Zwiększanie umiejętności programowania z mikrokontrolerami

Opracowane przez Mirela TIBU

"Grigore Moisil" Theoretical Highschool of Informatics , Iasi, Rumunia

A Trainers Toolkit To Foster STEM Skills Using Microcontroller Applications

Co-funded by the Erasmus+ Programme of the European Union

Project No. 2019-1-RO01-KA202-063965

This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Spis treści

Informatyka

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Cel

Wyrażanie twórczego sposobu myślenia, w konstruowaniu i rozwiązywaniu problemów

Kształtowanie nawyków stosowania określonych algorytmicznych pojęć i metod informatycznych w podejściu do różnorodnych problemów

Przejawianie postawy wobec nauki i wiedzy

Przejawianie inicjatywy i chęci do podejmowania różnorodnych zadań

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Opis

- Podejście ITC - Mikroprocesor VS Mikrokontroler
  - ✓ identyfikacja obszarów, w których komputery/systemy wbudowane są wykorzystywane w życiu codziennym
  - ✓ opisywanie architektury sprzętowej komputera i systemu wbudowanego
  - ✓ porównywanie cech mikroprocesorów i mikrokontrolerów
- Podejście programistyczne - zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolerów
  - ✓ strukturalne twierdzenia programistyczne - decyzje, pętle (IF, WHILE, FOR)
  - ✓ deklarowanie i wywoływanie funkcji void i non-void
  - ✓ użycie tablic w aplikacjach
  - ✓ analizowanie funkcjonalności urządzeń Arduino w celu rozpoznania etapów realizacji poleceń programistycznych

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Cele kształcenia

- Rozpoznawanie zastosowań komputerów w życiu społecznym - świadomość wpływu systemów wbudowanych na życie codzienne
  - Identyfikacja podobieństw i różnice między mikroprocesorem a mikrokontrolerem w architekturze systemów komputerowych i wbudowanych
  - Ćwiczenie implementacji elementów programowania strukturalnego - decyzje, pętle, funkcje; reprezentacja i wykorzystanie danych tablicowych
  - Wizualizacja efekt do wykonywania różnych sekwencji programów przez urządzenia oparte na mikrokontrolerach

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Co-funded by the Erasmus+ Programme of the European Union

Metodologia nauczania

- Objaśnienie
- Demonstracja
- Konwersacja
- Algorytmizacja
- Implementacje

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Grupa docelowa

Uczniowie szkół ponadgimnazjalnych - klasy 9-10

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

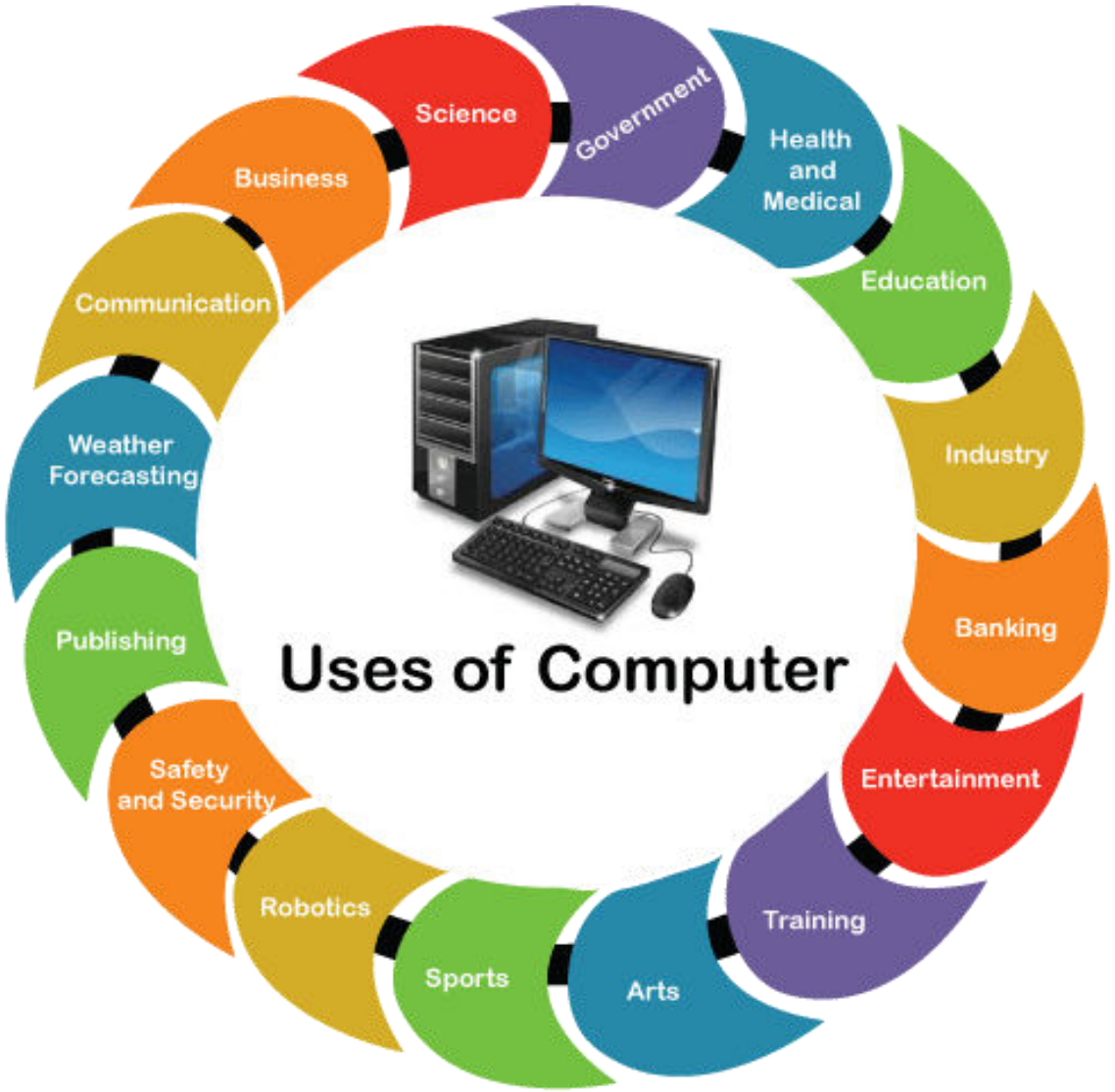
Co-funded by the Erasmus+ Programme of the European Union

Komputery i systemy wbudowane

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Komputery i życie codzienne



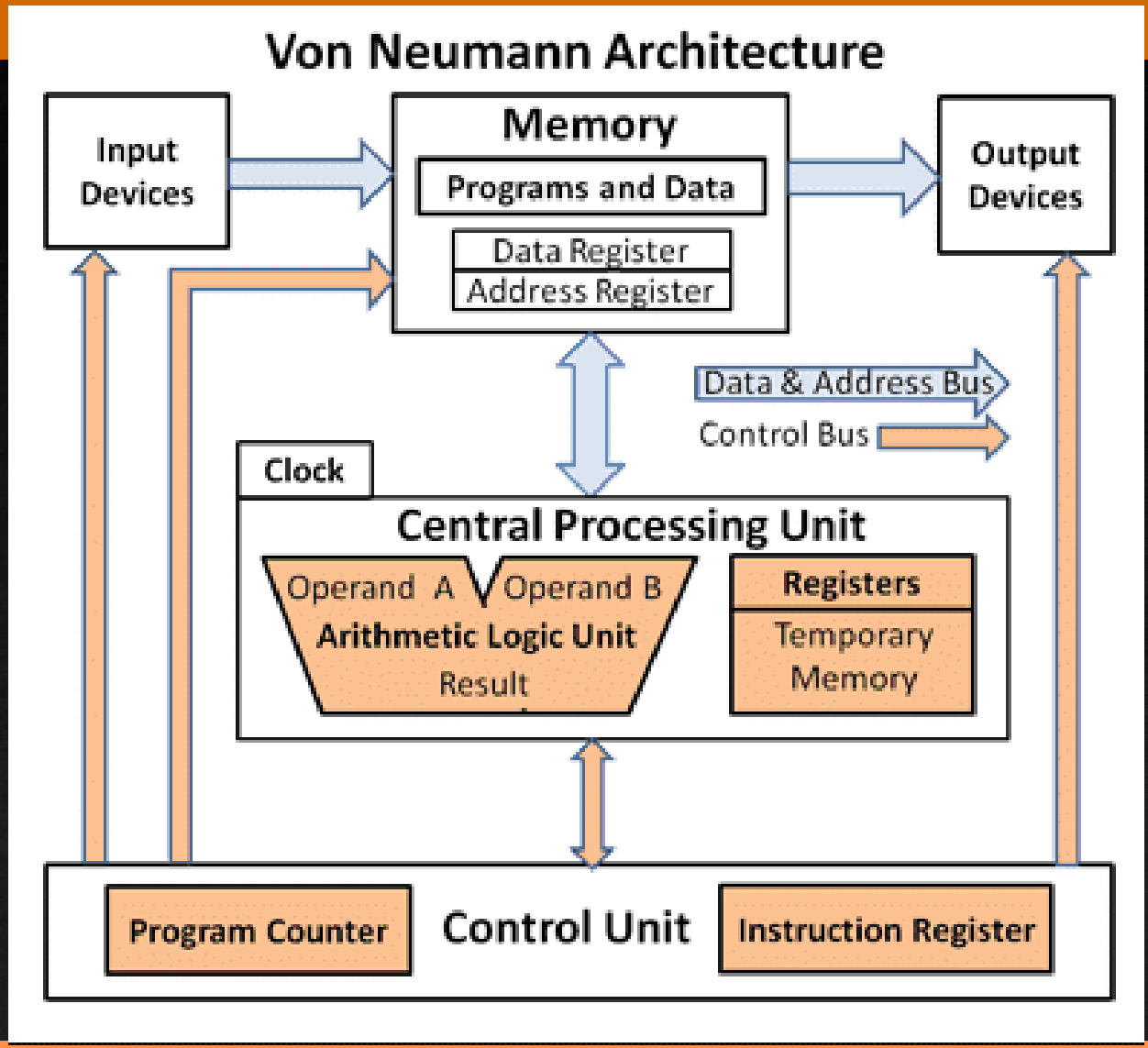
- Komputery są częścią naszego codziennego życia.
- Komputer = sprzęt + oprogramowanie
  - ✓ Sprzęt - komponenty fizyczne
  - ✓ Oprogramowanie - programy, procedury, które mówią komputerowi, co ma robić.
- GDZIE używamy komputerów?

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Architektura sprzętu komputerowego

- ✓ CPU = Mikroprocesor - "mózg" naszego komputera - wykonuje wszystkie operacje arytmetyczne i logiczne (ALU) i kontroluje wszystkie działania systemu
- ✓ Jednostka pamięci - przechowuje dane i programy
  - RAM – Pamięć o dostępie swobodnym
  - ROM – Pamięc tylko do odczytu
- ✓ Urządzenia wejścia/wyjścia



Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Co-funded by the Erasmus+ Programme of the European Union

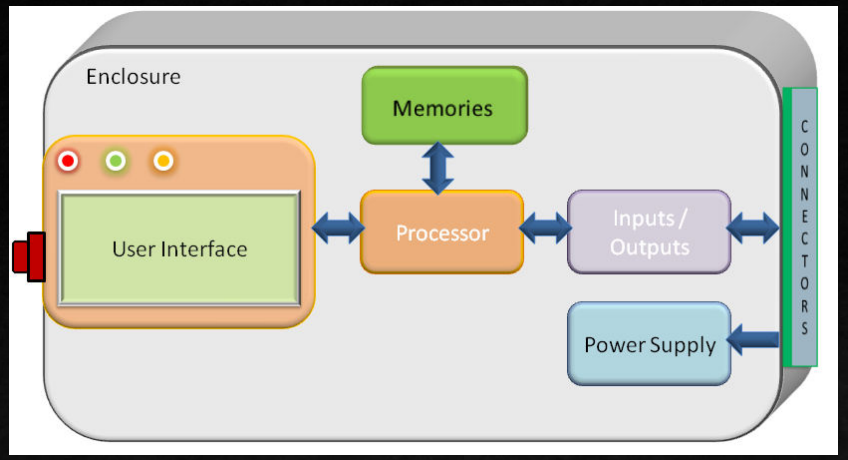
Systemy wbudowane w życiu codziennym

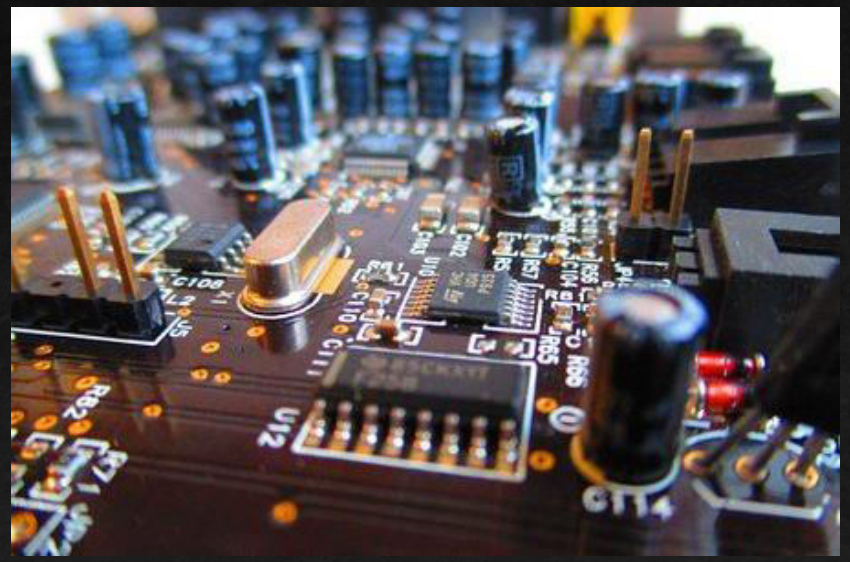
- System wbudowany to komputer specjalnego przeznaczenia, który jest używany wewnątrz urządzenia
- bazuje na mikrokontrolerze, który jest układem zoptymalizowanym do sterowania urządzeniami elektronicznymi; jest umieszczony w pojedynczym układzie scalonym, dedykowanym do wykonywania określonego zadania i realizacji jednej konkretnej aplikacji
- GDZIE stosujemy systemy wbudowane?

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

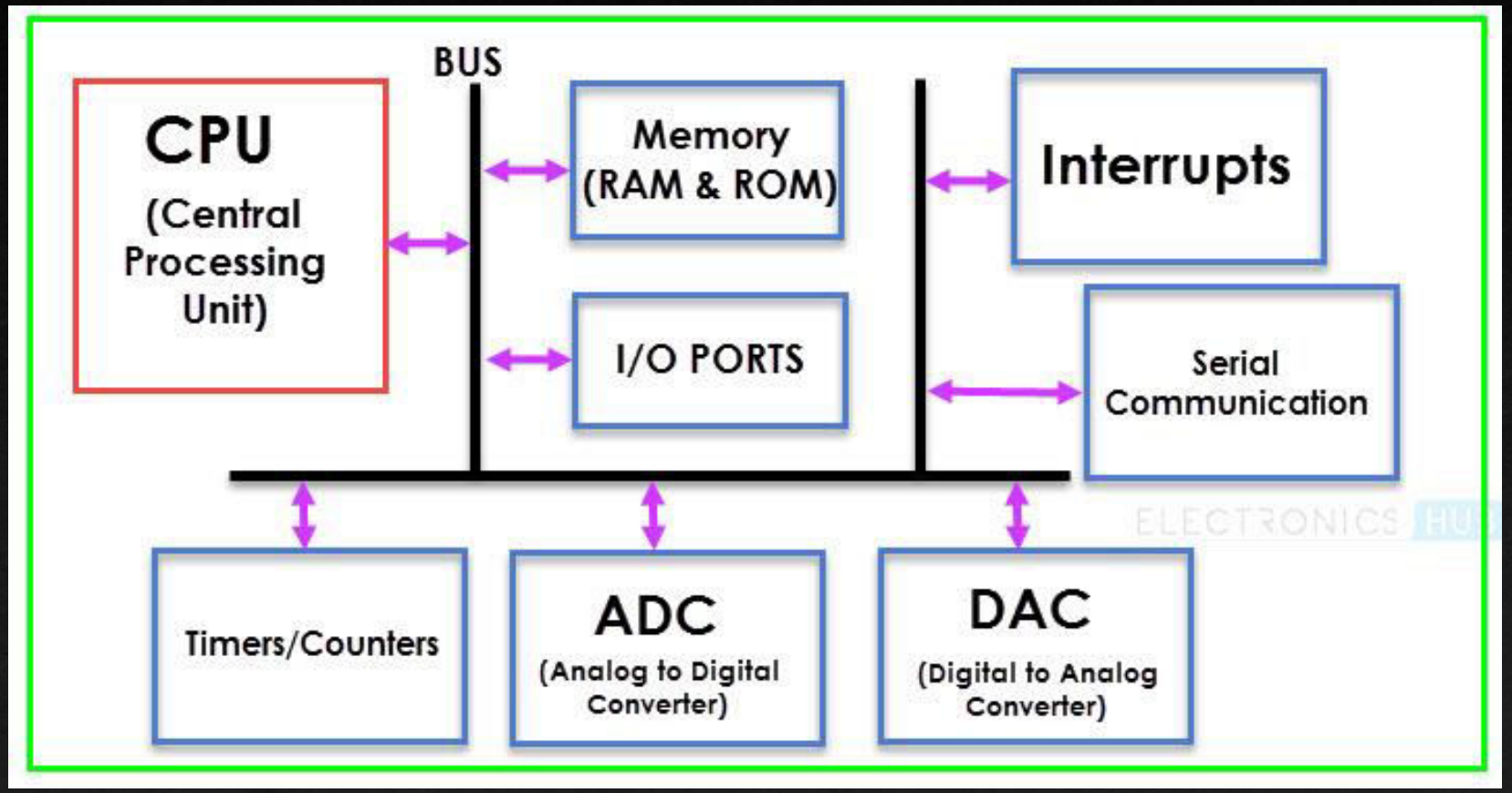
Co-funded by the Erasmus+ Programme of the European Union

Architektura systemów wbudowanych





- CPU - to mikrokontroler lub mikroprocesor



Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Który z nich jest lepszy?

Mikroprocesor	Mikrokontroler
jest sercem systemu komputerowego	serce systemu wbudowanego
jest to tylko procesor, więc pamięć i komponenty I/O muszą być podłączone zewnętrznie	posiada procesor wraz z pamięcią wewnętrzną i elementami wejścia/wyjścia
pamięć i I/O musi być podłączona zewnętrznie, więc obwód staje się duży	pamięć i I/O są już obecne, a obwód wewnętrzny jest mały
może być stosowany w systemach kompaktowych	stosowany jest w systemach kompaktowych
koszt całego systemu jest wysoki	niski koszt całego systemu
ze względu na zewnętrzne komponenty, całkowity pobór mocy jest wysoki. Nie jest to idealne rozwiązanie dla urządzeń zasilanych energią zmagazynowaną, takimi jak baterie	ponieważ ilość elementów zewnętrznych jest niewielka, całkowity pobór mocy jest mniejszy. Może być więc stosowany z urządzeniami zasilanymi energią zmagazynowaną, takimi jak baterie
większość z nich nie posiada funkcji oszczędzania energii	większość z nich oferuje tryb oszczędzania energii
stosowane głównie w komputerach osobistych	stosowany głównie w systemach wbudowanych
są oparte na modelu Von Neumanna	są oparte na architekturze harwardzkiej

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Który z nich jest lepszy?

Microprocessor	Microcontoller
mają mniejszą liczbę rejestrów, więc więcej operacji jest wykonywanych w pamięci	mają więcej rejestrów, więc programy są łatwiejsze do napisania
jest centralną jednostką obliczeniową na pojedynczym układzie scalonym opartym na silikonie	jest produktem ubocznym rozwoju mikroprocesorów z procesorem wraz z innymi urządzeniami peryferyjnymi
nie posiada pamięci RAM, ROM, jednostek wejścia-wyjścia, timerów i innych urządzeń peryferyjnych na chipie	posiada procesor wraz z pamięcią RAM, ROM i innymi urządzeniami peryferyjnymi osadzonymi na pojedynczym układzie scalonym
wykorzystuje zewnętrzną magistralę do interfejsu z pamięcią RAM, ROM i innymi urządzeniami peryferyjnymi	wykorzystuje wewnętrzną magistralę sterującą
Systemy oparte na mikroprocesorach mogą pracować z bardzo dużą prędkością ze względu na zastosowaną technologię	Systemy oparte na mikrokontrolerach pracują do 200MHz lub więcej w zależności od architektury.
jest używany w aplikacjach ogólnego przeznaczenia, które pozwalają na obsługę dużej ilości danych	jest używany dla systemów specyficznych dla aplikacji
Jest skomplikowany i kosztowny, z dużą liczbą instrukcji do przetworzenia	Jest to proste i niedrogie rozwiązanie z mniejszą liczbą instrukcji do przetworzenia.

Project No. 2018-1-RO01-KA203-063845
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

IF statement, declaration & call of void functions and const definition

Task: Program a device capable to read the state of a potentiometer (an analog input) and turns on an LED only if the potentiometer.
It must print the analog value regardless of the level.

```
const int analogPin = A0;
   // pin that the sensor is attached to
const int ledPin = 13;
   // pin that the LED is attached to
const int valmin = 400;
   // an arbitrary valmin level

void setup() {
   // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
   // initialize serial communications:
  Serial.begin(9600); }
```

```
void loop() {
  // read the value of the potentiometer:
  int analogValue = analogRead(analogPin);

  // if the analog value is high enough,
  // turn on the LED:
  if (analogValue > valmin) {
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
  // print the analog value:
  Serial.println(analogValue);
  delay(1);    // delay in between reads
}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera

Variable definition

```
<typeVar> nameVar [= value];
```

declares a variable of a specific type, witch will determin the size of the values, the length and type of memory representation

Usual types of variables used in Arduino apps

int = numeric type for variables/constants; it is represented on 4 bytes and can store values between aprox. $-2*10^9 \cdots 2*10^9$

bool = boolean type for variables/constants; it is represented in 1 byte and can store values false (0) and true (1)

Constant definition

```
const <typeConst> nameConst = value;
```

sets a constant of typeConst with specific value

Decision Structure

```
if (Condition) {
  instructions_A
   // do instructions_A if the
   // Condition is true
} else {
 instructions_B
  // do instructions_A if the
  // Condition is false
}
```

void Functions – declaration

```
void nameFunction(list of formal parameters)
{ declaration of local variables
  instructions
}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera

void Functions – declaration

```
void nameFunction(formal parameters)
{ declaration of local variables
  instructions
}
```

non-void Functions - declaration

```
resultType nameFunction(formal parameters)
{ declaration of local variables
  instructions
  return expression;  //
}
```

where formal parameters is a list of types and names of parameters used in function instructions

Call of a function

```
nameFunction(list of actual parameters)
```

- void functions - The call is an instruction
- non-void function – The call is an oprerand in expression with the same type as the resultType

! formal and actual parameters must have same type, number and must be in the same order

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera

Arduino specific functions

setup()

`void function` - called when a sketch starts, and will only run once, after each powerup or reset of the Arduino board similar with main(). Use it to initialize variables, pin modes, start using libraries, etc.

loop()

`void function` - loops consecutively, allowing Arduino program to change and respond
It is called after `setup()` function, which initializes and sets the initial values.

pinMode(pin, mode)

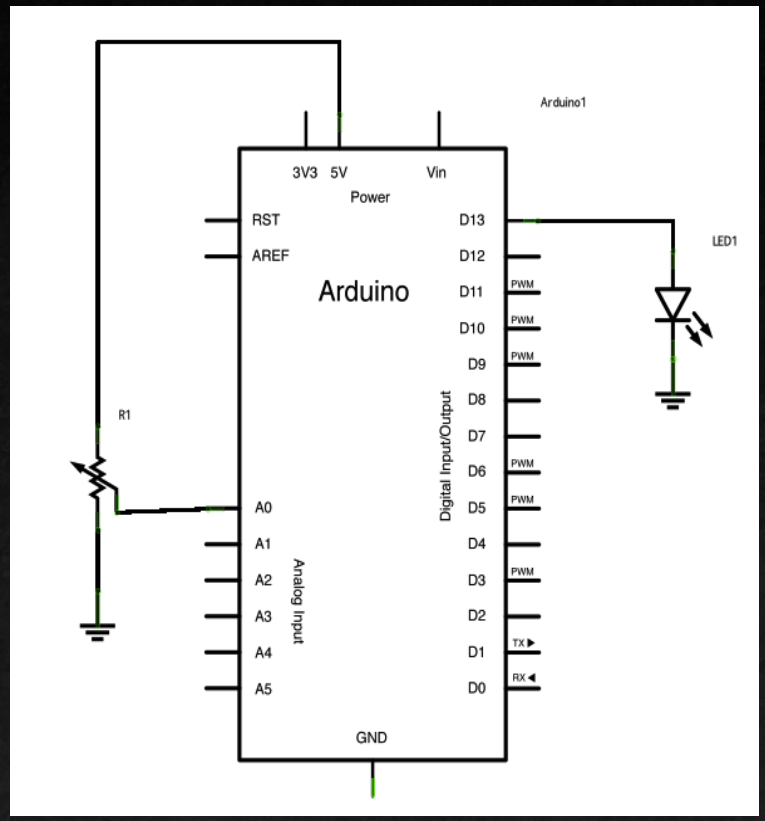
`void function with parametrers`

- `pin`: the Arduino pin number to set the mode off
- `mode`: INPUT, OUTPUT, or INPUT_PULLUP

delay(milisec)

`void function with parametrers`

- `milisec`: number of miliseconds to pause the program



electrical schema of the Arduino device

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Arduino specific functions

digitalWrite(pin, value)

void function with parametrers

- pin: the Arduino pin number.
- value: HIGH or LOW

digitalRead(pin)

function with parametrers

- pin: the Arduino pin num
- return value: HIGH or LOW

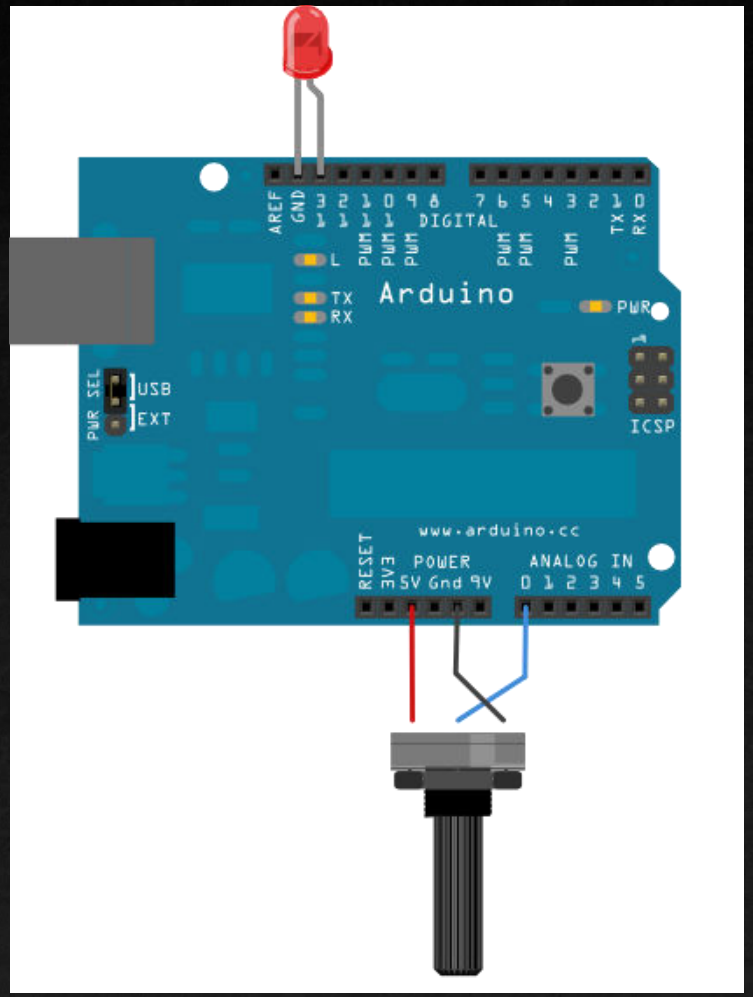
analogRead(pin)

int function with parametrer

- pin: the name of the analog input pin to read from (A0 to A5 on most boards)
- Returns the analog reading on the pin.

Arduino Pin Levels Constants HIGH and LOW

pin	INPUT	OUTPUT
HIGH	voltage > 3.0V	5V
LOW	voltage > 3.0V	0V



Arduino device

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

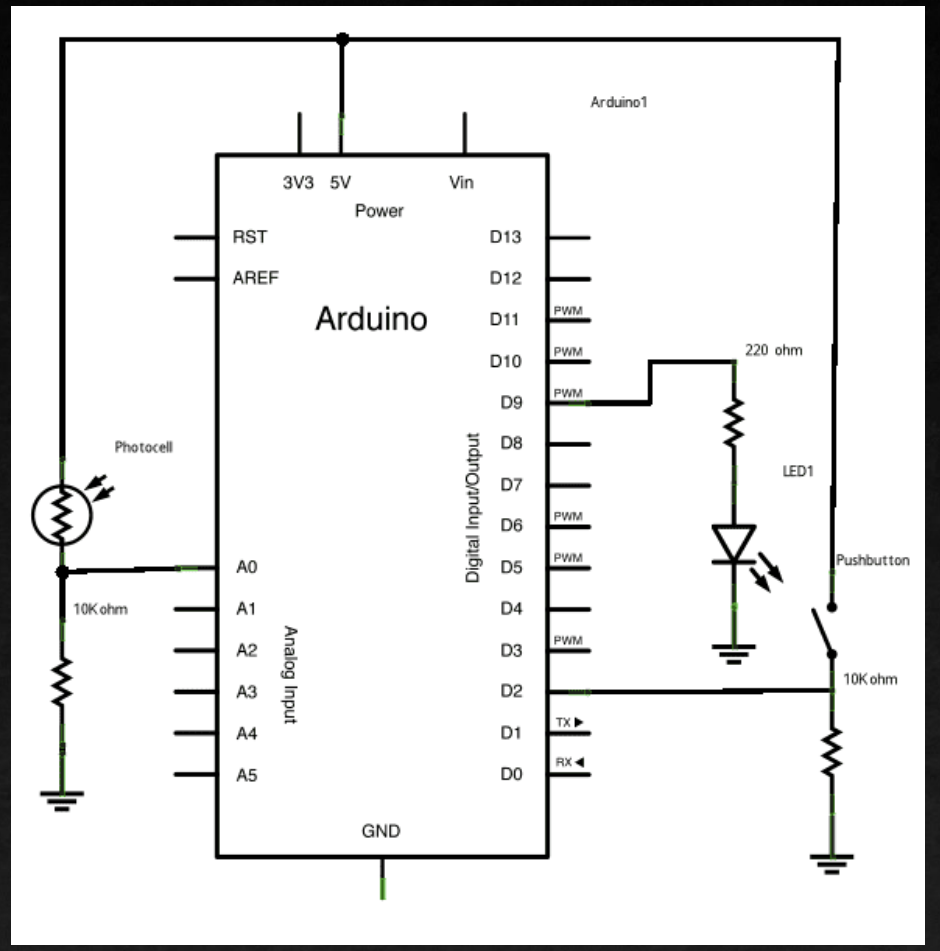
Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania za pomocą aplikacji Arduino

oświadczenie WHILE

Zadanie: Zaprogramować urządzenie zdolne do odczytywania przez pięć sekund danych wejściowych z czujnika oraz dokonać kalibracji poprzez określenie minimalnej i maksymalnej z oczekiwanych wartości dla odczytów dokonywanych podczas pętli.

```
const int sensorPin = A0;
    // pin that the sensor is attached to
const int ledPin = 9;
    // pin that the LED is attached to
    // variables:
int sensorValue = 0;
    // the sensor value
int sensorMin = 1023;
    // minimum sensor value
int sensorMax = 0;
    // maximum sensor value
```



Schema for Arduino device

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

oświadczenie WHILE

```
void setup() {
  // turn on LED to signal the start
  // of the calibration period:
     pinMode(13, OUTPUT);
     digitalWrite(13, HIGH);
  // calibrate during the first five seconds
  while (millis() < 5000) {
        sensorValue = analogRead(sensorPin);
        if (sensorValue > sensorMax){
            sensorMax = sensorValue;
        }
        if (sensorValue < sensorMin){
           sensorMin = sensorValue;
        }
    }
  digitalWrite(13, LOW);    // signal the end of the calibration period
}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera

Wyznaczanie wartości minimalnej i maksymalnej z zestawu wartości

Algorytm

Krok 1. Set the variables for sensorMin with the maximum value possible and sensor Max with the minimum value posibile

Krok 2. Compare current value with sensorMinand, if it is smaller, update sensorMin

krok 3. Compare current value with sensorMax and, if it is greater, update sensorMax

oświadczenie WHILE

```
while (Condition) {
  instructions_A
}
```

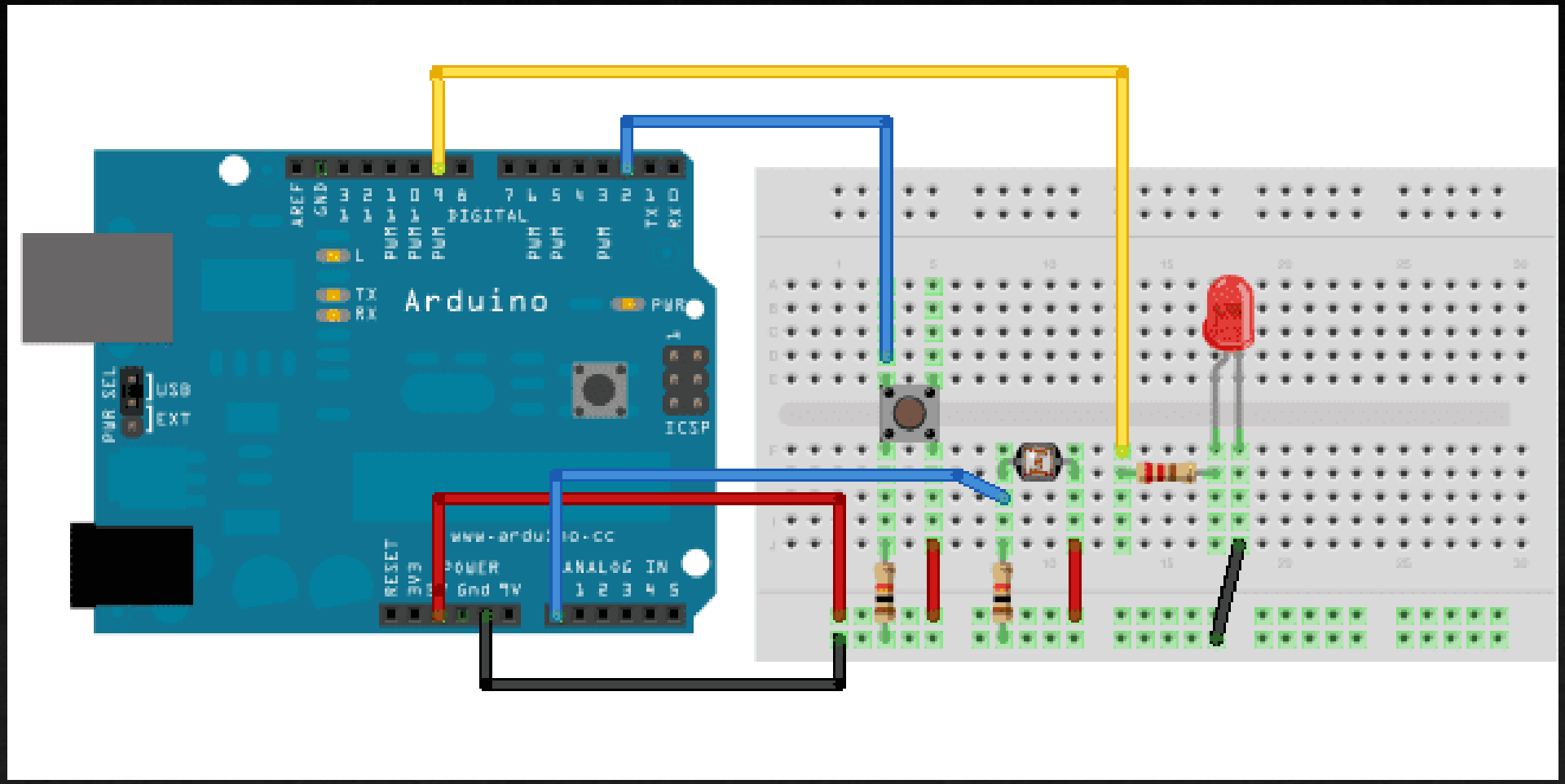
Execution

Krok 1. The Condition is evaluated
Krok 2. If the Condition is True
2.1. Instructions A will be executed
2.2. Go to Step 1.
If Condition is False, the program execution leave the loop

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera



Arduino device explaining WHILE

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Instrukcja FOR i manipulacja tablicami

```
int timer = 100;
   // The higher the number,
   // the slower the timing.
int ledPins[] = { 2, 7, 4, 6, 5, 3};
   // an array of pin numbers to which LEDs are attached
int pinCount = 6;
  // the number of pins (the length of the array)

void setup() {
  // the array elements are numbered from 0 to (pinCount-1)
  // use a for loop to initialize each pin as an output:
  for (int thisPin = 0; thisPin < pinCount; thisPin++) {
    pinMode(ledPins[thisPin], OUTPUT);
  }
}
```

Zadanie: Zaprogramować urządzenie zdolne do zapalania szereg diod LED dołączonych do pinów, których numery nie są ani sąsiadujące, ani niekoniecznie sekwencyjne. Aby to zrobić, numery pinów zapisz w tablicy ARRAY a następnie za pomocą pętli FOR iterować po tej tablicy.

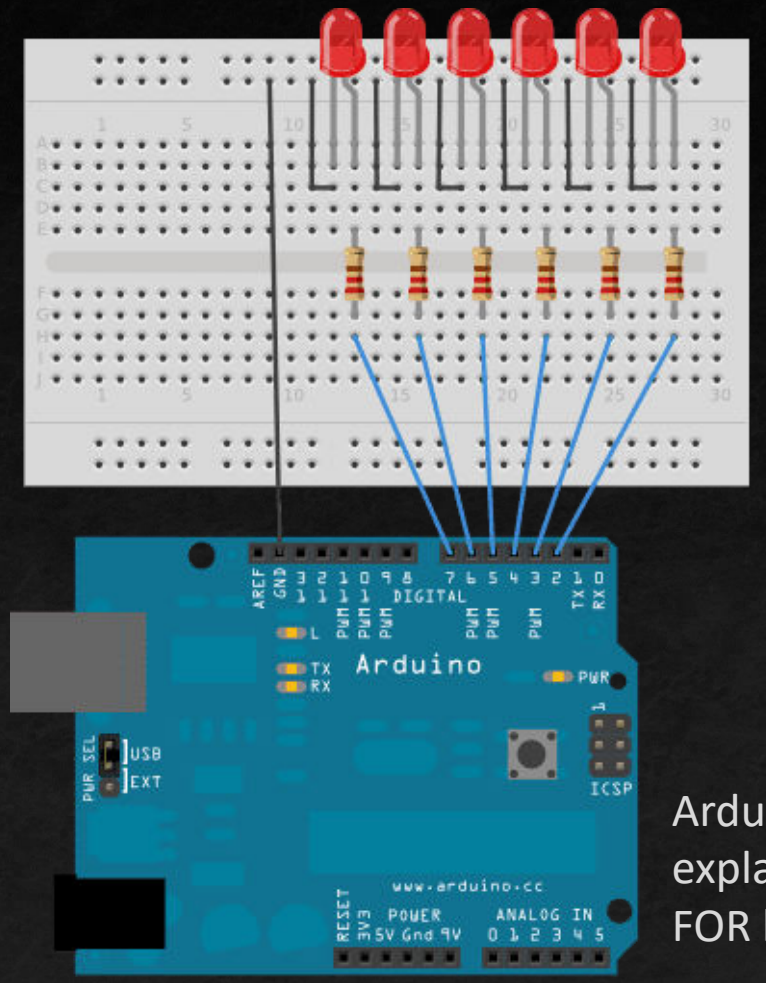
Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera

FOR statement and ARRAYs manipulation

```
void loop() {
// loop from the lowest pin to the highest:
 for (int thisPin = 0; thisPin < pinCount; thisPin++) {
   digitalWrite(ledPins[thisPin], HIGH);
      // turn the pin on:
   delay(timer);
   digitalWrite(ledPins[thisPin], LOW);
      // turn the pin off:
 }
      // loop from the highest pin to the lowest:
 for (int thisPin=pinCount-1; thisPin >= 0; thisPin--){
      // turn the pin on:
    digitalWrite(ledPins[thisPin], HIGH);
    delay(timer);
      // turn the pin off:
    digitalWrite(ledPins[thisPin], LOW);
 }
}
```

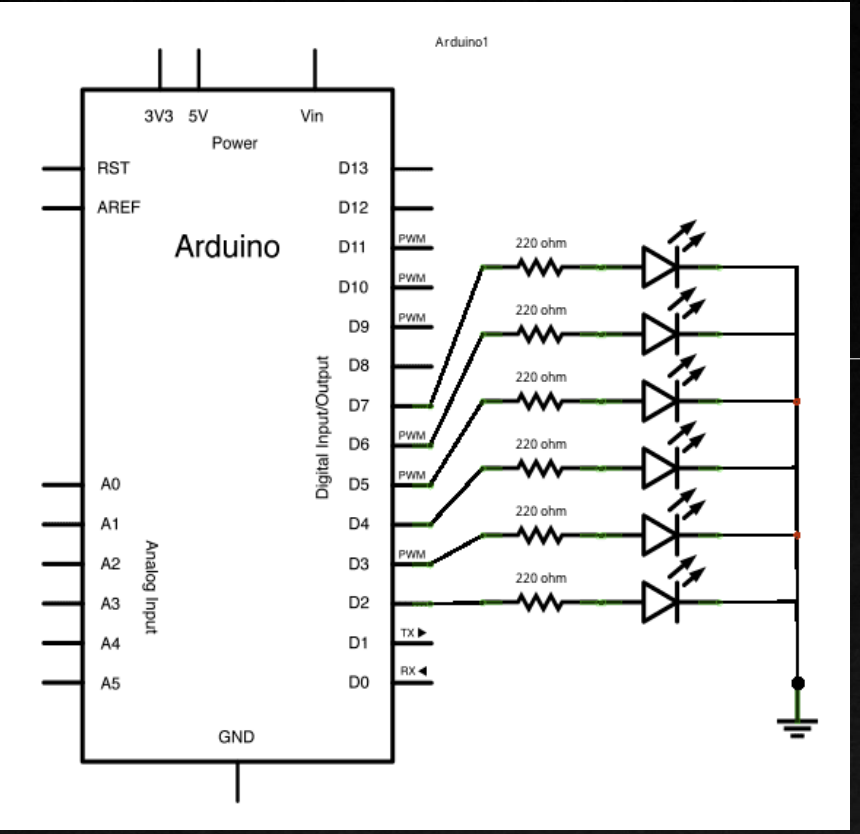


Arduino device explaining FOR loop

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera



Schemat dla urządzenia Arduino

FOR statement

```
for(int counter = initialVal; counter <= finalVal; counter++) {
     instructions_A
}
```

Execution

Step 1. The counter is set with initialValue

Step 2. If counter <= finaValue is True

    2.1. Instructions A will be executed

    2.2. counter is increased with 1

    2.3. Go to Step 2

Step 3. If counter <= finaValue is False, the program execution leave the loop

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Organizing data in ARRAYs

Array = a collection of data with the same type, organized in a contiguos memory zone and refered with a single name, witch is a pointer (memory adress) of the first element in the array.

Declaration:

```
valuesType arrayName[numberOfElements];
```

Initialization:

- along with the declaration

```
int ledPins[] = { 2, 7, 4, 6, 5, 3};
```

- by assignation

```
digitalWrite(ledPins[thisPin], HIGH);
```

- by reading values from input

Refering to a specific value from the array

A[expressionIndex]

expressionIndex is an integer from [0,count-1], indicating the position of the element in array

Parsing the ARRAY in order to analyze and process its elements:

left-rigt

```
for (int it = 0; it < count; it++)
    process(A[it]);
```

rigt-left

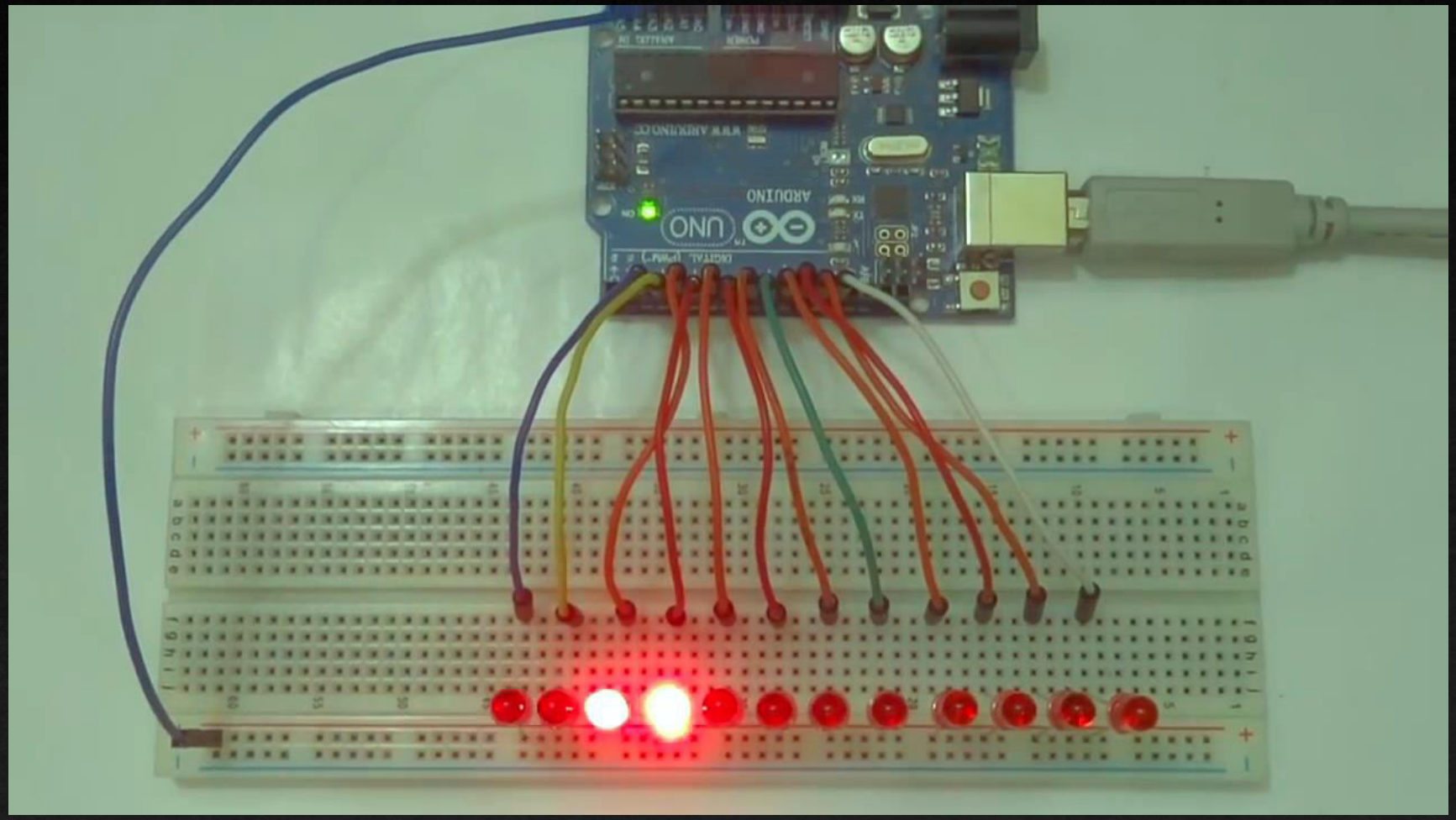
```
for (int it = count - 1; it >= 0; it--)
    process(A[it]);
```

where count is the number of array elements and it is an index used for parsing the array

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Zrozumienie koncepcji programowania na przykładzie aplikacji mikrokontrolera



Learning FOR loops and ARRAYs with Arduino device

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Objęte obszary naukowe

Architektura sprzętowa komputerów i systemów wbudowanych

Programowanie strukturalne - typy danych, statystyka (IF, WHILE, FOR), funkcje definiowane przez użytkownika

Programowanie urządzeń opartych na mikrokontrolerach (np. Arduino)

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Ocena

- test wielokrotnego wyboru
- mini-projekt w zespole 2-3 uczniów - programowanie urządzeń Arduino, które:

✓ opisać działanie innych specyficznych instrukcji w programowaniu strukturalnym

✓ do zastosowania w rzeczywistych sytuacjach - na przykład, działanie sygnalizacji świetlnej RGV

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Bibliografia

Webografia

- https://creativecommons.org/2008/10/22/wired-on-arduino-and-open-source-computing/
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/WhileStatementConditional
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ForLoopIteration
- https://commons.wikimedia.org/wiki/Category:Fritzing
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ForLoopIteration
- https://commons.wikimedia.org/wiki/Category:Arduino_projects

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union