WHAT IS CLAIMED IS:

1. An apparatus for operating an input/output (I/O) interface in a virtual machine, comprising:

one or more memory devices that stores a set of instructions; and

one or more hardware processors configured to execute the set of instructions to cause the apparatus to:

obtain a first mapping table that maps a set of host space virtual addresses to a first set of physical addresses associated with a first portion of the one or more memory devices;

obtain a second mapping table that maps a second set of physical addresses associated with a second portion of the one or more memory devices to the set of host space virtual addresses, wherein the second portion of the one or more memory devices is accessible by a virtual machine;

generate a third mapping table that maps the second set of physical addresses to the first set of physical addresses; and

map the first portion of the one or more memory devices associated with the first set of physical addresses to a configuration space of an I/O interface,

wherein the third mapping table is accessible by a device driver operating in the virtual machine and enables the device driver to directly access the configuration space of the

I/O interface to control the I/O interface by performing one or more memory operations to the first portion of the one or more memory devices associated with the first set of physical addresses.

2. The apparatus of claim 1, wherein the one or more hardware processors are further configured to execute the set of instructions to provide a virtualization controller and a host driver;

wherein the mapping of a set of host space virtual addresses to the first set of physical addresses is performed by the virtualization controller; and

wherein the mapping of the first portion of the one or more memory devices associated with the first set of physical addresses to the configuration space of the I/O interface is performed by the host driver after receiving an indication from the virtualization controller.

3. The apparatus of claim 1, wherein the one or more hardware processors are further configured to execute the set of instructions to cause the apparatus to:

map a set of guest space virtual addresses to the second set of physical addresses;

update the third mapping table to further map the set of guest space virtual addresses to the first set of physical addresses;

wherein the updated third mapping table is accessible by the device driver,

wherein the updated third mapping table enables the device driver to access the

configuration space of the I/O interface by performing one or more memory operations using the set of guest space virtual addresses.

4. The apparatus of claim 3, wherein the configuration space stores a task queue;

wherein the one or more hardware processors are further configured to execute the set of instructions to cause the apparatus to:

store, by the device driver, first data to be transmitted to the second hardware interface at a first guest space virtual address of the set of guest space virtual addresses;

store, by the device driver, the first guest space virtual address at the task queue; and

store, by the device driver and based on the updated third mapping table, a notification in the configuration space of the first hardware interface, to cause the first hardware interface to retrieve the first data using the first guest space virtual address and the updated third mapping table.

5. The apparatus of claim 3, wherein the configuration space stores a task queue;

wherein the one or more hardware processors are further configured to execute the set of instructions to cause the apparatus to:

receive, by the device driver from the first hardware interface, a notification that second data is received from a second hardware interface;

responsive to receiving the notification, obtain, by the device driver, a second guest space virtual address associated with the second data from the task queue; and

retrieve, by the device driver using the second guest space virtual address and the third mapping table, the second data.

6. The apparatus of claim 1, wherein the device driver is a VirtI/O Frontend Driver.

7. The apparatus of claim 2, wherein the virtualization controller is a QEMU (Quick Emulator).

8. The apparatus of claim 4, wherein the task queue comprises at least one of: a VRING table or a Virtqueue.

9. A method for operating an input/output (I/O) interface in a virtual machine, the method comprising:

obtaining a first mapping table that maps a set of host space virtual addresses to a first set of physical addresses associated with a first portion of the memory device;

obtaining a second mapping table that maps a second set of physical addresses associated with a second portion of the memory device to the set of host space virtual addresses, wherein the second portion of the memory device is accessible by a virtual machine;

generating a third mapping table that maps the second set of physical addresses to the first set of physical addresses; and

mapping the first portion of the memory device associated with the first set of physical addresses to a configuration space of an I/O interface,

wherein the third mapping table is accessible by a device driver operating in the virtual machine and enables the device driver to directly access the configuration space of the I/O interface to control the I/O interface by performing one or more memory operations to the first portion of the one or more memory devices associated with the first set of physical addresses.

10. The method of claim 9, wherein the mapping of a set of host space virtual addresses to the first set of physical addresses is performed by a virtualization controller; and

wherein the mapping of the first portion of the memory device associated with the first set of physical addresses to the configuration space of the I/O interface is performed by a host driver after receiving an indication from the virtualization controller.

11. The method of claim 9, further comprising:

mapping a set of guest space virtual addresses to the second set of physical addresses; and

updating the third mapping table to further map the set of guest space virtual addresses to the first set of physical addresses; and

wherein the updated third mapping table is accessible by the device driver,

wherein the updated third mapping table enables the device driver to access the

configuration space of the I/O interface by performing one or more memory operations using the set of guest space virtual addresses.

12. The method of claim 11, wherein the configuration space stores a task queue; the method further comprising:

storing, by the device driver, first data to be transmitted to the second hardware interface at a first guest space virtual address of the set of guest space virtual addresses;

storing, by the device driver, the first guest space virtual address at the task queue;

storing, by the device driver and based on the updated third mapping table, a notification in the configuration space of a first hardware interface, to cause the first hardware interface to retrieve the first guest space virtual address and the updated third mapping table.

13. The method of claims 11, wherein the configuration space stores a task queue; the method further comprising:

receiving, by the device driver from the first hardware interface, a notification that second data is received from a second hardware interface;

responsive to receiving the notification, obtaining, by the device driver, a second guest space virtual address associated with the second data from the task queue; and

retrieving, by the device driver using the second guest space virtual address and the third mapping table, the second data.

14. The method of claim 9, wherein the device driver is a VirtI/O Frontend Driver.

15. A non-transitory computer readable medium that stores instructions that are executable by one or more processors of an apparatus to perform a method for operating an input/output (I/O) interface in a virtual machine, the method comprising:

obtaining a first mapping table that maps a set of host space virtual addresses to a first set of physical addresses associated with a first portion of the memory device;

obtaining a second mapping table that maps a second set of physical addresses associated with a second portion of the memory device to the set of host space virtual addresses, wherein the second portion of the memory device is accessible by a virtual machine;

generating a third mapping table that maps the second set of physical addresses to the first set of physical addresses; and

mapping the first portion of the memory device associated with the first set of physical addresses to a configuration space of an I/O interface,

wherein the third mapping table is accessible by a device driver operating in the virtual machine and enables the device driver to directly access the configuration space of the I/O interface to control the I/O interface by performing one or more memory operations to the first portion of the one or more memory devices associated with the first set of physical addresses.

16. The medium of claim 15, wherein the mapping of a set of host space virtual addresses to the first set of physical addresses is performed by a virtualization controller; and

wherein the mapping of the first portion of the memory device associated with the first set of physical addresses to the configuration space of the I/O interface is performed by a host driver after receiving an indication from the virtualization controller.

17. The medium of claim 15, further comprising the set of instructions that is executable by the one or more processors of the apparatus to cause the apparatus to further perform:

mapping a set of guest space virtual addresses to the second set of physical addresses; and

updating the third mapping table to further map the set of guest space virtual addresses to the first set of physical addresses,

wherein the updated third mapping table is accessible by the device driver,

wherein the updated third mapping table enables the device driver to access the configuration space of the I/O interface by performing one or more memory operations using the set of guest space virtual addresses.

18. The medium of claim 17, wherein the configuration space stores a task queue; the method further comprising:

storing, by the device driver, first data to be transmitted to the second hardware interface at a first guest space virtual address of the set of guest space virtual addresses;

storing, by the device driver, the first guest space virtual address at the task queue;

storing, by the device driver and based on the updated third mapping table, a notification in the configuration space of a first hardware interface, to cause the first hardware interface to retrieve the first data using the first guest space virtual address and the updated third mapping table.

19. The medium of claim 17, wherein the configuration space stores a task queue; the method further comprising:

receiving, by the device driver from a first hardware interface, a notification that second data is received from a second hardware interface;

responsive to receiving the notification, obtaining, by the device driver, a second guest space virtual address associated with the second data from the task queue; and

retrieving, by the device driver using the second guest space virtual address and the third mapping table, the second data.

20. The medium of claim 15, wherein the device driver is a VirtI/O Frontend Driver.