**Pushpam motors application with scanning card system**

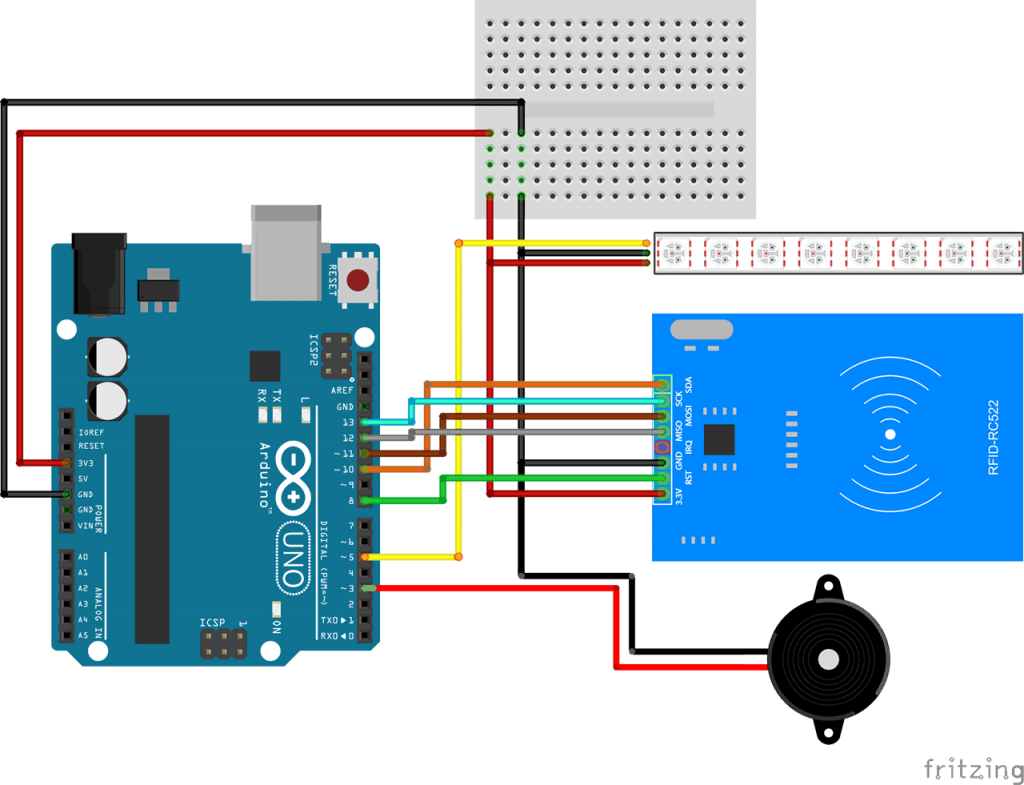
**Electronics Needs :-**

* **Ardiuno UNO R3 circuit**
* **Connector pins**
* **Buzzer**
* **Led**
* **Jumper wires**
* **Rfid reader RC522**
* **Rfid cards**
* **Ardiuno cable**
* **Soldering stuff**

**Notes :-**

**Long pin is always positive in leds and buzzers**

**Diagrams :-**

****

**No need of breadboard or white board**

**Integration with ionic Android App**

**Install serial plugin**

$ ionic cordova plugin add cordovarduino

$ npm install --save @ionic-native/serial

**After that,**

**Platforms -> android -> src -> main ->Androidmenifest.xml**

**After </application> tag**

**<uses-sdk android:minSdkVersion="19" android:targetSdkVersion="27" />**

**<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />**

**<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />**

**Goto plugins->cordovarduino->plugin.xml**

**If not available**

**<platform name="android">**

**<config-file target="res/xml/config.xml" parent="/\*">**

**<feature name="Serial">**

**<param name="android-package" value="fr.drangies.cordova.serial.Serial"/>**

**</feature>**

**</config-file>**

**<source-file src="src/android/fr/drangies/cordova/serial/Serial.java" target-dir="src/fr/drangies/cordova/serial" />**

**<source-file src="src/android/fr/drangies/cordova/serial/UsbBroadcastReceiver.java" target-dir="src/fr/drangies/cordova/serial" />**

**<lib-file src="lib/usbseriallibrary.jar" arch="device" /></platform>**

**Code :-**

**cardChecker()**

**{**

**// this.basic\_op.getStorage('rfidno').then(val=>{**

**// if(val != undefined)**

**// {**

**//{vid:'2341',pid:'0043',driver:'FtdiSerialDriver'}Arduino specification if required in requestPermission**

**this.rfid = "";**

**this.serial.requestPermission().then(()=>{**

**this.serial.open({baudRate: 9600,**

**dataBits: 8,**

**stopBits: 1,**

**parity: 0,**

**dtr: true,**

**rts: true,**

**sleepOnPause: false}).then(()=>{**

**this.device\_attach = true;**

**this.serial.registerReadCallback().subscribe(res=>{**

**var r = new Uint8Array(res);**

**this.rfid = this.rfid.concat(String.fromCharCode.apply(null,r));**

**if(this.rfid.length == 8)**

**{**

**this.rfid = this.rfid.substring(0,8);**

**// this.serial.close();**

**this.login(this.rfid);**

**}**

**},error=>{**

**this.basic\_op.alertboxDismiss("rfid card not read");**

**});**

**},error=>{**

**this.basic\_op.alertboxDismiss("connection problem issue");**

**});**

**},error=>{**

**let alert = this.alertCtrl.create({**

**title: 'Machine is not connected',**

**message : 'Machine is not connected.You have to connect the machine then dismiss.',**

**buttons: [**

**{**

**text: 'Dismiss',**

**handler: () => {**

**if(this.device\_attach){**

**this.basic\_op.alertboxDismiss("Machine is connected.");**

**}**

**else{**

**this.cardChecker();**

**}**

**}**

**}**

**],**

**enableBackdropDismiss : false**

**});**

**alert.present();**

**});**

**// }**

**// else{**

**// }**

**// },err=>{**

**// });**

**}**

**Ardiuno IDE**

1. **Download ardiuno ide**
2. **Connect Arduino uno to pc**
3. **Open ide**
4. **Goto sketch menu**
5. **Choose port and circuit**
6. **Write code**
7. **And finally upload in Arduino by clicking upload button.**

**Code :- scanning rfid with buzzer and led**

**#include "SPI.h"**

**#include "MFRC522.h"**

**#define SS\_PIN 10**

**#define RST\_PIN 9**

**#define SP\_PIN 8**

**#define Buzzer 3 //position pin of buzzer**

**#define Led 5 //positive pin of led**

**MFRC522 rfid(SS\_PIN, RST\_PIN);**

**MFRC522::MIFARE\_Key key;**

**void setup() {**

**pinMode(Buzzer, OUTPUT); // Set buzzer pin to an Output pin**

**pinMode(Led, OUTPUT);**

**Serial.begin(9600);**

**SPI.begin();**

**rfid.PCD\_Init();**

**}**

**void loop() {**

**char i = Serial.read();**

**if(i=='1')**

**{**

**}**

**else**

**{**

**if (!rfid.PICC\_IsNewCardPresent() || !rfid.PICC\_ReadCardSerial())**

**return;**

**// Serial.print(F("PICC type: "));**

**MFRC522::PICC\_Type piccType = rfid.PICC\_GetType(rfid.uid.sak);**

**// Serial.println(rfid.PICC\_GetTypeName(piccType));**

**// Check is the PICC of Classic MIFARE type**

**if (piccType != MFRC522::PICC\_TYPE\_MIFARE\_MINI &&**

**piccType != MFRC522::PICC\_TYPE\_MIFARE\_1K &&**

**piccType != MFRC522::PICC\_TYPE\_MIFARE\_4K) {**

**Serial.println(F("Your tag is not of type MIFARE Classic."));**

**return;**

**}**

**String strID = "";**

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

**strID +=**

**(rfid.uid.uidByte[i] < 0x10 ? "0" : "") +**

**String(rfid.uid.uidByte[i], HEX);**

**}**

**strID.toUpperCase();**

**rfid.PICC\_HaltA();**

**rfid.PCD\_StopCrypto1();**

**Serial.print(strID);**

**tone(Buzzer, 1000); // Send 1KHz sound signal...**

**digitalWrite(Led,HIGH);**

**delay(1000); // ...for 1 sec**

**noTone(Buzzer);**

**digitalWrite(Led,LOW);// Stop sound...**

**delay(1000);**

**}**

**}**