International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-5, March 2019

Multi-core Micro-controller Architecture with ZLPIC for High Performance Embedded Applications

Kulkarni Rashmi Manik, S Arulselvi, B Karthik

Abstract*: *The main objective is to propose a multi-core architecture for micro-controller implementation, considering availability of various microprocessor cores and advancement in fabrication technology. As it is possible to fabricate 2 million gates per square millimeter at advanced 40nm CMOS fabrication technology and availability of embedded FLASH technology on same process, the innovative multi-core micro-controller can be developed for high performance, low power features. ZLPIC (Zero Latency Programmable Interrupt Controller) module is added along with multi-core to enhance performance. Often embedded micro-controllers need variety of interrupt handling. Zero interrupt latency mechanism is very much required in embedded applications. As long as the interrupt actions are limited and predefined, it is possible to give zero latency response for handling interrupt. A simple device with limited programming model is proposed for achieving zero latency. The logic can be easily integrated with multi-core micro-controller architecture.

Index Terms*: *ARM, CMM(Chip Multi-core Micro-controller), CMOS, GPIO, IPDE, ISR, Interrupt Latency, NVM, SRAM and ZLPIC.

I. INTRODUCTION

With advancement in VLSI fabrication technology, today it is possible to fabricate device having more than a billion transistors. The microprocessors suitable for high end servers are fabricated at state-of-art 28nm or 16nm/7nm Fin-FET process. The micro-controller need to have on-chip embedded FLASH memory for code and data, and the fabrication technology available for embedded FASLH memories is at 40nm and 55nm. The suitable fabrication technologies for embedded micro-controllers include TSMC 40nm, UMC 55nm and Global Foundry 55nm technologies. The 40nm process promises the gate density of approximately 1900 K gates per square mm for implementing the design. The microprocessor cores take 0.02 to 0.03 sq mm area on 40nm technology. The embedded non-volatile memory can be integrated with microprocessor core ranging from16KB to 128 KB. The area required for 16KB NVRAM on 40nm fabrication technology is around 1 sq. mm.

Manuscript published on 30 March 2019.
*****Correspondence Author(s)
Kulkarni Rashmi Manik, Research Scholar/ECE, Bharath institute of higher Education and Research, Chennai, India.
S Arulselvi, Associate Professor, Department of ECE, Bharath institute of higher Education and Research, Chennai, India.
B Karthik, Associate Professor , Department of ECE, Bharath institute of higher Education and Research, Chennai, India.

The SRAM of 8KB is possible in 1 sq.mm at this fabrication technology. Choosing Microprocessor Core is an important decision in developing CMM SoCs. The microprocessor cores are available in wide variety for implementation of SoCs. Nature of end application needs to be analyzed for choosing appropriate processor core. Different varieties of ARM cores are commercially available. The range starts from simple 32-bit core, implementing simple 2-stage pipe-lined architecture to super-scalar 8-stage architecture. For Internet-Of-Things application it is required to choose a core having lowest power consumption. Whereas high end embedded application may need high performance core. Generally embedded micro-controllers are having single processor core integrated with FLASH memory, SRAM memory and various peripherals (e.g. Timers, UARTs, GPIOs, I2C, I2S, USB, ADC, DAC, PWM, RTC, and Watch Dog Timers). The single core puts limit on performance of micro-controller in handling various interfaces simultaneously and in executing hard deadline intensive application. Most of the micro-controller share the GPIOs. They are not available at one time, which also limits the micro-controller usage. Here multi-microcontroller with multiple GPIOs are preferred as system architecture. Integrating ZLPIC module enhances the performance of the SoC. ZLPIC is available as simple peripheral device for embedded systems or as an IP core for CMM SoCs. Embedded systems which need quick actions with almost no or very less processing can be taken care by this device. This we can resemble to quick reflexes of our body where action is taken swiftly. For example, our eye lids protect our eye from the dust particles. They won't send signal to brain. Same way, ZLPIC takes pre-programmed action without informing processor core. This device interfaced along with single processor to form an embedded system which responds multiple interrupts simultaneously with no time delay. On the other hand, ZLPIC integrated with CMM forms a SoC with improved real time performance. For example, robot, automobile systems, embedded systems in airborne applications, autonomous vehicle control systems etc. Literature Survey is given below:B. Mithu and G Stephan, presents research about power aware micro-controller design with virtual peripherals. [1] G. Vivek and co-authors proposed novel architecture for removing harmonics from multilevel inverter output. [2] A Mayer and F Hellwig presented the paper elaborating the performance improvements of different SoC architecture and implementation options [3]

Sakata and T. Hirotsu, presented the paper, that describes cost effective and dependable microcontroller architecture. [4] K Kim and J Park, introduced method for design and verification of 16bit microcontroller which is applicable for 8 bit micro-controllers as well. [5] K D Kramer and co-authors proposed details about tools and algorithms to access microcontroller architecture for specific applications. [6] J Saalmueller and J Wuertz, presented a paper which provides the power PC 440 based solution for fail-safe applications. [7] M.A.Perez-Quinones and J L Cruz-Rivera, presented the paper about design and implementation of prototype IDE for microcontroller MC68HC11.[8] A J Arellano and C M Lopez, presented the paper, which implements 16 bit integer microcontroller on FPGA for research purpose. [9] H Yue-li and co-authors presented paper about high performance 8-bit 8051 micro-controller with power efficiency. [10] Eleven to twenty nine references are various books and websites which author studied thoroughly.

II. PROPOSED METHODOLOGY

The step-by-step process for the design of high performance micro-controller is worked out.

- ✓ The initial step worked out is a multi-core architecture considering application performance requirement.
- ✓ To augment the expected performance, available processor core architectures are analyzed.
- ✓ The size and organization of Non-Volatile memory and SRAM memory for each processor core is worked out.
- ✓ The integration architecture for all processor core and set of all peripherals for supporting wide range applications is worked out.
- ✓ Next step is analysis of interrupt latency and generation of common output waveform requirements.

A. Analysis of Multi-core architecture

For achieving higher performance, very aggressive, deeply pipe-lined, super-scalar architectures supporting out-of-order execution were developed. These aggressive features even though operate at very high clock frequency consumes very high power and not at all suitable for embedded system. The simple minimum stage pipe-lined architectures consume less power and occupy small area when implemented on 40 nano metre fabrication technology. From embedded application side, many of the embedded applications are inherently muti-threaded. Exploiting thread level parallelism of application using multi-core architecture is far efficient, scalable and simpler than using aggressive single core. Moreover in embedded applications, different interfaces are used for sensors, actuators which need thread level parallelism. Always most of the micro-controllers today have a large set of interfaces sharing available IO pins. Implementing multi-core micro-controller architecture is a solution to get more performance and to handle all peripherals without multiplexing IO pins. The multiple- real-time tasks can be executed with multi-core micro-controller. The issues to be addressed in multi-core micro-controller architecture are as follows; i) processor-core and on-chip memory architecture, ii) Access of peripherals, iii) Configuring Code memory and iv)Power On Program flow. Additionally, feature of zero latency response to interrupts is added in CMM architecture. Many times, interrupt responses are typical and their Interrupt Sub-Routines (ISRs) are small in size. These interrupts do not need attention of CPU and can be handled by ZLPIC.

B. Analysis of Processor cores and On-chip Memory

The multiple cores execute instructions simultaneously from independent threads, so architecture of on-chip memory worked out to support parallel access by all cores. The choice is either to have a single monolithic code memory for all cores or to have smaller separate memory for each core. Having single memory slows down the application execution. It is necessary to have a separate code memory (Non Volatile Memory) for each core and a separate SRAM memory (scratch pad memory) for each core. The block diagrams explaining multi-core micro-controller are shown in figure 1 and figure 2. The figure 1 shows Processor Tile (PT)) and figure 2 shows the Scalable Multi-core Architecture. The number of microprocessor cores can be 4 or 8 or more.

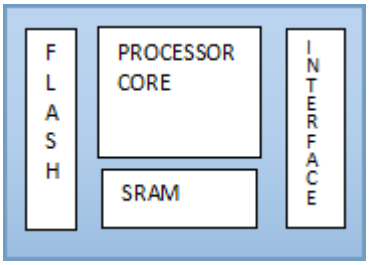

Figure.1 Processor Tile: PT

1. Processor Tile

The processor tile has one processor core interfaced with FLASH memory and SRAM memory. The processor executes the code from FLASH memory which is accessed in a single cycle. The SRAM is for storing run-time variable and it is also accessed in single cycle. The processor core has separate dedicated interfaces for FLASH and SRAM memory. Processor core has all interface signals available from “Interface” block for interfacing on-chip peripherals, interrupts signals from all peripherals and signals for interfacing Dual Port Shared Memories between cores. In figure 2, complete micro-controller architecture is shown. The Dual Port Memories are SRAMs having two ports and used for data exchange between two processor tiles. If we are integrating four processor tiles, then over all six Dual Port SRAM memories are required for data sharing between all processor tiles. If we are connecting eight processor tiles then 28 Dual Port SRAMs are required for data sharing between processor tiles. If we are scaling the architecture for 16 Processor Tiles or more then hierarchical grouping of Processor Tiles is required along with Dual Port SRAM to reduce number of Dual Port SRAM memories required for data sharing. For architecture with 16 processor tiles, we can put 4 processor tiles as one group (Super Tile) and create 6 Dual Port SRAM memories for within group data sharing. So we have architecture of 4 Super Tiles interconnected with additional DP SRAMS.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

We needs six Dual Port SRAM for each super tile and we need in addition only six Dual Ports SRAM memories for data sharing across the four Super Tiles. The architecture can be scaled for 64 Processor Tiles or 256 processor tiles.

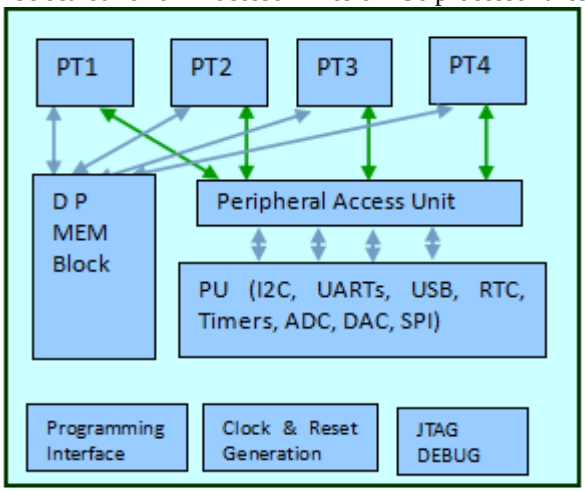

Figure 2. Scalable Multi-Core Architecture

2. Peripheral Access Unit

For higher performance, the access of the peripherals by any of the processor is completed with minimum cycles. Various options exist for interfacing peripherals with the processor tiles. For better performance and scalability in design, special block called Peripheral Access Unit (PAU) is proposed. All the peripherals are proposed to be in single unit called Peripherals Unit and each peripheral will have its own bus signals available for interfacing. PAU can accept the access request from all processor tiles separately and is having separate interface to each peripheral on other side. The logic design of PAU implements switch matrix such that as long as requests from processor tile are for different peripherals the access is granted immediately. Also PAU has provision to assign interrupts generated from peripheral to any one or more processor tile. So if PT1 is requesting for UART access and PT2 is requesting for ADC access, PAU allows simultaneous access of PT1 to UART and PT2 to ADC independently. The PAU also allows the interrupt signal to be generated from one processor tile to other tile. This will allow the intimation from one PT to other PT for exchanging real-time data sets. For up to 16 processor tiles we can have flat interface of all processor cores to PAUs. If processor tiles are 32 or more, then logical grouping of the processor tiles for accessing the peripherals is required or restricting peripheral access to only limited maximum 16 processor tiles is proposed. The other processor tiles have to be used for computation or running application part that does not need peripheral access. The Network-On-Chip is required for architectures more than 32 tiles.

3. Configuring Code Memory

As available in development environment of any other micro-controller, proposed CMM architecture comes with an Integrated Programmer Development Environment (IPDE), which has provision to compile user's application among the processor tiles in the CMM. The users can have selection on number of processor tiles for his application for getting required performance. The IPDE generates executable files and the on chip FLASH memories needs to be programmed using IPDE utilities. The IPDE profiles the user's application and suggests the required clock frequency of operation to be selected for each processor tile. The unused processor tiles enter in low power mode during execution of application.

4. Power-on Initialization Program

Once CMM starts after reset, all processor tiles execute the code configured into local FLASH. The processor tiles access required peripherals for reading real-time data and also access the required peripherals for sending the real-time data.

C. Analysis of ZLPIC

There are devices which process interrupts fast. However zero latency is ideal condition and which is not possible practically. All present processors and micro-controllers handle interrupt with priority. The highest priority interrupt also need processor to save all its registers and pointers on the stack and serve the interrupt. This certainly needs few clock cycles. Another important fact is highest priority interrupt is always only one and there cannot be multiple interrupts. Programmer forcibly needs to resolve the priorities of interrupt even though all are at highest priority. At a time only one interrupt is processed. Otherwise we need multiprocessor environment with advanced programmable interrupt controllers (APIC). In our real world, there are many applications where we need to process multiple interrupts simultaneously, all with highest priority. One option is to go for multiprocessor environment. However, it needs complex software and hardware design. A peripheral device called "Zero latency programmable interrupt controller (ZLPIC)" may be an optimum solution for the embedded systems which need multiple interrupts to be processed all with highest priority (i.e. With zero latency). On the top of that, if multiple ZLPIC cores are integrated with CMM architecture, result is super fast SoC for hard real tasking environment!

III. IMPLEMENTATION OF MICROCONTROLLER

Here two implementations are experimented. One multi-core architecture integrated with peripherals for generating control signals and interfaces. Other architecture integrates additional ZLPIC along with multi-core and peripherals. The applications requiring fixed output signaling shows performance improvement on ZLPIC based architecture.

A. Multi-Core Implementation

The architecture is perfectly scalable in respect of number of processor tiles; FLASH Memory size, SRAM size, Dual Port Memory Size, peripherals to be interfaced. This allows the implementation of a family of micro-controller having varying number of processor tiles and interfaces, suitable for variety of embedded applications. Also the basic processor core can be chosen among available cores from ARM M type or ARM R type.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The basic member of Multi-core micro-controller family can have two processor tiles interfaced with ADC, DAC, GPIO (16 bit), UART (2), SPI (1), I2C(4), RTC, Timers (4) suitable for low end applications. The FLASH size can be 2KB to 64KB per tile. The microprocessor tile needs approximately 1 sq. mm area when implemented with ARM M0 core with 4KB FLASH and 1Kb SRAM. We can interconnect 4 to 64 such processor tiles taking around 4 sq mm to 64 sq.mm area on silicon, when implemented on 40nm low power fabrication technology. To reduce power dissipation, techniques like software controlled selective shutdown of cores, peripherals and selecting appropriate operating frequencies are proposed.

B. ZLPIC Implementation

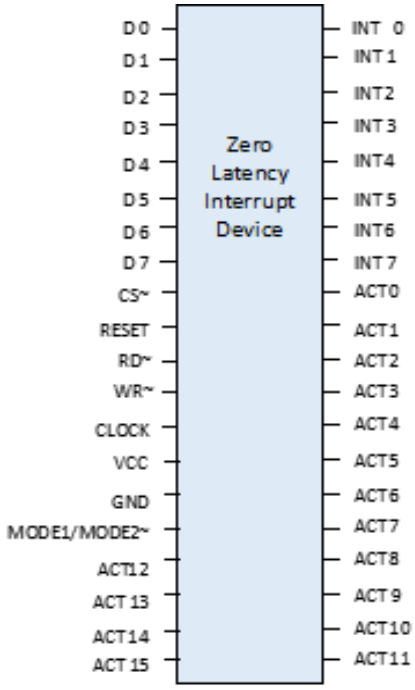

Figure 3. Block diagram of ZLPIC

The ZLPIC module implementation is described here in detail. At the lowest, this module can be interfaced with 8 bit micro-controller core 8051 and higher side with complex multi-core micro-controllers. Forty signals of the ZLPIC module are as shown in figure 3. It is possible to scale-up the output number of signals for complex applications.

The interface signal details are as follows: 1. D0-D7 data bus from controller 2. Clock, reset, RD~, WR~, M0/M1~, CS~ from controller 3. VCC and GND 4. INT0-INT7 hardware interrupts from external device 5. Act0-Act15 action pins to external device. Action pins can be set on or off according to the detection of corresponding mapped interrupt. On and off action may set external device parts/relays/sensors on or off.

1. Modes of ZLPIC

ZLPIC operates in two modes, mode1 and mode 2, as described below;

Mode1: In mode 1, ZLPIC provides eight hardware zero latency interrupts. All interrupts can be mapped to sixteen action pins (act0-act16). One interrupt can be mapped to multiple action pins. All the words described here have to be programmed in sequence to the ZLPIC from CPU.

Mode 1 initialization words with (Mode 1/Mode2) pin tied to VCC:-

Word 1 (d7-d0):- (d7-d5) Interrupt number 3bits, (d4-d1) Action pin number 4bits, d0 action - on/off.

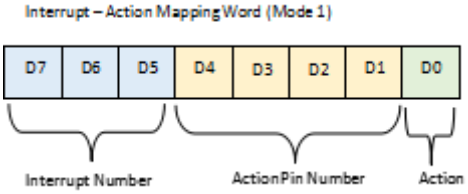

Figure 4. Interrupt Action Mapping Register

Word 2 (d7-d0):- (d7-d5) Action Type 3bits, (d4-d0) No of times action repeat 5bits- (00000b for repeat forever)

Action Type (d7-d5)

1. On (000b) 2. Off (001b) 3. Toggle (010b) 4.On-Time Gap1-Off-Time Gap2 (011b) 5. Off-Time Gap1-On-Time Gap2 (100b) 6. Toggle-Time Gap1-Toggle-Time Gap2 (101b)

Word 3: -Time Gap1 **Word 4**:- Time Gap2

Word 1 to Word 4 can be repeated for every interrupt during initialization of the device.

Mode 2: In mode2, action pins are available as a two eight bit ports. Each port can be mapped to int0 or int 1. After arrival of interrupt, data from the mapped port will be output according to initialization. Data can be a sequence of action for a robotic arm, CNC machine or actuators in vehicle guidance system. Two way data can be output on the port i.e. Sequential or random pattern. In sequential again it can be incremented or decremented according to the initialization of the ZLPIC. Mode 2 initialization words are described here sequentially.

Mode 2 initialization words (Mode 1/Mode2 pin tied to ground)

Word1:- Int No-D7, map to Port No.-D6, Int No-D5, map to Port No.-D4, Not Used-D3, Act1-D2,Act2-D1,Act3-D0

In mode two, only interrupt 0 and interrupt 1 cap be mapped to port. Rest other interrupts are mapped permanently as i.e. Int2 to act 5, int 3 to act 6, int 4 to act 7. Here act5, act 6, act 7 pins are int 5, int 6 and int 7 pins in mode 1. They act as an action pins in mode two for interrupt 2,3 and 4 respectively. Here Act1-D2,Act2-D1,Act3-D0 bits are actions specified for interrupt 2,3 and 4.

Word 2:- D7 - 0:-random data, 1:-sequential data, D6 and D5:- Data Variation Type, Step Unit- 5bits D4 to D0

For sequential data:-

Data Variation Types: -

Type 1(00):- Increment in steps and return to initial value after overflow

Type 2(01):-Decrements in steps and return to initial value after overflow

Type 3(10):-Increment by one and return to initial value by decrement

Type 4(11):-Decrement by one and return to initial value by increment

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Word 3:- No of Steps, Word **4:-** Start Value, **Word 5:-** Maximum Value
Word 6:- Minimum Value, **Word 7:-** Time Gap1 (Increment). **Word 8:** -Time Gap2 (Decrement), **Word 9:-** No. Of times repeat (00 values for repeat forever)
Thus sequential data can produce any waveform or regular data pattern.

For random data (D7=0 in words 2, mode 2):-
Word 2:- D7=0, D6=port number (0 or 1), D5 to D0 for number of bytes in random sequence, **Word 3:-** time gap between each step, **Word 4:-** Number of times repeat (00h value for repeat forever), Byte sequence (each port maximum 64 bytes) of both ports will be stored in internal 128 bytes of RAM. Word 1 is common for sequential and random data pattern.

2. Internal Architecture Diagram of ZLPIC

ZLPIC has a set of control registers for programming interrupt response as shown in figure 5. The Data buffers are for storing the response data to be sent on programmed output signals. The size of data buffer is configurable as single byte to 16 bytes per output signal.

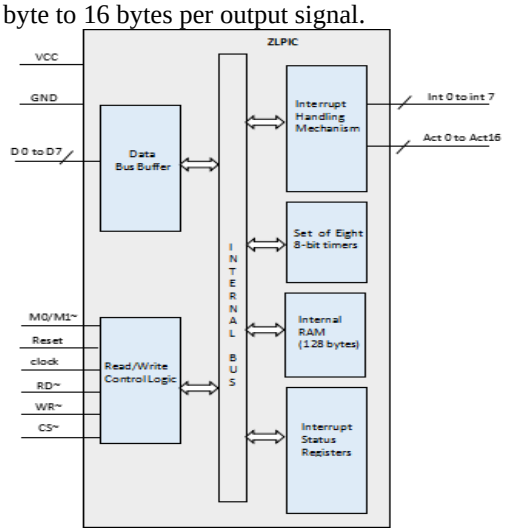

Figure 5. Internal Architecture of ZLPIC

C. CMM with ZLPIC Module

The ZLPIC module is integrated with CMM as shown in figure 6. The processor core can access the control registers in the ZLPIC and set the required response actions on output signals of ZLPIC. The external interrupts as well as interrupt from internal peripherals can be selected as input triggering for ZLPIC.

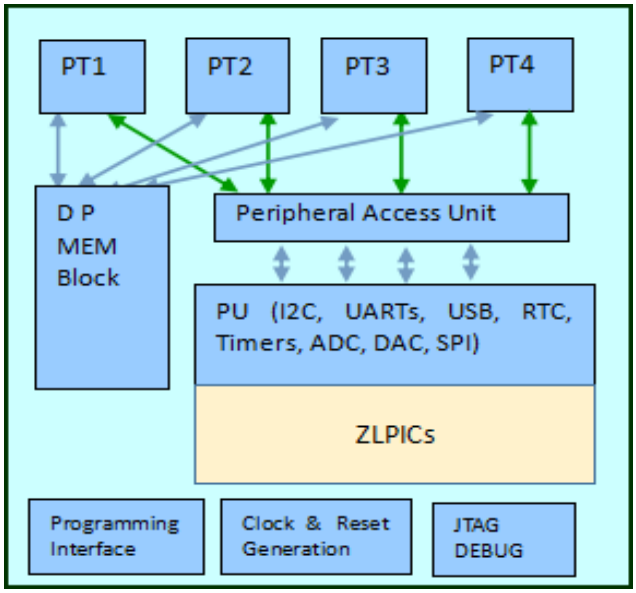

Figure 6. CMM with ZLPIC

D. ZLPIC Initialization Routine (Pseudo Code)

```
//zlpic initialization routine
#include<stdio.h>
.....
......
Void main( )
{
      Initarray[ ] = {01h, 60h, 50h, 50h, 23h, 60h, 50h, 50h}
      /*data initializes int0 and int1,
      In mode 1,
      Maps to action pin 0 and 1
      With repeat forever on and off
      With time gap 50h*/
      InitZLPIC(InitArray); //initialize ZLPIC and wait
      While(1);
}
int InitZLPIC(int *dataptr)
{
      int dataword;
      If(dataptr==NULL)
      return(0);
      while(dataptr != NULL)
      {
            Dataword=*dataptr;
            asm
                  {
                        MOV A,dataword;
                        OUT 90h;
                        //output to ZLPIC with mode0
                        //ZLPIC interfaced at 90h & 91h
                        //for mode0 and mode1
                  }
          dataptr++;
      }
     Return(1);
     }
```


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

IV. PERFORMANCE ANALYSIS

The performance parameter for micro-controller is peak Dhryston Million Instructions per second (MIPS) achievable from the architecture for selected processor core. The code FLASH is accessible in single clock cycle and hence it can supply one instruction per clock cycle. The SRAM and Dual Port Memories are accessible in one clock cycle for the processor. The FLASH memory access speeds are comparatively at 100 MHz, for size of 16 KB. The peak performance of multi-core micro-controller is estimated and performance depends on selected core and number of tiles integrated. The different multi-core implementations of ARM processor cores are considered for performance estimation. The performance for different ARM Cortex Series of 32-bit cores is bench-marked at various fabrication technologies and performance figures are referred from information at ARM web site. The table 1 indicates the comparative performance. Figure 7 indicates estimated performance for multi-core tiles of selected processor core operating at 100 MHz.

Table 1 Estimated performance

CPU No. Cores	Single Core MIPS	Quad Core MIPS	16 Core MIPS	64 Cores MIPS
ARM Cortex M0+	95	380	1440	6080
ARM Cortex M1	125	500	2000	8000
ARM Cortex M7	214	856	3424	13696
ARM Cortex R-7	250	1000	4000	16000

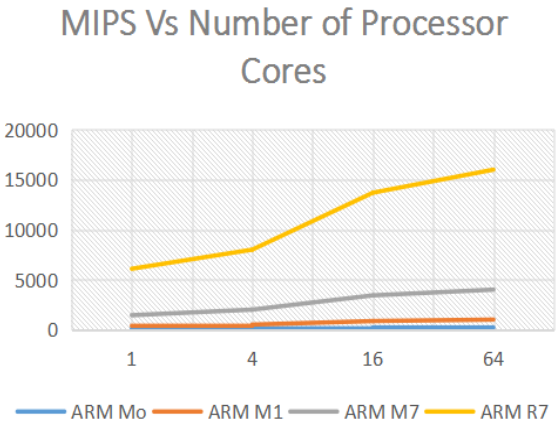

Figure 7. Performance of multi-core

The performance improves for architecture incorporating ZLPIC. The interrupt latency in micro-controller without and with ZLPIC is compared in figure 8 for three types of applications. Three types of real-time applications are control application, multimedia processing and data communication application.

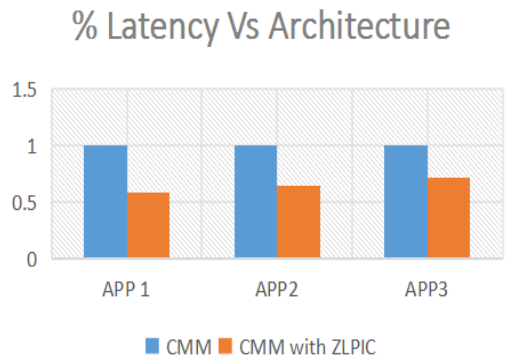

Figure 8. Latency Comparison

V. CONCLUSION

The multi-core architecture for micro-controller is proposed for achieving higher performance for embedded applications like IOT, Automobile, Medical Instrumentation and airborne applications. The fabrication technology integrating on-chip NVRAM provides path for implementation of low power multi-core micro-controller architecture for embedded applications. Additionally, performance of the Multi-core Micro-controller is enhanced with ZLPIC. The design is scalable and we can put more configuration registers to enrich the interrupt action.

REFERENCES

1. B. Mithu and G Stephan, "Low-power Oriented Microcontroller Architecture", *IEEE 2000 International Semiconductor Conference, October 2000, Romania.*
2. G. Vivek and Meenu D Nair, Mukti B, "Synchronized Microcontroller Architecture to eliminate dominant harmonics for quasi square waveform", *2016 IEEE 7th International Symposium on Power Electronics for Distributed Generation Systems, June 2016, Canada.*
3. A Mayer and F Hellwig, "System Performance Optimization Methodology for Infeneons's 32-Bit Automotive Microcontroller Architecture", *2008 Design Automation and Test, March 2008, Germany.*
4. T. Sakata and T. Hirotsu, "Cost Effective Dependable Microcontroller Architecture with Instruction Level Rollback for Soft Errors Recovery", *37th Annual IEEE/IFIP International Conference On Dependable Systems and Networks, June 200, UK*
5. K Kim and J Park, "Study on Microcontroller Design and Verification Methodology", *IEEE 7th Korea-Russia International Symposium on Science and Technology", July 2003, South Korea*
6. K D Kramer, T. Stolze and T Banse, "Benchmark to find the optimal microcontroller architecture", *IEEE, 2009 WRI World Congress on Computer Science and Information Engineering", April 2009, Los Angeles.*
7. J Saalmueller and J Wuertz, "Embedded Controllers for solving complex Industry Applications", *IEEE 2006 SoC Conference, Sep 2006, Taiwan.*
8. M.A.Perez-Quinones and J L Cruz-Rivera, "Integrated Development Environment for a microcontroller systems laboratory", IEEE Conference 29th Annual Frontiers in Education, Nov 1999, USA.
9. A J Arellano and C M Lopez, "Design of an academic microcontroller and its application to Authenticated Encryption", *IEEE 2014 International Conference on Electronics, Communications and Computers, Feb 2014, Mexico.*
10. H Yue-li,C Jia-lin, ran Feng, "Design of High Performance microcontroller", *IEEE CPMT Conference on High Density Microsystem Design and Packaging and Component Failure Analysis, July 2004.*

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

11. Robert Ashby, *My First Five PSoc 3 Designs*, Cypress Semiconductors.(Book)
12. Kai Hwang, *Advanced Computer Architecture*, ed. 2, McGraw-Hill. (Book)
13. David Patterson and John Hennessy, *Computer Architectures: A Quantitative Approach*, ed. 5, Morgan Koufmann, 2007.(Book)
14. Richard Kain, *Advanced Computer Architecture a system design approach*, ed. 1,Prentice Hall, 1996.(Book)
15. Max Dameika, *Software Development for Embedded Multi-core Systems*, ed. 1, Elsevier, 2008.(Book)
16. Michael Barr, *Programming embedded systems*, ed 2,O'*Rielly*,1999.(Book)
17. S.V.Altaf, *Microprocessors and Micro-controllers*, ed. 1, Imperial, 2011.(Book)
18. A.K.Ray and Bhurchundi, *Advanced Microprocessors and Interface*, ed. 3, McGrawHill, 2011.(Book)
19. A.P. Godse and A.O. Mulani, *Embedded Systems*, ed. 1. Technical Publications, 2009,(Book)
20. Douglas Hall, *Microprocessor and Interfacing*, ed. 2, McGraw Hill, 1986.(Book)
21. Raj Kamal, *Embedded Systems*, ed. 2, Tata McGraw Hill, 2011.(Book)
22. Byron Gottfried, *Programming with C*, ed 3, Tata McGraw Hill,2010(Book)
23. www.umc.com
24. www.microchip.com
25.www.arm.com
26. http://en.wikipedia.org/wiki/Intrrupt-latency
27. www.intel.in "Reduce interrupt latency in embedded systems-Intel"
28. www.smxrtos.com/articles/Isr_art/Isr_art.htm, "Minimizing Interrupt Latency - Multitasking Kernel"
29. www.tsmc.com

Kulkarni Rashmi Manik (Terkar Rashmi Anil) has received M. Tech. in Embedded Systems from Jawaharlal Nehru Technological University, Hyderabad, India in 2013 and B.E. in Electronics Engineering from Walchand College of Engineering, Sangli affiliated to Shivaji University, Kolhapur, Maharashtra, India. She is presently working as Assistant Professor in Noble College of Engineering and Technology for Women, Hyderabad. Her research areas of interest are embedded systems, System-on-Chip, ASIC design, Micro-controller architectures and Network-on-Chips.

Dr. S Aruselvi has completed B.E. (ECE) from Periyar Maniammai College of Technology for Women, Vallam (Tamil Nadu) in 1996. Subsequently she completed M.E.(C.C.E) from same college in 2007. She completed her PhD from Bharath University in 2017. Presently she is working as associate professor in Bharath university, Chennai. Her areas of interest are computer networking, Network-on-chips, System-on-Chips, wireless networks etc.