# 基于 WinCE 的发动机试验实时控制软件设计

许广柱，吴锦凤
（西安航天动力试验技术研究所，陕西 西安 710100）

**摘 要**：通过对 WinCE 操作系统线程、进程和定时器的分析，采用多媒体定时器和外围硬件配合的方式，结合 XXX 液体火箭发动机试车台增压/闭环控制系统实时时序控制的设计和应用，系统论述了在 WinCE 操作系统下，实时控制的实现原理和设计方法，并阐述了基于多媒体定时器实时控制软件设计框架的应用前景。

**关键词**：时间片；地址空间；控制逻辑；回调函数
**中图分类号**：V416-34 **文献标识码**：A **文章编号**：1672-9374 (2011) 05-0074-05

# Design of WinCE-based real-time control software for rocket engine test

XU Guang-zhu, WU Jin-feng
(Xi'an Aerospace Propulsion Test Technique Institute, Xi'an 710100, China)

**Abstract**: The concept and the design method of the real-time control under the WinCE operation system is described in this paper by the analysis of the WinCE operation system and its timer. The multimedia timer and the peripheral hardwares are employed in the design. The design and application of the real-time time sequence control in the pressurization and the close-cycle control system of the liquid rocket engine are introduced. The application prospect of the real-time control software design framework based on the multimedia timer is elaborated.

**Keywords**: time slice; address space; control logic; callback function

## 0 引言

工业控制在很多方面都有实时性的要求，如今，绝大部分工业控制软件都是基于 Windows 操作系统设计编制的，如何在多任务的 Windows 系统上实现实时控制是很多工业程序员头疼的问题。多年来，有的控制系统以编程极不便利的 DOS 操作系统为平台开发设计控制方案，有的购买价格昂贵的实时操作系统设计开发，不但需要人员再培训，且设计开发周期长。新建的 XXX 试车台控制系统，使用了成本相对低廉、实时性较好的嵌入式 WinCE5 操作系统进行设计。由于 WinCE 系统仍然是以 Windows 为核心，即多任务

**收稿日期**：2010-12-02；**修回日期**：2011-03-06
**作者简介**：许广柱（1975—），男，工程师，研究领域为液体火箭发动机试验系统测控技术

操作系统，实时性方面同样存在问题。因此，本文就此操作系统下的实时控制提供一套系统解决方案并论述软件框架的扩展和应用。

## 1 系统分析概述

### 1.1 WinCE 操作系统的时间分配

WinCE 操作系统是多任务系统，就是在“同一时间段内可以同时执行”多个应用软件，完成多个任务。其实，所谓“同一时间段内同时执行”是相对于使用者感觉而言的。事实上，在同一个时刻 WinCE 只能够完成一个指令，使用者感觉是“同时”，是因为 WinCE 给每个线程分配毫秒级的时间片，并按每线程优先级来依次执行操作，由于时间片很短，人眼睛是不能分辨执行顺序的，所以造成了“同时执行”的错觉，正因为这种时间片分配来执行不同任务的特性，导致了 WinCE 操作系统不能完成完全性质的实时性控制。

### 1.2 WinCE 系统的进程与线程

进程：一个进程通常定义为程序的一个实例。在 32 位 Windows 中，进程占据 4 GB 的虚拟地址空间。进程是没有活力的，就是说，一个 32 位 Windows 进程并不执行什么指令，它只占据着 4 GB 的地址空间，此空间中有应用程序 EXE 等 PE 文件的代码和数据。

线程：如上所述，进程只是一个静态的概念；为了让进程完成一些工作，进程必须至少占有一个线程，所以线程是描述进程内的执行。正是线程负责执行包含在进程的地址空间中的代码，如果没有线程执行进程地址空间中的代码，进程也就没有继续存在的理由，系统将自动清除进程及其地址空间。为了运行所有这些线程，操作系统为每个独立线程安排一些 CPU 时间，以轮转方式向线程提供时间片，所以 Windows 系统时间片的分配是以线程为基本单位的。创建一个 32 位 Windows 进程时，它的第一个线程称为主线程，由系统自动生成，然后可由这个主线程生成额外的线程，这些线程又可生成更多的线程。

### 1.3 WinCE 系统的线程优先级

如前所述，Windows 操作系统时间片分配是针对线程来说的，而线程执行的顺序是按照其优先级来编排的。对于 Windows CE 等嵌入式操作系统，优先级是以线程为单位，整个系统运行中的应用级线程缺省都是同等优先级。改变线程的优先级，或者说进行线程调度就可以改变 WinCE 系统时间片的分配方向以及线程自身执行的顺序。

改变优先级可以通过使用 SetThreadPriority () 函数来实现。Windows CE 系统是抢占式多任务系统，进程之间无优先级之分，只将线程分为 256 个优先级。0 优先级最高，255 最低，0 到 247 优先级对应于核心态线程。248 到 255 优先级对应于用户态线程，一般分配给普通应用程序线程使用。251 优先级 (THREAD_PRIORITY_NORMAL) 是正常优先级。255 优先级 (THREAD_ PRIORITY_IDLE) 为空闲优先级。249 优先级 (THREAD_ PRIORITY_HIGHEST) 是高优先级。

综上所述，如果可以在线程上争取到一定程度的优先级，也就是争取到了时间片分配和执行权的优先级，自然就可以达到相对实时性的控制效果。

### 1.4 WinCE 下的定时系统

32 位 Windows CE 系统下有以下 3 种定时系统。

1) 基于消息机制的定时器，通过 SetTimer () 函数设置，使用消息处理机制。由于其原理为消息，所以定时精度较低，一般为 55 ms 左右。

2) 多媒体定时器，通过 timeSetEvent () 函数设置，是内核级定时器，定时精度很高，可以达到 1 ms，使用回调函数形式执行任务代码。

3) CPU 计数器，通过 QueryPerformanceCounter () 和 QueryPerformanceFrequecy () 来获取计算机主频计数器值（从开机开始到调用时的时间间隔计数值）和 CPU 频率，可以做到微秒级的时间控制精度，但是没有相应的执行任务机制，只能作为高精度时间参考。

基于以上 3 个定时系统，在获取时间数值的

同时又要完成控制任务，1) 和 2) 是基本选择，至于时间精度方面就需要根据需求来选择使用。

### 1.5 多媒体定时器概述

一般情况下，应用系统无须参与系统线程调度工作，但对于比较复杂的高精度实时应用来说，有时需要应用自身参与本进程的内部线程的协调与调度 (或准调度) 工作，或直接使用高优先级线程执行控制任务以达到一定的实时性要求。

做到这一点，就需要依靠前文中提到的多媒体定时器，相应的接口函数以 Windows API 形式由动态链接库 coredll.dll 提供，C++头文件为 mmsystem.h。操作由 4 个接口函数和 1 个回调函数完成。4 个接口函数为：timeBeginPeriod ()，timeEndPeriod ()，timeSetEvent () 及 timeKillEvent ()。回调函数是完成任务的主体，timeSetEvent 第 3 个参数为此回调函数的地址。

通过函数 timeBeginPeriod () 设置最小的定时精度，可达到 1 ms 的精度，定时器回调函数执行时间间隔最小为 1 ms。该定时器不依赖于消息机制进行触发，当用户在程序中调用函数 timeSetEvent () 时，系统产生一个独立的 Win32 内核级线程，并将用户提供的用户级的时钟定时处理程序 (回调函数) 以回调的方式绑定在该线程中，而该线程又由硬件时钟中断 (很高的中断优先级) 的 ISR 来触发，因此利用 timeSetEvent () 编写的定时器程序，具有很强的中断优先权，远远高于其它的用户态线程，软件进程、线程不能打断其执行。综上所述，多媒体定时器表现出良好的实时性能。多媒体时钟的定时线程完全可以胜任直接执行主控制任务代码或者通过执行主控制逻辑操作内部线程的协调与调度工作。

### 1.6 基于 WINCE 系统的实时控制方法简述

综上几点，Windows CE 虽在实时性方面较其他 Windows 系统优秀，但仍属于多任务操作系统。相比实时操作系统，WinCE 下直接通过软件实现高精度时间控制是比较困难的。为了实现 10ms 级别的时间控制精度，可采用多媒体定时器结合外围硬件定时板的方式实现。利用 ISR 级中断的多媒体时钟轮询硬件定时器，并适时利用微秒级精度的 CPU 计数器进行时间补偿，以硬件定时时基为基准，完全可以达到 10 ms 级别的实时控制精度需求。

## 2 XXX 试车台增压/闭环控制系统的实现及应用扩展

### 2.1 增压/闭环控制系统的实现

XXX 试车台增压/闭环控制系统硬件采用研华定时器板 3780 和 IO 开关量板 3753。开关量板 3753 用于输入开关量的检测及输出开关量控制阀门动作；定时器板是实时性控制的硬件核心，其定时器的定时精度和稳定性影响整个控制系统的时间控制精度。

控制软件核心采用前文中所述的多媒体定时器，但因为多媒体定时器定时间隔精度较硬件定时器低，因而采用在多媒体定时器回调函数中获取 3780 硬件定时器数值，以获得满足系统时间精度要求的时间控制精度。

多媒体定时器的回调函数相当于一个优先级很高的定时线程，其自身的实时性保证了所有控制逻辑实现的实时性，配合高精度硬件定时器，可以做到高实时和高定时精度控制，对于试车控制系统来说完全满足要求。控制软件框架见图 1。

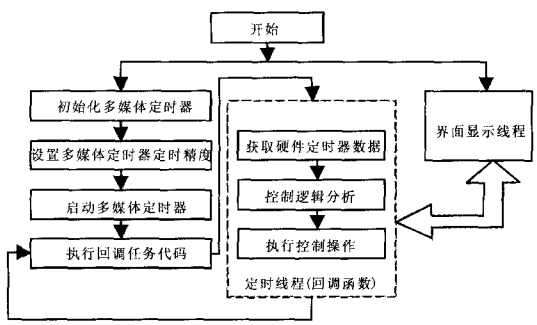


图 1 XXX 试车台实时控制系统软件框架图

Fig. 1 Block diagram of real-time control software for XXX test-stand

定时线程中取得硬件定时器计数值，计算并比较时间，如果到达指定的控制时间，则根据控制逻辑执行相应的操作。控制时间精度取决于硬件定时器和多媒体定时器的定时间隔精度，以及这两者的协调配合程度。增压/闭环控制系统多媒

体时钟定时间隔设定为 2 ms，实际测试整个控制系统千秒累计时间误差小于 20 ms，完全满足设计要求。

该软件框架设计过程中需要注意的是，回调函数中的所有工作必须在多媒体定时器的定时间隔内完成，否则会失去实时性并可能导致定时混乱。为保证多媒体定时器回调代码的实时性，有关界面显示工作的代码不能在回调函数中直接执行，需要设定一个专门用于显示的线程来完成此任务。

### 2.2 控制软件设计框架的应用扩展

图 1 所示的软件框架不仅仅应用在 XXX 试车台增压/闭环控制系统上，对其做相应的扩展就可以应用在众多的实时控制系统中，参见图 2。

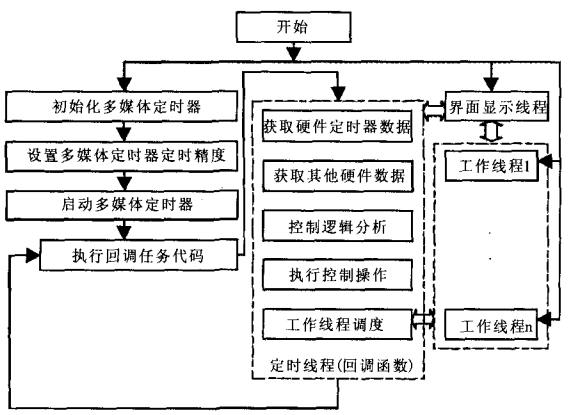


图 2 基于多媒体定时器的控制软件设计框架扩展图

Fig. 2 Design framework of control software based on multimedia timer

其中实时性要求很高的操作在回调函数本体中执行，其他辅助控制工作在工作线程中完成，并在多媒体定时器回调函数中通过分析任务需求，编制控制逻辑来完成对每个工作线程的调度，包括：线程优先级控制、工作线程逻辑顺序控制、工作线程之间的协调等任务。

线程的调度在前文中已有阐述。各个线程中途“挂起”SuspendThread () 或“终止”ExitThread () 或 TerminateThread () 的操作，正常情况下都是由自身完成的，并通过设置有关的“标志变量”值来“通知”定时线程，这样处理的特点是安全、简单、高效，而且也容易扩展新的线程。各个线程中途“恢复”ResumeThread () 或“启动”操作都是由定时线程完成的。线程间的同步可以通过设定临界区 Critical Section、互斥对象 Mutex、事件 Event 等来实现。

工业控制有些环节也要求很高时间精度的定时控制，关于这点可以使用 2 种方案来辅助完成：(1) 在回调函数中通过 API，Query Performance Counter () 和 Query Performance Frequecy () 来获取计算机主频计数器值和 CPU 频率，或者获取系统 CPU 时间戳，计算并比较时间，适时对多媒体定时做补偿修正，此方法可以很大程度上提高时间控制精度；(2) 制定外围高精度定时器板卡软件驱动，为控制软件提供 Windows 系统硬件中断级回调过程。软件编制控制逻辑在此回调中执行，并利用 CPU 时间戳进行进一步的时间控制。由于回调过程在硬件中断下驱动，时间控制精度很高，理论可达到微秒级，而且控制时间间隔非常稳定。需要注意，这种方式系统资源占用较大，在实际控制设计中必须考虑。

## 3 结束语

本文提供的设计框架，在多数工业控制应用场合都可以合理、有效的实现实时控制，虽然具体到每个应用环境还需要对架构进行适当调整，对工作逻辑控制进行合理配置，但基本思路都是适用的。理解并合理运用此架构，会为设计者带来很大的便利，其应用前景广大。该架构为今后试验控制系统的设计同样具有参考价值。

**参考文献：**

[1] BEVERIDGE Jim, WIEN Robert. Win32 多线程程序设计[M]. 武汉: 华中科技大学出版社, 2002.

[2] 傅曦, 齐宇. 嵌入式系统 Windows CE 开发技巧与实例[M].北京: 化学工业出版社, 2004.

[3] 孙鑫, 余安萍. VC++ 深入详解[M]. 北京: 电子工业出版社, 2006.

[4] RICHTER Jetfrey. Windows 核心编程[M]. 5 版. 北京: 清华大学出版社, 2008.

(编辑：马　杰)