

Master/slave protokoli višeg reda MODBUS, PROFIBUS i CANBUS

Industrijski sistemi i protokoli -
Drugi deo – predavanja 2 i 3

04/2024

prof. Darko Marčetić

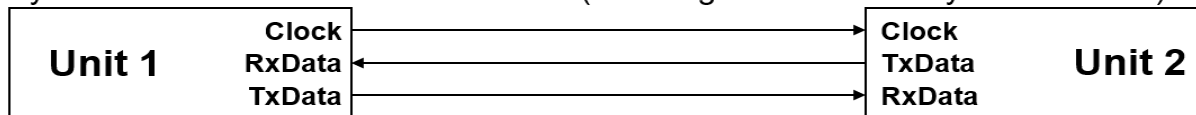
Uvod u protokole višeg reda

- ❑ Većina industrijski protokola višeg reda su slični
- ❑ Baziraju se na asinhronom serijskoj vezi,
- ❑ Predviđaju linijsku veza između uređaja (**magistrala, bus**) na koju se povezuju (kače) redom svi uređaji koji učestvuju (**multidrop**)
- ❑ Podržavaju **half-duplex komunikaciju** (dvosmerni prenos podataka ali ne u isto vreme)
- ❑ Protokoli su uglavnom **master/slave tipa**,
 - **jedan uređaj je master** i samo on može da započne prenos slanjem upitnika ili neke komande.
 - **Svi ostali uređaji su slave i slušaju poruke na liniji.**
 - Na poruku master uređaja **javlja se samo prozvani slave** i odgovara masteru.

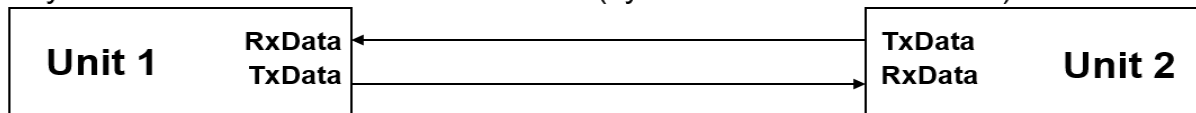
Uvod u protokole višeg reda

1. Industrijski protokoli se baziraju na asinhronom serijskom prenosu

- Synchronous Serial Data Transmission (clock signal is used for synchronization)



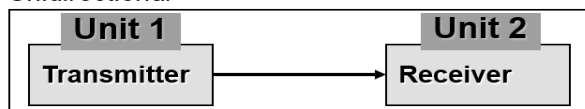
- Asynchronous Serial Data Transmission (synchronization via start bit)



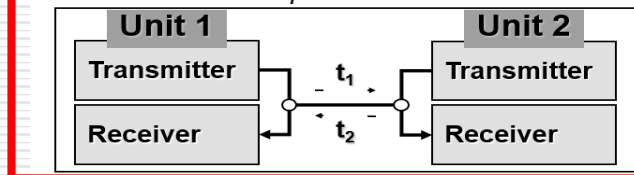
Postoje samo linije za podatke, **nema linije koja prenosi clock signal**
Uvek postoji neki START bit na početku prenosa jednog bajta poruke

2. Industrijski protokoli su half-duplex

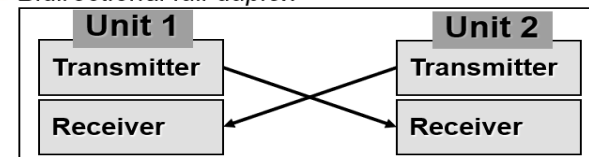
- Unidirectional



- Bidirectional half duplex



- Bidirectional full duplex



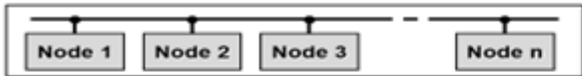
Tokom t_1 priča Unit 1, a tokom t_2 Unit 2

- ❑ **Half-duplex štedi novac, a dovoljan je** za većinu tipova komunikacije
- ❑ Svi čvorovi mogu i da „pričaju“ (Transmitter mode) i da „slušaju“ (Receiver mode), ali u jednom trenutku samo jedan čvor može da priča
- ❑ Ovo ograničenje dovodi do **potrebe za jasnim dogovorom ko kada sme da priča** (protokolom pristupa liniji) Druga opcija je kolizija i arbitraža

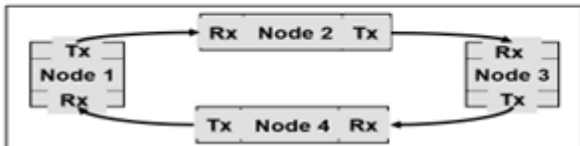
Uvod u protokole višeg reda

3. Industrijski protokoli koriste topologiju linijske veze

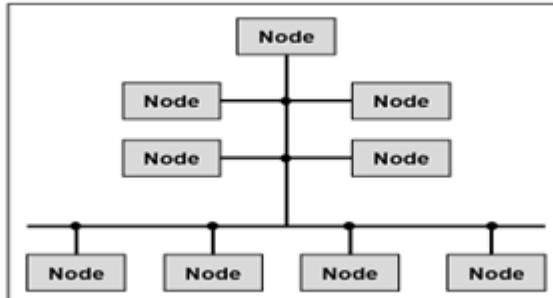
Linearna veza



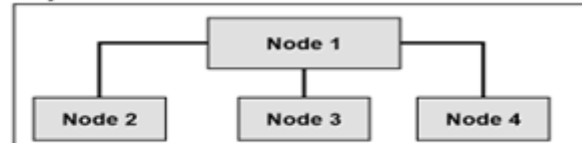
Prsten



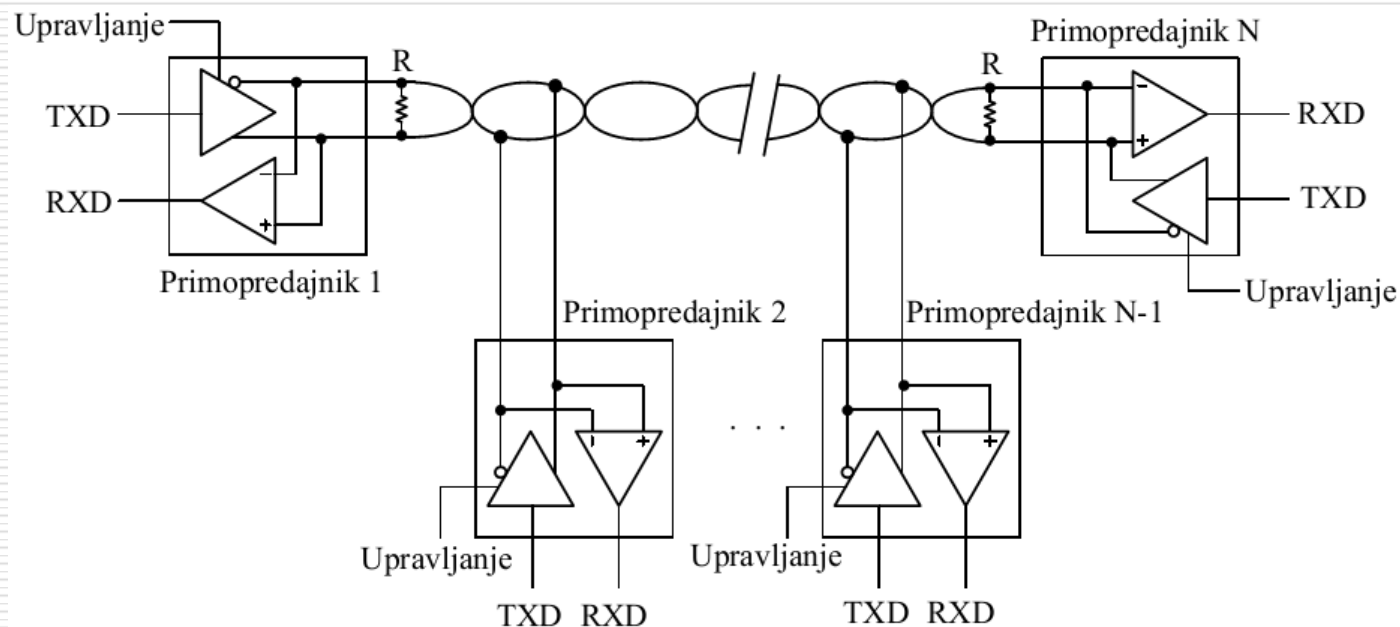
Stablo



Zvezda



❑ Multidrop linija na koju se povezuju svi akteri kao primopredajnici



Uvod u protokole višeg reda

4. Master – slave princip komunikacije

Industrijski protokoli višeg reda rešavaju problem komunikacije na half-duplex liniji primenom master-slave principa komunikacije

U industriji uvek postoji neki uređaj koji kontroliše ceo proizvodni proces ili neki njegov deo. Takođe, postoje uređaji tipa senzora koji mere parametre procesa i time pomažu uređaju koji ga kontroliše. Konačno, postoje i uređaji tipa aktuatora koji na osnovu komandi kontrolnog uređaja upravljaju nekim delom procesa.

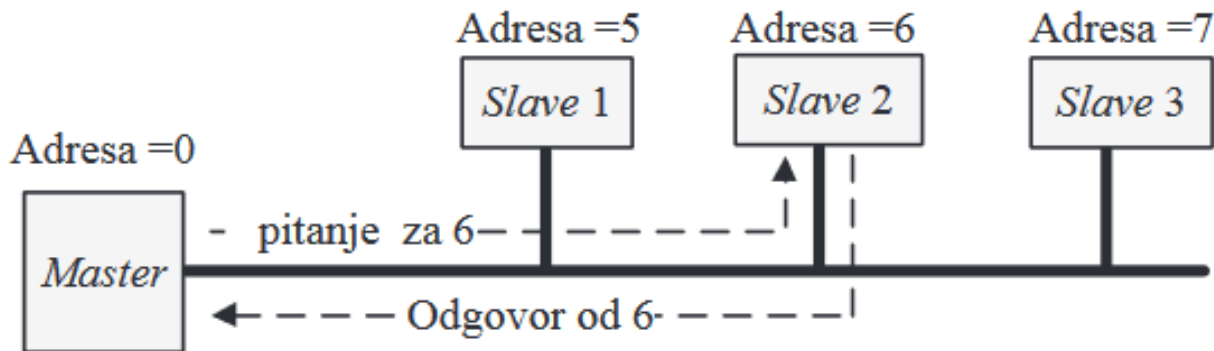
Komunikaciona mreža i protokol između ovih uređaja moraju da uvažavaju činjenicu da učesnici u kontroli procesa, a samim tim i učesnici u razmeni podataka nisu ravnopravni:

- Postoji samo jedan nadređeni (*master*) uređaj, koji prima informacije od senzora, ali i šalje komande ka aktuatorima. On je nadređeni uređaj često po pitanju procesa, ali obavezno po pitanju komunikacije. Nema potrebe da bilo ko inicira komunikaciju osim *master* uređaja. On jedini koristi podatke, tako da će on prozvati neki senzor ako mu zatrebaju neka merenja. Takođe, on upravlja procesom i prozvaće odgovarajući aktuator ako zaželi da promeni stanje njegovih izlaza i time utiče na proces.
- Svi ostali su podređeni (*slave*) uređaji, često po pitanju kontrole procesa, ali obavezno po pitanju komunikacije. Oni su svi u službi *master* uređaja, i praktično su njegovi robovi (*slave*). *Slave* uređaji nikad ne počinju komunikaciju, ali zato uvek odgovaraju na *master* poruku i izvršavaju naredbu koju im je *master* tom porukom poslao.

Ovi principi su osnova za izgradnju *master-slave* komunikacionog protokola.

Uvod u protokole višeg reda

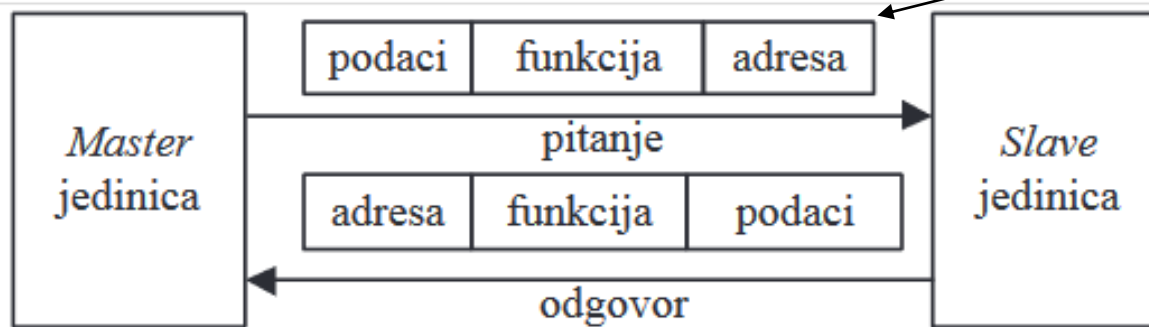
- ❑ Tipična konfiguracija industrijske mreže master-slave tipa
Primer je dat za jedan master i tri slave uređaja.



Slika 9.1 Tipična konfiguracija industrijske mreže

- U datom primeru:
1. Master šalje pitanje za slave br 6
 2. Slave br 6 odgovara

- ❑ Svi protokoli višeg reda definišu i strukturu telegrama koji se koristi za razmenu informacija između master i slave uređaja.



Slika 9.2 Tipična struktura master pitanja i slave odgovora

Uvod u protokole višeg reda

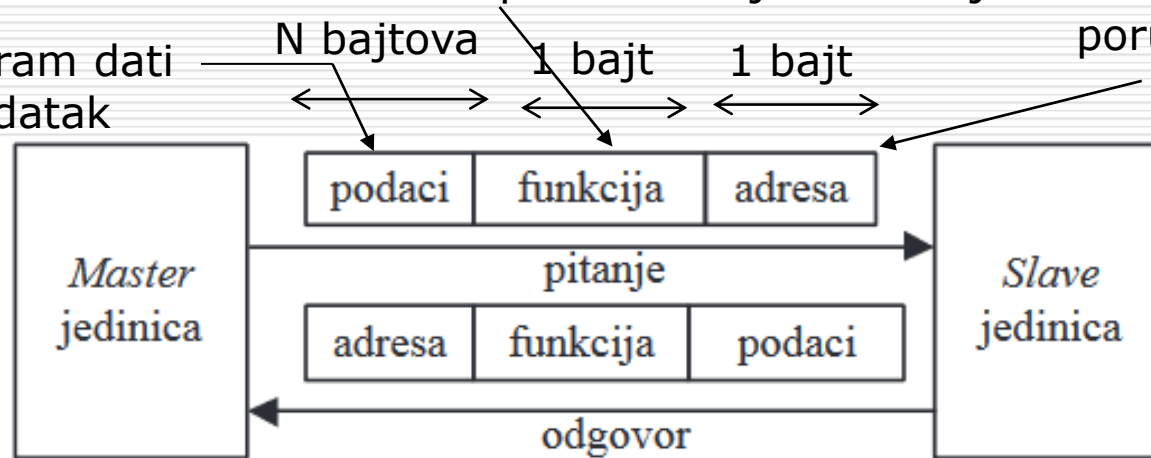
Primer protokola višeg reda (Marčetić protokol)

- ❑ Protokol koristi strukturu telegrama definisanu prethodnom slikom,
- ❑ protokol rezerviše **jedan bajt u polju adresa za adresu slave uređaja**,
- ❑ protokol rezerviše **jedan bajt u polju za funkciju** koju slave uređaj treba da izvrši.
- ❑ **Podržane su samo dve funkcije: 0 – čitanje, 1 – pisanje**,
- ❑ protokol rezerviše dva bajta u polju za podatke master telegrama, prvi bajt je adresa u slave memoriji (tabeli) koju master čita /piše drugi bajt je broj koji se piše (šalje se samo u slučaju funkcije pisanja),
- ❑ protokol rezerviše jedan bajt u polju za podatke slave telegrama, bajt se popunjava samo u slučaju odgovora na funkciju čitanja.

- Ako čitam moram ti dati adresu
- Ako pišem moram dati adresu ali i podatak

0 – hoću da čitam iz tvoje memorije
1- hoću da pišem u tvoju memoriju

Master u prvom bajtu šalje adresu (broj) slave-a kome je poruka namenjena



Slika 9.2 Tipična struktura master pitanja i slave odgovora

Uvod u protokole višeg reda

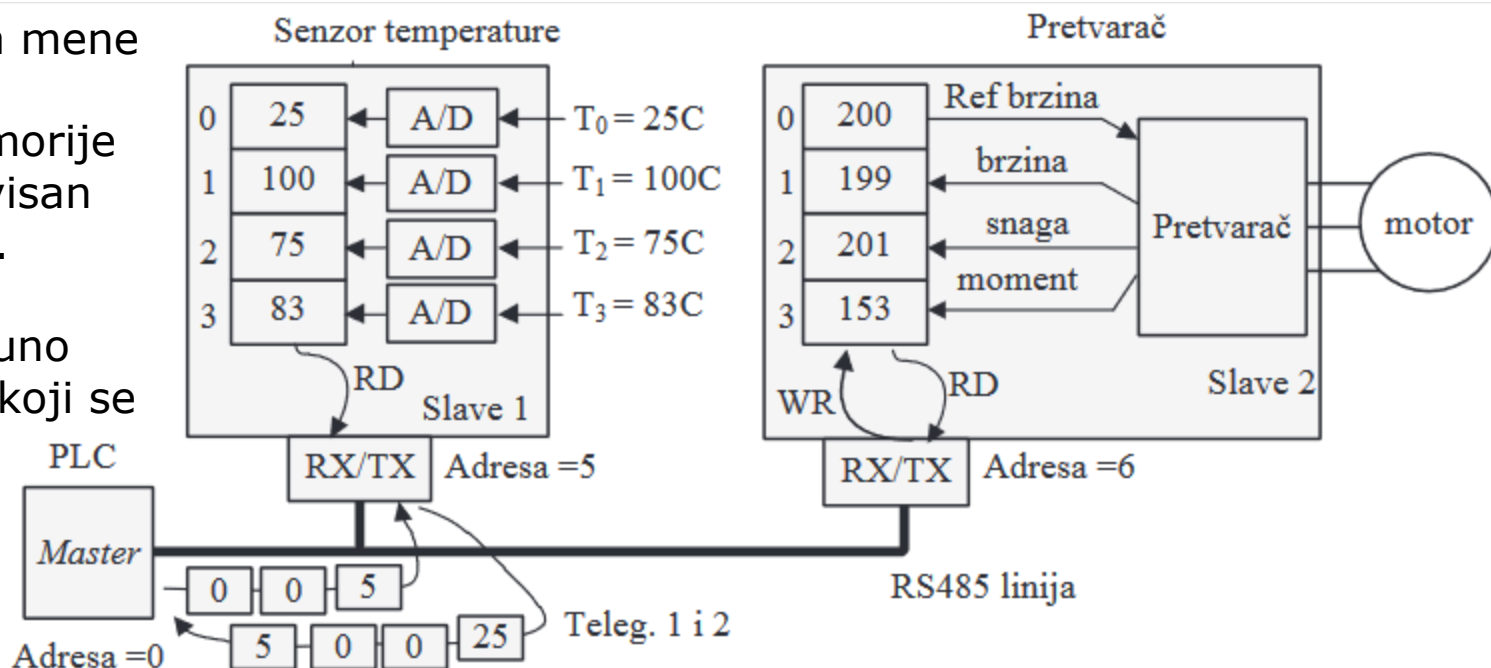
Primer protokola višeg reda (Marčetić protokol)

- ❑ Protokol koristi strukturu telegrama definisanu prethodnom slikom,
- ❑ protokol rezerviše **jedan bajt u polju adresa za adresu slave uređaja**,
- ❑ protokol rezerviše **jedan bajt u polju za funkciju** koju slave uređaj treba da izvrši.
- ❑ **Podržane su samo dve funkcije: 0 – čitanje, 1 – pisanje**,
- ❑ protokol rezerviše dva bajta u polju za podatke master telegrama, prvi bajt je adresa u slave memoriji (tabeli) koju master čita /piše drugi bajt je broj koji se piše (šalje se samo u slučaju funkcije pisanja),
- ❑ protokol rezerviše jedan bajt u polju za podatke slave telegrama, bajt se popunjava samo u slučaju odgovora na funkciju čitanja.

Svaki slave je za mene samo tablica

Deo interne memorije slave-a je rezervisan za komunikaciju.

Primer dva potpuno različita slave-a koji se slično tretiraju



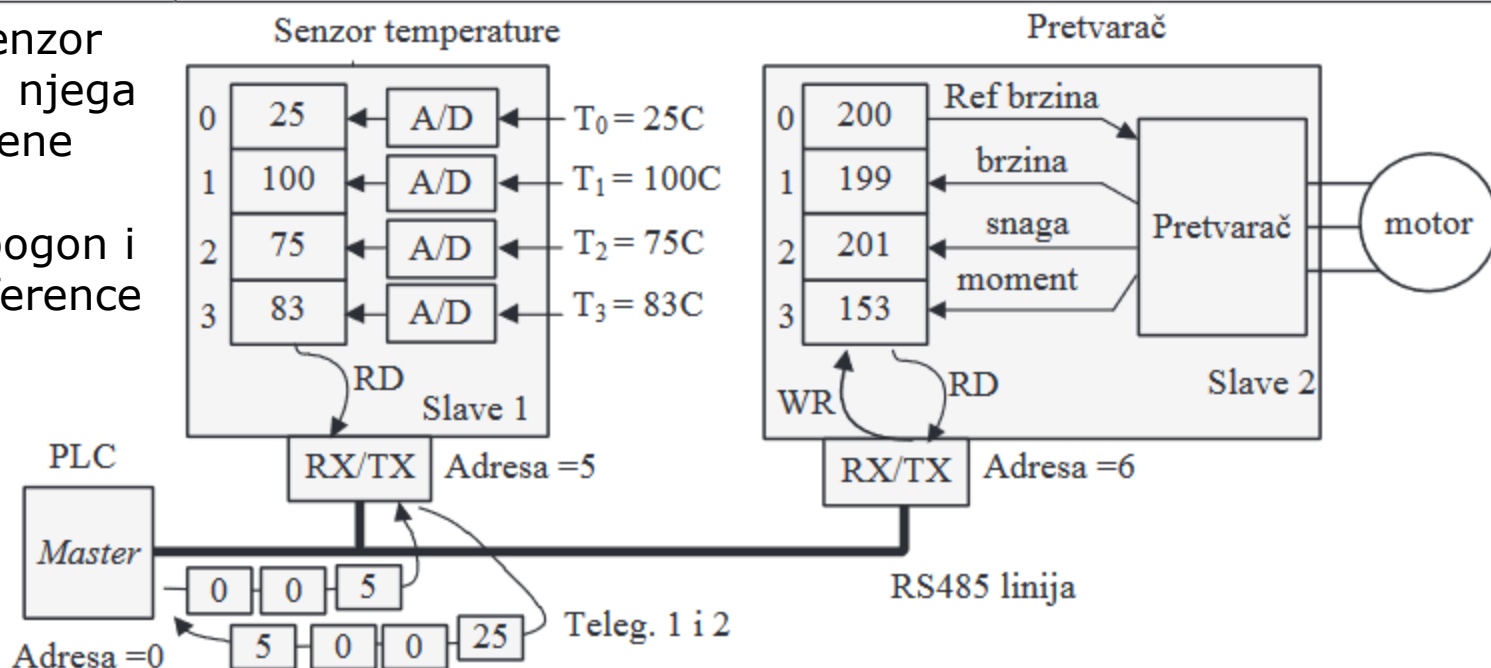
Uvod u protokole višeg reda

Primer poruka na liniji (Marčetić protokol)

Broj	Ko šalje	Poruka	Objašnjenje poruke i akcija
1	M	5 0 0	M adresira S1 (adresa 5), želi da čita (0), adresu 0 (0), na kojoj je vrednost T_0
2	S1	5 0 0 25	S1 odgovara (5), izvršeno čitanje (0), na adresi 0 (0) mi je broj 25
3	M	5 0 1	M adresira S1 (adresa 5), želi da čita (0), adresu 1 (1), na kojoj je vrednost T_1
4	S1	5 0 1 100	S1 odgovara (5), izvršeno čitanje (0), na adresi 1 (1) mi je broj 100
5	M	6 1 0 200	M adresira S2(6), želi da piše (1), na adresu (0), broj 200, time menja ref. brzinu
6	S2	6 1 0 200	S2 (6) potvrđuje da je upisan (1) na adresu (0) broj 200.
7	M	6 0 1	M adresira S2 (6), želi da čita (0), adresu 1, tj. da pročitava stvarnu brzinu motora
8	S2	6 0 1 199	S2 (6), izvršeno čitanje (0), na adresi 1 je 199, a zna se da je tu stvarna brzina

Slave ID=5 je senzor temperature i sa njega samo čitam merene vrednosti.

Slave ID =6 je pogon i njemu pišem reference a čitam merene vrednosti.



Modbus protokol

- ❑ **Primena u elektroenergetskom sistemu za prenos**
- ❑ **Master/slave tip protokola,**
- ❑ **Half-duplex** (dvosmerna ali ne i istovremena komunikacija)
- ❑ razvijen je od strane **Modicon-a, 1979 godine**
- ❑ Više detalja na **<http://www.modbus.org/>**

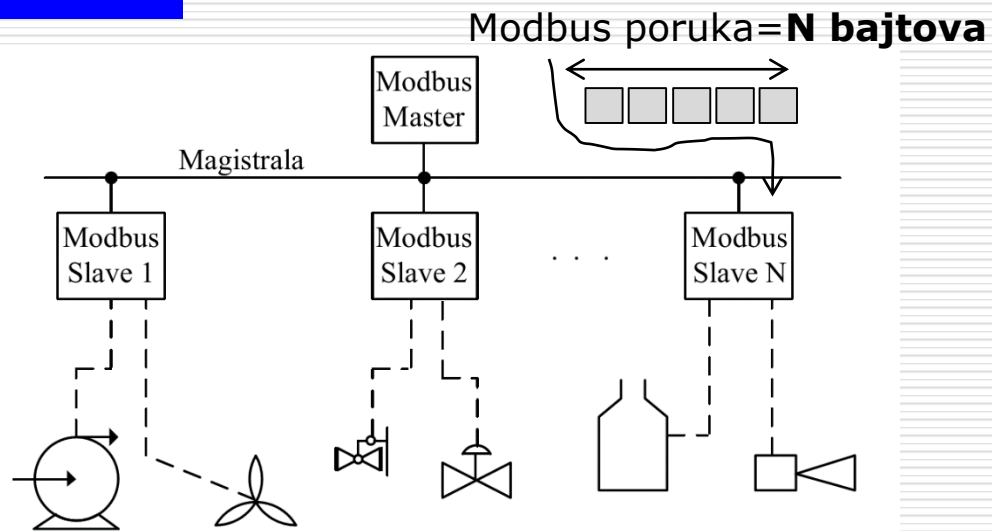
Struktura Modbus mreže (fizički sloj)

MODBUS fizička podloga

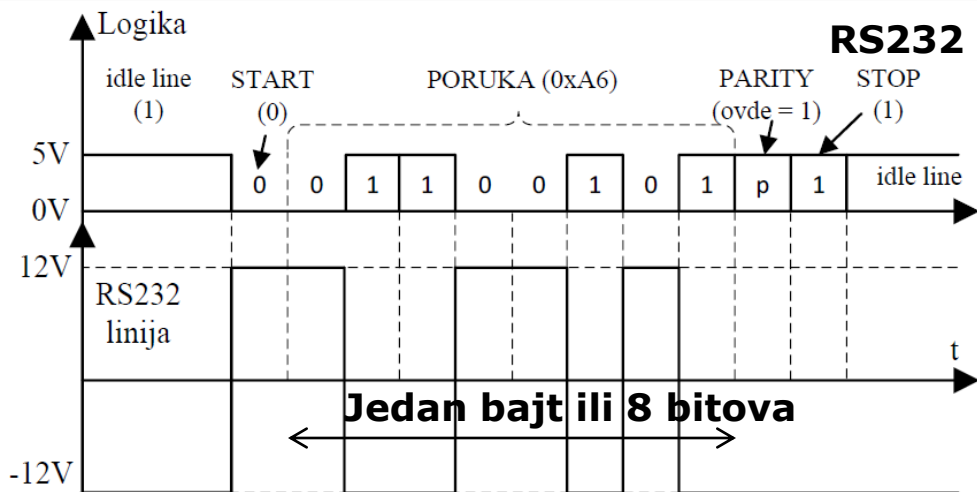
Fizička veza između svih aktera komunikacije.

Ovo je najniži sloj protokola koji treba da definiše kako se prenose bitovi/bajtovi poruke:

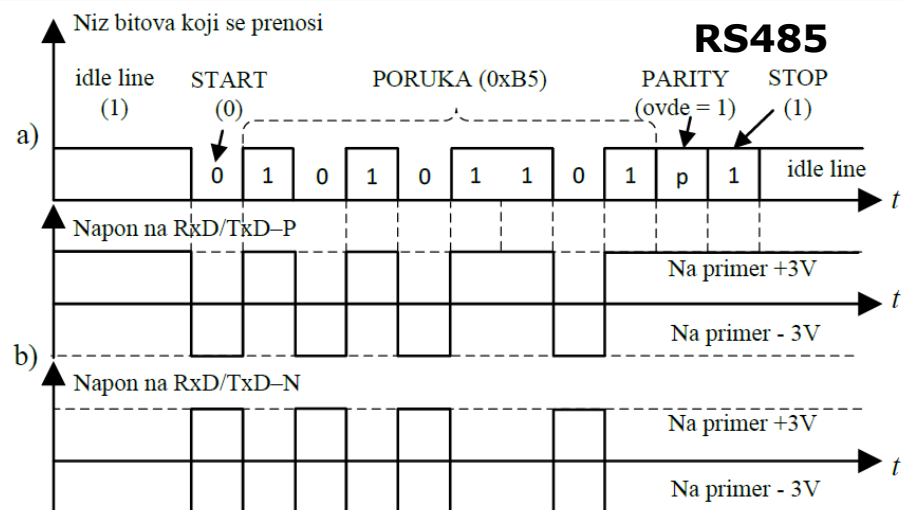
- ❑ Poruka se sastoji iz više bajtova,
- ❑ a bajtovi od 8 bitova informacije



Originalni Modbus za slanje bajta poruke koristi RS232 ili RS485 protokol. Time je sve definisano: naponski nivoi, konektori, da se bajt prenosi sa 10 ili 11 bitova: START+8 bitova informacije+ PARITY(opciono) +STOP



Modbus/Profibus/Canbus protokoli

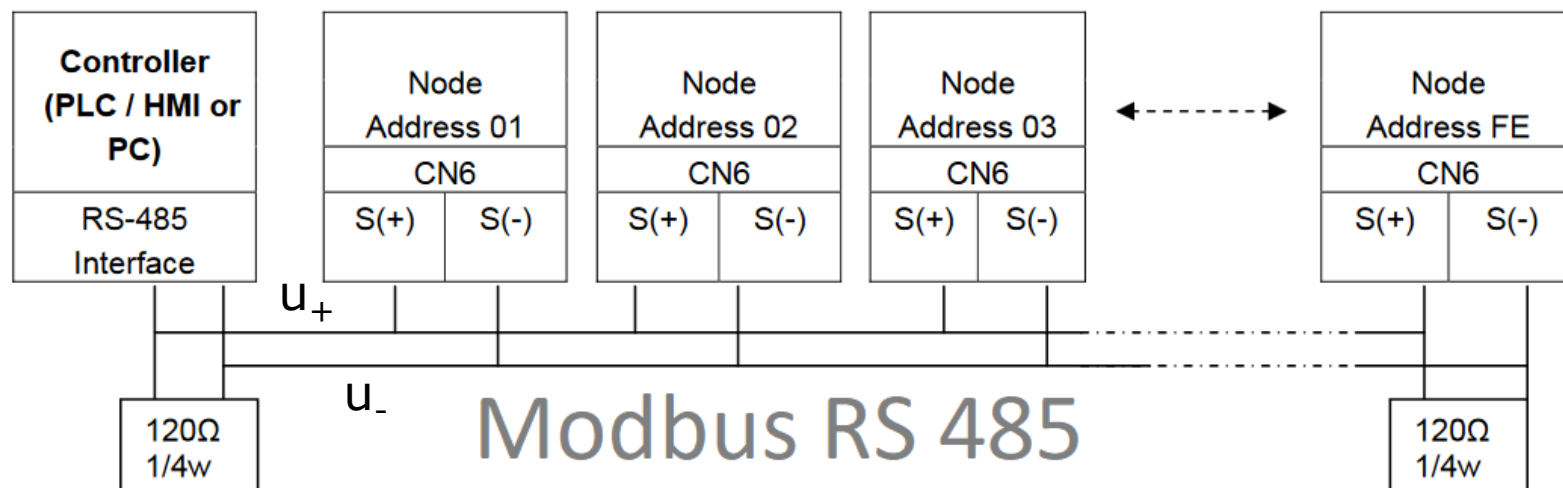


Struktura Modbus RS485 mreže (fizički sloj)

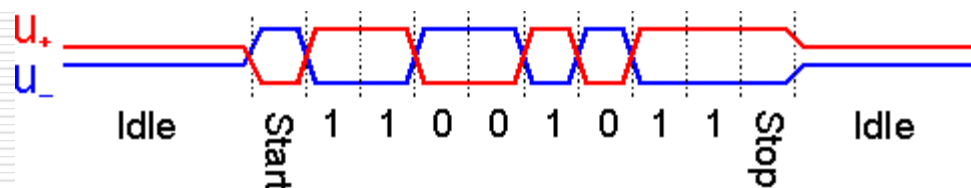
Modbus RS485 se danas skoro jedino koristi !!!

Struktura linije u slučaju RS485

- ❑ Jedan master i više slave jedinica se vezuju sa dve žice na koje se svi kače (multidrop linija)
- ❑ Na liniji se podaci prenose diferencijalno,
- ❑ Svi uređaji su diferencijalni primopredajnici,
- ❑ Samo jedan uređaj može da priča u jednom trenutku (half-duplex), zato master uvek počinje komunikaciju a samo prozvani slave odgovara
- ❑ Uređaji na krajevima linije moraju imati terminacioni otpornik 120 Ohm



Jedan bajt Modbus poruke po RS485 standardu se prenosi diferencijalno, na dve linije



Struktura Modbus poruka (nivo podataka)

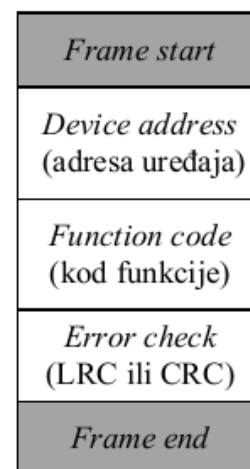
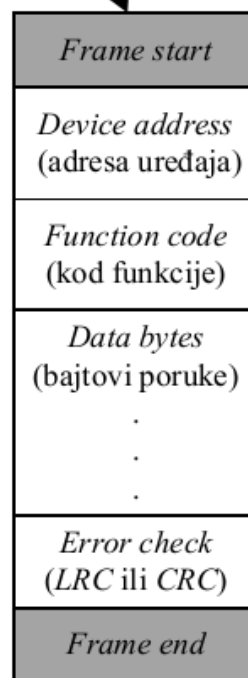
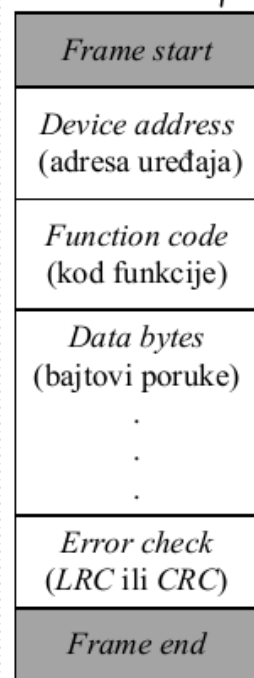
Modbus definiše strukturu poruka tj. skup bajtova koji čine poruku.

Ovo se često naziva nivo podataka jednog protokola (koji se naslanja na fizički nivo i koristi njegove usluge da prenese pojedine bajtove poruke)

Predviđene su dve vrste poruka

- master upit (pitalicu, query) koji svi slušaju i
- slave odgovor (response) koji šalje samo prozvani slave

1) Query poruka od mastera



2b) ili slave samo šalje potvrdu prijema (acknowledgment)

2a) Slave response (odgovor)

Pun odgovor prozvanog slave uređaja

Kratak odgovor (samo potvrda da je urađeno)

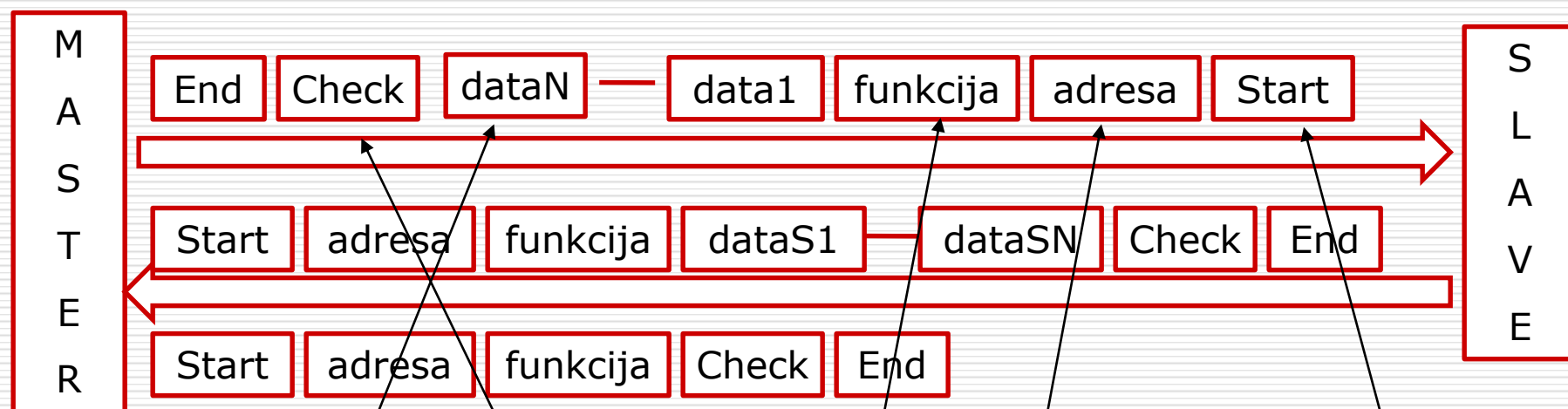
Prvo ide master pitanje

- Kome ?
(device address)
- Šta da radi ?
(function code)
- Sa čime da radi
(data bytes)

Struktura Modbus poruka

Kako se vrši master/slave razmena poruka po Modbus protokolu

- Pošto je half duplex veza samo jedan uređaj može da priča,
- **Zato master prvi šalje upit (query), a onda slave šalje odgovor**



NIZ PODATAKA
bajtova sa podacima zavisi od funkcije, npr ako je funkcija pisanje u slave sledi interna adresa u slejvu na koju se piše i broj koji master hoće da upiše

FUNKCIJSKI bajt
da prozvani rob zna šta da radi

ADRESNI bajt
da se zna kome je namenjena

START bajt
da se zna da je poruka krenula (nije obavezan)

CHECKSUM PODATAKA
bajtovi za proveru ispravnosti pristigle poruke

Modbus podržava ASCII i binarni prenos brojeva

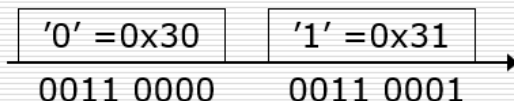
Ceo slajd informativno

Modbus ASCII i Modbus RTU (binarni) se razlikuju po

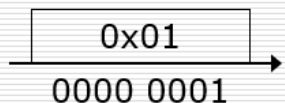
- načinu slanja jednog broja
 - Binarno (RTU), jedan bajt jedan broj, 0 -255
 - ASCII, dva ASCII karaktera za broj 00 - FF
- Po različitim start, stop i poljima greške u Modbus poruci

Primer za slanje broja 1

Po ASCII prenosu za broj 1 idu dva heksadecimalna karaktera broj 1d = 0x01 = '0' i '1'



Kod binarnog prenosa jedan broj je jedan bajt, tako da za broj 1d = 0000 0001b



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	MUL (null)	32	20	040	#32;	Space	64	40	100	#64;	@	96	60	140	#96;	`
1	1	001	SOH (start of heading)	33	21	041	#33;	!	65	41	101	#65;	A	97	61	141	#97;	a
2	2	002	STX (start of text)	34	22	042	#34;	"	66	42	102	#66;	B	98	62	142	#98;	b
3	3	003	ETX (end of text)	35	23	043	#35;	#	67	43	103	#67;	C	99	63	143	#99;	c
4	4	004	EOT (end of transmission)	36	24	044	#36;	\$	68	44	104	#68;	D	100	64	144	#100;	d
5	5	005	ENQ (enquiry)	37	25	045	#37;	%	69	45	105	#69;	E	101	65	145	#101;	e
6	6	006	ACK (acknowledge)	38	26	046	#38;	&	70	46	106	#70;	F	102	66	146	#102;	f
7	7	007	BEL (bell)	39	27	047	#39;	'	71	47	107	#71;	G	103	67	147	#103;	g
8	8	010	BS (backspace)	40	28	050	#40;	(	72	48	110	#72;	H	104	68	150	#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	#41;	)	73	49	111	#73;	I	105	69	151	#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	#42;	*	74	4A	112	#74;	J	106	6A	152	#106;	j
11	B	013	VT (vertical tab)	43	2B	053	#43;	+	75	4B	113	#75;	K	107	6B	153	#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	#44;	,	76	4C	114	#76;	L	108	6C	154	#108;	l
13	D	015	CR (carriage return)	45	2D	055	#45;	-	77	4D	115	#77;	M	109	6D	155	#109;	m
14	E	016	SO (shift out)	46	2E	056	#46;	.	78	4E	116	#78;	N	110	6E	156	#110;	n
15	F	017	SI (shift in)	47	2F	057	#47;	/	79	4F	117	#79;	O	111	6F	157	#111;	o
16	10	020	DLE (data link escape)	48	30	060	#48;	0	80	50	120	#80;	P	112	70	160	#112;	p
17	11	021	DC1 (device control 1)	49	31	061	#49;	1	81	51	121	#81;	Q	113	71	161	#113;	q
18	12	022	DC2 (device control 2)	50	32	062	#50;	2	82	52	122	#82;	R	114	72	162	#114;	r
19	13	023	DC3 (device control 3)	51	33	063	#51;	3	83	53	123	#83;	S	115	73	163	#115;	s
20	14	024	DC4 (device control 4)	52	34	064	#52;	4	84	54	124	#84;	T	116	74	164	#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	#53;	5	85	55	125	#85;	U	117	75	165	#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	#54;	6	86	56	126	#86;	V	118	76	166	#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	#55;	7	87	57	127	#87;	W	119	77	167	#119;	w
24	18	030	CAN (cancel)	56	38	070	#56;	8	88	58	130	#88;	X	120	78	170	#120;	x
25	19	031	EM (end of medium)	57	39	071	#57;	9	89	59	131	#89;	Y	121	79	171	#121;	y
26	1A	032	SUB (substitute)	58	3A	072	#58;	:	90	5A	132	#90;	Z	122	7A	172	#122;	z
27	1B	033	ESC (escape)	59	3B	073	#59;	;	91	5B	133	#91;	[	123	7B	173	#123;	{
28	1C	034	FS (file separator)	60	3C	074	#60;	<	92	5C	134	#92;	\	124	7C	174	#124;	
29	1D	035	GS (group separator)	61	3D	075	#61;	=	93	5D	135	#93;	]	125	7D	175	#125;	}
30	1E	036	RS (record separator)	62	3E	076	#62;	>	94	5E	136	#94;	^	126	7E	176	#126;	~
31	1F	037	US (unit separator)	63	3F	077	#63;	?	95	5F	137	#95;	_	127	7F	177	#127;	DEL

	Modbus/ASCII	Modbus/RTU
Tip karaktera	ASCII 0...9 and A..F	Binarni brojevi 0...255
Frame start	character ':'	3.5 chars silence (tišine)
Frame end	Characters CR/LF	3.5 chars silence (tišine)
Dozvoljena pauza	1 sec	1.5 times char length (1.5 trajanja karaktera)
Start bit	1	1
Data bitovi	7	8
Pariti bit	even/odd ili nema	even/odd ili nema
Stop bit	1 ili 2	1 ili 2
Tip polja za proveru greške	LRC Longitudinal Redundancy Check (8 bita)	CRC Cyclic Redundancy Check (16 bitova)

Detekcija greške Modbus poruke

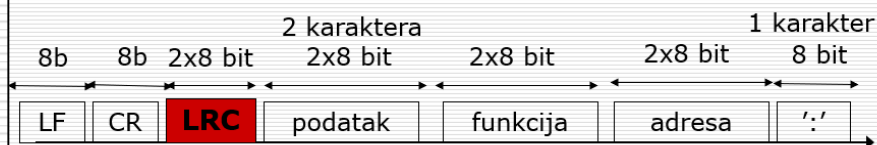
Na kraju Modbus poruke jedan ili dva bajta su rezervisani za proveru greške prenosa, popularno **Checksum polje** poruke. Oni nose vrednost koja je neka matem/log kombinacija svih brojeva poruke:

- **Računa na predaji na osnovu poslaih bajtova i utiskuje u poruku.**
- **Računa na prijemu na osnovu primljenih bajtova**
- **Prijemnik poredi dve checksum vrednosti i ispituje ima li greške.**

ASCII mod ima jedan checksum bajt, **Longitudinal Redundancy Check (LRC)** koji se računa :

- saberi sve bajtove u poruci, bez start bajta ':' i bez završnog CRLF.
- Bajtove sabirati u 8-bitnoj aritmetici, prenos ignorisati,
- Napravi II komplement dobijenog broja $(255 - x) + 1$

Račun samo informativno



U RTU modu se koriste dva **Cyclic Redundancy check (CRC)** bajta koji se računaju:

1. Postavlja se početna **crc** = 0xFFFF
 2. Zatim se vrši XOR nad **crc** i bajtom poruke,
 3. Ukoliko je LSB = 1 sadržaj **crc** se pomera udesno i opet se vrši XOR ccr sa unapred definisanim brojem (na primer 0xA001)
- Ukoliko je LSB = 0 sadržaj **crc** registra se pomera za jedno mesto udesno i u najviši bit (MSB) se upisuje nula,

Operacije se ponavljaju 8 puta, za svih 8 bitova jednog bajta

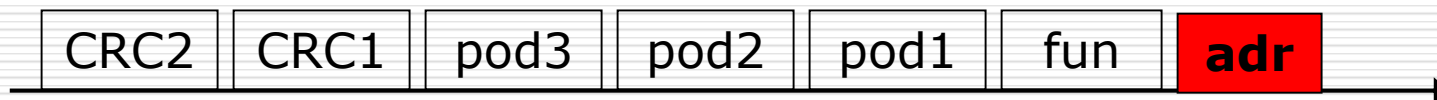
Zatim se sve to ponavlja za sve bajtove poruke i konačna **crc** se upisuje u 16-bitno polje.

Račun samo informativno



Adresiranje u Modbus sistemu

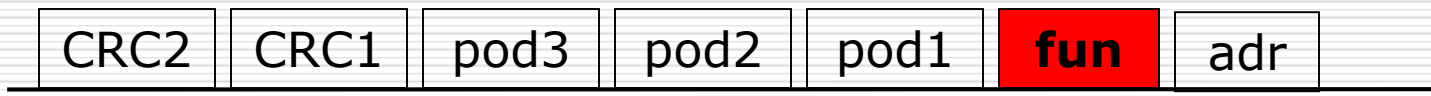
- ❑ **Prvi bajt u okviru Modbus poruke je adresa prijemnika.** Ovo je uvek samo jedan 8-bitni podatak, često nazivan **slave ID byte**.
- ❑ Svakom slave uređaju je dodeljena jedna adresa, **iz opsega 1 to 247**. Na ovaj način već posle prvog pristiglog byte-a poruke slave zna da li je poruka za njega ili treba da je ignoriše.
- ❑ Kada slave odgovara na poruku, on koristi istu (svoju) adresu u odgovoru.
- ❑ **Adresa 0 je rezervisana za broadcast**, i njome se šalje poruka namenjena svime **slave** uređajima na liniji , poruka koju svi primaju ali niko na nju ne odgovara.



Funkcije u Modbus sistemu

Drugi bajt u okviru Modbus poruke je funkcija koju slave treba da izvrši. Ako je slave prozvan, on mora da zna šta od njega tražimo da uradi ili da nam pošalje. Ovu informaciju nosi **function code** polje.

Kodovi MODBUS funkcija		
01	<i>Read coil status</i>	Čitaj stanje izlaznog pina
02	<i>Read input status</i>	Čitaj stanje ulaznog pina
03	<i>Read holding registers</i>	Čitaj stanje unutrašnjeg registra
04	<i>Read input registers</i>	Čitaj stanje ulaznog registra
05	<i>Force single coil</i>	Upiši (promeni) stanje izlaznog pina
06	<i>Preset single register</i>	Upiši stanje registra
07	<i>Read exception status</i>	Pročitaj status
15	<i>Force multiple coils</i>	Upiši (promeni) više izlaznih pinova
16	<i>Preset multiple registers</i>	Upiši vrednosti u više registara
17	<i>Report slave ID</i>	Javi adresu slave jedinice



Modbus funkcija 01 - Read coil status

Funkcija za čitanje bitova unutar slave jedinice (bit = coil)

❑ Svaki Modbus slave ima bit polje, u kome svaki bit ima adresu.

Master šalje poruku:

Function 01 - struktura <i>master query</i> poruke		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	1	<i>Function code</i>
3	0...255	<i>Starting address, high byte</i>
4	0...255	<i>Starting address, low byte</i>
5	0...255	<i>Number of coils, high byte</i>
6	0...255	<i>Number of coils, low byte</i>
7(...8)	<i>LRC/CRC</i>	<i>Error check value</i>

Predviđeno 2 bajta za adresu prvog bita unutar slave jedinice koji želimo da čitamo

Predviđeno 2 bajta da definišu koliko bitova se odjednom čita

Odgovor slave jedinice :

Function 01 - struktura <i>slave answer</i> poruke		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	1	<i>Function code</i>
3	0...255	<i>Number of data bytes N</i>
4...N+3	0...255	<i>Bit pattern of coil values</i>
N+4(...N+5)	<i>LRC/CRC</i>	<i>Error check value</i>

Samo `povraća` nazad adresu i funkciju, da master zna da je u redu.

Proračun koliko bajtova treba da se pošalje traženi bitovi, npr za 11 bitova treba 2 bajta

Bajtovi koji nose tražene bitove

Modbus funkcija 01 - primer brojnih vrednosti

Čitanje stanja diskretnih vrednosti izlaza počevši od broja 20 pa sve do broja 56 koji se nalaze na slave jedinici sa adresom 17 (MODBUS/RTU)

Zahtev:

11 01 00 13 00 25 0E 84

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 01 Kôd funkcije (*Read Coil Status*)
- 0013 Adresa na kojoj se nalazi podatak o stanju prvog pina (Coil 20 - 1 = 19 = 13 hex)
- 0025 Ukupan broj pinova/bitova za čitanje (od 20 do 56 = 37 = 25 hex)
- 0E84 CRC za proveru greške

Odgovor:

11 01 05 CD 6B B2 0E 1B 45 E6

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 01 Kôd funkcije (*Read Coil Status*)
- 05 Broj bajtova neophodan za slanje ($37 \text{ Coils} / 8 \text{ bits per byte} = 5 \text{ bytes}$)
- CD *Coils* 27 - 20 (1100 1101)
- 6B *Coils* 35 - 28 (0110 1011)
- B2 *Coils* 43 - 36 (1011 0010)
- 0E *Coils* 51 - 44 (0000 1110)
- 1B 3 *space holders* & *Coils* 56 - 52 (0001 1011)
- 45E6 CRC za proveru greške

Modbus funkcija 03 - Read holding registar

Funkcija za čitanje jednog ili više registra unutar slave jedinice
Master šalje poruku:

Function 03 query structure		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	3	<i>Function code</i>
3	0...255	<i>Starting address, high byte</i>
4	0...255	<i>Starting address, low byte</i>
5	0...255	<i>Number of registers, high byte</i>
6	0...255	<i>Number of registers, low byte</i>
7(...8)	<i>LRC/CRC</i>	<i>Error check value</i>

Predviđeno 2 bajta za adresu prvog registra unutar slave jedinice koji master želi čitati (65536 kombinacija 64K adresa)

Predviđeno 2 bajta da definišu koliko registara se odjednom čita (65536 kombinacija, 64K registara)

Odgovor slave jedinice:

Function 03 - struktura slave answer poruke		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	3	<i>Function code</i>
3	0...255	<i>Number of data bytes N</i>
4	Reg 1 Hig	<i>Vrednost u registru 1 - high byte</i>
5	Reg 1 Low	<i>Vrednost u registru 1 - low byte</i>
6	Reg 2 Hig	<i>Vrednost u registru 2 - high byte</i>
7	Reg 2 Low	<i>Vrednost u registru 2 - low byte</i>
	...	
	<i>LRC/CRC</i>	<i>Error check value</i>

svi registri su 16-bitni, pa za čitanje npr 2 registra slave će poslati 4 bajta

svi registri su 16-bitni, dakle za svaki podatak se očekuje 2 bajta

Modbus funkcija 03 - primer brojnih vrednosti

Čitanje izlaznih holding registara počevši od broja 40108 pa do broja 40110 koji se nalaze na slave jedinici sa adresom 17 (MODBUS/RTU)

Zahtev:

11 03 00 6B 00 03 76 87

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 03 Kôd funkcije (*Read Analog Output Holding Registers*)
- 006B Adresa prvog registra (40108–40001 = 107 = 6B hex)
- 0003 Ukupan broj registara (3 registra od 40108 do 40110)
- 7687 CRC za proveru greške

Odgovor:

11 03 06 AE 41 56 52 43 40 49 AD

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 03 Kôd funkcije (*Read Analog Output Holding Registers*)
- 06 Broj bajtova za prenos (3 registra x 2 bajta svaki = 6 bajtova)
- AE41 Sadržaj registra 40108
- 5652 Sadržaj registra 40109
- 4340 Sadržaj registra 40110
- 49AD CRC za proveru greške

2 bajta
za svaki
16b reg

Ne treba zamarati studente sa time, ali MODBUS je jasno definisao raspon adresa holding registare koji počinje od 40000. Tako da kada se šalje adresa 108 misli se na adresu 40108.

Modbus funkcija 16 - Preset multiple registers

Funkcija za pisanje u jedan ili više registara unutar slave.

Master šalje poruku:

Function 16 query structure		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	16	<i>Function code</i>
3	0...255	<i>Address of first register to write, high byte</i>
4	0...255	<i>Address of first register to write, low byte</i>
5	0...255	<i>Number of registers to write, high byte</i>
6	0...255	<i>Number of registers to write, low byte</i>
7	0...255	<i>Number of bytes to send</i>
8	0...255	<i>Value to wrtie in register 1 , high byte</i>
9	0...255	<i>Value to wrtie in register 1 , low byte</i>
10	0...255	<i>Value to wrtie in register 2 , high byte</i>
11	0...255	<i>Value to wrtie in register 2 , low byte</i>
	...	
	<i>LRC/CRC</i>	<i>Error check value</i>

Master šalje početnu adresu registra čiju vrednost hoće da menja

Zatim šalje koliko ukupno registara hoće da menja

Za svaki registar ide dva bajta, to je nova 16b vrednost koja će biti upisana u taj registar, zatim u sledeći, i tako dalje..

Odgovor slave jedinice:

Function 16 - struktura slave answer poruke		
Bajt	Vrednost	Opis
1	1...247	<i>Slave Device address</i>
2	16	<i>Function code</i>
3	0...255	<i>Address of first register that is changed, high byte</i>
4	0...255	<i>Address of first register that is changed, low byte</i>
5	0...255	<i>Number of changed registers, high byte</i>
6	0...255	<i>Number of changed registers, low byte</i>
	...	
	<i>LRC/CRC</i>	<i>Error check value</i>

Odgovor slave je potvrda da se u N registara počevši od neke adrese upisalo N novih vrednosti.

Modbus funkcija 16 - primer brojnih vrednosti

Master Query funkcije 16 za pisanje jednog ili više registara unutar slave. *MODBUS holding* registri počinju na adresi **40001**.

Zahtev:

11 10 00 01 00 02 04 00 0A 01 02 C6 F0

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 10 Kôd funkcije (*Preset Multiple Registers*, 16 = 10 hex)
- 0001 Adresa prvog registra (#40002-40001 = 1)
- 0002 Ukupan broj registara kojima se menja vrednost
- 04 Broj bajtova za prenos (*2 registers x 2 bytes each = 4 bytes*)
- 000A Vrednost koja se upisuje u registar 40002
- 0102 Vrednost koja se upisuje u registar 40003
- C6F0 CRC za proveru greške

Odgovor:

11 10 00 01 00 02 12 98

- 11 Adresa *slave* jedinice (17 = 11 hex)
- 10 Kôd funkcije (*Preset Multiple Registers*, 16 = 10 hex)
- 0001 Adresa prvog registra (#40002-40001 = 1)
- 0002 Ukupan broj registara kojima je promenjena vrednost
- 1298 CRC za proveru greške

PROFIBUS protokol

- ❑ **PROFIBUS (Process Field Bus)** protokol nastaje 1987. u Nemačkoj
- ❑ Više institucija i firmi započinje projekat bit-serijskog fieldbus sistema i kao rezultat se nameće PROFIBUS kao industrijski standard.
- ❑ PROFIBUS je i deo međunarodnih standarda IEC 61158 i IEC 61784.
- ❑ **Primena u sistemima automatskog upravljanja**
- ❑ **Master/slave tip protokola**
- ❑ **Half-duplex i linijska topologija**
- ❑ Profibus po potrebi **podržava više master uređaja**
- ❑ **Tri tipa PROFIBUS protokola**
 - ❑ **PROFIBUS DP, PROFIBUS PA i PROFINET**

Tipovi PROFIBUS: DP, PA i PROFINet

PROFIBUS DP (Decentral. periferije) se koristi u većini PROFIBUS aplikacija. Protokol je prost i optimiziran je za veoma brzu komunikaciju između PLC ili centralizovanog PC sa više rasprostranjenih ulazno/izlaznih uređaja, aktuatora i senzora na nivou polja. Uglavnom se koristi u proizvodnoj automatizaciji. Danas PROFIBUS DP podržava tri načina komunikacije na liniji i dostupan je u tri funkcijske verzije DP-V0, DP-V1 i DP-V2. Podržava rad i sa više master kontrolera, veliku brzinu prenosa (do 12 Mbit/s) i to preko oklopljene upletene parice (twisted pair) ili preko optičkih kablova.

PROFIBUS PA (Process Automation) je namenjen za jeftinu (dvožičnu) i pouzdanu komunikaciju potrebnu u procesnoj automatici. PROFIBUS PA se koristi za povezivanje više merne opreme sa procesnim kontrolerom preko jedne zajedničke linije i odlična je zamena za stari analogni sistem u kome je svaki merni signal prenošen zasebno, strujnim signalom u opsegu 4-20 mA. Pošto koristi strujne signale niske amplitude idealan je i za petro-hemijsku indust. Mana mu je ipak relativno spora komunikacija (31.25 kbit/s). Super mu je što istom linijom sa signalom dovodi i napajanje udaljenim slave jedinicama.

PROFINet je otvoren industrijski Ethernet standard nastao integracijom PROFIBUS DP sa Ethernet tehnologijom. PROFINet koristi TCP/IP i IT standarde, ali je u suštini Ethernet koji radi u realnom vremenu i potpuno se integriše sa PROFIBUS-om. On potpuno potiskuje iz upotrebe stari protokol PROFIBUS FMS.

Struktura PROFIBUS komunikacione mreže

PROFIBUS podržava tri načina prenosa podataka, tako da **postoje tri različita fizička nivoa ili različite strukture PROFIBUS mreža**

1. Naponski prenos (PROFIBUS – H2)

- ☐ Podrжан kod PROFIBUS DP, najvažniji i najčešće korišćen !
- ☐ Baziran na RS485 protokolu
- ☐ Half-duplex, multidrop zajednička magistrala (linija)
- ☐ Moguće su brzine prenosa od 9.6 kbit/s do 2 Mbit/s.
- ☐ Dozvoljena dužina kabla od 100 do 1200 m.

2. Optička veza (PROFIBUS - FO)

- ☐ Podrжан kod PROFIBUS DP kao Plastična ili staklena optička vlakna, dužina 80 m i 15 km, respektivno.
- ☐ Ovaj tip prenosa nije osetljiv na elektromagnetne smetnje
- ☐ uvek obezbeđuje galvansko odvajanje između stanica na liniji.

3. Strujni prenos (PROFIBUS-H1)

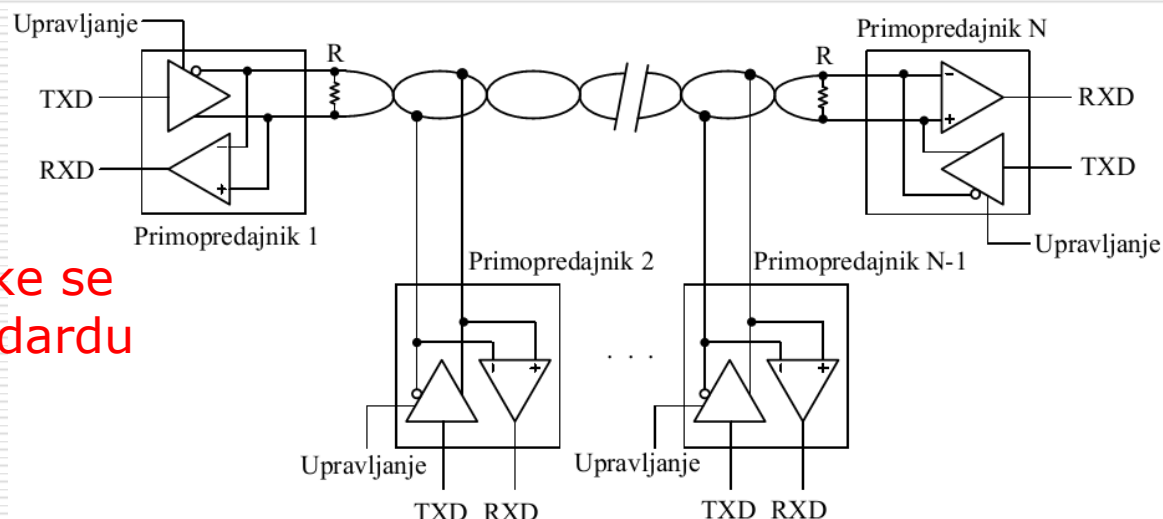
- ☐ Podrжан kod PROFIBUS PA i definisan standardnom , IEC 1158-2
- ☐ Koristi tehnologiju za prenos napajanja i podataka preko iste linije za vezu. Brzina prenosa na PROFIBUS PA liniji je relativno niska. 31.25 kbit/s.

Struktura PROFIBUS H2 mreže (naponski prenos)

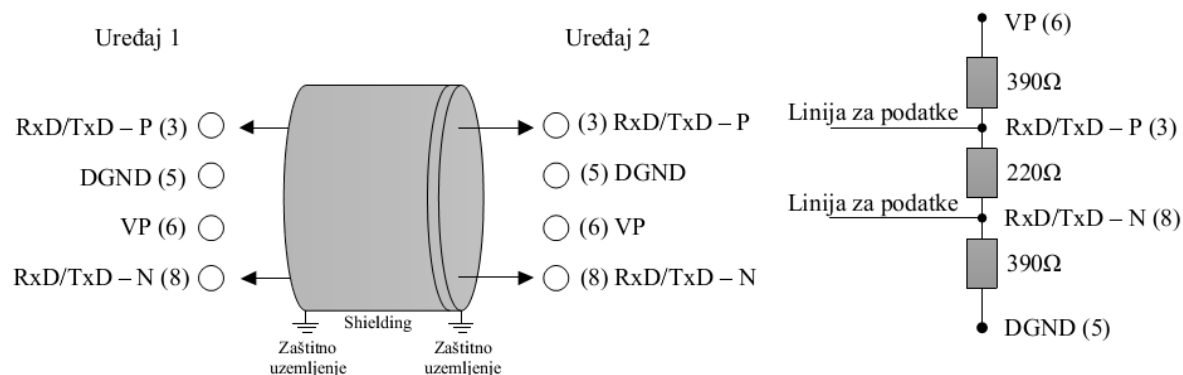
Pri naponskom prenosu PROFIBUS koristi standardnu RS485 multidrop liniju
Svi uređaji se kače na liniju, svi su primopredajnici, diferencijalni prenos....

Praktično ista linija kao
kod Modbus RS485

Jedan bajt PROFIBUS poruke se
isto prenosi po RS485 standardu
diferencijalno na dve linije



PROFIBUS predviđa liniju i sa 2 ali i sa 4 žice
□ Moguć prenos samo RxD/TxD+ i RxD/TxD-
□ Ali moguć i prenos napajanja VP i DGND



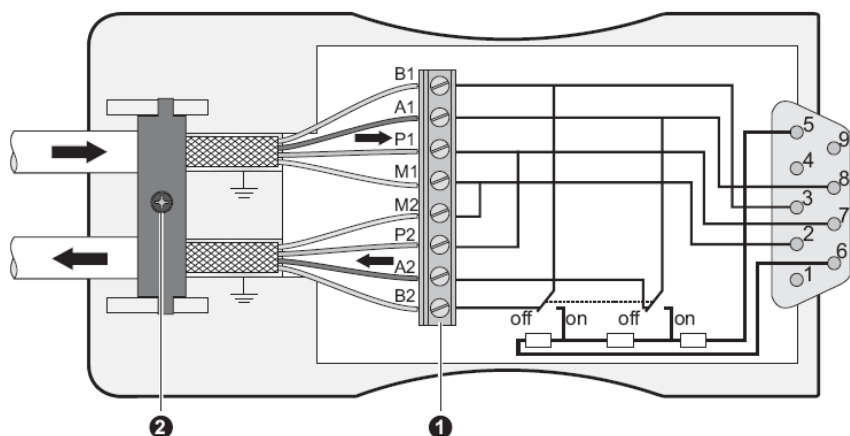
Uvrtnute parice



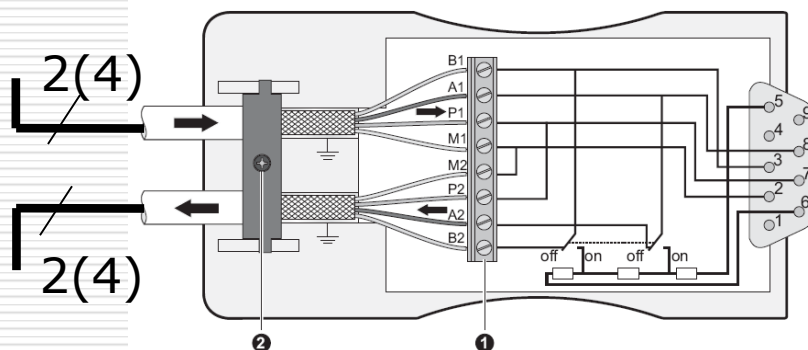
Struktura PROFIBUS H2 mreže (naponski prenos)

PROFIBUS DP konektor

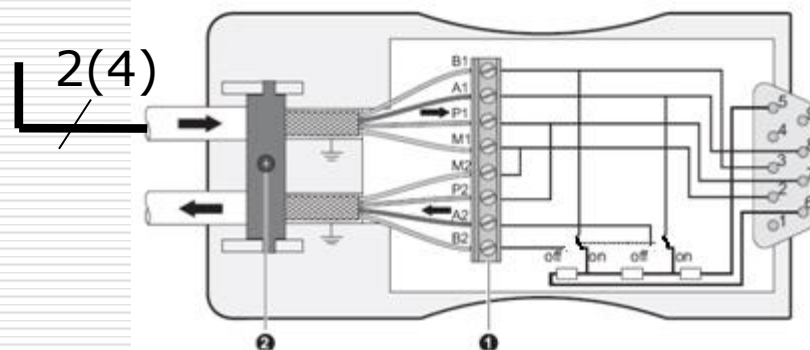
- ❑ Veza ka liniji (levo na slici) 8 konektora
 - ❑ Na konektore se vezuju 2 (4) ulazne žice i 2 (4) izlazne žice
 - ❑ Ako je terminalna tačka, vezuju se 2(4) ulaza i spoja terminacioni R
- ❑ Veza ka PLC /mikroprocesoru (desno) je 9-pinski konektor



Konektor klasičnog međuvvora na liniji



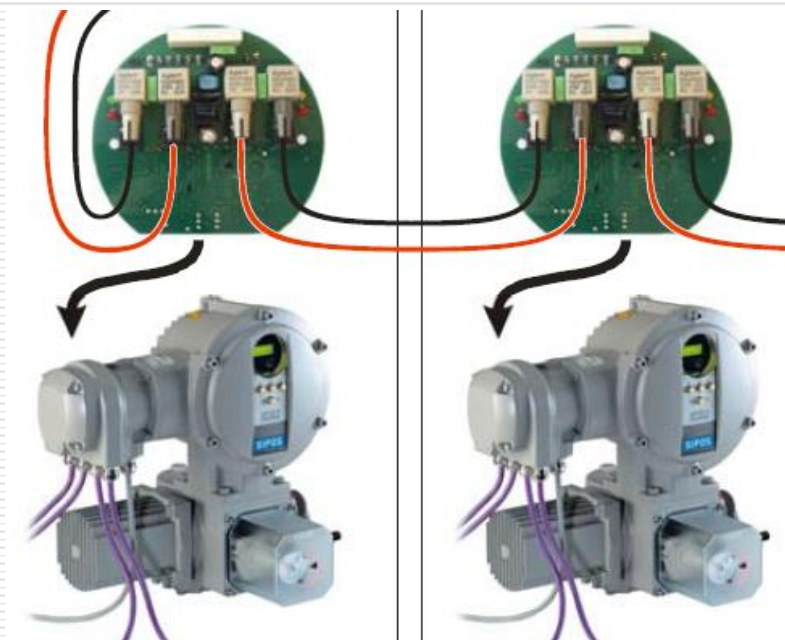
Čvor terminator linije



Struktura PROFIBUS FO mreže (optički prenos)

PROFIBUS podržava i optički prenos podataka. Ceo slajd informativno

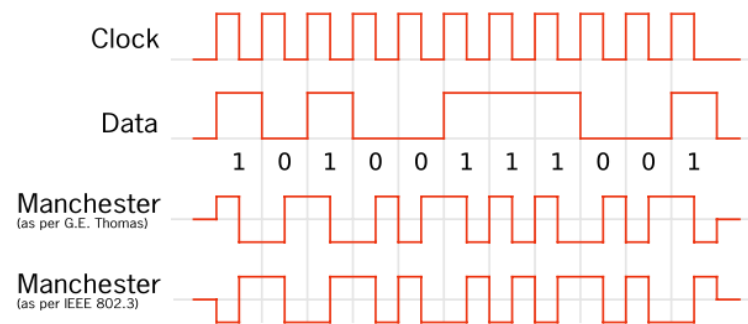
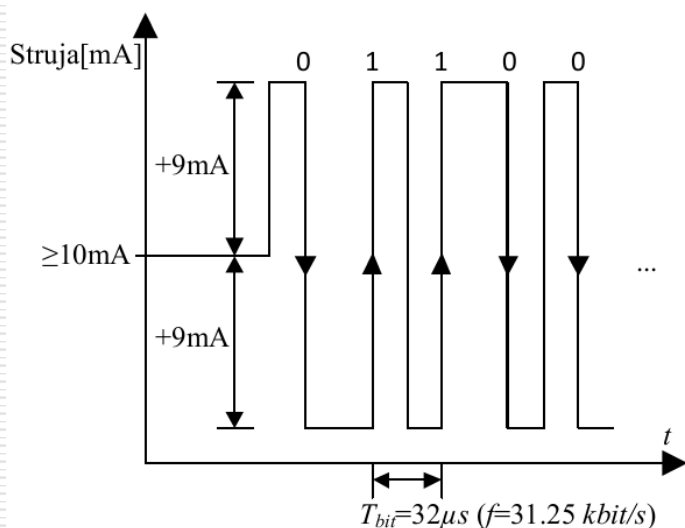
- ❑ Fizički link po PROFIBUS DP FO (FIBER OPTIC) standardu su plastična ili staklena optička vlakna.
- ❑ Maksimalna udaljenost za plastiku je 80 m a optička vlakna 15 km
- ❑ Prenos nije osetljiv na em. smetnje i obezbeđuje galvansko odvajanje.



Struktura PROFIBUS H2 mreže (strujni prenos)

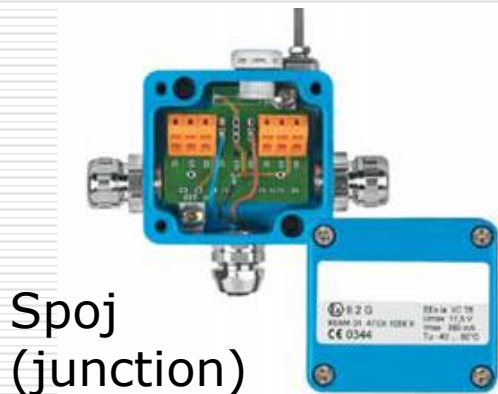
PROFIBUS podržava i strujni prenos podataka. Pošto struja ima DC nivo, preko linije se i napajaju udaljeni slave uređaji.

Da se izbegnu eventualni dugi intervali sa 1mA (prenos niza nula), nedovoljni za napajanje udaljenog uređaja, **koristi se MBP** (Manchester-encoded, Bus Powered) tehnologija za modulisanje podatka koji se prenose



Kod MBP uvek imamo tranziciju !

MBP uvek ima tranziciju na sredini bita, a u zavisnosti od informacije može imati i tranziciju na početku bita



terminator



Ceo slajd informativno

Nivo podataka PROFIBUS DP i PA – FDL nivo

Posle fizičkog nivoa, **PROFIBUS definiše drugi nivo komunikacije, nivo podataka (FDL- Fieldbus Data Link):**

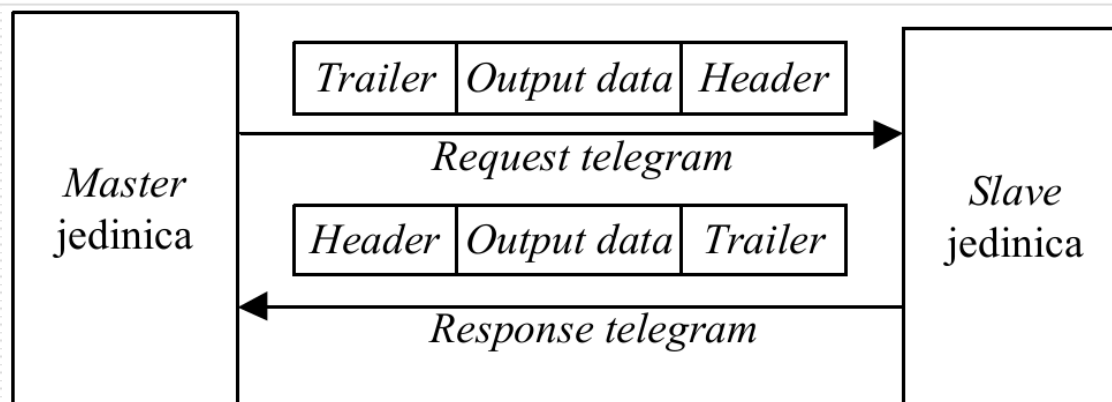
- ❑ kako se vrši kontrola pristupa liniji (master-slave procedura),
- ❑ kako se prenose podaci (definiše se skup PROFIBUS telegrama),
- ❑ kako se obezbeđuje sigurnost podataka (checksum polje).

Podaci se razmenjuju na liniji korišćenjen FDL telegrama

(fizička linija je slična sa Modbus RS485, ali telegrami su potpunu različiti)

- ❑ Aktivna master stanica (razni kontroleri) inicira komunikaciju slanjem telegrama odabranoj slave jedinici.
- ❑ Slave stanica (uglavnom senzori i aktuatori) odgovara na telegram od strane mastera koji je kontroliše.

Linija može imati više mastera, koji kontrolišu svoje slaves. Masteri ciklično razmenjuju token, a liniju koristi onaj master koja ima token.

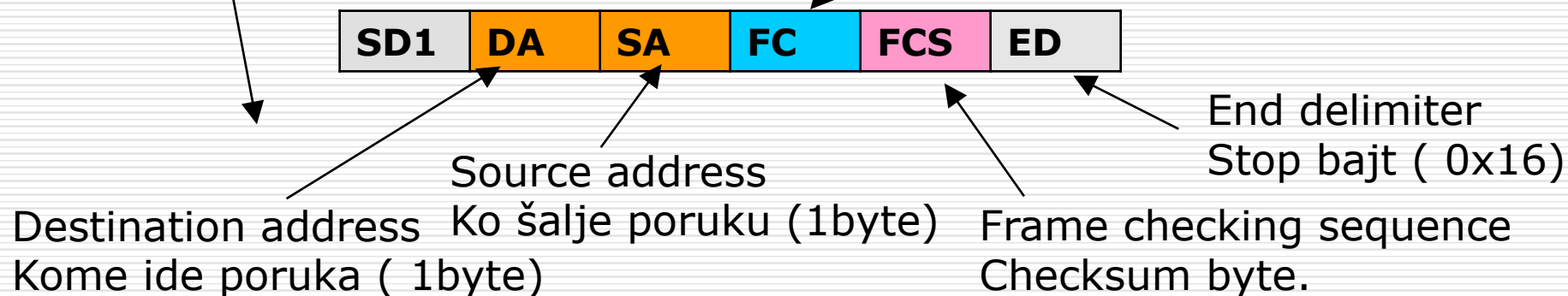


Struktura PROFIBUS poruka

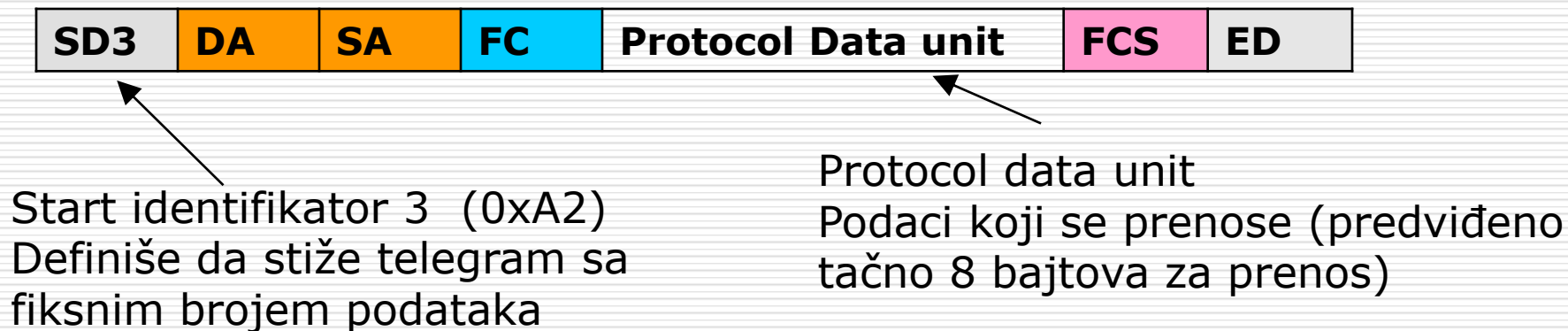
1. Telegram bez podataka

Start identifikator 1 (0x10)
Definiše start telegrama bez podataka

Function code
Kod funkcije/servisa koju treba izvršiti i njen prioritet na liniji (high ili low) (1byte)



3. Telegram sa fiksnim brojem podataka



Struktura PROFIBUS poruka

2. Telegram sa promenjivim brojem podataka

Start identifikator 2 (0x68) Stiže telegram sa varijabilnim brojem podataka

Podaci koji se prenose (broj im je promenjiv)



Koliko je telegram dugačak

Dužina = Protocol data unit + DA + SA + FC + DSAP + SSAP.

4. Prenošnje TOKEN-a

Poruka kojom jedan master preda kontoru linije drugom masteru

Start identifikator 4 (0xDC) Definiše TOKEN telegram



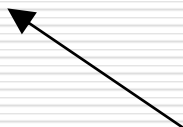
Destinaciona adresa novog mastera koji treba da preuzme liniju (1byte)

Izvorna adresa mastera koji je završio sa svojim ciklusom komunikacije na liniji i predaje TOKEN drugom masteru (1byte)

Struktura PROFIBUS poruka

4. Telegram potvrde -ACK

SC



Brief Acknowledgement (0xE5)

Potvrda prijema poruke i izvršenja komande. Samo jedan bajt, da se ne gubi vreme i mesto na liniji.

Posle svake poruke na liniji očekuje se potvrda sa destinacione adrese da je poruka stigla. SC je jednobajtna kratka potvrda, ona nema nikakav checksum ali jasno je:

- ☐ ako stigne bez greške onda je sve prošlo u redu.
- ☐ Ako ne stigne ili stigne sa greškom, svakako će se originalna poruka ponoviti.

Tip ACK koji FDL telegram zahteva se definiše u FC polju originalne poruke :

- ☐ SRD (Send and Request Data with Acknowledge)
- ☐ SDN (Send Data without Acknowledge)

Jako je važno ne gubiti vreme pri komunikaciji u realnom vremenu !
Ovo je kratka jednobajtna poruka kojom slave obavesti master da je uradio što je od njega traženo

Tipovi Profibus uređaja i dozvoljene adrese

Master jedinice (aktivne stanice)

- ❑ Class 1 masters - PLC, kontroleri, SCADA stanice, PC. To su centralni kontroleri koji ciklično razmenjuju podatke sa svojim distribuiranim slave jedinicama u tačno određenim vremenskim ciklusima.
- ❑ Class 2 masters – konfigur. uređaji, monitori linije, dijagnostičari. Uglavnom ne ostaju trajno spojeni za liniju, služe samo kao inženjerski alat pri konfiguraciji i dijagnostici PROFIBUS sistema.

Slave jedinice (pasivne stanice)

To su razni I/O blokovi, merni uređaji, aktuatori, ventili, električni pogoni. Ovi uređaji mere ili kontrolišu procesne veličine. Iako vode važnu ulogu u svom distribuiranom delu proizvodnog procesa (ili celom procesu), po pitanju komunikacije su pasivni, i odgovaraju samo na zahteve mastera.

U PROFIBUS mreži svaka jedinica mora imati jedinstvenu adresu

Adresa jedinice je uglavnom podešena lokalno. Alternativa je da se podesi preko konfiguracionog uređaja (class 2 master) tokom inicijalizacije.

- ❑ Postoji 128 adresa na jednoj PROFIBUS liniji, brojevi od 0 do 127.
 - adresa 127 je rezervisana za broadcast, poruka svima na liniji
 - Adrese od 0 do 3 koriste za uobičajeno za master uređaje
 - Adresa 126 je rezervisana za standardne uređaje (off the shelf)

Obavezne i opcione Profibus funkcije

Kada master pošalje FC (function code) slave mora da ga izvrši

Postoje obavezne (Mandatory) funkcije za sve PROFIBUS slave jedinice

Postoje i opcione (Optional) funkcije/servisi koje PROFIBUS slave jedinice mogu ali ne moraju da podržavaju

Spisak PROFIBUS funkcija je dat u sledećoj tabeli

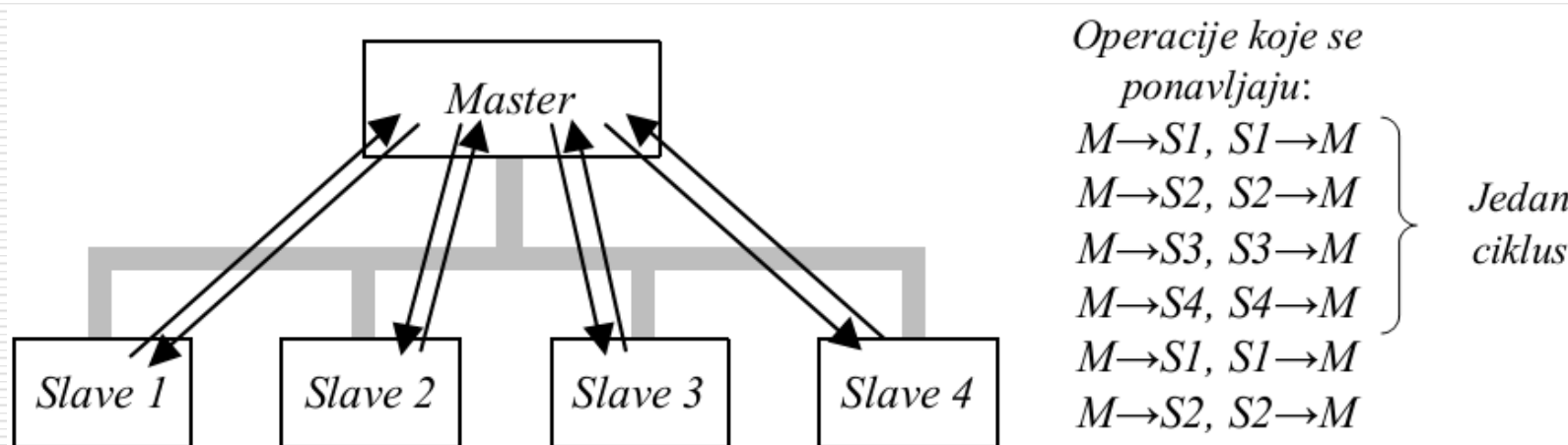
<i>Funkcija/servis</i>	<i>DP slave response</i>	<i>DP master I request</i>	<i>DP master II request</i>	<i>Nivo II servis</i>	<i>Opis funkcije</i>
<i>data_exchange</i>	<i>M</i>	<i>M</i>	<i>O</i>	<i>SRD</i>	Razmena podataka, izlazi ka <i>slave</i> , ulazi ka <i>master</i>
<i>RD_Inp</i>	<i>M</i>		<i>O</i>	<i>SRD</i>	Čitanje ulaza <i>slave</i> jedinice
<i>RD_Outp</i>	<i>M</i>		<i>O</i>	<i>SRD</i>	Čitanje izlaza <i>slave</i> jedinice
<i>Slave_Diag</i>	<i>M</i>	<i>M</i>	<i>O</i>	<i>SRD</i>	Dijagnostika <i>slave</i> jedinice
<i>Set_Prm</i>	<i>M</i>	<i>M</i>	<i>O</i>	<i>SRD</i>	Podešavanje parametara <i>slave</i> -a
<i>Chk_Cfg</i>	<i>M</i>	<i>M</i>	<i>O</i>	<i>SRD</i>	Provera konfiguracije <i>slave</i> -a
<i>Get_Cfg</i>	<i>M</i>		<i>O</i>	<i>SRD</i>	Čitanje konfiguracije <i>slave</i> -a
<i>Ser_Slave_add</i>	<i>O</i>		<i>O</i>	<i>SRD</i>	Promena <i>slave</i> adrese
<i>Global control</i>	<i>M</i>	<i>M</i>	<i>O</i>	<i>SRN</i>	Broadcast
<i>M-M-Comm</i>		<i>O (request & response)</i>	<i>O (request & response)</i>	<i>SRN/SDN</i>	Razmena podataka sa drugim <i>masterom</i>

Funkcija **data exchange** je vrlo popularna za rad u realnom vremenu.

Ona omogućuje da master šalje izlazne podatke a u odgovoru primi ulazne podatke. Na primer, posalje ref. Brzinu, a primi stvarnu brzinu motora.

Razmena podataka mastera i N slave uređaja

- ❑ Ciklična razmena podataka u kojoj master periodično, u krug , komunicira sa njemu dodeljenim slave jedinicama.



Cikličnu komunikaciju podržavaju sve Profibus master i slave jedinice.

Primer procene mogućnosti rada u realnom vremenu

- ❑ Potrebno je da master obrati svakoj od 4 slave jedinice svakih 5ms
- ❑ Koristi se data exchange funkcija sa telegramom sa fiksnih 8 data bajtova
- ❑ Profibus linija radi sa prenosom od 1Mbps , milion bitova u sekundi
- ❑ Smatrati da je kašnjenje u odgovoru slave jedinica zanemarljivo

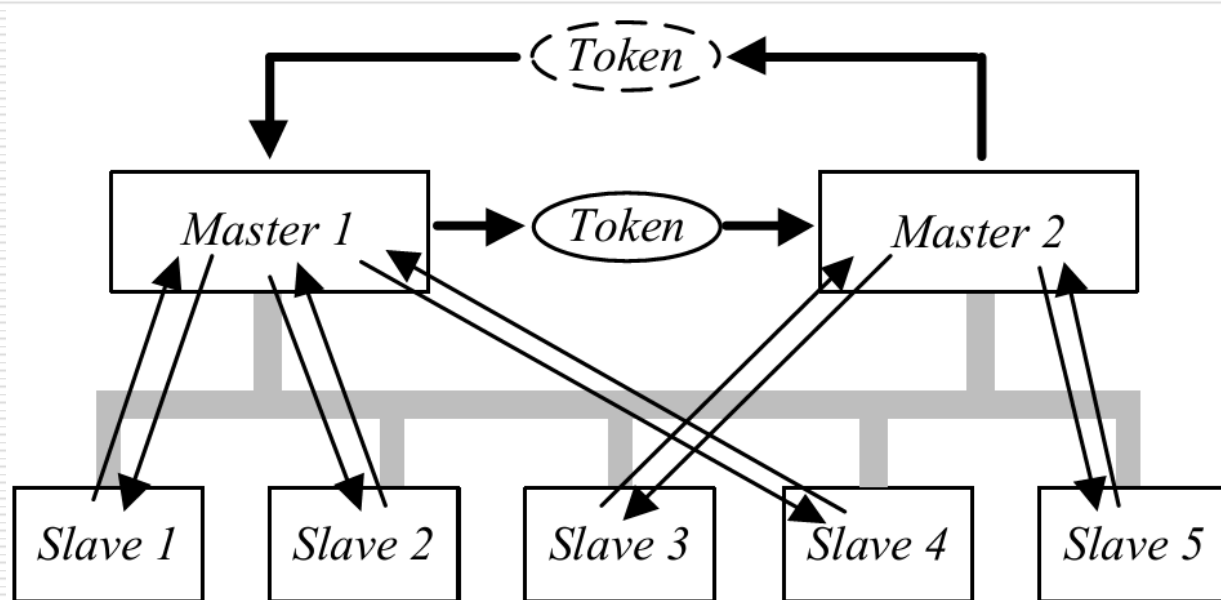
Min vreme jednog ciklusa = $4 * 14 * 2 * 10 * 1\mu s = 1.12ms < \text{potrebnih } 5ms$

4 slave — Br. bajtova u data exchange telegr. — Pitanje i odgovor — Br. bitova u RS485 bajtu START+8+STOP — Perioda jednog bita

Razmena podataka dva mastera i N slave uređaja

Ako postoje dva master uređaja na liniji:

- ❑ Master 1 ciklično razmeni podatke sa svojim slave jedinicama
- ❑ Zatim predaje token Masteru 2 i time mu prepušta liniju
- ❑ Master 2 razmeni podatke sa željenim slave jedinicama i vrati token



Operacije koje se ponavljaju:

$M1 \rightarrow S1, S1 \rightarrow M1$
 $M1 \rightarrow S2, S2 \rightarrow M1$
 $M1 \rightarrow S4, S4 \rightarrow M1$
 $M1 \rightarrow \text{token} \rightarrow M2$
 $M2 \rightarrow S3, S3 \rightarrow M2$
 $M2 \rightarrow S5, S5 \rightarrow M2$
 $M2 \rightarrow \text{token} \rightarrow M1$
 $M1 \rightarrow S1, S1 \rightarrow M1$
 $M1 \rightarrow S2, S2 \rightarrow M1$

Više mastera koristi istu Profibus liniju za komunikaciju sa svojim slave jedinicama. Primena: Jedan master kontroliše rad sistema automatskog upravljanja, drugi master se povremeno ubacuje u slobodne intervale, proverava rad, konfigurise ...

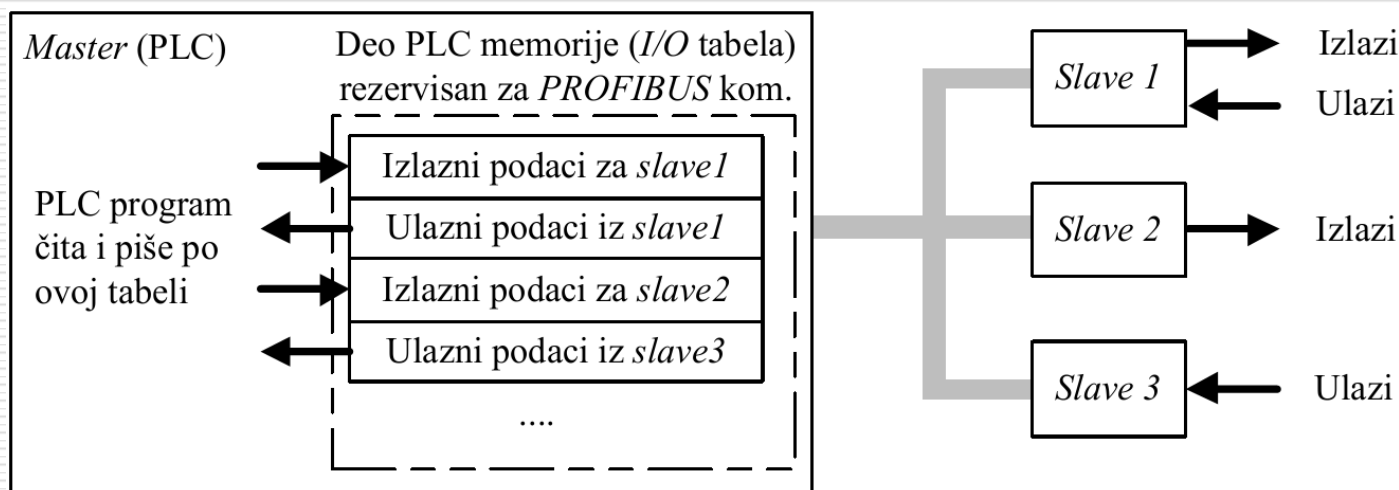
Kako PLC program koristi Profibus podatke?

PROFIBUS ciklična komunikacija je potpuno transparentna i automatska, pod kontrolom PROFIBUS driver programa unutar master jedinica.

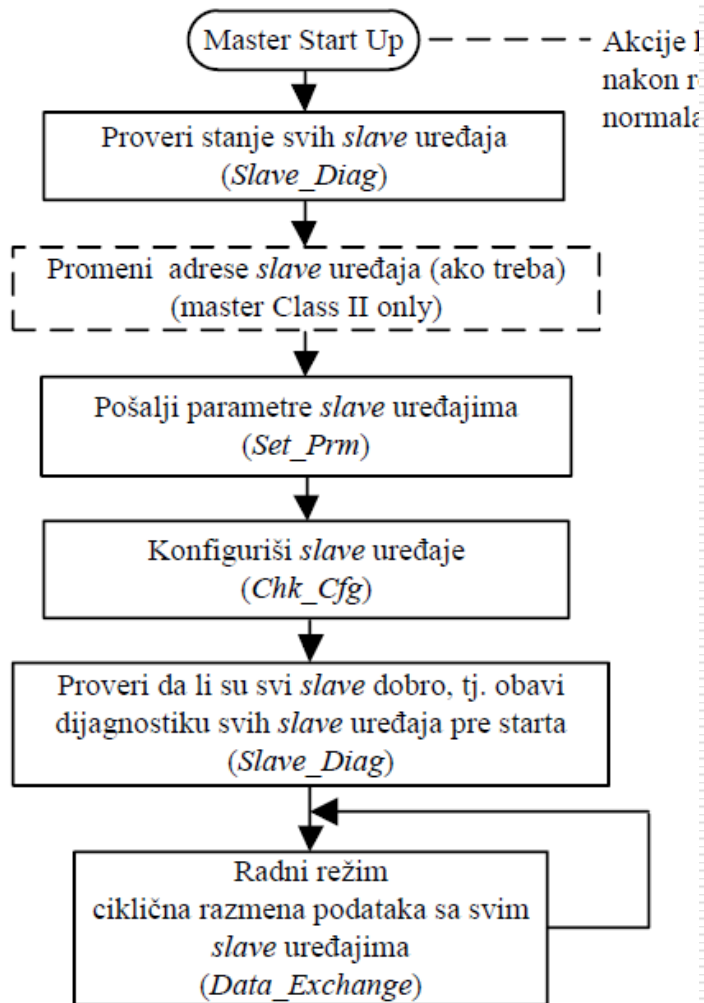
Razmena podataka sa PLC programom se vrši preko rezervisanih memorijskih lokacija !

- ❑ **Kada PLC program izračuna nove vrednosti koje treba poslati** slave jedinicama on te vrednosti prosto upiše u odgovarajuće memorijske lokacije
- ❑ Najnovije upisane vrednosti u tim lokacijama će biti poslate ka slave jedinicama kada za to dođe vreme u PROFIBUS periodičnom skan ciklusu,
- ❑ **kada PLC programu zatrebaju neke informacije** sa ulaza slave jedinica, on prosto pročita deo memorije rezervisan za ulazne podatke tih slave jedinice.
- ❑ Podaci u tim memorijskim lokacijama su već usveženi na najnovije vrednosti u poslednjem PROFIBUS skan ciklusu

PROFIBUS skan ciklus se izvršava u pozadi, nezavisno od korisničkog programa. Ovim se PLC program i rada PROFIBUS linije međusobno izoluju i ne remete.



Profibus startup i konfiguracija slave jedinica



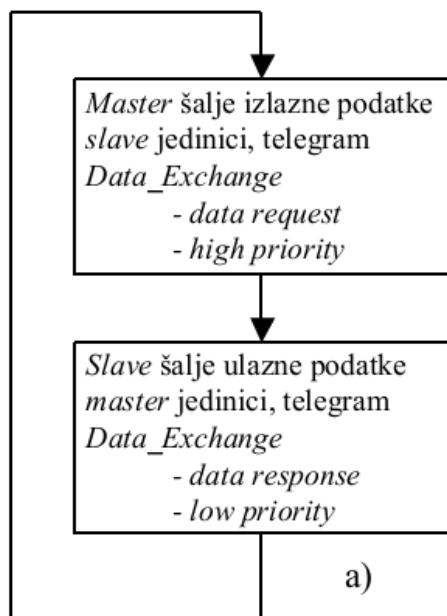
- Po dolasku napajanja PROFIBUS master izvršava svoju inicijalizaciju i zatim prelazi na
- Dijagnostiku svih njemu dodeljenih slave jedinica
- Menja adresu pojedinih jedinica ako je to neophodno
- Vrš parametrizaciju svake od njemu dodeljenih slave jedinica.
- Konfiguriše i proverava sve njemu dodeljene slave jedinica.
- Dijagnostiku svih njemu dodeljenih slave jedinica. Ovo se radi da bi se master uverio da su sve jedinice dobro podešene i spremne za rad tj za razmenu podataka sa masterom. Ako je sve ok prelazi se u
- Fazu ciklične razmene podataka, koja je radna faza.

Sličnu inicijalizaciju prolaze i sve slave jedinice. One ne ulaze u radni režim razmene podataka dok im se master ne javi, tj. dok nisu dobije konfiguracionu poruku od mastera. Ovim je obezbeđeno da se ceo distriburani SAU pokrene kontrolisano i istovremeno.

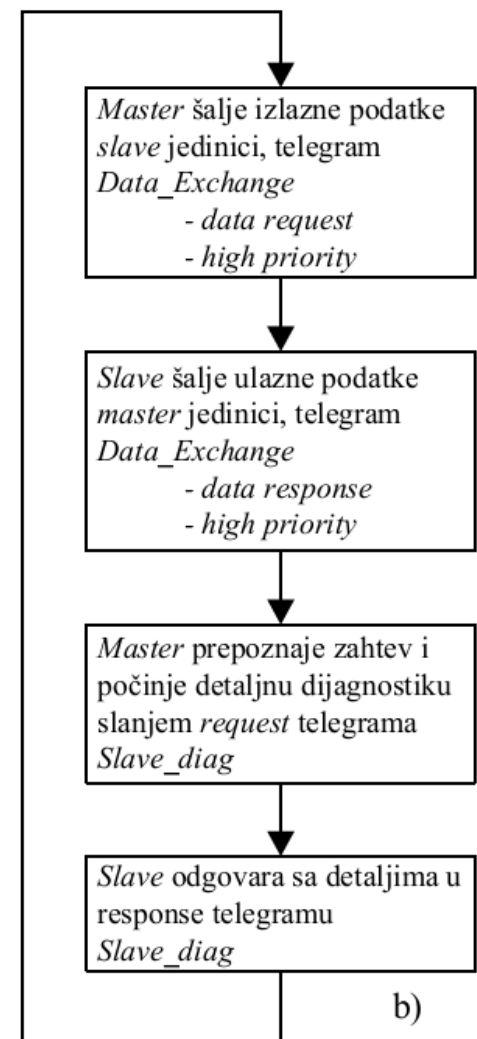
Profibus cikličan master-slave ciklus

Nakon uspešne inicijalizacije sistema, PROFIBUS startuje cikličnu razmenu podataka između master uređaja i aktivnih slave jedinica.

- ❑ U definisanom vremenu skeniranja master će se obratiti svakoj od slave jedinica, poslati joj referentne vrednosti, a dobiti njen status i merene vrednosti
- ❑ Ako nekoj slave jedinici treba veća pažnja, u odgovoru na podiže prioritetsnu zastavicu i master zna da nešto sa tim slave uređajem nije u redu



Normalan ciklus



nenormalan ciklus (slave zahteva pažnju)

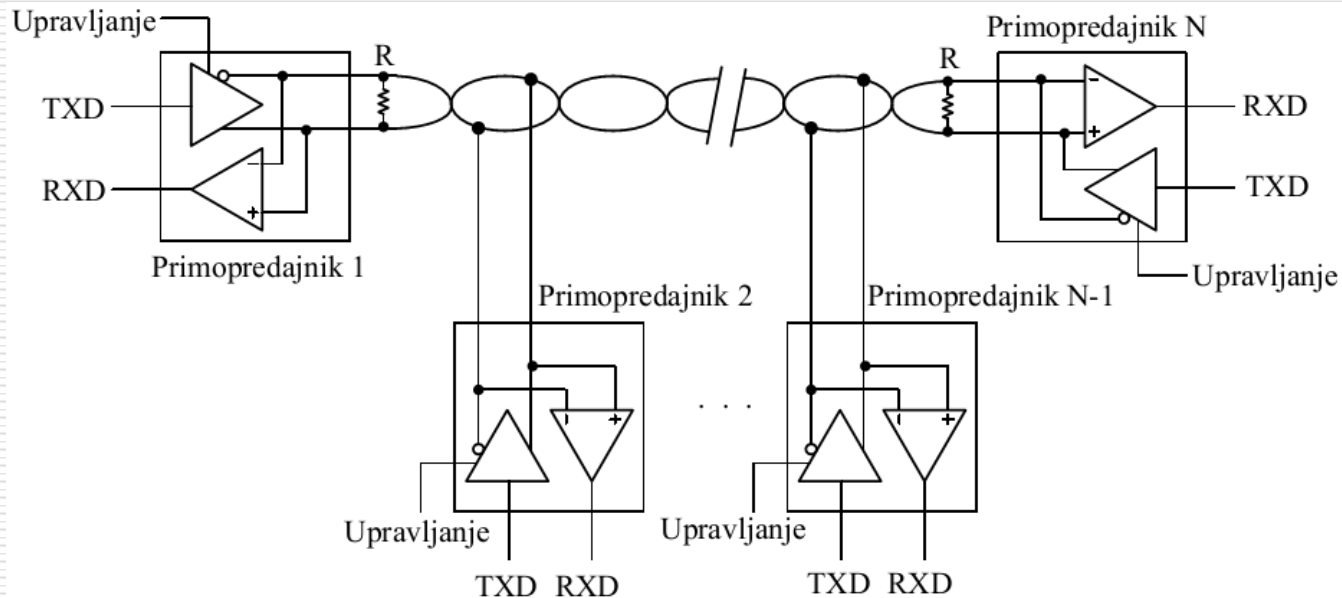
CANBUS protokol

□ Primena u autoindustriji

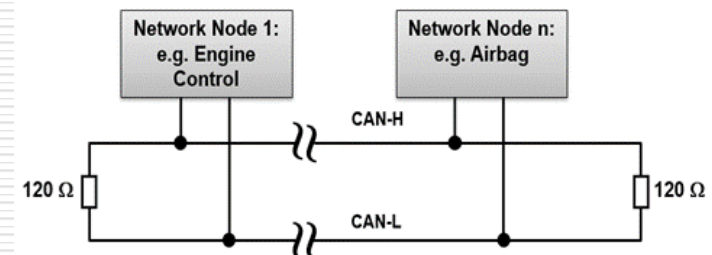
- Isto kao i Modbus i Profibus koristi half-duplex prenos i linijsku multidrop topologiju sa svim akterima primopredajnicima**
- U principu master/slave tipa ali slave jedinice po potrebi mogu da iniciraju prenos**
- CAN arbitraža na liniji, ako dva uređaja krenu istovremeno da prenose, pobeđuje onaj sa većim prioritetom funkcije ili adrese**

Struktura CANBUS mreže (fizički sloj)

- ❑ CAN magistrala (CAN bus) **slična/ista kao Modbus RS485 i kao PROFIBUS H1**

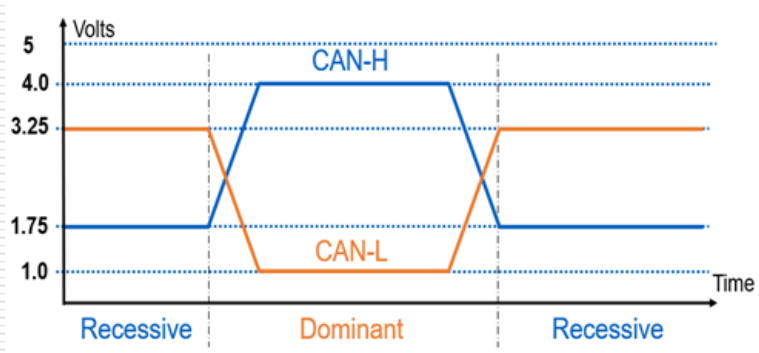


- ❑ CAN magistrala (bus) se sastoji od dve žice CAN High (CAN H) i CAN Low (CAN L), međusobno isprepletene i na čijim se krajevima nalaze dva otpornika od 120 Ohm
- ❑ Svaki primopredajnik ima dva režima rada
 - ❑ Prijem: na osnovu razlike napona H i L linije tumači da li neko šalje bit 1 ili 0
 - ❑ Predaja: Postavlja H liniju na visok napon a L na nizak za bit 1, za nulu obrnuto



Struktura CANBUS mreže (fizički sloj)

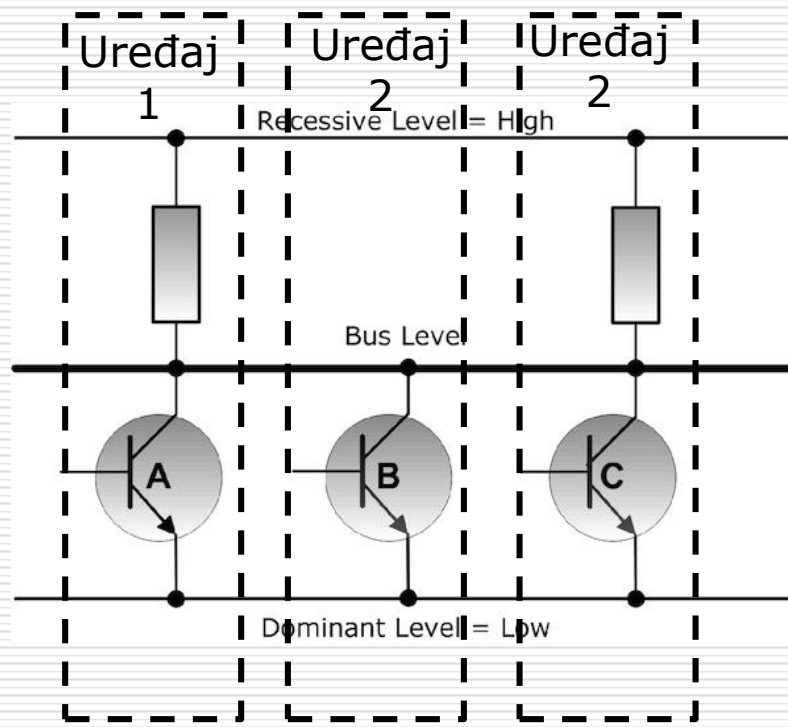
- ❑ Prenos jednog bita poruke na CAN magistrali nastaje tako što se menja razlika napona između CAN H i CAN L i na osnovu te razlike postoje 2 stanja: dominantna nula (0) i recesivna jedinica (1).



Naponski CAN nivoi se mogu menjati (manji nivo, brža promena ali veća šansa za grešku zbog šuma)

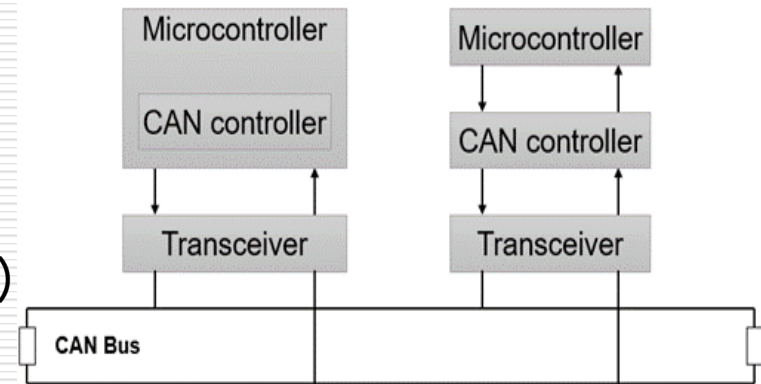
Kod CAN bus je nula dominantna jer je pravi tranzistor kao masi. Jedinica nastaje isključenjem tog tranzistora i pull-up otpornikom ka + napajanju. Kada uređaj npr 2 uključi tranzistor ka masi ta nula pobeđuje jedinice koju šalje npr uređaj 1 i 3 tako što isključe svoje tranzistor.

Ovaj mehanizam služi za detekciju kolizije (uređaj koji šalje 1 shvati da mu je na liniji neko postavio 0)



Struktura CANBUS mreže (fizički sloj)

- Svaki čvor koji učestvuje u CAN komunikaciji ima 3 osnovna dela:
 1. PLC ili mikrokontroler
 2. CAN kontroler
 3. Primopredajnik (CAN Transceiver)

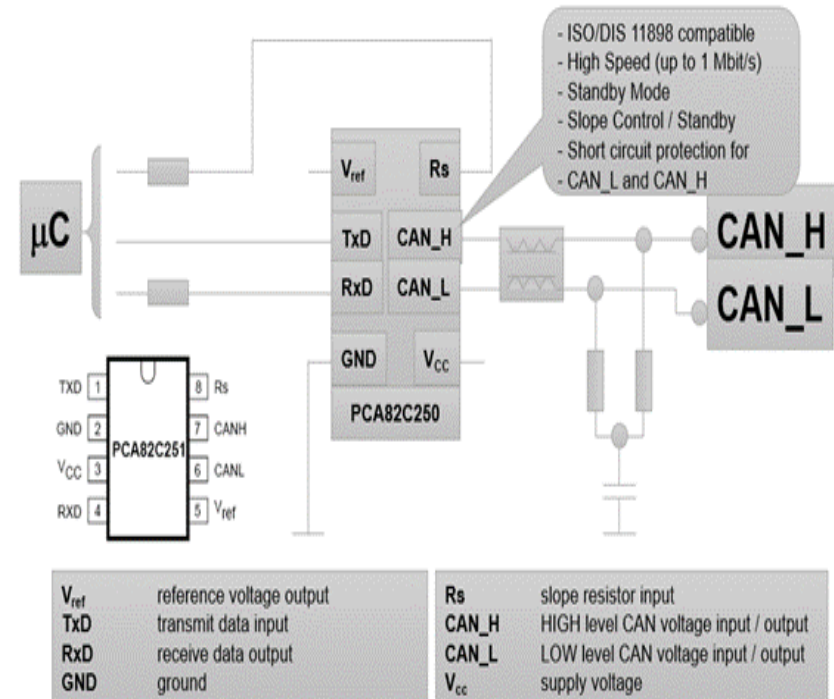
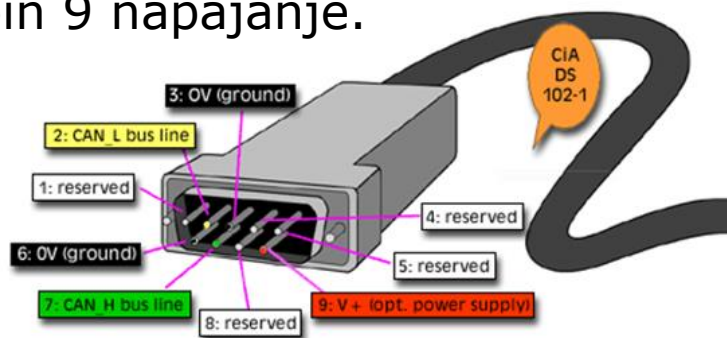


Primer CAN primopredajnika

High Speed Double-Wire transiver
PCA 82C250

Primer CAN Konektora –DB9

- 2 i 7 su CAN L i CAN H,
- 3 i 6 su uzemljenja,
- pin 9 napajanje.

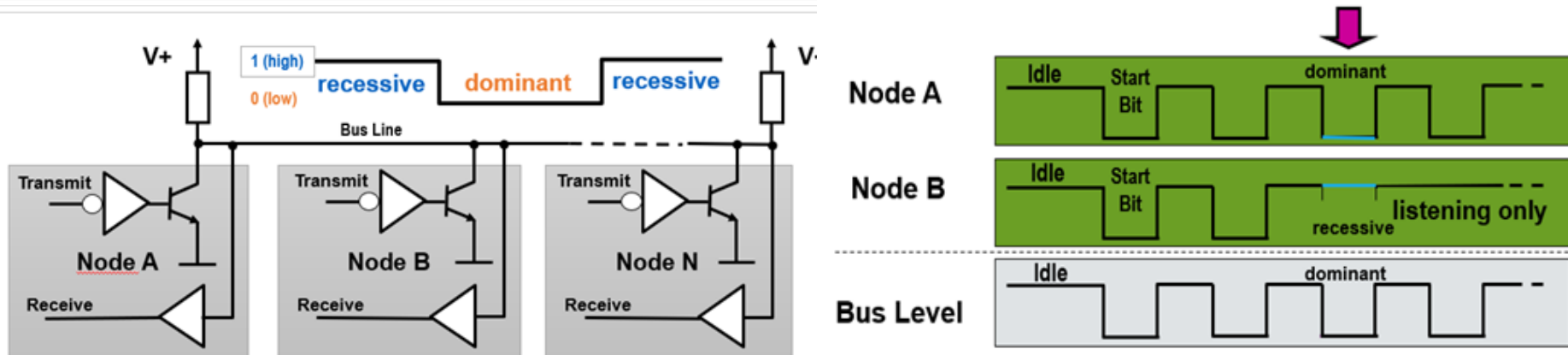


CAN pristup liniji -arbitraža podataka

- ❑ Prilikom CAN komunikacije svi čvorovi imaju pravo da šalju podatke ili pročitaju podatke koje je određeni čvor poslao na magistralu pa na osnovu sadržaja poruke da odluče da li će ih iskoristiti ili neće
- ❑ Kod CAN komunikacije (kao i kod Modbus i Profibus) fizička RS485 linija je half-duplex tako da svi mogu da šalju podatke, ali ne u isto vreme
- ❑ **Za razliku od Modbus i Profibus protokola koji pristup liniji rešavaju master-slave principom, CAN bus protokol dozvoljava veću slobodu.**
 - ❑ Pristup liniji kod CAN bus komunikacije je moguće realizovati po master – slave principu. U ovom slučaju se definiše ko je master, a ostalim akterima u CAN komunikaciji dodeljuje uloga slave jedinica. Pošto slave jedinice nikad ne počinju komunikaciju nikad neće doći do kolizije na liniji
 - ❑ CAN bus sistem komunikacije je moguće osmisliti tako da nema mastera i slejvova, svi čvorovi mogu da započnu komunikaciju kad hoće. Ali, ako slučajno počnu da pričaju u isto vreme, koji će od tih čvorova nastaviti da priča određuje se **arbitražom**. Arbitraža može da favorizuje čvor sa manjom adresom ili manjim brojem funkcije koju šalje (u svakom slučaju zbog jače nule na CAN bus liniji manji broj pobeđuje)
 - ❑ Uglavnom se koristi kombinacija, obično se zna ko je master tako da CAN bus većinu vremena radi na master –slave principu. Master je taj koji uglavnom započinje komunikaciju i dobija odgovor od nekog slejva, tako da uglavnom nema kolizije. Ali, dozvoljeno je nekom slejvu ako je u problemu da pošalje poruku preko reda, i ako dođe do kolizije tada ide arbitraža.

CAN pristup liniji - arbitraža podataka

- Na slici je prikazana linija sa tri čvora, A, B i N.
- Pretpostavimo da čvor A i čvor B baš u isto vreme izađu iz neaktivnog stanja. U tom slučaju oba čvora šalju prvo dominantni START (0) bit kako bi javili svima na liniji da započinju komunikaciju.
- Zasad niko ništa ne može da primeti, linija se promeni na START bit. Takođe u primeru sa slike su naredna tri bita u njihovim porukama ista (1 0 1), što se vidi i na magistrali, tako da oba čvora i dalje šalju.
- Ali, nakon toga čvor A šalje dominantni bit (0), dok sa druge strane čvor B šalje recesivni (1). Na magistrali 0 pobeđuje, tako da čvor A pobeđuje u arbitraži, dok čvor B gubi, shvata da mu 1 nije prošla. On odmah prestaje sa slanjem i nastavlja da sluša.
- Dakle iz prethodnog primera možemo da uočimo da je logika arbitraže što „manji“ ID poruke, to je veći prioritet. ID može biti adresa ili funkcija

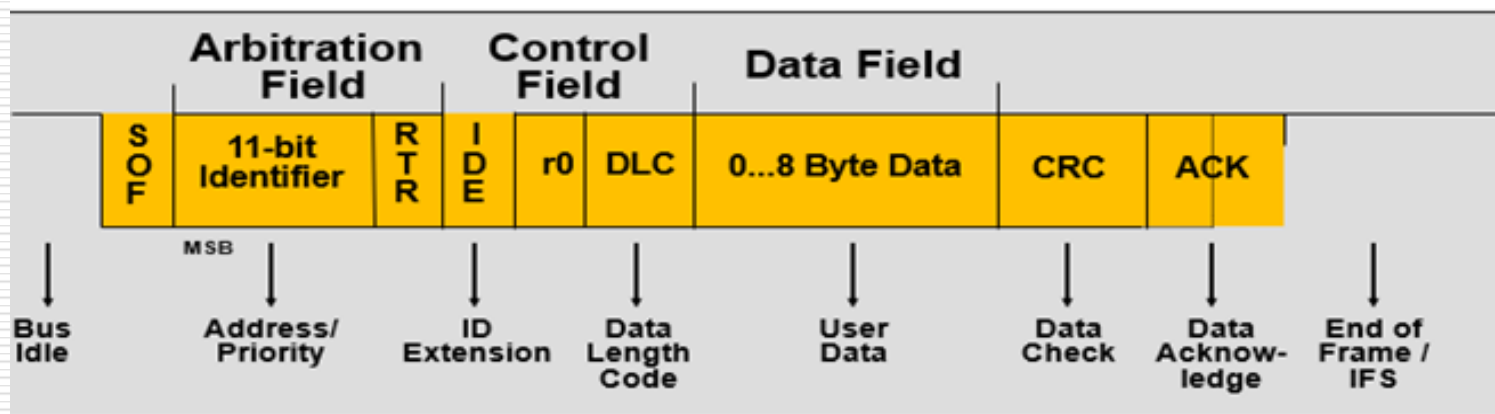


Struktura CAN poruke

- CAN protokol podržava različite formate poruka (jedna poruka jedan ram)
- Podržani su sledeći formati ramova:
 - ☐ **Standardni ram podataka**
 - ☐ **Prošireni ram podataka**
 - ☐ Ram sa zahtevom za podacima od drugog čvora
 - ☐ Ram poruke o grešci
 - ☐ Ram sa porukom o prekoračenju
 - ☐ Interval između dve poruke (međuramski interval)

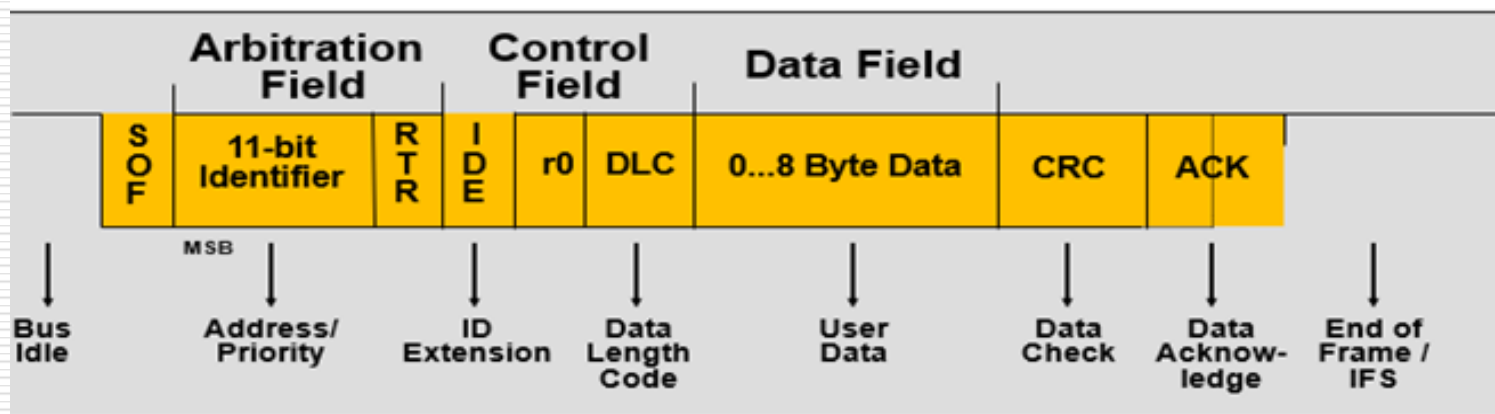
CAN poruka-standardni RAM

- ❑ Sve CAN poruke (pa i standardni ram) otpočinje bitom za početak koji se naziva SOF bit (*Start of frame*) koji je dominantan.
- ❑ Nakon SOF sledi identifikator sa 11-bitima, Manji broj identifikatora znači veći prioritet poruke ili onoga ko tu poruku šalje.
- ❑ 12-ti bit predstavlja zahtev za odgovorom od drugog čvora.
- ❑ Sledeće polje je kontrolno polje (6 bitova). Prvi bit govori da se radi o standardnom ramu. Drugi je rezervisan, poslednja 4 bita govore koje dužine je polje podataka (*Data Length Code - DLC*).
- ❑ Podaci koji se šalju se nalaze u sledećem polju čija je dužina određena DLC delom kontrolnog polja. Maksimalna dužina polja za podatke je 8 bajtova !



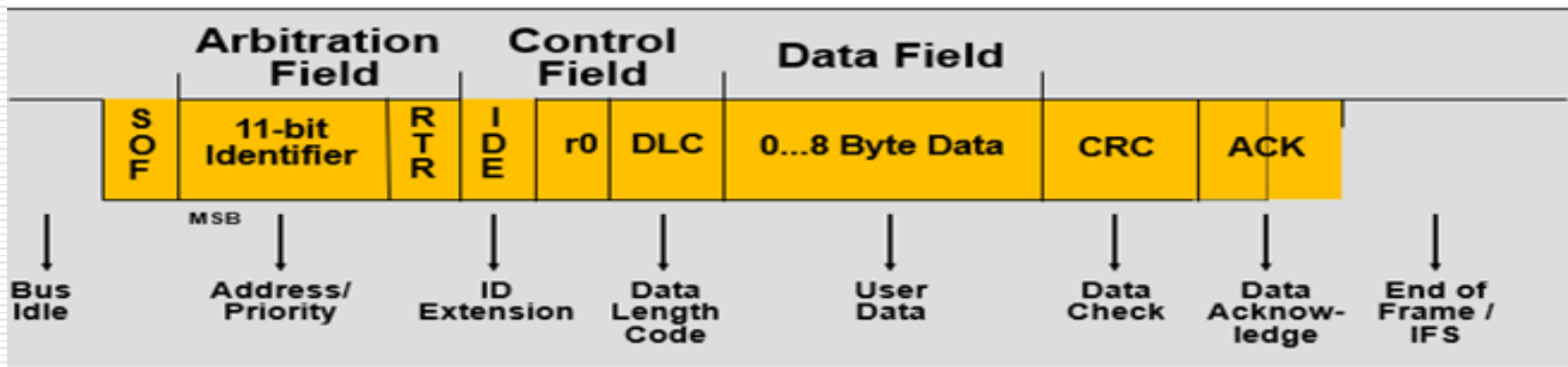
CAN poruka-standardni RAM

- ❑ Polje za proveru grešaka sledi nakon polja podataka i proveru se vrši pomoću CRC-a (Cyclic Redundancy Check).
- ❑ Sledi polje potvrde (jedan recesivan bit) koji predajnik šalje i u toku čijeg trajanja čvorovi koji su dobili poruku bez obzira da li je njima namenjena ili ne, i proverili njenu ispravnost odgovaraju sa potvrdom (dominantnim bitom). Ako je došlo do greške u prenosu nema te potvrde i predajnik shvata da mora da ponovi poruku, tj. da je ponovo pošalje.
- ❑ Nakon ovog bita sledi delimiter,
- ❑ Poslednje polje je polje kraja rama (eng.EOF – End Of Frame) koje čine 7 uzastopnih recesivnih bita.

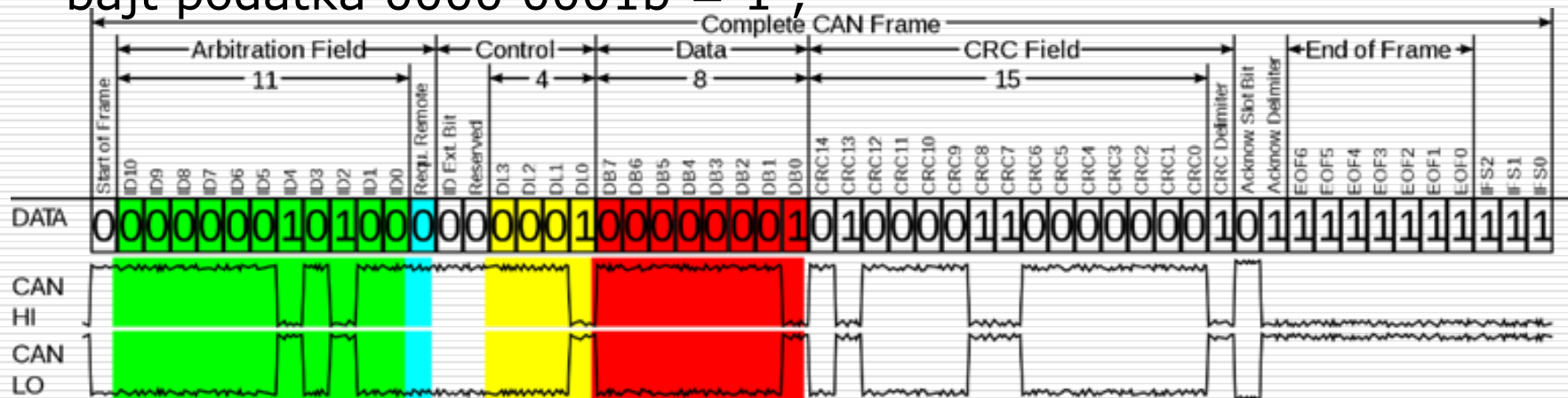


CAN poruka-standardni RAM

- ## ❑ Primer jednog standardnog CAN rama

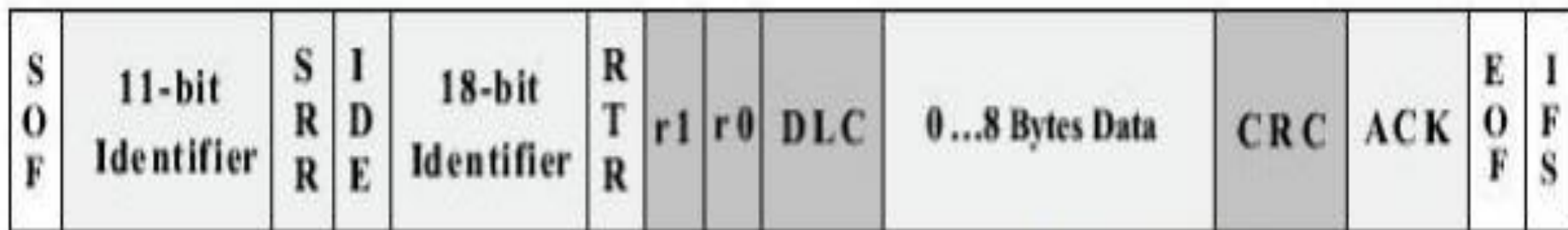


- ❑ Start je uvek bit 0 and End of frame uvek 7 jedinica
- ❑ U primeru je ID = 0 0001 0100b = 20, a šalje se samo jedan bajt podatka 0000 0001b = 1 ,



CAN poruka-prošireni RAM

- ❑ Prošireni ram podataka je sličan standardnom ramu podataka. Razlika je samo u identifikatoru. Kod proširenog rama podataka posle SOF bita sledi 38 bita indikatorskog polja. Prvih 11 bita predstavlja 11 značajnijih bita 29-bitnog indikatora (baznog indikatora), zatim sledi bit zahteva za slanjem podataka od drugog čvora (eng. Substitute Remote Request – SRR) koji je recesivan i bit opisa identifikatora rama (eng. Identifier Extension - IDE) koji je ovde recesivan kako bi se razlikovao prošireni od standardnog rama
- ❑ To znači da ukoliko još nije rešeno pitanje arbitracije na magistrali, a “takmiče” se standardni i prošireni ram, standardni ram će “pobediti”. Nakon toga slede 18 bita identifikatora i dominantni RTR bit.



Glavne prednosti/mane CAN sistema

Prednosti CAN sistema

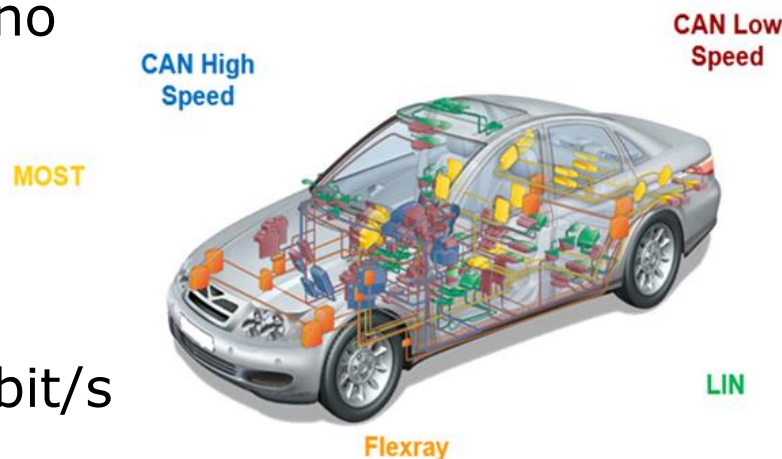
- ☐ Mala kompleksnost ožičenja
- ☐ Lako održavanje kablova
- ☐ Novi čvorovi se mogu lako dodati i ukloniti
- ☐ Otkazivanje jednog čvora ne utiče na druge čvorove u sistemu
- ☐ Centralizovana kontrola
- ☐ Svi uređaji na magistrali mogu da pročitaju istu poruku
- ☐ Zaštita od elektromagnetnog zračenja u slučaju kvara
- ☐ Jednostavna dijagnostika i lako održavanje

Mane CAN sistema

- ☐ Brzina prenosa podataka može biti mala u nekim primenama (ograničena na 1 Mb/s)
- ☐ Skupa implementacija, jer su potrebni specijalni kontroleri
- ☐ Kompletno isključenje sistema ukoliko je glavni CAN kabl oštećen

Primena CAN komunikacije -automotive

- ❑ CAN je jedan od mnogih protokola koji se koriste u autoindustriji (ali jedan od važnijih)
- ❑ U uobičajenim CAN komunikacionim protokolima na jednom linku može biti prisutno i do 30 čvorova, a u najnovijim automobilskim CAN protokolima moguće je istovremeno ostvariti komunikaciju između 70 čvorova.
- ❑ U zavisnosti od potrebne brzine i količine podataka koju je potrebno preneti u realnom vremenu:
 - ❑ brzi CAN protokol i
 - ❑ spori CAN protokol,
- ❑ Brzi CAN odlikuje prenos do 1 Mbit/s ali male razdaljine do 40m,
- ❑ Spori CAN protocol odlikuje brzina prenosa do 125 kbit/s na distance do 500m.



Primeri primena CAN komunikacije

- ❑ **Start-stop sistem** određuje kada će motor vozila biti ugašen kako bi se smanjila emisija štetnih gasova. Sistem preko CAN dobija sve podatke: sa senzora temperature motora, senzora brzine vozila, senzora klime, ...
- ❑ **Električna parkirna kočnica** Njena aktivacija ili deaktivacija je moguća samo u slučaju kada je pedala kočnice pritisnuta u suprotnom nije ispunjen uslov za promenu njenog stanja, podatak da je pedala kočnice aktivirana prosleđuje se putem CAN komunikacionog protokola do čvora koji vrši kontrolu nad električnom parkirnom kočnicom.
- ❑ **Hill hold** - aktivacija parkirne kočnice u slučajevima mirovanja na uzbrdici. Potrebni podaci za njenu automatsku aktivaciju potrebno je dobiti sa senzora koji mere brzinu vozila, nagib pod kojim se vozilo nalazi.
- ❑ **Sistem praćenja linija** komunicira putem CAN protokola sa čvorovima koji upravljaju motorom, upravljačem i kočionim sistemom vozila. Informacije sa senzora za detekciju linije se dostavljaju jedinicama koje regulišu brzinu ili rastojanje od bočne ili centralne linije puta.
- ❑ U najmodernijim vozilima **CAN se koristi za bržu aktivaciju vazdušnih jastuka** pri detekciji saobraćajne nezgode kada korisnik vozila koristi pojas za vezivanje nego kad nije upotrebljen. **Ova implementacija svedoči o važnosti brzine CAN komunikacije i prioritetima u CAN porukama.**
- ❑ Itd, itd, itd ...

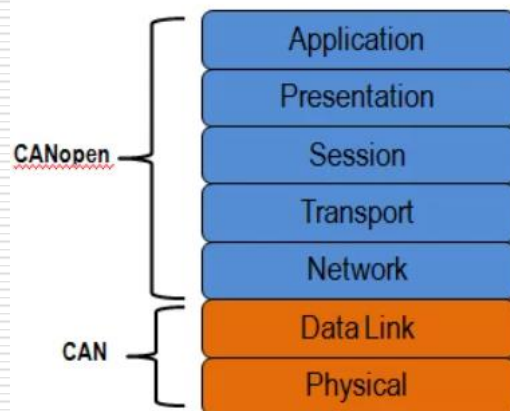
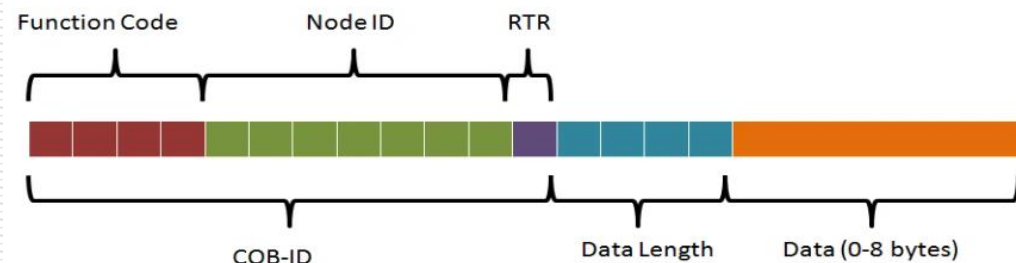
CANopen - nadogradnja CAN protokola

- ❑ CAN protokol samo definiše sve u vezi prenosa jednog CAN rama. On ne definiše šta treba da je u njemu. **I gde su definisane CAN funkcije koje uređaji treba da izvrše? Gde su adrese? Kada ko šalje? NEMA !!**
- ❑ CAN dakle treba nadograditi protokolom koji rešava dodatna pitanja kao što su: koje funkcije protokol podržava, koje su adrese dozvoljene, ko i kada šalje zahteve, ko odgovara na zahteve kada se zahtev pošalje ...

CANopen je dobar primer nadogradnje CAN protokola. On se oslanja na CAN da mu prenese jedan ram poruke ali sve ostale aspekte komunikacije on dalje definiše. Po OSI modulu komunikacije (više o tome kod interneta) CAN Open rešava sve sem DataLink (strukture ram poruke) i fizičkog dela protokola, to radi CAN.

1. CANopen definiše dozvoljene funkcije i adrese tako što deli 11-bitno CAN-ID polje:

- ❑ 4 bita za funkciju
- ❑ 7 bitova za adresu uređaja.

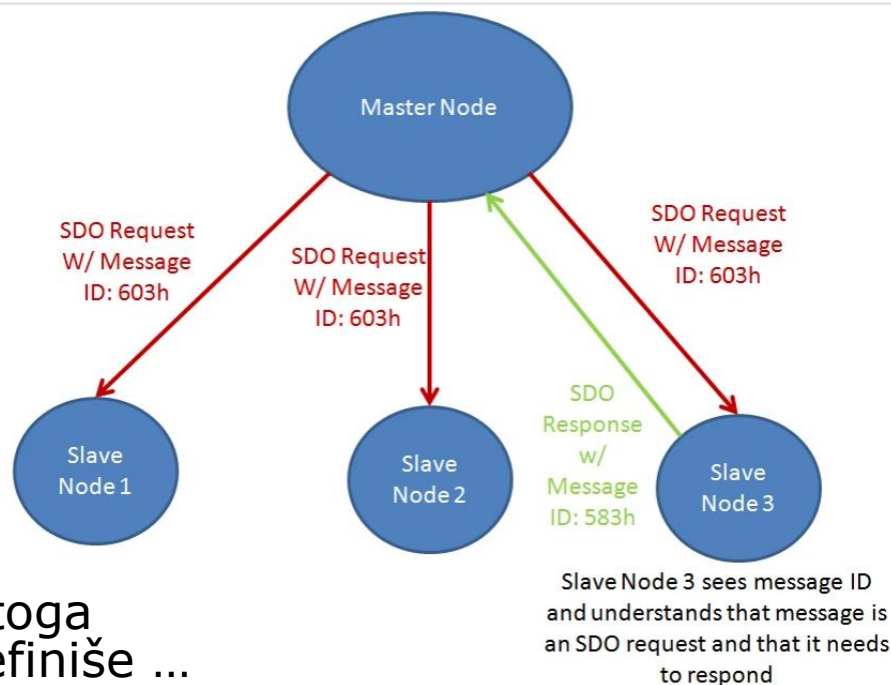


- ❑ Funkcija manjeg broja ima veći prioritet !
- ❑ CANopen time podržava maksimum 127 čvorova na liniji

CANopen - nadogradnja CAN protokola

2. CANopen takođe uvodi obaveznu funkciju Service Data Objects (SDO)

- ❑ Svaki čvor koji podržava CANopen mora da podržava SDO funkcionalnost, tj. da deluje kao server za zahteve za pisanje/čitanje.
- ❑ Time CANopen omogućuje da jedan glavni uređaj (praktično master) radi kao klijent raznim servisima na slave uređajima kao serverima
- ❑ Glavni uređaj (klijent) i ciklično čita/piše iz/u raznih servera,
- ❑ Uvek samo klijent počinje komunikaciju (slično je i na internetu!)
- ❑ Dakle, CANopen uređaji su ili SDO klijenti ili SDO serveri



To znači da je CANopen klijent/server tip protokola a ne master/slave.

Mada, sve je to vrlo slično po pitanju komunikacije:

Master – klijent

Slave – server

Još mnogo toga
CANopen definiše ...