MS PROJECT REPORT

DESIGN OF ANALOG MIXED SIGNAL NEURON FOR NEUROMORPHIC COMPUTATION

PREMSAGAR KITTUR (premsagar@g.ucla.edu) (804741631)
Subramanian S. Iyer (s.s.iyer@g.ucla.edu)
Sudhakar Pamarti (spamarti@ee.ucla.edu)

Table of content:

I. Introduction:

The human brain consists of millions of neurons that forms very complex and intricate neural connections. Through these intricate connections, human brain responds to an external stimulus. Whenever an external stimulus is applied and has reached a certain stimulus threshold, an electrical signal is transmitted from one neuron to the other, establishing neural connection.

The basic operation in a neuromorphic computation is multiplication. In a human brain, a neuron is connected to its neighboring neurons with certain strength or amplitude of connection known as *synaptic weights*. Each neuron is called a node and the strength of connection between any two nodes is simply called as weight. Figure 1 shows two neurons connected with certain synaptic weight (through synapse).

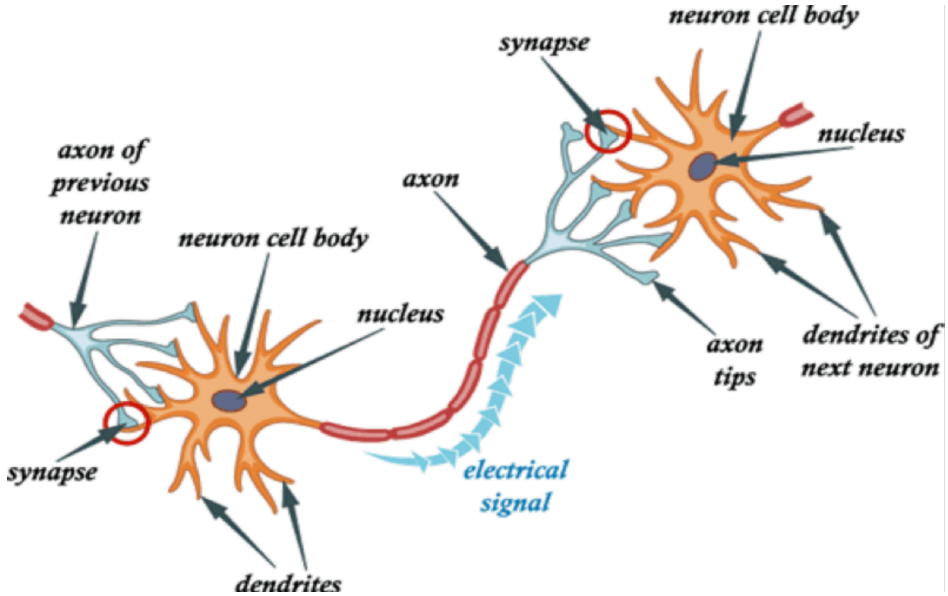


Fig 1: Connection between two neurons through synapse

One of the frequently used operation in a neuromorphic computation is Multiply and Accumulate operation, known as *MAC operation*. Input is multiplied with the synaptic weights and the result is accumulated to compare it with a threshold value. If the accumulated value is greater than the threshold value, then a spike is sent to the next neuron or node to establish the connection. If the accumulated value is less than the threshold value, then there is no spike or communication from one node to the other.

The above-mentioned functionality can be expressed mathematically using matrix form. A vector or set of inputs 'X' is multiplied by synaptic weights 'W', where both input 'X' and weight 'W' can be floating numbers. It can be expressed in matrix form as follows:

$$[X_0 \quad X_1 \quad . \quad . \quad . \quad X_n] * \begin{bmatrix} W_{11} & W_{12} & . & . & . & W_{1n} \\ W_{21} & W_{22} & . & . & . & W_{2n} \\ . & & . & . & . & . \\ . & & . & . & . & . \\ W_{n1} & W_{n2} & . & . & . & W_{nn} \end{bmatrix} = \begin{bmatrix} Y_0 \\ Y_1 \\ . \\ . \\ Y_n \end{bmatrix}$$

where, $Y_0 = X_0W_{11} + X_1W_{21} + \ldots + X_nW_{n1}$

$Y_1 = X_0W_{12} + X_1W_{22} + \ldots + X_nW_{n2}$

. .

. .

$Y_n = X_0W_{1n} + X_1W_{2n} + \ldots + X_nW_{nn}$

Output 'Y' is called accumulated weight or simply '*weighted sum*'

Over the past few decades, Moore's law has been consistent and has been helping digital logic technology to implement the neural networks successfully. This has led to tremendous change in our day to day lifestyle due to advancement in smart phone and smart devices utilizing artificial intelligence. However, as the neural networks grow complex or becomes deep (multi layered network), scaling of the devices becomes difficult and expensive. Due to high leakage in the advance nodes, power consumption becomes a dominant factor and impedes the scaling of the network.

Traditionally, MAC operation in a digital implementation of the neuromorphic systems are facilitated by an external large memory chip. As the size of the network grew, latency and power consumption for fetching/storing the data to and from an external memory chip became expensive. Clearly, there is a growing demand for high density, low area, non-volatile memories (NVM).

Few of the important challenges in a neuromorphic computation are matching a human's ability to learn and adopt from amorphous stimuli with energy efficiency of the human brain. Charge Trap Transistor (CTT) is an emerging non-volatile memory that requires no additional processing steps, hence is extremely cost effective compared to other traditional non-volatile memory (NVM) such as memristors [1], phase change memory [2], flash and so on.

II. System Architecture:

Our project aims at recognizing handwritten numbers as shown below in figure 2:

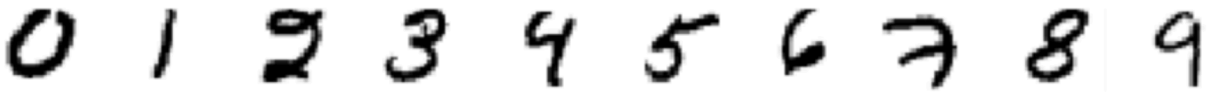


Figure 2: Handwritten numbers

It is very easy for a human brain to recognize these numbers but becomes difficult to program a computer to detect them. Algorithms to detect a simple number like '5' or '6' becomes complex as there are loops and strokes involved in them. When these complex algorithms have to be fast and power efficient, they tend to fall apart.

However, neural networks have a completely different approach to solve these kinds of problems. These networks take pre-defined set of images with labels on them as *'training data'* and tend to learn to recognize the numbers with the help of training data set. In other words, the network infers the rules to detect handwritten numbers using training data set. This implies, as the training data grows, the accuracy of detection goes high.

As mentioned in section I, on chip memory for storing weights is extremely important, as they scale better compared to a digital system with weights stored external to the processor chip. Choice of Charge Trap Transistor (CTT) [3] is a perfect fit for the scalability of the neural networks for the following reasons:

1) CTT acts as an analog memory, where a single MOSFET is capable of handling 8-bit value.

2) Charge trapping mechanism is enabled by the hafnium oxide present between the gate ploy and the channel.

3) In technology nodes less than 28nm, most of the processes have hafnium oxide, and thus there is no need to change any processing steps. This makes CTT cheaper compared to other NVMs.

4) As we have a mixed signal chip, digital controller for the chip is huge and the power consumption/leakage power affects the efficiency. Hence 22nm FDSOI is a perfect technology node for us to use.

5) Fully depleted silicon on insulator reduces the leakage factor by at least 10, when compared to bulk technologies.

6) Conductance of the CTT (by changing V_{th}) can be altered by programming them with voltages much less than flash NVM.

7) Further, as the technology scales down, the analog mixed signal circuits tend to become compact and power efficient.

All the above-mentioned reasons make 22nm FDSOI technology a perfect match for our neural network implementation.

We have four different modes of operations in order to use CTT in a neural network system. Frist, the synaptic weights has to be stored in the CTT memory cell, which is achieved by *programming* these devices. Next, weights stored in the CTT memory cell has to be verified, which is done by verifying the currents flowing through them under certain bias condition. This mode of operation is called *verification* mode. Once all the weights are programmed and verified, it can be used as an inference engine, and is called *inference* mode of operation. One of the biggest advantages of the CTT device is that the trapped charges can be de-trapped. This way, the programmed weights can be erased in the *erase* mode of operation.

CTT conductance can be altered by changing their V_{th} and the process of changing the V_{th} is called *programming [4]*. During programming, charges are trapped inside the hafnium oxide layer, which leads to higher gate voltage requirement for the inversion to take place. As shown in figure 3, this trapping of charge is achieved by applying high gate voltage train of pulse of about ~1.5V - 2.7V, when the drain voltage is held at about 1V - 1.5V.

Once the desired programming is achieved, it is verified using verification mode. Here, the neuron or the weighted sum calculator is forced to apply a drain voltage of about 200mV and the current through the programmed CTT device is read while gate voltage is about 300mV. If the desired current is not met, then the CTT device is either programmed again or erased based on the current level. This is shown in figure 3.

Inference mode of operation is similar to verification mode. Here the neuron applies 200mV at the drain of the CTT memory cell, while a PWM signal of 300mV amplitude is applied based on the input pattern. The current produced by all the CTT memory cells are then integrated on a capacitor to produce a weighted sum. Further details about inference mode is explained in section III.

In the erase mode of operation, a negative gate voltage of ~1V is applied to the gate, while the source and drain is held at ground. This leads to de-trapping of charges from the hafnium oxide, decreasing the V_{th} and increasing the conductance. This way weights can be fine-tuned on the CTT memory cell.

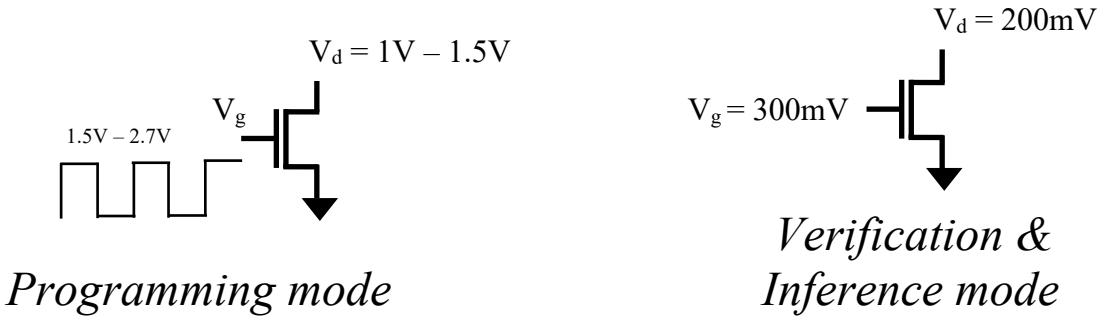


Figure 3: CTT mode of operation

A typical memristor or flash memory implementation of neural network consists of digital input, which is then converted to analog inputs before applying it to the synapse. The current generated by these devices (synapse) is proportional to the

conductance of the devices which can be altered by programming them in a specific pattern. Output current through the large array of synapse is integrated and converted back to digital signal. Figure 4 shows a typical analog implementation of the neural networks [5].

This type of implementation has several advantages and dis-advantages [6]. This system is not scalable beyond a point due to high power consumption and large area by ADC and DACs. Since the input applied is analog in nature, linearity is difficult to achieve, due to which the constraints on the ADC grows exponentially, leading to higher power consumption and larger area.

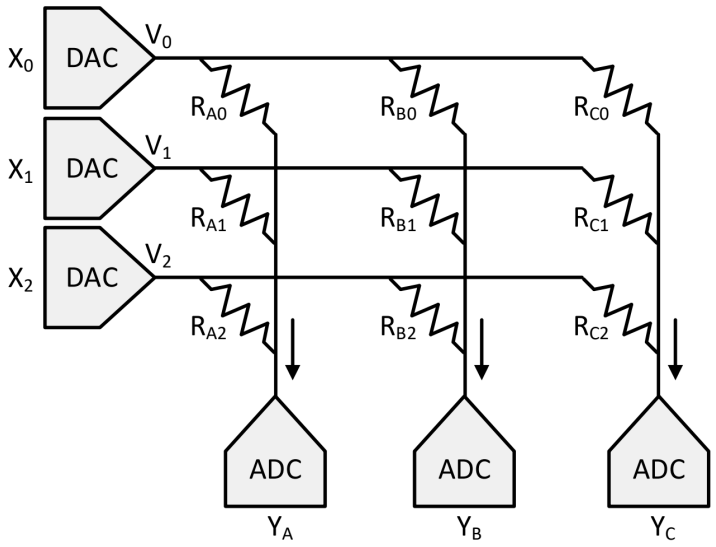


Figure 4: Analog implementation of neural networks

To overcome this draw back, we have used *Pulse Width Modulation* (PWM) scheme. All the digital inputs are mapped to a corresponding pulse duration at the gate of the CTT, which gives high linearity at the output (resultant current from any

CTT). This way, constraints on the current integrator is curtailed and the mixed signal integrator circuit can get away with smaller area.

To account for negative weights, we use twin cell architecture as shown in figure 5. If the weight is positive, then the right side CTT in the twin cell indicated by W^- is programmed higher to have a larger V_{th} (low conductance). This way, the differential current generated by the twin cell flows from BLt to BLc as shown in figure 4. If the weight is negative, W^+ is programmed higher to have a larger V_{th} (low conductance). We have used SLVT MOSFET as CTT memory which gives about 8-bit of programming resolution.

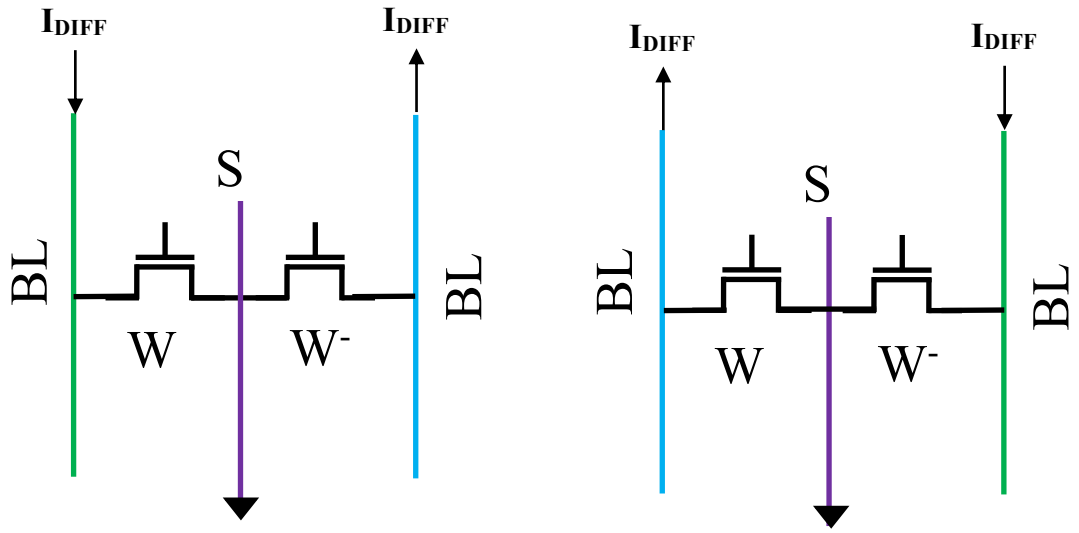


Figure 4: CTT twin cell architecture

All the required pulse width and pulse amplitude is generated using word line driver, indicated by 'WL'. This circuit

consists of set of level shifters with appropriate bias voltages, with which it is able to generate pulses of different amplitudes.

There are four important blocks in our NeuroCTT system, as shown in figure 5.

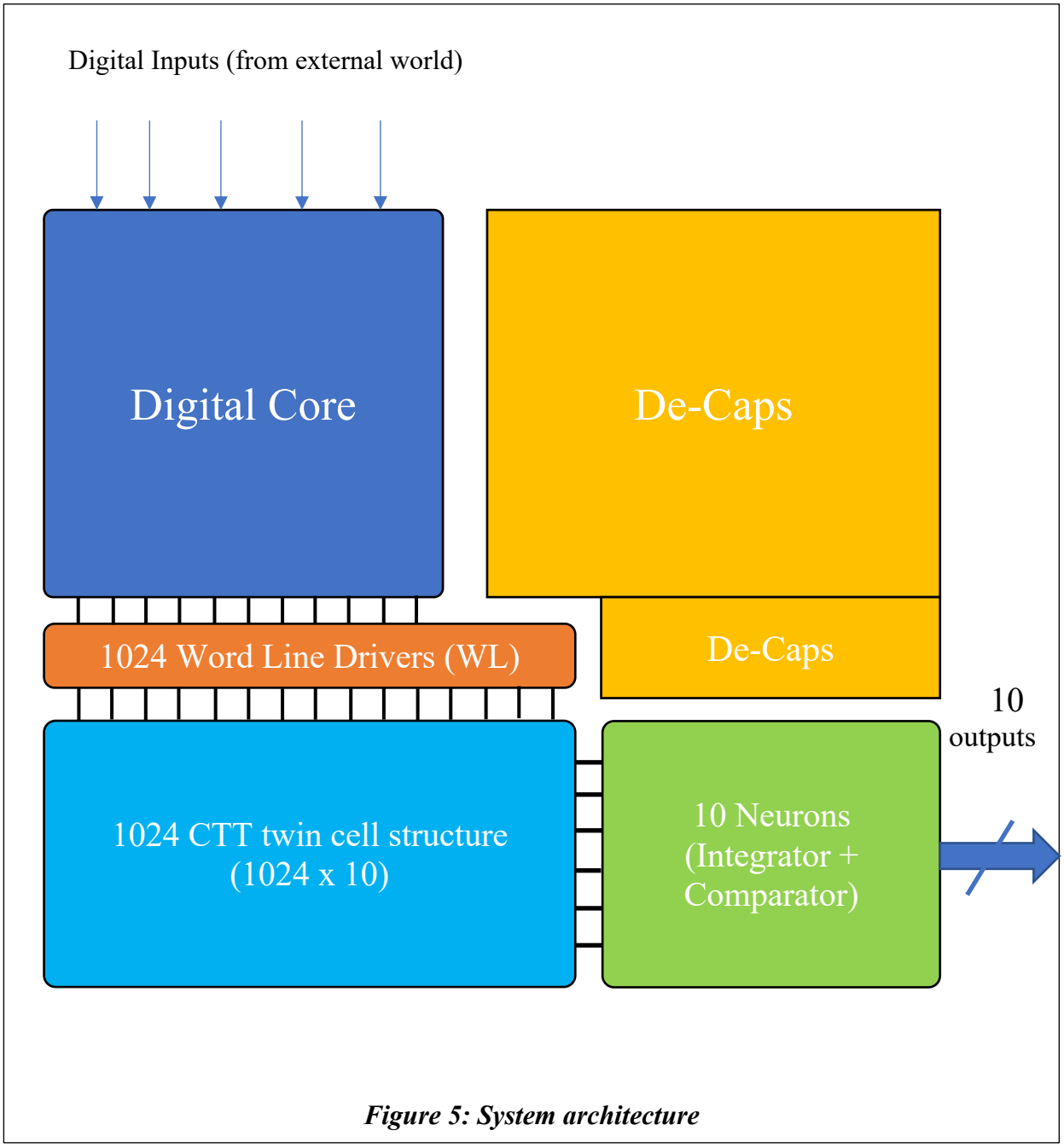


Figure 5: System architecture

Digital Core:

It takes in a high frequency clock along with the digital inputs. All the input pixels in an image is mapped to a digital input and is fed to the chip. Digital core block takes in serial inputs and converts it into parallel data which is necessary for the world line driver (WL).

Digital core has 1024 8-bit counter which produces output '1' until the counter is reset. Based on the digital input, counter is fed with a number between 0 to 128, and the counter starts to count down until it reaches 0 (reset state). This is how a pulse width modulated signal is generated.

Neurons have complicated digital signals since it works at high speed and is a mixed signal circuit. Digital core also provides all the necessary control signals to the neurons, which are all buffered before reaches its destination (inside the neuron).

Word line driver:

There are 1024 Word line drivers on chip. WL consists of set of level shifters which takes in digital inputs with certain pulse duration. During programming mode of operation, the WL driver has to provide a voltage of ~1.5V to 2.7V at the gate of the CTT twin cell. During inference and verification mode of operation, the WL driver should be able to provide 300mV at the gate (if the CTT has to be turned ON) and -300mV (if the CTT has to be turned

OFF completely). This is done by externally providing different bias voltages to the WL driver.

CTT twin cell

As mentioned above (Figure 4), twin cell CTT has two MOSFETs to handle positive and negative weights. All the synaptic weights are pre-determined and programmed on to these CTT devices. During programming mode, the current through the CTT which is being programmed could be as high as 1mA. In order to sink this high current, there are huge programming switches connected to BLt and BLc, which contributes to ~90fF to 95fF.

In order to protect the neurons from the high programming voltages at the BLt and BLc, there are protection switches in series with the differential current path. These switches are left floating during programming so that it protects itself and the neuron. In terms of layout considerations, to reduce the resistance, thick metals are used and stacked for BLt/c and the SLs.

Neurons:

Details about the neuron is presented in the next section.

De-caps:

Since WL driver, neurons and the digital core works at high frequency, a large amount of de-cap surrounding these blocks is necessary for smooth operation. Hence, MIM caps are used to create large de-caps using Metal 1 to Metal 5.

III. Design of integrator:

In order to understand why we need integrator; we need to understand what is a neuron? and what are its functionalities? Artificial neuron is called perceptron, developed in 1950s and 1960s by the scientist Frank Rosenblatt. A perceptron or a neuron takes several inputs denoted by X_1, X_2, X_3 … X_n as shown in figure 6 and produces a single binary output [7].

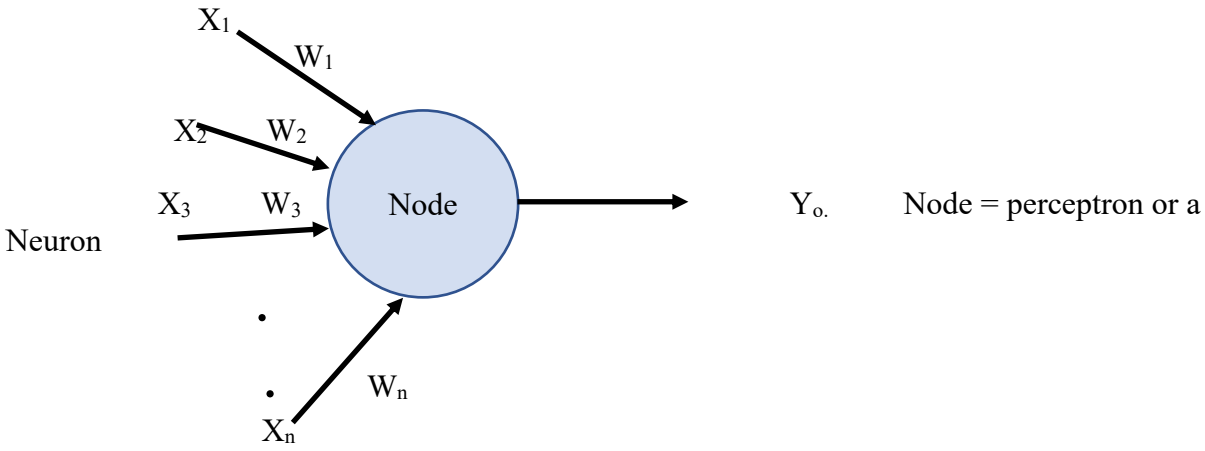


Figure 6: Perceptron or Neuron

Let us assume a node or a neuron that has three inputs X_1, X_2 and X_3. Now each arrow shown in figure 6 has certain weights (synaptic weights) associated with that and is indicated by W_1, W_2 and W_3. These weights are real numbers and it signifies the strength or the importance of that particular path (via input) to the output. For the simplest case of the perceptron, output of the neuron/perceptron is either '1' or '0' based on the weighted sum, which is mathematically expressed as $\sum(Xi * Wi)$.

If the weighted sum is less than a certain threshold number (which is a real number again), the output of the neuron is binary '0'. If the weighted sum is greater than the same threshold value, then the output of the neuron is binary '1'. So, the threshold value is one of the parameters of the neuron and is called hyper-parameter.

$$\text{Output} = \begin{cases} 0 & \text{if } \sum(Xi * Wi) \leq \text{threshold} \\ 1 & \text{if } \sum(Xi * Wi) \geq \text{threshold} \end{cases}$$

Let's denote the threshold value by 'b'. Simplifying the above equation, the threshold value can be moved to other side, which gives us an equation

$$\text{Output} = \begin{cases} 0 & \text{if } \sum(Xi * Wi) + b \leq 0 \\ 1 & \text{if } \sum(Xi * Wi) + b > 0 \end{cases}$$

The threshold 'b' is often known as bias and it can be thought as the easiness with which a neuron fires an output '1'. From the above equation, we can say that for a bigger bias term, it is easy for a neuron to fire an output '1'. Let us take an example

and build NAND gate using a neuron (since NAND gate is universal)

Let's suppose we have a perceptron/neuron with two inputs, each with weight −2 and −2, and an overall bias of 3. Figure 7 shows the neuron for implementing the universal NAND gate.

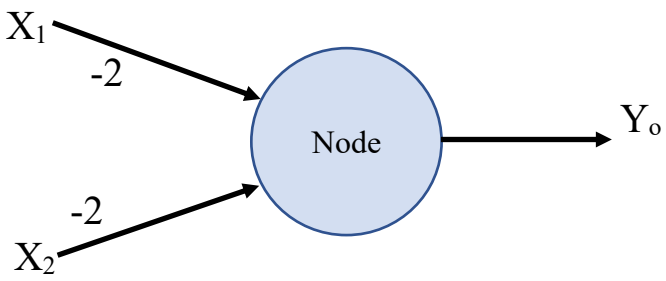


Figure 7: NAND gate using a neuron

If an input of (0,0) is applied, then the neuron will output '1'

$$(-2) * 0 + (-2) * 0 + 3 = 3 > 0$$

Similarly, if input (0,1) and (1,0) is applied, the weighted sum plus the bias term is greater than zero. Hence all these inputs (00, 01, and 10) will produce a binary output '1'. But if (1,1) is applied, we get a negative weighted sum:

$$(-2) * 1 + (-2) * 1 + 3 = -1 < 0$$

And thus, our neuron has implemented a NAND gate!

For now, we know that the neuron has to compute the weighted sum, which is nothing but an integrator! [8] In our CTT

case, various current pulses with different current amplitudes are generated based on the input pulse duration (generated by the counter in the digital core block) and the tuned conductance (synaptic weights) of the CTT device.

As our NeuroCTT system is a 1024 x 10 structure, there are 10 neurons in the system. As the network scales, the number of neurons will proportionally increase. Hence the requirement on the neuron's specifications are tight. Let us try to understand few critical specifications for the integrator. Figure 8 shows how the CTT memory cell is connected to the neuron.

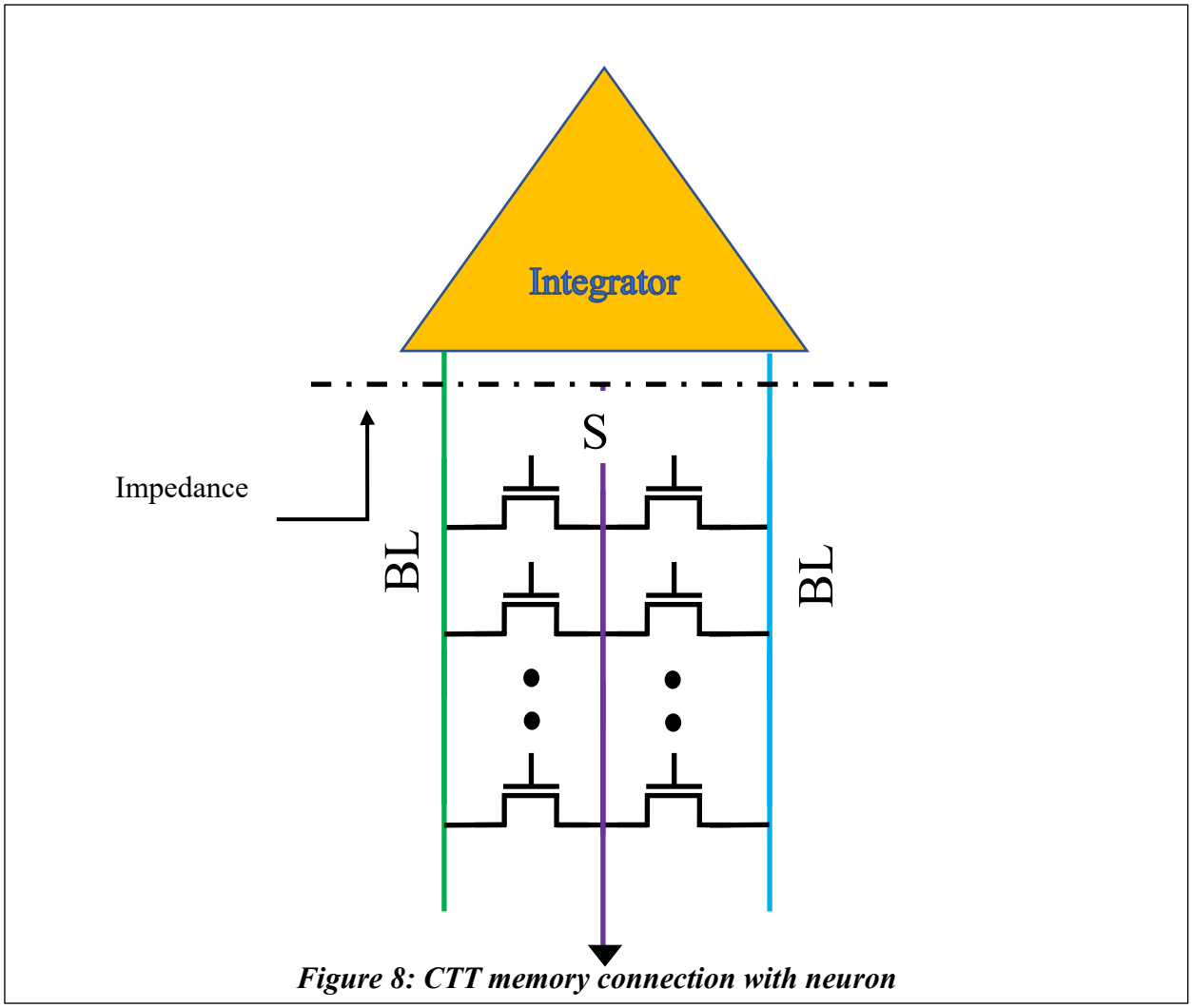


Figure 8: CTT memory connection with neuron

If we calculate the impedance seen through the line as shown in figure 8, then the input impedance of the integrator will be in parallel with the impedance of the CTT devices. If CTT device in inference mode of operation is modelled as a current source with leaky parallel impedance, then it is clear that the input impedance of the integrator has to be much less than the impedance of the CTT devices.

If the input impedance of the integrator is not less than the CTT device impedance, then most of the current generated by the CTT current source will be leaked through its own leaky parallel resistance. Hence, the input impedance of the integrator has to be at least 10 times less than that of the CTT devices.

In the calculation of energy consumed per MAC operation, most of the energy is spent in the integrator, since it has amplifiers which requires bias currents. Hence reducing the power consumption of the neuron in order to achieve energy efficient MAC operation is important. Further, as the system grows, the number of neurons in the network also grows proportionally. Hence, in order to have a solution which is scalable, reducing the power consumption in the neuron is critical.

Furthermore, we know that the current integration is done on a capacitor which is usually area consuming. As the system scales, the number of capacitors required in the system also increases proportionally. Therefore, the size of the circuit except the capacitor has to be small in order to reduce the area. Since

there will be more than one pole and we expect at least one zero in the system to make it stable, the requirement of compensation techniques comes in to picture. This means that the compensation circuit for the analog integrator has to be as small as possible (since compensation circuits are usually built using resistors and capacitors).

Interestingly, there is a trade-off between the size of the capacitor required for the integration, the speed of the circuit, the area, and the power consumption [9]. Since we are using pulse width modulation (PWM) technique, we can reduce the size of the capacitor by running the integrator very fast. This means that the high-speed operation will reduce the area of the integrator by reducing the size of the capacitor but will inevitably increase the power consumption. This is summarized in figure 9.

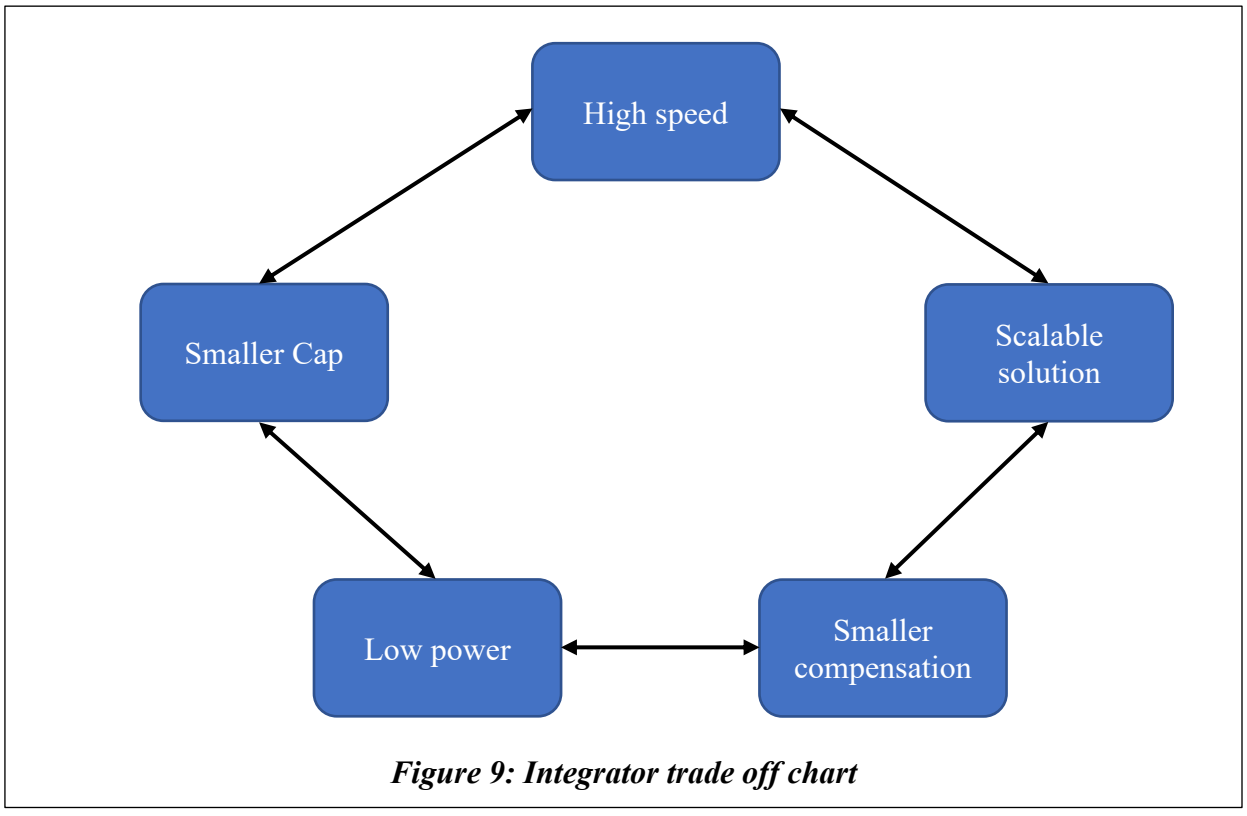


Figure 9: Integrator trade off chart

Traditional integrators:

A differential current integrator [10] has two inputs and two outputs as shown in figure 10. Two currents I_1 and I_2 flows into the input terminals 'plus (+)' and 'minus (-)' as shown below. From these currents I_1 and I_2, differential and common mode currents are determined as:

$$I_{CM} = (I_1 + I_2)/2$$

$$I_{DM} = (I_1 - I_2)/2$$

Hence, I_1 and I_2 can be expressed as:

$$I_1 = I_{CM} + \frac{I_{DM}}{2}$$

$$I_2 = I_{CM} - \frac{I_{DM}}{2}$$

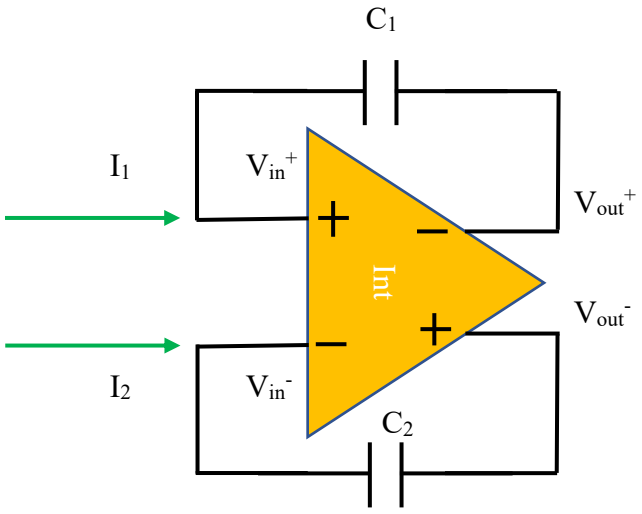


Figure 10: Traditional integrator

A traditional integrator consists of two integrating capacitors C_1 and C_2 as shown in figure 10. Current I_1 flows into the capacitor C_1 and current I_2 into the capacitor C_2. Hence, the voltage on the capacitor is also proportional to I_1 and I_2, which means it consists of differential and common mode components. Let's express the same in terms of voltages as:

$$V_{out}^{+} = V_{CM} + \frac{V_{DM}}{2}$$

$$V_{out}^{-} = V_{CM} - \frac{V_{DM}}{2}$$

Hence, when the differential output voltage is taken, that is:

$$V_{OUT} = V_{OUT}^{+} - V_{OUT}^{-}$$

The common mode component will be cancelled out and the differential mode components add together to produce:

$$V_{OUT} = V_{DM.}$$

Usually C_1 will be equal to C_2. In case of PWM technique, size of the capacitor is inversely proportional to the speed of operation (frequency of operation). Hence, in case of traditional integrators, if the frequency of operation is reduced by a factor of two (in order to save the power), the size would increase by a factor of four (since there are two capacitors whose size is inversely proportional to the frequency of operation). This is a huge set-back for scalable systems.

Hence, for an application like neuromorphic computation, which demand scalability of the system, there is a need for more robust trade off solutions.

NeuroCTT Integrator:

To overcome the above-mentioned drawbacks, we have developed an integrator which has better trade-off matrix compared to the traditional integrators. Figure 11 shows the block diagram of our integrator where we are using only one capacitor to integrate the differential current and produce a differential output voltage.

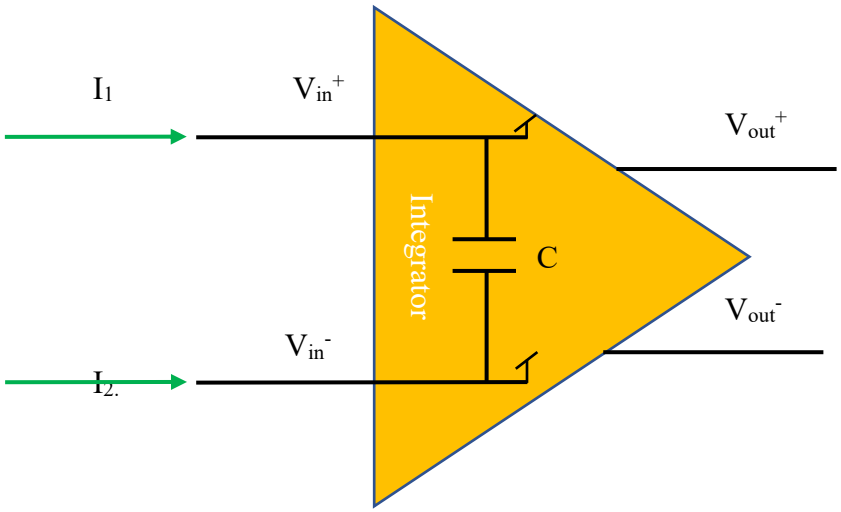


Figure 11: NeuroCTT Integrator with only one integrating capacitor

Now, let us look at all the important specifications (as discussed above) of this integrator.

Let us re-visit the output differential equation as shown below:

$$V_{out}^{+} = V_{CM} + \frac{V_{DM}}{2}$$

$$V_{out}^{-} = V_{CM} - \frac{V_{DM}}{2}$$

From this differential output voltage equations, if V_{CM} part is not computed, then we need only one capacitor with its value equal to half that of a traditional integrator. Since we have twin cell for the CTT memory, common mode current flows in only one direction through both BLt and BLc. However, the direction is opposite for differential currents as shown in figure 12.

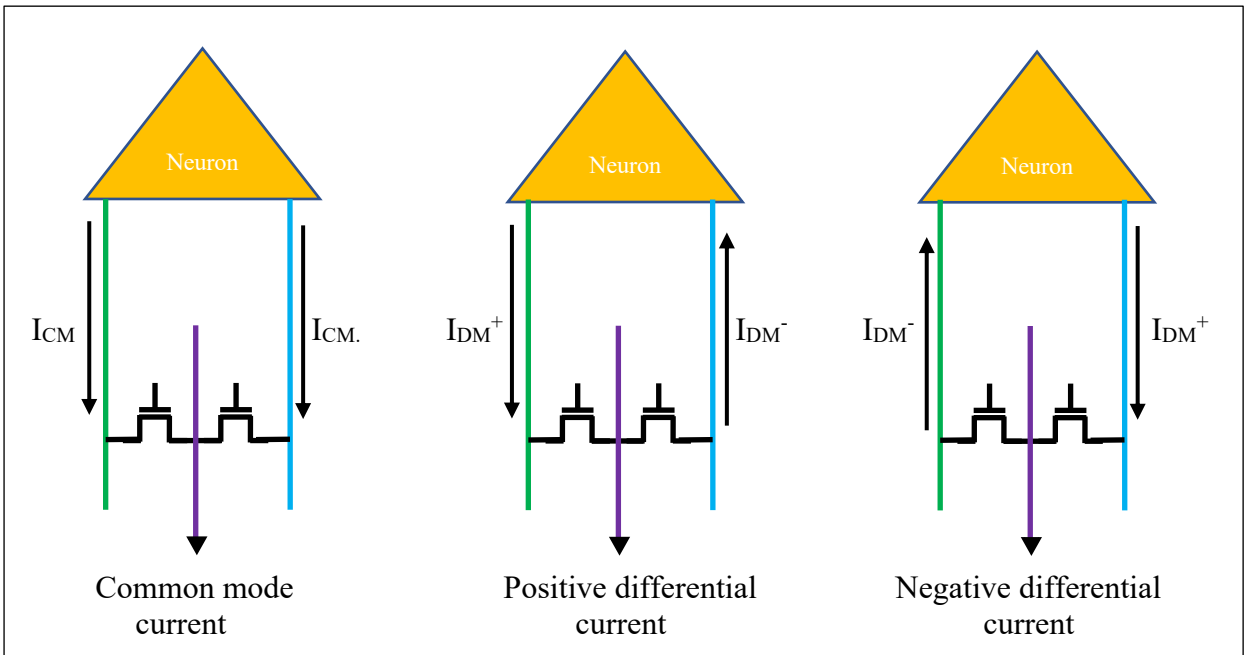


Figure 12: Common mode and differential mode current direction

If we take a capacitor and pump equal currents through its terminal in the same direction as shown in figure 13, there won't be any voltage developed across the capacitor. However, if equal and opposite currents are pumped through the therminals of the capacitor, then there will be a potential difference across the plates of the capacitor.

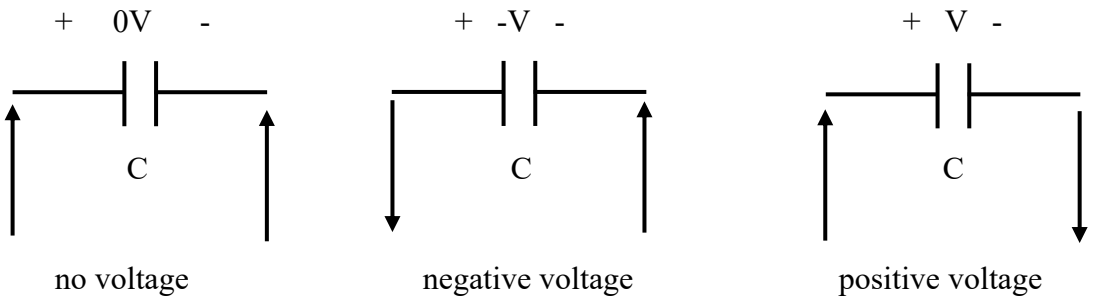


Figure 13: Voltage across capacitor with common mode and differential mode currents

This is how we have an integrator whose area is 4 times less then that of a traditional integrator. Even if the frequency of operation is reduce by a factor of two to reduce the power consumption, the area of the integrator is still half compared to a traditional integrator. This is a huge improvement in the trade off matrix except input impdenace. Now let us look at how to reduce the input impedance of the integrator.

Among the three configuration of amplifiers, namely common source, common drain, and common gate: common gate amplifier has the least input impedance. Further, if we add a gain loop, input impdance can be further reduce by a factor of (1 + A) times, where A being the open loop gain of the amplifier.

Figure 14 shows the CG amplifier with a gain loop and its small signal equivalent circuit.

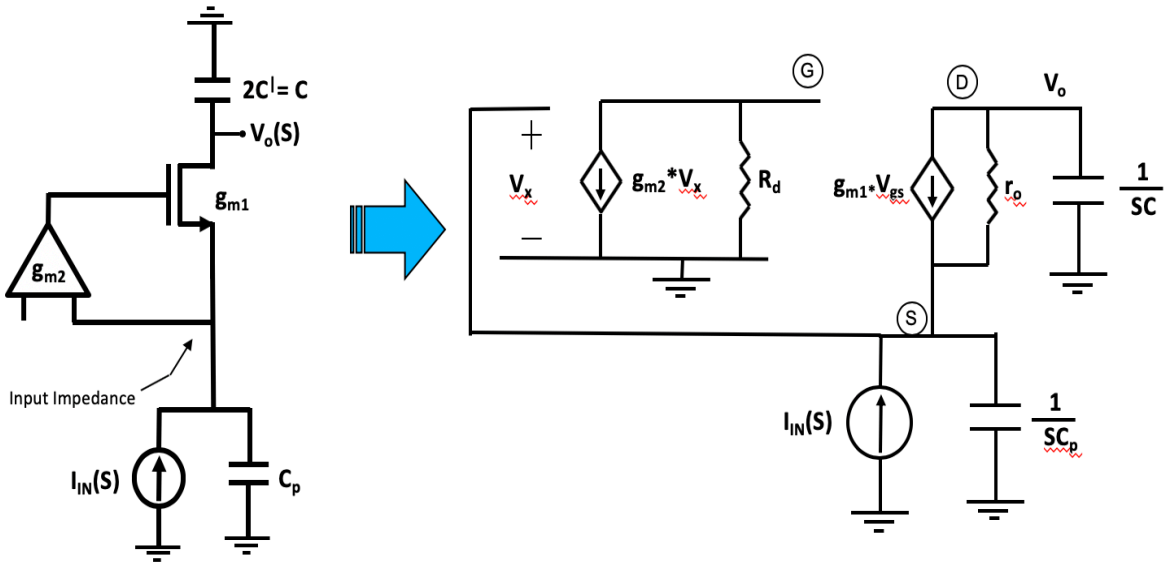


Figure 14: CG amplifier and its small signal equivalent circuit

Extending the same concept for fully differential amplifier, we can share the output capacitor $2C^{|} = C$ as shown in figure 14 with the other half of the fully differential amplifier. In order to provide a current path for the CTT memory cell, we add two current sources 'I_B' as shown in figure 15. This current source also provides necessary bias currents to MOSFET M_1 (figure 15) which increases the g_m of the circuit.

We will look at the input impedance calculation for this particular architecture in detail. But for now, in order to understand the importance of the current source, let's assume ~ $1/g_m$ is the input impedance. Hence, higher g_m improves the input impedance.

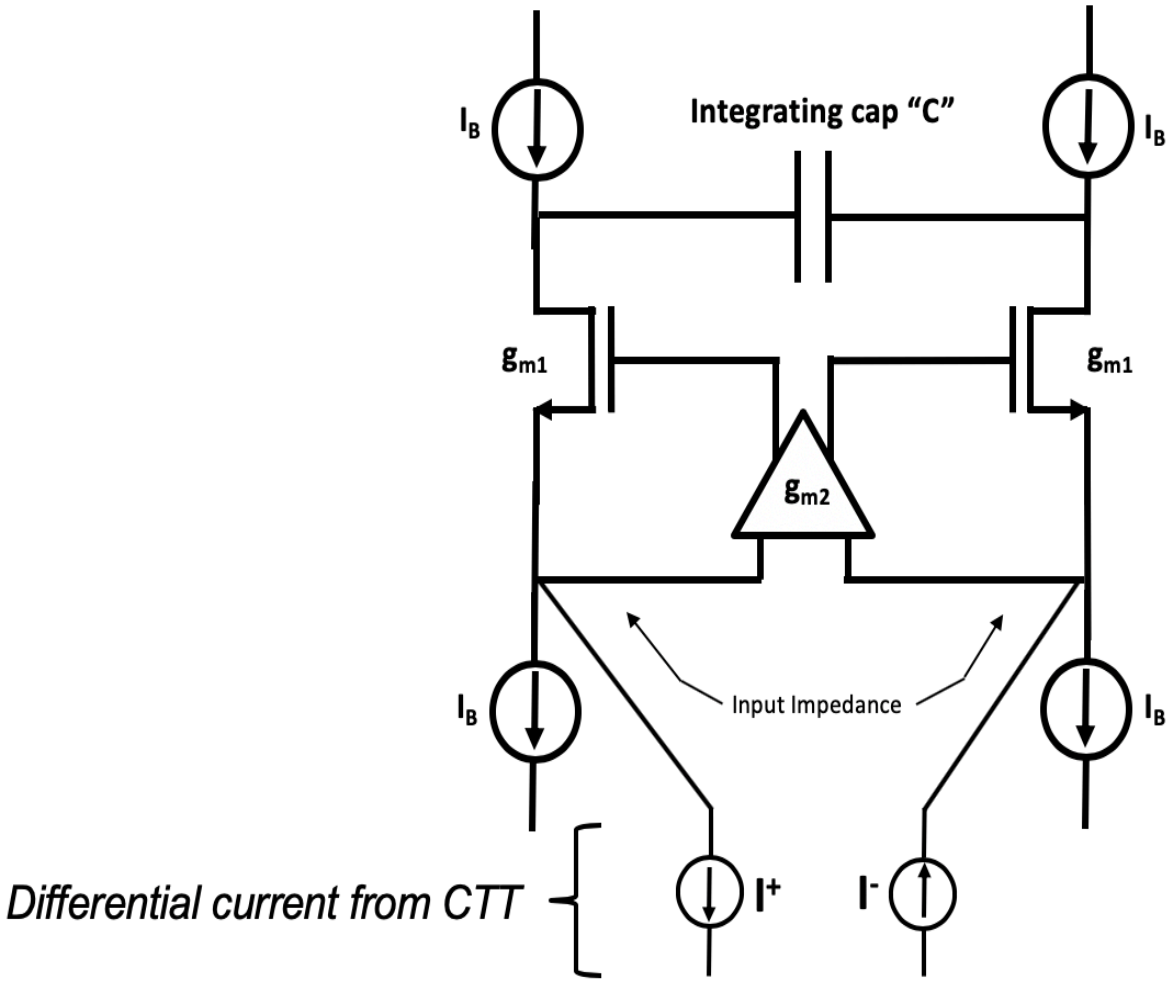


Figure 15: Fully differential integrator

Now, let's look at how high the bias current sources 'I_B' can go. From the initial argument, higher the bias current I_B, higher the g_m of the MOSFET M_1, which implies lower the input impedance. But, if I_B is doubled, size of the MOSFET M_1 also has to be doubled in order to keep the same bias conditions (saturation region of operation). This means that the parasitic gate capacitance will also increase by a factor of two.

We now know that, to reduce the size of the circuit, we need as less compensation as possible. If 'I_B' increases to an extent (along with the size of MOSFET M_1 in figure 15), such that the parasitic gate capacitance (C_g) pole comes inside the bandwidth, then we need a compensation circuit to introduce a zero and make the system stable.

In order to avoid compensation circuit all together, the open loop gain 'g_{m2}' (figure 15) is small enough to provide large bandwidth. This way, there is no compensation circuit (resistor and capacitor) in our integrator which leads to lower area. Let's take a closer look at our integrator as shown in figure 16.

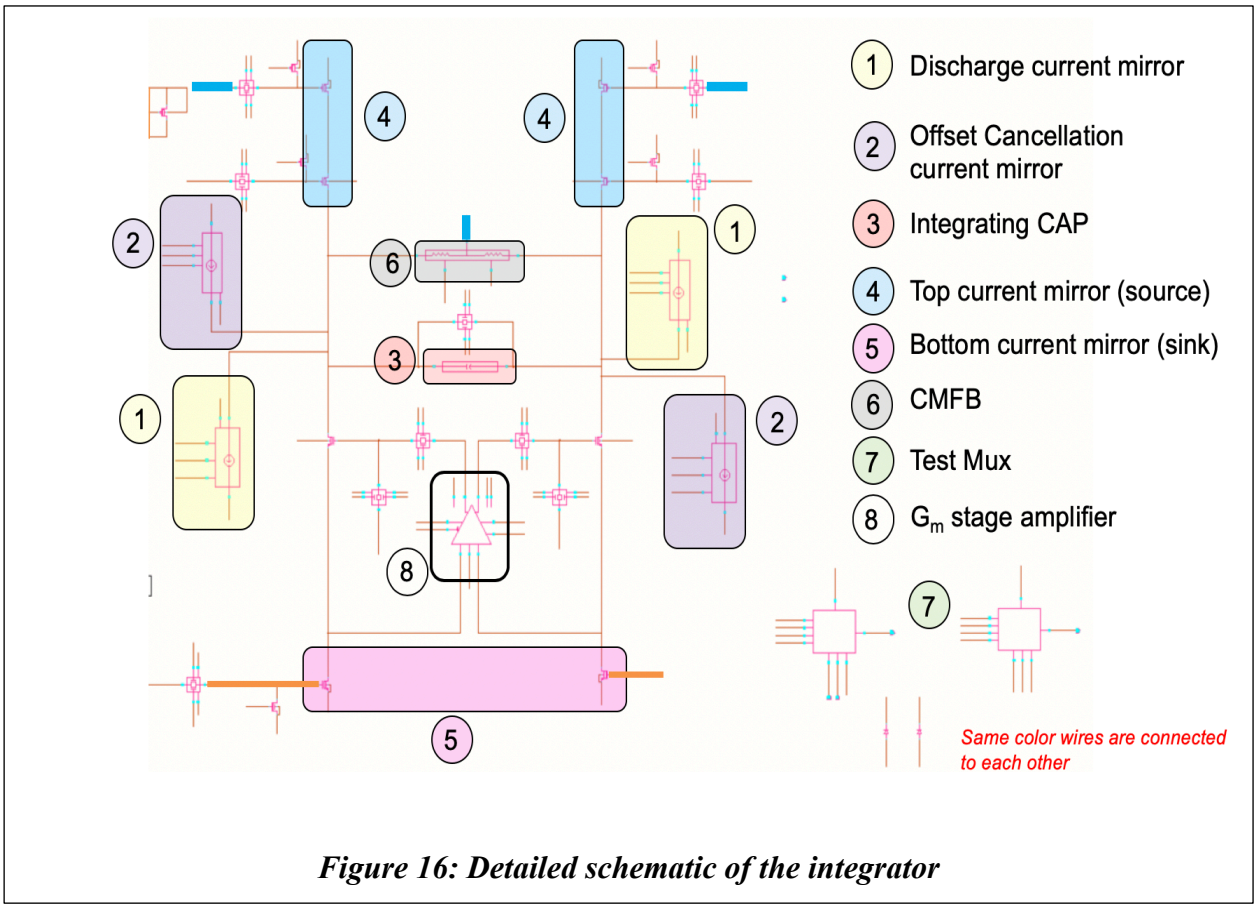


Figure 16: Detailed schematic of the integrator

Figure 17 and 18 shows the schematic of the loop gain amplifier which doesn't have any compensation circuit.

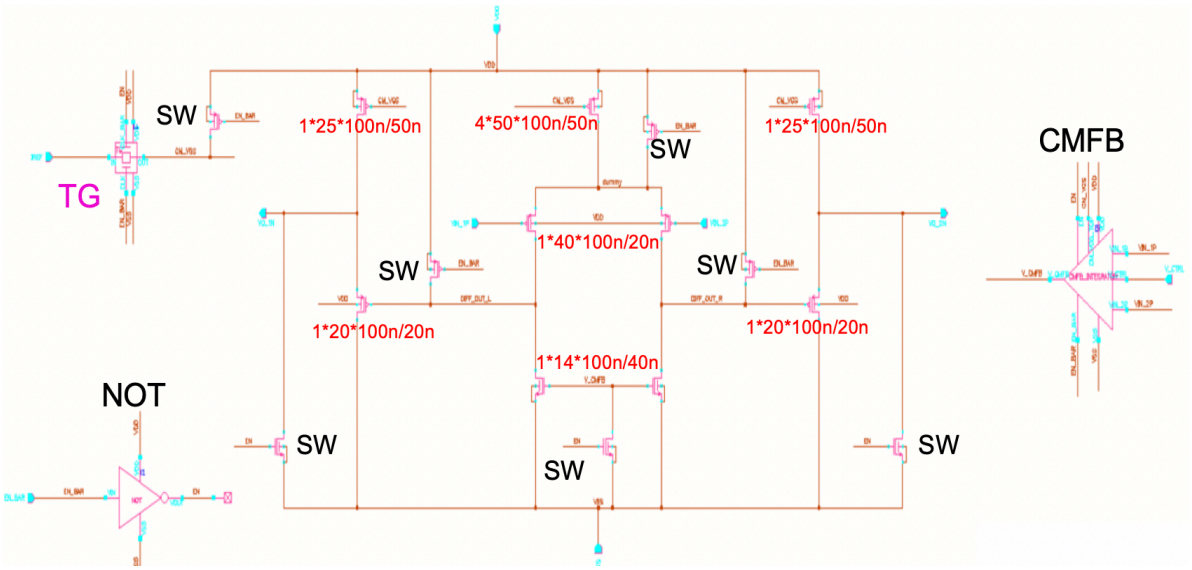


Figure 17: Schematic of loop gain amplifier

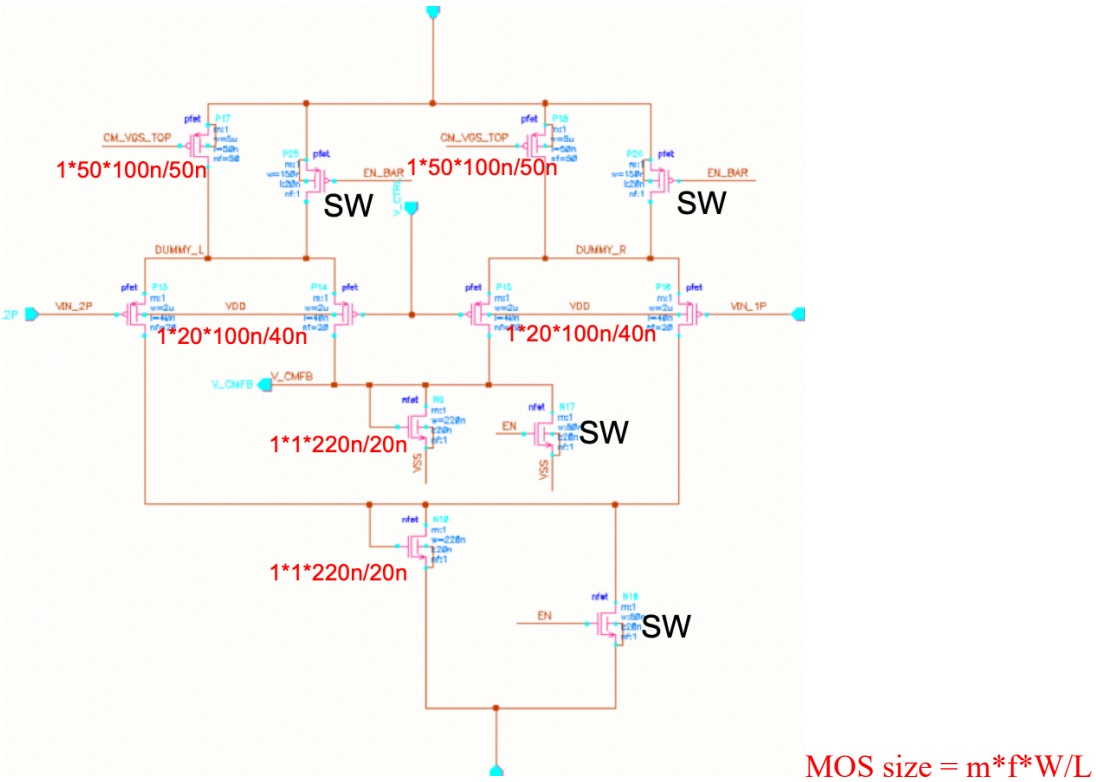


Figure 18: Schematic of common mode feedback amplifier

Input impedance

Now let's calculate the input impedance and carry out pole-zero analysis. Let's re-visit the half circuit of the integrator along with its small signal mode as shown in figure 19.

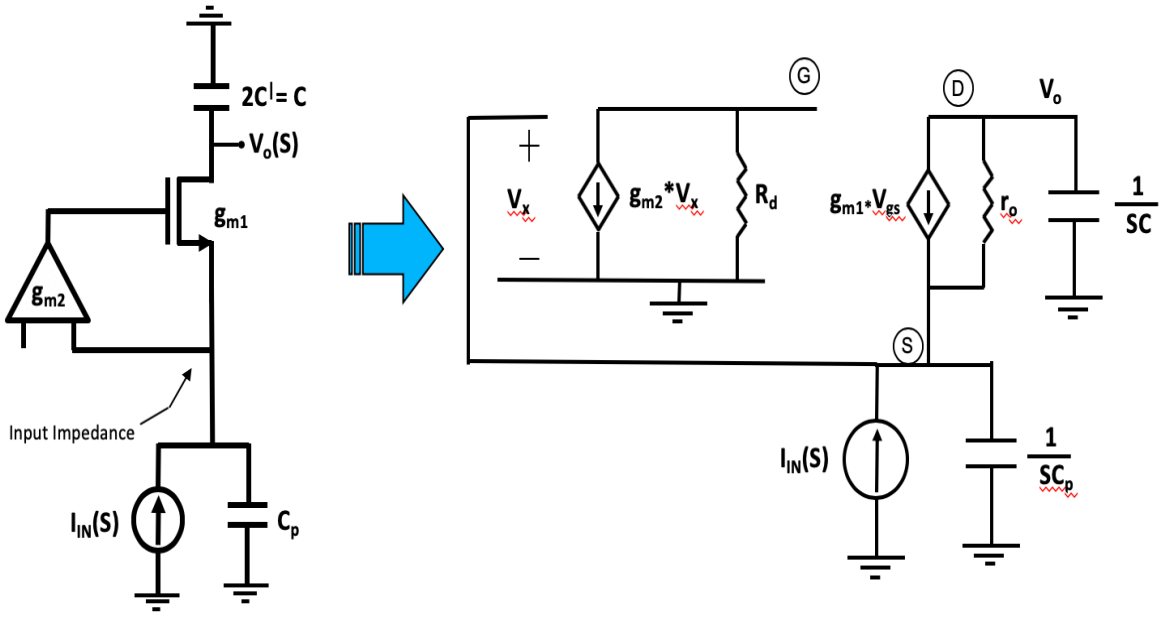


Figure 19: Integrator's half circuit and small signal model

For input impedance calculation, replace the current source $I_{IN}(s)$ with a known voltage source $\mathbf{V_X}$ and measure the current $\mathbf{I_X}$ flowing through it.

- **Case (1):** Ignoring $\mathbf{r_o}$ of the transistor

$$\Rightarrow g_{m1} * V_{gs} = g_{m1}[-g_{m2} * V_x * r_d - V_x]$$

$$\Rightarrow \frac{I_x - [g_{m1}[g_{m2} * r_d + 1] * V_x] = V_x}{SC_p}$$

Since we have negative feedback,

$$\frac{Vx}{I_x} = \frac{1}{g_{m1}[g_{m2}*r_d + 1] + SC_p}$$

- **Case (2):** *Considering* $\boldsymbol{r_o}$ *of the transistor*

$$\Rightarrow \quad g_{m1}*V_{gs} = g_{m1}[-g_{m2}*r_d*V_s(s) - V_s(s)]$$

$$\Rightarrow \quad V_o(s) = \frac{-g_m*V_{gs}}{SC}$$

$$\Rightarrow \quad V_o(s) = \frac{+g_m*\{V_s(s)[A+1]\}}{SC} \qquad \text{Where } A = -g_{m2}*r_d$$

$$\Rightarrow \quad V_s(s) = \frac{I(s) + (g_m*V_{gs})}{SC_p} = \frac{I(s) - \{g_m*V_s(s)[A+1]\}}{SC_p}$$

$$\Rightarrow \quad I(s) = V_s(s) * [SC_p + g_m*(A+1)]$$

$$\Rightarrow \quad V_s(s) = \frac{I(s)}{SC_p + g_m*[A+1]}$$

$$\Rightarrow \quad V_o(s) = \frac{g_m}{SC} \frac{I(s) * [A+1]}{[SC_p + g_m*(A+1)]}$$

Therefore,

$$\frac{V_o(s)}{I(s)} = \frac{g_m}{SC}\,\frac{[A+1]}{[SC_p + g_m*(A+1)]}$$

$$\tau = \frac{C_p}{g_m*[A+1]}$$

Noise derivation

Referring to the small signal circuit in figure 19, let consider $I_n(s)$ as a noisy current source at the output with respect to ground.

$$I(s) = V_s*SC_p + g_{m1}*V_s*(A+1) \quad (1)$$

Where $A = g_m * r_d$

$$V_o*SC = [g_{m1}*V_s*(A+1)] + I_n(S) \quad (2)$$

Using equation (1) & (2)

$$V_o(s) = \left[\frac{g_m*(A+1)*I(s)}{SC_p + g_m*(A+1)} + I_n(S)\right] * \frac{1}{SC}$$

Hence, the input referred noise is $\sqrt{4KT*\gamma*\left(\frac{2}{3}\right)*g_m*BW}$

Pole – Zero analysis

Referring back to figure 19, lets add parasitic capacitances at the nodes as shown in figure 20, to calculate pole - zero locations.

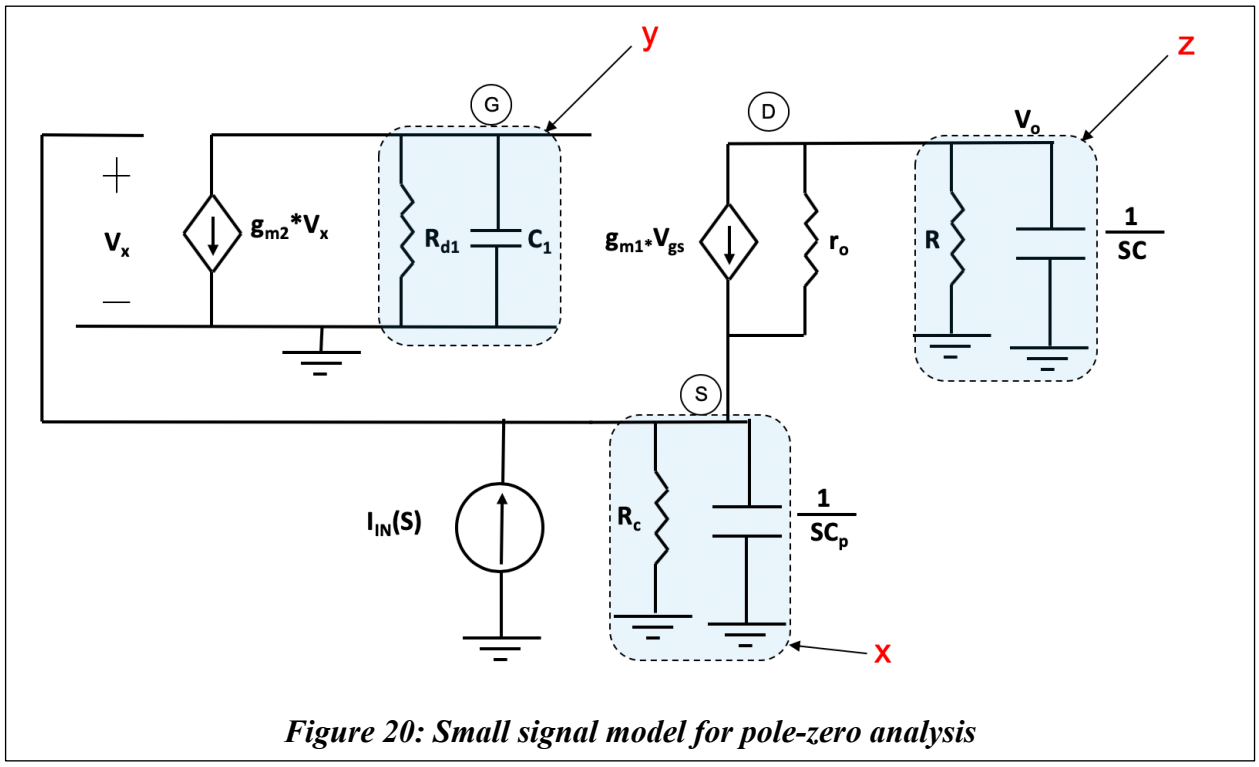


Figure 20: Small signal model for pole-zero analysis

In figure 20, notations x, y, and z represents parallel combination of respective 'R' and 'C'.

$$\Rightarrow V_{gs} = g_{m2}*V_x*y - V_x \quad \text{——} (1)$$

$$\Rightarrow I(s) + g_{m1}*Vg_s = \frac{V_x}{x} \quad \text{——} (2)$$

$$\Rightarrow V_o(s) = - g_{m1}*V_{gs}*z = z*V_x*g_{m1}*(g_{m2}*y + 1) \quad \text{——} (3)$$

Using equation (1) & (2)

$$V_x(s) = \left[\frac{I(s) * x}{1 + x * g_{m1} * (g_{m2} * y + 1)} \right]$$ → Substituting in equation (3)

$$\Rightarrow \frac{V_o(s)}{I(s)} = \frac{z * g_{m1} * (g_{m2} * y + 1) * x}{1 + g_{m1} * (g_{m2} * y + 1) * x} \quad (4)$$

Where,

$$\Rightarrow X = \frac{R_c}{1 + SR_cC_p} \qquad \Rightarrow y = \frac{R_{d1}}{1 + SR_{d1}C_1} \qquad \Rightarrow z = \frac{R}{1 + SRC}$$

Substituting x, y, and z in equation (4)

$$\Rightarrow \frac{V_o(s)}{I(s)} = \frac{g_{m1} * R * R_c (g_{m2}R_{d1} + SC_1R_{d1} + 1)}{(1+SRC)\left[g_{m1} * R_c \left[R_{d1} * (g_{m2} + SC_1) + 1\right] + \left[(1 + R_cSC_p) * (1 + R_{d1}SC_1)\right]\right]}$$

In total, the integrator burns ~250µA of current, operating at 0.9V of nominal VDD. Thus, it burns 225µW of power and achieves 6-bit resolution. In this version of the chip, we have only 300fF of integrating capacitor and hence has only 6-bit resolution. This is sufficient for our targeted application MNIST. This integrator occupies 283µm x 3µm.

Pin diagram and power up sequence

There is a total of 24 pins for this integrator including power, input and output pins. Each pin functionality is described in table 1.

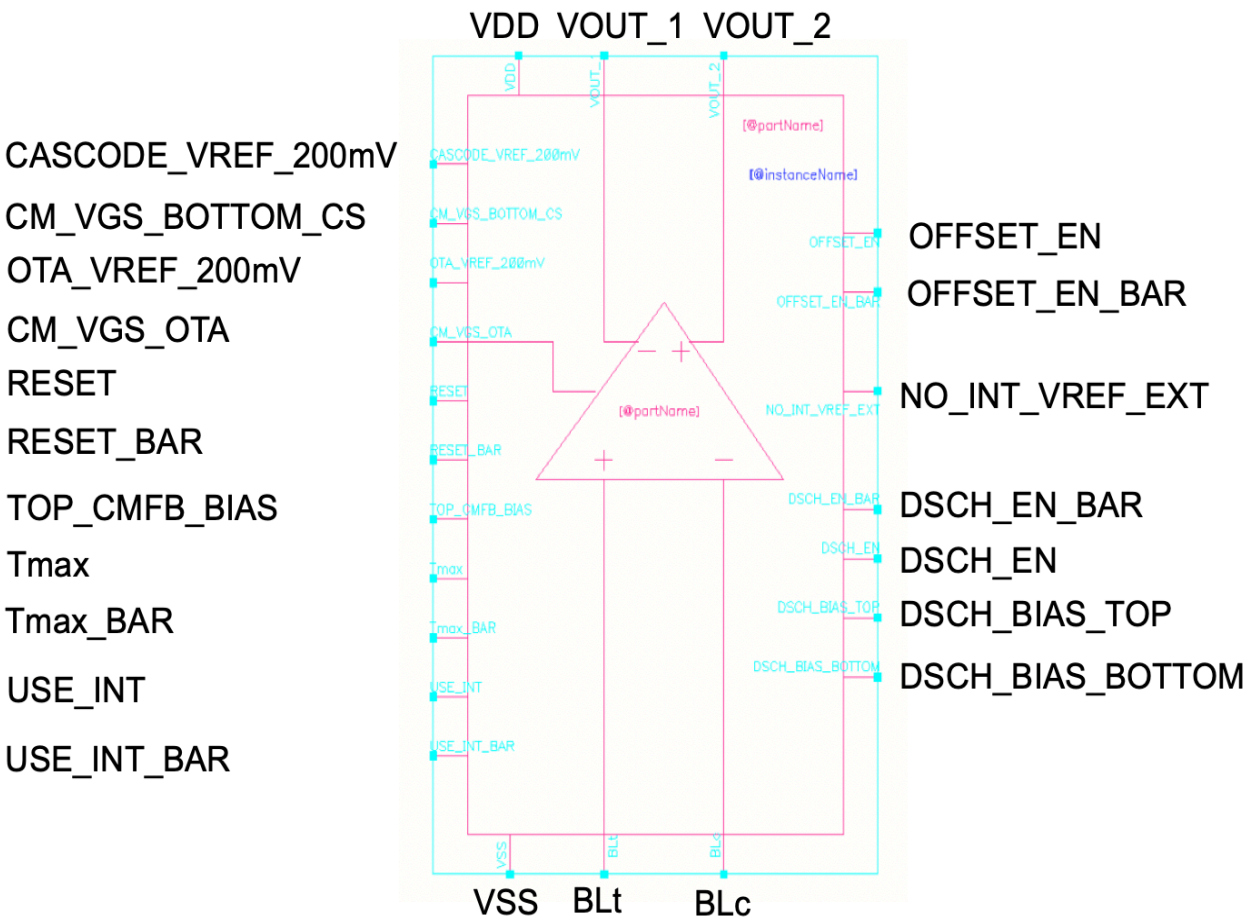


Table 1 summarizes functionality of each pins and describes if it's an input pin or an output pin.

Pin Name	Description	Direction	Domain	Comments
VDD	Power supply	INPUT (Analog)	0.9V	+/- 10% variation
VOUT_1	Positive node of the integrating Cap	OUTPUT (Analog)	0.9V	(VOUT_1 - VOUT_2) can be at max of 200mV for linear operation of the integrator
VOUT_2	Negative node of the integrating Cap	OUTPUT (Analog)	0.9V	
OFFSET_EN	Digital signals enabling the offset current mirrors	INPUT (Digital)	0.9V	
OFFSET_EN_BAR		INPUT (Digital)	0.9V	
NO_INT_VREF_EXT	Bias signal given to the MOS when Integrator is bypassed	INPUT (Analog)	0.9V	This is an Analog bias signal
DSCH_EN_BAR	Digital signals enabling the discharge current mirrors	INPUT (Digital)	0.9V	
DSCH_EN		INPUT (Digital)	0.9V	
DSCH_BIAS_TOP	Bias signal coming from the "BIAS_CKT_CHECK" block	INPUT (Analog)	0.9V	Bias voltages for "DSCH" and "OFFSET" current mirrors
DSCH_BIAS_BOTTOM		INPUT (Analog)	0.9V	
BLc	Bit line compliment current from CTT array	INPUT (Analog)	Current	
BLt	Bit line true current from CTT array	INPUT (Analog)	Current	
VSS	Power supply	INPUT (Analog)	0	Ground line
USE_INT_BAR	Digital signals used to bypass the integrator (If it is not working as designed)	INPUT (Digital)	0.9V	0V by default (uses integrator)
USE_INT		INPUT (Digital)	0.9V	0.9V by default (uses integrator)
Tmax_BAR	Integrator enable signals	INPUT (Digital)	0.9V	0V to enable the integrator
Tmax		INPUT (Digital)	0.9V	0.9V to enable the integrator
TOP_CMFB_BIAS	Gate bias voltage for the top CMFB loop (MOS as a resistor)	INPUT (Analog)	0.9V	0.9V by default
RESET_BAR	Switch to reset the integrating cap	INPUT (Digital)	0.9V	Recommended to use it after every weighted sum calculation
RESET		INPUT (Digital)	0.9V	
CM_VGS_OTA	Integrator's cuurent mirror bias voltage	INPUT (Analog)	0.9V	generated by "BIAS_CKT_CHECK" block
OTA_VREF_200mV	Vref signal for the integrator	INPUT (Analog)	0.9V	200mV by default
CM_VGS_BOTTOM_CS	Bias voltage for current mirrors at the bottom (attached to the gate of the input diff pair of the integrator)	INPUT (Analog)	0.9V	generated by "BIAS_CKT_CHECK" block
CASCODE_VREF_200mV	Gate reference voltage for the top cascode current mirror	INPUT (Analog)	0.9V	200mV by default

Table 1: Pin description

There are two modes of operation for the integrator. No matter how good any layout is done for the mixed signal circuit, there will be offsets. One of the common methods to handle this is to cancel the offset or calibrate it during inference mode of operation.

Hence, there are two power up sequence for the integrator. Figure 21 and 22 summarizes the power up sequence for this integrator. Any missing signals from these figures indicates that the signal is either grounded or is applied external to the chip.

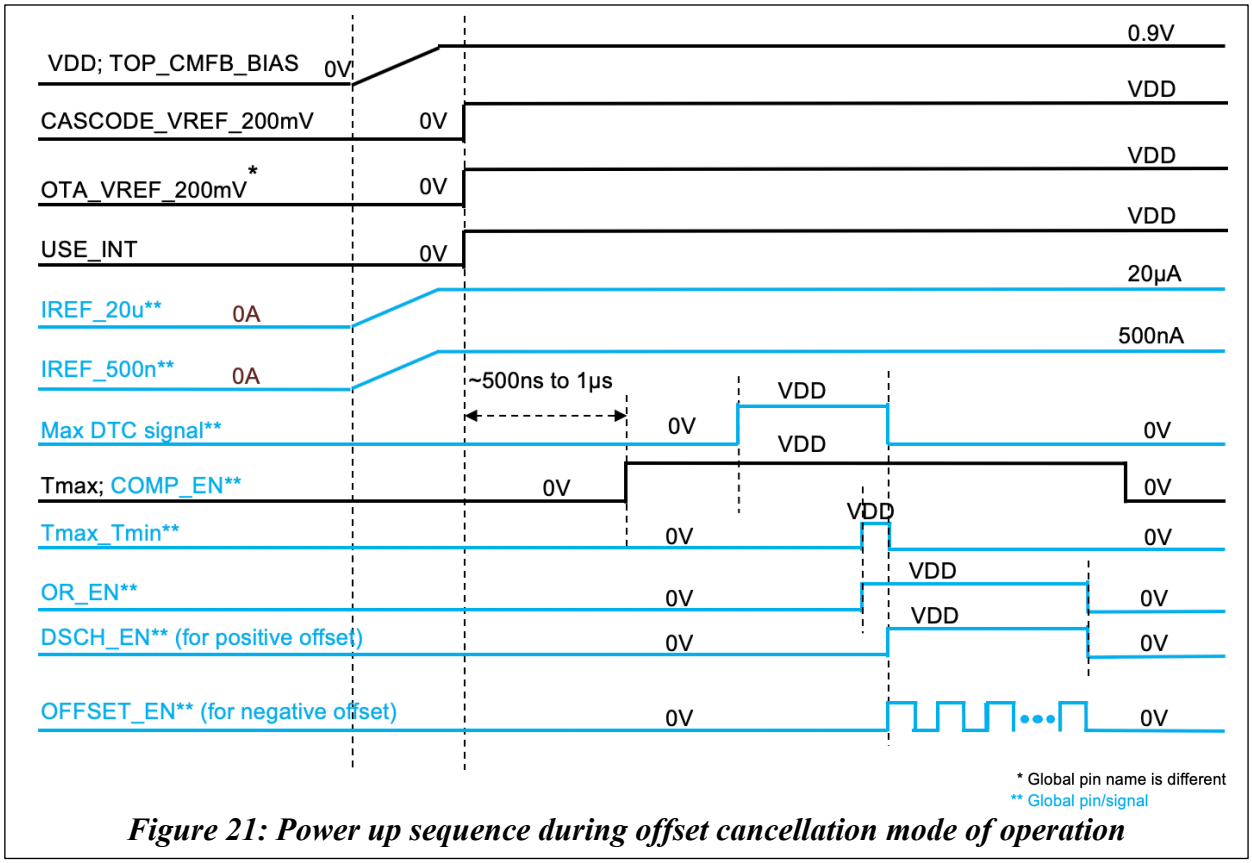


Figure 21: Power up sequence during offset cancellation mode of operation

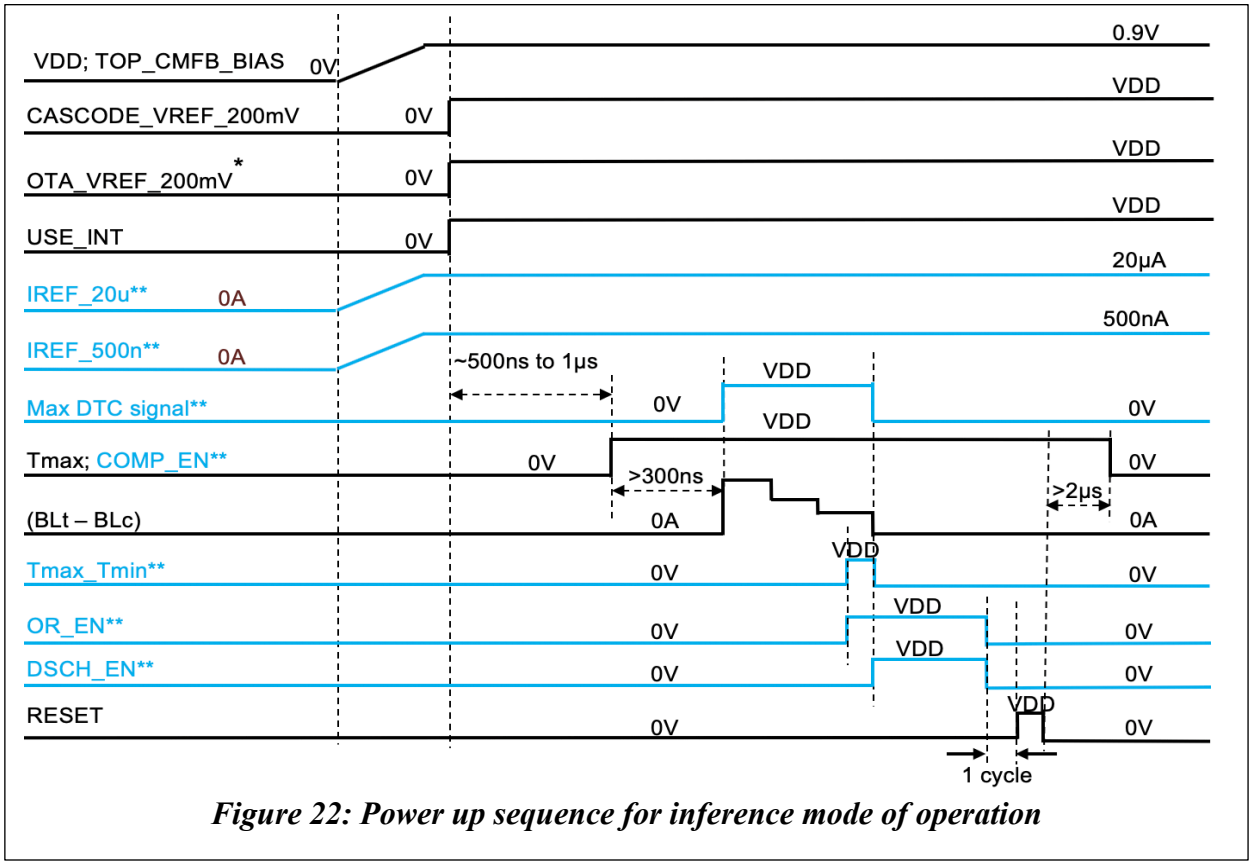


Figure 22: Power up sequence for inference mode of operation

IV. Simulation results of integrator:

In the previous section, we saw all the important equations and specifications governing the integrator. Now let's look at the simulation results for the same.

All simulations were carried out across process voltage and temperature (PVT) corners. Table 2 describes PVT simulation conditions.

PVT PARAMETERS						
SL NO	**PARAMETER**	**MIN**	**Typ**	**MAX**	**UNIT**	**COMMENTS**
1	TEMP	-40	27	85	°C	Tested for -40°C to 125°C
2	VDD	850	900	910	mV	Tested for 810mV to 990mV
3	CAP*	400	500	550	fF	Tested for 250fF to 600fF
4	CORNERS		TT; SS; FF; SF; FS			Passive component variations are included in these corners

Table 2: PVT Corners ** Cap refers to the parasitic cap at the drain node of CTT due to routing*

To characterize the integrator, three major type of analysis was carried out with PVT corners.

- STB Analysis (frequency analysis)
  - DC gain, Phase, Bandwidth (BW), Gain margin
- Noise Analysis
  - Noise current injection across PVT
- Transient Analysis
  - Waveforms of important nodes, linearity analysis

Frequency Response:

In the previous section, we saw the equation governing poles and zeros in this system. Figure 23 and 24 shows the test bench required for the frequency response of this integrator.

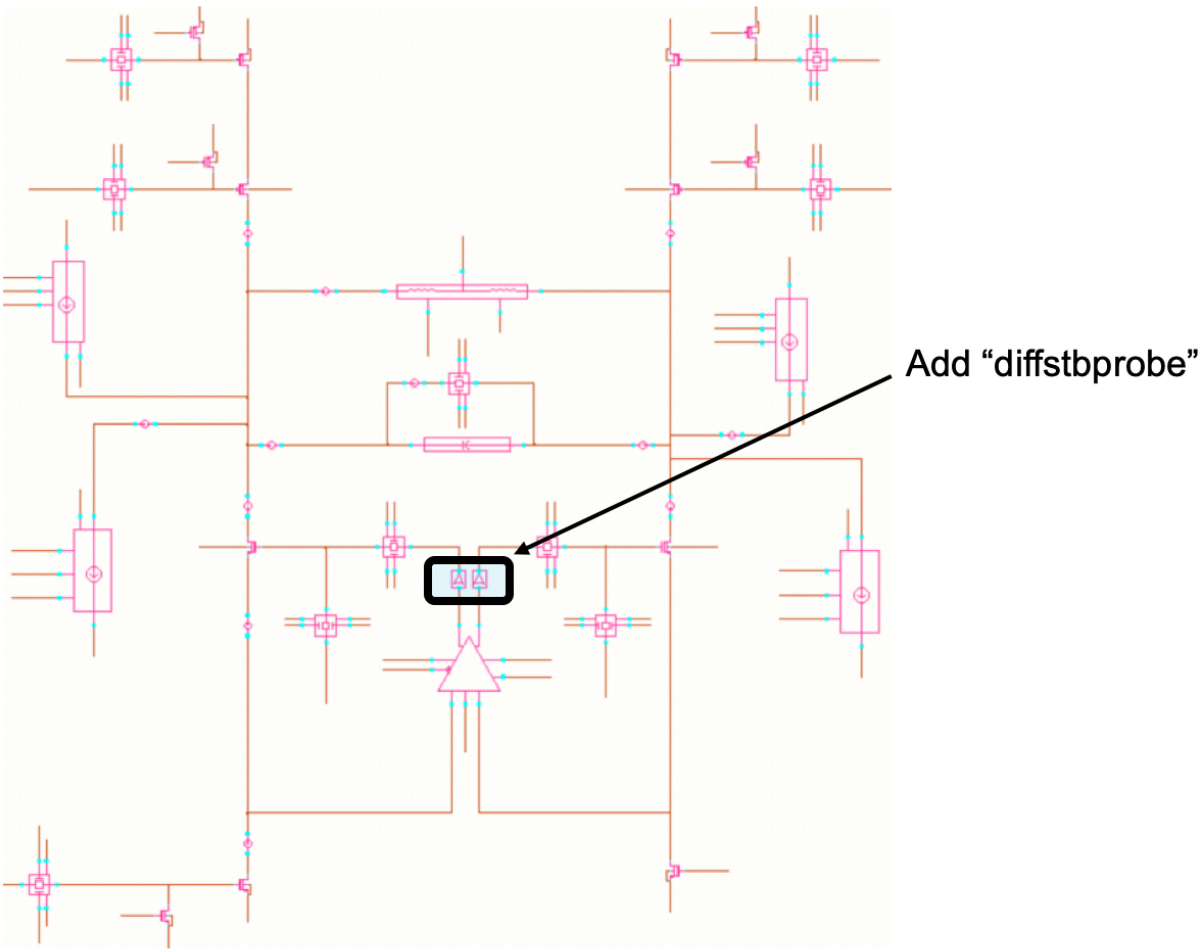


Figure 23: Addition of ‘diffstbprobe’ to extract frequency response

Figure 24 shows the testbench for AC simulation. For now, let’s assume comparator is just comparing when the integrating capacitor voltage reaches 0V across it.

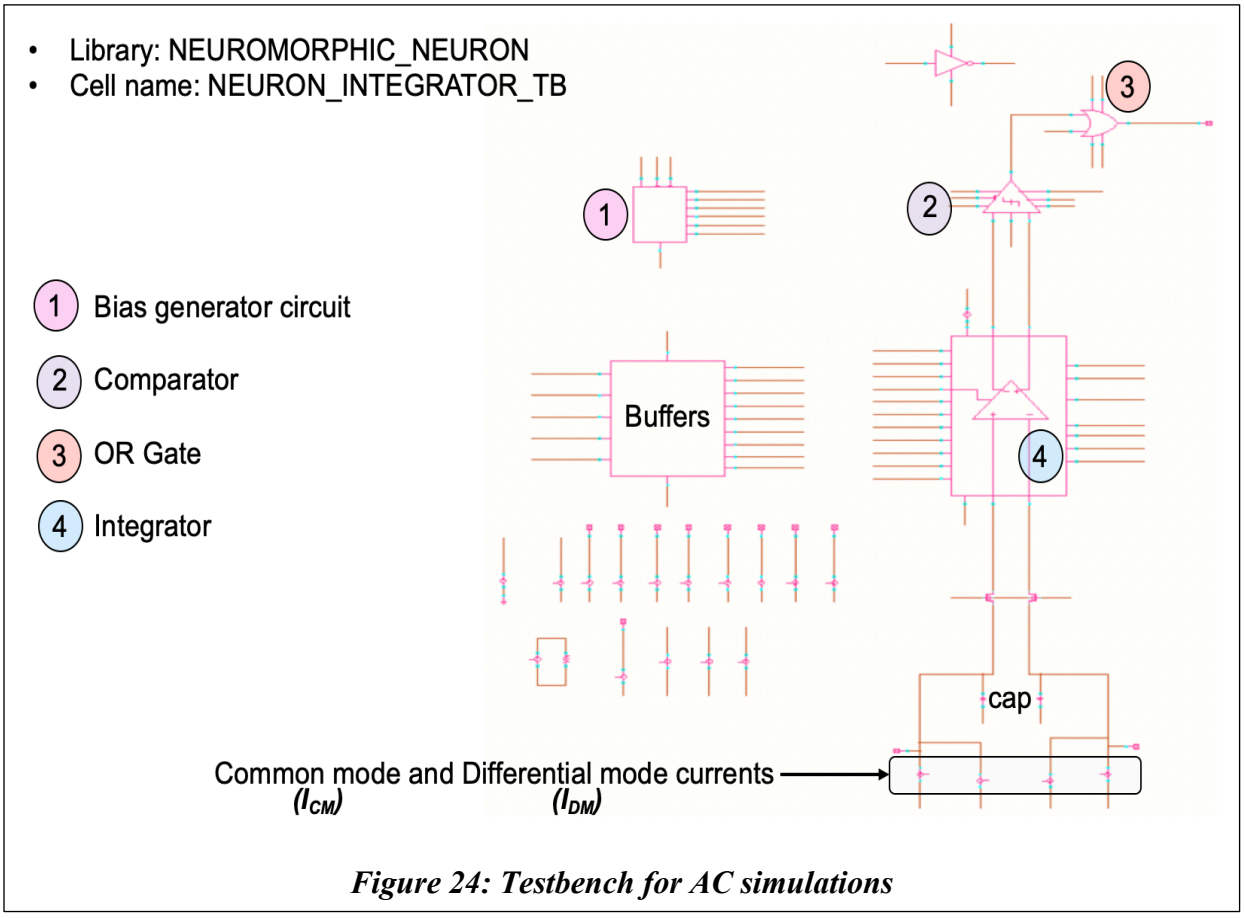
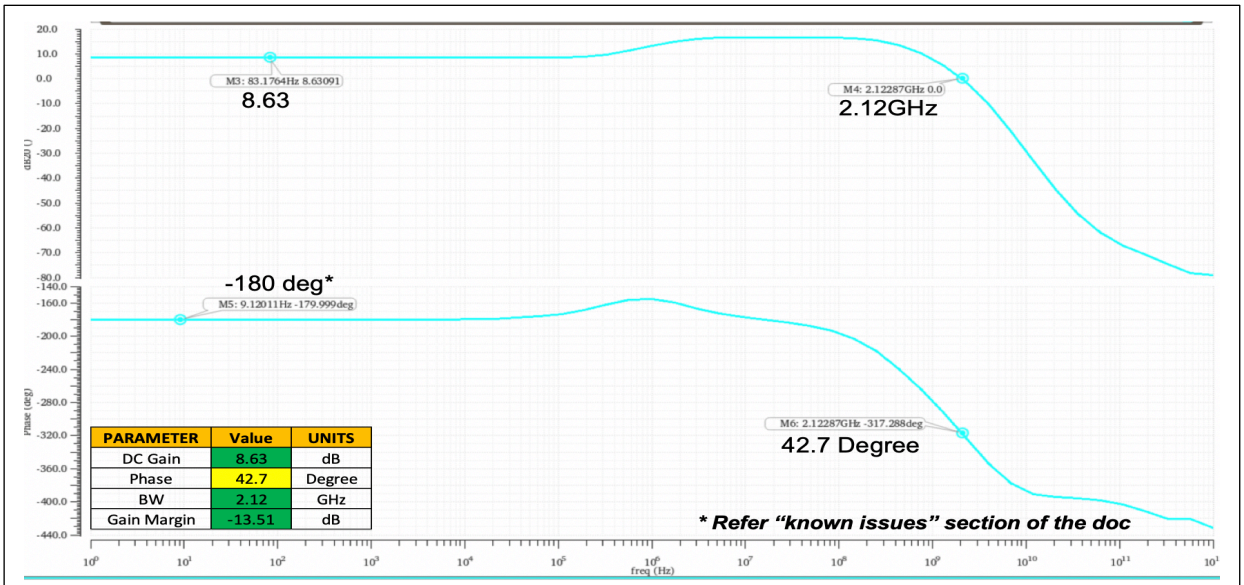


Figure 24: Testbench for AC simulations

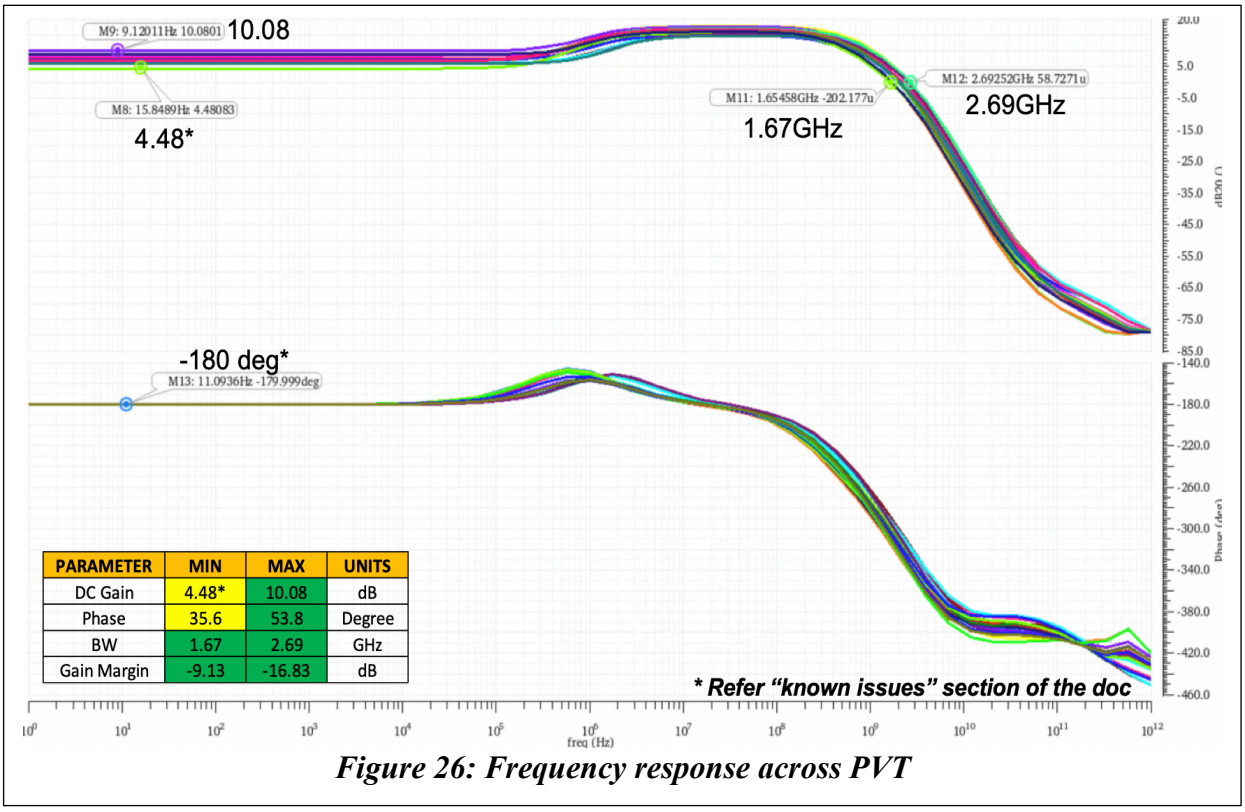
Figure 25 shows the frequency response in typical corner (VDD = 900mV, temp = 27-degree, Process corner = TT, Cap = 500fF)



PARAMETER	Value	UNITS
DC Gain	8.63	dB
Phase	42.7	Degree
BW	2.12	GHz
Gain Margin	-13.51	dB

Figure 25: Bode plots in typical corner

Figure 26 shows the frequency response of the integrator across PVT corner (Table 2).



PARAMETER	MIN	MAX	UNITS
DC Gain	4.48*	10.08	dB
Phase	35.6	53.8	Degree
BW	1.67	2.69	GHz
Gain Margin	-9.13	-16.83	dB

Figure 26: Frequency response across PVT

Transient analysis:

Now, let's look in detail about the top common mode feedback loop (CMFB). As shown in figure 27, the DC common mode voltage has to be established so that all the biasing is taken care. For this, we have added a resistor and capacitor in parallel as shown in figure 27. Thus, the CMFB resistors define the V_{gs} voltage across the top PMOS current source (shown in red color).

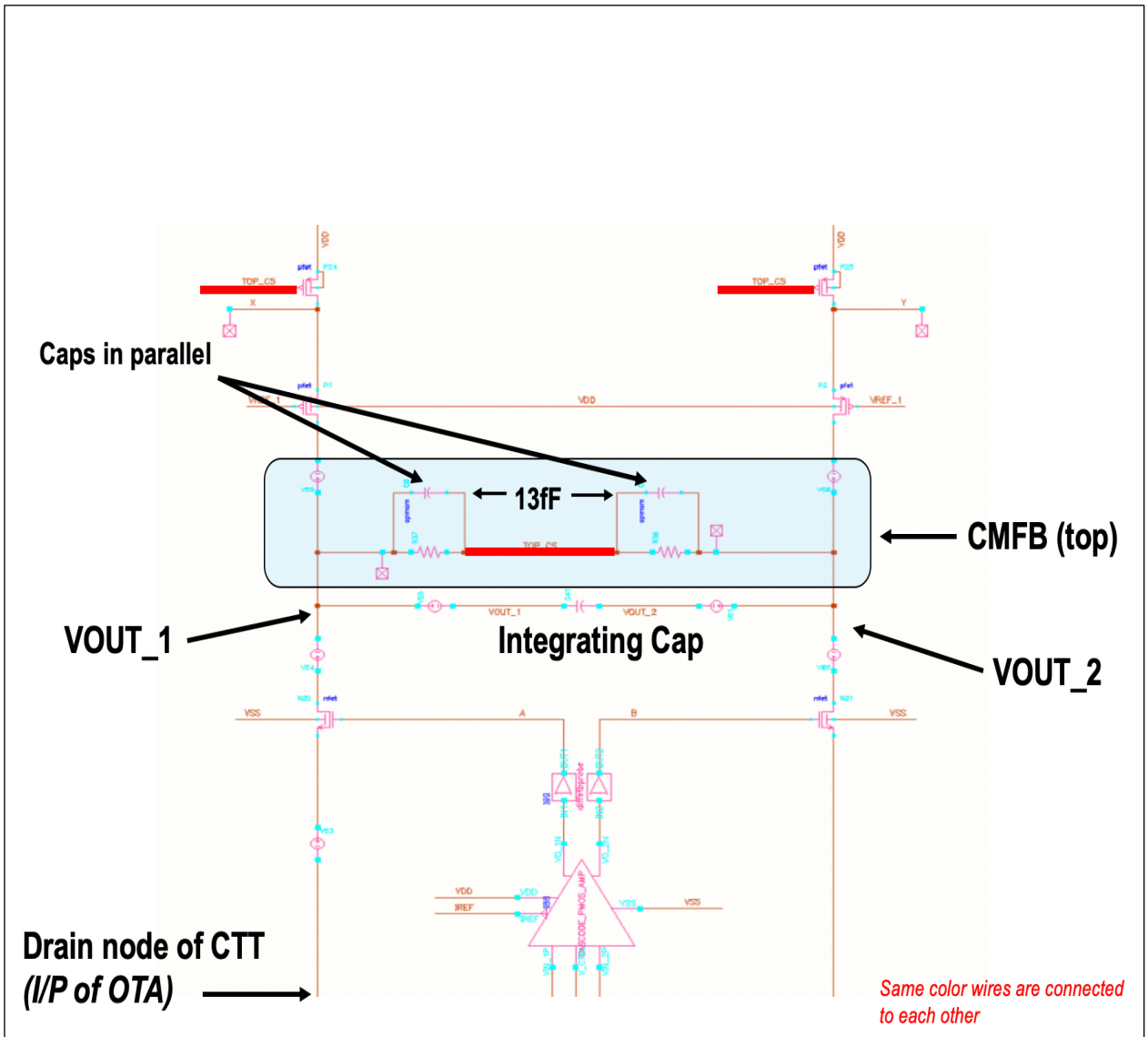


Figure 27: Schematic for top CMFB loop

Figure 28 and 29 shows the transient response of the integrator. Voltage across the capacitor is plotted as 'VOUT_1-VOUT_2' along with the drain node of CTT (inputs). It can be noted that the loop gain g_{m2} (figure 14) is sufficient enough and is regulated the drain node of CTT by not varying it more than a 2mV!

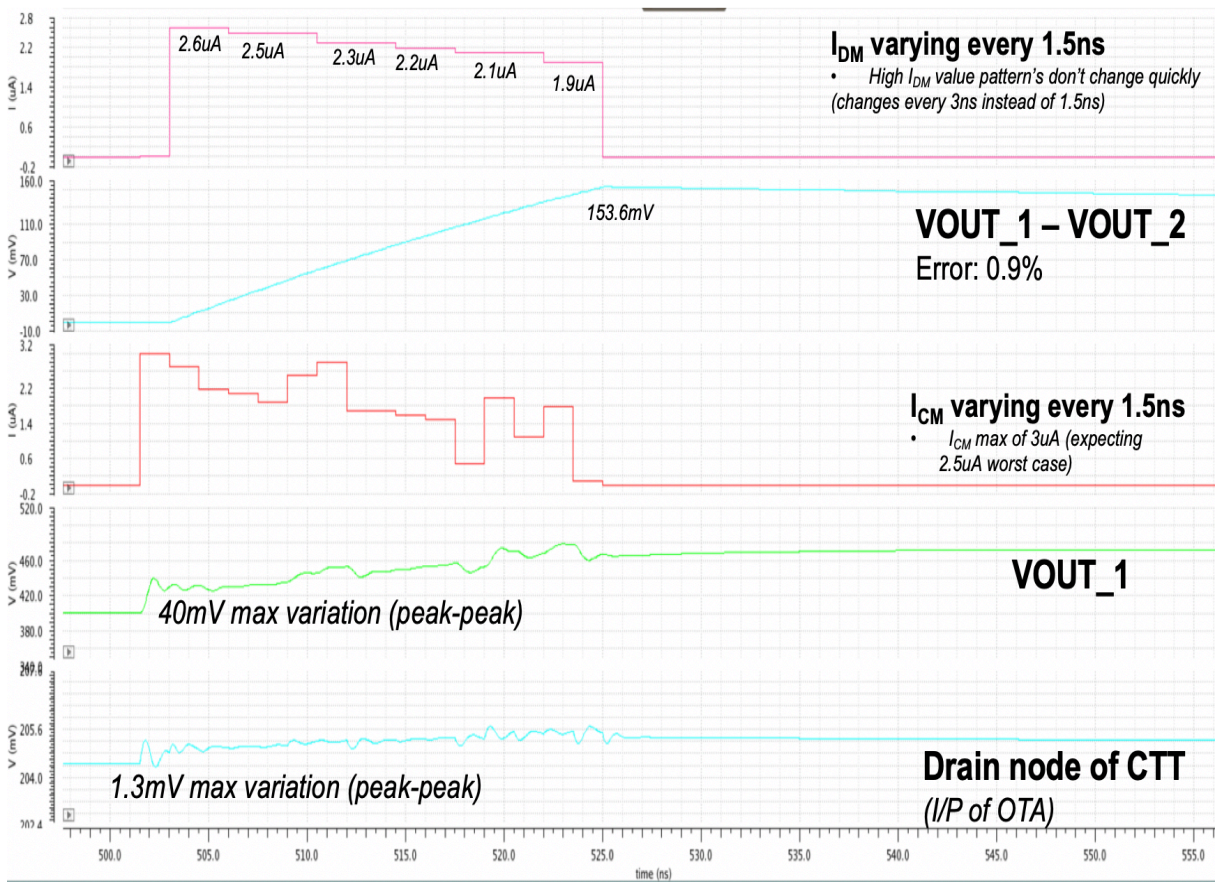


Figure 28: Transient response for one of the realistic, slow varying input patterns (both I_{DM} and I_{CM})

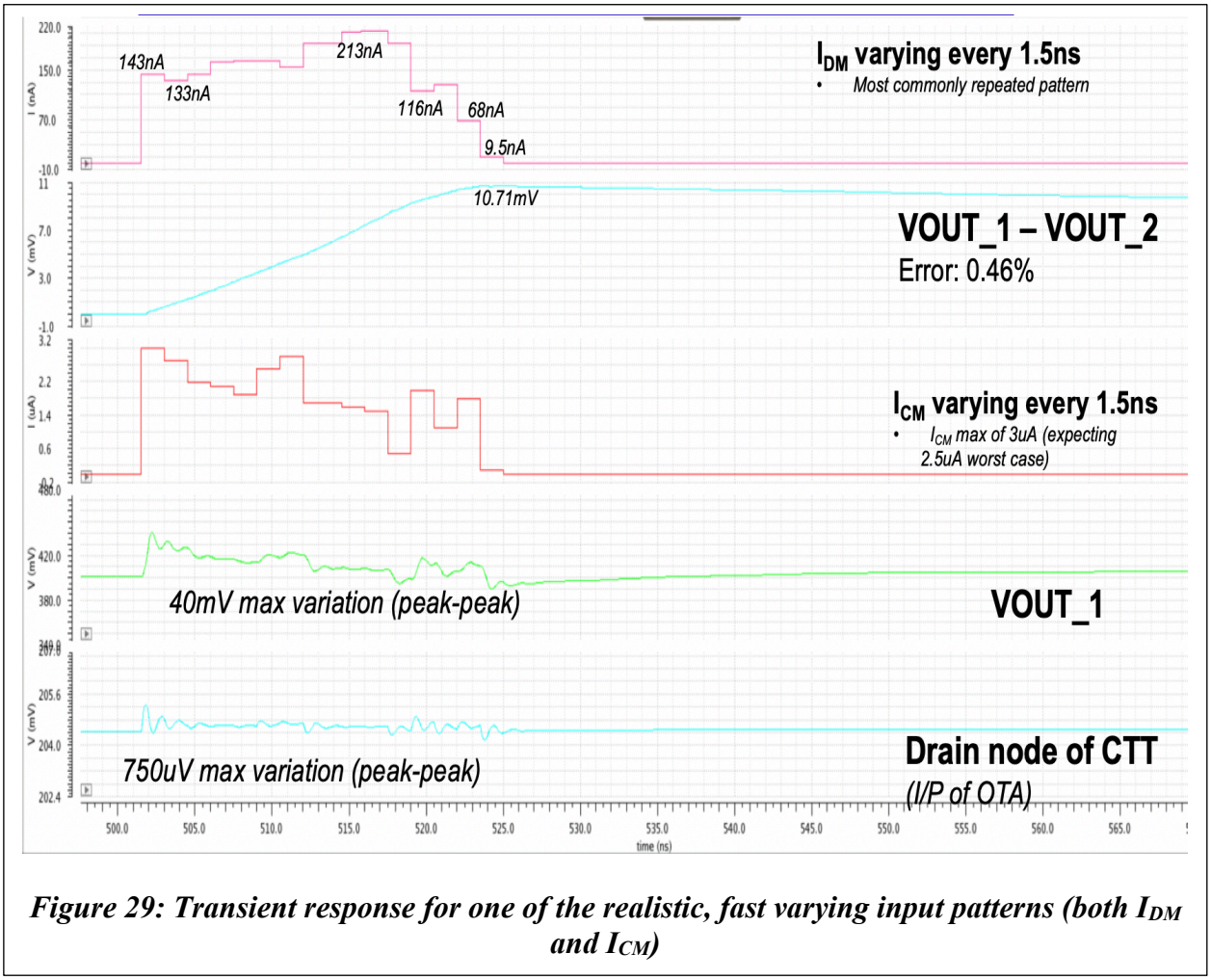


Figure 29: Transient response for one of the realistic, fast varying input patterns (both I_{DM} and I_{CM})

Following figures shows the transient response of the integrator for some random but realistic input patterns.

- ***Case 1:*** This current waveform is the max current for the MNIST data set (obtained from MATLAB simulations)

- ***Case 2:*** This current waveform is the typical current waveform for the MNIST data set (obtained from MATLAB simulations)

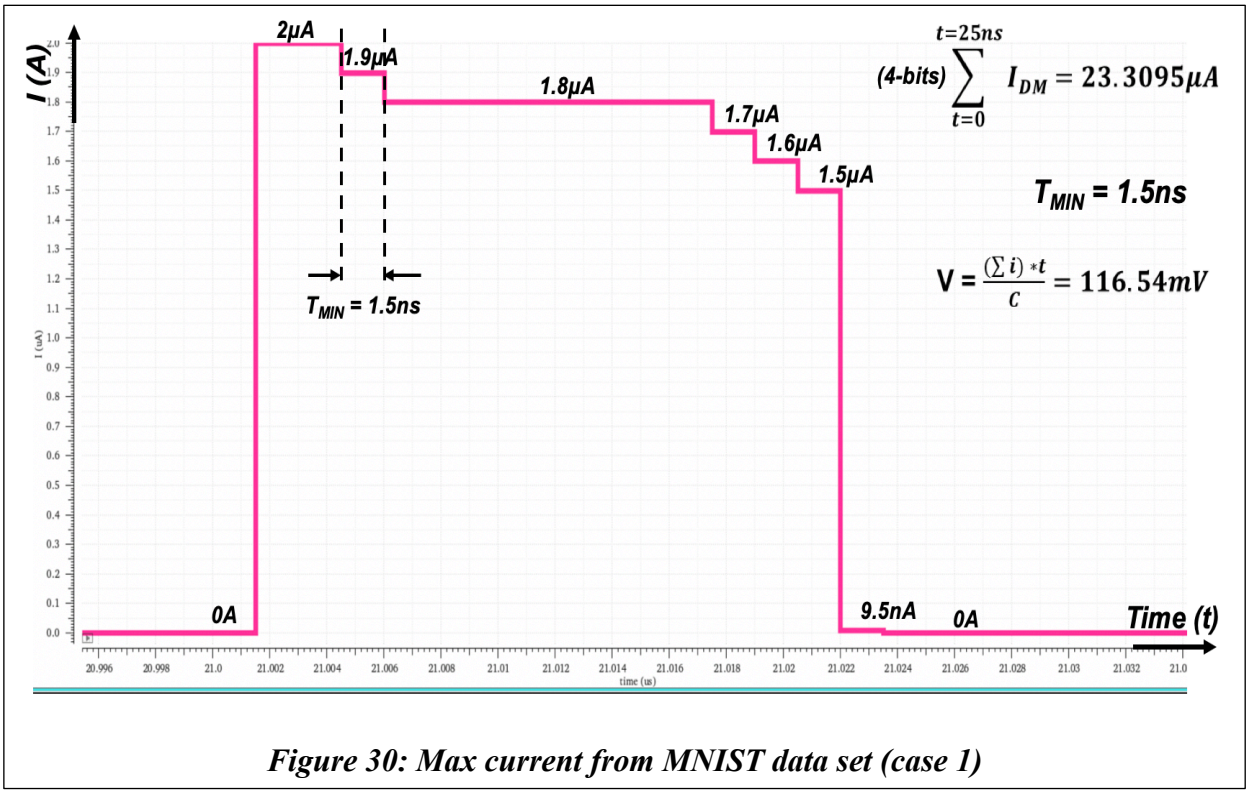


Figure 30: Max current from MNIST data set (case 1)

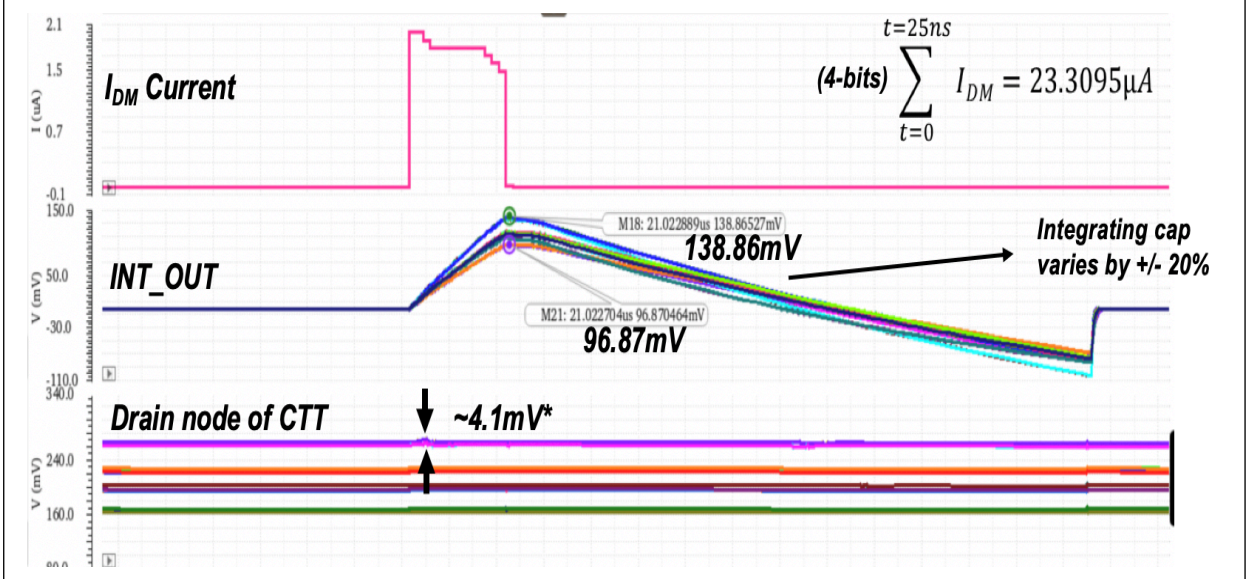


Figure 31: Transient response for the input pattern shown in figure 30

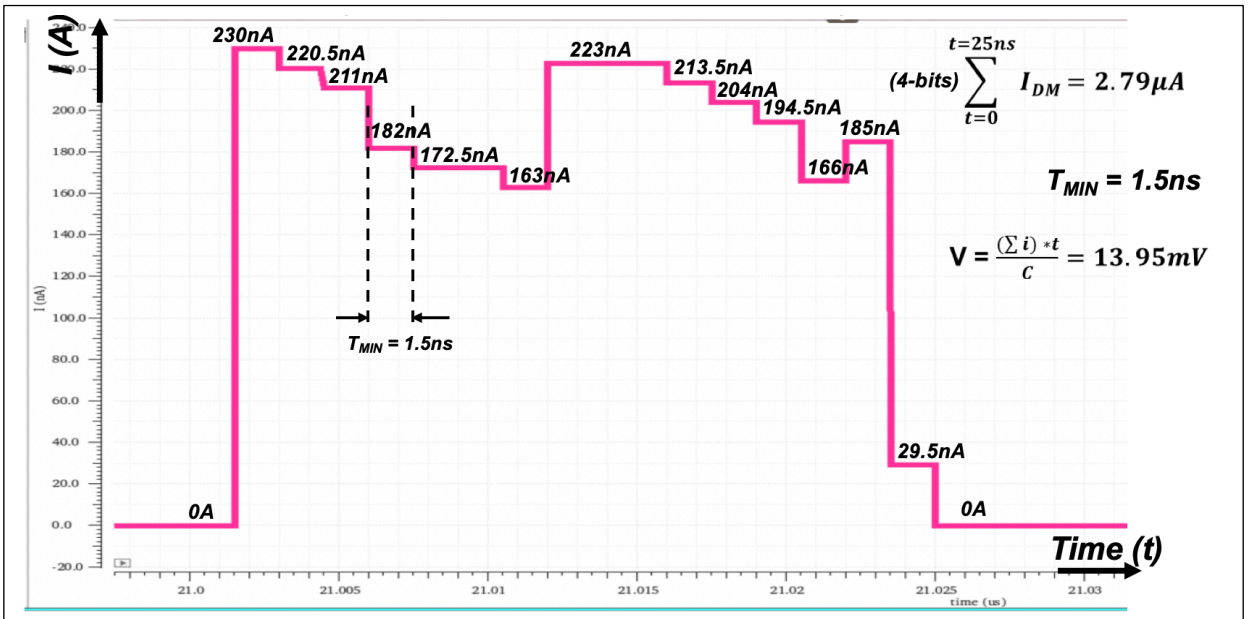


Figure 31: Typical current from MNIST data set (case 2)

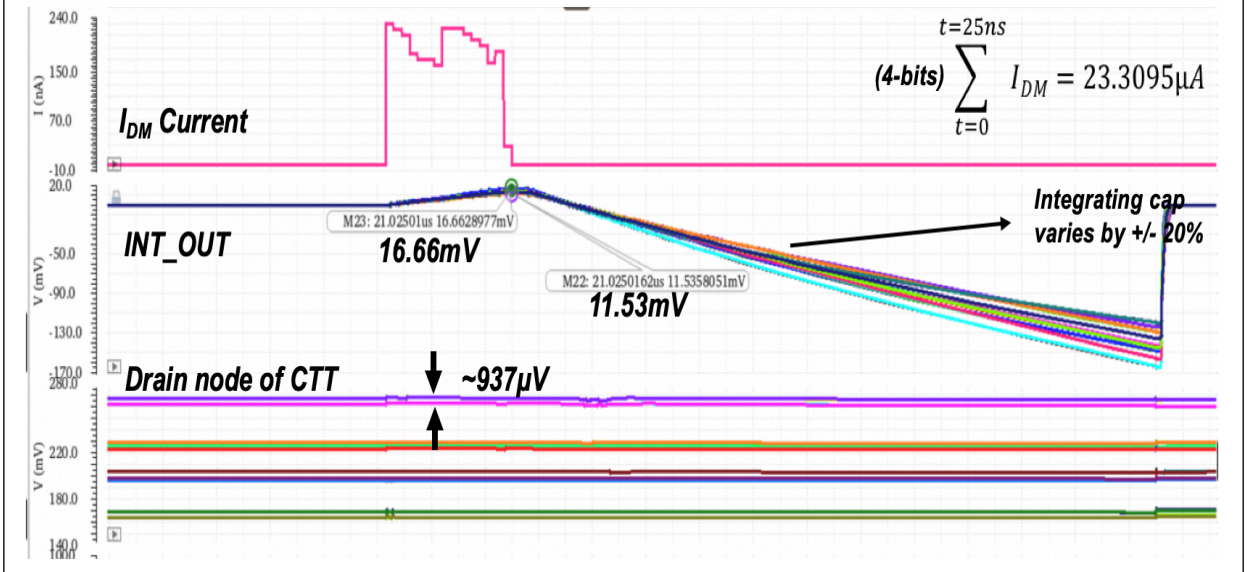


Figure 32: Transient response for the input pattern shown in figure 31

Linearity analysis:

In order to determine the smallest input pulse width (highest frequency) that the integrator can handle, (which in turn determines the resolution of the integrator), we perform linearity analysis. Figure 24 shows the testbench required for the same and figure 33 and 34 shows the linearity plots.

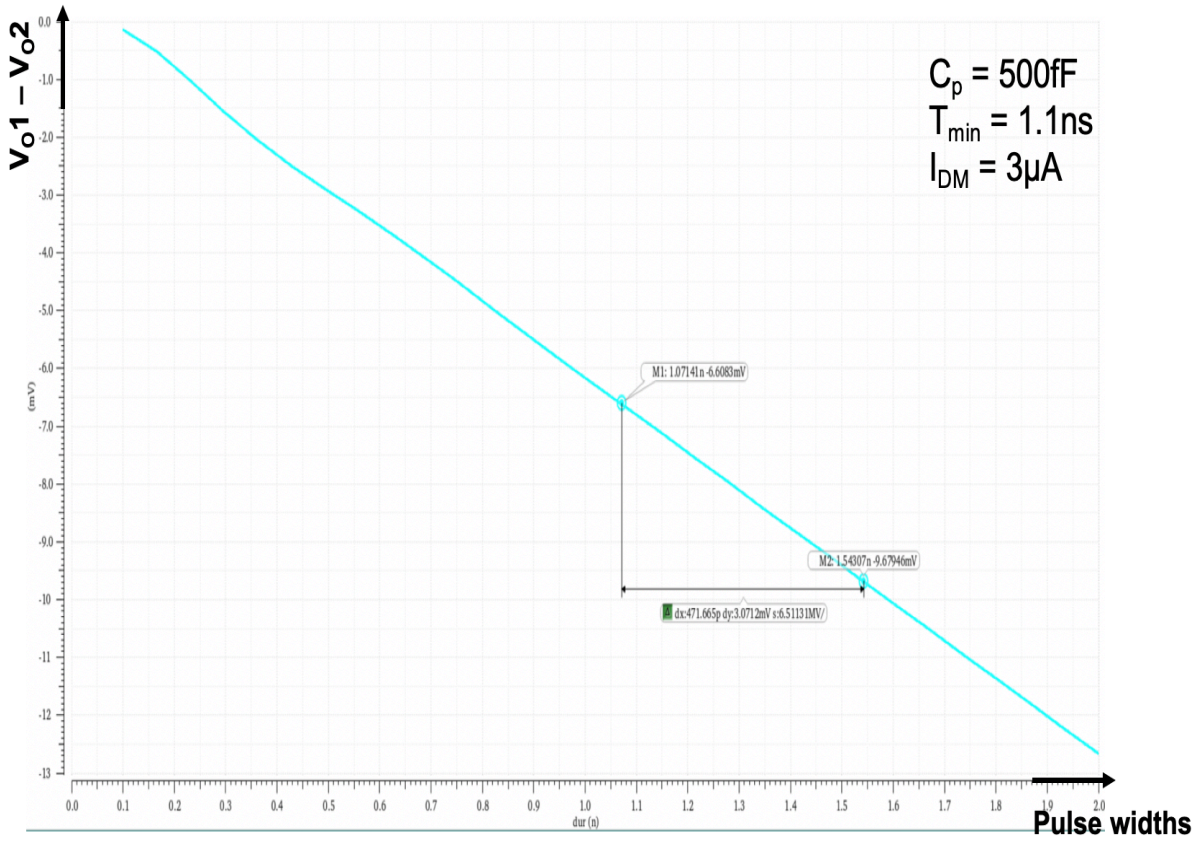


Figure 33: Linearity plot in typical corner

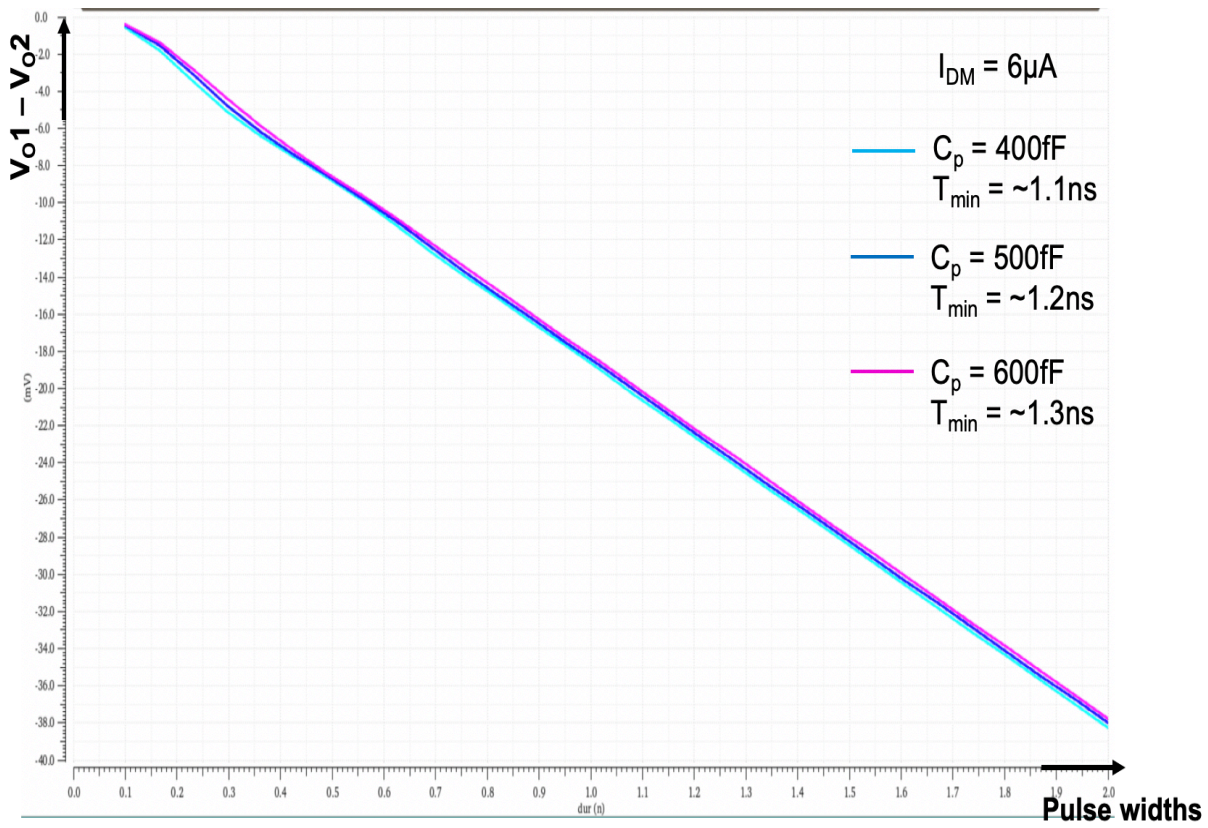


Figure 34: Linearity plots across PVT corners

In the next section, we will look at design of comparator, why do we need it? And so on…

V. Design of comparator:

Once the integrator computes the weighted sum from all the currents, there has to be a block/unit which decides whether the neuron should give any output or not. If there is any output that the neuron has to produce, how long a pulse should it be? All this is done by a block called *activation function.*

What if the activation function is linear? If the activation function is linear, then irrespective of whether the weighted sum is positive or negative, it is just scaled and sent out as an output. Hence, we need an activation function that determines whether to output or not. Along with that, the activation function has another functionality that has direct impact with training time of the weights.

Some of the famous activation functions are sigmoid function, Tan hyperbolic function, Rectifying Linear Unit (ReLU), leaky ReLU and so on. Among all the above mentioned, ReLU is the most popular among the community due to its ability to train the weights in a quicker manner.

ReLU transfer function looks as shown in figure 35. This can be implemented by a comparator since we have pulse width modulation technique. In figure 35, x-axis represents the voltage on the integrating capacitor (weighted sum) and the y-axis represents the output pulse width.

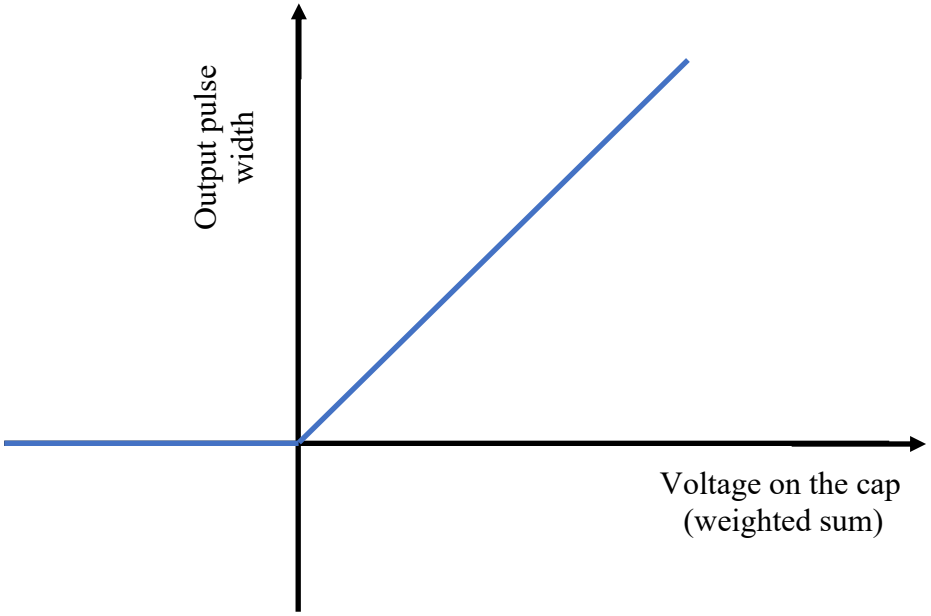


Figure 35: ReLU transfer function

Now let's look at important specifications for the design of our comparator. An ideal comparator trips when the difference voltage across its two terminal reaches exact 0V [11]. Along with that, it trips the output of the comparator exactly when the difference voltage reaches 0V. That is, in other words, there is no time delay between the difference voltage reaching 0V and the output of the comparator reaching 50% of its VDD. Figure 36 shows the ideal characteristics of the comparator.

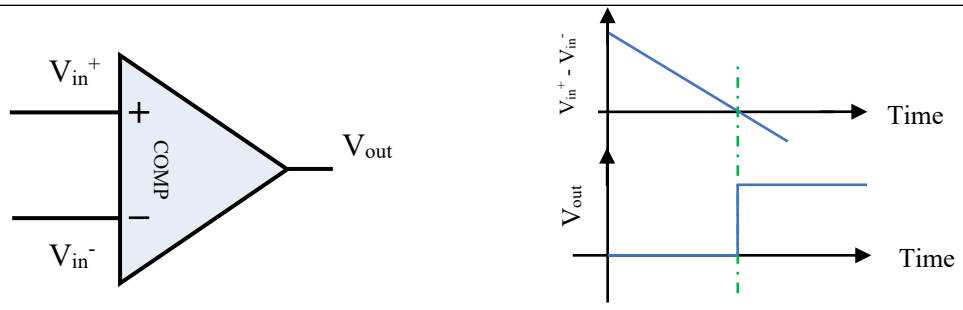


Figure 36: Ideal characteristics of comparator

The technique in getting closer to the ideal characteristics is to amplify the difference in voltage between two terminals as high as possible and tripping the comparator. In order to amplify the difference, we need high gain stage which is expensive in terms of power consumption and the area [12]. Hence, depending upon the application, this trade off varies. Also, as the amplifier stages grow, delay becomes worse. But fortunately, delay doesn't matter for our application.

As there are 1024 inputs, we need not apply them exactly at the same time since all columns are fairly independent. Having said that, we try to apply all the inputs in a time frame so that we can shut down the neuron after computing in order to save power (duty cycle it). Since output of the neuron goes as an input to the next layer (which we do not have in this version of the chip but hope to design a generic neuron for future use), we try to restrict the delay of the comparator. One more interesting point to note about the delay of the comparator is that if the comparator has a delay, then we expect all ten neurons (1024 x 10 is our network size) to have the same delay.

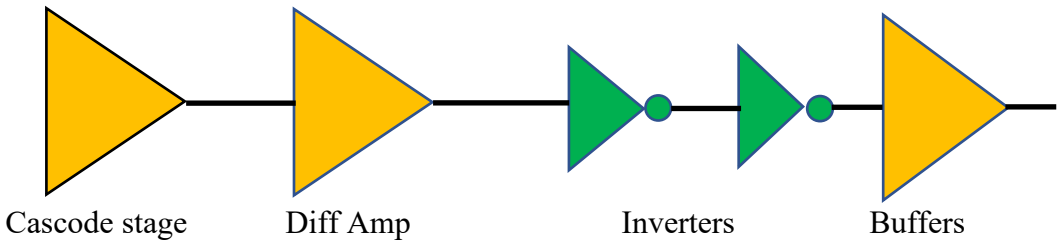


Figure 37: Comparator block diagram

Figure 37 shows the block diagram of our comparator. We have two gain stage followed by a decision-making block. The first stage of the amplifier is a cascode stage followed by simple differential amplifier as shown in figure 38. Both stages combined gives a gain of ~2000 and thus we are able to achieve a resolution [13] of just 113µV (comparator trips when the difference between two terminals is 113µV in either direction).

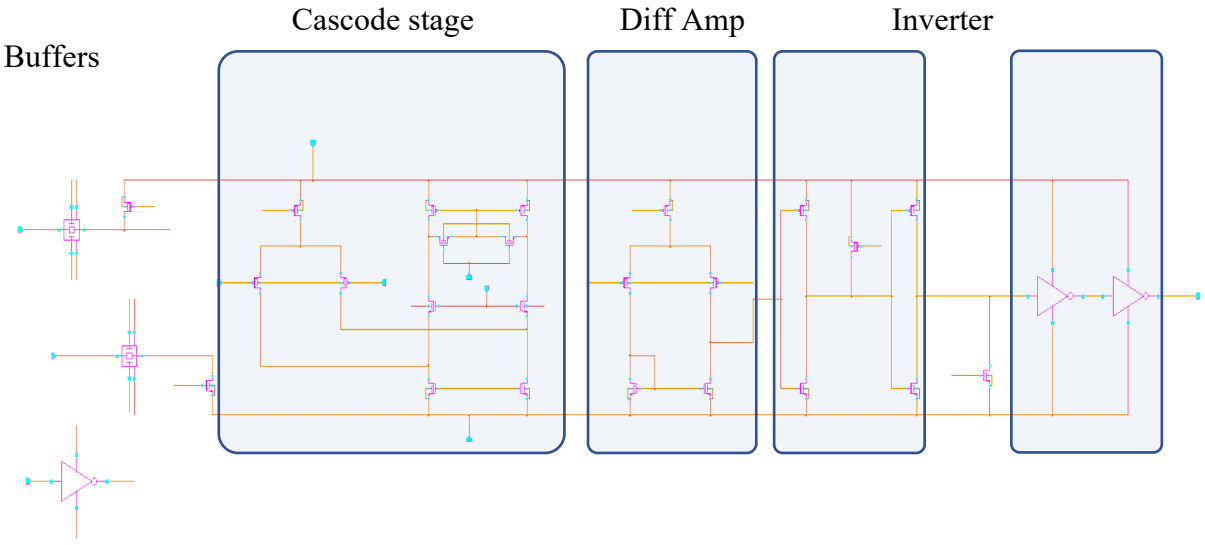


Figure 38: Comparator schematic

Comparator results are shown in the next section.

VI. Neuron simulation results:

From the previous sections, it is clear that a neuron consists of an integrator and a comparator (as a ReLU unit). In our chip, we are using IO IP which has output frequency limitation. Since the output pulse could be as small as 1ns pulse, it is difficult to send out this high frequency signal using the IO IP. Hence, we add a known fixed pulse to the output pulse generated by the comparator using an OR gate as shown in figure 39.

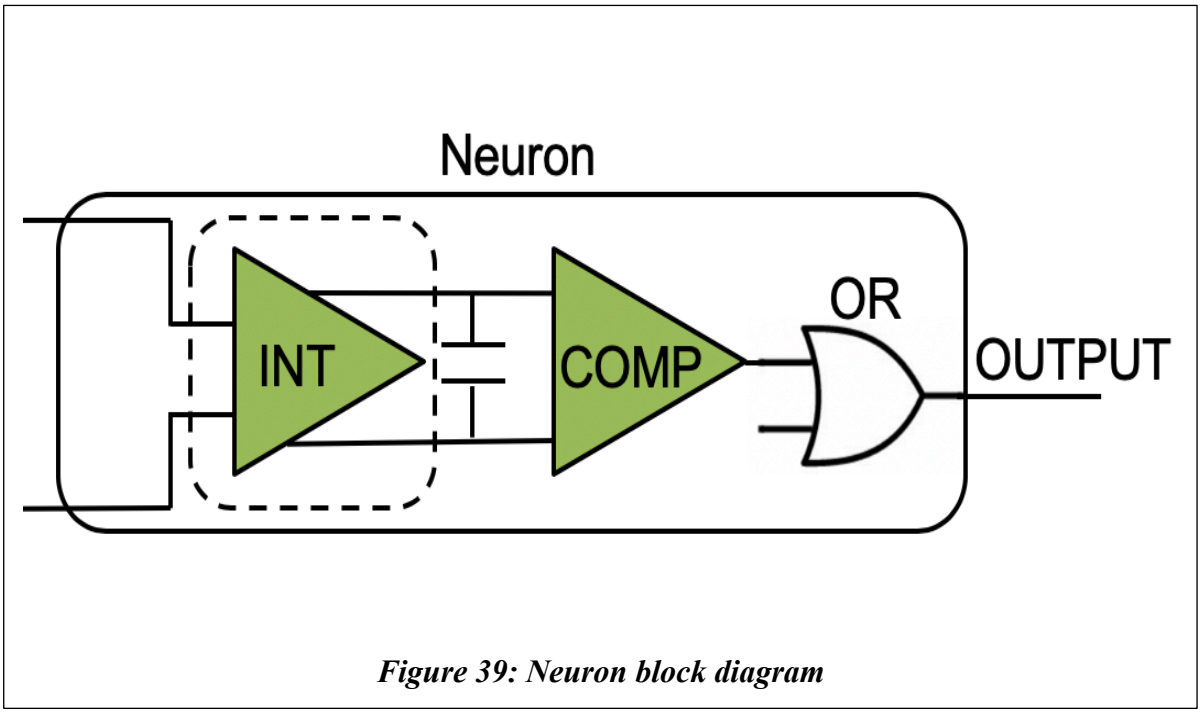


Figure 39: Neuron block diagram

One input of the OR gate shown in figure 39 is the output pulse generated by the comparator, and the other input is the known pulse width generated by the digital core to take care of the frequency limitation of the IO IP.

Following figures from 40 – 42 shows some simulation results of the neuron.

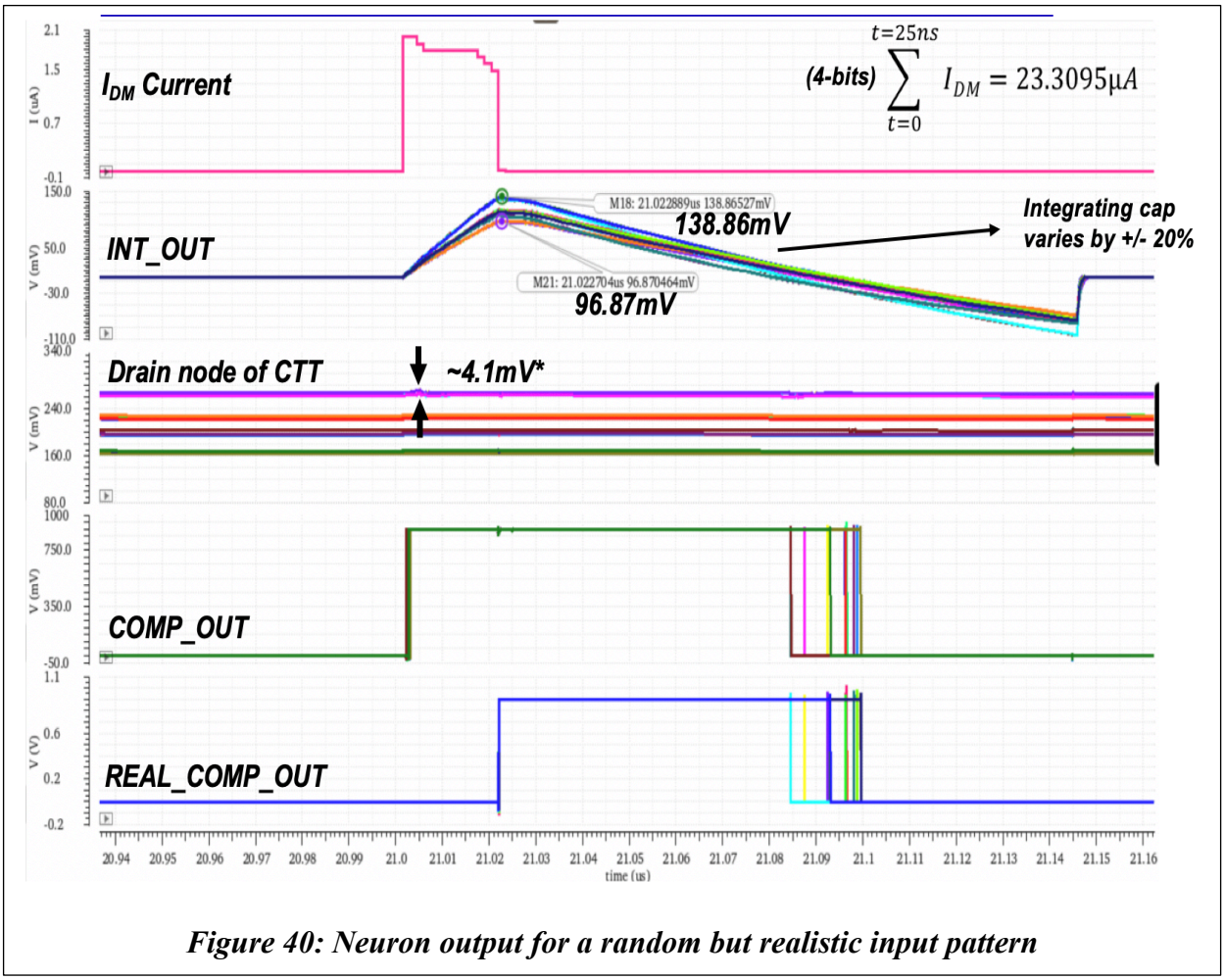


Figure 40: Neuron output for a random but realistic input pattern

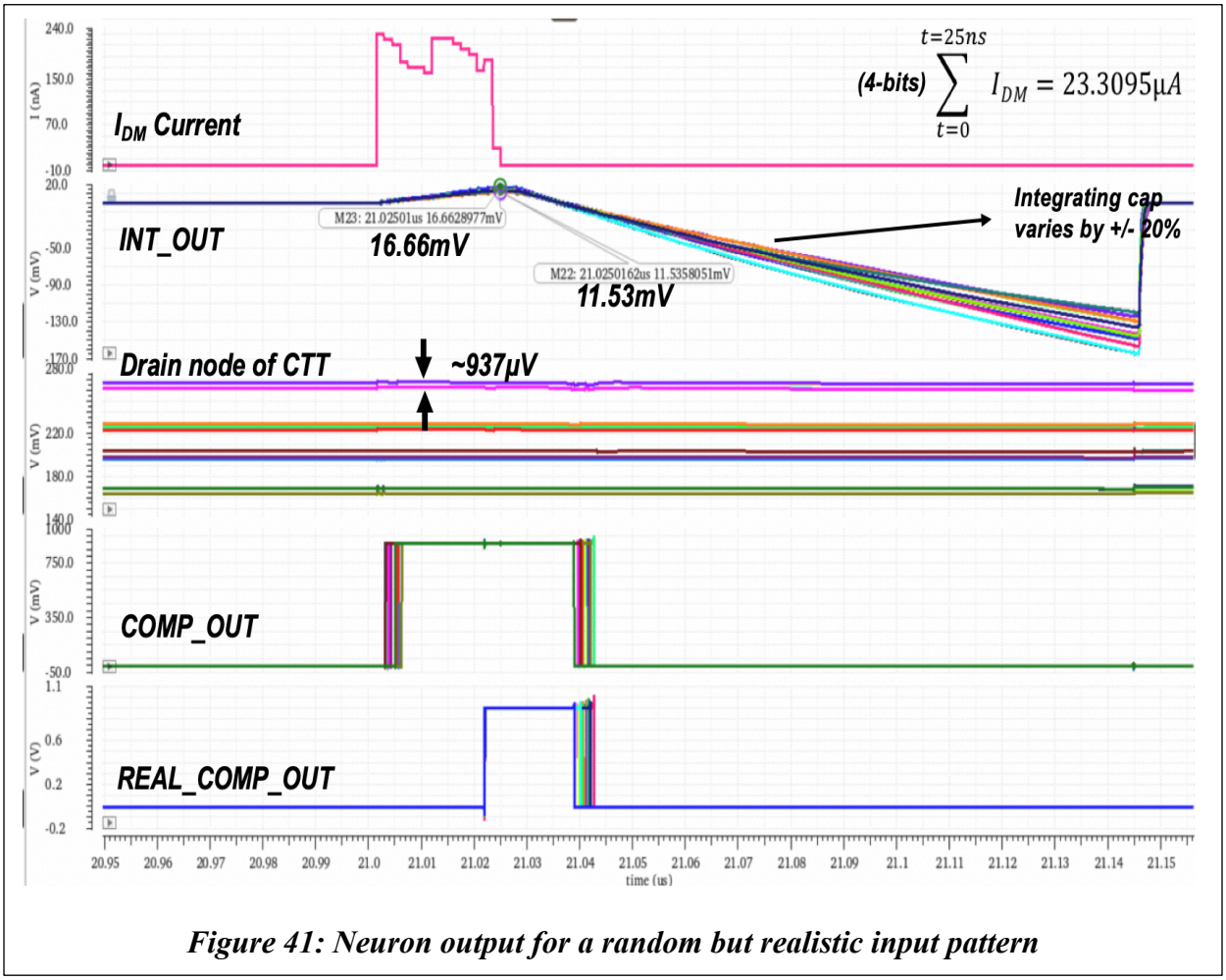


Figure 41: Neuron output for a random but realistic input pattern

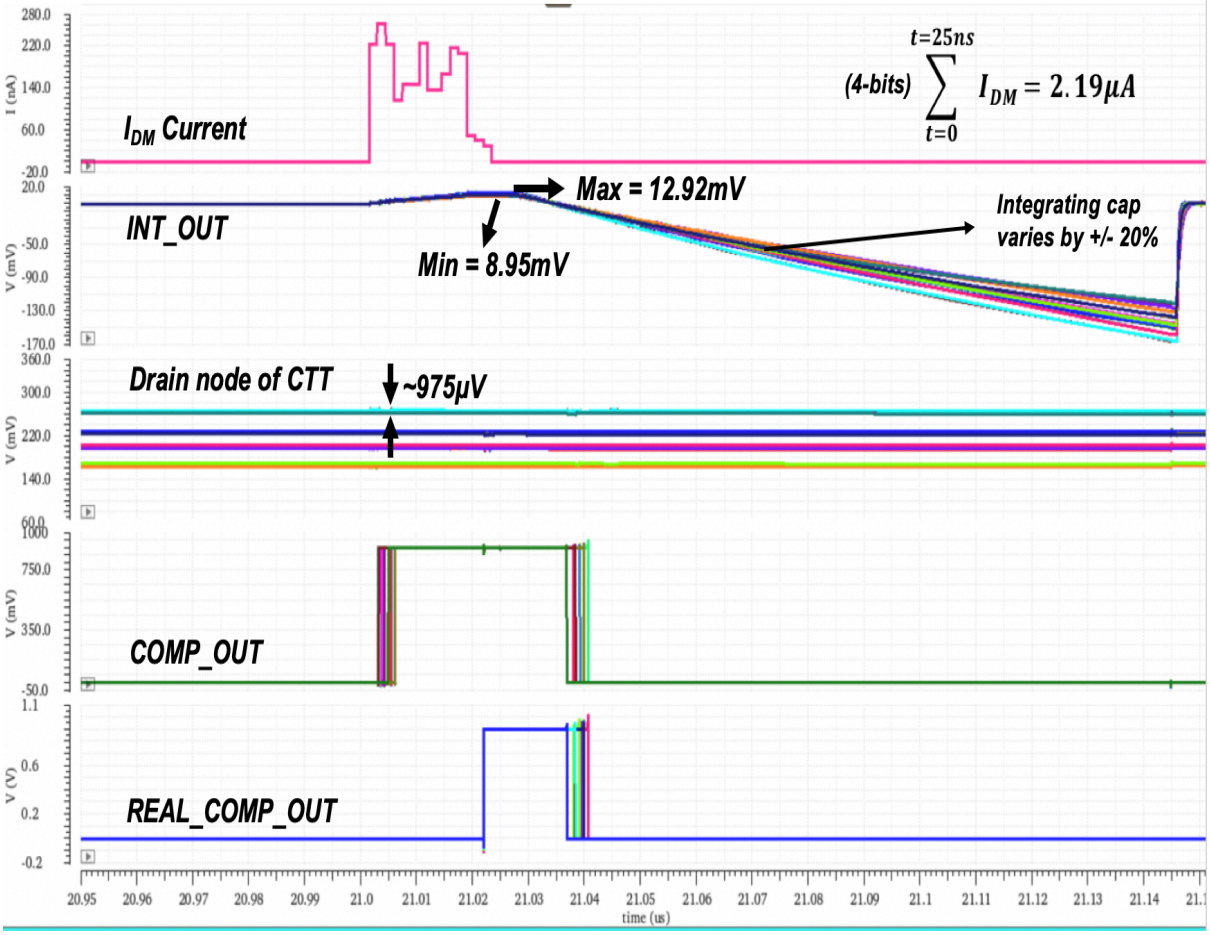


Figure 42: Neuron output for a random but realistic input pattern

VII. Layout of the neuron:

Each neuron occupies ~320µm x 3µm (X and Y direction respectively). Y-direction limitation is due to the height of CTT memory cell (neuron is pitch limited due to the height of CTT memory cell). Figures 43 to 45 shows snapshots of neuron layout done in 22nm FDSOI technology. It can be noted that the integrating capacitor is not the biggest part of the neuron layout.

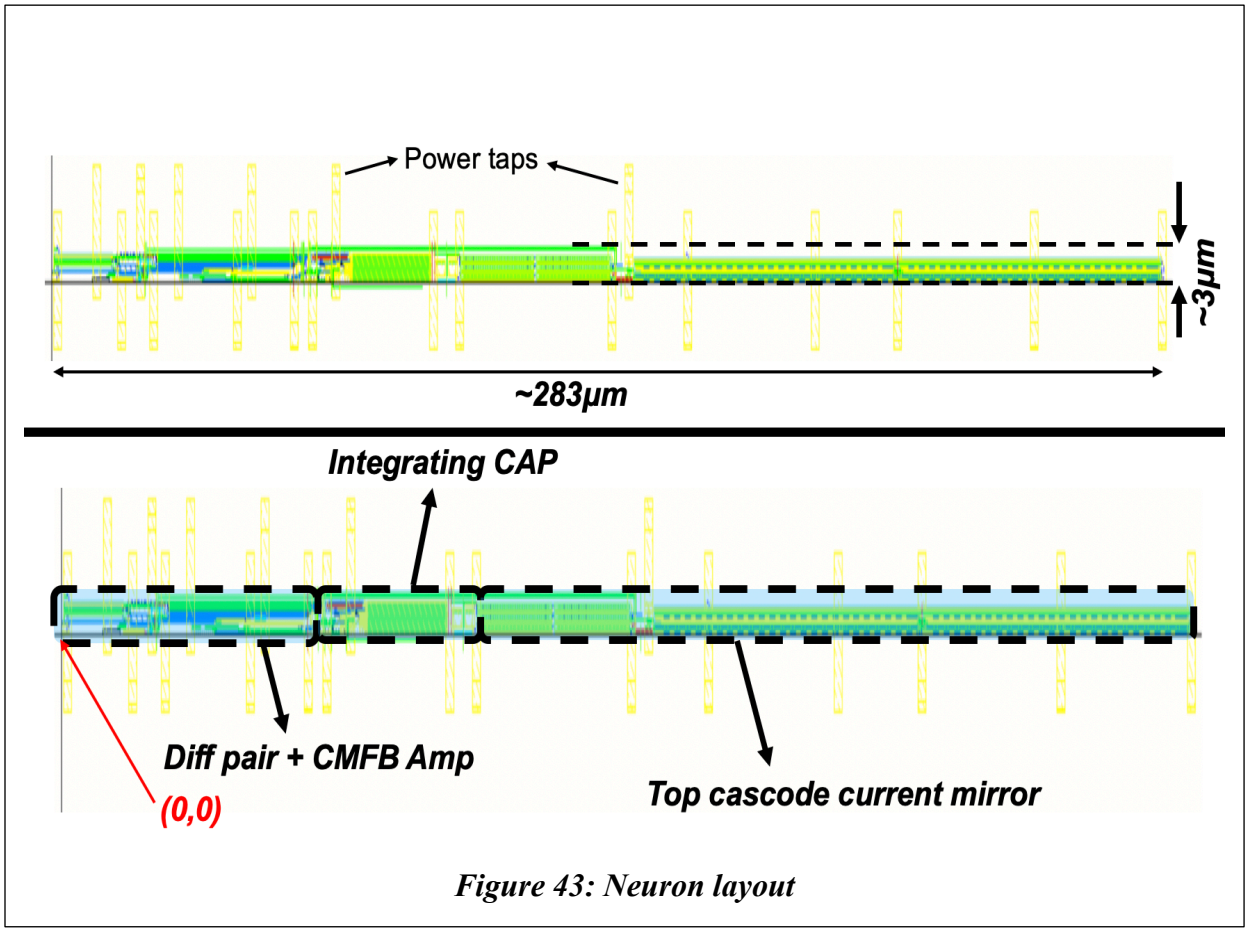


Figure 43: Neuron layout

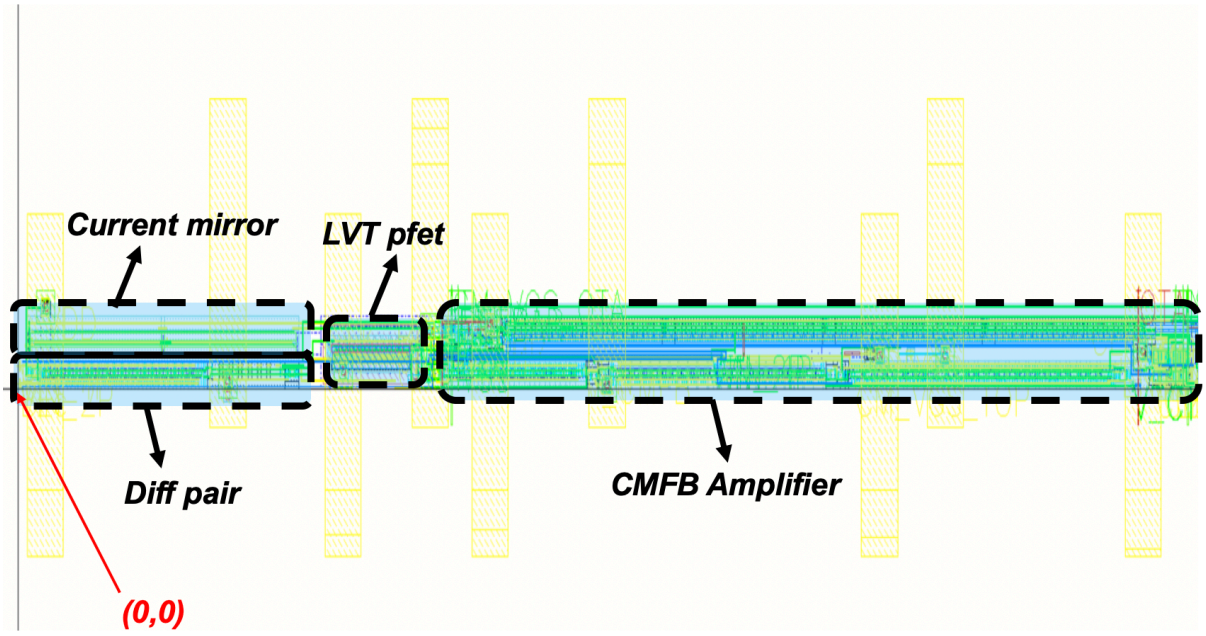


Figure 44: OTA layout

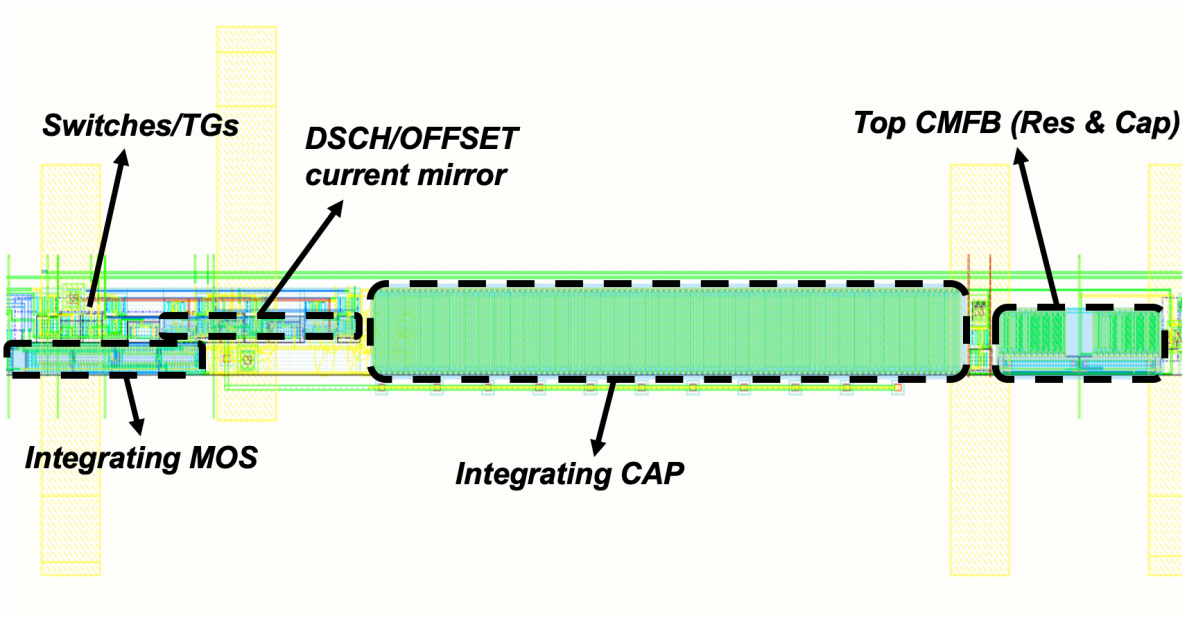


Figure 45: Integrating cap layout

VIII. Conclusions:

From the previous sections, we saw that the neuron consists of an integrator which computes weighted sum $\sum(Xi * Wi)$ and a comparator which acts as a ReLU unit. Integrator burn a total current of 250µA at 900mV of operating VDD and achieves 6-bit resolution using just 300fF of integrating capacitor.

Integrator provides an input impedance of less than 1KΩ while handling a minimum input current pulse of 1.5ns with lowest current resolution of 10nA. It occupies a total area of 283µm x 3 µm with current noise level as less as 8nA. Table 3 and 4 summarizes the integrator's important specifications.

Input Impedance	1/(gm1(A+1))
Time constant	Cpar/(gm1(A+1))

gm required	<1mS
Max variation at the input	<1mV
Tmin	~1.5ns
Tmax	64ns (6 bits)
Integrating cap C	300fF
Area of cap	3um*20um
Inoise	8nA
Imin/CTT	10nA
Power consumption	250uA*0.9V
Area	283um*3um

Table 3: Integrator's specifications

SL NO	PARAMETER	MIN	TYP	MAX	UNITS	COMMENTS
1	DC Accuracy	11.5m	14.2m	16.6m	V	Total current of 2.790µA; Cap varies by +/- 18%
		96.7m	115.4m	137.75m	V	Total max current (23.3095µA); Cap varies by +/- 18%
2	DC Gain	6	8.63	10.08	dB	Targeted low gain to avoid compensation components
3	Phase	35.6	42.7	53.8	Degree	Min phase at SS, 85°C, 400fF, 910mV
4	Gain margin	-9.13	-13.1	-16.83	dB	
5	BW	1.67	2.12	2.69	GHz	It can support min pulse width of 1.5ns
6	IQ	268.1	290	307.3	µA	

PVT PARAMETERS					
SL NO	PARAMETER	MIN	MAX	UNIT	COMMENTS
1	TEMP	-40	85	°C	Tested for -40°C to 125°C
2	VDD	850	910	mV	Tested for 810mV to 990mV
3	CAP*	400	550	fF	Tested for 250fF to 600fF
4	CORNERS	TT; SS; FF; SF; FS			Passive component variations are included in these corners

Table 4: AC simulation results

It can be seen from table 4 that the DC gain is as low as ~8dB so that we reduce the area of the integrator by completely avoiding the use of compensation networks usually consisting of passive components (resistor and capacitors).

Comparator achieves a resolution of ~120µV with a total gain of ~2000 by burn a total current of ~250µA at nominal operating VDD of 900mV. Thus, the neuron consumes a total of 450µW of power (500µA*0.9V) and occupies a total area of ~320µm x 3µm, achieving 6-bit of resolution using 300fF of integrating capacitor.

IX. Acknowledgement:

I thank Professor S.S Iyer for providing me an opportunity to work on this project and for continuously guiding me through the process. I would also like to thank Professor S. Pamarti for his continuous guidance and reviews on every aspect of the design. Also, I would like to thank Professor Janakiraman from IIT Madras, India, for providing his inputs on the design aspects.

We would like to thank DARPA and Global Foundries for funding our project and tape outs! I am thankful to my colleagues Xuefeng, Steven and Frank for working and sharing their knowledge during the course of this project.

X. References:

[1] D. Chabi, W. Zhao, D. Querlioz, J.-O. Klein, "Robust Neural Logic Block (NLB) Based on Memristor Crossbar Array", *IEEE/ACM International Symposium on Nanoscale Architectures*, pp. 137-143, 2011.

[2] S. Raoux, G. Burr, M. Breitwisch, C. Rettner, Y.-C. Chen, R. Shelby, M. Salinga, D. Krebs, S.-H. Chen, H.-L. Lung, C. Lam, "Phase-change random access memory: A scalable technology", *IBM J. Res. Dev.*, vol. 52, no. 4/5, pp. 465-479, 2008.

[3] X. Gu, Z. Wan and S. S. Iyer, "Charge-Trap Transistors for CMOS-Only Analog Memory," in IEEE Transactions on Electron Devices, vol. 66, no. 10, pp. 4183-4187, Oct. 2019

[4] F. Khan, E. Hunt-Schroeder, D. Moy, D. Anand, R. Katz, D. Leu, J. Fifield, N. Robson, S. Ventrone, T. Kirihata, "A Multi-Time Programmable Embedded Memory Technology in a Native 14nm FINFET Process using Charge Trap Transistors (CTTs)," Proceedings of the Government Microcircuit Applications & Critical Technology (GOMACTech) Conference, March 2019

[5] Laurent Gatet, Helene Tap-Beteille, Marc Lescure, "Analog Neural Network Implementation for a Real-Time Surface Classification Application", *IEEE SENSORS JOURNAL*, vol. 8, no. 8, pp. 1413-1417, AUGUST 2008

[6] L. Gatet, H. Tap-Beteille, M. Descure, "Analog Neural Network Implementation for a Real-Time Surface Classification Application", *IEEE J. of Sensors*, vol. 8, no. 8, pp. 1413-1421, August 2008.

[7] M. Ngwar, J. Wight, "A fully integrated analog neuron for dynamic multi-layer perceptron networks", *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, pp. 1-8, 2015.

[8] S. Szczęsny, "High speed and low sensitive current-mode CMOS perceptron", *Microelectron. Eng.*, vol. 165, pp. 41-51, Nov. 2016.

[9] Szymon Szcze˛sny “0.3 V 2.5 nW per Channel Current-Mode CMOS Perceptron for Biomedical Signal Processing in Amperometry”, IEEE SENSORS JOURNAL, VOL. 17, NO. 17, SEPTEMBER 1, 2017

[10] B. Razavi, "A circuit for all seasons—The switched capacitor integrator", *IEEE Solid-State Circuits Mag.*, vol. 9, no. 1, pp. 9-11, Jan. 2017.

[11] P.V.Satya Challayya Naidu, Neeru Agarwal1, Neeraj Agarwal2, “Design & Analysis of Novel Comparator without biasing for high performance application”, 5th International Symposium on Next-Generation Electronics (ISNE), 4-6 May 2016

[12] B. Razavi, B.A. Wooley, "Design Techniques For High-Speed High-Resolution Comparators", *IEEE Journal of Solid-State Circuits*, vol. 27, no. 12, pp. 1916-1926, 1992.

[13] C.P. Chong, K.C. Smith, "The Design of a High-Resolution CMOS Comparators", *Proc. of Int'l Symposium on Circuits and Systems*, vol. 2, pp. 1427-1430, 1989.