CSC 343 Operating Systems, Spring 2023, Dr. Dale E. Parson

Assignment 2, Implementing ManytoOne and ManyToMany user-to-kernel mode scheduling.
This assignment is due via **make turnitin** from the scheduleUserKernel2023 directory by **11:59 PM on Sunday March 12**. There is a 10% penalty for each day it is late, and I will not accept solutions after I go over my solution in class. I will go over this handout on February 23. The class Zoom recordings augment this handout.

The goal of this assignment is to start with my supplied one-to-one user mode -to- kernel mode access scheduler to the kernel as a starting point for you to implement many-to-one and many-to-many (that I call "many-to-fewer) kernel scheduling. There is a README.txt file with questions for you to answer after you have completed the code. **Answers in README.txt are worth 20% of this assignment, so remember that working code is not the end of the requirements**.

Perform the following steps to get my handout. You will code and test on **mcgonagall**, to which you can **ssh mcgonagall** from acad. **Do not change the name of the project directory, since my automated tests depend on that name**.

cd $HOME   # or start out in your login directory
mkdir OpSys # All of this semester's work goes under here, skip if you did it before.
cd ./OpSys
cp ~parson/OpSys/scheduleUserKernel2023.problem.zip scheduleUserKernel2023.problem.zip
unzip scheduleUserKernel2023.problem.zip
cd ./scheduleUserKernel2023
make clean test

Testing passes OneToOne.stm but fails ManyToOne.stm within the handout directory. Some subsequent failures may hang the compiler or simulation, requiring a control-C to abort. Successful **make clean test** finishes in about three seconds.

STEPS:
1. **cp OneToOne.stm ManyToOne.stm** and edit the latter model after you understand the working of OneToOne.stm and the STUDENT instructions copied into ManyToOne.stm. You can test my handout code with **make test_OneToOne** and your ManyToOne.stm with **make test_ManyToOne**.

One-to-One User-to-Kernel   **Many-to-One User-to-Kernel**   **Many-to-Many User-to-Kernel**
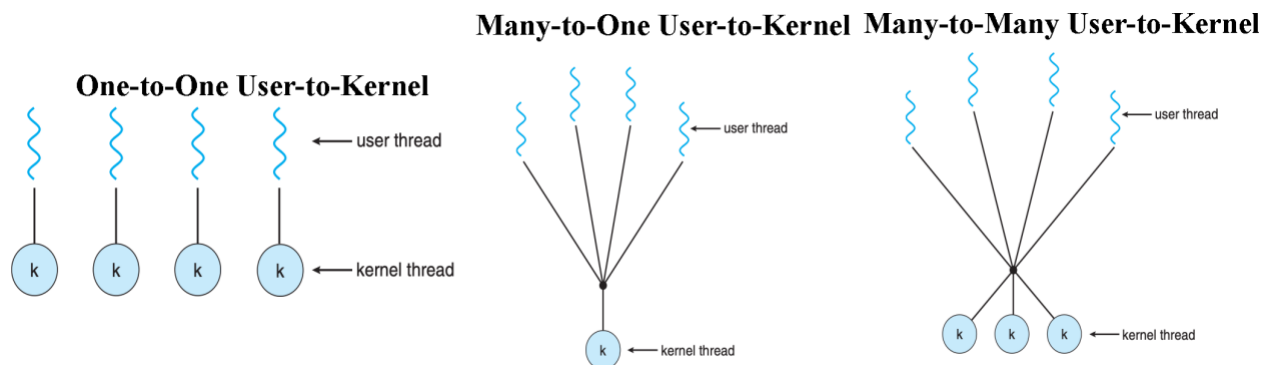
Figures from slides 14-16 of Textbook Chapter 4 PowerPoint

2. After **make test_ManyToOne** is working, **cp ManyToOne.stm ManyToMany.stm** and edit the latter model per STUDENT instructions. Running **make test_ManyToMany** tests that model, and **make test** tests all three models.

All of the detailed instructions and grading weights for your code additions appear in **STUDENT** comments in the source code. The other file is README.txt. We will go over your project requirements in class on 2/23.

I have also included a previous year's Mutex.stm to show how to enqueue a thread object in a STM Queue object and how to conditionally dequeue a thread from a non-empty Queue and send it an event from the dequeuing thread. This model is NOT part of this assignment, but Mutex.stm shows you how to enqueue and dequeue blocking threads in waiting queues and how to use model functions **waitForEvent** and **signalEvent** to communicate events between threads. You need to use Queues and signal between threads in the current assignment.

If you get an error message at run-time that gives an index into __codeTable__ like this:

Traceback (most recent call last):
  File "OneToOne.py", line 612, in <module>
    main()
  File "OneToOne.py", line 584, in main
    scheduler.__run__()
  File "/home/kutztown.edu/parson/OpSys/state2codeV17/CSC343Sim.py", line 145, in __run__
    waitingObject.__generator__.__next__() # run() the model
  File "OneToOne.py", line 150, in run
    exec(__codeTable__[1],globals,locals)
  File "nofile", line 1, in <module>
  File "nofile", line 1, in <listcomp>
NameError: name 'false' is not defined
0.09user 0.02system 0:00.14elapsed 77%CPU (0avgtext+0avgdata 10164maxresident)k
0inputs+8outputs (0major+5004minor)pagefaults 0swaps
make: *** [test_OneToOne] Error 1

[:-) ~/.../solutions/scheduleUserKernel2023] **./decode.py OneToOne.py 1**

__codeTable__[1] = compile('processor.WaitQueues = [Queue(false)   for i in range(0, processor.KernelRegions)]','nofile','exec'),

A successful test run appears as follows.

$ **make test**
COMPILING OneToOne
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7
/home/kutztown.edu/parson/OpSys/state2codeV17/State2CodeParser.py OneToOne.stm OneToOne.dot
OneToOne.py CSC343Compile CSC343Compile"
/bin/rm -f *.jpg *.png
COMPILING COMPLETED
SIMULATING (TESTING) OneToOne
/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log *.log
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. STMLOGDIR=~parson/tmp time
/usr/local/bin/python3.7 OneToOne.py 2 4 100000 12345 3"

MSG cmd line: ['OneToOne.py', '2', '4', '100000', '12345', '3'], usage USAGE: python THISFILE.py NUMCONTEXTS NUMFASTIO SIMTIME SEED|None LOGLEVEL
Simulation exiting at time 100001, which meets or exceeds limit 100000.

0.31user 0.07system 0:00.49elapsed 78%CPU (0avgtext+0avgdata 10532maxresident)k
0inputs+2248outputs (0major+5092minor)pagefaults 0swaps
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7 crunchlog.py OneToOne.log"

DIFFing OneToOne_crunch.py OneToOne_crunch.ref
OK: MIN_stallThread at 20.0% tolerance = 0
OK: MEAN_stallThread at 20.0% tolerance = 0
OK: MAX_stallThread at 20.0% tolerance = 0

COMPLETED (OK) SIMULATING (TESTING) OneToOne
COMPILING ManyToOne
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7 /home/kutztown.edu/parson/OpSys/state2codeV17/State2CodeParser.py ManyToOne.stm ManyToOne.dot ManyToOne.py CSC343Compile CSC343Compile"
/bin/rm -f *.jpg *.png
COMPILING COMPLETED
SIMULATING (TESTING) ManyToOne
/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log *.log
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. STMLOGDIR=~parson/tmp time /usr/local/bin/python3.7 ManyToOne.py 2 4 100000 12345 3"
MSG cmd line: ['ManyToOne.py', '2', '4', '100000', '12345', '3'], usage USAGE: python THISFILE.py NUMCONTEXTS NUMFASTIO SIMTIME SEED|None LOGLEVEL
Simulation exiting at time 100421, which meets or exceeds limit 100000.

0.14user 0.03system 0:00.25elapsed 72%CPU (0avgtext+0avgdata 10552maxresident)k
0inputs+384outputs (0major+5103minor)pagefaults 0swaps
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7 crunchlog.py ManyToOne.log"

DIFFing ManyToOne_crunch.py ManyToOne_crunch.ref
OK: MIN_stallThread at 20.0% tolerance = 499
OK: MEAN_stallThread at 20.0% tolerance = 17137.01
OK: MAX_stallThread at 20.0% tolerance = 19841

COMPLETED (OK) SIMULATING (TESTING) ManyToOne
COMPILING ManyToMany
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7 /home/kutztown.edu/parson/OpSys/state2codeV17/State2CodeParser.py ManyToMany.stm ManyToMany.dot ManyToMany.py CSC343Compile CSC343Compile"
/bin/rm -f *.jpg *.png
COMPILING COMPLETED
SIMULATING (TESTING) ManyToMany
/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log *.log
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. STMLOGDIR=~parson/tmp time

/usr/local/bin/python3.7 ManyToMany.py 2 4 100000 12345 3"
MSG cmd line: ['ManyToMany.py', '2', '4', '100000', '12345', '3'], usage USAGE: python THISFILE.py
NUMCONTEXTS NUMFASTIO SIMTIME SEED|None LOGLEVEL
Simulation exiting at time 100421, which meets or exceeds limit 100000.

0.23user 0.05system 0:00.38elapsed 76%CPU (0avgtext+0avgdata 10552maxresident)k
0inputs+1368outputs (0major+5098minor)pagefaults 0swaps
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7
crunchlog.py ManyToMany.log"

DIFFing ManyToMany_crunch.py ManyToMany_crunch.ref
OK: MIN_stallThread at 20.0% tolerance = 491
OK: MEAN_stallThread at 20.0% tolerance = 3538.58
OK: MAX_stallThread at 20.0% tolerance = 5562

COMPLETED (OK) SIMULATING (TESTING) ManyToMany

Any time a COMPILE succeeds, you can look at the graph for your state machine by running **make graphs** and then inspecting https://kuvapcsitrd01.kutztown.edu/~STUDENT/OneToOne.jpg or https://kuvapcsitrd01.kutztown.edu/~STUDENT/ManyToOne.jpg, where STUDENT is your login ID. If you can't get at it with a browser this way, use WinSCP or FileZilla to copy the JPEG file from your project directory to your local machine. Below are the final, correct graphs. It appears that **make graphs** now works on mcgonagall.

Once **make clean test** passes, **ANSWER THE QUESTIONS IN FILE README.txt** included in this project directory. Follow all instructions in README.txt.

Finally, turn it in by entering **make turnitin** and following the prompt. We do not use the turnin script in this course; instead **make turnitin** turns in the project; it prompts you for a carriage return (Enter) to complete its work.

I will distribute grades via email before the next class after the due date.
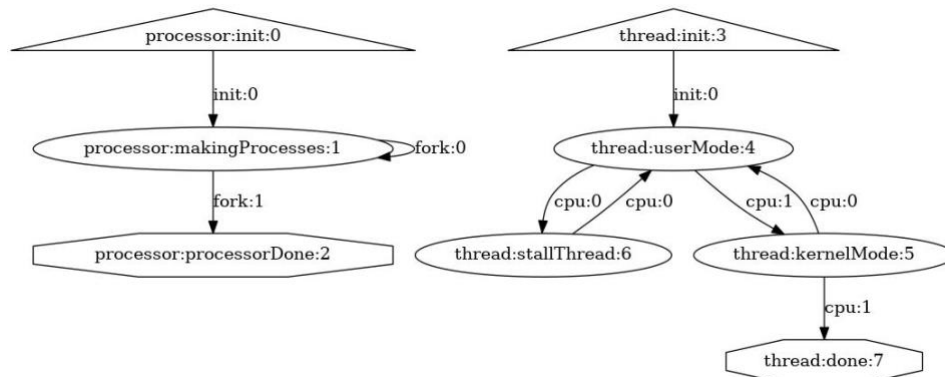
OneToOne.stm
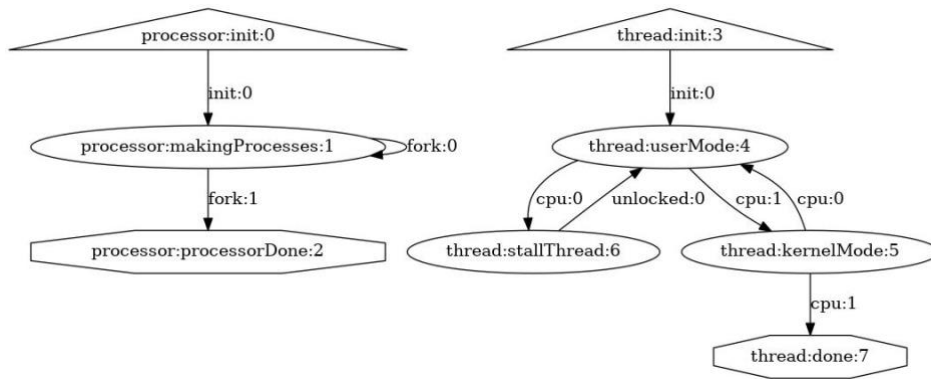
ManyToOne.stm and ManyToMany.stm; note the *unlocked* event is the only change from OneToOne.stm