

**Distribution Intelligence Systems
Coursework**

Stefan-Cristian Daitoiu

2317310

Introduction	3
Background Research	4
Point-to-Point Transmission System	4
Point-to-Multipoint Transmission System	4
Multi-Master Transmission System	5
Overview of Designed System	6
Microcontroller Selection	7
Slave Controller Design	9
SPI Communication Design	10
Proteus Circuit Implementation	11
Master Program Development	12
Slave Program Development	14
Analysis	15
System Operation Analysis	15
Testing Methodology	16
Test 1: Tail Light Operation	16
Test 2: Brake Light Operation	16
Test 3: Reverse Light Operation	16
Simulation Results	17
Quality Management Systems and Safety Considerations	17
Verification and Testing	17
Documentation	17
Reliability Improvement	18
Standards and Continuous Improvement	18
Conclusion	18
Bibliography	20

Introduction

Modern automotive systems rely heavily on embedded electronic control systems to improve vehicle safety, reliability and functionality. Traditional vehicle wiring systems used direct electrical connections between switches and lights, resulting in large wiring harnesses, increased weight and reduced flexibility. As vehicle technology has advanced, manufacturers have moved towards distributed electronic control systems where multiple microcontrollers communicate with each other to control different vehicle functions. [1]

A modern vehicle may contain dozens of electronic control units (ECUs), each responsible for controlling a specific function such as engine management, braking systems, infotainment, safety systems and lighting. These systems require reliable communication methods to allow individual controllers to exchange information efficiently. [2]

This project focuses on the design and implementation of a simplified automotive lighting control system using two PIC16F877A microcontrollers. The system demonstrates the principles of a distributed embedded system by separating input processing and output control between two processors. A master microcontroller is responsible for receiving driver inputs through switches, while a slave microcontroller processes these commands and controls the vehicle lighting outputs. [3]

The communication between the two microcontrollers is achieved using the Serial Peripheral Interface (SPI) protocol. SPI was selected because it provides fast communication, simple hardware implementation and is commonly used for communication between microcontrollers and peripheral devices. [4]

The designed system represents one corner of a vehicle lighting system and includes four essential lighting functions:

- Direction indicator
- Tail light
- Brake light
- Reverse light

The master PIC16F877A receives inputs from four switches representing driver commands. These commands are transmitted to the slave PIC through SPI communication. The slave microcontroller then controls the corresponding LEDs to simulate the vehicle lighting system.

The project was developed and tested using Proteus simulation software with programs written in C language using the CCS C Compiler. The purpose of this design is to demonstrate practical application of microcontrollers, digital communication protocols and embedded system design techniques.

Background Research

Point-to-Point Transmission System

A point-to-point transmission system is a communication method where information is transferred directly between two devices. In this arrangement, there is a dedicated communication path between one transmitting device and one receiving device. The transmitter sends information directly to the receiver without the need for addressing multiple devices. Point-to-point communication is one of the simplest forms of digital communication because the system only involves two devices. Since there is only one destination, the communication process is straightforward and does not require complex data management techniques. [5]

One major advantage of point-to-point systems is their simplicity. The hardware design is easy to implement because only two devices are connected. Communication errors are also easier to diagnose because there are fewer possible causes of failure. However, the main disadvantage of point-to-point communication is limited scalability. If additional devices need to communicate, additional connections must be created. For example, if multiple microcontrollers were required in an embedded system, every device would require its own communication link. This would increase wiring complexity, cost and physical size. [7]

In automotive applications, point-to-point communication may be suitable for simple systems where only two devices need to exchange information. However, modern vehicles require many controllers to communicate, meaning more advanced communication architectures are often preferred.

Point-to-Multipoint Transmission System

A point-to-multipoint transmission system allows one controlling device to communicate with multiple receiving devices. Unlike a point-to-point system, where communication occurs between only two devices, point-to-multipoint systems allow a single master device to control several slave devices. In this type of system, the master device is responsible for managing communication and transmitting information to the connected slave devices. The slave devices receive the transmitted information and perform the required operation. [8]

A common example within automotive applications is a central electronic control unit communicating with multiple lighting modules. Instead of every switch being physically connected to every light, the controller can send digital commands to individual modules. [9]

The main advantage of point-to-multipoint communication is reduced wiring complexity. Multiple devices can share a common communication network, reducing the number of physical connections required.[10] For example, in this project, the master PIC does not directly connect to the lighting outputs. Instead, it communicates with the slave PIC through SPI. This creates a modular system where the input and output sections are separated.

However, point-to-multipoint systems also introduce additional challenges. The master device becomes responsible for controlling communication between all connected devices. If the master fails, communication with the slave devices may stop. [11]

Protocols such as SPI and I²C can be configured for point-to-multipoint communication. SPI uses individual Slave Select lines to allow the master to choose which slave device should receive data. I²C uses addressing to identify different devices connected to the same bus. [12]

The system designed in this project follows a point-to-multipoint style architecture, although only one slave device is used. The design could easily be expanded by adding additional slave PIC controllers for other vehicle lighting sections.

Multi-Master Transmission System

A multi-master transmission system is a communication architecture where multiple devices connected to the same network can operate as the controlling device. Unlike a traditional master-slave system, where only one device controls communication, any device within a multi-master network can initiate data transmission. [13] Since multiple devices can attempt communication at the same time, additional control methods are required to prevent communication conflicts. These methods are known as arbitration techniques.

A common example of a multi-master communication system is the Controller Area Network (CAN) bus, which is widely used in automotive applications. CAN bus allows multiple electronic control units to communicate without requiring a central controller. For example, the engine control unit, braking system and transmission controller can all exchange information through the same communication network. [14]

However, multi-master communication systems are more complex compared with simple master-slave systems. The communication protocol must include methods for determining priority and preventing data collisions. For example, if two devices attempt to transmit simultaneously, the system must determine which message has priority and allow that device to continue transmission. [15]

Although the lighting system developed in this project does not use a multi-master architecture, understanding this type of communication is important because modern automotive systems commonly use distributed networks based on multi-master principles.

System Design

Overview of Designed System

The designed system is a multi-processor vehicle lighting control system developed using two PIC16F877A microcontrollers. The purpose of the system is to demonstrate how multiple embedded controllers can communicate together to perform a practical automotive function.

The system was designed using Proteus simulation software, with programming completed in the C language using the CCS C Compiler. The design consists of a master microcontroller responsible for reading driver inputs and a slave microcontroller responsible for controlling the vehicle lighting outputs.

The reason for separating the system into two microcontrollers is to demonstrate the advantages of distributed processing. In a real vehicle, different electronic control modules are responsible for different sections of the vehicle. For example, one controller may manage the front lighting system while another controls the rear lighting system. Communication between these controllers allows the vehicle to operate as a coordinated system.

In this project, the master controller represents the main vehicle control input unit, while the slave controller represents a lighting control module. The communication between the two devices is achieved using SPI communication.

The completed system consists of:

- Two PIC16F877A microcontrollers
- Four input switches connected to the master controller
- Four LED outputs connected to the slave controller
- SPI communication connection between both microcontrollers
- 5V power supply
- Crystal oscillators for clock generation

The four lighting functions implemented are:

1. Left indicator light
2. Tail light
3. Brake light
4. Reverse light

Microcontroller Selection

The PIC16F877A microcontroller was selected for both the master and slave devices because it provides suitable features for embedded communication applications. The integrated SPI module was an important factor in selecting this microcontroller because it allows reliable communication between multiple devices without requiring software-based serial communication.

The large number of input/output pins available also makes the PIC16F877A suitable for connecting switches and LEDs while maintaining separate communication lines.

Using the same microcontroller for both the master and slave simplified the design process because identical hardware configurations could be used. Only the software configuration changes depending on whether the device operates as the SPI master or SPI slave.

Master Controller Design

The master **U1** PIC16F877A was designed as the input processing unit. Its main purpose is to read the driver's lighting requests and transmit appropriate commands to the slave controller.

Four latching switches were connected to PORTB of the master microcontroller. The input configuration is shown below:

PIC Pin	Input Function
RB0	Indicator switch
RB1	Tail light switch
RB2	Brake light switch
RB3	Reverse light switch

Table 1: Master input pins

The switches were configured as digital inputs. Each switch changes the input state between logic LOW and logic HIGH depending on whether the switch is activated.

When a switch is activated, the master detects a HIGH signal and sends a corresponding command through SPI communication.

For example:

When the indicator switch is activated:

1. Master reads RB0 as HIGH.
2. Master generates the indicator ON command.
3. Command is transmitted through SPI.
4. Slave receives the command.
5. Slave begins flashing the indicator LED.

PIC Pin	Output Function
RB0	Indicator LED
RB1	Tail light LED
RB2	Brake light LED
RB3	Reverse light LED

Table 2: Slave output pins

The master does not directly control any lights. Instead, it acts as a communication controller, allowing the system to demonstrate separation between input processing and output control.

Slave Controller Design

The slave **U2** PIC16F877A was designed as the lighting control module. It receives commands from the master and controls the vehicle lighting outputs. The output configuration is shown below:

Command	Function
0x01	Enable indicator
0x02	Disable indicator
0x03	Turn tail light ON
0x04	Turn tail light OFF
0x05	Turn brake light ON
0x06	Turn brake light OFF
0x07	Turn reverse light ON
0x08	Turn reverse light OFF

Table 3: Commands

When a command is received, the slave compares the received data with predefined command values stored within the program.

SPI Communication Design

Serial Peripheral Interface (SPI) was selected as the communication protocol because it provides fast, full-duplex communication between microcontrollers.

SPI communication uses four main signals:

- Serial Clock (SCK)
- Master Out Slave In (MOSI)
- Master In Slave Out (MISO)
- Slave Select (SS)

The SPI connections used in the design are shown below: The master generates the SPI clock signal using RC3. This synchronises the data transfer between both devices.

The MOSI line transfers commands from the master to the slave, while the MISO line allows the slave to send information back if required.

Although this design does not require feedback from the slave, the MISO connection was included because it represents a complete SPI communication system.

The Slave Select line is used to activate communication with the slave device. The master pulls the Slave Select line LOW before sending data and returns it HIGH after transmission has completed.

Function	Master PIC	Slave PIC
Clock	RC3	RC3
MOSI	RC5	RC4
MISO	RC4	RC5
Slave Select	RA5	RA5

Table 4: Comms pins

Proteus Circuit Implementation

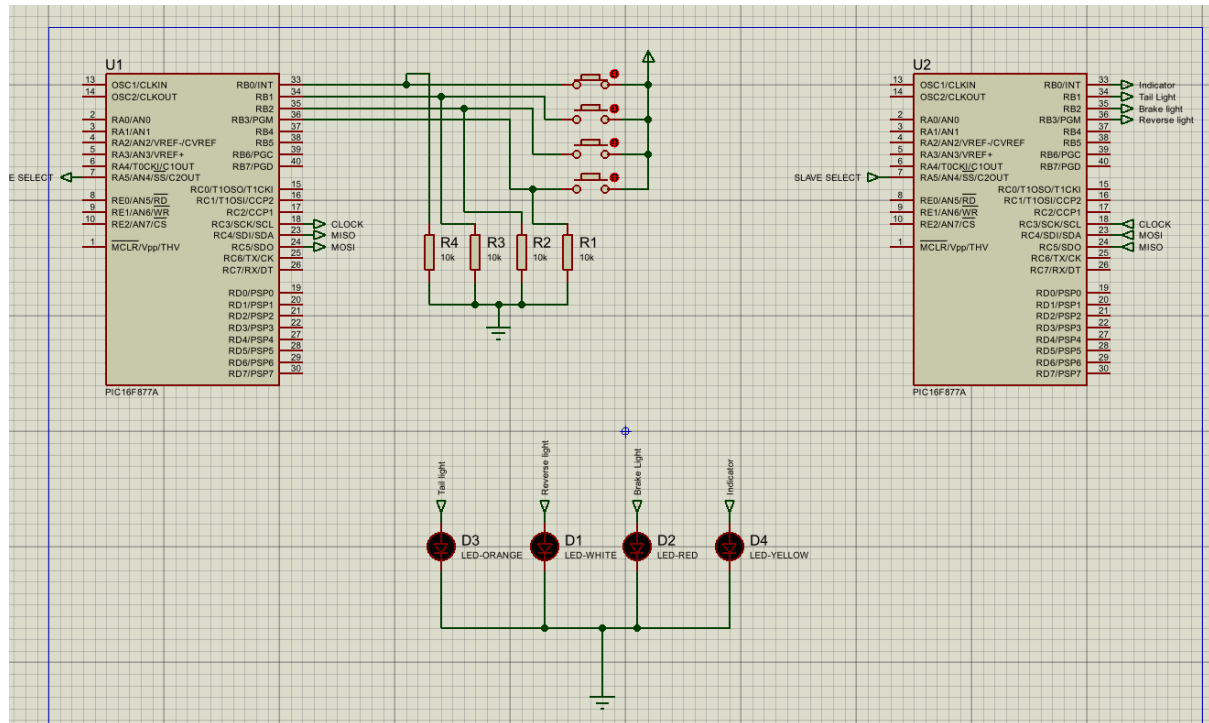


Figure 1: Complete circuit

The complete circuit was designed and simulated using Proteus software.

The master circuit contained four switches connected to PORTB. These switches represented the driver's lighting controls.

The slave circuit contained four LEDs connected to PORTB. These LEDs represented the vehicle lighting system.

Current limiting resistors were placed in series with each LED to prevent excessive current flow and protect the components.

The SPI communication wires were connected between the two microcontrollers, creating a complete embedded communication network.

The simulation allowed the operation of the system to be tested without requiring physical hardware. By activating the switches in Proteus, the response of the slave lighting module could be observed.

Master Program Development

The master program begins by configuring the PIC hardware.

The following operations are performed during startup:

1. Disable analogue functions.
2. Configure PORTB as digital inputs.
3. Initialise SPI communication as master.
4. Configure Slave Select output.

After initialisation, the program enters an infinite loop.

During each loop cycle:

1. The indicator switch state is checked.
2. The tail light switch state is checked.
3. The brake light switch state is checked.
4. The reverse light switch state is checked.

Depending on the switch state, the appropriate command is transmitted to the slave.

The master continuously updates the slave controller, ensuring that the lighting outputs always match the switch positions.

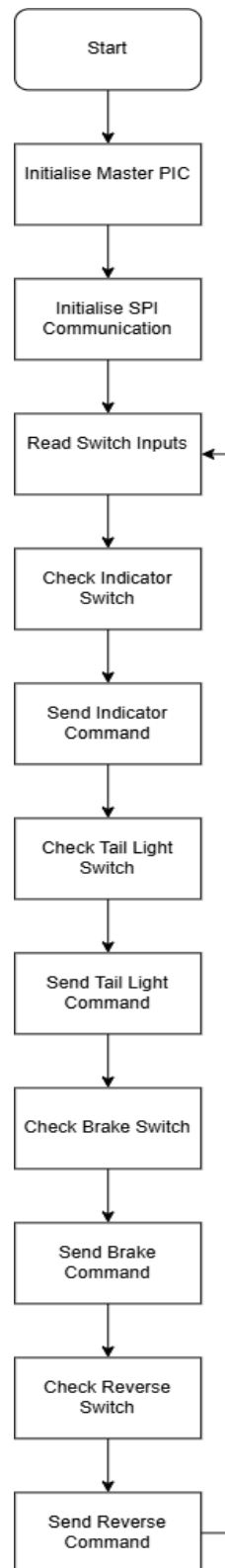


Figure 2: Master flowchart

Slave Program Development

The slave program begins by configuring the SPI module as a slave device.

The output pins connected to the LEDs are configured as digital outputs.

The slave then continuously monitors incoming SPI communication.

When data is received:

1. The command value is stored.
2. The command is compared against predefined values.
3. The appropriate lighting output is activated.

For the indicator function, a software timing routine was implemented.

When the indicator command is received:

- Indicator mode is enabled.
- The output pin changes state every 500 milliseconds.
- The LED flashes continuously until the OFF command is received.

This demonstrates how a slave controller can perform independent processing while still responding to commands from the master device.

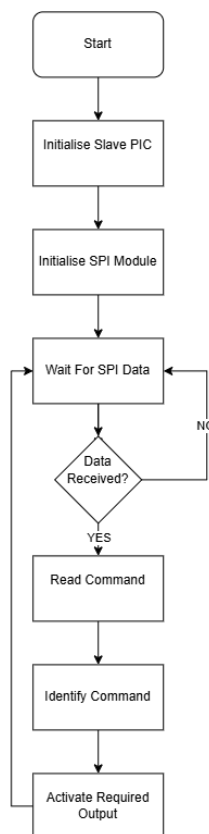


Figure 3: Slave flowchart

Analysis

System Operation Analysis

The completed multi-processor vehicle lighting control system demonstrates the successful integration of microcontrollers, embedded software and serial communication. The system operates using a distributed architecture where the master and slave PIC16F877A devices have separate responsibilities.

The master controller is responsible for user interaction and communication management, while the slave controller performs the physical control of the lighting outputs. This separation of tasks improves system organisation and represents the structure used in many real embedded automotive applications.

When the system is powered on, both PIC16F877A microcontrollers initialise their hardware configurations. The master configures its input pins and SPI module, while the slave configures its output pins and SPI module.

The master then continuously monitors the four input switches connected to PORTB. Each switch represents a vehicle lighting function:

- Indicator switch
- Tail light switch
- Brake light switch
- Reverse light switch

When a switch changes state, the master converts the input into a digital command. The command is then transmitted to the slave controller using SPI communication.

The slave continuously monitors the SPI communication channel. When a command is received, the slave identifies the command value and performs the required operation.

For example, when the brake switch is activated:

1. The master detects a HIGH signal on RB2.
2. The master sends the brake ON command through SPI.
3. The slave receives the command.
4. The slave activates RB2 output.
5. The brake LED illuminates.

When the brake switch is released, the master sends the brake OFF command and the slave disables the output.

Testing Methodology

Testing was performed using Proteus simulation software. The purpose of testing was to verify that the communication system and lighting outputs operated correctly.

Several functional tests were completed.

Test 1: Tail Light Operation

The tail light switch was activated.

Expected result:

- Master sends tail light ON command.
- Slave activates RB1 output.
- Tail light LED illuminates.

Actual result:

The tail light LED activated correctly.

Test 2: Brake Light Operation

The brake switch was activated.

Expected result:

- Brake command transmitted through SPI.
- Slave activates brake output.

Actual result:

The brake LED responded correctly when the switch was pressed.

Test 3: Reverse Light Operation

The reverse switch was activated.

Expected result:

- Reverse command sent by master.
- Slave activates reverse LED.

Actual result:

The reverse LED operated successfully.

Simulation Results

The Proteus simulation demonstrated that the designed system successfully achieved the required functionality.

The master PIC correctly detected input signals and transmitted commands through the SPI communication link. The slave PIC correctly decoded the received information and controlled the corresponding lighting outputs.

The simulation also demonstrated the advantage of separating system functions between multiple processors. The master controller handled user inputs, while the slave controller managed lighting outputs.

Quality Management Systems and Safety Considerations

Quality management systems are an essential part of engineering development because they provide structured processes to ensure that products and systems consistently meet required performance, reliability, and safety requirements. In automotive applications, quality management is particularly important because electronic system failures can directly affect vehicle operation and potentially create safety risks for drivers and other road users. [16]

A vehicle lighting control system requires a high level of reliability because functions such as indicators, brake lights, and reverse lights provide critical information to surrounding road users. Failure of these systems could reduce vehicle visibility and increase the risk of accidents. Therefore, the design process must consider not only functionality but also verification, reliability, fault detection, and compliance with relevant automotive standards. [17]

Verification and Testing

Verification and testing form a fundamental part of quality management by ensuring that an engineering system operates correctly under expected conditions. Testing should be performed throughout the development process to identify hardware or software faults before deployment. [18]

Repeated testing is particularly important in automotive electronic systems because components may be exposed to long operating periods, electrical noise, and environmental changes. Reliability testing helps determine whether the system can maintain correct operation throughout its expected lifetime. [19]

Documentation

Accurate documentation is an important aspect of engineering quality management because it allows systems to be understood, maintained, and improved throughout their lifecycle.

The documentation produced for this project includes circuit diagrams, software source code, hardware configuration details, testing results, and design decisions. Maintaining clear documentation allows future engineers to understand the operation of the master and slave controllers, identify potential improvements, and support future modifications or troubleshooting.

Reliability Improvement

Automotive electronic systems must provide reliable operation over extended periods, meaning additional features may be required to improve system robustness. Reliability improvements can include communication error detection, fault reporting mechanisms, LED failure monitoring, and backup operating modes. [20]

For example, if communication between the master and slave controllers was interrupted, the system could detect this failure condition and provide a warning signal or enter a predefined safe operating mode. Implementing fault detection methods reduces the likelihood of undetected failures and improves overall system safety.

Standards and Continuous Improvement

The development of automotive electronic systems is guided by industry standards that define requirements for safety, reliability, and electromagnetic compatibility. In a production vehicle application, systems such as the master and slave lighting controllers would need to consider standards including ISO 26262 functional safety requirements, automotive electromagnetic compatibility (EMC) requirements, and vehicle lighting regulations. [21][22]

Compliance with these standards helps ensure that electronic systems are suitable for automotive environments and operate safely under expected conditions. Continuous improvement is achieved through repeated testing, performance analysis, and redesign processes. By analysing system behaviour and identifying weaknesses, engineers can implement improvements that increase reliability, reduce failure risk, and improve overall system performance.

Conclusion

This project successfully designed and simulated a PIC-based multi-processor vehicle lighting control system using SPI communication.

Two PIC16F877A microcontrollers were used to create a distributed embedded system. The master controller was responsible for reading four user inputs, while the slave controller controlled the vehicle lighting outputs.

The implementation demonstrated the practical use of microcontrollers, digital communication protocols and embedded C programming. The SPI communication system allowed reliable communication between the two processors while maintaining a simple hardware design.

The Proteus simulation confirmed that the system operated correctly. All four lighting functions were successfully controlled:

- Indicator
- Tail light
- Brake light
- Reverse light

The project also demonstrated the benefits of dividing system responsibilities between multiple processors. This architecture improves organisation, reduces processing requirements on individual controllers and reflects the design approach used in modern automotive electronics.

Bibliography

- [1] Wikipedia Contributors, “Electronic Control Unit,” Wikimedia Foundation, July 2019. [Online]. Available: https://en.wikipedia.org/wiki/Electronic_control_unit
- [2] “Unlocking Vehicle Intelligence: A Practical Guide to Can and LIN Bus Networks,” May 2025. [Online]. Available: <https://copperhilltech.com/blog/unlocking-vehicle-intelligence-a-practical-guide-to-can-and-lin-bus-networks/>
- [3] M.-C. Bertola and C. Hartweg, *Français IRe Toutes Séries*. Hachette Education, 2018.
- [4] “Serial Peripheral Interface (SPI) Peripheral,” 2025. [Online]. Available: <https://www.microchip.com/en-us/products/microcontrollers/8-bit-mcus/peripherals/communication-connectivity/spi>
- [5] “Point-To-Point (Telecommunications),” Nov. 2021. [Online]. Available: [https://en.wikipedia.org/wiki/Point-to-point_\(telecommunications\)](https://en.wikipedia.org/wiki/Point-to-point_(telecommunications))
- [6] “[En-Ca]Point-to-Point - a Direct Link Between Two Communication Nodes Bypassing Intermediate Devices.,” *ITG CA*, Sept. 14, 2023. <https://ca.itglobal.com/glossary/point-to-point/>
- [7] “A Guide to Point to Point Network: What It Is and an Overview of Its Pros and Cons - Ruijie Networks,” 2023. [Online]. Available: <https://www.ruijie.com/en-global/support/tech-gallery/point-to-point-network>
- [8] “Point-To-Multipoint Communication,” *Digi.com*, 2021. https://docs.digi.com/resources/documentation/digidocs/90001456-13/concepts/c_point_to_multipoint.htm
- [9] “Clemson Vehicular Electronics Laboratory: Automotive Data Communication Buses.” [Online]. Available: https://cecas.clemson.edu/cvel/auto/auto_buses01.html
- [10] “Point-To-Multipoint Communication,” May 2021. [Online]. Available: https://en.wikipedia.org/wiki/Point-to-multipoint_communication
- [11] C. Diot, W. Dabbous, and J. Crowcroft, “Multipoint Communication: A Survey of Protocols, Functions, and Mechanisms,” *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 277–290, Apr. 1997, doi: 10.1109/49.564128.
- [12] C. Nelson, “Working with I2C Devices,” Mar. 2022. [Online]. Available: <https://learn.adafruit.com/working-with-i2c-devices/address-conflicts>

- [13] “26.4 Multi-Master Bus Systems, Arbitration, and Synchronization,” 2022. [Online]. Available:
<https://onlinedocs.microchip.com/oxy/GUID-0EC909F9-8FB7-46B2-BF4B-05290662B5C3-en-US-12.1.1/GUID-76E847C6-8B91-40D9-86F6-A917307D5BEE.html>
- [14] “Controller Area Network.” [Online]. Available:
<https://www.kth.se/social/upload/64/CAN.pdf>
- [15] “26.4 Multi-Master Bus Systems, Arbitration, and Synchronization,” 2022. [Online]. Available:
<https://onlinedocs.microchip.com/oxy/GUID-0EC909F9-8FB7-46B2-BF4B-05290662B5C3-en-US-12.1.1/GUID-76E847C6-8B91-40D9-86F6-A917307D5BEE.html>
- [16] “Aerospace & Automotive: Why Quality Management Is Key.” [Online]. Available:
<https://www.bsigroup.com/en-GB/insights-and-media/insights/blogs/aerospace-and-a-utomotive-why-quality-management-is-key/>
- [17] “Checking Your Browser - reCAPTCHA,” *Nih.gov*, 2026.
<https://pmc.ncbi.nlm.nih.gov/articles/PMC10383653/>
- [18] H. Y. Yeo, “Automotive Electronics Testing for Safe, Reliable Vehicles,” *Keysight.com*, 2026.
<https://www.keysight.com/blogs/en/tech/educ/2024/automotive-electronics-testing>
- [19] M. Rahmani, W. Hintermaier, B. Muller-Rathgeber, and E. Steinbach, “Error Detection Capabilities of Automotive Network Technologies and Ethernet - a Comparative Study,” *2007 IEEE Intelligent Vehicles Symposium*, pp. 674–679, June 2007, doi: 10.1109/ivs.2007.4290194.
- [20] “ISO 16750-4: Climatic Load Testing for Automotive Electrical and Electronic Equipment,” *Intertek.com*, 2019.
<https://www.intertek.com/automotive/standards/iso-16750-4/>
- [21] ISO, “ISO 26262-1:2018,” 2018. [Online]. Available:
<https://www.iso.org/standard/68383.html>
- [22] “The Road Vehicles Lighting Regulations 1989,” 2015. [Online]. Available:
<https://www.legislation.gov.uk/uksi/1989/1796>