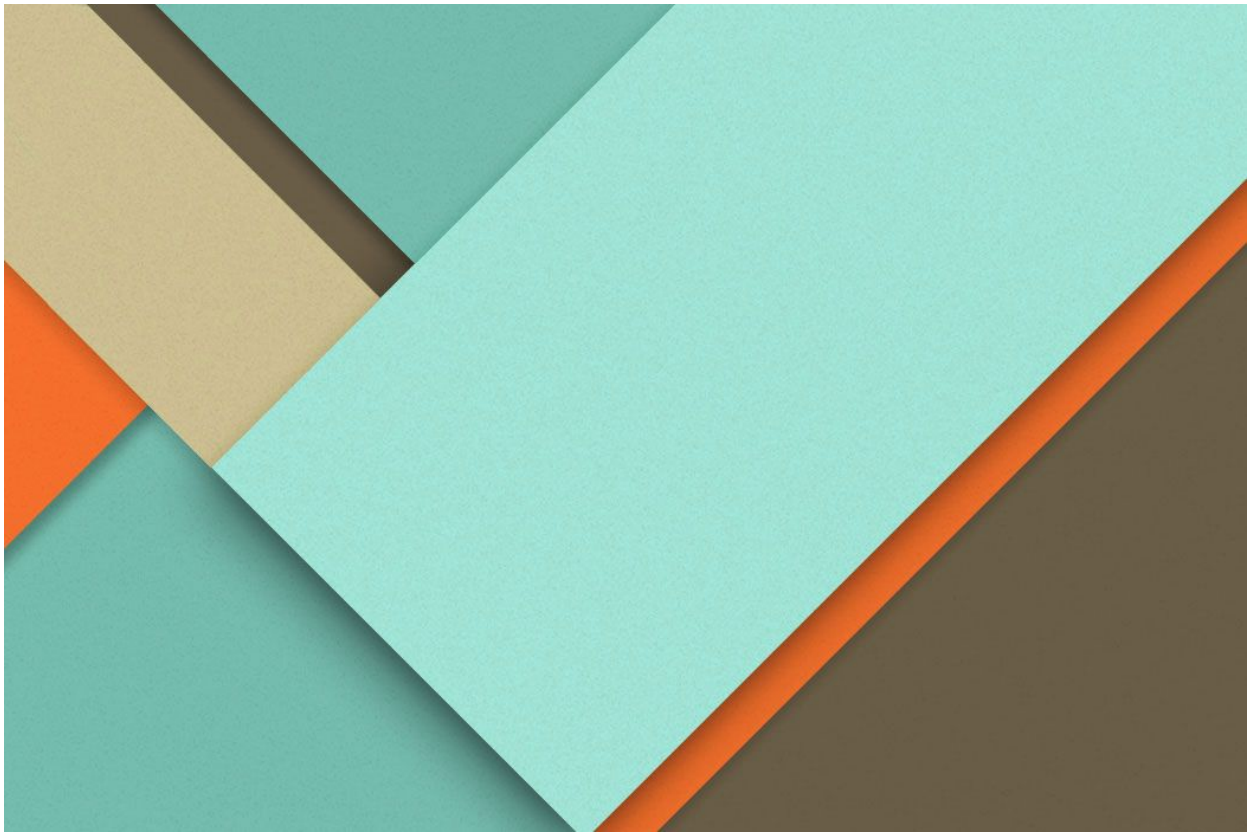

BATMAN

23.11.2019

Ahmed Mohamed Elzeny
ID 08

Code Structure

The code is based on 5 mutexes one is for the main crossing and four others for Each direction's queue any leaving bat has to signal 2 times, one is for the bat the current threat was in its right, and that condition variable is in the main mutex, the second signal is for the current thread's queue, we have also 4 boolean variables for each direction that are set when a bat is waiting in that direction, that helps detecting deadlocks, as a thread is launched in the beginning checking for when these four values are set, if so it signals the north thread to keep the bats flowing.

Important Functions

```
// is called by each bat, it has access to 4 mutexes and 4 cond var
// which when signaled releases the waiting thread to the cross mutex
void arrive(struct bat b){
    pthread_mutex_lock(&QMutex[b.dir]);
    if(WaitQ[b.dir]>0||WaitCross[b.dir]||WaitCrossing[b.dir]){
        WaitQ[b.dir]++;
 //       printf("%s waiten\n",getst(b.dir));
        pthread_cond_wait(&QCond[b.dir],&QMutex[b.dir]);
 //       printf("%s is signeled from queue\n",getst(b.dir) );
        WaitQ[b.dir]--;
    }


    printf("BAT %d from %s arrives at crossing\n", b.num, getst(b.dir));
    WaitCross[b.dir]=true;
    WaitCrossing[b.dir]=true;
    pthread_mutex_unlock(&QMutex[b.dir]);
    // code to check traffic in line, use counters, condition variables

}
```

```
// has access to the cross mutex, and 4 con variables, when signeled
// thier bat is allowed to cross
void cross(struct bat b){
    // code to check traffic from the right, use counters, condition variables etc
    pthread_mutex_lock(&CrossMutex);
    if(WaitCross[rightOf(b.dir)]||WaitCrossing[rightOf(b.dir)]){
        pthread_cond_wait(&CrossCond[b.dir],&CrossMutex);
//        printf("%s is signeled from cross\n",getst(b.dir) );
    }
    printf("BAT %d from %s crossing\n", b.num, getst(b.dir));
    WaitCross[b.dir]=false;
    sleep(1);
    // it takes one second for a BAT to cross

}

// signals the mentioned cond
void leave(struct bat b){
    printf("BAT %d from %s leaving crossing\n", b.num, getst(b.dir));
    WaitCrossing[b.dir]=false;
    pthread_cond_signal(&CrossCond[revRightOf(b.dir)]);
    pthread_cond_signal(&QCond[b.dir]);
    pthread_mutex_unlock(&CrossMutex);
    // code to check traffic, use counters, condition variables etc.
}
```

```
// checks for deadlocks every 1 second
// signals the north thread if detected
void* check(void* arg){
    bool done=false;
    while(1){
        sleep(1);

        if(WaitCross[0]&&WaitCross[1]&&WaitCross[2]&&WaitCross[3]&&!done){
            printf("DEADLOCK: BAT jam detected, signalling any waiting thread\n");
            pthread_cond_signal(&CrossCond[0]);

            done=true;
        }
        if(WaitCross[0]==false||WaitCross[1]==false||WaitCross[2]==false||WaitCross[3]==false){
            done=false;
        }
        if(DoneChecking)
            return 0;
    }
    return 0;
}
```

Problems faced me:

There were some cases where deadlock happened, even when I was checking for it, the problem was, i was signalling the north bat as soon as the 4 booleans “that indicates the bats waiting at the cross mutex” were all set, but when the north bat was last to arrive, the thread doesn’t make it to the line when it waits on the condition variable, so it signals nothing.

The solution was to apply some delay to the checking function so it let the threads settle

Before signaling, so i added a sleep call in the check() to make it periodic.

Sample runs

```
zeny@pop-os:~/Documents/projects/mutual execlusion$ ./main sewnseenww
BAT 0 from South arrives at crossing
BAT 0 from South crossing
BAT 2 from west arrives at crossing
BAT 3 from North arrives at crossing
BAT 1 from East arrives at crossing
BAT 0 from South leaving crossing
BAT 2 from west crossing
BAT 4 from South arrives at crossing
BAT 2 from west leaving crossing
BAT 8 from west arrives at crossing
DEADLOCK: BAT jam detected, signalling any waiting thread
0oddscsdccsdcdscs
BAT 3 from North crossing
BAT 3 from North leaving crossing
BAT 1 from East crossing
BAT 7 from North arrives at crossing
BAT 1 from East leaving crossing
BAT 4 from South crossing
BAT 5 from East arrives at crossing
BAT 4 from South leaving crossing
BAT 8 from west crossing
BAT 8 from west leaving crossing
BAT 9 from west arrives at crossing
BAT 7 from North crossing
BAT 7 from North leaving crossing
BAT 9 from west crossing
BAT 9 from west leaving crossing
BAT 5 from East crossing
BAT 5 from East leaving crossing
zeny@pop-os:~/Documents/projects/mutual execlusion$
```

Tracing an example

```
zeny@pop-os:~/Documents/projects/mutual execlusion$ ./main nsewnsewnsewwwweeesn
BAT 0 from North arrives at crossing
BAT 3 from west arrives at crossing
BAT 1 from South arrives at crossing
BAT 2 from East arrives at crossing
BAT 0 from North crossing
North waiten
South waiten
East waiten
North waiten
west waiten
South waiten
west waiten
west waiten
East waiten
west waiten
west waiten
East waiten
East waiten
East waiten
South waiten
North waiten
BAT 0 from North leaving crossing
North is signeled from queue
BAT 4 from North arrives at crossing
BAT 2 from East crossing
BAT 2 from East leaving crossing
East is signeled from queue
BAT 6 from East arrives at crossing
South is signeled from cross
BAT 1 from South crossing
BAT 1 from South leaving crossing
South is signeled from queue
BAT 5 from South arrives at crossing
west is signeled from cross
BAT 3 from west crossing
BAT 3 from west leaving crossing
North is signeled from cross
BAT 4 from North crossing
west is signeled from queue
BAT 7 from west arrives at crossing
BAT 4 from North leaving crossing
North is signeled from queue
BAT 8 from North arrives at crossing
East is signeled from cross
BAT 6 from East crossing
BAT 6 from East leaving crossing
South is signeled from cross
```

```
BAT 11 from west arrives at crossing
BAT 8 from North crossing
BAT 8 from North leaving crossing
North is signeled from queue
BAT 19 from North arrives at crossing
DEADLOCK: BAT jam detected, signalling any waiting thread
North is signeled from cross
BAT 19 from North crossing
BAT 19 from North leaving crossing
East is signeled from cross
BAT 10 from East crossing
BAT 10 from East leaving crossing
East is signeled from queue
BAT 15 from East arrives at crossing
South is signeled from cross
BAT 9 from South crossing
BAT 9 from South leaving crossing
South is signeled from queue
BAT 18 from South arrives at crossing
BAT 15 from East crossing
BAT 15 from East leaving crossing
South is signeled from cross
BAT 18 from South crossing
East is signeled from queue
BAT 16 from East arrives at crossing
BAT 18 from South leaving crossing
west is signeled from cross
BAT 11 from west crossing
BAT 11 from west leaving crossing
west is signeled from queue
BAT 12 from west arrives at crossing
BAT 12 from west crossing
BAT 12 from west leaving crossing
west is signeled from queue
BAT 13 from west arrives at crossing
BAT 16 from East crossing
BAT 16 from East leaving crossing
East is signeled from queue
BAT 17 from East arrives at crossing
BAT 13 from west crossing
BAT 13 from west leaving crossing
west is signeled from queue
BAT 17 from East crossing
BAT 14 from west arrives at crossing
BAT 17 from East leaving crossing
BAT 14 from west crossing
BAT 14 from west leaving crossing
zeny@pop-os:~/Documents/projects/mutual execlusion$
```