

CLAIMS:

1. A computer-implemented method of controlling execution of threads in a computing environment, said computer-implemented method comprising:

stopping, by a thread running in a core of the processor of the computing environment, execution of another thread executing within the core of the processor, the stopping using one or more controls in one or more shared registers of the processor, the one or more shared registers being shared by the thread and the other thread, the stopping comprising:

determining, by the thread, whether the other thread is prohibiting being stopped;

stopping, by the thread, instruction fetching and execution on the other thread, based on the determining indicating the other thread is not prohibiting being stopped; and

checking status of the other thread to determine whether execution of the other thread has stopped, wherein the performing the one or more operations is based on the checking indicating execution of the other thread has stopped and that the other thread did not change from not prohibiting being stopped to prohibiting being stopped after the determining and before the stopping;

performing by the thread, one or more operations within the processor after the other thread was stopped from executing within the processor by the thread; and

based on completing the one or more operations, allowing, by the thread, the other thread to continue executing within the processor.

2. The computer-implemented method of claim 1, wherein the stopping execution of the other thread comprises:

obtaining status information for the other thread; and

determining, based on the status information, whether execution of the other thread is stopped, wherein the one or more operations are performed based on the determining indicating execution of the other thread is stopped.

3. The computer-implemented method of claim 2, wherein the determining comprises using a drain instruction configured to obtain status of the other thread.

4. The computer-implemented method of claim 1, wherein the one or more controls includes one or more indicators in the one or more shared registers, and wherein the determining whether the other thread is prohibiting being stopped comprises checking at least one indicator of the one or more indicators.
5. The computer-implemented method of claim 1, wherein the one or more controls includes one or more indicators in the one or more shared registers, and wherein the stopping instruction fetching and execution on the other thread includes setting an indicator of the one or more indicators to stop instruction fetching and execution on the other thread.
6. The computer-implemented method of claim 1, wherein the checking status of the other thread comprises using a drain instruction, the drain instruction configured to hold instruction dispatch on the thread and query status for the other thread.
7. The computer-implemented method of claim 6, wherein the drain instruction specifies one or more conditions to be satisfied prior to the performing the one or more operations, and wherein the performing the one or more operations is executed based on a result of the drain instruction indicating satisfaction of the one or more conditions.
8. The computer-implemented method of claim 1, wherein the method further comprises:
 - re-determining, based on the checking indicating execution of the other thread has stopped, whether the other thread is prohibiting being stopped;
 - allowing execution on the other thread, based on the re-determining indicating the other thread is prohibiting being stopped; and
 - performing the one or more operations, based on the re-determining indicating the other thread is not prohibiting being stopped.
9. The computer-implemented method of claim 1, wherein the stopping comprises stopping execution of a plurality of threads.
10. A computer system for controlling execution of threads in a computing environment, said computer system comprising:
 - a memory; and
 - a processor in communications with the memory, wherein the computer system is configured to perform a method, said method comprising:

stopping, by a thread running in a core of the processor of the computing environment, execution of another thread executing within the core of the processor, the stopping using one or more controls in one or more shared registers of the processor, the one or more shared registers being shared by the thread and the other thread, the stopping comprising:

determining, by the thread, whether the other thread is prohibiting being stopped;

stopping, by the thread, instruction fetching and execution on the other thread, based on the determining indicating the other thread is not prohibiting being stopped; and

checking status of the other thread to determine whether execution of the other thread has stopped, wherein the performing the one or more operations is based on the checking indicating execution of the other thread has stopped and that the other thread did not change from not prohibiting being stopped to prohibiting being stopped after the determining and before the stopping;

performing by the thread, one or more operations within the processor after the other thread was stopped from executing within the processor by the thread; and

based on completing the one or more operations, allowing, by the thread, the other thread to continue executing within the processor.

11. The computer system of claim 10, wherein the stopping execution of the other thread comprises:

obtaining status information for the other thread; and

determining, based on the status information, whether execution of the other thread is stopped, wherein the one or more operations are performed based on the determining indicating execution of the other thread is stopped.

12. The computer system of claim 11, wherein the determining comprises using a drain instruction configured to obtain status of the other thread.

13. The computer system of claim 10, wherein the one or more controls includes one or more indicators in the one or more shared registers, and wherein the determining whether the other thread is prohibiting being stopped comprises checking at least one indicator of the one or more indicators.

14. The computer system of claim 10, wherein the one or more controls includes one or more indicators in the one or more shared registers, and wherein the stopping instruction fetching and execution on the other thread includes setting an indicator of the one or more indicators to stop instruction fetching and execution on the other thread.
15. The computer system of claim 10, wherein the checking status of the other thread comprises using a drain instruction, the drain instruction configured to hold instruction dispatch on the thread and query status for the other thread.
16. The computer system of claim 15, wherein the drain instruction specifies one or more conditions to be satisfied prior to the performing the one or more operations, and wherein the performing the one or more operations is executed based on a result of the drain instruction indicating satisfaction of the one or more conditions.
17. The computer system of claim 10, wherein the method further comprises:
 - re-determining, based on the checking indicating execution of the other thread has stopped, whether the other thread is prohibiting being stopped;
 - allowing execution on the other thread, based on the re-determining indicating the other thread is prohibiting being stopped; and
 - performing the one or more operations, based on the re-determining indicating the other thread is not prohibiting being stopped.
18. The computer system of claim 10, wherein the stopping comprises stopping execution of a plurality of threads.
19. A computer program product for controlling execution of threads in a computing environment, said computer program product comprising:
 - a computer readable storage medium readable by a processing circuit and storing instructions for execution by the processing circuit for performing a method comprising:
 - stopping, by a thread running in a core of the processor of the computing environment, execution of another thread executing within the core of the processor, the stopping using one or more controls in one or more shared registers of the processor, the one or more shared registers being shared by the thread and the other thread, the stopping comprising:

determining, by the thread, whether the other thread is prohibiting being stopped;

stopping, by the thread, instruction fetching and execution on the other thread, based on the determining indicating the other thread is not prohibiting being stopped; and

checking status of the other thread to determine whether execution of the other thread has stopped, wherein the performing the one or more operations is based on the checking indicating execution of the other thread has stopped and that the other thread did not change from not prohibiting being stopped to prohibiting being stopped after the determining and before the stopping;

performing by the thread, one or more operations within the processor after the other thread was stopped from executing within the processor by the thread; and

based on completing the one or more operations, allowing, by the thread, the other thread to continue executing within the processor.

20. The computer program product of claim 19, wherein the stopping execution of the other thread comprises:

obtaining status information for the other thread; and

determining, based on the status information, whether execution of the other thread is stopped, wherein the one or more operations are performed based on the determining indicating execution of the other thread is stopped.

21. The computer program product of claim 19, wherein the checking status of the other thread comprises using a drain instruction, the drain instruction configured to hold instruction dispatch on the thread and query status for the other thread, and wherein the drain instruction specifies one or more conditions to be satisfied prior to the performing the one or more operations, and wherein the performing the one or more operations is executed based on a result of the drain instruction indicating satisfaction of the one or more conditions.

22. The computer program product of claim 19, wherein the method further comprises:

re-determining, based on the checking indicating execution of the other thread has stopped, whether the other thread is prohibiting being stopped;

allowing execution on the other thread, based on the re-determining indicating the other thread is prohibiting being stopped; and

performing the one or more operations, based on the re-determining indicating the other thread is not prohibiting being stopped.