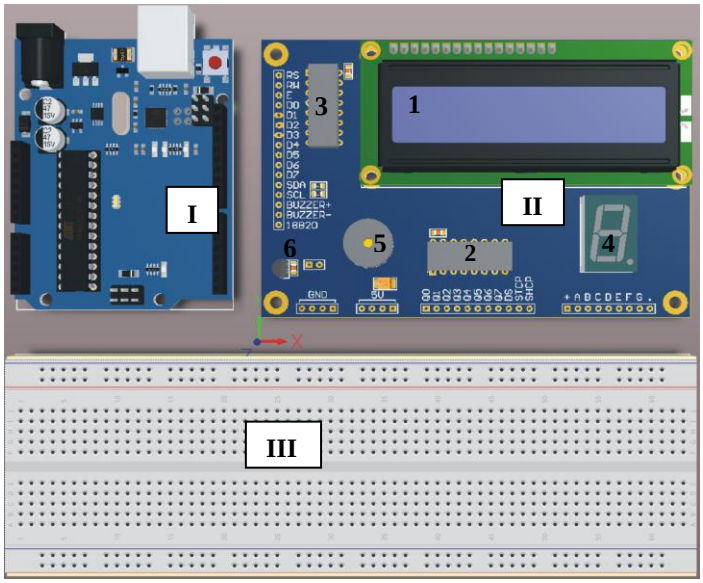
UKŁADY CYFROWE CZ II

CEL ĆWICZENIA

Celem ćwiczenia jest zapoznanie się z właściwościami wybranych układów cyfrowych oraz ich praktyczna realizacja przy pomocy zestawu edukacyjnego Arduino.

A1. OPIS TECHNICZNY STANOWISKA LABORATORYJNEGO

Całość ćwiczenia zostanie przeprowadzona w oparciu o widoczny na **Rys.1.** zestaw edukacyjny na bazie modułu **Arduino UNO**, który wspomagany środowiskiem programistycznym z gotowymi zestawami funkcji bibliotecznych napisanych na języku C umożliwia realizację wszystkich podstawowych układów cyfrowych oraz wybranych rozwiązań analogowych.



I - Moduł Arduino Uno
II - Zestaw układów peryferyjnych
1. Wyświetlacz LCD
2. Układ 74HC595N
3. Układ PCF8574
4. Wyświetlacz 7 segmentowy
5. Buzzer
6. Czujnik temperatury

III - Uniwersalna płytka montażowa

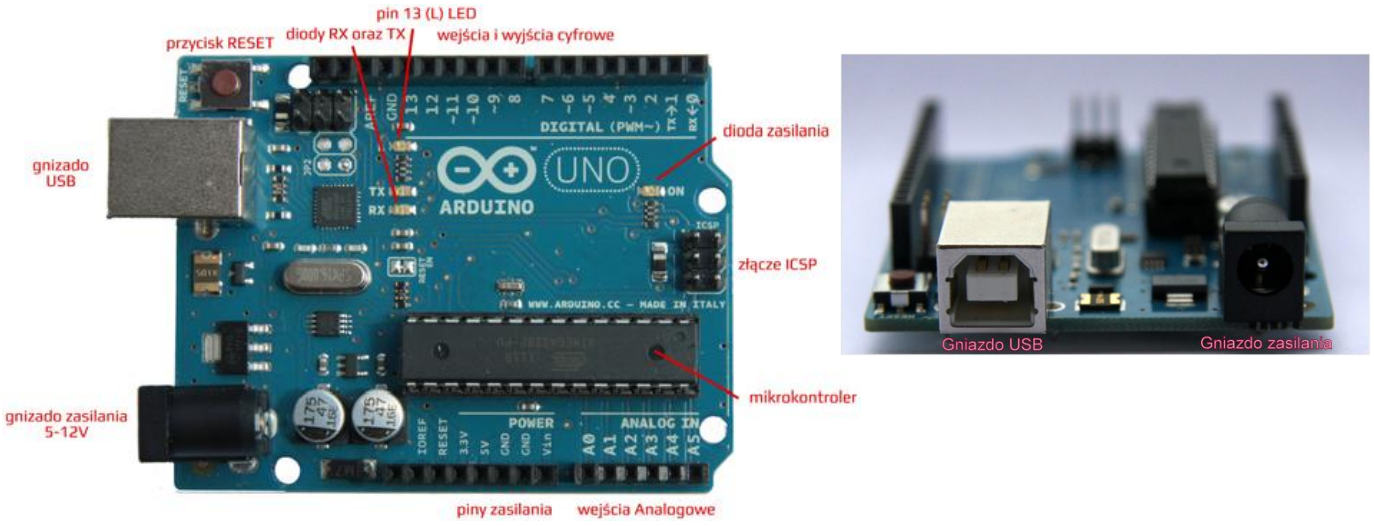
Rys.1. Zestaw edukacyjny na bazie Arduino Uno

A2. OPIS MODUŁU ARDUINO UNO

Najważniejszym elementem opisywanego tu stanowiska laboratoryjnego jest moduł Arduino. Krótki opis przedstawiony poniżej pomoże w rozpoznaniu jego możliwości, ułatwi jego obsługę oraz budowę układów .

A2.1 PLATFORMA SPRZĘTOWA

Arduino Uno jest podstawową i zarazem najpopularniejszą wersją z całej serii o w. w nazwie. Płytka zawiera mikrokontroler ATmega328, wyposażony w 14 cyfrowych wejść/wyjść z czego 6 można wykorzystać jako wyjścia PWM (np. do sterowania silnikami) oraz 6 analogowych wejść. Rozmieszczenie poszczególnych elementów przedstawione jest na **Rys. 2**. Układ taktowany jest sygnałem zegarowym o częstotliwości 16 MHz, posiada 32 kB pamięci programu Flash oraz 2 kB pamięci operacyjnej SRAM. Płytę Arduino Uno można zasilać z komputera albo zasilaniem zewnętrznym.

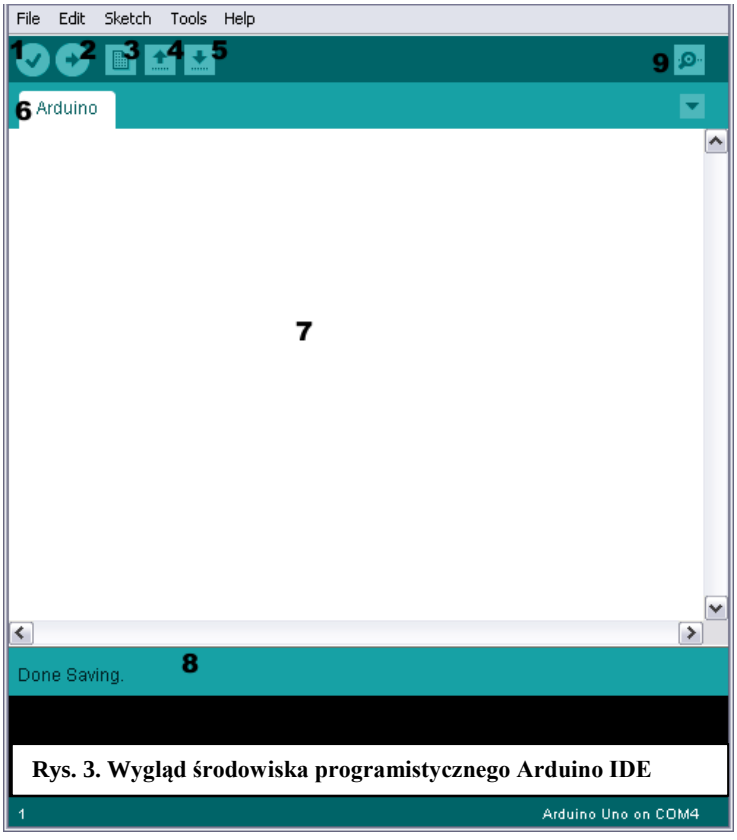


Rys. 2. wygląd płyty Arduino Uno i rozmieszczenie elementów

Poniżej przedstawione jest kilka cech, które wyróżniają moduły Arduino na tle innych płytek programowalnych:

Nazwa	Opis
Zainstalowany bootloader	Dzięki zainstalowanemu bootloaderowi do zaprogramowania urządzenia wystarczy odpowiedni przewód USB oraz oprogramowanie ze strony producenta.
Arduino Shield	Rozkład złącz umożliwia montaż dedykowanych nakładek tzw. Arduino Shield.
Wyprowadzenia cyfrowe	14 cyfrowych wejść/wyjść umożliwia m.in. sterowanie diodami LED, przekaźnikami oraz odczytywanie stanów przycisków.
Wydajność prądowa	Maksymalna wydajność prądowa pojedynczego wyprowadzenia wynosi 40 mA.
Wyjścia PWM	6 wyjść PWM pozwala np. na sterowanie silnikami oraz regulowanie jasności diod.
Wejścia analogowe	6 wejść wbudowanego przetwornika analogowo-cyfrowego o rozdzielczości 10-bitów obsługuje m.in. czujniki z wyjściem analogowym.
Komunikacja szeregowa	Urządzenie obsługuje popularne interfejsy komunikacyjne, m.in.: UART, I2C i SPI.
Pamięć wbudowana	Układ Atmega328 taktowany jest sygnałem o częstotliwości 16 MHz, posiada 32 kB pamięci programu Flash, 1 kB EEPROM oraz 2 kB pamięci operacyjnej SRAM.
Zasilanie złącze DC	Do zasilania Arduino można wykorzystać dowolny zasilacz o napięciu od 7 V do 12 V ze złączem DC 5,5 x 2,1 mm.
Zasilanie port USB	Płytkę można zasilać z komputera poprzez przewód USB pamiętając przy tym, że maksymalna wydajność prądowa portu USB wynosi 500 mA. Arduino posiada system chroniący gniazdo przed zwarciem oraz przepływem zbyt wysokiego prądu.
Złącze ICSP	Moduł posiada wyprowadzenia ICSP służące do podłączenia zewnętrznego programatora AVR.
Pin IOREF	Pin IOREF umożliwia bezpośredni dostęp do napięcia z jakim pracują wyprowadzenia I/O.
Wbudowana dioda LED	Podłączona dioda LED na pinie 13 umożliwia debuggowanie prostych programów.
Wyjście 3,3 V	Wbudowany regulator napięcia umożliwia zasilanie zewnętrznych urządzeń napięciem 3,3 V o poborze prądu do 50 mA.
Mikrokontroler w THT	Dzięki zastosowaniu obudowy przewlekanej THT w przypadku uszkodzenia mikrokontrolera głównego Atmega328P można go w prosty sposób wymienić.

A2.2 Środowisko programistyczne



Rys. 3. Wygląd środowiska programistycznego Arduino IDE

Środowisko programistyczne Arduino IDE (w. 1.6.6.) wygląda jak na **Rys. 3,** można je pobrać ze strony producenta www.arduino.cc lub www.arduino.org. Oprogramowanie jest ogólnodostępne i kompatybilne w modułem Arduino Uno. Język, który służy do programowania Arduino to język C/C++ obsługiwany przez środowisko Wiring. Arduino posiada wiele bibliotek, z gotowymi funkcjami do obsługi układów elektronicznych. Niektóre z nich zostaną użyte w ramach niniejszego ćwiczenia.

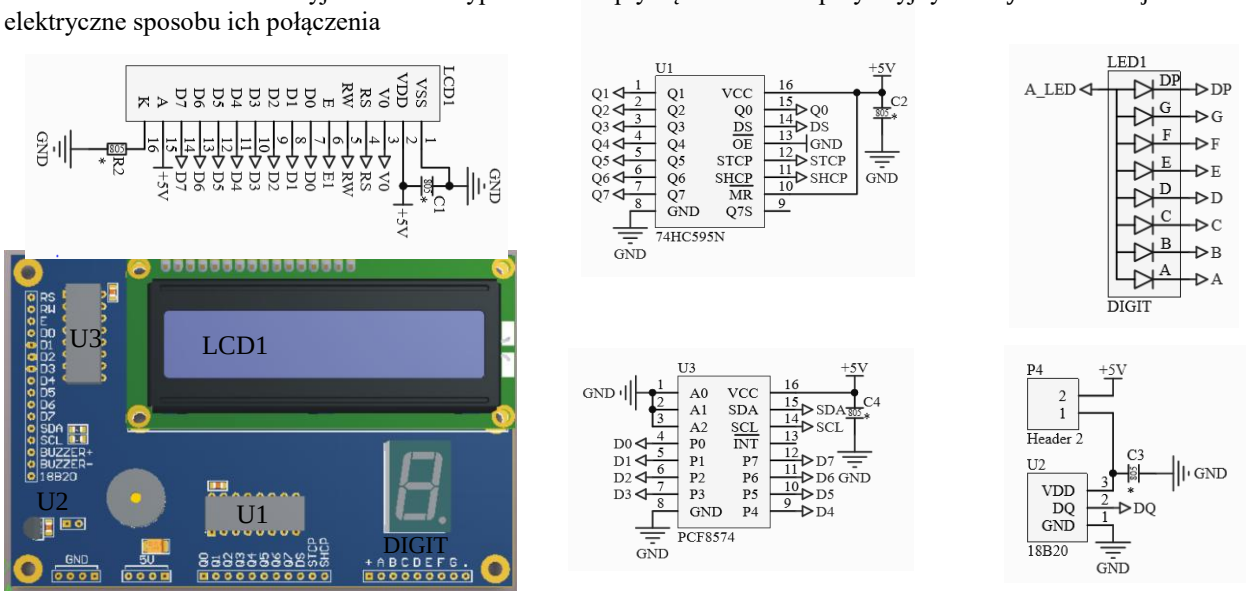
Opis okna:

1. Weryfikacja kodu programu
2. Wgranie programu do Arduino
3. Nowy dokument
4. Otwórz dokument
5. Zapisz dokument
6. Nazwa dokumentu
7. Miejsce do wpisania kodu
8. Komunikaty stanu oraz informacje o błędach
9. Okno komunikatów portu szeregowego

Działanie mikrokontrolera w module Arduino opiera się na wykonywaniu programu, który po edycji i kompilacji w widocznym na **Rys. 3** środowisku zostaje wgrany do jego pamięci. W zależności od tego z jakich elementów zestawiono układ program ten przetwarza dane wejściowe (np. z czujnika temperatur) i/lub steruje wyjściami tego układu (np. wyświetlaczem LED).
Każdy program Arduino składa się z dwóch funkcji głównych: `setup()` oraz `loop()`, które odpowiadają za działanie całego systemu. Funkcja `setup()` wykonywana jest tylko raz przy uruchomieniu układu, natomiast funkcja `loop()` wykonywana jest cały czas w trakcie działania układu. Można też definiować swoje własne funkcje, jednakże mogą być one używane tylko w ciele obu w.w. funkcji głównych.

A3. OPIS PŁYTKI Z UKŁADAMI PERYFERYJNYMI

Stanowisko laboratoryjne zostało wyposażone w płytkę z układami peryferyjnymi. **Rys. 4.** ilustruje schematy elektryczne sposobu ich połączenia



Rys. 4. Połączenia na płytce z układami peryferyjnymi

W trakcie realizacji poszczególnych zadań i projektów poszczególne elementy i układy widocznej tu płytki będą łączone albo bezpośrednio z wejściami lub wyjściami modułu Arduino lub tez z elementami umieszczonymi na uniwersalnej płytce montażowej. Poniżej przedstawiono specyfikacje wybranych elementów wchodzących w skład w.w. zestawienia.

A3.1 WYŚWIETLACZ LCD 2X16 ZNAKÓW (LCD1)

Popularny alfanumeryczny wyświetlacz LCD, zasilany napięciem 5 V. Charakteryzuje się prostą obsługą, wysoką dostępnością oraz licznym wsparciem dla wielu mikrokontrolerów.

parametry:

- Wyświetlacz LCD 2x16 znaków,
- Sterownik zgodny z HD44780
- Niebieski negatyw
- Podświetlanie białe diody LED, białe znaki
- Rozmiar modułu : 80 x 36 x 12 mm
- Wymiary znaku: 2,45 x 5,00 mm
- Zakres temperatur pracy: od -20 do +70 °C

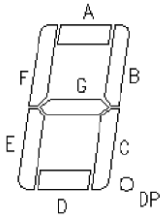
Rys. 5. Wyświetlacz LCD

opis wyprowadzeń:

Nr	Nazwa	Opis
1	VSS	Masa
2	VDD (VCC)	Zasilanie +5V
3	V0	Kontrast
4	RS	Wybór rejestru instrukcji wyświetlacza (stan niski) albo rejestru danych(wysoki)
5	R/W	Odczyt (stan niski) / Zapis (stan wysoki)
6	E	Odblokowanie wyświetlacza
7	DB0	Magistrala danych.
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	LEDA (LED+)	Zasilanie podświetlania +5V
16	LEDK (LED-)	Masa podświetlania

A3.2. WYŚWIETLACZ 8-SEGMENTOWY (DIGIT).

Pojedynczy wyświetlacz 8-segmentowy ze wspólną anodą umożliwia wyświetlanie cyfry wraz z kropką oraz niektórych liter.



Rys. 6. Wyświetlacz 8 segmentowy

A3.3. UKŁAD 74HC595 (U1)

Układ 74HC595 jest rejestrem przesuwnym (8-mio bitowy rejestr przesuwny z wejściem szeregowym) typu SIPO (Serial-In, Parallel-Out) o 8 wyjściach równoległych. Rejestr ten (*ang. shift register*) jest układem sekwencyjnym posiadającym wejście szeregowe i kilka wyjść równoległych. W uproszczeniu działa on w ten sposób, że z każdą zmianą impulsu zegara na wejściu, stany wyjść są przesuwane o jeden. Czyli stan wyjścia 1 jest przepisywany do wyjścia 2, 2 do 3, itd. Stan wejścia 1 jest ustalany na podstawie stanu wejścia szeregowego. Dzięki zastosowaniu układu 74HC595 można powiększyć liczbę dostępnych wejść i wyjść cyfrowych zajmując przy tym tylko trzy piny w module Arduino. Szczegółowy opis układu można znaleźć pod linkiem:
http://www.nxp.com/documents/data_sheet/74HC_HCT595.pdf

A3.4. UKŁAD 74HC595 (U3)

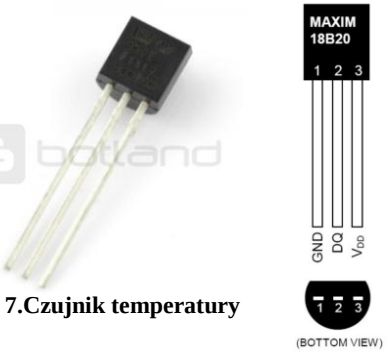
PCF8574 to 8 bitowy port równoległy sterowany magistralą i2c (TWI). Pozwala w prosty sposób zwiększyć ilość pinów cyfrowych w Arduino. Maksymalnie można podłączyć do 8 układów tego typu zyskując w ten sposób 64 dodatkowe piny cyfrowe. Możliwość podłączenia do modułu uzyskujemy poprzez: piny 4 (SDA) i 5 (SCL) z grupy "ANALOG IN" oraz piny 5V i GND z grupy "POWER". Szczegółowy opis układu można znaleźć pod linkiem:
http://www.ti.com/lit/ds/symlink/pcf8574.pdf

A3.5. CZUJNIK TEMPERATURY DS18B20 - CYFROWY 1-WIRE THT (U2)

Cyfrowy termometr w którym napięcie wyjściowe jest proporcjonalne do mierzonej temperatury z zakresu: od -55 °C do +125 °C. Zasilany jest napięciem od 3.0 do 5,5 V.

specyfikacja:

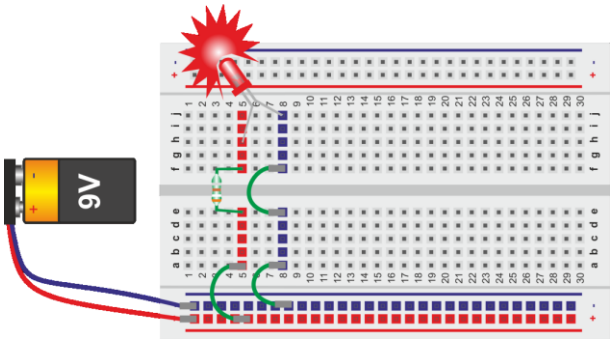
- Napięcie zasilania: od 3,0 V do 5,5 V
- Zakres pomiarowy: od -55 °C do 125 °C
- Dokładność: +/- 0,5 °C w zakresie -10 °C do 85 °C
- Rozdzielczość: od 9 do 12 bitów
- Obudowa THT TO92



Rys. 7.Czujnik temperatury

A4. UNIWERSALNA PŁYTKA MONTAŻOWA

Płytka montażowa wykorzystywana jest do tworzenia i testowania układów elektronicznych bez potrzeby lutowania. Na **Rys. 8** przedstawiono wersję z 400 polami, o wymiarach 82,5 x 53 mm. Bardzo pomocne podczas łączenia układów są kolorowe paski (niebieski i czerwony) umieszczone wzdłuż rzędów z pinami przeznaczonymi do polaryzacji i zasilania.



Rys. 8. Uniwersalna płytka montażowa oraz przykładowy sposób jej wykorzystania

Piny zasilania + i - połączone są w pionie j, natomiast pozostałe rzędy pinów (1-30 lub więcej) są połączone w poziomie. **Rys. 8.** pokazuje też sposób przenoszenia zasilania z rzędu na rząd za pomocą zworek lub elementów elektronicznych (np. rezystorów).

A5. DODATKOWE ELEMENTY WYPOSAŻENIA STANOWISKA

Oprócz standardowego wyposażenia stanowiska laboratoryjnego umieszczonego na jego płycie głównej w ramach ćwiczenia wykorzystywane są także elementy elektroniczne opisane poniżej.

A5.1. DIODY LED

Diody LED służą zazwyczaj do sygnalizacji stanów logicznych oraz wskazują poziom sygnału dla budowanych układów. Poniżej opisano podstawowe parametry wykorzystywanych elementów.

parametry diody żółtej 5 mm:

- Soczewka w kolorze żółtym
- Obudowa: DIP 5mm
- Długość emitowanej fali: 589 nm
- Jasność: 40 mcd
- Kąt świecenia: 60 °
- Parametry pracy:
  - Prąd I_F: 25 mA
  - Napięcie U_F: 2.0 V

parametry diody czerwonej 5 mm:

- Soczewka w kolorze czerwonym
- Obudowa: DIP 5 mm
- Długość emitowanej fali: 625 - 645 nm
- Jasność: 450 - 800 mcd
- Kąt świecenia : 70 °
- Parametry pracy:
  - Prąd I_F: 20 mA
  - Napięcie U_F: 2.0 - 2.3 V

parametry diody zielonej 5 mm:

- Soczewka w kolorze zielonym
- Obudowa: DIP 5 mm
- Długość emitowanej fali: 571 nm
- Jasność: 100 - 150 mcd
- Kąt świecenia: 50 °
- Parametry pracy:
  - Prąd I_F: 20 mA
  - Napięcie U_F : 2.3-2.5 V

Rys. 9. Diody LED

A5.2. REZYSTORY THT 1/4 W 220Ω

Rezystory są wykorzystywane przede wszystkim do zabezpieczenia diod LED. Podczas uruchamiania układów zawsze należy pamiętać o podłączeniu rezystora szeregowo z diodą, ponieważ jej dopuszczalny prąd przewodzenia jest dużo mniejszy od wydajności prądowej zasilania z modułu Arduino !!!

specyfikacja:

- Rezystancja: 220 Ω
- Moc znamionowa: 1/4 W
- Tolerancja: 5 %
- Montaż: przewlekany THT

Rys. 10. Rezystory

A5.3. PRZEWODY POŁĄCZENIOWE I ZWORKI

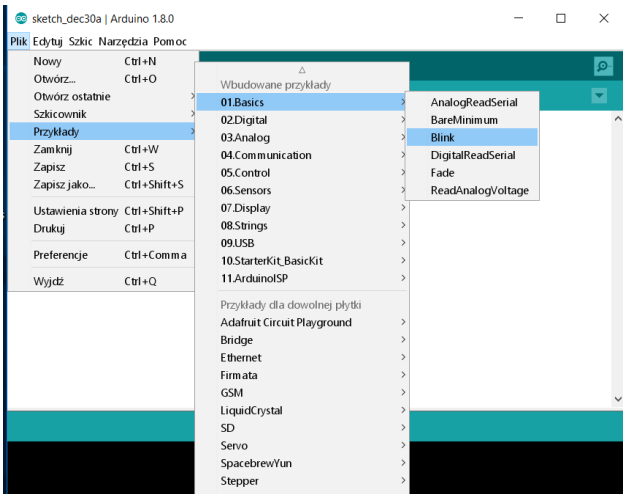
Zestaw przewodów o rożnych długościach do łączenia pól i elementów elektronicznych na uniwersalnych płytkach stykowych (testowych). Przewody są zagięte i pozbawione izolacji na końcach, aby w łatwy sposób można było łączyć wybrane elementy obwodów umieszczone wcześniej na płytkach stykowych. Zworki można zrobić samemu np. ze skrętki komputerowej.

Rys,11. Przewody połączeniowe

B. ĆWICZENIA I ZADANIA.

B.1. URUCHOMIENIE STANOWISKA

Przed przystąpieniem do realizacji zadanych układów wymagane jest sprawdzenie poprawności działania stanowiska laboratoryjnego. W tym celu po podłączeniu płyty Arduino do portu USB należy uruchomić środowisko programistyczne Arduino oraz wgrać na płytę przykładowy program sterujący (patrz **Rys. 12**). Jeśli operacja została przeprowadzona prawidłowo na płycie Arduino powinna migać żółta dioda oznaczona jako „**L**” z częstotliwością 1 Hz (1 raz na sekundę). Aby sprawdzić możliwość programowania modułu należy zmienić kod źródłowy (patrz **Rys. 13**) zaimplementowanego przykładu i wgrywając ponownie program na płytę zaobserwować zaistniałe zmiany. Zmiana ta może polegać na ustawieniu parametru funkcji `delay()` na wartość 2 razy mniejszą (500), co powinno spowodować dwukrotnie szybsze miganie diody. Jak pokazuje przykładowy kod widoczny na **Rys. 13** definiowanie funkcji **`setup()`** polega na określeniu w jej ciele konfiguracji wejść i wyjść modułu Arduino. W naszym przypadku użyto w tym celu funkcji

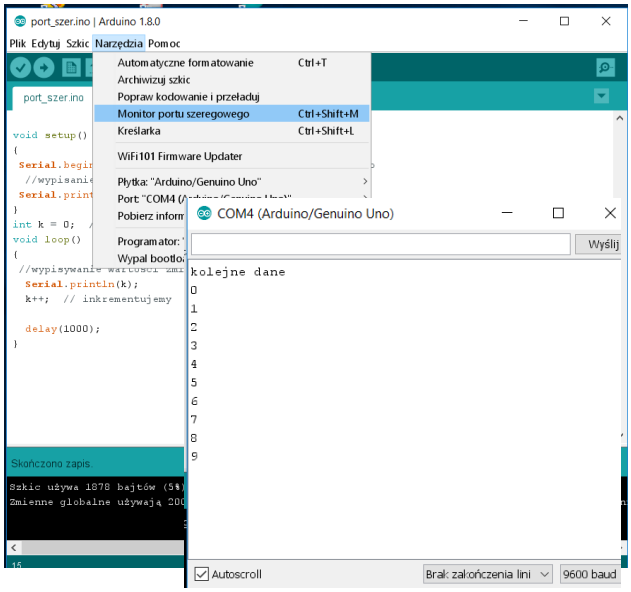


Rys. 12. Wgrywanie przykładowego programu na płytę

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Rys. 13. Fragment przykładowego programu testowego dla moduluArduino

`pinMode(pin,mode)`, która określa tryby pracy poszczególnych pinów. Argument `pin` to numer pinu układu ARDUINO, natomiast `mode` to jedna z predefiniowanych stałych: `INPUT`, `OUTPUT`, `INPUT_PULLUP` związana z wybranym trybem pracy modułu. Treść funkcji **`loop()`** powinna zawierać przebieg procesu działania układu. W podanym przykładzie w jej ciele użyto funkcji `digitalWrite(pin,value)`, która ustawia wartość logiczną dla argumentu określonego w nazwie `pin`. W naszym przypadku wywołanie tej funkcji daje efekt zaświecenia diody dla wartości `value=HIGH` oraz zgaszenia diody dla wartości `value=LOW`, co jest związane z podawaniem na wskazane wyjście odpowiednio napięcia stanu wysokiego (5V) i niskiego (0V). Ponieważ funkcja **`loop()`** jest wywoływana cyklicznie przez cały czas działania programu celem zaobserwowania efektów jego działania użyto funkcji `delay(time)`, która pozwala na chwilowe zatrzymanie działania programu na czas w milisekundach określony przez argument `time`. Pomocnym elementem środowiska programistycznego Arduino jest **monitor portu szeregowego**. Pozwala on na śledzenie danych przesyłanych pomiędzy komputerem a płytą modułu. Aby przetestować jego prawidłowe działanie należy zaimplementować kod programu widocznego na **Rys.14.**

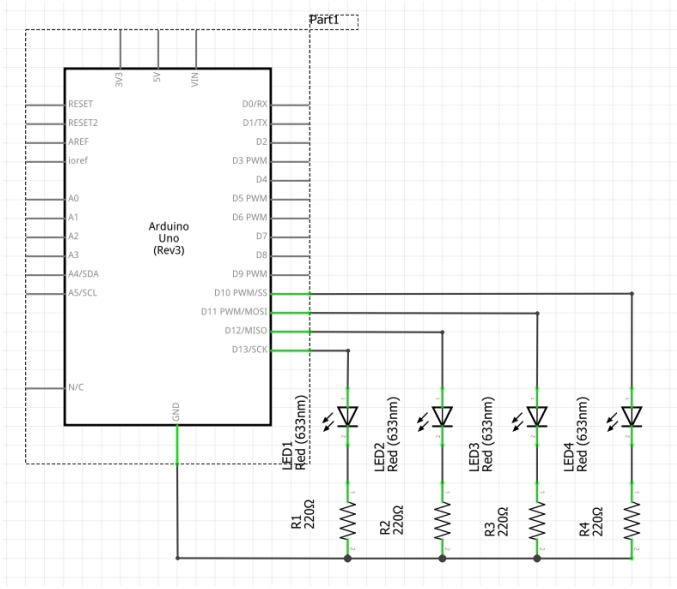



```
void setup()
{
 Serial.begin(9600);   // inicjalizacja portu szeregowego
  //wypisanie tekstu na ekran monitora portu
 Serial.println("kolejne dane");
}

int k = 0;  // inicjalizacja zmiennej do obserwacji

void loop()
{
 //wypisywanie wartości zmiennej k na ekranie monitora
 Serial.println(k);
 k++;  // inkrementujemy zmienną
 delay(1000);  czekamy na kolejny obieg pętli
}
```

Rys. 14. Monitor portu szeregowego i przykładowy kod programu testującego jego działanie

B2. STEROWANIE DIODAMI LED

a) Bazując na doświadczeniach zdobytych w trakcie realizacji punktu **B1** zrealizować układ widoczny na **Rys. 15**. Dla sposobu sterowania diodami określonego przez prowadzącego należy opracować kod źródłowy, następnie uruchomić stworzoną aplikację i przeprowadzić testy sprawdzające poprawność jej działania, po czym udokumentować wykonane zadanie poprzez fotografię realizacji na stanowisku laboratoryjnym.



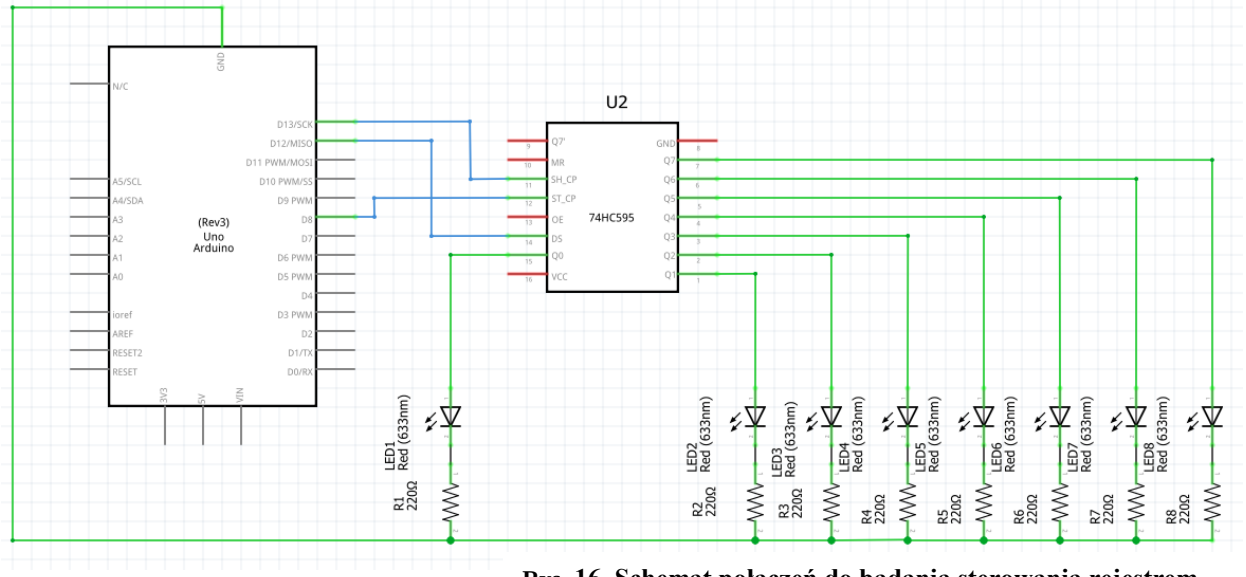
b) Przetestować możliwość sterowania diod przy pomocy innych (wskazanych przez prowadzącego) wyjść. Zmodyfikować w tym celu kod źródłowy programu sterującego i po ponownym wgraniu do modułu sprawdzić efekty jego działania. Udokumentować realizację zadania poprzez odpowiednie adnotacje i zdjęcia.

Do opracowania dokumentacji wykonanych zadań można wykorzystać ogólnodostępne narzędzie o nazwie **fritzing** dostępne pod adresem **http://fritzing.org/home/**. Jest to prosty w obsłudze program, którym można zarówno edytować obwody elektryczne z wykorzystaniem modułów Arduino jak też projektować rozkład przestrzenny elementów zarówno w widoku ogólnym jak też na płytce PCB

Rys. 15. Schemat połączeń do badania sterowania diodami LED

B3. STEROWANIE REJESTREM 74HC595 (U1)

a) Bazując na doświadczeniach zdobytych w trakcie realizacji punktu **B1** i **B2** zrealizować układ widoczny na **Rys. 16**. Następnie dla sposobu sterowania układem u2 określonego przez prowadzącego należy opracować



Rys. 16. Schemat połączeń do badania sterowania rejestrem

```
// Deficja pinu ST_CP dla układu 74HC595
int latchPin = 8;
// Deficja pinu SH_CP dla układu 74HC595
int clockPin = 12;
// Deficja pinu  DS dla układu  74HC595
int dataPin = 11;
void setup()
{
  //konfiguracja wyjść modułu
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}
void loop() {
 for (int j=0; j<256; j++)
  //ustawienie zezwolenia wpisu informacji
     digitalWrite(latchPin, LOW);
  // wczytanie bajtu informacji
     shiftOut(dataPin, clockPin, MSBFIRST, j);
  // zatrzasnięcie informacji w rejestrze
     digitalWrite(latchPin, HIGH);}
```

Rys. 17. Przykładowy program do sterowania rejestrem

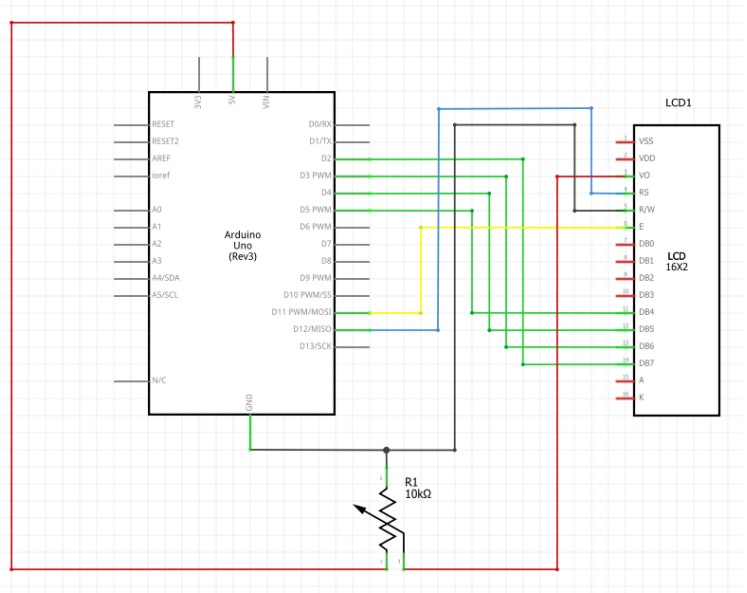
kod źródłowy programu sterującego, wzorując się na przykładzie podanym na **Rys. 17.** W kolejnym etapie wgrać i uruchomić stworzoną aplikację a następnie przeprowadzić testy sprawdzające poprawność jej działanie. W przypadku niewłaściwego działania sterowania skorygować odpowiednio program celem eliminacji widocznych błędów. Po zakończeniu testów udokumentować wykonane zadanie poprzez fotografię realizacji na stanowisku laboratoryjnym.

b) Wykorzystać monitor portu szeregowego do obserwacji danych wysyłanych do badanego rejestru. Zmodyfikować w tym celu kod źródłowy programu sterującego a następnie wgrać i uruchomić zmodyfikowany program. Po pozytywnym zakończeniu testów udokumentować jego działanie poprzez kopie ekranów monitora.

c) Wykorzystać dostępny wyświetlacz 8-SEGMENTOWY do wizualizacji danych zapamiętywanych przez rejestr.

B4.STEROWANIE WYŚWIETLACZEM LCD

a) Połączyć układ widoczny na **Rys. 18**. Następnie dla sposobu sterowania układem LCD określonego przez prowadzącego bazując na kodzie źródłowym należy opracować odpowiedni program sterujący oraz przetestować i udokumentować jego działanie



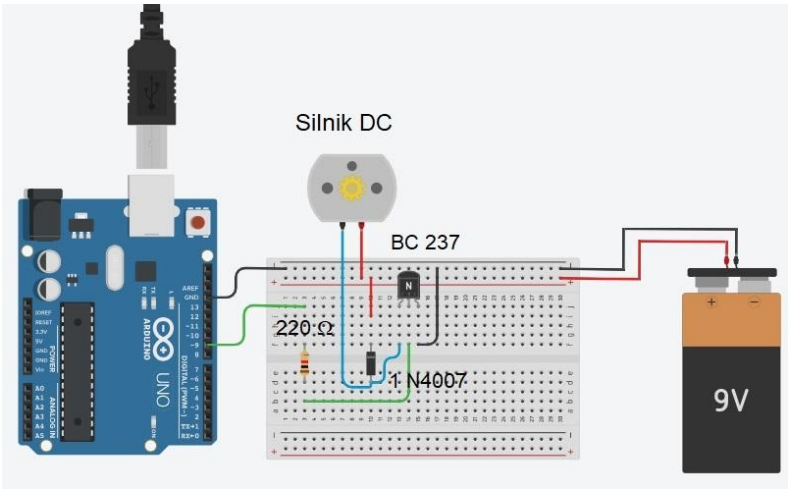

```
// dołaczenie bblioteki obsługującej wyświetlacz LCD
#include <LiquidCrystal.h>

// konfiguracja pinów modułu Arduino
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // okreslenie liczby wierszy i kolumn wyświetlacza LCD
  lcd.begin(16, 2);
  // wypisanie komunikatu tekstowego.
  lcd.print("Ekran LCD");
}

void loop() {
  // ustawienie kursora na 0 wiersz i 1 kolumnę
  lcd.setCursor(0, 1);
  // ustawienie czasu wyświetlania do resetu ekranu
  lcd.print(millis()/1000);
}
```

Rys. 18. Sposób podłączenia wyświetlacza LCD i przykładowy program do sterowania przy pomocy modułu Arduino

b) Napisać kod źródłowy do sterowania wyświetlaczem LCD z wykorzystaniem monitora portu szeregowego. Przetestować jego działanie i udokumentować wyniki.

B5. STEROWANIE SILNICZKIEM PRĄDU STAŁEGO

a) Połączyć układ widoczny na **Rys. 19**. Następnie przetestować i udokumentować jego działanie dla przykładowego kodu podanego obok schematu.



Silnik_DC

```
// Pin PWM do sterowania prędkością silnika
const int motorPin = 9;

void setup() {
  // Ustaw pin PWM jako wyjściowy
  pinMode(motorPin, OUTPUT);
}

void loop() {
  // Przykładowe sterowanie prędkością silnika
  for (int speed = 0; speed <= 255; speed += 5) {
    analogWrite(motorPin, speed);
    delay(100); // Czekaj przez 100 ms
  }

  for (int speed = 255; speed >= 0; speed -= 5) {
    analogWrite(motorPin, speed);
    delay(100); // Czekaj przez 100 ms
  }
}
```

Rys. 19. Sposób podłączenia układu sterowania silniczkiem DC przy pomocy modułu Arduino

b) Zmodyfikować układ zgodnie z wytycznymi prowadzącego i udokumentować jego działanie

C) OPRACOWANIE WYNIKÓW

- Opracowanie i realizacja punktów B2 (ocena 3.0)
- Opracowanie i realizacja punktów B2 + B3 a) (ocena 3.5)
- Opracowanie i realizacja punktów B2 + B3 b) c) (ocena 4.0)
- Opracowanie i realizacja punktów B2 + B3 +B4 (ocena 4.5)
- Opracowanie i realizacja punktów B2 + B3 +B4 + B5 (ocena 5.0)

D) LITERATURA

1. http://e-learning.prz.edu.pl (materiały wykładowe)
2. https://www.arduino.cc/
3. http://fritzing.org/home/.
4. https://www.tinkercad.com/circuits