REV	DATA	ZMIANY
0.1	26.05.2024	*Maciej Zieliński*

System pomiaru temperatury w obudowie drukarki 3D

Autor: Maciej Zieliński
Akademia Górniczo-Hutnicza
Elektronika i Telekomunikacja – Systemy wbudowane
Nr. Indeksu: 407530

Kraków, 2024

Spis treści

Lista oznaczeń

ABS	Akrylonitrylo-Butadieno-Styren
CRC	Cyclic Redundancy Code
IDE	Integrated Development Environment
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LSB	Least Significant Bit
MSB	Most Significant Bit
PCB	Printed Circuit Board
THT	Through-Hole Technology
UTP	Unshielded Twisted Pair
--	--

1. Wstęp

Celem projektu było stworzenie i wdrożenie prostego systemu do monitorowania temperatury w obudowie drukarki 3D. Potrzeba stworzenia takiego systemu wynika głównie z wrażliwości niektórych materiałów termoplastycznych na skokowe zmiany temperatury, jak np. filament ABS. Takie zmiany powodują problem polegający na odrywaniu się wydruku od stołu roboczego, kurczeniem się materiału w trakcie wydruku oraz nie trwałym połączeniem warstw. Obudowa pozwala na utrzymanie stałej temperatury otoczenia podczas druku oraz zapewnia ochronę przed np. nie pożądanymi podmuchami powietrza. Natomiast stworzony system pozwala na kontrolowanie temperatury wewnątrz takiej obudowy. Wykorzystanie stworzonego układu przyczynia się do zwiększenia niezawodności i jakości wydruków 3D.

2. Wymagania systemowe

Głównymi założeniami projektu są:

- Zaprojektowanie układu do pomiaru temperatury z 4 czujników. Utworzenie schematu ideowego.
- Wytworzenie prototypu PCB układu wraz z montażem końcowym elementów.
- Implementacja oprogramowania na mikrokontroler do odczytu danych z czujników temperatury i wyświetlania ich na wyświetlaczu 7-segmentowym.
- Pomiar z 4 czujników temperatury rozłożonych w różnych punktach pomiarowych wewnątrz obudowy drukarki 3D.
- Możliwość pomiaru i wyświetlenia temperatury w zakresie 0-99 °C z dokładnością do 0.1 °C.
- Minimalizacja rozmiarów rzeczywistych PCB.

3. Funkcjonalność

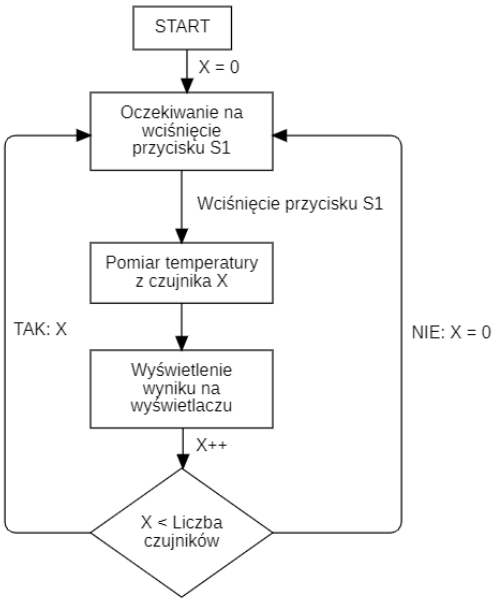
Głównym zadaniem zaprojektowanego układu jest pomiar temperatury w dwóch trybach:

- **Pojedynczy pomiar temperatury z jednego czujnika.**

Uproszczony schemat blokowy pomiaru dla tego trybu przedstawiono na Rys. 1.

Układ oczekuje na przerwanie wywołane wciśnięciem przycisku S1. Następnie dokonuje pomiaru z czujnika określonego numerem X i wyświetla odczytaną wartość na wyświetlaczu. Przy każdym wciśnięciu przycisku S1 numer X jest inkrementowany. Jeśli wartość X przekroczy maksymalną liczbę czujników zostaje wyzerowana.

Dodatkowo wbudowane diody LED na płytce PCB sygnalizują, z którego czujnika wyświetlana jest aktualnie temperatura.



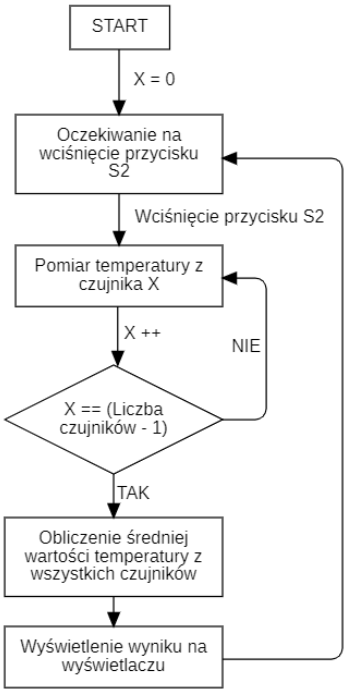
Rys. 1: Schemat blokowy – pojedynczy pomiar temperatury

- **Pomiar średniej wartości z wszystkich czujników**

Uproszczony schemat blokowy pomiaru dla tego trybu przedstawiono na Rys. 2.

Układ oczekuje na przerwanie wywołane wciśnięciem przycisku S2. Następnie dokonuje pomiaru ze wszystkich czujników. Po odczytaniu wszystkich wartości obliczana jest wartość średnia z uzyskanych wyników. Wynik obliczeń zostaje wyświetlony na wyświetlaczu.

Dodatkowo na płytce PCB znajduje się dioda sygnalizująca pracę w trybie pomiaru średniej wartości temperatury.



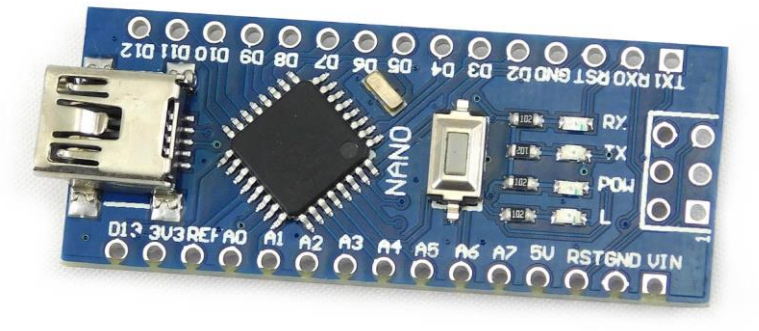
Rys. 2: Schemat blokowy – pomiar średniej wartości temperatury

4. Projekt Techniczny

4.1. Dobór elementów

- **Arduino Nano**

Jako mikrokontroler wybrano klon Arduino Nano przedstawiony na Rys. 3. Wybrano ten układ ze względu na prostotę obsługi, dużą ilość gotowych bibliotek oraz niską cenę. Dodatkowym atutem jest niewielki rozmiar układu, który pozwala zminimalizować wielkość płytki PCB. Arduino Nano jest wystarczające dla projektu pod kątem wydajności oraz ilości pinów wejściowych/wyjściowych, więc nie ma konieczności stosowania bardziej zaawansowanych mikrokontrolerów np. z rodziny STM32.



Rys. 3: Arduino Nano. Źródło [1]

- **Czujnik temperatury DS18B20**

Jako czujnik temperatury wybrano cyfrowy termometr DS18B20. Czujnik w obudowie TO92 posiada 3 wyprowadzenia: VDD (zasilanie), GND (masa) oraz DQ (sygnał cyfrowy). Charakteryzuje się niską ceną oraz zakresem pomiarowym od -55°C do 125°C, więc w pełni obejmuje zakres z założeń projektowych. Do komunikacji z mikrokontrolerem wykorzystywany jest interfejs 1-Wire, który pozwala na zredukowanie liczby potrzebnych przewodów oraz pinów w mikrokontrolerze do realizacji połączeń. Rozdzielczość pomiaru jest konfigurowalna przez użytkownika i może wynosić 9, 10, 11 lub 12 bitów, czyli odpowiednio: 0.5°C, 0.25°C, 0.125°C, 0.0625°C. W projekcie zastosowano 12-bitową rozdzielczość czujnika, aby finalnie uzyskać rozdzielczość 0.1°C na wyświetlaczu.

- **Wyświetlacz BA56-12SRWA**

Wyświetlacz BA56-12SRWA firmy Kingbright jest to wyświetlacz LED 7-segmentowy ze wspólną anodą umożliwiający wyświetlanie 3 cyfr wraz z kropkami. Zaletą wybranego rozwiązania jest mały rozmiar oraz niska cena w porównaniu z wyświetlaczami LCD. Natomiast do wad wyświetlacza LED zalicza się konieczność wykorzystania dużej ilości pinów wyjściowych mikrokontrolera oraz zastosowania rezystorów ograniczających prąd dla diod LED w wyświetlaczu.

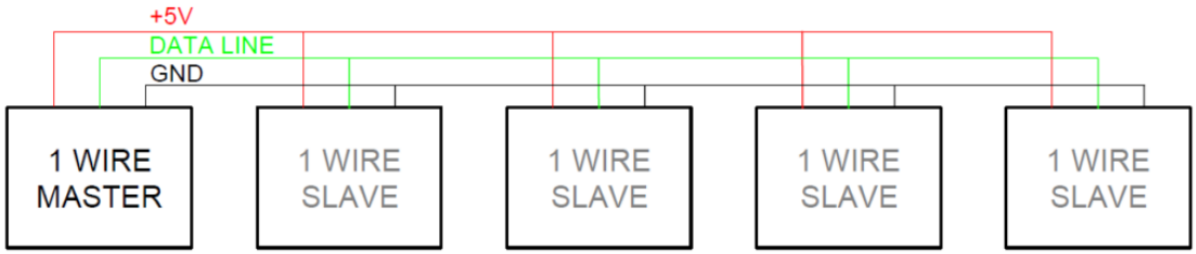
- **Pozostałe komponenty**

Dodatkowymi komponentami wykorzystanymi w projekcie są:

  - Diody LED o średnicy 5mm sygnalizujące, z którego czujnika wyświetlany jest aktualnie pomiar.
  - Przyciski monostabilne służące do wykonywania pomiarów z kolejnych czujników.
  - Rezystory THT 470 Ω służące do ograniczenia prądu płynącego przez diody LED
  - Rezystor THT 4.7 kΩ rezystor podciągający dla linii DQ (czujnik DS18B20)

4.2. Interfejs 1-Wire

Odczyt wartości temperatury z czujników wykonywany jest przy użyciu interfejsu 1-Wire. W projekcie stworzono sieć 1-Wire o topologii liniowej zaprezentowanej na Rys. 4. Gdzie urządzeniem typu MASTER jest mikrokontroler, a urządzeniami typu SLAVE są czujniki temperatury. Topologia zakłada, że czujniki podłączone są bezpośrednio do magistrali lub za pomocą krótkich odgałęzień (krótszych niż 3m). W przypadku tego projektu każdy z czujników dołączony jest do magistrali za pomocą przewodu o długości ok. 160 cm.



Rys. 4: Topologia liniowa sieci 1-Wire. Źródło [2]

Kolejnym aspektem był dobór przewodów. W projekcie wybór padł na nieekranowaną skrętkę CAT5 UTP, ze względu na niską pojemność i rezystancje żył. Jako kabel do czujnika użyto 2 skręconych par przewodów. W jednej z skręconych par znajduje się linia danych i masy GND, natomiast w drugiej parze jeden przewód to zasilanie +5V, a drugi pozostał nie podłączony.

4.3. Schemat połączeń

Pełny schemat układu przedstawiono na Rys. 5. Na schemacie można wyróżnić 3 główne bloki: Arduino Nano, wyświetlacz 7-segmentowy oraz blok czujników.

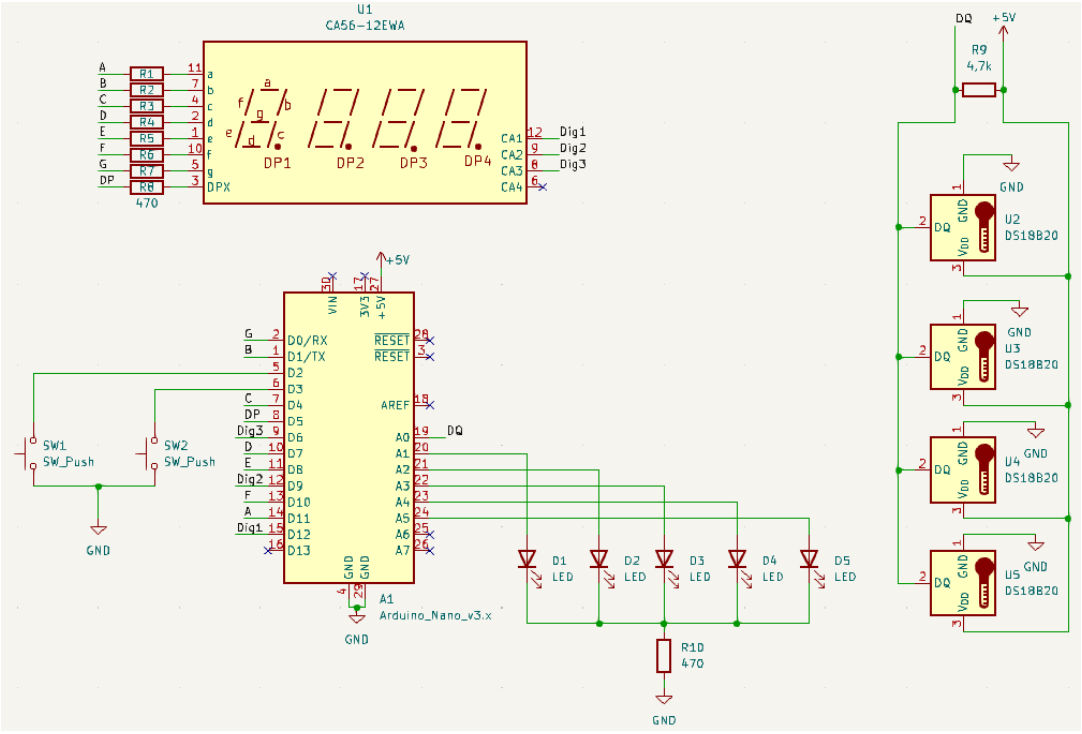
Ze względu na brak wyświetlacza BA56-12SRWA w programie Kicad wykorzystano wyświetlacz CA56-12EWA, który posiada identyczne wyprowadzenia jak w przypadku wykorzystywanego elementu docelowego. Wyprowadzenie dla czwartej cyfry jest nie podłączone. Katody diod LED w wyświetlaczu podłączone są przez rezystor 470Ω do pinów cyfrowych mikrokontrolera.

Czujniki DS18B20 połączone zostały równolegle do wspólnej linii sygnałowej DQ.

Przyciski monostabilne połączone zostały do pinów cyfrowych D2 i D3, które umożliwiają obsługę przerwań zewnętrznych w zastosowanym mikrokontrolerze.

Analogowe piny A0-A5 skonfigurowane zostały programowo jako piny cyfrowe.

Rezystor ograniczający prąd R10 jest wspólny dla wszystkich diod D1-D5 ze względu na to, że w jednym momencie świeci tylko jedna z diod, więc nie ma konieczności zastosowania osobnych rezystorów dla każdej z diod. Pozwala to na zredukowanie liczby elementów i połączeń.



Rys. 5: Schemat połączeń układu.

4.4. Projekt PCB

Projekt płytki PCB przedstawiono na Rys. 6. Płytka jednowarstwowa o wymiarach 79.6mm x 79.8mm. Dodatkowo na głównej warstwie wygenerowano pole masowe w celu ograniczenia wydzielania ciepła i eliminacji zakłóceń. Do podłączenia mikrokontrolera oraz czujników wykorzystano złącza typu goldpin, które umożliwiają szybkie odłączenie/podłączenie czujników/mikrokontrolera od układu.

Na Rys. 7 oraz Rys. 8 przedstawiono wygląd płytki przed montażem elementów oraz po montażu. W tabeli Tab. 1 przedstawiono parametry techniczne PCB:

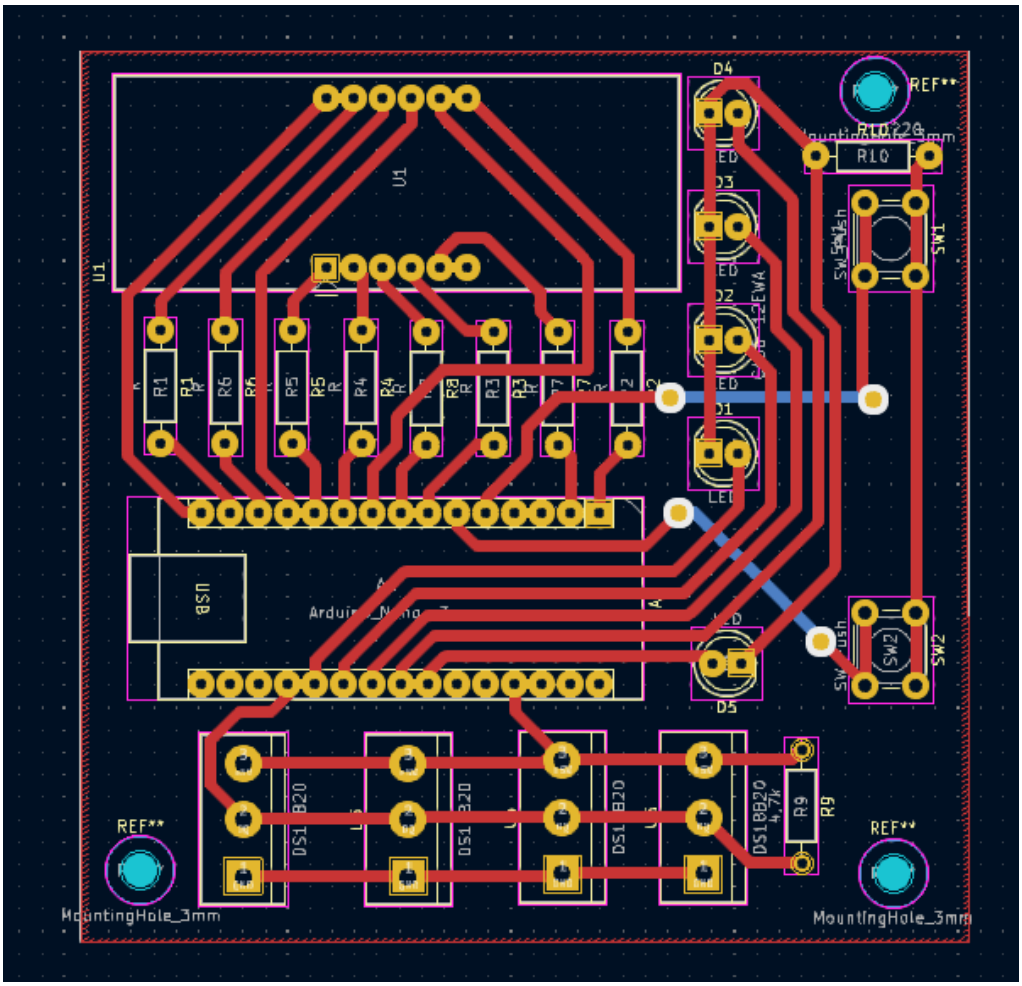
Tab. 1: Parametry techniczne płytki PCB.

Wymiary [mm x mm]	79.6 x 79.8
Typ laminatu	FR4
Laminat	Jednostronny
Grubość laminatu [mm]	1.6
Grubość warstwy miedzi [mm]	0.035
Szerokość ścieżek [mm]	1
Technologia montażu elementów	THT

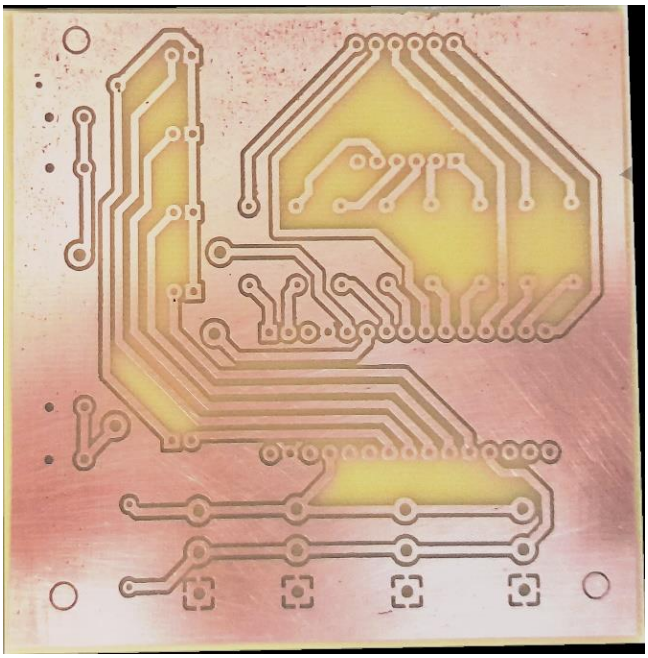
W tabeli Tab. 2 przedstawiono spis wykorzystanych elementów:

Tab. 2: Spis wykorzystanych elementów.

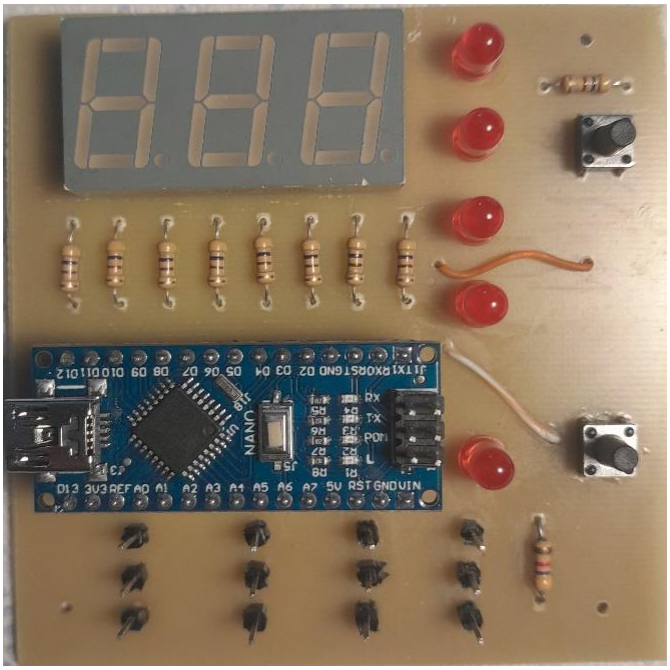
Nazwa elementu	Dane techniczne	Ilość
Arduino Nano	--	1
Czujnik temperatury DS18B20	Obudowa TO-92	4
Dioda LED	THT, średnica 5mm	5
Rezystor	THT, 470Ω	9
Rezystor	THT, 4.7k Ω	1
Wyświetlacz 7-segmentowy LED	BA56-12SRWA	1
Przycisk monostabilny	THT, wymiar 6 x 6 x 9.5 mm	2
Złącze goldpin [wtyk]	Raster 2.54mm	12
Złącze goldpin [gniazdo]	Raster 2.54mm	30
Zworka	--	2



Rys. 6: Projekt płytki PCB



Rys. 7: PCB przed montażem



Rys. 8: PCB po złożeniu

4.5. Diagram klas

Rys. 9: Diagram klas

Klasa Sensor:
Obiektem klasy jest czujnik temperatury. W klasie zdefiniowane są pola prywatne, które przechowują ostatnią odczytaną temperaturę z czujnika oraz jego unikalny 64-bitowy adres. Metodami klasy są funkcje, które umożliwiają odczyt i zapis wartości do prywatnych pól klasy.

Klasa DS18B20:
Autor: Mathias Munk Hansen. Github [2]
Pola i metody klasy DS18B20 umożliwiają komunikację z czujnikami temperatury DS18B20 wykorzystanymi w projekcie. Klasa DS18B20 agreguje całkowicie (ang. Composition) klasę OneWire.

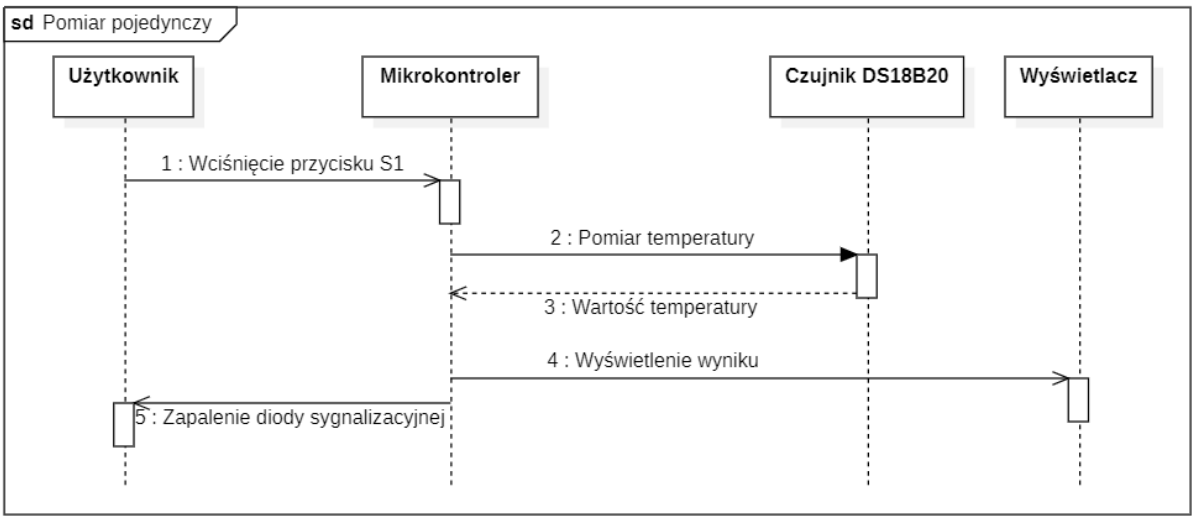
Klasa OneWire:
Autorzy: Jim Studt, Paul Stoffregen. Github [3]
Obiektem klasy jest interfejs OneWire. Służy do komunikacji mikrokontrolera z czujnikami temperatury DS18B20.

Klasa SevenSeg:
Autor: Sigvald Marholm. Github [4]
Obiektem klasy jest wyświetlacz LED 7-segmentowy. Metody i pola klasy umożliwiają konfigurację i wyświetlanie wartości na wyświetlaczu 7-segmentowym.

4.6. Diagram sekwencji

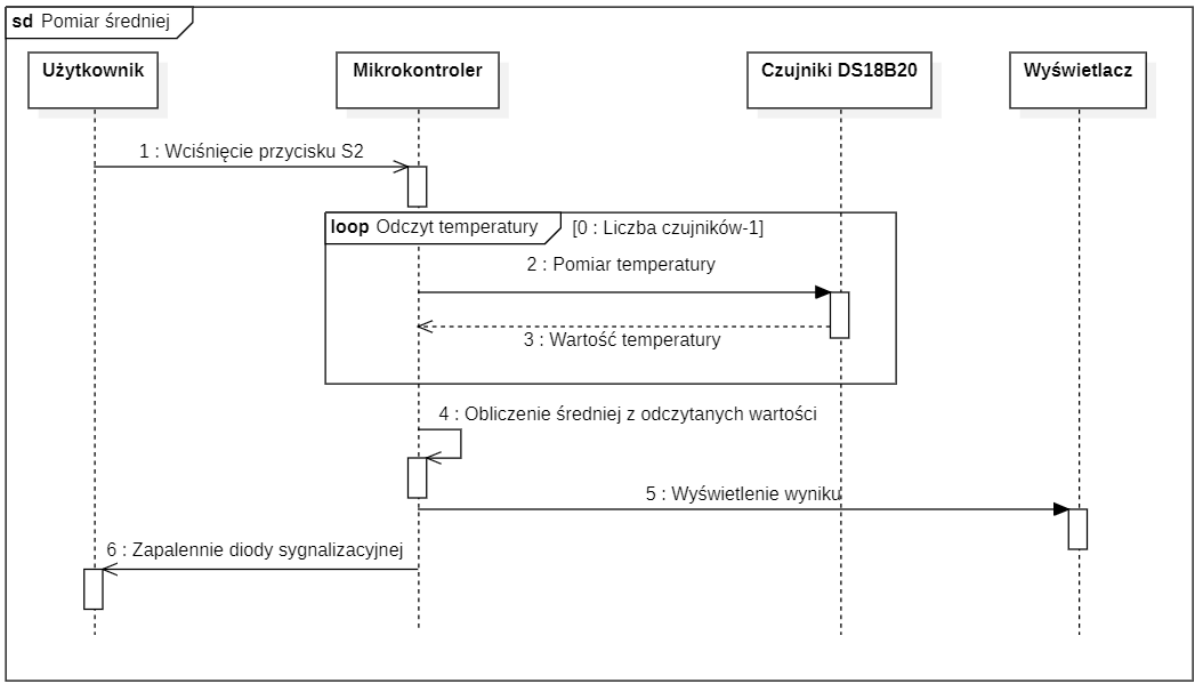
Na Rys. 10 oraz Rys. 11 przedstawiono diagramy sekwencji dla pomiarów w dwóch trybach odpowiednio pomiar z jednego czujnika oraz pomiar średniej wartości z wszystkich czujników.

Dla trybu pomiaru pojedynczego, mikrokontroler oczekuje na przerwanie wywołane przez użytkownika poprzez wciśnięcie przycisku S1 na płytce PCB. Mikrokontroler posiada licznik wciśnięć, który definiuje numer czujnika, z którego zmierzona ma być temperatura. Licznik po każdym wciśnięciu jest zwiększany o 1. Po osiągnieciu wartości równej liczbie czujników, licznik zostaje wyzerowany i liczy od nowa. Mikrokontroler po przerwaniu odczytuje wartość temperatury z odpowiedniego czujnika. Następnie po otrzymaniu informacji zwrotnej, wyświetla wynik na wyświetlaczu oraz zapala diodę sygnalizacyjną odpowiadającą numerowi czujnika. W przypadku błędu odczytu z czujnika wyświetlany jest błąd. Informacje na temat obsługi wyjątków przedstawiono w punkcie 4.7.



Rys. 10: Diagram sekwencji dla pomiaru z pojedynczego czujnika.

Dla trybu pomiaru średniej wartości ze wszystkich czujników, mikrokontroler oczekuje na przerwanie wywołane przez użytkownika poprzez wciśnięcie przycisku S2. Następnie w pętli dokonuje pomiaru temperatury dla wszystkich czujników. Posiadając wynik pomiaru ze wszystkich dostępnych czujników obliczana jest średnia, a następnie wyświetlana na wyświetlaczu. Dodatkowo dla użytkownika zapalana jest dioda sygnalizująca tryb pomiaru średniej.



Rys. 11: Diagram sekwencji dla pomiarów średniej wartości z wszystkich czujników.

4.7. Obsługa wyjątków oraz błędów

Implementacja systemu zawiera podstawową obsługę wyjątków oraz błędów, które mogą wydarzyć się podczas pracy urządzenia. Obsługa zdarzeń niepożądanych jest na bieżąco modyfikowana oraz rozwijana tuż po wykryciu tego typu zjawisk. Zdefiniowane kody błędów zestawiono w Tab. 3:

Tab. 3: Zestawienie kodów błędu wraz z nazwami.

Kod błędu	Nazwa błędu
Er1	Nie znaleziono sensora
Er2	Błędny odczyt z czujnika
--	--

Opis kodów błędu:

- *Nie znaleziono sensora* – Błąd sygnalizuje znalezienie mniej czujników niż zostało zdefiniowane podczas inicjalizacji urządzenia (Liczba znalezionych czujników powinna

wynosić 4). Informacja o błędzie wyświetlona zostaje na wyświetlaczu w postaci napisu „*Er1*”. Wykrycie błędu nie blokuje działania całego urządzenia. Możliwa jest dalsza praca urządzenia i odczytywanie temperatury z dostępnych czujników.
Obsługa zdefiniowana w pliku func.cpp w funkcji find_sensor().

- *Błędny odczyt z czujnika* – Błąd sygnalizuje błędny odczyt z czujnika. Informacja o błędzie wyświetlona zostaje na wyświetlaczu w postaci napisu „*Er2*” oraz zapalona jest dioda odpowiadająca numerowi czujnika.
  Obsługa zdefiniowana w pliku głównym w funkcji loop().

Obsługa błędów bez wyświetlania informacji użytkownikowi:

- *Błąd odczytu w trakcie pomiaru średniej* - W przypadku pojawienia się błędu odczytu z czujnika podczas pomiarów średniej wartości ze wszystkich czujników, wartość średnia obliczana jest tylko na podstawie temperatury i liczby czujników, z których poprawnie odczytano wartości.
- *Redukcja drgań styków* - Zredukowanie wpływu drgań styków wykorzystanych przycisków, które wpływa na poprawne zliczanie ilości wciśnięć. W tym celu w funkcji obsługi przerwania wstawiono opóźnienie o wartości 10 ms, po którym sprawdzany jest stan na wejściu pinu do którego podłączono przycisk. Jeśli po zastosowanym opóźnieniu stan nadal jest niski to wykonywane są operacje odpowiednie dla przypisanego przycisku. W przeciwnym razie, program wychodzi z obsługi przerwania. Przykład kodu dla przycisku S1 przedstawiono poniżej:

```
delay(10);                          // Eliminate debouncing
if(digitalRead(S1_PIN) == LOW){     // Eliminate debouncing cdn.
  ……
  }
```

Zastosowane kroki znacznie wpływają na zredukowanie błędów zliczania ilości wciśnięć podczas działania systemu, ale nie zapewniają pełnej ochrony przed możliwymi błędami.

5. Opis realizacji

Wykorzystane narzędzia do realizacji projektu oraz przeprowadzenia testów:

- **KiCad 7.0** – to oprogramowanie typu Open Source do projektowania PCB, które zapewnia kompleksowe narzędzia do projektowania układów elektronicznych. W tym projekcie, KiCad został wykorzystany do stworzenia schematu połączeń oraz dedykowanego PCB dla systemu pomiarowego. Dzięki bogatej funkcjonalności, KiCad umożliwił projektowanie schematów elektrycznych, tworzenie układów PCB oraz generowanie plików do produkcji.

- **Arduino IDE 2.0.2** - to środowisko programistyczne, które umożliwia łatwe programowanie mikrokontrolerów z rodziny Arduino. W tym projekcie, Arduino IDE zostało wykorzystane do pisania, kompilacji oraz wgrywania kodu źródłowego na mikrokontroler Arduino Nano. Dzięki prostocie i intuicyjności Arduino IDE, możliwe było szybkie tworzenie oraz testowanie kodu sterującego systemem. Środowisko umożliwia także szybki dostęp do wielu gotowych bibliotek udostępnionych przez innych autorów.

- **Logic 2.4.14** – to oprogramowanie do analizy sygnałów cyfrowych, które umożliwia precyzyjną diagnostykę oraz debugowanie układów elektronicznych. W projekcie zostało wykorzystane do sprawdzenia poprawności działania interfejsu 1-Wire.

- **Analizator logiczny Saleae** – narzędzie do analizy cyfrowych sygnałów logicznych. Posiada on 8 kanałów, co pozwala na jednoczesne monitorowanie wielu sygnałów. Wraz z oprogramowaniem Logic pozwolił na sprawdzenie poprawności działania interfejsu 1-Wire.

6. Opis wykonanych testów

6.1. Test interfejsu 1-Wire

Test polegał na sprawdzeniu poprawności transmisji danych miedzy mikrokontrolerem, a czujnikiem DS18B20. Dane przesyłane po interfejsie 1-Wire przechwycono przy pomocy analizatora stanów logicznych Saleae.

Interesujący fragment transmisji danych przedstawiono na Rys.12:

Rys. 12: Transmisja danych przy użyciu interfejsu 1-Wire.

Przedstawiony fragment porównano z diagramem sekwencji transmisji danych udostępnionym przez producenta (Rys. 13).

Tx	64-bit ROM code	Master sends DS18B20 ROM code.
Tx	BEh	Master issues Read Scratchpad command.
Rx	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.

Rys. 13: Fragment diagramu sekwencji transmisji danych. Źródło [3].

Na 64-btowy ROM CODE składa się 8 bitowy FAMILY CODE (0x28), 48 bitowy numer seryjny czujnika oraz 8 bitowa wartość sumy kontrolnej CRC. Następnym bajtem jest komenda odczytu 0xBE. Ostatnim etapem jest transmisja z czujnika 9 bajtów danych, z których pierwsze 2 bajty reprezentują 16-bitową wartość temperatury. W przypadku testowym jest to 0x7F (MSB) oraz 0x01 (LSB).
Zgodnie z dokumentacją czujnika DS18B20 obliczono temperaturę:

$$0x017F = 383_{(10)}$$
$$T = 383 / 16 = 23{,}9375\ [°C]$$

Temperatura na wyświetlaczu wyniosła: 23,9 [°C]

Wynik testu: Porównując otrzymany przebieg transmisji danych oraz diagram sekwencji udostępniony przez producenta można uznać, że transmisja danych między mikrokontrolerem i czujnikami odbywa się poprawnie.

6.2. Test eliminacji drgań styków przycisku

Test polegał na sprawdzeniu efektywności zastosowanych kroków w celu eliminacji drgań styków wykorzystanych przycisków.

Procedura pomiarowa:
- wciśnięto przycisk
- obserwowano zachowanie układu

Pomiar przeprowadzono dla dużej ilości prób i różnego czasu trwania wciśnięcia przycisku.
Możliwe zachowania układu:
- poprawne zliczenie pojedynczego wciśnięcia (pożądane)
- wielokrotne zliczenie pojedynczego wciśnięcia (niepożądane)
- brak zliczenia pojedynczego wciśnięcia (niepożądane)

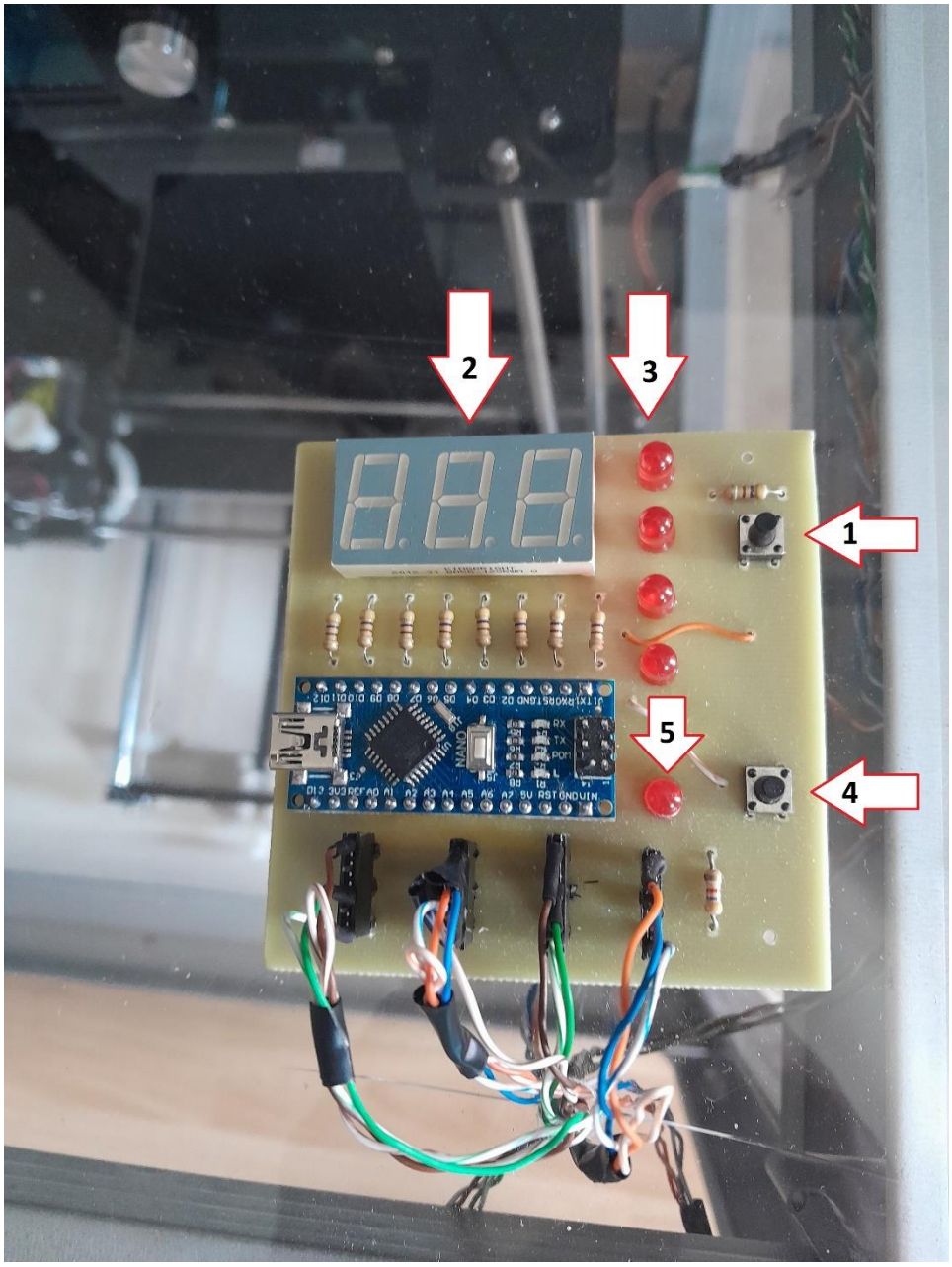
Wyniki pomiaru zestawiono w Tab. 4:

Tab. 4: Wyniki pomiaru dla testu eliminacji drgań styków:

Poprawne zliczenie	Brak zliczenia	Wielokrotne zliczenie	Ilość prób
92	1	7	100

Z wyników można zauważyć, że zastosowana metoda eliminacji drgań styków zapewnia ponad 90% procentową skuteczność. Osiągnięty wynik jest wystarczający dla tego projektu.

7. Podręcznik użytkownika



Rys. 14: Zdjęcie podglądowe stworzonego układu.

7.1. Tryb pomiaru pojedynczego

W trybie pomiaru pojedynczego odczytywana jest wartość temperatury kolejno z każdego czujnika. Pomiar uruchamiany jest poprzez wciśnięcie przycisku S1 (1). Po poprawnym odczycie wyświetlona zostaje temperatura na wyświetlaczu (2) oraz zapalana jest dioda (3) sygnalizująca, z którego czujnika odczytano temperaturę.

Przykłady:

- W celu odczytu temperatury z czujnika nr. 3 należy wcisnąć trzykrotnie przycisk S1.
- Jeśli ostatnią odczytaną wartością jest temperatura z czujnika nr. 2 to aby odczytać wartość z czujnika nr. 1 należy wcisnąć trzykrotnie przycisk S1.

7.2.Tryb pomiaru wartości średniej

W celu odczytania średniej wartości temperatury ze wszystkich czujników należy przycisnąć przycisk S2 (4). Następnie na wyświetlaczu (2) wyświetlona zostanie obliczona średnia z pomiarów oraz zapalona zostanie dioda (5) sygnalizująca prace w trybie pomiaru wartości średniej. Każdorazowe wciśnięcie przycisku S2 (4) wykonuje kolejny pomiar.

7.3. Rozwiązywanie problemów

W tabeli Tab. 5 zestawiono możliwe kody błędów mogące pojawić się na wyświetlaczu podczas pracy systemu. Pod tabelą przedstawiono możliwe przyczyny i rozwiązania błędu.

Tab. 5: Zestawienie kodów błędu wraz z nazwami.

Kod błędu	**Nazwa błędu**
Er1	Nie znaleziono sensora
Er2	Błędny odczyt z czujnika
--	--

- **Kod błędu: Er1, Er2**

Możliwe przyczyny:
- Brak podłączonego czujnika do płytki PCB
- Brak styku w złączu
- Przerwany przewód łączący czujnik z płytką PCB
- Uszkodzony czujnik

Możliwe rozwiązania:
- Reset mikrokontrolera
- Poprawa styku złącza
- Podłączenie czujnika do płytki
- Wymiana czujnika

UWAGA! Po każdej modyfikacji należy zresetować mikrokontroler w celu przeprowadzenia ponownej inicjalizacji czujników przez system.

Bibliografia

[1] NETTIGO, https://nettigo.pl/products/klon-arduino-nano-v3-atmega328p-ch340
[Dostęp online: 15.05.2024r]
[2] Ntronic, „ZASADA DZIAŁANIA, PROJEKTOWANIE ORAZ BUDOWANIE NIEZAWODNYCH SIECI 1 WIRE”, https://ntronic.pl/jak-projektowac-niezawodna-siec-1wire/
[Dostęp online: 15.05.2024r]
[3] Maxim Integrated, DS18B20 Programmable Resolution 1-Wire Digital Thermometer, https://www.analog.com/media/en/technical-documentation/data-sheets/DS18B20.pdf
[Dostęp online: 15.05.2024r]
[4] Kingbright, BA56-12SRWA, https://www.kingbrightusa.com/images/catalog/SPEC/BA56-12SRWA.pdf [Dostęp online: 15.05.2024r]
[5] DS18B20, https://github.com/matmunk/DS18B20 [Dostęp online: 15.05.2024r]
[6] OneWire, https://github.com/PaulStoffregen/OneWire [Dostęp online: 15.05.2024r]
[7] SevenSeg, https://github.com/sigvaldm/SevenSeg [Dostęp online: 15.05.2024r]