

OSLAB 2

吴泓宇 181250155

实验要求

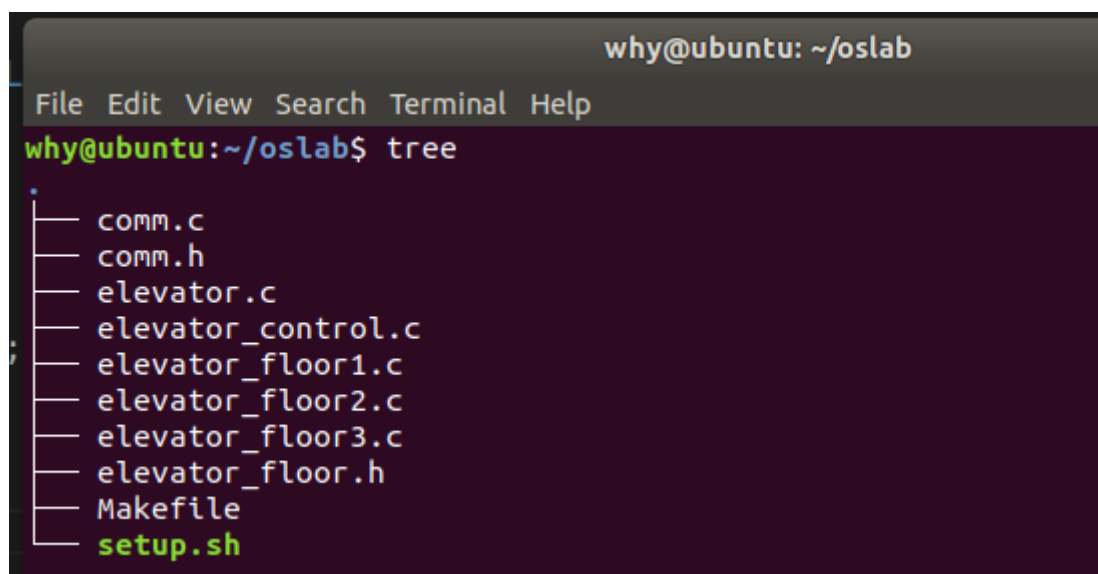
电梯模拟程序

实验任务：在 Linux 上编写程序，模拟一个三层楼房中的电梯运行。要求：

1. 该程序运行后由 5 个进程组成，分别模拟 1 楼、2 楼、3 楼的控制面板，电梯内的控制面板，以及电梯本身；
2. 每一楼层的控制面板进程由 2 个线程组成：线程 1 显示电梯的运行情况（所处楼层，运动方向等）；线程 2 接受用户的指令（按向上或者向下键）；
3. 电梯内的控制面板进程也由两个线程组成：线程 1 显示电梯的运行情况，线程 2 接受用户的按键（开门、关门、目的楼层）；
4. 电梯进程在上升、下降、停止、开门及关门这几个事件中循环；
5. 可以按照自己对电梯运行的理解合理调整上述需求；
6. 模拟程序应尽量多使用课程所学的进程间（线程间）通信、同步技术；
7. 图形界面或者控制台皆可。

架构设计

项目目录结构如下：



```
why@ubuntu: ~/oslab
File Edit View Search Terminal Help
why@ubuntu:~/oslab$ tree
.
├── comm.c
├── comm.h
├── elevator.c
├── elevator_control.c
├── elevator_floor1.c
├── elevator_floor2.c
├── elevator_floor3.c
├── elevator_floor.h
├── Makefile
└── setup.sh
```

各个模块的功能如下：

- `comm.c/h`: 封装共享内存的相关接口
- `elevator.c`: 电梯，只有显示状态作用

- `elevator_control.c`: 电梯内的控制台，一切指令的最终执行者，包含两个线程，一个显示电梯状态，一个接受开关门、目的楼层指令
- `elevator_floor1/2/3.c`: 各个楼层的控制台，包含两个线程，一个显示电梯状态，一个接受上下楼指令
- `elevator_floor.h`: 声明一些全局变量
- `Makefile`: makefile文件
- `setup.sh`: 程序入口

核心功能实现

多线程

采用了pthread线程库

进程间通信和同步

我采用了共享内存+互斥锁的方式，由`elevator_control`开辟一块共享内存空间，划分为六块区域，分别存放电梯当前楼层，电梯当前状态，1、2、3楼的信号，以及一把公共的mutex锁。

将mutex锁放在共享内存中的方式使得它由线程锁“进化”为一把进程锁。

共享内存和互斥锁的代码实现：

```
int main(int argc, char** argv)
{
    shmid = CreateShm(4096);
    shm_addr = shmat(shmid,0,0);
    *shm_addr = floor_num;
    *(shm_addr + 1) = dir;
    *(shm_addr + 2) = 0;*(shm_addr + 3) = 0;*(shm_addr + 4) = 0;

    // shared mutex
    p_mutex_shared = (pthread_mutex_t*) shm_addr + 5 ;
    pthread_mutexattr_t mutextattr;
    pthread_mutexattr_init(&mutextattr);
    pthread_mutexattr_setpshared(&mutextattr, PTHREAD_PROCESS_SHARED);
    pthread_mutex_init(p_mutex_shared, &mutextattr);
}
```

共享内存可以用来同步，可是如何让线程自动地去访问共享内存呢？在这里我借助了C库中的signal模块，用setitimer来定时发送一个时钟信号，这个信号会激活进程执行相应的函数。如下图为`elevator_control`的信号激活函数：

```

void alarm_func(int sn)
{
    if (p_mutex_shared == NULL)
    {
        printf("p_mutex_shared is NULL.\n");
        return ;
    }

    time_t tm = time(NULL);
    //printf("%s\t", ctime(&tm));
    if (pthread_mutex_trylock(p_mutex_shared) == 0)
    {
        //printf(GREEN "pthread_mutex_trylock success.\n" NONE);

        floor_num = *shm_addr;
        dir = *(shm_addr+1);
        for(int i = 2; i < 5; ++i) {
            if(*(shm_addr+i) == 1) {
                move(i-1);
                *shm_addr = floor_num;
                *(shm_addr + 1) = dir;
                *(shm_addr+i) = 0;
            }
        }
        //printf(RED "after mutex floor is: %d, state is :%s\n" NONE, floor_num, dirmap[dir]);

        pthread_mutex_unlock(p_mutex_shared);
    }
    else
    {
        printf(BLUE "pthread_mutex_trylock failed.\n" NONE);
    }
    //printf("#####\n");
}

```

这是elevator_control的子线程,它会不断检查全局的floor_num与上一次检查的floor_num的差异,若时钟信号同步后全局变量被修改,它就会将新的状态打印出来。其他进程中的状态打印也类似。

```

void* thread(void* arg)
{
    int cur_floor = -1, cur_dir = -1;
    pthread_detach(pthread_self());
    while(1){
        if(cur_floor!=floor_num || cur_dir != dir){
            printf("-----\n");
            printf("This is the sub thread of elevator controler\n");
            printf("The current floor is: %d\n", floor_num);
            printf("The current state of elevator is: %s\n", dirmap[dir]);
            printf("-----\n\n");
        }
        cur_floor = floor_num;
        cur_dir = dir;
    }

    pthread_exit(0);
}

```

运行截图

运行setup, 自动打开5个bash并分别运行5个进程

```
SLAB8 1 #include <pthread.h>
.vscode File Edit View Search Terminal Help
comm.c This is the main thread of elevator controller
comm.h -----
elevator This is the sub thread of elevator controller
elevator_control The current floor is: 1
elevator_control.c The current state of elevator is: STOP
elevator_floor.h -----
elevator_floor1 Please Input the floor you want to go, o to open the door, c to close the door, q
elevator_floor1.c to quit
elevator_floor2
elevator_floor3
elevator_floor3.c
elevator.c
Makefile
setup.sh

Terminal
File Edit View Search Terminal Help
This is the main thread of elevator
Fri Dec 4 06:37:12 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
Fri Dec 4 06:37:13 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
Fri Dec 4 06:37:14 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
Fri Dec 4 06:37:15 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
Fri Dec 4 06:37:16 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
Fri Dec 4 06:37:17 2020
pthread_mutex_trylock success.
=====
after mutex floor is: 1, state is: STOP
-----\n"
n");

Terminal
File Edit View Search Terminal Help
This is the main thread of elevator floor 1
This is the sub thread of elevator floor 1
The current floor is: 1
The current state of elevator is: STOP
-----
Fri Dec 4 06:37:12 2020
pthread_mutex_trylock success.
=====
Please Input the direction you want to go, u to go up, d to go down
n ,q to quit
Fri Dec 4 06:37:13 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:14 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:15 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:16 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:17 2020
pthread_mutex_trylock success.
=====

Terminal
File Edit View Search Terminal Help
This is the main thread of elevator floor 2
This is the sub thread of elevator floor 2
The current floor is: 1
The current state of elevator is: STOP
-----
Fri Dec 4 06:37:12 2020
pthread_mutex_trylock success.
=====
Please Input the direction you want to go, u to go up, d to go down
,q to quit
Fri Dec 4 06:37:13 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:14 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:15 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:16 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:17 2020
pthread_mutex_trylock success.
=====

Terminal
File Edit View Search Terminal Help
This is the sub thread of elevator floor 3
The current floor is: 1
The current state of elevator is: STOP
-----
Fri Dec 4 06:37:12 2020
pthread_mutex_trylock success.
=====
Please Input the direction you want to go, u to go up, d to go down
,q to quit
Fri Dec 4 06:37:13 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:14 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:15 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:16 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:37:17 2020
pthread_mutex_trylock success.
=====
```

可以看到，进程之间已经在不断获取互斥锁来同步电梯的状态

在controller上按键，比如当前楼层为1，去3楼：

```
File Edit View Search Terminal Help
3
go to floor 3
-----
This is the sub thread of elevator controller
The current floor is: 1
The current state of elevator is: CLOSING_DOOR
-----
This is the sub thread of elevator controller
The current floor is: 1
The current state of elevator is: UP
-----
This is the sub thread of elevator controller
The current floor is: 3
The current state of elevator is: OPENING_DOOR
-----
This is the sub thread of elevator controller
The current floor is: 3
The current state of elevator is: CLOSING_DOOR
-----

File Edit View Search Terminal Help
Fri Dec 4 06:38:08 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
Fri Dec 4 06:38:09 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
Fri Dec 4 06:38:10 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
Fri Dec 4 06:38:11 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
Fri Dec 4 06:38:12 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
Fri Dec 4 06:38:13 2020
pthread_mutex_trylock success.
after mutex floor is: 3, state is: STOP
=====
-----\n"n");

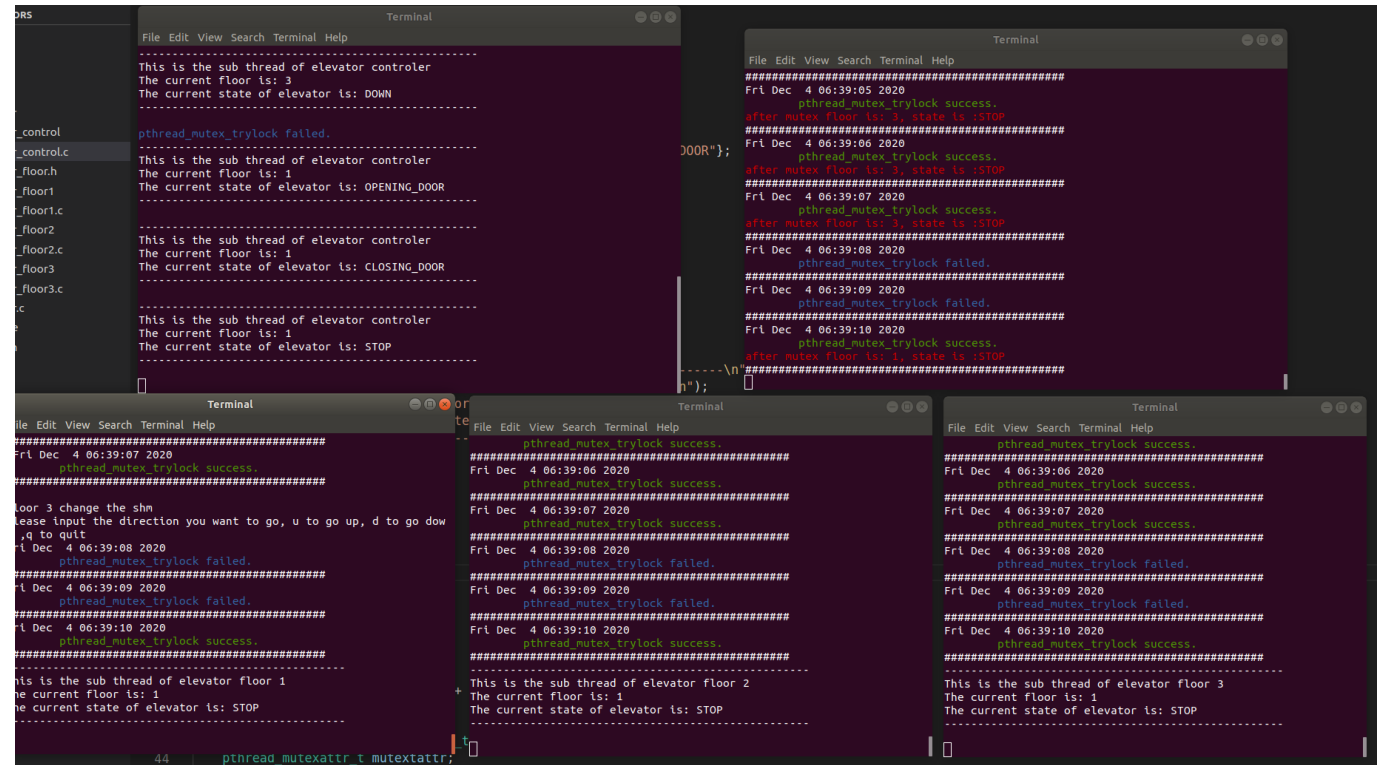
File Edit View Search Terminal Help
pthread_mutex_trylock success.
Fri Dec 4 06:38:06 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:07 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:08 2020
pthread_mutex_trylock success.
=====
This is the sub thread of elevator floor 1
The current floor is: 3
The current state of elevator is: STOP
-----
Fri Dec 4 06:38:09 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:38:10 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:38:11 2020
pthread_mutex_trylock success.
=====

File Edit View Search Terminal Help
pthread_mutex_trylock success.
Fri Dec 4 06:38:05 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:06 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:07 2020
pthread_mutex_trylock success.
=====
This is the sub thread of elevator floor 2
The current floor is: 3
The current state of elevator is: STOP
-----
Fri Dec 4 06:38:09 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:38:10 2020
pthread_mutex_trylock success.
=====

File Edit View Search Terminal Help
pthread_mutex_trylock success.
Fri Dec 4 06:38:06 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:07 2020
pthread_mutex_trylock success.
Fri Dec 4 06:38:08 2020
pthread_mutex_trylock success.
=====
This is the sub thread of elevator floor 3
The current floor is: 3
The current state of elevator is: STOP
-----
Fri Dec 4 06:38:09 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:38:10 2020
pthread_mutex_trylock success.
=====
Fri Dec 4 06:38:11 2020
pthread_mutex_trylock success.
=====
```

子线程开始模拟，并打印出电梯状态的改变，同时，其余各个进程也通过同步的方式捕捉到了状态改变，并作相应显示，在这里的一个问题是，control进程可以打印电梯完整的状态变化链，而其他进程只能捕捉到其中一个状态，这应该是时钟同步更新周期造成的。这里是1秒，但是，如果时间同步周期设置太短，又会直接导致没有东西打印出来...原因推测是主线程占用CPU太多导致子线程无法获得CPU资源？

若此时在一个楼层（如一楼）的控制台输入u(即up)，表示该层用户要上楼，则电梯则会从三楼下来：



这个操作涉及了其他进程对共享内存的读和写，要频繁访问互斥锁，因此可以发现其他各个进程都出现了互斥锁获取失败的情况，在读写完成后又恢复正常。说明互斥锁正常工作。

心得

本次实验中实现的电梯调度逻辑较为简单，主要是在设计共享内存和互斥锁上花了功夫，网上很多资料都说 pthread_mutex 无法用于跨进程的场合，但经过实践我还是实现了。这是我第一次实现多进程多线程的编程，踩了很多的坑，各种库函数的用法都是从零学起，但最后还是实现了比较完备的功能，成就感还是很大的。