**西安电子科技大学**

**操作系统课程设计**

**第1次实验报告**



实验题目： Alarm-Clock

学生姓名： 秦龙（15030120044）

班 级： 1503012

指导教师： 吴锦桥

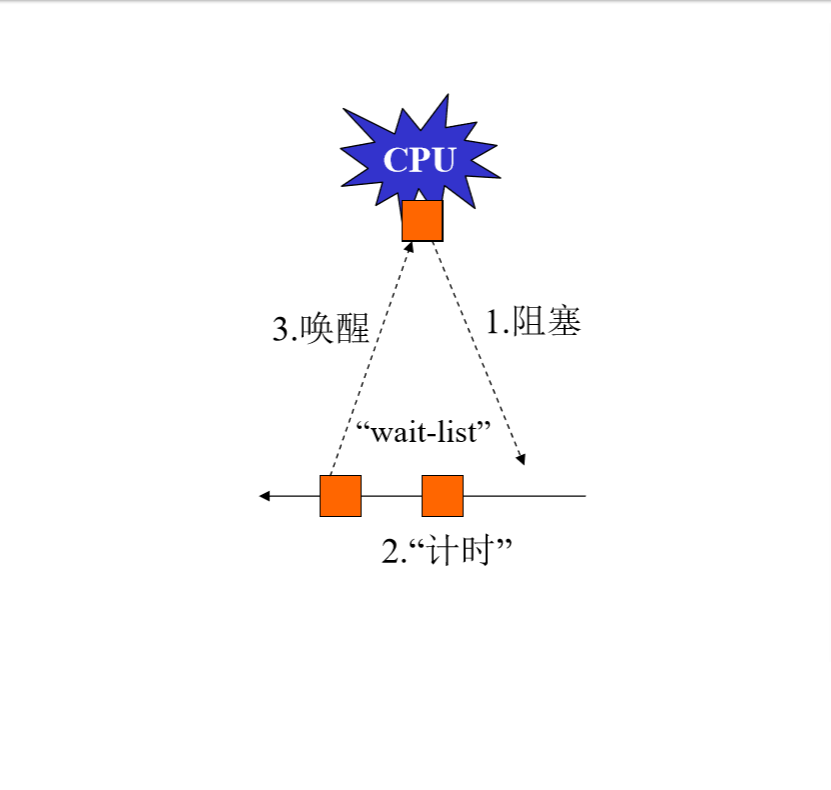
时 间： 2017年12月

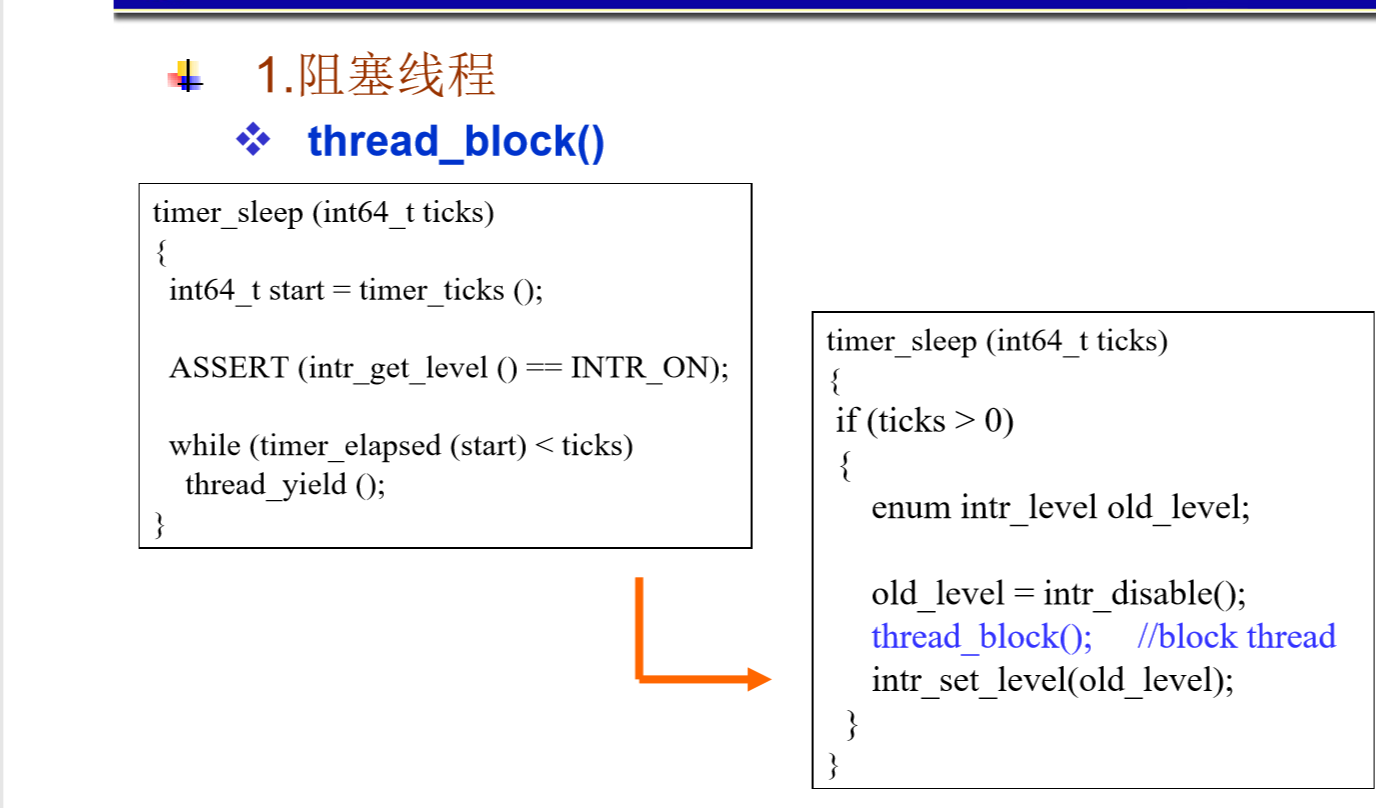
1. **实验名称 Alarm-Clock**
2. **实验要求：**

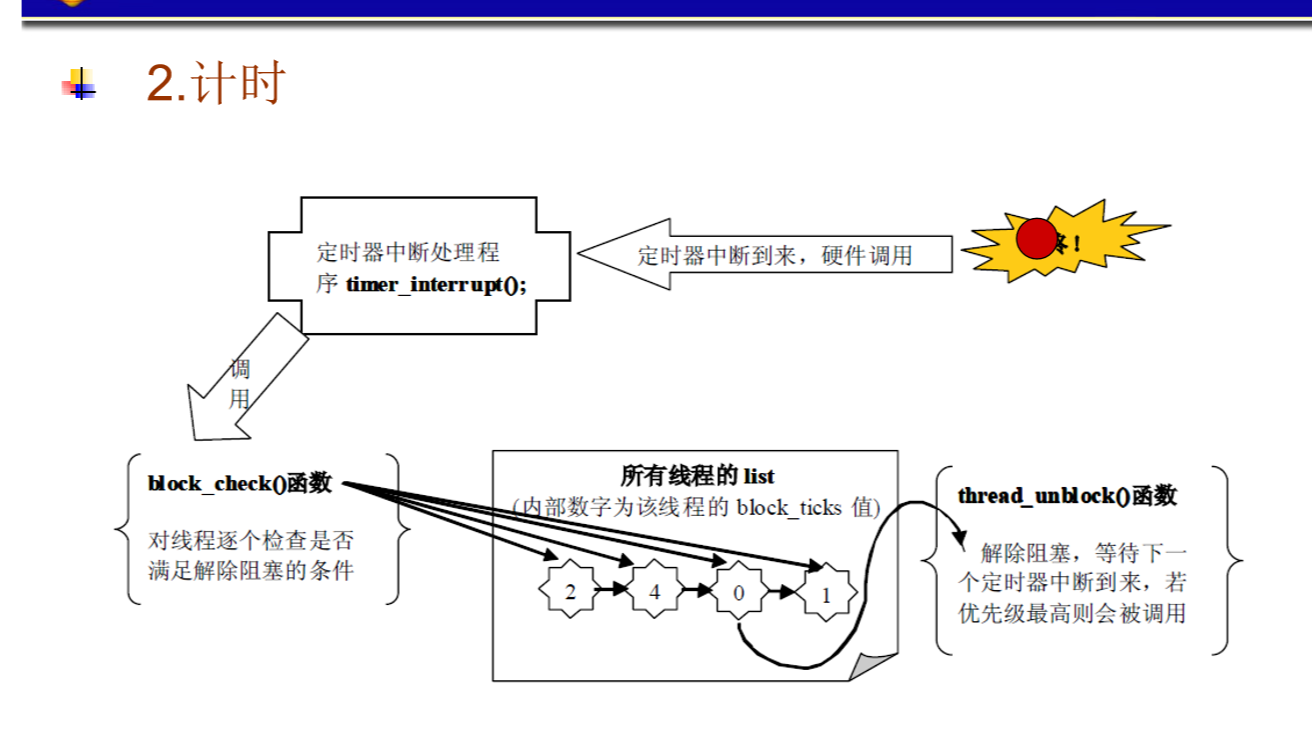
在pintos中有这样一个函数timer\_sleep（），该函数实现了让调用它的线程睡眠一段时间，然后唤醒。在pintos中已经实现该函数只是使用的是忙等待的方法。

**实验过程**

**实验方案图解**

****

****

****

一：实验要求

了解实验中使用到的相关源代码；

掌握源代码的数据结构等信息；

掌握pintos中线程的四种基本状态；

了解系统的驱动力 定时器中断以及中断处理过程；

二：分析：

对timer\_sleep（）分析可知在一个while循环中将会多次执行从而形成“忙等待”，要避免“忙等待”需要把timer\_sleep中的while循环替换掉，因此问题转化为如何替换函数中的while循环。

三：设计：

改变检测ticks定时器中断时间，此处改变原来用循环检测的方法，改用当每个定时器中断到来后再去检查一下wait\_list中的线程，看是否已经达到了唤醒的时机。如果时间到了，马上将其唤醒，反之不做任何的操作。

为了在每个定时器的中断到来时去检查wait\_list，我们可以改变timer\_sleep函数来实现，计入遍历链表的操作，检查每个线程的等待时间。

为了得到每个线程被阻塞了多长时间，我们需要对thread结构体进行一定的改变，加入一个记录当前线程需要等待时间的整形变量，之后每次中断的时候检查一次并且依次减一，当减为0时，意味着等待结束，就可以将其调入ready\_list中。

四：实现：

Timer\_sleep 函数原型：

Void timer\_sleep (int64\_t ticks) {

//ticks为等待的时间长度

int64\_t start = timer\_ticks ();

记录开始时的系统时间

ASSERT (intr\_get\_level () == INTR\_ON);

while(timer\_elapsed(start) <ticks)

//

如果

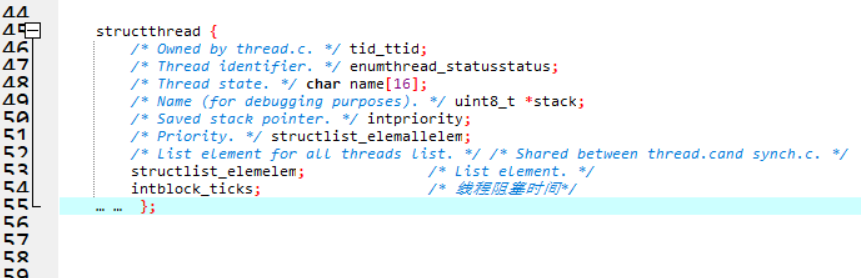
 elapse

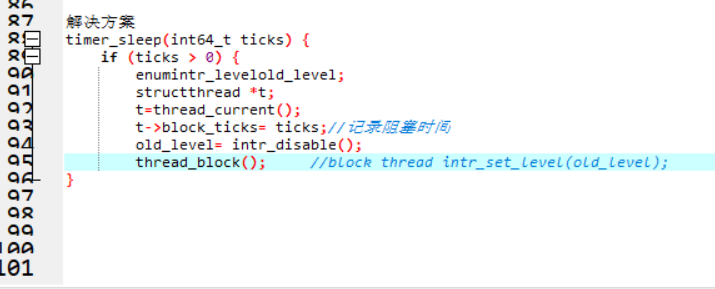
（流逝）的时间>=ticks时就返回。

thread\_yield ();

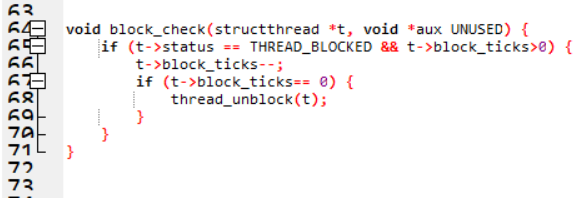
否则将持续占用 cpu

1改造thread结构体(thread.h)

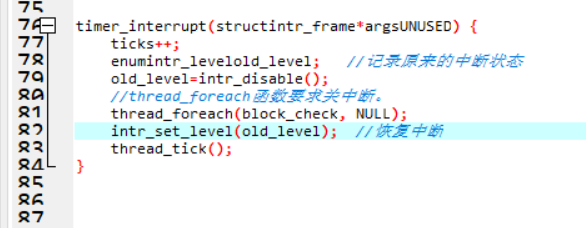




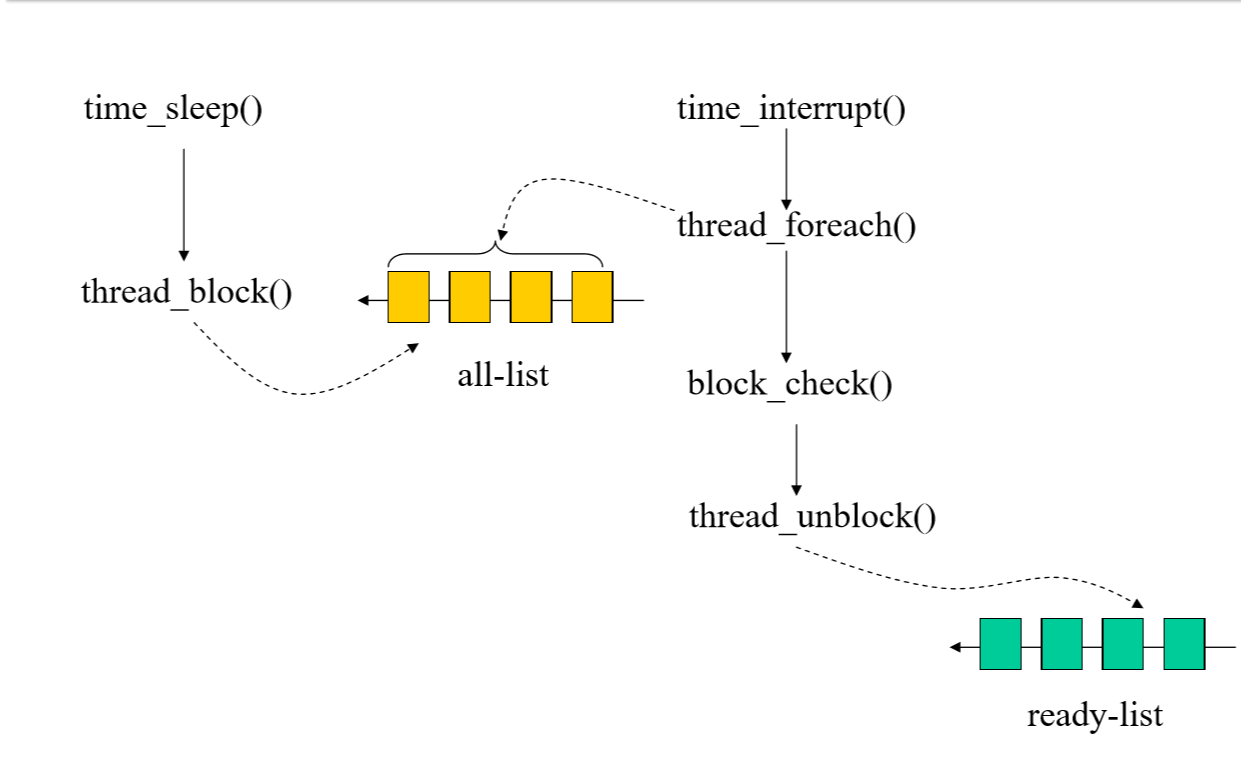
2定义block\_check()；



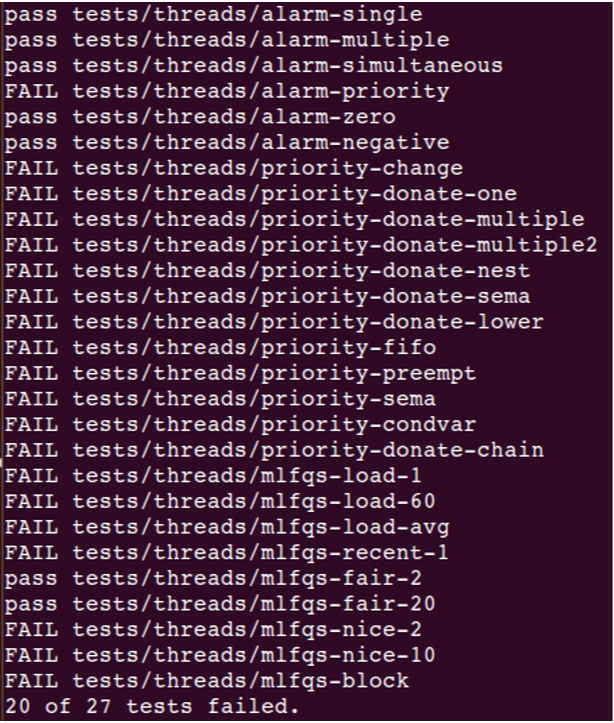
3调用block\_check()；



整个的函数调用关系如下：



**四、实验结果**



**五、心得体会**

通过本次试验接触了一些关于Linux编程的问题，知道了一些 Linux的基本命令与使用方法。也是第一次知道pintos系统，了解了最基本的一些操作。最重要的收获就是对线程操作上的使用，比如线程的创建，优先级的设置，线程之间的同步等等。