此程序循环运行步骤为：

While 1 {

select（3秒）

处理收到的设备状态（心跳、控制指令回复，仅记录进数据库）

根据socket链，查询数据库，发送指令，记录错误等。

每n秒，全数据库轮询

硬件心跳超时轮询处理（仅删除超时socket）

}

**心跳，回复中的设备状态处理**

仅更新设备状态，以及更新状态获取时间。

**设备真实状态 student\_no order dev\_state action**

0(关闭) -1 0

-1 1 set dev\_state=0；

!=-1 OPEN 0

!=-1 OPEN 1 set dev\_state=0；

!=-1 CLOSE 0

!=-1 CLOSE 1 set dev\_state=0；

如果close\_t=0,则close\_t=time();

1(开启) -1 0 set dev\_state=1；

-1 1

!=-1 OPEN 0 set dev\_state=1；

如果open\_t=0,则open\_t=time();

如果open\_t！=0，且close\_t！=0，中断时间为time()-close\_t,累加进break\_t中，set close\_t=0;

!=-1 OPEN 1

!=-1 CLOSE 0 set dev\_state=1；

!=-1 CLOSE 1

**注： 恢复设备至未占用模式，不涉及dev\_state操作。具体为：**

**set student\_no = -1**

**order = NONE**

**order\_recv\_t = 0**

**order\_send\_t = 0**

**open\_t = 0**

**close\_t = 0**

**remark=’’**

**break\_t = 0**

**所有计费公式为： close\_t – open\_t – break\_t**

**下同。**

**数据库轮询处理逻辑**

首先根据状态获取时间和当前时间，判读设备是否断线。（及时计费）

大于30秒未联系，则为断线，开始计费。但不更改设备真实状态。

此操作每5秒进行一次；

**student\_no order action**

！=-1 OPEN open\_t!=0,计费，关闭时间为当前时间-30秒，

恢复设备至未占用状态；

open\_t==0,恢复设备至未占用状态；

CLOSE 同上；

**student\_no order dev\_state action**

-1 0 如果remark不为空，设置其为空；

如果order\_recv\_t不为空，设置其为空；

如果order\_send\_t不为空，设置其为空；

如果open\_t不为空，设置其为空；

如果close\_t不为空，设置其为空；

如果break\_t不为空，设置其为空；

-1 1 order\_recv\_t为空，设置首次检测到的时间，

设置order\_recv\_t、order\_send\_t发送关闭指令；

now-order\_recv\_t<=15 同时 now-order\_send\_t>=5;发送关闭指令；

now- order\_recv\_t>=15同时remark！=’err’，则set remark=’err’,记录错误；

！=-1 OPEN 0 如open\_t为0, 由order\_recv\_t，order\_send\_t

发送OPEN指令；

如超时，则恢复设备至未占用模式；

open\_\_t不为0（开启后又中断），且close\_t=0,则设置close\_t=time();如果close\_t>0 且 time()-close\_t>30, 则产生计费（关闭时间为close\_t），恢复设备至未占用模式；

（不主动发送指令，通过硬件主动读状态，打开设备）

！=-1 OPEN 1 判断是否开启时间超过最大开启时间；

是，则set order=CLOSE, order\_recv\_t, order\_send\_t,发送CLOSE指令；

！=-1 CLOSE 0 产生计费；恢复设备至未占用模式；

！=-1 CLOSE 1 由order\_recv\_t，order\_send\_t发送CLOSE指令；

如超时，产生计费（order\_recv\_t为关闭时间）；

恢复设备至未占用模式；

**注：本文中，不对student\_no==0的用户计费；**