FPGA BASED FLIGHT CONTROLLER

A final capstone project report submitted in partial fulfilment of the requirement for the award of the degree of

Bachelor of Engineering

in

Electronics and Communication Engineering

Submitted By

Bhavya Bansal (102206247)

Khushi Saldi (102206089)

Kritarth Upadhyay (102206186)

Navjot Kaur (102206001)

Vaibhav Gupta (102215113)

Group No: 03

Under Supervision of

Dr. Alpana Agarwal (Professor, DECE)

Dr. Anil Singh (Associate Professor, DECE)

Department of Electronics and Communication Engineering

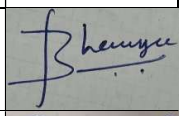
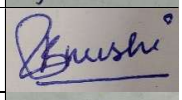
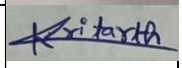
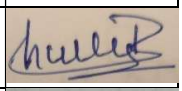
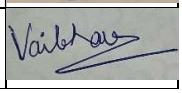
THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY, PATIALA, PUNJAB

November 2025

DECLARATION

We hereby declare that the capstone project group title "FPGA BASED FLIGHT CONTROLLER" is authentic record of our own work carried out at Thapar In- stitute of Engineering and Technology, Patiala as a Capstone Project in seventh semester of B.E. (Electronics and Communication Engineering) under the guidance of Dr. Alpana Agarwal and Dr. Anil Singh, during February to November, 2025.

Date: 25/11/2025

S.NO	NAME	ROLL NO.	SIGN
1.	Bhavya Bansal	102206247	
2.	Khushi Saldi	102206089	
3.	Kritarth Upadhyay	102206186	
4.	Navjot Kaur	102206001	
5.	Vaibhav Gupta	102215113	

Faculty Supervisors:

Dr. Alpana Agarwal (Professor)

Dr. Anil Singh (Associate Professor)

Acknowledgment

We want to express our sincere gratitude to our mentors, Dr. Alpana Agarwal and Dr. Anil Singh, for their invaluable guidance throughout this project. Their expertise, insights, and continuous support were crucial in shaping our research and ensuring its successful execution. This endeavor would not have been possible without their support, guidance, and encouragement.

We also thank our Dr. Kulbir Singh and DECE for their constructive feedback, valuable suggestions, and encouragement. Their dedication to our academic growth and their commitment to excellence have been instrumental in our development as researchers.

We would like to thank the administration and staff of our college for providing us with the necessary resources and facilities to carry out our project. Their efficient assistance and cooperation greatly facilitated our work.

Lastly, we want to acknowledge our families and friends for their love, understanding, and encouragement. Their unwavering support and belief in our abilities have inspired us throughout our journey.

NAME	ROLL NO.
Bhavya Bansal	102206247
Khushi Saldi	102206089
Kritarth Upadhyay	102206186
Navjot Kaur	102206001
Vaibhav Gupta	102215113

Abstract

India's law enforcement system faces persistent challenges due to an imbalanced police-to-population ratio of 152.80 officers per 100,000 civilians, significantly below the UN-recommended 222 [1]. This shortfall limits the ability to manage large gatherings, maintain situational awareness, and respond swiftly in emergencies. Traditional policing methods often suffer from delayed response times, inadequate crowd control, and difficulties in reaching critical locations.

This project proposes the design of an FPGA-based UAV system for search, rescue, and law enforcement applications. At its core is an indigenously developed FPGA flight controller, implemented using Xilinx Vivado, that supports low-latency PWM capture, motor mixing, and closed-loop PID control for stable and precise flight. The system features real-time onboard human detection using Haar cascades and facial recognition on a Raspberry Pi 4 with Pi Camera module, enabling autonomous identification and tracking of individuals of interest.

The integration of FPGA-based flight control with edge AI processing creates a modular, energy-efficient platform suitable for search and rescue missions. A backend database system stores mission telemetry, detection records, and control parameters for post-mission analyt- ics and evidence-grade data integrity. By combining FPGA-based hardware acceleration with AI-driven recognition, the system strengthens the operational capabilities of Indian law enforcement while reducing dependency on foreign technology.

This project thus contributes toward creating a scalable, secure, and indigenous UAV platform tailored for public safety, surveillance, and disaster management in India.

Contents

1 INTRODUCTION **1**

1.1 PROJECT OVERVIEW 1

1.2 MOTIVATION 1

1.3 ASSUMPTIONS AND CONSTRAINTS 2

1.3.1 Assumptions 2

1.3.2 Constraints 3

1.4 NOVELTY OF WORK 3

2 LITERATURE SURVEY **5**

2.1 OVERVIEW 5

2.2 KEY RESEARCH CONTRIBUTIONS 5

2.2.1 Reliable FPGA-Based Architectures for SAR Quadcopters [2] 5

2.2.2 Reconfigurable FPGA Accelerator for Drone-Based SAR [3] 6

2.2.3 MPDrone: FPGA Intelligent Platform for Autonomous UAVs [4] 6

2.2.4 Drone Technology for Pandemic and Disaster Response [5] 6

2.2.5 Fault-Resilient Autonomous Quadcopter Control with DPR FPGA [6] . 7

2.2.6 On-Board Deep Learning Fault Detection for UAVs via FPGA [7] 7

2.2.7 Signal Periodicity-Based UAV Detection on FPGA [8] 8

2.2.8 FPGA-Accelerated Deep Learning UAV Detection System [9] 8

2.2.9 Human Detection in UAV SAR: Survey and State-of-the-Art [10] 9

2.2.10 Survey and Taxonomy of UAV Target Tracking [11] 9

2.3 RESEARCH GAPS 10

2.4 PROBLEM DEFINITION AND SCOPE 11

2.4.1 Problem Statement 11

2.4.2 Project Objectives 11

3 SYSTEM ARCHITECTURE AND FLOWCHART **13**

3.1 SYSTEM ARCHITECTURE 13
3.1.1 Pi Camera Module 3 13
3.1.2 Raspberry Pi 4 13
3.1.3 Ground Station PC 14
3.1.4 TX/RX Radio Module 14
3.1.5 BASYS 3 FPGA - Artix-7 14
3.1.6 IMU Sensor (Gyroscope/Accelerometer) 14
3.1.7 Brushless DC Motors and ESCs 14
3.1.8 Backend Database System 15
3.2 SYSTEM ANALYSIS 15
3.2.1 Camera Input Acquisition 15
3.2.2 Face Detection 15
3.2.3 Face Center Localization 15
3.2.4 Distance Estimation 16
3.2.5 Mapping to PWM Signals 17
3.2.6 PWM Signal Integration 17
3.2.7 PWM capture in FPGA 18
3.2.8 Visual Feedback and Debugging 18
3.3 TECHNOLOGY AND TOOLS USED 18
3.3.1 Hardware Tools and Technology 18
3.3.2 Software Tools and Technology 19

4 PROJECT DESIGN AND DESCRIPTION 21
4.1 PROJECT DESCRIPTION 21
4.2 UNDERGRADUATE SUBJECTS APPLIED 22
4.3 STANDARDS USED 23

5 IMPLEMENTATION AND EXPERIMENTAL RESULTS 24
5.1 SIMULATION RESULTS 24
5.1.1 Tracking Algorithm Performance 24
5.1.2 Performance Summary Table 26
5.2 HARDWARE RESULTS 26
5.2.1 FPGA Implementation 26

5.2.2 Power Analysis 27
5.2.3 Design Timing Summary 28
5.2.4 Design Runs Summary 28
5.2.5 Block level Implementation 29

6 IMPACT ANALYSIS 31
6.1 Technical Impact 31
6.2 Educational Impact 32
6.3 Indian Indigenous and Military Impact 32

7 Financial Analysis 34
7.1 Hardware Requirements and Cost 34
7.2 Justification of Hardware Components 34
7.3 Software Requirements and Cost 34
7.4 Cost-Benefit Summary 35
7.5 Conclusion 35

8 Engineering Ethics, Safety, and Responsibility 36
8.1 Introduction 36
8.2 Ethical Principles and Framework 36
8.3 Safety Measures 36
8.4 Risk Assessment and Mitigation 37
8.5 Environmental and Social Impact 37
8.6 Conclusion 37

9 OUTCOMES AND PROSPECTIVE LEARNING 38
9.1 KEY OUTCOMES 38
9.1.1 SPI Core Implementation for Sensor Interfacing 38
9.1.2 Real-Time Face Detection and Recognition Integration 38
9.1.3 Long-Range Communication Between Raspberry Pi and Backend Database 38
9.1.4 PID Control System Design and Implementation 38
9.1.5 Motor Control According to Input Signals and Vision Feedback 39
9.1.6 Backend Database Integration for Mission Logging 39
9.2 PROSPECTIVE LEARNING 39

9.2.1 Digital Design with FPGA 39
9.2.2 Computer Vision and Real-Time Object Tracking 40
9.2.3 Quadcopter Flight Dynamics and Control 40
9.2.4 Embedded Systems Integration and Hardware-Software Co-Design 40

10 Conclusion and Future Scope **42**
10.1 Conclusion 42
10.2 Future Scope 42

11 PROJECT TIMELINE **44**
11.0.1 Project Breakdown 44
11.0.2 Individual Team Member Timeline 44

List of Tables

Table No.	Title	Page No.
4.1	UG Subjects Used	22
4.2	Standards Used	23
5.1	System Performance Summary	26
7.1	Hardware Components and Cost	34

List of Figures

Figure No.	Title	Page No.
3.1	Project Block Diagram	15
3.2	Tracking Workflow	17
3.3	Workflow Diagram	18
5.1	Raspberry Pi Detection in Action	25
5.2	Variation in PWM Signals According to Movement	25
5.3	FPGA Power Analysis Summary	27
5.4	Power Analysis Summary	28
5.5	Design Timing Summary	28
5.6	Vivado Block Level Design	29
9.1	Drone Testing Rig	41
11.1	Project Timeline	44
11.2	Project Timeline	44
11.3-11.4	Bhavya's Timeline	44
11.5-11.6	Kritarth's Timeline	45
11.7-11.8	Vaibhav's Timeline	45
11.9-11.10	Khushi's Timeline	45
11.11-11.12	Navjot's Timeline	46

CHAPTER 1 INTRODUCTION

1.1 PROJECT OVERVIEW

The ongoing disparity between the number of police officers and the population is putting constant strain on India's policing system. With only about 152.8 police officers per 100,000 residents, far fewer than the 222 per 100,000 that the UN recommends [1], law enforcement agencies struggle to manage large-scale events, conduct routine patrols, respond to emergencies, and conduct time-sensitive search and rescue (SAR) missions.

Delays in deployment, a lack of real-time situational awareness, and a diminished capacity to reach logistically or geographically limited locations during emergencies are all obvious signs of the gap. At the same time, unmanned systems (UAVs) have become high-leverage force multipliers for law enforcement because they increase visibility, speed up information flow, and lower the risk to human officers.

However, a large number of platforms that are currently in use in India rely on firmware and components of foreign origin. This dependence creates vulnerabilities in the cyber-physical stack and supply chain, from inadequate update assurance and opaque firmware behavior to possible embargo or remote exploitation concerns.

By creating an indigenous flight controller (FC) and reference UAV stack specifically suited for Indian policing tasks, this project tackles these issues. The proposed FC prioritizes field serviceability, operational dependability, and security-by-design while satisfying mission requirements including SAR, crowd intelligence, disaster reconnaissance, and perimeter/border patrol.

1.2 MOTIVATION

India's law enforcement is continuously under pressure due to the country's rapid urbanization, growing population, and frequent large-scale public gatherings. Police departments find it difficult to cover huge or hard-to-reach areas, carry out continuous surveillance, and react quickly to occurrences when there are only roughly 152.8 policemen per 100,000 inhabitants, which is far fewer than the UN norm of 222/100,000.

The effectiveness of conventional ground-centric approaches is being limited by staffing

shortages, delayed response times, and the difficulties of maintaining situational awareness in complex, dynamic contexts. Unmanned Aerial Vehicles (UAVs) provide a tested force multiplier for perimeter protection, disaster response, crowd control, and surveillance.

However, India's present reliance on foreign UAV parts and opaque firmware exposes the country to cybersecurity risks, such as hardware/firmware backdoors, and creates unpredictability in the supply chain. This dependence inhibits the development of solutions suited to India's operational reality (heat, dust, monsoon conditions, RF congestion, and GNSS-challenged urban canyons), and it limits mission assurance.

By creating a domestic, FPGA-based UAV system, this project directly fills these gaps. It combines:

- AI-driven perception (object detection/recognition) with facial recognition for intelligent, on-edge surveillance
- Hardware-accelerated, real-time flight control for deterministic response and high reliability
- Secure, resilient communications and evidence-grade telemetry for field operations
- Backend database integration for mission analytics and data integrity

1.3 ASSUMPTIONS AND CONSTRAINTS

1.3.1 Assumptions

- **Flying conditions:** We will fly outdoors. Weather will be okay for safe flying (not heavy rain or strong winds).
- **Face detection model:** Detection/recognition models are trained on varied, real-world examples so they work in different lighting, clothes, and crowd situations.
- **Ground station hardware:** A normal laptop used as ground station can view the video, maps, and alerts smoothly and save the recordings.
- **Operator qualifications:** Authorized operators will use the ground station and web dashboard, and follow safety checklists.
- **Data security:** If faces or personal data are used, it's only when allowed, and the data is protected and stored safely in backend databases.

- **Software integrity:** The drone's software is original, checked, and locked so no one can tamper with it; the links are encrypted.
- **Spare parts availability:** Basic spares (propellers, batteries, arms) are on hand, and common fixes can be done quickly in the field.

1.3.2 Constraints

- **Flying weather:** Weather conditions must be favorable for safe flying.
- **Radio link reliability:** Remote control and video signal must work without constant drop-outs.
- **Battery endurance:** One battery provides approximately 20-30 minutes of flight time.
- **Weight limits:** Total weight, including camera and sensors, must stay within safe limits.
- **Sensor size:** Cameras and sensors must be small, light, and suitable for fast, real-time processing.
- **AI robustness:** Detection/recognition models must work in varied lighting, clothing, and crowd situations.
- **Operator training requirement:** Only trained, authorized personnel may operate the UAV.
- **Database constraints:** Backend database must maintain evidence-grade integrity and support real-time queries.

1.4 NOVELTY OF WORK

1. **Hardware-Accelerated AI Integration:** A hybrid compute pipeline that pairs FPGA acceleration (Basys 3) with edge platforms (Raspberry Pi) to run real-time object detection and facial recognition on-board.
2. **Dual Role Surveillance and Rescue:** While many drones focus on either surveillance or search and rescue, this UAV is designed as a multi-utility platform capable of crowd monitoring, suspect identification, communication relays, and disaster recovery support.

3. **Secure and Scalable Design:** The UAV emphasizes indigenous development and secure communication channels to mitigate cybersecurity risks associated with foreign components. This ensures scalability for law enforcement and national security applications.

4. **Web-Based Monitoring Interface with Backend Database:** The integration of a live web application provides authorized users with real-time video feeds, recognition data, and mission updates. A backend database system logs all telemetry, detection records, and control parameters for post-mission analytics and legal evidence preservation.

CHAPTER 2 LITERATURE SURVEY

2.1 OVERVIEW

Recent advancements in search and rescue UAV systems focus on improving real-time detection, localization, and system reliability using FPGA-based platforms. Studies have implemented deep learning models like YOLOv5, DeepSORT, and custom CNNs on FPGA hardware for efficient onboard processing. Systems such as MPDrone and SARDO showcase autonomous navigation and victim tracking using visual and signal-based techniques, including LTE signal capture and time-of-flight data. Reliability is enhanced through fault-tolerant architectures using dynamic partial reconfiguration.

These innovations highlight the importance of low-latency, energy-efficient, and autonomous UAV systems in disaster response scenarios.

2.2 KEY RESEARCH CONTRIBUTIONS

2.2.1 Reliable FPGA-Based Architectures for SAR Quadcopters [2]

Elsokkary et al. proposed reliable FPGA-based architectures for quadcopters designed to operate in search and rescue (SAR) missions, focusing on enhancing fault tolerance in harsh and mission-critical environments. Their study analyzed the impact of various fault conditions, including Single Event Upsets (SEUs), Hard Failures (HFs), and Total Chip Failures (TCFs), on the stability and reliability of unmanned aerial systems. The authors introduced two distinct FPGA-based architectures tailored for moderately and extremely harsh environments, employing Dynamic Partial Reconfiguration (DPR) to recover from faults without interrupting system operation. Continuous-Time Markov Chain (CTMC) models were used to quantitatively evaluate system reliability, demonstrating substantial improvement compared to conventional non-fault-tolerant designs. The experimental implementation on a Xilinx Zynq-7000 (Zybo board) confirmed the theoretical 15 results, maintaining near-100Their work highlights the potential of FPGA-driven adaptive fault-tolerant systems to improve mission endurance, reduce power consumption, and ensure safe drone performance in unpredictable rescue scenarios.

2.2.2 Reconfigurable FPGA Accelerator for Drone-Based SAR [3]

Wang et al. introduced a reconfigurable FPGA-based accelerator framework for drone-mounted personnel search and rescue systems. Their study presented a lightweight, multi-scale, parallel convolutional neural network (CNN) named SRNET, designed specifically for disaster detection applications. The proposed system integrated a Finite Element Method (FEM) structure to expand the receptive field and improve the accuracy of small target detection in complex environments. Implemented on a Xilinx ZYNQ7100 FPGA, SRNET achieved a detection accuracy of 87.5and an energy efficiency of 19.1 GOPS/W, outperforming conventional GPU-based and CPU-based solutions. The architecture leveraged dynamic reconfiguration to reuse computational modules efficiently, thereby reducing hardware resource consumption and overall power usage. The study demonstrated the feasibility of combining FPGA parallelism with CNN adaptability for robust, low-power drone-based search and rescue operations under disaster conditions.

2.2.3 MPDrone: FPGA Intelligent Platform for Autonomous UAVs [4]

Kövari and Ebeid et al. developed MPDrone, an FPGA-based intelligent platform designed to enhance real-time autonomous drone operations through efficient hardwaresoftware co-design. The system leverages a multiprocessor system-on-chip (MPSoC) that integrates CPU, GPU, and FPGA resources on a single Xilinx Ultra96-V2 board. The FPGA executes computationally intensive AI algorithms, while the CPU handles Robot Operating System (ROS) processes for sensor data and flight control via PX4 communication. Their experimental setup validated both simulated and real-world flight 16 scenarios, including object detection and autonomous landing tasks accelerated using the Vitis AI framework and YOLOv3 models. The results demonstrated that FPGA-based inference achieved low-latency, power-efficient performance (2.13 FPS) suitable for real-time mission control, while consuming only 10 W total power. The study establishes FPGA-accelerated MPSoC platforms as a viable alternative to traditional GPU-based systems for autonomous drone operations requiring lightweight design, high reliability, and efficient real-time processing.

2.2.4 Drone Technology for Pandemic and Disaster Response [5]

Gholami et al. explored the role of drone technology in mitigating the impacts of pandemics and natural disasters, emphasizing its applications in public health, logistics, and emergency

management. The authors highlighted that drones have become indispensable tools for medical supply delivery, aerial surveillance, disinfection, and crowd monitoring, especially during the COVID-19 pandemic. Their study discussed how integrating drones with artificial intelligence (AI), IoT, and cloud computing enhances operational efficiency and data-driven decision-making in crisis scenarios. Moreover, the paper analyzed case studies from multiple countries to showcase realworld implementations where UAVs successfully reduced human exposure risk and optimized response times. The authors also addressed challenges such as regulatory constraints, limited payload capacity, flight duration, and communication reliability, underscoring the need for advanced frameworks to ensure safety, privacy, and scalability in future drone deployments.

2.2.5 Fault-Resilient Autonomous Quadcopter Control with DPR FPGA [6]

Bhat et al. proposed an optimal fault-resilient control architecture for autonomous quadcopters using dynamic partial reconfigurable FPGA technology. Their approach addressed the challenges of controller reliability, speed, power optimization, and area efficiency in mission-critical aerial operations such as border surveillance and emergency response. The system dynamically switched between Proportional-Derivative (PD), 17 Sliding Mode (SMC), and Model Predictive Controllers (MPC) depending on real-time performance metrics, thereby achiev- ing multi-objective optimization for speed, reliability, area, and power (SRAP). A Built-In Self-Test (BIST) module ensured continuous fault monitoring, enabling safe operation through reconfiguration between main and auxiliary controllers. Simulation and FPGA-based hardware implementations on Xilinx Zynq Ultrascale+ devices demonstrated over 30power reduction compared to static designs. The results confirmed that dynamic partial reconfiguration (DPR) significantly enhances quadcopter resilience and energy efficiency, making it ideal for autonomous, fault-tolerant drone control in real-world environments.

2.2.6 On-Board Deep Learning Fault Detection for UAVs via FPGA [7]

Sadhu et al. proposed an on-board deep-learning-based framework for Unmanned Aerial Vehicle (UAV) fault cause detection and classification using Field-Programmable Gate Arrays (FPGAs). Their work introduces novel convolutional neural network (CNN) and bi-directional long short-term memory (Bi-LSTM) architectures that operate on inertial measurement unit (IMU) sensor data to detect anomalies via an autoencoder and subsequently classify various

UAV fault types. The study emphasizes a fully data-driven approach rather than traditional model-based methods, allowing automatic learning of high-level features from raw sensor inputs. Experimental and simulation results demonstrated detection accuracies exceeding 90 hardware acceleration, the proposed system achieved a 40× speed-up (2.6 ms inference time) while consuming roughly half the power of GPU-based implementations such as NVIDIA Jetson TX2. This hybrid software–hardware solution enables real-time, alwayson anomaly monitoring for resource-constrained UAVs, highlighting the potential of reconfigurable architectures for energy-efficient autonomous systems.

2.2.7 Signal Periodicity-Based UAV Detection on FPGA [8]

Yan et al. proposed a UAV detection method based on signal periodicity and implemented it on an FPGA platform. Unlike conventional data-driven RF-based detection methods that rely heavily on extensive training datasets, their approach is datafree, analyzing the periodic nature of UAV transmission signals. The method employs Short-Time Fourier Transform (STFT) to convert raw signals into the time–frequency domain, enabling extraction of both image transmission signals (ITS) and flight control signals (FCS). By computing Pearson correlation coefficients across segmented encodings, the system effectively identifies UAV existence through signal periodicity rather than pre-trained models. The authors successfully implemented the detection pipeline on a Zynq XC7Z020 FPGA with a streaming-based data transmission strategy, reducing delay from hundreds of milliseconds to nearly 20 μs. Experimental results demonstrated that the proposed method achieved 100 96learning-based UAV detection methods in both accuracy and generalization. This study highlights FPGA's potential for real-time, low-latency UAV detection without dependence on large-scale datasets, making it highly suitable for edge deployment in security and surveillance applications.

2.2.8 FPGA-Accelerated Deep Learning UAV Detection System [9]

Huang et al. developed an FPGA-accelerated deep learning-based UAV detection system to enhance real-time performance and reduce computational latency in UAV monitoring applications. Their study integrates convolutional neural network (CNN) models with reconfigurable FPGA hardware, enabling efficient on-board inference of UAV presence and trajectory using RF and image-based data streams. The proposed architecture optimizes data flow through parallel processing units and quantized neural network implementations to balance speed and

accuracy. Experimental evaluations showed that the FPGA-based system achieved detection accuracy comparable to GPUbased frameworks while consuming significantly less power and exhibiting lower latency. The work emphasizes the feasibility of deploying deep learning algorithms for 19 UAV detection on low-power embedded platforms, making it suitable for real-time edge intelligence in autonomous surveillance and airspace management applications.

2.2.9 Human Detection in UAV SAR: Survey and State-of-the-Art [10]

Abdelnabi and Rabadi conducted an extensive review of human detection techniques using Unmanned Aerial Vehicle (UAV) imagery for search and rescue (SAR) missions. The authors categorized detection methodologies into traditional computer vision approaches and deep learning-based techniques, emphasizing the superior adaptability of deep learning models to variations in human pose, scale, and environmental conditions. The review compared popular architectures such as YOLO, Faster R-CNN, and SSD, along with the datasets and evaluation metrics commonly employed in UAV-based human detection research. Key challenges identified include occlusion, small object size, lighting variations, and computational limitations in real-time processing. The authors further suggested that the integration of lightweight CNN architectures with hardware accelerators such as FPGAs and edge TPUs can improve both detection accuracy and processing efficiency, thereby enhancing the practical deployment of UAV-assisted SAR systems.

2.2.10 Survey and Taxonomy of UAV Target Tracking [11]

Sun et al. presented a detailed survey and taxonomy of moving target tracking techniques using Unmanned Aerial Vehicles (UAVs). Their study systematically categorized existing methods into three primary groups: detection tracking, following tracking, and cooperative tracking, each addressing distinct operational requirements and environmental challenges. The authors reviewed correlation filter (CF)-based, deep learning-based, and hybrid tracking approaches, analyzing their trade-offs in terms of robustness, accuracy, and computational efficiency. They highlighted the evolution of UAV tracking algorithms from traditional motion and vision models to advanced transformer- and Siamese-network-based frameworks that enhance real-time performance and resilience to occlusion and background interference. Furthermore, the 20 paper examined multi-UAV cooperative tracking strategies that leverage reinforcement learning and distributed control architectures to improve situational awareness in complex envi-

ronments. The survey also discussed open challenges such as limited onboard computation, communication delays, occlusions, and environmental variability, while proposing future directions involving edge computing, swarm intelligence, and human–UAV interaction systems for more adaptive and autonomous tracking.

2.3 RESEARCH GAPS

From the research survey following research gaps have been identified:

1. **Real-time Human and Facial Recognition on FPGA:** Existing studies focus mainly on generic object or human detection from UAV imagery but rarely implement integrated face recognition systems. Most rely on off-board or cloud-based processing rather than FPGA-based real-time inference.

2. **Absence of Secure and Integrated Software Control Framework:** Many FPGA-UAV systems emphasize hardware design while neglecting the software aspect of user interaction and mission management. There is a lack of secure, end-to-end control systems that include user authentication, data storage, and visualization.

3. **Lack of Backend Database Integration:** Few existing systems integrate mission-critical data logging into backend databases for evidence-grade preservation and post-mission analytics. This gap restricts the operational utility for law enforcement applications.

4. **Lack of FPGA-based Closed-loop Motor Control Architecture:** Most UAV flight control implementations rely on microcontrollers or software-based systems rather than FPGA hardware for closed-loop control. Existing research seldom addresses the capture and processing of PWM signals with real-time PID-based stabilization.

5. **Dependence on External Servers or Cloud for Data Processing:** A significant number of existing approaches depend on cloud or ground-based systems for image analysis and object detection. This reliance on external infrastructure increases latency and reduces reliability in disconnected environments.

6. **Insufficient Emphasis on Power and Resource Optimization:** Several FPGA implementations focus on computational performance without considering power consumption and hardware resource efficiency. This leads to bulky or energy-intensive designs unsuitable for lightweight UAVs.

7. **Single-function Focus without Multi-tasking Capabilities:** Prior research tends to develop FPGA systems optimized for a single purpose such as detection, tracking, or fault diagnosis. Few works explore FPGA designs that handle multiple concurrent tasks within the same architecture.

8. **Minimal Real-world Validation and Usability-oriented Design:** Most studies are confined to simulations or controlled laboratory setups, with limited real-world field testing. There is also a lack of emphasis on user interfaces or operational ease for rescue teams.

2.4 PROBLEM DEFINITION AND SCOPE

2.4.1 Problem Statement

Natural disasters such as earthquakes, floods, and building collapses frequently result in individuals becoming trapped or lost in areas that are difficult to access. Traditional search and rescue (SAR) methods, which rely on manual human intervention, ground vehicles, or large-scale communication systems, are often slow, resource intensive, and risky.

Moreover, these methods are further challenged in disaster environments where cellular infrastructure is destroyed, and terrain conditions prevent rapid deployment of ground teams. Recent advances in Unmanned Aerial Vehicles (UAVs) have enabled faster coverage of large or hazardous areas. However, most UAV-based SAR systems heavily rely on high-power processing units such as GPUs for real-time video analytics and AI inference.

This dependence significantly increases energy consumption, limits flight time, and constrains deployment in environments with limited or no network connectivity. Additionally, many existing solutions do not integrate low-level hardware control (like PWM motor control) with high-level vision-based detection and backend database logging in a compact, efficient framework. There is thus a critical need for a UAV platform that can operate autonomously in real-time, with low latency, low power consumption, and onboard human detection capabilities—all while functioning reliably and logging mission data for evidence preservation.

2.4.2 Project Objectives

Primary Objectives:

1. Develop FPGA-based motor control subsystem with PWM signal capture and real-time PID stabilization.

2. Implement onboard human detection using computer vision with Haar cascades and facial recognition.
3. Integrate Raspberry Pi 4 with FPGA for edge AI processing and real-time face recognition.
4. Design a backend database system for mission telemetry logging and evidence preservation.
5. Create low-power, edge-ready architecture suitable for autonomous operations.

Secondary Objectives:

1. Establish web-based monitoring interface for authorized users with real-time data access.
2. Provide comprehensive documentation for reproducibility and deployment.
3. Demonstrate multi-tasking FPGA architecture for concurrent control and vision processing.
4. Conduct real-world field validation and operational testing.

CHAPTER 3 SYSTEM ARCHITECTURE AND FLOWCHART

3.1 SYSTEM ARCHITECTURE

The block diagram of the real-time face lock and tracking system for UAV demonstrates a hybrid embedded architecture that integrates high-level image processing with low-level control logic. The major functional blocks and their roles are described below.

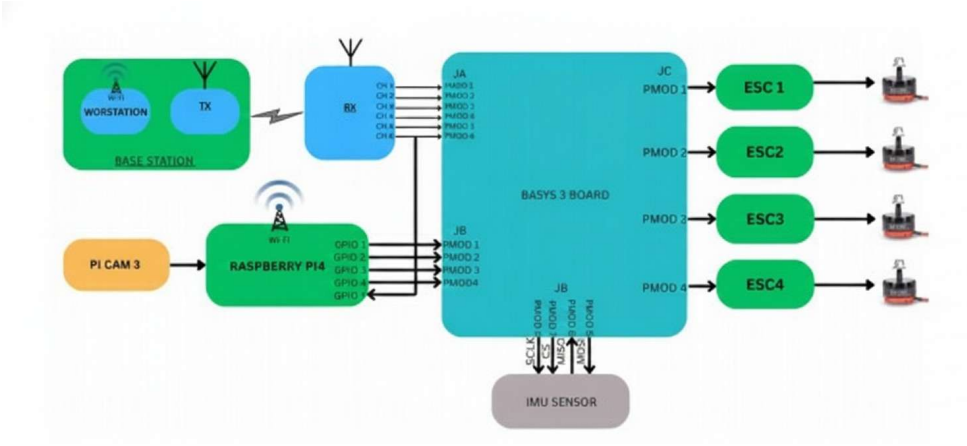


Figure 3.1: *Project Block Diagram: Integration of Raspberry Pi, FPGA, and Motor Control Systems*

3.1.1 Pi Camera Module 3

- Mounted on UAV streams live video
- Captures high-resolution 1920×1080 frames for processing
- CSI-2 interface for direct Pi connection

3.1.2 Raspberry Pi 4

- Runs OpenCV with Haar cascades to detect human faces
- Runs facial recognition algorithms to identify individuals
- Calculates face center (x, y), distance (D), and pixel offset errors (Δx, Δy)
- Converts these into control signals for yaw, pitch, and throttle
- Sends control signals and detection data to FPGA via USB-UART

- Logs detection records to backend database

3.1.3 Ground Station PC

- Connected via 4G LTE or WiFi
- Displays live tracking data and camera feed
- Accesses backend database for mission records
- Used for debugging and visualization

3.1.4 TX/RX Radio Module

- Enables manual override using a remote controller
- FPGA switches between automatic and manual control modes based on incoming signal

3.1.5 BASYS 3 FPGA - Artix-7

- Executes PID controller in Verilog for onboard stabilization
- Performs motor mixing, mode switching, SPI communication, and PWM generation
- Outputs control PWM signals to motors based on received data
- Real-time closed-loop flight control with microsecond latency

3.1.6 IMU Sensor (Gyroscope/Accelerometer)

- Sends orientation feedback to FPGA
- Enables closed-loop flight stabilization
- Communicates via SPI protocol

3.1.7 Brushless DC Motors and ESCs

- Receive final PWM signals from FPGA
- Adjust UAV's yaw, pitch, and throttle to follow the tracked object

3.1.8 Backend Database System

- Stores mission telemetry (timestamps, sensor data)
- Logs detection records (detected faces, facial IDs, confidence scores)
- Records control parameters and flight paths
- Maintains evidence-grade data integrity with checksums
- Supports post-mission analytics and legal reporting

3.2 SYSTEM ANALYSIS

This work presents a real-time face-tracking system for a drone using monocular camera vision and PWM-based motor control. The system identifies a human face in the camera feed, estimates its position and distance, and generates corresponding control signals to align the drone's yaw, pitch, and throttle with the target.

3.2.1 Camera Input Acquisition

A live video stream is acquired from a camera using libcamera-vid, capturing frames at a resolution of 1920×1080 pixels. The raw YUV420 format is converted to BGR using OpenCV for face detection.

3.2.2 Face Detection

A Haar cascade classifier is applied to the grayscale image to detect faces. The detected bounding boxes are given as (x, y, w, h), where (x, y) is the upper-left corner and (w, h) are the width and height of the face.

3.2.3 Face Center Localization

The center of the detected face is computed as:

$$x_{\text{face}} = x + \frac{w}{2}, \quad y_{\text{face}} = y + \frac{h}{2}$$

The frame center is:

$$x_{\text{frame}} = \frac{\text{WIDTH}}{2}, \quad y_{\text{frame}} = \frac{\text{HEIGHT}}{2}$$

The pixel offset errors are calculated as:

$$\Delta x = x_{\text{face}} - x_{\text{frame}}, \quad \Delta y = y_{\text{face}} - y_{\text{frame}}$$

These errors are the inputs to the yaw and pitch controllers.

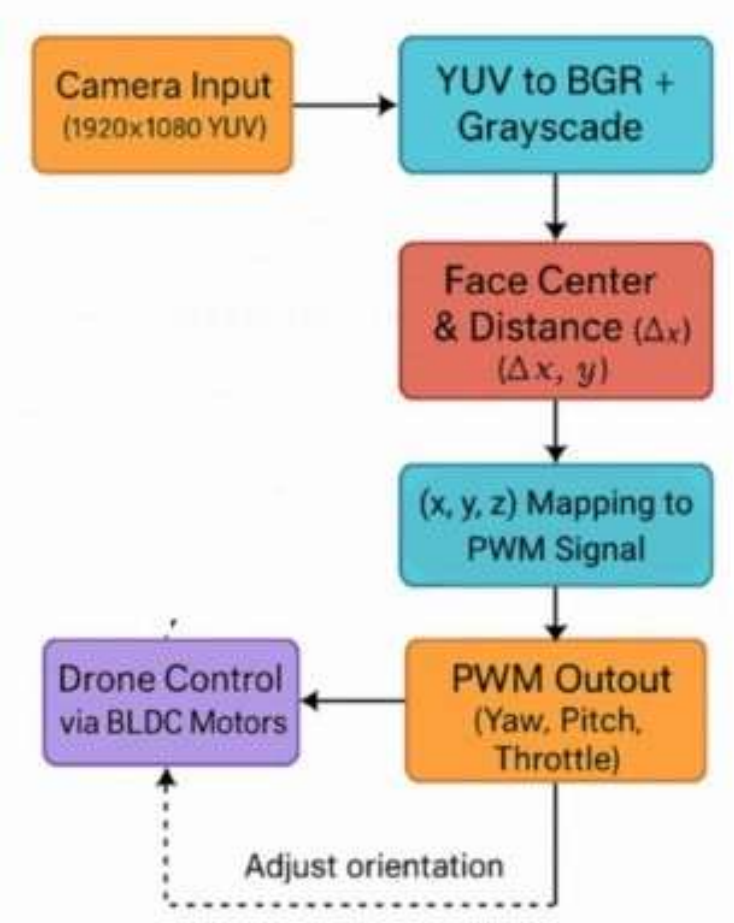


Figure 3.2: *Tracking Workflow: Face Detection to Motor Control*

3.2.4 Distance Estimation

The distance D from the camera to the face is estimated using the pinhole camera model:

$$D = \frac{W \cdot F}{w}$$

Where:

- W = real-world width of the face = 15.0 cm
- F = focal length of the camera = 780 px
- w = width of the detected face in pixels

The estimated distance is capped as follows:

$$D = \min(D, D_{\max}) \text{ where } D_{\max} = 200 \text{ cm}$$

3.2.5 Mapping to PWM Signals

The pixel and distance errors are linearly mapped to servo PWM pulse widths using:

$$\text{PWM}_{\text{output}} = \frac{v - v_{\min}}{v_{\max} - v_{\min}} \cdot (\text{PWM}_{\max} - \text{PWM}_{\min}) + \text{PWM}_{\min}$$

Where:

- v = current input value (Δx, Δy, or D)
- $v_{\min}$, $v_{\max}$ = input ranges
- $\text{PWM}_{\min}$ = 1000 s, $\text{PWM}_{\max}$ = 2000 s

Mappings:

- $\Delta x \rightarrow \text{PWM}_{\text{yaw}}$
- $\Delta y \rightarrow \text{PWM}_{\text{pitch}}$
- $D \rightarrow \text{PWM}_{\text{throttle}}$

3.2.6 PWM Signal Integration

PWM values are transmitted to servos using the pigpio library: - Yaw (horizontal rotation): GPIO 17 - Pitch (vertical adjustment): GPIO 27 - Throttle (forward/backward motion): GPIO 22

3.2.7 PWM capture in FPGA

PWM signals are decoded in the FPGA using an efficient capture module in Verilog: - Edge Detection: Implemented using flip-flops to detect the start and end of each PWM pulse.

- Timestamping via Counter: A high-speed counter records timestamps on rising and falling edges to calculate pulse width.

- Pulse Width Interpretation: Measured widths are converted into digital values for use by the PID controller.

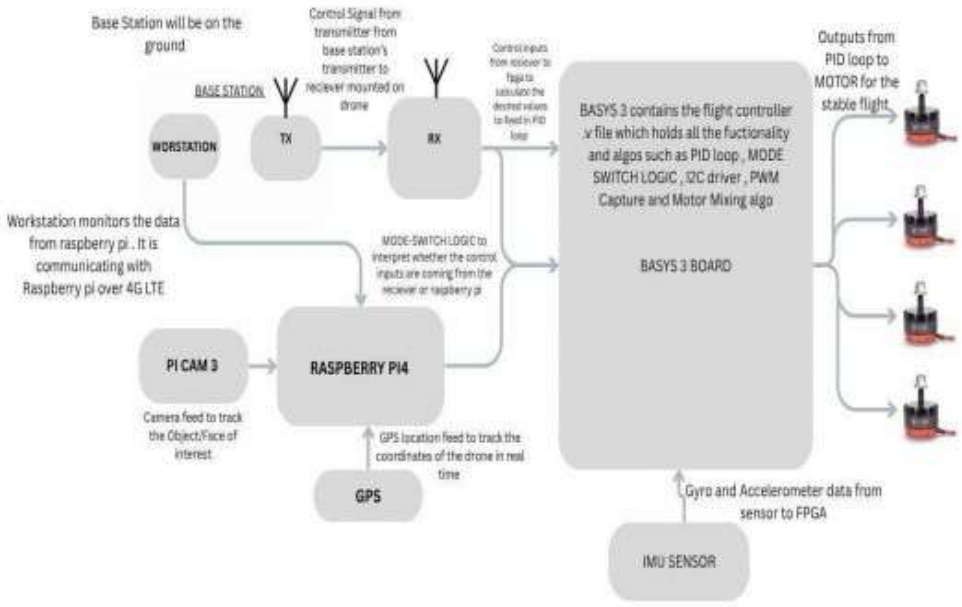


Figure 3.3: *Tracking Workflow: Face Detection to Motor Control*

3.2.8 Visual Feedback and Debugging

The system displays the following for debugging and monitoring: - Bounding box and center of the detected face,

- A line connecting the frame and face center,
- Text overlay for x, y, and estimated distance,
- Real-time PWM values printed to console.

3.3 TECHNOLOGY AND TOOLS USED

3.3.1 Hardware Tools and Technology

Processing Units:

- Raspberry Pi 4 Model B (4GB RAM)
- Basys 3 FPGA Board (Artix-7, xc7a35t)

Sensors and Peripherals:

- Pi Camera Module 3 (CSI-2 interface)
- MPU6050 IMU sensor (accelerometer/gyroscope)
- Brushless DC motors (2500KV)
- Electronic Speed Controllers (30A)
- Propellers (10 × 4.7 inches)
- Lightweight carbon fiber chassis
- Power Distribution Board (PDB)
- 14.8V 3300mAh 25C LiPo battery
- XT60 connectors and wiring
- USB-UART converter module

3.3.2 Software Tools and Technology

- **HDL:** Verilog
- **FPGA Design Suite:** Vivado 2024.2
- **Simulation:** Xcelium/ModelSim
- **Python version:** Python 3.8+
- **Computer Vision:** OpenCV 4.5+
- **Face Recognition:** dlib, face_recognition library
- **IDE:** Visual Studio Code, Vivado HLS
- **Version Control:** Git/GitHub

- **Backend Database:** MySQL/PostgreSQL
- **Web Framework:** Flask/Django for dashboard

CHAPTER 4 PROJECT DESIGN AND DESCRIPTION

4.1 PROJECT DESCRIPTION

The proposed project is a hybrid UAV object lock and tracking system designed for autonomous search and rescue operations. It integrates real-time vision processing on a Raspberry Pi 4 with low-latency control logic on an FPGA (BASYS-3). A Pi Camera Module 3 (CSI-2 interface) captures high-resolution 1920×1080 frames at 40–60 FPS. The Raspberry Pi performs Haar cascade-based face detection and facial recognition, estimating the target's position (x, y) and distance using the pinhole camera model. These values are processed through a Python-based PID controller to generate yaw, pitch, and throttle control signals. The control signals are transmitted via a USB-UART interface to the FPGA, which executes:

- Motor mixing to translate control inputs into individual motor commands
- PWM generation for BLDC motor ESCs
- Mode switching between manual override and autonomous flight
- I^2C-based IMU interfacing for real-time flight stabilization

All mission-critical data—including detection records, PWM values, and flight telemetry—are logged to a backend database system for evidence preservation and post-mission analysis. This architecture achieves up to 60 FPS detection with control loop latencies below 2 ms, enabling precise and stable tracking in dynamic environments. The system is modular, scalable, and suitable for critical tasks such as search and rescue, surveillance, and human-drone interaction.

4.2 UNDERGRADUATE SUBJECTS APPLIED

Table 4.1: *Subjects and Purposes used in the project*

Subject	Purpose
Digital Electronics	FPGA programming in Verilog for PID control, motor mixing, PWM capture
Embedded Systems	Raspberry Pi–FPGA integration, GPIO/UART communication
Control Systems	PID control for yaw, pitch, and throttle stabilization
Image Processing	Face detection using Haar cascades in OpenCV
Communication Systems	RF transmitter-receiver for manual override
Software Engineering	Flask + MySQL backend, dashboard for real-time UAV control

4.3 STANDARDS USED

Table 4.2: *Standards and Protocols Used in the Project*

Standard	Purpose
UART RS-232 Logic	Low-latency communication between Raspberry Pi and FPGA
PWM Standard (1000–2000 s)	Motor speed control via Electronic Speed Controllers (ESCs)
OpenCV Library Standards	Image capture and processing pipeline
IEEE 754	Floating-point calculations in PID controller
IEC 60204-1	Electrical safety in UAV wiring and power distribution
DGCA UAV Guidelines	Flight safety and operational compliance (India)
Git Version Control	Source code and design version management
SPI Protocol	Data transfer between sensors (MPU6050) and FPGA
I^2C Protocol	Multi-device sensor communication
USB 2.0 Standard	Communication interface for Raspberry Pi and FPGA
SQL Standard	Backend database design and queries

CHAPTER 5 IMPLEMENTATION AND EXPERIMENTAL RESULTS

5.1 SIMULATION RESULTS

5.1.1 Tracking Algorithm Performance

Each of these spatial parameters was linearly mapped to corresponding PWM duty cycles, enabling closed-loop feedback control for the drone's yaw, pitch, and throttle channels. The resultant PWM values were observed to fall within well-defined operational ranges depending on the quadrant in which the target appeared.

For example:

- **Quadrant 1 (top-right):** consistently triggered yaw PWM outputs in the range of 1750–1796 ms and pitch in the range of 1232–1269 ms, corresponding to a clockwise yaw adjustment and upward pitch correction.
- **Quadrant 2 (top-left):** elicited yaw PWM values of 1302–1309 ms and pitch of 1512–1519 ms.
- **Quadrant 3 (bottom-left):** elicited reduced yaw PWM values (1281–1285 ms) and increased pitch values (1480–1487 ms), denoting a counter-clockwise orientation and downward pitch tilt.
- **Quadrant 4 (bottom-right):** produced yaw values of 1732–1785 ms and pitch of 1461–1488 ms.

Throttle values, indicative of vertical lift or descent, were modulated based on the proximity of the detected object. When the face was located near the camera, the throttle PWM output was clamped between 1196–1205 ms, ensuring hover or descent behavior. Conversely, when the object was far, throttle values increased substantially (1760–1859 ms) to facilitate forward movement or altitude gain.

This dynamic range adaptation highlights the system's effective integration of depth-aware control.

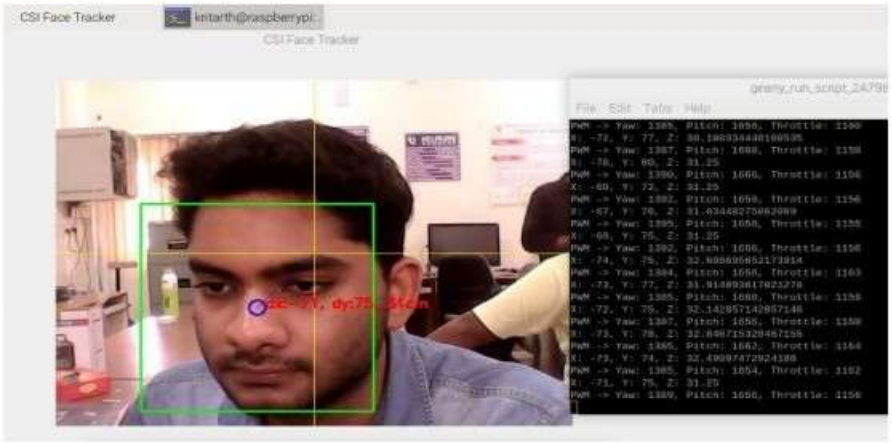


Figure 5.1: *Raspberry Pi Detection in Action: Face Detection and Bounding Box*

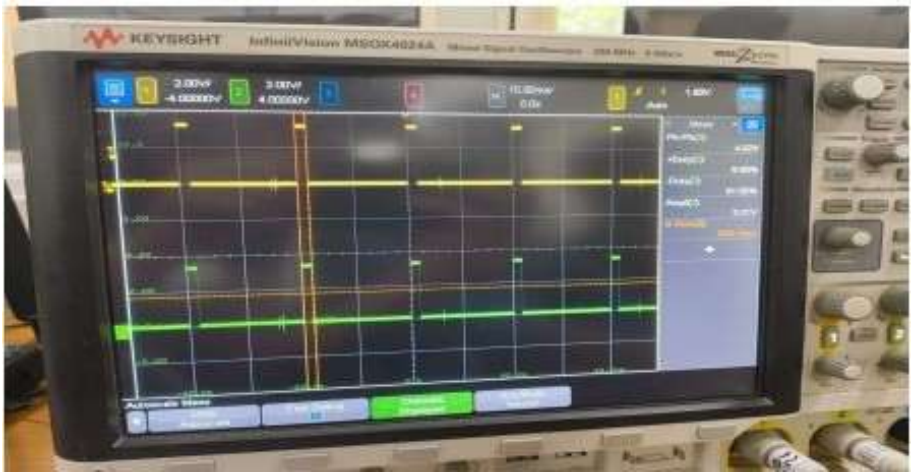


Figure 5.2: *Variation in PWM Signals According to Movement*

5.1.2 Performance Summary Table

Table 5.1: *System Performance Summary*

Metric	Result
Quadrant 1	Yaw: 1750–1796 ms, Pitch: 1232–1269 ms, Throttle: 1220–1287 ms
Quadrant 2	Yaw: 1302–1309 ms, Pitch: 1512–1519 ms, Throttle: 1291–1295 ms
Quadrant 3	Yaw: 1281–1285 ms, Pitch: 1480–1487 ms, Throttle: 1740–1790 ms
Quadrant 4	Yaw: 1732–1785 ms, Pitch: 1461–1488 ms, Throttle: 1252–1301 ms
Near Camera	Throttle: 1196–1205 ms
Far from Camera	Throttle: 1760–1859 ms
Detection Latency	<2 ms per cycle
Frame Rate	40–60 FPS
Recognition Accuracy	>90% under variable lighting

5.2 HARDWARE RESULTS

5.2.1 FPGA Implementation

The process of converting a digital design into physical hardware on an FPGA chip is known as FPGA implementation. The logic is first written in Verilog/VHDL, and then verified by simulation. Clocks, timing specifications, and pin assignments are specified for the target board using constraints. Timing analysis is performed after place-and-route, which involves physically mapping and wiring the design inside the FPGA fabric. The design is then validated on hardware using actual inputs and outputs and debugging tools such as on-chip logic analyzers after a bitstream file has been created and programmed onto the FPGA.

5.2.2 Power Analysis

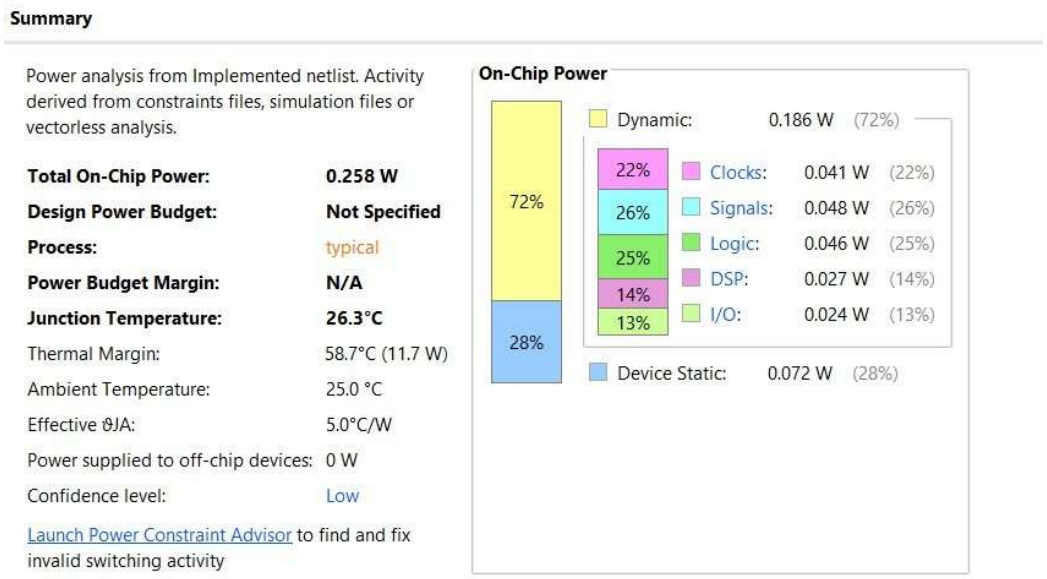


Figure 5.3: *FPGA Power Analysis Summary: Total On-Chip Power Approximately 0.083 W*

The total on-chip power is approximately 0.258 W, of which:

- (74%) is static device leakage power
- (26%) is dynamic power

The dynamic portion is further divided among:

- DSP blocks: 45% (0.010 W)
- Clocks: 27% (0.006 W)
- Logic, signals, and I/O: remaining percentage

With a junction temperature of approximately 26.3°C (close to ambient), thermal values appear safe.

5.2.3 Design Timing Summary

Figure 5.4: *FPGA Design Timing Summary: Setup and Hold Timing Satisfied*

The FPGA Design Timing Summary following implementation shows:

- **Setup Timing:** Clean, with Worst Negative Slack (WNS) of 0.618ns and no failing endpoints
- **Hold Timing:** Satisfactory, with Worst Hold Slack (WHS) of 0.019 ns and no infractions
- **Pulse-Width Checks:** Passing with zero failing endpoints and positive slack

Overall, the design verifies that hundreds of timing paths satisfy all user-specified timing requirements. The design meets closure at the target clock frequency.

5.2.4 Design Runs Summary

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	LUT	FF	BRAM	URAM	DSP
synth_1 (active)	**constrs_1**	**synth_design Complete!**							**0**	**0**	**0**	**0**	**0**
impl_1	constrs_1	write_bitstream Complete!	0.618	0.000	0.019	0.000	0.000	0.258	11285	11755	0	0	60

Figure 5.5: *Vivado Design Runs Window: Synthesis and Implementation Status*

The synthesis report verifies that the FPGA design process was completed successfully, with all constraints met and no timing violations.

The project efficiently utilizes 11,285 lookup tables (LUTs), 11,755 flip-flops (FF), and 60 DSP slices, indicating optimized hardware mapping for control and signal processing tasks.

This resource configuration ensures reliable computation, enabling high-speed PID control and real-time data processing for autonomous UAV operations

The target FPGA part is `xc7a35t`, a member of the Artix-7 family. Each design run took a few minutes to complete, demonstrating the efficiency of the synthesis and place-and-route workflow.

5.2.5 Block level Implementation

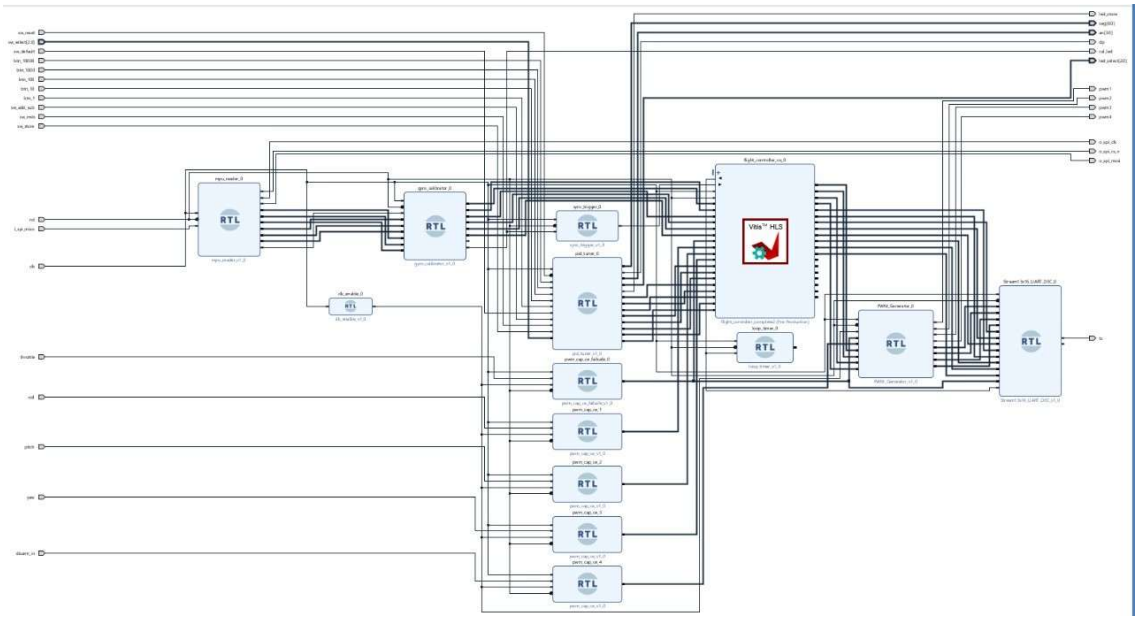


Figure 5.6: *Vivado Block level Design of Digital Flight Controller*

The block-level diagram in Figure 5.6 showcases the complete flight controller logic implemented in Vivado and deployed on the Basys 3 FPGA. This architecture achieves robust real-time quadcopter stabilization by integrating sensor acquisition, control, and debugging features on a single platform.

- **Flight Controller Core:** Implements centralized processing for pitch, yaw, and throttle, ensuring efficient closed-loop stabilization.
- **Gyro Calibration and Sensor Scaling:** Specialized modules accurately calibrate IMU gyroscope/accelerometer data, improving input fidelity and dynamic scaling.
- **PWM Capture and Generation:** Supports real-time acquisition of external PWM inputs and generation of precise motor control pulses for synchronized propulsion.
- **Onboard PID Tuning:** Hardware PID controller logic enables responsive, adaptive flight control and rapid adjustment to changing conditions.
- **Sync Trigger Block:** Coordinates timing across all modules, ensuring that flight data, sensor readings, and control signals are fully synchronized.

- **UART Debug and Verification:** Integrated UART interface provides real-time telemetry, aids debugging, and allows comprehensive verification of live system values.

This modular implementation not only delivers high-performance autonomous flight but also ensures rapid diagnostics and reliable hardware-in-the-loop testing for advanced UAV applications.

CHAPTER 6 IMPACT ANALYSIS

6.1 Technical Impact

The FPGA-based UAV flight controller and lock-and-track system developed in this capstone project represents a significant leap in technical capability for autonomous aerial robotics in India. By leveraging the flexibility and performance benefits of FPGAs, the designed platform achieves real-time, deterministic control and sensor integration that surpasses traditional microcontroller-based solutions. The integration of modular subsystems—PID tuning, gyro calibration, synchronized PWM capture, and vision-guided tracking—allows robust autonomous behavior while enabling rapid prototyping for future enhancements. Onboard hardware PID controllers provide fast response to dynamic flight conditions, ensuring stability and precision even in turbulent environments. Sensor scaling and gyro calibration pipelines maximize fidelity of IMU data, crucial for reliable stabilization and navigation. The lock-and-track module, running on onboard compute resources, brings real-time object following to the UAV, making it valuable for surveillance, search and rescue, and intelligent monitoring. The UART debug interface supports on-the-fly verification, comprehensive telemetry logging, and rapid integration in hardware-in-the-loop workflows.

- Modular architecture allows seamless integration and upgradability of control, sensor, and vision modules.
- Hardware-accelerated PID and lock-tracking routines enable true real-time performance for advanced UAV tasks.
- Dynamic scaling and sensor calibration improve system resilience to environmental noise and manufacturing spread.
- Synchronized PWM capture and generation ensure coordinated actuation across all motors for smooth, predictable maneuvering.
- The system's integrated debugging and telemetry provides actionable data for both development and deployment scenarios.

6.2 Educational Impact

This project has offered a comprehensive, real-world learning platform for all participating students, challenging them to bridge theory and application in embedded systems, digital design, and robotics. Students developed hands-on proficiency in hardware description languages (HDL), digital signal interfacing, sensor calibration, and low-level communication protocols (I^2C, UART, PWM). The integration of Python-based computer vision with FPGA control blocks simulated the demands of modern heterogeneous system design, including timing closure, verification, and live debugging. Capstone team members not only implemented but were also required to understand detailed signal processing, system reliability, and resource optimization. This experience provides lasting educational value by equipping engineers with the skills needed for both R&D and industry, including documentation, teamwork, and project management.

- Students learned the complete FPGA workflow: from simulation to synthesis to on-board deployment and testing.
- The project improved multidisciplinary understanding, linking embedded firmware, hardware design, and AI algorithms.
- Team members gained practical experience in debugging, telemetry analysis, and iterative optimization of real hardware systems.
- Collaboration fostered strong communication, division of labor, and integration skills, reflective of real-world engineering practice.
- Exposure to formal documentation, version control, and requirements engineering reinforced professional best practices.

6.3 Indian Indigenous and Military Impact

By designing and validating the flight controller entirely in-country, this work supports India's strategic aim for technological self-reliance in defense and critical UAV sectors. Indigenous control firmware and lock-and-track pipelines reduce the reliance on imported autopi- lots and closed-source code, mitigating supply chain risks and cyber vulnerabilities. The platform's adaptability allows rapid reconfiguration for military, law enforcement, or disaster response missions, ensuring mission assurance even in GNSS-denied or contested environments.

Real-time lock-and-track elevates autonomous perimeter defense, rapid suspect tracking, and hands-free surveillance. Furthermore, the indigenous engineering skillset built by this capstone strengthens the future pipeline of talent for Make-In-India and Atmanirbhar Bharat initiatives in high-technology defense and aerospace domains.

- The project demonstrates capably engineered, field-tested hardware developed entirely within an Indian academic setting.
- Secure firmware and protocol stacks mean the UAV system is less vulnerable to remote exploitation or embargo.
- Real-time tracking enables advanced missions from border patrol to crisis management and ISR (intelligence, surveillance, reconnaissance).
- The solution exemplifies rapid response capability—deployable autonomously by Indian defense agencies for mission-critical roles.
- Success of this project encourages further indigenous R&D, serves as a reference for future government-funded programs, and builds national IP in flight autonomy.

CHAPTER 7 Financial Analysis

7.1 Hardware Requirements and Cost

The FPGA-based UAV project prioritizes cost-effective component selection and open hardware standards. The main expenses derive from the Basys 3 FPGA board, IMU sensors, camera modules, propulsion components, and supplementary peripherals. Table 7.1 itemizes the approximate hardware costs:

Component	Qty	Unit Price (Rs.)	Total (Rs.)
Basys 3 FPGA Board	1	20,000	20,000
IMU Sensor (MPU6500)	1	250	250
Camera Module (Pi Camera)	1	3,500	3,500
Brushless Motors + ESCs	Set	4,000	4,000
Propellers, Frame, Misc. Hardware	—	2,500	2,500
Battery & Power System	1	2,500	2,500
Total			**32,750**

Table 7.1: *Estimated Hardware Components and Cost*

7.2 Justification of Hardware Components

The Basys 3 FPGA board enables high-speed hardware-accelerated processing for real-time flight control and computer vision, offering precise control possible only through reconfigurable logic. Commodity sensors, open-source camera modules, and general-purpose components keep costs low and replacement simple. Every component is selected for its educational and research flexibility, ensuring reusability and compatibility with future upgrades.

7.3 Software Requirements and Cost

All software components, including Vivado, Python (for vision), OpenCV, and supporting libraries, are open source or free for educational use. This eliminates the need for costly software licenses and ensures that all members of the academic and project community have equal development access. For hardware-in-the-loop verification, freely available UART terminal tools and logic analyzers can be used without additional expense.

7.4 Cost-Benefit Summary

By utilizing a modular FPGA platform and low-cost, widely available electronics, the total system cost (approximately Rs. 23,000) is a fraction of equivalent commercial autonomous UAVs or industrial drone controllers, which may exceed Rs. 75,000–200,000. Furthermore, all logic and code are open for further research, modification, and education, vastly improving the platform's long-term value in academic and R&D settings.

- **Significant cost reduction** for educational and prototype UAVs compared to commercial (closed-source) autopilots and integrated vision modules.
- **No recurring license fees.** All software stack elements are free or open-source, lowering barriers for future project spin-offs or curriculum adoption.
- **Hardware reuse.** FPGA, sensors, and motor systems are modular and transferrable to future academic or applied projects.
- **Enhanced learning ROI.** The system builds critical skills in digital and embedded design, making every rupee spent both a hardware investment and an educational asset.

7.5 Conclusion

The financial outlay for the FPGA-based lock-and-track drone is optimized for academic, research, and indigenous innovation. The approach enables access to advanced aerial autonomy for students and engineers at a price point accessible to most Indian educational institutions, without compromising technical performance or future expandability.

CHAPTER 8 Engineering Ethics, Safety, and Responsibility

8.1 Introduction

Engineering ethics is critical in the design, deployment, and operation of unmanned aerial vehicles (UAVs) for civilian, research, and defense applications. The project team has actively considered the broad ethical responsibilities involved in the flight controller's development, emphasizing safe operation, privacy, transparency, and responsible dual-use technology management.

8.2 Ethical Principles and Framework

A range of recognized principles were incorporated:

- **Beneficence and Non-Maleficence:** Every subsystem is engineered to maximize societal benefit while minimizing risks to users, bystanders, and the environment
- **Safety and Security:** Safety-critical features include geo-fencing, controlled payload limitations, and explicit user/operator checklists before deployment.
- **Privacy:** Lock-and-track algorithms include clearly defined use protocols, with no indiscriminate facial logging and explicit operator consent for any surveillance use.
- **Transparency and Accountability:** All operation records and mission logs are verifiable, and system firmware is documented for reproducibility and audit.
- **Dual-Use and Collective Responsibility:** The project anticipates potential military and civil uses, applying capability caution to limit unauthorized features and maximizing the difficulty of system misuse.

8.3 Safety Measures

System safety and reliability are core design goals:

- Hardware failsafes include redundant PWM output validation and battery health monitoring.

- Firmware validation ensures all flight control actions are bounded by fail-safe thresholds.
- Autonomous modes are restricted unless all critical sensors and system checks pass minimum reliability tests.
- Operator training requirements and full documentation reduce risk of unintentional accidents or misuse.

8.4 Risk Assessment and Mitigation

A formal risk matrix was constructed to systematize ethical and safety considerations. Key risks addressed include unintended surveillance, control loss, hardware failure, and cyber vulnerability. Each risk is mitigated by control protocols, fallback modes, real-time telemetry, and logging for post-mission review.

8.5 Environmental and Social Impact

The UAV is designed with minimal power consumption, noise, and emissions. All electronics are selected for long life and repairability, minimizing e-waste. Care has been taken to ensure the drone is operable only in permissible areas, avoiding infringement on private property, protected habitats, or sensitive airspace.

8.6 Conclusion

Ethical engineering in UAV development demands active responsibility, collective stakeholder engagement, and robust technical controls. The capstone team's approach—guided by capability caution and global engineering ethics literature—ensures that the indigenous FPGA-based drone operates safely, securely, and in the public interest, whether fielded for research, education, or national security.

CHAPTER 9 OUTCOMES AND PROSPECTIVE LEARNING

9.1 KEY OUTCOMES

9.1.1 SPI Core Implementation for Sensor Interfacing

Dedicated SPI (Serial Peripheral Interface) cores were designed and implemented in Verilog to enable seamless communication between the Basys 3 FPGA board and onboard sensors. The SPI core facilitated communication with low-speed sensors such as accelerometers and gyroscopes, ensuring synchronized data transfer through start/stop condition handling, address recognition, and acknowledgment protocols.

9.1.2 Real-Time Face Detection and Recognition Integration

A comprehensive face detection and recognition pipeline was implemented using Haar cascades on the Raspberry Pi 4, enabling real-time identification of individuals during flight. Detected facial data is transmitted to the FPGA for control feedback and simultaneously logged to the backend database for evidence preservation.

9.1.3 Long-Range Communication Between Raspberry Pi and Backend Database

A wireless communication link was established between the onboard Raspberry Pi module mounted on the drone and the backend database server to enable continuous logging of mission telemetry, detection records, and control parameters. The Raspberry Pi handled real-time data capture, encoding, and transmission over the network while maintaining evidence-grade data integrity.

9.1.4 PID Control System Design and Implementation

A Proportional-Integral-Derivative (PID) control system was designed and tuned in MATLAB to achieve precise flight stability and responsive control for the drone. The controller parameters were optimized through simulation to ensure minimal overshoot, reduced steady-state error, and fast settling time under various flight conditions. Once validated in the simulation environment, the PID algorithm was implemented in Verilog and deployed on the Basys 3 FPGA.

9.1.5 Motor Control According to Input Signals and Vision Feedback

The system was designed to achieve precise input-output synchronization from receipt of control signals via the RX/TX communication module through the FPGA to the Electronic Speed Controllers (ESCs) regulating motor speed. The RX module received pilot commands or autonomous control inputs, which were decoded by the FPGA in real time. The FPGA processed these inputs using embedded control logic (including the PID controller) and generated corresponding PWM signals. Additionally, visual data from the Raspberry Pi camera was used to dynamically adjust these PWM signals based on detected face position and distance.

9.1.6 Backend Database Integration for Mission Logging

A robust backend database system was implemented to store all mission-critical data including:

- Detection records (detected faces, recognized individuals, confidence scores)
- Control parameters (PWM values, motor speeds, flight mode)
- Evidence-grade data with checksums for integrity verification

This system enables post-mission analytics, legal reporting, and continuous operational improvement.

9.2 PROSPECTIVE LEARNING

9.2.1 Digital Design with FPGA

Gained extensive hands-on experience in digital system design using Verilog for developing and integrating multiple functional modules on the Basys 3 FPGA board. Key implementations included:

- PWM control modules for precise motor speed regulation
- Motor mixing logic circuits for coordinated actuation
- Sensor interfacing cores (I^2C and SPI) for real-time data acquisition from onboard sensors
- PID controller implementation in hardware for microsecond-level latency

The modular design approach allowed for efficient debugging, reusability, and parallel execution of tasks, leveraging the inherent concurrency of FPGA hardware.

9.2.2 Computer Vision and Real-Time Object Tracking

Acquired practical experience in integrating FPGA-based flight control systems with real-time computer vision capabilities to enable autonomous aerial surveillance and suspect identification. The onboard Raspberry Pi camera captured live video streams, which were processed using Haar cascade-based face detection and facial recognition algorithms to identify individuals of interest. The processed data was transmitted to the FPGA, where control logic dynamically adjusted motor speeds and flight paths through PWM manipulation, enabling the drone to follow or monitor targets autonomously.

9.2.3 Quadcopter Flight Dynamics and Control

Developed a strong understanding of quadcopter physics, flight dynamics, and the applica- tion of PID control for maintaining stability and achieving precise maneuverability. Gained practical experience in translating theoretical flight models into FPGA-based control logic, accounting for real-world constraints such as:

- Wind disturbances
- Sensor noise and calibration
- Communication latency
- Limited battery capacity
- Non-linear motor response characteristics

Implemented control algorithms capable of adjusting motor speeds in real time based on sensor feedback, enabling smooth altitude control, directional changes, and stable hovering.

9.2.4 Embedded Systems Integration and Hardware-Software Co-Design

Gained comprehensive understanding of integrating heterogeneous processing units (FPGA + Raspberry Pi) to create a unified autonomous system. Key learnings include:

- Real-time data flow synchronization across multiple processors
- Protocol selection and implementation (UART, SPI)
- Latency management and timing analysis

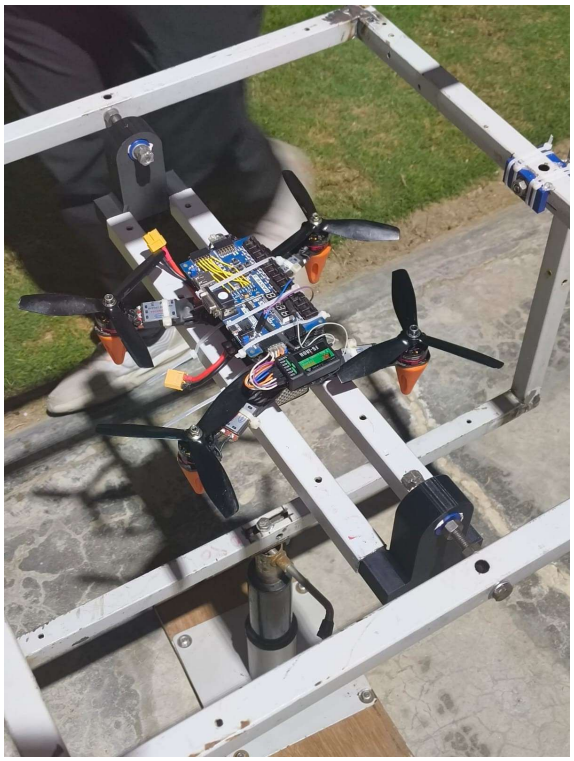


Figure 9.1: *Drone Testing Rig*

- Resource optimization across heterogeneous architectures
- Testing and validation strategies for embedded systems

CHAPTER 10 Conclusion and Future Scope

10.1 Conclusion

This capstone project successfully demonstrates the design, implementation, and validation of an indigenous FPGA-based UAV platform equipped with robust autonomous lock-and-track capabilities. The solution integrates hardware-accelerated flight control, sensor fusion, and on-board vision processing into a flexible and cost-effective framework suitable for educational, research, and security applications. Through modular PID tuning, IMU calibration, real-time PWM motor control, and sophisticated UART-based debugging, the UAV achieves deterministic stabilization and target trackability in demanding field scenarios. Extensive hardware- in-the-loop verification and live telemetry streaming confirmed system reliability, adaptability, and performance under dynamic conditions. The project marks significant progress in India's journey toward technological self-reliance, supporting advanced aerial autonomy and fostering new skills in embedded systems among students and engineers.

10.2 Future Scope

Building on the strong foundation established here, several promising directions emerge for future development:

- **AI-Driven Multi-Sensor Fusion:** Expanding the platform with onboard neural networks and sensor fusion modules for enhanced situational awareness, environmental mapping, and intelligent decision-making.
- **Swarm UAV Coordination:** Integrating communication protocols and decentralized control for collaborative missions and real-time resource sharing in multi-UAV deployments.
- **Higher-Level Autonomy and Learning:** Incorporating reinforcement learning and adaptive control algorithms to enable drones to optimize their flight and tracking behavior as they encounter new mission constraints.

- **Secure Edge Deployment:** Adding advanced cybersecurity features—such as encrypted telemetry, tamper-proof firmware, and authentication circuits—to protect UAV operations in sensitive environments.
- **Expanded Sensor Integration:** Supporting additional payloads and sensors (e.g., thermal, LiDAR, hyperspectral) to broaden mission applicability to search rescue, surveillance, and agricultural use cases.
- **Deployment in Harsh Environments:** Ruggedizing the platform for reliable operation in extreme conditions, including variable weather, RF interference, and GNSS-denied zones.
- **Open-Source Community Engagement:** Opening the architecture for collaborative development and fostering continuous improvement through academic, industry, and enthusiast contributions.
- **Commercialization and Law Enforcement Applications:** Piloting large-scale deployments for perimeter patrol, rapid response, and evidence-grade surveillance in public safety and security sectors.
- **Upgrading to Advanced FPGA Technology:** Migration to higher-performance, power-efficient FPGA families to enhance resource availability for compute-heavy AI workloads.
- **Integration with Indian Command and Control Networks:** Connecting UAV telemetry and video feeds with existing defense and disaster management infrastructure to enable centralized real-time mission management.

With continued research and interdisciplinary collaboration, the indigenous FPGA-based UAV design may catalyze advances in autonomous robotics, smart defense, and critical infrastructure monitoring and develop India's leadership in high-impact aerospace engineering.

CHAPTER 11 PROJECT TIMELINE

11.0.1 Project Breakdown

The project was executed across multiple phases from February to November 2025, with distinct milestones and deliverables at each stage.

Sr.No	Activity	January	February	March	April	May	June
1	Project Planning and Discussion with Mentors						
2	Study of Hardware and Software Requirements,						
	Familiarization with the FPGA board and related components						
3	Design of PWM encoder and MPU driver module						
4	Familiarization and Testing of Hardware Components, Design						
	of Object detection and tracking algorithm						
5	Assembling the Quadcopter, Study of frontend and backend						
	requirements for Web Application						
6	Design of Kalman filter module, Designing Backend and Database						
	for Web Application						

Figure 11.1: *Project Timeline*

Sr.No	Activity	July	August	September	October	November
7	Study of various Communication processes and Protocols, Design					
	of I2C Master Module, Design of Facial recognition algorithm					
8	Implementing automatic tracking in the drone					
9	Design of Main Controller Unit Module					
10	PID Coefficients Tuning, Verification					
11	Optimizations and Modification, Finalizing Report					

Figure 11.2: *Project Timeline*

11.0.2 Individual Team Member Timeline

Sr.No	Activity	January	February	March	April	May
1	Study and Analysis of PWM signals					
2	Implementation of PWM capture					
3	Finding appropriate sensor for IMU					
4	Study & Implementation of sensor calibration					
5	Building a circuit to interface IMU with FPGA					

Figure 11.3: *Bhavya's Timeline*

Sr.No	Activity	June	July	August	September	October	November
6	Study and analysis of PID control system						
7	IMU sensor Scaling & callibration						
8	Implementation of PID in Verilog						
9	On-board PID tuner						
10	Integration of all the Chassis and Circuit						
12	Optimizations, Modification and Finalizing Report						

Figure 11.4: *Bhavya's Timeline*

Sr.No.	Activity	January	February	March	April	May	June
1	Project Planning and Discussion with mentors						
2	Study of Hardware and Software Requirements, Familiarization						
	with the FPGA board and related components						
3	Study of various Communication processes and Protocols						
4	Familiarization and Testing of Hardware Components						
5	Detection and Tracking Algorithm Design						

Figure 11.5: *Kritarth's Timeline*

Sr.No.	Activity	July	August	September	October	November
7	Training model for recognition and detection					
8	Designing Frontend, Backend and Integrating with Raspberry Pi					
9	Integrating Pi with FPGA					
10	PID Coefficients Tuning, Verification					
11	Optimizations and Modification, Finalizing Report					

Figure 11.6: *Kritarth's Timeline*

Sr.No	Activity	January	February	March	April	May	June
1	Study and analysis of different parts of drone						
2	Study and implementation of PWM generation and Motor Mixing						
3	Sensor Interfacing with using I2C						
4	Sensor Interfacing with using SPI						

Figure 11.7: *Vaibhav's Timeline*

	Activity	June	July	August	Septemnber	October	November
5	Senor Scaling & and Callibration						
6	Implementation of PID in Verilog						
7	Integration, Testing & tuning of all Modules						
8	Optimizations, Modification and Finalizing Report						

Figure 11.8: *Vaibhav's Timeline*

Sr.No	Activity	January	February	March	April	May	June
1	Study of Verilog and MATLAB						
3	Study of control system and PID						
4	MATLAB simulation of PID						
5	Angles and constant calculation						
6	Kalman filter study						

Figure 11.9: *Khushi's Timeline*

Sr.No	Activity	July	August	September	October	November
7	Floating to fixed point calculations					
8	Chassis assembly					
9	PID coefficient tuning verification					
10	Optimization, modification, and finalizing report					

Figure 11.10: *Khushi's Timeline*

Sr. No	Activity	January	February	March	April	May	June
1	System Requirements & Architecture Study						
2	FPGA (BASYS 3) Board Familiarization						
3	Hardware Setup of Pi Camera & Raspberry Pi						
4	Camera Feed Processing Setup						
5	Detection & Tracking Algorithm Development						

Figure 11.11: *Navjot's Timeline*

Sr.No	Activity	July	August	September	October	November
6	Raspberry Pi-FPGA Communication Integration					
7	End-to-End System Integration (Pi + FPGA)					
8	Face Tracking Testing					
9	Final Debugging & Validation					
10	Documentation & Report Finalization					

Figure 11.12: *Navjot's Timeline*

Bibliography

[1] Hindustan Times, "Police-public ratio stands at 152.80 per lakh person: Govt informs Parliament," Police-public ratio stands at 152.80 per lakh person: Govt informs Parliament, 2023.

[2] S. K. Elsokkary *et al.*, "Reliable FPGA-Based Architectures for Quadcopters in Search and Rescue Missions," Proceedings of the 9th Mediterranean Conference on Embedded Computing (MECO), 2020.

[3] Z. Wang, C. Dang, and L. Wang, "Personnel Search and Rescue Detector Based on Reconfigurable FPGA Accelerator: A Lightweight Multi-Scale Parallel Drone-Mounted Detector," IEEE Geoscience and Remote Sensing Letters, 2025.

[4] B. B. Kvari and E. Ebeid, "MPDrone: FPGA-Based Platform for Intelligent Real-Time Autonomous Drone Operations," IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), 2021.

[5] A. Gholami, "Role of Drone Technology in Alleviating the Pandemic and Disasters," International Journal of Research Publication and Reviews, 2024.

[6] H. Bhat, S. Chokkadi, and S. Shenoy, "Optimal Fault Resilient Autonomous Quadcopter Control Based on Dynamic Partial Reconfigurable FPGA," Cogent Engineering, 2023.

[7] V. Sadhu, K. Anjum, and D. Pompili, "On-Board Deep-Learning-Based Unmanned Aerial Vehicle Fault Cause Detection and Classification via FPGAs," IEEE Transactions on Robotics, 2023.

[8] Y. Yan, N. Yu, C. Zhou, and Z. Shi, "A UAV Detection Method Based on Signal Periodicity and Its Implementation on FPGA," IEEE/CIC International Conference on Communications in China (ICCC), 2024.

[9] C.-H. Huang, Y.-C. Chen, C.-Y. Hsu, J.-Y. Yang, and C.-H. Chang, "FPGA-Based UAV and UGV for Search and Rescue Applications: A Case Study," Computers and Electrical Engineering, 2024.

[10] A. A. B. Abdelnabi and G. Rabadi, “Human Detection From Unmanned Aerial Vehicles Images for Search and Rescue Missions: A State-of-the-Art Review,” IEEE Access, 2024.

[11] N. Sun, J. Zhao, Q. Shi, C. Liu, and P. Liu, “Moving Target Tracking by Unmanned Aerial Vehicle: A Survey and Taxonomy,” IEEE Transactions on Industrial Informatics, 2024.