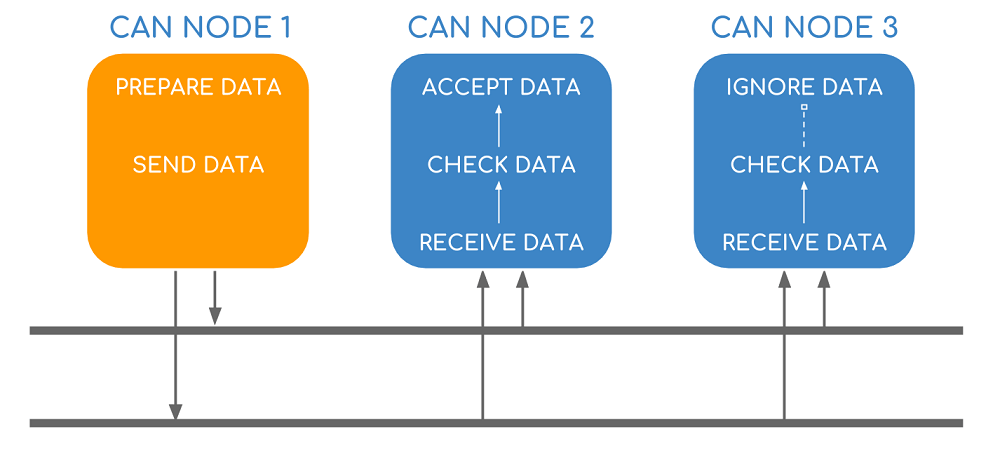
~ COPERTINA ~

1. What is it
2. How it’s done
3. Timing analysis

CONTROLLER AREA NETWORK (CAN)

The CAN is a protocol largely used in automotive that allows a robust communication between several ECUs and sensors on board a vehicle. Each ECU (e.g. engine control unit, airbags, automatic transmission, ESP, audio system) represents a node connected to the CAN bus. Nowadays, cars have around 70 ECUs. CAN system is not a point-to-point communication system, it is instead a broadcast transmission typology, making it a smarter, cheaper and lighter solution than point-to-point.

The main purpose of the CAN is to allow any ECU to communicate with the entire system without causing an overload to the controller computer. The ECUs communicate through a single CAN interface and the system allows for central error diagnosis and configuration among all ECUs.



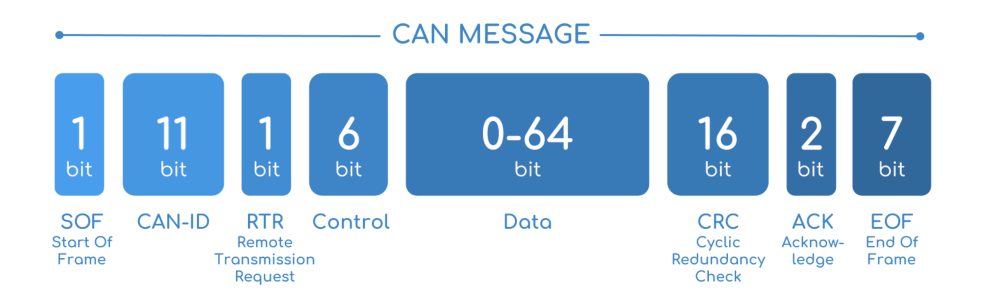
Every node receives all transmitted messages, decides relevance and chooses whether it needs the data on the bus or not. CAN messages are prioritized according to each node’s ID (lower ID means higher priority), making it a very efficient system. Furthermore, it is a flexible protocol since it is possible to include additional nodes over time.

Each ECU reads the bus through a buffer and each ECU writes on the bus through a transistor. The bus is also called a “wired AND” because the 0 level (0 V ground, logical bit value 0) is the dominant level: if one ECU writes a 0 on the bus, the logical value on the bus will be a 0 regardless of what other ECUs are writing.

If two different nodes are waiting for another to end transmission, they will probably start transmitting together once the bus is free. To avoid collisions on the bus, bitwise arbitration is used: all the ECUs with a transmission request start simultaneously to send the identifier of their respective CAN message to be transmitted, bitwise from the most significant to the least significant bit. Knowing that a 0 bit is dominant on the bus, each ECU compares the vale in the bus with the value it sent (bit monitoring) and if it reads a 0 when it pushed a 1 it means that there is another ECU with higher priority that is willing to transmit. Lower priority ECU will then stop transmitting and start listening to the higher priority ECU transmitting. This is why lower IDs have higher priority.

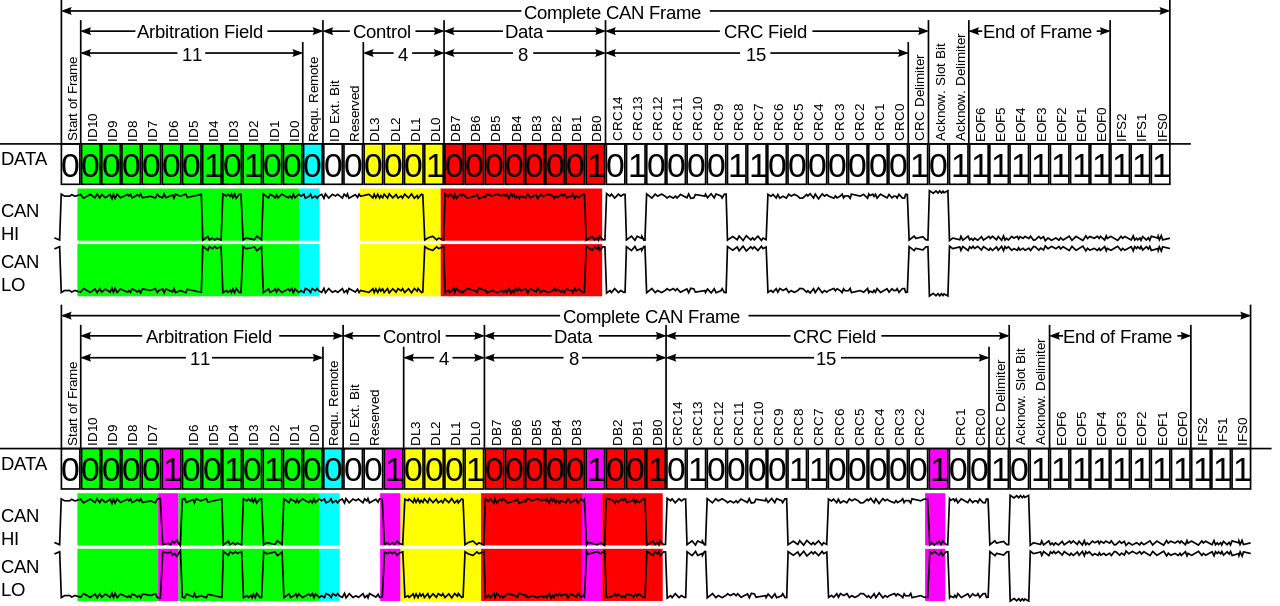
The whole message sent by a node could be subdivided into 8 different blocks:

* Start Of Frame (SOF): a dominant 0 to tell the other ECUs that a message is coming;
* Identifier (ID): gives each message a priority;
* Remote Transmission Request (RTR): a forced transmission from other ECUs;
* Control: informs the length of the data expressed in Bytes (0 to 8 B);
* Data: the actual information;
* Cyclic Redundancy Check (CRC): check used to ensure data integrity (i.e. detects errors);
* Acknowledgment (ACK): a bit that indicates if the CRC process is correct;
* End Of Frame (EOF): marks the end of a CAN message.



At the end of a message, 3 bits of interframe are used as well.

Not to lose synchronization among ECUs in long consecutive bit sequences, bit stuffing is performed by the transmitting node from the beginning of a message up to the CRC (not the CRC delimiter). Stuffing adds an opposite bit after five consecutive equal bits (purple bits in the following figure).



CAN bus’s speed is 1 Mb/s for a 40 m long bus. The speed decreases with longer buses to 40 Kb/s in a 1 Km long bus.

The push for increased vehicle functionality may require changes to the core CAN technology. Increased functionality could for example mean an increase in the data load or a way not to leave precious data unharvested.

The former is allowed by CAN FD (i.e. Flexible Data-rate) which increases the payload by a factor 8 and allows for a higher data bit rate (i.e. up to 8 Mb/s with a with a payload of 64 B).

The latter is allowed by CANopen, used to optimize standard CAN applications. CANopen is extensively used in industrial robotics, production machinery, medical equipment and speciality vehicles because they all need a kind of memory built-in. The collection of data is very important even if you don’t really know how to use those data and CANopen, built on CAN protocol, can in addition record data. CANopen provides a protocol that standardizes communication between devices and applications from different manufacturers.

* Process Data Objects (PDO)
* Service Data Objects (SDO)
* SYNC
* NMT
* EMCY

HOW WE IMPLEMENTED THE CAN