

A
Project Report
On
**DESIGN OF WINDOWED WATCH DOG TIMER FOR EMBEDDED SYSTEM
FAULTY CONTROL**
Submitted in partial fulfillment of the requirement for the award of

BACHELOR OF TECHNOLOGY
In
ELECTRONICS AND COMMUNICATION ENGINEERING
By

DONEKAL NANDINI	(22AM5A0404)
YEDDULA SAHITHYA	(21AM1A0474)
GOPIREDDY THARUN KUMAR	(21AM1A04B2)

Under the esteemed guidance of

Dr. K. VASUDEVA REDDY M.Tech, Ph.D.

Professor, Dept. of ECE



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SVR ENGINEERING COLLEGE
(AUTONOMOUS)

Approved by AICTE, New Delhi | Recognized under Sections 2(f) and 12(B) of
the UGC Act, 1956 | Affiliated to JNTUA, Ananthapuramu
Accredited by NAAC and NBA | ISO 9001:2015 Certified Institution
Ayyalur Metta, Nandyal District – 518502, Andhra Pradesh

2021-2025

SVR ENGINEERING COLLEGE

(AUTONOMOUS)

Approved by AICTE, New Delhi | Recognized under Sections 2(f) and 12(B) of
the UGC Act, 1956 | Affiliated to JNTUA, Ananthapuramu
Accredited by NAAC and NBA | ISO 9001:2015 Certified Institution
Ayyalur Metta, Nandyal District – 518502, Andhra Pradesh



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled “**DESIGN OF WINDOWED
WATCH TIMER DOG FOR EMBEDDED SYSTEM FAULTY CONTROL**” is the
bonafide work done and submitted by

DONEKAL NANDINI (22AM5A0404)

YEDDULA SAHITHYA (21AM1A0474)

GOPIREDDY THARUN KUMAR (21AM1A04B2)

In partial fulfillment of the requirement for the award of the degree of **Bachelor of
Technology in Electronics and Communication Engineering** in the **SVR
ENGINEERING COLLEGE (Affiliated to Jawaharlal Nehru Technological
University Ananthapuramu)** is a record of Bonafide work carried out by them under our
guidance and supervision.

The results embodied in this thesis have not been submitted to any other university or
institute for the award of any degree.

Project Guide:

Dr.K. VASUDEVAREDDY M. Tech, Ph.D.

Professor

Dept. of ECE, SVREC.

Head of the Department:

Dr. G. LAKSHMI NARAYANA M. Tech, Ph.D.

Professor & HOD

Dept. of ECE, SVREC.

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project report entitled “**DESIGN OF WINDOWED WATCH DOG TIMER FOR EMBEDDED SYSTEM FAULTY CONTROL**” was carried out by us during the academic year 2024–2025, in partial fulfillment of the requirements for the award of Bachelor Technology in Electronics and Communication Engineering from SVR Engineering College, Of affiliated to Jawaharlal Nehru Technological University, Ananthapuramu. We further declare that this project has not been submitted to any other university or organization for the award of any other degree or diploma

DONEKAL NANDINII (22AM5A0404)

YEDDULA SAHITHYA (21AM1A0474)

GOPIREDDY THARUN KUMAR (21AM1A04B2)

ACKNOWLEDGEMENT

We sincerely take this opportunity to express our heartfelt gratitude to all the distinguished individuals who have supported and guided us in successfully completing this project.

First and foremost, we extend our deep gratitude to our guide, **Dr. K. VASUDEVA REDDY** ^{M.Tech, Ph.D}, Professor of ECE, S.V.R. Engineering College, for his invaluable guidance, continuous support, and encouragement throughout the course of this project. His insightful discussions and motivation provided us with an excellent environment to work in.

We are also grateful to our project coordinator, **Dr. D. Raghunatha Rao** ^{M.Tech, Ph.D}, Associate Professor in the Department of ECE, S.V.R. Engineering College, for his unwavering encouragement and guidance during the project.

Furthermore, we extend our sincere thanks to the **Head of the Department** of ECE, **Dr. G. Lakshminarayana**, ^{M.Tech, Ph.D.}, for his constant encouragement and assistance, which played a vital role in the successful completion of this project.

We would also like to express our gratitude to our esteemed Principal, **Dr. P. Mallikarjuna Reddy Garu**, for his motivation and for providing all the necessary facilities that enabled us to carry out this work effectively within the college premises.

We are deeply thankful to our Honorable Chairman, **Sri S. Venkata Rami Reddy Garu**, and our Honorable Managing Director, **Sri S. Dinesh Reddy Garu**, for their continuous moral support and for ensuring that we receive a quality education from experienced faculty members.

Additionally, we sincerely appreciate **all the faculty and staff** of the Electronics and Communication Engineering Department, who have directly or indirectly contributed to our learning and the successful execution of this project.

Finally, we express our heartfelt gratitude to our **family and friends**, whose unwavering support, encouragement, and love have been our greatest strength throughout this journey

DONEKAL NANDINI (22AM5A0404)

YEDDULA SAHITHYA (21AM1A0474)

GOPIREDDY THARUN KUMAR (21AM1A04B2)

Vision and Mission of the institute

Vision

To produce competent engineers with strong basics of engineering knowledge, a mindset of lifelong learning and the necessary complementary skills needed to be successful professionals.

Mission

- ❖ To impart Affordable and Quality Education to meet the needs of Society
- ❖ To provide Ambient Teaching - Learning Environment, Adequate Infrastructure and Resources.
- ❖ To empower the students through Complementary Skills such as Decision making, Interpersonal, Ability to apply knowledge across disciplines, Ability to work effortlessly with others.

Vision and Mission of the Department

Vision

To produce highly skilled, creative and competitive Electronics and Communication Engineers to meet the emerging needs of the society.

Mission

- To impart a strong foundational knowledge and essential skills in Electronics and Communication Engineering through innovative teaching and hands-on learning experiences, enabling graduates to effectively identify, analyses and solve complex engineering problems
- To promote critical thinking, ethical practices, and social responsibility, preparing students to excel in diverse professional environments while fostering a commitment to lifelong learning and continuous professional development
- To cultivate well-rounded competencies and complementary skills in students, empowering them for successful careers, advanced studies, and entrepreneurial ventures, while instilling a strong sense of professional ethics and social awareness aligned with societal needs.

Program Educational Objectives (PEOs)

- PEO1: Graduates apply their knowledge of mathematics and science to identify, analyze and solve problems in the field of Electronics and develop sophisticated communication systems.
- PEO2: Graduates embody a commitment to professional ethics, diversity and social awareness in their professional career.
- PEO3: Graduates exhibit a desire for life-long learning through technical training and
- professional activities.

Program Specific Outcomes (PSOs)

- PSO1: Apply the fundamental concepts of electronics and communication engineering to design a variety of components and systems for applications including signal processing, image processing, communication, networking, embedded systems, VLSI and control system.
- PSO2: Select and apply cutting-edge engineering hardware and software tools to solve complex Electronics and Communication Engineering problems.

DEPT.	ELECTRONICS AND COMMUNICATION ENGINEERING													
PROGRAM OUTCOME ATTAINMENT														
PO 1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems													
PO 2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences													
PO 3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.													
PO 4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.													
PO 5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.													
PO 6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice													
PO 7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.													
PO 8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.													
PO 9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.													
PO 10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.													
PO 11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments													
PO 12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.													
PSO 1	PSO1: Apply the fundamental concepts of electronics and communication engineering to design a variety of components and systems for applications including signal processing, image processing, communication, networking, embedded systems, VLSI and control system													
PSO 2	PSO2: Select and apply cutting-edge engineering hardware and software tools to solve complex Electronics and Communication Engineering problems													
TITLE	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
Design of windowed watch dog timer for embedded system faulty control	3	3	3	2	2	2	1		2	1	2	2	3	3



Abstract

Embedded systems are at the core of modern electronics, with industries reporting a 25% annual increase in their adoption due to advancements in automation, control, and communication technologies. In such systems, timing faults can lead to critical failures, affecting the reliability of automotive, medical, and aerospace applications. Existing 2x2 Watchdog Timers (WDT) are inefficient at handling complex timing faults, leading to up to 30% system downtime due to undetected failures. Current 2x2 WDT controllers suffer from limited fault detection capability and poor timing resolution, which can result in undetected errors in real-time embedded systems. This paper introduces a novel Windowed Watchdog Timer (WDT) design that enhances fault detection operation by monitoring both early and late faults within a specified window. The improved WDT interface provides seamless integration with microcontrollers, enhancing timing precision and system reliability. This design addresses timing faults more effectively, leading to a more robust embedded system.



TABLE OF CONTENTS

LIST OF TOPICS	PAGE NO
ABSTRACT	i
TABLE OF CONTENTS	ii
LIST OF FIGURES	v
LIST OF TABLES	vi
CHAPTER 1	
INTRODUCTION	1-7
1.1 Overview	1
1.2 Research Motivation	1
1.3 Existing System	2
1.4 Problem Statement	2
1.5 Research Objective	3
1.6 Performance Metrics	3
1.7 Advantages	5
1.8 Applications	6
CHAPTER 2	8-10
LITERATURE SURVEY	8
Introduction	8



CHAPTER 3	11-14
Existing Systems	11
3.1 Introduction	11
3.2 Existing WDT Controller	12
3.3 Drawbacks	13
3.4 Summary	14
CHAPTER 4	15-20
Proposed system	15
4.1 Introduction	15
4.2 Proposed Methodology	15
4.3 Fault Detection Operation in WDT	17
4.4 WDT Interface	19
CHAPTER 5	21-28
VLSI DESIGN	21
5.1 Introduction	21
5.2 Asics	22
5.3 FPGA	23
5.4 Verilog	
5.5 Filed Programmable Gate Arrays Filed	28



CHAPTER 6	29-43
SOFTWARE TOOLS	29
6.1 Introduction	29
6.2 Hardware Requirements	29
6.3 Software Requirements	29
6.4 Introduction to XLINIX Ise	29
6.5 XLINIX Design Process	30
6.6 XLINIX Software	31
6.7 FPGA Design Flow	40
CHAPER 7	44-49
RESULTS AND DISCUSSION	44
CHAPTER 8	50-51
CONCLUSION AND FUTURE SCOPE	50
8.1 Conclusion	50
8.2 Future Scope	50
REFERENCES	



LIST OF FIGURES

NAME OF FIGURES	PAGE NO
Figure 4.1. Proposed Watchdog timer operational diagram	16
Figure 4.2. Fault detection logic using FSM	18
Figure 4.3. Interfacing watch dog timer with multiple Input-outputs	19
Figure 6.1. FPGA design flow	40
Figure 6.2. FPFA Synthesis	41
Figure 6.3. FPGA Translate	42
Figure 6.4. FPGA map	42
Figure 6.5. FPGA place and route	43
Figure 7.1. Simulation output	44
Figure 7.2. Design summary	45
Figure 7.4. Setup time summary	45
Figure 7.5. Hold time summary	46
Figure 7.6. Power summary	46



LIST OF TABLES

NAME OF THE TABLES	PAGE NO
Table.1. Area performance comparison	47
Table.2. Delay Performance comparison	48
Table.3. Power performance comparison	49

DESIGN OF WINDOWED WATCH DOG TIMER FOR EMBEDDED SYSYEM FAULTY CONTROL

CHAPTER 1

Introduction

1.1 Overview

Watchdog Timer (WDT) is a key component in chip multiprocessors (CMPs) as it supports communication between many cores. However, WDT for modern chips contribute up to 30% of the chips overall power budget. Also, as chips continue to shrink, the contribution of WDT power is projected to increase. Routers. WDT Controller comprises several input ports, a number of output ports, a switching matrix connecting the input ports to the output ports, and a local port to access the IP core connected to this router. The router also contains a logic block that implements the flow control policies.

- The first and most important ones are the links that physically connect the nodes and implement the communication.
- The second block is the router, which implements the communication protocol.
- The last building block is the network adapter (NA) or network interface (NI). This block makes the logical connection between the IP cores and the network

On-chip routers are a crucial component in System-on-Chip (SoC) designs, facilitating communication between various components on the chip. Improving latency and throughput in on-chip networks is essential for enhancing the overall performance of integrated circuits. Some strategies and considerations include:

Topology Design: Choosing an efficient network topology is critical. Mesh, torus, or hierarchical topologies are commonly used, and the selection depends on the specific requirements of the application.

Routing Algorithms: Implementing efficient routing algorithms can significantly impact latency and throughput. Adaptive or dynamic routing algorithms can adapt to changing traffic patterns and optimize communication paths.

Quality of Service (QoS): Implementing QoS mechanisms can prioritize certain types of traffic, ensuring that critical data receives preferential treatment in terms of routing and resources.

1.2 Research motivation

The motivation for Very Large-Scale Integration (VLSI) based WDT design stems from the imperative need to efficiently manage and enhance on-chip communication within complex integrated circuits.

The efficiency of the WDT's integrated circuit becomes pivotal in determining the overall success of data transmission. If the WDT's integrated circuit operates with optimal efficiency, it ensures perfect reception of data by User 2. However, in the event of inefficiency or malfunction within the WDT's integrated circuit, there is a risk of.

1.3 Existing System

The implementation of a circuit-switching protocol in a WDT Controller involves designing the hardware components such as switch fabrics, buffers, and control logic to support the establishment, maintenance, and release of dedicated paths. It also requires careful consideration of the layout and interconnection of these components to optimize for performance, power, and area constraints, which are critical factors in VLSI design.

It's worth noting that while circuit-switching can offer advantages in terms of low latency and deterministic behaviour, it may not be as flexible as packet-switching in handling variable traffic patterns and may lead to inefficient resource utilization for bursty or sporadic communication. The choice between circuit-switching and packet-switching depends on the specific requirements and characteristics of the targeted application.

A packet switching protocol in a WDT Controller involves designing and optimizing the hardware components, including the switching fabric, buffers, routers, and interconnection network. This requires careful consideration of factors such as power consumption, area utilization, and overall performance. Packet switching is widely used in modern WDT architectures due to its flexibility, adaptability to varying traffic patterns, and efficient resource utilization.

1.4 Problem Statement

The growing complexity and scale of modern VLSI designs pose significant challenges for on-chip communication, particularly in the context of WDT Controllers. As the number of processing elements and IP cores integrated on a single chip continues to rise, the existing WDT Controller architectures encounter limitations in terms of bandwidth and scalability.

Design of windowed watch dog timer for embedded system faulty control

Consequently, there is a pressing need to address these on-chip communication bottlenecks and enhance the efficiency of WDT Controllers to support the increasing demands of contemporary VLSI designs.

Power consumption remains a critical concern in VLSI design, and WDT Controllers are no exception. The continuous evolution of semiconductor technologies allows for the integration of a higher number of components on a chip, but it also brings challenges related to power dissipation. WDT Controllers, responsible for managing data flow among various components, contribute significantly to the overall power consumption. Inefficient routing algorithms, excessive data movement, and suboptimal buffer management within the routers can lead to unnecessary power consumption. Balancing the need for high-performance communication with power-efficiency is a delicate trade-off that necessitates innovative approaches in the design and implementation of WDT Controllers for VLSI applications. VLSI systems are often subjected to dynamic workloads, varying traffic patterns, and potential hardware faults. WDT Controllers need to dynamically adapt to these changes to ensure optimal performance and reliability. The challenge lies in designing routers that can efficiently reconfigure their routing paths, allocate resources dynamically, and maintain fault tolerance in the presence of hardware failures.

1.5 Research Objective

A WDT is a new complex system-on-chip (SoC) designs that provide efficient on chip communication networks. It allows measurable communication and allows decoupling of communication. The data is routed through the networks in terms of packets. The paper presents an innovative approach to designing WDT Controllers that prioritize area efficiency while facilitating seamless D2D communication. Existing WDT architectures typically employ either circuit switching or packet switching techniques, each with its own limitations. Circuit switching can lead to high latency due to setup time, while packet switching may suffer from increased power consumption and congestion.

1.6 Performance Metrics

1.6.1 Area Metrics

LUT (Look Up Table): The LUT is a lookup table. For a 4-input LUT, it is actually a 4-bit address bit. A bit-bit memory can store 16-bit data, so we can use the LUT to build distributed RAM in FPGA design. This also explains why we have to implement pipeline implementation in our design. It implements digital logic functions.

Design of windowed watch dog timer for embedded system faulty control

IO (Input & Output): Number and types of processing elements to be integrated, communication requirements are between the cores which are like (bandwidth, latency, etc.). The power constraints, area constraints and Performance requirements these are the inputs Layout of the network topology and specifying how the processing elements are interconnected. Routing algorithms and protocols to ensure efficient data transfer between cores.

FF (FlipFlop): In a WDT design using VLSI, flip-flops are used for storing and synchronizing data as it travels through the network. Flip-flops are used in various elements of the WDT, including routers, switches, buffers. They play a critical role in the proper functioning of a WDT design by providing storage, synchronization necessary for efficient data transfer and communication between components.

Buffering: Flip-flops are used in buffers to temporarily store data packets as they move through the network. These flip-flops ensure that data is properly synchronized and transmitted in the correct timing.

1.6.2 Power Metrics

Starting Power (or Leakage Power): This is the power consumed by the WDT when it is in the idle or standby state. When there is no active data transmission, then there is still some power dissipation due to leakage currents in transistors. Minimizing starting power is crucial for energy-efficient designs, mainly in battery-powered devices.

Dynamic Power: Dynamic power is associated with the energy consumption during the active operation of the WDT, when data is being transmitted.

Logical Power: This refers to the power consumed by the logical and arithmetic operations within the WDT components. It includes power dissipated in flip-flops, logic gates, and other computational elements. Reducing logical power often involves the design at the gate level and choosing efficient logic structures, optimizing clocking schemes.

1.6.3 Delay Metrics:

Setup Delay: Setup delay refers to the time required for data to stabilize at the input of a flip-flop before the arrival of the clock edge. The setup delay is critical for ensuring that data signals are stable and valid before by the receiving elements. Here the Violating setup time constraints can lead to data and timing errors.

Design of windowed watch dog timer for embedded system faulty control

Hold Delay: Hold delay is the minimum time data must remain stable after the arrival of the clock edge to ensure correct data capture by the flip-flop. In WDT designs, hold delay is essential for maintaining the integrity of data as it propagates through the network.

Logic Delay: Logic delay refers to the time taken for a signal to propagate through combinational logic elements such as gates and multiplexers. In WDT designs, logic delay contributes to the overall latency of data transmission between network nodes.

Net Delay: Net delay encompasses the propagation delay of signals traveling through interconnects such as wires, metal layers, and routing switches in the WDT. It includes both intrinsic delay (due to the physical properties of the interconnects) and routing delay (due to congestion and contention in the network).

1.7 Advantages

WDT Controllers provide several advantages in VLSI design, including scalability, modularity, reduced latency, increased bandwidth, fault tolerance, power efficiency, flexible topologies, and support for QoS, making them a suitable choice for complex SoC designs

Scalability: WDT Controllers enable the design of scalable systems. As the number of IP blocks or cores on a chip increases, traditional bus-based communication architectures become inefficient due to contention and increased latency.

Modularity: WDT Controllers facilitate a modular design approach. IP blocks can be designed and optimized independently, and the WDT provides a standardized communication infrastructure. This modularity simplifies the design process and allows for easier integration and reuse of IP blocks.

Reduced Latency: WDT Controllers can be designed to minimize communication latency between different IP blocks. The packet-switched nature of WDT allows for concurrent communication between multiple IP blocks, reducing the overall latency compared to traditional bus architectures.

Increased Bandwidth: WDT Controllers can support higher bandwidth communication compared to bus-based architectures. The packet-switched nature of WDT allows for more efficient utilization of the available communication resources, leading to increased overall system bandwidth.

Fault Tolerance: WDT can be designed with fault-tolerant features, allowing for reliable communication even in the presence of faults or failures. Redundant paths and adaptive routing algorithms can be implemented to ensure that communication continues in the event of a failure.



Power Efficiency: WDT Controllers can be optimized for power efficiency. By implementing power management techniques such as dynamic voltage and frequency scaling, and by efficiently managing the communication between IP blocks, WDT contribute to overall power savings in the SoC.

Flexible Topologies: WDT support a variety of network topologies, such as mesh, torus, or hierarchical structures. Designers can choose the topology that best suits the specific requirements of the application, balancing factors like communication efficiency, area utilization, and power consumption.

QoS (Quality of Service): WDT Controllers can be designed to support QoS requirements, ensuring that critical traffic receives higher priority and lower latency. This is important in applications where real-time or high-performance communication is essential.

1.8 Applications

Multicore Processors: In a multicore processor, where multiple processing cores share resources and communicate with each other, a Hybrid Connected WDT Controller can optimize communication paths and reduce latency. This is critical for achieving efficient parallel processing and avoiding bottlenecks in communication between cores.

System-on-Chip (SoC) Designs: SoCs integrate various components such as processors, memory, and peripherals onto a single chip. A Hybrid Connected WDT Controller can enhance the communication efficiency among these components, reducing latency and improving overall system throughput.

Heterogeneous Architectures: Connected WDT Controllers are beneficial in systems with heterogeneous architectures, where different types of processing elements, accelerators, and specialized hardware components coexist. By providing adaptive and efficient communication paths, these routers can optimize data transfers between diverse components, enhancing overall system performance.

High-Performance Computing (HPC) Systems: In HPC systems, which involve massive parallelism and data-intensive computations, reducing communication latency is crucial. Hybrid Connected WDT Controllers can facilitate faster data exchanges between nodes, improving overall system throughput and scalability.

Customized Accelerators and Coprocessors: For systems incorporating specialized accelerators or coprocessors, a Hybrid Connected WDT Controller can help in creating efficient communication channels between the main processor and these accelerators.



Design of windowed watch dog timer for embedded system faulty control

Internet of Things (IoT) Devices: In IoT devices where power efficiency is critical, optimizing communication paths can reduce energy consumption. Hybrid Connected WDT Controllers can help in designing low-latency and high-throughput communication architectures for IoT devices, enhancing their overall performance.

Embedded Systems: In embedded systems, especially those requiring real-time processing, minimizing communication delays is essential. Hybrid Connected WDT Controllers can be employed to design efficient communication structures, ensuring timely data transfer and reducing latency.

Data Centres: In large-scale data centres, where numerous servers and networking components need to communicate efficiently, Hybrid Connected WDT Controllers can play a role in optimizing the data flow and minimizing communication bottlenecks.



CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

Watchdog timers have a rich history rooted inside the evolution of embedded systems and their need for reliability and fault tolerance [1]. The idea of a watchdog timer dates to the mid-20th century when the earliest laptop structures had been becoming extra prevalent. These early systems, regularly large and highly priced, were used for numerous commercial and scientific programs [2]. The preliminary task in these systems became making sure their non-stop operation, specifically in important packages where device disasters ought to lead to tremendous outcomes. The idea of a watchdog timer emerged as a solution to this problem.

The idea of a watchdog timer turned into honest at its inception. It involved a timer or counter circuit that needed to be periodically reset with the aid of the software program going for walks on the main processor [3]. If the timer changed into now not reset within a predefined time c program languageperiod, it would trigger a reset signal, efficiently rebooting the system. This primary watchdog mechanism provided a protection net towards software program crashes or hardware disasters.

As microcontrollers and embedded systems became greater generic in various industries [4], the idea of watchdog timers found its way into these compact and specialized computing gadgets. Microcontroller producers began integrating dedicated hardware watchdog timers into their products, making it easier for developers to implement fault tolerance of their embedded packages. Watchdog timers evolved to grow to be greater configurable and flexible. This era noticed the introduction of programmable watchdog timers that allowed builders to set the timeout period in keeping with their unique utility necessities [5]. This configurability was important for adapting watchdog timers to a wide variety of embedded structures, from automobile control units to purchaser electronics. The idea of windowed watchdog timers emerged as a response to the need for even finer manipulate over system reliability [6]. A windowed watchdog timer [7] has two windows: a service window and a frame window. The provider window requires the software program to periodically "pet" the watchdog inside a particular time frame, stopping it from triggering a reset.

The frame window, alternatively, defines a broader interval wherein the watchdog [8] must be serviced as a minimum once.

Design of windowed watch dog timer for embedded system faulty control

Additionally, they provide dynamic initialization alternatives, considering extra sophisticated triggering techniques, which include outside timers or sensors. These innovations make windowed watchdog timers even more adaptable and dependable in a huge variety of applications.

In [10], Nikiema et al. Proposed a method for enhancing the dependability of RISC-V cores in facet computing devices. They centered on robust machine design and testing of their approach. The technique worried rigorous checking out and verification tactics to make sure the reliability of RISC-V cores for edge computing applications. One disadvantage of this work turned into the dearth of specific information concerning the checking out methodologies employed. Ahangari et al. [11] provided an structure tailored for protection-critical transportation structures. Their technique aimed to make sure the fault tolerance and reliability of such systems. They integrated superior protection mechanisms into the architecture to cope with the unique challenges posed by transportation structures. However, the paper did no longer delve into the specific implementation demanding situations or ability barriers of their proposed architecture.

Pagonis et al. [12] explored strategies to increase the fault tolerance of Commercial Off-The-Shelf (COTS) FPGAs in space packages. Their research focused on mitigating Single Event Upsets (SEUs) in Multi-Processor System-on-Chip (MPSoC) configurations. The study supplied treasured insights into SEU mitigation strategies for spaceborne FPGAs. A capacity quandary of this work will be the specific applicability of the strategies to space environments. Mattos et al. (in [13]) utilized the HARV-SoC platform for reliable sensing programs in radiation harsh environments. They evolved a machine-on-chip solution tailor-made for such environments, emphasizing the importance of strong sensing abilities. The observe showcased the sensible utility in their method in difficult conditions. However, it did now not discover capacity demanding situations in scaling the device for broader contexts. Zagan and Găitan [14] implemented an FPGA-based hardware-elevated Real-Time Operating System (RTOS) with a focal point on actual-time occasion managing. Their work aimed to beautify the real-time abilities of embedded systems through hardware acceleration. This technique demonstrated the capability blessings of hardware acceleration in RTOSs. However, the examine did now not address potential trade-offs or scalability troubles in actual-international applications.

Palumbo et al. [15] targeted on improving the detection of Hardware Trojan Horses in microprocessors the use of Hamming Codes. Their research aimed to beautify the safety and reliability of microprocessors by way of detecting malicious hardware alterations. This approach showcased the importance of hardware-degree protection mechanisms. However, it did now not delve into the computational overhead added via the Hamming Codes or capability fake positives.

Design of windowed watch dog timer for embedded system faulty control

However, it did not offer specific answers or strategies for addressing those vulnerabilities. In [19], Akanksha and Kammari designed an improved watchdog timer tailored for MAC (Medium Access Control) packages. Their approach aimed to decorate the reliability and fault tolerance of MAC protocols in wi-fi communicate structures. The research supplied a practical implementation of an stepped forward watchdog timer. However, it did not talk ability boundaries or challenges in deploying the timer in real-world MAC programs. Chaithanya et al. [20] provided the microarchitecture and design of a watchdog timer for a RISC-V-based totally SoC. Their work centered at the specific design concerns and microarchitecture of the watchdog timer. This take a look at verified the feasibility of incorporating a watchdog timer into RISC-V-primarily based SoCs. However, it did no longer delve into capacity alternate-offs in phrases of place or power consumption.



Chapter 3

Existing System

3.1 Introduction

In the realm of modern computing, the demand for faster and more efficient data transmission has propelled the evolution of WDT architectures. Among these, the 2x2 WDT Controller stands as a fundamental building block, orchestrating the intricate dance of data packets across interconnected nodes. In this introduction, we delve into the essence of 2x2 WDT Controllers, unravelling their significance, architecture, and role in shaping the landscape of contemporary computing. The digital era is characterized by an insatiable hunger for speed and connectivity. From streaming high-definition videos to executing complex algorithms, the seamless flow of data underpins the very fabric of our technological existence. At the heart of this digital ecosystem lies the network, a web of interconnected nodes that facilitates the exchange of information. Traditionally, networks have been realized through dedicated wires or buses, but as the scale and complexity of systems grew, these conventional approaches proved inadequate.

Enter the era of network-on-chip, where the principles of networking are seamlessly integrated into the very architecture of microprocessors. WDT represents a paradigm shift in the design of on-chip communication systems, offering scalability, flexibility, and performance that far surpass traditional bus-based designs. At the core of WDT lies the router, a pivotal component responsible for directing data packets between various processing elements. The 2x2 WDT Controller, as its name suggests, operates within a 2-dimensional grid, connecting adjacent nodes along rows and columns. Despite its apparent simplicity, the 2x2 router plays a crucial role in orchestrating the flow of data within the network. Its architecture embodies a delicate balance between efficiency and complexity, optimizing resource utilization while minimizing latency and power consumption.

To comprehend the inner workings of a 2x2 WDT Controller, it is essential to dissect its architecture layer by layer. At its core lies the routing logic, a sophisticated mechanism that determines the path traversed by each data packet. Unlike traditional routers, which rely on static routing tables, WDT Controllers employ dynamic routing algorithms to adapt to changing network conditions. This dynamicity enables WDT Controllers to circumvent congested paths and optimize performance in real-time. Adjacent to the routing logic lies the switching fabric, a network of crossbars or switches that facilitate the interconnection of input and output ports.



Furthermore, the 2x2 WDT Controller incorporates various mechanisms for flow control and congestion management. Flow control protocols regulate the rate of data transmission, preventing buffer overflow and ensuring smooth operation under varying traffic loads. Congestion management algorithms dynamically adjust routing decisions to alleviate congestion hotspots and maintain optimal network performance. In addition to its functional components, the 2x2 WDT Controller embodies several design considerations aimed at maximizing efficiency and scalability. Through techniques such as pipelining and parallelism, WDT Controllers exploit parallelism at various levels to enhance throughput and reduce latency. Moreover, advancements in semiconductor technology have enabled the integration of multiple routers within a single chip, paving the way for highly interconnected and scalable WDT architectures.

3.2 Existing WDT Controller

Begin by outlining the specifications and requirements for the 2x2 WDT Controller design. This includes factors such as throughput, latency, power consumption, and area constraints. Define the objectives of the router, considering the specific application or system it will be integrated into.

Step 1: Architecture Initialization: Determine the overall architecture of the 2x2 WDT Controller, considering factors such as the number of input/output ports, routing algorithms, buffering strategies, and control mechanisms. Choose between centralized or decentralized control architectures based on the system requirements and design goals.

Step 2: Input and Output Port Configuration: Define the configuration of input and output ports for the 2x2 router. Each port should be designed to handle incoming and outgoing data packets efficiently. Establish protocols for data transmission and reception at the input and output ports, ensuring compatibility with the circuit switching protocol.

Step 3: Switching Fabric Design: Design the switching fabric, which forms the core of the router and facilitates the routing of data packets between input and output ports. Determine the routing algorithm to be used for circuit switching, considering factors such as path selection, congestion management, and fault tolerance.

Step 4: Buffering and Flow Control: Implement buffering mechanisms to temporarily store incoming data packets during transit through the router. Design flow control mechanisms to regulate the flow of data packets within the router, preventing congestion and ensuring smooth transmission.

Step 5: Control Logic Implementation: Develop the control logic for the router, including state machines, arbitration logic, and signalling protocols. Implement mechanisms for packet scheduling, routing table management, and error handling to ensure reliable operation.

Design of windowed watch dog timer for embedded system faulty control

Develop the control logic for the router, including state machines, arbitration logic, and signalling protocols. Implement mechanisms for packet scheduling, routing table management, and error handling to ensure reliable operation. Conduct thorough testing and verification of the 2x2 WDT Controller design using simulation tools and hardware emulation platforms. Verify the functionality of the router under various operating conditions and traffic scenarios, ensuring that it meets the specified requirements

3.3 Drawbacks

Limited Bandwidth: A 2x2 WDT Controller inherently has limited bandwidth since it only supports two input and two output channels. This can become a bottleneck when dealing with a large number of cores or when the communication demands are high.

Scalability Issues: As the number of cores or components in the system increases, the 2x2 router may not scale efficiently. This can result in increased contention for the limited resources and a potential degradation in performance.

High Latency: Circuit switching, while providing dedicated paths for communication, can introduce higher latency compared to other switching techniques like packet switching. This latency can impact the overall performance of the system, especially in applications where low-latency communication is critical.

Complex Routing Algorithms: Implementing efficient routing algorithms in a 2x2 WDT Controller with circuit switching can be challenging. As the network scales, the complexity of the routing algorithms may increase, leading to higher power consumption and potential performance degradation.

Increased Hardware Overhead: Circuit switching requires additional hardware components, such as buffers and control logic, to establish and maintain dedicated communication paths. This increases the overall hardware overhead and may impact the chip area and power consumption.

Fixed Path Establishment: Circuit switching establishes fixed paths for communication, which might not be suitable for dynamic and varying communication patterns. In applications where communication patterns change frequently, the fixed paths may not be optimal, leading to suboptimal performance.



Fault Tolerance: Circuit switching may face challenges in handling faults or failures in the network. If a link or a router fails, the dedicated circuit may need to be reestablished, causing interruptions and potential performance degradation.

Complex Design: The design and implementation of a circuit-switched 2x2 WDT Controller can be more complex compared to simpler switching protocols. This complexity can increase the risk of design errors and make it more challenging to meet performance and power efficiency goals.

3.4 Summary

The evolution of WDT architectures has revolutionized modern computing, with the 2x2 WDT Controller standing as a cornerstone in facilitating seamless data transmission across interconnected nodes. WDT Controllers integrate networking principles into microprocessor architecture, offering scalability and performance far surpassing traditional bus-based designs. At the heart of WDT lies the router, orchestrating data flow between processing elements. The 2x2 WDT Controller operates within a 2-dimensional grid, connecting adjacent nodes along rows and columns, balancing efficiency and complexity to optimize resource utilization and minimize latency and power consumption. Dissecting its architecture reveals routing logic, dynamic routing algorithms adapting to network conditions, and switching fabric enabling interconnection. Flow control mechanisms regulate data transmission rate, while congestion management algorithms dynamically adjust routing to maintain performance. However, the 2x2 WDT Controller with a circuit switching protocol faces drawbacks including limited bandwidth, scalability issues, high latency, complex routing algorithms, increased hardware overhead, fixed path establishment, fault tolerance challenges, and complex design, challenging its efficiency in handling dynamic communication patterns and faults.



Chapter 4

Proposed System

4.1 Introduction

A watchdog timer is a crucial component in embedded systems and computer systems designed to enhance reliability by monitoring the system's operation and taking corrective actions when necessary. Its primary purpose is to detect and recover from malfunctions that might cause a system to become unresponsive or fail. The watchdog timer operates as a hardware-based countdown timer integrated into the system's microcontroller or processor. It must be regularly reset or "fed" by the software running on the system. If the timer is not reset within a predefined time interval, it assumes that the system has malfunctioned or stalled, triggering a corrective action.

The way a watchdog timer functions is straightforward yet effective. When the system initializes or during normal operation, software periodically resets the watchdog timer, typically by writing a specific value to a designated hardware register. This action "feeds" the watchdog, preventing it from reaching zero and triggering a timeout. If, for any reason, the software fails to reset the watchdog timer within the specified time period (known as the timeout period), the watchdog timer will expire. Upon expiration, the watchdog generates a reset signal or other predefined actions. This reset signal can force a system reset, reboot the system, or execute a predefined recovery routine to restore the system's normal operation.

The watchdog timer is commonly used in scenarios where system failures could lead to critical consequences, such as in industrial control systems, automotive electronics, medical devices, and mission-critical computing applications. By continuously monitoring the system's health and automatically initiating recovery actions, the watchdog timer helps prevent system crashes, hangs, or other failures that could compromise safety, security, or operational continuity.

Furthermore, watchdog timers are configurable to suit different application requirements.

4.2 Proposed Methodology

This section gives a detailed evaluation of the proposed watchdog timer device model. Figure 1 offers an in-intensity depiction of the purposeful block diagram of the proposed watchdog timer, dropping light on its internal structure and operation. This watchdog timer is specifically designed for FPGA implementation, where unique timing manipulates, and fault tolerance are critical factors

Design of windowed watch dog timer for embedded system faulty control

At its middle, the watchdog timer is synchronized with the aid of the SYSCLK input, that's notably impartial of the processor's clock. This independence guarantees that the watchdog operates autonomously and reliably, no matter capacity versions in the processor's clock pace. One great factor of this layout is the predefined sets of window lengths, that are decided primarily based at the precise utility necessities. These predefined values play a vital function in organising the timing windows for watchdog tracking. Importantly, these window lengths may be decided on with the aid of writing to corresponding bits in the configuration sign in: SWLEN for the service window and FWLEN for the frame window. These values are adjusted after energy-as much as cater to the precise wishes of the embedded machine.

A key feature of this watchdog timer layout is the automated locking mechanism. Once the window lengths are configured after strength-up, the corresponding configuration fields turn out to be locked. In other phrases, any tries to modify these settings are intentionally disabled to save you accidental modifications. To change the window lengths in instances where it turns into necessary, a 16-bit unencumber check in is incorporated into the design. To initiate the amendment technique, software should carry out successive writes to this free up sign up with specific statistics patterns (0xAAAA and 0x5555). Notably, the second one writes ought to arise inside a particular 10 μ s window following the first write. If this timing requirement isn't always strictly met, the configuration fields' write get entry to stay disabled, ensuring the integrity of the watchdog timer's settings.

The watchdog timer's operation is initiated while an excessive-to-low transition is detected at the INIT sign. This signal serves because the cause for the graduation of the service window. Crucially, the service window operates the use of a derived clock, denoted as SWCLK, that's extensively slower than the SYSCLK.

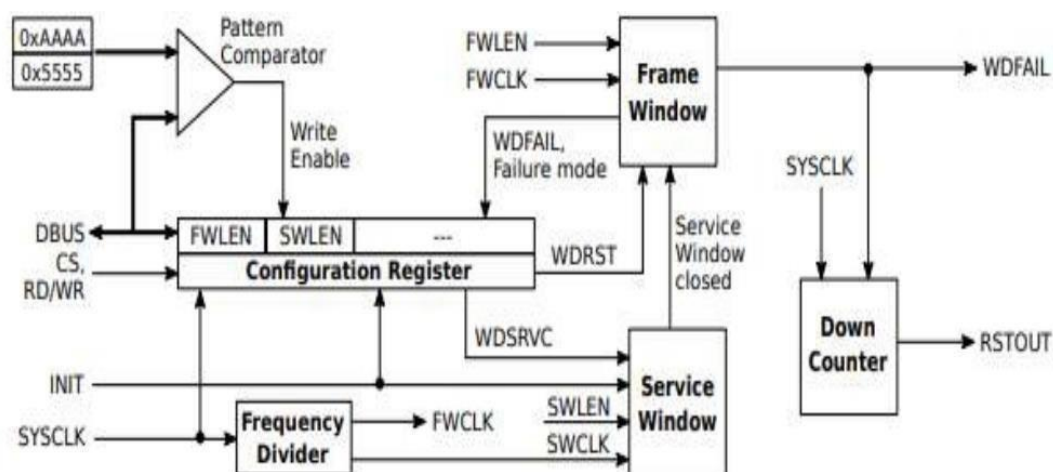


Figure 4.1. Proposed watchdog timer operational diagram.

Design of windowed watch dog timer for embedded system faulty control

The operational status of the service window is regularly updated in the watchdog's configuration register. When the watchdog is correctly serviced within the service window, the counters immediately halt, and the frame window phase begins. The frame window follows a similar mechanism to the service window but operates using a separate derived clock, referred to as FWCLK. This clock is also slower than the SYSCLK, contributing to resource efficiency. The frame window includes an offset up/down counter, akin to the one in the service window, and a main counter with functionalities resembling those in the service window. The offset-up counter calculates the offset between the termination of the service window and the subsequent rising edge of the FWCLK. This ensures precise synchronization between the two-timing windows. The main counter counts for (FWLEN - 1) times, after which the offset down counter takes over to complete the frame window's timing control. A critical aspect of this watchdog timer's functionality is its reset initialization and fault detection logic, which is depicted in a finite state machine (FSM) diagram in Figure 4.2.

Upon initializing, the WDFAIL output is asserted, indicating an initial watchdog failure state. To initiate the watchdog timer's operation, a rising edge on the WDRST bit is required. This rising edge prepares the watchdog timer for initialization. Once the service window opens, a rising edge on the WDSRVC bit is crucial for deasserting the WDFAIL output and commencing the window counters' operation. However, if the watchdog is serviced incorrectly within the service window, the entire initialization process is discarded. In such cases, the software must repeat the entire procedure to ensure proper watchdog operation. The WDFAIL signal is deasserted only when the watchdog is successfully initialized and enters its operational phase.

The length of this counter may be tailor-made based totally on the want for storing debug data or appearing other duties at some stage in the fault recovery system. Upon the counter's expiration, the watchdog timer asserts its RSTOUT output excessive, signifying the want for a device reset. It's important to observe that the reset counter remains nonfunctional throughout the strength-up section, and the RSTOUT output is first set to a low kingdom. The automated enablement of the reset counter happens handiest when the watchdog is initialized for the first time.

4.3 Fault Detection Operation in Watchdog Timer

Figure 2 gives an FSM design that encapsulates the critical functionalities of the watchdog timer's reset initialization and fault detection logic. This FSM plays an essential function in making sure the right functioning and reliability of the watchdog timer. Upon start, the WDFAIL output is asserted, indicating an initial watchdog failure condition.

Design of windowed watch dog timer for embedded system faulty control

As the service window opens, a growing area at the WDSRVC bit takes place, serving as a pivotal event inside the watchdog's operation. This event triggers several essential moves within the FSM. Firstly, it asserts the WDFAIL output, indicating that the watchdog timer is now equipped for operation. Additionally, the window counters in the watchdog timer are activated, initiating the counting processes that govern the service and frame windows. It is vital to observe that incorrect servicing of the watchdog timer during this phase results in the whole initialization process being discarded. In such instances, the software program must repeat the entire procedure to make sure proper watchdog capability. The WDFAIL sign stays asserted until the watchdog timer is successfully initialized, serving as a dependable indicator of its fame. This suggests a failure or anomaly inside the gadget, and the watchdog configuration sign in is promptly updated to document the failure's nature and status.

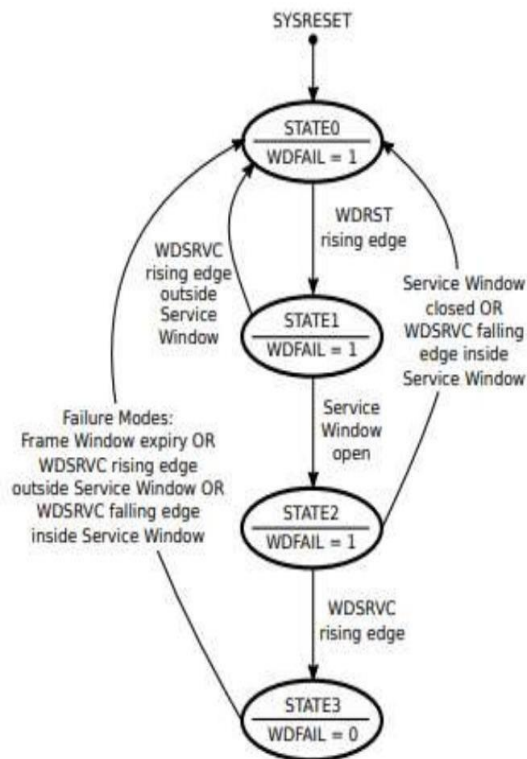


Figure 4.2. Fault detection logic using FSM.

Furthermore, the statement of the WDFAIL signal initiates a reset counter inside the watchdog timer. The reason of this counter is to run for a predefined period, which may be determined based on the need to save debugging information or carry out recuperation actions.

Upon the counter's expiration, the watchdog timer asserts its RSTOUT output to a high state. This action means that a gadget reset has been triggered in reaction to the watchdog failure. Notably, during the energy-up phase, the reset counter remains nonfunctional, and the RSTOUT output is about to a low state.

It is only when the watchdog is initialized for the primary time that the counter will become routinely enabled, considering managed device resets in reaction to RSTOUT change.

4.4 Watchdog Timer Interface

Two important bit fields within the configuration register are WDRST and WDSRVC. The WDRST field is liable for resetting the watchdog timer when vital, making sure that it keeps featuring successfully. Conversely, the WDSRVC subject is utilized for servicing the watchdog, efficaciously preventing it from triggering a timeout event. It additionally illustrates the automated updating of the INIT input and the WDFAIL output in the configuration sign in. This computerized replace mechanism guarantees that the watchdog timer stays synchronized with the device's kingdom. Furthermore, the SWSTAT subject within the configuration register holds the present timescale of the provider window, at the same time as the FLSTAT field is dedicated to logging any watchdog failure modes, supplying vital diagnostic records within the occasion of a failure.

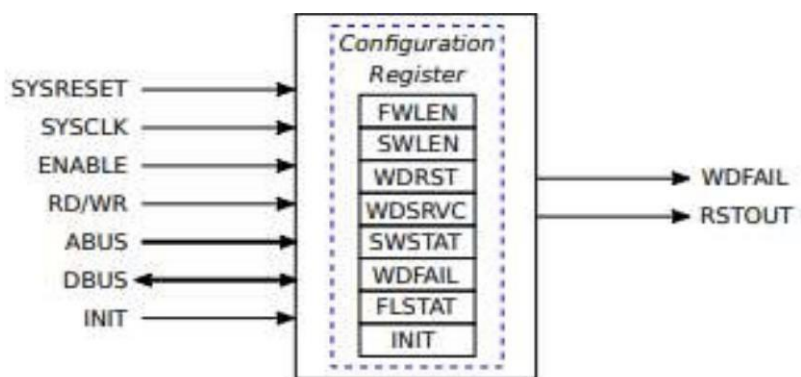


Figure 4.3. Interfacing watch dog timer with multiple input-outputs.

Control inputs to the watchdog timer encompass ENABLE and RD/WR, which facilitate each analysing from and writing to the configuration sign up, enabling dynamic adjustments to its settings. The ABUS (address bus) and DBUS (data bus) alerts, also visible inside the discern, represent the critical pathways for communicate within the watchdog timer. The ABUS is chargeable for sporting cope with data, specifying the area in memory to examine from or write to, even as the DBUS consists of the real data being transferred. The proposed watchdog timer layout introduces the idea of a service window and a frame window.



Design of windowed watch dog timer for embedded system faulty control

The service window's duration is considerably shorter than that of the frame window. These window periods may be adjusted with the aid of software program after the machine's electricity-up by means of enhancing the SWLEN and FWLEN bit fields in the configuration check in. Importantly, after the initial configuration, any attempts to modify these values are intentionally disabled. To make changes, a stringent unlock procedure must be followed, preventing accidental alterations that could lead to system instability. The frame window defines the periodicity at which the watchdog should be serviced. Typically, its period is slightly longer than the principal loop's execution time in the embedded control system.

This guarantees that the watchdog is serviced as a minimum once during every cycle of the system's operation, contributing to its reliability. The technique of driving the INIT sign to the watchdog timer can range. One method entail triggering the INIT signal on the end of the main loop, following the finishing touch of positive sanity exams. Alternatively, an external c language timer can be employed to generate the INIT sign, getting rid of the want for direct processor intervention.



CHAPTER- 5

VLSI DESIGN

5.1 Introduction

The hardware sector has experienced phenomenal growth during the last two decades, primarily owing to the rapid developments in entering technologies, an immense scale system plan-to put it clearly, because of VLSI 's strategy. The number of utilizations of the integrated circuits in the upper register, transmitted communications and purchaser devices has slowly and quickly grown. Regularly, the requisite computational power (or, at the ends of the day, knowledge) of these applications is the key trigger for the rapid development in this area. Provides a snapshot of the unmistakable trends in computer developments across the past several decades. New developments in the driving edge (e.g. low-speed video and phone correspondence) also provide end-customers a precise measure of managing power and versatility.

VLSI has been around for quite a time, there really is nothing new about it, but as a result to developments in the field to PCs, there has been a remarkable proliferation of devices that can be used to design VLSI circuits. Nearby, in compliance with Moore 's rule, the ability of the IC has improved exponentially over the years in terms of measurement efficiency, usage of available land, yield. The cumulative effect between these two advancements were that people will now be willing to put various applications of the IC's, opening up new wildernesses. Models are embedded structures, where intelligent gadgets are put within everyday items, and uniform figuring where little computing gadgets accumulate to such an extent that even the shoes you carry will potentially perform something useful, such as testing the pulses.

Verilog was developed while fashion designers were looking for instruments to pursue varying degrees of reenactment. By the mid-1980s, there were switch-level test systems, door-level test systems, practical test systems (regularly specifically built for programming) with no simple way to enter them. However, the larger the wall, the modern programming dialects themselves were / are fundamentally sequential and hence \"semantically confused\" when showing the simultaneity of computerized hardware.



Design of windowed watch dog timer for embedded system faulty control

Research on the IEEE specification (IEEE-1364) began in 1992 and the final draft was authorized in December 1995. As a consequence, Verilog has become a global norm-which would further extend its market development and use. At present, there is a noble trend to extend Verilog beyond clearly sophisticated circuits.

Although Verilog grew from developments in privately held companies, its main adversary emerged in the American Guard Division (DoD). Throughout 1981, the DoD sponsored the laboratory on hardware representing dialects as part of the Exceptionally Fast Coordinated Circuits (VHSIC) initiative and, throughout 1983, the outcome developed the VHSIC Computer Portrait Language (VHDL) project. There is, of necessity, a query as to which language is stronger. Indeed, this, clearly, is a hard problem to address without triggering the resources and answers of encouraging the separation of the less-favoured words. However, the accompanying focuses included in the ongoing discussion at the VHDL and Verilog news meetings.

The fundamental aspect is the structural nature of the language-since Verilog relies on C and VHDL relies on ADA. Verilog is simpler to understand since C is a much less complicated language. It also creates an increasingly minimal code: simpler to compose as well as peruse. For fact, the large majority of experts who certainly know C (as opposed to people who know ADA) makes studying and planning faster. VHDL is unique and encourages users to classify their own forms, given the reality that the principal styles chosen are either the basic categories of the language itself or those defined by the IEEE. The positive being that the type-check being conducted by a programmer who will the the blunders; the downside would be that the form-checking has to be achieved confidently.

Verilog has two simple priorities over VHDL It makes a switch-level demonstration that a few architects consider useful for the analysis of new circuits. It ensures that certain signs are presented to \"unknown\" which assure that certain architects will create a basic rationale for indicating their sign-the foundation sorts in VHDL will be zero and the \"hasty\" architect may exclude a worldwide reset.

5.2 ASICS

Innovations in IC assembly have helped us to create quick and efficient circuits in moderate and minimized PCs. This also means that we can cram a ton of highlight into a specific area

5.3 FPGA

Completely obtained FPGA chips with two hundred or considerably more rational entryways, with configurable interconnections, are open to shoppers for their specific programming equipment to recognize the necessary usefulness. This configuration style gives way to rapid prototyping but rather, furthermore, to financially sound chip design, especially for low-volume applications. A standard configurable field door display (FPGA) chip involves I / O support, progression of configurable rationale squares (CLBs) and circuit board structures. Interconnect preparation is completed by handling Slam cells for whom the output terminals are attached to the inputs of the integrated circuits of the MOS pass. The FPGA General Strategy. An increasingly accurate picture of where the switch lattices used for interconnecting are located.

A fundamental CLB (XC2000 platform from XILINX) Which comprises four information signal terminals (A, B , C , D), a clock signal terminal, a device programmable multiplexer, an SR-hook and a look-in table (LUT). The LUT is a physical document containing the algebraic Reality List function\\|. In this way, it may generate either power up to four variables, or even two of three specific component capacities.

CLB is built in such a way that the range may be tailored to provide any specific logical power. Further development of CLBs to display dynamic capacities was also carried out. The standard structure stream of the FPGA chip closes with the utilitarian meaning of its highlights, using a firm information jargon such as VHDL. Mixed engineering is then designed (or divided) into circuits or logical cells utilizing the technique. The chip plan at this stage is completely depicted in terms of accessible rational cells. Next, the area and steering stage assigns singular framework cells to the FPGA Locals (CLBs) and chooses examples of directing between mobiles as per the net recap. Its on-chip is completed in the wake of the steering.

The most leeway of the FPGA-based plan was its brief pivot time, e.g. the time required from the earliest starting point of the plan procedure until there is a useful chip. Because the alteration of the FPGA chip does not include a physical assembly level, a working model can be acquired almost anytime a new invention is expected.

The door monitor (GA) appears after the FPGA, despite the prototyping capabilities of the snappy. Although the client coding is used to structure the FPGA chip, that of the entrance exhibition is finish

Design of windowed watch dog timer for embedded system faulty control

The traditional cell-based framework is one of the most common, specially design types that require a complete custom cover collection to be made. The general entity is the Commonly Called Police Agency. All commonly used rational cells are created, described and stored in a standard cell library, in this plan style. The regular library can include two to three hundred cells, like inverters, NAND gateways, NOR doors, AOI buildings, OAI gateways, D-locks, and flip-flops. Can kind of door can have a few executions to give the various ventilators an adequate capacity to move. For example, the inverter input may include normal size diodes, double-size transistors and four-size transistors such that the chip designer may select the correct size to reach high circuit speed and design thickness. Each cell representation is performed for a wide range of classes. It's built out of

- Wait period versus loading power
- Virtual format of the circuit
- Length of the process evaluation period
- Replication of a system deficit
- Details regarding the Location and-Course cell
- Cover with details

In order to facilitate the controlled cell placement and regulation of the communications between the cells, each telecommunications layout is built at a fixed height to such an degree that the separate cells may be positioned next to each other to frame the columns. In even the most portion, the power and ground rails travel relating to the top and bottom edges of the cell and hence the neighboring cells exchange common energy and ground transport. Input and output pins are positioned at the upper and lower ends of the cell. Common today's cell model. Note that the nMOS semiconductors are closest to the ground rail while the pMOS transistors are closest to a track.

5.4 VERILOG

Verilog HDL is an Equipment Request Word (HDL). The Device Description Language is a phrase used to describe a specific structure, such as a gadget or part of a computer. One could describe a staggered advanced program. For example, the HDL may depict the course of action of wires, resistors and transistors on the Coordinated Circuit (IC) board, e.g. the degree of the switch, or it may.



Design of windowed watch dog timer for embedded system faulty control

. For example, the HDL may depict the course of action of wires, resistors and transistors on the Coordinated Circuit (IC) board, e.g. the degree of the switch, or it may depict coherent doors and flip lemon on the PC, e.g. the door stage. The registers and the development of data vectors between registers are characterized by a significantly deeper level.

In 1985, the Passage Data System Association, currently an entity from the Rhythm Data Frameworks Gathering Frameworks, Inc., propelled Verilog. Before May 1990, Verilog HDL was the exclusive language of Rhythm with the establishment of Open Verilog Worldwide (OVI). Rhythm was buoyed to extend the language to open space, in the hope that, with the more widespread usage of the software, the demand in Verilog HDL-related programming would grow much more rapidly. Rhythm noted that Verilog HDL's purchasers were required to acquire terminology and execute Verilog-upheld design procedures in a variety of other programming and market firms.

Verilog HDL empowers the system designer to define models to a significant degree of contemplation, e.g. specific or social degree and lower implementation speeds (i.e. door and turn stage) culminating in Tremendous Variety Joining (VLSI) Organized Circuits (IC) systems and chip manufacturing. The main application of HDLs is the concept to show before the author has to agree to the assembly.

There are two standard display dialects currently available in the business, VHDL and Verilog. In fact, the difficulty of the ASIC and FPGA models has contributed to the growth of the number of experienced hardware experts with specific programming and their own large-scale and super-cell libraries written in either VHDL or Verilog. As a consequence, it is important for fashioners to know both VHDL and Verilog, and for EDA programming merchants to have assets that offer the air that empowers the two dialects to be used immediately. For example, a planner may have an idea of a VHDL-composed PCI transport interface, but may decide to remember it for a Verilog-composed full-scale design. HDLs have numerous advantages over the regular idea of subordinate schematics.

Proposals may be spoken to use HDLs at a relatively abstract basis. Architects can render their concept of RTL without specifying a particular assembly procedure.

Design of windowed watch dog timer for embedded system faulty control

At a time when another innovation is developing, planners do not need to reshape their circuit. They are essentially embedding the RTL description in the framework blending strategy and using the most recent manufacturing advances to create a further net overview of the entry - level positions. The rationale union technique for the most recent advancements should benefit the business and positioning circuit.

The door level schematics for highly complicated systems are virtually infinite. The principle of HDL is here to stay.

The architecture of HDL is equivalent to the programming of the Computer. A less complicated solution to designing and checking circuits is a composite description with remarks. In comparison to the entry level schematics, this also provides a short image of the architecture. The door level schematics for highly complicated systems are virtually infinite.

The principle of HDL is here to stay. As computerized circuit intricacy develops exponentially and progressively advanced EDA gadgets, HDLs are currently the overarching approach for expansive PC plans. No mechanized circuit design will proceed to refuse HDL commitment.

Verilog HDL has grown as a standard equipment description term. Verilog HDL provides a range of feasible highlights

Verilog HDL is an easy-to-understand and easy-to-use, commonly available interface specification resource. The form of the sentence is distinct from the programming language C. Architects with C design experience will find Verilog HDL simple to learn.

Verilog HDL empowers the convergence of various thinking levels within a common paradigm. A designer that, along such lines, represent a model of equipment as well as doors, screens, RTLs or codes of behavior are concerned. A developer wants to learn just one vocabulary on a daily basis in terms of trigger and specific levels architecture.



- Verilog HDL incorporates the most commonly known logical amalgamation programming. This enables the terminology of nature to be utilized by professional engineers.
- Both development traders are offering Verilog HDL libraries the ability to re-enact a postlogic mix. Consequently, the architecture of the chip in Verilog HDL takes into account the most comprehensive commercial tendency.
- The programming language interface (PLI) is a ground-breaking highlight that empowers the programmer to write a custom C code to communicate with Verilog \ 's internal knowledge frameworks. Using the PLI, software engineers can develop the Verilog HDL simulator to suit their requirements.
- Electronic speed and advancement of the circuit developed quickly. Fashioners have replied by carrying out deliberation at more substantial rates. Architects just need to think about the highlights. The EDA methods work with inputs into the implementation.
- As of now, the most widely-recognized example is to construct RTL-level HDLs, as reasoning union devices will create RTL-level entryway netlists. Social replication enabled architects to render measurements and conduct circuits legitimately, and then to use EDA programming to execute analysis and advancement at each point of the system. Despite the replication of conduct, wide recognition has not been achieved. RTL engineering is still inconceivably regular today. Today, Verilog HDL is continually being developed in order to conform with the laws of existing research methods.
- Strategies for organized testing and proclamation have emerged. Deliberate testing presents efficient numerical techniques for checking the accuracy of Verilog HDL definitions and for maintaining the sameness between RTL and netlists at the point of entry.
- Present-day testing dialects have also earned easy acknowledgment. The parallelism and equipment structures of the HDLs were aligned with the article plan of C++. These dialects further enhance the creation, verification and inclusion of programmed upgrades. Such dialects don't substitute Verilog HDL anyway. They only boost the productivity of the test process. In addition, the architecture can be alluded to by Verilog HDL.

Design of windowed watch dog timer for embedded system faulty control

- The entry-level netlist provided by the justification combination software is not suitable for fast and timing-basic circuits such as chip. All of these times, architects have often honestly integrated meaning at the entry point of the RTL description to establish perfect execution.

5.5 Field Programmable Gate Arrays Field

Prior to the introduction of the programmable rationale, design justification circuits were focused on board level using common parts or door level in exorbitantly organized (normal) application-explicit circuits. FPGA is an integrated circuit comprising a few relative logical cells (64 to more than 10,000) that can be regarded as ordinary components. Each rational cell can autonomously take on any one of a few personalities. The connection and programmable switch device interconnect the individual cells.

Likewise, with microchips, FPGAs conceivably automatically charge or boot themselves from a special, volatile memory stick. On the other hand, Straightforward 3 age FPGAs may be accessed or arranged by an external \smart user,\, e.g. a microchip, DSP processor, microcontroller, Cpu, or board analyzer, proportional to the chip periphery. In any case, the information way of the gadget is either sequential to release pin details or byte-wide for ideal execution or for less difficult interfaces to pr.FPGA is an island-style FPGA range, consisting of a mixture of steering channels and logical emphasis. Two I / O cushions fall into a single line area or segment distance. Both steering channels have a common diameter (wire tally).Each loop must be scheduled in the smallest FPGA square that it may require. First of all, a circuit containing 14 rational squares and 10 I / O cushions will be scheduled on an FPGA consisting of a 4x4 rational square display. Recall that, in this FPGA model, three out of twenty test circuits used in the FPGA problem (bigkey, des, and dsip) are limited.



CHAPTER 6

SOFTWARE TOOLS

6.1 Introduction

The key tools used in this venture can be separated into two groups.

- Hardware requirement
- Software requirement

6.2 Hardware Requirements

- FPGA KIT

A server address in which Xilinx ISE 10.1i software can be easily controlled is needed in the hardware part, i.e. with a throughput such of Pentium III, 1GB of RAM, 20GB of storage device.

6.3 Software Requirements

- XILINX 13.2

This includes the Xilinx ISE 13.2 software upgrade, where the Verilog code could be used to execute the specification.

6.4 Introduction to XILINX ISE

This device is useful in the preparation, execution, re-enactment and integration of Verilog illustrations for use on FPGA chips.

ISE: Integrated Environment of Software

affirmative Change ecosystem and tentative set-up of FPGA or CPLD-centric electronic systems

- POLY centralized assembly of GUI dynamic systems
- A shrewd mix system focused (XST: Xilinx Synthesis Technology)
- XST provides various lingos:

Design of windowed watch dog timer for embedded system faulty control

- Verilog
- VHDL
- XST centralizes a database overall view of criteria
- No Supports each one of the methods required to complete the agreement:
- Link Translate, map, location and path
- Apply a bit of time for playback

In this case, Verilog can be used to build a test seat to validate the survivability of the model using documentation on the host Computer to depict barrels, to interact with the user and to differentiate results from conventional ones.

A Verilog indicates that this is transformed to "entrances and cables" which are mapped to a configurable contraction, e.g. a CPLD or FPGA, and then the actual equipment is highlighted, instead of the Verilog code being "executed" but it's on a main processor or some other kind of.

6.4.1 IMPLEMENTATION

- [XST] Synthesis
- Convert (NGDB Construction)
- Converts all netlists of input designs and then translates
- The outcomes into a single unified file explaining reasoning and limitations.
- [MAP] Mapping
- Maps the computer function logic.
- Take a netlist and group the logical components into CLBs and IOBs (FPGA components).
- Position and road (PAR)
- Position and bind FPGA cells.
- Generation of bit streams

6.5 XILINX DESIGN PROCESS

Step 1: Design entry

- HDL (Verilog or VHDL, ABEL x CPLD), Graphical sketches, Bubble diagram

Phase 2: Synthesis:

- affirmative Translates.v,.vhd,.sch documents to a netlist record (.ngc)
- Step 3:Implementation
- FPGA: Translate / Map / Place and Route, CPLD: Fitter
- Step 4: Configuration / Programing
- No Add a BIT report to a FPGA
- Apply the JEDEC system document to the CPLD
- Attach system MCS paper to Flash PROM
- Rehabilitation will take place during stage 1 , 2 , 3

The frameworks used as part of this method is XILINX ISE 13.2 of replication and propagation. The proposals are written up in the verilog language.

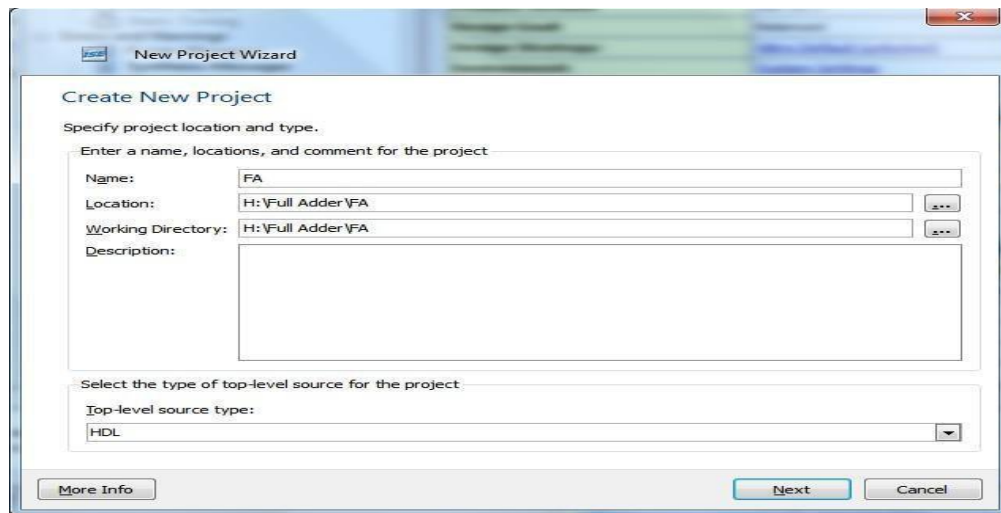
6.6 XILINX SOFTWARE

Use Xilinx Field Programmable Gate Array (FPGA) or Complex Programmable Logic Design (CPLD), Xilinx Software is indeed a series of programming devices used during updated computerized circuitry architecture. The architecture method comprises of: (a) encoding of the layout; (b) incorporating and implementing the design; (c) useful recreation; and (d) analyzing and evaluating. Using above-mentioned CAD tools, computerized descriptions may be inserted in a number of ways: and use a schematic segment template, using a system description dialect (HDL) – Verilog or VHDL or a combination of both. Also the root of the software that includes the use of Verilog HDL may be considered for this supposition.

Users could even launch Xilinx Tools mostly on information systems by clicking the Project Navigator button.

It will open the Project Navigator tab on the computer. Exhibits this frame (see Figure 1) The final step you've hit

Choose File->New Project to create a new project. A latest project window (Figure 2) should be drawn up on the desktop. Taking the requisite submissions out as follows:



Create New Project

Specify project location and type.

Enter a name, locations, and comment for the project

Name: FA

Location: H:\Full Adder\FA

Working Directory: H:\Full Adder\FA

Description:

Select the type of top-level source for the project:

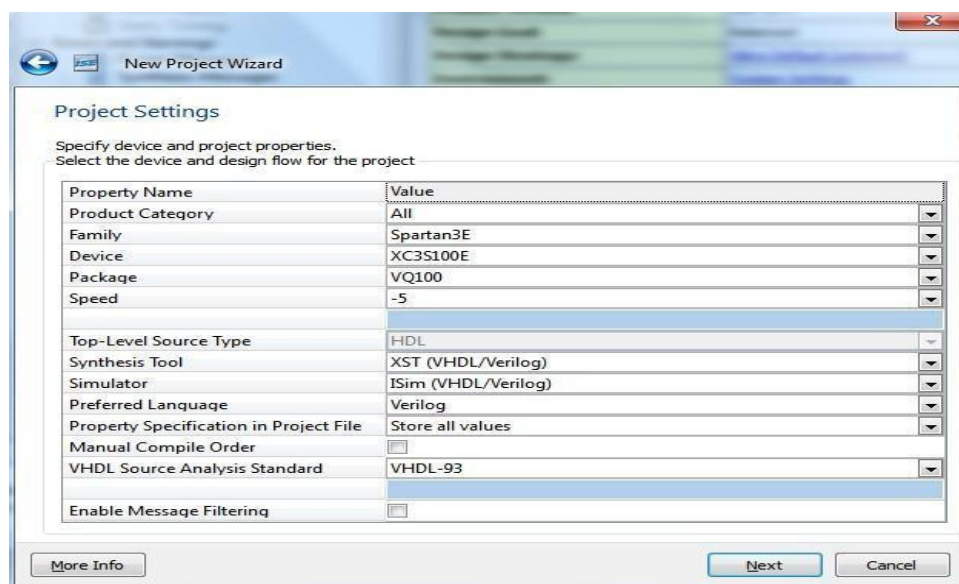
Top-level source type: HDL

More Info Next Cancel

Name of Project: Type the name of your latest project

Project Position: The folder where you'd like latest project to also be located (Note: do NOT define the design of the site as a Screen folder or folder in the Xilinx\bin directory. The H: drive is the safest way to put it. The location path of the project is NOT to include any gaps in it, e.g. H:\Full Adder\F A and Not to be used).

Clicking on the NEXT button would open the section below:



Project Settings

Specify device and project properties.

Select the device and design flow for the project

Property Name	Value
Product Category	All
Family	Spartan3E
Device	XC3S100E
Package	VQ100
Speed	-5
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

More Info Next Cancel

Click mostly on 'value' field with each of the coefficients of allocation below, and select from the set of attributes that appear.

Design of windowed watch dog timer for embedded system faulty control

Community Device: FPGA / CPLD community used. In this blog entry, we must use the Spartan3E FPGA.

Device: Actual amount of the machine. You should join XC3S100E for this laboratory (which can be included in the attached sensor chip)

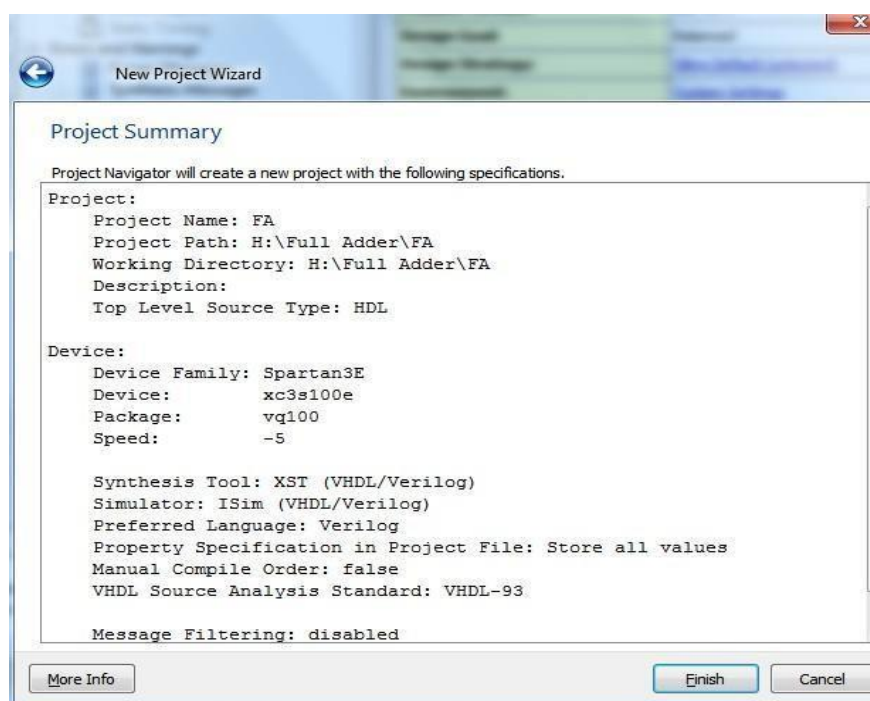
Package: The package is ordered with pin number. The Spartan FPGA used in this research lab is bundled in the VQ100 package.

Speed Grade: the speed limit is "-5."

Synthesis Device: XST [VHDL / Verilog]

Simulator: Device used to model and validate the implementation of technologies. Modelsim simulation software is embedded into the Xilinx ISE. Use "Modelsim-XE Verilog" because it is easy to be using the simulation or also the Xilinx ISE Simulators.

Then click the NEXT button to transfer this same inputs.

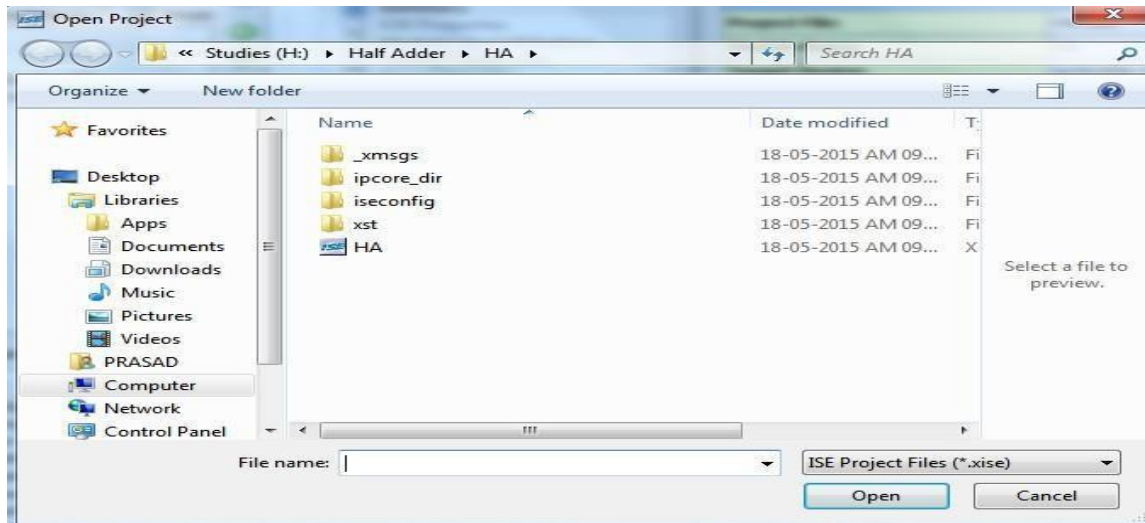


To access a project overview tab, press the Start button.

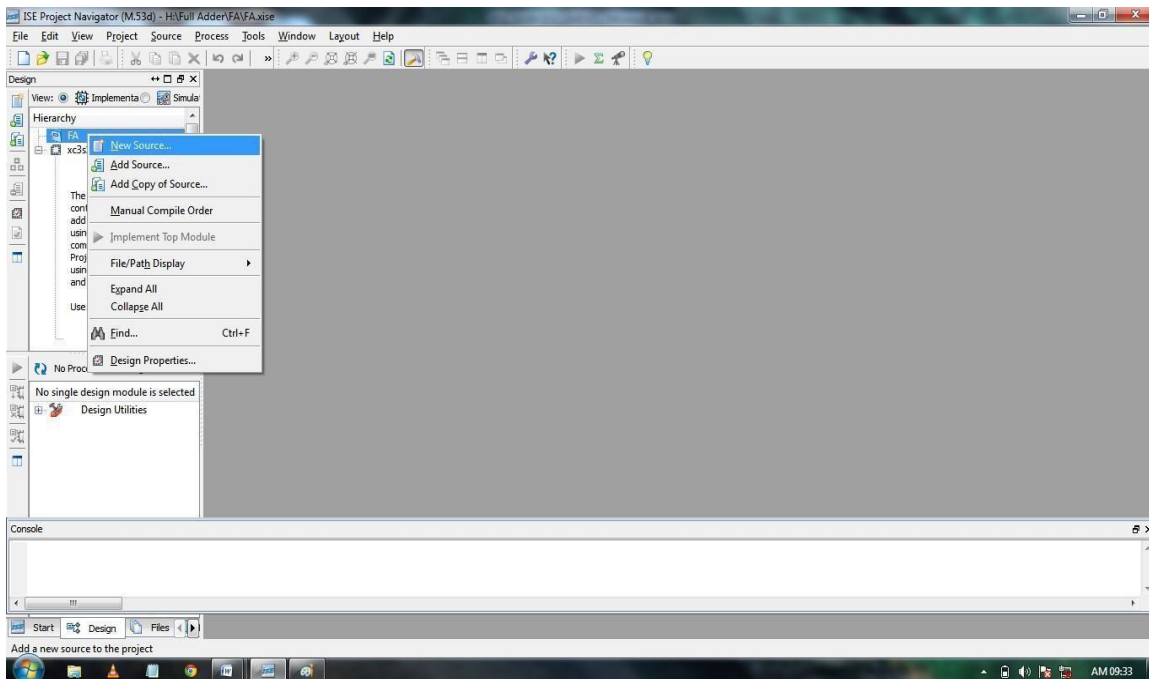
To access an current project in Xilinx Software, select File->Open Project to view a collection of projects on the system. Choose the project you like, and press OK.

Clicking the NEXT button on the browser above would launch the tab below:

Design of windowed watch dog timer for embedded system faulty control



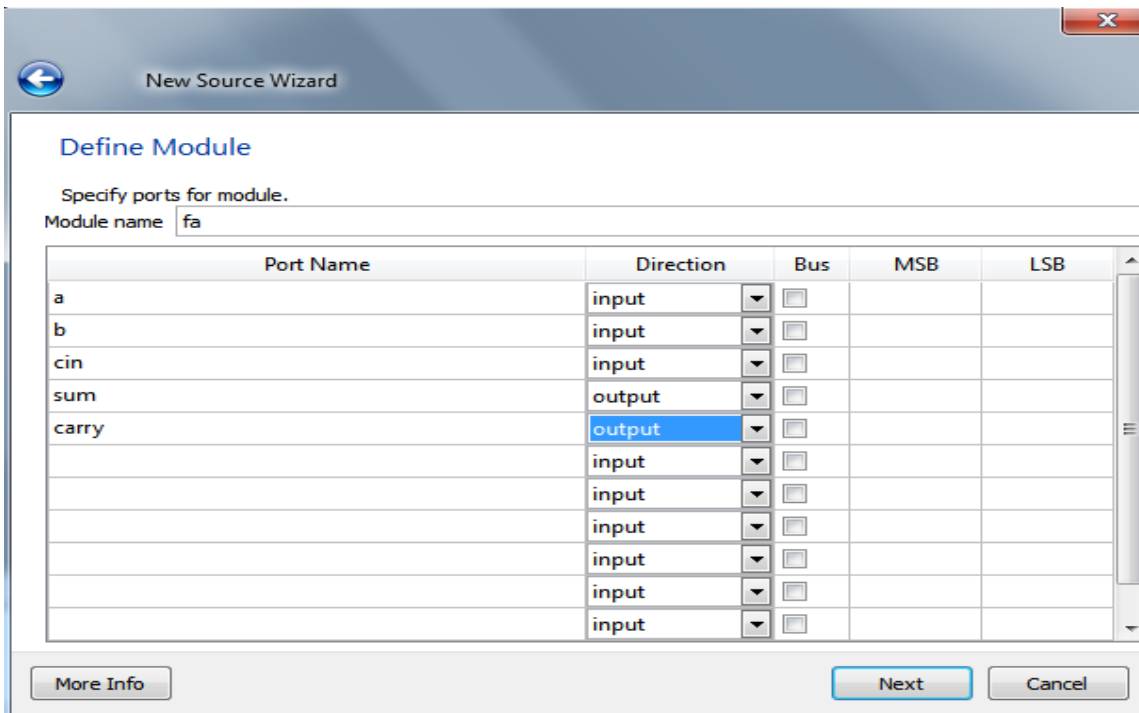
Click this same NEW SOURCE button to create a new source file.



A pop-up window has been opened.

Select Verilog Module then type the name of the project in "File Name:." Then click Next to recognize the entries. This will pop up the following window.

Design of windowed watch dog timer for embedded system faulty control



Define Module

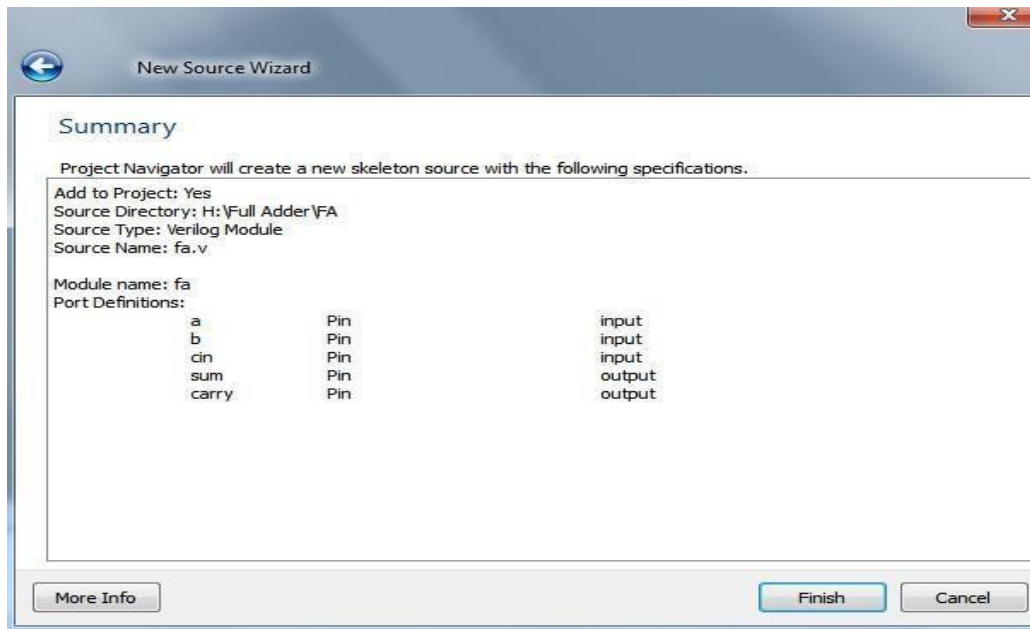
Specify ports for module.

Module name:

Port Name	Direction	Bus	MSB	LSB
a	input	<input type="checkbox"/>		
b	input	<input type="checkbox"/>		
cin	input	<input type="checkbox"/>		
sum	output	<input type="checkbox"/>		
carry	output	<input type="checkbox"/>		
	input	<input type="checkbox"/>		
	input	<input type="checkbox"/>		
	input	<input type="checkbox"/>		
	input	<input type="checkbox"/>		
	input	<input type="checkbox"/>		
	input	<input type="checkbox"/>		

Buttons: More Info, Next, Cancel

In the title of the Port post, enter the names of all of the input and output pins, and identify the direction in the appropriate place. You may provide a description of the Vector / Bus by attempting to input the relevant bit numbers across the MSB / LSB columns. Then, select the Next button to bring up a window that lists all of the newly created source material.



Summary

Project Navigator will create a new skeleton source with the following specifications.

Add to Project: Yes
 Source Directory: H:\Full Adder\FA
 Source Type: Verilog Module
 Source Name: fa.v

Module name: fa

Port Definitions:

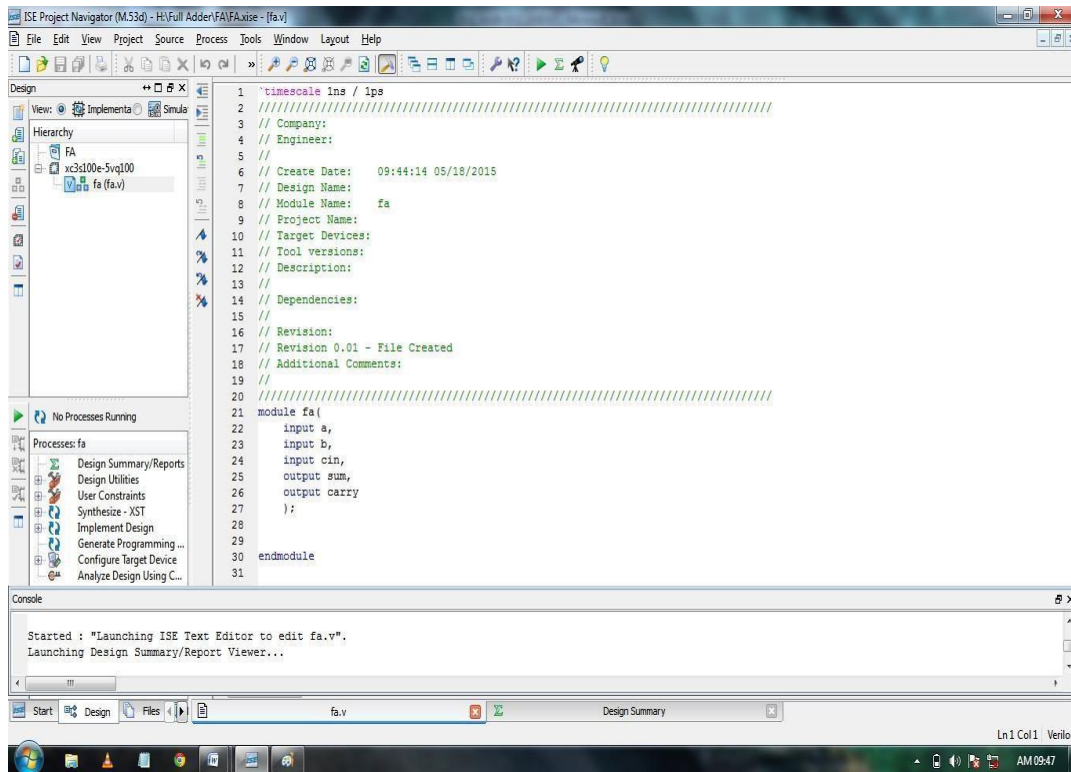
Port Name	Direction
a	input
b	input
cin	input
sum	output
carry	output

Buttons: More Info, Finish, Cancel

Design of windowed watch dog timer for embedded system faulty control

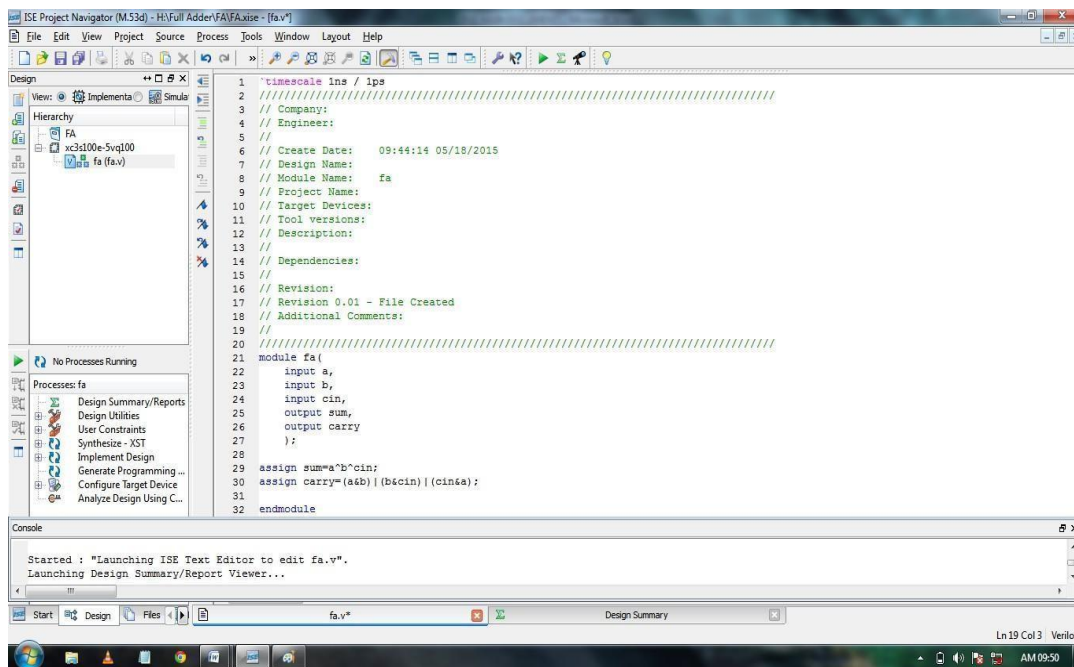
Click Finish to continue.

The source file also will be shown in the navigation pane window.



The overview of the data source may be used as a word processor to make any modifications that are necessary to the data source. This displays both of the input / output keys. Save your Verilog files on a daily basis by pressing the File->Save menu element. Verilog programs can also be changed in any text editor and added to the project directory using "Download Copy Code".

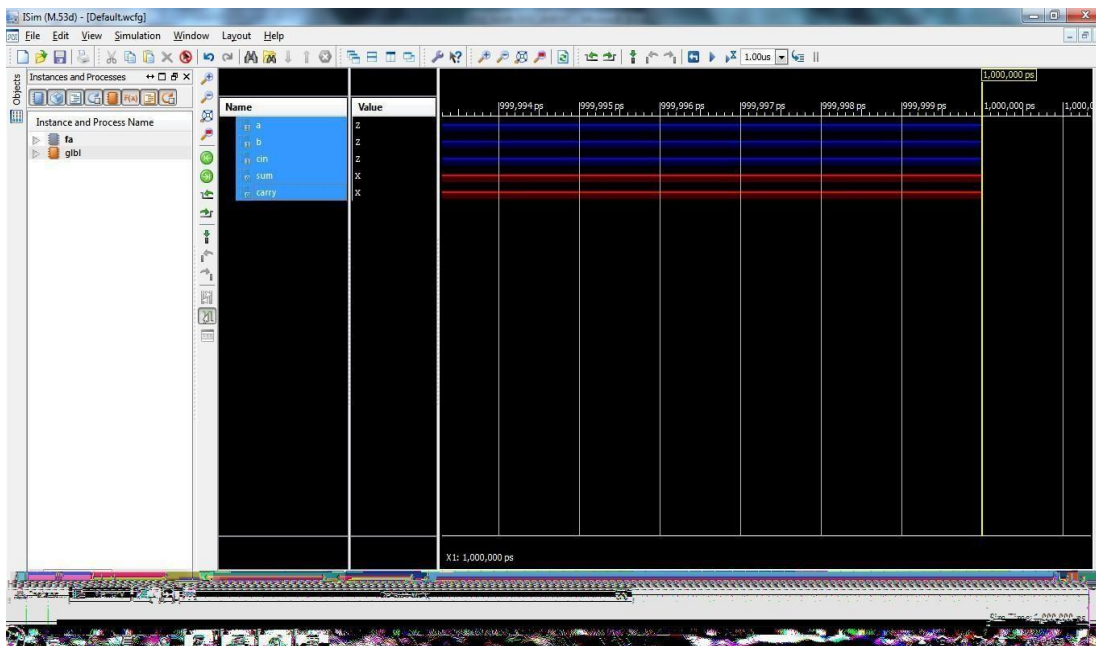
Design of windowed watch dog timer for embedded system faulty control



Simulating and Viewing the Output Waveforms

Click upon this simulation button to select an existing file and expand the ISim Simulator, as well as click upon this Behavioral Check Syntax button to check Errors.

Click upon this replicate model theory when there are no mistakes. A pop-up window opens.

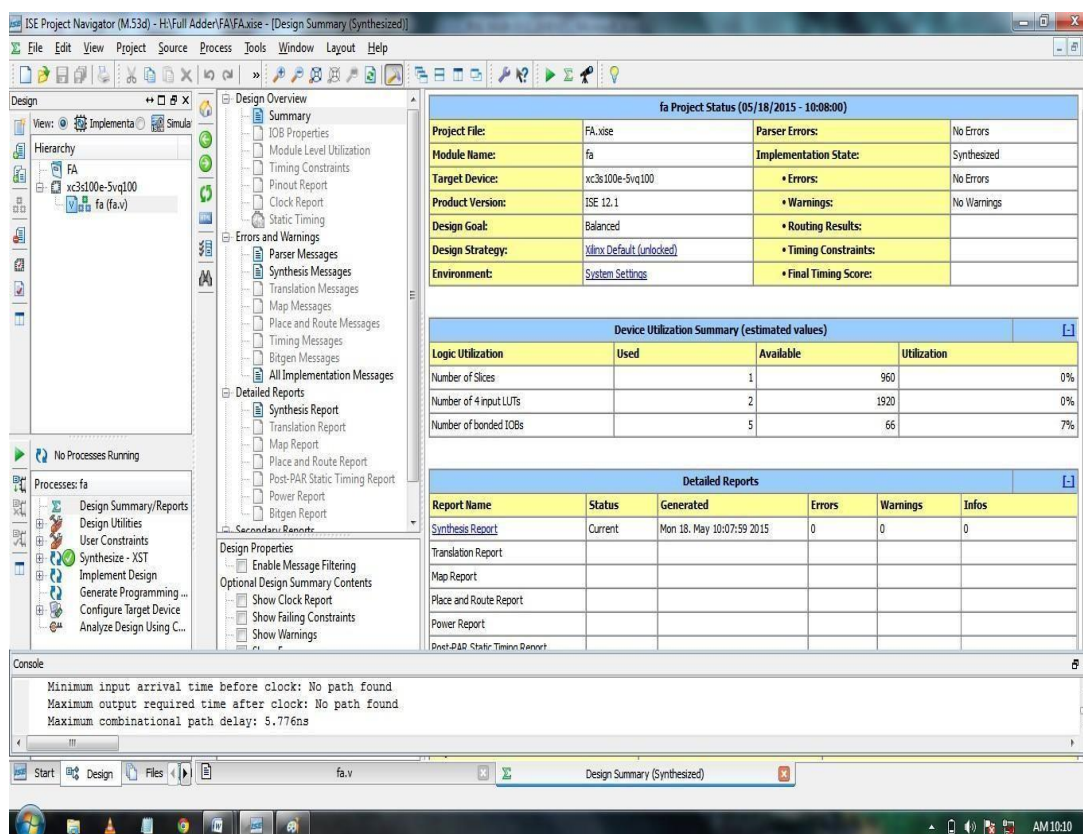


Design of windowed watch dog timer for embedded system faulty control

Here, we will send the data. Rated speed on the chosen data, press on the constant power, and press OK to access the data input.

Select Run option to test different output waveforms throughout the toolbar.

Click Setup to pick an current file and double click Synthesize-XST. If there are any errors, resolve it. Since there are no mistakes, click on the description of the template and the documentation.



The screenshot displays the ISE Project Navigator window for project 'fa.v'. The Design Overview pane on the left shows the project hierarchy. The central Design Summary pane is titled 'fa Project Status (05/18/2015 - 10:08:00)' and contains the following data:

fa Project Status (05/18/2015 - 10:08:00)			
Project File:	fa.v	Parser Errors:	No Errors
Module Name:	fa	Implementation State:	Synthesized
Target Device:	xc3s100e-5vq100	Errors:	No Errors
Product Version:	ISE 12.1	Warnings:	No Warnings
Design Goal:	Balanced	Routing Results:	
Design Strategy:	Virtex Default (unlocked)	Timing Constraints:	
Environment:	System Settings	Final Timing Score:	

Below this, the Device Utilization Summary (estimated values) is shown:

Logic Utilization	Used	Available	Utilization
Number of Slices	1	960	0%
Number of 4 input LUTs	2	1920	0%
Number of bonded IOBs	5	66	7%

The Detailed Reports section at the bottom shows a table with columns: Report Name, Status, Generated, Errors, Warnings, and Infos. The Synthesis Report is listed as 'Current' and generated on 'Mon 18 May 10:07:59 2015' with 0 errors and 0 warnings.

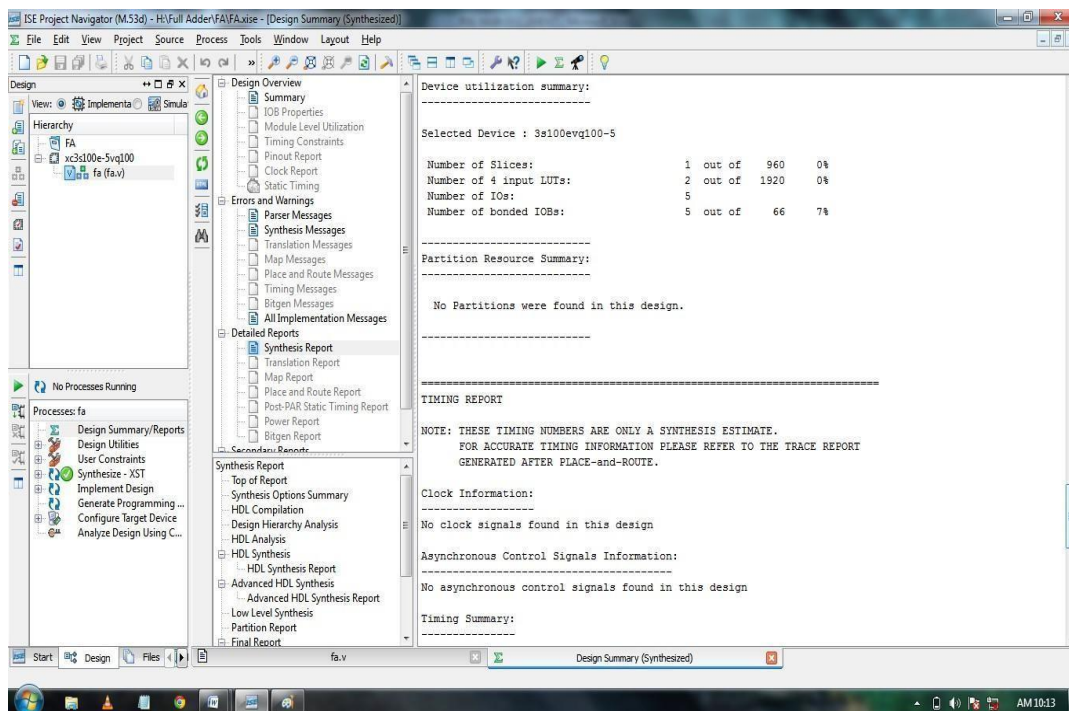
The Console pane at the bottom displays the following messages:

```

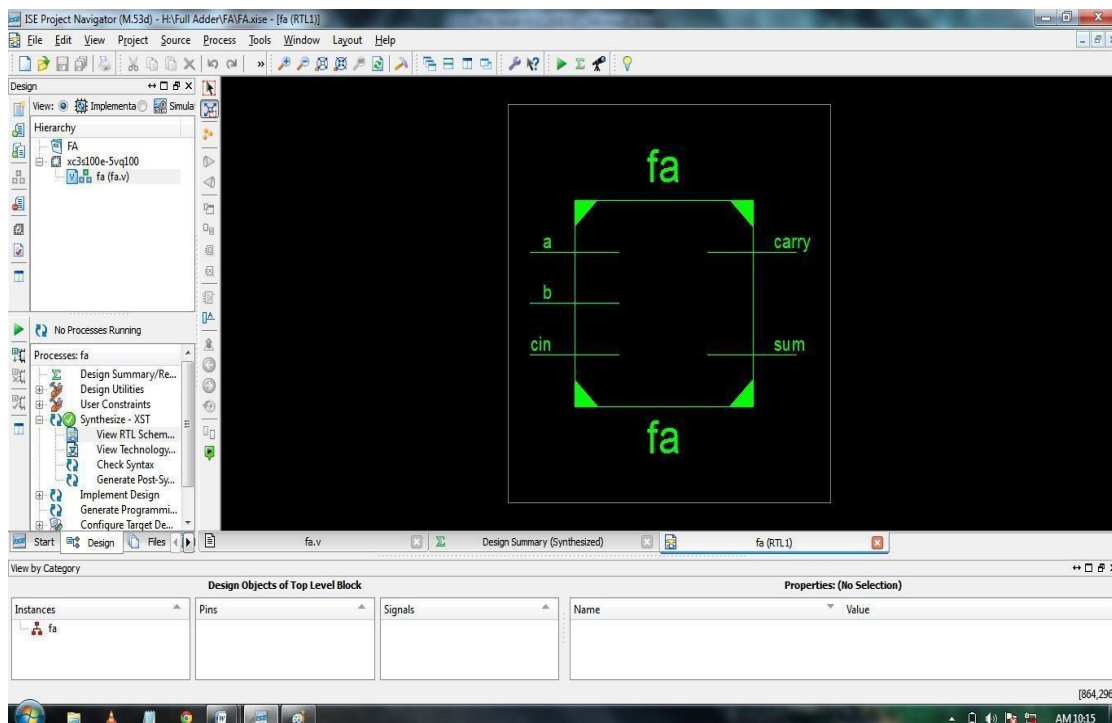
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 5.776ns
  
```

Open the Simulation Page in the Comprehensive Documents seeing the Description of Tool Usage with Timing Analysis for the current mission.

Design of windowed watch dog timer for embedded system faulty control

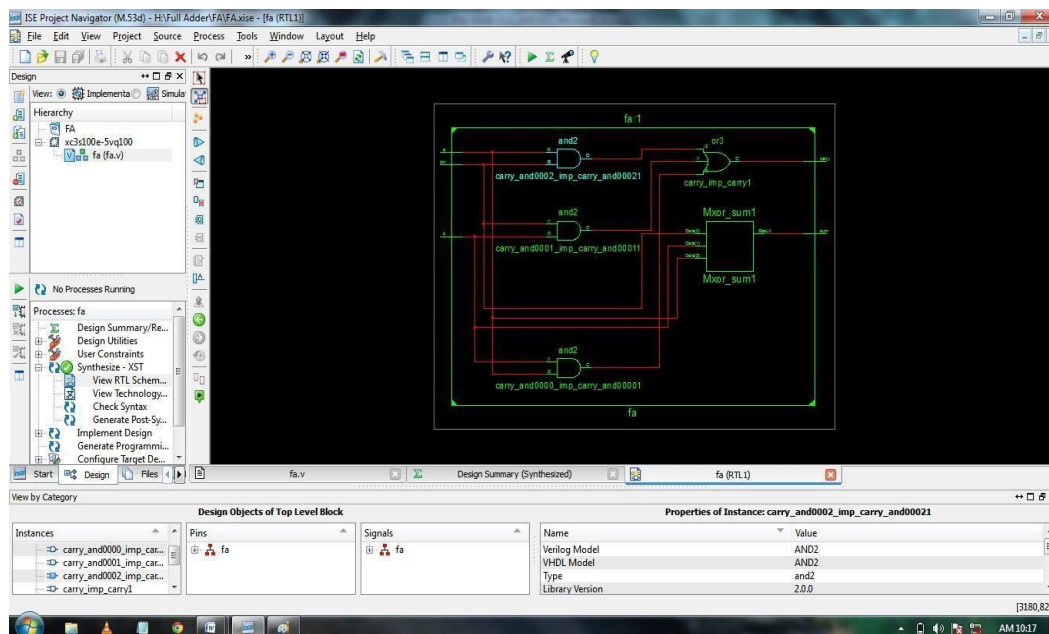


Extend Synthesize-XST as well as press Display RTL Configuration and select OK.



To view the internal modules, click the top module to open the Top module window.

Design of windowed watch dog timer for embedded system faulty control



6.7 FPGA DESIGN FLOW:

For this section of the tutorial, humans 're going to get a brief introduction to the FPGA interface flow. A simplified version of the proposed design is shown in the process flow.

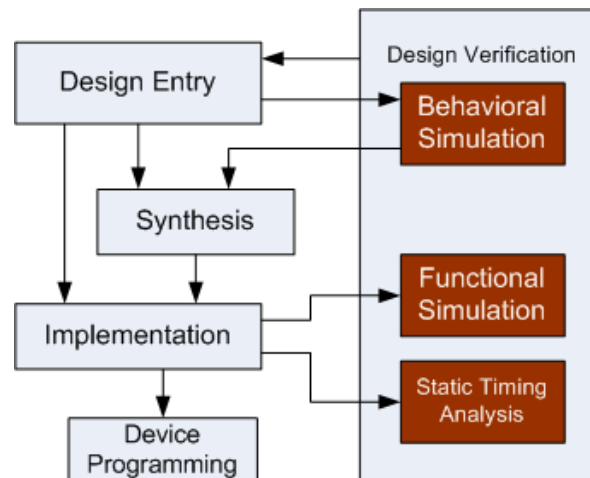


Figure 6.1 FPGA Design Flow

Distinctive approaches are essential for moving the program. Schematic, System Definition Word & mixture of together, and so forth. Collection of a strategy be contingent on the designer & the program. Will fashioner need discuss further with Hardware, the best choice would be at that stage throughout the Schematic portion. If the enterprise is multifaceted the expensive considers strategy in an adjective way formerly HDL is best alternative.

HDLs refer to a degree of contemplation which may separate the creators after the basic elements of the application of machinery. Inovative-based portion provides considerably more perceptibility to fashioners inside the appliances. For the persons who are placed machinery, it is the best option. The technique that is seen every now and then is state-machines.

To developers who feel the program is a succession of nations, it is the right option. Be that as it can, the movement of the state machine equipment is restricted. We must handle the HDL dependent program portion of this report.

Organize a method that follows VHDL or Verilog code in such a gadget chart. That is, the whole circuit of transparent components (entryways, turn lemon, etc ...) for the system. Will the template contain upwards of one sub-plan , for example. In order to access the cpu, we need the CPU as one item and the RAM as another part, and so on, at this point the adaptation mechanism generates a list with each outline element Synthesis phase checks the formatting code and deconstructs the detail control chain this ensures that perhaps the template is updated to the summary engineering selected by fashioner. The corresponding netlist(s) is removed from the NGC(Native Generic Circuit) record (for Xilinx ® Synthesis Technology (XST)).

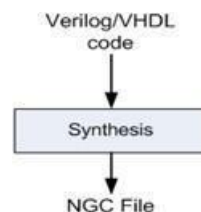


Figure 6.2. FPFA systhesis

The development of DWT and IDWT in this research is accomplished through using Verilog HDL but is sequenced by XILINX ISE Method on the Spartan 3E FPGA unit. This method encompasses:

- Translate
- Map
- Place and Route

Method connects all the data netlists and imperatives to a clear database of the setup. Such details are saved as a record of the NGD (Native Generic Database). Usage of NGD Construct software would be necessary for this. Here, The only describing constraints are the distribution of the ports in the outline to the actual objects (e.g. wires, buttons, holds, etc.)

The gizmo-focused and the certainty of the space specifications of the outlining. This information is included in the UCF (bondholders File) tab.

The methods used to make or alter the UCF are Speed, Restriction Editor and so forth.

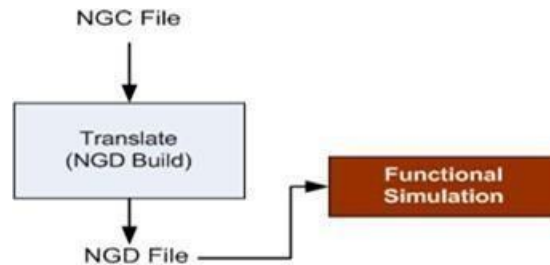


Figure. 6.3. FPGA Translate

System isolates the whole circuit in post-squares of intelligent elements to such an degree which they can blend into the squares of FPGA justification. That implies overview fits the NGD record reasoning into the FPGA component focused (Combinational Logic Blocks (CLB), Input Output Blocks (IOB)), and produces an NCD (Native Circuit Description) document that mentally speaks to the FPGA segment mapped plan.

For the this reason the guide program is used.

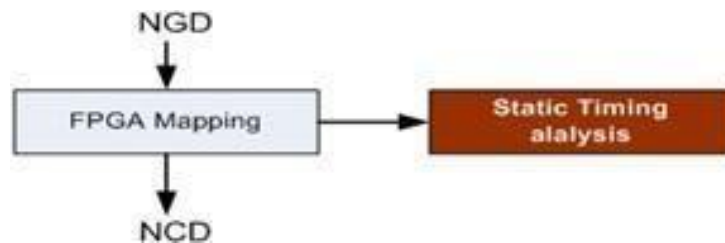


Figure. 6.4. FPGA Map

This technique is done in regular software. The position and course method incorporates the sub-obstructions as per the shortcomings from the guidance protocol into justification hinders and integrates the squares of the justification. E.g. When a sub square is put in a logical square that is extremely similar to an IO mark, it will spare time at that stage, but it could cause any other constraint.

Thus, the place and course process consider tradeoff between each of the limitations

The PAR system takes as information the plotted NCD log, and generates a completely guided NCD report by way of produce. Produce NCD greatest contains instruction details.

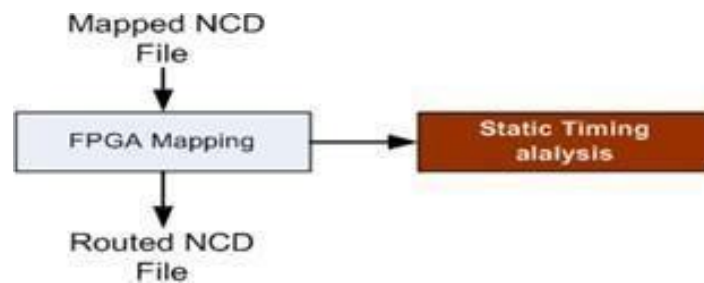


Figure. 6.5. FPGA Place and Route

The programmable gate array (FPGA) will already have the software loaded onto it. Despite this, the design will be converted to a format that is comprehensible by the FPGA so that it may be implemented. The BITGEN system will manage the changeover when it occurs. After that, the NCD record is sent on to the BITGEN software so that it may generate a bit stream (also known as a BIT document) that can be used to organize the FPGA device as a target. Using a link should make doing this task simple and quick. The approach will determine whether or not the connection option is available.

Verification can be done at various stages of the process.

It is as a question of paramount significance re-enactment steps; which are observed in the outline stream chain of command. Such replication is conducted to test RTL (conduct) code before the combination protocol and to verify that the program is functioning as applied.

Reenactment of the actions may be carried out either on VHDL or on Verilog plans. Signs and variables are identified in this process, methods and capacities are pursued, and breakpoints are set.

It is a simple replication, thereby enabling the developer to regulate the HDL cypher if the desired usefulness achieved in a small timeframe. Although description is structured the point of entry, the properties of timing and usage of materials are still elusive.

Useful reproduction gives data regarding the circuit 's rational task. After the Translate method, the architect may test the usefulness of the design that uses this technique. In case the benefit isn't shocking, the fashioner has to carried out changes in the code at that stage and take action into the program stream again.

Chapter 7

Results and Discussions

This section gives a detailed simulation analysis of the proposed watch dog timer system. Here, the simulations are conducted using Xilinx-Vivado software tool. Figure 7.1 provides a visual representation of the watchdog timer's initialization and general operation (simulation) waveforms. At power-up or reset, the watchdog enters a failed state, indicated by the asserted high state of the WDFAIL output. It becomes the software's responsibility to initialize and activate the watchdog for ongoing operation. To achieve this, the software must first toggle the watchdog reset (WDRST) field in the configuration register from low to high. Subsequently, servicing the watchdog within the designated service window will result in the de-assertion of the WDFAIL flag, signifying that the watchdog is now operational. Importantly, the frame window is intentionally kept larger than the system's frame time, ensuring that another service window initiates before the current frame window expires.

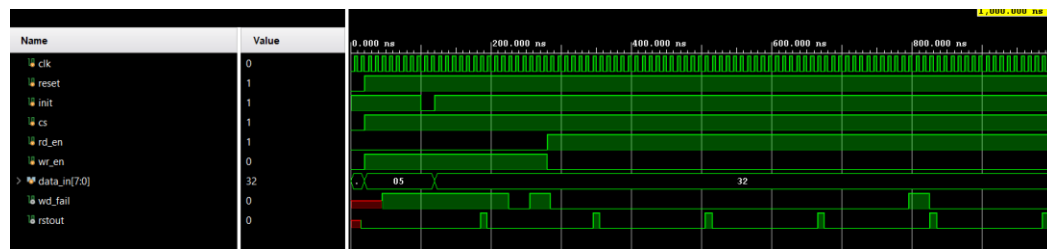


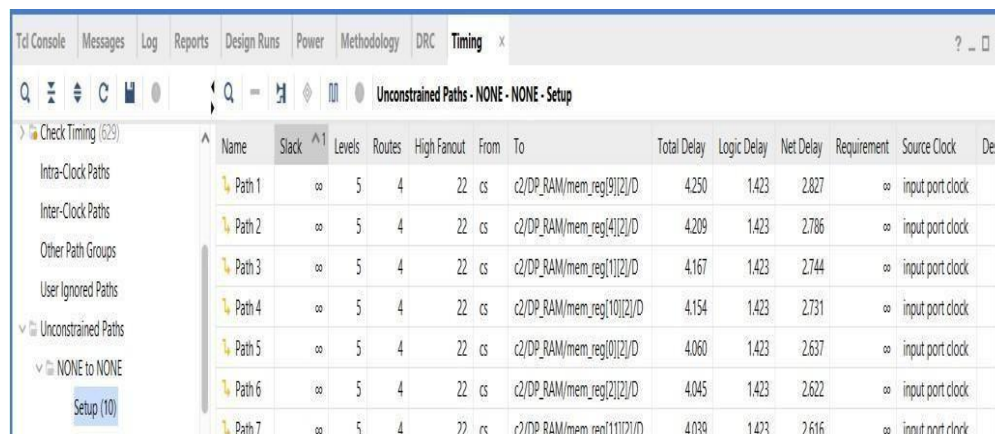
Figure 7.1. Simulation output

Figure 7.2 affords a concise design summary, presenting critical information approximately the FPGA implementation of the watchdog timer. It consists of key metrics that spotlight the utilization of precise resources in the FPGA layout. The proposed design makes use of fifty-seven out of a complete of 230400 available Look-Up Tables (LUTs) inside the FPGA. LUTs are essential good judgment elements in an FPGA that may be configured to perform diverse logical operations. The proposed layout uses 27 out of a total of 460800 available Flip-Flops (FF) inside the FPGA. Flip-Flops are critical for storing and synchronizing facts within the FPGA. Further, 12 out of 464 available Input/Output (IO) pins within the FPGA are employed by the design. IO pins are used for connecting the FPGA to external devices or interfaces. Finally, 1 out of 544 available Global Buffer (BUFG) sources within the FPGA is utilized. BUFGs are essential for dispensing clock indicators in the FPGA layout, ensuring right synchronization.

Resource	Estimation	Available	Utilization...
LUT	57	230400	0.02
FF	27	460800	0.01
IO	12	464	2.59
BUFG	1	544	0.18

Figure 7.2. Design summary.

Figure 7.4 affords a summary of setup time measurements for the layout. It affords crucial records about the timing characteristics of the FPGA-based watchdog timer implementation. The setup delay for the layout is 4.250 ns (nanoseconds). This total delay represents the time it takes for alerts to propagate through the complete design, from enter to output. Logic Delay is 1.42 ns of the total delay is attributed to good judgment delay. Logic delay refers to the time it takes for signals to pass via the combinational logic elements within the layout, which includes gates and turn-flops. Finally, 2.8 ns of the whole put off is because of net put off.

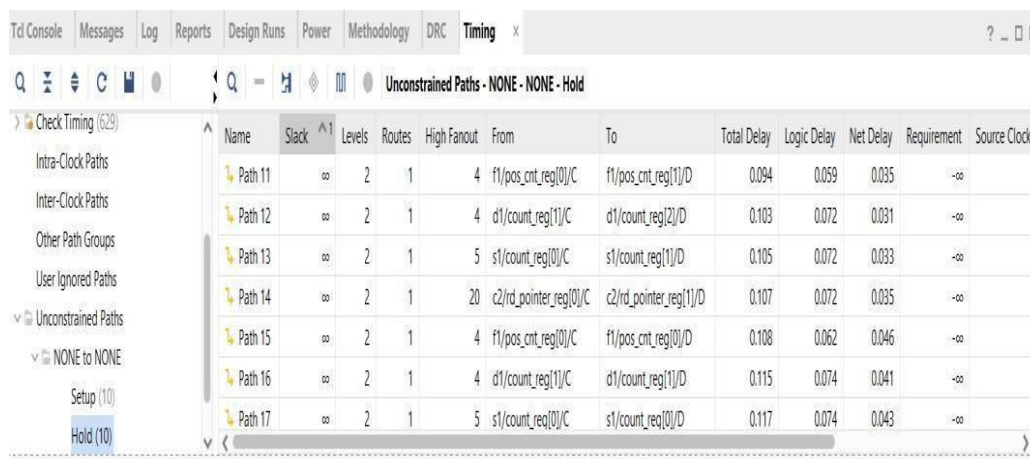


Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Des
Path 1	∞	5	4	22	cs	c2/DP_RAM/mem_reg[9][2]/D	4.250	1.423	2.827	∞	input port clock	
Path 2	∞	5	4	22	cs	c2/DP_RAM/mem_reg[4][2]/D	4.209	1.423	2.786	∞	input port clock	
Path 3	∞	5	4	22	cs	c2/DP_RAM/mem_reg[1][2]/D	4.167	1.423	2.744	∞	input port clock	
Path 4	∞	5	4	22	cs	c2/DP_RAM/mem_reg[10][2]/D	4.154	1.423	2.731	∞	input port clock	
Path 5	∞	5	4	22	cs	c2/DP_RAM/mem_reg[0][2]/D	4.060	1.423	2.637	∞	input port clock	
Path 6	∞	5	4	22	cs	c2/DP_RAM/mem_reg[2][2]/D	4.045	1.423	2.622	∞	input port clock	
Path 7	∞	5	4	22	cs	c2/DP_RAM/mem_reg[11][2]/D	4.039	1.423	2.616	∞	input port clock	

Figure 7.4. Setup Time summary.

Figure 7.5 presents a summary of hold time measurements for the design. It provides essential information about the timing characteristics of the FPGA-based watchdog timer implementation, specifically focusing on hold time requirements. The total hold time delay for the design is 0.094 ns (nanoseconds). Hold time is the minimum time that a signal must remain stable after the clock edge arrives to ensure correct data capture in a flip-flop or register. Here, 0.059 ns of the total hold time delay is attributed to logic delay. Then 0.035 ns of the total hold time delay is due to net delay.

Design of windowed watch dog timer for embedded system faulty control



Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 11	∞	2	1	4	f1/pos_cnt_reg[0]/C	f1/pos_cnt_reg[1]/D	0.094	0.059	0.035	-∞	
Path 12	∞	2	1	4	d1/count_reg[1]/C	d1/count_reg[2]/D	0.103	0.072	0.031	-∞	
Path 13	∞	2	1	5	s1/count_reg[0]/C	s1/count_reg[1]/D	0.105	0.072	0.033	-∞	
Path 14	∞	2	1	20	c2/rd_pointer_reg[0]/C	c2/rd_pointer_reg[1]/D	0.107	0.072	0.035	-∞	
Path 15	∞	2	1	4	f1/pos_cnt_reg[0]/C	f1/pos_cnt_reg[0]/D	0.108	0.062	0.046	-∞	
Path 16	∞	2	1	4	d1/count_reg[1]/C	d1/count_reg[1]/D	0.115	0.074	0.041	-∞	
Path 17	∞	2	1	5	s1/count_reg[0]/C	s1/count_reg[0]/D	0.117	0.074	0.043	-∞	

Figure 7.5. Hold Time summary.

Figure 7.6 provides a summary of power consumption measurements for the FPGA-based watchdog timer design. It offers important information about the power characteristics of the implementation. The total power consumption for the design is 1.7 units, which is typically measured in watts (W). Within the total power, it specifies that 1.160 units are attributed to total dynamic power. Dynamic power is the power consumed by the FPGA when it is actively performing logic operations and transitioning between states. It includes different components. Here, Signal Power (0.288 units) represent the power consumed due to signal transitions and activity. Then, Logic Power (0.320 units) represent the power consumed by the logic elements during their operation. Then, the IO Power (0.552 units) represent the power consumed by Input/Output (IO) operations and interconnections. Within the total power, Total Static Power indicates that 0.597 units is the power consumed by the FPGA even when it is not actively performing logic operations and is typically associated with leakage currents. It includes different components. Here, PL Static (0.584 units) represent the static power consumption in the programmable logic (PL) part of the FPGA. Then, PS Static (0.013 units) represent the static power consumption in the processing system (PS) part of the FPGA, if applicable.

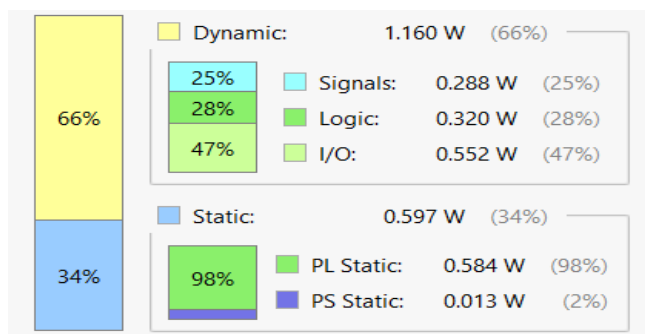


Figure 7.6 Power summary

Design of windowed watch dog timer for embedded system faulty control

According to Table 1, in terms of Look-Up Tables (LUTs), the proposed system demonstrates a remarkable 68% reduction, utilizing only 57 LUTs compared to the existing system's 178. This represents a substantial improvement in logic resource efficiency. Similarly, in the case of Input/Output Blocks (IoBs), the proposed system achieves a 54% reduction, utilizing 27 IoBs compared to the existing system's 59. The Input/Output (IO) resources also see a substantial 73% reduction, with the proposed system using only 12 IOs, while the existing system employs 45. Finally, in terms of Global Buffers (BUFGs), the proposed system showcases an impressive 86% reduction, utilizing just 1 BUFG, whereas the existing system requires 7. These improvements clearly highlight the efficiency gains achieved by the proposed system in terms of resource utilization, paving the way for a more optimized and effective watchdog timer implementation.

Table. 1. Area performance comparison.

Resource	Existing System	Proposed System
LUTs	178	57
IoBs	59	27
IO	45	12
BUFG	7	1

According to Table 2, in terms of Logic Setup Delay, the proposed system showcases a remarkable 45% reduction, with a setup delay of 1.42 ns compared to the existing system's 2.57 ns. This signifies a significant enhancement in the efficiency of logic path setup times. Similarly, for Net Setup Delay, the proposed system achieves a noteworthy 21% reduction, with a setup delay of 2.8 ns compared to the existing system's 3.57 ns. The Total Setup Delay, which considers the combined setup delays, is also substantially improved, showcasing a 31% reduction with the proposed system's 4.250 ns compared to the existing system's 6.14 ns. Regarding Hold Delays, the proposed system achieves a substantial 43% reduction in Logic Hold Delay, with a hold delay of 0.059 ns compared to the existing system's 0.103 ns. Additionally, for Net Hold Delay, there is a remarkable 69% reduction, with the proposed system's 0.035 ns compared to the existing system's 0.113 ns.

Table. 2. Delay performance comparison.

Dealy (ns)	Existing System	Proposed System
Logic Setup Delay	2.57	1.42
Net Setup Delay	3.57	2.8
Total Setup Delay	6.14	4.250
Logic Hold Delay	0.103	0.059
Net Hold Delay	0.113	0.035
Total Hold Delay	0.216	0.094
Total Delay	6.37	4.3

According to Table 3, in terms of Signal Power, the proposed system demonstrates an impressive 31% reduction, with a signal power consumption of $0.288 \mu\text{W}$ compared to the existing system's $0.418 \mu\text{W}$. This represents a significant enhancement in power efficiency related to signal transitions and activity. Similarly, for Logic Power, the proposed system achieves a notable 32% reduction, with a logic power consumption of $0.320 \mu\text{W}$ compared to the existing system's $0.471 \mu\text{W}$. Additionally, in the case of IoB Power, there is a substantial 9% reduction, with the proposed system's IoB power consumption at $0.552 \mu\text{W}$ compared to the existing system's $0.610 \mu\text{W}$. The Total Dynamic Power, which combines all dynamic power components, also experiences a significant 26% reduction, with the proposed system's total dynamic power at $1.160 \mu\text{W}$ compared to the existing system's $1.564 \mu\text{W}$. Turning to static power, the proposed system achieves a remarkable 20% reduction in PL Static Power, with a power consumption of $0.584 \mu\text{W}$ compared to the existing system's $0.735 \mu\text{W}$. Moreover, for PS Static Power, there is an exceptional 87% reduction, with the proposed system's PS static power consumption at $0.013 \mu\text{W}$ compared to the existing system's $0.101 \mu\text{W}$. The Static Power, which includes both PL and PS static power, demonstrates a significant 28% reduction, with the proposed system's static power consumption at $0.597 \mu\text{W}$ compared to the existing system's $0.831 \mu\text{W}$.

Table. 3. Power performance comparison.

Power (uw)	Existing Filter	Proposed Filter
Signal Power	0.418	0.288
Logic Power	0.471	0.320
IoB Power	0.610	0.552
Total Dynamic Power	1.564	1.160
PL Static	0.735	0.584
PS Static	0.101	0.013
Static Power	0.831	0.597
Total Power Consumption	2.4	1.7



Chapter 8

Conclusion and Future Scope

8.1 Conclusion

The windowed watchdog timer offers a strong solution for reinforcing the reliability and precision of embedded manage systems. It addresses current drawbacks related to conventional watchdog timers through presenting flexible window configuration, advanced security features, and dynamic initialization techniques. These improvements translate into advanced machine reliability, reduced vulnerability to software errors, and adaptableness to converting machine situations. In protection-essential and actual-time applications, the windowed watchdog timer proves to be an asset, preventing device screw ups and ensuring the smooth operation of embedded manage systems. The future holds promising possibilities for the evolution of watchdog timer generation and its integration into a extensive range of applications. Continuously enhancing security features to guard in opposition to potential assaults or unauthorized get admission to to configuration registers, further fortifying device integrity.

8.2 Future Scope

The future scope of watchdog timers encompasses several promising directions driven by advancements in embedded systems, IoT (Internet of Things), and real-time computing technologies. These developments are poised to enhance system reliability, security, and efficiency across diverse industries.

- **Integration with AI and Machine Learning:** One exciting area is the integration of watchdog timers with artificial intelligence (AI) and machine learning algorithms. By analyzing patterns of system behavior and failure, intelligent watchdogs can adaptively adjust their timeout thresholds and recovery strategies. This can lead to more proactive fault detection and optimized system recovery, reducing downtime and improving overall system performance.
- **Enhanced Security Measures:** With the growing concerns around cybersecurity, watchdog timers can play a crucial role in detecting and mitigating potential attacks or unauthorized access. Future watchdog implementations may include sophisticated security features such as intrusion detection, anomaly detection, and secure boot mechanisms.



- **Application in Edge Computing:** As edge computing continues to gain traction, watchdog timers will become essential in distributed computing environments where devices operate autonomously with limited human intervention. In edge scenarios, watchdog timers can ensure the continuous operation of critical services and applications, even in remote or harsh environments.
- **Advanced Fault Tolerance Mechanisms:** Future watchdog timers may incorporate advanced fault tolerance mechanisms beyond simple reset actions. These could include dynamic workload migration, self-healing algorithms, or redundancy strategies to maintain system integrity and availability in the face of failures.
- **Standardization and Interoperability:** With the proliferation of IoT devices and heterogeneous computing systems, there will be a push towards standardization and interoperability of watchdog timers across different platforms and architectures. This will simplify integration efforts and ensure consistent behavior and performance across diverse embedded systems.
- **Energy-Efficient Designs:** As energy efficiency becomes a key consideration in embedded systems, future watchdog timers will aim to minimize power consumption while maintaining robust monitoring capabilities. This may involve innovations in low-power design techniques or adaptive timer management strategies.
- **Automotive and Autonomous Systems:** In automotive and autonomous systems, watchdog timers will play a critical role in ensuring the safety and reliability of onboard electronics. Future developments may focus on integrating watchdog functionalities with advanced driver-assistance systems (ADAS) and autonomous vehicle platforms to address stringent safety requirements.
- **Healthcare and Medical Devices:** The healthcare industry will benefit from watchdog timers in medical devices and equipment to ensure continuous operation and patient safety. Future advancements may focus on integrating watchdog functionalities with real-time monitoring systems and telemedicine applications.



References

- [1] Steblevska, I., T. Feuerstake, and M. R. Daymond. "Supervisor watchdog circuit to monitor an accelerator beam and control the safety interlock system." *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 1056 (2023): 168595.
- [2] Neeharika, T., et al. "Maximizing efficiency: building and running an AMBA APB protocol with open-source solutions." (2023): 99-103.
- [3] Yang, Weitao, et al. "System-on-chip single event effect hardening design and validation using proton irradiation." *Nuclear Engineering and Technology* 55.3 (2023): 1015-1020.
- [4] Aviles, Pablo M., et al. "Supervised Triple Macrosynchronized Lockstep (STMLS) Architecture for Multicore Processors." *IEEE Access* 11 (2023): 128706-128723.
- [5] Arshath, Mohamed. "Detection Of Soft Errors in Clock Synthesizers and Latency Reduction Throgh Voltage Scaling Mechanism." *Journal of VLSI circuits and systems* 6.1 (2024): 43-50.
- [6] Albalooshi, Amina, Abdul-Halim M. Jallad, and Prashanth R. Marpu. "Fault Analysis and Mitigation Techniques of the I2C Bus for Nanosatellite Missions." *IEEE Access* 11 (2023): 34709- 34717.
- [7] dos Santos, Douglas Almeida, Pablo M. Aviles, André Martins Pio de Mattos, Mario García Valderas, Luis Entrena, Almudena Lindoso, and Luigi Dilillo. "Hybrid Hardening Approach for a Fault-Tolerant RISC-V System-on-Chip." In *RADECS 2023-European Conference on Radiation and Its Effects on Components and Systems*. 2023.
- [8] Azambuja, J. R., Sousa, F., Rosa, L., & Kastensmidt, F. L. (2023). Evaluating the efficiency of software-only techniques to detect SEU and SET in microprocessors. *arXiv preprint arXiv:2309.16876*.
- [9] Mo, Prasad Pa, and Arvind Singh. "FPGA based object parameter detection for Embedded Vision Application." *International Journal of Computing and Digital Systems* 14.1 (2023): 1091-1099.
- [10] Nikiema, P. R., Palumbo, A., Aasma, A., Cassano, L., Kritikakou, A., Kulmala, A., ... & Traiola, M. (2023, July). Towards dependable RISC-V cores for edge computing devices. In *2023 IEEE 29th International Symposium on On-Line Testing and Robust System Design (IOLTS)* (pp. 1-7). IEEE.



-
- [10] Mattos, A. M., Santos, D. A., Imianosky, C., Melo, D. R., & Dilillo, L. (2023, June). Using HARV-SoC for Reliable Sensing Applications in Radiation Harsh Environments. In *2023 9th International Workshop on Advances in Sensors and Interfaces (IWASI)* (pp. 227-232). IEEE.
- [11] Zagan, Ionel, and Vasile Gheorghită Găitan. "FPGA implementation of hardware accelerated RTOS based on real-time event handling." *The Journal of Supercomputing* (2023): 1-31.
- [12] Palumbo, Alessandro, et al. "Improving the Detection of Hardware Trojan Horses in Microprocessors via Hamming Codes." *2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE, 2023.
- [13] Cherezova, Natalia, et al. "Understanding fault-tolerance vulnerabilities in advanced SoC FPGAs for critical applications." *Microelectronics Reliability* 146 (2023): 115010.
- [14] Groshev, A., Solodilov, M., Gusev, P., & Malysheva, A. (2023). Formation of a management strategy for innovation and investment activities of an enterprise. In *E3S Web of Conferences* (Vol. 458, p. 05034). EDP Sciences.
- [15] Santos, D. A., Mattos, A. M., Melo, D. R., & Dilillo, L. (2023, October). Characterization of a Fault-Tolerant RISC-V System-on-Chip for Space Environments. In *2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)* (pp. 1-6). IEEE.
- [16] Akanksha, Kammari, and R. Vyshnavi Duggaraju Kasturi. "Design of an Improved Watchdog Timer for MAC Applications." *computing* 52.6 (2023).
- [17] Chaithanya, D. J., et al. "Microarchitecture and Design of a Watchdog Timer for aRISC-V based SoC." *2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)*. IEEE, 2023.
-