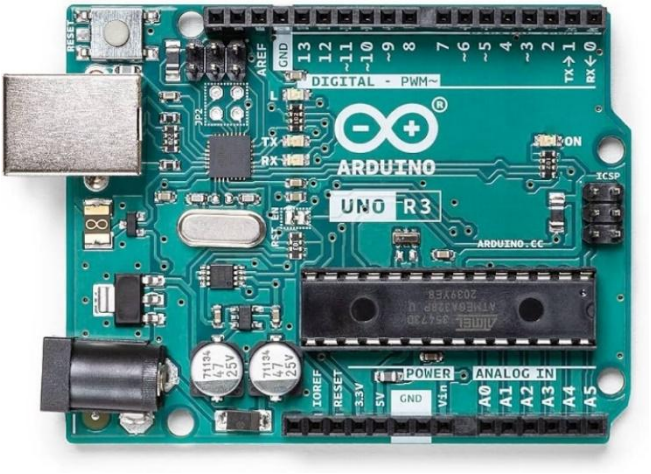
Materiał dodatkowy do ćwiczeń 2, 4, 6 i 8

Podstawy budowy i programowania Arduino oraz budowania układów elektronicznych (na przykładzie Arduino Uno)



Arduino umożliwia wykonywanie kodu na odpowiednio przygotowanych obwodach, obsługiwanych przez piny wejścia/wyjścia na płytce. Z naszego punktu widzenia uwagę należy zwrócić na:

- Piny zasilania: 5V, 3.3V, GND – pozwalają one w ogóle doprowadzić prąd do obwodów
- Cyfrowe piny wejścia/wyjścia (oznaczone od 0–13)* – odczytują lub przypisują wartości cyfrowe – stan niski (LOW) lub wysoki (HIGH), przy czym:
  - Piny 0 i 1 oznaczone jako Tx i Rx mogą służyć również jako interfejs UART*
  - Piny oznaczone znakiem ~ (3, 5, 6, 9, 10, 11)* obsługują technologię PWM – dzięki pewnej symulacji działają jak analogowe piny wyjścia (pozwalają przypisać wartości z zakresu 0–255*
- Analogowe piny wejścia – pozwalają odczytać wartości analogowe – odczytanym napięciom przypisane zostają wartości za zakresu 0–1023*

Programowanie Arduino jest w środowisku Arduino IDE w języku o składni zbliżonej do języka C++. Przyjrzyjmy się kilku cechom języka Arduino:

Program składa się z dwóch głównych bloków funkcyjnych:

```
void setup() {

}

void loop() {

}
```

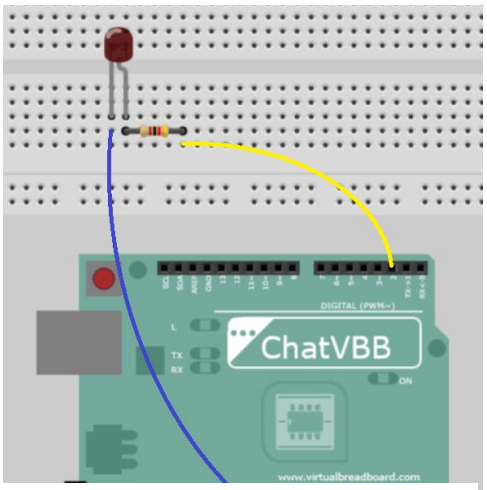
Blok `setup()` wykonywany jest raz na początku działania programu – w tym bloku są np. definiowane kierunki działania pinów (instrukcja `pinMode`). Blok `loop()` to pętla nieskończona, w której cyklicznie wykonywany jest właściwy program. Makra, zmienne, stałe mogą być definiowane przed blokiem `setup()`.

Prześledźmy, jak wygląda przykładowy kod, obsługujący prosty obwód:

<u>Na co zwrócić uwagę:</u>

Dioda LED jest elementem <u>asymetrycznym</u> – dłuższa nóżka jest anodą, krótsza – katodą.

Na schematach poszczególne elementy podłączone są poglądowo. Piny Arduino mogą być dobrane dowolnie – należy jedynie przypisać właściwe piny do odpowiednich zmiennych w kodzie



Anodę podłączamy za pośrednictwem rezystora 220Ω do pinu cyfrowego 2, katodę do GND

Użyte komponenty:

- Dioda LED
- Rezystor 220Ω

Przykładowy kod:

```
const int ledPin = 2;
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

1. Definiujemy stałą (`const`) typu integer (`int`) `ledPin`, przypisując jej wartość `2`. Jest to stała przechowująca informację o pinie, do którego podłączona jest dioda LED (w tym przypadku jest to pin cyfrowy 2, zgodnie ze schematem konstrukcyjnym)
2. W bloku `setup` przy pomocy funkcji `pinMode` ustalamy, że pin `ledPin` (2) będzie działał jako urządzenie wyjścia (`OUTPUT`) – analogicznie urządzenia wejścia (przyciski, czujniki, itp.) będziemy definiowali jako `INPUT`
3. W bloku `loop` najpierw przypisujemy do pinu `ledPin` wartość `HIGH` (funkcja `digitalWrite` przypisuje wartości `LOW` i `HIGH`), a następnie po odczekaniu 1s (funkcja `delay` ustawiona na 1000ms) przypisujemy wartość `LOW`. Analogicznie będą działać funkcje `digitalRead` (odczytanie wartości cyfrowej), `analogWrite` i `analogRead`. Pamiętać należy jednak, że odczyt analogowy `analogRead` możliwy jest wyłącznie na pinach analogowych (numer poprzedzony literą A), zaś zapis analogowy `analogWrite` wyłącznie na pinach PWM. Funkcja `analogRead` będzie zwracała wartości z przedziału 0–1023, zaś w funkcji `analogWrite` przypisać możemy wartości od 0 do 255.

Krótka ściąga:

Instrukcja	Wartości	Piny
digitalWrite	LOW/HIGH	Dopuszczalne wszystkie (cyfrowe i analogowe)
digitalRead	LOW/HIGH	Dopuszczalne wszystkie (cyfrowe i analogowe)
analogWrite	0–255	Cyfrowe PWM (3, 5, 6, 9, 10, 11)*
analogRead	0–1023	Analogowe (A0–A5)*

* wszystkie charakterystyki pinów podane są dla Arduino Uno

Kilka innych, przydatnych konstrukcji

Instrukcja warunkowa `if`:

```
if (warunek == wartość) {

}
```

Pętla `for`:

```
for (int i=0; i<10; i++)
{

}
```

Nieskończona pętla `while`:

```
while (true) {

  if (warunek) break;
}
```

spełnienie warunku przerywa pętlę