ISSN 2347 - 3983

Volume 8. No. 5, May 2020

International Journal of Emerging Trends in Engineering Research

Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter16852020.pdf

https://doi.org/10.30534/ijeter/2020/16852020

A Real Time Linearization of NTC Thermistor using Hybrid Neuro-Fuzzy Logic based on VLSI Technology

Rajesh T. Jadhav [1], **Dr.Vikram S. Patil** [2]

[1] Research Scholar, Shivaji University, Kolhapur, Maharashtra, India, rajeshtukaramjadhav@gmail.com

[2] Director, ADCET, Ashta, & Research Guide, Shivaji University, Kolhapur, Maharashtra, India, vikrams.patil@gmail.com

ABSTRACT

Nonlinear sensors and digital solutions are used in many data acquisition system designs. As the input-output characteristic of most sensors is nonlinear in nature. Hence obtaining data from a nonlinear sensor by using an optimized device has always been a design challenge. Linearization of non linear sensor in digital environment is a vital step in the instrument signal conditioning process.This paper proposes a real time implementation of Hybrid Neuro Fuzzy Logic (HNFL) in Field Programmable Gate Array (FPGA) to linearize the natural non linear characteristics of NTC thermistor. Linearization is achieved by using Adaptive Neuro Fuzzy Inference System (ANFIS) based on Takasi-Sugeno-Kang (TSK) fuzzy inference system (FIS) whose membership functions parameters are adjusted using back propagation and/or gradient algorithms. The network training is carried out in Matlab for obtaining the optimized membership functions parameters for ANFIS implemented on a Waxwing Spartan 6 FPGA Development Board using VHDL. Single precision floating point arithmetic subroutines are developed in IEEE-754 format. Graphical programming language is used for simulation, real time data acquisition and storage.

Key words : ANFIS, FPGA, Sensor Linearization, VLSI.

1. INTRODUCTION

Sensors are the fundamental elements which are used in most of the measurement circuits to monitor the physical quantity (temperature, pressure, etc) or to give feedback signals to the control unit. Low-cost sensors with high sensitivity and resolution, with linear characteristics are required [1]. Generally Sensors gives analog output, which may sometimes shows nonlinear behaviour. This is due to natural nonlinear characteristic of sensor itself, dynamic nature of the environment, inherent sensor's noise, aging and data loss due to transients or intermittent faults [2]. It is essential to have linear characteristics of the sensor as it will improve the system performance [3]. Linearization of this nonlinear behavior of sensors has always been a designed challenge. Linearization of nonlinear sensor in the digital environment is a vital step in the instrument signal conditioning process [4]. Several linearization techniques have been mentioned in many research works. These techniques are classified into three main classes. Analog hardware-based linearization circuit [5-6], software based linearization algorithms [7-11] and hybrid analog to digital conversion solutions [12].

Analog circuits are frequently used for improving the linearity of sensor characteristics, which implies additional analog hardware and typical problems particular to analog circuit such as temperature drift, gain and offset errors. Using the second technique, sensor nonlinearities can be compensated by means of arithmetic operations, if an accurate sensor model is available (direct computation of the polynomials), otherwise with the use of multidimensional look-up tables. Direct computation of the polynomial method is more accurate but requires a longer time for computation, while the look-up table method, though faster, is not very accurate [9-12]. The third technique is performed by interfacing a passive or an active nonlinear analog circuit between the sensor and an analog to digital converter (ADC) [13].

Linearization of non-linear sensors characteristics is often a quite complex and computationally intensive task. Hence Neural Networks and Fuzzy Systems which are two branches of artificial intelligence are gaining widespread acceptance in the field of learning and intelligent control [14-15]. This is mainly due to their intrinsic parallelism, their learning and adaptation capabilities and, to some extent, also to their increased fault tolerance. Fuzzy control and Neural Network control have many advantages as above, but fuzzy control also has a drawback that you have to set new control laws and membership functions every time types of system change even after you set control laws and membership functions. And neural network has a drawback that while learning, it can easily fail onto local minimum instead of global minimum, and it take much time to make as many neurons learn as how complicated the system. In order to make up for the defects, research on integration of neural network and fuzzy logic that is Hybrid Neuro-Fuzzy Logic (HNFL) is under way. A proposed HNFL is an intelligent system that combines qualitative knowledge of symbolic fuzzy rules and learning capabilities of neural networks.

Recently, application of neural networks and fuzzy logic techniques has emerged as a promising area of research in the field of instrumentation and measurement [16-19]. Neuro-Fuzzy system's modeling capability was demonstrated by S. N. Engin et al. [20]. It is shown that the ANFIS can modelize a nonlinear system very accurately by means of data taken from mathematical model [21].

FPGAs belong to the wide family of programmable logic component [22], their densities are now exceeding 10 million gates [23]. FPGAs can be defined as a matrix of configurable logic blocks (combinatorial and/or sequential), linked to each other's by an interconnection network that is also entirely reprogrammable. FPGAs technology allows developing specific hardware architectures within a flexible programmable environment. This specificity of FPGA gives the designer a new degree of freedom comparing to microprocessors implementation, since the hardware architecture of the synthesized system is not imposed a priori. Motivated by reducing the complexity and the cost of the measurement system, a new architecture based ANFIS is proposed to synthesize the linearization function of the measurement chain. The ANFIS system model is obtained using characteristics data of sensor. As an example, a negative coefficient temperature (NTC) thermistor sensor is taken, and any nonlinear sensor can be used in this work.

The obtained circuit is realized on Waxwing Spartan 6 FPGA Development Board from Numato Lab. Moreover, the developed device can be easily modified, implemented and used in many applications of instrumentation and control. An analysis and test of the implemented architecture show that the device does an accurate linearization.

2. ACTUAL SYSTEM DESCRIPTION

The actual system block diagram is shown in Figure 1 and Figure 2 shows the actual implemented system. The NTC thermistor, 1K resistor and +5V voltage source are connected in series and this configuration is a voltage divider circuit. The non linear analog voltage across 1K resistor is given to the ADC (MCP 3202). Further the digital output of ADC is given to the digital device named Waxwing Spartan 6 FPGA (XC6SLX45) Development Board. An ANFIS is implemented in FPGA. The ANFIS takes the digital output (in non linear form) from ADC and processes it and gives a linearize digital signal, which is further given to DAC (MCP 4921) and personal computer. Subroutines were developed in VHDL code and then implemented in Waxwing Spartan 6 FPGA Development Board, so that digital device FPGA can communicate with ADC, DAC and personal computer. In personal computer, graphical programming language such as LabVIEW is used for simulation and real time data acquisition and storage.

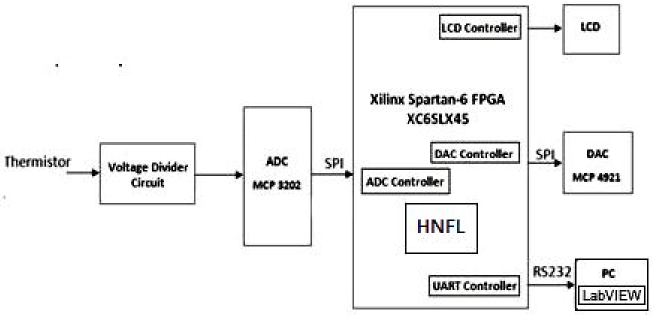


Figure 1: Actual System Block Diagram

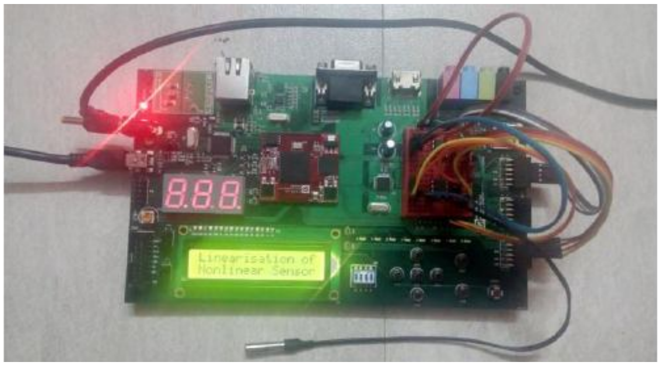


Figure 2: Actual System

The schematic of voltage divider circuit is shown in Figure 3.

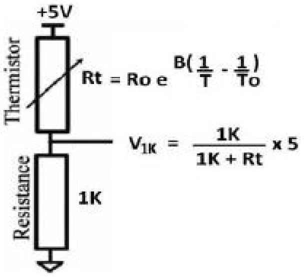


Figure 3: Schematic of voltage divider circuit

The thermistor used in a voltage divider circuit is a NTC Thermistor whose resistance R_T at temperature T can be modeled by

$$R_T = R_O \text{ e} xp\left[\beta\left(\frac{1}{T} - \frac{1}{T_O}\right)\right] \tag{1}$$

where the NTC thermistor used in this work has R_O = 10,000 ohms, is the resistance at a reference temperature T_O = 298 K (25 °C) and β = 3950, with a tolerance of ±10%.

3. ANFIS LINEARIZER MODELIZATION

3.1 Elements of ANFIS Architecture

As ANFIS is going to be hardware implemented, hence it is necessary to have a detailed knowledge of its architecture. Figure 4 shows the ANFIS architecture wherein the square nodes denotes the functions with parameters to be learnt whereas circular nodes represent fixed operations.

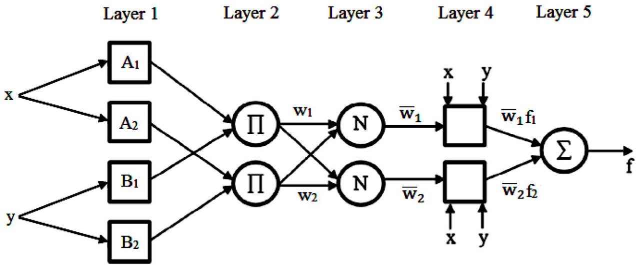


Figure 4: ANFIS Architecture

If x is A_1 and y is B_1 then according to Sugeno rule form
$f_1 = p_1x + q_1y + r_1$
(2)

Here inputs x and y represents premise variables of the fuzzy rule. A_1, B_1 represents premise parameters and p_1, q_1 and r_1 represents consequence parameters. Using f_1 and f_2 functions and w_1 and w_2 weight values, the final output function F is given by (3).

$$F = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \overline{w_1} f_1 + \overline{w_2} f_2 \qquad (3)$$

With reference to Fig. 5, ANFIS Linearizer shows five layers [21].
Layer 1: Every node is adaptive in this layer. Here fuzzification process takes place. Output of each node is given by (4).

$$O_{1,i} = \mu_{A_i}(x) \qquad for\, i = 1,2 \qquad (4)$$
$$O_{1,i} = \mu_{B_{i-2}}(x) \qquad for\ i = 3,4$$

Thus $O_{1,i}(x)$ represents membership grade for inputs x and y. The membership functions could be trapezoidal, triangular or any other type.
Layer 2: In this layer, nodes are fixed and output of each node is given by (5) which represents a weight of the rule.

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y) \qquad for\, i = 1,2 \qquad (5)$$

Layer 3: In this layer, nodes are fixed. The ratio of the ith rule's firing weight to the sum of all rules weights is computed and is given by (6).

$$O_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2} \qquad (6)$$

Layer 4: In this layer, nodes operate as a function block, whose variables represents the input values and parameters are adaptive. Overall output (TSK output) of this layer is given by (7).

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i}\left(p_i x + q_i y + r_i\right) \qquad (7)$$

Here p_i, q_i and r_i denotes consequent parameters to be determined.
Layer 5: Output of this layer is the summation of all the input signals. The final output is given by (8).

$$O_{5,i} = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (8)$$

Classically, the characteristic of a nonlinear sensor can be linearized using analog or digital electronic circuit [25]. The first method, which is based on using logarithmic operational amplifier (which correspond to the inverse of sensor's characteristic), is very fast, but it gives good results only if the measure signal depends on measurand, and/or disturbance variables are time invariant [26]. In other words, if the sensor has a great sensibility to noise, it is difficult to linearize using analog electronic circuit. The second methods are more accurate and flexible but they need many clock cycles to give output, as a result, they are ineffective against applications that need very fast information treatment [25]. Logarithmic interpolation is done on the sensor's inverse characteristic with two parameter logarithmic function corresponding to the inverse of (1). With reference to Figure 1, ANFIS architecture has single input and single output. The ANFIS architecture has to learn sensor's inverse characteristic. Table 1 represents learning phase results for different approaches. The error between the ANFIS output and sensor's inverse characteristic output represents mean square errors (MSE).

In this work, ANFIS architecture is going to be hardware implemented. Analysis of the learning phase results presented in Table 1 guide us to choose the method highlighted in gray; two input triangle membership functions and two linear output membership functions with three parameters each one.

Table 1: Learning Phase Results

Input Membership		Training Method	Output Membership Type	Error	Epoch
Type	Number				
Triangle	2	Hybrid	Constant	0.089562	500
			Linear	0.042277	200
		Back Propagation	Constant	0.10639	500
			Linear	0.20319	500
	3	Hybrid	Constant	0.065469	250
			Linear	0.069862	50
		Back Propagation	Constant	0.073769	500
			Linear	0.066717	500
	4	Hybrid	Constant	0.028184	100
			Linear	0.039662	20
		Back Propagation	Constant	0.035134	500
			Linear	0.031191	500
Trapeze	2	Hybrid	Constant	0.052064	300
			Linear	0.043712	200
		Back Propagation	Constant	0.051347	500
			Linear	0.14182	500
	3	Hybrid	Constant	0.043692	200
			Linear	0.018114	200
		Back Propagation	Constant	0.044042	500
			Linear	0.040508	500
	4	Hybrid	Constant	0.032136	200
			Linear	0.0081451	220
		Back Propagation	Constant	0.034315	500
			Linear	0.029078	500
Bell Shape	2	Hybrid	Constant	0.068148	500
			Linear	0.029257	500
		Back Propagation	Constant	0.11602	500
			Linear	0.17479	500
	3	Hybrid	Constant	0.014575	500
			Linear	0.0094959	230
		Back Propagation	Constant	0.066668	500
			Linear	0.032527	500
	4	Hybrid	Constant	0.0080987	500
			Linear	0.0068352	120
		Back Propagation	Constant	0.063041	500
			Linear	0.026443	500
Gauss	2	Hybrid	Constant	0.21132	500
			Linear	0.044937	500
		Back Propagation	Constant	0.23609	500
			Linear	0.2041	500
	3	Hybrid	Constant	0.059039	500
			Linear	0.017174	420
		Back Propagation	Constant	0.12191	500
			Linear	0.041409	500
	4	Hybrid	Constant	0.018454	500
			Linear	0.010377	250
		Back Propagation	Constant	0.12445	500
			Linear	0.036323	500
Gauss2	2	Hybrid	Constant	0.074492	500
			Linear	0.027118	500
		Back Propagation	Constant	0.11443	500
			Linear	0.17679	500
	3	Hybrid	Constant	0.03082	500
			Linear	0.016948	100
		Back Propagation	Constant	0.083049	500
			Linear	0.055935	500
	4	Hybrid	Constant	0.0150537	500
			Linear	0.0089628	400
		Back Propagation	Constant	0.062011	500
			Linear	0.030194	500

3.2 Generating Input and Target Data

Input data is generated with the help of data sheet provided by the manufacturer of thermistor. The data sheet provides the values of thermistor resistance with respect to temperature. From these values the input data that is thermistor non linear voltage across 1k ohm resistor is calculated with the help of voltage divider formula The non linear data generated that is V_{1K} is then plotted with respect to temperature by taking the help of third party software. The corresponding plot is shown in Figure 5. Once again by taking the help of third party software, the linear fit is obtained along with the slope and intercepts values. The corresponding linear fit is shown in Figure 6. From this linear fit the target data was generated to train the ANFIS.

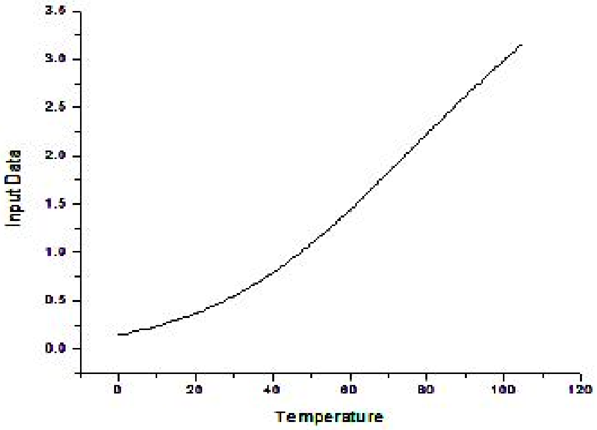


Figure 5: Temperature v/s Input Data Plot

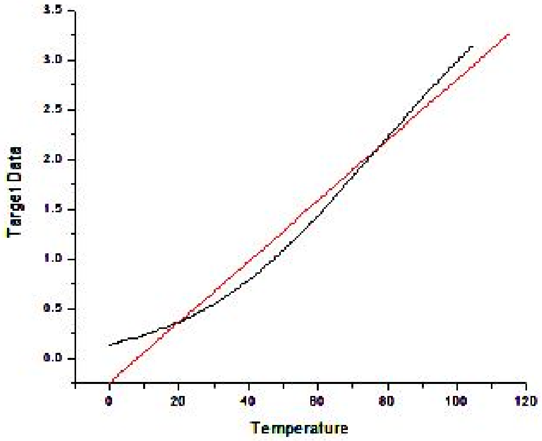


Figure 6: Linear Fit Plot.

3.3 Training ANFIS

For linearization of nonlinear sensor's characteristic, a neuro-fuzzy toolbox from Matlab-Simulink software was used for training the ANFIS linearizer. With the help of given input-output data set of modeled system, ANFIS creates a Takasi-Sugeno-Kang fuzzy inference system (TSK FIS). This makes fuzzy system to learn from data set of modeled system. Table 2 illustrate the parameters obtained in the learning phase of the ANFIS.

Table 2: Parameters for ANFIS architecture

Input Membership			Output Membership		
Tri_1	a	-3.13	f_1	p	0
	b	-35		q	4.5
	c	5.169		r	-0.03
Tri_2	a	0.21	f_2	p	0
	b	3		q	1.225
	c	6.305		r	0.5

4. FPGA DESIGN & IMPLEMENTATION

4.1 Elements of ANFIS Architecture

ANFIS architecture for linearization of nonlinear sensor's characteristic is illustrated by Figure 7.

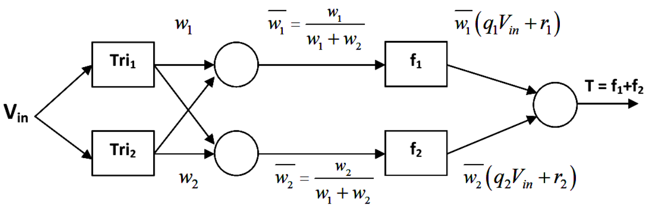

w_1
$\overline{w_1} = \frac{w_1}{w_1+w_2}$
$\overline{w_1}(q_1V_{in}+r_1)$
Tri_1
f_1
V_{in}
$T = f_1+f_2$
Tri_2
f_2
w_2
$\overline{w_2} = \frac{w_2}{w_1+w_2}$
$\overline{w_2}(q_2V_{in}+r_2)$

Figure 7: ANFIS architecture for linearization

With reference to Figure 7, the overall output is given by (9). Equation (10) gives the expression for $Tri_i(x)$ function

.

$$f = \frac{(q_1x+r_1)Tri_1(x)+(q_2x+r_2)Tri_2(x)}{Tri_1(x)+Tri_2(x)} \tag{9}$$

$$Tri_i(x) = \begin{cases} 0 & \text{if } x \le a_i \\ \frac{x-a_i}{b_i-a_i} = \frac{1}{b_i-a_i}x - \frac{a_i}{b_i-a_i} & \text{if } a_i \le x \le b_i \\ \frac{c_i-x}{c_i-b_i} = \frac{-1}{c_i-b_i}x + \frac{c_i}{c_i-b_i} & \text{if } b_i \le x \le c_i \\ 0 & \text{if } c_i \le x \end{cases} \tag{10}$$

Hence, for the implementation of ANFIS linearizer, two circuits each for the expressions given by (9) ($f_1(x)$ and $f_2(x)$) and (10) ($Tri_1(x)$ and $Tri_2(x)$) are required. Here q, r, a, b and c are the parameters obtained in the learning phase of the ANFIS.

4.2 FPGA Implementation of Linearizer

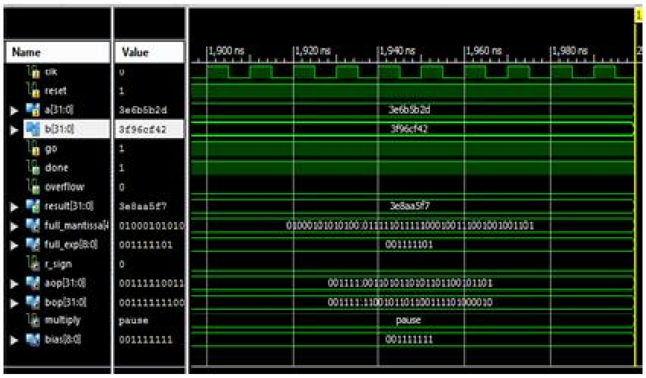
A. FPGA Implementation of Floating Point Arithmetic

In FPGA implementation of ANFIS linearizer, decimal floating point (FP) arithmetic plays an important role. IEEE-754 industry standard is used for FP number presentation. Decimal number represented in this standard consists of a sign bit S, 8 bits of an exponent E, and 23 bits of an unsigned fraction M [24]. Figure 8 shows the same

1 bit *S*	8 bit *Exponent E*	23 bit *Unsigned Fraction M*

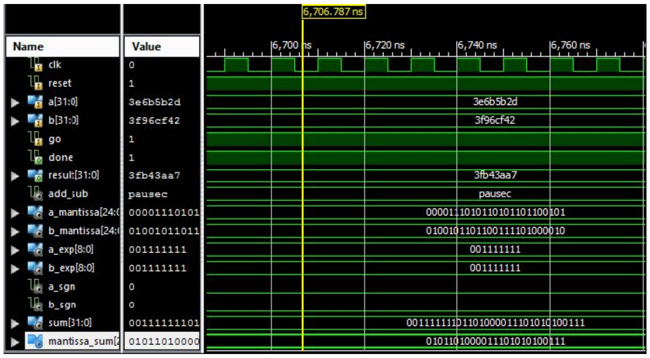
Figure 8: IEEE-754 Floating Point Representation.

The IEEE 32 bit floating adder/substractor, multiplier and divider have been implemented in this work to carry out the 32-bit floating point arithmetic. Test bench result for floating point multiplier and adder are shown in Figures 9 and 10 respectively.



```
Name              Value
clk               0
reset             1
a[31:0]           3e6b5b2d
b[31:0]           3f96cf42
go                1
done              1
overflow          0
result[31:0]      3e8aa5f7
full_mantissa[4   01000101010
full_exp[8:0]     001111101
r_sign            0
aop[31:0]         00111110011
bop[31:0]         00111111100
multiply          pause
bias[8:0]         001111111
```

Figure 9: Test bench result for floating point Multiplier.



```
Name              Value
clk               0
reset             1
a[31:0]           3e6b5b2d
b[31:0]           3f96cf42
go                1
done              1
result[31:0]      3fb43aa7
add_sub           pausec
a_mantissa[24:0   00001110101
b_mantissa[24:0   01001011011
a_exp[8:0]        001111111
b_exp[8:0]        001111111
a_sgn             0
b_sgn             0
sum[31:0]         00111111101
mantissa_sum[2    01011010000
```

Figure 10: Test bench result for floating point Adder.

B. FPGA Implementation of ANFIS

The functional diagram of the proposed FPGA-based ANFIS linearizer is shown in Figure 11 which contains two adder and multiplier circuits along with seven different essential blocks, namely:

- The ROM block for storing parameters of all the membership functions and the pre calculated constants in order to avoid usage of additional digital dividers and adders circuits for optimizing FPGA hardware resources.

- Two blocks (Tri_1 and Tri_2) for input membership functions. Each function needs two multipliers, a substractor and an adder.

- Two blocks (f_1 and f_2) for the output membership functions. Each function needs a multiplier and an adder.

- S^{-1} block for computing 1/S value to S input.

- Control unit block for the correct functioning of the different blocks at the right moments.

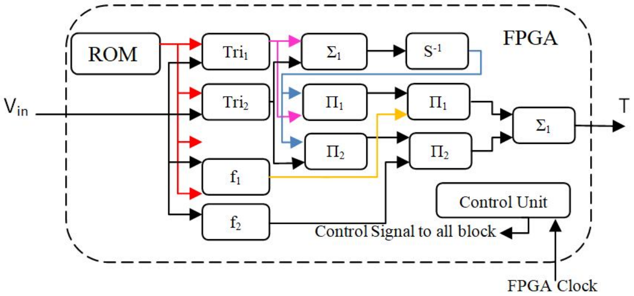


Figure 11: Functional diagram of the digital ANFIS

The following Figure illustrate state machine diagram for control unit.

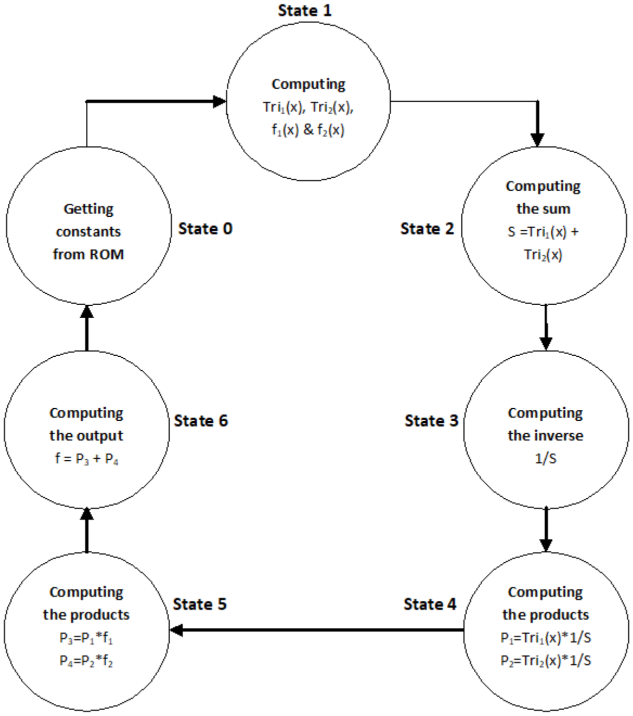

State 1
Computing $Tri_1(x)$, $Tri_2(x)$, $f_1(x)$ & $f_2(x)$
State 0
Getting constants from ROM
State 2
Computing the sum $S = Tri_1(x) + Tri_2(x)$
State 3
Computing the inverse 1/S
State 4
Computing the products $P_1 = Tri_1(x)*1/S$ $P_2 = Tri_2(x)*1/S$
State 5
Computing the products $P_3 = P_1*f_1$ $P_4 = P_2*f_2$
State 6
Computing the output $f = P_3 + P_4$

Figure 12: State Machine Diagram

Table 3 illustrate Xilinx Synthesis Report summary for ANFIS.

Table 3: Xilinx Synthesis Report

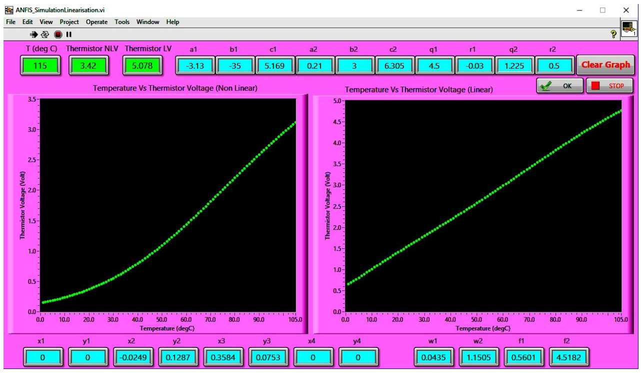
Device Utilization Summary			
Slice Logic Utilization	**Used**	**Availabl e**	**Utilizatio n**
Number of Slice Registers	4259	54576	7%
Number of LUTs	251	27,288	19%
Number of occupied Slices	1734	6,822	25%
Number of MUXCYs	2992	13,644	21%
Number of bounded IOBs	26	218	11%
Number of DSP48A1s	0	58	0%

6. SOFTWARE IMPLEMENTATION

The ANFIS simulation and real time data acquisition software was developed in user friendly and graphical programming language known as Lab VIEW (Evaluation Copy) and ANFIS code was developed in VHDL.

7. RESULTS AND DISCUSSION

FPGA based thermistor signal linearizer has been developed by using HNFL with minimum resources and without much loss in speed. Figure 13 shows simulation results for one set of input and output membership functions. Figure 14 shows the real time linearization of thermistor characteristics. The results obtained from the FPGA validate the LabVIEW simulation.

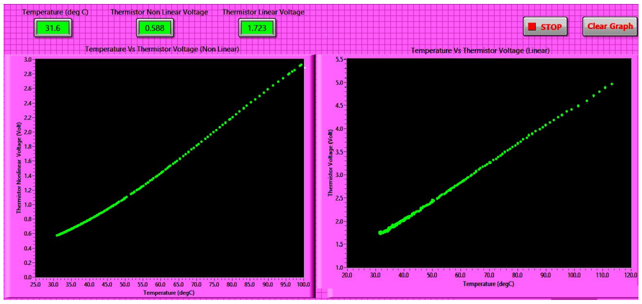

ANFIS_SimulationLinearisation.vi

T (deg C)	Thermistor NLV	Thermistor LV	a1	b1	c1	a2	b2	c2	q1	r1	q2	r2
115	3.42	5.078	-3.13	-35	5.169	0.21	3	6.305	4.5	-0.03	1.225	0.5

x1	y1	x2	y2	x3	y3	x4	y4	w1	w2	f1	f2
0	0	-0.0249	0.1287	0.3584	0.0753	0	0	0.0435	1.1505	0.5601	4.5182

File Edit View Project Operate Tools Window Help
Clear Graph
OK
STOP
Temperature Vs Thermistor Voltage (Non Linear)
Temperature Vs Thermistor Voltage (Linear)
Thermistor Voltage (Volt)
Temperature (degC)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
105.0

Figure 13: Front panel of simulation software



Temperature (deg C)	Thermistor Non Linear Voltage	Thermistor Linear Voltage
31.6	0.588	1.723

Figure 14: *Front* panel of real time data acquisition software

The values extracted from the saved data files are used to trace the graphs of Figure 15 and Figure 16.

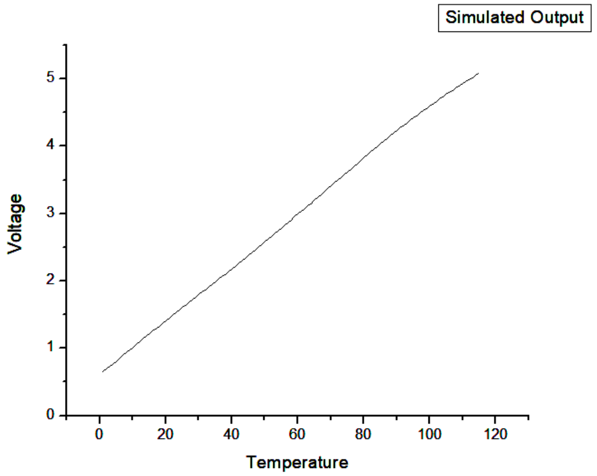


Figure 15: Simulated Linearized Output

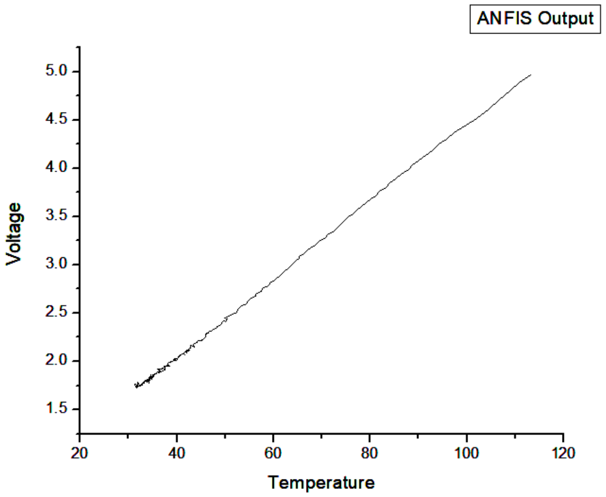


Figure 16: Hardware ANFIS Linearized Output

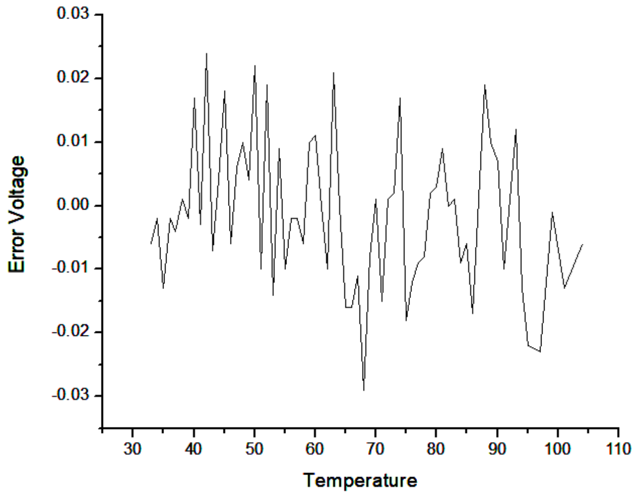


Figure 17: Error between simulated and designed linearizer

Figures 15 and 16 shows the simulated and hardware ANFIS model linearized output respectively. Figure 17 shows the error between the simulated model and hardware ANFIS model. These graphs shows a maximum error in absolute value of 2.8 $\times 10^{-2}$ °C between the simulated model and ANFIS FPGA model.

7. CONCLUSION

The proposed design for ANFIS linearizer is very much useful in sensor applications; it is possible to implement sensors with linearized output digital code. This solution appears to be of lower cost and suitable for VLSI integration, with or without the sensor.

Future works include implementation of hardware co-simulation of some smart features like linearization of nonlinear sensor, auto calibration of sensor, sensor drift compensation and sensor fault detection on Xilinx's FPGA using Xilinx's System Generator tool which helps in the evaluation, testing and validation of a new algorithm, a new component or a new prototype without damaging the actual system.

REFERENCES

[1] G. Bucci, M. Faccio and C. Landi, New ADC with Piecewise Linear Characteristic: Case Study – Implementation of a Smart Humidity Sensor, IEEE Trans, Instrumentation and Measurement, vol. 49, No. 6, pp. 1154-1166, Dec. 2000. https://doi.org/10.1109/19.893250

[2] N. N. Charniya, Some Features of Neural Networks based Intelligent Sensors and Design Issues, International Journal of Computer Applications, vol. ICCIA, No.2, pp. 1-4, March 2012.

[3] Alaa Abdul Hussein Salman , Fadhil Rahma Tahir, Mofeed Turky Rashid, ―Design and implementation model for linearization sensor characteristic by FPAA, Iraq J. Electrical and Electronic Engineering, Vol. 11 No.2, 2015.

[4] C. Alippi, A. Ferrero and V. Piuri, ―Artificial intelligence for instruments and measurement applications‖, IEEE Instrumentation and Measurement Magazine, vol. 1, No. 1, pp. 9-17, March 1998. https://doi.org/10.1109/5289.685492

[5] M. Bouhedda and M.Attari, "Synthesis and FPGA–implementation based neural technique of a nonlinear ADC model," *International Scientific Journal of Computing (ISJC)*, ISSN:1727-6209, Vol.4, Issue 1, August 2005, pp. 27-33.

[6] C. Renneberg, "Analog circuits for thermistor linearization with Chebyshev-optimal linearity error," *18th European Conference on Circuit Theory and Design, ECCTD 2007*, Sevilla (Spain), 27-30 Aug. 2007, pp 910–913.

[7] Hamit Erdem, "Implementation of software-based sensor linearization algorithms on low-cost microcontrollers," *ISA Transactions*, Volume 49, Issue 4, October 2010, pp. 552–558. https://doi.org/10.1016/j.isatra.2010.04.004

[8] L.R. Klopfenstein, "Software linearization techniques for thermocouples, thermistors, and Rtds," *ISA Trans.*, Vol. 33, 1994, pp. 293-305.

[9] D.K. Anvekar, B.S. Sond, "Transducer output signal processing using dual and triple microprocessor systems," *IEEE Trans. Instrum. Meas.*, Vol. 38, 1989, pp. 834-836. https://doi.org/10.1109/19.32204

[10] P.N. Mahana, F.N. Trofimenkoff, "Transducer output signal proceesing using an eight-bit microcomputer," *IEEE Trans. Instrum. Meas.*, Vol. 35, 1986, pp. 182-186.

[11] A. Flammini, D. Marioli, A. Taroni, "Transducer output signal processing using an optimal look-up table in microcontrollerbased systems," *Electron. Lett.*, Vol. 33, 1997, pp 1197-1198.

[12] D. Patranabis, S. Gosh, C. Bakshi, "Linearizing Transduser characteristics," *IEEE Trans. Instrum. Meas.*, Vol. 37, 1988, pp. 66-69.

[13] M. Bouhedda, M.Attari and B. Granado, "Hybrid Analog FPGA Implementation of Neural Nonlinear ADC For Instrument Applications," *The Mediterranean Journal of Measurement and Control*, SoftMotor Ltd, ISSN:1743-9310, United Kingdom (UK), Volume 3, No. 4, October 2007, pp. 164-172.

[14] Aaron Don M., Patrick Bernard T., Arsenic S. Publico, Mharela Angela A. Tan A, "Fuzzy Neural Control System" International Journal of Emerging Trends in Engineering Research, Vol. 7.9 pp. 323-327, 2019. https://doi.org/10.30534/ijeter/2019/15792019

[15] Aaron Don M., Patrick Bernard T., Arsenic S. Publico, Mharela Angela A. Tan A, "Fuzzy Logic Control System with Gaussian Membership Functions" International Journal of Emerging Trends in Engineering Research, Vol. 7.9 pp. 328-332, 2019. https://doi.org/10.30534/ijeter/2019/16792019

[16] M. Attari, F. Boudjema, M. Heniche, "Linearizing a Thermistor Characteristic in the Range of Zero to 100°C With Two Layers Artificial Neural Network," *IEEE Instrumentation and Measurement Technology Conference*, IMTC/95, Waltham, Massachusetts (USA), April. 1995, pp. 119-122.

[17] M. Attari, F. Boudjema, M. Heniche, "An Artificial Neural Network to linearize a G (Tungsten vs. Tungsten 26% Rhenium) Thermocouple Characteristic in the Range of Zero to 2000°C," *IEEE International Symposium on Industrial Electronics*, ISIE/95, Athens (Greece), Vol.1, July 1995, pp. 176-180.

[18] M.R.G. Meireles, P.E.M. Almedia, M.G. Simoes, "A comprehensive review for industrial applicability of artificial neural networks," *IEEE Trans. Ind. Electon*, Vol. 50, 2003, pp. 585-601.

[19] P. Daponte, D. Grimaldi, "Artificial neural networks in measurement," *Measurement*, Vol. 23, 1998, pp 93-115. https://doi.org/10.1016/S0263-2241(98)00013-X

[20] S. N. Engin, J. Kuvulmaz and V. E. Ömurlü, "Fuzzy control of an ANFIS model representing a nonlinear liquid-level system," *Neural Computing & Applications*, September 2004, Volume 13, Issue 3, pp 202-210.

[21] Jyh-Shing Roger Jang, "ANFIS: adpative network based inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, N°3, May/June 1993. https://doi.org/10.1109/21.256541

[22] A. Volnei Pedroni, *Circuit design with VHDL*, MIT Press, 2004.

[23] Xilinx Data book, available on line: www.xilinx.com, last accessed date: 01/18/2013.

[24] IEEE 754-2008, IEEE Standard for Floating-Point Arithmetic, 2008.

[25] G. Asch, *Capteurs en Instrumentation Industrielle*, seventh ed., Dunod, Paris, 2010.

[26] P. Horowitz and W. Hill, *The art of Electronics*, second ed., Cambridge University Press, 1989.