**C H A P T E R 1**

**Introduction**

* 1. **Overview**:

The car parking project is present in most places, but It can be difficult to prevent other vehicles from parking in your space in cities due to the scarcity of parking spaces. Parking spaces for commercial buildings, properties, and apartments without garages are open to other vehicles without parking permits, allowing them to sneak in and take your spot.

Getting parking protection is the best way to preserve and protect your parking space. This will help avoid stressful and frustrating situations. Hiring a parking guard is an excellent way to secure reserved and private parking spaces.

It has become increasingly popular in protecting public and private parking spaces due to its simplicity, safety and quality. This is one of the primitive methods in securing private parking spaces. So, we needed to devise smart ways to save the parking space of its owner

**1.2 Statement of the problem**

The usual parking process has a lot of problems that need to be solved, including:

Lost time looking for a place to line up, knowing available slots help

Reduce time and effort when looking around the parking area.

Traffic congestion and congestion as a result of the increase in the number of cars, which directs every driver to it.

A specific slot helps reduce traffic as much as possible.

Environment efficiency is the ability to accurately direct the driver into an available space.

Many environmental benefits, it reduces carbon dioxide emissions, noise and other pollutants.

**CH.1**

**1.3 Purpose:**

In this project, the idea of smart parking will be designed to save each parking for its owner, and we want to make sure that this idea will succeed on the ground or not? And will it make a difference in the facilities so that it is circulated to the owners of the facilities?

So that there is an iron cylinder connected to the servo motor and it is high off the ground so that it saves the position of the parking.

owner's car and when the owner enters, he scans the RFID card or enters the password, the parking lamp of your car will open and the servo motor will automatically start to return the cylinder under the ground to the same parking lot.

Therefore, the driver can stop the car.

When the driver wants to get out of the parking lot, when he is 25 seconds away from the parking lot, the servo motor will lift the roller up, and no one can line up.

**1.4 Literature review:**

The idea of a smart parking system was proposed in too many previous projects, Here we will present some projects:

Lanxin Wei, Qisheng Wu, Mei Yang, Wei Ding, Bo Li, Rong Gao designed a smart parking management system based on the RFID (radio frequency identification) and internet. when users arrive at the target parking lot, they can query the location of them parking spaces by positioning terminals installed inside of the parking lot [1].

Thanh Nam Pham, Ming-Fong Tsai, Duc Binh Nguyen, Chyi-Ren Dow, and Der-Jiunn Deng introduced an algorithm that increases the efficiency of the current cloud-based smart-parking system and developed a network architecture based on the internet-ofthings technology. a system that helps users automatically find a free parking space at the least cost based on new performance metrics to calculate the user parking cost by considering the distance and the total number of free places in each car park [2].

**CH.1**

Abhirup Khanna, Rishi Anand presented an IOT-based cloud-integrated smart parking system. the proposed smart parking system consists of an on-site deployment of an IOT module that is used to monitor and signalize the state of availability of each single parking space. a mobile application is also provided that allows an end-user to check the availability of parking space and book a parking slot accordingly [3].

-Mehala Chandran, Nur Fadila Mahrom, Thennarasan Sabapathy, Muzammil Jusoh, Mohd Nasrun Osman, Mohd Najib Yasin, N.A.M Hambali, R.Jamaluddin, N.Ali, Yasmin Abdul Wahab proposed a design of smart parking system where it helps the users to reserve parking slots using an Android application[4].

**1.5 Thesis outlines:**

This graduation project file is divided into five chapters:

Chapter one: an introduction to our project.

Chapter two: The theory on which the project is based.

Chapter three: provides an overview of all components used in the project.

Chapter four: software implementation.

Chapter five: The conclusion and recommend of the project.

**C H A P T E R 2**

**Flow chart and project cost**

**2.1 Project work cost:**

We calculated the cost of the project as a proposal based on a realistic experience, and the results were as follows:

|  |  |  |
| --- | --- | --- |
| No. | Item | cost (dollar) |
| 1 | 4\*Servo motor | 28 $ |
| 2 | 9\*IR sensor | 45 $ |
| 3 | Arduino nano | 16 $ |
| 4 | Power supply | 12 $ |
| 5 | RFID | 30 $ |
| 6 | Circuit shield | 10 $ |
| 7 | Power regulator | 5 $ |
| 8 | Screw & nuts | 25 $ |
| 9 | 6\*Led | 6 $ |
| 10 | Buzzer | 1 $ |
|  | Miscellaneous | 12 $ |
| Total | | **191 $** |

Table 2.1: Project work cost.

**CH.2**

**2.2 Flow Chart for program:**

Here we are going to describe our model using the flow charts:

Input

RFID

No

Servo motor

IR sensor

No

Yes

Output

Figure 2.1: Flow Chart for program

**C H A P T E R 3**

**Hardware**

**3.1 Introduction:**

This chapter will review the overall project scheme with all electrical components and circuits that have been used in the project including the brain "microcontroller”, the input side of the micro the controller which is the measurement components and the output side.

**3.2 Microcontroller:**

A microcontroller is a compact integrated circuit designed to govern a specific operation in an [embedded system](https://internetofthingsagenda.techtarget.com/definition/embedded-system)[5]. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip. Sometimes referred to as an embedded controller or microcontroller unit (MCU), microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances, among other devices. They are essentially simple miniature personal computers (PCs) designed to control small features of a larger component, without a complex front-end operating system (OS).

A microcontroller is embedded inside of a system to control a singular function in a device. It does this by interpreting data it receives from its I/O peripherals using its central processor. The temporary information that the microcontroller receives is stored in its data memory, where the processor accesses it and uses instructions stored in its program memory to decipher and apply the incoming data. It then uses its I/O peripherals to communicate and enact the appropriate action.

Microcontrollers are used in a wide array of systems and devices. Devices often utilize multiple microcontrollers that work together within the device to handle their respective tasks.

**CH.3**

The basic internal designs of microcontrollers are pretty similar. Figure 1.4 shows the block diagram of a typical microcontroller. All components are connected via an internal bus and are all integrated on one chip. The modules are connected to the outside world via I/O pins.

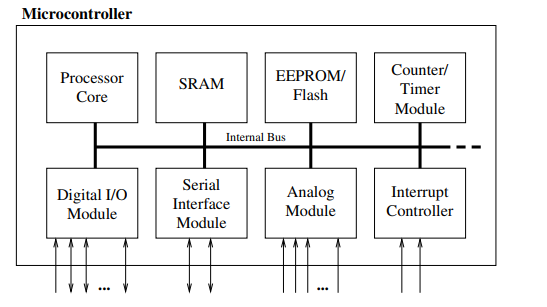


Figure 3.1: Basic layout of a microcontroller

**3.2.1 Arduino Nano board:**

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328P (Arduino Nano 3.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one.

It contains everything needed to support the microcontroller; simply connect it to a computer with a Mini-B USB cable to get started. It has a form factor that enables it to be easily placed on a breadboard

**CH.3**

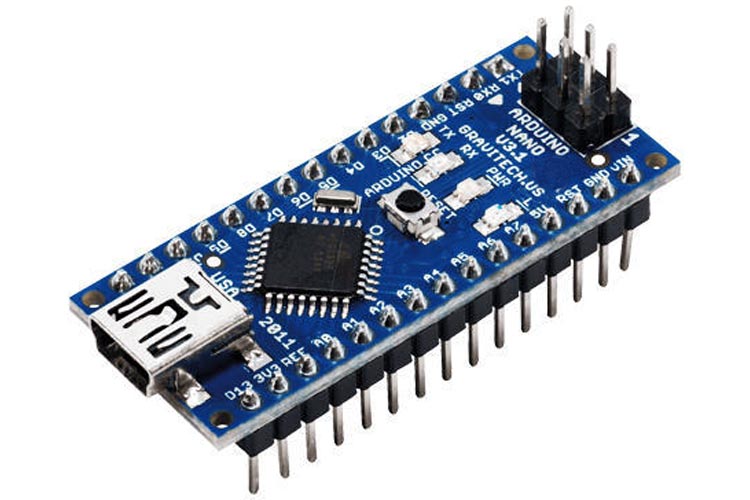


Figure 3.2: Arduino Nano board structure

So, we used “Arduino Nano” because it has a high number of Input and output pins which more flexibility, The structure of the pins is shown in Figure 3.2

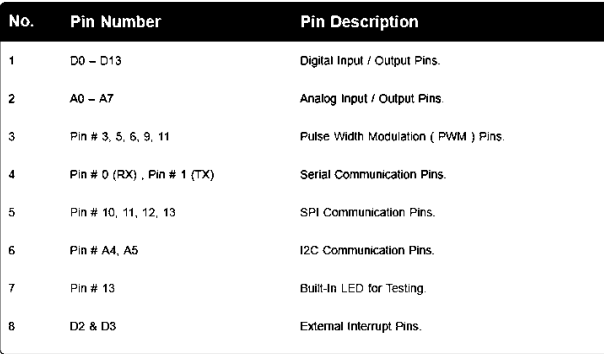


Figure3.3: Arduino Nano pin description.

**CH.3**

**3.2.2 Basics of Arduino Nano:**

Arduino Nano is a microcontroller board designed by Arduino.cc.

The microcontroller used in the Arduino Nano is Atmega328, the same one as used in Arduino UNO. It has a wide range of applications and is a major microcontroller board because of its small size and flexibility. So, now let's have a look at its basic features.

**Basic Features of Arduino Nano:**

Here are few of its basic features which you must know if you are thinking to work on this great microcontroller board:

• It has 22 input/output pins in total.

• 14 of these pins are digital pins.

• Arduino Nano has 8 analogue pins.

• It has 6 PWM pins among the digital pins.

• It has a crystal oscillator of 16MHz.

• Its operating voltage varies from 5V to 12V.

• It also has a mini USB Pin which is used to upload code.

• It also has a Reset button on it

**CH.3**

**3.3 Drive circuit:**

We have used in this project a servo motor because it works within a closed system in which information and data are used in the form of signals coming from the machine to be moved to control movement in both movement, location and direction. The servo starts working when a signal (either digital or analogue) reaches it from the system by moving with a certain speed in a certain direction, then it receives the signals returning to it from the machine to control the speed and location, depending on that it is associated with an encoder to provide feedback signals, and the more signals, the more servo Accuracy and efficiency.



Figure 3.4: Servo Motor

For example, the position of the servo motor(sg90) depends on the oscillation length. Where it receives a vibration every approximately 20 parts of a second. If the vibration is in one part of a second, then the servo angle is zero, and if it is 1.5

**CH.3**

parts of a second, it will be in the middle, and if it is two parts of a second, it will be at an angle of 180 degrees.

Key features of servo motor:

• Very small micro size

• Can lift 3.75lb positioned 1cm from center of shaft

• 180-degree rotation

• Analog drive

• Low cost

The Servo Driver circuit allows users to experiment with servos without the need for a computer or a microcontroller device to program the travel limits. The circuits use 556 timers to generate the control pulses and two potentiometers to adjust the widths of the pulse.

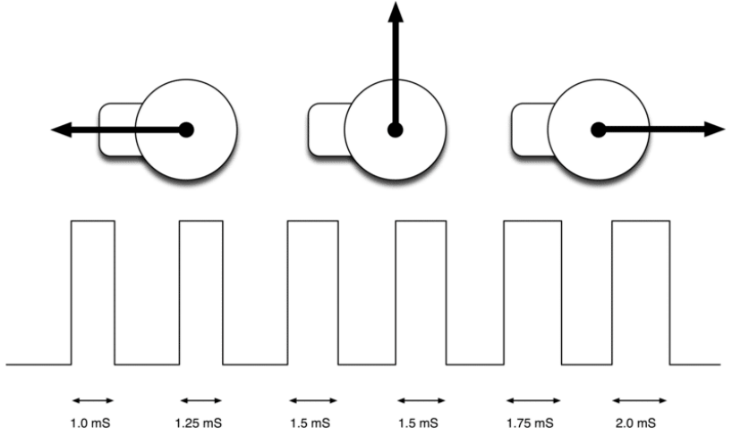


Figure 3.4.1: Servo Motor rotation.

**CH.3**

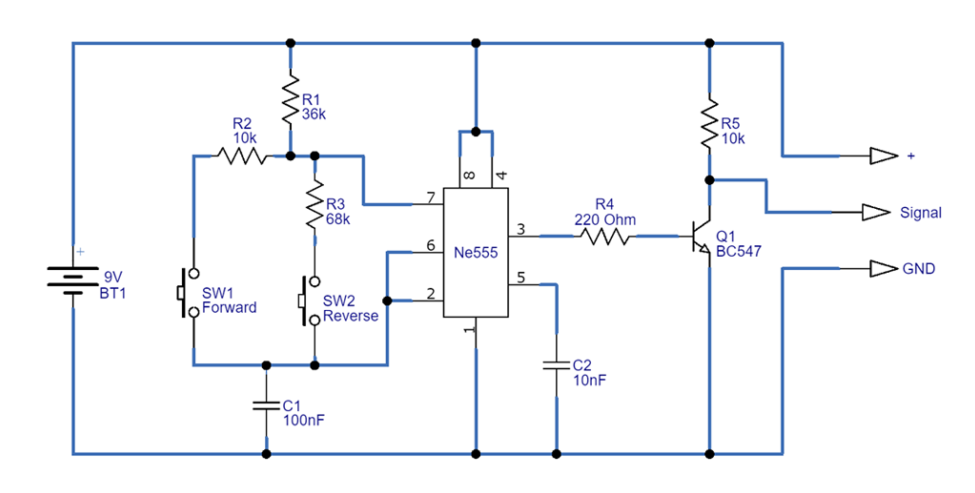


Figure 3.5: servo motor drive circuit

Table 3.1 shows each motor’s specifications from the datasheet.

|  |  |
| --- | --- |
| Rated voltage | 4.8 ~ 5v |
| Torque | 2.5 (k.g-cm) |
| Operating speed | 0.1s/60° |
| Weight of motor | 9gm |
| Stall current | one amp |

Table 3.1: Motor specifications.

**CH.3**

**3.4 IR sensor:**

We will use the IR sensor for this project so that it gives a signal when the car arrives, and the motor remains constant on its movement until the car moves again, and the motor returns to its normal position.[6]

The IR sensor has a 3-pin connector that interfaces it to the outside world. The connections are shown in a figure 3.5:

### 

figure 3.6: IR sensor.

VCC  is the power supply pin for the IR sensor which we connect to the 5V pin on the Arduino.

OUT pin is a 5V TTL logic output. LOW indicates no motion is detected; HIGH means motion is detected.

GND Should be connected to the ground of the Arduino.

The **working of the IR sensor module** is very simple, it consists of two main components: the first is the IR transmitter section and the second is the IR receiver section(photo transistor ). In the transmitter section, **IR led** is used and in the receiver section, a **photodiode** is used to receive infrared signal and after some signal processing and conditioning, you will get the output.

**CH.3**

An IR proximity sensor works by applying a voltage to the onboard **Infrared Light Emitting Diode** which in turn emits infrared light. This light propagates through the air and hits an object, after that the light gets reflected in the photodiode sensor. If the object is close, the reflected light will be stronger, if the object is far away, the reflected light will be weaker. If you look closely toward the module. When the sensor becomes active it sends a corresponding **Low signal** through the output pin

that can be sensed by an Arduino or any kind of microcontroller to execute a particular task. The one cool thing about this module is that it has two onboard

LEDs built-in, one of which lights on when power is available and another one turns on when the circuit gets triggered.

**3.5 Radio-frequency identification (RFID):**

An [RFID](https://en.wikipedia.org/wiki/Radio-frequency_identification) or radio frequency identification system consists of two main components, a tag attached to the object to be identified, and a reader that reads the tag. A reader consists of a radio frequency module and an antenna that generates a high frequency electromagnetic field. Whereas the tag is usually a passive device (it does not have a battery). It consists of a microchip that stores and processes information, and an antenna for receiving and transmitting a signal.[7].

**CH.3**

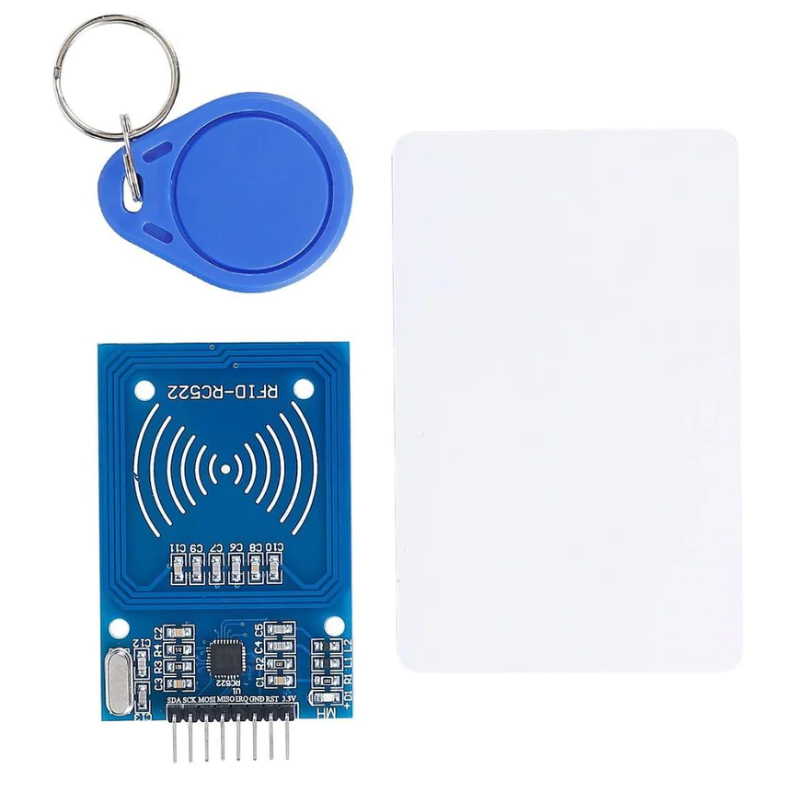


Figure 3.7: Radio-frequency identification

When the tag is brought close to the reader, the reader generates an electromagnetic field. This causes electrons to move through the tag’s antenna and subsequently powers the chip.

The chip then responds by sending its stored information back to the reader in the form of another radio signal. This is called a backscatter. The reader detects and interprets this backscatter and sends the data to a computer or microcontroller.

### The RFID connections are shown in a figure 3.7:

**CH.3**

### 

figure 3.8: The RFIDconnections

In our project, we use RFID based digital parking management system ISO 16963 RFID standard for item management - unique identifier of RF tag.

There are two main international RFID standards organizations or bodies that govern RFID:

ISO - International Standards Organization ·

EPC global - Electronics Product Code Global Incorporated

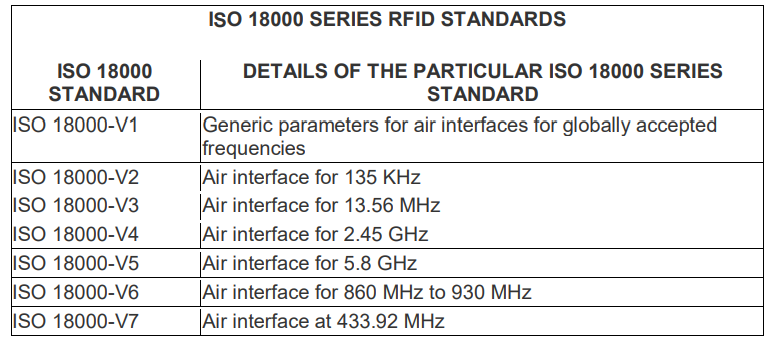
****

Table 3.2: ISO 18000 series RFID standards

**CH.3**

**3.6 The buzzer:**

An active buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.



Figure 3.9: The Buzzer.

**3.7 power supply:**

We have used a power supply to supply 12 volts to the Arduino and a regulator to supply 5 volts to the sensors and motors**.**

**CH.3**

**3.7 Light-emitting diode (LED):**

LED is a semiconductor device that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor.[5] White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device.



Figure 3.10: light-emitting diode (LED)

**3.8 Wires:**

Female/Female Jumper Wires and Male/Male Jumper Wires are used in this project.

**CH.3**

****

Figure 3.11: Male to male wires

****

Figure 3.12: Female to female wires

**3.9 Hardware design**

Hardware design involves the design and development of computer hardware components, including circuit boards, microchips, schematics, and scanners, as well as the creation of the physical parts of a product

**CH.3**

**3.9.1 Virtual circuit schematic for the project:**

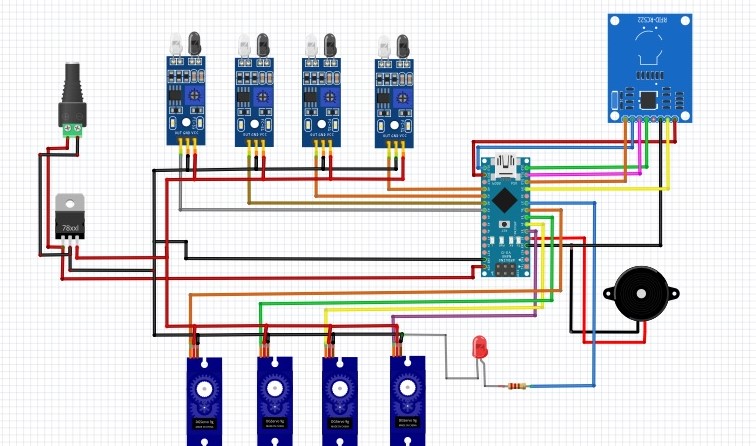
****

Figure 3.13: Schematic circuit.

**3.9.2 The Pc board:**

A printed circuit board, or PC board, or PCB, is a non-conductive material with conductive lines printed or etched. Electronic components are mounted on the board and the traces connect the components together to form a working circuit or assembly.

**CH.3**

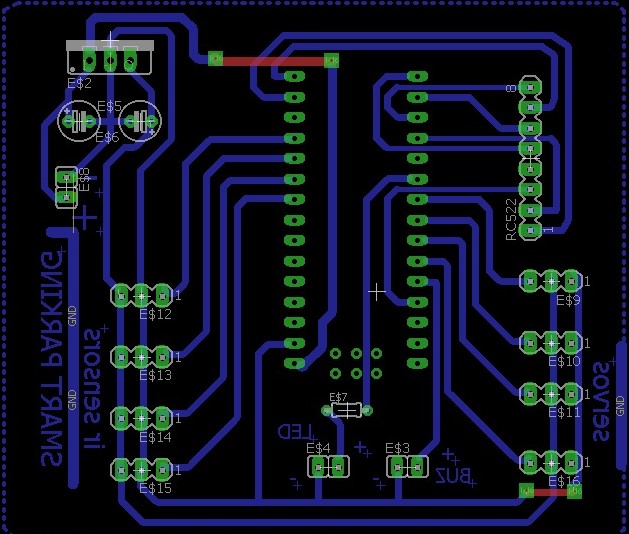
****

Figure 3.14: Pc board

**3.9.3 Practical circuit diagram of the project:**

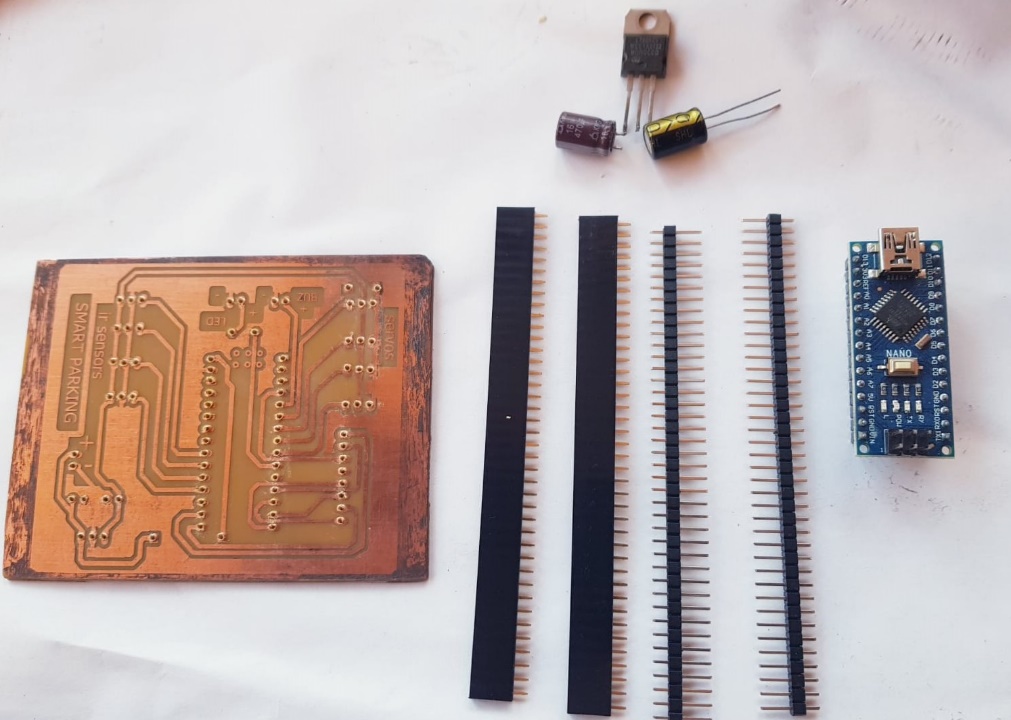
****

Figure3.15: Practical circuit diagram

**C H A P T E R 4**

**Software**

**4.1 Introduction:**

To find out that the parking owner has arrived, he scans his RFID card, so we will program each parking card on the RFID card.

Then it will give a signal to the servo motor to make its turn lower the pillar so that the owner can line up his car.

And when the owner leaves the parking lot designated for him, the IR sensor senses the movement of the car, so we will program it to return the pillar to its place within 5 seconds.

We linked all this explanation in a practical way on an electronic board that includes the micro-Arduino, the RFID, the motor, the IR sensor, the LED, and the buzzer.

**4.2 Arduino Nano:**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. The hand gesture analyzing algorithm is written.

in this platform which is then uploaded to the microcontroller. I used ARDUINO 1.8.19 [8].

****

Figure 4.1: Arduino IDE

**CH.4**

**4.2.1 Software procedure**

Installing Arduino 1.8.19:

• First of all, we install Arduino the newest one on my laptop

• Second, we set the code of the transmitter and the receiver code according to the components.

• In the beginning I check every piece with single code .to see whether its work properly or not. for example, I checked the dc motor with suitable code alone to see if it works well

. • I checked also mpu6050. HC-05, HC-06, H-bridge and other items alone.

• I Combines all the codes samples and created the whole code and experiment it, till I finished the whole project.

**Arduino code:**

**{**

#include <SPI.h>

#include <MFRC522.h>

#include <Servo.h>

#define SS\_PIN 10

#define RST\_PIN 9

#define LED\_PIN 7

#define BUZZER\_PIN 2

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

Servo servo1;

Servo servo2;

Servo servo3;

Servo servo4;

**CH.4**

const int sensorPin1 = A0;

const int sensorPin2 = A1;

const int sensorPin3 = A2;

const int sensorPin4 = A3;

int sensorValue1 = 0;

int sensorValue2 = 0;

int sensorValue3 = 0;

int sensorValue4 = 0;

enum GateState {

CLOSED,

OPEN,

CAR\_ENTERED,

CAR\_EXITED

};

struct Gate {

Servo servo;

int sensorPin;

GateState state;

};

Gate gate1 = {servo1, sensorPin1, CLOSED};

Gate gate2 = {servo2, sensorPin2, CLOSED};

Gate gate3 = {servo3, sensorPin3, CLOSED};

Gate gate4 = {servo4, sensorPin4, CLOSED};

void setup() {

Serial.begin(9600);

SPI.begin();

mfrc522.PCD\_Init();

servo1.attach(3);

**CH.4**

servo2.attach(4);

servo3.attach(5);

servo4.attach(6);

pinMode(LED\_PIN, OUTPUT);

pinMode(BUZZER\_PIN, OUTPUT);

noTone(BUZZER\_PIN);

closeGate(gate1);

closeGate(gate2);

closeGate(gate3);

closeGate(gate4);

}

void loop() {

if (mfrc522.PICC\_IsNewCardPresent() && mfrc522.PICC\_ReadCardSerial()) {

byte uid[4];

for (byte i = 0; i < mfrc522.uid.size; i++) {

uid[i] = mfrc522.uid.uidByte[i];

}

Serial.print("RFID card detected with UID: ");

for (byte i = 0; i < mfrc522.uid.size; i++) {

Serial.print(uid[i] < 0x10 ? "0" : "");

Serial.print(uid[i], HEX);

}

Serial.println();

if (checkCardUID(uid, 0x93, 0xA0, 0x86, 0xA9)) {

processCard(gate1);

} else if (checkCardUID(uid, 0x93, 0x50, 0x6D, 0xA9)) {

processCard(gate2);

} else if (checkCardUID(uid, 0x33, 0x39, 0x6D, 0xA9)) {

**CH.4**

processCard(gate3);

} else if (checkCardUID(uid, 0xA3, 0x31, 0x69, 0xA9)) {

processCard(gate4);

}

}

checkGate(gate1);

checkGate(gate2);

checkGate(gate3);

checkGate(gate4);

}

bool checkCardUID(byte\* uid, byte b0, byte b1, byte b2, byte b3) {

return (uid[0] == b0 && uid[1] == b1 && uid[2] == b2 && uid[3] == b3);

}

void processCard(Gate& gate) {

Serial.println("Card detected!");

digitalWrite(LED\_PIN, HIGH);

tone(BUZZER\_PIN, 500);

delay(300);

noTone(BUZZER\_PIN);

delay(300);

digitalWrite(LED\_PIN, LOW);

if (gate.state == CLOSED) {

openGate(gate);

}

}

void checkGate(Gate& gate) {

if (gate.state == OPEN || gate.state == CAR\_ENTERED) {

**CH.4**

if (gate.state == OPEN) {

int sensorValue = analogRead(gate.sensorPin);

if (sensorValue < 500) {

Serial.println("Car detected!");

gate.state = CAR\_ENTERED;

}

} else if (gate.state == CAR\_ENTERED) {

int sensorValue = analogRead(gate.sensorPin);

if (sensorValue > 500)

delay(5000);

{ Serial.println("Car moved!");

gate.state = CAR\_EXITED;

}

}

}

if (gate.state == CAR\_EXITED) {

closeGate(gate);

}

}

void openGate(Gate& gate) {

gate.servo.write(130);

gate.state = OPEN;

}

void closeGate(Gate& gate) {

gate.servo.write(30);

gate.state = CLOSED;

}

**}**

**CH.4**

****

Figure4.2 : The Arduino code

**CH.4**

****

Figure4.2.1 : The Arduino code

**CH.4**

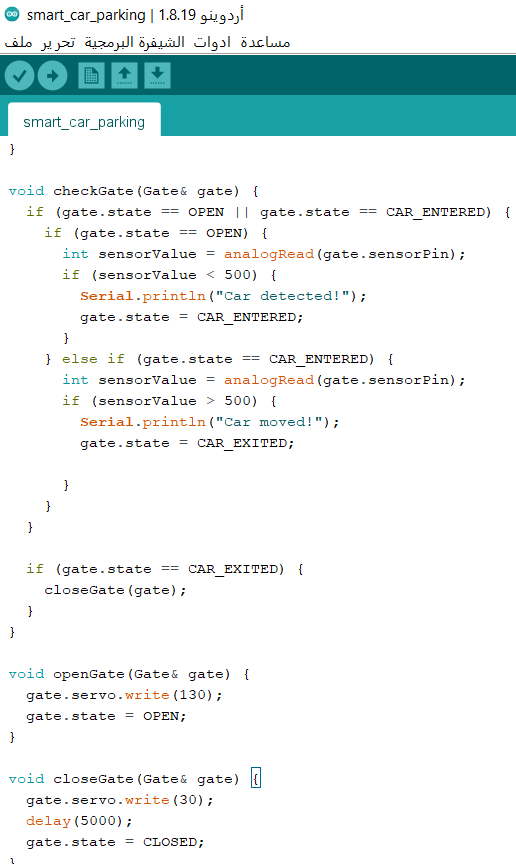
****

Figure4.2.2 : The Arduino code

**CH.4**

**4.3 IR sensor:**

The IR sensor is one of the most common sensors in the Arduino area, due to the technology that can be deployed with it.

Depending on infrared technology, sending messages between controllers or detecting objects, as well as detecting other fires.

This sensor depends in its work on sending invisible red radiation and receiving it closely.

The transmitter and receiver are on the sensor which makes it very easy to set up.  
Here we will write programming code in general for IR sensor :

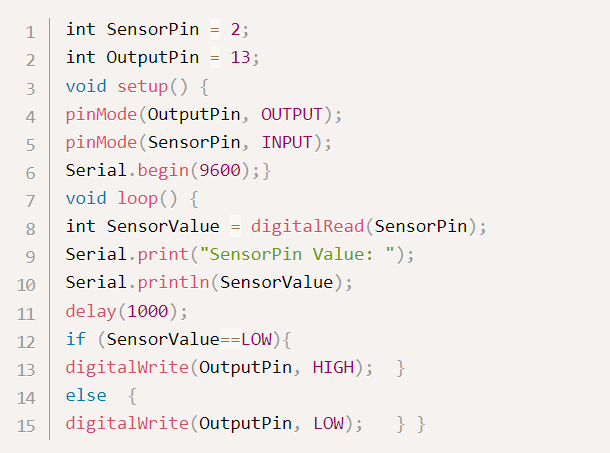


Figure 4.3: general programming code for IR sensor

**CH.4**

**4.4 Radio-frequency identification (RFID):**

Every RFID system consists of three components: a scanning antenna, a transceiver and a transponder. When the scanning antenna and transceiver are combined, they are referred to as an RFID reader or interrogator. There are two types of RFID readers -- fixed readers and mobile readers. The RFID reader is a network-connected device that can be portable or permanently attached.

It uses radio waves to transmit signals that activate the tag. Once activated, the tag sends a wave back to the antenna, where it is translated into data.

The transponder is in the RFID tag itself. The read range for RFID tags varies based on factors including the type of tag, type of reader, RFID frequency and interference in the surrounding environment or from other RFID tags and readers. Tags that have a stronger power source also have a longer read range.

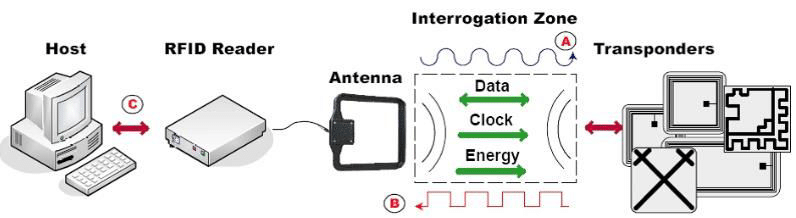


Figure 4.4: Principle of an RFID system operation

**C H A P T E R 5**

**Conclusions and Recommendation**

**5.1 Conclusions and recommendation:**

In conclusion, this project completed the design of the smart parking system depending on tags or RFID sensors to locate reserved parking spaces.

In addition to the manager's mobile application that displays and saves user data including names and balances.

According to the previous reasons the project is considered economically efficient (feasible), so we recommend to invest in this smart parking project.

**References:**

[1] Lanxin Wei, Qisheng Wu, Mei Yang, Wei Ding, Bo Li, & Rong Gao. (2012). Design and Implementation of Smart Parking Management System based on RFID and Internet. School of Electronic and Control Engineering Chang’an University, China.

[2] Thanh Nam Pham, Ming-Fong Tsai, Duc Binh Nguyen, Chyi-Ren Dow, & Der-Jiunn Deng. (2015). A Cloud-Based Smart-Parking System Based on Internet-of-Things Technologies. Department of Information Engineering and Computer Science, Feng Chia University, Taichung 407, Taiwan, Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua 500, Taiwan.

[3] Abhirup Khanna, & Rishi Anand. (2016). IoT based Smart Parking System. University of Petroleum and Energy Studies (UPES) Dehradun, Uttarakhand, India.

[4] Mehala Chandran, Nur Fadila Mahrom, Thennarasan Sabapathy, Muzammil Jusoh, Mohd Nasrun Osman, Mohd Najib Yasin, N.A.M Hambali, R.Jamaluddin, N.Ali, & Yasmin Abdul Wahab. (2019). An IoT Based Smart Parking System. Biolectromagnetic Research Group, School of Computer and Communication Engineering, Universiti Malaysia Perlis, Biolectromagnetic Research Group, School of Microelectronic Engineering, Universiti Malaysia Perlis, Nanotechnology & Catalysis Research Centre, Deputy Vice Chancellor (Research and Innovation Office), Universiti Malaya.

.

[5] The information was extracted from Microcontroller text book

<http://paginas.fisica.uson.mx/horacio.munguia/aula_virtual/Recursos/Microcontroller_02.pdf>

[6] IR sensor Retrieved from <https://makerlab-electronics.com/products/ir-proximity-sensor?_pos=1&_psq=Ir+sensor&_ss=e&_v=1.0>

## [7] RFID URL Regulated Product(s), Radiation-Emitting Products,2018 :

<https://www.fda.gov/radiation-emitting-products/electromagnetic-compatibility-emc/radio-frequency-identification-rfid>

[8] Download the Arduino IDE. Retrieved from. <https://www.arduino.cc/en/Main/Software>