Neural net implementation of steam properties on FPGA

R. V. S. Krishna Dutt[1], R. Ganesh[2], P. Premchand[3]
[1]Department of Computer Science and Engineering, CVR College of Engineering, Hyderabad, India
[2]Deptartment of Electronics and Communication Engineering, CVR College of Engineering, Hyderabad, India
[3]Department of Computer Science and Engineering, Osmania College of Engineering, Hyderabad, India

Article Info

Article history:

Received Jun 2, 2021
Revised Jul 27, 2021
Accepted Aug 11, 2021

Keywords:

Artificial neural network
FPGA
IEE754 floating point
Thermodynamic properties
Verilog HDL
Xilinx ISE

ABSTRACT

Real time applications like model predictive control, monitoring and data reconciliation of power plants and industrial processes employ nonlinear mathematical models and require thermodynamic properties and their derivatives of working fluids. Applications like super heater temperature control based on energy balance and real time data reconciliation, require an efficient and a compact method for simultaneous estimation of thermodynamic properties, and their partial derivatives suitable for implementation in field-programmable gate array (FPGA). However, the complex mathematical formulations of these properties prohibit direct implementations in FPGAs. Single artificial neural network (ANN) architecture is used to replace the entire code in higher level languages, running into a few thousand lines. FPGA implementation of a compact neural network for the entire range of thermodynamic properties is presented. Large arguments in sigmoid function are factored into a product of integer and a fractional part which is represented using series approximation with five terms only and the integers are represented in look up table (LUT). This ensures optimum storage and computational burden for the above applications. The ANN is implemented in IEEE 754 floating point with synthesis in Xilinx ISE design suite using Verilog HDL. The results are presented for a typical pressure versus saturation temperature.

Corresponding Author:

R. V. S. Krishna Dutt
Department of Computer Science and Engineering, CVR College of Engineering
Vastunagar, Ibrahimptnam, Hyderabad, India
Email: Krishnadutt.rvs@cvr.ac.in

1. INTRODUCTION

Recent trends in power plant and industrial applications indicate developments like data reconciliation (DR), Guo *et al.* [1] model predictive control (MPC), Wang *et al.* [2] and optimization, Niegodajew *et al.* [3] to improve performance of plants and processes. They are based on the first principle thermodynamic formulations as well complex mathematical models. They are limited to offline simulations or implemented in standalone desktop systems. Direct implementation of these algorithms in microcontrollers and field-programmable gate arrays (FPGAs), often encountered in power and process plant distributed control systems (DCS), is difficult and computationally inefficient. One of the constraints is efficient implementation of thermodynamic properties of the working fluids like water, steam and gasses, in the FPGAs. The applications require real time prediction of thermodynamic properties like enthalpy, along with their first and second order partial derivatives. The properties are represented by nonlinear Gibbs free energy formulations [4]. These functions are implemented in higher level languages running into 5000-7000 lines of code. These are evaluated iteratively. They are repeatedly called in the applications and are not amenable for implementation in microcontrollers or FPGAs. Additionally, their partial derivatives are computed numerically by repeated calls to these functions. This makes them unsuitable for real time computations in FPGAs. Recent research is focused on developing application specific properties of water and steam covering sub-critical and super-critical regions implemented in Modelon's Modelica thermal power library and Spline-Based table look-up method, [5], [6]. However, many of these efforts are not directly suited for implementation in FPGAs used in the commercial DSC platform. Cardaliaguet and Euvrard [7] reported artificial neural networks (ANNs) with explicit weights to approximate a function and its derivatives and highlighted usage for control. Hashem and Schmeiser [8] showed that a linear combination of a set of feed forward ANNs trained to approximate a function and its derivatives improve accuracy as against a single ANN. While these lay the foundation for using ANNs for approximating a function and its derivatives, Krishnadutt and Krishnaiah [9] demonstrated the power of a few simple ANNs to simultaneously approximate steam properties and the corresponding derivatives of entire steam properties; thus replacing thousands of lines of code in higher language. These ANNs approximate the complex steam properties to the required degree of accuracy for real time applications.

In spite of the availability of many high-level language tools, the progress in implementing ANNs in FPGA as seen in literature [10]-[15] is slow. Recent surveys [16]-[18] highlight that FPGA based neural network accelerators are preferred over the application specific integrated circuits (ASICs) due to their energy efficiency with a small footprint, but many problems and challenges need to be addressed. These bring out the challenges in training deep neural networks (DNNs) with FPGA and propose a performance metric and evaluation workflow to compare the FPGA-based systems for DNN training specific to computer vision tasks. However, for a pretrained network, only device utilization and energy efficiency may be considered. Reviews highlight major issues like data representation, implementation of inner product between weight matrix and previous layer outputs, activation functions, device utilization, energy efficiency, for different applications.

Non-linear activation function is the main concern of many researchers while implementing an ANN in FPGA. Piazza *et al.* [19] reported adaptable look up table (LUT)-based activation functions for neurons, in learning by backward difference with different learning rates and two look up tables, one for weights and the for coefficients respectively, whereas Bieu *et al.* [20] computed the sigmoid function and its derivative in digital hardware by a sum of steps resulting in area-efficiency. Reconfiguration capabilities of the Atmel FPGA are exploited by Lysight *et al.* [21] for implementing larger ANN with individual layers of the network with time multiplexing on the logic array at the cost of system performance. Tisan *et al.* [22] investigated different approximating functions from the point of view of hardware resource utilization and induced errors and concluded that piecewise linear approximation of the activation function is the best. Problems encountered in implementing an ANN in VHDL are reported by P. Dondon *et al.* [23] wherein sigmoid activation is approximated by sampling of Logsig function with argument between 0 to 1. Ngah *et al.* [24] used combination of second order non-linear function (SONF) and differential LUT for implementing an ANN. The two-step approach is reported to have an improved accuracy that is 10 times better than that of using only SONF and twice better than just using LUT. Li *et al.* [25] implemented a neuron block with sigmoid function using the CORDIC algorithm.

Some of the applications of ANN implementation on FPGA include classification of the region of pixels *i.e.*, hand regions by Krips *et al.* [26] using three inputs representing RGB values with a single hidden layer and one output with data represented by integers and weights scaled up and rounded off to the nearest integer. Recognition of digits using a network with 300 inputs and 10 outputs with a single neuron is reported by Latino *et al.* [27]. A method of configurable MLP with a single neuron block with floating point add and multiply units along with activation function as LUT for a smart position sensor of solar panels has been studied by Dąbrowski *et al.* [28]. A neural classifier with fixed point representation and 12-bits for detecting damaged toothed gears using vibroacoustic signals is highlighted by Polat and Yildirim [29].

These studies consider sigmoid function with argument in the range (0-1). FPGA implementation of an ANN for simultaneous prediction of thermodynamic properties and their derivatives required in advanced applications like MPS, DR, and optimization is not reported. Custom ANNs suitable for simultaneous estimation of both enthalpy and its derivatives suitable for power plant applications are taken up for implementation in FPGA. In the following sections, FPGA implementation of a single neural network called SteamNet is described. The simultaneous use of LUT and Taylor series for large arguments of sigmoid function is presented.

2. CUSTOM NETS FOR THERMODYNAMIC PROPERTIES OF WATER/STEAM

Figure 1 represents water and steam properties used in industrial applications [4]. The different regions represent subcooled 1). Supercritical water/steam. 2). Super heat steam. 3). Saturation. 4). High temperature steam. 5). Respectively. They are represented by highly nonlinear functions. The properties like enthalpy, entropy, specific volume, are dependent on process parameters like pressure and temperature.

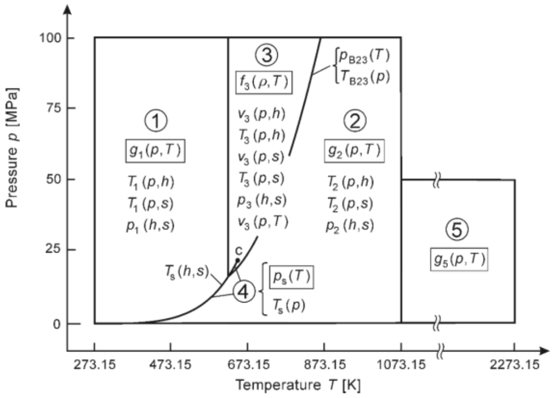


Figure 1. Regions of steam properties as per IAPWS-IF97

The enthalpy functions for all the regions are represented by different neural networks as shown in Figure 2. The networks SupNet, SubCNet, SatVnet and SatFnet represent enthalpy in superheated, sub cooled, saturated vapor and saturated fluid regions of Figure 1. P2T network is used to decide the zone in which steam property is required. Function Φ(P, T) in Figure 2 represents thermodynamic property, for example enthalpy, as a function of pressure P and temperature T. A single neural network, SteamNet, with compact architecture 2x10x5x3, shown in Figure 3, is used to represent all these different networks. The architecture is chosen such that the accuracy of and both the derivatives $\frac{\partial\phi}{\partial P}$ and $\frac{\partial\phi}{\partial T}$ are obtained with good accuracy. SteamNet stores different weight and bias matrices suitable for different regions of Figure 2. SteamNet is designed with sigmoid activation function represented by (1) and (2). Sigmoid function takes the weighted sum of outputs from the previous layer. FPGA implementation is achieved by using single perceptron with sigmoid activation function.

$$\mathrm{Z} = \sum(W_{ij}\,X_i + b_j) \tag{1}$$

$$Y = \frac{1}{(1+e^{-Z})} \tag{2}$$

Where W_{ij} and b_j are weight matrix between layers i and j and bias vector at layer j respectively. X_i is the input vector in the i^{th} layer. In the SteamNet, typical values of W_{ij} and b_j are in the range (-16.1735 to +23.174). For normalized values of X_i, this results in larger values of |Z|, greater than 1. Accurate exponentiation of |Z|, results in higher order terms of Taylor series which results in an overflow in FPGA due to factorial terms in denominators in series representation. However, acceptable accuracy with few terms in series is obtained when |Z| < 1.

Hence, the exponential function appearing in (2) is recast as in (3) and (4) to facilitate its implementation in FPGA.

$$e^Z = e^a e^b \tag{3}$$

$$Z = a + b \tag{4}$$

Argument Z is split into a and b, such that -1<= a <=1 and b is an integer; e^b is a set of predefined constants (B_j) stored in look up table (LUT). Term e^a is obtained by Taylor series using (5).

$$e^a = 1 + a(1 + a(C_1 + a\left(C_2 + a\big(C_3 + a(C_4 + aC_5)\big)\right)) \tag{5}$$

where C_1 to C_5 are constants representing (1/2!) to (1/6!). The above implementation involves multiply, add and comparator blocks. ReLu, (6), which is another option, widely used in neural nets, involves only a comparator. It is simple and requires only a comparator as opposed to many adders, multipliers and comparators in sigmoid.

$$Y = \max(0, Z) \tag{6}$$

Saving in device utilization is seen in ReLU as compared to sigmoid vide Table.1. However, ReLu is less accurate as shown in Figure 4. Hence, SteamNet regression is implemented using sigmoid. Marginal gain in device utilization in SteamNet can be achieved by reducing the number of terms in (5), vide Figure 5.

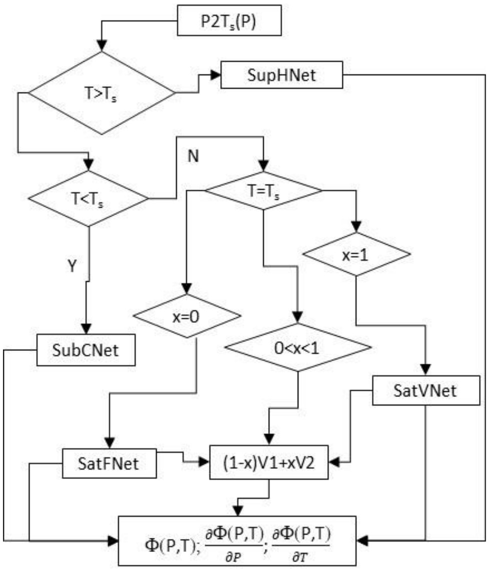
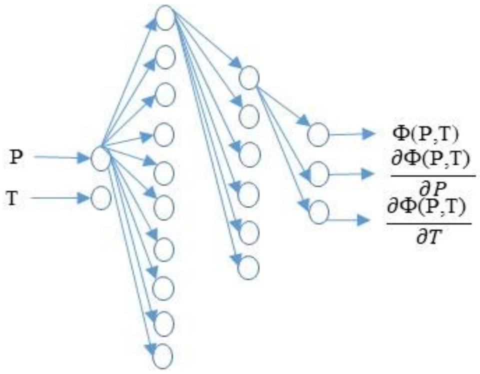

P2T$_s$(P)
T>T$_s$
SupHNet
N
T<T$_s$
T=T$_s$
x=1
Y
x=0
0<x<1
SubCNet
SatVNet
SatFNet
(1-x)V1+xV2
Φ(P,T); $\frac{\partial\Phi(P,T)}{\partial P}$; $\frac{\partial\Phi(P,T)}{\partial T}$

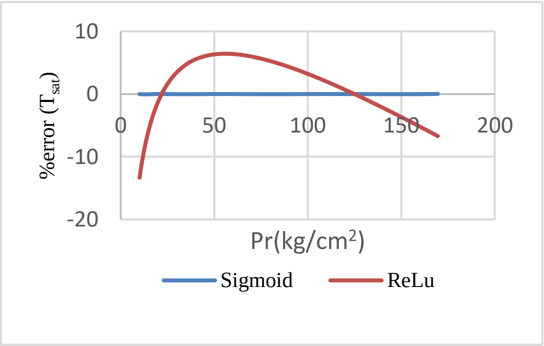
Figure 2. SteamNet for different water-steam zones


P
T
Φ(P,T)
$\frac{\partial\Phi(P,T)}{\partial P}$
$\frac{\partial\Phi(P,T)}{\partial T}$

Figue 3. SteamNet architecture

Table1. Device ultilization summary (estimated values)

Logic utilization	Used		Available		Utilization	
Act. Function	Sigmoid	ReLu	Sigmoid	ReLu	Sigmoid	ReLu
No. Slice LUTs	8567	31	150720	150720	5%	0%
No. LUT-FF pairs	0	0	8567	31	0%	0%
No Bonded IOBs	64	64	600	600	10%	10%



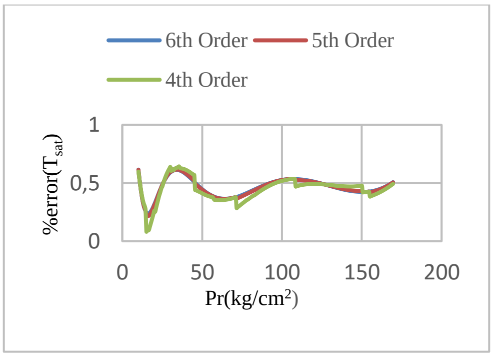
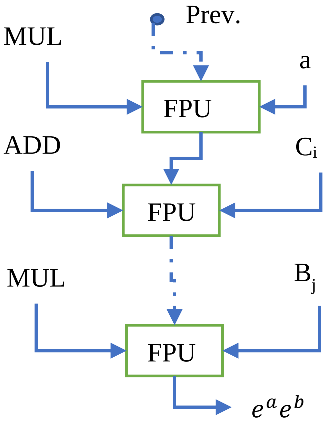


Figure 4. Regression error - Sigmoid vs ReLu

Figure 5. Tsat error with truncated Sigmoid

Figure 6 shows implementation of (5) using generic floating-point unit (FPU), based on the IEEE754 32-bit floating format. FPU provides operations like ADD, MULTIPLY, DIVIDE and

SUBTRACT. The SteamNet requires four or five coefficients, depending on accuracy for e^a and nine constants for e^b. Figure 7 shows the percent error in sigmoid implemented in Verilog as per (5) as compared to standard numerical library functions.



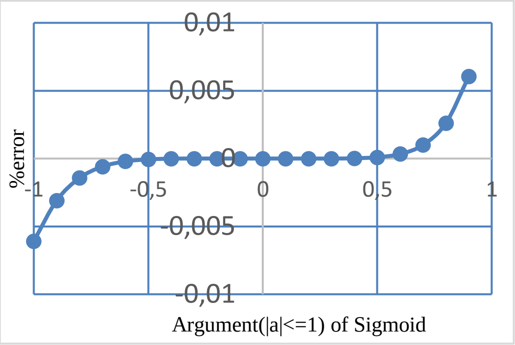


Figure 6. Exp. function with large arguments

Figure 7. Sigmoid error in verilog HDL

3. STEAMNET IMPLEMENTATION ON FPGA

All layers of SteamNet shown in Figure 8 are implemented by a single neuron model. Inputs to each of the ANNs are normalized to be in the range (0.1–0.9), by (7) to avoid asymptotic saturation in sigmoid function. Equation (8) de-normalizes the outputs to get actual values in engineering units.

$$X_n = \frac{0.8(X-X_{min})}{(X_{max}-X_{min})} + 0.1 \tag{7}$$

$$Y = \frac{(Y_n-0.1)(Y_{max}-Y_{max})}{0.8} + Y_{min} \tag{8}$$

Function P2T in the SteamNet, P2T with pressure (Pr - kg/cm^2) as input and saturation temperature (T$_{sat}$ -^{o}C) as output using a custom three-layer network is realized using Xilinx ISE Design suite and Verilog HDL with Virtex family FPGA. Figure 9 and Figure 10 show the structural model of steam Net using Xilinx ISE synthesis tool and the device utilization summary of steam net implemented in Virtex XC6VCX240T FPGA board, respectively.

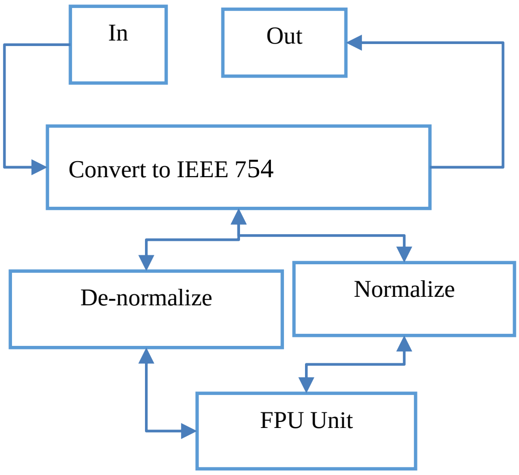


Figure 8. Normalizing and De-normalizing (input-output)

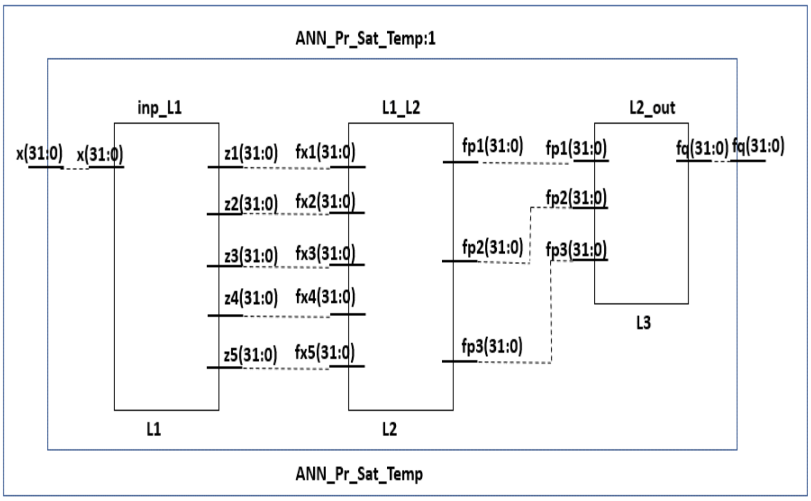
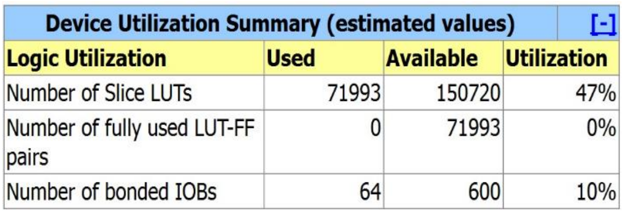


Figure 9. SteamNet structural model in verilog



Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	71993	150720	47%
Number of fully used LUT-FF pairs	0	71993	0%
Number of bonded IOBs	64	600	10%

Figure 10. Device utilization of SteamNet

4. RESULTS AND DISCUSSION

The above section details the implementation of SteamNet using a single neuron model wherein different weight matrices and bias vectors, at the three layers of the network in Figure 3, are stored in memory. Computation of the output is achieved by repeated usage of single neuron model implemented as per Figure 6 and Figure 8. The Figure 11, Figure 12 and Figure 13 show the layer-1, layer-2 and layer-3 input-output simulation results respectively, obtained from Xilinx ISE simulator for a typical normalized input of 0.1 kg/cm^2. The simulation shows, at time t=0, for layer-1, an input value of x=oxh3dcccccd produces outputs at five neurons as fx1=ox3ee39af7, fx2=oxf4ea1cf, fx3=3eddf270, fx4=3ed6dece and fx5=3f0d5195. Similarly, the outputs are shown for the second and third layers. This results in a normalized output of T_{sat} of 0.14791287 corresponding to 63.699696 deg C. This has a maximum error of 0.61% as shown in Figure 14.

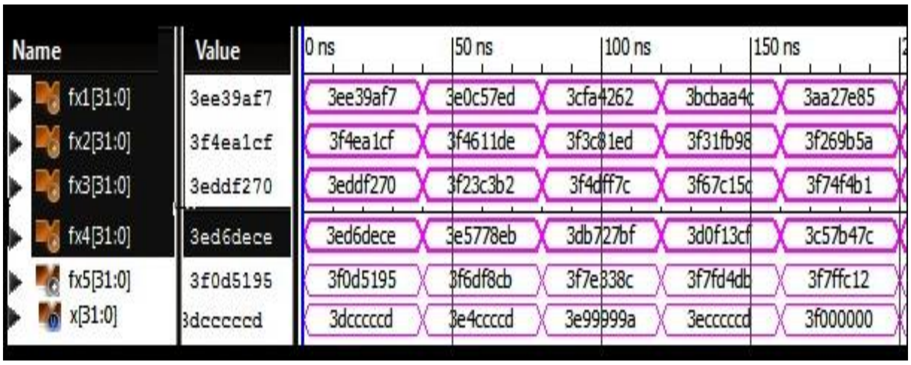
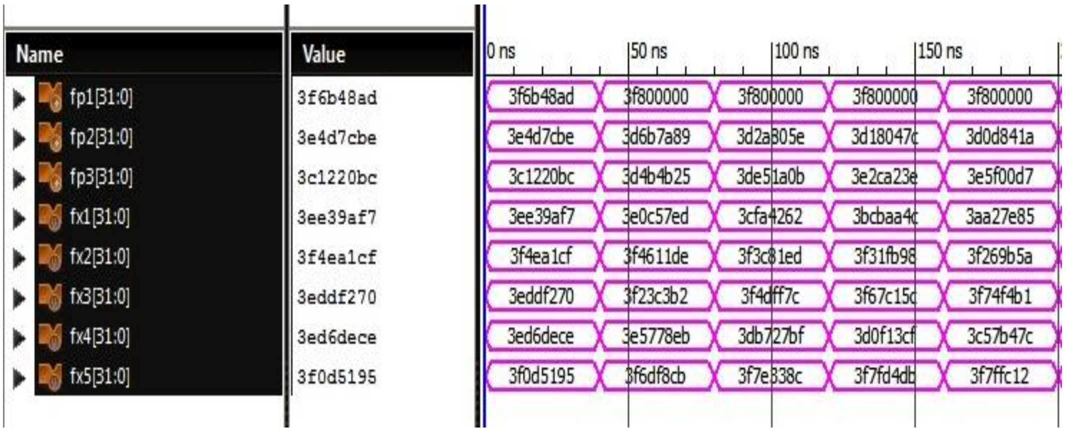
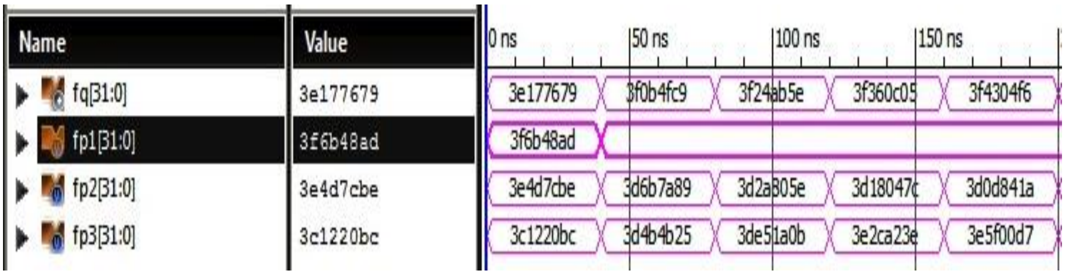


Figure 11. Layer-1 simulation results



```
Name         Value      0 ns       |50 ns     |100 ns    |150 ns
fp1[31:0]    3f6b48ad   3f6b48ad   3f800000   3f800000   3f800000   3f800000
fp2[31:0]    3e4d7cbe   3e4d7cbe   3d6b7a89   3d2a805e   3d18047c   3d0d841a
fp3[31:0]    3c1220bc   3c1220bc   3d4b4b25   3de51a0b   3e2ca23e   3e5f00d7
fx1[31:0]    3ee39af7   3ee39af7   3e0c57ed   3cfa4262   3bcbaa4c   3aa27e85
fx2[31:0]    3f4ea1cf   3f4ea1cf   3f4611de   3f3c81ed   3f31fb98   3f269b5a
fx3[31:0]    3eddf270   3eddf270   3f23c3b2   3f4dff7c   3f67c15c   3f74f4b1
fx4[31:0]    3ed6dece   3ed6dece   3e5778eb   3db727bf   3d0f13cf   3c57b47c
fx5[31:0]    3f0d5195   3f0d5195   3f6df8cb   3f7e838c   3f7fd4db   3f7ffc12
```

Figure 12. Layer-2 simulation results



```
Name         Value      0 ns       |50 ns     |100 ns    |150 ns
fq[31:0]     3e177679   3e177679   3f0b4fc9   3f24ab5e   3f360c05   3f4304f6
fp1[31:0]    3f6b48ad   3f6b48ad
fp2[31:0]    3e4d7cbe   3e4d7cbe   3d6b7a89   3d2a805e   3d18047c   3d0d841a
fp3[31:0]    3c1220bc   3c1220bc   3d4b4b25   3de51a0b   3e2ca23e   3e5f00d7
```

Figure 13. Layer 3 simulation results

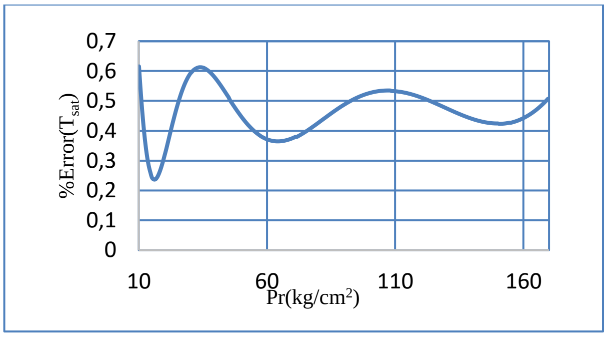


Figure 14. Tsat Error - SteamNet vs ASME steam table

5. CONCLUSION

A pre-trained compact single neural net for simultaneous estimation of steam properties and their partial derivatives is taken up for implementation in FPGA which is useful for real time control and monitoring of power plants. This technique enables entire steam properties and their derivatives to be implemented in FPGA without the need for porting nearly 5000 lines of C/C++ code. FPGA implementation of one network showing the relationship between pressure and saturation temperature is demonstrated. Verilog HDL simulation results obtained for XC6VCX240T device of Virtex family with IEEE 754 floating-point 32-bit data representation with an efficient sigmoid activation having large arguments are presented. Error in the sigmoid function for the argument range (-30-30) is within 0.02% which gives a mean accuracy of final outputs of the SteamNet to be around 0.5%. Results of FPGA simulation on a target system show the cycle time is of the order of 700ns. Device utilization for the network are presented using the approach. This technique enables to implementation of manytranscendental functions in FPGA without the need for porting the iterative complex codes; hence it is expected many real time applications like enthalpy-based steam temperature control in boilers, often used in power and process, can be easily developed.

ACKNOWLEDGEMENTS

The authors acknowledge the support and encouragement provided by the management of CVR College of Engineering, Ibrahimpatnam, Hyderabad, India.

REFERENCES

[1] S. Guo, P. Liu and Z. Li, “Inequality constrained nonlinear data reconciliation of a steam turbine power plant for enhanced parameter estimation,” *Energy*, vol. 103, pp. 215–230, 2016, doi: 10.1016/j.energy.2016.02.158.

[2] D.Wang, X.Wu and J. Shen, “An efficient robust predictive control of main steam temperature of coal-fired power plant,” *Energies*, vol. 13, no. 15, p. 3775, 2020, doi: 10.3390/en13153775.

[3] P. Niegodajew, M. Marek, W. Elsner, and Ł. Kowalczyk, “Power plant optimisation effective use of the nelder-mead approach,” *Processes*, vol. 8, no. 3, p. 357, 2020, doi: 10.3390/pr8030357.

[4] W. Wagner and H.-J. Kretzschmar, *International Steam Tables-Properties of Water and Steam based on the Industrial Formulation IAPWS-IF97*, Springer Science & Business Media, 2007.

[5] M. Åberg., J. Windahl., H. Runvik., F. Magnusson., “Optimization-friendly thermodynamic properties of water and steam,” *Proceedings of the 12th International Modelica Conference*, 2017, doi: 10.3384/ecp17132449.

[6] “Guideline on the fast calculation of steam and water properties with the spline-based table look-up method (SBTL),” *The International Association for the Properties of Water and Steam*, IAPWS G13-15, July 2015. [Online]. Available: http://www.iapws.org/relguide/SBTL.html.

[7] P. Cardaliaguet and G. Euvrard, “Approximation of a function and its derivative with a neural network,” *Neural Networks*, vol. 5, no. 2, pp. 207-220, 1992, doi: 10.1016/S0893-6080(05)80020-6.

[8] S. Hashem and B. Schmeiser, “Approximating a function and its derivatives using MSE-optimal linear combination of trained feedforward neural networks,” *Proc. of 1993 World Congress on Neural Networks*, vol. 1, pp. 617-620, 1993.

[9] R.V.S. Krishnadutt and J. Krishnaiah, “Neural nets for thermodynamic properties”, *Proceedings of the First International conference on Computational Intelligence and Informatics, Advances in Intelligent Systems and Computing*, vol. 507, 2017, doi: 10.1007/978-981-10-2471-9_68.

[10] A. R. Omondi and J. C. Rajapakse, “Neural networks in FPGAs,” *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP '02.*, 2002, pp. 954-959 vol.2, doi: 10.1109/ICONIP.2002.1198202.

[11] J. M. Granado, M. A. Vega, R. Perez, J. M. Sanchez and J. A. Gomez, “Using FPGAs to implement artificial neural networks,” *2006 13th IEEE International Conference on Electronics, Circuits and Systems*, 2006, pp. 934-937, doi: 10.1109/ICECS.2006.379943.

[12] H. Hu, J. Huang, J. Xing and W. Wang, “Key issues of FPGA implementation of neural networks,” *2008 Second International Symposium on Intelligent Information Technology Application*, 2008, pp. 259-263, doi: 10.1109/IITA.2008.239.

[13] A. Muthuramalingam, S. Himavathi and E. Srinivasan., “Neural network implementation using FPG: A issues and applications,” *International Journal of Electrical and Computer Engineering*, vol. 2, no. 12, 2008, doi: 10.5281/zenodo.1084402.

[14] J. Misra and I. Saha, “Artificial neural networks in hardware: A survey of two decades of progress,” *Neurocomputing*, vol. 74, no. 1-3, pp. 239-255, 2010, doi: 10.1016/j.neucom.2010.03.021.

[15] K. Guo, S. Zeng, J. Yu, Y. Wang and H. Yang., “A survey of FPGA based neural network accelerator,” *ACM Transactions on Reconfigurable Technology and Systems*, vol. 12, no. 1, pp.1-26, 2018, doi: 10.1145/3289185.

[16] Z. Li, Y. Wang, T. Zhi and T. Chen, “A survey of neural network accelerators,” *Frontiers of Computer Science*, vol. 11, no. 5, pp. 746–761, 2017, doi: 10.1007/s11704-016-6159-1.

[17] L. S. Mezher, "Hamming neural network application with FPGA device," *International Journal of Reconfigurable and Embedded Systems (IJRES)*, vol. 10, no. 1, pp. 37-46, 2021, doi: 10.11591/ijres.v10.i1.pp37-46.

[18] Y. Tao, R. Ma, M. Shyu and S. Chen, “Challenges in energy-efficient deep neural network training with FPGA,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 1602-1611, doi: 10.1109/CVPRW50498.2020.00208.

[19] F. Piazza, A. Uncini and M. Zenobi, “Neural networks with digital LUT activation functions,” *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, 1993, pp. 1401-1404 vol.2, doi: 10.1109/IJCNN.1993.716806.

[20] V. Bieu, J. Peperstraete, J. Vandewalle and R. Lauwereins, “Close approximations of sigmoid functions by sum of steps for VLSI implementation of neural networks,” *The Scientific Annals, Section: Informatics*, vol. 40, no. 1, pp. 31-50, 1994.

[21] P. Lysight, J. Law and D. Girma.,“Artificial neural net implementation on a fine grained FPGA,” in *Field-Programmable Logic Architectures, Synthesis and Applications*, vol. 849, 1994, 10.1007/3-540-58419-6_126.

[22] A. Tisan, S. Oniga, D. Mic and A. Buchman, “Digital implementation of the sigmoid function for FPGA circuits,” ACTA *TECHNICA NAPOCENSIS Electronics and Telecommunications*, vol. 50, no. 2, pp. 15-20, 2009.

[23] P. Dondon, J. Carvalho, R. Gardere, P. Lahalle, G. Tsenov and V. Mladenov, “Implementation of a feed-forward Artificial Neural Network in VHDL on FPGA,” *12th Symposium on Neural Network Applications in Electrical Engineering (NEUREL)*, 2014, pp. 37-40, doi: 10.1109/NEUREL.2014.7011454.

[24] S. Ngah, R. A. Bakar, A. Embong and S. Razali., “Two-steps implementation of sigmoid function for artificial neural network in field programmable gate array,” *ARPN Journal of Engineering and Applied Sciences*, vol. 11, no.7, pp. 4882-4888, 2016.

[25] Z. Li, Y. Huang and W. Lin, “FPGA implementation of neuron block for artificial neural network,” *2017 International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, 2017, pp. 1-2, doi: 10.1109/EDSSC.2017.8126431.

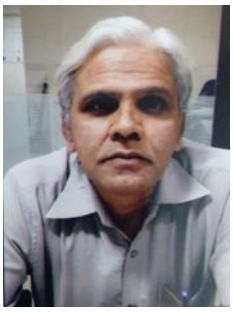
[26] M. Krips, T. Lammert and A. Kummert, “FPGA implementation of a neural network for a real-time hand tracking system,” *Proceedings First IEEE International Workshop on Electronic Design, Test and Applications '2002*, 2002, pp. 313-317, doi: 10.1109/DELTA.2002.994637.

[27] C. Latino, M. A. Moreno-Armendariz and M. Hagan, “Realizing general MLP networks with minimal FPGA resources,” *2009 International Joint Conference on Neural Networks*, 2009, pp. 1722-1729, doi: 10.1109/IJCNN.2009.5178680.

[28] D. Dąbrowski, E. Jamro and W. Cioch, “Hardware implementation of artificial neural networks for vibroacoustic signals classification,” *ACTA PHYSICA POLONICA A*, vol. 118, no. 1, pp. 41-44, 2010.

[29] O. Polat and T. Yildirim, “FPGA implementation of a general regression neural network: an embedded pattern classification system,” *Digital Signal Processing*, vol. 20, no. 3, pp. 881-886, 2010, doi: 10.1016/j.dsp.2009.10.013.

BIOGRAPHIES OF AUTHORS



R. V. S. Krishna Dutt is currently professor of Computer Science and Engineering, CVR College of Engineering, Hyderabad, India. He worked earlier as General Manager, Corporate Research & Development, BHEL, Hyderabad, India. He had successfully led interdisciplinary enterprise level development projects in the last four decades. Currenttly focused on driving innovation and startups with government funding. He has few publications and patents to his credit. He holds a B.Tech in Electrical engineering from NIT,Warangal, MS in Applied Mechanics from IIT, Chennai and M.Tech in Computer science and engineering, University College of Engineering, Osmania University, Hyderabad, India. He has nearly four decades of experience related to the power industry. His areas of interest are application of ML techniques to industrial problems.

R. Ganesh. is currently Associate Professor of Electronics and Communication Engineering, CVR College of Engineering, Hyderabad, India. He worked as a team member in Research & Development projects funded by DRDO, ECIL in the areas of OFDM and Multi Channel Analyzers. He has guided 15 M.Tech Projects and 19 B.Tech. Projects.He has published a total of 15 journal and conference papers. He has coordinated and conducted 18 workshops/training programs for the benefit of faculty and students in the filed of VLSI using different EDA tools. He also delivered 15 training sessions as a resource person for VLSI workshops and training programs. He obtained B.Tech degree in Electronics and Communication Engineering, M.Tech in VLSI System Design from JNTUH, Hyderabad and Advanced Post Graduate Diploma in VLSI from Vedant, SCL, Chandigadh. Presently, he is pursuing Ph.D. from JNTUH, Hyderabad. His areas of interest are the design and applications of VLSI, 4G/5G Communications and AI/ML algorithms using VLSI.

P. Premchand obtained B.Tech in Electrical Engineering from NIT, Jamshedpur, India. M.E. and Ph.D degrees in Computer Science and Systems Engineering from Andhra University, Visakhapatnam, India. He worked as a lecturer in the Dept. of CSE, Andhra University. He later joined as Associate professor, in the Dept. of CSE, University College of Engineering, Osmania University, Hyderabad, India and later became a Professor of CSE in the same college. He served as Director of AICTE, at New Delhi. He also served as Head, Dept. of CSE, Chairman Board of studies in CSE, Dean Faculty of Engineering at University College of Engineering, Osmania University, Hyderabad. He is currently Professor, Dept. of CSE and Dean, Faculty of Informatics at Osmania University, Hyderabad, India. He is a member of the selection committee of ISRO, NRSA, ADRIN and NGRI (Govt. of India Organizations). He has guided 31 Doctoral research scholars and many are pursuing Ph.D degrees under his guidance. He has presented several papers in National and international Conferences and journals.