MANSOURA UNIVERSITY
FACULTY OF ENGINEERING
ELECTRONICS & COMMUNICATION ENGINEERING DEPARTMENT

Automobile Security System

Based on Face Recognition

Project members	ID number
Abdelrahman Ashraf El-Nahas	1000103685
Abdelrahman El-Sayed Helaly	1000103899
Abdelrahman Mohamed Abdelhady	1000104256
Abdallah Mohamed Roshdy	1000151207
Ali Ahmed El-Gendy	1000103657
Mohamed Reda El-Sayed	1000103786
Mostafa Ghazy Ahmed	1000103697

Supervisor

Prof. Dr. Sherif Keishk

Academic Year: 2021 – 2022

July 2022

ABSTRACT

In this, technology became linked to everything, and the positive effects of this connection had some negative effect, which enabled thieves to penetrate these things and make them vulnerable to theft, and for this the development of the security system

And in the field of cars, technology has become in every part of the car, and we can say that the car has become a world that contains many organisms that talk to each other through their own language and in a specific style “protocol”.

In order to protect this “protocol” from being penetrated by any thief and to protect the safety of data transmission between the objects inside the car, we are talking in this project “Automobile security system based on face recognition” about protecting the car from operating it from thieves or undesirable persons.

When a person buys a car, he submits photos of himself and of trusted people to the database of the system

and through that, when anyone enters the car, the car will not be unlocked except through the face recognition system. If the result is that the person is trusted, the car will work, but if it is not It will take several measures according to its own protection system.

TABLE OF CONTENTS *(after updating the Table of contents, Apply TOC1 Style to it)*

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1 Introduction

1.1 What is an Embedded System?

Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer" – Marilyn Wolf

Electronic innovation has made extraordinary steps and these days the nature of electronic parts—execution, power, and unwavering quality—empowers utilizing them notwithstanding for basic frameworks. In the meantime, the diminishing expense of electronic innovation permits them to be utilized to support any function in a car.

Generally, a vehicle contains a dozen to nearly 100 electronic control units (ECUs). There are two main modules of electronic systems i.e. information-entertainment and hard-real-time control of mechanical parts.

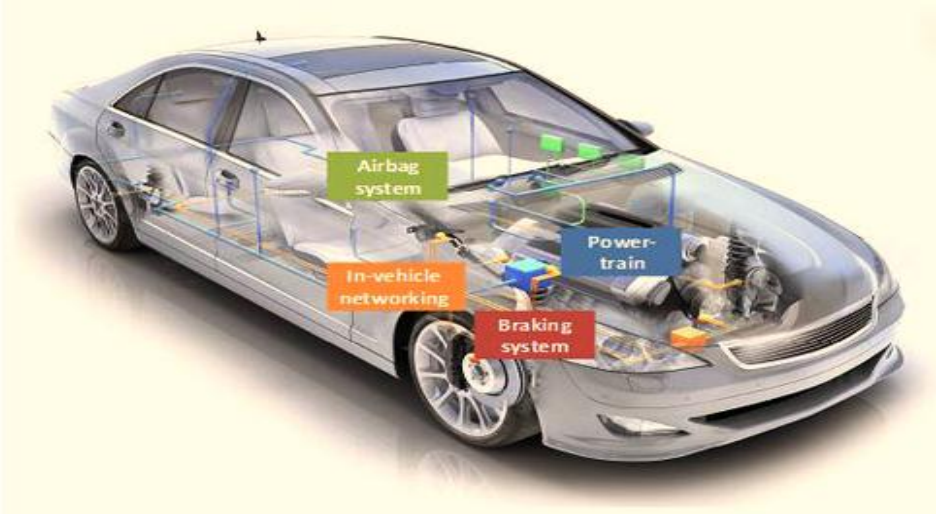


Figure 1-1 what is an Embedded System?

The different type of microcontrollers used in embedded systems are:

- 8-bit or 16-bit Microcontroller
- Serial and Parallel Input-Output
- Analog to Digital and Digital to Analog Conversions
- Flash Memory and PROM
- Signal generators and Timers

1.2 Why automotive embedded system is in a great demand?

1.2.1 Growing vehicles sales

An extensive range of industries are involved in the designing, development and selling automobiles. Every year the sales and production of vehicles are increasing globally due to growing demand of consumers. Every vehicle manufactured is equipped with embedded system, this will drive the Global Automotive Embedded System Market.

The most used embedded system in a vehicle includes adaptive cruise control, airbag, telematics, traction control, in-vehicle entertainment system, emission control system, parking system, navigation systems, collision sensors, climate control, radio, anti-lock braking system etc.

As per TechSci Research, "Global Automotive Embedded System Market By Vehicle Type (Passenger Cars, LCV and HCV), By Type (Embedded Hardware and Embedded Software), By Component (Sensors, Microcontrollers, Transceivers and Memory Devices), By Region, By Company, Competition, Forecast & Opportunities, 2025", Global automotive embedded system market is projected to grow over 7% CAGR for the forecast period. Rising focus on vehicle safety features and increasing demand from consumers for electric vehicles as well as strict emission standards are the contributing factors strengthening the market growth. Based on the application, the market has been segmented into infotainment & telematics, body electronics, safety & security, and powertrain & chassis control. Due to the rising demand for safety features such as anti-lock braking systems (ABS) in the vehicles, the market for safety and security applications is expected to grow at the highest rate in the forecast period.

1.2.2 Increasing Focus on Vehicle Safety Features

The demand for safety features in vehicles has been incorporated by different companies based on the requirement by the end user and further motivated by the regulations imposed the regulatory bodies. During recent years, the number of traffic fatalities has come down throughout the developed economies. These number are becoming stagnant and the whole credit goes to widespread use of safety systems. To include safety feature in the vehicles, OEM's are focusing on technological advancements which includes many sensors and advanced technical systems in the vehicle. To inbuilt all the sensor and technical features in vehicle the need of embedded system arises. These technological advancements are further fuelling the global automotive embedded system market.

Major example of active safety system could be recognised as pedestrian recognition, adaptive speed control, blind spot detection, lateral collision warning, cooperative lane changing indication, merging assistance, car breakdown warning, integrated car safety, etc. These features are anticipated to minimize the accident and accident related deaths. This is one of the major reason for the growth in the demand for embedded system market globally.

1.2.3 Increasing demand of automation

As the demand for safety and security of drivers and passengers along the roadside is increasing, the demand for connected car devices is also soaring globally. The connected car devices are also considered helpful in analysing accidents and breakdown data to provide valuable inputs both to car makers and road infrastructure designer and designer. Moreover, projects such as GALILEO, EGNOS, European Emergency Call in Europe, Brazil's SINIAV & SIMRAV, Russia's ERA GLONASS United State's Dynamic Speed Harmonization (SPD-HARM), Queue Warning (Q-WARN), Cooperative Adaptive Cruise Control (CACC), etc. will be enhancing the demand for connected car devices across the globe.

The increasing demand for connected car will drive the automotive embedded system market globally due to increasing demand for sensors and engineering systems in the vehicles.

1.2.4 Growing Automotive Component Industry

The automotive component industry is expected to increase at a significant CAGR over the next five years due to increasing production and sales of vehicles every year. Every vehicle produced require components and embedded complete devices often including hardware and mechanical parts and it controls many devices. This will boost the demand for automotive embedded system market.

CHAPTER 2

PROBLEM DEFINIITIONS & CHALLENGES

2 Problem Definitions & Challenges

2.1 Problem statement

In the first decade of the twenty-first century, the rate of car theft in the world, whether by stealing the car by breaking into it directly or by stealing the key from its owner

Even with the development of technology and its extensive use in the automotive industry, there were many loopholes that could enable thieves to steal the car or tamper with the data inside it

According to an American statistics National Insurance Crime Bureau (NICB) President and CEO David Giloy told lawmakers in Congress that some cities saw auto theft rise more than 280 % between 2019 and 2021.

Thus, we note that even with the increase of technology that represents a kind of features and protection for the car, car theft is still a top problem.

Therefore, our project is focused on reaching a solution to this problem and working to raise the protection system for the car and protect it from being even operable in the hands of someone other than its owner or trusted persons.

Therefore, the challenge in this project is how to make the car recognize who is its owner, who are the trusted people, and who are the opposite.

And what is her behavior in the event that a thief tries to steal her even if he has the car key, and this is what we will try to reach in the next chapters.

2.2 Project Challenges

1- Challenges related to OCV

2- Challenges related to Memory

3- Challenges related to Communication

4- Challenges related to AUTOMOTIVE system

2.2.1 Challenges related to OCV

As we will see in the coming chapters, we will work on the microcontroller ATMEGA32 and it is known that it is programmed in the C language, but the part of the OCV is programmed in Python, so the challenge is that the microcontroller does not have any operating system to govern this process, and this makes it difficult for Programming in more than one language for more than one process on the same board.

2.2.2 Challenges related to Memory

The software solution must organize the new software application into volatile or nonvolatile memory of the client device so that it can be executed when the update process completes. The solution must ensure that a previous version of the software is kept as a fallback application in case the new software has problems. Also, we must retain the state of the client device between resets and power cycles, such as the version of the software we are currently running, and where it is in memory, The maximum storage capacity of the controller's RAM must also be taken into account.

2.2.3 Challenges related to Communication

The biggest challenge in communication is the process associated with the challenge mentioned in 2.2.1 How to link the Python language from the operating system of the facial recognition system to the C language in the controller.

And also the challenge of adhering to the intercom system between the parts of the car and protecting the transmitted data in this system.

2.2.4 Challenges related to AUTOMOTIVE system

Operating the car’s protection system and linking it to other systems, which is the challenge based on the project, so that all systems are connected simultaneously and smoothly.

2.3 Project Requirements

1 - Safety

The database of photos of trusted people must be made accessible through the manufacturer and the owner with their own code

2 – Security

The data transmitted between the ECUs must be through a protocol specific to the CAN according to the framework of the protocol so that it is not difficult to steal it from the hacker.

3 - Reliability

Would make sure the data is transferred from one ECU to another ECU is correct and must wait till all data is transferred first before processing the data, so a reliable communication protocol must be used.

4 - Scalability

Should be designed to support large number of embedded devices . The software must be designed in a way to support scaling too.

2.4 Project objective

higher security

The rise in the protection system of the car, because it became the main factor for opening and operating the car is that the person inside is a trusted person and registered on the database.

CHAPTER 3

SYSTEM DESIGN

3 System Design

3.1 Abstract view

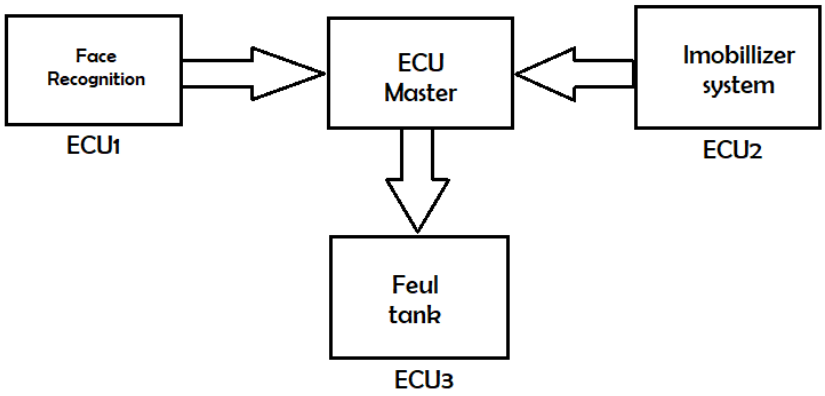


Figure 3-1 Abstract view

1- Face Recognition system

This system sends the result (0 or 1) from ECU1 to the master ECU , and accordingly the master ECU takes action according to the result sent to him from ECU2.

2- Immobilizer system

The immobilizer system sends the output result if the start button is pressed with the main key in the owner's possession from ECU2 to the master ECU, and accordingly the master takes action by linking with the result of ECU1.

3- Master ECU

It is the board responsible for taking action according to the results it reaches from ECU1 ,ECU2 and the Web , and send it to the fuel tank ECU to activate the car or take another action in case it is stolen.

3.2 Implemented view

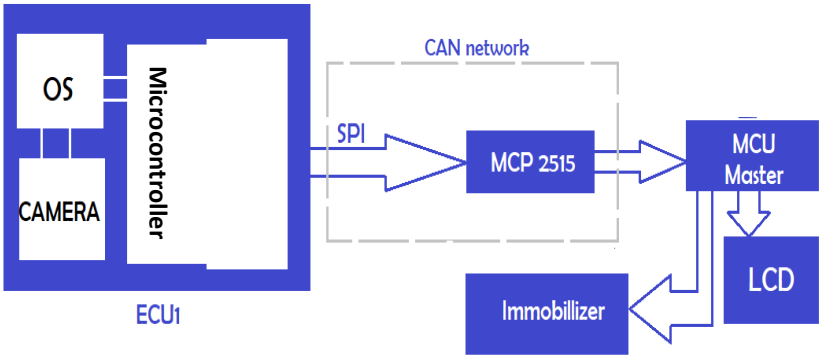


Figure 3-2 Implemented view

In the executive part of the project

ECU1 identifies the face through OCV and sends the result if the person is trusted or not to the controller in this ECU to the master through the CAN network.

And the immobilizer system sends its result whether the button is pressed or not through the CAN.

The master ECU decides whether the feature is activated or not through the car's website or not, and accordingly, the master sends the activation signal to the tank or not.

>> We have shown the output of this system on the LCD to illustrate the process that the car takes.

3.3 System components

We can consider that components has divided into major parts :

1 – Face Recognition system components

2 – Network componentsS

3 – Output components

3.3.1 Face Recognition system components

1-MCU "Atmega32"

2-Camera "Laptop"

3-Transfer Block

3.3.1.1 MCU "Atmega32"

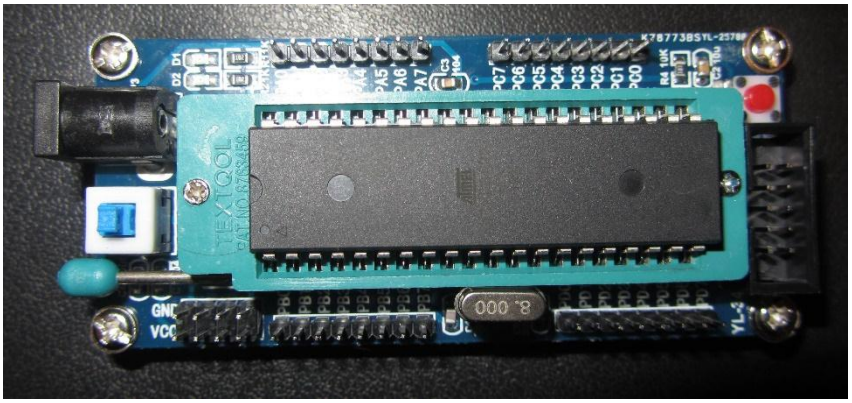


Figure 3-3 ATMEGA32

The task of MCU in this part is to receive the result from the recognition system by USART network (in ECU1).

3.3.1.2 Camera "in Laptop"

Figure 3-4 Laptop Camera

The task of this component is to recognize & analysis the face id of the driver & compare it with the IDs in the data base.

3.3.1.3 Transfer block

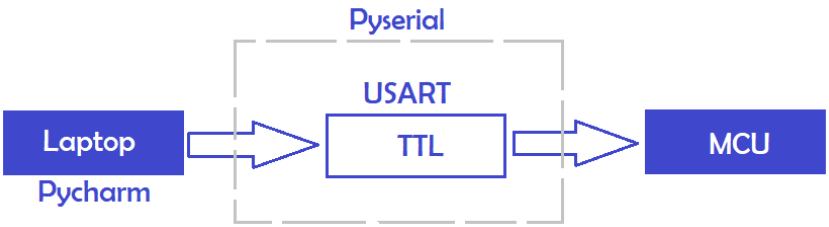


Figure 3-5 Transfer block

Figure 3-6 TTL

Consist of:

Pycharm : The IDE where the development of OCV is coded.

Pyserial : The program through which the result is sent from Pycharm to MCU.

TTL : The USB TTL Serial cables are a range of USB to serial converter cables which provide connectivity between USB and serial UART interfaces.

MCU: have its main task as explained.

3.3.2 Network components

1-CAN component "MCP2515"

2-TTL

3.3.2.1 MCP2515

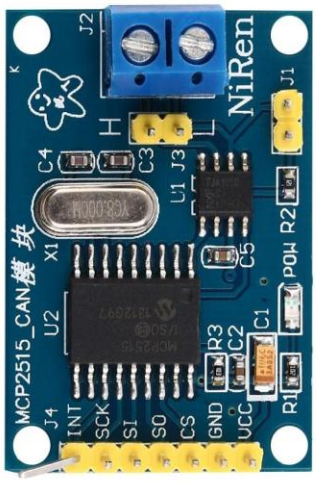


Figure 3-7 MCP2515

The MCP2515 is a second generation stand-alone CAN controller. It is pin and function compatible with the MCP2510 and also includes upgraded features like faster throughput, data byte filtering, and support for time-triggered protocols.

It can be programmed by SPI

3.3.2.2 TTL

The same its main task as explained in 3.3.1

3.4 Output components

LCD

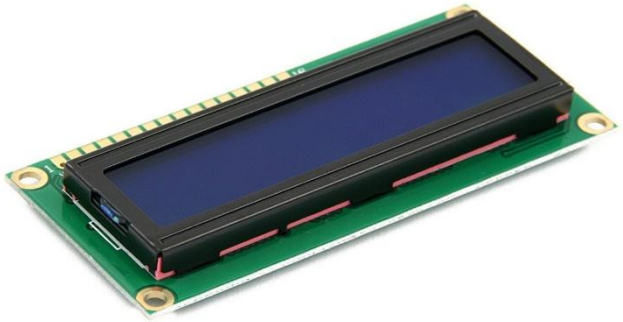


Figure 3-8 LCD 16x2

The main task is to view the result of the output of the system if the car is on or not or the person is trusted or not

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in color or monochrome. LCDs are available to

display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayed or hidden. For instance: preset words, digits, and seven-segment displays, as in a digital clock, are all good examples of devices with these displays. They use the same basic technology, except that arbitrary images are made from a matrix of small pixels, while other displays have larger elements. LCDs can either be normally on (positive) or off (negative), depending on the polarizer arrangement. For example, a character positive LCD with a backlight will have black lettering on a background that is the color of the backlight, and a character negative LCD will have a black background with the letters being of the same color as the backlight. Optical filters are added to white on blue LCDs to give them their characteristic appearance.

CHAPTER 4

SECURITY SYSTEM

BY

FACE RECOGNITION

4 Security system by Face Recognition

4.1 Backgrounds

Authentication is one of the major challenges of the information systems era.

From Among other things, recognizing the human face is one of the known techniques that can be user authentication.

The target of this system is It can detect intruders into restricted or highly secure areas, and help reduce human errors.

4.2 System components

This system consists of two parts: the hardware part and the software part.

4.2.1 Hardware Part

The device consists of a laptop camera and the first use of it is to take pictures of the trusted person to be stored in the Database of the trusted people

The second use of it is to know whether the person who uses the car is a trusted person or an undesirable person to drive the car.

4.2.2 Software Part

While the software part consists of the face detection and facial recognition algorithms program.

The face was detect and recognize using OpenCV.

4.2.2.1 What is OpenCV ?

In the field of Artificial Intelligence, Computer Vision is one of the most interesting and Challenging tasks. Computer Vision acts like a bridge between Computer Software and visualizations around us.

It allows computer software to understand and learn about the visualizations in the surroundings.

For Example: Based on the color, shape and size determining the fruit. This task can be very easy for the human brain however in the Computer Vision pipeline, first we gather the data, then we perform the data processing activities and then we train and teach the model to understand how to distinguish between the fruits based on size, shape and color of fruit.

Currently, various packages are present to perform machine learning, deep learning and computer vision tasks.

By far, computer vision is the best module for such complex activities.

OpenCV is an open-source library, It is supported by various programming languages such as R, Python.

It runs on most of the platforms such as Windows, Linux and MacOS.

4.2.2.2 What is Face Detection?

In computer vision, one essential problem we are trying to figure out is to automatically detect objects in an image without human intervention. Face detection can be thought of as such a problem where we detect human faces in an image. There may be slight differences in the faces of humans but overall, it is safe to say that there are certain features that are associated with all the human faces. There are various face detection algorithms but Viola-Jones Algorithm is one of the oldest methods that is also used today.

Face detection is usually the first step towards many face-related technologies, such as face recognition or verification. However, face detection can have very useful applications.

4.2.2.3 What is Face Recognition?

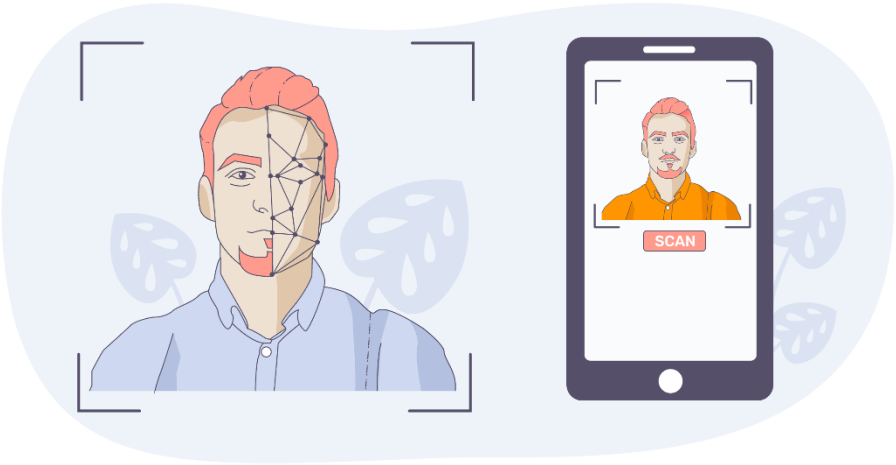


Figure 4-1 Face Recognition.

Face recognition is a method of identifying or verifying the identity of an individual using their face. There are various algorithms that can do face recognition but their accuracy might vary.

Here I am going to describe how we do face recognition using deep_learning.

So now let us understand how we recognise faces using deep learning.

We make use of face embedding in which each face is converted into a vector and this technique is called deep metric learning.

As shown in Figure 4-2 we divide this process into three simple steps for easy understanding:

Figure 4-2 deep metric learning.

- **Face Detection:**

The first task we perform is detecting faces in the image or video stream. Now that we know the exact location/coordinates of face, we extract this face for further processing ahead.

- **Feature Extraction:**

Now that we have cropped the face out of the image, we extract features from it. Here we are going to use face embeddings to extract the features out of the face. A neural network takes an image of the person's face as input and outputs a vector which represents the most important features of a face As shown in Figure 4-2. In machine learning, this vector is called embedding and thus we call this vector as face embedding.

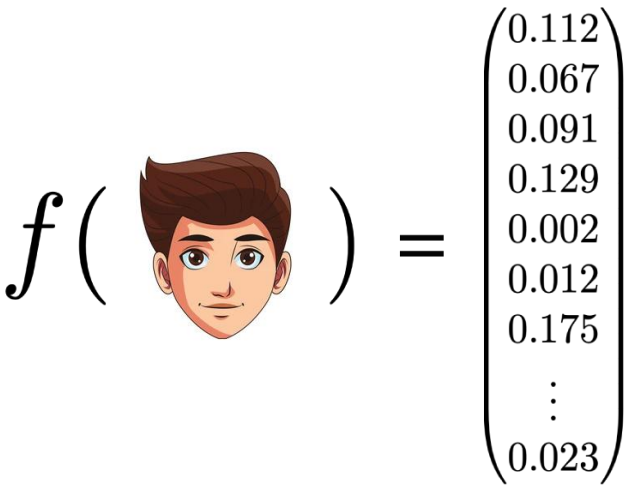


Figure 4-3 function which takes an image of a face as input and outputs a vector of the most important face features.

Now how does this help in recognizing faces of different persons?

While training the neural network, the network learns to output similar vectors for faces that look similar.

For example, if I have multiple images of faces within different timespan, of course, some of the features of my face might change but not up to much extent.

So in this case the vectors associated with the faces are similar or in short, they are very close in the vector space. Take a look at the below diagram for a rough idea:

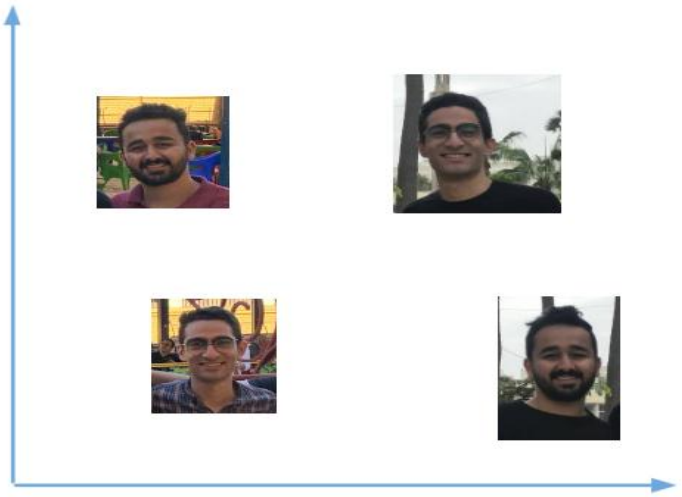


Figure 4-4 Images before training.

Now after training the network, the network learns to output vectors that are closer to each other(similar) for faces of the same person(looking similar). The above vectors now transform into:

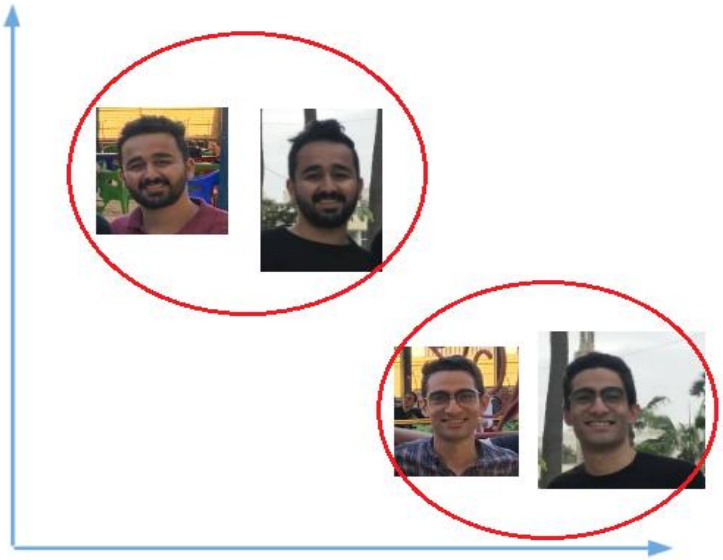


Figure 4-5 Images after training

We are not going to train such a network here as it takes a significant amount of data and computation power to train such networks. We will use a pre-trained network trained by Davis King on a dataset of ~3 million images. The network outputs a vector of 128 numbers which represent the most important features of a face.

Now that we know how this network works, let us see how we use this network on our own data. We pass all the images in our data to this pre-trained network to get the respective embeddings and save these embeddings in a file for the next step.

- **Face classification:**

Now that we have face embeddings for every face in our data saved in a file, the next step is to recognise a new image that is not in our data. So the first step is to compute the face embedding for the image using the same network we used above and then compare this embedding with the rest of the embeddings we have. We recognise the face if the generated embedding is closer or similar to any other embedding as shown below:

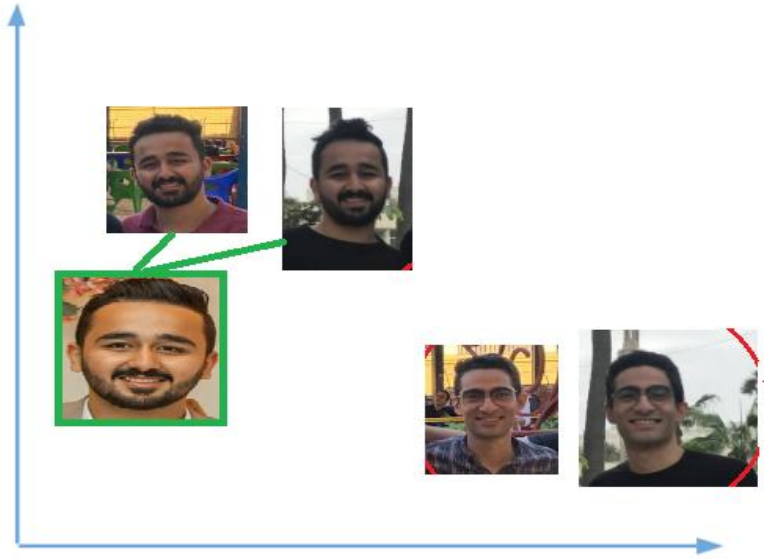


Figure 4-6 face classification

So we passed image In our example above, we saved the embeddings of Mostafa. Thus when we compared the new embedding with the existing one, the vector for Mostafa is closer to the other face embeddings of Mostafa

4.3 Preformance

Table 4.2.2.3-1 Preformance

DataBase	Test	preformance
20 Input Photos	100 Person	94.3 %

CHAPTER 5

SYSTEM INTERFACE

5 System Interface

Interface is the external environment that the microcontroller deals with, from which the external system is configured, either through direct contact with the controller or through a protocol

5.1 SWCs

Software components It is the software part necessary for the work of the external environment of the micro, through which communication to and from the controller of external devices is carried out, and it communicates internally with I/Os (input-output) memory.

SW drivers:

1-DIO (Digital input output) Driver.

2-PORT Driver.

5.1.1 DIO Driver

A Digital Input Output is a peripheral that deals with digital signals, either by generating a digital signal (Output Mode) or by receiving it (Input Mode). The basic block unit for the DIO pin is the Tri-State Buffer. Any DIO pin is consisting of a Tri-State buffer as a main component. The Tri-State buffer controls the direction of the data, A to B or B to A.

Figure 5-1 Tri-State Buffer.

5.1.2 PORT Driver

The same peripheral (Digital input output) that deals with signals,

But this SWC is responsible for dealing with only input signals to handle it & read it from the DDR register.

Registers of DIO peripheral:

1- DDR (Data Direction Register) in this register we can define the pin is output or input Set 0 Set 1 Input Output

2- PORT : This register is used in output mode to set the digital output value set 1 this pin carry 5v set 0 this pin carry 0v

3- PIN: we use this register in case the pin is defined input if 1 The Pin is connected to 5v if 0 The Pin in connected to 0v

>> The size of each register is 8 bit, every bit corresponding to 1 Pin of the port.

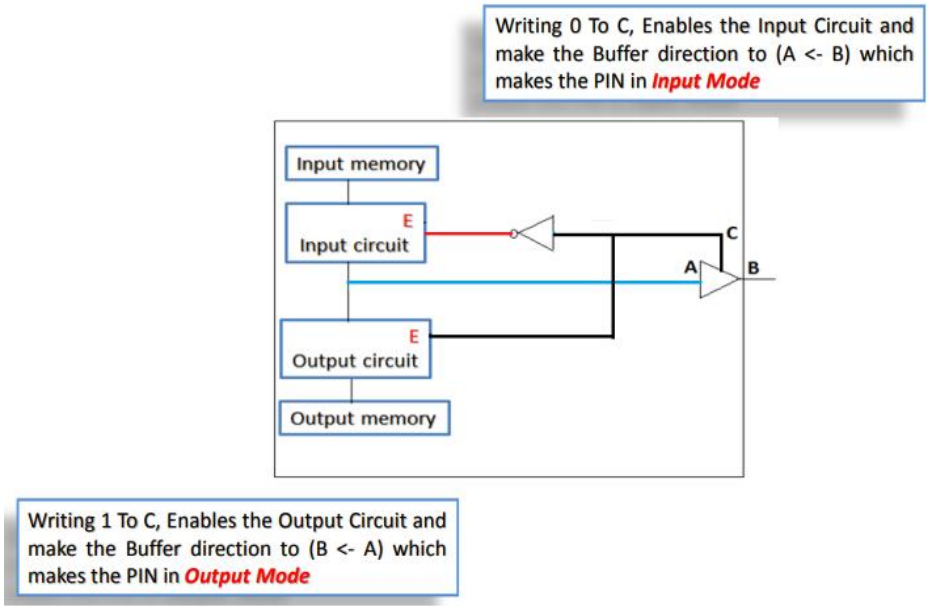


Figure 5-2 Port driver.

5.2 Hardware Interface

- LCD
- MCP2515
- TTL

The interface connection of the system through this diagram :

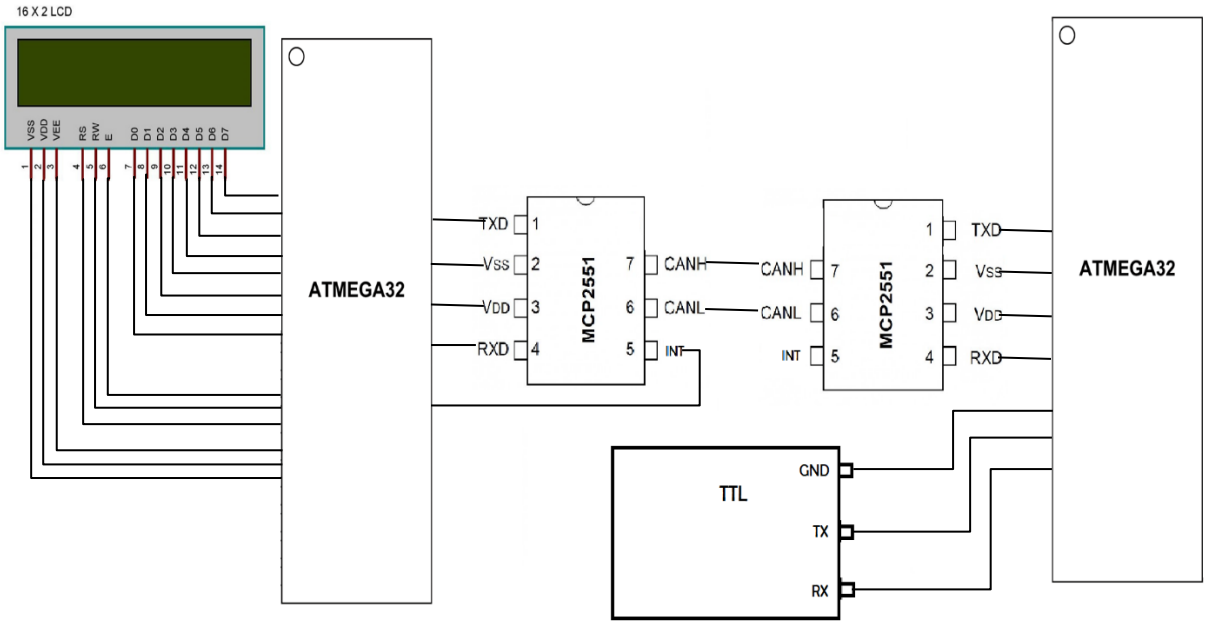


Figure 5-3 SYSTEM INTERFACE.

CHAPTER 6

COMMUNICATION

6 Communication

6.1 Communication concepts

Communication Protocols are a set of rules that allow two or more communication systems to communicate data via any physical medium.

The rules, regulations, synchronization between communication systems, syntax to be followed and semantics are all defined by the term protocol.

Protocols can be implemented by both hardware and software or combination of both. Analog and digital communication systems use various communication protocols widely.
In addition, each protocol has its own application area.
Embedded System is an electronic system or device which employs both hardware and software. A processor or controller takes input from the physical world peripherals like sensors, actuators etc., processes the same through appropriate software and provides the desired output.
In Most of the complex systems, the functionality is divided into subsystems, each subsystem is an embedded system with microcontroller and it is called ECU (Electronic Control Unit).

These ECUs need to share the data between each other, which mean. they need to communicate with each other.

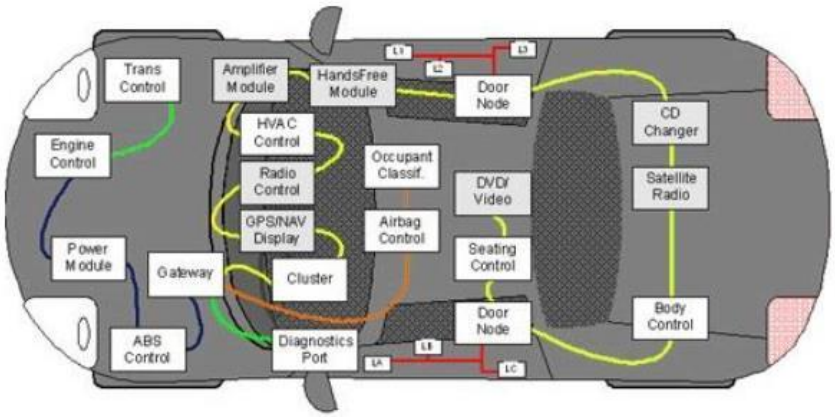


Figure 6-1 Car ECUs.

We need communication in embedded systems for:

- Exchanging data between different subsystems within the same system.
- Reduce the complexity of a system by splitting it into different subsystems.
- transfer the data on different distances and on different mediums.

6.2 Communication Protocols types

Communication protocols are broadly classified into two types:

- Inter System Protocol
- Intra System Protocol

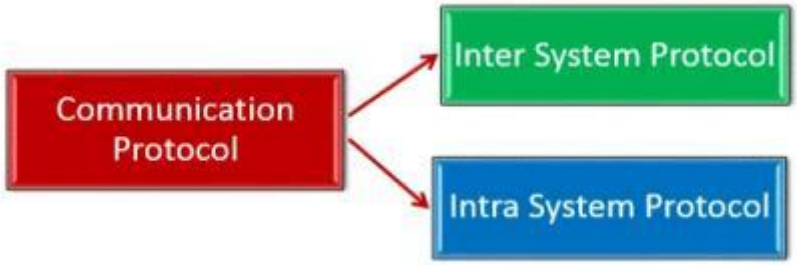


Figure 6-2 Communication protocols.

6.2.1 Inter System Communication Protocols

Inter system protocols establish communication between two communicating devices like between PC and microprocessor kit, developmental boards, etc. In this case, the communication is achieved through inter bus system.

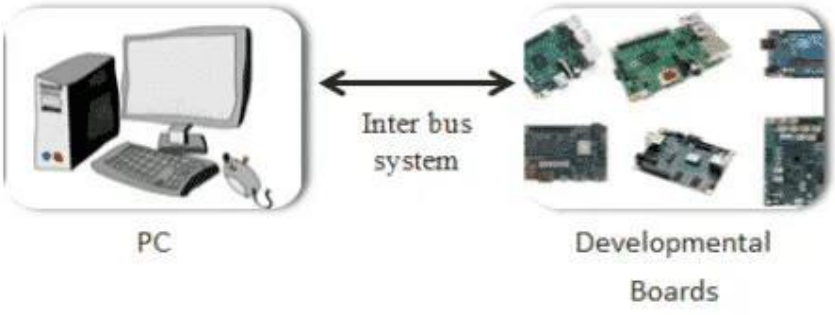


Figure 6-3 Inter system protocols.

Inter system protocol can be categorized into:

- USB Communication protocols
- UART Communication protocols
- USART Communication protocols

6.2.2 Intra System Communication Protocols

The Intra system protocol establishes communication between components within the circuit board.

In embedded systems, intra system protocol increases the number of components connected to the controller.

Increase in components lead to circuit complexity and increase in power consumption.

Intra system protocol promises secure access of data from the peripherals.

Intra system protocol can be categorized into:

- I2C Communication Protocol
- SPI Communication Protocol
- CAN Communication Protocol

6.3 communication protocols specification

6.3.1 Medium

In the communication process, a medium is a channel or system of communication the means by which information is transmitted between the sender and the receiver. The term is also known as a channel and it could be wired or wireless.

Figure 6-4 medium.

To choose the most suitable medium for any application we need to make a fair comparison between both options.

6.3.1.1 Speed

Both wired and wireless systems can achieve high speed communication. However, wired
communication tend to be void of any dead spots that are occasionally present in wireless
communications.

A wired network is never weighted down by unexpected or unnecessary traffic.

6.3.1.2 Security

Wireless communications are less secure than wired communications since the communication signal are transmitted through the air.
Any unauthorized user is unable to connect to the wireless network unless their devices is connected using wires if the network was properly secured.

6.3.1.3 Cost

Wireless communication has lower cost than wired systems.
Wires might be very expensive like optical fiber.

6.3.1.4 Installation

Wired communications installations can take longer to set up because more components are required to complete the process.
Wireless communications installations are less complex and easier.

6.3.1.5 Mobility

Wireless communications allow you to be more mobile with the flexibility to access the network from any location with no hassles with cables.

• Factor	wired	wireless
• Speed	High	Low
• security	More secure	Less secure
• cost	High	Low
• installation	complex	Easy
• mobility	limited	Not limited

6.3.2 Transmation technique

Serial communication **Vs** parallel communication.

6.3.2.1 Serial communication

In serial communication, the data bits are transmitted serially over a common communication link one after the other.
It does not allow simultaneous transmission of data because only a single channel is utilized. So, it can be connected between two points that are separated at a large distance with concerning each other, but it will take more time.

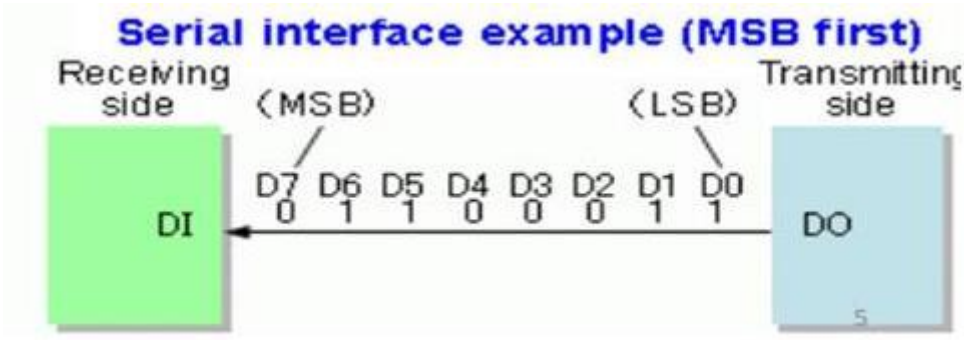


Figure 6-5 serial communication.

6.3.2.2 Parallel communication

In parallel communication, the various data bits are simultaneously transmitted using multiple communication links between the sender and receiver.

Despite using a single channel between the sender and receiver, various links are used and each bit of data is transmitted separately over the entire communication link.

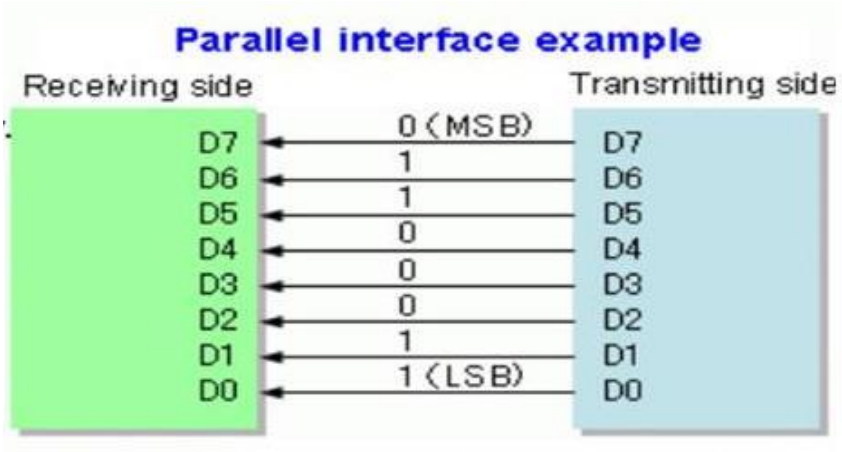


Figure 6-6 parallel communication.

Despite parallel communication looks better, serial communication is the most used because of the parallel communication drawbacks like:

1. Complex connection
2. Cross talk
3. Data skew

6.3.3 Node to node relationship

There are two types of node-to-node relationships:

1. Pear to pear
2. Master and slave

6.3.3.1 Pear to pear

In this type of communication the communicating nodes can send to each other any time with no privileges.

6.3.3.2 Master and slave

In this type of communication there is a master node that can send data to any other nodes (Slaves).
The master is the only node that can initiate the communication, the slave can never initiate the communication.
The slave can send data to master only when the master permits the slave to send.
The Master / Slave network can be divided to:

- Single Master Single Slave (SMSS)
- Single Master Multi Slave (SMMS)
- Multi Master Multi Slave (MMMS)

6.3.4 Synchronization

Synchronous communication **Vs** Asynchronous communication.

6.3.4.1 Synchronous communication

It is a type of communication the nodes of the network share the same clock.

Figure 6-7 Synchronous communication.

6.3.4.2 Asynchronous communication

It is a type of communication that doesn't have a shared clock between the communicating

nodes, instead the nodes are configured with the communicating rate and each node is having its own clock generator system that generates this clock.

Figure 6-8 Asynchronous communication.

6.3.5 Data Direction

Simplex **Vs** Half Duplex **Vs** Full Duplex.

6.3.5.1 Simplex channel

The data is in one direction, from a transmitter to receiver.

6.3.5.2 Half duplex channel

The data is bidirectional, each node can transmit and receive but not in the same time.

6.3.5.3 Full duplex channel

The data is bidirectional and each node can transmit and receive at the same time.

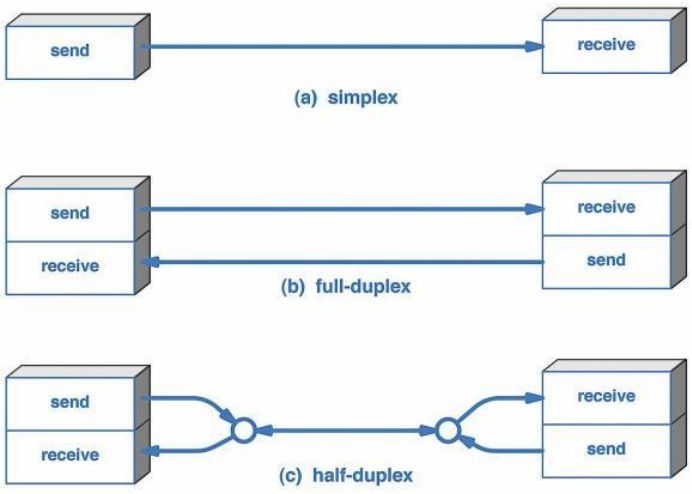


Figure 6-9 Data Direction.

6.3.6 Throughput

It deals with the efficiency of a system.

It defined as the percentage of data bits to the frame size.

It can be defined as the rate of production or operation of a defined process over a stated period.

In communication, the frame may include some information that are not the data like the address of the receiver.

6.4 protocols we are going to use

In our system we use three main communication protocols:

• UART Communication Protocol

• SPI Communication Protocol

• CAN Communication Protocol

In each protocol we are going to talk about its concepts, type, specifications and role in the project.

6.4.1 UART Communication Protocol

UART stands for Universal Asynchronous Receiver Transmitter.

It is a serial communication protocol that consists of one wire for transmitting data and one wire to receive data.

A common parameter is the baud rate known as "bps" which stands for bits per second.

If a transmitter is configured with 9600bps, then the receiver must be listening on the other end at the same speed.

6.4.1.1 UART Concepts

What is the difference between Bit Rate and Baud Rate?

The bit rate is the number of bits transmitted per second, whereas, the baud rate is the

number of symbols transmitted per second.

The symbol is a signal unit that is defined by the protocol, it may be one bit or more.

Therefore, baud rate is always less than or equal to the bit rate but never greater.

In UART, the symbol is one bit, therefore the baud rate is equals to the bit rate in UART Communication protocol.

6.4.1.2 UART Specification

Specifications:

- **Medium:** Wired.
- **Transmation technique:** Serial.
- **Node to Node relationship:** Pear to Pear.
- **Synchronization:** Asynchronous.
- **Data Direction:** Full Duplex.
- **Throughput:** minimum throughput = 55% and maximum throughput = 81%

6.4.1.3 UART Hardware Interface

Each node has a line called Tx (Transmission line) and another one called Rx (Receive Line).

The Tx of one node shall be connected to Rx of the other node and vice versa.

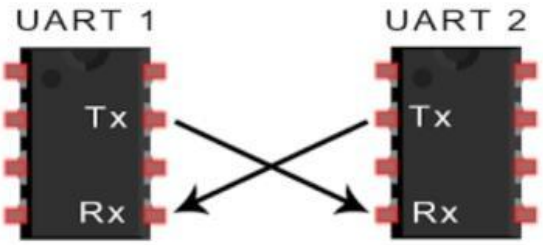


Figure 6-10 UART Hardware interface.

6.4.1.4 UART Data Frame Format

Frame Format:

- **Start bit:** 1 bit indicates the start of a new frame, always logical low.
- **Data:** 5 to 9 bits of sent data.

• **Parity bit**: 1 bit for error checking.

• **Even parity**: clear parity bit if number of 1s sent is even.

• **Odd parity**: clear parity bit if number of 1s sent is odd.

• **Stop bit**: 1 or 2 bits indicate end of frame, always logic high.

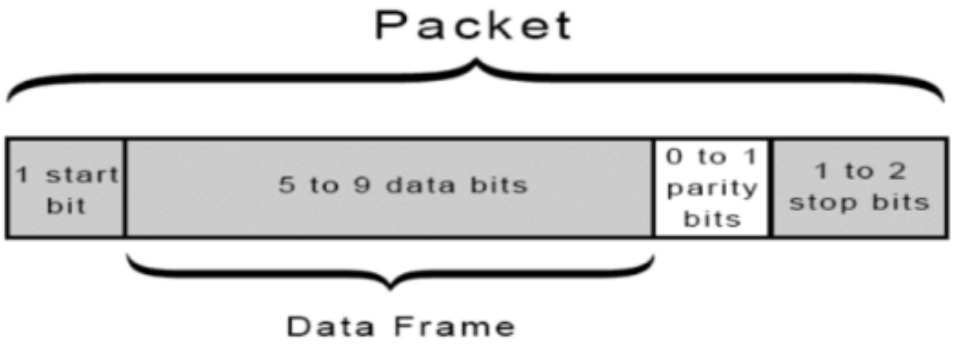


Figure 6-11 Frame Format.

6.4.1.5 UART type

UART is Inter System communication Protocol.

6.4.2 SPI Communication Protocol

SPI stands for Serial Peripheral Interface.

SPI bus is a synchronous serial communication interface specification used for short distance communication.

The SPI bus can operate with a single master device and with one or more slave devices.

6.4.2.1 SPI Concepts

The SPI operation is a shift register operation, so there is no frame format.

The master swaps a bit with the slave every clock cycle.

6.4.2.1.1 Steps of SPI data transmission

The master sends the data one bit at a time to the slave along the MOSI line. The slave reads the bits as they are received.

If a response is needed, the slave returns data one bit at a time to the master along the MISO line, then the master reads the bits as they are received.

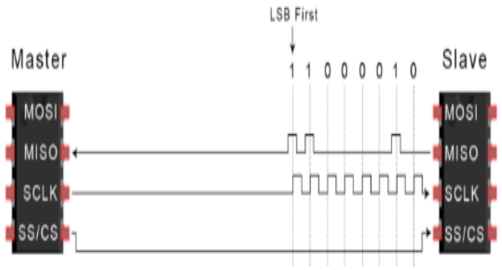


Figure 6-12 SPI data transmission.

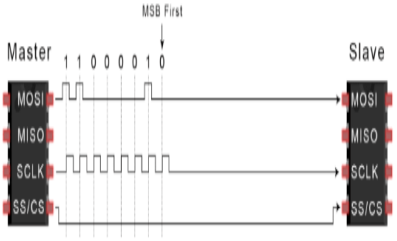
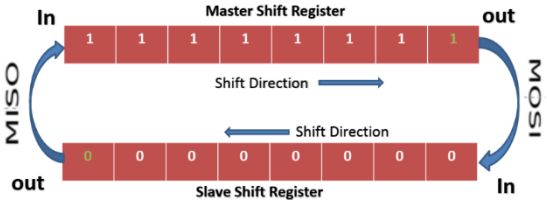


Figure 6-13 SPI data transmission.

6.4.2.1.2 The interconnection between master and slave

The SPI Master initiates the communication cycle when pulling low the Slave Select (SS) pin of the desired Slave.

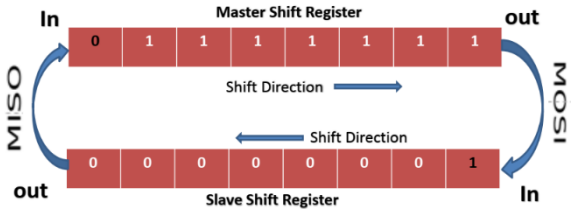
Master and Slave prepare the data to be sent in their respective Shift Registers.



6.4.2.1.3 SPI data transmission and receiver

the Master generates the required clock pulses on the SCK line to interchange data.

Data is always shifted from Master to Slave on MOSI, line, and from Slave to Master on MISO, line.



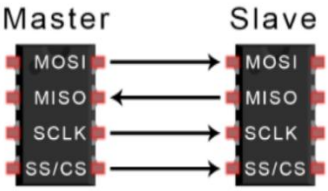
6.4.2.2 SPI Specification

Specifications:

- **Medium:** Wired.
- **Transmation technique:** Serial.
- **Node to Node relationship:** Master and salve (SMMS).
- **Synchronization:** Synchronous.
- **Data Direction:** Full Duplex.
- **Throughput:** Throughput =100%.

6.4.2.3 SPI Hardware Interface

• SCLK: Serial Clock (output from master).
• MOSI: Master Output, Slave Input (output from master).
• MISO: Master Input, Slave Output (output from slave).
• SS: Slave Select (active low, output from master).



6.4.2.4 SPI Clock

The clock signal synchronizes the output of data bits from the master to the sampling of bits by the slave.

One bit of data is transferred in each clock cycle, so the speed of data transfer is determined by the frequency of the clock signal.

SPI communication is always initiated by the master since the master configures and generates the clock signal.

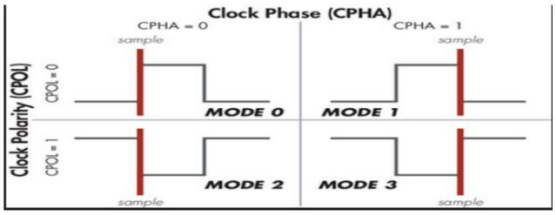
SPI Clock Parameters:

Clock Polarity: Defines the idle state of the clock.

Clock Phase: Define the leading action to be taken.

Because the clock has 2 edges; rising and falling edge, it is configurable to choose what to be done with the first edge (Called also Leading Edge).

The master can choose to send data (Called also Toggle or Setup) with the leading edge, or to choose to read data with leading edge (called also Sample).



6.4.3 CAN Communication Protocol

CAN stands for Controller Area Network.

Even today, CAN is still performing useful services in motor vehicles in networking ECUs in the powertrain, chassis and convenience areas.

Above all, CAN is characterized by very reliable data transmission that satisfies the real-time requirements of target usage areas.

6.4.3.1 CAN Concepts

6.4.3.1.1 CAN network

A CAN network consists of a number of CAN nodes which are linked via a physical transmission medium (CAN bus).

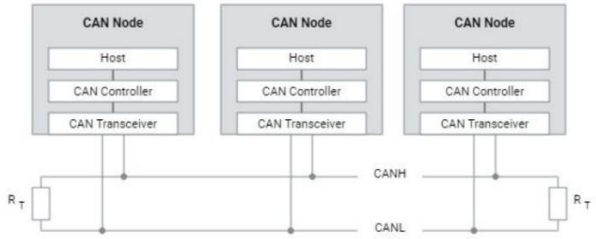
In practice, the CAN network is usually based on a line topology with a linear bus to which a number of electronic control units are each connected via a CAN interface.

A twisted two-wire line is the physical transmission medium used.

A CAN transceiver always has two bus pins, one for the CAN high line (CANH) and one for the CAN low line (CANL).

Physical signal transmission in a CAN network is based on transmission of differential voltages (differential signal transmission).

This effectively eliminates the negative effects of interference voltages induced by motors, ignition systems and switch contacts.



An electronic control unit (ECU) that wants to participate in CAN communication requires a CAN interface.

This comprises a CAN controller and a CAN transceiver.

The CAN controller fulfills communication functions prescribed by the CAN protocol, which relieves the host considerably.

The CAN transceiver connects the CAN controller to the physical transmission medium.

Usually, the two components are electrically isolated by optical or magnetic decoupling, so that although overvoltages on the CAN bus may destroy the CAN transceiver, the CAN controller and the underlying host are preserved.

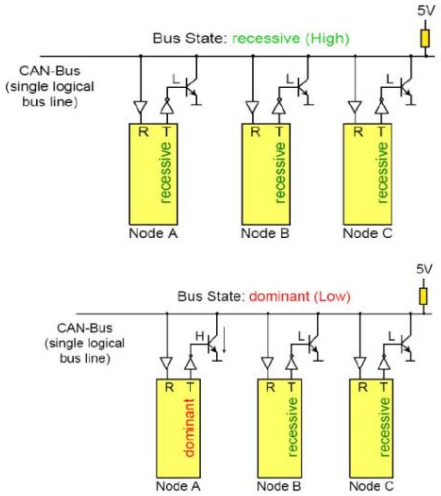
6.4.3.1.2 CAN bus logic

A basic prerequisite for smooth communication in a CAN network is clear distinctions between dominant and recessive bus levels.

The dominant bus level corresponds to logical “0”, and the recessive bus level corresponds to logical “1”,and the dominant bus level overwrites the recessive bus level.

When different CAN nodes send dominant and recessive bus levels simultaneously, the CAN bus assumes the dominant bus level.

The recessive bus level only occurs if all CAN nodes send recessive levels.



6.4.3.1.3 Communication principle

CAN network is based on a combination of multi-master architecture and line topology, essentially each CAN node is authorized to place CAN messages on the bus in a CAN network.

The transmission of CAN messages does not follow any predetermined time sequence, rather it is event-driven.

A method of receiver selective addressing is used in a CAN network to prevent dependencies between bus nodes and thereby increase configuration flexibility.

Every CAN message is available for every CAN node to receive (broadcasting).

A prerequisite is that it must be possible to recognize each CAN message by a message identifier (ID) and node-specific filtering.

Although this increases overhead, it allows integration of additional CAN nodes without requiring modification of the CAN network.

6.4.3.1.4 Addressing

Each can node stores a database called CAN DB. This database identify the owner of each message and the receivers of it.

Only one owner per message ID is allowed. So that only one node can send a certain message.

However, the message can be received by many nodes. Once a node send a message, all other nodes receive the message and check their DB, if the received ID matches a record in the database then the node continue listening to the frame. Otherwise, the node would neglect the frame.

6.4.3.1.5 Arbitration

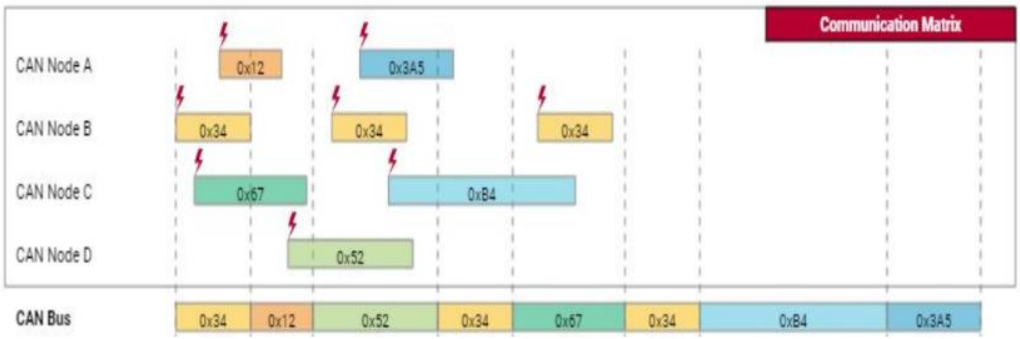
In case of simultaneous bus access, a bitwise bus arbitration ensures that the highest priority CAN message among the CAN nodes prevails.

In principle, the higher the priority of a CAN message the sooner it can be transmitted on the CAN bus.

In case of poor system design, low priority CAN messages even run the risk of never being transmitted.

All CAN nodes wishing to send place their identifier of the CAN message bitwise onto the CAN bus, from most significant to least significant bit.

In this process, the wired-AND bus logic upon which the CAN network is based ensures that a clear and distinct bus level results on the bus.



6.4.3.2 CAN Specefication

Specifications:

- **Medium:** Wired.
- **Transmation technique:** Serial.
- **Node to Node relationship:** Master and salve (MMNS).
- **Synchronization:** Asynchronous.
- **Data Direction:** Half Duplex.
- **Throughput:** minimum throughput = 0% and maximum throughput = 40%

6.4.3.3 CAN data frame format

6.4.3.3.1 Data frame

Data frames serve to transmit user data.

A data frame is made up of many different components, each individual component carries out an important task during transmission.

Tasks to be performed are: Initiate and maintain synchronization between communication partners, establish the communication relationships defined in the communication matrix and transmit and protect the user data.

SOF:

Transmission of a data frame begins with the start bit (Start of Frame — SOF).

It is transmitted by the sender as a dominant level which produces a signal edge from the previous recessive (bus idle) level which is used to synchronize the entire network.

In order for the receivers not to lose synchronism to the sender during transmission of the frame, they compare all recessive-to-dominant signal edges with their preset bit timing. In case of deviation, receivers re-synchronize by the amount of the relevant phase error (re-synchronization).

ID:

Following the SOF is the identifier (ID).

This sets the priority of the data frame, and together with the acceptance filtering it provides for sender-receiver relations in the CAN network that are defined in the communication matrix.

RTR:

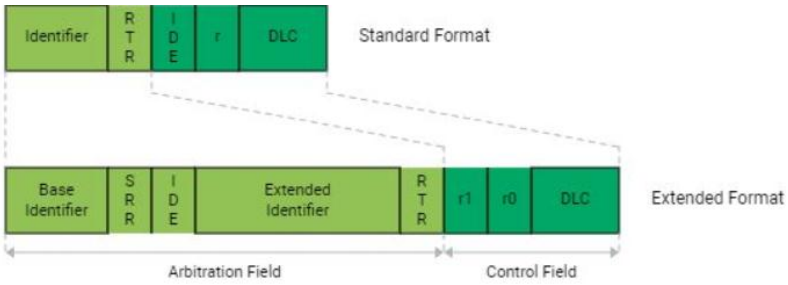
Next comes the RTR bit (Remote Transmission Request).

It is used by the sender to inform receivers of the frame type (data frame or remote frame). A dominant RTR bit indicates a data frame.

IDE:

The IDE bit (Identifier Extension bit) which follows serves to distinguish between standard format and extended format.

In standard format the identifier has 11 bits, and in extended format 29 bits.



DLC:

The DLC (Data Length Code) communicates the number of payload bytes to the receivers.

The payload bytes are transported in the data field. A maximum of eight bytes can be transported in one data frame.

CRC & ACK:

The payload is protected by a checksum using a cyclic redundancy check (CRC) which is ended by a delimiter bit.

Based on the results of the CRC, the receivers acknowledge positively or negatively in the ACK slot (acknowledgement) which also is followed by a delimiter bit.

EOF:

After this the transmission of a data frame is terminated by seven recessive bits (End Of Frame — EOF).

Reserved bits:

In classical CAN, reserved bits are dominant bits used for frame check.

IFS:

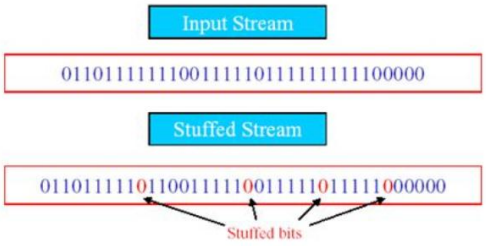
(Inter Frame Space — IFS) In classical CAN, reserved bits are dominant bits used for frame check.

6.4.3.3.2 Bit stuffing

Senders must transmit a complementary bit at the latest after transmitting five homogeneous bits; a stuff bit is added even if a complementary bit followed the five homogeneous bits anyway.

This technique is used for frame synchronization and to ensure the frame is correctly sent.

Bit stuffing begins with transmission of the SOF and ends with transmission of the last bit of the CRC sequence.



6.4.3.4 Error detecting, tracking and reporting

6.4.3.4.1 Error detecting

To detect corrupted messages, the CAN protocol defines five mechanisms: bit monitoring, Form Check, Stuff Check, ACK Check and Cyclic Redundancy Check.

Bit monitoring (Sender Task).

The sender compares the sent bit level with the actual bus level.

A bit error exists if the sender detects a discrepancy between the two levels, the sender then stop sending data and convert to be a listener.

Form check (Receiver Task).

The form check serves to check the format of a CAN message.

Each CAN message always exhibits the same bit sequences at certain positions. They are the CRC delimiter, ACK delimiter and EOF.

Senders always transmit these message components recessively, and a format error exists if a receiver detects a dominant bus level within one of these message components in the Form Check.

CRC Check (Receiver Task).

In the cyclic redundancy check (CRC) the polynomial R(x) associated with the arriving data or remote frame should equal a multiple of the generator polynomial G(x) specified by ISO 11898-1. If this is not the case (CRC error), then the data or remote frame was corrupted during its transmission.

ACK check (Sender Task).

The acknowledgement mechanism defined in the CAN protocol specifies that all receivers must acknowledge every arriving CAN message right after the cyclic redundancy check.

A single positive acknowledgement is sufficient to signal to the sender that at least one receiver received the CAN message it transmitted correctly, if not a single positive acknowledgement arrives at the sender, then an acknowledgement error has occurred (ACK error).

Stuff check (Receiver Task).

The stuff check serves to check the bit stream. The CAN protocol specifies that the sender must transmit a complementary bit after five homogeneous bits for synchronization purposes.

There is a stuffing error if more than five homogeneous contiguous bits are received.

6.4.3.4.2 Error reporting

The CAN protocol prescribes that if the error-detecting CAN node is experiencing a local disturbance, it must inform all CAN nodes connected to the CAN network.

The error-detecting CAN node transmits an error signal (error flag) for this purpose, which is made up of six dominant bits.

This is an intentional violation of the bit stuffing rule, and it generates a bit stuffing error.

Transmission of an error flag ensures that all other CAN nodes will also transmit an error flag (secondary error flag) and thereby also terminate the regular data transmission just like the sender of the primary error flag. Depending on the situation, the primary and secondary error flags might overlap.

Transmission of an error flag is always terminated by an error delimiter.This consists of eight recessive bits. The error delimiter replaces the ACK delimiter and the EOF of a regular message transmission, so that together with the obligatory transmission pause (ITM — Intermission) on the CAN bus, this results in eleven recessive bits (bus-idle identifier).

6.4.3.4.3 Error tracking

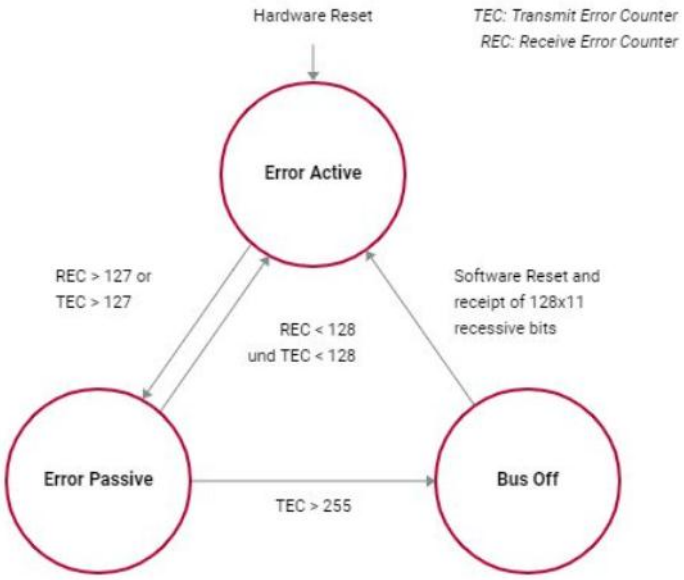
To assure network-wide data consistency, each node in a CAN network has the right to terminate any CAN message interpreted as faulty.

Consequently, each CAN controller has a TEC (Transmit Error Counter) and a REC (Receive Error Counter).

In case of successful transmission of a data or remote frame, the relevant error counter is decremented (TEC=TEC-1; REC=REC-1).

Detection and subsequent transmission of a error flag causes the relevant error counter to be incremented according to certain rules.

For the sender the following rule applies: TEC=TEC+8. Error-detecting receivers initially increment their REC by one unit (REC=REC+1). For the error causing receiver: REC=REC+8.



Active Error

Depending on the specific error count, a CAN controller handles switching of the error state.

After the start, a CAN controller assumes the normal state Error Active. In this state, the CAN controller sends six dominant bits (active error flag) after detecting an error.

When a limit is exceeded (TEC>127; REC>127), the CAN controllers switch over to the "Error Passive" state.

Passive Error

CAN controllers in the Error Passive state can only indicate a detected error by sending six homogeneous recessive bits.

This prevents the error-detecting receivers from globalizing detected errors. In addition, when sending two consecutive data or remote frames, CAN controllers that are in the "Error Passive" state must wait the "Suspend Transmission Time" (8 bits).

Bus Off

If a CAN controller fails or if there are extreme accumulations of errors, a state transition is made to the Bus Off state.

The CAN controller disconnects from the CAN bus, and the Bus-Off state can only be exited by intervention of the host (with a mandatory waiting time of 128 x 11 bits) or by a hardware reset.

CHAPTER 7

SOFTWARE ARCHITECTURE

7 Software Architecture

Software architecture refers to the fundamental structures of a software system and the discipline of creating such structures and systems. Each structure comprises software elements, relations among them, and properties of both elements and relations. The architecture of a software system is a metaphor, analogous to the architecture of a building It functions as a blueprint for the system and the developing project, laying out the tasks necessary to be executed by the design teams.

7.1 SW Design

7.1.1 SW Design process

Figure 7-1 SW Design process.

7.1.2 SW Design approaches

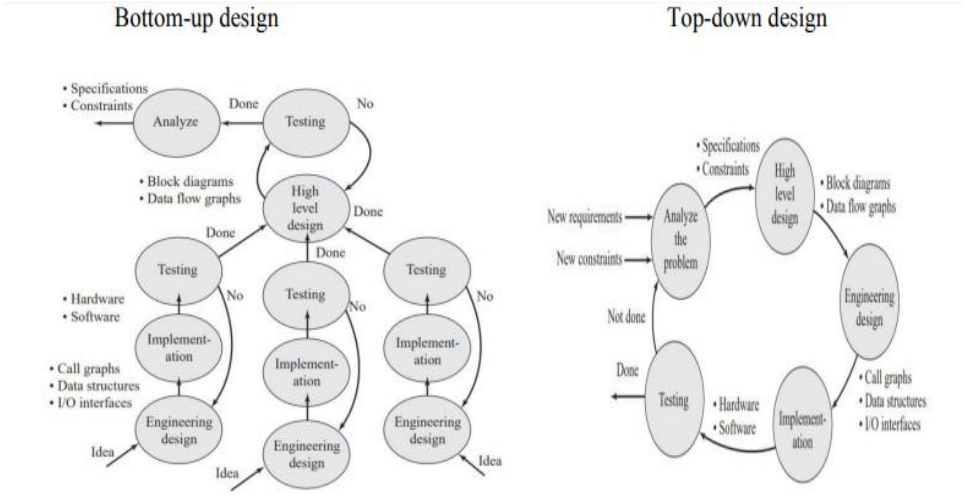


Figure 7-2 SW Design approaches.

7.1.3 SW Design Types

- Static Design
- Dynamic Design
- Logic Design

7.1.3.1 Static design

It represents some important concepts like modularity and architectural patterns.

• Modular programming define how SW system is divided into SW components.

• Architectural patterns: define how to organize modules.

➢ Layered architecture.

➢ Event triggered.

➢ Micro services

7.1.3.2 Logic design

What 's the role of physical design?

• Design folder structure for the project.

• Design make system for the project.

• Design the output from the project.

• Define files dependencies and implement constraints on them. How physical design reduces time to market?

• It optimizes compilation time. If a change occurred in the logic of a module, you do not have to recompile the hole project files again but recompile the changed module.
• It optimizes liking time. By creating a library for each folder. Link the object files in each folder into one file. At the final stages, link all libraries. So, if a change occurred in a folder, you recompile these files and link them to produce a library to be linked with other libraries.

• It optimizes the flashing timing by dividing the output file and define addresses for each part and if a change occurred, flash the changed part only.

7.1.3.3 Dynamic design

• Dynamic Design always answer how the modules talk with each other by implementing communication pattern which define 2

• It defines when module APIs' will be triggered.

➢ Event or Time triggered.

➢ Client server.

➢ Service pr Signal oriented.

7.2 Layered Architecture

7.2.1 Definition

The software architecture is a structuring way used to define software elements and relationships between them. In Embedded Systems we use a major type of software architecture called Layered Architecture.

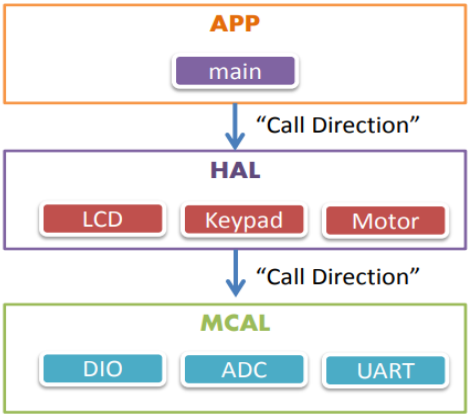


Figure 7-3 Layered Architecture.

MCAL : Microcontroller Abstraction Layer

Software related to any peripheral inside the microcontroller

HAL : Hardware Abstraction Layer

Software related to any on board hardware element

APP : Application Layer

System application (main)

7.2.2 Importance

1- Modularity

In a Layered architecture we separate the user application from the hardware drivers from the microcontroller specific drivers.

2- Portability

Changing any part of the software part would change its layer only. For example, if we need the same application with a new microcontroller, we shall only change the MCAL.

3- Reusability

Code could be easily reused in different applications and systems.

4- Maintainability

Debugging and Testing is now much easier in small parts of the software instead of having a very long and complex one.

7.2.3 Implementation

Figure 7-4 Implementation ECU1.

Figure 7-5 Implementation ECU2.

7.3 AUTOSAR

7.3.1 Definition

AUTOSAR (Automotive Open System Architecture) is a worldwide development partnership of vehicle manufacturers, suppliers, service providers and companies from the automotive electronics, semiconductor and software industry.

AUTOSAR aims to standardize the software architecture of Electronic Control Units (ECUs). AUTOSAR paves the way for innovative electronic systems that further improve performance, safety, and security.

Figure 7-6 AUTOSAR.

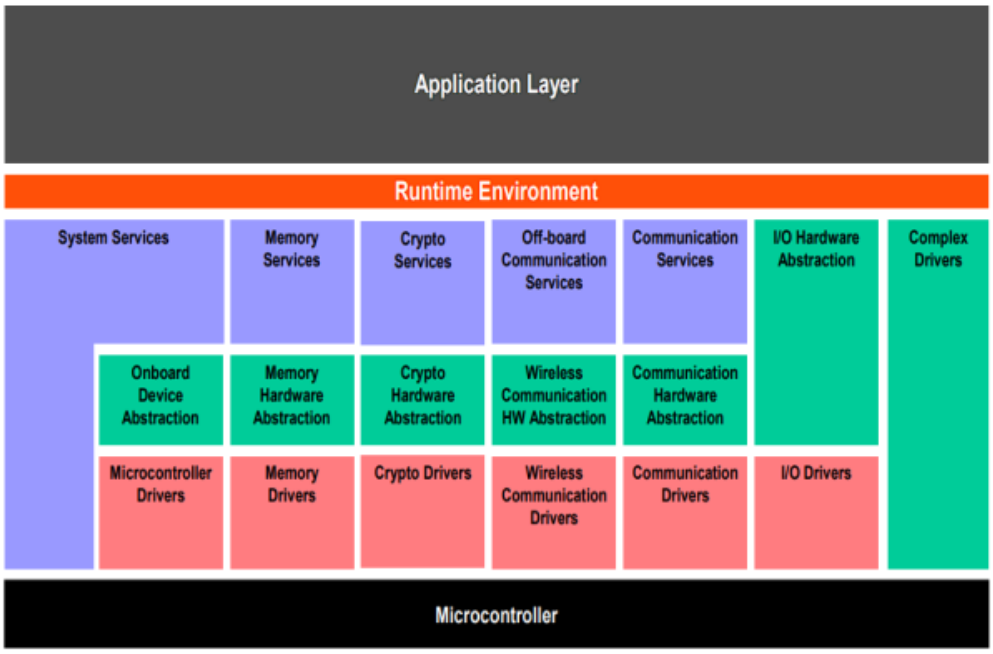


Figure 7-7 AUTOSAR.

7.3.2 Importance

• Hardware and software – widely independent of each other.

• Development can be decoupled (through abstraction) by horizontal layers, reducing development time and costs.

• Reuse of software enhances quality and efficiency.

• Establish development distribution among suppliers.

• Compete on innovative functions with increased design flexibility.

• Simplify software and system integration.

• Reuse software modules across OEMs.

• Increase efficiency of application development.

• Invent new business models.

• Interface with development processes.

- Embed tools into an overall tool environment.
- Enable new business models by means of standardized interfaces.
- Easily understand how automotive software is developed.

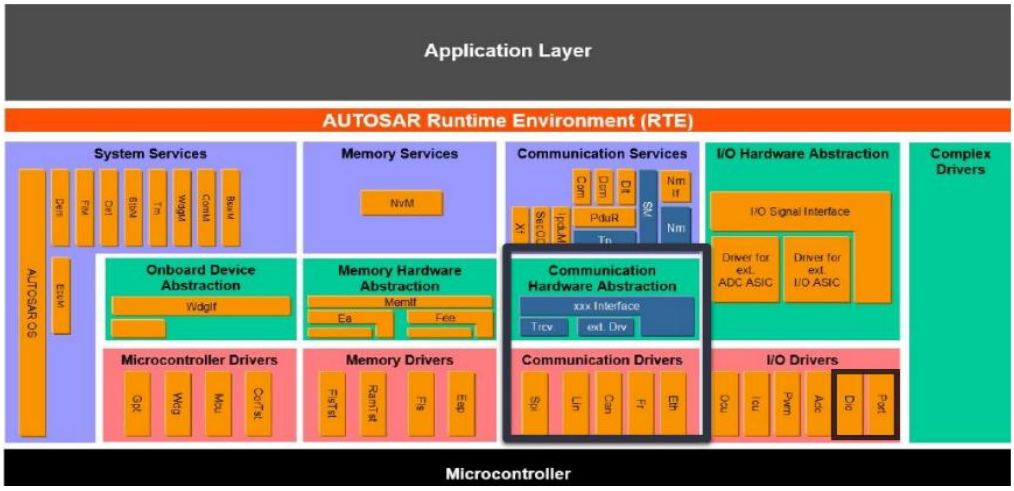


Figure 7-8 AUTOSAR.

CHAPTER 8

CONCLUSION

8 Conclusion

8.1 Overall system

8.1.1 FaceRecgnition

The system can be used to reduce the increased vehicle theft and allows the owner to identify the intruder thereby having the vehicle under his/her control.

The results obtained through the face recognition shows that it can be relied upon to ensure safety of vehicle.

The system is also reliable to be used in other authorization applications involving robotics, border management, banking security involving ATMs etc.

8.1.2 Communication between OS and Microcontroller

Communication between the OS and the microcontroller through the TTL through the UART protocol , Although it is not a standard protocol, it has an active role in this project, which is how to transfer data between two different programming languages.

8.1.3 Communication between ECU1 and ECU2

In this part, the standard communication protocol used in the automotive industry was used, which is the CAN to transfer data between units and each other.

8.1.4 Result

Reaching a final result by opening the car or not in front of the person using the car according to the database registered on the facial recognition system, thus achieving the highest protection for the car

8.2 Benfits

- Higher security
- Increase Technology in Automotive industry

REFERENCES

REFERENCES

AUTOSAR …….. https://www.autosar.org/

Automotive Book "Basic of Automobile"

Art of embedded book based on AVR

ATMEGA32 datasheet.

MCP2515 datasheet.

"Understanding and Using the Controller Area Network Communication Protocol" by A. Ghosal, Haibo Zeng, and Marco Di Natale.