

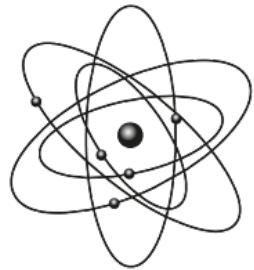
Zamek iButton

Projekt z Systemów Wbudowanych





A G H



**WYDZIAŁ FIZYKI
I INFORMATYKI STOSOWANEJ**

**Andrzej Brzyski
Kacper Starzyk**

Akademia Górnictwo-Hutnicza im. Stanisława Staszica w Krakowie
Informatyka Stosowana
2023

1 Cel projektu

Celem zajęć projektowych było stworzenie systemu na mikrokontrolerze LPC 1768 który będzie komunikował się z pastylką iButton DS1996 przy pomocy interfejsu elektronicznego 1-Wire.

2 Instrukcja dla użytkownika

Po załadowaniu programu do mikrokontrolera na płytce, po przyłożeniu do gniazda jednego określonego na twardo w kodzie programu iButtona (czarna obudowa pastylki) system rozpoznaje przyłożoną pastylkę jako poprawną i na ekranie LCD wyświetla się odpowiadająca wiadomość („Access granted”). Również wtedy mała lampka LED w gnieździe zapala się na krótki okres czasu. Jeżeli do gniazda zostanie przyłożony jakikolwiek inny iButton to na ekranie LCD zostanie wyświetlona wiadomość („Access denied”). Po wcisnięciu bazowego Joysticka na płytce w dół zostanie wyświetlona wiadomość „Change valid code”. Będzie ona widoczna do momentu aż użytkownik przyłoży do gniazda dowolny iButton zmieniając kod poprawny. Wtedy zostanie wyświetlona wiadomość „Code changed – press center button”. Tak jak wiadomość wskazuje system potwierdza zmianę kodu i po ponownym wcisnięciu joysticka w dół system powróci do bazowej funkcjonalności – sprawdzanie czy przyłożony iButton jest poprawny. Jednak tym razem odpowiedni kod będzie zawarty w pastylce, którą użytkownik zmienił. Wraz z rozpoczęciem zmiany kodu lampka LED w gnieździe zapali się i będzie włączona aż nowy kod zostanie ustawiony.

3 Implementacja

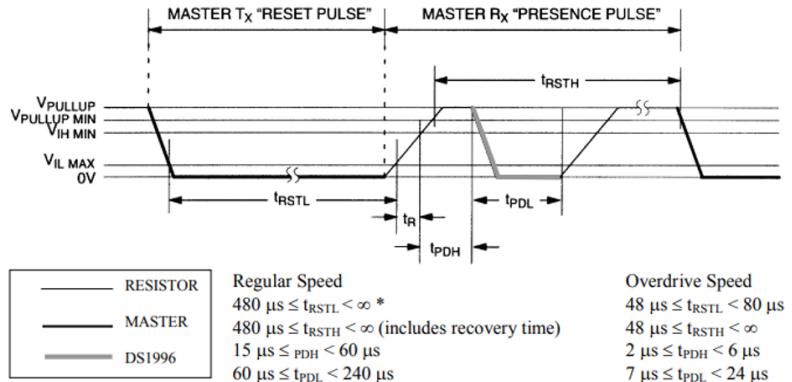
3.1 Elektronika

Gniazdo do iButtona zawiera kostkę z czterema kabelkami. Dwa odpowiadają za małą lampkę LED i one zostały odpowiednio podpięte do uziemienia i zasilania. Trzeci kabel to główne uziemienie gniazda a ostatni został podpięty przy użyciu płyty stykowej do zasilania 3.3V (użyto rezystora dołączonego w pakiecie z pastylkami), oraz jednego z pinu GPIO. Ekran LCD został podpięty do odpowiedniego wejścia na płycie, dla testów został również użyty interfejs komunikacyjny UART, który podpięto do płytka w jedno z 4 możliwych gniazd.

3.2 Kod

Na początku jest inicjalizacja ekranu LCD i Joysticka. Po tym ustawiane są parametry dla interfejsu UART tak by można było przy jego pomocy wysyłać na komputer dane, które są odczytywane przy użyciu sesji w programie MobaXterm . Następnie konfigurowany jest pin GPIO – została ustawiona funkcja 0, tryb został ustawiony na PIN_MODE_TRISTATE, a open drain został ustawiony na normal. Tristate pinu oznacza, że jest on w stanie wysokiej impedancji. Pin może ciągnąć do 5V, 0V albo stać się wysoką impedancją. Następnie zaczęto komunikację z iButtonem poprzez wystawienie na magistralę przez układ master impulsu zerującego. Trwa on 480 us i po tym czasie przestaje utrzymywać linie w stanie niskim i przechodzi w tryb odbierania. Magistrala jest wyciągania do stanu wysokiego przez pullup rezistor. Układ slave odpowiada na to impusem obecności. Jeżeli po 70us od zwolnienia magistrali przez mastera na pinie GPIO znajduje się 0(stan niski) to inicjalizacja przebiega pomyślnie.

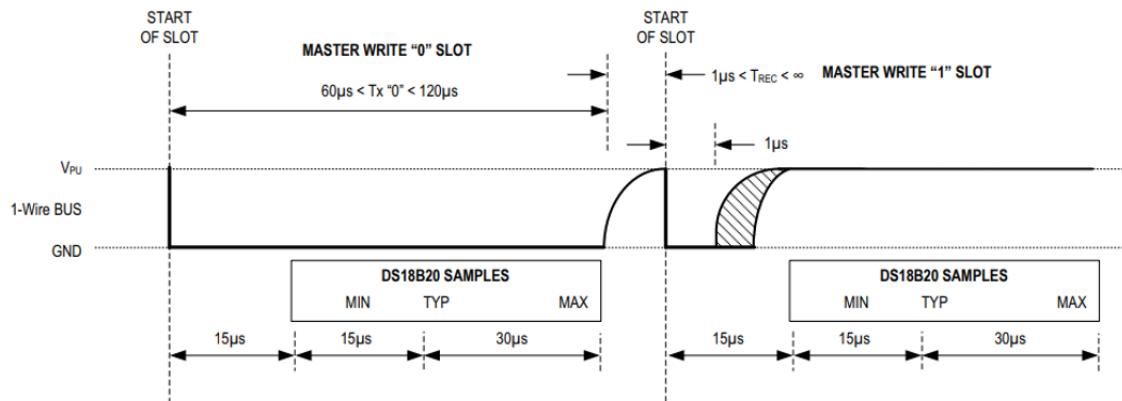
INITIALIZATION PROCEDURE "RESET AND PRESENCE PULSES" Figure 10



* In order not to mask interrupt signaling by other devices on the 1-Wire bus, $t_{RSTL} + t_R$ should always be less than 960 μ s.

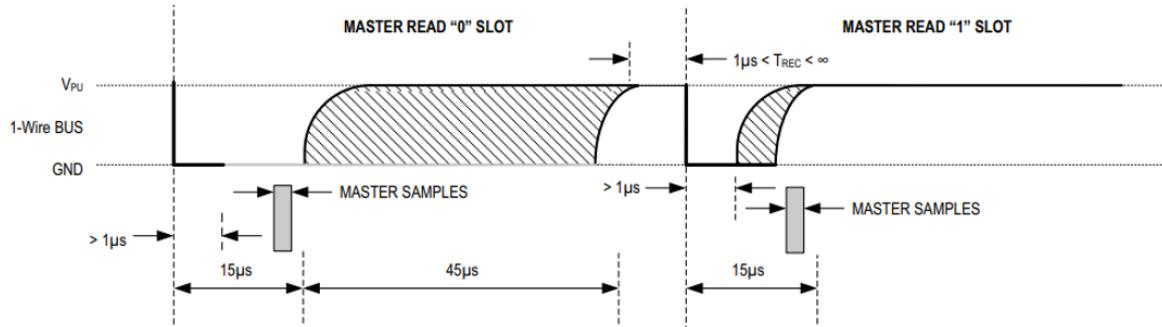
Rysunek 1: Procedura inicjalizacji z dokumentacji do danego iButtona

Odczytywanie danej pastylki zaczyna się od wywołania funkcji Read ROM która pozwala masterowi na przeczytanie kodu rodziny, kodu unikatowego urządzenia oraz 8 bitowej sumy kontrolnej. Komenda ta wymaga by tylko jeden iButton znajdował się w danej chwili na magistrali. Funkcja ta zostaje wywołana przez odpowiednie ustawianie linii na stan niski lub wysoki z odpowiednimi przerwami między nimi. Wpiswanie danych do urządzenia zachodzi podczas „Write Time Slots”. Aby wpisać 0 nakazano masterowi obniżenie magistrali i utrzymanie jej w takim stanie przez 100 us po czym pozwolono by rezystor pullup ustawił ją z powrotem na stan wysoki i oczekał 10 us. Aby wpisać logiczne 1 magistrala jest ustawiana na stan niski a po 8 us rezystor pullup ustawia znowu stan wysoki i oczekiwane jest 70 us. Proces ten jest powtarzany dla każdego bitu w komendzie.



Rysunek 2: „Write Slots” z dokumentacji do termometru DS18B20 – też wykorzystujący system 1-Wire

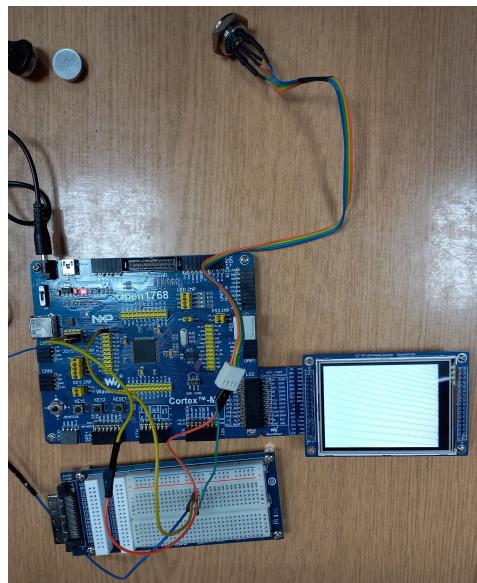
Po wywołaniu funkcji, master jest gotowy na odbieranie numeru identyfikacyjnego urządzenia. Proces ten odbywa się w trakcie „Read Slots” które polegają na ustawieniu magistrali na stan niski, oczekaniu 3 us po czym master pozwala rezystorowi na podniesie magistrali na stan wysoki i oczekuje 15us. Po tym czasie jeżeli magistrala jest w stanie wysokim to jest to logiczna 1 natomiast w innym przypadku master odczytuje wartość 0. Proces ten jest powtarzany dla każdego bitu w numerze seryjnym pastylki.



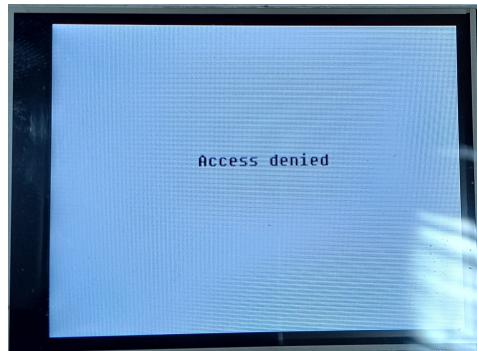
Rysunek 3: „Write Slots” z dokumentacji do termometru DS18B20 – też wykorzystujący system 1-Wire

Ostatnim elementem jest porównywanie tablic: z kodem poprawnego urządzenia, z kodem przyłożonej pastylki który został odczytany z użyciem powyższej procedury. Jeżeli kody są takie same to wypisuje się stosowny komunikat. Zmiana poprawnego kodu to nadpisanie jednej tablicy przez drugą. Kontrolowanie diody LED polega na zapisywaniu do odpowiednich rejestrów (FIOSET, FIOCLR) wartości 0.

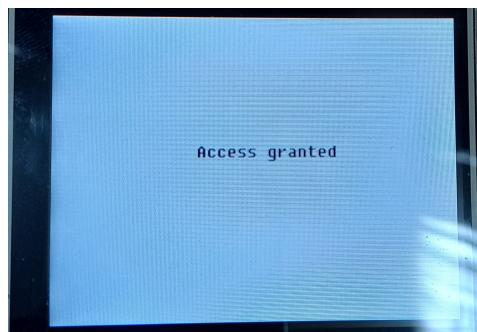
4 Prezentacja urządzenia



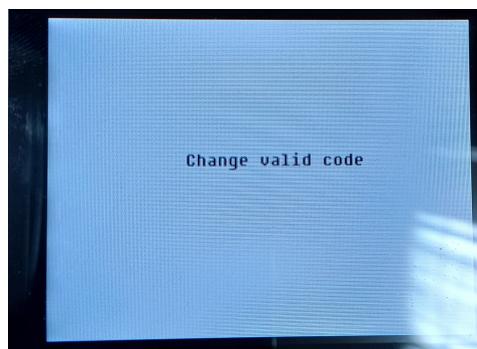
Rysunek 4: Zamek iButton



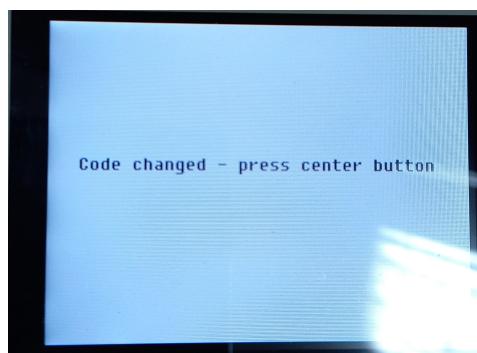
Rysunek 5: Ekran interfejsu w przypadku przyłożenia złego iButtona



Rysunek 6: Ekran interfejsu w przypadku przyłożenia dobrego iButtona



Rysunek 7: Zmiana kodu wchodzącego iButtona



Rysunek 8: Potwierdzenie zmiany kodu



Rysunek 9: Gniazdo iButtona