基于 RTLinux 系统的制导火箭弹实时仿真计算机系统*

章 校[1],董国才[2]

(1 国防科学技术大学,长沙 410073;2 中国兵器工业第 203 研究所,西安 710065)

摘 要:选用 RTLinux 系统配置硬件接口卡,开发相应的驱动程序来实现制导火箭弹模型的实时解算和实时数据交换,完成制导火箭半实物仿真试验,测试结果表明本系统能满足制导火箭半实物仿真试验对仿真计算机的要求。

关键词:制导火箭弹;半实物仿真;实时系统;驱动程序

中图分类号:V448.15 **文献标志码**:A

Design of Real Time Simulate Computer System Based on RTLinux

ZHANG Xiao[1], DONG Guocai[2]

(1 National University of Defense Technology, Changsha 410073, China;
2 No. 203 Research Institute of China Ordnance Indastries, Xi'an 710065, China)

Abstract: Selecting RTLinux system and configuring hardware interface card, driver program was empoldered to achieve the model of guided rocket real time computation and data exchanged to complete the guided rocket hardware in the loop simulating. The test result indicated the system can complete the requirement of the simulate computer of the guided rocket hardware in the loop experiment.

Keywords: guided rocket; hardware in the loop simulating; real time system; driver program

0 引言

文中介绍了基于 RTLinux 系统的实时仿真计算机系统的组成、技术要求、测试结果和相关硬件驱动程序的开发,设计完成了一个能完成制导火箭弹半实物仿真试验的实时仿真软件。该软件能完成该项目部分制导部件参与情形下的半实物仿真试验。

1 系统构成

本系统的硬件组成如下:Celoren800 CPU、256MB 内存、CF 硬盘卡、主板、电源板 IO 卡、4 串口卡和网卡。

本系统的软件组成如下:Linux 操作系统、RTLinux 实时内核、C/C++开发工具、通讯板卡(IO、串口、网卡)驱动程序、应用程序等。

2 系统的技术指标和测试结果

为了使所搭建的实时环境能满足实时仿真要求,对 RTLinux 系统提出的技术指标如下:

1)定时器分辨率不大于 1μs,定时器周期不大于 1ms;

2)定时误差:每帧最大误差小于 30μs,累积误差小于 30μs,误差大于 10μs 的帧数小于 1%;

3)定时器具有自动校正功能,即每两帧进行自动补齐。中断响应时间无负载时小于 15μs,有负载(如磁盘读写)时小于 25μs,上下文切换时间小于 2μs。

使用高精度示波器结合系统自带的时间测试函数,测出系统的各项指标如下:

1)1ms 定时器与高精度示波器相比较的误差为 0.1μs;

* 收稿日期:2008-10-15
作者简介:章校(1981-),男,江苏沭阳人,硕士研究生,研究方向:系统仿真。

2)连续运行 1000s,最大误差 4.3μs,累积误差为 0.6μs;

3)最大中断响应时间无负载时为 2.9μs,有负载时为 3.5μs,上下文切换最长时间为 1.3μs。

3 系统软件的设计

实时仿真计算机系统的软件部分主要由硬件驱动程序、实时定时器软件、仿真软件、非实时参数设置软件组成。硬件驱动程序主要完成实时仿真计算机系统与参试部件之间的实时数据交换;实时定时器软件主要确保仿真软件在规定的时间内能完成数学模型运算和数据交换工作;仿真软件主要完成仿真模型的运算;非实时参数设置软件主要完成仿真前的仿真初始参数设置工作。实时仿真计算机系统的软件程序流程图如图 1 所示。

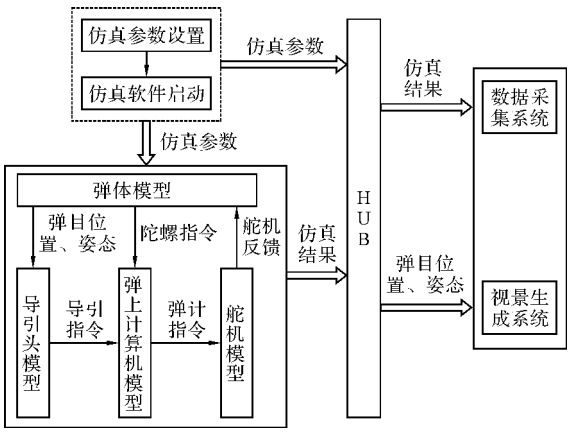

图 1 实时仿真计算机系统的软件程序流程图

3.1 硬件驱动程序

3.1.1 IO 驱动程序

IO 驱动程序由 3 个子函数组成,IO_Init()函数实现板卡初始化,IO_Write()函数完成数据发送,IO_Read()函数完成数据读取。各子程序源代码如下:

```
#define BASEADDRESS_IO 0x200
voidIO_Init()
{
    rtl_outb(0x00, BASEADDRESS_IO);
    rtl_outb(0x00, BASEADDRESS_IO + 1);
    rtl_outb(0x0,  BASEADDRESS_IO + 2);
    rtl_outb(0x80, BASEADDRESS_IO + 3);
}
void IO_Write(unsigned char value)
{
    rtl_outb(value, BASEADDRESS_IO + 2);
}
unsigned char IO_Read()
{
    return rtl_inb(BASEADDRESS_IO + 2);
}
```

3.1.2 RS232 驱动程序

RS232 驱动程序由 3 个子函数组成,RS232_Init()函数实现板卡初始化,RS232_Write()函数完成数据发送,RS232_Read()函数完成数据读取。各子程序源代码如下:

```
#define COM        0x3fd
#define B_VALUE   3

int RS232_Init()
{
    unsigned char out=1;
    int count=0;
    while(out==1 &&( count++ < 1000))
    {
        out = rtl_inb(COM);
        out = out&1;
    }
    if(count == 1000)
    {
        printf("RS232 Init Failure!\n");
        return 0;
    }
    printf("RS232 Init Succeed!\n");
    rtl_outb(0x80,    COM-0x02); //              b
    rtl_outb(0x00,    COM-0x04); //              9
    rtl_outb(B_VALUE, COM-0x05); //
                                  she zhi bo te lv   8
    rtl_outb(0x03,    COM-0x02); //              b
    rtl_outb(0x00,    COM-0x04); //
                                  ping bi zhong duan  9
    return 1;
}
void RS232_Write(unsigned char Value)
{
    rtl_outb(Value,  COM-0x05);
}
unsigned char RS232_Read( )
```

```
{
    unsigned char out=1;
    int count=0;
    while(out==1 &&( count++ < 1000))
    {
        out = rtl_inb(COM);
        out = out&1;
    }
    if(count == 1000)
    {
        printf(" Read RS232 Data Failure ! \n");
        return 0;
    }
    return rtl_inb(COM-0x05);
}
```

3.1.3 UDP 通讯子程序

UDP 通讯主要负责完成与外部视景计算机的数据通讯,周期大约为 5ms,不需要实时发送,主要在非实时代码部分实现。

3.2 实时仿真环境搭建

实时仿真环境包括实时部分和非实时部分,实时部分主要完成仿真模型计算和实时数据交换,非实时部分主要完成系统各部分的初始化、参数初始化、UDP 通讯、实时线程启动等工作。实时定时器的功能包括定时周期设置、每两帧自动校准、超时帧数统计、数学模型结算和实时数据通讯等工作。实时线程源代码如下:

```
#define PERIOD (1000 * 1000)
void * thread_code(void * t)
{
    struct timespec next, next1, next2;
    int EndState = 1; /* Simulate End */
    int FrameTimeOutCount = 0; /* FrameTime Out Count */
    unsigned char RS232Data=0, IOData=0;
    int FrameNum = 0;
    rtl_clock_gettime( RTL_CLOCK_REALTIME, &next );
    while (EndState ) {
        timespec_add_ns( &next, PERIOD );
        rtl_clock_gettime( RTL_CLOCK_REALTIME, &next1 );
        while(数据通讯完)
        {
            RS232_Write(RS232Data);
            IO_Write(IOData);
            RS232Data = RS232_Read();
            IOData    = IO_Read();
        }
        Simulate_model();// 完成模型解算
        rtl_clock_gettime( RTL_CLOCK_REALTIME, &next2 );
        timespec_sub( &next2, &next1 );
        if( next2.tv_nsec> PERIOD ) // 超时帧数统计
        {
            FrameTimeOutCount++;
            printf( " FrameTime Out ! FrameTime = %.3f us\n", ext2.tv_nsec/1000.0f);
            if(FrameTimeOutCount>100 )  //超时帧数超过 100,仿真结束
            {
                EndState = 0;
            }
        }
        rtl_clock_nanosleep( RTL_CLOCK_REALTIME,
                RTL_TIMER_ADVANCE | RTL_TIMER_ABSTIME,
                &next, NULL);
        FrameNum++; // 运行帧数统计
        if( 模型解算完成 )
    {
            EndState = 0;
        }
    }
    return NULL;
}
```

3.3 实时仿真系统实时性测试

为了测试本系统的实时性,将某火箭弹仿真模型移植进去,与外部交互的数据为一个 Unsigned char 类型的 IO 数据、十个 Unsigned char 类型的 RS232 数据、30 个 float 类型的 UDP 数据,定时器周期为 1ms,采用高精度示波器结合系统自带的时间测试函数来测试模型解算时间和最大帧时间,具体方法如下:

1) 定时器通过 IO 连续发出 1s 的周期为 1ms 的方波信号,高精度示波器采集这些方波信号,同时通过系统自带的时间测试函数记录方波

信号实际输出的周期，比较示波器采集的方波周期与时间测试函数记录方波周期的误差，以确定时间测试函数所记录的时间结果是否可信；

2）定时器启动后首先记录当前时间，与上一帧的时间差就是定时器的运行周期。测试结果见表 1。

表 1 示波器与时间函数测试结果 μs

次数	测试方法		次数	测试方法	
	示波器	时间函数		示波器	时间函数
1	1000.22	1000.41	5	999.86	999.94
2	1000.20	1000.38	6	1000.32	1000.51
3	1000.12	1000.25	7	999.98	1000.03
4	999.90	1000.04	8	1000.14	1000.20

运行仿真软件，使用时间函数记录系统运行的最大周期为 1000.43μs，最小周期为 999.89μs，结果可以满足火箭弹制导控制系统半实物仿真对实时性的要求。

4 结论

使用基于 RTLinux 所搭建的实时仿真环境，将某火箭弹仿真模型移植进去，经测试完全满足实时性要求。由此可见，基于 RTLinux 所搭建的实时仿真环境，可以在一定程度上满足火箭弹制导控制系统半实物仿真对实时性的需求，具有一定的推广价值。提供的定时器完全可以满足本项目的半实物仿真的实时性要求。所提供的硬件接口可以满足部分制导部件参试下的半实物仿真要求。为了能完成复杂的半实物仿真，还需要开发相应的硬件驱动程序，优化仿真软件，开发各种算法模块，使本系统在实时性、接口通讯、模型开发等方面更加强大、优化、简单。

参考文献：

[1] 郭春生 朱兆达. 硬实时操作系统-RTLinux[J]. 电子技术应用，2002，28(4)：17－19.

[2] Matt Welsh，Matthias Kalle Delheimer，Lar Kaufman. Linxu 权威指南[M]. 洪峰，译. 三版. 北京：中国电力出版社，2000.

[3] 龚建伟，熊光明. Visual C＋＋/Turbo C 串口通信编程实践[M]. 北京：电子工业出版社，2004.

（上接第 212 页）

验结束时产品可靠性估计，加深对产品可靠性增长规律的认识。由表 3 可见，形状参数 β 各个阶段的均值相差并不大，满足失效机理不变的假设。由图 3 可见，从统计意义上看，Weibull 过程的尺度参数 $\alpha_1 > \alpha_2 > \alpha_3 > \alpha_4$，表明随试验阶段的推进，失效率不断降低，生动地描述了产品的可靠性增长过程。

5 结论

针对小样本情况下产品延缓修正的研制过程，利用非齐次泊松过程建立多阶段 Bayes 可靠性增长模型；采用 Gamma-Beta 分布作为有先验信息情况下的先验分布，符合可靠性增长的特点；实例表明该方法能够充分利用产品研制所有阶段的专家经验和试验数据，适合小子样场合的可靠性增长评估。

参考文献：

[1] 梅文华，陈家鼎. 可靠性增长试验[M]. 北京：国防工业出版社，2003.

[2] Guida M，Pulcini G. Bayesian reliability assessment of repairable systems during multi-stage development programs[J]. IIE Transactions. 2005，37(11)：1071－1081.

[3] 张金槐，唐雪梅. Bayes 方法[J]. 修订版. 长沙：国防科技大学出版社，1993.

[4] Guida M，Pulcini G. Bayesian reliability-growth modeling of repairable mechanical systems[C]// Proceedings of the 3rd International Conference on Mathematical Methods in Reliability-MMR2002，Trondheim(Norway)，2002：263－266.