STUDIA UBB PHYSICA, Vol. 63 (LXIII), 1-2, 2018, pp. 37-48
(RECOMMENDED CITATION)
doi:10.24193/subbphys.2018.04

DIGITAL DATA RECORDING OF TRANSIENT PHENOMENA IN RC CIRCUITS WITH ARDUINO

M. TODICA[1,*]

ABSTRACT. The work demonstrates the possibility to acquire data from any physics experiment using very simple equipment, Arduino Uno and Processing software. The algorithm was tested to record data of transient phenomena in RC circuits. Despite its simplicity the method provides accurate and reliable data, being usefully for didactic and scientific experiments.

Keywords: *digital data recording, RC transient phenomena, Arduino Uno.*

INTRODUCTION

Recording digital data from scientific experiment, simple or complex, is today a common stage performed with dedicated equipment. Generally the strategy of data recording is implemented into the recording equipment following specific protocols, which cannot be modified by the user. In these cases the experimental data can be collected quickly without special skills requested to the user, [1-4]. The simplicity of the use can lead to incomplete understanding of the procedure, and to the impossibility to adjust or to improve the parameters of the acquisition. Understanding the protocol of digital data acquisition, and implement this procedure for some physics

[1] "Babes-Bolyai" University, Faculty of Physics, M. Kogalniceanu No 1, 400084 Cluj-Napoca, Romania
**Corresponding author: mihai.todica@phys.ubbcluj.ro*

experiments, using simple equipments, represents the aim of this work. We propose a simple method for digital data recording using a common computer, the software Processing, available free on the internet, and the very popular programmable board, Arduino Uno. The method has been tested to record data of the transient phenomena in RC circuits. The main advantage of the method is its simplicity, economy of resources, and the availability for a large category of users. The method provides accurate and reliable data, being useful for didactic and scientific experiments.

EXPERIMENTAL

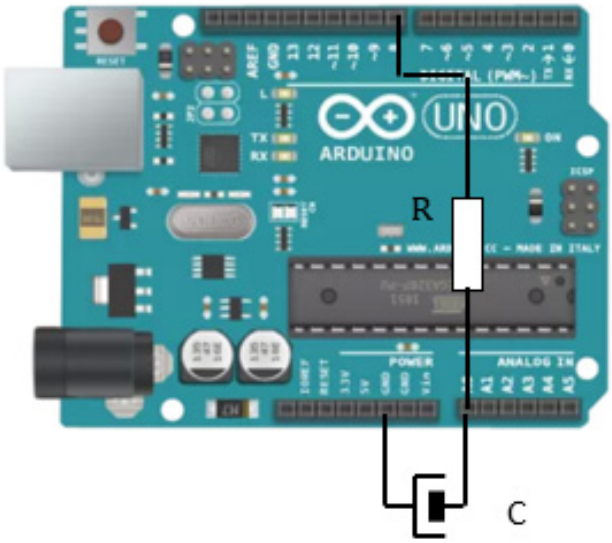

Fig. 1. RC circuit connected to the Arduino board

We used for this work the Arduino Uno board, which is a powerful programmable controller, easy to code with Arduino IDE, (C++ language). The board has many digital pins, which can be set as entries or outputs, and many analog pins which can be set as entries. The digital IN/OUT pins accepts and provides TTL signals, at low impedance. The analog pins are high impedance and accept signals up to +5V. The analog signals are converted to

digital on the range 0-1023 points, so that the data must be mapped from 1024 points to 5V if we want to have decimal output of the entry signal. For this project we used only one analog entry, A_0, which is connected to one end of the capacitor in order to measure the voltage at its ends. Due to the high impedance, we can neglect the influence of this entry on the measuring process. The RC circuit consists of a 24 μF capacitor connected between the ground and digital pin D_8 trough a resistor R=470 KΩ. The circuit is presented in figure 1. The recorded data are displayed using an ordinary computer and Processing software, available free on the internet, [5]. The data are displayed on the processing windows and then can be copied and processed with Origin, Kaleidagraph or other software for mathematically analyze.

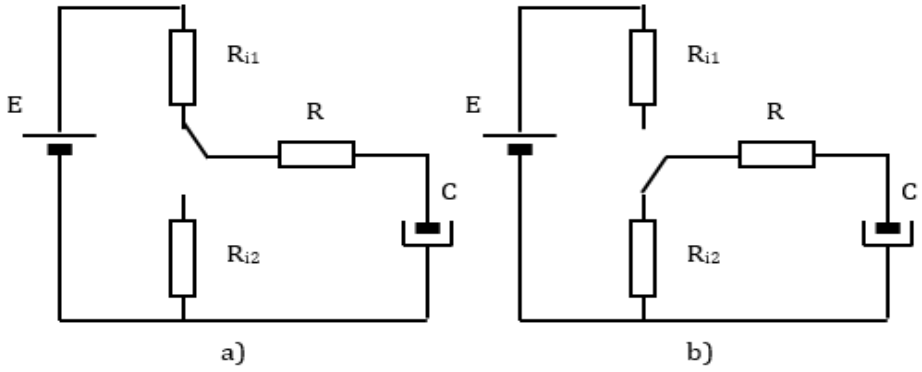

Fig. 2. RC charging-discharging equivalent circuits.
a) charging circuit; b) discharging circuit

RESULTS AND DISCUSSION

In the high state the pin D_8 is connected to +5V trough a low inner resistance R_{i1} of the Arduino board, and the capacitor C is charged via the resistors R and R_{i1}. The schema of the equivalent charging circuit is presented in figure 2a. The voltage on the capacitor is described by the equation, [6, 7]:

$$U_C(t) = E\left[1 - exp\left(\frac{-t}{R_{ch}C}\right)\right] \quad (1)$$

where *E* is the maximum voltage of the charging source, (in our case E=5V), *C* is the capacitance of the capacitor and R_{ch} the total charging resistance, (in our case $R_{ch} = R_{i1} + R$). In the low state the pin D_8 is connected to the ground trough the low inner resistor R_{i2}, (Fig. 2.b). The voltage of the capacitor is described by the equation:

$$U_C(t) = U_1\, exp\left(\frac{-t}{R_{dch}C}\right) \quad (2)$$

where U_1 is the voltage of the capacitor at the start moment of the discharging process, and R_{dch} the total discharging resistance, (in our case $R_{dch} = R_{i2} + R$). The inner resistances R_{i1}, R_{i2}, of the Arduino board are small compared with the charging-discharging resistance *R* and can be neglected. Then we can rewrite the equations 1 and 2 as follows:

$$U_C(t) = E\left[1 - exp\left(\frac{-t}{RC}\right)\right] \text{ and } U_C(t) = U_1\, exp\left(\frac{-t}{RC}\right) \quad (3)$$

When the code is running the pin D_8 is set up and down repeatedly ensuring the charging and discharging of the capacitor. The charging and discharging times, expressed in milliseconds, are described in IDE code by the variable *interval A*. These times are set equals, otherwise one of the processes, charging or discharging should be dominant and the system should reach a limit state, saturation charging, or completely discharging. The sampling time is defined by the variable *interval B*. We can change these parameters in order to obtain the best charging-discharging curves. To start the experiment we connect the Arduino board to the PC via USB port, we open the Arduino IDE, select the right Port, (to which the Arduino board is connected), and upload the following code.

```
//Arduino code

int AnalogPin0 = A0;
int ledState = LOW;
unsigned long previousMillisA = 0;
unsigned long previousMillisB=0;
unsigned long time=0;
const long intervalA = 20000; //charging time
const long intervalB = 1000; //sampling time
int led = 8; //charging-discharging pin

void setup()
{
Serial.begin(9600);
pinMode(led, OUTPUT);
}

void loop()
{
unsigned long currentMillisA = millis();
unsigned long currentMillisB = millis();
unsigned long time = millis(); //time counter
if (currentMillisA - previousMillisA >= intervalA)
{
previousMillisA = currentMillisA;
if (ledState == LOW)
{
ledState = HIGH;
} else
{
ledState = LOW;
}
```

```
digitalWrite(led, ledState);
}
if (currentMillisB - previousMillisB >= intervalB)
{
int Uc = analogRead(AnalogPin0);
Serial.print(Uc, DEC);
Serial.print(","); //send data to Processing
Serial.print(time);
Serial.print(".");
previousMillisB = currentMillisB;
}
}

//End of Arduino code.
```

The charging-discharging time and the voltage of the capacitor are sent to Processing via serial port. Processing is used only to display the data, the time and U_C. To do this we open Processing software and we upload the next code. We must set exactly the same port as set in the Arduino code.

```
//Processing code
import processing.serial.*;
PFont F;
Serial myPort;
String Uc="";
String SentTime="";
String data="";
int iSentTime;
float iUc;
int index1=0;
```

```
void setup()
{
F = createFont("Arial", 18, true);
size (1000, 600);
smooth();
myPort = new Serial(this,"COM5", 9600);
//change the port according to your Arduino
myPort.bufferUntil('.');
}
void draw()
{
line(50, height-50,width-50,height-50);
line(50, height-50,50,50);//draw frame axis with origin in (50;50)
textFont(F);
fill(20);
text("Uc (V)",70,40);
text("time (ms)",width-100,height-20);
text("RC transient",500,40);
fill(255,200,0);
ellipse(iSentTime+50, height-iUc*height/1023-50,10,10);
//displays time and Uc data

if (iSentTime >= width)
{
iSentTime=iSentTime-width;
background(200);
//refresh the screen when time exceeds the width
}
}

void serialEvent (Serial myPort)
{
```

```
data = myPort.readStringUntil('.');
data = data.substring(0,data.length()-1); //read the data from the string
index1 = data.indexOf(",");
Uc= data.substring(0, index1);
SentTime= data.substring(index1+1, data.length());
iUc=float(Uc);
iSentTime = int(SentTime)/100;
print(SentTime,",");
println(iUc*5/1023); //maps data from 0-1023 points to 0-5V
}
//End of Processing code.
```

The real data of charging-discharging, processed with Processing, are presented in figure 3. The data, in decimal form, are displayed on the windows working area of Processing, can be copied and then processed

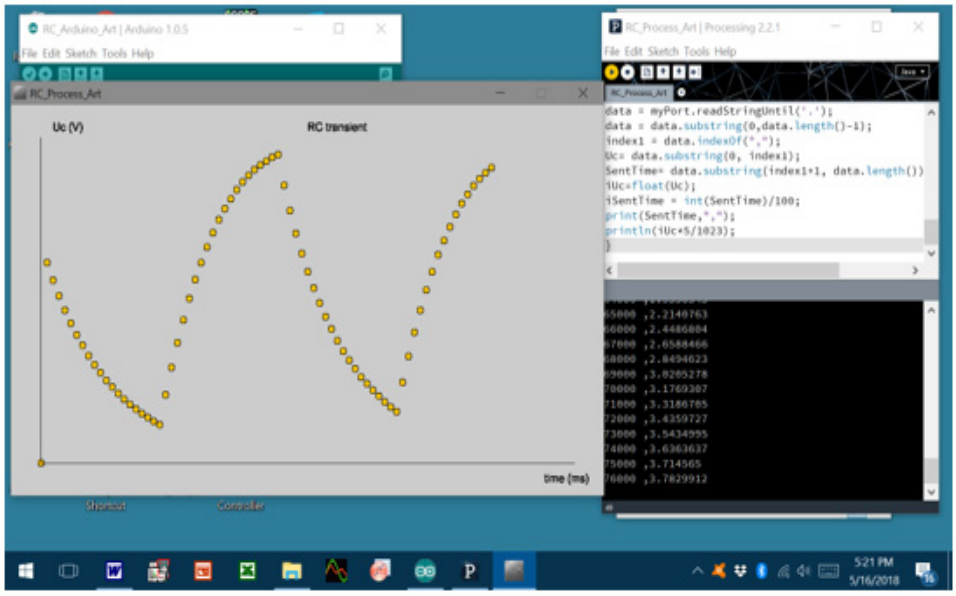

Fig. 3. Print screen of the working area of Processing. The data Uc and time are displayed on the working windows of Processing.

with specific software. The plot of recorded data with Kaleidagraph is presented in figure 4. Both curves, charging and discharging, represent exponential curves. However for quantitative analyze the discharging curve is most useful, allowing direct and very accurate measure of time constant $\tau = RC$. We can fit the experimental discharging data with the equation (3), but more suggestive is the following representation [8]:

$$ln\left(U_C\right) = ln\left(U_1\right) - \frac{t}{RC} \tag{4}$$

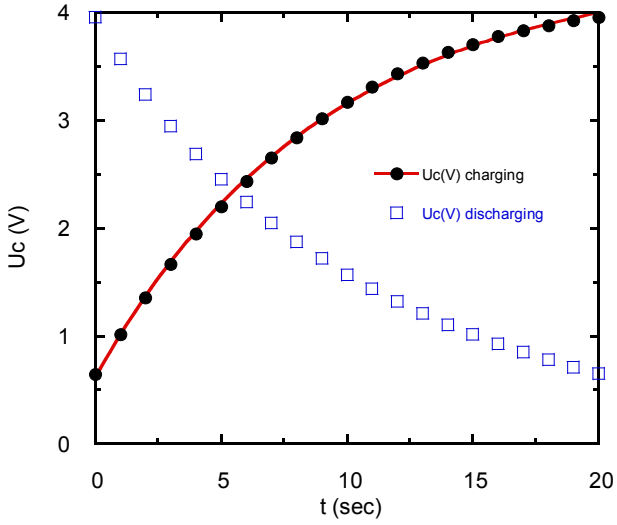

Fig. 4. The experimental data $U_c(t)$ of charging-discharging process. The charging data are fitted with equation 7.

We obtain a straight line with the intercept $ln\left(U_1\right)$ and the slope $-\frac{1}{RC}$, (Fig. 5). From here we found $U_1 = 3.9V$. We can see that the slope of the line is not affected by the value of the initial voltage U_1 of the capacitor. We found the value $\tau_{exp} = 11.22s$ in good agreement with the theoretical value $\tau_{theor} = 11.28s$ obtained with the known values C=24 μF and R=470 KΩ. The accuracy of calculation is 0.17%.

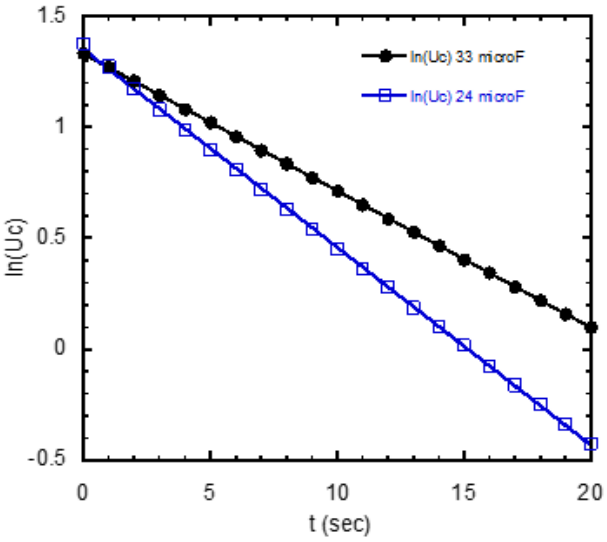

Fig. 5. Logarithmic representation of the discharging voltage of the capacitor for C=24 μF and C=33 μF.

The analyze of charging data is more complicate and gives less accurate values for the time constant. The equation 3 describes the charging process starting from zero voltage of the capacitor, (completely discharged). In our experiment the discharging process is not completely, so that the charging process starts from a given voltage U_{C0}. This voltage can be expressed considering the charging process starting at *t = 0* and ending at time t_0. The process is described by the equation:

$$U_{C0} = E\left[1 - exp\left(\frac{-t_0}{RC}\right)\right] \tag{5}$$

In our experiment we start the measurements at t_0 and stop the recording data at the instant *t*, when the voltage of the capacitor is $U_C(t)$.

We doesn't known the value of t_0, we can measure only the time interval $\Delta t = t - t_0$ from the start to the stop of the charging. The voltage $U_C(t)$ of the capacitor at the instant *t* is given by the equation:

$$U_C(t) = E\left[1 - exp\left(\frac{-(t_0 + \Delta t)}{RC}\right)\right] \tag{6}$$

Combining equations (5) and (6) we found the relation which describes the voltage of the capacitor during the charging from U_{C0} to $U_C(t)$.

$$U_C(t) = \frac{U_{C0}}{\left[1 - exp\left(\frac{-t_0}{RC}\right)\right]}\left[1 - exp\left(\frac{-(t_0 + \Delta t)}{RC}\right)\right] \tag{7}$$

U_{C0}, t_0, and $\tau^*_{exp} = RC$ are unknown parameters. We can fit the experimental data with equation (7), (Fig. 4). We obtained the value $\tau^*_{exp} = 9.04s$ smaller than the theoretical value $\tau_{theor} = 11.28s$. The accuracy of this method of is 19%, much small that the previous method. The source of errors is given by the great number of unknown parameters, three. In conclusion we can affirm that the method of discharging is the best way to calculate the time constant *RC*. To validate ours hypotheses we repeated the experiment with another capacitor *C*=33 μF*. Using the discharging protocol we obtained the value $\tau_{exp} = 16.2s$ and $U_1 = 3.8V$, (Fig. 5). Compared with the theoretical value $\tau_{theor} = 15.5s$ we obtained an accuracy of 4.5%. With equation (7) we found $\tau^*_{exp} = 15.8s$ that means an accuracy of 1.9%. In this case both algorithm of calculation provide good values for the time constant. Taking into account the obtained results, we suggests to apply both algorithms of analyze and average of the data.

CONCLUSION

The work demonstrates the possibility to use very simple equipment for digital data acquisition from physics experiments. We applied the method to collect the data from the transient RC experiment, but the protocol can be applied to other experiments. The method allows quantitative calculation of the parameters of the charging-discharging phenomena in RC circuits. The method is based on the use of very simple equipment, Arduino Uno board, and free software processing available on the internet. Despite its simplicity the method can be used for didactic and scientific experiments, providing accurate and reliable data.

REFERENCES

1. S. D. Anghel, Bazele Electronicii analogice si digitale, Presa Universitară Clujeană, Cluj-Napoca, **2007**, ISBN: 978-973-610-554-8
2. I. Burda, Microprocesoare si microcontrolere, Presa Universitară Clujeană, Cluj-Napoca, **2002**, ISBN 973-610-046-4.
3. D. Rădoiu, G.D. Popescu, Introducere în ştiinţa sistemelor de calcul, Editura Universităţii "Petru Maior", Târgu-Mureş, **1999**, ISBN 973-98726-7-0
4. A. Nicula, M. Todica, G. Buzas, S. Astilean, I. Berindean., Studia, Univ. "Babes-Bolyai", Physica, XXXIII, 2, (1988), pg. 15-19.
5. https://processing.org/download/
6. R. M. Eisberg, L. S. Lerner, Physics Foundations and Applications, McGraw-Hill, London, **1982**, ISBN 0-07-066268-1.
7. B. M. Yavorsky and A. A. Pinsky, Fundamental of Physics, Mir Publishers, Moscow, **1987**.
8. M. Todica, C. V. Pop, Fizica generala aplicata, Presa Universitara Clujeana, Cluj-Napoca, **2007**, ISBN (10)973-610-498-2; ISBN (13) 978-973-610-498-5