Design and Simulation of Low Power Time-Domain Current-Mode Analogue Neural Network architecture in 180nm CMOS

Wes Gaunt

January 2020

1 Introduction

A low power time-domain neural network is of interest to various researchers in Dr Pantelis Georgiou's group, and further afield. In [5], a pH sensor with a pulse length is presented, with pulse length ranging from $2.9\mu S$ to $550nS$. In a later paper by the same researchers [6], 64 of the pH-to-PWM pixels presented in [5] are incorporated into a single integrated circuit. As these devices become larger, the power budget required to transmit data for processing will scale accordingly. On chip analysis by a neural network will help to reduce the communication power requirements and could help make rapid, on-body disease diagnosis a reality.

This project will involve the design and simulation of Time-Domain Analogue Neural Network in CMOS. This builds on previous work carried out by researchers in Dr Georgiou's group at Imperial College London in collaboration with ARM Research.

The orginal work was simulated using a flexible Pseudo-CMOS technology which had several limitation. This project aims to first replicate the original work described in [2] in a conventional 180nm CMOS technology, exploring its performance and limitations, and describing any tradeoffs with regard to the performance of the architecture. More generally, the project will aim to determine if an analogue neural network is a viable option low power option for interfacing to low power sensors and what performance gains they can offer over purely digital implementations.

Contents

2 Artificial Neural Networks and Neurons

There is currently a great deal of interest in machine learning, a technique used to identify patterns in data previously reserved for humans. One bio-inspired approach to ML uses artificial neural networks, which are systems inspired by human brains, designed to mimic the way that humans and animals think. Up until this point, the systems that build up neural networks have been digital computers, either programmed in software, synthesised using programmable logic or fabricated in an ASIC. One negative aspect of these approaches, when compared with the biological equivalent (a brain) is their high power consumption.

A lower power solution can utilise Analogue computing techniques to build neural networks. Whilst there is a inherent accuracy/dynamic range limit of analogue techniques, next-generation bio-chips which produce large amounts of data [5] could make use of on-chip processing for power efficiency.

2.1 Neural Network Introduction

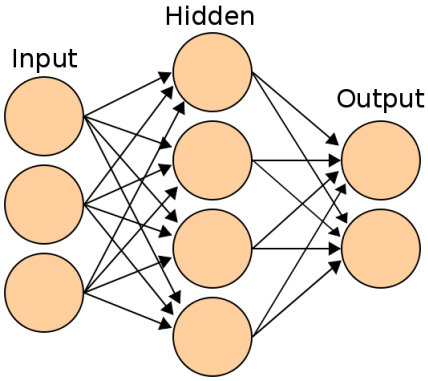

Figure 1: A simple neural network

A neural network is a network of nodes, known as neurons (see Chapter 1 of [4]), which are designed to mimic biological neurons. Each neuron can take one or more inputs, which are multiplied by a weighting factor, w. The output is the weighted sum of all inputs, with the weight being an inherent property of the neuron. Multiple neurons are arranged in 'layers', as in fig.1. Each layer (there are often multiple hidden layers) feeds its output signal to the next layer. There can be any number of intermediate layers between the input and output layer, 1 is shown for simplicity in the figure above.

Between these layers, there is another function, called an activation function - which maps the output of the neuron to another value (normally -1 to 1 or 0 to 1). Non-linear activation functions are generally preferred, although non linear activation methods to implement this has not yet been investigated, several ideas are presented in the next steps part of the paper.

Using this model, each neuron can be mathematically modelled as a *Multiply*

Accumulate operation, shown in fig.2, and described in the following equation.

$$y = \mathbf{XW} + b \tag{1}$$

Where $\mathbf{X}$ and $\mathbf{W}$ are vectors in the form:

$$\mathbf{X} = \begin{Bmatrix} x_n & \cdots & x_0 \end{Bmatrix}, \mathbf{W} = \begin{Bmatrix} w_n \\ \vdots \\ w_0 \end{Bmatrix} \tag{2}$$

Where n is the number of inputs.

The multiply accumulate operation is therefore:

$$y = \mathbf{XW} + b \tag{3}$$

$$y = \sum_{i=0}^{n} x_i w_i + b \tag{4}$$

The MAC operation is depicted below

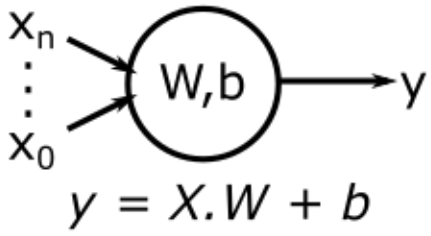

Figure 2: An artificial neuron, modeled as a MAC

The first steps in the project are to build towards a single MAC - recreating the architecture described in [2], which will accept positive and negative weights and inputs and will be easily scaleable in number of inputs and dynamic range.

3 Time domain input MAC design using current sources and capacitors

The previous work described in [2], described constructing a Neural Network using analogue techniques, which operated in the *time domain*, with magnitudes represented by pulse length. This is very similar to the *spike domain* that biological neurons utilise. This system was being designed for fabrication on a flexible substrate, which is produced by PragmatIC, a startup in Cambridge that promises to deliver flexible integrated circuits which are $< 20\mu m$ thick [7]. One drawback of this technology is that only NMOS devices can be fabricated, and with no P-MOSFETs available, buffers and p-type current mirrors cannot be created. Instead of CMOS logic, pseudo-CMOS logic was used which had several drawbacks.

3.1 The multiplier: Current-Voltage relationship of the capacitor

The architecture relied on the current-voltage-charge relationships of the capacitor:

$$I(t) = \frac{dQ(t)}{dT} \tag{5}$$

$$V(t) = \frac{Q(t)}{C} \tag{6}$$

This leads to the following relationship:

$$I(t) = C\frac{dV(t)}{dt} \tag{7}$$

This implies that a constant current applied to a capacitor would cause it to charge or discharge at a constant rate, with a constant rate of voltage change (slew). A higher current causes a higher slew rate.

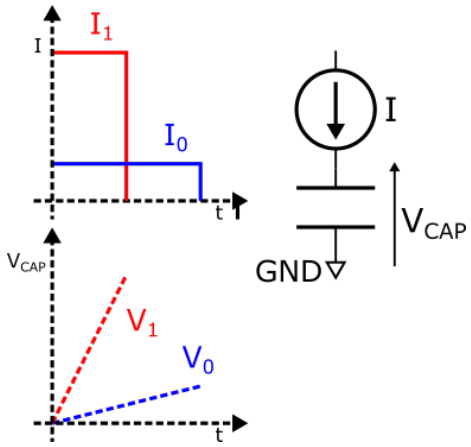

Figure 3: A higher current leads to a faster slew for the same capacitance

This is shown in the diagram above. The voltage change of the capacitor is a linear function of both current and time, as shown by (7).

$$I(t) = C\frac{dV(t)}{dt} = C\frac{\Delta V}{\Delta t} \tag{8}$$

$$\Delta V = \frac{1}{C} \cdot I \cdot \Delta t \tag{9}$$

Both the current and the time can be controlled, with the capacitor voltage proportional to their product. This is the basis of the analogue multiplier architecture.

3.2 The accumulator: Kirchoff's Current Law or Theory of Current Superposition

Kirchoff's Current Law states that the total current coming out of a node will be equal to the total current going into a node. This is shown diagrammatically below. In the context of charging capacitors, this is more easily visualised in

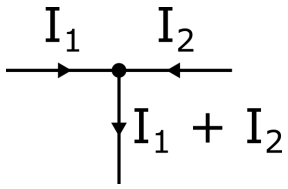

Figure 4: Kirchoff's Current Law

terms of charges delivered, due to the current-charge relationship in (5).

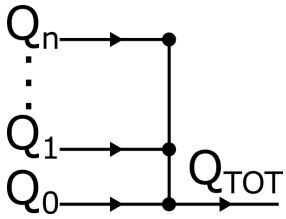

Figure 5: Charge In = Charge Out

The situation in fig.5 will only occur at non-capacitive nodes. Where a node has a high capacitance and impedance (high τ), charge will accumulate accordingly:

$$Q_{TOT} = \sum_{i=0}^{n} Q_i \tag{10}$$

This is the basis of the accumulator with charge stored on the plate of a capacitor.

3.3 MAC architecture

Both of these techniques can be used together to form an analogue Multiply Accumulate function. Both the current I, and the time t can be controlled, from (5) and (10)

$$Q(w, x) = I(w) \cdot t(x) \tag{11}$$

$$Q_{TOT} = \sum_{i=0}^{n} I(w_i) \cdot t(x_i) \tag{12}$$

This is an analogue of the equation (4) of a MAC, with the multiplicands x and w encoded in pulse duration and current magnitude respectively.

The voltage at the end of a set duration will be proportional to the product of each individual current and duration. The simplified schematic in fig.6 shows

the MAC architecture that of a single artificial neuron, with n inputs. As both W and b characterise the neuron, it can be easier to think of the b input as a weight of b with a unity input. The W vector term in red is property of the neuron and X (blue) is the input vector. In this manner, the voltage across the capacitor will be proportional to the dot product of the two terms.

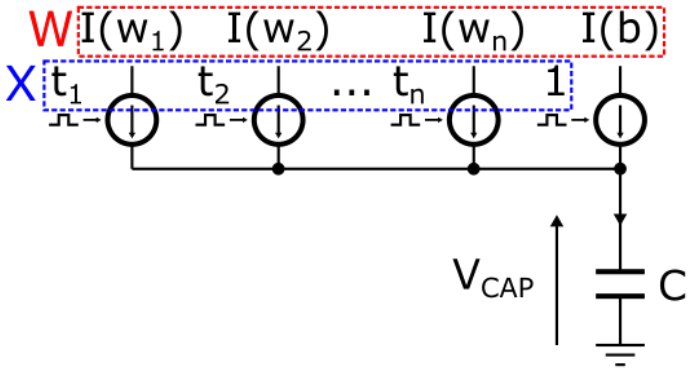

Figure 6: A simplified analogue, time domain artificial neuron

$$\mathbf{X} = \begin{Bmatrix} x_n & \cdots & x_0 & 1 \end{Bmatrix}, \mathbf{W} = \begin{Bmatrix} w_n \\ \vdots \\ w_0 \\ b \end{Bmatrix} \tag{13}$$

The subsequent steps now involve building up a current source architecture with a programmable magnitude (W) and time domain (pulse length) input (X) - which will make up the multiplier. Multiple multiplier blocks can be utilised in parallel to add or subtract charge from a capacitor, forming the accumulator. All values of x, w and b will be bounded between ± 1.

4 Binary weighted current mirrors

The current source will be split into two complementary systems, allowing bidirectional current flow which will allow the voltage on a capacitor plate to be both raised and lowered by adding and subtracting charge. The current sink will be made using NMOS, whilst the source will be made using PMOS.

4.1 NMOS current sink

The first step is creating the NMOS mirror. This takes a reference current, I_{IN} and multiplies it by the 4-bit binary word $D[3:0]$, which gives a range of 0-15. The multiplication is carried out by increasing the width of the mirroring transistors. This mirror is made up of NMOS devices, it can only 'sink' current. This will, when coupled with a capacitor, pull the capacitor plate voltage towards VSS (0V).

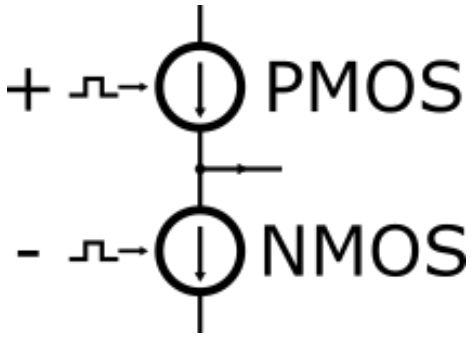

Figure 7: CMOS current sink/source

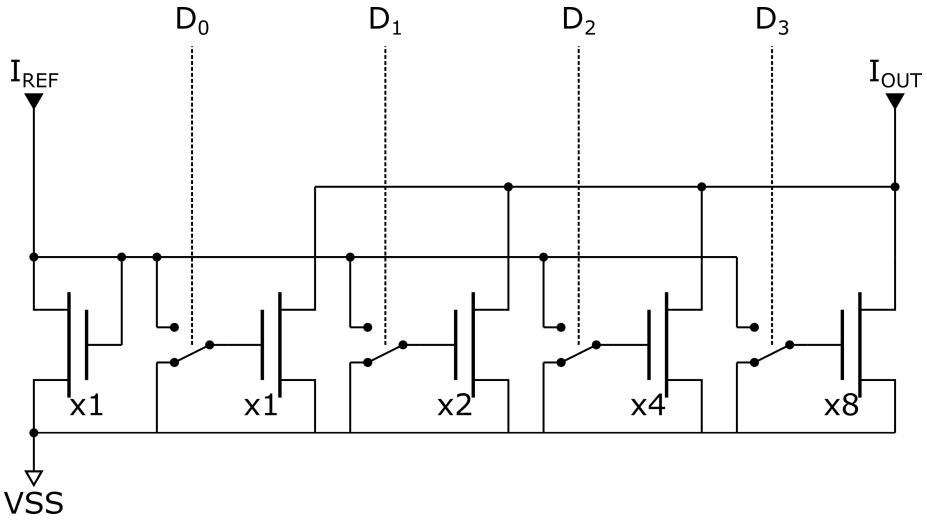

Figure 8: The first design of the NMOS (current sink) mirror

The higher the multiplicative factor D, the higher the voltage slew-rate will be.

This design was created within the *Cadence Virtuoso* environment, with subsequent transient simulations carried out.

Fig.10 shows one of the first tests, with the predicted slew-rate in orange, and simulation results in blue.

The initial results in 10 show a non-linear output current relationship, and under closer inspection it is revealed that the current is not starting to flow immediately after activation, but after a (0.1μs) onset delay, as shown in fig.11. This leads to lower than expected average slew at low currents.

Increasing the current and capacitance by 2 orders of magnitude causes this delay to disappear, and running the same set of transient tests produces a more linear current source. This is good for linearity, however it is not good for power consumption (which is proportional to current). With higher currents, larger capacitors will have to be used for charge accumulation with correspondingly larger areas.

The low currents and lack of linearity could however be utilised as a non

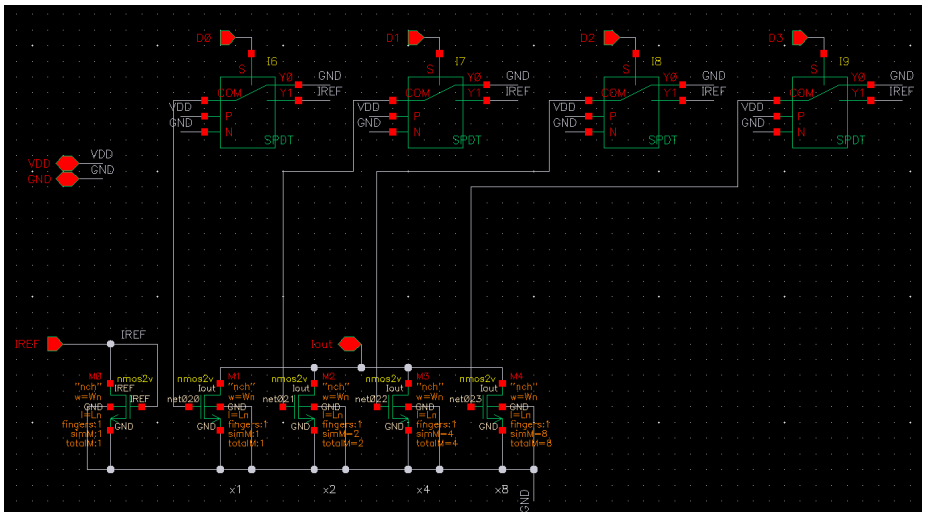

Figure 9: Cadence screenshot of the NMOS mirror

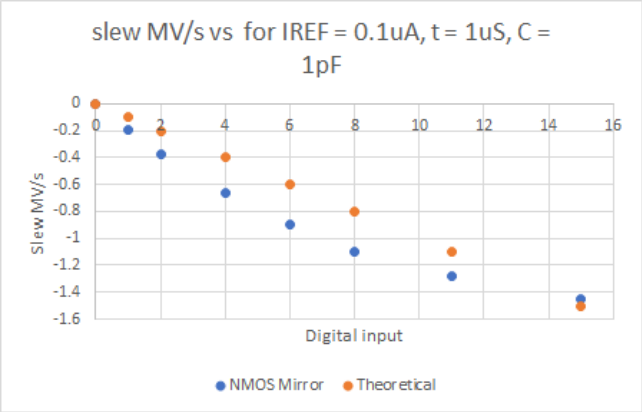

Figure 10: The slew rate of the NMOS current mirror vs input code, D[3:0] for a 0.1μA reference current

linear activation function (see next steps section) using the machine learning techniques documented in [4].

Various currents were tested via a parametric sweep bwtween 1μA and 20μA with the NMOS current mirror, and 5μA was found to provide a good balance of capacitor size and linearity.

4.2 PMOS current source

The PMOS source is the complementary of the NMOS sink. Functionality is identical with current sourced from VDD (1.8V).

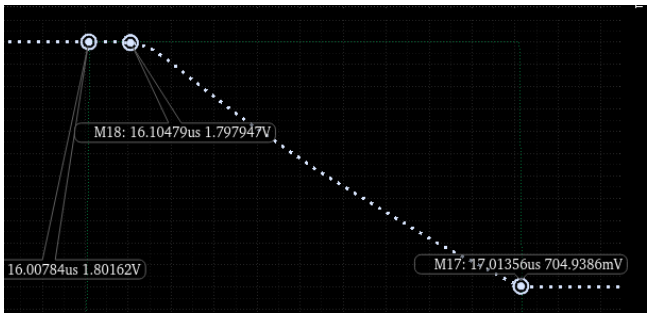

Figure 11: This trace shows a capacitor being charged with a constant current source, a 0.1 μS onset delay is shown

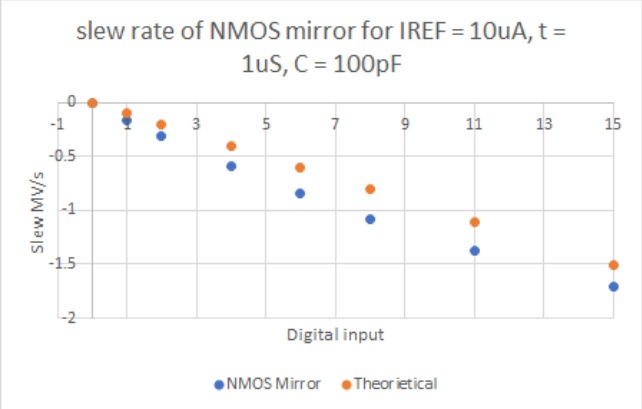

Figure 12: The slew rate of the NMOS current mirror vs input code, D[3:0] for a 10μA reference current

5 Analogue Multiply Accumulate Neurons

5.1 Source/Sink multiplier block

Both binary-weighted mirrors can be used in conjunction to form a current source that can be controlled with the pulse length of either IN_P or IN_N (the sign of IN is determined by which line is toggled) and W. This is shown in the diagram below. This single current source uses 16 CMOS switches in total; an improved version which removes some redundant switches is presented in the next steps/improvements section of this report.

This was implemented and tested using *Virtuoso* and it worked as expected.

5.2 Verilog-A control blocks

To test the design, two testbenches (for IN and W) were written using *Verilog-A*, which read data from .csv files and encoded it correctly. The input was encoded in the time domain, where IN 0 - 1 was encoded as a $0\mu S$ - $1\mu S$ pulse on either the postive IN_P output, or the negative IN_N output. The weighting, W and b, was encoded in binary with where the magnitude 0 - 1 was mapped

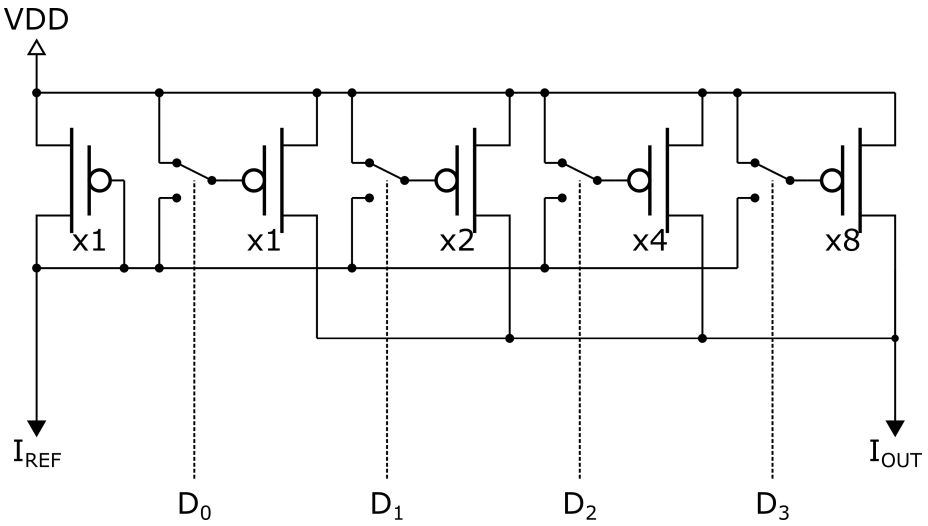

Figure 13: The first design of the PMOS (current source) mirror

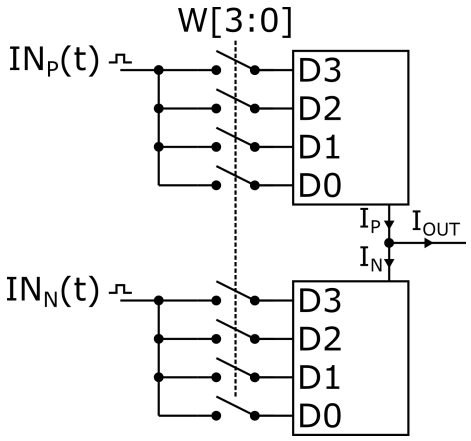

Figure 14: The source/sink multiplier block

to 0-15 in binary, with a sign bit indicating negative values (signed magnitude representation).

6 Testing methodology and initial results

Two of the multiplier blocks shown in fig.14 were used in parallel to create a *2-input neuron*, (no b input yet) as described in fig.7. This was tested with a pre-prepared data-set. An output capacitor was first charged to a mid rail virtual ground of $\frac{VDD}{2} = 0.9V$, with charge subsequently added or subtracted by the mirrors. The voltage on the capacitor plate was measured after the maximum time step ($1\mu S$), and the results exported to MATLAB where the y output of the model was compared with the voltage change. The first results are shown in fig.16 below, with different weight pairings indicated in the legend.

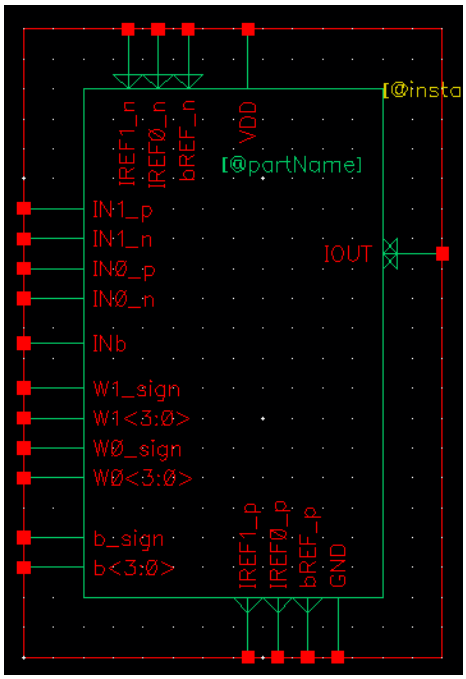

Figure 15: The 2 input neuron cell in cadence

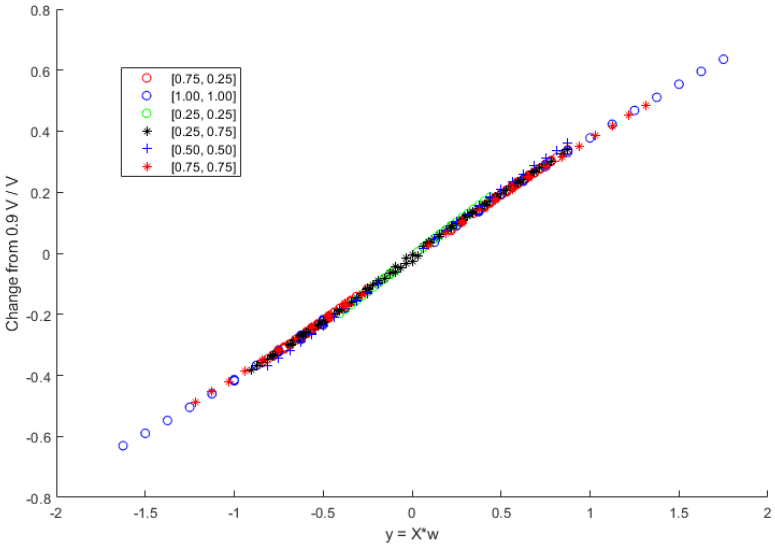

Figure 16: The first results of the bidirectional current source, the ΔV is shown on the y-axis, whilst the ground truth generated from MATLAB is shown on the x-axis. The legend indicates different weight pairs.

The model used to test the neuron was the simple function with a binary output, this is known as the activation function:

$$y = \begin{cases} 1, & XW \geq 0 \\ 0, & XW < 0 \end{cases} \tag{14}$$

The neuron scored 100% in the first test, matching the model and scoring the same as the MATLAB code.

Test	**Inputs**	**b?**	**Model**	**Analogue design**
1	2	N	100	$200/200 = 1$
2	2	Y	100	$191/200 = 0.955$
3	skipped			
4	3	Y	0.895	$179/200 = 0.895$

Table 1: Results for the first three tests

7 Time domain output

The output of the neuron in is still voltage encoded, with magnitude proportional to the output value. To feed forward to a neuron in a subsequent layer requires the output encoded as a pulse width proportional to the value. This is a similar principle of operation of a dual-slope ADC, which is described in Chapter 29 of [1]. After the maximum time-step, the accumulation capacitor will have reached its final voltage, either above or below the virtual ground of 0.9V. The next phase, the discharge phase, can subsequently be performed by a constant current source which will result in a linear voltage slew as shown in fig.3. With a comparator and the appropriate logic, a pulse with duration proportional to magnitude can be produced.

A comprehensive digital controller is shown in fig.17, this has been created in CMOS and simulated with a testbench, shown in fig.18.

This method of time domain output works very well, and produces the correct polarity of pulses with length proportional to output value. The results are shown in fig.19.

8 Reducing the number of switches

In fig.8, the current sink design was present which utilised four switches to scale the current from 0-15 of I_{REF}. The mirror also requires 4 more switches which are required to route the time-domain pulse, resulting in 2 switches for each bit in the weight word. A second design has been designed, but not yet tested, which is shown in fig.9.

This reduces the number of switches by half, requiring only $n + 1$ switches for an n bit mirror. This same approach can be applied to the complementary PMOS mirror.

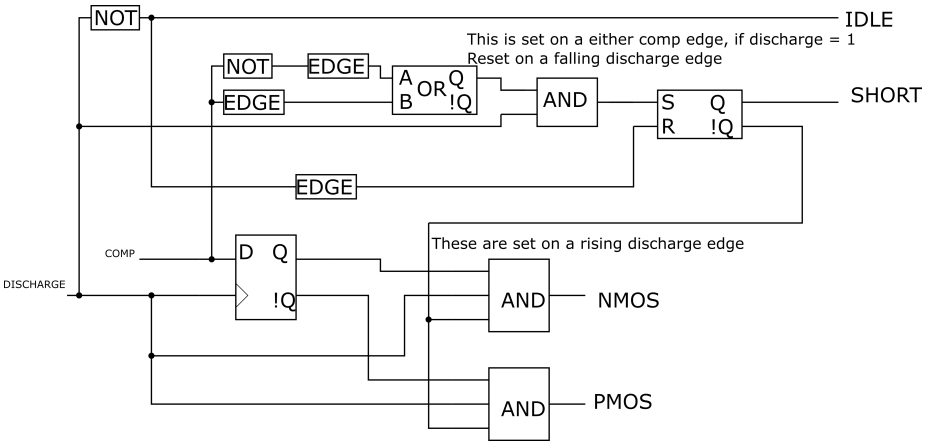

Figure 17: The time-domain output finite state machine that can control the capacitor discharge and produce a pulse of the correct sign with width proportional to voltage magnitude on the capacitor

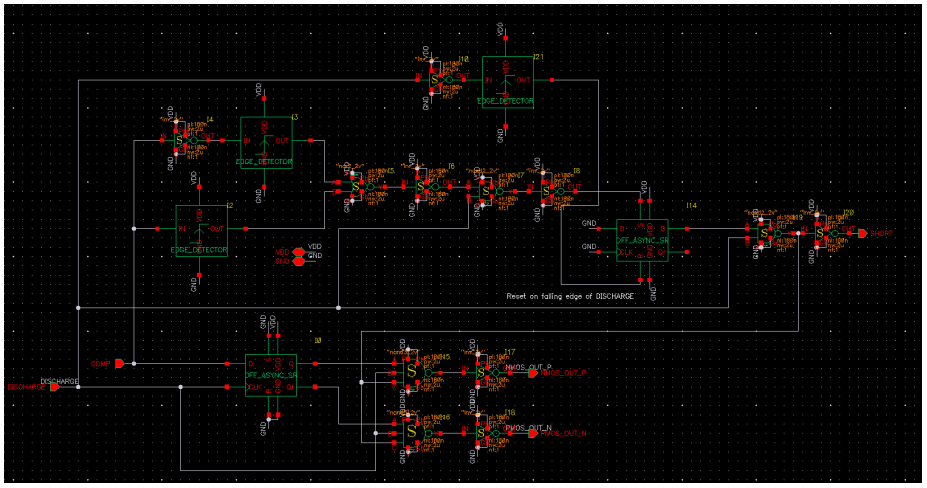

Figure 18: The Time-Domain output Finite State machine in cadence, this has some extra circuitry in order to reduce undesired states

9 Next steps: Towards a multi-layer neural network

The neuron presented in this report is an integral first step in creating a neural network. However, a multiple layer network time domain neural network has still not been created, using multiple neurons. To reach this stage, several hurdles have to be overcome, including multiple layers, time domain outputs and activation functions. If time allows, project scope could also expand to include time domain input weightings, using a time-to-current topology for the current references.

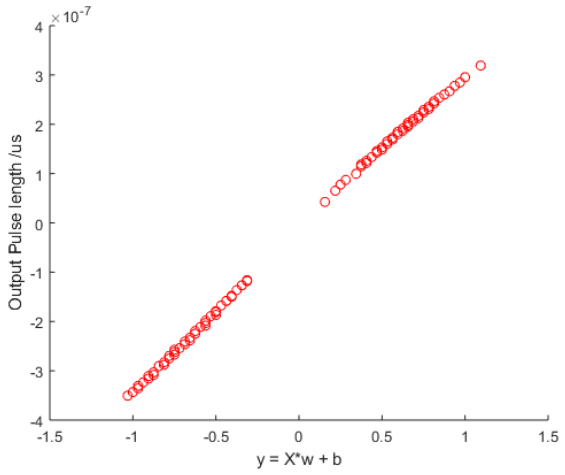

Figure 19: The results from the Time-Domain output circuit. Pulses on the OUT_P signal are unchanged, while pulses on the OUT_N signal have been negated for ease of plotting

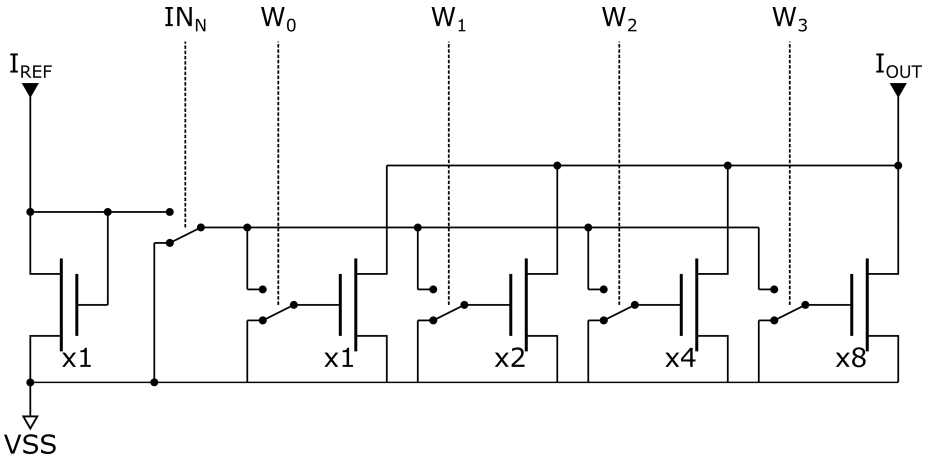

Figure 20: The switch reduced binary weighted NMOS current mirror

9.1 Non-linear activation functions

Another feature of neural networks is the non-linear activation functions that and map neuron outputs then to the next input. In the most basic implementation, this fires the neuron if above some threshold, which is a step function described in equation (14).

More sophisticated activation functions map a neuron y-value non-linearly to the output, as shown by the sigmoid function in fig.22:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{15}$$

The current neuron implementation outputs a voltage proportional to the

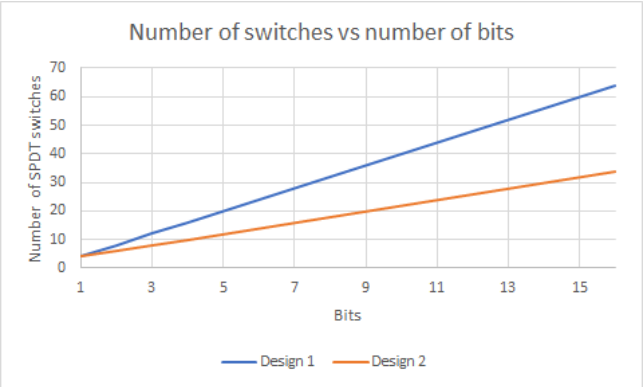

Figure 21: Removing redundant switches in both the NMOS and PMOS mirrors offers an area and power saving

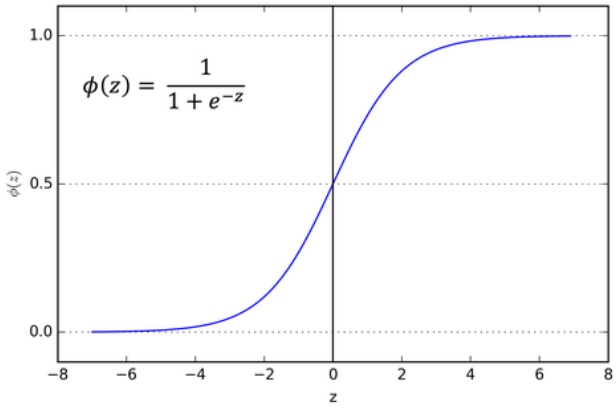

Figure 22: A Sigmoid function is a very popular choice of non-linear activation function

output value, as shown in fig.16. Any output function will have to map this voltage to a time domain pulse (see previous section).

Various analogue techniques provide non linearity *de-facto*, and whilst this is normally undesirable can be used to in this context to an advantage, and could actually allow an improvement over a digital implementation to closer resemble biological neural network.

9.1.1 Asymmetric discharge current mirrors

The time domain output circuit shown in fig.17 is an example of a linear output function, as shown in fig.19 it maps the output value as a pulse length, with the negative values weighted identically to the positive values. However

One non linear activation function is the leaky rectified linear function, shown in fig.23. This scales the negative values by a much smaller amount, ϵ, whilst mapping any positive values without modification. A normal value for $\epsilon = 0.01$.

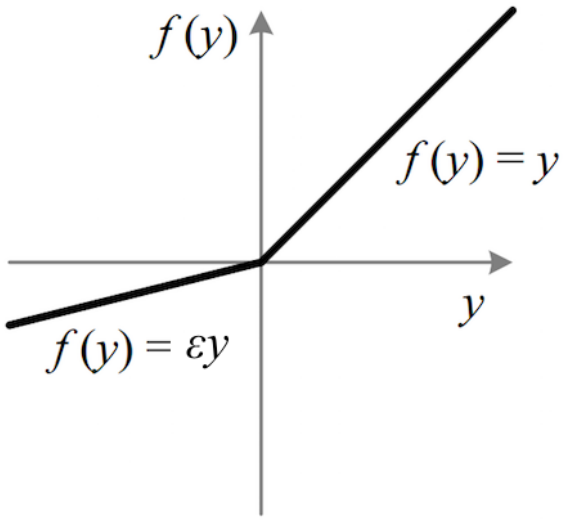

Figure 23: The leaky rectified linear unit function

This scaling can easily be accomplished by discharging the accumulation capacitor with a much greater $(1/\epsilon)$ PMOS current than that of the complementary NMOS current. This results in much shorter negative pulse lengths than the postive pulse lengths, and creates the leaky rectified linear function. This is shown in fig.24; with the PMOS current 100x that of the NMOS current ($\epsilon = 0.01$).

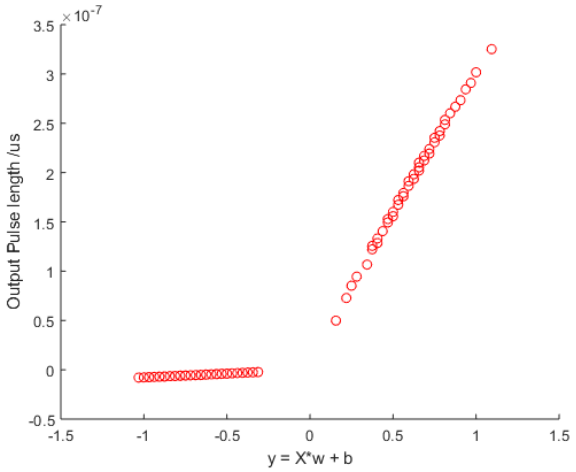

Figure 24: The time domain output with leaky modification, $\epsilon = 0.01$

This method implementing a non-linear activation function is easy to modify, or tune, by changing the PMOS mirror current the gradient of the negative outputs can be changed. In fig.25, the current has been set to only 10x of the

NMOS current, and so pulse length is 1/10 of its original value.

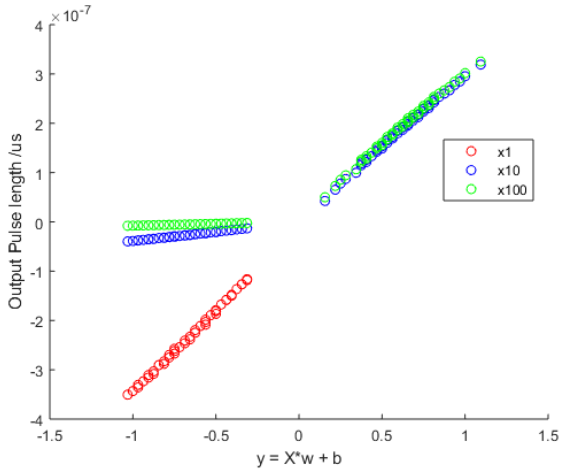

Figure 25: The time domain output with leaky modification, showing 3 different values of ϵ.

9.1.2 Current mirror onset delay

Another source of non linearity was presented by the current mirror neuron onset delay, which some literature suggests may be similar to the true characteristic of a neuron [8]. By reducing the reference current (I_{REF}), the delay and shape of the activation function can be tuned, and capacitor size can be reduced appropriately saving chip area. This deserves to be investigated further.

9.1.3 MOSFET operating point limit

Another source of non linearity is also due to a physical constraint of the MOSFET device. When the voltage difference between the drain and source terminals V_{DS} drops too low, it leaves saturation mode and the drain current drops. In the case of the current-mode MAC, this leads to a lower than desired slew rate on the capacitor plate. This could lead to tailing off which and may will resemble the sigmoid function in fig.22. However, this is not as tunable as the current onset delay, and so may be harder to obtain a specific desired activation function.

9.1.4 Accumulation capacitor saturation

Another method which is proposed is using the finite accumulation capacitance as a built in piecewise-linear activation function. This form of function is a popular choice, with the *Rectified Linear Unit* (ReLU) function being one of the most popular choices. As the accumulation capacitance decreases, the speed of charge accumulation (slew) increases for a constant current. This results in saturation as the capacitor cannot go beyond the voltage rails, shown in fig.26 below.

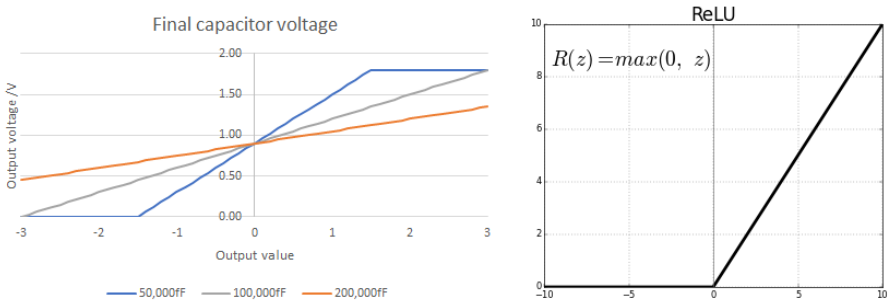

Figure 26: The capacitor provides a simple piecewise-linear activation function which has similarities to the ReLU function

This method offers the advantage of being easily tunable, either in manufacturing or by using a voltage dependant MOS capacitor which is a physical property of the MOSFET device. This could offer great advantages for a *learning* network - and will be further investigated.

9.2 Time domain weighting

The neuron architecture still requires the weight to be binary encoded. For simplicity, using a time domain approach to weighting could be accomplished using the following circuit, which builds on the principles outlined in this report. In this design, an input capacitor is charged using a constant current source,

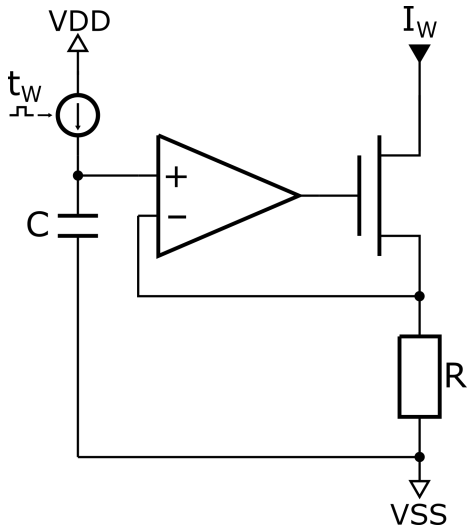

Figure 27: A simplified time-to-current circuit

with its voltage proportional to the pulse length t_W. This is followed by a transconductance amplifier with a gain of $\frac{1}{R}$. However this uses an 2 operational amplifiers (a further gain stage is needed internally) and has a relatively high power consumption.

10 Conclusion

This report has described the process of designing a time domain current-mode analogue neural network in 180nm CMOS. Building up from current mirrors to a functioning neuron with 3 inputs. The next steps will be cascading multiple neurons in order to realise a simple neural network.

11 Safety, Ethical and Legal considerations

11.1 Safety

The project will be carried out purely on a PC at Imperial College London, therefore there will be no added health or safety issues over that of normal office work.

11.2 Ethical

Machine Learning and artificial intelligence is subject to an intense debate regarding the abuse of Artificial Intelligence, with many business leaders urging caution [3]. In [9], the author describes how data privacy concerns should be addressed and what a researchers concerns should be. However concerning, in this project, which is dealing with very small sensors - it is not forseen that there are ethical arguments to answer to.

11.3 Legal

There are currently no legal considerations relevant to the project; with a collaborative understanding between ARM Research and Imperial College London. This has the potential to change if the project is successful and goes further.

References

[1] R. Baker. *CMOS Circuit Design, Layout, and Simulation*. Aug. 1997. ISBN: 978-0780334168.

[2] Matt Douthwaite et al. "A Time-Domain Current-Mode MAC Engine for Analogue Neural Networks in Flexible Electronics". In: Oct. 2019, pp. 1–4. DOI: 10.1109/BIOCAS.2019.8919190.

[3] *Google boss Sundar Pichai calls for AI regulation*. Jan. 2020. URL: https://www.bbc.co.uk/news/technology-51178198.

[4] Kevin Gurney. *An Introduction to Neural Networks*. USA: Taylor & Francis, Inc., 1997. ISBN: 1857286731.

[5] Nicolas Moser, T.s Lande, and Pantelis Georgiou. "A novel pH-to-time ISFET pixel architecture with offset compensation". In: May 2015, pp. 481–484. DOI: 10.1109/ISCAS.2015.7168675.

[6] Nicolas Moser, T.s Lande, and Pantelis Georgiou. "A robust ISFET array with in-pixel quantisation and automatic offset calibration". In: Oct. 2016, pp. 50–53. DOI: 10.1109/BioCAS.2016.7833722.

[7] PragmatIC. *New Company Brochure*. URL: https://www.pragmatic.tech/assets/media/pragmatic-new-company-brochure.pdf.

[8] H. A. Swadlow and S. G. Waxman. "Axonal conduction delays". In: *Scholarpedia* 7.6 (2012). revision #125736, p. 1451. DOI: 10.4249/scholarpedia.1451.

[9] Apara Venkateswaran. *Ethics in Machine Learning*. June 2017. URL: https://towardsdatascience.com/ethics-in-machine-learning-9fa5b1aadc12.