CAN基础 交流

汇报人：李小雨　　2017/5/10

Contents

目 录

一、概述

概述

– CAN的起源

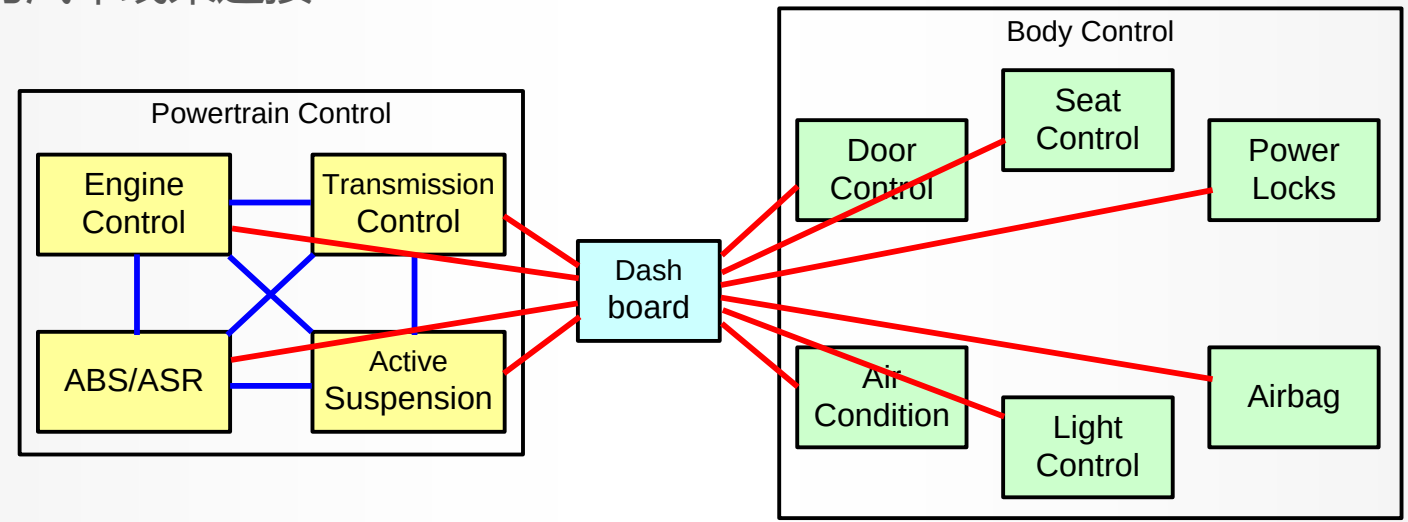
CAN—Controller Area Network

是20世纪80年代初德国Bosch公司为解决现代汽车中众多控制单元、测试仪器之间的实时数据交换而开发的一种串行通信协议

概述

– CAN的起源

✓ 传统的汽车线束连接

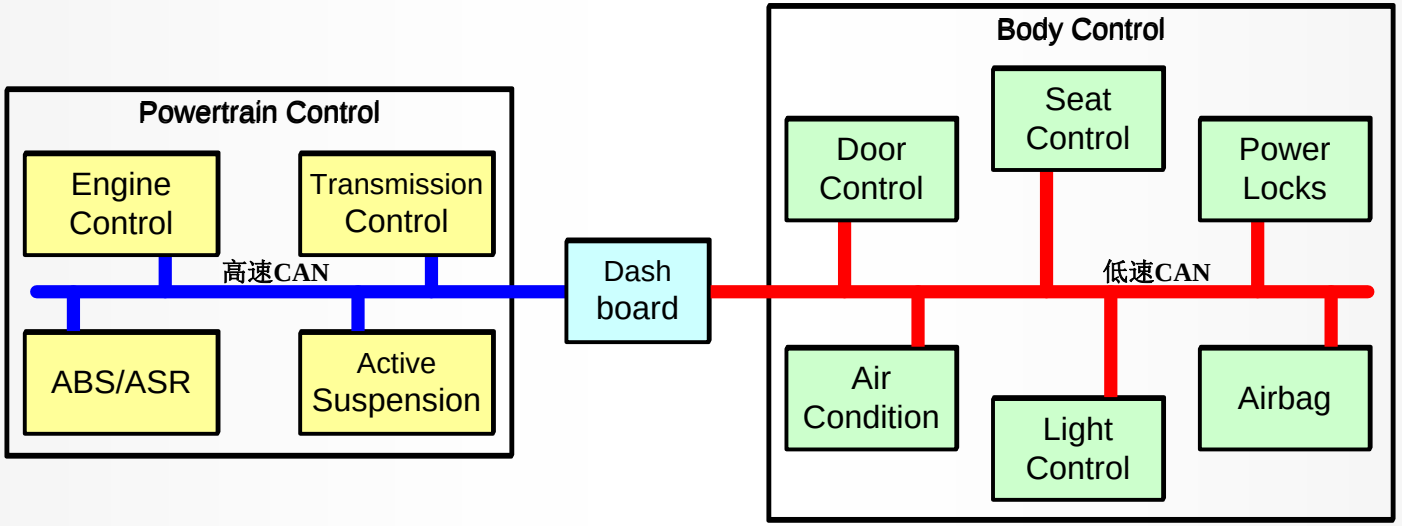


Powertrain control:动力系统控制
Dash board: 仪表盘
Airbag: 气囊
Transmission control:变速控制
Active suspension : 主动悬挂
ASR: 防滑驱动控制系统

概述

– CAN的起源

✓ 汽车的CAN网络



概述

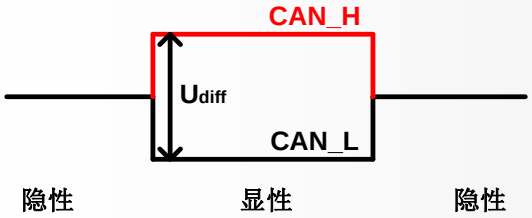
– CAN的历史

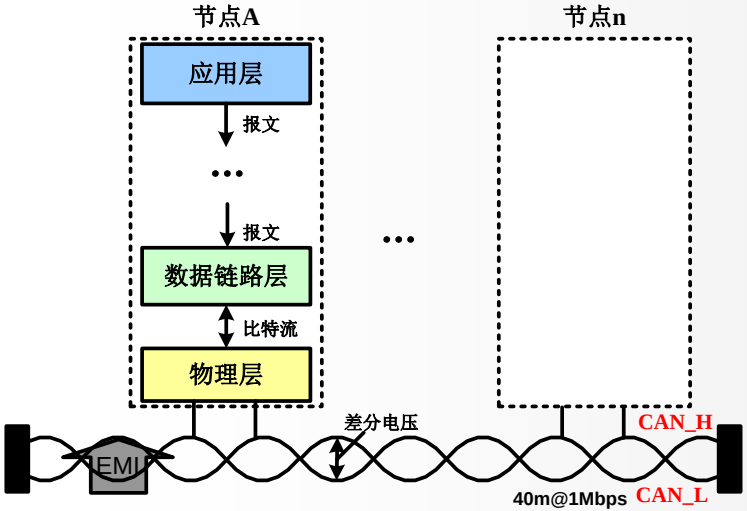
- ✓ 1983年，Bosch开始研究车上网络技术
- ✓ 1986年，Bosch在SAE大会公布CAN协议
- ✓ 1987年，Intel和Philips先后推出CAN控制器芯片
- ✓ 1991年，Bosch颁布CAN 2.0技术规范，CAN2.0包括A和B两个部分
- ✓ 1991年，CAN总线最先在Benz S系列轿车上实现
- ✓ 1993年，ISO颁布CAN国际标准 ISO-11898
- ✓ 1994年，SAE颁布基于CAN的J1939标准
- ✓ 2003年，VW发布带35个ECU的新型Golf……
- ✓ 未来，CAN总线将部分被FlexRay所取代，但CAN总线将仍会被持续应用相当长的时间

概述

– CAN的特性

- ✓ 采用双线差分信号
- ✓ 协议本身对节点的数量没有限制，总线上节点的数量可以动态改变
- ✓ 广播发送报文，即报文可以被所有节点同时接收

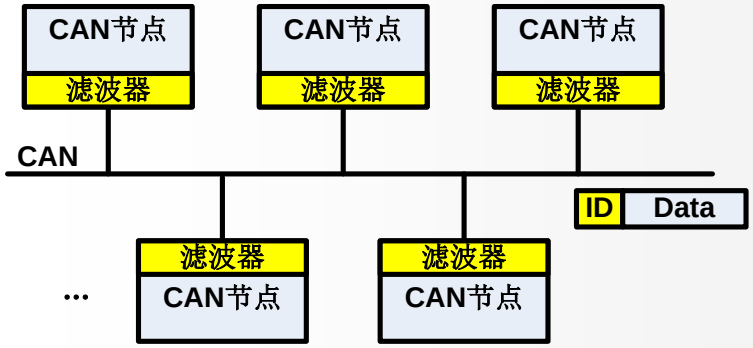




概述

– CAN的特性

- ✓ 多主站结构，各节点平等，
  优先权由报文ID确定
- ✓ 每个报文的内容通过标识符识别，
  标识符在网络中是唯一的
  - – 标识符描述了数据的含义
  - – 某些特定的应用对标识符
    的分配进行了标准化
- ✓ 根据需要可进行相关性报文过滤



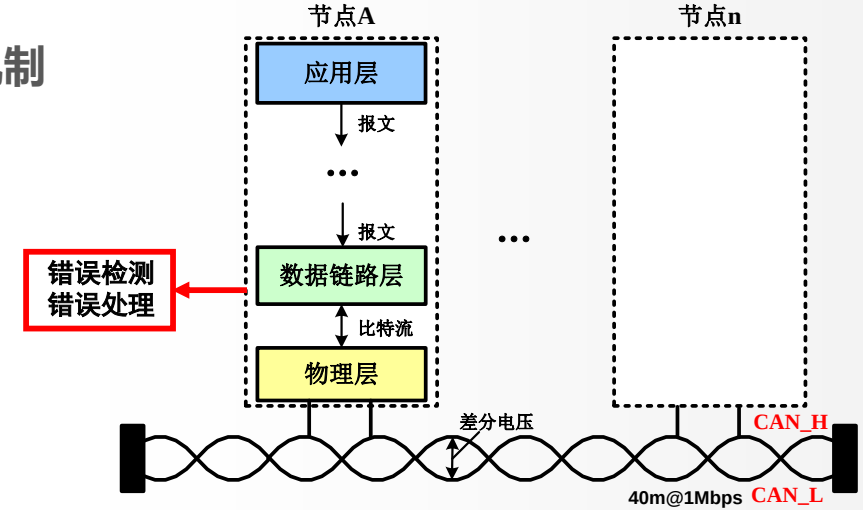
概述

– CAN的特性

保证系统数据一致性

CAN提供了一套复杂的错误检测与错误处理机制

- 比如CRC检测
- 接口的抗电磁干扰能力
- 错误报文的自动重发
- 临时错误的恢复
- 以及永久错误的关闭

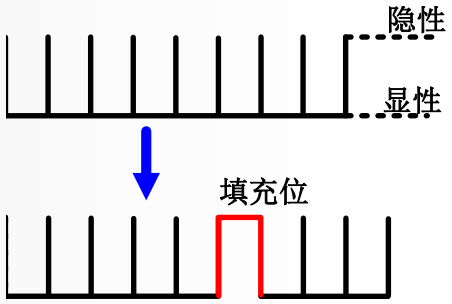


概述

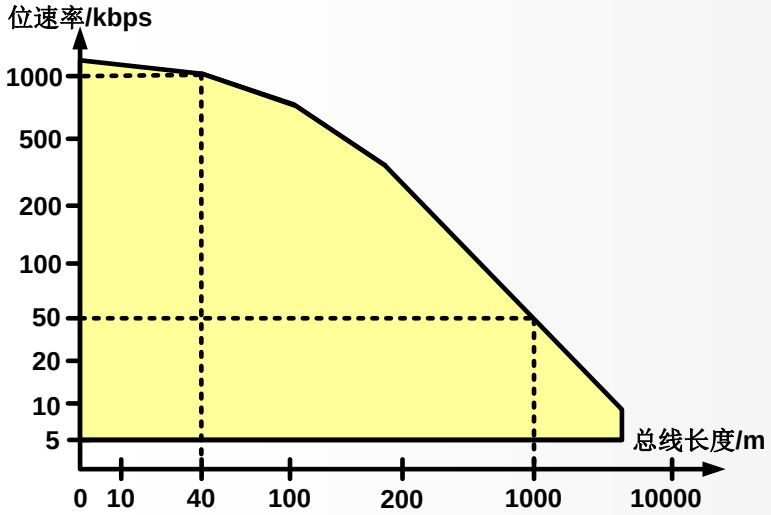
– CAN的特性

- ✓ 使用双绞线作为总线介质，传输速率可达1Mbps，总线长度<=40米,
- ✓ 采用NRZ和位填充
- ✓ 的位编码方式

NRZ和位填充



位速率与总线长度的关系



概述

– CAN的特性总结

- ✓ 载波侦听，网络上各个节点在发送数据前都要检测总线上是否有数据传输
- ✓ **网络上有数据➔**不发送数据，等待网络空闲
- ✓ **网络上无数据➔**立即发送已经准备好的数据
- ✓ 多路访问，网络上所有节点收发数据共同使用同一条总线，且发送数据是广播式的
- ✓ 冲突避免，节点在发送数据过程中要不停地检测发送的数据，确定是否与其它节点数据发生冲突

二、CAN总线标准

CAN总线标准

– OSI参考模型

底层标准化

BOSCH CAN2.0

ISO 11898

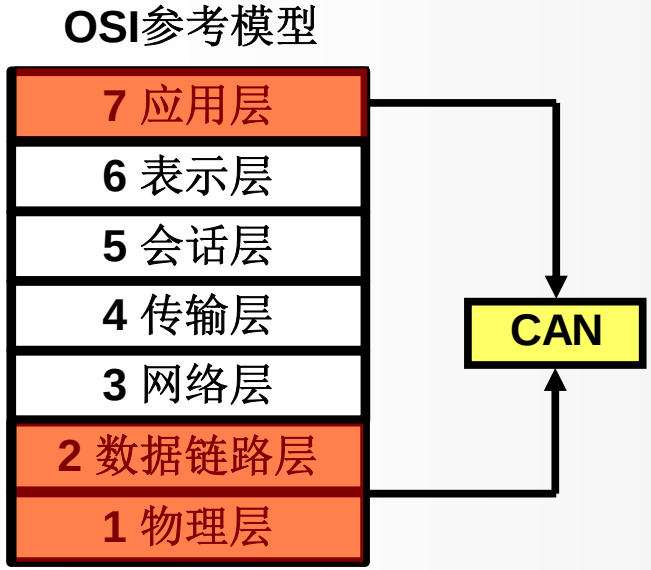
高层协议

CiA CANopen

SAE J1939

用户可以自己定义

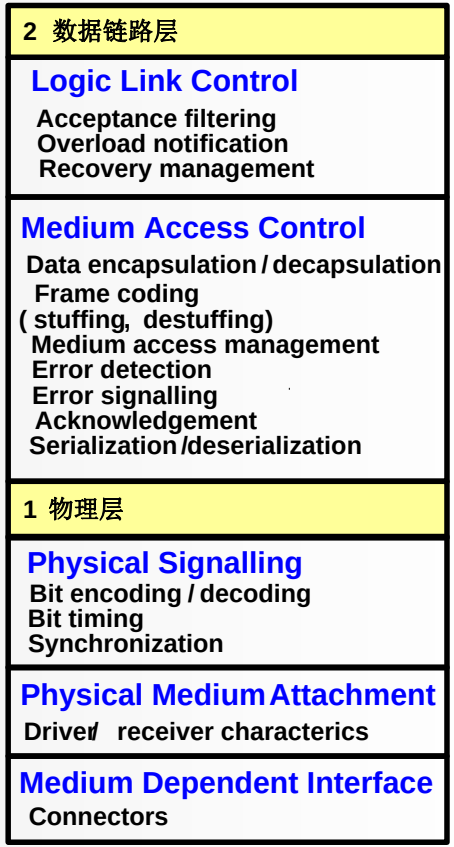
.....



开放系统互连参考模型
(Open System Interconnect 简称OSI)

CAN总线标准

– 各层执行功能



2 数据链路层	LLC	LLC, Logical Link Control 逻辑链路控制
	MAC	MAC, Medium Access Control 媒介访问控制
1 物理层	PLS	PLS, Physical Signaling Sublayer 物理信令子层
	PMA	PMA, Physical Medium Attachment 物理介质连接
	MDI	MDI, Medium Dependent Interface 介质相关接口

三、数据链路层

- 通信机制
- 数据帧
- 错误检测
- 帧格式

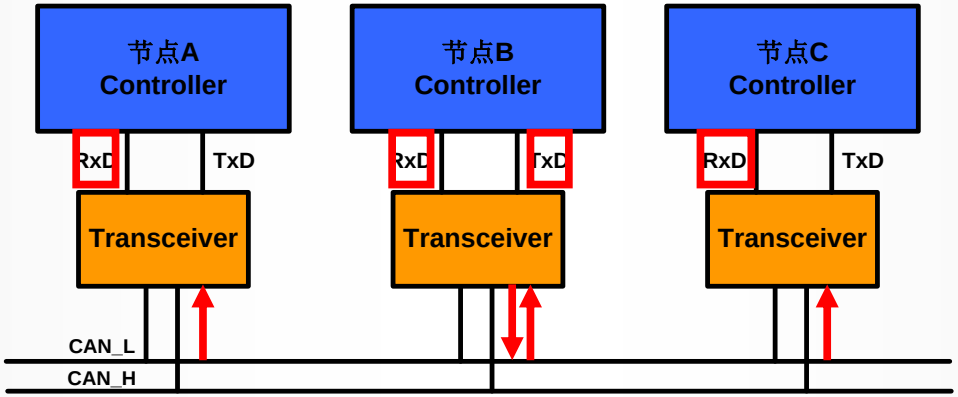
www.eontronix.com

通信机制

– 报文发送

节点发送报文时要检测总线状态

- ✓ 只有总线处于空闲，节点才能发送报文
- ✓ 在发送报文过程中进行“回读”，判断送出的位与回读的位是否一致

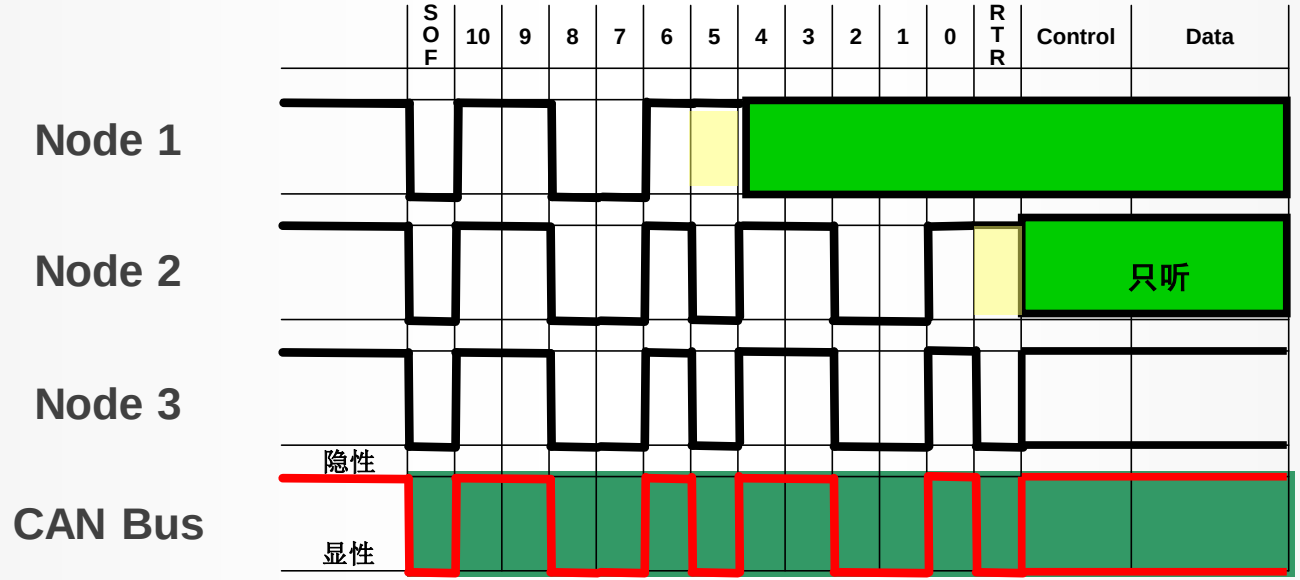


通信机制

– 报文发送

线与-机制

- ✓ 通过ID进行仲裁
- ✓ 显性位能够覆盖隐性位➔ID值越小，报文优先级越高

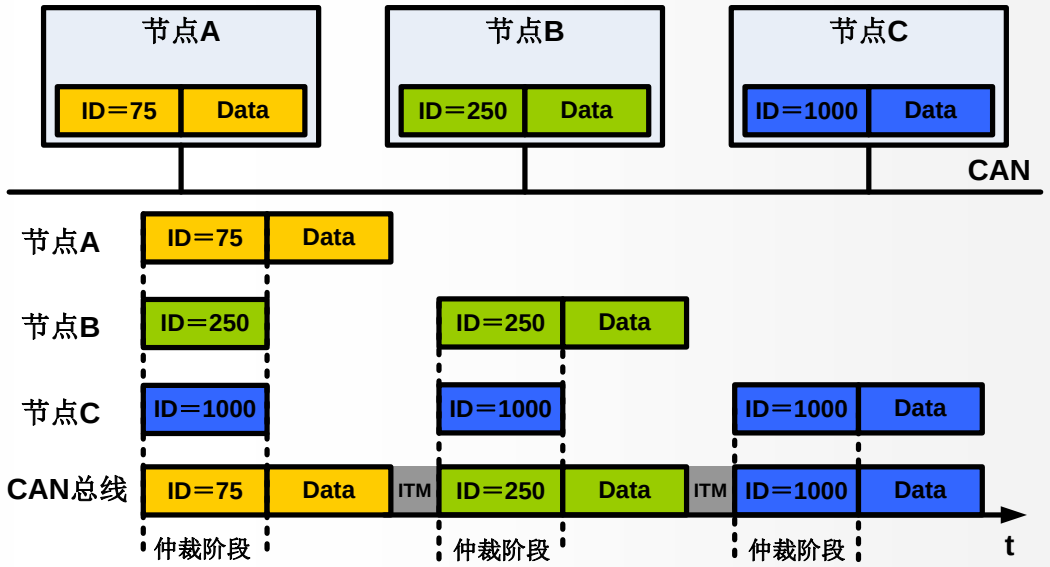


通信机制

– 报文发送

非破坏性仲裁

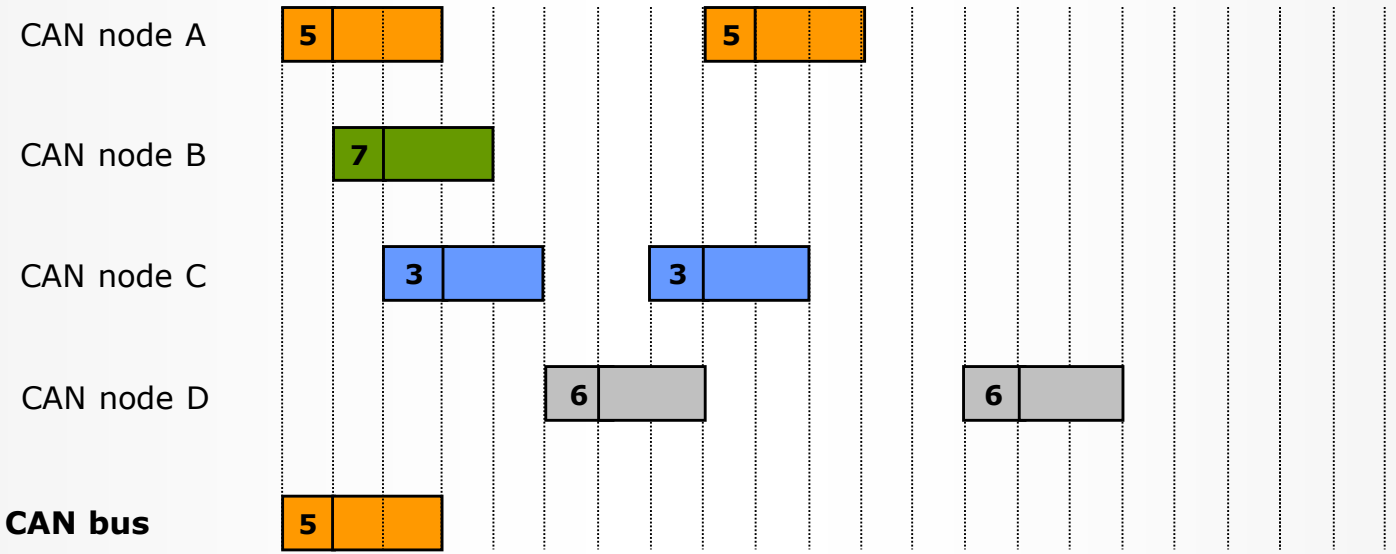
- ✓ 退出仲裁后进入“只听”状态
- ✓ 等待总线再次空闲时进行报文重发



通信机制

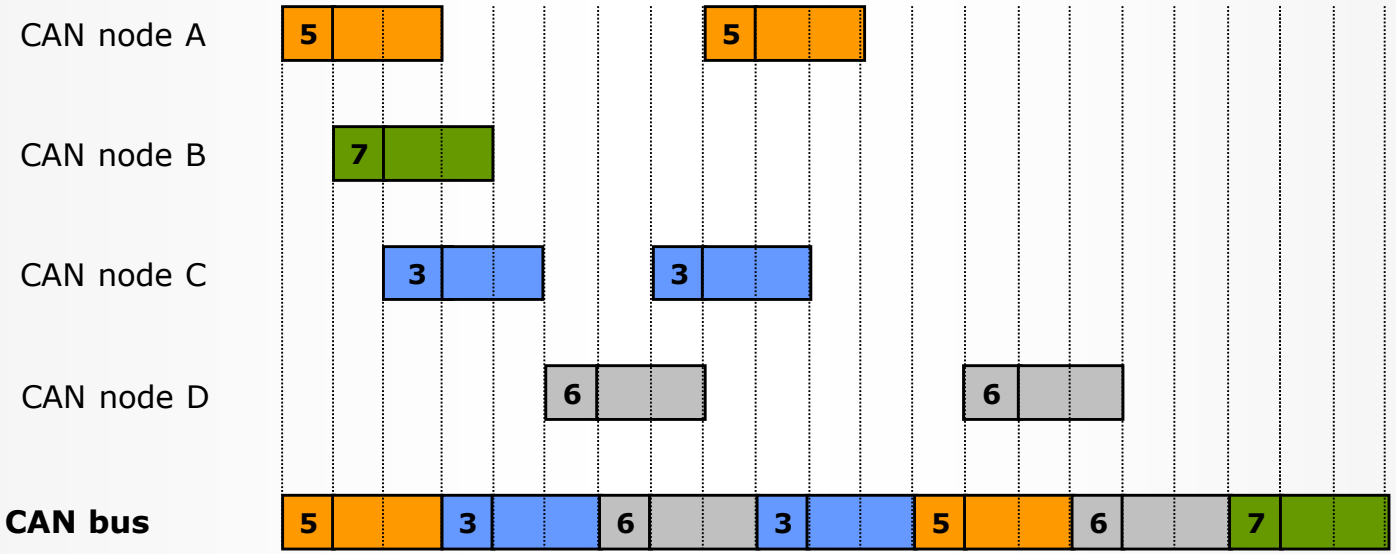
练习1：CAN总线访问仲裁机制

如图所示，A、B、C、D四个节点在不同的时刻分别往总线上发送ID为5、7、3、6的消息。请画出消息在总线上出现的顺序（假设每帧报文的传输时间占3格）。



通信机制

练习1答案：CAN总线访问仲裁机制



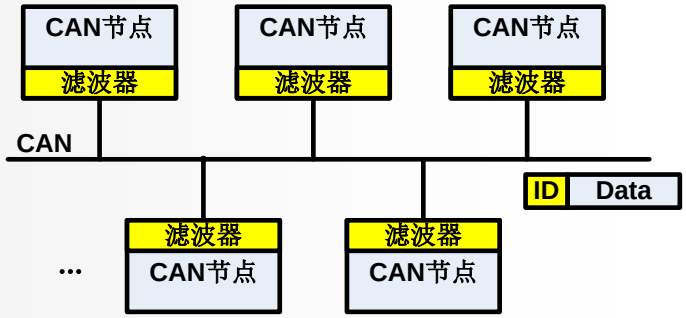
通信机制

– 报文接收过滤

通过滤波器，节点可以对接收的报文进行过滤

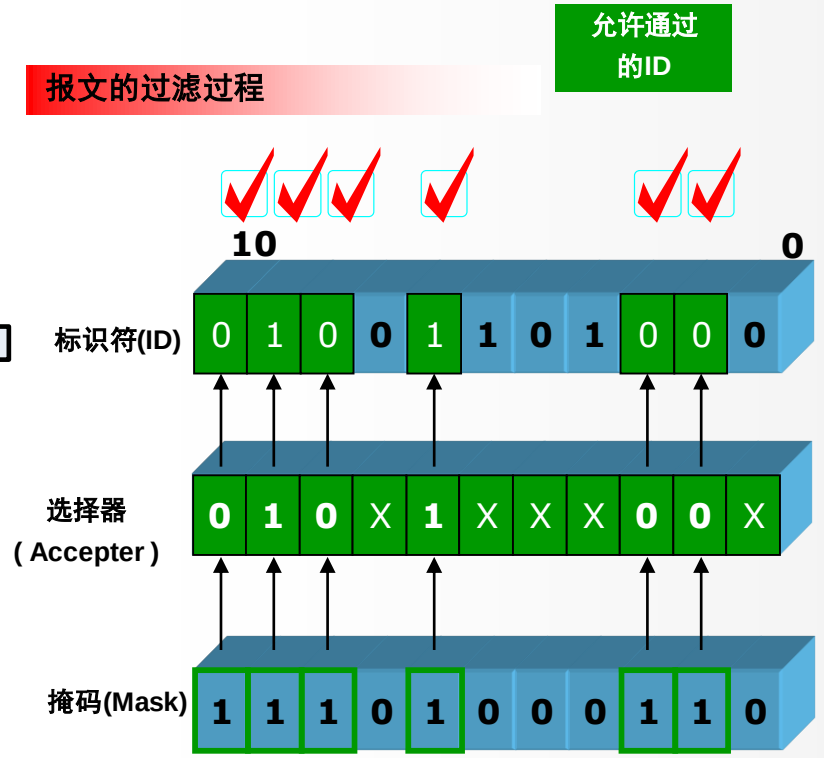
- 如果相关➔接收；
- 如果不相关➔过滤

接收滤波器(Acceptance Filter)



- 接收规则：比较消息ID与选择器中和接收过滤相关位是否相同
- 接收过滤相关位：由掩码定义
  - 1 = 与消息过滤有关
  - 0 = 与消息过滤无关

报文的过滤过程

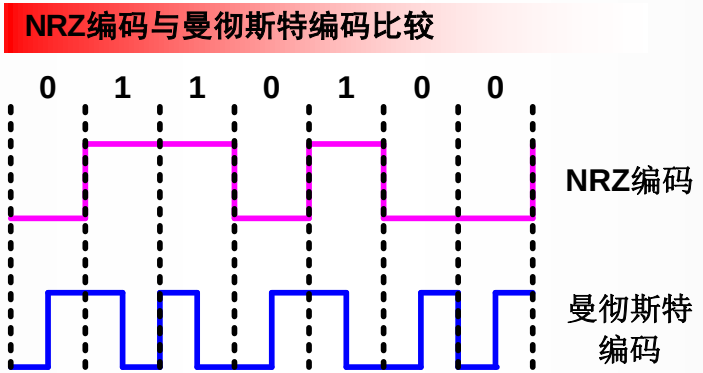


www.eontronix.com

– NRZ编码与位填充

NRZ（非归零）编码

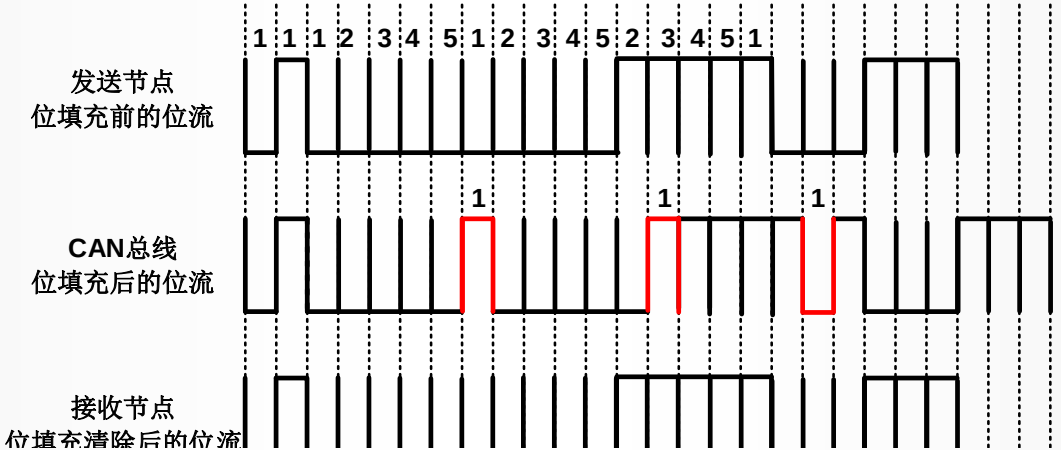
- ✓ NRZ编码确保报文紧凑➔在相同带宽情况下，NRZ编码方式的信息量更大
- ✓ NRZ不能保证有足够的跳变延用于同步，容易带来节点间计时器误差的累积
  ➔位填充➔保证有足够的跳变沿用于同步



通信机制

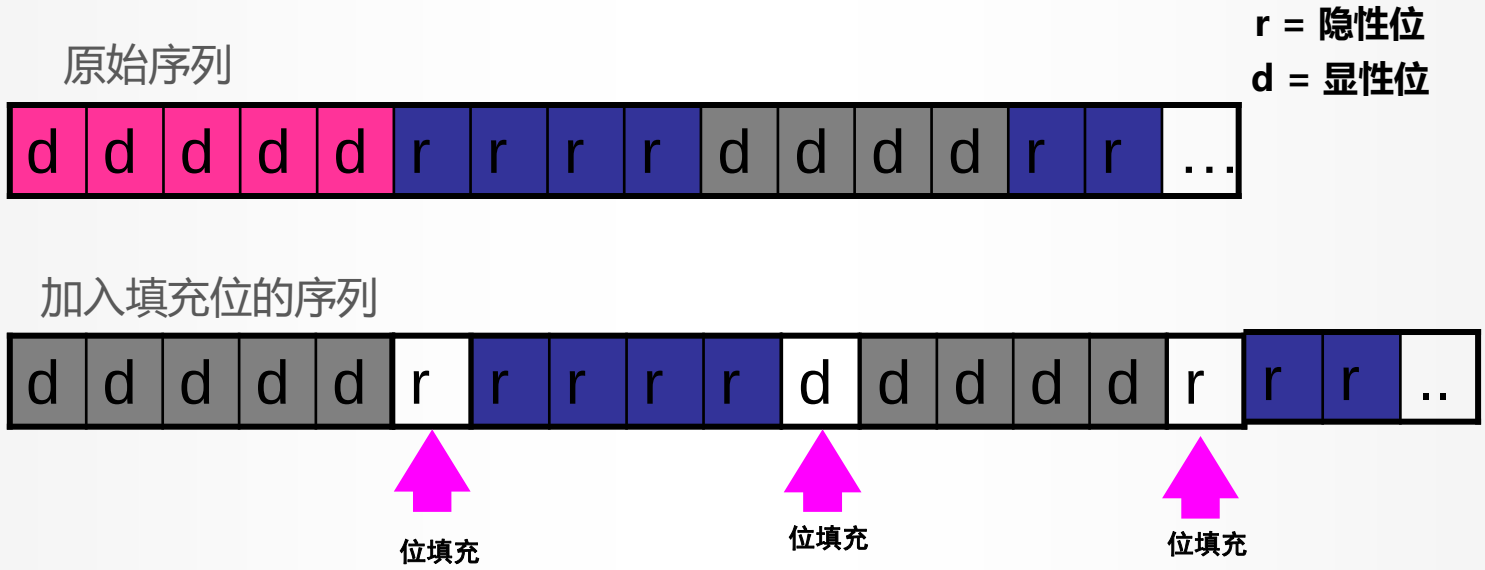
位填充

- ✓ 发送节点发送5个连续的相同极性位后，在位流中自动插入一个极性相反的位➔位填充
- ✓ 接收节点对相同极性位的数量进行检测，从位流中将填充位去掉➔清除填充



通信机制

– 位填充使总线传输效率降低



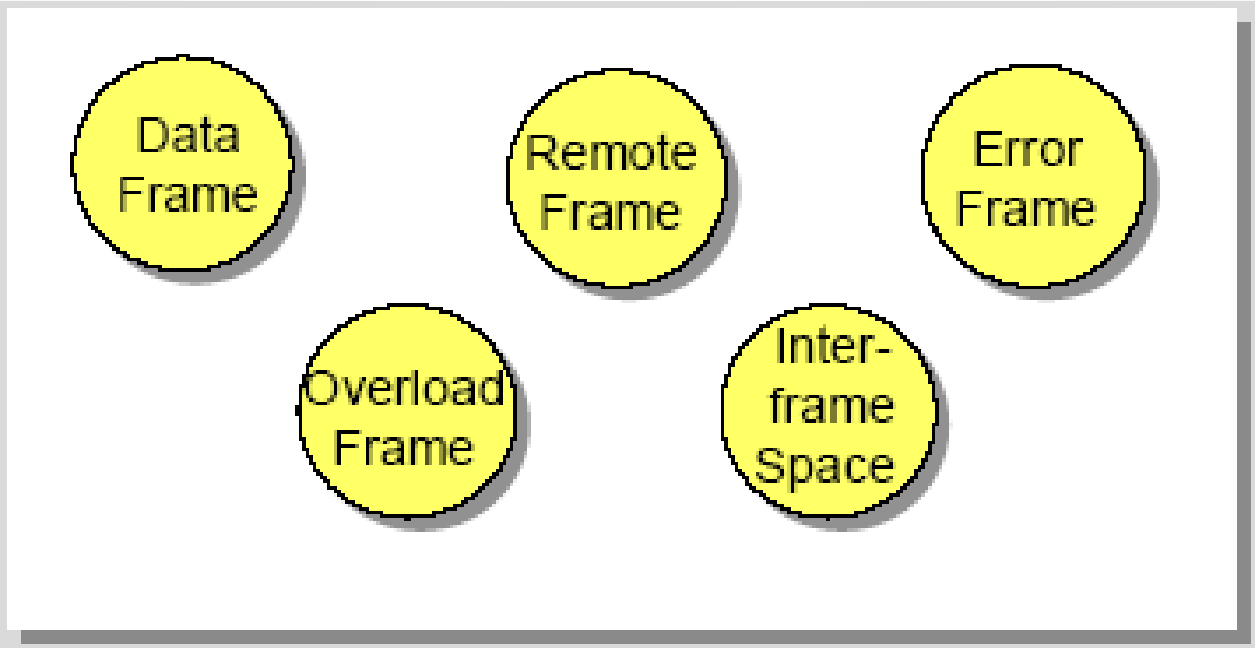
最坏情况下，每发送N位，最多可能插入
（N - 1）/4 个填充位！

三、数据链路层

- 通信机制
- 数据帧
- 错误检测
- 帧格式

数据帧

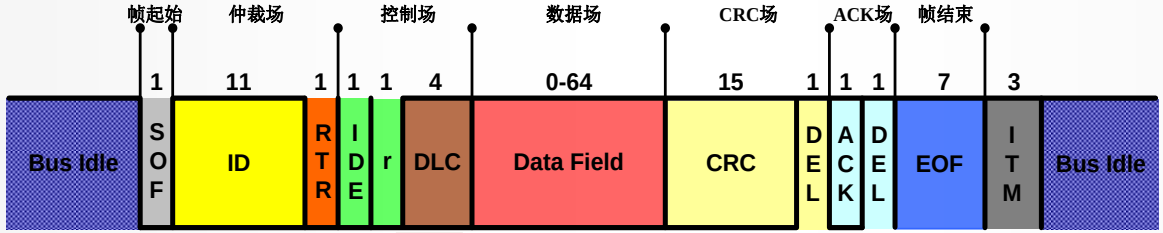
– 现有的帧格式


Data Frame
Remote Frame
Error Frame
Overload Frame
Inter-frame Space

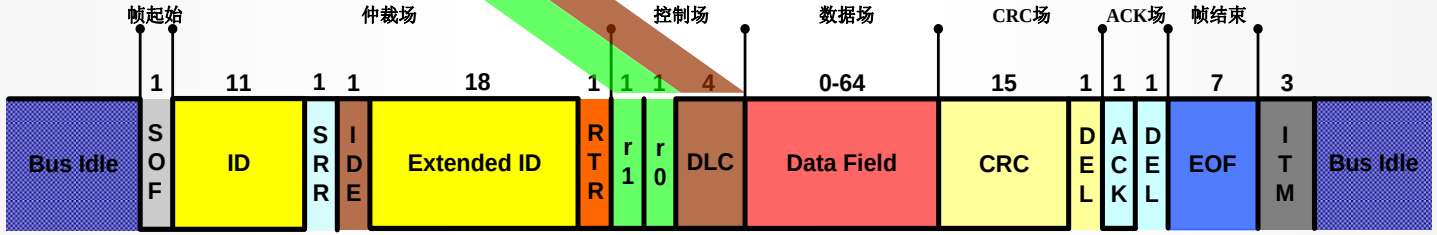
数据帧

– 数据帧的两种帧格式

标准帧



扩展帧

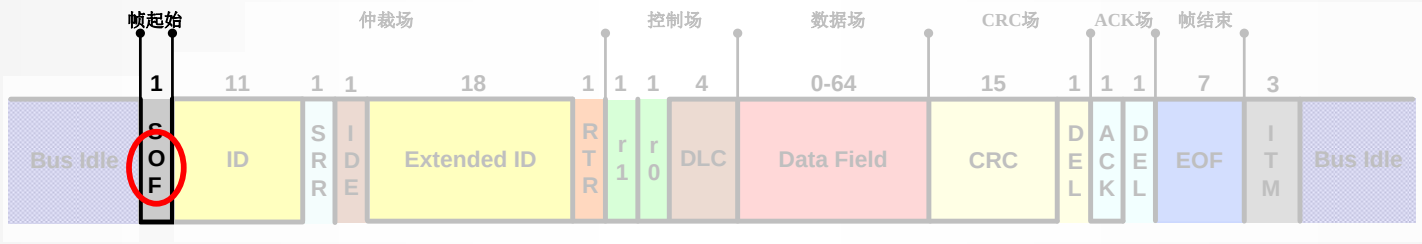


数据帧

– 缩写

缩写	中文全称	英文全称
SOF	帧起始	Start of Frame
RTR	远程发送请求	Remote Transmission Request
IDE	标识符扩展	Identifier Extension
DLC	数据长度代码	Data Length Code
ACK	应答	Acknowledgement
EOF	帧结束	End of Frame
ITM	间歇场	Intermission
SRR	代替远程请求	Substitute Remote Request
DEL	界定符	Delimeter

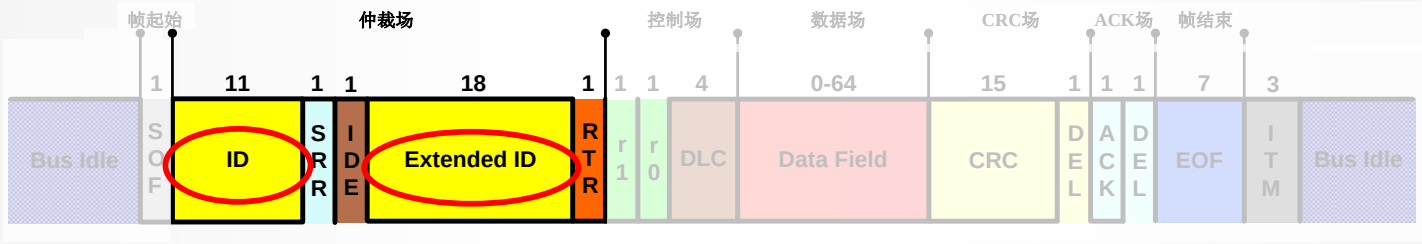
数据帧



帧起始

- ✓ 标识一个数据帧的开始，用于同步
- ✓ 一个显性位
- ✓ 只有在总线空闲期间节点才能够发送SOF

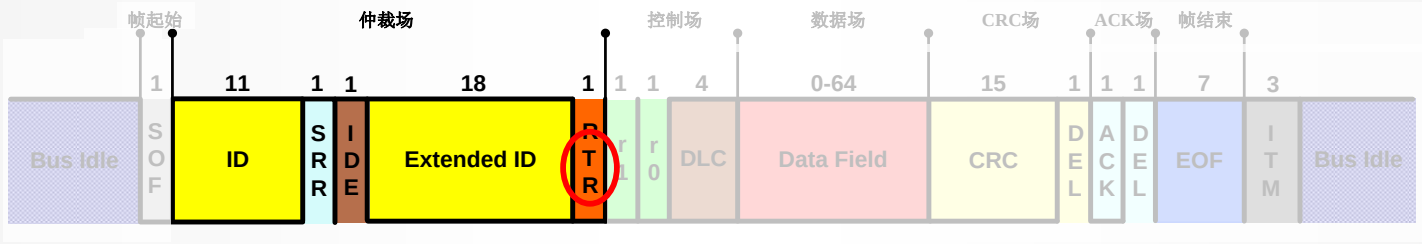
数据帧



ID　唯一确定一条报文

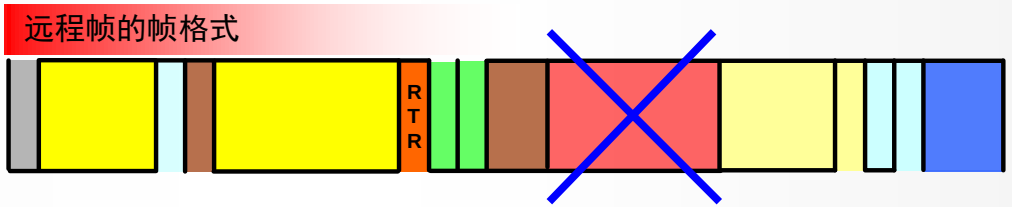
- ✓ 表明报文的含义，可以包含报文的源地址和目标地址
- ✓ 确定报文的仲裁优先级，ID数值越小，优先级越高
- ✓ 标准帧➔11位；扩展帧➔29位

数据帧

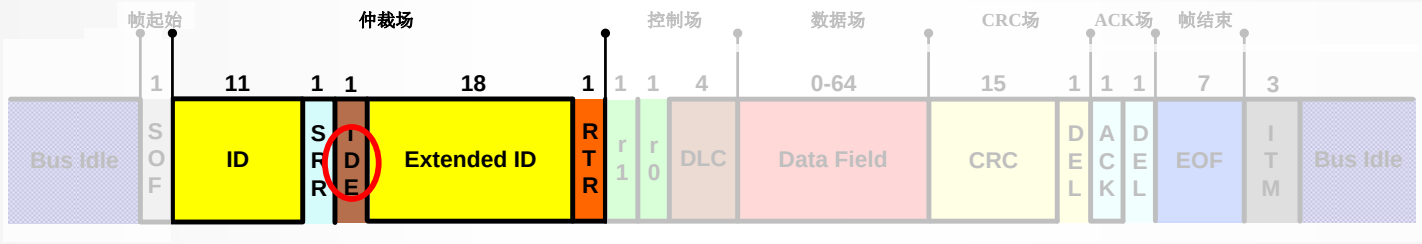


RTR位用于区分数据帧和远程帧

- ✓ 数据帧，RTR=0；
- ✓ 远程帧，RTR=1；



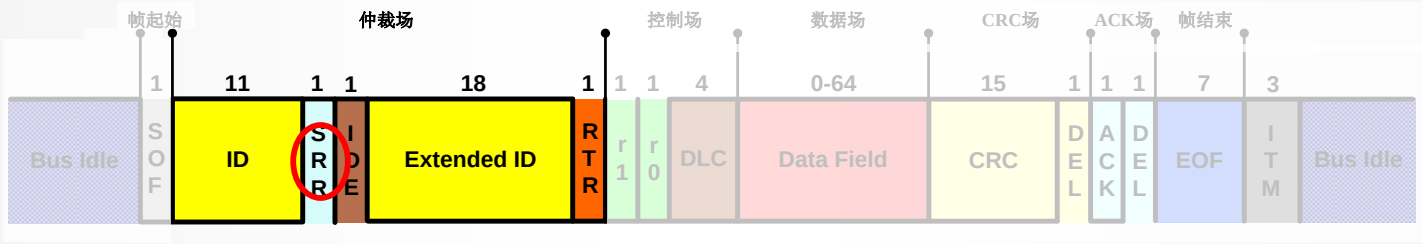
数据帧



IDE位 用于区分标准帧和扩展帧

- ✓ 标准帧，IDE=0（11位ID）
- ✓ 扩展帧，IDE=1（29位ID）

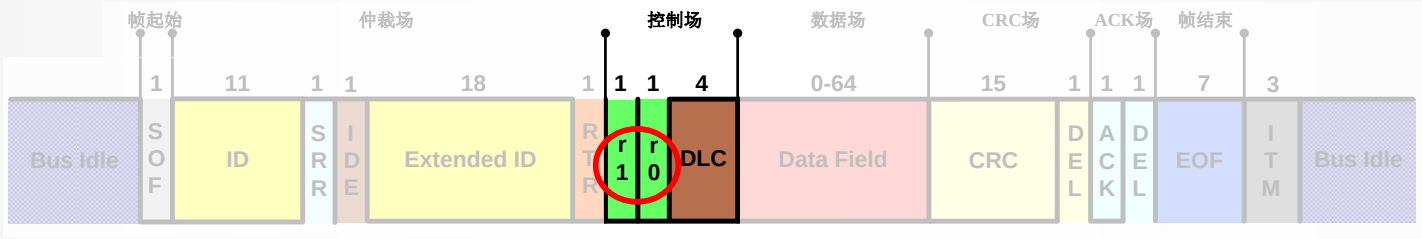
数据帧



SRR位表明在该位代替了标准帧中的RTR

- ✓ 该位无实际意义
- ✓ SRR永远置1

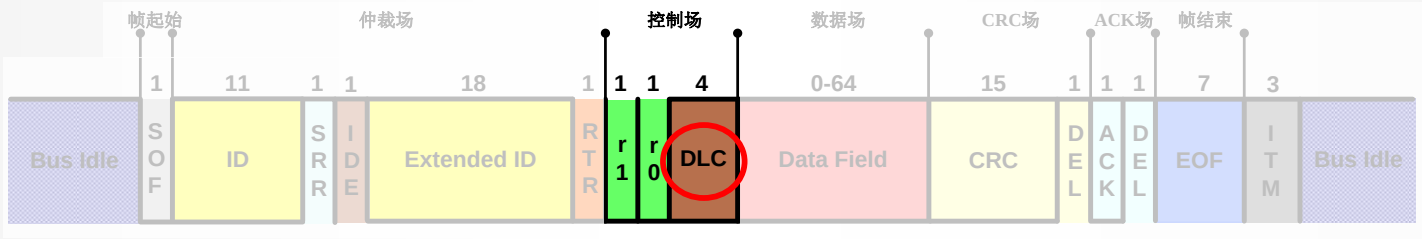
数据帧



r0、r1位

- ✓ 两个保留位
- ✓ 当前置0

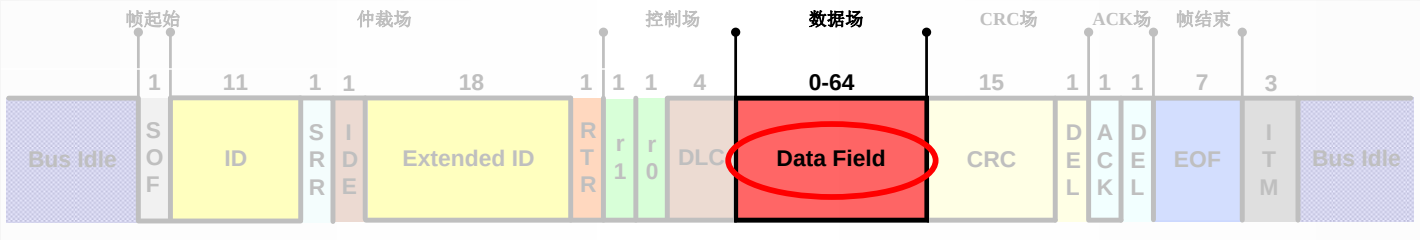
数据帧



DLC包含4位，表示数据场包含数据的字节数

- ✓ DLC=0-8
- ✓ DLC=9-15➔DLC=8

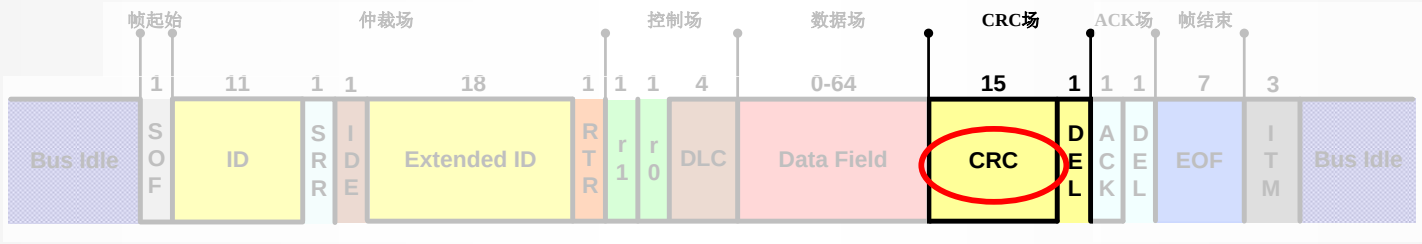
数据帧



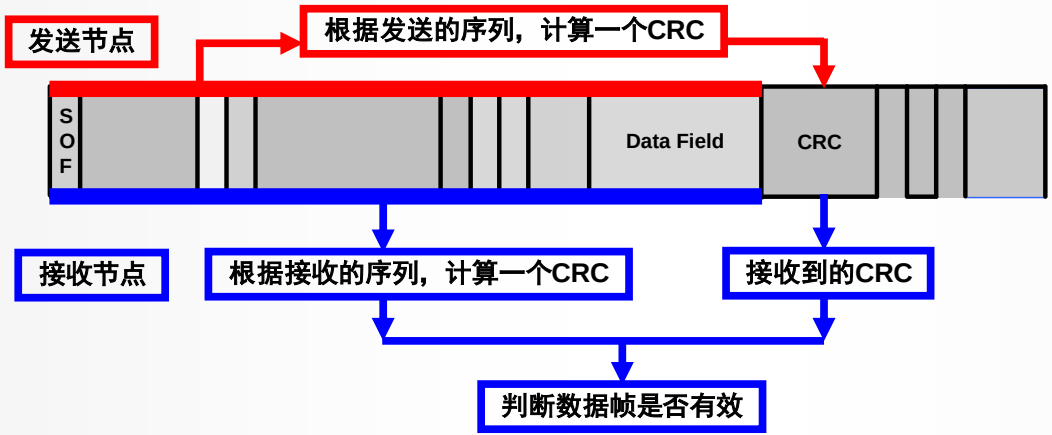
数据场

- ✓ 具有0-8个字节长度，由DLC确定
- ✓ 包含CAN数据帧发送的内容

数据帧

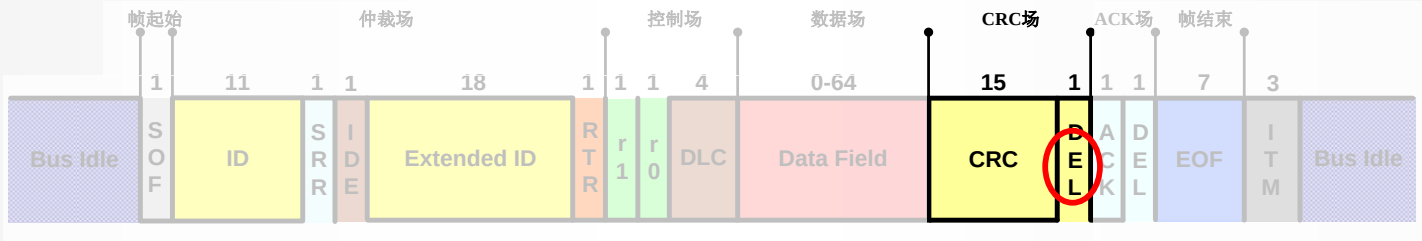


CRC用于进行CRC校验



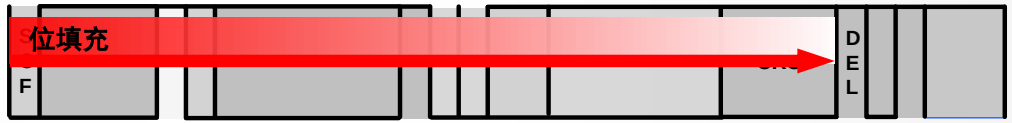
www.eontronix.com

数据帧

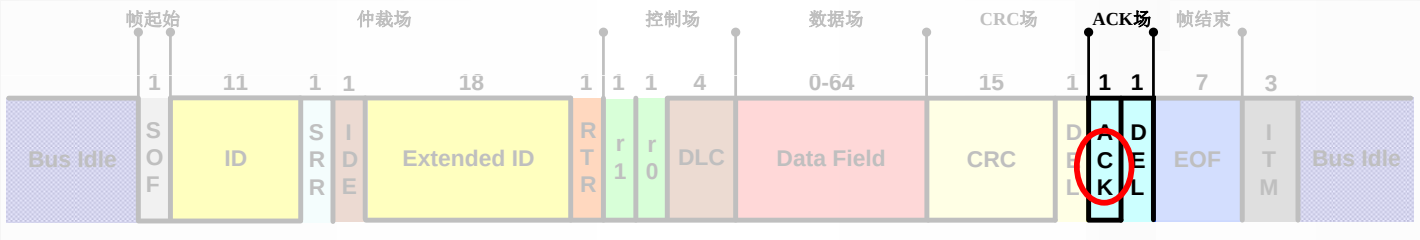


CRC界定符

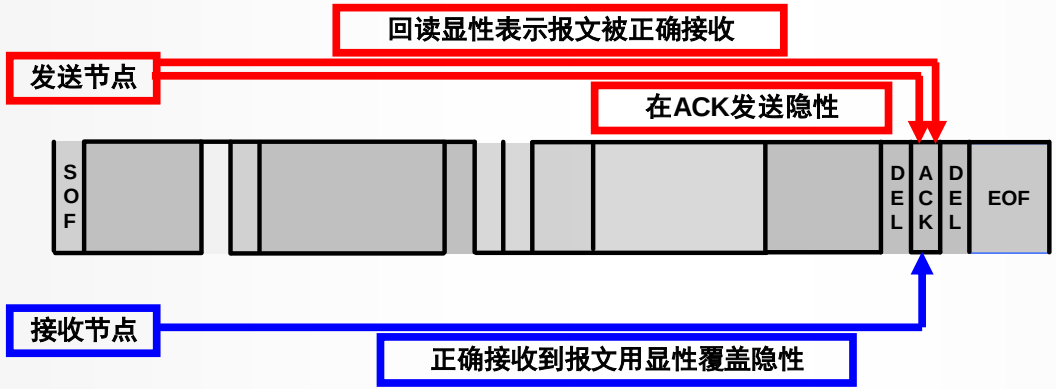
- ✓ 界定CRC序列
- ✓ 固定格式，1个隐性位
- ✓ CRC界定符之前进行位填充



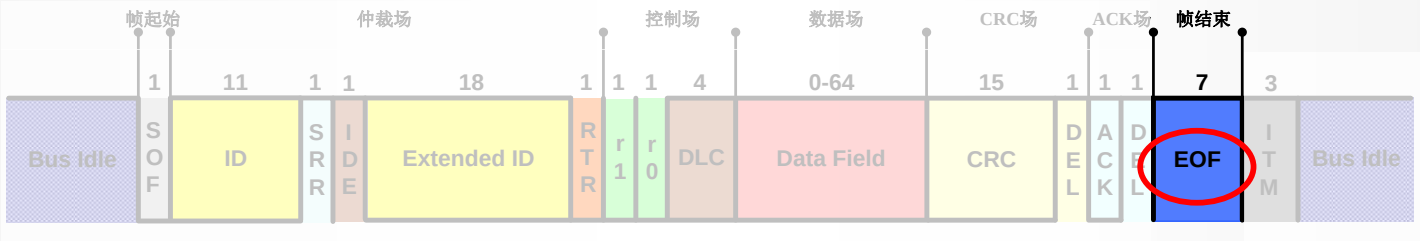
数据帧



ACK确定报文被至少一个节点正确接收



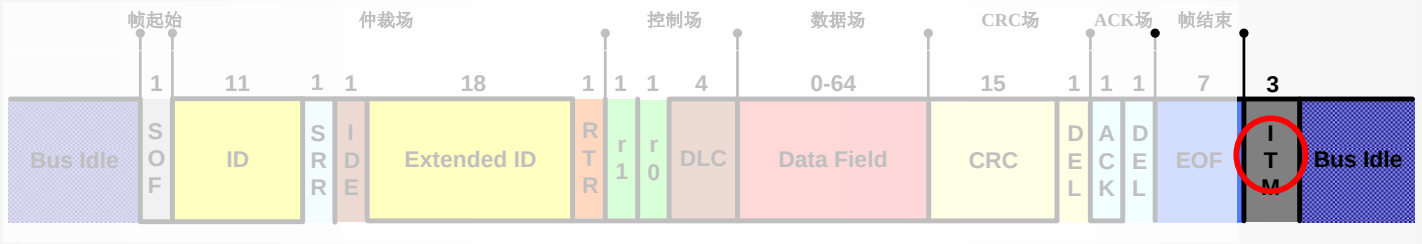
数据帧



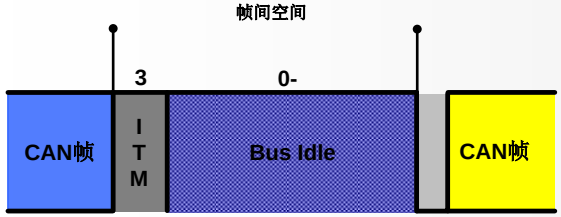
EOF

- ✓ 表示数据帧结束
- ✓ 固定格式，7个连续的隐性位

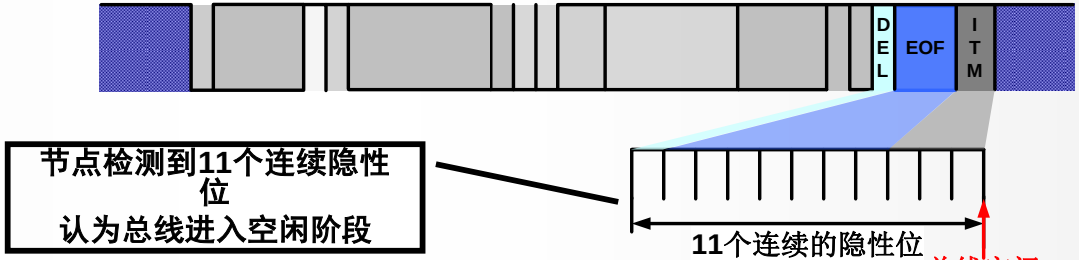
数据帧



ITM(间歇场)

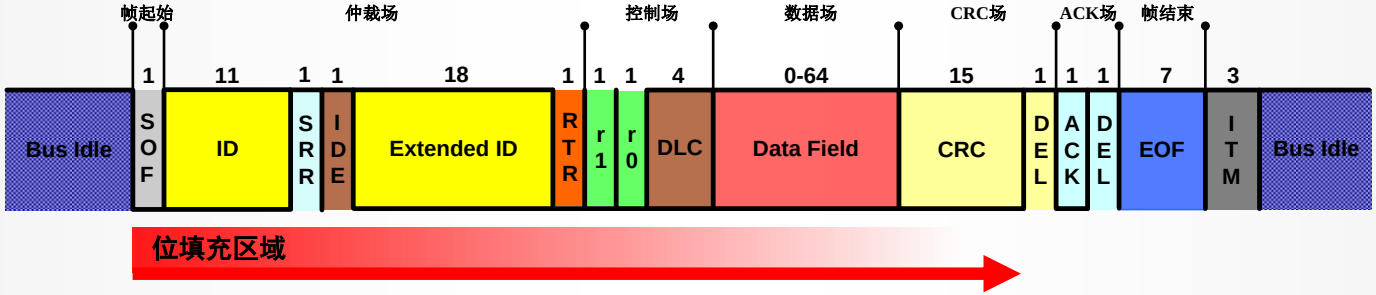


- ✓ 固定格式
- ✓ 3个连续的隐性位
- ✓ ITM之后进入总线空闲状态，此时节点可以发送报文



www.eontronix.com

位填充区域



- SOF之前的总线空闲区域，不需要同步，无需进行位填充
- CRC之后的位域都是固定格式，不允许位填充操作

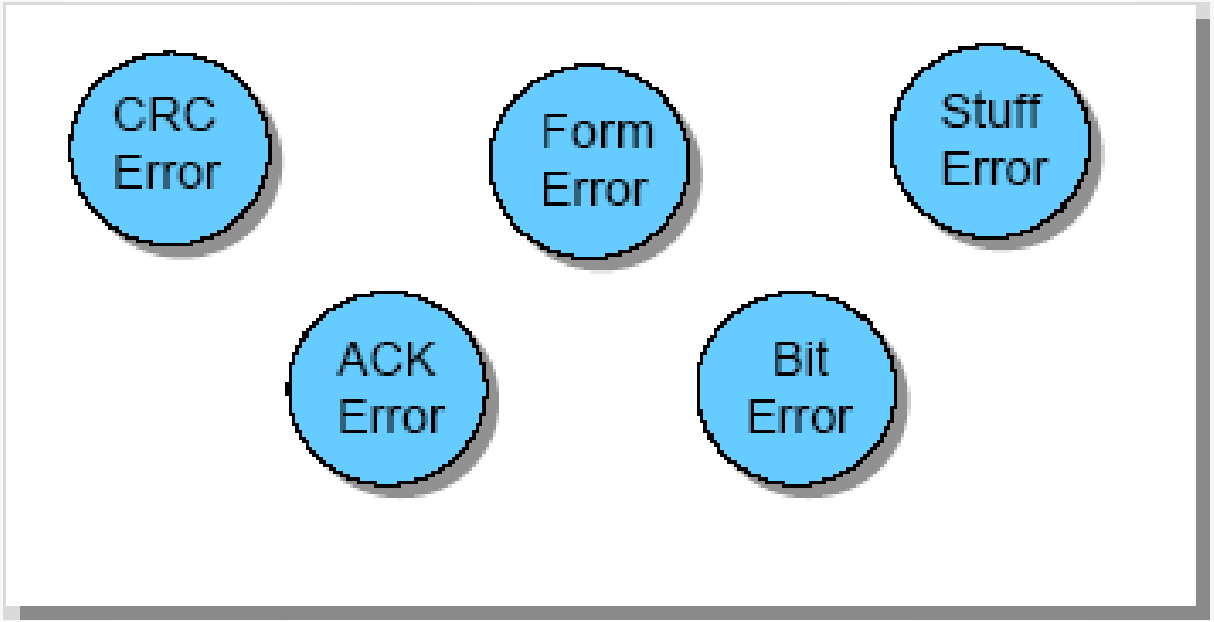
三、数据链路层

- 通信机制
- 数据帧
- 错误检测
- 帧格式

www.eontronix.com

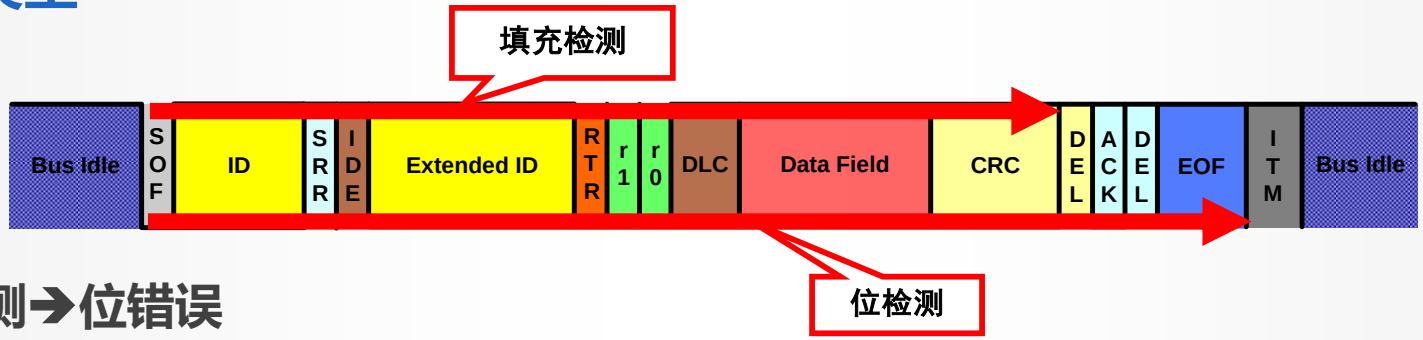
错误检测

– 可检测的错误



错误检测

– 错误类型



位检测➔位错误

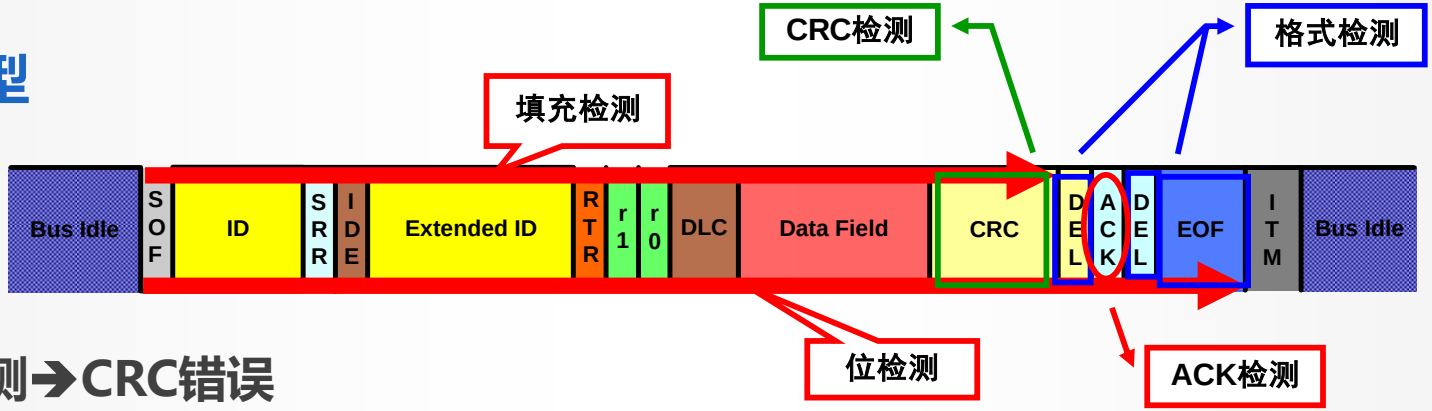
- ✓ 节点检测到的位与自身送出的位数值不同
- ✓ 仲裁或ACK位期间送出“隐性”位，而检测到“显性”位不导致位错误

填充检测➔填充错误

- ✓ 在使用位填充编码的帧场（帧起始至CRC序列）中，不允许出现六个连续相同的电平位

错误检测

– 错误类型



CRC检测➔CRC错误

✓ 节点计算的CRC序列与接收到的CRC序列不同

格式检测➔格式错误

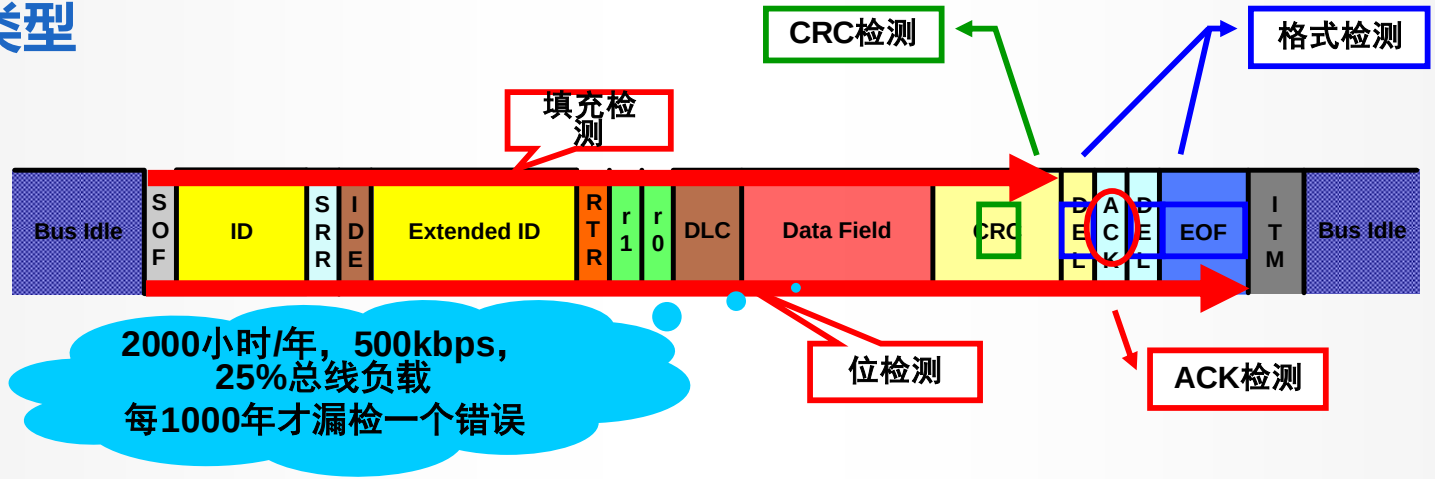
✓ 固定格式位场（如CRC界定符、ACK界定符、帧结束等）含有一个或更多非法位

ACK检测➔ACK错误

✓ 发送节点在ACK位期间未检测到“显性”位

错误检测

– 错误类型



发送节点➔ 位 错 误、格式错误、ACK错误

接收节点➔ 填充错误、格式错误、 CRC错误

检测到错误后，发送错误标志

✓ 位错误、填充错误、格式错误或ACK错误产生后➔错误标志在下一位发送

✓ CRC错误➔错误标志在ACK界定符后发送

错误检测

– 错误界定

- ✓ 每个节点都含有REC和TEC
- ✓ 当接收错误产生时，REC增加；
- ✓ 正确接收到数据帧，REC减少
- ✓ TEC类似
- ✓ REC、TEC的数值会引发节点状态改变

REC： Receive Error Counter 接收错误计数器
TEC： Transmit Error Counter 发送错误计数器

错误检测

– 节点的三种状态

Error Active

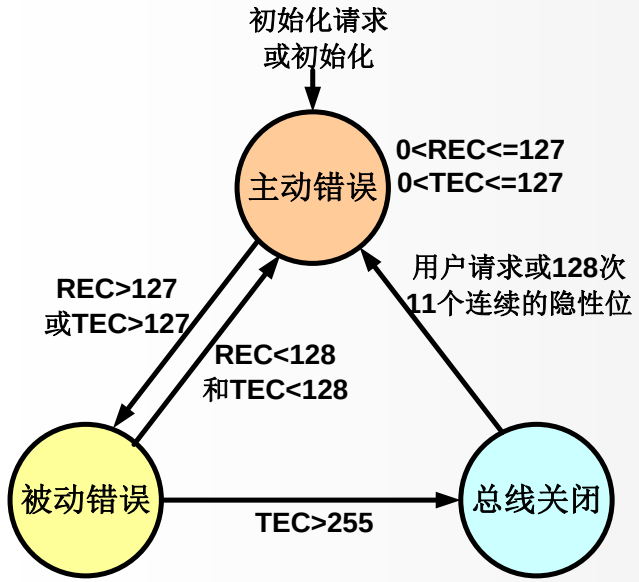
- ✓ 正常的进行总线通信
- ✓ 错误产生时，发送主动错误标志（6个连续显性位）

Error Passive

- ✓ 能够进行总线通信
- ✓ 限制（连续2次报文发送）
- ✓ 错误产生时，发送被动错误标志（6个连续隐性位）

Bus Off

- ✓ 不能收发任何报文



特例：如果总线上只有一个节点，该节点发送数据帧后得不到应答，TEC最大只能计到128即节点只会进入被动错误状态而不会进入总线关闭状态

www.eontronix.com

错误检测

– 错误帧的格式

包括错误标志与错误界定符

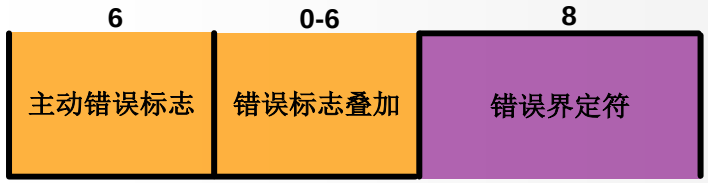
错误标志

- ✓ Active：6个显性位；
- ✓ Passive：6个隐性位
- ✓ 错误标志违背“位填充规则”，其他节点也会发送错误标志错误标志6-12位长度

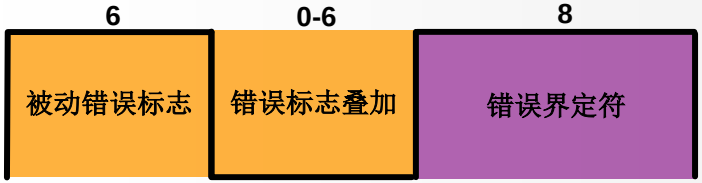
错误界定符

- ✓ 8个连续隐性位
- ✓ 节点发送错误标志后，发送隐性位，直至检测到总线上出现隐性位，之后发送7个剩余隐性位

主动错误帧的帧格式



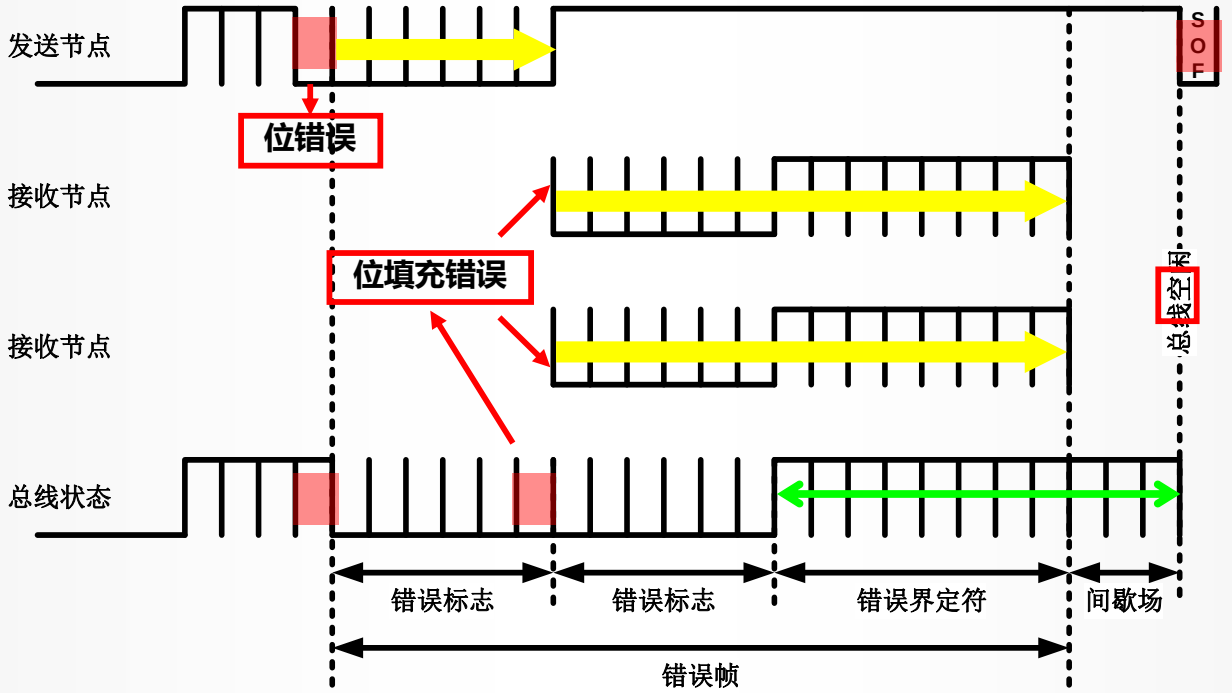
被动错误帧的帧格式



错误检测

– 错误帧的发送

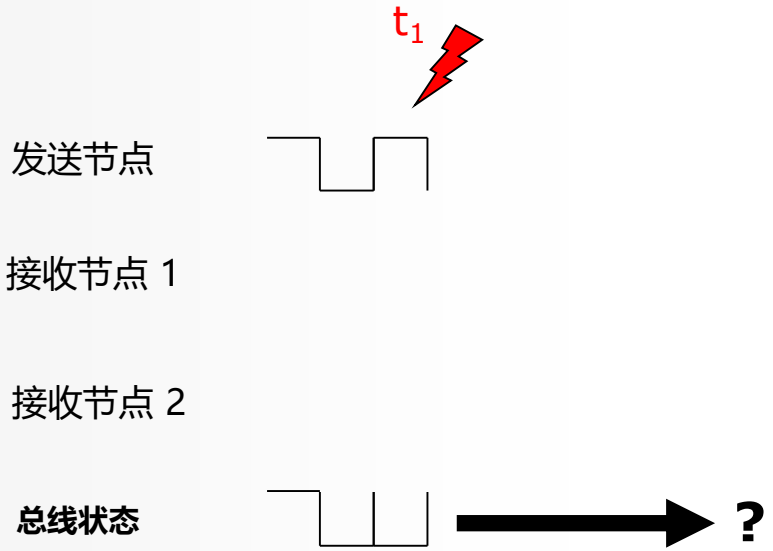
局部错误



错误检测

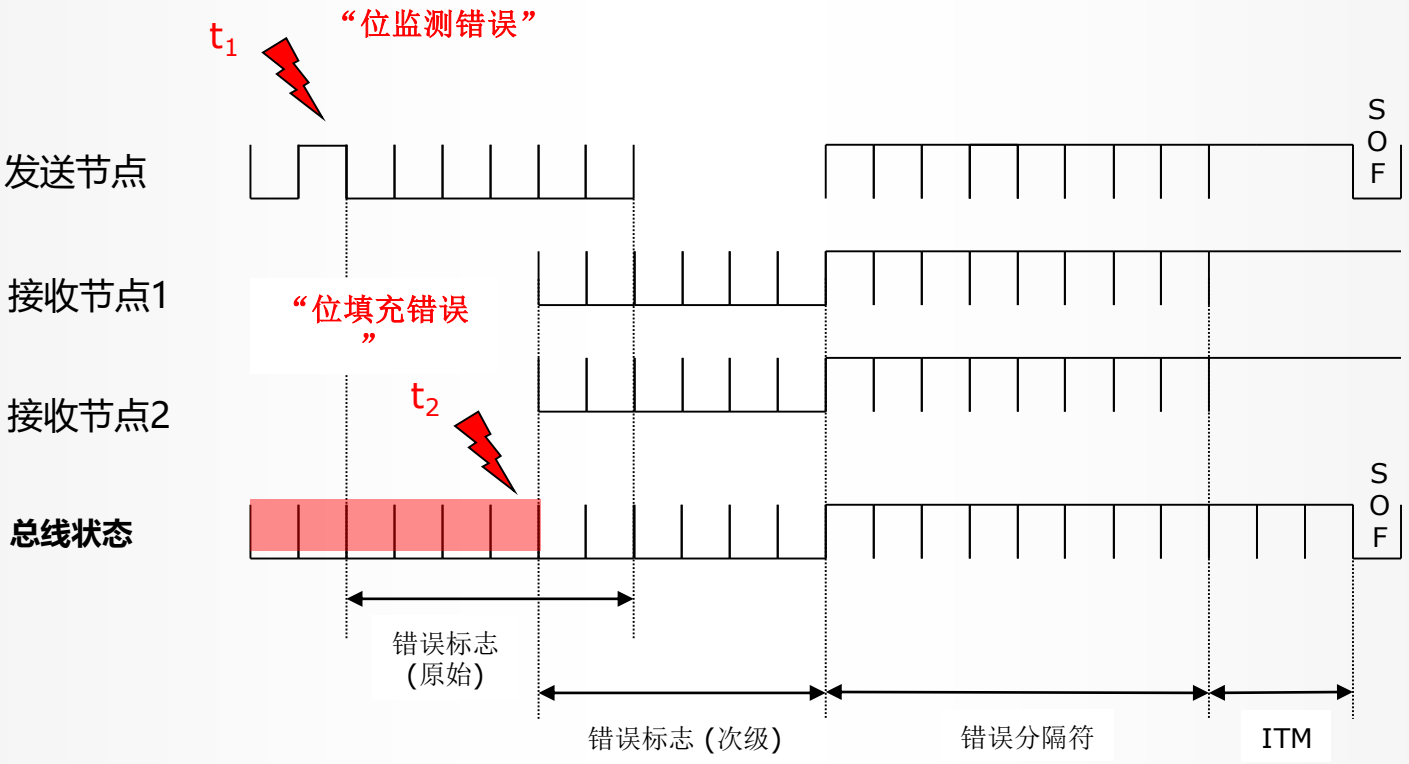
练习2：位错误

假设在时刻point t1 发送节点检测到了一个位监测错误，请完成两个接收节点的信号响应。



错误检测

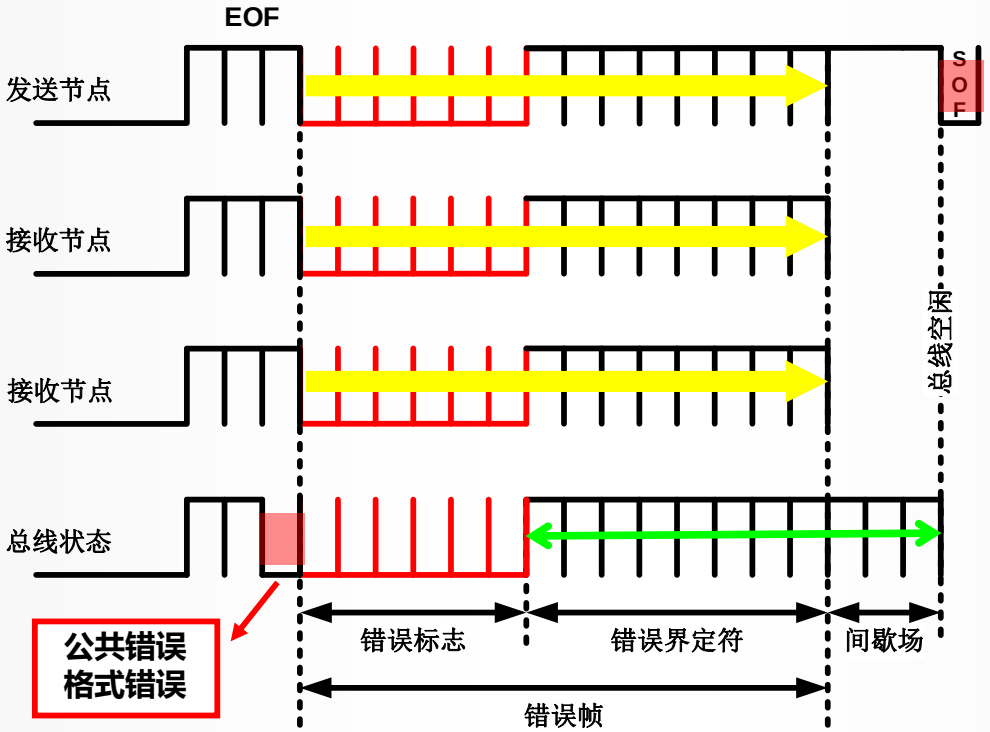
练习2：答案


t1
“位监测错误”
发送节点
S
O
F
接收节点1
“位填充错误
”
接收节点2
t2
总线状态
S
O
F
错误标志
(原始)
错误标志 (次级)
错误分隔符
ITM

错误检测

– 错误帧

格式错误



错误检测

– 被动错误

- ✓ 由发送节点发送的被动错误标志，会诱发接收节点发送错误标志，特例
  - ➢ 仲裁期间，多个节点在发送报文
  - ➢ CRC序列结束前的6个位内发送，并且CRC序列的最后几位都是隐性（ACK）
- ✓ 由接收节点引起的被动错误标志不会诱发总线上的任何活动
- ✓ 被动错误节点检测到总线上6个连续相同的极性位后，认为错误标志已经送出

三、数据链路层

- 通信机制
- 数据帧
- 错误检测
- 帧格式

www.eontronix.com

帧格式

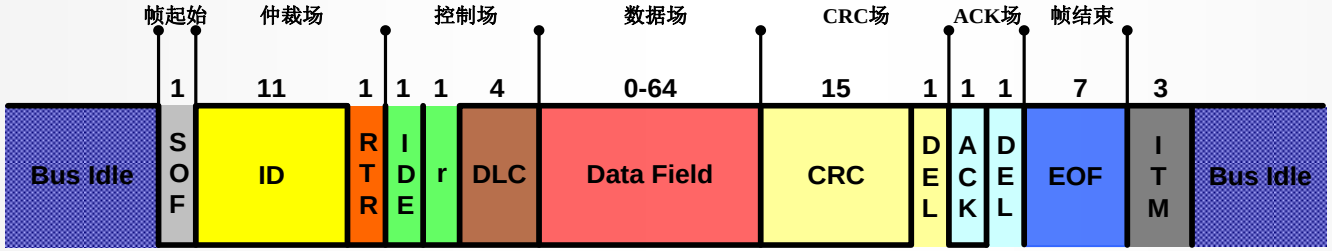
– CAN的帧格式

- ✓ 数据帧➔携带从发送节点至接收节点的数据
- ✓ 远程帧➔向其他节点请求发送具有同一标识符的数据帧
- ✓ 帧间空间➔数据帧（或远程帧）通过帧间空间与前述的各帧分开
- ✓ 错误帧➔节点检测到错误后发送错误帧
- ✓ 超载帧➔在先行的和后续的数据帧（或远程帧）之间附加一段延时

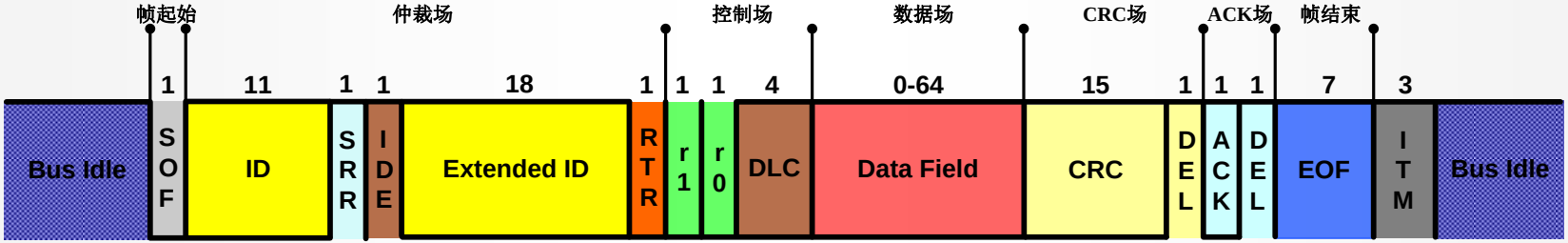
帧格式

– 数据帧

✓ 标准帧



✓ 扩展帧

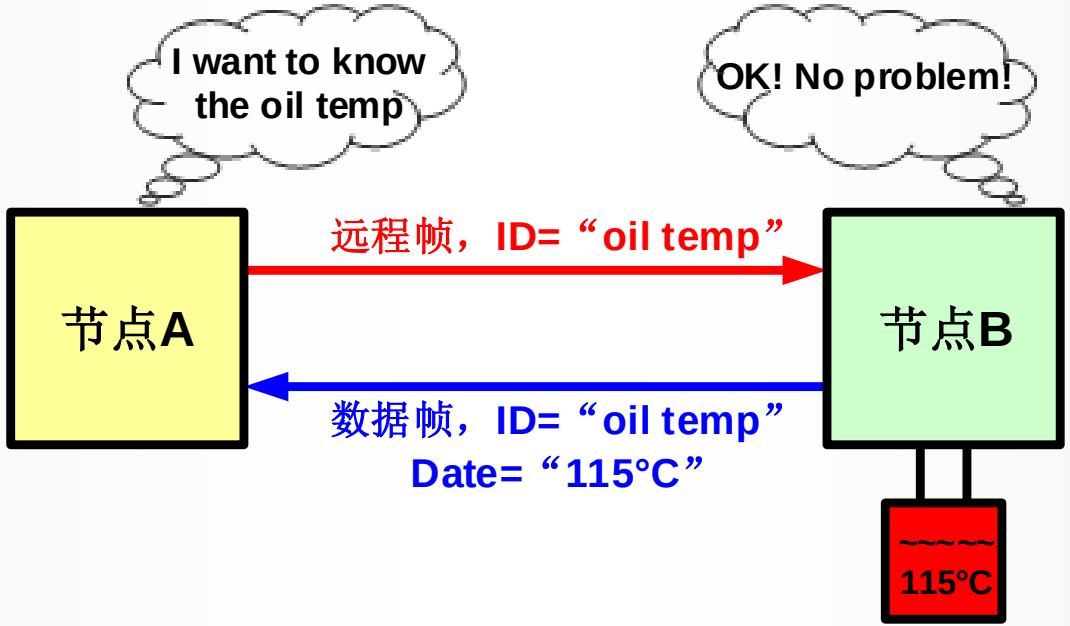


帧格式

– 远程帧

– 远程帧的使用

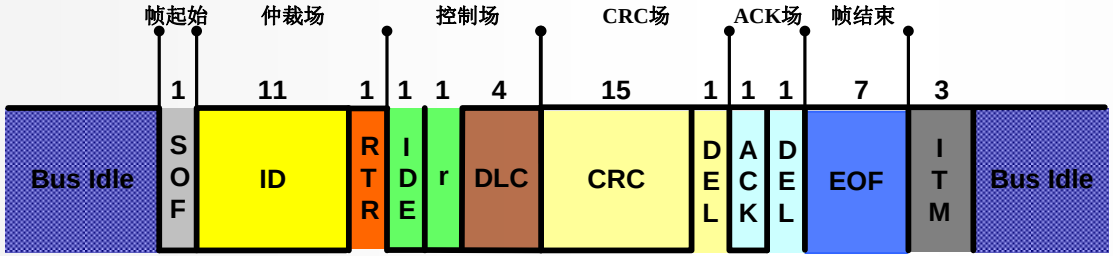
✓向其他节点请求发送具有同一标识符的数据帧



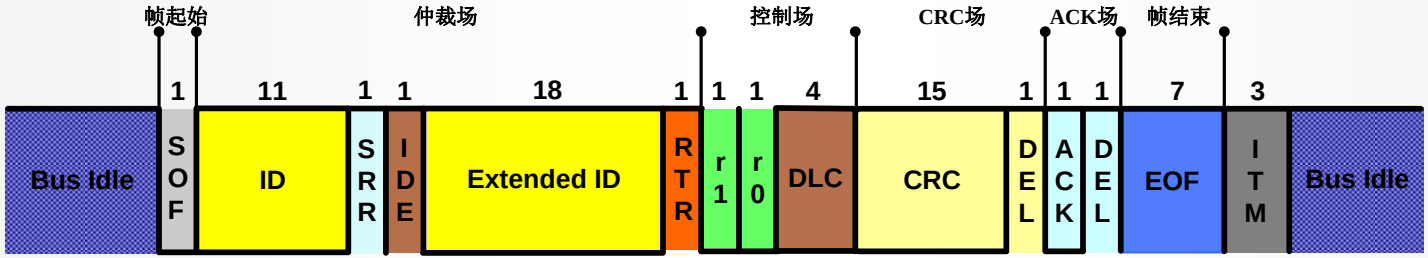
帧格式

– 远程帧

✓ 对应标准数据帧的远程帧



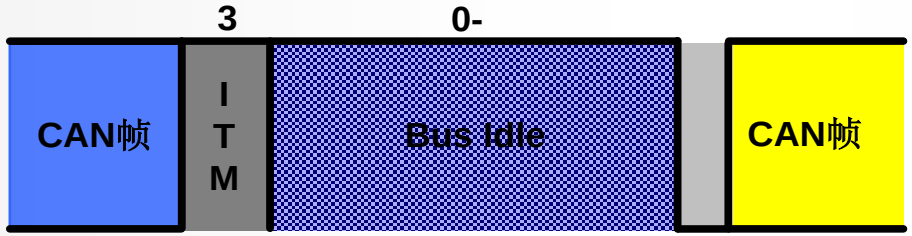
✓ 对应扩展数据帧的远程帧



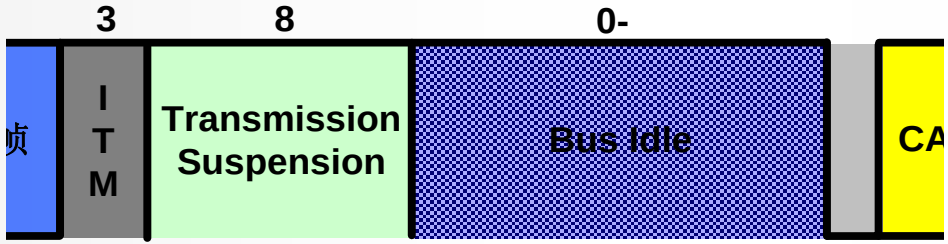
帧格式

– 帧间空间

✓ 主动错误节点使用的帧间空间格式



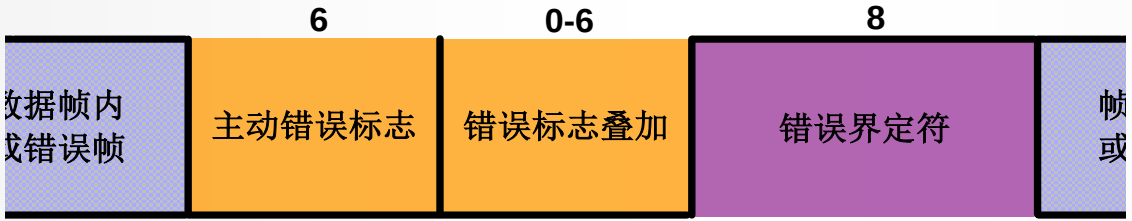
✓ 被动错误节点使用的帧间空间格式



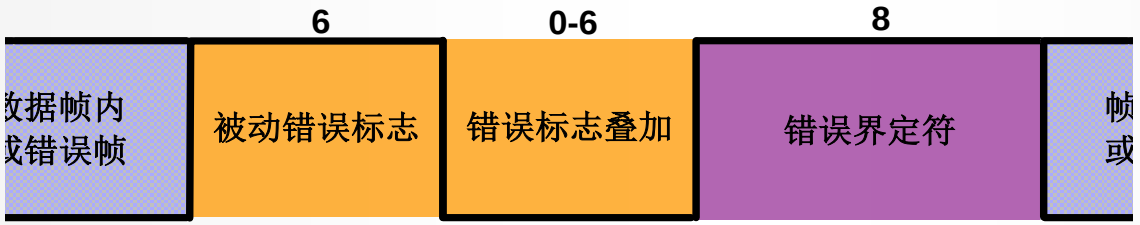
帧格式

– 错误帧

✓ 主动错误节点使用的错误帧格式



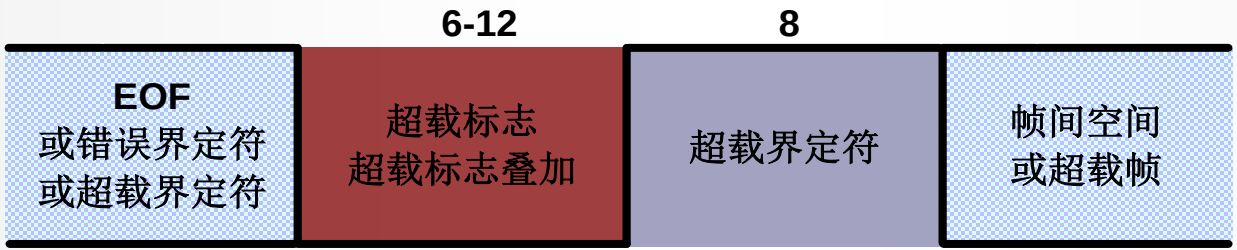
✓ 被动错误节点使用的错误帧格式



帧格式

– 超载帧

- ✓ 超载帧用以在先前的和后续的数据帧（或远程帧）之间提供一附加延时
- ✓ 大部分高层协议不使用超载帧



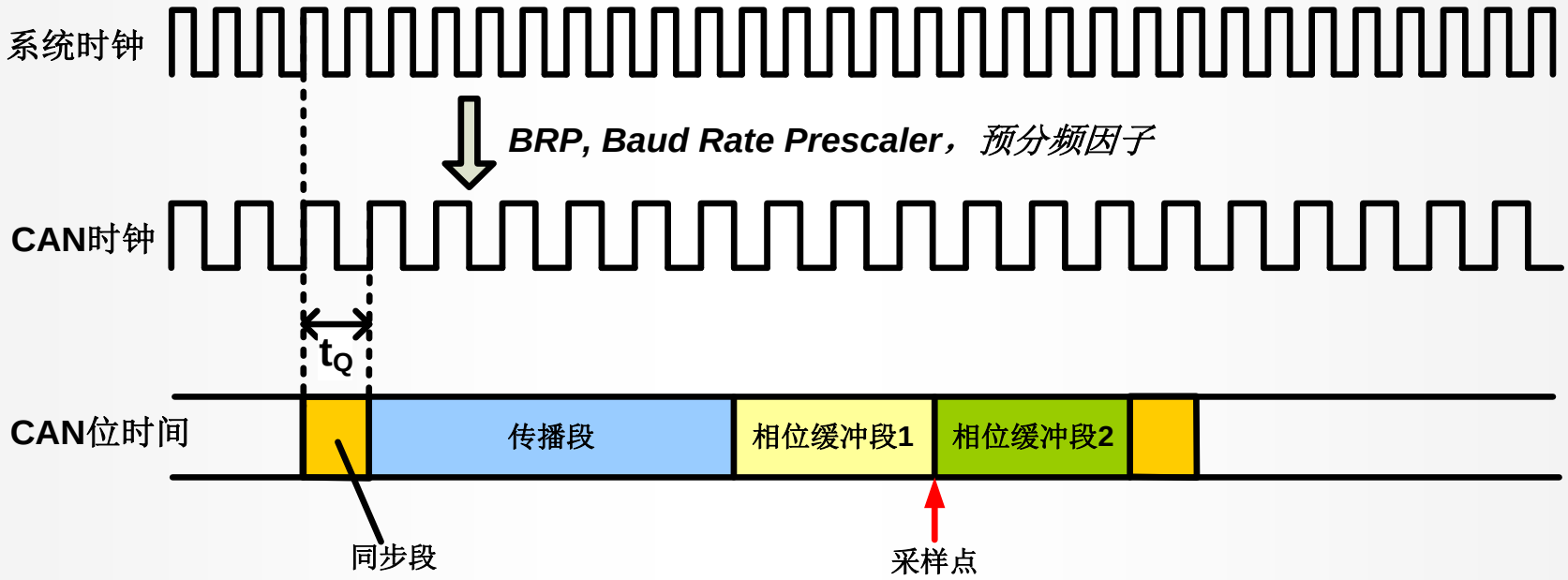
四、位定时与同步

- 位定时
- 同步

位定时

– 位定时

✓ Time quantum时间 份额



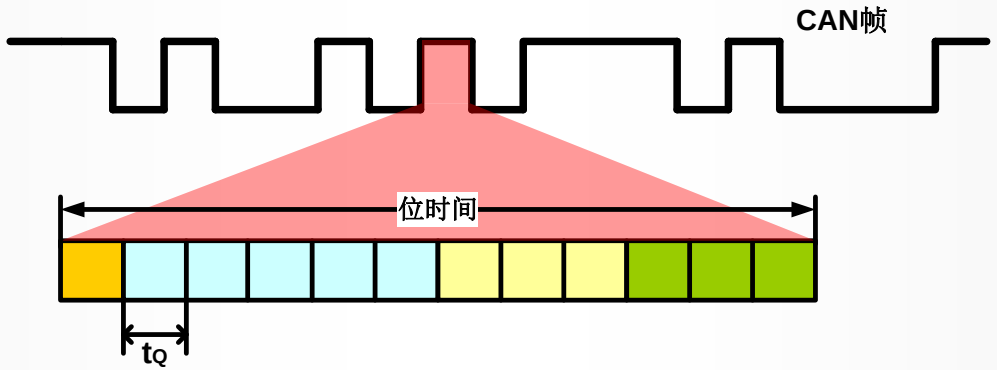
时间份额来源于对系统时钟可编程的分频

位定时

– 位定时

✓ 波特率

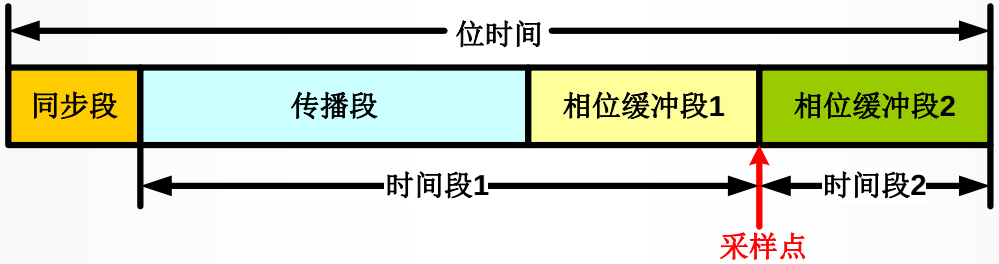
- 波特率由编程设置的时间份额长度和数量确定
- 波特率=1/位时间



位定时

– 位定时

✓ 位时间的组成

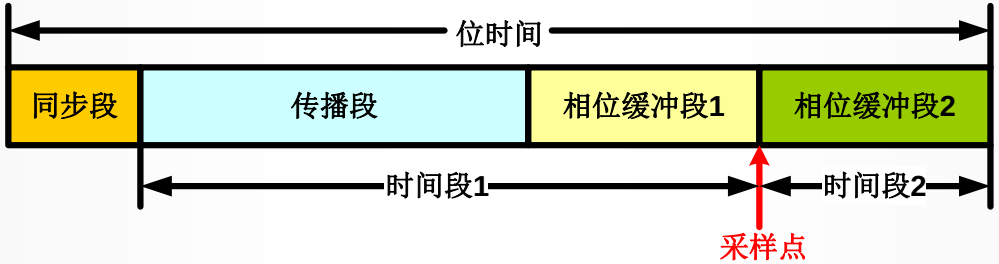


- 一个位时间包含4个时间段，8-25个时间份额（Time Quantum）
- 为方便编程，许多CAN模块将传播段和相位缓冲段1合并为一个时间段，即只有3个时间段

位定时

– 位定时

✓ 同步段—Synchronization Segment

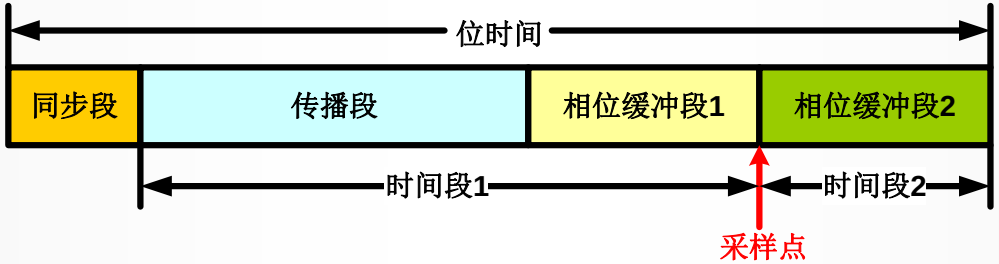


- 一个位的输出从同步段开始
- 同步段用于同步总线上的各个节点，跳变沿产生在此段内
- 固定长度，1个时间份额

位定时

– 位定时

✓ 传播段—Propagation Segment

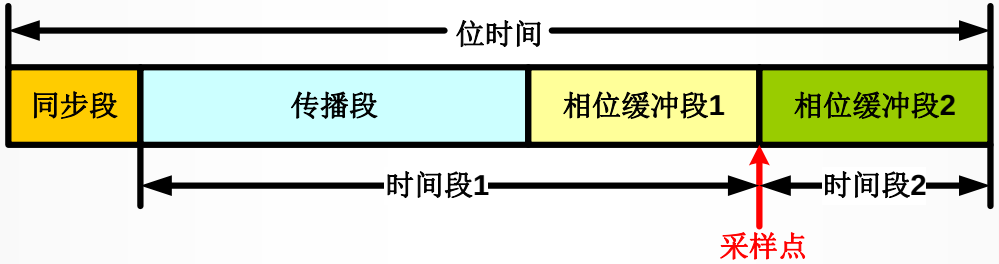


- 传播段用于补偿信号通过网络和节点传播的物理延迟
- 传播段长度应能保证2倍的信号在总线的延迟
- 长度可编程（1...8个时间份额或更长）

位定时

– 位定时

- ✓ 相位缓冲段1—Phase Buffer Segment1
- ✓ 相位缓冲段2—Phase Buffer Segment2



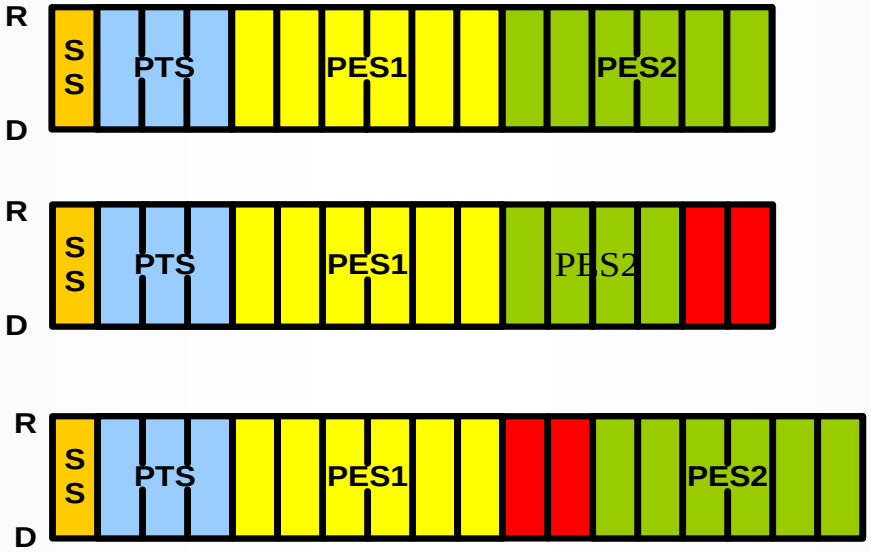
- ➢ 允许通过重同步对相位缓冲段1加长和相位缓冲段2缩短➔最大长度=SJW(同步跳转宽度)
- ➢ 相位缓冲段1末端进行总线状态的采样
- ➢ 长度可编程（1...8个时间份额或更长）
- ➢ SJW长度强制设置

位定时

– 位定时

✓ 同步跳转宽度—Synchronization Jump Width

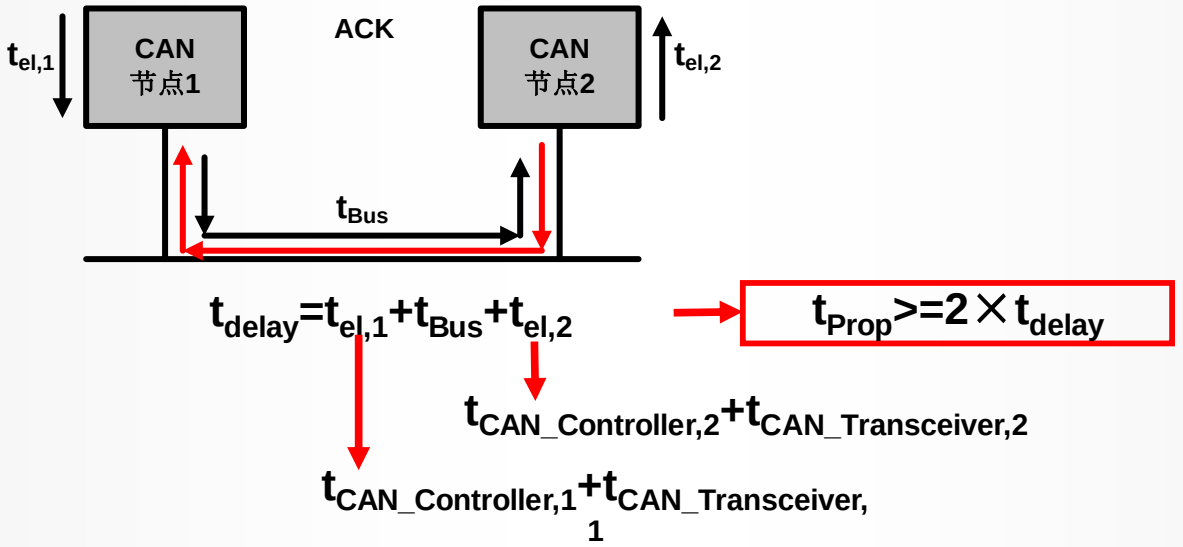
- SJW为PES1和PES2调整的最大长度
- SJW必须小于PES1和PES2的最小值



– 位定时

✓ 延迟时间的确定

➢ 保证2倍信号在总线的延迟



位定时

– 位定时

✓ 位定时参数确定

- T(Bit) = 1/Baudrate
- T(tq) = T(Bit) / NBT
- T(Prop_Seg)= 2*(DelayTransceiver+DelayBus)
- Prop_Seg = T(Prop_Seg) / T(tq)
- If (NBT-1-Prop_Seg) /2 为偶数
  Phase_Seg1 = Phase_Seg2 = (NBT-1-Prop_Seg) / 2
  else
  Phase_Seg1 = (NBT-1-Prop_Seg)/2, Phase_Seg2 = (Phase_Seg1)+1
- SJW = min (Phase_Seg1 , 4)

– 位定时

✓ 位定时确定的用例:

给定，MCU晶振8MHz，位速率1Mbps，总线长度20m，单位总线延迟5ns/m，物理接口的发送接收延迟150ns@85C (From Freescale AN1798)

① 总线的物理延迟= 20X5= 100ns
② tProp = 2X (100+150) =500ns
③ 选择BRP = 1，tQ = 125ns，NBT = 8
④ PROP_SEG = 500/125=4
⑤ NBT - PROP_SEG - 1 = 3, PHASE_SEG1 = 1, PHASE_SEG2 = 2
⑥ RJW= min { 4, PHASE_SEG1 }=1

四、位定时与同步

- 位定时
- 同步

同步

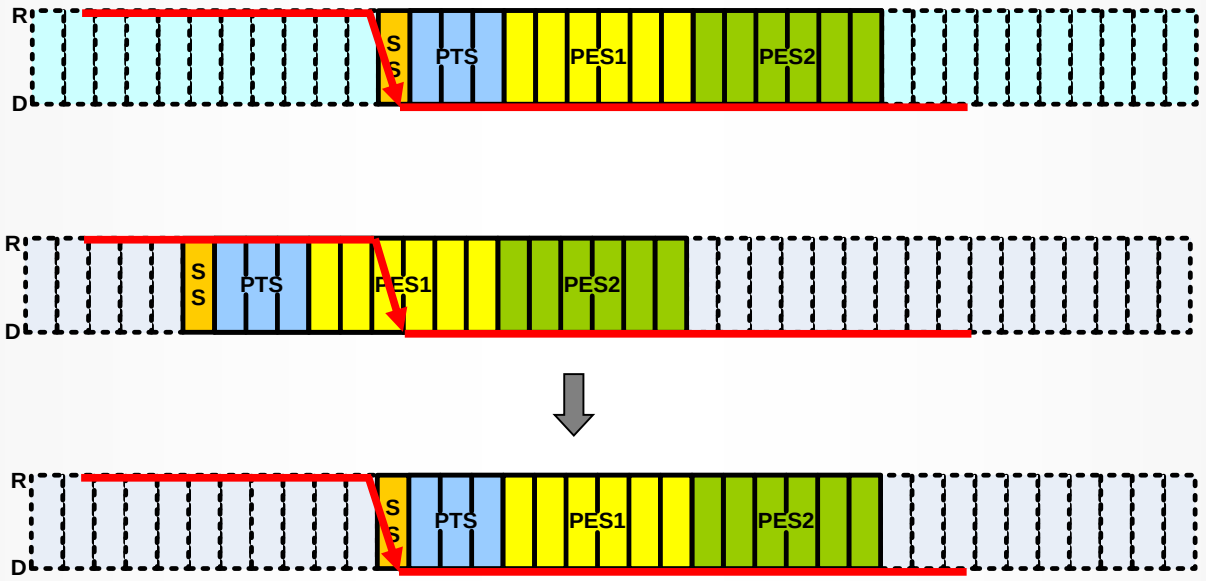
– 同步

- ✓ **CAN的同步包括硬同步和重同步两种同步方式**
- ✓ **同步规则：**
  - ➢ 一个位时间内只允许一种同步方式
  - ➢ 任何一个“隐性”到“显性”的跳变都可用于同步
  - ➢ 硬同步发生在SOF➔所有接收节点调整各自当前位的同步段，使其位于发送的SOF位内
  - ➢ 重同步发生在一个帧的其他位场内，当跳变沿落在了同步段之外
  - ➢ 在SOF到仲裁场有多个节点同时发送的情况下，发送节点对跳变沿不进行重同步

同步

– 同步

✓ 硬同步 发生在SOF位➔所有接收节点调整各自当前位的同步段，调整宽度不限

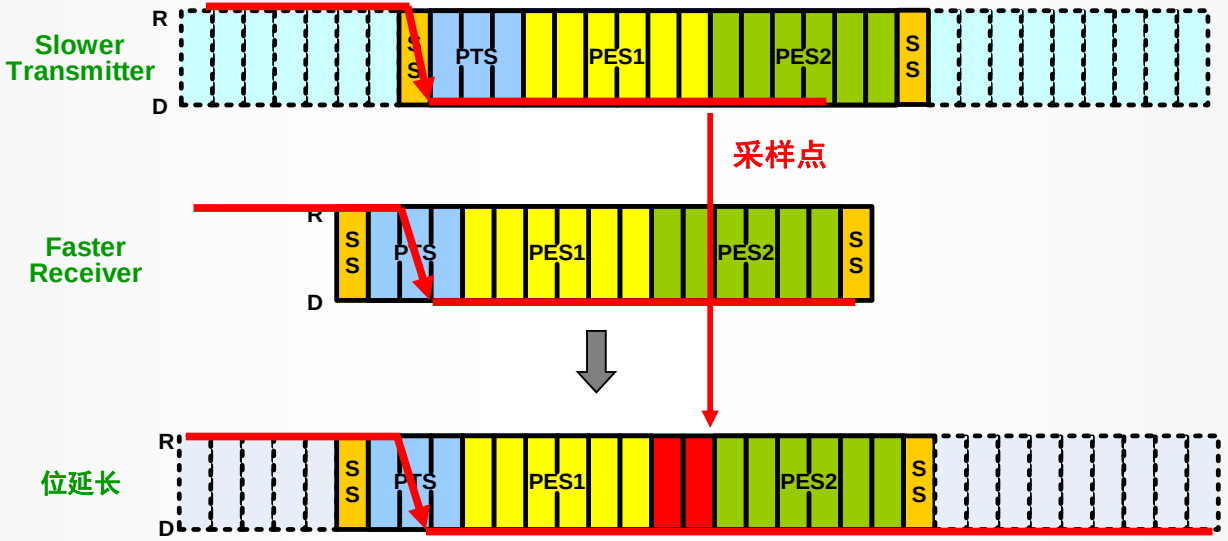


同步

– 同步

✓ 重同步

➢ 相位误差为正，跳变沿沿位于采样点之前➔相位缓冲段1 增长

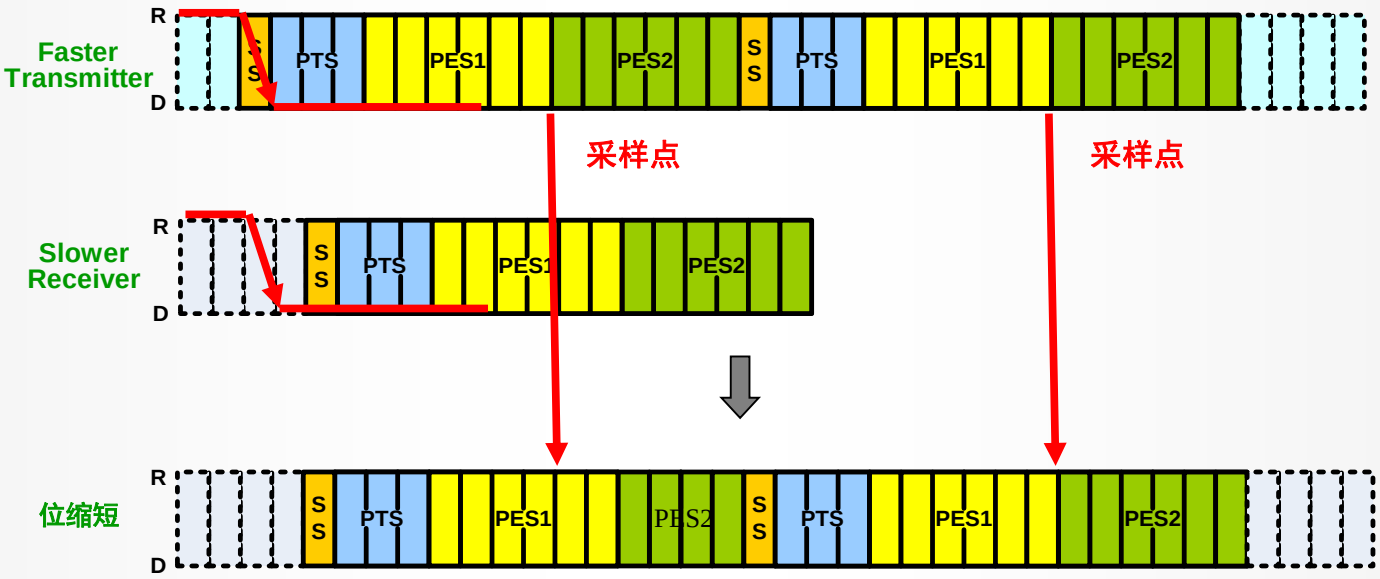


同步

– 同步

✓ 重同步

➢ 相位误差为负，跳变沿位于前一个位的采样点之后➔相位缓冲段2缩短



五、物理层

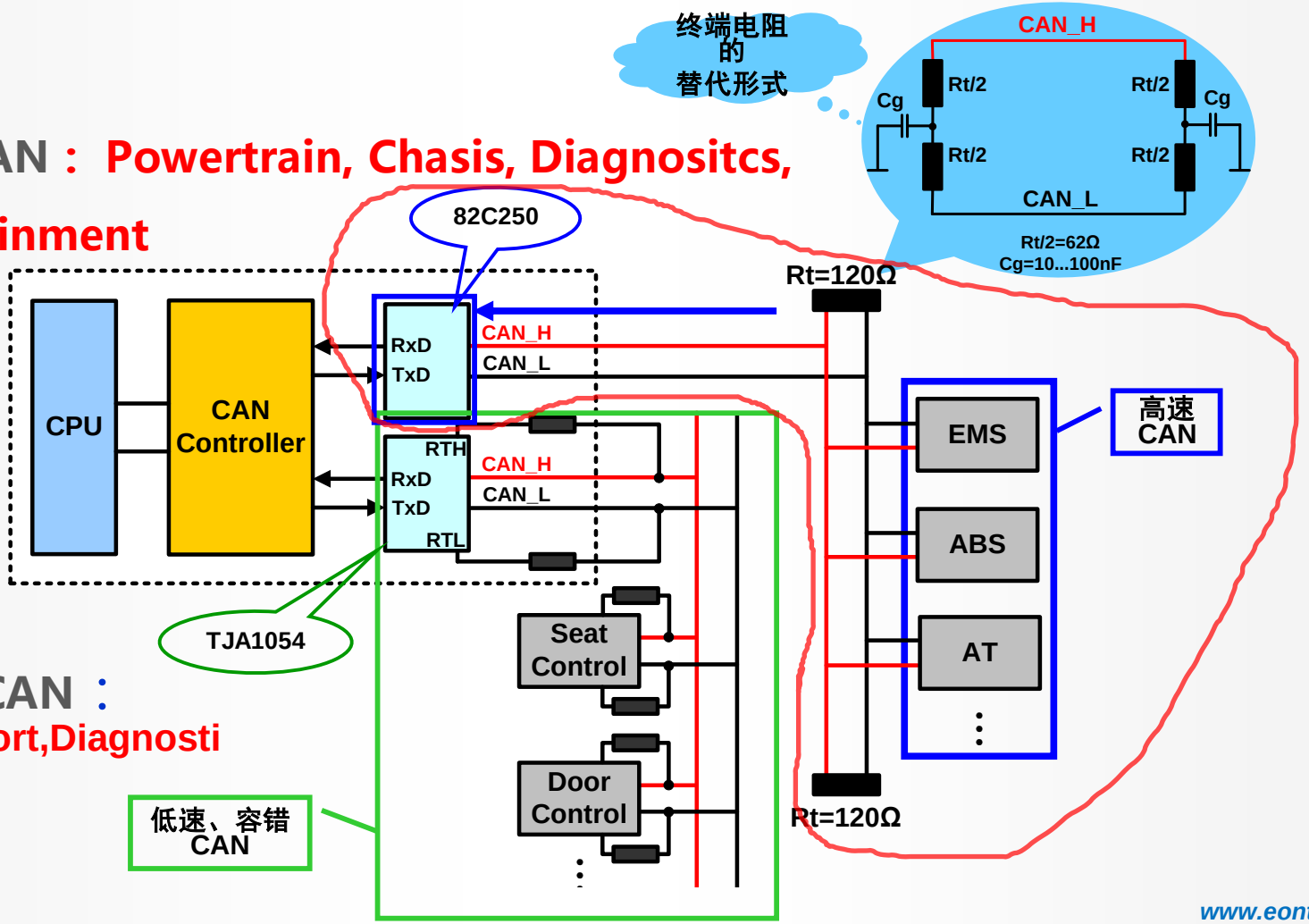
- 高速CAN
- 低速容错CAN

物理层

– 应用领域

✓ 高速CAN： Powertrain, Chasis, Diagnositcs, Infotainment

✓ 低速CAN： Comfort,Diagnostics

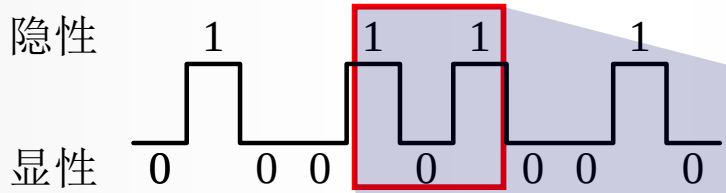


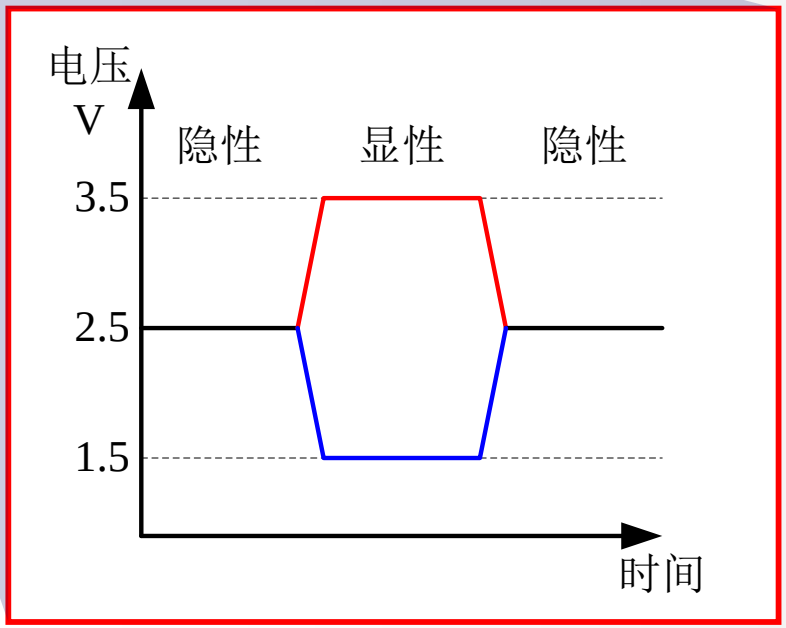
五、物理层

- 高速CAN
- 低速容错CAN

高速CAN

– 总线电压





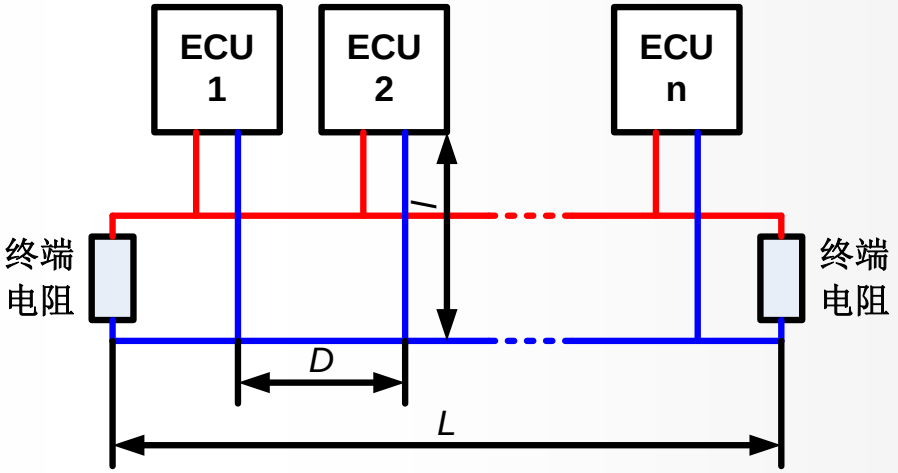
隐性表示1，显性表示0

www.eontronix.com

高速CAN

– 拓扑结构

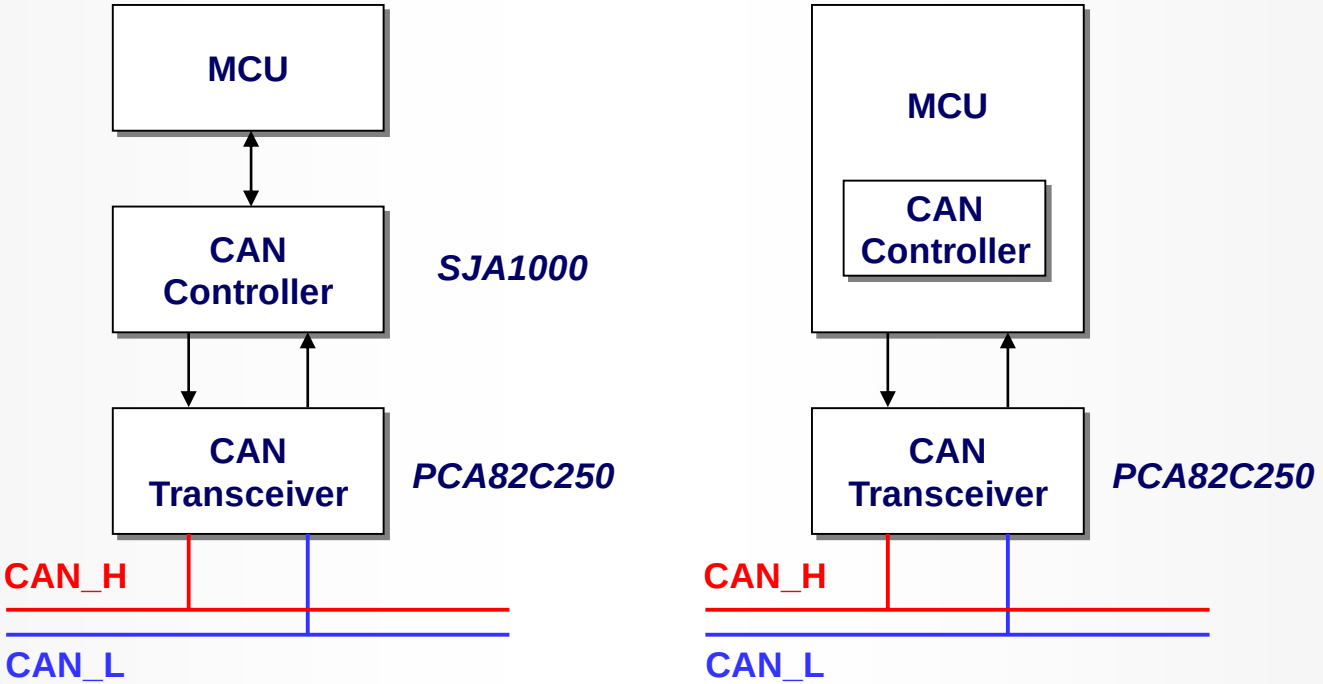
- ➢ CAN线长度
- ➢ 终端电阻R



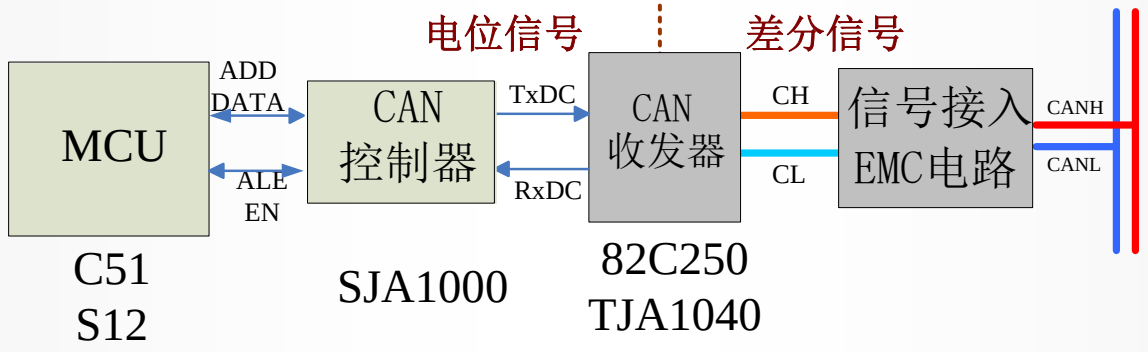
参数	符号	单位	数值		
			最小	名义	最大
总线长度	*L*	m	0		40
支线长度	*l*	m	0		0.3
节点距离	*D*	m	0.1		40
终端电阻	R_L	m	100	120	130

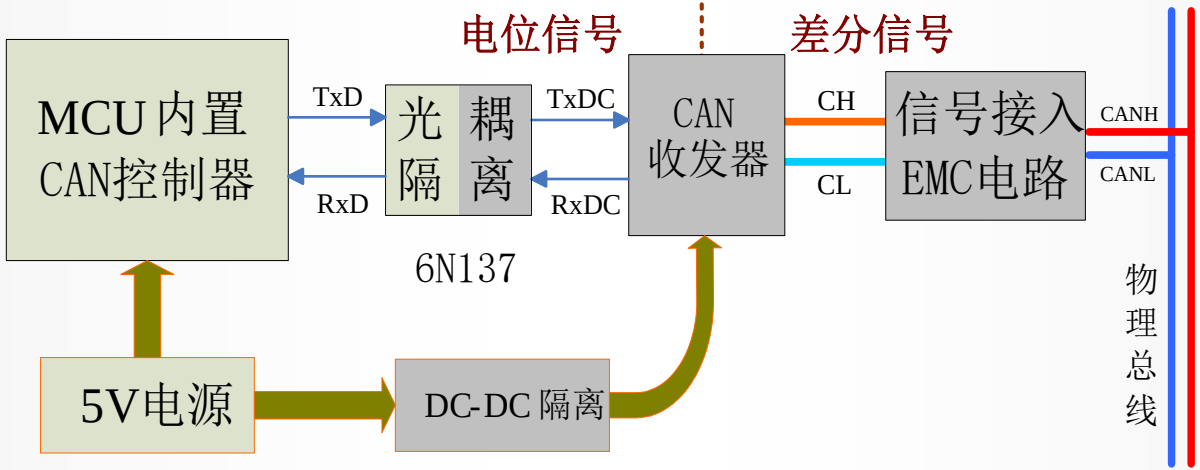
高速CAN

– 电路设计


MCU
CAN Controller
SJA1000
CAN Transceiver
PCA82C250
CAN_H
CAN_L
MCU
CAN Controller
CAN Transceiver
PCA82C250
CAN_H
CAN_L

高速CAN

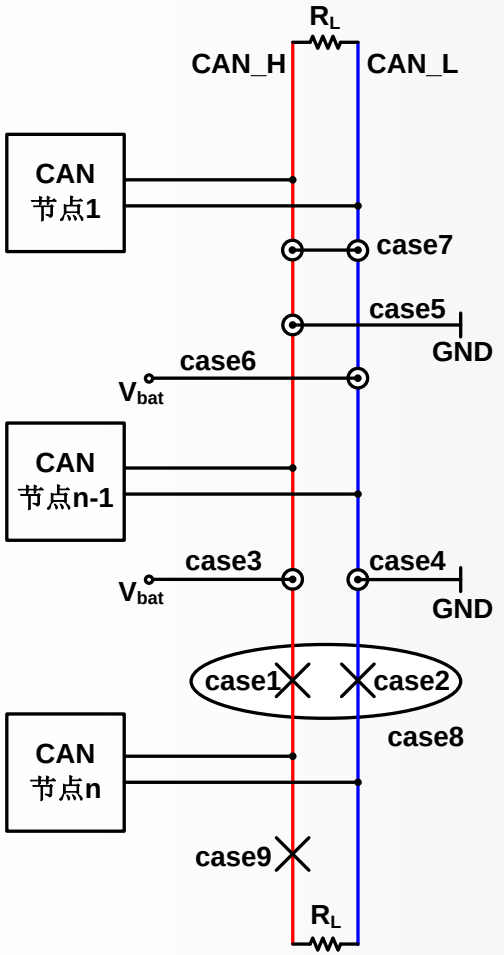

电位信号
差分信号
MCU
ADD
DATA
ALE
EN
CAN
控制器
TxDC
RxDC
CAN
收发器
CH
CL
信号接入
EMC电路
CANH
CANL
C51
S12
SJA1000
82C250
TJA1040


电位信号
差分信号
MCU内置
CAN控制器
TxD
RxD
光
隔
耦
离
TxDC
RxDC
CAN
收发器
CH
CL
信号接入
EMC电路
CANH
CANL
6N137
物
理
总
线
5V电源
DC-DC 隔离

高速CAN

– CAN线故障的容错性能

(1) CAN_H开路
(2) CAN_L开路
(3) CAN_H对VBAT短路
(4) CAN_L对GND短路
(5) CAN_H对GND短路
(6) CAN_L对V_{BAT}短路
(7) CAN_H对CAN_L短路
(8) CAN_H和CAN_L开路
(9) 终端电阻开路

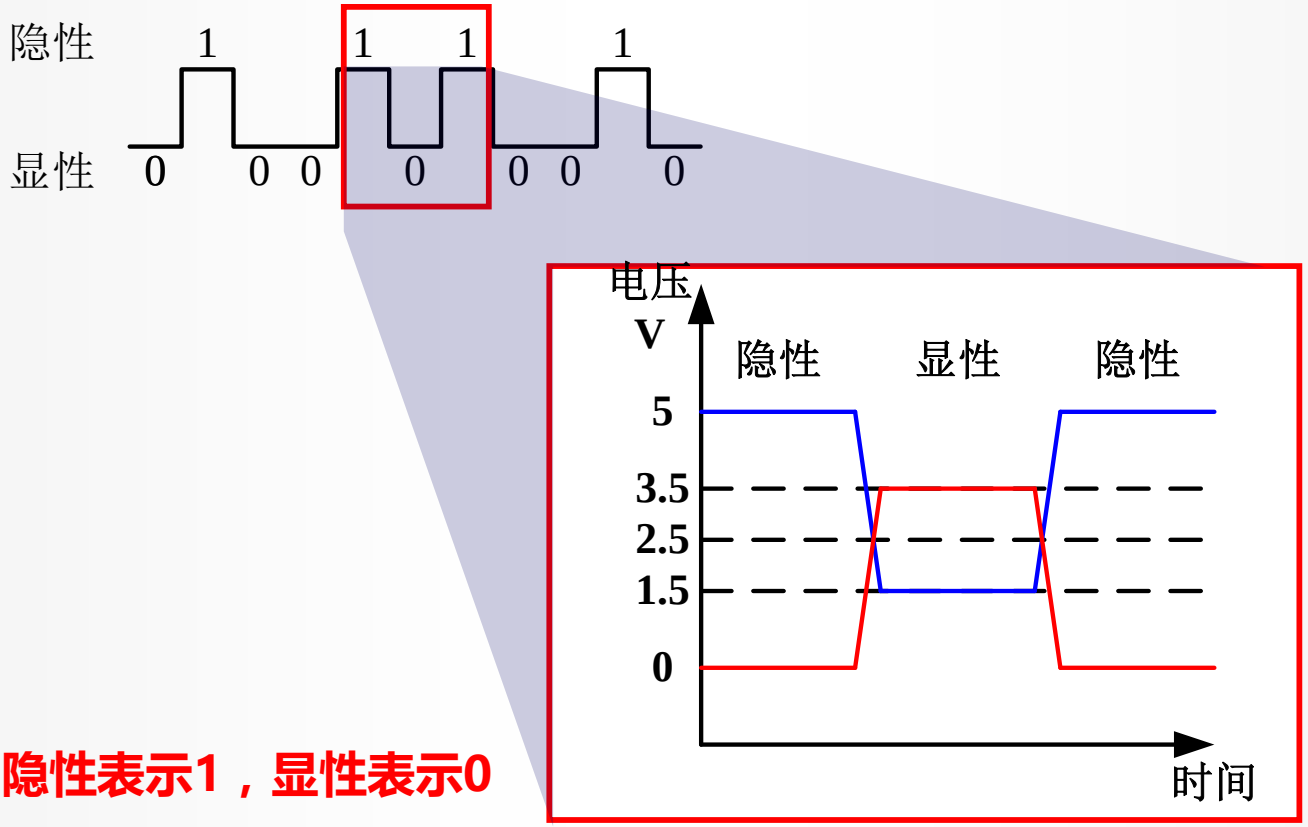


五、物理层

- 高速CAN
- 低速容错CAN

低速容错CAN

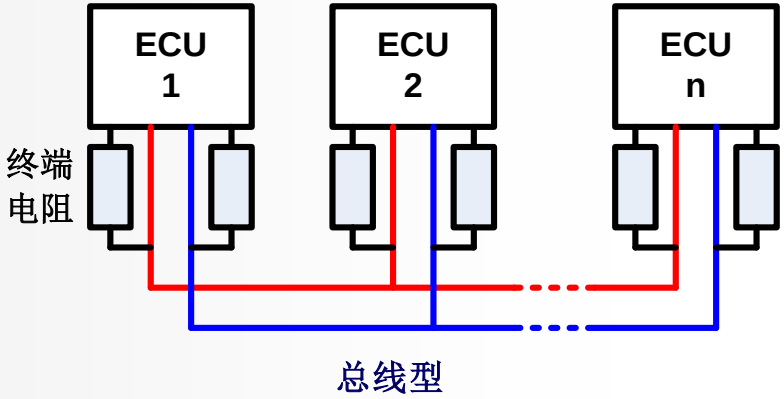
– 总线电压

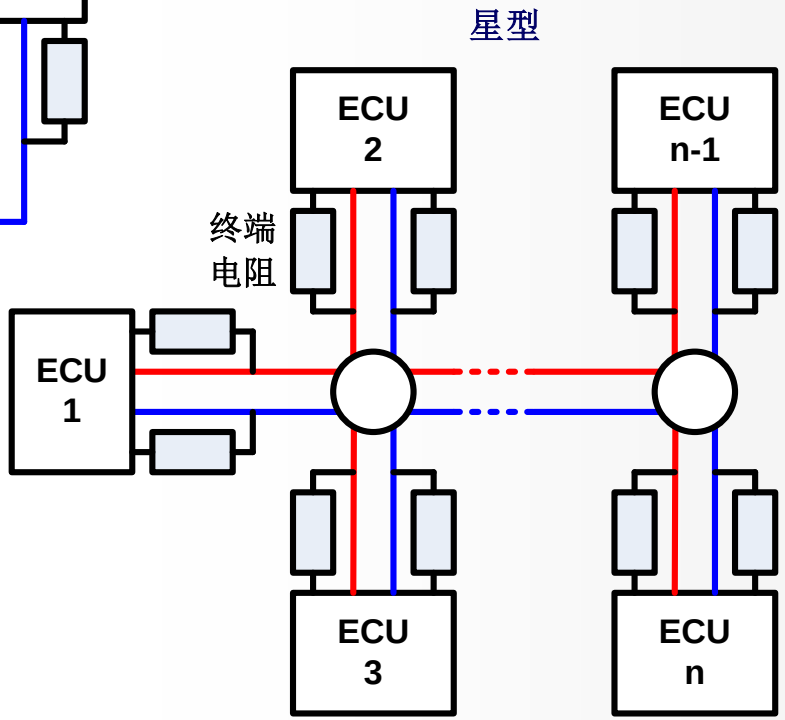


隐性表示1，显性表示0

低速容错CAN

– 拓扑结构





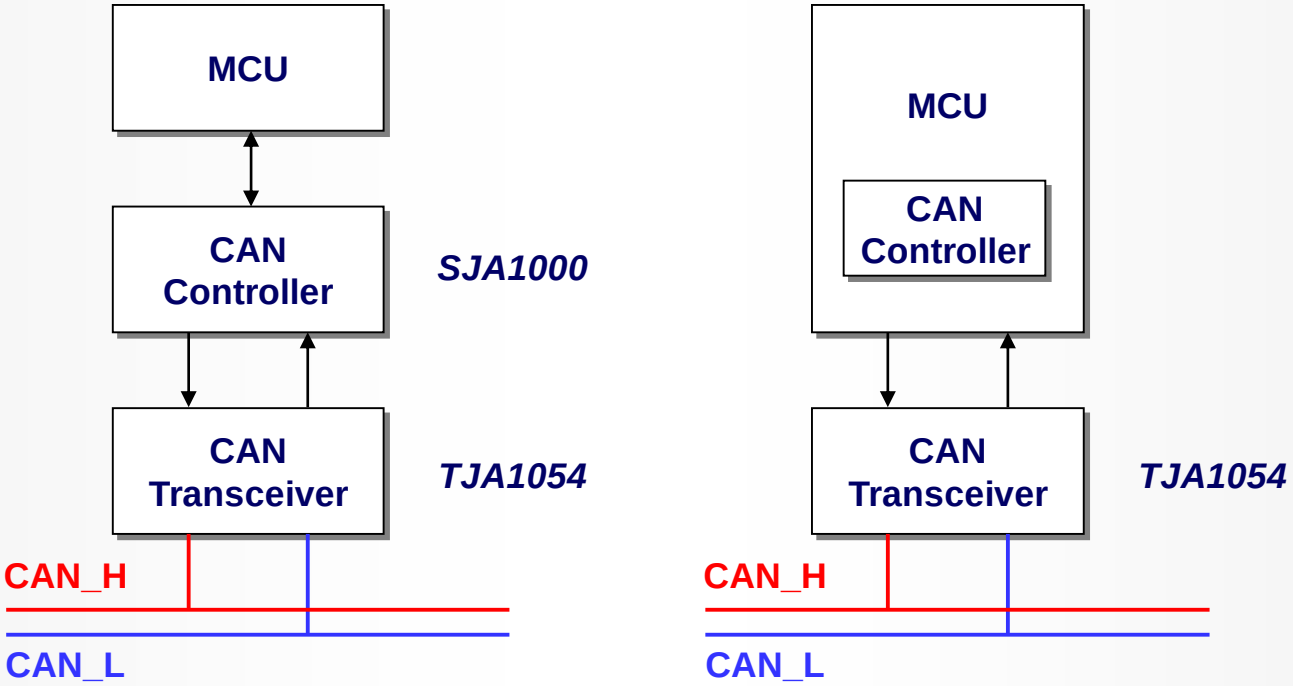
- 4.7K欧>R>500欧
- R_{all}>100欧
- 可采用星型连接

CAN线总长度<40m

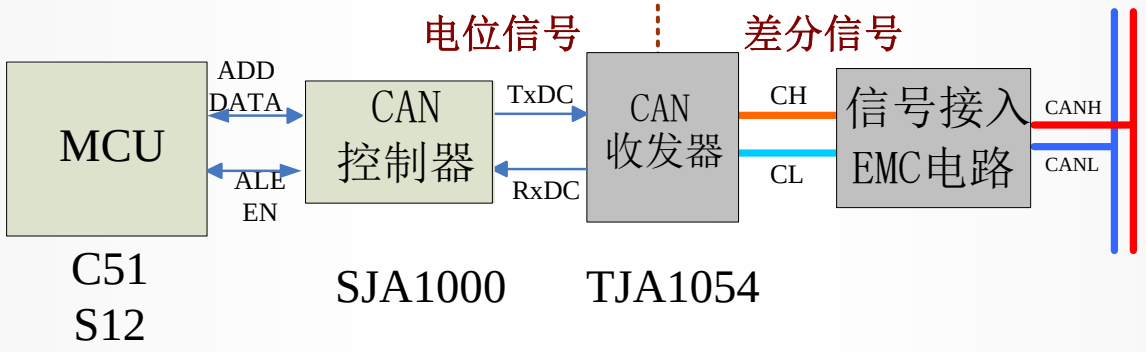
www.eontronix.com

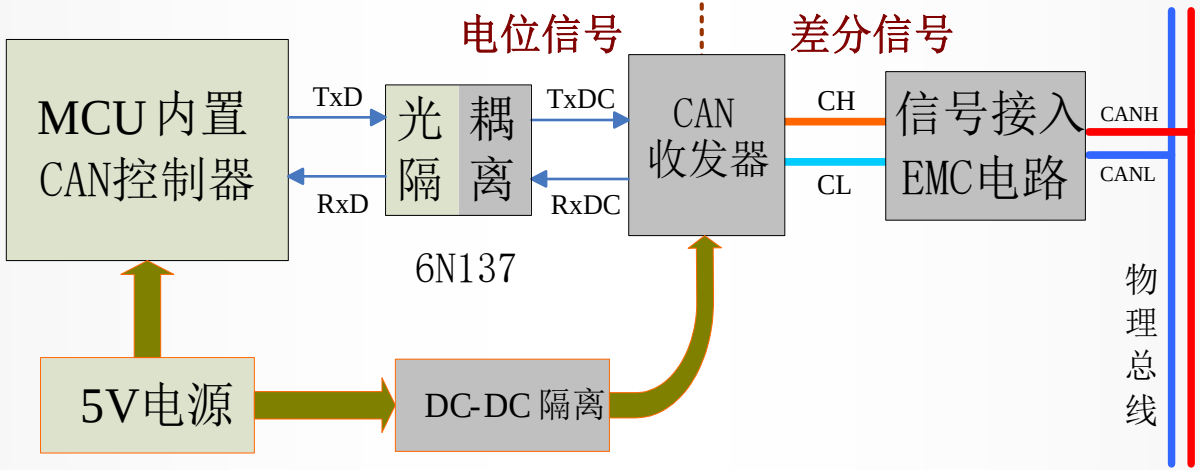
低速容错CAN

– 电路设计



低速容错CAN

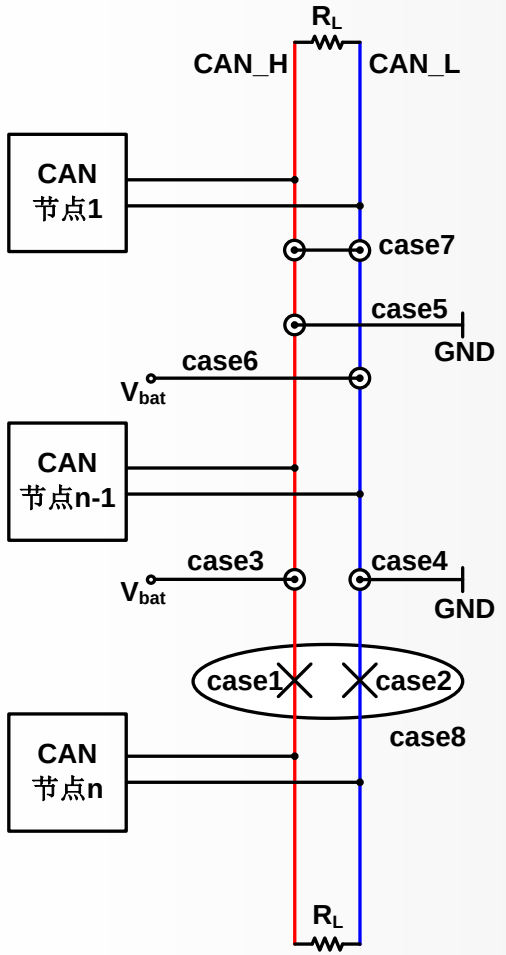

电位信号
差分信号
MCU
ADD
DATA
ALE
EN
CAN
控制器
TxDC
RxDC
CAN
收发器
CH
CL
信号接入
EMC电路
CANH
CANL
C51
S12
SJA1000
TJA1054


电位信号
差分信号
MCU内置
CAN控制器
TxD
RxD
光
隔
耦
离
TxDC
RxDC
CAN
收发器
CH
CL
信号接入
EMC电路
CANH
CANL
6N137
物
理
总
线
5V电源
DC-DC 隔离

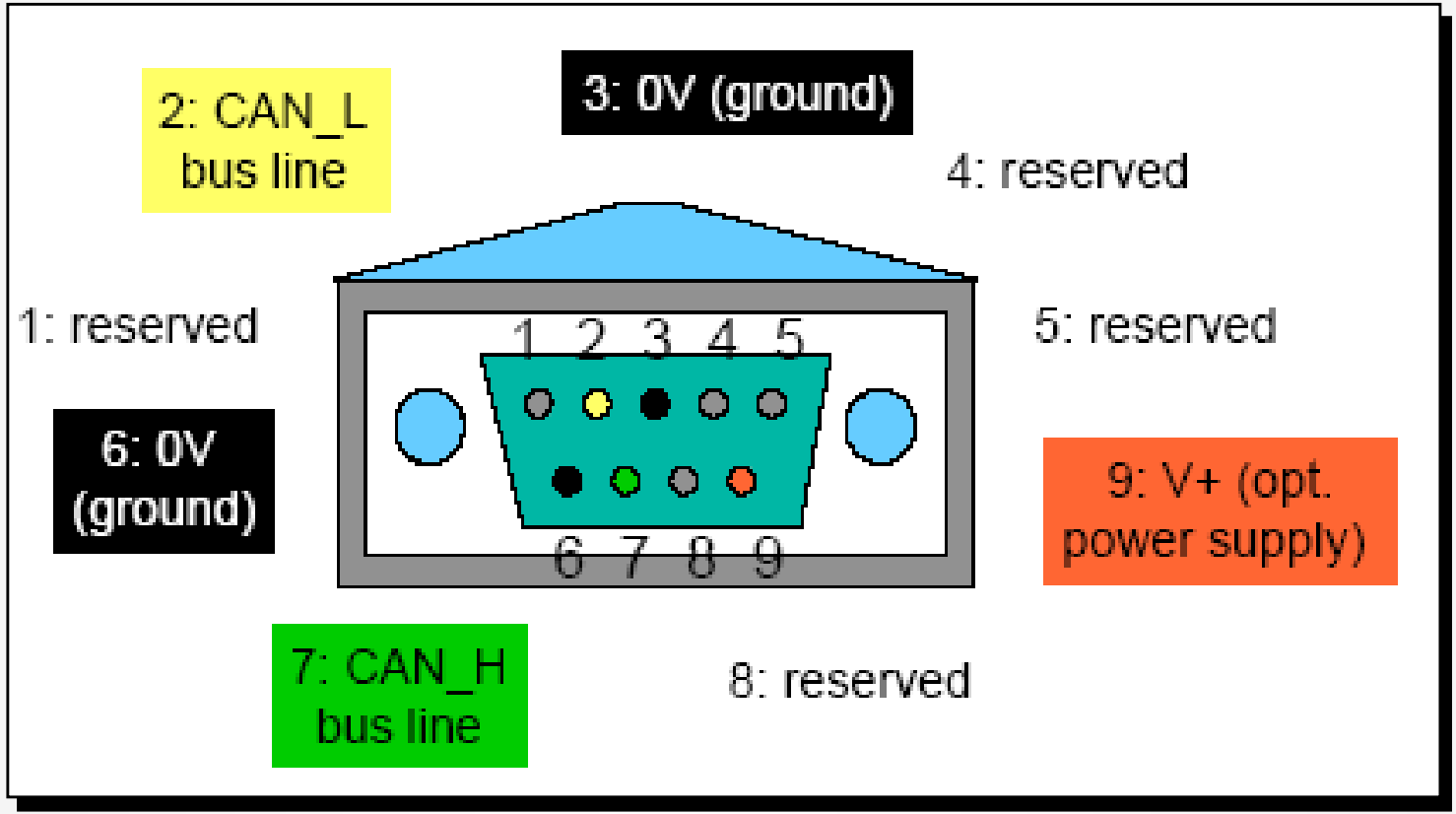
低速容错CAN

– CAN线故障的容错性能

(1) CAN_H开路
(2) CAN_L开路
(3) CAN_H对V_{BAT}短路
(4) CAN_L对GND短路
(5) CAN_H对GND短路
(6) CAN_L对V_{BAT}短路
(7) CAN_H对CAN_L短路
(8) CAN_H和CAN_L开路



CAN引脚定义


2: CAN_L bus line
3: 0V (ground)
4: reserved
1: reserved
5: reserved
1 2 3 4 5
6 7 8 9
6: 0V (ground)
9: V+ (opt. power supply)
7: CAN_H bus line
8: reserved

谢谢大家！