

WHAT IS CLAIMED IS:

1. A method for process handling, performed by a virtual machine monitor, comprising:
 - detecting a blocking time of a kernel process in response to an ending of handling a current interrupt event in an interrupt queue;
 - triggering a soft interrupt event of a virtual processor when the blocking time of the kernel process exceeds a preset time threshold; and
 - sending an intermediate start instruction to the virtual processor in response to the soft interrupt event of the virtual processor being triggered, to make the virtual processor run for a preset time period.
2. The process handling method according to claim 1, wherein before detecting the blocking time of the kernel process in response to the end of handling of the soft interrupt event, the method further comprises:
 - handling a current interrupt event in an interrupt queue in response to detecting the interrupt event being triggered.
3. The process handling method according to claim 2, wherein before handling the current interrupt event in the interrupt queue in response to detecting the interrupt event being triggered, the method further comprises:
 - acquiring a run queue corresponding to the kernel process, and
 - executing the run queue in units of preset running units of the virtual processor, according to an arrangement sequence of the preset running units in the run queue, until a process scheduling command is received, wherein one or more preset running units of the virtual processor are placed in the run queue.

4. The process handling method according to claim 3, wherein before acquiring the run queue corresponding to the kernel process, and executing the run queue in units of preset running units of the virtual processor, according to an arrangement sequence of the preset running units in the run queue until a process scheduling command is received, the method further comprises:

initializing the kernel process.

5. The process handling method according to claim 4, wherein initializing the kernel process comprises:

creating the kernel process;

in response to receiving a mount request sent by the virtual processor, placing one or more preset running units of the virtual processor in the run queue corresponding to the kernel process, and sending a mount success message to the virtual processor; and

starting the kernel process in response to receiving a kernel process wake-up request sent by the virtual processor.

6. The process handling method according to any one of claims 1-5, further comprising:

in response to receiving a preset running request, sending an intermediate wake-up request to the virtual processor, and entering a sleep state after receiving a wake-up success message sent by the virtual processor, wherein the intermediate wake-up request carries information about the preset running request; and

entering a running state in response to receiving a preset running request handling end message sent by the virtual processor.

7. The process handling method according to any one of claims 1-5, further comprising:

handling a next interrupt event in the interrupt queue in response to an end of the preset time period.

8. A method for process handling, performed by a virtual processor, comprising:
initializing the virtual processor;

starting an intermediate running in response to receiving an intermediate start instruction sent by a virtual machine monitor in response to a soft interrupt event of the virtual processor being triggered, wherein a length of time for the intermediate running is a preset time period; and

stopping the intermediate running in response to an ending of the preset time period.

9. The process handling method according to claim 8, wherein initializing a virtual processor comprises:

generating one or more preset running units;

sending a mount request to the virtual machine monitor to place the one or more preset running units in a run queue corresponding to a kernel process of the virtual machine monitor; and

in response to receiving a mount success message sent by the virtual machine monitor, sending a kernel process wake-up request to the virtual machine monitor and entering a sleep state, so that the virtual machine monitor starts the kernel process.

10. The process handling method according to claim 8 or 9, further comprising:

in response to receiving an intermediate wake-up request sent by the virtual machine monitor, entering an intermediate running state from the sleep state, and sending a wake-up success message to the virtual machine monitor, wherein the intermediate wake-up request carries information about a preset running request; and

executing the preset running request, and after an end of handling the preset running request, sending a preset running request handling end message to the virtual machine monitor to enter the sleep state.

11. A method for process handling, comprising:

initializing a kernel process of a virtual machine monitor and a virtual processor;

in response to a soft interrupt event of the virtual processor being triggered, performing an intermediate start on the virtual processor when an interrupt event of the kernel process is triggered and a preset condition is met, so that the virtual processor runs for a preset time period; and

handling, by the virtual machine monitor, a next interrupt event in an interrupt queue in response to an ending of the preset time period.

12. The process handling method according to claim 11, wherein initializing the kernel process of the virtual machine monitor further comprises:

creating the kernel process;

in response to receiving a mount request sent by the virtual processor, placing one or more preset running units of the virtual processor in a run queue corresponding to the kernel process, and sending a mount success message to the virtual processor; and

starting the kernel process in response to receiving a kernel process wake-up request sent by the virtual processor.

13. The process handling method according to claim 11 or 12, wherein initializing a virtual processor further comprises:

generating one or more preset running units;

sending a mount request to the virtual machine monitor to place the one or more preset running units in a run queue corresponding to the kernel process of the virtual machine monitor; and

in response to receiving a mount success message sent by the virtual machine monitor, sending a kernel process wake-up request to the virtual machine monitor and entering a sleep state, so that the virtual machine monitor starts the kernel process.

14. The process handling method according to claim 11 or 12, wherein performing the intermediate start on the virtual processor when the interrupt event of the kernel process is triggered and a preset condition is met comprises:

detecting, by the virtual machine monitor, a blocking time of the kernel process in response to an end of the handling of a current interrupt event in the interrupt queue;

triggering, by the virtual machine monitor, a soft interrupt event of the virtual processor when the blocking time of the kernel process exceeds a preset time threshold;

sending, by the virtual machine monitor, an intermediate start instruction to the virtual processor in response to that the soft interrupt event of the virtual processor is triggered;

starting, by the virtual processor, an intermediate running in response to receiving the intermediate start instruction sent by the virtual machine monitor, wherein a length of time for the intermediate running is a preset time period; and

stopping, by the virtual processor, the intermediate running in response to an ending of the preset time period.

15. The process handling method according to claim 14, wherein before detecting, by the virtual machine monitor, a blocking time of the kernel process in response to an end of the handling of the current interrupt event in the interrupt queue, the method further comprises:

handling, by the virtual machine monitor, the current interrupt event in the interrupt queue in response to detecting the interrupt event being triggered.

16. The process handling method according to claim 15, wherein before handling, by the virtual machine monitor, the current interrupt event in the interrupt queue in response to detecting the interrupt event being triggered, the method further comprises:

acquiring, by the virtual machine monitor, a run queue corresponding to the kernel process, and executing the run queue in units of preset running units of the virtual processor, according to an arrangement sequence of preset running units in the run queue, until a process scheduling command is received, wherein one or more preset running units of the virtual processor are placed in the run queue.

17. The process handling method according to claim 11 or 12, further comprising: performing, by the virtual machine monitor, an intermediate wake-up for the virtual processor in response to receiving a preset running request.

18. The process handling method according to claim 17, wherein performing, by the virtual machine monitor, the intermediate wake-up for the virtual processor in response to receiving the preset running request comprises:

sending, by the virtual machine monitor, an intermediate wake-up request to the virtual processor in response to receiving the preset running request, wherein the intermediate wake-up request carries information about the preset running request;

in response to receiving the intermediate wake-up request sent by the virtual machine monitor, the virtual processor entering an intermediate running state from a sleep state, and sending a wake-up success message to the virtual machine monitor;

the virtual machine monitor entering the sleep state after receiving the wake-up success message sent by the virtual processor;

the virtual processor executing the preset running request, and after the handling of the preset running request ends, sending a preset running request handling end message to the virtual machine monitor to enter the sleep state; and

the virtual machine monitor entering a running state in response to receiving the preset running request handling end message sent by the virtual processor.

19. An apparatus for process handling, comprising:

a detecting module configured to detect a blocking time of a kernel process in response to an end of the handling of a current interrupt event in an interrupt queue;

a triggering module configured to trigger a soft interrupt event of a virtual processor when the blocking time of the kernel process exceeds a preset time threshold; and

a first sending module configured to send an intermediate start instruction to the virtual processor in response to that the soft interrupt event of the virtual processor is triggered, so that the virtual processor runs for a preset time period.

20. The process handling apparatus according to claim 19, wherein before the detecting module, the apparatus further comprises:

a first handling module configured to handle the current interrupt event in the interrupt queue in response to detecting an interrupt event being triggered.

21. The process handling apparatus according to claim 20, wherein before the first handling module, the apparatus further comprises:

a first running module configured to acquire a run queue corresponding to the kernel process, and execute the run queue in units of preset running units of the virtual processor, according to an arrangement sequence of preset running units in the run queue, until a process scheduling command is received, wherein one or more preset running units of the virtual processor are placed in the run queue.

22. The process handling apparatus according to claim 21, wherein before the first running module, the apparatus further comprises:

a first initializing module configured to initialize the kernel process.

23. The process handling apparatus according to claim 22, wherein the first initializing module further comprises:

a creating sub-module configured to create the kernel process;

a mount sub-module configured to, in response to receiving a mount request sent by the virtual processor, place one or more preset running units of the virtual processor in the run queue corresponding to the kernel process, and send a mount success message to the virtual processor; and

a first starting sub-module configured to start the kernel process in response to receiving a kernel process wake-up request sent by the virtual processor.

24. The process handling apparatus according to any one of claims 19 to 23, further comprising:

a second sending module configured to, in response to receiving a preset running request, send an intermediate wake-up request to the virtual processor, and enter a sleep state after receiving a wake-up success message sent by the virtual processor, wherein the intermediate wake-up request carries information about the preset running request; and

a second running module configured to enter a running state in response to receiving a preset running request handling end message sent by the virtual processor.

25. The process handling apparatus according to any one of claims 19 to 23, further comprising:

a second handling module configured to handle a next interrupt event in the interrupt queue in response to an end of the preset time period.

26. An apparatus for process handling, comprising:

a second initializing module configured to initialize a virtual processor;

a first starting module configured to start an intermediate running in response to receiving an intermediate start instruction sent by a virtual machine monitor in response to a soft interrupt event of the virtual processor being triggered, wherein a length of time for the intermediate running is a preset time period; and

a stopping module configured to stop the intermediate running in response to an end of the preset time period.

27. The process handling apparatus according to claim 26, wherein the second initializing module further comprises:

a generating sub-module configured to generate one or more preset running units;

a first sending sub-module configured to send a mount request to the virtual machine monitor to place the one or more preset running units in a run queue corresponding to a kernel process of the virtual machine monitor; and

a second sending sub-module configured to, in response to receiving a mount success message sent by the virtual machine monitor, send a kernel process wake-up request to the virtual machine monitor and enter a sleep state, so that the virtual machine monitor starts the kernel process.

28. The process handling apparatus according to claim 26 or 27, further comprising:

a third sending module configured to, in response to receiving an intermediate wake-up request sent by the virtual machine monitor, enter an intermediate running state from the sleep state, and send a wake-up success message to the virtual machine monitor, wherein the intermediate wake-up request carries information about a preset running request; and

a third running module configured to run the preset running request, and after the handling of the preset running request ends, send a preset running request handling end message to the virtual machine monitor, and enter the sleep state.

29. A process handling apparatus, comprising:

a third initializing module configured to initialize a kernel process of a virtual machine monitor and a virtual processor;

a second starting module configured to in response to a soft interrupt event of the virtual processor being triggered, perform an intermediate start on the virtual processor when an interrupt event of the kernel process is triggered and a preset condition is met, so that the virtual processor runs for a preset time period; and

a third handling module configured to handle a next interrupt event in an interrupt queue in response to an end of the preset time period.

30. The process handling apparatus according to claim 29, wherein a part of initializing the kernel process of the virtual machine monitor in the third initializing module is configured to:

create the kernel process;

in response to receiving a mount request sent by the virtual processor, place one or more preset running units of the virtual processor in a run queue corresponding to the kernel process, and send a mount success message to the virtual processor; and

start the kernel process in response to receiving a kernel process wake-up request sent by the virtual processor.

31. The process handling apparatus according to claim 29 or 30, wherein a part of initializing the virtual processor in the third initializing module is configured to:

generate one or more preset running units;

send a mount request to the virtual machine monitor to place the one or more preset running units in a run queue corresponding to the kernel process of the virtual machine monitor; and

in response to receiving a mount success message sent by the virtual machine monitor, send a kernel process wake-up request to the virtual machine monitor and enter a sleep state, so that the virtual machine monitor starts the kernel process.

32. The process handling apparatus according to claim 29 or 30, wherein the second starting module further comprises:

- a detecting sub-module configured to detect a blocking time of the kernel process in response to an end of the handling of a current interrupt event in the interrupt queue;

- a triggering sub-module configured to trigger a soft interrupt event of the virtual processor when the blocking time of the kernel process exceeds a preset time threshold;

- a third sending sub-module configured to send an intermediate start instruction to the virtual processor in response to that the soft interrupt event of the virtual processor is triggered;

- a second starting sub-module configured to start an intermediate running in response to receiving an intermediate start instruction sent by the virtual machine monitor, wherein a length of time for the intermediate running is a preset time period; and

- a stopping sub-module configured to stop the intermediate running in response to an end of the preset time period.

33. The process handling apparatus according to claim 32, before the detecting sub-module, the apparatus further comprises:

- a handling sub-module configured to handle the current interrupt event in the interrupt queue in response to detecting that an interrupt event is triggered.

34. The process handling apparatus according to claim 33, before the handling sub-module, the apparatus further comprises:

a first running sub-module configured to acquire a run queue corresponding to the kernel process, and execute the run queue in units of preset running units of the virtual processor, according to an arrangement sequence of preset running units in the run queue, until a process scheduling command is received, wherein one or more preset running units of the virtual processor are placed in the run queue.

35. The process handling apparatus according to claim 29 or 30, further comprising:

a wake-up module configured to perform an intermediate wake-up for the virtual processor in response to receiving a preset running request.

36. The process handling apparatus according to claim 35, wherein the wake-up module further comprises:

a fourth sending sub-module configured to send an intermediate wake-up request to the virtual processor in response to receiving the preset running request, wherein the intermediate wake-up request carries information about the preset running request;

a fifth sending sub-module configured to, in response to receiving the intermediate wake-up request sent by the virtual machine monitor, enter an intermediate running state from a sleep state, and send a wake-up success message to the virtual machine monitor;

a first status changing sub-module configured to enter the sleep state after receiving the wake-up success message sent by the virtual processor;

a second running sub-module configured to run the preset running request, and after the handling of the preset running request ends, send a preset running request handling end message to the virtual machine monitor, and enter the sleep state; and

a second status changing sub-module configured to enter a running state in response to receiving the preset running request handling end message sent by the virtual processor.

37. An electronic device, comprising a memory and a processor, wherein the memory is configured to store one or more computer instructions, and the one or more computer instructions are executed by the processor to implement steps of the method according to any one of claims 1 to 18.

38. A computer-readable storage medium storing computer instructions thereon, wherein the computer instructions, when executed by one or more processors, implement steps of the method according to any one of claims 1 to 18.