ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Usage of Artificial Neural Networks in Power Estimation of Benchmark Circuits

D.Prasad
Associate Professor,
Department of ECE,
Ramanandatirtha Engineering College,
Nalgonda.

P.Rahul Reddy
Associate Professor,
Department of ECE,
Ramanandatirtha Engineering College,
Nalgonda.

Abstract:

Power estimation at an earlier stage is important in VLSI circuits, because it has a significant impact on the reliability of these circuits. Power estimation is a trade -off between precision of estimation and estimation time. Simulation based power estimation techniques are time consuming. This work reports an artificial neural network based method for power estimation of ISCAS'89 Benchmark circuits, by employing Back Propagation Neural Network (BPNN) and Radial Basis Function Neural Network (RBFNN). This method can estimate power quickly and precisely from Inputs and Outputs (I/O) and gate information of the VLSI circuit, without requiring detailed structure of the circuit and its interconnection. Power estimation results reported in the literature for International Symposium on Circuits and Systems 1989 (ISCAS' 89) Benchmark circuits are used to the train the neural networks. The power estimate results for the tested circuits are validated by performing regression analysis. The BPNN is trained with various training functions and a comparative study on various training algorithms for power estimation is made. RBFNN is also trained and tested with the same data sets and the results are compared with power estimation results of BPNN.

Keywords: ***Back Propagation Neural Network (BPNN), Radial Basis Function Neural Network (RBFNN), Benchmark Circuit and Very Large Scale Integration (VLSI).***

1. Introduction

It is impractical to stimulate a large VLSI circuit exhaustively with all possible representative input vectors, to measure power. Hence power is measured for a specific set of random vectors, and is termed as average power consumption. Power dissipation depends on the operating environment of the circuit, namely the input vectors being fed in. Methods to estimate average power can be classified into two major categories namely simulative and non-simulative. Simulation based category includes Monte Carlo approach proposed by Burch et al (1993). In this approach, the circuits are simulated for different input combinations and the average power is calculated as an average of the simulated values. Exhaustive simulation is done to get accurate results because it takes care of spatial and temporal correlations within the circuit. However, it is time consuming. Further, this technique is most preferable for combinational circuits but not for sequential circuits. Monte Carlo technique decouples the combinational portion of the sequential circuit from the flip-flops and analyze them separately which leads to inaccuracies in power estimation.

Non simulative based approaches are classified as probabilistic and statistical approaches. Under Probabilistic category, an algorithm to propagate the transition density values from the inputs throughout the circuit was proposed by Najm (1993). This method overcomes the pattern dependency problem by using the probabilities to describe the set of all possible logic signals. In-order to achieve good accuracy, one must model the correlations among these logic signals

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

which can be expensive. Hence this technique usually trades off accuracy for speed. An improved Monte Carlo method was proposed by Saxena et al (1997) which considered the correlations in time and space between the inputs, internal nodes and state nodes. This method saves time compared to only simulation method.

Under statistical category, a Distribution Independent Power Estimator (DIPE) technique for estimating average power in sequential circuits was proposed by Yuan et al (1998). In this method the circuit is simulated for a random set of input vectors and the power value obtained during simulation are observed. The simulation for switching activity is terminated using a stopping criterion based on central limit theorem. Under statistical category, a stratified random sampling approach to estimate power was proposed by Ding et al (1998). The statistical methods mentioned above are confidence interval based and hence require more iteration to converge. Least square estimation of average power proposed by Murugavel et al (2002) minimizes the mean square error value during each iteration and is more time saving when compared to Monte Carlo approach. The least square technique minimized the mean square error during each iteration by using two statistical algorithms namely Sequential Least Square (SLS) and Recursive Least Square (RLS).

Bayesian Networks to estimate the switching activity in VLSI circuits proposed by Bhanja and Ranganathan (2003) encapsulates all the dependencies both in internal nodes and inputs within a reasonable time and accuracies. Genetic Algorithm based method for peak power estimation proposed by Yi-Ling Liu et al (2009), derives a small set of input patterns associated with peak power. However correlation between Peak Switching Frequency (PSF) and Peak Power value are not reported. PSF is used to represent the peak power consumption in VLSI circuits. All the above methods require the detailed structure of the VLSI circuit and partial simulation results. This accounts for a significant amount of time, which is in proportion with the scale of integration of the circuit.

Neural Network based power estimation method for ISCAS'89 Benchmark circuits proposed by Ligang Hou et al (2006) uses Back Propagation (BP) algorithm with network training function named Levenberg-Marquardt (trainlm). The regression analysis for the best case reported has Slope 0.915, Y-intercept of -0.00963 and Regression value of 0.994. The slope value and the regression value in this method deviates from the ideal power estimator by 8.5% and 0.6% respectively.

2. Proposed method for power estimation

text The proposed Neural Network method for power estimation consists of two phases. In the first phase the training of the network is carried out and in the second phase the testing of the network is done. The steps in training phase and testing phase are discussed below.

Training Phase

Step 1: Input vectors extracted from ISCAS'89 Benchmark circuits are used to train the neural network.

Step 2: Each parameter in the input vectors and their corresponding target vectors are normalized. The normalization range is between -1 to +1, when the activation function chosen for neurons in the second hidden layer, third hidden layer and output layer of the proposed BPNN are Tan-sig. The normalization range is between 0 to 1, when the activation function chosen for neurons in the second hidden layer, third hidden layer and output layer of the proposed BPNN are Log-sig. For RBFNN the normalization is done between -1 and +1.

Step 3: Normalized input vectors and their corresponding normalized target vectors are used to train the neural network.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Testing Phase

Step 1: Input vectors left out in the training process are used for testing.

Step 2: The various parameters in the input test vectors are also normalized in the same manner as that of normalization employed during training.

Step 3: Network will generate normalized outputs vectors for these normalized test input vectors.

Step 4: Normalized output vectors are converted back to their original value by applying the reverse normalization process.

Step 5 : The output vectors obtained for these test inputs are compared with the expected outputs and validation of the estimated results is performed using Regression analysis The database used for training the neural networks is obtained from ISCAS'89 Benchmark circuits. 20 Benchmark circuits from ISCAS'89 are used to train the BPNN and RBFNN. The input vectors for the different circuits contain information regarding the number of inputs, outputs, D flip-flops, inverters, gates, AND gates, NAND gates, OR gates and NOR gates.

Table 1 lists the training data set used for BPNN and RBFNN. In Table 5.1, IN represents the number of inputs, OUT represents the number of outputs, DFF represents the number of D flip-flops and INV represents the number of inverters. The output vector is the Monte Carlo power values reported by Saxena et al (1997) for the various ISCAS'89 Benchmark circuits. The neural networks are tested for various ISCAS'89 Benchmark circuits reported in Table 2.

Table.1 Training Data Set for Neural Networks

Benchmark Circuit	IN	OUT	DFF	INV	GATE COUNT	AND	NAND	OR	NOR	Monte Carlo (MC) Power mw/ MHz
S208	10	1	8	38	66	21	15	14	16	0.00698
S298	3	6	14	44	75	31	9	16	19	0.00912
S349	9	11	15	57	104	44	19	10	31	0.01856
S386	7	7	6	41	118	83	0	35	0	0.0162
S400	3	6	21	58	106	11	36	25	34	0.01065
S420	18	1	16	78	160	49	29	28	34	0.00903
S444	3	6	21	62	119	13	58	14	34	0.01172
S713	35	23	19	254	139	94	28	17	0	0.03743
S820	18	19	5	33	256	76	54	60	66	0.02831
S838	34	1	32	158	288	105	57	56	70	0.01292
S953	16	23	29	84	311	49	114	36	112	0.02458
S1238	14	14	18	80	428	134	125	112	57	0.06347
S1423	17	5	74	167	490	197	64	137	92	0.07181
S1494	8	19	6	89	558	354	0	204	0	0.06018
S5378	35	49	179	1775	1004	0	0	239	765	0.23357
S9234	19	22	228	3570	2027	955	528	431	113	0.28004
S15850	14	87	597	6324	3448	1619	968	710	151	0.51991
S35932	35	320	1728	3861	12204	4032	7020	1152	0	1.22048
S38417	28	106	1636	13470	8709	4154	2050	226	2279	1.14518
S38584	12	278	1452	7805	11448	5516	2126	2621	1185	1.87987

Table .2 Test Data for Neural Networks

Benchmark Circuit	IN	OUT	DFF	INV	GATE COUNT	AND	NAND	OR	NOR
S344	9	11	15	59	101	44	18	9	30
S382	3	6	21	59	99	11	30	24	34
S641	35	24	19	272	107	90	4	13	0
S1488	8	19	6	103	550	350	0	200	0
S13207	31	121	669	5378	2573	1114	849	512	98

3. Neural Network Design and Simulation

The Neural Network architectures namely BPNN and RBFNN have been designed and implemented in Matlab for power estimation.

3.1 BPNN Design and Simulation

A four layer BPNN with three hidden layers and one output layer is designed. Activation functions namely, Pure Linear is applied to the first hidden layer and Tan-sig or Log-sig is applied to the rest of the hidden layers and output layer. BPNN is trained using eleven different training algorithms. Error goal of the BPNN

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

is fixed at 10-8. The network was trained for the following variations in inputs and layer sizes.

- Number of input chosen as either 8 or 9.
- Number of neurons in the Ist hidden layer varied between 7 and 10.
- Number of neurons in the IInd hidden layer varied between 12 and 16.
- Number of neurons in the IIIrd hidden layer varied between 14 and 17.

Tables 3 and 4 list the Regression results for training functions with variable learning rate and momentum constant Traingdm and Traingdx. Learning rate is varied in the range 0.3 to 0.8. The momentum constant is varied between
0.1 and 1.

Table.3 Regression Results for Traingdm and Traingdx with Tan-Sigmoidal Activation Function

Traingdm							
Input	Layer Size	Learning Rate	Momentum Constant	Epochs	Slope (M)	Y-intercept (B)	Regression (R)
9	8:15:15:1	0.35	0.9	400	0.9506	0.0017	0.9972
9	9:15:15:1	0.2	0.2	400	0.9906	0.0027	0.9936
9	8:14:15:1	0.2	0.1	400	1.0823	5.76E-04	0.9953
9	8:13:14:1	0.1	0.7	400	0.9443	0.0054	0.9993
8	9:15:15:1	0.1	0.9	400	1.0412	0.0027	0.9999
Traingdx							
Input	Layer Size	Learning Rate	Momentum Constant	Epochs	Slope (M)	Y-intercept (B)	Regression (R)
9	8:14:15:1	0.2	0.9	225	0.6776	0.0039	0.9957
9	8:13:15:1	0.4	0.9	265	0.7221	0.0065	0.9974
9	8:15:15:1	0.4	0.6	225	2.3382	-0.0386	0.9946
9	8:15:15:1	0.4	0.9	225	-6.79E-09	0.0016	-0.2902
8	8:14:15:1	0.2	0.9	225	1.0117	5.45E-05	0.9967
8	8:13:15:1	0.4	0.9	265	1.045	0.0051	0.9999
8	8:15:15:1	0.4	0.6	225	1.0247	0.0066	0.9993
8	8:15:15:1	0.4	0.9	225	1.0148	0.0012	0.9983

Table 3 indicates the regression results when the activation function for neurons in second hidden layer, third hidden layer and output layer of the proposed BPNN are Tan-sig. Table 4 indicates the regression results when the activation function for neurons in second hidden layer, third hidden layer and output layer of the proposed BPNN are Log-sig. . The highlighted rows in Tables 3and 4 indicate the best possible regression values obtained for Traingdx training function.

Table.4 Regression Results for Traingdm and Traingdx with Log-Sigmoidal Activation Function

Traingdm							
Input	Layer Size	Learning Rate	Momentum Constant	Epochs	Slope (M)	Y-intercept (B)	Regression (R)
9	9:15:15:1	0.4	0.7	1000	1.0213	0.0147	0.9803
9	9:15:15:1	0.4	0.77	1000	0.6596	0.0621	0.994
9	8:15:15:1	0.8	0.6	500	1.6386	-0.0186	0.9956
9	7:14:15:1	0.46	0.5	775	0.922	0.0248	0.998
8	9:15:15:1	0.4	0.7	1000	1.0045	0.0121	0.9956
8	7:14:15:1	0.4	0.9	800	1.0328	0.0162	0.9954
8	8:15:15:1	0.8	0.6	500	1.1489	-0.0105	0.9971
8	7:14:15:1	0.46	0.5	775	1.0668	0.0327	0.9908
Traingdx							
Input	Layer Size	Learning Rate	Momentum Constant	Epochs	Slope (M)	Y-intercept (B)	Regression (R)
9	7:15:15:1	0.3	0.9	225	0.7099	0.014	0.9979
9	7:14:15:1	0.4	0.89	225	1.2209	-0.0071	0.9935
9	7:16:14:1	0.3	0.9	225	1.1618	-0.0016	0.9943
9	7:15:15:1	0.3	0.89	225	1.8022	-0.0115	0.9937
8	7:15:15:1	0.3	0.9	225	1.0049	-2.32E-04	0.9957
8	7:14:15:1	0.4	0.89	225	1.0443	2.77E-04	0.9982
8	7:16:14:1	0.3	0.9	225	1.0489	-8.35E-04	0.9956
8	7:15:15:1	0.3	0.89	225	1.0951	-2.90E-03	0.9954

From Tables 3 and 4, it is observed that most of the readings for momentum based algorithms have actual slope values that deviate from the ideal value of slope by a significant amount when nine inputs are considered for training. Hence we can conclude that momentum based algorithms work well when the training process has only 8 inputs. The parameter 'GATE COUNT' does not influence the training process in a better way to achieve power estimation.

3.2 RBFNN Design and Simulation

The error goal is chosen as 5x10-11. The spread value is varied between 0 and 1. The number of inputs is chosen as either 8 or 9. The epochs were varied and the regression results are observed. Table 5 shows the regression results for RBFNN.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Table.5 Regression Results for RBFNN

Input	Spread Value	Epochs	Slope (M)	Y-itercept (B)	Regression (R)
9	0.5	60	1.0467	2.31E-05	1
9	0.495	60	1.0168	9.08E-04	1
9	0.498	60	1.0347	3.79E-04	1
9	0.501	60	1.0527	-1.56E-04	1
8	0.331	60	0.9432	0.0033	0.9998
8	0.28	60	0.9284	0.0038	0.9998
8	0.35	60	0.9386	0.0034	0.9998
8	0.324	100	0.9438	0.0033	0.9998

It is observed that the Regression results obtained for RBFNN are much closer to values of an ideal estimator.

RBFNN with a spread value of 0.495 with 60 epochs and 9 inputs is the best estimator for power estimation. Table 6 lists the comparison of proposed work with Neural Network based architectures for power estimation existing in literature.

Table 6 Comparison of the Regression Analysis of various Training Functions with Trainlm Existing in the Literature

Training Function	Layer Size	Number of Inputs	Epoch	Activation Function	Slope (M)	Y-intercept (B)	Regression (R)
Trainlm Ligang Hou et al (2006)	6:15:15:1	9	2139	Tan-Sig	0.915	-0.000963	0.994
Trainlm Ligang Hou et al (2006)	9:15:15:1	9	879	Tan-Sig	0.464	0.00882	0.753
Traingdx	8:13:15:1	8	265	Tan-Sig	1.045	0.0051	0.9999
Traingdx	7:14:15:1	8	225	Log-Sig	1.0443	2.77E-04	0.9982

4. Conclusion

For attaining good power estimation the ideal values of R, M and B are 1, 1 and 0 respectively. The learning rate, momentum constant, layer size and training functions of the BPNN are varied. Traingdx outperforms Traingdm under training functions with momentum constant. RBFNN works well for 9 inputs. RBFNN with 9 inputs, 60 epochs and spread value of 0.495 yields regression results which are very close to ideal values. Traingdx training function outperforms trainlm proposed in the literature in terms of better regression values.

5. References

1. Burch R., Najm F., Yang P. and Trick T. (1993), 'A Monte-Carlo Approach for Power Estimation', IEEE Transactions on VLSI systems, Vol. 1, No. 1, pp. 63-71.

2. Najm F. N. (1993), 'Transition density : A new measure of activity in digital circuits', IEEE Transactions on computer-aided design of integrated circuits and systems, ISSN 0278-0070, Vol. 12, pp. 310-323.

3. Saxena V., Najm F. M. and Hajj I. N. (1997), 'Monte Carlo Approach for Power Estimation in Sequential circuits', in Proceedings of European Design and Test Conference, Paris, France, pp. 416-420.

4. Murugavel A., Ranganathan N., Chandramouli R. and Chavali S. (2002), 'Least-square estimation of average power in digital CMOS circuits', IEEE Transactions on VLSI systems, Vol. 10, No. 1, pp. 55-58.

5. Bhanja S. and Ranganathan N. (2003), 'Switching activity estimation of VLSI circuits using Bayesian networks', IEEE Transactions on VLSI System, Vol. 11, No. 4, pp. 558-567.

6. Ligang Hou, Liping Zheng and Wuchen Wu (2006), 'Neural Network Based VLSI Power Estimation', 8th International Conference on Solid-State and Integrated Circuit Technology, Shanghai, China, pp. 1919- 1921.

7. Yi-Ling Liu, Chun-Yao Wang, Yung-Chih Chen and Ya-Hsin Chang (2009) 'A Novel ACO-based

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Pattern Generation for Peak Power Estimation in VLSI Circuits', IEEE international Symposium on Quality Electronic Design , USA, pp. 317-323.

8. http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/nnet.pdf.