

Jon Hoffman

Zostań mistrzem Arduino

Projekty
dla początkujących
i zaawansowanych

Helion 

Packt 

Tytuł oryginału: Mastering Arduino: A project-based approach to electronics, circuits, and programming

Tłumaczenie: Radosław Meryk

ISBN: 978-83-283-5522-4

Copyright © Packt Publishing 2018. First published in the English language under the title ‘Mastering Arduino – (9781788830584)’

Polish edition copyright © 2019 by Helion SA

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Helion SA dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Helion SA nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/zomiar>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

O autorze	13
O recenzentach	14
Przedmowa	15
Rozdział 1. Arduino	19
Historia Arduino	20
Czym jest Arduino?	22
Odkrywamy płytke Arduino UNO R3	22
Zasilanie Arduino	24
Używanie do zasilania Arduino pinów VIN/GND	24
Używanie do zasilania Arduino wejścia zasilania DC	25
Używanie do zasilania Arduino złącza USB	26
Płytki Arduino shield	27
Piny na płytce Arduino	28
Piny cyfrowe	29
Analogowe piny wejściowe	29
Piny PWM	29
Piny zasilania	30
Piny transmisji szeregowej	30
Piny SPI	30
Odmiany płytki Arduino	31
Arduino Micro	31
Arduino Mega 2560	32
Lilypad	32
Arduino Nano	33
Zamienniki płytek Arduino	33
Podsumowanie	36

Rozdział 2. Podstawy elektroniki	37
Elektroniczne bloki konstrukcyjne	38
Zasilanie	38
Wejście	39
Wyjście	39
Układ sterujący	39
Multimetr	40
Elementy elektroniczne	43
Rezystor	43
Potencjometr	43
Przełączniki	44
Tranzystor	45
Dioda LED	46
Kondensator	46
Układ scalony	47
Czym jest energia elektryczna?	48
Prąd	49
Napięcie	49
Oporność	50
Prawo Ohma	51
Czym jest moc?	51
Kody kolorów rezystorów	52
Podsumowanie	53
Rozdział 3. Schematy obwodów	55
Czym jest obwód?	55
Fritzing	58
Diagramy Fritzing	59
Schematy symboliczne	60
Obwody równoległe i szeregowe	61
Obwody szeregowe	61
Rezystancja	62
Napięcie	62
Prąd	62
Obwody równoległe	62
Rezystancja	63
Napięcie	63
Prąd	63
Spadek napięcia	64
Zapalanie diody LED	64
Podsumowanie	66
Rozdział 4. Podstawy prototypowania	67
Tworzenie miejsca do pracy	68
Używanie bezlutowej płytki prototypowej	69
Kable Dupont	73

Prototypowanie	73
Cztery bloki budulcowe projektu elektronicznego	75
Tworzenie schematu	76
Budowa prototypu	76
Pierwszy prototyp	76
Podsumowanie	79
Rozdział 5. Arduino IDE	81
Szkic Arduino	81
Arduino IDE	82
Odkrywanie IDE	83
Konfigurowanie Arduino w środowisku IDE	84
Arduino Web Editor	86
Odkrywanie edytora	87
Konfigurowanie Arduino w środowisku Arduino Web Editor	88
Przykłady	88
Biblioteki Arduino	92
Monitor szeregowy	95
Witaj świecie!	98
Echo	99
Podsumowanie	101
Rozdział 6. Programowanie Arduino — podstawy	103
Nawiasy klamrowe	104
Średniki	104
Komentarze	104
Zmienne	105
Typy danych	105
Boolean	106
Byte	106
Integer	106
Long	107
Double i float	107
Char	107
Tablice	108
Tablice znaków	110
Stałe	110
Funkcje arytmetyczne	111
Operatory porównania	112
Operatory logiczne	113
Rzutowanie	113
Podejmowanie decyzji	114
Pętle	116
Funkcje	117
Podsumowanie	119

Rozdział 7. Programowanie Arduino — więcej niż podstawy	121
Ustawianie trybu pinu cyfrowego	122
Zapis pinów cyfrowych	123
Odczyt pinów cyfrowych	124
Zapis pinów analogowych	125
Odczyt pinów analogowych	126
Struktury	127
Unie	129
Dodawanie zakładek	130
Praca z zakładkami	133
Programowanie obiektowe	134
Biblioteka String	137
Podsumowanie	139
Rozdział 8. Czujnik ruchu	141
Wprowadzenie	141
Potrzebne komponenty	144
Schematy połączeń	144
Kod	146
Uruchamianie projektu	147
Zadanie dodatkowe	147
Podsumowanie	148
Rozdział 9. Czujniki środowiskowe	149
Wprowadzenie	149
Potrzebne komponenty	152
Schematy połączeń	152
Kod	152
Uruchamianie projektu	159
Zadanie dodatkowe	160
Podsumowanie	160
Rozdział 10. Unikanie przeszkód i wykrywanie kolizji	161
Wprowadzenie	161
Czujnik zderzeniowy	162
Czujnik unikania przeszkód	163
Ultradźwiękowy wykrywacz zasięgu	164
Potrzebne komponenty	165
Schematy połączeń	165
Kod	166
Uruchamianie projektu	168
Zadanie dodatkowe	169
Podsumowanie	170

Rozdział 11. Zabawa ze światłem	171
Wprowadzenie	171
Potrzebne komponenty	173
Schematy połączeń	174
Kod	175
Dioda LED RGB	175
NeoPixel shield	176
Uruchamianie projektu	179
Zadanie dodatkowe	179
Podsumowanie	180
Rozdział 12. Zabawa z dźwiękiem	181
Wprowadzenie	181
Potrzebne komponenty	183
Schematy połączeń	183
Kod	184
Korzystanie z funkcji tone()	184
Odtwarzanie dzwonka w formacie RTTTL	187
Zadanie dodatkowe	189
Podsumowanie	190
Rozdział 13. Korzystanie z wyświetlaczy LCD	191
Wprowadzenie	191
Potrzebne komponenty	193
Schematy połączeń	193
Kod	195
Rysowanie linii	196
Wyświetlanie tekstu	196
Obracanie tekstu	198
Podstawowe kształty	198
Figura wypełniona	199
Prostokąt	200
Wypełniony prostokąt	200
Zaokrąglony prostokąt	201
Wypełniony zaokrąglony prostokąt	201
Zadanie dodatkowe	202
Podsumowanie	202
Rozdział 14. Rozpoznawanie mowy i synteza głosu	203
Wprowadzenie	203
Potrzebne komponenty	206
Schematy połączeń	206
Kod	207
Uruchamianie projektu	209
Zadanie dodatkowe	209
Podsumowanie	210

Rozdział 15. Silniki prądu stałego i ich sterowniki	211
Wprowadzenie	211
Potrzebne komponenty	215
Schematy połączeń	215
Kod	217
Uruchamianie projektu	218
Zadanie dodatkowe	219
Podsumowanie	219
Rozdział 16. Serwosilniki	221
Wprowadzenie	221
Potrzebne komponenty	223
Schematy połączeń	223
Kod	224
Zadanie dodatkowe	226
Podsumowanie	226
Rozdział 17. Korzystanie z przełączników	227
Wprowadzenie	227
Potrzebne komponenty	230
Schematy połączeń	231
Kod	232
Zadanie dodatkowe	233
Podsumowanie	233
Rozdział 18. Zdalne sterowanie Arduino	235
Wprowadzenie	235
Potrzebne komponenty	238
Schematy połączeń	239
Kod	239
Zadanie dodatkowe	243
Podsumowanie	244
Rozdział 19. Tworzenie robota	245
Wprowadzenie	245
Podwozie i ruch	246
Silniki i zasilanie	249
Robot autonomiczny — unikanie przeszkód i wykrywanie kolizji	252
Zdalne sterowanie robotem	255
Udzielanie informacji zwrotnych użytkownikom	255
Ruch obrotowy	256
Projekty nierobotyczne	257
Stacja pogodowa	257
Inteligentny termostat	257
Czujnik zbliżania się	257
Zadanie dodatkowe	258
Podsumowanie	258

Rozdział 20. Bluetooth LE	259
Wprowadzenie	259
Technologie radiowe Bluetooth LE	261
Topologie sieciowe	263
Profile Bluetooth LE	265
Moduł Bluetooth HM-10	270
Potrzebne komponenty	271
Schematy połączeń	272
Projekt 1. Transmisja szeregową	273
Polecenie testowe	276
Zapytanie o wersję oprogramowania	276
Przywracanie ustawień fabrycznych	276
Restart modułu	277
Zapytanie o adres MAC (Media Access Control)	277
Ustawianie nazwy	277
Zapytanie o nazwę	277
Ustawianie interwału rozgłaszania	277
Zapytanie o interwał rozgłaszania	278
Ustawianie typu rozgłaszania	278
Zapytanie o typ rozgłaszania	278
Ustawianie szybkości transmisji	278
Zapytanie o szybkość transmisji	279
Ustawianie identyfikatora cechy	279
Ustawianie identyfikatora usługi	279
Zapytanie o identyfikator usługi	279
Ustawianie roli	279
Zapytanie o rolę	279
Wyczyszczenie informacji o ostatnio połączonym urządzeniu	280
Próba nawiązania połączenia z ostatnio połączonym urządzeniem	280
Próba nawiązania połączenia z adresem	280
Ustawianie kodu PIN	280
Zapytanie o kod PIN	281
Ustawianie mocy modułu	281
Zapytanie o moc modułu	281
Ustawianie trybu łączenia	281
Zapytanie o tryb łączenia	281
Ustawianie powiadomień	282
Zapytanie o ustawianie powiadomień	282
Projekt 2. Sterowanie diodą LED	286
Projekt 3. Czujnik środowiskowy	288
Co nowego w Bluetooth 4.1, 4.2 i 5.0?	292
Bluetooth 4.1	292
Bluetooth 4.2	292
Bluetooth 5.0	293
Bluetooth mesh	293
Zadanie dodatkowe	293
Podsumowanie	294

Rozdział 21. Bluetooth Classic	295
Wprowadzenie	295
Radio Bluetooth	297
Topologia sieci	298
Potrzebne komponenty	299
Schematy połączeń	299
Projekt 1. Konfigurowanie modułów Bluetooth	301
Polecenie testowe	304
Reset urządzenia	304
Zapytanie o oprogramowanie firmware	304
Przywracanie ustawień domyślnych	305
Zapytanie o adres modułu	305
Ustawianie trybu modułu i zapytanie o tryb modułu	305
Ustawianie parametrów UART i zapytanie o te parametry	305
Ustawianie trybu połączenia i zapytanie o ten tryb	306
Ustawianie adresu powiązania i zapytanie o ten adres	306
Projekt 2. Połączenie szeregowo, wysyłanie danych	309
Projekt 3. Zdalne sterowanie joystickiem	312
Podsumowanie	316
Skorowidz	317

Unikanie przeszkód i wykrywanie kolizji

Jeśli tworzysz autonomicznego robota, który musi unikać przeszkód, zdalnie sterowany samochód, który musi wykryć, kiedy w coś uderzy, lub nawet drukarkę 3D, która musi wiedzieć, kiedy głowice drukujące osiągną granicę obszaru drukowania, w Twoim projekcie musi się znaleźć jakiś system unikania przeszkód lub wykrywania kolizji. W niniejszym rozdziale przyjrzymy się kilku czujnikom, które można wykorzystać do wykrywania przeszkód, oraz kilku systemom wykrywania kolizji.

W tym rozdziale zapoznasz się z następującymi zagadnieniami:

- Jak korzystać z czujnika zderzeniowego?
- Jak korzystać z czujnika na podczerwień do unikania przeszkód?
- Jak korzystać z ultradźwiękowego dalmierza?

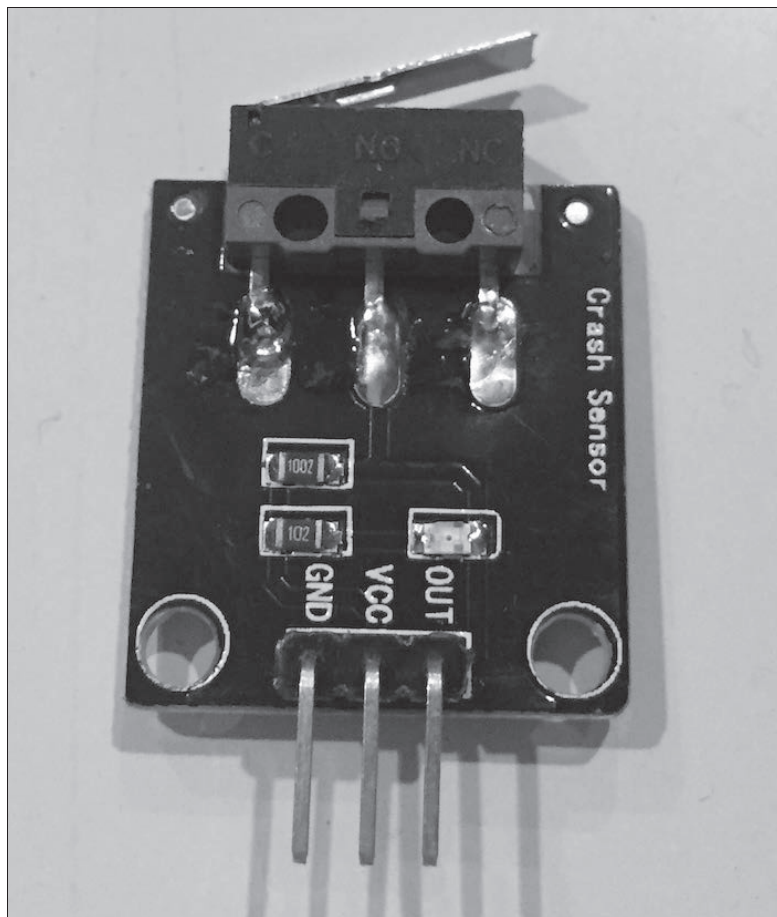
Wprowadzenie

W tym rozdziale przyjrzymy się trzem czujnikom, których można użyć, aby dodać do projektów funkcjonalność unikania przeszkód i (lub) wykrywania kolizji. Omówimy następujące czujniki:

- **Czujniki zderzeniowe:** służą do wykrywania kolizji, a także jako wyłączniki krańcowe dla drukarek 3D.
- **Czujniki na podczerwień do unikania przeszkód:** służą do unikania przeszkód w robotyce.
- **Ultradźwiękowe wykrywacze zasięgu:** służą do unikania przeszkód w robotyce. Mają także wiele innych zastosowań komercyjnych i wojskowych.

Czujnik zderzeniowy

Czujnik zderzeniowy jest prostym przełącznikiem wyposażonym w rodzaj wzmacniacza, dzięki któremu zyskuje duży obszar do wykrywania kolizji. Prosty czujnik zderzeniowy pokazano na poniższym zdjęciu:



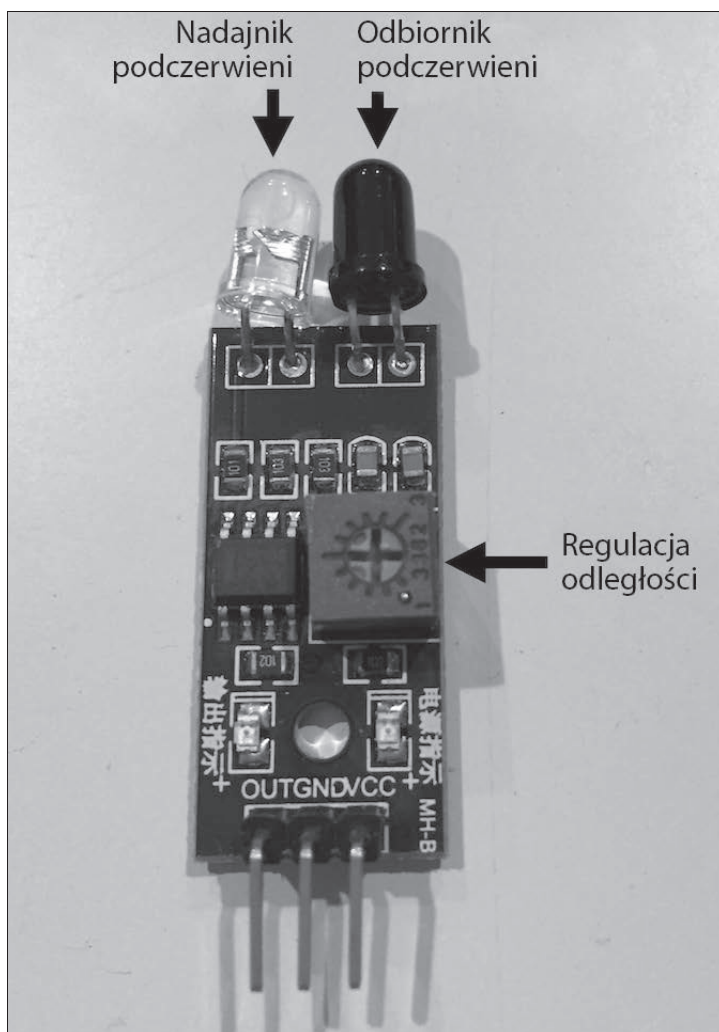
Czujnik zderzeniowy pokazany na zdjęciu wymaga prostego przełącznika mechanicznego, takiego jak te, które są stosowane do ograniczników w drukarkach 3D. Taki przełącznik jest zamocowany na końcu płytki drukowanej. Dzięki temu można go łatwo zamontować na podwoziu robota lub innych powierzchniach. Idea działania czujnika zderzeniowego bazuje na założeniu, że gdy czujnik w coś uderzy, to załącza przełącznik.

Czujnik zderzeniowy ma trzy piny, wyraźnie oznaczone jako **GND**, **VCC** i **OUT**. Pin **GND** należy podłączyć do szyny masy, natomiast pin **VCC** — do szyny zasilania płytki prototypowej. Pin **OUT** można połączyć bezpośrednio z wyjściem cyfrowym na Arduino za pośrednictwem rezystora podciągającego 4,7 kΩ.

Czujnik na podczerwień do unikania przeszkód składa się z nadajnika podczerwieni, odbiornika podczerwieni i potencjometru regulującego odległość, na jakiej czujnik potrafi wykrywać przeszkody. Czujnik unikania przeszkód, który został użyty w projekcie zaprezentowanym w tym rozdziale, przedstawiono na kolejnych zdjęciach.

Czujnik unikania przeszkód

Emiter na czujniku na podczerwień do unikania przeszkód emituje promieniowanie podczerwone i jeśli przeszkoda znajduje się przed czujnikiem, część promieniowania jest odbijana wstecz i odbierana przez odbiornik. Jeśli przed czujnikiem nie znajduje się żaden przedmiot, promieniowanie się rozproszy, a odbiornik niczego nie odbierze.

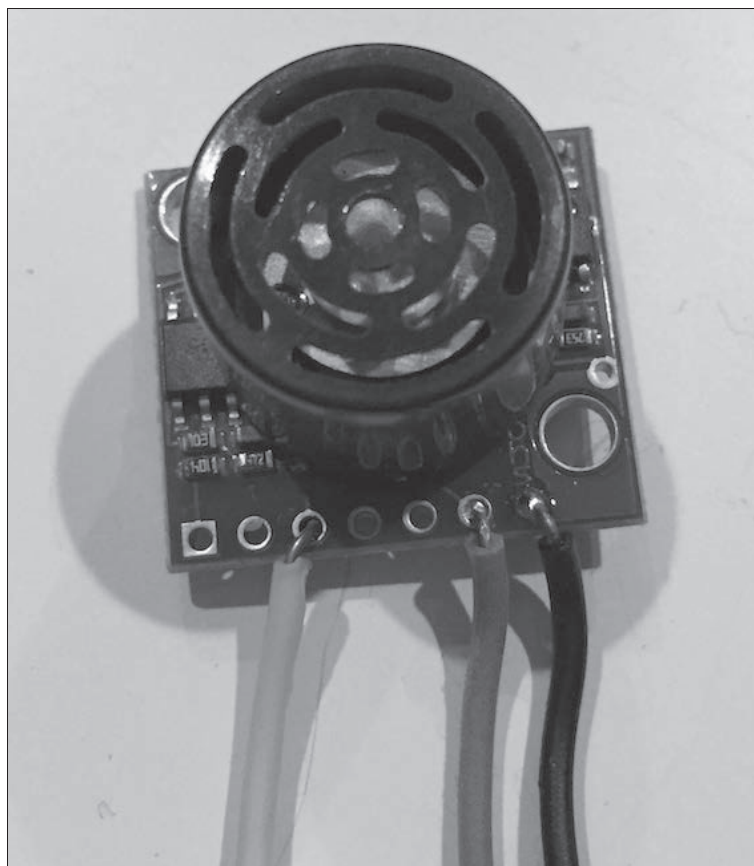


Piny na czujniku są wyraźnie oznaczone — od lewej do prawej — **OUT**, **GND** i **VCC**. Pin **GND** jest podłączony do szyny masy, natomiast pin **VCC** — do linii zasilania na płytce prototypowej. Pin **OUT** jest podłączony bezpośrednio do cyfrowego wyjścia Arduino. Jeżeli sygnał na pinie **OUT** jest w stanie **LOW**, to oznacza, że wykryto obiekt. Jeśli wyjście jest w stanie **HIGH**, to oznacza, że nie wykryto żadnego obiektu.

Regulator odległości dostosuje odległość wykrywaną przez czujnik. Jeśli obracamy regulator w kierunku przeciwnym do ruchu wskazówek zegara, zmniejszamy odległość, natomiast jeśli obracamy go w kierunku zgodnym z ruchem wskazówek zegara, to ją zwiększamy. Czujnik wykrywa obiekty znajdujące się w odległości od 2 cm do 30 cm.

Ultradźwiękowy wykrywacz zasięgu

Trzecim czujnikiem, którego będziemy używać, jest dalmierz ultradźwiękowy MaxSonar EZ1. Ten czujnik jest jednym z moich ulubionych. Stosowałem go w prawie wszystkich autonomicznych robotach, które budowałem, w celu określania odległości od pobliskich obiektów. Dalmierz ultradźwiękowy EZ1 przedstawiono na poniższym zdjęciu:



Na potrzeby przykładu pokazanego w tym rozdziale będziemy używać pinów czujnika 3, 6 i 7. Pin 3 służy jako analogowe wyjście, pin 6 podłączamy do zasilania (VCC), a pin 7 — do masy. Piny 4 i 5 obsługują linie RX i TX komunikacji szeregowej, natomiast pin 2 to wyjście szerokości impulsu. W projekcie zaprezentowanym w tym rozdziale nie będziemy jednak używać tych wyjść.

Dalmierz ultradźwiękowy działa poprzez wysyłanie ultradźwiękowego impulsu w określonym kierunku. Jeśli na ścieżce impulsu jest obiekt, to sygnał jest odbijany w postaci echa. Czujnik określa odległość od obiektu, mierząc czas potrzebny do odebrania echa.

Czujnik ultradźwiękowy EZ1 potrafi wykrywać i mierzyć odległość od obiektów w zakresie od 0 do 6,45 metra (254 cale). Ten czujnik praktycznie nie ma martwej strefy i wykrywa obiekty znajdujące się tuż przed nim.

Potrzebne komponenty

Do wykonania projektu w tym rozdziale będziemy potrzebować następujących komponentów:

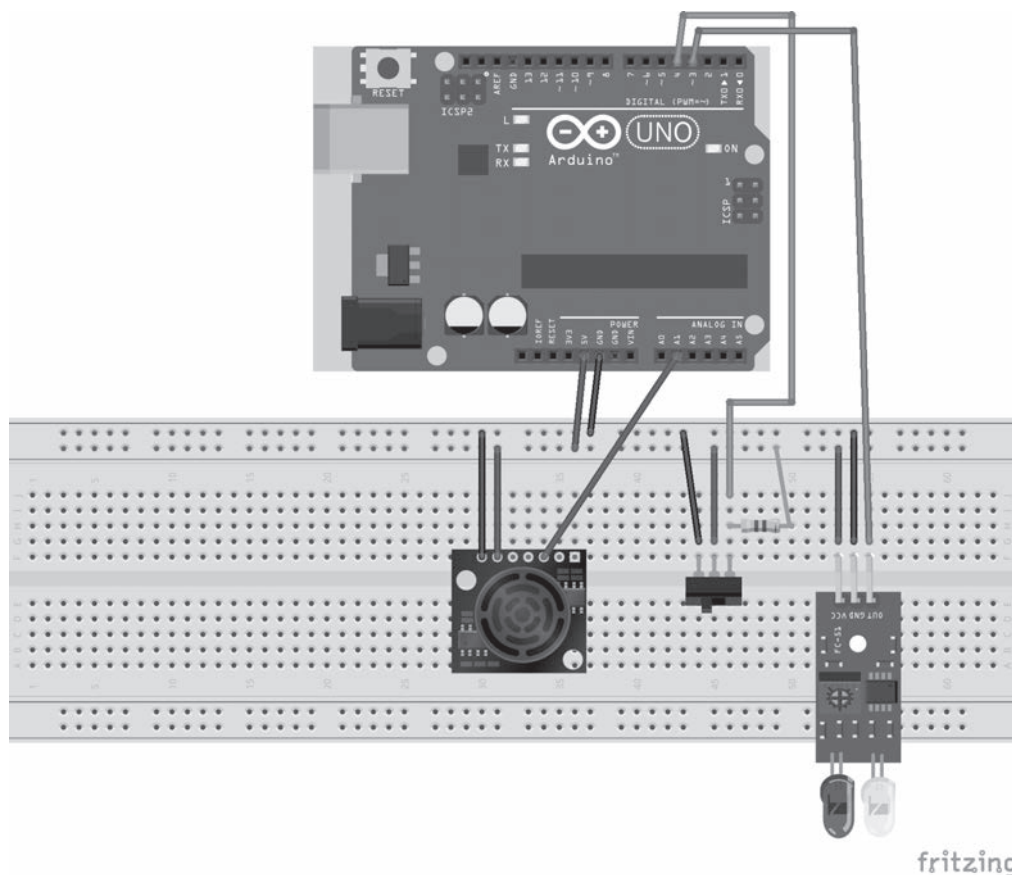
- jednej płytki Arduino Uno lub kompatybilnej,
- jednego czujnika zderzeniowego,
- jednego czujnika na podczerwień do unikania przeszkód,
- jednego czujnika ultradźwiękowego EZ1,
- jednego rezystora 4,7 k Ω ,
- przewodów łączących,
- jednej płytki prototypowej.

Schematy połączeń

Schemat Fritzing dla tego projektu przedstawiono na rysunku na następnej stronie.

Środkowy czujnik na schemacie reprezentuje czujnik zderzeniowy. Użyłem tego elementu ze względu na to, że w programie Fritzing nie ma elementu reprezentującego czujnik zderzeniowy. Przełącznik na schemacie ma taki sam układ pinów jak czujnik zderzeniowy zaprezentowany we wcześniejszej części tego rozdziału.

Jak widać na schemacie, wszystkie piny masy czujników są połączone z szyną masy na płytce prototypowej, a wszystkie piny VCC czujników są podłączone do szyny zasilania na płytce prototypowej.



Analogowy sygnał wyjściowy ultradźwiękowego czujnika EZ1 jest podłączony do analogowego pinu A1 na płytce Arduino, czujnik zderzeniowy — do cyfrowego pinu 3, a czujnik podczerwieni — do cyfrowego pinu 2. Do czujnika zderzeniowego podłączono także rezystor podciągający 4,7 kΩ. Teraz, gdy podłączyliśmy czujniki do Arduino, przyjrzyjmy się kodowi dla tego projektu.

Kod

Kod zaczyna się od trzech makr definiujących piny, do których podłączone są trzy czujniki. Makra mają taką postać:

```
#define COLLISION_SWITCH 4
#define IR_SENSOR 3
#define RANGE_SENSOR A1
```

Na podstawie powyższych makr widzimy, że czujnik zderzeniowy jest podłączony do cyfrowego pinu 4, czujnik podczerwieni — do cyfrowego pinu 3, a dalmierz ultradźwiękowy — do analogowego pinu 1. Teraz trzeba ustawić tryb dla dwóch pinów cyfrowych, z których korzystamy, oraz zainicjować monitor szeregowy. Możemy to zrobić, dodając do funkcji `setup()` następujący kod:

```
Serial.begin(9600);
pinMode(COLLISION_SWITCH, INPUT);
pinMode(11, INPUT);
```

Zaczynamy od zainicjowania monitora szeregowego. Następnie konfigurujemy piny czujnika zderzeniowego i czujnika podczerwieni jako wejścia. Teraz musimy dodać kod do funkcji `loop()`, której zadaniem będzie odczyt wartości z czujników. Zaczniemy od kodu odpowiedzialnego za odczyt czujnika zderzeniowego:

```
int collisionValue = digitalRead(COLLISION_SWITCH);
if (isnan(collisionValue)) {
    Serial.println(" Błąd odczytu czujnika zderzeniowego");
    return;
}

if (collisionValue == LOW) {
    Serial.println("Wykryto kolizję");
}
```

Ten kod rozpoczyna się od użycia funkcji `digitalRead()` w celu odczytania pinu, do którego jest podłączony czujnik zderzeniowy. Następnie wykorzystano funkcję `isnan()` w celu sprawdzenia, czy funkcja `digitalRead()` zwróciła poprawną wartość. Jeśli wartość zwrócona przez funkcję jest niepoprawna (nie jest liczbą), to na konsoli szeregowej wyświetlamy komunikat o błędzie, a następnie wywołujemy instrukcję `return`, aby wyjść z pętli.

Jeśli wartość zwrócona przez funkcję `digitalRead()` jest prawidłowa, to sprawdzamy, czy wartość reprezentuje stan `LOW`. Jeśli tak, to wykryliśmy przeszkodę. Sygnalizujemy to wysłaniem komunikatu do konsoli szeregowej. Teraz dodajemy kod obsługi czujnika podczerwieni:

```
int irValue = digitalRead(IR_SENSOR);
if (isnan(irValue)) {
    Serial.println(" Niepowodzenie odczytu czujnika podczerwieni");
    return;
}
if (irValue == LOW) {
    Serial.println("Wykryto promieniowanie IR");
}
```

Ten kod jest dokładnie taki sam jak kod obsługi czujnika zderzeniowego, z wyjątkiem tego, że odczytaliśmy pin czujnika podczerwieni i sprawdziliśmy odczytaną z niego wartość. Dodajmy teraz kod obsługi ultradźwiękowego czujnika odległości:

```
int anVolt = analogRead(RANGE_SENSOR);
if (isnan(anVolt)) {
    Serial.println(" Błąd odczytu czujnika odległości");
    return;
}
```

```

}
int mm = anVolt*5;
float inch = mm/25.4;
Serial.println(mm);
Serial.print("W milimetrach: ");
Serial.println(mm);
Serial.print("W calach: ");
Serial.println(inch);
Serial.println("-----");
delay(1000);

```

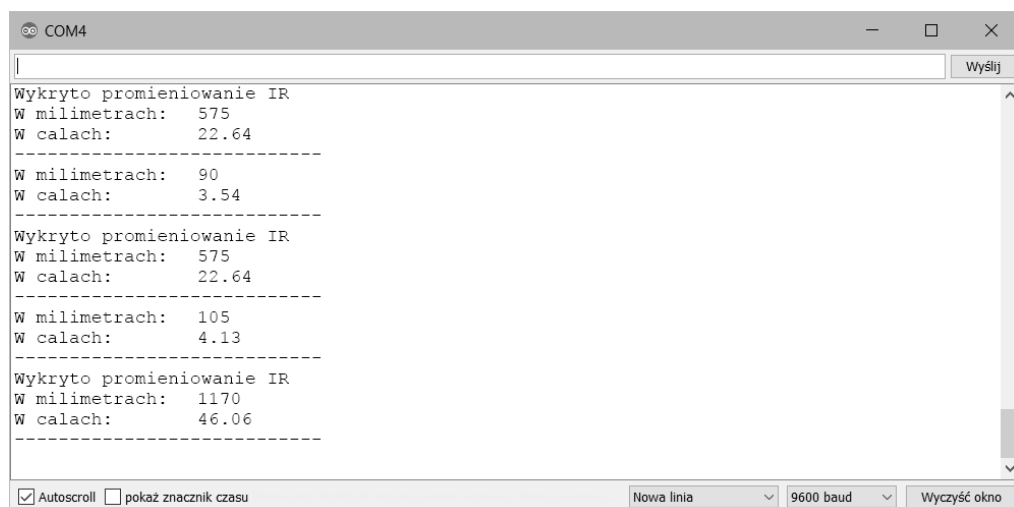
Powyższy kod rozpoczyna się od użycia funkcji `analogRead()` w celu odczytania pinu, do którego jest podłączony dalmierz ultradźwiękowy. W dalszej kolejności używamy funkcji `isnan()`, aby sprawdzić, czy została zwrócona poprawna wartość.

Następnie obliczamy odległość od obiektu w milimetrach i w calach. Liczby użyte w obliczeniach pochodzą z arkusza danych czujnika. Mogą się różnić w zależności od używanego modelu. Na końcu funkcji `loop()` wprowadzamy niewielkie opóźnienie, aby wstrzymać wykonywanie pętli.

Projekt jest teraz gotowy do uruchomienia.

Uruchamianie projektu

Po uruchomieniu tego projektu powinniśmy zobaczyć wynik podobny do pokazanego na poniższym zrzucie ekranu:



Na rzucie ekranu można zauważyć, że obiekt dwukrotnie wykrył kolizję za pomocą czujnika podczerwieni (co widać po komunikacie „Wykryto promieniowanie IR” na konsoli szeregowej). Można zauważyć także, że raz zadziałał czujnik zderzeniowy (komunikat „Wykryto kolizję” na konsoli szeregowej). Widzimy na rzucie również odległość, jaką zmierzył dalmierz ultradźwiękowy od najbliższego obiektu.

Zadanie dodatkowe

Zadanie dodatkowe w tym rozdziale będzie się różniło od większości zadań. Tak naprawdę nie ma w nim projektu do wykonania. Zamiast tego jest zadanie wymagające myślenia. Wyzwanie polega na tym, aby zastanowić się, jak można zastosować wszystkie trzy czujniki razem do stworzenia autonomicznego robota. Aby to zrobić, zastanów się, jak działają poszczególne czujniki:

1. **Czujnik zderzeniowy:** cyfrowy czujnik, który jest wyzwalany w przypadku, gdy na coś wpadnie.
2. **Czujnik podczerwieni:** cyfrowy czujnik, który zostaje wyzwolony, gdy w pobliżu znajdzie się jakiś obiekt.
3. **Ultradźwiękowy wykrywacz zasięgu:** analogowy czujnik używany do wykrywania odległości obiektu od czujnika.

Oto odpowiedzi:

Ultradźwiękowe czujniki odległości są zdecydowanie najdroższe, więc zwykle używam tylko dwóch czujników, które montuję w przedniej części robota. Są one używane przez robota do obchodzenia przeszkód. Dzięki możliwości określenia, jak daleko przeszkoda znajduje się od robota, możemy wyposażać go w logikę potrzebną do podjęcia decyzji, kiedy należy skręcić, a także — w przypadku zastosowania dwóch czujników ultradźwiękowych — logikę pozwalającą zdecydować, w którą stronę skręcić. Czujników ultradźwiękowych możemy również użyć do stworzenia mapy pomieszczenia.

Czujniki podczerwieni są niedrogie i mogą być używane na bokach i z tyłu robota, aby nie wpadł on na przeszkodę podczas obracania lub cofania. Ponieważ są dużo tańsze niż czujniki ultradźwiękowe, możemy użyć wielu czujników podczerwieni. Dzięki temu możemy pokryć pełny obszar wokół robota. Moglibyśmy również zastosować czujniki podczerwieni w podstawie, aby upewnić się, czy robot nie odjeżdża z półki.

Czujniki zderzeniowe są również bardzo tanie i mogą być używane dookoła robota do obsługi zderzeń z obiektami, których nie wykryły czujniki ultradźwiękowe lub czujniki na podczerwień. Największy problem z czujnikami ultradźwiękowymi i czujnikami podczerwieni jest związany z wysokością ich umieszczenia na robocie. Jeśli są zamontowane zbyt wysoko, mogą mieć problemy z wykrywaniem przeszkód znajdujących się nisko nad ziemią. Do ich wykrywania można użyć czujników zderzeniowych.

Podsumowanie

W tym rozdziale omówiliśmy kilka czujników, które można wykorzystać do unikania przeszkód oraz wykrywania kolizji. Czujnik zderzeniowy jest czujnikiem cyfrowym, którego można użyć do określenia, kiedy urządzenie w coś uderzy. Czujnik na podczerwień do unikania przeszkód jest również czujnikiem cyfrowym — rozpoznaje on, kiedy czujnik znajduje się w pewnej odległości od przeszkody. Ultradźwiękowy czujnik odległości jest czujnikiem analogowym, który można wykorzystać do określenia odległości od przeszkody.

W następnym rozdziale przyjrzymy się różnym typom diod LED i zobaczymy, jak możemy je wykorzystać w naszych projektach.

Skorowidz

A

- analogowe piny wejściowe, 29
- aplikacja echa, 99
- Arduino, 22
 - IDE, 82
 - Mega 2560, 32
 - Micro, 31
 - Nano, 33
 - Web Editor, 86
- ATmega328, 24

B

- bezlutowa płytka prototypowa, 69
- biblioteka
 - Arduino servo, 221
 - NeoPixel Adafruit, 176
 - String, 137
- biblioteki Arduino, 92
- bloki
 - budulcowe projektu, 75
 - konstrukcyjne, 38
- Bluetooth 2.0 + EDR, 296
- Bluetooth 2.1 + EDR, 296
- Bluetooth 3.0 + HS, 297
- Bluetooth 4.1, 292
- Bluetooth 4.2, 292
- Bluetooth 5.0, 293
- Bluetooth Classic, 295
 - komponenty, 299
 - konfigurowanie modułów, 301

- połączenie szeregowe, 309
- radio, 297
- schematy połączeń, 299
- topologia sieci, 298
- wysyłanie danych, 309
- zdalne sterowanie
 - joystickiem, 312

- Bluetooth LE, 259
 - czujnik środowiskowy, 288
 - komponenty, 271
 - moduł HM-10, 270
 - połączenia, 264
 - profile, 265
 - schematy połączeń, 272
 - sterowanie diodą LED, 286
 - topologie sieciowe, 263
 - transmisja szeregowa, 273
- Bluetooth mesh, 293
- brzęczyk piezoelektryczny, 181
- budowa prototypu, 76

C

- cykl roboczy, 29
- czujnik DHT-11, 207
- czujnik PIR, 141
- czujnik ruchu, 141
 - kod, 146
 - komponenty, 144
 - schematy połączeń, 144
 - uruchamianie projektu, 147
- czujnik środowiskowy, 149, 288
 - kod, 152
 - komponenty, 152

- schematy połączeń, 152
- uruchamianie projektu, 159
- czujnik unikania przeszkód, 163
- czujnik zbliżania się, 257
- czujnik zderzeniowy, 162

D

- diagramy Fritzing, 59
- dioda
 - LED, 46
 - LED RGB, 171, 175
- dodawanie zakładki, 130
- dźwięk, 181
 - kod, 184
 - komponenty, 183
 - odtwarzanie dzwonka, 187
 - pliki RTTTL, 187
 - schematy połączeń, 183

E

- EDR, Enhanced Data Rate, 296
- edytor, 87
- elementy elektroniczne, 43
- energia elektryczna, 48
- EOT, End Of Transmission, 309

F

format RTTTL, 187
 Fritzing, 58
 funkcja, 117
 analogRead(), 126
 analogWrite(), 125
 digitalRead(), 124
 digitalWrite(), 123
 drawPixel(), 196
 loop(), 310, 314
 pinMode(), 122
 setRotation(), 198
 setTextSize(), 197
 tone(), 184
 funkcje arytmetyczne, 111

G

gniazdo zasilania prądem
 stałym (DC), 23

H

historia Arduino, 20

I

iCSP dla ATmega328, 23
 IDE, 83
 IDE, Integrated Development
 Environments, 81
 instrukcja if, 114
 inteligentny termostat, 257

K

kable Dupont, 73
 kody kolorów rezystorów, 52
 komentarze, 104
 kondensator, 46
 konfigurowanie Arduino
 w środowisku Arduino Web
 Editor, 88
 w środowisku IDE, 84
 konfigurowanie modułów
 Bluetooth, 301
 konwerter analogowo-cyfrowy,
 ADC, 29

L

LAP, Lower Address, 298
 licencja GNU LGPL, 22
 Lilypad, 32

M

miejsce do pracy, 68
 moc, 51
 monitor szeregowy, 95
 MOVI, 203
 multimetr, 40

N

NAP, Non-Significant Address,
 298
 napięcie, 49
 nawiasy klamrowe, 104
 NeoPixel shield, 176

O

obracanie tekstu, 198
 obszar
 kodowania, 84, 87
 statusowy, 84, 87
 obwody równoległe, 61
 napięcie, 63
 prąd, 63
 rezystancja, 63
 obwody szeregowy, 61
 napięcie, 62
 prąd, 62
 rezystancja, 62
 obwód, 55
 odczyt pinów
 analogowych, 126
 cyfrowych, 124
 OOP, Object-oriented
 programming, 134
 operatory
 logiczne, 113
 porównania, 112
 oporność, 50
 OID, Organizationally
 Unique Identifier, 298

P

pasek
 menu, 83, 87
 poleceń, 83, 87
 pasywne czujniki
 podczzerwieni, 141
 pętla
 do-while, 117
 for, 116
 while, 116
 pierwszy
 prototyp, 76
 szkic, 98
 pilot
 na podczerwień, 236
 radiowy, 243
 piny, 28
 analogowe, 29
 cyfrowe, 29
 PWM, 29
 SPI, 30
 transmisji szeregowy, 30
 VIN/GND, 24
 zasilania, 30
 pliki RTTTL, 187
 płytka
 Arduino UNO R3, 22
 Arduino shield, 27
 NeoPixel, 172
 shield MOVI, 205
 polecenie testowe, 276, 304
 połączenie szeregowy,
 wysyłanie danych, 309
 port USB, 23
 potencjometr, 43
 prawo Ohma, 51
 prąd, 49
 stały (DC), 37
 zmienny (AC), 37
 profil
 atrybutów (GATT), 267
 dostępu (GAP), 265
 program Fritzing, 58
 programowanie, 103, 121
 obiektowe, OOP, 134
 prototypowanie, 73
 próba nawiązania połączenia
 z adresem, 280
 z urządzeniem, 280

przekazniki, 227
 etykieta NC, 228
 etykieta NO, 228
 kod, 232
 komponenty, 230
 schematy połączeń, 231

przełączniki, 44
 przycisk RESET, 23
 przywracanie ustawień
 domyślnych, 305
 fabrycznych, 276
 PWM, Pulse-Width
 Modulation, 29, 125

R

radio Bluetooth, 297
 regulator napięcia, 23
 reset urządzenia, 304
 restart modułu, 277
 rezystor, 43
 robot, 245
 informacje zwrotne, 255
 obrotowy, 256
 podwozie, 246
 ruch, 246
 silnik, 249
 unikanie przeszkód, 252
 wykrywanie kolizji, 252
 zasilanie, 249
 zdalne sterowanie, 255
 rozpoznawanie mowy, 203
 kod, 207
 komponenty, 206
 schematy połączeń, 206
 uruchamianie projektu, 209
 rysowanie linii, 196
 rzutowanie, 113

S

schematy
 obwodów, 55
 symboliczne, 60
 serwsilniki, 221
 kod, 224
 komponenty, 223
 schematy połączeń, 223
 silnik prądu stałego, 211
 kod, 217
 komponenty, 215

schematy połączeń, 215
 uruchamianie projektu, 218
 silniki, 249
 SOH, Start Of Heading, 309
 spadek napięcia, 64
 SPI, Serial Peripheral
 Interface, 30
 SSP, Simple Secure Pairing,
 296
 stacja pogodowa, 257
 stałe, 110
 sterowanie diodą LED, 286
 sterownik silnika L298, 213
 struktury, 127
 synteza głosu, 203
 szkie, 81

Ś

średniki, 104
 światło, 171
 dioda LED RGB, 171, 175
 kod, 175
 komponenty, 173
 płytka NeoPixel, 172
 schematy połączeń, 174
 uruchamianie projektu, 179

T

tablice, 108
 znaków, 110
 technologia
 EDR, 296
 SSP, 296
 termistor, 149
 topologie sieciowe, 263, 289
 transmisja szeregowa, 273
 tranzystor, 45
 tryb pinu cyfrowego, 122
 tworzenie
 robota, 245
 schematu, 76
 typ danych
 boolean, 106
 byte, 106
 char, 107
 double, 107
 float, 107
 integer, 106
 long, 107

U

UAP, Upper Address Part, 298
 układ
 ADC, 29
 L293D, 214
 scalony, 47
 sterujący, 39
 ultradźwiękowy wykrywacz
 zasięgu, 164
 unie, 129
 unikanie przeszkód, 161, 252
 czujnik unikania przeszkód,
 163
 czujnik zderzeniowy, 162
 kod, 166
 komponenty, 165
 schematy połączeń, 165
 ultradźwiękowy wykrywacz
 zasięgu, 164
 uruchamianie projektu, 168
 urządzenie na podczerwień,
 243
 ustawianie
 adresu powiązania, 306
 identyfikatora cechy, 279
 identyfikatora usługi, 279
 interwału rozgłaszania, 277
 kodu PIN, 280
 mocy modułu, 281
 nazwy, 277
 parametrów UART, 305
 powiadomień, 282
 roli, 279
 szybkości transmisji, 278
 trybu modułu, 305
 trybu połączenia, 281, 306
 typu rozgłaszania, 278

W

wejście, 39
 zasilania DC, 25
 wyczyszczenie informacji, 280
 wyjście, 39
 PWM, 23
 wykrywanie kolizji, 252, *Patrz
 także* unikanie przeszkód
 wyświetlacz LCD, 191
 figura wypełniona, 199
 kod, 195

wyświetlacz LCD
 komponenty, 193
 obracanie tekstu, 198
 podstawowe kształty, 198
 prostokąt, 200
 rysowanie linii, 196
 schematy połączeń, 193
 wypełniony prostokąt, 200
 wypełniony zaokrąglony
 prostokąt, 201
 wyświetlanie tekstu, 196
 zaokrąglony prostokąt, 201

Z

zakładki, 130, 133
 zamienniki płytek Arduino, 33
 zapalanie diody LED, 64
 zapis pinów
 analogowych, 125
 cyfrowych, 123

zapytanie
 o adres MAC, 277
 o adres modułu, 305
 o adres powiązania, 306
 o identyfikator usługi, 279
 o interwał rozgłaszania, 278
 o kod PIN, 281
 o moc modułu, 281
 o nazwę, 277
 o oprogramowanie
 firmware, 304
 o parametry UART, 305
 o rolę, 279
 o szybkość transmisji, 279
 o tryb modułu, 305
 o tryb połączenia, 281, 306
 o typ rozgłaszania, 278
 o ustawianie powiadomień,
 282
 o wersję oprogramowania,
 276

zasilanie, 24, 38
 silników, 251
 zdalne sterowanie, 235
 joystickiem, 312
 kod, 239
 komponenty, 238
 robotem, 255
 schematy połączeń, 239
 zewnętrzny reset, 24
 zintegrowane środowisko
 programistyczne, IDE, 81
 złącza
 analogowe, 24
 cyfrowe, 23
 ICSP dla USB, 23
 USB, 26
 zmienne, 105

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

Arduino już masz. Teraz potrzebujesz tylko wyobraźni!

Arduino od wielu lat cieszy się niesłabnącą popularnością wśród miłośników elektroniki, robotyki i tych, którzy lubią w domowym zaciszu zbudować sobie własny gadżet. Arduino może komunikować się z komputerem, posiada także porty służące do podłączania zewnętrznych elementów elektronicznych, takich jak silniki, przekaźniki, fotodiody, diody laserowe, głośniki, mikrofony itp. Środowisko służące do programowania Arduino jest dostępne za darmo. To wszystko sprawia, że platforma może posłużyć do budowy najprzeróżniejszych robotów, sterowników, czujników czy interfejsów do komunikacji z innymi urządzeniami.

Ta książka jest wszechstronnym przewodnikiem, dzięki któremu w pełni wykorzystasz Arduino. Z jej pomocą szybko zdobędziesz wiedzę o elementach elektroniki i programowania, aby wkrótce tworzyć zaawansowane projekty Arduino. Znajdziesz tu mnóstwo praktycznych wzorów i przykładów do ćwiczeń. Rozpoczniesz od podstaw elektroniki, dzięki którym zrozumiesz zagadnienia dotyczące komponentów, układów i prototypów. Następnie poznasz podstawy kodowania, dowiesz się, jak posługiwać się Arduino IDE, jak podłączyć Arduino do komputera i jak uruchamiać własne projekty. Kolejne rozdziały książki dotyczą małych projektów, dzięki którym nauczysz się wykorzystywać do budowania własnych urządzeń wyświetlacze LCD, silniki krokowe, syntezatory mowy, a nawet technologie bezprzewodowe!

W tej książce:

- podstawy teorii elektroniki i obwodów
- Arduino IDE i podstawowe operacje w języku C
- sterowanie diodami LED za pomocą kodu
- stosowanie silników krokowych w budowie robota
- zdalne sterowanie Arduino za pomocą RF i Bluetooth
- budowa generatora dźwięków z przyciskami

Jon Hoffman od ponad ćwierćwiecza zajmuje się administrowaniem systemami i sieciami oraz ich bezpieczeństwem, a także tworzeniem aplikacji i architekturą systemów. Od 2008 roku tworzy oprogramowanie na platformę iOS. Jego prawdziwą pasją jest podejmowanie wyzwań w zakresie technologii informatycznych i oczywiście pokonywanie napotkanych problemów.

Helion 	<i>Sprawdź nasze szkolenia!</i>  SZKOLENIA AKADEMIA IT & BUSINESS WWW.SZKOLENIA.HELION.PL	KOD KORZYŚCI <i>Sięgnij po więcej!</i>   ISBN 978-83-283-5522-4  9 788328 355224
 helion.pl		
 0 801 339900		
 0 601 339900		
INFORMATYKA W NAJLEPSZYM WYDANIU		Cena: 57,00 zł

Packt