"Vasile Alecsandri" University of Bacău
Faculty of Sciences
Scientific Studies and Research
Series Mathematics and Informatics
Vol. 32, No 2 , 65 -78

LOW-COST EXPERIMENTS FOR COMPUTER SCIENCE EDUCATION WITH ARDUINO

DAN POPA

Abstract. Nowadays, the actual post pandemic context of computer science teaching is having some special attributes: big number of students, low budgets, need for STEAM, need of what we can call “pre-robotics” abilities. Otherwise, the costs should be kept low. This paper is a workaround the need of buying expensive Arduino kits. Instead, it is focused on how to teach elementary computer science skills with less accessories added to an Arduino Uno board and is proposing a set of lessons based on this idea and the principle of learning by discovery.

I. INTRODUCTION

Motivation: Even affordable for the in class use, Arduino Starter Kits are almost 6.3 to 8.5 times more expensive than the cheaper Arduino UNO Compatible board itself. Also, taking care of more than 200 small pieces or/and wires, counting them, putting them in storage places, replacing them when they are lost, may become a problem. This paper have proposing a set of programs designed with two goals in mind: Teaching how to program Arduino UNO Compatible boards *and* using no electronic devices or pieces attached to the board, but only the board itself. A second effect: beginners will not burn the board or the attached electronics, because such electronics did not exists (at least during first lessons). The experiments presented here are using a form of learning by discovery.

Keywords and phrases: Arduino Uno board, learning by discovery
(2020) Mathematics Subject Classification: 97P50

Experiment #1: Identifying resources

In this experiment we will connect the Arduino UNO board to the computer, (using a common USB printer cable). The computer is using Windows (in this case Windows 10) and a version of Arduino IDE (in this case Arduino IDE 1.8.10) but others configurations are possible. The board is seen as connected to a com interface (in this case COM 5). We have also tested the codes on the same computer but using Linux: Fedora 25 i386, and the Arduino IDE included in the distribution. The device is seen as */dev/ttyUSB0 or /dev/ttyUSB1.* Next paragraphs assumes the reader have no problem installing Arduino IDE.

The first experiment will check the board in order to find the on board leds and identify them.

```
// Schetch 27decA.
// Every led with the number i from 0 to 13 will flash for i+1 times.  This allow us to see if the D0
// output is connected to a led on the board. Note: Some Arduino like Mega will require a bigger
//number.
int i = 1;
int led = 13;


void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
for (int j=0; j <14; j++)
  { led = j;
    for (int k=1; k <= j+1; k++)
        {
        pinMode(led,OUTPUT);
        digitalWrite(led,HIGH);
        delay(100);
        digitalWrite(led,LOW);
        delay(100);
        }
  }
}
```

Interpreting the results for Arduino UNO Compatible: A led will flash one time, a led will flash two times, a led will flash 14 times. This leds are connected to the D0, D1 and D13 digital pins. Other boards may have more or less leds onboard.

Experiment #2: Test of the synchronous blink

In this test we will try to make a synchronous blink of two leds. If the test will *not* succeed we will try to imagine one led being somehow placed after a *NOT* circuit. As a consequence we have to send LOW instead HIGH to that Dx pin in order to have the same visual effect, in the same setup.

```
// Schetch: 27decB
// Flashing two onboard leds.
// See if they are flashing simultaneous or not !!
// OK. Dan P.

int led=1;
int led2=13;

void setup() {
  // put your setup code here, to run once:
  pinMode(led,OUTPUT);
  pinMode(led2,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
      digitalWrite(led,HIGH);
      digitalWrite(led2,HIGH);
      delay(100);
      digitalWrite(led,LOW);
      digitalWrite(led2,LOW);
      delay(100);
}
```

Interpretation of the results: If the leds are flashing synchronously, they are similarly connected, otherwise there is a NOT gate or even a simple transistor which acts as a NOT gate.

In our experiment: they did not flash simultaneously. After testing other pairs, (0,13),(0,1) the conclusion was we should treat the led connected to D13 like being after a NOT gate.

Experiment #3: Flashing two leds connected to D1 and D13

```
// Schetch: 28DecA
// Flashing two onboard leds on Arduino UNO
// If they are not in sync, we should negate one signal, i.e. changing LOW with HIGH

int led=1;
int led2=13;

void setup() {
  // put your setup code here, to run once:
  pinMode(led,OUTPUT);
  pinMode(led2,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
      digitalWrite(led,HIGH);
      digitalWrite(led2,LOW);
      delay(100);
      digitalWrite(led,LOW);
      digitalWrite(led2,HIGH);
      delay(100);
```

Interpretation: Synchronous blink, means we have managed to use the onboard leds, even connected via a NOT gate.

Experiment #4: Flashing three leds. Is D1 negated ?

Critics may argue that when comparing the D1 with D13 in the dual blinking led experiment may not show exactly which of the led was somehow negated. Let’s suppose this was D1. In this case the following experiment may leads to synchronous blinking of three leds.

```
// Schetch: 28decB
   //// Flashing three leds on thea Arduino UNO board.
   // Known:  led 1 and led 13 are not synchronized.
   // So they will be differently commanded.
   // The same was for: led3=0 ca si led2=13 – they did not sync.


   int led=1;
   int led2=13;
   int led3 =0;


   void setup() {
     // put your setup code here, to run once:
     pinMode(led,OUTPUT);
     pinMode(led2,OUTPUT);
     pinMode(led3,OUTPUT);
   }
   void loop() {
     // put your main code here, to run repeatedly:
          digitalWrite(led,HIGH);
          digitalWrite(led2,LOW);
          digitalWrite(led3,LOW);
          delay(100);
          digitalWrite(led,LOW);
          digitalWrite(led2,HIGH);
          digitalWrite(led3,HIGH);
          delay(100);
   }
```

Interpretation: If all three leds are blinking simultaneously, the negated led is D1. Otherwise it will be D13 (from previous experiments). In our case they did not blink simultaneously, so the only negated led is that one which is connected to D13.

Experiment #5: Can we change the name of the variables, in order to use a clear code ?

Actually, we know the answer is yes, but a teacher may select to show this example to his students, especially if the have problems in following and understanding the code.

```
// Schetch: 28decC
// Flashing three leds on the Arduino UNO Compatible board.
// The synchronous blink is programmed.
// Variable names are including the number of the on board pins
int led1=1;
int led13=13;
int led0=0;

void setup() {
  // put your setup code here, to run once:
  pinMode(led0,OUTPUT);
  pinMode(led1,OUTPUT);
  pinMode(led13,OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
        digitalWrite(led0,HIGH);
        digitalWrite(led1,HIGH);
        digitalWrite(led13,LOW);
        delay(100);
        digitalWrite(led0,LOW);
        digitalWrite(led1,LOW);
        digitalWrite(led13,HIGH);
        delay(100);
}
```

Interpretation: not necessary.

Experiment #6: Coding using subprograms

Using a subprogram to display the state of three leds on a single call may be an improvement. The teacher may want to use this example to show the benefit of using subprograms.

```
// Schetch: 28decD
// Prepairing other effects of lights.
// Just prepairing subroutines, a function and a procedure, but still producing the same effect,
// three blinking leds.

int led1=1;
int led13=13;
int led0=0;

void setup() {
  // put your setup code here, to run once:
  pinMode(led0,OUTPUT);
  pinMode(led1,OUTPUT);
  pinMode(led13,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
        display(HIGH,HIGH,HIGH);
}

int neg(int V)
{ if (V==LOW)
   return HIGH;
  if (V==HIGH)
    return LOW;
}

void display(int A,int B , int C)
{
        digitalWrite(led0,A);
        digitalWrite(led1,B);
        digitalWrite(led13,neg(C));
        delay(100);
        digitalWrite(led0,neg(A));
        digitalWrite(led1,neg(B));
        digitalWrite(led13,C);
        delay(100);
  }
```

Interpretation: In order to get complex effects made from simple effects we may use procedures. In order to call complex data transformation, the user we may use functions.

Note: The author especially avoided to discuss the “!” operator at this point.

Experiment #7: Let’s make the light moving or floating

In this experiment we will try to “move” the lights. Of course, the movement is only in the mind of the viewer. I also had used the code for a short film with a Christmas tree, this code had produced the lights.

```
// Schetch: 28decF
// Let’s try to move the light (or the shadow).
// The movement is in the mind of the viewer.

int led1=1;
int led13=13;
int led0=0;

void setup() {
  // put your setup code here, to run once:
  pinMode(led0,OUTPUT);
  pinMode(led1,OUTPUT);
  pinMode(led13,OUTPUT);
}
```

```
void loop() {
  // put your main code here, to run repeatedly:
      display(HIGH,LOW,LOW);
      display(LOW,HIGH,LOW);
      display(LOW,LOW, HIGH);
}

int neg(int V)
{ if (V==LOW)
   return HIGH;
  if (V==HIGH)
    return LOW;
}

void display(int B,int A , int C)
{
      digitalWrite(led0,A);
      digitalWrite(led1,B);
      digitalWrite(led13,neg(C));
      delay(200);
}
```

Experiment #7b: What happened when changing HIGH with LOW and LOW with HIGH

This experiment is left for the reader.

Experiment #8: Debugging a three led "moving light" program using the serial interface

In this experiment we will try, (and not succeed), to debug a three led "moving light" program, using the serial interface connected to the USB port and the Tools -> Serial Monitor window of the Arduino IDE. Uncomment the Serial.xxxxx statements and try the program.

```
// Schetch: 28decG
// Trying to debug a three led "moving light" program.
// Notice: It is not possible to simultaneously use D0 and  D1  for serial transmission and
output!
```

```
int led1=1;
int led13=13;
int led0=0;

void setup() {
  // put your setup code here, to run once:
  pinMode(led0,OUTPUT);
  pinMode(led1,OUTPUT);
  pinMode(led13,OUTPUT);
  //Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
        display(HIGH,HIGH,LOW);
        display(LOW,HIGH,HIGH);
        display(HIGH,LOW, HIGH);
}

int neg(int V)
{ if (V==LOW)
    return HIGH;
  if (V==HIGH)
     return LOW;
}
void display(int B,int A , int C)
{
        digitalWrite(led0,A);
        digitalWrite(led1,B);
        digitalWrite(led13,neg(C));
        delay(200);

        //Serial.print(" A=");
        //Serial.print(A);
        //Serial.print(" B=");
        //Serial.print(B);
        //Serial.print(" C=");
        //Serial.print(C);
        //Serial.println();
```

Interpretation: If it succeed the serial is independent of the output of the D0, D1, (D13) pins. If it not succeed the serial interface uses some of this pins. Visual inspecting the board we can find a notice near D0 and D1 pins: TX -> 1 , RX <- 0.

Experiment #9: Using the serial interface and D13

Due to the fact that only one led from the Arduino UNO Compatible Board is not connected to the serial interface, this makes it the single led which can be commanded using the Serial Monitor Window of the Arduino IDE environment.

Speaking of the Roumanian resources, a teacher may want to find this example in the volume [PEC]: https://ftp.utcluj.ro/pub/users/peculea/CAN/Laboratoare/Introducere%20Arduino/Introducere%20Arduino.pdf At page 15, the example can be found, together with a list of References.

Experiment #10: Interrupts

The site www.tutorialspoint.com offered an Arduino book which includes an example concerning the use of interrupts.

It is made using: *attachInterrupt(digitalPinToInterrupt(**pin**), **blink**, CHANGE)*; in the setup() routine , where the ***pin*** variable is set to **1**. As an effect, of the previous call, every change on that ***pin*** will call the *interrupt service routine*, which in this case is called **blink**.

Blink will produce a change of a state variable, switching between two states. One is negated of the other. The state variable starts with a LOW value, but you can use HIGH too.

The *loop()* is simply sending the value of the state variable (which is LOW or HIGH) to the led attached to the D13 digital output.

The tutorial recommended to use a *volatile* variable for the state, declared as: *volatile int state = LOW.*

The code is leaved as an exercise for the reader.

Interpretation: The D13 led flashing with a lower frequency than the D1 serial input, being a visual witness of the fact that the interrupt service routine is called and the state is changing.

Note: Some boards may require: *attachInterrupt(pin, interruptserviceroutine, mode),* the *mode* being: CHANGE or LOW or even FALLING.

Experiment #11: PWM output. Fading effect.

Some digital pins of the Arduino UNO Compatible boards are labeled with a "~" sign indicating that they can use a PWM modulation. It allows the programmer to use an *analogWrite(pin, value)* to produce an output which may vary from a low (0) to a high value (255). Unfortunately the D13 pin is not labeled with "~", meaning that it is not PWM capable. Pins D0 and D1 are in the same situation, on Arduino UNO Compatible Board. As a result, if the reader opens a book like [CCG], [TPA] or [ILC], the experiment explained in the PWM chapter is surely including a more or less lot of external components. The set varies from a resistor and a led to a more complex environment of an actuator.

Idea: The teacher can show the idea of PWM using simply the Arduino board, by connecting with a wire (can be a simple paper clip !) the selected PWM digital output with the D13 pin **in read mode.**

Note: Never connect a PWM output with GND, 5Vcc or any other output, it may damage the circuit, if no protections are included.

A bit different version of the source from [CCG] pp. 275-276 can make the trick and produce a fading effect on the D13 led, when D11 and D13 are connected.

```
// PWM demo – without extra electronic devices.

int led = 11;         // close to pin 13
int intrare = 13;   // the onboard led is here

// After uploading the code, connect with a wire (at least a paper clip) the followings pins:
// 13 - input with LED (and not function) with 11 -  PWM output

int lumina = 0;
int delta = 5;

void setup() {
  // put your setup code here, to run once:
  pinMode(intrare,OUTPUT);
  digitalWrite(intrare,HIGH);  // try to put the led  OFF
  pinMode(led , OUTPUT);
  pinMode(intrare, INPUT); // Otherwise do not connect the pins.
}

void loop() {
  // put your main code here, to run repeatedly:

  analogWrite(led,lumina);  //  lumina is 0 in the beginning !
  lumina += delta ;         // increasing lumina
  if (lumina == 0 || lumina == 255) {
    delta = -delta;
    delay(500);
  };
  delay(40);  //waiting 40 ms
}
```

Interpretation: The PWM mode can create a fading effect simulating an apparent continuous variation of the light of a led or speed of a motor.

Experiment# 12: Serial communication with two boards

In [ILC] Chap.4 there is a little experiment concerning the serial (USART) transmission between two boards, using nothing more than three wires: TX, RX, and GND. The example is using D0 and D1 pins and a cross-over cable.

Conclusions

Even if this paper did not exhaust the list of possible experiments, we have proof that is perfectly possible to present the basic aspects concerning the use and programming of the Arduino UNO Compatible Board without using extra electronic devices, than, let's say, a paperclip. This is lowering the cost of the first lessons, with a factor from 6.3 to 8.5, depending of the provider of materials involved (computer is excluded).

References:

[PEC] https://ftp.utcluj.ro/pub/users/peculea/CAN/Laboratoare/Introducere%20Arduino/Introducere%20Arduino.pdf pg.15

[TPA] *** www.tutorialspoint.com – Simply Easy Learning Arduino, pp. 114, pp. 138.

[CCG] Catalin Cazan Gheorghiu – Electronica si Robotica , Primii Pasi, Libris Editorial, 2018. Pp.275-276

[ILC] Sebastian Petru SABOU - Îndrumător laborator microcontrolere ARDUINO - U.T. PRESS CLUJ-NAPOCA, 2018 , ISBN 978-606-737-341-7, pp. 11 and Chap. 4.

"Vasile Alecsandri" University of Bacău,
Department of Mathematics and Informatics,
Calea Mărășești 157, Bacău 600115, ROMANIA
e-mail: danvpopa@ub.ro, popavdan@yahoo.com