



## 1. Opening page

Electrical Engineering

# **Project Name: Real-Time License Plate Recognition System Using FPGA**

## **Project Chapter Document Abstract**

**Student Name:** Dar Eshel Epstein

**Supervisor's Name:** Binyamin Abramov



## 2. Table of Contents

<b>1. Opening page</b>	1
<b>3. Abstract</b>	3
<b>4. Introduction</b>	3
5.1. Goals Objectives and Measures	3
5.1.1. Goals	3
5.1.2. Objectives	4
5.1.3. Measures	4
5.2. Alternative's solution	5
5.3. Engineering challenges	6
<b>5. Literature review</b>	7
6.1. Literature review of solution for LPR systems	7
6.1.1. Encoder–Decoder Frameworks for LPR [6]	7
6.1.2. Hybrid Edge–Cloud Computing Systems [7]	7
6.1.3. Layout-independent ALPR Systems [8]	7
6.1.4. Multinational LPR [9]	7
6.1.5. Recognition in Fog-Haze Environments [10]	7
6.2. Conclusion and Comparison	8
<b>6. Methods</b>	9
7.1. Guideline for Designing an LPR System	9
7.2. Overview of the LPR System Design	10
7.3. Essential Resources and Learning for LPR System Development	11
<b>7. Results</b>	12
8.1. Development Stages	12
8.2. Testing Methods	13
8.2.1. Overview of the testing process	13
8.2.2. Technical Testing Procedures	13
<b>8. Discussion</b>	14
9.1. Risk management	14
9.2. Solutions and Contingency Plans	15
<b>9. Summary and conclusions</b>	16
<b>10. References</b>	17
<b>11. Appendices</b>	18
12.1. Work Plan	18
12.2. Acknowledgments	18



### 3. Abstract

This project focuses on the creation of a real-time license plate recognition system that employs Field-Programmable Gate Arrays (FPGAs) for enhanced hardware acceleration. The system is designed to provide accurate and efficient license plate detection and processing in real-time, suitable for automotive and surveillance applications.

Emphasizing hardware optimization and software integration, this project introduces a hybrid approach to real-time license plate recognition, integrating the computational prowess of FPGAs with advanced software methodologies.

### 4. Introduction

License plate recognition (LPR) technology is crucial for applications such as traffic management, law enforcement, parking control, and toll collection. However, conventional LPR systems, typically software-based and running on general-purpose computers, often face challenges such as slow processing speeds and high operational costs. To address these issues, this project proposes a novel real-time LPR system that utilizes field-programmable gate arrays (FPGAs), renowned for their parallel processing capabilities and high-speed image processing.

The development process begins with software preprocessing to ensure optimal image quality for recognition. Subsequently, the hardware capabilities of FPGAs are harnessed for efficient and accurate character recognition during the OCR (Optical Character Recognition) phase. This system represents a cohesive integration of software and hardware, meticulously tailored to meet the high-performance demands of LPR applications. The paper will detail the design, implementation, and experimental results, with a focus on the system's optimization and its potential to significantly enhance intelligent transportation systems.

### 5.1. Goals Objectives and Measures

#### 5.1.1. Goals

- The main goal of this project is to design and implement a system that can process and recognize license plate numbers from a camera in real time.
- The system should be able to capture and analyze images of license plates from different angles, distances, lighting conditions and backgrounds.
- The system should be able to extract the license plate, segment the characters and identify the number using optical character recognition (OCR) algorithms in real time.
- The system should be able to display the recognized license plate number on a screen or store it in a database for further use.



### 5.1.2. Objectives

- Review the existing literature and methods for license plate recognition.
- Select the appropriate hardware platform, camera and FPGA board for the system.
- Design and implement the preprocessing for the OCR algorithm that will run in real-time.
- Design and implement the OCR algorithms in real-time using hardware description languages (HDL).
- Test and evaluate the system performance and accuracy using various license plate images and videos.
- Compare and analyze the advantages and disadvantages of the hardware-based system versus the software-based system.
- Document and present the project results and findings.

### 5.1.3. Measures

In the formulation of performance targets for the real-time license plate recognition system, reference was made to industry-standard guidelines, as outlined in [1]. These guidelines offer a benchmark for the desired operational parameters of LPR systems, ensuring that the project's goals are both challenging and achievable. Table 1 presents the key performance metrics and their targets, as adapted from the recommendations provided in [1]:

Performance Metric	Unit	Target
Processing Speed	ms	100
Recognition Accuracy	%	At least 85
Camera distant from plat	Meters	3-6
Angle if the camera	degrees	Up to 30 from plate
Illumination for LPR	Lux	Minimum 50 (Dimly lit room)

Table 1

#### **Environmental Adaptation Disclaimer:**

While this license plate recognition system is designed with robustness in mind, it is important to note that extreme environmental conditions may impact its performance. The deployment of cameras and preprocessing algorithms has been optimized to enhance system reliability under a variety of conditions. However, absolute performance under all possible environmental scenarios cannot be guaranteed (heavy rain, sandstorm, heavy fog and similar extreme environmental scenarios).



## 5.2. Alternative's solution

In addition to the proposed real-time license plate recognition system, several alternative solutions were considered:

- **NVIDIA's AI Models [2]:**  
Utilizing NVIDIA's cutting-edge AI models provides a robust framework for LPR systems. NVIDIA's technology is known for its high processing power and efficiency, which can significantly enhance image recognition tasks. However, they require specific NVIDIA hardware, which may limit their applicability in certain environments.
- **Platerecognizer.com Technology [3]:**  
Platerecognizer.com provides cloud-based solutions and on-premises software for license plate recognition. Although their on-premises solution is designed to support processing in near real-time, it requires a robust computing environment with substantial processing power to manage the workload efficiently and minimize latency [4].
- **OpenCV and Python [5]:**  
Combining OpenCV with Python is a popular choice for image processing and computer vision projects. This approach allows for the development of custom LPR algorithms that can be tailored to specific requirements. Like Platerecognizer.com's technology, it requires a powerful computer for real-time processing.
- **Embedded Platforms:**  
For a more customized solution, embedded platforms can be employed. These systems often utilize OpenCV and Python for image processing tasks, similar to the methods shown in [4]. Embedded platforms provide the flexibility to design a system that closely aligns with the unique needs of the project.

Each alternative solution has its own set of advantages and limitations, which are detailed in Table 2. The selection of the most appropriate solution will depend on the specific requirements and constraints of the project.

Solution	Pro	Cons
NVIDIA's AI Models [2]	High performance Real-time processing	Requires NVIDIA hardware
Platerecognizer.com Technology [3]	High accuracy Near Real-time processing	Requires streaming of video to a high-performance computing unit for near real-time processing [4].
Embedded Platforms	Low power and portable	Can be challenging to implement complex algorithms
OpenCV and Python [5]	Open-source Flexible	Requires a powerful computer for real-time processing

Table 2



### 5.3. Engineering challenges

Developing a real-time license plate recognition system involves several engineering challenges that must be carefully navigated to ensure the system's effectiveness and reliability:

- **Hardware Compatibility:**  
A primary challenge is ensuring compatibility between the selected FPGA board and camera. The system must handle the computational demands of real-time processing without bottlenecks, necessitating the selection of hardware that can communicate effectively and process data at the required speeds.
- **Algorithm Optimization:**  
Optimizing algorithms for real-time license plate recognition requires accuracy and efficiency on the FPGA, as well as consideration of the embedded CPU's constraints used for preprocessing. The embedded CPU typically has limited processing power, demanding the development of highly efficient preprocessing routines that minimize computational load while preparing image data for the FPGA.
- **System Integration:**  
Another challenge is integrating the hardware and software components into a cohesive system. The camera, FPGA board, and any additional sensors or modules must work in unison to ensure reliable performance, requiring meticulous planning and testing for seamless system integration.
- **Environmental Conditions:**  
The LPR system must be robust enough to operate reliably under various environmental conditions, including lighting changes, weather conditions, and the dynamics of vehicle speeds and angles. Adapting to these conditions is critical for maintaining system performance.



## 5. Literature review

### 6.1. Literature review of solution for LPR systems

License Plate Recognition (LPR) systems are pivotal in intelligent transport systems, providing a means for automated vehicle identification. Recent advancements have focused on enhancing recognition accuracy and adapting to diverse environmental conditions. This literature review examines five significant contributions to the field.

#### 6.1.1. Encoder–Decoder Frameworks for LPR [6]

The Article introduced the EDF-LPR, an encoder–decoder framework that directly detects and recognizes license plate characters without considering the format of the license plate. This approach addresses the challenge of detecting plates with different formats and scales, achieving a detection rate of 99.51% and a recognition rate of 95.3% at about 40 frames per second Test on NVIDIA GeForce GTX 1080 GPU.

#### 6.1.2. Hybrid Edge–Cloud Computing Systems [7]

The Article proposed a light vehicle LPR system that leverages hybrid edge–cloud computing. This system aims to reduce latency and energy consumption by using channel pruning to reconstruct the backbone layer and deploying network models on edge gateways. The result is an impressive accuracy rate of 97% with a total number of parameters of only 0.606 MB.

#### 6.1.3. Layout-independent ALPR Systems [8]

The Article proposed developing an efficient and layout-independent ALPR system based on the YOLO detector. Their system unifies license plate detection and layout classification, achieving an end-to-end recognition rate of 96.9% across eight public datasets. This system outperformed previous works and commercial systems, demonstrating real-time performance even with multiple vehicles in the scene.

#### 6.1.4. Multinational LPR [9]

The Article tackled the challenge of multinational LPR with a system that uses generalized character sequence detection. Their approach, based on YOLO networks, includes steps for LP detection, unified character recognition, and multinational LP layout detection, showcasing its effectiveness across datasets from various countries.

#### 6.1.5. Recognition in Fog-Haze Environments [10]

The Article addressed the difficulty of recognizing license plates in fog-haze environments with their LPRFH method<sup>10</sup>. Utilizing a dark channel prior algorithm and a convolution-enhanced super-resolution convolutional neural network, their method can accurately recognize license plates even in severe fog-haze conditions.



## 6.2. Conclusion and Comparison

In conclusion, the comparative analysis presented in the table underscores the diverse methodologies employed to enhance LPR systems. From encoder–decoder frameworks and hybrid computing to layout-independent recognition and adaptability in challenging environmental conditions, each approach contributes uniquely to the field. In table 3. We highlight the strengths and limitations of these systems in relation to an FPGA-based solution, providing a clear perspective on potential areas for further development. As LPR technology continues to evolve, we can anticipate ongoing improvements in accuracy and efficiency, expanding its applicability in real-world scenarios.

Solution	Pros	Cons	Comparison with our proposed system
<b>Encoder–Decoder Frameworks for LPR [6]</b>	High accuracy and fps. Good for various plate formats.	May require substantial computational resources.	The proposed system could incorporate similar techniques on a hardware level of the FPGA for potentially faster processing.
<b>Hybrid Edge–Cloud Computing Systems [7]</b>	Low network size, good accuracy. Less cloud dependency.	Limited by edge device capabilities.	The proposed system offers on-device processing without cloud reliance, potentially more efficient.
<b>Layout-independent ALPR Systems [8]</b>	High accuracy, robust to conditions. Good fps on GPU.	Requires a high-end GPU for best performance.	The proposed system can provide real-time processing with potentially lower hardware requirements.
<b>Multinational LPR [9]</b>	Handles various LP layouts. Quick processing per image.	Required powerful GPU and 24GB RAM to run in Realtime	The proposed system aim to leverage hardware acceleration for enhanced real-time processing efficiency.
<b>Recognition in Fog-Haze Environments [10]</b>	Effective in poor visibility. Utilizes advanced image processing.	Specific to fog-haze conditions.	The proposed system could incorporate similar techniques for robustness in various environments.

Table 3





## 6. Methods

### 7.1. Guideline for Designing an LPR System

The design of an LPR system is a complex process that involves several critical steps. Each step must be carefully considered to ensure the system's accuracy and efficiency. Below is a guideline that outlines these steps, incorporating insights from the reviewed literature.

#### Step 1: Image Acquisition

The first step involves capturing the image of the vehicle's license plate. This is typically done using cameras positioned to capture clear images of the plates under various lighting and weather conditions.

#### Step 2: Preprocessing

Once the image is captured, preprocessing is essential to enhance the image quality. This may include normalization, grayscale conversion, and noise reduction. For instance, Article [10]. Used a dark channel prior algorithm to handle fog-haze environments, which could be adapted for other adverse conditions.

#### Step 3: License Plate Detection

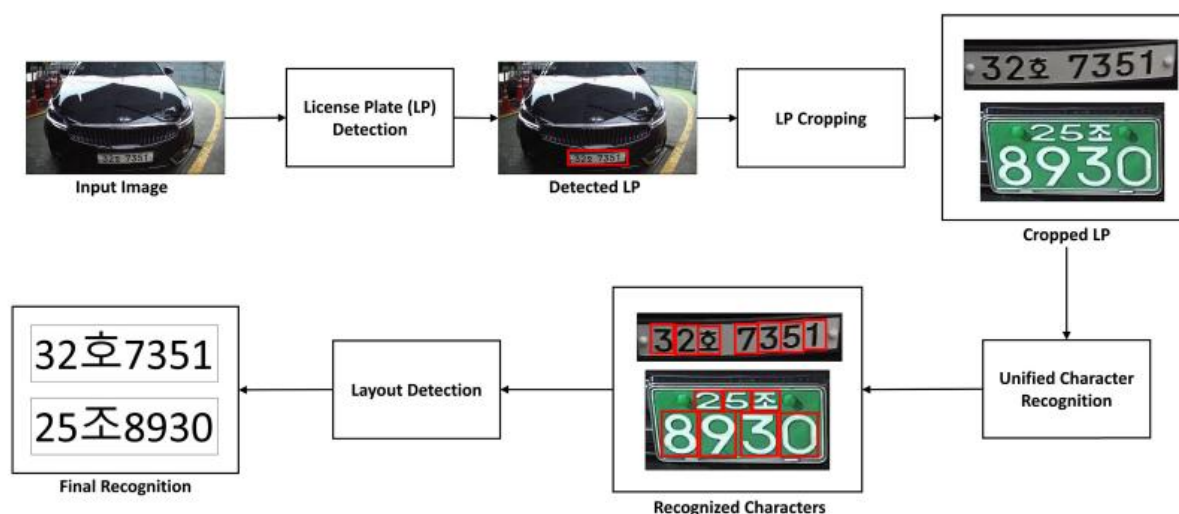
The system must then detect the license plate within the image. Techniques like the YOLO detector, as used in Article [8]. and Article [9]. are effective for this purpose. They allow for real-time detection even with multiple vehicles in the scene.

#### Step 4: Character Segmentation

After detecting the license plate, the next step is to segment the characters. This can be challenging due to different font styles and sizes. Encoder-decoder frameworks like EDF-LPR [6] can be beneficial here, as they do not rely on the format of the license plate.

#### Step 5: Character Recognition

This step involves recognizing each character on the license plate. Deep learning models are commonly used for this task. The system proposed in Article [6]. Achieved high recognition rates and could serve as a model for developing your character recognition module.



Article [9], Figure 4. Block diagram of Proposed ALPR System.



## 7.2. Overview of the LPR System Design

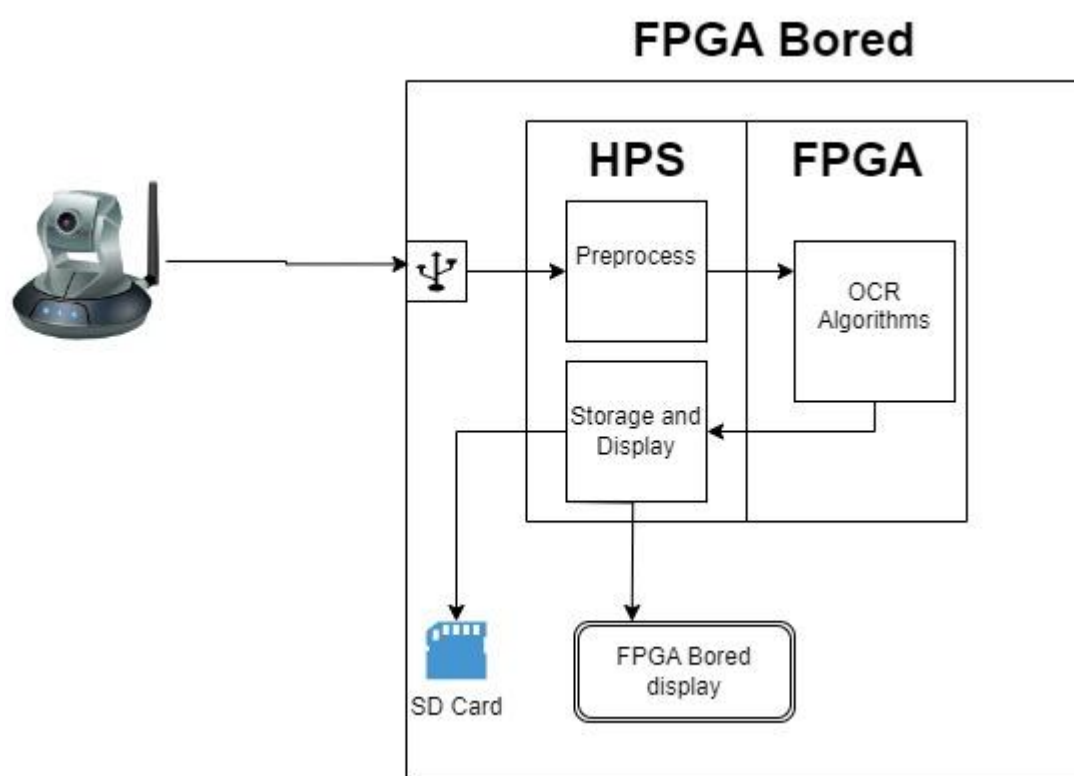
After a detailed analysis of LPR systems as discussed in Section 7.1 of other Articles, we introduce an optimized method for LPR. The system diagram visually outlines this approach, which provides a high-level view of the LPR system architecture.

The LPR system is designed to efficiently capture and process vehicle license plate numbers. The process starts with a camera that captures images of vehicles. The system then employs a preprocessing phase where the license plate is detected and isolated from the vehicle image. This step is crucial as it ensures that only the license plate is focused upon for the next stage.

Once the license plate is identified, the image is cropped to highlight only the license plate. This image is then processed by the FPGA component, which runs Optical Character Recognition (OCR) algorithms specifically designed to recognize and decode the license plate number.

The system's architecture, comprising the Hardware Processing System (HPS) and the FPGA, is optimized to handle these tasks effectively. The HPS manages the initial image processing, while the FPGA is dedicated to the OCR algorithms that extract the license plate number.

This approach ensures that the LPR system is streamlined for its primary function—recognizing and processing the numbers on the license plates—with high accuracy and efficiency.



High level system diagram



## 7.3. Essential Resources and Learning for LPR System Development

In this section, we will refine our focus on the specific resources that will be instrumental in the development of our LPR system:

### Resource Identification for Each Step:

- **Image Preprocessing:** Techniques that cater to image quality enhancement, particularly under varying environmental conditions, will be identified.
- **Real-Time Object Detection:** We will explore state-of-the-art object detection algorithms that are optimized for embedded systems, ensuring real-time processing without compromising accuracy.
- **OCR Algorithms:** Considering OCR algorithms that are specifically optimized for FPGAs, focusing on those that provide high accuracy and efficiency in character recognition.

### FPGA Resources:

- **FPGA Selection:** After evaluating various options, the DE 10 Standard FPGA by Terasic was chosen [12]. This board aligns with our project's specifications, offering a dual-core ARM Cortex-A9 processor (HPS). Its computational capabilities and adaptability make it ideal for our preprocessing requirements. The DE 10 Standard has received approval for purchase, confirming its suitability for our LPR system development.
- **Learning to Work with FPGA:** To effectively utilize the DE 10 Standard FPGA, we will gather resources such as official documentation, tutorials, and community forums. These will provide the necessary guidance on programming and interfacing with the board.

### Skill Development:

- **FPGA Programming:** Mastery of hardware description languages (VHL) such as VHDL or Verilog will be essential. Additionally, understanding the integration of software with FPGA for running OCR algorithms will be a key skill to develop.
- **System Integration:** Learning how to integrate the HPS with the FPGA to create a seamless workflow from image capture to character recognition will be crucial.

By focusing on these resources and areas of learning, we aim to ensure that the LPR system is not only theoretically sound but also practically viable and capable of meeting real-world demands. The DE 10 Standard FPGA will play a pivotal role in this development, providing the necessary hardware flexibility and power to process images and recognize characters efficiently.



## 7. Results

Our project's aim to deliver a real-time LPR system using FPGA technology is guided by the performance measures detailed in section 5.1.3. These measures establish a benchmark for the system's effectiveness, ensuring that the design and implementation are in line with industry-standard guidelines. With an aimed processing speed of **100ms** and a recognition accuracy target of at **least 85%**, the system is expected to mark a significant advancement in LPR technology. The measures also define operational parameters such as camera distance and angle, as well as minimum illumination levels, which are essential for the system's operation under various environmental conditions. This section of the results will explore the projected outcomes of the project, detailing how these measures are anticipated to be achieved or surpassed, thus affirming the system's performance against the set benchmarks.

### 8.1. Development Stages

Given the intricate nature of the LPR system and the swift processing capabilities of FPGA logic, particularly for the OCR component, we anticipated that this segment would operate with exceptional speed.

Therefore, our initial development focus was on the preprocessing stage. As outlined in section 5.1.3., our total processing time is set at **100ms**. With this in mind, we aimed to design the preprocessing stage to be efficient enough not to consume more than a prudent portion of our total processing time, estimated at **80%**. This strategic approach allowed us to maximize the capabilities of the HPS, setting a robust foundation for the subsequent stages of the license plate recognition process and ensuring a seamless transition to real-time application.

Upon the successful completion of the preprocessing stage and isolation of the license plate, our next objective will be to commence the development of the OCR algorithm tailored for the FPGA. This algorithm will be designed to take full advantage of the FPGA's processing power, ensuring swift and accurate translation of the visual license plate data into machine-encoded numbers.



## 8.2. Testing Methods

To ensure the robustness and accuracy of our LPR system, we have implemented a multi-faceted testing methodology, with each component undergoing rigorous evaluation:

### 8.2.1. Overview of the testing process.

#### Algorithm testing:

- **Preprocessing Component Testing:** Test for accuracy in license plate detection and processing speed.
- **FPGA OCR Component Testing:** Evaluate for character recognition accuracy and robustness under various conditions.

#### Testing Protocols

- **Individual Component Testing:** Assess each algorithm with static images to ensure component-specific reliability.
- **Live Camera Preprocessing Testing:** Verify real-time capabilities of the preprocessing algorithm with a live feed.
- **Full System Testing:** Conduct end-to-end testing with a live camera to validate the integration and performance of the complete system.

### 8.2.2. Technical Testing Procedures

The technical testing procedures are divided into several stages, each targeting a specific component or the system as a whole:

- **Preprocessing Algorithm Testing:** A set of photos or a live feed from a camera will be processed to confirm that the preprocessing algorithm accurately isolates the license plate.  
To measure the real runtime, will be logged the process and the average time will be calculated to ensure the system operates within the expected timeframe.  
Additionally, a GPIO on the board can be toggled from 1 to 0 to mark the start and end of the process, respectively. This signal can be captured with an oscilloscope.
- **OCR Algorithm Testing:** The OCR algorithm will initially be validated using computer simulation tools like Modelsim. A set of preprocess photos from the preprocess algorithm will be sent to the FPGA, with that we will confirm that each photo output number matches the license plate number.  
To measure the real runtime, a GPIO on the board will be toggled from 1 to 0 to mark the start and end of the process, respectively. This signal can be captured with an oscilloscope. Alternatively, runtime logging can be implemented on the HPS, recording the time when the processed image is sent to the FPGA and when the license plate number is received back from the FPGA.
- **Full System Testing:** The entire system will be tested from the moment it begins processing an image until the license plate number is stored on the SD card. This end-to-end test will log the system's performance and confirm that all components work together seamlessly in real-time conditions. Additionally, a GPIO on the board can be toggled from 1 to 0 to mark the start and end of the process, respectively. This signal can be captured with an oscilloscope.



## 8. Discussion

### 9.1. Risk management

As we embark on the development of our LPR system, it is imperative to anticipate potential obstacles that may arise. The success of our project hinges not only on our technical prowess but also on our ability to foresee and manage risks. Table 4 presents a comprehensive overview of the identified risks, their potential impact, and the strategies we have devised to mitigate them. This proactive approach ensures that we are prepared to address challenges promptly and maintain the momentum of our project.

Risk	Impact Level	Likelihood	Mitigation Strategy
Compatibility issues with the camera and the board	High	Low	Use cameras recommended by the manufacturer [12].
Inability to find/create a real-time object detection algorithm for HPS	High	Medium	Downsample the image; offload processing to FPGA.
Inability to fit full OCR to the FPGA	High	Medium	Work with binary images; process characters sequentially.
Extreme environmental conditions affecting performance	Low	High	Use cameras suited for harsh conditions; improve preprocessing algorithms.

Table 4



## 9.2. Solutions and Contingency Plans

Outlined in the risk management section, here are solutions and fallback plans for noted issues, with explanations of how to counteract them:

- **Compatibility Solutions:** To mitigate the risk of compatibility issues between the camera and the FPGA board, it is imperative to select cameras that are explicitly recommended by the board's manufacturer. This proactive measure will significantly reduce the likelihood of integration difficulties and ensure seamless system functionality.
- **Real-time object detection Algorithm:** In the event of challenges with the implementation of a real-time object detection algorithm for the HPS, consider implementing an image down sampling strategy coupled with offloading substantial processing tasks to the FPGA. This dual approach is designed to balance the computational load effectively, thereby facilitating real-time operational capabilities.
- **OCR Optimization:** To circumvent the constraints of embedding a full OCR on the FPGA, the strategy involves processing binary images and handling character recognition in a sequential manner. This tailored approach is expected to enhance the OCR process efficiency within the FPGA's operational limitations.
- **Environmental Adaptation:** For extreme environmental conditions that could affect performance, use cameras suited for harsh conditions and improve preprocessing algorithms. This will help in maintaining system reliability and accuracy in varying conditions. Although as noted in section 5.1.3, reliability cannot be guaranteed under extreme conditions.



## 9. Summary and conclusions

The project aims to develop a swift and reliable license plate recognition system, primarily for traffic management and law enforcement, where quick access to license plate data is crucial. The system comprises five main stages: image capture, preprocessing (license plate isolation), character segmentation, character recognition, and storage of the recognized license plate number. The FPGA executes the critical tasks of character segmentation and recognition, while the HPS handles the other stages. This architecture negates the need for an external PC, as all processing is done within the SoC/FPGA board.

The project faces several challenges and risks, such as hardware compatibility, algorithm optimization, and system integration. To overcome these challenges, the project adopts various strategies, such as selecting appropriate cameras and FPGA boards, designing tailored OCR algorithms, and using standard interfaces and protocols. The project also identifies potential risks and their mitigation plans, such as testing the system under different environmental conditions, verifying the performance benchmarks, and documenting the design process.

The project aims to design a system architecture that can process and recognize license plates in real-time, implement a preprocessing stage on the FPGA. The project also intends to explore the strengths and limitations of FPGA technology for image processing applications, highlighting the trade-offs between speed, accuracy, and resource utilization. However, the project does not guarantee that the system will work reliably under extreme environmental conditions, as this depends on many factors beyond the scope of the project.

The next step of the project is to start the development of the preprocessing stage for the FPGA, which is essential for the OCR algorithm development. The project will conduct comprehensive testing to ensure the system meets the performance benchmarks of processing speed and recognition accuracy. The project will also evaluate the system's performance under various environmental conditions and compare it with existing LPR solutions. The project aims to deliver a novel and effective LPR system using FPGA technology that can contribute to the advancement of image processing applications.

To summarize, this project advocates for an innovative LPR solution utilizing FPGA technology, capable of real-time processing and recognition of license plates. The system is composed of five key stages, executed by both the FPGA and the HPS, thereby obviating the necessity for an external PC. The project confronts a variety of challenges and risks associated with hardware compatibility, algorithm enhancement, and system integration, offering strategies for mitigation and resolution. It also delves into the balance between speed, accuracy, and resource allocation inherent in FPGA technology for image processing tasks. The project is committed to rigorous testing and evaluation of the system's performance under different scenarios and in comparison, to extant LPR systems. The project hopes to deliver a reliable and effective LPR system that can benefit traffic management and law enforcement.





















## 10. References

1. License Plate Capture Camera Guide: [The Best License Plate Capture Camera Guide \(cctvcameraworld.com\)](https://cctvcameraworld.com) last retrieve at 12/7/2024
2. Nvidia's AI Models license plate recognition: [Creating a Real-Time License Plate Detection and Recognition App | NVIDIA Technical Blog](#) last retrieve at 12/7/2024
3. Platerecognizer.com Technology: [Automatic License Plate Recognition - High Accuracy ALPR \(platerecognizer.com\)](#) last retrieve at 12/7/2024
4. Platerecognizer.com Technology installation Guide: [Stream Installation Guide | Plate Recognizer](#) last retrieve at 12/7/2024
5. License Plate Recognition with OpenCV and Tesseract OCR: [License Plate Recognition with OpenCV and Tesseract OCR - GeeksforGeeks](#) last retrieve at 12/7/2024
6. Gao, Fei et al. "EDF-LPR: A New Encoder–Decoder Framework for License Plate Recognition." IET intelligent transport systems 14.8 (2020): 959–969. Web.
7. Leng, Jiancai et al. "A Light Vehicle License-Plate-Recognition System Based on Hybrid Edge–Cloud Computing." Sensors (Basel, Switzerland) 23.21 (2023): 8913-. Web.
8. Laroca, Rayson et al. "An Efficient and Layout-independent Automatic License Plate Recognition System Based on the YOLO Detector." IET intelligent transport systems 15.4 (2021): 483–503. Web.
9. Henry, Chris, Sung Yoon Ahn, and Sang-Woong Lee. "Multinational License Plate Recognition Using Generalized Character Sequence Detection." IEEE access 8 (2020): 35185–35199. Web.
10. Jin, Xianli et al. "Vehicle License Plate Recognition for Fog-haze Environments." IET image processing 15.6 (2021): 1273–1284. Web.
11. Lelis Baggio, Daniel, and Daniel Lelis Baggio. Mastering openCV 3 : Gets Hands-on with Practical Computer Vision Using openCV 3. Second edition. Birmingham, England ; Packt, 2017. Print.
12. Terasic DE10-Standard: <https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=1081> last retrieve at 27/7/2024
13. Yu, Ke, Minguk Kim, and Jun Rim Choi. "Memory-Tree Based Design of Optical Character Recognition in FPGA." Electronics (Basel) 12.3 (2023): 754-. Web.



## 11. Appendices

### 12.1. Work Plan

	Task Mode ▾	Task Name ▾	Duration ▾	Start ▾	Finish ▾
✓		Pick a project	65 days	Thu 01/02/24	Wed 01/05/24
✓		Literature review	15 days	Wed 01/05/24	Tue 21/05/24
✓		Design draft digram	39 days	Thu 02/05/24	Tue 25/06/24
✓		Resources for Relevent parts	4 days	Wed 26/06/24	Sun 30/06/24
✓		Order parts	6 days	Mon 01/07/24	Mon 08/07/24
		Work on SOW	19 days	Tue 09/07/24	Sat 03/08/24
✓		SOW presentation	32 days	Sun 09/06/24	Sat 20/07/24
		Submit SOW to supervisor	1 day	Mon 05/08/24	Mon 05/08/24
		Submit SOW	13 days	Tue 06/08/24	Thu 22/08/24
		Engineering Report	117 days	Wed 07/08/24	Thu 16/01/25
		Poster presentation	1 day	Tue 22/04/25	Tue 22/04/25
		Submit draft of final report	1 day	Thu 19/06/25	Thu 19/06/25
		Submit final report	1 day	Thu 24/07/25	Thu 24/07/25

### 12.2. Acknowledgments

The author acknowledges the use of AI technologies for computational assistance in the development of this project. Specifically, Microsoft Copilot Pro provided support in generating content and refining ideas that contributed to this research.