doi: 10.3969/j.issn.1006-1576.2011.03.006

基于 CAN 总线的星上交互式仿真框架

程龙，蔡远文，刘党辉
（装备指挥技术学院 航天装备系，北京 101416）

摘要：为解决星上仿真系统的接口规范以及仿真交互协调的问题，建立基于 CAN 总线框架的交互式星上仿真系统。依照各组成部分进行建模，形成多个仿真成员，在各成员驱动程序接口之上进行二次封装，形成统一接口，再配合 CAN 总线仿真框架，构建星上仿真系统。设计并实现了各仿真成员之间统一的通信接口，规范了仿真流程框架，构成了各仿真成员相互交互、协调运行的基础。实验结果证明，该仿真框架传输效率高、运行稳定、适应性好，达到了预期效果。

关键词：CAN 总线；仿真；框架；卫星
中图分类号：N945.13; V474　　**文献标志码**：A

On-Board Interactive Simulation Framework Based on CAN Bus

Cheng Long, Cai Yuanwen, Liu Danghui
(Dept. of Space Equipment, Institute of Command & Technology of Equipment, Beijing 101416, China)

Abstract: To solve the problem of interfaces and communications between different members in an on-board simulation system, an on-board interactive simulation system based on CAN bus is established. Several simulation members are built according to the composition of real system. Interfaces of drivers included in simulation members are packaged into a uniform interface, thus the on-board simulation system is established with the foundation of CAN bus simulation framework. Uniform communication interface of each simulation member is designed and realized, the framework of simulation flow is established, and foundation of intercommunion and operation between simulation members is laid. Experiment results prove that the framework can work stably with great efficiency and high flexibility, and expected effect is achieved.

Keywords: CAN bus; simulation; framework; satellite

0　引言

近年来，人们对航天器的应用日益增加，对航天器的功能集成度、信息传输与处理能力等提出了更高的要求，而这些都极大地依赖于星上数据传输总线的性能。CAN 总线支持分布式控制，在汽车与工业控制系统中的应用验证了其数据传输速率、可靠性等与 1553B 相当[1]。由于其性能及价格的优势，在工业控制以及卫星上得到广泛使用[2]，尤其在卫星应用方面，完全可以取代昂贵的 1553B 总线[3]。目前，已有不少卫星采用了 CAN 总线作为命令与控制总线[4]或数据传输总线，而且在将来的小卫星机内测试技术[5]和卫星编队技术[6]中，CAN 总线也是一种非常适合的选择。故需要建立基于 CAN 总线框架的交互式星上仿真系统。

在基于 CAN 总线框架的交互式星上仿真系统中，星上各分系统（部分）运行自身的模型，这些模型既可以是实物的，也可以是数字的，只要该分系统对外接口符合 CAN 总线协议即可，也不必限制其内部的具体仿真运行形式。各分系统再通过 CAN 总线相互收发信息，组成一个完整的星上仿真系统。

而建立基于 CAN 总线框架的交互式星上仿真系统，是一个系统集成的过程，涉及各部分的接口规范及其协调问题。如星上各分系统采用的 CAN 接口控制器可能不同，底层驱动各不相同，这就要求我们必须按照软件工程化中通用化、标准化的要求对仿真系统中的总线传输框架进行规范；再如，各个分系统的运行并非完全独立，它们互相之间需要有信息的交互，如何统一信息交互的顺序，保证仿真时间的稳定推进，是整个系统能合理有效运行的前提。

笔者对基于 CAN 总线的交互式仿真框架进行设计与实现，以解决上述星上仿真系统的接口规范以及仿真交互协调的问题。

1　仿真系统结构

卫星通常可划分为有效载荷和卫星平台 2 部

收稿日期：2010-11-22；修回日期：2010-12-06
基金项目：国家“863”项目（2008AA7045007）
作者简介：程龙（1981—），男，甘肃人，博士，讲师，从事飞行器测试发控研究。

分。其中，卫星平台可又可分为结构与机构、热控制、电源、姿态与轨道控制、测控、数据管理、总体电路、返回等分系统[7]。依照各组成部分进行建模，形成多个仿真成员，再配合 CAN 总线仿真框架，构建实用、灵活、可扩展的星上仿真系统，结构如图 1。

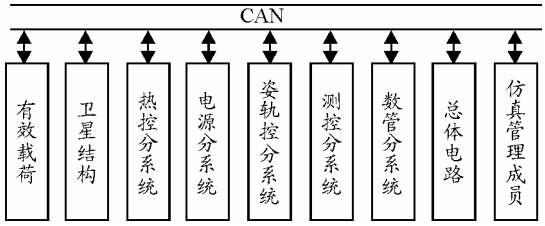

图 1　系统结构

在仿真系统中，除了对应实际系统的各仿真成员之外，还包含一个仿真控制成员，用于进行仿真运行的管理与控制。

2　基于仿真的 CAN 传输框架设计与实现

2.1　总体思路

在仿真系统集成的过程中，首先要解决各仿真成员之间的通信接口问题。依照各自的设计需求，各仿真成员的运行环境不一，有的在 PC 机上，有的则运行在嵌入式系统中。而且虽然在通信的最底层都满足 CAN 协议，但各自的 CAN 设备出于不同厂商，采用的 CAN 控制器会有所不同，驱动程序接口也各不相同。要实现仿真通信接口的规范，首先要在各成员驱动程序接口之上进行二次封装，形成统一接口，以此接口进行系统内部的信息传输，即采用统一的 CAN 传输框架，就可以实现跨硬件平台、跨操作系统的交互式仿真，如图 2。

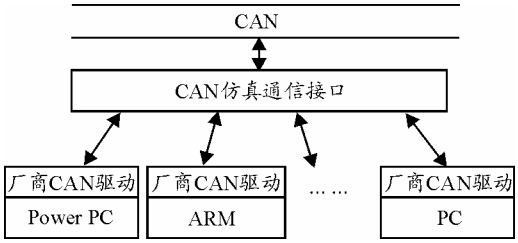

图 2　系统传输接口示意

图 2 中的 CAN 仿真通信接口，也就是一个基于仿真的 CAN 传输框架，具体由 CCANMemmber 类进行实现，并将其封装为动态链接库的形式，以方便开发人员利用。该类适合于普通仿真成员，也适合于仿真管理成员，只是在仿真过程中所调用的函数有所区别，其接口如下：

```
class _declspec(dllexport) CCANMember
{
private:
BYTE CanConfig[11];// CAN 配置参数
public:
CCANMember(int n);//初始化仿真成员，输入为成员 ID
~CCANMember(void);
int MemberID;//仿真成员 ID
DataBuffer RevBuffer;//数据缓冲区
int MemberNum;//加入系统的成员数（仿真管理成员）
void InitCAN();//初始化 CAN 总线
void GetData();//接收总线数据
void SendJoinIns();//发送“加入请求”（分系统成员）
void SendDataReq();//发送“数据请求”（分系统成员）
void SendStepReq();//发送“推进请求”（分系统成员）
void WaitPermit();//等待“推进许可”（分系统成员）
void WaitDataReq(); //等待“数据请求”（仿真管理成员）
void WaitStepReq(); //等待“推进请求”（仿真管理成员）
void SendDataPermit();
//发送“数据许可”（仿真管理成员）
void SendStepPermit();
//发送“推进许可”（仿真管理成员）
void SendData(int nParaIndex,float ParaDate);
//发送浮点型数据
void SendData(int nParaIndex,CString ParaDate);
//发送字符串
void SendData(int nParaIndex,double ParaDate);
//发送双精度数据
void SendData(int nParaIndex,int ParaDate);
//发送整型数据
};
```

为使框架简单明了，便于使用，在对其参数与函数进行设计时应尽量精简，满足仿真运行要求即可。针对不同平台与 CAN 控制器的底层驱动封装，集中在 InitCAN()、GetData()，和 SendData()3 个函数当中，其中，SendData()针对不同的传输数据类型进行重载（其输入参数为约定好的参数索引号和要发送的数据）。即针对不同的分系统，只需按照它们的 CAN 驱动接口对上述 3 个函数进行修改即可，这个工作由一名专门的 CAN 驱动接口封装人员来完成，而负责各分系统成员开发的人员则统一面向

CCANMember 类进行信息的传输编程，不用关注本分系统究竟用的是何种 CAN 控制器，更不用在底层驱动接口的调用上花费精力。

2.2 数据的收发设计与测试

在对 CAN 驱动接口进行封装时，要根据各厂商提供的驱动接口进行详细的设计与测试，确保封装的有效性和可靠性。举例来说，在仿真过程中，某成员可能需要不间断地进行多个数据帧的发送，该成员运行于 PC 机，采用 PCI 接口的 CAN 总线通讯卡，但该通讯卡原驱动接口中的发送函数 CAN_Trans()在连续使用时会产生延迟，其原因可能是计算机指令速度快于通讯卡响应能力，在计算机连续调用 CAN_Trans()函数时，在通讯卡一侧造成了阻塞，CAN 控制器接着便进行了延迟处理。这样便不能达到整个仿真系统所要求的发送速率。因此，在使用 CAN_Trans()后，必须调用查询总线状态函数 CAN_Readreg()，进行一次总线状态的查询，查询到总线已空闲可以放置数据时，才能再一次使用 CAN_Trans()函数，这样便能使卡的发送速率达到所设置的波特率。主要代码如下：

```
CAN_Trans(0,CanBuffer,50);//发送一帧数据
UCHAR CanReg[2];
CanReg[0]=2;
do{CAN_ReadReg(0, CanReg);}
while((CanReg[1]&12)!=12)//查询寄存器状态
```

另外，在发送时还要根据需要进行数据的校验措施，以避免出现传输错误。这些都是 CAN 驱动接口封装人员需要考虑的。

在数据的接收方面，各厂家一般会提供即时查询和缓存查询 2 种方法。

1) 即时查询方法

此方法采用一个函数对收到的每一帧进行查询接收，实际编程中对其循环调用即可。但在测试过程中发现，当总线速率较高（如 500 Kbps）时，采用此方法会有丢帧的现象发生。

2) 缓存查询方法

厂家提供的原接口一般称之为中断接收方式，因为在硬件上采用中断的方式将所有接收到的数据自动暂存到卡上一个若干字节大小的缓冲区内。在软件实现过程中，只需定时查询该缓冲区即可，此方法能保证以较快的速度接收数据，效果比较好。

笔者采用缓存查询方法实现数据的接收，为此对多点发送单点接收的性能进行了测试，主要检查在大速率的接收过程中，是否会出现数据的冲突、混淆或丢失现象，如图 3。

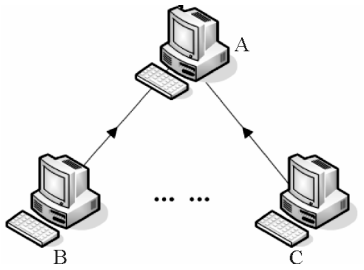

图 3 数据收发测试

图 3 中，A 机为 PC 机，WindowsXP 系统，采用某公司 PCI 接口的 KPCI-8110 型 CAN 总线通讯卡；B 机为嵌入式系统，采用某公司开发的 TE2440-II 型 ARM9 嵌入式工控板，CAN 控制器为 MT 公司的 MCP2510；C 机也为嵌入式系统，选用某公司的 EPC-8900M/I-L 型嵌入式工控机主板，该主板则采用基于 SJA1000 的 CAN 控制器驱动接口结构。

总线波特率设为 1 000 Kbps，B 机、C 机同时向 A 机发送 250 帧数据，A 机采用缓存查询方法进行接收，没有发生数据混淆或丢失的情况，如图 4。

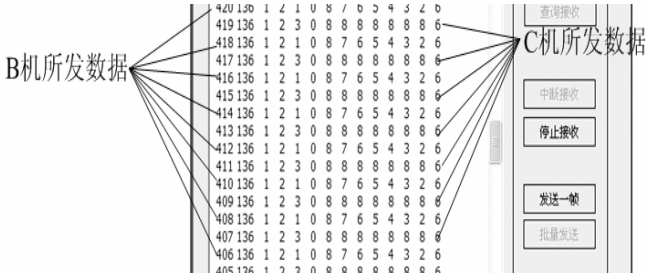

图 4 数据收发测试结果

3 仿真流程

在设计并完成 CAN 传输接口框架的同时，还需要与之结合制定出合理有效的仿真流程框架，来作为各仿真成员相互交互、协调运行的基础。系统仿真过程状态如图 5。

仿真过程可分为以下 3 步：

1) 仿真管理成员创建仿真系统，并不断查询总线数据，接收各分系统成员的“加入请求”，同时，各分系统成员发出“加入请求”之后便各自等待仿真管理成员发来“推进许可”，当所有分系统成员都加入系统后，仿真管理成员便发出“推进许可”。

2) 仿真管理成员发出“推进许可”后，便不断查询各分系统成员的“数据请求”，同时，各分系统成员进行自身的仿真运算，结束后发出“数据请求”，并等待“数据许可”，当仿真管理成员收到所有成员发出的“数据请求”后，便发出“数据许可”。

3) 仿真管理成员发出“数据许可”后，又不断

查询各分系统成员的“推进请求”。这时，各分系统成员发送数据的触发条件有所不同，第一个分系统成员收到仿真管理成员发来的“数据许可”后便将自身本仿真周期的运算结果发送到总线上，发送完毕后发送“推进请求”并等待“推进许可”，由于在等待推进许可函数 WaitPermit()中包含有获取数据函数 GetData()，因此，在此期间如果其它成员向总线上发布数据，第一个分系统成员也会接收到。第二个分系统成员收到第一个分系统成员的“推进请求”消息后再向总线上发送数据，发送完毕后发送“推进请求”并等待“推进许可”，依此类推。除了第一个成员之外，其他成员都是以上一成员的“推进请求”作为发送数据的触发条件。这样就保证了各分系统成员有序的依次向总线发送数据，避免了冲突，而且也确保在某个成员发送数据的同时，其他的所有成员都在查询总线数据，以免丢失数据。

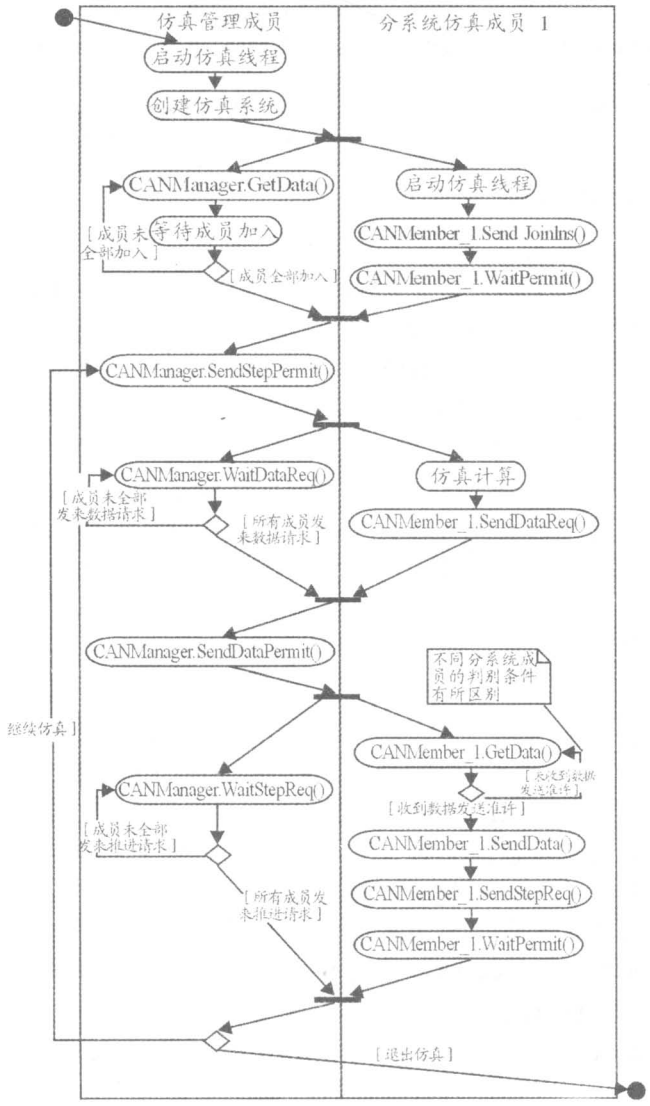

图 5　系统仿真过程

可见，在仿真过程中，仿真管理成员扮演着非常重要的角色，它协调并控制着整个仿真系统的信息传递顺序与握手关系，主要表现在“等待数据请求—发送数据许可”和“等待推进请求—发送推进许可”2 个关键节点。仿真管理成员在这两个节点进行控制，确保所有成员均完成上一步的工作后才能进入下一步。至于 CAN 总线数据的传输方式可以按照各成员之间交互数据的需求关系采用滤波方式，也可采用广播方式。

需要指出的是，各分系统成员的程序仿真框架是类似的，它只是保证仿真运行的一个环境，由一名专门的框架设计与实现人员进行维护即可。这样，各分系统成员的开发人员就能够把大部分精力放在成员的核心功能—仿真计算的开发上，提高了开发效率和系统的可维护性。

4　结束语

目前，整个星上仿真系统已经完成，且运行良好，在仿真管理、信息传输速度、可靠性等方面均满足要求。经过实践证明，该仿真框架传输效率高、运行稳定、适应性好，即使是对 CAN 总线不熟悉的开发人员也能很快地接受、使用，提高了仿真系统的开发效率、可扩展性和可靠性。下一步，将考虑对已有仿真框架进行扩展，使之能将实物仿真成员集成到系统中。

参考文献：

[1] 肖龙龙, 程谋森, 张为华. 运载器控制系统CAN总线数据传输仿真与评估研究[J]. 弹箭与制导学报, 2009, 29(2): 256-260.

[2] 刘淑芬, 孙昕. CAN 总线在卫星中的应用技术研究[J]. 航天控制, 2004, 22(6): 79-83.

[3] Gianluca Casarosa, Michele Apuzzo, Luca Fanucci, Bruno Sarti. Characterization of the EMC Performances of the CAN Bus in a Typical System Bus Architecture for Small Satellites[C]. 9th EUROMICRO Conference on Digital System Design (DSD'06), 2006: 338-345.

[4] Ferrer Albert, Parkes Steve. Unified communication infrastructure for small satellites[C]. 60th International Astronautical Congress 2009, 2009: 3776-3780.

[5] Yan Meizhi. Research on built-in testing methodology of small satellite[C]. 60th International Astronautical Congress 2009, 2009: 6991-6994.

[6] 王继河, 王峰, 兰盛昌, 等. 基于微型核的双星编队实时仿真系统[J]. 系统仿真学报, 2008, 20(2): 328-331.

[7] 徐福祥. 卫星工程概论[M]. 北京: 中国宇航出版社, 2003: 9.