# FlexRay Automotive Communication Bus

S. Venkatesan

Acknowledgement: The contents, example scripts and some figures are copied from various sources.
Thanks to all authors and sources made those contents public and usable for educational purpose

# Introduction

- FlexRay communications bus
  - deterministic
  - fault-tolerant
  - high-speed bus system

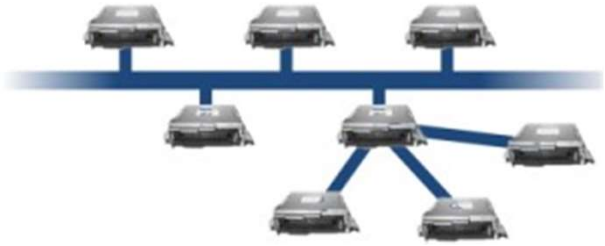
- FlexRay delivers the error tolerance and time-determinism performance requirements for x-by-wire applications (i.e. drive-by-wire, steer-by-wire, brake-by-wire, etc.).

- An alternate for CAN bus

# Wiring

- Uses **unshielded twisted pair** cabling to connect nodes together.
- Supports single- and dual-channel configurations which consist of one or two pairs of wires respectively.
- Differential signaling on each pair of wires reduces the effects of external noise on the network without expensive shielding.
- Most FlexRay nodes typically also have power and ground wires available to power transceivers and microprocessors.
- In the beginning, it uses one twisted wire and later with two twisted wire to cater the demand.
- Typical FlexRay networks have a cabling impedance between 80 and 110 ohms, and the end nodes are terminated to match this impedance.

# FlexRay Topology



https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/flexray-automotive-communication-bus-overview.html?srsltid=AfmBOoofFV-K37cbE5soxEUnqw-1Y3DT5yJjaecjfGaRV1QcCkLO8lZ6

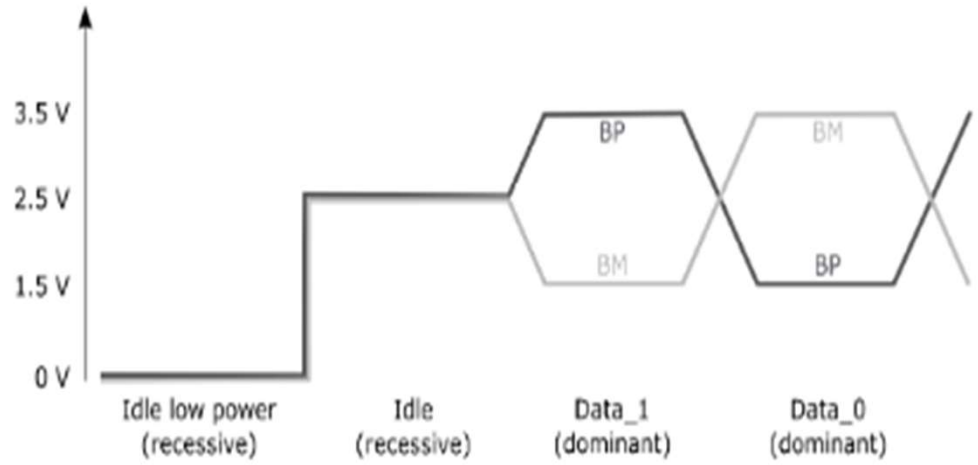
# Communication

- FlexRay manages multiple nodes with a **Time Division Multiple Access** or TDMA scheme.

- Every FlexRay node is synchronized to the same clock, and each nodes waits for its turn to write on the bus.

- Because the timing is consistent in a TDMA scheme, FlexRay is able to guarantee **determinism** or the consistency of data deliver to nodes on the network.

# FlexRay Bus Levels

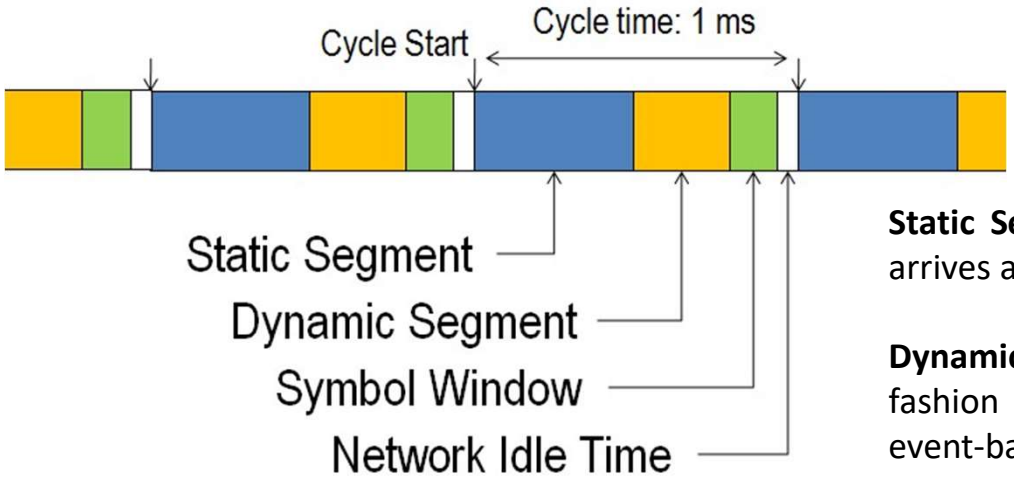



BP (bus plus) and BM (bus minus

# Communication Cycle

- The duration of a cycle is fixed when the network is designed, but is typically around 1-5 ms.

- There are four main parts to a communication cycle




**Static Segment:** Reserved slots for deterministic data that arrives at a fixed period.

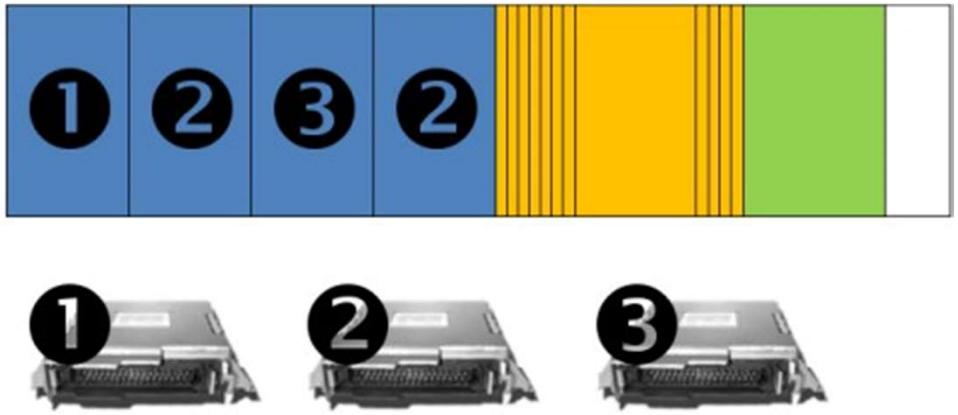
**Dynamic Segment:** The dynamic segment behaves in a fashion similar to CAN and is used for a wider variety of event-based data that does not require determinism.

**Symbol Window:** Typically used for network maintenance and signaling for starting the network.

**Network Idle Time:** A known "quiet" time used to maintain synchronization between node clocks.

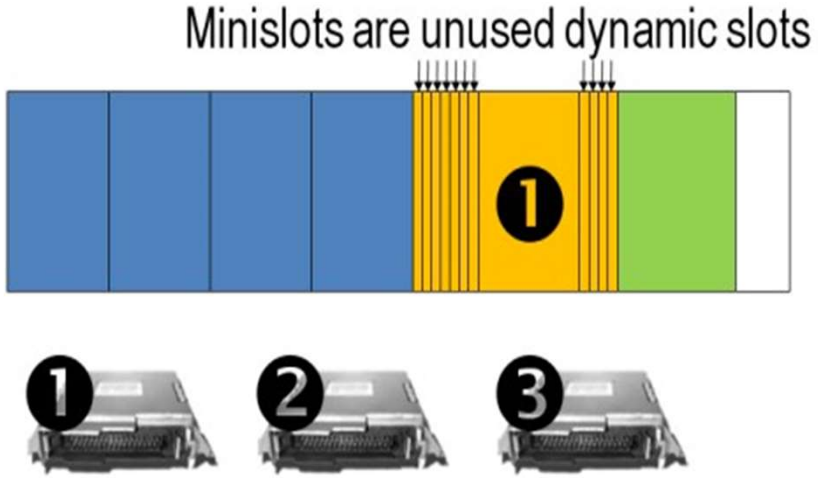
https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/flexray-automotive-communication-bus-overview.html?srsltid=AfmBOoofFV-K37cbE5soxEUnqw-1Y3DT5yJjaecjfGaRV1QcCkLO8lZ6

# Static Segment



# Dynamic Segment

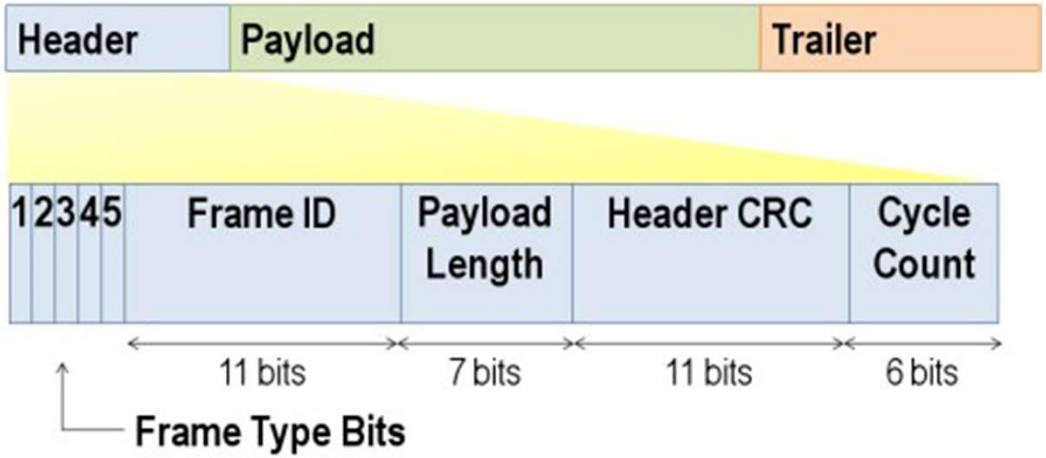
- The segment is a fixed length, so there is a limit of the fixed amount of data that can be placed in the dynamic segment per cycle.
- To prioritize the data, **minislots** are pre-assigned to each frame of data that is eligible for transmission in the dynamic segment. A minislot is typically a **macrotick** (a microsecond) long.
- Higher priority data receives a minislot closer to the beginning of the dynamic frame.

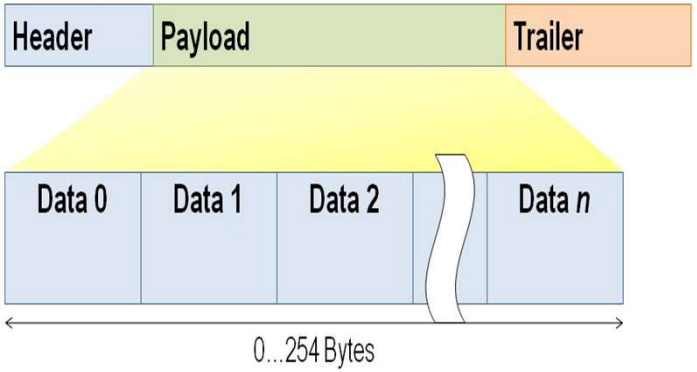



- Macrotick: This unit of measurement is typically one millisecond long and is used for perfect synchronization. So, six macro ticks are 6 milliseconds!

# Data Security and Error Handling

- Fault-tolerance by allowing single or dual-channel communication.
- For security-critical applications, the devices connected to the bus may use both channels for transferring data.
- However, it is also possible to connect only one channel when redundancy is not needed, or to increase the bandwidth by using both channels for transferring non-redundant data.



# Clock synchronization and cold starting

- FlexRay has the unique ability to sync up nodes on a network without an external synchronization clock signal.

- It uses 2 special types of frames:
  - **Startup Frames** and **Sync Frames**.

- The action of starting up the FlexRay bus is known as a **cold-start** and the nodes sending the startup frames are usually known as cold-start nodes.

- The startup frames are analogous to a start trigger, which tells all the nodes on the network to start.

- Once the network is started, all nodes must synchronize their internal oscillators to the network's macrotick.

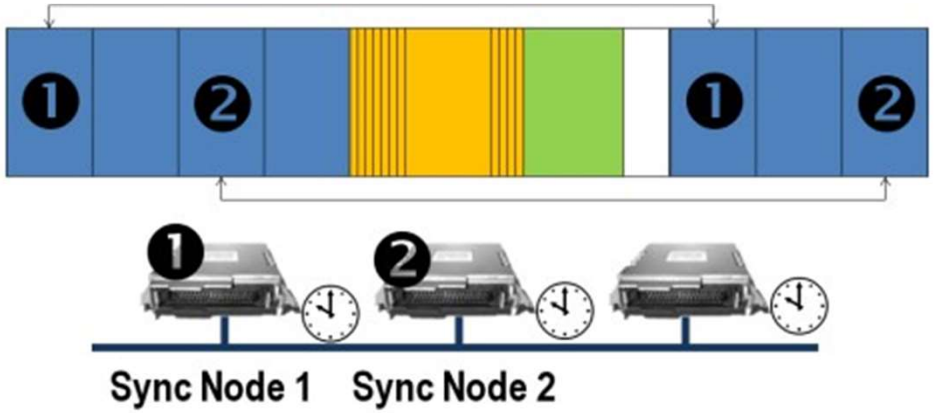
# Clock synchronization and cold starting

- This can be done using two synchronization nodes.
  - Can be any two separate nodes on the network that pre-designated to broadcast special sync frames when they are first turned on.

- Other nodes on the network wait for the sync frames to be broadcast, and measure the time between successive broadcasts in order to calibrate their internal clocks to the FlexRay time.

- The sync frames are designated in the FIBEX (FIeld Bus EXchange) configuration for the network.

- Once the network is synchronized and on-line, the network idle time (white space in the diagram) is measured and used to adjust the clocks from cycle-to-cycle to maintain tight synchronization.



# Vehicles

- Audi
- Mercedes
- BMW
- Land Rover
- Rolls-Royce Ghost

# Comparison

| | LIN | CAN | Flexray | MOST |
|---|---|---|---|---|
| **Speed** | 40 kbit/s | 1 Mbit/s | 10 Mbit/s | |
| **Wires** | 1 | 2 | 2/4 | |
| **Cost** | Low | Medium | High | |
| **Medium access or Bus access** | Polling method | CSMA-CR method | TDMA method | Timing Master |
| **Topology** | Bus topology | Bus topology | Bus/Star topology | Daisy Chain or Ring |
| **Message transmission** | Synchronous | Asynchronous | Synchronous/Asynchronous | |
| **Error checking mechanism** | Checksum over the Protected Identifier and Data fields | CRC computation over the entire frame | Two CRC: Header and Trailer CRC for entire frame | |
| **Cabling impedance** | 1k ohms | 120 ohms | 80-110 ohms | |
| **Range** | 40 meters | 40 meters | 10 meters | |
| **Data Length** | 2, 4 and 8 bytes | 8 bytes | 254 bytes | |
| **Communication** | Event Triggered however on request | Event Triggered | Time/Event triggered | |
| **Year** | 2002 | 1979 | 2006 | 2008 |

# Frame Format

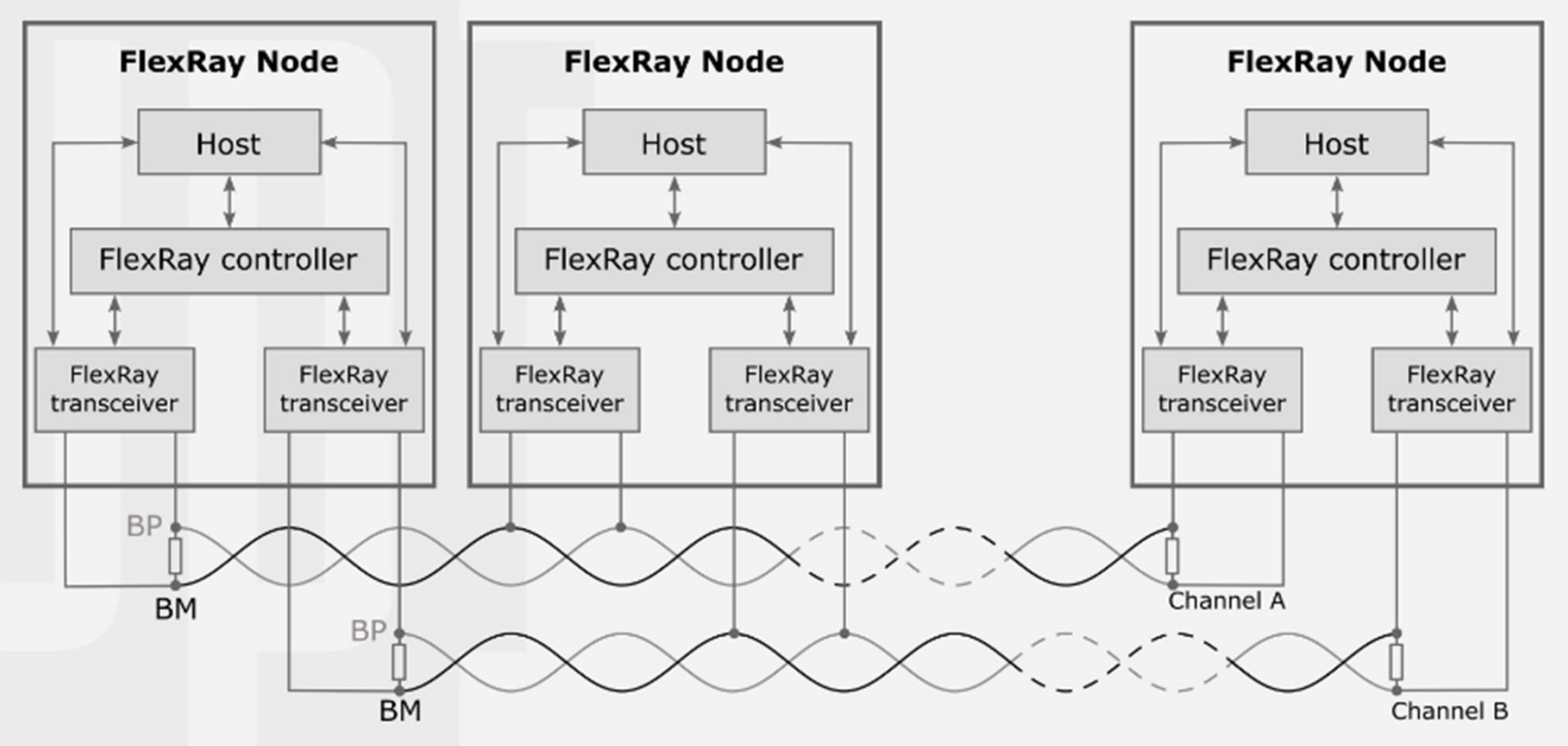
- Transmission Start Signal (TSS) – bit 0
- Frame Start Signal (FSS) – bit 1
- *m* times:
  - Byte Start Signal 0 (BSS0) – bit 1
  - Byte Start Signal 1 (BSS1) – bit 0
  - 0th bit of *i*-th byte
  - 1st bit of *i*-th byte
  - 2nd bit of *i*-th byte
  - ...
  - 7th bit of *i*-th byte
- Frame End Signal (FES) – bit 0
- Transmission End Signal (TES) – bit 1

# Idle State and Frame

- If nothing is being communicated, the bus is held in state 1 (high voltage), so every receiver knows that the communication started when the voltage drops to 0.

- Note that 8-cycle per bit has nothing to do with bytes. Each byte takes 80 cycles to transfer. 16 for BSS0 and BSS1 and 64 for its bits. Also note that BSS0 has value 1, and BSS1 has value 0.

# Layers


FlexRay Node
Host
FlexRay controller
FlexRay transceiver
FlexRay transceiver
FlexRay Node
Host
FlexRay controller
FlexRay transceiver
FlexRay transceiver
FlexRay Node
Host
FlexRay controller
FlexRay transceiver
FlexRay transceiver
BP
BM
BP
BM
Channel A
Channel B

# Clock synchronization

- Clocks are resynchronized when the voted signal changes from 1 to 0, if the receiver was in either idle state or expecting BSS1.

- As synchronization is done on the voted signal, small transmission errors during synchronization that affect the boundary bits may skew the synchronization no more than 1 cycle.

# Synchronization

- Errors that happened in the example:
  - Because of a single-bit error during synchronization, the synchronization was delayed by 1 cycle
  - Receiver clock was slower than sender clock, so receiver missed one cycle (marked X). This will not happen again before the next synchronization due to limits on maximum allowable clock drift.
  - Because of a single-bit error during transmission, a bit was voted wrongly near the result.
  - Despite so many errors, the communication was received correctly.
  - The green cells are sampling points. All except the first are synchronized by the 1->0 edge in the transmission fragment shown.

| Signal to be sent | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|
| Signal sent | 11111111 | 00000000 | 11111111 | 00000000 | 11 |
| On the bus | 11111111 | 01000000 | 11111111 | 00000010 | 11 |
| Received | 11111111 | 01000000 | 111111X1 | 00000010 | 11 |
| 5-maj voted | 11111110 | 10000001 | 11111X10 | 00000101 | 1 |

# Local Interconnect Network (LIN) Protocol


- The LIN bus provides a total of 64 IDs. IDs 0 to 59 are used for signal-carrying (data) frames, 60 and 61 are used to carry diagnostic data, 62 is reserved for user-defined extensions, and 63 is reserved for future protocol enhancements.

- The LIN bus uses a master/slave approach that comprises a LIN master and one or more LIN slaves.

- Comprising 16 nodes (one master and up to 15 slaves).

- All messages are initiated by the master with at most one slave replying to a given message identifier. The master node can also act as a slave by replying to its own messages.

- Break is to identify the Start of frame

# Media Oriented Systems Transport

- The serial MOST bus uses a daisy-chain topology or ring topology and synchronous serial communication to transport audio, video, voice and data signals via plastic optical fiber (POF) (MOST25, MOST150) or electrical conductor (MOST50, MOST150) physical layers.

# FlexRay Attacks

- Physical layer – Full DoS Attack, Targeted DoS attack (Static or Dynamic)

- Data-link layer - Full DoS attack (loss of synchronization), Targeted DoS attack (static or dynamic), Message spoofing.

| | Static | Dynamic |
|---|---|---|
| Full DoS Attack | Yes | Yes |
| Targeted DoS Attack | Yes | Yes |
| Message Spoofing | No (if occupied) | Yes |

# Solutions

- Send null frames in their slot to avoid message spoofing.
  - However hard in the dynamic segment

# References [Accessed on 2/9/2024]


- https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/introduction-to-the-local-interconnect-network-lin-bus.html
- https://www.influxbigdata.in/flexray
- https://www.prodigytechno.com/difference-between-lin-can-and-flexray-protocols
- https://www.aut.upt.ro/~pal-stefan.murvay/teaching/nes/Lecture_08_FlexRay.pdf
- https://en.wikipedia.org/wiki/FlexRay
- https://www.aut.upt.ro/~pal-stefan.murvay/papers/practical-security-exploits-flexray-in-vehicle-communication-protocol.pdf
- https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/flexray-automotive-communication-bus-overview.html?srsltid=AfmBOopycFhRc1y6-M5WQt-6A07EcrZ29jaihIxobA46weJZm913EepD#section--1927873432