

# CANduino – przygotowanie środowiska Arduino IDE do pracy oraz pierwszy program

Projekt PSIK

Roman Wawrzaszek

2016

# Plan zajęć

- Dystrybucja składników oprogramowania.
- Przygotowanie środowiska.
- Ustanowienie komunikacji z płytką Arduino Due.
- Wprowadzenie do programowania w środowisku Arduino.
- Kompilacja i uruchomienie programu.

# Dystrybucja składników oprogramowania.

1. Założyć folder CANduino
2. Do folderu CANduino skopiować:
  - Folder „arduino-1.6.7”
  - Folder „winusb”
  - Plik uruchomieniowy „zadig\_2.1.2.exe”

# Przygotowanie środowiska.

1. Wejść do folderu „arduino-1.6.7” i uruchomić program „arduino”.
2. W menu 'Narzędzia' wybieramy opcję "Arduino DUE Programming port". Ponieważ prawdopodobnie jej nie ma, realizujemy punkty 3-6.
3. W menu 'Narzędzia / Płytki' wybieramy „menedżer płytek”.
4. Wybieramy "Arduino SAM Boards (32-bits ARM Cortex-M3) by Arduino (dopisek: Płytki dołączone w tej paczce: 'Arduuino Due'
5. Wybieramy 'Instaluj,.
6. Jak się zainstaluje to zamykamy okienko instalacji dodatków i w menu 'Narzędzia / Płytki' szukamy opcji 'Arduino DUE Programming port'

# Ustanowienie komunikacji z płytką Arduino Due.

1. Wypakowujemy płytki oraz kabelek USB (niebieski).
2. Podłączamy ostrożnie wtyczkę micro USB (ta mniejsza) do złącza 'Programming,.
3. Podłączamy kabelek USB do gniazda USB komputera. System Windows wyszukuje sterownika ale nie będzie mógł go odnaleźć. Należy cierpliwie poczekać a następnie przejść do punktu 6. (Punkty 4 i 5 są alternatywne na wypadek problemów z instalacją)
4. Uruchomić program „zadig” i instalujemy „Arduino Due Port” \*).
5. Jeśli opcja ‚Port’ w menu ‚Narzędzia, jest nieaktywna (szara) należy zaktualizować sterownik portu Arduino poprzez manager urządzeń systemu Windows. Aktualizując sterownik należy wskazać katalog z plikami winusb. \*)
6. Po wykonaniu punktu 5 opcja Port w menu Narzędzia powinna być już aktywna.

\*) Kroki opcjonalne – w razie problemów.

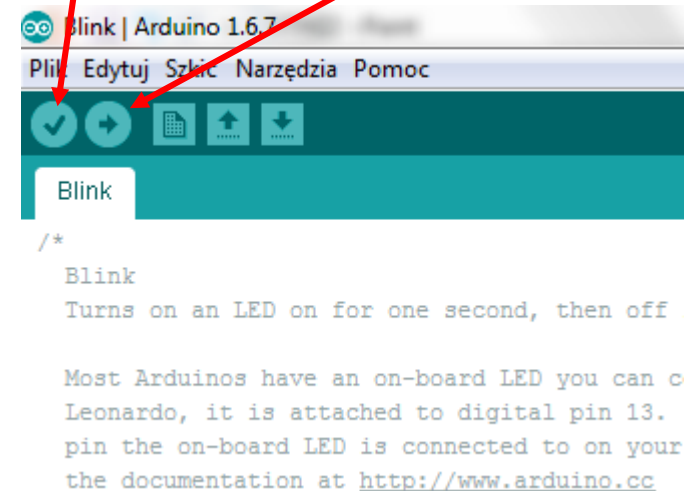
# Wprowadzenie do programowania w środowisku Arduino.

Pierwszy program – migająca dioda.

1. W górnym menu wybieramy: Plik/Przykłady/01.Basics/Blink.
2. Klikamy na przycisk kompilacji.
3. Klikamy na przycisk kompilacji i programowania.
4. Po chwili, dioda na płycie Arduino DUE zaczyna migać!
5. Umiemy już zaprogramować mikrokontroler! Tylko co w zasadzie zrobiliśmy??? ;)

Przycisk sprawdzenia i kompilacji kodu

Sprawdzenie, kompilacja oraz programowanie mikrokontrolera



# Wprowadzenie do programowania w środowisku Arduino.

## Procedura inicjująca – „setup”

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}
```

Informujemy procesor że jego ,noga' (port) nr 13 ma być wyjściem, czyli to procesor będzie ten port zasilat lub nie.

## Główna pętla programu

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);           // wait for a second
  digitalWrite(13, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);           // wait for a second
}
```

### Procesor:

- załącza napięcie do nogi 13,
- czeka 1000ms
- Odłącza nap. z nogi 13,
- Czeki 1000ms,
- Wraca na początek pętli głównej.

Wniosek: Mikrokontroler realizuje zadania zgodnie z zadanym algorytmem, czyli ,przepisem' który dokonuje konfiguracji linii wejścia/wyjścia (tzw. porty I/O – input/output). Procesor może być źródłem sygnału (linia OUTPUT) lub może ,słuchać' na danej linii reagując tak jak tego sobie życzy programista. Może zmienimy argument funkcji ,delay'?

# Wprowadzenie do programowania w środowisku Arduino.

## Jak rozmawiać z mikrokontrolerem?? (Program Dimmer 2)

```
const int ledPin = 13;          // the pin that the LED is attached to

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

void loop() {
  byte brightness;

  // check if data has been sent from the computer:
  if (Serial.available()) {
    // read the most recent byte (which will be from 0 to 255)
    brightness = Serial.read();
    if (brightness >= '0' && brightness <= '9') // ignore non-digits
    {
      brightness = 25 * (brightness - '0'); // scale to 0, 25, 50, 75, etc.
      // set the brightness of the LED
      analogWrite(ledPin, brightness);
    }
  }
}
```

Uruchomić program Dimmer2 oraz terminal do komunikacji poprzez port RS z prędkością 9600.