MoboLab – roboty i tablety w Twojej szkole
Obszar II. „Stwórz własnego robota”
Scenariusze lekcji i zajęć pozalekcyjnych

SCENARIUSZ 15. BUZZER – PIEZOELEKTRYCZNY GŁOŚNICZEK

scenariusz zajęć pozalekcyjnych

autor: Michał Podziomek

redakcja: Agnieszka Koszowska

SŁOWA KLUCZOWE:

Arduino, programowanie, elektronika, buzzer, dźwięk

KRÓTKI OPIS ZAJĘĆ:

Podczas zajęć uczniowie i uczennice poznają i/lub utrwalają wiedzę o mikrokontrolerze Arduino. Przygotowują prosty układ z wykorzystaniem **buzzera** (brzęczyka). Poznają i/lub utrwalają podstawowe pojęcia programistyczne (skrypt, program, algorytm, sterowanie, warunek, pętla, wyrażenie "if... else"). Korzystając z wyrażenia warunkowego **if** tworzą program pozwalający na sterowanie dźwiękiem buzzera.

WIEDZA I UMIEJĘTNOŚCI ZDOBYTE PRZEZ UCZNIA / UCZENNICĘ:

- wie, czym są mikrokontrolery i do czego służą,
- zna pojęcia: mikrokontroler, skrypt, program, algorytm, sterowanie, warunek, pętla,
- zna projekt Arduino, wie, czym jest platforma Arduino, z jakich części się składa,
- potrafi w podstawowym stopniu samodzielnie obsługiwać Arduino (podłączyć płytkę do komputera, wgrać prosty program),
- Uczeń potrafi podłączyć buzzer i przełączniki do Arduino,
- zna podstawowe elementy interfejsu środowiska programistycznego Arduino IDE i podstawowe komendy języka Arduino IDE: pinMode(), digitalWrite(), delay(),
- rozumie zasadę działania funkcji digitalWrite() i potrafi wykorzystać ją w praktyce,

- zna podstawowe elementy języka **Scratch**, potrafi stworzyć prosty skrypt w tym języku.

GRUPA DOCELOWA:

Starsze klasy szkoły podstawowej (VII-) i klasy gimnazjalne (po dostosowaniu: możliwość realizacji w młodszych klasach: I-III i IV-VI szkoły podstawowej).
W młodszych klasach – możliwość wykorzystania programu mBlock (po przejściu scenariusza nr 18. *Programowanie Arduino z wykorzystaniem programu mBlock*) lub Scratch for Arduino (po przejściu scenariusza nr 1. *Wprowadzenie do Arduino*).

LICZBA UCZNIÓW/UCZENNIC W GRUPIE:

Liczba optymalna: 12, liczba maksymalna: 16

CZAS TRWANIA ZAJĘĆ:

90 min (lub 2 x 45 minut)

STOPIEŃ TRUDNOŚCI/SKOMPLIKOWANIA (w skali od 1 do 5 dla obszaru II. „Stwórz własnego robota”):

1

POTRZEBNY SPRZĘT I OPROGRAMOWANIE:

- komputer (przenośny lub stacjonarny),
- program Arduino IDE (do pobrania ze strony: http://www.arduino.org/downloads),
- (opcjonalnie) program mBlock (do pobrania ze strony: http://www.mblock.cc/download/) lub Scratch for Arduino (do pobrania ze strony: http://s4a.cat/),
- płytka Arduino UNO i kabel USB A-B (dla każdego uczestnika lub dla pary uczestników),
- płytka stykowa,
- oporniki 220 omów,
- przewody połączeniowe,
- projektor i laptop (w części teoretycznej).
- oprogramowanie mBlock (w razie potrzeby),
- brzęczyk piezoelektryczny,
- 3 włączniki,
- płytka prototypowa,
- przewody połączeniowe do płytki prototypowej.

CO NALEŻY PRZYGOTOWAĆ PRZED ZAJĘCIAMI:

- zainstalować program Arduino IDE,
- (opcjonalnie): zainstalować program **mBlock** lub **Scratch for Arduino**,
- sprawdzić, czy wszystkie komputery wykrywają podłączone Arduino,
- przeczytać dokładnie scenariusz,
- zapoznać się z materiałami dodatkowymi (w części „Pigułka wiedzy i inspiracji"),
- wykonać samodzielnie zadania zawarte w scenariuszu,
- przy każdym stanowisku komputerowym rozłożyć elementy zestawu Arduino, które będą wykorzystywane na tych zajęciach,
- dopasować stopień trudności zadania do potrzeb i możliwości klasy, dla której organizowane są zajęcia według wskazówek zawartych w scenariuszu.

KOMPETENCJE OSOBY PROWADZĄCEJ:

- wie, czym jest projekt Arduino, zna podstawowe informacje o projekcie,
- potrafi przynajmniej w stopniu podstawowym obsługiwać Arduino,
- zna podstawowe pojęcia z zakresu elektroniki,
- zna podstawowe pojęcia programistyczne,
- wie, dlaczego warto uczyć się programowania i jakie korzyści daje posiadanie umiejętności programistycznych,
- potrafi zachęcić do nauki programowania zarówno chłopców, jak i dziewczynki.

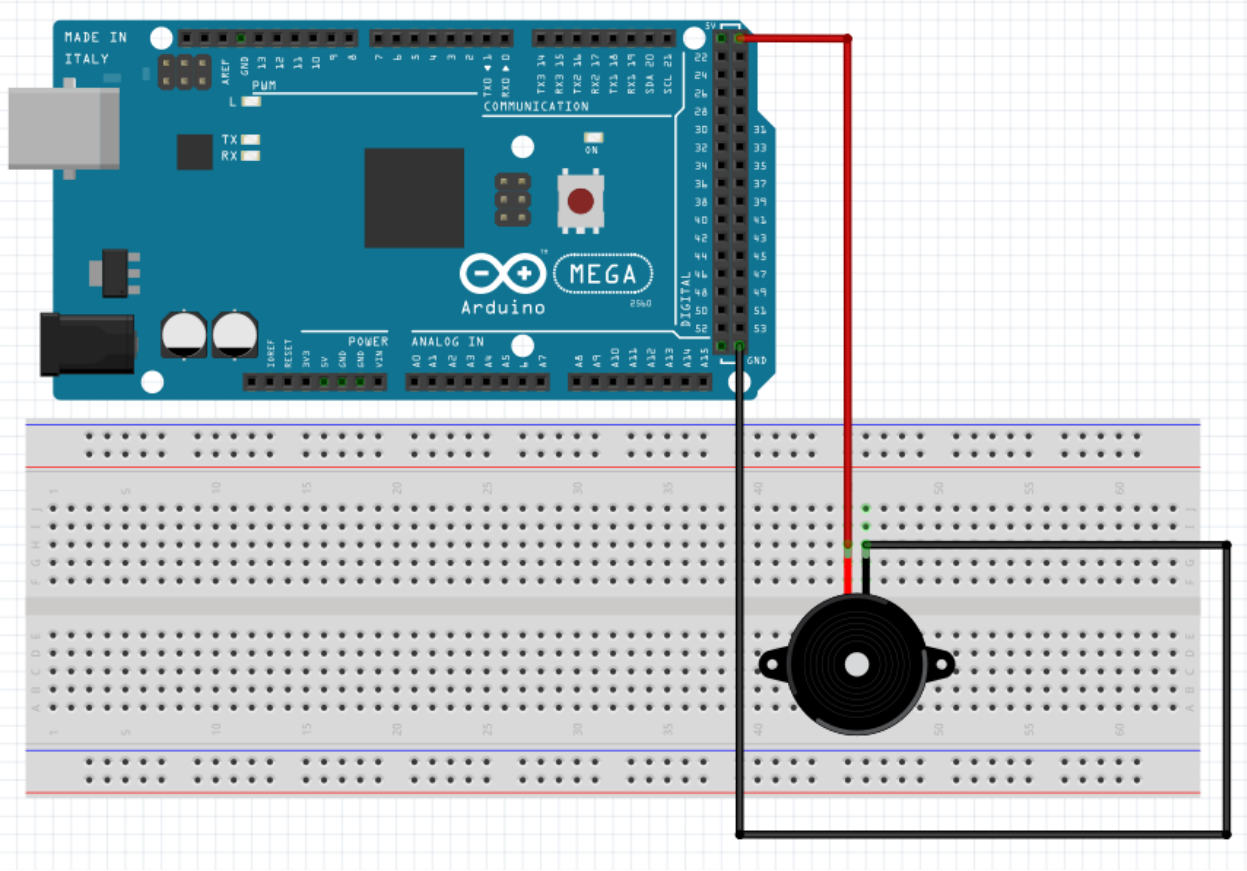
PRZEBIEG ZAJĘĆ:

Podłączenie Arduino, uruchomienie programu Arduino IDE i przypomnienie podstawowych informacji – ok. 15 minut

Uwaga! Informacje o tym, jak podłączyć Arduino, uruchomić program Arduino IDE i Scratch for Arduino, a także podstawowe informacje niezbędne przy rozpoczynaniu pracy z Arduino zawierają scenariusze 1 i 2. Tę część zajęć warto powtarzać za każdym razem w takim zakresie, jaki jest potrzebny, do czasu aż podstawowy materiał zostanie utrwalony.

Podłączamy buzzer, czyli element wydający dźwięk (brzęczyk) – 15 minut

Informacje o tym, jak podłączyć brzęczyk, znajdują się w scenariuszu nr 14 pt. *Instrument muzyczny- granie na przyciskach*. Buzzer podłączamy tak, jak zostało to pokazane na następującym schemacie:

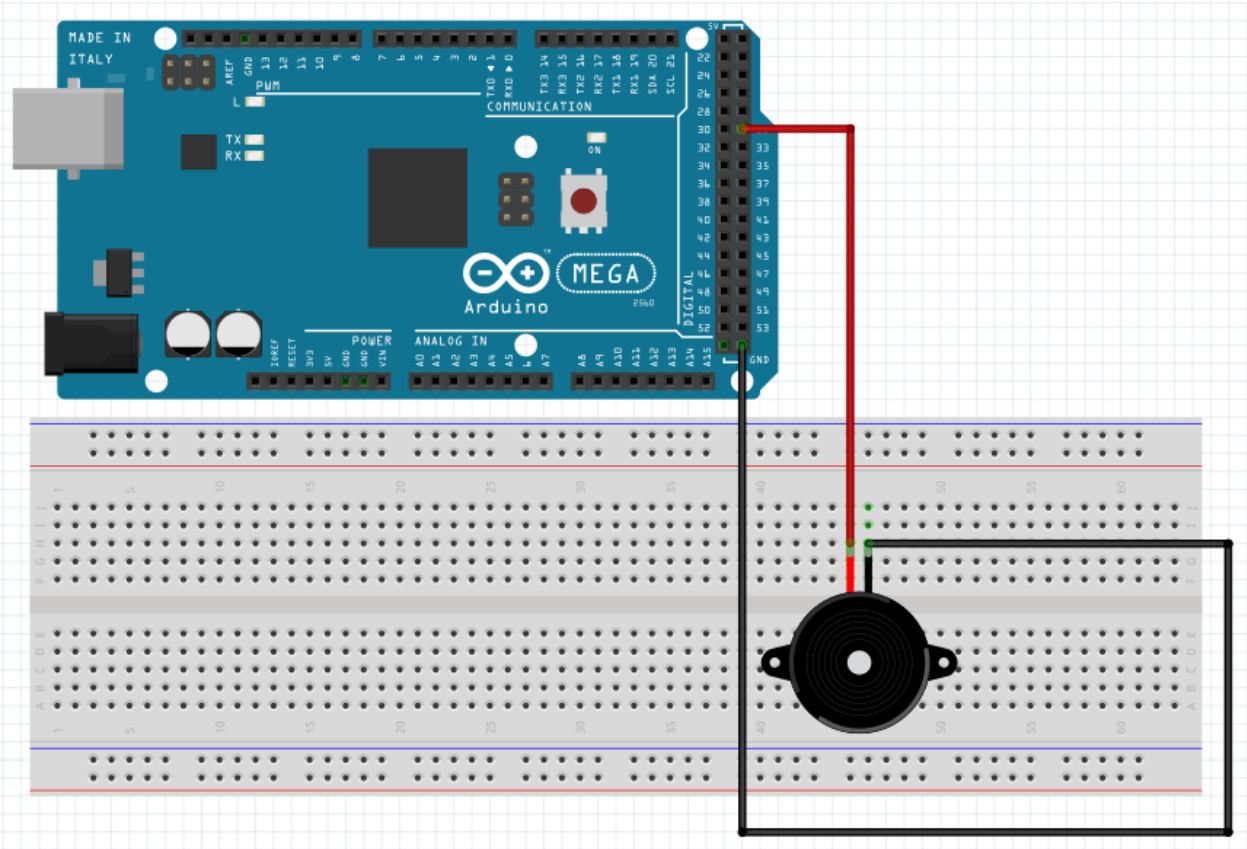


Podłączamy buzzer pod pin sygnałowy i piszemy program do grania melodii – 15 minut

W zadaniach z poprzednich zajęć używaliśmy płytki Arduino jako zwykłego źródła prądu. Prąd przepływał z niej przez brzęczyk, a brzęczyk grał, kiedy przewody były połączone.

Płytka Arduino, a konkretnie procesor, który się na niej znajduje (w przypadku Arduino Mega jest on oznaczony symbolem ATMEGA2560 oraz logiem producenta ATMEL) może działać jako bardzo szybki włącznik i wyłącznik. Jego faktyczna

szybkość włączania i wyłączania jest zależna od prędkości, z jaką działa kod, który załadujemy na naszą płytkę. Zmieniamy schemat w sposób następujący:



Teraz czerwony przewód, który poprzednio był na pinie PWR (+5V), jest podłączony pod pin typu DIGITAL, czyli cyfrowy. Jest to pin, który może być włączany i wyłączany za pomocą procesora na płytce. Teraz możemy więc napisać program, który wykorzysta tę cechę. Tworzymy w programie Arduino nowy szkic i wpisujemy poniższy program, ze szczególną ostrożnością sprawdzając, czy nie popełniliśmy błędu:

```
void setup(){
  // najpierw musimy powiedzieć Arduino, którego pinu będziemy używali do kontrolowania buzzera
  // oraz że pin ten będzie wyjściem (OUTPUT), czyli dawał prąd na zawołanie.
  pinMode(8, OUTPUT);
}
void loop(){
```

```
  // teraz każemy procesorowi włączyć pin 8 - czyli ustawić go na pozycję HIGH
  digitalWrite(8, HIGH);
  // następnie zaczekamy sekundę
  delay(1000);
  // i każemy wyłączyć pin 8 - czyli ustawić na pozycję LOW
  digitalWrite(8, LOW);
  // i zaczekamy kolejną sekundę
  delay(1000);
}
```

Teraz ładujemy kod na Arduino. Jeżeli wykonaliśmy wszystko poprawnie, buzzer powinien zacząć wydawać trwające sekundę dźwięki z również trwającymi sekundę przerwami.

Zadania dla uczniów:

- *Co się stanie, jeśli usuniemy ostatnią pozycję z cześci void loop() programu, czyli delay(1000)?*
- *Co trzeba zrobić, by dźwięk był wygrywany szybciej, wolniej, z dłuższymi lub krótszymi przerwami?*
- *Czy potrafisz napisać program, który zagra inny rytm?*

Jeżeli moduły początkowe zostały wykonane przez grupę w założonym czasie, w tym miejscu następuje koniec tego modułu szkoleniowego. Jeżeli grupa dała radę wykonać je szybciej, możemy przejść do dalszego etapu.

W tym miejscu możliwy jest podział zajęć na dwie części (kolejna część scenariusza będzie realizowana na następnych zajęciach).

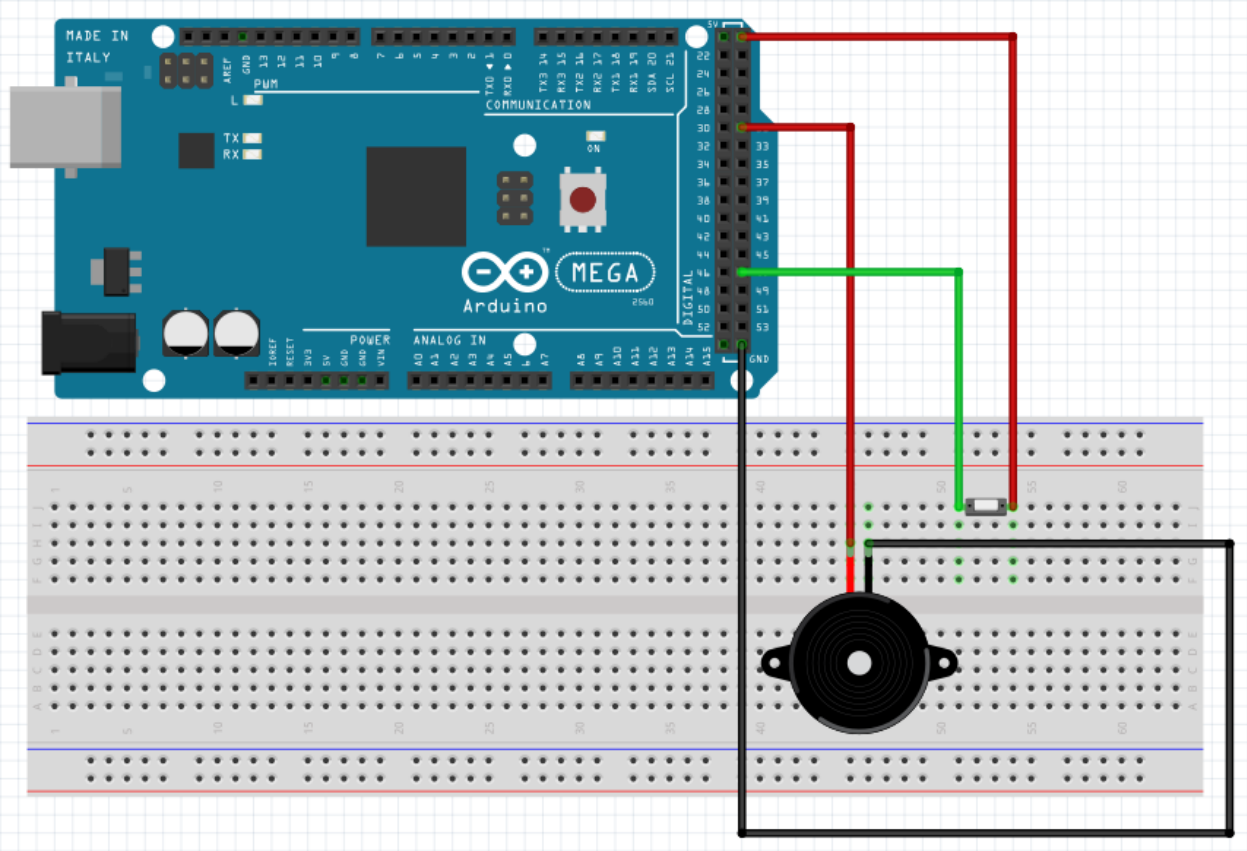
Przypomnienie materiału, odtworzenie układu – 10 minut

Rozpoczynamy od krótkiego przypomnienia materiału z poprzedniej części zajęć i odtworzenia układu zbudowanego na poprzednich zajęciach.

Dalsze zadania – 35 minut

Dodajemy do układu przełącznik.

Użyjemy przełącznika do uruchomienia części programu. Przełącznik będzie podłączony do osobnego pina i jego stan (włączony/wyłączony) będzie odczytywany przez Arduino. Podłączamy Arduino tak jak na poniższym obrazku, gdzie przełącznik jest podłączony do pinu 2:



Następnie otwieramy nowy projekt i przygotowujemy się do modyfikowania kodu. Będziemy w nim robić dwie nowe rzeczy:

Zmienne, czyli symboliczne słowa, do których przypisujemy wartość w programie, która się zmienia. Np. w naszym przypadku będzie się zmieniał stan przycisku - raz będzie włączony, raz wyłączony.

Żeby zdefiniować zmienną, musimy napisać jaki jest jej typ. Na początek użyjemy zmiennej która jest liczbą całkowitą, po angielsku integer, w skrócie – „int”:

int stanPrzełącznika = 0;

Ten przykład definiuje (czyli określa) zmienną i nadaje jej wartość początkową 0 za pomocą znaku „=”. Pojedynczy znak „=” zmienia wartości. Jeżeli chcemy sprawdzić czy jakać wartość jest równa innej, używamy PODWÓJNEGO znaku równości, czyli „==”.

Instrukcje warunkowe polegają na tym, że jeżeli jakiś określony warunek zostanie spełniony, program wykona operację. Np. jeżeli nasza zmienna **stanPrzełącznika** jest równa **HIGH**, czyli na pin płynie prąd, będziemy chcieli wykonać jakąś operację. Po angielsku **„jeżeli”** to **„if”**.

Zwracamy uwagę na podwójny znak równości - on jest porównaniem dwóch wartości. Pojedynczy używamy do ich zmieniania! (wtedy wartość po lewej zostanie zmieniona na wartość po prawej stronie znaku).

```
if (stanPrzełącznika == HIGH) {
  // tu opisujemy co ma zostać wykonane
}
```

Nasz kod powinien wyglądać w sposób następujący:

```
void setup(){
  // najpierw musimy powiedzieć Arduino, którego pinu będziemy używali do kontrolowania buzzera
  // oraz że pin ten będzie wyjściem (OUTPUT), czyli dawał prąd wtedy, kiedy mu każemy.
  pinMode(8, OUTPUT);

  // następnie mówimy który pin będzie odczytywał stan przycisku (włączony, wyłączony),
  // oraz że pin ten będzie wejściem (INPUT), czyli odbierał sygnał.
  pinMode (2, INPUT);
}

void loop(){
  // dodamy teraz zmienną - czyli symboliczne słowo, do którego będziemy przypisywali wartość
  // za każdym razem jak zacznie się pętla. Wartość będzie brana z pinu przycisku.
  // Żeby odczytać wartość z pina na którym jest przycisk,
```

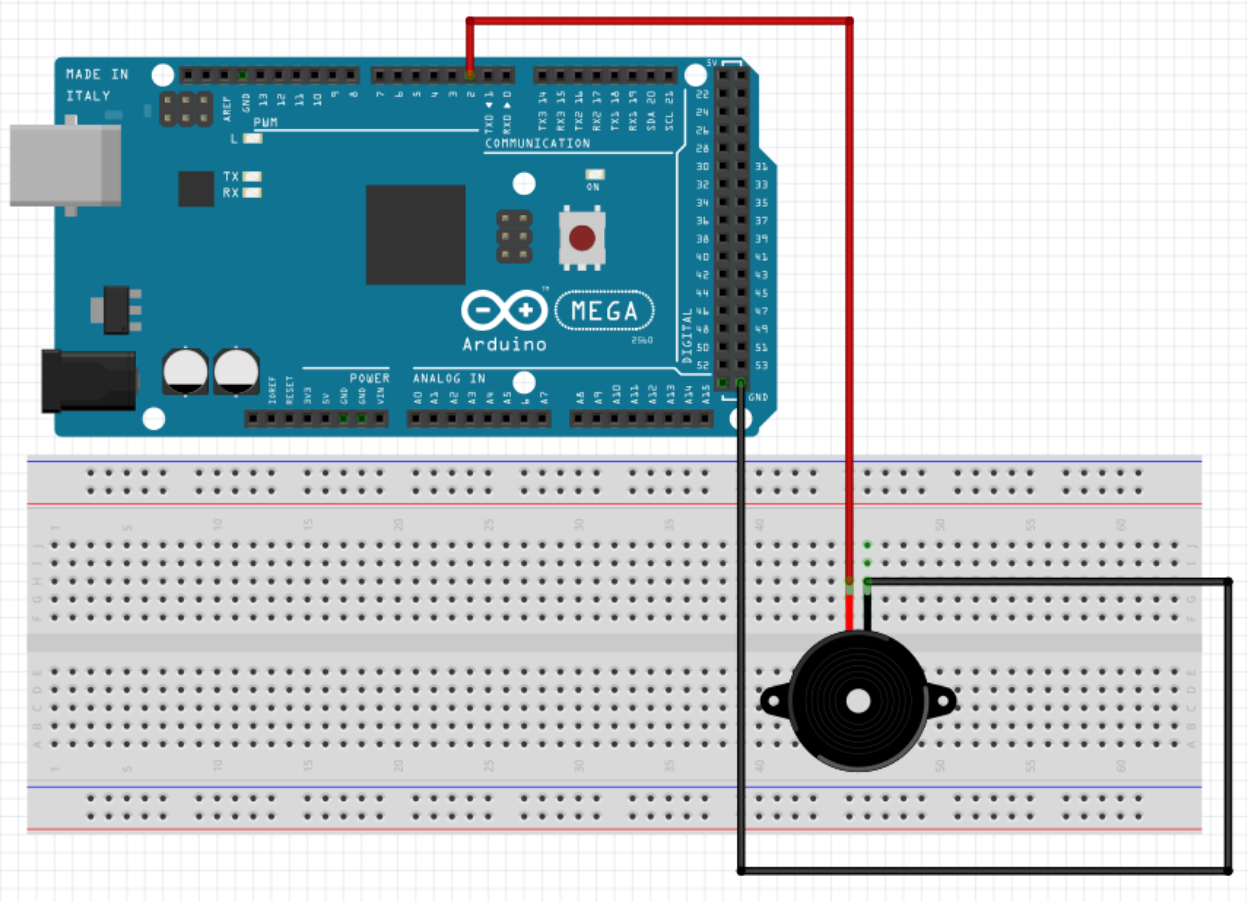
```
  //używamy komendy digitalRead(numerPinu)
  int zmiennaStanPrzelacznika = digitalRead(2); // "int"

  // teraz dodamy komendę warunkową - jeżeli, ang. "if".
  // Jeżeli przycisk jest włączony (czyli na pin płynie prąd - stan HIGH), wykonaj:
  if (zmiennaStanPrzelacznika == HIGH) {
    digitalWrite(8, HIGH); // załączamy głośnik
    delay (200); // czekamy 0,2 sekundy
    digitalWrite(8, LOW); // wyłączamy głośnik
    delay(200); // czekamy 0,2 sekundy

    digitalWrite(8, HIGH); // załączamy głośnik
    delay (200); // czekamy 0,2 sekundy
    digitalWrite(8, LOW); // wyłączamy głośnik
    delay(200); // czekamy 0,2 sekundy

    digitalWrite(8, HIGH); // załączamy głośnik
    delay (200); // czekamy 0,2 sekundy
    digitalWrite(8, LOW); // wyłączamy głośnik
    delay(200); // czekamy 0,2 sekundy

    // jak widać, blok kodu jest skopiowany kilkukrotnie, żeby sygnał się powtórzył.
    // jest na to inny i lepszy sposób z użyciem "for" albo "while" -
    // osoby chętne mogą wyszukać w internecie,
    // ale zostanie omówiony przy okazji innych ćwiczeń.

  }
}
```

Teraz płytka powinna odgrywać buzzerem kilka sygnałów po naciśnięciu lub przytrzymaniu włącznika.

Gramy komendami z Serial Monitor przy pomocy AnalogWrite

Jeżeli dobrze się przyjrzymy, w oknie Serial Monitor zauważymy pole do wpisywania wartości oraz przycisk z komendą **SEND**. Możemy wysyłać przy pomocy tego pola sygnały do płytki Arduino.

Zrobimy więc tak, by płytka Arduino wysyłała do naszego brzęczyka prąd o różnym napięciu – zależnym od wpisanej przez użytkownika wartości w Serial Monitor. Ponieważ wykorzystywany przez nas dotychczas **DigitalWrite** ma tylko dwie wartości – **HIGH** lub **LOW**, musimy użyć innej funkcji. Jest nią **AnalogWrite**. Ma ona wartości od 0 do 255. By ją wykorzystać, musimy nieco zmienic układ naszej płytki, tak, by brzęczyk był podpięty do jednego z pinów oznaczonych jako PWM:



Następnie tworzymy nowy szkic, i wpisujemy do niego poniższy kod:

```
void setup() {
  // opiszemy nasz pin analogowy
  pinMode(9, OUTPUT);

  // rozpoczniemy komunikację między płytką a komputerem:
  Serial.begin(9600);
}
```

```
void loop(){
  // jeżeli nowe dane otrzymane od użytkownika... (”available" - "dostępne")
  if (Serial.available()){

    // ... to odczytaj je, przetłumacz do liczby całkowitej (parseInt),
    // i zapisz do zmiennej komendaUzytkownika
    int komendaUzytkownika = Serial.parseInt();

    // ogranicz (constrain) ją do wartości między 0 a 255,
    // bo tylko tyle możemy wysłać maksymalnie przez pin analogowy
    komendaUzytkownika = constrain(komendaUzytkownika, 0, 255);

    // następnie ustaw wartość pina na tę którą podał użytkownik
    analogWrite(9, komendaUzytkownika);

    // i dla pewności pokaż nam ją w monitorze Serial
    Serial.println(komendaUzytkownika);
  }
}
```

Tym sposobem możemy sterować tonami dźwięków wydawanymi przez buzzer wpisując ich wartości do SerialMonitor.

MOŻLIWE MODYFIKACJE DLA MŁODSZYCH KLAS:

Pracując z uczniami w młodszych klasach można wykorzystać zamiast Arduino IDE program mBlock. W przypadku zajęć z młodszymi dziećmi warto zwrócić uwagę na ewentualne problemy z dokładnym podłączaniem przewodów.

ZADANIE SPRAWDZAJĄCE UMIEJĘTNOŚCI ZDOBYTE PODCZAS ZAJĘĆ:

Uczeń / uczennica, pracując samodzielnie albo w dwu- lub trzyosobowym zespole buduje układ z wykorzystaniem Arduino i buzzera. Zadanie polega na podłączeniu buzzera do płytki, wytłumaczeniu, gdzie biegną piny zasilające i sygnałowe, oraz sposobu podłączenia przełączników, a także wyjaśnieniu zasady działania programu, w szczególności wyrażenia warunkowego "if".

PIGUŁKA WIEDZY I INSPIRACJI DLA OSÓB PROWADZĄCYCH:

Kurs programowania Arduino Forbot:
http://forbot.pl/blog/artykuly/programowanie/kurs-arduino-w-robotyce-1-wstep-id936

Podstawowe informacje na temat prądu elektrycznego:
http://forbot.pl/blog/artykuly/podstawy/podstawy-elektroniki-1-napiecie-prad-opor-zasilanie-id3947

Jak działa płytka stykowa (prototypowa):
https://pl.wikipedia.org/wiki/P%C5%82ytka_prototypowa

Wywiad z Imogen Heap, artystką budującą elektronikę do tworzenia muzyki
https://www.youtube.com/watch?v=ci-yB6EgVW4

Czym jest brzęczyk, i jaka jest jego zasada działania:
https://pl.wikipedia.org/wiki/Brz%C4%99czyk

Zasady bezpieczeństwa w postępowaniu z modułami Arduino:
https://www.rugged-circuits.com/10-ways-to-destroy-an-arduino/

Scenariusz został opracowany na potrzeby projektu „MoboLab – roboty i tablety w Twojej szkole". Celem projektu jest zwiększenie kompetencji informatycznych z zakresu programowania i wykorzystywania technologii mobilnych w uczeniu się, a także kreatywności, innowacyjności i umiejętności współpracy w zespole z wykorzystaniem TIK, uczniów / uczennic z (UCZ) z 6 szkół podnadgimnazjalnych i 4 gimnazjów Wołomina i Zielonki. Projekt dofinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego (Regionalny Program Operacyjny Województwa Mazowieckiego na lata 2014-2020, Oś Priorytetowa X. Edukacja dla rozwoju regionu, Działanie 10.1. Edukacja ogólna i przedszkolna, Poddziałanie 10.1.2. Edukacja ogólna w ramach ZIT).