


Politechnika Wrocławska

Technika cyfrowa 2

wykład 9

Katedra Metrologii Elektronicznej i Fotonicznej
Andrzej Stępień



Interfejs


zbiór niezależnych od urządzeń elementów mechanicznych, elektrycznych i funkcjonalnych koniecznych w procesie wymiany informacji pomiędzy urządzeniami.

PN-83/T-0653



Pojęcia podstawowe

- ANSI** (American National Standard Institution) - Amerykański Narodowy Urząd Normalizacyjny
- Tranceiver / Driver** - nadajnik, urządzenie/układ nadający/wysyłający dane
- Receiver** - odbiornik, urządzenie/układ odbierający dane
- Master** - w systemach wieloprocesorowych urządzenie/układ nadrzędny, inicjalizujący i kończący transmisję
- Slave** - w systemach wieloprocesorowych urządzenie/układ podporządkowany, obsługiwany przez urządzenie nadrzędne
- Data Rate. Baud Rate** - szybkość transmisji, liczba bitów przesyłanych znaków w jednostce czasu (bps - baud per second)
- Half Duplex** - czasowe rozdzielenie, naprzemienne nadawanie i odbiór; możliwość różnej szybkości pracy nadajnika i odbiornika
- Full Duplex** - równoczesne niezależne nadawanie i odbiór znaków z tą samą szybkością nadajnika i odbiornika



Typy transmisji

- Unbalanced Data Transmission** - szeregową transmisją danych przy wykorzystaniu jednego przewodu dla nadajnika i dla odbiornika, np. RS-232, I²C, CAN
- Balanced Data Transmission** - szeregową transmisją danych przy wykorzystaniu dwóch przewodów, przeważnie różnicowo, dla nadajnika i dla odbiornika, np. RS-485, CAN, USB
- RS-232** - Recommended Standard 232 (rok 1969)
EIA/TIA-232 - Electronic Industries Association / Telecommunication Industries Association
- I²C Bus** - Inter Integrated Circuit Bus; Philips
- CAN** - Controller Area Network; Bosch
- USB** - Universal Serial Bus; Intel



Kodowanie danych ze składową stałą

Binary Data 1 0 1 1 0 0 1 0

NRZ
(Non Return to Zero)




bezpośrednia zamiana stanów logicznych na sygnały znaków (np. RS232)

NRZI
(Non Return to Zero Inverted)



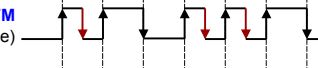
zmiana sygnału dokonywana w połowie transmitowanego znaku - stanu '0';
brak zmiany sygnału dla stanu '1' (np. USB)



Kodowanie danych bez składowej stałej

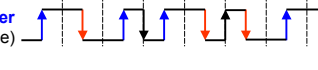
Binary Data 1 0 1 1 0 0 1 0

FM
(Biphase Space)



zamiana stanu logicznego na początku transmitowanego bitu:
dodatkowa zmiana w połowie transmitowanego bitu dla stanu '0'

Manchester
(Biphase)



zmiana sygnału dokonywana w połowie transmitowanego znaku:
zbocze opadające dla stanu '0'
zbocze narastające dla stanu '1'

Politechnika Wrocławska

Transmisja synchroniczna

Wymagane minimum dwie linie sterujące:

- danych (Data)
- taktująca (CLK)

stan linii danych

stan linii taktującej

lub

stan linii taktującej

Politechnika Wrocławska

M37530M4 - Mitsubishi

S_{DATA} at Serial I/O2 Output / Input

Transfer Clock

SIO2CON Register

Transmit/Receive Shift Stop Flag:
0 - shift in progress
1 - shift completed

Transfer Direction Selection Bit:
0 - LSB first
1 - MSB first

Internal Synchronous Clock Selection Bit:
000 - $f_{XIN}/8$
001 - $f_{XIN}/16$
010 - $f_{XIN}/32$
011 - $f_{XIN}/64$
110 - $f_{XIN}/128$
111 - $f_{XIN}/256$

S_{DATA} pin selection bit:
0 - I/O port
1 - S_{DATA} I/O

Politechnika Wrocławska

SPI or Microwire

- SPI** (Serial Peripheral Interface Bus or SPI bus) developed by **Motorola**
- Microwire** developed by **National Semiconductor**
- Synchronous interfaces are characterized by the presence of a dedicated receive/transmit clock signal
- A "Master" device usually outputs a clock signal that is received by all "Slave" devices to receive and transmit data in synch
- The advantage: Each device works with the transmit/receive clock of the master independent of any oscillator variations of each individual device; so these interfaces are very suitable for use with cheap oscillators that have large frequency variations
- In addition I²C is level sensitive - in contrast to Microwire and SPI, which are edge sensitive.

Politechnika Wrocławska

SPI & Microwire Signals

A Microwire multiple slave configuration looks similar, with the difference that the master's SO pin is connected to the slaves' SI pins and the slaves' SO pins are connected to the master's SI pin:

| SPI Signals | Microwire Signals |
|----------------------------|---------------------------------------|
| SCLK – Serial Shift Clock | SK – Serial Shift Clock |
| MOSI – Master Out Slave In | SO – Serial Out (both master & slave) |
| MISO – Master In Slave Out | SI – Serial In (both master & slave) |
| /SS – Slave Select | /CS – Chip Select |

Politechnika Wrocławska

SPI / Microwire - signal

MISO - Master Input Slave Output
SCK - Serial ClOck

MOSI - Master Output Slave Input
/SS - Slave Select

MSB MASTER LSB

MSB SLAVE LSB

8 BIT SHIFT REGISTER

8 BIT SHIFT REGISTER

SHIFT ENABLE

SPI CLOCK GENERATOR

V_{CC}

SI – Serial In (both master & slave)
SK – Serial Shift Clock

SO – Serial Out (both master & slave)
/CS – Chip Select

Politechnika Wrocławska

SPI - mode 0, 2

SPI Transfer Format with CPHA = 0 (początek aktywnego sygnału taktującego SCK)

SCK (CPOL = 0) mode 0
SCK (CPOL = 1) mode 2

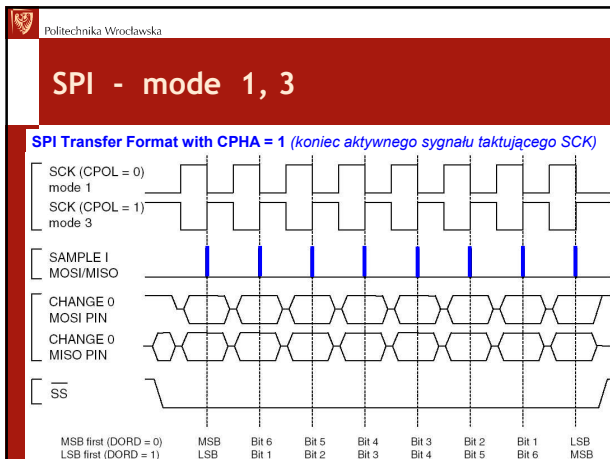
SAMPLE I MOSI/MISO

CHANGE 0 MOSI PIN
CHANGE 0 MISO PIN

SS

MSB first (DORD = 0)
LSB first (DORD = 1)

MSB LSB Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0



Politechnika Wrocławska

Dokumenty

[i2c_bus_specification_3.pdf](#)
THE I2C-BUS SPECIFICATION
 VERSION 2.1, JANUARY 2000
 Philips Semiconductors
<http://www.semiconductors.philips.com/i2c>

[smbus20.pdf](#)
System Management Bus (SMBus) Specification
 Version 2.0, August 3, 2000
 SBS Implementers Forum:
 Duracell, Inc., Energizer Power Systems, Inc., Fujitsu, Ltd.,
 Intel Corporation, Linear Technology, Inc., Maxim Integrated Products,
 Mitsubishi Electric Semiconductor Company, PowerSmart, Inc.,
 Toshiba Battery Co. Ltd., Unitorde Corporation, USAR Systems, Inc.
http://www.smbus.org/specs/spec_content.htm

Politechnika Wrocławska

Właściwości (1/3)

- dołączanie lub odłączanie układów bez zmiany konfiguracji magistrali
- 2-przewodowa, 2-kierunkowa magistrala:
 - **SDA** (Serial Data) - linia danych
 - **SCL** (Serial Clock) - linia zegara (taktująca)
 - obie linie magistrali typu: Open-Drain lub Open-Collector
- każdy układ ma własny, indywidualny adres:
 - 7 bitowy w trybie standardowym (Standard Mode)
 - 10 bitowy w trybie szybkim (Fast Mode)
 - brak zewnętrznych układów dekodujących adresy urządzeń
- szeregową, 8-bitową, 2-kierunkową wymianę danych
- wbudowane protokoły wymiany danych i testowania stanu magistrali
- ograniczenie **szybkości** transmisji danych:
 - 100 kbit/s (Standard)
 - 400 kbit/s (Fast)
 - 3,4 Mbit/s (High-Speed, **Hs**)

Politechnika Wrocławska

Właściwości (2/3)

- każdy układ może pełnić rolę nadajnika danych (**Transmitter**) lub odbiornika danych (**Receiver**), wyjątkiem są sterowniki wyświetlaczy, które mogą być jedynie odbiornikami
- każdy układ może być układem zarządzającym (**Master**) lub podporządkowanym (**Slave**)
- układem nadrzędnym (**Master**) jest układ inicjalizujący transmisję, pozostałe układy stają się automatycznie podporządkowanymi (**Slave**)
- magistralą może zarządzać wiele kontrolerów (**Multi-Master**)
- wbudowane procedury **arbitrażu** w przypadku dostępu do magistrali więcej niż jednego układu zarządzającego (połączenie typu AND na linii SCL)
- procedury **synchronizacji** sygnału taktującego (SCL) w przypadku równoczesnego dostępu do magistrali kilku układów

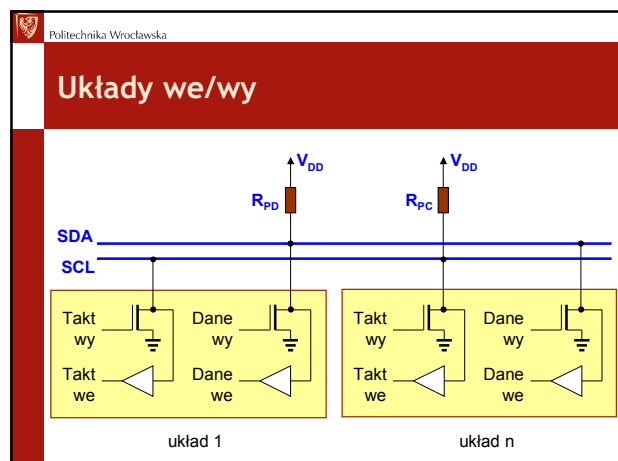
Politechnika Wrocławska

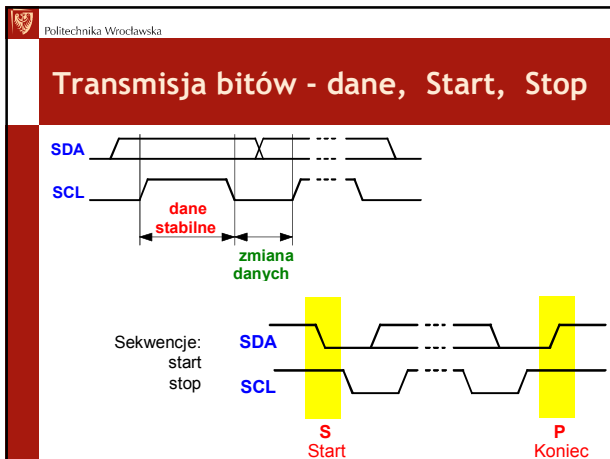
Właściwości (3/3)

- ograniczenie **pojemności** linii SDA i SCL do 400 pF w trybie Standard Mode Fast Mode
- **poziomy** logiczne sygnałów zależne od sposobu dołączenia rezystorów R_p

SDA SCL V_{dd} R_p

- wbudowane mechanizmy **filtracji zakłóceń** impulsowych magistrali
- układy wykonane w **różnych technologiach**: NMOS, CMOS, bipolarne
- interfejs I²C-Bus **zintegrowany** z układami aplikacyjnymi
- **programowalne adresy** urządzeń peryferyjnych
- różnorodne **aplikacje**: przyrządy pomiarowe .. wideorekordery

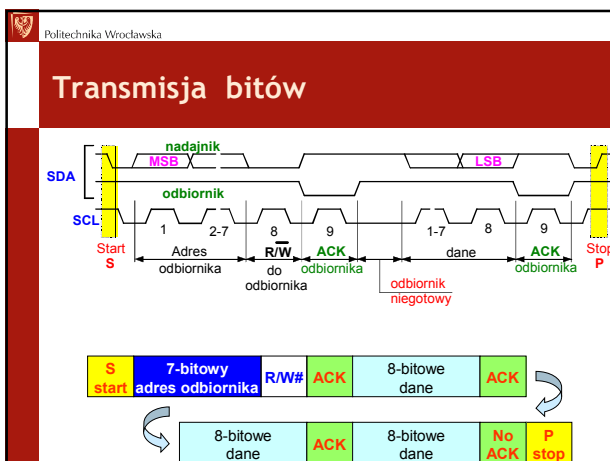




Politechnika Wrocławska

Struktura bitów

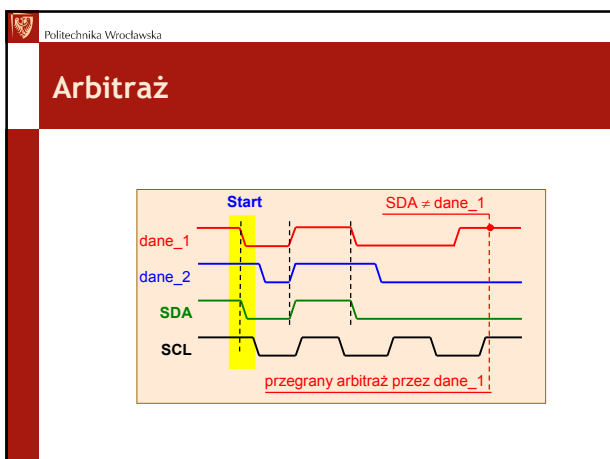
- sygnał taktujący **SCL** generowany zawsze przez układ nadrzędny (Master)
- transmisja **adresu** 7-bitowa w trybie standardowym (Standard Mode) i 10-bitowa w trybie szybkim (Fast Mode)
- transmisja **danych** zawsze 8-bitowa
- liczba transmitowanych danych nieograniczona
- transmisję rozpoczyna sekwencja **startu** (S)
- bit najbardziej znaczący (**MSB**) transmitowany jako pierwszy
- bit **R/W#** określa kierunek transmisji danych:
 - = 0 dane do odbiornika
 - = 1 dane do nadajnika
- bit potwierdzenia (**ACK** - Acknowledge) kończy zawsze transmisję adresu lub danych; dodatkowy 9 impuls taktujący SCL generowany przez urządzenie nadrzędne



Politechnika Wrocławska

Arbitraż

- arbitraż dla linii SDA w czasie SCL = HIGH
- arbitraż dla kolejnych bitów adresu, bitu kierunku R/W# oraz bitów danych
- pozostaje nadajnik, który jako pierwszy wygenerował stan niski (LOW) na linii SDA, podczas gdy inne utrzymywały stan wysoki (HIGH)
- układ przegrywający arbitraż zwalnia linie SDA i SCL
- jeśli nadrzędny nadajnik (master) przegrywa arbitraż, to generuje sygnał taktujący (linia SCL) do zakończenia bieżącego bajtu
- arbitraż zabroniony w trakcie powtarzanej sekwencji startu i stopu



Politechnika Wrocławska

7-bitowa transmisja danych - wpis

- dane z nadrzędnego nadajnika (Master)
- do podporządkowanego odbiornika (Slave)
- z podaniem wewnętrznego adresu odbiornika
- bez zmiany kierunku transmisji danych (nadawanie)

ACK generuje Slave

S 7-bitowy adres odbiornika R/W 0 ACK 8-bitowy adres wewnętrzny odbiornika ACK

8-bitowe dane do Slave ACK 8-bitowe dane do Slave ACK P

Politechnika Wrocławska

7-bitowa transmisja danych - odczyt

- dane z podporządkowanego odbiornika (Slave)
- do nadrzędnego nadajnika (Master)
- z podaniem wewnętrznego adresu odbiornika
- ze zmianą kierunku transmisji danych (odbior)

ACK generuje Slave

ACK generuje Master

Politechnika Wrocławska

Transmisja asynchroniczna

- Brak linii sygnału taktującego
- UART** (Universal Asynchronous Receiver - Transmitter)
- ACE** (Asynchronous Communication Elements)
- SCI** (Serial Communication Interface)

uniwersalny układ do 2-przewodowej, asynchronicznej transmisji szeregowej:

- ramka danych: bit startu, 5-8 bitów danych, 1-2 bity kontrolne, 1-2 bity stopu
- dane kodowane w standardzie NRZ
- oddzielne linie nadajnika i odbiornika
- transmisja full duplex

Politechnika Wrocławska

MC68HC708MP16 - Motorola (1/2)

- transmisja 8- lub 9-bitowa
- podział czasu trwania bitu na 16 części
- rozpoczęcie odbioru znaku po sprzętowym wykryciu zbocza opadającego i bitu startu
- eliminacja szumów i zakłóceń poprzez 3-krotne próbkowanie linii odbiorczej

RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16

RT Clock

RxD

Start Bit

Data Bit

Politechnika Wrocławska

MC68HC708MP16 - Motorola (2/2)

RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16

RT Clock

RxD

Start Bit

Data Bit

| stan RxD dla: RT3, RT5, RT7 RT7, RT8, RT9 | weryfikacja bitu startu/bit danych | NF (Noise Flag) bitu startu/bit danych |
|--|---------------------------------------|---|
| 0 0 0 | tak 0 0 | 0 0 |
| 0 0 1 | tak 0 0 | 1 1 |
| 0 1 0 | tak 0 0 | 1 1 |
| 0 1 1 | nie 1 1 | 0 1 |
| 1 0 0 | tak 0 0 | 1 1 |
| 1 0 1 | nie 1 1 | 0 1 |
| 1 1 0 | nie 1 1 | 0 1 |
| 1 1 1 | nie 1 1 | 0 0 |

Politechnika Wrocławska

Stabilność generatora taktującego

RxD TxD

Start LSB MSB Stop

Bit_k

maksymalna zmiana czasu wewnętrznego testu odebranego bitu:

$$\delta T = \frac{1}{2} - \frac{1}{16}$$

$\delta T < 3\%$ dla 8 bitów danych (teoret. < 4,3%)
 $\delta T < 2,8\%$ dla 9 bitów danych (teoret. < 4,0%)

dodatkowe założenia:

- N = 10 lub 11, liczba przesyłanych bitów znaków
- taka sama tolerancja częstotliwości f_{OSC} nadajnika i odbiornika
- dodatkowe opóźnienia w wewnętrznej strukturze odbiornika
- stany nieustalone w liniach transmisyjnych

Politechnika Wrocławska

C51 - Baud Rate

| Baud Rate derived from: | Interface Mode | Baud Rate |
|-----------------------------|-------------------------|--|
| Timer 1 in Mode 1 | 1, 3 variable Baud Rate | $\frac{2^{SMOD}}{2} * \frac{1}{16} * \text{Timer 1 Overflow Rate}$ |
| Timer 1 in Mode 2 | 1, 3 variable Baud Rate | $\frac{2^{SMOD}}{2} * \frac{1}{16} * \frac{f_{OSC}}{12 * (256 - TH1)}$ |
| Oscillator | 2 - fixed Baud Rate | $\frac{2^{SMOD}}{2} * \frac{1}{16} * \frac{f_{OSC}}{2}$ |
| Baud Rate Gen. in C515/C517 | 1, 3 variable Baud Rate | $\frac{2^{SMOD}}{2} * \frac{f_{OSC}}{1250}$ |
| Baud Rate Gen. in LPC932 | 1, 3 variable Baud Rate | $\frac{CCLK}{(BRGR1, BRGR0) + 16}$ |

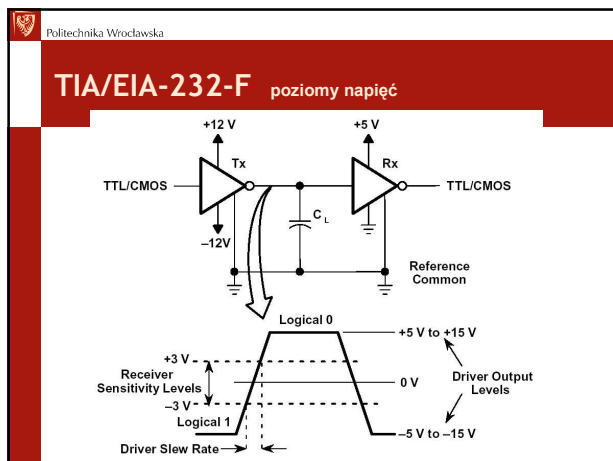
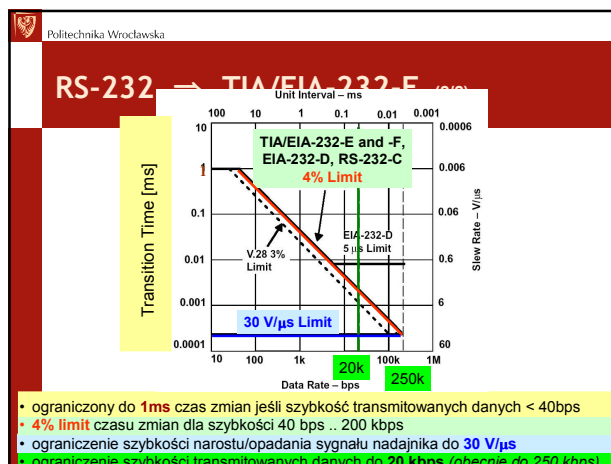
| Rezonator kwarcowy 11.059.200 MHz | | | | | |
|--|------|----------------|---------------|---------------------|--------------------|
| $f_{osc} = 11.059.200 \text{ Hz}$, $BD = 0$ | | | | | |
| Baud Rate | SMOD | TH1 (ideal) | TH1 (real) | Baud Rate (real) | δBd [%] |
| 1.200 | 0 | 232 | 232 | 1.200 | 0 |
| | 1 | 208 | 208 | 1.200 | 0 |
| 2.400 | 0 | 244 | 244 | 2.400 | 0 |
| | 1 | 232 | 232 | 2.400 | 0 |
| 4.800 | 0 | 250 | 250 | 4.800 | 0 |
| | 1 | 244 | 244 | 4.800 | 0 |
| 9.600 | 0 | 253 | 253 | 9.600 | 0 |
| | 1 | 250 | 250 | 9.600 | 0 |
| 19.200 | 0 | 254,5 | 254 | 14.400 | -25 |
| | 1 | 255 | 255 | 28.800 | +50 |
| | 1 | 253 | 253 | 19.200 | 0 |

| Rezonator kwarcowy 12.000.000 MHz | | | | | |
|--|------|----------------|---------------|---------------------|--------------------|
| $f_{osc} = 12.000.000 \text{ Hz}$, $BD = 0$ | | | | | |
| Baud Rate | SMOD | TH1 (ideal) | TH1 (real) | Baud Rate (real) | δBd [%] |
| 1.200 | 0 | 229,96 | 230 | 1.201,9 | 0,16 |
| | 1 | 203,92 | 204 | 1.201,9 | 0,16 |
| 2.400 | 0 | 242,98 | 243 | 2.403,8 | 0,32 |
| | 1 | 229,96 | 230 | 2.403,8 | 0,32 |
| 4.800 | 0 | 249,49 | 249 | 4.464,3 | 6,99 |
| | 1 | 242,98 | 243 | 4.807,7 | 0,16 |
| 9.600 | 0 | 252,74 | 253 | 10.416,7 | 8,5 |
| | 1 | 249,49 | 249 | 8.928,6 | -7,0 |
| 19.200 | 0 | 254,47 | 254 | 15.625 | -19 |
| | 1 | 252,74 | 253 | 31.250 | 63 |

RS-232 \Rightarrow TIA/EIA-232-F (1/2)

Jose M. Soltero, Jing Zhang, and Ernest Cox Linear Products
Low-Voltage, Single-Supply 232-Standard Interface Solutions
Application Report, SLLA083A - SEPTEMBER 2000

- wprowadzony w 1962 w celu standaryzacji szeregowej wymiany danych między:
 - DTE (Data Terminal Equipment) - początkowo PC
 - DCE (Data Circuit-Terminating Equipment) - początkowo modem
- najtańszy, niskonapięciowy, najpowszechniejszy standard szeregowej wymiany danych: myszka, ploter, drukarka, skaner, zewnętrzny modem itp.
- obecnie standard TIA/EIA-232-F określający relację między:
 - szybkością transmitowanych danych
 - czasami transmisji
 - szybkością narostu i opadania zboczy sygnału



TIA/EIA-232-F parametry elektryczne

| PARAMETER | SPECIFICATION/FEATURE | COMMENT |
|-------------------------------------|---|---|
| Mode of operation | Single-ended | One driver and one receiver per line |
| Maximum cable characteristic | 2500 pF | Most cables typically are 15 m to 20 m long |
| Maximum data signaling rate | 20 kbps | Most discrete-IC vendors violate this specification. |
| Maximum common-mode voltage | $\pm 25 \text{ V}$ | |
| Driver output levels | Unloaded $\leq \pm 25 \text{ V}$ Loaded $\pm 5 \text{ V to } \pm 15 \text{ V}$ | |
| Driver load | 3 kΩ to 7 kΩ | |
| Driver slew rate | $\leq 30 \text{ V/μs}$ | |
| Driver output short-circuit current | $\leq 100 \text{ mA}$ | Driver output to any conductor |
| Receiver input resistance | 3 kΩ to 7 kΩ | |
| Receiver sensitivity | $\pm 3 \text{ V}$ | Minimum required input voltage |
| Noise margin | $\pm 2 \text{ V}$ | Difference between minimum driver output voltage level and minimum receiver input voltage |

Politechnika Wrocławska

Historia

- 1985** - Robert Bosch GmbH proponuje interfejs szeregowy w celu ujednolicenia wymiany danych w przemyśle samochodowym (lotniczym), jako odpowiedź na wymagania stawiane przez dwóch niemieckich producentów samochodów: Daimler-Benz i BMW (obecnie także: VW, Renault, PSA, Volvo, Saab i inni)
- podstawowe wymagania:**
 - niskie koszty linii transmisyjnych,
 - duża szybkość przesyłanych danych,
 - duża niezawodność i wiarygodność transmitowanych informacji,
 - odporność na zakłócenia elektryczne,
 - automatyczna (sprzętowa) detekcja możliwie wszystkich błędów występujących w trakcie transmisji danych
- protokół szeregowy** wymiany danych między węzłami nadawczymi i odbiorczymi, między warstwami **fizycznymi** i **logicznymi**

Politechnika Wrocławska

Identyfikator

- w przeciwieństwie do innych standardów, **identyfikator** węzła **nie jest** jego **adresem**
- ten sam **węzeł może mieć wiele różnych identyfikatorów** w zależności od pełnionej funkcji lub sposobu komunikacji
- CAN 2.0A** (1991) zgodność z wersją 1.2, z **11-bitowym identyfikatorem** węzłów, (praktycznie 2032 węzłów)
- CAN 2.0B** (1991) standard rozszerzony z **29-bitowym identyfikatorem** węzłów:
 - w trybie **pasywnym** - nadawanie i odbiór ramek w trybie standardowym z kontrolą błędów transmisji i ramek w trybie rozszerzonym bez sygnalizacji błędów transmisji,
 - w trybie **aktywnym** nadawaniu i odbiorowi ramek w trybie standardowym i rozszerzonym towarzyszy kontrola błędów transmisji.

Politechnika Wrocławska

Magistrala

- elastyczność systemu** - dołączanie kolejnych węzłów bez konieczności zmian sprzętowych i programowych
- konfiguracja sieci typu **Multi-Master**, zorientowana obiektowo
- niezawodność magistrali** - detekcja i sygnalizacja przez każdy węzeł błędów transmisji:
 - wszystkich błędów globalnych,
 - wszystkich błędów lokalnych nadajników,
 - do 5 przypadkowych błędów w ramce,
 - do 15 błędów w ramce powodujących zmianę długości poszczególnych części ramki,
 - dowolnego błędu nieparzystości w ramce,
 - grupowanie danych w ramki, od 0 do 8 bajtów w ramce,
 - prawdopodobieństwo niewykrycia błędu w ramce mniejsze niż: $\text{stopa_błędów} * 4,7 * 10^{-11}$

Politechnika Wrocławska

Transmisja ramek

- automatyczna detekcja błędów** transmisji:
 - grupowanie danych w ramki (**Frame**),
 - kontrola nadawanych danych i stanu magistrali,
 - cykliczna kontrola nadmiarowa (**CRC**),
 - bity separacji i potwierdzenia (**ACK**),
 - dotatkowe bity synchronizacji (**Stuff bits**),
- 2-przewodowa-różnicowa** lub 1-przewodowa magistrala, prowadzona za pośrednictwem kabla koncentrycznego, skrętki lub światłowodu,
- szybkość** transmisji danych od **5 Kbitów/s** .. do **1Mbita/s** (40 m),
- maksymalna **odległość** węzłów ograniczona do **10 km** (5 Kbitów/s).

Politechnika Wrocławska

Ramka 2.0B

Extended Format

| Arbitraż | | Bity kontrolne | | Dane | Suma kontrolna | Potw. | Koniec ramki | Przerwa | Oczekiwanie |
|-------------------|----------------------------|----------------------------|-----|------------|-------------------|---------------|--------------|------------|-------------|
| Arbitration Field | | Control Field | | Data Field | CRC Field | ACK | EOF | INT | Bus Idle |
| SOF | 11 bitowy identyfikator ID | 18 bitowy identyfikator ID | RTR | DLC | Bajty danych 0..8 | 15-bitowy CRC | 7 bitów | min 3 bity | |
| 1D28..1D18 | | 1D17..1D0 | | | | | | | |

SOF - Start Of Header
 ID - Identifier
 RTR - Remote Transmit Request
 SRR - Substitute Remote Request
 IDE - Identifier Extension bit
 r1, r0 - reserved bits

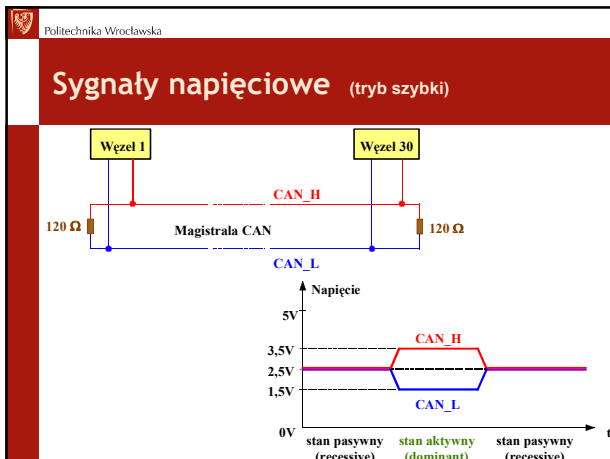
DLC - Data Length Code
 CRC - Cyclic Redundancy Code
 ACK - ACKnowledge
 EOF - End Of Frame
 INT - INTerframe space

Politechnika Wrocławska

Kodowanie

ze względu na brak sygnału zegarowego w magistrali i problemy **synchronizacji** układów peryferyjnych (np. SLIO/CAN P82C150 - Philips) ma wewnętrzny generator synchronizowany stanem magistrali, wprowadzono **kodowanie bitów metodą dostawiania bitu (Bit Stuffing)**:

- po 5 bitach aktywnych (**dominant**) wstawiany jest 1 bit pasywny (**recessive**),
- po 5 bitach pasywny (**recessive**) wstawiany jest 1 bit aktywny (**dominant**)



Politechnika Wrocławska

Standardy

| | |
|--------------|--|
| CANaerospace | www.mstock.com (<i>Michael Stock Flight Systems</i>) |
| CANary | www.atmel.com |
| CANopen | www.can-cia.de |
| CAN Kingdom | www.cankingdom.org |
| DeviceNet | www.odva.org |
| NMEA2000 | communication protocol based upon the J1939 Controller Area Network standard from the National Marine Electronics Association (NMEA) to interconnect various electronic units onboard ships and smaller recreational and commercial vessels. |
| PeliCAN | www.semiconductors.philips.com |
| SAE J1939 | www.sae.org (<i>The Society of Automotive Engineers</i>) |

Politechnika Wrocławska

Wybrane aplikacje

CANaerospace (CAN-based avionics system network): the 1-Mbit/s CAN bus is used as a backbone network for flight state sensors, navigation systems and several research PCs driving high resolution flat panel displays installed in the cockpit. They will be running the SuSE 8.0 Linux operating system and use nVidia graphics accelerators.

The **automotive** industry uses CAN as the in-vehicle network (IVN) for the engine management (*CAN high-speed networks* (e.g. 500 kbit/s), the body electronics like door and roof control, air conditioning, and lightning, as well as for the entertainment control (*lower data-rates*, e.g. 125 kbit/s).

Lifts and escalators have always used embedded CAN networks - all applicable devices in a lift, such as panels, controller, doors, drives, light barriers, etc. are linked to each other and controlled via CAN.

CAN is used as embedded network in **medial devices** such as in X-ray machines. Complete operating rooms are equipped with a CAN network that manages all functions. CAN is also used as embedded network in patient beds. In addition, complete hospital control systems with voltage control, indication and control units, multi cube power meters and digital I/O, and visualization software are networked via CAN.

Politechnika Wrocławska


Dokumenty

v. 1.1 September 23, 1998

Compaq
Intel
Microsoft
NEC

v. 2.0 April 27, 2000

Compaq
Hewlett-Packard
Intel
Lucent
Microsoft
NEC
Philips



www.usb.org

Politechnika Wrocławska

Data Flow Types

- The USB supports functional data and control exchange between the USB host and a USB device as a set of either uni-directional or bi-directional **pipes**.
- USB data transfers take place between host software and a particular **endpoint** on a USB device.
- Such associations between the host software and a USB device endpoint are called pipes. In general, data movement through one pipe is independent from the data flow in any other pipe.
- A given USB device may have many pipes. As an example, a given USB device could have an endpoint that supports a pipe for transporting data to the USB device and another endpoint that supports a pipe for transporting data from the USB device.

Politechnika Wrocławska

Data Flow Types - basic types of data transfers

- Control Transfers:**
Used to configure a device at attach time and can be used for other device-specific purposes, including control of other pipes on the device.
- Bulk Data Transfers:**
Generated or consumed in relatively large and bursty quantities and have wide dynamic latitude in transmission constraints.
- Interrupt Data Transfers:**
Used for characters or coordinates with human-perceptible echo or feedback response characteristics.
- Isochronous Data Transfers:**
Occupy a prenegotiated amount of USB bandwidth with a prenegotiated delivery latency. (Also called streaming real time transfers).

A **pipe supports only one of the types of transfers** described above for any given device configuration.

Politechnika Wrocławska

Data Speed - Lines (D+ and D-)

- Low Speed:** 1,5 Mbit/s $\pm 1.5\%$ (15,000ppm)
- Full Speed:** 12 Mbit/s $\pm 0.25\%$ (2,500ppm) / host $\pm 0.05\%$ (500ppm)
- High Speed:** 480 Mbit/s $\pm 0.05\%$ (500 ppm)

Example Full-speed CMOS Driver Circuit

Politechnika Wrocławska

Device Speed Identification

The **high-speed** device leaves the D+ pull-up resistor connected, leaves the high-speed terminations disabled, and drives the high-speed signaling current into the D- line.

Politechnika Wrocławska

Data Encoding/Decoding

- The USB employs **NRZI** data encoding when transmitting packets. In NRZI encoding, a "1" is represented by no change in level and a "0" is represented by a change in level.
- The **high** level represents the **J** state on the data lines in this and subsequent figures showing NRZI encoding. A string of zeros causes the NRZI data to toggle each bit time.

Politechnika Wrocławska

Bit Stuffing / Synchronization (SYNC)

Bit stuffing is enabled beginning with the **Sync Pattern** (0x80) and throughout the entire transmission. The data "one" that ends the Sync Pattern is counted as the first one in a sequence. Bit stuffing by the transmitter is always enforced, without exception.

Politechnika Wrocławska

Packet Formats

Token Packet: Sync (8 bits), PID (8 bits), Addr (7 bits), ENDP (4 bits), CRC5 (5 bits), EOP ($\geq 2 \cdot T_{PERIOD}$), Idle

SOF Packet: Sync (8 bits), PID (8 bits), Frame# (11 bits), CRC5 (5 bits), EOP ($\geq 2 \cdot T_{PERIOD}$), Idle

Data Packet: Sync (8 bits), PID (8 bits), Data (0..1023 bytes), CRC16 (16 bits), EOP ($\geq 2 \cdot T_{PERIOD}$), Idle

Handshake Packet: Sync (8 bits), PID (8 bits), EOP ($\geq 2 \cdot T_{PERIOD}$), Idle

PID Packet Format: PID₀, PID₁, PID₂, PID₃, PID₀, PID₁, PID₂, PID₃

Sync Pattern: 0 0 0 0 0 0 0 1

Politechnika Wrocławska

PID Packet Formats

| Pocket Type | PID Name | PID ₀ | PID ₁ | PID ₂ | PID ₃ | PID ₀ | PID ₁ | PID ₂ | PID ₃ |
|-------------|----------|------------------|------------------|------------------|------------------|------------------|------------------|------------------|------------------|
| Token | OUT | 1 | 0 | 0 | 0 | | | | |
| Token | IN | 1 | 0 | 0 | 1 | | | | |
| SOF | SOF | 1 | 0 | 1 | 0 | | | | |
| Token | SETUP | 1 | 0 | 1 | 1 | | | | |
| Data | Data0 | 1 | 1 | 0 | 0 | | | | |
| | Data1 | 1 | 1 | 0 | 1 | | | | |
| Handshake | ACK | 0 | 1 | 0 | 0 | | | | |
| | NAK | 0 | 1 | 0 | 1 | | | | |
| | STALL | 0 | 1 | 1 | 1 | | | | |
| Special | PRE | 0 | 0 | 1 | 1 | | | | |

SOF marker, frame number

Data packet PID even
Data packet PID odd

Error-free data packet
Device cannot accept or send data
Endpoint halted

Enables low-speed

Politechnika Wroclawska

What is USB Enumeration ?

- **Enumeration** is the process by which a USB device is attached to a system and is assigned a specific **numerical address** that will be used to access that particular **device**. It is also the time at which the USB host controller queries the device in order to decide what type of device it is in order to attempt to assign an appropriate driver for it.
- Some of the **basic commands** issued by the host to the device are:
 - **Get Device Descriptor** – Overall information about the device (manufacture, firmware version ...)
 - **Set Address** – Instructs the device change it's current address settings
 - **Get Configuration Descriptor** – How the endpoints will be used
 - **Get Interface Descriptor** – Various different interface that the device may use
 - **Get String Descriptor** – Unicode strings for Manufacture and Product
- This process is a **fundamental step for every USB device**, fore without it, the device would never be able to be used by the OS.

Politechnika Wroclawska

Standard Device Descriptor

The device descriptor is sent by the device when the Host sends a GET_DESCRIPTOR request with a DEVICE Descriptor type.

```
struct usb_device_descriptor {
    Uchar bLength;           /* Size of this descriptor in bytes */
    Uchar bDescriptorType;   /* DEVICE descriptor type */
    UInt16 bcdUSB;           /* Binay Coded Decimal Spec. release */
    Uchar bDeviceClass;      /* Class code assigned by the USB */
    Uchar bDeviceSubClass;   /* Sub-class code assigned by the USB */
    Uchar bDeviceProtocol;   /* Protocol code assigned by the USB */
    Uchar bMaxPacketSize0;   /* Max packet size for EP0 (8, 16, 32, 64) */
    UInt16 idVendor;         /* Vendor ID */
    UInt16 idProduct;        /* Product ID assigned by the manufacturer */
    UInt16 bcdDevice;        /* Device release number */
    Uchar iManufacturer;     /* Index of manu. string descriptor */
    Uchar iProduct;          /* Index of prod. string descriptor */
    Uchar iSerialNumber;     /* Index of S.N. string descriptor */
    Uchar bNumConfigurations; /* Number of possible configurations */
};
```

Politechnika Wroclawska

Device Descriptor - example

DEVICE Descriptor:

```
code struct usb_device_descriptor usb_device_descriptor = {
    0x12,           /* Size of this descriptor in bytes */
    0x01,           /* DEVICE descriptor type */
    0x1001,         /* Binay Coded Decimal Spec. release */
    0x00,           /* Class code assigned by the USB */
    0x00,           /* Sub-class code assigned by the USB */
    0x00,           /* Protocol code assigned by the USB */
    0x08,           /* Max packet size for EP0 (8, 16, 32, 64) */
    0xEB03,         /* Vendor ID (0x03EB - Atmel) */
    0x320,          /* Product ID assigned by the manufacturer (0x2003 - HID Keyboard) */
    0x0001,         /* Device release number */
    0x01,           /* Index of manu. string descriptor */
    0x02,           /* Index of prod. string descriptor */
    0x03,           /* Index of S.N. string descriptor */
    0x01           /* Number of possible configurations */
};
```

Politechnika Wroclawska

1 Sync SOF SetUp (Addr=00, Ep=0) Data0 (8 bytes) ACK Get Descriptor
 2 Sync SOF IN (Addr=00, Ep=0) Data1 (8 bytes) ACK Device Descriptor
 3 Sync SOF OUT (Addr=00, Ep=0) Data1 (0 bytes) ACK Null packet
 4 Reset (10ms)
 5 Sync SOF SetUp (Addr=00, Ep=0) Data0 (8 bytes) ACK Device address
 6 Sync SOF IN (Addr=00, Ep=0) Data1 (0 bytes) ACK Null packet
 7 Sync SOF SetUp (Addr=02, Ep=0) Data0 (8 bytes) ACK Get Descriptor
 8 Sync SOF IN (Addr=02, Ep=0) Data1 (8 bytes) ACK Device Descriptor #1
 9 Sync SOF IN (Addr=02, Ep=0) Data0 (8 bytes) ACK Device Descriptor #2
 10 Sync SOF IN (Addr=02, Ep=0) Data1 (2 bytes) ACK Device Descriptor #3
 11 Sync SOF OUT (Addr=02, Ep=0) Data1 (0 bytes) ACK Null packet
 12 Sync SOF SetUp (Addr=02, Ep=0) Data0 (8 bytes) ACK Config. Descriptor
 13 Sync SOF IN (Addr=02, Ep=0) Data1 (8 bytes) ACK Configuration
 14 Sync SOF IN (Addr=02, Ep=0) Data0 (8 bytes) ACK Config. + Interface
 15 Sync SOF IN (Addr=02, Ep=0) Data1 (8 bytes) ACK Interface + Endpoint
 16 Sync SOF IN (Addr=02, Ep=0) Data0 (1 bytes) ACK Endpoint
 17 Sync SOF OUT (Addr=02, Ep=0) Data1 (0 bytes) ACK Null packet