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

SCENARIUSZ 20. CZUJNIK HALLA

scenariusz zajęć pozalekcyjnych

autor: Michał Podziomek

redakcja: Agnieszka Koszowska

SŁOWA KLUCZOWE:

Arduino, programowanie, elektronika, czujnik Halla

KRÓTKI OPIS ZAJĘĆ:

Podczas zajęć uczniowie i uczennice poznają **czujnik Halla**, jego zasadę działania i zastosowanie. Budują obwód z wykorzystaniem Arduino, płytki stykowej, **buzzera** i czujnika Halla. Programują **moduł z czujnikiem Halla**, tak aby odczytać wartości z czujnika i wysłać sygnał dźwiękowy do buzzera.

WIEDZA I UMIEJĘTNOŚCI ZDOBYTE PRZEZ UCZNIA / UCZENNICĘ:

- wie, czym są mikrokontrolery i do czego służą,
- 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),
- potrafi podłączyć buzzer i moduł z czujnikiem Halla do Arduino,
- wie, co to jest czujnik Halla,
- 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,
- przewody połączeniowe,
- czujnik natężenia prądu z sensorem Halla,
- bateria 9V, cienki drut miedziany w izolacji, gwóźdź, brzęczyk piezoelektryczny,
- projektor i laptop (w części teoretycznej).

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 Arduino, kabel USB A-B, płytkę stykową, przewody połączeniowe i inne elementy wykorzystywane w tym scenariuszu,
- 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:

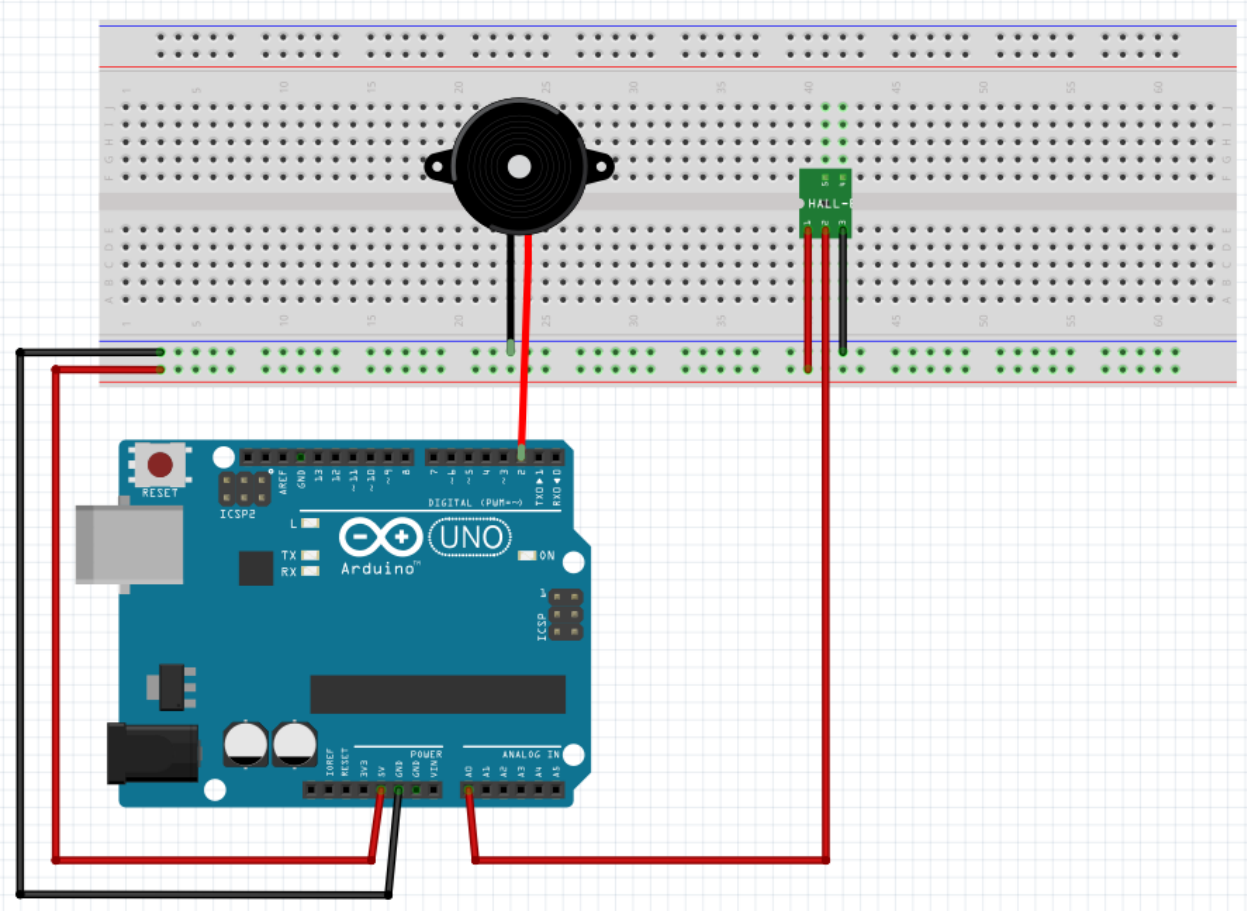
Podłączenie Arduino, uruchomienie programu Arduino IDE i przypomnienie podstawowych informacji – ok. 15 minut

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.

Poznajemy czujnik pola magnetycznego, czyli czujnik Halla – 15 minut

Czujnik Halla to układ pracujący na prądzie stałym, zmieniający właściwości napięcia przez niego płynącego w zależności of pola magnetycznego, w którym się znajdzie. Do pina 1 dostarczamy napięcie, czyli PWR. Z pina 2 odbieramy sygnał, czyli zmienione napięcie. Pin 3 jest podpięty do uziemienia, GND.

W zestawach Arduino dostępny jest czujnik Halla zamontowany w układzie do pomiaru prądu (czyli tzw. amperomierza). Układ zawiera duży czujnik Halla (ośmiopinowy), natomiast sama płytka ma dwa konektory: duży zielony ze śrubkami, do przyłączenia przewodów, na których będzie przeprowadzany pomiar, oraz trzy piny do podłączenia do płytki Arduino. Są one oznaczone identycznie jak na małych czujnikach Halla, czyli VCC (zasilanie), OUT (czyli pin z którego płynie sygnał), oraz GND (czyli uziemienie). Podłączmy go teraz do płytki wraz z głośnikiem – według schematu jak na poniższym rysunku:



Buzzer, czyli brzęczyk, jest zapięty do pinu numer 2, oznaczonego **PWM**. Jest to pin analogowy, pozwalający na wysyłanie określonych przez nas wartości napięcia do brzęczyka. To spowoduje, że możemy na brzęczyku wygrywać różne tony, zmieniając wielkość napięcia do niego biegnącego. Druga nóżka, w naszym wypadku czarna, jest zapięta do wspólnej linii **GND**, czyli uziemienia.

Używamy długich rzędów pinów z boku płytki stykowej, biegnących koło niebieskiej i czerwonej linii, do poprowadzenia linii zasilania PWR i uziemienia GND, po czym podłączamy do nich komponenty w zaznaczony sposób.

Piny na środku płytki stykowej, rozdzielone rowkiem, biegną prostopadle do pinów bocznych (czyli tych przy liniach niebieskiej i czerwonej). Zwracamy uwagę na ich zielone zakolorowanie w okolicach przycisków. Te piny, w liniach zgodnie z zakolorowaniami, są ze sobą połączone.

Czujnik Halla jest podłączony pod **pin A0** oznaczony jako **Analog Input**. Jest to pin potrafiący odbierać i kwantyfikować (czyli zamieniać na liczbę) napięcie na niego płynące. Pracuje na zakresie od 0 do 5 Voltów i ma 1023 stopnie czułości. Oznacza to, że potrafi rozpoznawać napięcie 5V/1024=0.049V, czyli 4.9mV (miliVoltów). Maksymalna prędkość odczytu czujnika to 10 000 razy na sekundę.

Przygotowujemy program sterujący czujnikiem – 15 minut

Chcemy móc odczytywać wartości z czujnika Halla i przy ich pomocy grać dźwiękiem na buzzerze. Nasza płytka będzie się zachowywała bardzo podobnie jak wydający dźwięki licznik Geigera, tylko zamiast promieniowania będzie rozpoznawała zmienne pole magnetyczne.

ZMIENNE – VARIABLES

Będziemy wykorzystywać **zmienne**, czyli symboliczne słowa, do których przypisujemy w programie wartość, która się zmienia. Np. w naszym przypadku zmienna będzie przechowywała wartość odczytu czujnika Halla. Żeby zdefiniować zmienną, musimy napisać jaki jest jej typ. Na początek użyjemy zmiennej która jest liczbą całkowitą, po angielsku integer, w skrócie – **int**:

int stanCzujnika = 0;

Ten przykład definiuje (czyli określa) zmienną i nadaje jej wartość początkową 0 za pomocą znaku **=**. Pojedynczy znak „=” zmienia wartości.

Uwaga! Jeżeli chcemy porównać wartości, używamy PODWÓJNEGO znaku równości, czyli **==**. Takie wyrażenie program potraktuje jak pytanie, na które odpowie wartością logiczną: **TRUE** lub **FALSE** (prawda, lub fałsz).

ANALOG WRITE

Funkcja **analogWrite** pozwala wysłać różne wartości napięcia poprzez wyznaczone do tego piny. Piny które są obsługiwane przez tę funkcję, to te oznaczone **PWM** (Pulse Width Modulation - modulacja szerokości impulsów). Różnica z **digitalWrite** polega na tym, że digitalWrite może być tylko włączone, lub nie, wysyłając maksymalny dostępny prąd.

analogWrite ma swoje minimalne i maksymalne wartości które skaluje do wysyłanego prądu. Muszą się one mieścić w przedziale od 0 do 255.

Dla zainteresowanych: opis funkcji **analogWrite** jest dostępny w języku angielskim pod poniższym adresem:
https://www.arduino.cc/en/Reference/AnalogWrite

O tym, czym jest PWM, możemy poczytać na Wikipedii pod poniższym adresem:
https://pl.wikipedia.org/wiki/Modulacja_szerokości_impulsów

Polecamy również przestudiować w wolnym czasie dołączone do oprogramowania Arduino przykłady.

ANALOG READ

Funkcja **analogRead** pozwala odczytać wartość napięcia z pinu typu wejścia analogowego, czyli **analog in**. Na płytce piny te są oznaczone cyfrą poprzedzoną literą A, np. A0, A1, A2. Wartość analogowa od cyfrowej różni się skalą – wartość cyfrowa ma dwie wartości - prąd lub brak prądu. Wartość analogowa ma ich wiele więcej – w naszym przypadku to 1023 wartości. Odczytuje napięcie od 0 do 5 Voltów, a więc 5V/1023 = 0.004887(...)V, a więc w przybliżeniu - 0.005Volta.

Więcej o analogRead możemy poczytać w języku angielskim na stronie projektu Arduino pod adresem:
https://www.arduino.cc/en/Reference/analogRead

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.

Programowanie układu – 20 minut

Przystąpimy teraz do pisania kodu, który odczyta wartości z czujnika Halla i wyśle odpowiedni sygnał dźwiękowy przy pomocy buzzera. Napotkamy tutaj jeden problem, który warto teraz omówić.

Mianowicie z czujnika Halla odbierzemy wartości w przedziałach 0-1023, a na buzzer musimy wysyłać wartości w przedziałach 0-255. Będziemy musieli zatem zmapować wartości względem siebie, tzn. „ścisnąć" wartości odbierane z czujnika Halla tak, by zamknęły się w 255 (czyli: żeby wartość 1023 czujnika Halla po „ściśnięciu" była równa 255). Użyjemy do tego celu funkcji **map()**, czyli „mapuj".

MAP

Map to funkcja pozwalająca dopasować do siebie (przeskalować) dwa zakresy danych. Np. gdy odbieramy dane z czujnika na pinie analogowym wejścia, otrzymujemy wartości w przedziale od 0 do 1023. Jeżeli chcemy natomast wysłać dane na pin analogowy wyjścia, mamy do dyspozycji tylko wartości od 0 do 255. Musimy więc „skurczyć", czyli przeskalować zakres wartości otrzymywanych z sensora tak, by jego maksymalna wartość 1023 odpowiadała wartości 255. Używamy tej funkcji w sposób następujący:

wartosc_zmapowana = map(odczyt, 0, 1023, 0, 15);

W nawiasie w kolejności: odczyt z sensora, zakres minimalny odczytu, zakres maksymalny odczytu, zakres minimalny na który skalujemy, zakres maksymalny na który skalujemy. Uwaga! map() nie działa poprawnie dla bardzo małych wartości, ponieważ nie pracuje na ułamkach.

CONSTRAIN

Ogranicza wartość wysyłaną na pin. Np. pin PWM akceptuje wartości od 0 do 255. Nie chcemy wysyłać na niego wartości większych. Zatem wysyłając wartość i chcąc

mieć pewność, że nie będzie mniejsza od 0 ani większa od 255, używamy na niej funkcji **Constrain()**:

wartosc_wysylana = (wartosc_wysylana, 0, 255) // czyli nasza wartość, zakres minimalny, zakres maksymalny).

Napiszmy teraz program:

```
void setup(){
// deklarujemy piny
pinMode (A0, INPUT); // pin na którym jest podpięty czujnik Halla
pinMode (2, OUTPUT); // pin na którym jest podpięty buzzer
}

void loop(){
  // zczytujemy wartość z czujnika Halla
  int wartoscHalla = analogRead(A0);

  // mapujemy wartość czujnika Halla na dźwięk buzzera
  int dzwiekBuzzera = map(wartoscHalla, 0, 1023, 0, 255);

  // dla pewności ograniczamy wartość wysyłaną na buzzer do maksymalnej 255.
  // warto zauważyć że zmienna modyfikuje samą siebie. Jest to częsta praktyka.
  dzwiekBuzzera = constrain(dzwiekGlosnika, 0, 255);

  // gramy na buzzerze
  analogWrite(2, dzwiekBuzzera);
}
```

Ładujemy kod na płytkę Arduino. Przy włączaniu pojedynczych przełączników, każdy z nich powinien wydawać inny dźwięk.

Zadania dla uczniów – 15 minut

Poniżej przykłady proponowanych zadań do wykonania przez uczniów:

1. Użyj baterii i drucika żeby zbudować elektromagnes

Nawijamy na gwóźdź cienki przewód elektryczny w cienkiej izolacji. Jeden koniec zapinamy na biegun dodatni baterii 9V. Drugi zapinamy na biegun ujemny. Zbliżamy go do czujnika Halla i obserwujemy zmiany dźwięku na buzzerze. Następnie zwiększamy lub zmiejszamy ilość zwojów. I ponownie zbliżamy do buzzera, obserwując zmianę.

2. Korzystamy z modułu Halla jako wskaźnika prądu, czyli amperomierza

Podłączamy oba bieguny baterii 9V do DUŻEGO ZIELONEGO KONEKTORA ZE ŚRUBKAMI na module czujnika Halla. Uwaga! bateria pracuje na napięciach mogących uszkodzić płytkę Arduino – nie wolno zetknąć jej biegunów z żadnym innym elementem - może to trwale uszkodzić płytkę! Obserwujemy dźwięk buzzera. Spróbujmy dołączyć różne rezystory do układu przy pinach baterii i zaobserwować zmiany dźwięku buzzera.

MOŻLIWE MODYFIKACJE DLA MŁODSZYCH KLAS:

Pracując z uczniami w młodszych klasach można wykorzystać zamiast Arduino IDE program S4A (Arduino for Scratch) lub 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, płytki stykowej, buzzera i czujnika Halla. Zadanie polega na podłączeniu płytki z buzzerem i czujnikiem Halla, wytłumaczeniu, gdzie biegną piny zasilające i sygnałowe, oraz wyjaśnieniu zasady działania programu, w szczególności funkcji „map”.

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

Czym jest zjawisko Halla i jakie są jego zastosowania:
https://pl.wikipedia.org/wiki/Zjawisko_Halla

Opis stosowanego modułu z czujnikiem Halla:
http://www.gotronik.pl/pdf_datasheet.php?products_id=1720

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

Funkcja Map():
https://www.arduino.cc/en/reference/map

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).