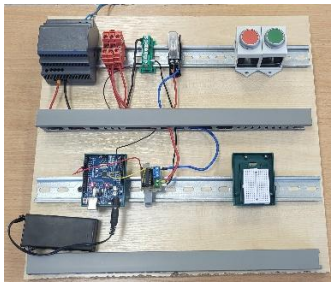
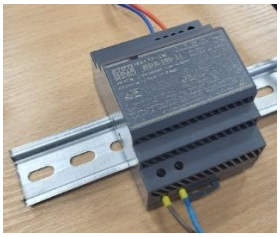
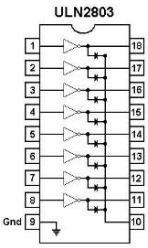
Ćwiczenie 33

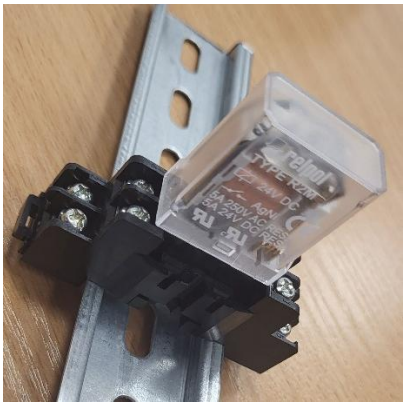
Temat: "Interfejs Mocy" – Arduino steruje przekaźnikiem 24V

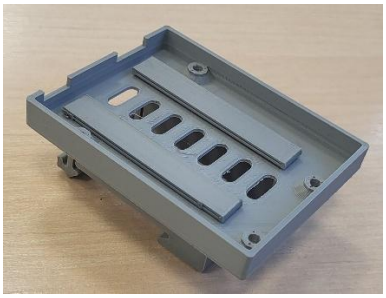
Cel ćwiczenia: Cel ćwiczenia: Zbudowanie pierwszego układu na płycie montażowej z szynami DIN. Zrozumienie, jak odseparować logikę (Arduino 5V) od obwodu wykonawczego (Przekaźnik 24V) za pomocą bezpiecznego interfejsu **ULN2803**.







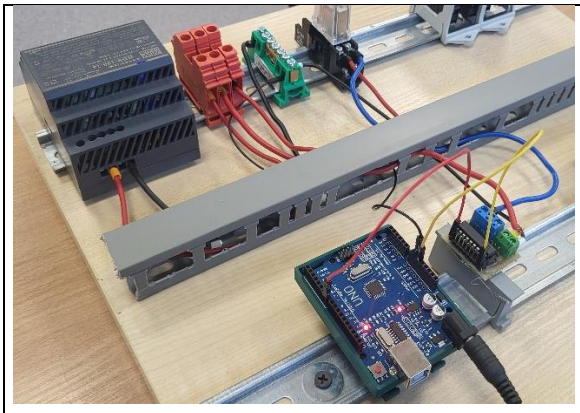


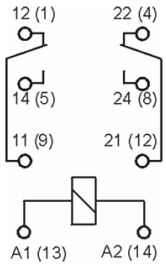


Potrzebne komponenty:

- Płytka montażowa z szynami DIN (opcjonalnie, można na stole)
- Arduino UNO
- Zasilacz **24V DC**
- Płytka stykowa
- Układ scalony **ULN2803** (driver 8-kanałowy)
- Przekaźnik (model: **Relpol R2M 24V DC**) wraz z podstawką na szynę DIN
- Czujnik indukcyjny (model: **E2S-H4N1 5V NPN**)
- Przewody połączeniowe (cienkie do płytki stykowej, grubsze do 24V)
- Metalowy element do testowania czujnika

Część 1: Podstawowy obwód i kod – Sterowanie cewką 24V





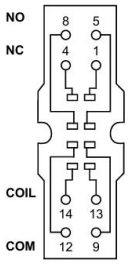
2P - przełączne

GZ2

Do R2M

Z zaciskami śrubowymi
Maksymalny moment dokręcenia zacisku: 0,7 Nm
Montaż na szynie 35 mm wg PN-EN 60715 lub na płycie
65,2 x 20 x 25 mm
Dwa tory prądowe
7 A, 250 V AC

Schemat połączeń

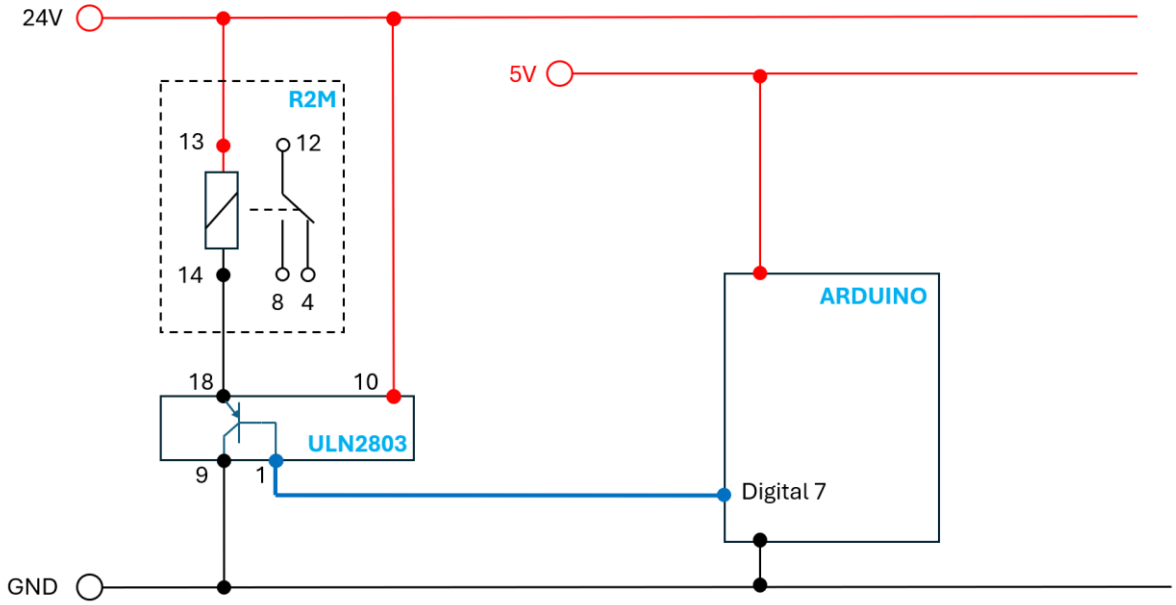


Akcesoria

Zbudujemy układ, w którym pin Arduino będzie bezpiecznie "cykał" przemysłowym przekaźnikiem 24V.

Obwód

1. **Montaż na płycie**:
   - Zamontuj podstawkę przekaźnika na szynie DIN. Włóż przekaźnik Relpol.
   - Ustaw zasilacz 24V DC obok.
2. **Układ na płytce stykowej**:
   - Umieść układ ULN2803 na płytce stykowej.
   - Podłącz pin **GND** (Pin 9) układu ULN2803 do szyny **GND** na płytce stykowej.
   - Podłącz szynę **GND** płytki stykowej do pinu **GND** Arduino. (**Ważne: wspólna masa dla logiki**).
3. **Obwód sterowania (Logika 5V)**:
   - Podłącz pin wyjściowy Arduino (np. Pin 7) do wejścia interfejsu (np. Pin 1 / 1B układu ULN2803).
4. **Obwód wykonawczy (Moc 24V):**
   - Podłącz **+24V** z zasilacza do jednego z zacisków cewki przekaźnika (np. **13**).
   - Podłącz drugi zacisk cewki przekaźnika (np. **14**) do wyjścia interfejsu (np. Pin 18 / 1C układu ULN2803).
   - Podłącz **0V (GND)** z zasilacza 24V do szyny **GND** płytki stykowej (wspólna masa).
   - **WAŻNE (Zabezpieczenie ULN)**: Podłącz pin COM (Pin 10) układu ULN2803 do **+24V** z zasilacza. (To aktywuje wbudowane diody tłumiące przepięcia z cewki).



Kod (Arduino IDE - C++)

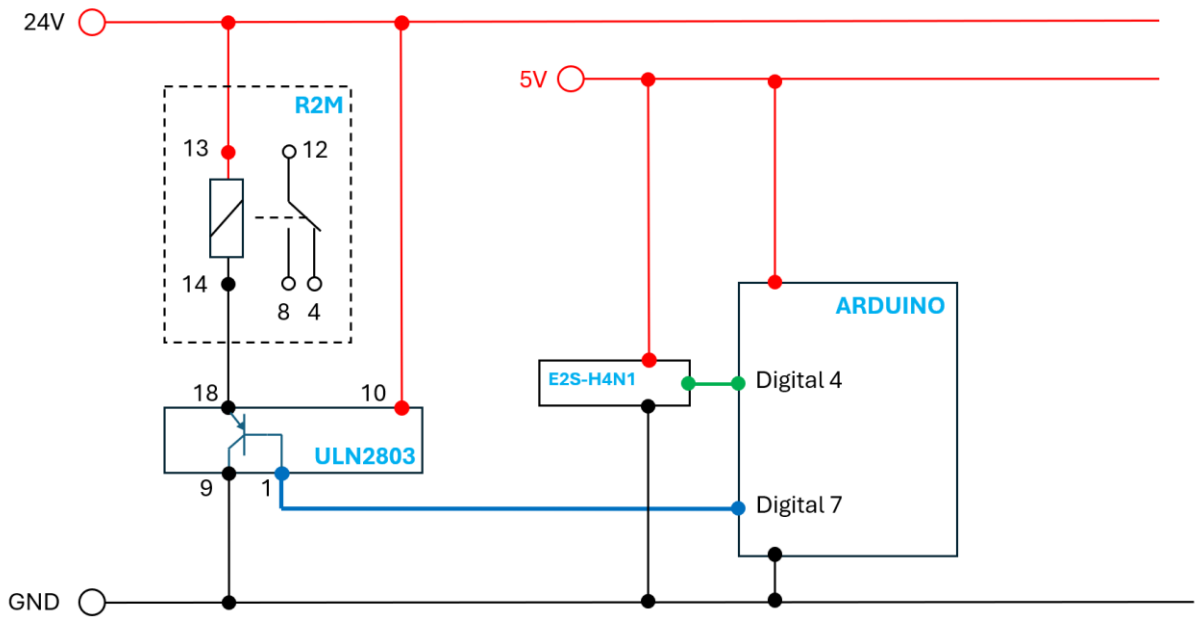
To ćwiczenie ma dwa oddzielne obwody napięciowe (5V i 24V), które spotykają się na układzie ULN2803. **Wspólna masa (GND/0V) jest kluczowa!**

```
/*
 * ZADANIE 33: TEST INTERFEJSU ULN2803
 */

const int pinSterowaniaPrzekaznikiem = 7; // Pin Arduino podłączony do wejścia ULN2803

void setup() {
  pinMode(pinSterowaniaPrzekaznikiem, OUTPUT);
}

void loop() {
  // Załącz przekaźnik (usłyszysz "CYK")
  digitalWrite(pinSterowaniaPrzekaznikiem, HIGH);
  delay(1000); // Czekaj 1 sekundę

  // Wyłącz przekaźnik (usłyszysz "CYK")
  digitalWrite(pinSterowaniaPrzekaznikiem, LOW);
  delay(1000); // Czekaj 1 sekundę
}
```

Część 2: Samodzielna rozbudowa

Zadanie 33.1: Integracja czujnika (5V -> 24V)

Cel: Zbudowanie kompletnego łańcucha sterowania. Program na Arduino ma odczytywać sygnał z przemysłowego czujnika 5V (E2S-H4N1 5V NPN) i na jego podstawie sterować przemysłowym przekaźnikiem 24V za pomocą bezpiecznego interfejsu (drivera) ULN2803.



Czujnik E2S-H4N1 5V NPN (Normalnie Otwarty - NO) działa jak przełącznik podłączony do masy (GND). Gdy niczego nie wykrywa, jego wyjście "wisi w powietrzu" (stan nieustalony). Gdy wykryje metal, zwiera wyjście do GND. Aby poprawnie to odczytać, musimy użyć wewnętrznego rezystora "podciągającego" w Arduino.

Obwód

To ćwiczenie ma dwa oddzielne obwody napięciowe (5V i 24V), które spotykają się na układzie ULN2803. Wspólna masa (GND/0V) jest kluczowa!

- **Obwód Logiki (5V - Arduino i Sensor):**
  - Podłącz Arduino do komputera przez USB.
  - **Zasilanie czujnika E2S-H4N1 5V NPN:**
    - Przewód **Brązowy** -> pin 5V na Arduino.
    - Przewód **Niebieski** -> pin **GND** na Arduino.
  - **Sygnał czujnika E2S-H4N1 5V NPN:**
    - Przewód **Czarny** -> pin cyfrowy 4 na Arduino.
  - **Zasilanie Interfejsu (Logika):**
    - Umieść ULN2803 na płytce stykowej.
    - Podłącz pin **GND** (Pin 9) układu ULN2803 do szyny GND na płytce stykowej.
    - Połącz szynę **GND** płytki stykowej z pinem **GND** Arduino.
- **Obwód Wykonawczy (24V - Przekaźnik):**
  - Włącz zasilacz 24V DC.
  - Podłącz wyjście **0V (Masa)** zasilacza 24V do szyny **GND** na płytce stykowej (tej samej, do której podłączone jest **GND** Arduino i **GND** układu ULN2803). **TO JEST KROK KRYTYCZNY (Wspólna Masa).**
  - Podłącz wyjście **+24V** zasilacza do zacisku cewki przekaźnika R2M (na podstawce oznaczony jako 13).
  - Podłącz pin **COM** (Pin 10) układu ULN2803 do +24V z zasilacza (to aktywuje wbudowane diody zabezpieczające cewkę).
- **Połączenie Interfejsu (Sterowanie):**
  - Podłącz pin wyjściowy Arduino (np. Pin7) do wejścia interfejsu (np. Pin 1 / 1B układu ULN2803).
  - Podłącz wyjście interfejsu (np. Pin 18 / 1C) do drugiego zacisku cewki przekaźnika R2M (na podstawce oznaczony jako 14).

Zmodyfikuj kod tak, aby zbliżenie metalowego obiektu do czujnika 5V powodowało załączenie przekaźnika 24V.

Ćwiczenie 33.2: Implementacja Timera Nieblokującego (Impuls Monostabilny)

Cel: Nauczenie się fundamentalnej techniki programowania w automatyce: tworzenia timera, który wykonuje zadanie przez określony czas bez blokowania reszty programu. Zbudujesz układ, w którym czujnik działa tylko jako wyzwalacz 3-sekundowego cyklu pracy przekaźnika.

Obwód jest identyczny jak w ćwiczeniu 33.1. Nie wymaga żadnych modyfikacji. Logika w całości realizowana jest w kodzie.

Polecenie (Logika do zaimplementowania):

Twoim zadaniem jest napisanie programu od nowa (lub zmodyfikowanie poprzedniego), który będzie realizował następującą logikę:

1. Program ma czekać na pojawienie się detalu przed czujnikiem **E2S-H4N1 5V NPN**.
2. Wykrycie detalu (czyli zmiana stanu na pinie czujnika z **HIGH** na **LOW**) ma działać jak pojedynczy wyzwalacz (trigger).
3. W momencie wykrycia detalu, program ma natychmiast załączyć przekaźnik R2M (podając stan **HIGH** na pin sterujący ULN2803).
4. Przekaźnik R2M ma pozostać włączony przez **dokładnie 3 sekundy**.
5. Po upływie 3 sekund, program ma **automatycznie wyłączyć** przekaźnik R2M – **nawet jeśli detal nadal znajduje się przed czujnikiem**.
6. System ma być gotowy na wykrycie *następnego* detalu dopiero po zakończeniu 3-sekundowego cyklu.

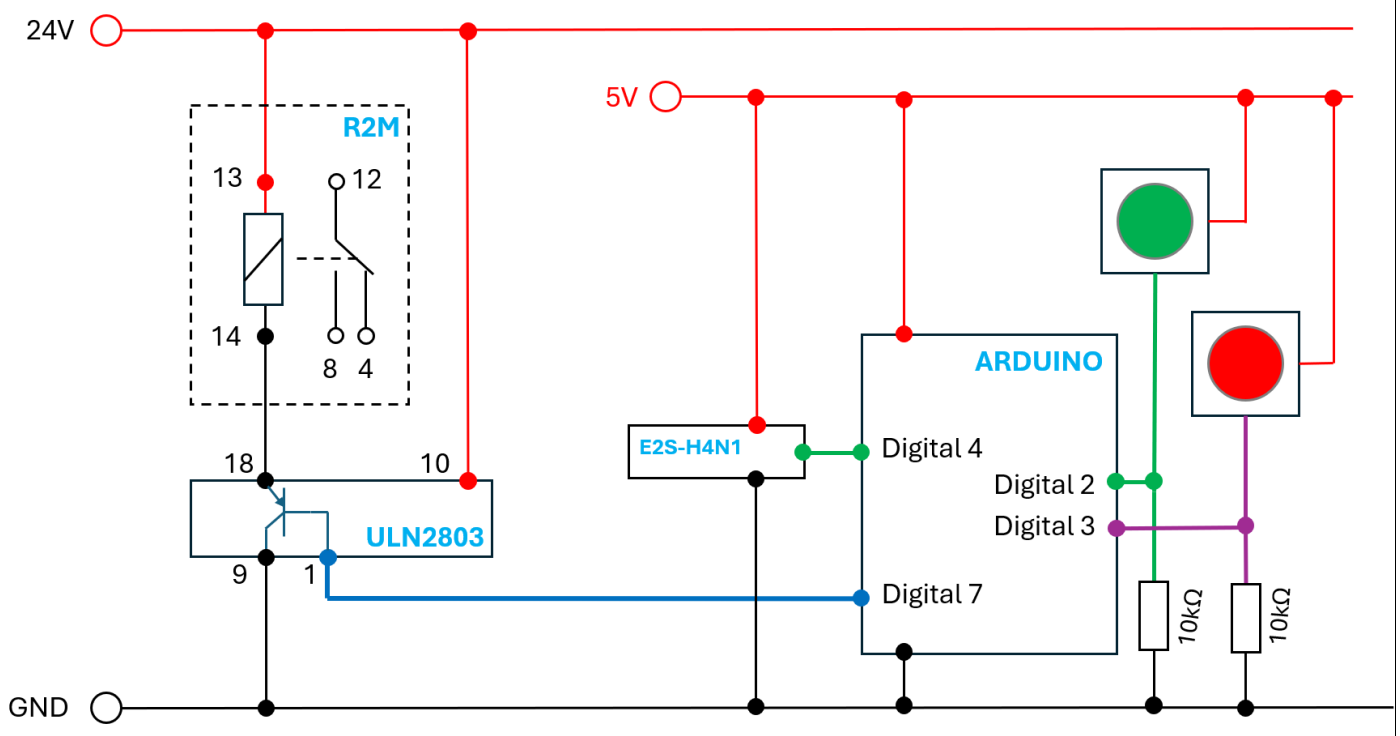
Wymaganie Krytyczne (Jak to zrobić):

Do odmierzenia 3 sekund **absolutnie nie wolno** używać funkcji *delay(3000);*.
Funkcja *delay()* zatrzymuje cały program, co jest niedopuszczalną praktyką w systemach sterowania (w tym czasie program nie mógłby np. reagować na inne zdarzenia).
Zamiast tego musisz zaimplementować timer nieblokujący oparty na funkcji *millis()*:

1. Będziesz potrzebować zmiennej globalnej do zapisania "czasu startu", np. *unsigned long czasStartuPracy;*.
2. Będziesz też potrzebować "flagi" (zmiennej *bool*), która informuje program, czy timer właśnie odlicza, np. *bool timerJestAktywny = false;*.
3. Gdy wykryjesz detal (i timer nie jest aktywny), ustawiasz *timerJestAktywny = true;* i zapisujesz czas: *czasStartuPracy = millis();*.
4. W pętli *loop()*, jeśli *timerJestAktywny* jest *true*, ciągle sprawdzasz warunek: *if (millis() - czasStartuPracy >= 3000)*.
5. Gdy warunek zostanie spełniony, wyłączasz przekaźnik i resetujesz flagę: *timerJestAktywny = false;*.

Zadanie 33.3: Pełny automat START/STOP/CZUJNIK

Cel: Zbudowanie kompletnego, hybrydowego automatu sterującego (logika 5V, wykonanie 24V). Program będzie implementował pełny cykl przemysłowy: logikę START/STOP, sekwencję pracy z timerem oraz licznik cykli. Jest to symulacja prostej linii produkcyjnej.



- Dodaj do wejść Arduino dwa przyciski panelowe 22mm (START/STOP) zamontowane na szynie DIN (podłączone do 5V Arduino).

  Dodaj przycisk S1 (START):

  - Podłącz jeden zacisk do **5V** na Arduino.
  - Podłącz drugi zacisk do pinu cyfrowego **2** ORAZ przez rezystor **10kΩ** do **GND** (układ pull-down).

  Dodaj przycisk S2 (STOP):

  - Podłącz jeden zacisk do **5V** na Arduino.
  - Podłącz drugi zacisk do pinu cyfrowego **3** ORAZ przez rezystor **10kΩ** do **GND** (układ pull-down).

Twoim zadaniem jest napisać program, który będzie działał jako **automat stanów (State Machine)**. Program musi realizować następującą, precyzyjną sekwencję:

1. **Stan 0: ZATRZYMANY (STOP)**
   - To jest stan domyślny po uruchomieniu programu.
   - Przekaźnik R2M musi być **wyłączony**.
   - Program w tym stanie czeka **tylko** na naciśnięcie przycisku **S1 (START)**.
   - Po naciśnięciu START, program ma przejść do Stanu 1 (GOTOWY) i zresetować licznik cykli do 0.
2. **Stan 1: GOTOWY (Czeka na detal)**
   - Program jest "uzbrojony".
   - Przekaźnik R2M jest **wyłączony**.
   - Program w tym stanie czeka na sygnał z czujnika **E2S-H4N1 5V NPN** (wykrycie zbocza HIGH -> LOW).
   - Po wykryciu detalu, program ma przejść do Stanu 2 (PRACA).
3. **Stan 2: PRACA (Timer 3s aktywny)**
   - W momencie wejścia w ten stan, program musi **natychmiast włączyć** przekaźnik **R2M**.
   - Program musi uruchomić **nieblokujący timer (na millis())** ustawiony na **3 sekundy**.
   - Przekaźnik R2M pozostaje włączony przez te 3 sekundy.
   - Po upływie 3 sekund, program musi **wyłączyć** przekaźnik R2M, **zwiększyć licznik cykli** o 1 i sprawdzić limit.
   - Jeśli limit (5 cykli) **nie został** osiągnięty, program wraca do Stanu 1 (GOTOWY) i czeka na kolejny detal.
   - Jeśli limit (5 cykli) **został** osiągnięty, program przechodzi do Stanu 0 (STOP) i czeka na ponowne naciśnięcie START.

4. **Funkcja Globalna: STOP**
    - Przycisk **S2 (STOP)** musi być sprawdzany w każdej pętli loop() i mieć **najwyższy priorytet**.
    - Naciśnięcie S2 w **dowolnym momencie** (czy to w stanie GOTOWY, czy w trakcie 3-sekundowej PRACY) musi **natychmiast** przerwać cykl, wyłączyć przekaźnik R2M i ustawić program w Stanie 0 (ZATRZYMANY).

Pliki: [zad33-1.INO, zad33-2.INO, zad33-3.INO] przesłać na haukeprojekt@gmail.com ,
w tytule podać „**3XY(Z)_cwiczenie33**” gdzie X=klasa, Y=grupa, Z=numer_z_dziennika