BULETINUL INSTITUTULUI POLITEHNIC DIN IAȘI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iași
Volumul 69 (73), Numărul 4, 2023
Secţia
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ
DOI:10.2478/bipie-2023-0023

REAL TIME MONITORING DEVICE FOR HOME APPLICATIONS

BY

MARIUS HAGAN[1], OANA GEMAN[2], CRISTIAN AGHION[1,*], ADRIAN BREZULIANU[1], BO ESKEROD MADSEN[3] and ALEXANDER KRISTENSEN[3]

[1]“Gheorghe Asachi” Technical University of Iași, Faculty of Electronics, Telecommunications and Information Technology, Iași, Romania
[2]“Ștefan cel Mare” University of Suceava, Faculty of Electrical Engineering and Computer Science, Suceava, Romania
[3]Remoni Technologies RO SRL, Iași, Romania

Received: September 20, 2024
Accepted for publication: December 28, 2024

Abstract. This paper presents a device (data concentrator) capable of collecting data from different sensors connected by wire. Based on the RS485 communication protocol, it can package the data and send it further to cloud by means of GSM communication and FTP File Transfer Protocol. Such a device becomes very useful nowadays because it allows continuous monitoring of signaling in case of an emergency (fire, flood, security alarm). Special attention has been paid to the security of data transmission using a secure FTP channel.

Keywords: monitoring, sensors, microcontroller, safety, communications.

---

*Corresponding author; *e-mail*: aghion@etti.tuiasi.ro

1. Introduction

This paper presents a data concentrator device that receives information from sensors and then sends it further to a cloud server through a secure transmission channel. The increased interest in this area is evidenced by a number of scientific papers such as (Jeong *et al*., 2022) that use such Real Time inclusive devices to cover the medical field as well. Another paper (Woong *et al*., 2011) presents a power management system using sensors and a gateway acting as a data concentrator. Thus, through the gateway data is sent to cloud, from where users can access a web page where they can see and later control the electricity consumption. Finally, in the paper (Mudita *et al*., 2023) the authors study an IoT device where the sensor network nodes are managed predictably.

This paper focuses on researching the stage of monitoring and control of the so-called smart houses, respectively "intelligent houses" from the perspective of monitoring consumption (water, electricity, gas) and other essential information in order to transmit and (later) upload this data to cloud in order to be processed (Brezulianu *et al*., 2019).

Smart buildings cover a range of technical solutions, from the physical characteristics of the building and its location to the features and options offered by a smart home. We will now explore those smart house solutions that are versatile and can be applied in a wide range of situations without requiring changes to the physical structure of the house (Pracheta and Bhaskar, 2019). In general, a smart home, requires to collect as much information as possible from both inside and outside. To do this, different types of sensors are used, such as: security, fire and smoke, water consumption, electricity consumption, oxygen, light, ambient pressure, temperature and gas sensors inside/outside the house.

To complete the smart home concept, in addition to the information received from the sensors mentioned above, data communications (television and internet) are also needed.

The information obtained from these sensors is transmitted to one or more control circuits that control and monitor the "smart" home.

The use of sensor information plays a crucial role in the monitoring and control circuit, helping to make decisions on operating schemes. In this context, there are two concepts of control - temporal and behavioural:

- The temporal concept focuses on analysing the data received from the sensors to make decisions and control schemes that reach certain predefined thresholds, such as a minimum temperature of 24 degrees and a maximum temperature of 27 degrees.
- In terms of the behavioural approach, data from sensors is processed collectively to create a picture of the smart home. For example, if it is found that the temperature inside the house is too high, the outside temperature and the position of the sun will be checked to make an appropriate decision: activating the sunshades outside the building and turning on the air conditioning system.

Activating the blinds will reduce the level of light received by the building, but the light sensors inside will provide the necessary light for optimal operation in the rooms (kitchen, bedroom, living room et.). The implementation of this concept represents, to a large extent, a balanced approach to the use of sensor data, offering the possibility of developing a smart home in the broadest sense of the term.

In the evolution from the "smart house" concept to the next stage, called the "intelligent houses", additional options are added regarding increased data communication with the outside world and access to services offered by service companies (multimedia, cleaning, shopping, etc.). These options can fall into various categories, such as security, entertainment, social services, maintenance and provisioning, etc. For example, in the area of security, in addition to activating an alarm if sensors detect movement inside or outside the home, a smart home has additional options such as: turning on video surveillance cameras and spotlights, getting higher resolution images, calling the homeowner, calling a security company, blocking areas from safe access, and so on.

Figure 1 presents a general block diagram showing the connection of information from different sensors (including biological signals - medical sensors) to a data concentrator called "command circuit" that can communicate locally with a smartphone or in cloud (storage space - database) with the option to process the received information.

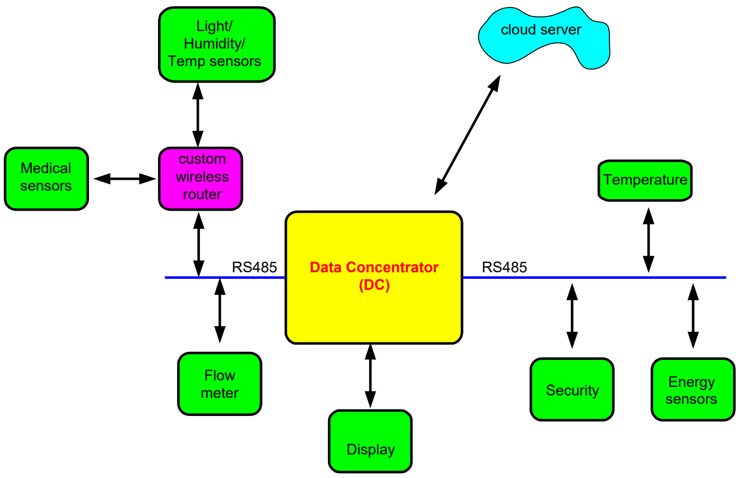

Fig. 1 – Simplified block diagram of the monitoring device.

2. Practical circuit

Commercial RS232-RS485 data conversion modules were used to achieve the data communication between sensors and the Data Concentrator (DC), in both directions (Fig. 2: Sparkfun RS485 Breakout module (left photo) and generic RS485 modules (right photo)). The data transmission protocol is RS485 due to the robustness of data transmission with low error rate per unit time.

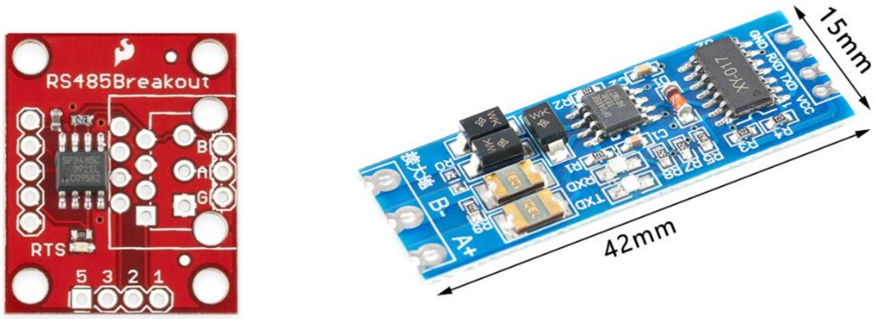

Fig. 2 – RS485 Module Breakout Sparkfun (red board) and generic RS485 Module (blue board).

These RS485 communication modules were connected to PIC16F1937 microcontrollers (Barleanu *et al*., 2012) in order to transmit data to a data concentrator in an encoded form following the RS485 protocol. The communication modules shown above receive input data from the PIC16F1937 microcontroller via the UART protocol (using the RX and TX pins), and these modules send the information onwards (half-duplex) to a Data Concentrator (Comsa and Haimovich, 2012). It should be mentioned that while the UART protocol/communication can work full-duplex, the RS485 protocol/communication can only work half-duplex. Although there is apparently this disadvantage, in reality the error rate of sending data is much lower with RS485 compared to UART (Valachi *et al*., 2009).

Figures 3 and 4 show the connection of the RS485 communication modules to the PIC16F1937 microcontroller development board.

Fig. 3 – Sparkfun development board and RS485 communication module.

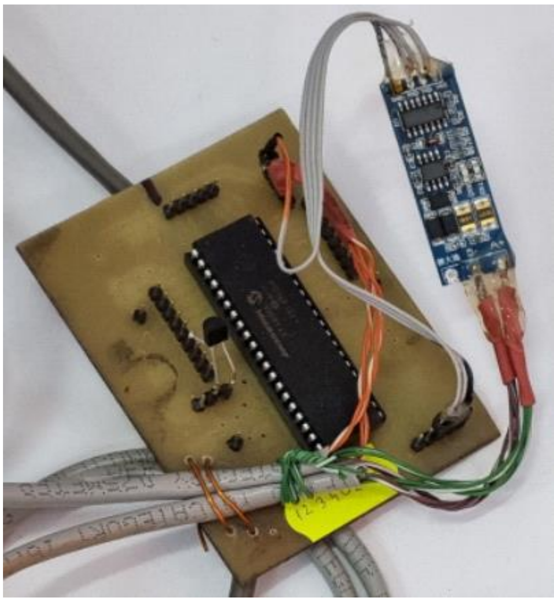

Fig. 4 – Development board with PIC16F1937 and RS485 communication module.

A total of 3 such development boards were made with the PIC16F1937 microcontroller, to which various sensors were connected. For specific test applications, temperature sensors, humidity sensors, safety and security sensors, atmospheric pressure sensors, etc. were connected and analogue voltages were applied using a potentiometer. In fact, all these tests were carried out in order to demonstrate the proper functioning of the data transmission from the sensors, through the microcontroller, then using the RS485 protocol to a Data Concentrator, which pushes the data to cloud by means of GSM communication (i.e., FTP - File Transfer Protocol). At this point, it matters less whether it is a temperature information, or a water, electricity or gas consumption value since the focus was on demonstrating the proper functioning of the whole data transmission chain: from the sensor to the storage of the sensor value in a file in cloud.

The following is the UART data transmission program implemented on the PIC16F1937 microcontroller. The information that is being transmitted is read from the analogue to digital converter, but it could be any input data received from external devices. For this reason, the program below is a general program for sending data using UART communication. It should be mentioned that the RS485 communication module is connected to the microcontroller pins RX and TX and it sends the data further to the Data Concentrator.

```
/*Program 1*/
#include <xc.h>
__PROG_CONFIG(1,0x3F84);      // config. uC (WDT=dis|OSC=int)
__PROG_CONFIG(2,0x3533);      // config. uC (PLL (OSC*4)=ON)
unsigned char z, a=0, rx1, rx2, rx3, rx4, rx5, rx6, rx7, rx8;
unsigned char digit1, digit2, digit3, digit4, digit5;
unsigned long c, b;
void init (void);
void interrupt ia2(void)      ;
unsigned int read_adc(void);
void main(void)
{
 init();
 while (1)
        {
       b = read_adc();
       b = b*8;
       b -= 500;
       if(b>0) b = b;
       else    b = 0;
       digit5 = (b % 10);
       digit4 = (b / 10)%10   ;
       digit3 = (b / 100)%10  ;
       digit2 = '+'-'0';
       digit1 = 0;
        }
}
unsigned int read_adc(void)
{
       GO_nDONE = 1;          //
       while(GO_nDONE);       //
       return ADRESH;         //
}
void init (void)
{
       OSCCON = 0xF0; ANSELA = 0xE0; TRISA = 0xE0; ANSELB = 0x00; TRISB =
0x00; PORTA = 0xFF;
       PORTB = 0xFF;  TRISC = 0xBF;  ANSELD = 0x00;
       TRISD = 0xFF;  ANSELE = 0x00; TRISE = 0x00;
       ADCON0  = 0x11; ADCON1  = 0x63; FVRCON = 0x8A;
       SPBRG = 0x33;  RCSTA = 0x90;  TXSTA = 0x20;
       ICER = 1;      TMR1IE = 1;    PEIE = 1;
       EIG = 1;
}
void interrupt ia2(void)      // interrupt
{//    RA4 = 1;
       if(RCIF)
       {
              rx8 = rx7;     rx7 = rx6;          rx6 = rx5;
              rx5 = rx4;     rx4 = rx3;          rx3 = rx2;
              rx2 = rx1;     rx1 = RCREG;
              if(rx8 == 0x31)       // slave address
              {
                     if(rx7 == 0x32) // read holding register
                     {
              if((rx6 == 0x33) && (rx5 == 0x34))    // address 0x3334
       {
              if((rx4 == 0x30) && (rx3 == 0x32))    // address 0x3334 //
              for(z=0; z<((rx4-0x30)*10 + (rx3-0x30)); z++)
              {
```

```
                         TXREG = digit2+0x30;while (!TRMT);
                         TXREG = digit3+0x30;while (!TRMT);
                         TXREG = digit4+0x30;while (!TRMT);
                                             }
                                     }
                         }
                 }
         }
}
```

In order to make Data Concentrator, the PCB was designed as shown in Fig. 5.

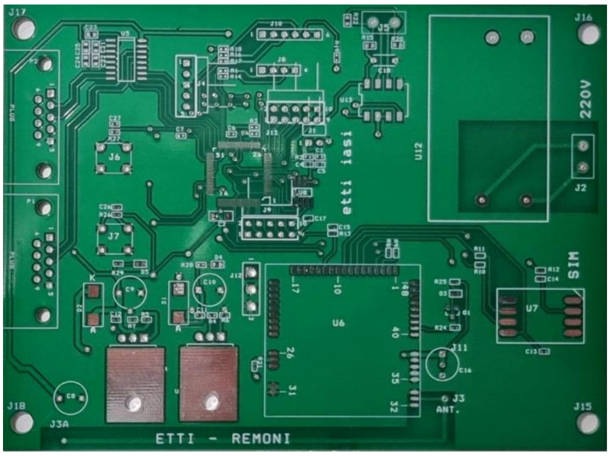

Fig. 5 – Printed Circuit Board for Data Concentrator.

The PCB motherboard was populated with electronic components for the Data Concentrator: the PIC32MX695F512L microcontroller, the SIM900D GSM module, the RS485 communication module and various electronic components, as shown in Fig. 6.

The PIC32MX695F512L microcontroller initiates sensor data readings for each sensor in turn. Thus, the PIC32MX695F512L microcontroller transmits a reading request associated with each sensor through the UART communication, by means of RS485 communication modules. As seen in Fig. 6, the 3 sensor modules are connected to the RS485 communication wires, which are part of the PIC16F1937 microcontroller. The PIC16F1937 microcontroller recognizes the code received from the PIC32MX695F512L microcontroller and in this case responds back to it with the corresponding sensor value.

Below is some of the code corresponding to the reading of several categories of sensors:

```
if(temp01==1) temp02 += 8+3; // water 12345678;01
if(temp01==2) temp02 += 3+3; // temperature +27;02
if(temp01==3) temp02 += 8+3; // gigacalorimeter 12345678;03
if(temp01==4) temp02 += 8+3; // electricity 12345678;04
if(temp01==5) temp02 += 8+3; // light sensor 12345678;05
```

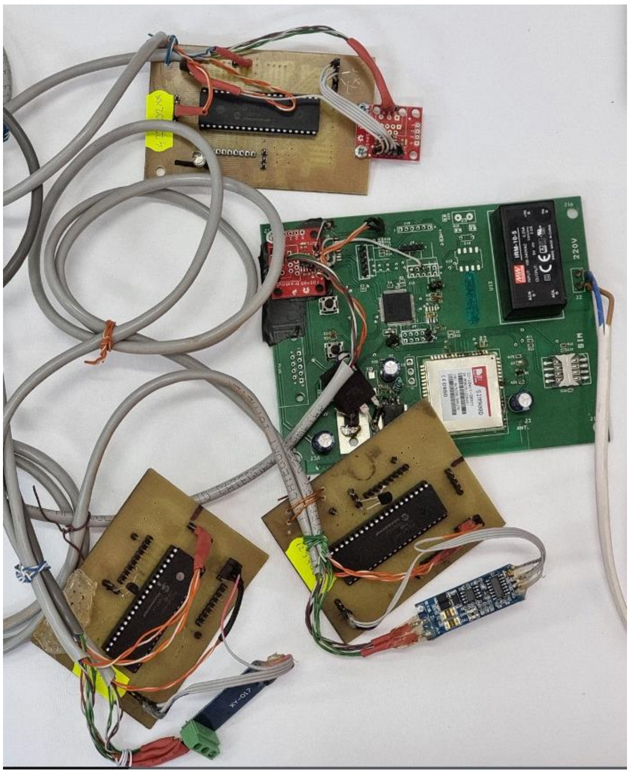

Fig. 6 – Data concentrator and three development boards containing sensors.

Thus, at time intervals set by the data concentrator (e.g. every 5 minutes), the PIC32MX695F512L microcontroller initiates the reading of one or more sensors connected to the RS485 communication bus.

Data acquisition is done by sensors installed and configured to measure certain parameters. This data collected by the sensors is critical in detecting problems that can lead to hazards to industrial facilities and personnel working around them.

The purpose of data collection is both to signal alert-type notifications in the event of thresholds being exceeded and to anticipate trends indicating rapid growth that may reach a critical threshold in a very short time. This predictive analysis allows proactive and effective intervention, minimizing risks and maximizing the safety and efficiency of the monitored system.

Data from the sensors is transmitted over the GSM (Global System for Mobile Communications) network to which the Data Concentrator is connected using FTP (File Transfer Protocol).

The data arrives on the FTP server as a formatted content file with separators between the information blocks.

The stages of data processing are as follows:

1) Data reception
2) File detection
3) Format file from original to JSON data type
4) Transmit JSON via API to an application that saves the data in the database.
5) Threshold check, prediction and notifications

Description of the actions taking place on the server:

1) Data reception is done on a server with text formatting similar to the one below.

Example line from a file received via FTP from the Data Concentrator.

```
002|2022:12:11|20:52:35|Vbat=4.02|T=+22C|P=01577|T2=+20C*
```

2) File detection is done by an Inotify service.

Inotify is a file notification system built into many Linux distributions. It allows file system events to be monitored and can be used to run commands or scripts when specific changes are detected, such as the creation of a new file

```
#!/bin/bash
DIRECTORY="/home/sensor_data/raw "
SCRIPT="/home/sensor_data/process_raw.sh"
inotifywait -m -e create "$DIRECTORY" | while read path action file; do
    echo "The file '$file' was detected , start procces_raw.sh"
    /bin/bash "$SCRIPT"
done
```

3) Formatting the text from the original to Json is done via a bash script similar to the following content:

```
files="/home/sensor_data/raw/*.txt";
for f in $files
from
   echo "Processing $f";
   sed 's/\r//' $f > $f.processing;
   sed -i 's/?/g' $f.processing
   destination=$(basename $f);
   destination=${destination//:/_};
   destination=${destination//|/_};

   output="";
   separator="";
   empty="";
   while read p; do
     IFS='|' read -r -a array <<< $p;
     read_data="${array[1]//:/-} ${array[2]}"
     timestamp="$(date --date "$read_data" +"%s")"
     temp=${array[4]/T=/};
     temp=${temp/C/};
     temp2=${array[6]/T2=/};
     temp2=${temp2/C*/};

     pressure=${array[5]/P=/}
     pressure=$(bc <<< "scale=2; ($pressure/1000 - 0.3975)/0.1025"); # V3
     echo "Pressure value for $value V: $pressure Bar"
     signal=${array[7]/S:/};
```

```
     signal=$(echo "$signal" | grep -o '[0-9]\+')
     echo "P2 $pressure";
     output="$output $separator $(cat <<-END
        { "ID":"${array[0]}", "date":"${array[1]}", "time":"${array[2]}",
"t1":"$temp",             "t2":"$temp2",             "p_bar":"$pressure",
"v_bat":"${array[3]/Vbat=/}",                   "P_VOLT":"${array[5]/P=/}",
"signal":"${signal}",                            "P_ABS":"${array[5]/P=/}",
"TIMESTAMP":"${timestamp}"}
        END
        )"
     separator=","$'\n';
   done < $f.processing
  output1="$output";
   output="[${output}]";
   echo $output;
   echo "$output" >> /home/sensor_data/post_processed/live_data.json;
   mv $f /home/sensor_data/post_processed/$destination.original;
   rm -f $f.processing

#################### JSON processing for API ##################
echo '{"measurement":[ ' > /home/sensor_data/alert/live.json
  echo "$output1" >> /home/sensor_data/alert/live.json;
 echo       ']       ,       "sensorBoxCode":       "SB1_L1_OP1"}'       >>
/home/sensor_data/alert/live.json;
/home/sensor_data/send_json.sh
done
```

4) Data transmission to the database is maintained by the following script:

```
#!/bin/bash
# URL ADRESS for backend
URL="http://IP_BACKEND:PORT/api/measurements/new"

# Path to JSON
JSON_FILE="/home/sensor_data/alert/live.json"

if [ ! -f "$JSON_FILE" ]; then
  echo "JSON file doesn't exist: $JSON_FILE"
  exit 1
fi

# Send data with POST request and wait for HTTP response code
status_code=$(curl -s -o /dev/null -w "%{http_code}" -X POST -H "Content-
Type: application/json" --data-binary "@$JSON_F$

if [ "$status_code" = "201" ]; then
  echo " The data was sent successfully."
else
  echo " Error: The data failed to send .HTTP code: $status_code"
fi
# Print HTTP status code
echo "HTTP code: $status_code"
```

After processing by the backend process, data will be recorded in the database, in a table called measurement_date in the form:

```
SELECT id, measurement_date, measurement_value, sensor_id
FROM public.measurement;
```

where sensor_id is the id of the sensor registered by the backend process and so we can graph measurement_date / measurement_value, as shown in Fig. 7 and Fig. 8.

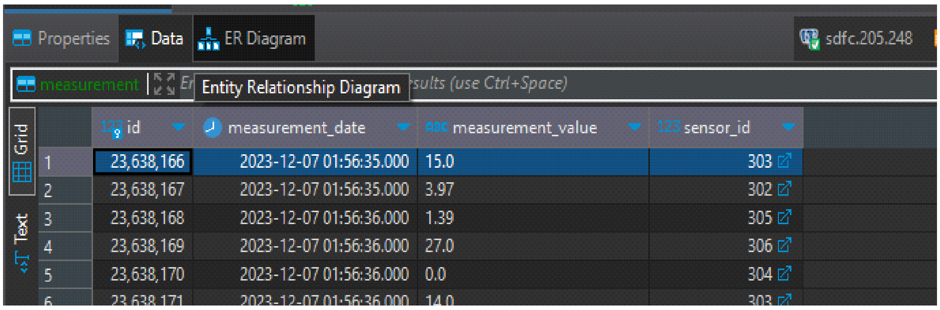

Fig. 7 – Data collection from a temperature sensor.

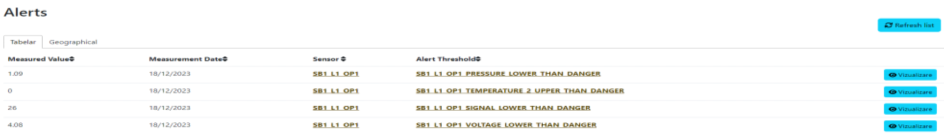

Fig. 8 – Collection of data from an atmospheric pressure sensor.

5) In the backend + frontend process verification application, you can view the last detected alerts. These alerts can be used to send e-mail and telegram notifications.

In Fig. 9 is presented the Data Concentrator map that shows the real-time measurements and data from the associated sensors:

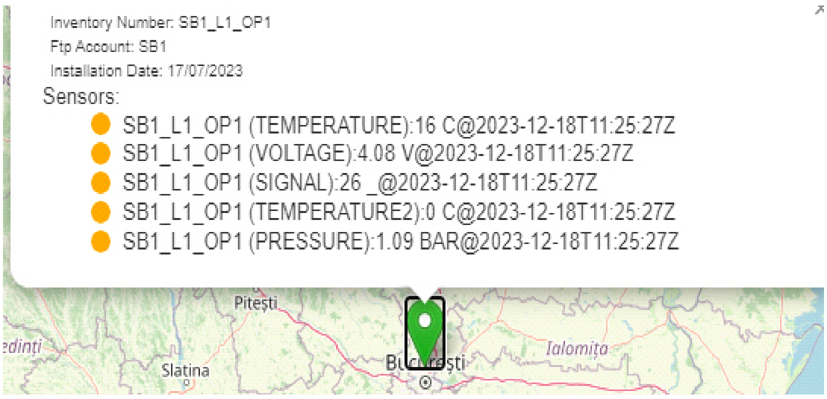

Fig. 9 – Viewing data from a specific Data Contractor (we can have more than one).

In Fig. 10 you can see the past measurements (history):

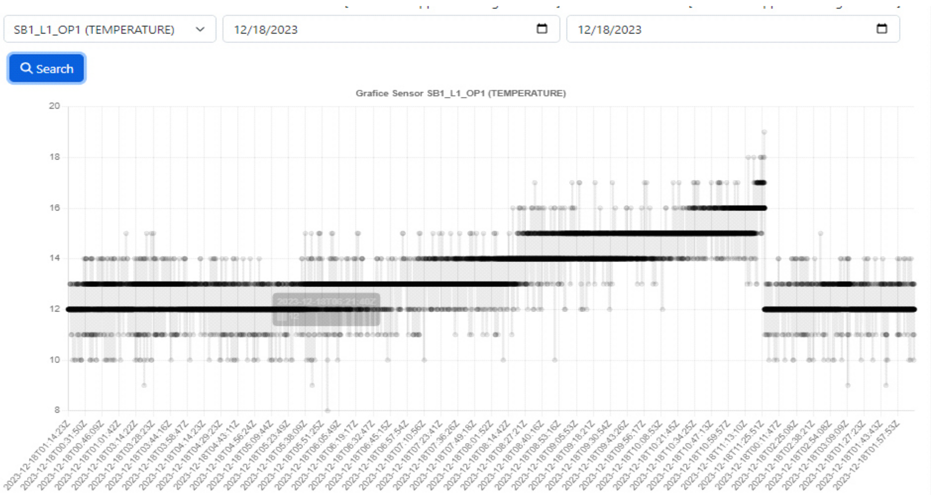

Fig. 10 – Viewing historical data for a specific sensor.

6) Prediction. Using a python script that connects to the database, we can read the values for the last N measurements of a parameter and, according to a threshold established for the difference between the values, we can determine a worrying trend of reaching the limits in the immediate future. We use the Pandas library, which is popular for data manipulation and analysis in Python.

Connecting to the Database:

```
database_url = "postgresql://user:password@host_address/database_name"
engine = create_engine(database_url)
```

Extracting Data from the Database:

execute an SQL query to retrieve the last num_samples records for a specific sensor_id. The results are stored in a Pandas DataFrame.

```
sql_query = f "SELECT measurement_date, measurement_value FROM
public.measurement WHERE sensor_id = {sensor_id} ORDER BY measurement_date
DESC LIMIT {num_samples};"
data = pd.read_sql_query(sql_query, engine)
```

Data Analysis and Alert Generation:

```
data['measurement_value'] = data['measurement_value'].astype(float)
if data['measurement_value'].diff().abs().mean() > threshold:
    print("Alert: Worrying trend detected for sensor_id", sensor_id)
```

3. Conclusion

This paper presents the practical implementation of a system consisting of three PIC16F1937 microcontrollers to which various sensors (temperature, humidity, atmospheric pressure et.) were connected so that data from these sensors can be sent to a Data Concentrator using RS485 wire data communication. The data concentrator contains the PIC32MX695F512L microcontroller and a SIM900D GSM module. The sensor data are concatenated in tables (matrices) and sent to a server as *.txt file using FTP File Transfer Protocol with the GSM module, based on user and password authentication. The realization of this project contains elements that can be easily procured, are inexpensive, so that its production on a large scale can easily be achieved for commercial purposes. Once the sensor data arrives on a cloud server, it is processed to create a database, to graphically visualize the sensor information and to create alerts if short-term changes in values are significant. These predictions are useful in usage scenarios where it is known what are the usual ranges of values for each sensor.

Acknowledgements. This paper was supported by the project: "Research and Implementation of Innovative Solutions for Monitoring Consumption in Technical Installations Using Artificial Intelligence", beneficiary S.C. REMONI TECHNOLOGIES RO S.R.L in partnership with "Gheorghe Asachi" Technical University of Iaşi, Financing Contract No. 400/390076/26.11.2021, SMIS Code 121866, financed by POC/163/1/3.

REFERENCES

Barleanu A., Baitoiu V., Stan A., *FIR Filtering on ARM Cortex-M3*, Proceedings of the 6th WSEAS European Computing Conference, WSEAS, 2012, pp. 490-494.

Brezulianu A., Aghion C., Hagan M. Gh., Geman O., Zbancoic M., Hemanth D.J., Hoang S., *IoT Based Heart Activity Monitoring Using Inductive Sensors*, MDPI Sensors 2019, 19(15), 3284, https://doi.org/10.3390/s19153284.

Comsa C., Haimovich A.M., *Performance Bound for Time Delay and Amplitude Estimation from Low Rate Samples of Pulse Trains*, Signal Proc. Conf (EUSIPCO), 2012 Proc. of the 20th European, 455-459.

Jeong J.-W., Lee W., Kim Y.-J., *A Real-Time Wearable Physiological Monitoring System for Home-Based Healthcare Applications*, Sensors 2022, 22, 104, https://doi.org/10.3390/s22010104.

Mudita Uppal, Deepali Gupta, Nitin Goyal, Agbotiname Lucky Imoize, Arun Kumar, Stephen Ojo, Subhendu Kumar Pani, Yongsung Kim, Jaeun Choi, *A Real-Time Data Monitoring Framework for Predictive Maintenance Based on the Internet of Things*, Complexity, vol. 2023, Article ID 9991029, 14 pages, 2023, https://doi.org/10.1155/2023/.

Pracheta P.H., Bhaskar P.C., *Development of Real Time Life Secure and Tracking* System for Swimmers, Conference: 3rd International Conference on Computing Methodologies and Communication (ICCMC), 2019, DOI: 10.1109/ ICCMC.2019.8819859.

Valachi A., Timis M., Danubianu M., *Some Contributions to Synthesis and Implementation of Multifunctional Registers*, 11th WSEAS Int. Conf. on

Automatic Control, Modelling & Simulation (acmos'09), Istanbul, Turkey, May 30 - June 1, 2009, pp.146-149.
Woong Hee Kim, Sunyoung Lee, Jongwoon Hwang, *Real-time Energy Monitoring and Controlling System based on ZigBee Sensor Networks*, International Symposium on Intelligent Systems Techniques for Ad hoc and Wireless Sensor Networks (IST-AWSN), Procedia Computer Science 5 (2011) 794-797, doi:10.1016/j.procs.2011.07.108.

DISPOZITIV DE MONITORIZARE ÎN TIMP REAL UTILIZAT ÎN APLICAȚII CASNICE

(Rezumat)

În acest articol este prezentat un dispozitiv (concentrator de date) capabil să strângă date de la diverși senzori conectați prin fir, folosind protocolul de comunicație RS485, să împacheteze datele și să le trimită mai departe pe un server în cloud folosind comunicația GSM și protocolul FTP File Transfer Protocol. Un astfel de dispozitiv devine foarte util în zilele noastre deoarece permite monitorizarea continuă a casei și semnalizarea unor alarme în cazul în care apare o urgență (incendiu, inundație, alarma de securitate). A fost acordată o atenție sporită securității transmiterii datelor folosind un canal securizat de trimitere a datelor – Secure FTP. A mai fost realizată o predicție a apariției unor evenimente de tip alarmă în condițiile în care datele provenite de la senzori determină o pantă crescătoare a valorilor și într-un timp definit, relativ scurt, inerent acest lucru va duce la depășirea unor valori limită.