

Ten projekt stworzyłem jako połączenie elektroniki, programowania i wizualizacji danych - dokładnie taki zestaw, który pozwala poczuć, że nawet prosty układ z Arduino może działać jak miniaturowy system SCADA. Zbudowałem od podstaw działający tor komunikacyjny: od czujnika ultradźwiękowego mierzącego odległość, przez diody i buzzer reagujące na tę odległość w czasie rzeczywistym, po przesyłanie danych do Node-RED, zapis w InfluxDB i wizualizację całego procesu w Grafanie. Wszystko działa na żywo, bez opóźnień, a efekt końcowy naprawdę daje satysfakcję.

Zacząłem od części sprzętowej. Na klasycznym Arduino Uno R3 zamontowałem czujnik HC-SR04, sześć diod LED (po dwie zielone, pomarańczowe i czerwone) oraz buzzer. Wszystko połączyłem na płytce stykowej, stosując rezystory do LED-ów i wspólną masę. Stworzyłem taki układ, w którym sensor mierzy odległość od obiektu (w moim przypadku był to nawet model bolidu F1 ustawiany w różnych miejscach), a Arduino w zależności od wyniku decyduje, które diody mają się świecić i czy buzzer ma być aktywny. Dla dużych odległości świecą się tylko zielone diody, w strefie pośredniej zapalają się pomarańczowe i zmienia się dźwięk buzzera, a w strefie krytycznej czerwone LED-y migają razem z ostrzejszym sygnałem akustycznym.

Następnie napisałem kod na Arduino. Obsługuje on czujnik, oblicza odległość na podstawie czasu powrotu impulsu, steruje LED-ami i buzzerem, a na końcu tworzy ramkę JSON zawierającą wszystkie informacje: zmierzoną odległość, aktualny stan każdej diody i stan buzzera. Tę ramkę wysyłam co 100 ms przez port szeregowy, dzięki czemu całość działa płynnie i nadaje się do dalszego przetwarzania.

Kolejnym etapem było wykorzystanie Node-RED. Podłączyłem węzeł Serial In skonfigurowany pod mój port COM i prędkość 9600, a następnie przepuściłem dane przez węzeł JSON, aby zamienić tekst na obiekt JavaScript. Dodałem węzeł debug, żeby na żywo obserwować przychodzące dane i potwierdzić, że Arduino generuje poprawne wartości. Dalej przygotowałem dwa oddzielne przepływy: pierwszy formatuje dane pod MQTT, a drugi pod InfluxDB. Dane z pierwszego toru trafiają do lokalnego brokera Mosquitto, dzięki czemu mogę je przeglądać np. w MQTT Explorerze. Drugi tor wykorzystuje węzeł InfluxDB Out i wysyła pomiary do bazy InfluxDB 2.0 - dokładnie do bucketa „arduino” w organizacji „moje-lab”.

Skonfigurowałem InfluxDB tak, aby Node-RED miał odpowiednie uprawnienia, a następnie dodałem InfluxDB jako źródło danych w Grafanie. Dzięki temu mogłem stworzyć własny dashboard, który pokazuje w czasie rzeczywistym wszystkie kluczowe informacje z mojego układu. Przygotowałem panel z bieżącą odległością, wskaźnik stanu buzzera, kafelki informujące o stanie każdej diody.. Całość wygląda bardzo przejrzysto i od razu widać, jak układ reaguje na przesuwanie obiektu przed czujnikiem.

Pare fotek z konfiguracji:

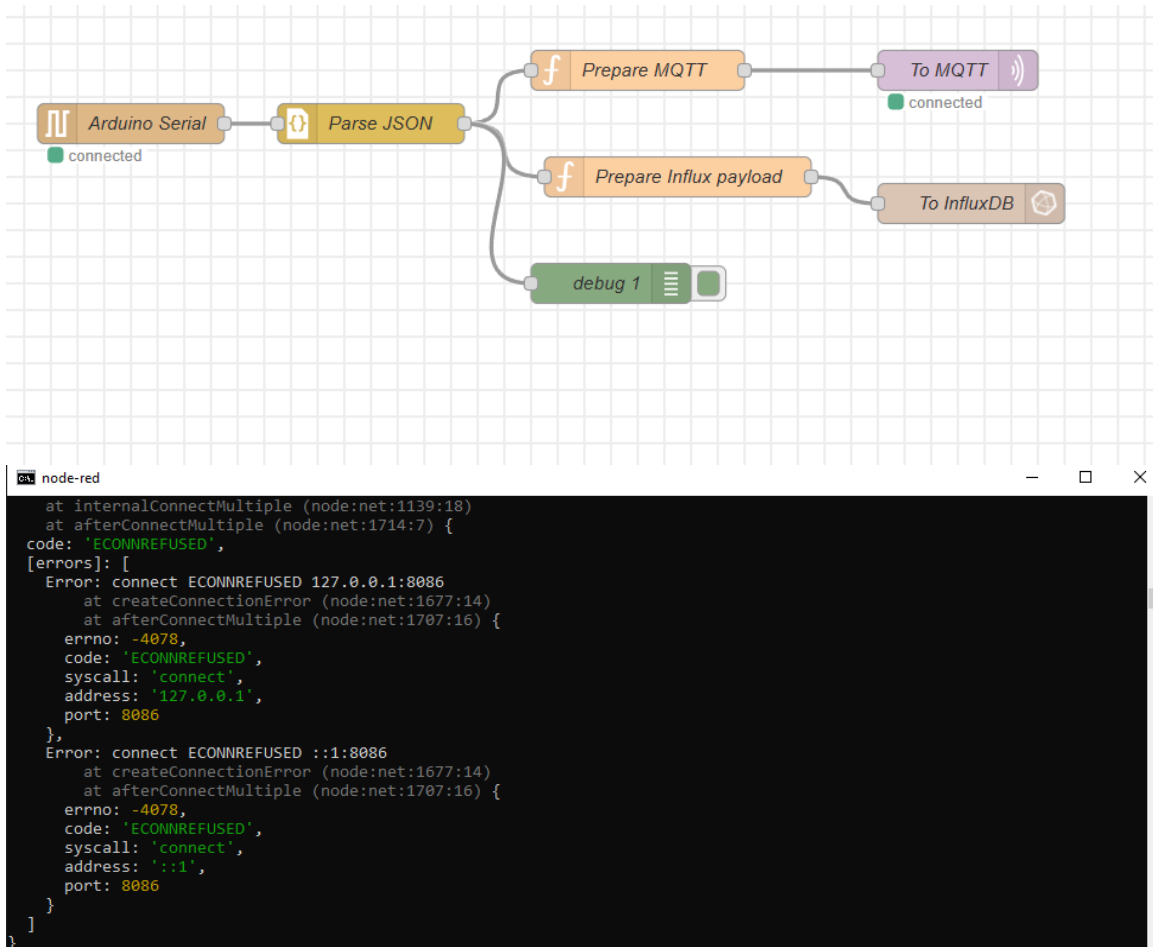
**Edit serial in node**

Delete Cancel Done

**Properties**

**Name** Arduino Serial

**Serial Port** COM3:9600-8N1



Stworzyłem również gałąź do komunikacji z MQTT, które działa u mnie przez mosquitto:

```
Wiersz polecenia - mosquitto -c "C:\Users\PC\Desktop\inzynierka\mosquitto\mosquitto.conf" -v
... (116 bytes))
1764424116: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424116: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424116: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424116: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (115 bytes))
1764424116: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (115 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424117: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424118: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424118: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
1764424118: Received PUBLISH from nodered_707ec86c9cb1e10b (d0, q0, r0, m0, 'arduino/status',
... (116 bytes))
```

⚙️ Properties

⚙️

📄

🔍

🔑 Name

To InfluxDB

📄 Server

[v2.0] InfluxDB2

▼

✎

+

👤 Organization

moje-lab

📄 Bucket

arduino

📡 Measurement

arduino\_status

⌚ Time Precision

▼

Tutaj dodałem token i resztę potrzebnych danych:

⚙️ Properties

⚙️

📄

🔑 Name

InfluxDB2

📄 Version

2.0

▼

📄 URL

http://localhost:8086

🔒 Token

.....

⌚ Connection timeout (seconds)

5000

☐ Verify server certificate