Informační a řídicí systémy I.
Průmyslové komunikace II

Pavel Balda
ZČU v Plzni, FAV, KKY

Osnova přednášky

- Ethernet powerlink
- Modbus

Doporučená literatura

- http://ethernet.industrial-networking.com
- http://www.ethernet-powerlink.org
- http://www.br-automation.com
- http://www.can-cia.org
- Modbus Application Protocol Specification V 1.1b. Dostupná na http://modbus.org

Kolize v Ethernetu

- Dlouhou dobu byly hlavní překážkou rozšíření Ethernetu do průmyslu

Co je Ethernet Powerlink

- Ethernet byl vyvinut před cca 30 lety
  - Nebyl a nemohl být brán ohled na budoucí potřeby v průmyslu
  - Není např. určen pro přenos dat v reálném čase mezi výkonnými pohony a špičkovými PLC
- Ethernet Powerlink (EPL) je nový protokol vyvinutý firmou B&R Automation (Bernecker + Rainer Industrie-Elektronik GmbH)
  - Jeho specifikace je spravována organizací EPSG (Ethernet Powerlink Standardization Group), která zaručuje otevřenost standardu a jeho další rozvoj
  - Umožňuje deterministický přenos dat s periodami cyklu až 200 mikrosekund a velmi přesným časováním. Jeho nepřesnost (tzv. jitter) je menší než 1 mikrosekunda !
  - Je jediným takto výkonným protokolem, který neporušuje žádné standardy Ethernetu!
  - Poprvé je umožněno využít „sílu“ informačních technologií ve světě automatizace
  - Je vhodný zejména pro řízení pohonů, vstupy a výstupy, vizualizaci a výměnu dat mezi programovatelnými automaty.

5

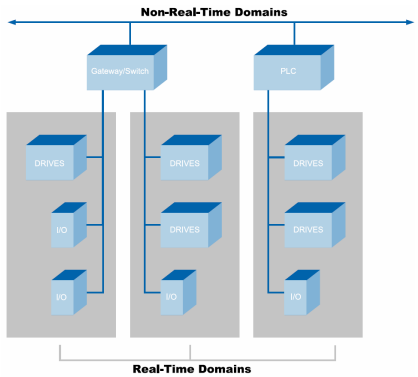
Proč Ethernet Powerlink?

- Žádné riziko, že „dojdou součástky“
  - Ethernet Powerlink může běžet na jakémkoliv standardním integrovaném Ethernetovém obvodu
- Volnost implementace
  - Zařízení podporující EPL mohou běžet např. na těchto platformách: Altera FPGA, ARM, Hilscher NetX, Hyperstone, Intel XScale, Freescale, Infineon, NetSilicon, atd.
- Dostupnost technické podpory
  - Není třeba začínat od nuly, existují „Evaluation kits“, lze i zakoupit sw implementaci na klíč
- Univerzalita
  - V EPL hardwaru mohou běžet i jiné průmyslové protokoly, např. Modbus TCP, Ethernet/IP nebo Profinet SRT
  - EPL umožňuje jakoukoliv architekturu sítě, např. hvězdu, strom, daisy chain.
  - EPL umožňuje rozšiřování za běhu (hot plugging), proto je ideální jako páteřní síť pro modulární stroje
- EPL je Ethernet
  - Jakákoliv standardní síťová technologie může být kombinována s EPL

6

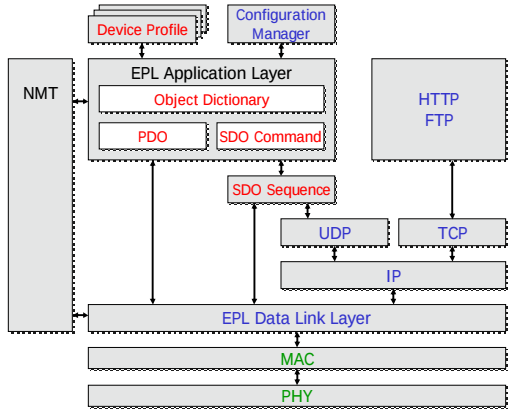
Struktura sítě

- EPL rozlišuje dva typy domén:
  - Real-Time domény
    - Slouží pro komunikaci dat požadovaných v reálném čase
  - Non-Real-Time domény
  - Méně kritická data jsou transparentně routována mezi oběma typy domén pomocí rámců IP protokolu.
  - Rozdělení odpovídá hierarchii podnik – stroj a požadavkům na zvýšené zabezpečení komunikace na úrovni stroje



7

ISO/OSI model pro EPL (1/2)



8

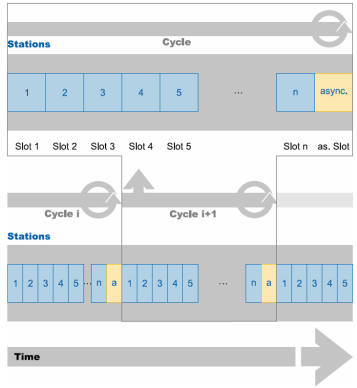
ISO/OSI model pro EPL (2/2)

- NMT – Nework Management
  - Mechanismus pro řízení a monitorování konzistence sítě během nabíhání (boot-up) i provozu (running)
- Object Dictionary a Device Model
  - Obecná metoda pro specifikaci dat, parametrů a funkcí, které jsou poskytovány daným zařízením (nebo typem zařízení)
- PDO – Process Data Object
  - Obecný mechanismus umožňující specifikovat data, vyměňovaná mezi různými zařízeními
- SDO – Service Data Object
  - Obecný mechanismus pro přenos většího množství dat (než pomocí PDO), např. konfiguračních dat
- Device Profiles
  - Standardizované definice dat, parametrů a funkcí určitých typů zařízení, např. pohonů, vstupně-výstupních modulů, enkodérů, PLC, apod.
- Uvedené pojmy jsou kompatibilní se specifikací CANopen (protokol CAN, Controller Area Network)

9

Spojová vrstva (Data Link Layer)

- Deterministické časování
  - Dosaženo cyklickým střídáním (rozvrhováním, schedule) všech připojených uzlů pro přístup k fyzické vrstvě
- Rozvrh je rozdělen na dvě fáze:
  - Izochronní fáze – přenášejí se časově kritická data
  - Asynchronní fáze – rezerva pro přenost časově nekritických dat
- V síti existuje speciální uzel, tzv. Managing node, přidělující přístup k fyzickém médiu pomocí řídicích zpráv.
- V daném čase má tento přístup právě jedno zařízení, tím je zabráněno vzniku kolizí



10

Adresování v EPL

- EPL používá MAC (Media Access Control) adresy podle normy IEEE 802.3 (normy specifikující standardy fyzické vrstvy Ethernetu)
  - Každé zařízení má svou MAC adresu
- Navíc každému uzlu (zařízení) v real-time doméně je přiděleno tzv. EPL node ID
  - Node ID lze volit přepínačem na předním panelu daného zařízení
- Kromě toho může mít každé zařízení i svou IP adresu
  - tj. i real-time zařízení mohou být dostupná odkudkoliv z Internetu
  - Zařízení v real-time doméně mají přiděleny lokální IP adresy, odvozené od daného Node ID. Pro umožnění přístupu z internetu se používá NAT (Network Address Translation)
  - Mechanismus přidělování IP address pomocí DHCP není vhodný, protože se adresa může měnit. Proto se v EPL nepoužívá

11

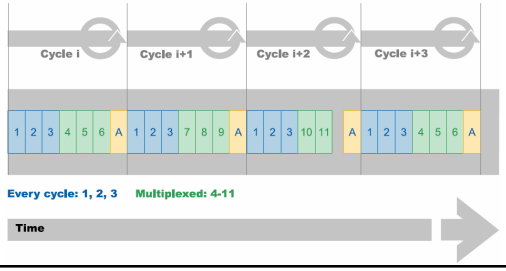
Pracovní režimy EPL zařízení

- Zařízení, které umožňuje komunikaci v EPL umí pracovat v režimech:
  - Základní Ethernetový režim (Basic Ethernet Mode) – zařízení pracuje přímo v existující síti Ethernet, pokud není zapotřebí přenášet data v reálném čase. Toto je přednastavený mód po zapnutí zařízení
  - Pre-Operational Mode – zavádění inicializačních a konfiguračních dat pomocí asynchronního kanálu během startu systému nebo po připojení do existující sítě
  - Ethernet Powerlink Mode – po dokončení startu vstoupí zařízení do tohoto režimu (běh v reálném čase). Řídicí uzel dohlíží na časování. Perioda cyklu závisí na množství izochronních a asynchronních dat a na počtu uzlů
- Základní cyklus má následující fáze:
  - Startovací fáze (Start phase) – všechny uzly sítě se synchronizují na hodiny (clock) řídicího uzlu
  - Izochronní fáze – řídicí uzel přiřadí pro přenos časově kritických dat každému uzlu pevné časové okénko. Ostatní uzly mohou tato data poslouchat během celé fáze (publish/subscribe)
  - Asynchronní fáze – řídicí uzel udělí právo poslat data jednomu konkrétnímu uzlu. V této fázi se používají standardní IP protokoly a adresování

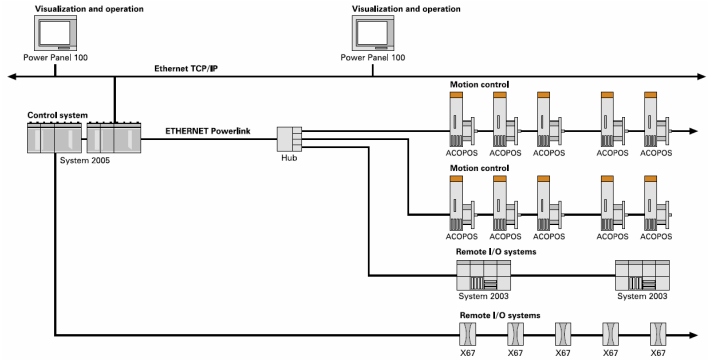
12

Využití přenosového média

- Izochronní data lze přenášet:
  - V každém cyklu (viz 1,2,3) – maximální rychlostí
  - Multiplexovaně (viz 4 až 11) – data jsou stále ještě časově kritická, mohou však být přenášena s delší periodou (násobek základní periody cyklu). O přiřazení časových oken rozhoduje řídicí uzel



Příklad: Struktura ŘS založeného na EPL



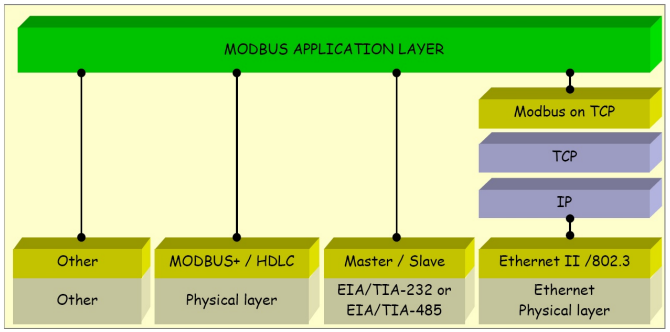
Modbus

- Komunikační protokol uvedený na trh firmou Modicon (dnes součást Schneider Electric) v roce 1979
- Sloužil pro komunikaci s Modicon PLC
- Původně komunikoval po sériové lince RS232 nebo RS485
- Stal se standardním protokolem v průmyslu hlavně z následujících důvodů:
  - Je otevřený, publikovaný a dostupný bez poplatků
  - Je jednoduchý, lze jej implementovat za několik dnů, ne měsíců
  - Komunikuje bity a wordy (16 bitové) bez dalších omezení na výrobce
- Existují dvě základní verze (obě komunikují po sériových linkách)
  - Modbus RTU (Remote Terminal Unit, viz SCADA) – binární verze protokolu, kde na konci příkazu je CRC (cyclic redundancy check)
  - Modbus ASCII – textově čitelná verze zakončená LRC (longitudinal redundancy check)
- Později přibyla verze Modbus TCP (nebo Modbus over TCP/IP) lišící se od Modbus RTU tím že pakety posílá uvnitř datagramů IP.

Omezení protokolu Modbus

- Plynou z omezení PLC na konci 70. let minulého století
- Podporuje jen 1bitové a 16bitové proměnné
  - V různých verzích je přidána (nestandardní) podpora i dalších typů, např. 32 bitová čísla v pohyblivé řádové čárce (float), 8 a 32 bitová celá čísla, apod.
- Nejsou podpořeny velké bloky binárních dat (např. pro download konfigurace)
- Komunikační uzel nemůže zjistit strukturu (popis) dat
- Protože Modbus je Master/Slave protokol, nemůže zařízení (slave) oznámit „výjimku“ (např. alarm) z vlastní iniciativy
  - Master musí takové stavy testovat, což zvyšuje zatížení komunikace
- Maximální počet zařízení na jedné lince je 254

ISO-OSI model pro Modbus (1/2)



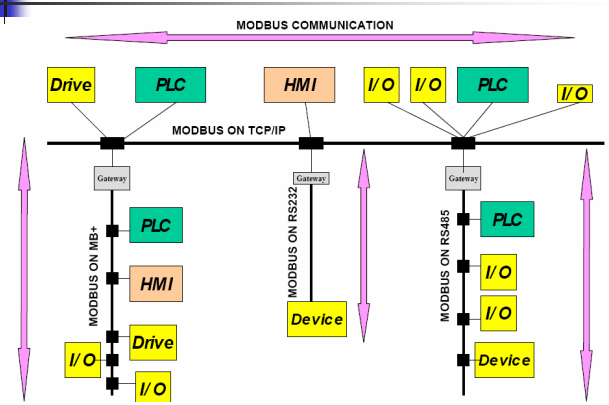
17

ISO-OSI model pro Modbus (2/2)

- Modbus je aplikační protokol (7. vrstva OSI modelu)
- Architektura klient/server po různých typech sítí
- Protokol typu požadavek/odpověď (request/reply)
- Služby protokolu Modbus se specifikují pomocí kódů funkcí (function codes)

18

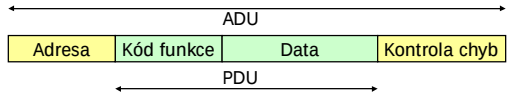
Př.: Architektura sítě s protokolem Modbus



19

Protokolová datová jednotka Modbus

- Protokolová datová jednotka (PDU, Protocol Data Unit)
  - Nezávislá na nižších síťových vrstvách
- Aplikační datová jednotka (ADU, Application Data Unit)
  - Vznikne z PDU přidáním dalších položek pro přizpůsobení protokolu Modbus konkrétním sítím

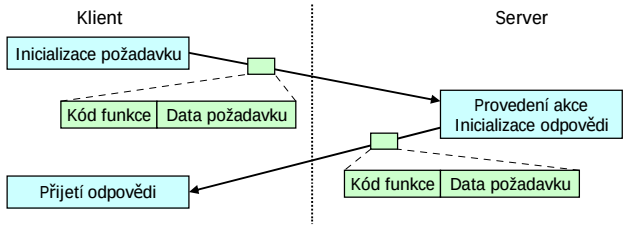


- Kód funkce (function code) volí klient pro provedení požadované akce
  - 1 bajt v rozsahu 1...255 (0 je nepřípustný kód)
- Data obsahují další informaci pro daný kód funkce zadaný klientem
  - Např. adresa registru, počet zpracovávaných položek, apod.
  - Pro některé funkce jsou data prázdná
- 16 bitová data (word) se zapisují v pořadí vyšší bajt, nižší bajt (Big Endian)

20

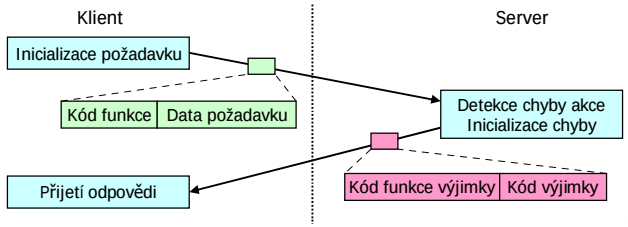
Zpracování transakce – bez chyby

- Klient: připraví požadavek a vyšle jej
- Server: přijme požadavek a provede příslušnou akci
- Server: v případě, že akce skončila v pořádku, připraví odpověď se stejným kódem funkce, jaký obdržel předtím od klienta a odpověď vyšle
- Klient: Přijme odpověď



Zpracování transakce – s chybou

- Klient: připraví požadavek a vyšle jej
- Server: přijme požadavek a provede příslušnou akci
- Server: v případě, že akce skončila chybou, připraví odpověď se kódem funkce složeném z kódu od klienta, v němž je nejvyšší bit nastaven na logickou 1 a v datech je uveden kód výjimky (1 byte)
- Klient: Přijme odpověď



Datový model protokolu Modbus

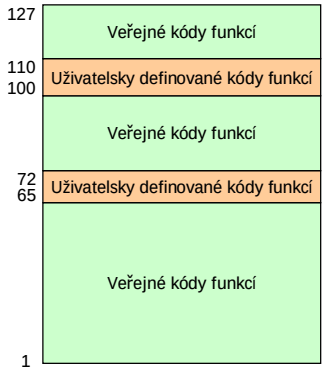
- Datový model je dán tabulkami jednobitových a 16 bitových (word) veličin:

Tabulky	Typ	Přístup
Logické vstupy (Discrete inputs) *	1 bit	Jen čtení (RO)
Cívky (Coils) **	1 bit	Čtení-zápis (RW)
Vstupní registry (Input Registers) *	16 bitů	Jen čtení (RO)
Přídržné registry (Holding registers) **	16 bitů	Čtení-zápis (RW)

- *) Hodnoty mohou být poskytovány vstupním subsystémem
- **) Hodnoty mohou být měněny z aplikačního programu
- Pro každou tabulku dovoluje Modbus volbu až 65536 hodnot
- Rozvržení tabulek v paměti je záležitostí konkrétní implementace
  - Jednotlivé tabulky mohou být odděleny
  - Mohou se však i překrývat

Kategorie kódů funkcí

- Veřejné kódy funkcí (public function codes)
  - Veřejně zdokumentované, jedinečné kódy
- Uživatelsky definované kódy (user defined function codes)
  - Uživatel si může zvolit a implementovat kódy, které nejsou definovány specifikací
  - Existují dva intervaly uživatelských kódů: od 65 do 72 a od 100 do 110
  - Není zaručena jedinečnost kódu



Kódy funkcí pro přístup k datům (Data Access)

Přístup	Typ veličin	Funkce	Kód
1 bitový přístup	Fyzické diskrétní vstupy	Read Discrete Inputs	02
	Interní bitové nebo fyzické výstupy (Coils)	Read Coils	01
		Write Single Coil	05
		Write Multiple Coils	15
16 bitový přístup	Fyzické vstupní registry	Read Input Register	04
	Interní registry nebo fyzické výstupní registry	Read Holding Registers	03
		Write Single Register	06
		Write Multiple Registers	16
		Read/Write Multiple Registers	23
		Mask Write Registers	22
		Read FIFO Queue	24
File record access		Read File Record	20
		Write File Record	21

25

Kódy funkcí pro diagnostiku a ostatní

Oblast	Funkce	Kód	Subkód
Diagnostika	Read Exception Status	07	
	Diagnostic	08	00-18, 20
	Get Com Event Counter	11	
	Get Com Event Log	12	
	Report Slave ID	17	
	Read Device Identification	43	14
Ostatní	Encapsulated Interface Transport	43	13, 14
	CANopen General Reference	43	13

Následující snímky představují několik funkcí protokolu Modbus

26

Funkce 01 – Read Coils

- Požadavek (Request)

Kód funkce	1 bajt	0x01
Počáteční adresa	2 bajty	0x0000 až 0xFFFF
Počet cívek (coils)	2 bajty	1 až 2000 (0x7D0)

- Odpověď (Response)

Kód funkce	1 bajt	0x01
Počet bajtů	1 bajt	N=(Počet cívek + 7) DIV 8
Stav cívek (coils)	N bajtů	

- Chyba

Kód funkce	1 bajt	0x81
Kód výjimky	1 bajt	01 nebo 02 nebo 03 nebo 04

- Příklad
  - Čtení diskrétních (bitových vstupů) č. 20 až 38
  - Požadavek: `0x01 0x00 0x13 0x00 0x13`
  - Odpověď: `0x01 0x03 0xCD 0x6B 0x05`

27

Funkce 02 – Read Discrete Inputs

- Požadavek (Request)

Kód funkce	1 bajt	0x02
Počáteční adresa	2 bajty	0x0000 až 0xFFFF
Počet vstupů	2 bajty	1 až 2000 (0x7D0)

- Odpověď (Response)

Kód funkce	1 bajt	0x02
Počet bajtů	1 bajt	N=(Počet vstupů + 7) DIV 8
Stav cívek (coils)	N bajtů	

- Chyba

Kód funkce	1 bajt	0x82
Kód výjimky	1 bajt	01 nebo 02 nebo 03 nebo 04

28

Funkce 03 – Read Holding Registers

- Požadavek (Request)

Kód funkce	1 bajt	0x03
Počáteční adresa	2 bajty	0x0000 až 0xFFFF
Počet vstupů = N	2 bajty	1 až 125 (0x7D)

- Odpověď (Response)

Kód funkce	1 bajt	0x03
Počet bajtů	1 bajt	2xN
Hodnoty registrů	2xN bajtů	

- Chyba

Kód funkce	1 bajt	0x83
Kód výjimky	1 bajt	01 nebo 02 nebo 03 nebo 04

- Příklad
  - Čtení registrů č. 108 až 110
  - Požadavek: `0x03 0x00 0x6B 0x00 0x03`
  - Odpověď: `0x03 0x06 0x02 0x2B 0x00 0x00 0x00 0x64`
  - Výsledek: Reg. 108 má hodnotu 555, reg. 109 má hodnotu 0 a reg. 110 má hodnotu 100

29

Funkce 06 – Write Single Register

- Požadavek (Request)

Kód funkce	1 bajt	0x06
Adresa registru	2 bajty	0x0000 až 0xFFFF
Hodnota registru	2 bajty	0x0000 až 0xFFFF

- Odpověď (Response)

Kód funkce	1 bajt	0x06
Adresa registru	1 bajt	0x0000 až 0xFFFF
Hodnota registru	2 bajty	0x0000 až 0xFFFF

- Chyba

Kód funkce	1 bajt	0x86
Kód výjimky	1 bajt	01 nebo 02 nebo 03 nebo 04

30

Funkce 16 – Write Multiple Registers

- Požadavek (Request)

Kód funkce	1 bajt	0x10
Počáteční adresa	2 bajty	0x0000 až 0xFFFF
Počet registrů = N	2 bajty	0x0000 až 0x007B
Počet bajtů	1 bajt	2xN
Hodnoty registrů	2xN bajtů	hodnota

- Odpověď (Response)

Kód funkce	1 bajt	0x10
Počáteční adresa	2 bajty	0x0000 až 0xFFFF
Počet registrů	2 bajty	1 až 123 (0x7B)

- Chyba

Kód funkce	1 bajt	0x90
Kód výjimky	1 bajt	01 nebo 02 nebo 03 nebo 04

31

Kódy chyb

32