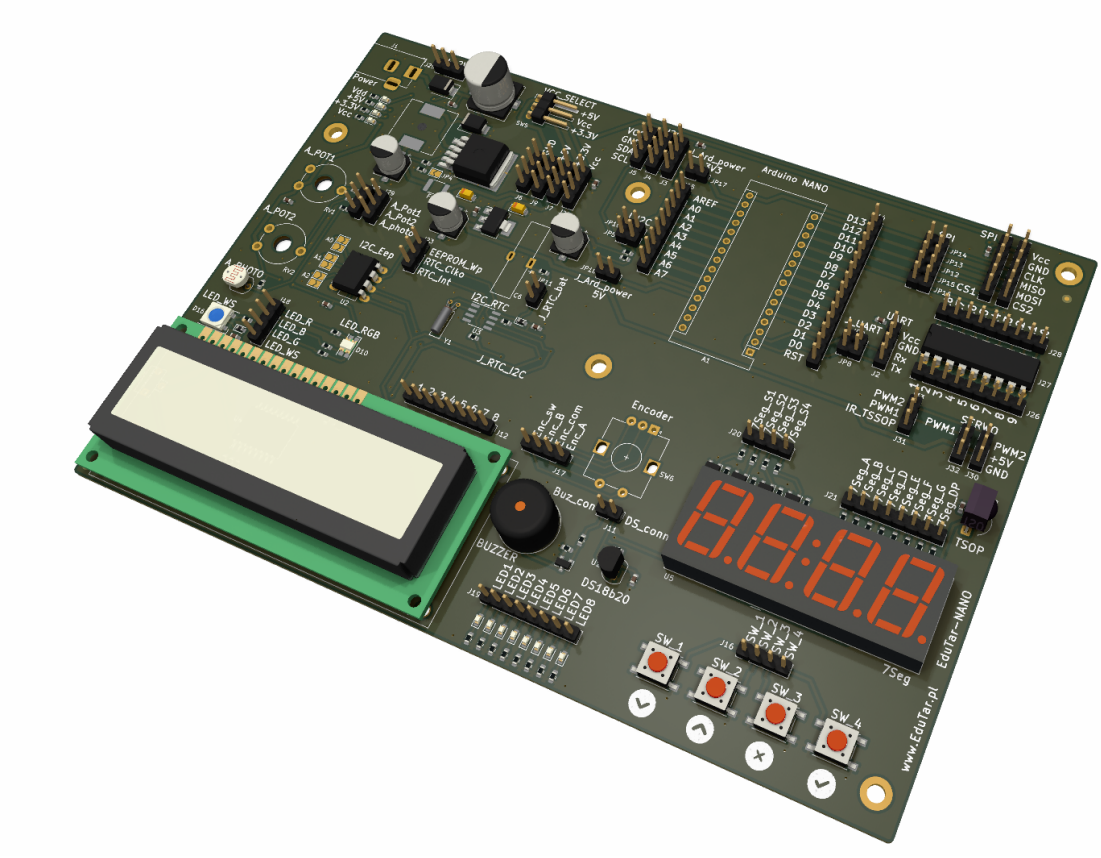
Płytka edukacyjna EduTar - NANO



Instrukcja dla wersji płytki V1.1

4 stycznia 2023

Spis treści

1 EduTar – płytka edukacyjna dla systemów embedded 3
1.1 Sekcja zasilania płytki 5
1.2 Wyświetlacz LCD 2x16 5
1.3 Wyświetlacz 7-segmentowy 6
1.4 Linijka diodowa 6
1.5 Cyfrowa dioda RGB–WS2812 7
1.6 Odbiornik podczerwieni–TSOP2236 8
1.7 Dioda RGB 8
1.8 Przyciski typu tact switch 10
1.9 Złącze do podłączenia modułu klawiatury zewnętrznej 10
1.10 Impulsator 11
1.11 Układ zadajnika sygnału analogowego 12
1.12 Analogowy czujnik jasności 12
1.13 Układ pamięci nieulotnej–EEPROM 12
1.14 Układ zegara czasu rzeczywistego - RTC 13
1.15 Cyfrowy czujnik temperatury–DS18b20 13
1.16 Generator sygnału akustycznego 14
1.17 Złącze rozszerzeń dla płytki Arduino Nano 16

2 Interfejsy komunikacyjne 16
2.1 UART 16
2.2 I2C 17
2.3 SPI 18

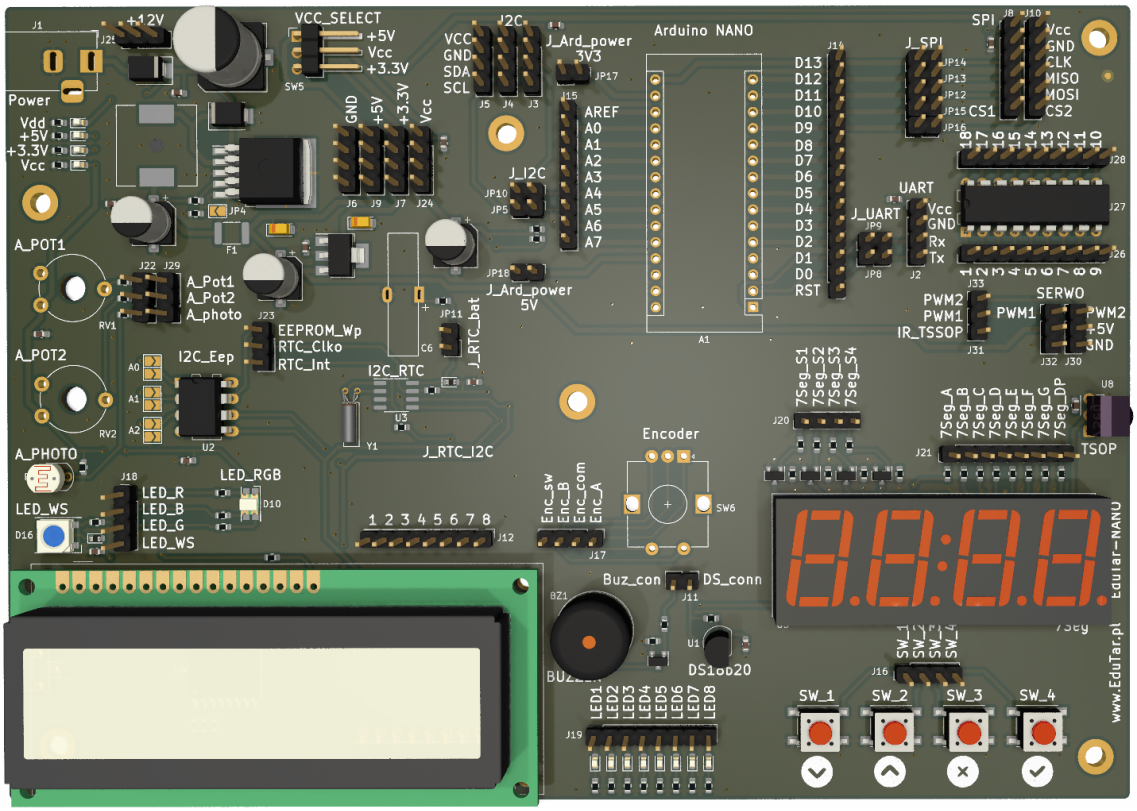
3 Moduł mikrokontrolera - płytka Arduino NANO 19
3.0.1 Instalacja sterowników Arduino NANO 19
3.1 Środowisko programistyczne - Arduino IDE 21
3.1.1 Programowanie w języku C - środowisko Ardunio . . . 23
3.1.2 Programowanie w języku asembler - środowisko Ardunio 25
3.1.3 Struktura programu Arduino 27
3.2 Środowisko programistyczne - PlatformIo 28

4 Moduł mikrokontrolera - płytka Nucleo 29

5 Moduł mikrokontrolera - płytka ESP32 30

1 EduTar – płytka edukacyjna dla systemów embedded

Płytka edukacyjna powstała jako platforma do prowadzenia zajęć dydaktycznych dla grup uczących się programować mikrokontrolery, jak również dla hobbystów chcących łatwo rozpocząć swoją przygodę ze światem embedded. Została wyposażona w zestaw podstawowych elementów elektronicznych, który zapewni pokrycie tematyczne dla kursów podstawowych i zaawansowanych. Wszystkie niezbędne elementy zostały zamontowane na stałe w procesie lutowania, co ogranicza możliwość powstania uszkodzeń i zagubienia elementów.



Rysunek 1: Widok płytki z góry

Płytka została wyposażona w następujące elementy:

- gniazdo do podłączenia bezpośrednio płytek Arduino NANO, STM32 Nucleo-32,

- istnieje możliwość podłączenia innych płytek (ESP32, STM32 Blue Pill i innych) za pomoca dodatkowych przejściówek,
- wbudowaną przetwornicę DC/DC dostarczająca zasilania 5V, przy zasilaniu z zewnętrznym zasilaczem 12V,
- stabilizator napięcia dla 3.3V,
- zworkę umożliwiającą wybór napięcia głównego na płytce,
- wyświetlacz LCD 2x16,
- wyświetlacz 7-segmentowy z 4 cyframi,
- 8x dioda led,
- dioda RGB,
- cyfrowa dioda RGB WS2812b,
- odbiornik podczerwieni TSOP2236,
- 4x przycisk typu tact switch,
- złącze do podłączenia klawiatury membranowej 4x4,
- impulsator z opcją przycisku,
- moduły analogowe - dwa potencjometry do niezależnego zadawania napięcia i czujnik optyczny w postaci fotorezystora,
- moduły do zapoznania z interfejsem I2C - pamięć EEPROM i układ RTC z podtrzymywaniem bateryjnym,
- czujnik temperatury DS18b20, komunikujący się po interfejsie 1-wire,
- buzzer dający możliwość wprowadzenia komunikacji dźwiękowej,
- podstawkę DIP18 do podłączenia dowolnego układu w obudowie DIP i rozszerzenia możliwości zestawu o inne ćwiczenia,
- wyprowadzone dodatkowe złącza dla interfejsów I2C, SPI, UART.

Wszystkie elementy posiadają czytelny opis na płytce. Wyprowadzenia i połączenia dokonywane są przy użyciu zworek lub kabli połączeniowych żeńsko-żenskich. W dalszej części dokumentacji opisano szczegółowo wszystkie elementy płytki.

1.1 Sekcja zasilania płytki

Płytka posiada możliwość zasilania zewnętrznym zasilaczem o napięciu 12V. Zasilacz należy podłączyć do gniazda DC 2.1/5.5mm oznaczonego napisem„Power". Wbudowana przetwornica impulsowa DC/DC zmniejszy napięcie z 12V na napięcie 5V, służące do zasilania układów na płytce. Dodatkowo został zamontowany również stabilizator 3.3V dla modułów zasilanych takim napięciem - np. STM32, ESP32 itd.

Wyboru napięcia pracy modułów na płytce dokonuje się za pomocą zworki „SW1". Założenie zworki pomiędzy pin 3.3V i VCC ustawia zasilanie modułów. na poziomie napięcia 3.3V. Analogicznie, ustawienie zworki na pinach Vcc i 5V, powoduje ustalenie zasilania na poziomie napięcia 5V.

Płytkę można również zasilać bezpośrednio z podłączonego modułu. Dla podłączonego modułu Arduino Nano, zasilanie pobierane będzie z podłączonego portu USB, więc jego wydajność będzie ograniczona do możliwości podłączonego portu. Zasilanie z portu USB jest wystarczające do wykonania większości ćwiczeń na płytce.

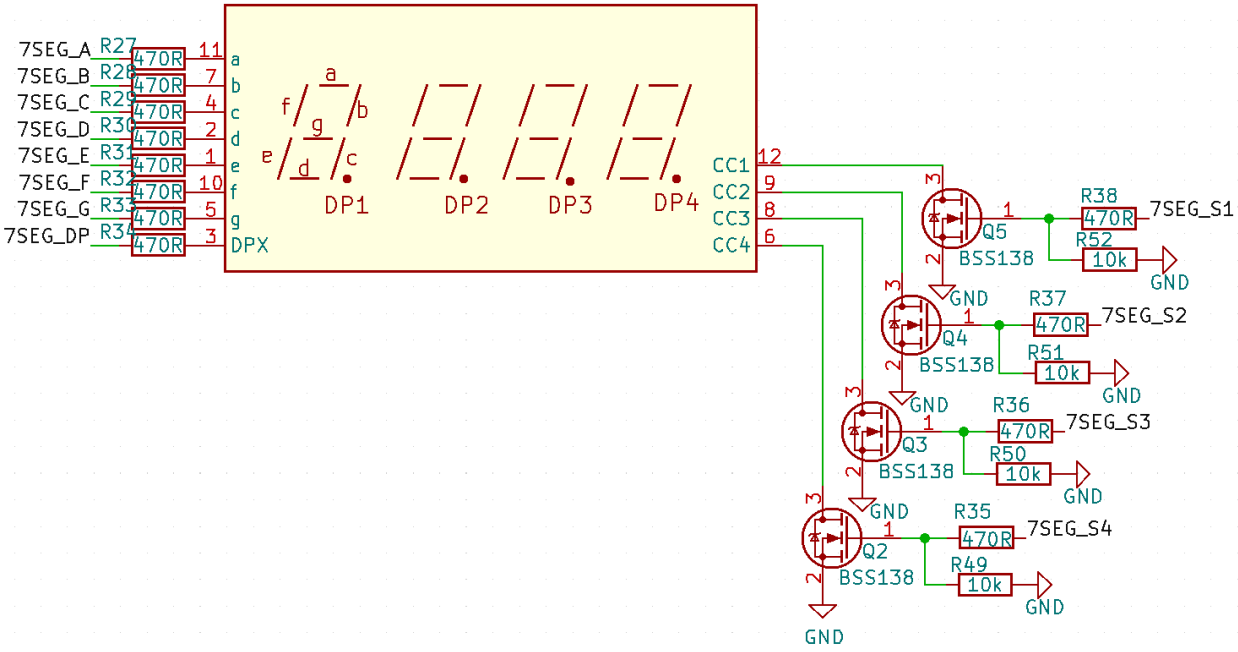
1.2 Wyświetlacz LCD 2x16

Moduł wyświetlacz LCD podłączony jest przez ekspander portów wykonany na układzie PCF8574 komunikującym się przy wykorzystaniu interfejsu I2C. Pozwala to ograniczyć ilość połączeń do minimum. Szczegółowy opis komunikacji z wykorzystaniem interfejsu I2C został zamieszczony w innym rozdziale. Adres urządzenia na interfejsie I2C to 0x27. Do łatwego sterowania wyświetlaczem można wykorzystać bibliotekę pod Arduino „LiquidCrystal_I2C.h", która jest dostępna w repozytorium bibliotek Arduino lub dołączona do przygotowanego oprogramowania. Przykładowy kod wyświetlający napis „Hello World" na wyświetlaczu LCD znajduje się poniżej:

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);


void setup() {
  lcd.init();
  lcd.backlight();
  lcd.print("Hello!!!");
}

void loop() {

}
```

1.3 Wyświetlacz 7-segmentowy

Płytka EduTar-NANO zawiera zintegrowany wyświetlacz 7-segmentowy składający się z 4 pól. Schemat elektroniczny podłączenia wyświetlacza przedstawiono na rysunku 2. Wyświetlacz jest typu: wspólna katoda. Do sterowania poszczególnymi segmentami służą piny oznaczone jako „7SEG_A”...„7SEG_DP” na złączu „J21”, sterowanie tych pinów odbywa się przez podanie stanu wysokiego. Dodatkowo w celu uruchomiania segmentu należy wysterować tranzystor odpowiedzialny za uruchomienie poszczególnych pól wyświetlacza. Tranzystory sterujemy stanem wysokim w celu uruchomienia wybranego pola i sygnał podajemy na piny „7SEG_S1”...„7SEG_S4” na złączu „J20”. Jeśli chcemy sterować niezależnie każdym polem należy wykorzystać technikę multipleksowania.

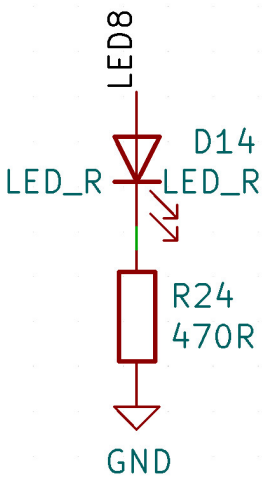


Rysunek 2: Schemat podłączenia wyświetlacza 7 segmentowego.

1.4 Linijka diodowa

Linijka diodowa składa się z 8 diod LED świecących w tym samym kolorze. Schemat podłączenia jednej diody przedstawiono na rysunku 3.

Diody połączone są na stałe z GND przez rezystor ograniczający płynący prąd. Wysterowanie diody LED odbywa się przez podanie stanu wysokiego na listwie kołkowej „J19” (środkowa, dolna cześć płytki) i pinach:



Rysunek 3: Schemat podłączenia diody LED.

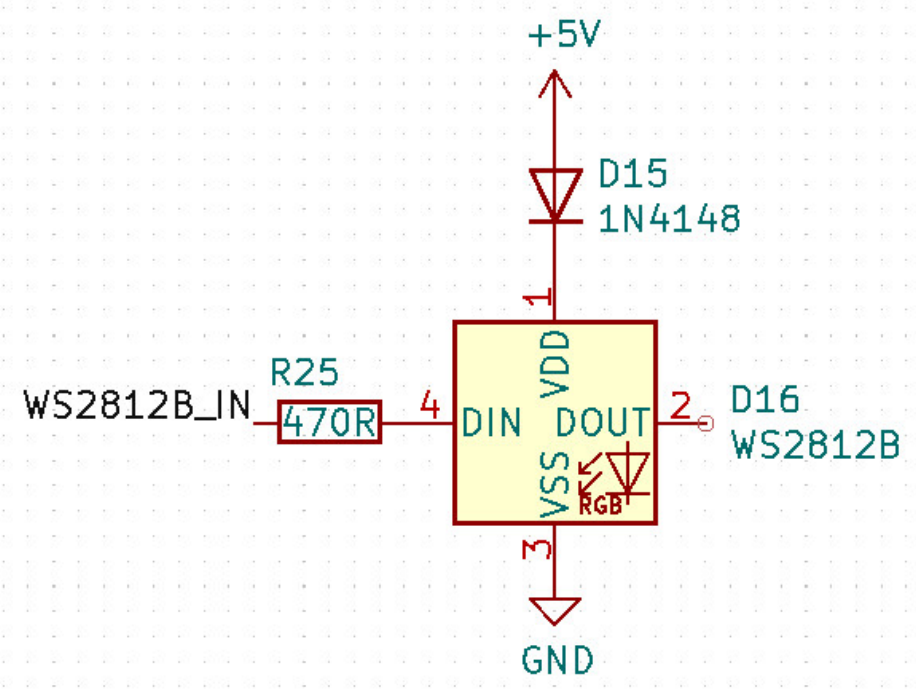
- pin „LED1” - dioda numer 1,
- pin „LED2” - dioda numer 2,
- pin „LED3” - dioda numer 3,
- pin „LED4” - dioda numer 4,
- pin „LED5” - dioda numer 5,
- pin „LED6” - dioda numer 6,
- pin „LED7” - dioda numer 7,
- pin „LED8” - dioda numer 8.

Każdy pin pozwala niezależnie wysterować wybraną diodę LED.

1.5 Cyfrowa dioda RGB–WS2812

Dioda WS2812 jest cyfrową diodę RGB. Jej sterowanie odbywa się przy wykorzystaniu tylko jednego pinu oznaczonego na płytce etykietą „LED_WS”, na złączu „J18” Schemat podłączenia diody cyfrowej umieszczono na rysunku 4. Sterowanie poszczególnymi kolorami możliwe jest przez podawanie odpowiednio przygotowanego sygnału cyfrowego, zgodnego z protokołem komunikacji przygotowanym przez producenta. Dużą zaletą zastosowania takich cyfrowych diod jest możliwość łączenia ich szeregowo i budowaniu długich

łańcuchów świetlnych. Każda dioda w łańcuchu może być wysterowana niezależnie, więc uzyskuje się możliwość tworzenia niesamowitych animacji i efektów świetlnych.



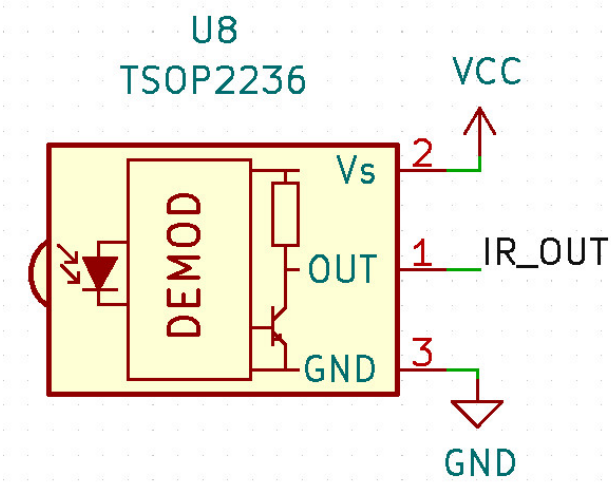
Rysunek 4: Schemat podłączenia diody cyfrowej WS2812.

1.6 Odbiornik podczerwieni–TSOP2236

Schemat podłączenia odbiornika TSOP2236, ze zintegrowanym filtrem podczerwieni umieszczono na rysunku 5. Odbiornik IR pozwala odbierać sygnał wysyłany przez popularne piloty zdalnego sterowania wyposażone w komunikację IR. Przykładem takich pilotów są urządzenia proste urządzenia do sterowania zdalnego odbiornikami telewizyjnymi, radiami, klimatyzatorami itd.

1.7 Dioda RGB

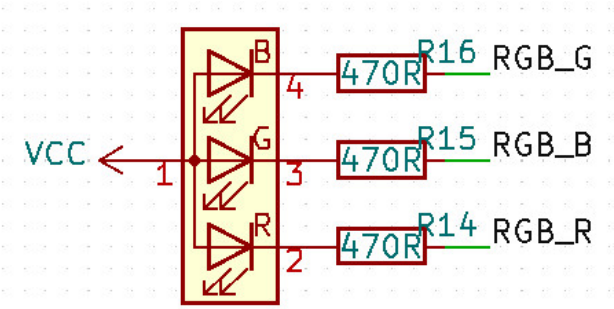
Dioda RGB umożliwia tworzenie barwnych efektów przez możliwość przenikania się 3 kolorów takich jak czerwony, zielony i niebieski. Dodatkowo każdy z kolorów można wysterować sygnałem PWM i uzyskać zmianę jasności świecenia. Zastosowana dioda ma wspólną anodę, biegun dodatni. Stero-



Rysunek 5: Schemat podłączenia odbiornika IR TSOP2236.

wanie takiej diody odbywa się przez podanie sygnału niskiego na jej katody. Wbudowane szeregowo rezystory ograniczają płynący prąd przez diodę, schemat podłączenia przedstawiono na rysunku 6. Wyprowadzenia katod diody dostępne są na złączu „J18” (lewa dolna strona płytki, pinach odpowiedzialnych za poszczególne kolory:

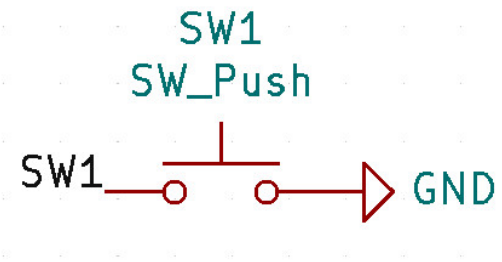
- pin „LED_R” - kolor czerwony,
- pin „LED_B” - kolor niebieski,
- pin „LED_G” - kolor zielony,



Rysunek 6: Schemat podłączenia diody RGB.

1.8 Przyciski typu tact switch

Płytka została wyposażona w 4 niezależne mono-stabilne przycisku typu tact switch. Przyciski po naciśnięciu powodują zwarcie do masy, schemat podłączenia przycisku zaprezentowano na rysunku 7.



Rysunek 7: Schemat podłączenia przycisku.

Podłączenie poszczególnych przycisków dostępne jest na listwie kołkowej „J16" i pinach:

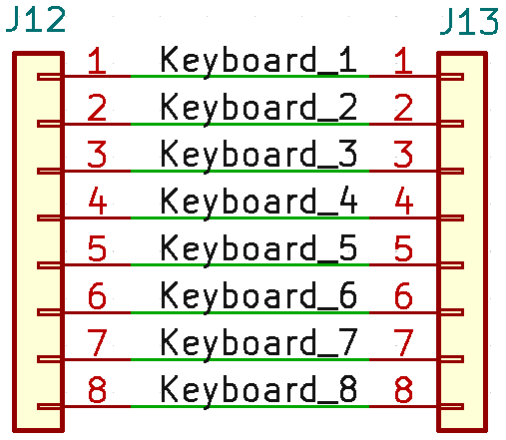
- pin „SW_1" - przycisk „SW_1" z symbolem strzałki w dół,
- pin „SW_2" - przycisk „SW_2" z symbolem strzałki w górę,
- pin „SW_3" - przycisk „SW_3" z symbolem X,
- pin „SW_4" - przycisk „SW_4" z symbolem fajki,

Pod przyciskami znajdują się okrągłe symbole sugerującą możliwą funkcję przycisku w programie. NP. przycisk „SW_1" i „SW_2" można wykorzystać do zwiększania lub zmniejszania liczby, przewijania menu programu itd. Przyciski nie posiadają wbudowanych rezystorów podciągając do napięcia zasilania, więc pozostawienie pinów nie ustawionych oznacza brak możliwości jednoznacznego określenia sygnału gdy przycisk nie jest wciśnięty. Zaleca się ustawienie odpowiednich wyprowadzeń w mikrokontrolerzy jako wejścia z „pull-up"

1.9 Złącze do podłączenia modułu klawiatury zewnętrznej

Schemat podłączenia złącza znajduje się na rysunku 8. Złącze „J13" w postaci listy kątowej goldpin znajduje się pod wyświetlaczem LCD. Umożliwia ono łatwe podłączenie dołączonej do zestawu klawiatury membranowej o rozmiarze 4x4 (dodatkowe elementy zestawu zależą od wybranej konfiguracji).

Po drugiej stronie wyświetlacza LCD znajduje się złącze „J12”, do którego należy podpiąć przewody potrzebne do obsługi klawiatury.

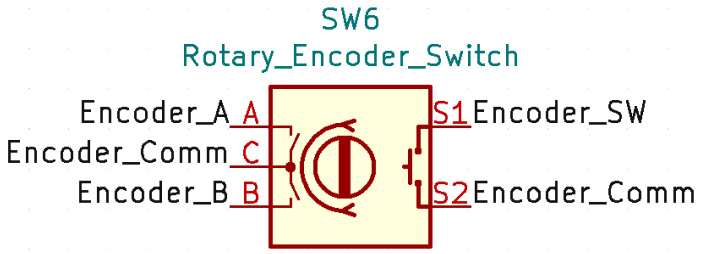


Rysunek 8: Schemat podłączenia złącza dla klawiatury matrycowej.

1.10 Impulsator

Inaczej enkoder mechaniczny, najczęściej wykorzystywany w interfejsie użytkownika do zadawania wartości danej zmiennej przez wykonanie ruchu obrotowego. Podobny w działaniu do potencjometru, ale może obracać się dowolną ilość razy, a na jego wyjściu jest sygnał kwadraturowy. Dodatkowo posiada wbudowany przycisk, którego uruchamia się przez naciśnięcie osi obrotu.

Schemat podłączenia impulsatora przedstawiono na rysunku 9.

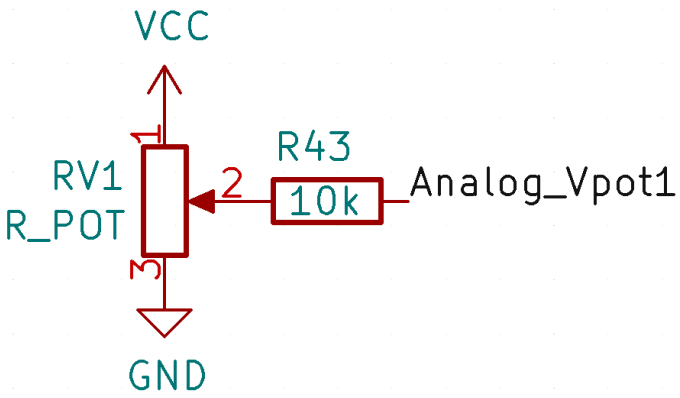


Rysunek 9: Schemat podłączenia impulsatora.

1.11 Układ zadajnika sygnału analogowego

Jako dwukanałowy zadajnik sygnału analogowego wykorzystano potencjometry w postaci dzielnika napięciowego. W celu zabezpieczenia potencjometru przed uszkodzeniem w wyniku błędnego podłączenia, na jego wyprowadzeniu pomiarowym dodano szeregowo rezystor o wartości 10k. Wyprowadzenia potencjometrów znajdują się na listwie kołkowej „J22" i „J29"- wyprowadzenia są powielone w celu łatwego podłączeniu zewnętrznego układu pomiarowego. Wyprowadzanie „A_Pot1" jest to odczep środkowy potencjometru o takim samym oznaczeniu. Analogicznie działa wyprowadzenie oznaczone „A_Pot2"

Schemat podłączeni pojedynczego potencjometru przedstawiono na rysunku 10.



Rysunek 10: Schemat podłączenia potencjometru.

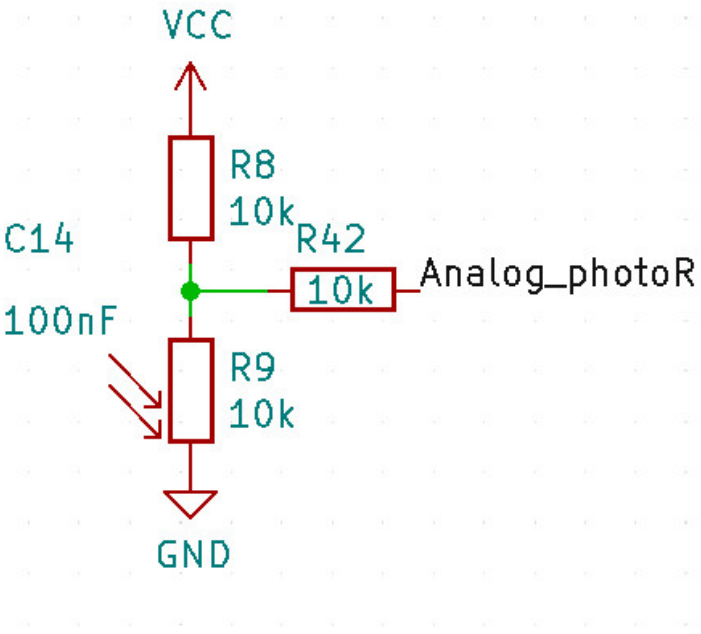
1.12 Analogowy czujnik jasności

Czujnik poziomu światła został wykonany na podstawie dzielnika napięcia wyposażonego w fotorezystor. Wyprowadzenie sygnału analogowego z dzielnika znajduje się na listwie kołkowej „J22" i „J29', na pinie „A_Photo".

Szczegółowy schemat modułu czujnika jasności przedstawiono na rysunku 11. Wraz ze wzrostem jasności otoczenia, rośnie ilości światła padającego na czujnik, co powoduje zmniejszenie rezystancja fotorezystora R9. W wyniku czego na wyprowadzeniu „Analog_photoR" maleje napięcie.

1.13 Układ pamięci nieulotnej–EEPROM

Pamięć eeprom znajduje się w mikrokontrolerze zastosowanym w module Arduino NANO, więc nie jest montowana w postaci zewnętrznego układu.



Rysunek 11: Schemat podłączenia czujnika jasności.

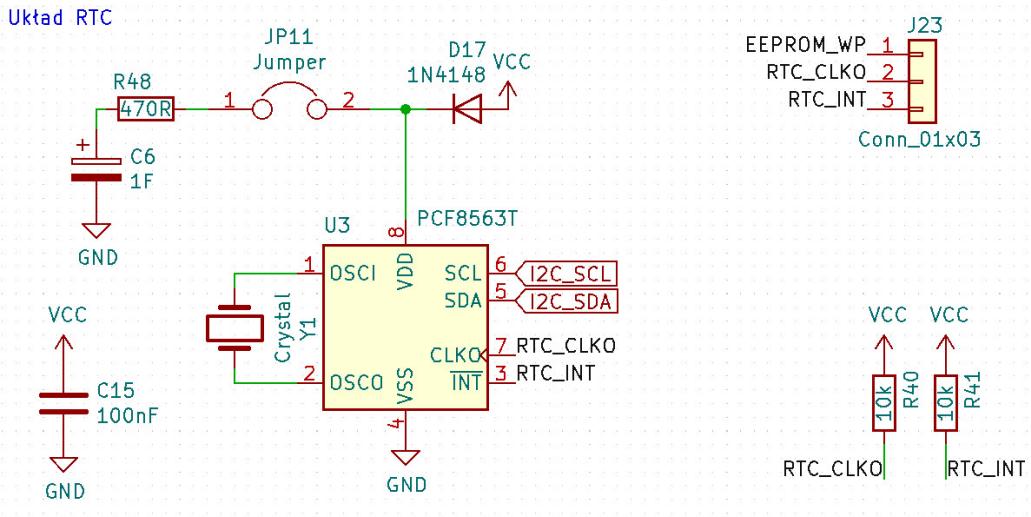
1.14 Układ zegara czasu rzeczywistego - RTC

Jako układ czasu rzeczywistego zastosowano dedykowany scalony element o oznaczeniu PCF8563. Schemat podłączenia układu RTC znajduje się na rysunku 12. Układ RTC komunikuje się z mikrokontroler przez wykorzystanie magistrali I2C, której działanie i używanie zostało omówione w innym rozdziale.

1.15 Cyfrowy czujnik temperatury–DS18b20

Do pomiaru temperatury wykorzystano popularny i powszechnienie stosowany czujnik cyfrowy firmy Dallas o symbolu „DS12B20". Podstawowe parametry czujnika:

- dokładność pomiaru:
- napięcie zasilania: od 3.0 V do 5.5 V,
- zakres pomiarowy: od -55C do 125C,
- obudowa: TO92.

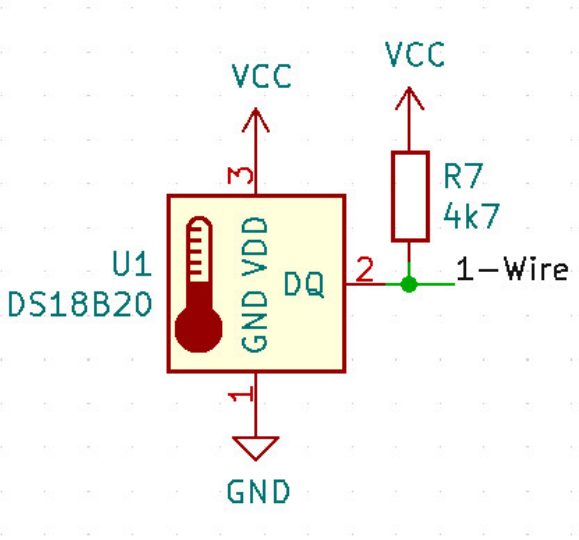


Rysunek 12: Schemat podłączenia układu RTC PCF8563T.

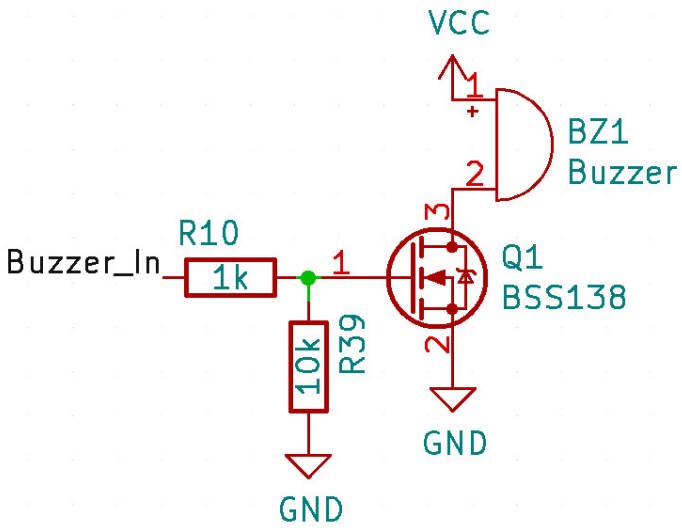
Zaleta wybranego czujnika jest jego dokładność pomiaru, duży zakres mierzonych temperatur i kalibracja temperaturowa dokonana przez producenta. Komunikacja z czujnikiem odbywa się przy wykorzystaniu interfejsu 1-wire. Interfejs wymaga do odczytania danych tylko jednej linii, co przekłada się bezpośrednio na prostą budowę rozległych instalacji i ograniczenie kosztów powodowanych przez ilość dodatkowego okablowania. Wyprowadzenie czujnika umieszczono na listwie kołkowej „J11" (środkowa, dolna część płytki) i pinie „DS_conn". Schemat modułu czujnika temperatury znajduje się na rysunku 13.

1.16 Generator sygnału akustycznego

Generator sygnału akustycznego pozwala zapoznać kursanta z sposobem generowania dźwięku za pomocą buzzer. Buzzer jest sterowany przez tranzystor, którego wyprowadzenie znajduje się na listwie kołkowej „J11" (środkowa, dolna część płytki) i pinie „Buz_con". Podanie sygnału wysokiego powoduje przewodzenie przez tranzystor „Q1" i uruchomienie buzzera, czyli wygenerowanie dźwięku. Schemat podłączenia buzzera znajduje się na rysunku 14.



Rysunek 13: Schemat podłączenia czujnika temperatury DS18b20.



Rysunek 14: Schemat podłączenia generatora sygnału akustycznego.

Złącze J14			
Opis na PCB	Pin mikrokontrolera	Dodatkowe funkcje	
D13	PB5	SPI–SCK	LED na płytce Arduino Nano
D12	PB4	SPI–MISO	
D11	PB3	SPI–MOSI/OC2A	
D10	PB2	SPI–SS/OC1B	
D9	PB1	OC1A	
D8	PB0		
D7	PD7		
D6	PD6	OC0A	
D5	PD5	OC0B	
D4	PD4		
D3	PD3	OC2B//INT1	
D2	PD2	INT0	
D1	PD0	UART–Tx	
D0	PD1	UART–Rx	

Tabela 1: Opis wyprowadzeń na złączu J14.

1.17 Złącze rozszerzeń dla płytki Arduino Nano

Na płytce znajdują się złącza J14 i J15 podłączone bezpośrednio do pinów złącza A1 do którego wkłada się moduł Arduino Nano. Złącza kołkowe męskie zostały opisane i ułożone tak aby ich wykorzystanie było jak najłatwiejsze.

2 Interfejsy komunikacyjne

Niektóre moduły umieszczone na płytce zostały podłączone do wybranych interfejsów komunikacyjnych, w celu uproszczenia korzystania z płytki edukacyjnej. W zależności od interfejsu, jak i wybranego modułu głównego ich podłączenie i sterowanie może odbywać się w różny sposób. W dokumentacji skupiono się na wykorzystaniu elementów z modułem głównym w płytki Arduino NANO. Podłączenie innych płytek może wymagać innej konfiguracji płytki EduTar-NANO, co zostało opisane szczegółowo w rozdziałach dotyczących poszczególnych modułów głównych.

2.1 UART

Interfejs komunikacyjny wymagający do przesyłu danych dwukierunkowo, dwóch linii danych:

Złącze J15			
Opis na PCB	Pin mikrokontrolera	Dodatkowe funkcje	
AREF	AREF		
A0	PC0	ADC0	
A1	PC1	ADC1	
A2	PC2	ADC2	
A3	PC3	ADC3	
A4	PC4	I2C–SDA/ADC4	
A5	PC5	I2C–SCL/ADC5	
A6		ADC6	tylko wejście analogowe
A7		ADC7	tylko wejście analogowe

Tabela 2: Opis wyprowadzeń na złączu J15.

- linia danych Rx,
- linia danych Tx,

Domyślnie na płytce Arduni Nano jest on podpięty do portu USB przez dedykowany moduł komunikacji jak „CH340", „FT232" itd. (zależy od wersji/producenta płytki Arduino Nano). Jednocześnie stanowi programator dla naszego modułu, jak również umożliwią na komputerze utworzenie portu COM i komunikacje pomiędzy komputerem a mikrokontrolerem.

2.2 I2C

Interfejs komunikacyjny I2C wykorzystuje do komunikacji dwie linie:

- „SCL" – linia sygnału zegarowego,
- „SDA" – linia danych,
- masa, GND – należy pamiętać aby masa wszystkich podłączonych układów była taka sama, zapewnić jednakowy poziom potencjału.

Komunikacja odbywa się w standardzie Master-Slave. Gdzie masterem jest mikrokontroler zarządzający transmisją i sterujący pozostałymi układami typu slave. Modułów typu slave może znajdować się na jednej magistrali I2C wiele. Każdy moduł slave musi posiadać swój unikatowy adres, po którym master jest w stanie nawiązać z nim komunikacje.

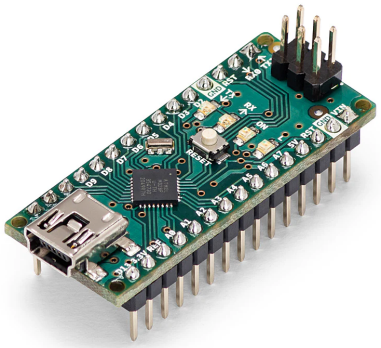
Moduły podłączone do interfejsu I2C na płytce EduTar-NANO:

- wyświetlacz LCD – adres: 0x27,
- układ RTC –adres: 0xa3,

2.3 SPI

3 Moduł mikrokontrolera - płytka Arduino NANO

Płytka Arduino Nano jest jedną z najpopularniejszych płytek z ekosystemu Arduino. Jej podstawową zaleta jest niewielki rozmiar, wyposażenie w mikrokontroler Atmega328 i moduł portu USB zapewniający możliwość komunikacji i programowania bezpośrednio za pomocą portu USB i komputera. Oryginalna płytka została przedstawiona na rysunku 3, ze względu na swoją popularność na rynku jest dostępnych wiele klonów tej płytki. Klony są to płytki wyprodukowane przez różne firmy trzecie, które nie należą do fundacji Arduino, więc kupując taki moduł nie przyczyniamy się do wsparcia finansowego projektu Arduino i jego dalszego rozwoju. Jeśli chodzi o samą funkcjonalność, to klony są w pełni sprawnymi płytkami i idealnie nadają się do rozpoczęcia swojej przygody z mikrokontrolerami. Przykładowy klon został zaprezentowany na rysunku 16.

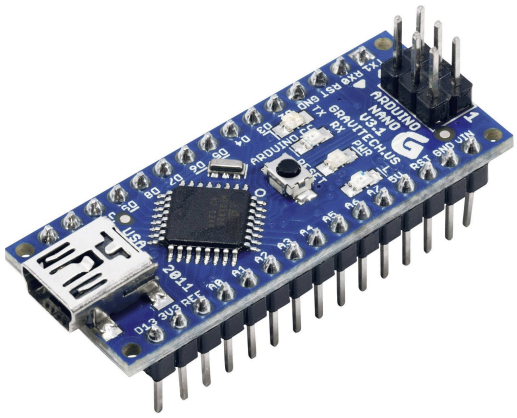


Rysunek 15: Oryginalna płytka Ardunino Nano.

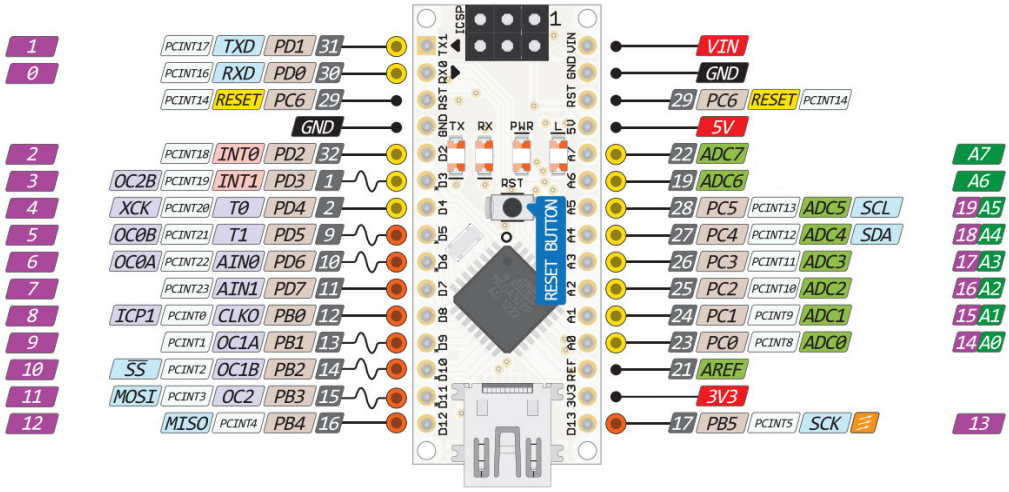
Płytka Arduino Nano posiada 30 wyprowadzeń w postaci dwóch list kołkowych po 15 pinów. Każdy pin posiada przypisane funkcję i w zależności od wybranego zadania, musi zostać odpowiednio skonfigurowany i wykorzystany. Dokładny opis pinów na płytce można znaleźć na rysunku 17. Dodatkowo na płytce umieszczono złącze kołkowe 2x3pin, służy ono do programowania mikrokontrolera przy wykorzystaniu zewnętrznego programatora po interfejsie ISP.

3.0.1 Instalacja sterowników Arduino NANO

Płytka Arduino Nano do komunikacji z komputerem wykorzystuje układy pośredniczące konwertujące interfejs UART na USB. Większość płytek jest automatycznie wykrywana przez system operacyjny i są instalowane sterowniki



Rysunek 16: Główne okno aplikacji Arduino IDE.



Rysunek 17: Arduino IDE - wybór płytki Arduino Nano

urządzeń. Po zainstalowaniu w menadżerze urządzeń powinno być widoczne nowe urządzenie portu „COM".

Jeśli tak się nie stanie to należy ręcznie zainstalować wymagany sterownik w zależności od wersji płytki Arduino Nano i zamontowanego chipu. Do najpopularniejszych należą:

- układ firmy FTDI (FT232)
  `http://edutar.pl/wp-content/uploads/2023/01/ftdi-driver.zip`
- układ firmy WCH.CN (CH340)
  `http://edutar.pl/wp-content/uploads/2023/01/CH341SER.zip`

Inną opcją jest zainstalowanie środowiska Arduino, które zawiera sterowniki do swoich oryginalnych płytek. Środowisko można pobrać ze strony Arduino z zakładki software:
`https://www.arduino.cc/en/software`

Po zainstalowaniu sterowników system sam powinien wykryć płytę i przypasować port COM w systemie.

3.1 Środowisko programistyczne - Arduino IDE

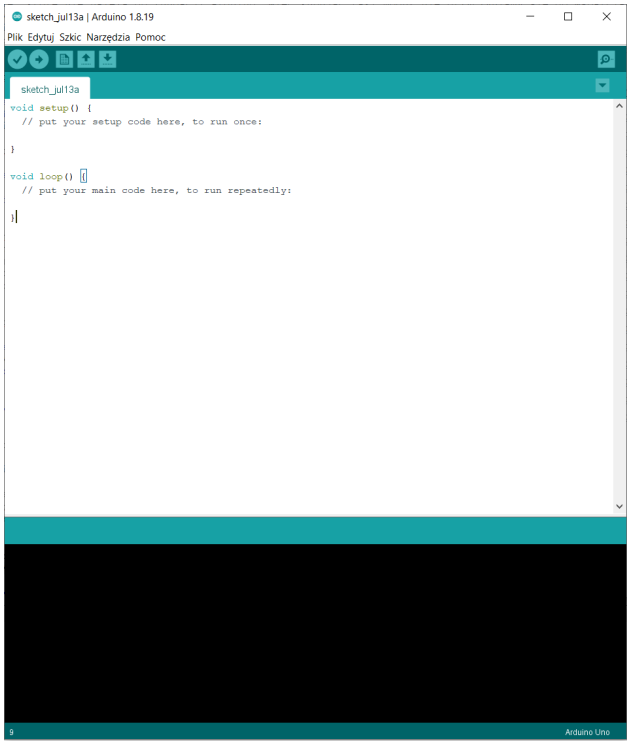
Najprościej swoją przygodę z nauką programowania systemów embedded na podstawie płytki Arduino Nano zacząć od wykorzystania gotowego środowiska programistycznego jakim jest „Arduino IDE". Jest to oprogramowanie przygotowane przez twórców platformy Arduino. Instalacja sprowadza się do ściągnięcia i uruchomienia odpowiedniej wersji dla naszego systemu operacyjnego z oficjalnej strony:
`https://www.arduino.cc/en/software/`
Jeśli nie chcemy instalować dodatkowego oprogramowania na swoim komputerze, to istnieje możliwość programowania przez przeglądarkę internetową, dokładny opis tego sposobu znajduje się na stronie:
`https://docs.arduino.cc/arduino-cloud/getting-started/getting-started-web-editor`
Oprogramowaniu Arduino IDE zostało przygotowane do współpracy z płytkami Arduino. Posiada przygotowane konfiguracje dla modułów, dzięki czemu wymagana konfiguracja jaką musi dokonać użytkownik, sprowadza się do wyboru odpowiedniej płytki z menu aplikacji.

Opis interfejsu aplikacji Arduino IDE w wersji 1.8.19

Po ściągnięciu i rozpakowaniu plików uruchamiamy aplikację „arduino". Po uruchomieniu, otworzy się okno aplikacji widoczne na rysunku 18. Aplikacja powinna być w języku polskim, jeśli chcemy korzystać z innego języka interfejsu to odpowiedniej zmiany możemy dokonać w ustawieniach programu. Górna cześć aplikacji zawiera menu umożliwiające skonfigurowanie aplikacji, zapisanie projektu, wybranie parametrów płytki itd.

Ustawienie płytki Arduino Nano

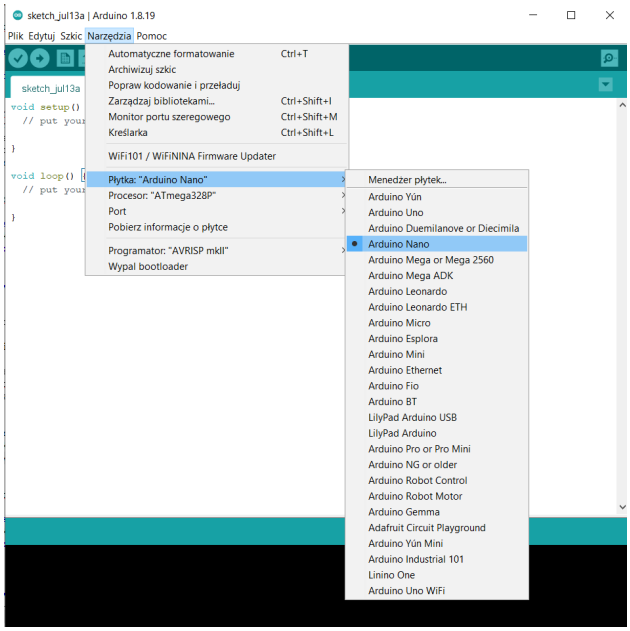
Programowanie Arduino Nano należy rozpocząć od wybrania odpowiedniej konfiguracji. Wybieramy z menu „Narzędzia - > Płytka..." i z rozwijanej listy dostępnych konfiguracji zaznaczamy „Arduino Nano" - rysunek 19. Po tym kroku automatycznie zostanie utworzona odpowiednia konfiguracja z



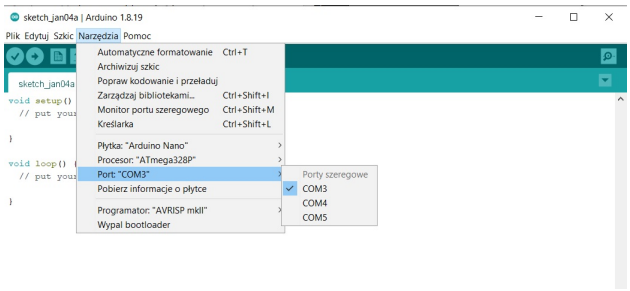
Rysunek 18: Płytka Ardunio Nano z opisem wyprowadzeń.

mikrokontrolerem ATmega328P, częstotliwością taktowania ustawioną na 16 MHz.

Kolejnym krokiem, który musimy wykonywać przy każdym podłączeniu nowej płytki Arduino do komputera jest wybór odpowiedniego portu COM. Podłączając płytkę do komputera za pomocą kabla USB, system operacyjny wykrywa nowe urządzenie i instaluje odpowiednie sterowniki (szczegółowy opis instalacji sterowników został umieszczony w innym rozdziale). Po zainstalowaniu sterowników płytka widoczna jest w systemie jako port „COMn", gdzie „n" to numer portu i jest przyznawany przez system operacyjny. Najczęściej jak dodajemy nowe urządzenie to numer jego portu będzie największy. Jeśli tak nie będzie, to należy wybrać inny numer. Najłatwiej zrobić to zapamiętując listę dostępnych portów COM z podłączoną płytką, następnie odłączyć płytkę od komputera i ten numer, który zniknął jest numerem naszej płytki. Port COM ustawiamy w oprogramowaniu Arduino przez wybranie z menu „Narzędzia -> Port" i wybranie odpowiedniego portu - rysunek 20. Wybrany port będzie jednocześnie pełnił rolę programatora jak i umożliwi komunikację za pomocą interfejsu UART pomiędzy mikrokontrolerem a komputerem.



Rysunek 19: Ustawienie płytki Arduino Nano w środowisku programistycznym.



Rysunek 20: Ustawienie portu COM w środowisku programistycznym.

3.1.1 Programowanie w języku C - środowisko Ardunio

Domyślnie środowisko Arduino jest wyposażone w zestaw bibliotek napisanych w języku C++, których zadaniem jest szybsze tworzenie programów. Uzyskuje się to przez wykorzystanie warstwy wyższego rzędu, która niweluje konieczność zapoznania się z rejestrami mikrokontrolera i pracy na nich. Utworzono szereg gotowych funkcji i bibliotek, które dostarczają użytkownikowi gotowych funkcji do operowania sygnałami wejściowi/wyjściowymi, kanałami ADC, przerwaniami itd. Dokładny opis tych funkcji można znaleźć na stronie Arduino.

Czasem jednak dla osób dopiero zaczynających swoją przygodę z progra-

mowaniem mikrokontrolerów, wykorzystanie gotowych funkcji nie jest dobra formą startu. Dlatego też, powstaje konieczność rozpoczęcia nauki programowania przez otworzenie dokumentacji mikrokontrolera i zapoznanie się z jego rejestrami. Jak przez ostawianie 0 i 1 na odpowiednich pozycjach w rejestracjach można ustawiać stany pinów, generować sygnał PWM, czy uruchamiać przerwania. W tym celu można napisać program w czystym języku C bezpośrednio w środowisku Ardunio. Taki program kompilujemy jak każdy inny i wgrywamy bez żadnych komplikacji. Poniżej przedstawiono przykładowy kod programu, który mruga diodą LED na płytce Arduino Nano podłączoną do pinu „PB5":

```
#include <avr/io.h>
#include <stdlib.h>
#include <string.h>
#include <util/delay.h>

#ifndef _BV
#define _BV(bit)          (1<<(bit))
#endif
#ifndef sbi
#define sbi(reg,bit)      reg |= (_BV(bit))
#endif

#ifndef cbi
#define cbi(reg,bit)      reg &= ~(_BV(bit))
#endif

int main() {
//Ustawienie pinu PB5 jako wyjscie
  //sbi(DDRB,PB5);
  DDRB|=(1<<PB5);

  while (1) {
  //Ustawienie pinu PB5 w stanie wysokim
   //sbi(PORTB,PB5);
   PORTB|=(1<<PB5);
   _delay_ms(500); //poczekanie 500ms-0,5s


 //Ustawienie pinu PB5 w stanie niskim
   //cbi(PORTB,PB5);
   PORTB&=~(1<<PB5);
   _delay_ms(500);  //poczekanie 500ms-0,5s
  }
}
```

3.1.2 Programowanie w języku asembler - środowisko Ardunio

Dla bardziej zaawansowanych programistów, którzy chcą poznać dokładnie jak działa mikrokontroler, bez ufania na zabiegi jakie poczyni kompilator np. języka C. Jak również poznać tajniki ustawiania rejestrów, zarządza stosem pamięci, kontrolowania przepływu zmiennych w pamięć i wszystkie operacje na poziomie maszynowym. Warto zaprezentować język programowania Asembler. Środowisko Arduino umożliwiaj skompilowanie programu bezpośrednio napisanego w języku Asembler i załadowanie do płytki Arduino Nano. w tym celu należy wykonać poniższe kroki:

1. utworzyć pusty folder z miejscem na nowy projekt, nazwijmy go „test”,

2. wchodzimy do utworzonego folderu i tworzymy plik projektu arduino, plik nazywamy „test.ino” - rozszerzenie pliku musi być „ino”, plik zostawiamy pusty,

3. tworzymy plik o nazwie „test.S”, rozszerzenie musi być „S”, w którym piszemy program w języku asembler,

4. tak przygotowany projekt, możemy uruchomić w środowisku Arduino, skompilować i wgrać do mikrokontrolera.

Przykładowy program w języku asembler przedstawiono poniżej (program mruga diodą LED na płytce Arduino Nano podłączoną do pinu „PB5”):

```
; Blink LED on PB5

#define __SFR_OFFSET 0

#include "avr/io.h"

.global main

main:
  sbi   DDRB, 5     ; Set PB5 as output

blink:
  sbi   PINB, 5     ; Toggle PINB
  ldi   r25, hi8(500)
  ldi   r24, lo8(500)
  call  delay_ms
  jmp   blink

delay_ms:
  ; Delay about (r25:r24)*ms. Clobbers r30, and r31.
  ; One millisecond is about 16000 cycles at 16MHz.
  ; The inner loop takes 4 cycles, so we repeat it 3000 times
  ldi   r31, hi8(4000)
  ldi   r30, lo8(4000)
1:
  sbiw    r30, 1
  brne    1b
  sbiw    r24, 1
  brne    delay_ms
  ret
```

3.1.3 Struktura programu Arduino

Uruchamiamy środowisko Arduino i z menu wybieramy „Plik -> Nowy". Oprogramowanie Arduino automatycznie utworzy nam nowy projekt i utworzy pusty szablon programu do którego należy wkleić kod przedstawiony poniżej (program mruga diodą LED na płytce Arduino Nano podłączoną do pinu „PB5"):

```
void setup() {
  //ustawienie portu PB5 - numeracja w Arduino 13, jako wyjscie
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);    //ustawienie stanu wysokiego
                          //na PB5-numeracja Arduino 13
  delay(1000);               //poczekanie 1000ms-1s
  digitalWrite(13, LOW);     //ustawienie stanu niskiego
                          //na PB5-numeracja Arduino 13
  delay(1000);               //poczekanie 1000ms-1s
}
```

Programy w środowisku Arduino składają się z dwóch funkcji:

- funkcja „void setup()" - funkcja uruchamiana raz po uruchomieniu mikrokontrolera lub jego restarcie. W jej środku należy dokonywać jednorazowych czynności polegających na konfiguracji peryferiów, ustawianiu wartości zmiennych itd.
- funkcja „void loop()" - funkcja wywoływana cyklicznie, tutaj piszemy właściwy kod programu.

Funkcje te można porównać do klasycznego programu w języku C jak na listingu poniżej:

```
int main() {
  setup();       //wywolanie funkcji po uruchomieniu programu

  while(1) {
    loop();      //wywolywana cyklicznie
  }

  return 0;
}
```

3.2 Środowisko programistyczne - PlatformIo

4 Moduł mikrokontrolera - płytka Nucleo

5 Moduł mikrokontrolera - płytka ESP32