Interfacing Binary Networks to Multi-valued Signals

Felix Schürmann, Steffen G. Hohmann, Karlheinz Meier, Johannes Schemmel
Kirchhoff Institute for Physics, University of Heidelberg
Im Neuenheimer Feld 227, 69120 Heidelberg, Germany
Email: felix.schuermann@kip.uni-heidelberg.de

***Abstract*— Data processing of real-world problems usually leads to the use of quasi-continuous discrete quantities. If artificial neural networks are to be used they need to interface multi-valued signals. For the case of a mixed-mode analog hardware neural network of neurons with binary inputs and outputs, this paper demonstrates that it is possible to combine these to form integer input and output neurons. A precision of 6 bits has been reached. Together with programmable synapses connecting n-bit neurons to m-bit neurons they represent parameterizable building blocks useful for networks of variable precision.**

I. Introduction

The reasons for building hardware neural networks arise from either performance requirements or scalability considerations. Both aspects drove the development of a mixed-mode analog neural network (ANN) ASIC [1] and resulted in the implementation of a simplified Perceptron model which has binary neuron inputs and outputs and analog synapses. In order to use this ANN ASIC for real-world applications, but also for specialized neural network benchmark problems, multi-valued quantities need to be interfaced [2] [3] [4].

One approach is to simply present the data as a bitstream to the network and make the enconding part of the learning task. In other situations it may be more suitable to feed the multi-valued data directly into the network, i.e., generate an activation proportional to the input value. The overall performance is influenced by this decision [5]. Although neural networks often work as classifiers and only binary outputs are needed, in other cases multi-valued outputs may be desirable to obtain a fine-grained output.

In [6] it was shown how multi-valued input and output neurons can be realized for the special case of 3-bit integer neurons using a first prototype ANN ASIC [7]. In this paper these results are reproduced with a second generation ANN ASIC [1]. The initial results are generalized by showing the parameterizablity of the variable neurons and synapses, the so called building blocks. The parameters varied are the number of bits, i.e., the precision of the inputs and outputs, as well as the weighting of the synapses connecting n-bit inputs with m-bit outputs. A precision of 6 bits is reached.

II. ANN Framework

All results presented were obtained using the ANN framework described in [6] with the ANN ASIC called HAGEN (Heidelberg AnaloG Evolvable neural Network). The framework consists of a standard PC with a PCI-based interface card using a programmable logic to control the ANN ASIC, which actually performs the neural network operations.

The purpose of this ANN framework is to combine a hardware implemented neural network of large size and good scaling behaviour with the flexibility found in software implementations. The used mixed-mode analog ANN ASIC [1] is realized in a $0.35\mu m$ CMOS process and implements a Perceptron model with binary neuron inputs and outputs and analog synapses. A synaptic weight is represented by an analog current and the neuron evaluates the difference between the summed positive and negative currents. The simplicity of the synapses and neuron circuits allows for highly integrated ANNs. The used ASIC realizes 32768 synapses, distributed across four equally sized blocks of 128 input and 64 output neurons.

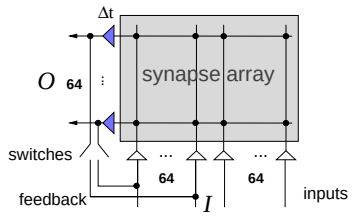

Fig. 1. Network block of the ANN ASIC (HAGEN).

All experiments were performed using one of the four network blocks. Nevertheless, the whole concept of the ANN framework as described earlier is capable of extending this to all four network blocks or even across chip boundaries. Within the block all outputs can be individually fed back into a predetermined set of 64 of the 128 input neurons (see Fig. 1). The network operation is clocked, i.e., recurrent information is fed into the network one cylce after it is available at the outputs. This behavior is described by:

$$O(t+\Delta t)_i = \Theta(\sum_j \omega_{ij} I(t)_j + \sum_k \omega_{ik} O(t)_k)$$

With Δt being the time needed for one network cycle, ω_{mn} being the elementary synaptic weights, $I \in \{0,1\}$ being the inputs and $O \in \{0,1\}$ being the outputs. The activation function is the Heaviside function Θ. It is therefore possible to implement multi-layer feedforward and recursive network topologies as seen in Fig. 2.

III. Variable Network Resources

A. Setup

The results presented in this paper rely on the homogeneity of the chip's resources since theoretically calculated weights

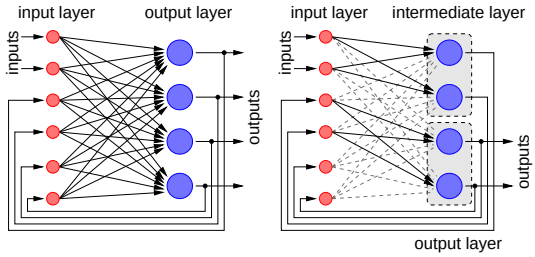

Fig. 2. **Left:** recurrent network, **right:** configured as a two-layer network by setting some synapses to zero (dashed lines).

are used. The weights of the ANN ASIC are written with 10-bit digital-to-analog converters (DACs) and a sign bit, but variations during the fabrication of the ASIC as well as electrical fluctuations, such as noise and crosstalk, can cause a smaller effective analog precision. The weight storage units of the ASIC therefore have built-in correction capacitances and it is important to note that the chosen network model allows the compensation of the major sources of mismatch by only additive offsets. It is possible to evaluate the variations and use bias synapses and single synapse offsets to calibrate the network. For the experiments, the neuron biases were compensated, but single synapse offsets were not considered because their fixed-pattern noise is in the order of their variation over time. Technical details on the hardware and the calibration procedure can be found in [8].

Variations of the electronic environment and in the ASIC itself can cause different network responses for an repeatedly applied activation pattern. Any kind of network evaluation and measurement therefore has to be performed several times. The measurement of the input current of the neurons, the network activity, is done by repeatedly applying the activation pattern and sweeping a measurement synapse until the neuron is activated 5 out of 10 times. This is the point where the measurement synapse compensates the network activity and the neuron flips from inactive to active. This is done for two different measurement synapses. In order to perform a measurement free of the individual synapse offsets, a third sweep is done using both synapses simultanously. The three measurements allow to eliminate the two synapse offsets while still obtaining the activity. These sweeps are done bottom-up and top-down in order to eliminate systematic errors and are repeated twice for each activation pattern. The sweep is done in least significant bit (LSB) increments of the 10-bit resolution of the synapse. The maximum current of each synapse is set by the periphery to a full scale value of about $FS = 45\mu A$.

B. Interfacing Multi-valued Inputs

A straightforward way to feed multi-valued signals into a network with binary inputs is to consider n inputs as a group encoding an n-bit integer. The integer value then needs to be translated into an analog network activity according to the significance of the representing bits. The LSB induces ω, the next bits 2ω, $2^2\omega$ and so on. The most significant bit (MSB) induces $2^{n-1}\omega$ into the network. This is a digital-to-analog conversion with ω normally chosen not to exceed the dynamic range of a single 1-bit synapse for a full source neuron activation, i.e., $\omega \in [-FS/(2^n-1), FS/(2^n-1)]$.

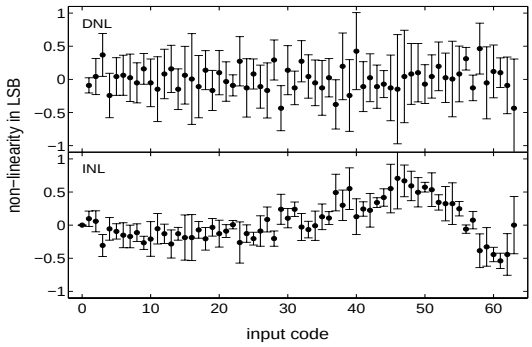

Fig. 3. Differential and integral non-linearity versus the input code for a 6-bit DAC.

A good measure for the performance of a DAC is the differential non-linearity (DNL) which describes the deviation of any of the analog output changes caused by an LSB change from its ideal step size. The DNL is calculated after a gain and offset correction (linear fit). Another measure is the integral non-linearity (INL) which accounts for the deviation from a reference curve drawn as a staight line through the end points. Both measures are given in LSB:

$$DNL = \frac{y(n) - y(n-1)}{y_{idealLSB}} - 1, \quad 0 < n < 2^N$$

$$INL = \frac{y(n)}{y_{idealLSB}} - n, \quad 0 \leq n < 2^N$$

Here $y(n)$ is the actual value measured for the n-valued digital input code and $y_{idealLSB}$ is the current ideally represented by an LSB and N the number of bits.

Fig. 3 shows the performance of a single precalculated 6-bit DAC versus the 64 possible input codes. The weight ω is chosen as to get a maximum activation of 0.8 of the full scale. The given error bars are the standard deviations of the repetitive measurements. Since the DNL is always smaller than 1 LSB, the monotonicity of the DAC is guaranteed. In order to characterize a DAC, one gives the maximums of the absolute non-linearity, $\text{DNL}_{max} = max(|DNL|)$ and $\text{INL}_{max} = max(|INL|)$. Table I lists the characteristic values together with the gain normalized to 1 and the absolute offset. The left column shows the characteristics of the DAC plotted in Fig. 3 with the errors being the mean measurement error.

If several integer inputs shall be connected to the same output neuron, the variation of DACs placed in the same synapse row is important. The characteristic values for an exemplary neuron are given in the second column of Table I. For the measurement, the 6-bit DAC was shifted horizontally besides for the 10 columns at the right side of the synapse array which are used for measurement and biasing. The horizontal displacement measurements of 6-bit DACs done for all output neurons are listed in the third column and give an estimation

TABLE I

CHARACTERISTICS OF VARIOUS 6-BIT DACS GIVEN IN LSB.

	single DAC	same neuron	block	block (5-bit)
DNL_{max}	0.5 ± 0.4	0.5 ± 0.3	1.1 ± 0.6	0.4 ± 0.2
INL_{max}	0.7 ± 0.2	0.9 ± 0.2	1.7 ± 0.7	0.7 ± 0.3
gain	1 ± 0.07	1 ± 0.07	1 ± 0.1	1 ± 0.1
offset	0 ± 0.2	0 ± 0.2	-0.1 ± 0.7	-0.1 ± 0.2

of the homogeneity of the whole network block. The values are the weighted means and the errors are the widths of the distributions covering 70% of the measurement results. For comparison the whole block measurement is repeated for 5-bit DACs (last column of Table I, given in corresponding LSB).

These benchmarks show that the homogeneity for a single output neuron is quite good. Even for the whole block the non-linearity errors are below 2 LSB. A closer examination shows that the increased non-linearity error is mainly caused by every eighth output neuron. They show non-equidistant steps in the characteristic curve, but monotonicity is not violated. This effect may be due to an incorrect timing during the programming of the weight storage process, in which case the problem may be fixed in the future. In case the observed effect is inherent to the hardware, appropriate offsets may be used to correct it.

C. A Variable Synapse

Physically, a synapse in the ANN ASIC connects a single binary input to a single neuron with binary output. In case of n input neurons forming a multi-valued input, their synapses connecting to the same output have to be treated as a fixed group. However, it is possible to introduce a common scaling factor $\Omega \in [-1, 1]$. This ensemble then represents a synapse of weight Ω capable to interface an n-bit neuron (see Fig. 5), in the following it is referred to as an n-to-1 bit synapse. If m output neurons are used and their synapses connecting the same n-bit input are identically configured and simultaneously scaled by Ω, they form an n-to-m bit synapse. The case $n = m = 1$ is the elementary synapse and $\Omega = \omega$.

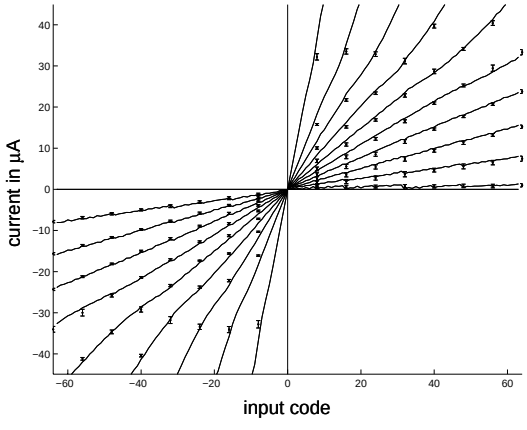

Fig. 4. Characteristic curves of a synapse varying Ω.

In Fig. 4 the analog activity for a single output neuron is plotted versus the inducing 6-bit integer input for a scaled Ω. In order to obtain equidistant angles, Ω was scaled according to $\Omega = tan(n/N * \pi/2)$ with $N = 9$ and n going from $-N$ to N. The error bars show the standard deviation from the mean of 6 identically configured neurons, which together form the n-to-m-bit synapse used to obtain the ADC shown below. For visualization reasons the errorbars are not plotted for all input codes. Negative Ω are flipped to the lower left quadrant for visualization, i.e., the presented input codes are positive. The INL_{max} is below 3 LSB besides for the zero weight. Towards smaller currents the neuron circuit works less precise. Theoretically, the 6-to-m-bit synapse has a resolution of 5 bits (the remaining resolution of the weight storage units plus the sign bit) in the range of $\Omega \in [-1, 1]$. It can have values outside this interval, but then the synapse is not 6-bit anymore since it saturates for higher input codes.

D. Obtaining Multi-valued Outputs

Similar to the combination of binary input neurons for multi-valued inputs it is possible to group elementary output neurons to act as an m-bit integer output in the range of $[0, 2^m - 1]$. The task to be performed by this group of neurons is to measure the analog network activity present at their inputs and represent this activity as an integer. They act as analog-to-digital converters (ADCs). This can be realized by configuring a recurrent network topology with self-inhibiting feedback connections acting as a successive approximation ADC. The proposed solution imposes two conditions: First, the participating output neurons need to be excited by the same analog activity and, second, the analog activity has to stay stable over the course of the successive approximation.

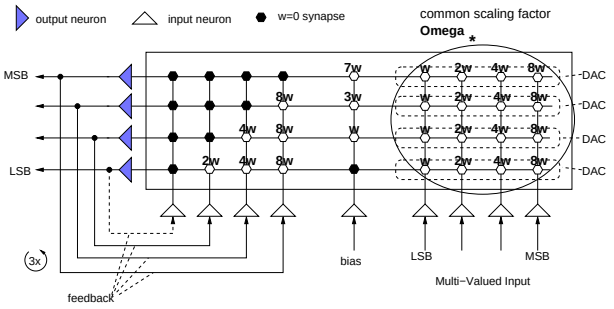

Fig. 5. Network configuration of a 4-bit input being connected to a 4-bit ADC by a 4-to-4-bit synapse of weight Ω.

For an m-bit output with linear activation function the necessary resources are found in a straightforward fashion: one binary input is used as a bias (constantly activated) and adjusts the threshold values for the participating bits according to their significance, i.e., for the LSB 0, for the next bit ω and for the MSB $(2^m - 1)\omega$. After one network cycle, the MSB has reached its final state and subtracts $2^m\omega$ from all other bit lines if active. After the second cycle, the second most significant bit is stable and will go on to adjust the lower significant bit lines and so on. This process is completed after m network cycles. The easiest way to ensure stability of the solution is to prohibit feedback connections of bits to themselves or higher

significant bits, i.e., the upper left and the diagonal of the weight matrix need to be zero. Compare Fig. 5 to see the exemplary network configuration of a 4-bit ADC. On the right hand side a 4-bit input with a common weight of Ω is shown and the left side depicts the bias and the 4-bit ADC.

A single m-bit output neuron requires, besides its obligatory m elementary output neurons, $m-1$ feedback connections to $m-1$ elementary input neurons and binds $(m-1) \times m$ synapses. The outputs are valid after $m-1$ network cycles. If used within a network of variable neurons, all m outputs need to be fed back increasing the required network cycles by 1. The necessary bias neuron can be shared.

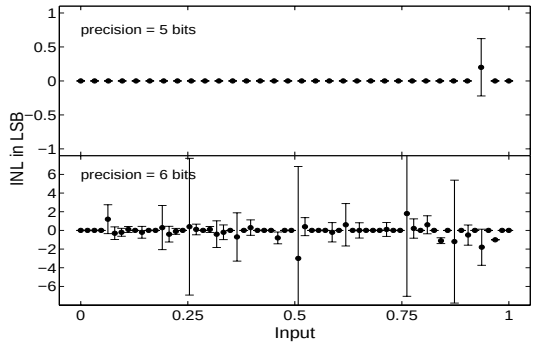

Fig. 6. Integral non-linearity of a 5-bit and a 6-bit ADC plotted in corresponding LSB versus the normalized input code.

In Fig. 6 the scaling behavior of the ADC building block is shown. The used setup is similar to the one shown in Fig. 5, but with 5 and 6 bits. The activity is induced by DAC building blocks with a maximum current of 0.8 of the full scale FS. For 5-bit and 6-bit ADCs the integral non-linearity is plotted versus the input code. Each input code is applied 10 times. The plotted error bars show the standard deviation. The performance of the 5-bit ADC is almost perfect. In the 6-bit case there are deviations of a few LSB for input codes where the MSB or the second most important bit turn on and the lower bits go off.

IV. Networks of Variable Network Resources

In the previous section it was demonstrated that the analog properties of the ANN ASIC allow to combine its elementary resources to multi-bit resources. One aspect is to be able to connect several multi-bit inputs to a single output neuron by weighted connections. The second is the ability to reproduce the activity to several neurons as to get a multi-valued output.

In terms of resources used this allows to build whole networks of neurons with variable precision efficiently. Input can be provided by the variable neuron input blocks, which then can be connected via the proposed n-to-m-bit synapses to the variable output blocks acting as inner and output neurons. The blocks for the output neurons remain unchanged once configured according to the desired precision, whereas the synapse blocks are scaled in order to represent the connection strength.

V. Conclusions & Outlook

In the presented paper it has been shown that the concepts of combining the elementary resources of a mixed-mode analog ANN ASIC introduced in [6] can be (a) transferred to a second generation ANN ASIC and (b) generalized to parameterizable building blocks. These building blocks are variable network resources, namely multi-bit inputs and outputs scaling in the number of bits. Variable synapses connecting neurons of variable precision were realized up to a precision of 6 bit. The ANN ASIC provides enough analog precision to connect several 6-bit inputs operating as DACs to a single neuron while maintaining a scalable weight Ω with 5-bit resolution. For the multi-bit outputs it was shown that it is possible to induce the same activity onto several output neurons, thus making it possible to use a group of m 1-bit output neurons as an m-bit successive approximation ADC for m up to 6.

The presented results show that the simplified Perceptron model implemented in the ANN ASIC can be used to interface multi-valued signals in a straightforward fashion, although physically there are only single bit inputs and outputs provided. It is possible to configure only the input and/or output layer of a network with these variable network resources while using the elementary neurons and synapses for the inner layers. Current projects in our group use the variable inputs for interfacing real-world data for a classifier task. Ultimately, the training algorithm can choose what kind of network resources are needed to solve the given problem. Under the constraint of minimizing the used resources, it may come up with networks using a variety of different types of neurons, thus allowing to analyze the criticality of precision in certain stages of the problem solving.

References

[1] J. Schemmel, F. Schürmann, S. Hohmann, and K. Meier. An integrated mixed-mode neural network architecture for megasynapse ANNs. In *Proceedings of the 2002 International Joint Conference on Neural Networks IJCNN'02*, pages 2704–2710. IEEE Press, 2002.

[2] S. Hettich and S. D. Bay. The UCI KDD archive [http://kdd.ics.uci.edu]. University of California, Department of Information and Computer Science, Irvine, USA, 1999.

[3] L. Prechelt. Proben1: A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, 38 pages, Fakultät für Informatik, Universität Karlsruhe, 1994.

[4] K.J. Lang and M.J. Witbrock. Learning to tell two spirals apart. In D.S. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of 1988 Connectionist Models Summer School*, pages 52–59. Morgan Kaufmann Publishers Inc., 1988.

[5] P. J. B. Hancock. Data representations in neural nets: an empirical study. In D.S. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 11–20. Morgan Kaufmann Publishers Inc., 1988.

[6] F. Schürmann, S. Hohmann, J. Schemmel, and K. Meier. Towards an Artificial Neural Network Framework. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R.S. Zebulum, editors, *Proceedings of the 2002 NASA/DoD Conference on Evolvable Hardware*, pages 266–273. IEEE Computer Society, 2002.

[7] J. Schemmel, K. Meier, and F. Schürmann. A VLSI implementation of an analog neural network suited for genetic algorithms. In *Proceedings of the International Conference on Evolvable Systems ICES 2001*, pages 50–61. Springer Verlag, 2001.

[8] J. Schemmel, S. Hohmann, K. Meier, and F. Schürmann. A mixed-mode analog neural network using current-steering synapses. In *Analog Integrated Circuits and Signal Processing*. Kluwer, in press.