odd-bit columns is selected by the E/O signal, and EN[3:0] enables one of the LDCAs to read the LBL value to be read to GBL.
- Employs an overlapped precharging technique to increase the throughput of the DIMAC.

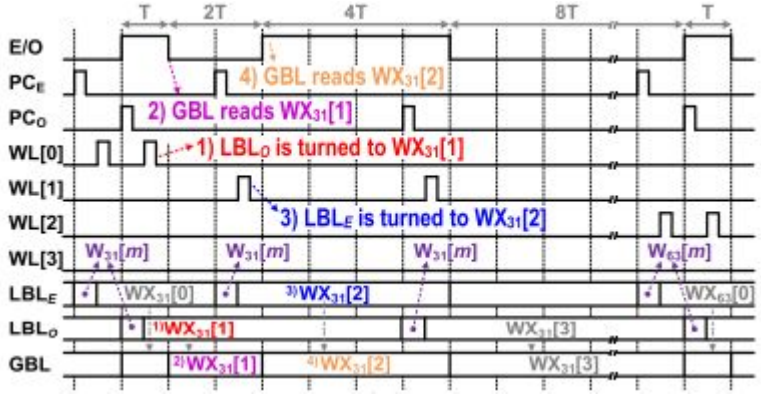
In-Memory AND Operation





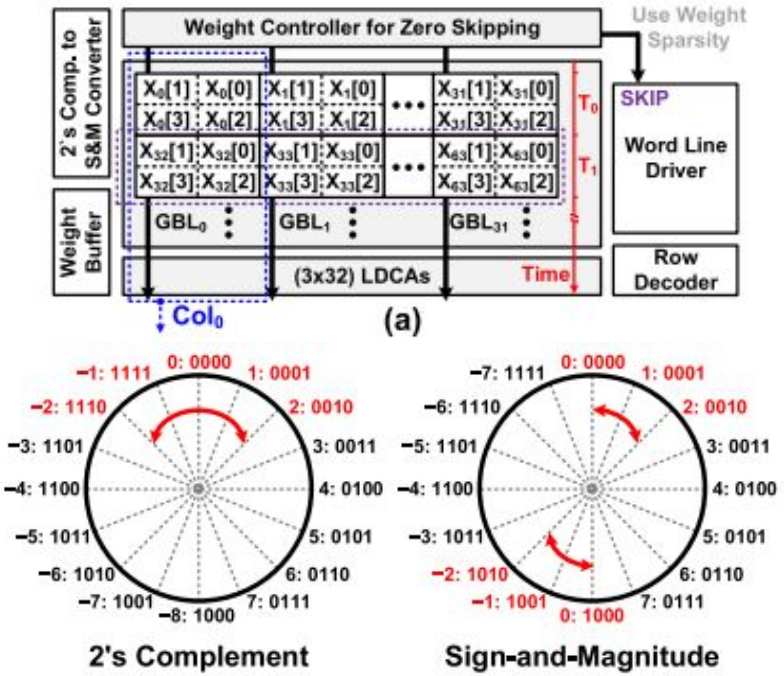
Cell	Local PCH	LBL
X[n]	W[m]	
0	0	0
0	1	0
1	0	0
1	1	1

Timing Diagram of PWM Signal Generation



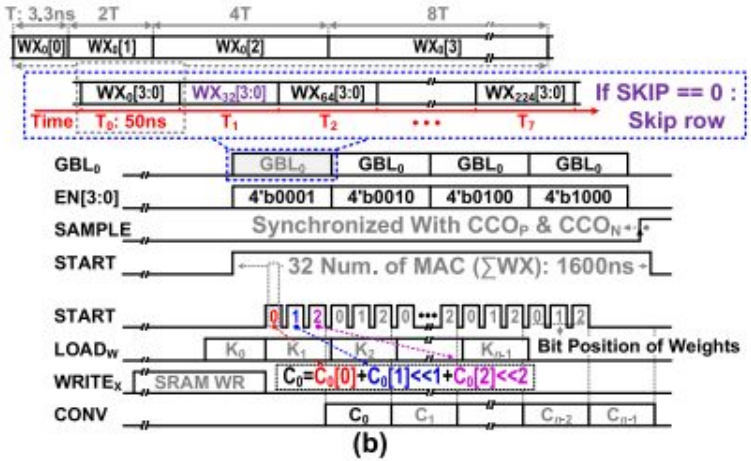
- Precharging of either LBL_E or LBL_O is done when PC_O or PC_E is high respectively.
- Driving the WL to 1 enables the in-memory and operation to be performed on the respective precharged LBL.
- Then, the E/O signal is flipped for the GBL is read the corresponding LBL.
- For example, if the bit position of X[n] is an odd number, LBL_O is turned to WX[n] while the GBL will still be connected to LBL_E . Then E/O signal is flipped to read LBL_O
- Increases the throughput by hiding the precharging time in the PWM signal.
- Therefore, DIMAC takes 15 cycles to produce the PWM signal of 1 MAC operation (W[m]X[3:0])

Zero-Skipping Operation



- Weight controller reads only the non-zero value in the fetched weights and forwards each bit to LPC.
- If the weights have zero values, the SKIP signal and GBLs are turned low to disable the Word Line Driver and Weight Driver.
- High Skip rate is to be achieved in order to lower power consumption and increase energy efficiency.
- As we can see the below fig, positive weights have high bit-level sparsity in both number systems, while negative weights have high bit-level sparsity only in the S&M system.
- Thus they employ the S&M number system for weights over the 2's complement system.

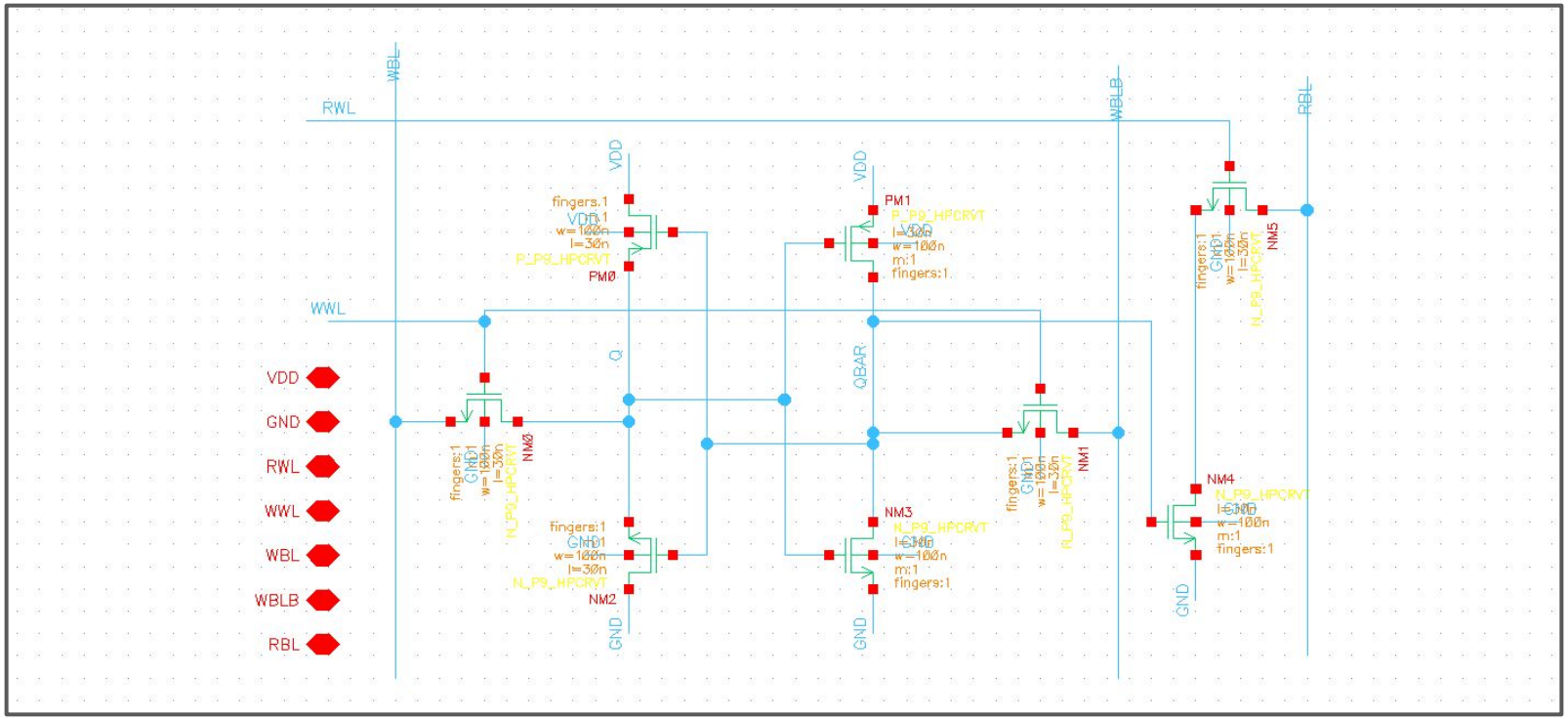
Overall Timing Diagram of MAC Operation with Multi Bit Weights



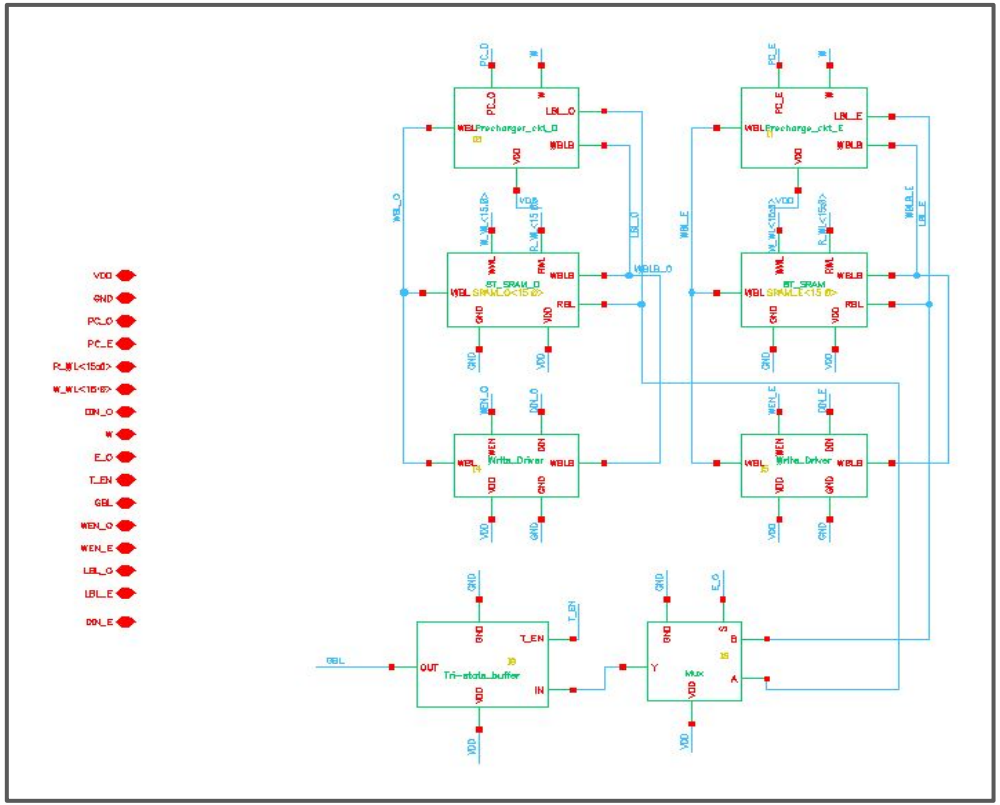
- Initially, the user defined inputs are stored in the SRAM cells by the writing in them.
- Then the first MAC operation, $C_0[0] = \Sigma W[0]X[3:0]$, starts after all the weights of K_0 are stored in the weight buffer.
- 32 MAC operations are performed sequentially in one LDCA group which constitutes to 480 (32 x 15) cycles, which is 1600 ns at an operating frequency of 300MHz.
- Since there are a total of 32 LDCA operating in parallel, there will be a total of 1024 MAC operations.
- Then, this MAC result of $C_0[0]$ is sampled by the readout circuit in PNMAC.
- Similarly, $C_0[1]$ and $C_0[2]$ are calculated in the same way and finally, the remaining MAC operations are performed in the digital domain to calculate the final MAC result as:

DIMAC Schematics, Waveforms and Total energy consumption

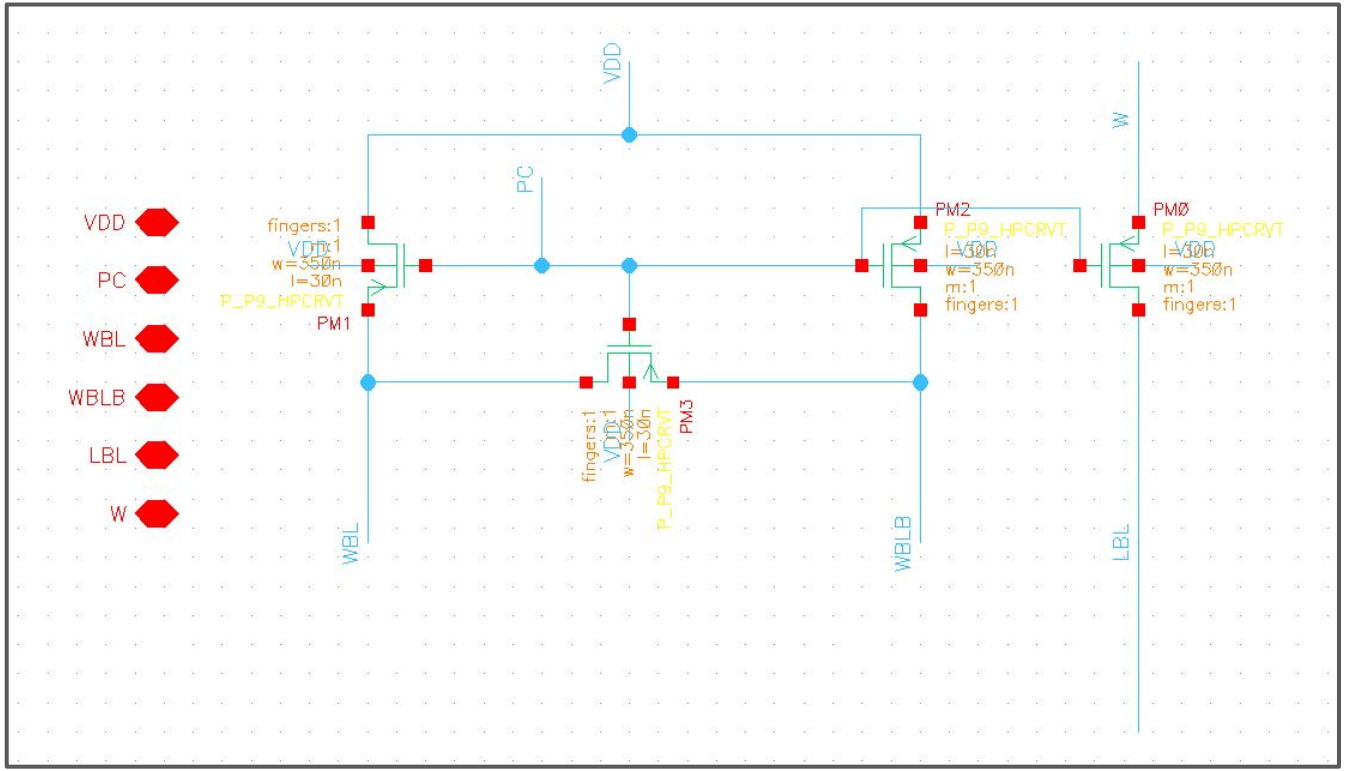
8T SRAM Cell


WBL
RWL
WBLB
RBL
VDD
WWL
Q
QBAR
GND
PM0
PM1
NM0
NM1
NM2
NM3
NM4
NM5
VDD
GND
RWL
WWL
WBL
WBLB
RBL

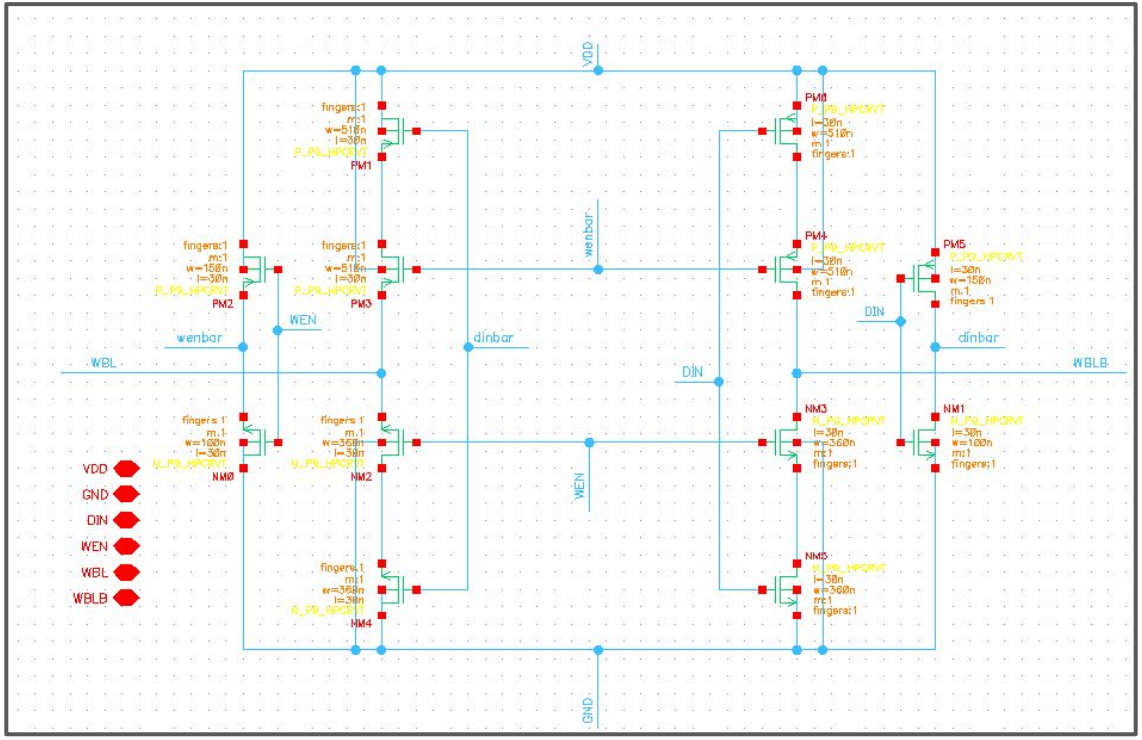
LDCA Schematic (16 x 2)



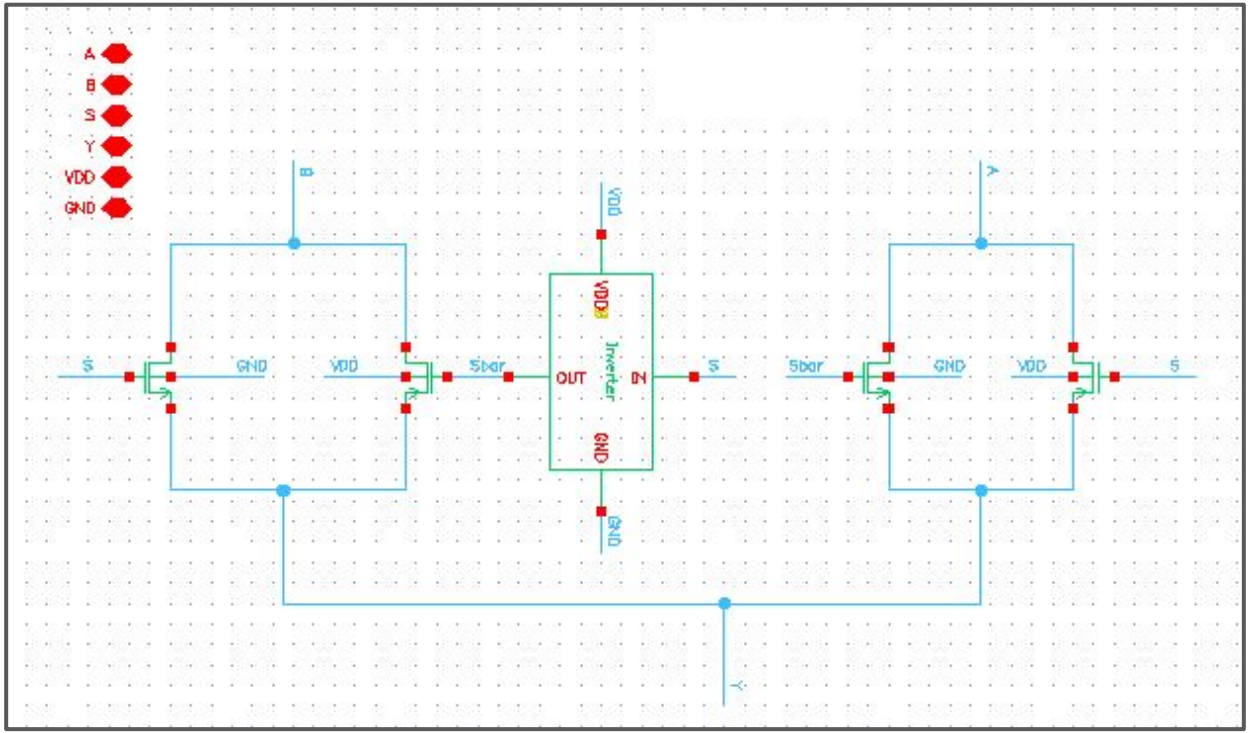
Local Precharger



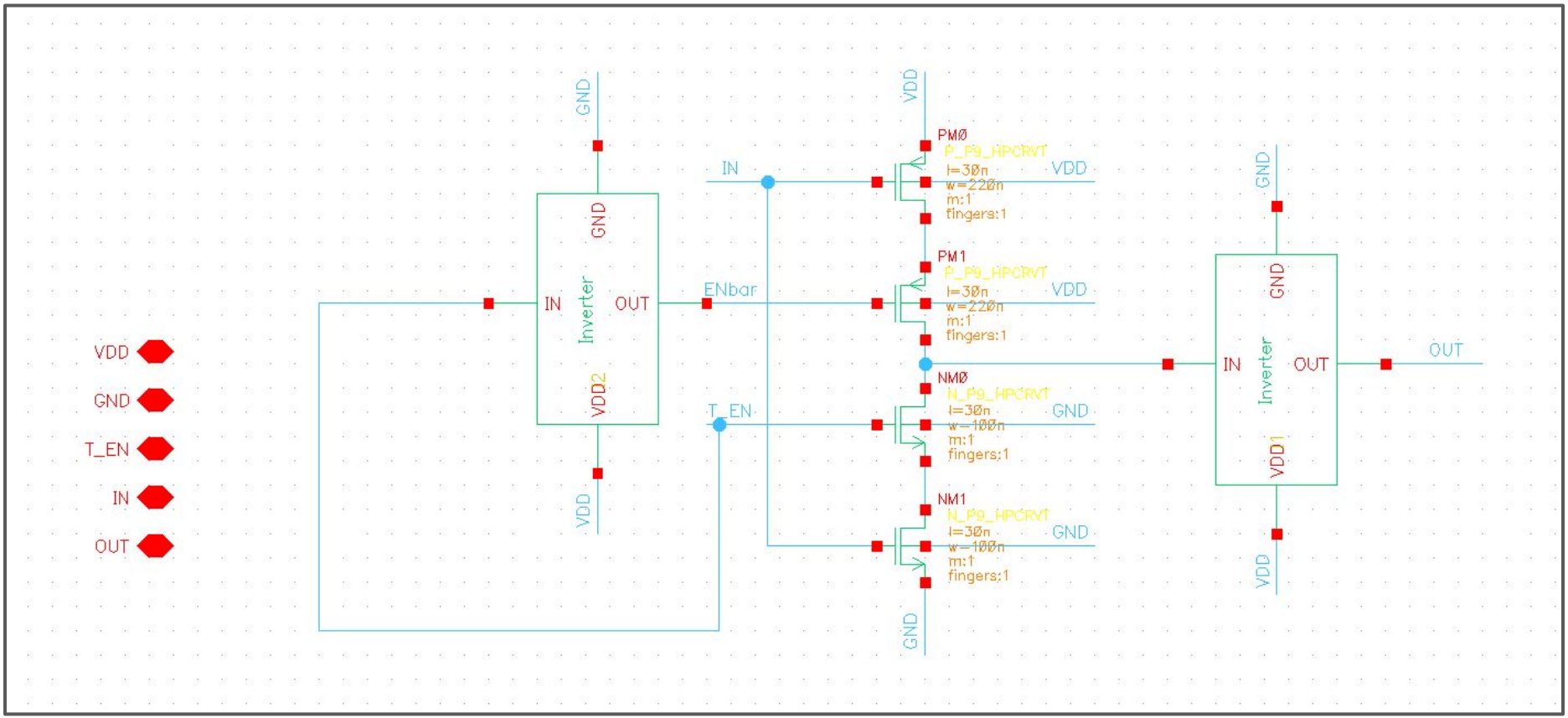
Write Driver



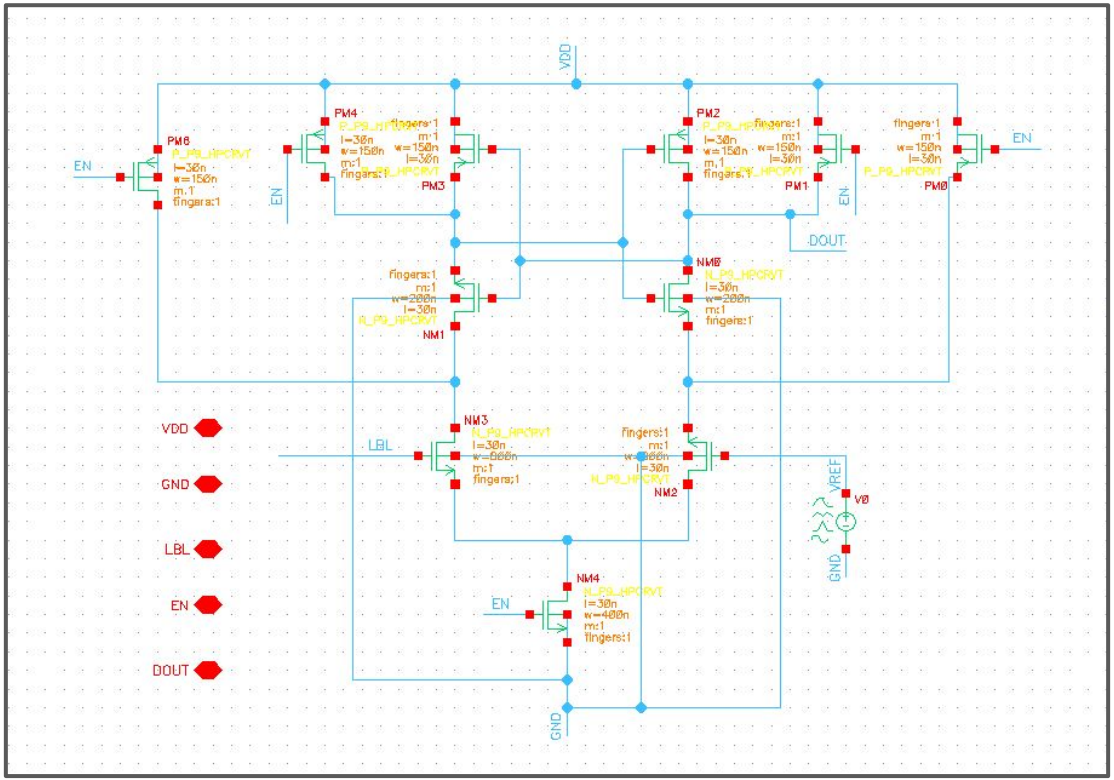
2:1 Multiplexer



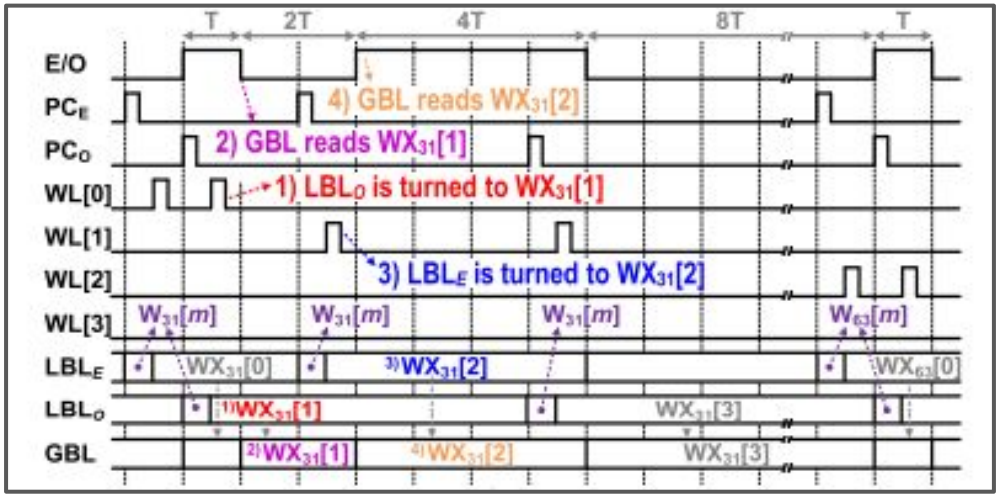
Tri-State Buffer



Sense - Amplifier



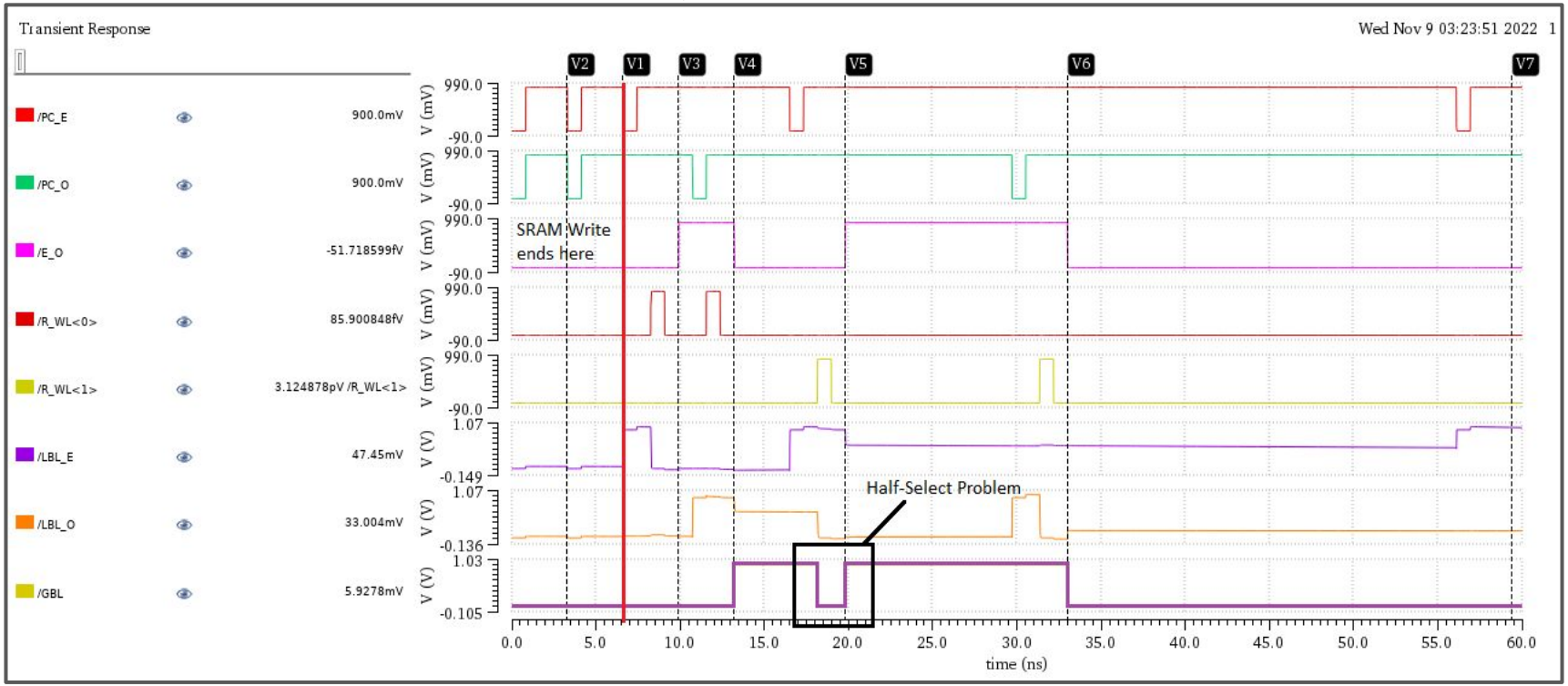
Proposed PWM generation Timing Diagram



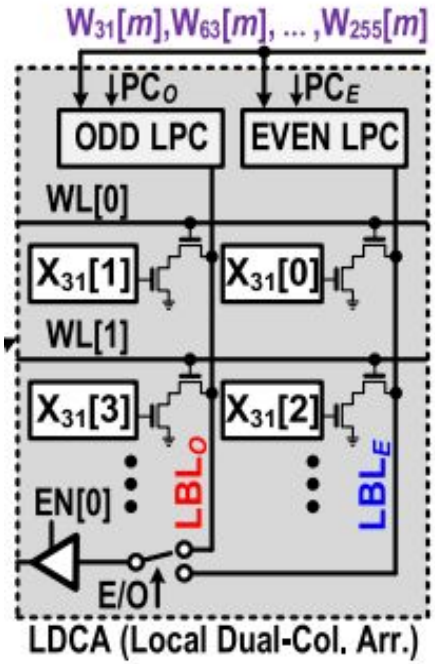
We have taken the preclk signal of the Local Precharger to be “active low” signal since we have used a PMOS to precharge the local bitline.

Half Select Problem in the PWM signal timing diagram

W = 1, X = 0110



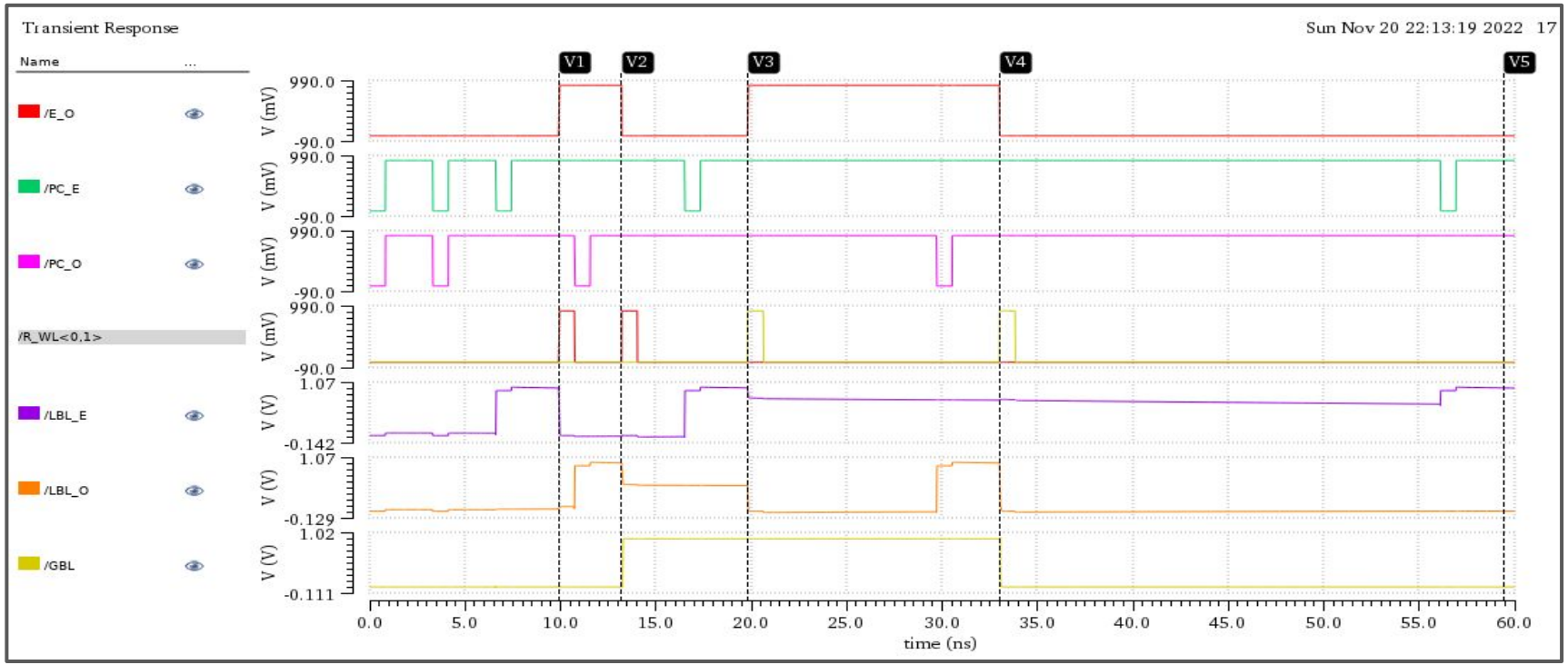
Half Selection Problem



- The turning on of WL to perform the AND operation of the even bit would also turn on the NMOS transistor for odd bit stored SRAM.
- This would flip the GBL value being driven by LBL_O in the overlapping period.

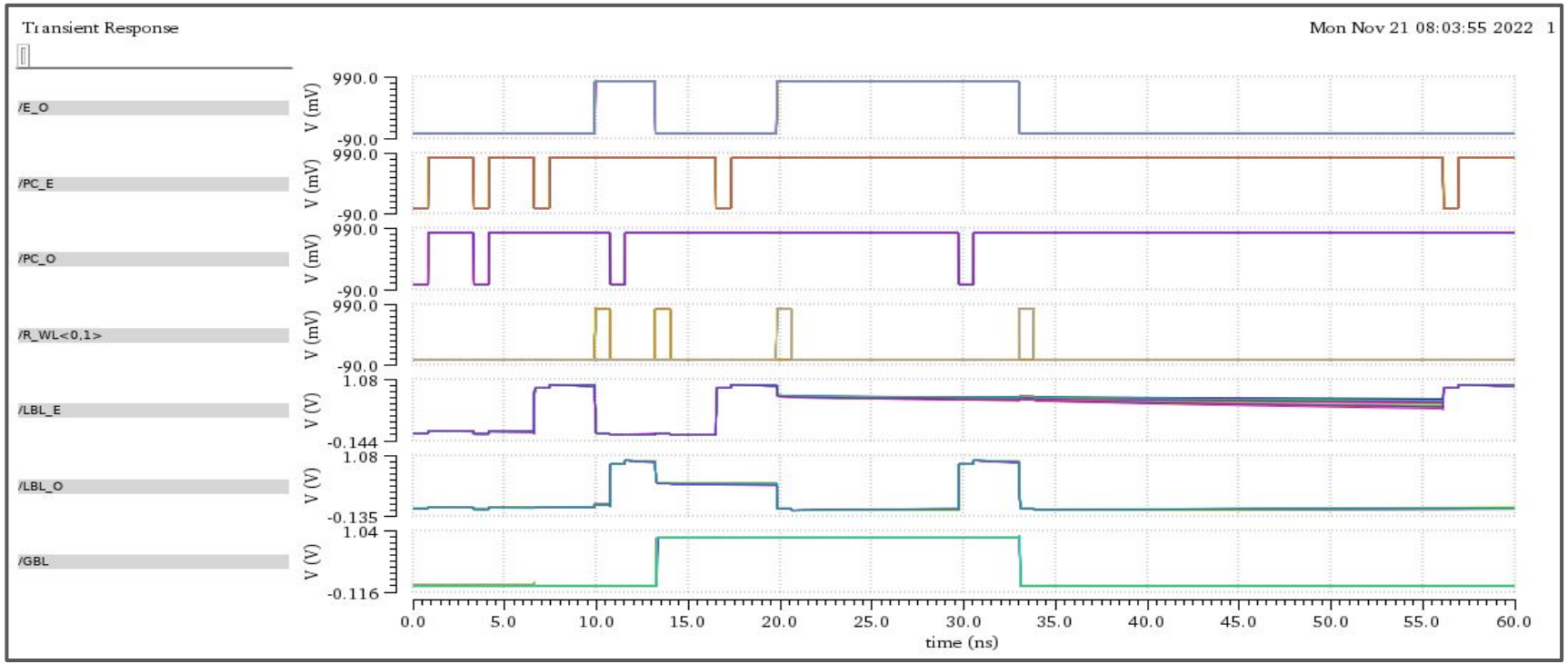
Eliminating the Half-selection problem and generating PWM signal

W = 1, X = 0110

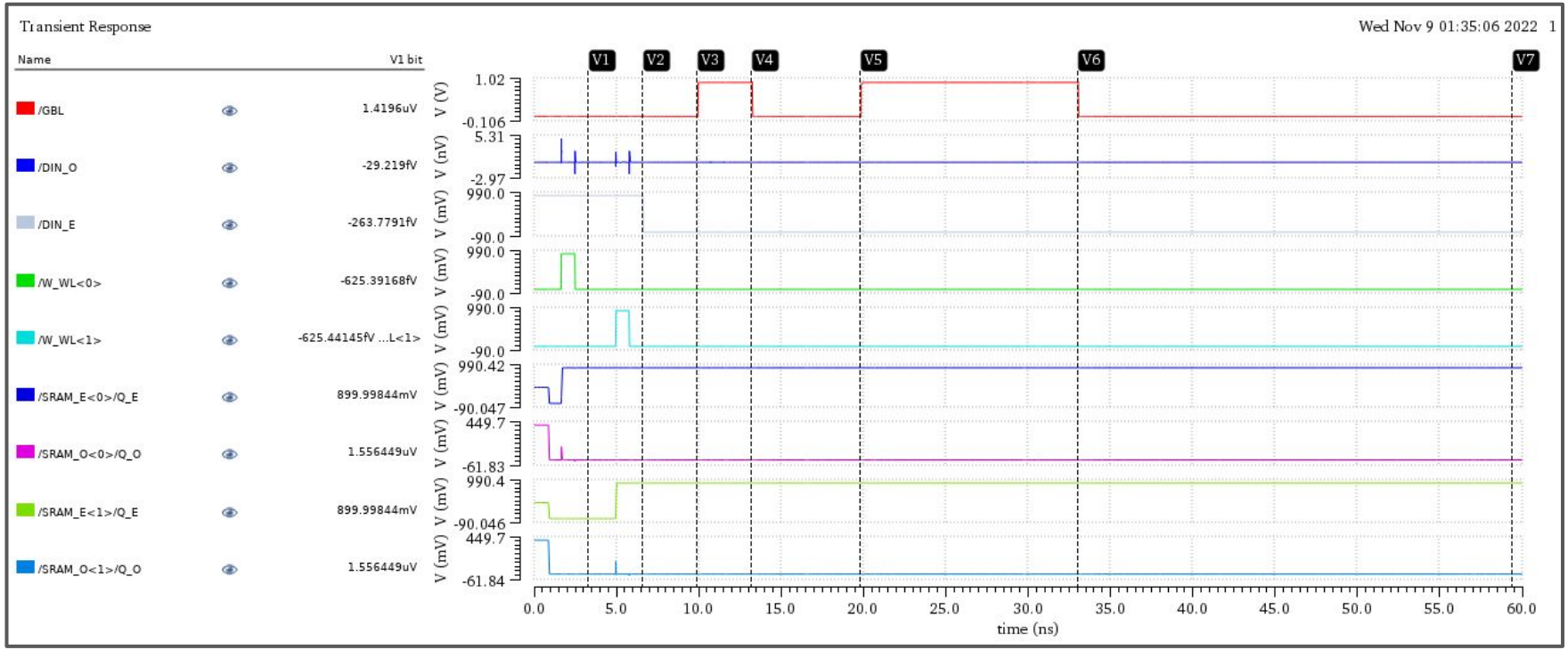


Monte-Carlo Output of Error Free PWM Signal

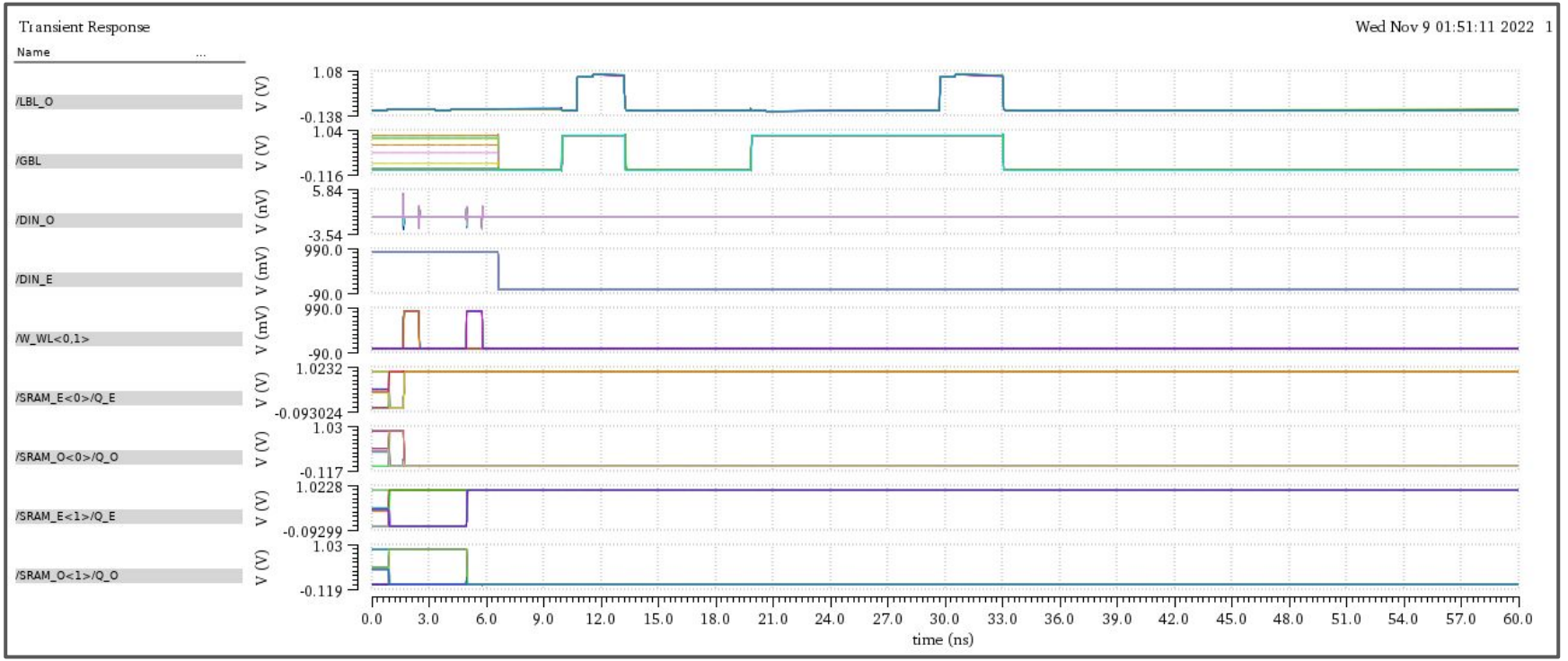
W = 1, X = 0110



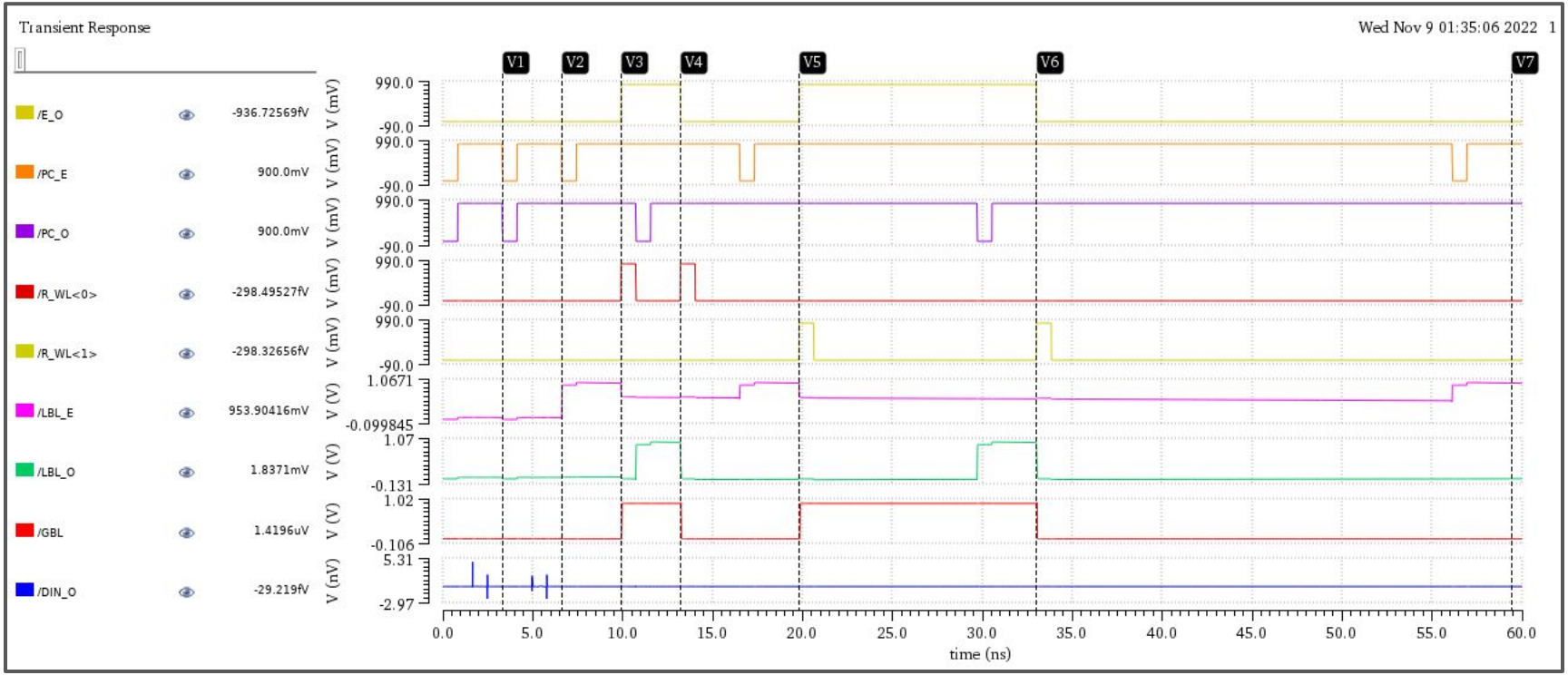
SRAM Write (X=0101, W =1)



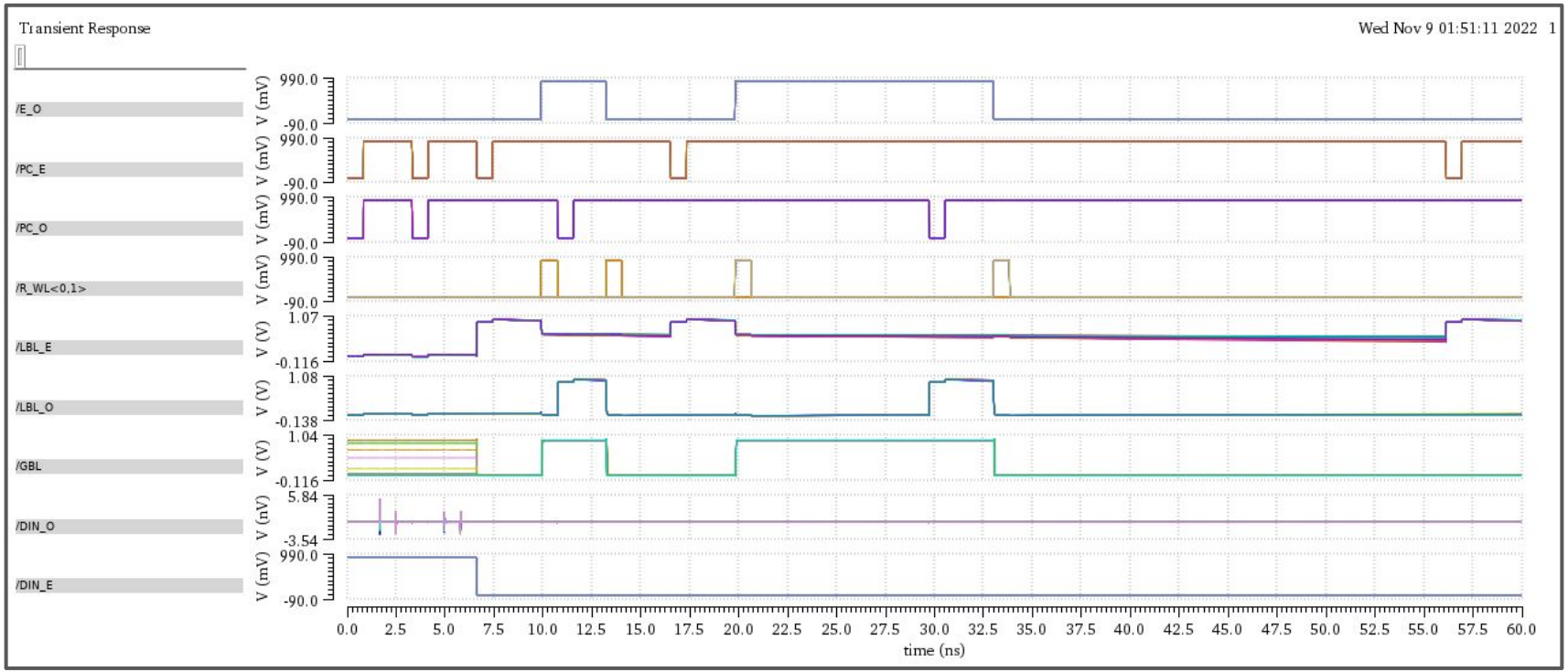
SRAM Write as generated by Monte Carlo simulation



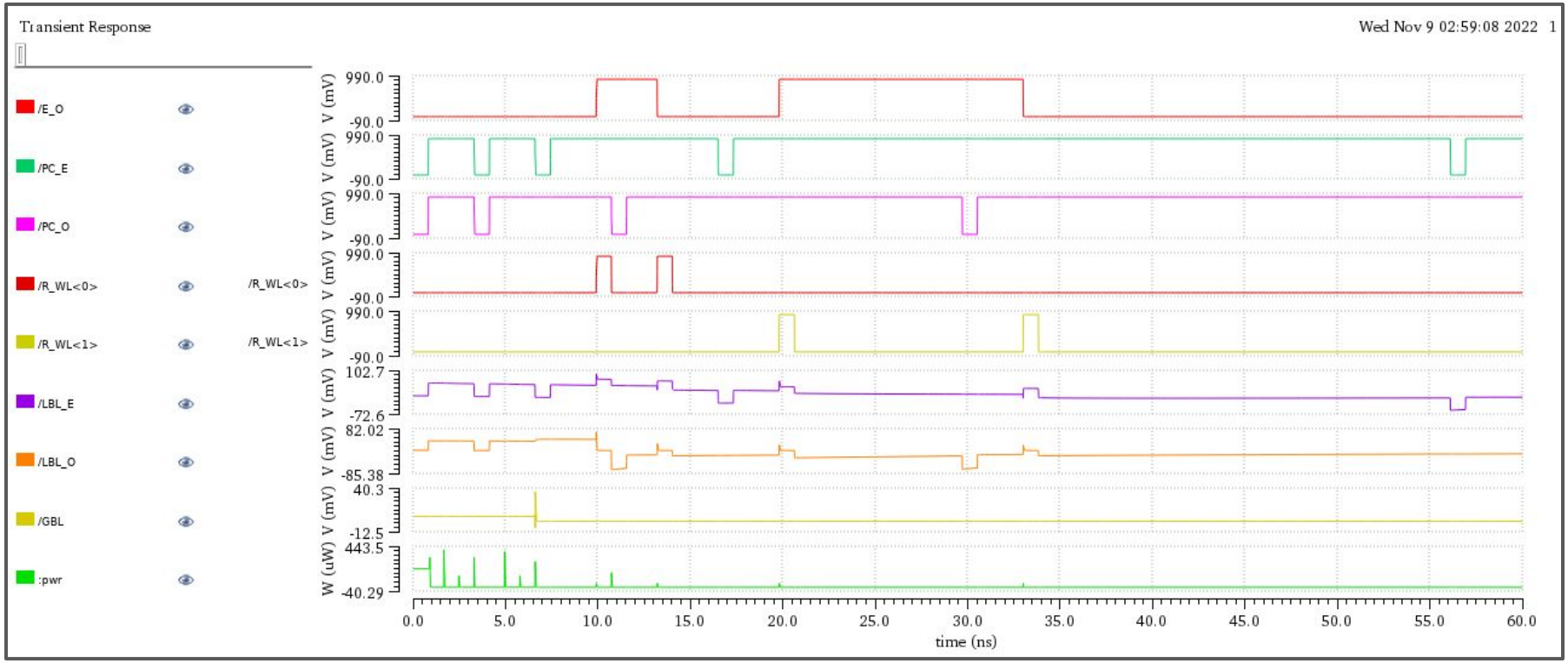
PWM signal generation (X=0101, W=1)



PWM signal generation by Monte Carlo simulation

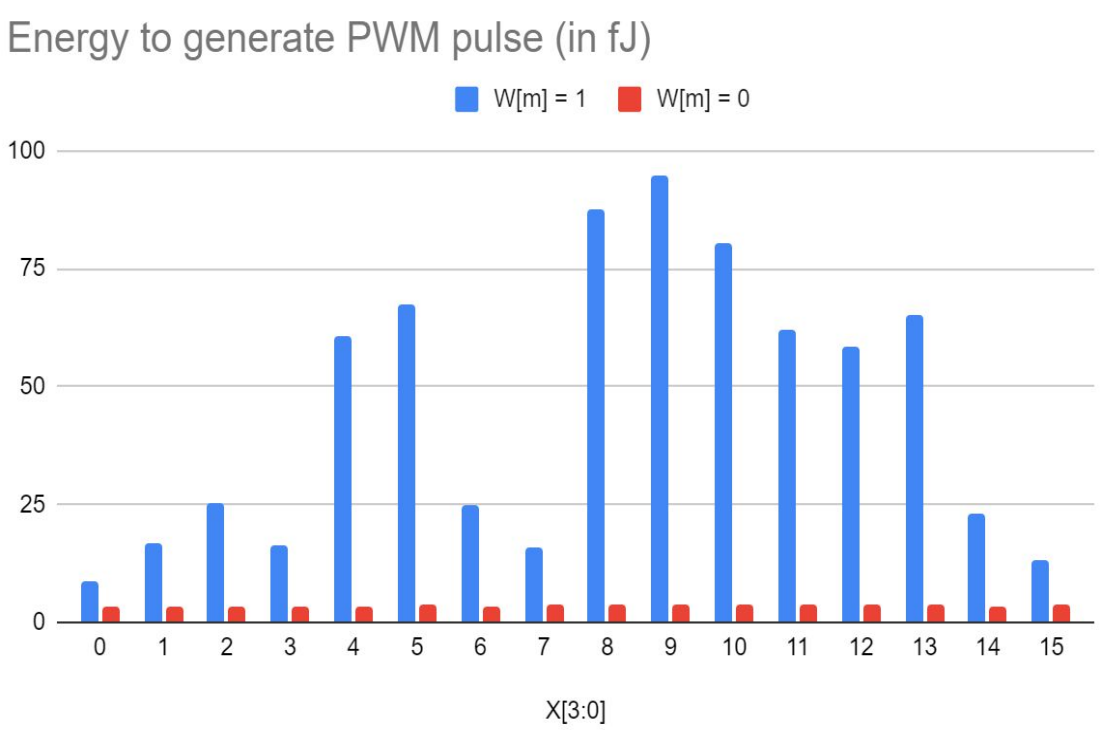


PWM signal generation (X=0110, W=0)



Total energy generated by one MAC operation

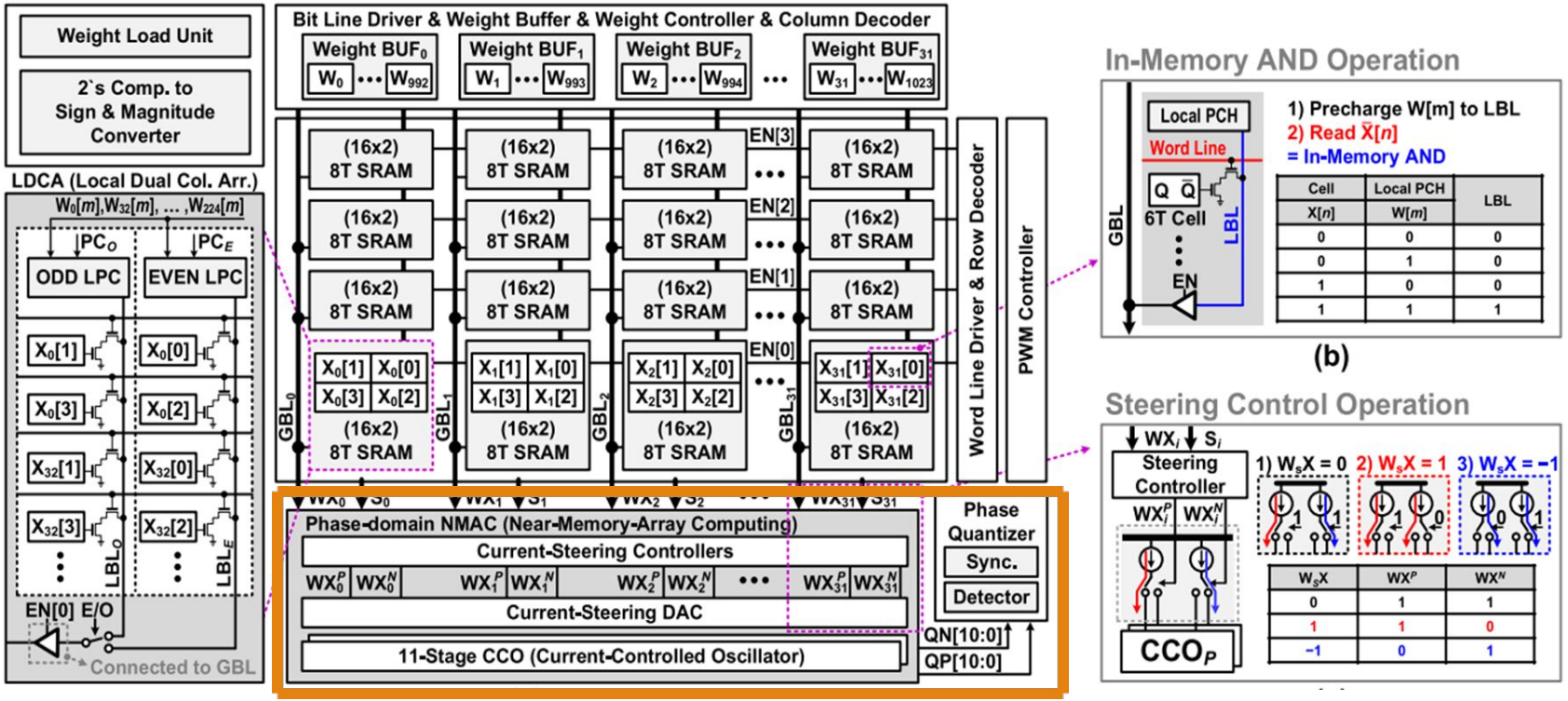
X[3:0]	Total Energy (in fJ)	
	W[m] = 1	W[m] = 0
0	8.838	3.535
1	16.68	3.541
2	25.11	3.538
3	16.41	3.547
4	60.85	3.555
5	67.42	3.563
6	24.71	3.56
7	15.67	3.573
8	87.36	3.571
9	94.89	3.583
10	80.36	3.579
11	62.15	3.569
12	58.33	3.578
13	64.98	3.58
14	22.95	3.55
15	13.37	3.842



Delay Computation

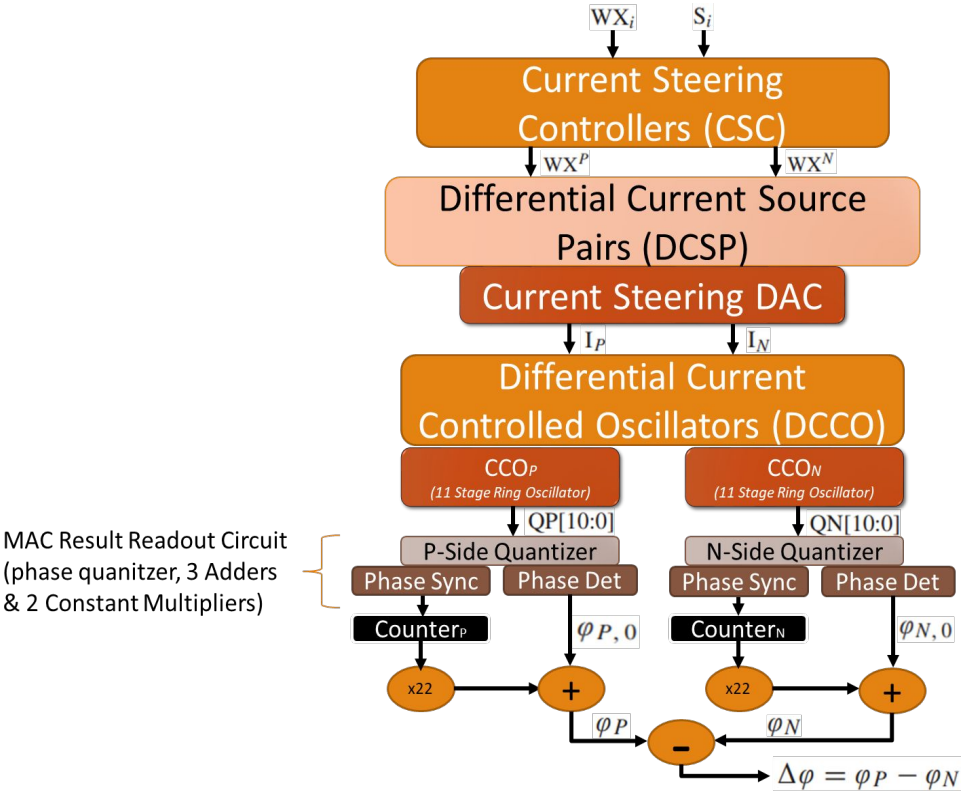
Data Stored in the SRAM	Worst Case Computation Delay
X[0]	10.98 ps
X[1]	15.8 ps
X[2]	19.1 ps
X[3]	20.3 ps

Phase-Domain Near-Memory-Array Computing (PNMAC)



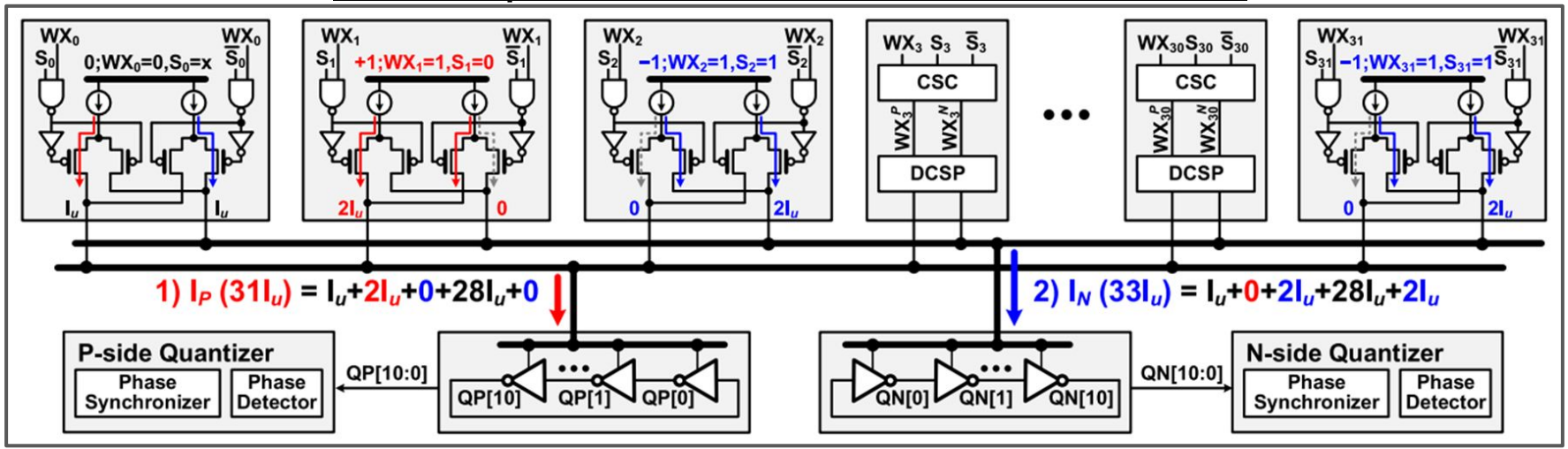
Phase-Domain Near-Memory-Array Computing (PNMAC)

Architecture:



Phase-Domain Near-Memory-Array Computing (PNMAC)

32 Parallel phase domain bit wise accumulation in PNMAC



The following circuit includes:

a) Current Steering Controller (CSC).

b) Differential Current Source Pairs (DCSP).

c) Current Steering DAC.

d) 11 stage Ring Oscillators (CCOP and CCON)

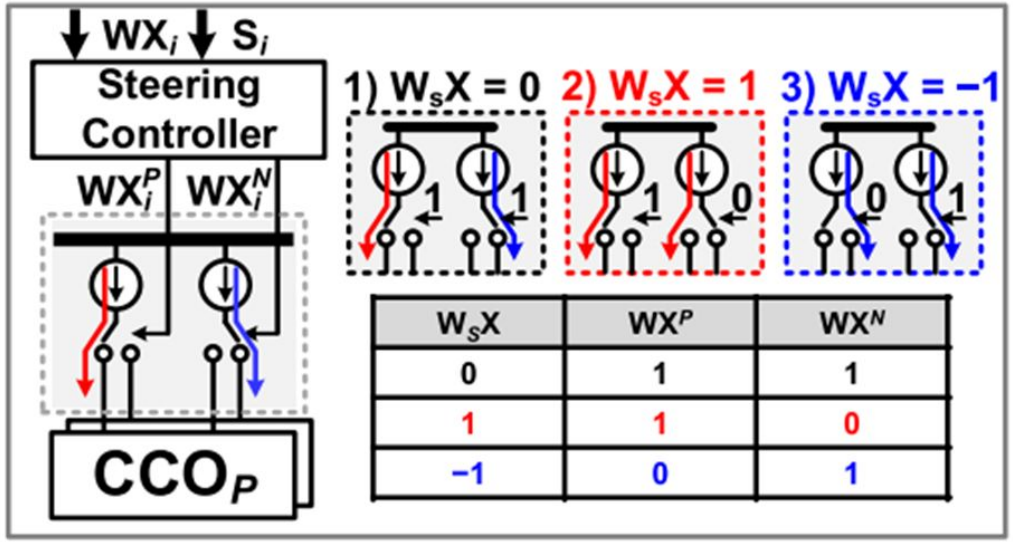
e) P and N side Quantizers.

$$I_P = 32I_u + I_u \sum_{i=0}^{31} (1 - 2S_i)WX_i$$

$$I_N = 32I_u + I_u \sum_{i=0}^{31} (2S_i - 1)WX_i$$

Current Steering Operation

- For $W_SX = 0$, $S = X$, $I_P = I_N = Iu$;
- For $W_SX = 1$, $S = 0$, $I_P = 2Iu$, $I_N = 0$;
- For $W_SX = -1$, $S = 1$, $I_P = 0$, $I_N = 2Iu$;



W_SX	WX^P	WX^N
0	1	1
1	1	0
−1	0	1

Current Equations

The DCSPs Generate P-side current, I_P , and N-side current, I_N .

I_P and I_N are expressed as,

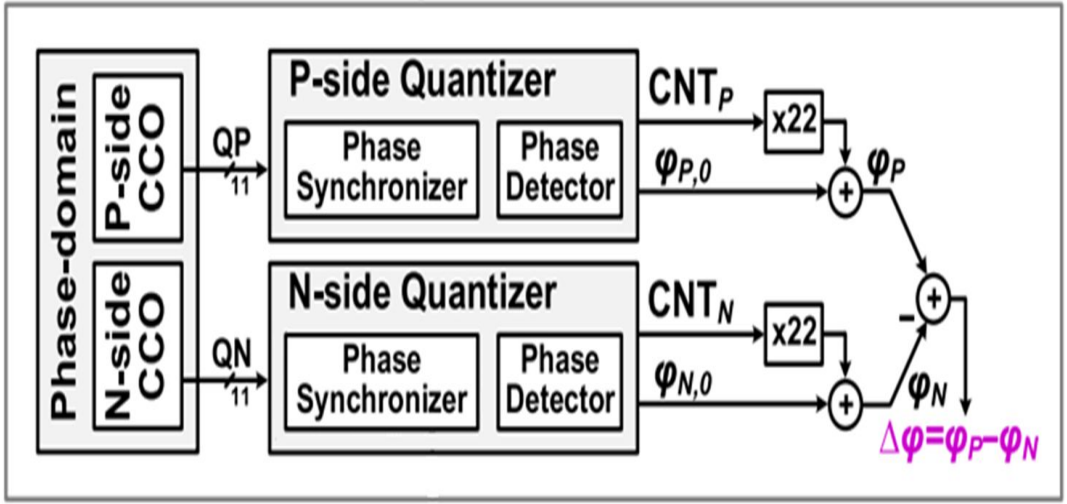
$$I_P = 32I_u + I_u \sum_{i=0}^{31} (1 - 2S_i) WX_i$$

$$I_N = 32I_u + I_u \sum_{i=0}^{31} (2S_i - 1) WX_i$$

Readout Circuit to Sample Final MAC results

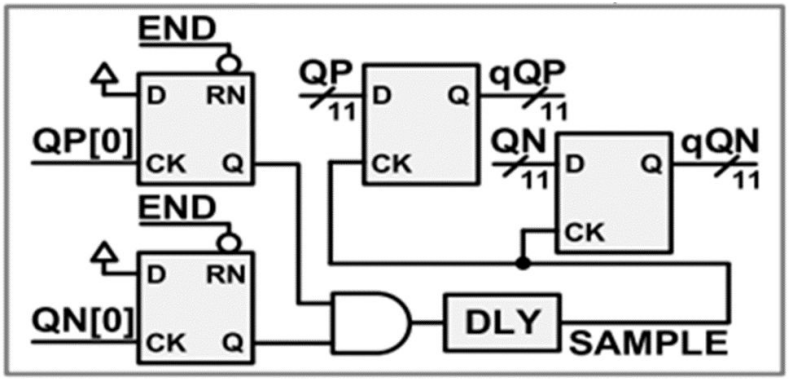
MAC-result-readout circuit consists of:

- 2 Phase Quantizers
  - Phase Synchronizer
  - Phase Detector
- 3 Adders
- 2 Constant Multipliers



Phase Synchronizers

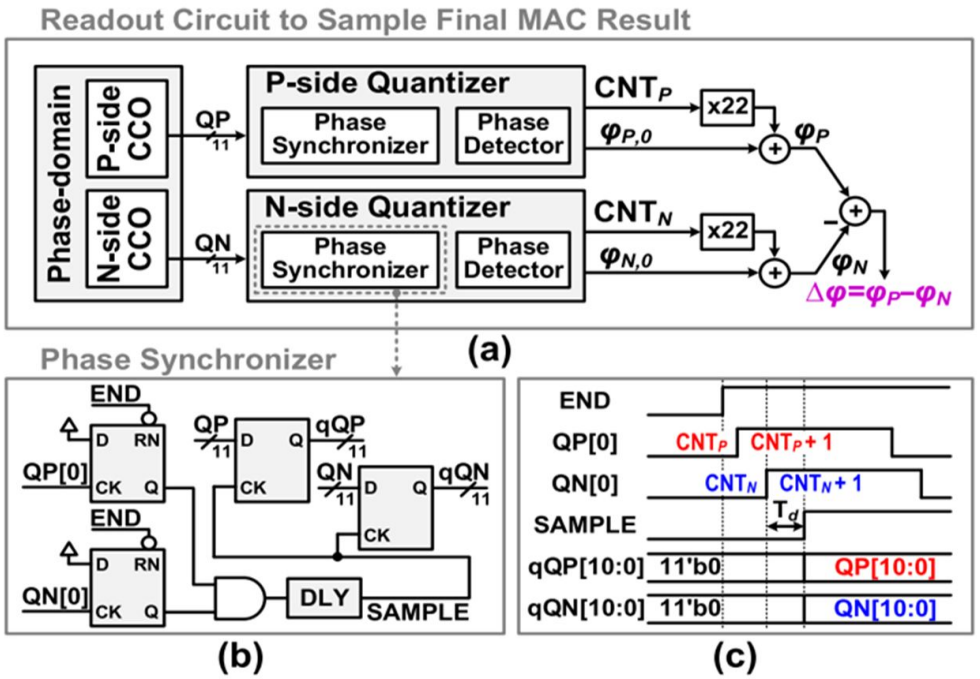
Function: To prevent metastable states



- QP and QN are asynchronous with main clock signal.
- It Comprise of
  - 2 D-Flip Flops for rising edge detection.
  - An AND gate
  - A delay cell (Delay=Td).
  - 2 sets of D Flip Flops latching the DCCO phases.
- Enabled by END signal triggered at the end of MAC operation

Working of Readout Circuit with Timing Diagram

- 2 D Flip-Flops used for rising edge detection, An AND gate, A Delay cell and 2 Sets of D Flip-Flops latching the DCCO phases.
- The rising edge detection synchronizers are enabled by the END signal triggered at the end of MAC operation to detect rising-edge of QP[0] and QN[0].
- The SAMPLE signal, which is synchronized to QP[0] and QN[0],is switched from low to high after delay time, Td, passes.
- Then, all the data are sampled at the rising edge of SAMPLE.



Challenges

The Phases of the CCOP and CCON are expressed as follows:

$$\varphi_P = 22 \times \mathrm{CNT}_P + \varphi_{P,0}$$

$$\varphi_N = 22 \times \mathrm{CNT}_N + \varphi_{N,0}$$

The phase difference between them is,

$$\Delta\varphi = \varphi_P - \varphi_N$$

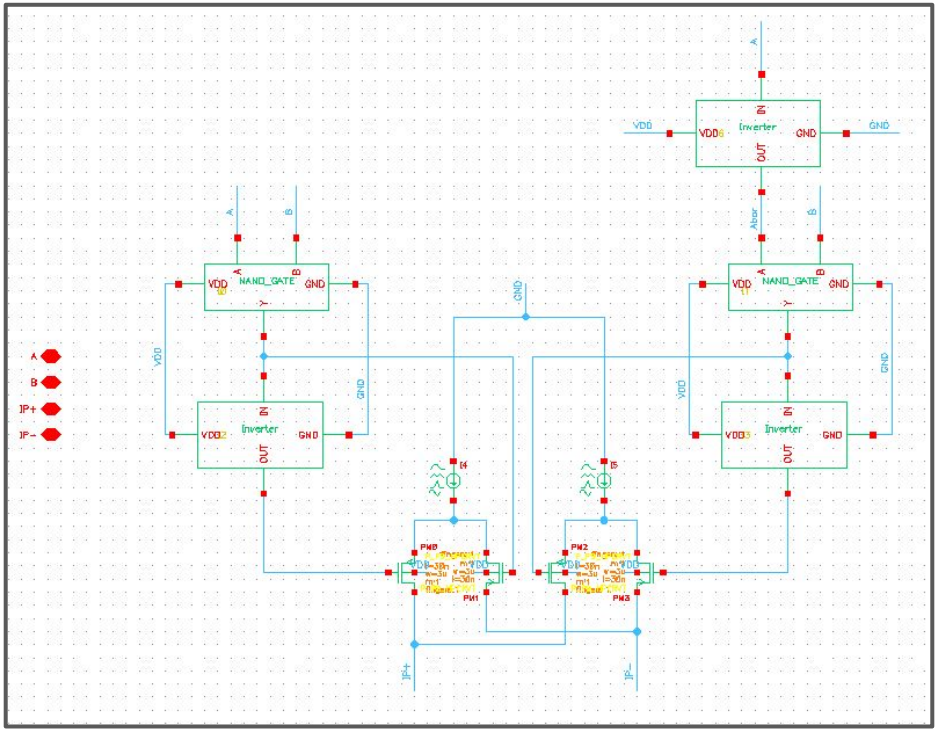
The final computational result is given by,

$$\sum_{i=0}^{num-1} (1 - 2\mathrm{S}_i)\mathrm{WX}_i = \alpha_{CCO} \times \Delta\varphi$$

PNMAC Schematics and Waveforms

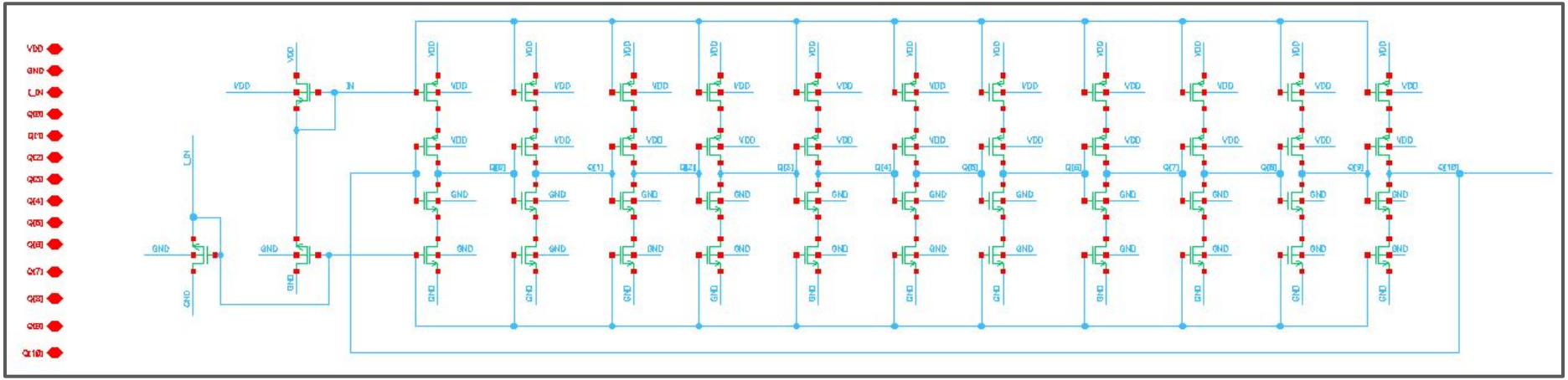
Current Steering Controller

Current Steering Controller: Steers the current to either P side or N side based on the sign bit value.



Current Controlled Oscillator

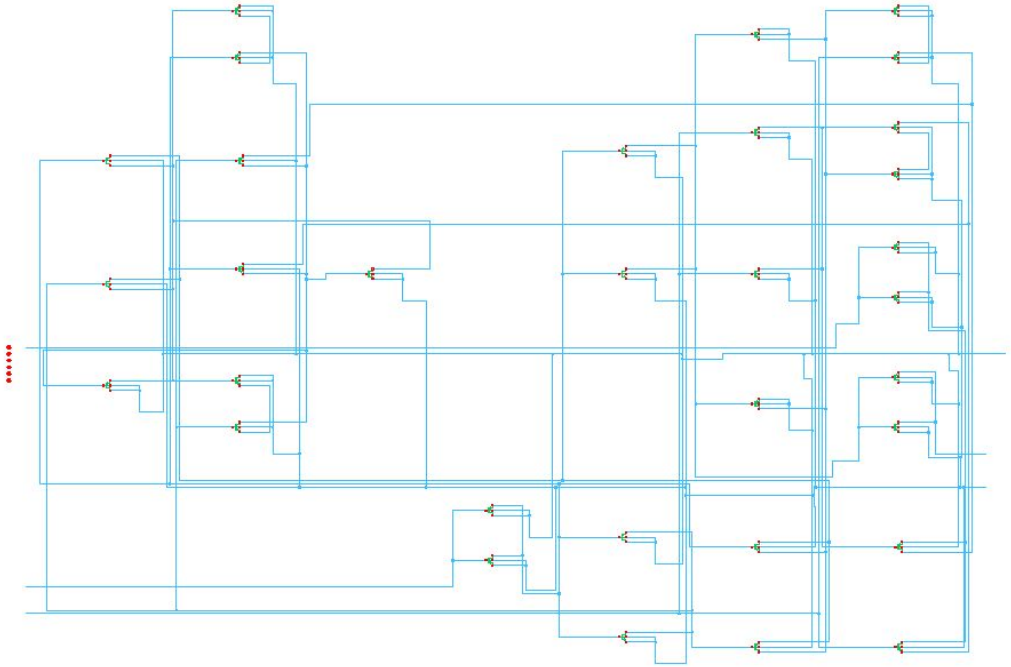
Current Controlled Oscillators: To convert the current values into frequency and phase difference.



D-Flip Flop

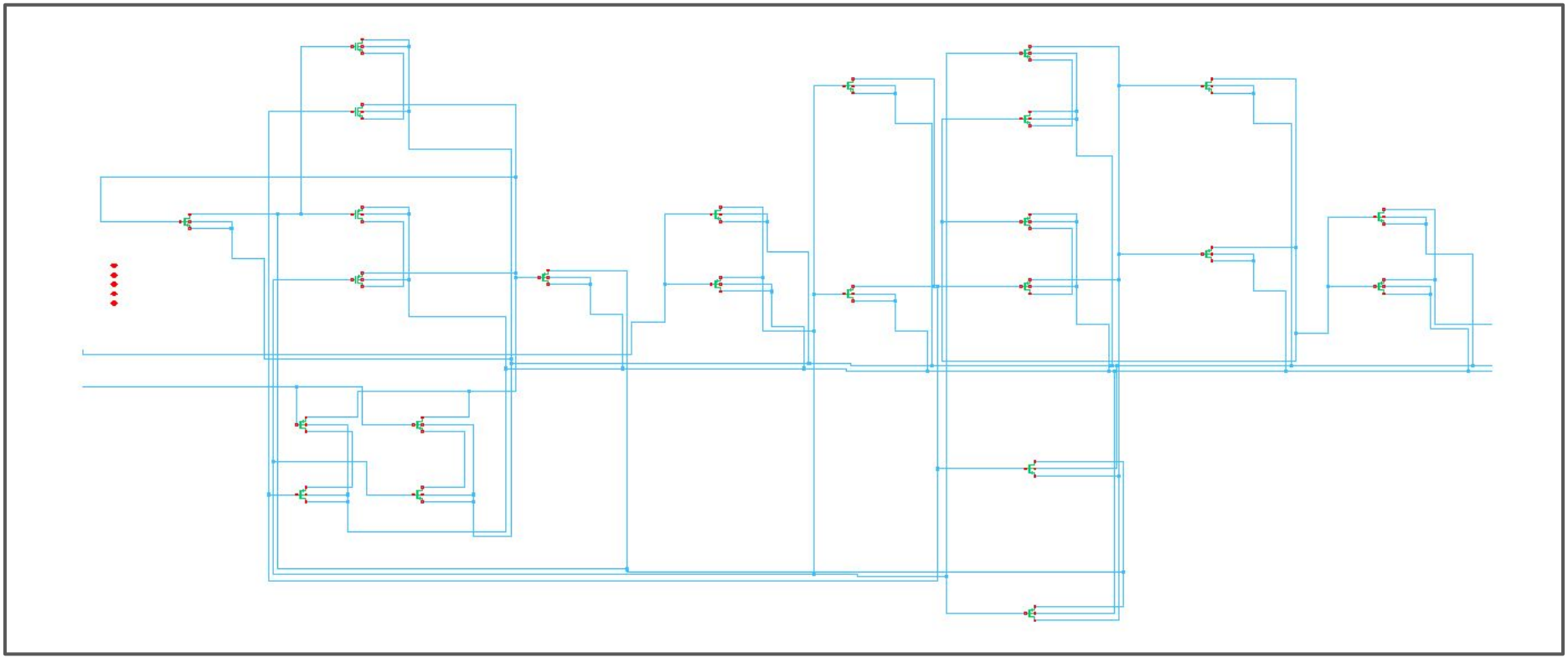
D-Flip Flop: 2 Types of D flip flops are used, with and without enable bit.

Type-1:



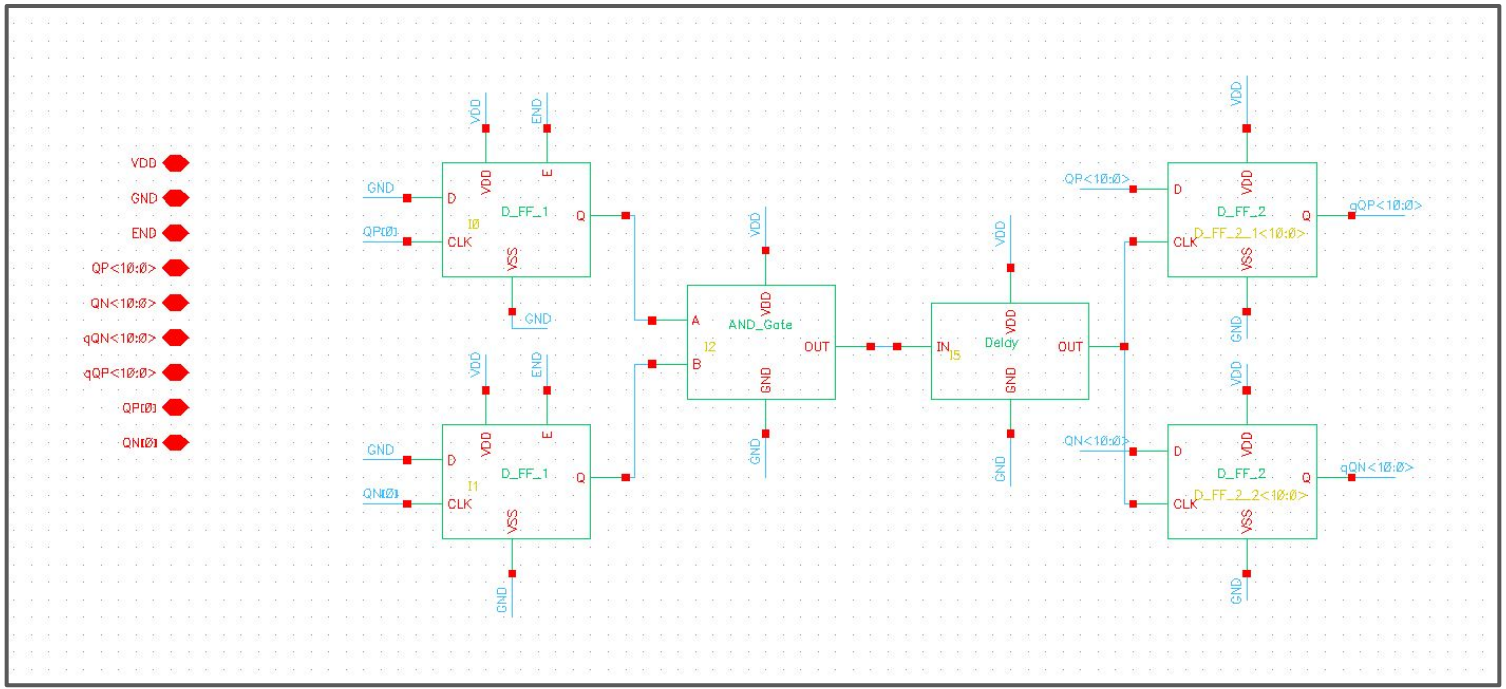
D-Flip Flop

Type-2:



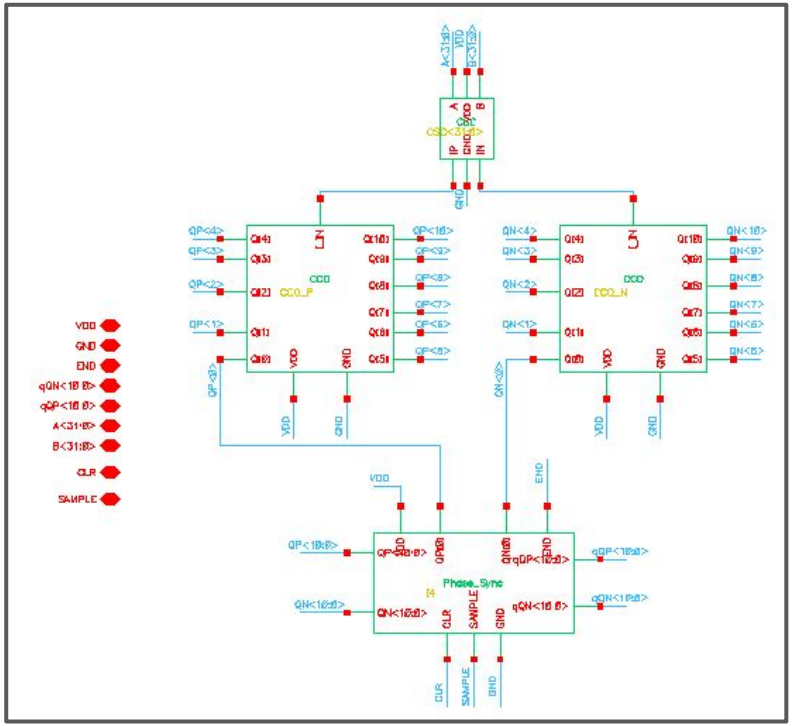
Phase synchronizer and Detector

Phase Synchronizer and detector: To synchronize the QP[0] and QN[0] signals for preventing metastable states and for detecting the phase value of the signal

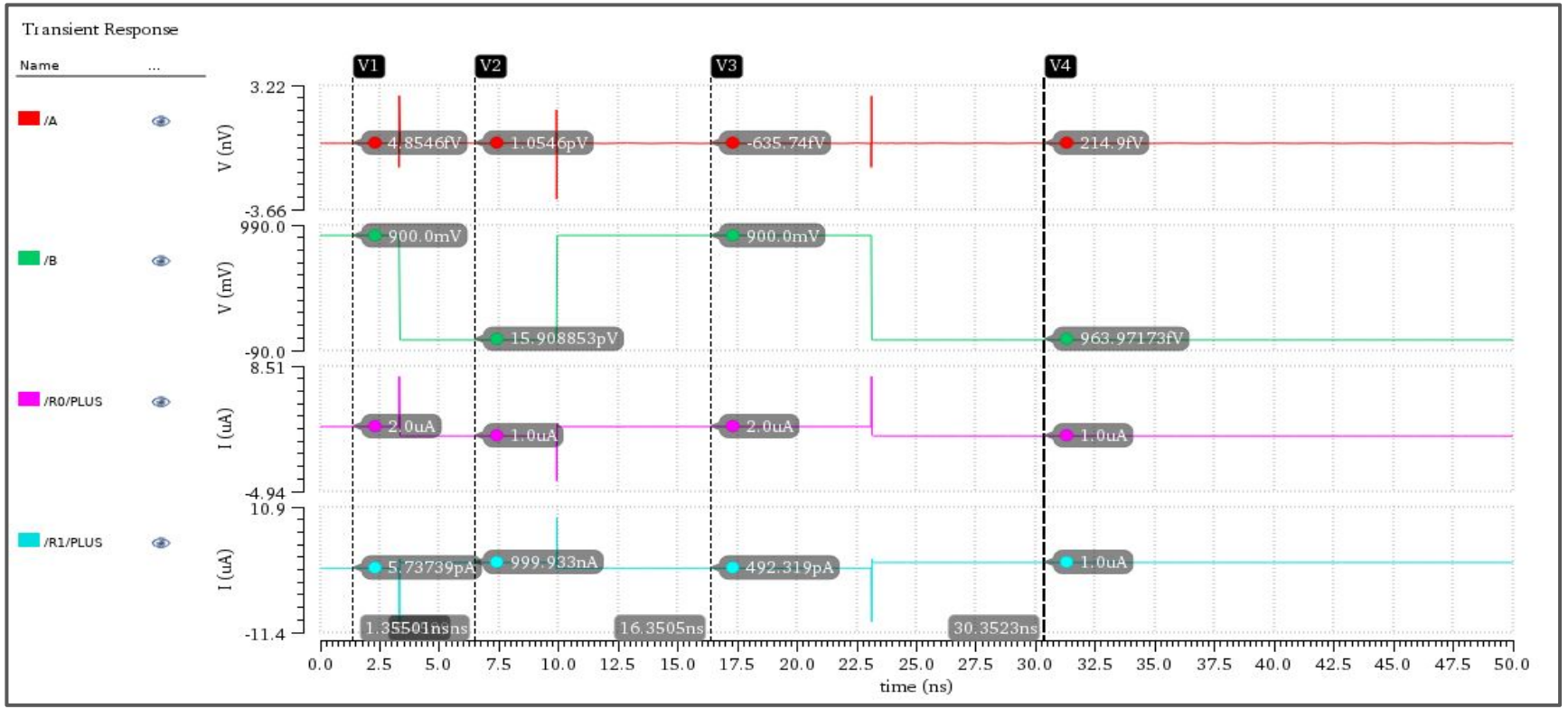


PNMAC Circuit

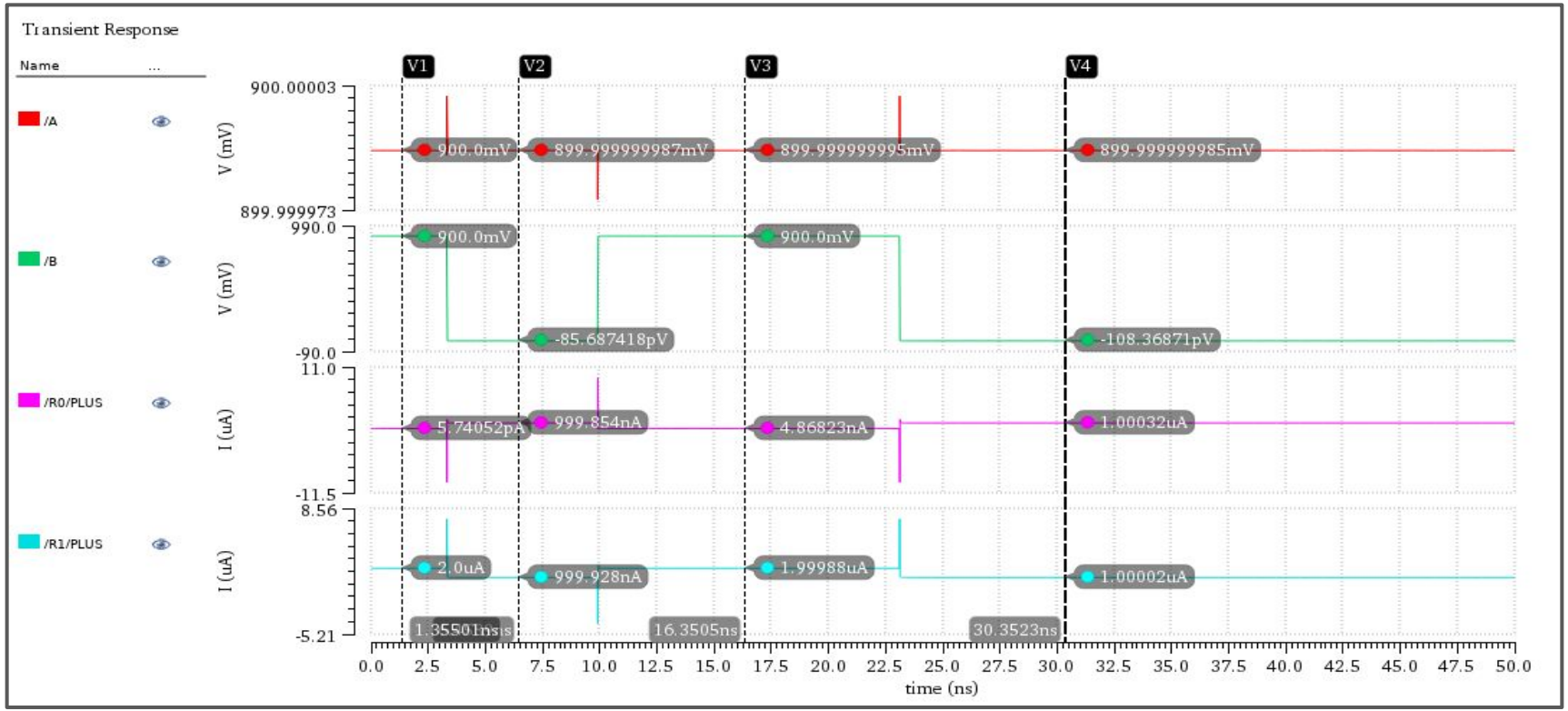
This circuit consists of CSCs, CCOs and Phase synchronizer circuit



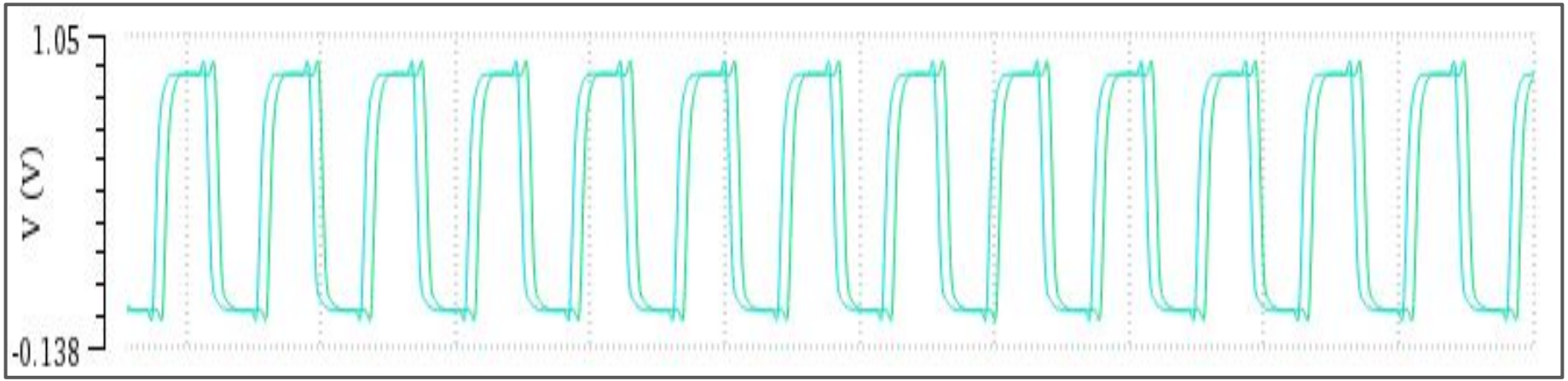
CSC Output for X=0101, S=0



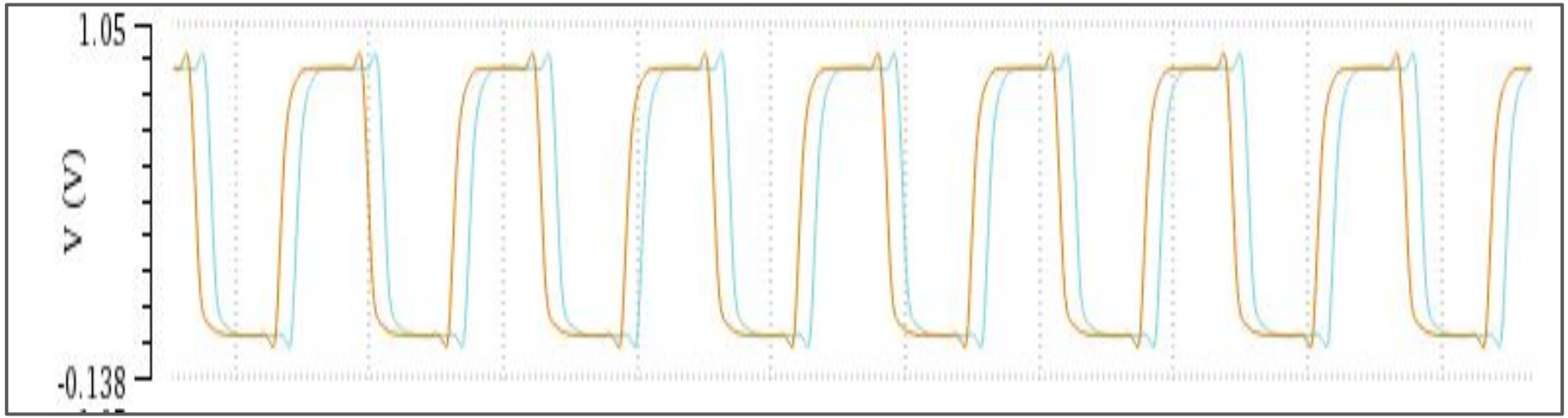
CSC Output for X=0101, S=1



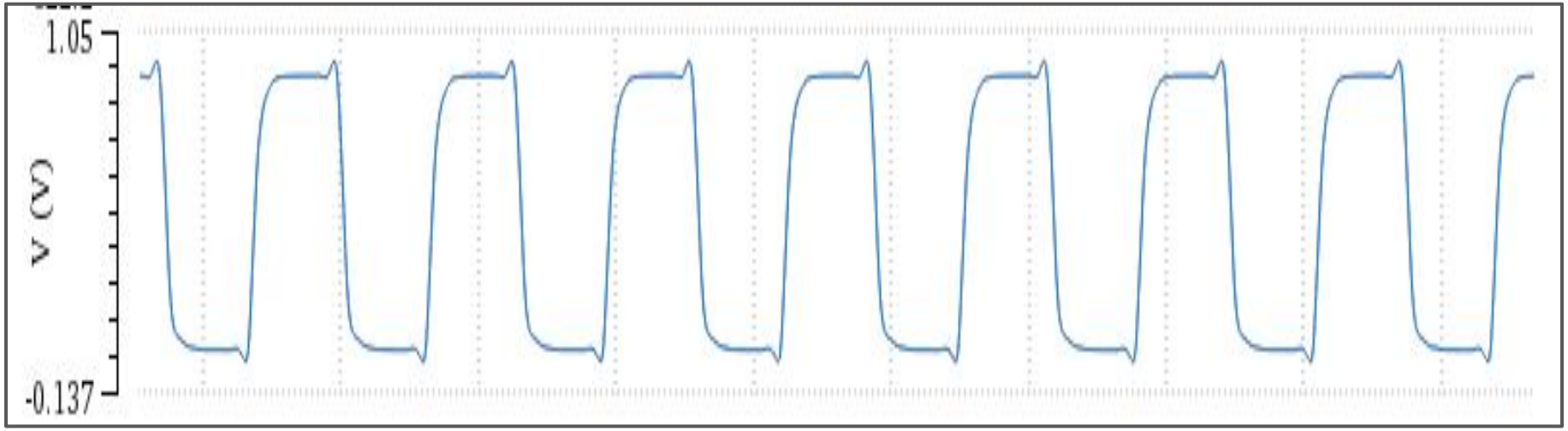
CCO Output for X=0101, S=0



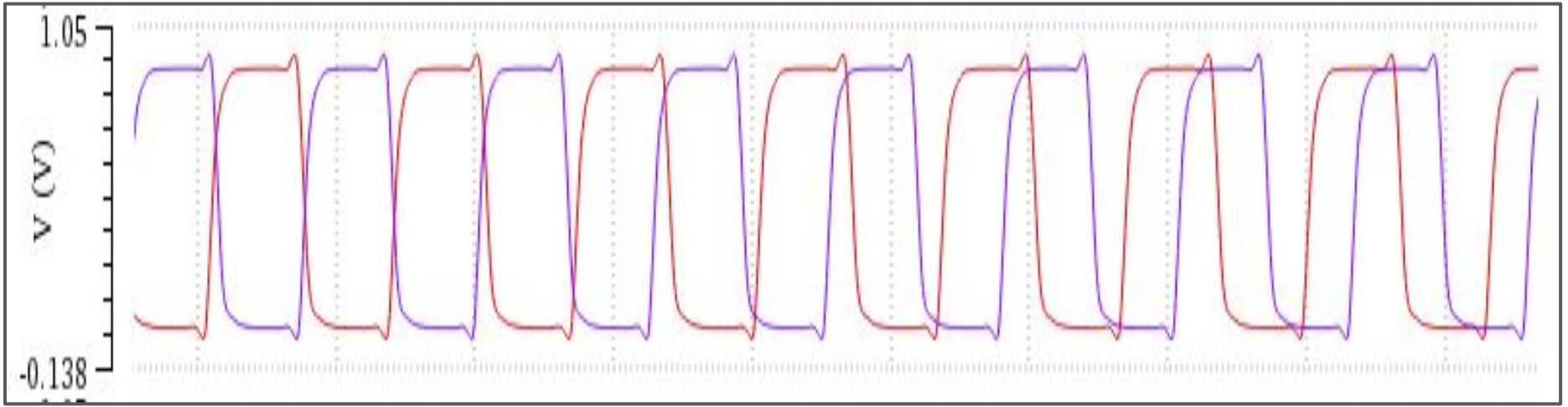
CCO Output for X=0101, S=1



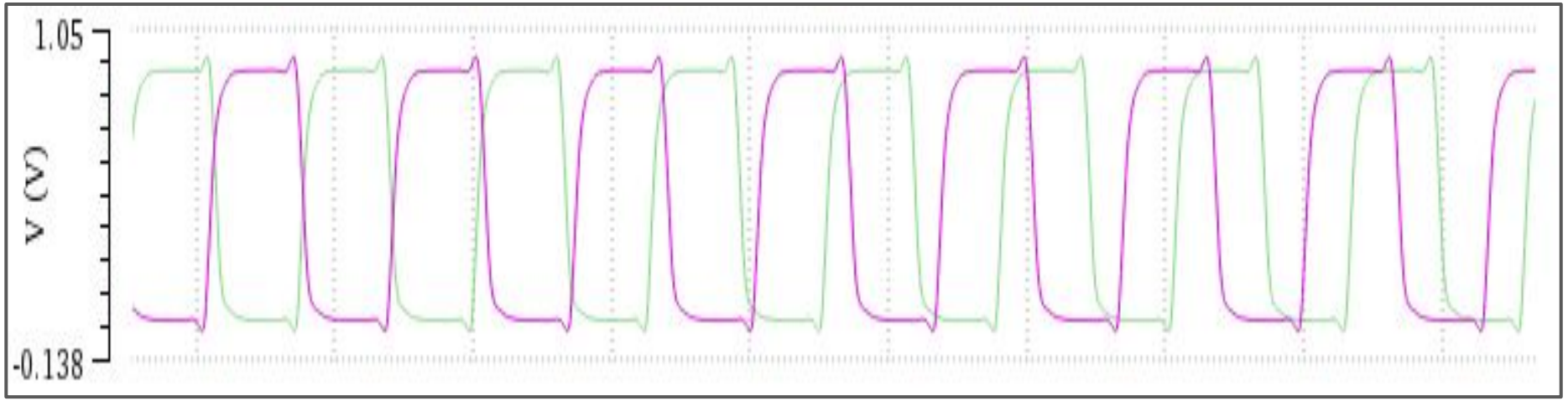
CCO Output for X=0000



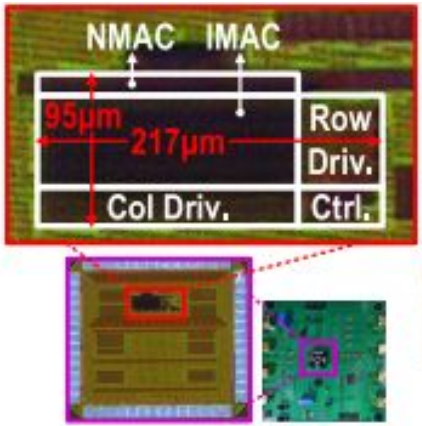
CCO Output for X=1010, S=0

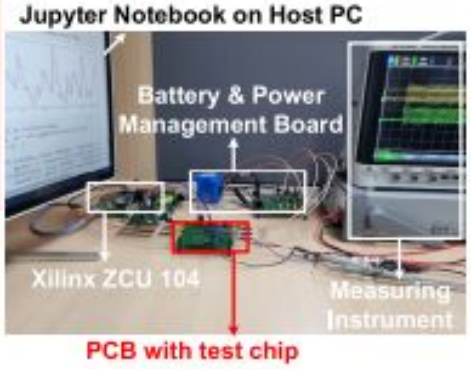


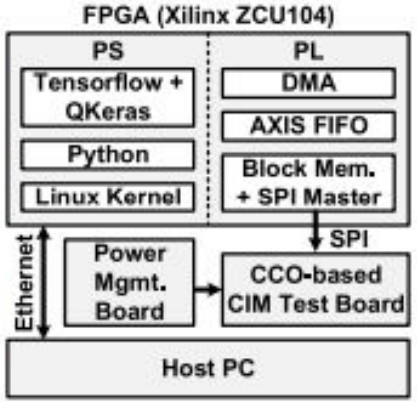
CCO Output for X=1010, S=1



Experimental Setup



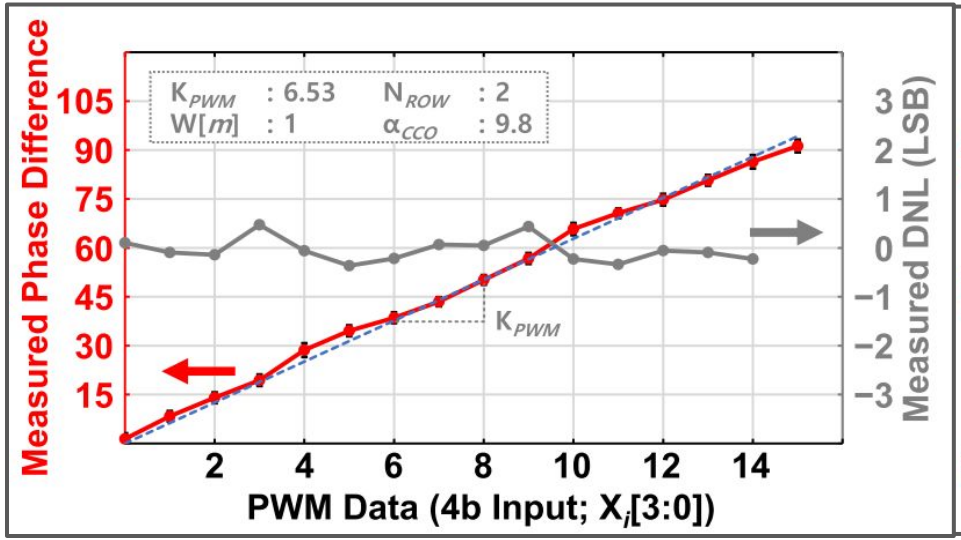


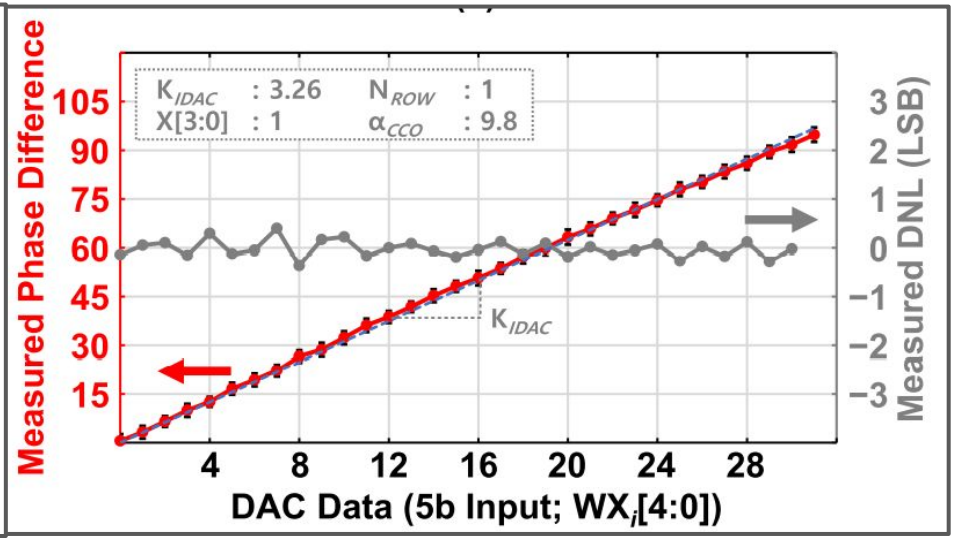


Chip Summary

Technology	65nm
Computation Domain	Phase domain
Cell & CIM Structure	8T IMAC + NMAC
Array Size	4kb
IMAC Macro / NMAC Area (mm^2)	0.0206 / 0.0001
Supply Voltage (V)	0.75-0.8
Clock Frequency	300MHz
Energy per Classification (nJ)	831.9
CIM macro Power (mW)	0.019
Throughput (GOPS)	0.42
Energy Efficiency (TOPS/W)	22.42

Results and Plots Presented in the Paper

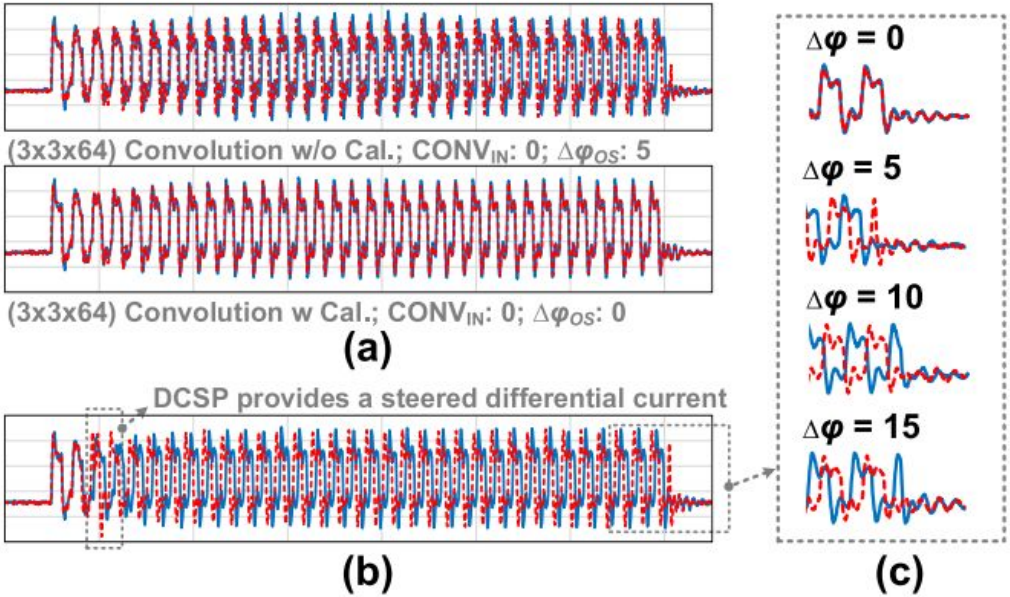




Measured Transfer functions and Differential Non Linearity of PWM and IDAC

Results and Plots Presented in the Paper

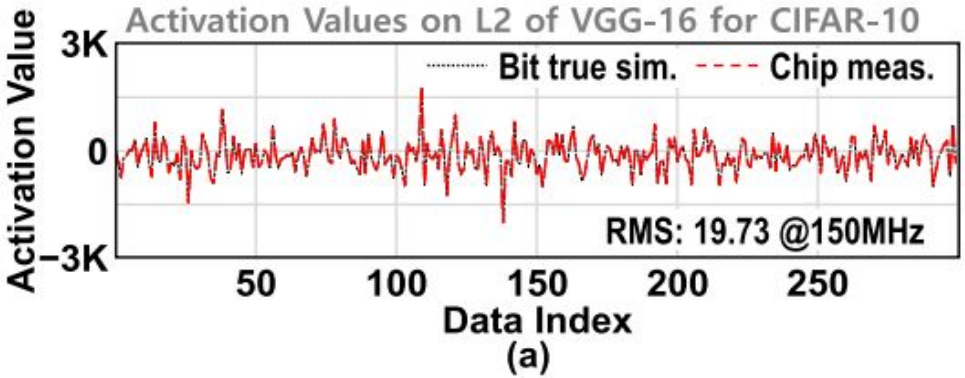
$$\sum_{i=0}^{num-1} (1 - 2S_i)WX_i = \alpha_{CCO} \times \Delta\varphi$$

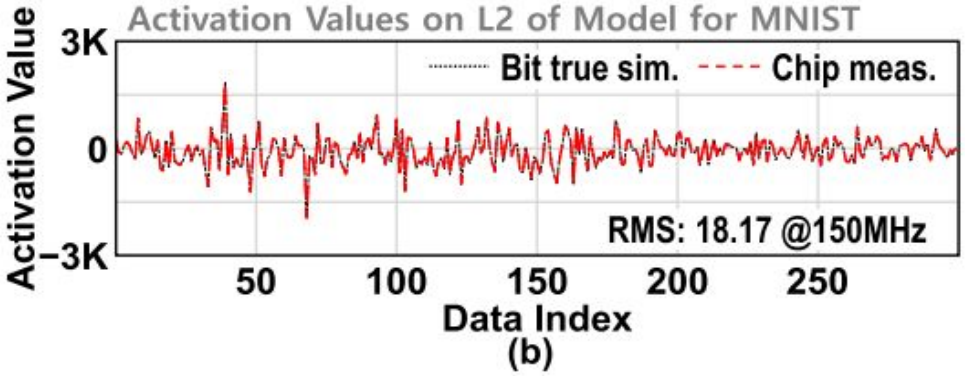


Measured waveforms of the DCCO outputs (a) with and without calibration, (b) with a differential current, and (c) at the end of operation according to various input data.

Results and Plots Presented in the Paper

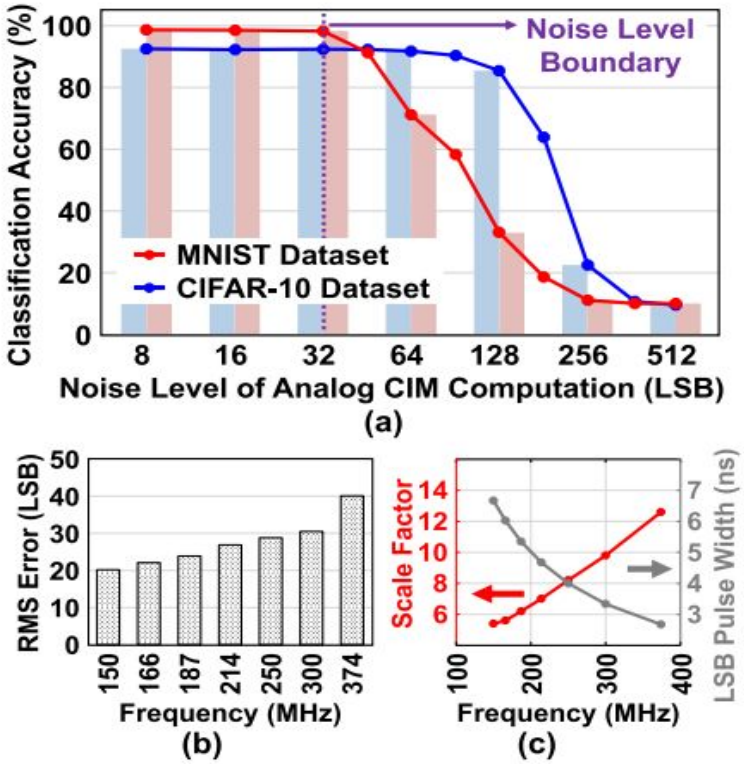
Measured and simulated activation values of the CNN: (a) values on the VGG-16 network for CIFAR-10 and (b) values on the simple-CNN network for MNIST.





Results and Plots Presented in the Paper

(a) Simulated classification accuracy vs. the noise level of the analog CIM computation for MNIST and CFAR-10 datasets, (b) measured RMS error vs. the operation frequency, and (c) measured scale factor and LSB pulse width vs. the operation frequency.



Comparison with other Works

	ISSCC '20 [13]	JSSC '19 [16]	ISSCC '19 [17]	ISSCC '20 [10]	ISSCC '21 [15]	This work
Technology	28nm	28nm	40nm	7nm	22nm	65nm
Domain	Analog	Phase	Time	Analog	Digital	Phase
Cell & CIM Structure	6T IMAC + NMAC	Only MAC engine	Only MAC engine	8T IMAC	6T Cell + Digital	8T IMAC + NMAC
Array Size	64kb	-	-	4kb	64kb	4kb
Macro Area (mm^2)	0.3230	[2]0.0012	[2]0.124	0.0032	0.202	0.0206
Bit Precision (Input / Weight / Output)	4~8b / 4~8b / 12~20b	8b / 8b / 10b	8b / 1b / 8b	4b / 4b / 4b	1~8b / 4~16b / 16~25b	4b / 1.5~4b / 10b
Supply Voltage (V)	0.7~0.9	0.7	0.375~1.1	0.8	0.72	0.75~0.8
Dataset	CIFAR-10	MNIST	MNIST	MNIST	-	MNIST / CIFAR-10
Measured Accuracy (Software Baseline) (%)	91.5 (91.7)	98.1 (98.2)	[7]98.42 (98.92)	98.5 (99.63)	-	98.1 / [7]92.3 (98.36 / 92.48)
Total CIM / NMAC Power (mW)	[1]1.825 / -	- / [2]0.152	- / [2]0.030	[1]1.061 / -	[1]37.078 / -	[3]0.019 / [2]0.0033
MAC rate (MHz)	[4]119~244	780	0.19~3.12	[4]181	[4]55~100	20
Throughput (GOPS)	124.88 (4b/4b)	[5,6]8.512	[6]0.182	372.4	3300 (4b/4b)	0.42 (4b/4b)
Energy Efficiency (TOPS/W)	68.44 (4b/4b)	[2,5,6]56.0	[2,6]6.05	351	89 (4b/4b)	[3]22.4 / [2]128.6 (4b/4b)

[1] Calculated by dividing throughput (GOPS) by energy efficiency (TOPS/W)
[2] Metrics for MAC engine or NMAC
[3] Total CIM (DIMAC+PNMAC) power and energy with a 59.8% zero-skipping rate
[4] Calculated by dividing 1 by cycle time (ns)
[5] Calculated when the input activation rate is 3%
[6] Normalized to 4b input and 4b weight
[7] Simulation result with circuit non-ideal effects

References

- X. Si et al., “A twin-8T SRAM computation-in-memory unit-macro for multibit CNN-based AI edge processors,” IEEE J. Solid-State Circuits, vol. 55, no. 1, pp. 189–202, Jan. 2020.
- H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, “A 64-tile 2.4- Mb in-memory-computing CNN accelerator employing charge-domain compute,” IEEE J. Solid-State Circuits, vol. 54, no. 6, pp. 1789–1799, Jun. 2019.
- S. K. Gonugondla, M. Kang, and N. R. Shanbhag, “A Variation-tolerant in-memory machine learning classifier via on-chip training,” IEEE J. Solid-State Circuits, vol. 53, no. 11, pp. 3163–3173, Nov. 2018.
- Z. Jiang, S. Yin, J.-S. Seo, and M. Seok, “C3SRAM: An in-memory-computing SRAM macro based on robust capacitive coupling computing mechanism,” IEEE J. Solid-State Circuits, vol. 55, no. 7, pp. 1888–1897, Jul. 2020.
- J.-H. Kim, J. Lee, J. Lee, J. Heo, and J.-Y. Kim, “Z-PIM: A sparsityaware processing-in-memory architecture with fully variable weight bitprecision for energy-efficient deep neural networks,” IEEE J. Solid-State Circuits, vol. 56, no. 4, pp. 1093–1104, Apr. 2021.
- Y. Toyama, K. Yoshioka, K. Ban, S. Maya, A. Sai, and K. Onizuka, “An 8 bit 12.4 TOPS/W phase-domain MAC circuit for energyconstrained deep learning accelerators,” IEEE J. Solid-State Circuits, vol. 54, no. 10, pp. 2730–2742, Oct. 2019.
- A. Sayal, S. S. T. Nibhanupudi, S. Fathima, and J. P. Kulkarni, “A 12.08-TOPS/W all-digital time-domain CNN engine using bidirectional memory delay lines for energy efficient edge computing,” IEEE J. Solid-State Circuits, vol. 55, no. 1, pp. 60–75, Jan. 2020.