W trakcie video przestawiam nieco protokół usb 2.0 bazując na ogólno dostępnych materiałach w sieci, piszę sterownik usb implementujący klase CDC, a następnie modyfikuje tą implementacje aby utworzyć urządzenie kompozytowe, dokładniej urządzenie zbudowane z klas CDC i HID jednocześnie.

Uogólniając video to ma na celu przedstawienie w jaki sposób korzystać z dostępnych informacji dotyczących protokołu usb, jak poruszać się po dokumentacjach USB, oraz jak samodzielnie napisać prosty sterownik usb korzystając z instrukcji obsługi mikrokontrolera. Video to powstało głównie dlatego że chciałem utrwalić sobie wiedzę dotyczącą USB. Zaobserwowałem że gdy przygotuje samodzielnie jakieś materiały zapamiętuje zdecydowanie więcej. Zatem aspekt edukacyjny dla ciebie widzu może być lepszy lub gorszy, wierzę jednak że coś wartościowego przedstawię jasno i korzystnie dla ciebie.

Niestety jak wszystko i to video ma swoje wady. Nie sposób przedstawić wszystkiego rzetelnie na temat USB, sam nie znam rozumiem jeszcze najlepiej tego protokołu i jest jeszcze sporo przede mną do poznania. A na obecną chwilę mam już serdecznie dość tego protokołu, i chcę zapamiętać to co już wiem żeby nie musieć tego więcej powtarzać..

Video pewnie będzie dosyć długie jednak nie chciałem go dzielić na części, trudniej się wówczas zmotywować do nagrywania. W związku z tym zamiast tego oznaczę tylko odpowiednio rozdziały jako poszczególne fragmenty.

Do dzieła!

1. O standardach można

USB 1.1 Urządzenia spełniające warunki tej specyfikacji mogą pracować z szybkością (full speed) 12 Mbit/s (1,5 MB/s) i (low speed) 1,5 Mbit/s (0,1875 MB/s). Data ogłoszenia specyfikacji: 23.08.1998.

USB 2.0 (high speed) Urządzenia zgodne z warunkami nowej specyfikacji mogą pracować z maksymalną szybkością 480 Mbit/s (60 MB/s). Rzeczywista szybkość przesyłu danych zależy od konstrukcji urządzenia. Według testów portalu CNet[12] maksymalna prędkość zapisu kształtuje się w granicach od 25 do 30 MB/s, a odczytu od 30 do 42 MB/s. Jest to głównie spowodowane tym, że transmisja danych przez port odbywa się w trybie half duplex na jednej parze przewodów. Urządzenia w standardzie USB 2.0 są w pełni kompatybilne ze starszymi urządzeniami. Data ogłoszenia specyfikacji: 27.04.2000.

USB 3.1 Gen 1 (SuperSpeed) (wcześniej noszący nazwę USB 3.0) Urządzenia zgodne z warunkami nowej specyfikacji mogą pracować z szybkością 5 Gbit/s[1]. W transmisji stosuje się kodowanie 8b/10b, przez co rzeczywista przepustowość łącza danych wynosi 4 Gbit/s (500 MB/s)[13]. Nowy standard oprócz standardowych przewodów (dla kompatybilności w dół z USB 2.0 i 1.1) do szybkich transferów wykorzystuje dwie dodatkowe, ekranowane pary przewodów w full duplex. Pierwsza prezentacja tej technologii odbyła się na targach Consumer Electronics Show 2008. Data ogłoszenia specyfikacji: 17.11.2008.

USB 3.1 Gen 2 (SuperSpeed+) Standard ogłoszony 31 lipca 2013. Do powszechnego użytku wszedł w 2015 roku. Prędkość maksymalna to 10 Gbit/s, a moc może wynosić 100 W. Standard 3.1 jest kompatybilny wstecz.

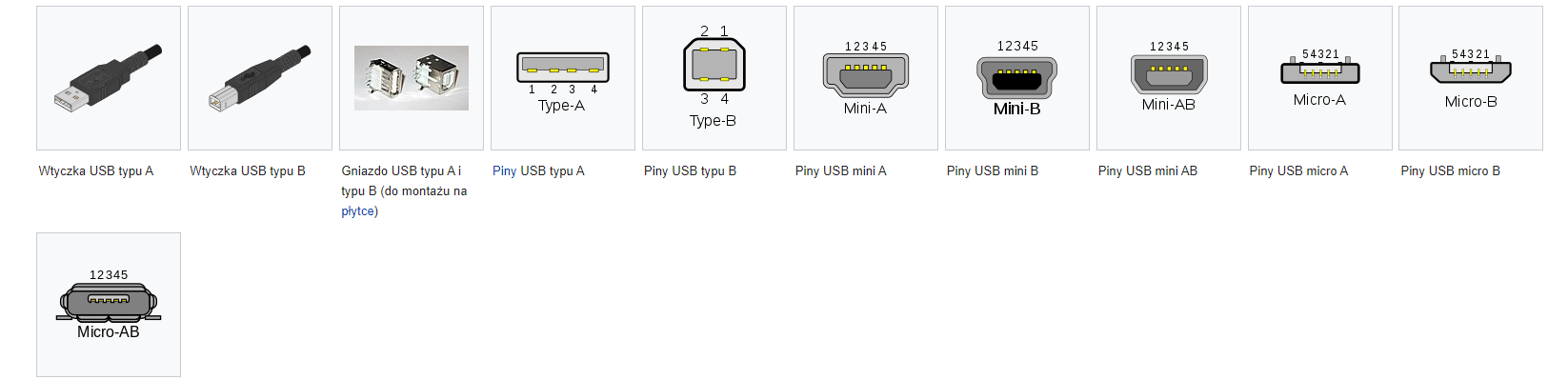
Standard złącza USB typ C 1.0 ogłoszony 11 sierpnia 2014 i typ C 1.1 ogłoszony 3 kwietnia 2015. Wymiary złącza to 8,3 na 2,5 mm. Ze względu na dużo większe możliwości zasilania nowym złączem zabronione jest tworzenie adapterów lub przewodów pozwalających podłączyć do hosta z gniazdem A urządzenie zaprojektowane dla złącza C. Nie należy nazywać złącza typu C złączem „3.1” – standard transmisji USB 3.1 Gen 2 można przesyłać z użyciem złącza typu A/B. Do złącza typu C zaprojektowano rozszerzenie pod nazwą USB Power Delivery pozwalające na negocjację ograniczeń napięcia i prądu dla jak najwydajniejszego ładowania lub zasilania urządzeń (nawet do 20 V i 5 A).

USB 3.2 Standard ogłoszony we wrześniu 2017, wstecznie kompatybilny. Wprowadza dwa nowe tryby SuperSpeed+ przez złącze USB typ C o szybkości transmisji danych wynoszące 10 i 20 Gbit/s (1250 i 2500 MB/s). Zwiększenie przepustowości pasma jest wynikiem operacji wielopasmowej na istniejących przewodach, które były przeznaczone dla funkcji flip-flop złącza typu C.

1. Podstawy

1.1 Wtyczki

https://www.usbmadesimple.co.uk/ums\_1.htm

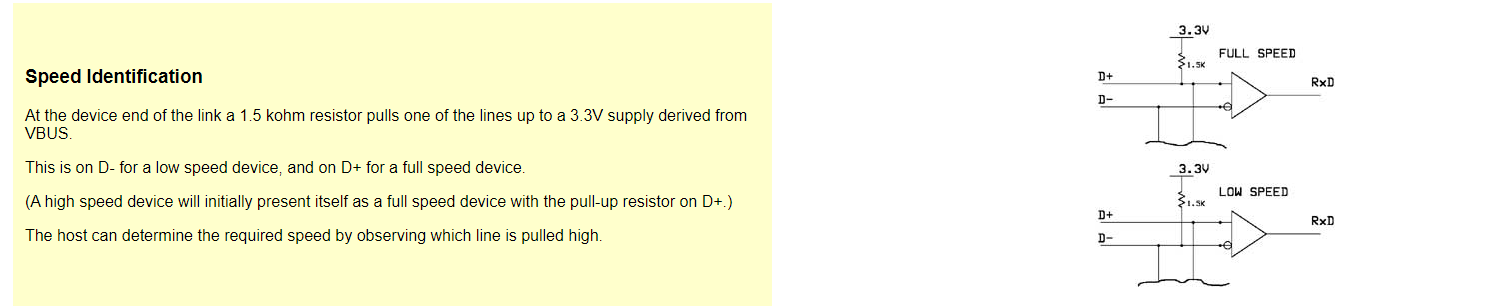


* 1. o przepływie danych

<https://www.usbmadesimple.co.uk/ums_3.htm>

* 1. Jak jest wykrywane podłączenie urządzenie, otg i host, te rezystorki terminujące

1. <https://www.usbmadesimple.co.uk/ums_3.htm>



urządzenie HS rozpoczyna pracę jako FS i w fazie zerowania negocjuje większą szybkość transmisji. Wykorzystując tak zwany świergot w czasie trwania sygnału zerującego, jeśli podłączane urządzenie sygnalizuje szybkość FS i kontroler lub koncentrator, który wykrył jego podłączenie, umożliwia komunikację HS. Procedurę negocjacji rozpoczyna urządzenie, wymuszając na liniach danych impuls świergotu K. Kontroler lub koncentrator potwierdza gotowość do pracy HS, wysyłając na przemian impulsy świergotu K i J. Jeśli procedura ta zakończy się powodzeniem, następuje przełączenie obwodu sygnalizacyjnego z wersji FS na wersję HS.

1. ta struktura gwiazdy

ale nic nie mowic za wiele na razie o hubach…

1. Pakiety w USB trzeba po opowiadać
2. O Deskryptorkach

<https://www.beyondlogic.org/usbnutshell/usb5.shtml#DeviceDescriptors>

1. Dokładnie o danych sterujących

Te w Indexy i inne

Żadania specyficzne dla klas…

1. potem ze dwa słowa o HUBach i ta struktura gwiazdy

<https://www.usb.org/document-library/usb-20-specification> - pokazac rozdzial o hubach

strona 455 w dokumentacji 2.0,

<https://techcommunity.microsoft.com/t5/microsoft-usb-blog/how-does-usb-stack-enumerate-a-device/ba-p/270685>

z tego ważne jest to zdanie na temat enumeracji..:

First Port Reset

Once the port debounce has completed successfully, the hub driver will issue a reset request for the port.  In normal operation this will result in the port status transitioning to a connected and enabled state, and the device itself will respond to the default USB address of 0.

**Port resets of all USB devices are serialized via an “enumeration lock” on a per host controller basis, as only one USB device can be enabled with the default USB address 0 at a any one time.  The hub driver will acquire the enumeration lock prior to issuing the first port reset request, and will release it when the device has been assigned a non-default USB address, or when enumeration has been cancelled.  The hub driver uses a 5 second timeout for the port reset request in case it never completes.**

While waiting for the first port reset to complete, the hub driver must be able to deal with the following events:

1. Enumeracja

<https://www.perytech.com/USB-Enumeration.htm> na podstawie tego opisać enumeracje w USB ze dwa słówka

Pokazać dokumentacje usb… jak najwięcej…

<https://www.usb.org/document-library/class-definitions-communication-devices-12>

<https://www.usb.org/document-library/device-class-definition-hid-111>

<https://www.usb.org/document-library/mass-storage-class-specification-overview-14>

1. I jazda z implementacją

**Out znaczy wyjściowy z HOSTA**

**IN znaczy wejściowy dla HOSTA**

Pokazać skąd pobrać cmsis driver

<https://github.com/orgs/STMicroelectronics/repositories?q=cmsis_device_f4&type=all&language=&sort>=

<https://github.com/STMicroelectronics/cmsis_core>

szukanie na git hubie stMicroelectronics

Na tej podstawie przygotowałem sterownik GPIO, Systemu i RTC

Strona 782 w RMie na temat urządzenia USB

iepint

pewna nie jasność, ale to jest tak że przerwanie TXFE jest generowane gdy faktycznie nie ma nic do wysłania w TxFIFO co jest powiązane z OTG\_FS\_DIEPTSIZx tzn. przerwanie faktycznie gdy „XFRSIZ==0” dla wybranego endpointa

no kiepsko to opisali jakoś tak poplątane ale dobra i oczywiście z tym rejestrem: OTG\_FS\_DIEPEMPMSK

No dobra czyli po ustawiane że przerwanie od FifoInEmpty może być, liczba danych ustawiona i potem trzeba sprawdzić w jakim endpoincie wystąpiło przerwanie ze fifo empty i podjąć akcje.

W moim przypadku implementuje że fifoEmptyTotal i mam że xfrc będzie chyba zawsze z txfe no ale to tam się pomyśli jeszcze

Wali przerwanie gdy jest to ustawione, gdy transfer size spadnie do zera,

O czym nie powiedziano/ powiedziec na koncu video:

O zarzadzaniu zasilaniem, budzeniach i itp.

Powiedziec ze hosta to możemy kiedys zaimplementowac..

O implementacj od ST tylko tych starych bazujących na standard peripherial libraries…

bo na halowska o dupe wytluc

Do wszystkiego najlepiej używać dokumentacji z usb.org i tego nutshell’a, cypressa, i innych ☺