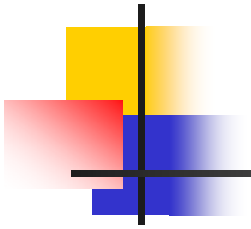
CAN BUS

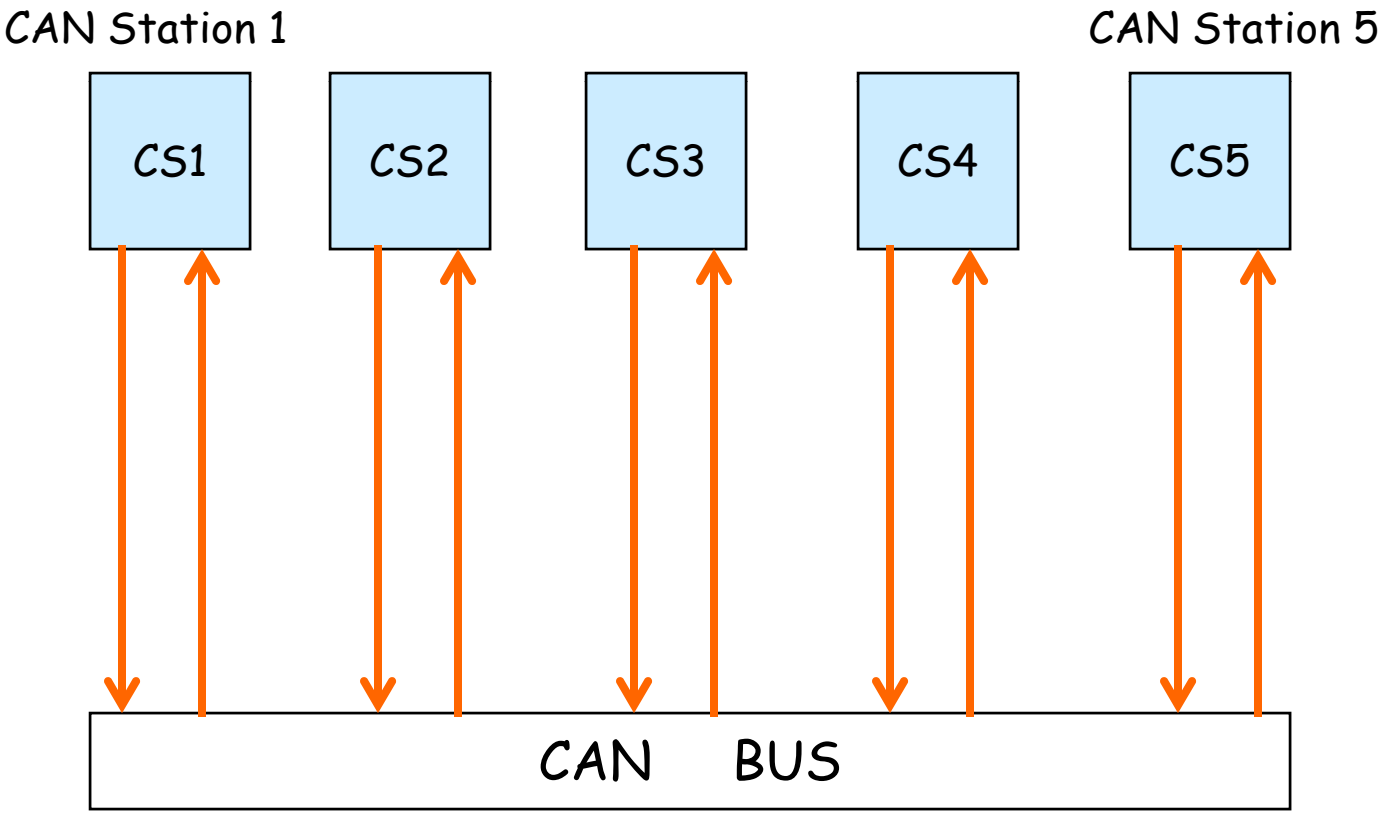
Abhishek Borkar(200911021)
Chandra Kanth (200701149)



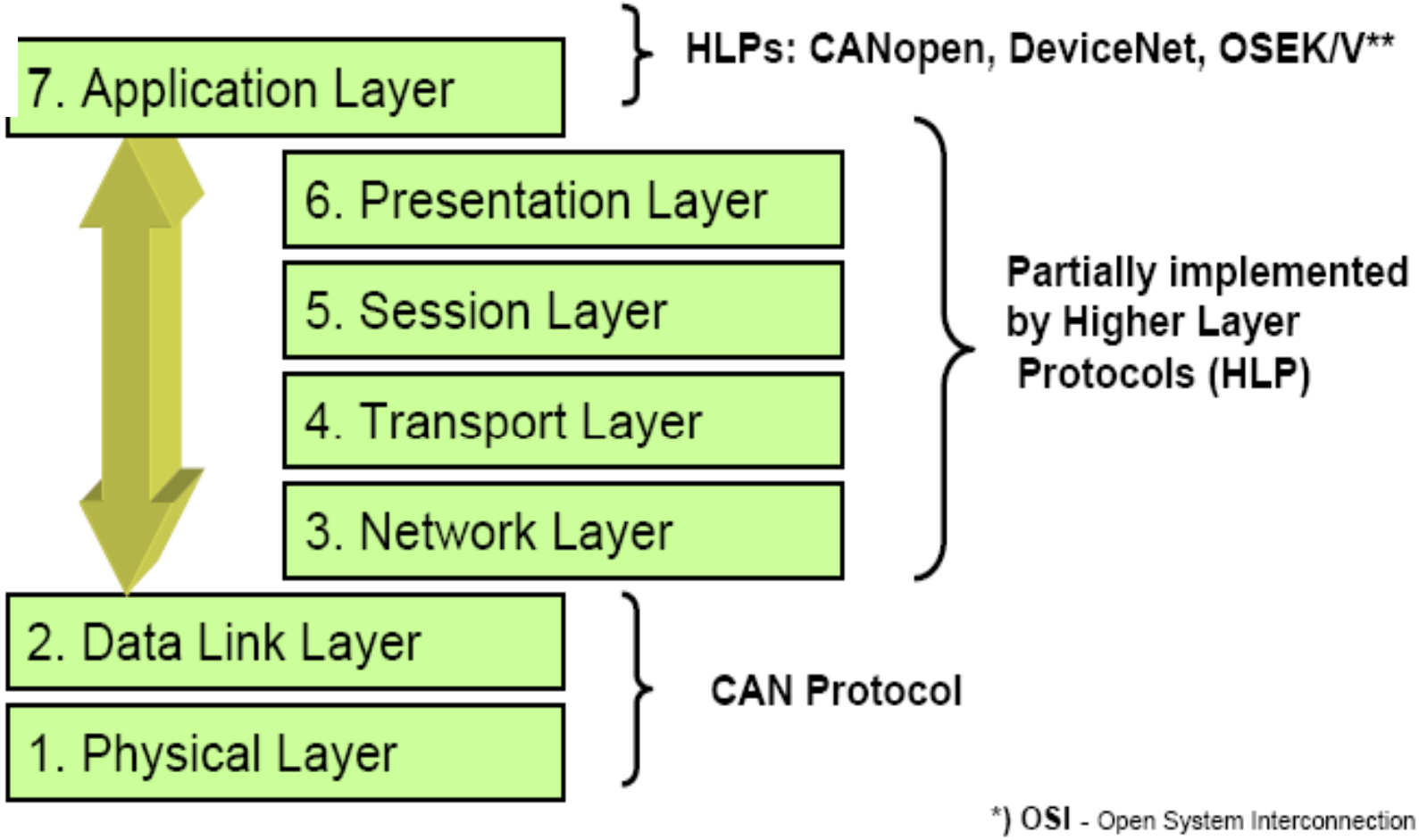
Introduction

- Controller Area Network (CAN) is a fast serial bus that is designed to provide
  - an efficient,
  - Reliable and
  - Economical link between sensors and actuators.
- CAN uses a twisted pair cable to communicate at speeds up to 1Mbit/s with up to 40 devices.
- The protocol was developed in 1980 by BOSCH for automotive applications
- It was developed to simplify the wiring in automobiles.

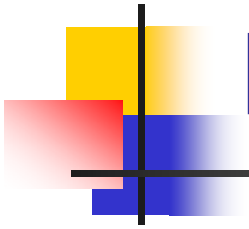
CAN Architecture



OSI Reference Model

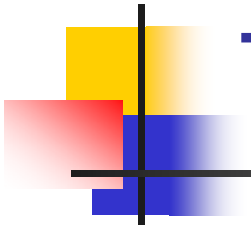


*) **OSI** - Open System Interconnection



HLPs (High Level Protocol)

- CANKingdom
- CANOpen
- DeviceNet
- J1939



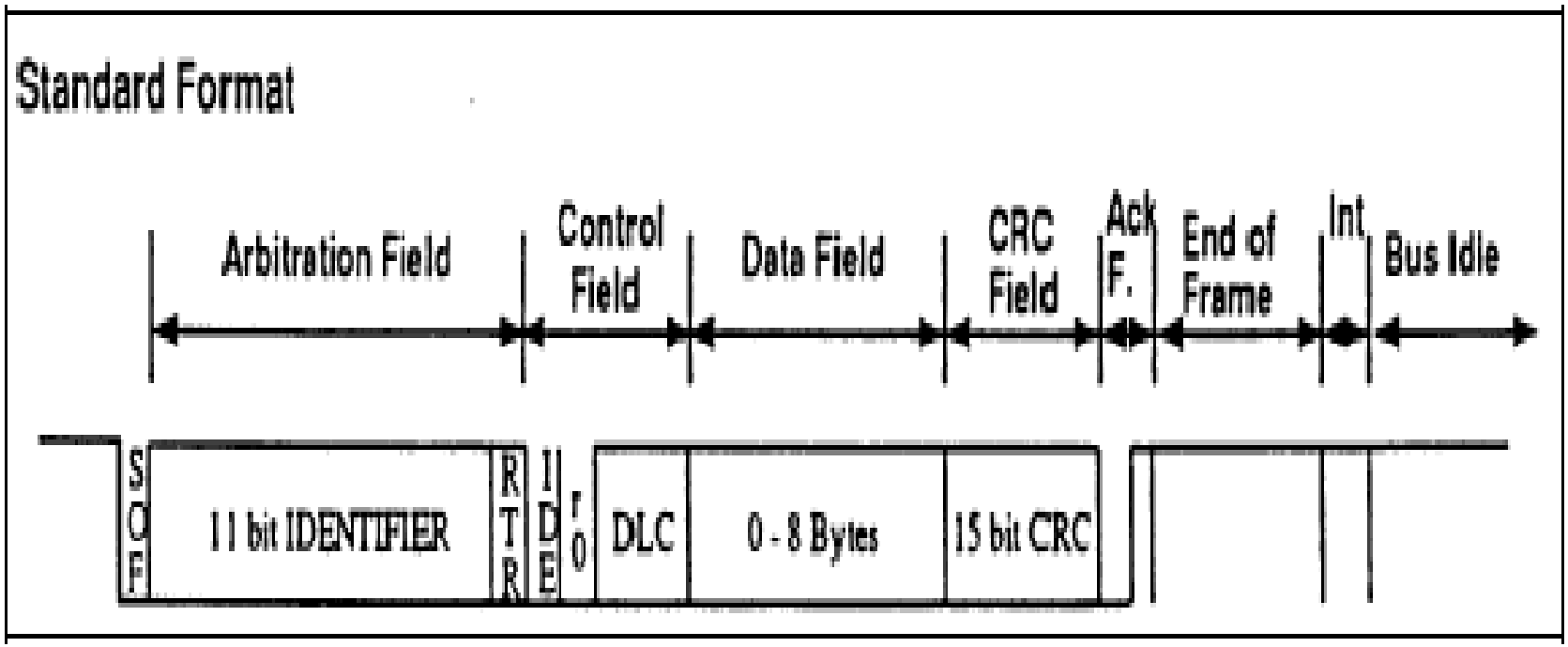
Types of CAN

- There are two types of CAN implementations depending in the size of the identifier field.

1) STANDARD: 11-bit wide identifier field.

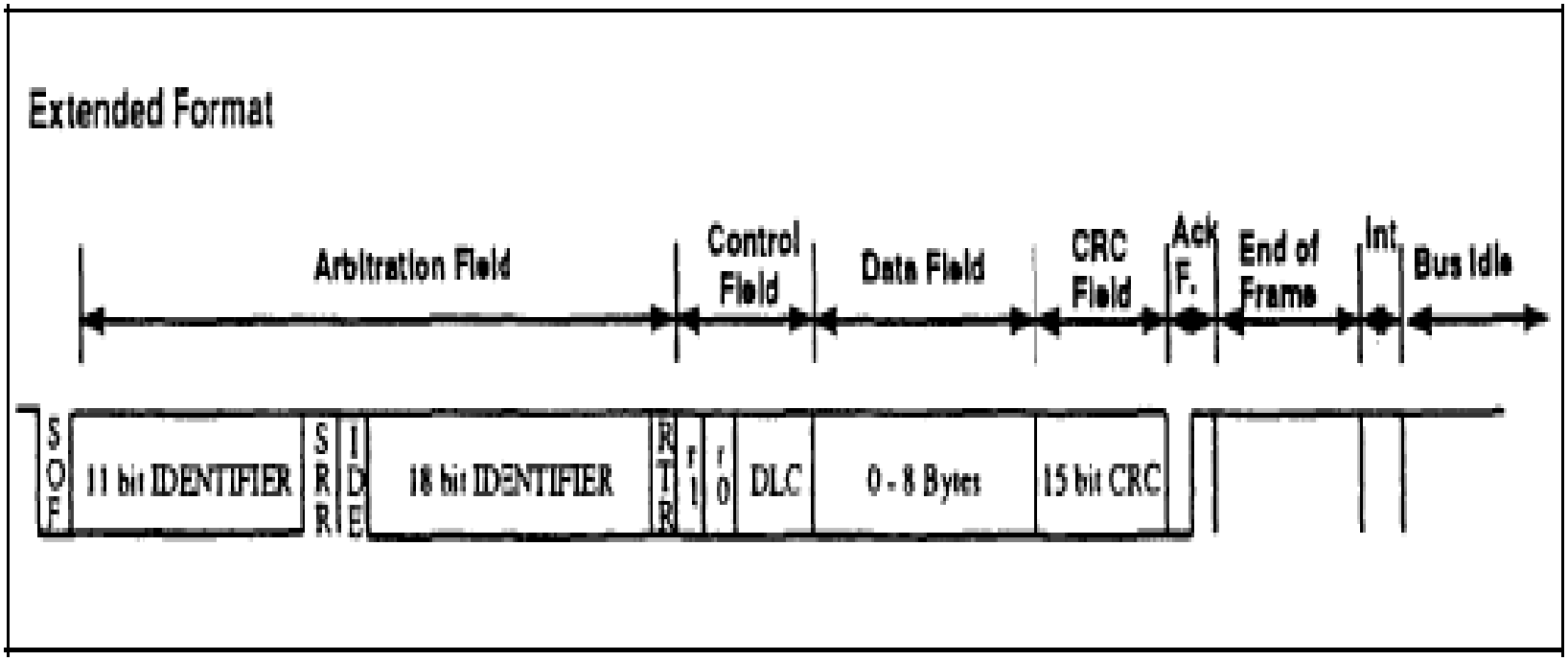
2) EXTENDED: 29-bit wide identifier field.

11 Bit Format



Message frame for standard format (CAN Specification 2.0A)

29 Bit Format



Message frame for standard format (CAN Specification 2.0A)

Messages Used For Communication

Four kinds of message frames are used.

1. **Data Frames**

Carries data from a transmitter to the receivers.

Bits:	1	12/32	6	0-64	16	2	7	
	SOF	Arbitration Field	Control field	Data field	CRC field	ACK field	EOF	Interframe space

2. **Remote frame**:

Transmitted by a bus unit to request the transmission of the data frame with the same identifier.

Bits:	1	11/29	6	16	2	7
	SOF	Arbitration field	Control field	CRC field	ACK field	EOF

3. **Error Frame**:

Any unit on detecting a error transmits an error frame.

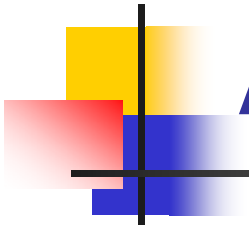
Bits:	6	8
	Error flag	Error delimiter

4**. Overload Frame**:

Detection of any one of overload condition make node to transmit overload frame**.**

Bits:	6	8
	Overload flag	Overload delimiter

Addressing in CAN



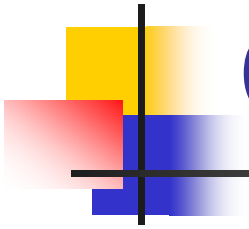
- Nodes do not have proper addresses
- Rather, each message has an 11-bit “field identifier”
  - In extended mode, identifiers are 29 bits
- Everyone who is interested in a message type listens for it
  - Works like this: “I’m sending an oxygen sensor reading”
  - Not like this: “I’m sending a message to node 5”

Features:

- It is designed for control and not for transmission of large blocks of data.
- Any node can access the bus when the bus is quiet
- Non-destructive bit-wise arbitration is used which allows 100% bandwidth usage without loss of data.
- Variable message priority based on 11-bit (or 29 bit) packet identifier
- Peer-to-peer and multi-cast reception
- Automatic error detection, signaling and retries.
- Data packets are 8 bytes long.

CAN Bus Vs Point-to-Point Connections

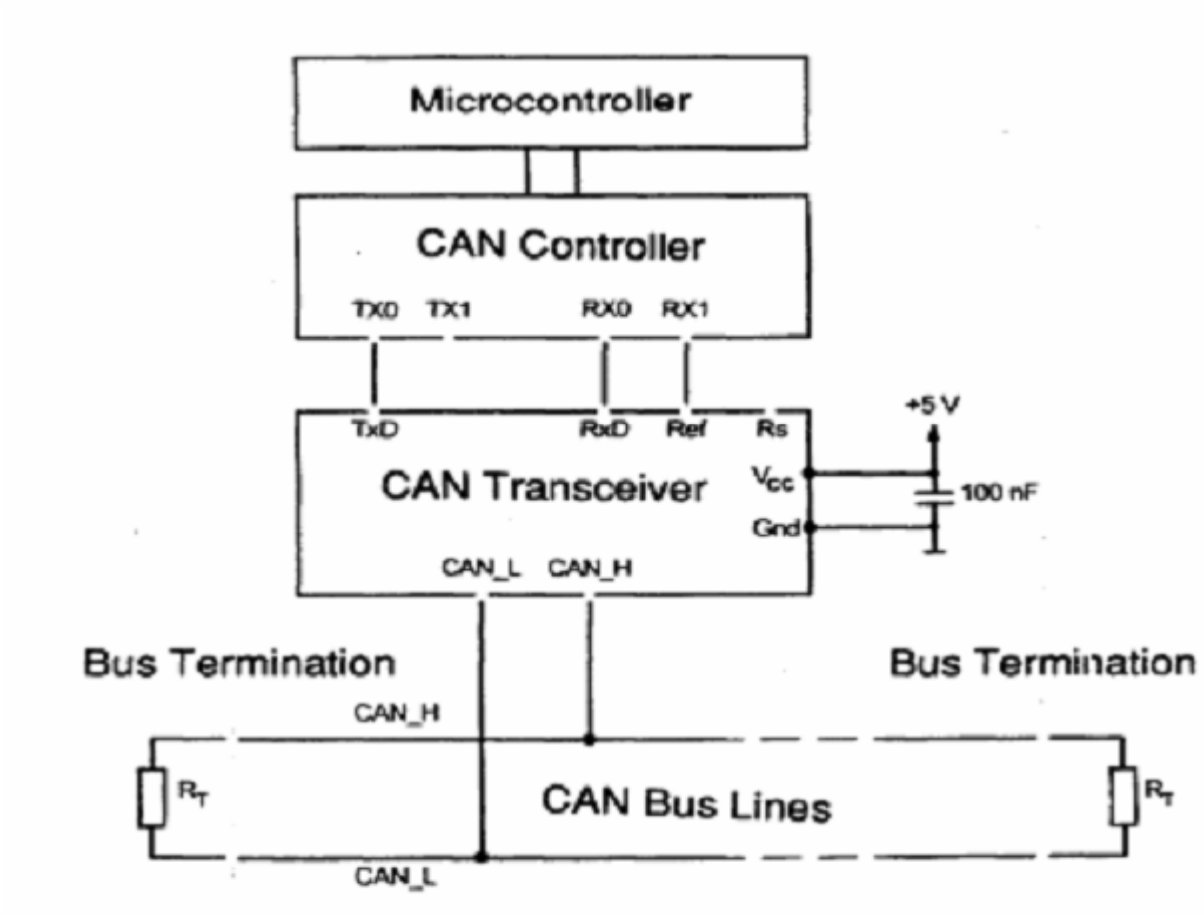
- By introducing one single bus as the only means of communication as opposed to the point-to-point network, we traded the channel access simplicity for the circuit simplicity.
- Since two devices might want to transmit simultaneously, we need to have a MAC protocol to handle the situation.
- CAN manages MAC issues by using a unique identifier for each of the outgoing messages
- Identifier of a message represents its priority.



CAN Interface

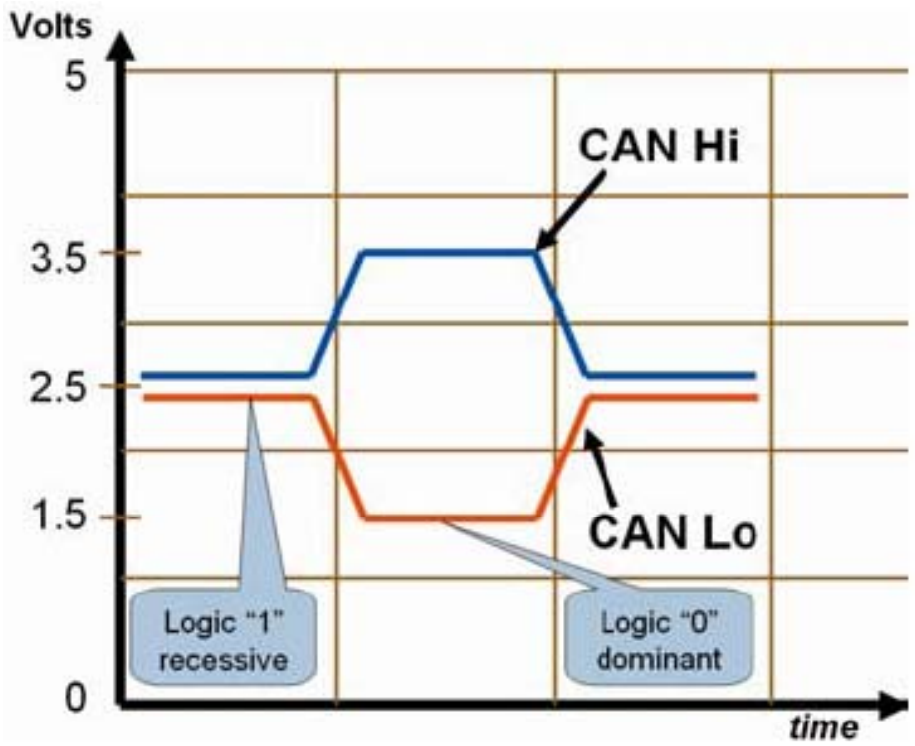
- Microcontroller
- CAN controller
- CAN transceiver
- Twisted pair wire terminated at both ends by 120 Ohm.

CAN Interface



Voltage Levels

- CAN Hi voltage with respect to ground changes between 2.5 to 4 volts nominal.
- CAN Lo changes from 2.5 to 1 volt.
- Therefore the difference between the two is either 0 volts (is logical “1”) or 2 volts (is logical “0”).
- 0 is known as the “recessive” state and 2 volts is the “dominant” state.



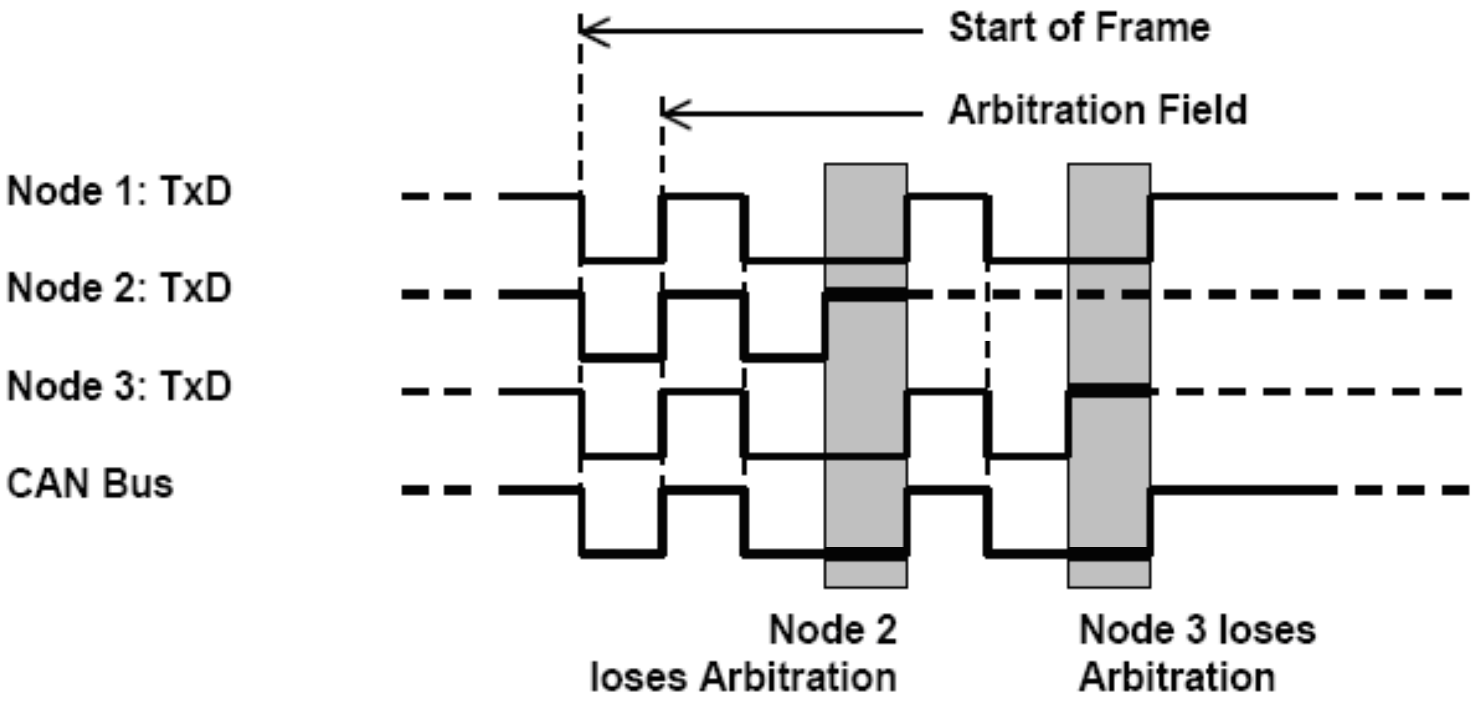
CAN bus logic

Node A	Node B	Node 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

"Wired-AND" function:
as soon as one node transmit
a dominant bit (zero)
the bus is in the dominant state

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

Bus Access and Arbitration



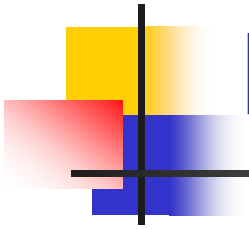
Carrier Sense Multiple Access/Collision Detection and Arbitration by Message Priority

Error Checking

- **Stuffing error** - a transmitting node inserts a high after five consecutive low bits (and a low after five consecutive high). A receiving node that detects violation will flag a bit stuffing error.
- **Bit error** - A transmitting node always reads back the message as it is sending. If it detects a different bit value on the bus than the one it sent, and the bit is not part of the arbitration field or in the acknowledgement field, an error is detected.
- **Checksum error** - each receiving node checks CAN messages for checksum errors.
- **Frame error** - There are certain predefined bit values that must be transmitted at certain points within any CAN Message Frame. If a receiver detects an invalid bit in one of these positions a Form Error (sometimes also known as a Format Error) will be flagged.
- **Acknowledgement Error** - If a transmitter determines that a message has not been ACKnowledged then an ACK Error is flagged.

Error Handling

- Every node is in error-active or error-passive state
  - Normally in error-active
- Every node has an error counter
  - Incremented by 8 every time a node is found to be erroneous
  - Decremented by 1 every time a node transmits or receives a message correctly
- If error counter reaches 128 a node enters error passive state
  - Can still send and receive messages normally
- If error counter reaches 256 a node takes itself off the network



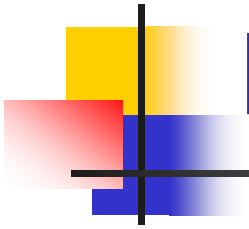
Microcontrollers with CAN

There are several manufacturers of microcontrollers with a CAN bus module viz.,

- Atmel (with both 8051 and AVR core)
- Microchip's PIC
- ST microelectronics
- ARM

Application Areas

- Automotive & Transportation
- Medical
- Building Automation
- Domestic & Food distribution appliances
- Robotics
- Production Automation
- Agriculture



References

- CAN tutorials - By ATMEL
- Automotive electronics : CAN and LIN buses - By Matrix Multimedia
- CAN Primer: Creating your own Network - By Keil
- http://electrosofts.com/can/