A vector quantization circuit for trainable neural networks

Fabio Ancona, Giorgio Oddone, Stefano Rovetta,
Gianni Uneddu, and Rodolfo Zunino

University of Genova, Via all'Opera Pia 11a 16145 Genova (Italy)
E-mail Rovetta@dibe.unige.it

Abstract – *Vector quantization systems are usually implemented in hardware by realization of an algorithm, usually exploiting accelerated techniques for codebook search. These implementations are not well suited for the use as analog electronic neural networks building blocks. This paper presents an analog, fully parallel implementation of vector quantization exploiting a large number of simple processors. The circuit features large-dimensional (64) vectors and a medium-to-high density of units per chip. Moreover, the winner-take-all block features a linear output that replicates the value of the winning distance, in addition to the winner's location flag. This makes it possible to use the system in trainable networks without need for further circuitry.*

INTRODUCTION

Signal processing applications are to date a major (maybe the most important) area of interest for the application of neural network processors [1]. The challenge posed by heavy-duty tasks such as image compression and real-time video coding/decoding can be successfully overcome by neural algorithms, which are inherently suited to deal with non-linear, non-Gaussian and non-stationary signals [2]. The main goal is now to devise efficient hardware implementations which should allow these algorithms to be applied on line and in real time.

Current "conventional" solutions usually resort to digital hardware implementing optimized algorithms, distributed over specialized modules in a chipset or provided by specialized functions of dedicated DSPs [3].

We present a circuit that implements in analog VLSI the feedforward step of a vector quantization system. The design is based on neural networks principles: a high number of simple parallel analog processors is connected in a network with the minimum amount of centralized control functions, and the function is implemented directly by the structure rather than by a sequence of elementary steps (an algorithm).

The circuit is currently under design, so the results presented here are preliminary; only selected parts of the circuit will be described. In the full version of the paper, additional details will be presented.

NEURAL NETWORKS AS IMAGE PROCESSORS

The image compression principle adopted by most neural network image-processing systems is that of vector quantization [4, 5], since other principles (such as transform coding) are translated into neural structures with less advantage.

Image compression is accomplished by the following procedure. The original image is divided into square blocks containing a limited number of pixels. The pixel values are used as components of the vector to be input in the transmitter. The transmitter then searches the set of available reference vectors (collectively referred to as the codebook), and selects the reference vector featuring the minimum distortion (distance) with respect to the input vector. The index of the best-matching vector is then transmitted, and the receiver uses the reference vector to represent the input vector and reconstruct an approximate version of the original image.

Figure 1: Functional block diagram of the circuit

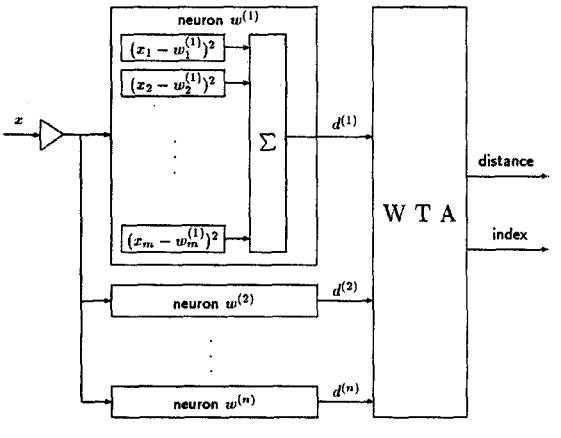


Figure 2: The square–of–difference circuit

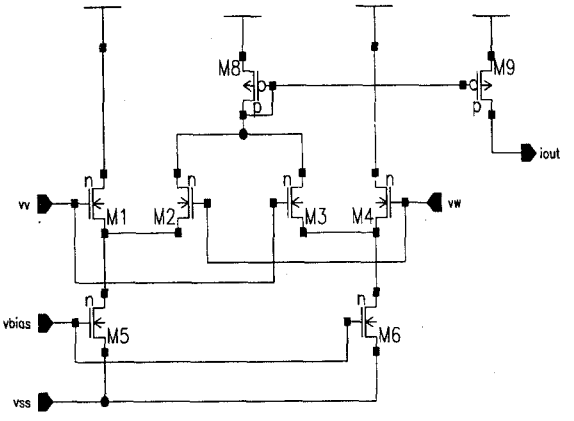
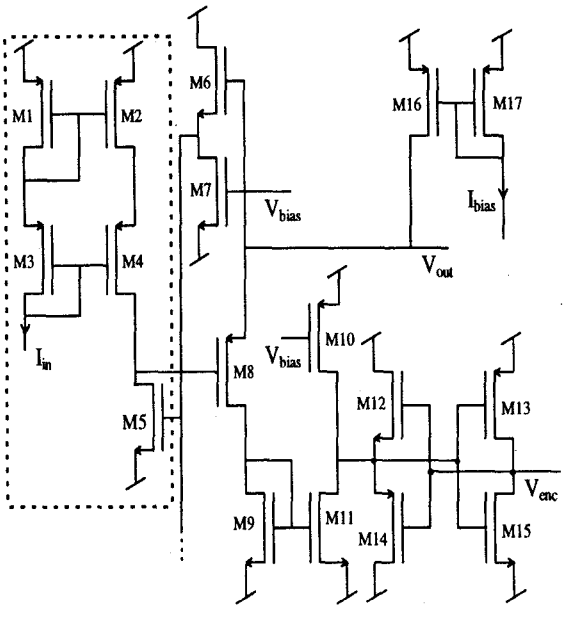


Figure 3: The winner–take–all circuit



Vector quantization is often implemented by the conventional, digital approach [6, 7, 8, 9] by selection of a fixed codebook (usually by the well known Linde–Buzo–Gray [10] algorithm) and an optimized algorithm for vector search. However the LBG algorithm requires batch training, hence the training set should be available completely at each training step; moreover, from the standpoint of distortion minimization, it suffers from a large number of local minima, which means that resources (memory to store the codebook and time to search it) are not optimally exploited. Neural networks performing VQ are usualy based on on–line adaptation, with more sophisticated training algorithms (Kohonen's SOM [11], Martinetz' Neural Gas [12]). On–line training (also termed training by pattern or stochastic optimization) falls less easily into local minima and offers the valuable advantage of being applicable even during the actual functioning, thus allowing the system to deal with time–varying input statistics (nonstationary input).

All these models, while different in the training process, share the same feedforward step, which is simply the search of the best match among a set of reference vectors or prototypes, with respect to an input vector. This is usually a time–consuming process, since it implies computing a quite large set of distances and selecting the minimum. The circuit proposed here performs a fully parallel vector search, so it need not an optimized search algorithm. It is therefore suited as a building block for any kind of vector–quantization system.

CIRCUIT DESCRIPTION AND DESIGN CONSIDERATIONS

The block diagram of the circuit is illustrated in Fig. 1. All the illustrated functional blocks are physically present, and there is no multiplexing. Therefore, the circuit features $O(1)$ time complexity (independent on both the number n and the dimension m of the vectors in the codebook). Each neuron k reads the input vector and computes its euclidean distance from the locally stored reference vector $w^{(k)}$. Euclidean distance is the distortion measure commonly adopted in implementations of vector quantization. The outputs of all neurons feed the winner-take-all layer, and the final circuit output is represented by a binary encoding of the best–matching neuron and by a voltage proportional to the corresponding distance value.

The availability of the distortion value corresponding to the winning neuron makes the circuit suitable for implementing all types of vector quantizer, with or without on–line training, including all neural algorithms such as the two quoted in the previous section.

The requirements for this circuits are those of real–world applications; this accounts for a number of non–conventional design choices. The usual subthreshold design has been discarded, since its sensitivity to external and internal noise and interferences is too high. The standard circuits building blocks have been redesigned or redimensioned to cope with the increased voltage ranges and power dissipation requirements.

The vector dimension is 64 (corresponding to elementary image blocks of size 8x8), which reduces the number of prototypes (for a fixed area consumption) with respect to other projects, such as for instance in [13], which features a vector dimension $m = 25$. This in turn implies that

the overall circuit will be realized as a connection of homogeneous building–block chips, but this is not a drawback, since it will allow an easier power dissipation and a customized configuration of the codebook size for each application. Each chip is estimated to implement about 64 neurons (codebook of $m = 64$) in an area of about 1cm x 1cm. No special circuitry will be required to connect many chips into a single vector quantizer, only a supplementary competitive layer to select the best match among the partial best matches of each chip.

The external analog signals, *i.e.*, the input vector components and the output distortion value, are represented as voltages. However, the internal signal exchange is done in current mode, which simplifies greatly the circuits.

The overall scheme of the circuit is as follows (refer to figure 1). The input vector is buffered and read by the 32 neurons. Each neuron is composed of 64 partial distortion stages, computing the squared difference between vector component and prototype component. This block is based essentially on the same principles as the well–known Gilbert multiplier cell, but modified (see Fig. 2) for a wider range and a better precision of the square function, with a limited number of components.

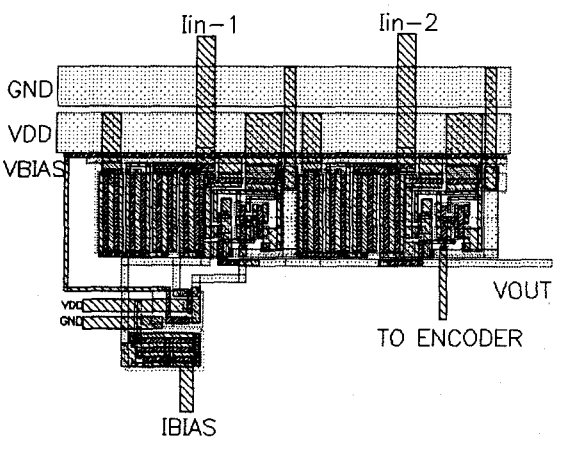
Reference vectors are memorized in a digital memory. There are 64 D/A converters, one for each component, and their output voltages are multiplexed over the neurons to refresh local memory elements (capacitors). To avoid leakage during switching (mainly due to clock feed–through effects) dummy switches are used [14]. The refresh cycle time is 1ms.

The last stage of the circuit is a competitive layer. The best matching vector is selected by a winner–take–all circuit which has been modified to yield the distortion value relative to the winning prototype. In the following, the competitive stage of the circuit will be detailed.

THE WINNER–TAKE–ALL BLOCK

The circuit that implements the selection among the reference vectors implies several modifications on the standard scheme by Lazzaro *et al.* [15]. When realized with a large number of inputs, Lazzaro's scheme is not stable, since the smallest inputs can receive a reverse current flowing from the largest ones. Moreover, the input impedance (the drain impedance of a MOS device) can be quite high if the input transistors are to be kept within limited dimensions. Therefore, if the input branches are not connected to ideal current generators, this may cause problems eventually affecting the precision of the circuit.

Figure 4: Layout of the winner–take–all circuit



The proposed solution, illustrated in Fig. 3, features many improvements. The input lines (dotted rectangle on the upper left) read an input current and mirror it on the "competition" node (gate of M5). The upper mirror enhances the circuit's performance with non-ideal input current generators. Moreover, each input line is biased with a constant current offset, which modifies the operating point so that the input impedance is greatly reduced. Generation of the bias current, about $40\mu A$, does not require additional circuitry, but only a proper dimensioning of the previous stage in the circuit.

The subsequent stages are two complementary replicas (one with n-channel MOS, the other with p–channel MOS) of the output stage of a standard WTA. This arrangement creates a competition that selects the minimum (rather than the maximum) current. This allows us to avoid adding external circuitry to reverse the behavior of a standard WTA.

There are two output voltages, V_{out} and V_{enc}. If the input range is within reasonable limits, the input-output relation between the winning current and the voltage V_{out} is linear, and the proportionality factor depends only on geometrical and physical parameters, but not on input signals. This is another effect of the current offset.

The other voltage, V_{enc}, constitutes a 1–out–of–n encoding of the winning input position. Devices M12–M15 push the value of V_{enc} to either the maximum or the minimum value, so that there is a logic "1" corresponding to the winner, and a logic "0" elsewhere. These values are then encoded into a 6-bit word (not shown).

For the simulation with the HSPICE program (level 13), $1\mu m$ technology, the following parameters were used: Vdd = 2.5V, Vss = -2.5V; $k_n = 60\mu A\ /V^2$; $V_{th} = 0.8V$. Transistor sizes are the

following:

MOS	W/L	MOS	W/L
M1-M4	100/3	M8	15/3
M5	10/3	M9-M11	5/2
M12	2/3	M13-M14	6/3
M6	8/2	M15	2/3
M7	3/15	M16-M17	39/3

The layout of this block is shown in Figure 4.

The input/output proportionality factor is 5kΩ $\pm$4.5%, for operating temperature varying from 40°C to 80°C. The minimum input variation that can be discriminated (resolution) is less than 0.4μA , with input current ranging from 40μA to 110μA (80μA swing). This allows a precision of at least 7 bits, which is attained when the resolution is 80μA / 128 levels = .625 μA or less.

CONCLUSIONS

In this paper, an overall description and preliminary results about an analog VLSI vector quantizer have been reported, with a more detailed description of selected features. The circuit is not oriented to global integration, but to modular system building. Since the operating speed of the quantizer is very high, and limited by the inter–chip communications, the applications of this design will be in the field of high–throughput image compression, such as real–time videocompression and multimedia.

The whole project will be described thoroughly in a paper currently in preparation.

REFERENCES

[1] Robert D. Dony and Simon Haykin, "Neural network approaches to image compression", *Proceedings of the IEEE*, vol. 83, no. 2, pp. 288–303, February 1995.

[2] Simon Haykin, "Neural networks expand SP's horizons", *IEEE Signal Processing Magazine*, vol. 13, pp. 24–49, March 1996.

[3] Ryota Kasai and Toshihiro Minami, "An overview of video coding VLSIs", *IEICE Transactions on Electronics*, vol. E77, no. 12, pp. 1920–1929, December 1994.

[4] Allen Gersho, "On the structure of vector quantizers", *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 157–166, March 1982.

[5] R.M. Gray, "Vector quantization", *IEEE Acoustic, Speech and Signal Processing Magazine*, vol. 1, pp. 4–29, 1984.

[6] Rajeev Jain, Avanindra Madisetti, and Richard L. Baker, "An integrated circuit design for pruned tree–search vector quantization encoding with an off–chip controller", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 2, pp. 147–158, June 1992.

[7] Heonchul Park and Viktor K. Prasanna, "Modular VLSI architectures for real–time full–search–based vector quantization", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 4, pp. 309–317, August 1993.

[8] Wai-Chi Fang, Chi-Yung Chang, Bing J. Sheu, Oscal T.-C. Chen, and John C. Curlander, "VLSI systolic binary tree–searched vector quantizer for image compression", *IEEE Transactions on VLSI Systems*, vol. 2, no. 1, pp. 33–44, March 1994.

[9] Kevin Tsang and Belle W. Y. Wei, "A VLSI architecture for a real–time code book generator and encoder of a vector quantizer", *IEEE Transactions on VLSI Systems*, vol. 2, no. 3, pp. 360–364, September 1994.

[10] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizers design", *IEEE Transactions on Communications*, vol. COM–28, pp. 84–95, January 1980.

[11] Teuvo Kohonen, *Self Organization and Associative Memories*, Springer Series in Information Sciences. Springer, 1982.

[12] T.M. Martinetz, S.G. Berkovich, and K.J. Schulten, "'Neural gas' network for vector quantization and its application to time–series prediction", *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 558–569, 1993.

[13] Wai-Chi Fang, Bing J. Sheu, Oscal T.-C. Chen, and Joongho Choi, "A VLSI neural processor for image data compression using self–organization networks", *IEEE Transactions on Neural Networks*, vol. 3, no. 3, pp. 506–518, May 1992.

[14] Cristoph Eichenberger and Walter Guggenbuhl, "On charge injection in analog CMOS switches and dummy switch compensation techniques", *IEEE Transactions on Circuits and Systems*, vol. 37, no. 2, pp. 256–264, February 1990.

[15] J. Lazzaro, R. Ryckebush, M. A. Mahowald, and C. Mead, "Winner-take-all networks of $O(n)$ complexity", in *Advances in Neural Information Processing Systems II*, San Mateo, 1989, pp. 703–711, Morgan Kaufmann.