**本 科 实 验 报 告**

实验名称： **CAN总线实验**

学 员： 于鹄杰 学 号： 201506021075

培养类型： 工程技术 年 级： 2015级

专 业：网络工程（军事物联网方向） 所属学院： 计算机学院

指导教员： 逄德明 职 称： 讲师

实 验 室： 实验大楼-612 实验日期： 2017秋季学期

国防科学技术大学训练部制

1. **实验目的**

本实验主要包括CAN总线系列实验，通过实验使学生熟悉CAN总线系统结构，双CAN通信原理，掌握双CAN节点设计以及三子网CAN网关设计。

1. **实验内容**
2. 开发环境搭建；
3. 按键与中断实现；
4. 双CAN节点通信；
5. 三子网CAN网关系统设计。
6. **实验环境**

1、STM32F4xx\_DSP\_StdPeriph\_Lib\_V1.4.0软件；

2、Windows 操作系统；

3、联网PC机；

4、Secure CRT软件；

5、硬件：STM32F4xx、MCP2515、407VET6。

1. **实验结果**
2. 用框图描述一个4个CAN口的CAN网关系统，并说明各个部件在CAN协议中属于哪一层。

如图1，

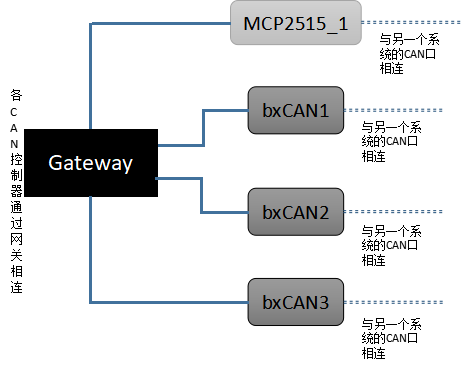


图 1 4个CAN口的CAN网关系统图

上述为4个CAN口的CAN网关系统，其中CAN控制器MCP2515和bxCAN通过与其它系统相连，接收消息并发送消息到网关Gateway，网关Gateway根据转发策略进行转发，发到MCP2515\_1或者bxCAN，再由MCP2515\_1和bxCAN将消息发送出去。

其中网关Gateway属于应用层，CAN控制器MCP2515和bxCAN属于数据链路层，CAN口和收发器属于物理层。

1. 描述CAN帧结构，并说明软件实现时应该如何实现一个CAN帧。

如图2，

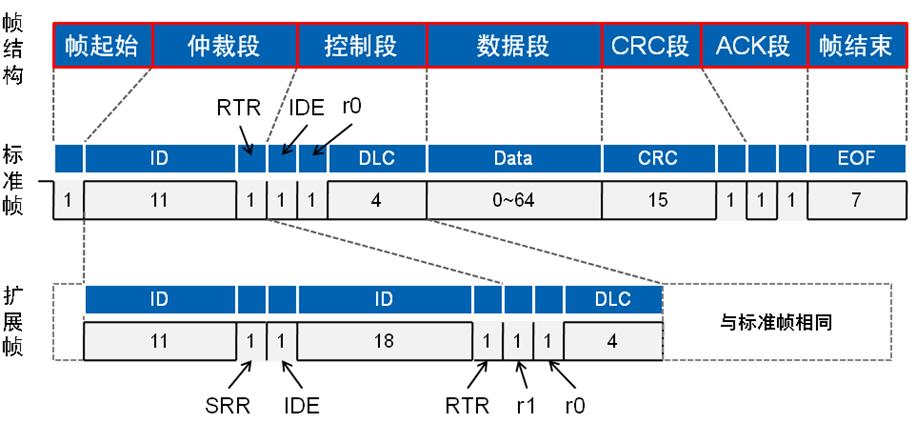


图 2 CAN数据帧结果图

CAN数据帧有两种，标准帧和扩展帧。一个CAN标准数据帧的结构由7部分组成：帧起始（SoF，Start of Frame）、仲裁域（Arbitration Field）、控制域（Control Field）、数据域（Data Field）、CRC域（CRC Field）、应答域（ACK Field）、帧结尾（End of Frame）。

标准格式数据帧的最小位数为44，最大位数为108。一般地，将帧起始、仲裁场和控制场作为CAN头部（19位），CRC场、应答场和帧结束作为CAN尾部（25位）。

**帧起始**：标志着数据帧和远程帧的起始，仅由一个“显性”位（“0”）构成。该状态表明某个阶段开始发送消息，所有的站都必须同步于首先开始发送的那个站的帧起始前沿；错误帧和超载帧没有“帧起始”。

**仲裁场**：包括标识符（ID）和远程发送请求（RTR）。标识符的长度为11位，这些位以ID-10至ID-0的顺序发送，最低位为ID-0，其中最高7位（ID-10 ~ID-4）必须不是全“隐性”； RTR位在数据帧中，必须是“显性”的，而在远程帧中，RTR位必须是“隐性”的。

**控制场**：包括2位将来用于扩展的保留位（IDE、r0）和4位数据长度码（DLC）。

**数据场**：由数据帧中被发送的数据组成，它可包括0至8个字节，每个字节包括8位，其中首先发送最高有效位。

**CRC场**：包括CRC序列、CRC界定符；参加CRC校验的位场包括帧起始、仲裁场、控制场、数据场（假若存在）在内的无填充位流；CRC序列后跟随CRC界定符，它只包括一个“隐性”位。

**应答场**：包括应答间隙和应答界定符（2位）。发送器在应答场发2个“隐性”位；接收器正确接收后，在应答间隙发送1个“显性”位（接收器发送“应答”）。

**帧结束**：每个数据帧和远程帧均是由7个“隐性”位组成的标志序列界定的。

**软件实现：**

在软件中，在stm32f4xx\_can.h中定义了CAN帧的结构：CanTxMsg（用于发送）和CanRxMsg（用于接收）。如图3:

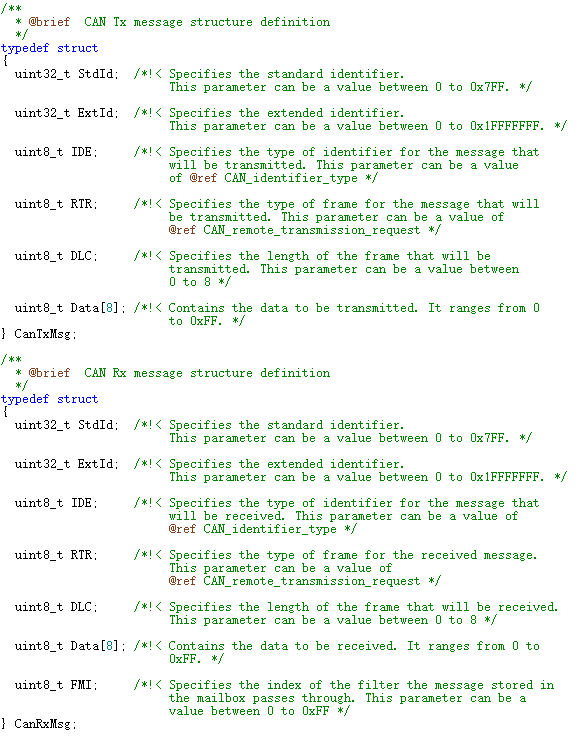


图 3 CAN帧在软件中定义

从图3可以看出，除了以上所说的CAN帧结构外，接收帧和发送帧之间在帧结构的定义上存在一点不同：接收帧额外有一个FMI位，这个位主要是用于进行接收的过滤，可以用来过滤掉一些不需要的数据。

在软件中，通过为帧结构中参数赋值来构建帧,如图4：

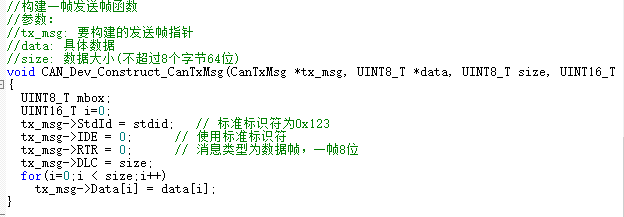


图 4 发送帧函数构建帧

1. 按CAN协议层，描述实验三中双CAN通信数据的流向。

实验三要求由bxCAN1发送数据，由bxCAN2接收数据，如图5是bxCAN1的发送帧函数：

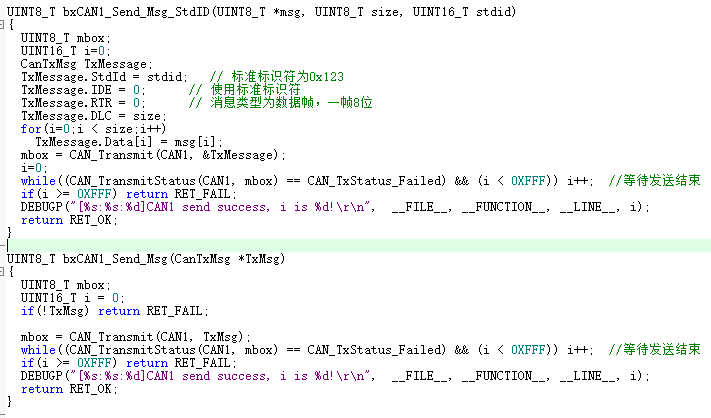


图 5 bxCAN1发送帧函数

如图6是bxCAN2的接收函数：

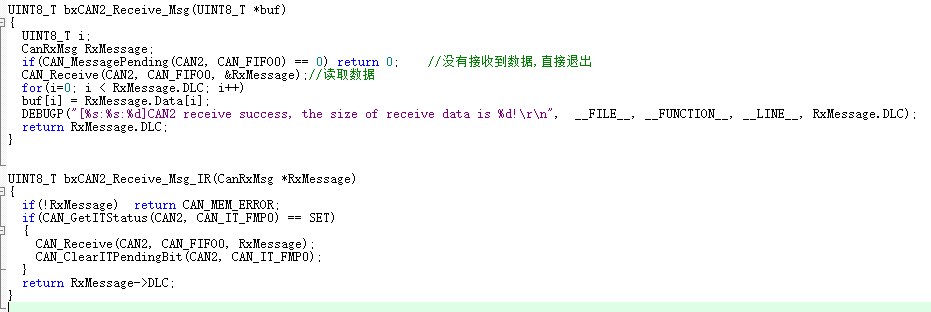


图 6 bxCAN接收帧函数

具体流向过程如下：

①应用层中将数据封装入CAN帧中，组成合法的帧结构；

②调用函数bxCAN1\_Send\_Msg\_StdID()，在物理层上CAN1口通过设定好的总线速率等参数向外发送封装好的数据；

③封装好的数据在链路层上通过导线向外传送；

④由于bxCAN2设置为和bxCAN1相等的总线速率，CAN2口会注意到该数据。该数据从链路层流入bxCAN2的物理层后，若有滤波机制则检查帧标识符，然后向上传送数据，此时数据从bxCAN2的物理层流入CAN应用层；

⑤解封，将数据从CAN帧中读出，通过printf()函数打印在显示屏上。

1. 描述实验四的实验系统，其中各个部分是如何模拟的，CAN数据是如何流向的，写出自己的理解。

如图7为实验四的实验系统：

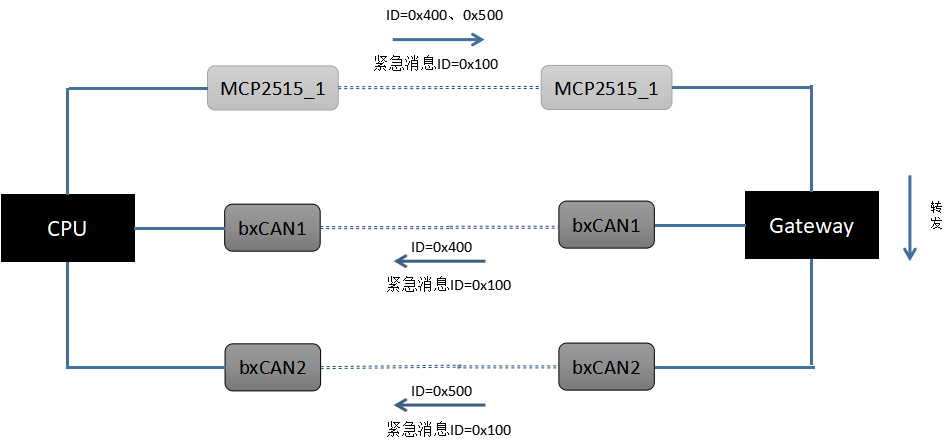


图 7 实验四结构图

实验四的实验系统可分为两部分，CPU和网关。具体结构功能如图8和图9：

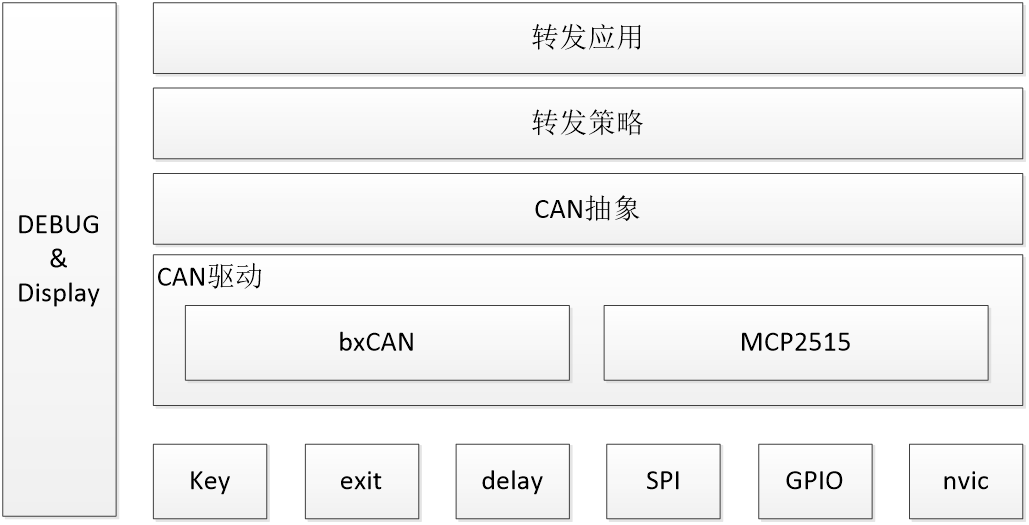


图 8 Gateway系统图

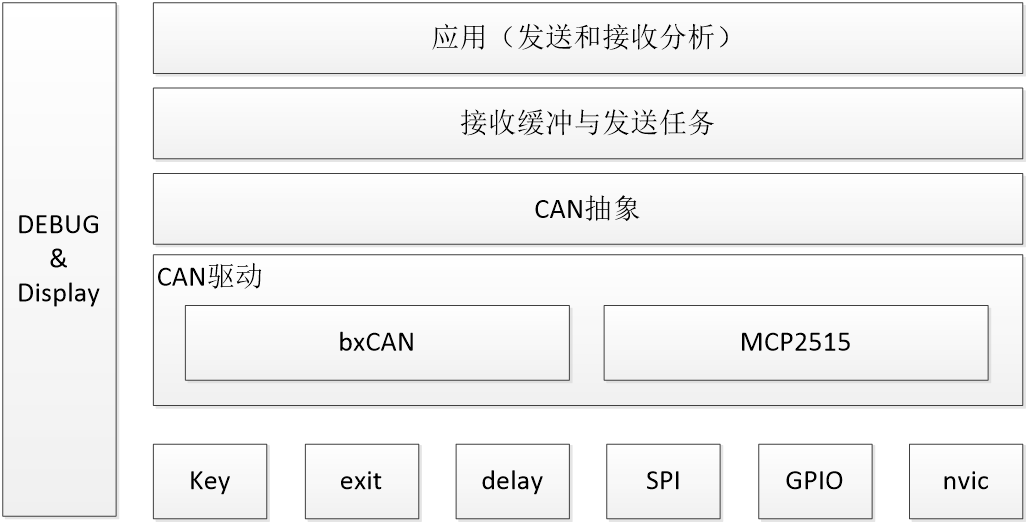


图 9 CPU系统图

模拟过程和数据流向过程如下：

实验四由两个组合作完成，一组的实验板模拟CPU，另一组的实验板模拟网关，每块实验板上外接MCP2515\_1芯片；两块实验板之间bxCAN1，bxCAN2，和MCP2515对应端口相连接，到此完成实验系统模拟组装。

分以下两种情况讨论该实验中数据流向：

1. 当不按下key2时，即发送的数据帧标识符为“0x400-0x5FF”，此时为低优先级非紧急消息。

①由CPU通过MCP2515向外发送数据帧；

②Gateway组的MCP2515接收数据并将其加入缓冲区队列（若满则丢弃报错）；

③Gateway从收缓存队列取数据帧，检查其标识符，若在0x400~0x4FF内，则加入bxCAN1的发缓存队列（若满则丢弃并报错）；若该标识符在范围0x500~0x5FF内，则加入bxCAN2的发缓存队列（若满则丢弃并报错）；若都不在，则“throw it!”。

④Gateway依次检查bxCAN1和bxCAN2的发缓存队列，若非空，则分别向CPU对应的CAN口发送数据后加入到CPU对应的收缓存队列（若满则丢弃并报错）。

⑤CPU分别从bxCAN1和bxCAN2的收缓存队列的队头取出一个数据帧，解封后通过printf()函数打印在显示屏上。

1. 当按下key2时，即发送的数据帧标识符为“0x100-0x1FF”，此时为高优先级紧急消息。

①由CPU通过MCP2515向外发送数据帧；

②Gateway组的MCP2515接收数据并将其加入缓冲区队列（若满则丢弃报错）；

③Gateway从收缓存队列取数据帧，检查其标识符，若在0x100~0x1FF内，则加入bxCAN1和bxCAN2的发缓存队列（若满则丢弃并报错）；若不在，则“throw it!”。

④Gateway依次检查bxCAN1和bxCAN2的发缓存队列，若非空，则分别向CPU对应的CAN口发送数据后加入到CPU对应的收缓存队列（若满则丢弃并报错）。

⑤CPU分别从bxCAN1和bxCAN2的收缓存队列的队头取出一个数据帧，解封后通过printf()函数打印在显示屏上。

1. 描述你在实验四中所负责模块的实现原理，如果可以，讲出你遇到了什么问题（实现、和同伴组合），如果有调试记录可以一并附上。

在实验四中，我主要负责Gateway组的转发应用层工作，也做了一部分的转发策略结构构建工作，主要是构建完转发策略结构后应用它，并完成转发，具体便为routing.c的编写,如图10、11、12、13。

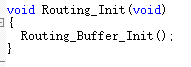


图 10 Routing\_Init函数

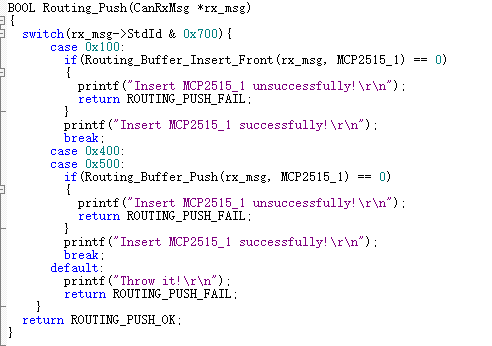


图 11 Routing\_Push函数

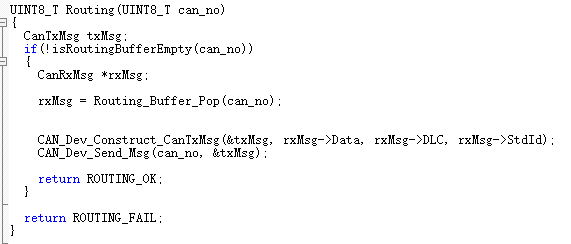


图 12 Routing函数

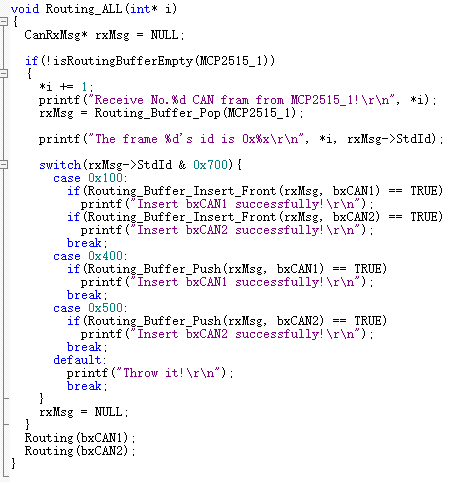


图 13 Routing\_All函数

Routing\_Init()函数中调用了Routing\_Buffer\_Init()对缓存队列进行初始化。

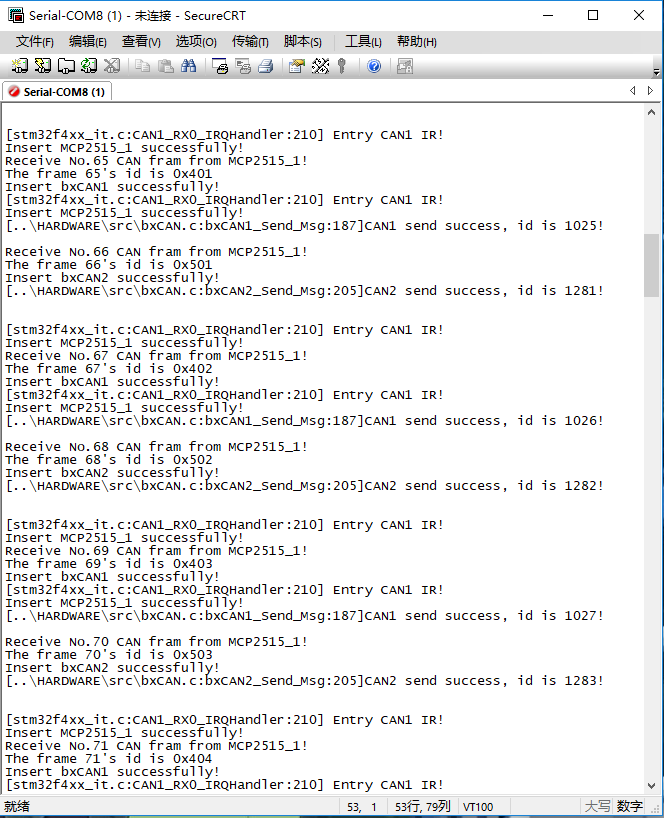
当MCP2515收到数据的时候，会调用Routing\_Push()函数，将收到的帧加入收缓存队列。

判断MCP2515收缓存队列的状态，如果非空，则从收缓存队列中取出一帧放入对应的发缓存队列，然后根据数据帧标识符调用Routing（）函数对bxCAN1和bxCAN2进行转发。

1. 描述你们组合完成后的实验现象，附上实验结果，并从数据产生到数据流动的过程来分析一下为什么会出现这样的结果。

两组合作完成实验后，实验结果如图：

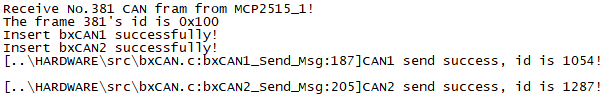
1. 当发送非紧急消息时（0x400-0x5FF）：



当CPU发送的数据帧标识符在0x400-0x4FF时，Gateway的MCP2515接收到消息后，转发到bxCAN1口，再发送到CPU的bxCAN1.

当CPU发送的数据帧标识符在0x500-0x5FF时，Gateway的MCP2515接收到消息后，转发到bxCAN2口，再发送到CPU的bxCAN2.

1. 当发送紧急消息时（0x100-0x1FF）：



当CPU发送的数据帧标识符在0x100-0x1FF时，Gateway的MCP2515接收到消息后，转发到bxCAN1口和bxCAN2口，再对应发送到CPU的bxCAN1和bxCAN2.

1. **实验心得**

通过此次实验，我对CAN总线通信有了更深一步的了解。实践是检验真理的唯一标准。上课的时候，我只是学了CAN总线的帧结构及通信原理，但对其通信的具体过程并不清楚，但在实验具体实践的过程中，虽然遇到了不少的问题，但通过阅读资料和与小组成员合作，我对帧的结构、帧过滤机制、节点收发机制进一步熟悉了。

最后非常感谢黄教员这些天来的耐心授课以及悉心解疑，不仅带给了我们知识，也带来了舒适的环境。