

Implementacja interfejsu 1wire w VHDL przy użyciu Spartan-3E oraz DS18S20

Krzysztof Cabała 210047

Kinga Wilczek 210063

23 maja 2016

Spis treści

1	Założenia projektowe	4
2	Wstęp teoretyczny	4
2.1	Interfejs	4
2.2	Komunikacja	4
2.2.1	Inicjalizacja i reset	4
2.2.2	Zapis bitu	4
2.2.3	Odczyt bitu	5
2.3	Konwersja i odczyt temperatury	5
2.4	Double dabble	5
3	Implementacja	6
3.1	Schemat główny	6
3.2	Termometer	6
4	Implementacja podstawowych operacji	8
5	Transmisja bajtu	10
6	Implementacja transmisji bajtu	10
7	Algorytm double dabble	10
8	Implementacja double dabble	10

Spis rysunków

1	Przykładowe połączenie urządzeń	4
2	Diagram czasowy dla procedury inicjalizacji	4
3	Diagramy czasowe dla procedury zapisu i odczytu	5
4	Schemat ogólny projektu	7
5	Moduł Termometer	8
6	Moduł obsługi podstawowych operacji bitowych	9
7	Maszyna stanów BusController	9
8	Symbol IOBuf	9
9	Moduł transmisji bajtu	10
10	Maszyna stanów ByteModule	10

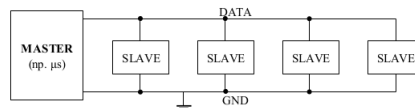
1 Założenia projektowe

Celem projektu było przygotowanie układu obsługującego sensor Dallas DS18S20 na platformie Xilinx Spartan-3E. W tym celu zaimplementowano obsługę magistrali One wire oraz interpretację wyniku. Efektem działania układu jest wyświetlona wartość temperatury na wyświetlaczu LCD zgodnym ze standardem HD44780.

2 Wstęp teoretyczny

2.1 Interfejs

Interfejs 1wire jest interfejsem opracowanym przez firmę Dallas Semiconductor do komunikacji między dwoma lub większą liczbą urządzeń przy wykorzystaniu zaledwie jednej linii danych, linii GND (konieczne odniesienie dla poprawnego rozpoznawania stanów logicznych) oraz zasilania Vcc. W ramach oszczędności przewodów ogranicza się połączenia do dwóch linii, wtedy układ zasilany jest pasywnie z linii danych. Przykładowe połączenie przedstawia schemat:



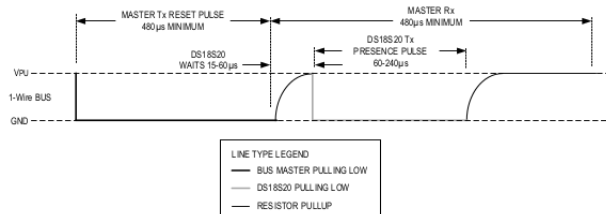
Rysunek 1: Przykładowe połączenie urządzeń

Wyróżnia się urządzenia typu Master (najczęściej mikrokontroler) oraz Slave (peryferia).

2.2 Komunikacja

2.2.1 Inicjalizacja i reset

Każda próba komunikacji urządzeń master i slave musi zacząć się od sekwencji składającej się z sygnału reset, wysyłanego przez master, po którym następuje sygnał obecności układu slave. Sygnał reset to wymuszony stan 0 trwający przynajmniej $480\mu s$. Następnie master oczekuje na sygnał obecności innego urządzenia na linii. Następuje wówczas zwolnienie magistrali, co powoduje podciągnięcie jej do stanu wysokiego przez rezystor pull-up. Urządzenie slave wykrywa wówczas narastające zbocze linii i po upływie $15\mu s$ - $60\mu s$ sygnalizuje swoją obecność poprzez wymuszenie stanu niskiego na okres $60\mu s$ - $240\mu s$.



Rysunek 2: Diagram czasowy dla procedury inicjalizacji

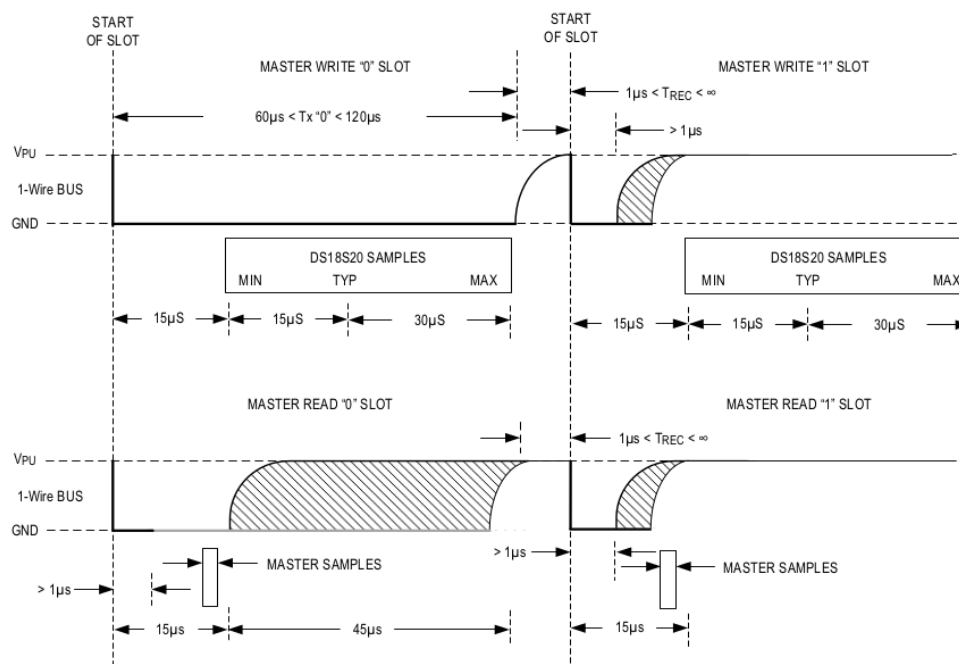
2.2.2 Zapis bitu

Operacje zapisu bitu realizowane są w ściśle określonych slotach czasowych. Długość jednego slotu wynosić zwykle $60\mu s$. Próbkowanie dokonywane jest mniej więcej w środku slotu celem uodpornienia na błędy. Pomiedzy kolejnymi operacjami wymagana jest przynajmniej $1\mu s$ odstęp.

Zapis rozpoczyna się wystawianiem linii danych przez master na poziom niski. Zapis 0 wymaga utrzymania jej w tym stanie przez cały slot. Zapis 1 jest nieco bardziej skomplikowany. Master musi w czasie nie dłuższym niż $15\mu\text{s}$, ale nie krótszym niż $1\mu\text{s}$ zwolnić magistralę tak aby w momencie próbkowania (po $30\mu\text{s}$ od zbocza opadającego) była w stanie wysokim.

2.2.3 Odczyt bitu

Odczyt bitu również wymaga $60\mu\text{s}$ slotu oraz $1\mu\text{s}$ przerwy. Odczyt rozpoczyna się wymuszeniem przez master stanu niskiego na linii danych na czas nie krótszy niż $1\mu\text{s}$ i zwolnienie jej (powrót do stanu wysokiego). Po tym sygnale sterowanie linią przejmuje urządzenie slave, wysyłające bit 0 lub 1. Slave po wykryciu zbocza opadającego wymusza stan niski (dla 0) lub utrzymuje wysoki (dla 1) linii danych. Sygnał musi być wtedy spróbkowany przez master. Przed upłynięciem czasu końca slotu magistrala zostaje zwolniona przez slave, co powoduje jej powrót do stanu wysokiego.



Rysunek 3: Diagramy czasowe dla procedury zapisu i odczytu

2.3 Konwersja i odczyt temperatury

Aktualny odczyt temperatury zapisany jest w dwóch pierwszych bajtach pamięci Scratchpad. Po poprawnej inicjalizacji czujnika DS18S20 ich zawartość odpowiada temperaturze $+85^{\circ}\text{C}$. W celu zmierzenia aktualnej temperatury należy wysłać do termometru komendę konwersji, zresetować układ i odczytać pamięć.

2.4 Double dabble

Double dabble jest powszechnie wykorzystywanym algorytmem do konwersji liczb binarnych na kod BCD (Binary-Coded Decimal - system dziesiętny zakodowany dwójkowo).

Algorytm polega na wykonaniu n iteracji (w zależności od długości ciągu bitów). Początkowo wynikowy kod BCD jest zainicjalizowany jako ciąg zer podzielonych na grupy po 4 bity. Podczas każdej iteracji wykonywane jest przesunięcie o jeden bit w lewo, a na koniec doklejany jest jeden bit ciągu wejściowego.

Jeżeli przed kolejnym przesunięciem wartość w jednej z grup w kodzie BCD jest wyższa niż 4 to następuje dodanie binarnej 3. Po wykonaniu odpowiedniej ilości iteracji algorytm kończy swoje działanie. Ostatecznie wynikowy ciąg jest podzielony po cztery bity, które odpowiadają kolejnym cyfrom wyniku.

3 Implementacja

3.1 Schemat główny

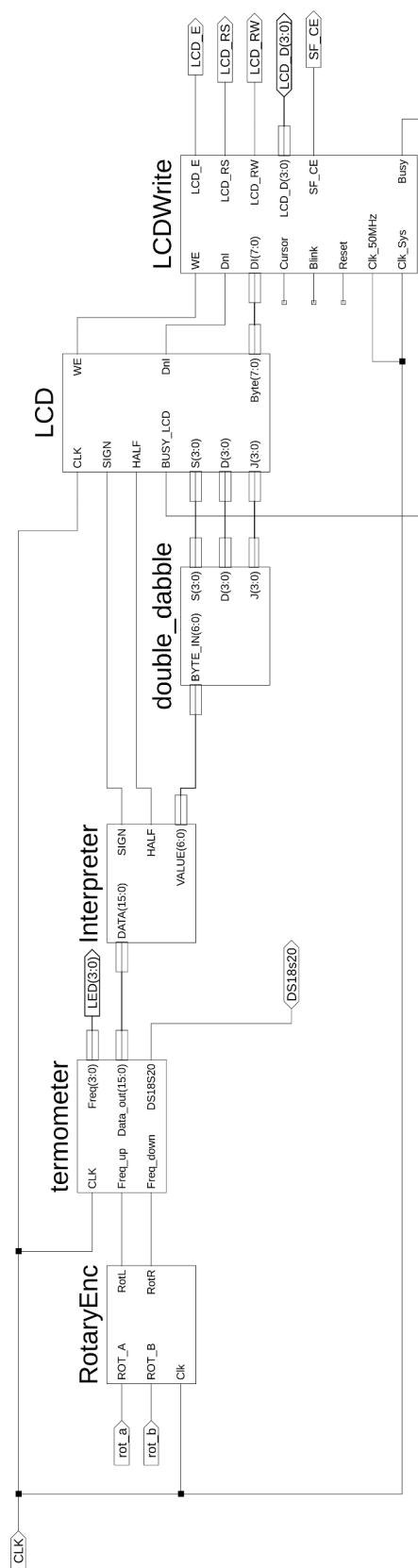
Funkcjonalność projektu opiera się na 6 podstawowych modułach widocznych na rysunku 4:

- Termometer - obsługa sensora DS18S20;
- RotaryEnc- sterowanie częstotliwością pomiaru;
- Interpreter - interpretacja 2 bajtowego wyniku;
- double_dabble - zamiana liczby binarnej bez znaku na kod BCD;
- LCD - generowanie zawartości wyświetlacza LCD;
- LCDWrite - sterownik wyświetlacza.

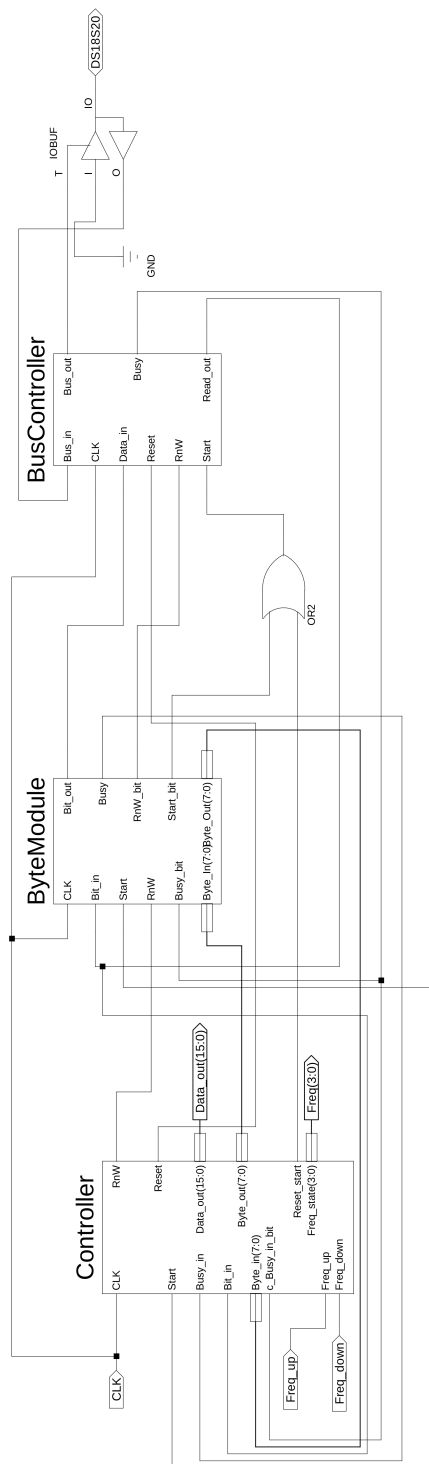
3.2 Termometer

Moduł Termometer jest najbardziej rozbudowanym modulem składającym się z 4 układów (rysunek 5):

- Controller - obsługa sekwencji komunikacji i konwersji temperatury;
- ByteModule - obsługa komunikacji na poziomie bajtu (instrukcji);
- BusController - obsługa magistrali Onewire;
- IOBuf - ustala kierunek komunikacji.



Rysunek 4: Schemat ogólny projektu



Rysunek 5: Moduł Termometer

4 Implementacja podstawowych operacji

Za realizację podstawowych operacji odpowiada moduł BusController.

Poniżej przedstawiono schemat blokowy tego automatu.

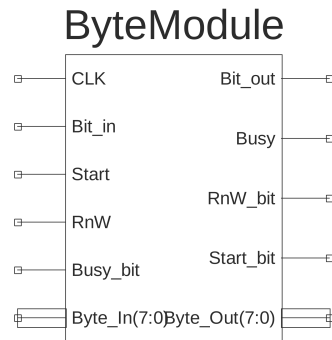
Poprawna obsługa magistrali wymaga jej zwalniania poprzez ustawienie w stan wysokiej impedancji (podciągnięcie do Vcc przez rezystor pull-up). Służy do tego element IOBuf. Podanie logicznego zera na wejście T otwiera bufor wyjściowy i przekazuje sygnał podawany na pin I (GND - Logiczne 0). Logiczne

5 Transmisja bajtu

Komunikacja master - slave jest dwukierunkowa. Dane przesyłane są w formie bajtów, kolejność bitów określa zasada najpierw najmłodszy.

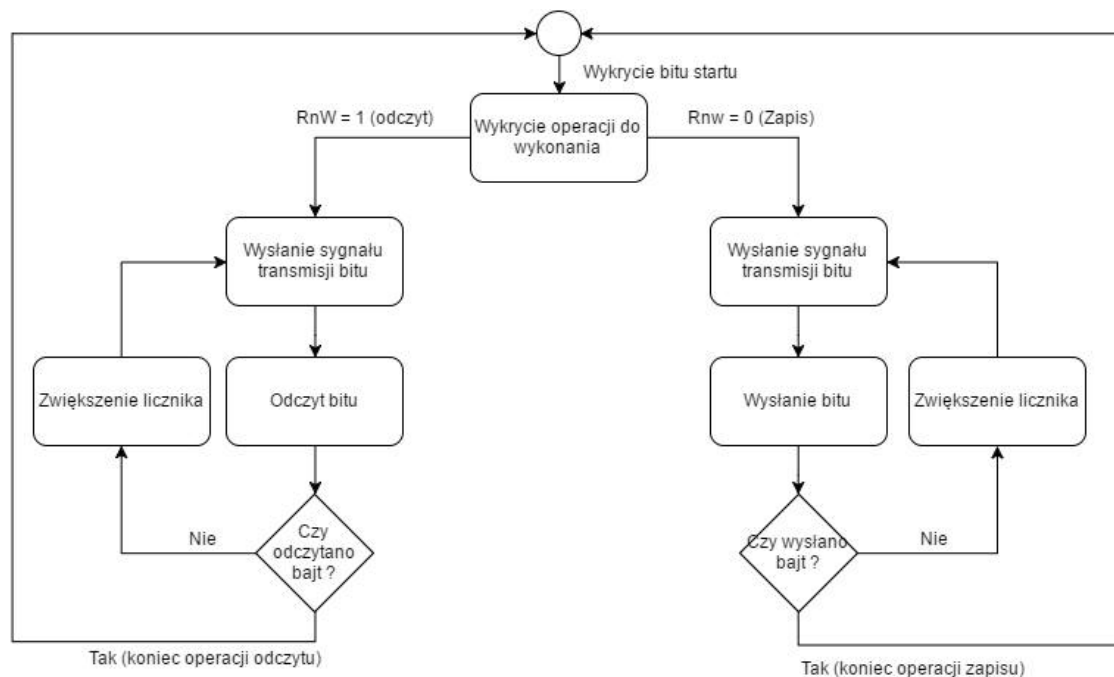
6 Implementacja transmisji bajtu

Transmisja bajtu realizowana jest poprzez następny moduł - ByteModule.



Rysunek 9: Moduł transmisji bajtu

Automat zawiera po 4 stany na odczyt i zapis oraz jeden stan oczekiwania na sygnał startu.



Rysunek 10: Maszyna stanów ByteModule

7 Algorytm double dabble

8 Implementacja double dabble