

Interfejsy w Systemach Komputerowych - ULTIMATE

SonMati
Ervelan
Doxus

1 stycznia 2015

Pytania i odpowiedzi

1 RS-232

Prawda/Fałsz

- RS-232 jest portem przeznaczonym do synchronicznej transmisji znakowej. Generator taktu odpowiedzialny za wyprowadzanie znaków typowo ustawiany jest na: 1200bd, 2400bd, 4800bd, 9600bd, 19200bd.

RS-232 jest portem przeznaczonym do asynchronicznej transmisji znakowej. Da się sztucznie stworzyć synchroniczną transmisję.

- Linie kontrolne w interfejsie RS-232 to: DTR, DSR, RTS, CTS, RI, DCD. Pary DTR/DSR i RTS/CTS wykorzystywane są do realizacji handshake'u w połączeniach bezmodemowych.

Tak, te pary linii mogą być wykorzystywane do handshake'a podczas gdy RxT i TxT zajmują się przesyłem danych.

- Transakcja w systemie MODBUS składa się z zapytania (query) wysyłanego przez stację Slave i odpowiedzi odsyłanej przez stację Master.

Jest odwrotnie - zapytanie wysyła Master, a odpowiedź odsyła Slave.

- W trybie transmisji ASCII znacznikiem początku ramki jest znak ':', a kooca ramki para znaków CR LF. W trybie transmisji RTU znacznikiem początku ramki jest znak 'Ctrl-A', a kooca para znaków CTRL-Y CTRL-Z.

Zdanie jest poprawne dla ASCII. Dla RTU, znacznikiem początku i końca ramki jest przerwa o długości minimum 4T, gdzie T jest czasem trwania jednego znaku.

- Standard RS-232 transmituje znaki synchronicznie, bity w znakach [asynchronicznie]

Ostatnie słowo ucięte, więc spekuluję że tak właśnie było napisane. To nieprawda, jest odwrotnie.

- Standard RS-422 pozwala na osiągnięcie szybkości 10MBodów na odległość 100m.

IMO pozwala, na slajdzie 12 jest napisane że 10 Mbd przy zasięgu DO 100m - czyli 100m chyba też.

- Liniami kontrolnymi w RS-232 nie są linie TxT, RxT, SG.

Owszem, TxT i RxT są liniami danych, a SG to po prostu masa.

- System MODBUS składa się z faz zapytania i odpowiedzi.

Tak właśnie jest.

- W systemie MODBUS
 - Obowiązuje master/slave.

Peunie, a w dodatku Slave'ów może być wielu.

- Prędkości transmisji wynoszą od 1200 do 19200bd.

Jak najbardziej.

- Ramka w ASCII może mieć format 7N2 (lub np. 7E1, 7O1).

Tak, patrz warstwa fizyczna MODBUS.

- Ramka w RTU może mieć format 8N2 *(lub np. 8E1, 8O1).

Tak, patrz warstwa fizyczna MODBUS.

- W trybie transmisji RTU jest kontrola błędów CRC.

Tak, jest elementem budowy ramki RTU.

- Bit kontrolny w RS-232 zależy od bitu danych i bitu stopu.

Bit kontrolny służy do kontroli parzystości/nieparzystości, nie ma związku z bitem stopu.

- Za pomocą RS-232 możemy połączyć ze sobą 2 stacje DCE
Połączyć możemy dwie stacje DTE, lub DTE z DCE. Dwie stacje DCE łączą się za pomocą łącza telefonicznego.
- W MODBUS kontrola błędów jest realizowana za pomocą LRC lub CRC.
Tak, LRC wykorzystywane jest w trybie ASCII, CRC w trybie RTU.
- Do portu RS 485 można podłączyć tylko jedno urządzenie, ale za to obsługiwać go z dużo większą szybkością i na większą odległość niż jest to możliwe w przypadku interfejsu RS 232.
Można podłączyć do 32 stacji.
- Format ramki w protokole Modbus jest następujący: znacznik początku ramki, adres urządzenia slave, adres mastera, pole danych, znacznik końca ramki.
Opis nie pasuje ani do trybu ASCII, ani RTU
- RS 232 jest portem przeznaczonym dla asynchronicznej transmisji znakowej, realizowanej zazwyczaj w trybie dupleksowym, czyli dwukierunkowej transmisji niejednociennej (naprzemiennej)
Tryb dupleksowy jest równoczesny, to półduplexowy jest niejednocoesny.
- W interfejsie RS 232 linie TxD i RxD służą do transmisji znaków, natomiast DTR, RTS to wyjścia kontrolne, a DSR, CTS, RI i DCD to wejścia kontrolne.
Indeed
- Multipleksowanie urządzeń ze znakowym portem asynchronicznym pozwala na ich kontrolę poprzez jeden port RS-232.
Żeby kontrolować kilka urządzeń z jednego portu potrzebny jest koncentrator. Jeśli "używanie koncentratora" równa się "multipleksowanie", to PRAWDA.
- Węzeł podrzędny w systemie MODBUS po wykryciu błędu w komunikacie wysyła potwierdzenie negatywne do węzła nadzawanego.
W odpowiedzi pole to jest wykorzystywane do pozytywnego lub negatywnego potwierdzenia wykonania poleceń.
- Czy w trybie ASCII systemu MODBUS każdy bajt wysyłany jest jako znak z przedziału 0x00, 0xFF?
Bajt dzielimy na 2 części i wysyłamy jako 2 znaki z przedziału 0-9 i Ah-Fh

2 USB

Prawda/Fałsz

- Kontrola urządzenia USB odbywa się poprzez zapisy komunikatów do bufora o numerze 0 i odczyście informacji statusowych z bufora o numerze 0.
Zgadza się.
- W przypadku błędu transmisji każda transakcja USB jest powtarzana, ponieważ niedopuszczalne jest przekazywanie danych przekłamanych.
Transakcje izochroniczne nie są powtarzane w przypadku błędu transmisji.
- Hub nie dopuszcza ruchu full speed do portów, do których są podłączone urządzenia low speed.
Tak, urządzenie lowspeed blokuje możliwość włączenia fullspeed na całym porcie.
- Reset portu USB polega na rekonfiguracji hosta, po której host zapisuje tablicę deskryptorów do urządzenia podłączonego do tego portu.
Reset portu USB polega na rekonfiguracji urządzenia. W następcej procedurze enumeracji między innymi dochodzi do odczytu tablicy deskryptorów z urządzenia przez host.
- Typowa transakcja USB składa się z pakietów żądania i odpowiedzi, z których każdy potwierdzany jest osobnym potwierdzeniem.
Typowa transakcja USB składa się z pakietów token, data i handshake. Transakcje izochroniczne nie są potwierdzane.

- W systemie USB urządzenia zgłaszą żądania do hosta, który je kolejkuje i następnie obsługuje w kolejności pojawiania się zgłoszenia.
Urządzenia nie zgłaszą żądań, tylko są odpytywane przez hosta. Host nie tworzy jednej kolejki, tylko w miarę możliwości stara się obsługiwać wszystkie urządzenia jednocześnie, równomiernie, zapobiegając zawłaszczeniu.
- W USB można połączyć kaskadowo do 5 hubów, korzystających z zasilania magistralowego
Podłączyć je można tylko korzystając z zasilania zewnętrznego lub hybrydowego. Przy zasilaniu magistralowym zabraknie zasilania już na drugim hubie. Co więcej, należy mieć na uwadze maksymalne dopuszczalne opóźnienie sygnału, które przy przejściu przez 5 hubów jest osiągane - 350ns. Urządzenia podpięte do 5'tego huba mogą nie działać poprawnie.
- Mechanizm data toggle w USB służy do przywracania synchronizacji pomiędzy hostem i urządzeniem, utracone na skutek wystąpienia błędów w pakietach danych.
Mechanizm data toggle zabezpiecza przed utratą synchronizacji pomiędzy hostem i urządzeniem na skutek błędu w potwierdzeniu odsyłanym przez odbiorcę.
- Host kontroler USB komunikuje się z interfejsem magistrali USB urządzenia peryferyjnego za pomocą fizycznego kanału komunikacyjnego.
Tak, używamy kabla.
- Kamera internetowa może przesyłać obraz do komputera za pomocą transferu izochronicznego z szybkością LowSpeed w interfejsie USB.
Z tabelki można wyczytać, że dla transferu izochronicznego nie można wykorzystać szybkości LowSpeed.
- Pakiety USB przesyłane z szybkością LowSpeed muszą być poprzedzone pakietem preamble
Tak, jest on charakterystyczny dla pakietów przesyłanych z szybkością LowSpeed
- Urządzenie peryferyjne USB 2.0 może być podłączone do host kontrolera za pośrednictwem maksymalnie sześciu hubów.
Aby spełnić normę (ograniczenie czasowe oczekiwania na odpowiedź), można połączyć za pośrednictwem maksymalnie 5 hubów.
- Pole PID w pakiecie USB zabezpieczone jest 16-bitową sumą kontrolną CRC.
Pole PID zabezpieczone jest 4-bitowym polem kontroli, będącym prostą negacją bitów pola PID.
- Do portu dolnego huba podłączane mogą być tylko wtyki USB typu B.
Tylko wtyki typu A.
- Transakcja dzielona w USB 1.1 składa się z dwóch części: SSPLIT i CSPLIT.
Takie czary dopiero w USB 2.0
- W przypadku połączenia USB HighSpeed wykonywane jest podparcie linii D- do Vcc za pośrednictwem rezystora 1,5k.
Po podłączeniu urządzenia High Speed wpierw jest ono identyfikowane jako Full Speed, więc wykonywane jest podparcie linii D+ do Vcc za pośrednictwem rezystora 1,5k. Następnie, poprzez chirp ("dwierkanie") host i urządzenie ustalają, czy możliwa jest komunikacja w trybie High Speed. Jeśli tak, usuwane jest podparcie przez rezistor, a obwód zamykany jest terminatorem.
- W kodowaniu NRZI co sześć jedynek jest wstawiany bit synchronizacji "0".
Pomieszczone pojęcia. W kodowaniu NRZI nie występuje dodawanie bitu synchronizacji. Proces ten nazywa się bit stuffing. Zdanie było by poprawne, gdyby brzmiało np. W kodowaniu NRZI z bit stuffingiem co sześć.
- Transakcje kontrolna i przerwaniowa w USB 1.1 są transakcjami aperiodycznymi z gwarantowanym pasmem w ramach jednej mikroramki.
Transakcja kontrola jest transakcją aperiodyczną. Transakcja przerwaniowa jest transakcją periodyczną.
- W kontrolerze OHC transakcje izochroniczne są porządkowane/kolejkowane w drzewo/strukturę drzewiastą.
Tak, OHC wykorzystuje strukturę drzewa, a UHC tablicę wskaźników (listę podwieszaną).

- Standard USB 2.0 wymaga skręconych, ekranowanych kabli.

Well, High speed all the way, więc wymaga

- Transfer kontrolny i przerwaniowy są transferami aperiodycznymi.

Było podobne pytanie. Transfer kontrolny jest aperiodyczny, transfer przerwaniowy jest periodyczny.

- Wielowarstwowa architektura USB 2.0 składa się z 3 warstw.

Tak - warstwa interfejsu magistrali USB, warstwa urządzenia USB, warstwa funkcji urządzenia

- W porcie USB dane są dzielone na transakcje.

Dane w ramce są dzielone na transakcje, więc tak

- Hub podłączony do portu USB ma obciążalność 100uA.

Hub podłączony do portu USB bez własnego zasilania (zasilanie magistralowe) ma obciążalność dla portów dolnych do 100mA na port (maksymalną 400mA na cały hub). Hub z zasilaniem zewnętrznym lub hybrydowym ma obciążalność do 500mA na port.

- W systemie USB do mechanizmów kontroli danych należą:

– Przełączanie pakietów danych

Tzw. Data Toggle

– Wykrywanie braku aktywności na linii danych;

– Zabezpieczenie znacznika SOF lub EOF

Reakcją jest natomiast objęte wystąpienie falszywego znacznika końca pakietu (false EOP)

– kodowanie LRC

Pakiety zabezpieczone są kodowaniem CRC.

- Wydajność dolnego portu (USB 2.0) wynosi 500mA.

Nie wiadomo. Zasilany Hub może wystawić te 500mA, ale niezasilany już tylko 100mA

- USB 2.0 ma parę przewodów ekranowanych.

Taki upgrade.

- W kodowaniu NZR wstawia się dodatkowe bity synchroniczne.

Dodatkowe bity synchroniczne wstawia się w kodowaniu NRZI

- Urządzenie USB 2.0 może zasygnalizować swoją niegotowość do zapisu danych z szybkością High-Speed wysyłając pakiet PING-NYET.

Wychodzi na to, że niegotowość zgłasza samym NYET? Pyta – PING, odpowiada (niegotowość) NYET. I Tak cały czas, chyba że dostanie ACK. ACK – wykonanie transakcji OUT. NYRT – host kontynuuje wysyłanie zapytań PING

- W systemie deskryptorów urządzenia USB może wystąpić kilka deskryptorów urządzenia, konfiguracji, interfejsów I punktów końcowych.

Deskryptor urządzenia może być jeden. Innych – konfiguracji, interfejsu, końcowych może być więcej.

- Hub USB ma przerwaniowy punkt końcowy, który wykorzystuje do powiadamiania hosta o podłączeniu urządzenia USB do któregoś z jego portów dolnych.

Chyba.

- Na wierzchołku wielopoziomowego, hierarchicznego układu deskryptorów USB znajduje się deskryptor konfiguracji. Na szczycie znajduje się pojedynczy deskryptor urządzenia.

- Transfer masowy I izochroniczny USB 1.1 są przykładami transferów aperiodycznych z zagwarantowanym pasmem w ramach jednej mikroramki.

Izochroniczny jest periodyczny, masowy nie ma zagwarantowanego pasma (wg tabelki z prędkościami)

- W deskryptorze konfiguracji USB jest jakiś pole statusowe, które mówi o maksymalnym poborze prądu. Dla wartości 50 urządzenie pobiera 50mA.

Pole to jest tak skonstruowane, żeby wartość zmieściła się w jednym bajcie, ze skokiem co 2mA. Dlatego urządzenie, które zglasza, że 50 może zasysać maksymalnie 100mA.

- Uszeregowanie transakcji w USB nie zależy od implementacji kontrolera.
W OHC przerwaniowe są w strukturze drzewa, a w UCH listy podwieszanej, co ma wpływ na uszeregowanie (do sprawdzenia).
- Host może zasygnalizować chęć zapisu danych do urządzenia wysyłając pakiet NYET do urządzenia USB 2.0, które z kolei odpowiada pakietem PING jeśli jest gotowe do zapisu.
To host posyła PING - zapytanie, czy urządzenie jest gotowe do zapisu. Te odsyła ACK - gotowe, lub NYET - jeszcze nie.

3 IEEE 1394 Firewire

Prawda/Fałsz

- Po resecie w systemie FireWire (IEEE1394) wykonywane są procedury TREEID i SELFID .
Po resecie następuje „TREEID” odpowiedzialne za ustalenie węzła głównego a później „SELFID” odpowiedzialne za rozesyłanie adresów do poszczególnych portów.
- Transakcja dzielona IEEE1394 umożliwia wykorzystanie magistrali przez inne węzły po zakooczeniu subakcji żądania (request), a przed wysłaniem odpowiedzi (response).
Tak, pomiędzy żądaniem a odpowiedzą magistrala jest wolna i można ją wykorzystać
- Transakcje asynchroniczne IEEE1394 są uprzywilejowane w stosunku do transakcji izochronicznych i w związku z tym zawsze wykonywane są na początku cyklu.
Jest odwrotnie. To izochroniczne są uprzywilejowane nad asynchronicznymi.
- Urządzenie IEEE 1394 będące konsumentem zasilania może posiadać co najwyżej jedno 6-kontaktowe gniazdo IEEE 1394.
Tak po prostu jest.
- Numery węzłów IEEE 1394 nadawane są podczas procedury samoidentyfikacji na podstawie wartości wewnętrznych liczników odebranych pakietów SelfID.
Tak, SELFID przypisuje każdemu węzlowi unikatowy identyfikator pełniący rolę adresu, a następnie rozsyła je w formie pakietów selfid z każdego portu do wszystkich pozostałych węzłów.
- Pakiet potwierdzenia odbioru asynchronicznego pakietu żądania zapisu bloku danych w pierwszej fazie transakcji asynchronicznej IEEE 1394 zawiera sumę kontrolną w formie parzystości.
Tak, zawiera kod potwierdzenia i parzystość - patrz budowa pakietów
- W IEEE 1394 pakiet nowego cyklu (SCP) jest wysyłany ZAWSZE co 125us.
Izochroniczne owszem, ale asynchroniczne w ramach interwału równych szans, którego długość zależy od liczby węzłów asynchronicznych jednocześnie żądających dostępu do łącza.
- IEEE 1394 posiada osobne pary ekranowanych przewodów (dla) TPA i TPB.
Tak wynika z przekroju budowy
- Pole adresowe w IEEE1394 składa się z numeru magistrali (10b), numeru węzła (6b) i adresu w węźle (48b).
Wszystko się zgadza.
- W systemie IEEE1394 węzeł A o szybkości S100 połączono z węzłem B o szybkości S200 za pośrednictwem węzła C o szybkości S400, co zwiększyło szybkość transmisji pomiędzy węzłami A i B w stosunku do ich połączenia bezpośredniego.
Nie, węzeł A wciąż nadaje z szybkością S100, a teraz dodatkowo musi przejść przez węzeł C
- W interfejsie IEEE1394 przerwa pomiędzy subakcjami transakcji asynchronicznych jest mniejsza od przerwy pomiędzy transakcjami izochronicznymi, co powoduje ich uprzywilejowanie podczas arbitrażu.
Izochroniczne mają pierwszeństwo, to raz. Dodatkowo przerwa pomiędzy izochronicznymi zazwyczaj jest krótsza od tej pomiędzy asynchronicznymi.

- Biorący udział w arbitrażu asynchronicznym węzeł A (dotyczy interfejsu IEEE 1394) nie może uzyskać dostępu do łącza, bo zdominował je asynchroniczny węzeł B, który ciągle wygrywa arbitraż. *Kolejność przydziału zależy od położenia węzła w drzewie systemu. Węzły położone bliżej korzenia uzyskają dostęp przed węzłami bardziej oddalonymi. Łączny czas wykonania subakcji przez wszystkie węzły nazywa się fairness interval. W tym czasie każdy węzeł uzyska dostęp do magistrali, aby wykonać subakcję. Dostęp do magistrali węzeł zarządzający przekazuje "rotacyjnie". Następna subakcja będzie mogła być wykonana w kolejnym interwale równych szans.*
- Odbiorca transakcji asynchronicznej wygrał arbitraż I odesłał pakiet odpowiedzi. Inicjator transakcji odebrał odpowiedź I oczekuje na wygranie arbitrażu w celu odesłania pakietu potwierdzenia. *Nie, istnieje coś takiego jak dostęp natychmiastowy - potwierdzenie wysyła się za pośrednictwem warstwy PHY, która nie rywalizuje o dostęp do magistrali.*
- W przypadku transakcji dołączanych (interfejs IEEE 1394) nie trzeba rywalizować o dostęp do magistrali w celu odesłania odpowiedzi. *Transakcja dołączana dostarcza dane przy potwierdzeniu, a ono nie wymaga arbitrażu.*
- W interfejsie IEEE 1394 szansa wygrania arbitrażu wzrasta wraz ze wzrostem odległości (mierzzone liczbą węzłów pośredniczących) węzła ubiegającego się o dostęp do magistrali od korzenia. *Jest na odwrót – bliżej korzenia, większe szanse.*
- Wartość przerw i ograniczeń czasowych w interfejsie IEEE 1394 są “na sztywno” określone przez standard i nie mogą być korygowane. *Przerwa pomiędzy subakcjami transakcji asynchronicznej może być regulowana.*
- W przykładowym interfejsie IEEE 1394 występują tylko transakcje asynchroniczne. Jeżeli uczestnikiem (inicjatorem lub odbiorcą) transakcji asynchronicznej jest korzeń, to zawsze wygra on arbitraż jako pierwszy. *Zgodnie z zasadą "Bliżej korzenia, większe szanse" korzeń wygrywa wszystko. Jednak występują transakcje izochroniczne i asynchroniczne.*
- Kontroler cyklu musi być korzeniem w topologii IEEE1394, bo musi wysyłać sygnał CSP. *Chyba.*
- Węzeł w IEEE1394, który zainicjalizował transakcję asynchroniczną może być odbiorcą transakcji zainicjalizowanej przez inny węzeł w tym samym Interwale Równych Szans. *Raczej tak, węzły asynchroniczne do wykonania transakcji nie wymagają alokacji pasma (subakcja). Also - zarządzanie magistralą stara się umożliwić jednoczesne jej wykorzystanie przez różne transfery.*

4 IEEE-488 i SCPI

Prawda/Fałsz

- GPIB (IEEE-488) jest interfejsem równoległym, opartym na 8-bitowej, 2 kierunkowej magistrali danych i 8 sygnalach sterujących: REN, IFC, ATN, SRQ, EOI, NRFD, NDAC, DAV *Tak, bity wysyła się ósemkami, stąd m. in. podaje się prędkość w bajtach na sekundę*
- SCPI to język programowania na bazie języka C wyposażony w biblioteki funkcji sterujących urządzeniami pomiarowo-kontrolnymi. *SCPI jest językiem kontroli urządzenia (i nie bazuje na C).*
- Znak ‘:’ w rozkazach SCPI reprezentuje przejście pomiędzy poziomami w rozgałęzionej strukturze subsystemu, natomiast prefiks ‘*’ oznacza rozkaz wspólny. *Tak, dwukropki służą do precyzowania zapytania, gwiazdka jako nagłówek komunikatu wspólnego.*
- System statusowy urządzenia SCPI składa się tylko z jednego, 8-bitowego rejestrów, w którym bit B6 jest zgłoszeniem żądania obsługi. *Sklada się z minimum dwóch rejestrów, których układ jest wielopoziomowy (hierarchiczny).*

- Kontrola szeregową I kontrola równoległa to mechanizmy automatycznego wykrywania urządzeń podłączonych do systemu IEEE 488.

Kontrola szeregową I równoległa służą do identyfikacji urządzeń zgłaszających żądanie obsługi.

- Maska związana z bajtem statusowym SCPI służy do blokowania ustawiania wybranych bitów bajtu statusowego.

maską związaną z bitem statusowym jest rejestr maski żądania obsługi, który odpowiada za selekcję bitów powodujących zgłoszenie żądania, ale nie ma on wpływu na stan samych bitów w bajcie statusowym. Bajt statusowy jest rejestrem zbiorczym swoich rejestrów nadzorczych, więc jego wartość zależy od wartości tamtych rejestrów i ich masek.

5 Inne

Prawda/Fałsz

- Interfejsy USB, IEEE1394 oraz RS-232 udostępniają zasilanie systemowe i mają mechanizmy zarządzania zasilaniem.

USB i IEEE-1394 owszem, ale nie RS-232.

Opracowanie materiałów

1 RS-232 – szeregowy port znakowy

1.1 Co to jest?

Standard RS-232 został wprowadzony w 1962 r. w celu normalizacji interfejsu pomiędzy *urządzeniem końcowym dla danych* (DTE - Data Terminal Equipment), a *urządzeniem komunikacyjnym* (DCE - Data Communication Equipment). Na zajęciach zajmujemy się tak naprawdę zrewindowaną wersją standardu: RS-232C, wprowadzoną w 1969 roku.

RS-232C umożliwia przesył danych na niewielkie odległości - do 15 metrów - oraz niewielką szybkość - do 20 kb/s - przez niesymetryczne łącze.

1.2 Charakterystyka interfejsu RS-232

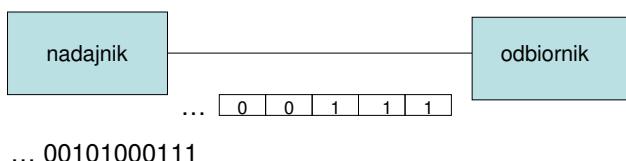
Łącze szeregowe przeznaczone do asynchronicznej transmisji znakowej realizowanej zazwyczaj w trybie półdupleksem.

1.2.1 Transmisja danych

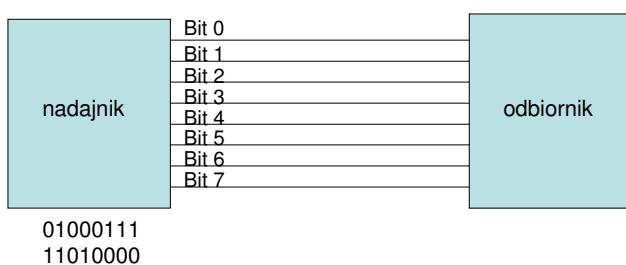
Szeregową asynchroniczną transmisję znakową w trybie półdupleksem (praktycznie tylko w tym trybie, ale mogą być inne). CIEKAWOSTKA: ma budowę dupleksową.

1.2.2 Rodzaje transmisji

- Szeregowa - sekwencyjne przesyłanie bitów w ustalonej kolejności (od LSD lub MSB) po jednej linii transmisyjnej.



- Równoległa - przesyłanie bitów słowa po przyporządkowanej każdemu bitowi linii transmisyjnej (bitы przesyłane równolegle, słowa przesyłane szeregowo).



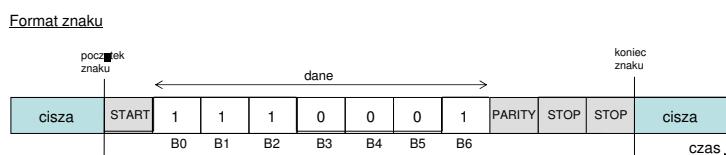
1.2.3 Definicje danych

- "0" - 0V
- "1" - 12V

- Czas transmisji jednego bitu T - stały, nie większy niż czas propagacji.
- $\frac{1}{T}$ - liczba bitów przesyłana w jednostce czasu. Standardowe wartości 110, 150, 300, 600, 1200, 2400 ... [b/s]
- Impulsy rozeznające - sprawdzają stan bitów odebranych (następują co T, które narzuca nadawca).

1.2.4 Jednostka informacyjna - znak

Jednostka informacji o ścisłe określonym formacie. Odbiorca dysponuje impulsami próbkującymi, które rozpoznają stan sygnału (odpytują).



1.2.5 Przekazywanie konfiguracji

Aby nadawca i odbiorca mogli się porozumieć i interpretować znaki w ten sam sposób, muszą zostać tak samo skonfigurowane. Innymi słowy, muszą posiadać ten sam takt nadawania.

Metody:

- dodatkowe łącze
- jako element konfiguracji (wykorzystanie generatorów kwarcowych, synchronizm częstotliwościowy)

Nominalne położenie impulsu = ok. $\frac{1}{2} \times T$ - pośrodku, największe bezpieczeństwo próbkowania. Służą do tego układy korekcji fazy impulsu - liczniki, zliczają liczbę impulsów na wejściu i dają 1 na wyjściu.

1.2.6 Definicja znaku

- **START** - bit kontrolny, znacznik początku (SOF - Start Of Frame) - jałowy z punktu widzenia przesyłanej informacji i służący jedynie w celu synchronizacji. START zapewnia $\frac{n}{2}$ jako stan licznika (fazy impulsu).
- **DANE** - 7-8 bitów (kiedyś też 5-6, obecnie już nieużywane), które są treścią znaku, począwszy od bitu najmniej znaczącego (LSB - least significant bit). Tym bitem jest B0.
- **PARITY** – bit kontroli poprawności znaku, służy jako zabezpieczenie informacji. Może, ale nie musi występować. Jednak decyzja o jego występowaniu ma charakter globalny - dotyczy każdego znaku w danej transmisji. Jego stan określa zasadę:
 - Kontrola parzystości (Even parity) - polega na sprawdzeniu liczby jedynek na polu danych i ustawnieniu bitu kontrolnego na "1" w przypadku nieparzystej liczby jedynek lub na "0" w przypadku parzystej (uzupełnienie do parzystości).
 - Kontrola nieparzystości (Odd parity) - polega na sprawdzeniu liczby zer na polu danych i ustawnieniu bitu kontrolnego na "1" w przypadku nieparzystej liczby zer lub na "0" w przeciwnym przypadku.
 - Brak kontroli (None)

Ten bit kontroli pozwala wykryć przekłamanie w transmisji danych pod warunkiem, że liczba przekłamań jest nieparzysta.

- **STOP** – 1 lub 2 bity kontrolne, znacznik końca znaku.

1.2.7 Konwencja nazewnicza rodzajów transmisji

[Ilość bitów danych][Rodzaj kontroli][Liczba bitów stopu]

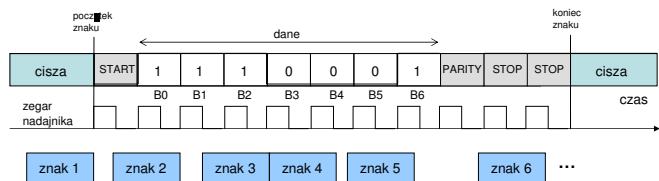
Przykłady:

- 7E2 - 7 bitów danych, kontrola parzystości, 2 bity stopu (10 bitów + START = 11 bitów)
- 8O1 - 8 bitów danych, kontrola nieparzystości, 1 bit stopu (11 bitów)
- 8N2 - 8 bitów danych, brak kontroli, 2 bity stopu (11 bitów)

1.2.8 Rodzaje transmisji

- **Synchroniczna** - elementy informacji wysypane w takt zegara nadajnika. W ten sposób przesyłane są bity w ramach pojedynczej jednostki informacyjnej.
- **Asynchroniczna** - wysyłanie elementów informacji niesynchronizowane zegarem nadajnika. W ten sposób są wysyłane poszczególne jednostki - ich wprowadzanie nie jest sygnalizowany żadnym sygnałem, więc odstęp między nimi jest dowolny.
Czas trwania bitu nazywa się *odstępem jednostkowym* i oznaczamy go t_b . Jego odwrotność ($f = \frac{1}{t_b}$) określa szybkość transmisji w bodach, gdzie 1 [bd] = 1 [bit/s].

1.2.9 Transmisja w RS-232



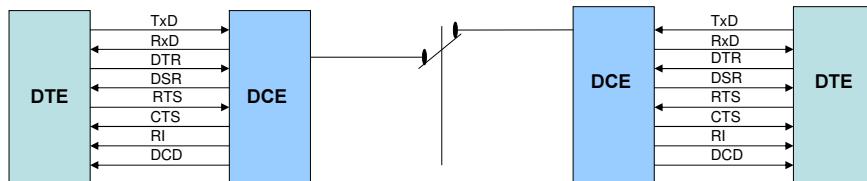
- Synchroniczne wysyłanie bitów
- Asynchroniczne wysyłanie znaków
 - Polega na wysyłaniu pojedynczych znaków, które mają ścisłe określony format.
 - Brak sygnału zegarowego określającego momenty wysyłania znaków.
 - Odstępy między znakami nieokreślone.

1.2.10 Tryby transmisji

- **Simpleksowa** - jednokierunkowa, z nadajnika do odbiornika.
- **Półdupleksowa (HDX)** - dwukierunkowa, niejednoczesna (w danej chwili czasu jedno urządzenie jest nadajnikiem, a drugie odbiornikiem). Zakłada istnienie tylko jednej linii transmisyjnej. Wymaga konfiguracji (informacja, kto kiedy nadaje).
- **Dupleksowa (FDX)** - dwukierunkowa, jednoczesna (w danej chwili czasu oba urządzenia mogą spełniać rolę nadajnika lub odbiornika). Brak konieczności sprawdzania czy łącze jest wolne oraz mechanizmu rezerwacji łącza.

1.3 Komunikacja DTE-DCE - sygnały w porcie RS-232

Komunikacja dwóch stacji DTE przez komutowane łącze telefoniczne.



| Urządzenia | | | | |
|-----------------|------------------------------|--------------------|-----------------|----------|
| DTE | Data Terminal Equipment | Komputer | | |
| DCE | Data Communication Equipment | Modem | | |
| Linie (sygnały) | | | | |
| Skrót | Nazwa | Znaczenie | Przeznaczenie | Kierunek |
| TxD | Transmitted Data | Dane nadawane | Linia danych | Wyjście |
| RxD | Received Data | Dane odbierane | Linia danych | Wejście |
| DTR | Data Terminal Ready | Gotowość DTE | Linia kontrolna | Wyjście |
| DSR | Data Set Ready | Gotowość DCE | Linia kontrolna | Wejście |
| RTS | Request to Send | Żądanie nadawania | Linia kontrolna | Wyjście |
| CTS | Clear To Send | Zgoda na nadawanie | Linia kontrolna | Wejście |
| RI | Ring Indicator | Wskaźnik wywołania | Linia kontrolna | Wejście |
| DCD | Data Carrier Detected | Wykrycie nośnej | Linia kontrolna | Wejście |
| SG | Signal Ground | Masa sygnałowa | Masa | — |

1.3.1 Fazy pracy układu

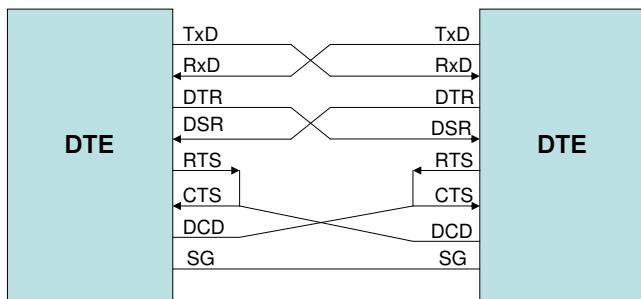
- Tryb nawiązywania połączenia
- Tryb transmisji danych (wtedy nas interesują dupleksy i inne)

1.3.2 Linie w złączu RS-232

- Linie danych: TxD, RxD
- Linie kontrolne: DTR, DSR, RTS, CTS, RI, DCD

1.4 Połączenie bezmodemowe DTE-DTE

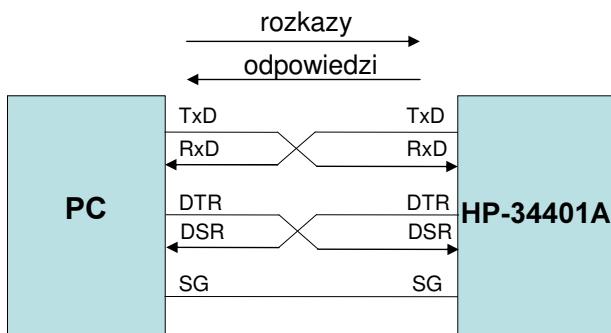
Przykład połączenia dla transmisji dupleksowej.



- PG, SG - masa
- TxD, RxD - dane
- RTS, CTS, DCD, DSR, DTR - sterowanie

1.5 Kontrola transmisji: handshake i protokół XON/XOFF

1.5.1 Handshake

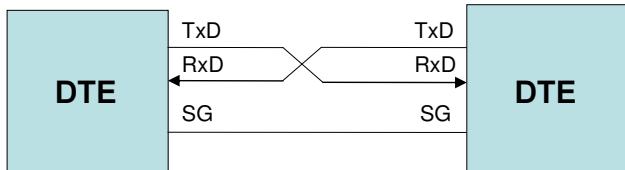


- DTR = 1 - zgoda na nadawanie
- DTR = 0 - brak zgody na nadawanie
- DTR informuje, czy bufor jest zapełniony. DSR sprawdza go u partnera przed wysłaniem dalszych danych.
- Analogiczna sytuacja, kiedy podłączone są RTS i CTS zamiast DTR i DSR. RTS wystawia informację, CTS sprawdza.

1.5.2 Protokół XON/XOFF

Występuje przy wymianie informacji w trybie dupleksowym. Umożliwia blokowanie i odblokowywanie transmisji danych. Np. drukarka - gdy skończy się papier w trakcie drukowania, przesył jest blokowany. Gdy użytkownik uzupełni papier, transmisja jest wznowiana. Taki protokół XON/OFF nazywany jest programowym (software XON/OFF). Rozwiążanie hardware to transmisja półduplexowa za pośrednictwem

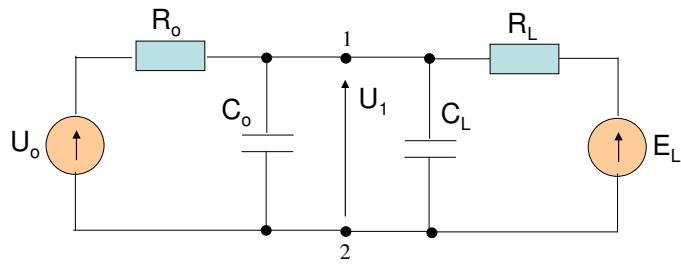
sygnałów w kanale wtórnym.



- XON – ASCII 19 (CTRL-S)
- XOFF – ASCII 17 (CTRL-Q)

1.6 Parametry elektryczne sygnałów

Poniżej przedstawiono schemat "obwodu stykowego" złożonego ze źródła sygnału, toru transmisyjnego i odbiornika. Parametry zdefiniowano przy założeniu, że szybkość transmisji nie przekracza 20 kbd.



Model obwodu transmisyjnego

Poziomy sygnałów

- $|U_o| < 25 \text{ V}$
- $3 \text{ k}\Omega < U_o < 7 \text{ k}\Omega$
- $I_{zwarcia} < 0,5 \text{ A}$
- $|E_L| < 2 \text{ V}$
- $C_o + C_L < 2500 \text{ pF}$
- Zmiana $U_1 < 30 \text{ V}/\mu\text{s}$
- 1 nadajnik – 1 odbiornik**

1. Sygnał danych:

- $-15 \text{ V} < U_1 < -3 \text{ V}$ 1 logiczna
- $+3 \text{ V} < U_1 < +15 \text{ V}$ 0 logiczne

2. Sygnały kontrolne:

- $-15 \text{ V} < U_1 < -3 \text{ V}$ 0 logiczne
- $+3 \text{ V} < U_1 < +15 \text{ V}$ 1 logiczna

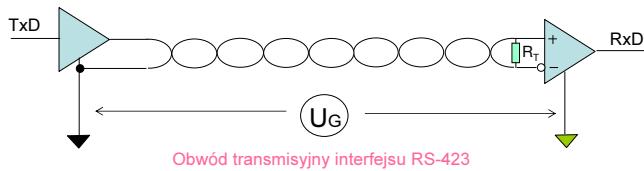
Na rysunku powyżej: kiloomy oraz mikrosekundy.

Wada: jest to obwód represyjny, da się go silnie zakłócić poprzez różnicę potencjałów pomiędzy masami.

1.7 Standardy RS-423, RS-422, RS-485

Niesymetryczna przesyłanie danych w RS-232C ogranicza szybkość i odległość transmisji, a ponadto nie jest zabezpieczone przed zakłóceniami zewnętrznymi. Aby to polepszyć wymyślono inne standardy.

1.7.1 RS-423A



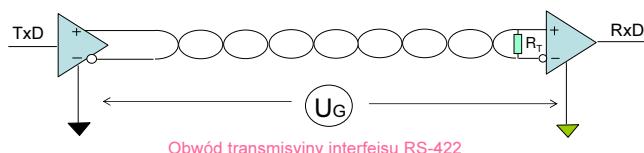
- szybkość do 100 kbd (przy zasięgu do 30 m)
- zasięg do 1200 m (przy szybkości do 3 kbd)

Standard RS-423A określa elektryczną charakterystykę napięciowego obwodu transmisyjnego złożonego z niesymetrycznego nadajnika oraz symetrycznego (różnicowego) odbiornika. Takie obwody stosuje się do przesyłania sygnałów binarnych pomiędzy DTE i DCE, które reprezentują dane lub funkcje sterujące. Zastosowanie różnicowego obciążenia pozwala na znaczne zmniejszenie wpływu napięcia wspólnego U_G powstało na skutek różnicy potencjałów masy nadajnika i odbiornika, jak również przesłuchów między nimi.

Standard wymaga aby dla każdego kierunku transmisji istniał przynajmniej jeden niezależny przewód powrotny.

Typowa prędkość wynosi 100 kb/s przy odległości do 30 m.

1.7.2 RS-422A



- szybkość do 10 Mbd (przy zasięgu do 100 m)
- zasięg do 1200 m (przy szybkości 100 kbd)

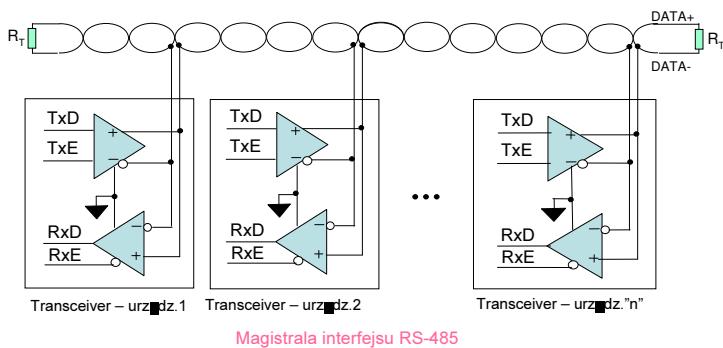
Wykorzystuje pełną symetryzację łącza, zapewnia szybką transmisję w obecności zakłóceń. Standardy RS-423 oraz RS-485 określają symetryczny, zrównoważony system transmisji danych złożony z:

- różnicowego nadajnika
- dwuprzewodowego zrównoważonego toru przesyłowego
- odbiornika o różnicowym obwodzie wejściowym.

Standard RS-422A nie wprowadza ograniczeń na minimalną i maksymalną częstotliwość, a jedynie na zależność między szybkością zmian sygnału, a czasem trwania bitu.

1.7.3 RS-485A

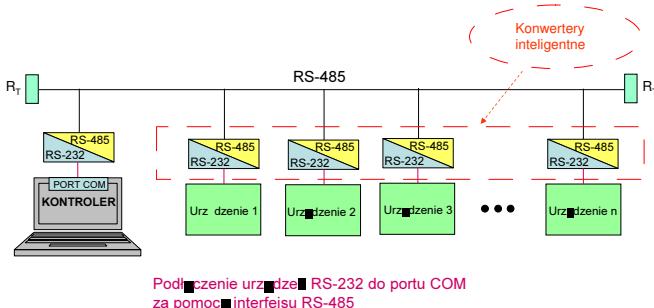
Standard RS-485A jest rozwinięciem RS-422. Łącznie RS-485A jest również zrównoważone i symetryczne, przy czym dopuszcza się nie tylko wiele odbiorników, ale i wiele nadajników podłączonych do jednej linii. Nadajniki muszą być trójstanowe.



1.8 Systemy komunikacyjne oparte na łączu znakowym

1.8.1 System oparty na szeregowym łączu znakowym

Podłączenie urządzenia RS-232 do portu COM z pom. int. RS-485



R_T- Rezystory zabezpieczające przed niekorzystnym odbiciem fali (tzw. Terminatory).

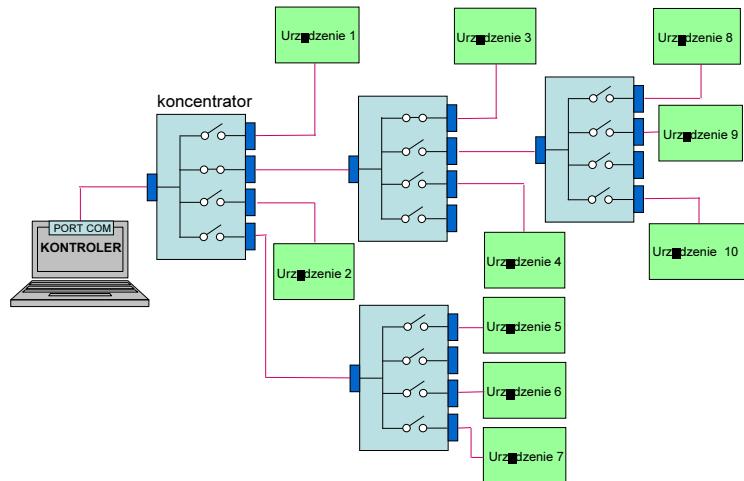
Problem: dostęp do magistrali kontrolera i urządzeń systemu

Rozwiązanie: Implementacja protokołu komunikacyjnego (warstwa łączu danych). Komputer zarządza innymi urządzeniami w całym systemie. Do zbudowania tego wystarczają proste przejściówki do zmian sygnałów.

Komunikacja:

- Selekcja urządzenia kontrolującego (master) - generuje on rozgłoszenie (broadcast) do wszystkich urządzeń i zbiera dane.
- Selekcja urządzenia odbierającego - konieczna gdy wiele urządzeń chce przesłać odpowiedź do mastera, co może powodować konflikt.
 - nadanie identyfikatorów (adresacja urządzeń)
 - zastosowanie przejściówek - są intelligentne i odpowiadają za dostęp do urządzenia.

1.8.2 System oparty na łączu znakowym



Koncentrator zawiera 4 klucze portu RS. Dostęp jest tylko do jednego wyjścia naraz.

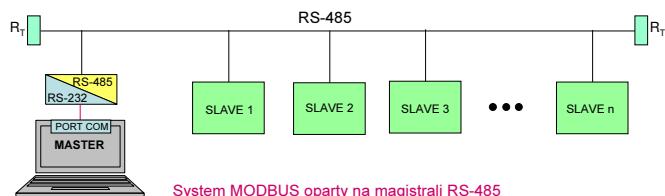
Kaskadowe połączenie - przełącznik podłączony do przełącznika. Pojawia się problem wyboru drogi do urządzenia, która musi być znana. Koncentratory muszą mieć informacje o **mapie urządzeń**.

1.9 System MODBUS

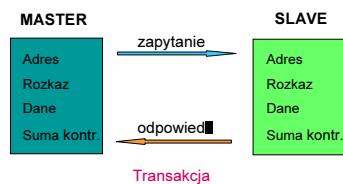
Interfejs MODBUS został opracowany w firmie Modicon i jest przyjętym standardem w dla asynchronicznej, znakowej wymiany informacji pomiędzy urządzeniami systemów pomiarowo-kontrolnych.

1.9.1 Charakterystyka

- Reguła dostępu do łącza na zasadzie Master-Slave.
- Zabezpieczenie przesyłanych komunikatów przed błędami
- Potwierdzenie wykonania rozkazów zdalnych i sygnalizacja błędów
- Mechanizmy zabezpieczające przed zawieszeniem systemu
- Wykorzystanie asynchronicznej transmisji znakowej zgodnej z RS-232C



System MODBUS oparty na magistrali RS-485



1.9.2 Transakcja

Jedno urządzenie może inicjować transakcje (master), a pozostałe (slave) odpowiadają jedynie na zapytania mastera. Transakcja składa się z polecenia (query) wysyłanego z master do slave oraz z odpowiedzi

(response) przesyłanej ze slave do master. Odpowiedź zawiera dane żądane przez master lub potwierdzenie realizacji jego połączenia. Wykrycie końca kończy fazę w której następuje przekazanie łącza masterowi.

1.9.3 Format wiadomości

Dotyczy zarówno poleceń jednostki nadzorowej, jak i odpowiedzi podległych.

- Adres
- Kod funkcji reprezentujący rozkaz, pierwszy bit rozkazu oznacza jego rodzaj. 0 - normalny, 1 - szczegółowy.
- Dane
- Kontrola błędów (dla pracy w warunkach przemysłowych)

W przypadku odpowiedzi odpowiednio w polach znajdują się:

- Adres (swój, slave'a, do kontroli poprawności)
- Pole potwierdzenia realizacji rozkazu
- Dane żądane przez master
- Kontrola błędów

1.9.4 Rodzaje transakcji

- **Adresowana** - przeznaczona dla pojedynczej jednostki slave
- **Rozgłoszeniowa** (broadcast) - wysyłana do wszystkich jednostek podległych. Na ten rodzaj polecenia jednostki nie przesyłają odpowiedzi.

1.9.5 Rodzaje odpowiedzi

- **Normalna** - w przypadku poprawnego wykonania polecenia.
- **Szczególna** - jeżeli slave wykryje błąd przy odbiorze wiadomości lub nie jest w stanie wykonać polecenia, to przygotowuje specjalny komunikat o wystąpieniu błędu i przesyła jako odpowiedź. W przypadku tej wiadomości jest ona **powiększona** o 128 - miejsce na kod błędu.

1.9.6 Parametry protokołu

- Reguła dostępu do łącza: Master-Slave
- Zakres adresów: 1 - 247 (identyfikatory slave'ów)
- Adres rozgłoszeniowy: 0, rozpoznawany przez wszystkie slave'y
- Kontrola błędów: LRC/CRC, ograniczenie czasowe odpowiedzi
- Wymagana ciągłość przesyłania znaków w ramce

1.9.7 Ramka w systemie MODBUS

W systemie MODBUS wiadomości są zorganizowane w ramki o określonym początku i końcu. Umożliwia do odbiornikowi odrzucenie ramek niekompletnych i sygnalizację błędów.

1.9.8 Rodzaje transmisji ramek

- ASCII
- RTU

1.9.9 Ramka w trybie ASCII

Każdy bajt wiadomości przesyłany jest w postaci dwóch znaków ASCII. Zaletą tego rozwiązania jest to, że pozwala na długie odstępy między znakami (1 s) bez powodowania błędów.

Format znaku

| SOF | | EOF | | | | |
|--------|-----------------|------------------|------------------|---------------|--------|--------|
| ':' | ADRES 1 bajt | ROZKAZ 1 bajt | DANE n bajtów | LRC 1 bajt | CR | LF |
| 1 znak | 2 znaki | 2 znaki | 2n znaków | 2 znaki | 1 znak | 1 znak |

- System kodowania: heksadecymalny, znaki ASCII 0-9, A-F. Jeden znak heksadecymalny zawarty jest w każdym znaku ASCII wiadomości.
- Jednostka informacyjna: ograniczona znakami start (na początku) i stop (na końcu), 10-bitowa.
- Znacznikiem początku ramki jest znak dwukropka (":") - ASCII 3Ah).
- Dopuszczalne znaki dla pozostałych pól (poza znacznikiem końca ramki) to 0-9, A-F.
- Pole funkcji: dwa znaki w trybie ASCII
- Podsumowując: wykorzystujemy 2 znaki heksadecymalne do przesyłu 1go znaku ASCII. Dzielimy ten na dwie części i przesyłamy w dwóch pakietach po 10 bitów.
- Urządzenie po wykryciu znacznika początku sprawdza czy pole adresowe zawiera jego własny adres. Jeżeli tak, to odczytuje zawartość pola funkcji i pola danych.
- Pole kontrolne LRC (1-bitowe) zabezpiecza część informacyjną. Sumuje część informacyjną bajtu i uzupełnia do 2.
- Ramka kończy się przesłaniem dwóch znaków: CR i LF.
- Ramkę kończy przerwa czasowa trwająca co najmniej $3.5 \times$ (długości znaku)
- Ramki muszą być przesyłane w postaci ciągłej, tzn. odstęp między kolejnymi znakami tworzącymi ramkę nie może być większy niż $1.5 \times$ (długości znaku).

Stosowane jest zabezpieczenie części informacyjnej ramki kodem LRC (Longitudinal Redundancy Check).

1.9.10 Ramka w trybie RTU

| SOF | | EOF | | | | |
|-------------------|-----------------|------------------|------------------|----------------|-------------------|--|
| | ADRES 1 bajt | ROZKAZ 1 bajt | DANE n bajtów | CRC 2 bajty | | |
| $\geq 4 \times T$ | 1 znak | 1 znak | n znaków | 2 znaki | $\geq 4 \times T$ | |

W trybie RTU wiadomości zaczynają się odstępem czasowym trwającym minimum $3.5 \times$ (czas trwania pojedynczego znaku), w którym panuje cisza na łączu (można to zrealizować np. przez odmierzanie czasu trwania znaku przy zadanej na łączu szybkości bodowej).

- Pierwszym polem informacyjnym jest adres urządzenia
- Dopuszczalne znaki w ramach pól ramki: 0-9, A-F
- Zakres kodów operacji: 1 - 255
- Urządzenia stale monitorują magistralę. Jak adres odebrany w wiadomości zgadza się z ich własnym, to lecą dalej.
- Ramkę kończy przerwa czasowa trwająca co najmniej $3.5 \times$ (długości znaku)

- W przypadku gdy nowa wiadomość pojawia się przed upływem niezbędnej przerwy to będzie ona potraktowana jako kontynuacja poprzedniej wiadomości. Doprowadzi to do błędu sumy kontrolnej.
- Ramki muszą być przesyłane w postaci ciągłej, tzn. odstęp między kolejnymi znakami tworzącymi ramkę nie może być większy niż $1.5 \times$ (długości znaku).
- Przekroczenie odstępu powoduje uznanie ramki za niekompletną i błędna.
- Kontrola danych typu CRC - 2-bajtowe, silniejsze niż LRC.

1.9.11 Warstwa fizyczna

- Asynchroniczna transmisja znakowa
- Formaty znaków
 - Tryb ASCII: 7E1, 7O1, 7N2
 - Tryb RTU: 8E1, 8O1, 8N2
- Szybkość: od 1200 bd do 19200 bd
- Rodzaj łącza:
 - Magistrala RS-485
 - Multipleksowany RS-232
- Rodzaj transmisji (zależny od łącza):
 - różnicowa dla RS-485
 - odniesiona do masy dla RS-232

1.10 Kontroler RS-232 w komputerze PC

2 USB – Uniwersalny interfejs szeregowy

2.1 Zalety

- Podłączenie dużej liczby różnorodnych urządzeń
- Automatyczne wykrywanie włączenia urządzenia do systemu oraz jego odłączenia
- Przeprowadzanie wszystkich operacji konfiguracyjnych, w tym instalacji sterownika, bez udziału użytkownika
- Szybka transmisja
- Zasilanie urządzenia bezpośrednio z portu

2.2 Parametry

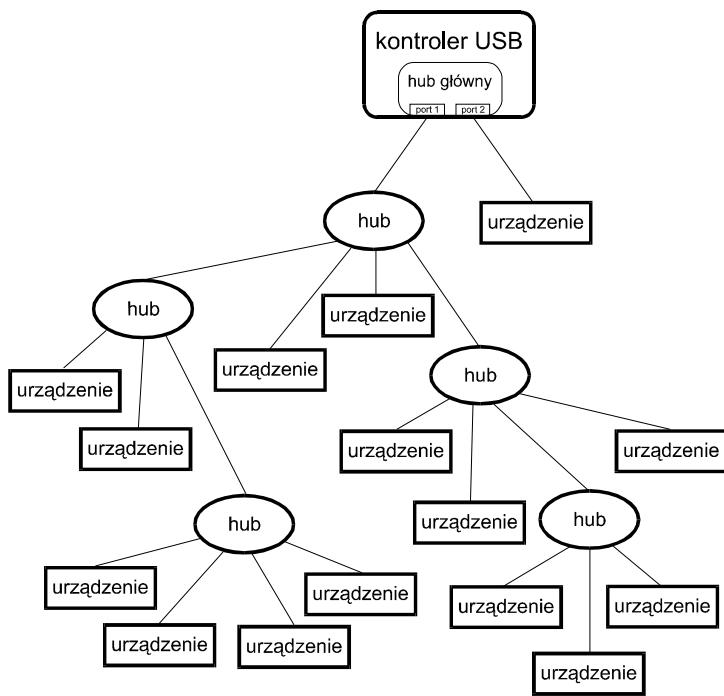
Dla USB 1.1 oraz 2.0

- Szybkość transmisji: 12 Mb/s (1.1), 480 Mb/s (2.0)
- Złożoność kontrolera w stacji host: 10000 bramek
- Złożoność kontrolera w urządzeniu: 1500 do 2000 bramek

2.3 Charakterystyka systemu USB

2.3.1 Podstawowe właściwości interfejsu USB

- **Gorące podłączenie** - włączanie/wyłączanie urządzeń bez wyłączania systemu oraz konfiguracja bez udziału użytkownika.
- **Jeden typ złącza** - złącze typu A w koncentratorze i B w urządzeniu (dwa kontakty linii danych oraz dwa zasilania).
- **Duża liczba podłączanych urządzeń** - moduły komunikacyjne, jak hub czy koncentrator, posiadają kilka portów USB oraz umożliwiają łączenie kaskadowe w celu rozszerzenia liczby portów do podłączanie urządzeń. W ten sposób można uzyskać wielopoziomową gwiazdową strukturę. Ostatecznie do USB można podłączyć maksymalnie **127** urządzeń.
- **Różne szybkości transmisji**
 - Mała (*low speed*): 1,5 Mb/s
 - Pełna (*full speed*): 12 Mb/s
 - Wysoka (*high speed*): 480 Mb/s
- **Zasilanie** - USB posiada własny system dystrybucji zasilania: minilanie dostarcza 4.7 V/100 mA, maksymalnie 500 mA. Przy braku aktywności na magistrali przez 3s urządzenia przechodzą w tryb zmniejszonego poboru prądu (SUSPEND), gdzie mogą pobierać maksymalnie 500 μ A. Wyjście z tego trybu jest możliwe poprzez inicjatywę hosta bądź urządzenia (*remote wake-up*)
- **Protokół komunikacyjny, detekcja błędów** - obowiązuje złożony, pakietowy protokół k. z kontrolą poprawności przesyłu. Żądania przesłania danych (*transfer*) dzielone są na transakcje, które z kolei złożone są z pakietów: kontrolnego, danych i potwierdzenia. Pakiety są zabezpieczone sumą kontrolną. W przypadku wykrycia błędu jest możliwe powtórzenie transakcji.
- **Transfery USB** - 4 typy transferów:
 - Kontrolny (control transfer)
 - Masowy (interrupt transfer)
 - Przerwaniowy (bulk transfer)
 - Izochroniczny (isochronous transfer)
- **Niski koszt** - nieduża złożoność układów interfejsowych
- **Schemat blokowy systemu USB**



2.3.2 Środowisko sygnałowe

Łącze transmisyjne oparte jest na obwodzie różnicowym. Różnicowy nadajnik jest połączony z różnicowym odbiornikiem przez parę przewodów.

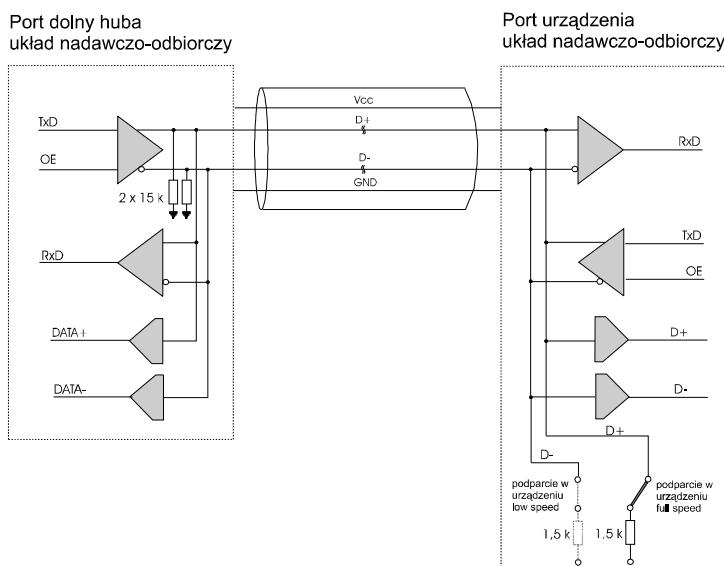
W przypadku transmisji wolnej wystarcza para nieskręcana i nieekranowana, natomiast transmisja pełna lub wysoka wymaga pary skręcanej i ekranowanej.

Zasilanie jest przekazywane przewodami $V_{cc}(+5V)$ oraz GND (masa).

Trójstanowy nadajnik można zablokować sygnałem OE , co oznacza blokadę portu. Wówczas wyjście nieodwracające ($D+$) i odwracające ($D-$) przy zablokowanym nadajniku zostają na potencjale masy (patrz rezystory $15k$).

Wzmacniacze odniesione do masy monitorują napięcie na liniach $D+$ i $D-$.

Obwód transmisyjny w standardzie USB

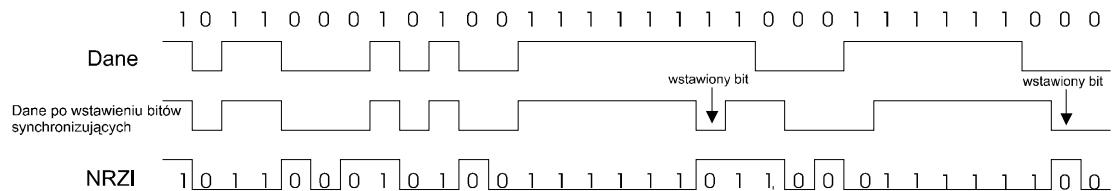


Stany magistrali USB

- Logiczne „1” w obwodzie różnicowym (D_+) - (D_-) > 200 mV
- Logiczne „0” w obwodzie różnicowym (D_+) - (D_-) < -200 mV
- Plus 9 innych stanów

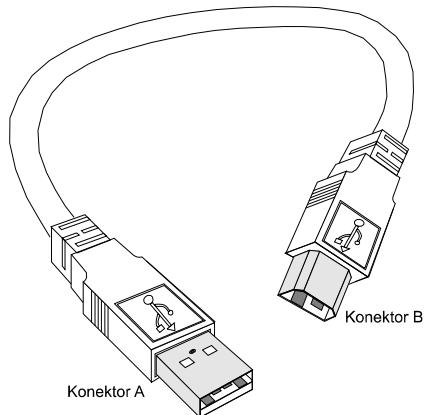
Kodowanie bitów w systemie USB

W systemie USB przyjęto kodowanie NRZI (*Non Return to Zero Inteverted*) ze wstawianiem bitu synchronizującego (*bit stuffing*). Na początku każdego bitu o wartości logicznej zero następuje zmiana sygnału, natomiast po każdym 6 kolejnych bitach o wartości jeden wstawiany jest bit o wartości zero. Umożliwia to prawidłowe poznawanie bitów w sygnale odebranym bez potrzeby przesyłania sygnału zegarowego nadajnika. Bity wstawione są usuwane po rozpoznaniu stanu bitów w odebranym sygnale.



2.3.3 Środowisko fizyczne

Podstawowy kabel do podłączenia urządzenia USB



W standardzie USB wyróżnia się dwa rodzaje złączy

- Konektor A – strona portu dolnego (hub), ”od strony urządzenia”
- Konektor B – strona portu górnego (urządzenie), ”od strony hosta”

Złącza USB posiadają 4 zaciski: po dwa dla magistrali danych i zasilania. W samym złączu kontakty zasilania są nieco wysunięte w przód przed kontakty danych, a więc po włożeniu najpierw jest podłączone zasilanie, a potem dane. **Rodzaje kabli**

| Nr kontaktu | Nazwa sygnału | Kolor przewodu w kablu |
|-------------|---------------|------------------------|
| 1 | Vcc | Czerwony |
| 2 | +DATA | Biały |
| 3 | -DATA | Zielony |
| 4 | GND | Czarny |

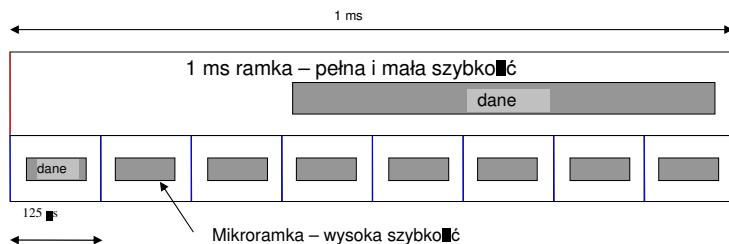
- Kabel przeznaczony dla urządzeń „low speed”
 - Nieekranowany
 - Zawiera dwie, nieskręcone pary przewodów: dla danych (28 AWG) i zasilania (20-28 AWG)

- Kabel przeznaczony dla urządzeń „full speed i high speed”
 - Zawiera ekranowaną parę skręcaną (28 AWG) dla danych
 - I nieekranowaną parę (20-28 AWG) dla zasilania
- Czas propagacji sygnału w kablu: < 30 ns przy przesyle z częstotliwością 1-16 MHz. Wymagany dla urządzeń małej i pełnej szybkości. Czas propagacji wpływa na długosc kabla:
 - 5 m dla 6ns/m
 - 3 m dla 10ns/m

2.3.4 Ramki i mikro ramki

W systemie USB dane są przekazywane:

- W ramkach o czasie trwania 1 ms dla małej i pełnej szybkości transmisji.
- W mikroramkach o czasie trwania 125 μ s dla wysokiej szybkości transmisji.



Ramka 1 ms

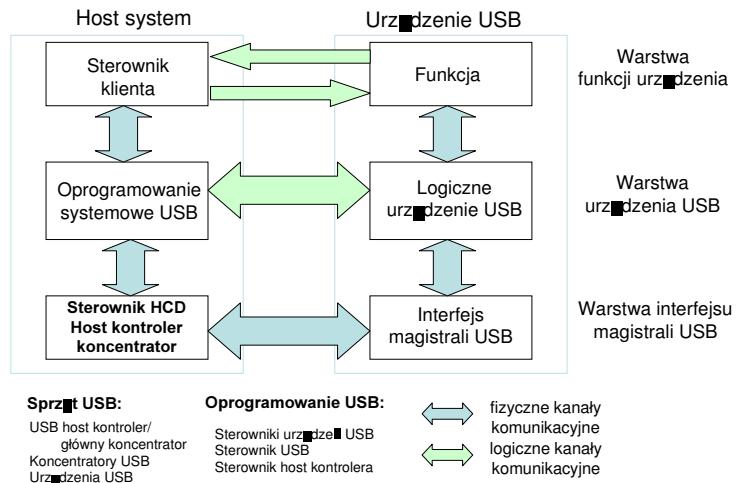
- Czas pomiędzy dwoma kolejnymi pakietami SOF (*StartOfFrame*) nazywany jest ramką. Początek ramki wyznacza pakiet SOF zawierający 11 bitów danych i 5 bitów CRC. Dane w pakiecie SOF reprezentują kolejny numer ramki. Licznik ramek przepelnia się co 2048 ms.
- Pełna szybkość transmisji
 - Maksymalny teoretyczny rozmiar ramki w bitach: $1 \text{ ms} \times 1/12 \text{ Mhz} = 12000 \text{ bitów}$
 - Powyższy rozmiar ramki w bajtach: $12000 : 8 = 1500$.
 - Pakiety kontrolne zajmują ok. 300 bajtów.
 - Praktyczna góra granica liczby transmitowanych w ramce: 1200 bajtów.
- Mała szybkość transmisji
 - Max rozmiar ramki w bitach: $1 \text{ ms} \times 1/1,5 \text{ Mhz} = 1500 \text{ bitów}$
 - Max rozmiar ramki w bajtach: $1500 : 8 = 187$

Mikroramka 125 μ s

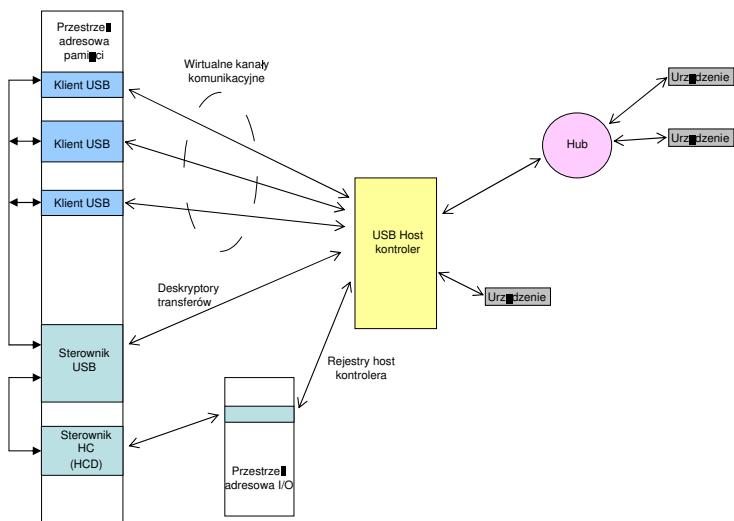
- Czas pomiędzy dwoma kolejnymi pakietami SOF wysyłanymi przez high speed huba nazywa się mikroramką.
- Mikroramka trwa 125 μ s
- Na 1 ramkę przypada 8 mikroramek
- Wysoka prędkość transmisji - szybkość transmisji w mikroramce wynosi 480 Mhz.

2.3.5 Model komunikacyjny

Warstwowy model komunikacyjny przedstawiający odpowiadające sobie elementy sprzętowo-programowe



po stronie hosta i urządzenia.



2.3.6 Transfery USB

Wyróżniono 4 typy transferów przeznaczone dla różnych urządzeń z różnymi wymaganiami komunikacyjnymi. Każdy wymaga innego kanału komunikacyjnego (*pipe*).

- **Masowy (bulk transfer)** - time delivery accuracy + quality delivery accuracy ("jak najlepiej w miarę możliwości").

Przeznaczony do komunikacji z urządzeniami, do których zapisuje się lub z których odczytuje duże ilości danych, jednak czas przekazywania danych oraz regularność ich dostarczania nie są krytyczne. Ważna jest kontrola przekazywanych danych, błędy transmisji muszą zostać wykryte i skorygowane poprzez żadanie ponownego przesłania błędnie bloku danych.

Nie ma zagwarantowanego pasma, transmisja ta jest realizowana, jeżeli inne transfery nie wykorzystają przydzielonych im zasobów.

Może być realizowany tylko z pełną lub wysoką szybkością.

Wymagane jest, aby pakiet danych miał stałą wartość (*MaxPacketSize*): 8, 16, 32 lub 64. Wszystkie kolejne pakiety danych muszą mieć tę stałą wartość z wyjątkiem ostatniego, który sygnalizuje koniec transferu. Przykłady: drukarka, skaner.

- **Przerwaniowy** (*interrupt transfer*) - quality delivery accuracy.

Do komunikacji z urządzeniami, do których zapisuje się lub z których odczytuje się niewielkie bloki danych, jednak trzeba robić to regularnie, w ustalonych odstępach czasowych.

Parametry transferu zawarte są w deskryptorze pobieranym na etapie konfiguracji urządzenia.

Ma zagwarantowane pasmo, może być wykorzystany do obsługi urządzeń low speed i full speed.

Rozmiar pakietu danych zależy od wybranej szybkości: dla low speed jest to 8, dla full speed to maksymalnie 64. Wszystkie kolejne pakiety danych muszą mieć tę stałą wartość z wyjątkiem ostatniego, który sygnalizuje koniec transferu.

Przykłady: mysz, klawiatura.

- **Izochroniczny** (*isochronous transfer*) - time delivery accuracy.

Przeznaczony do komunikacji z urządzeniami, do których zapisuje się lub z których odczytuje duże ilości danych, przy czym krytyczna jest regularność dostarczania danych.

Nie jest ważna kontrola przekazywania danych, nie jest dopuszczalne ponowne wysłanie bloku danych.

Dla komunikacji w pełnej lub wysokiej szybkości. Parametry transferu USB odczytuje z deskryptora urządzenia.

Graniczna wartość parametru *MaxPacketSize* to 1023 bajty na ramkę.

Przykład: odtwarzanie muzyki w aparaturze audio.

- **Kontrolny** (*control transfer*) - time delivery accuracy + quality delivery accuracy.

Musi być obsługiwany przez każde urządzenie USB, ponieważ jest przeznaczony do jego konfiguracji i nadzoru. W ramce zarezerwowano stosowane pasmo dla realizacji transferów kontrolnych (10% czasu trwania ramki).

Obsługuje wszystkie 3 rodzaje prędkości transferu.

Dzieli się na 3 etapy:

- Przekazanie rozkazu - 8-bitowe pole danych udostępnia informację o liczbie bajtów danych przesyłanych w etapie przekazani danych, o ile ten występuje.
- Przekazanie danych - pakiety danych mogą mieć rozmiar nie większy niż 64 bajty (parametr *MaxPacketSize*).
- Przekazanie statusu.

Podział transferów

- Transfery aperiodyczne: kontrolny i masowy
- Transfery periodyczne: przerwaniowy i izochroniczny
- Transfery strumieniowe (*stream transfer*), przekazywanie danych bajt po bajcie: przerwaniowy, masowy i izochroniczny
- Transfery komunikatowe (*message transfer*), struktura komunikatów i ich znaczenie określa standard: kontrolny

2.3.7 Zarządzenie magistralą USB

Wszystkie transfery USB są inicjowane i odbywają się pod kontrolą hosta. Host usiłuje jednocześnie realizować transfery wielu urządzeń, tak, aby jedno urządzenie nie blokowało dostępu.

Niektóre transfery mają priorytety. Transferem izochronicznym i przerwaniowym nadano 90% pasma, a kontrolnym 10%. Oznacza to, że transfer masowy nie jest gwarantowany - jeżeli pozostałe 3 rodzaje rezerwują maksymalny możliwy czas, to transfer masowy się nie odbędzie.

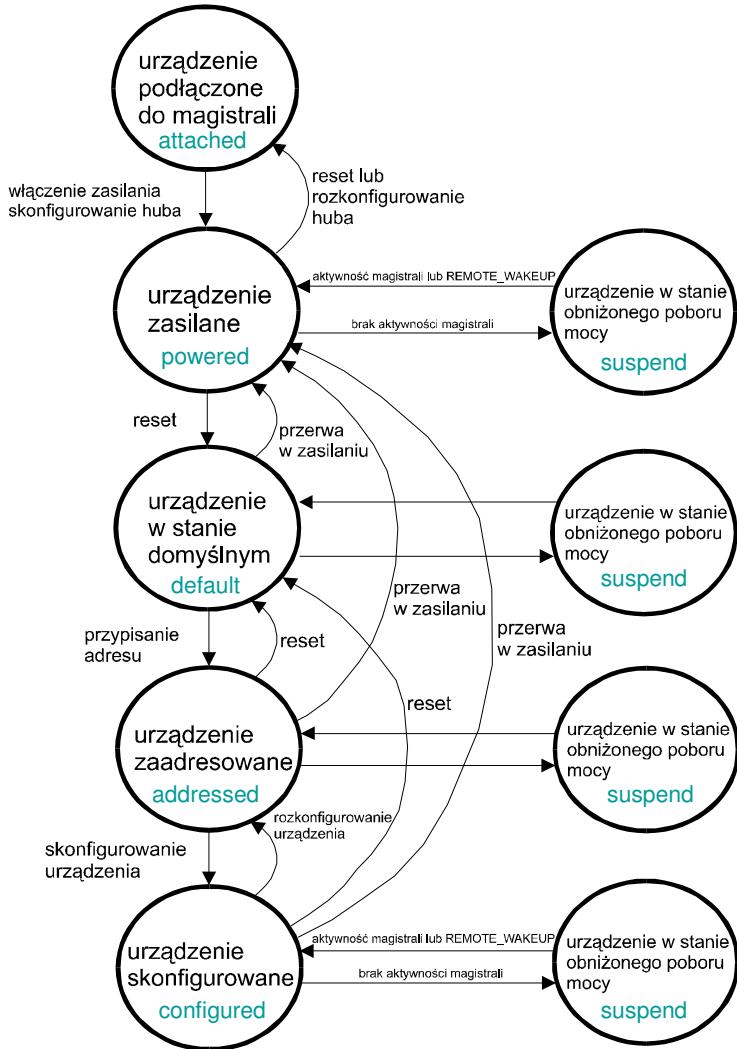
W praktyce jednak rzadko to się zdarza i 3 transfery nie wykorzystują pełnego pasma, co pozwala kontrolerowi USB na przydzielenie niewykorzystanego miejsca transferom masowym.

Kontroler dąży do jak najlepszego wykorzystania miejsca w ramce, w które może wstawić nawet wiele transakcji danego transferu masowego.

Działanie kontrolera oparte jest na zasadzie "usilnych starań" (*best efforts*), aby jak najsprawniej (w jak najkrótszym czasie) zrealizować wszystkie żądane transfery. Jednak jeżeli w ramce 1 ms brakuje miejsca na transferu masowego to musi on czekać.

2.3.8 Stany urządzenia USB

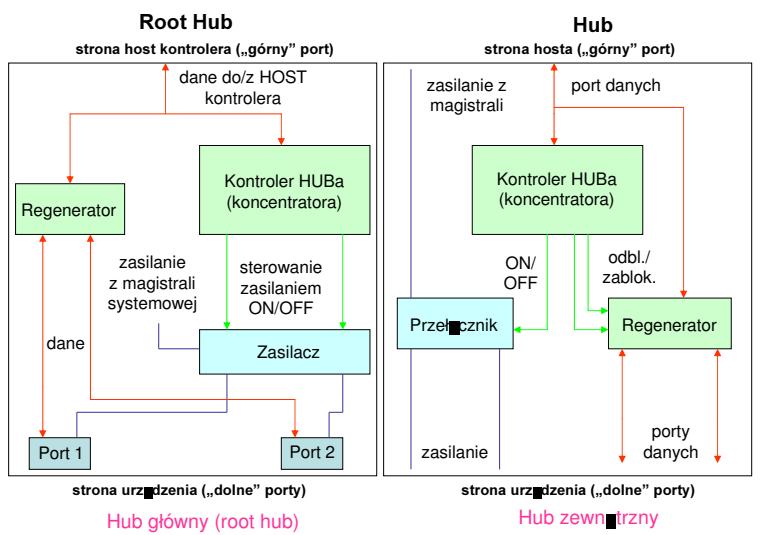
Zanim urządzenie USB zostanie skonfigurowane i udostępnione aplikacjom musi przejść przez kilka stanów przedstawionych na rysunku



2.3.9 Hub w systemie USB

Huby to urządzenia pośredniczące w komunikacji z urządzeniami końcowymi.

- Zwiększają liczbę portów dostępnych dla urządzeń oraz możliwe jest ich kaskadowe łączenie.
- Port huba od strony hosta nazywa się górnym (*upper stream port*), natomiast porty od strony urządzeń dolnym (*down stream port*).
- Hub przeważnie posiada 4 porty dolne, rzadko kiedy inną liczbę jak np. 7.
- Hub nie tylko przekazuje dane, ale pośredniczy również w dystrybucji zasilania do urządzeń.
- Huby zgodne ze standardem 1.x są zdolne do komunikacji z małą lub pełną szybkością. Obowiązuje zasada, że do urządzeń low speed nie wolno kierować ruchu full speed. Hub potrafi wykryć jakie urządzenie jest podłączone do danego portu dolnego i wie, kiedy należy go zablokować.
- Transfery low speed są poprzedzone specjalną preambułą, których zadaniem jest odblokowanie portów dolnych do których podłączone są urządzenia low speed.



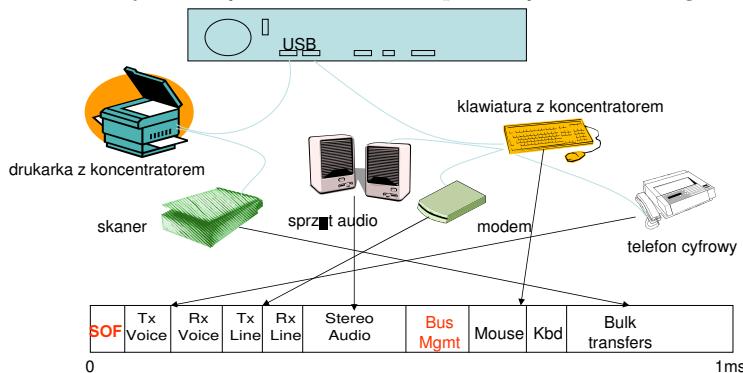
2.4 Protokół komunikacyjny

Transakcje USB zbudowane są z pakietów, przy czym typowa transakcja zawiera:

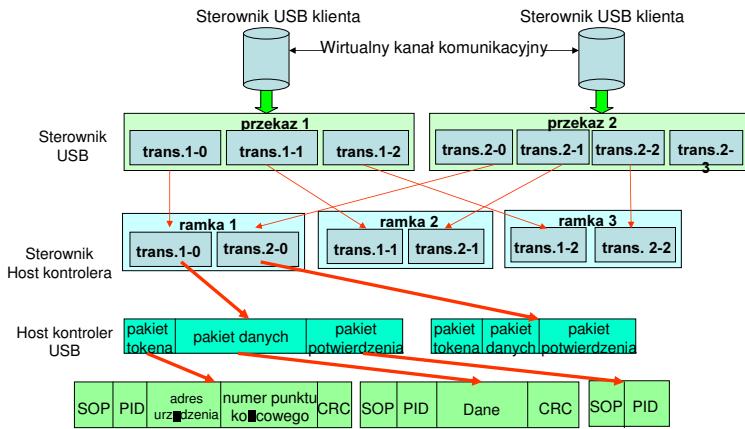
- Pakiet tokena (*token packet*) - jest elementem kontrolnym transakcji i informuje o jej rodzaju: kontrolna lub przeznaczona do zapisywania danych.
- Pakiet danych (*data packet*) - pole przeznaczone dla danych zapisywanych do urządzenia lub odczytywanych z urządzenia.
- Pakiet potwierdzenia (*handshake packet*) - informuje jednostkę nadającą o odebraniu danych lub instrukcji sterującej przez odbiorcę.

2.4.1 Właściwości

- Komunikacja z urządzeniami USB za pomocą ramek o długości maksymalnie 1 ms.

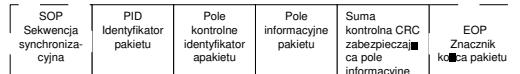


- Pakietowa struktura komunikatów



Struktura bloków komunikacyjnych w USB

2.4.2 Format i typy pakietów USB



Format pakietu

| Typ pakietu | PID |
|-----------------|-------|
| Token In | 1001b |
| Token Out | 0001b |
| Token Setup | 1101b |
| Token SOF | 0101b |
| Data 0 | 0011b |
| Data 1 | 1011b |
| Handshake ACK | 0010b |
| Handshake NAK | 1010b |
| Handshake STALL | 1110b |
| Preamble | 1100b |

Kody PID (USB 1.1)

Pakiety szczególne:



Pakiet preambuły poprzedzający pakiet przesyłany z małą szybkością

2.4.3 Reakcja na błędy

Wykrywanie błędów i kontrola transmisji

- Kontrola poprawności pakietów (zabezpieczenie pola PID sumą CRC)
 - W przypadku pakietu Token - 5-bitowa suma
 - W przypadku pakietu Data - 16-bitowa suma

Wykrycie błędu powoduje odrzucenie pakietu.

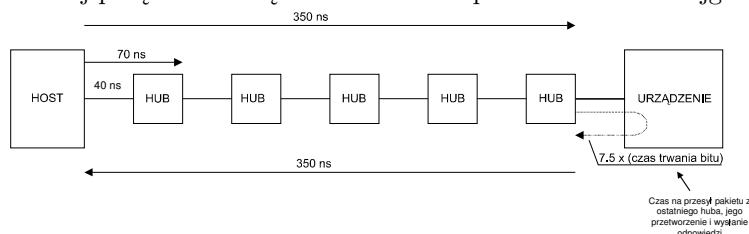
- Reakcja na fałszywy znacznik końca pakietu (*false EOP*)
- Ograniczenie czasowe oczekiwania na odpowiedź
- Przełączanie kolejnych pakietów danych (mechanizm *data toggle*)
- Wykrywanie transakcji występujących po zakończeniu ramki (tzw. „paplanie” – *babble*)
- Wykrywanie braku aktywności magistrali (LOA – *Loss Of Activity*)

Transfery kontrolny, przerwaniowy i masowy wysyłają pakiet ponownie, jeżeli jest on błędny, nie informując odbiorcy o tym. Transfer izochroniczny nie.

2.4.4 Czas obiegu (round trip delay)

Ograniczenie czasowe oczekiwania na odpowiedź. To podstawowy mechanizm informowania nadawcy o niepowodzeniu w poprawnym przekazaniu pakietu. Ograniczenie czasowe nie może być mniejsze od $16^*(\text{czas trwania bitu})$, ani większe od $18^*(\text{czas trwania bitu})$.

Poniżej połączenie urządzenia z hostem przez 5 hubów – najgorszy przypadek.



Jest on podstawa do rozważań na temat ograniczeń tego czasu.

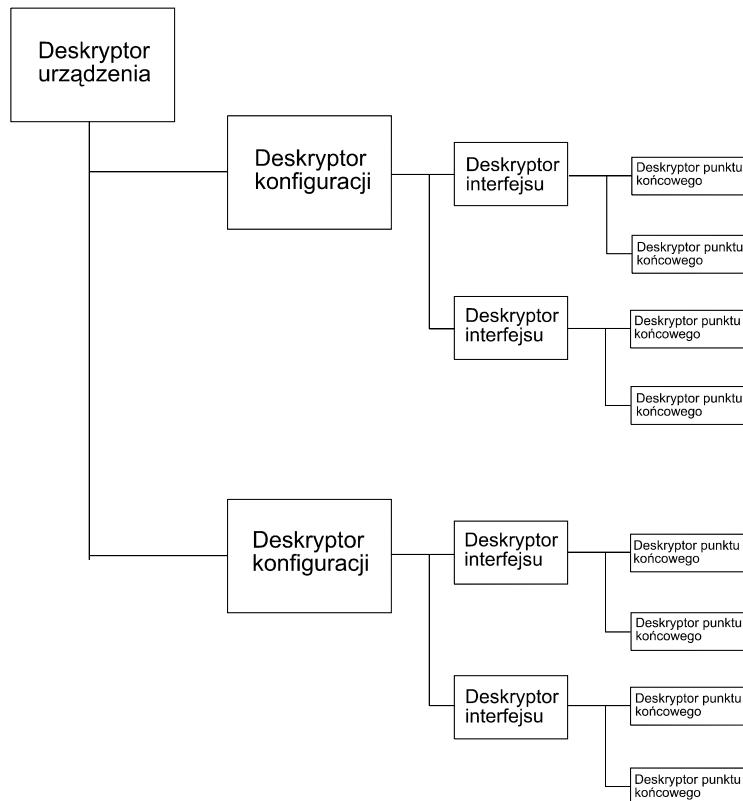
- Round trip delay: $2 \times 350 \text{ ns} = 700 \text{ ns}$
- Czas trwania 1 bitu full speed: $1/12 \text{ MHz} = 83 \text{ ns}$
- Round trip delay w bitach dla full speed: $700 \text{ ns}/83 \text{ ns} = 8,5 \text{ bitu}$
- Timeout oczekiwania na odpowiedź: $7,5 \text{ bitu} (\text{specyfikacja}) + 8,5 \text{ bitu} = 16 \text{ bitów}$

2.5 Deskryptory w urządzeniach USB

W każdym urządzeniu USB znajduje się pełna informacja o sposobie komunikacji z urządzeniem, udostępniana podczas procesu enumeracji.

Informacja ta jest przechowywana w deskryptorach - tablicach o ścisłe określonej strukturze.

2.5.1 Hierarchiczna struktura deskryptorów w urządzeniu USB



2.5.2 Rodzaje deskryptorów

- Deskryptor urządzenia - m.in. liczba konfiguracji dostępnych w urządzeniu.
- Deskryptor konfiguracji - opisuje każdą konfigurację z osobna. Zawiera informację o liczbie interfejsów przypisanych danej konfiguracji.
- Deskryptor interfejsu - posiada go każdy interfejs, który określa m.in. liczbę punktów końcowych z nim związanych.
- Deskryptor punktu końcowego - charakteryzuje każdy punkt końcowy.
- Deskryptor łańcuchowy - zawiera kod określający język tekstu lub sam właściwy tekst.

2.6 Wykrywanie i konfiguracja urządzeń

Cechą USB jest automatyczne wykrywanie urządzeń połączenia zasilania w systemie lub połączenia urządzenia do systemu. Procedura enumeracji zajmuje się rozpoznaniem urządzenia, sprawdzeniem czy komunikacja jest możliwa, przydzieleniem adresu, konfiguracją i instalacją sterownika.

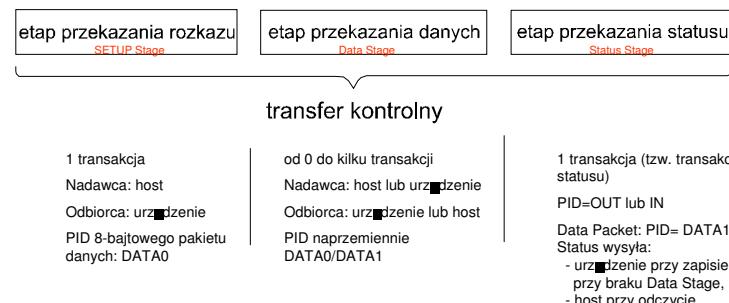
2.6.1 Procedura enumeracji urządzenia

1. Automatyczne wykrycie urządzenia - powoduje ustawienie bitu w rejestrze statusu huba, który odpowiada temu portowi.
2. Odblokowanie portu i generacja RESETU - przejście do stanu domyślnego.
3. Odczyt deskryptora urządzenia - skierowanie stosownego rozkazu do punktu końcowego 0 o adresie 0. Jest tylko jedno takie urządzenie w systemie - te, które właśnie podlega procesowi enumeracji.
4. Przypisanie adresu - unikalny adres dla urządzenia
5. Odczyt pozostałych deskryptorów - jeżeli istnieją
6. Konfiguracja urządzenia
7. Instalacja sterownika klienta
8. Urządzenie jest dostępne dla aplikacji

2.7 Kontrola urządzenia – transfer kontrolny

Transfer kontrolny służy do sterowania urządzeniem. Urządzenie USB otwiera kontrolny kanał komunikacyjny pomiędzy hostem, a punktem końcowym 0 w urządzeniu.

2.7.1 Etapy transferu kontrolnego



1. Przekazanie rozkazu (*Setup Stage*) - jak w każdym pakiecie Token, następne pola zawierają adres urządzenia oraz punktu końcowego. Kolejnym elementem jest 8-bajtowy pakiet danych typu Data 0, który specyfikuje rozkaz.

- Przekazanie danych (*Data Stage*) - jeżeli 8 bajtów z powyższego nie wystarcza, wtedy jest ten etap. Liczbę bajtów do przekazania określa słowo Length w polu transakcji Setup. Jeżeli ta liczba nie przekracza wartości *MaxPacketSize* to do przekazania danych wystarcza jedna transakcja, w przeciwnym razie odpowiednio dzieli się na więcej transakcji (sufit(Length / MPC)).
- Przekazanie statusu (*Status Stage*) - potwierdzenie wykonania transferu lub poinformowanie o niepowodzeniu w realizacji. Normalnie występuje do zakończenia transakcji, ale host może rozpoczęć ją wcześniej, w Data Stage (wtedy DS jest przerywane).

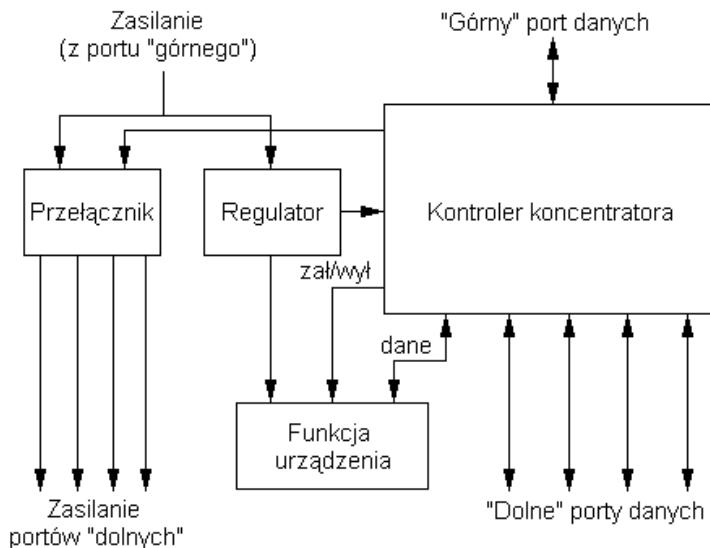
2.8 Hub USB

2.8.1 Co to jest?

Standard USB definiuje oddzielną klasę urządzeń: klasa HUB. Jest to tak ważna część tego systemu, że została zdefiniowana w standardzie USB, a nie we własnej oddzielnej specyfikacji.

Typowy hub posiada jeden deskryptor.

2.8.2 HUB zasilany z magistrali



2.8.3 Proces konfiguracyjny huba

- Odczyt standardowych deskryptorów urządzenia
- Przypisanie hubowi adresu
- Załączenie zasilania na porty, co jest niezbędne do detekcji podłączonych do portu urządzeń
- Odczyt punktu końcowego, "zmiany w hubie", w celu wykrycia urządzeń podłączonych do portów
- Odblokowanie portu w celu umożliwienia dostępu do urządzenia

2.8.4 Punkt końcowy huba "zmiany w hubie"

- Cyklicznie odczytywany, umożliwia monitorowanie zmian na portach dolnych huba, co z kolei umożliwia wykrywanie dołączania i usuwania urządzeń.
- Odpytywanie odbywa się przez wykorzystanie kanału przerwaniowego.
- Hub change point*: informuje o wystąpieniu:

- Zmiany w zasilaniu lub nadmiernym obciążeniu prądowym huba
 - Zmiany na jednym lub kilku portach dolnych spowodowanej dołączeniem lub odłączeniem urządzenia.
- Odczyt *Hub change point* zwraca bajt statusowy huba. Jeżeli hub posiada więcej niż 7 portów dolnych - zwracane są dwa bajty.
 - Jeżeli wszystkie bity w punkcie końcowym są ustawione na 0 (brak zmian), hub nie odsyła bajtu statusowego.
 - Nie mylić z punktem końcowym 0 przez który wykonywana jest konfiguracja i kontrola.

2.8.5 Działanie

| Numer bitu | Znaczenie dla wartości 1 |
|------------|---|
| 0 | Wykrycie zmiany zasilania lub przełączenia huba |
| 1 | Wykrycie zmiany na porcie dolnym 1 |
| 2 | Wykrycie zmiany na porcie dolnym 2 |
| 3 | Wykrycie zmiany na porcie dolnym 3 |
| 4 | Wykrycie zmiany na porcie dolnym 4 |
| 5 | Wykrycie zmiany na porcie dolnym 5 |
| 6 | Wykrycie zmiany na porcie dolnym 6 |
| 7 | Wykrycie zmiany na porcie dolnym 7 |

Bajt statusowy huba

Jedeli w bajcie statusowym któryś z bitów o numerach od 1 do 7 jest ustawiony na 1, to host rozkazem **GET_PORT_STATUS** z argumentem **wskazujacym numer portu** odczytuje dwa 16 bitowe rejestry stanu:
 • status portu (*port status*),
 • zmiana statusu portu (*port status change*)

| Numer bitu | Znaczenie |
|------------|--|
| 0 | Podłączenie urządzenia do portu: 0 – urządzenie nie jest podłączone 1 – urządzenie jest podłączone |
| 1 | Odblokowanie lub zablokowanie portu: 0 – port zablokowany 1 – port odblokowany |
| 2 | Urządzenie w stanie suspend 0 – stan normalny 1 – stan suspend |
| 3 | Signalizacja przełączania portu: 0 – port pracuje normalnie, 1 – przełączanie portowe portu |
| 4 | Reset urządzenia: 0 – praca normalna, 1 – inicjacja resetu |
| 7-5 | Zarezerwowane, ustawione na 0 |
| 8 | Znaczenie portu: 0 – odłączone zasilanie 1 – port zasilany |
| 9 | Szybkość urządzenia: 0 – urządzenie full speed 1 – urządzenie low speed |
| 15-10 | Bity zarezerwowane, ustawione na 0 |

Status portu

Jedeli w bajcie statusowym bit 0=1, to host rozkazem **GET_HUB_STATUS** odczytuje dwa 16 bitowe rejestry stanu:
 • status huba (*hub status*),
 • zmiana statusu huba (*hub status change*)

| Numer bitu | Znaczenie |
|------------|--|
| 0 | Stan zasilania lokalnego: 0 – zasilanie lokalne jest poprawne 1 – zasilanie lokalnego napięcia zasilania |
| 1 | Signalizacja przełączania prądowego: 0 – działanie huba normalne 1 – przełączanie prądowe huba |
| 15 – 2 | Bity zarezerwowane, ustawione na 0 |

Status huba

| Numer bitu | Znaczenie |
|------------|--|
| 0 | Zmiana zasilania lokalnego: 0 – nie wystąpiła zmiana zasilania lokalnego 1 – wystąpiła zmiana zasilania lokalnego |
| 1 | Zmiana przełączania prądowego: 0 – układ kontroli nie sygnalizował wystąpienia przełączania 1 – układ kontroli sygnalizował wystąpienie przełączania |
| 15 – 2 | Bity zarezerwowane, ustawione na 0 |

Rejestr zmian statusu huba

2.9 Zasilanie urządzeń w systemie USB

USB jest również systemem dystrybucji i zarządzania zasilaniem urządzeń.

2.9.1 Możliwe zasilania urządzeń USB

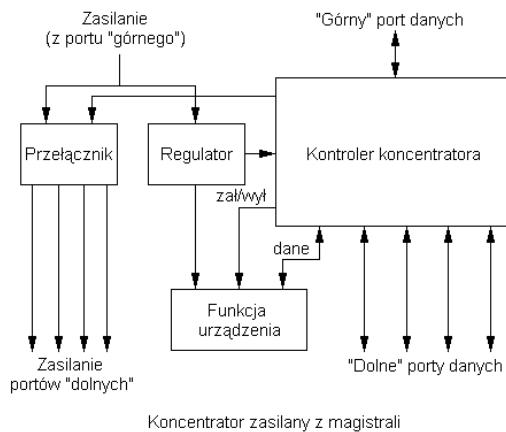
- Urządzenie zasilane z magistrali
- Urządzenie zasilane z własnego źródła
- Urządzenie zasilane częściowo z magistrali i własnego źródła

2.9.2 Zasilanie hubów i pozostałych urządzeń



Dopuszczalne spadki napięcia zasilania.

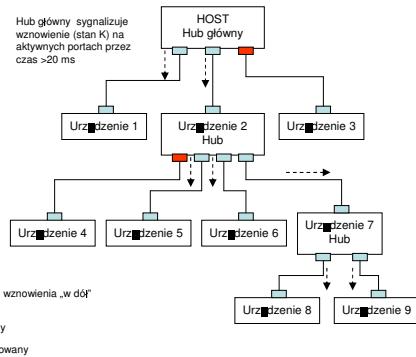
2.9.3 Urządzenie zasilane z magistrali



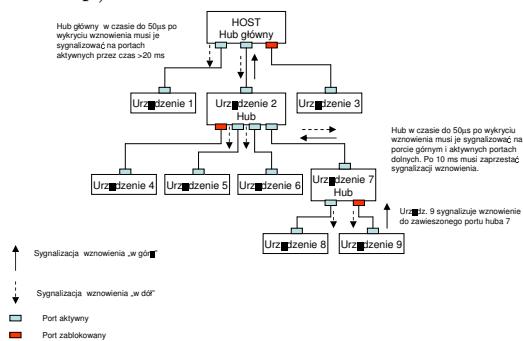
Dla portu zasilanego z magistrali USB i podłączonego do portu o obciążalności 500 mA, 100 mA jest zarezerwowane na zasilanie jego układu. Oznacza to, że 400 mA pozostaje dla jego portów dolnych. Ponieważ każdy port musi mieć zapewnione minimum 100 mA, taki hub może mieć nie więcej niż 4 porty dolne (stąd pewnie przyjęty standard).

2.9.4 Zarządzanie zasilaniem

- Wprowadzenie w **stan zawieszenia** pracy systemu – *suspend*. Pozostając w zawieszeniu urządzenie zachowuje swoją konfigurację, co pozwala jego wznowienie (*resume*) do normalnego działania.
 - **Globalne (global suspend)** - rozkaz SetPortFeature (PORT_SUSPEND) adresowany do huba głównego, zawiesza cały system
 - **Częściowe (selective suspend)** - rozkaz SetPortFeature (PORT_SUSPEND) adresowany do huba zewnętrznego, w którym wybrany port (lub część systemu) ma zostać zawieszony.
 - Zawieszone porty nie propagują ruchu „w dół” oraz nie przekazują „w górę” żadnych sygnałów od urządzeń do nich podłączonych.
 - Urządzenia w stanie zawieszenia zachowują swój stan, co umożliwia wznowienie ich pracy bez ponownej konfiguracji.
 - Hub w stanie zawieszenia dodatkowo blokuje wszystkie nadajniki, zatrzymuje wewnętrzne zegary i zachowuje stan wszystkich portów dolnych.
 - Urządzenie w stanie zawieszenie pobiera prąd $< 500[\mu A]$
 - Urządzenie automatycznie przechodzi do stanu *SUSPEND* po wykryciu braku aktywności magistrali przez 3 ms. (USB wysyła ramki co 1 ms)
 - * Urządzenia pełnej szybkości wykorzystują pakiety SOF
 - * Urządzenia małej szybkości wykorzystują generację przejścia ze stanu jałowego do stanu K w odstępach czasu nie większych niż 3 ms.
- **Wznowienie** pracy systemu - *resume*
 - Globalne
 - Wake-up
 - Wznowienie pracy urządzenia można nastąpić
 - * Z inicjatywy kontrolera, jako wznowienie po zawieszeniu globalnym lub częściowym



- * Z inicjatywy urządzenia, po wystąpieniu zdarzenia wymagającego obsługi („budzenie” - *Wakeup*)



- Wznowienie po zawieszeniu globalnym - rozpoczyna hub główny wykonując rozkaz wznowienia pracy systemu. Oznacza gotowość wszystkich hubów i urządzeń do wykonywania transakcji.
- Wznowienie po częściowym zawieszeniu - może wykonać:
 - * Host rozkazem *ClearPortFeature* (PORT SUSPEND) adresowanym do huba w którym znajduje się zawieszony port.
 - * Urządzenie podłączone do zawieszonego portu poprzez *Wakeup*

2.10 Rozwiązania host kontrolerów

2.10.1 Co to jest?

Host kontrolera (*Host Controller Driver*) przetwarza poszczególne IRP na transakcje, tworzy listę transakcji w ramach ramki, przygotowuje scenariusz wykonania każdej transakcji i kontroluje jej wykonanie.

2.10.2 Rozwiązania

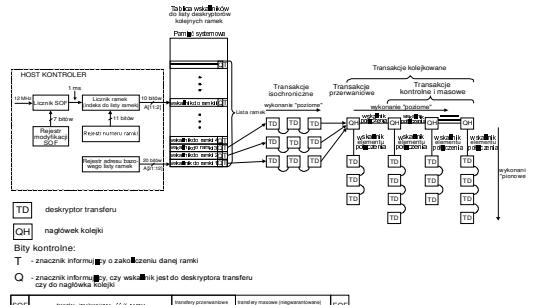
Opracowano dwa rozwiązania host kontrolera:

- Uniwersalny host kontroler - *Universal Host Controller* (Intel)
- Otwarty host kontroler – *Open Host Controller* (Compaq, Microsoft, National Semiconductor)

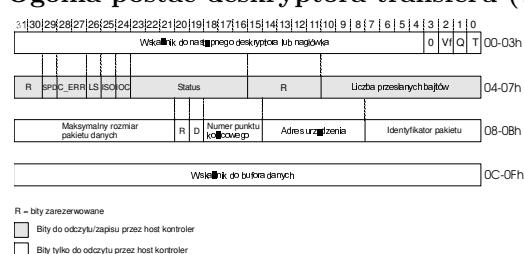
Są do siebie bardzo podobne, różnią się tylko sposobem obsługi oraz współpracą z hubem głównym.

2.10.3 Uniwersalny host kontroler (UHC)

Zasada działania kontrolera UHC



Ogólna postać deskryptora transferu (TD)

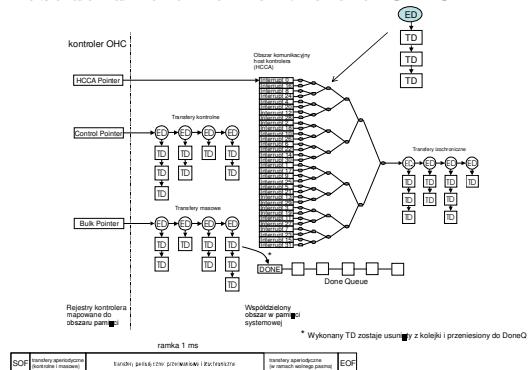


Ogólna postać nagłówka kolejki (QH)

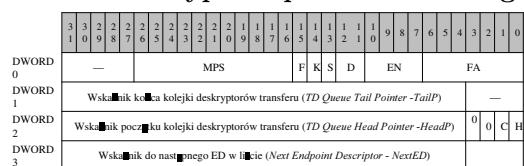


2.10.4 Otwarty host kontroler (OHC)

Zasada działania kontrolera OHC



Format deskryptora punktu końcowego (ED)



Ogólna postać deskryptora transferu (TD)

| | | | | | | | | | | | | | | | | | | | | | | |
|---------|----|----|---|----|----|---|---|---|---|---|----|---|----|---|----|---|---|---|---|---|---|---|
| | 3 | 3 | 2 | 2 | 7 | 2 | 5 | 2 | 3 | 2 | 21 | 2 | 19 | 8 | 17 | 4 | 5 | 4 | 3 | 2 | 1 | 0 |
| DWORD 0 | CC | EC | T | DI | DP | R | — | | | | | | | | | | | | | | | |
| DWORD 1 | | | | | | | | | | | | | | | | | | | | | | |
| DWORD 2 | | | | | | | | | | | | | | | | | | | | | | |
| DWORD 3 | | | | | | | | | | | | | | | | | | | | | | |

Wskazówka do bufora (Current Buffer Pointer - CBP)
Wskazówka do następnego TD (Next TD)
Wskazówka do końca bufora (Buffer End - BE)

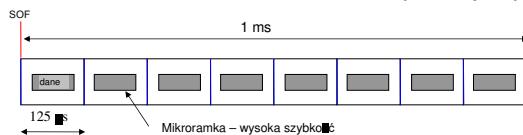
2.11 USB 2.0 - rozszerzenie standardu

2.11.1 Ważniejsze elementy wprowadzone w USB 2.0

- Wysoka szybkość transmisji (high speed) - 480 Mb/s
- Protokół PING-NYET
- Transakcja SPLIT
- Komunikacja z szerokopasmowym punktem izochronicznym
- Nowe typy pakietów

2.11.2 Wysoka szybkość transmisji

Podział ramki na 8 mikroramek wysokiej szybkości



- Mikroraamka trwa 125 μ s
- Na 1 ramkę przypada 8 mikroramek
- Wysoka szybkość transmisji - w mikroraamce wynosi 480 Mhz.

2.11.3 Protokół PING-NYET

Potwierdzenie NYET (*NOT YET*) dla urządzeń *high speed*.

Problem: przy zapisie do urządzenia, jeżeli nie jest ono „gotowe na dane” potwierdzenie negatywne przychodzi dopiero po pakiecie danych – strata czasu.

Rozwiążanie:

TOKEN PING – zapytanie urządzenia, czy jest gotowe do przyjęcia danych.
Możliwe odpowiedzi i reakcja hosta:

- ACK – wykonanie transakcji OUT
- NYRT – host kontynuuje wysyłanie zapyta PING

Korzyści: lepsze wykorzystanie magistrali (PING jest krótki).

2.11.4 Transakcja SPLIT

Problem: Host i hub wysokiej szybkości komunikują się z urządzeniem małej lub pełnej szybkości. Szybkie przesyłanie danych pomiędzy hostem i hubem oraz wolne pomiędzy hubem i urządzeniem – konieczność buforowania danych w hubie.

Rozwiążanie:

Transakcja dzielona, złożona z dwóch części:

- SSPLIT (*Start Split* - rozpoczęcie transakcji dzielonej)
- CSPLIT (*Complete Split* – zakończenie transakcji dzielonej).
- Pomiędzy tymi dwoma elementami transakcji dzielonej mogą być wykonywane inne transakcje z wysoką szybkością.

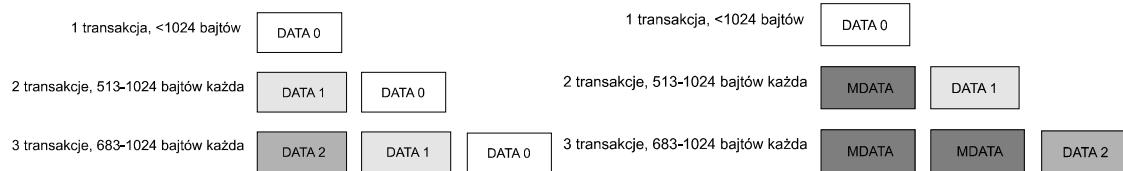
2.11.5 Komunikacja z szerokopasmowym punktem izochronicznym

Komunikacja z „normalnym” izochronicznym punktem końcowym zakłada jedną transakcję na ramę lub mikroramkę.

W przypadku punktów szerokopasmowych, obsługiwanych tylko przez kanały *high speed*, istnieje możliwość przekazania w jednej mikroramce większej ilości danych wykonując bezpośrednio po sobie od jednej do trzech transakcji.

Dane w takiej sekwencji transakcji muszą być oznaczone, przy czym nie wystarczą już „znaczniki” Data 0 i Data 1, dlatego wprowadzono dwa kolejne typy pakietów danych: **Data 2** i **MData**.

Pakiet **Data 2** wykorzystywany jest przy odczycie danych z urządzenia, natomiast **MData** i **Data 2** przy zapisie do urządzenia.



Dozwolone sekwencje pakietów danych
przy odczycie szerokopasmowym

Dozwolone sekwencje pakietów danych
przy zapisie szerokopasmowym

2.11.6 Specjalne pakiety TOKEN wprowadzone w USB 2.0

Kody pakietów specjalnych zgodnie z USB 2.0

| Typ pakietu | PID |
|-----------------------|-------|
| Pakiety danych | |
| Data 2 | 0111b |
| MData | 1111b |
| Pakiety potwierdzenia | |
| NYET | 0110b |
| Pakiety specjalne | |
| PRE | 1100b |
| ERR | 1100b |
| SPLIT | 1000b |
| PING | 0100b |
| Reserved | 0000b |

3 IEEE-488 and SCPI standards

Standard IEEE-488, ogłoszony w 1973 roku, określa organizację systemów ATE (*Automatic Test Equipment*), przeznaczonych do akwizycji danych i sterowania urządzeń w obiektach skupionych lub rozproszonych. Te mają zastosowanie w testowaniu wyborów, pomiarach laboratoryjnych, sterowaniu procesami przemysłowymi itp.

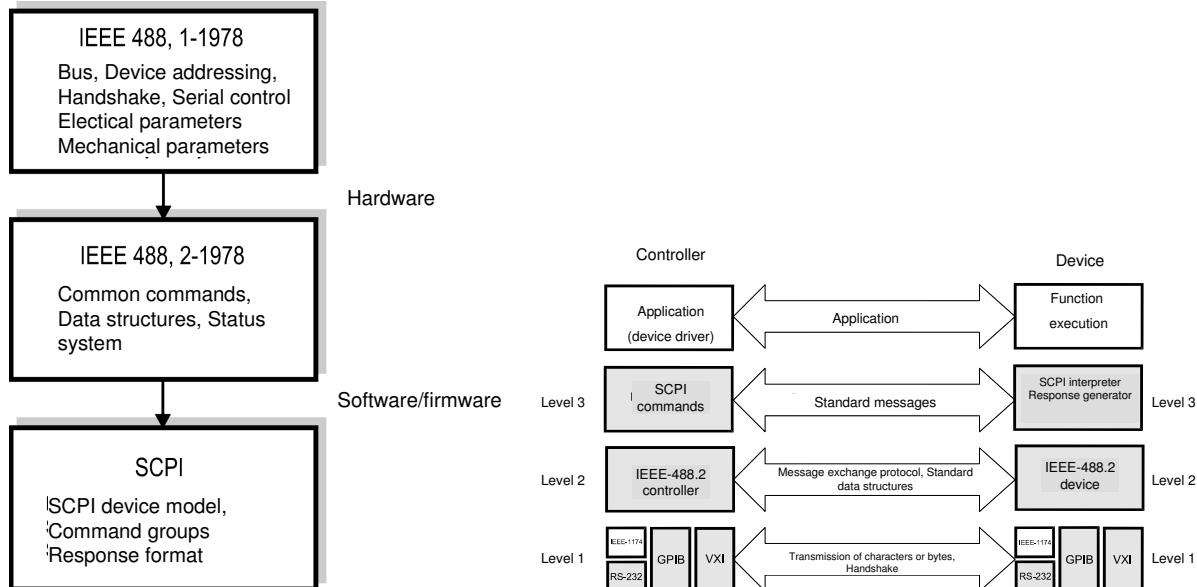
Urządzenia systemu ATE wyposażone są w układ komunikacyjny (interfejs), który umożliwia wymianę danych za pomocą łącza interfejsowego. Pracą systemu zarządza kontroler, który jest nadzorowany w stosunku do pozostałych urządzeń, które pełnią rolę przyrządów wykonawczych.

Standard IEEE-488 znany jest również jako GPIB (*General Purpose Interface Bus*). Określa on dokładnie właściwości elektryczne i mechaniczne interfejsu, jego protokoły i funkcje, jednak nie normuje przesyłanych treści. Tym zajmuje się uzupełnienie standardu w postaci IEEE-488.2 (1973) oraz SCPI (1900).

Protokół systemu ATE można podzielić na 3 poziomy:

- Poziom 1, złożony z
 - Warstwa fizyczna (GPIB), określa łącze transmisyjne i sposób przesyłania jednostek
 - Warstwa łącza danych, dostarcza reguły dostępu do łącza, sposób adresacji itp.
- Poziom 2, IEEE-488.2 kontroler < – > IEEE-488.2 urządzenie
Standard IEEE-488.2 określa protokół wymiany komunikatów, syntaktykę komunikatów i strukturę danych.
- Poziom 3, rozkazy SCPI < – > Interpretacja rozkazów, generacja odpowiedzi
SCPI normuje treść komunikatów przesyłanych pomiędzy kontrolerem a urządzeniem.
- Poziom 4, Aplikacja < – > Wykonanie funkcji urządzenia (ale to już nas raczej nie interesuje).

Ilustracja rozwoju standardu IEEE-488 oraz model komunikacyjny systemu ATE



3.1 IEEE-488 (GPIB)

Standard GPIB ogłoszono w 1979 roku, jest definicją warstwy fizycznej oraz łącza danych dla systemów pomiarowo-kontrolnych, przeznaczonych do:

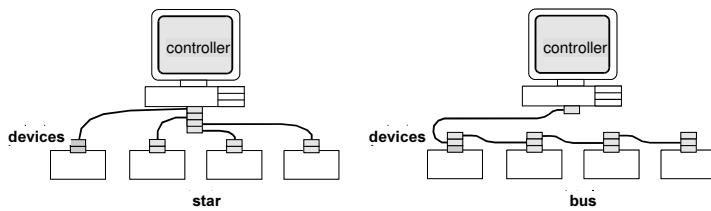
- Testowania wyborów
- Pomiarów laboratoryjnych i przemysłowych

3.1.1 Interface profile

Standard ten określa budowę i działanie części komunikacyjnej systemu opartego na magistrali, do której można:

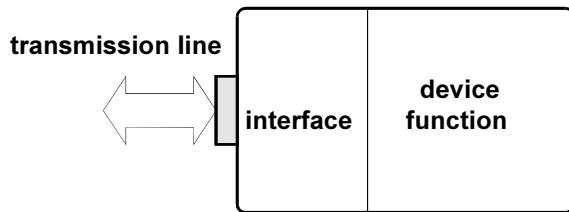
- Podłączyć do 15 urządzeń,
- Rozłożonych na niewielkim obszarze (zasięg do 20 m),
- Maksymalna odległość między kolejnymi urządzeniami to 2 m,
- Wymieniających między sobą dane rzędu 300 do 500 kB / s

Komunikacją zarządza kontroler, który konfiguruje system do komunikacji, obsługuje zgłoszenia żądania obsługi i wykonuje podstawowe operacje. **Dwie podstawowe konfiguracje systemu GPIB**



Podział urządzenia systemowego na część interfejsową i urządzenie właściwe

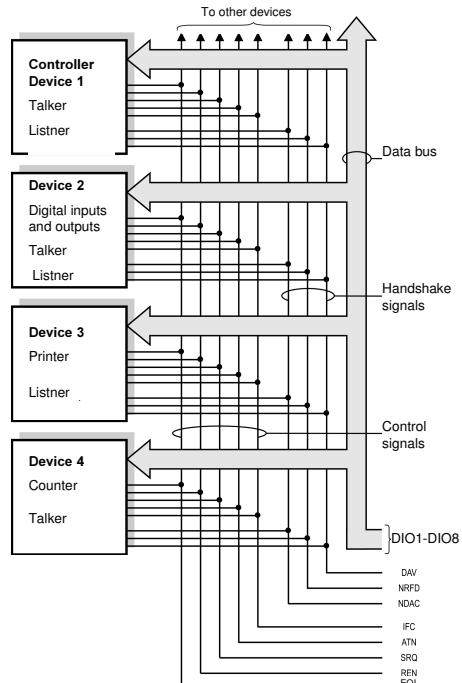
Można tak podzielić każde urządzenie wykonawcze. Część interfejsowa to zdolność urządzenia do wymiany komunikatów z innymi elementami systemu, a urządzenie właściwe wykonuje podstawowe funkcje urządzenia.



3.1.2 Magistrala

Właściwości

- Magistrala zawiera 16 linii
 - 8 dla transmisji danych
 - 8 dla sygnałów sterujących
- Jest to łącze równoległe, umożliwia przesłanie wszystkich bitów bajtu
- DIO1-DIO8 – dwukierunkowa magistrala danych
- 5 sygnałów kontrolnych:
 - ATN (*Attention* - wyjście), określa tryb pracy magistrali danych
 - * ATN=0 - transmisja bajtów danych
 - * ATN=1 - transmisja instrukcji sterujących (adresów lub rozkazów)
 - REN (*Remote Enable* - wyjście), 1 oznacza, że system jest zarządzany zdalnie przez kontroler
 - IFC (*Interface clear* - wyjście), 1 powoduje wyzerowanie części interfejsowych urządzeń
 - SRQ (*Service request* - wejście), 1 oznacza, że w systemie jest co najmniej jedno urządzenie żądające obsługi
 - EOI (*End Of Identity* - wyjście or wejście), zależny od stanu linii ATN
 - * EOI=1 razem z ATN=0 (EOI=1 ^ ATN=0) - realizacja znacznika końca bloku danych
 - * EOI=1 razem z ATN=1 (EOI=1 ^ ATN=1) – kontrola równoległa
- 3 sygnały sterujące (*Byte handshake signals*)



Rysunek 1: Reprezentacja magistrali

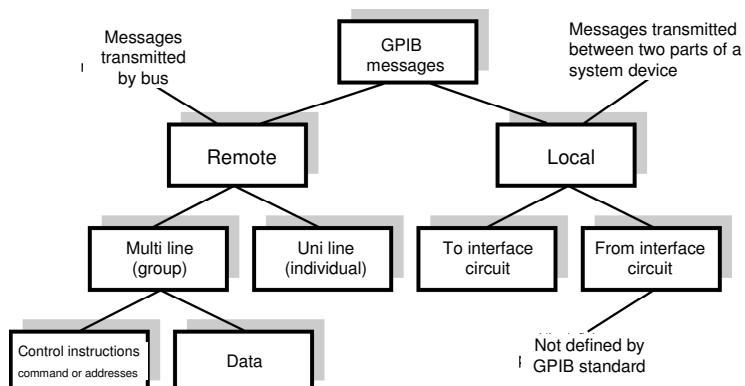
- NRFD (*Not Ready For Data* – wyjście or wejście) - jest co najmniej jeden odbiorce niegotowy do przyjęcia danych
- DAV (*Data Valid* – wyjście or wejście) - bajt znajdujący się na magistrali jest ważny i może zostać odczytany
- NDAC (*Not Data Accepted* – wyjście or wejście) - przyjęcia bajtu na magistrali nie zostało potwierdzone przez wszystkich odbiorców

3.1.3 Funkcje interfejsowe

Każda funkcja reprezentuje pewną określoną zdolność urządzenia jako jednostki systemu GPIB. Ciekawostka: każda funkcja implementowana jest tylko w tych urządzeniach, które jej potrzebują.

- SH (*Source Handshake*) - inicjator transmisji, dla urządzeń mogących nadawać dane
- AH (*Acceptor Handshake*) - akceptor transmisji, dla urządzeń mogących odbierać dane
- T (*Talker*) - nadawca, umożliwienie wprowadzenia bajtu na magistralę danych
- L (*Listener*) - odbiorca, umożliwienie odebranie bajtu z magistrali danych
- SR (*Service Request*) - zgłoszenie żądania obsługi
- DC (*Device Clear*) - zerowanie urządzenia, umożliwienie zdalnego zerowania (reset)
- DT (*Device Trigger*) - wyzwalanie urządzenia, umożliwienie zdalnego wyzwolenia operacji (pomiaru)
- RL (*Remote/Local*) - zdalne / lokalne, umożliwienie przełączania trybu urządzenia
- PP (*Parallel Poll*) - kontrola równoległa
- C (*Controller*) - kontroler, umożliwienie zdalnego zarządzania systemem GBIB i przekazywanie zarządzania innym kontrolerom - w systemie może być wiele kontrolerów, jednak tylko jeden może być aktywny, reszta musi być uśpiona.

3.1.4 Podział komunikatów



komunikaty dzielimy na zdalne(przesyłane po magistrali) i lokalne (wymieniane między częścią interfejsową, a urządzeniem właściwym).

Komunikaty zdalne dzielimy na:

- wieloliniowe: przenoszone przez magistrale danych. Tutaj wyróżniamy:
 - instrukcje sterujące (adresy lub rozkazy)
 - dane
- jednoliniowe: przenoszone przez wydzieloną linię sterującą

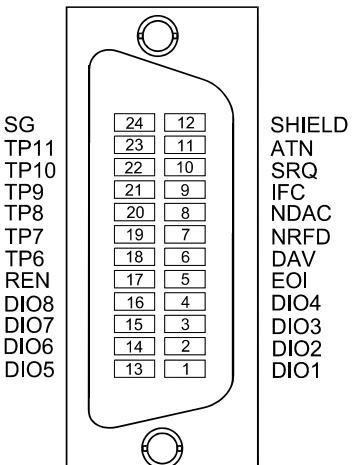
Podział funkcjonalny komunikatów:

- UC (*Universal Commands*) - rozkazy uniwersalne, przeznaczone dla wszystkich urządzeń w systemie
- AC (*Addressed Commands*) - rozkazy adresowane, przeznaczona tylko na konkretnych urządzeń
- AD (*Addresses*) - adresy, komunikaty umożliwiające wyznaczenie urządzenia nadającego i urządzeń odbierających
- HS (*Handshake*) - komunikaty synchronizacji, kontrolujące transfer bajtu przez magistralę danych
- DD (*Device Depended (data)*) - komunikaty zależne od urządzenia, ich znaczenie jest zależne od przyrządu
- ST (*Status*) - komunikaty statusowe, udostępniają informacje o stanie urządzenia
- SE (*Secondary*) - komunikaty wtórne, adres lub rozkaz wtórny

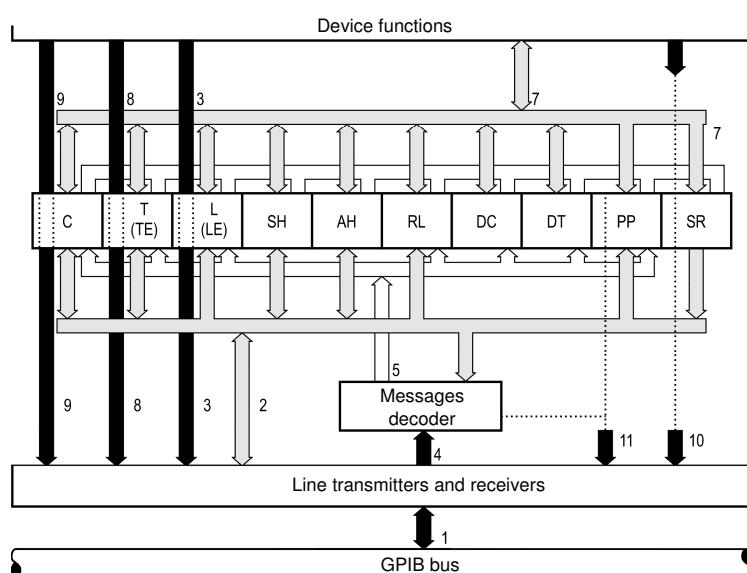
3.1.5 Złącze interfejsu (connector)

W systemie GPIB zastosowano 24-stykowe złącze.

- Wszystkie sygnały mają wspólny potencjał odniesienia (masę SG).
- Linie sterujące: DAV, NRFD, NDAC, IFC, SRQ oraz ATN są wykonane w postaci skręcanej pary przewodów, w których przewód powrotny jest połączony odpowiednio do punktów TP1, TP2, ... TP11.
- Zacisk SHIELD stanowi kontakt do podłączenia ekranu kabla.



Rysunek 2: Złącze



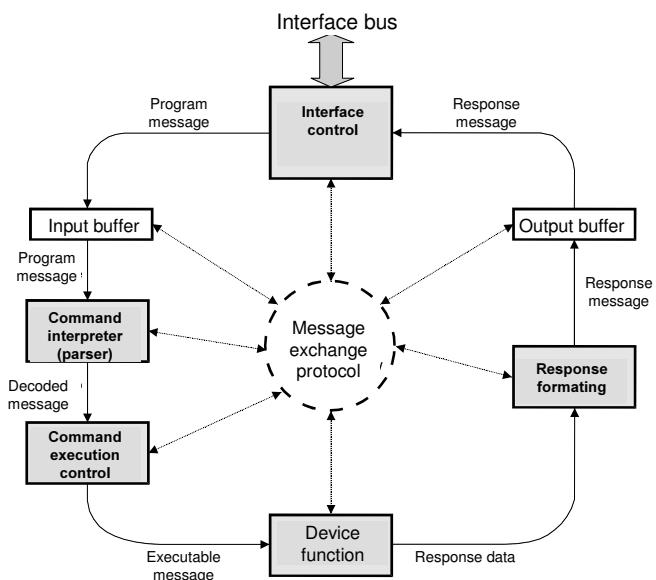
3.2 Standard IEEE-488.2

Standard ten określa:

- Protokół wymiany komunikatów pomiędzy kontrolerem, a urządzeniem na poziomie warstwy 2
- Synchronizację działania kontrolera i urządzenia
- Strukturę systemu statusowego w urządzeniu i sposób jego obsługi
- Zestaw tzw. rozkazów wspólnych

3.2.1 Schemat funkcjonalny urządzenia

Ilustracja przepływu danych i ich przetwarzania



3.2.2 Protokół wymiany komunikatów

Jego zadaniem jest zapewnienie poprawnego transferu wiadomości pomiędzy kontrolerem, a urządzeniem i określenie reakcji urządzenia na naruszenia reguł protokołu.

Ogółem komunikacja pomiędzy kontrolerem a urządzeniem polega na:

- wysyłaniu do urządzenia komunikatów programujących, które można podzielić na rozkazy bez odpowiedzi i zapytania
- odbieraniu odpowiedzi z urządzenia na wysłane zapytania

Zasady komunikacji kontrolera z urządzeniem (*Device responsibility*)

1. Rozkazy są wykonywane przez urządzenie zgodnie z kolejnością ich wprowadzenia do przyrządu (poza ściśle określonymi wyjątkami)
2. Urządzenie jest zobowiązane odesłać odpowiedź tylko na wcześniej odebrane zapytanie (przez kontroler)
3. Odpowiedzi na zapytania są odsyłane w kolejności odpowiadającej odebranym zapytaniom

Reguły protokołu komunikacyjnego (*Controller responsibility*)

1. Kontroler nie może odebrać odpowiedzi na zadane zapytanie, zanim nie zakończy rozpoczętego komunikatu programowego.
2. Kontroler nie może wysłać kolejnego komunikatu programowego, jeżeli nie odebrał w całości wszystkich odpowiedzi na zadane wcześniej zapytania.
3. Kontroler nie może podejmować próby odczytu odpowiedzi z urządzenia, jeżeli wcześniej nie wysłał żadnego zapytania.
4. Zapytanie powodujące odesłanie odpowiedzi o nieokreślonym rozmiarze musi być umieszczone jako ostatnie zapytanie w komunikacie programowym (zawierającym kilka zapytań).
5. Istnieje niebezpieczeństwo zablokowania urządzenia w przypadku wprowadzenia do urządzenia komunikatu programowego zawierającego kilka zapytań.

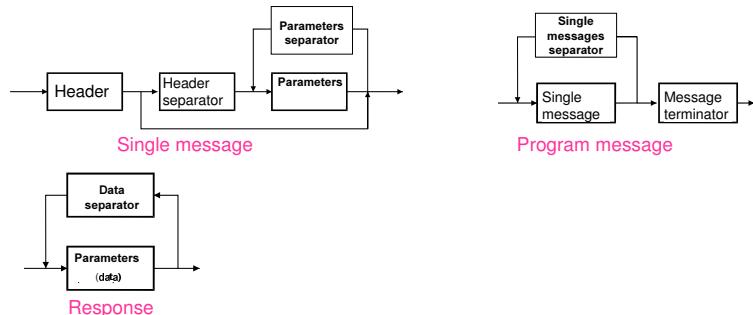
3.3 Standard SCPI

SCPI (*Standard Commands for Programmable Instruments*), normalizuje treści komunikatów przepływających pomiędzy kontrolerem, a urządzeniem. Można go uważać za język kontroli, określono w nim budowę i reguły syntaktyczne komunikatów programowych wysyłanych do urządzenia i odpowiedzi z urządzenia.

Wydodzi się z normy IEEE-488.2.

3.3.1 Budowa komunikatów

Komunikaty składają się z jednego lub kilku komunikatów jednostkowych. Na końcu komunikatu programowego musi wystąpić znacznik terminujący.



Komunikat programowy:

- **Nagłówek:** tekst określający znaczenie komunikatu. Może to być pojedyncze słowo kluczowe lub ciąg słów oddzielonych znakiem dwukropka (':', 3Ah)
- **Separator nagłówka:** występuje pomiędzy nagłówkiem, a pierwszym parametrem jako jeden lub kilka znaków niedrukowalnych ASCII z zakresu od 0 do 31 (dec), oprócz NL, 0Ah. Najczęściej jest to jedna lub kilka spacji.
- **Parametry:** określają wartości związane z rozkazem. Separatorem pomiędzy parametrami jest znak przecinka (',', 2Bh), przy czym białe znaki są ignorowane.
- **Separator komunikatów jednostkowych:** jest to znak średnika (';', 3Bh)
- **Terminator komunikatu:** znacznik terminujący, który jest wymagany do zakończenia komunikatu. Jest to znak LF (NL, 0Ah)

Odpowiedź:

- Zawierają tylko dane,
- a reszta jak w powyższym

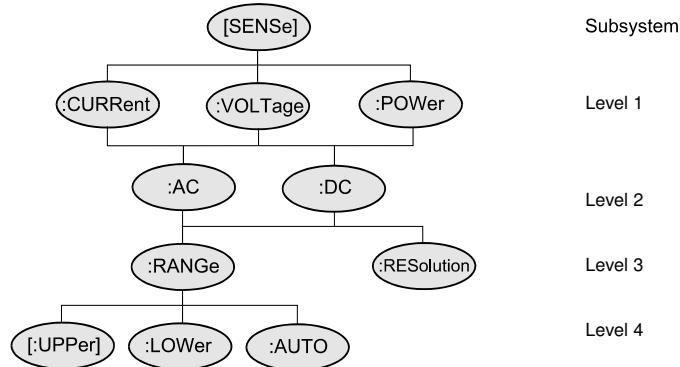
Zapytanie:

- Nagłówki zapytań mają na końcu znak pytajnika ('?', 3Fh)

Komunikaty wspólne:

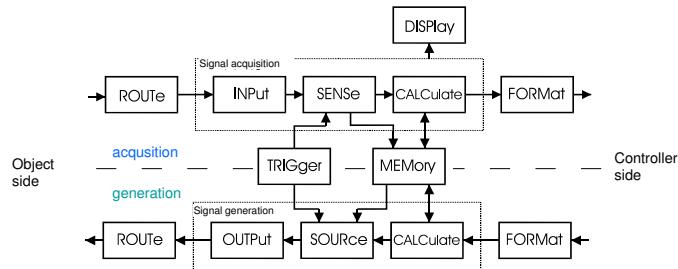
- Posiadają prefiks w postaci znaku gwiazdki ('*', 2Ah)

Drzewiasta struktura rozkazów SCPI (jest to fragment drzewa podsystemu [SENSe]):



3.3.2 Model urządzenia w standardzie SCPI

Schemat blokowy ilustrujący przepływ sygnału i danych w urządzeniu. Poszczególne bloki reprezentują podstawowe funkcje urządzenia, a ich nazwy są jednocześnie nazwami grup rozkazów (podsystemów), które obsługują daną funkcjonalność.



Wyróżnia się dwie części:

- akwizycję sygnału ("czujnik")
- generację sygnału ("źródło")

3.3.3 Podsystemy w standardzie SCPI

Podsystem to grupa funkcjonalności. W każdym konkretnym urządzeniu występują tylko te, które są mu potrzebne.

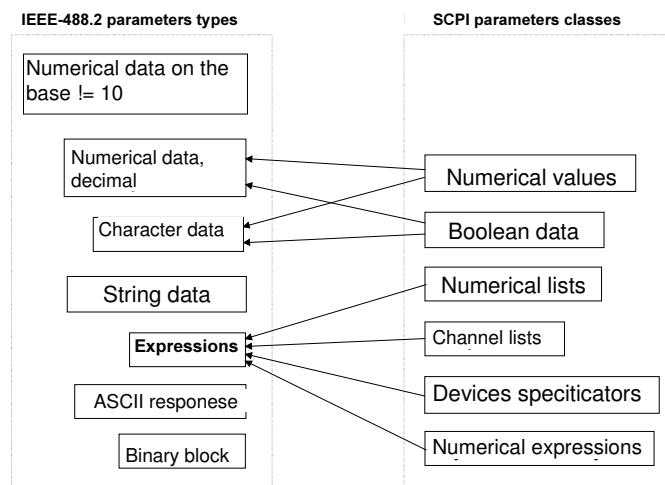
Owych podsystemów jest kilkanaście, tutaj są przedstawione najistotniejsze:

- **CALCulate** - odpowiedzialny za matematyczne przetwarzanie wyników i danych po akwizycji sygnału lub przed generacją sygnału
- **INput** - przekształcanie sygnału (wzmocnieniem, filtracją itp.) przed właściwą konwersją dokonywaną w bloku SENSe
- **OUTput** - kontrola przekształtników sygnału wyjściowego (po generacji przez blok SOURCE)
- **ROUTE** - odpowiedzialny za podłączenie sygnału wejściowego do układu przekształcającego sygnał lub wyjściowego do zacisków wyjściowych przyrządu
- **SENSe** - kontrola podstawowego modułu akwizycji sygnału (bloku wykonującego konwersję analogowo-cyfrową)

- **SOURce** - kontrola modułu generacji sygnału
- **SYSTem** - podsystem umożliwiający "utrzymanie" przyrządu (kontrolę pomocniczych parametrów)

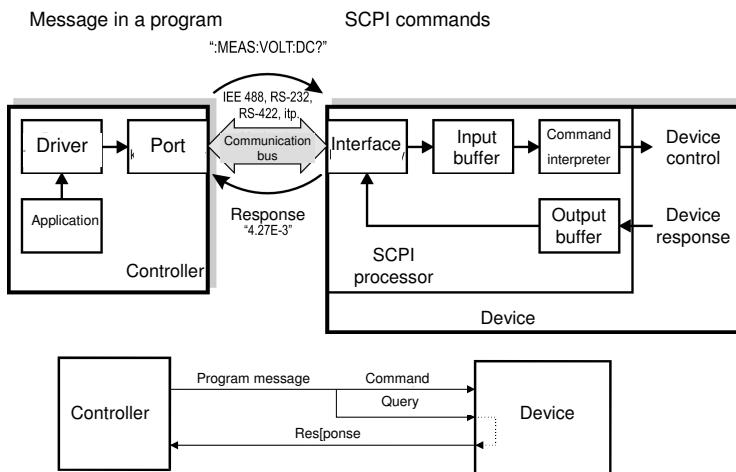
3.3.4 Parametry

Typy parametrów w komunikatach programowych i odpowiedziach zostały określone w standardzie IEEE-488.2, natomiast w SCPI wyróżniono kilka klas parametrów w ramach wybranych typów IEEE-488.2.



3.3.5 Kontrola urządzeń

Urządzenie kontrolowane przez rozkazy SCPI

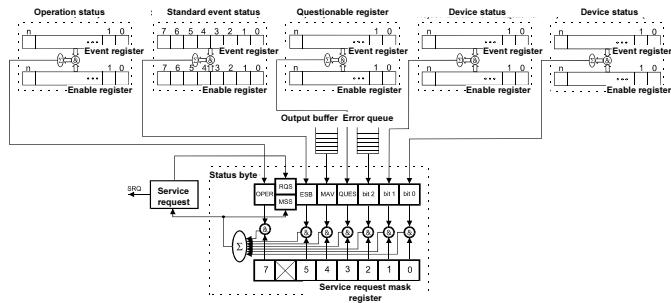


3.3.6 System rejestrów statusu

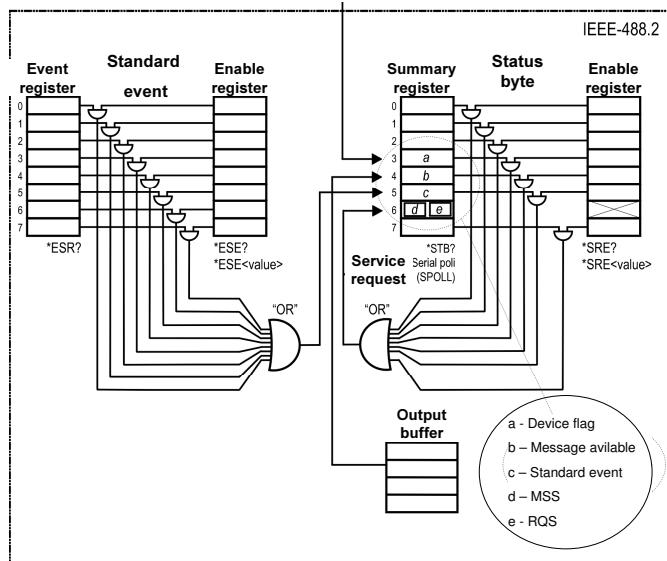
Urządzenie przechowuje informację o stanie w tzw. SRS, który ma strukturę hierarchiczną, złożoną z 5 grup rejestrów statusowych. Każda z nich składa się z rejestrów zdarzeń i rejestrów maski.

1. Bajt statusowy - umieszczona w najwyższym poziomie hierarchii, udostępnia uogólnioną informację o stanie urządzenia.
2. Rejestr zdarzeń standardowych - 8-bitowy rejestr na drugim poziomie hierarchii, informuje o wystąpieniu zdarzeń uważanych za "typowe" dla urządzenia SCPI.
3. Rejestr znaczników urządzenia - 16-bitowy rejestr na drugim poziomie hierarchii, informuje o "jakości" wyników pomiaru i sprawności ważniejszych układów w przyrządzie.

- Rejestr operacji - 16-bitowy rejestr na drugim poziomie hierarchii informujący o stanie wykonywanych przez przyrząd operacji.
- Rejestr statusu zależny od urządzenia - specyficzny dla urządzenia, liczba i znaczenie bitów są od niego zależne.



Minimal SCPI status system

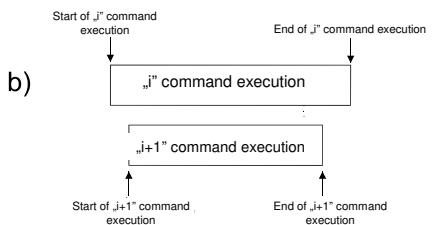
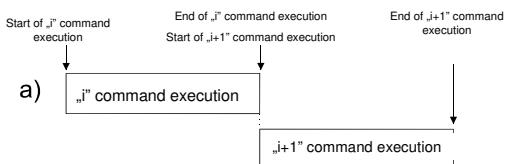


3.3.7 Synchronizacja urządzeń z aplikacją w standardzie SCPI

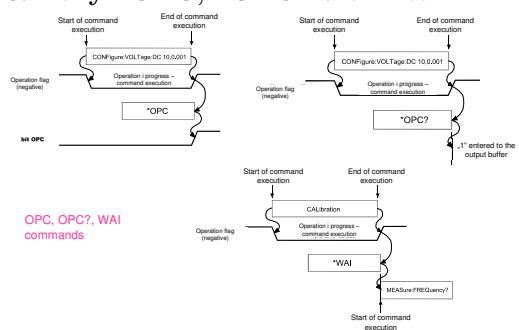
Zadaniem synchronizacji jest zapewnienie właściwej sekwencji operacji urządzeń w systemie. W urządzeniach SCPI synchronizacja aplikacji sterującej z operacjami wykonywanymi przez urządzenie oparta jest na rozkazach *OPC, *OPC? oraz *WAI.

Rodzaje rozkazów Dzielimy je na:

- Sekwencyjne - wykonywane jeden po drugim w kolejności wprowadzania ich do przyrządu.
- Nakładające się - możliwe wykonywanie wielu rozkazów jednocześnie.



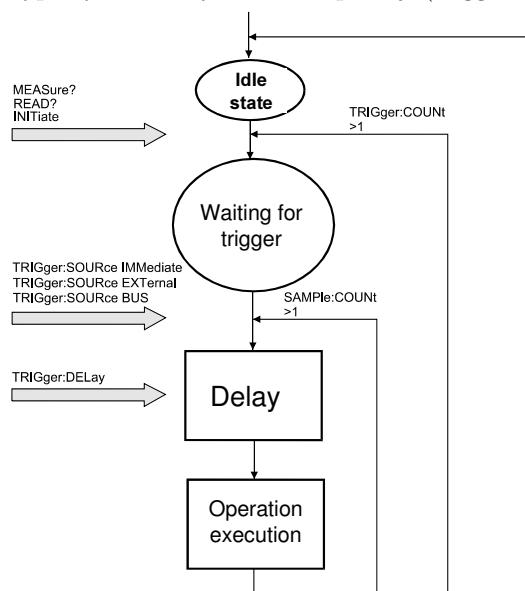
Rozkazy *OPC, *OPC? oraz *WAI



3.3.8 Wyzwalanie operacji (device trigger)

Standard SCPI umożliwia uzależnienie wyzwolenia od spełnienia szeregu warunków i wystąpienia różnych zdarzeń (elastyczność).

Typowy układ wyzwalania operacji (trigger circuit in HP-34401A multimeter)



- 4 IEEE-1394 (FireWire)**
- 5 Tłumienie zakłóceń w rozproszonych systemach komputerowych**