Operating System CP3

許木羽 / 111000177

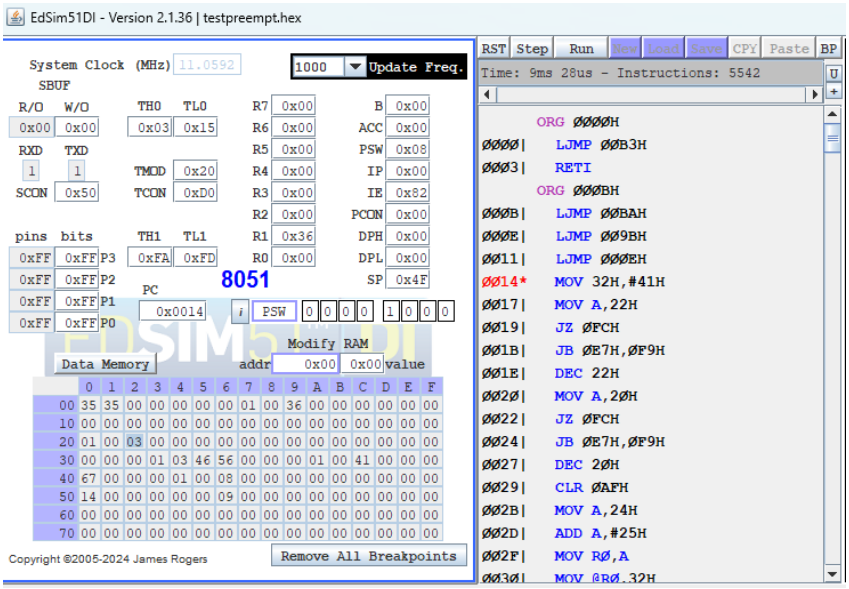
Instead of CPU busy waiting, we try to use semaphore here. Besides, the advantage is better Thread Synchronization.

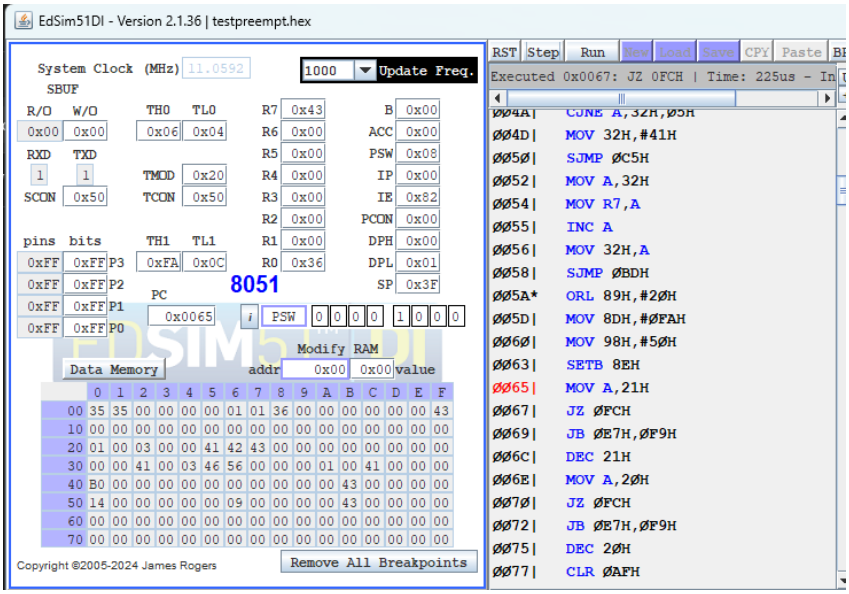
```
PS D:\College\OS\ppc3> mingw32-make clean
del /Q *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym *.asm *.lk
PS D:\College\OS\ppc3> mingw32-make
sdcc -c  testpreempt.c
sdcc -c  preemptive.c
preemptive.c:93: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
sdcc  -o testpreempt.hex testpreempt.rel preemptive.rel
PS D:\College\OS\ppc3>
```

```
      Value  Global                           Global Defined In Module
      -----  -------------------------------  ------------------------
C:    00000014  _Producer                       testpreempt
C:    0000005A  _Consumer                       testpreempt
C:    0000009B  _main                           testpreempt
C:    000000B3  __sdcc_gsinit_startup           testpreempt
C:    000000B7  __mcs51_genRAMCLEAR             testpreempt
C:    000000B8  __mcs51_genXINIT                testpreempt
C:    000000B9  __mcs51_genXRAMCLEAR            testpreempt
C:    000000BA  _timer0_ISR                     testpreempt
C:    000000BE  _Bootstrap                      preemptive
C:    000000E4  _ThreadCreate                   preemptive
C:    00000161  _ThreadYield                    preemptive
C:    000001BB  _ThreadExit                     preemptive
C:    000001D1  _myTimer0Handler                preemptive
```

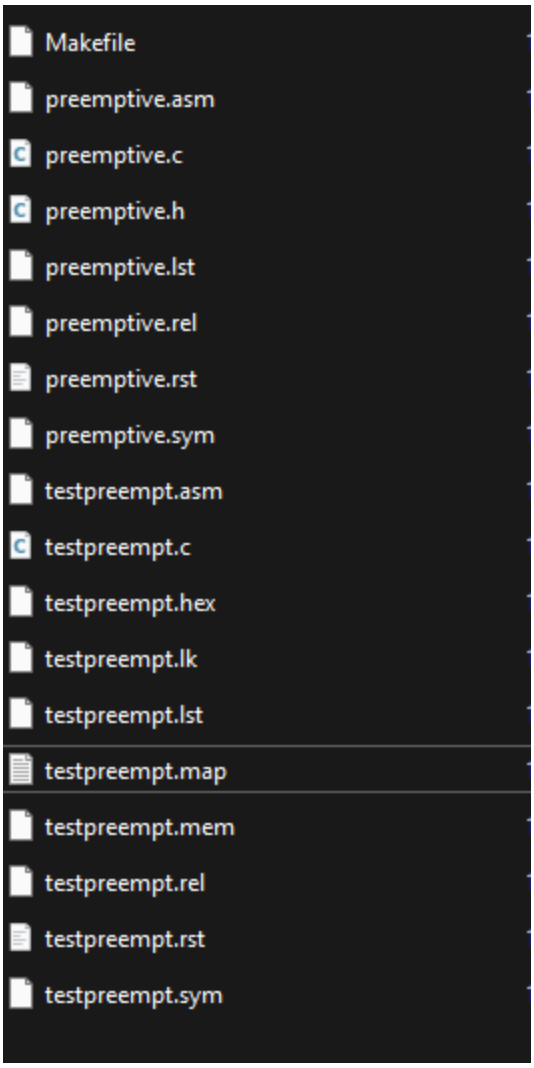
```
   Value  Global                           Global Defined In Module
   -----  -------------------------------  ------------------------
  00000000  .__.ABS.                        preemptive
  00000020  _mutex                          testpreempt
  00000021  _full                           testpreempt
  00000022  _empty                          testpreempt
  00000023  _head                           testpreempt
  00000024  _tail                           testpreempt
  00000025  _globalBuffer                   testpreempt
  00000032  _producerChar                   testpreempt
  00000033  _currentThreadID                preemptive
```

Semaphore implementation by creating a new label in the initialization. Inside the label, it'll check if the value of semaphore is 1, if it's it'll go to the next line of code, but if it's not, it will jump back to the label and wait. It'll reset the value to 0 after receiving a 1 signal.





Above is Producer and Below is Consumer when waiting signal. In this case, the Consumer code of waiting starts from 5A and need to wait 0x21 signal, and if it the case it will repeating his step. Also, for the Producer need to wait but in my case it's often to have the signal ready so it will not have a self loop.


Makefile
preemptive.asm
preemptive.c
preemptive.h
preemptive.lst
preemptive.rel
preemptive.rst
preemptive.sym
testpreempt.asm
testpreempt.c
testpreempt.hex
testpreempt.lk
testpreempt.lst
testpreempt.map
testpreempt.mem
testpreempt.rel
testpreempt.rst
testpreempt.sym