International Journal on Advanced Science Engineering Information Technology

Vol.10 (2020) No. 5
ISSN: 2088-5334

Tire Pressure Monitoring System Using an Android Application

Carlos Robles Algarín[a,1], Jullbreider Pinto[a,2], Edgar Giraldo[a,3]

[a]*Facultad de Ingeniería, Universidad del Magdalena, Carrera 32 No. 22-08, Santa Marta, 470004, Colombia*
E-mail: [1]croblesa@unimagdalena.edu.co; [2]jullbreider@gmail.com; [3]edgargiraldo.ing@gmail.com

***Abstract*— Driving a vehicle with inadequate tire pressure can generate more ground friction, causing an increase in fuel consumption and, therefore in CO_2 emissions. Another consequence is uneven tread wear, affecting braking distance and vehicle control. This paper presents the implementation of a Tire-Pressure Monitoring System (TPMS), which is a system that alerts the driver of a vehicle about a tire pressure change. For this, four prototypes were designed to monitor and transmit the tire pressure and temperature of a vehicle. For the transmitter circuits design, ATmega328 microcontrollers, NRF24L01 transceiver modules, Honeywell NBP series pressure sensors and LM35 temperature sensors were used. In addition, a receiver that incorporates an NRF24L01 module to receive the signals coming from the transmitters was developed. The received data are sent via Bluetooth, with the HC-05 module, to an Android application developed in App Inventor, which is an open-source web application. To install the circuits on the tires, compact cases were designed in Solidwork, which were printed using the Prusa i3 3D printer. The results obtained demonstrate the effectiveness of the monitoring system and the accuracy of the measured data, as well as the relevance of the Android application to alert the driver in a simple way about any pressure change in the tires. These results suggest the possibility of using the prototype developed in realistic scenarios to monitor tire pressure in vehicles without this technology.**

***Keywords*— tire pressure monitoring system; transceivers NRF24L01; HC-05 bluetooth module; arduino nano; android application.**

I. INTRODUCTION

The pressure of the tires plays a fundamental role in the safety and fuel consumption of vehicles. When a vehicle moves with low tire pressure, it consumes more fuel and loses grip, especially on wet floors. This situation increases the braking distance, causes instability in the vehicle, and even runs the risk that the tire will burst. A poorly calibrated tire wears faster than the others, decreasing its lifespan and affecting the brakes and suspension of the vehicle in the long term [1], [2]. According to a study conducted by Michelin in Brazil, it was found that 45% of the drivers use tire pressure out of the ranges recommended by manufacturers. About 20% of cases are dangerous, because accidents can occur due to rupture or low tire pressure [3]. A similar study was carried out in Europe by Bridgestone, which found that 71% of drivers use tire pressure [4].

In this context, the Tire-Pressure Monitoring Systems (TPMS) are highlighted, which consist of an electronic device that allows measuring the air pressure in the tires of different types of vehicles [5]. Their primary function is to alert the driver to a loss of tire pressure. There are two types of TPMS: indirect and direct [6]. The indirect TPMS does not use sensors to measure the inflation pressure of the tires. In this method, the pressure is measured indirectly, from the speed of rotation of each tire. The system compares the speed difference of the tires using the speed sensors of the ABS module to determine when there is an error in the inflation pressure. This system has the disadvantage that it does not measure the exact pressure and temperature of the tires. Only when a reference speed value is achieved, this functionality is activated. Also, if the vehicle is stopped, it cannot perform the pressure measurement [7], [8].

Direct TPMS uses sensors installed on each tire to report pressure data to the electronic control unit [9], [10]. Compared with the indirect method, in this technique, more accurate data are obtained; but additional costs are generated due to the installation of the pressure sensors [11]. The temperature is another variable to consider in the calibration of tires. Fluctuations in temperature modify tire pressure. According to the Tire Rack Company, the general rule is that, for every 12°C of temperature change, tire pressure will increase by 2% in hotter temperatures and decrease in the same proportion in colder temperatures. This means that in standard tires (usually inflated between 30 and 50 psi) used in cars, trucks, and light trucks, the maximum change is one psi [12].

In some studies, different investigations have used direct TPMS to address the problem described by different approaches. A direct TPMS prototype was applied using wireless communication [13]. The prototype was energized from the rotation of the tires. Besides, pressure sensors, voltage-to-frequency converter, and frequency-to-voltage converter were used. Another study developed a direct TPMS with fuel leak detection, for which they used gas, pressure, and temperature sensors, the Atmega16 microcontroller, and radio frequency transmitters [14].

A direct TPMS highlights the use of the MAX1473 chip for wireless communication, as well as the development of an alert system to help drivers avoid accidents [15]. TPMS solution for a monitoring and alert module in real-time is characterized by being compact and easily fitted onto a tire [16]. In this sense, this paper presents the design and implementation of a direct TPMS that presents as a novelty a compact design made in Solidworks and printed using the Prusa i3 3D printer, which is easily adaptable to the valve stems of tires. Besides, it highlights the use of a radiofrequency transceiver and a Bluetooth module to monitor the pressure and temperature of each tire of the vehicle. It was decided to use the NRF24L01 transceivers in each tire due to their small size and low cost. This work is structured as follows: Section 2 presents the design and implementation of the transmitters and the receiver of the monitoring system. Section 3 shows the experimental results of the TPMS under different operating conditions. Finally, section 4 summarizes the main conclusions.

II. Materials and Methods

The designed TPMS is made up of 4 data acquisition and transmission devices, a receiving device, and an Android application to visualize the information in real-time. Next, the description of the prototypes designed for the transmitters, for the receiver and the Android application is presented.

A. *Prototype Designed for Transmitters*

Figure 1 shows the schematic diagram of the transmitter module, which is composed of the ATmega328 microcontroller, NBP pressure sensor, LM35 temperature sensor, NRF24L01 transceiver, and INA128 instrumentation amplifier [17]. This amplifier was used as a coupling device between the differential output of the pressure sensor and the microcontroller.

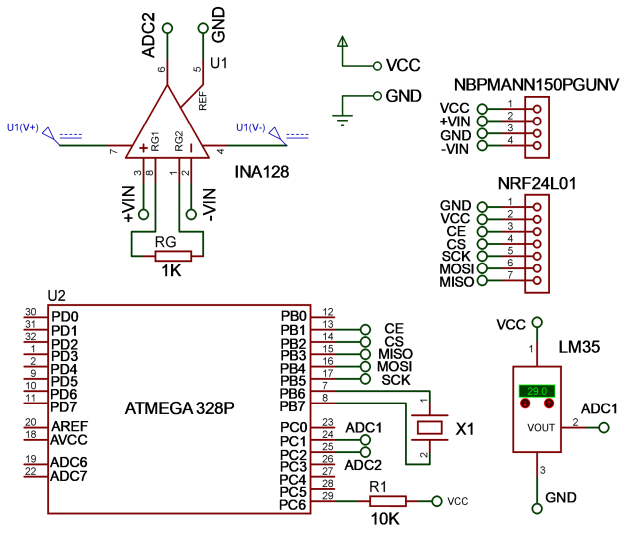


Fig. 1 Transmitter circuit diagram

To measure the pressure of tires, Honeywell NBP Series sensors were used, which are uncompensated sensors utilized to measure absolute and relative pressure in medical and industrial applications. These sensors have a primary performance and are ideal for applications where the user wants to make its compensation, calibration, and amplification. Sensors in this series have a wide operating temperature range from -40 °C to 125 °C and offer different pressure ranges and communication options. Table 1 shows the specifications of the pressure sensor used in this work.

TABLE I
Specifications of the Pressure Sensor

Specifications	Value
Accuracy	±0.25 %
Pressure Range	0 psi to 150 psi
Output	Analog
Measurement Type	Gage
Series Name	Basic NBP
Signal Conditioning	Unamplified
Output Calibration	No
Temperature Compensation	No
Supply Voltage	5 V
Supply Current	1.5 mA

Cheap and simple to use sensors (LM35) were used to measure the temperature. These sensors are characterized by having an output voltage linearly proportional to the centigrade temperature, with a scale factor of 10mV/°C and accuracy of 0.5 °C. To transmit the data of the pressure and temperature sensors, the NRF24L01 transceiver was used, which is an ultra-compact module with very low consumption. It works at 2.4GHz frequencies and is ideal for telemetry and peripheral control projects.

To process the signals of the pressure and temperature sensors, the ATmega328 microcontroller was used, which is also responsible for sending the acquired data to the transceiver module using the SPI protocol. To synchronize the transmitters and avoid collision of data in the receiver, the following notation was implemented:

- Tire 1: front left (P1, T1)
- Tire 2: front right (P2, T2)
- Tire 3: left rear (P3, T3)
- Tire 4: right rear (P4, T4)

Where P is the pressure and T the temperature.

Because the transmitters send the data through the same channel, a 4-byte data frame was designed:

- Byte 1: temperature.
- Byte 2: pressure.
- Byte 3: transmitter number.
- Byte 4: free byte for future improvements.

The codes developed for the four transmitters have the same structure, the difference lies in the characters that identify them: Transmitter 1: AA, Transmitter 2: BB, Transmitter 3: CC and Transmitter 4: DD. Figure 2 shows the code developed for transmitter 3. The libraries used are specific to the NRF24L01 transceiver and can be accessed from the official Arduino website. The code used for the other transmitters has the same structure, where the only difference is in byte 3 with which the transmitter number is identified. The programming of the microcontrollers was performed in the open-source Arduino Software [18]-[21].

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"

#define CE_PIN 9
#define CSN_PIN 10

#include "printf.h"

const uint64_t pipes[2] = {0xF0F0F0F0C4LL,
0xF0F0F0F0E1LL};
RF24 radio(CE_PIN, CSN_PIN);

int sentemp1=A1;//Temperature
float senpres1=A2;// Pressure
int pres1;

uint32_t rf=0XCC;// Transmitter 3
uint32_t t1;
uint32_t t2;
int in=0xF;

void setup() {
 Serial.begin(9600);
 printf_begin();
 radio.begin();
 radio.setDataRate( RF24_250KBPS );
 radio.setPayloadSize(sizeof(uint32_t));
 radio.disableCRC();
 radio.setRetries(0,0);
 radio.setAutoAck(false);
 radio.openWritingPipe(pipes[1]);
 radio.stopListening();
}
```

```
void loop() {
 int temp1=analogRead(sentemp1); //Temperature
    temp1=(temp1*200/1023);

 float respres1=analogRead(senpres1);// Pressure
      respres1=(respres1*3.9/1023);
      pres1=(respres1*1000/26);

 Serial.print ("ENVIANDO DE TRANSMISOR 3");
 Serial.println();

 Serial.print("TEMPERATURA 3 = ");
 Serial.print(temp1);
 Serial.print("C");
 Serial.println();

 Serial.print("PRESION 3 = ");
 Serial.print(pres1);
 Serial.println("Psi");

 t1=temp1;
 t2=pres1;

 uint32_t packet=0;
 packet=in|(rf<<4)|(t2<<12)|(t1<<20);

 Serial.print("packet= ");
 Serial.println(packet,HEX);
 Serial.println();

 radio.startWrite( &packet, sizeof(uint32_t));

 delay(300);
}
```

Fig. 2 Code developed for the transmitter 3

Finally, Table 2 shows the power consumption of each of the components that make up the circuit of one (1) transmitter.

TABLE II
POWER CONSUMPTION OF ONE TRANSMITTER

Device	Power (mW)
ATmega328 Microcontroller	16.5
Pressure Sensor	7.5
Temperature Sensor	0.57
Instrumentation Amplifier	2.5
Transceiver NRF24l01	0.5
Total	27.57

Taking into account the power consumption of the devices, the CR2450 battery to power the transmitters were used. These batteries have an output voltage of 3 V and a nominal capacity of 500 mAh. It should be noted that each transmitter uses three batteries: two are used to power the INA128 instrumentation amplifier and the other is used to power the other circuits.

B. Prototype Designed for the Receiver

Figure 3 shows the diagram of the circuit implemented for the receiver, which is composed of NRF24L01 module, the Arduino Nano board, and the HC-05 Bluetooth module. The operation of this device begins with the reception of the data from the transmitters of each tire. The reception is made through the radio frequency transceiver NRF24L01. The data received are processed by the Arduino Nano and sent to the HC-05 Bluetooth module, in order to visualize them in the Android application.

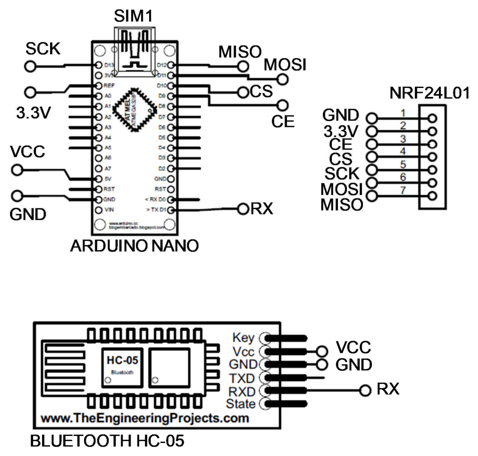


Fig. 3 Receiver circuit diagram

```
void receptor_rf();
{
 uint32_t packet;
 if ( radio.available())
 {
  radio.read( &packet, sizeof(uint32_t));
  x3=(packet&0x00000FF0)>>4; //No. of
Trasnmitter
  x2=(packet&0x000FF000)>>12; // Temperature
  x1=(packet&0xFF00000)>>20; // Pressure

  if (x3==0xAA)
  {
   EEPROM.write(DP1,x1);
   EEPROM.write(DT1,x2);
  }

  if (x3==0xBB)
  {
   EEPROM.write(DP2,x1);
   EEPROM.write(DT2,x2);
  }

   if (x3==0xCC)
  {
   EEPROM.write(DP3,x1);
   EEPROM.write(DT3,x2);
  }
```

```
 if (x3==0xDD)
   {
    EEPROM.write(DP4,x1);
    EEPROM.write(DT4,x2);
   }

  }
  delay(500);
}

void loop() {
 receptor_rf();

 P1=EEPROM.read(DP1);
 T1=EEPROM.read(DT1);
 P2=EEPROM.read(DP2);
 T2=EEPROM.read(DT2);
 P3=EEPROM.read(DP3);
 T3=EEPROM.read(DT3);
 P4=EEPROM.read(DP4);
 T4=EEPROM.read(DT4);

 sprintf(buffer, "%d,%d,%d,%d,%d,%d,%d,%d",
P1,T1,P2,T2,P3,T3,P4,T4);
 Serial.println(buffer);
 delay(500);
 }
```

Fig. 4 Code developed for the receiver

For communication between the receiving device and the Android application, the HC-05 Bluetooth module was used. This module was configured in the following way: name: Tire_Care, address: 20: 16: 02: 45: 93, password: 112233, operating mode: slave, and transmission speed: 9600 baud/s. As a processing device, an Arduino Nano board was used, which is a small, low cost and compact board. It is based on the Atmega328P microcontroller, and its characteristics make it ideal for applications where the size of the prototype is an important factor.

The receiver was programmed similarly to transmitters using the NRF24L01 module. Data are received using the RF receiver function, in which the transmitter that sends the packet is identified and stored in the variable x3. Then, the temperature and pressure data (variables x2 and x1) are stored in the EEPROM memory of the Arduino Nano. Once the data coming from the RF module are obtained, the next step is to send them via Bluetooth through a serial communication of 9600 baud/s. Figure 4 shows the code developed for the prototype of the receiver.

C. Android Application

This application was made in the cloud-based tool MIT App Inventor 2, which is a development environment to make applications for Android devices. This tool is characterized by being free and very easy to use due to its programming in a visual and block the way. Besides, it has an emulator to observe the design modifications in real-time. Figure 5 shows the user interface development, which allows to adjust the configuration and visualize the pressure and temperature of the 4 tires of the vehicle in real-time. The notation used for each tire is:

- Tire 1: front left (D.I.)
- Tire 2: front right (D.D.)
- Tire 3: left rear (T.I.)
- Tire 4: right rear (T.D.)

When a tire has a different pressure than the reference value, the application alerts the user by highlighting the state of the tire in red.

D. Case Design

For the design of the cases, the SolidWorks software was used to have a compact prototype that can be attached to the valves of the tires. Figure 6 shows the modeling performed. The upper part of the case, Figure 6 (a), was designed to install the battery and access it directly. With this part, it is possible to change the batteries without accessing the rest of the circuits of the prototype. In the upper part, Figure 6 (b), the other devices are installed, such as the sensors and the printed circuit board. This part has a thread that adapts to the value of the vehicle, which was modeled to avoid air leakage and incorrect temperature readings. In the circular hole, the pressure sensor is installed, while in the other hole, the temperature sensor is installed. In Figure 6 (c), the two assembled parts are shown.

Once the design of the case was finished, a file with extension. stl was generated to obtain the file with extension .gcode in the Repetier-Host software. This software is free and is used to calibrate the 3D printer.

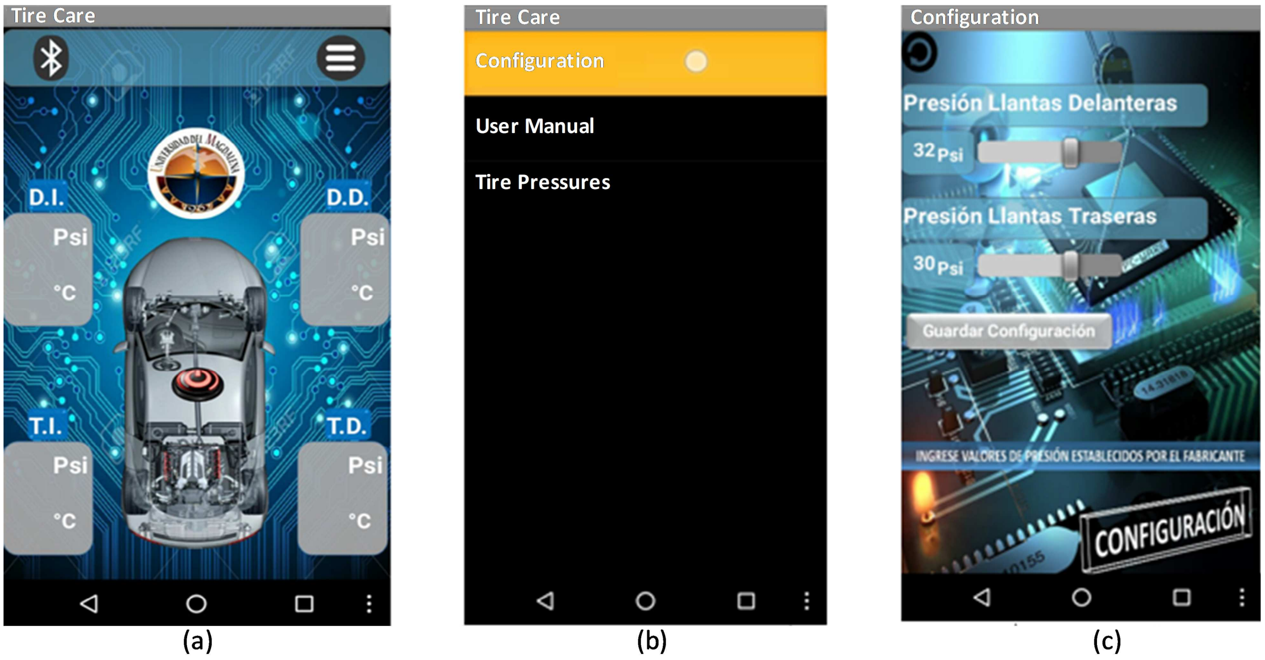


Fig. 5 (a) User interface of the Android application, (b) interface to select the options, and (c) configuration interface.

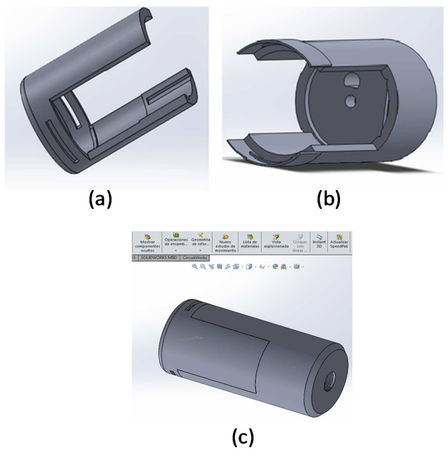


Fig. 6 Case designed for receivers: (a) upper part, (b) lower part, and (c) assembled case

Finally, the Prusa i3 3D printer was used to print the cases with an ABS material of 3 mm (See Figure 7 (a)). The total printing time was 79 minutes. Figure 7 (b) shows the prototypes for the four transmitters and the prototype of the receiver.

III. Results and Discussion

The initial tests focused on determining the reliability of the wireless communication system. The prototype was installed in a stationary vehicle with 32 psi tire pressure. A pressure sensor provided by the Certicar S.A company was used (Calibration certificate from the GyJ Metrología S.A.S company). The configuration of the four transmitters is the same because they all transmit on the same channel (0xF0F0F0F0E1LL). Figure 8 shows the results obtained for the transmitters installed on tires 1 and 2, where the temperature fluctuates by 1 °C between the two tires. The pressure obtained was 32 psi in each tire, which agrees with the initial value that was pre-established. In these tests, the transmission speed of the data was 250 Kbps in all the transmitters.

Concerning the transmitters installed on tires 3 and 4, the same temperature was obtained for both tires (29 °C) and the same pressure of 33 psi. This value differs by one psi from the initial value at which the four tires were calibrated. Figure 9 shows data sent by the receiver to the Android application. Figure 10 shows the results obtained directly in the Android application, in which a reference pressure of 32 psi was adjusted for the front tires and a reference pressure of 30 psi for the rear tires. In this way, the application highlights in red the rear tires to alert the driver of the vehicle, because the received value of 33 psi is higher than the reference value of 30 psi.

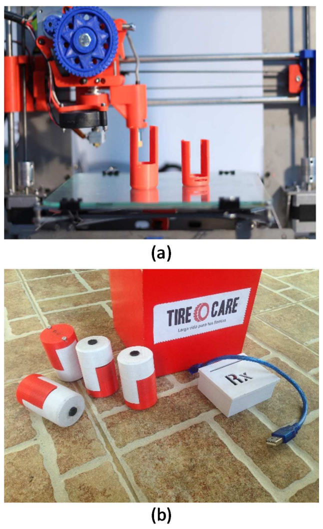
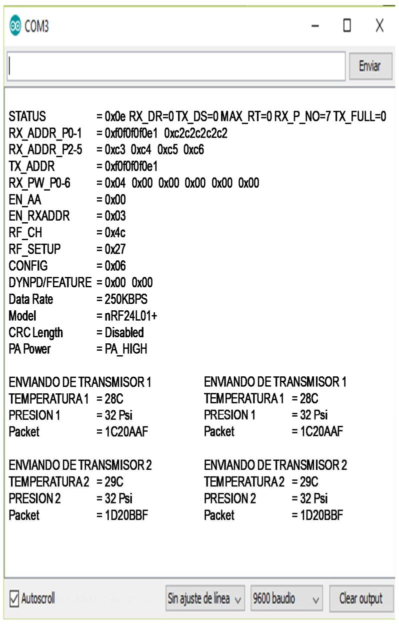


Fig. 7 (a) Prusa i3 3D printer, and (b) Prototypes designed for the 4 transmitters and the receiver

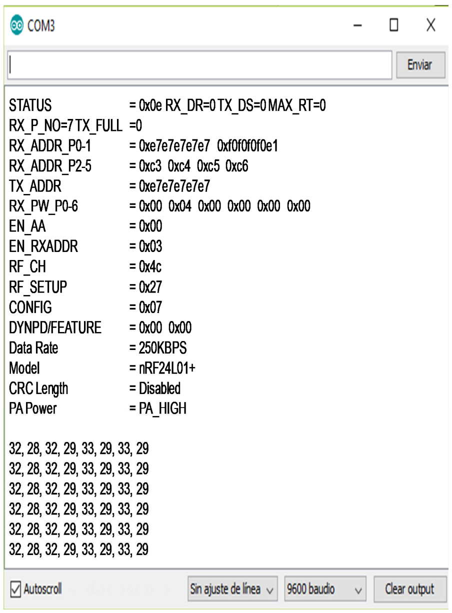


```
STATUS          = 0x0e RX_DR=0 TX_DS=0 MAX_RT=0 RX_P_NO=7 TX_FULL=0
RX_ADDR_P0-1    = 0xf0f0f0f0e1  0xc2c2c2c2c2
RX_ADDR_P2-5    = 0xc3  0xc4  0xc5  0xc6
TX_ADDR         = 0xf0f0f0f0e1
RX_PW_P0-6      = 0x04  0x00  0x00  0x00  0x00  0x00
EN_AA           = 0x00
EN_RXADDR       = 0x03
RF_CH           = 0x4c
RF_SETUP        = 0x27
CONFIG          = 0x06
DYNPD/FEATURE   = 0x00  0x00
Data Rate       = 250KBPS
Model           = nRF24L01+
CRC Length      = Disabled
PA Power        = PA_HIGH

ENVIANDO DE TRANSMISOR 1             ENVIANDO DE TRANSMISOR 1
TEMPERATURA 1   = 28C                TEMPERATURA 1   = 28C
PRESION 1       = 32 Psi             PRESION 1       = 32 Psi
Packet          = 1C20AAF            Packet          = 1C20AAF

ENVIANDO DE TRANSMISOR 2             ENVIANDO DE TRANSMISOR 2
TEMPERATURA 2   = 29C                TEMPERATURA 2   = 29C
PRESION 2       = 32 Psi             PRESION 2       = 32 Psi
Packet          = 1D20BBF            Packet          = 1D20BBF
```

Fig. 8 Results obtained for transmitters 1 and 2 using the Arduino serial monitor



```
STATUS             = 0x0e RX_DR=0 TX_DS=0 MAX_RT=0
RX_P_NO=7 TX_FULL  =0
RX_ADDR_P0-1       = 0xe7e7e7e7e7  0xf0f0f0f0e1
RX_ADDR_P2-5       = 0xc3  0xc4  0xc5  0xc6
TX_ADDR            = 0xe7e7e7e7e7
RX_PW_P0-6         = 0x00  0x04  0x00  0x00  0x00  0x00
EN_AA              = 0x00
EN_RXADDR          = 0x03
RF_CH              = 0x4c
RF_SETUP           = 0x27
CONFIG             = 0x07
DYNPD/FEATURE      = 0x00  0x00
Data Rate          = 250KBPS
Model              = nRF24L01+
CRC Length         = Disabled
PA Power           = PA_HIGH

32, 28, 32, 29, 33, 29, 33, 29
32, 28, 32, 29, 33, 29, 33, 29
32, 28, 32, 29, 33, 29, 33, 29
32, 28, 32, 29, 33, 29, 33, 29
32, 28, 32, 29, 33, 29, 33, 29
32, 28, 32, 29, 33, 29, 33, 29
```

Fig. 9 Results obtained for the receiver using the Arduino serial monitor

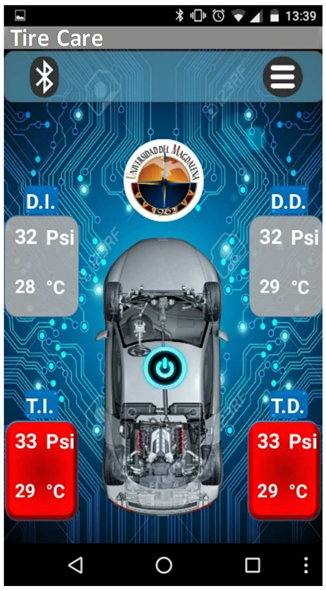


Fig. 10 Results obtained in the Android application

TABLE III
EXPERIMENTAL RESULTS OF TIRE PRESSURE

No.	Pressure Gauge	Prototype	Percent Error (%)	No.	Pressure Gauge	Prototype	Percent Error (%)
1	26	24	7.69	16	32	31	3.13
2	26	26	0.00	17	32	31	3.13
3	26	26	0.00	18	32	32	0.00
4	26	25	3.85	19	32	32	0.00
5	26	25	3.85	20	32	33	3.13
6	28	28	0.00	21	34	34	0.00
7	28	28	0.00	22	34	33	2.94
8	28	29	3.57	23	34	35	2.94
9	28	29	3.57	24	34	34	0.00
10	28	28	0.00	25	34	34	0.00
11	30	31	3.33	26	36	36	0.00
12	30	32	6.67	27	36	36	0.00
13	30	31	3.33	28	36	36	0.00
14	30	30	0.00	29	36	36	0.00
15	30	30	0.00	30	36	34	5.56

After verifying the operation of each block, 30 pressure measurements were performed on each tire. Table 3 shows the results obtained and the percent error that was calculated, taking as reference the pressure obtained with a calibrated pressure gauge. The biggest error obtained was 7.69%, which is equivalent to a pressure difference of 2 psi between the measured value and the reference value.

In general, the results obtained demonstrate the functionality and accuracy of the direct TPMS developed. The importance of the NRF24L01 transceiver and accuracy of temperature and pressure sensors are highlighted. The Android application is fully functional and simple to configure by users, allowing them to check the alerts to tire pressure changes in a didactic way. It also highlights the use of free tools such as Arduino, App Inventor 2, and Repetier-Host.

IV. CONCLUSIONS

In this paper, we proposed the implementation of a direct TPMS prototype. Experiments demonstrated the effectiveness of RF and Bluetooth communications with the NRF24L01 and HC-05 modules, respectively. The percent errors in the pressure data were in the range from 0% to 7.69%. This means that the TPMS developed meets the goal of the investigation. Also, it was possible to design a compact case that could be assembled in each of the four tires of a vehicle, avoiding pressure losses.

As future work, the research group will focus on developing a more compact device, using surface mount devices that could not be found in the local market. In addition, other types of pressure sensors and low-cost microcontrollers will be reviewed to reduce implementation costs and improve the accuracy of the data.

Acknowledgment

The research reported in this paper has been supported by the Vicerrectoría de Investigación of the Universidad del Magdalena.

References

[1] K. Y. Chen, and C. F. Yeh, "Preventing Tire Blowout Accidents: A Perspective on Factors Affecting Drivers' Intention to Adopt Tire Pressure Monitoring System," *Safety*, vol. 4, pp. 1-14, Apr. 2018.

[2] M. Toma, C. Andreescu, and C. Stan, "Influence of tire inflation pressure on the results of diagnosing brakes and suspension," *Procedia Manufacturing*, vol. 22, pp. 121-128, 2018.

[3] (2018) The Michelin website. [Online]. Available: https://goo.gl/jXhbJ3

[4] (2011) Portal Automotriz website. [Online]. Available: https://goo.gl/MoLxz3

[5] A. E. Kubba, and K. A. Jiang, "Comprehensive Study on Technologies of Tyre Monitoring Systems and Possible Energy Solutions" *Sensors*, vol. 14, pp. 10306-10345, 2014.

[6] J. Zhang, Q. Liu, and Y. Zhong, "A Tire Pressure Monitoring System Based on Wireless Sensor Networks Technology," in *Proc. of MMIT*, 2008, p. 30.

[7] H. Hariri, J. Kim, W. Kim, L. Frechette, and P. Masson, "Performance validation of printed strain sensors for active control of intelligent tires," *Appl. Acoust*., vol. 123, pp. 73-84, 2017.

[8] R. Isermann, and D. Wesemeier, "Indirect Vehicle Tire Pressure Monitoring with Wheel and Suspension Sensors," *IFAC Proceedings Volumes*, vol. 42, pp. 917-922, 2009.

[9] N. Hasan, A. Arif, M. Hassam, S. Ul Husnain, and U. Pervez, "Implementation of tire Pressure Monitoring System with wireless communication," in *Proc. of CCCA*, 2011, p. 3.

[10] Q. Kang, Z. Xie, Y. Liu, and M. Zhou, "125KHz wake-up receiver and 433MHz data transmitter for battery-less TPMS," in *Proc. of International Conference on ASIC*, 2017, p. 25.

[11] Q. Kang, X. Huang, Y. Li, Z. Xie, Y. Liu, and M. Zhou, "Energy-Efficient Wireless Transmissions for Battery-Less Vehicle Tire Pressure Monitoring System," *IEEE Access*, vol. 6, pp. 7687-7699, 2017.

[12] (2018) The TireRack website. [Online]. Available: https://goo.gl/d5tPzJ

[13] N. N. Hasan, A. Arif, and U. Pervez, "Tire pressure monitoring system with wireless communication," in *Proc. of CCECE*, 2011, p. 8.

[14] L. Chandreshkumar, J. Pranav, C. Hemraj, and G. Bokade, "Tire Pressure Monitoring System and Fuel Leak Detection," *IJERA*, vol. 3, pp. 345-348, 2013.

[15] T. Xiangjun, "The design and research of tire pressure monitoring system," in *Proc. of ICITB*, 2016, p. 17.

[16] C. Sharmila, and V. Vinod, "Design of a real-time tire pressure monitoring system for LMVs," in *Proc. of 2016 Online International Conference on Green Engineering and Technologies*, 2016, p. 1.

[17] A. Polo, P. Narvaez, and C. Robles Algarín, "Implementation of a Cost-Effective Didactic Prototype for the Acquisition of Biomedical Signals," *Electronics*, vol. 7, pp. 1-23, May. 2018.

[18] C. Robles Algarín, J. Callejas Cabarcas, and A. Polo Llanos, "Low-Cost Fuzzy Logic Control for Greenhouse Environments with Web Monitoring," *Electronics*, vol. 6, pp. 1-12, Sept. 2017.

[19] R. Kimura, M. Ohsumi, and L. Susanti, "Development of thermal insulation material using coconut fiber to reuse agricultural industrial waste," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 8, pp. 805-810, 2018.

[20] L. Benny, and P. K. Soori, "Prototype of parking finder application for intelligent parking system," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, pp. 1185-1190, 2017.

[21] F. A. Dwiputra, B. Achmad, Faridah, Herianto, "Accelerometer-based recorder of fingers dynamic movements for post-stroke rehabilitation," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, pp. 299-304, 2017.