International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 3, Issue 11, November 2014

F.P.G.A Implementation of Way Tagged Cache into Chip Multiprocessors

Minu Senan[1] **Babu P Kuriakose**[2]

[1]MTech VLSI & Embedded Systems, MACE [2]Associate Professor, ECE, MACE

***Abstract*— Energy consumption has become a critical issue in many high performance microprocessors. Chip multiprocessors, CMP is only way to build high performance microprocessors. Most of the power dissipated in microprocessor is contributed by on chip cache memories. This paper includes a CMP architecture in which way tagging is applied to cache memories for reducing energy consumption. A two core processor architecture with a shared L3 cache has been selected. For each core, the way information of L2 cache is maintained in the L1 cache and that of L3 cache in L2 cache. The IO power report after implementation shows that about 50% of power reduction can be achieved by including way tagging into CMP architecture compared to conventional CMP.**

***Index Terms*—Chip Multiprocessor, Way tagging, Write through policy, LRU replacement policy.**

I. INTRODUCTION

Multi level cache hierarchies have been employed in each core of ***Chip multiprocessors, CMPs***[2]. For better tolerance to soft errors, cache ***write through policies*** is widely adopted in many high performance microprocessors. Write through policy maintains data consistency at all levels of hierarchy. Under write through policy, whenever a data is updated at the L1 cache, the L2 cache is also updated with the same data resulting in an increase in write access to L2 cache. This consequently increases energy consumption. Thus a significant fraction of the total processor power budget is contributed by multilevel on chip cache hierarchies. Different techniques for reducing the energy consumption of cache hierarchies have been widely studied in the past. But these schemes are not applicable to CMPs due to filtering effects. In our work, a CMP architecture with two cores is considered. The last cache in a CMP architecture is always a shared cache which is shared by all processor cores. Each core employs a two level ***way tagged cache architecture***[1]. Way tagging helps to reduce the energy consumption of each core thereby reducing the total chip power consumption.

For a two level cache hierarchy, all data residing in the L1 cache will have its identical copies in the L2 cache. For all location in the L2 cache, a tag is attached which is send to L1 cache when a data is written to L1 cache. So, for all data in L1 cache, its exact location at the L2 cache is known. Hence instead of accessing all the locations, only the desired way is activated thus reducing the energy consumption significantly. Also, L3 cache which is shared by two processor cores is partitioned, which results in further energy reduction.

The rest of this paper is organized as follows. Section II describes the overview of proposed CMP architecture and also explains the internals of way tagged cache architecture. Section III discusses the simulation platform for the proposed architecture. Section IV presents power analysis and compares the way tagged cache architecture with the existing two level cache hierarchy. Finally, section V presents the conclusion.

II. OVERVIEW

A two core processor architecture[7] with a shared L3 cache is considered for our work. Each processor core includes a two level way tagged cache architecture. The L3 cache is the largest and is the last level cache. The overall architecture is depicted in the Fig.1 below.

Only one processor core can access the shared L3 cache at a moment. At that time, the other processor core will be in the waiting state. A control signal is provided which controls the access of each core to the shared L3 cache.

The two level cache hierarchy in each processor core includes way tagging for reducing the on chip energy consumption. Thus it helps in reducing the overall energy consumed by the whole processor.

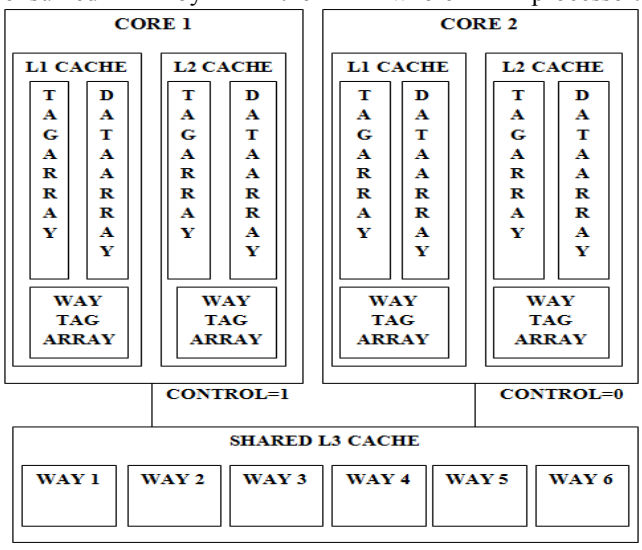

Fig: 1 System overview

ISSN: 2278 – 909X All Rights Reserved © 2014 IJARECE

A. TWO LEVEL CACHE ARCHITECTURE

Consider a two level cache hierarchy, in which L1 cache is 2 way set associative and L2 cache is 4 way set associative[1]. The architecture of way tagged cache includes tag array, data array, way tag array, way decoder and so on.

Under cache write through policy, all data residing in L1 cache will have its identical copies in the L2 cache. The location information of these data copies residing at the L2 cache is available at the L1 cache from the way tag array. The access to L2 cache is controlled by way enable signals such as ***way1***, ***way2***, ***way3*** and ***way4*** generated by way decoder.

Considering the read access to cache memory. For a read hit in L1 cache, there is no need to access L2 cache, since data is available from L1 cache itself. In this case, all ways in the L2 cache is deactivated by way enable signals to reduce the energy consumption.

For a read miss in L1 cache, the corresponding location information is not available at the way tag array. In this case all ways in the L2 cache is activated. If read hit occurs at the L2 cache, the corresponding data is outputted from L2 cache. The same data along with its tag address and way tag gets replaced at the data memory, tag memory and way tag array of L1 cache. A ***least recently used replacement policy***[5] is utilized here. That is, the location which has not been recently accessed is selected for replacement.

Considering the write access to the L1 cache. Since write through policy is utilized here, when a data is updated at the L1 cache, the L2 cache also needs to be updated with the same data. For a write hit in L1 cache, the way information of that location at L2 cache is available from way tag array. Hence the corresponding way is only enabled and the requested location is updated with the new data.

For a write miss in L1 cache, all ways is activated and data is written to the corresponding location if a hit occurs in L2 cache. The least recently used replacement policy replaces the least accessed location with the same data.

Since for write hit, only one way is enabled, it reduces the energy consumption. ***Xilinx ISE Design Suite 14.2 platform***[8] is used for simulation and synthesis.

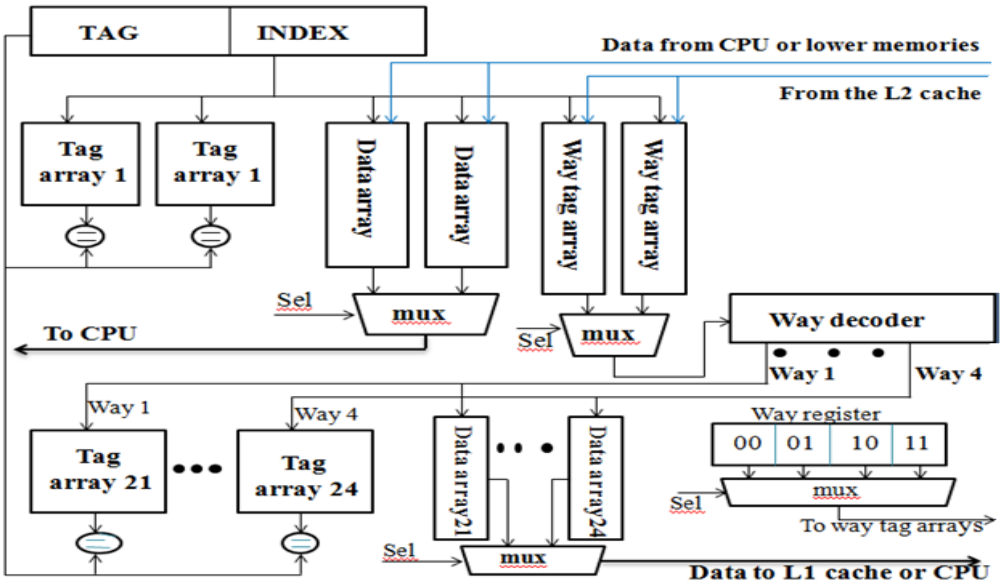

Fig. 2 Way tagged cache architecture.

III. SIMULATION

Xilinx ISE Design Suite 14.2 platform is selected for simulation and synthesis. The programming is done using behaviuoral style of modeling in ***Verilog HDL.***

A. TAG ARRAY

Tag Array[3] includes a decoder and a tag memory. Decoder decodes the address requested by the processor and enables only the corresponding location of tag memory. Tag memory stores the address of data located in main memory.

Address requested by the processor is compared with the address stored in tag memory. The RTL view of the tag memory modeled using verilog HDL is shown below.

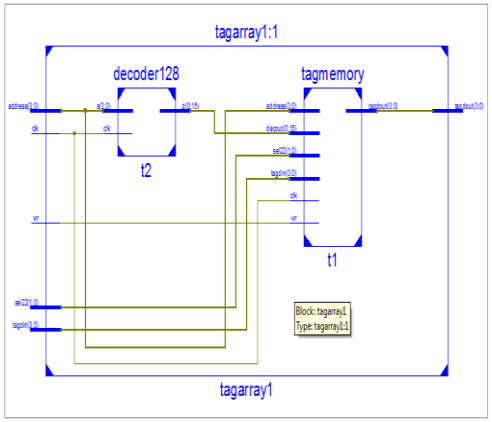

Fig. 3 Tag array

A read/write control signal “wr” is also there. When wr=1; it performs write operation and when wr=0; it performs read operation.

B. TAG COMPARATOR

Tag comparator[3] compares the address requested by microprocessor with the address of data stored in tag memory. If a hit occurs, tag comparator generates enable signal for activating the corresponding data array. The RTL view of the cache tag comparator modeled using verilog HDL is shown below.

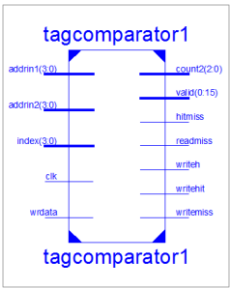

Fig.4 Tag comparator

C. DATA ARRAY

Data array[3] includes a decoder and a data memory. For each address location, data memory stores the corresponding data. Whenever, processor requests a memory location, the location is either updated with new data in case of write access or data is outputted from the location in case of read access. A read/write control signal is also there. When wr=1, data memory performs write operation and when wr=0, it performs read operation. The RTL view of data memory is shown below.

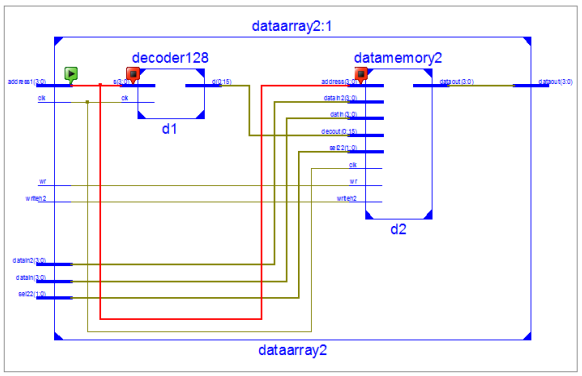

Fig.5 Data array

D. WAY TAG ARRAY

It includes a decoder, an and array and a memory for holding the way information of L2 cache[1]. The way tag array is situated along with L1 cache. Whenever a data is loaded from L2 cache to L1 cache in the case of a cache miss in L1 cache, the way information of L2 cache is also loaded into way tag array. In the case of a write hit in L1 cache, the way tag is outputted from way tag array. It is then provided to way decoder. The way decoder decodes the way tag and generates enable signals for activating a single way in the L2 cache.

The write/read signal of the way tag array is same as that of data array. Another external control signal, update is also there. The update signal is kept low in case of a read operation to the way tag array. The decoder decodes the address provided by microprocessor. And array generates signals for activating each location of way tag array depending upon read/write miss/hit in the L1 cache. The RTL schematic of the way tag array created is shown below.

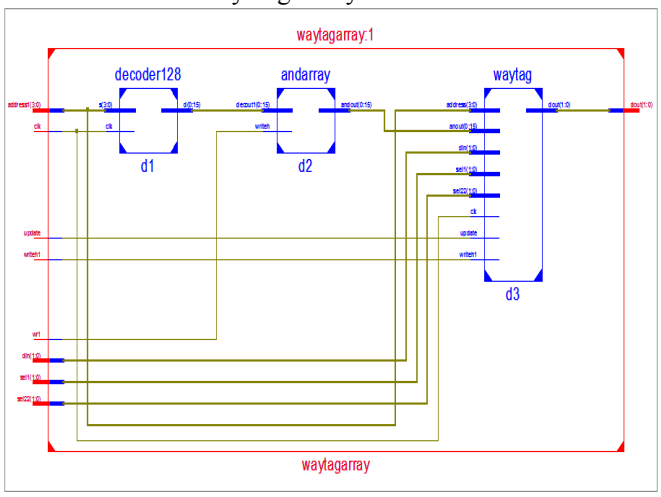

Fig. 6 Way tag array

The size of each line in the way tag can be expressed as

$$linesize = \log_2 N \quad (1)$$

Where '*N*' is the no: of ways in L2 cache.

Here, since L2 cache is 4 way set associative, line size of way tag is 2 bit ($\log_2 4$).

Since, the size of way tag is small, it doesn't incurs any area overhead and hence no performance degradation.

E. WAY DECODER

Way decoder[1] decodes the way tags read from way tag arrays and generate enable signals for activating only desired ways in the L2 cache. For a write hit in the L1 cache, the way decoder acts as a n to N decoder that generates only one way enable signal. The RTL schematic of the proposed way decoder is shown below

The signals write miss and read generated by tag comparator determines the operating mode of way decoder.

The signals bit0 and bit1 is provided by way tag array. For a write miss and a read miss, we need to activate all ways in the L2 cache. Hence the signals way1, way2, way3 and way4 is made high by write miss and read signals.

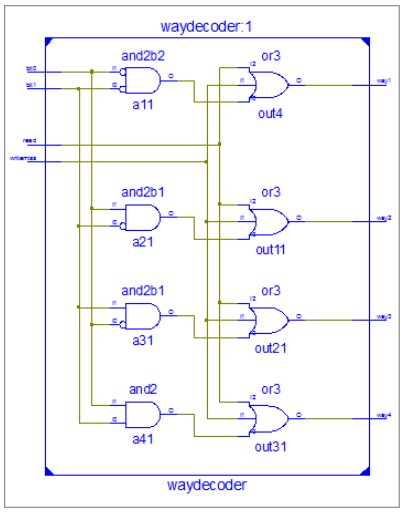

Fig. 7 Way decoder

The RTL view of the designed way tagged cache is given below.

ISSN: 2278 – 909X All Rights Reserved © 2014 IJARECE

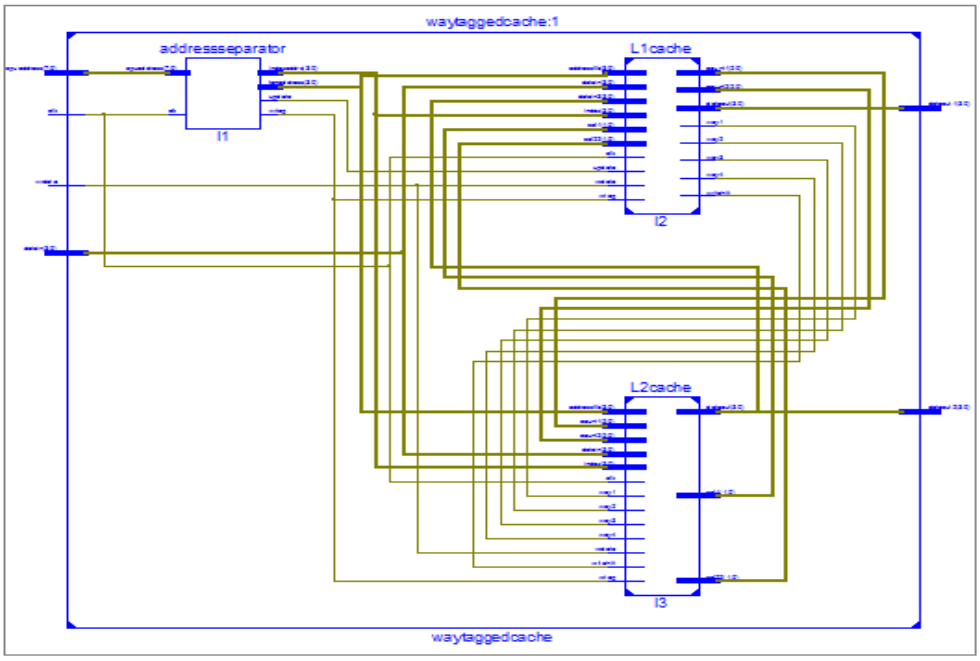

Fig. 8 Way tagged cache

F. REPLACEMENT POLICY

Cache replacement policy is a major design issue of any memory hierarchy. The efficiency of replacement policy influences both hit rate and access latency of a cache system. As the cache associativity becomes higher, the replacement policies become more vital. Various cache replacement policies[4][5] exist such as ***Most Recently Used, random, FIFO, Least Recently Used***, etc. Here I have selected ***Least Recently Used replacement policy*** for evicting a cache line in L1 cache in case of a cache miss.

LRU replacement policy assumes that pages that are used recently will be used again soon. This policy throws out pages that has been unused for longest time. Normally, LRU policy maintains a linked list of pages, most recently used at front and least at rear. The linked list is updated at every memory reference. This can be somewhat slow and the hardware has to update the linked list on every memory reference.

Alternatively, we can keep a counter[6] on each page table entry of cache memory which keeps the track of no: of times a memory location is accessed. The counter gets incremented each time a location is accessed. The page with lowest count value is selected for replacement in case of cache miss in L1 cache.

IV. POWER ANALYSIS

Xilinx ISE Design Suite 14.2[8] platform provides ***XPower Analyzer Tool*** for estimating the total power consumed during each read and write access.

A two level conventional cache architecture without way tagging has also been created for comparing the power consumed by conventional cache architecture without way tagged cache architecture. The system architecture of the conventional cache architecture is shown below.

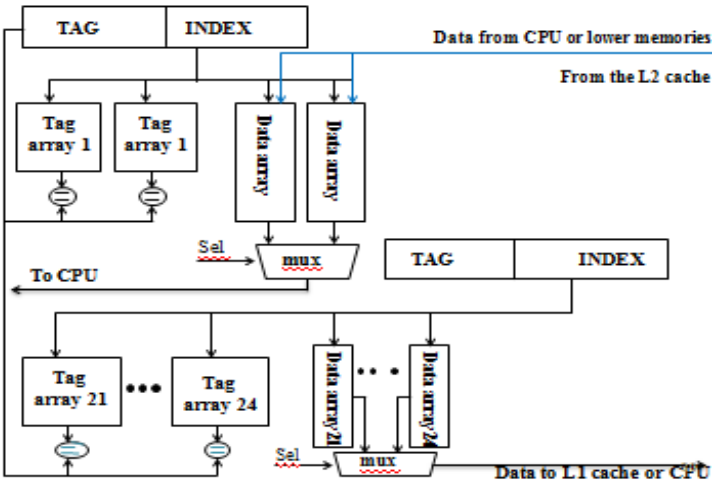

Fig. 9 Conventional cache architecture

The ***static power*** dissipated in both cases is the same. The summary of the static power report is shown in fig: 10.

Dynamic power analysis is done after simulating the proposed architectures.

The ***input/output power report*** of the conventional cache architecture and the proposed way tagged cache

architecture during write hit in L1 cache is illustrated below in fig:11 & fig:12.

Device		On-Chip	Power (W)	Used	Available	Utilization (%)
Family	Spartan3e	Clocks	0.000	1	--	--
Part	xc3s250e	Logic	0.000	2	4896	0
Package	tq144	Signals	0.000	9	--	--
Temp Grade	Commercial	IOs	0.000	10	108	9
Process	Typical	Leakage	0.052			
Speed Grade	-4	Total	0.052			

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.015	0.000	0.015
Vccaux	2.500	0.012	0.000	0.012
Vcco25	2.500	0.002	0.000	0.002

	Total	Dynamic	Quiescent
Supply Power (W)	0.052	0.000	0.052

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	37.5	83.0	27.0

Characterization	
PRODUCTION	v1.2,06-23-09

Fig: 10 Static power report

Name	Power (W)	I/O Standard	Signal Rate	% High	Clock (MHz)	Clock Name	Input Pins	Output Pins	Bidir Pins
IOs									
wrtag	0.00000	LVCMOS25	0.0	0.0	Async	Async	1	0	0
wrdata	0.00000	LVCMOS25	0.0	100.0	Async	Async	1	0	0
clk	0.00002	LVCMOS25	20.0	50.0	Async	Async	1	0	0
addresstlb (4)	0.00000	LVCMOS25	0.0	25.0	Async	Async	4	0	0
datain (4)	0.00000	LVCMOS25	0.0	50.0	Async	Async	4	0	0
dataout1 (4)	0.00000	LVCMOS25_12_SLOW	0.0	0.0	0.0	clk_BUFGP	0	4	0
dataout2 (4)	0.00000	LVCMOS25_12_SLOW	0.0	0.0	0.0	clk_BUFGP	0	4	0
index (4)	0.00000	LVCMOS25	0.0	0.0	Async	Async	4	0	0
Total	0.00002					Count	15	8	0

Fig: 11 Power report-conventional cache

Name	Power (W)	I/O Standard	Signal Rate	% High	Clock (MHz)	Clock Name	Input Pins	Output Pins	Bidir Pins
IOs									
wrdata	0.00000	LVCMOS25	0.0	100.0	Async	Async	1	0	0
clk	0.00001	LVCMOS25	10.0	50.0	Async	Async	1	0	0
dataout1 (4)	0.00000	LVCMOS25_12_SLOW	0.0	0.0	Async	Async	0	4	0
dataout2 (4)	0.00000	LVCMOS25_12_SLOW	0.0	0.0	Async	Async	0	4	0
Total	0.00001					Count	2	8	0

Fig: 12 Power report-way tagged cache

The IO power consumed by the two level conventional cache architecture is 0.00002 while that in the way tagged cache architecture is reduced to 0.00001.

T.hus, by analyzing the report, it is clear that a significant amount of power reduction can be achieved by incorporating the way information of L2 cache into the L1 cache. Thus waytagging helps to reduce the total power dissipated in cache memories with minimal area overhead and no performance degradation. Also, applying this way tagged cache architecture to CMP architecture helps to save the energy consumed by high performance microprocessors.

V. CONCLUSION

The proposed architecture is implemented on ***Xilinx Spartan 3E F.P.G.A -xc3s 250e-4 tq144*** and analyzed the results for various read operation. The existing power reduction techniques can also be incorporated in to this architecture resulting in a further power reduction. Also, for increasing the speed of the microprocessor, we can utilize the cache compression. Cache partitioning can also be applied to the shared L3 cache for attaining further power reduction in CMP architecture.

ISSN: 2278 – 909X All Rights Reserved © 2014 IJARECE

ACKNOWLEDGEMENT

I am thankful to the Electronics and communication engineering department, Mar Athanasius college of Engineering for their great support and also for providing necessary technical help whenever needed which leads to the successful completion of my work. I express sincere gratitude to my beloved guide, **Mr. Babu P Kuriakose** (Associate Professor, MACE) for his valuable guidance and encouragement throughout the whole work. Above all, I would like to thank God Almighty for his blessings which gives me the strength and courage for the successful completion of my work.

REFERENCES

[1] Jianwei Dai and Lei Wang. "An Energy-Efficient L2 cache Architecture Using Way Tag Information Under Write Through Policy", ieee transactions on very large scale integration (v.l.s.i) systems, vol. 21, no.1 jan 2013.
[2]http://academic.udayton.edu/scottschneider/courses/ECT 466/Course Notes/LSN 4 - Instruction-level Parallelism/Chip Multiprocessor Architecture
[3] Yogesh S Watile, A.S Khobragde "F.P.G.A Implementation of cache memory", International Journal of Engineering Research and Applications, Vol 3, Issue 3,May-June 2013, pp.283-286.
[4] http://www.inf.ed.ac.uk
[5]http://www.dauniv.ac.in/downloads/CArch_PPTs/Comp ArchCh09L04RelplacementPolicies.pdf
[6]http://www.ece.ncsu.edu/arpers/Papers/iccd05-counterrepl.pdf
[7] http://www.cs.cmu.edu/
[8] http://www.xilinx.com/

Minu Senan received her B Tech Degree in Electronics and Communication Engineering from Younus College of Engineering & Technology, Kollam in the year 2012 and currently pursuing M Tech in VLSI & Embedded Systems from Mar Athanasius College of Engineering, Kothamangalam.

Babu P Kuriakose presently working as Associate Professor at Mar Athanasius College of Engineering, Kothamangalam.