Simulation & design of a VLSI embedded system using Verilog Coding with Modelsim approach in FPGA scenarios for AI appications in automotive sectors

[1]Vaishnavi Patil, [2]Dr. Pavithra G., [3]Dr. T. C. Manjunath*

[1]Final Year (Seventh Sem) ECE Student, Dept. of Electronics & Communication Engg.,
Dayananda Sagar College of Engineering, Bangalore, Karnataka
[2]Associate Professor, Project Guide, Dept. of Electronics & Communication Engg.,
Dayananda Sagar College of Engineering, Bangalore, Karnataka
[3]Professor & Head, Project Guide, Dept. of Electronics & Communication Engg.,
Dayananda Sagar College of Engineering, Bangalore, Karnataka

*Corresponding Author : Dr. Manjunath, Ph.D. (IIT Bombay), Sr. Memb. IEEE, FIETE, FIE,
Email : tcmanju@iitbombay.org

Abstract

In modern systems, versatility, efficiency, and cost-effectiveness are paramount. A system should optimize its resource utilization and offer programmability to dynamically adjust its functionalities as needed. Software-based design provides this flexibility, explaining the growing popularity of software-based design systems. In a world marked by rapid innovations, having tools that enable precise modeling is crucial for reducing design time and obtaining comprehensive data on static and dynamic parameters. Programming languages like VHDL (VHSIC Hardware Description Language) and Verilog serve as robust platforms for design simulation and testing, forming the foundation for ASIC/FPGA-based design. VHDL, initially developed for Very High-Speed Integrated Circuits (VHSIC), excels at describing the behavior & the structures of the electrical electronical systems, particularly digital hardware designs like ASICs, FPGAs, and conventional digital circuits. While Verilog is another popular tool, we chose VHDL due to our greater familiarity and comfort withinit. Field type of Programmable Gated Array [FPGAs] present an alternatives to Programmable Logic based Devices [PLDs] & ASICs. As the name suggests, FPGA's offer a significant advantage in terms of programmability. Unlike their PLD predecessors, FPGAs can typically be reprogrammed multiple times, allowing designer multiple opportunities to modify their circuits. To execute our project, we opted for VHDL as the Hardware Description Language (HDL), used Xilinx ISEs 6.1 for synthesis', & employed ModelSims for VHDL code simulation. These choices were made to leverage VHDL's capabilities, and the selected tools facilitated seamless integration and simulation of our VHDL codes.

Keywords: Microcontroller, FPGA, VHDL, Modelsim.

1. Introduction Remarks

In this section, we provide a concise overview of VHDL programming and the FPGA system design, as outlined in [1]. Additionally, we present the paper's structure within this framework..

2. Architectural Description / Spartan-IIE Array

Figure 1 consists of five primary configurable components which comprises of the spartan kit as mentioned below one by one.

- IOB's (Input-Output Block): These serve as the interfaces between the packages pin & the internal logics.
- CLB's (Configurable Logic Block): These are the functional element used to construct most of the logics.
- Dedicated blocks RAM memory, each with a capacity of 4096 bits.
- Clock DLL's (Delay-Locked Loops): These are responsible for compensating for clock type of distributions delays & managing clocked domains controls.
- A versatileable multi type of level interconnection structures.

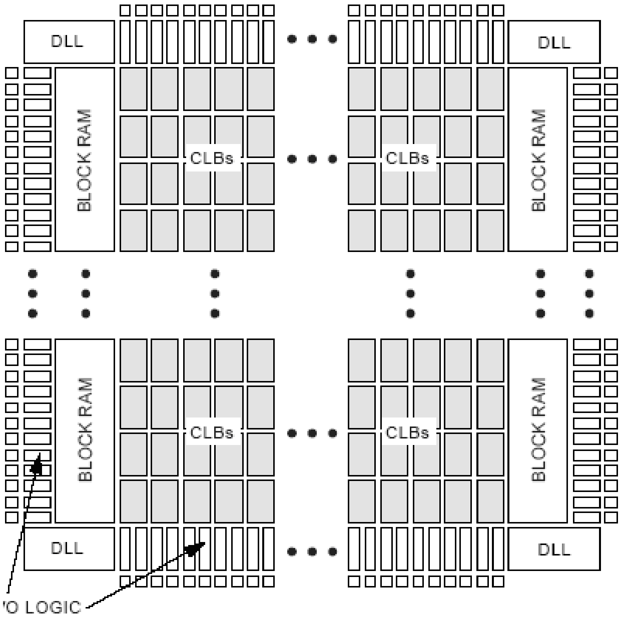

Fig. 1 : Basic SPARTAN 3 Block diagram

As depicted in Fig. 1, the CLB's serve as the core logical structures, offering convenient connectivity to all supports & routings components. Surrounding the logic and memory elements, the IOBs are strategically positioned to facilitating the swift & straightforward signal routings within & beyond the chips. The configuration of the logic elements and interconnect resources is determined by value stored in static memories cell. The values initialize the memory cell upon power-up & could be re-loaded when needed to modify the device's functionality. Subsequent sections will provide an in-depth exploration of each of these components [2].

3. Input - Output Blocks

The Spartan-IIE IOBs, illustrated in Fig. 2, is equipped with inputs & outputs designed to accommodate a wider range of input – output based signal standard. The high speeded interfaces & are capable of the support giving to advanced memory and bus connections. Table 1 provides a comprehensive list of supported standards, including the required reference, o/p & the terminational voltage required for compliance. Within each IOB, three registers serve dual functions, acting w.r.t. one of them as the edge based trigger type of D type flip-flop's or-as the levels based sensitivity type of latch. Additionally, each of the IOB's is equipped with an shared clock based signals [CLK] that operates the 3 register, also, the individual Clock Enabled CE) signals for each of the registers [3].

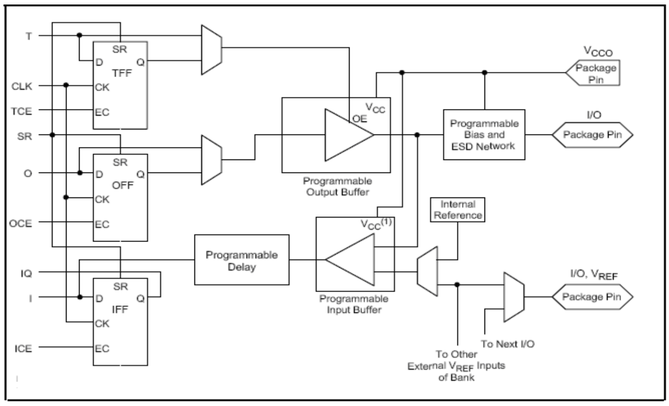

Fig. 2 : Input - output block

Besides the shared CLKs & CE controlled signals, the 3 registers also have a shared Set-Reset [SR] signal. Each's of these SR signals could be individually configure as either a synchronous Sets, an synchronous Resets, an asynchronous Presets or a asynchronous Cleared instructional sets.. Additionally, while not depicted in the figure / diagram, polarity control is a software-controlled feature. This means that both the i/p & the o/p buffers, along with all the IOB controlled signal, can have their polarities independently adjusted [4]. Alongside the shared CLK's & CE controlled signals, then trio of register also utilize the Set--Reset [SR] signal. These signals could be individually tailored for each register, allowing for configuration as an synchronous Sets, synchronous Resets, asynchronous Presets, or asynchronous Clears. While not depicted in the figure, the software's governs another noteworthy feature: polarity based controls. All input & output buffer, as well as the IOB controlled signal, possess in-dependent polarities control settings.

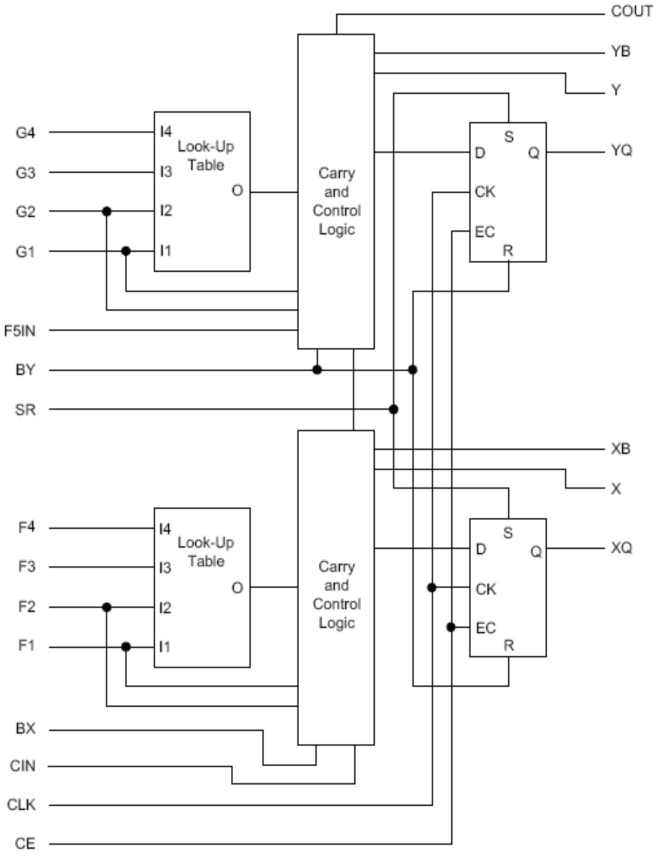

Fig. 3 : Overall circuit diagram

Each user I/O pad comes equipped with optional pulled up & pulled down resistor, as well as an optionally weaker keeping circuits. Before configuration, all outputs that are not part of the configuration process are being forced in to a higher impedanced states. During this phase, the pulled down type of resistor & weak keeping circuit remain in-active, even then, the users have the options to pull inputs up's. The activation of pulled up resistor before configurations are globally governed by the configuration's modular pins of the IC chips. If the pulled up resistor is not enabled, all pins will remain in a floating state. Consequently, it becomes necessary to provide external pull up's or pull-down's resistors for pins for requiring a clearly defined logical levels before configurations. Furthermore, all pads are safeguarded over the potential damages from the electro-static discharges [ESD] & the over voltage transient. Following configurations, clamping diode is connected to the VCCO to adhere to LVTTLs, PCIs, HSTLs, SSTLs, CTTs & the AGP based standards [5].

4. Configurable Logical Blocks

Then, fundamental component of the Spartan based IIE CLB are the Logical Element (LC). Each L-C comprises a four i/p based functional generators, then the carry logics & a storage elements. The output generated by the functional generators within each of the LC either will be driving the CLB's o/p or then or serves as the D inputs for the flip-flops. Within a Spartan based 'IIE' CLBs, there are 4 LC's, organised into 2 identical slice.. Figure 4 illustrates a single slice. Beyond the four basic LCs, the Spartan-IIE CLB also incorporates logics which will be combining the functional generator for offering the functionalities involving 5 – 6 – 8 input [6]. The fundamental unit within the Spartan based IIE CLBs are referred to as the logical element [LC]. An L-C will be consisting of a four i/p generator, the carry based logic & a storages elements. The outputs generated by the functional generators within every LC can either drive the CLB outputs or served as the D type inputs for the flip flops. Within a Spartan-IIE CLBs, there are 4 LC's, organised into two similar slices. Figure 4 illustrates a solo slices. Apart from the four foundational LC's, the Spartan based IIE CLBs incorporates additional logics that amalgamates functional generator to offer functionalities involving 5 or 6 or 7 inputs.

5. Look Up Table

LUTs with four inputs are used to create Spartan IIE function generators. Each LUT can serve as a function generator as well as a (16 x 1) bit synchronous type of RaM. Additionally, a slice's two LUTs can be joined to produce a (16 x 2) bit, (32 x 1) bit, or 16 x 1-bit dual type of port synchronous type RaM. An 16 bit shift registers which is capable of recording high-speed or burst-mode data can also be provided by the Spartan-IIE LUT. Additionally, this modes could be utilized for storing of the datas in programs like DSP Processor [7]. In addition to acting as a functional generators, every of the LUT also has the ability to work as a (16 x 1) bit synchronous type of RaM. Additionally, a (16 x 2)-bit or (32 x 1) bit synchronous RAM, or even a 16 x 1-bit dual-port synchronous RAM, can be produced by joining the two LUTs in a slice. The Spartan-IIE LUT also has the capacity to act as a 16 bit shift registers, which makes it useful for activities like burst type of mode or some of the high speed data's that could capture. Additionally, this mode could be successfully applied to the storing of datas in programs like the DSP.

6. Storage Type of Elemental Devices

Then every storages of the components in the Spartan type of IIE slice could be configure in a variety of ways, enabling them to work either as levelled type of sensitive latch / edge type of triggered D type flipflops. When necessary, the functional generator could be skipped and the storage elements can take D inputs straight from the sliced i/p's or from the internal functional generator of the slice. Every sliced synchronous sets & the resetted signals for ex., SR & BY, in addition to the clock and clock enable signals. A storage element's configuration-defined startup state is started by the SR signal, however the BY signal can make it go the other way. The signal could also be configure to run in the asynchronous modes. It should be noted that the 2 flipflops w.r.t. the slices can independently invert all control signals. [8].

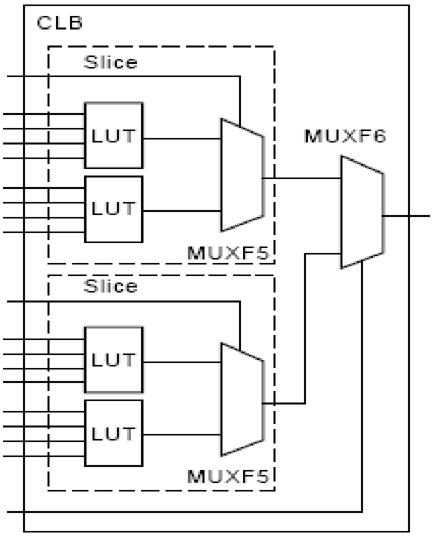

Fig. 4 : F5 F6 Multiplexers

7. Additional Logic

In each slice, the F5 multiplexer merges the outputs from the function generators (as shown in Figure 5). This fusion yields the capability to serve as a function generator capable of implementing any five i/p functions, a 4 to 1 multiplexers or chosen functions involving upto 9 i/p's. Likewise, the F6-multiplexers takes the o/p's from all the 4 functional generator within the CLB's and selects 1 of the 2 outputs from the F5 multiplexers. This arrangement allows for the execution of any six i/p functions, an 8:1 multiplexers or the implementation of selected function using upto 20 input [9].

8. Arithmetic Logic

Specialized carry logic within the Spartan-IIE offers rapid arithmetic carry capabilities, ideal for higher speed arithmetical function. Each slice of then Spartan-IIE CLB support two distinct carry chain, resulting in a total heights of 2 bits per CLBs for these chains.   The arithmetical logics within this system include an XOR gates, enabling the implementation of a one bit full type of adder with in an LC. Furthermore, there is a particular type of 'AND' gate that enhances the power & efficiencies of the multiplier's implementation & the particular carry paths could also be cascaded functional generator, facilitating the implementation of expansive logic functions [10].

7. BUFT's

Each Spartan-IIE CLB is equipped with a pair of 3-state drivers, known as BUFTs, designed to facilitate the driving of on-chip buses. Additionally, the IOB's situated on both the right & left sides of the CLB could also be contributed to driving these on-chip buses. Each BUFT within the Spartan-IIE is equipped with its own independent tri-stated controlled pins & an separate inputted pins. The tri stated controlled pins operates on an active type of Low enabled [T] basis. When all BUFTs connected to a particular nets were being inactivated, then nets assumes a higher state. It's worth noting that there will be no mandatory requirement to in-stantiate the pull-up's till it is specifically required for the purpose of simulation. Importantly, at the same time to drive a multiple BUFT's on-to the same type of net doesn't result in contentions. In the event that a net is drives both low & high concurrently, the nets will register as Low [11].

8. Clock Distribution

The Spartan based IIE families will ensures efficient higher speeds and minimal skew type of clock distributions by utilizing the primary type of global routing resource outlined earlier. Figure 5 illustrates a standard clock distribution network. The device features 4 type of global buffers, positioned with 2 @ the upper portion of the centre & 2 @ the then immediate bottom of the center IC chip [12]. The next buffers serve as drivers for the four principal global nets, which, in turn, distribute signals to various clock pins. Additionally, there are 4 dedicated type of clock pad, strategically positioned adjacently to either of every global type of buffer.

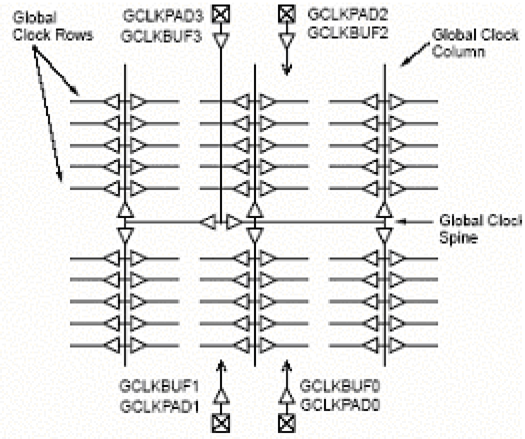

Fig. 5 : Clock distributions

9. Delay Type of Locked LOOPS (DLLs)

A fully digital Delay type of Locked Loops [DLL] is connected to every global clocked i/p buffers to eliminate any potential skewed b/w the i/p of the clock pad & the internal type of clocked i/p pins which are available thro'out the device module. Then, the clock input pad and the internal clock-input pins throughout the device. The ability to driven 2 global clocked network exists within a single DLL's. The distribution clock and input clock are both actively monitored by the DLL, which in the automatic mode will modifies a clock delay component (as shown in Figure 6). By adding further delay, this makes sure that clock edges arrive at internal type of flipflops actually 1 clock time period after which they will initially do at the inputs. Essentially, this closedloop device ensures that clocks edges completely line up with those hitting the input when they reach the internal flip-flops, essentially eliminating any clock-distribution delay [13].

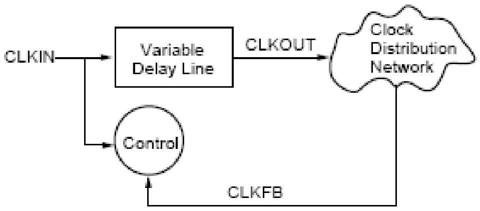

Fig. 6 : Pictorial Clock distribution

10. VHDL

VHDL actually is a specialized programming languages tailored for describing any type of digital system behavior, encompassing everything from basic logic gates to complex microprocessors and custom chips. It excels at capturing the precise electric properties & aspects of circuit's behaviors, including signal fall & the rise times, gate function operation & the delay [14]. These VHDL simulation models can serve as foundational elements within larger circuits, facilitating simulation through schematics, block diagrams, or system-level VHDL descriptions. Beyond its role in system simulation, VHDL also functions as a versatile general-purpose programming language, enabling the translation of intricate electronic circuit behaviors into a design system for either automated circuit synthesis or system-level simulation [15].

While VHDL shares similarities with high-level programming languages like Pascal, C, and C++ in terms of structured design techniques and control/data representation, it uniquely accommodates concurrent event descriptions. This feature is vital because hardware described using VHDL inherently operates concurrently. An important application of VHDL is creating test benches, which encapsulate circuit performance specifications by defining stimuli and expected outputs for behavior verification over time [16]. Test benches should be integral components of any VHDL project, developed in tandem with circuit descriptions. VHDL offers versatility in describing electronic hardware across various levels of abstraction. Understanding the 3 categories of the abstract, viz., the algo, register level transfer, the gate levelled transfer or the RGL schematic & the gate level schemtic—is essential for FPGA/ASIC design. Algorithms are not synthesizable, RTL is the synthesis input, and the output post-synthesis is represented by the gate level. Based on temporal concerns, these degrees of abstraction are distinct from one another. [17].

11. Algorithm

In hardware design, different levels of abstraction exist for VHDL descriptions. At the algorithmic level, you define a sequence of instructions without detailed timing information or a clock. Some behavioral synthesis tools can work with algorithmic VHDL code, although they might require artificial constraints like an 'algorithm' clock to synchronize operations [18].

12. RTL

Moving to the RTL (Register Transfer Level) description, you introduce an explicit clocks, and all operation will be scheduled within particular type of clock cycle, without specifying detail type of delay well below the cycle levels. While a single global clock isn't mandatory, it's often preferred. Retiming, which allow the operation to be rescheduled across every clock cycle is a feature in commercially available synthesis tools, but it's not as flexible as in behavioral synthesis [19].

13. RTL Synthesis

The gate-level description comprises gates& register instantiated from a technical based library, that includes technology specific delays data for each gate [18].

14. GATE

Currently, designing hardware in VHDL primarily occurs at the RTL level because of the capabilities of synthesis tools. The RTL level offers a balance between high-level abstraction and low-level implementation details. While gate-level descriptions are too low-level, focusing on implementation, and algorithmic descriptions are often too high-level for most synthesis tools. It's anticipated that as synthesis technology evolves, algorithmic VHDL might become the norm, but for now, VHDL coding at the RTL level remains the most effective approach for synthesis tool input [20].].

15. Design Flows using the VHDL

Every practical designs scenario, every of the similar steps that is being explained in the previous steps could or can often be further splitted down into a number of smaller & smaller steps, more manageable sub-steps. Additionally, parts of the design process may require iteration as errors and issues are discovered [21].

System-Level Verification: Initially, VHDL can be employed to models & do the simulation of various aspects of the complete systems, which may consist of 1 or more device. This system-level modeling can take two forms: a comprehensive system description that allows for FPGA-ASIC specifications validation before abridged designs or a partial descriptions which will abstract every specific system properties, such as performances modeling to identifying of the potential bottleneck [22].

Once the idealized system’s modules architectures & the partition will be clearly defined, the elaborated designs step for each FPGA/ASIC can start. This entails generating a series of test cases in VHDL and capturing the detailed desgns in the VHDL coding and @ the RTL levels. To ensure appropriate interpretation of the specification, these tasks are frequently completed by multiple teams. If automatic logic synthesis is intended, it is essential that the RTL VHDL be synthesizable.. Generating effective test cases is a significant undertaking requiring structured approaches and creative problem-solving, as the qualities of the last & final FPGA - ASIC heavily relies on test case coverages [23].

RTL Verifications : The RTL’s VHDL design will be finally subjected to simulation to validation its functionality against the specifications. RTL’s simulations are considerably faster than gate-level simulation, and it's commonly recommended to invest time in extensive simulation rather than cutting corners. In practice, a substantial portion of the design cycle (around 70-80%) is dedicated to writing and simulation of the VHDL @ & above the register's transfer levels, while the remaining time (20-30%) is allocated to gate-level synthesis and verification [24].

Look Ahead Syntheses : While few of the preliminary synthesis work might be carried out at the beginning of the design processes & the data of the aera required for the VLSI integrations for assessing architectural decisions and confirming VHDL syntheses understanding, the primary synthesis production run is typically deferred til functional simulations are completed. This approach avoids unnecessary investment in synthesis until the design's functionality has been adequately validated. [25].

16. VHDL CODES

The VHDL codes for various devices are as shown below.

A. VHDL CODE FOR UART TRANSMITTER

B. VHDL CODE FOR UART RECEIVER

17. CONCLUSIONS

Here, w.r.t. this proposed research papers, the authors have given the simulation & designing of a µC dependent framework of the system using VHDL / Modelsim codings with FPGA concepts. The code was developed, the program was run & its effectiveness was observed. VHDL Code was simulated and implemented using a FPGA kit and excellent results were obtained. Field-programmable door clusters (FPGA's) are option in contrast to programmable rationale gadgets (PLD's) and ASICs. As their name suggests, FPGA's offer the huge advantage of being promptly programmable. In contrast to their fore bearers in the PLD classification, FPG’'s can (by and large) be modified over and over, giving creators different chances to change their circuit. For then, execute the undertaking VHDL's is chosen as HDLs. Xilinx ISE version 6.1 as the apparatus to blend and Modelsims for recreating the VHDL Code.

References

[1] Ayala Kenneth, The 8051 microcontroller : Architecture, Programming and Applications, *Text Book*, India.

[2] https://en.wikipedia.org/wiki/Microcontroller

[3] Moskowitz, Sanford L., “Advanced Materials Innovation: Managing Global Technology in the 21st century”, *John Wiley & Sons*., pp. 165–167, 2016.

[4] https://www.maximintegrated.com/en/design/technical-documents/app-notes/6/691.html

[5] S.L. Toral, J.M. Quero, M. Elena, and L. G. Franquelo, “A microprocessor based system for ECG telemedicine and telecare”, *Proc. IEEE Int. Symp. Circuits Sys. ISCAS 2001*, Sydney, Australia, vol. 4, pp. 526–529, 2001.

[6] Welch Allyn OEM Technologies, “ECG/Respiration ASIC: ECG 3-lead, 5-lead, 12-lead and SP Signal Processing”, *Internet Article*, 11/09/2003.

[7] D. Lucani, G. Cataldo, J. Cruz, G. Villegas, and S. Wong, “A portable ECG monitoring device with Bluetooth and Holter capabilities for telemedicine applications”, *Proc. 28th IEEE EMBS Ann. Int. Conf.*, New York City, USA, 2006.

[8] I. Romero Legameta, P.S. Addisson, N. Grubb, C.E. Robertson, K.A.A. Fox, and J.N. Watson, “Real-time classification of ECGs on a PDA”, *IEEE Trans. Inform Technol. Biomed*., vol. 30, pp. 565–568, 2003.

[9] R. Istepainan, R. García, A. Alesanco, and S. Olmos, “A novel realtime multilead ECG compression and de-noising method based on the wavelet transform”, *IEEE Trans. Inform. Technol. Biomed*., vol. 30, pp. 593–596, 2003.

[10] S. Borromeo, C. Rodriguez-Sanchez, F. Machado, J.A. Hernandez Tamames, and R. de la Prieta, “A reconfigurable, wearable, wireless ECG system”, *Proc. 29th Ann. Int. Conf. IEEE EMBS*, Lyon, France, 2007.

[11] C.-N. Chien, H.-W. Hsu1, J.-K. Jang, C.-L. Rau1, and F.-S. Jaw, “Microcontroller-based wireless recorder for biomedical signals”, Proc. 27th *IEEE Ann. Conf. Engin. Medicine and Biology*, Shanghai, China, 2005.

[12] ECG Front-End Design is Simplified with MicroConverter”, *Analog Devices Application*, India.

[13] U. Rajendra Acharya, A. E. Spaan, J. Suri, and S. M. Krishnan, “Research article”, *Advances in Cardiac Signal Processing, Springer*, 2007.

[14] T. Degen and H. Jäckel, “A pseudodifferential amplifier for bioelectric events with DC-offset compensation using two-wired amplifying electrodes”, *IEEE Trans. Biomed. Engg*., vol. 53, no. 2, pp. 300–310, 2006.

[15] J. Webster, “Medical Instrumentation Application and Design”, *John Wiley & Sons*, 2010.

[16] Microchip. PIC16F876 Datasheets [Online].

[17] AAMI, American National Standard, Safe Current Limits for Electromedical Apparatus (ANSI/AAMI ES1-1993), *Association for the Advancement of Medical Instrumentation*, 1993.

[18] J. Pan and W.J.Tompkins, “A real time QRS detection algorithm”, *IEEE Trans. Biomed. Engg*., vol. 32, pp, 230–236.

[19] Heart-Rate and EKG Monitor Using the MSP430FG439, *Texas Instruments Application Report*, 2008.

[20] Li Tan, “Digital Signal Processing: Fundamentals and Applications”, *Elsevier*, 2008.

[21] Rahimunnisa Nagma, Pavithra G., Dr. T.C.Manjunath, “Traffic aware video coding using scalable video coding (SVC)”, *2nd Int. Conf. on recent trends in Signal processing, Image Processing & VLSI, ICrtSIV - 2015*, Don Bosco Inst. of Tech., Bangalore, Karnataka, India, 15-16 May 2015.

[22] Avin Richards, Dr. T.C.Manjuanth, “Polymer Memory Advancement in the VLSI designs”, *National Conf. of Electromechanical & Automated Systems* (NCEMAS-10), Dept. of EEE, Sri Eshwar College of Engg., Kondampatii Post, Vadasittur, Kinathukadavu, Coimbatore-641202, Tamil Nadu, India, Paper No. 19, pp. 104-111, 18th Dec. 2010.

[23] Pavithra G., Dr. T.C.Manjunath, Dr. Prathibha Harish (Medical College Doctor), “Simulation, detection & implementation of chronic eye ailments using real time embedded systems – A review”, *International Journal of Management, Technology & Engineering, UGC Approved Journal* Serial No. 45550, Impact Factor 6.3, Volume IX, Issue VI, ISSN No. : 2249-7455 (IJMTAE-19), DOI:16.10089.IJMTE.2019.V9I6.19.28560, Thomson Reuters’ Research ID : H-2418-2017 Indexed Journals, An ISO 7021 : 2008 Certified Journal, pp. 767-772, June 2019.

[24] Pallavi R. Bhat, Pavithra G, Dr. T.C. Manjunath, “Development of a GUI for finding the moments of objects in a image using shape analysis”, *JASC: Journal of Applied Science and Computations*, ISSN NO: 1076-5131, DOI:16.10089.JASC.2018.V5I12.453459.1500227, ISO 9001:2008: Certified, JASC Approved By: UGC, NSL, NISCAIR, CSIR, Volume V, Issue XII, pp. 1003-1008, IF - 5.8, Paper id 142, An ISO 7021 : 2008 Certified Journal, DOI : 16.10089/JASC, UGC Approved Journal- 41238, Dec. 2018.

[25] Shashikanth Patil, Dr. T.C.Manjunath, Pavithra G., “A Novel Method of Identifying Different Patterns in Pattern Recognition”, *IFERP’s International Journal of Engineering Research in Mechanical and Civil Engineering* (IJERMCE), DOI : 01.1617/vol4/iss5/pid93174, ISSN (Online) 2456-1290, ISSN (Online) 2456-1290, indexed by Google Scholar, Thomson Reuters ID, ORCiD Research ID, Impact Factor 3.8, Vol. 2, Issue 5, pp. 967-970, May 2017.