Application Note

SPC11X8/SPD11X8 LIN 通信使用指南

2020 年 7 月 – 版本 1

概述

本文主要介绍在 SPC11X8/SPD11X8 UART 模块的基础上实现 LIN 通信协议。

目录

表格列表

图片列表

1 LIN 总线协议概述

LIN 总线是一个低成本的串行通信网络，是对 CAN 总线等其它汽车多路网络的一种补充，适用于对网络的带宽、性能或容错功能没有过高要求的应用。

1.1 LIN 通信原理

主节点负责控制整条 LIN 总线，决定何时发送、以及向 LIN 总线上发送哪一帧数据。从节点只需要在检测到总线上的 Break 信号时做出回应即可。

在发送消息帧时，主节点负责：

- 发送帧头（Header），由 Break 信号、Sync 字节和 ID 组成
- 监测数据字节和校验字节，对这些字节的一致性进行评估

当接收到主节点发来的 ID 字节，从节点就会进行数据的接收或者发送：

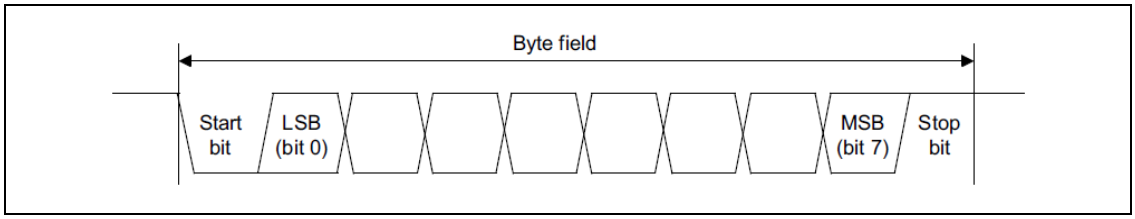
- 检查接收到的 ID 字节
- 根据 ID 字节的值，决定采取下面哪一种操作
  - 接收数据
  - 发送数据
  - 什么都不做

注意：当发送数据时，从节点会发送 1 ~ 8 个数据字节以及一个校验字节。

1.2 LIN 通信帧

LIN 通信是基于标准的 UART 协议格式，即一个起始位，8 个数据位（LSB 优先）和一个停止位。

图 1-1： LIN 通信帧字节格式



LIN 通信帧是由一个帧头（Header）和一个响应（Response）组成的。

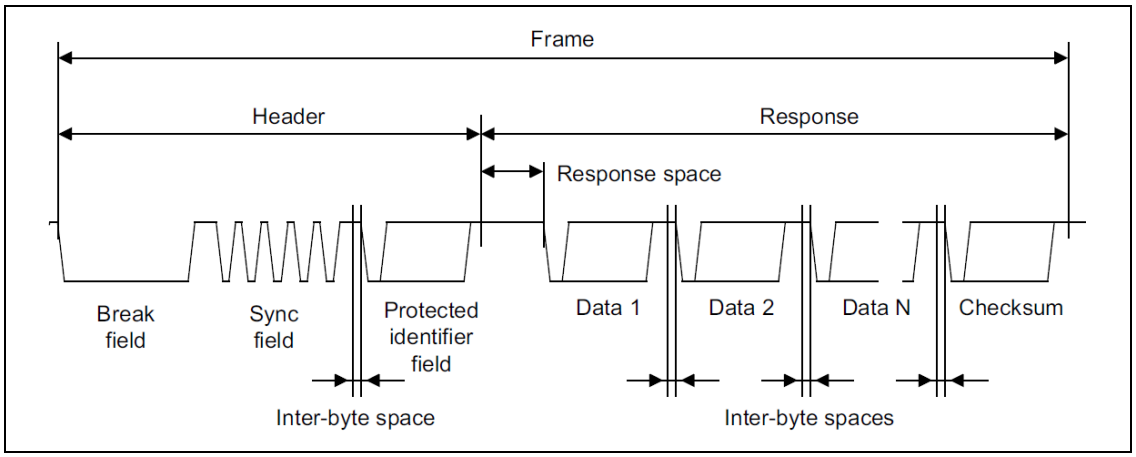
帧头 Header 包括：

- Break 域
- Sync 域
- ID（Protected identifier）域

响应 Response 包括：

- 1 ~ 8 个字节数据
- 1 个校验字节

图 1-2： LIN 通信帧格式

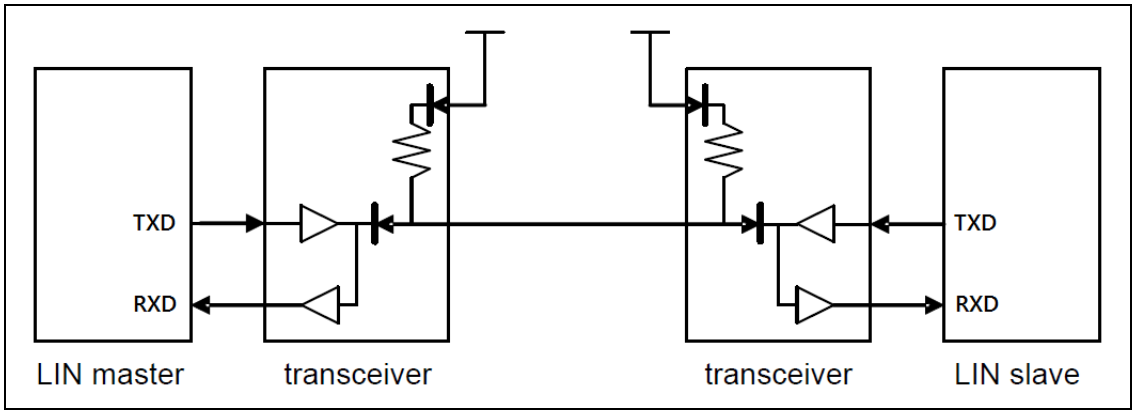


帧头 Header 总是由主节点发送的，响应 Response 只能由一个节点进行发送：主节点或者从节点均可。在某一时刻，只能有一个节点发送响应 Response，但是可以有多个节点接收响应 Response。

2 LIN 通信系统

本文采用的 LIN 通信实验的系统如图 2-1 所示，它包括一个主设备和一个从设备（它们通过 LIN 收发器相连接）。

图 2-1： LIN 通信系统



当 SPC11X8/SPD11X8 作为主节点时，Break 信号可以通过下面的方法产生：

- 将寄存器位 UARTLCR.SB 设为“1”，此时 UART TXD 输出低电平
- 等待 N 个位时间（bit time），其中 N ≥ 13，满足 Break 信号所需要的低电平时间
- 将寄存器位 UARTLCR.SB 设为“0”，此时 UART TXD 输出高电平
- 等待 M 个位时间（bit time），其中 M ≥ 1，满足 Break delimiter 所需要的高电平时间

当 SPC11X8/SPD11X8 作为从节点时，可以通过下面的方法检测 Break 信号：

- 查询寄存器位 UARTLSR.BI 的值，如果为“1”，则表明收到 Break 信号；如果为“0”，则表明未收到 Break 信号。

从图 2-1 可以看出，无论是主节点还是从节点，在从 TXD 发送数据的时候，LIN 收发器都会将发送的数据反馈到 RXD。因此，LIN 节点在发送完一个字节数据后，都会同时收到一个相同的反馈字节。可以通过比较收到的反馈字节和发送字节数据是否相同，来判断通信是否正常。需要特别说明的是，在主节点发送完 Break 信号后，收到的反馈字节应为 0x00。

3 LIN 驱动函数

在 LIN 的驱动库中，已经有下列驱动函数可供调动，可大大方便用户使用和理解，具体如表 3-1 所示。

表 3-1: LIN 驱动函数列表

函数名	功能及参数说明
uint32_t LIN_ReadByte(uint8_t *pu8Data, uint32_t u32Timeout)	从 LIN 总线上读取一个字节数据
uint32_t LIN_CheckFeedbackByte(uint8_t u8CheckData, uint32_t u32Timeout)	校从 LIN 总线读取反馈字节数据并对其进行校验
uint32_t LIN_WriteByte(const uint8_t u8Data, uint32_t u32Timeout)	向 LIN 总线上发送一个字节数据
uint8_t LIN_CalChecksum(uint8_t u8ID, uint8_t *pu8Data, uint8_t u8Len)	计算 LIN 帧消息的 Checksum
uint32_t LIN_SendBreak(uint32_t u32Baudrate, uint32_t u32Timeout)	发送 LIN 帧消息的 Break 域信号（主节点）
uint32_t LIN_WaitBreak(uint32_t u32Timeout)	等待检测到 Break 域信号（从节点）
uint32_t LIN_SendHeader(uint32_t u32Baudrate, uint8_t u8ID, uint32_t u32Timeout)	发送 LIN 帧消息的帧头 Header（主节点）
uint32_t LIN_WaitHeader(uint8_t *u8ID, uint32_t u32Timeout)	等待检测到 Break 域信号，并接收 LIN 帧消息的帧头 Header（从节点）
uint32_t LIN_SendData(uint8_t u8ID, uint8_t *pu8Data, uint8_t u8Len, uint32_t u32Timeout)	发送 LIN 帧消息的响应 Response
uint32_t LIN_ReceiveData(uint8_t u8ID, uint8_t *pu8Data, uint8_t u8Len, uint32_t u32Timeout)	接收 LIN 帧消息的响应 Response

4 LIN 通信代码实现

4.1 LIN 主节点实现

LIN 主节点的数据收发功能可以直接调用函数 LIN_MasterControl()函数实现。LIN_MasterControl()函数是通过调用表 3-1 中的函数，实现了帧头 Header 的发送以及响应 Response 的发送或者接收。当然，用户也可以根据自身情况使用表 3-1 中的函数实现符合自己需求的 LIN 主节点。

LIN_MasterControl()函数的输入参数说明如下：

LIN_ReadWriteEnum	RnW	：	LIN_READ – 主节点接收响应 Response 数据； LIN_WRITE – 主节点发送响应 Response 数据
uint32_t	u32Baudrate	：	LIN 通信波特率，单位 bps
uint8_t	u8ID	：	要发送的 ID
uint8_t *	pu8Data	：	指针，指向一个数组，该数据用于存放要发送的响应 Response 数据或者接收到的响应 Response 数据
uint8_t	u8Len	：	响应 Response 数据长度
uint32_t	u32Timeout	：	超时

4.2 LIN 从节点实现

LIN 从节点的数据收发功能可以直接调用函数 LIN_SlaveControl()函数实现。LIN_SlaveControl()函数是通过调用表 3-1 中的函数，实现了帧头 Header 的接收以及响应 Response 的接收或者发送。当然，用户也可以根据自身情况使用表 3-1 中的函数实现符合自己需求的 LIN 从节点。

LIN_SlaveControl()函数的输入参数说明如下：

LIN_ReadWriteEnum	RnW	：	LIN_READ – 从节点接收响应 Response 数据； LIN_WRITE – 从节点发送响应 Response 数据
uint8_t *	pu8ID	：	接收到的 ID
uint8_t *	pu8Data	：	指针，指向一个数组，该数据用于存放要发送的响应 Response 数据或者接收到的响应 Response 数据
uint8_t	u8Len	：	响应 Response 数据长度
uint32_t	u32Timeout	：	超时

5 修订记录

表 5-1: 文档修订记录

日期	版本	修改内容
2020-07-12	1	初始版本