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

SCENARIUSZ 23. CZUJNIK ODBICIOWY

scenariusz zajęć pozalekcyjnych

autor: Michał Podziomek

redakcja: Agnieszka Koszowska

SŁOWA KLUCZOWE:

Arduino, programowanie, elektronika, czujnik odbiciowy

KRÓTKI OPIS ZAJĘĆ:

Podczas zajęć uczniowie i uczennice uczą się, jak stworzyć układ elektroniczny za pomocą Arduino oraz **czujnika odbiciowego**. Przygotowują układ i tworzą program pozwalający na odczyt odległości z wykorzystaniem czujnika odbiciowego.

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),
- 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,
- rozumie działanie sensora odległości składającego się z pary diod - nadajnika i odbiornika pracujących w zakresie podczerwieni,
- 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”):

2

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 odbiciowy, buzzer
- opornik 100K omów,
- 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 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.

Omawiamy zasadę działania czujnika odbiciowego – 10 minut

Zajęcia są poświęcone czujnikowi odbiciowemu i wykorzystywaniu go w układach elektronicznych z Arduino. Rozpoczynamy od omówienia zasady działania czujnika.

Czujnik odbiciowy to urządzenie, które emituje pole elektromagnetyczne lub strumień promieniowania elektromagnetycznego i mierzy jego wartość powrotną, rejestrując jego zmiany. Obiekt wprowadzający zmiany nazywamy **celem czujnika**. Dzięki takiemu czujnikowi możemy łatwo zbudować mechanizm

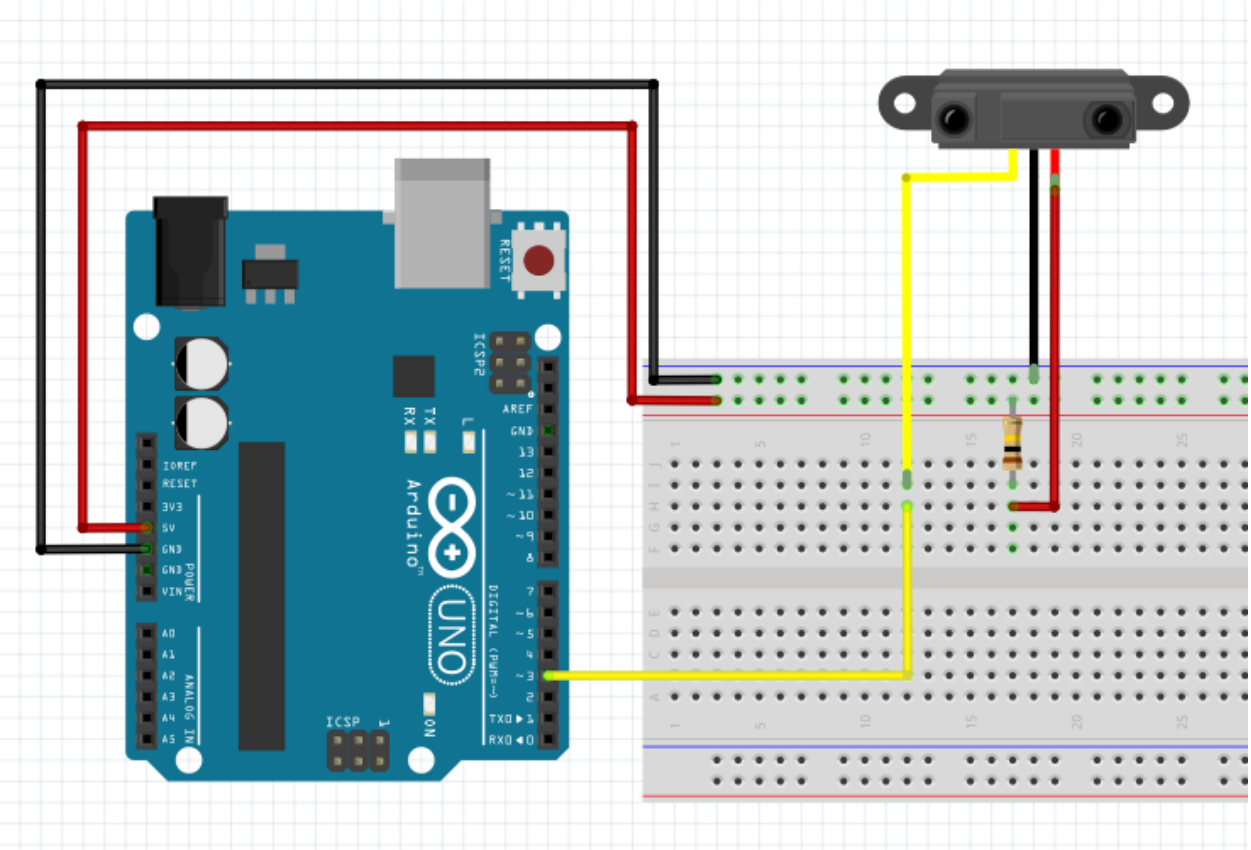
włączający światło w pokoju, np. w momencie, gdy znajdzie się w nim człowiek, lub wyłączający światło, gdy pomieszczenie opuści ostatnia osoba.

Czujnik, na którym pracujemy, zawiera w sobie diodę oraz fotorezystor i odbiera wyłącznie światło podczerwone. Przykładowe pytania do uczniów:

- *jakie inne zastosowania czujnika na podczerwień są Wam znane?*
- *Czy (i jakie) macie pomysły na wykorzystanie takiego czujnika?*
- *jakie inne czujniki odbiciowe znacie?*

Montujemy układ – 10 minut

Połączenia należy wykonać tak, jak na poniższym schemacie. Wykorzystujemy opornik 100K omów.



Tworzymy program – 10 minut

Na początek napiszemy prosty program który pozwoli nam odczytać dane z czujnika i zobaczyć, jaki jest ich zakres. Będziemy korzystać z funkcji **analogRead.**

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

```
void setup(){
  pinMode(A3, INPUT);
  Serial.begin(9600);
}
void loop(){
  Serial.println(analogRead(A3));
}
```

Możemy obejrzeć rezultat w **Serial Monitor** i/lub **Serial Plotter** zbliżając rękę do sensora i ją oddalając.

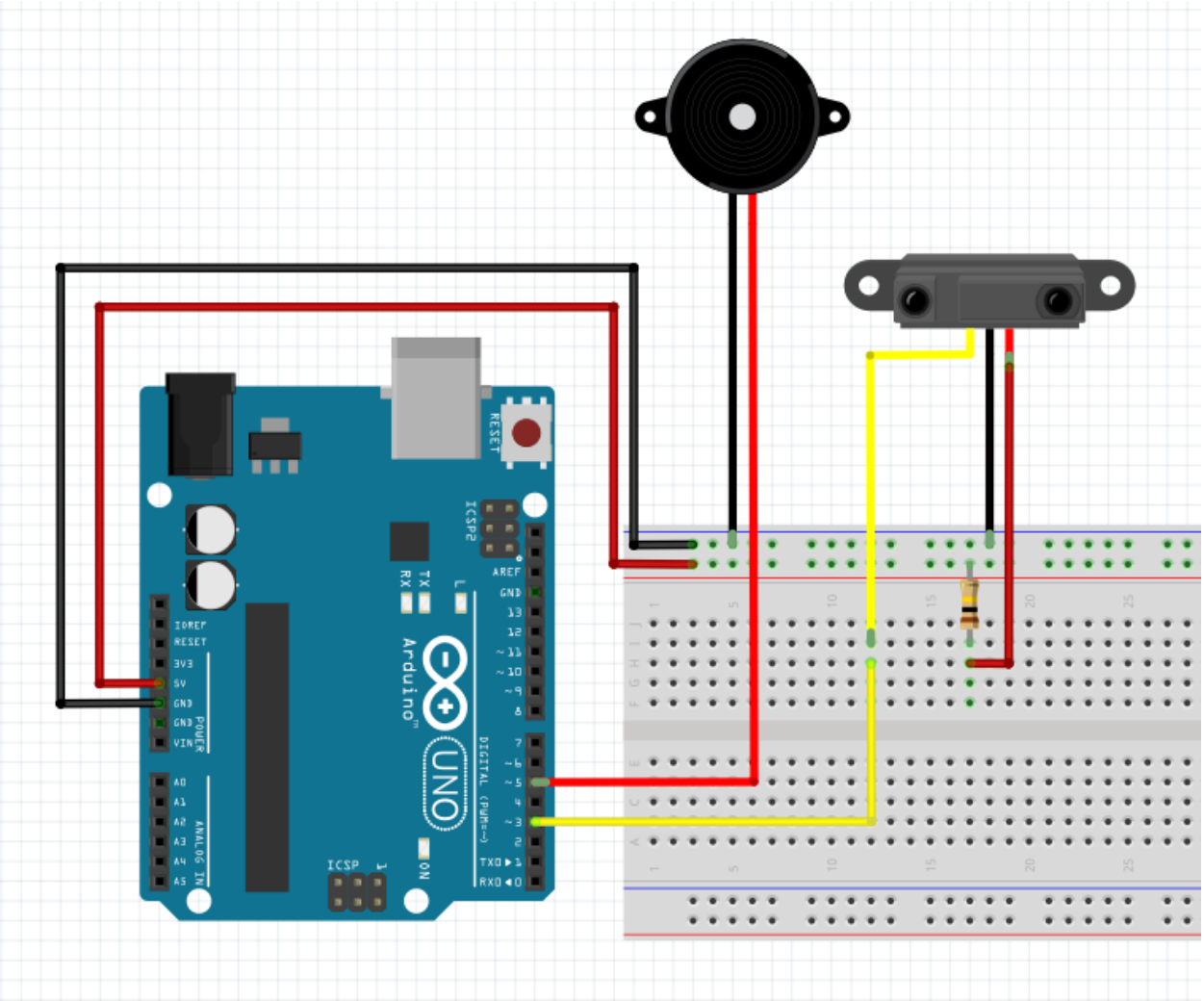
W tym miejscu możliwy jest podział zajęć na dwie części (kolejna część scenariusza będzie realizowana na następnych zajęciach).

Przypomnienie materiału, odtworzenie układu – 10 minut

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

Modyfikujemy układ – 10 minut

Dodajemy do płytki buzzer, którym będziemy grać dźwięk o wysokości zależnej od odległości obiektu. Modyfikujemy układ w następujący sposób:



Modyfikujemy kod – 25 minut

Dodajemy do kodu obsługę dla brzęczyka oraz automatyczne mapowanie wartości z sensora na wartości brzęczyka. Spowoduje to autokalibrację brzęczyka w miarę jak sensor będzie pracował na swojej pełnej skali wartości. Będziemy korzystać ze **zmiennych**, funkcji **analogWrite** i **map** oraz **instrukcji warunkowej**.

ZMIENNE – VARIABLES

Zmienne to symboliczne słowa, do których przypisujemy w programie wartość, która się zmienia. Może to być odczyt z czujnika, który chcemy potem jakoś zmienić albo np. zaokrąglona wartość liczby Pi. 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.

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.

INSTRUKCJE WARUNKOWE

Instrukcja warunkowa to instrukcja, która jest wykonywana tylko wtedy, gdy zostanie spełniony określony warunek. Warunek ten jest określany przy pomocy zapytania, np. *Czy przycisk włączony? Czy wartość na sensorze jest większa lub mniejsza od jakiejś liczby? Czy wartość na sensorze 1 jest większa niż na sensorze 2?* Musimy oczywiście właściwie sformuować to zapytanie. Wygląda ono w sposób następujący:

if (mojaZmienna == 10){ // jeżeli mojaZmienna jest równa 10
// wykonaj kod tutaj zawarty
}

Podwójny znak równości **==** jest tzw. **operatorem relacji**. Więcej informacji o operatorach relacji jest dostępnych np. w artykule na Wikipedii:
https://pl.wikipedia.org/wiki/Operator_relacji

Wynikiem zapytania (mojaZmienna == 10) będzie stwierdzenie **prawda** lub **fałsz** (**TRUE** lub **FALSE**).

Przykłady innych operatorów relacji to:

- a<b , a>b - mniejsze/większe
- a<=b , a>=b - mniejsze równe/większe równe
- a!=b - nie jest równe (wykrzyknik to znak zaprzeczenia)

```
int sensorMaksimum = 0;

void setup(){
  pinMode(A3, INPUT);
  pinMode(A5, OUTPUT);
  Serial.begin(9600);
}

void loop(){
  int wartoscOdczytu = analogRead(A3);
  Serial.println(wartoscOdczytu);

  if (sensorMaksimum < wartoscOdczytu){ // ustalamy maksymalne wartości dla sensora
    sensorMaksimum = wartoscOdczytu;
  }

  // gramy na buzzerze dźwięk odpowiadający aktualnemu położeniu obiektu,
  // zmapowanemu na maksymalną wartość odczytu z sensora
  analogWrite(A5, map( wartoscOdczytu, 0,  sensorMaksimum, 0, 255);
}
```

Zadanie dodatkowe

Oprogramować czujnik z wykorzystaniem operacji **PIN** i **DDR** poznanych w scenariuszu nr 21. *Nadajnik i odbiornik podczerwieni*.

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. Klasy młodsze mogą wykonać zadania dodatkowe, jeżeli pozwoli na to czas i doświadczenie uczestników.

ZADANIE SPRAWDZAJĄCE UMIEJĘTNOŚCI ZDOBYTE PODCZAS ZAJĘĆ:

Uczeń / uczennica, pracując samodzielnie albo w dwu- lub trzyosobowym zespole buduje układ z wykorzystaniem Arduino, płytki stykowej i czujnika odbiciowego. Zadanie polega na prawidłowym podpięciu pełnego układu, tak aby współdziałał z napisanym kodem, wyjaśnieniu zasady działania czujnika odbiciowego i określeniu – na podstawie odczytu z Serial Monitor i/lub Serial Plotter – przedziału wartości i orientacyjnej precyzji czujnika.

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

Projekt automatycznego włącznika/wyłącznika światła:
http://www.instructables.com/id/Automatic-Switch-ON-And-OFF-Light/

Biblioteka specjalizująca się w odczcie danych z prostych sensorów odbiciowych:
http://forum.arduino.cc/index.php?topic=101156.0

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).