# Comparative Study on Analog and Digital Neural Networks

**Vipan Kakkar**

SMVD University, India

**Summary**
For the last two decades, lot of research has been done on neural networks, resulting in many types of neural networks. These neural networks can be implemented in number of ways. Due to the revival of research interest in neural networks, some important technological developments have been made in VLSI. This paper discusses comparative study between analog implementation and digital implementation for neural networks. The discussion topics include power-consumption, area, robustness, and implementation efficiency of these implementation techniques respectively. It can be estimated whether an analog or digital neural network is optimum for a specific application. It is observed that the choice between analog and digital neural networks is application dependent. The goal is to estimate which type of implementation should be used for which class of applications. This work is based on the study of neural implementations restricted only to pattern classification and focuses on widely used layered feed-forward neural network.

***Key words:***
*Neural Networks, Digital design, analog design, VLSI, power consumption, robustness, efficiency, applications.*

## 1. Introduction

NN (Neural network) is basically a system that performs a function roughly divided into pattern classification and function approximation. In other words neural network is another way of performing a function. A direct implementation of an existing algorithm outperforms a trained neural network in the field of performance. However, a neural network implementation may be smaller and/or faster than implementation of the exact algorithm. Though it is clear that neural network should not be used for a simple function for which an algorithm is already known, these networks can be valuable for complex and difficult algorithmic functions. This work is based on the VLSI design of neural networks for classification functions. Neural networks can be implemented in large number of ways as seen in Figure 1.

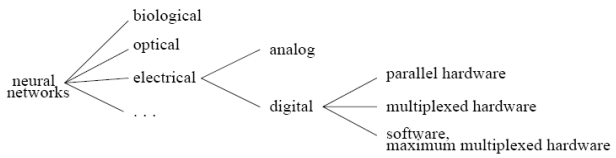


**Figure 1: Neural Network Implementation Techniques**

The scope of this paper is electrical neural implementation. As an example, signal processing in neural networks is in principle in parallel, therefore, it is simply decided to implement neural networks in analog hardware or in parallel digital hardware. Software implementation can be flexible, but slower and not area as well as power efficient. Software implementation can be area and power efficient if the operating speed required for a function is low. This paper discusses the comparison between analog, digital, and mixed (i.e. analog and digital) implementation of neural networks in terms of power consumption, area and robustness. This is explained in the following sections.

### 1.1 Neural Networks: General Description

Neural networks typically consist of large number of simple processing units, called neurons. Each neuron has a multi-dimensional input signal, the input vector and produces one signal output signal $Y$ as shown in Fig. 2, where is the weight vector. The output signal $Y$ is typically a non-linear, saturating function.

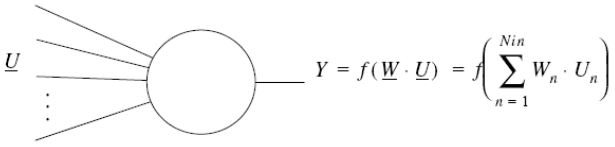


**Figure 2: Single neuron of a neural network**

A neural network is made up of a number of artificial neurons and a huge number of interconnections between them. Fig. 3 shows a typical flow-diagram of feed-forward neural network [12].

Analog VLSI implementation of artificial neural networks represents one of approaches to enhance the computational capabilities in real-time information processing. Character recognition, retrieval of data/image from fragments, pattern recognition and speech synthesis are some applications of artificial neural networks. These neural networks consist of massive parallel layers of neurons interconnected with synapses as shown in Figure 3. The main function of the synapse cell is to achieve linear multiplication of input and a weight. These synaptic connections are implemented using Analog multipliers. Applications like multi layer feed forward networks

Manuscript received July 5, 2009
Manuscript revised July 20, 2009

require large number of interconnected neurons and synaptic connections (multipliers). Therefore careful design of multiplier is crucial in achieving compact silicon area, minimizing power consumption and improving input range.

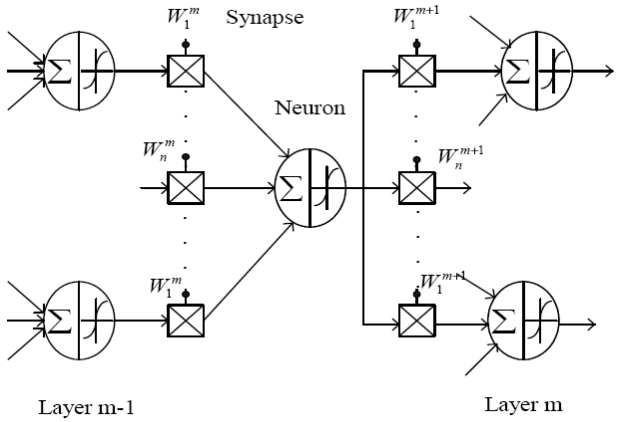


**Figure 3: Flow graph of Neural Network [2]**

Over the past decade a huge diversity of hardware for ANNs has been designed. An investigation on neural network hardware devices proposed in the literature reveals that a functional block-level representation as shown in Fig. 4 is suitable for describing almost all neural network architecture [17]. The activation block, which evaluates the weighted sum of the inputs is always on the neuro-chip. Other blocks, i.e. the Neuron State Block, Weights Block, and the Transfer Function Block may be on the chip, or off the chip. A host computer may perform some of these functions and computations.

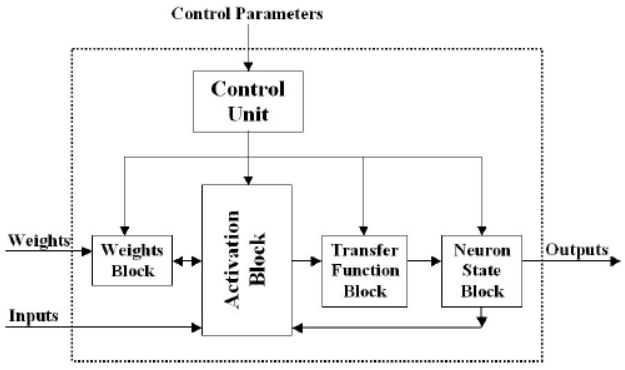


**Figure 4: Block level structure of Neural Network**

A lot of research on analog hardware neural networks is based on statements that can be found in neural network literature. The two most important ones are listed below:

- the limited accuracy of analog components [for neural networks] is not a serious problem because neural networks are forgiving to component errors [25]
- for neural networks, the need for precision and for large Signal-to-Noise-Ratio is replaced by that for real-time collective processing [24]

These statements imply that neural networks can be built using inaccurate analog hardware, where the inaccuracy includes mismatch, noise, drift etc. The first statement is correct as far as relatively small static errors are concerned. However, non-static errors over the entire operating range are not compensated. Therefore, this statement is in general wrong. The second statement is stronger than the first one, and will also be shown to be incorrect in this paper.

### 1.2 Implementation of Learning

Learning implies that neural network is capable of changing its input/output behavior as a result of changes in the environment. Learning may be implemented in analog, digital or software. It can be calculated [9] that the resolution demand for weight adaptations during training is very high. Typical resolution figures for digital weight resolution during learning (training) are in the range of 12-18 bits. For analog weight adaptation, the magnitude of the systematic errors in the weight adaptations is about the analog equivalent of the 12-18 bit digital resolution. These high resolution figures severely complicate implementing on-chip weight-adaptation circuits for neural networks. However, some possibly feasible alternative algorithms for on-chip analog learning are proposed [23].

To set the proper weights for *not*-learning analog neural network chips, two methods are generally applied: *off-chip learning*, and *with-chip learning*. With *off-chip learning*, the weights are typically calculated on a computer that runs a model of the neural network. After this offline learning, the weights are down-loaded to the neural network chips. *With-chip learning* incorporates the hardware neural network *in* the training and therefore *ideally* compensates both for static-errors in the hardware neural network and compensates for static mismatch between the simulated model and the actual hardware realisation. However, most errors are non-static over the entire operating range (including temperature range).

## 2. Comparison: Power Consumption

### 2.1 Analog Neural Network

It is well known from literature [1] that the minimum power consumption in *analog* hardware is completely determined by the achieved accuracy and operating

frequency. Usually, the accuracy of an analog circuit is expressed in terms of Signal-to-Noise-Ratio (SNR), which gives the ratio between signal power and noise-power. In actual circuits, the inaccuracy in the SNR is the sum of all noise, all mismatch, all drifts etc. In this paper, only thermal noise is assumed to be present.

The linearity, supply voltage, power dissipation and noise are the main metrics of performance [3]. We try to design some specific structures or topologies for the analog multiplier that have low power dissipation while at the same time keeping good linearity, low supply voltage and low noise.

### 2.2 Digital Neural Network

For *digital* signal processing, the power consumption also depends on SNR requirements. The number of bits necessary to construct a certain bipolar range with accuracy SNR requires. Note that for digital neural networks, the inaccuracy is half the LSB-size.

Neural networks require large numbers of multiplications, which may be multiplexed onto a few physical digital multipliers. In the extremes, one may either use $N$ multipliers or one may multiplex the operations by a factor $N$ on one single digital multiplier (as in software). The multiplexing has no effect on the summed power required for the multiplications (assuming negligibly small power required for the multiplexing itself).

### 2.3 Analog vs Digital Neural Network

In this work, it was observed that the power consumption in feed-forward neural networks is a function of:

- the number of input signals
- the number of first-layer neurons
- the required resolution in input space for the classification tasks to be performed

The neural network can be implemented in a few ways, depending on the domain of both the inputs and of the signal processing. For maximum power-efficiency:

- *digital signal processing on digital signals* must be used if the input signals are available in the digital domain only
- *digital signal processing (incl. ADC) on analog signals* must be used if both the number of first-layer neurons is over about 10 (dependent on the efficiency of analog, ADC and digital hardware) depending upon SNR (Signal-to-Noise Ratio)
- *analog neural network* is to be preferred for maximum power efficiency for all other cases.

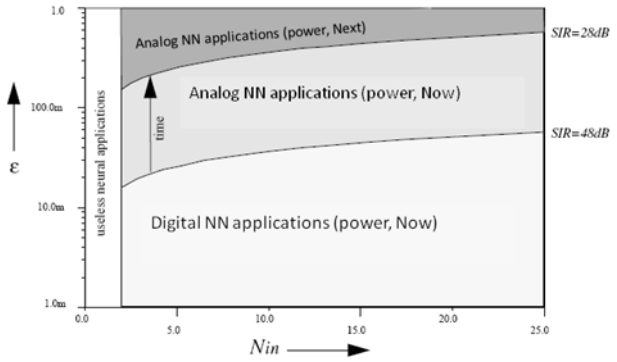


**Figure 5: Application for Analog NN in terms of ϵ and $N_{In}$**

This work limits the class of useful applications for fully analog neural networks. Potentially useful applications for analog neural networks (from only a power consumption point of view) can hence be represented as shown in Fig. 5. These estimations were derived for a special-purpose analog neural network for simple training sets. The class of useful applications for general purpose analog neural networks is a subclass of the one shown in Fig. 5.

The applications corresponding to the white area in Fig. 5 should either not be done using neural networks or should be implemented in digital hardware (with or without AD-conversion of input signals).

It follows from the (rough) estimations that the *power efficiency of digital neural networks including ADC with few first-layer neurons* is generally *only somewhat lower* than the power efficiency of fully analog neural networks. This is due to the fact that the analog part (the ADC) of this NN limits the power-efficiency.

## 3. Comparison: Area

In this section, some estimations and comparison on required chip area for fully analog implementations, digital realisations with AD-conversion for input signals, and for fully digital implementations of feed-forward neural networks are presented. In the estimations, only the first layer (and input-signal conversion where necessary) is taken into account because for the optimized neural network system, the second layer is identical.

### 3.1 Multiplier Size

In this sub-section, estimations on multiplier size in analog and digital hardware are given.

### 3.1.1 Analog Multiplier

Fig. 6 gives some performance metrics of an analog multiplier. Neural network contain many multipliers, which consume large part of power dissipation.

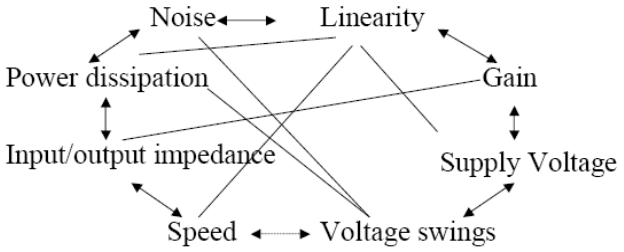


**Figure 6: Metrics of Analog Multiplier**

Within the "Analog Neural Hardware" project a modified MOS resistive analog multiplier was proposed for usage in neural nets. For good matching of the four MOS transistors over the total operating range (including temperature range) and to prevent short-channel effects, the length and width of the transistors cannot be minimum. The MOS transistors are in fact floating-gate MOSTs (MOS Transistors), which are also used to store the weights. A certain minimum floating-gate capacitance is required in order to be able to accurately store and load the charge.

It appears from simulations and calculations that if the W/L is ½ in the 130 nm process, the demands on matching, on noise, and minimum capacitance of the floating gate can be satisfied.

Accuracy measurements on the multiplier cell are still to be performed. Note that circuitry to program the floating-gate charge also takes a vast amount of chip area (on-chip charge-pumps or high-voltage selection transistors). Literature reports the following sizes, types and accuracy for neural multipliers:

**Table 1: Accuracy and area figures for analog multipliers**

| Multiplier | Equivalent Accuracy | Area ($um^2$) | Author |
|---|---|---|---|
| MDAC | 5 bits | 18,000 | Coggins et al [14] |
| Differential pair | 5-8 bits | 3,600 | Lont et al [18] |
| Gilbert multiplier | 8 bits | 4,500 | Shima et al [19] |

### 3.1.2 Digital Multiplier

According to a report on VHDL-synthesized digital multipliers in 130-nm technology, the area clearly is bigger than analog multiplier.

The output of a NxM bits multiplication is (N+M) bits; if less output bits are required, the chip area decreases considerably.

### 3.1.3 Analog vs Digital Area for the Multiplier

Direct comparison of required area for analog and digital multipliers is clearly in favour of analog circuits. However, direct comparison cannot be done because the function of the digital multiplier may be multiplexed: one digital multiplier may be used to mimic N-multipliers.

The required chip-area for digital must therefore be divided by the multiplexing-factor, which in turn depends on the ratio between maximum operating frequency of the multiplier and the operating frequency of multipliers without multiplexing.

## 3.2 Adder Size

### 3.2.1 Analog Adder

In neural network with an analog first layer, the addition of the output signals of the multipliers (to obtain the weight input of the neuron) can be done by simply interconnecting the multiplier outputs. The addition of the individual output currents is then conform the Kirchhoff current law and uses almost no chip area:

$$A_{adder} \approx 0 \quad (1)$$

Note that the speed of the addition is determined by the capacitance at the output node and the output-resistance of all multipliers connected to the output node.

### 3.2.2 Digital Adder

For neural networks, the result of multiplications is summed to obtain a weighted input signal. To implement this, one may use separate multipliers (which include adders) and a separate adder, or one may use a Multiply ACcumulate block (MAC). In the MAC the addition block is shared by the multiplier and the adder. The MAC requires about 50% more chip area than a multiplier, while an N-bits adder has about the same size as an NxN bit multiplier.

### 3.3 Analog vs Digital Area for the Adder

As a rough approximation, the size of an analog neural network is completely determined by the network structure; there is no dependency on the operating frequency and no multiplexing can be done. For digital neural networks, various functions may be multiplexed. The multiplexing rate is dependent on the ratio between maximum operating frequency of the digital circuits and the required operating frequency of the neural network. This means that the size of a digital neural network depends among others on the operating frequency.

The total required chip area for a neural network with inputs, first-layer neurons is then approximately given by:

$$A_{total} = N_{neuron1} \cdot N_{in} \cdot \frac{Area}{multiplication}(freq) + overhead \quad (2)$$

where the overhead consists of ADCs in case of an analog-digital neural network; the number of ADCs also depends on the operating frequency of the neural network and on the maximum conversion frequency of the ADC.

In the estimations for the size of a digital neuron, it is assumed that a MAC block (Multiply and ACcumulate) was used. It follows that for high operating frequencies analog neural networks occupy less chip area than their digital counterparts. With every new process generation, the break-even frequency shifts towards higher frequencies because the area of digital systems scale down, whereas the area required for analog circuits hardly scale.

## 4. Comparison: Weights

Categorized by storage types, there are five kinds of synapse circuits [5]: capacitor only [10]–[11], capacitor with refreshment [13]–[20], capacitor with EEPROM [4], digital [15], [16], and mixed D/A [17] circuits.

Capacitor weights are compact, but they have leakage problems and large in size to prevent unwanted weight-decay. Capacitor weights with refreshment can solve leakage problem, but they need off chip memory.

EEPROM weights are compact nonvolatile memories (permanent storage), but they are process sensitive and hard to program. Digital weights are usually large, requiring around 16-bit precision to implement learning. The mixed D/A weight storage [17] is a balanced solution when permanent storage is necessary. It will be shown that to set an analog weight, much more chip area is required than for setting a binary value in an EEPROM. Furthermore, the robustness of analog EEPROMs is far worse than the robustness of binary EEPROMs. Although normal EEPROMs are assumed, the same results hold for flash EEPROMs.

#### 4.1.1 Charge-trapping in analog EEPROMs [8]

Programming an analog value in an EEPROM cell requires that the floating-gate charge be set accurately. During programming, the floating-gate charge value can be set accurately incorporating the Vt change due to charge-trapping. However, after programming a part of the oxide-trapped charges is de-trapped which generally results in a net change of the transconductance of the EEPROM. Clearly, the carefully set function of the EEPROM drifts away.

In neural networks using EEPROMs to store analog weights, the inaccuracy margins ultimately determine the sensitivity for these initial weight changes. Therefore, the initial weight-change decreases the inaccuracy-margin available for other types of inaccuracies.

This implies that due to charge-trapping power consumption increases.

#### 4.1.2 Charge-trapping in digital EEPROMs

If EEPROMs are used to store binary values, the inaccuracy margin for each EEPROM cell is about 50% of the maximum programmed floating-gate charge. This margin is usually large enough not to notice de-trapping of oxide-charges.

Both for analog and for digital neural networks, weight are needed. This section discusses some basic problems that occur when storing weights in EEPROM structures. After this discussion, weight-set-circuits for on-chip setting of the weight in EEPROM are presented. It will be shown that to set an analog weight, much more chip area is required than for setting a binary value in an EEPROM. Furthermore, the robustness of analog EEPROMs is far worse than the robustness of binary EEPROMs. Although normal EEPROMs are assumed, the same results hold for flash EEPROMs.

### 4.2 Retention

#### 4.2.1 Digital retention

It is well known from literature that EEPROMs used in digital applications have a certain retention time. This retention-time is the time period in which the stored charge drops by 50%. For digital EEPROMs, this retention-time is about:

$$T_r \approx 30 \text{ years} \quad (3)$$

#### 4.2.2 Analog retention

Some estimations on analog retention time have been reported in literature. To estimate the retention time for any accuracy, we assume a linear decay of the stored charge. Because leakage through gate-oxide is strongly dependent on the field across the gate-oxide, it is fair to assume that the actual decay of stored charge is a function that at first is stronger than linear with time, and eventually saturating at some level. To estimate a best-case retention time for any accuracy, we assume a linear decay of the stored charge. Then, with every additional bit of accuracy, the retention time drops by about by a factor 2. 10 Years retention of 5-bits accuracy or the same figures have been reported [6], [7]. It follows that using these values

$$T_r \geq 10 \text{ year} \cdot \frac{\text{accuracy}}{2^{-5}} \text{ for accuracy} \geq 5\text{bits} \quad (4)$$

If lower accuracy is required, in number of bits, the greater-than-or-equal-to must be reversed. Using the digital retention time as a starting-point in the calculation of the analog retention- time, a more pessimistic analog retention-time results. Assuming a linear drop in stored charge at the floating-gate (a best-case situation):

$$T_r \geq 30 \text{ year} \cdot \frac{\text{accuracy}}{2} = 2 \text{ years} \cdot \frac{\text{accuracy}}{2^{-5}} \quad (5)$$

which estimation is significantly different from the previous one. Measurements are required to determine the actual retention-time for EEPROMs storing analog values.

### 4.3 Ease of programming

In fully analog neural networks, the analog weights are stored as analog charges on floating gates. To minimize the effects of for example mismatch and drift, feed-back is required for proper programming of the weights. This means that each individual EEPROM must incrementally be charged until the correct weight is set; this correct weight must somehow be measured during programming. It can be concluded that programming time is relatively long and that the circuit overhead is large for proper adjustment of the analog weights in analog neural nets.

In digital neural networks (with or without AD-conversion), the weights can be down-loaded directly. to the binary EEPROMs. Virtually no feed-back is required because the digital neural network does not suffer from effects such as mismatch and drift; only for effects such as window-closing [21] some simple verify scheme is required. Therefore, programming a digital neural network requires almost no extra circuits nor significant programming time.

### 4.4 On-chip weight-set circuits

EEPROMs can be programmed in two ways: the programming voltages can be applied externally or can be generated on-chip. In this section, some statements on on-chip weight-setting circuits for analog and digital storage in EEPROMs are given; an extensive discussion of circuitry is beyond the scope of this report.

#### 4.4.1 Digital weight loading circuit

In digital circuits, binary values may be stored on the floating-gate of an EEPROM-like device in a standard double-poly process. The on-chip weight setting circuit only has to make high-voltage pulses to tunnel through the gate-oxide. Although for reliability issues it is best to use a variable programming voltage, this appears not to be needed for neural networks because the EEPROMs are not likely to be programmed as many times as ordinary EEPROMs.

The programming circuit therefore only has to provide a high-enough voltage.

#### 4.4.2 Analog weight loading circuit

Programming *analog* weights at the floating gate of an EEPROM requires that the charge on the floating-gate be set accurately. This means that for accurately setting the floating-gatecharge, direct feed-back of this charge is required: the programming voltage must be pulsed. Because furthermore the variance on the tunnelling-voltage is not negligibly small, it may be clear that the programming voltage must be ramped.

### 4.5 Conclusions

Non-volatile storage of weight in neural networks requires some kind of E(E)PROM memory. These weight-values can both be stored as analog values and as digital words. Storing analog values on floating-gate devices has several problems concerning retention, robustness and difficult programming. Digital storage of words in relatively simple; it is inherently robust, retention does not seem to be a problem for neural applications, and programming is relatively simple.

For volatile storage of neural weights, storing analog weights is also much more difficult than storing binary values. Volatile storage of analog values required some kind of periodical refresh of the stored value, whereas digital volatile storage can be done in for example flipflops.

It is noted that then the stored value is destroyed only when the power-supply is turned off: no periodical refresh is therefore required.

## 5. Comparison: Other Parameters

In this chapter, some pros and cons of analog, digital and mixed AD neural networks that have not been discussed earlier in this report are discussed. The topics in this chapter include general pointers concerning robustness, scaling of circuits, and ease-of-design.

### 5.1 Robustness

In analog hardware, special precautions must be taken to minimize the effects of e.g. substrate-noise, power-supply variations, radiation, temperature variations, mobility reduction, matching, drift, leakage a.s.o. At the cost of increased chip-area and increased power consumption, analog hardware can be made *relatively* insensitive for these effects. Leakage and radiation effects in E(E)PROMS that store analog weights directly affect the classification performance of the neural network. Some EEPROM-specific issues were discussed in more detail in section 4.

Digital circuits are inherently very robust for the effects listed in the previous paragraph. As digital circuits have only two states, the inherent inaccuracy margin is about half the supply voltage. Moreover, effects of for example radiation and leakage are typically negligibly small for *digital* weight storage on EEPROMs.

### 5.2 Scaling properties with new processes

Digital circuits scale down very well with new processes, because digital down-scaling is the main drive for new processes. With the scaling down many issues in signal integrity, leakage and noise margin arises. With scaling of digital circuits, usually only limited design effort is required.

New processes are designed to optimize the implementation of digital circuits. This means that only the points that are important to minimize power and area consumption for digital circuits are optimized. Because of this, with new processes:

- the power supply voltage is decreased
- the performance of transistors is optimized for digital circuits

As new processes are designed to optimize the implementation of digital circuits, implementation of analog circuits in CMOS in future processes is not automatically improved. On the contrary, for example the lower supply voltage, the increased 1/f noise of transistors, the worsened transistor behavior etc. at least complicate the implementation of analog circuits in future processes. If analog neural networks can still be made in scaled-down processes, transferring an existing analog neural network circuit to newer processes requires complete redesigns.

## 6. Overall conclusions

It follows that for classification problems, digital neural networks (with or without AD-conversion of input signals) are to be preferred. Furthermore, a digital neural network (with or without ADC) always outperforms the analog neural network at a large number of other points:

- a digital neural network is inherently robust for effects such as drift, mismatch, noise, etc. An analog neural network is not inherently robust
- a digital network can be (almost automatically) generated from a logic description of its function; an analog neural network must be a full custom design
- loading digital weights is relatively easy, no feed-back is required; loading analog weights does require feed-back to compensate for example for mismatch, and suffers from effects such as leakage and charge-trapping in the oxide
- digital circuits scale very well with new processes, and virtually no redesign is required. Analog neural networks hardly scale, require a total redesign and, worse, may not be implementable in future processes

a digital neural network requires only standard inaccurate circuits; if A-to-D conversion is required, the only accurate circuit is this ADC which is nowadays a standard building block. An analog neural network requires that every sub-circuit satisfies specific accuracy demands, which are non-minimum.

Further, it is observed, that for minimum power consumption an analog feed-forward neural network is to be preferred for a number of classification problems (over digital neural nets). Analog neural networks are power-efficient if:

- the *number of first-layer neurons is small.* In this case the *power* consumption of the analog network is the lowest (note that a digital neural network with AD-conversion of input signals is only somewhat less power-efficient for not-very-low spatial resolution tasks)
- the number of first-layer neurons is arbitrary, but with *low spatial resolution required for the classification task*. In this case also the *power* dissipation of analog

neural networks is the lowest, and the required spatial distance between different classes (in the N-dimensional input space) must be larger than 10%-100% of the magnitude of the input signals

- the *operation speed must be very high*. Analog feed-forward neural networks can be *smaller* than digital ones. This is because digital functions then can not be multiplexed; the chip area required for elementary neural operations such as multiplications and additions is then relatively small in analog hardware

## 6. Acknowledgments

I am thankful to J. Richards for his help on some topics in neural networks.

## 7. References

[1] E.A.Vittoz, "Low-Power Design: Ways to Approach the Limits", in IEEE International Solid-State Circuits Conference, Dig. Paper, 1994, pp. 14-18

[2] Dominique Coue and George Wilson, "A four quadrant sub-threshold mode multiplier for analog neural-network applications", *IEEE Transactions on Neural Networks*, Vol. 7, No. 5, September 1996, pp. 1212-1219.

[3] B. Razavi, "Design of analog CMOS integrated circuits," *New York: McGraw-Hill, 2001*.

[4] A. J. Montalvo, R. S. Gyurcsik, and J. J. Paulos, "An analog VLSI neural network with on-chip perturbation learning," *IEEE J. Solid-State Circuits*, vol. 32, Apr. 1997.

[5] A. J. Montalvo, R. S. Gyurcsik, and J. J. Paulos, "An analog VLSI neural network with on-chip perturbation learning," *IEEE J. Solid-State Circuits*, vol. 32, Apr. 1997

[6] A.Kramer, M.Sabatini, R.Canegallo, M.Chinosi, P.L.Rolandi, and P.Zabberoni, "Flashbased Programmable Non-linear capacitor for Switched-Capacitor Implementations of Neural Networks", Proc. IEDM, 1994, pp. 449-451.

[7] E.Sackinger and W.Guggenbuhl, "An Analog Trimming Circuit Based on a Floating-Gate Device", IEEE J. Solid-State Circuits, 1988, pp. 1437-1440.

[8] S.M.Sze, "Physics of Semiconductor Devices", New York: Wiley, 1981.

[9] A.J.Annema, "Feed-Forward Neural Networks: Vector Decomposition Analysis, Modelling and Analog Implementation", Norwell MA: Kluwer Academic Publishers, 1995.

[10] T. Morie and Y. Amemiya, "An all-analog expandable neural network LSI with on-chip backpropagation learning," *IEEE J. Solid-State Circuits*, vol. 29, Sept. 1994

[11] C. Schneider and H. Card, "Analog CMOS synaptic learning circuits adapted from invertebrate biology," *IEEE Trans. Circuits Syst.*, vol. 38, Dec. 1991.

[12] K.Nakamura, M.Hotta, L.R.Carley, and D.J.Allstot, "An 85 mW, 10b, 40 Msample/s CMOS Parallel-Pipelined ADC", IEEE J. Solid-State Circuits, vol. 30, March 1995, pp. 173-183.

[13] J. A. Lansner and T. Lehmann, "An analog CMOS chip set for neural network with arbitrary topologies," *IEEE Trans. Neural Networks*, vol. 4, May 1993.

[14] R.Coggins, M.Jabri, B.Flower, and S.Pickard, "A Hybrid Analog and Digital VLSI Neural Network for Intracardiac Morphology Classification", IEEE J. Solid-State Circuits, vol. 30, 1995, pp. 542-550.

[15] T. Shima, T. Kimura, Y. Kamatani, T. Itakura, Y. Fujita, and T. Iida, "Neuro chips with on-chip back-propagation and/or Hebbian learning," *IEEE J. Solid-State Circuits*, vol. 27, Dec. 1992.

[16] P. W. Hollis and J. J. Paulos, "Artificial neural networks using MOS analog multipliers," *IEEE J. Solid-State Circuits*, vol. 25, June 1990.

[17] T. Lehmann, E. Bruun, and C. Dietrich, "Mixed analog/digital matrixvector multiplier for neural network synapses," *Analog Integrated Circuits Signal Processing*, vol. 9, 1996.

[18] J.B.Lont and W.Guggenbuhl, "Analog CMOS Implementation of a Multilayer Perceptron with Nonlinear Synapses", IEEE Tr. on Neural Networks, vol. 3, 1992, pp. 457- 465.

[19] T.Shima, T.Kimura, Y.Kamatani, T.Itakura, Y.Fujita, and T.Iida, "Neuro Chips with On-Chip Back-Propagation and/or Hebbian Learning", IEEE J. Solid-State Circuits, vol. 27, 1992, pp. 1868-1875.

[20] B. Linares-Barranco, E. Sanchez-Sinencio, A. Rodriguez-Vazquez, and J. L. Huertas, "A CMOS analog adaptive BAM with on-chip learning and weight refreshing," *IEEE Trans. Neural Networks*, vol. 4, May 1993.

[21] C.Kaya, D.K.Y.Liu, J.Paterson, and P.Shah, "Buried Source-Size Injection (BSSI) for Flash EPROM Programming", IEEE Electron Device Letters, vol. 13, Sept. 1992, pp. 465-467.

[22] J. Liu and M. A. Brooke, "A fully parallel learning neural network chip for combustion instability control," in *Proc. Int. Joint Conf. Neural Network*, July 1999, pp. 2323–2328.

[23] M.Jabri and B.Flower, "Weight Perturbation: An Optimal Architecture and Learning Technique for Analog VLSI Feedforward and Recurrent Multilayer Networks", IEEE transactions on Neural Networks, vol. 3, 1992, pp. 154-157.

[24] E.Vittoz, "Analog VLSI for Advanced Signal Processing", in Proc. Workshop on Future Information Processing Technologies, Helsinki, Sept 4-8, 1995.

[25] J. van der Spiegel, P.Mueller, D.Blackman, P.Chance, C.Donham, R.Etienne-Cummings, and P.Kinget, "An Analog Neural Computer with Modular Architecture for Real-Time Dynamic Computations", IEEE J. Solid-State Circuits, vol. 27, Jan 1992, pp. 82-92.

**Vipan Kakkar** received his B.Tech degree and M.Tech. degrees in Electrical Engineering in 1995 and 1998, respectively. He received his doctorate in VLSI from Delft University of Technology, Netherlands in 2002. During 2001-2008, he worked in Netherlands in VLSI design for audio and video applications. He worked in Netherlands for 8 years at different positions. Presently, he is working as Associate Professor in school of Electronics and Communications Engineering at SMVD University, India.