International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 5, October 2019, pp. 3423~3431
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i5.pp3423-3431

An optimized method towards formal verification of mixed signals using differential fed neural network over FFNN

Vidya D. S., Manjunath Ramachandra
Assam Don Bosco University, Guwahati, India

Article Info

Article history:

Received Oct 10, 2018
Revised Mar 26, 2019
Accepted Apr 15, 2019

Keywords:

Area
Formal verification
Mixed signals
Neural networks
Power

ABSTRACT

Today, the semiconductor industries are rapidly usinganalog and mixed signals to achieve cost-effective solutions on a System on Chip (SoC) design. The SoC device is a part of analog, digital and essential mixed-signal models/circuits merged on a semiconductor device, which provides the platform to build modern retail/consumer electronics appliances with smart technology. In order to evaluate the mixed signals, the conventional approaches are not effective with respect to its performance, time and manufacturing cost. Thus, the recent researches were much interested in formal verification technique as it provides the evidence of conscious algorithms in a system. The demand for formal verification in the SoC designs in the context of software and hardware platform is high because of its cost and accuracy. Thus, the paper introduces atechnique of formal verification for mixed signals by using training models of the Differential fed neural network (DFNN) over feedforward neural network (FFNN). The formal verification is performed through equivalence checking by using recently adopted designs as reference designs. The outcomes of the verification techniques suggests that DFNN based technique improves the training accuracy and optimizes the hardware resources like area, power than the FFNN based technique.

Corresponding Author:

Vidya D. S.,
Research Scholar,
Assam Don Bosco University,
Guwahati, India.
Email: dsvidhya0770@gmail.com

1. INTRODUCTION

Various forms of research activities were made in the recent past which provided a breakthrough for the computer-aided design (CAD) practices, mainly in the design of hardware description language (HDL) meant for analysis of the mixed signals (MS) behavior [1]. Some of the important design mechanisms were presented to various form of design procedures by using different design techniques. The prime intention of CAD technique is analysis and design verification but which is challenging task and it needs the lot of expertise and in-depth understanding of analog and mixed signals (AMS) behavior [2, 3]. The verification process involves various kinds of issues in all the stages of the design procedure. Also, the wide range of difference can be observed in the different classes of AMS designs [4]. The major problem during the verification process for AMS designs at adaption level of the analog circuits are the characteristics defined directly in terms of uninterrupted signal quantities [5]. Also, the problem associated with the proper functionalities of integrated designs need to meet the system performance of the system with respect to power utilization and area [6]. This paper introduces a formal verification method to verify the mixed signals by training and learning the networks using Feedforward neural network (FFNN) and differential fed neural network (DFNN). The entire paper is organized with the sections like Review of existing research (Section 1.2) to understand the current state of art in the verification of mixed signals, Proposed system model

(Section 2) discusses techniques adopted for formal verification, results and analysis (Section 3) dealing with performance analysis of both the FFNN and DFNN techniques and conclusion gives the concluding points of the paper in (Section 4).

The existing researches were selected and reviewed from the official publications like IEEE, Springer, Elsevier meant with analog, mixed signals verification approaches. The concept discussion on verification of software on chip (SoC) including analog circuitry is addressed in Kundert and Chang [6] which involves the methodical process, a necessity of verification, implementing analog verification. A modeling approach for analog/mixed signals is found in Gang [7] which uses Verilog coding. The work [7] adopted a top-level methodology which brings integration of simulation and modeling in the Cadence environment. Through this methodology, [7] can achieve efficient and accurate model. Similar research addressing verification of SoC with a mixed signal is found in Yang et al. [8] by using the Verilog model. The design of [8] builds an equivalent model of high-level radio frequency (HLRF) by integrating Verilog language with mixed signals. The verification analysis is performed by using digital methods like random data generation, manual verification methodology, assertion-based verification, and coverage-driven verification. From the analysis, it has been found that [8] yields faster verification than other existing approaches. A research survey addressing existing pitfalls in formal verification techniques of analog and mixed signals is found in Vidya and Manjunath [9]. This kind of verification is the critical stage of the design which assures the circuit design preciseness and its behavior with respect to the design of the system. The work [9] discussed various aspects of the standard techniques and models which execute effectiveness in terms of area of circuit design.

The work of Harinarayana et al. [10] presented an automated verification model for mixed-signal SoCs designed for modern application by integrating analog, digital and mixed signals. The verification is performed at different levels and different modes. A framework based on differential learning approach to performing the formal verification is presented in Vidya [11] by considering mixed signals. The technique has presented an analytical modeling approach that considers an algorithm to generate the multiple mixed signals meant for a feasible operation of the mixed signal circuits, an algorithm for training and verification. From outcomes analysis, it was found that [11] achieved 98.7% accuracy with the better speed of response than other machine learning algorithms.The work of Chitti et al. [12], presented a unique test model for different SoCs in which Cadence verification methodology is adopted for the SoC environment. From the outcome analysis [12] found an effective reduction in simulation time of SoC with respect to other existing works. Similarly for application prospective Bouziane et al. [13] presented an optimal verification for speaker modeling by using personalized background models. Similar kind of work is found in Kumari and Jayanna [14] which is meant for verification of limited data speakers.

From a review of all the recent literature, it has been observed that significant works carried on formal verification of mixed signals based on the software approaches are quite rare in terms of hardware architectures. Thus, relevant works are needed to be focused on the research issues by considering hardware-based approaches. On analyzing recent works analysis it is been found that very less work towards the analog and mixed signals found with formal verification of mixed signals. Most of the research works have been evolved with industrial applications prospectives. Also, in hardware modeling prospectives some of the techniques were adopted formal verification technique with training of mixed signals, Fuzzy networks, etc. Also, it is been observed that very less emphasis given for formal verification of mixed signals in real-time scenario. Existing works are lacks with optimizations approachesfor formal verification of mixed signals on Digital IC's and were lags with low cost-effective solutions for SoC platform. Thus, there is a need of *"novel optimized, cost-effective approach"*

2. PROPOSED SYSTEM MODEL

The prime intention of this work is to perform the formal verification of mixed signals by using training/Learning networks like differential fed neural network (DFNN) [15] or Feedforward neural network (FFNN) [16]. In order to design the hardware architecture of both training algorithms DFNN and FFNN a backpropagation (BP) algorithm is used. Both the DFNN and FFNN are having perceptron layer architecture with 2-3-1 network type which is trained by back propagation to update the weights and represent the input and outputs. The major objective is to implement the hardware architecture of training networks like DFNN and FFNN for formal verification of mixed signals to improve the performance with high accuracy in real time applications. The overview of the implemented research methodology is given inthe form of design flow as described in Figure 1.

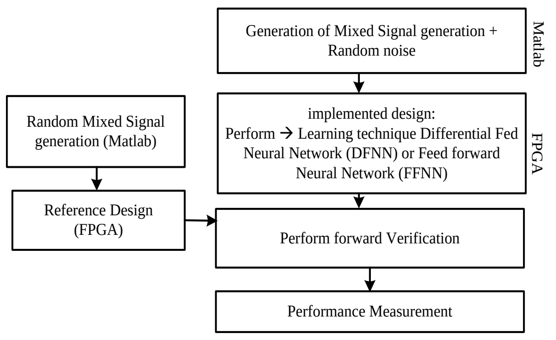
Generation of Mixed Signal generation + Random noise
Matlab
Random Mixed Signal generation (Matlab)
implemented design:
Perform → Learning technique Differential Fed Neural Network (DFNN) or Feed forward Neural Network (FFNN)
FPGA
Reference Design (FPGA)
Perform forward Verification
Performance Measurement

Figure 1. Research methodology of the proposed design

The design flow involvestwo different set of designs: 1) implemented design and 2) reference design to perform formal verification. The design consists of mixed signal generation module along with random noise addition using Matlab which is followed by hardware architecture of differential fed or feed-forward neural network model using Backpropagation training model. The reference design includes random mixed signal generation by using Matlab and its outcomes will be saved as a text file and then followed with reference model whih consists of IP Core memory generation by using saved text file as a memory coefficient file. The implemented design and reference design are modeled by RTL coding and the models outcomes are considered for formal verification by using equivalence checking. The performance analysis carried out by considering hardware constraints like Area, power utilization and evaluated the accuracy rate with error findings. The following discussion will give the detailed explanation of the proposed approach towards formal verification.

2.1. Mixed signal generation

The generation of multiple mixed signals is introduced to test the formal verification with respect to DFNN or FFNN training. The generation of the mixed signal generation is performed by using MATLAB which includes both analog and digital form of the signals and are characterized by current and voltage electrical factors. Based on thesecurrent and voltage factors, the mixed signals are categorizedas stable or unstable state signals. Further, on the basis of higher and lower electrical factors, the multiple mixed signals are generated and are feasible for performance analysis.

In order to generate the multiple mixed signals, the time interval of t= 0:2 is defined with a spacing of 0.0005 and sampling frequency of fs= 5 Hz. Later, a round function ‘Rf' is considered to generate the data signal with the help sine function and its outcome will be round off to nearest hexadecimal value. The training network is defined as 2-3-1 structures to separate two signals are fed as one mixed signal to the network. The (mixed signal-1) ms1 can be defined as follows. The data signal is represented by ‘Ya’ is defined as

$$Y_a = R_f \times (c1 + a1 \times \sin(2\Pi \times fs \times t)) \tag{1}$$

Add the pseudo-random numbers to the data signal to form the mixed signal M1 is represented below

$$M1 = R_f \times (Y_a + 0.1 \times Rn(1, Y_a)) \tag{2}$$

where Rn is normally distributed pseudorandom numbers which generates the ‘Ya' data signals, similarly, for the second signal to the network as defined as,

$$Y_b = R_f \times (c2 + a2 \times \sin(2\Pi \times fs \times t)) \tag{3}$$

$$M2 = R_f \times (Y_b + 0.1 \times Rn(1, Y_b)) \tag{4}$$

Constant value set to c1 = 5, c2 = 15 and amplitude set to a1 = 4.1 and a2 = 10. Hence, equations (2) and (4) are considered as a single mixed signal ‘ms1’ by using M1 and M2 respectively to the training network.Similarly, change the amplitudes and constants with different pseudo-random to generate the multiple mixed signals. Generate the M3 and M4 using (2) and (4) to form the mixed signal ‘ms2' using different constants c3 = 4, c4 = 15 and amplitudes a3 = 4 and a4 = 5.1. Similarly, for mixed signal ‘ms3’ generation, generate the M5 and M6, using equations (2) and (4) with different constants c5= 5, c6=5 and amplitudes a5=4.05 and a6=4. The peak voltage or current electric factors are changing with respect to amplitude value changes in the mixed signal generation. The variation in the amplitude values will decide the high electricity or low electricity factor. If add more complex pseudo-random number the data signal will become the unstable state. The generated multiple mixed signals ‘ms1', ‘ms2', and ‘ms3' are fed to the training network individually to perform the formal verification for performance analysis.

2.2. Module for training

The training module is used to train the different types of neural networks like DFNN and FFNN for formal verification using mixed signals. The design flow of the Training module is represented in Figure 2. First the network type is defined by using DFNN and FFNN. Then, the numbers of samples are selected and are fed to the network form the generated mixed signals. Then construct the neural networks.

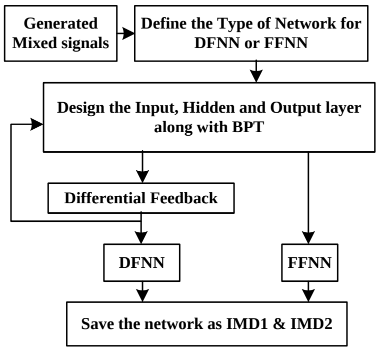
Generated Mixed signals
Define the Type of Network for DFNN or FFNN
Design the Input, Hidden and Output layer along with BPT
Differential Feedback
DFNN
FFNN
Save the network as IMD1 & IMD2

Figure 2. The design flow of training module

For both the DFNN and FFNN networks, the network structure is fixed 2-3-1. The number of neurons to be trained is decided by the network structure. The proposed network structure includes two input layers, three hidden layers, and a single output layer along with backpropagation training (BPT) Module. For DFNN Module differential feedback will be fed to all the three layers includes input, hidden and output layers and also based on the Differential flow the corresponding output will be generated to each layer. The outputs of the DFNN and FFNN are saved as an implemented design-1 (IMD1) and implemented design-2 (IMD2) respectively. The corresponding DFNN and FFNN hardware architectures are explained in below sessions.

2.3. Feed forward neural network (FFNN)

The hardware architecture of the feed-forward neural network with backpropagation training is presented with a 2-3-1 network structure in Figure 3. Each of the input layers receives the inputs from the multiple mixed signals like ms1 or ms2 or ms3. Each mixed signal ms1 is having two signals are inputs to din1 and din2 input layer. In the input layer, two input nodes are used, each input node is having, single 8-bit input din1 and three weights w1, w2, and w3. The three multipliers are used to multiply the din1 with weights w1, w2 and w3 to generate the il1, il2, and il3. Similarly use input node2 to generate the il4, il5, and il6. The hidden layer receives the six inputs from the input layers. The hidden layer mainly includes three separate linear neuron models. Each linear neuron module has two multipliers; one carry look ahead adder and linear function using LUT (Look-up-Table). The 16-bit inputs il1 and il2 are multiplied with weights wh1 and wh2 to get the products p1 and p2 which are added using carry look ahead adder.The carry looks ahead adder is fast parallel prefix adder which improves the speed of the hardware by reducing the time which is defined by the carry bits. The CLA outputs are inputs to LUT, which is a linear function, represented in hardware. The LUT receives the input from the adder; based on inputs it is capable of learning and updating the final hidden values. The three linear neurons are generated the outputs hl1, hl2, and hl3 which are input to the output layer. The output layer is having three multiplier and sigmoid model. The three inputs are multiplied with weights wo1, wo2, and wo3 and outcomes are added using adder module, the results are used in the sigmoid model to generate the final results. The sigmoid function provides the better way of learning the weights in neural networks by using its derivatives and bring nonlinearity into the networks. The backpropagation model is used to provides the training to the all the three layers by calculating and updating the new weights into it.

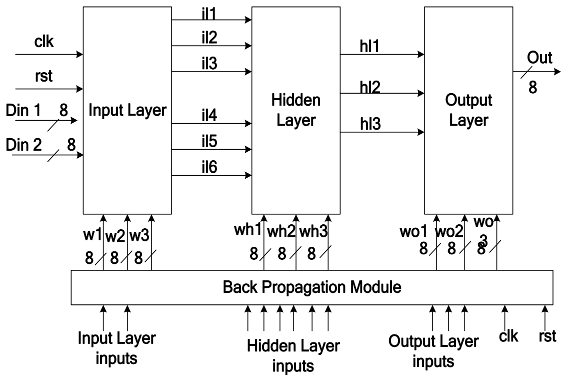

Figure 3. Feedforward neural network (FFNN) using BPT

2.4. Differential fed neural network (DFNN)

The hardware architecture of Differential fed neural network with Differential feedback module along with backpropagation is represented with 2-3-1 network structure in Figure 4. The architecture is similar to feed forward neural network except for differentially feedback and few changes in the designs. The Differential feedback is applicable to all the three layers.The din1 and din2 are the mixed signals applied as an input to the input layer and generates the six different outputs il1, il2, il3, il4, il5, and il6 along with differential feedback. The six outputs are inputs to the hidden layer. The hidden layer includes 3- different linear neuron models, which generates the three outputs hl1, hl2, and hl3 along with differential feedback. The output layer has 3 inputs are multiplied by three weights wo1, wo2, wo3, the three multiplied outcomes are ol1, ol2 and ol3 are feedback to the differential feedback model and the output of the differential feedback model is input to output layer which is followed by sigmoid function to generate the final 8-bit output ‘out'.

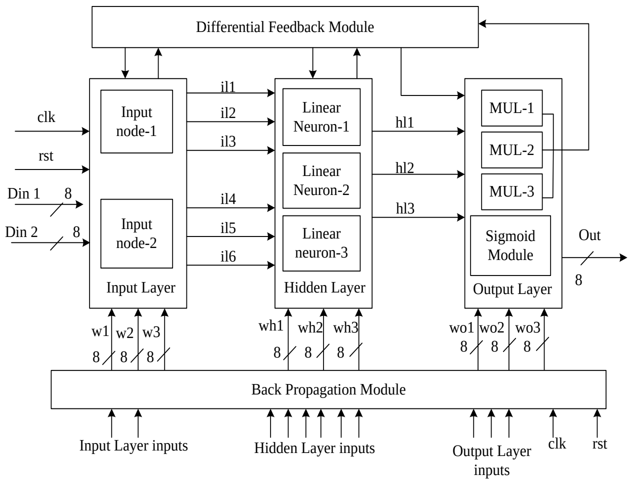

Figure 4 Differential fed neural network (DFNN)

The theoretical concept of differential feedback in DFNN is considered from [17], by using differential feedback, training will be brought down for the number of iteration or samples. According to the theoretical aspects from [17], the first order differential feedback with reduced layers output will be,

$$\sum(w_i \times m_i) + (b_1 \times d_1) \tag{5}$$

where w_i is the weights and mi is mixed signal inputs. The b_1 will be an autoregressive term and set to 1 and d_1 is 1st order differential. For differential orders of feedback, a set of DFNN will form a multiple of parallel structures. The proposed DFNN uses the three differentials of 2nd order differential feedback, for example in the output layer, (ol_2-ol_1), (ol_3-ol_2) and (ol_3-ol_1). The differential feedback of output layer is expressed as,

$$df\_o1 = wo_1 \times (ol_2 - ol_1) + wo_2 \times (ol_3 - ol_2) + wo_3 \times (ol_3 - ol_1) \quad (6)$$

where df_ol represents the 2nd order differential feedback of output layer and wo_l, wo_2 and wo_3 are updated weights. The df_ol is feedback to output layer is an input; in the output layer the sigmoid model generate the final training output based on the df_ol. The same concept is applicable to the other two layers includes input and hidden layers.By using the Differential feedback in the DFNN, the training time will be reduced, which boosts the performance in formal verification. The hardware complexity will be reduced with optimization of area and power than Feedforward neural network (FFNN). The back-propagation model is used in feed differential fed neural networks to determine the gradient that requires estimating the weights to be used in the given network. By using this model, adjusting the weights to reduce the error rate. The hardware implementation of the back-propagation model involves two major stages includes propagation and weight updation. The propagation stage involves generating the output data in the network using propagation forward. Based on the Target training network, the propagation of the activations back through the neural network to calculate the difference using targeted output and actual output from all the layers. The weight update stage involves the weight's difference, and layers activation is multiplied to generate the updated weight. The learning rate impacts the quality and speed of the training network. The learning rate 'L_r' is defined with one. If L_r is more, neuron training will be fast, and if it is less, neuron training will be more accurate. Consider all the layers inputs include input, hidden, and output layers add layer-wise individually, and the added outputs are set op1, op2, and op3 respectively. Once the propagation starts, reset all the layers weights and initialize to zero and Update the memory with actual data with three address location for three layers. The memory will be updated based on the counter. The difference Di' will be calculated using the actual data which is updated from memory is subtracted by output layer output 'O_p.' If the Difference 'D_i' will be less than memory actual data and counter is two, a counter will be reset to zero otherwise it will be increment till reaches two and a valid signal will be updated. The Different layers Inputs/ outputs 'D_o' will be updated based on the layer-wise. For output layer weight updation, op1, op2 and op3 will be considered as 'D_o.' For input and hidden layer weight updation, mixed signals will be the 'Do.' The weight updation will be defined as below:

$$\text{Updated Weight} = I_w + (Lr \times Di \times Do \times Op \times (1 - Op)) \quad (7)$$

From the equation (7) where the I_w is initial weights, The O_p will be defined based on the layers output For Output layer, O_p will be updated same and for input and hidden layers, the op1, op2, and op3 will be considered as 'O_p.'

2.5. Module for formal verification

The formal verification is verification technique which agrees or disagrees on the quality of related algorithms or models with respect to suitable formal methods or models or exact specifications in hardware and software viewpoint. In proposed design uses suitable models to tests the formal verifications of mixed signals. The formal modeling involves the formal equivalence checking; it is processed and used during the design and development of digital IC's, to prove formally that the two models of the designs reveal the same behavior. The formal verification of mixed signals is carried out using training networks DFNN and FFNN to find sits accuracy, and its typical design flow is represented in Figure 5. The complete formal verification is done based RTL-RTL modeling verification which is done through equivalence checking. The equivalence checking contains two main models namely implemented design and reference design. Hence, need to verify both the models of the designs exhibits the same features. For analysis purpose, The DFNN network saved as Implemented design-1(IMD1) and FFNN Network saved as Implemented design-2 (IMD2) as a trained data. The implemented designs IMD1 and IMD2 are designed with a hardware architecture of DFNN and FFNN respectively. Figure 5 shows the reference mixed signal generation which is similar to implemented designs, but here randomize more with pseudo numbers to form the mixed signals for formal verification with different inputs functionality. The DFNN reduce the training time internally while processing the formal verification of mixed signals. The output of reference randomly generated mixed signals are checking the equivalence with trained implemented design outcomes to check the accuracy.

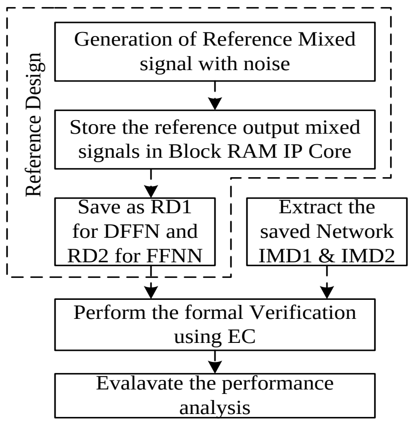
Reference Design
Generation of Reference Mixed signal with noise
Store the reference output mixed signals in Block RAM IP Core
Save as RD1 for DFFN and RD2 for FFNN
Extract the saved Network IMD1 & IMD2
Perform the formal Verification using EC
Evalavate the performance analysis

Figure 5. Typical design flow of formal verification

The reference design includes the reference mixed signals generation is done by using Matlab with noise which is similar to implemented design mixed signal generation. The output of reference mixed signals is stored in the text file. Which is having around 4000 samples and each sample is 8-bit and stores these samples as a coefficient file (.coe) in IP Core based Block RAM Model which is having the reference output signals. Save the reference design model has RD1 for DFNN and RD2 for FFNN. This implemented design IMD1 and reference design RD1 are tested for formal verification by using equivalence checking. In equivalence checking, both the IMD1 and RD1 are matched with same features, then consider the formal verification is "successful" if the features are not matched it is "unsuccessful," use a counter method to check the errors in the models.The accuracy of the formal verification models will find, if both the model's features are same, then counter is reset to zero. If not, counts start counting, when both the IMD1 and RD1 sampled data are not matching till the last samples. Based on the count, the accuracy will be calculated for formal verification. In order to perform the performance analysis, formal verification of mixed signals using DFNN is compared with the FFNN training method. The proposed formal verification of mixed signals using differential fed neural network provides better performance in terms of accuracy and less hardware complexity over the Feedforward neural network. The performance analysis of both the designs will be explained in the next section.

3. RESULT AND DISCUSSION

The proposed Formal verification of multiple mixed signals using training algorithms includes Feedforward Neural Network (FFNN), and Differentially-Fed Neural Network (DFNN) of 2-3-1 network Results are analyzed in the below section. The proposed work ismodeledover Xilinx ISE 14.7 environment using Hardware language Verilog-HDL and waveform simulated using Modelsim 6.5 simulator and implemented on Artix-7 FPGA.The synthesized results of proposed designs FFNN and DFNN which are tabulated in Table 1. Each design includes both implemented and reference design for formal verification of mixed signals.The resource utilization includes 334 slices LUT's of DFNN over 357 of FFNN with an improvement of nearly 7 %. The other resource utilization in both designs is almost the same.

Table 1. Device utilization of the proposed DFNN along with FFNN for formal verification

Logic Utilization	FFNN	DFNN
Number of Slice Registers	195	203
Number of Slice LUTs	357	334
Number of fully used LUT-FF pairs	122	124
Number of bonded IOBs	48	48
Number of Block RAM/FIFO	2	2
Number of DSP48E1s	39	39

The power utilization of DFNN and FFNN for formal verification of mixed signals is analyzed using X-Power analyzer tool from Xilinx ISE. The Total powers consumed by the DFNN and FFNN networks are 0.112W and 0.117W respectively with an improvement of 5%. The dynamic power utilization of both DFNN and FFNN network is 0.030W and 0.035W respectively which are tabulated in Table 2.

Table 2. Power consumptionof the proposed DFNN along with FFNN for formal verification

Total Power summary	FFNN	DFNN
Total Power (W)	0.117	0.112
Dynamic Power (W)	0.035	0.03

The maximum operating frequency is obtained from Artix-7 FPGA for the DFNN, and FFNN networks are 59.252 MHz and 37.605 MHz respectively with an improvement of 36.54 % which indicates that DFNN works at a higher operating rate than the FFNN Module. The proposed work targets to aim for formal verification of multiple mixed signals using trained networks like DFNN and FFNN. The higher rate of accuracy from the trained or learning network is a prime performance parameter for formal verification.The analysis of formal verification is done by equivalence checking for both DFNN and FFNN designs. Each design is having implemented (RTL) and reference (RTL) Modules and by using their design outputs, check it is equivalent, if yes, then the verification is 100% successful. If not, If it is different using equivalence checking with a counter method to find out the errors to get the exact accuracy of the trained network. To find an accuracy of training methods like FFNN and DFNN for formal verification, the analysis is carried out over multiple test environments by changing the network type or a number of samples which feed to the input layers as multiple mixed signals. The implemented designs contain complete hardware architecture of DFNN or FFNN which is fed by separate mixed signals as an input to the top implemented design architecture, followed by three hidden layers and single output layer gives the final results. The reference design contains actual trained data which is obtained from the mixed signal generation by Matlab then store it in Block RAM IP cores which gives complete reference model.In DFNN Training module, differential feedback which is generated from the output layer and fed back to the output layer as one of the inputs which improves the training time and accuracy of the training network.

The formal verification outcomes with an accuracy of multiple mixed signals are represented in Figure 6. The proposed DFNN over FFNN training technique found the better average accuracy of three mixed signals. In that mixed signal-3 gives highest accuracy rate of 98.97 % using DFNN over 95.95 % of FFNN with improvements, when compared to other two mixed signals. The following Table.3 indicates the outcoems of trained network. It is observed that, the average accuracy of three mixed signals using DFNN for formal verification is 97.55 %. Similarly for FFNN, the 93.96% is the average accuracy of three mixed signals while performing for formal verification, which clearly indicates that DFNN training network which gives better outcomes and satisfactory over FFNN training network with an improvement of nearly 3.7% of accuracy and which is applicable to do formal verification in system on chip level, and also for complex digital modelling verification.

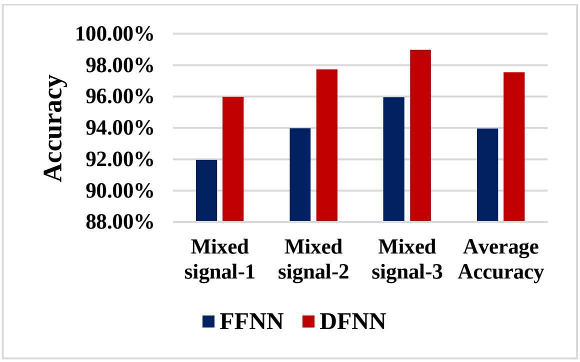

Figure 6. Formal verification outcomes with an accuracy of multiple mixed signals

4. CONCLUSION

This paper has presented a cost-effective, optimized solution to formal verification of multiple mixed signals using DFNN over FFNN on the hardware platform. In order to reduce the complexity of mixed signals in SOC design level, formal verification is best-suited solutions to improve the accuracy over error rate The proposed designs gives the solutions to the challenges is verification across SOC for multiple mixed signals. The proposed work is designed over a differentially fed neural network (DFNN) which provides the optimal solutions for performing formal verification of different types of mixed signals. The proposed Formal verification model of the both DFNN and FFNN Training modules includes Multiple Mixed signal generations, Implemented design using DFNN or FFNN, both the Reference Designs and formal verification using equivalence checking. The RTL-RTL verification is done for both the networks. The chip level constraints improvement done by using DFNN which includes area optimized by nearly 7% in terms of Slice LUT's, Total power consumption on-chip level is nearly 5 % and operating frequency of nearly 36 % over FFNN. The performance of DFNN in terms of accuracy improvement is 3.7% over FFNN.

REFERENCES

[1] R. E. Bryant, et al., "Limitations and challenges of computer-aided design technology for CMOS VLSI," *Proceedings of the IEEE*, vol/issue: 89(3), pp. 341-365, 2001.

[2] M. S. Erden, et al., "A review of function modeling: Approaches and applications," *Ai Edam*, vol/issue: 22(2), pp. 147-169, 2008.

[3] M. H. Zaki, et al., "Formal verification of analog and mixed-signal designs: A survey," *Microelectronics journal*, vol/issue: 39(12), pp. 1395-1404, 2008.

[4] B. Wielinga and G. Schreiber, "Configuration-design problem solving," *IEEE Expert*, vol/issue: 12(2), pp. 49-56, 1997.

[5] S. Gupta, et al., "Towards formal verification of analog designs," *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design. IEEE Computer Society*, 2004.

[6] K. Kundert and H. Chang, "Verifying all of an SOC-analog circuitry included," *IEEE Solid-State Circuits Magazine*, vol/issue: 1(4), 2009.

[7] P. Gang, "Behavioral modeling and simulation of analog/mixed-signal systems using verilog-ams," *Information, Computing and Telecommunication, 2009. YC-ICT'09. IEEE Youth Conference on. IEEE*, 2009.

[8] X. Yang, et al., "Mixed-signal system-on-a-chip (soc) verification based on systemverilog model," *System Theory (SSST), 2013 45th Southeastern Symposium on. IEEE*, 2013.

[9] Vidhya D. S. and M. Ramachandra, "Research Trends in Formal Verification Process for Analog and Mixed Signal Design," *International Journal of Computer Applications*, vol/issue: 109(11), Jan 2015.

[10] G. S. Harinarayan, et al., "Automated Full Chip SPICE simulations with self-checking assertions for last mile verification & first pass Silicon of mixed signal SoCs," *System-on-Chip Conference (SOCC), 2016 29th IEEE International*. IEEE, 2016.

[11] Vidhya D. S. and M. Ramachandra, "A Novel Design in Formal Verification Corresponding to Mixed Signals by Differential Learning," *Computer Science On-line Conference*. Springer, Cham, 2017.

[12] S. Chitti, et al., "A Unique Test Bench for Various System-on-a-Chip," *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 7(6), pp. 3318-3322, 2017.

[13] A. Bouziane, et al., "Towards an Optimal Speaker Modeling in Speaker Verification Systems using Personalized Background Models," *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 7(6), pp. 3655-3663, Dec 2017.

[14] T. R. J. Kumariand and H. S. Jayanna, "Limited Data Speaker Verification: Fusion of Features," *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 7(6), pp. 3344-3357, Dec 2017.

[15] M. Wojciechowski, "Solving differential equations by means of feed-forward artificial neural networks," *International Conference on Artificial Intelligence and Soft Computing. Springer, Berlin, Heidelberg*, 2012.

[16] J. Ilonen, et al., "Differential evolution training algorithm for feed-forward neural networks," *Neural Processing Letters*, vol/issue: 17(1), pp. 93-105, 2003.

[17] Manjunath R., et al., "Differential learning algorithm for Artificial Neural Networks," *International Journal of Computer Applications*, vol/issue: 1(1), pp. 65-70, Feb 2010.

BIOGRAPHIES OF AUTHORS

Vidhya.D.S. is a Phd candidate in Electronics and communication engg at Assam Don Bosco university, Guwahati She received her M tech degree in VLSI design and Embedded system from VTU university. She has more than 14 years of teaching experience Her research are working different application with Embedded Micro controllers, signal processing and VLSI.

Manjunath Ramachandra got his PhD from Bangalore university. He has a blend of academic and industry experience over two decades. He has authored 170 Research papers in international conferences and journals and a book. His areas of interest include hardware, signal processing, artificial intelligence etc.