CIS 721 - Real-Time Systems

Lecture 16: Real-Time Communication

Mitch Neilsen

neilsen@cis.ksu.edu

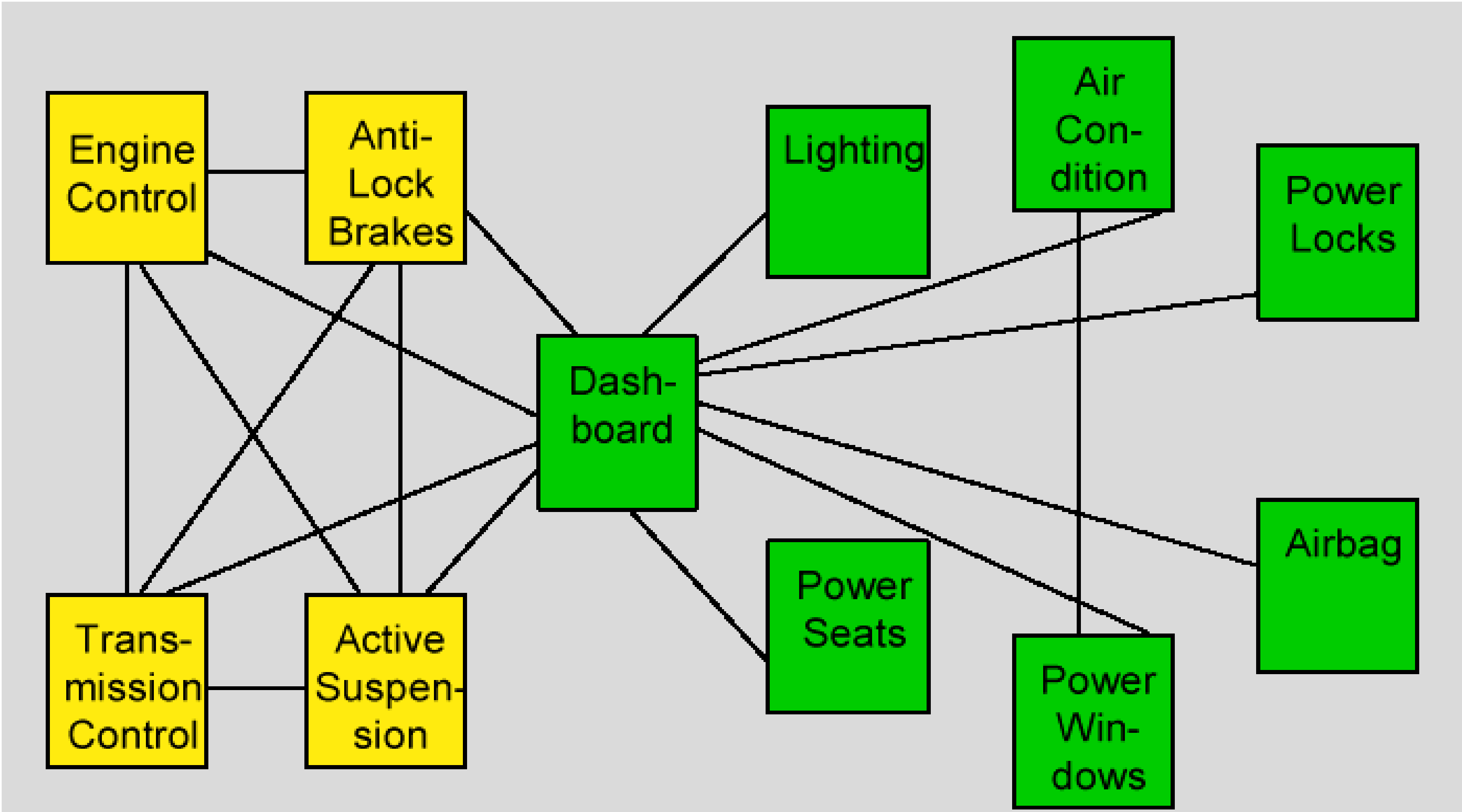
Outline

- Real-Time Communication
  - Controller Area Network
  - CAN Higher Layer Protocols
    - CAN Kingdom
    - DeviceNet
    - CANOpen
    - CiA
  - Time-Triggered Protocols (TTP/C, TTP/A,TTCAN)

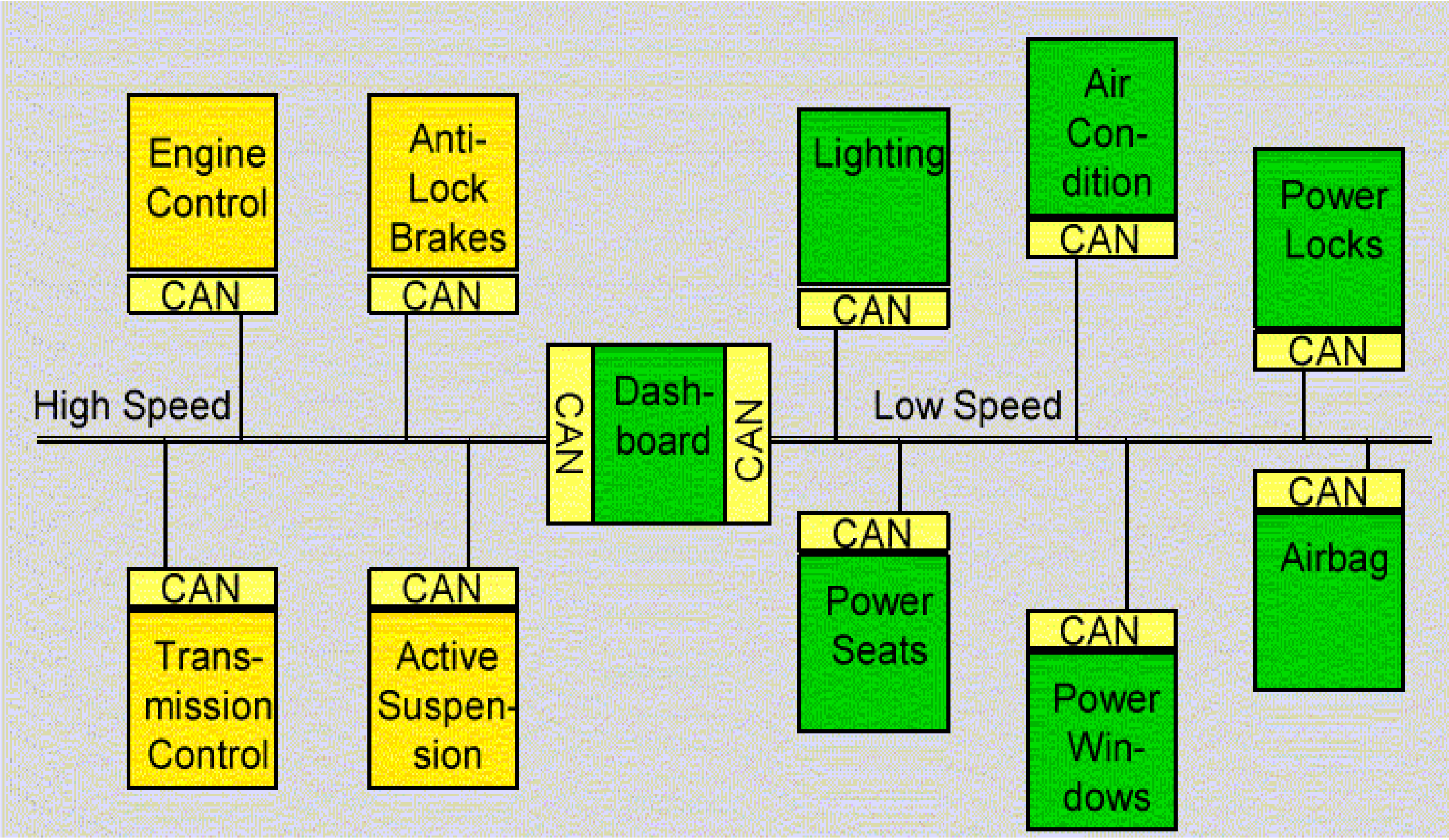
Controller Area Network

- A **Controller Area Network (CAN)** is an advanced serial bus system that efficiently supports distributed, real-time control.
- Originally developed for use in automobiles by Bosch GmbH, Germany, in the late 1980s (refer to **can2spec.pdf**).
- CAN is internationally standardized by the International Organization for Standardization (ISO 11898) and others standards organizations.

Typical Automobile Wiring

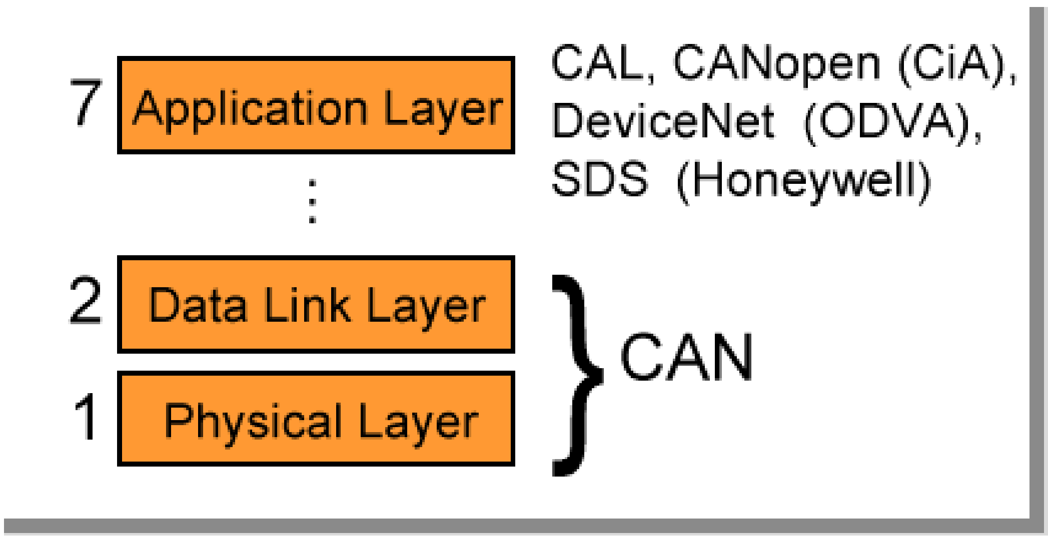


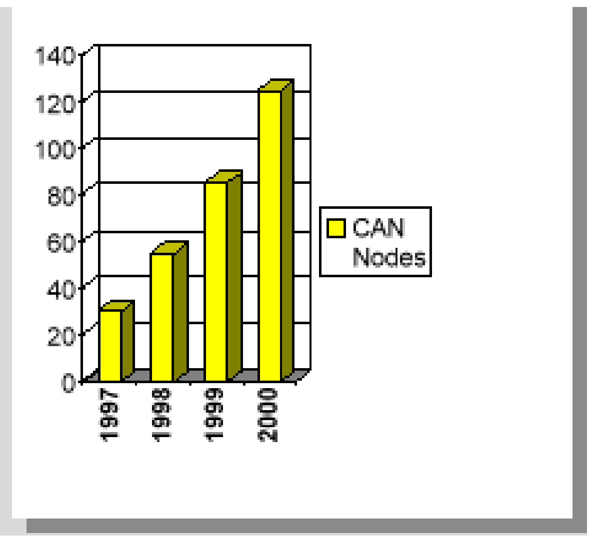
Automobile Wiring With CAN



ISO Reference Model

- CAN protocols fall within the Data Link Layer and the Physical Layer.
- CAN is most widely used in the automotive and industrial automation markets.





CAN Operating Principles

- Data messages transmitted on a CAN bus do not contain the address of either the sender or the receiver.
- The content of each message is labeled by an **identifier** that is unique throughout the network.
- All other nodes on the network receive the message (broadcast).
- Messages are filtered based on their relevance. If relevant, the message is processed by the receiver (multicast).

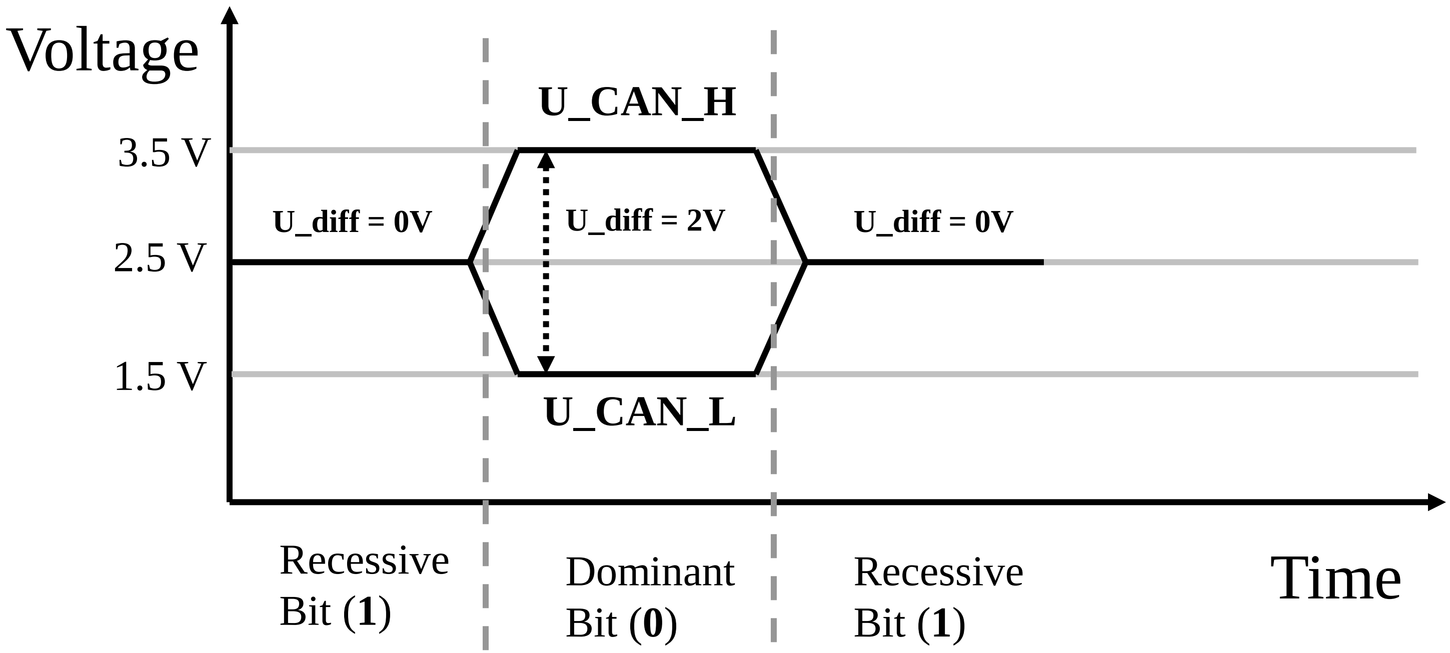
Identifiers

- The **unique identifier** also determines the priority of the message. The lower the numerical value of the identifier, the higher the priority.
- The highest priority message is guaranteed to gain access to the bus.
- Lower priority messages are automatically retransmitted in the next bus cycle, or in a subsequent bus cycle if there are still other, higher priority messages waiting to be sent.

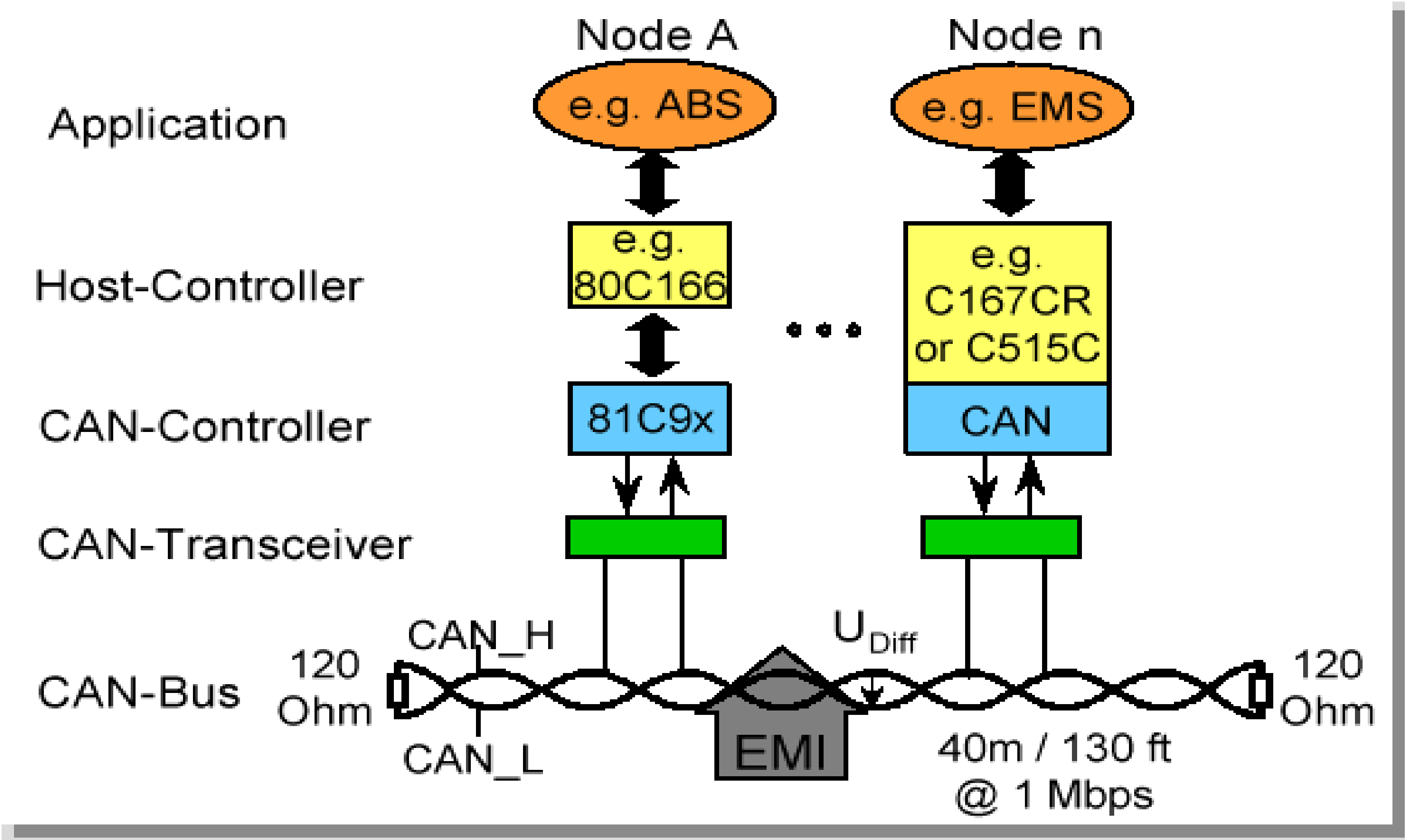
Physical Layer

- The two wire bus is usually **twisted pair** (either shielded or unshielded). Flat pair (telephone type) cable also performs well, but generates more noise, and may be more susceptible to external sources of noise (EMI).
- **Non Return to Zero (NRZ)** bit encoding (with bit-stuffing) for data communication on a differential two wire bus. NRZ encoding ensures compact messages with a minimum number of transitions and high resilience to external disturbance.

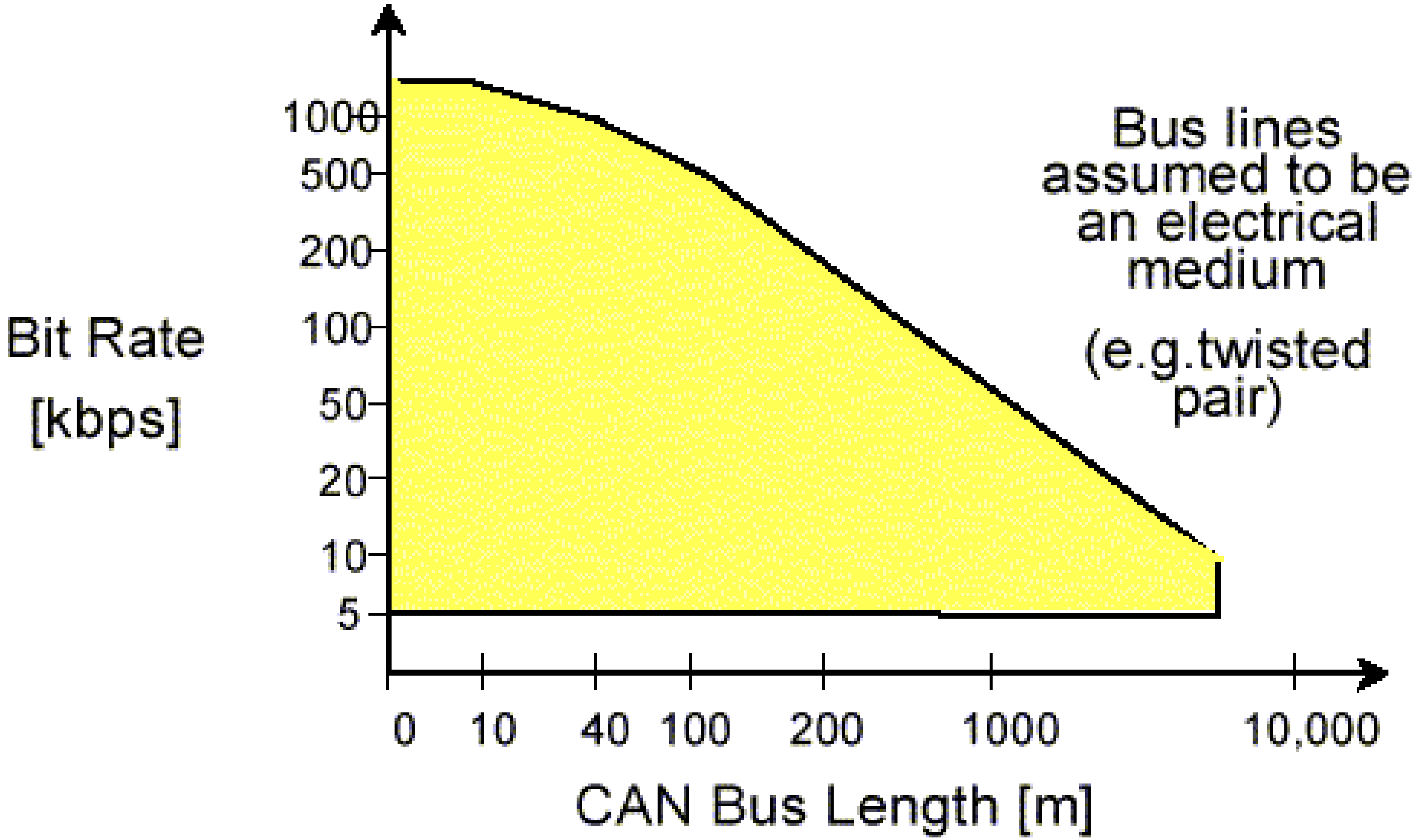
Recessive and Dominant Bits



Basic Concepts



Bus Length vs. Bit Rate



Robustness

- CAN will operate in extremely harsh environments and the extensive error checking mechanisms ensure that any transmission errors are detected.
- The ISO11898 standard “recommends” that bus interface chips be designed so that communication can still continue even if:
  - Either of the two wires in the bus is broken.
  - Either wire is shorted to power.
  - Either wire is shorted to ground.

Network Flexibility

- New nodes that only receive data, and only need existing transmitted data, can be added to the network (while the network continues to operate) without the need to make any changes to existing hardware or software.
- Measurements needed by several controllers can be transmitted via the bus, thereby removing the need for each controller to have its own individual sensor.

Medium Access Control (MAC)

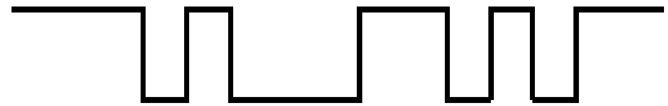
- CAN uses a Carrier Sense, Multiple Access with Collision Detection Protocol (CSMA/CD).
- Unlike Ethernet, when frames are transmitted at the same time, **non-destructive bitwise arbitration** allows the highest priority message to gain bus access.

Bitwise Arbitration

- The priority of a CAN message is determined by the binary value of its identifier.
- The numerical value of each message identifier is assigned during the initial phase of system design.
- The identifier with the lowest numerical value has the highest priority. Bus conflicts are resolved by **non-destructive bit-wise arbitration**, in accordance with the wired-and mechanism where dominant bits (0) overwrite recessive bits (1).

Wired-AND Mechanism

Two logic states
possible on the bus:
“1” = recessive
“0” = dominant



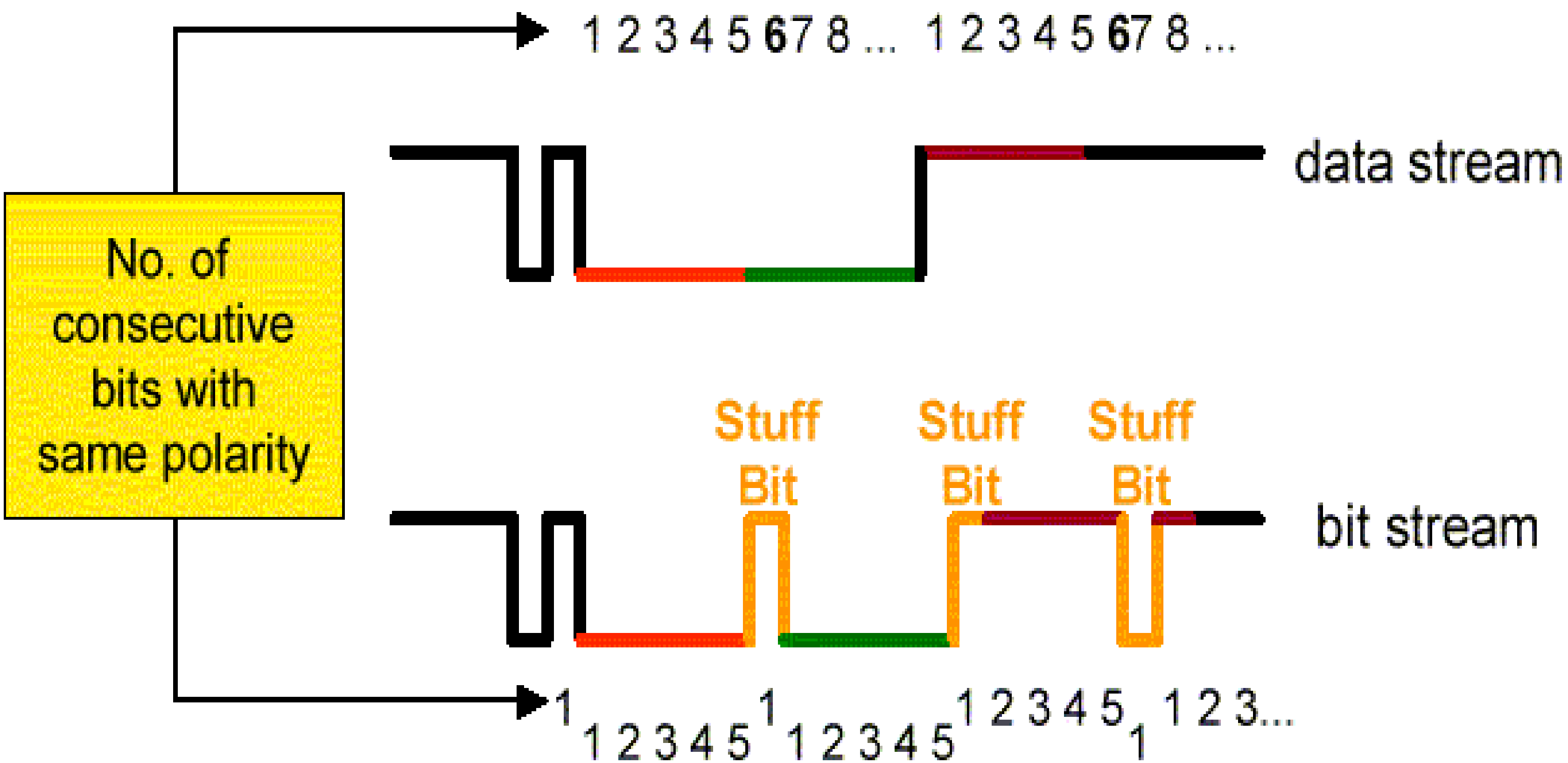
“1” = recessive
“0” = dominant

A	B	C	BUS
D	D	D	D
D	D	R	D
D	R	D	D
D	R	R	D
R	D	D	D
R	D	R	D
R	R	D	D
R	R	R	R

As soon as one node nodes transmits
a dominant bit (zero):
Bus is in the dominant state.

Only if all nodes transmit
recessive bits (ones):
Bus is in the recessive state.

Bit Stuffing



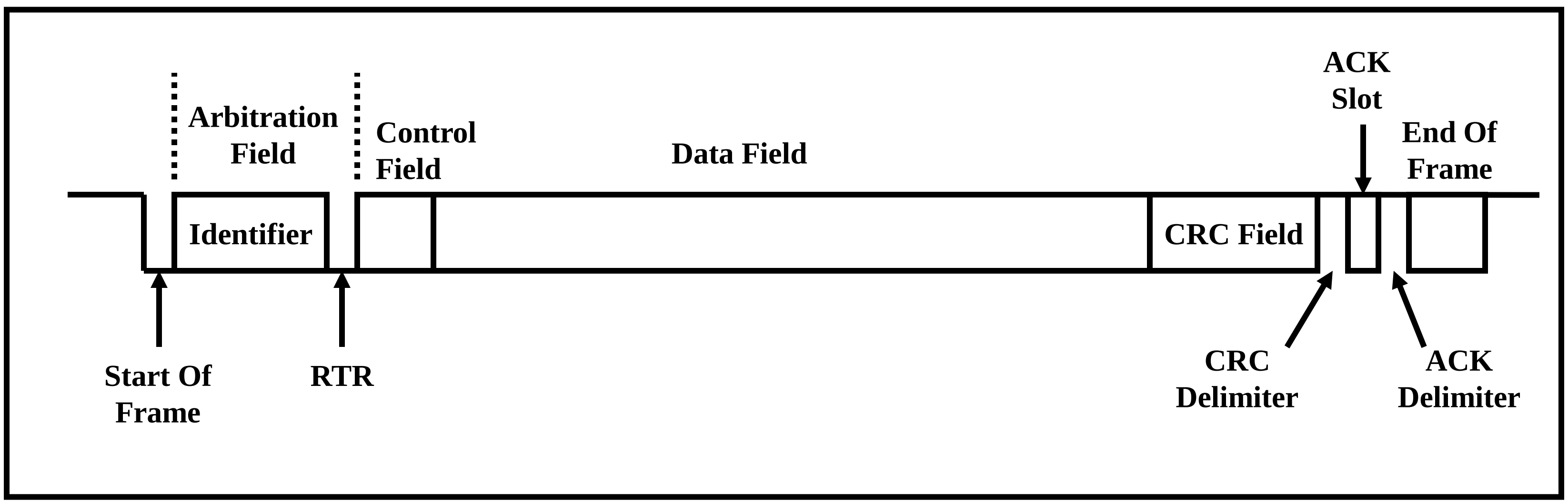
CAN Message Format

- In a CAN system, data is transmitted and received using Message Frames. Message Frames carry data from a transmitting node to one, or more, receiving nodes.
- The Standard CAN protocol (version 2.0A) supports messages with 11 bit identifiers.
- The Extended CAN protocol (version 2.0B) supports both 11 bit and 29 bit identifiers.

CAN Data Frame

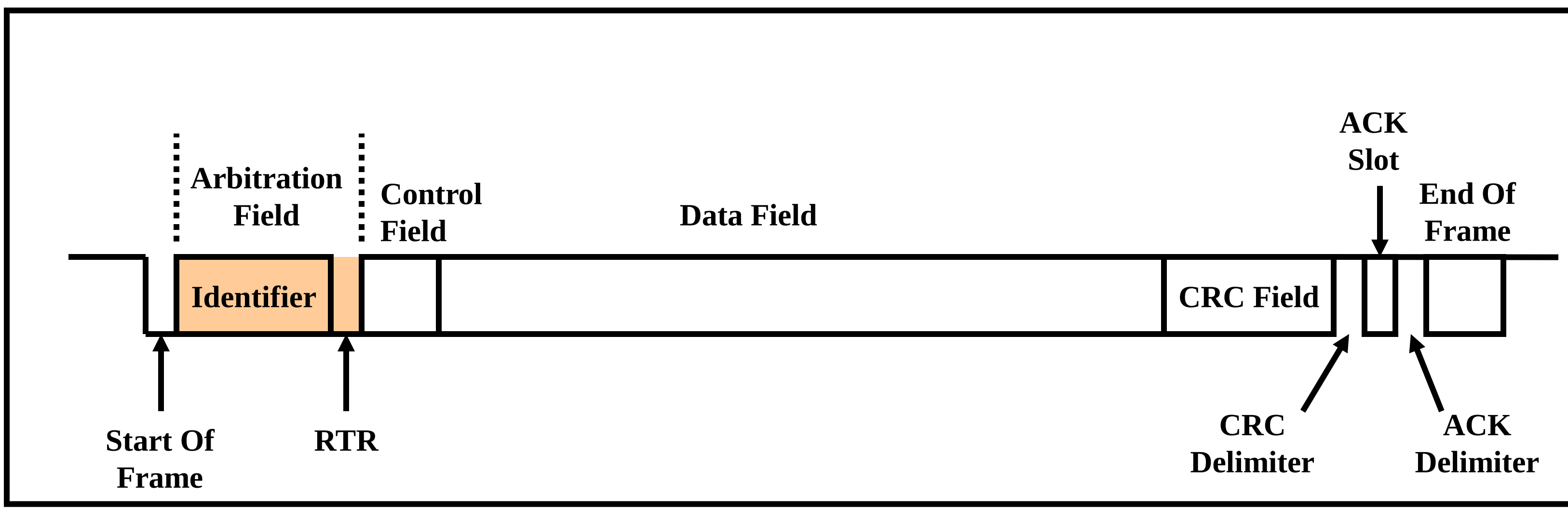
Data Frames are used to transmit messages to one or more receivers (multicast).

The CAN 2.0A Data Frame Format is shown below.



Arbitration Field

- Determines message priority.
  - For CAN 2.0A, an 11-bit Identifier and one bit, the RTR bit, which is dominant.
  - For CAN 2.0B, a 29-bit Identifier (which includes two recessive bits: SRR and IDE) and the RTR bit.

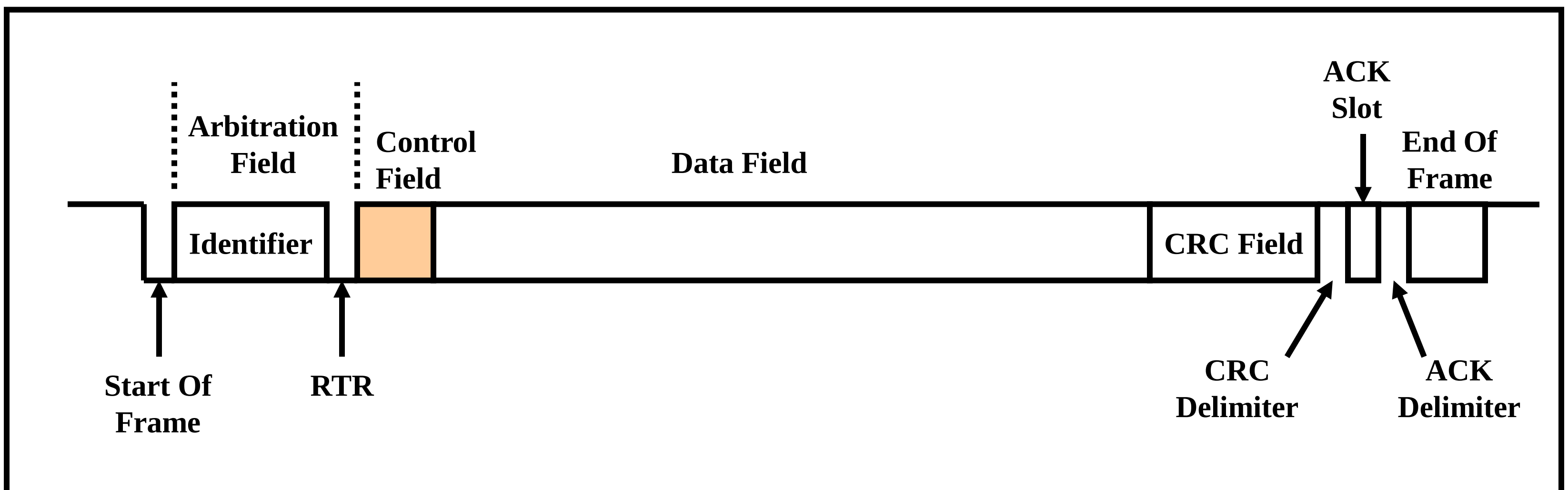


Number of Identifiers

- The number of unique identifiers available to users, on a single 2.0A network, is $(2^{11} - 2^{4}) =$ 2,032.
- The number of unique identifiers available on a single 2.0B network is in approximately 2^{29} (which is over 536 million!)

Control Field

- The Control Field contains six bits:
  - two dominant bits (r0 and r1) that are reserved for future use, and
  - a four bit Data Length Code (DLC). The DLC indicates the number of bytes in the Data Field.



Data Field

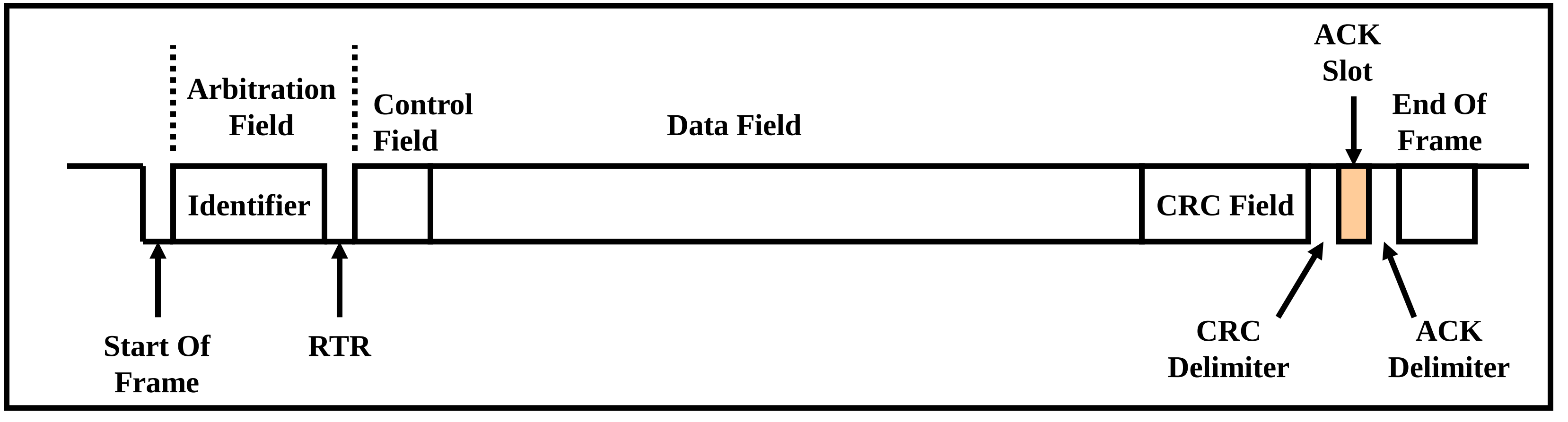
- The Data Field contains from zero to eight bytes of data.

Cyclic Redundancy Code (CRC)

- The CRC Field contains a 15-bit CRC code used for error detection and one recessive bit used as a delimiter.
- $G(x) = x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1$

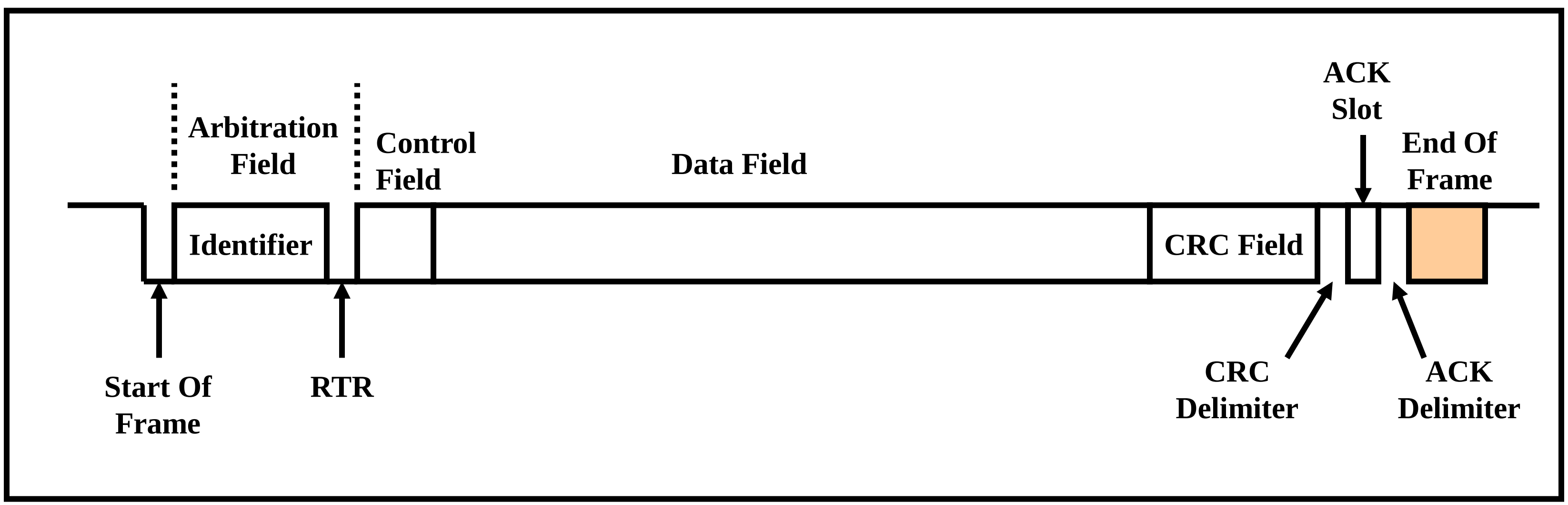
Acknowledgement Slot

- **Any** CAN controller that is able to correctly receive the message sends an Acknowledgement bit at the end of the message.
- The transmitter checks for the presence of the Ack bit and retransmits the message if no acknowledgement was detected.



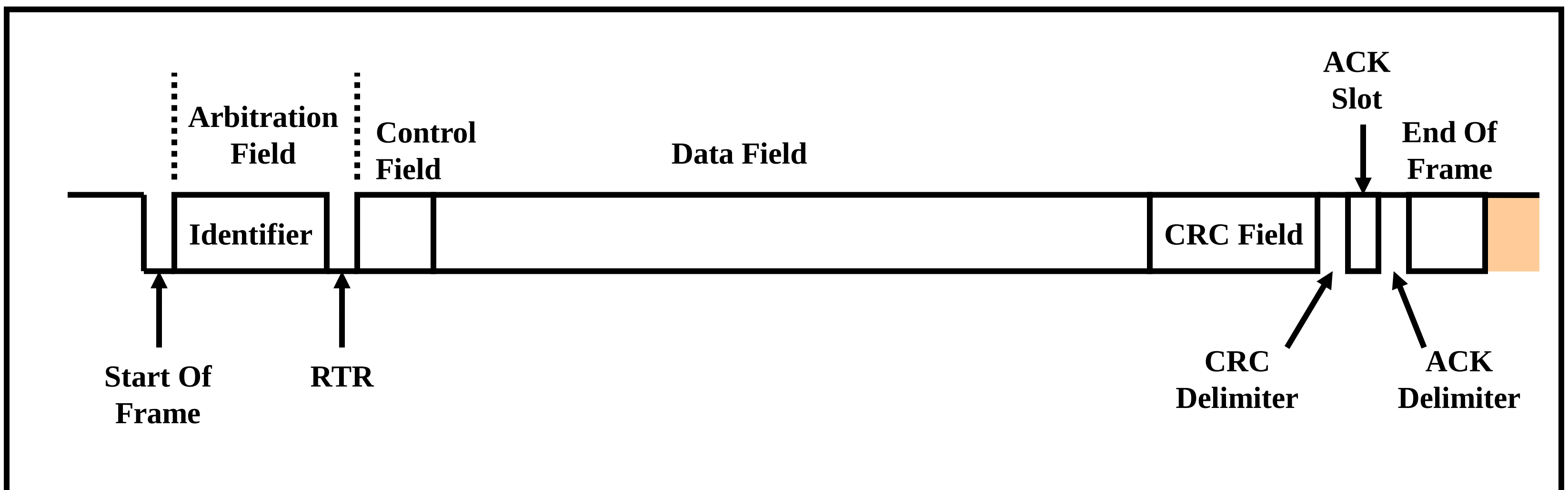
End Of Frame

- The End of Frame Field consists of seven recessive bits.



Intermission Field

- Following the End Of Frame are three recessive bits called the Intermission Field.
- After the Intermission interval, the bus is recognized to be idle (free).



Notes

- The presence of an Acknowledgement Bit on the bus does not mean that any of the **intended** addressees has received the message. The only thing we know is that **one or more** nodes on the bus has received it correctly.
- The Identifier in the Arbitration Field does not, despite its name, necessarily identify the contents of a message.
- Bus Idle time may be of any arbitrary length including zero.

Frame Types

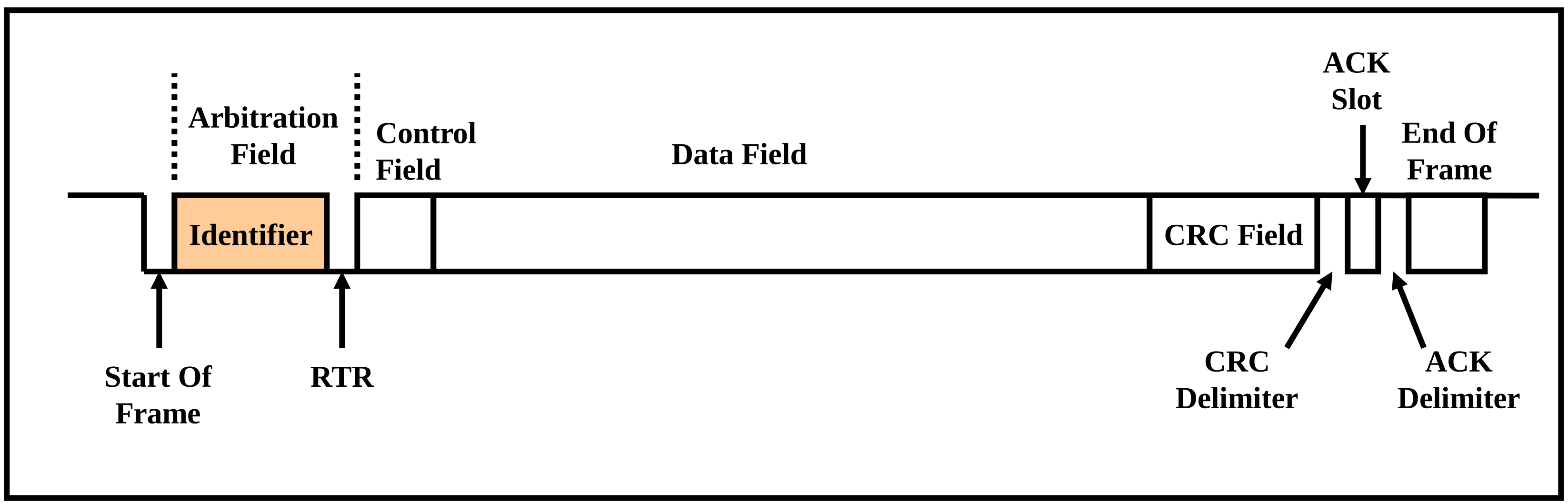
There are four different message types (or frame types) on a CAN bus:

- the Data Frame,
- the Remote Frame,
- the Error Frame, and
- the Overload Frame.

1. Data Frame

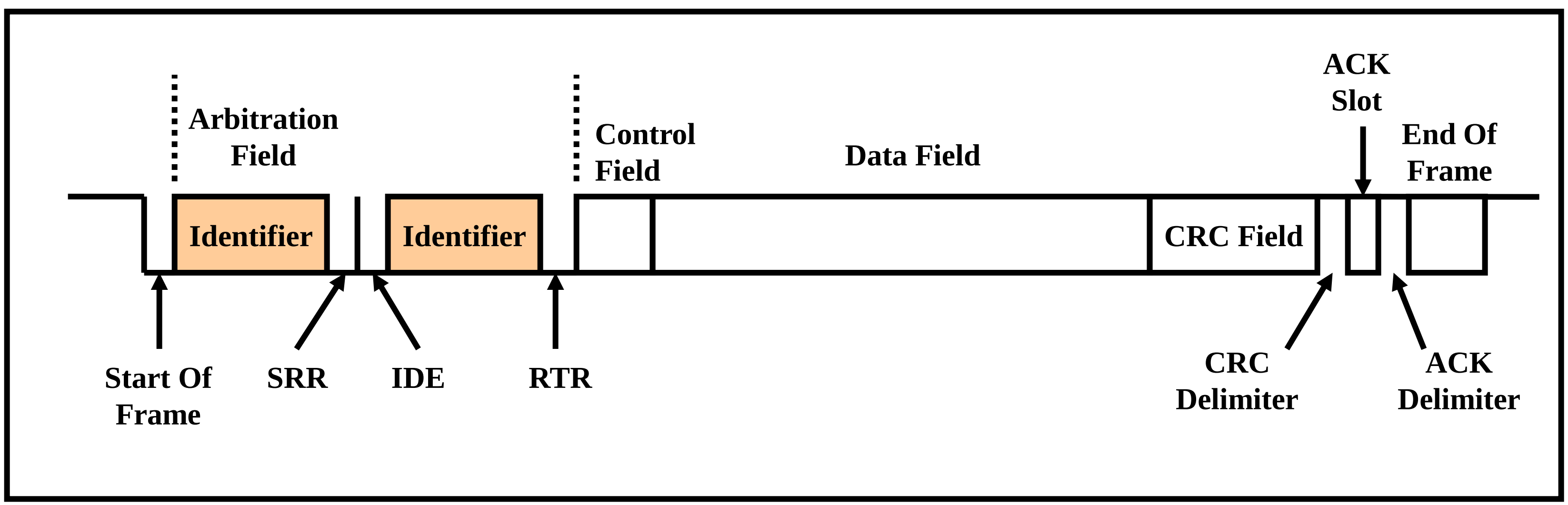
Data Frames are used to transmit messages to one or more receivers (multicast).

The CAN 2.0A Data Frame Format is shown below.



CAN 2.0B Format

- The CAN 2.0B Format uses a 29-bit identifier divided into an 11-bit base identifier and a 18-bit extended identifier.



Version 2.0A and 2.0B Compatibility

- Most 2.0B controllers are completely backward compatible with 2.0A controllers and can transmit, and receive messages in either format.
- However, there are two types of 2.0A controllers:
  - The first is capable of transmitting and receiving only messages in 2.0A format.
  - The second, called 2.0B passive, is able to send and receive 2.0A messages, and acknowledge receipt of 2.0B messages. However, 2.0B messages are then ignored (dropped).

SRR and IDE Bits

- The distinction between the two formats is made using the **Identifier Extension (IDE)** bit.
  - For an extended frame (29-bit ID), IDE = 1.
- A **Substitute Remote Request (SRR)** bit is also included in the Arbitration Field.
  - The SRR bit is always transmitted as a recessive bit to ensure that, in the case of arbitration between a Standard Data Frame and an Extended Data Frame, the Standard Data Frame will always have higher priority if both messages have the same base id (11-bit ID).
  - **Note:** The SRR bit appears in the same position as the RTR bit in a Standard Data Frame.

2. Remote Frame

- The **Remote Transmission Request** (RTR) bit is used to discriminate between a transmitted Data Frame and a request for data from a remote node; e.g., a Remote Frame.
- If the RTR bit is set (1), the frame is a Remote (Request) Frame.
- If the RTR bit is not set (0), the frame is the corresponding Data (Response) Frame.
- Several higher-level CAN protocols, such as SAE J1939 and NMEA 2000 do not use remote frames.

Notes

- In the unlikely event that a Data Frame and a Remote Frame with the same identifier are transmitted at the same time, the Data Frame wins arbitration due to the dominant RTR bit following the identifier.
- In this way, the node that transmitted the Remote Frame receives the requested data immediately.

3. Error Frame

- An Error Frame is transmitted in response to an error condition.
- An “active” Error Frame has higher priority than Data Frames.
- A “passive” Error Frame does not.

Types of Errors

- **Bit Error** - the sending node monitors the bus to ensure that the data transmitted is the same as the data received; exceptions include Ack bits and Arbitration field bits.
- **Stuff Error** - bit stuffing is used, if a 6th successive bit (either 0's or 1's) is detected, then a stuff error is reported.
- **CRC Error** - the CRC code computed is different than the code transmitted.

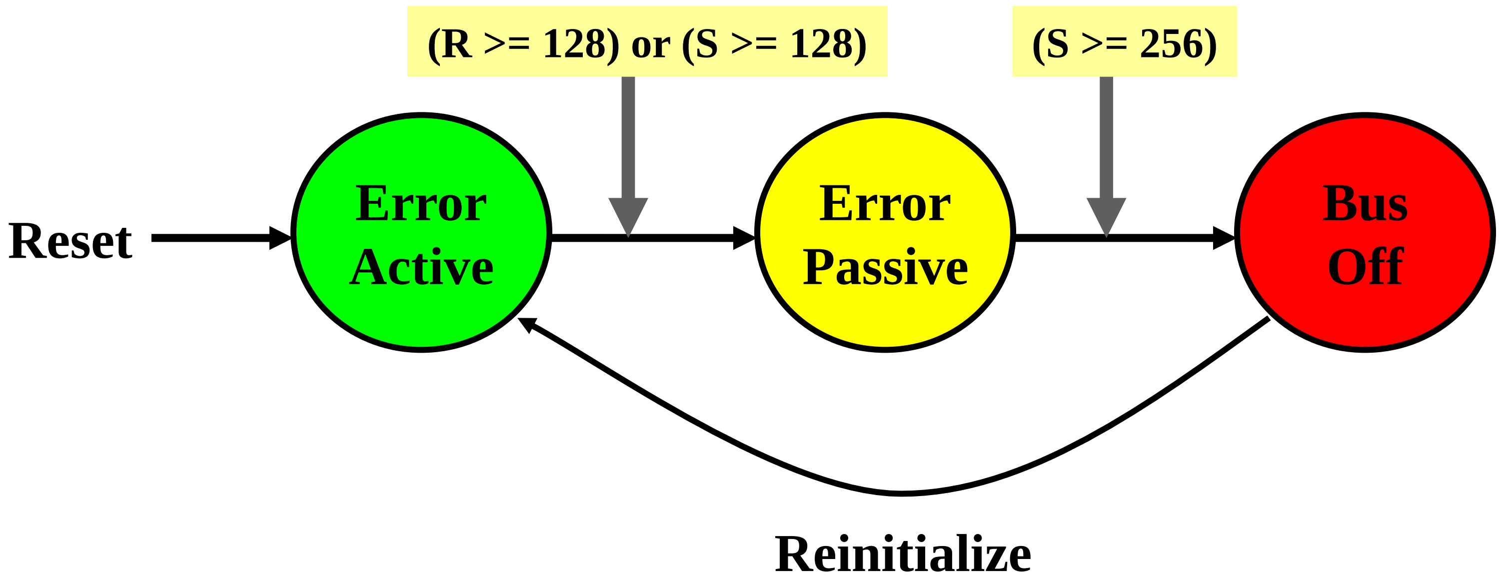
Types of Errors (continued)

- **Form Error** - a form error is detected when a fixed form bit (constant bit) is different than what is expected.
- **Acknowledgement Error** - an ack. error is detected whenever the sender does not monitor a dominant bit in the Ack Slot; e.g., no other node acknowledges receipt of the frame.

Fault Confinement

- A node may be in one of three states:
  - **Active Error State** - normal operation
  - **Passive Error State** - don’t generate active error messages
  - **Bus Off** - don’t generate any messages
- An error count > 96 indicates a “heavily disturbed” bus.

Error States



R = Receive Error Count,
S = Send (Transmit) Error Count

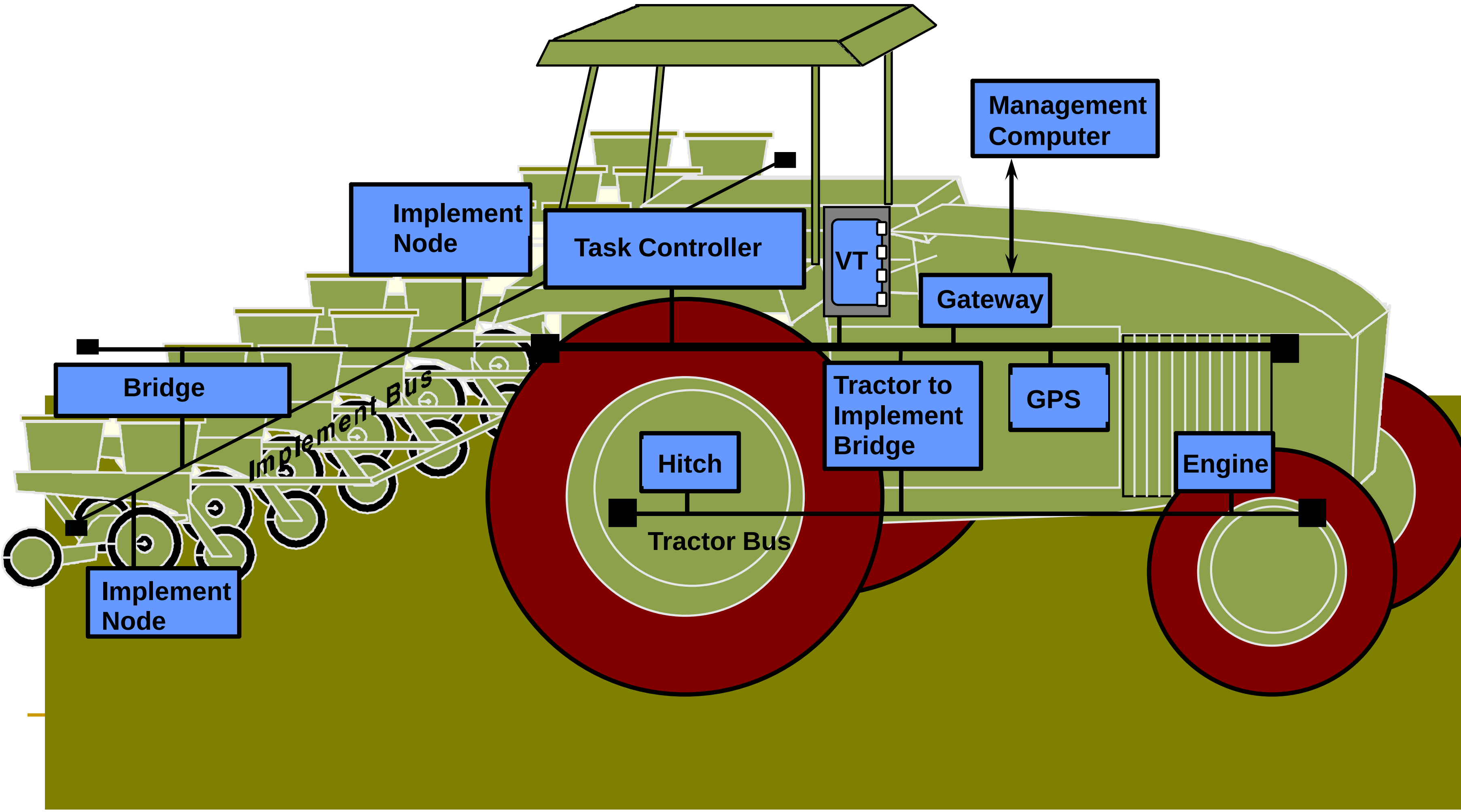
4. Overload Frame

- An Overload Frame is used by an overwhelmed receiver to indicate an overload by inserting some “dummy” bits onto the bus.
- Most higher level protocols do not use overload frames.

Higher Level Protocols

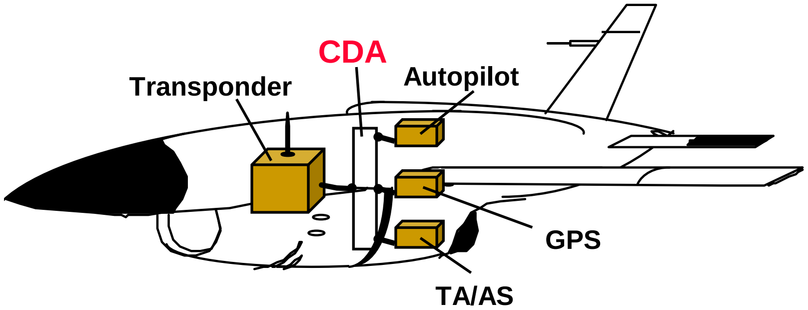
- SAE J1939 - for in-vehicle, heavy truck, and agricultural networking
- NMEA 2000 - for networking marine craft
- CANopen - CiA (CAN in Automation)
- DeviceNet - ODVA, (Allen-Bradley, etc.)
- SDS - Honeywell
- CDA 101 - DoD
- **CAN Kingdom** - Kvaser

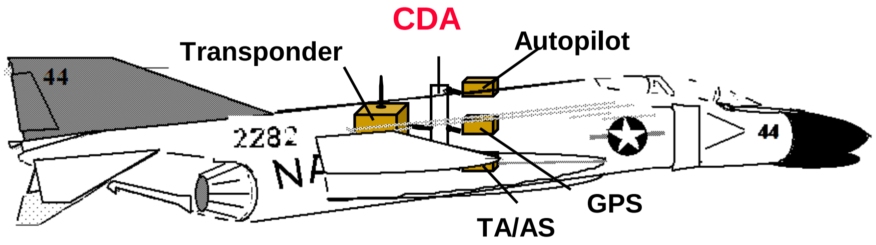
Controller Area Network (SAE J1939)



Common Digital Architecture (CDA 101)

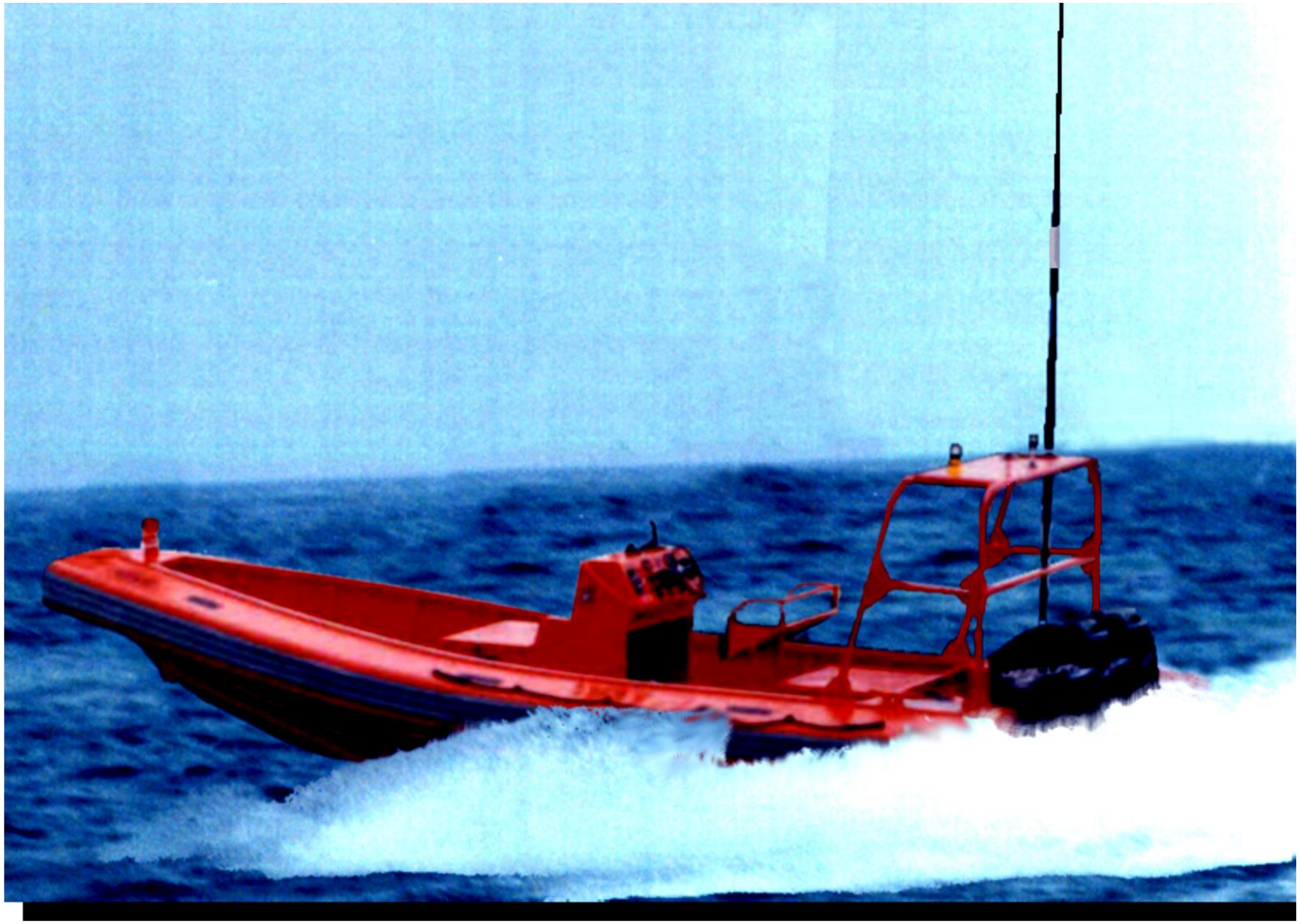
- A standard architecture for interconnecting target vehicle electronics



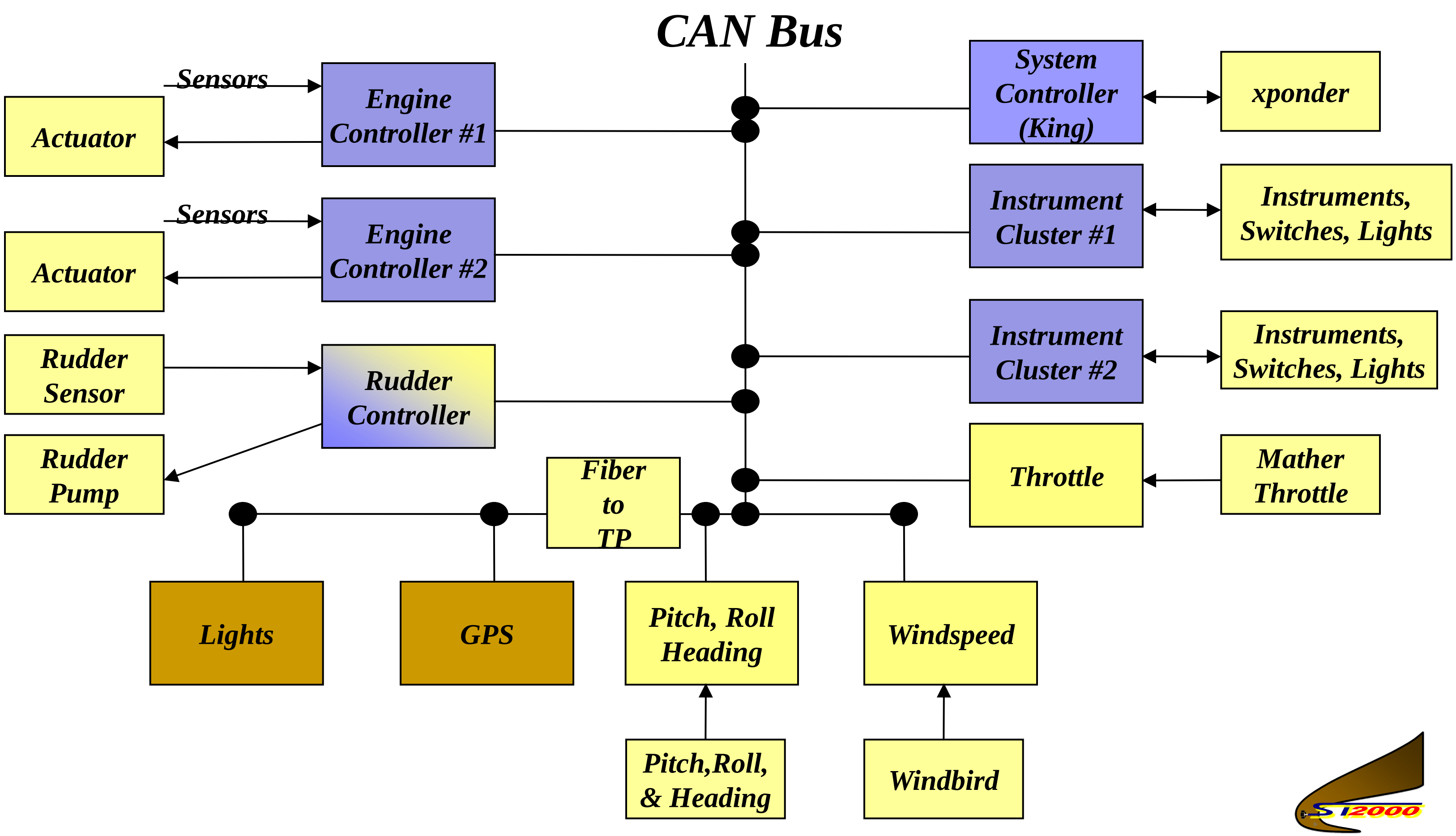


- Interface standards
  - Hardware
  - Software

Seaborne Target ST2000



Seaborne Target ST2000



Typical CAN Node

Optically Isolated RS-232

CAN Choke

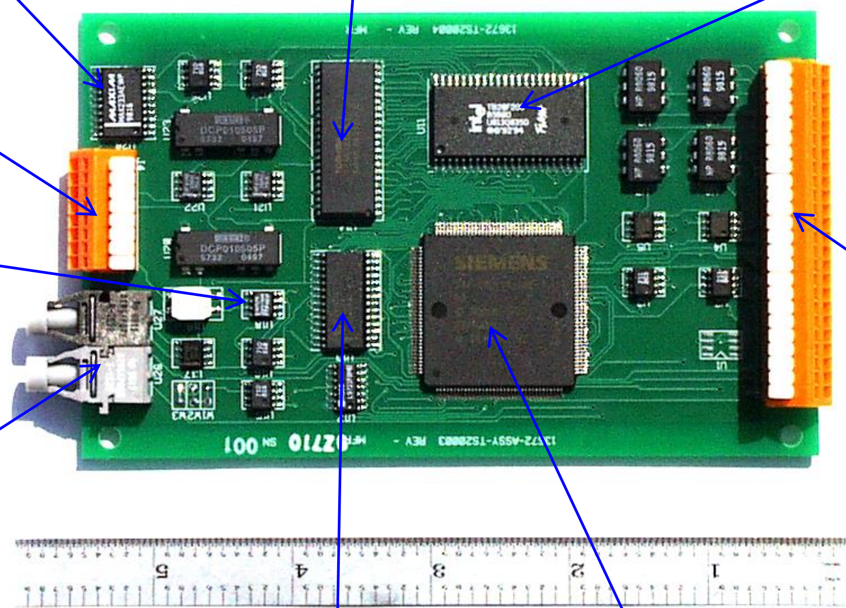
CAN Transceiver

Fiber Optic Transceivers

RAM

FLASH

Optically Isolated I/O



Extra CAN Controller (SJA1000)

uController

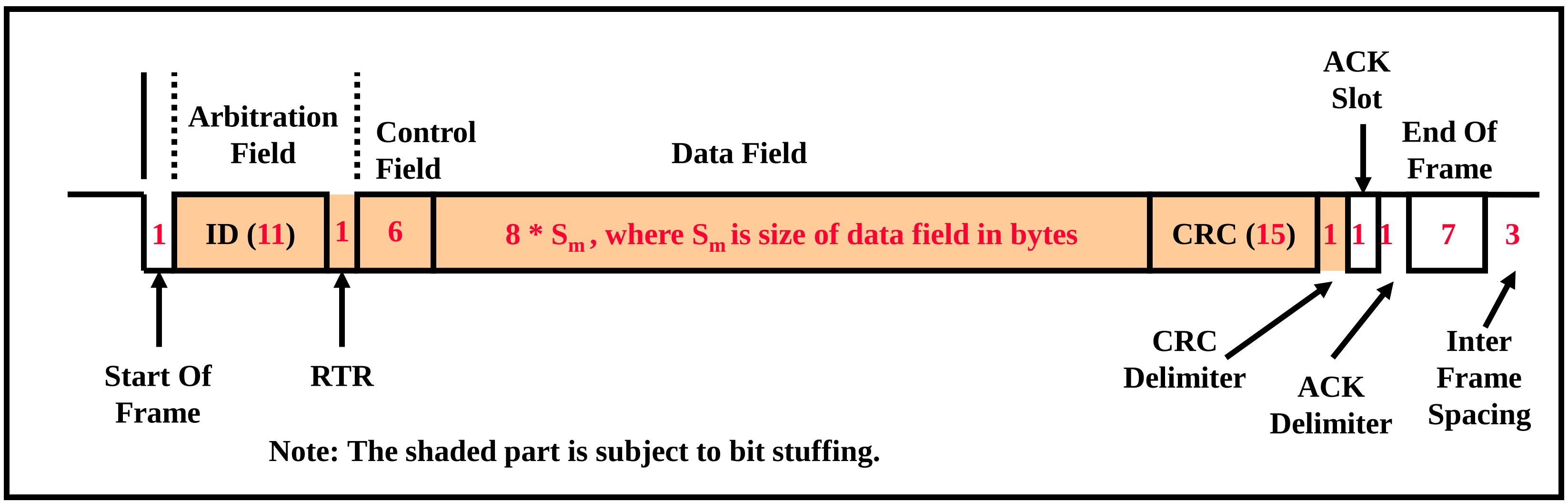
CAN Analysis - Assumptions

- Only **data frames** are transmitted on the bus, and no errors occur.
- All messages are transmitted periodically; however, messages may experience queuing delay, called **jitter**.

Data Frame Size

Data Frames are used to transmit messages to one or more receivers (multicast).

The CAN 2.0A Data Frame Format is shown below.

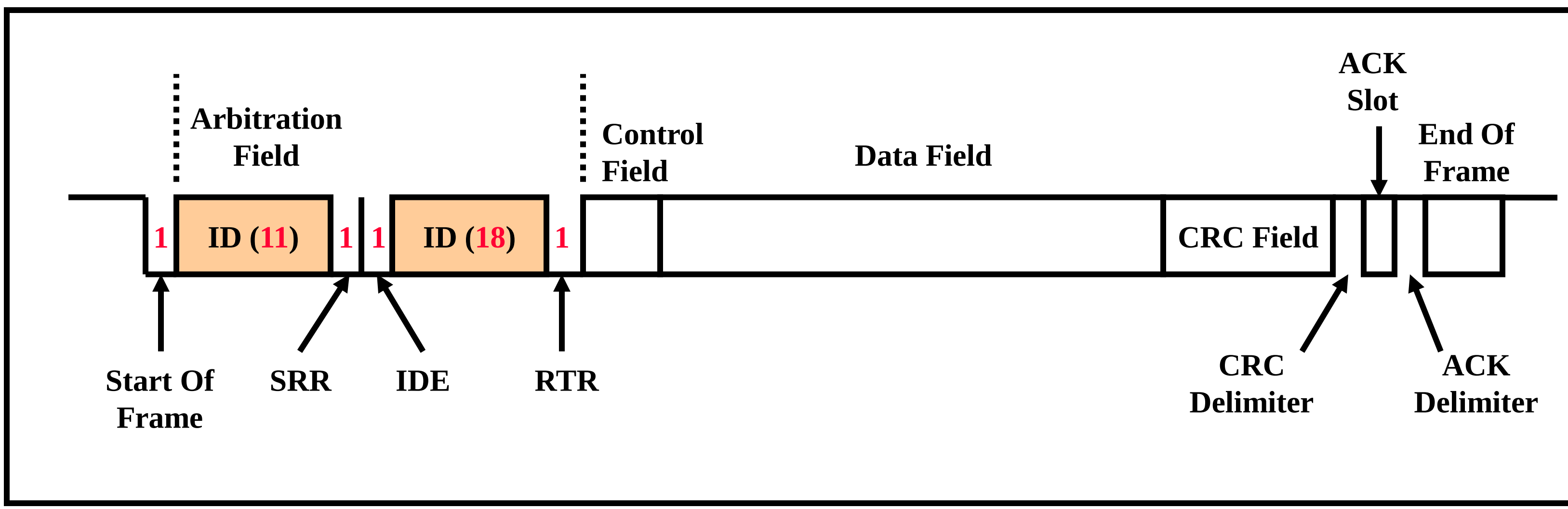


Worst-Case Tx Time (CAN 2.0A)

- Let C_m denote the worst-case transmission time for a data frame.
- For a data frame carrying 8 bytes of data:
  - $S_m = 8$
  - $C_m = 130 * \tau_{bit}$, where τ_{bit} is the bit transfer rate; e.g., for a 1Mbps CAN bus, τ_{bit} is 1 microsecond.

CAN 2.0B Format

- The CAN 2.0B Format uses a 29-bit identifier divided into an 11-bit base identifier and a 18-bit extended identifier.
- All other fields are the same as 2.0A.



Worst-Case Tx Time (CAN 2.0B)

- Let C_m denote the worst-case transmission time for a data frame.
- For a data frame carrying 8 bytes of data:
  - $S_m = 8$
  - $C_m = 154 * \tau_{bit}$, where τ_{bit} is the bit transfer rate; e.g., for a 250 kbps CAN bus, τ_{bit} is 4 microsecond.

CAN Analysis

- K. Tindell, A. Burns, and A. Wellings, “Calculating Controller Area Network (CAN) Message Response Times”, 1994.
- Use Classical Scheduling Theory to analyze controller area networks.
  - Response Time Analysis

Worst-Case Response Time

- The **worst-case response time** (R_i) for message i is given by $R_i = J_i + w_i + C_i$ where:
  - J_i is the **maximum jitter** of message i,
  - w_i is the **maximum queuing delay** of message i (due to both higher priority messages and a single lower priority message already on the bus); e.g., **interference + blocking time**, and
  - C_i is the **transmission time** of message i.

Response Time Analysis

- The **queuing delay** (w_i) for message i is given by the implicit equation:

$$w_i = B_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i + J_j + \tau_{\mathrm{bit}}}{T_j} \right\rceil C_j$$

where B_i denotes **worst-case blocking time**:

$$B_i = \max_{j \in lp(i)} (C_j)$$

Summary

- **Next:**
  - CAN Higher-Layer Protocols
    - CAN Kingdom
  - Time-Triggered Protocols