Politechnika Śląska w Gliwicach
Instytut Informatyki

USB
Uniwersalny Interfejs Szeregowy

Opracował:
mgr inż. Michał Maćkowski

1. Wstęp

Wraz z rozwojem systemów informatycznych wzrasta zapotrzebowanie na przepustowość interfejsów komunikacyjnych. Szeroka grupa urządzeń peryferyjnych stawia różne wymagania komunikacyjne, a istniejące już przez wiele lat standardy takie jak RS-232 czy IEEE-1284 nie były w stanie sprostać wymaganiom rynku. W takich okolicznościach powstał standard USB (Universal Serial Bus). Elastyczność i wygoda stosowania uniwersalnej magistrali szeregowej nie oznacza jednak, że USB jest interfejsem prostym. Poniżej warstwy aplikacyjnej kryje się szereg skomplikowanych mechanizmów komunikacyjnych obsługujących kilka rodzajów transferów oraz sposobów kodowania.

2. Charakterystyka systemu USB

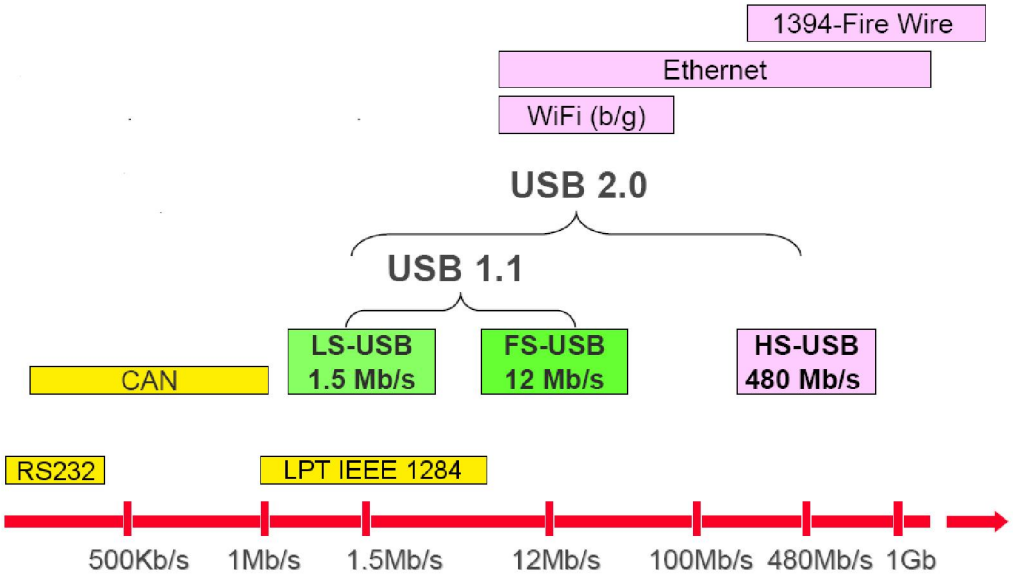
Rozwój systemów informatycznych oraz coraz większa grupa urządzeń elektronicznych mających możliwość wymiany informacji z komputerem skłoniła producentów sprzętu i oprogramowania do opracowania standardu komunikacyjnego, który zapewniłby prostotę dołączania urządzeń do systemu, a równocześnie dużą funkcjonalność oraz uniwersalność. Pojawienie się na rynku uniwersalnej magistrali szeregowej USB opracowanej przez firmy Microsoft, Intel, Compaq, IBM, DEC było rozwiązaniem dotychczasowych problemów związanych z ograniczoną ilością dostępnych w komputerze portów komunikacyjnych RS-232, LPT IEEE 1284 czy zbyt małą oferowaną przez te porty szybkością transmisji danych. Słowo „uniwersalność" w aspekcie omawianego standardu komunikacyjnego nie odnosi się jedynie do możliwości podłączenia różnego rodzaju urządzeń. Podstawowymi właściwościami interfejsu USB, dzięki którym uzyskał on taką popularność oraz nadaną przez producentów nazwę uniwersalnej magistrali szeregowej są:

- Automatyczne wykrywanie włączenia i odłączania urządzenia w systemie.
- Możliwość podłączenia do systemu dużej liczby urządzeń.
- Duży zakres szybkości transmisji: Low Speed - 1,5Mb/s, Full Speed - 12Mb/s, High Speed - 480Mb/s dopasowany do odpowiedniej klasy urządzeń.
- Zasilanie urządzeń prosto z portu.
- Przeprowadzanie bez udziału użytkownika wszystkich operacji konfiguracyjnych takich jak rozpoznanie urządzenia w systemie, nadanie adresu i instalacja sterownika.

3. Porównanie standardu USB z innymi standardami komunikacyjnymi

Szeroka gama dostępnych na rynku portów komunikacyjnych była powodem powstawania coraz większej ilości konfliktów oraz problemów z podłączaniem urządzeń peryferyjnych do komputera. Wraz ze wzrostem ilości danych przesyłanych pomiędzy urządzeniami i komputerem okazało się, że prędkości oferowane poprzez stosowane do tej pory standardy komunikacyjne takie jak RS-232 i LPT IEEE-1284 stały się niewystarczające.

Standardem są obecnie płyty główne posiadające wbudowane kontrolery szybkich zewnętrznych portów szeregowych (IEEE 1394 oraz USB) umożliwiających podłączanie zarówno urządzeń peryferyjnych (głównie USB), jak również cyfrowego sprzętu audio-wideo. Portem, który w dużej mierze „opiera się" dominacji rynku przez standard USB jest interfejs RS-232. Sytuacja taka ma miejsce głównie ze względu na jego użyteczność w zewnętrznych urządzeniach, związaną w dużej mierze z łatwością implementacji i obsługi. Kolejną zaletą RS-232 w stosunku do USB jest równoprawność podłączonych urządzeń (za to protokół nie określa kwestii związanych z adresacją i nie umożliwia podłączania wielu urządzeń). Chęć ciągłego korzystania z tych portów widać po popularności przejściówek między nimi a USB. Jednak mimo kilku zalet, jakimi charakteryzują się standardy RS-232, IEEE-1284 oraz PS/2 można zaobserwować praktycznie całkowitą ich degradację oraz brak implementacji tych interfejsów poprzez producentów nowych płyt głównych. Na rysunku 1 przedstawiono porównanie dostępnych na rynku standardów komunikacyjnych. Można zauważyć że standard USB dzięki zróżnicowanej prędkości transmisji dopasowanej do różnego rodzaju urządzeń w dużej mierze spełnia wymagania użytkowników oraz producentów sprzętu komputerowego oferując prędkości transmisji danych od 1,5Mb/s do 480Mb/s. Rozwiązaniem oferującym większą prędkość transmisji, a zarazem podobną prostotę użytkowania w porównaniu ze standardem USB jest standard 1394-FireWire. Standard FireWire, mimo większych możliwości, nie jest tak popularny jak USB głównie za sprawą polityki firmy Intel. Umieszczanie dodatkowego, dedykowanego układu (poza chipsetem płyty głównej) podnosi koszty produkcji płyty, dlatego producenci płyt implementują FireWire jedynie w droższych modelach. Jednocześnie organizacja producentów promujących standard USB pracuje nad wersją standardu USB 3.0, którego zadaniem będzie przejęcie części rynku obecnie zdominowanej przez interfejs FireWire.



Rys. 1 Porównanie dostępnych na rynku interfejsów komunikacyjnych.

4. Cel ćwiczenia

Celem ćwiczenia jest zaznajomienie się z budową i zasadą działania interfejsu USB. Zadanie polega na zaimplementowaniu w dostarczonym urządzeniu interfejsu, dzięki któremu urządzenie będzie widoczne w systemie operacyjnym jako mysz lub klawiatura USB.

Urządzenia USB o podobnych atrybutach i sposobie obsługi mogą korzystać z tego samego sterownika urządzenia (*device driver*) dzięki wprowadzeniu podziału urządzeń na klasy (*USB device class*). Informacje o klasie urządzenia znajdują się w deskryptorze urządzenia (na polach klasa, podklasa i protokół), informacje te wykorzystywane są następnie przez system operacyjny do zlokalizowania odpowiedniego dla danej klasy sterownika urządzenia. Wartość 0 na polach klasa i podklasa oznacza, że interfejsy w ramach danej konfiguracji mają własne kody klasy i działają niezależnie.

Klasy urządzeń USB objęte są oddzielnymi standardami i w nich, a nie w normie USB, należy szukać szczegółowych informacji o atrybutach i sposobie dostępu do urządzeń (interfejsów) danej klasy. Informacje te zawarte są w specyficznych dla klasy deskryptorach, które pobiera i interpretuje sterownik klasy urządzenia. Specyfikacja klasy ułatwia wykonanie sterownika urządzenia jak również oprogramowania komunikacyjnego w urządzeniu USB. Zbędne jest ponowne definiowanie atrybutów i usług, bo zawarte są one w specyfikacji – wystarczy tylko poprawnie je zaimplementować.

5. Klasa Human Interface Device

Klasa HID (*Human Interface Device Class*) obejmuje wiele urządzeń interfejsu użytkownika takich jak np.: mysz, klawiatura, joystick. Oprócz wymienionych urządzeń, w klasie HID znajdują się również wyświetlacze, indykatory, manipulatory, panele kontrolne itp. Należy zwrócić uwagę, że jak każda klasa, HID odnosi się do interfejsu w ramach wybranej konfiguracji urządzenia.

Urządzenie HID określają dwa dokumenty: *Device Class Definition* for *Human Interface Devices* (definiuje klasę HID) oraz *HID Usage Tables* (definiuje wartości umożliwiające hostowi właściwą interpretację danych HID) sporządzone i rozwijane przez organizacje *USB Device Working Group*.

Do komunikacji z hostem, urządzenia klasy HID wykorzystują transfery kontrolne i/lub przerwaniowe. Dane przesyłane pomiędzy hostem a urządzeniem HID są zorganizowane w specyficzne struktury zwane raportami. Odczyt raportu odpowiada przesłaniu danych z urządzenia do hosta (*Input Report*), zapis raportu odpowiada przesłaniu danych z hosta do urządzenia (*Output Report*).

Format raportu określa związany z nim deskryptor. Raporty mogą być bardzo proste (np. zawierać tylko kilka bajtów reprezentujących jedyną wielkość) lub bardzo złożone, obejmujące wiele parametrów o różnym znaczeniu, typie i rozmiarze.

Przekazanie raportów w urządzeniach z klasy HID może odbywać się:

- za pośrednictwem transferu kontrolnego z punktem końcowym 0 (wykorzystywany jedynie do przekazania danych, których czas dostarczenia nie jest krytyczny);

- za pośrednictwem transferu przerwaniowego z wejściowym punktem końcowym (IN endpoint - wykorzystywany do periodycznego odczytu danych, które muszą być dostarczone w określonych odstępach czasowych);
- za pośrednictwem transferu przerwaniowego z wyjściowym punktem końcowym (OUT endpoint - wykorzystywany do periodycznego zapisu danych, które muszą być dostarczone w określonych odstępach czasowych);

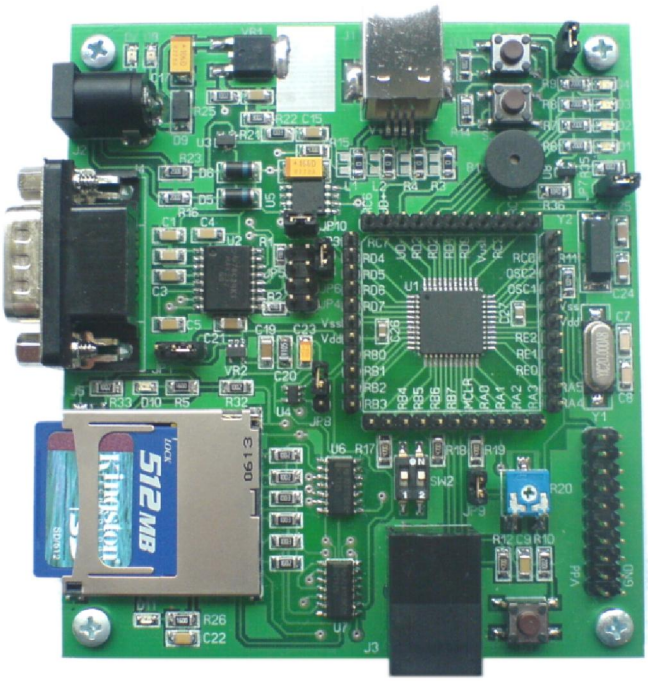
Wykorzystanie punktu końcowego 0 do przekazywania danych w ramach raportów nie jest zalecane ze względu na małą efektywność transferu kontrolnego. Wygodniejszy i bardziej optymalny do przekazywania raportów jest transfer przerwaniowy. Obowiązkowe dla każdego urządzenia HID jest posiadanie przynajmniej jednego wejściowego przerwaniowego punktu końcowego, natomiast wyjściowy przerwaniowy punkt końcowy pozostaje opcją.

Do przekazywania raportów za pomocą transferu kontrolnego służą dwa rozkazy specyficzne dla klasy HID: *SET_REPORT* i *GET_REPORT*. Specyficzne właściwości interfejsu klasy HID określają deskryptor klasy HID i związane z nim deskryptory raportów. W każdym urządzeniu HID występuje co najmniej jeden deskryptor raportu. Deskryptory raportu podzielone są na pola o określonych kodach, które definiują szereg parametrów raportu (*HID Usage Tables*).

6. Budowa urządzenia

Zaprojektowane i zbudowane do celów laboratorium urządzenie zostało oparte o 8-bitowy mikrokontroler PIC18LF4550 firmy Microchip. Mikroprocesor ten posiada zintegrowany kontroler USB zgodny ze standardem w wersji 2.0 i obsługującym tryby transmisji Low Speed 1,5Mb/s oraz Full Speed 12Mb/s, (brak jest trybu High Speed). Urządzenie ma postać zestawu uruchomieniowego (rysunek 2), poprzez wyprowadzenie na „zewnątrz" wszystkich interfejsów mikroprocesora. Zaimplementowano również obsługę kart pamięci Flash Secure Digital (komunikacja odbywa się poprzez interfejs szeregowy SPI mikrokontrolera), dzięki czemu urządzenie w zależności od wgranego oprogramowania jest rozpoznawane w systemie jako urządzenie:

- Klasy HID *(ang. Human Interface Device Class)* – do komunikacji z hostem wykorzystuje się transfery kontrolne i przerwaniowe.
- Klasy MSD (*ang. Mass Storage Device Class)* – do komunikacji z komputerem wykorzystywane są transfery kontrolne oraz masowe. Przykładem tej klasy są np. pamięci zewnętrzne Pendrive.



Rys. 2 Urządzenie z interfejsem USB – zestaw uruchomieniowy

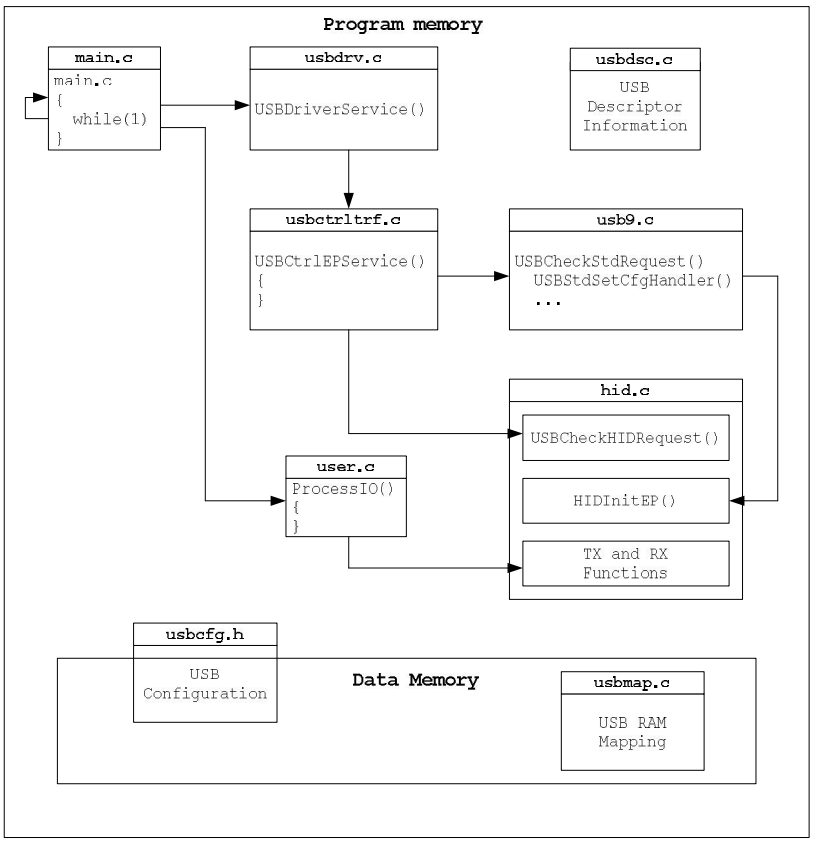
7. Budowa programu

Zawarte w mikrokontrolerze Microchip PIC18LF4550 oprogramowanie bazuje na modułowej budowie i implementuje rozkazy odpowiedzialne za komunikację poprzez interfejs USB. Na rysunku 3 przedstawiono schemat oprogramowania oraz uproszczoną ścieżkę wykonywania programu.

W głównej pętli programu wykonywana jest obsługa dwóch zdarzeń:

1. zdarzenia pochodzące od interfejsu USB, `USBDriverService()` (polling oraz obsługa przerwań od interfejsu USB);
2. zdarzenia pochodzące od użytkownika `ProcessIO();`

Sterowanie urządzeniem odbywa się przy wykorzystaniu transferu kontrolnego, dla którego USB otwiera kontrolny kanał komunikacyjny pomiędzy hostem a punktem końcowym 0 w urządzeniu. Standard USB definiuje 11 rozkazów standardowych (*Standard Request*), które służą do kontroli urządzeń. Zestaw ten mogą uzupełniać rozkazy związane z daną klasą urządzeń USB (*Class-specific Request*) zdefiniowane w normie opisującej klasę, jak również rozkazy specyficzne, określone przez producenta urządzenia. W momencie pojawienia się na magistrali USB transferu kontrolnego wywoływana jest funkcja, `USBCtrlEPService()`, której zadaniem jest obsługa punktu końcowego o numerze 0. Transfer kontrolny dzieli się na trzy etapy: etap przekazania rozkazu (*Setup Stage*), etap przekazania danych (*Data Stage*) i etap przekazania statusu (*Status Stage*). Przekazanie każdego rozkazu wymaga wykonania transakcji Setup określającej kod rozkazu (*Request*) i jego argumenty.



Rys. 3 Schemat oprogramowania oraz uproszczona ścieżka wykonywania programu

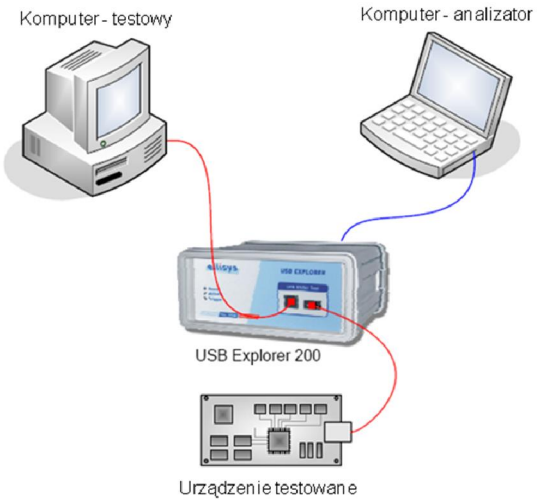
W zależności od przekazanego przez host kodu rozkazu, jego obsługa odbywa się w funkcji `USBCheckStdRequest()` dla rozkazów standardowych, zgodnie ze specyfikacją USB lub w funkcji `USBCheckHIDRequest()` dla rozkazów specyficznych dla danej klasy urządzeń (np. *Human Interface Device*).

Jedną z najważniejszych cech USB jest zdolność do automatycznego wykrywania urządzeń po włączeniu zasilania w systemie lub włączeniu urządzenia do systemu. Proces dodawania urządzenia do systemu nazywany jest procedurą enumeracji, której zadaniem jest rozpoznanie urządzenia, sprawdzenie czy w ramach dostępnych środków możliwa jest komunikacja z urządzeniem, przydzielenie adresu urządzeniu, skonfigurowanie urządzenia i instalacja odpowiedniego sterownika pośredniczącego w komunikacji z urządzeniem. Powyższy proces realizowany jest przez funkcje zdefiniowane w pliku `usb9.c`. Proces enumeracji kończy się wybraniem odpowiedniej konfiguracji w urządzeniu za pośrednictwem odpowiedniego rozkazu – `USBStdSetCfgHandler()`. Innym zadaniem wymienionej funkcji jest wywołanie funkcji `HIDInitEP()` odpowiedzialnej za inicjalizację punktu końcowego do komunikacji z urządzeniem HID.

W celu wysłania lub odbioru komunikatu do klasy HID należy wykorzystać funkcje `HIDRxReport()` oraz `HIDTxReport()`. Parametry i sposób wywołania tych funkcji znajduje się w pliku `hid.c` (komentarz w nagłówku funkcji).

8. Stanowisko testowe

Po napisaniu oprogramowania, należy wykorzystać stanowisko testowe, które składa się z komputera, urządzenia z interfejsem USB oraz analizatora protokołu USB Explorer 200 firmy Ellisys, rysunek 4. Rola analizatora sprowadza się do przechwytywania wszystkich ramek przekazywanych pomiędzy komputerem testowym oraz urządzeniem i przesłania ich do drugiego komputera „analizatora" z zainstalowanym oprogramowanie do analizy przechwyconych ramek. Wykorzystanie analizatora umożliwia przetestowanie napisanego oprogramowania i przeanalizowanie komunikatów przesyłanych pomiędzy komputerem a urządzeniem.



Rys. 4 Stanowisko testowe służące do analizy przesyłanych danych z wykorzystaniem analizator sprzętowego - USB Explorer 200 firmy Ellisys.

9. Przygotowanie do ćwiczenia

a) Przeczytanie instrukcji.

b) Zaznajomienie się z materiałami z wykładów: http://rose.aei.polsl.pl/~wmiel/usb-wyklad.pdf
Wojciech Mielczarek: USB – Uniwersalny interfejs szeregowy, Helion 2005.

c) Zaznajomienie się ze specyfikacją standardu USB 2.0 „Universal Serial Bus Revision 2.0 specification": http://www.usb.org/developers/docs/ zwłaszcza z rozdziałem 9 „USB Device Framework" w pliku usb_20.pdf - budowa deskryptorów, interfejsów, punktów końcowych.

d) Zapoznanie się ze specyfikacją "Device Class Definition for Human Interface Devices (HID)": http://www.usb.org/developers/devclass_docs/HID1_11.pdf

e) Zapoznanie się z dokumentacją „HID Usage Tables":
http://www.usb.org/developers/devclass_docs/Hut1_12.pdf
zwłaszcza z rozdziałami: 3, 4 oraz 10.