Neuromorphic Computing

Raiyan Siddique and Marc Eftimie

December 2023

1 Introduction

The goal of this project is to explore and design two fundamental components of neuromorphic computing, the neuron and the synapse. Although there were many synapse and neuron designs researched, for the purposes of implementation we chose a Leaky Integration and Fire Neuron paired with a memristor emulator or Current Multiplier Charge Injector (CMCI) Synapse. LIF Neurons were chosen because they are the most common neuron used for high frequency high efficiency architectures. The CMCI synapse was chose to show integration with digital training schemes. The memristor emulator was chosen as it is the best way to simulate a memristor, which is where the field is heading in terms of synapse design.

2 Neuron

2.1 Schematic

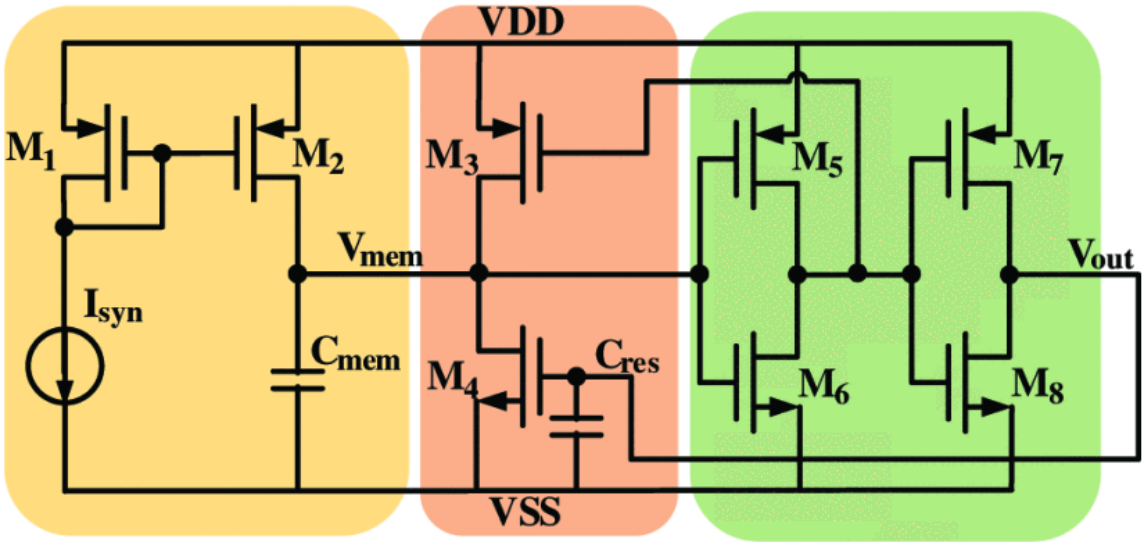


Figure 1: Original Design of Neuron based on reference paper[1]

The above circuit features three key blocks: the Orange Block serves as an Integration Block, utilizing a current mirror with transistors to replicate synaptic currents. The input current is mirrored and is integrated by C_{mem}, which progressively increases V_{mem}. The Red Block acts as a Reset Block, using transistors and a capacitor, discharging and resetting the membrane voltage after spike generation.

Finally, the Green Block, or Spike Generation Block, uses two back-to-back inverters to produce spikes, encoding integrated inputs and managing spike timing and intervals[1]. Compared to other designs in the literature, this neuron is compact and does not require any peripheral devices to generate any bias voltages, slowing variables or other miscellaneous controls.

2.1.1 Modifications

There are two main problems with the original design. The first is that the PMOS current mirror requires a pulling Synaptic current, whereas the currents generally provided by nearly all synapse design or pushing or injection currents. To remedy this a simple NMOS current mirror was used. The second design change that was made to separate the reset block from the synapse. Because any synapse would be directly connected to the reset block, which would effect the behaviour of the neuron, the output of the neuron was separated by two inverters. This separation ensures consistent behaviours, without interference from the synapse. The capacitor values were also increased from the paper to ensure spiking operation in a more appropriate current range for this technology.

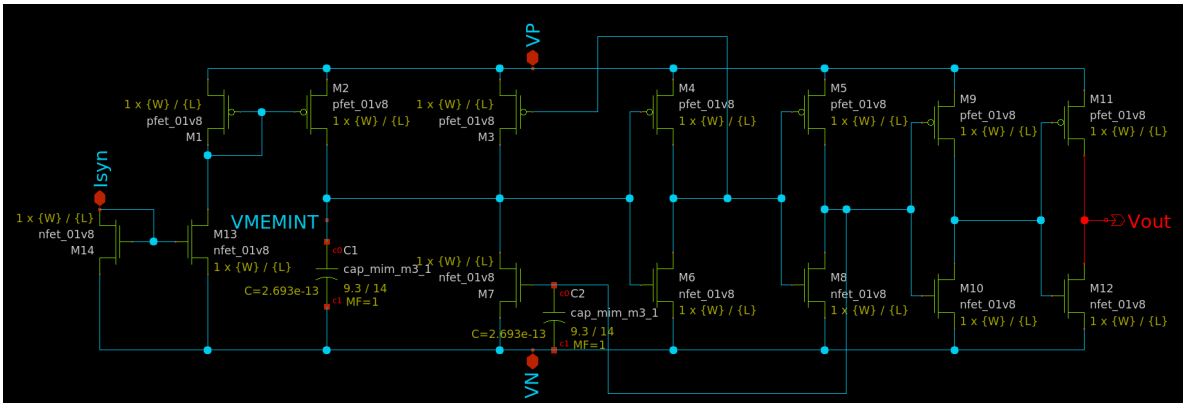


Figure 2: Final Design of Neuron

2.1.2 Simulation Results

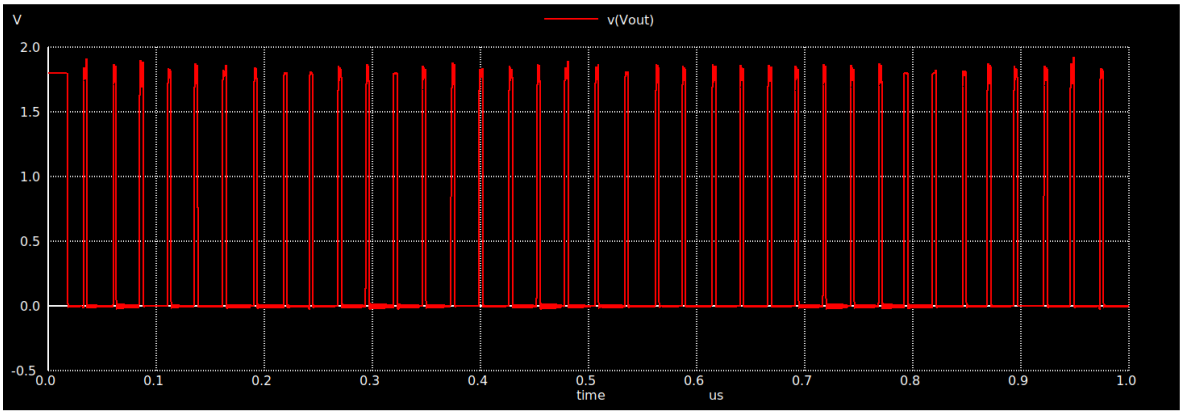


Figure 3: Output Spikes of Neuron given a constant Synaptic current

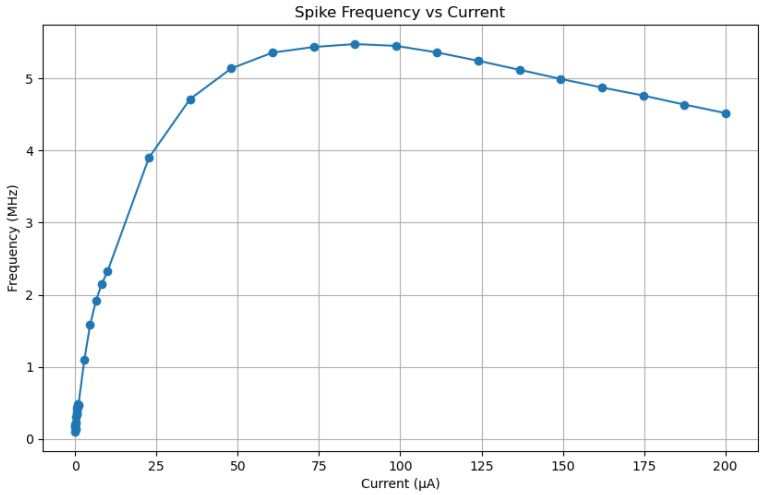


Figure 4: Relationship between Input Synaptic Current and Output Frequency. The safe operating region for an input synaptic current is from 0A to 50uA, and the circuit peaks a a spiking frequency of around 5.7mHz

From the above figures, our neuron has the necessary properties needed to be used in neuromorphic chips. It has the ability to generate output spikes at varying frequencies depending on the varying input current into the circuit. Although there is a drop-off in the frequency past 75uA, in an real neuromorphic application the neuron would almost never be put under those conditions, as the inputs into the neuron are also spikes.

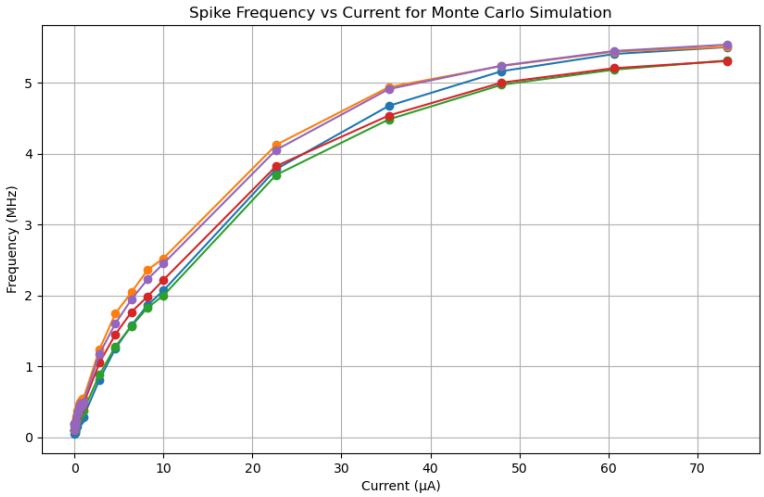


Figure 5: Monte Carlo simulation of Neuron.

The goal of the Monte Carlo simulations of the neuron is to show a consistent characteristic in spiking frequency as current increases. The transistors are sized to be double minimum width and length, as having the transistors be any smaller causes significant deviance from what should be a smooth logarithmic curve. Any of the variations in the actual numerical value of frequency is not of concern, as the frequencies are close enough to be considered consistent for this application. Furthermore, the training of neuromorphic chips means that when the synapses are weighted and used these variations are considered and circuit is able to adapt to them.

2.1.3 Layout

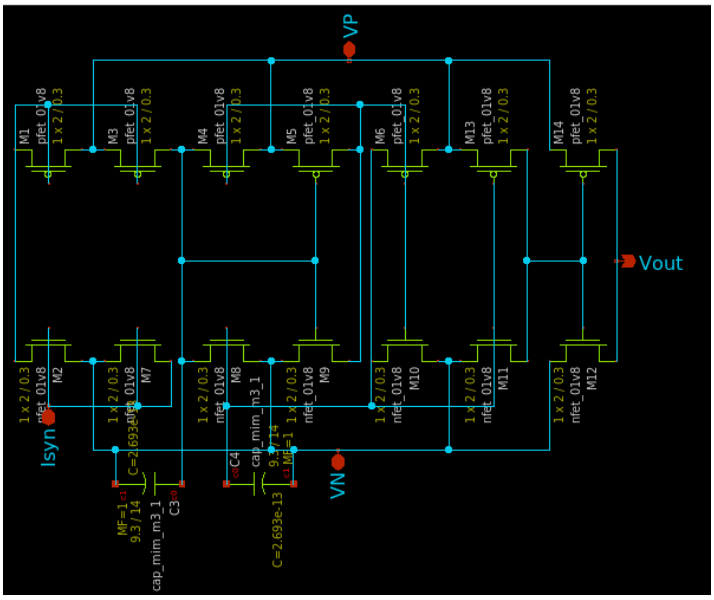


Figure 6: Neuron Layout Driven Schematic. No Analog Layout techniques were used because any matching differences would be considered when training the circuit.

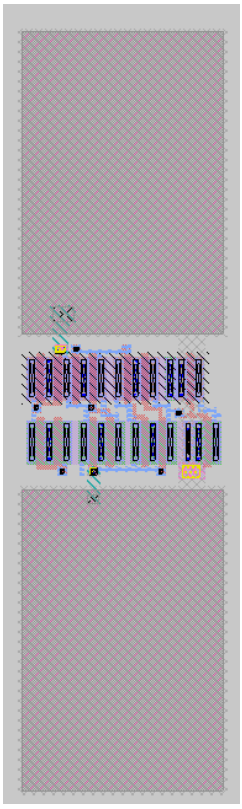


Figure 7: Neuron Layout of size 9.620um x 35.520um

2.1.4 Test Harnesses and Post Layout Simulation Results

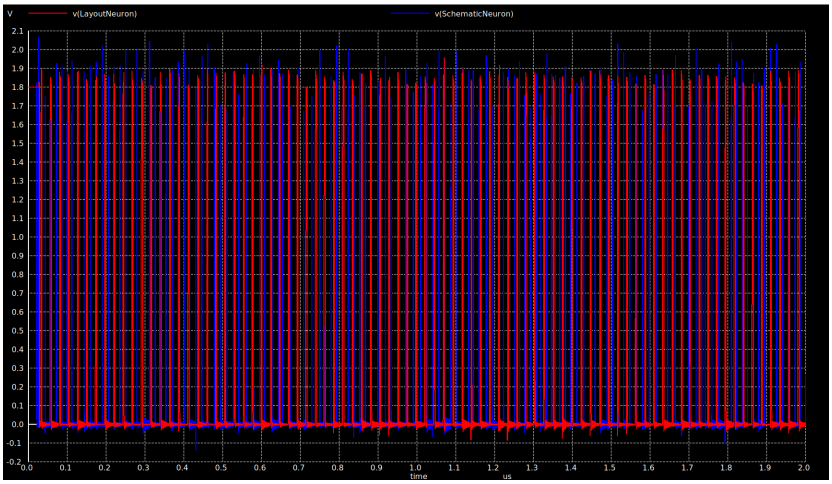


Figure 8: Results comparing layout neuron (blue) and schematic neuron (red). The results are almost identical.

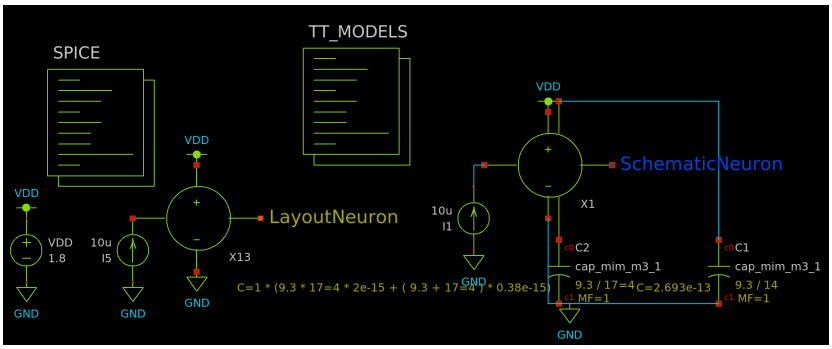


Figure 9: Test Harness to compare the layout of neurons to the schematic of the neuron.

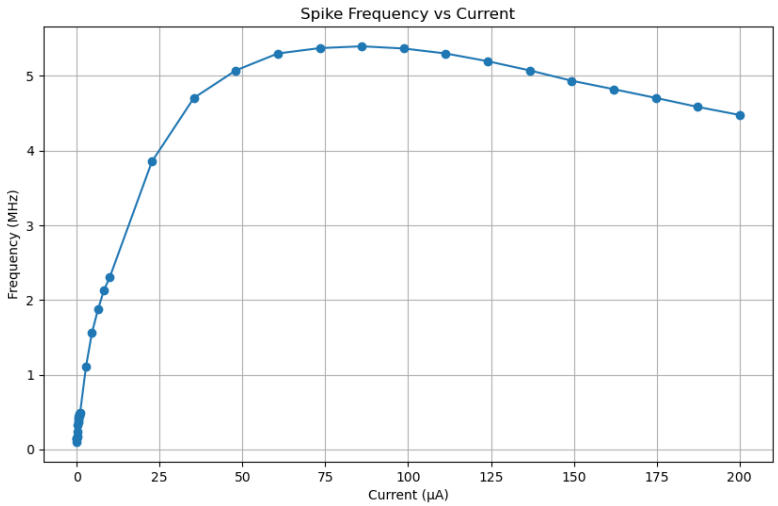


Figure 10: The frequency analysis of the layout. Although the peak is slightly lower, the behaviour of the neuron is nearly identical to the schematic.

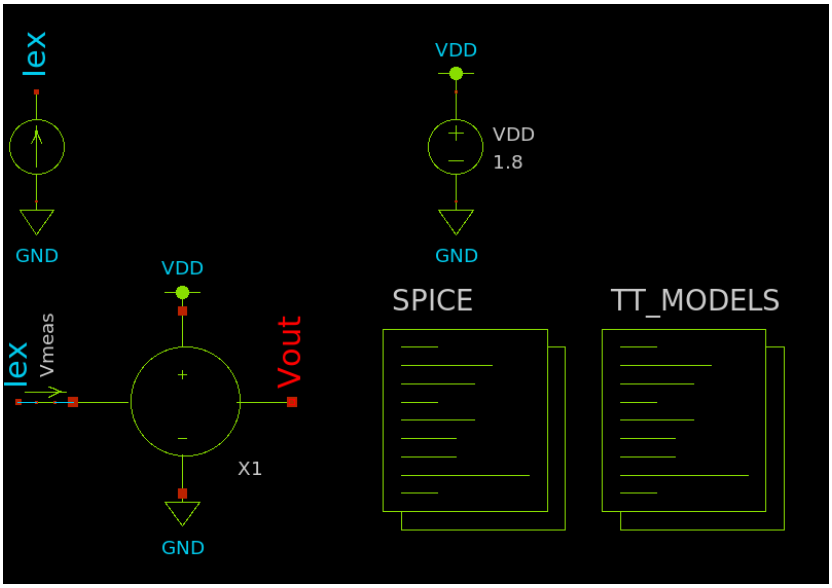


Figure 11: Test Harness to that is used to get the spiking frequency vs current relationship. Iex is stepped up using a PWL function.

2.1.5 LVS

```
Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__cap_mim_m3_1 and Circuit 2 cell sky130_fd_pr__cap_mim_m3_1 are black box
Equate elements:  no current cell.
Device classes sky130_fd_pr__cap_mim_m3_1 and sky130_fd_pr__cap_mim_m3_1 are equivalent.
Flattening unmatched subcell neuron in circuit neuron_top.spice (0)(1 instance)
Flattening unmatched subcell besrour_neuron_cm_lds_lvs in circuit neuron_top_lvs_cap.spice (1)(1 inst

Subcircuit summary:
Circuit 1: neuron_top.spice                |Circuit 2: neuron_top_lvs_cap.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__nfet_01v8 (7)                |sky130_fd_pr__nfet_01v8 (7)
sky130_fd_pr__pfet_01v8 (7)                |sky130_fd_pr__pfet_01v8 (7)
sky130_fd_pr__cap_mim_m3_1 (2)             |sky130_fd_pr__cap_mim_m3_1 (2)
Number of devices: 16                      |Number of devices: 16
Number of nets: 9                          |Number of nets: 9
-----------------------------------------------------------------------------------------
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes neuron_top.spice and neuron_top_lvs_cap.spice are equivalent.

Final result: Circuits match uniquely.
.
```

3 Synapse

3.1 Current Multiplier Charge Injector Synapse

After picking a Neuron, our focus was finding a synapse option that was compatible with Leaky Integration and Fire models. One avenue of exploration was the Current Multiplier Charge Injector Synapse[5].

3.1.1 CMCI Design

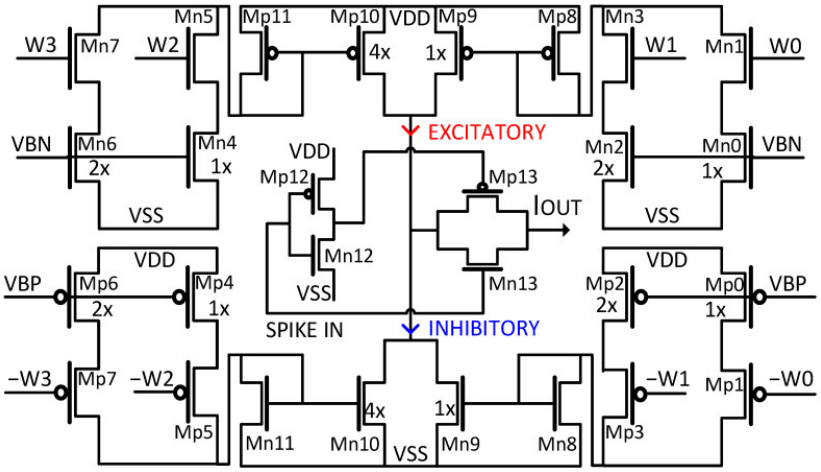


Figure 12: CMCI Design from Paper[5]

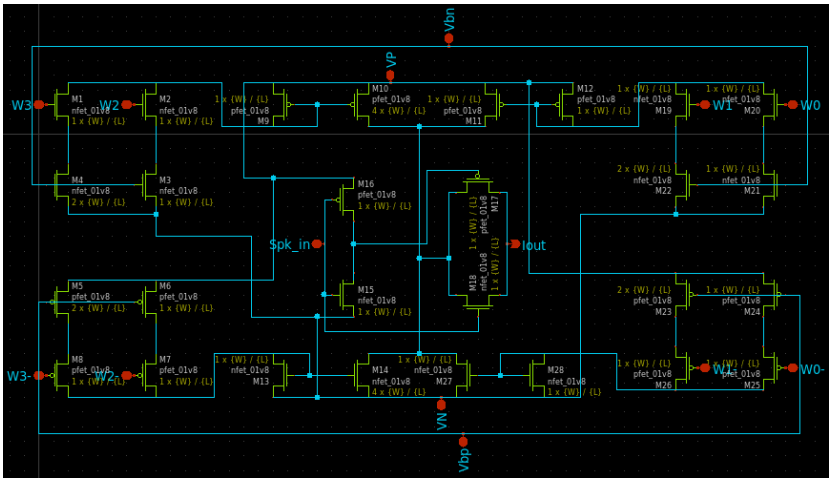


Figure 13: Implemented Schematic of Circuit. The chosen width is 3um and the chosen length is 0.5um. These sizes were chosen as the result of Monte Carlo simulations to ensure that any variation in the circuit does not break its behaviour.

The synapse circuit in the discussed design operates on a Current Multiplier Charge Injector (CMCI) mechanism. This circuit incorporates binary-weighted current mirrors divided into excitatory and inhibitory portions, each consisting of NMOS and PMOS branches. When an input spike is received, the weight of the synapse determines the current's magnitude to be either injected or ejected. The final current is then accumulated on or ejected from the neuron's membrane potential capacitor. A transmission gate is used on the output to prevent the circuit from effecting other synapses when there is no high input coming into the circuit. This CMCI circuit would need 5 bits per synapse, where the

MSB determines the sign on whether or not the negative weights are turned low, which turns on the inhibitory circuit, or positive weights are turned high to excite the current.

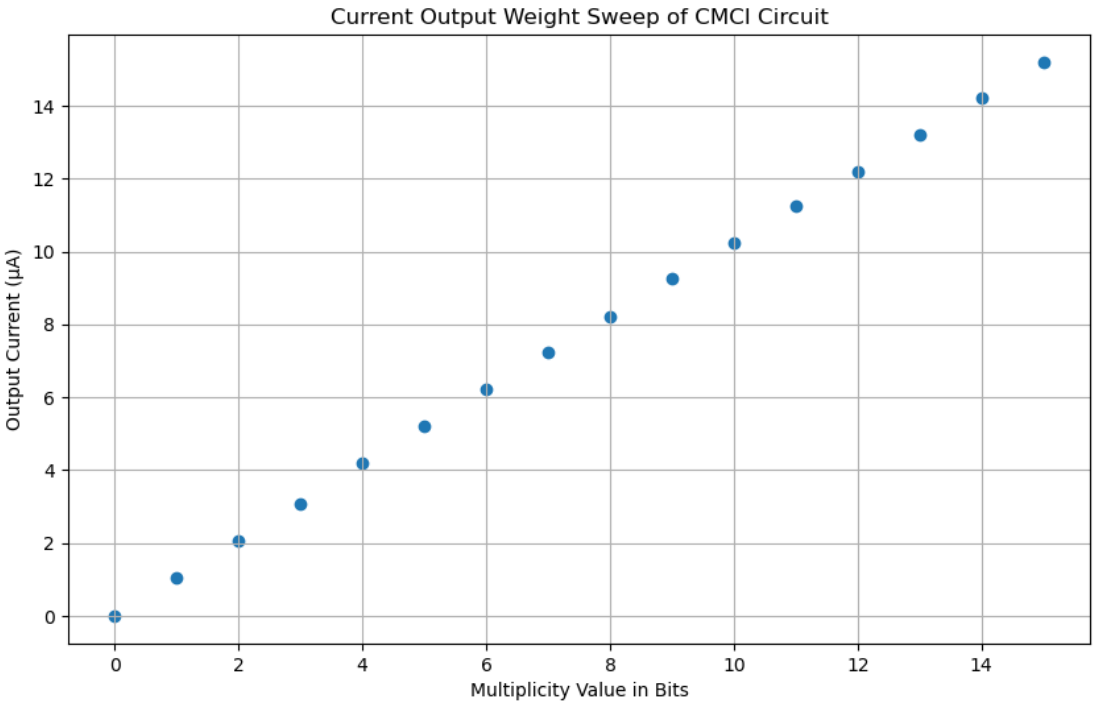


Figure 14: Linear Relationship between stepping up weight and the increasing output current

Above shows, the exciting behaviour of the circuit, where stepping up in weights linearly increase the output current.

3.1.2 Neuron Synapse Connection

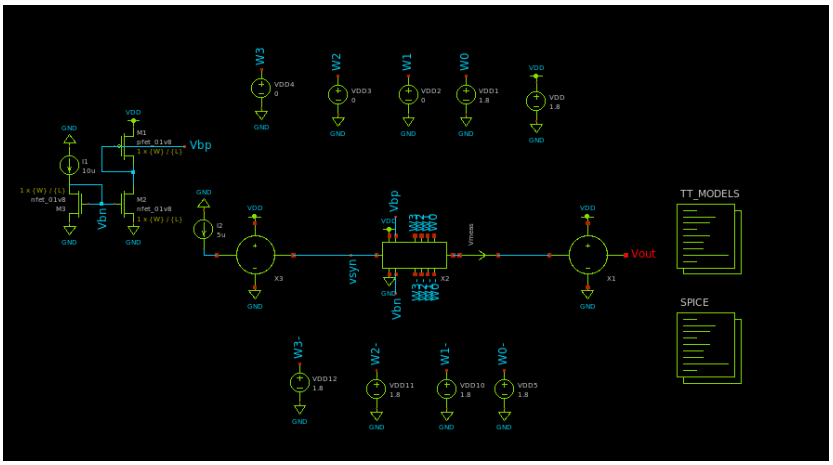


Figure 15: Test Bench to connect CMCI Synapse to Neuron. The Neuron connected on the input stage is meant to generate spikes for reference.

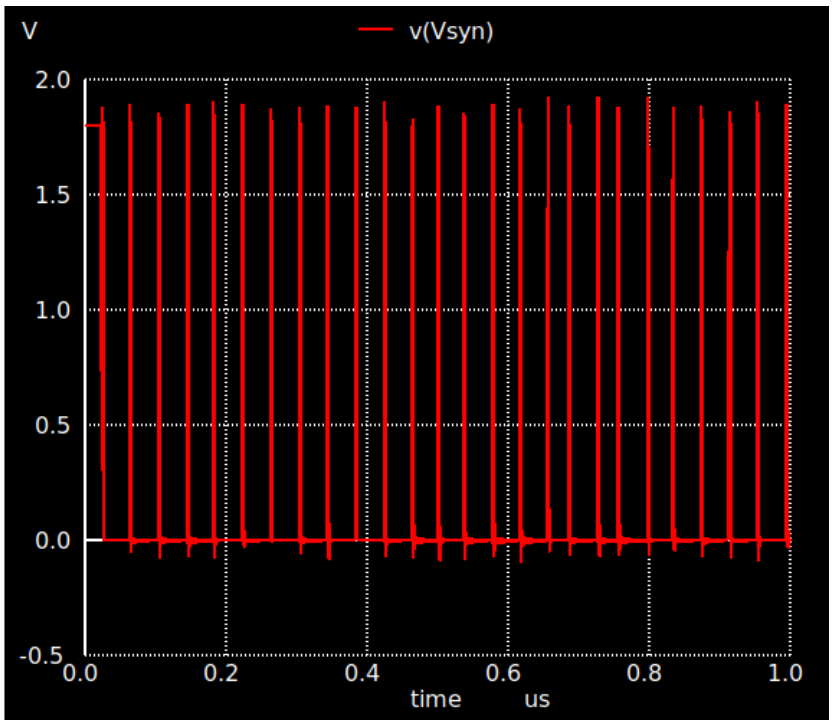


Figure 16: Spikes Coming out of the First Neuron and being pushed into the synapse.

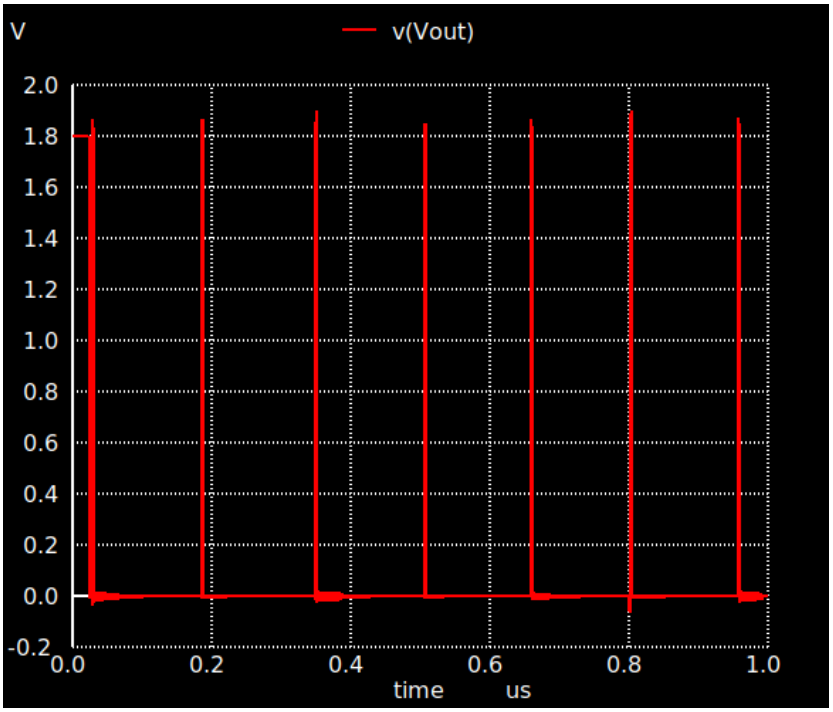


Figure 17: Output Spikes of Neuron when the weight excitory (positive weights) are 0001

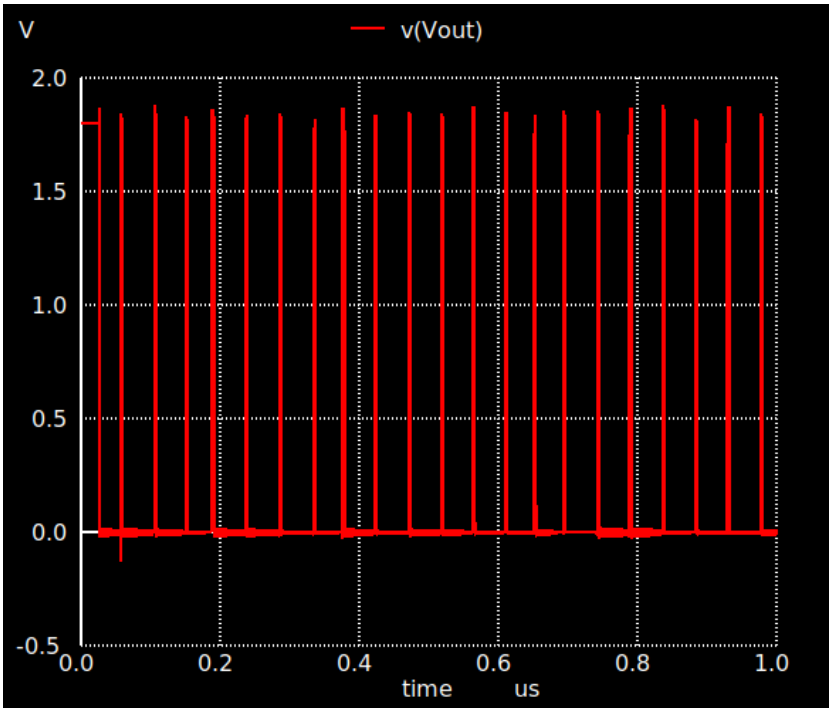


Figure 18: Output Spikes of Neuron when the weight excitory (positive weights) are 1001. It is evident that it is now significantly faster than the previous weight output.

The above figures show that our synapse can be used to increase or decrease the spiking frequency at the output. Although this highlights the effectiveness of our design, this is not necessarily a useful structure. Most neural networks have multiple synapes connected to the input of the neuron.

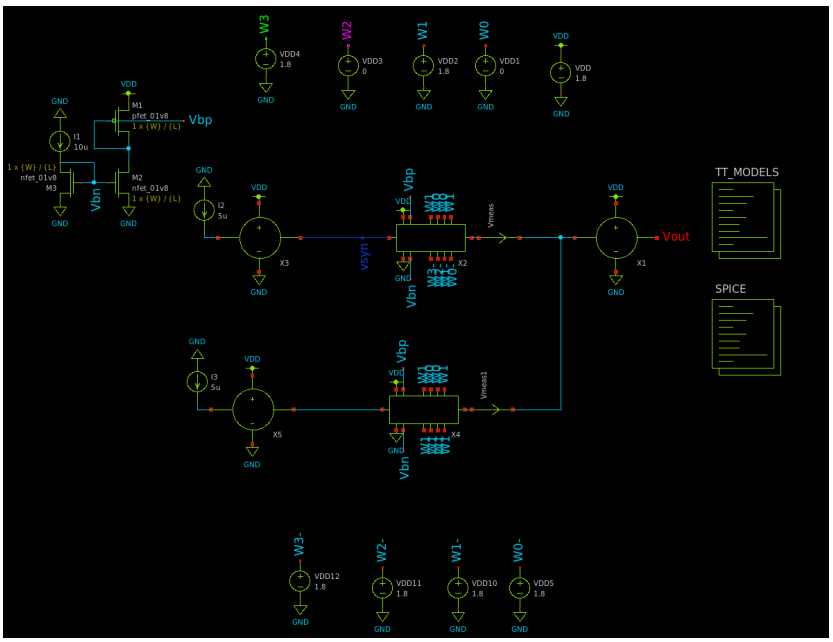


Figure 19: Test Bench Connecting two synapses into the input of one neuron.

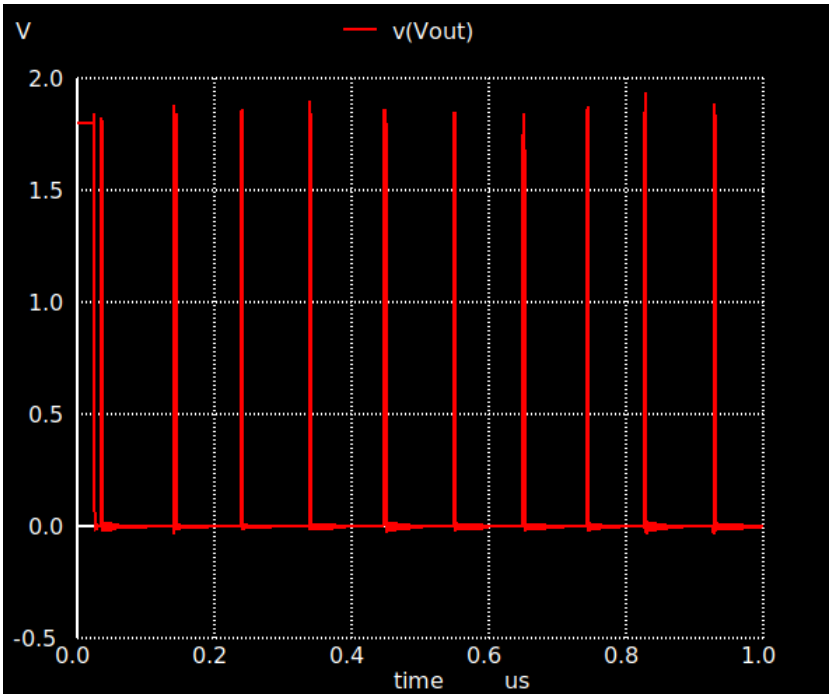


Figure 20: When both synapses receive the same input, and have a weight of 0001, they excite each other and increase the spiking frequency at the output neuron.

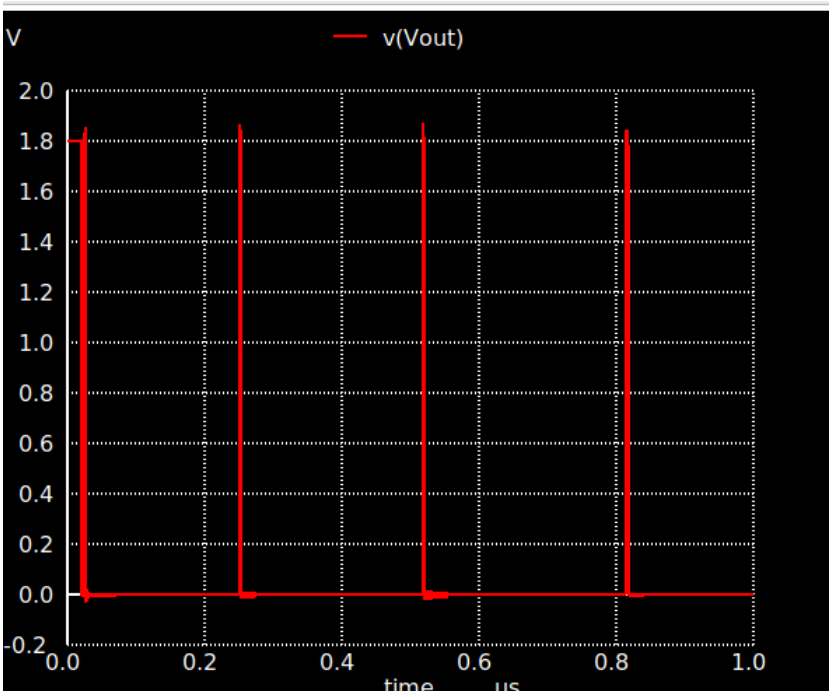


Figure 21: When both synapses receive the same input, and have one has positive weight of 0001 and one has a negative weight of 0111, one inhibits the other, reducing the spiking frequency at the output neuron.

This shows that our system can be used to emulate both positive and negative weights and can be put in a configuration to be used to replicate almost any neural network. The CMCI synapse is compatible with almost every digital training scheme, meaning that it can be integrated into a variety of systems.

3.1.3 Layout

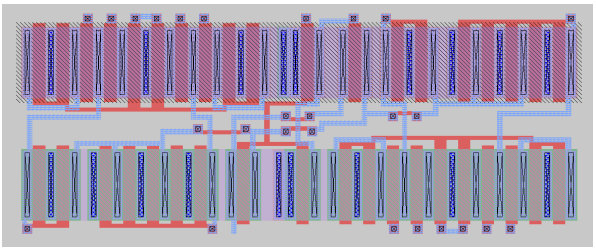


Figure 22: Layout for CMCI circuit. It is 4.100um x 3.795um

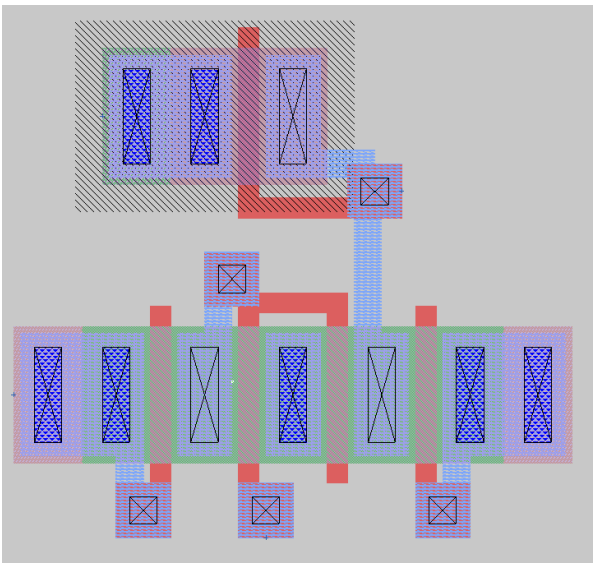


Figure 23: Layout for Bias Generator circuit. It is 23.800um x 9.250um

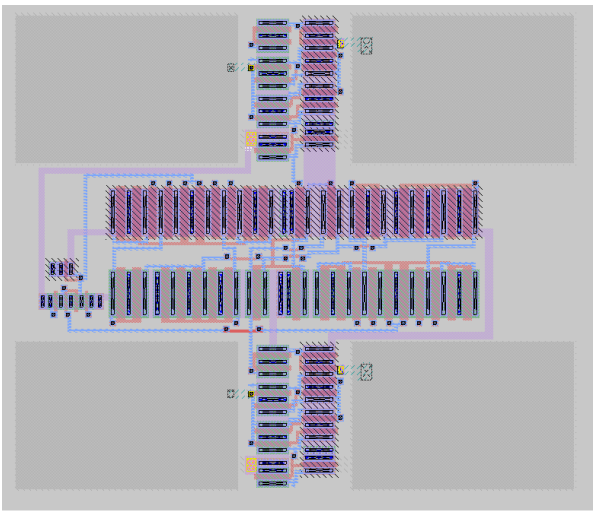


Figure 24: Neuron Synapse Neuron configuration for the CMCI circuit. It is 35.520um x 30.270um

3.1.4 Post Layout Simulation

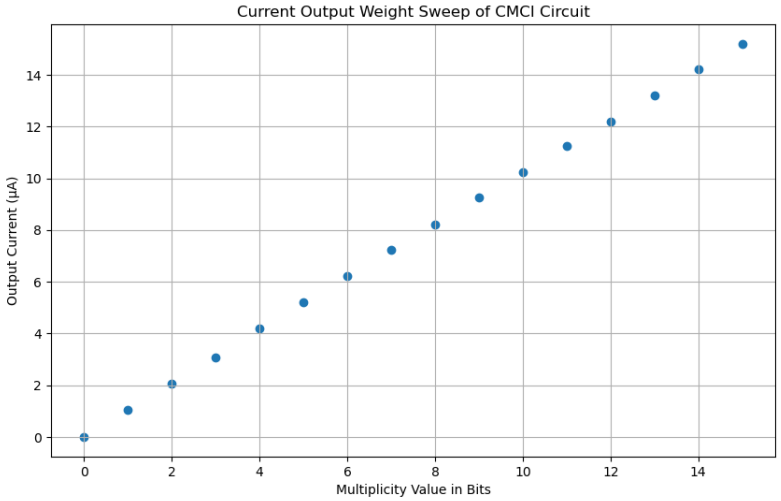


Figure 25: Post Layout Simulation of Current Output vs Input Positive Weight. The behaviour is nearly identical to the schematic.

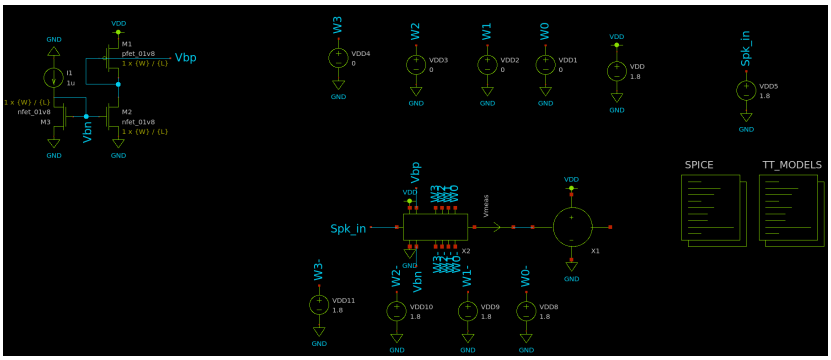


Figure 26: Test harness of Current Output vs Input Positive Weight

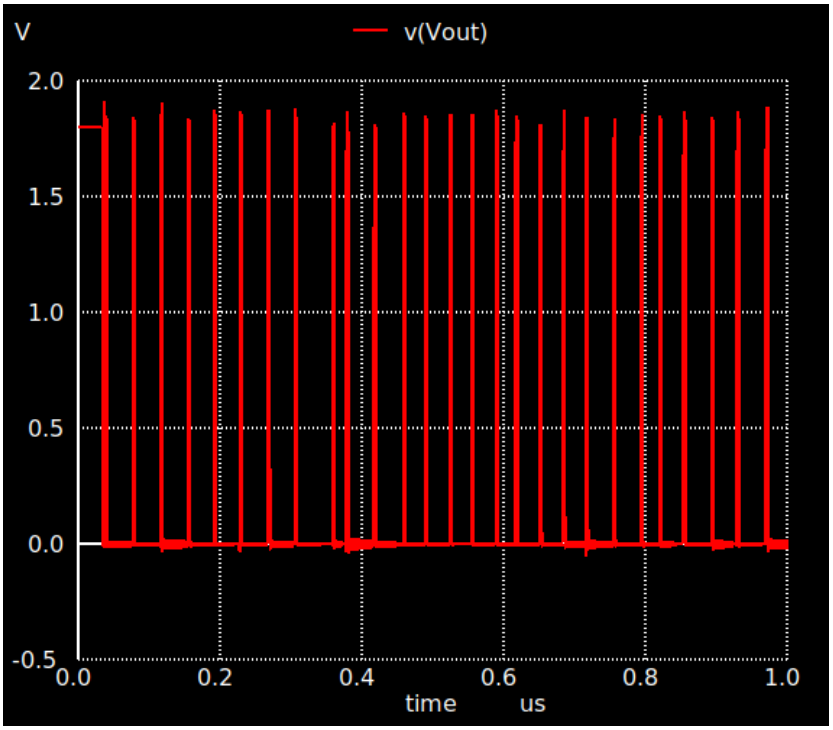


Figure 27: Output of Neuron Synapse Neuron Connection when positive weight is 1001

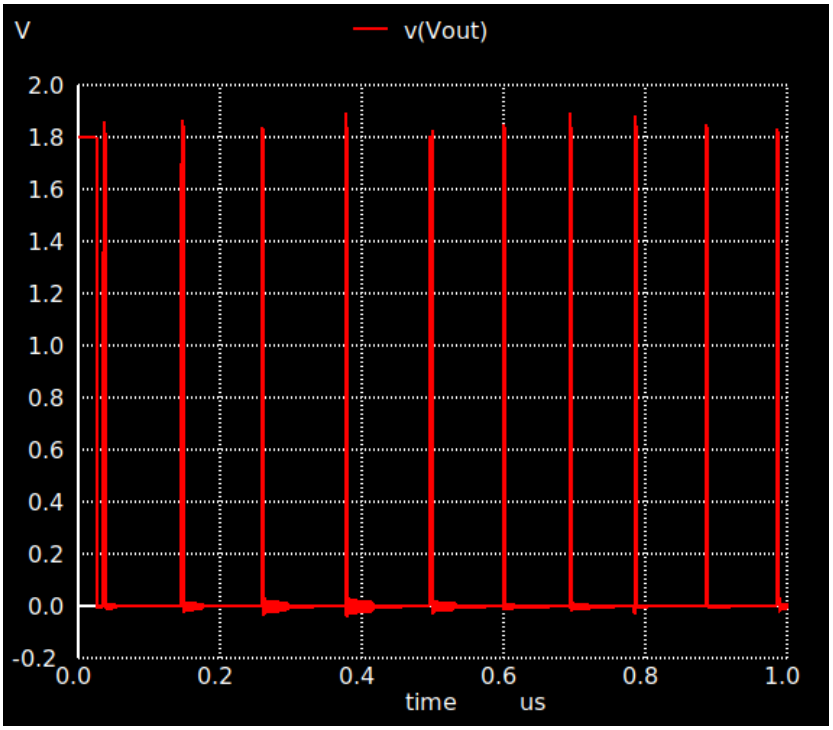


Figure 28: Output of Neuron Synapse Neuron Connection when positive weight is 0001. The output spikes are less frequent than when at a higher weight.

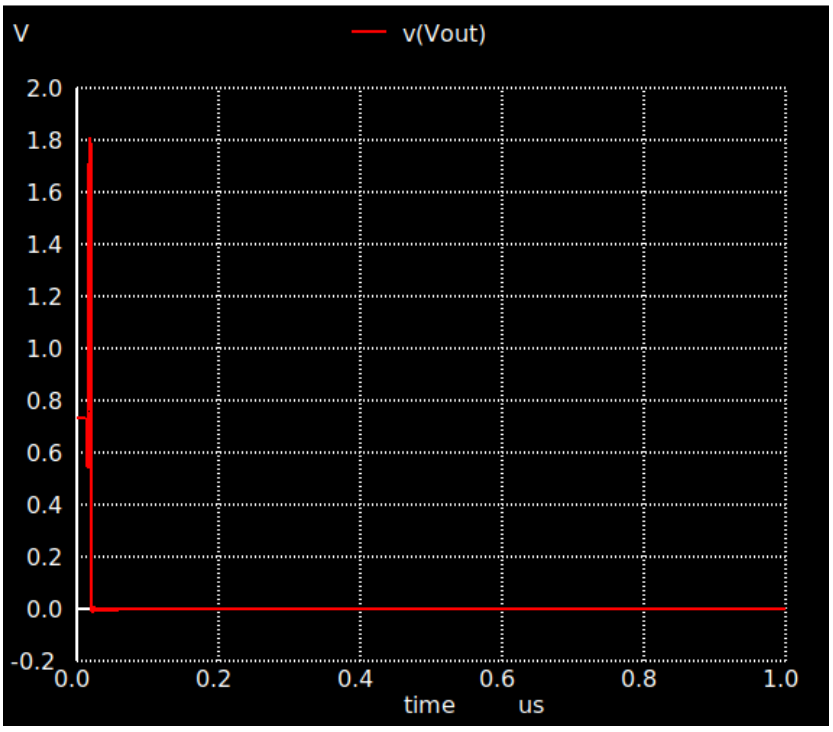


Figure 29: Output of Neuron Synapse Neuron Connection when negative weight is 0111. The output spikes are fully inhibited.

The postlayout simulations of the behaviour of the CMCI synapse and of it connected in the neuron-synapse-neuron configuration discussed in previous sections, shows that that the layout and schematic and essentially identical behaviour.

3.1.5 LVS

CMCI Synapse

```
Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.
Flattening unmatched subcell CMCI_synapse in circuit CMCI_AND_NEURON.spice (1)(1 instance)

Class CMCI_synapse_lds.spice (0):  Merged 10 parallel devices.
Class CMCI_AND_NEURON.spice (1):  Merged 10 parallel devices.
Subcircuit summary:
Circuit 1: CMCI_synapse_lds.spice          |Circuit 2: CMCI_AND_NEURON.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__nfet_01v8 (19->14)           |sky130_fd_pr__nfet_01v8 (19->14)
sky130_fd_pr__pfet_01v8 (19->14)           |sky130_fd_pr__pfet_01v8 (19->14)
Number of devices: 28                      |Number of devices: 28
Number of nets: 28                         |Number of nets: 28
-----------------------------------------------------------------------------------------
Resolving symmetries by property value.
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes CMCI_synapse_lds.spice and CMCI_AND_NEURON.spice are equivalent.
```

```
Final result: Circuits match uniquely.
.
```

LVS of Neuron-Synapse-Neuron Configuration

```
Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__cap_mim_m3_1 and Circuit 2 cell sky130_fd_pr__cap_mim_m3_1 are black box
Equate elements:  no current cell.
Device classes sky130_fd_pr__cap_mim_m3_1 and sky130_fd_pr__cap_mim_m3_1 are equivalent.
Flattening unmatched subcell besrour_neuron_cm_lds in circuit besrour_CMCI_lds_lvs.spice (0)(2 instan
Flattening unmatched subcell CMCI_synapse_lds in circuit besrour_CMCI_lds_lvs.spice (0)(1 instance)
Flattening unmatched subcell bias_gen_lds in circuit besrour_CMCI_lds_lvs.spice (0)(1 instance)
Flattening unmatched subcell bias_gen in circuit CMCI_besrour_top.spice (1)(1 instance)
Flattening unmatched subcell CMCI_synapse in circuit CMCI_besrour_top.spice (1)(1 instance)
Flattening unmatched subcell neuron_top in circuit CMCI_besrour_top.spice (1)(2 instances)
Flattening unmatched subcell neuron in circuit CMCI_besrour_top.spice (1)(2 instances)

Class besrour_CMCI_lds_lvs.spice (0):  Merged 10 parallel devices.
Class CMCI_besrour_top.spice (1):  Merged 10 parallel devices.
Subcircuit summary:
Circuit 1: besrour_CMCI_lds_lvs.spice      |Circuit 2: CMCI_besrour_top.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__pfet_01v8 (34->29)           |sky130_fd_pr__pfet_01v8 (34->29)
sky130_fd_pr__nfet_01v8 (37->32)           |sky130_fd_pr__nfet_01v8 (37->32)
sky130_fd_pr__cap_mim_m3_1 (4)             |sky130_fd_pr__cap_mim_m3_1 (4)
Number of devices: 65                      |Number of devices: 65
Number of nets: 40                         |Number of nets: 40
-----------------------------------------------------------------------------------------
Resolving symmetries by property value.
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes besrour_CMCI_lds_lvs.spice and CMCI_besrour_top.spice are equivalent.

Final result: Circuits match uniquely.
.
```

Bias Generator LVS

```
Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.

Subcircuit summary:
Circuit 1: bias_gen.spice                  |Circuit 2: bias_gen_lds.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__nfet_01v8 (4)                |sky130_fd_pr__nfet_01v8 (4)
```

```
sky130_fd_pr__pfet_01v8 (1)                |sky130_fd_pr__pfet_01v8 (1)
Number of devices: 5                       |Number of devices: 5
Number of nets: 4                          |Number of nets: 4
-----------------------------------------------------------------------------------------
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes bias_gen.spice and bias_gen_lds.spice are equivalent.

Final result: Circuits match uniquely.
.
```

3.2 Memristor Emulator

One fundamental component of modern Neuromorphic computing is the memristor or "memory resistor". The memristor is a passive circuit element that maintains a relationship between the time integrals of current and voltage across a two-terminal element. The purpose of this section of exploration into the memristor was to find a CMOS compatible memristor emulator that can exhibit the fundamental behaviours that define a memristor.

3.3 Original Paper Design

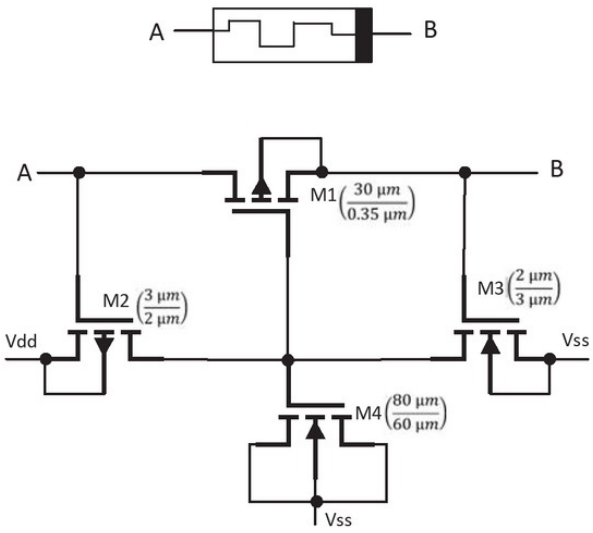
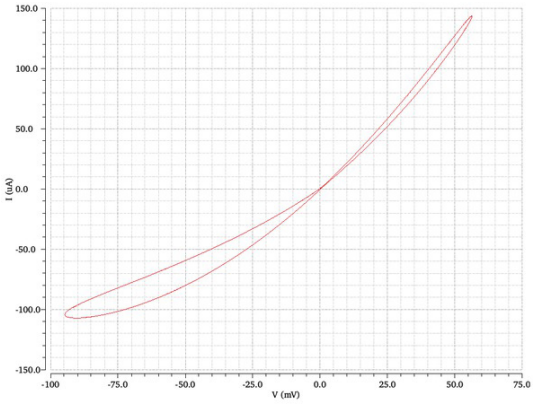
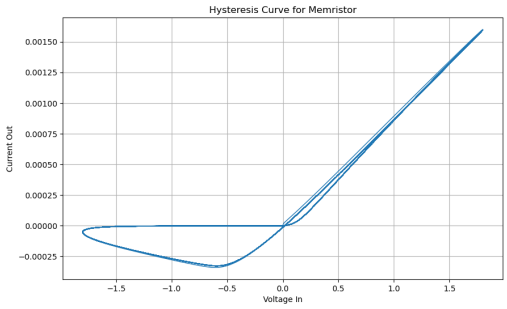


Figure 30: Orignal 4 Transistor CMOS Memristor Emulator[2]

The above design is a simple 4 transistor CMOS memristor emulator that exhibits the same fundamental properties that makes a memristor useful for various applications. The drain to source resistance across M1 is the resistance of the memristor, which is changed by the voltage of M4. M4 is connected as a capacitor, where the drain, source and bulk are connected to the negative voltage supply. M2 and M3 function as feedback circuits that change the voltage across M4[2]. Although there were other available designs for memristor emulator, this one was chosen for its simplicity and the wide range of resistance levels, which will be discussed later.

3.3.1 Hysteresis Curve





1

Figure 31: Left is the results of hysteresis curve at 2MHz from the original paper, which is run at a magnitude of 200mV[2]. Right is the results of our replicated circuit at 2Mhz but from +1.8V to -1.8V. The reason for this discrepancy is that generally, hysteresis curve are generated from positive to negative power supply to capture the full range of operation of the memristor.

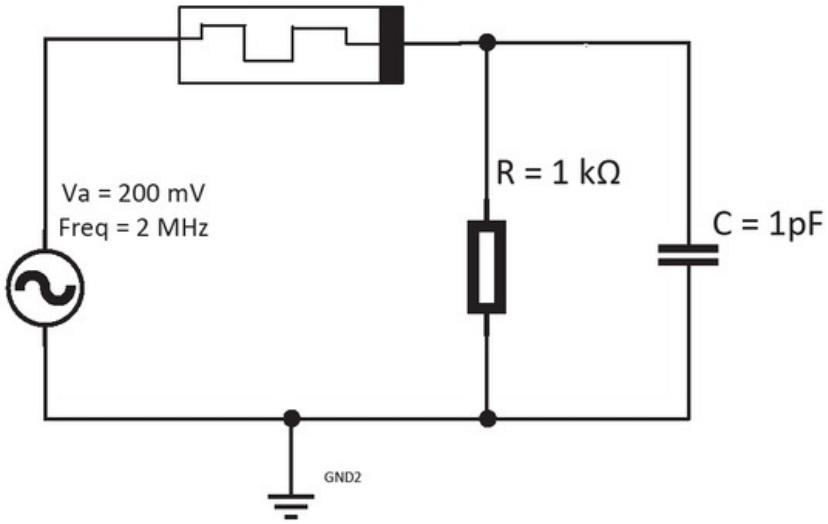


Figure 32: Original test harness used to generate both hysteresis graphs.[2] The only difference between the simulation ran between our work and the referenced paper is that our memristor emulator is tested from 1.8V to -1.8V instead of the 200mV range of the referenced paper.

The hysteresis curve is the most important characterization of the useful properties of the memristor. It shows that the memristors current state changes based on its previous state, proving that it is able to store memory. The hysteresis curve give both the read and write regions for the memristor, and characterizes the on and off resistance of the memristor. These graphs are a benchmark to prove that a given design has the ability to store memory and has the general properties of the memristor.

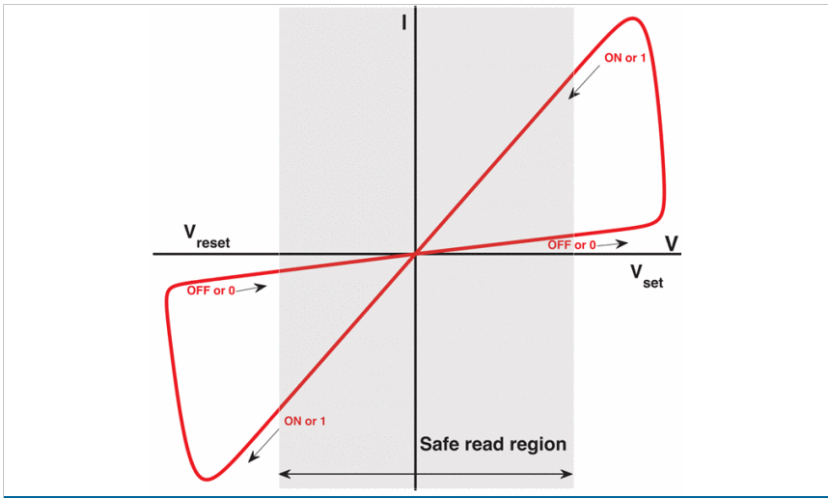


Figure 33: Plot to show the safe read region of the memristor as well as its on and off states based on previous voltage[3]. The write region is towards the ends of the graph, which in our case is close to the power supply.

The only drawback of this design is the large sizing of M4 and M1, which cause the circuit to take up 70um x 85um^2. In more advanced neuromorphic designs, the size of the synapse if important as we want to densely pack and fit as many synapses as we can into a given chip.

3.3.2 Sized Down Design

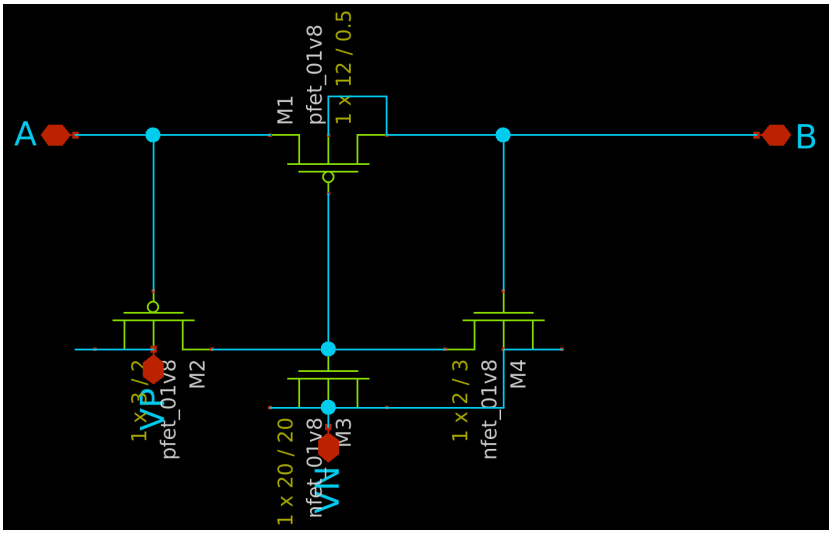


Figure 34: Schematic of sized down memristor emulator.

In order to size down the design, we focused on two transistors M4 and M1. M1 was sized down to 12/0.5. Although this down sizing in the width and increase in length, increases the drain source resistance of M1, when done in conjunction with reducing the size of the NMOS capacitor that is M4 it increases the stability of the hysteresis curve, especially at the ends of the power supply. M4 was sized down to 20um x 20um, but kept large enough to still store charge and maintain the characteristics of the memristor. This NMOS capacitor was not sized down arbitrarily but was instead taken from a different memristor emulator that uses a comparable structure[4]. The results of simulation show that the memristor has comparable hysteresis curve and this smaller design still preserves the properties of the memristor.

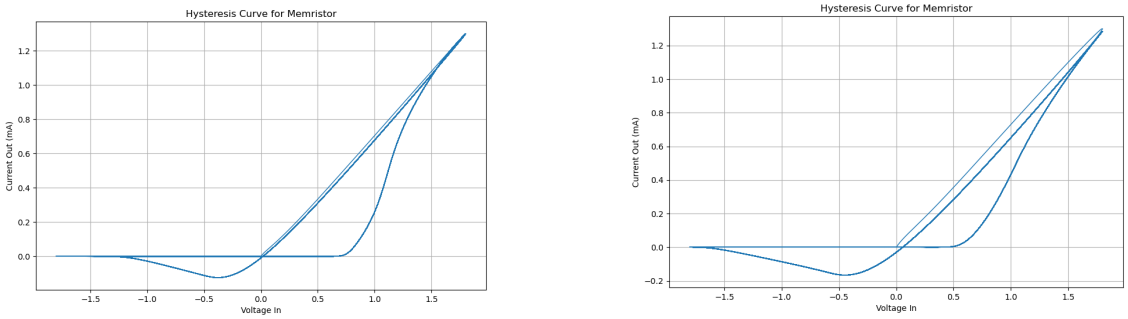


Figure 35: Left is the hysteresis curve at 3Mhz from -1.8 to 1.8. Right is the hysteresis at 9Mhz. These graphs are comparable to other memristor emulators as shown below.

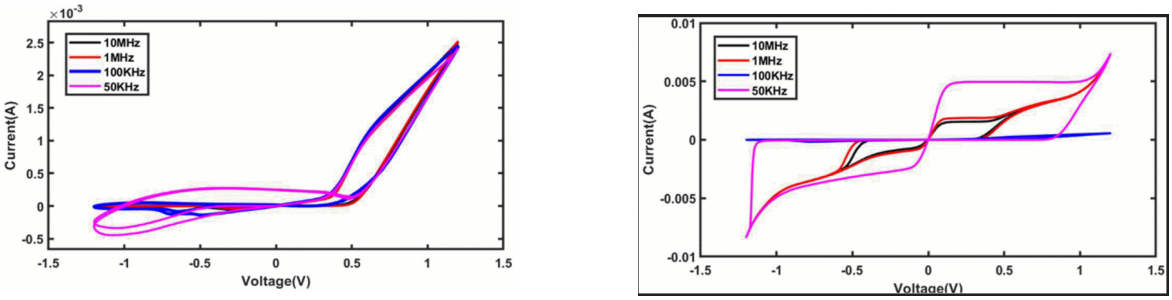


Figure 36: Hysteresis graphs for comparison from other memristor emulators in the literature[4].

When put in layout, the final design takes up an area of 21.500 x 24.440, which is more than 11 times smaller in area than the proposed paper design. No common centroid or dummy devices were added, as to keep the size and design small, in order to, theoretically, densely pack the synapses, as any mismatch would be taken care off when a circuit using this design is trained.

3.3.3 Memristor with External Transistor as a Tunable Resistor

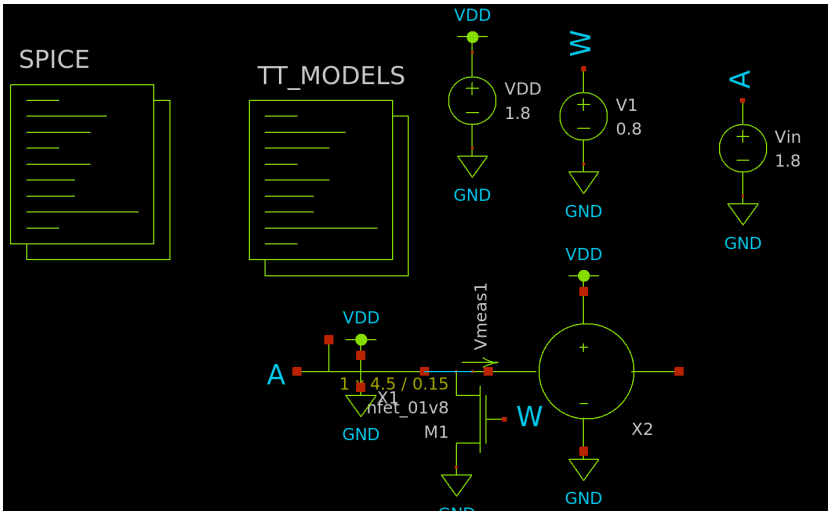


Figure 37: Test Bench configuration for Memristor with External Transistor

Due to the constraints of the technology, as well as for integration with the proposed neuron design, the memristor's VSS is grounded. This has the effect of flattening the negative voltage component of the hysteresis graph.

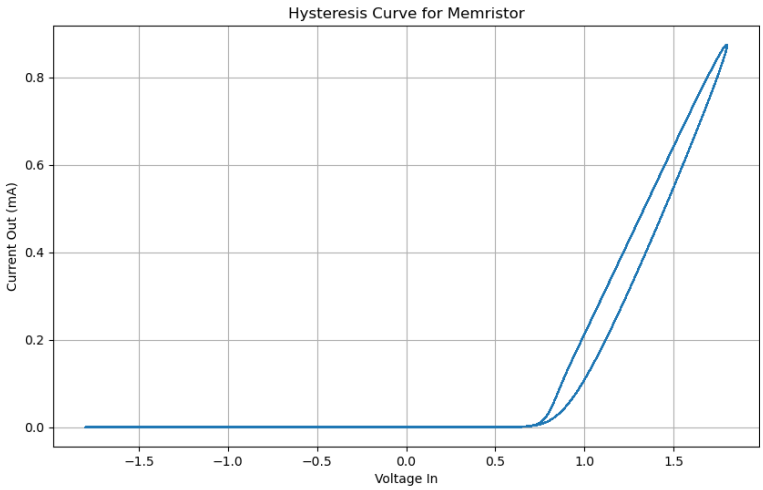


Figure 38: Frequency Analysis of Grounded Memristor Emulator to show attempted Hysteresis Curve

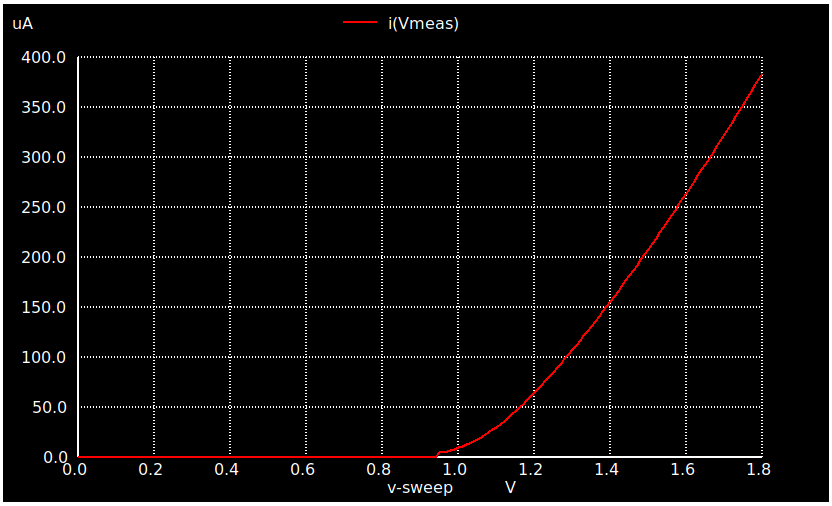


Figure 39: Current Voltage Characteristic of Grounded Memristor. Because the input into the synapse is spikes, rather than a constant voltage, for use in a spiking neural network, this current level is more than suitable. Another interesting application of this circuit that can be explored in the future is its use as a RELU function.

However, in order to add tunable resistor values, we can use the 1 Transistor 1 Memristor configuration to further tune output current of the memristor[2]. Not only is this useful for our use case as tunable resistor, but when used in a proper VSS to VDD configuration, this transistor doubles as a training mechanism for the memristor[2].

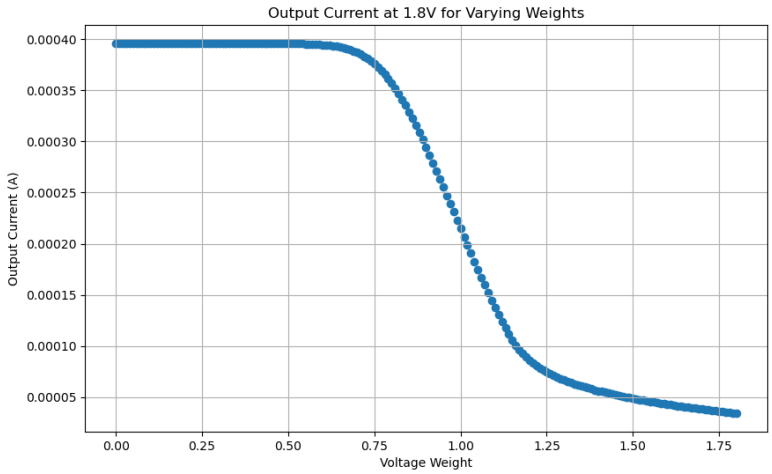


Figure 40: Output current as a function of Voltage Weight

With this resistor-like configuration, we maintain the properties of being able to tune the synapse to be almost entirely excitatory or inhibitory. By being able to drop the output current to close to 0 or up to 400uA means that we can use this circuit in nearly all training analog schemes for spiking neural networks. The last modification made to this circuit was adding a transmission gate to separate the output of the synapse when connected to the neuron. Thus, the synapse is only effects the neuron when a spike occurs. This idea was taken from the synapse option discussed earlier[5], where the inhibitory or excitatory current is separated from the input into a neuron using a transmission gate.

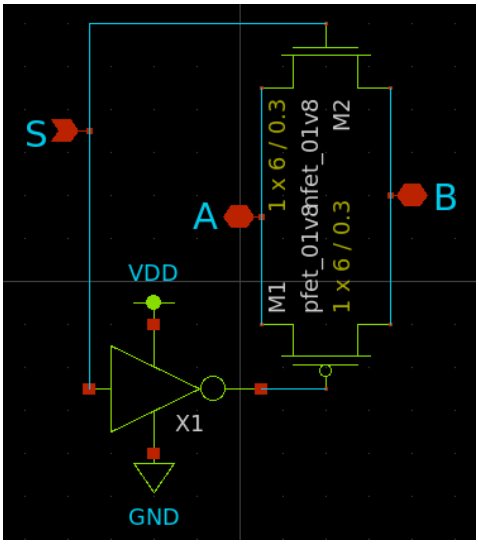


Figure 41: Transmission gate design used in this report. The sizing is made to be as small as possible without breaking in Monte Carlo simulation.

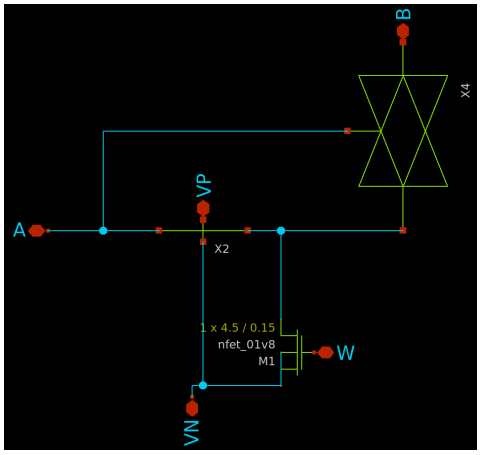


Figure 42: Circuit of Synapse that is grounded because of the conductivity of the p-subtrate diffusion layer.

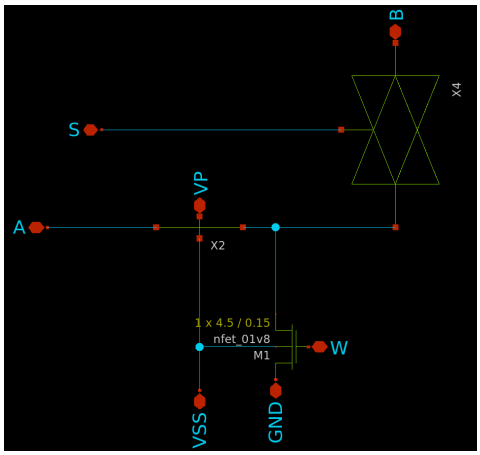


Figure 43: Theoretical circuit that can be used to train the memristor.

Above is the a theoretical circuit that can be used to train the memristor, if put in a technology that allows two different NMOS Bulk voltages. Once placed in a circuit, the select on the transmission gate can either be shorted to the input when being used in a forward propogation phase, or put into another select pin for training. When the select is low, the input of the memristor can be pulsed in intervals and W can be set to a specific value to set the memristor to a specific value[2].

3.4 Neuron Synapse Connection

To show the use cause of this system, we use a simple neuron-synapse-neuron configuration to show that our system can be used to change the spiking frequency of a neuron using varying weights on the synapse.

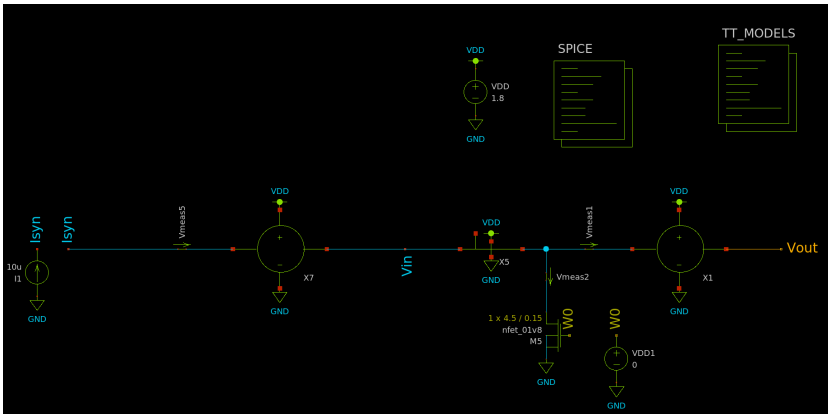


Figure 44: Test Harness Configuration. Input Current of 10uA is put into the first neuron to provide a constant spiking behaviour. The output from the next neuron is measured to evaluate how much the weight effects spiking frequency.

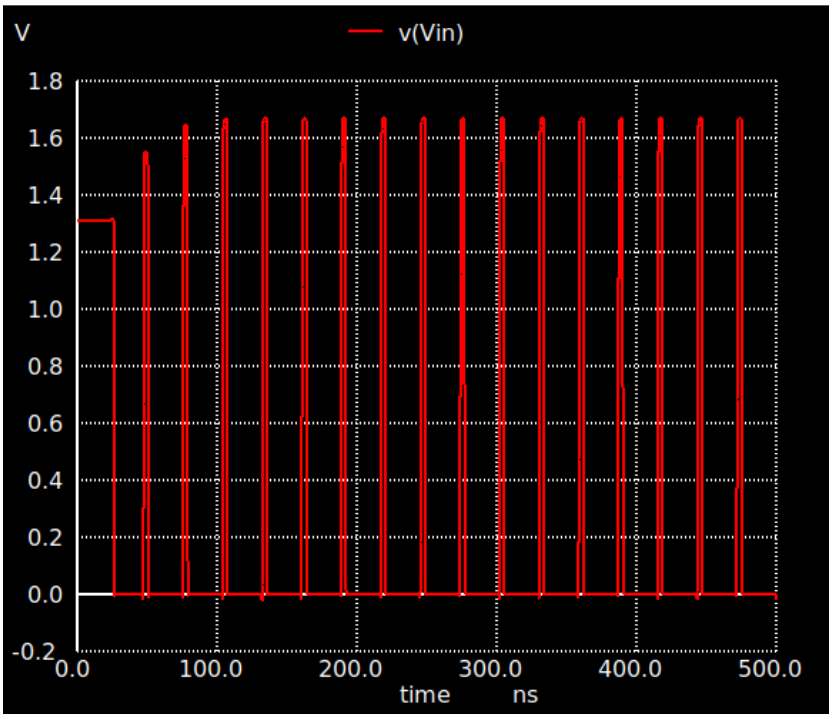


Figure 45: Input into the synapse from the neuron for every test. The spiking behaviour is very consistent and is the same across simulations.

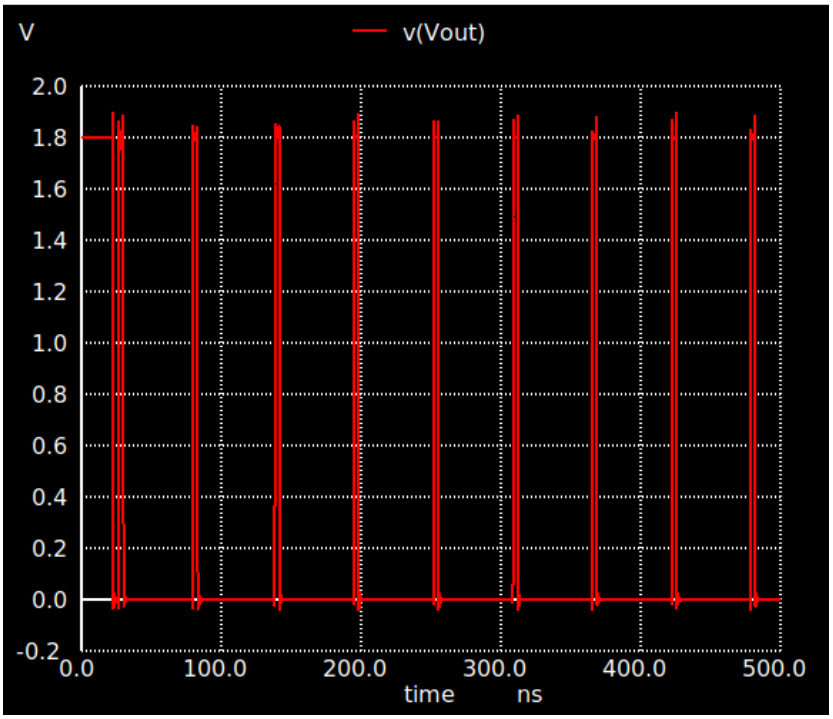


Figure 46: Output of the last neuron when the weight on the synapse is 0. The spiking output has a very frequency.

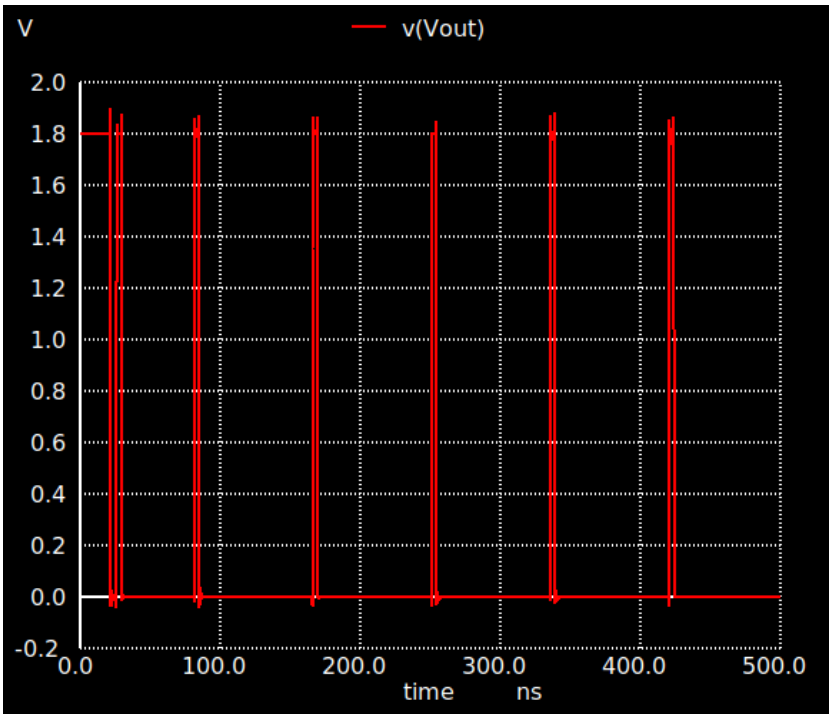


Figure 47: Output of the last neuron when the weight on the synapse is 0.6. The spiking output has a lower frequency.

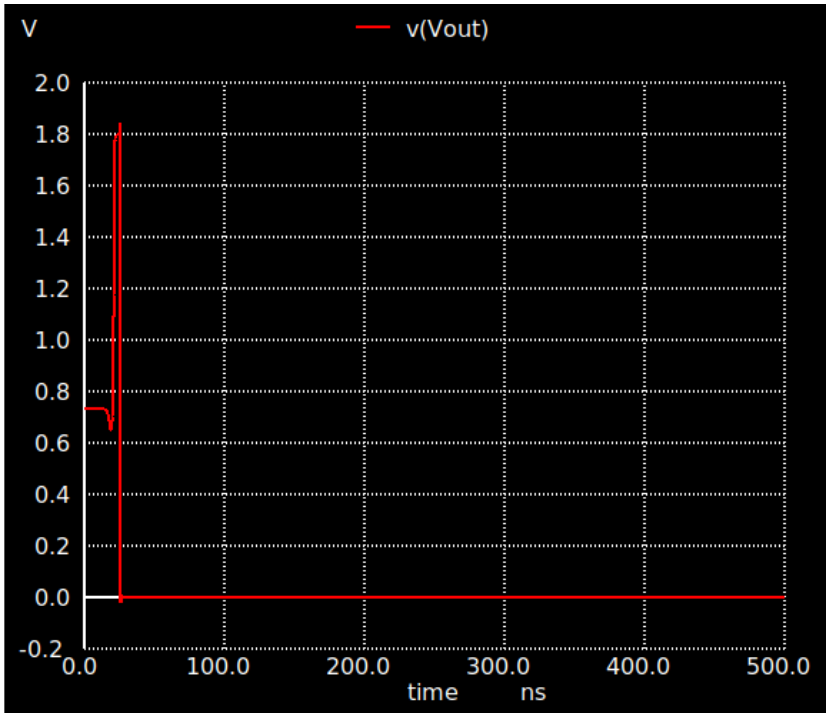


Figure 48: Output of the last neuron when the weight on the synapse is 1.8. The spiking behaviour of the neuron is fully inhibited and has a frequency of 0.

This synapse has all the properties needed to be used in a spiking neural network. By being able to adjust the output frequency of a neuron given a weight on the synapse, means that this structure can be used to create a neural network. The structure is not just limited to connected only 1 synapse per neuron and can be used to create more complicated neural networks.

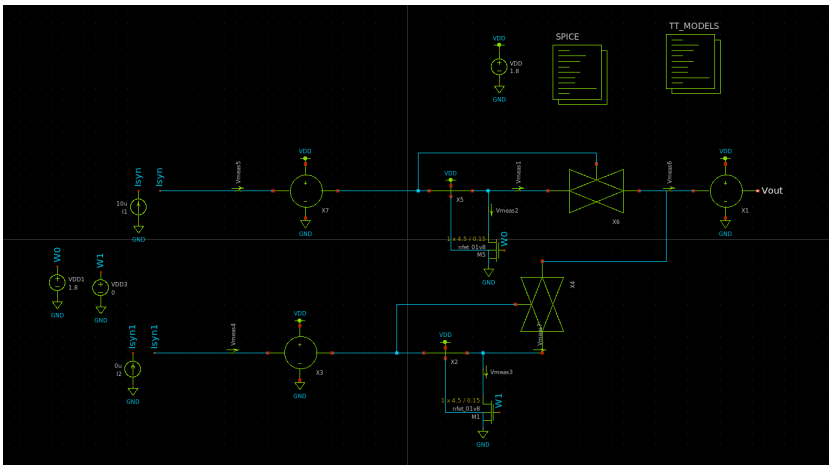


Figure 49: 2 synapses connected to a neuron. For clarity, the transmission gate, 1T1M transistor and memristor are shown, but is the final synapse design shown in the previous section. Both have the same input spiking. Any number of synapses can be connected to the neuron.

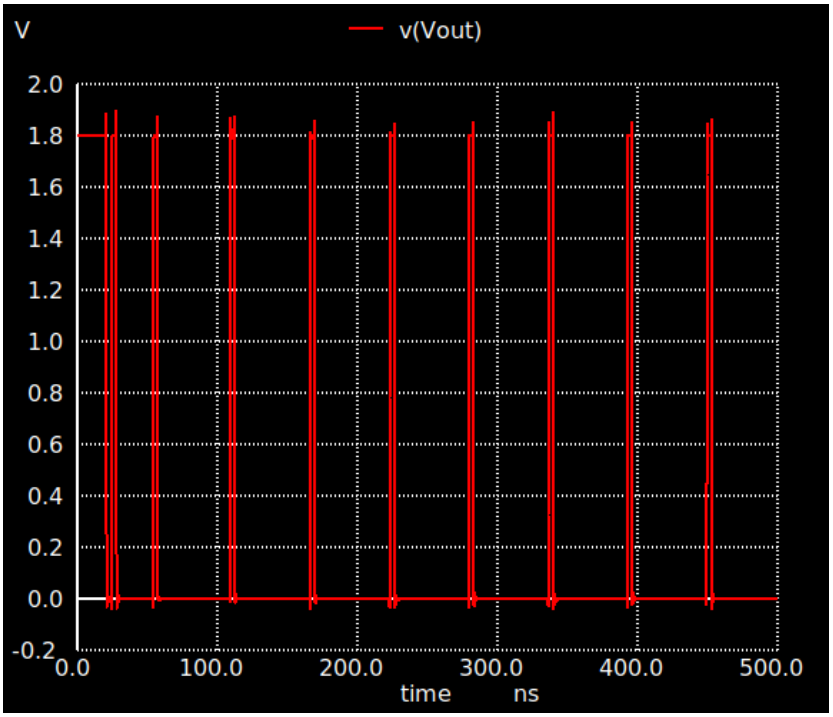


Figure 50: Output of the last neuron when both weights are 0.6. The output spiking is faster than when there is a single synapse of weight 0.6 connected to the neuron.

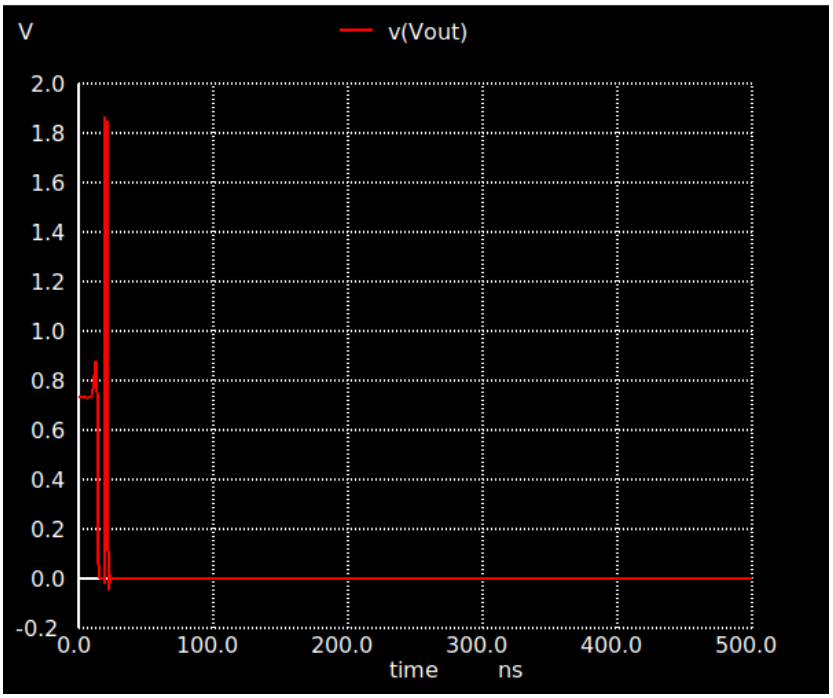


Figure 51: Output of the last neuron when W0 is 1.8 and W1 is 0. The output spikes are all attenuated. This shows the behaviour that synapses can be used to inhibit other synapses, modeling the idea of a negative weight in a neural network.

The ability of the synapses to both inhibit and excite neurons, representing the negative and positive weights that are in neural networks, means that these components can be used to build any neural network with reasonable accuracy, given an appropriate training circuit. This circuit can be

used with nearly any analog Spiking Timing Dependent Plasticity circuit as there are is a wide range of Voltage weights that can be decremented and incremented based on specific conditions.

3.4.1 Layout

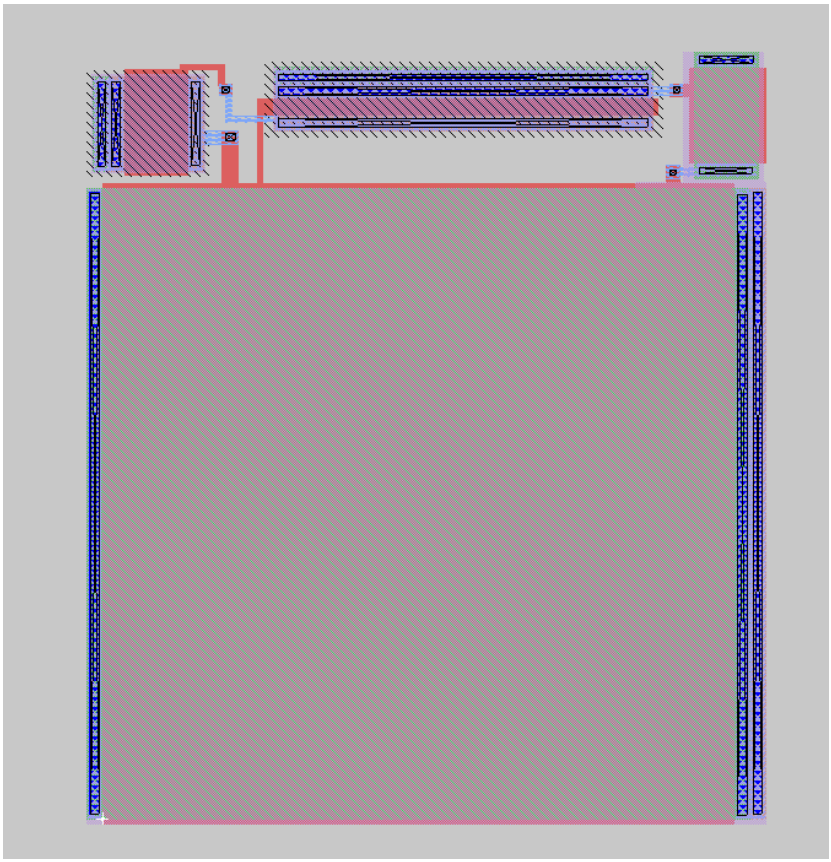


Figure 52: Layout of Memristor. It is sized at 21.500um x 24.440um.

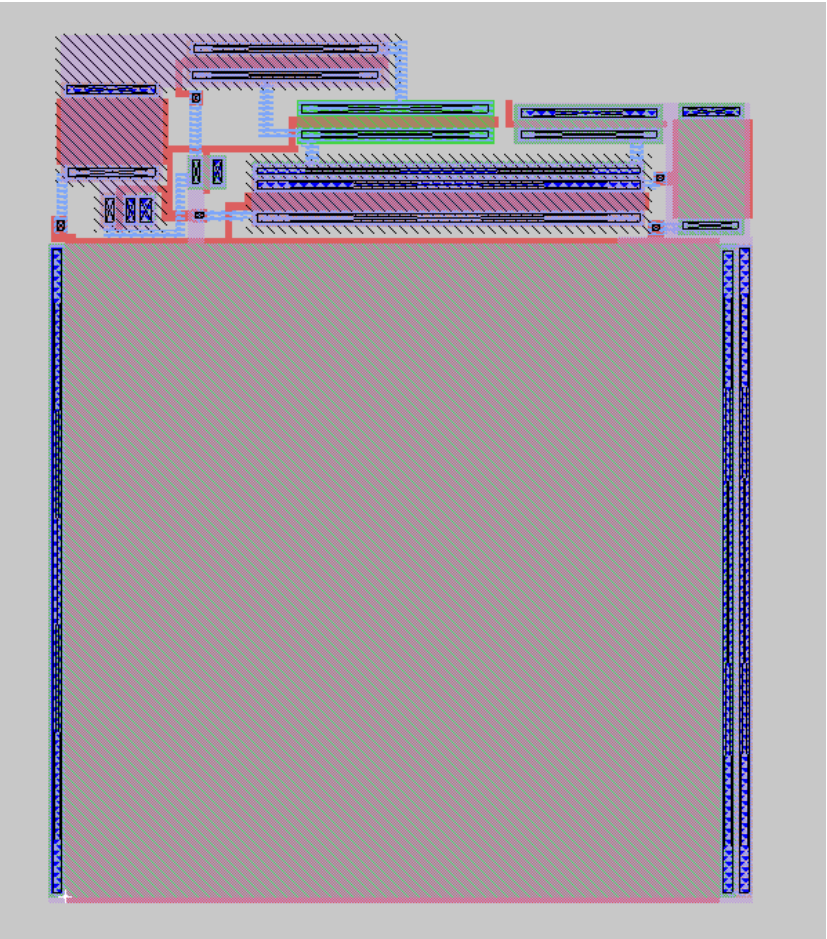


Figure 53: Layout of Memristor with 1T1M and Transmission Gate. It is sized at 21.500um x 26.560um.

3.4.2 Post-Layout Simulation

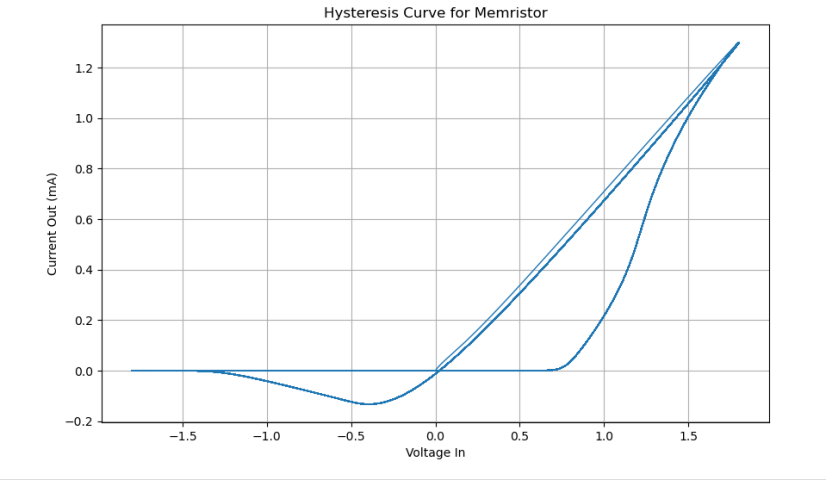


Figure 54: Hysteresis curve of layout at 4MHz. The results show nearly identical behaviour to the schematic.

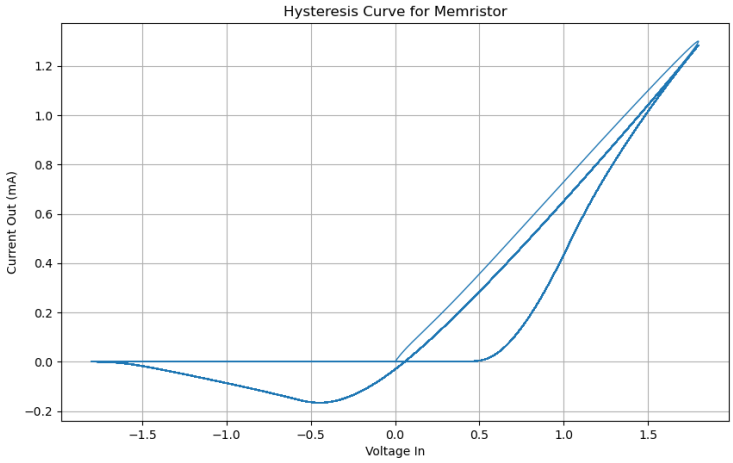


Figure 55: Hysteresis curve of layout at 9MHz. The results show nearly identical behaviour to the schematic.

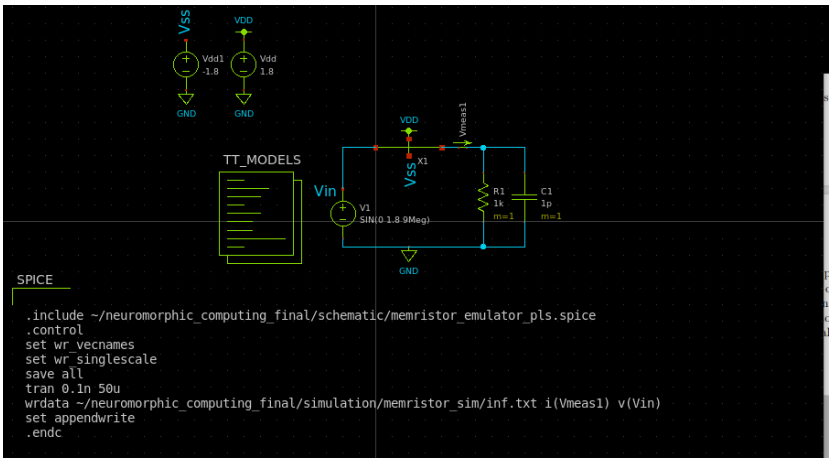


Figure 56: Test Harness for Hysteresis curve

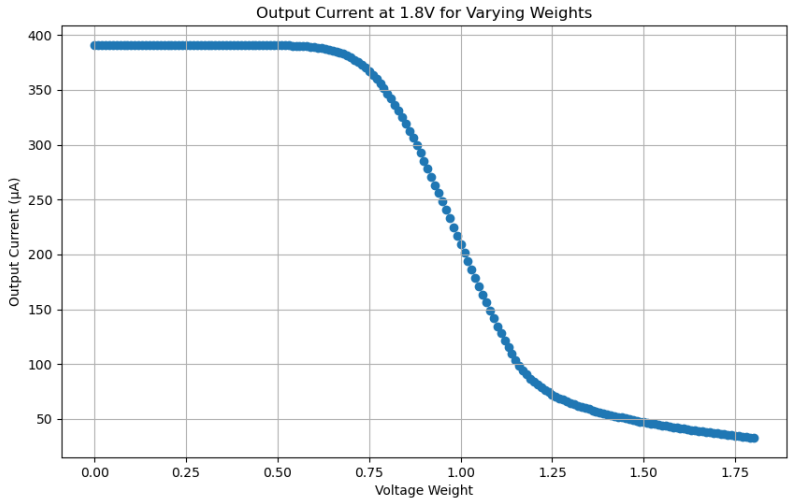


Figure 57: Post Layout Simulation of Weight Value and output current for the Memristor in the Resistor Configurtion. The behaviour is nearly identical to the schematic with almost the same current levels.

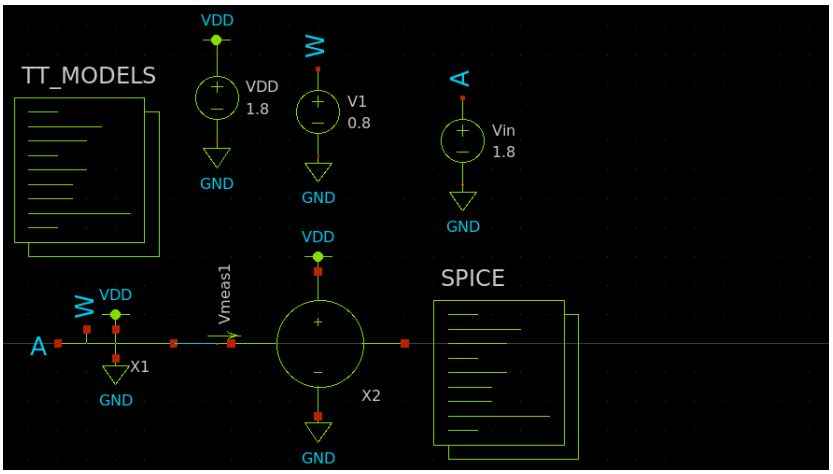


Figure 58: Test Harness used for current output vs weight voltage.

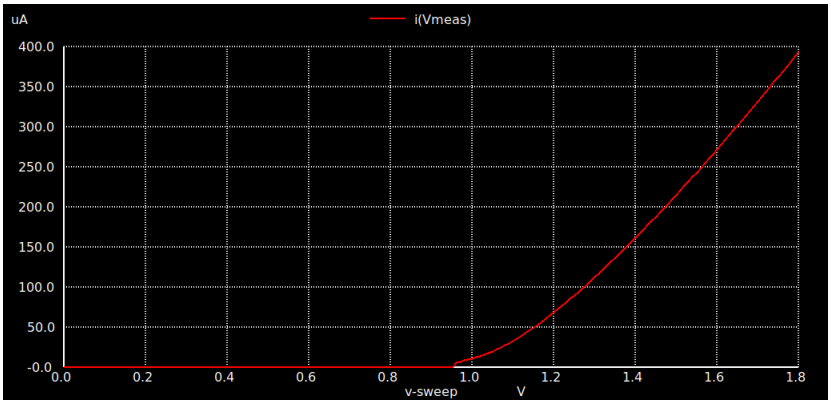


Figure 59: Current Voltage characteristic of Memristor

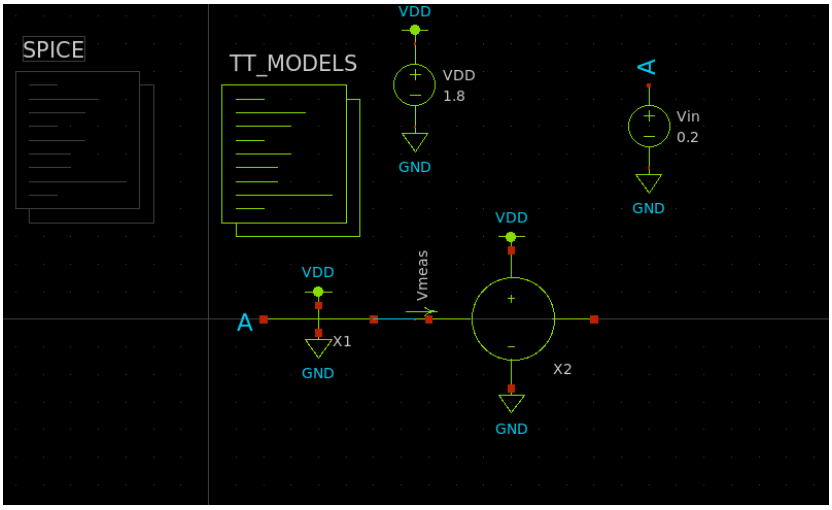


Figure 60: Test Harness Current Voltage characteristic of Memristor

3.4.3 LVS

Memristor

```
Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.

Subcircuit summary:
Circuit 1: mem_lvs.spice                   |Circuit 2: memristor_emulator.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__pfet_01v8 (2)                |sky130_fd_pr__pfet_01v8 (2)
sky130_fd_pr__nfet_01v8 (2)                |sky130_fd_pr__nfet_01v8 (2)
Number of devices: 4                       |Number of devices: 4
Number of nets: 5                          |Number of nets: 5
-----------------------------------------------------------------------------------------
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes mem_lvs.spice and memristor_emulator.spice are equivalent.

Final result: Circuits match uniquely.
.
```

Memristor in Resistor Configuration

```
Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are equivalent.

Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are equivalent.
Flattening unmatched subcell memristor_emulator in circuit res.spice (1)(1 instance)
Flattening unmatched subcell tg in circuit res.spice (1)(1 instance)
Flattening unmatched subcell inverter in circuit res.spice (1)(1 instance)
```

```
Subcircuit summary:
Circuit 1: memristor_emulator_res.spice    |Circuit 2: res.spice
-------------------------------------------|-------------------------------------------
sky130_fd_pr__nfet_01v8 (5)                |sky130_fd_pr__nfet_01v8 (5)
sky130_fd_pr__pfet_01v8 (4)                |sky130_fd_pr__pfet_01v8 (4)
Number of devices: 9                       |Number of devices: 9
Number of nets: 8                          |Number of nets: 8
-----------------------------------------------------------------------------------------
Netlists match uniquely.
Cells have no pins;  pin matching not needed.
Device classes memristor_emulator_res.spice and res.spice are equivalent.

Final result: Circuits match uniquely.
.
```

4 Works Cited

1. M. Besrour et al., "Analog Spiking Neuron in 28 nm CMOS," 2022 20th IEEE Interregional NEWCAS Conference (NEWCAS), Quebec City, QC, Canada, 2022, pp. 148-152, doi: 10.1109/NEWCAS52662.2022.9842088.

2. Abd, Hamam and König, Andreas. "A Compact Four Transistor CMOS-Design of a Floating Memristor for Adaptive Spiking Neural Networks and Corresponding Self-X Sensor Electronics to Industry 4.0" tm - Technisches Messen, vol. 87, no. s1, 2020, pp. s91-s96. https://doi.org/10.1515/teme-2020-0024

3.A. Adeyemo, X. Yang, A. Bala and A. Jabir, "Analytic models for crossbar write operation," 2016 Sixth International Symposium on Embedded Computing and System Design (ISED), Patna, India, 2016, pp. 313-317, doi: 10.1109/ISED.2016.7977104.

4. K. S. Swaroop and G. Ahmad, "Design of CMOS Memristor Emulator and Logic Gates For Neuromorphic Circuits," 2023 International Symposium on Devices, Circuits and Systems (ISDCS), Higashihiroshima, Japan, 2023, pp. 1-6, doi: 10.1109/ISDCS58735.2023.10153559.

5. Asghar MS, Arslan S, Al-Hamid AA, Kim H. A Compact and Low-Power SoC Design for Spiking Neural Network Based on Current Multiplier Charge Injector Synapse. Sensors (Basel). 2023 Jul 10;23(14):6275. doi: 10.3390/s23146275. PMID: 37514571; PMCID: PMC10383375.

5 GitHub Repository

https://github.com/raiyansiddique/neuromorphic_computing_final