# PSI 3561

## ELETRÔNICA AUTOMOTIVA

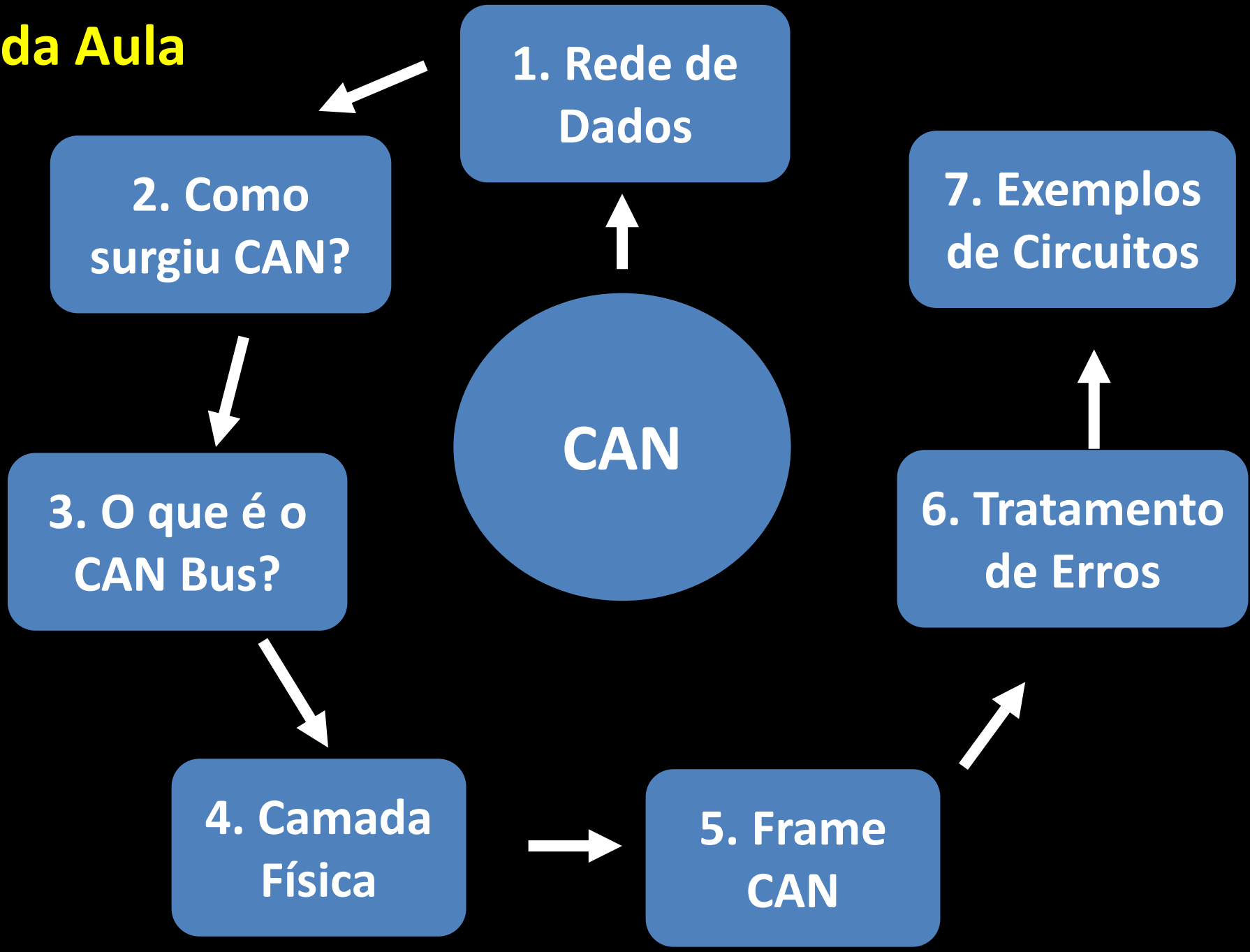
# Rede de Comunicação CAN
(**C**ontroller **A**rea **N**etwork)

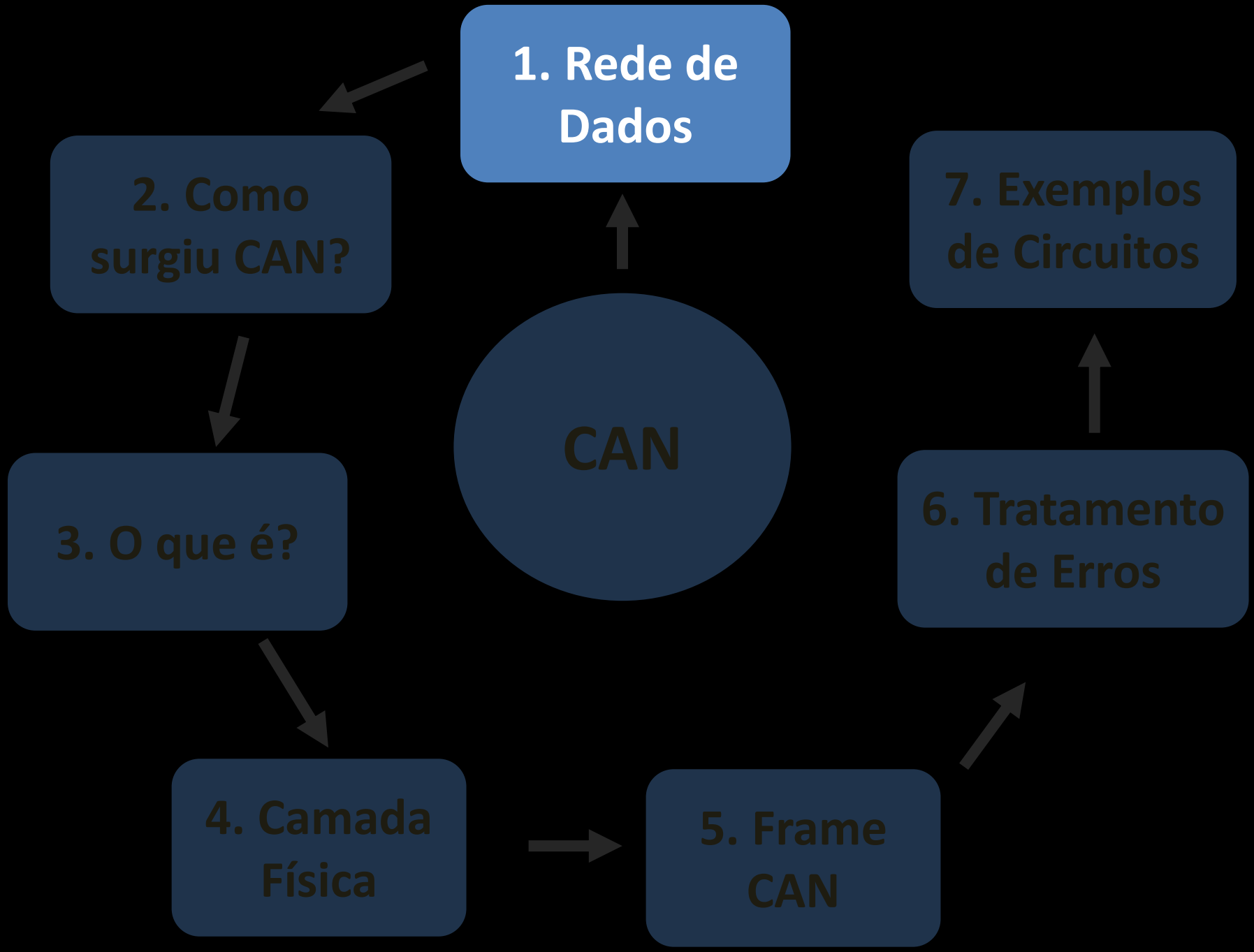
Prof. Leopoldo Yoshioka
Abril 2019

# Plano de Aula

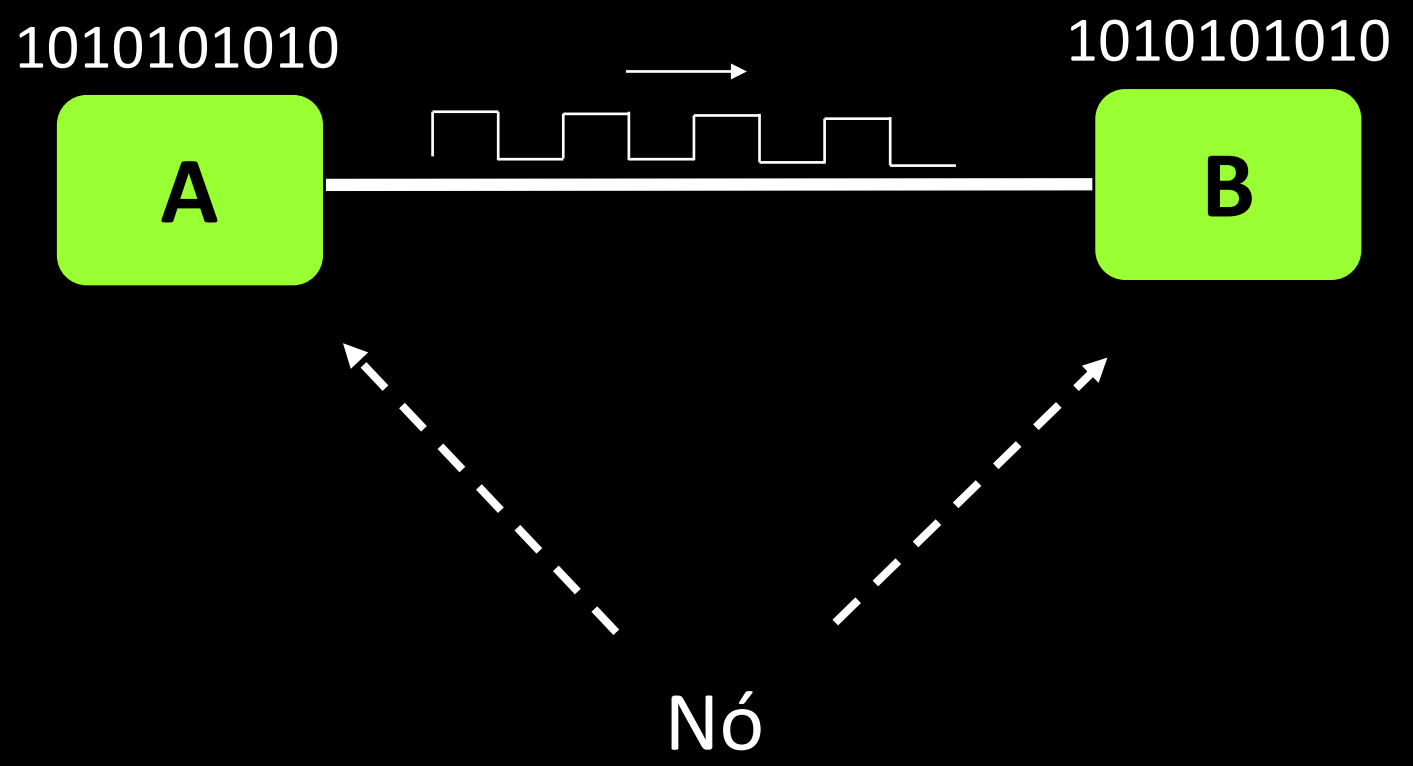
- Rede CAN
- Estudo de caso:  Audi A8
- Atividade prática no dinamômetro (IEE)

# Sumário da Aula

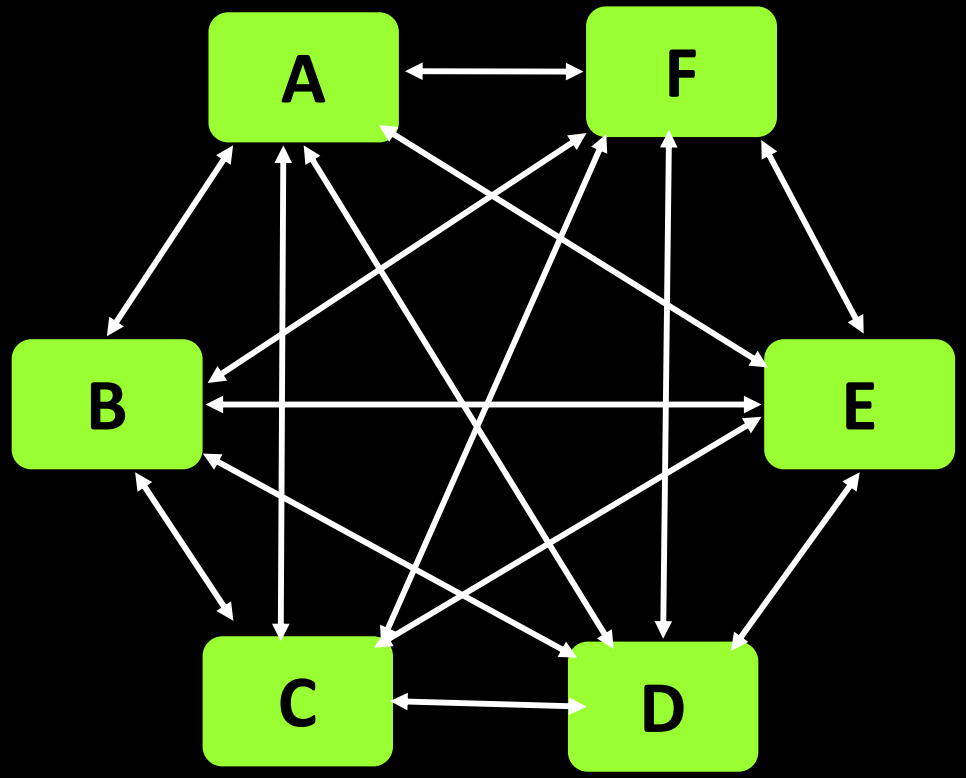



1. Rede de Dados
2. Como surgiu CAN?
3. O que é?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

# Comunicação de Dados – conceitos básicos

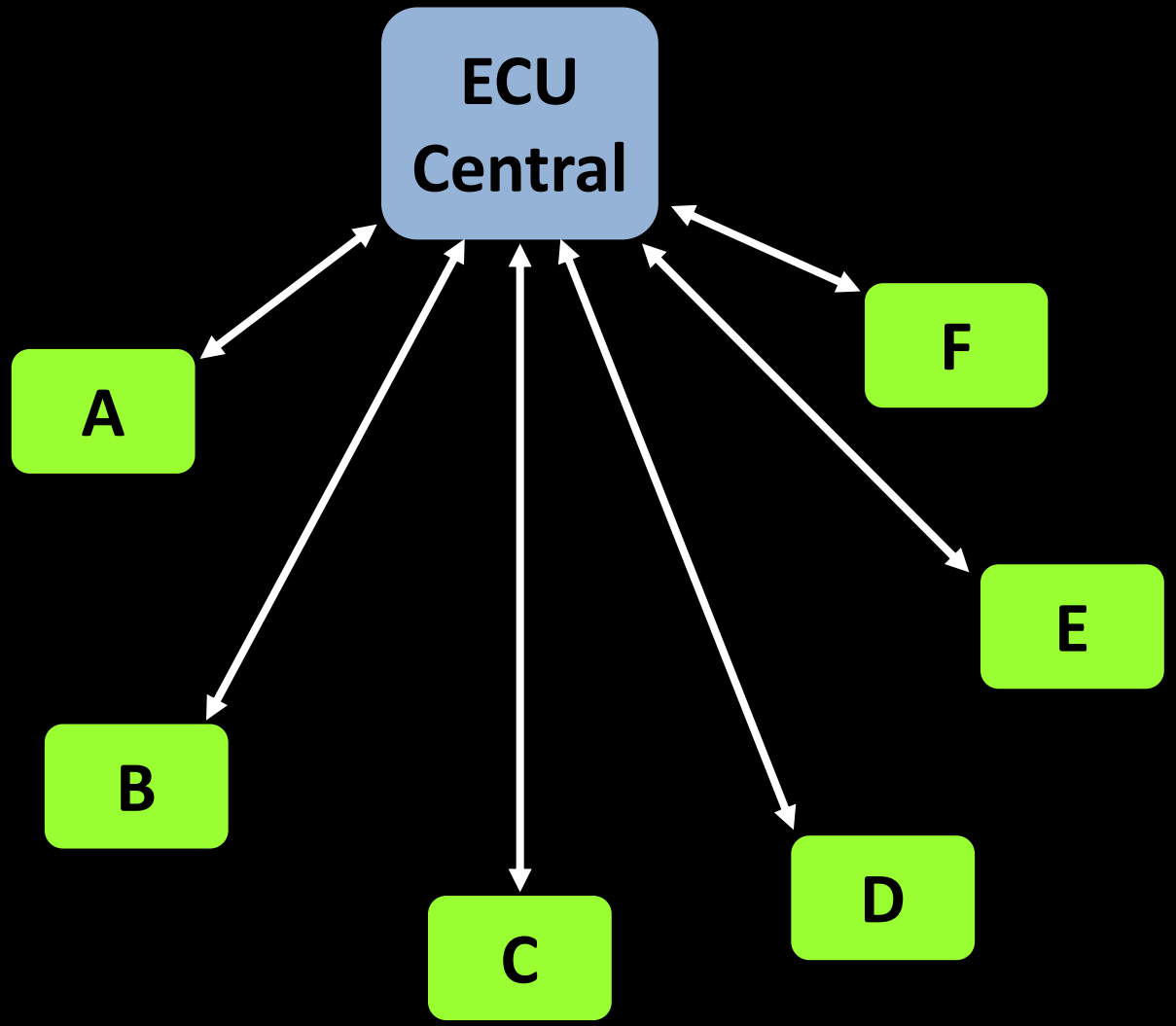


# 1 - Arquitetura Ponto-a-ponto (estrela)

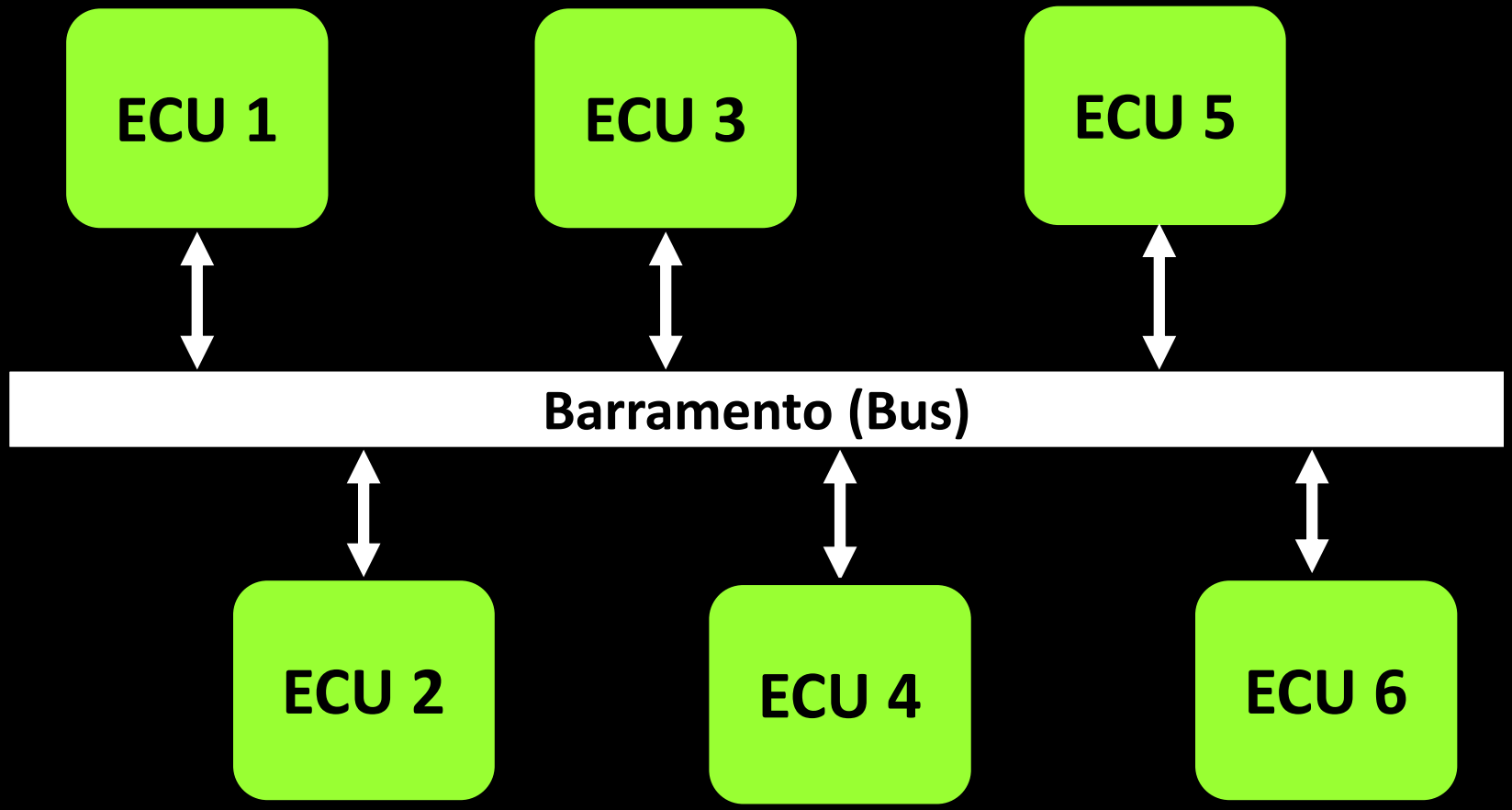



- 6 Nós
- 15 Conexões
- Estrela (ponto-a-ponto)

# 2 - Arquitetura Centralizada

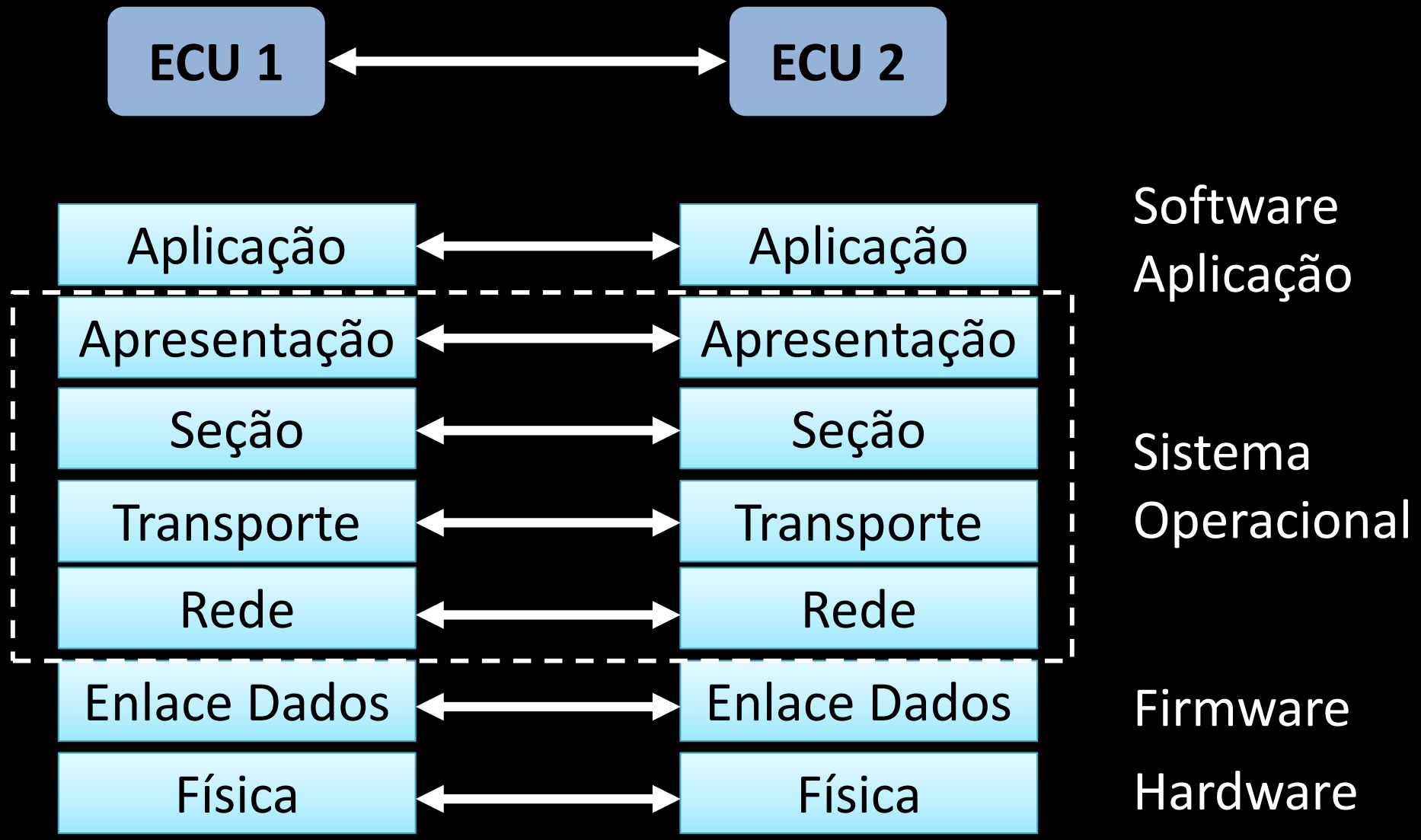


# Arquitetura Distribuída (barramento)



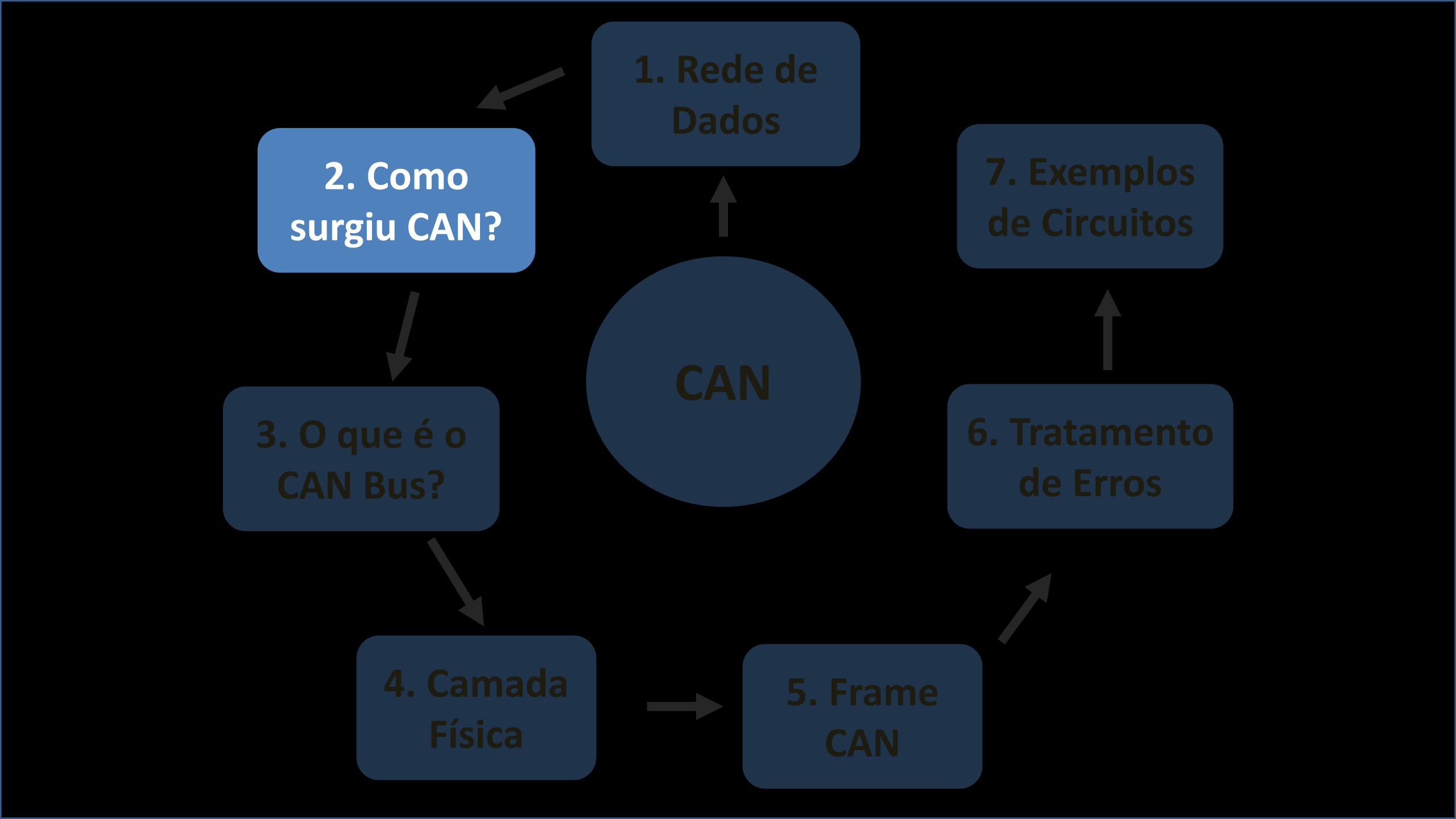

- Quem?
- Quando?
- Como?
- O que?

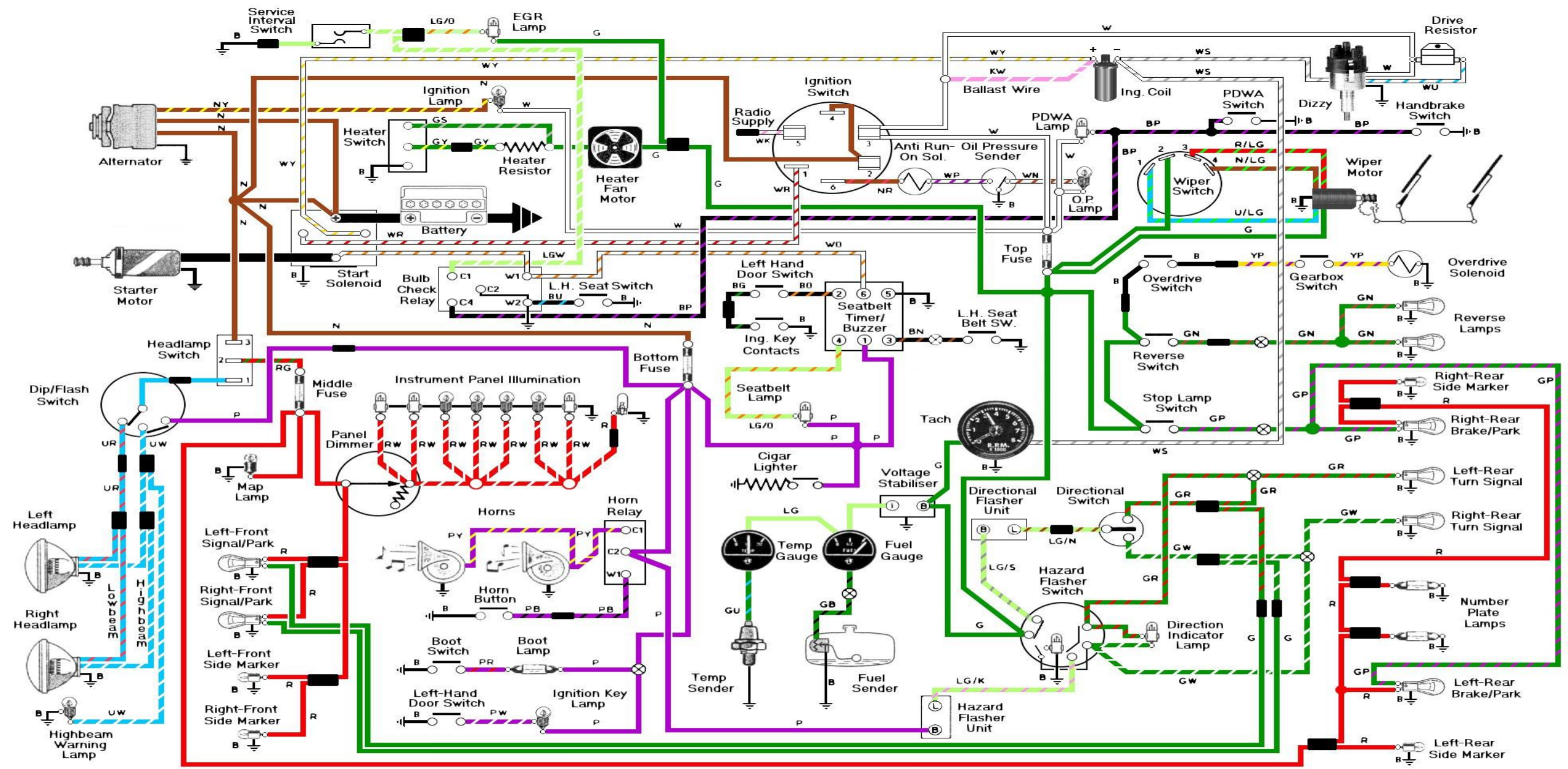
# Modelo OSI (Open System Interconnection)



# Descrição das Camadas OSI

| CAMADA | FUNÇÃO |
|---|---|
| 7 - Aplicação | Funções especialistas (transferência de arquivos, envio de e-mail, terminal virtual) |
| 6 - Apresentação | Formatação dos dados, conversão de códigos e caracteres |
| 5 - Sessão | Negociação e conexão com outros nós, analogia |
| 4 - Transporte | Oferece métodos para a entrega de dados ponto-a-ponto |
| 3 - Rede | Roteamento de pacotes em uma ou várias redes |
| 2 - Enlace | Detecção de erros |
| 1 - Física | Transmissão e recepção dos bits brutos através do meio físico de transmissão |


1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN




Note: The L.H. seat switch at the bulb check relay is shown "as wired" in this car. It is not functional.

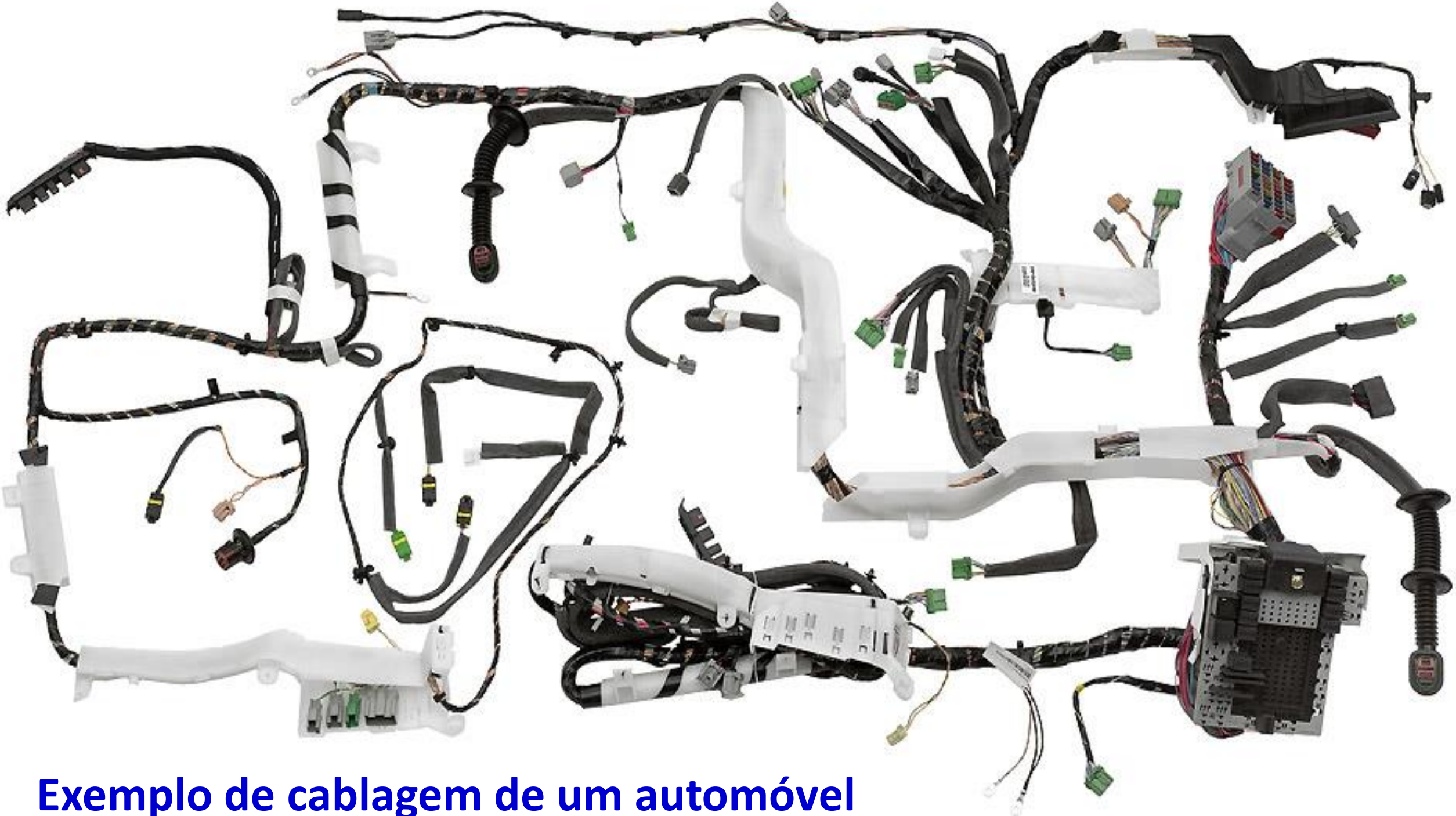
B=black
G=green
K=pink
LG=light green
N=brown
O=orange
P=purple
R=red
S=slate (gray)
U=blue
W=white
Y=yellow

*Spitprint 76*

*1976 Triumph Spitfire, Federal Version*

*"Bela" 11/21/98 tomomalley@meganet.net*

*FM50855U AUG 1976*



**Exemplo de cablagem de um automóvel**

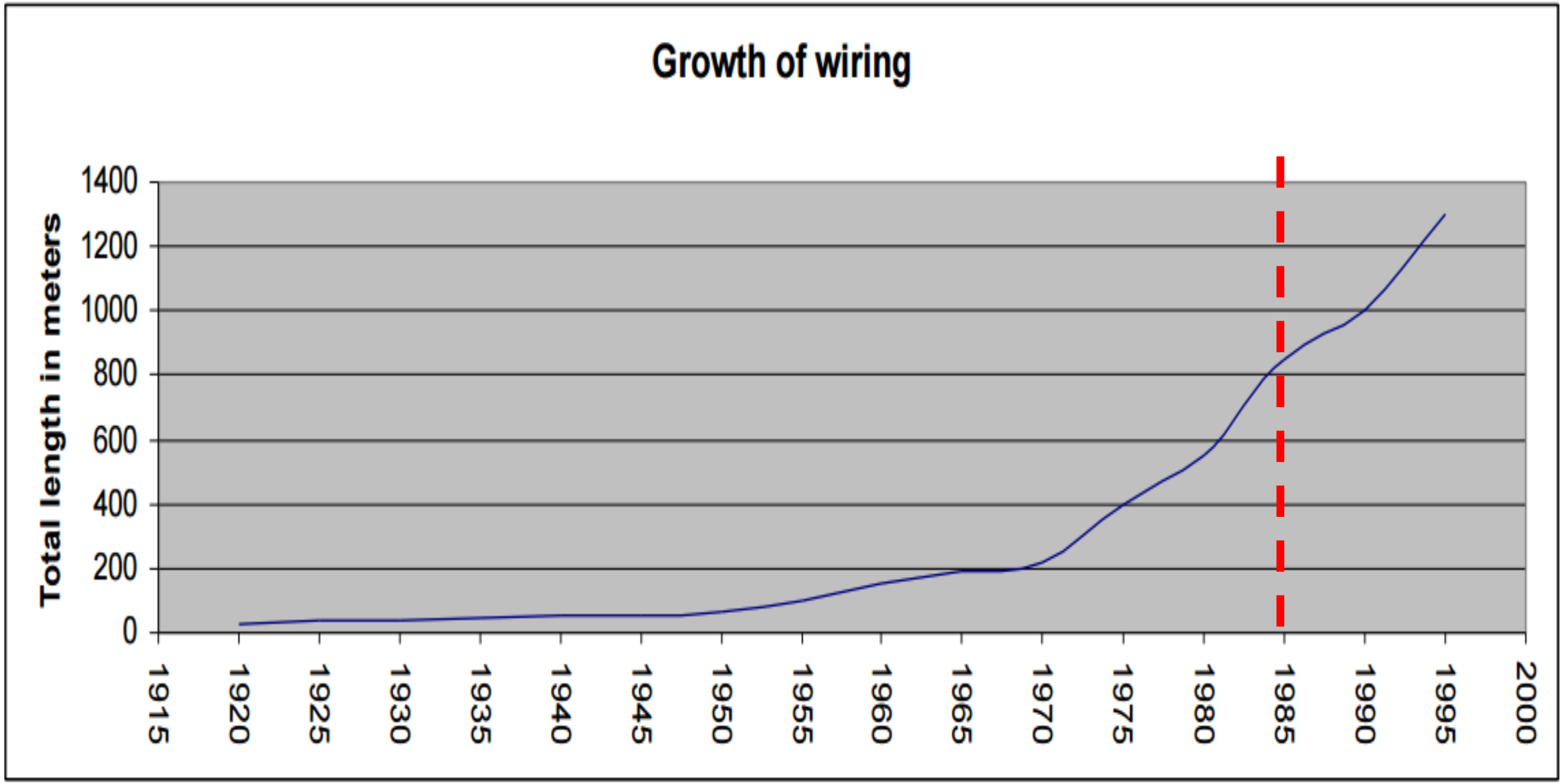



**Figure 1 Growth of Automotive Wiring**

# Histórico

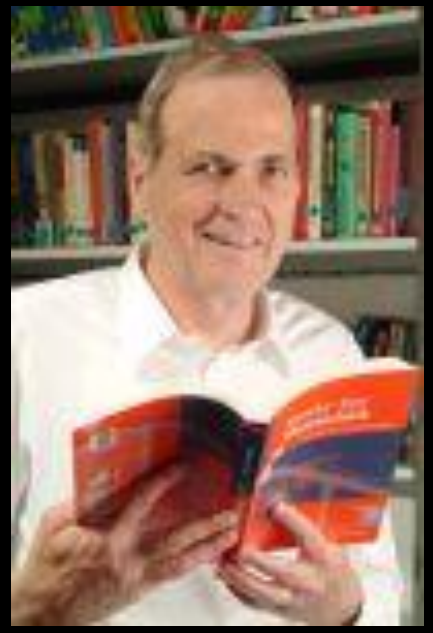
- No início da década de 1980 não existia um sistema de comunicação específico para interconectar dispositivos eletrônicos de um automóvel.

- Engenheiros da Bosch procuravam uma rede que pudesse ser utilizado em automóveis, mas perceberam que não existia nada que atendesse às necessidades.

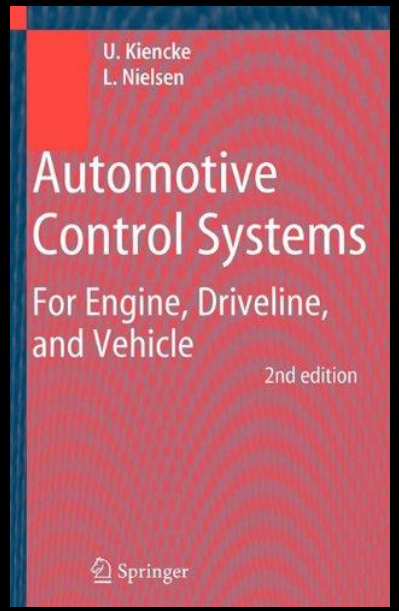
- Em 1983, Uwe Kiencke iniciou o desenvolvimento de um novo barramento serial.

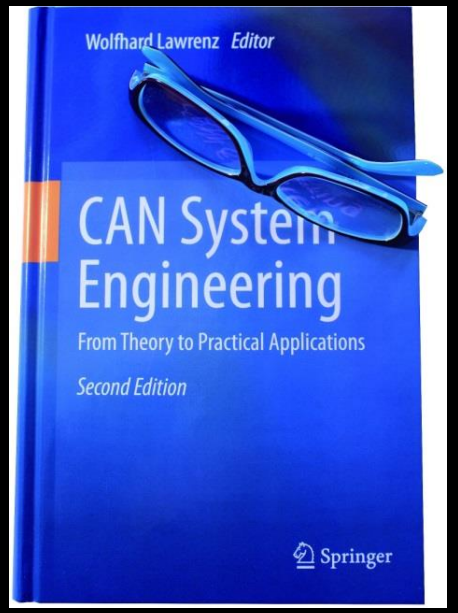
- O professor Wolfhard Lawrenz, Univesity of Applied Science in Braunschweig-Wolfenbuttel foi contratado como consultor. Ele propôs um novo protocolo de rede chamado “Controller Area Network”.

Prof. Uwe Kiencke



Prof. Wolfhard Lawrenz



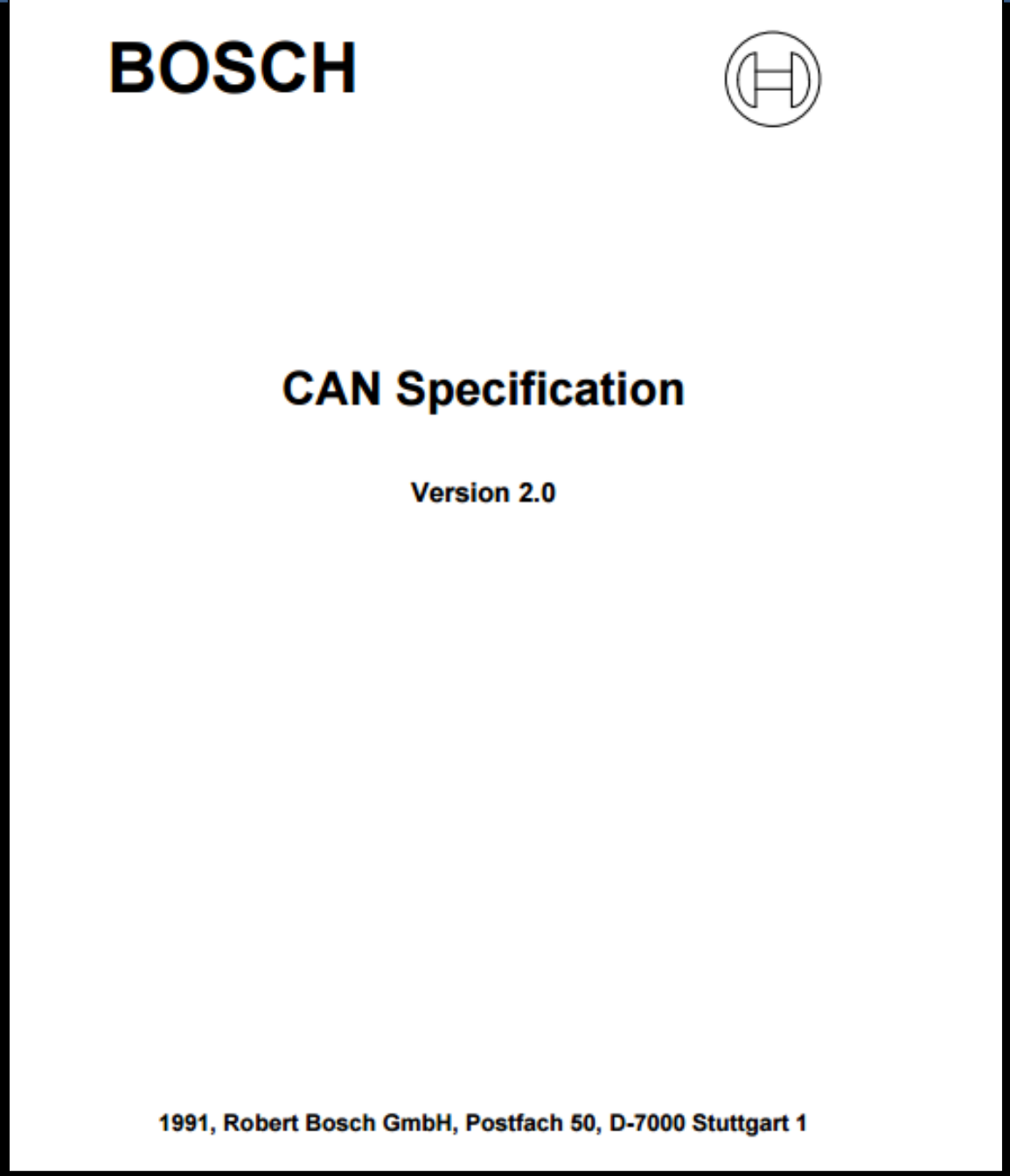



- **Em fevereiro de 1986 foi anunciado no Congresso SAE em Detroit**


http://papers.sae.org/860391/

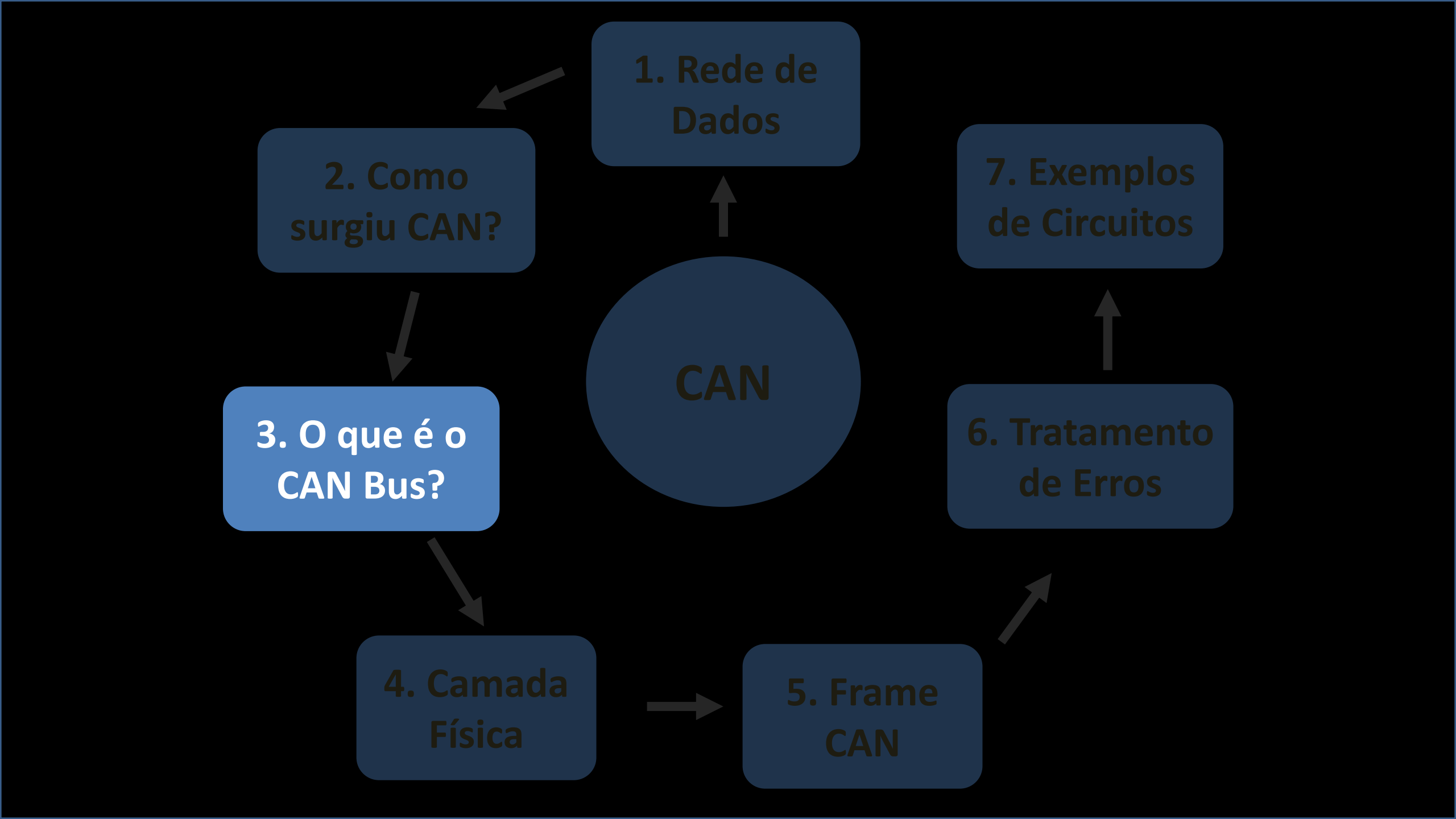

Primeiros documentos publicados pela Bosch descrevendo o protocolo CAN (ver 1.0)

| | |
|---|---|
| **1983:** | Start of the Bosch internal project to develop an in-vehicle network |
| **1986:** | Official introduction of CAN protocol |
| **1987:** | First CAN controller chips from Intel and Philips Semiconductors |
| **1991:** | Bosch's CAN specification 2.0 published |
| **1991:** | CAN Kingdom CAN-based higher-layer protocol introduced by Kvaser |
| **1992:** | CAN in Automation (CiA) international users and manufacturers group established |
| **1992:** | CAN Application Layer (CAL) protocol published by CiA |
| **1992:** | First cars from Mercedes-Benz used CAN network |
| **1993:** | ISO 11898 standard published |
| **1994:** | 1st international CAN Conference (iCC) organized by CiA |
| **1994:** | DeviceNet protocol introduction by Allen-Bradley |
| **1995:** | ISO 11898 amendment (extended frame format) published |
| **1995:** | CANopen protocol published by CiA |
| **2000:** | Development of the time-triggered communication protocol for CAN (TTCAN) |




- Documento de 72 pag.
- Parte A (11 bits)
- Parte B (29 bits)

http://www.kvaser.com/software/7330130980914/V1/can2spec.pdf


1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

# CAN BUS – visão geral (1/3)

- É um padrão de comunicação automotivo que permite que as ECUs se comuniquem umas com as outas, sem a necessidade de um computador mestre.

- É uma comunicação do tipo “broadcast”. Todos os nós podem podem “escutar” todas transmissões.

- Não é possível um determinado nó enviar uma mensagem para outro nó específico. Todos os nós, invariavelmente, captarão todo tráfego.

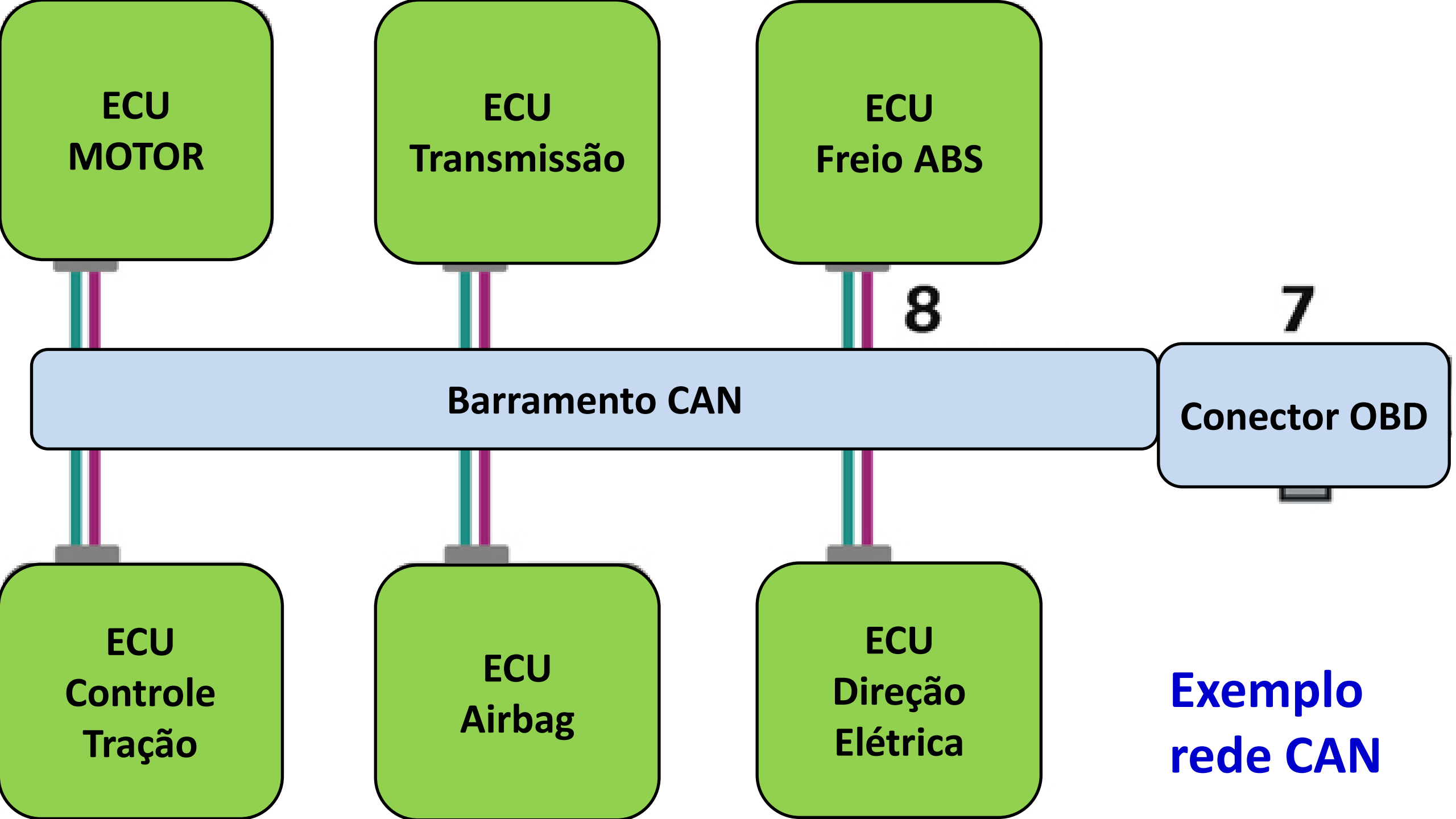
- Entretanto, o hardware CAN, filtrará localmente as mensagens de modo que cada nó reagirá somente às mensagens de seu interesse.

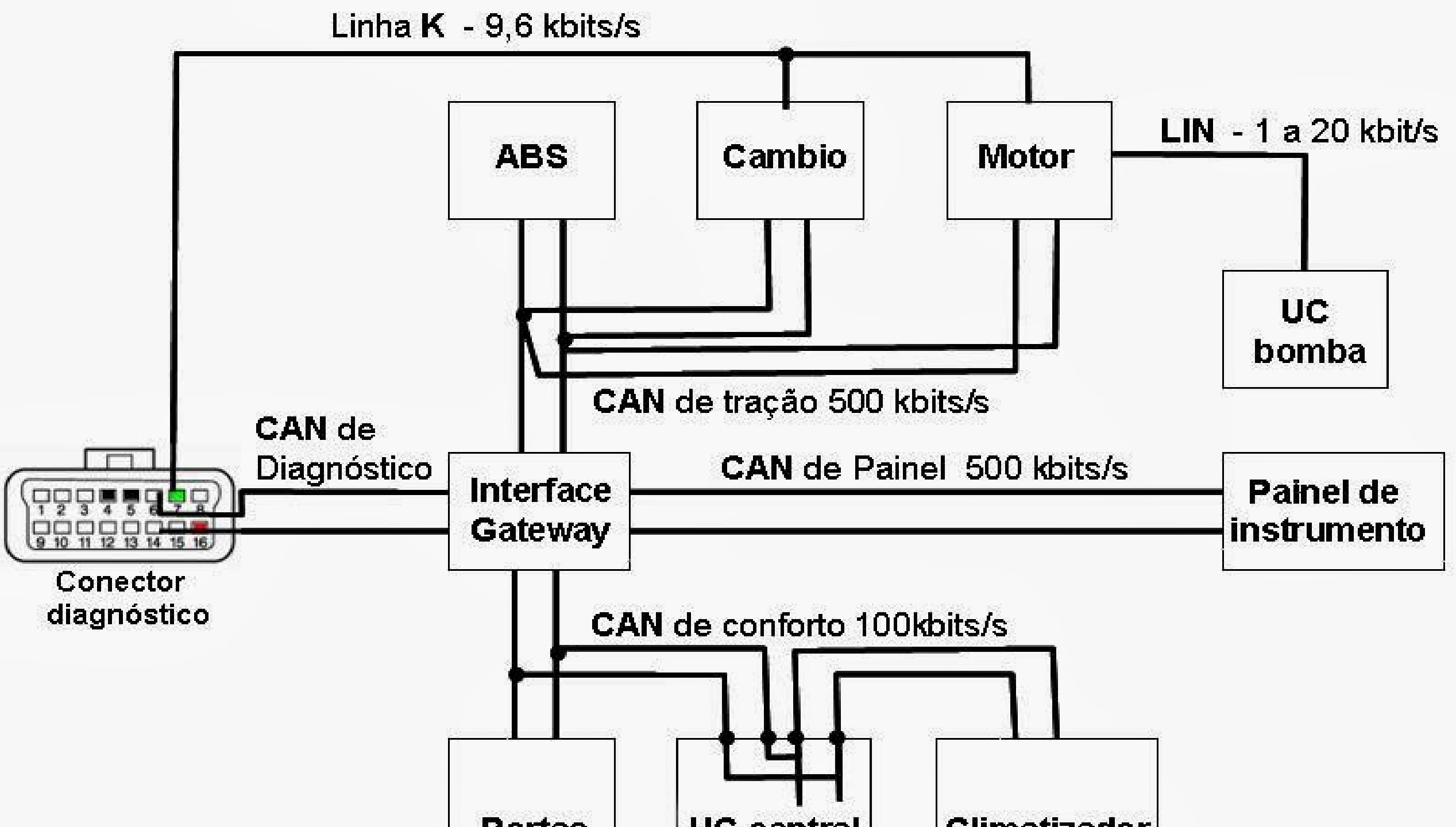
# CAN BUS – visão geral (2/3)

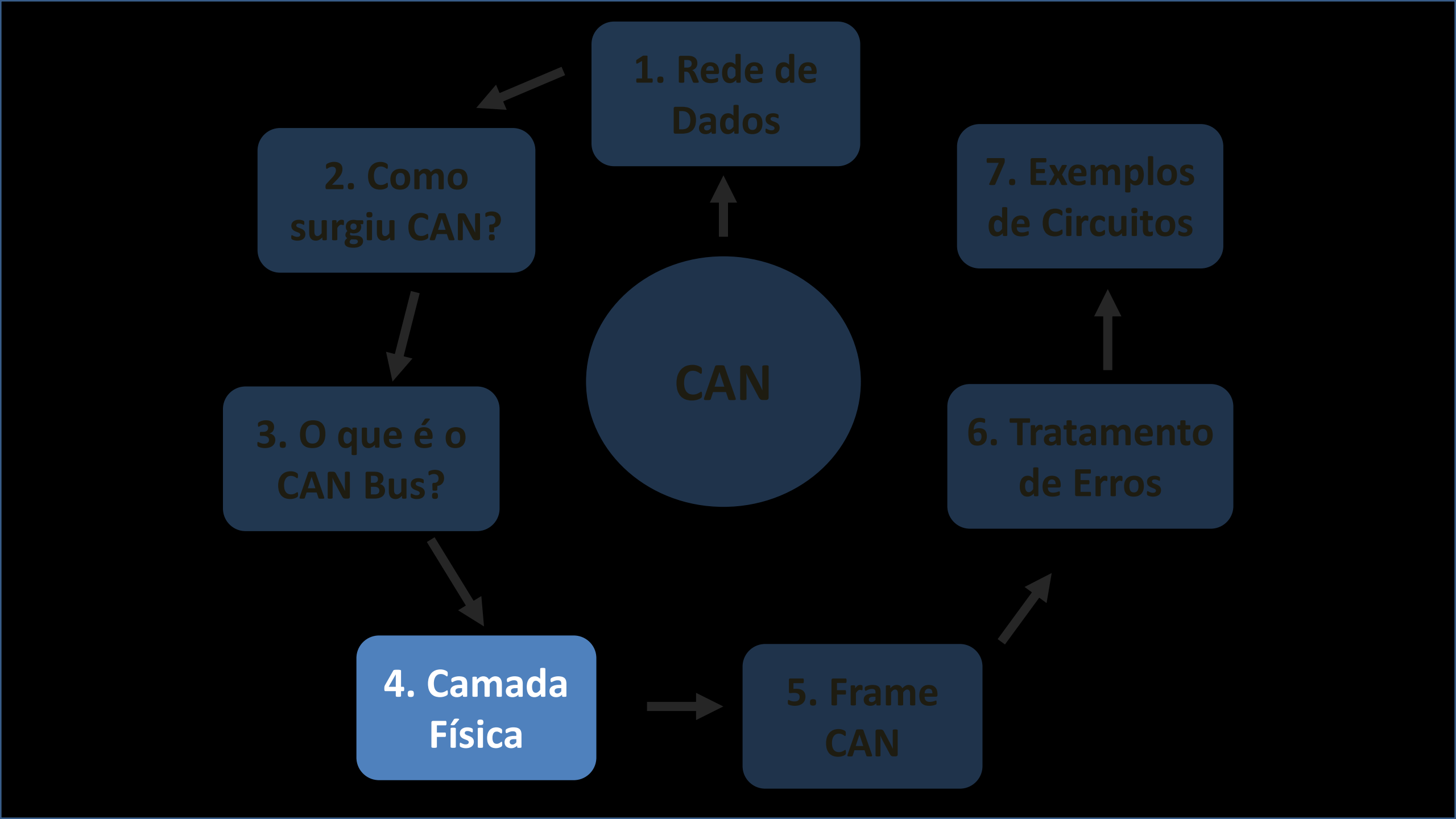
- É um barramento constituído por uma rede de dois fios com uma taxa de transmissão de até 1 Mbps.
- Através do barramento CAN são conectados vários componentes eletrônicos tais como: ECU de motor, microcontroladores, disposivos, sensores, atuadores entre outras distribuído pelo veículo.
- O CAN utiliza um protocolo baseado em mensagem.
- Mecanismo de **arbitragem** não-destrutiva – garantia de acesso ao barramento àquela mensagem de prioridade mais elevada, sem atraso

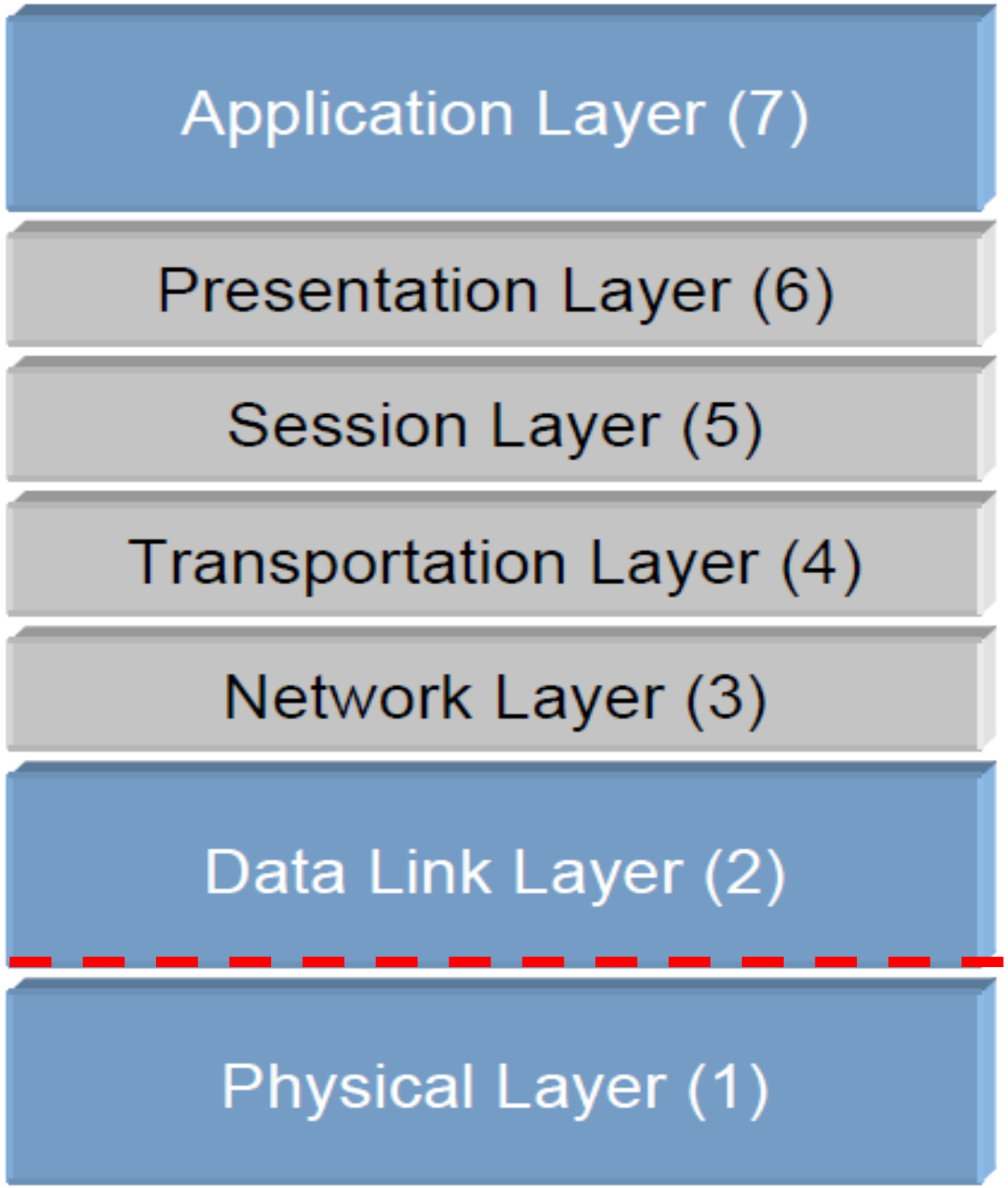
# CAN BUS – visão geral (3/3)

- Vários mecanismos de detecção de erros: Desconexão automática de nós com falhas a fim de manter a comunicação entre os demais nós.
- As mensagens não são identificados pelos endereços dos nós do transmissor ou do receptor, como é na maioria dos sistemas de transmissão de dados.
- A identificação é feita pelo conteúdo da mensagem. O identificador tem também a função de especificar a prioridade da mensagem dentro do sistema.


ECU
MOTOR
ECU
Transmissão
ECU
Freio ABS
8
7
Barramento CAN
Conector OBD
ECU
Controle
Tração
ECU
Airbag
ECU
Direção
Elétrica
Exemplo
rede CAN


Linha K - 9,6 kbits/s
ABS
Cambio
Motor
LIN - 1 a 20 kbit/s
UC bomba
CAN de tração 500 kbits/s
CAN de Diagnóstico
Interface Gateway
CAN de Painel 500 kbits/s
Painel de instrumento
Conector diagnóstico
CAN de conforto 100kbits/s


1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

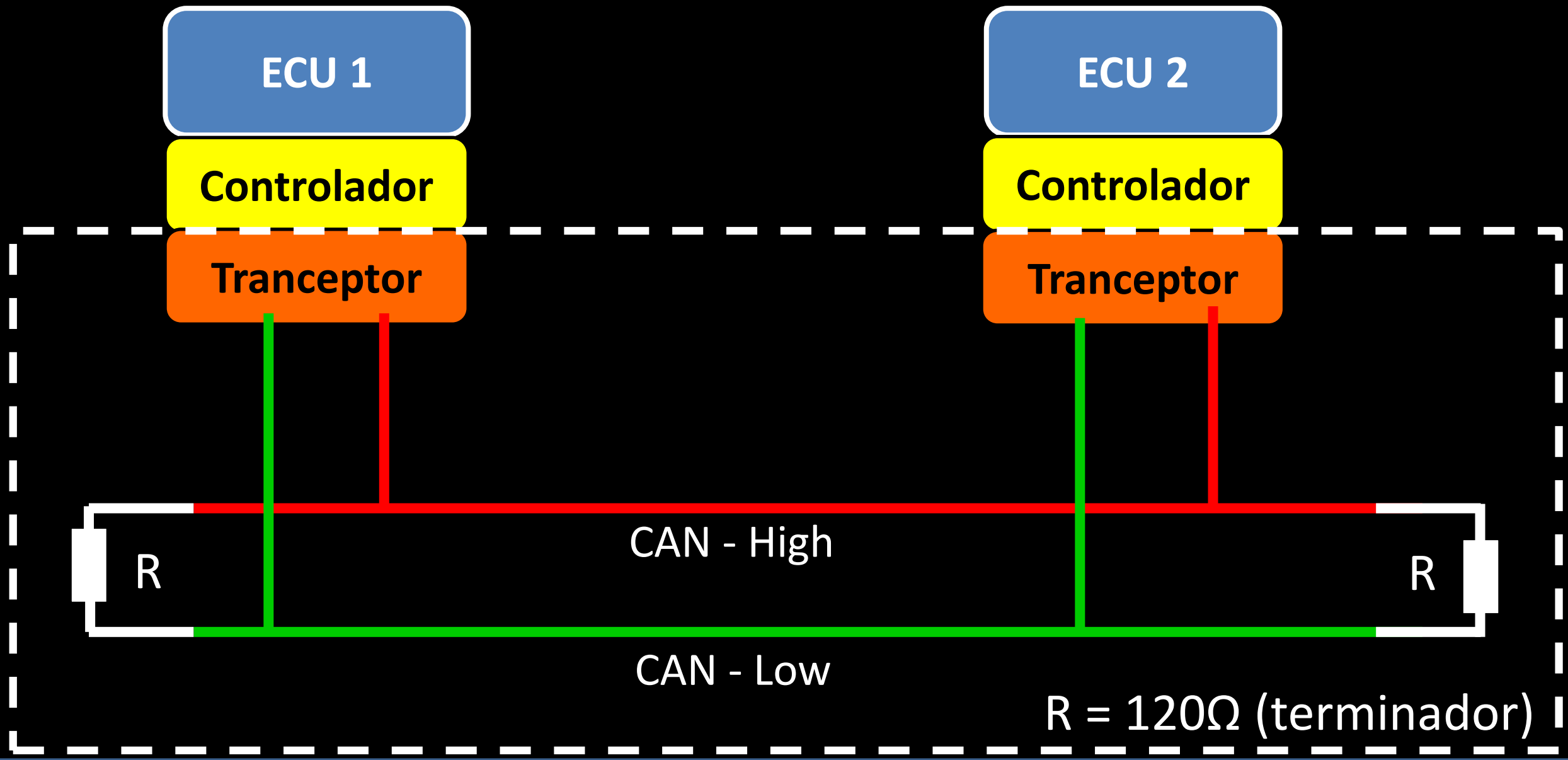



Layer 7:
Execution of the field bus tasks

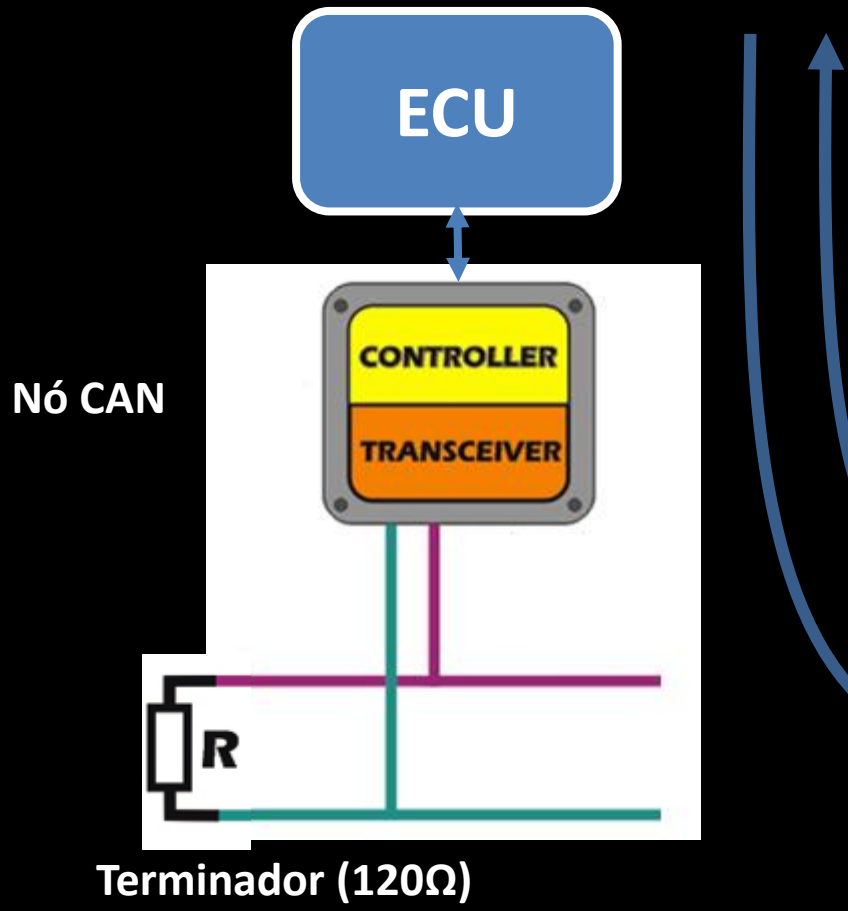
Layer 2:
Bus access, frame format and testing, addressing

Layer 1:
Definition of transmission medium and plug-and-socket connection, level, coding, bit rate

# Camada Física



# Componentes:




**CAN Controller** - é a interface lógica entre a ECU e o CAN Tranceiver. Processa os dados de transmissão (**Tx**) como também os dados de recepção(**Rx**)

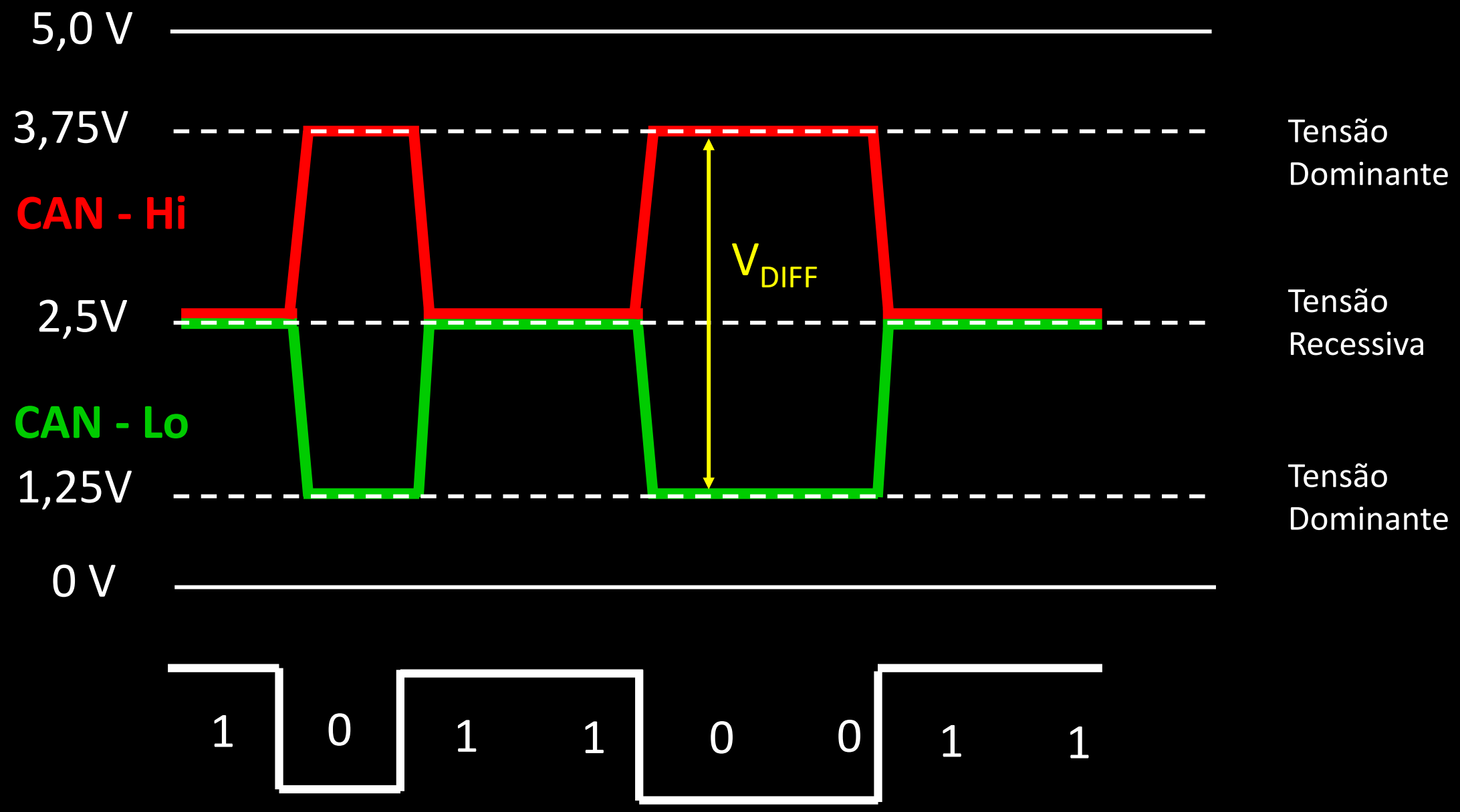
**CAN Transceiver** - é a interface elétrica entre o CAN Controller e o barramento CAN. Converte o dado em sinal elétrico e vice-versa.

**Rx**

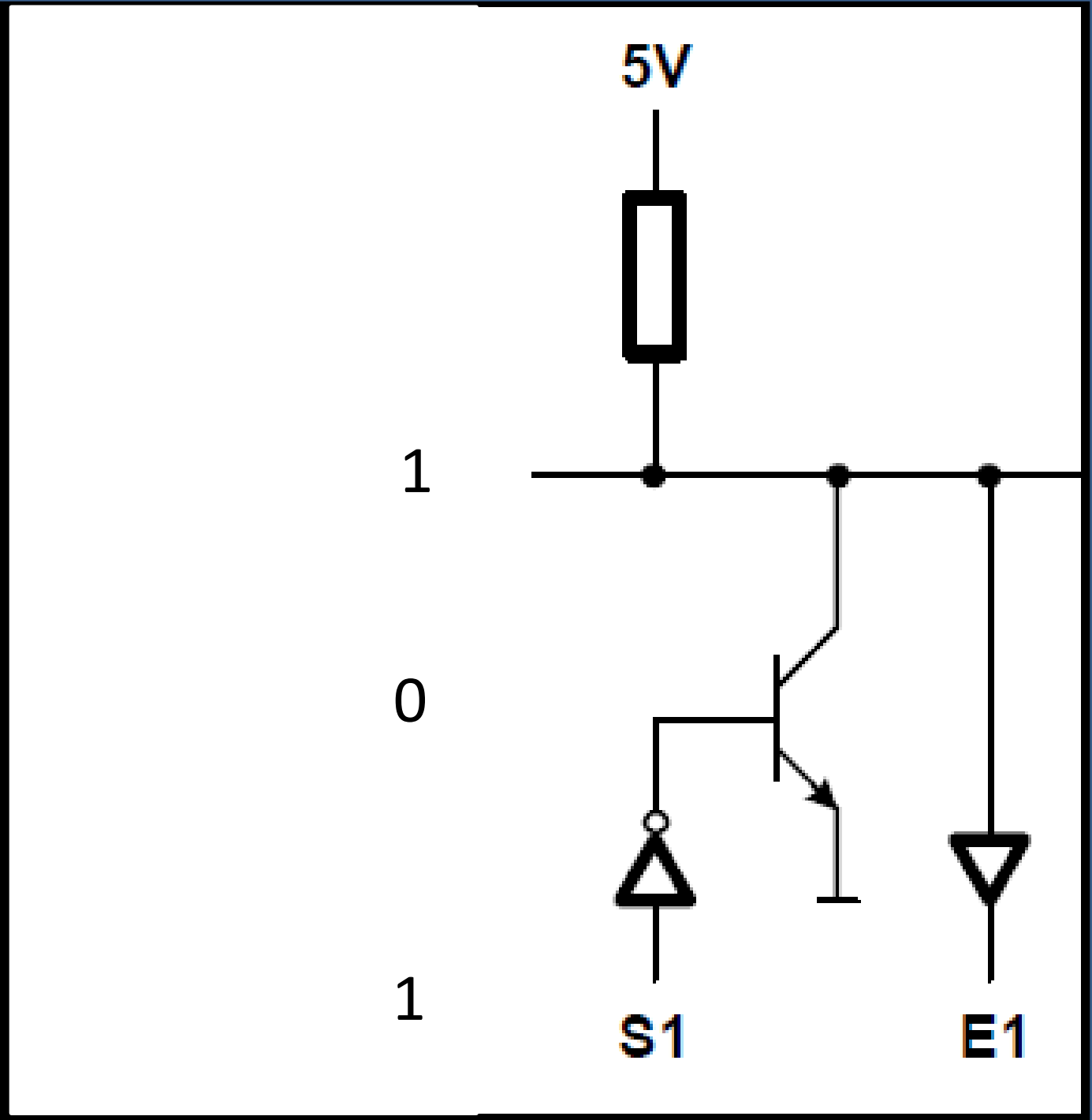
**Tx**

**Terminador** - é um resistor, tipicamente de 120Ω. Evita que o dado enviado seja refletido, provocando um eco.

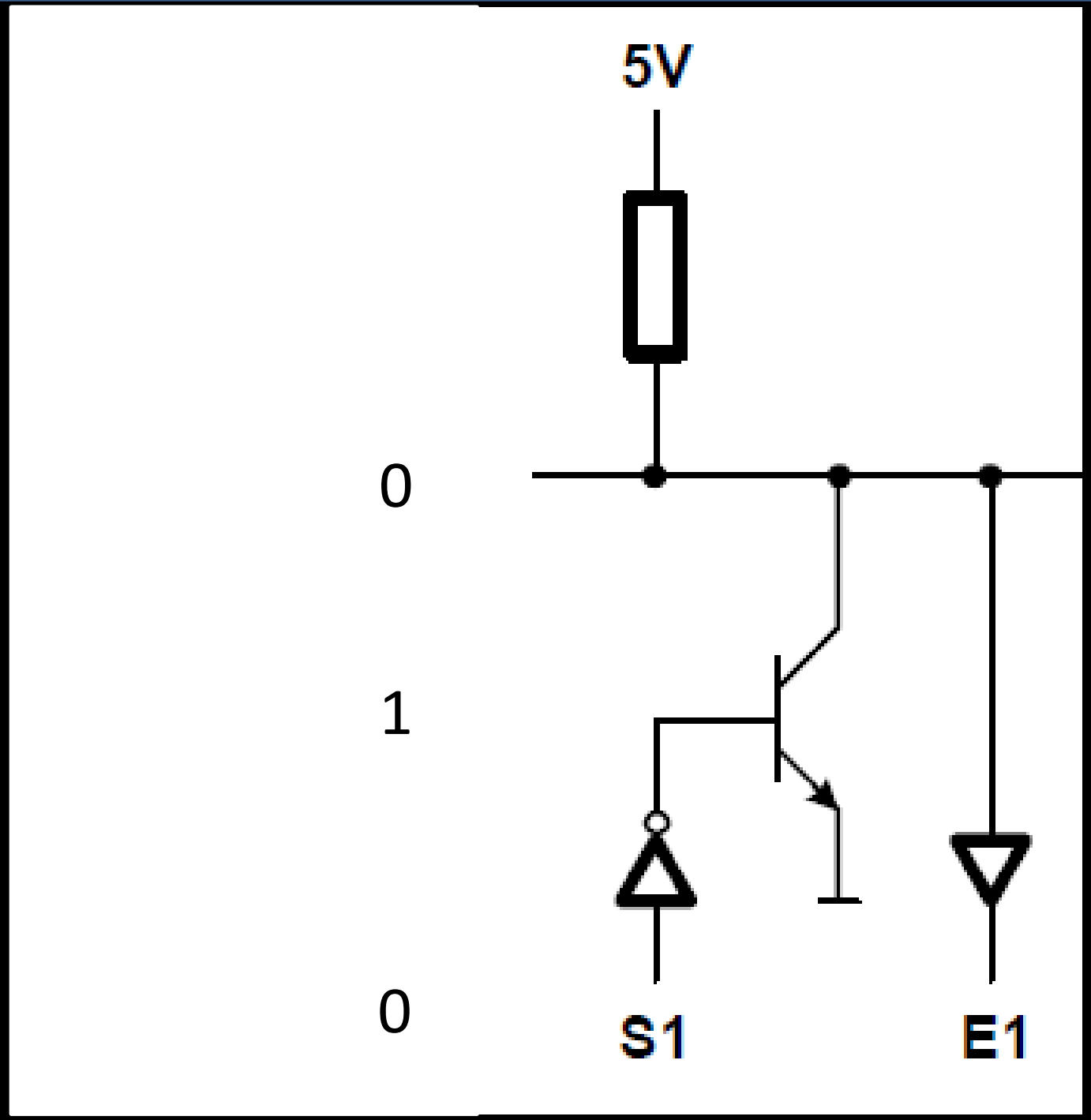
# Níveis de tensão



# Circuito do Driver



# Circuito do Driver



| | level | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| S1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| S2 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| S3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| Bus | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

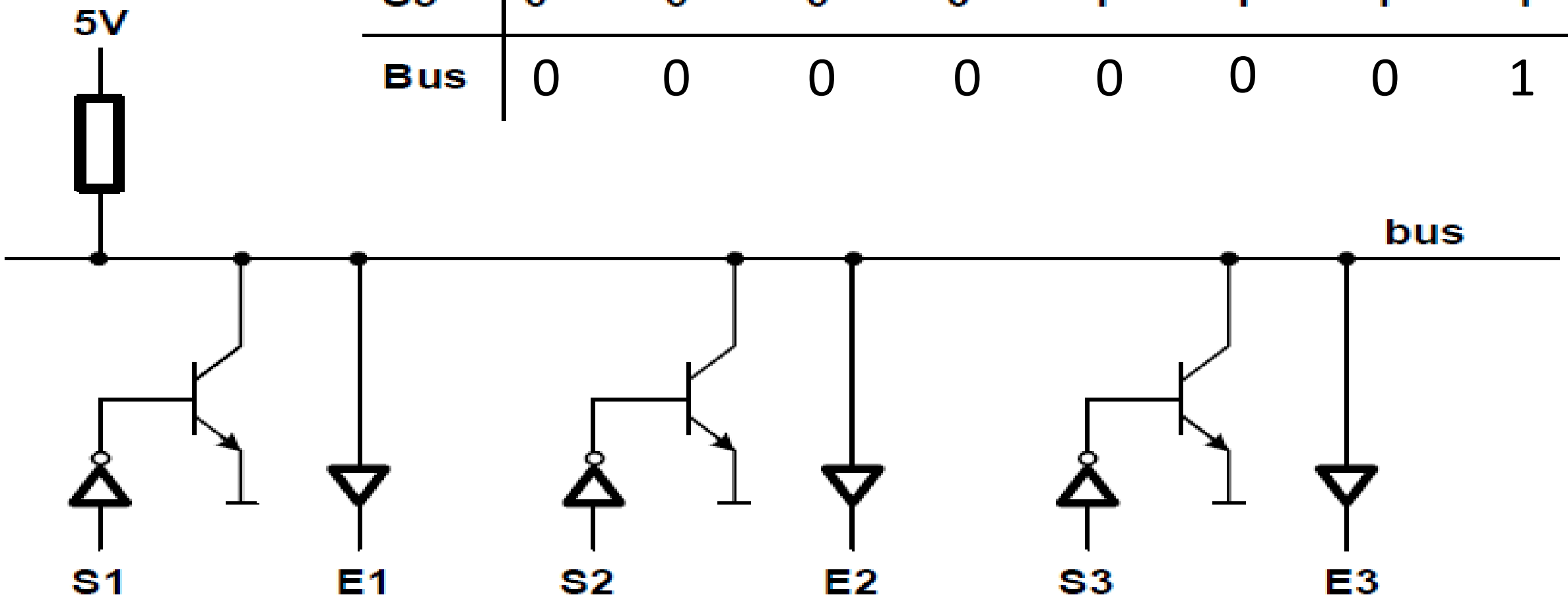
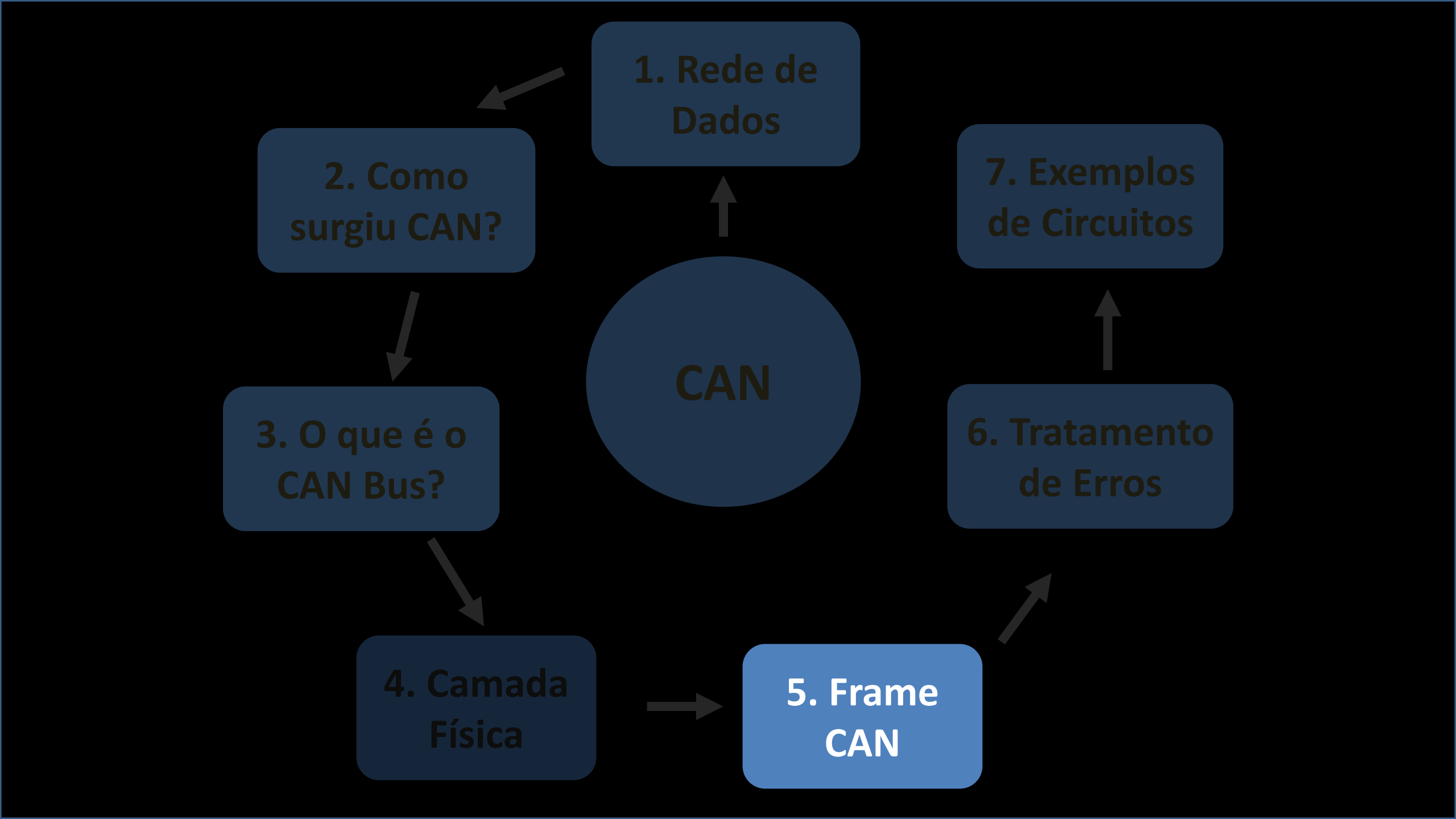
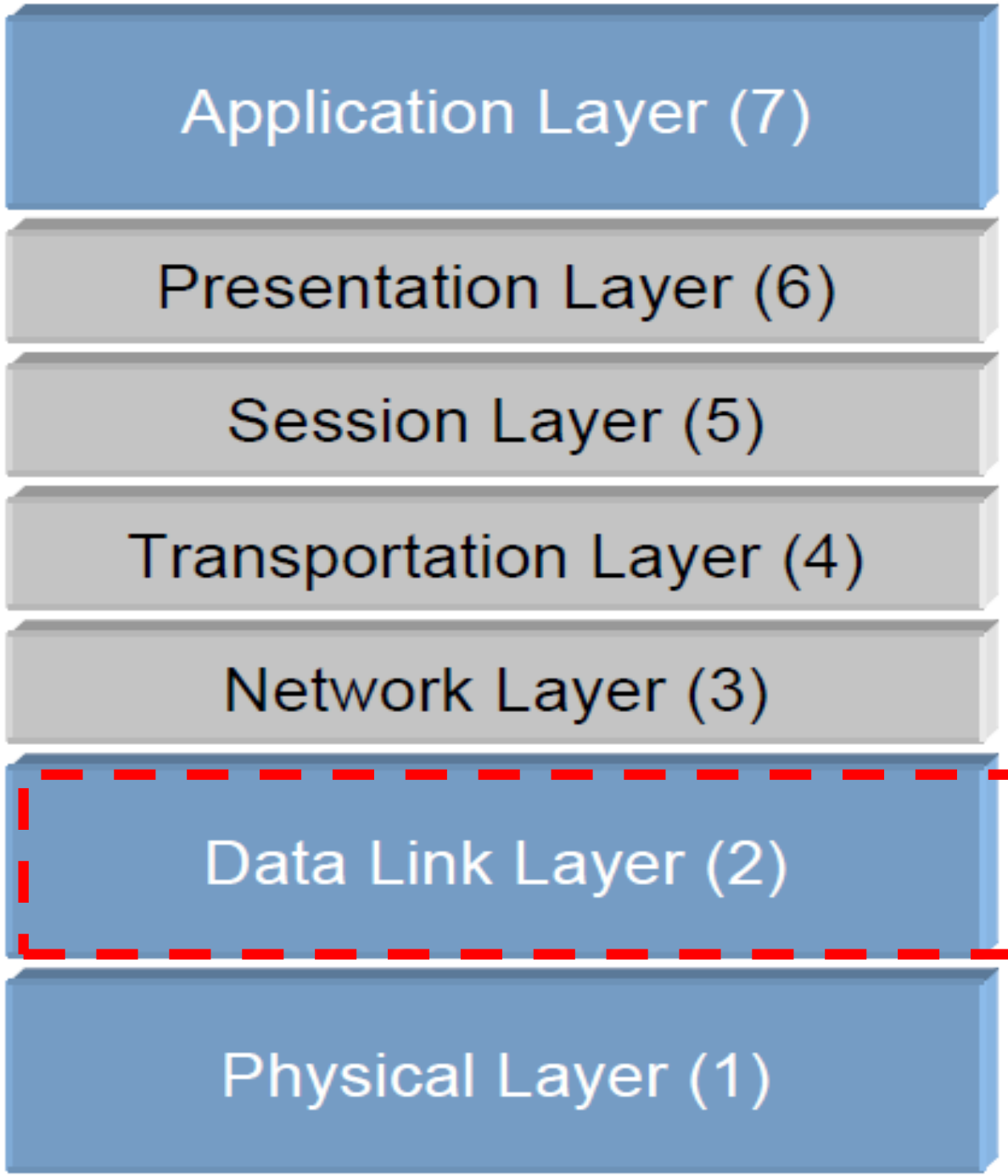



*Figure 1: wired-and Circuit*


1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

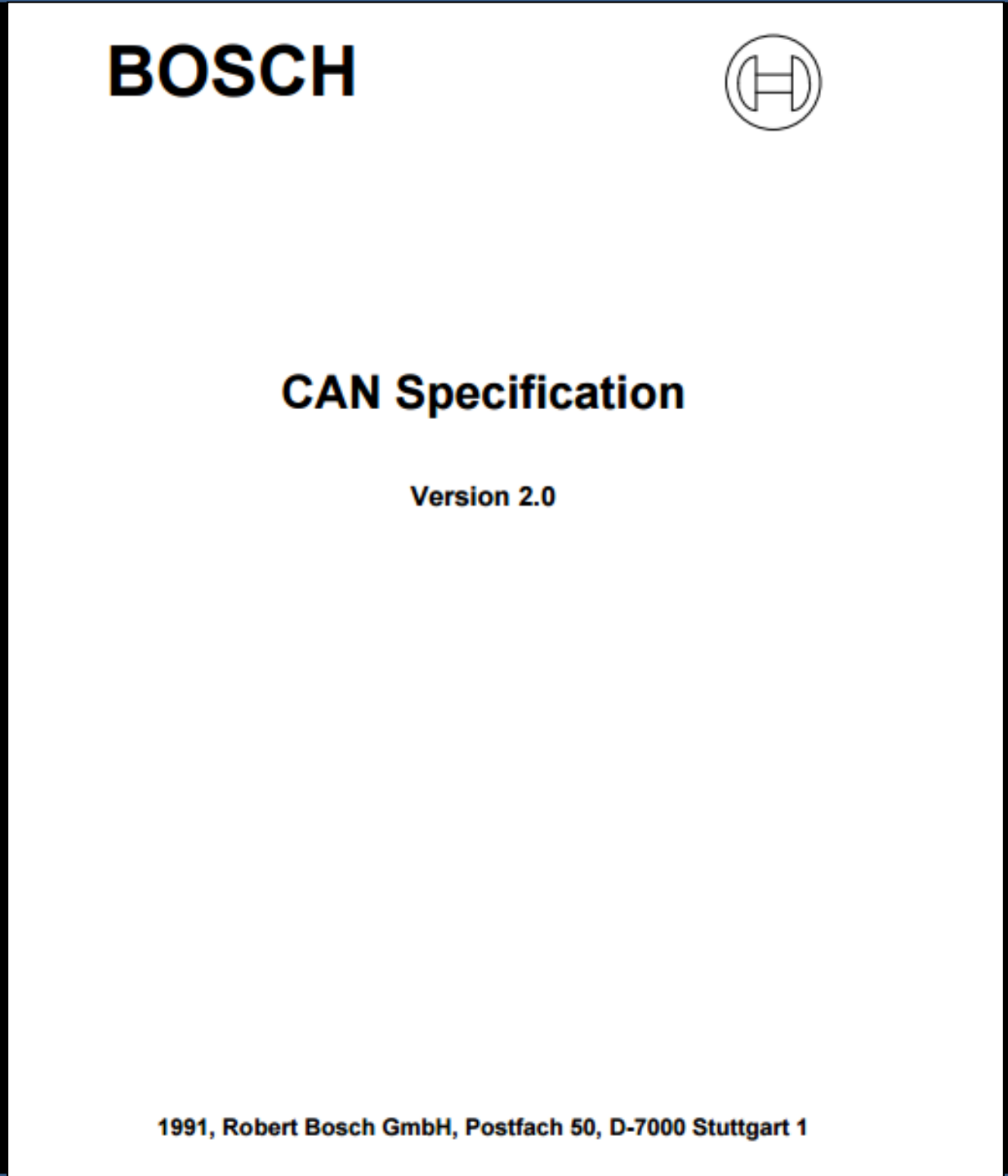



Layer 7:
Execution of the field bus tasks

Layer 2:
Bus access, frame format and testing, addressing

Layer 1:
Definition of transmission medium and plug-and-socket connection, level, coding, bit rate

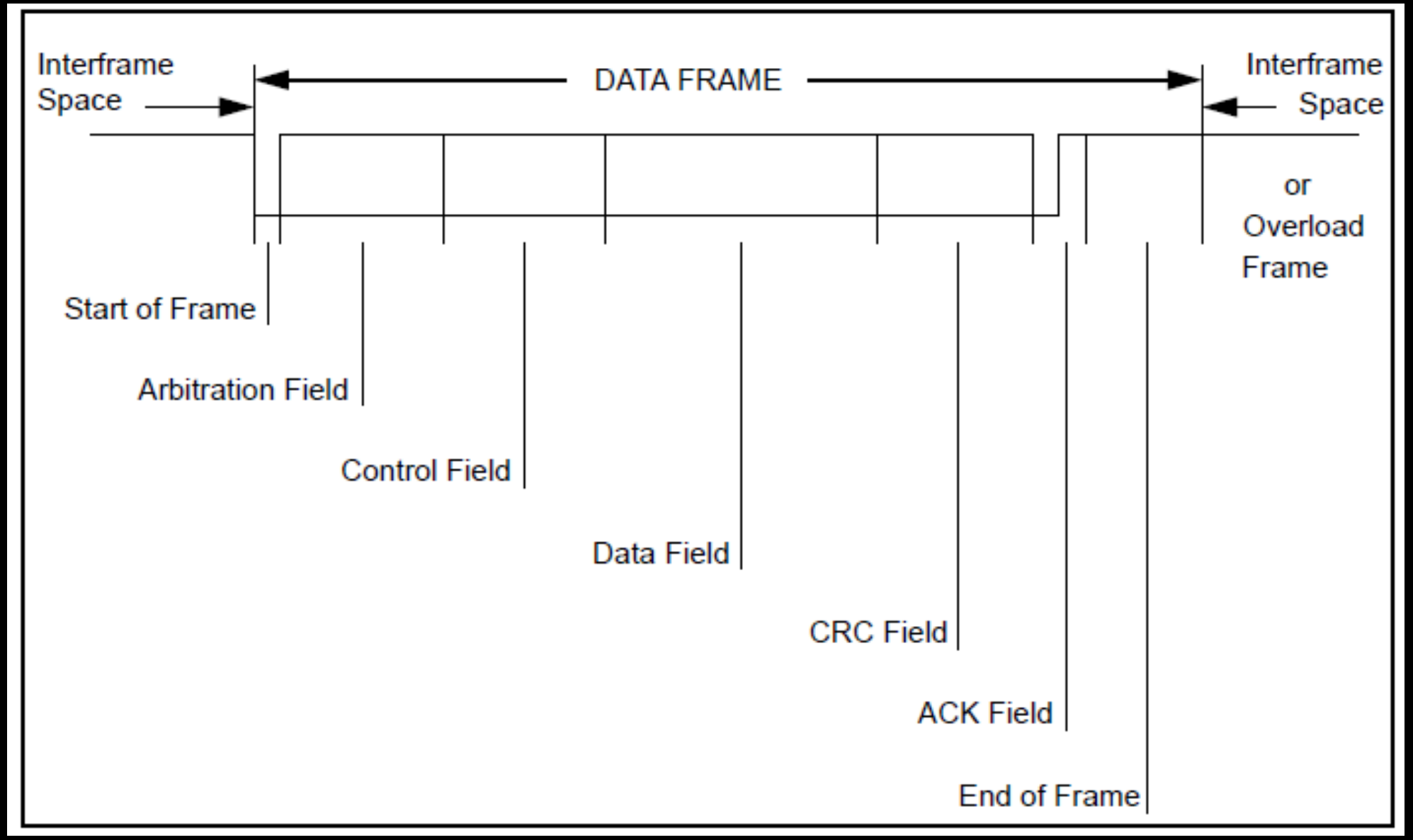
# Onde encontrar informações?



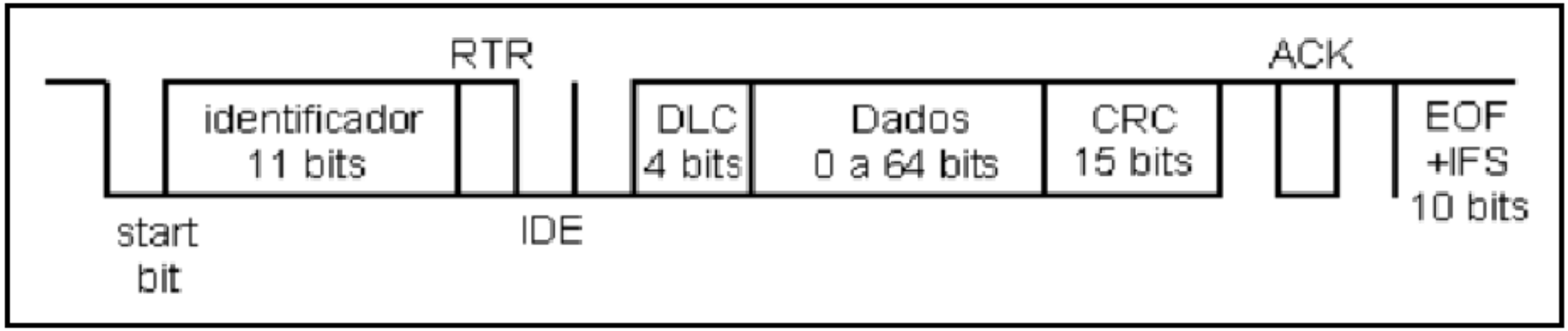
# Princípio de Operação da Rede CAN

- **Baseado em troca de mensagens**
- **Dois tipos de mensagens:**
  - Dados (Data Frame)
  - Requisição (**R**emote **T**ransmission **R**equest)

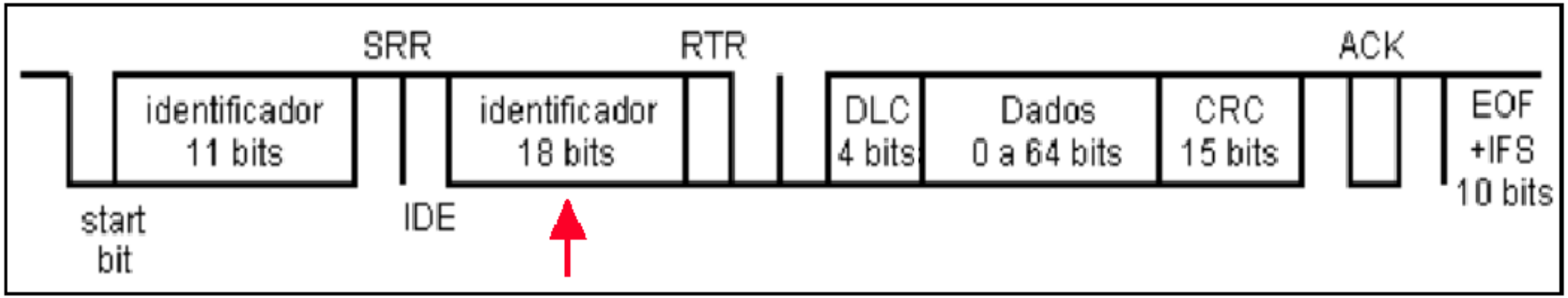
# Data Frame


Interframe Space
DATA FRAME
Interframe Space
or Overload Frame
Start of Frame
Arbitration Field
Control Field
Data Field
CRC Field
ACK Field
End of Frame

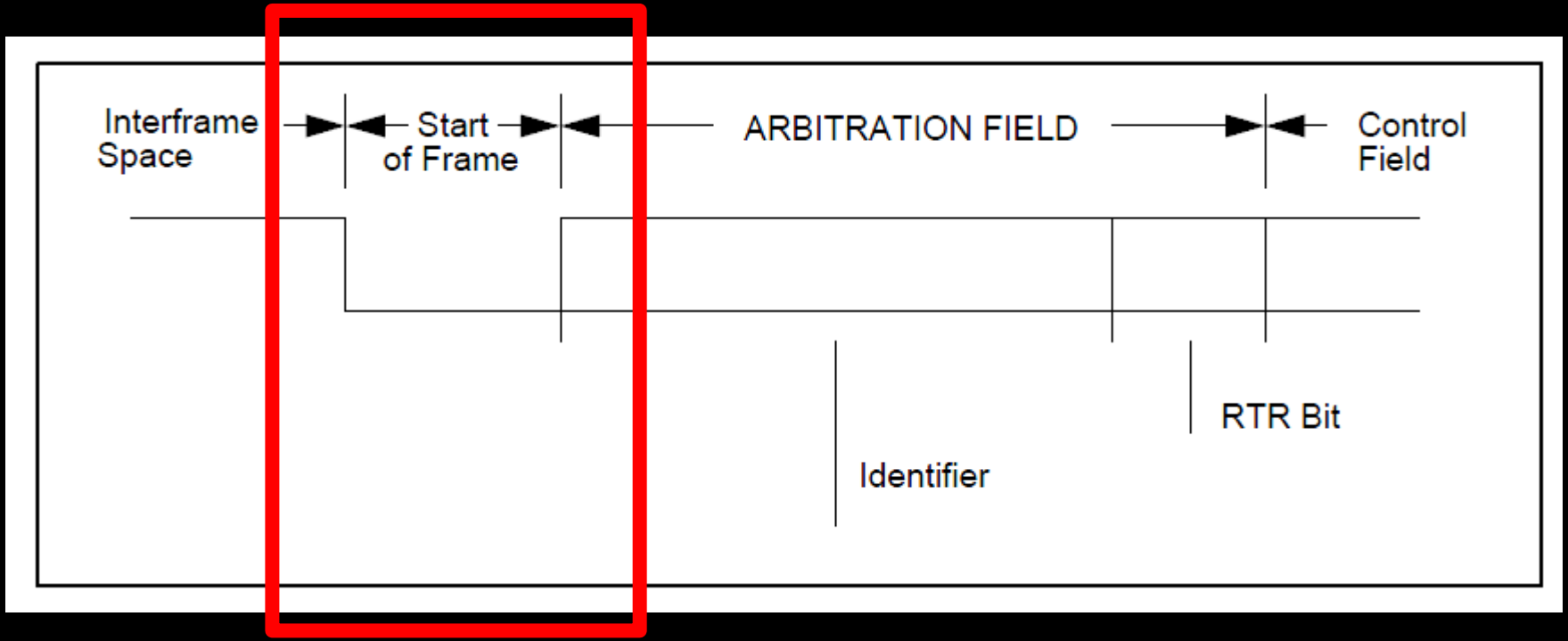
# Data Frame



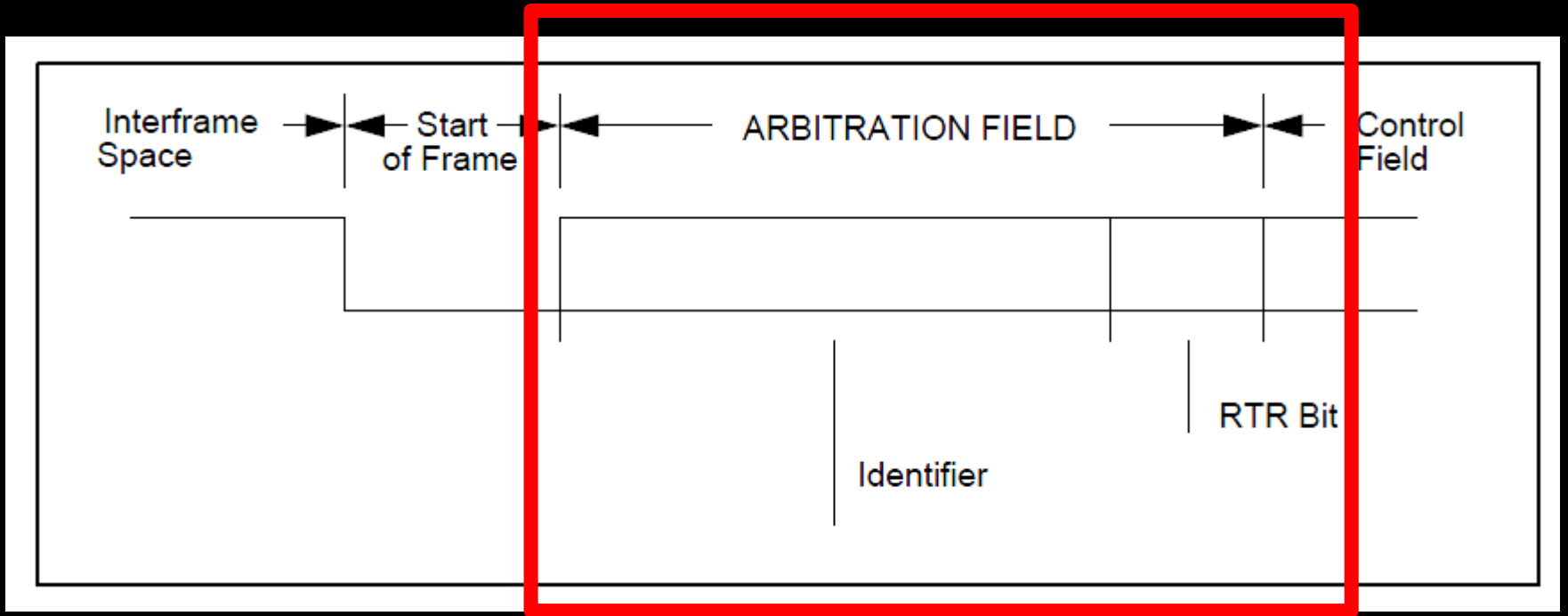

**CAN 2.0A** Pode-se ter até 2048 tipos distintos de mensagens



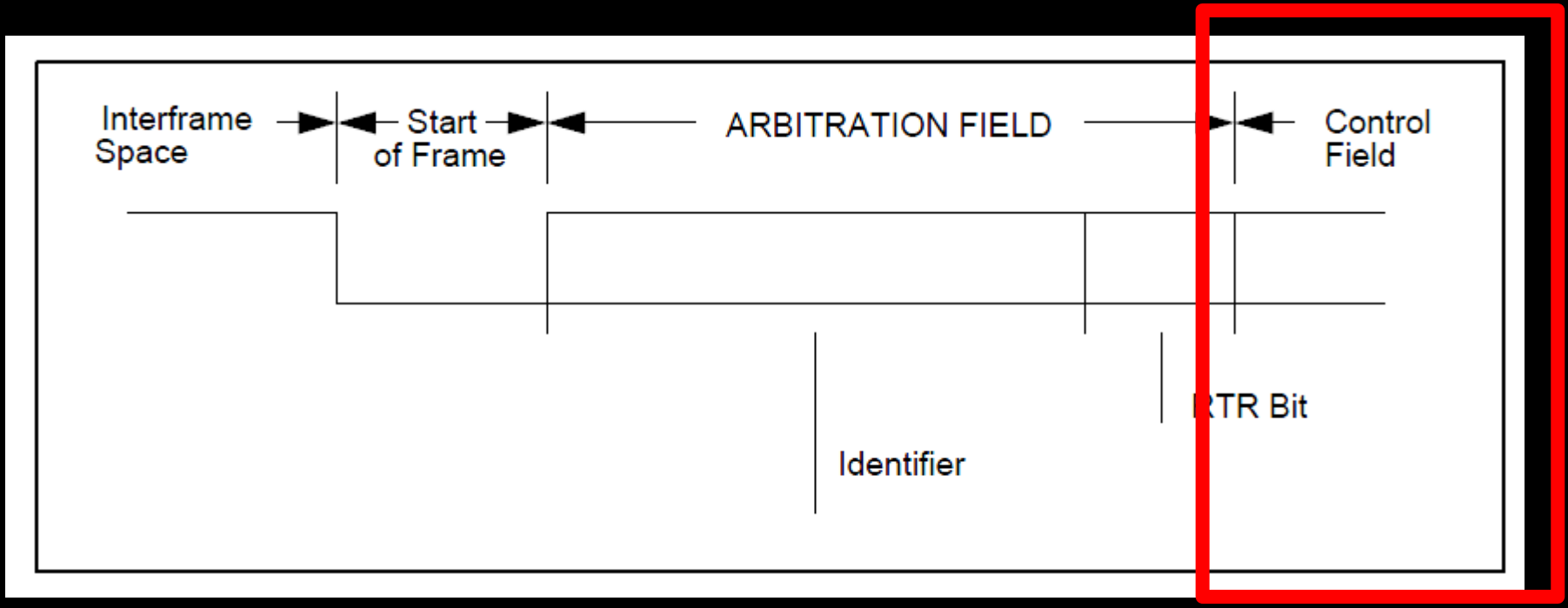

**CAN 2.0B** Pode-se ter até 537 milhões de tipos distintos de mensagens




- START OF FRAME (1 bit dominante): marca o início do DATA FRAME
- Um estação só pode iniciar uma transmissão quando o Barramento estiver livre (BUS IDLE).
- Uma vez que uma estação inicia a transmissão, todas as demais estações devem se sincronizar (a partir do START OF FRAME).

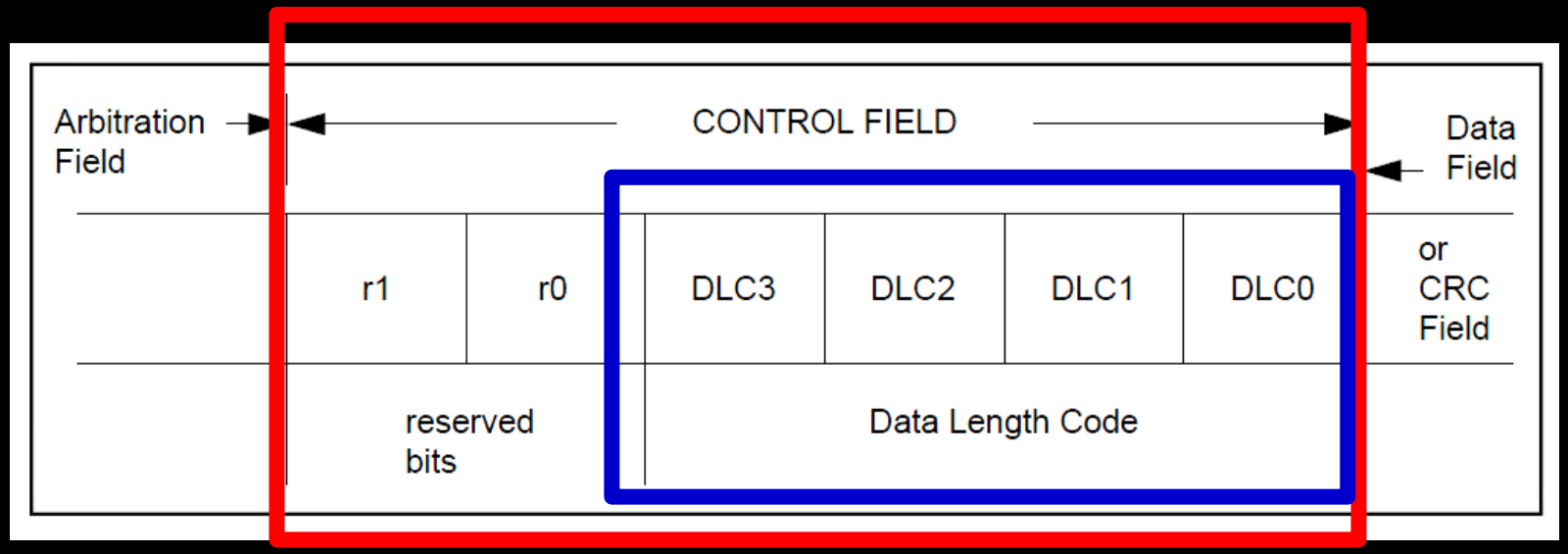



- **ARBITRATION FIELD** consiste de um IDENTIFICADOR e o RTR-bit.
- **IDENTIFICADOR** consiste de 11 bits. São transmitidos de ID-10 a ID-0. O bit menos significativo é o ID-0. Os 7 bits mais significativos (ID-10 a ID-4) não podem ser todos recessivos.
- **RTR-bit** (Remote Transmission Request). Nos casos de DATA FRAME o RTR-bit deve ser "dominante". No caso de REMOTE FRAME o RTR-bit deve ser "recessivo".



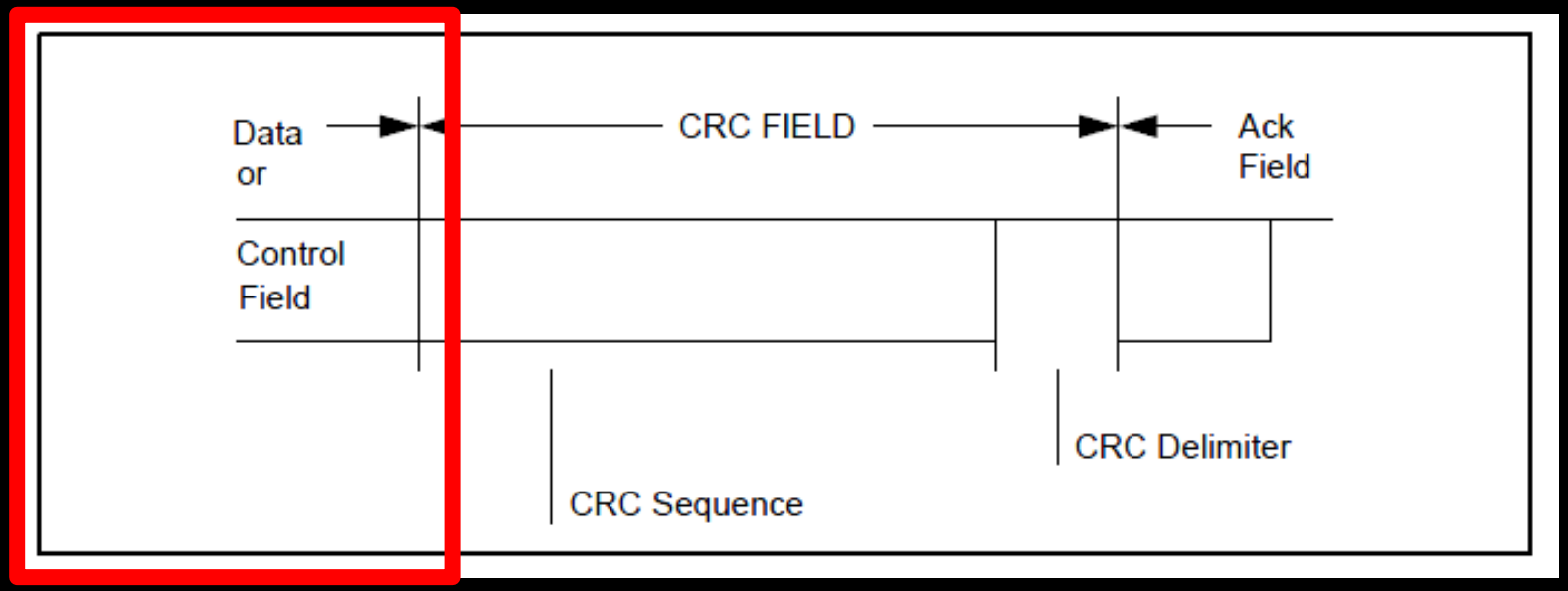

- **CONTROL FIELD** consiste de 6 bits. Inclui DATA LENGTH CODE e dois bits reservado para expansão futura. O bit reservado deve ser enviado como bit “dominante”.

- **DATA LENGTH CODE** - consiste de 4 bits e indica o tamanho do dado em termo de número de bytes (0 a 8). É transmitido dentro do campo CONTROL FIELD.

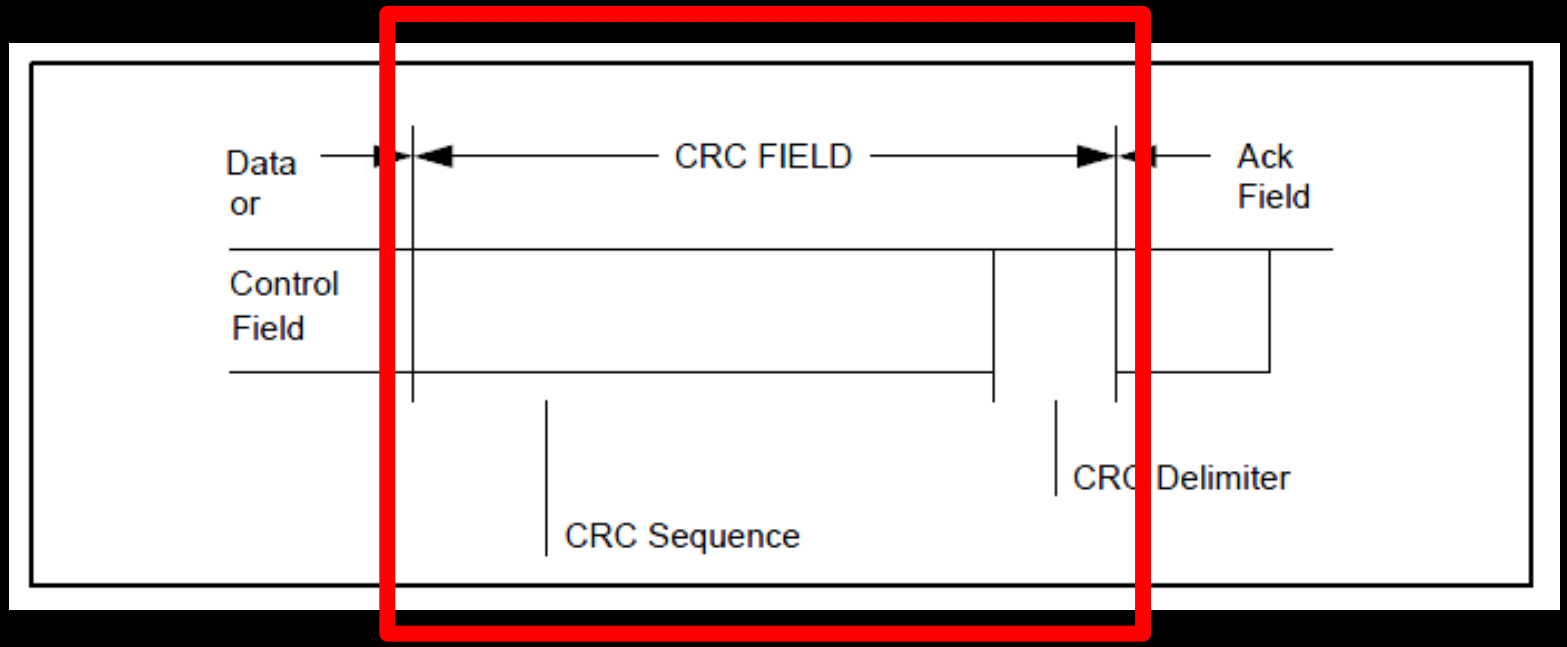



| Number of Data Bytes | Data Length Code | | | |
|---|---|---|---|---|
| | DLC3 | DLC2 | DLC1 | DLC0 |
| 0 | d | d | d | d |
| 1 | d | d | d | r |
| 2 | d | d | r | d |
| 3 | d | d | r | r |
| 4 | d | r | d | d |
| 5 | d | r | d | r |
| 6 | d | r | r | d |
| 7 | d | r | r | r |
| 8 | r | d | d | d |

Número de bytes do Data Frame: 0 a 8



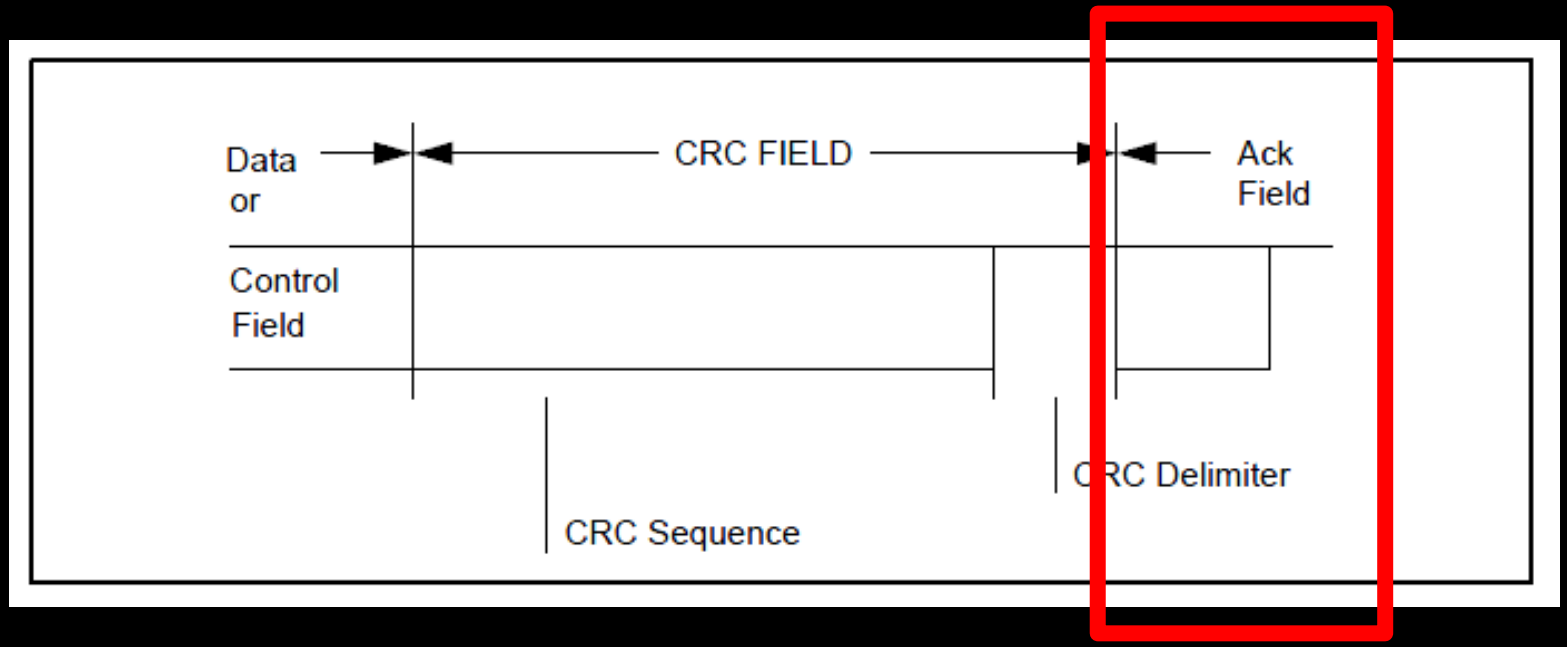

- **DATA FIELD** consiste de dados a serem transferidos dentro do DATA FRAME. Pode conter de 0 a 8 bytes sendo que o MSB transmitido primeiro.



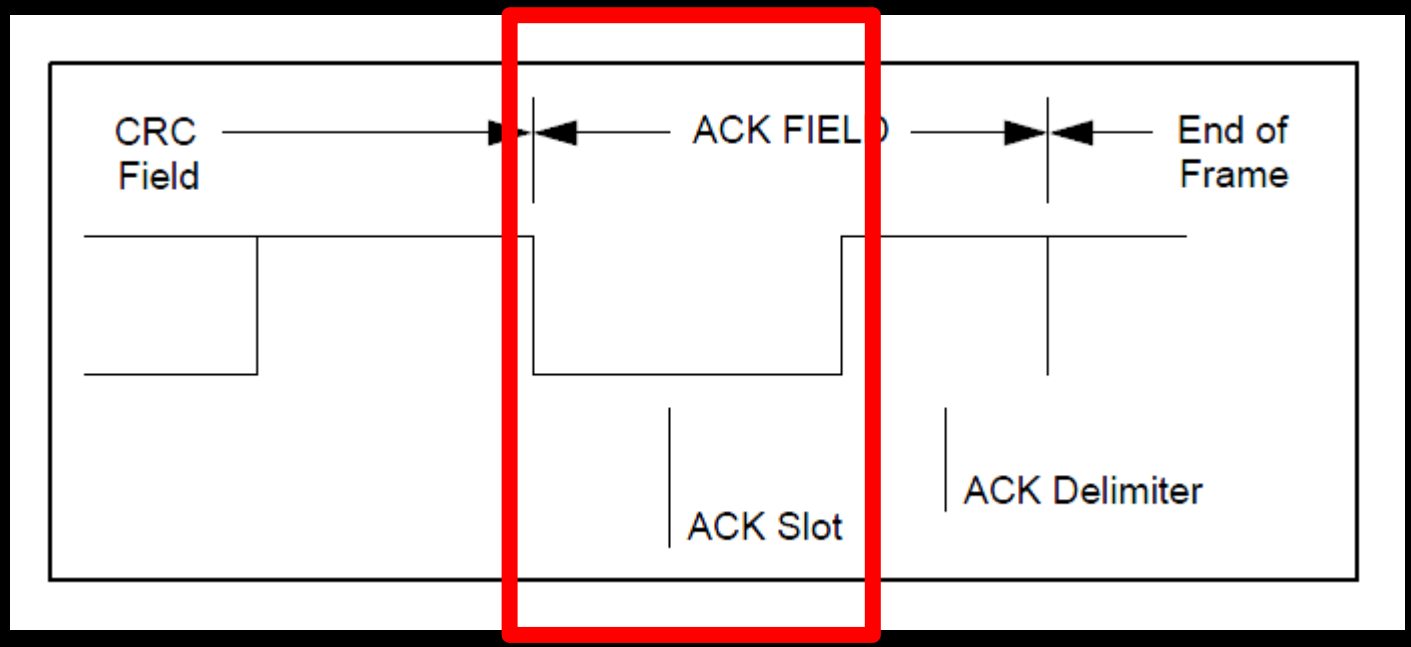

- **CRC FIELD** - contém CRC SEQUENCE seguido pro CRC DELIMITER (bit recessivo).

- **CRC SEQUENCE** – visa garantir a consistência do frame por meio da de código de redundância cíclica (CRC) obtida através de um cálculo polinomial. É obtido calculando o resto da divisão polinomial entre:

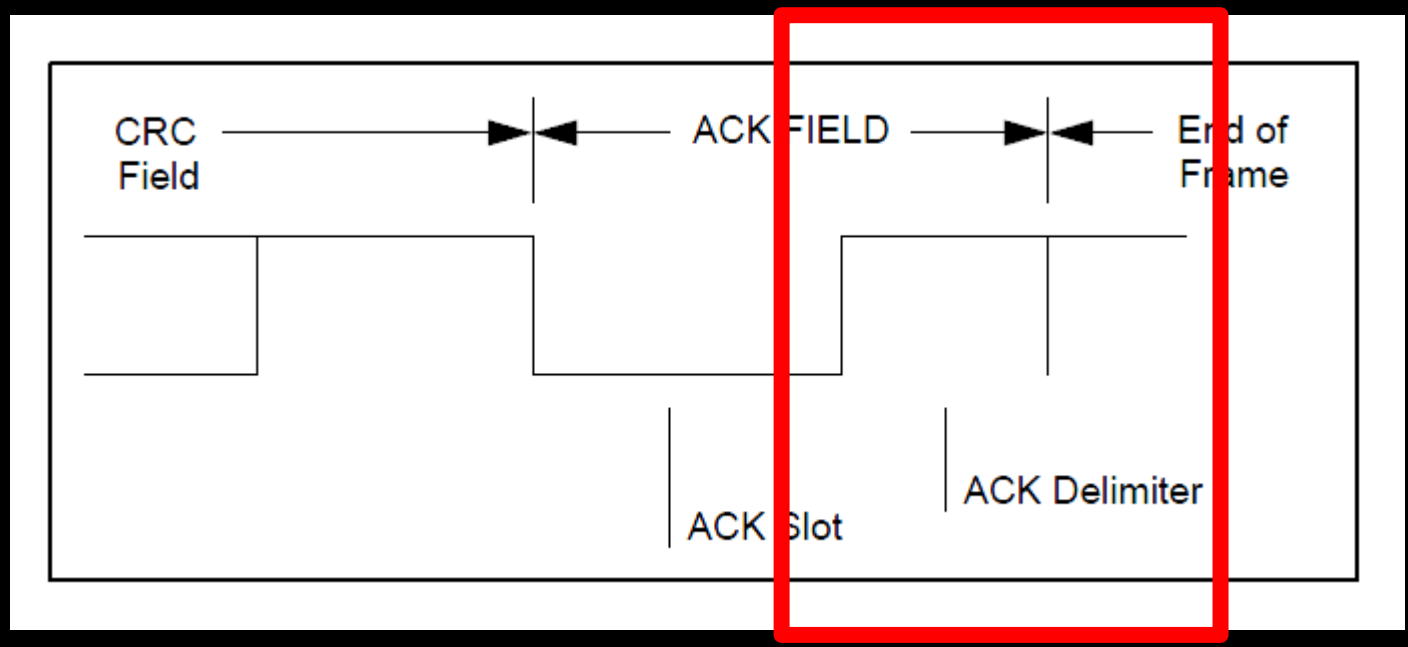
  - Polinômio com os coeficientes formados por: STF, ARBITRATION, CONTROL e DATA.
  - Polinômio gerador: $X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1$



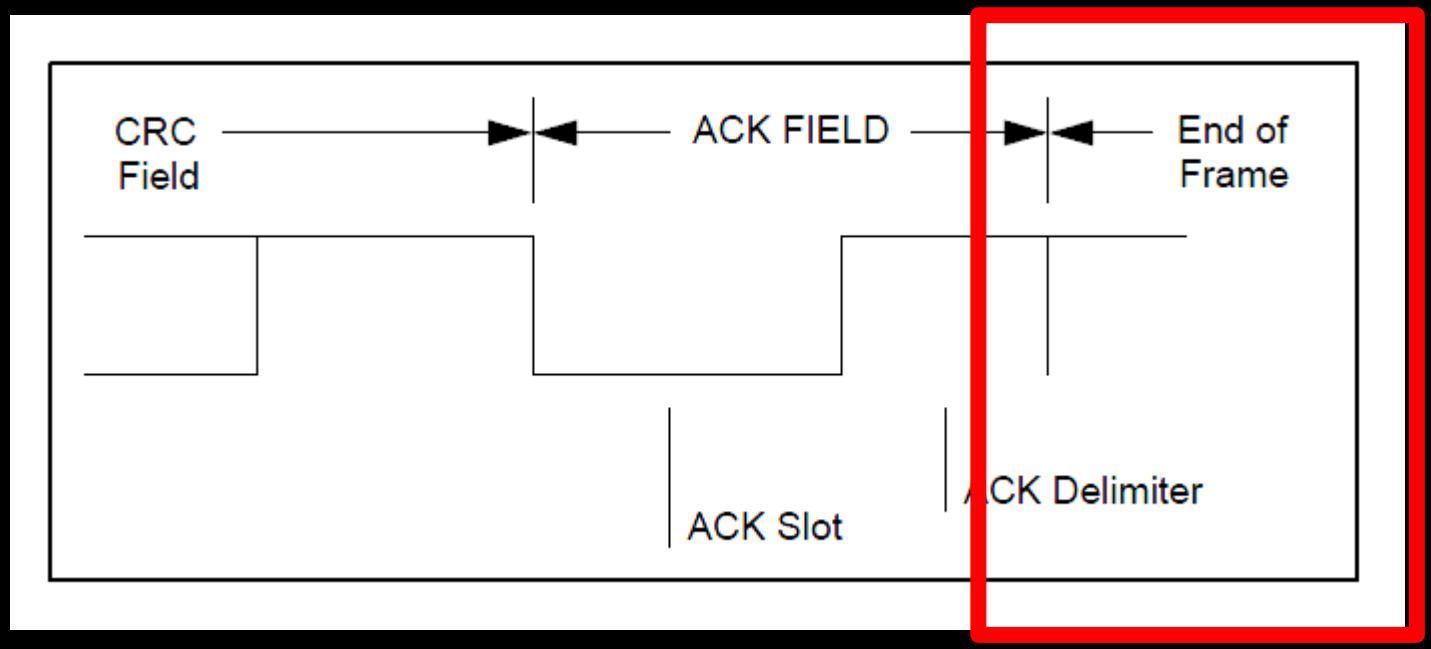

- **ACK FIELD -** tem dois bits de comprimento e contém o ACK SLOT e o ACK DELIMITER.
  - No campo ACK FIELD o transmissor envia dois bits “recessivos”.
  - O receptor que receber uma mensagem válida corretamente, reporta isto ao transmissor, enviando um bit “dominante” durante o ACK SLOT (corresponde a um ACK).



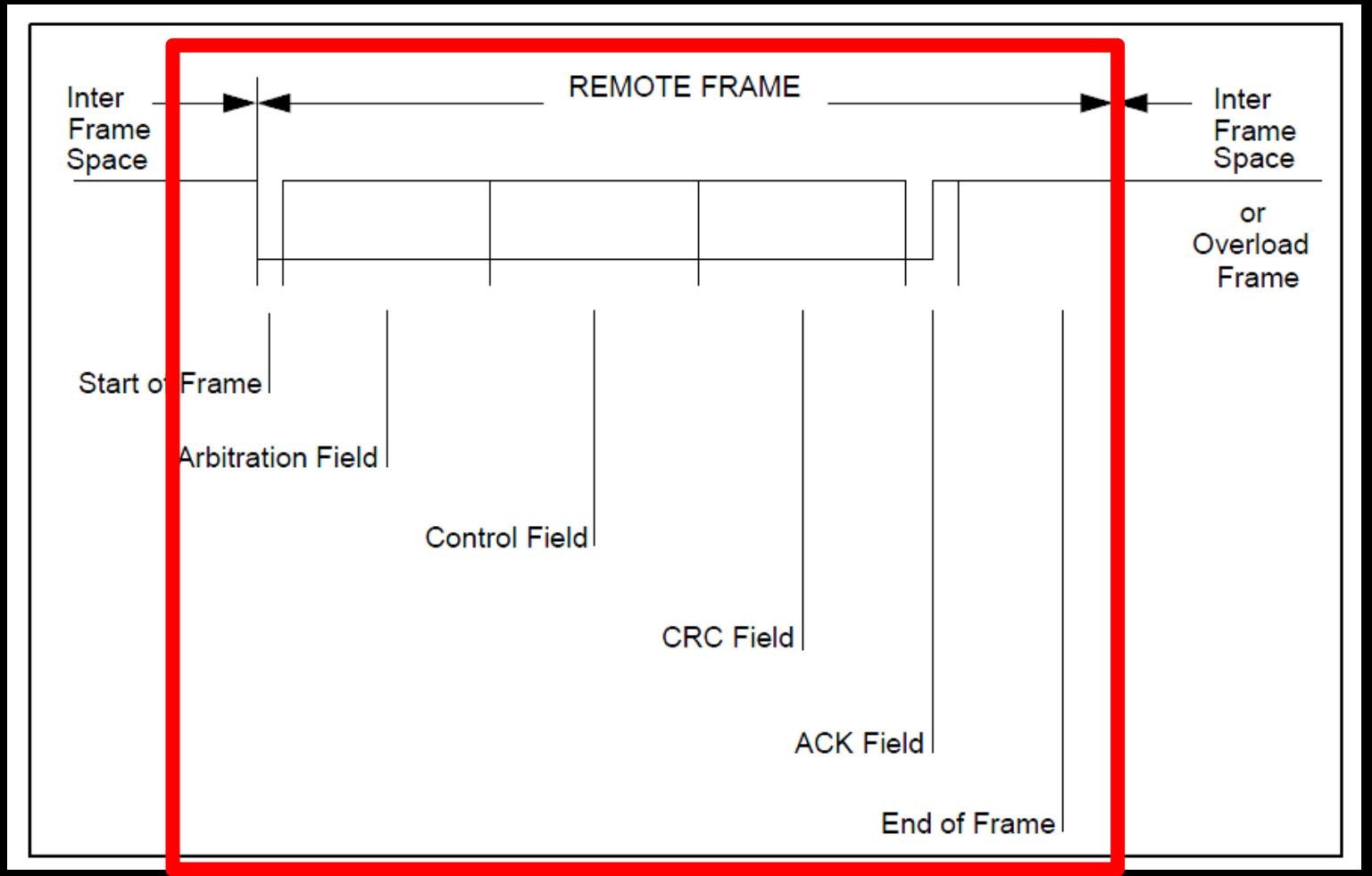

- **ACK SLOT -** todas estações que receberem CRC SEQUENCE válido reporta este fato durante o ACK SLOT sobrescrevendo o bit “recessivo” do transmissor com um bit “dominante”.




- **ACK DELIMITER -** é o segundo bit do ACK FIELD e deve ser um bit "recessivo". Como consequência o ACK SLOT é cercado por dois bits recessivos (CRC DELIMITER e o ACK DELIMITER).



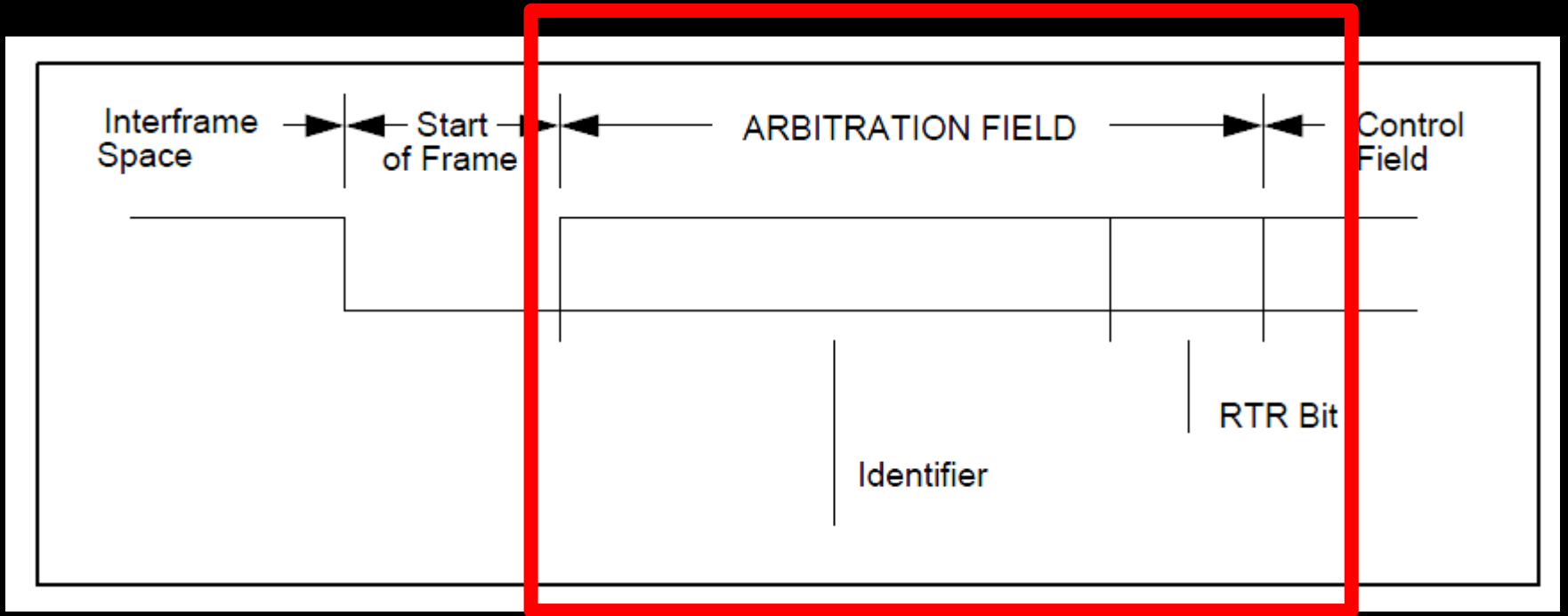

- **END of FRAME** - cada DATA FRAME e o REMOTE FRAME é delimitado por uma sequência de sinalização (flags) consistindo de sete bits recessivos.




- **REMOTE FRAME -** Não possui o campo DATA FIELD, independentemente do valor do DATA LENGTH CODE que pode possuir qualquer valor admissível entre 0 e 8. O valor do DATA LENGTH é o mesmo do DATA FRAME correspondente. A polaridade do RTR bit serve para indicar se o frame é um DATA FRAME ou um REMOTE FRAME.

## Processo de Arbitragem:

- O que fazer quando 2 ou mais nós querem transmitir dados ao mesmo tempo?
- A rede CAN resolve a situação utilizando um processo denominado “Arbitragem”.
- Arbitragem consiste num processo de determinação da mensagem mais prioritária.




- **ARBITRATION FIELD** consiste de um IDENTIFICADOR e o RTR-bit.
- **IDENTIFICADOR** consiste de 11 bits. São transmitidos de ID-10 a ID-0. O bit menos significativo é o ID-0. Os 7 bits mais significativos (ID-10 a ID-4) não podem ser todos recessivos.
- **RTR-bit** (Remote Transmission Request). Nos casos de DATA FRAME o RTR-bit deve ser "dominante". No caso de REMOTE FRAME o RTR-bit deve ser "recessivo".

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | |
|---|---|---|---|---|
| **A** | **205** | **0 0 1 0** | **0 0 0 0** | **0 1 0 1** |
| **B** | **250** | **0 0 1 0** | **0 1 0 1** | **0 0 0 0** |
| **C** | **400** | **0 1 0 0** | **0 0 0 0** | **0 0 0 0** |
| Ganha o Barramento | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | | | | | | | | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | **0** | | | | | | | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | **0** | **1** | | | | | | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | **0** | **1** | **0** | | | | | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | **0** | **1** | **0** | **0** | | | | | | | |

# Exemplo de processo de Arbitragem:

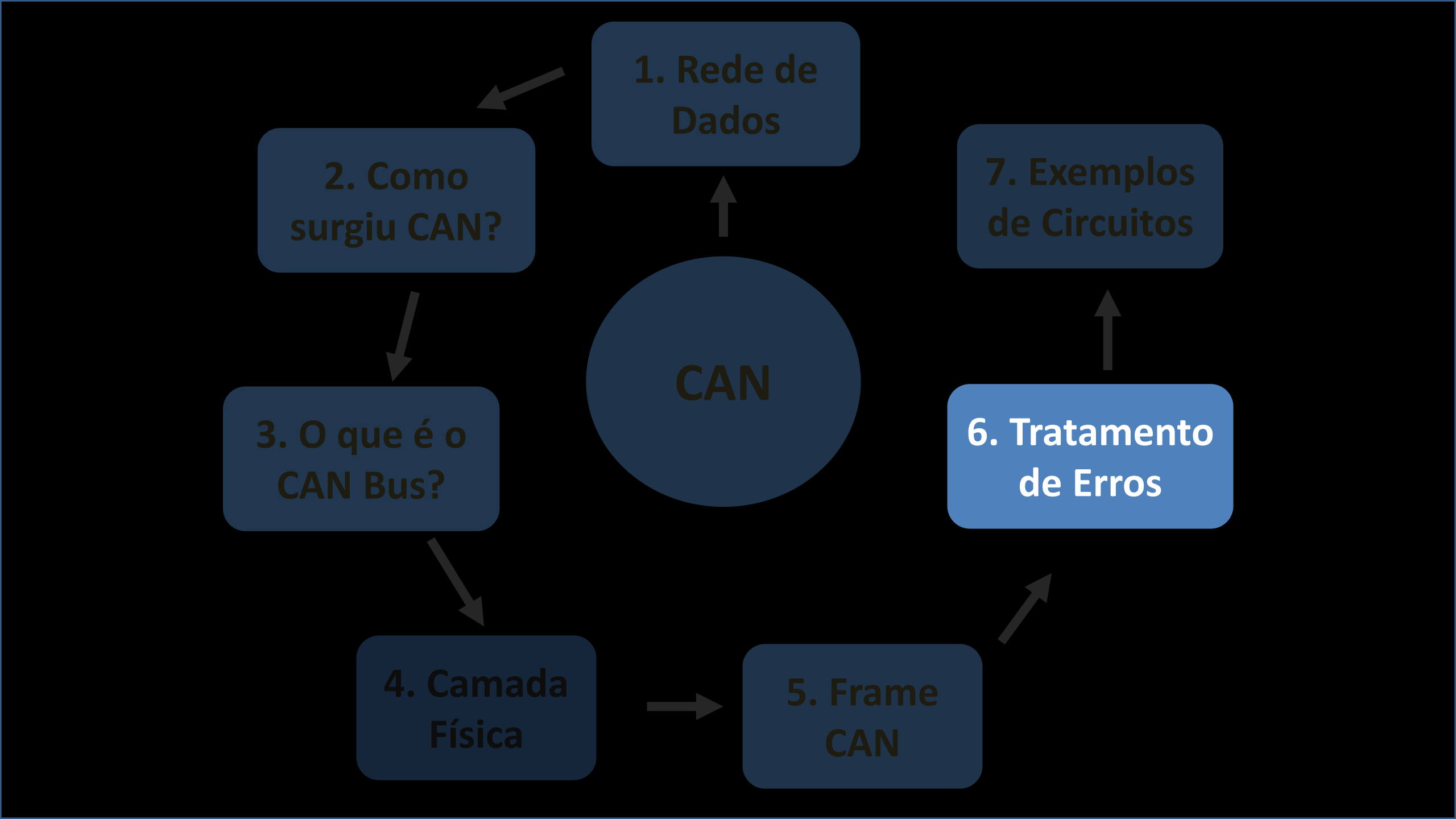
| Nós da Rede | ID | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Ganha o Barramento | | **0** | **0** | **1** | **0** | **0** | **0** | | | | | | |

# Exemplo de processo de Arbitragem:

| Nós da Rede | ID | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | | | |
| **B** | **250** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | | | |
| **C** | **400** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | | |
| Ganha o Barramento | **205** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | | | |

## No processo de Arbitragem:

- um bit Dominante sobrescreve eletricamente um bit Recessivo.
- Um módulo interrompe a sua transmissão caso perceba que existe outro módulo está transmitindo uma mensagem com prioridade maior (quando o seu bit recessivo for sobrescrito por um dominante.
- O módulo com maior prioridade prossegue a transmissão.


1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

## TIPOS DE MENSAGENS:

São 4 os tipos de mensagens trafegando nas redes **CAN**. São elas:

- Mensagens enviadas quando o sistema esta num estado funcional:
  - Data Frame
  - Remote Frame

- Mensagens enviadas quando o sistema esta num estado característico de erro:
  - Overload Frame
  - Error Frame

- **DATA FRAME**: transporta informações correntes do sistema, dados normais. Por exemplo, sinalizar temperatura do óleo).
- **REMOTE FRAME**: mensagem solicitando informação do sistema. Por exemplo, quando um dispositivo precisa verificar se há água no reservatório do sistema de limpeza do pára-brisa, este envia uma mensagem para o controlador do reservatório solicitando o estado atual do mesmo.
- **OVERLOAD FRAME**: mensagem que serve para indicar um estado ocupado do sensor, ou seja, quando um dispositivo envia uma mensagem de REMOTE FRAME para um dispositivo ocupado é devolvido uma mensagem OVERLOAD FRAME indicando que o dispositivo está congestionado.
- **ERROR FRAME**: mensagem que indica que não foi possível enviar uma mensagem de controle por ser recessiva. Serve também para indicar que um dispositivo apresenta falhas.

- **DATA FRAME**: transporta informações correntes do sistema, dados normais. Por exemplo, sinalizar temperatura do óleo).
- **REMOTE FRAME**: mensagem solicitando informação do sistema. Por exemplo, quando um dispositivo precisa verificar se há água no reservatório do sistema de limpeza do pára-brisa, este envia uma mensagem para o controlador do reservatório solicitando o estado atual do mesmo.
- **OVERLOAD FRAME**: mensagem que serve para indicar um estado ocupado do sensor, ou seja, quando um dispositivo envia uma mensagem de REMOTE FRAME para um dispositivo ocupado é devolvido uma mensagem OVERLOAD FRAME indicando que o dispositivo está congestionado.
- **ERROR FRAME**: mensagem que indica que não foi possível enviar uma mensagem de controle por ser recessiva. Serve também para indicar que um dispositivo apresenta falhas.

- **DATA FRAME**: transporta informações correntes do sistema, dados normais. Por exemplo, sinalizar temperatura do óleo).
- **REMOTE FRAME**: mensagem solicitando informação do sistema. Por exemplo, quando um dispositivo precisa verificar se há água no reservatório do sistema de limpeza do pára-brisa, este envia uma mensagem para o controlador do reservatório solicitando o estado atual do mesmo.
- **OVERLOAD FRAME**: mensagem que serve para indicar um estado ocupado do sensor, ou seja, quando um dispositivo envia uma mensagem de REMOTE FRAME para um dispositivo ocupado é devolvido uma mensagem OVERLOAD FRAME indicando que o dispositivo está congestionado.
- **ERROR FRAME**: mensagem que indica que não foi possível enviar uma mensagem de controle por ser recessiva. Serve também para indicar que um dispositivo apresenta falhas.

- **DATA FRAME**: transporta informações correntes do sistema, dados normais. Por exemplo, sinalizar temperatura do óleo).
- **REMOTE FRAME**: mensagem solicitando informação do sistema. Por exemplo, quando um dispositivo precisa verificar se há água no reservatório do sistema de limpeza do pára-brisa, este envia uma mensagem para o controlador do reservatório solicitando o estado atual do mesmo.
- **OVERLOAD FRAME**: mensagem que serve para indicar um estado ocupado do sensor, ou seja, quando um dispositivo envia uma mensagem de REMOTE FRAME para um dispositivo ocupado é devolvido uma mensagem OVERLOAD FRAME indicando que o dispositivo está congestionado.
- **ERROR FRAME**: mensagem que indica que não foi possível enviar uma mensagem de controle por ser recessiva. Serve também para indicar que um dispositivo apresenta falhas.

# DETECÇÃO DE ERRO (ERROR HANDLING)

- Faz parte intrínseca do protocolo CAN, e é uma parte essencial do desempenho da rede CAN.
- O mecanismo de error handling visa detectar a ocorrência de mensagens erradas no barramento CAN. Assim, o transmissor pode retransmitir uma mensagem com erro.
- Todo controlador CAN pertencente ao barramento sempre está tentando detectar erros numa mensagem.
- Se um erro é encontrado, o nó descobridor irá transmitir um ERROR FLAG, avisando outros nós de que há um destruidor de tráfego de rede.
- Os outros nós irão detectar o erro causado pelo ERROR FLAG (caso ainda não tenham detectado antes o erro original), e tomará uma ação apropriada, ou seja, descartará a mensagem com erro.

# DETECÇÃO DE ERRO (ERROR HANDLING) (cont.)

- Cada nó mantém dos contadores de erros. O Contador de Erros de Transmissão e o Contador de Erros de Recepção.
- Existem várias regras governando como esses contadores serão incrementados e/ou decrementados.
- Em essência, o transmissor com falha irá incrementar o seu Contador de Erros de Transmissão mais rapidamente do que os receptores irão incrementar os seus Contadores de Erros de Recepção.
- Isto ocorre porque há uma boa chance de que o ele próprio é o causador de falha.
- Quando um Contador de Erro ultrapassa um certo valor, o nó irá se tornar "ERROR PASSIVE", ou seja, desligar-se-á do barramento, o que significa que esse nó não participará do tráfego do barramento.
- O uso de contadores de erro, não só permite ao nó CAN detectar a falha como também confinar o erro.

# MECANISMOS DE DETECÇÃO DE ERRO

O protocolo CAN define cinco maneiras distintas de detectar erros. Duas delas funcionam ao nível de bits, e outros três no nível de mensagens.

- Bit Monitoring
- Bit Sutuffing
- Frame Check
- Acknowledgement Check
- Cyclic Redundant Check (CRC)

**Bit Monitoring**

- Cada transmissor da rede CAN monitora  a sua própria mensagem (ou seja, lê a própria mensagem). Se o nível do bit lido for diferente do que transmitiu, um BIT ERROR é sinalizado. ( menos durante o processo de arbitragem).

**Bit Stuffing**

- Quando cinco bits consecutivos de mesmo nível for transmitido por um nó, ele irá adicionar automaticamente um sexto bit de nível oposto à sequência de bits repetidos. O receptor irá remover esse bit extra.
- Isto é feito para remover o excesso de componente DC do barramento, mas isso também cria uma oportunidade para o receptor detectar erros. Caso mais de cinco bits consecutivos de mesmo nível ocorrer num barramento, o STUFF ERROR é sinalizado.

## Bit Monitoring

- Cada transmissor da rede CAN monitora  a sua própria mensagem (ou seja, lê a própria mensagem). Se o nível do bit lido for diferente do que transmitiu, um BIT ERROR é sinalizado. ( menos durante o processo de arbitragem).

## Bit Stuffing

- Quando cinco bits consecutivos de mesmo nível for transmitido por um nó, ele irá adicionar automaticamente um sexto bit de nível oposto à sequência de bits repetidos. O receptor irá remover esse bit extra.
- Isto é feito para remover o excesso de componente DC do barramento, mas isso também cria uma oportunidade para o receptor detectar erros. Caso mais de cinco bits consecutivos de mesmo nível ocorrer num barramento, o STUFF ERROR é sinalizado.

## Frame Check

- Algumas partes da mensagem CAN tem formatos fixos, ou seja, o padrão define qual nível deve ocorrer e quando. Por exemplo: CRC DELIMITER, ACK DELIMITER, END OF FRAME etc.
- Se o CAN controller detecta um valor inválido em um desses campos fixo sinalizará com FORM ERROR.

## Acknokedgement Check

- Todos nós que receberem uma mensagem corretamente (independente de estar ou não interessado no conteúdo) devem escrever um nível dominante no ACK SLOT da mensagem. O transmissor irá escrever um nível recessivo nesse lugar.
- Se o transmissor não conseguir detectar um nível dominante no ACK SLOT, um ACKNOWLEDGEMENT ERROR será sinalizado.

## Frame Check

- Algumas partes da mensagem CAN tem formatos fixos, ou seja, o padrão define qual nível deve ocorrer e quando. Por exemplo: CRC DELIMITER, ACK DELIMITER, END OF FRAME etc.
- Se o CAN controller detecta um valor inválido em um desses campos fixo sinalizará com FORM ERROR.

## Acknokedgement Check

- Todos nós que receberem uma mensagem corretamente (independente de estar ou não interessado no conteúdo) devem escrever um nível dominante no ACK SLOT da mensagem. O transmissor irá escrever um nível recessivo nesse lugar.
- Se o transmissor não conseguir detectar um nível dominante no ACK SLOT, um ACKNOWLEDGEMENT ERROR será sinalizado.

## Cyclic Redundancy Check

- Cada mensagem contém 15 bits de código de verificação de erro do tipo CRC (Cyclic Redundancy Checksum);
- Qualquer nó que detectar um CRC diferente daquela que ela mesma calculou irá gerar um CRC ERROR.

# Modos de Falhas do Barramento

A norma ISSO 11898 enumera os seguintes modos de falha do barramento CAN:

1. CAN_H interrompido
2. CAN_L interrompido
3. CAN_H em curto com a bateria (+)
4. CAN_L em curto com o terra (-)
5. CAN_H em curto com o terra (-)
6. CAN_L em curto com a bateria (+)
7. CAN_L em curto com o fio CAN_H
8. CAN_H e CAN_L interrompido no mesmo local
9. Perde da conexão com o terminador de rede

Nesses casos a norma recomenda que o barramento deve sobreviver com S/N reduzido.

# Modos de Falhas do Barramento

A norma ISSO 11898 enumera os seguintes modos de falha do barramento CAN:

1. CAN_H interrompido
2. CAN_L interrompido
3. CAN_H em curto com a bateria (+)
4. CAN_L em curto com o terra (-)
5. CAN_H em curto com o terra (-)
6. CAN_L em curto com a bateria (+)
7. CAN_L em curto com o fio CAN_H
8. CAN_H e CAN_L interrompido no mesmo local
9. Perde da conexão com o terminador de rede

Nesse caso os nós restantes devem sobreviver

# Modos de Falhas do Barramento

A norma ISSO 11898 enumera os seguintes modos de falha do barramento CAN:

1. CAN_H interrompido
2. CAN_L interrompido
3. CAN_H em curto com a bateria (+)
4. CAN_L em curto com o terra (-)
5. CAN_H em curto com o terra (-)
6. CAN_L em curto com a bateria (+)
7. CAN_L em curto com o fio CAN_H
8. CAN_H e CAN_L interrompido no mesmo local
9. Perde da conexão com o terminador de rede

Nesse caso é “opcional” que o barramento sobreviva com S/N reduzido.

## Modos de Falhas do Barramento

A norma ISSO 11898 enumera os seguintes modos de falha do barramento CAN:

1. CAN_H interrompido
2. CAN_L interrompido
3. CAN_H em curto com a bateria (+)
4. CAN_L em curto com o terra (-)
5. CAN_H em curto com o terra (-)
6. CAN_L em curto com a bateria (+)
7. CAN_L em curto com o fio CAN_H
8. CAN_H e CAN_L interrompido no mesmo local
9. Perde da conexão com o terminador de rede

Nesse caso é “opcional” que o barramento sobreviva com S/N reduzido.

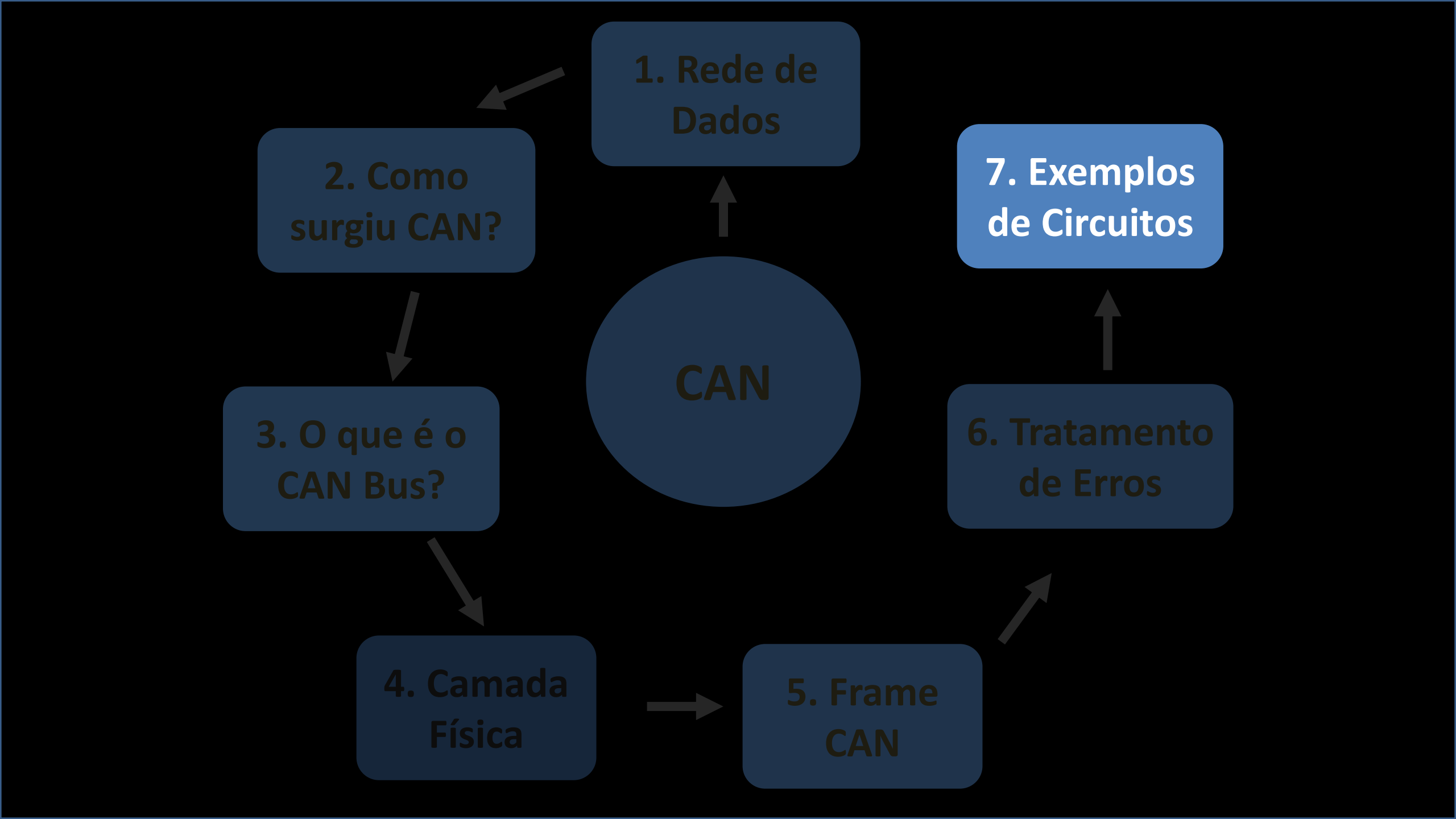

1. Rede de Dados
2. Como surgiu CAN?
3. O que é o CAN Bus?
4. Camada Física
5. Frame CAN
6. Tratamento de Erros
7. Exemplos de Circuitos
CAN

FIGURE 3: CAN BUS

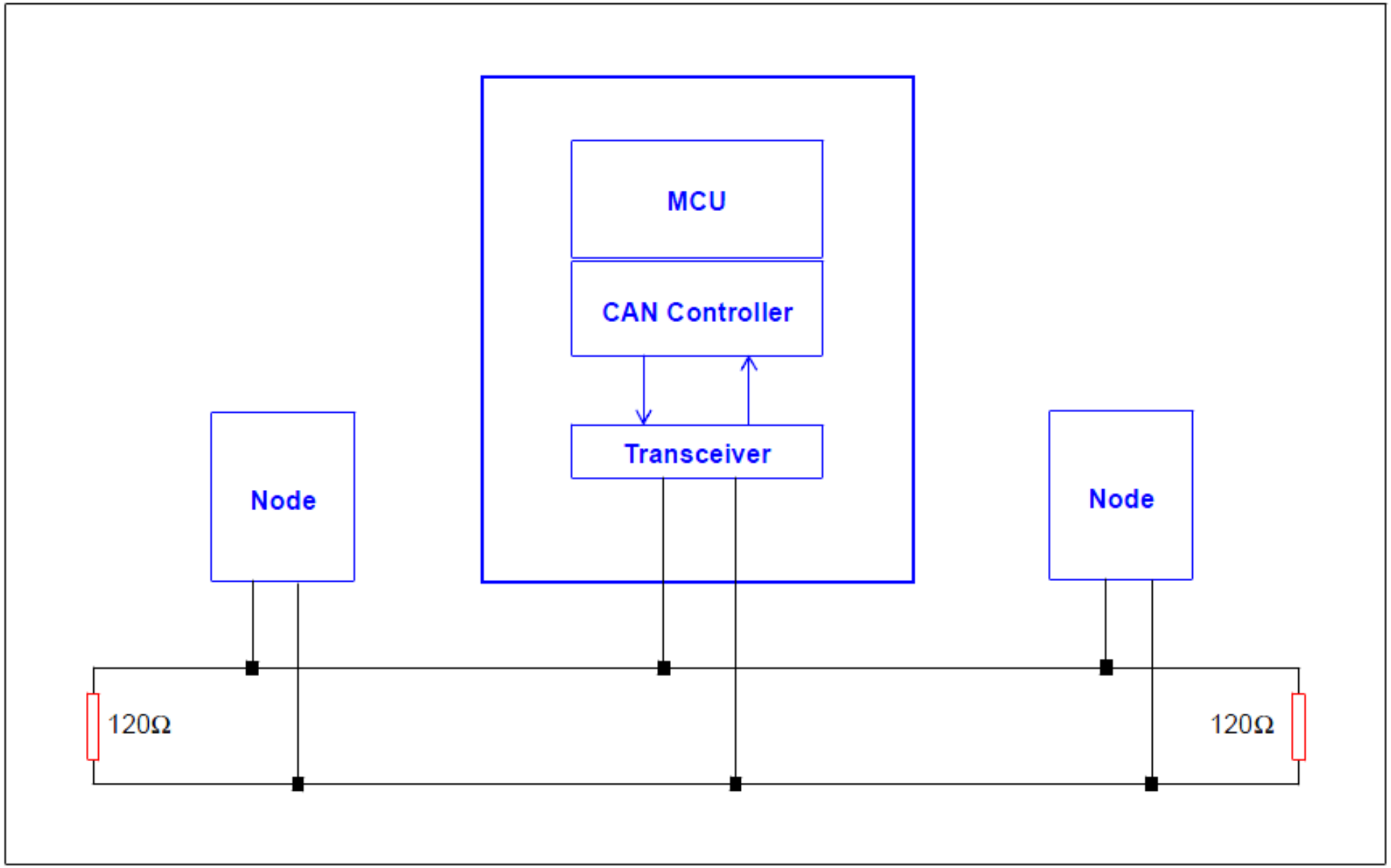
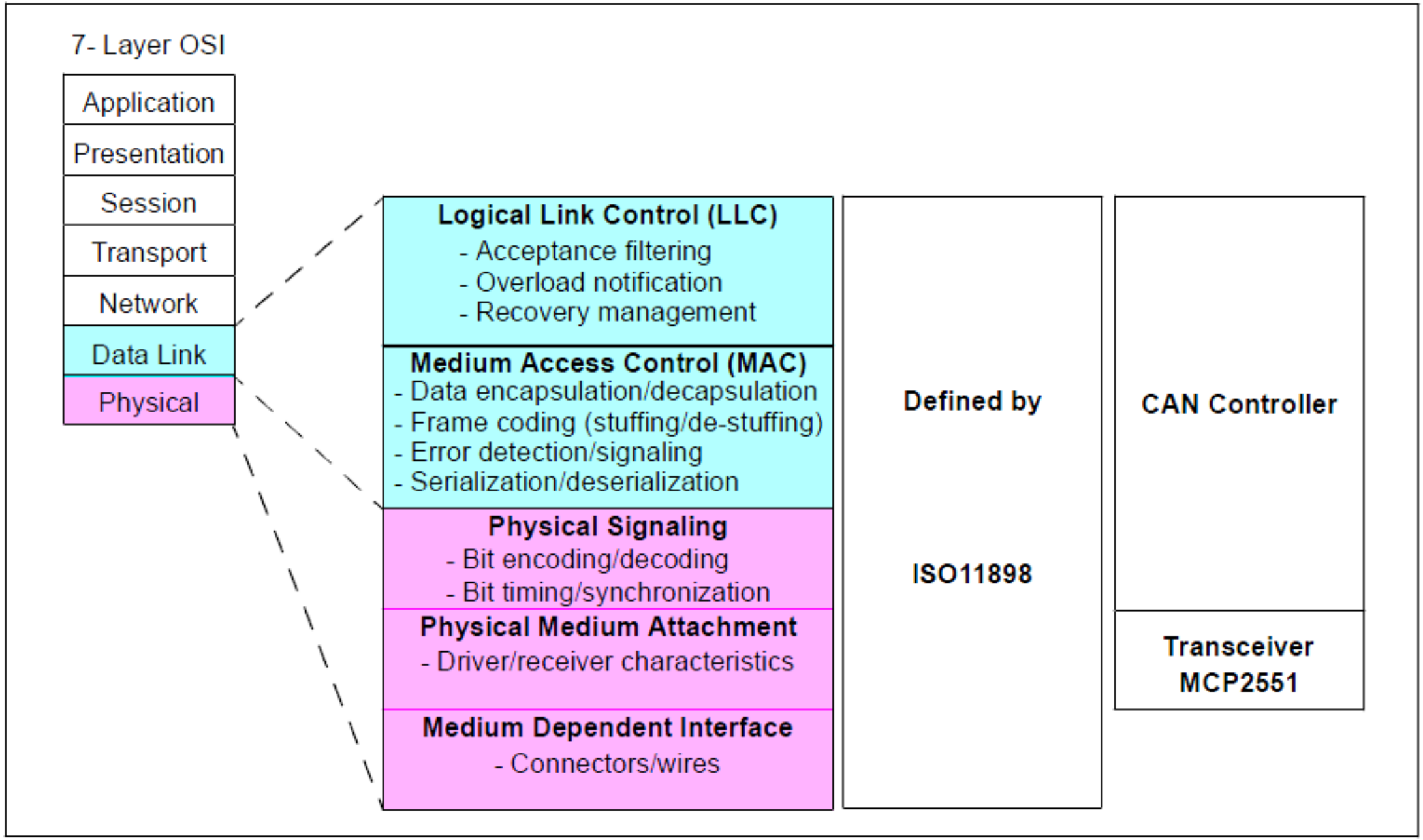


**FIGURE 1: CAN AND THE OSI MODEL**



MICROCHIP

# MCP2515

## Stand-Alone CAN Controller with SPI Interface

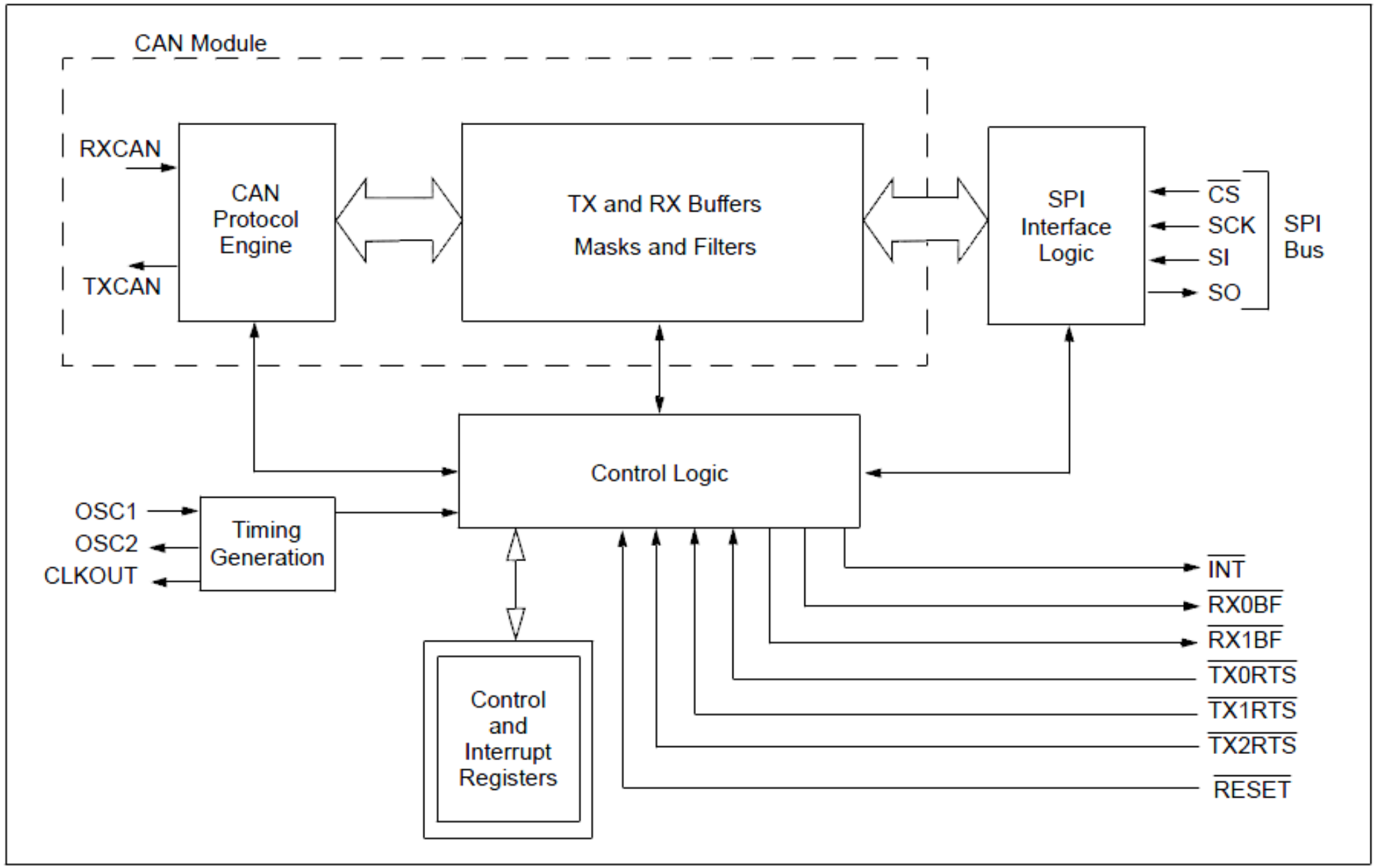
### Features:

- Implements CAN V2.0B at 1 Mb/s:
  - 0 – 8 byte length in the data field
  - Standard and extended data and remote frames
- Receive Buffers, Masks and Filters:
  - Two receive buffers with prioritized message storage
  - Six 29-bit filters
  - Two 29-bit masks
- Data Byte Filtering on the First Two Data Bytes (applies to standard data frames)
- Three Transmit Buffers with Prioritization and Abort Features
- High-Speed SPI Interface (10 MHz):
  - SPI modes 0,0 and 1,1
- One-Shot mode Ensures Message Transmission is Attempted Only One Time
- Clock Out Pin with Programmable Prescaler:
  - Can be used as a clock source for other device(s)
- Start-of-Frame (SOF) Signal is Available for Monitoring the SOF Signal:
  - Can be used for time-slot-based protocols and/or bus diagnostics to detect early bus degradation
- Interrupt Output Pin with Selectable Enables
- Buffer Full Output Pins Configurable as:
  - Interrupt output for each receive buffer
  - General purpose output
- Request-to-Send (RTS) Input Pins Individually Configurable as:
  - Control pins to request transmission for each transmit buffer
  - General purpose inputs
- Low-Power CMOS Technology:
  - Operates from 2.7V – 5.5V
  - 5 mA active current (typical)
  - 1 µA standby current (typical) (Sleep mode)
- Temperature Ranges Supported:
  - Industrial (I): -40°C to +85°C
  - Extended (E): -40°C to +125°C

# MCP2515

# CAN Controller

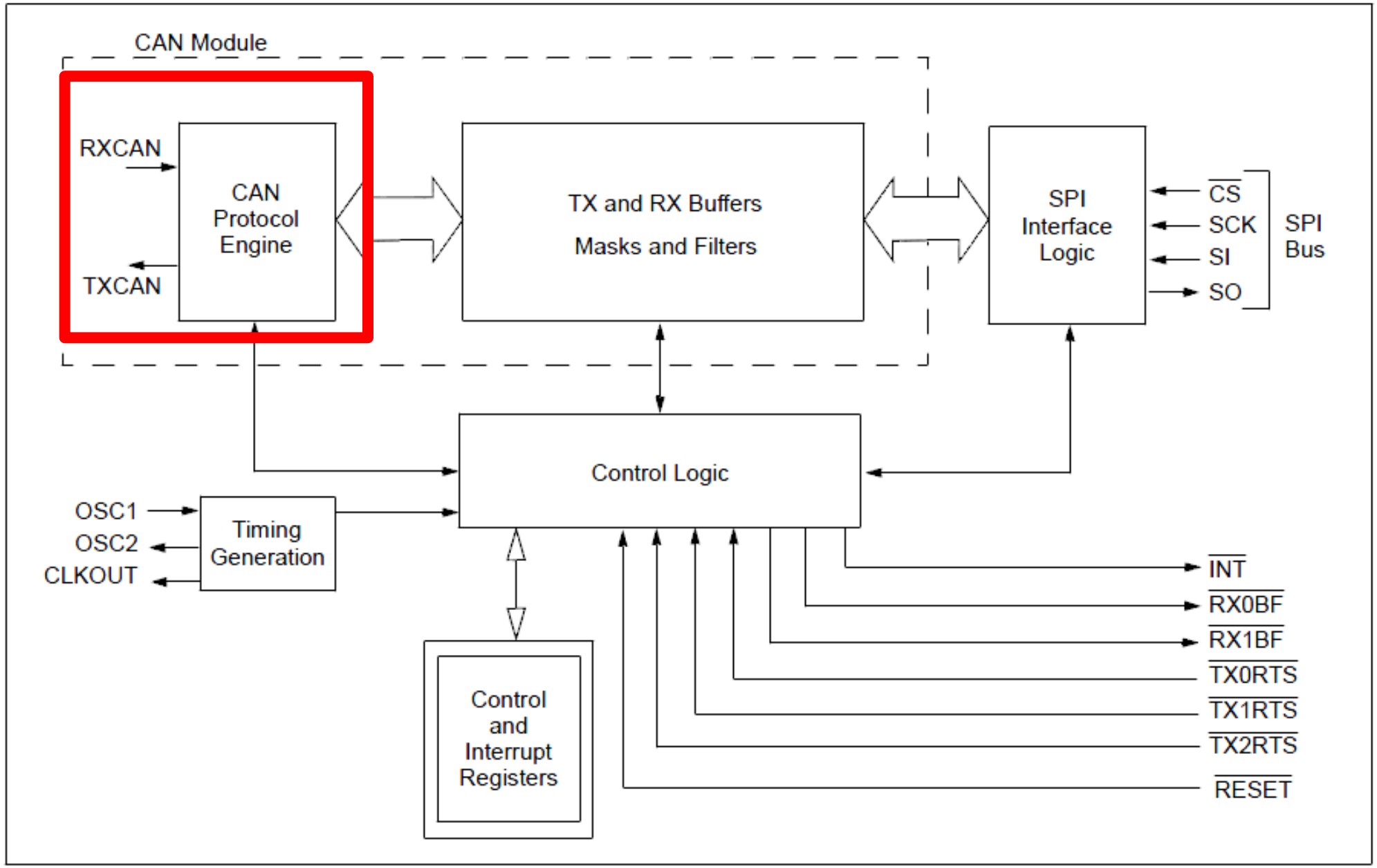
**FIGURE 1-1: BLOCK DIAGRAM**



# MCP2515

# CAN Controller

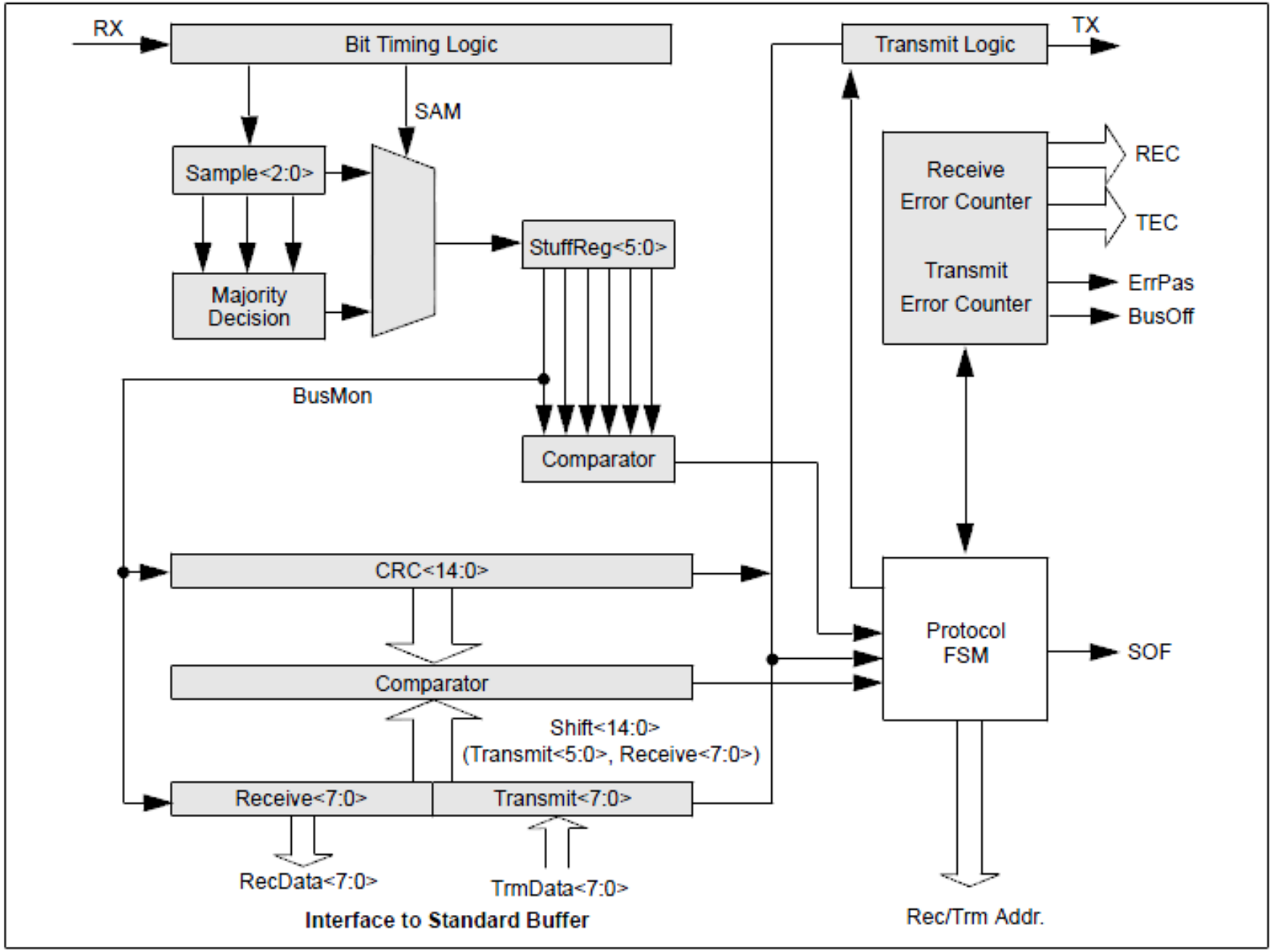
**FIGURE 1-1: BLOCK DIAGRAM**



# MCP2515

# CAN Controller

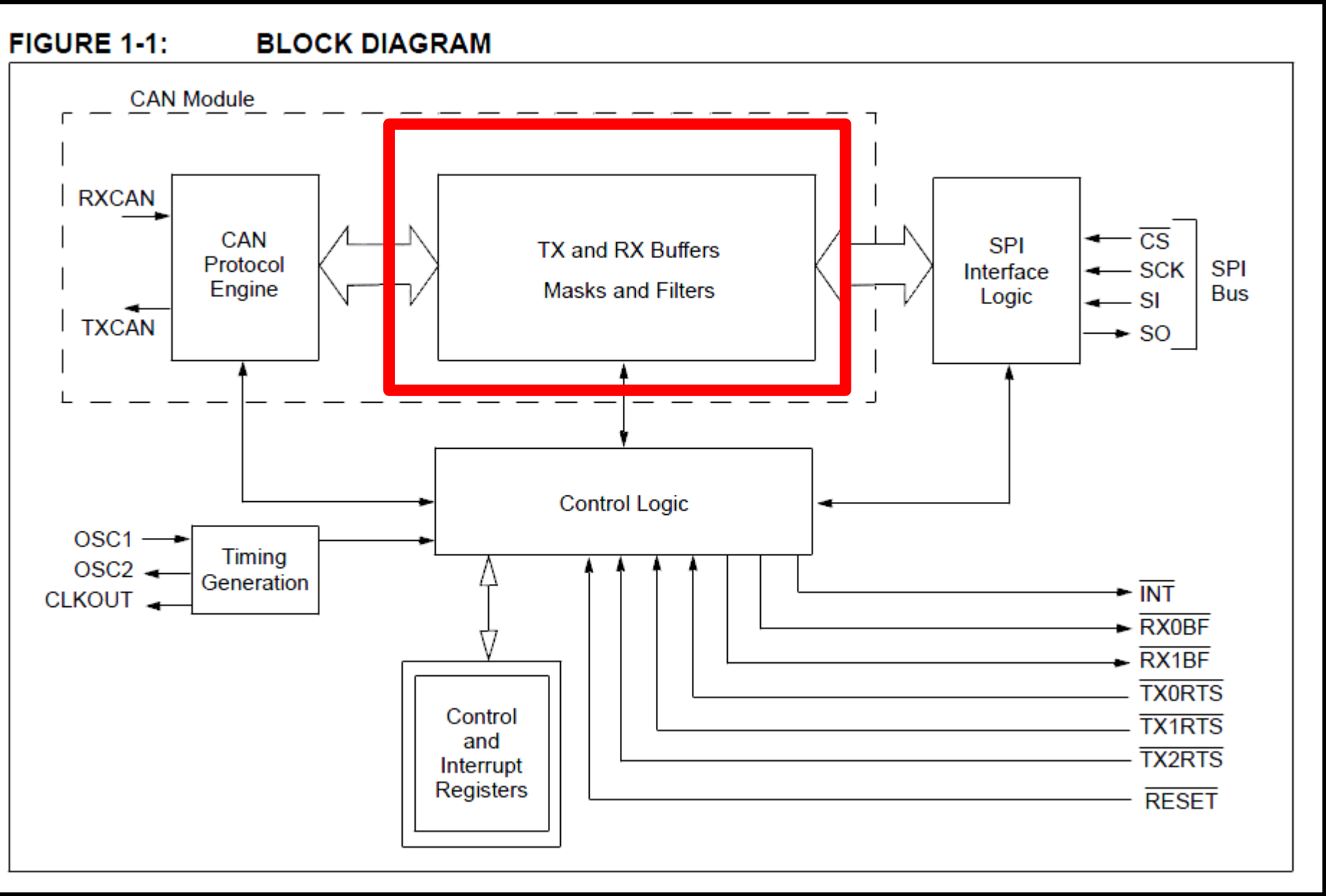
FIGURE 1-4: CAN PROTOCOL ENGINE BLOCK DIAGRAM



# MCP2515

# CAN Controller

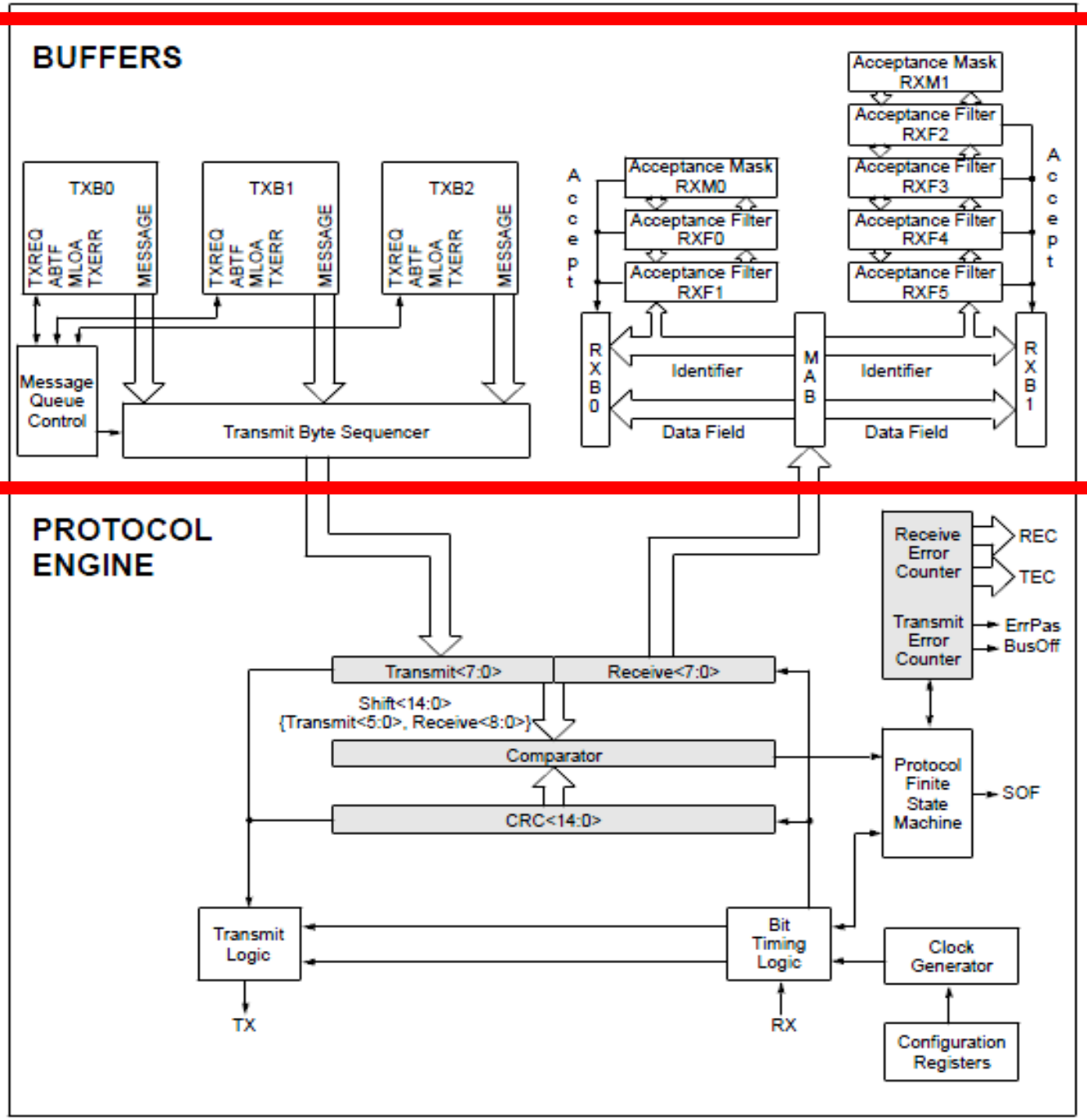
FIGURE 1-1: BLOCK DIAGRAM



# MCP2515

# CAN Controller

FIGURE 1-3: CAN BUFFERS AND PROTOCOL ENGINE BLOCK DIAGRAM



# MCP2551

## CAN Tranceiver

FIGURE 6: MCP2551 BLOCK DIAGRAM

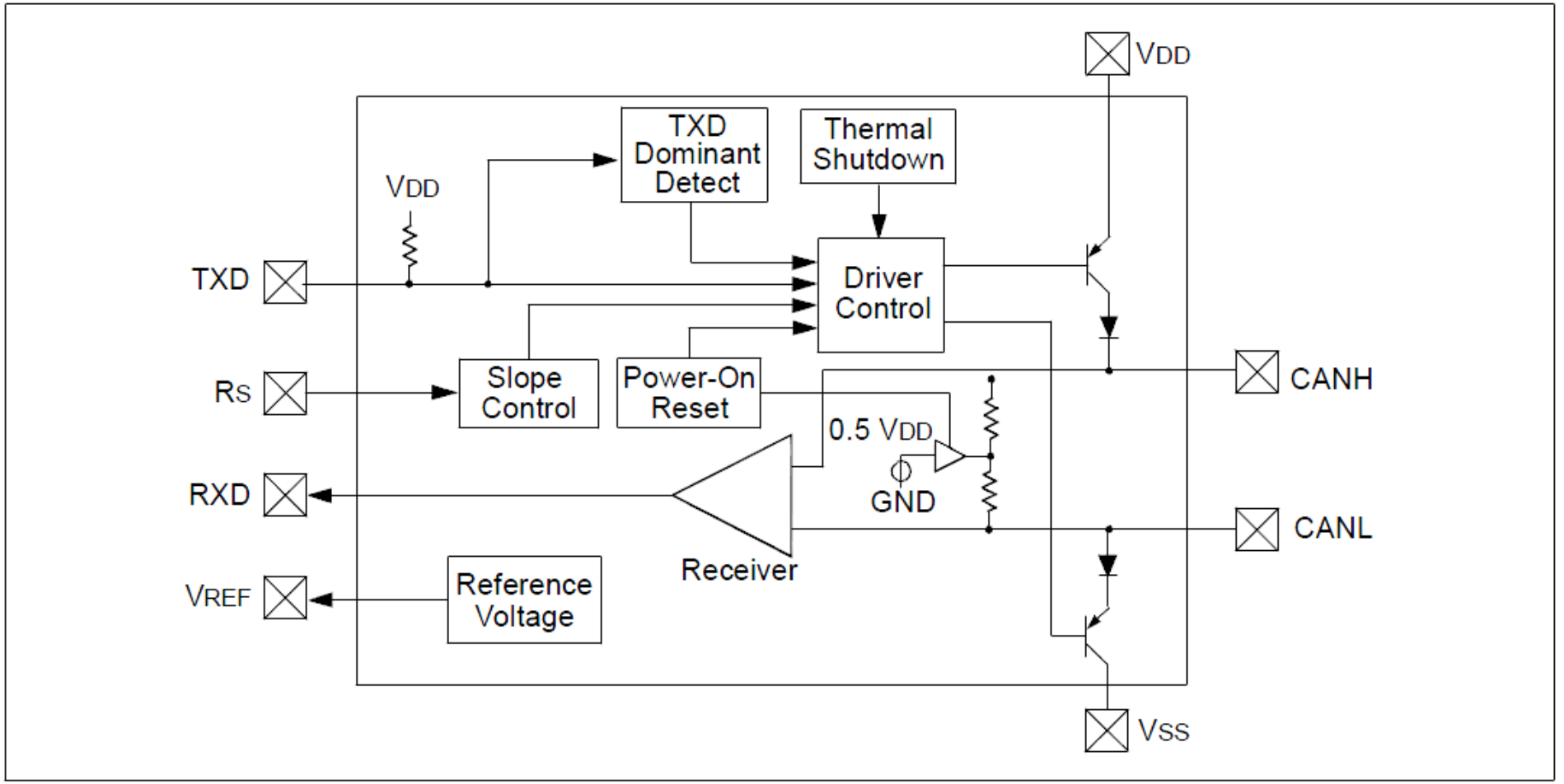
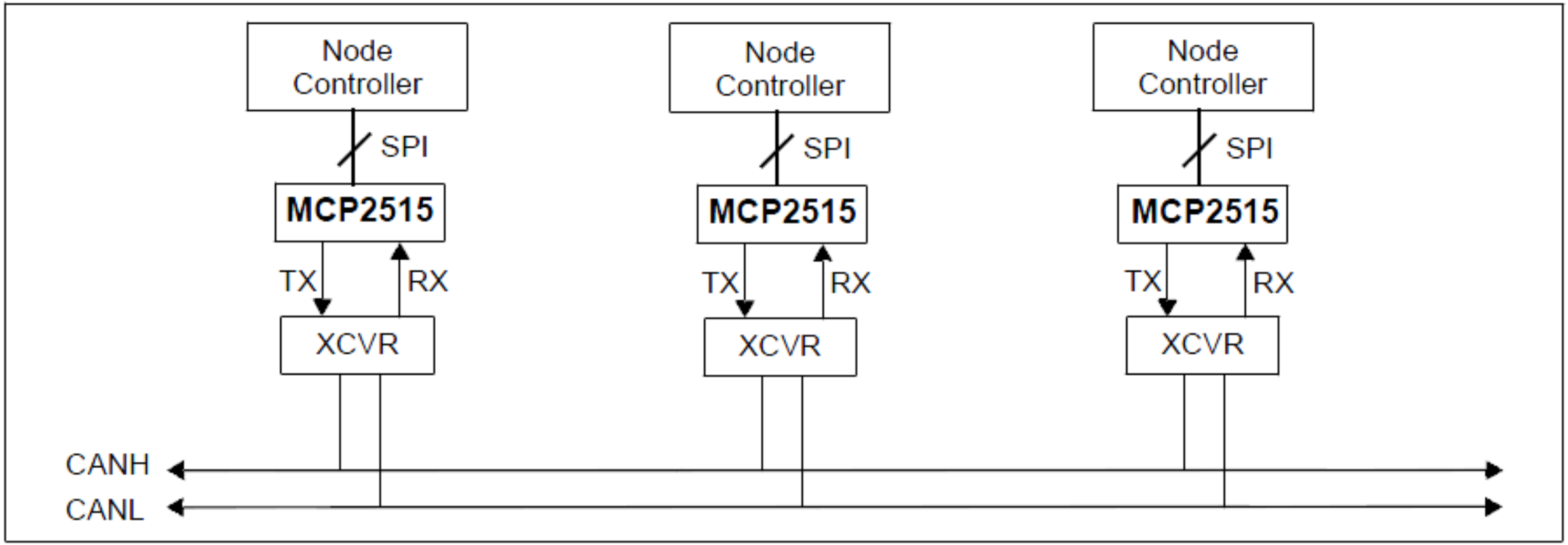
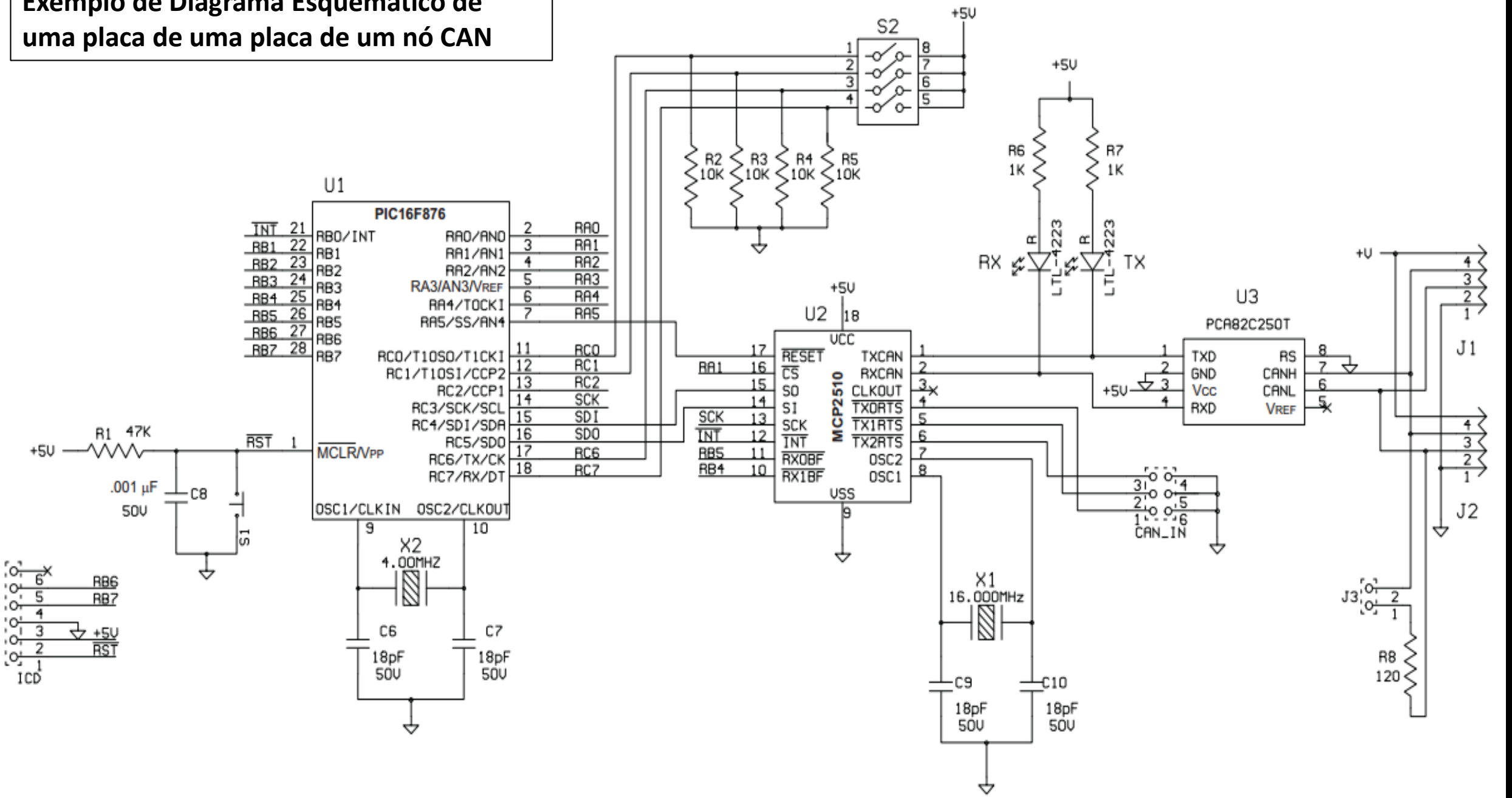


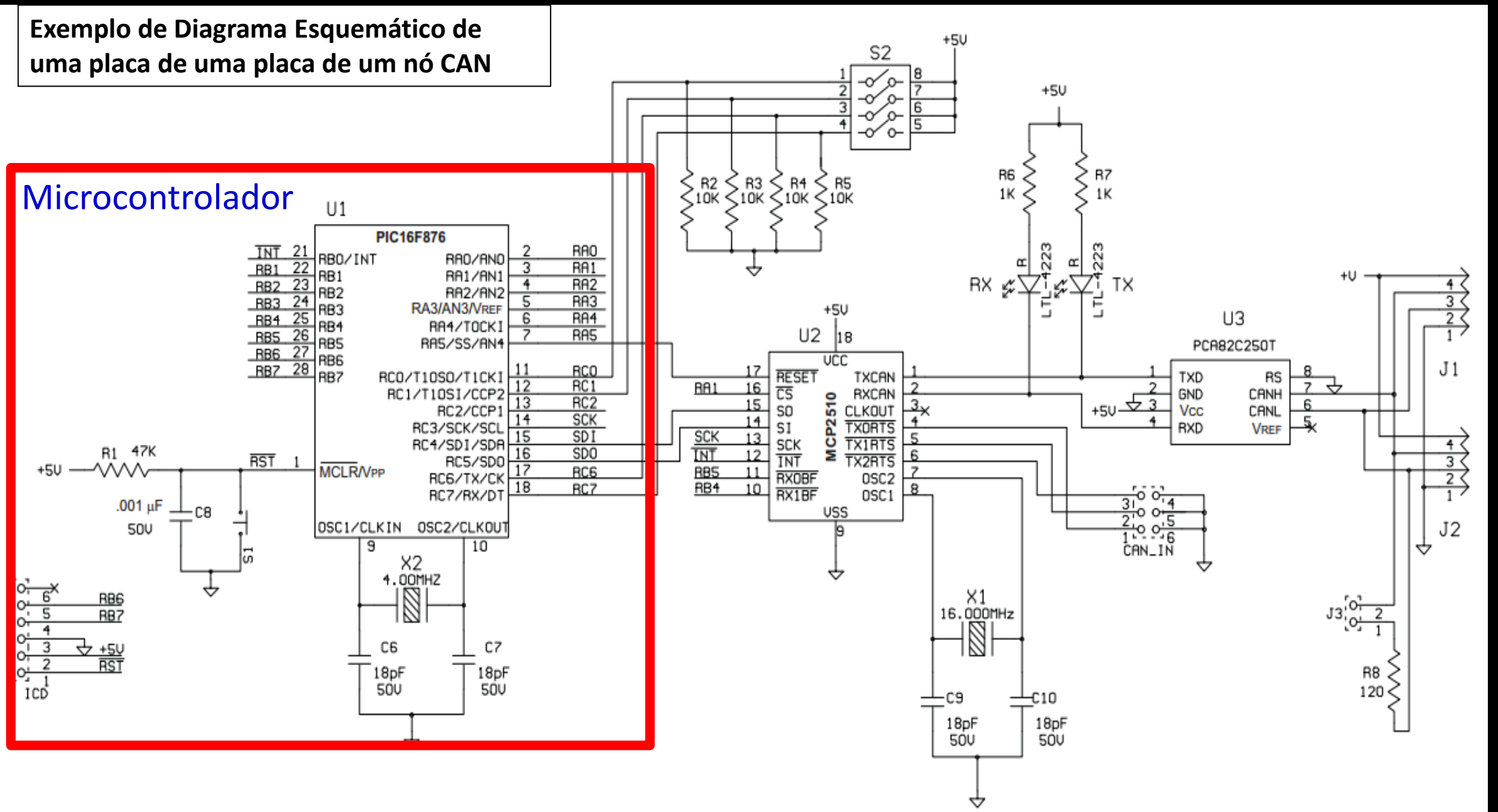
**FIGURE 1-2: EXAMPLE SYSTEM IMPLEMENTATION**



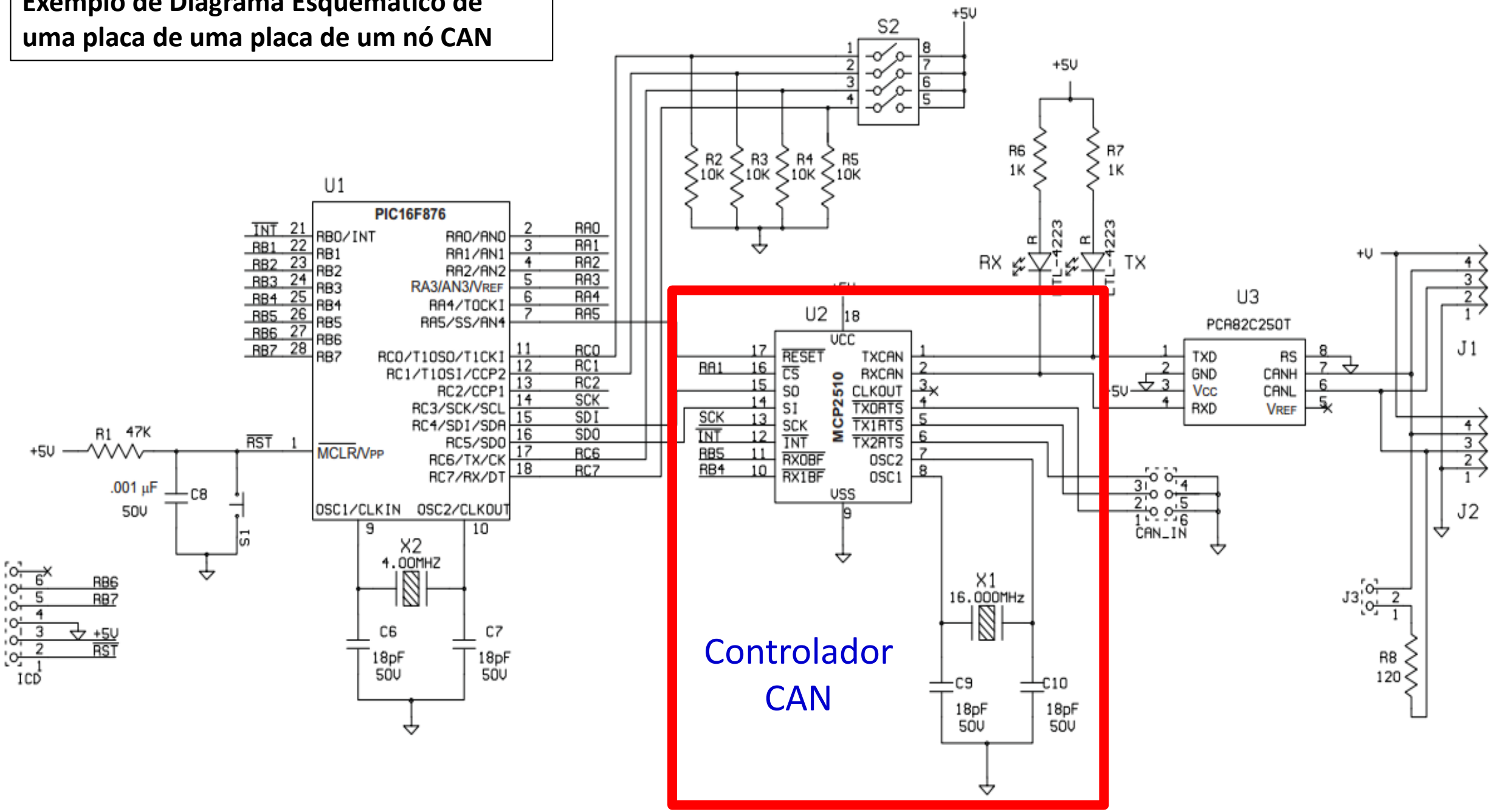
**Exemplo de Diagrama Esquemático de uma placa de uma placa de um nó CAN**



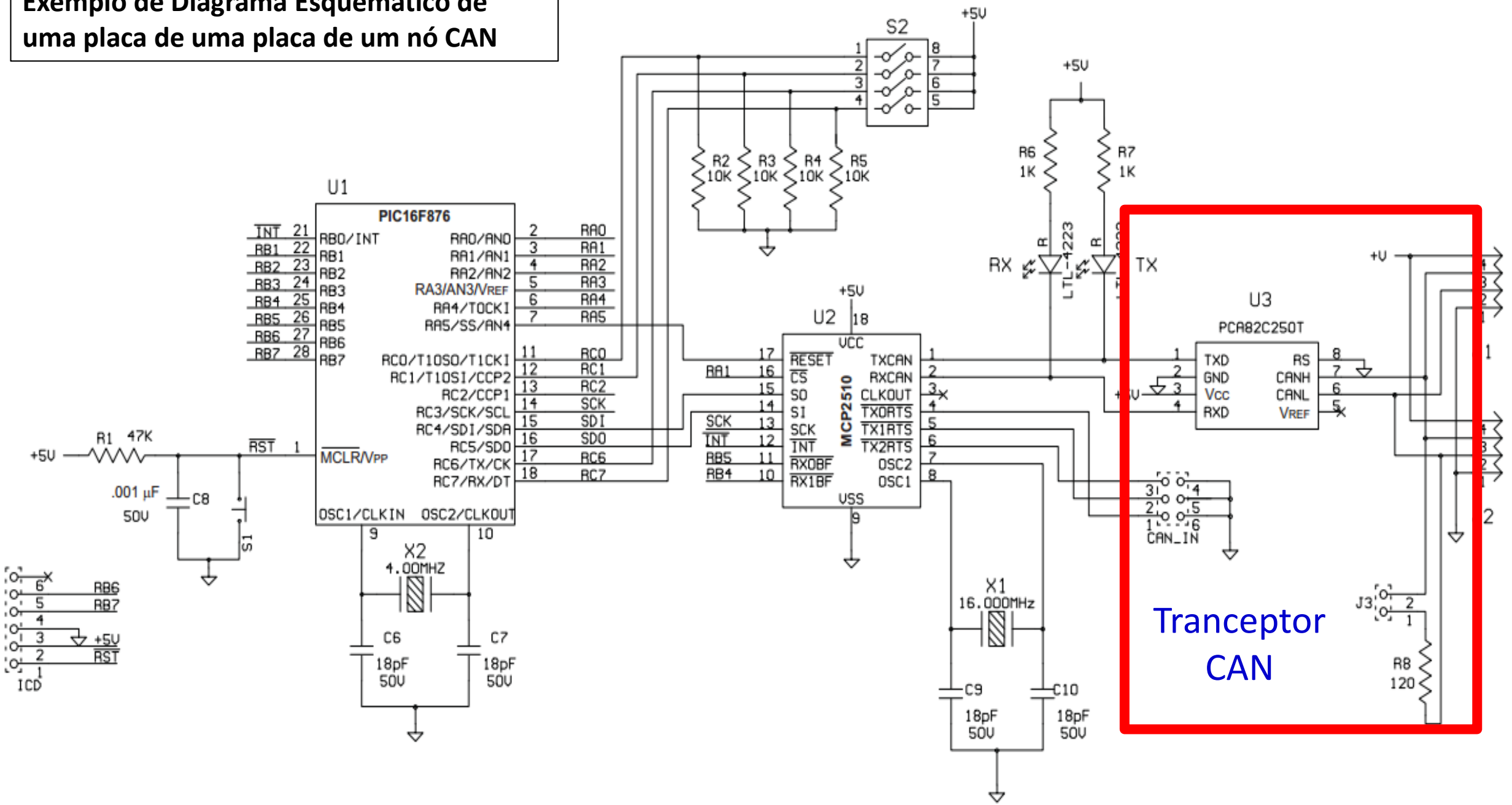
**Exemplo de Diagrama Esquemático de uma placa de uma placa de um nó CAN**



**Exemplo de Diagrama Esquemático de uma placa de uma placa de um nó CAN**



**Exemplo de Diagrama Esquemático de uma placa de uma placa de um nó CAN**



# MCP2515 Stand-Alone CAN Controller PICtail™ Demo Board User's Guide

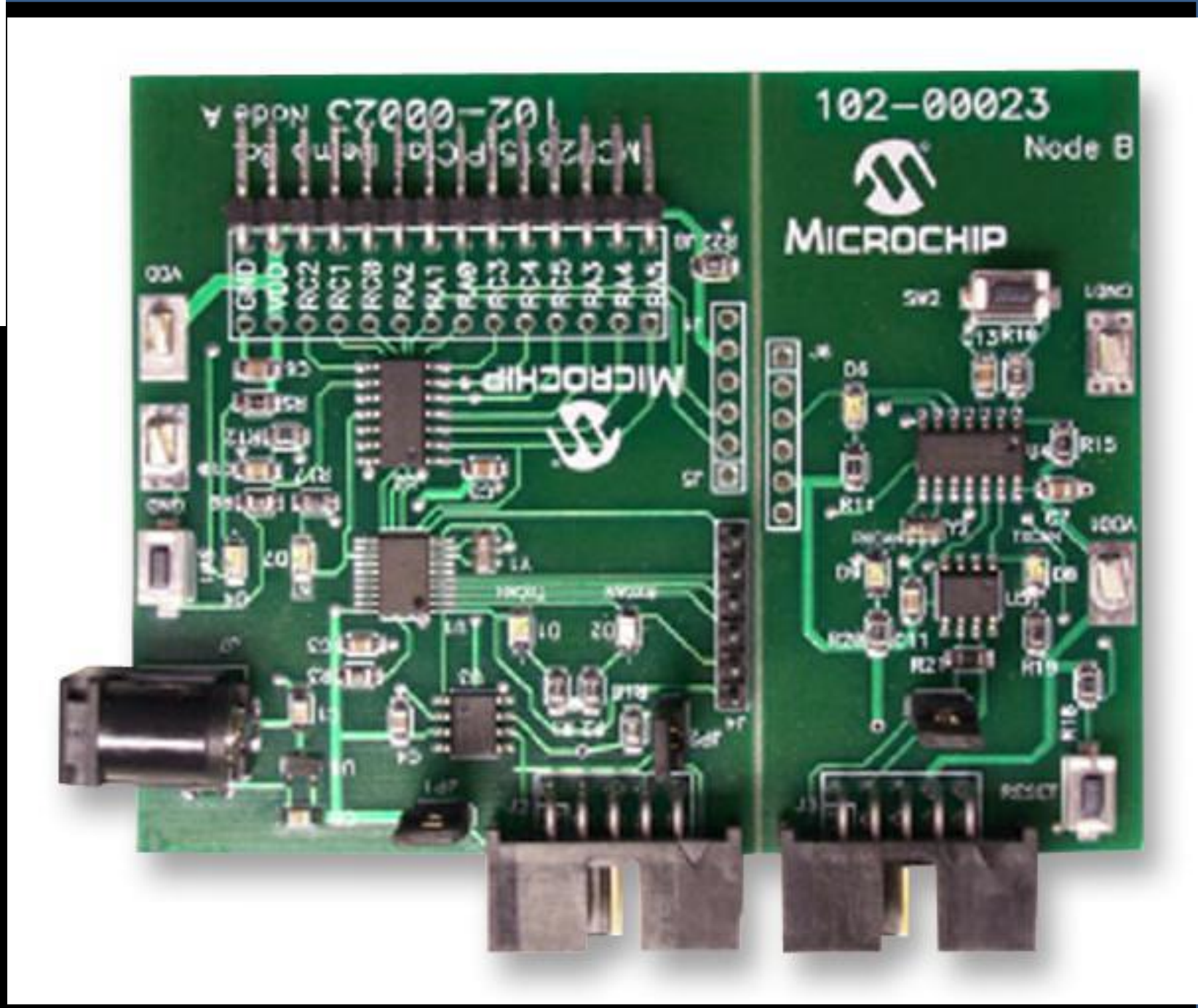


**FIGURE 2-2: FIRMWARE FLOW DIAGRAM**

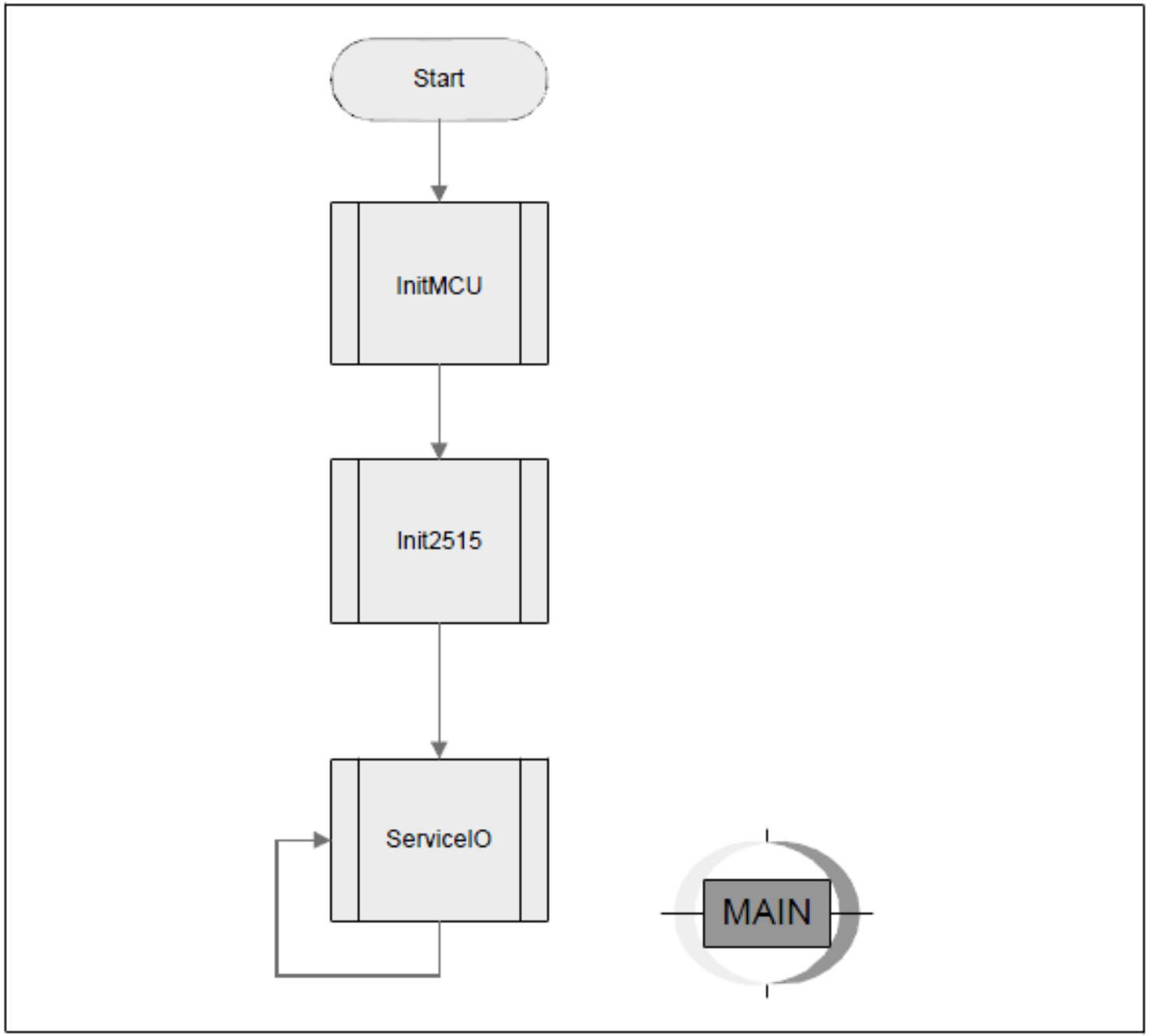
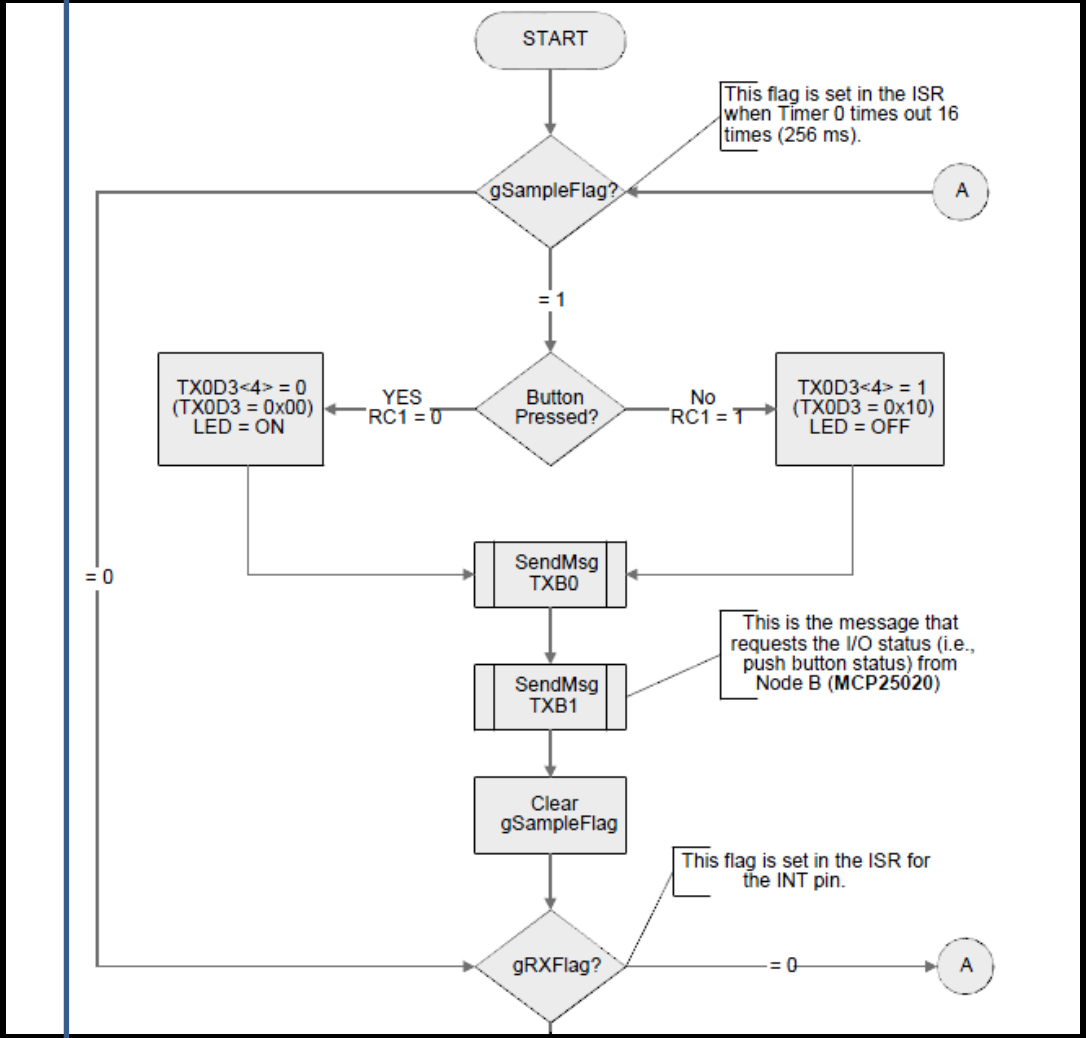
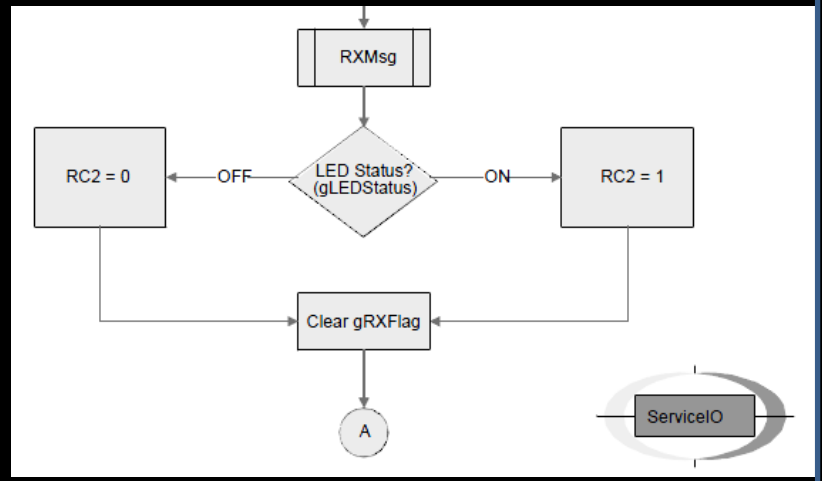


**FIGURE 2-3: SERVICE I/O FIRMWARE FLOW**



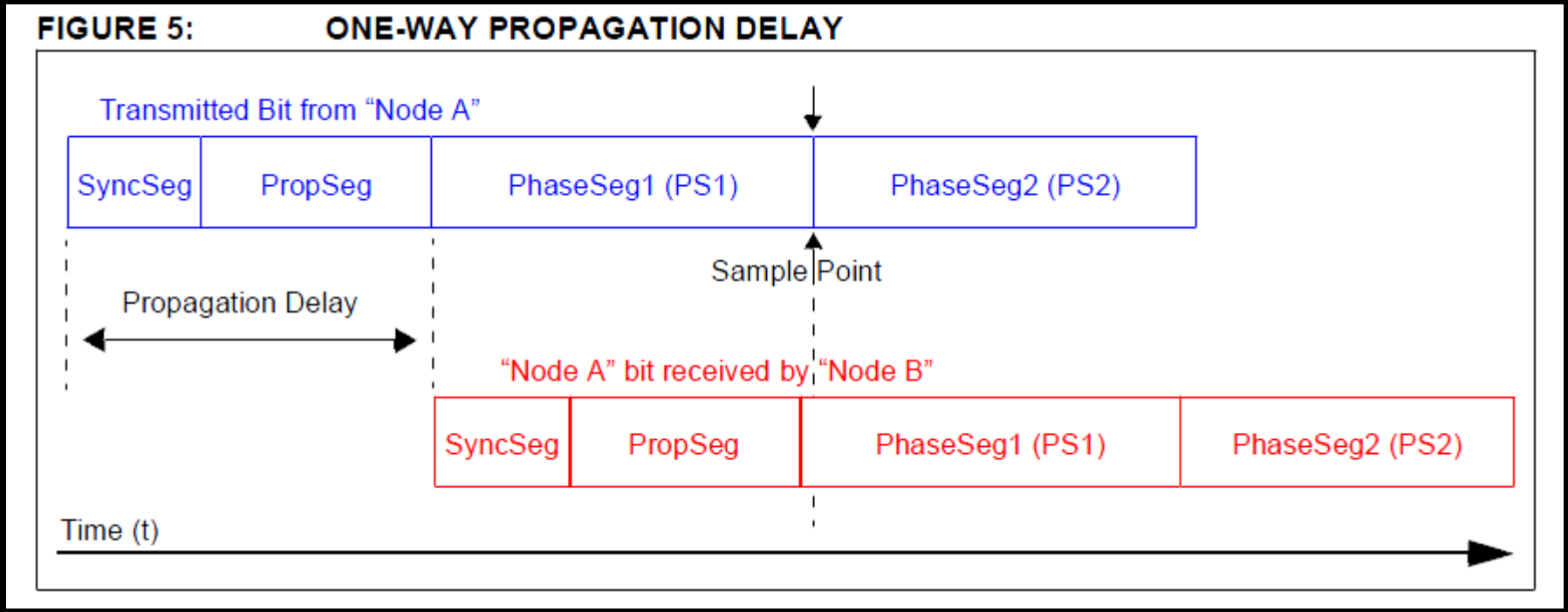



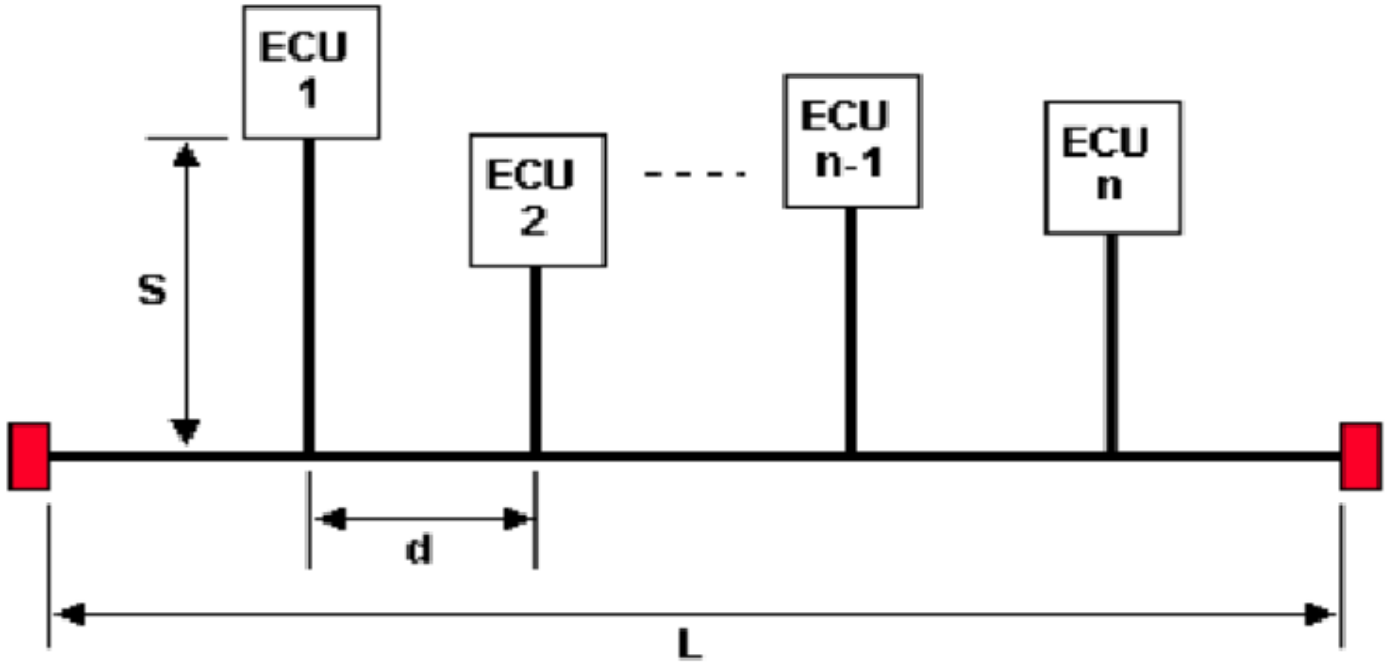
## Comprimento do Barramento

- A ISO 11 898 especifica que o tranceptor seja capaz de suportar um barramento de até 40 metros a uma taxa de 1 Mb/s.

- Um barramento mais longo pode ser alcançado reduzindo-se a taxa de transmissão. O maior limitação do comprimento do barramento é o atraso de propagação dos dados.

- O tempo de propagação é calculado por: $\mathbf{tp = 2.(t_{bus} + t_{comp} + t_{drv})}$

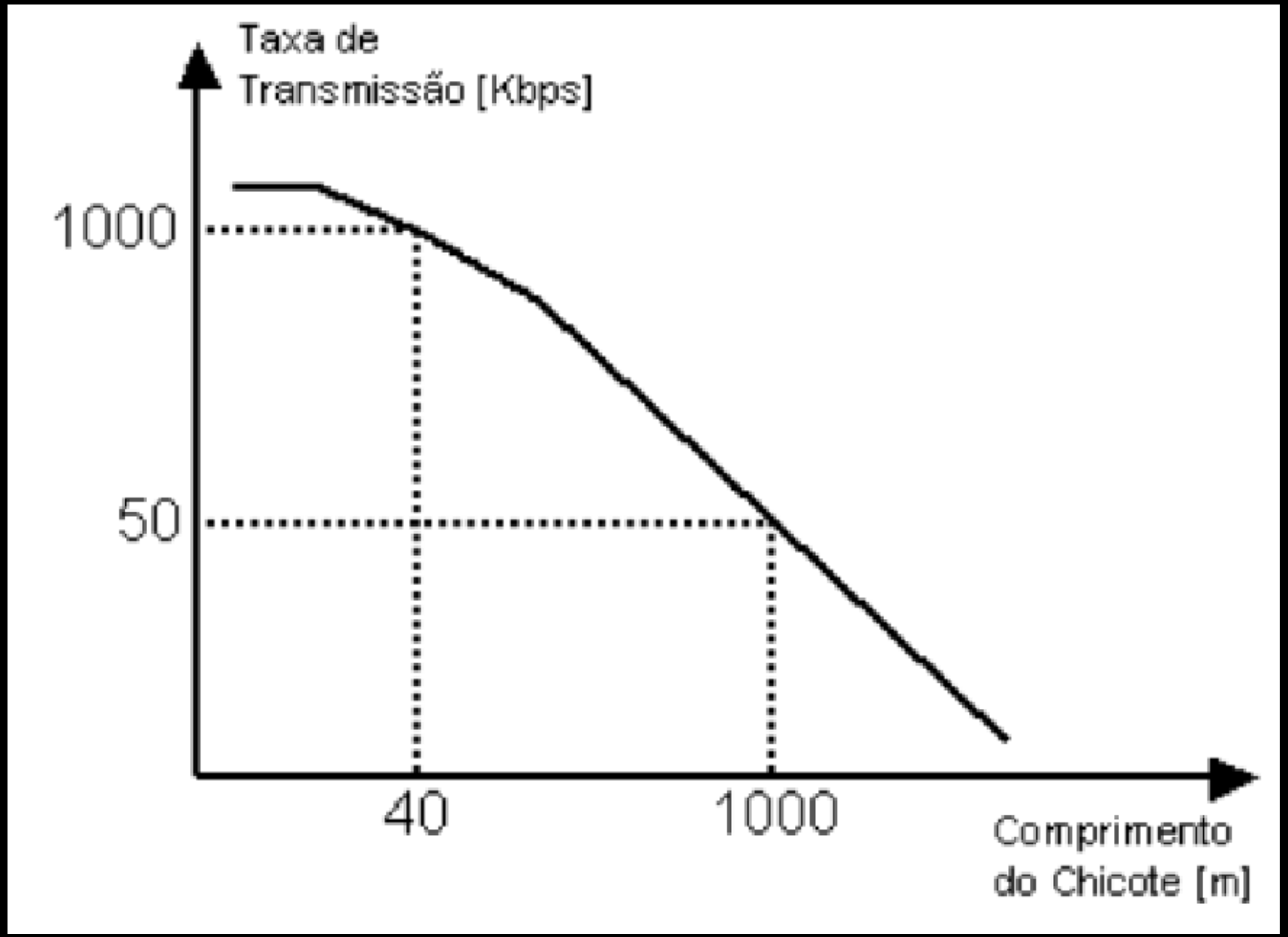





**Onde:** S (máximo comprimento da ramificação) = 0,3m
d (mínima distância entre ramificações) = 0,1m
L (máximo comprimento da rede a 1Mbps) = 40m

Obs: O valor da distância "d" deve ser aleatório.

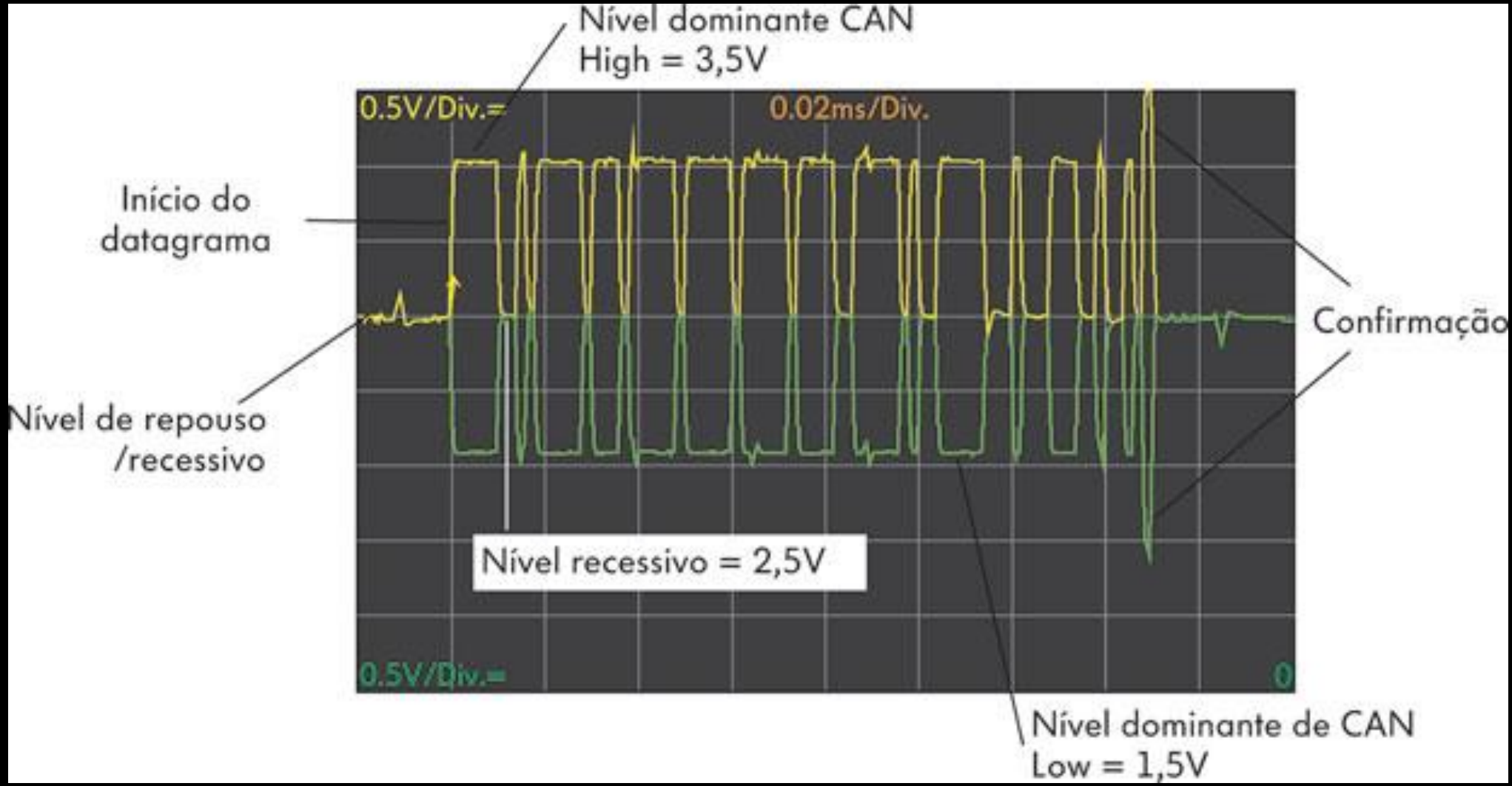
**Exemplo de Chicote**
(Alexandre Guimarães, GM)




(Alexandre Guimarães, GM)

## Can Bus Maximum Length

| Bus Speed | Bus Length (L) | Cable Stub Length (l) | Node Distance (d) |
|---|---|---|---|
| 1 Mbit/Sec | 40 meters (131 feet) | 0.3 meters (1 foot) | 40 meters (131.2 feet) |
| 500 kbits/Sec | 100 meters (328 feet) | 0.3 meters (1 foot) | 100 meters (328 feet) |
| 100 kbits/Sec | 500 meters (1640 feet) | 0.3 meters (1 foot) | 500 meters (1640 feet) |
| 50 kbits/Sec | 1000 meters (3280 feet) | 0.3 meters (1 foot) | 1000 meters (3280 feet) |




Através de um osciloscópio é possível capturar os sinais das redes CAN High e CAN Low e desta forma comprovar se os sinais estão plausíveis, comparando-os com os sinais padrões (tensões dominante e recessiva)

# Outras Redes:

**Table 2.** Grouping of selected automotive bus systems

| Subbus | Event-triggered | Time-triggered | Multimedia | Wireless |
|---|---|---|---|---|
| LIN | CAN | FlexRay | MOST | Bluetooth |
| K-Line | VAN | TTP | D2B | GSM |
| $I^2C$ | PLC | TTCAN | GigaStar | Wi-Fi |

Quanto custa ?

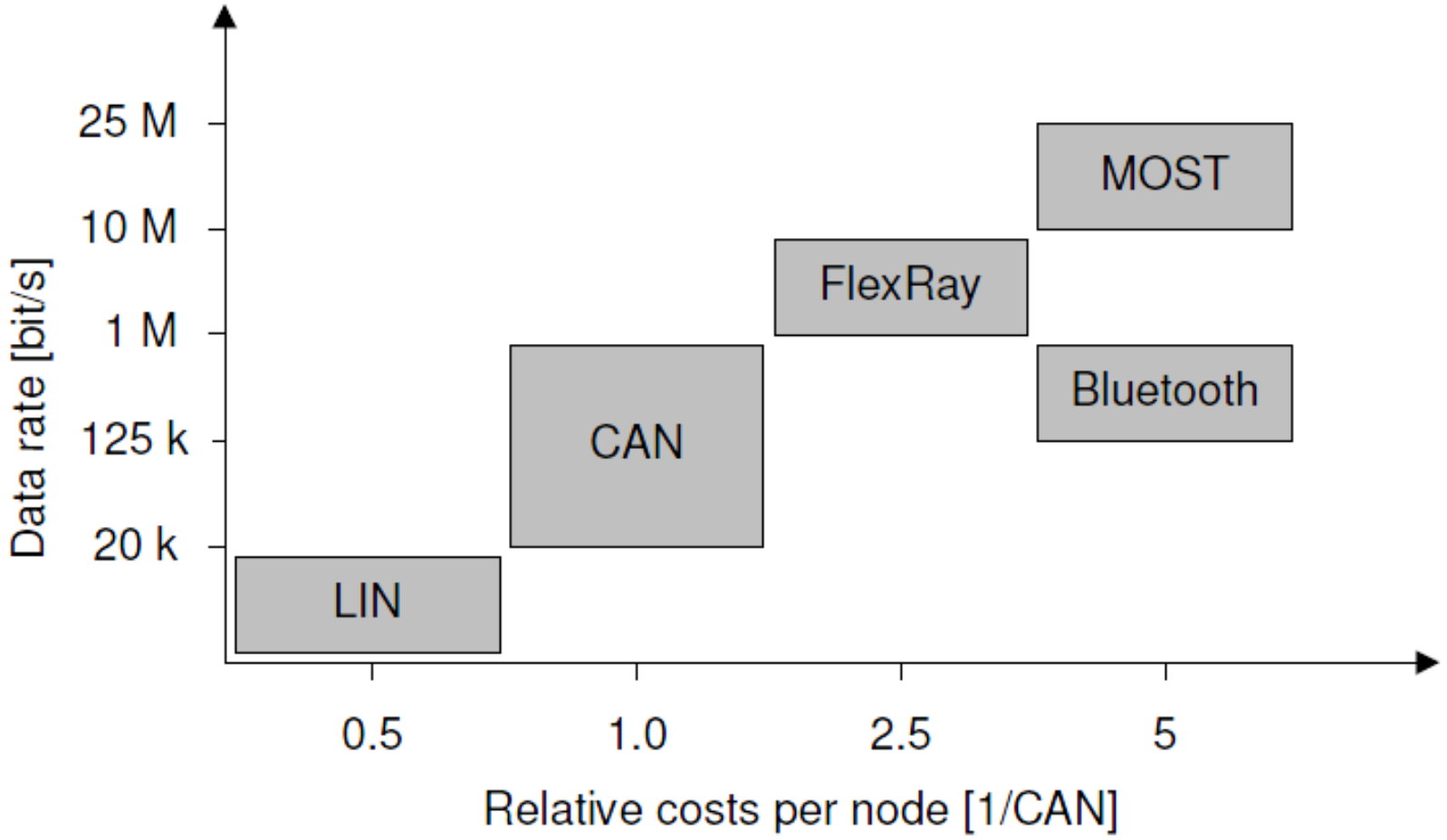



**Fig. 1.** Data rates and relative costs of automotive bus systems

**Table 3.** Properties of selected automotive bus systems

| Bus | LIN | CAN | FlexRay |
|---|---|---|---|
| **Adapted for** | Low-level subnets | Soft real-time | Hard real-time |
| **Target application examples** | Door locking<br>Climate regulation<br>Power windows<br>Light, rain sensor | Antilock break system<br>Driving assistants<br>Engine control<br>Electronic gear box | Break-by-wire<br>Steer-by-wire<br>Shift-by-wire<br>Emergency systems |
| **Architecture** | Single-master | Multi-master | Multi-master |
| **Access control** | Polling | CSMA/CA | TDMA<br>FTDMA |
| **Transfer mode** | Synchronous | Asynchronous | Synchronous<br>Asynchronous |
| **Data rate** | 20 kbit/s | 1 Mbit/s | 10 Mbit/s |
| **Redundancy** | None | None | 2 Channels |
| **Error protection** | Checksum<br>Parity bits | CRC<br>Parity bits | CRC<br>Bus Guardian |
| **Physical layer** | Single-wire | Dual-wire | Dual-wire, Optical fiber |
| **Security** | None | None | None |

# FIM