108021209 李思諭

what does your timer-0 ISR have to do to support these multiple delays and now()?
In my myTimer0Handler,
I add

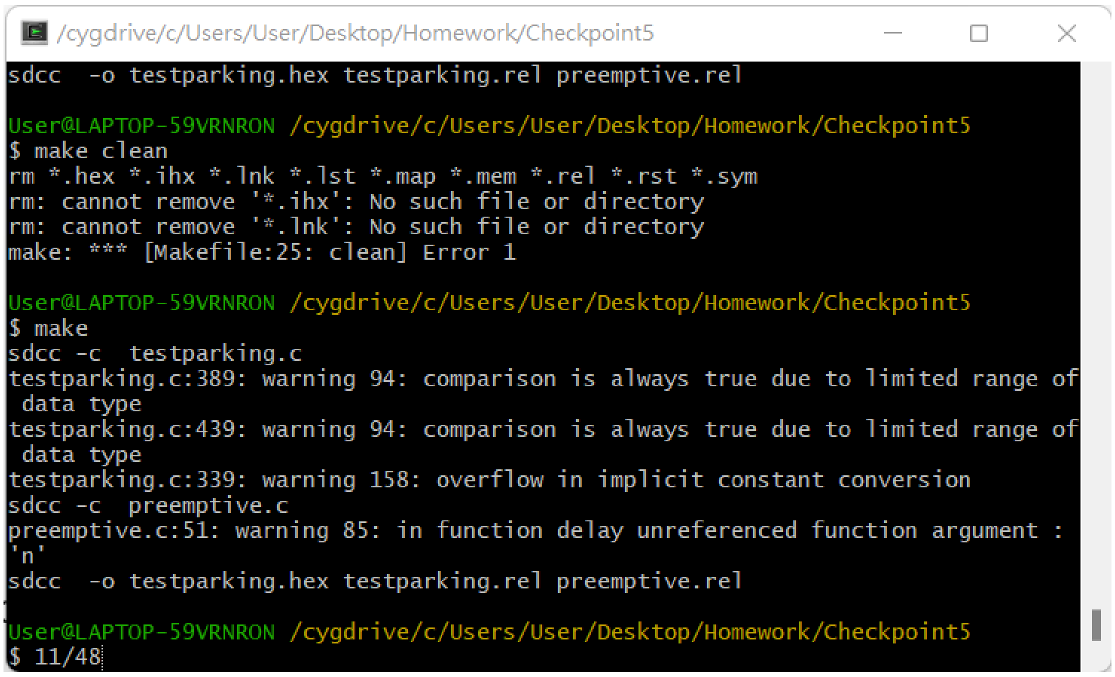
```
        Time++;

        if(Time%16 == 0){
                for(int i=0;i<2;i++){
                        if(delay_time[i]>0){
                                delay_time[i]--;
                        }
                }
        }
```

to support these multiple delays and now().

what if all threads call delay() and happen to finish their delays all at the same time? How can you ensure the accuracy of your delay? (i.e., between n and n+0.5 time units)?
When executing myTimer0Handler, Time will increase by one, i.e., Time++.
And my time unit is 16*(the time between executing myTimer0Handler).
In the period of time unit(在任意的 time unit 這段時間內), there is at least one chance for all threads to take a note of when the car leaves the spot.
We can take a note of when the car leaves the spot at the corrected time and ensure the accuracy of my delay.

How does the worst-case delay completion (i.e., all threads finish delaying at the same time) affect your choice of time unit?
If all threads finish delaying at the same time, I need to choose the time unit large enough so that I can ensure every thread can have one chance to take a note of when the car leaves the spot.
Thus, time unit >= 4*(the time between executing myTimer0Handler).
I choose time unit = 16*(the time between executing myTimer0Handler).

A typescript showing compilation of your code



```
sdcc  -o testparking.hex testparking.rel preemptive.rel

User@LAPTOP-59VRNRON /cygdrive/c/Users/User/Desktop/Homework/Checkpoint5
$ make clean
rm *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym
rm: cannot remove '*.ihx': No such file or directory
rm: cannot remove '*.lnk': No such file or directory
make: *** [Makefile:25: clean] Error 1

User@LAPTOP-59VRNRON /cygdrive/c/Users/User/Desktop/Homework/Checkpoint5
$ make
sdcc -c  testparking.c
testparking.c:389: warning 94: comparison is always true due to limited range of
 data type
testparking.c:439: warning 94: comparison is always true due to limited range of
 data type
testparking.c:339: warning 158: overflow in implicit constant conversion
sdcc -c  preemptive.c
preemptive.c:51: warning 85: in function delay unreferenced function argument :
'n'
sdcc  -o testparking.hex testparking.rel preemptive.rel

User@LAPTOP-59VRNRON /cygdrive/c/Users/User/Desktop/Homework/Checkpoint5
$ 11/48
```

The result after executing the code when the delay time is:

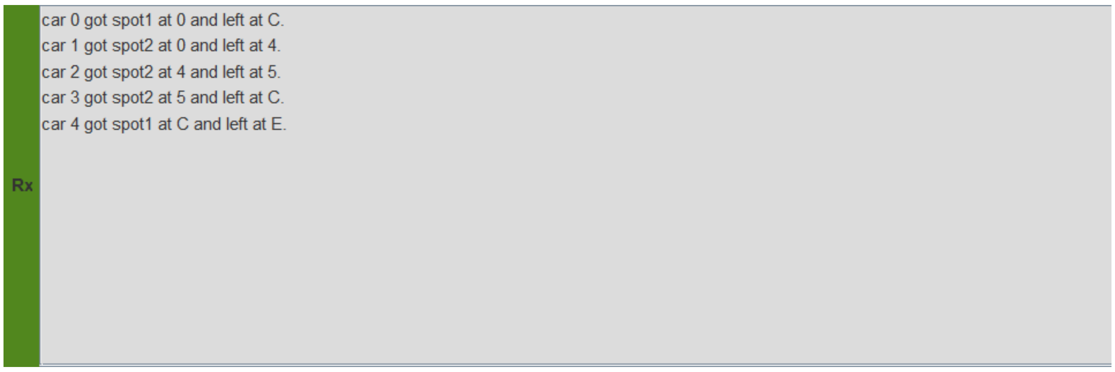
Car0:12

Car1:4

Car2:1

Car3:7

Car4:2



```
car 0 got spot1 at 0 and left at C.
car 1 got spot2 at 0 and left at 4.
car 2 got spot2 at 4 and left at 5.
car 3 got spot2 at 5 and left at C.
car 4 got spot1 at C and left at E.
```

In Car0,

When spot[i] is zero and car_spot[0] is zero, I take a note of which spot Car0 is in and when Car0 get the spot, set the delay time, and set spot[i] to 1.

When (delay time for Car0 is zero) and (spot[car_spot[0]-1]) and (car_leave_spot_time[0] is zero), I set the spot Car0 just left to 0 and take a note of when Car0 left and execute ThreadExit().

Similar for Car1, Car2, Car3, Car4.

Using the semaphore to limit the creation of thread

I create a semaphore num_thread in the main function.

And when num_thread is 2, it means that there is only one thread and the thread is for the main function.

I add

```
if(fp != main){
            SemaphoreWait(num_thread);
      }
```

in ThreadCreate function.

When we are not create a thread for main, we need to do SemaphoreWait(num_thread);.

I add SemaphoreSignal(num_thread); in ThreadExit function. It means that if we want to create a thread, we can create a new thread.

I add

```
if(bitmap_for_Thread == 0){ //bitmap_for_Thread==0000
                              while(1); // the last thread to exit, enter an infinite loop
                  }
```

in ThreadExit function.

It means that the last thread is exiting, and then it should enter an infinite loop.