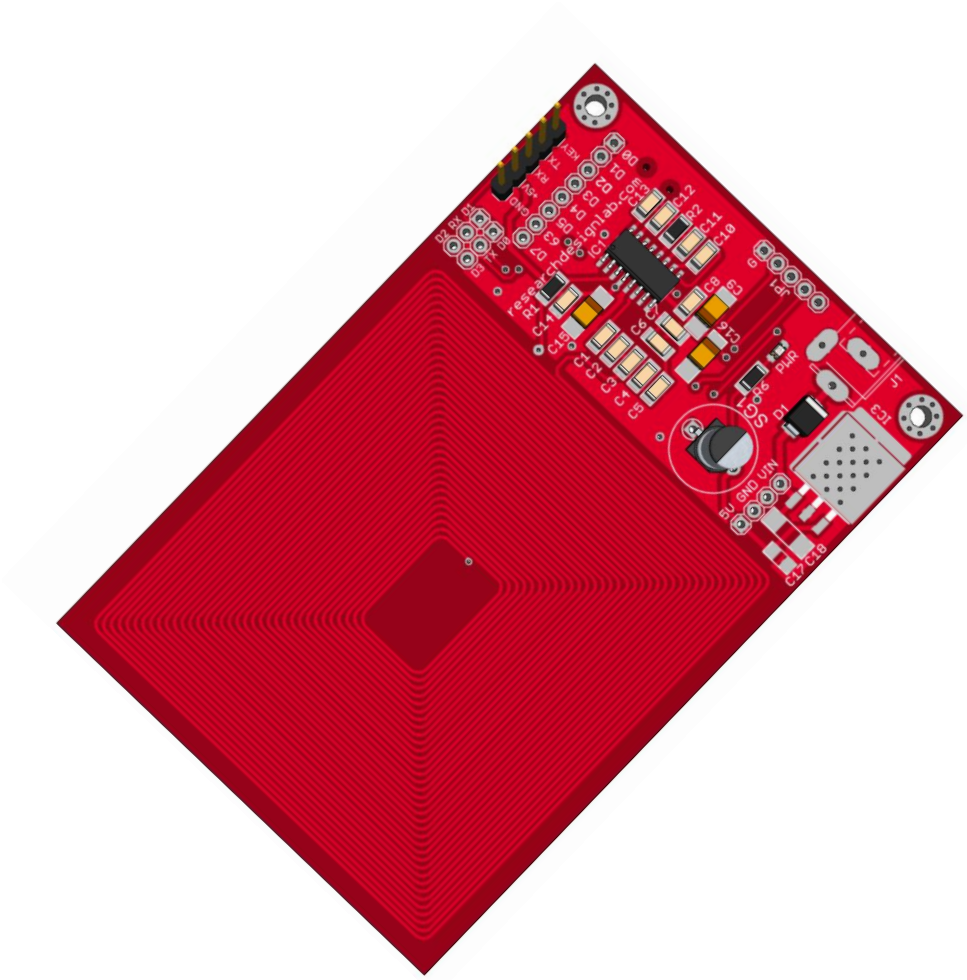
RFID Reader-Serial



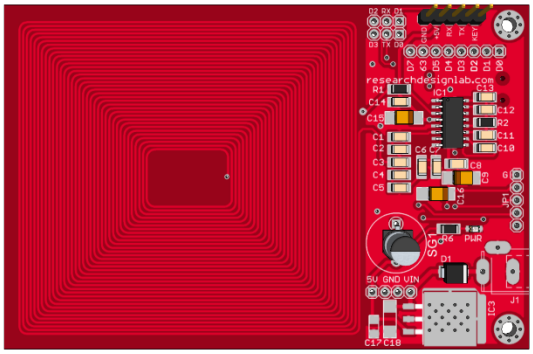
ORDER CODE: RDL616

Document Version: V1.0

Contents

1. INTRODUCTION

Low Frequency 125kHz RFID Reader with serial interface with range 0-10 CM. Board level RFID readers can easily integrated with custom build application / solution via serial communication. Interfacing Wi-Fi module with this serial RFID reader can connect IoT application software, for monitoring and identification.



2. FEATURES

- Low-cost method for reading passive RFID tags.
- Built in Antenna
- On-Board Power LED
- Current Requirement <120mA
- Serial Communication TTL at 9600 baud.
- Detecting Range up to 0-10cms.
- High quality PCB FR4 Grade with FPT Certified.

3. APPLICATIONS

- RFID based attendance system
- RFID based smart shopping system
- RFID based medical file tracking system
- RFID based inventory management system
- RFID based access control system -security
- RFID based library management system
- RFID based security guard monitoring system
- RFID based asset tracking system

- RFID based vehicle parking system
- RFID based toll gate collection system
- RFID hotel room access control system
- RFID based product identification for blind
- RFID based blind indoor navigation system
- RFID based dual authentication system for software application
- RFID based industry supply chain control system
- RFID based smart conveyor

4. SPECIFICATIONS

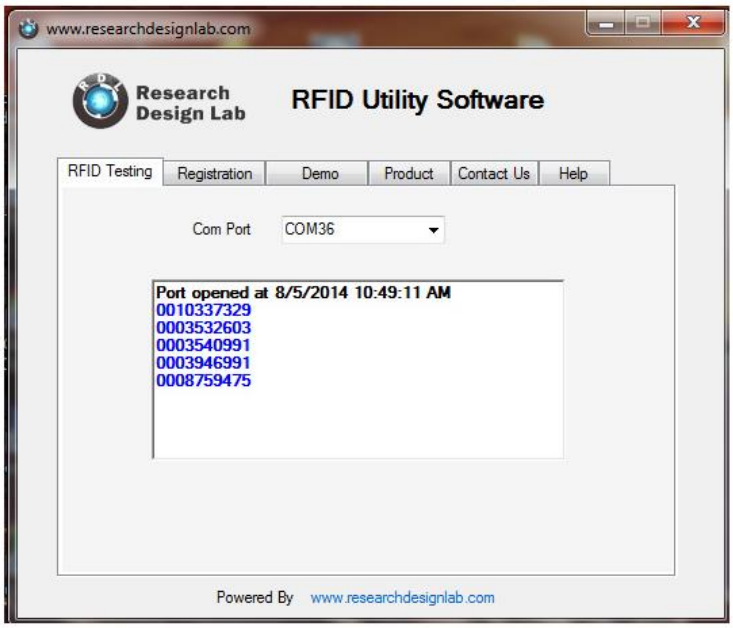
Specifications	
Frequency	125Khz
Max Current	<120mA
Scan time	<0.5sec
Communication serial UART	9600 bps
Detecting Range	Up to 0-10cm*
LED Indication	Power ON
Dimension(L * W)	104MM * 61MM

* Range depends on card/Tag Manufacturer

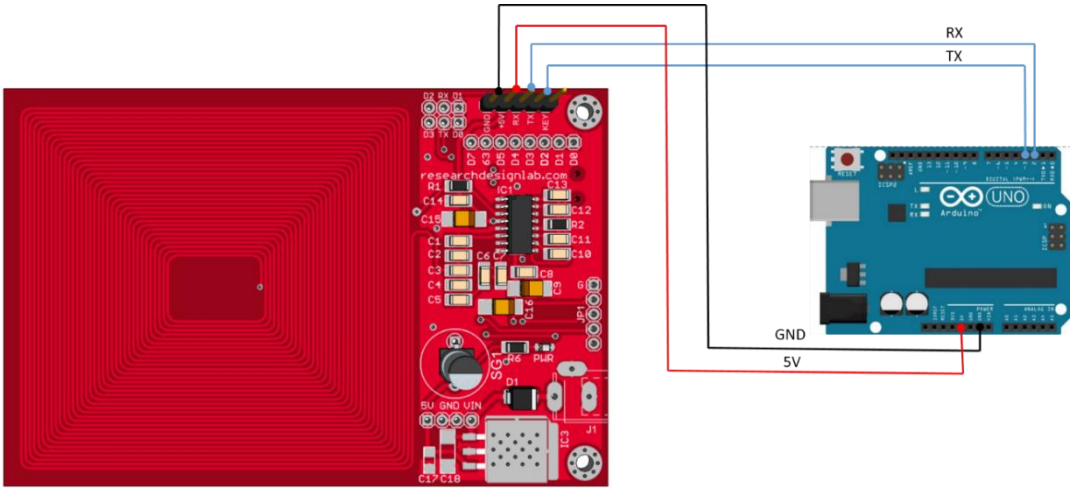
5. SOFTWARE LINK

Download RFID Utility Software

https://researchdesignlab.com/projects/RFID%20Utility%20V2.rar

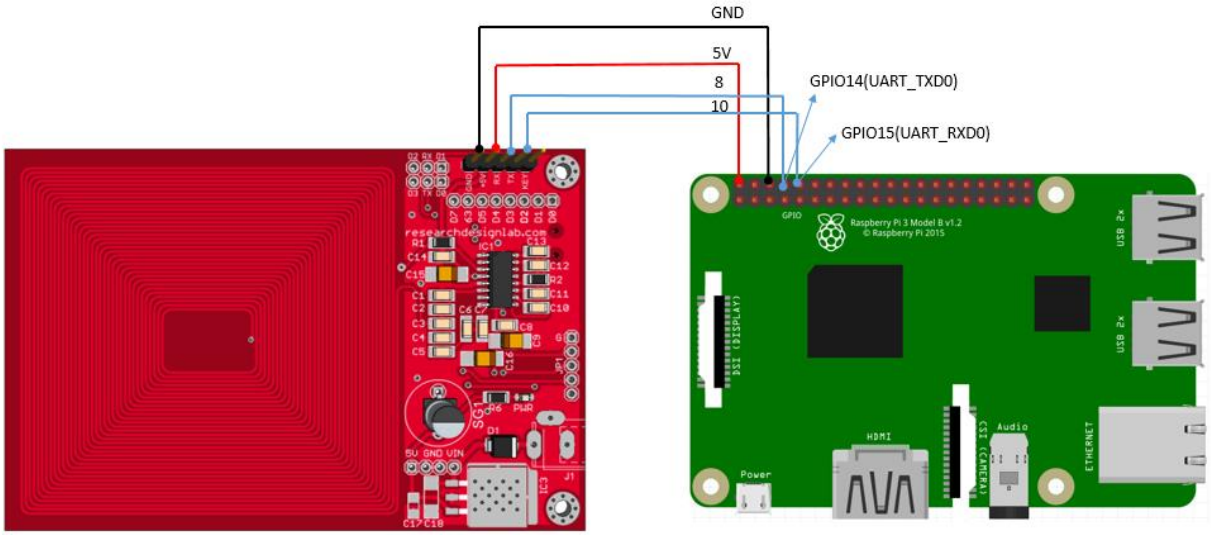


6. Arduino Connection Diagram




```
/*
 Receives from software serial, sends to hardware serial.
  The circuit:

 * RFID TX is digital pin 3 (connect to RX of other device)
 * 5v to 5v
 * Gnd to Gnd
 */
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3); // RX, TX
void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
  Serial.println("---RFID Reader---");

  // set the data rate for the SoftwareSerial port
  mySerial.begin(9600);
}
void loop() // run over and over
{
  if (mySerial.available())
    Serial.write(mySerial.read());
}
```

7. Raspberry Pi Connection Diagram

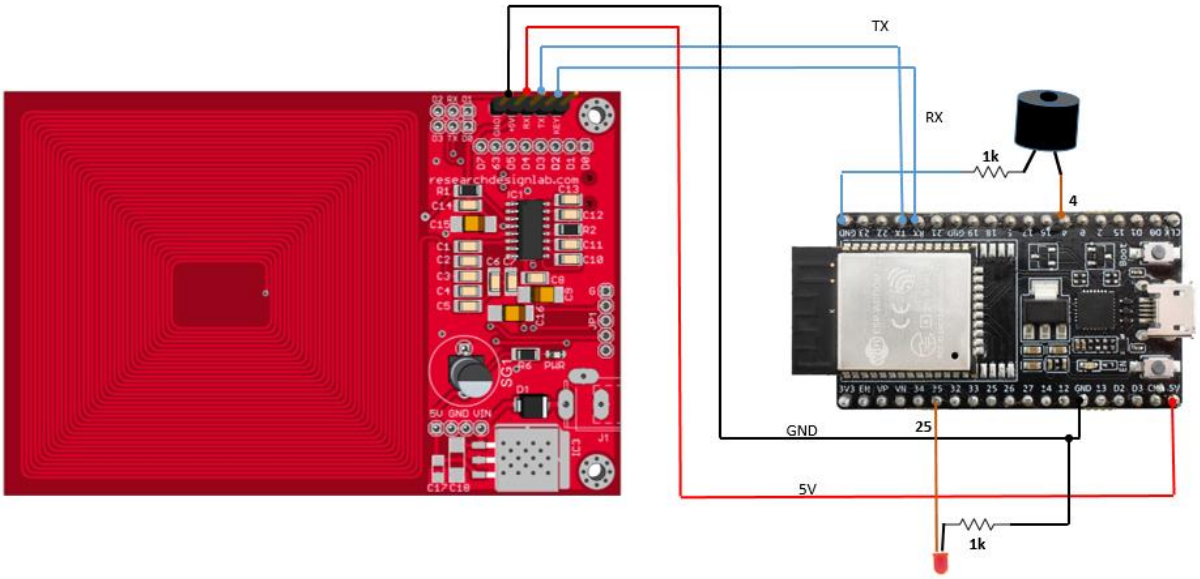



```
/*
* Project name:ft232 brekaout board testing / RFID 125khz
* Copyright
(c) Researchdesignlab.com
* Test configuration:
BOARD: RASPBERRY PI
*/
#include <stdio.h>
#include <stddef.h>
#include <termios.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <string.h>
unsigned char rx_buffer[10];
```

```
unsigned char test[64];
unsigned char number[64];
unsigned char number_check[64]="+918123902188";
int count,i,flag,k;
int main()
{
FILE *ofp_uart1_tx, *ofp_uart1_rx;
termios uart1;
int fd,f;
int m,n,d,e,a,b,c,q,r,x,y,z,t,u,o;
//IN CASE OF RASPBERRY PI THE FILE NAME IS /dev/AMA0
if((fd = open("/dev/ttyUSB1", O_RDWR | O_NOCTTY )) < 0)// ttyUSB*  changes as devices plugged in and out.
{
printf("Unable to open uart1 access.\n");
}
if(tcgetattr(fd, &uart1) < 0)
{
printf("Could not get attributes of UART1 at ttyO1\n");
}
if(cfsetospeed(&uart1, B9600) < 0)
{
printf("Could not set baud rate \n");
}
else
{
printf("Baud rate: 9600\n");
}
uart1.c_iflag = 0;
uart1.c_oflag = 0;
```

```
uart1.c_lflag = 0;
tcsetattr(fd, TCSANOW, &uart1);
tcflush(fd,TCIFLUSH);
while(1)
{
read(fd, rx_buffer, (1));
printf("%c\n",rx_buffer[0]);
}
}
```

8. ESP 32 Connection Diagram




```
#include <WiFi.h>
#include <Arduino.h>
#include <HTTPClient.h>
#include <EEPROM.h>
#include <Wire.h>
#include <WiFiClient.h>
#include <WiFiAP.h>
/*****************{\"USER\":\"" + RFID + "\",\"TOKEN\":\"" + TOKEN + "\",\"DEVICE-ID\":\"" + DEVICEID + "\"}**********************/
```

```
#ifndef PERSONAL
char SSID12[15];
char PASSWORD12[15];
char DEVICEID[20];
char TOKEN[15];
char config_par_key[4][60] = {"SSID:", "PWD:", "DEVICE ID:", "TOKEN"};
uint8_t config_par_addr[4] = {10, 30, 50, 70};
uint8_t data_uploaded_flag = 1;
char RX_BUFFER[15];
uint8_t rx_buf_index = 0;
char json_upload_data_temp[30];
#endif
uint8_t MEMORY_INIT = 1;
uint8_t BUZZ_INIT = 1;
const char* host = "rdlabc.000webhostapp.com";
char cnt = 0;
const char *ssid = "Enter your ssid";
const char *password = "Enter your password";
WiFiServer server(8080);
char Channel = 0;
char SMS[60];
uint16_t ADDRS_EP, i;
uint8_t arraySize;
String RFID = "";
String jason_string = "";
String url = "/uploaddata.php";

void setup()
{
  Serial.begin(9600);
  delay(100);
  pinMode(18, INPUT_PULLUP);
  // We start by connecting to a WiFi network
  pinMode(25, OUTPUT);
  pinMode(4, OUTPUT);
  digitalWrite(25, LOW);
  digitalWrite(4, LOW);
  EEPROM.begin(4096);
  delay(250);
//digitalRead(18)==LOW
```

```
if (digitalRead(18) == LOW)
 //if (MEMORY_INIT==1)
 {
  Serial.println("CONFIG... ");
  WiFi.softAP(ssid, password);
  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.begin();
  delay(250);
  while (1)
  {
   REC_DATA();
   delay(10);
  }
  }
 Serial.println("Start123... ");
 memset(SMS, '\0', sizeof(SMS));
 read_ee(10);
 for (int i = 0; i < arraySize; i++)
 SSID12[i] = SMS[i];
 memset(SMS, '\0', sizeof(SMS));
 read_ee(30);
 for (int i = 0; i < arraySize; i++)
 PASSWORD12[i] = SMS[i];
 memset(SMS, '\0', sizeof(SMS));
 read_ee(50);
 for (int i = 0; i < arraySize; i++)
 DEVICEID[i] = SMS[i];
 memset(SMS, '\0', sizeof(SMS));
 read_ee(70);
 for (int i = 0; i < arraySize; i++)
 TOKEN[i] = SMS[i];
 WiFi.begin(SSID12, PASSWORD12);
 delay(250);
 while (WiFi.status() != WL_CONNECTED) {
 Serial.print(".");
 digitalWrite(25, !digitalRead(25));
 delay(500);
 cnt++;
```

```
   if (cnt > 100)
   ESP.restart();
  }
  cnt = 0;
  digitalWrite(25, HIGH);
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  delay(550);
  while ( Serial.available() > 0) Serial.read();
  delay(100);
  }
void loop()
{
 if (WiFi.status() == WL_CONNECTED) //Check WiFi connection status
 {
   rx_buf_index = 0;
   Channel = 0;
   BUZZ_INIT = 0;
   Serial.println("RDL_RFID");
   while (Channel != 1)
   {
    while (Serial.available()) {
      // get the new byte:
      char c = (char)Serial.read();
      Serial.print(c);
      Serial.println();
      // add it to the inputString:
      if ((c >= '0') && (c <= '9')) {
        RFID += c;

      } else {
        if (c == 13)
        {
          Channel = 1;

        }
      }
    }
```

```
}

if (Channel == 1)
{
  Serial.print("connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }

  Serial.print("Requesting URL: ");
  Serial.println(url);
  Serial.println(RFID);
      jason_string = "{\"USER\":\"" + RFID + "\",\"TOKEN\":\"" + TOKEN + "\",\"DEVICE-
ID\":\"" + DEVICEID + "\"}";
  RFID = "";
  Serial.println(jason_string);

  // This will send the request to the server
  client.print(String("POST /") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Accept: *" + "/" + "*\r\n" +
               "Content-Length: " + jason_string.length() + "\r\n" +
               "Content-Type: application/json\r\n" +
               "\r\n" + jason_string
               + "\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }
```

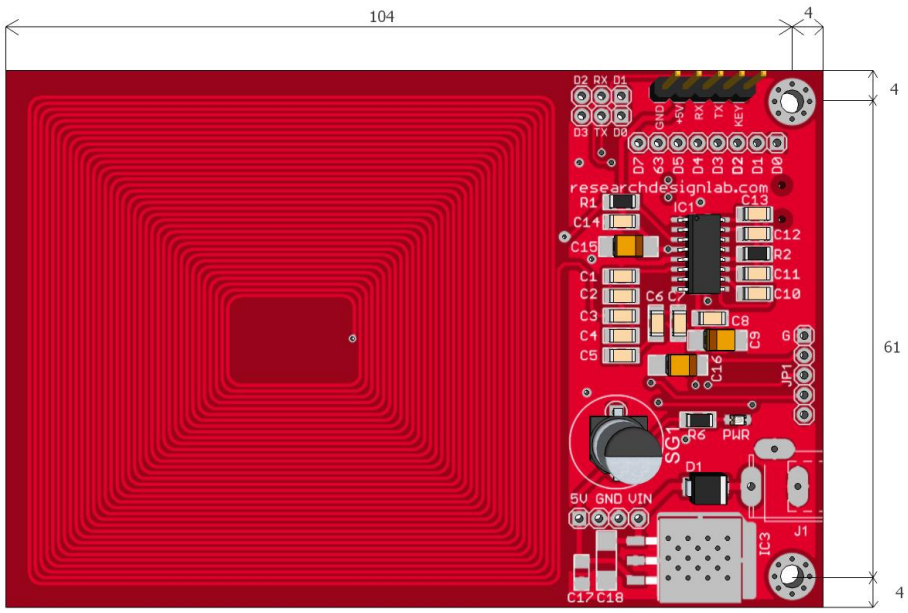
```
   // Read all the lines of the reply from server and print them to Serial
    while (client.available()) {
      String line = client.readStringUntil('\r');
      Serial.print(line);
      BUZZ_INIT = 1;
    }
    if (BUZZ_INIT == 1) {
      digitalWrite(4, HIGH);
      delay(500);
      digitalWrite(4, LOW);
      Serial.println();
      Serial.println("closing connection");
    }else {
      BUZZ_INIT = 0;
    }
  }
  Channel = 0;
}

else {
  WiFi.begin(SSID12, PASSWORD12);
  delay(250);
  while (WiFi.status() != WL_CONNECTED) {

    Serial.print(".");
    digitalWrite(25, !digitalRead(25));
    delay(500);
    cnt++;
    if (cnt > 100)
      ESP.restart();
  }
  cnt = 0;
  digitalWrite(25, HIGH);
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  delay(550);
}
delay(300);
```

```
  // loop while the client's connected
}

void read_ee (uint16_t framAddr)//, uint8_t values[])
{
  memset(SMS, '\0', sizeof(SMS));
  arraySize = EEPROM.read(framAddr);

  for (i = 0; i < arraySize; i++)
  {
    framAddr = framAddr + 1;
    SMS[i] = EEPROM.read(framAddr);

    // if(DIS_E==1)
    Serial.write(SMS[i]);
    delay(10);
  }

  // if(DIS_E==1)
  Serial.println();

}

uint8_t getConfigParKey_index(uint16_t paraddr)
{
  int i = 0;

  for (i = 0; i < sizeof(config_par_addr); i++)
  {
    if (config_par_addr[i] == paraddr)
    {

      return i;
    }
  }
  return NULL;
}

void REC_DATA()
```

```
{

 WiFiClient client = server.available();   // listen for incoming clients

 if (client) {                             // if you get a client,
  Serial.println("New Client.");           // print a message out the serial port
  String currentLine = "";                // make a String to hold incoming data from the client
  while (client.connected()) {
   while (client.available()) {
    char ch_byte = client.read();
    Serial.write(ch_byte);
    if (ch_byte == '$')
    {
     uint8_t x = 0, AX1 = 0, answer = 0;
     char* token;
     memset(SMS, '\0', sizeof(SMS));

     do {
      if (client.available() > 0) {
       ch_byte = client.read();
       Serial.write(ch_byte);
       SMS[x] = ch_byte;  AX1++;
       if ((SMS[x] == 10) && (SMS[x - 1] == 13))
       {
        answer = 1;
       }
       if (ch_byte == '=') AX1 = 0;
       x++;
       if (x > 150)x = 0;
      }
     } while (answer == 0);   // Waits for the asnwer with time out

     SMS[x] = '\0';
     x--;
     SMS[x] = '\0';
     x--;
     SMS[x] = '\0';
     AX1 = AX1 - 2;

     if (strstr(SMS, "AT+READ") != NULL)
```

```
        {

          token = strtok(SMS, "=");
          token = strtok(NULL, "=");
          ADDRS_EP = atoi(token);
          read_ee(ADDRS_EP);

          client.print(config_par_key[getConfigParKey_index(ADDRS_EP)]);
          client.println(SMS);
        }

        else if (strstr(SMS, "AT+WRITE") != NULL)
        {
          token = strtok(SMS, "=");
          token = strtok(NULL, "=");
          ADDRS_EP = atoi(token);
          token = strtok(NULL, "=");
          delay(1);
          EEPROM.write(ADDRS_EP, AX1);
          for (i = 0; i < AX1; i++)
          { ADDRS_EP = ADDRS_EP + 1;
            EEPROM.write(ADDRS_EP, token[i]);
            delay(1);
          }
          EEPROM.commit();
          Serial.println("OK");
          client.println("OK");
        }


        while (Serial.available()) {
          ch_byte = Serial.read();
        }
      }
    }
  }
}
```

9. DIMENSIONS

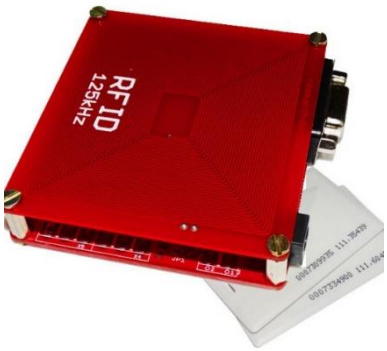


10. RELATED PRODUCTS

RFID Reader – USB

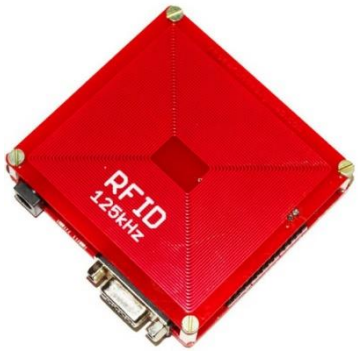
ORDER CODE: RDL/RFID/14/001/V1.0

RFID Reader - Serial



ORDER CODE: RDL/RRS/14/001/V1.0

Wi-Fi - RFID Reader 125 KHz



ORDER CODE: RDL/RFID-WIFI/14/001/V1.0

Industrial RFID Reader

ORDER CODE: RDL839

Wi-Fi RFID Reader-IoT Application

ORDER CODE: RDL738

Handheld UHF RFID Reader

ORDER CODE: RDL767

UHF RFID Reader

ORDER CODE: RDL822