STREAM01 / Mastère SE course E10

Distributed Embedded Systems and realtime networks

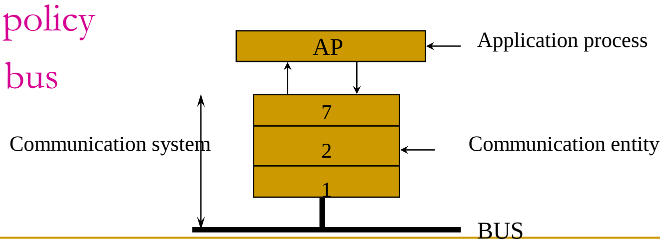
Industrial networks CAN / VAN

Marie-Agnès Peraldi-Frati
AOSTE Project
I3S-INRIA
map@unice.fr

January 2008 1

CAN and VAN networks

- Controller Area Network / Vehicle Area Network
- Design by BOSCH / PSA
- Dedicated to automotive applications
- Norme ISO with VAN et J1850
- Client-Serveur policy
- Asynchronous bus



CAN / VAN protocol

- Main properties
  - multi-master / multi-slaves behavior
  - Communication by broadcast or point à point
  - Medium access with non destructive bit arbitration
  - Message hierarchy
  - Latency guaranty
  - Error detection and monitoring
  - Automatic relay of erroneous messages
  - Automatic disjunction of failure nodes

CAN protocol

- **Physical layer**
  - Manage the physical transfer of frame
  - Transmission speed from 125Kbits/s to 1Mb/s
  - Length from 30 to 250 meters
  - Physical medium not imposed by the standard
- **Data link layer**
  - 2 sub-layers
  - LLC : achieves the **filtering** of **received frames**, and the **link** with the **application layer** (data transfer requests) **and** the **recovering from errors**
  - MAC implements the **bus arbitration**, the **data demand**, the reception from other layers and the acknowledge to these layers.
  - MAC **encapsulates** and **de-encapsulate** data
  - MAC inserts **synchronization bit** («stuffing», de-stuffing), detect errors.

CAN protocol

- **Routing of Message**
  - A message is located by an identifier.
  - This identifier indicates the content and the direction of data
  - Arbitration is made according to the recessive or dominant characteristic of the identifier.
- **Addressing**
  - Addressing is based on messages
  - Impossible to address nodes directly
  - An identifier gives the priority (arbitration field 11 bits)

VAN protocol… in addition …

- **Protocol :** possible to include in the request frame
  - Response mechanism
  - Acknowledge mechanism
- **Addressing**
  - 12 bits for the identifier
  - 28 bytes for the data field
- **Transfer**
  - Message can be sent to a single node (one to one communication)
  - Message can be broadcasted
  - Message with data request may have a contiguous response (inside the frame) or a deferred response

6

VAN protocol : Different communications

- **Master Mode**
  - Operation start at the symbol N° 0 of the VAN frame (Rank 0)
  - Reception of data
- **Slave Mode**
  - Generate data only on request from a master
    - Response inside the request frame (Rank16)
    - Data reception
    - Acquisition of broadcasted data
- **Synchronous mode**
  - Same characteristic of a slave node
  - Difference : it can produce data apart from solicitation of a master
  - Synchronization point is the start bit (Rank 1)

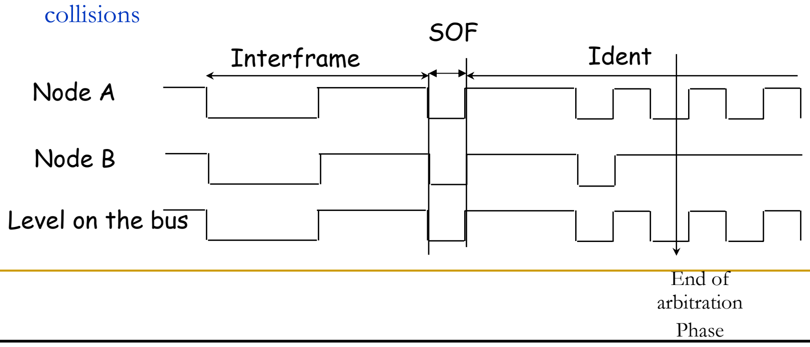
7

Medium access

- Multi-master
  - A node in an **«error active» state** can access the bus as soon as the bus is traffic free.
  - If several nodes want to emit simultaneously on the bus
    - The node with the higher identifier keeps the bus
    - The others withdraw
- Arbitration with contention
  - Physical transmitters compare the level of each bit with the bit present on the bus
    - If there are equal with the one on the bus the nodes continue the transmission
    - When a recessive bit is transmitted whereas a dominant one is detected on the bus => the node stop the transmission
    - When a dominant bit is transmitted whereas a recessive bit is detected => The node detect a «bit error».

Medium access

- CSMA (Carrier Sense Multiple Access)/CR Protocol
  - A frame can be transmitted after an idle delay on the bus (Inter frame separation >= 3 bits)
- Collisions solving
  - An identifier must have a single value on the system
  - It must exist a unique station that produce a frame with this identifier
  - Transmitting request frame with the same characteristics can produce unsolvable collisions



Medium reading

- Filtering of the frame in reception
  - By identifier on the entering frame
  - Proceeded by the DLL layer by comparison with the identifier expected
  - Advantage : reduction of the treatments at the application layer

10

VAN frame

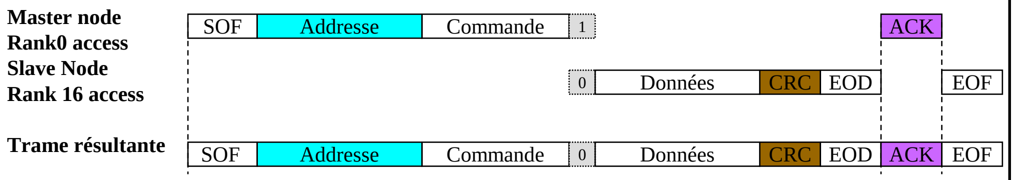
- format

SoF Preamble + Start bit	Arbitrage (12 bits)	Commande (4bits)	Données (28 octets)	CRC (15bits)	EOD (1 bit)	ACK (1 bit)	EOF (1 bit)

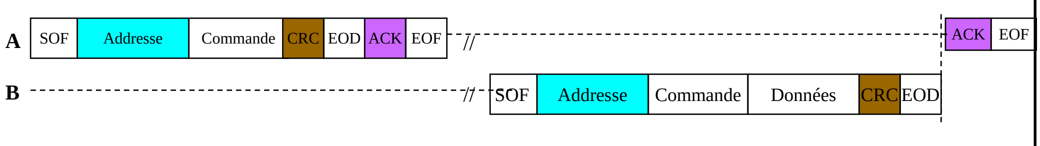
- EXT (reserved),
- RAK (acknowledge request),
- read/write,
- RTR (remote transmit request)

Reading a data on VAN

- Reading with response inside the frame

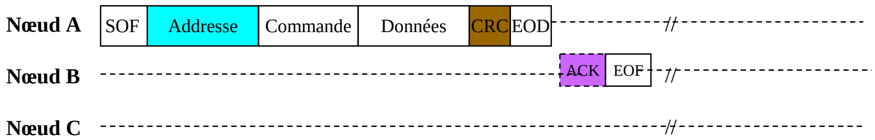


- Reading with deferred response



Transfert of data

- Point to point mode

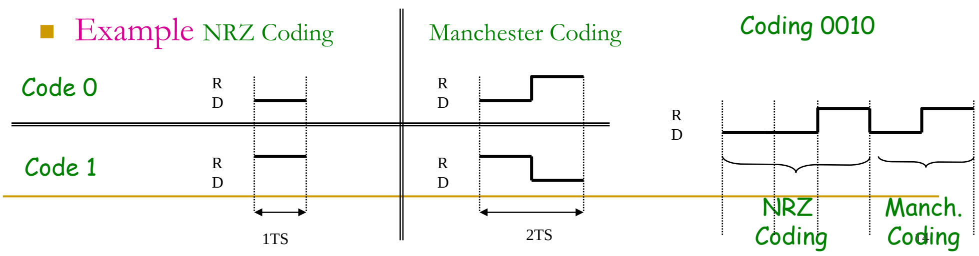


  - The more secure mode because ACK available inside the frame
- Broadcast mode
  - No specific node
  - DLL monitors and filters the access to data
  - Not possible to ACK

13

VAN frame coding

- E-Manchester (Enhanced Manchester) code
  - 4 Bits groups
  - 3 primary bits encoded in NRZ (No Return to Zero)
  - 4th encoded using Manchester with a double NRZ
  - E_Mancherster coding not applied to synchronization symbols
- This coding allows
  - A systematic synchronization every 5 time slots.
  - Improve error detection
- Example



Manchester Coding of VAN frames

- Préambule : 8TS
- Start bit : 2TS
- Ident field: 15 TS (12 bits +3)
- Command field : 5TS (4 bits+1)
- Data field : n x 1OTS (n bytes)
- CRC field : 18 TS (15 bits+3)
- Symbols EOD 2TS, ACK 2TS, EOF 8TS
- Symbol IFS : 4 TS
- Efficiency :

$$C = nx10 / (64+(nx10)$$

15

CAN frames

- **Standard frames**
  - 4 specific frame + a inter-frame for a temporal delay
    - request frame
    - data frame
    - error frame
    - overload frame :
    - Inter-frame : Temporal distance between data and remote frames

CAN frame

- **Request frame** Treq = 44 micros à 1Mb/s

  7 fields

SoF (1bit)	Identifier (11 bits)	RTR (1 bit)	Command (6bits)	CRC (15+1bits)	ACK (1+1bits)	EOF (7bits)

- **Data frame** Tdat = 108 micros à 1MB/s

  8 fields

SoF (1bit)	Identifier (11 bits)	RTR (1 bit)	Command (6bits)	Data 64bits)	CRC (15 +1bits)	ACK (1+1bits)	EOF (7bits)

CAN encoding

- **NRZ encoding** (Non return to Zero) with bit stuffing
  - for 5 successive bits with the same level, the protocol adds a bit with an inverse polarity in the frame
- **Bit stuffing** is applied on specific fields of the data frame and the request frame:
    - SOF
    - Identifier,
    - Command,
    - Data,
    - 15 bits of CRC
  - The 10 remaining bits are not stuffed.

- This encoding improves error detection

18

Communication errors

- In a distributed system it can happen that external electromagnetic Interference causes the correlated mutilation of all messages: a blackout
- Communication system should detect and continue with its operation as soon as the blackout disappears.
  - Errors must be reported consistently

19

Errors management

- Two kinds of errors can be detected by a node during transmission or by observation of the medium.
- **Errors :**

  From the physical layer :
  - **Bit error** : on the data or control field
  - **Encoding error** : code violation on the data field or the control field

  On the frame
  - **Error in the format** : encoding error or synchronization error on the symbols SOF, EOD, ACK, EOF
  - **Error on CRC**
  - **Error on ACK**: an Ack is present in the frame whereas ack was not requested and vice versa.

  These errors are reported by sending an error frame on the bus.

20

Error detection mechanisms

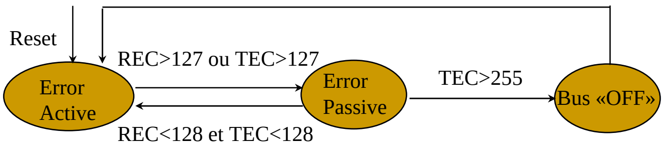
- Bit Error or bit stuffing error for CAN
  - The transmitter checks if the bit level on the bus is the same level than the bit he has transmitted. If its not the case => bit error
  - More than 5 consecutive bits with the same level are transmitted on the bus => Bit error
  - If a disturbance modify one of the 5 bits prior to the stuff bit, this error is impossible to detect with bit stuffing but the CRC can detect the error.
- CRC error or acknowledge error
  - The CRC calculus is different on the transmitter side and the receiver side.
  - Acknowledge error is detected by a transmitter when it observes an absence of a dominant bit during ACK slot.
- Error inside the frame

Mechanism for the management of errors confinement

- « Confinement » of errors
  - This mechanism allows to locate the hardware defects and perturbations.
  - The closest node must react with the highest priority
    - two internal counters :
      - the Transmit Error Counter
      - the Receive Error Counter
    - Weighted counting of errors.
    - This mechanism is implemented on hardware.

Mechanism for «confinement » errors management

- Three error states
  - **error active :** counters between 0 and 127.
  - **error passive** : counters between 128 and 255.
  - **bus off** : over 255



Outcome after errors detection

- Localization of the erroneous node
  - Counters give a local vision of errors in transmission or reception.
  - Counters of a node responsible for an error are incremented much more than other nodes
  - detection possible by
    - Stations that switch in a «bus off» state
    - Error frame with their «error flag»
    - The access to local counters is not possible

Errors report

- A node which detect an error must report it to the network
  - To inform the partners of the network and to take adapted decision to correct the error
  - To notify the other partners from the local state of nodes at the moment
  - «local quality» of the network
- The node transmits a different error flag depending of its counters states
  - A passive error flag in case of «error passive» state
  - An active error flag in case of «error active» state

Errors report

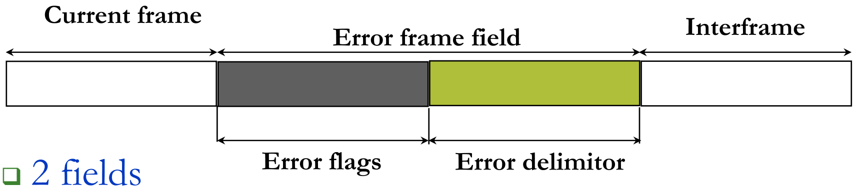
- **Error frame**
  - The trigger of error frame occurs at different instants depending of the origin of the error:
    - Bit error
    - bit stuffing
    - Structure error
    - Ack error

    => The error frame is sent by the node that detects the error just after the EOF bit of the current frame.
    - error on CRC

    => In case of CRC error : the error frame transmission starts after the ACK field.
- The first field is a concatenation of error flags from nodes
- The second field is 6 recessive bits

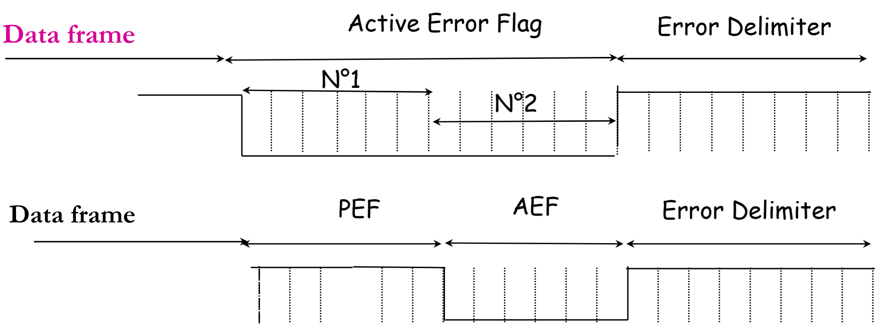
Errors report

- Error frame



- 2 fields
  - Field for Error flags : max size = 12 bits
    - active error flag = 6 dominant bits
    - passive error flag = 6 recessive bits
  - Error Delimiter = 8 recessive bits

Example of an error frame transmission

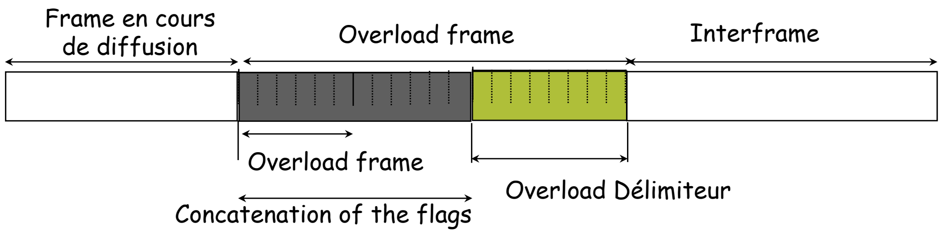


Error management

- Errors recovery
  - Automatic retransmission of the frame up to its correct transmission.
  - In case of persistent error counter overflow and the node take off
- Message validity
  - For the transmitter : if there is no error detection up to the end of EOF
  - For the receiver : if there is no error detection up to next to last bit of the frame
  - In all other case messages are considered as invalid.

The other frames

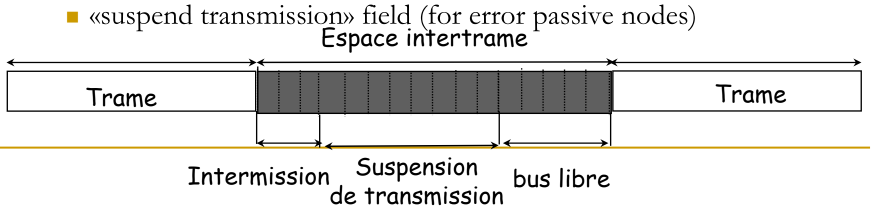
- Overload frame :
  - Indicates that a node is overloaded during a time lapse
  - Two fields
    - Overload flags,
    - Field delimiter



  - The maximum number of overload frame is 2

The other frames

- Inter frame
  - Data frame and request frame are distant from the others by an interframe.
  - Error frame and overload frame are not disjoined by an inter frame.
  - Inter frame is composed of 2 or 3 fields:
    - «intermission» field
    - «bus idle» field
    - «suspend transmission» field (for error passive nodes)



CAN Network load without reemission of frames

$$\rho_{can} = \sum_{j=1}^{m} \frac{T_{trj}}{S_j} + \sum_{i=1}^{n} \frac{T_{tri}}{P_i}$$

- *Sj* the minimal temporal distance between 2 sporadic frames,
- *Pi* the period of a a periodic frame i,
- *Ttrm* is the transfer time of the message *m*

32

Network load with a max of n emission of periodic frames

$$\rho_{can} = \sum_{j=1}^{n} \frac{T_{trj}}{Sj} + \sum_{i=1}^{n} \left(\frac{T_{tri}}{P_i} + \frac{(23 * \tau_{bit} + T_{tri}) * (n-1)}{P_i} \right)$$

23 is the number max of bits in an error frame

33

Worst Case Response time in a CAN network

- **T_m** the period of a given message *m*
- **J_m** the jitter on the queuing of the message *m*
- **b_m** the number of byte in the message *m*
- **C_m** the transmission time of message *m* onto the network
- ***Worst case response time*** **R_m** is the longest time between the queuing of a message and the time the message arrives at destination.
- **R_m** is the sum of a queuing delay **W_m** the transmission delay **C_m** and the jitter:
  - **$R_m = W_m + J_m + C_m$**

34

Worst Case Response time in a CAN network

$$W_m = \max_{\forall k \in hp(m)} (C_k) + \sum_{\forall j \in hp(m)} \left\lceil \frac{W_m + J_j + \tau_{bit}}{T_j} \right\rceil * C_j$$

Also called the ***blocking time***

p_m the priority of the message *m*

$hp(i)$ the set of message with priority > *i*

$lp(i)$ the set of messages with priority < *i*

J_j the jitter maximum for the message *m*.

The jitter represents the temporal distance between the demand of communication transmission of the host processor and the effective demand at the physical level. This delay is bounded and fixed by the suppliers.

35

Extension of CAN

- The 11 bits Identification field may be insufficient

  => extended CAN :
  - 29 bits identifier
- Compatibility between CAN standard et extended CAN
  - Its possible to put about both kind of frames over the same network

Application layer

- No standard for this layer (layer 7 of OSI)
- The protocol provides
  - The identification of the messages transmitted
  - The max size of message is 8 bytes
  - Acknowledgment in case of correct transmission
  - A secure transport of information
- Application layer had two bytes to the frame
  - throughput is reduced
  - Interoperability and reuse increased

Application layer

- Three application layers encapsulate the CAN protocol
  - CAL from CIA consortium
  - Device Net from Allen Bradley
  - SDS from Honeywell

Application layer

Device NET

- Device net protocol is integrated to the data field
- Messages are cut into several parts if there are too long.
- Data exchange is triggered
  - strobing
  - polling
  - periodically : every millisecond
  - By state exchange

Application layer

Device NET

- Different operating mode for data transfer
  - Producer consumer mode for input/Output data
    - Bandwidth optimization
    - A producer broadcast information
    - Several consumers keep the information
  - Client-Server mode for one to one messages
    - Point to point relation
    - Useful in case of reconfiguration or diagnosis of a particular node

References

H. Kopetz,
Real-Time Systems : Design Principles for Distributed Embedded Applications, Kluwer, 1997.

CIA Web site http://www.can-cia.org/can/protocol/

K. Tindell and A. Burns
Guaranteed Messsage Latencies for Distributed Safety-Critical Hard Real-Time Control Networks YCS 229, Department of Computer Science, University of York (1994).

K. Tindell1, A. Burns, A. Wellings
Analysis of hard real-time communications
Realtime systems P259-263, 1994

41