南京凌鸥创芯电子有限公司

**LKS08X CAN应用笔记**

@ 2021, 版权归凌鸥创芯所有

机密文件，未经许可不得扩散

目录

[1. 概述 3](#_Toc5709)

[2. 硬件描述 3](#_Toc2089)

[3. 软件使用 3](#_Toc456)

[3.1. 时钟开启，GPIO复用 3](#_Toc24513)

[3.2. CAN模块初始化： 3](#_Toc31749)

[3.3. Id 滤波设置 3](#_Toc159)

[3.4. 发送数据 4](#_Toc13008)

[3.5. 接收数据 4](#_Toc14253)

[4. 例程演示： 4](#_Toc1143)

[5. CAN\_DMA传输 6](#_Toc17127)

[5.1. CAN\_DMA软件配置： 6](#_Toc13409)

[5.2. 硬件连接： 7](#_Toc18439)

[5.3. 测试结果： 8](#_Toc18820)

# 概述

CAN是汽车网络的标准协议，CAN的高性能和可靠性已被认同。

CAN控制器根据两根线上的电位差来判断总线电平，总线电平分为显性电平和隐形电平。发送方通过使总线电平发生变化，将消息发送给接收方。

# 硬件描述

Lks08x 提供can协议的TTL接口，分别为RX,TX全双工接口。与节点设备需要通过硬件转换芯片进行总线连接(注意CAN测试时需连接CAN收发器且CAN收发器供电使用5V供电，或者将芯片CNA\_TX与CAN\_RX进行自测)。如图1，

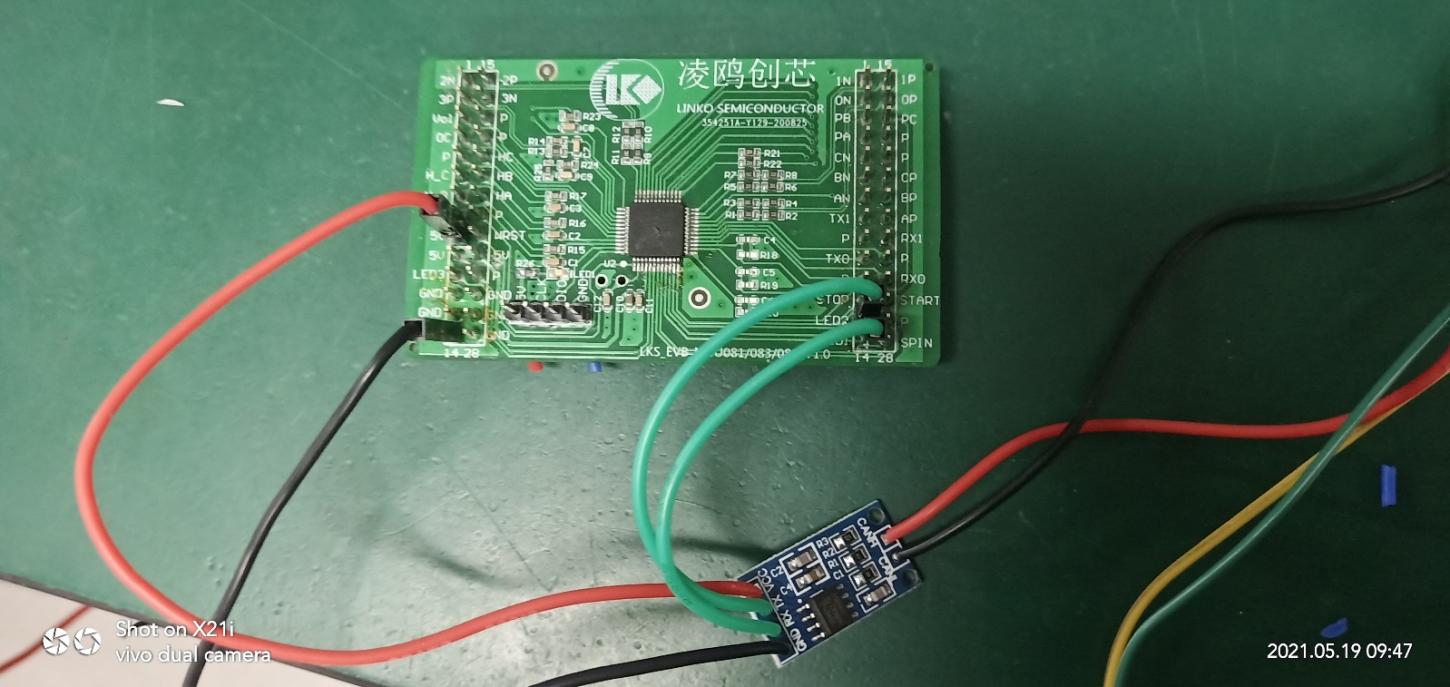


图 1

# 软件使用

## 时钟开启，GPIO复用

## CAN模块初始化：

void CAN\_Init(u8 btr0,u8 btr1)；

通过btr0，btr1配置波特率。在lks08x 用户手册中给出不同波特率下的推荐btr0 和 btr1,经过验证可以可靠运行。该函数同时开启的发送完成中断，接收完成中断，数据溢出中断，仲裁丢失中断和总线错误中断。

## Id 滤波设置

单滤波id设置：

void ID\_Filter(u32 acr,u32 amr,u8 ide);

acr低16位为校对的id值，amr低16位存放需要匹配的位掩码：即amr中的第BITn 为1时，则接收到的id不需要与ACR的第BITn值相同，若amr中的第BITn为0时，则接收到的id的第BITn位需要与ACR的第BITn位相同。如acr = 0X04d1 amr = 0xfffe, 则匹配的id 的为 0Xxxxxxxx1。

ide: 1：id为扩展帧长度格式 0：id为标准帧长度格式

双id滤波设置：

void ID\_Filter\_Dual(u32 acr,u32 amr,u8 ide)；

acr低16位为校对的id1值，amr低16位存放id1需要匹配的位掩码.

acr高16位为校对的id2值，amr高16位存放id2需要匹配的位掩码.

## 发送数据

u8 My\_CAN\_Send\_Msg(u32 id,u8 ide,u8 rtr, u8 \*msg,u8 len)；

id：待发送的id值

ide: 0：标准帧；1：扩展帧

rtr: 0：数据帧；1：远程帧

msg: 待发送的数据区

len: 发送的数据长度

## 接收数据

void CAN\_Receive\_Msg\_IR(u32\* id,u8\* ide,u8\* rtr,u8 \*buf)；

id：获取接收到的id值

ide：1：接收到扩展帧；0：接收到标准帧

rtr：1：接收到远程帧；0：接收到数据帧

buf：获取接收到的数据

# 例程演示：

硬件设备：Lks08x开发板和CAN分析盒。如图2

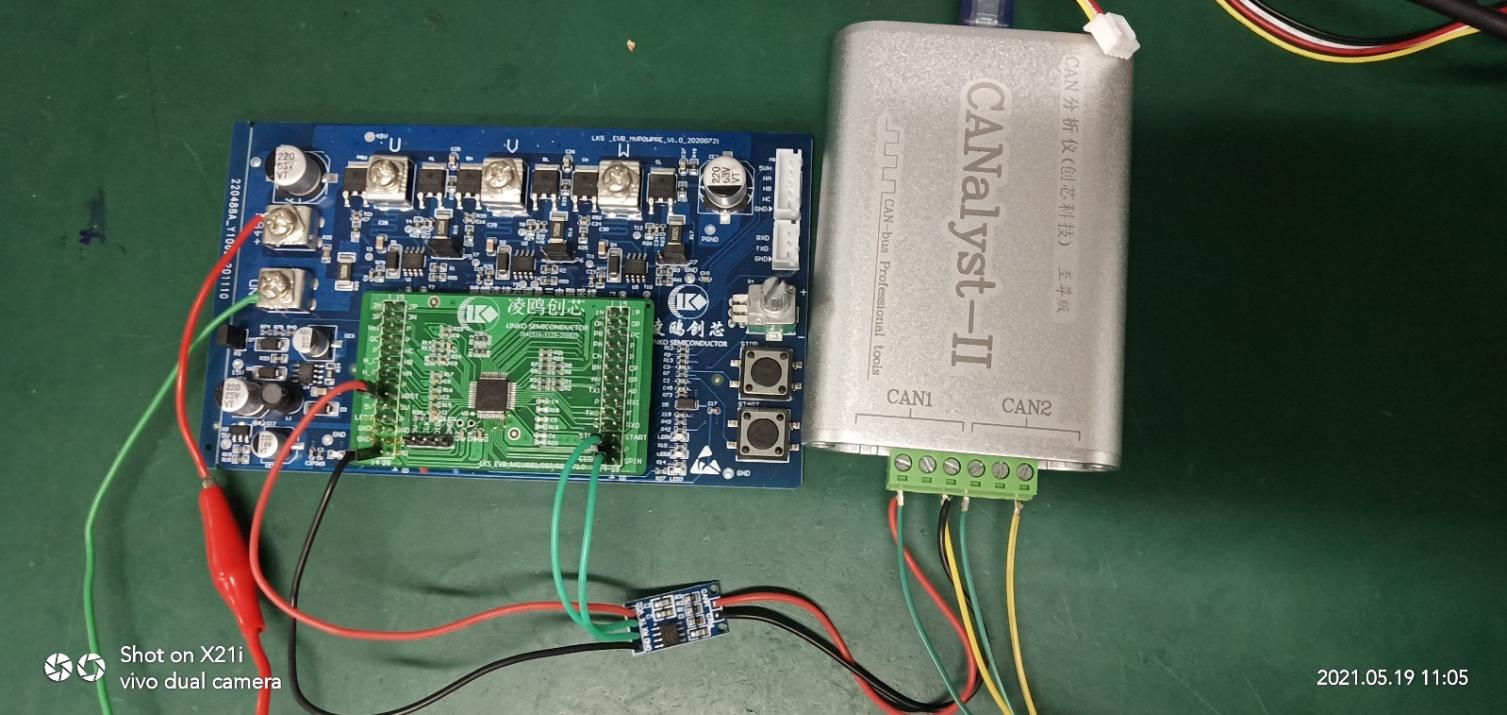


图 2

功能实现：

CAN分析盒发送数据给开发板，开发板再将收到的数据回传。

波特率：500k，发送周期10ms。

测试结果：可以正常收发。如图3

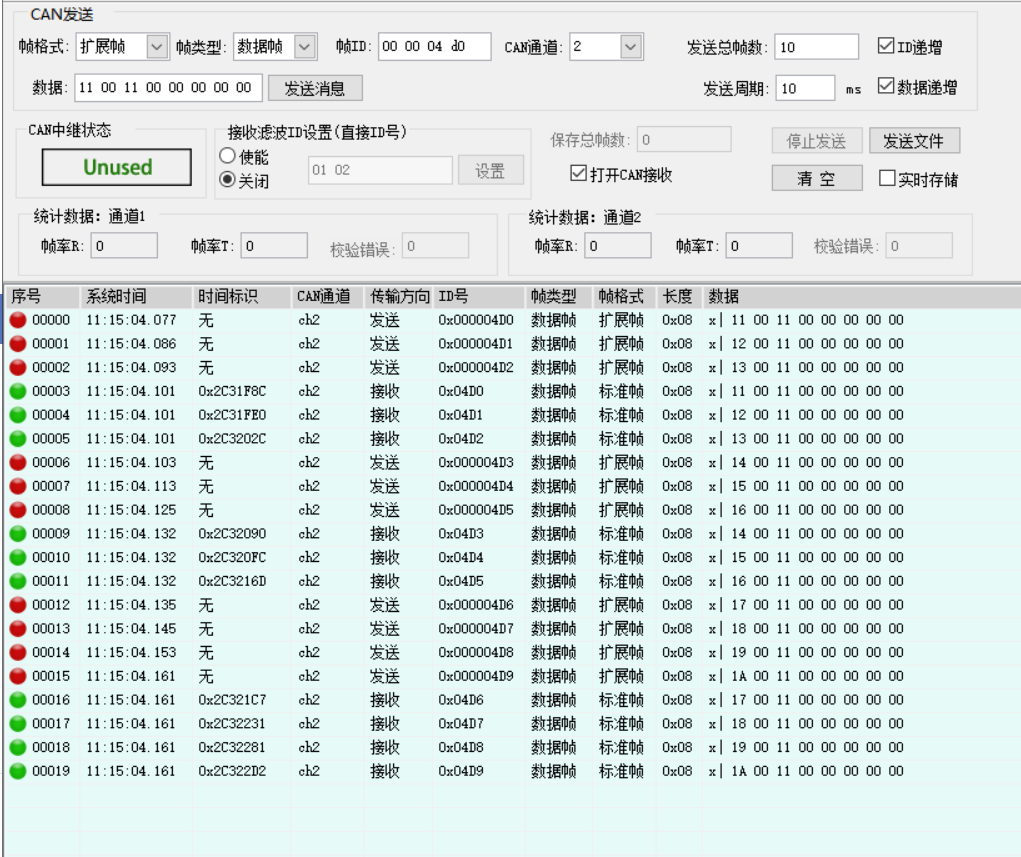


图 3

# CAN\_DMA传输

## CAN\_DMA软件配置：

1、初始化 DMA 模块，将本次发送的数据来源，接收的数据去向配置好，传输长度配置完毕。

2、初始化 GPIO 模块，将 CAN 复用的 GPIO 配置完毕。

3、初始化 CAN 接口，控制寄存器配置完毕。

4、触发 CAN 接口，进入发送状态

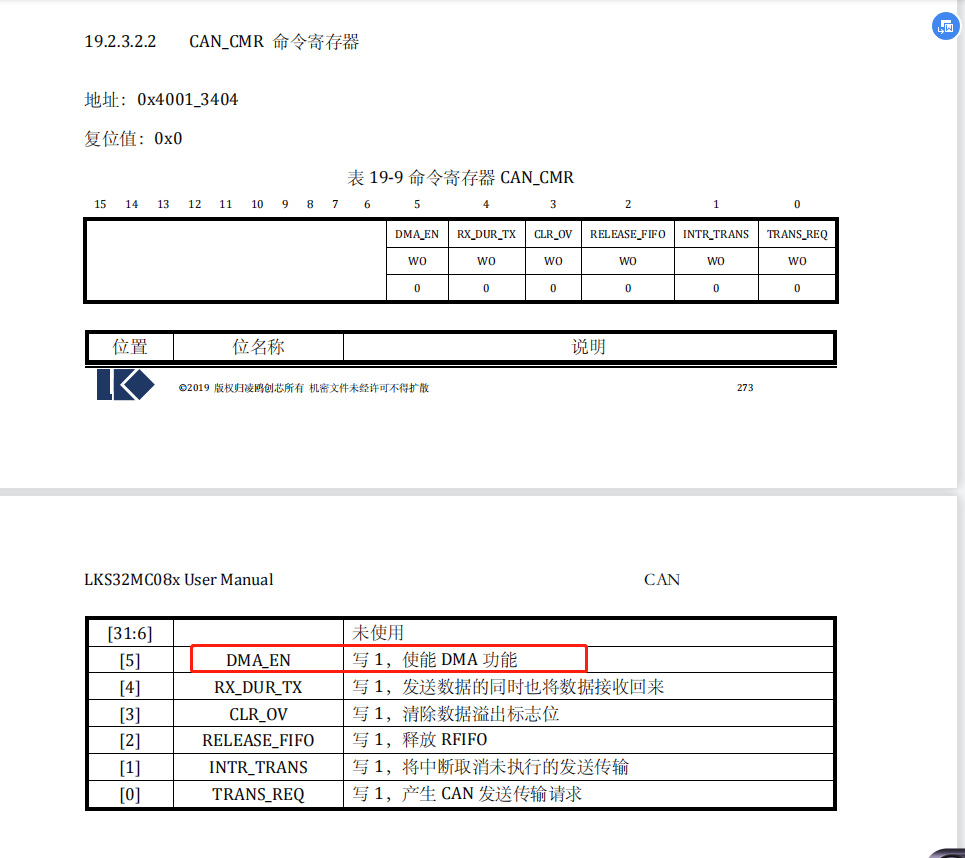
其实CAN\_DMA与CAN\_MCU传输的区别只是将CAN\_CMR的BIT5位DMA\_EN由0置1，然后配置DMA\_CH3即可。

图 4

本CAN 模块设计的 DMA不同于其它模块的DMA搬移操作，需要MCU介入部分搬移操作。假定当前配置CAN模块发送N帧数据，那么第一帧数据需要MCU搬移到CAN模块寄存器中，后续帧（N-1）的数据可由 DMA 实现搬移。

CAN\_TX\_DMA例程中一次发送5帧数据，第一帧数据通过My\_CAN\_Send\_Msg()函数MCU发送给CAN模块，剩余四帧通过DMA搬移到CAN模块进行数据传输。其中需要注意，DMA搬移时数据，数组中每组数据的前三个字节包含了地址、标准帧/数据帧、数据、数据长度信息，后面8个字节是要发送的数据，具体每帧数据的前3个字节如何赋值如下。（该例程采用2.0B协议,SFF地址格式）

frame\_inf |= ide << 7; //ide为0标准帧/1扩展帧

frame\_inf |= rtr << 6; //rtr 为0,数据帧;1,遥控帧

frame\_inf |= len << 0; //len为发送数据长度

CAN\_TX[0]= frame\_inf; //发送TX帧信息

CAN\_TX[1] = id >> 3; //TX ID0

CAN\_TX[2] = (id & 0X07) << 5; //TX ID1

如CAN\_TX\_DMA例程中发送地址0x5555，0标准帧，0数据帧，发0x00,0x01,0x11,0x12,0x13,

0x14,0x15,0x16数据，数据长度8。那一帧数据：CAN\_TX[11]={0x08,0xAA,0xA0,0x00,0x01,0x11,0x12,0x13,0x14,0x15,0x16}。

## 硬件连接：

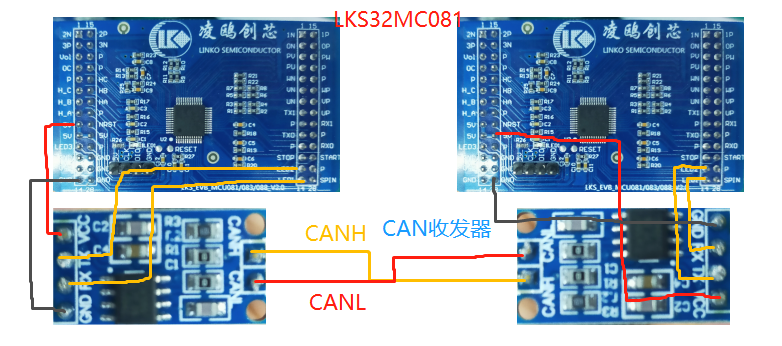
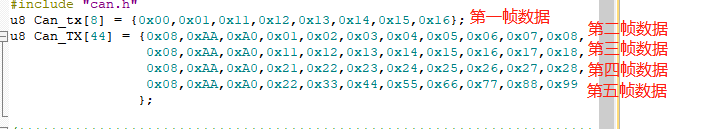


图 5

## 测试结果：

CAN\_TX发送的数据：



CAN\_RX接收的数据：

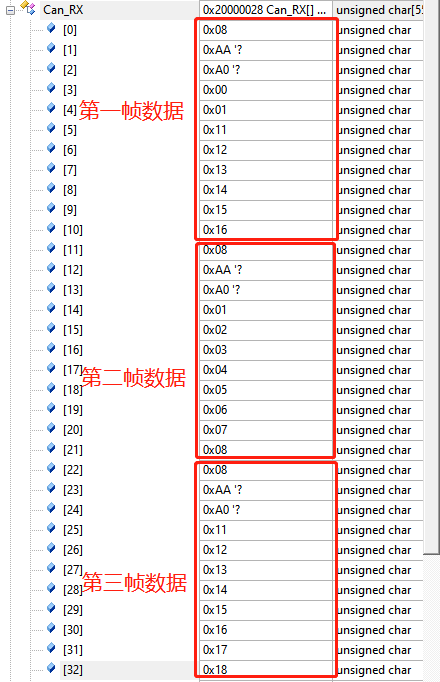
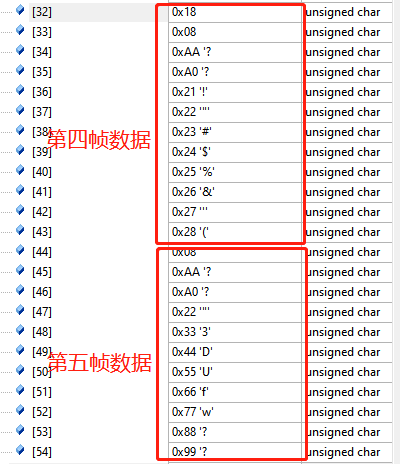
 

图 6