Podstawy techniki mikroprocesorowej 2

Lab 5 - I2C (TWI) – Pamięć EEPROM AT24C32

Imię i Nazwisko:	Remigiusz Mielcarz, Grzegorz Salzburg
Nr indeksu:	252887, 252912

Termin zajęć: dzień tygodnia, godzina:	Środa 14:10-17:10 TP
Numer grupy ćwiczeniowej:	Y03-45f
Data wykonania ćwiczenia:	19.01.2022
Termin do oddania sprawozdania:	02.02.2022
Prowadzący kurs:	Dr inż. Krzysztof Halawa

Spis treści

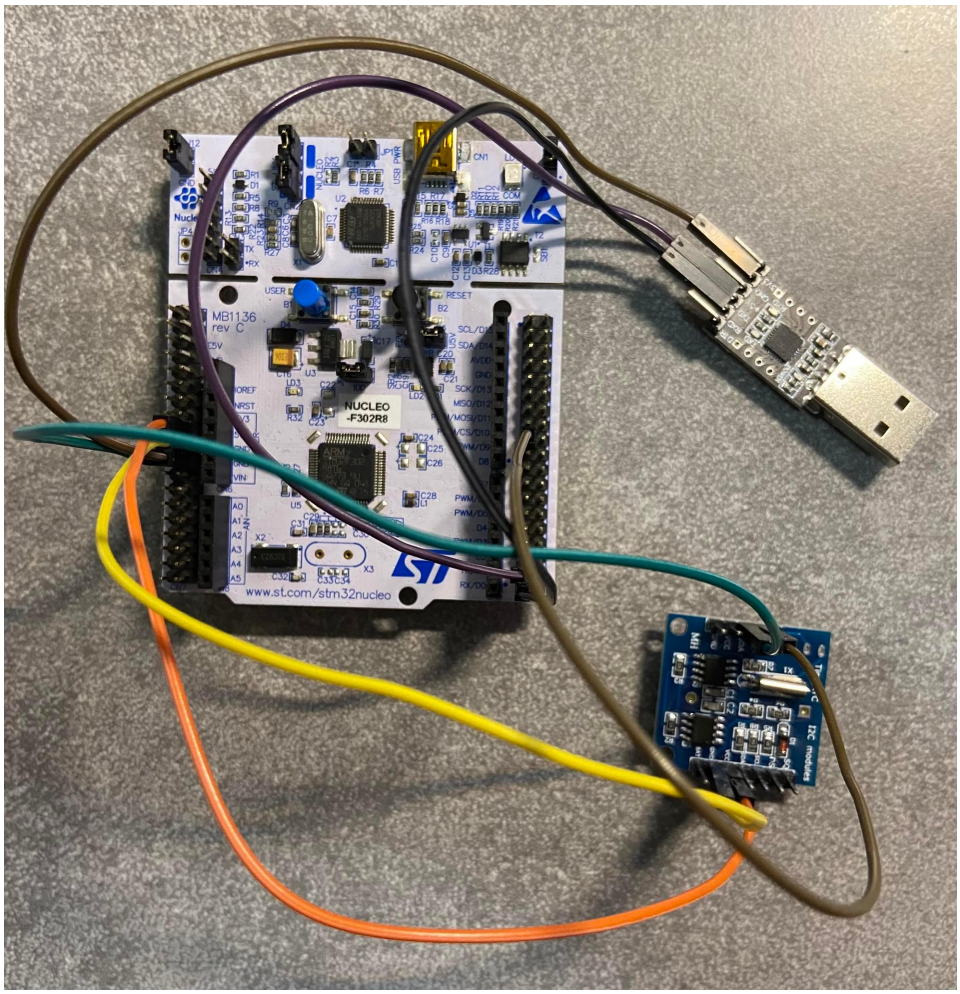
1 Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z magistralą I2C (TWI ang. Two Wire Interface) do komunikacji z modułem zawierającym EEPROM AT24C32 oraz zegarem czasu rzeczywistego DS1307.

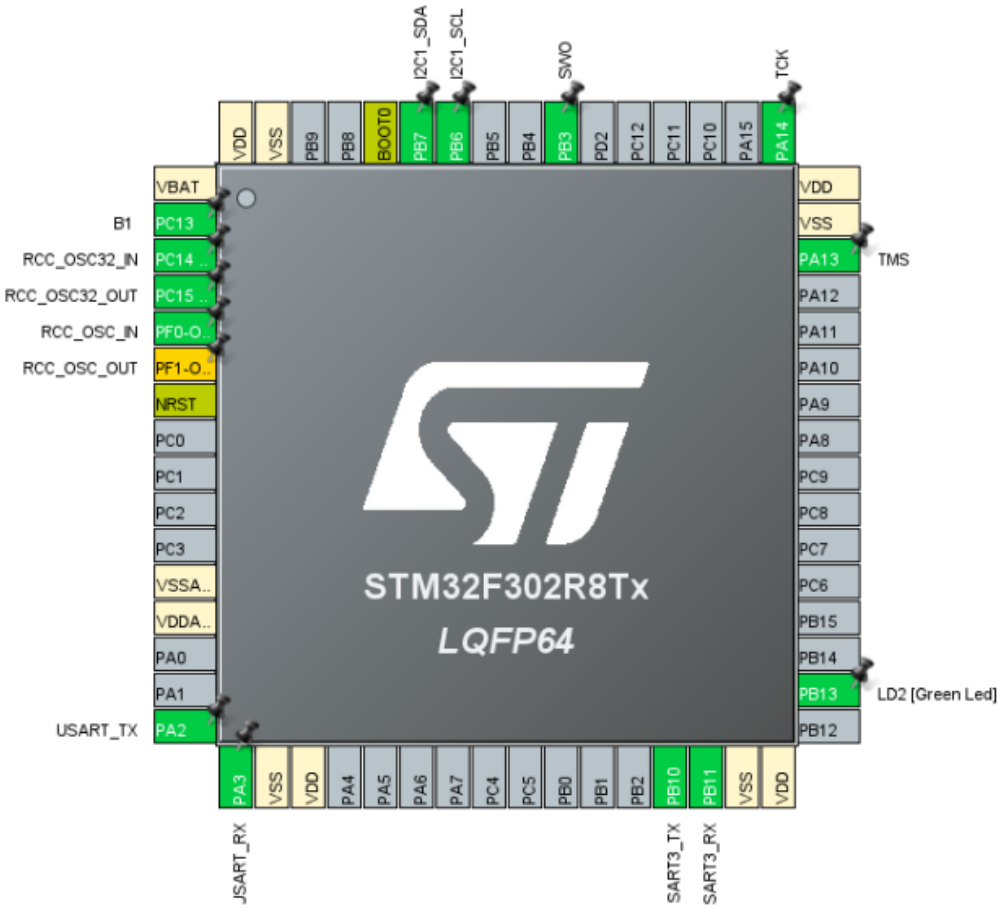
2 Zadania do wykonania

- Znalezienie wyprowadzeń SDA i SCL mikrokontrolera. Następnie podłączenie nucleo z modułami
- Zapisanie kolejnych cyfr indeksów osób z grupy w pamięci EEPROM od adresu 1000. Następnie należy je odczytać i wysłać przez UART do komputera (Zadanie numer 1).
- Napisanie programu, który zliczał naciśnięcia niebieskiego przycisku na płytce nucleo. Zliczona liczba powinna być zapisywana w EEPROM. Liczba naciśnięć powinna być wysyłana przez UART (Zadanie numer 2)

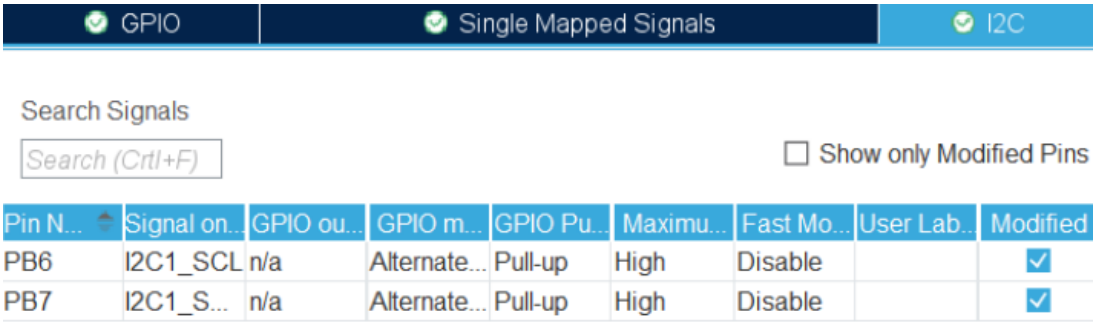
3 Schemat podłączenia



4 Schemat i konfiguracja pinów mikrokontrolera



Rysunek 1: Konfiguracja pinów mikrokontrolera w środowisku CubeIDE.



Rysunek 2: Szczegóły konfiguracji

5 Kod programu

5.1 Kod do zadania numer 1

```
uint8_t Device_Address = 0xA0;  // Adres uk adu
uint16_t Memory_Address = 1000; // Adres pami ci
HAL_Delay(100);

uint32_t indexes[2] = {252912, 252887}; // Indeksy

for (int i=0; i<2; i++)
{
    HAL_I2C_Mem_Write(&hi2c1, Device_Address, Memory_Address+i*6, 2, (uint8_t*)&indexes[i],
    sizeof(indexes[i]),100); // Zapis

    HAL_Delay(100);
}

HAL_Delay(100);
uint32_t results[2] = {0 ,0};

for(int j = 0; j<2 ;j ++)
{
    HAL_I2C_Mem_Read(&hi2c1, Device_Address ,Memory_Address+j*6, 2, (uint8_t*)&results[j],
    sizeof(results[j]), 100); // Odczyt

    HAL_Delay(100);
}

for(int k = 0; k<2; k++)
{
    printf("%ld\r\n", results[k]); // Wypisanie rezultatu
}

int transmit_data(int c)
{
    HAL_UART_Transmit(&huart2, (uint8_t*)&c, 1, HAL_MAX_DELAY); // Transmitacja danych
    return 1;
}
```

5.1.1 Omówienie programu i screen konsoli

Rysunek 3: Konsola

- W pamięci EEPROM od adresu 1000 zapisano kolejne cyfry numerów naszych indeksów. z grupy. Następnie je odczytano i wysłano przez UART do komputera
- Zapis do pamięci wykonywano za pomocą funkcji HAL_I2C_Mem_Write. Funkcja ta zawiera argumenty takie jak: struktura opisująca interfejs i2c, adres układu, adres w pamięci, bufor na dane które chcemy wysłać, wielkość bufora (liczba bajtów), maksymalny czas transmisji, długość adresu.
- Odczyt z pamięci wykonywano za pomocą funkcji HAL_I2C_Mem_Read. Argumenty funkcji są analogiczne do funkcji powyżej.
- Funkcja transmit_data odpowiada za przesył danych (numerów indeksów) przez UART
- W konsoli możemy potwierdzić, że prawidłowo wyświetlają się oba indeksy

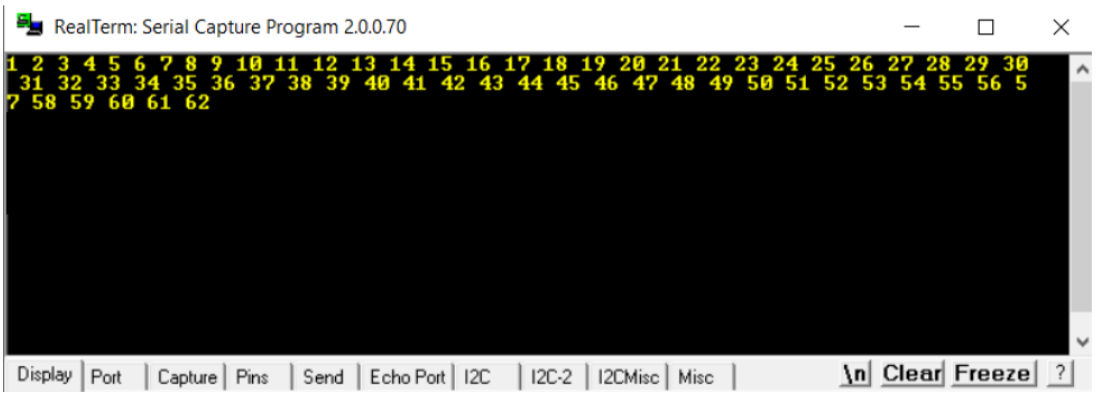
.

5.2 Kod do zadania numer 2

```
1  uint16_t MemoryAddress = 1;   // Adres pami ci
2  uint8_t DeviceAddress = AxA0; // Adres uk adu
3
4  uint8_t reply = 0;
5  uint8_t count_number_cts = 0;
6
7  HAL_I2C_Mem_Read(&hi2c1, DeviceAddress, MemoryAddress, 2, (uint8_t*)&count_number,
8  sizeof(count_number), 100); // Timeout 100
9
10 count_number_cts = count_number;
11
12 while (1)
13 {
14     if(count_number_cts != count_number)
```

```
15      {
16          HAL_Delay (100);
17
18          HAL_I2C_Mem_Write(&hi2c1, DeviceAddress, MemoryAddress, 2, &count_number,
19          sizeof(count_number), 100);
20
21          HAL_Delay (100);
22
23          HAL_I2C_Mem_Read(&hi2c1, DeviceAddress, MemoryAddress, 2, (uint8_t*)&reply,
24          sizeof(reply), 100);
25
26          printf("%d ", reply);
27          count_number_cts = count_number;
28      }
29  }
30
31  void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) // Przerwanie na niebieski przycisk
32  {
33      if(GPIO_Pin == B1_Pin)
34      {
35          count_number++;
36      }
37  }
```

5.2.1 Omówienie programu i screen konsoli



Rysunek 4: Konsola

- W tym programie używamy tych samych funkcji HAL_I2C_Mem_Write i HAL_I2C_Mem_Read. Ich argumenty są omówione w omówieniu zadania 1.
- Funkcja HAL_GPIO_EXTI_Callback odpowiada za obsługę przerwań dla naszego niebieskiego przycisku.
- W konsoli możemy zaobserwować ilość naciśnięć naszego przycisku. Możemy tą wartość także sprawdzić w debuggerze.