Budapesti Műszaki és Gazdaságtudományi Egyetem
Közlekedésautomatikai Tanszék

Járműfedélzeti rendszerek II.

6. előadás

Dr. Aradi Szilárd

LIN (Local Interconnect Network) kommunikációs hálózat

- 1980-as években jelentek meg az UART alapú soros megoldások a gépjárművekben,
- Gyors elterjedés <- alacsony költség
- Különböző megoldások a fizikai szinten
- Különböző megoldások a protokollokban
- Felmerült a szabványosítás iránti igény ->
- LIN Consortiun (Audi, BMW, DaimlerChrysler, VW, Volvo – Volcano Tech., Motorola)

A LIN busz főbb tulajdonságai

- Egy Master (mester), több Slave (szolga) koncepció
- Alacsony költségű UART alapú integrált áramköri megoldás, egyszerű állapotgép alapú működés
- Kvarc, illetve rezonátor nélküli szinkronizációs lehetőség a szolgák esetében
- Determinisztikus jelátvitel előre számítható jelterjedési idővel
- Alacsony költségű egyvezetékes megoldás
- 20kbit/s sebesség
- Diagnosztikai támogatás

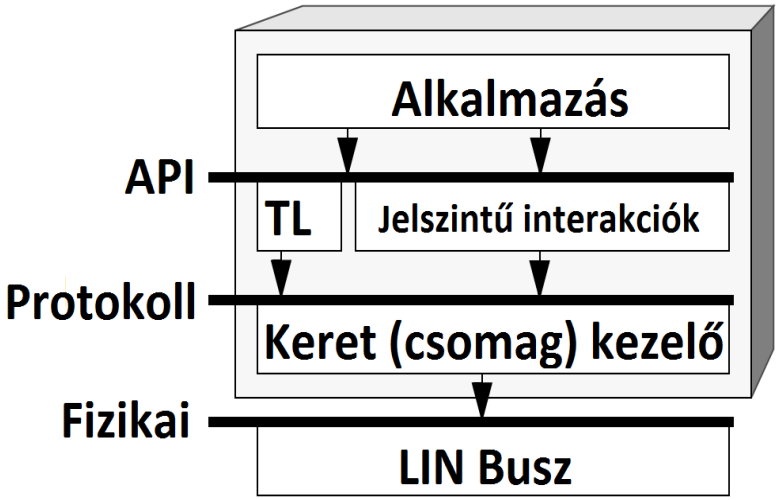
A protokoll főbb tulajdonságai

- Ez a protokoll a következő előnyös tulajdonságokat eredményezi:
  - **Flexibilitás**: A LIN hálózathoz tetszőlegesen lehet (szolga-) csomópontokat hozzáadni a többi szolgacsomópont szoftveres vagy hardveres átkonfigurálása nélkül.
  - **Üzenetcímzés**: A CAN-hez hasonlóan az üzenet címzettjét a keret azonosítója definiálja.
  - **Multicast**: Egyszerre több csomópont is felhasználhatja a keretek adatait.

LIN rétegek

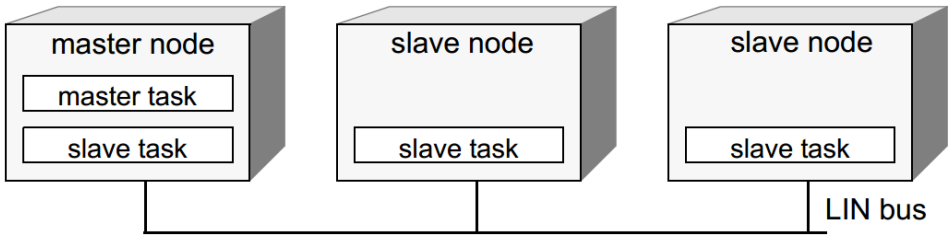
- A LIN szabvány a kommunikációt négy rétegre osztja, amelyet három interfész köt össze. A fizikai réteg és az alkalmazás között a keretkezelő és a jelfeldolgozó szolgáltat kapcsolatot.

Sokszor a jelek és keretek kezelése egyben, szoftveresen megvalósítva alkot közös interfészt az alkalmazás felé.



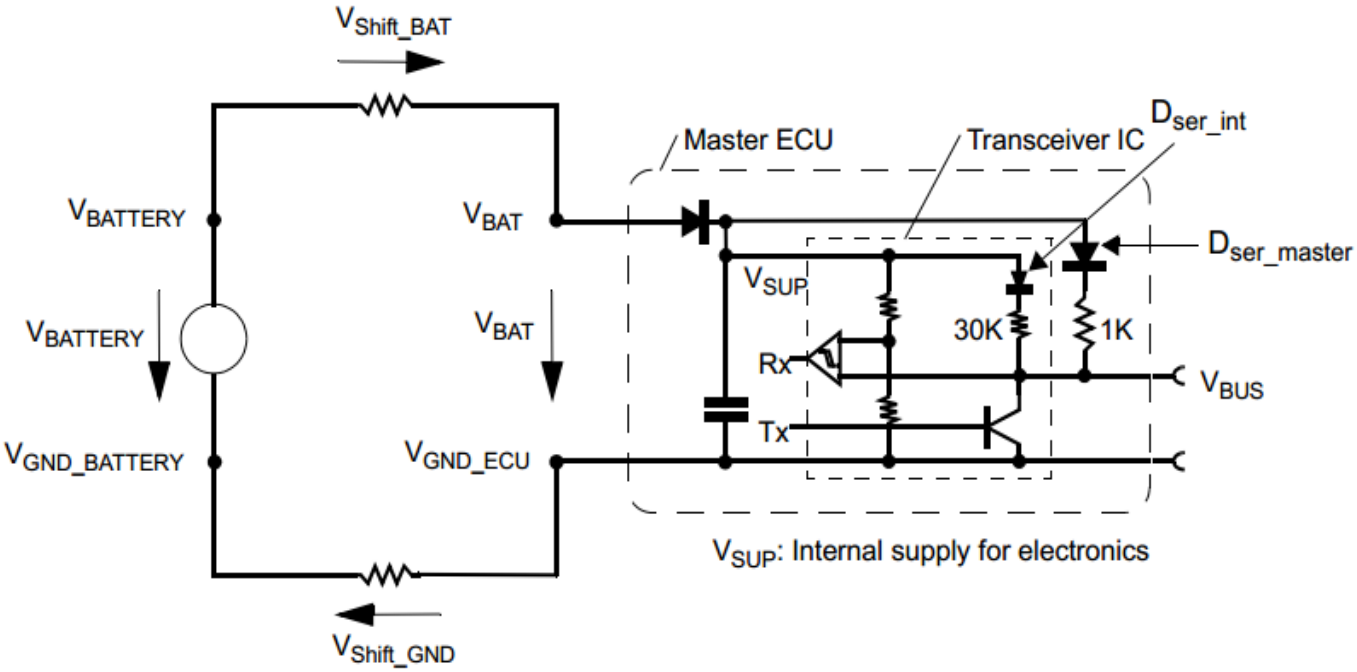
A LIN hálózat logikai felépítése

- Egy LIN hálózat egy mester és több szolga **folyamatból** áll. A mester csomópont mind mester, mind szolga feladatokat is ellát, míg a többi csomópont csak szolga folyamatokat végez.



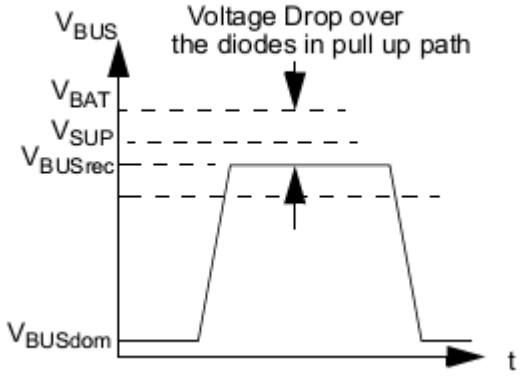
LIN Fizikai réteg

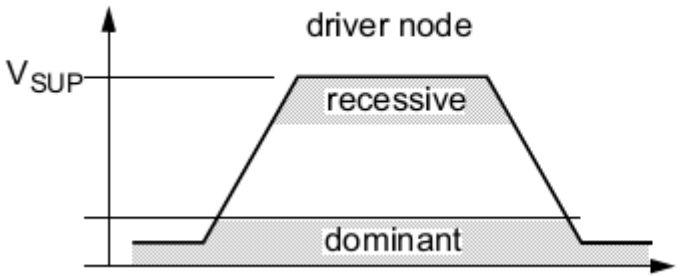
- „Egy vezetékes”
- Echo

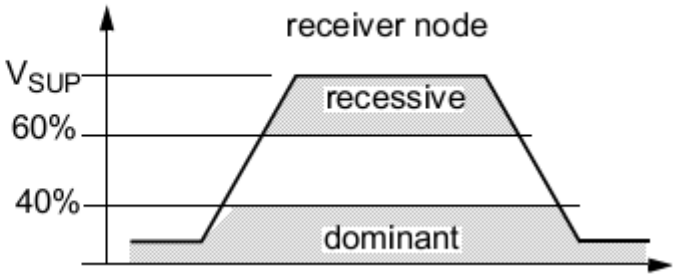


Jelszintek

- V_{BUSrec} Tipikusan 12V
- Meghajtás 20%-80%
- Fogadás 40%-60%

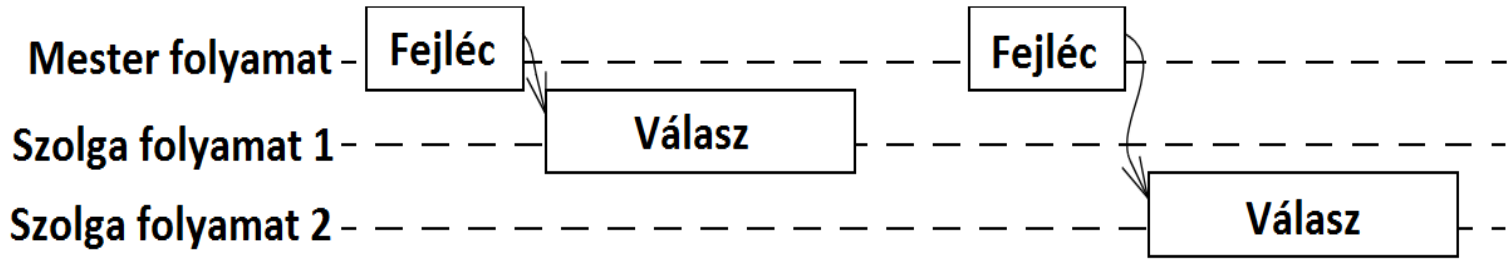






LIN Keretek
Frames

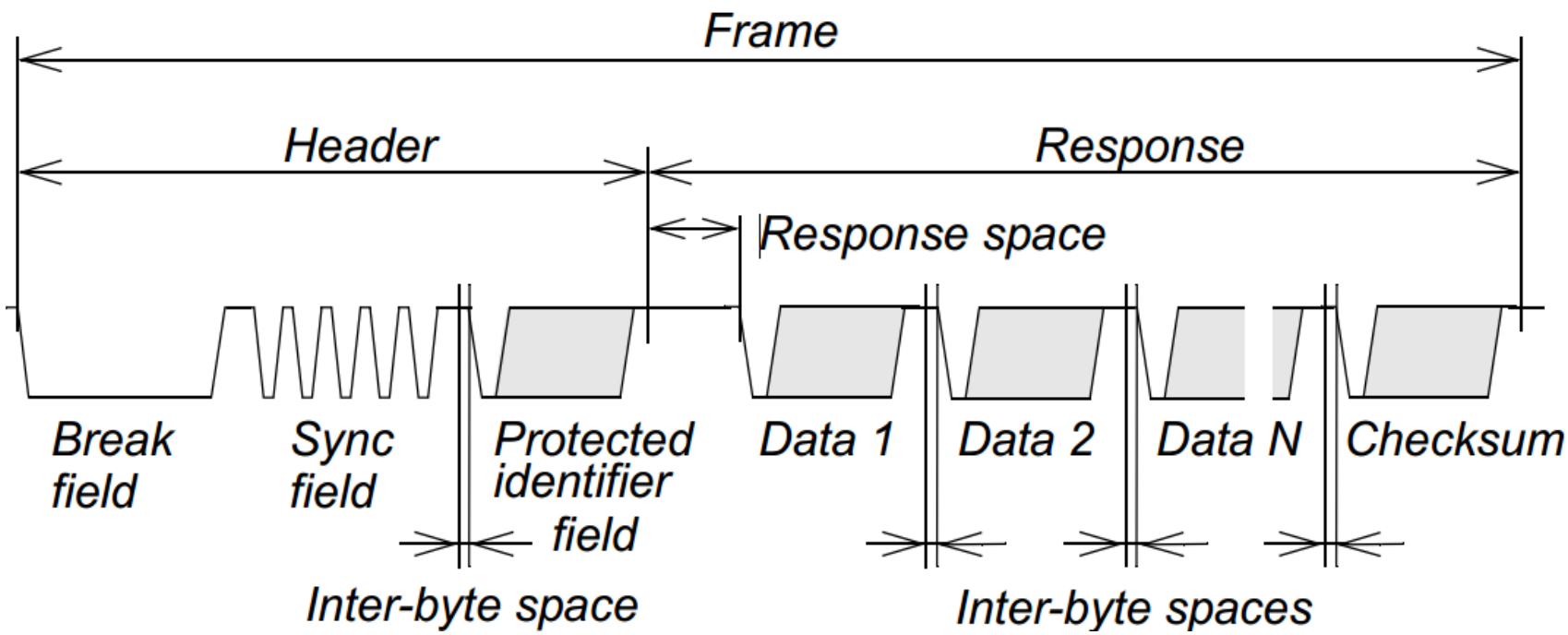
- Egy adatkeret a mesterfolyamat által szolgáltatott fejlécből (header) és a szolgafolyamat által szolgáltatott válaszból (response) áll.



LIN Keretek általános felépítése

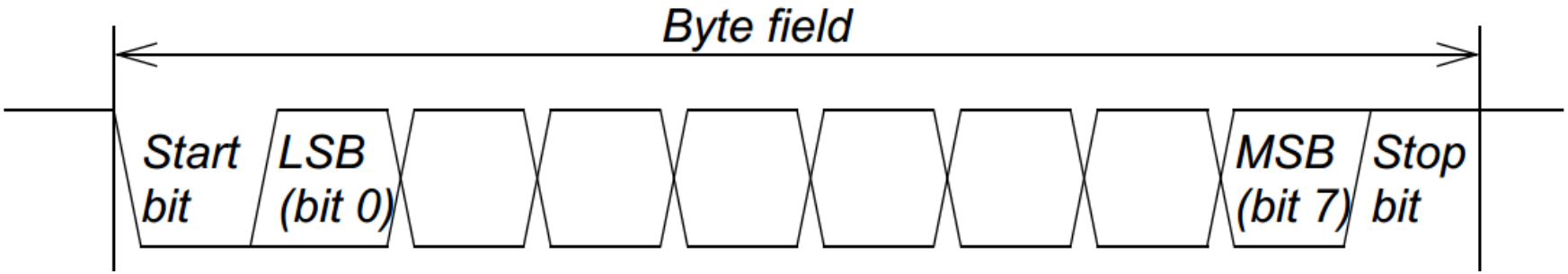
- Fejléc felépítése:
  - Break Signal (Legalább 13 bit hosszúgágú)
  - Szinkronizációs mező
  - Keretazonosító
- Válasz felépítése:
  - Adatmező
  - Ellenőrzőösszeg

LIN Keretek általános felépítése



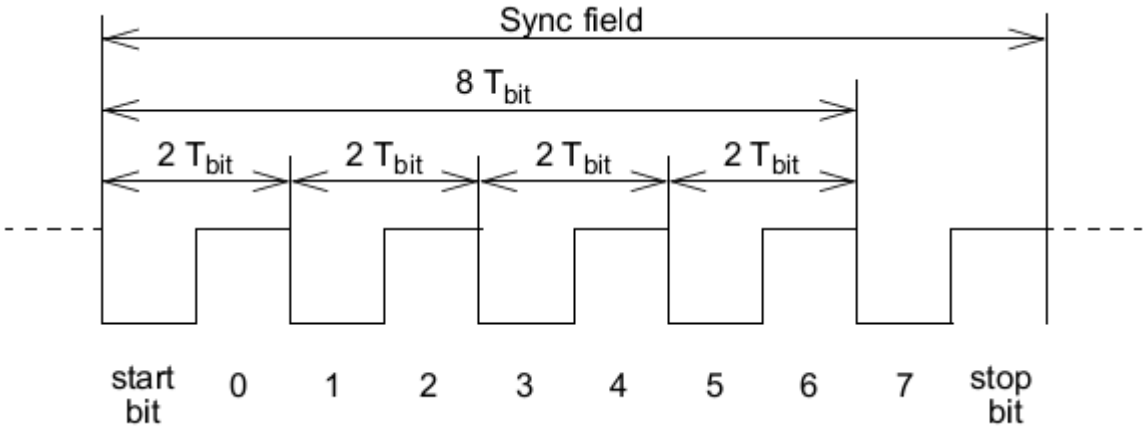
LIN keret byte mezőjének formátuma

- Start bit (Domináns 0)
- LSB-MSB adat
- Stop bit (Recesszív 1)
- (Plusz inter-byte space)



Szinkronizáció

- A szinkronizáció technikailag egy ‘0x55’ adat byte kiküldését jelenti
- A szolga node így a lefutó élek távolságából meghatározhatja a „bithosszt”



Azonosító mező 1

- A *védett azonosító* (protected identifier) mező egy olyan byte mező, amely két almezőből áll: a keretazonosítóból és a hozzá tartozó paritásból. A keretazonosító hat bitből áll, így 0..63 értéket vehet fel, és háromféle csoportba sorolható:
  - 0..59 (0x3B) alkalmazandó a jeltovábbító keretekhez;
  - 60 (0x3C) mester igény (master request) és 61 (0x3D) szolgaválasz (slave response) a diagnosztikai keretek azonosítói; illetve
  - 62 (0x3E) és 63 (0x3F) fenntartott azonosítók későbbi protokollbővítéshez.

Azonosító mező 2

- A paritás a keretazonosító bitjeiből számítandó a következő módon:

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$

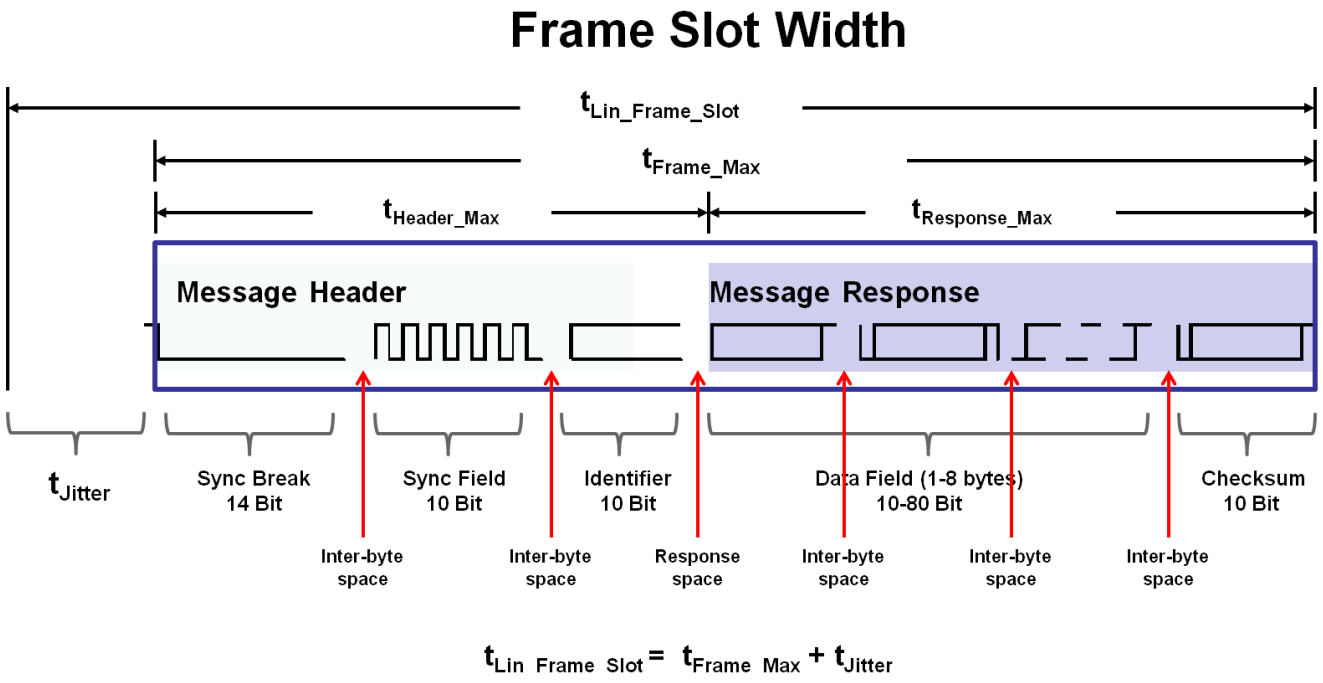
$$P1 = \neg(ID1 \oplus ID3 \oplus ID4 \oplus ID5)$$

Adatmező

- Egy keret egytől nyolc byte-ig terjedő adatot szállíthat, de a keret hossza előre definiált kell, hogy legyen. A LIN keret utolsó eleme az ellenőrzőösszeg, amely két alapvető formájú lehet:
  - Klasszikus ellenőrzőösszeg, amely az adatbyte-ok átvitellel együtt vett összege invertálva; diagnosztikai keretek, illetve a LIN 1.x protokollt használó szolgák esetében alkalmazandó.
  - Továbbfejlesztett ellenőrzőösszeg (Enhanced checksum), amely a védett azonosítót is beleveszi az összeg számolásába, amelyet a LIN 2.x protokollt alkalmazó eszközök használnak.
- Az ellenőrzőösszegek használata értelemszerűen keretenként előre definiált kell, hogy legyen.

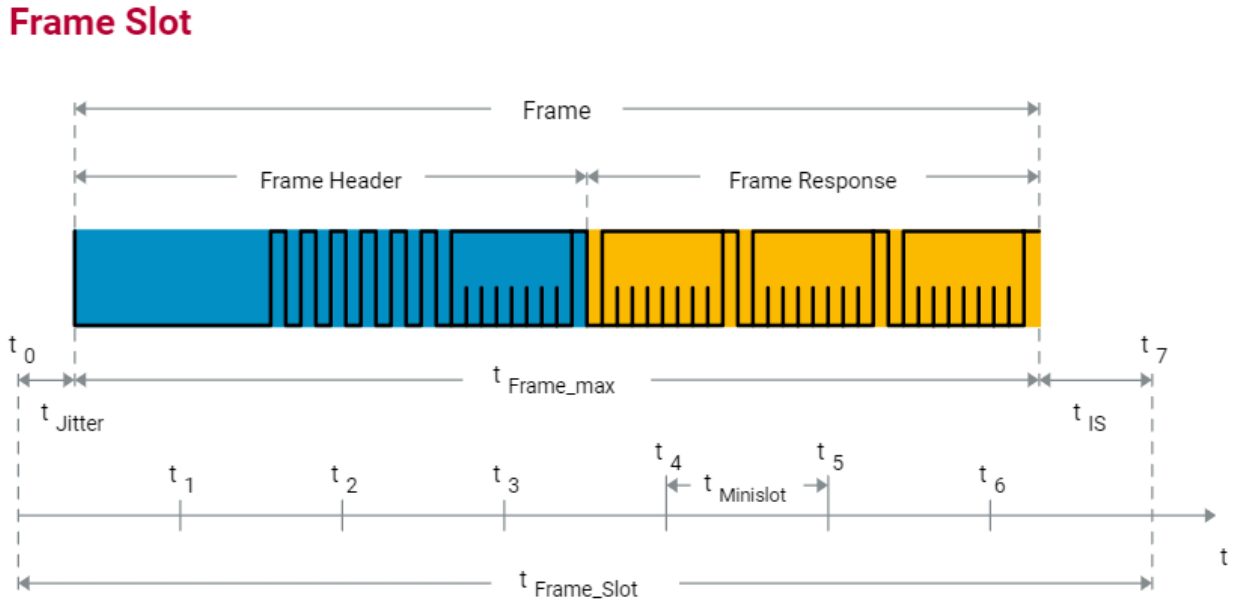
Időzítés

- A LIN keret elméleti hossza a bitidők, a válasz előtt eltelt idő, az egyes bájtok közötti idő és a mester pontatlansága alapján számolható.
- A bájtok között idő maximum a bitidőkből számolt hossz 40%-ával növelheti a keret idejét.



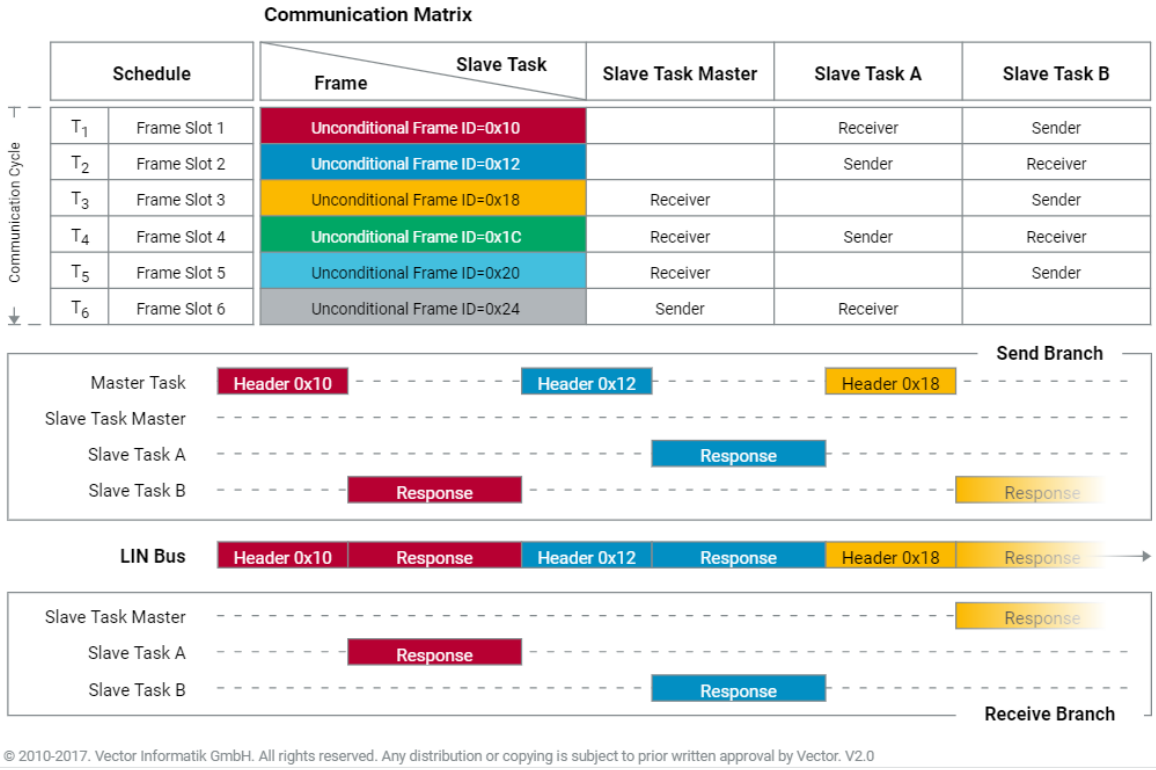
Keret időrés meghatározása

- Minden keret egy meghatározott keret pozícióban keletkezik (frame slot).A keret pozíciók hossza megegyezik az előzőekben bemutatott keret a leghosszabb elméleti hosszával, plusz egy keret-közi állandó idővel. Ezt 5 ms-os egységekben adják meg.



Időzítési tábla (Schedule table)

- A LIN mester folyamata az időzítési táblát (schedule table) alapján kéri le az üzeneteket.
- Mivel a közeghozzáférést egyedül a mesterfolyamat vezérli, ezért konfliktus csak speciális esetben alakulhat ki.



Feltétel nélküli keret (Unconditional frame)

- Egyszerű jeleket szállítanak
- Keretazonosítójuk 0 és 59 közötti értékeket vehetnek fel
- Átvitelük a hozzájuk tartozó frame slot-ban történik
- A keret fejlécét a mester folyamat írja
- Az adatbeírásra kötelezett szolga folyamat kitölti az adatmezőket
- A fogadásra kötelezett szolgák értelmezik

Eseményvezérelt keret (Event triggered frame)

- Célja a LIN hálózat sávszélességének optimalizálása
- Több szolga figyelése történhet, egyszerre ritkán fellépő események esetében
- A mester a feltételes keret fejlécét kitölti
- A válaszadó szolgák egyike feltölti
- Ekkor ritkán konfliktus fordulhat elő
- A konfliktust a mester érzékeli és kezeli

Eseményvezérelt keret konfliktuskezelés

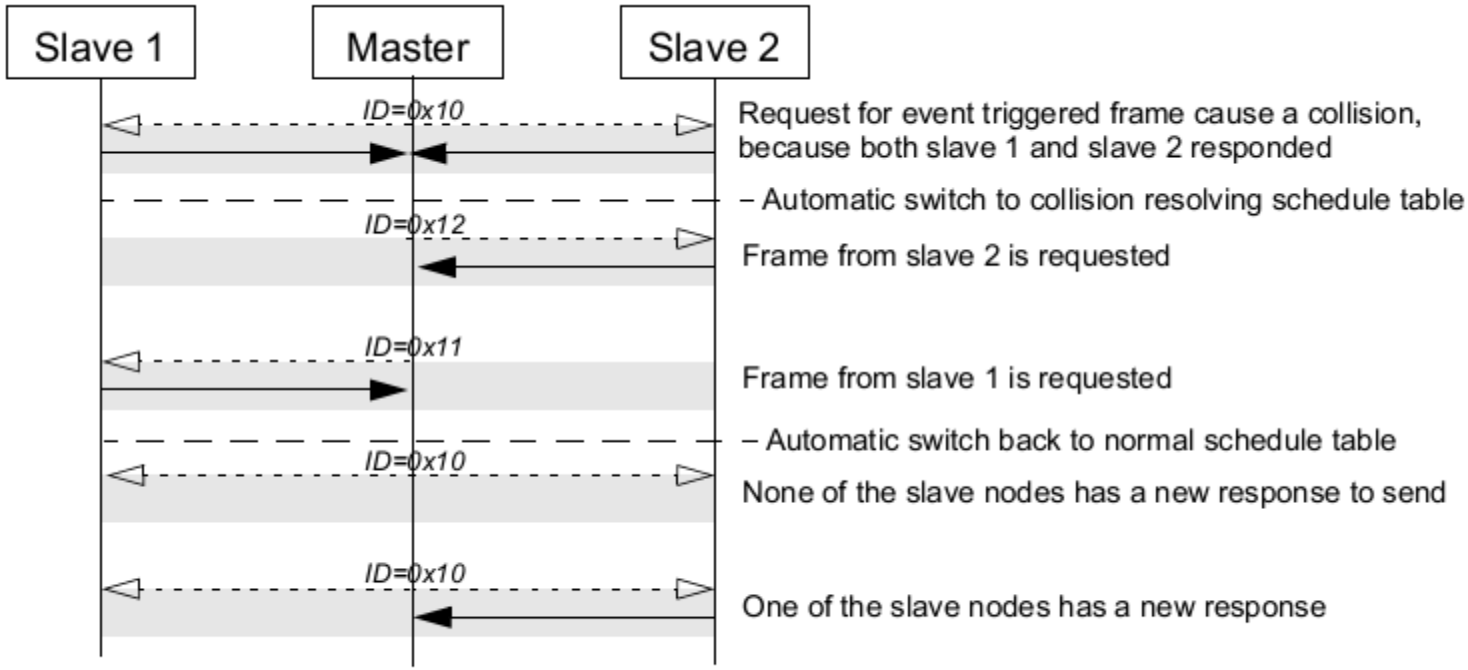


Figure 2.10: Event-triggered frame example.

Szórványosan megjelenő keret (Sporadic frame)

- Olyan adatokat hordoznak, amelyek nem ütemes rendszerességgel kell, hogy megjelenjenek a hálózaton
- Több ilyen keret foglalhatja el ugyanazt a frame pozíciót az időzítési táblában
- Időzítésükről a mester folyamat dönt

Diagnosztikai keret (Diagnostic Frames)

- A szállítási réteg információját továbbítják a hálózaton
- Két azonosítója lehet:
  - 60, mester igény (master request)
  - 61, szolga válasz (slave response)
- Több speciális információt tartalmaz az adatmezőben
  - Eszközcím (NAD – Node Address)
  - Protocol Control Information (PCI)
  - Service Identifier (SID)
  - Response identifier (RSID)

Diagnosztikai keret Felépítés, PCI

Request	NAD	PCI	SID	D1	D2	D3	D4	D5	PCI-type = SF
	NAD	PCI	LEN	SID	D1	D2	D3	D4	PCI-type = FF
	NAD	PCI	D1	D2	D3	D4	D5	D6	PCI-type = CF
Response	NAD	PCI	RSID	D1	D2	D3	D4	D5	PCI-type = SF
	NAD	PCI	LEN	RSID	D1	D2	D3	D4	PCI-type = FF

- SF: single frame
- FF: first frame
- CF: consecutive frame

Type	PCI type				Additional information			
	B7	B6	B5	B4	B3	B2	B1	B0
SF	0	0	0	0	Length			
FF	0	0	0	1	Length/256			
CF	0	0	1	0	Frame counter			

Diagnosztikai keret
LEN, NAD

- LEN: maximum üzenethossz 4095 (0xFFF). Alsó bájt a LEN mezőben, a felső 4 bit a PCI mezőben (Lásd előző dia!) helyezkedik el.
- NAD: eszköz címek az alábbi táblázat szerint kerülhetnek kiosztásra.

NAD value	Description
0	Reserved for go to sleep command, see Section 2.6.3
1 - 125 (0x7D)	Slave node addresses (NAD)
126 (0x7E)	Functional node address (functional NAD), only used for diagnostics (using the transport layer)
127 (0x7F)	Slave node address broadcast (broadcast NAD)
128 (0x80) - 255 (0xFF)	Free usage. Diagnostic frames with the first byte in the range 128 (0x80) to 255 (0xFF) are allocated for free usage since the LIN 1.2 standard. See user defined diagnostics Section 5.2.6.

Fenntartott keretek (Reserved Frames)

- A LIN2.x hálózatban a 62,63 azonosítójú keretek nem használhatóak
- Későbbi protokollbővítésre vannak fenntartva

Jelek kezelése

- Egy keretben két típusú adat kerülhet továbbításra:
  - Jel (Signal), amely egy **skalár**, vagy **byte tömb**, amely a keret adatrészébe van csomagolva. a jelek pozíciója egy adott keretazonosítón belül állandó.
  - **Diagnosztikai üzenetek**, amelyek két fenntartott azonosítójú keretben közlekedhetnek.

Skalár jelek

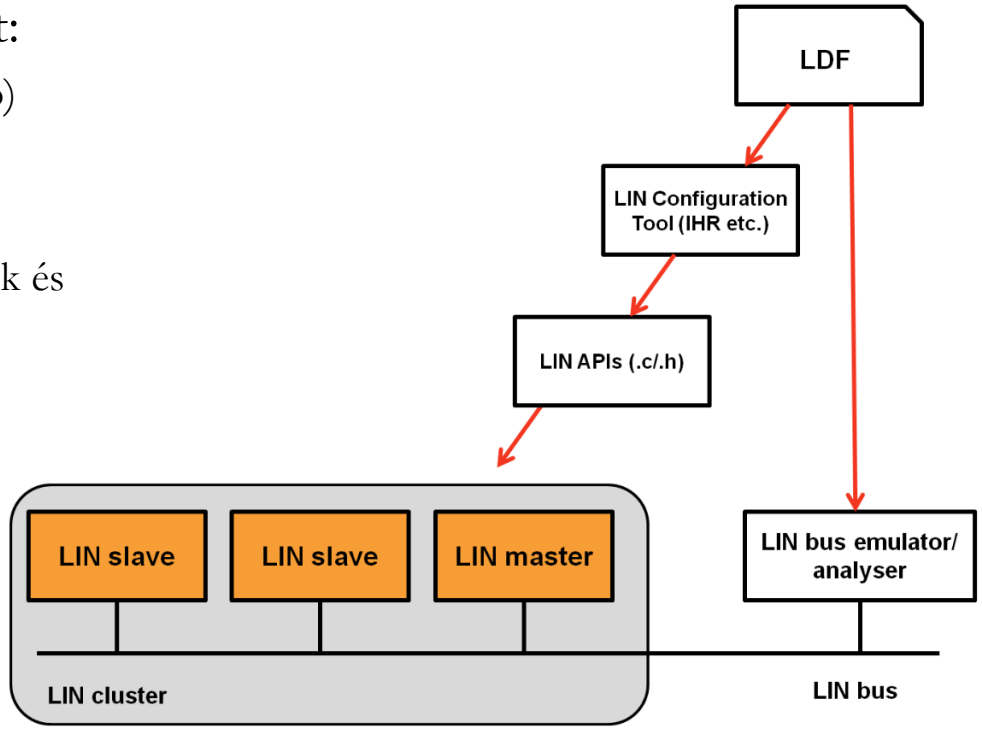
- Hosszuk 1-16 bit között mozoghat
  - Az 1 bites jelet boolean jelnek nevezzük
  - A 2-16 bitesek unsigned-ként kezelendők
- Minden jelnek egy szolgáltatója van, de azt több keretben is küldheti.
- Minden jelhez tartozik egy kezdeti érték, amely a jel első írásáig érvényesnek tekintendő
- A jelek kódolása LSB-MSB sorban történik
- A jelek átnyúlhatnak a kerethatárokon

Byte array jelek

- 1-8 byte hosszúak lehetnek
- Minden esetben kötöttek a keret byte határaihoz

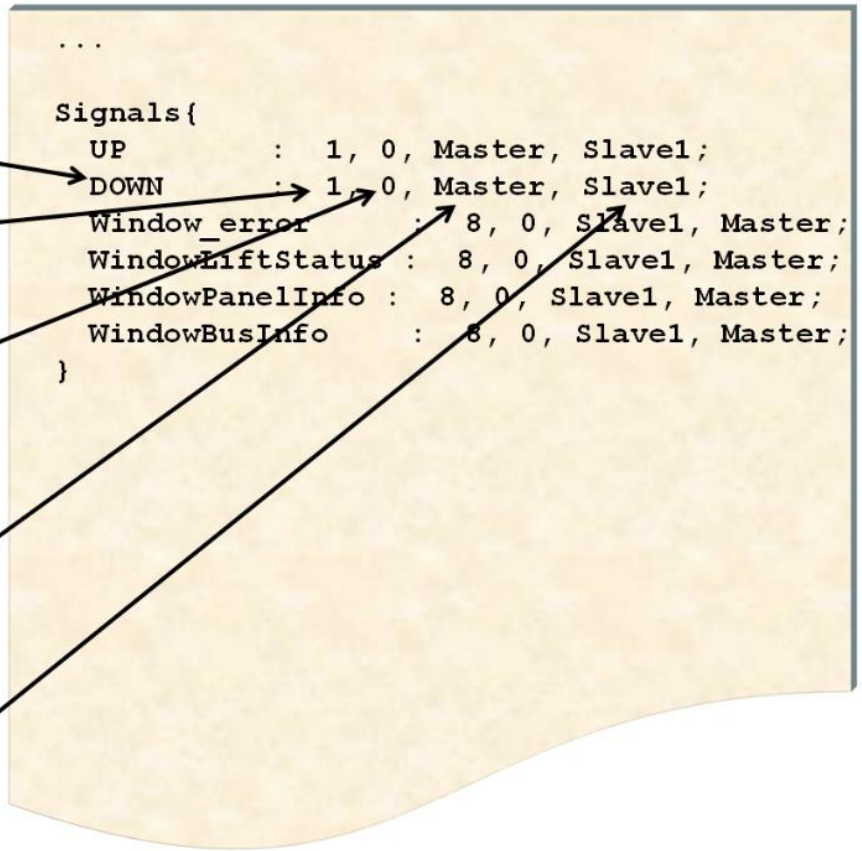
A hálózat leírása, LDF

- Szöveges fájl a hálózat leírására (konfigurációjára)
- A LIN Consortium fejlesztése
- Egy fájl leír egy teljes hálózatot:
  - Protocol Version(Protokoll verzió)
  - Language Version (Nyelv verzió)
  - Bus Speed (Busz sebesség)
  - Publishers and subscribers (küldők és fogadók)
  - Signals (Jelek)
  - Frames (Adatkeretek)
  - Schedule table (Időzítési tábla)



LDF Signal Definition Section

- **Signal name**
  - Unique names
- **Signal size**
  - Bit (0..7)
  - Byte (8)
  - Integer (16)
  - Array (16..64)
- **Initial Value**
  - After POR
- **Publisher**
  - Who sends the Response
  - Only one node
- **Subscriber**
  - Who read the Response
  - Multiple nodes possible
    - Inter slave operatibility

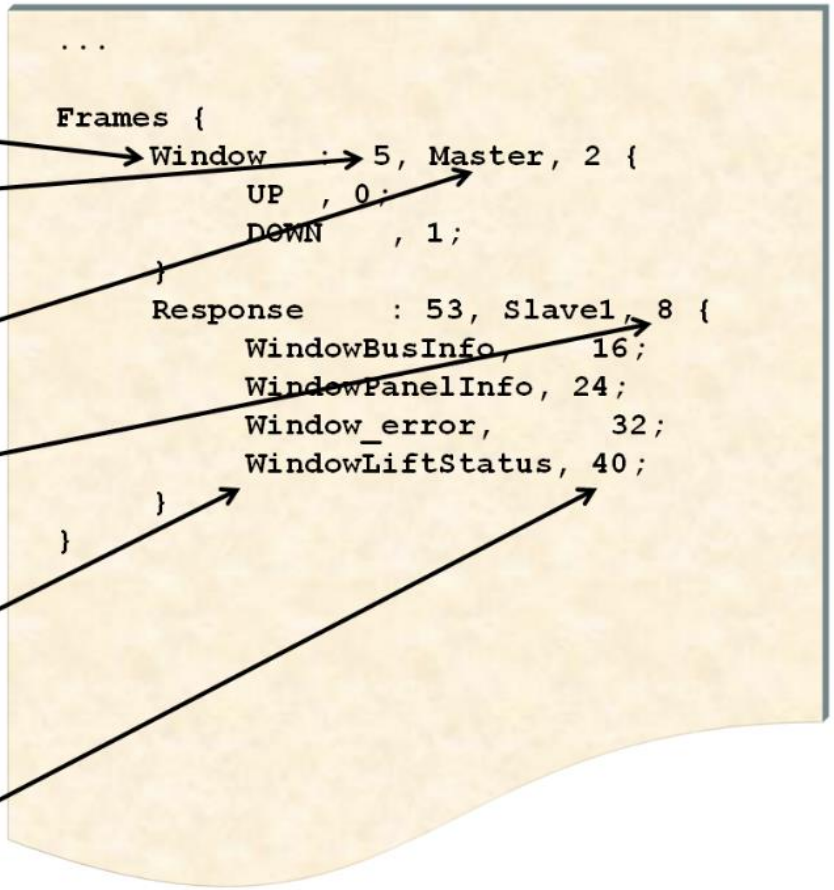


```
...

Signals{
  UP            :  1, 0, Master, Slave1;
  DOWN          :  1, 0, Master, Slave1;
  Window_error      :  8, 0, Slave1, Master;
  WindowLiftStatus  :  8, 0, Slave1, Master;
  WindowPanelInfo   :  8, 0, Slave1, Master;
  WindowBusInfo     :  8, 0, Slave1, Master;
}
```

LDF Frame Definition Section

- **Frame name**
  - Unique names
- **Identifier**
  - Unprotected
  - 0 to 64 integer
- **Publisher**
  - Answers to this identifier
- **Response length**
  - Checksum byte excluded
  - Max data response 8 bytes
- **Signals in Response**
  - Defined in section „Signals“
- **Offset**
  - Bit position where signal starts in response



Vége

Köszönöm a figyelmet!