基于 C8051F040 CAN 总线的节点通信研究①

Research on Nodes Communication Based on CAN Bus of C8051F040

宋　薇　刘晓洁　韩润萍　(北京服装学院 信息工程学院 北京 100029)

摘　要： 简要介绍了 C8051F040 单片机 CAN 控制器的结构，设计了 CAN 通信节点的硬件电路，并详细阐述了一种基于 FIFO 技术的 CAN 多消息通信的软件设计方法。硬件电路采用 CAN 收发器 TJA1040 建立了 C8051F040 CAN 控制器与 CAN 总线之间的连接，并由 6N137 与 TLP521 实现了光耦隔离。软件部分对数据发送节点与接收节点的通信程序作了分别介绍，并由较详细的源代码说明 CAN 多消息通信中一些关键寄存器的设置。文中所介绍的软件设计方法具有一定的通用性。

关键词： C8051F040 CAN 总线 通信节点 FIFO

控制器局域网 CAN(Controller Area Network)是由德国 Bosch 公司在 80 年代初为解决现代汽车内部测量与执行部件间大量的数据交换而提出的一种串行通信协议。CAN 支持多主工作方式，通信可以使用多种物理介质，如双绞线、光纤等，通信速率最高可达 1Mbit/s，通信距离最远可达 10km。CAN 总线规范现已广泛应用在工业控制领域，是公认的最有前途的现场总线之一。

1　引言

Cygnal 公司开发的 51 系列单片机 C8051F040 是完全集成的混合信号片上系统型微控制器[1]。本文首先利用 C8051F040 自带的 CAN 控制器和外接的 CAN 收发器设计了 CAN 通信节点的硬件电路，然后详细介绍了一种可用于多消息数据传输的 CAN 通信软件设计方法。

2　通信节点的硬件设计

文中 CAN 通信节点由带有 CAN 控制器的微处理器和 CAN 收发器组成。CAN 收发器建立 CAN 控制器和物理总线之间的连接，控制逻辑电平信号在 CAN 控制器与物理总线之间的传输，CAN 控制器执行 CAN 协议，用于信息缓冲和滤波[2]。下面先介绍 C8051F040 的 CAN 控制器。

2.1 C8051F040 的 CAN 控制器

C8051F040 的 CAN 控制器主要是由 CAN 核、消息 RAM、寄存器、消息处理器和模块接口构成。CAN 核用于协议控制和消息串并转换；消息 RAM 用于存储消息对象和标志符；寄存器用来控制与配置整个 CAN 控制器；消息处理器用于控制 CAN 核和消息 RAM 之间的数据传输；模块接口用于 CPU 与整个 CAN 控制器交换数据[3]。C8051F040 的 CAN 控制器有 32 个消息对象，可以被配置为发送或接收数据。所有数据发送和接收的协议处理由 CAN 控制器独立完成，不用 CIP-51 干预，因此 CAN 通信占用的 CPU 带宽很小。

2.2 硬件实现

C8051F040 的 CAN 控制器是一个协议控制器，不提供物理层驱动器，需外接 CAN 收发器件才能挂接在 CAN 网上与其它节点进行通信。本文采用的 CAN 收发器为 TJA1040。TJA1040 是高速 CAN 收发器 PCA82C250 的后继产品，提供了与 PCA82C250 基本相同的功能，此外还采用了最新的 EMC 技术，抗电磁干扰性能有了很大提升。另外为了实现 CAN 总线与 C8051F040 单片机系统的完全隔离，硬件设计中采用了光耦隔离器件 6N137 和 TLP521。节点硬件原理

① 基金项目:北京市教育委员会科技发展计划面上项目(KM200810012005)
收稿时间:2008-10-09

图如图 1 所示：

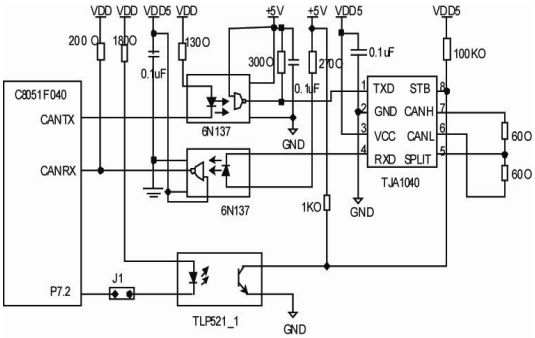

图 1　CAN 通信节点硬件原理图

3　通信节点的软件设计方法

C8051F040 的 CAN 控制器是一个协议控制器，不提供物理层驱动器，需外接 CAN 收发器件才能挂接在 CAN 网上与其它节点进行通信。本文采用的 CAN 收发器为 TJA1040。TJA1040 CAN 网上的通信有单消息通信、多消息通信、远程帧通信等。单消息通信只涉及到一个消息对象，实现起来比较简单。远程帧通信实际上是要求接收数据的节点发送一个远程请求帧，请求远程节点回复一个数据帧。多消息通信是将多个消息对象中的数据一次性地进行收发。由于涉及到多消息通信的资料甚少，本文特以此为例来说明 CAN 通信的软件设计方法。

基于 C8051F040 的 CAN 通信程序涉及众多寄存器的使用。通过编程实践，笔者认为值得重点关注的寄存器有：命令掩码寄存器、消息控制寄存器、仲裁寄存器。准确地设置这些寄存器是 CAN 通信功能能否实现的关键所在。

文中通信实验是在两个硬件结构完全相同的节点之间进行的。通信介质为双绞线，通信距离为 20m。PC 机通过 JTAG 口对节点通信程序进行在线调试与维护。如下通信程序中设定节点 1 为数据发送端，节点 2 为数据接收端，多消息通信程序的基本功能为：将待发送的 2 个长度均为 8 byte 的数组数据(ss1、ss2)写入节点 1 的 2 个消息对象中，然后传送至节点 2，节点 2 收到节点 1 发送来的消息报文后产生中断，在其中断服务程序中，通过接收函数将接收到的消息报文读入接收数组(rr1、rr2)中。节点 1 的通信程序包括 CAN 初始化和发送模块，节点 2 的通信程序包括 CAN 初始化和接收模块。由于一个数据帧最多只能传送 8 byte 的数据量，因此应用了一种 FIFO(First In First Out)技术，实现了一次性传送 2 个数据帧。本文使用的软件开发工具为 Cygnal IDE，编程语言为 C。下面将逐一介绍节点 1、2 的通信程序。

3.1 节点 1 的通信程序

3.1.1 CAN 初始化

CAN 初始化的内容包括时钟设置、I/O 口配置、CAN 启动、消息对象初始化。系统时钟源采用外部晶振，为 22.1184MHz。通过 I/O 口配置将 CANTX (CAN 发送脚)配置为推挽输出模式。下面介绍 CAN 启动和消息对象的初始化。

3.1.1.1 CAN 启动

CAN 启动包括波特率和 CAN 初始状态的设置。在同一个 CAN 网络中只能使用同一种波特率进行通信。若某个节点波特率设置错误，则会影响整个 CAN 网络通信。下为 CAN 启动部分的源码：

```
void start_CAN(void)
  { SFRPAGE = CONFIG_PAGE;
    STB1040 = 0x00; //模式转换：从隐性到显性
    SFRPAGE = CAN0_PAGE;
    CAN0CN |= 0x41;  //置位 CCE 和 INIT，开始波特率设置
    CAN0ADR= BITREG; // 指向 Bit Timing 寄存器
    CAN0DAT = 0x2640; // 波特率设置为 1Mbps，包括了缓冲段
    CAN0CN |= 0x06;  // 允许错误中断和状态变化中断
    CAN0CN &= ~0x41; }   //清除 CCE 和 INIT 位，启动 CAN
```

3.1.1.2 消息对象的初始化

应用 FIFO 技术传输多帧数据时有一点需要特别注意，即消息控制寄存器(IFx Message Control Register)中 EOB 位的设置。对于组成数据块多个消息对象中的最后一个消息对象，该位置 1，对于其它消息对象，该位置 0。在传输单数据帧的场合，消息对象的 EOB 位必须置 1[4,5]。节点 1 消息对象的初始化程序如下：

```
void init_FIFO_Buffer(void)
 { char MsgNum;
    SFRPAGE = CAN0_PAGE;
    CAN0ADR = IF1CMDMSK; // 指向 IF1 命令掩码寄存器
```

```
    CAN0DAT = 0x00B3;          //设定方向为写，改变除了 Mask 位外所有的消息对象信息
    CAN0ADR = IF1ARB2;         //指向 IF1 仲裁寄存器 2
    CAN0DAT = 0xA000;          //设定消息对象有效且报文采用标准格式，传输方向为发送
    For(MsgNum=1; MsgNum<3; MsgNum++)
    {if( MsgNum= =1 )          //对 1 号消息对象
      {CAN0ADR = IF1MSGC;
      CAN0DAT = 0x0008;}// IF1 消息控制寄存器，EOB=0，数据块没结束；数据帧长度为 8 byte
    else                       //对 2 号消息对象
    {CAN0ADR = IF1MSGC;
      CAN0DAT = 0x0088;} // EOB=1，数据块结束
    CAN0ADR = IF1CMDRQST; // 指向 IF1 命令请求寄存器
    CAN0DATL = MsgNum;}} //将以上配置写入 1、2 号消息对象中
```

3.1.2 发送模块

发送模块的功能是将待发送的数组数据由消息对象读入发送缓冲区(IF1)中，整个发送模块由 2 个函数组成，源程序如下：

```
void transmit_message ( char MsgNum , unsigned char *p )
{unsigned char j;
    SFRPAGE = CAN0_PAGE;
    CAN0ADR = IF1CMDMSK;
    CAN0DAT = 0x0087;          //方向为写，改变消息对象 8 个 byte
    CAN0ADR = IF1ARB2;
    CAN0DATH |= 0x80;          //消息对象有效，标准帧格式，方向为发送
    CAN0ADR = IF1DATA1;        //指向数据寄存器
    For(j=0; j<8; j+=2)
    { CAN0DATH =*p;
      p++;
      CAN0DATL =*p;
      p++;}                    //写入 8 byte 数据
    CAN0ADR = IF1CMDRQST;
    CAN0DATL = MsgNum;}        //发送数据帧至消息对象中
  void transmit_FIFO_block（void）//传送整个 FIFO 数据块
  {unsigned char blockIndex=1;
    transmit_message(blockIndex，ss1);  //传送数组 ss1
    blockIndex++;
    transmit_message(blockIndex，ss2);} //传送数组 ss2
```

3.2 节点 2 的通信程序

3.2.1 CAN 初始化

在接收端的 CAN 初始化中，时钟设置、I/O 口配置、CAN 启动跟发送端一致，在此不再赘述。消息对象的初始化与发送端的情况稍有不同，其源代码如下：

```
void init_FIFO_Buffer（void）
{ char MsgNum;
    SFRPAGE = CAN0_PAGE;
    CAN0ADR = IF2CMDMSK;
    CAN0DAT = 0x00B8;
    CAN0ADR = IF2ARB2;
    CAN0DAT = 0x8000;          //消息对象可用，报文采用标准格式且传输方向为接收
    For(MsgNum=1;MsgNum<3; MsgNum++)
    { if(MsgNum= =1)           //对 1 号消息对象
      {CAN0ADR = IF2MSGC;
      CAN0DAT = 0x0008;  // 数据块没结束
      } else                   //对 2 号消息对象
      {CAN0ADR = IF2MSGC;
    CAN0DAT = 0x2488;}}} //当报文成功接收，IntPnd 自动置 1；数据块结束
```

3.2.2 接收模块

节点 2 的通信程序中需要重点说明的是接收函数。由于应用了 FIFO 技术，为此在接收函数中，先读取 IF2 消息控制寄存器(IF2 Message Control Register)，判断其 NewDat 位是否已置位，若已置位，则将接收缓冲区中的数据读入接收数组中。下为接收函数的源代码：

```
void downloadFIFO(unsigned char *p，unsigned char *q)
{unsigned char j，MsgNum;
    SFRPAGE = CAN0_PAGE;
    CAN0ADR = IF2CMDMSK;
    CAN0DAT = 0x0033;
    MsgNum =1;
    CAN0ADR = IF2CMDRQST;
    CAN0DATL = MsgNum;
```

```
    CAN0ADR = IF2MSGC;
      If((CAN0DAT&0x8000)!=0)//判断 NewDat 位是否为 1
       { CAN0ADR = IF2DATA1;
          For(j=0; j<8; j+=2)
       { *p=CAN0DATH;
          p++;
        *p=CAN0DATL;
    p++;}}                    //将缓冲区中的数据读入第一个接收数组中
    MsgNum++;
    CAN0ADR = IF2CMDRQST;
    CAN0DATL = MsgNum;
    CAN0ADR = IF2MSGC;
    If((CAN0DAT&0x8000)!= 0)
     { CAN0ADR = IF2DATA1;
    For(j=0; j<8; j+=2)
     { *q=CAN0DATH;
    q++;
    *q=CAN0DATL;
    q++;}}}                   //将缓冲区中的数据读入第二个接收数组中
```

在中断服务程序中调用接收函数来完成数据的接收[6]。进入中断服务程序时，先读 CAN 状态寄存器 CAN0STA，判断其 RxOK 位是否为 1，若为 1，说明 CAN 成功接收到新的报文，然后调用接收函数读取。中断服务源程序如下：

```
void CAN_ISR(void)interrupt 19
{ SFRPAGE = CONFIG_PAGE;
  EA = 0;
  SFRPAGE=CAN0_PAGE;
  status = CAN0STA;          //读状态寄存器的值
  if((status&0x0010)!= 0)    //判断 RxOK 位是否为 1
   { downloadFIFO（rr1，rr2）}      //调用接收函数将收到的报文存入接收数组 rr1、rr2 中
  SFRPAGE = CONFIG_PAGE;
  EA = 1;}
```

上述程序运行结果如图 2 所示，可以看出接收端数组数据与发送端的完全吻合，验证了设计的正确性。

4　结论

本文基于 C8051F040 单片机自带的 CAN 控制器，设计了通信部分的硬件电路。针对多消息通信程序设计中令人难以掌握的众多寄存器的设置问题，本文通过仅实现双节点间传输 2 个数据帧的通信程序代码，简明有效地说明了其中涉及的寄存器的设置方法，同时也给出了多消息通信程序设计的基本方法。

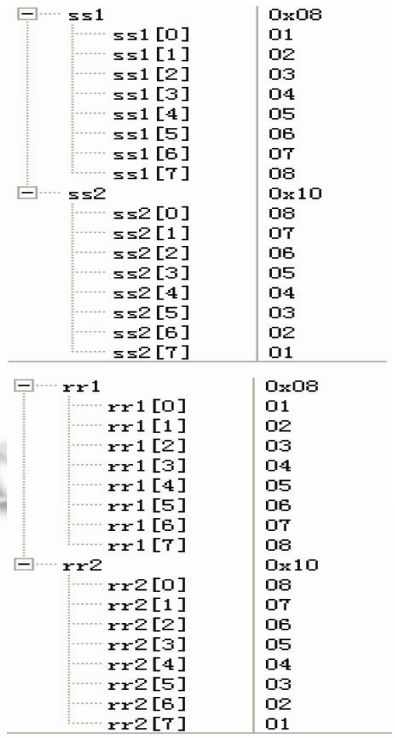

图 2　程序运行结果对照图

参考文献

1 李刚,林凌.与 8051 兼容的高性能,高速单片机－C8051Fxxx.北京:北京航空航天大学出版社, 2002:1－4.

2 潘佚.基于 C8051F040 的 CAN 总线智能节点的设计.现代电子技术, 2006,219(4):49－51.

3 Robert Bosch GmbH.C_CAN User's Manual.Revision 1.2, 2000:29－36.

4 童长飞.C8051F 系列单片机开发与 C 语言编程.北京:北京航空航天大学出版社, 2005:212－213.

5 潘琢金译.C8051F04X 混合信号 ISP FLASH 微控制器数据手册.Rev1.4,新华龙电子有限公司, 2004:197－203.

6 胡晓拓,等.对 C8051F040 的 CAN 控制器的分析及应用.微计算机信息, 2007,11(2):105－107.