**CHAPTER 4**

**SYSTEM SPECIFICATIONS**

**4.1. HARDWARE SPECIFICATION**

**4.1.1. IR SENSOR**: Infrared radiation is an electromagnetic wave with wavelength of 700nm to 1 mm. It is emitted by objects with temperature above 0 kelvin. Furthermore, intensity and wavelength of infrared radiation depends on the temperature of the object. The infrared sensors are the sensors that detect/measure infrared radiation or change in the radiation from outer source or inbuilt source. Also, sensors that uses the property of infrared radiations to detect the changes in surrounding are termed as infrared sensors.

A general working method of infrared sensors is as follows:

* **Infrared Source**: The source of infrared is either in built or from outside environment. Range of detection and wavelength of infrared radiation to be detected can be configured.
* **Transmission medium**: Vacuum, Air, Optical fibers.
* **Optical system**: To converge the infrared radiation into the detector. Optical lenses or mirrors. The material for optical system is chosen according to their transmittance/reflectance for desired wavelength of IR.
* **Detector**: Thermal detector has detector material and is independent to wavelength. Quantum detector is wavelength dependent.
* **Signal processing**: The signals produced by the detector are small so amplification is required.

Types of IR Sensors:

* **Passive infrared sensors**: They are basically Infrared detectors. Passive infrared sensors do not use any infrared source and detects energy emitted by obstacles in the field of view.
* **Active infrared sensors**: They are consist of two elements: infrared source and infrared detector. Infrared sources include an LED or infrared laser diode.Infrared detectors include photodiodes or phototransistors. The energy

emitted by the infrared source is reflected by an object and falls on the infrared detector.

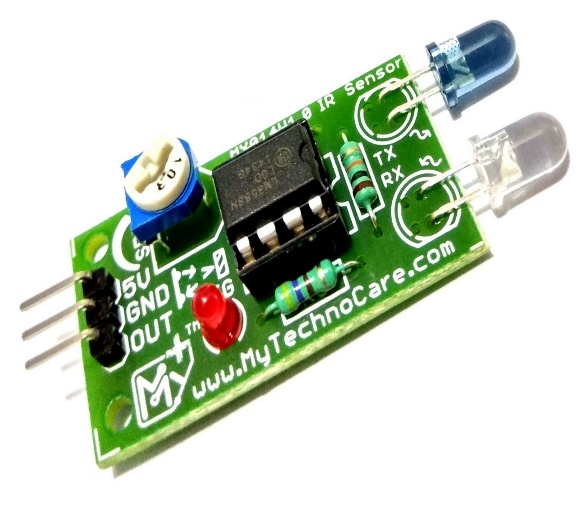
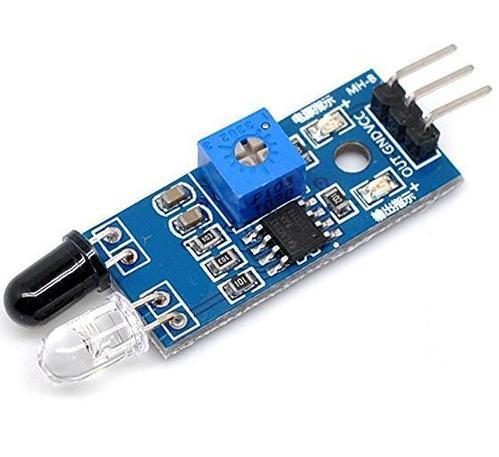


Figure 5: IR Obstacle Avoidance Sensor

IR Sensor consists of:

* **IR Transmitter**: Infrared Transmitter is a light emitting diode (LED) which emits infrared radiations. Hence, they are called IR LED’s. Even though an IR LED looks like a normal LED, the radiation emitted by it is invisible to the human eye. There are different types of infrared transmitters depending on their wavelengths, output power and response time.

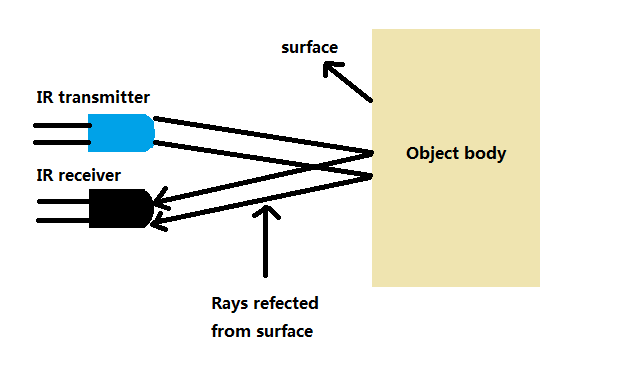
IR transmitters can be found in several applications. Some applications require infrared heat and the best infrared source is infrared transmitter. When infrared emitters are used with Quartz, solar cells can be made.

* **IR Receiver**: Infrared receivers are also called as infrared sensors as they detect the radiation from an IR transmitter. IR receivers come in the form of photodiodes and phototransistors. Infrared Photodiodes are different from normal photo diodes as they detect only infrared radiation.

Different types of IR receivers exist based on the wavelength, voltage, package, etc. When used in an infrared transmitter – receiver combination, the wavelength of the receiver should match with that of the transmitter.

The principle of an IR sensor working as an Object Detection Sensor can be explained using the following figure. An IR sensor consists of an IR LED and an IR Photodiode

together they are called as Photo – Coupler. When the IR transmitter emits radiation, it reaches the object and some of the radiation reflects back to the IR receiver. Based on the intensity of the reception by the IR receiver, the output of the sensor is defined.

Figure 6: IR Transmitter and Receiver

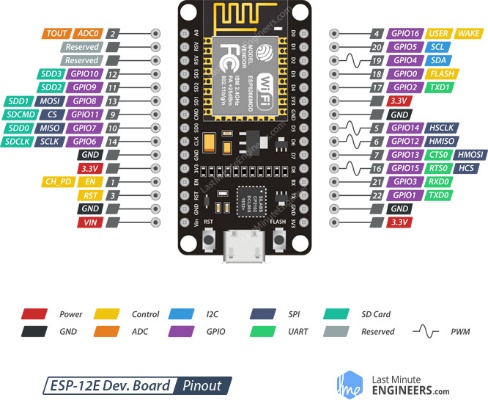
**4.1.3. NODEMCU ESP8266**: NodeMCU is an open-source firmware and development kit that helps you to prototype or build IoT product. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266.



Figure 7: NodeMCU ESP8266

MCU stands for Microcontroller Unit - which really means it is a computer on a single chip. A microcontroller contains one or more CPUs (processor cores) along with memory and programmable input/output peripherals. They are used to automate automobile engine control, implantable medical devices, remote controls, office machines, appliances, power tools, toys etc.

The ESP8266 NodeMCU has total 30 pins that interface it to the outside world. The connections are as follows

Figure 8: NodeMCU ESP8266 Pin Diagram

* **Power Pins**: There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.
* **GND**: This is a ground pin of ESP8266 NodeMCU development board.
* **I2C Pins**: They are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface

functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

* **GPIO Pins**: ESP8266 NodeMCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.
* **ADC Channel**: The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.
* **UART Pins**: ESP8266 NodeMCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.
* **SPI Pins**: ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:
* 4 timing modes of the SPI format transfer
* Up to 80 MHz and the divided clocks of 80 MHz
* Up to 64-Byte FIFO
* **SDIO Pins**: ESP8266 features Secure Digital Input/output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.
* **PWM Pins**: The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs, i.e., between 100 Hz and 1 kHz.
* **Control Pins**: They are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.
* **EN Pin**: The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
* **RST Pin**: This is used to reset the ESP8266 chip.
* **WAKE Pin**: This is used to wake the chip from deep-sleep.

**4.2. SOFTWARE REQUIREMENTS**

**4.2.1. ARDUINO (IDE)**: The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.



Figure 9: Arduino IDE

The source code for the IDE is released under the GNU General Public License, version- 2. The Arduino IDE supports the languages C and C++ using special rules of

code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

In October 2019 the Arduino organization began providing early access to a new Arduino Pro IDE with debugging and other advanced features.

**4.2.2. EMBEDDED C**: Embedded C Programming is the soul of the processor functioning inside each and every embedded system we come across in our daily life, such as mobile phone, washing machine, and digital camera. Each processor is associated with an embedded software. The first and foremost thing is the embedded software that decides functioning of the embedded system. Embedded C language is most frequently used to program the microcontroller.

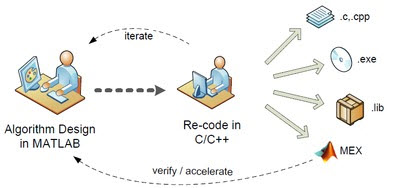
[](https://www.elprocus.com/wp-content/uploads/2014/02/13.jpg)

Figure 10: Embedded C

Earlier, many embedded applications were developed using assembly level programming. However, they did not provide portability. This disadvantage was overcome by the advent of various high level languages like C, Pascal, and COBOL. However, it was the C language that got extensive acceptance for embedded systems, and it continues to do so. The C code written is more reliable, scalable, and portable.

C language was developed by Dennis Ritchie in 1969. It is a collection of one or more functions, and every function is a collection of statements performing a specific task. C

language is a middle-level language as it supports high-level applications and low-level applications. Before going into the details of embedded C programming, we should know about RAM memory organization.

Salient features of the language:

* C language is a software designed with different keywords, data types, variables, constants, etc.
* Embedded C is a generic term given to a programming language written in C, which is associated with a particular hardware architecture.
* Embedded C is an extension to the C language with some additional files.

The embedded system designers must know about the hardware architecture to write programs. These programs play prominent role in monitoring and controlling external devices. They also directly operate and use the internal architecture of the microcontroller, such as interrupt handling, timers, serial communication and other available features.

Differences between C and Embedded C:

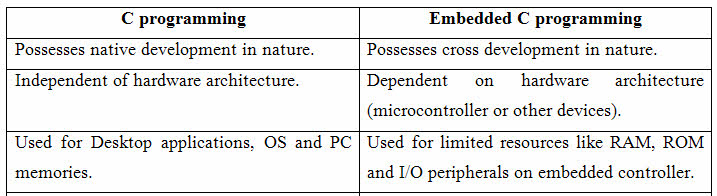
[](https://www.elprocus.com/wp-content/uploads/2014/02/22.jpg)

Table 1: Diff. b/w C and Embedded C

Advantages of embedded C program:

* Its takes less time to develop application program.
* It reduces complexity of the program.
* It is easy to verify and understand.
* It is portable in nature from one controller to another.

We hope that we have been successful in providing an easy and approachable way for the beginners of Embedded C programming. Better understanding of the Embedded C programming is the most essential prerequisite for designing embedded based project.

In addition to this, a better understanding and proper knowledge about embedded C programming help students immensely in the selection of a rewarding career.

**4.2.3. FIREBASE**: Firebase is a Backend-as-a-Service (BaaS) that started as a YC11 start-up and grew up into a next-generation app-development platform on Google Cloud Platform.

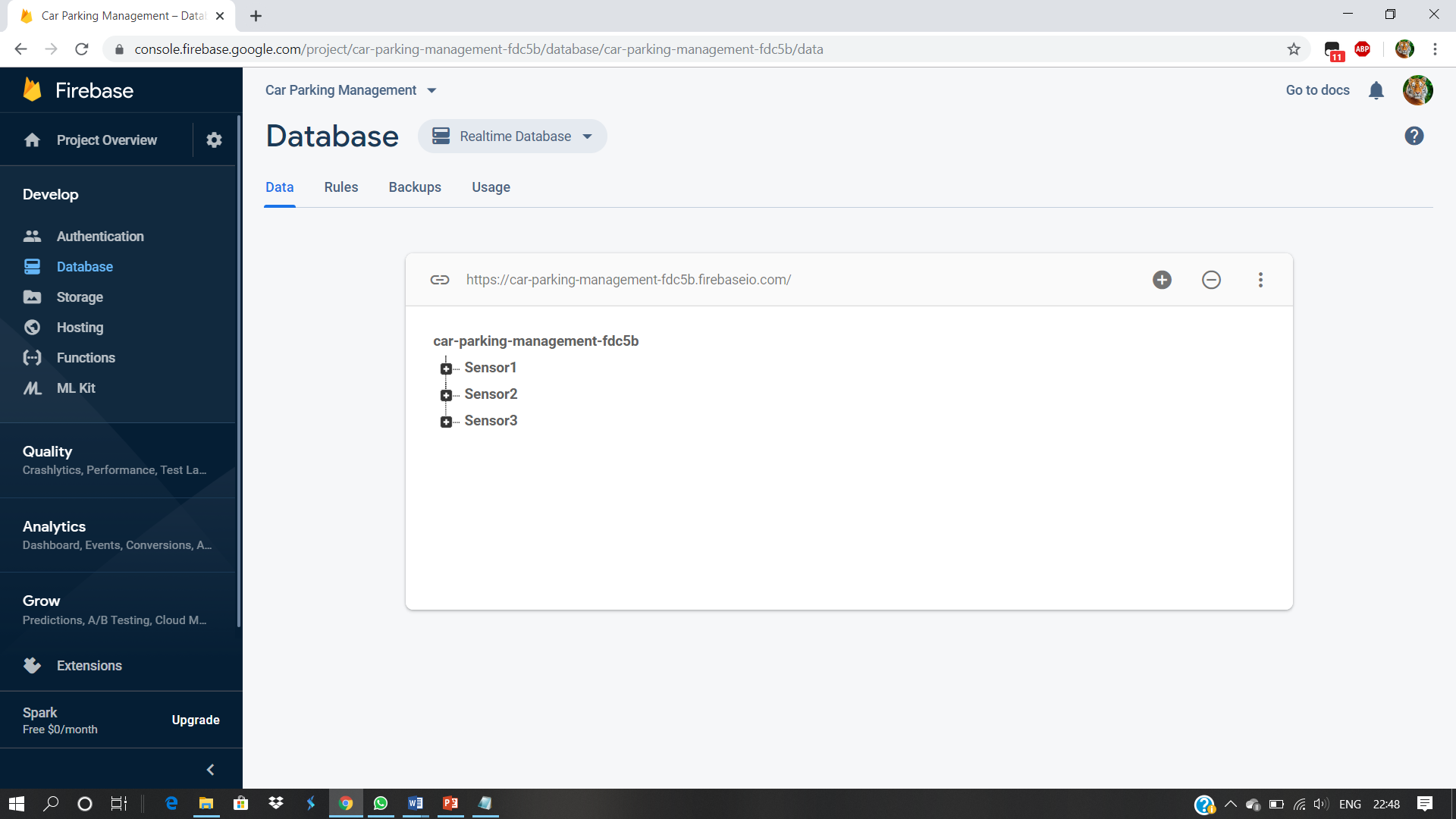


Figure 11: Firebase

So what is Firebase?

* **Real-time Database**: Real-time Database is a cloud-hosted database with the data stored as JSON (JavaScript Object Notation). It provides the real-time synchronization between every connected client - any changes or update is automatically sent to all users. The offline mode does not cause data loss as the Database SDK stores new data to disk and when after the connection is re-established, the synchronization takes place. The application server is not required for the access to Database from mobile device or web browser.
* **File Storage:** Firebase Storage provides a simple way to save binary files most often images, but it could be anything to Google Cloud Storage directly from

the client. Firebase Storage has it’s own system of security rules to protect your GCloud bucket from the masses, while granting detailed write privileges to your authenticated clients.

* **Authentication:** Authentication is a service for ensuring the protection of user’s data using a password, phone number, or various social authentications. It includes very convenient SDKs (Software Development Kits), ready-to-use UI libraries, and backend services. There are two ways to sign in your users: FirebaseUI, a drop-in solution that can be easily customized to match your app design - or Firebase Authentication SDK, a solution for manual implementation of several methods of sign-in.
* **Hosting:** Hosting is a web content hosting for developers. It ensures fast and secure deployment of web applications to a global content-delivery network (CDN). Hosting provides release management and version review by using one-click rollbacks.

Firebase Advantages:

* Easy-to-use and well-documented
* High integration capacities
* High performance
* Suitable for real-time applications
* Wide functionality
* Support for web, Android, and iOS
* Simple control dashboard
* Can replace any piece of backend software
* High scalability

Firebase Disadvantages:

* Limited free plan
* Difficult execution of complex queries for data
* Learning curve requires time
* Different architectures and approaches of tools
* Lack of optimized search and query functions

**4.2.4. ANDROID STUDIO:** Android is one of the most popular mobile device platforms. The Android platform allows developers to write managed code using Java (http://www.developer.com/java) to manage and control the Android device. Android Studio is a popular IDE developed by Google for developing applications that are targeted at the Android platform. Note that Android Studio has replaced Eclipse as the IDE of choice for developing Android applications.



Figure 12: Android Studio

The following features are provided in the current stable version:

* Gradle-based build support
* Android-specific refactoring and quick fixes
* Lint tools to catch performance, usability, version compatibility and other problems
* ProGuard integration and app-signing capabilities
* Template-based wizards to create common Android designs and components
* A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
* Support for building Android Wear apps
* Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
* Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Android Studio supports all the same programming languages of IntelliJ (and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ that Android Studio is built on supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.