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

SCENARIUSZ 21. NADAJNIK I ODBIORNIK PODCZERWIENI

scenariusz zajęć pozalekcyjnych

autor: Michał Podziomek

redakcja: Agnieszka Koszowska

SŁOWA KLUCZOWE:

Arduino, programowanie, elektronika, podczerwień, IR

KRÓTKI OPIS ZAJĘĆ:

Podczas zajęć uczniowie i uczennice uczą się, jak stworzyć układ elektroniczny za pomocą Arduino oraz **czujnika IR**: **nadajnika i odbiornika podczerwieni**. Przygotowują program pozwalający odczytywać dane wysyłane do układu z pilota.

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, na czym polega działanie układu, w którym dwa elementy komunikują się za pomocą 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”):

5

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,
- odbiornik i nadajnik podczerwieni,
- 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.

Wstęp – rozmowa o komunikacji IR – 15 minut

Zajęcia są poświęcone komunikacji za pomocą nadajnika i odbiornika podczerwieni (IR). Rozpoczynamy od dyskusji z uczniami, podczas której zadajemy takie pytania, jak:

- *gdzie jest stosowana komunikacja IR?*
- *na czym ona polega?*
- *co może ją zakłócić, np. czy będzie działała w słoneczny dzień?*
- *czy sygnał z jednego nadajnika może być odebrany przez wiele czujników?*

Funkcje **digitalWrite** i **digitalRead** są zbyt wolne, by ich użyć do poprawnej obsługi komunikacji IR. W tym celu musimy bezpośrednio odczytywać informację z pinów, z pominięciem funkcji digitalRead().

digitalRead() poza samym odczytywaniem wartości pina wykonuje wiele dodatkowych operacji sprawdzających, np. czy pin jest przygotowany do odczytu. W rezultacie, użycie jej zabiera nam wiele cennych cykli procesora i jest bardzo wolne.

Żeby odebrać dane z diody IR, użyjemy pewnego triku:

- odczytamy wartości wszystkich pinów na danym porcie (Arduino ma kilka portów z pinami: **B** (cyfrowe piny od 8 do 13), **C** (piny analogowe), **D** (cyfrowe piny od 0 do 7),
- z tego wyciągniemy informację o pinie, który nas interesuje,
- przetworzymy ją na wyjście.

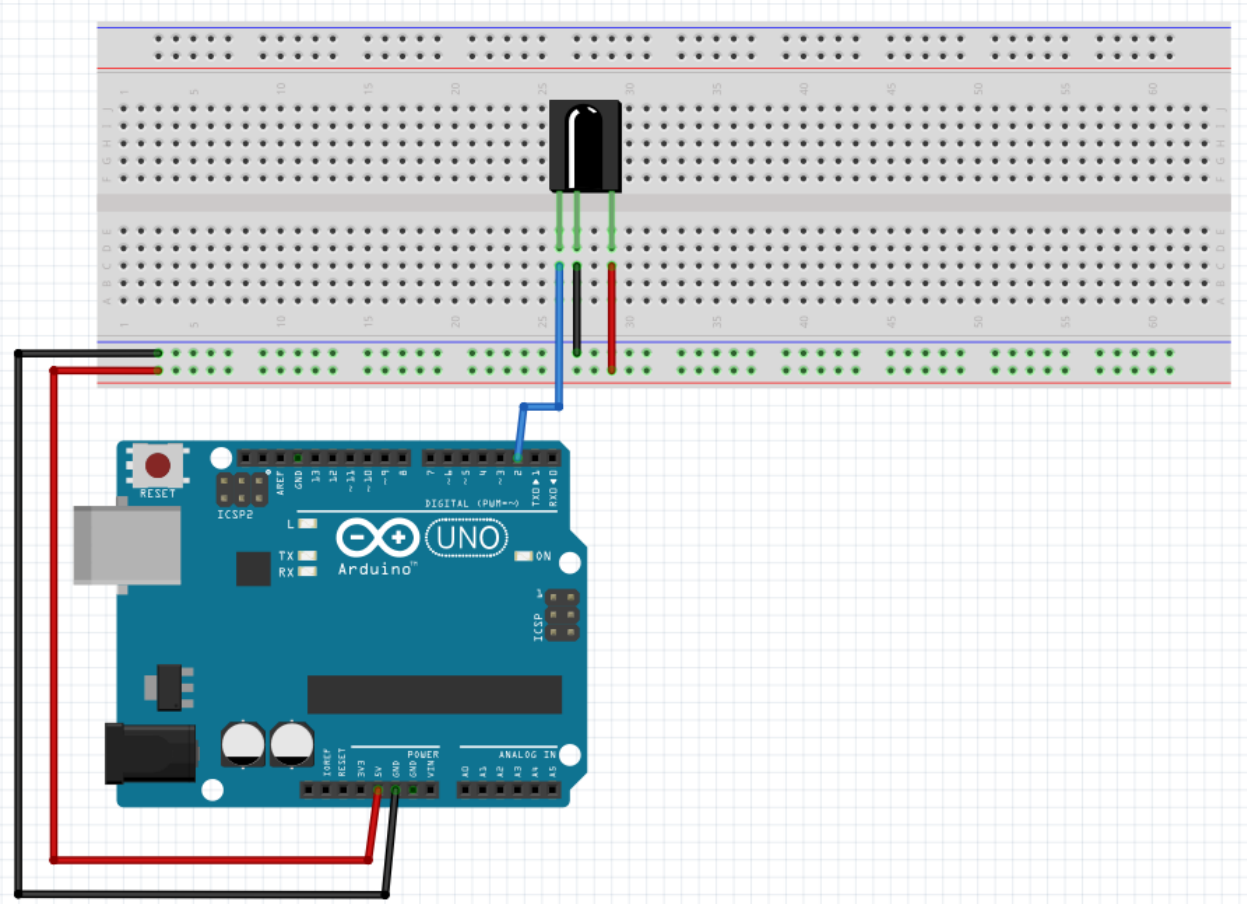
Uwaga! Unikamy używania pinów 1 i 2, ponieważ służą one do programowania Arduino. Jakiekolwiek zmiany na tych portach mogą spowodować, że nie będziemy w stanie programować naszej płytki bez użycia specjalnego narzędzia! Więcej informacji w dokumentacji pod adresem:
https://www.arduino.cc/en/Reference/PortManipulation

Arduino nie ulegnie uszkodzeniu, natomiast trzeba będzie użyć specjalnego programatora, by przywrócić piny do poprzedniego stanu. **Jest bardzo ważne, by nie popełnić błędu w przepisywaniu kodu!**

Omawiamy operacje matematyczne na liczbach binarnych, binarny odczyt i zapis portów.

Montaż układu – 15 minut

Płytkę należy podłączyć w sposób przedstawiony na poniższym diagramie:



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

Żeby odebrać dane z Arduino i czujnika IR, normanie użylibyśmy poniższego kodu:

```
void setup(){
  Serial.begin(9600);
  pinMode(2, INPUT);
}
void loop(){
```

```
  Serial.writeln(digitalRead(2));
}
```

Natomiast sygnał z nadajnika i odbiornika podczerwieni działa zbyt szybko by zostać obsłużonym przez ten kod. Będziemy musieli napisać nasz kod bardziej profesjonalnie – tak by był wykonywany w mniejszej liczbie cykli. Użyjemy do tego komendy **PIN**, która pozwala nam na bezpośrednie adresowanie pinów. Jest ok. 50 do 100 razy szybsza od wbudowanej funkcji **digitalRead**, ale jest również bardziej skomplikowana i bardziej niebezpieczna dla naszego Arduino. Dlatego musimy zachować uwagę i ostrożność! Nasz kod będzie wyglądał następująco:

```
// w miliseknd ach - to jest całkowity czas nadawania sygnału przez pilota,
// czyli wszystkie impulsy w komendzie którą wydaje
#define dlugoscSygnalu 65000

// Rozdzielczość, czyli co ile milisekund mamy sprawdzać sygnał z pilota
#define rozdzielczosc 20

// Pulsy będziemy przechowywali w tzw. tablicy, po angielsku "array".
// pracujemy na wartościach binarnych, i nie znamy czestotliwości nadawania
// naszego pilota. Wiemy że wysyła jedynki i zera - sygnał i brak sygnału.
// dwie jedynki trwają 2x dłużej niż jedna. Zera tak samo.
// więc możemy zapisać w tablicy pary 1 i 0
// ale zamiast ich wartości wpiszemy czas przez jaki były nadawane.
uint16_t pulsy[100][2]; // 100 par - pierwsza to HIGH, druga to LOW.
                        // będziemy zapisywać czas - jak długo są nadawane.
uint8_t obecnyPuls = 0; // to będzie adres aktualnie używanego impulsu w
tablicy

void setup(){
  // Inicjujemy piny grupy D czyli, od 7 do 0, komendą DDR. A dokładnie:
  // B - oznacza tryb binarny
  // 1 oznacza pin do zapisu, 0 - pin do odczytu.
  // piny są w kolejności od największego, 8 cyfr, czyli: 7, 6, 5, 4, 3, 2, 1, 0
  // piny 1 i 0 są używane do komunikacji z komputerem,
  // więc muszą mieć wartości 1 i 0, by nie uszkodzić Arduino!!!
  DDRD = B11111110; // 7 jedynek, 1 zero.
```

```
  // Uruchamiamy komunikację serial
  Serial.begin(9600);

}

void loop(){
  // liczniki czasu które będą kończyły impuls, nadamy im wartość później
  uint16_t wysokiTimer = 0;
  uint16_t niskiTimer = 0;

  // odczytujemy wartość z pinu w sposób szybszy niż digitalRead().
  // PIND zwraca nam ciąg 8 binarnych liczb - po jednej dla pinu.
  // Czyli jeżeli pin 7 będzie LOW a reszta HIGH, dostaniemy 01111111,
  // jeżeli pin 6 będzie LOW a reszta HIGH, dostaniemy 10111111, itd.
  // interesuje nas pin 2, więc musimy wyciągnąć tylko jego wartość.
  // pin 0 ma adres 00000001
  // pin 1 ma adres 00000010
  // pin 2 ma adres 00000100 itd.
  // jeżeli zrobimy tak: PIND & (adres binarny pinu 2), otrzymamy tylko jego
wartość
  // w postaci binarnej - czyli 0 lub 1.
  // "&" top operator który wykonuje operację AND na każdym bicie dwóch liczb.
  // czyli 11 & 00 = 00; 10 & 11 = 10; 11 & 11 = 11; i tak dalej.

  while (PIND & (B00000100)){ // tak długo jak pin 2 ma stan HIGH
    // liczymy czas
    wysokiTimer ++;
    delayMicroseconds(rozdzielczosc);

    // Jeżeli puls jest dłuższy od naszej wartości zdefiniowanej na początku
programu,
    // wymuszamy koniec nadawania:
    if ((wysokiTimer >= dlugoscSygnalu) && ( obecnyPuls !=0)){
      drukujPulsy(); // tę funkcję zdefiniujemy potem - wydrukuje zebrane
impulsy do serial monitor
      obecnyPuls = 0; // przygotujemy program do czytania kolejnego impulsu
      return;  // wracamy na początek pętli programu
    }
```

```
  }
  // Jeżeli tu doszliśmy, to znaczy że stan "1" z kodu powyżej uległ zmianie
  // przed końcem nadawania sygnału, i możemy zapisać jego czas trwania "wysokiTimer" do
  // tablicy "pulsy", w parze wartości numer "obecnyPuls", na pierwszej pozycji pary,
  // odpowiadającej wartości HIGH:
  pulsy[obecnyPuls][0] = wysokiTimer;

  // A teraz zrobimy ponownie to samo, ale dla sygnału "0":
  while(!(PIND & (B00000100))){ // wykrzyknik zamienia nam znak pinu na przeciwny
    niskiTimer++;
    delayMicroseconds(rozdzielczosc);

    if ((niskiTimer >= dlugoscSygnalu) && ( obecnyPuls != 0)){
      drukujPulsy();
      obecnyPuls = 0;
      return;
    }
  }
  pulsy[obecnyPuls][1] = niskiTimer;

  // przeszliśmy całą pętle, i udało nam się odczytać dwie wartości dla 0 i jedynki,
  // więc przechodzimy do następnej pary pulsów
  obecnyPuls++;
}

// definiujemy funkcję drukującą pulsy
void drukujPulsy(){
  Serial.print("Odbiór impulsów z pilota IR");
  Serial.print("Format: długość stanu HIGH, długość stanu LOW, w mikrosekundach");
  for ( uint8_t i = 0; i < obecnyPuls-1; i++){
    Serial.print(pulsy[i][0] * rozdzielczosc, DEC); // DEC drukuje w liczbach dziesiętnych
  Serial.print(" , ");
  Serial.print(pulsy[i][1] * rozdzielczosc, DEC);
```

```
  Serial.print("\n");
  }
}
```

Odbieramy dane z pilota – 15 minut

Kierunkujemy pilota na sensor IR i naciskamy dowolny przycisk. Sprawdzamy, w jaki sposób wyglądają informację zwracane przez płytkę za pośrednictwem Serial Monitor.

Jeżeli w pomieszczeniu są dostępne inne piloty IR, na przykład od rzutnika, tablicy interaktywnej lub telewizora, możemy nakierować je na naszą płytkę i zobaczyć, jakie dane będą się wyświetlały po naciśnięciu przycisków.

Na bazie danych z serial monitor można oszacować dla pilota:

- jaka jest jego częstotliwość nadawania,
- jakie są jego długości sygnałów,
- przetłumaczyć wartości czasowe na wartości w bitach.

Zadanie dodatkowe

Dodać układ z nadajnikiem IR, oraz tak zaprogramować płytkę, by wysyłała odebrany sygnał IR dalej lub sama generowała sygnał.

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 I-III mogą wykonać zadanie przy użyciu digitalRead/digitalWrite, wtedy brak responsywności kodu może być podstawą do teoretycznego omówienia bezpośredniej kontroli pinów, oraz zagłębienie się w szczegóły funkcji digitalRead. Klasy IV-VI 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 oraz nadajnika i odbiornika podczerwieni. Zadanie polega na podłączeniu płytki z czujnikiem IR, objaśnieniu

układu własnymi słowami, oraz wyjaśnieniu sposobu programowania układu, w szczególności funkcji PIN i DDR, oraz dlaczego nie możemy wykorzystać do odczytu czujnika funkcji wbudowanych.

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

Oficjalna dokumentacja manipulacji portów Arduino:
https://www.arduino.cc/en/Reference/PortManipulation

Szybkie pisanie na porty Arduino i porównanie z funkcjami wbudowanymi
http://www.billporter.info/2010/08/18/ready-set-oscillate-the-fastest-way-to-change-arduino-pins/

Część kodu bazowana na dokumencie Adafruit, w domenie publicznej:
https://learn.adafruit.com/ir-sensor/using-an-ir-sensor

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).