A NOVEL APPROACH FOR IMPLEMENTING OF A LOG-SIGMOID FUNCTION ON A FPGA DEVICE USING THE SFLOAT24 MATH LIBRARY – AN MODELLING –

M. C. Miglionico
Department of Architecture and Industrial Design "Luigi Vanvitelli"
San Lorenzo Abbey – Second University of Napoli
Via S. Lorenzo, Monastero di San Lorenzo, I-81031 Aversa (CE) – ITALY
E-mail: mcristina.miglionico@unina2.it

F. Parillo*
Department of Electrical Engineering and Information "M. Scarano"
University of Cassino, Via G. Di Biasio 43, I-03043 Cassino (FR) - ITALY
E-mail: f.parillo@unicas.it

ABSTRACT

Artificial Neural Networks (ANN_s) base their processing capabilities in parallel architectures. This makes them useful to solve pattern recognition, system identification and control problems. This work focuses the configuration of a Field Programmable Gate Array (FPGA) to realize an Activation Function (AF) utilized in ANN_S. The most popular AF is the Log-Sigmoid function, which has different possibilities of realizing in digital hardware. In particular a two parallel processes approach, in order to optimize hardware resources of the used FPGA device, is presented here. The first process consists in a fixed-point computation of 2^n that requires minimal hardware resources. The second process executes a second order symmetric polynomial function extended to adder output of an Artificial Neuron (AN). The log-sigmoid function has been obtained multiplying the functions obtained by two parallel processes. The second process has been implemented by means the custom developed *sfloat24* math library.
This function is one the most used activation function in the ANN_S because its differentiable nature that makes it compatible with any classical back propagation algorithm.
The simulation results show that the proposed approach of implementing the log-sigmoid activation function is feasible, and it outperforms by high-speed response.

Keywords: Artificial Neuron (AN), Field Programmable Gate Array (FPGA), Log-Sigmoid function, *sfloat24* math library.

1. Introduction

Artificial Neural Networks (ANN_S) are the computational models of human brain. ANN_S applications range from function approximation to pattern classification and recognition. The ANN_s are used in all the circumstances or/and cases study, where is very hard or impossible to solve the formulated problem using a closed analytical formulation. Hardware implementation of ANN_S to utilize the parallelism can follow analog, digital or mixed signal design technique. Matured and flexible digital design in Very Large Scale Integration (VLSI) can be implemented on Application Specific Integrated Circuits ($ASIC_S$) or Field Programmable Gate Arrays ($FPGA_S$).

* Corresponding author

ASIC design has some drawbacks like the ability to run only specific algorithm and limitations on the size of a network, instead, $FPGA_S$ offers a suitable alternative that has flexibility with an appreciable performance. It maintains high processing density, which is needed to utilize the parallel computation in an ANN.
Every digital module is concurrently instantiated on the FPGA and hence, operates in parallel. (Saichand, V.; Nirmala, D.M.; Arumugam, S.; Mohankumar, N., 2008).
As well known, in the literature are shown several approximation methods used to implement a log-sigmoid function. These methods are based on piecewise linear approximation (PWL), lookup table (LUT), and hybrid methods. Generally, the LUT implementations are the fastest, but they consume a large area of silicon. The PWL approximation method approximates the function with a limited number of linear segments, with linear approximations the accuracy of AF depends by the number of the used linear segments. In this case the resolution of AF is proportional to the number of the used segments. (Namin, A.H.; Leboeuf, K.; Muscedere, R.; Huapeng Wu; Ahmadi, M., 2009).
Neurons activation functions are one of the major challenges in hardware implementation of ANN_S.
Both digital and analog modules can be used to realize the activation function in hardware implementations depending on the type of the ANN. ANN_S can be generally categorized into three groups: digital, analog, and hybrid (mixed-signal) neural networks. In digital neural networks, both synaptic weight storage cells and activation function are realized by digital gates such as lookup tables (LUT_S) which are generally used to approximate the activation function. In analog neural networks, on the other hand, analog circuits are used both to estimate the activation function and to store the synaptic weights.
The third group of ANN_S are Hybrid Neural Networks (HNN_S) which are a combination of digital and analog gates. In HNN_S, analog circuits are employed to realize the activation function while weights are stored digitally (Khodabandehloo, G.; Mirhassani, M.; Ahmadi, M., 2012). It is obviously that the nonlinear activation function implementation with higher accuracy improves the learning and generalization capabilities of neural networks.
In this paper the authors present the design of an AN on an ALTERA® Cyclone III EP3C25F324C8 FPGA evaluation board.
The input stage of implemented neuron (Figure 1) is based on a *sfloat24* adder. For implementing the sigmoid activation function is used a novel analytical approximation of the same. This approach is based in using a 2^n, with n an integer number, function in conjunction with a second order symmetrical polynomial interpolation in order to approximate, with a minimum error, the following function:

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

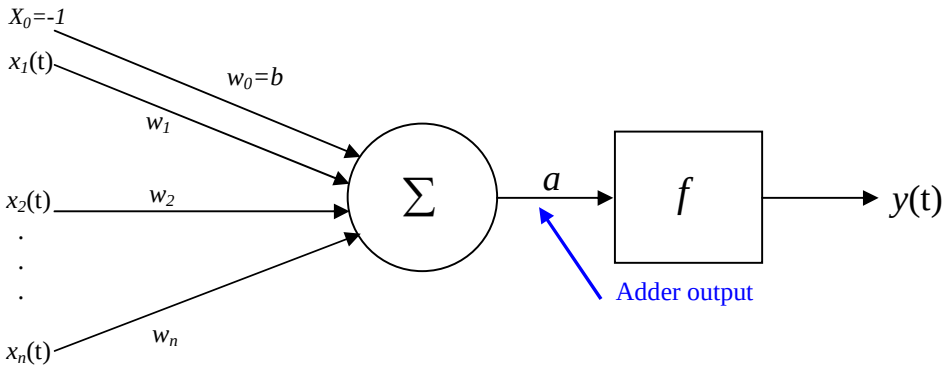

Figure 1. Block diagram of an artificial neuron.

y(t) is the output of the neuron. W is the input vector, $x_i(t)$, $i \in [0;n]$ are the inputs. A generic AN can be generalized as the following:

$$y = f(a) = f(\sum_{i=0}^{n} w_i \cdot x_i) = f(W \otimes X) \qquad (2)$$

with $X = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$ and $W = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ . \\ . \\ w_n \end{bmatrix}$

where: x_0 is equal to -1 and $w_0=b$, where b is a bias and/or a threshold.

2. Sigmoid Function Modelling

A sigmoid function can be approximate also using a 5th order Taylor series (Bezborah, A., 2012), CORDIC theory (M.C. Miglionico, F. Parillo, 2011a) or using a very simplified math expression as the following:

$$f_{ss} = 0.5 + (0.5 \cdot \frac{a}{1+|a|}) \qquad (3)$$

The advantage of using the expression (3) that it is very simple (M.C. Miglionico, F. Parillo, 2011b) to be implemented, but the absolute difference between the expressions (1) and (3) is about equal to ± 0.1. Using the expression (3) is possible to implement ANN_S capable to solve simple decisional cases, where the usage of (3) in place of the (1) is not a critical factor.
To approximate the expression (1) with a minimum error, in this work, has been used the expression:

$$f(x) = 0.5 \cdot \left(1 + s - \frac{s}{2^{s \cdot x}}\right) \qquad (4)$$

where *s* is equal to -1 for *x*<=0 else *s*=1.
It is well know that the exponential function implementation, in floating point number format, is a hard task if a FPGA device is used. On this devices family is only possible to implement the power of 2 using integer numbers, the VHDL syntax to perform this operation is:

```
Power_of_2:=2**i
```

with *i* any integer number.
The floating point exponential on a FPGA device with reference to the expression (4) is implemented by writing a VHDL algorithm based on execution of two parallel processes.
In the first process is considered the expression:

$$y_i(k,s) = 2^{s \cdot k} \qquad (5)$$

where *k* is the integer part of *x* and *s* is the already mentioned sign function and it is depicted in the Figure 2.

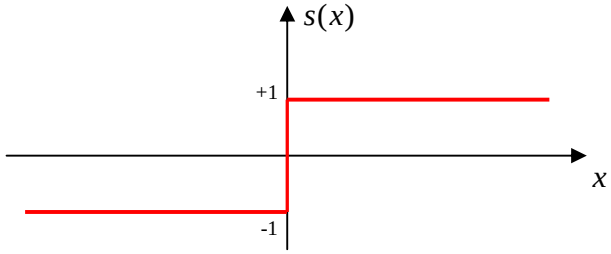

Figure 2. Graph of Sign function.

This first part is easily synthesizable on any FPGA device, using only integer numbers the (4) could be written as:

$$f(x) = 0.5 \cdot \left(1 + s - \frac{s}{y_i(k,s)} \right) \qquad (6)$$

f(x) assumes only discrete values as shown in the Figure 3.

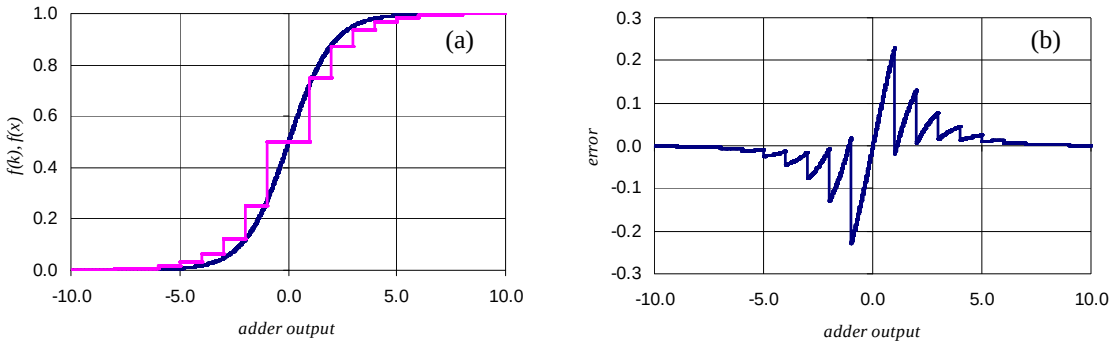

Figure 3. (a) Approximation of the Sigmoid activation function obtained by considering the expressions (5) and (6) and (b) the error with a pure floating point Sigmoid.

Obviously approximating the sigmoid function only by using the (5) as it is possible to see the Figure 3, the ANN does not work in correct manner because the great quantization error of the activation function, in particular, in the interval [-5,+5] of the adder output.
In the second process is considered the following polynomial expression:

$$y_p(x,s) = a \cdot x^2 + s \cdot b \cdot x + c \qquad (7)$$

with $a = 0.35$ $b = s \cdot 0.65$ and $c = 1.0$;
The expression (7) is very simple to implement, because it is based only on the usage of adders and multipliers operators. This approach is more efficient and less complex, at same time, it gives a higher accuracy than the PWL method mentioned in section 1.
The (7) can be implemented on any capable FPGA device using the developed *sfloat24* math library (W. Kahan, 1996) (F. Parillo, 2012a) (F. Parillo, 2012b).
It's a periodic function, with first kind discontinuities, in the interval $[-\infty;0]$ and $[0;+\infty]$ respectively and symmetrical respect the origin, as shown in Figure 4.

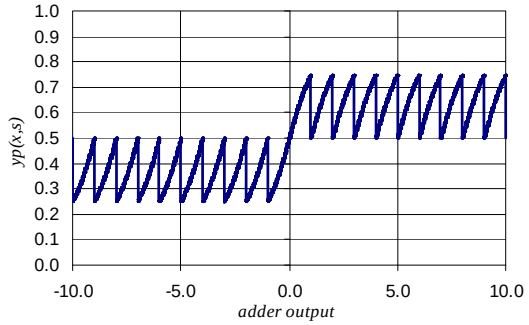

Figure 4. Graph of the expression (7) in function of the adder output.

A last the *sfloat24* approximation of Log-Sigmoid function can be obtained as:

$$f(x) = 0.5 \cdot \left(1 + s - \frac{s}{y(x)}\right) \qquad (8)$$

with: $y(x) = y_i(k,s) \cdot y_p(x,s) = 2^{s \cdot x}$.

3. FPGA Implementation – Simulation results –

It is possible, on the basis of expressions (5) and (7), to build the sigmoid function. The code of the above mentioned expression has been written using the VHDL syntax.
The code has been implemented using the ALTERA® Quartus II 9.1 development tool. The *sfloat24* log-sigmoid activation function occupies only 13% of total logic elements of the used Cyclone® III EP3C25F324C8 device and occupying less than 1% of the dedicated logic registers.
The VHDL code has been written by considering two parallel processes running concurrently on the basis of the analytical approach described in the section 2 as shown in the Figure 5.

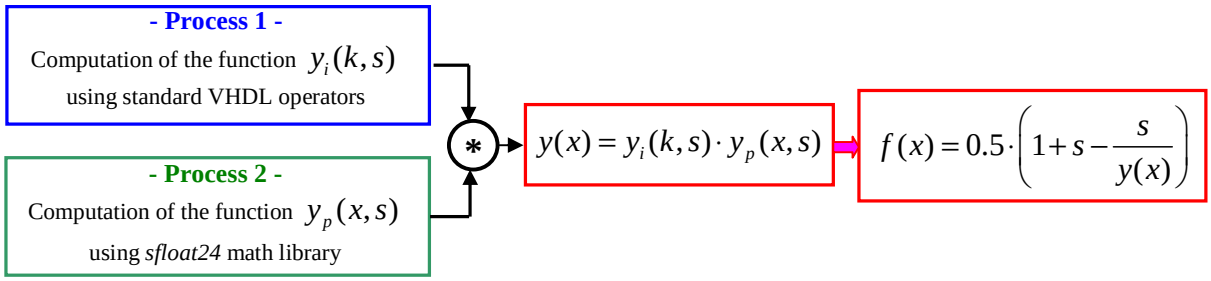

Figure 5. Operating principle of the proposed algorithm.

In the following is shown the obtained Sigmoid function compared to the same function built with Simulink® blocks. It is important to underline that the Matlab® operates with double precision floating (64 bit) point numbers.
The simulation results when the sampling time T_s has been fixed to a 2.5 µS, show that the output error varies in the range ±0.0025 about (Figure 6). Other simulations have been performed with different sampling times. In all the tests results that the entire system has a latency time at maximum of 6 clock cycles. Improvement of the overall performances could be reached by using a sample time less than 2.5 µS. For a common industrial/decisional application the used sample time produces already good results.

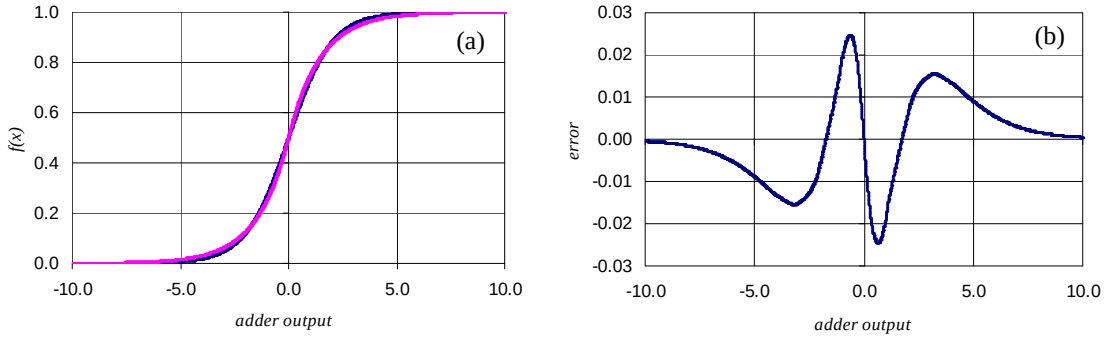

Figure 6. Simulation results – (a) Sigmoid activation function and (b) the absolute error between Sigmoid generated by the *sfloat24* math library and one generated by Simulink® math function block.

The system presents a stable performance comparing to the any external disturbance. In fact, $FPGA_S$ offer the stability of a typical digital implementation on a standard microprocessor and performances comparable to an analog implementation. Due to the low device logic elements occupation, on the used FPGA evaluation board it is possible to implemented a full ANN capable to accommodating a complex Bank Credit Risk Management system, as depicted in (M.C. Miglionico, F. Parillo, 2012), where as example of a decision system credit has been considered the situation of a current account holder that requires a loan at proper credit institute with its risk evaluation has been taken into consideration.
The risk is represented by the variable *R*. The variable x_0 indicates whether the current account holder has got (value = 1) a real estate property or hasn't (value = -1). The current account holder has got a mortgage loan (x_1 = 1) or hasn't (x_1 = -1), or (x_1 = 0) if he does not own any property. The variable x_2 represents the availability of a profit (value = 1) or not (value = -1) if the profit is non-existent or insignificant (x_2 = 0). The loan applicant has a good behaviour (x_3 = 1), middle (x_3 = 0) or bad (x_3 =-1) with the credit institute. To solve this problem the ANN was constituted only by 3 neurons in the hidden layer and 1 neuron in the output layer, the maximum error, difference between desired output and the actual ones, reached is 0.1%.
To examine more complex situations it is necessary to implement ANN_S with a major number of neurons. With the used low cost FPGA device this operation is not possible, because it is not capable to accommodate other neurons.

4. Conclusions

The obtained results show the validity and the feasibility of implementing a Sigmoid AF using both the custom *sfloat24* math library and the proposed analytical approach. On the used FPGA device is possible to implement also a full ANN capable to solve a Bank Credit Risk Management problem or other complex decisional problem. Furthermore, the speed or execution or latency of the ANN can be precisely controlled with the amount of reuse of *sfloat24* arithmetic elements. The proposed analytical approach to implement a Sigmoid function, with few considerations gives the possibility to add also new math operators in the *sfloat24* math library, for example, e^{-x} and e^{x} functions.
Using a more complex FPGA device it is possible to implement a network with an interesting number of neurons working in parallel using only a single chip.
The implemented sigmoid module is highly accurate and can easily be reconfigured for any sigmoid slope and any desired error tolerance.

REFERENCES

Saichand, V.; Nirmala, D.M.; Arumugam, S.; Mohankumar, N. (2008). FPGA Realization of Activation Function for Artificial Neural Networks. *Eighth International Conference on Intelligent Systems Design and Applications*, ISDA 2008, Vol. 3, pp. 159 - 164.

Namin, A.H.; Leboeuf, K.; Muscedere, R.; Huapeng Wu; Ahmadi, M. (2009). Efficient hardware implementation of the hyperbolic tangent sigmoid function, *IEEE International Symposium on Circuits and Systems*, ISCAS 2009, pp. 2117-2120.

Khodabandehloo, G.; Mirhassani, M.; Ahmadi, M. (2012) Analog Implementation of a Novel Resistive-Type Sigmoidal Neuron, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Volume 20, Issue 4, pp. 750-754.

Bezborah, A. (2012). A Hardware Architecture for Training of Artificial Neural Networks Using Particle Swarm Optimization, *Third International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pp. 67-70.

M.C. Miglionico, F. Parillo, (2011a). Modelling a neuron using a custom math library *sfloat24* – Implementation of a sigmoid function on a FPGA device –, *ISHAP Conference Sorrento Italy: 15 – 18 June 2011*, *http://204.202.238.22/isahp2011/dati/autor.html*, *Online Proceedings ISSN 1556-8296*, Proceedings of the International Symposium on the Analytic Hierarchy Process for Multicriteria Decision Making, Publication date: 15 June 2011.

M.C. Miglionico, F. Parillo. (2011b). A BP Neural Network Application in Bank Credit Risk Management System using a sfloat24 custom math library – FPGA implementation –, *A.M.A.S.E.S. Meeting, XXXV Edition*, September 15-17 2011, Pisa, ITALY.

W. Kahan. (1996). Lecture notes on the Status of IEEE Standard 754 for Binary Floating Point Arithmetic. *Electrical Engineering and Computer Science* – University of California Berkeley CA 94720-1776, 31 May 1996.

F. Parillo. (2012a). *sfloat24* Converter Tool version 1.1, Software developed under NI® LabView, publication date: 05 June 2012, link: https://decibel.ni.com/content/docs/DOC-22721

F. Parillo. (2012b). Dual Boost High Performances Power Factor Correction (PFC) control strategy implemented on a low cost FPGA device, using a custom *sfloat24* developed math library, *IEEE International 47th Universities Power Engineering Conference (UPEC 2012)*, Brunel University of London, 04 - 07 September 2012, Digital Object Identifier: 10.1109/UPEC.2012.6398654, Publication Year: 2012 , pp. 1 - 6

M.C. Miglionico, F. Parillo. (2012). An Application in Bank Credit Risk Management System employing A BP Neural Network based on *sfloat24* custom math library using a low cost FPGA device, *Advances in Computational Intelligence Communications in Computer and Information Science, Springer magazine*, Vol. 300, pp. 84-93, link: http://rd.springer.com/chapter/10.1007/978-3-642-31724-8_10.