Control Area Network (CAN) Bus

Prof. Sergio Saponara

Sistemi Elettronici per Automazione e Robotica 2024/2025

Overview

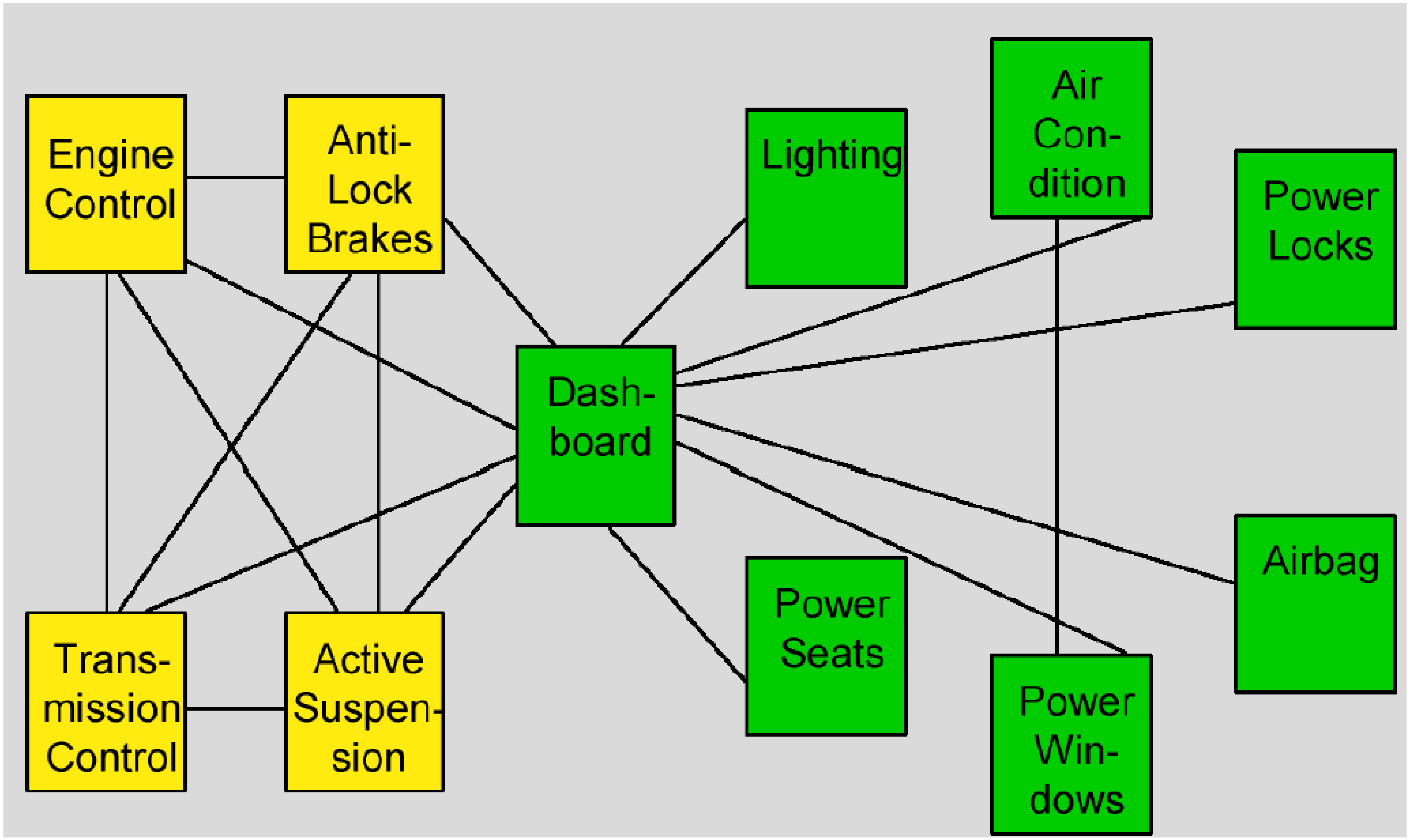
- CAN is an important embedded protocol
- Primarily automotive, but used in many other places
- CAN specifies:
  - Physical layer
  - Protocol layer
  - Message filtering layer (with add-on protocols)
- Note
  - How message prioritization achieved
  - How “small” nodes can be kept from overloading with received messages

The Development of CAN

The development of CAN began when more and more electronic devices were implemented into modern motor vehicles. Examples of such devices include engine management systems, active suspension, ABS, gear control, lighting control, air conditioning, airbags and central locking. All this means more safety and more comfort for the driver and of course a reduction of fuel consumption and exhaust emissions.

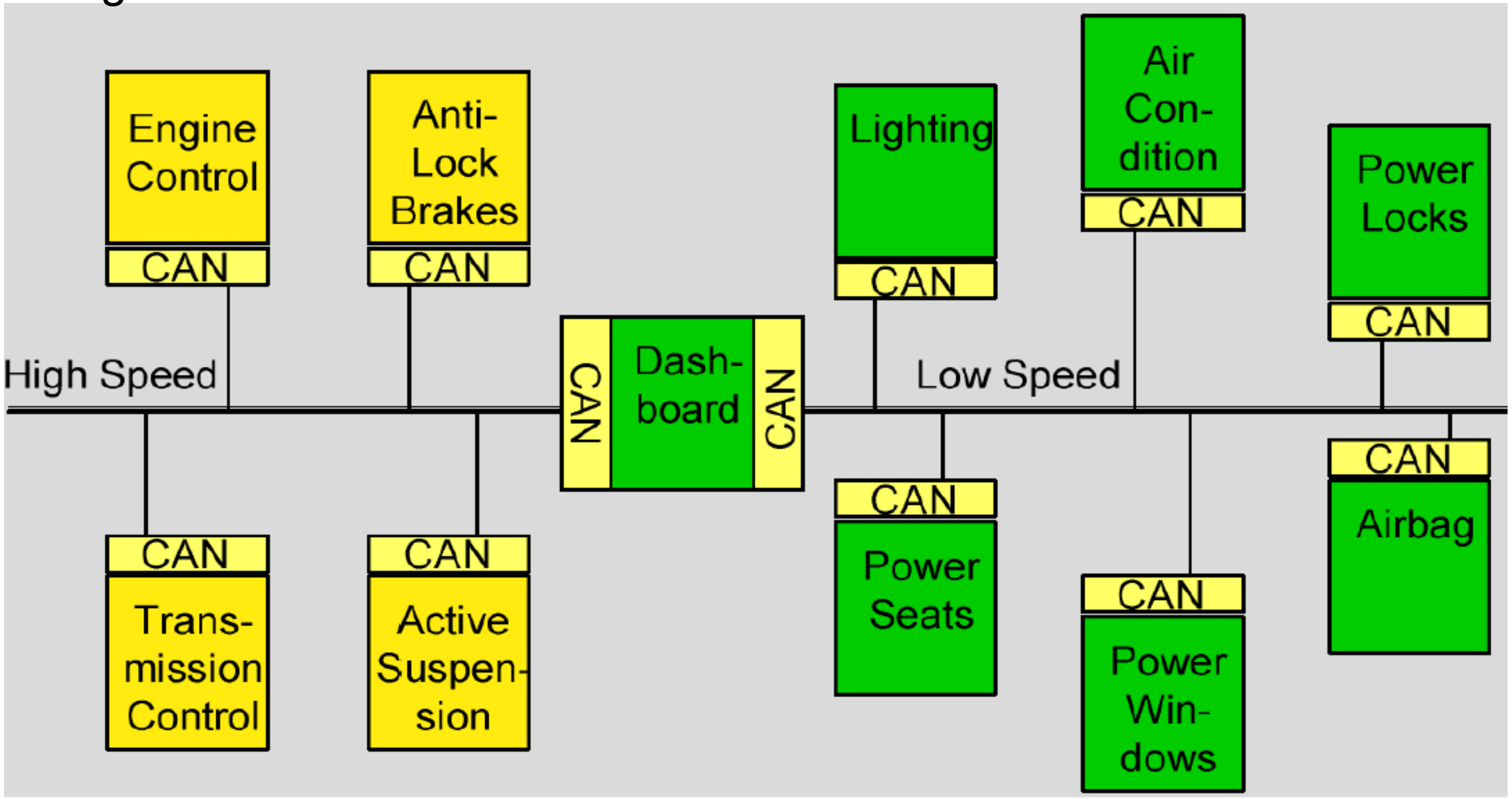
To improve the behavior of the vehicle even further, it was necessary for the different control systems (and their sensors) to exchange information. This was usually done by discrete interconnection of the different systems (i.e. point to point wiring). The requirement for information exchange has then grown to such an extent that a cable network with a length of up to several miles and many connectors was required. This produced growing problems concerning material cost, production time and reliability.

Before CAN

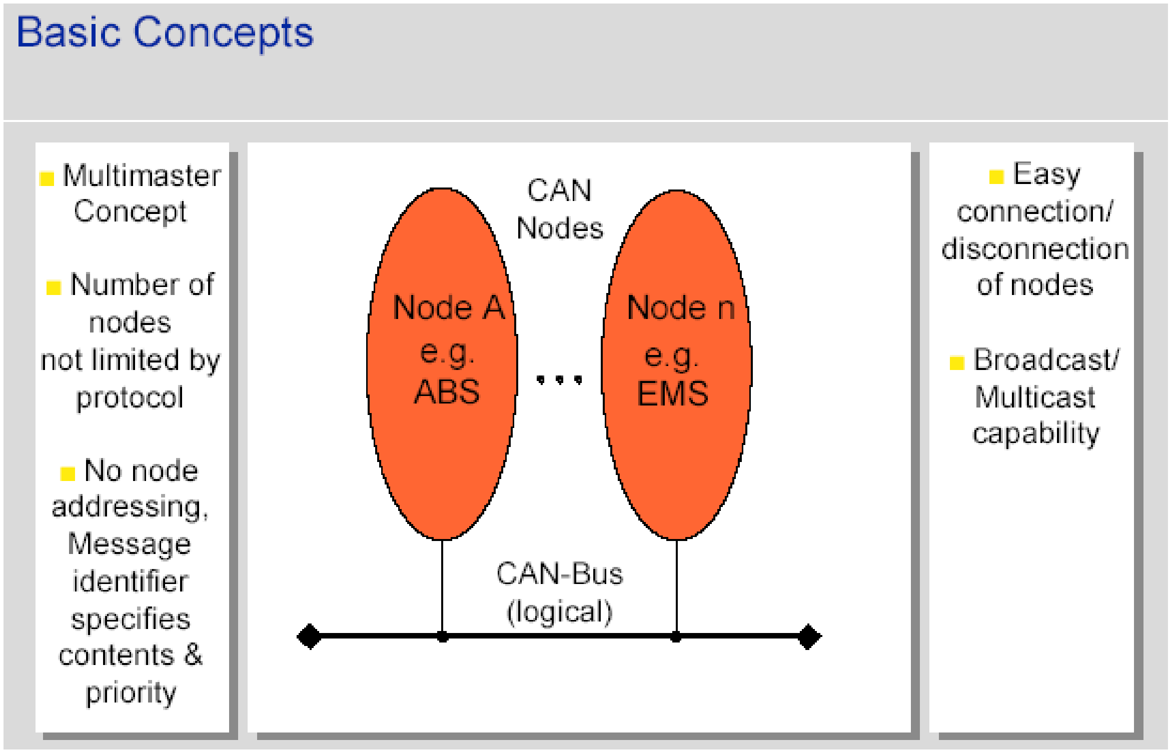

Engine Control
Anti-Lock Brakes
Lighting
Air Con-dition
Power Locks
Dash-board
Airbag
Trans-mission Control
Active Suspen-sion
Power Seats
Power Win-dows

With CAN

The solution to this problem was the connection of the control systems via a serial bus system. This bus had to fulfill some special requirements due to its usage in a vehicle. With the use of CAN, point-to-point wiring is replaced by one serial bus connecting all control systems. This is accomplished by adding some CAN-specific hardware to each control unit that provides the "rules" or the protocol for transmitting and receiving information via the bus.

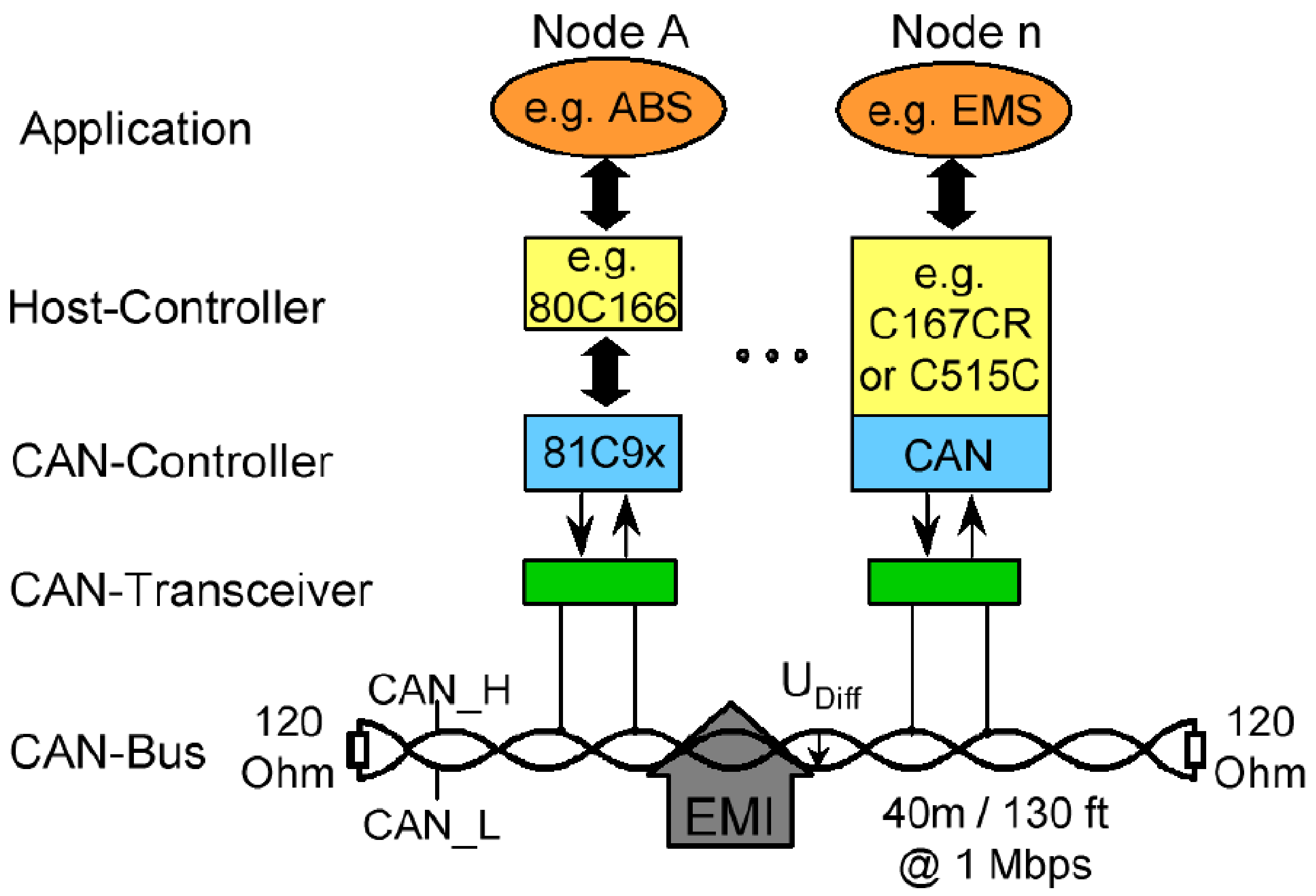


The CAN bus


Basic Concepts
Multimaster Concept
Number of nodes not limited by protocol
No node addressing, Message identifier specifies contents & priority
CAN Nodes
Node A e.g. ABS
...
Node n e.g. EMS
CAN-Bus (logical)
Easy connection/ disconnection of nodes
Broadcast/ Multicast capability

- CAN is a broadcast type of bus.
  - This means that all nodes can "hear" all transmissions. There is no way to send a message to just a specific node; all nodes will invariably pick up all traffic. The CAN hardware, however, provides local filtering so that each node may react only on the “interesting” messages.

Basic Configuration



CAN Bus Overview

- The physical layer uses differential transmission on a twisted pair wire. The bus uses Non-Return To Zero (NRZ) with bit-stuffing.
- The nodes are connected to the bus in a *wired-and* fashion: if just one node is driving the bus to a logical 0, then the whole bus is in that state regardless of the number of nodes transmitting a logical 1.
- Max. transfer rate of 1000 kilobits per second at a maximum bus length of 40 meters or 130 feet when using a twisted wire pair which is the most common bus medium used for CAN.
- Message length is short with a maximum of 8 data bytes per message and there is a low latency between transmission request and start of transmission. The messages are protected by a CRC type checksum

CAN Bus Overview

- The bus access is handled via the advanced serial communications protocol Carrier Sense Multiple Access/Collision Detection with Non-Destructive Arbitration. This means that collision of messages is avoided by bitwise arbitration without loss of time.
- There is no explicit address in the messages, instead, each message carries a numeric value which controls its priority on the bus, and may also serve as an identification of the *contents* of the message.
- An elaborate error handling scheme that results in retransmitted messages when they are not properly received.
- There are effective means for isolating faults and removing faulty nodes from the bus.

Basic Bit Encoding

NRZ = Non-Return-To_Zero

- Fewer transitions (on average) = less EMI, but requires less oscillator drift

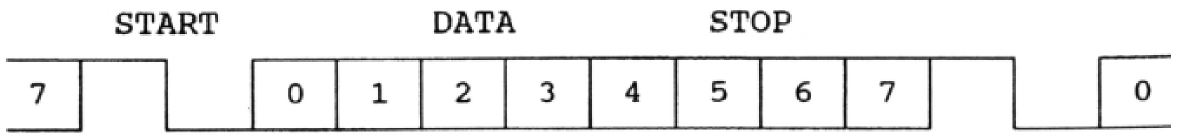
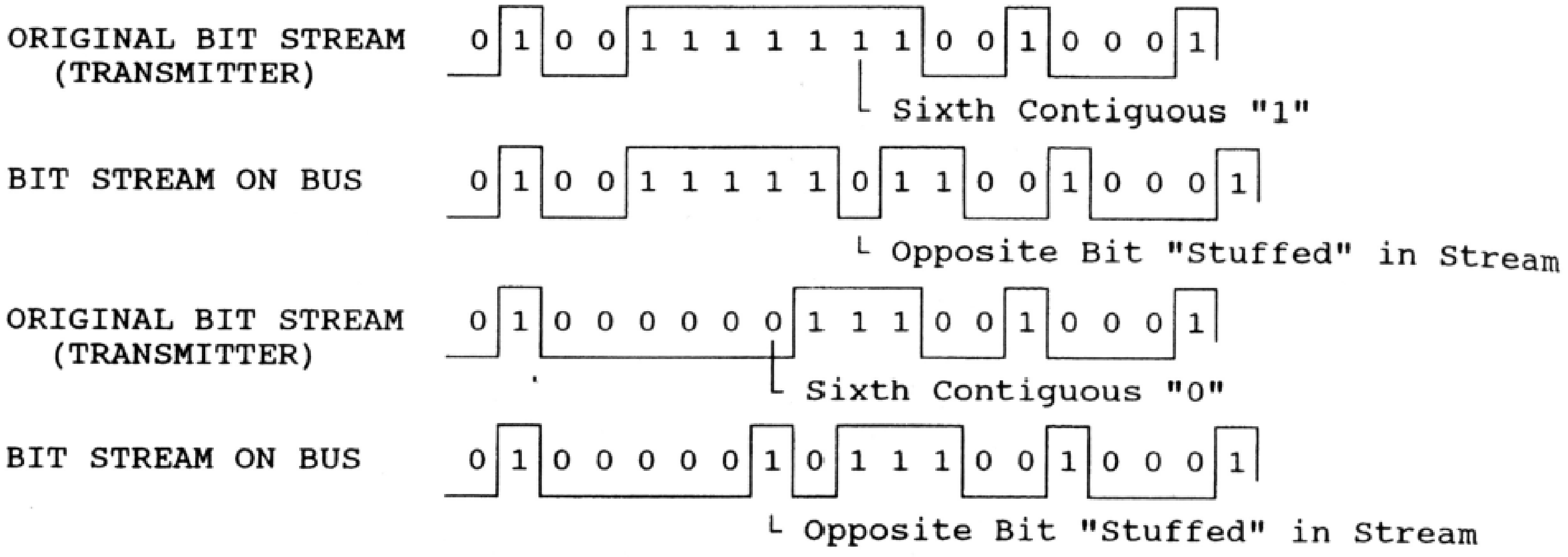


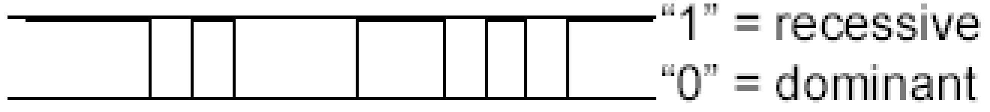
FIGURE 26.21 A 10-bit NRZ waveform (LSB first).

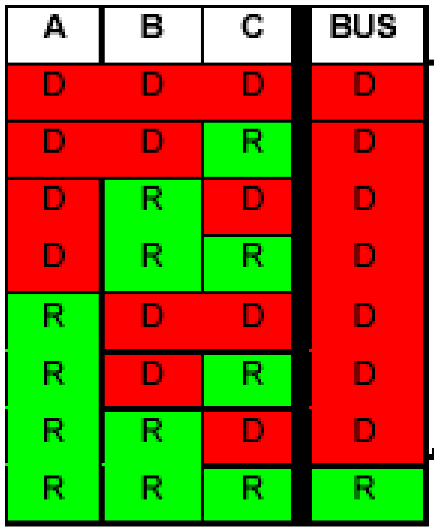
- Bit stuffing relaxes oscillator drift requirements



CAN Bus Characterstics

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



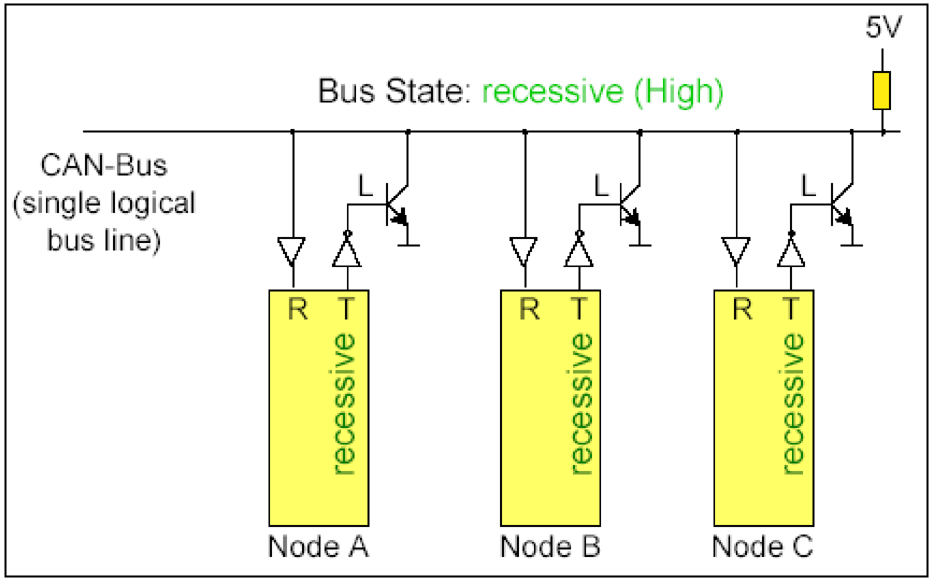


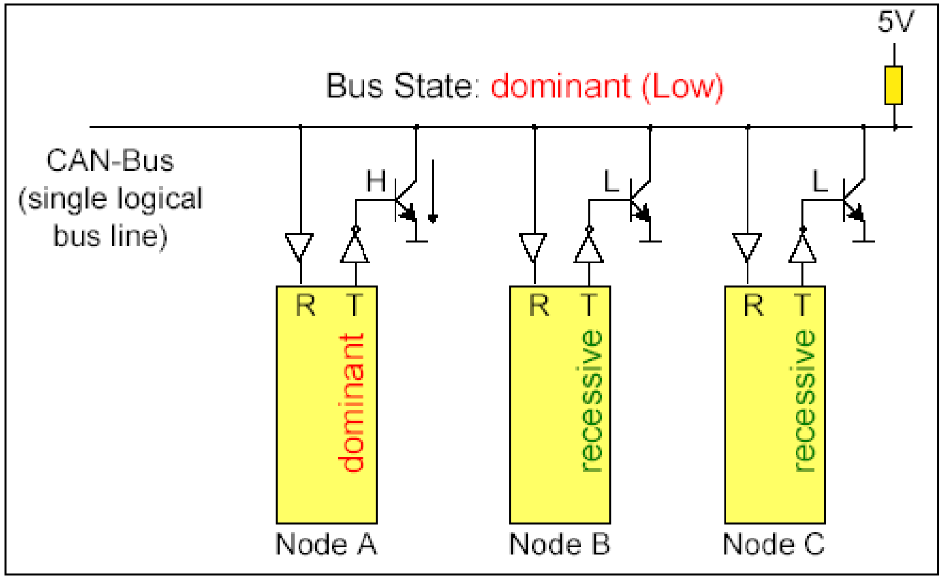
A	B	C	BUS
D	D	D	D
D	D	R	D
D	R	D	D
D	R	R	D
R	D	D	D
R	D	R	D
R	R	D	D
R	R	R	R

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.

Bus Characteristics – Wired AND


5V
Bus State: recessive (High)
CAN-Bus (single logical bus line)
L
L
L
R
T
recessive
R
T
recessive
R
T
recessive
Node A
Node B
Node C



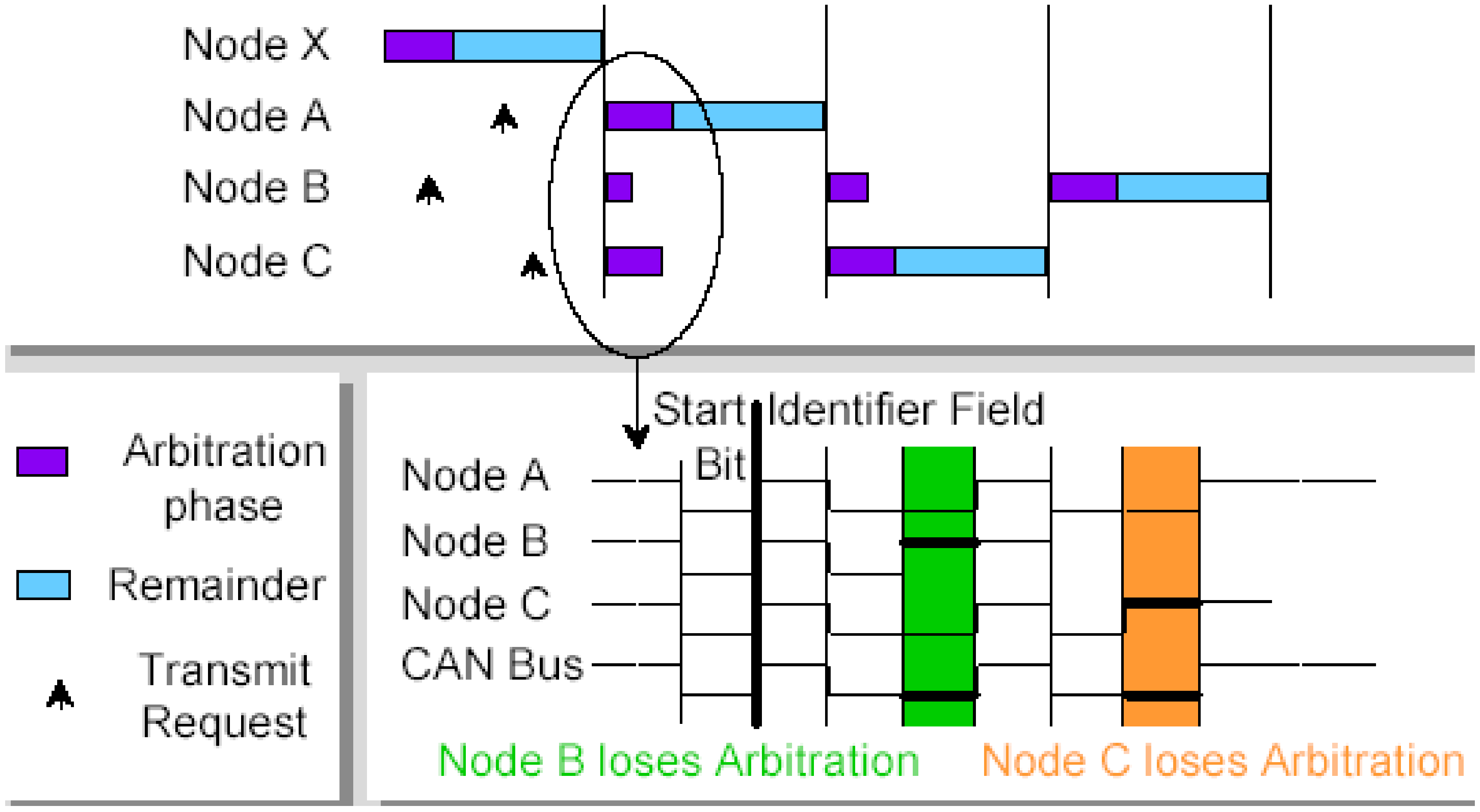
Only if all nodes transmit recessive bits (ones), the Bus is in the recessive state.

If any one node transmits a dominant bit (zero), the bus is in the dominant state.

T is Transmitter, R is receiver. Note nodes can therefore check the line while transmitting. This is important particularly during arbitration.

Bus Access and Arbitration – CSMA/CD NDA

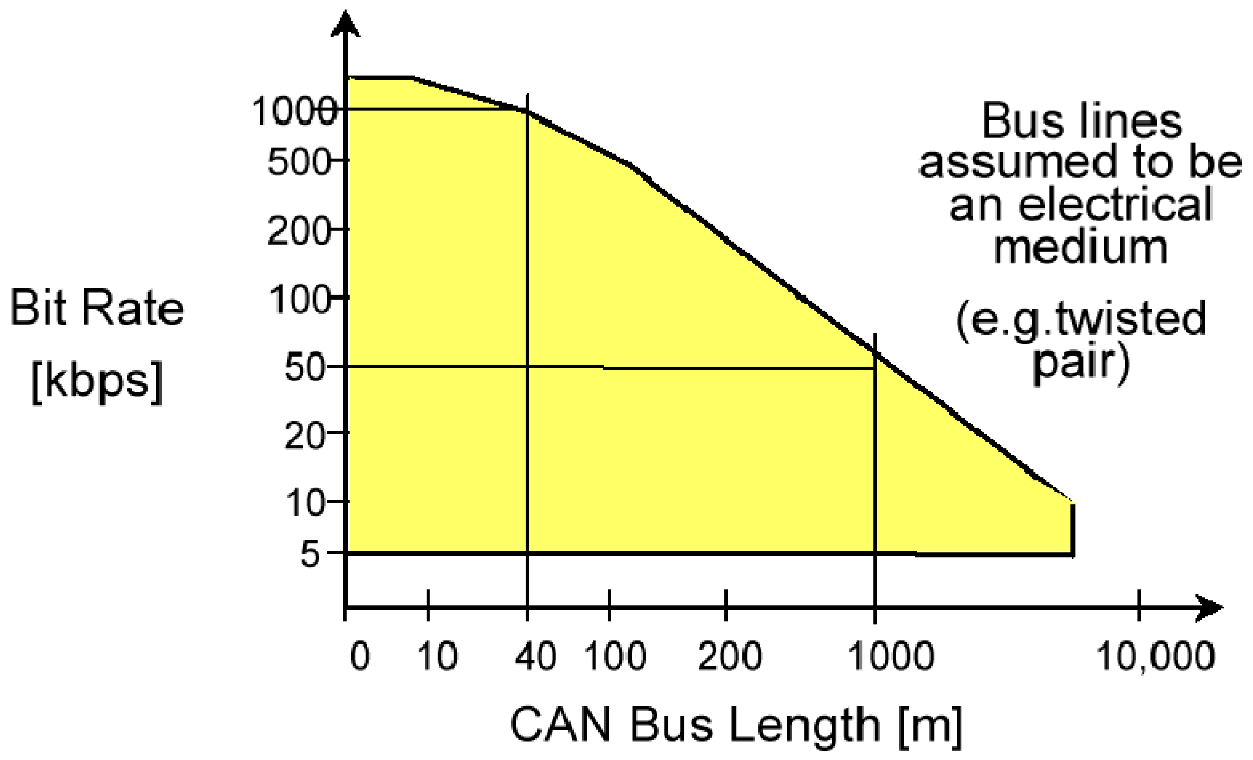
CSMA/CD NDA – Carrier Sense Multiple Access/Collision avoidance by Non Destructive arbitration



Bus Transmission Speed

Arbitration limits bus speed. Maximum speed = 2 x t_{pd}

tpd = propagation delay of electrical medium



The Can Protocol

- Specifies how small packets of data may be transported from point A to point B using a shared communications medium.
- It (quite naturally) contains nothing on topics such as
  - flow control
  - transportation of data larger than can fit in a 8-byte message
  - node addresses
  - establishment of communication, etc.

Higher layer protocols

- Higher layer protocols are used in order to
  - standardize startup procedures including bit rate setting
  - distribute addresses among participating nodes or kinds of messages
  - determine the layout of the messages
  - provide routines for error handling at the system level
- Some high layer protocols
  - Device net
  - CANKingdom
  - CANopen

The CAN Standard

- The CAN standard defines four message types
  - Data Frame – the predominantly used message type
  - Remote Frame
  - Error Frame
  - Overload Frame
- The messages uses a clever scheme of bit-wise arbitration to control access to the bus, and each message is tagged with a priority.
- The CAN standard also defines an elaborate scheme for error handling and confinement.
- CAN may implemented using different physical layers, and there are also a number of different connector types in use.

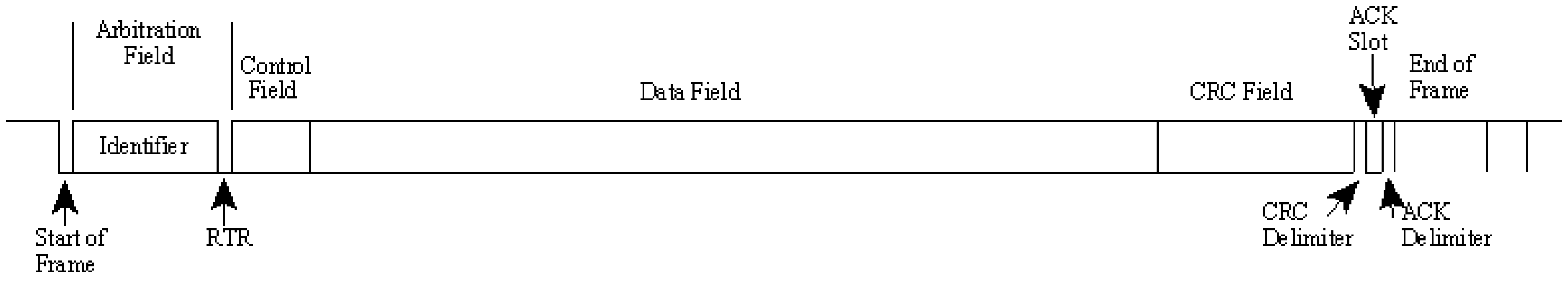
1. The Data Frame

- **Summary: "Hello everyone, here's some data labeled X, hope you like it!"**
- The Data Frame is the most common message type. It comprises the following major parts (a few details are omitted for the sake of brevity):
- the Arbitration Field, which determines the priority of the message when two or more nodes are contending for the bus. The Arbitration Field contains:
  - For CAN 2.0A, an 11-bit Identifier and one bit, the RTR bit, which is dominant for data frames.
  - For CAN 2.0B, a 29-bit Identifier (which also contains two recessive bits: SRR and IDE) and the RTR bit.
- the Data Field, which contains zero to eight bytes of data.
- the CRC Field, which contains a 15-bit checksum calculated on most parts of the message. This checksum is used for error detection.
- an Acknowledgement Slot; ***any*** CAN controller that has been able to correctly receive the message sends an Acknowledgement bit at the end of each message. The transmitter checks for the presence of the Acknowledge bit and retransmits the message if no acknowledge was detected.

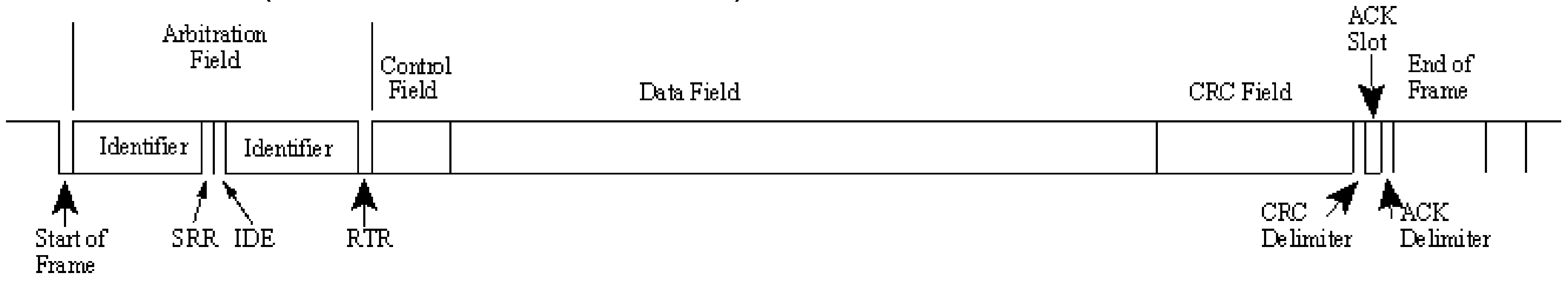
CAN Data Frames

Note 1: It is worth noting that 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
Note 2: The Identifier in the Arbitration Field is not, despite of its name, necessarily identifying the contents of the message.

- CAN 2.0A ("standard CAN" 11-bit ID) Data Frame.



- CAN 2.0B ("extended CAN" 29-bit ID) Data Frame.



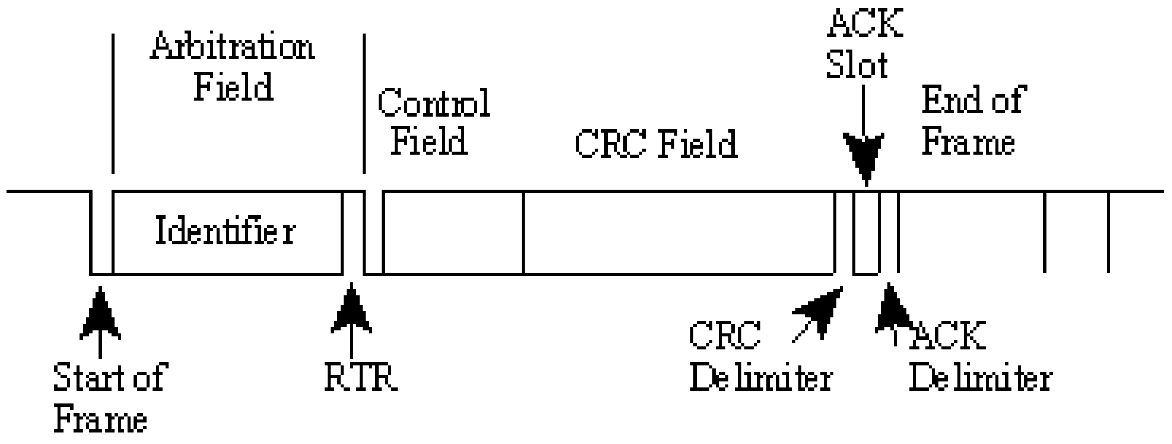
2. The Remote Frame

- ***Summary: "Hello everyone, can somebody please produce the data labeled X?"***

- The Remote Frame is just like the Data Frame, with two important differences:
  - It is explicitly marked as a Remote Frame (the RTR bit in the Arbitration Field is recessive), and
  - there is no Data Field.
- The intended purpose of the Remote Frame is to solicit the transmission of the corresponding Data Frame. If, say, node A transmits a Remote Frame with the Arbitration Field set to 234, then node B, if properly initialized, might respond with a Data Frame with the Arbitration Field also set to 234.
- Remote Frames can be used to implement a type of request-response type of bus traffic management. In practice, however, the Remote Frame is little used. It is also worth noting that the CAN standard does not *prescribe* the behaviour outlined here. Most CAN controllers can be programmed either to automatically respond to a Remote Frame, or to notify the local CPU instead.

Remote Frame (contd.)

- There's one catch with the Remote Frame: the Data Length Code *must be set to the length of the expected response message.* Otherwise the arbitration will not work.
- Sometimes it is claimed that the node responding to the Remote Frame is starting its transmission as soon as the identifier is recognized, thereby "filling up" the empty Remote Frame. ***This is not the case*.**
- A Remote Frame (2.0A type):



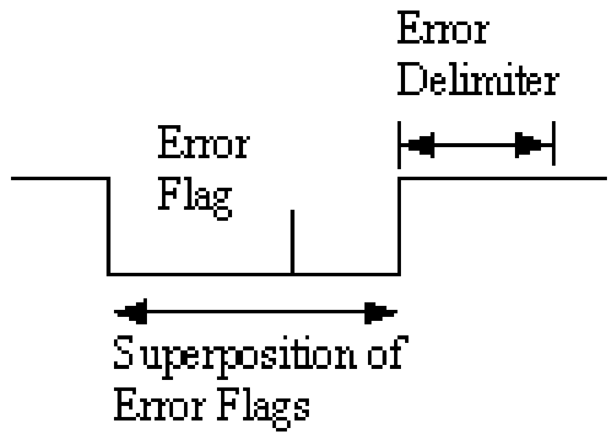
3. The Error Frame

Summary: (everyone, aloud) "OH DEAR, LET'S TRY AGAIN"

Simply put, the Error Frame is a special message that violates the framing rules of a CAN message. It is transmitted when a node detects a fault and will cause all other nodes to detect a fault - so they will send Error Frames, too. The transmitter will then automatically try to retransmit the message. There is an elaborate scheme of error counters that ensures that a node can't destroy the bus traffic by repeatedly transmitting Error Frames.

The Error Frame consists of an Error Flag, which is 6 bits of the same value (thus violating the bit-stuffing rule) and an Error Delimiter, which is 8 recessive bits. The Error Delimiter provides some space in which the other nodes on the bus can send their Error Flags when they detect the first Error Flag.

The Error Frame



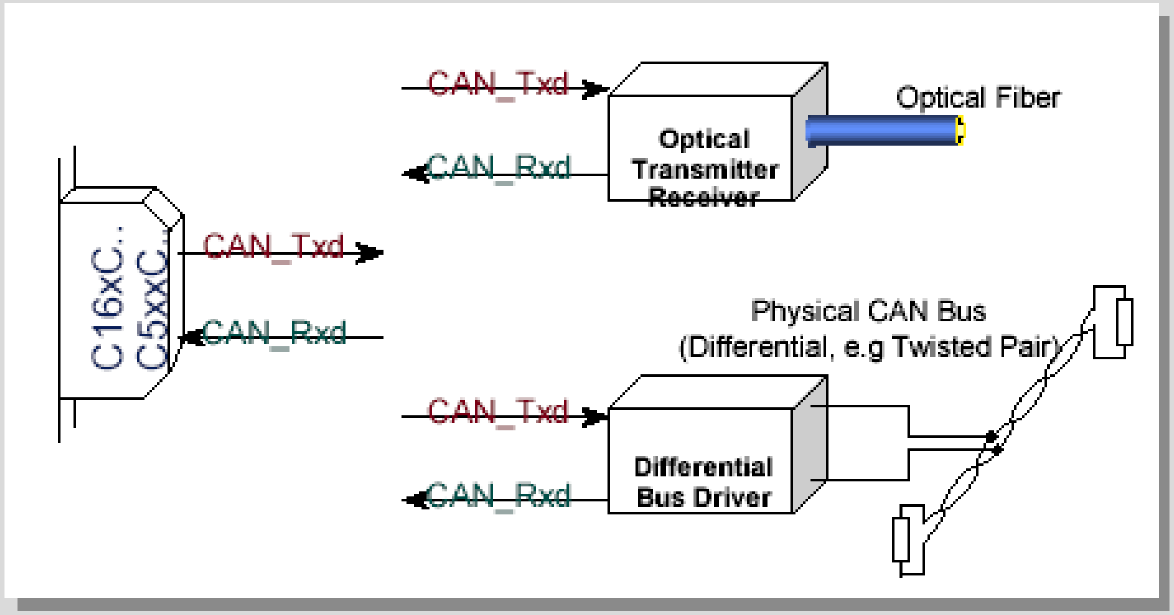
4 The Overload Frame

Summary: "I'm a very busy little 82526 device, could you please wait for a moment?"

- The Overload Frame is mentioned here just for completeness. It is very similar to the Error Frame with regard to the format and it is transmitted by a node that becomes too busy. The Overload Frame is not used very often, as today's CAN controllers are clever enough not to use it. In fact, the only controller that will generate Overload Frames is the now obsolete 82526

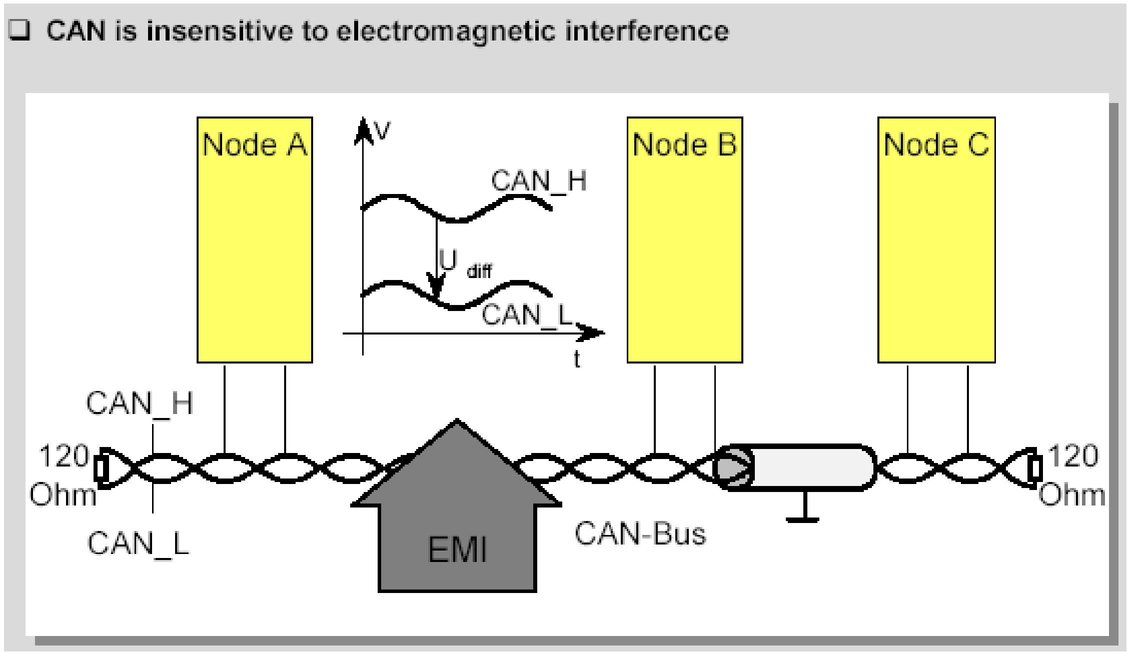
ISO Physical Layer

- Usual ISO Physical Layer :-
  - Bus wires twisted pair, 120R Termination at each end
  - 2 wires driven with differential signal (CAN_H, CAN_L)



One of the most common and cheapest implementations is to use a twisted wire pair. The bus lines are then called "CAN_H" and "CAN_L". The two bus lines CAN_H and CAN_L are driven by the nodes with a differential signal. The twisted wire pair is terminated by terminating resistors at each end of bus line, typically 120 ohms.

CAN and EMI



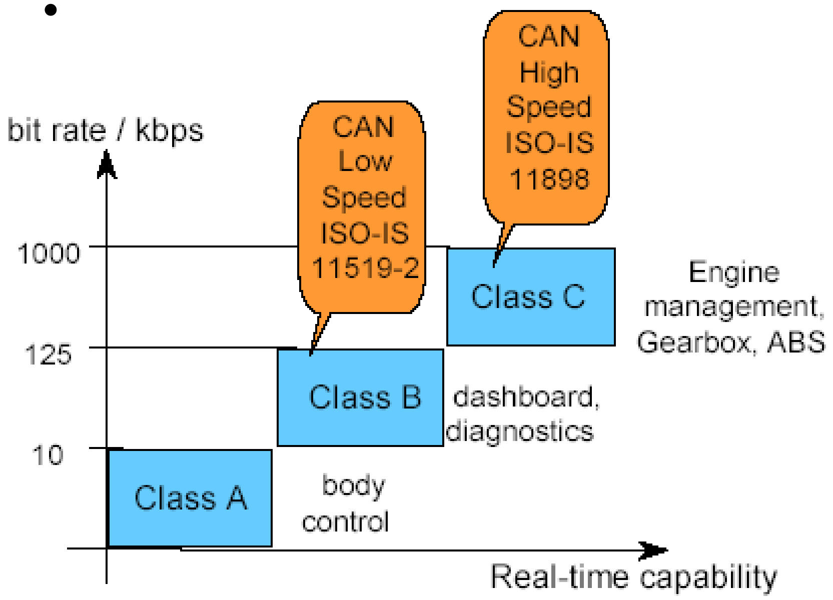
Due to the differential nature of transmission CAN is insensitive to electromagnetic interference, because both bus lines are affected in the same way which leaves the differential signal unaffected.
To reduce the sensitivity against electromagnetic interference even more, the bus lines can additionally be shielded. This also reduces the electromagnetic emission of the bus itself, especially at high baudrates.

Standardisation

•Vehicle bus system applications can be separated in three different categories according to their real-time capabilities.

- Class A for a low speed bus with bit rates up to 10 kbps, e.g for body control applications,
- Class B for a low speed bus with bit rates from 10 kbps to 125 kbps, e.g. for dashboard and diagnostics,
- Class C for a high speed bus with bit rates from 125 kbps to 1 Mbps for real time applications like engine management, Gearbox, ABS etc.

•

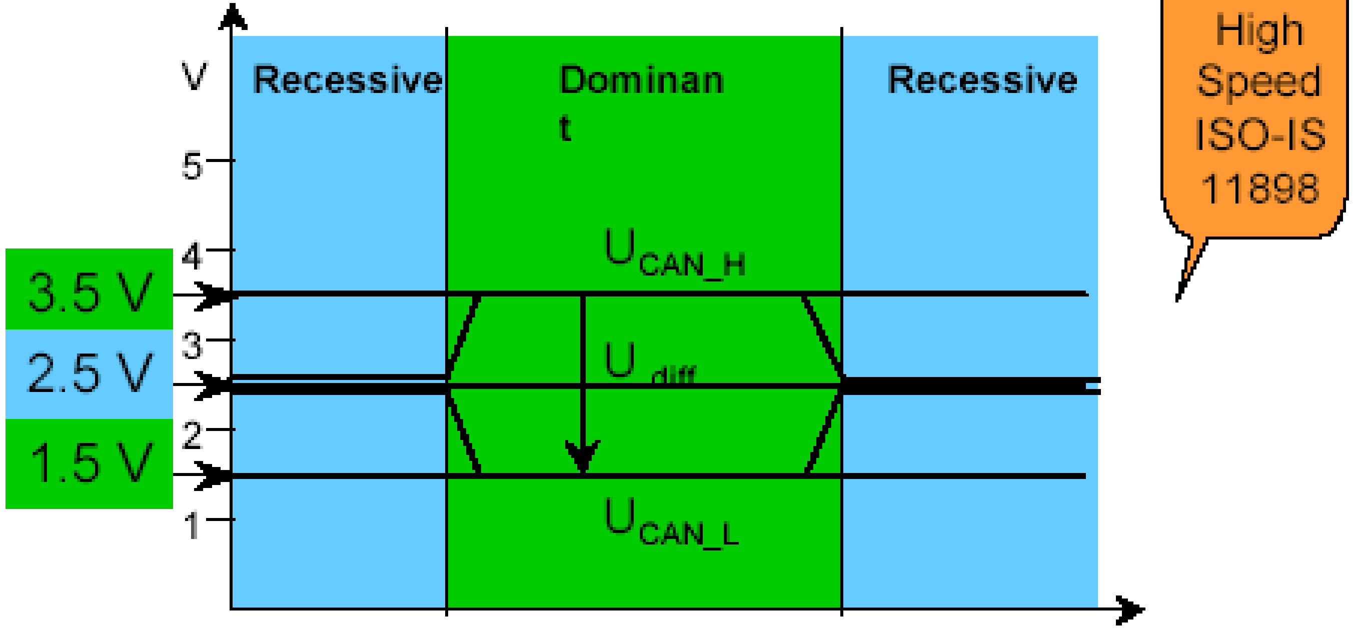


For the use of CAN in vehicles two standards have been defined for the bus interface:

CAN High Speed according to ISO-IS 11898 for bit rates between 125 kbps and 1 Mbps

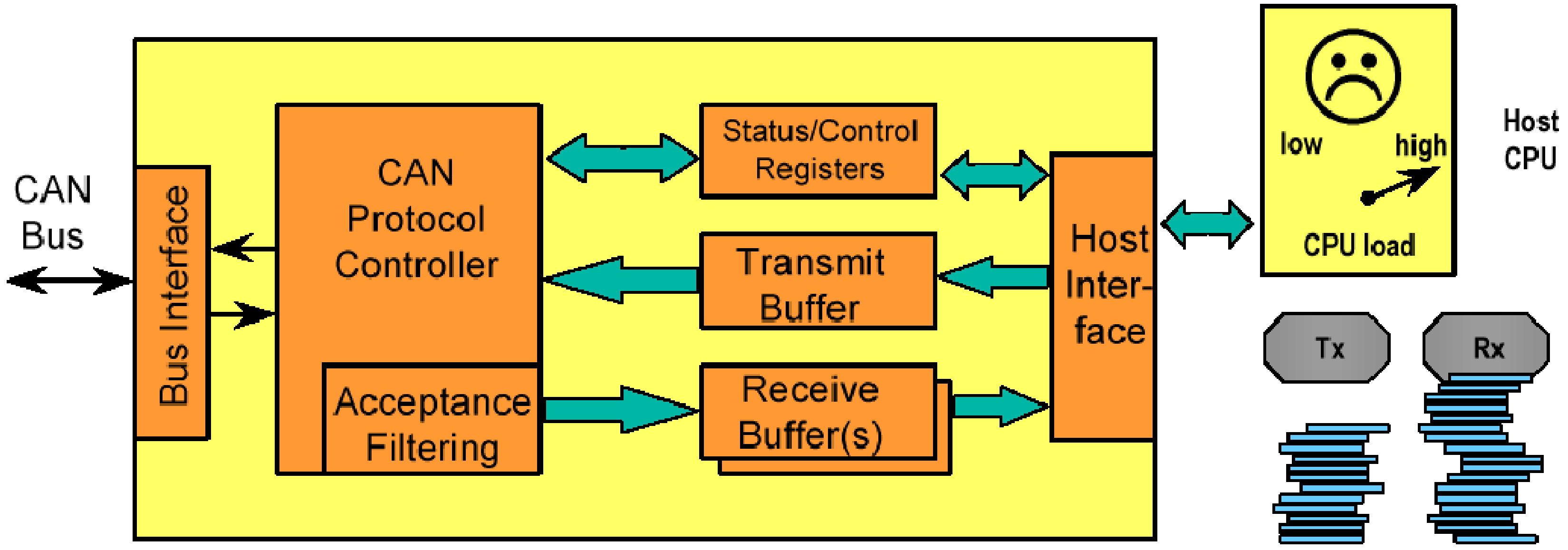
CAN Low Speed according to ISO-IS 11519-2 for bit rates up to 125 kbps

Bus Levels according to ISO-IS 11898



•These are the bus levels according to ISO-IS 11898. A recessive bit is represented by both CAN bus lines driven to a level of about 2.5 V so that the differential voltage between CAN_H and CAN_L is around 0 V.
•A dominant bit is represented by CAN_H going to about 3.5 V and CAN_L going to about 1.5 V. This results in a differential voltage for a dominant bit of about 2V.

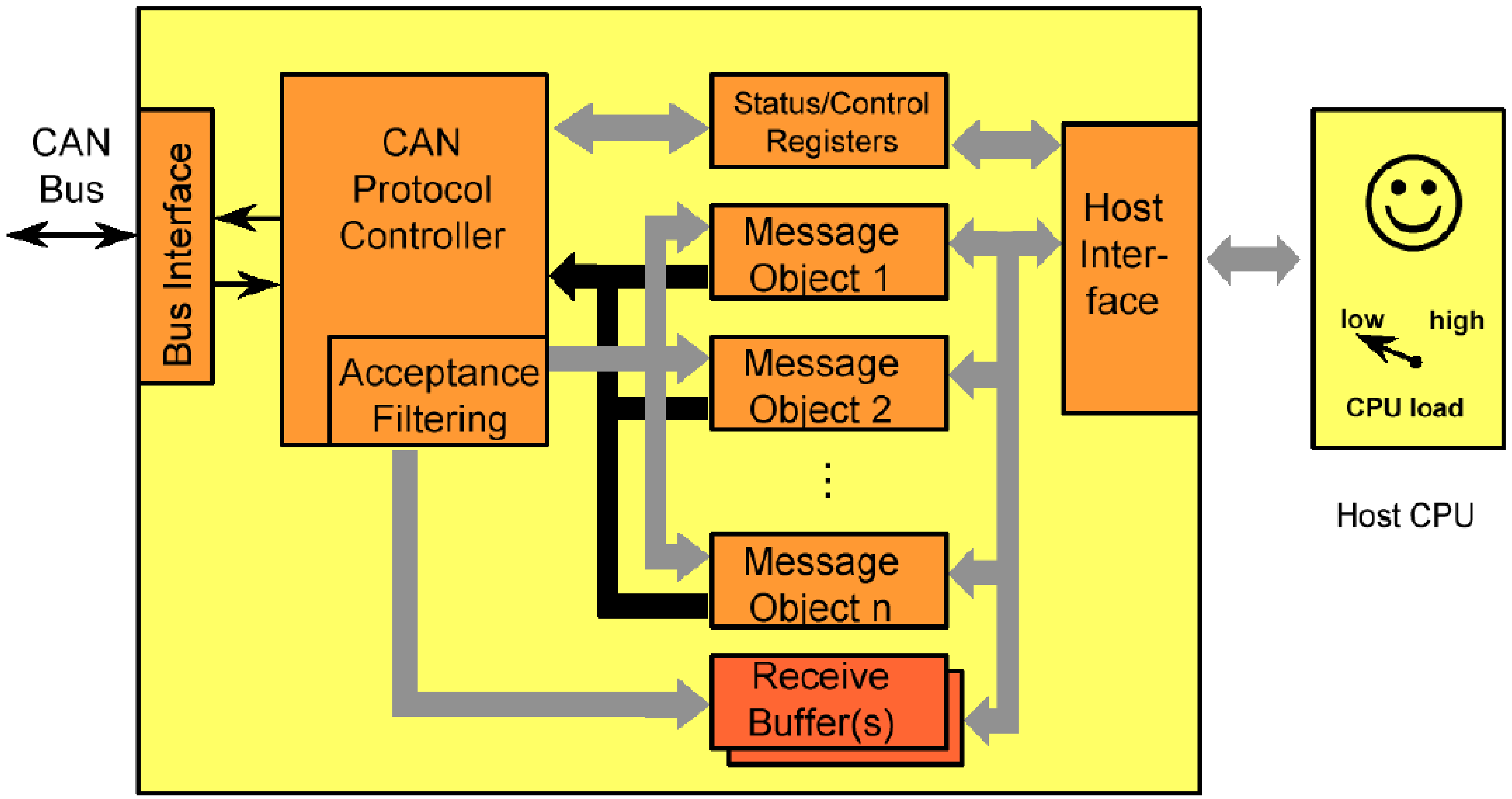
A Basic CAN controller



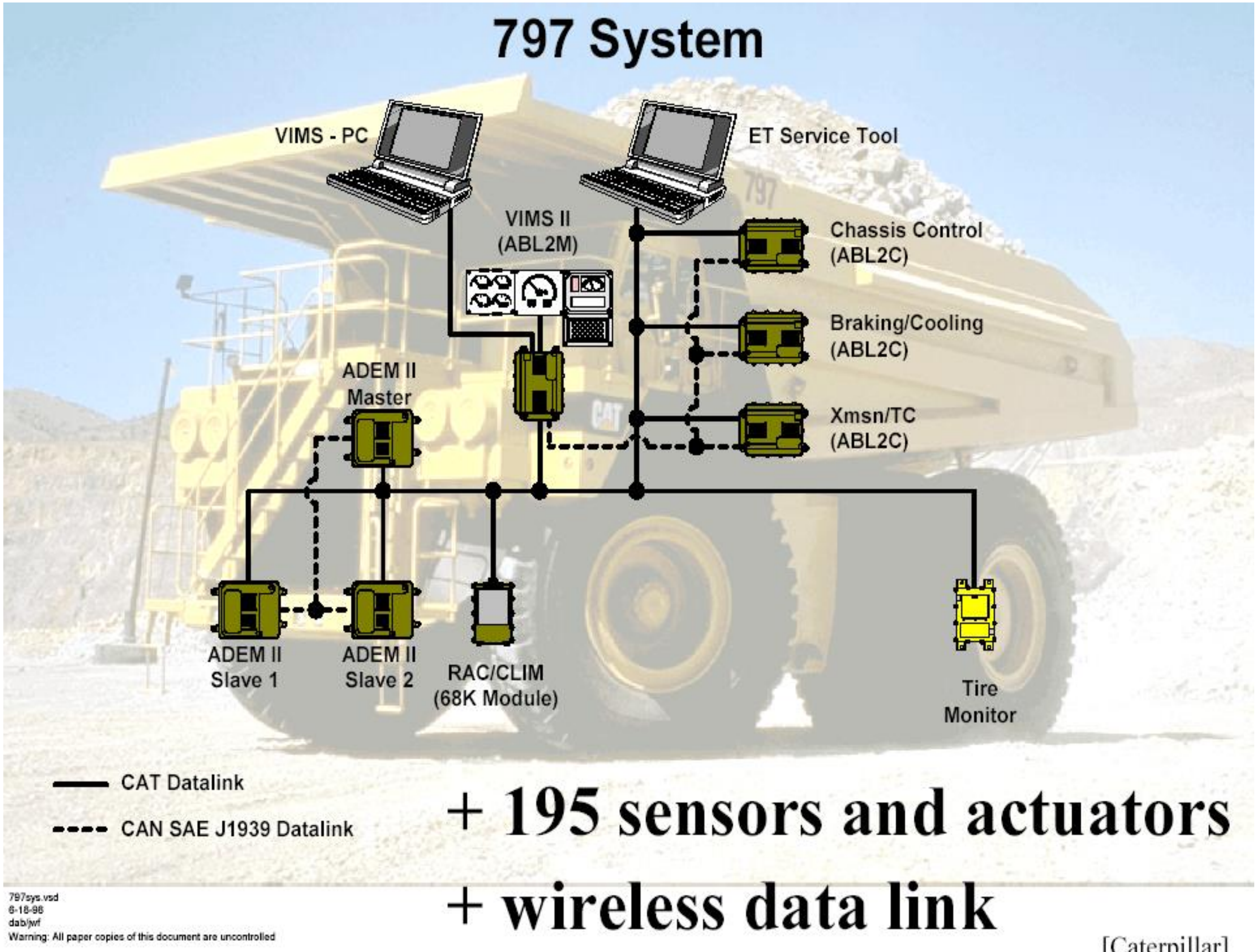
➢ Cheap CAN controller – CPU could get overrun with messages even if it didn’t need them.

Full CAN Controller

- Hardware message filters sort & filter messages without interrupting CPU



CAN (SAE J1939) Example: Caterpillar 797

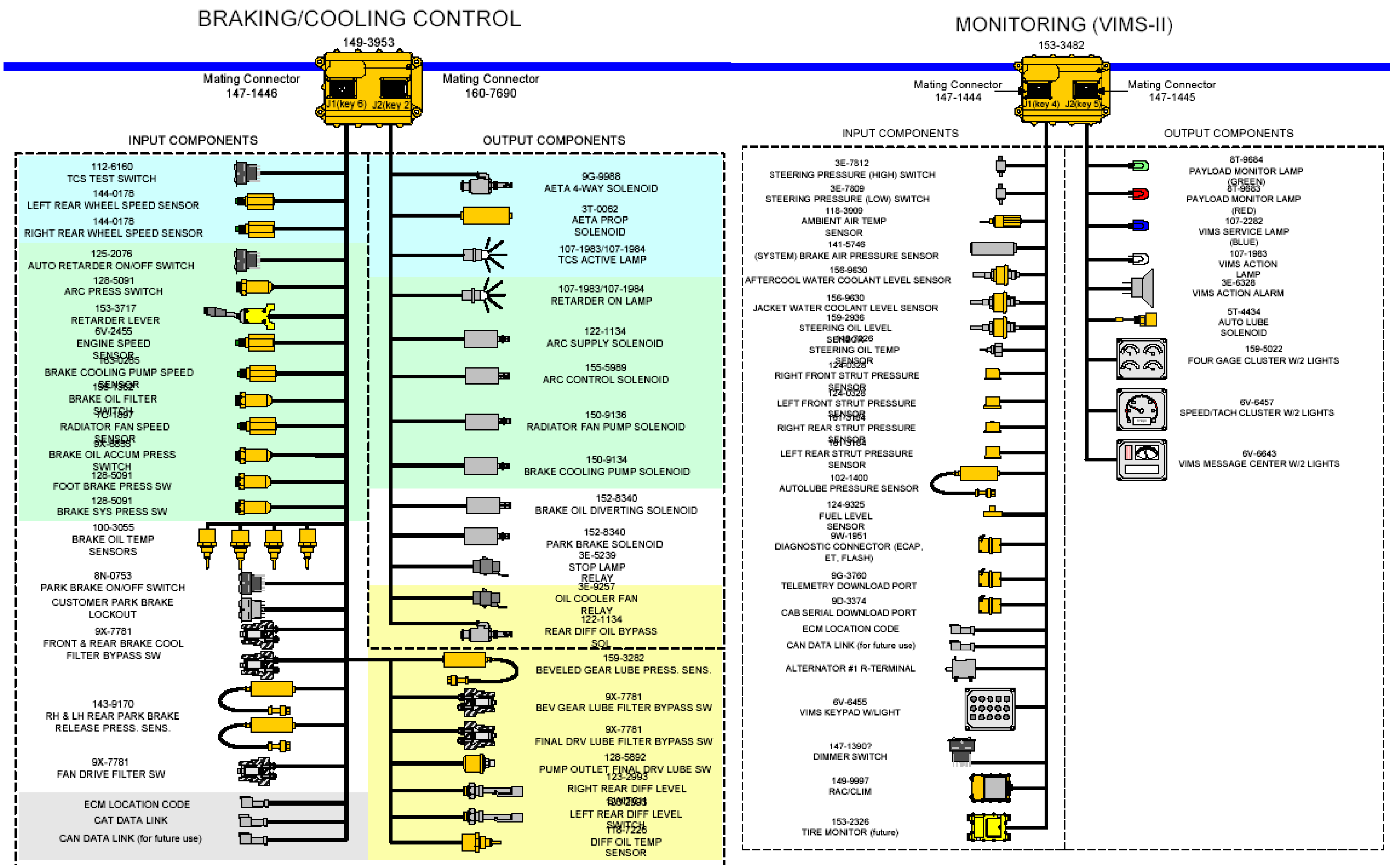


+ 195 sensors and actuators

+ wireless data link

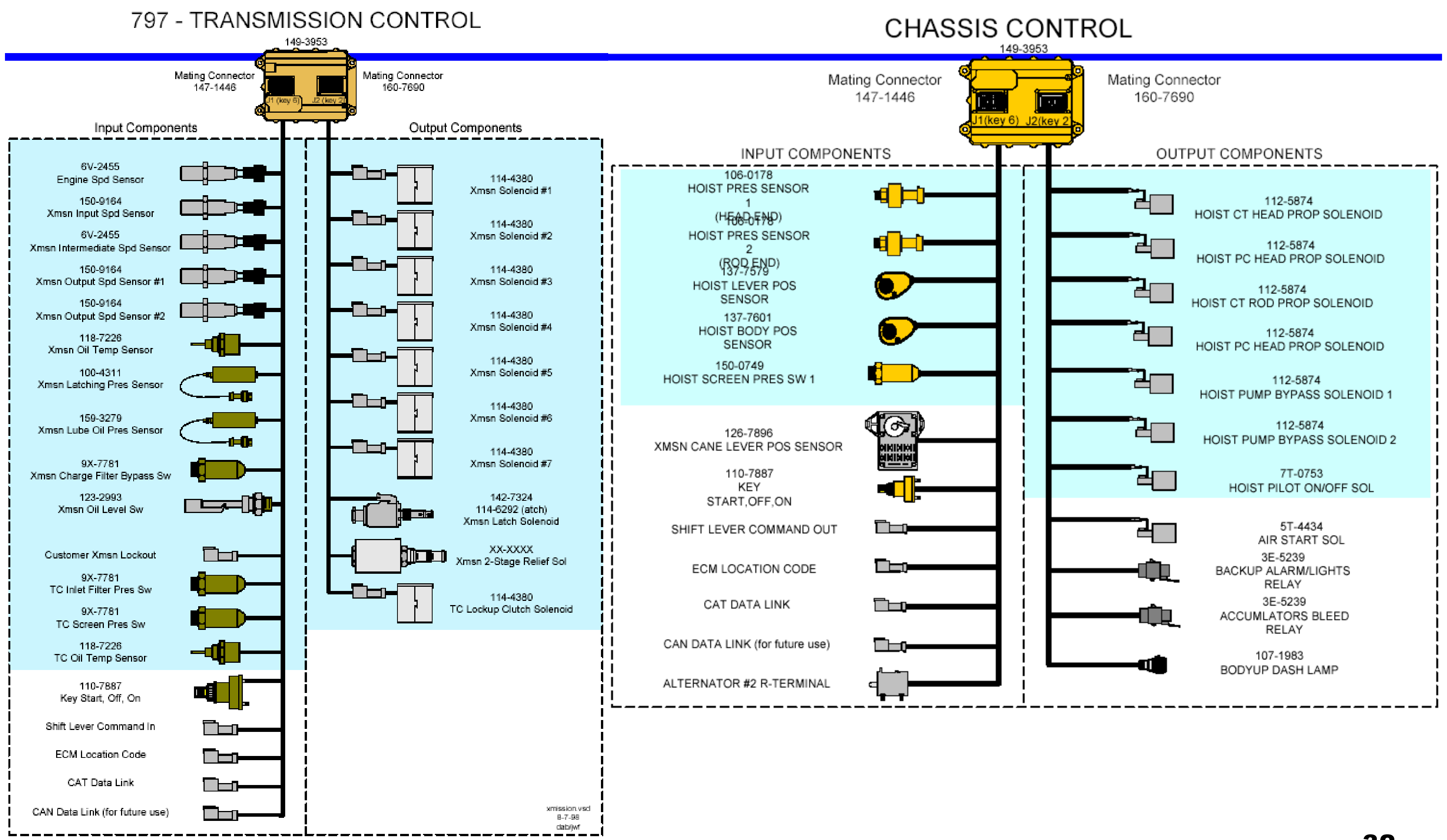
[Caterpillar]

Caterpillar example



BRAKING/COOLING CONTROL
149-3953
Mating Connector 147-1446
Mating Connector 160-7690
J1(key 6)
J2(key 2)
INPUT COMPONENTS
OUTPUT COMPONENTS
112-6160 TCS TEST SWITCH
144-0178 LEFT REAR WHEEL SPEED SENSOR
144-0178 RIGHT REAR WHEEL SPEED SENSOR
125-2076 AUTO RETARDER ON/OFF SWITCH
128-5091 ARC PRESS SWITCH
153-3717 RETARDER LEVER
6V-2455 ENGINE SPEED SENSOR
163-0285 BRAKE COOLING PUMP SPEED SENSOR
156-1362 BRAKE OIL FILTER SWITCH
7C-1897 RADIATOR FAN SPEED SENSOR
9X-8553 BRAKE OIL ACCUM PRESS SWITCH
128-5091 FOOT BRAKE PRESS SW
128-5091 BRAKE SYS PRESS SW
100-3055 BRAKE OIL TEMP SENSORS
8N-0753 PARK BRAKE ON/OFF SWITCH
CUSTOMER PARK BRAKE LOCKOUT
9X-7781 FRONT & REAR BRAKE COOL FILTER BYPASS SW
143-9170 RH & LH REAR PARK BRAKE RELEASE PRESS. SENS.
9X-7781 FAN DRIVE FILTER SW
ECM LOCATION CODE
CAT DATA LINK
CAN DATA LINK (for future use)
9G-9988 AETA 4-WAY SOLENOID
3T-0062 AETA PROP SOLENOID
107-1983/107-1984 TCS ACTIVE LAMP
107-1983/107-1984 RETARDER ON LAMP
122-1134 ARC SUPPLY SOLENOID
155-5989 ARC CONTROL SOLENOID
150-9136 RADIATOR FAN PUMP SOLENOID
150-9134 BRAKE COOLING PUMP SOLENOID
152-8340 BRAKE OIL DIVERTING SOLENOID
152-8340 PARK BRAKE SOLENOID
3E-5239 STOP LAMP RELAY
3E-9257 OIL COOLER FAN RELAY
122-1134 REAR DIFF OIL BYPASS SOL
159-3282 BEVELED GEAR LUBE PRESS. SENS.
9X-7781 BEV GEAR LUBE FILTER BYPASS SW
9X-7781 FINAL DRV LUBE FILTER BYPASS SW
128-5892 PUMP OUTLET FINAL DRV LUBE SW
123-2993 RIGHT REAR DIFF LEVEL SWITCH
123-2993 LEFT REAR DIFF LEVEL SWITCH
118-7226 DIFF OIL TEMP SENSOR
MONITORING (VIMS-II)
153-3482
Mating Connector 147-1444
Mating Connector 147-1445
J1(key 4)
J2(key 5)
INPUT COMPONENTS
OUTPUT COMPONENTS
3E-7812 STEERING PRESSURE (HIGH) SWITCH
3E-7809 STEERING PRESSURE (LOW) SWITCH
118-3909 AMBIENT AIR TEMP SENSOR
141-5746 (SYSTEM) BRAKE AIR PRESSURE SENSOR
156-9630 AFTERCOOL WATER COOLANT LEVEL SENSOR
156-9630 JACKET WATER COOLANT LEVEL SENSOR
159-2936 STEERING OIL LEVEL SENSOR
102-2226 STEERING OIL TEMP SENSOR
124-0328 RIGHT FRONT STRUT PRESSURE SENSOR
124-0328 LEFT FRONT STRUT PRESSURE SENSOR
161-3184 RIGHT REAR STRUT PRESSURE SENSOR
161-3184 LEFT REAR STRUT PRESSURE SENSOR
102-1400 AUTOLUBE PRESSURE SENSOR
124-9325 FUEL LEVEL SENSOR
9W-1951 DIAGNOSTIC CONNECTOR (ECAP, ET, FLASH)
9G-3760 TELEMETRY DOWNLOAD PORT
9D-3374 CAB SERIAL DOWNLOAD PORT
ECM LOCATION CODE
CAN DATA LINK (for future use)
ALTERNATOR #1 R-TERMINAL
6V-6455 VIMS KEYPAD W/LIGHT
147-1390? DIMMER SWITCH
149-9997 RAC/CLIM
153-2326 TIRE MONITOR (future)
8T-9684 PAYLOAD MONITOR LAMP (GREEN)
8T-9683 PAYLOAD MONITOR LAMP (RED)
107-2282 VIMS SERVICE LAMP (BLUE)
107-1983 VIMS ACTION LAMP
3E-6328 VIMS ACTION ALARM
5T-4434 AUTO LUBE SOLENOID
159-5022 FOUR GAGE CLUSTER W/2 LIGHTS
6V-6457 SPEED/TACH CLUSTER W/2 LIGHTS
6V-6643 VIMS MESSAGE CENTER W/2 LIGHTS

Caterpillar example



797 - TRANSMISSION CONTROL
149-3953
Mating Connector 147-1446
Mating Connector 160-7690
J1 (key 6)
J2 (key 2)
Input Components
Output Components
6V-2455 Engine Spd Sensor
150-9164 Xmsn Input Spd Sensor
6V-2455 Xmsn Intermediate Spd Sensor
150-9164 Xmsn Output Spd Sensor #1
150-9164 Xmsn Output Spd Sensor #2
118-7226 Xmsn Oil Temp Sensor
100-4311 Xmsn Latching Pres Sensor
159-3279 Xmsn Lube Oil Pres Sensor
9X-7781 Xmsn Charge Filter Bypass Sw
123-2993 Xmsn Oil Level Sw
Customer Xmsn Lockout
9X-7781 TC Inlet Filter Pres Sw
9X-7781 TC Screen Pres Sw
118-7226 TC Oil Temp Sensor
110-7887 Key Start, Off, On
Shift Lever Command In
ECM Location Code
CAT Data Link
CAN Data Link (for future use)
114-4380 Xmsn Solenoid #1
114-4380 Xmsn Solenoid #2
114-4380 Xmsn Solenoid #3
114-4380 Xmsn Solenoid #4
114-4380 Xmsn Solenoid #5
114-4380 Xmsn Solenoid #6
114-4380 Xmsn Solenoid #7
142-7324 114-6292 (atch) Xmsn Latch Solenoid
XX-XXXX Xmsn 2-Stage Relief Sol
114-4380 TC Lockup Clutch Solenoid
xmission.vsd
8-7-98
dab/jwf
CHASSIS CONTROL
149-3953
Mating Connector 147-1446
Mating Connector 160-7690
J1(key 6)
J2(key 2)
INPUT COMPONENTS
OUTPUT COMPONENTS
106-0178 HOIST PRES SENSOR 1 (HEAD END)
106-0178 HOIST PRES SENSOR 2 (ROD END)
137-7579 HOIST LEVER POS SENSOR
137-7601 HOIST BODY POS SENSOR
150-0749 HOIST SCREEN PRES SW 1
126-7896 XMSN CANE LEVER POS SENSOR
110-7887 KEY START,OFF,ON
SHIFT LEVER COMMAND OUT
ECM LOCATION CODE
CAT DATA LINK
CAN DATA LINK (for future use)
ALTERNATOR #2 R-TERMINAL
112-5874 HOIST CT HEAD PROP SOLENOID
112-5874 HOIST PC HEAD PROP SOLENOID
112-5874 HOIST CT ROD PROP SOLENOID
112-5874 HOIST PC HEAD PROP SOLENOID
112-5874 HOIST PUMP BYPASS SOLENOID 1
112-5874 HOIST PUMP BYPASS SOLENOID 2
7T-0753 HOIST PILOT ON/OFF SOL
5T-4434 AIR START SOL
3E-5239 BACKUP ALARM/LIGHTS RELAY
3E-5239 ACCUMLATORS BLEED RELAY
107-1983 BODYUP DASH LAMP

"SAE Standard Workload" (53 messages) V/C = Vehicle Controller

Signal Number	Signal Description	Size /bits	J /ms	T /ms	Periodic /Sporadic	D /ms	From	To
1	Traction Battery Voltage	8	0.6	100.0	P	100.0	Battery	V/C
2	Traction Battery Current	8	0.7	100.0	P	100.0	Battery	V/C
3	Traction Battery Temp, Average	8	1.0	1000.0	P	1000.0	Battery	V/C
4	Auxiliary Battery Voltage	8	0.8	100.0	P	100.0	Battery	V/C
5	Traction Battery Temp, Max.	8	1.1	1000.0	P	1000.0	Battery	V/C
6	Auxiliary Battery Current	8	0.9	100.0	P	100.0	Battery	V/C
7	Accelerator Position	8	0.1	5.0	P	5.0	Driver	V/C
8	Brake Pressure, Master Cylinder	8	0.1	5.0	P	5.0	Brakes	V/C
9	Brake Pressure, Line	8	0.2	5.0	P	5.0	Brakes	V/C
10	Transaxle Lubrication Pressure	8	0.2	100.0	P	100.0	Trans	V/C
11	Transaction Clutch Line Pressure	8	0.1	5.0	P	5.0	Trans	V/C
12	Vehicle Speed	8	0.4	100.0	P	100.0	Brakes	V/C
13	Traction Battery Ground Fault	1	1.2	1000.0	P	1000.0	Battery	V/C
14	Hi&Lo Contactor Open/Close	4	0.1	50.0	S	5.0	Battery	V/C
15	Key Switch Run	1	0.2	50.0	S	20.0	Driver	V/C
16	Key Switch Start	1	0.3	50.0	S	20.0	Driver	V/C
17	Accelerator Switch	2	0.4	50.0	S	20.0	Driver	V/C
18	Brake Switch	1	0.3	20.0	S	20.0	Brakes	V/C
19	Emergency Brake	1	0.5	50.0	S	20.0	Driver	V/C
20	Shift Lever (PRNDL)	3	0.6	50.0	S	20.0	Driver	V/C
21	Motor/Trans Over Temperature	2	0.3	1000.0	P	1000.0	Trans	V/C
22	Speed Control	3	0.7	50.0	S	20.0	Driver	V/C
23	12V Power Ack Vehicle Control	1	0.2	50.0	S	20.0	Battery	V/C
24	12V Power Ack Inverter	1	0.3	50.0	S	20.0	Battery	V/C
25	12V Power Ack I/M Contr.	1	0.4	50.0	S	20.0	Battery	V/C
26	Brake Mode (Parallel/Split)	1	0.8	50.0	S	20.0	Driver	V/C