Instruction Based Power Estimation Method in AURIX Microcontroller

Kiran Guruprasad Shetty P S
Electronics and Communication Department
R V College of Engineering
Bengaluru, India
kiranguruprasadshetty@gmail.com

Dr. Ravish Aradhya H V
Electronics and Communication Department
R V College of Engineering
Bengaluru, India
ravisharadhya@rvce.edu.in

Abstract: *Power estimation is a very prominent aspect in microcontrollers which aims to to be more efficient in terms of power. A new method of estimation of power based on the execution of instruction in AURIX, which is an automotive micro- controller is proposed. The main aim of this method is to estimate the power in perspective of program(software) or instruction level which is constantly processed in microprocessor which is more accurate when compared with the previous methodologies. The estimation is done based on some set of instructions which is used in AURIX for Data transfer/storing in to memories, Data processing and Data Execution for various application. Most of the previous methodologies are all not accurate due to the abstraction levels.*

Keywords— Power Estimation Method (PEM), Trace data, Trace buffer, Trace IP

I. INTRODUCTION

The next generation microcontrollers used in automobiles should specifically estimate power consumption before the code execution on board. This feature enables to predict required power for the drive train. There has been tremendous development in the field of embedded system to predict the power as accurate as possible [2][3]. Most of the method takes in hardware power consumption but it compromises on software part [4]. So, a new trend has started where power estimation is performed based on software. Chips like Intel 486DX2 enables us to exactly predict the power consumption during the execution of instruction [5][6]. This power consumption value is 40% of the maximum power consumption. Various methodologies in low power VLSI design have been proposed in different abstraction levels such as structural, logical and algorithmic method. All these techniques calculate the dynamic power dissipation. When system is working at a particular frequency, voltage and temperature, power consumption is usually specified as static power consumption [1]. So, design of mixed hardware and software system is a critical process as time to market and competition is increasing day by day. Most of the methodologies in power reduction techniques always estimates at higher abstraction level. In [9], 32-bit RISC processor is shown with power estimation model based on software with various frequencies. Most modern designers are designing compiler which almost minimizes power as much as possible. The power estimation method estimates the power of the whole system by utilizing the different methods related to the processing of the microcontroller [7][8]. This method usually estimates the power of the commands executed in the microprocessor when software/application is running. So the same technique can be implemented to different processors with varying software application. The above process can also be used in different areas of SOC and applications [10][11].

II. PROPOSED POWER ESTIMATION METHOD

The block diagram of trace IP which usually records transaction like writing, reading, fetching done is recorded is shown in figure 1. IT also includes different trace signals with its time data, memory location and cycle time required to execute an instruction. So, the data which is captured by the trace IP for a individual test case executed on a processor can be seen by MCDS trace viewer. This MCDS trace viewer usually contains the trace data which were captured earlier and stored in the .csv format with current measurement in mA. The different transaction captured by the trace IP at different times is shown in figure 1 as T signal. Each trace source is having a block for observation which contains trace units or sampling unit to generate trace messages. These trace messages are stored in data buffers which are synchronized by the cross connect unit and sent back to trace host. All the trace messages with its timing information is based on the trace clock. The memory efficient message aligner present in the memory controller is used to sort the time of the trace messages in trace buffer.

In figure 2, it can be observed the transaction flow of the trace buffer. The proposed method estimates the power of the given program which gives us the instruction level power estimation. The program consists of various number of com- mands/instructions which are executed by the CPU. To estimate the total power consumption individual value of power of a particular instruction is first estimated. This power value is termed as Instruction cost. In the next section implementation of the above concept is explained.

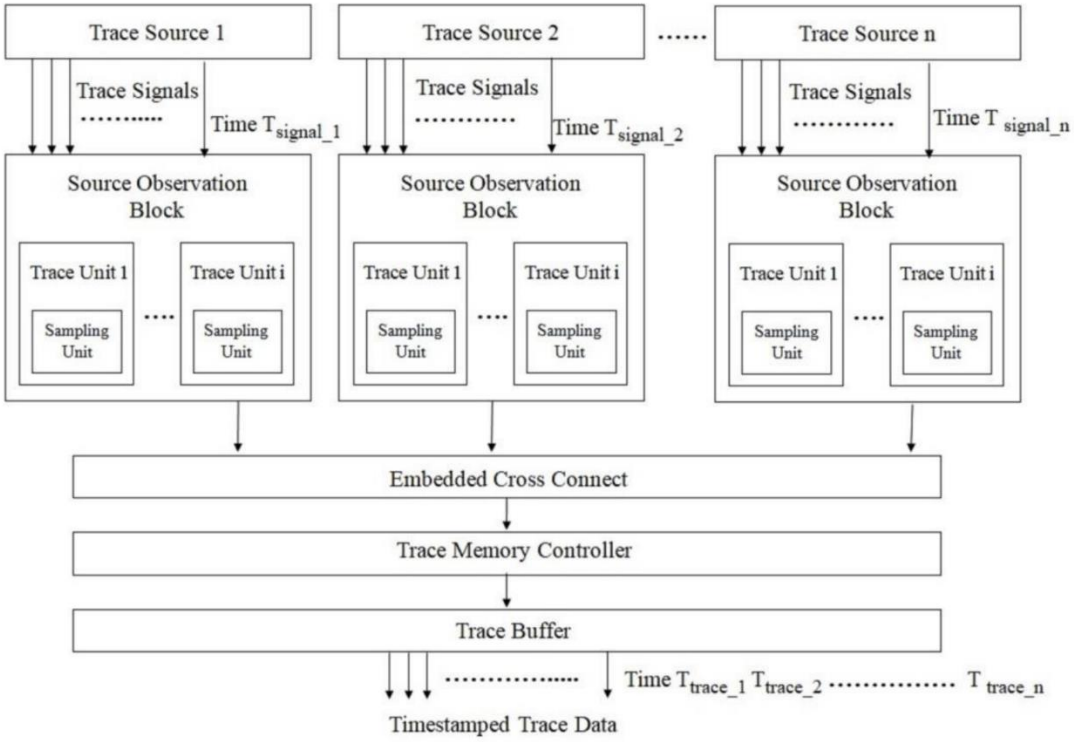


Fig. 1. Trace IP

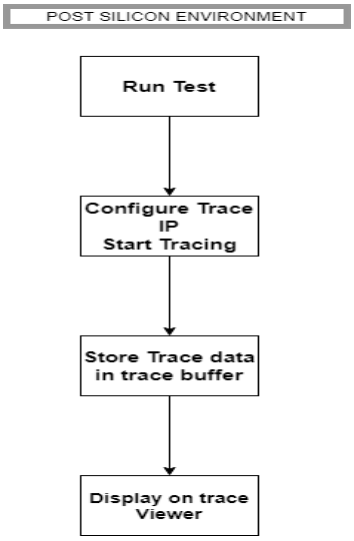


Fig. 2. Transaction Flow

III. DESIGN ARCHITECTURE & IMPLEMENTATION

The creation of trace data and current measurement from the trace buffer are categorized into two categories such as validation data and training data. This validation data is also called as test data. Initially, data obtained from training is used to design the model. So, to estimate the power prediction error for a particular model which is selected by the training data, validation data is used. In figure 3, it is observed the architecture of power estimation method. From the figure 3, the calculation of I err, where it is fed to the linear regression model which is called learning algorithm. This model is made to give the proper/accurate estimation of current value.

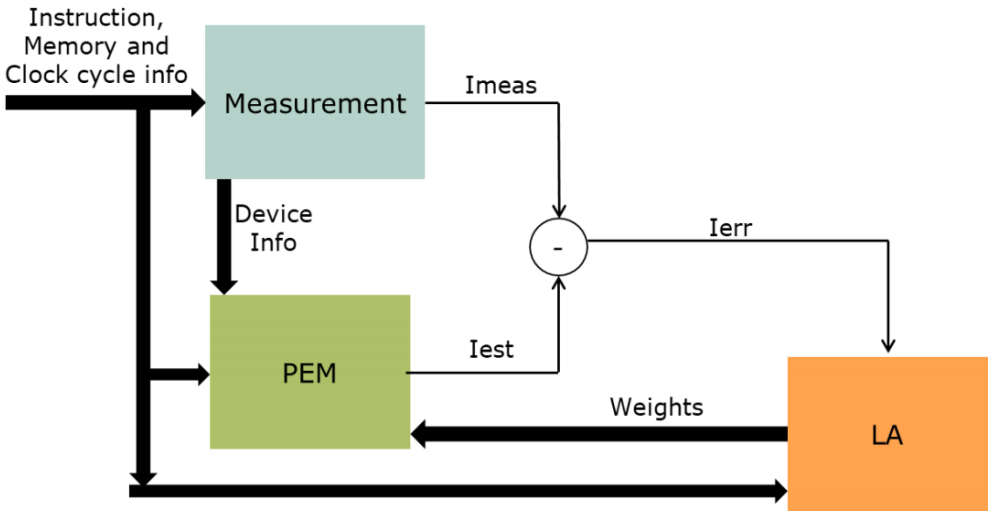


Fig. 3. Power Estimation Method

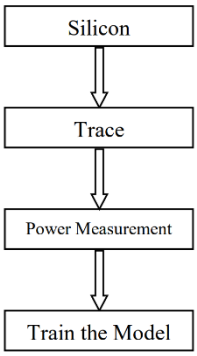


Fig. 4. Flow diagram

The flow diagram of the power estimation method is given below and it is shown in figure 4. The detailed steps are shown in figure 5. It mainly consists of four steps.

They are

a) Writing a test case and obtaining the debug data after executing the test case.

b) Tracing the current fluctuation for various kinds of instructions and repeating the previous step.

c) Capturing the debug data (Current) and feed the value to the model.

d) Estimate the data according to the learning algorithm.

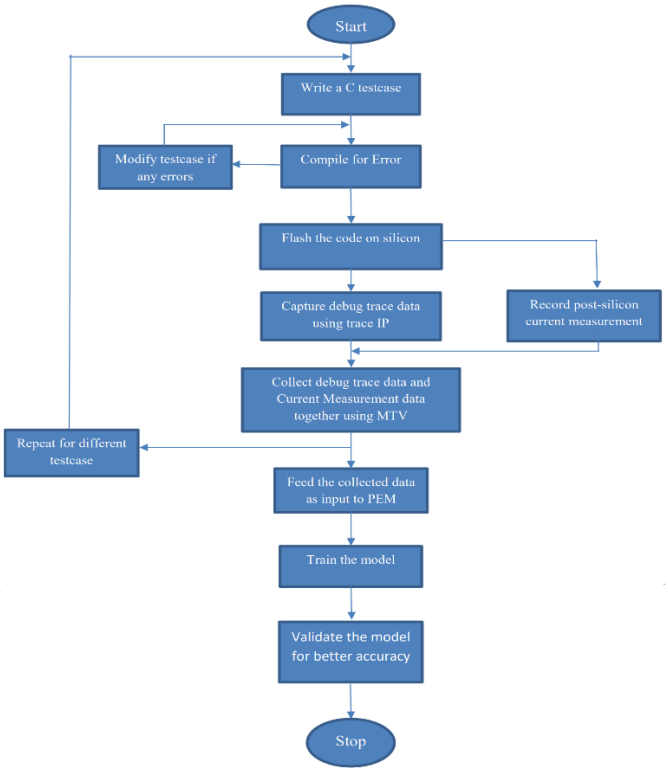


Fig. 5. Detailed flow diagram

IV. MEASUREMENTS & RESULTS

Transaction recorded(captured) in the third step of the flow is shown in figure 6 and figure 7. In figure 6, constant data of 32 bit is written into a particular memory by one of the cores of silicon. In figure 7, constant data of 32 bit is read into a particular memory by one of the cores of silicon.

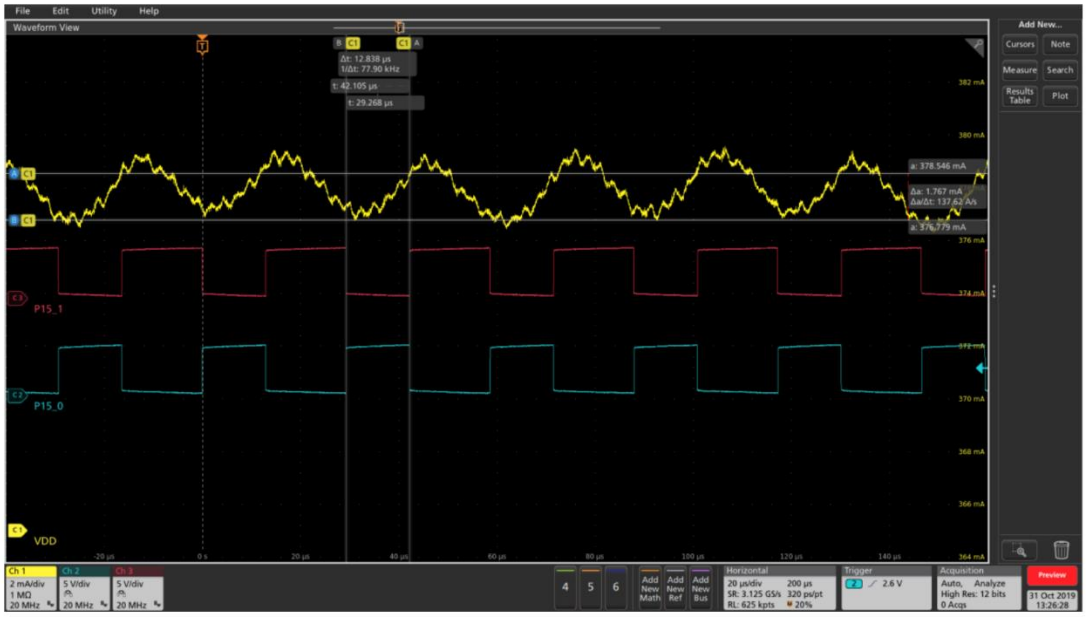


Fig. 6. Constant data of 32 bit is written

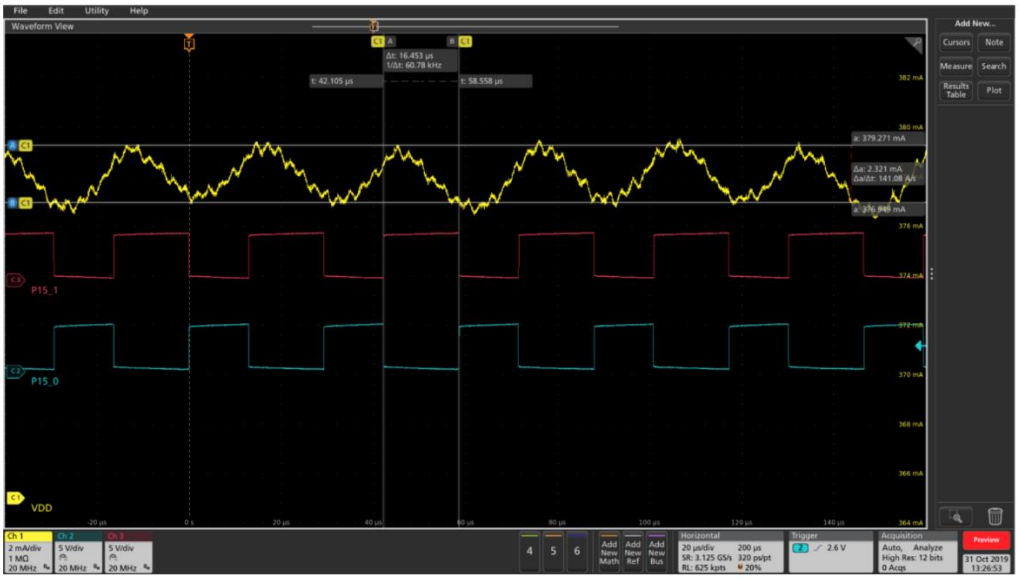


Fig. 7. Constant data of 32 bit is read

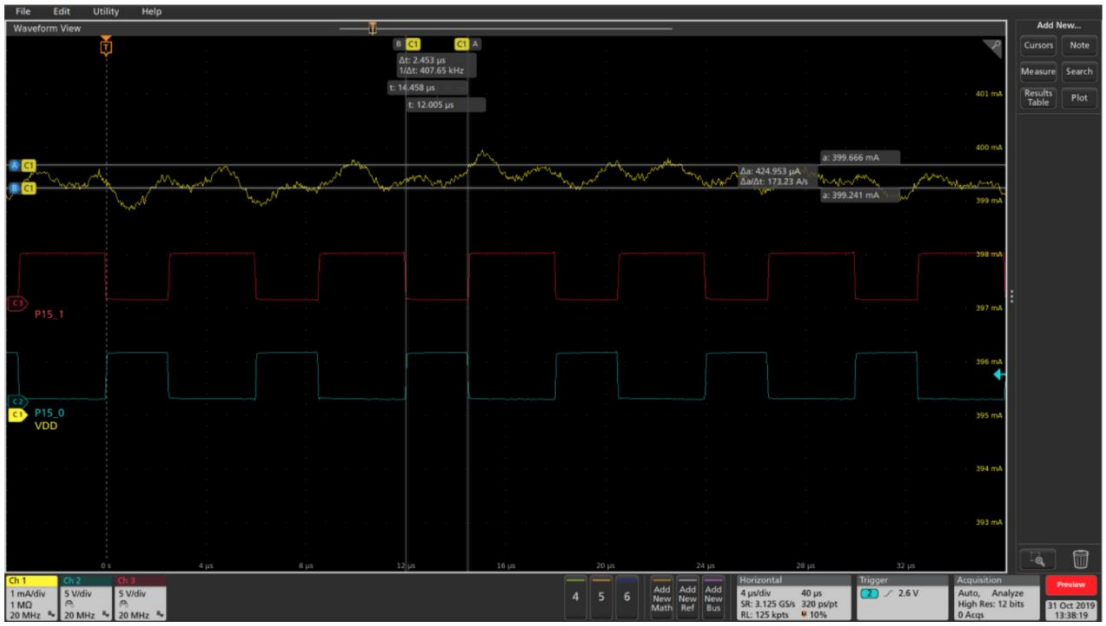


Fig. 8. Random data of 32 bit is written

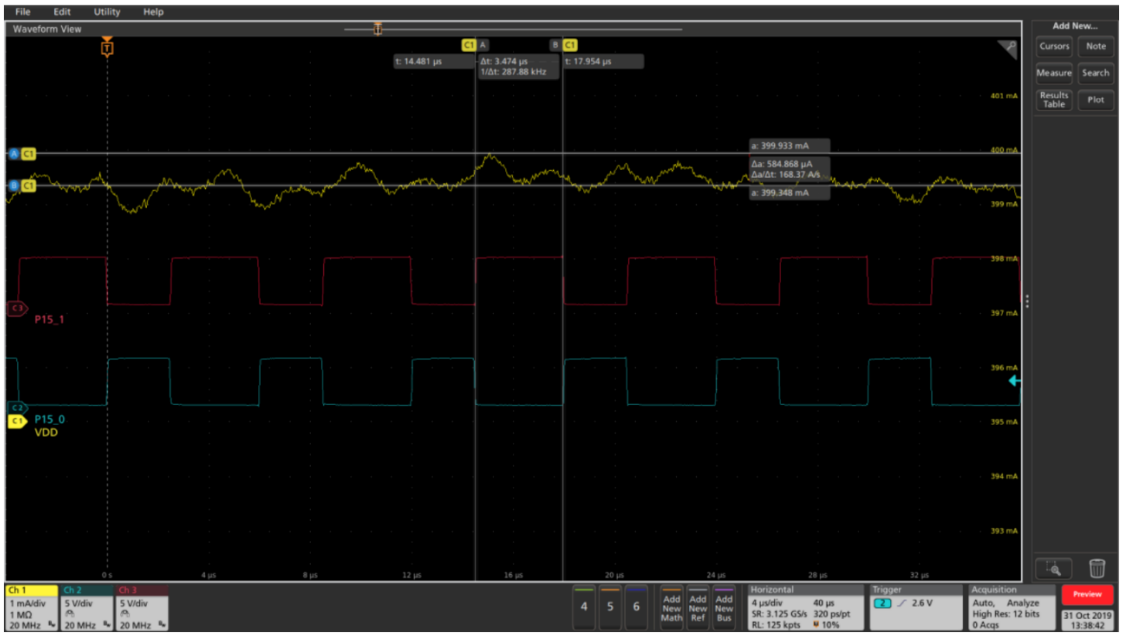
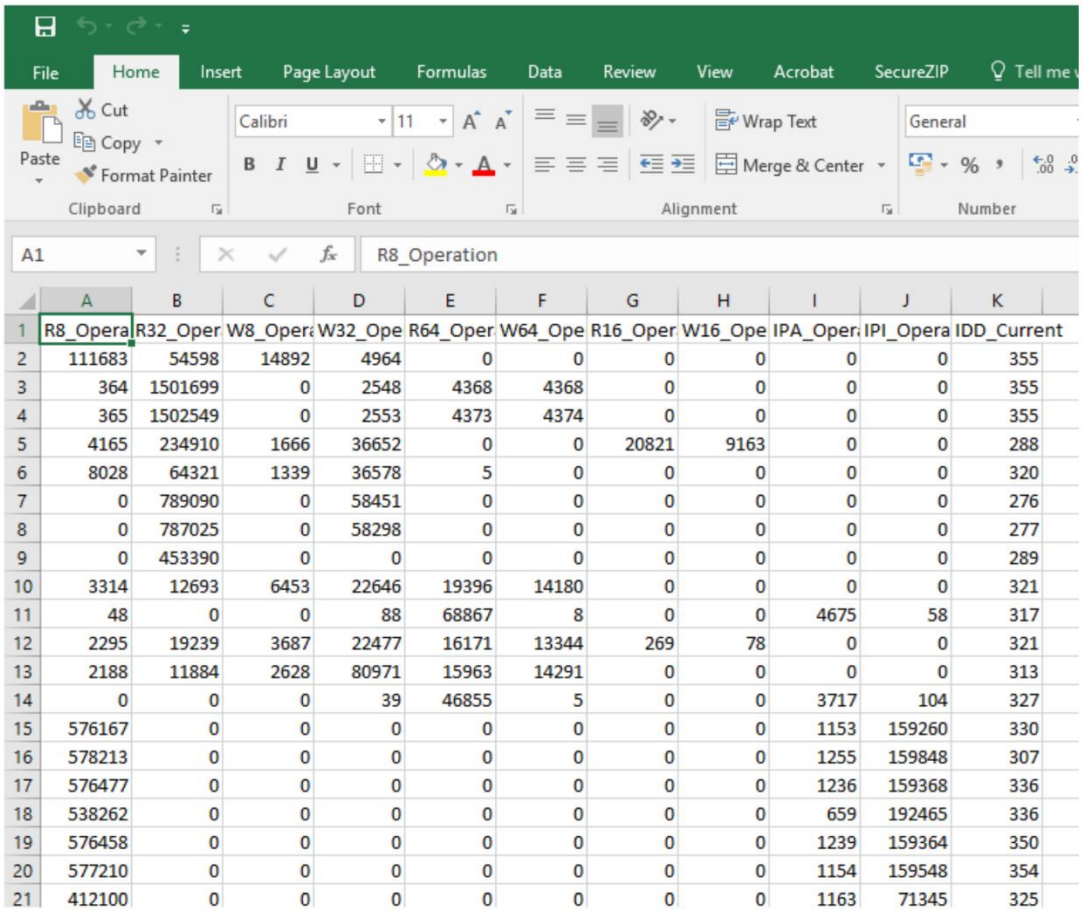


Fig. 9. Random data of 32 bit is read

From these captured data, different input parameters for building the model is shown in figure 10 where various kinds of operands are listed in columns with its current measurements in rows. Initially around 40 test cases are built and accordingly data set, buffer and model are created with the size of validation as 0.25 and 180 epochs. This gave 91.22% accuracy.



	A	B	C	D	E	F	G	H	I	J	K
1	R8_Opera	R32_Oper	W8_Opera	W32_Ope	R64_Oper	W64_Ope	R16_Oper	W16_Ope	IPA_Opera	IPI_Opera	IDD_Current
2	111683	54598	14892	4964	0	0	0	0	0	0	355
3	364	1501699	0	2548	4368	4368	0	0	0	0	355
4	365	1502549	0	2553	4373	4374	0	0	0	0	355
5	4165	234910	1666	36652	0	0	20821	9163	0	0	288
6	8028	64321	1339	36578	5	0	0	0	0	0	320
7	0	789090	0	58451	0	0	0	0	0	0	276
8	0	787025	0	58298	0	0	0	0	0	0	277
9	0	453390	0	0	0	0	0	0	0	0	289
10	3314	12693	6453	22646	19396	14180	0	0	0	0	321
11	48	0	0	88	68867	8	0	0	4675	58	317
12	2295	19239	3687	22477	16171	13344	269	78	0	0	321
13	2188	11884	2628	80971	15963	14291	0	0	0	0	313
14	0	0	0	39	46855	5	0	0	3717	104	327
15	576167	0	0	0	0	0	0	0	1153	159260	330
16	578213	0	0	0	0	0	0	0	1255	159848	307
17	576477	0	0	0	0	0	0	0	1236	159368	336
18	538262	0	0	0	0	0	0	0	659	192465	336
19	576458	0	0	0	0	0	0	0	1239	159364	350
20	577210	0	0	0	0	0	0	0	1154	159548	354
21	412100	0	0	0	0	0	0	0	1163	71345	325

Fig. 10. Captured current values for 91.22% accuracy with 0.25 validation size and epoch of 180

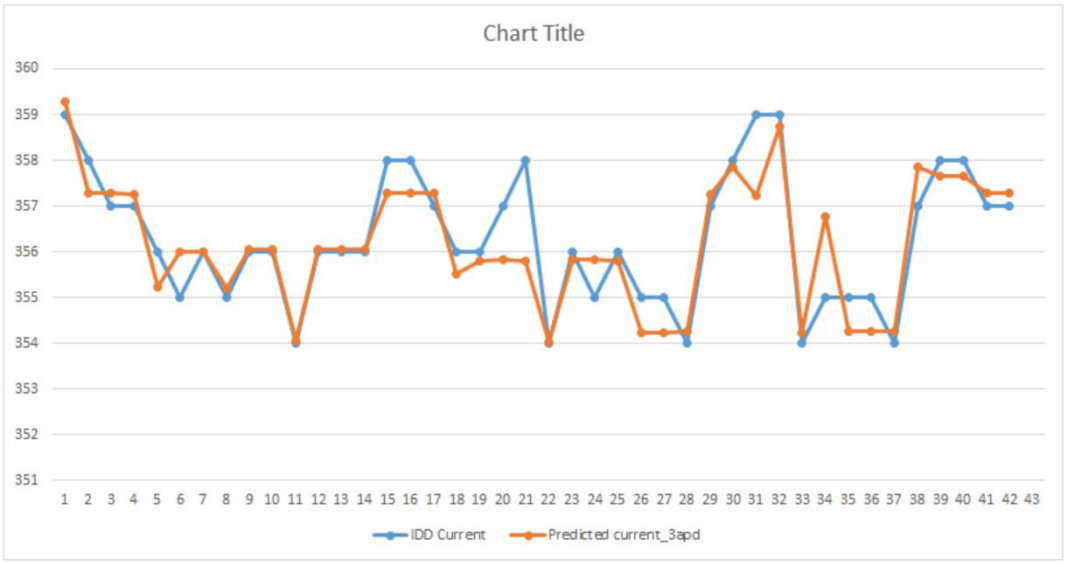
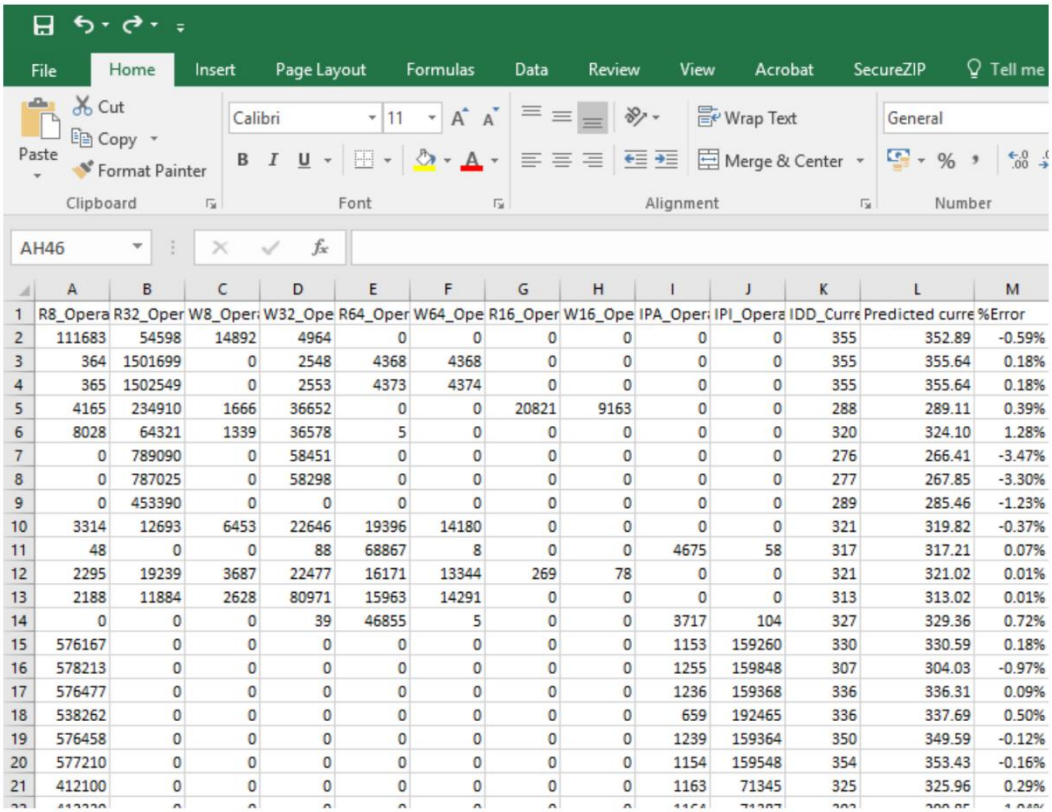


Fig. 11. graph of 91.22% accuracy with 0.25 validation size and epoch of 180

In figure 11, it is observed that the current predicted and the current measured graph with 91.22% accuracy. In figure 12, the number of test cases were increased to 110 and epochs was produced to 80 which gave an improved accuracy of 98.88%.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	R8_Opera	R32_Oper	W8_Oper	W32_Ope	R64_Oper	W64_Ope	R16_Oper	W16_Ope	IPA_Oper	IPI_Opera	IDD_Curre	Predicted curre	%Error
2	111683	54598	14892	4964	0	0	0	0	0	0	355	352.89	-0.59%
3	364	1501699	0	2548	4368	4368	0	0	0	0	355	355.64	0.18%
4	365	1502549	0	2553	4373	4374	0	0	0	0	355	355.64	0.18%
5	4165	234910	1666	36652	0	0	20821	9163	0	0	288	289.11	0.39%
6	8028	64321	1339	36578	5	0	0	0	0	0	320	324.10	1.28%
7	0	789090	0	58451	0	0	0	0	0	0	276	266.41	-3.47%
8	0	787025	0	58298	0	0	0	0	0	0	277	267.85	-3.30%
9	0	453390	0	0	0	0	0	0	0	0	289	285.46	-1.23%
10	3314	12693	6453	22646	19396	14180	0	0	0	0	321	319.82	-0.37%
11	48	0	0	88	68867	8	0	0	4675	58	317	317.21	0.07%
12	2295	19239	3687	22477	16171	13344	269	78	0	0	321	321.02	0.01%
13	2188	11884	2628	80971	15963	14291	0	0	0	0	313	313.02	0.01%
14	0	0	0	39	46855	5	0	0	3717	104	327	329.36	0.72%
15	576167	0	0	0	0	0	0	0	1153	159260	330	330.59	0.18%
16	578213	0	0	0	0	0	0	0	1255	159848	307	304.03	-0.97%
17	576477	0	0	0	0	0	0	0	1236	159368	336	336.31	0.09%
18	538262	0	0	0	0	0	0	0	659	192465	336	337.69	0.50%
19	576458	0	0	0	0	0	0	0	1239	159364	350	349.59	-0.12%
20	577210	0	0	0	0	0	0	0	1154	159548	354	353.43	-0.16%
21	412100	0	0	0	0	0	0	0	1163	71345	325	325.96	0.29%

Fig. 12. Captured current values for 98.88%

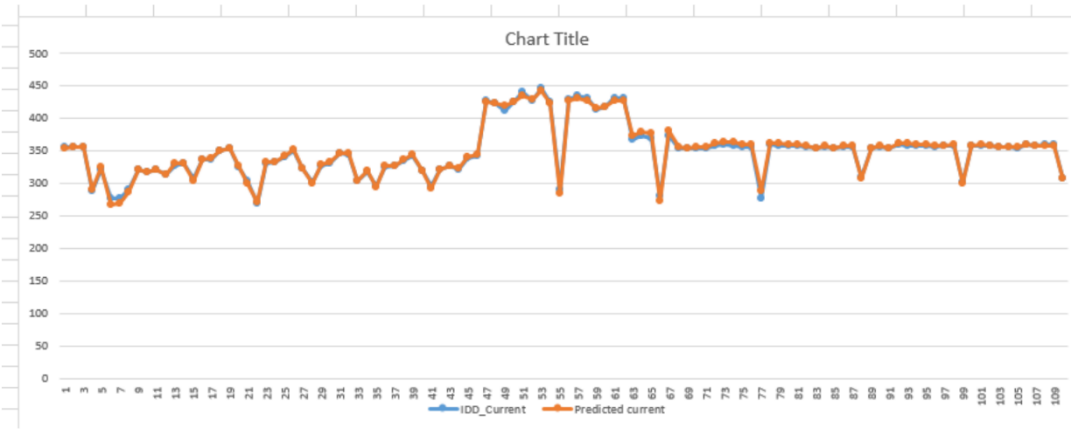


Fig. 13. Graph for Captured current values for 91.22%

In figure 13, it is observed that the current predicted and the current measured graph with 98.88% accuracy.

In figure 14, error definition for the previous model is shown with 98.88% accuracy.

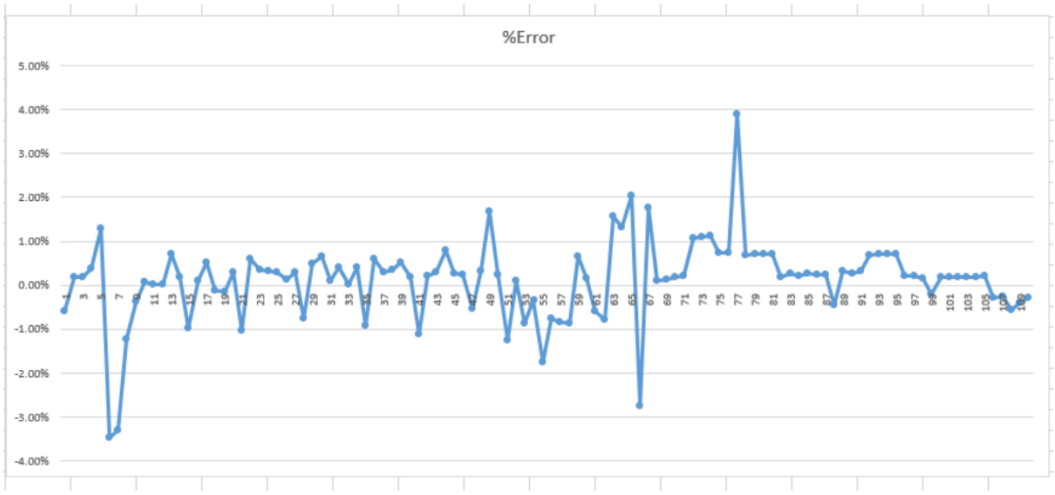


Fig. 14. Error graph for 98.88%

V. CONCLUSION & FUTURE SCOPE

As the number of test cases increases, accuracy also in- creases which enables us to determine the power consumption as accurate as possible. This method is suitable for estimating the power consumption of microcontrollers accurately. The main bottleneck in this method capturing and measuring the current which acts as a trace data. This work is mainly on the simulation part. This work can be implemented to emulation also in the future to create a better model for power estimation.

REFERENCES:

[1] S. Yang, C. Chiu, C. Chang, C. Chen, C. Meng and K. Chen, ”87% Overall High Efficiency and 11µA Ultra-Low Standby Current Derived by Overall Power Management in Laptops With Flexible Voltage Scaling and Dynamic Voltage Scaling Techniques”, in IEEE Transactions on Power Electronics, vol. 31, no. 4, pp. 3118-3127, April 2016, doi: 10.1109/TPEL.2015.2450746.

[2] C. Brandolese, W. Fornaciari, F. Salice and D. Sciuto, “An Instruction- level Functionality-based Energy Estimation Model for 32-bits Mi- croprocessors”, DAC ’00: Proceedings of the 37th Annual Design Automation Conference, June 2000, Pages 346–351.

[3] V. Tiwari and M.T.-C. Lee, “Power analysis of a 32-bit Embedded Microcontroller”, VLSI Design Journal, 1996, pp. 1-6.

[4] T.Sato, M.Nagamatsu and H.Tago, “Power and performance simulator: ESP and its application for 100MIPS/W class RISC design”, Proceedings of 1994 IEEE Symposium on Low Power Electronic, San Diego, CA, October 1994, pp. 46-47.

[5] S. Jain, L. Lin and M. Alioto, ”Processor Energy–Performance Range Extension Beyond Voltage Scaling via Drop-In Methodologies,” in IEEE Journal of Solid-State Circuits, vol. 55, no. 10, pp. 2670-2679, Oct. 2020, doi: 10.1109/JSSC.2020.3005778.

[6] C. Zheng and D. Ma, ”A 10-MHz Green-Mode Automatic Reconfig- urable Switching Converter for DVS-Enabled VLSI Systems,” in IEEE Journal of Solid-State Circuits, vol. 46, no. 6, pp. 1464-1477, June 2011, doi: 10.1109/JSSC.2011.2131770.

[7] Y. Lee et al., ”A DVS Embedded Power Management for High Efficiency Integrated SoC in UWB System,” in IEEE Journal of Solid-State Circuits, vol. 45, no. 11, pp. 2227-2238, Nov. 2010, doi: 10.1109/JSSC.2010.2063610.

[8] D. Bull, S. Das, K. Shivshankar, G. Dasika, K. Flautner and D. Blaauw, ”A power-efficient 32bit ARM ISA processor using timing- error detection and correction for transient-error tolerance and adap- tation to PVT variation,” 2010 IEEE International Solid-State CircuitsConference - (ISSCC), San Francisco, CA, USA, 2010, pp. 284-285, doi: 10.1109/ISSCC.2010.5433919.

[9] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[10] Aradhya H.V.R and Goudar S, “Development and Analysis of Pa- rameters to Evaluate Design Partitioning of SoC,” Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA-2020, 2020, pp. 416–421.

[11] Aradhya H.V.R, Aktab M.L.U, Saberi F, “Development of a Ran- dom Test Generator for Multi-Core Processor Design Verification,” Proceedings of the lrd International Conference on Electronics and Communication and Aerospace Technology, ICECA-2019, 2019, pp. 1200–1204.