

Dokumentacja projektu na przedmiot “Systemy Wbudowane”

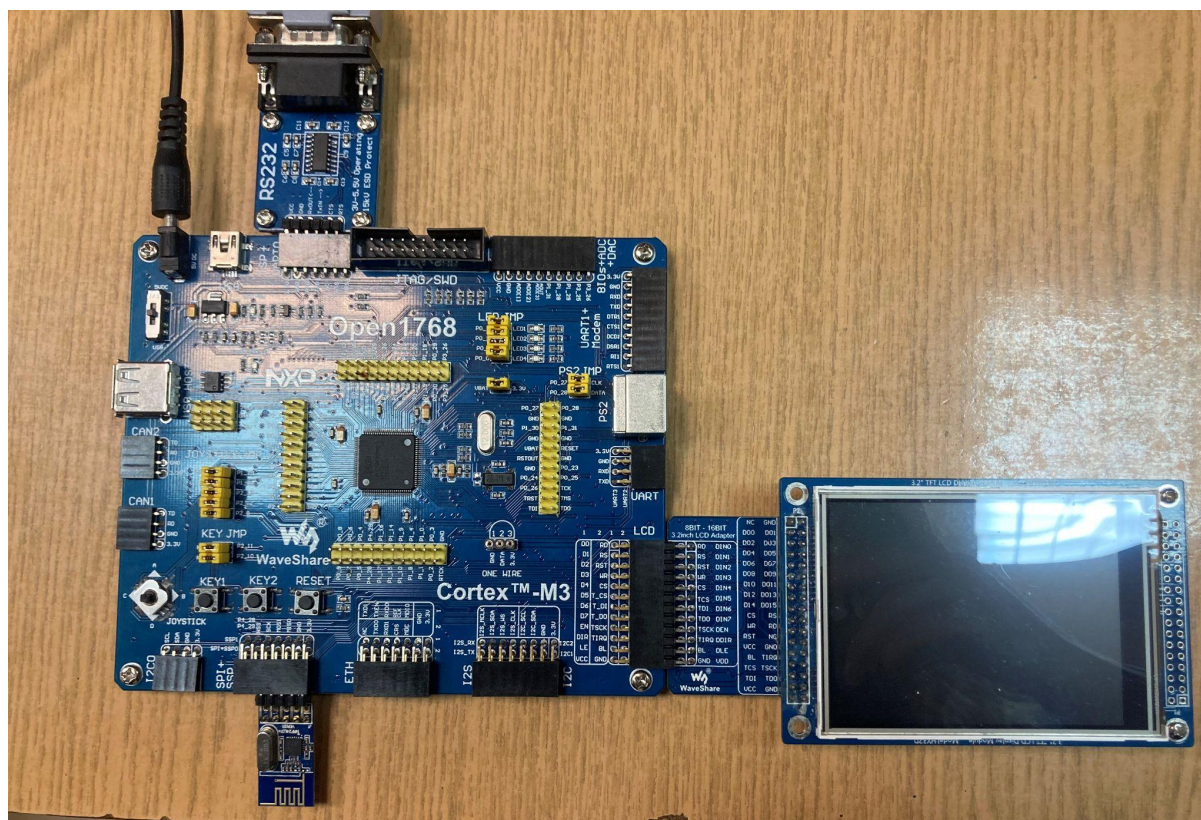
Mateusz Lewandowski, Karolina Klimek

1. Opis projektu

Celem projektu było stworzenie oprogramowania na mikrokontroler LPC 1786 pozwalającego na obustronną komunikację krótkimi wiadomościami poprzez moduł radiowy NRF24I01. Do realizacji użyto również wyświetlacza LCD oraz modułu UART. Projekt zakłada, że użyto co najmniej dwóch mikrokontrolerów z wgranym oprogramowaniem oraz określonymi wyżej rozszerzeniami.

1.1 Instrukcja uruchomienia

A) Podłączanie modułów



Rys.1 Zdjęcie płytki z wszystkimi podłączonymi modułami.

Rys. 1 prezentuje:

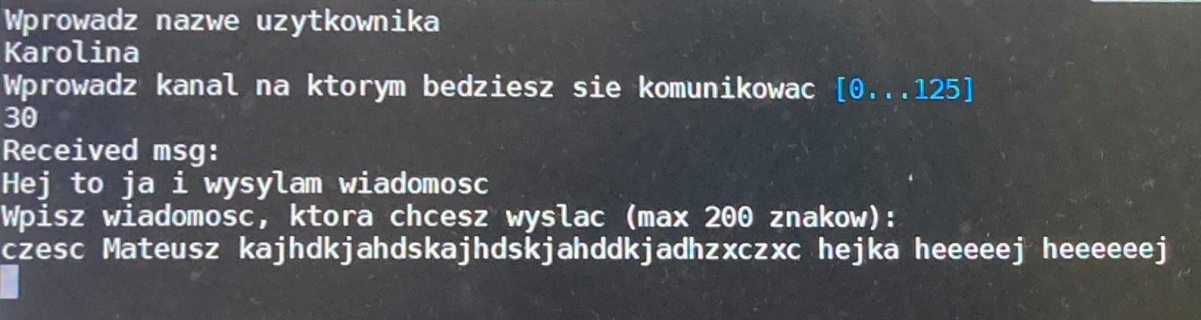
- moduł UART RS232 podłączony do UART0
- moduł NRF24I01 podłączony do SSP0 po prawej stronie portu SPI/SSP0
- wyświetlacz LCD podłączony do wejścia wyświetlacza

Do skorzystania z funkcjonalności konieczne jest podłączenie modułów rozszerzających tak jak na Rys. 1.

W celu osiągnięcia komunikacji wymagane jest podłączenie modułu UART do komputera i otworzenie aplikacji (np. MobaXterm), która pozwoli na obserwowanie zmian na użytym do połączenia porcie. Otwarcie kanału z połączeniem wymaga również podania odpowiedniej prędkości transmisji, która musi być równa dokładnie 115200 bps.

B) Rozpoczęcie komunikacji

Uruchomienie mikrokontrolera skutkuje pojawieniem się informacji tekstowej w karcie, która obsługuje port: "Podaj nazwę użytkownika". Taka sama informacja pojawi się wtedy na ekranie LCD. Od użytkownika oczekuje się wprowadzenia bezpośrednio przez klawiaturę komputera wybranej przez niego nazwy użytkownika, która w trakcie wpisywania będzie wyświetlana również na ekranie LCD. Wciśnięcie klawisza enter spowoduje przyjęcie nazwy użytkownika i przejście do następnego kroku w programie. Na ekranie komputera wyświetli się wtedy prośba o wpisanie numeru kanału komunikacji. Do skutecznej komunikacji między dwoma mikrokontrolerami niezbędne jest ustawienie takiego samego portu komunikacji. Po zatwierdzeniu kanału, oczom użytkownika ukazuje się menu główne z wyborem wysłania wiadomości. Wybranie opcji pisania wiadomości jest możliwe poprzez naciśnięcie joysticka na mikrokontrolerze, wtedy wyświetlacz LCD ukaże menu nowej wiadomości, a aplikacja obserwująca port komputera wyświetli informację o oczekiwaniu na wpisanie wiadomości. Wpisywanie wiadomości odbywa się poprzez klawiaturę komputera, a naciśnięcie enter spowoduje pobranie wiadomości do bufora. Użytkownik ma wtedy możliwość zdecydowania o wysłaniu wiadomości lub anulowaniu, które to opcje będą widoczne na wyświetlaczu LCD. Manewrowanie joystickiem w górę i w dół pozwoli na wybór jednej z opcji. Wybrana opcja to ta, przy której trójkąt jest żółty. Wybranie opcji "WYSLIJ" spowoduje radiowe nadanie wiadomości, która może być odebrana przez inny mikrokontroler, o ile jest włączony i ma wybrane takie same ustawienia kanału. Opcję tę można kliknąć wiele razy i za każdym wysłana będzie ta sama wiadomość. Wybranie "WYJDZ" przeniesie użytkownika z menu wysyłania wiadomości z powrotem do menu głównego, gdzie użytkownik może wybrać opcje napisania wiadomości, jeżeli chce wysłać nową.



```
Wprowadz nazwe uzytkownika
Karolina
Wprowadz kanal na ktorym bedziesz sie komunikowac [0...125]
30
Received msg:
Hej to ja i wysylam wiadomosc
Wpisz wiadomosc, ktora chcesz wyslac (max 200 znakow):
czesc Mateusz kajhdkjahdskajhdsjkjahddkjadhzcxczc hejka heeeeej heeeeeej
```

Rys.2 Zrzut ekranu z aplikacji MobaXterm prezentujący komunikację z mikrokontrolerem, odebraną wiadomość i wpisywanie nowej wiadomości do wysłania

Jak widać na Rys.2 wiadomość zostanie automatycznie odebrana przez kontroler i wyświetlona dla użytkownika. Należy zaznaczyć, że moduł NRF jest w stanie nadać jednorazowo dane o wielkości do 32 bitów, a więc niezależnie od długości wpisanej wiadomości nadane zostanie maksymalnie 31 znaków plus znak końca wiadomości (w przeciwieństwie do tego, co sugeruje Rys. 2).



Rys.3 Zdjęcie ekranu LCD wyświetlającego menu wysyłania wiadomości, po jej wysłaniu.

2. Dokumentacja techniczna

2.1. main.c

2.1.1. Zmienne globalne

char user[30] - bufor na nazwę użytkownika

int channel - numer kanału radiowego z przedziału [0; 125]

msgPieceBufReceive[_Buffer_Size] - bufor na wiadomość odbieraną

msgPieceBufSend[_Buffer_Size] - bufor na wiadomość wysłaną

char Address[_Address_Width] - tablica adresów do modelu NRF24I01

2.1.2. int main(void)

W funkcji main są wywoływane funkcje inicjalizujące oraz konfiguruje potrzebne peryferia takie jak:

- UartConfigure()
- SSPInit()
- Delay_Init()
- LcdConfiguration()
- init_ILI9325()
- initGUI()

Funkcja następnie czyści ekran wyświetlacza LCD i prosi użytkownika o podanie nazwy użytkownika oraz kanału radiowego na którym chce się komunikować poprzez terminal komputera do którego mikrokontroler jest podłączony przez UART.

Po uzyskaniu kanału, main inicjalizuje moduł radiowy ustawiając go na kanał wybrany przez użytkownika oraz wyświetla na wyświetlaczu LCD menu główne aplikacji.

Następnie funkcja poprzez pętlę while nasłuchuje ruchów joystickiem i sprawdza czy moduł radiowy odebrał jakąś wiadomość. W przypadku kiedy zostanie odebrana jakaś wiadomość, program wyświetla ją w terminalu komputera.

2.2. MyNrfFunctions.c

Plik zawiera funkcje odpowiedzialne za komunikację z modułem radiowym takie jak wysłanie wiadomości i odebranie wiadomości.

2.2.1 void NRF24L01_Receive(char Buf[_Buffer_Size])

Funkcja pobierająca otrzymaną przez moduł NRF wiadomość do podanego jako argument bufora Buf. Sprawdza ona status modułu i w przypadku pojawienia się wiadomości w kolejce Rx FIFO pobiera ową wiadomość.

2.2.2 void NRF24L01_Send(char Buf[_Buffer_Size])

Funkcja wysyłająca podaną wiadomość w argumencie Buf. Zmienia ona na początku tryb działania modułu radiowego z Rx na Tx, co pozwala przeprowadzić transmisję. Następnie wpisuje bufor z wiadomością do bufora Tx modułu i wysyła ją. Tuż przed wyjściem z funkcji tryb pracy modułu zmienia się raz jeszcze z Tx na Rx co pozwala modułowi na dalsze nasłuchiwanie wiadomości przychodzących.

2.2.3 bool msgToReceivePresent()

Funkcja sprawdzająca czy w kolejce Rx modułu radiowego nie znajduje się wiadomość. Jeżeli się znajduje funkcja zwraca true, w przeciwnym wypadku false.

2.2.4 void getMsgAndPrint()

Funkcja pobierająca wiadomość z kolejki Rx modułu radiowego a następnie wyświetlająca ją w terminalu komputera z poprzedzającym podpisem "Received msg:". Po przesłaniu wiadomości do komputera poprzez port UART funkcja czyści kolejkę Rx poprzez wywołanie funkcji NRF24L01_Flush_RX(). Zapobiega to ponownemu pobraniu tej samej wiadomości.

2.3. MyGuiFunctions.c

Plik, który zawiera funkcje kształtujące wygląd menu na wyświetlaczu LCD, korzystając z funkcji zawartych w MyLcdFunctions.h.

2.3.1 void drawGUI(int opt)

Funkcja kształtująca wygląd menu głównego z opcją wysyłania wiadomości. Przyjmuje jeden argument typu integer ze względu na fakt, że w MyJoystickFunctions.c jest zdefiniowany pointer na funkcje rysujące GUI, który zmienia funkcje na które wskazuje i konieczne było zachowanie zgodności typów z funkcją drawWriteMessageGUI(int opt).

2.3.2 void drawWriteMessageGUI(int opt)

Funkcja kształtująca wygląd GUI wysyłania wiadomości. Zawiera opcje "WYSLIJ" i "ANULUJ". W zależności od przekazanego argumentu jeden z wyświetalnych trójkątów jest koloru żółtego (wskazuje na wybraną opcję), a drugi cyjanowego.

2.3.3 void printMessageAccepted()

Funkcja wypisująca na ekran "Wiadomosc zostala przyjeta, kliknij wyslij" .

2.3.4 void printMessageSent()

Funkcja wypisująca na ekran "Wyslano wiadomosc" .

2.4. MyLcdFunctions.c

Plik zawierający funkcje odpowiedzialne za obsługę wyświetlacza LCD mikrokontrolera, takie jak rysowanie kształtów, wyświetlanie wiadomości oraz czyszczenie wyświetlacza. W tym pliku używane są biblioteki dostarczone wcześniej na ćwiczeniach laboratoryjnych, takie jak:

- `asciiLib.h` - zawiera przepisy na wyświetlanie poszczególnych znaków
- `LCD_ILI9325.h` - zawiera funkcję inicjalizującą wyświetlacz LCD
- `Open1768_LCD.h` - zawiera funkcje potrzebne do komunikacji z wyświetlaczem LCD

2.4.1 void BresenhamLine(const int x1, const int y1, const int x2, const int y2, const int color)

Funkcja rysuje na wyświetlaczu LCD linię zaczynającą się w punkcie (x1, y1) i kończącą się w punkcie (x2, y2) w wybranym kolorze (żółty lub biały). Rozmiary wyświetlacza określone są w pliku `LCD_ILI9325.h` i wynoszą (240px, 320px). Gdy przekazany argument `color` jest jedynką to funkcja rysuje w kolorze żółtym, w przeciwnym wypadku w cyjanowym.

2.4.2 void printLetter(char let, int row, int col)

Funkcja rysuje zadaną literę `let` na ekranie LCD w danym rzędzie `row` i kolumnie `col`. Numeracja rzędów i kolumn zaczyna się od 1. Każdy rząd ma 15px wysokości, a każda kolumna 8px szerokości.

2.4.3 void printLine(char* line, int row, int col)

Funkcja wyświetlająca linię znaków bazując na funkcji `printLetter()`. Wyświetla linię w rzędzie `row` zaczynając od podanej kolumny `col`.

2.4.4 void printText(char* line, int row, int col)

Funkcja wyświetlająca długi tekst odpowiednio zawijając go na wyświetlaczu. Wyświetla tekst zaczynając od rzędu `row` i kolumny `col`. Dzieli wiadomość na mniejsze i posługując się funkcją `printLine()`, wyświetla je na ekran. Funkcja dzieli wiadomości w taki sposób aby to samo słowo zaczynało się w jednym rzędzie a kończyło w następnym.

2.4.5 void clearLetterAtPosition(int row, int col)

Czyści pojedynczą literę na pozycji (row, col).

2.4.6 void clearFragment(int rowStart, int colStart, int rowEnd, int colEnd)

Czyści prostokątny fragment ekranu o przeciwległych wierzchołkach (rowStart, colStart) i (rowEnd, colEnd) posługując się przy tym funkcją `clearLetterAtPosition()`.

2.4.6 void clearScreen()

Czyści cały ekran.

2.4.7 void printTriangle(int x, int y, int color)

Wyświetla trójkąt, którego górny wierzchołek ma współrzędne (x, y). Gdy argument `color` jest równy jeden kolor rysunku jest żółty, gdy jest inny rysunek jest cyjanowy.

2.4.8 void drawBorder()

Wyświetla ramkę o predefiniowanych parametrach (po 10px od każdej krawędzi).

2.5. MyUsartFunctions.c

Plik zawierający funkcje odpowiedzialne za obsługę portu UART. Zawiera funkcje przeznaczone do odbierania i wysyłania danych oraz do konfiguracji modułu UART po stronie mikrokontrolera.

2.5.1 void sendUsartString(char string[])

Wysyła podaną jako argument wiadomość poprzez port UART do komputera z którym mikrokontroler jest podłączony.

2.5.2 void sendUsartStringNewLine(char string[])

Wysyła podaną jako argument wiadomość poprzez port UART do komputera z którym mikrokontroler jest podłączony. Dodatkowo ustawia kursor w terminalu na nową linię.

2.5.3 char getChar()

Pobiera i zwraca znak z wejścia UART.

2.5.4 void getResponseFromUart(char* buffer, int maxLen)

Pobiera odpowiedź od użytkownika z terminalu komputera o maksymalnej długości maxLen i zapisuje ją do podanego bufora buffer. W momencie kiedy użytkownik wpisuje wiadomości te same znaki zostają wysłane z powrotem, tak aby użytkownik na bieżąco widział co pisze. Jednocześnie wpisywana wiadomość wyświetlana jest na ekranie LCD mikrokontrolera z zawijaniem tekstu. Użytkownik może przejść do następnej linii na terminalu komputera naciskając symbol ('). Ma też możliwość usuwania wcześniej napisanych znaków za pomocą backspace. Po naciśnięciu backspace wprowadzony wcześniej znak zostanie usunięty z bufora wiadomości do zwrócenia jak i zostanie wyczyszczony z ekranu LCD. Użytkownik zatwierdza wiadomość naciskając enter.

2.5.5 void UartConfigure()

Konfiguruje port UART do dalszej komunikacji z urządzeniem zewnętrznym połączonym z mikrokontrolerem.

2.5.6 void receiveMessageFromUart(char* sendMsgBuff, int size)

Wyświetla wiadomość w terminalu komputera proszącą o wprowadzenie wiadomości do wysłania przez radio. Następnie posługując się funkcją getResponseFromUart() pobiera wiadomości o maksymalnej długości size.

2.5.7 void getUser(char* uzytkownik, int size)

Wyświetla wiadomość w terminalu komputera proszącą użytkownika o wprowadzenie jego nazwy. Następnie pobiera odpowiedź do bufora uzytkownik za pomocą funkcji getResponseFromUart().

2.5.8 int getChannel()

Wyświetla wiadomość w terminalu komputera proszącą użytkownika o wprowadzenie kanału radiowego na którym będzie się komunikować. Następnie pobiera odpowiedź za pomocą funkcji getResponseFromUart() i dokonuje konwersji na int. Jeżeli po konwersji

wartość kanału nie mieści się w granicach [0, 125] generuje wiadomość proszącą o wpisanie kanału radiowego raz jeszcze.

2.5.9 void getUserAndChannel(char* user, int* channel, int userSize)

Wyświetla na ekranie LCD wiadomość o podążaniu za instrukcjami wyświetlanymi w terminalu portu na urządzeniu zewnętrznym. Następnie wywołuje funkcje `getUser(user, userSize)` oraz `getChannel()`. Następnie przypisuje otrzymane wartości do `user` oraz `channel` i czyści ekran LCD.

2.6. MyJoystickFunctions.c

Plik, w którym znajdują się funkcje obsługujące reagujące na joystick za pomocą funkcji zaimplementowanych w JoystickOpen_1768.c zaimplementowanych do obsługi płytki OPEN 1768.

2.6.1 struct GUI

Zawiera pola bool opt1, bool opt2 i wskaźnik na funkcję obsługującą GUI, która przyjmuje int i zwraca void.

2.6.1 void innitGUI()

Funkcja, która inicjalizuje globalną instancję struktury GUI o nazwie choice.

2.6.2 void resetOptions()

Funkcja, która przywraca wartości z inicjalizacji w instancji struktury GUI o nazwie choice na polach opt1 i opt2.

2.6.3 void changeOptions()

Funkcja, która zamienia wartości instancji struktury GUI o nazwie choice na polach opt1 i opt2 na przeciwne.

2.6.4 void JoystickManage()

Funkcja, która pobiera informację o stanie joysticka i w zależności od nich zmienia zamienia wartości na polach opt1 i opt2 z choice. Na naciśnięcie joysticka wywołuje funkcję, na którą wskazuje pole GUI z choice lub rozpoczyna wysyłanie wiadomości.

2.7. Biblioteki zewnętrzne

Do wykonania projektu użyto biblioteki stworzonej przez zewnętrznego autora <https://www.luisdigital.com/proyectos/nrf24l01/> odpowiadającej za konfigurację i obsługę na niższym poziomie modułu NRF24L01.

Powyższa biblioteka wymagała małych modyfikacji, aby doprowadzić ją do działania na LPC 1786:

- w pliku cpu_lpc1000.c, funkcja SSPInit(), 31-33 - zmiany pinów konfiguracyjnych SSP z SSP1 na SSP0.
- w pliku cpu_lpc1000.h - zmiany pinów z SSP1 na SSP0 odpowiedzialne za wejścia CE i CSN

Do obsługi wyświetlacza zostały użyte biblioteki: Open2768_LCD oraz LCD_ILI9325.

Do obsługi joysticka została użyta biblioteka Joystick_Open1768.