Using Intel® Virtualization Technology (Intel® VT) with Intel® QuickAssist Technology

Application Note

April 2023

Document Number: 330689-015

Legal Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more on Intel's Performance Index site .

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

See Intel's Legal Notices and Disclaimers .

Contents

Figures

Tables

Revision History

Document Number	Revision Number	Description	Revision Date
330689	015	Updated: • Section 3.1 – Q: How can I pass through the Intel® QAT PF to a guest?	April 2023
330689	014	Updated: • Legal Notices & Disclaimers • Table 1 – Related Documents • Section 2.1 – Updating the BIOS Settings • Section 2.2 – Installing and Configuring the Host Operating System • Section 2.3 – Installing the Guest OS Image • Section 2.4.1 – Installing Intel® QuickAssist Technology Software on Host • Section 2.4.3 – Pass-through PCI Device • Section 2.4.4 – Installing Intel® QuickAssist Technology Software on Guest • Section 3.1 – Q: How can I pass through the Intel® QAT PF to a guest? • Section 3.2 – Q: What's the QuickAssist (QAT) device ID for my hardware?	February 2023
330689	013	Updated: • Added Section 3.2: What's the Quickassist (QAT) device ID for my hardware?	October 2022
330689	012	Updated: • Section 2.4.1 Added note about certain situations where VFs might report "up" status while only being available to the LKCF but not APIs • Updated content with IntelOne text font	March 2022
330689	011	Updated: • Section 2.2, Installing and Configuring the Host Operating System • Section 2.4.1, Installing Intel® QuickAssist Technology Software on Host • Section 2.4.2, Verifying SR-IOV on Host • Section 2.4.4, Installing Intel® QuickAssist Technology Software on Guest	January 2021
330689	010	Updated Appendix A, FAQ	October 2020
330689	009	Updated relevant resource materials	March 2020
330689	008	Updated information on enabling virtual functions in Intel® QAT	December 2018

Document Number	Revision Number	Description	Revision Date
330689	007	• Added device ID for Intel® Xeon® processor D family. • Made updates to focus on the Intel® QAT v1.7 hardware and software	September 2018
330689	006	• Removed section on using QEMU* KVM command line interface • Added device ID for Intel® C62x Chipset and Intel Atom® C3000 Processor Product Family	August 2017
330689	005	Updated: • Section 1.4.2, Limitations • Section 1.5, Software Requirements • Section 2.2, Installing and Configuring the Host Operating System • Section 2.3, Installing the Guest OS Image • Section 2.4.1, Using the libvirt* Virtual Machine Manager GUI • Section 2.4.3, Pass-through PCI Device • Section 2.4.2.1, Installing Updated QEMU* KVM • Section 2.4.2.2, Pass-through the PCI Device • Section 2.4.2.4, Starting the Guest • Section 2.5, Running Acceleration Services Simultaneously in Host and Guest • Appendix A, FAQ	July 2016
330689	004	Updated: • Section 1.5, Software Requirements • Section 2.4.1, Installing Intel® QuickAssist Technology Software on Host • Section 2.5, Running Acceleration Services Simultaneously in Host and Guest	February 2015
330689	003	Updated: • Section 2.5, Running Acceleration Services Simultaneously in Host and Guest. • Added Appendix A, FAQ	November 2014
330689	002	Updated: • Section 2.5, Running Acceleration Services Simultaneously in Host and Guest.	September 2014

Document Number	Revision Number	Description	Revision Date
330689	001	First "public" version of the document. Based on "Intel confidential" document number 476488-1.4 with the revision history of that document retained for reference purposes. Updated: • Section 1.2 • Removed Fedora 14 information from Section 1.5, Software Requirements and Section 2.4.1, "Using the libvirt* Virtual Machine Manager GUI" on page 10. • Added new step at the end of Section 2.2, "Installing and Configuring the Host Operating System" on page 8. • Updated Section 2.5, Running Acceleration Services Simultaneously in Host and Guest.	July 2014
476488	1.4	Updated: • Modified step 8 in Section2.3, Installing the Guest OS Image. • Added Section 2.5, Running Acceleration Services Simultaneously in Host and Guest.	March 2014
476488	1.3	Updates to make applicable to multiple platforms that use Intel® QuickAssist Technology.	June 2013
476488	1.2	Added new FAQ items, deleted outdated FAQ items.	February 2013
476488	1.1	Added Limitations	October 2012
476488	1.0	Initial release	September 2012

§

1 Introduction

This document discusses the following topics related to using Intel® Virtualization Technology (Intel® VT) with the Intel® QuickAssist Technology (Intel® QAT) Software:

- Features and limitations
- Build and installation

1.1 About this Document

Users of this document are expected to be familiar with virtualization technologies.

In this document, for convenience:

- *Software package* is used as a generic term for the Intel® QuickAssist Technology Software package.
- *Acceleration drivers* is used as a generic term for the software that allows the Intel® QuickAssist Software Library APIs to access the Intel® QuickAssist Accelerator(s) integrated in Intel® QAT.

1.2 Related Documents and References

This section provides references to find current software and documentation.

Associated software and collateral can be found on the open source website: https://01.org/intel-quickassist-technology

The below table includes a list of related documentation.

Table 1. Related Documents

Document Title	Document Number
Intel® QuickAssist Technology API Programmer's Guide	330684
Intel® QuickAssist Technology Cryptographic API Reference Manual	330685
Intel® QuickAssist Technology Data Compression API Reference Manual	330686
Intel® QuickAssist Technology Software for Linux* – Getting Started Guide – Customer Enabling Release	336212
Intel® QuickAssist Technology Software for Linux* – Release Notes – Customer Enabling Release	336211
Intel® QuickAssist Technology Software for Linux* – Programmer's Guide – Customer Enabling Release	336210
Intel® QuickAssist Technology Driver for Linux* – Customer Enabling Release (1.x HW)	NA

Note: Sample configuration files are included with the software package.

1.3 Terminology

The below table includes a list of related documentation.

Table 2. Terminology

Term	Description
CLI	Command Line Interface
GigE	Gigabit Ethernet
GUI	Graphical User Interface
Intel® QAT	Intel® QuickAssist Technology Software
Intel® VT	Intel® Virtualization Technology
IOMMU	Input-Output Memory Management Unit
KVM	Kernel-based Virtual Machine
PCH	Platform Controller Hub
PCI	Peripheral Component Interconnect
SR-IOV	Single-root Input/Output Virtualization
PF	Physical Function
VF	Virtual Function
VM	Virtual Machine

1.4 About Intel® QAT Software

This section lists the features and limitations of the software.

1.4.1 Features

- Peripheral Component Interconnect (PCI) pass-through with Kernel-based Virtual Machine (KVM).
- Single-root Input/Output Virtualization (SR-IOV) with KVM.

1.4.2 Limitations

- SR-IOV may not work on GNU*/Linux* kernel versions v2.6.38 or older.
- KVM limitation: the maximum number of Virtual Functions (VF) that can be mapped to a single VM that is specific to the qemu-kvm version.

1.5 Software Requirements

Software requirements will vary by the particular use case.

Required: Intel® QAT Software for Linux*

Note: Intel® recommends using the same version of the Intel® QAT driver on both host and guest OS. Consult your Intel® representative if you have a requirement to use different versions of the driver.

These instructions were tested against the following Linux* distribution: CentOS*.

1.6 Supported Intel® QAT Endpoints and Their Device IDs

The below table includes a list of related documentation.

Table 3. Supported Intel® QAT Endpoints and Their Device IDs

Intel® QAT Endpoint	Physical Function (PF) Device ID	VF Device ID
8925-8955	0435	0443
Intel® C620 Series Chipsets	37c8	37c9
Intel® Atom® C3000 Processor Product Family	19e2	19e3
Intel® Xeon® processor D family	6f54	6f55

§

2 Using Intel® QAT Software with KVM

Intel® Virtualization Technology can use both SR-IOV and PCI pass-through for the acceleration services. SR-IOV enables the creation of VFs from a single Intel® QAT acceleration device to support acceleration for multiple Virtual Machines (VMs). If you do not need to share a single Platform Controller Hub (PCH) device with accelerator capabilities between multiple VMs, PCI pass-through is sufficient. The following sections describe the steps necessary to enable this functionality, with a focus on the SR-IOV use case.

2.1 Updating the BIOS Settings

Note: The BIOS settings for your system may differ from the following steps:

1. Power on the development board - watch closely for the prompt to enter the BIOS setup. Press **F2** when prompted.
2. Enable the *VT-d* parameter in BIOS - the option may be available under:

   Advanced > System Agent (SA) Configuration > VT-d
3. Enable the *SR-IOV* parameter in BIOS - the option may be available under:

   Advanced > System Agent (SA) Configuration > SRIOV

Note: Enabling the SR-IOV BIOS parameter is not required if you are not using SR-IOV.

4. Press **F4** to Save and Exit - the BIOS changes are saved and the system will boot.

2.2 Installing and Configuring the Host Operating System

1. Install the CentOS* v7 64-bit version. If necessary, consult the Getting Started Guide section "Installing the Operating System" (refer to Table 1), taking note that this guide assumes one of those CentOS* v7 64-bit versions as the host OS when SR-IOV is used.

Note: CentOS* v7 requires the `intel_iommu=on` kernel boot parameter to use SR-IOV and VT-d functionality.

2. Install virtualization related packages using the following command (root privileges required):

```
# yum -y install @virtualization
```

Note: Alternatively, use `yum -y groupinstall Virtualization`. This will install qemu-kvm qemu-img virt-manager, libvirt*, libvirt* -python, python-virtinst libvirt* -client, virt-install, virt-viewer and all other required dependencies.

3. If the libvirtd service is not running, start it by using the commands:

```
# chkconfig libvirtd on
# service libvirtd start
```

4. Verify SR-IOV hardware capabilities using the command:

```
# lspci -vnd 8086:<Device ID>
```

Refer to Section 1.6 for a list of Intel® QAT supported devices and their device IDs.

It should display one of the capabilities as:

```
Capabilities: [140] Single Root I/O Virtualization (SR-IOV)
```

5. Verify BIOS settings using the command:

```
# lsmod | grep kvm
kvm_intel         42122 0
kvm     257132 1 kvm_intel
```

6. Ensure that the system supports VT extensions:

```
# egrep '^flags.*(vmx|svm)' /proc/cpuinfo
```

Note: If nothing is printed out after executing the above command, then the system does not support VT extensions.

7. If kernel boot parameters changed, restart the system:

```
# shutdown –r now
```

8. Power on the system and proceed with the instructions in the following sections.
9. Once the system is restarted, check for DMAR and IOMMU messages, similar to the following:

```
# dmesg | grep –e DMAR –e IOMMU
[       0.000000] ACPI: DMAR 000000007b79c000 00080 (v01 INTEL INTEL ID
00000001 INTL 20091013)
[       0.000000] Intel-IOMMU: enabled
[       0.064454] dmar: IOMMU 0: reg_base_addr fbffc000 ver 1:0 cap
d2078c106f0466 ecap f020df
[   0.065560] IOAPIC id 8 under DRHD base  0xfbffc000 IOMMU 0 [
0.065919] IOAPIC id 9 under DRHD base  0xfbffc000 IOMMU 0 [ 2.168898]
DMAR: No ATSR found
[       2.169358] IOMMU 0 0xfbffc000: using Queued invalidation [
        2.169728] IOMMU: Setting RMRR:
[       2.170091] IOMMU: Setting identity map for device 0000:00:1d.0
[0x7a23f000 - 0x7a241fff]
[       2.170767] IOMMU: Prepare 0-16MiB unity mapping for LPC
[       2.171133] IOMMU: Setting identity map for device 0000:00:1f.0
[0x0 - 0xffffff]
```

Note: If the above command fails, a BIOS update or kernel reconfiguration may be required.

2.3 Installing Guest OS Image

This section describes how to use the libvirt* Virtual Machine Manager Graphical User Interface (GUI) to create the guest OS installation.

Note: The instructions in this section use the GUI approach; information on using the command line interface (CLI) is available at: http://libvirt.org/virshcmdref.html

Using the steps below, enter the root password when prompted:

1. Start the Virtual Machine Manager GUI by selecting it from the top main menu:

Applications > System Tools > Virtual Machine Manager.

2. Open a connection to a Hypervisor by choosing **File > Add Connection**.
3. Choose **QEMU*/KVM** for Hypervisor.
4. Make sure **Connect to remote host** is NOT checked.
5. Make sure **Autoconnect** is checked.
6. Click **Connect**.
7. After a connection is opened, select the **localhost (QEMU*)** and right-click to select **New**.

   Create a new VM with the *New VM* window displayed:

   Enter the **Name** for the Guest VM, selecting **Local install media (ISO image or CDROM)** and clicking **Forward**.

   Select **Use CDROM or DVD**, insert the OS installation CDROM/DVD into the CDROM/DVD drive and make sure that the mounted CDROM appears in box [**Media Unknown (dev/sr0)**]. Select the OS type and version and then click **Forward**.

   Choose **Memory (RAM)** in MB and number of **CPUs** settings (assign a sufficient amount, but it should not affect the Host OS, e.g., for 4 GB RAM and 8 cores, allocate Guest OS < 2 GB RAM and 4 cores CPU). Click **Forward**.

Note: Many platforms will show twice the actual number of cores due to simultaneous multi-threading.

   Make sure **Enable storage for this VM** is checked. Select **Create a disk image on the computer's hard drive** and specify a sufficient amount of hard drive space in GB (20 GB is recommended and at least 18 GB may be required). Make sure **Allocate entire disk now** is checked. Click **Forward**.

   Review the information from Steps 1 through 4. Note the **Ready to begin installation of <Name>** and the **Storage** path to the Guest VM image (this will be used if using the QEMU* CLI). Click **Finish** to begin the installation of the Guest OS.

8. Follow the steps provided in the "Installing the Operating System" section of the appropriate *Getting Started Guide* (refer to Table 1) to install the Guest OS.
9. Shut down the guest OS.

   By default, the guest image is created in the `/var/lib/libvirt/images` directory. This image can be used by libvirt* APIs (virsh tools) and qemu-kvm to run the guest.

2.4 Installing and Configuring Intel® QuickAssist Technology Software

The following sections detail the steps to use the libvirt* Virtual Machine Manager GUI, though similar steps are possible using the command line interface.

2.4.1 Installing Intel® QuickAssist Technology Software on Host

Note: If you are not using SR-IOV and are instead passing through a Physical Function (PF) for acceleration services on one guest only, it is not required to install the Intel® QAT Software package on the host.

Note: The configure script included with the software package will automatically take care of certain build environment details, including copying over the correct sample configuration files. If you are not using an included script to build and install the software, you must perform these operations yourself, using the included script as a guide.

Note: On more recent kernels, it may be required to have the vfio-pci module inserted with "disable_denylist=1". Note that this can be done once with "sudo modprobe vfio-pci disable_denylist=1" or persistently by adding the option "options vfio-pci disable_denylist=1" to /etc/modprobe.d/vfio-pci.conf.

1. Enable the SR-IOV build on the host by using:

```
# ./configure --enable-icp-sriov=host
```

2. Install the QAT software package:

```
# make install
```

3. Restart qat_service:

```
# service qat_service restart
```

4. Enable the QAT VFs:

```
# service qat_service_vfs stop
# service qat_service_vfs start
```

Note: A subset of QAT VFs can be started by altering the qat_service_vfs arguments.

Note: In some situations, QAT VFs can appear "UP" using `# service qat_service_vfs status`, but not accessible for use with applications such as cpa_sample_code or APIs. Upon initial QAT package installation using command `# make install`, QAT VFs will be in an uninstantiated state and unavailable for use. If a targeted QAT VF is started using `# service qat_service_vfs start <qat_vf_device>`, all other QAT VFs will be reported as "UP" as well. However, these remaining QAT VF devices are only available for use by the LKCF (Linux Kernel Crypto Framework) and not available for running APIs. The same situation occurs after QAT VFs are shutdown using `# service qat_service_vfs shutdown` and again, starting a targeted QAT VF using `# service qat_service_vfs start <qat_vf_device>`.

Note: As indicated above, all QAT VFs should be started using `# service qat_service_vfs start`. If a subset of QAT VF devices are being targeted for enablement, a QAT VF **`stop`** command should then be issued using `# service qat_service_vfs stop`, followed by starting the targeted QAT VF or VFs using `# service qat_service_vfs start <qat_vf_device>`. By performing this stop/start sequence, QAT VFs will now be properly reported as either "UP" or "DOWN" indicating availability for API use.

2.4.2 Verifying SR-IOV on Host

Note: If you are not using SR-IOV, skip this section.

Note: Sample configuration files have been included in the software package.

1. **Optional**: View the sample SR-IOV configuration files that were copied to the `/etc` directory. Any software instances that are specified in the PF (non-VF) configuration files will not be created. The sample SR-IOV configuration file sets the number of kernel service instances to **0**.
2. Verify the VFs by running the following command in the host OS. As an example, with one high-end Intel® C620 Series Chipsets in the system, the output would have 16 or more 37c9 devices, as shown below:

```
# lspci -nd 8086:37c9
3d:01.0 0b40: 8086:37c9 (rev 04)
3d:01.1 0b40: 8086:37c9 (rev 04)
3d:01.2 0b40: 8086:37c9 (rev 04)
3d:01.3 0b40: 8086:37c9 (rev 04)
3d:01.4 0b40: 8086:37c9 (rev 04)
3d:01.5 0b40: 8086:37c9 (rev 04)
3d:01.6 0b40: 8086:37c9 (rev 04)
3d:01.7 0b40: 8086:37c9 (rev 04)
3d:02.0 0b40: 8086:37c9 (rev 04)
3d:02.1 0b40: 8086:37c9 (rev 04)
3d:02.2 0b40: 8086:37c9 (rev 04)
3d:02.3 0b40: 8086:37c9 (rev 04)
3d:02.4 0b40: 8086:37c9 (rev 04)
3d:02.5 0b40: 8086:37c9 (rev 04)
3d:02.6 0b40: 8086:37c9 (rev 04)
3d:02.7 0b40: 8086:37c9 (rev 04)
3f:01.0 0b40: 8086:37c9 (rev 04)
3f:01.1 0b40: 8086:37c9 (rev 04)
3f:01.2 0b40: 8086:37c9 (rev 04)
3f:01.3 0b40: 8086:37c9 (rev 04)
3f:01.4 0b40: 8086:37c9 (rev 04)
3f:01.5 0b40: 8086:37c9 (rev 04)
3f:01.6 0b40: 8086:37c9 (rev 04)
3f:01.7 0b40: 8086:37c9 (rev 04)
3f:02.0 0b40: 8086:37c9 (rev 04)
3f:02.1 0b40: 8086:37c9 (rev 04)
3f:02.2 0b40: 8086:37c9 (rev 04)
3f:02.3 0b40: 8086:37c9 (rev 04)
3f:02.4 0b40: 8086:37c9 (rev 04)
3f:02.5 0b40: 8086:37c9 (rev 04)
3f:02.6 0b40: 8086:37c9 (rev 04)
```

```
3f:02.7 0b40: 8086:37c9 (rev 04)
da:01.0 0b40: 8086:37c9 (rev 04)
da:01.1 0b40: 8086:37c9 (rev 04)
da:01.2 0b40: 8086:37c9 (rev 04)
da:01.3 0b40: 8086:37c9 (rev 04)
da:01.4 0b40: 8086:37c9 (rev 04)
da:01.5 0b40: 8086:37c9 (rev 04)
da:01.6 0b40: 8086:37c9 (rev 04)
da:01.7 0b40: 8086:37c9 (rev 04)
da:02.0 0b40: 8086:37c9 (rev 04)
da:02.1 0b40: 8086:37c9 (rev 04)
da:02.2 0b40: 8086:37c9 (rev 04)
da:02.3 0b40: 8086:37c9 (rev 04)
da:02.4 0b40: 8086:37c9 (rev 04)
da:02.5 0b40: 8086:37c9 (rev 04)
da:02.6 0b40: 8086:37c9 (rev 04)
da:02.7 0b40: 8086:37c9 (rev 04)
```

As another example, with one Intel® Communications Chipset 8925 to 8955 Series device in the system, the output would have 32 0443 devices, as shown below:

```
# lspci -nd 8086:0443
bb:01.0 0b40: 8086:0443
bb:01.1 0b40: 8086:0443
bb:01.2 0b40: 8086:0443
bb:01.3 0b40: 8086:0443
bb:01.4 0b40: 8086:0443
bb:01.5 0b40: 8086:0443
bb:01.6 0b40: 8086:0443
bb:01.7 0b40: 8086:0443
bb:02.0 0b40: 8086:0443
bb:02.1 0b40: 8086:0443
bb:02.2 0b40: 8086:0443
bb:02.3 0b40: 8086:0443
bb:02.4 0b40: 8086:0443
bb:02.5 0b40: 8086:0443
bb:02.6 0b40: 8086:0443
bb:02.7 0b40: 8086:0443
bb:03.0 0b40: 8086:0443
```

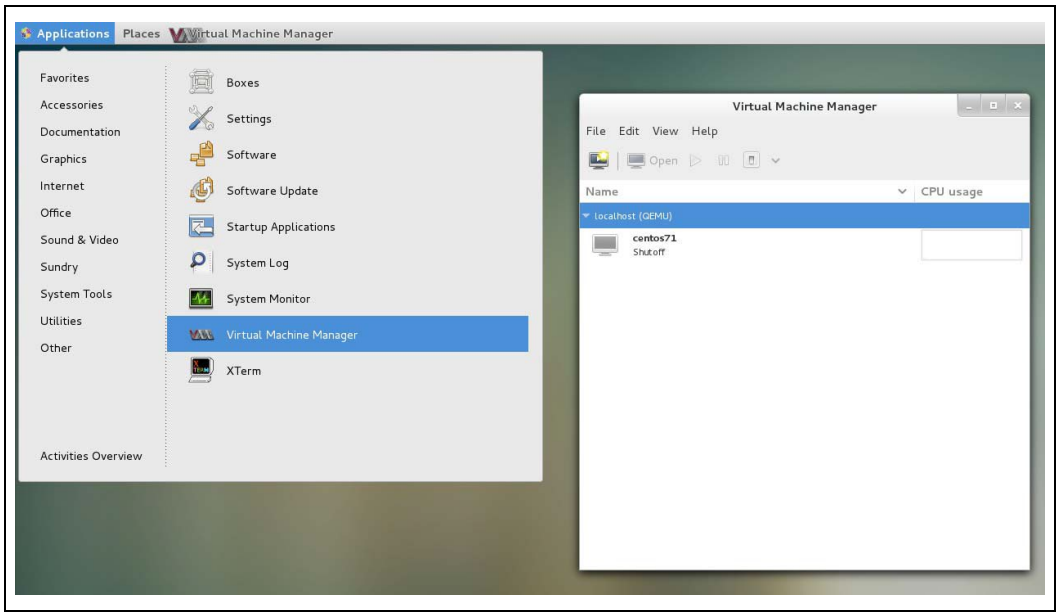
```
bb:03.1 0b40: 8086:0443
bb:03.2 0b40: 8086:0443
bb:03.3 0b40: 8086:0443
bb:03.4 0b40: 8086:0443
bb:03.5 0b40: 8086:0443
bb:03.6 0b40: 8086:0443
bb:03.7 0b40: 8086:0443
bb:04.0 0b40: 8086:0443
bb:04.1 0b40: 8086:0443
bb:04.2 0b40: 8086:0443
bb:04.3 0b40: 8086:0443
bb:04.4 0b40: 8086:0443
bb:04.5 0b40: 8086:0443
bb:04.6 0b40: 8086:0443
bb:04.7 0b40: 8086:0443
```

Refer to Table 3 for supported devices and their device IDs.

2.4.3 Pass-through PCI Device

1. Start Virtual Machine Manager using **Application > System Tools > Virtual Machine Manager**.

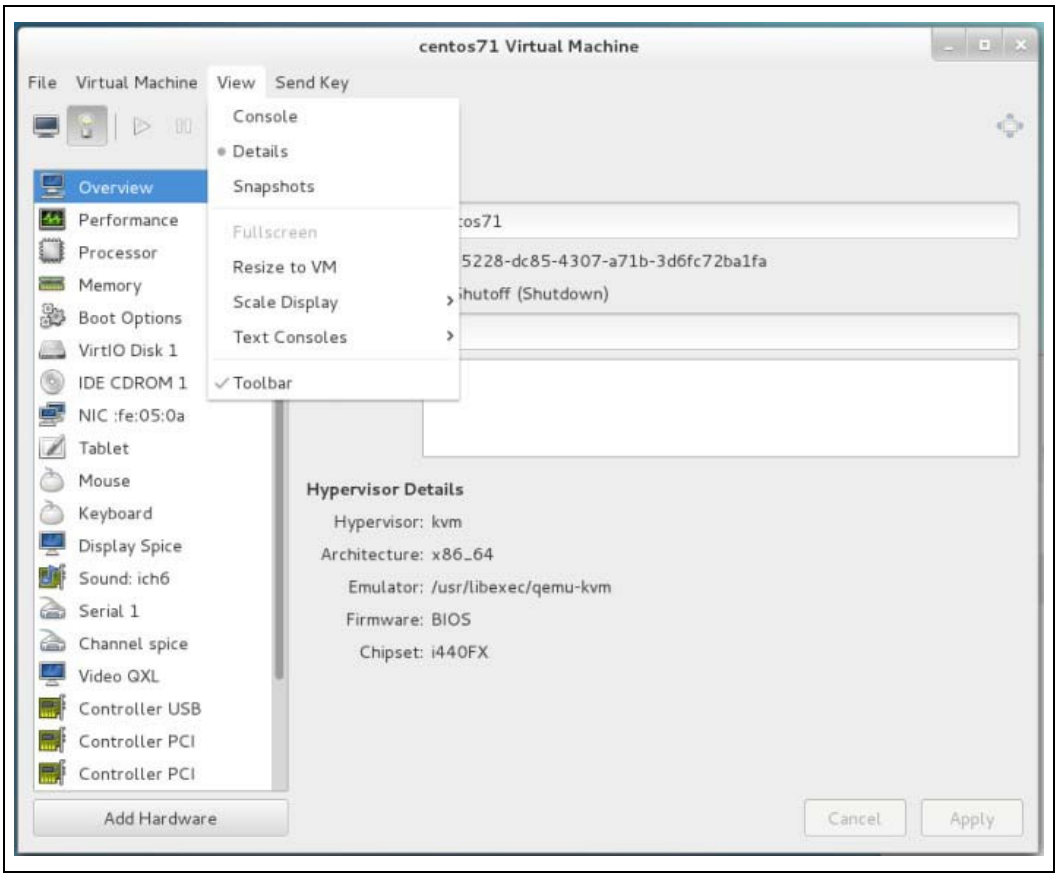
Figure 1. Virtual Machine Manager



2. Right-click on the guest and click **Open** (do not run the guest).

   A new window for the VM is displayed. Go to **View > Details**.

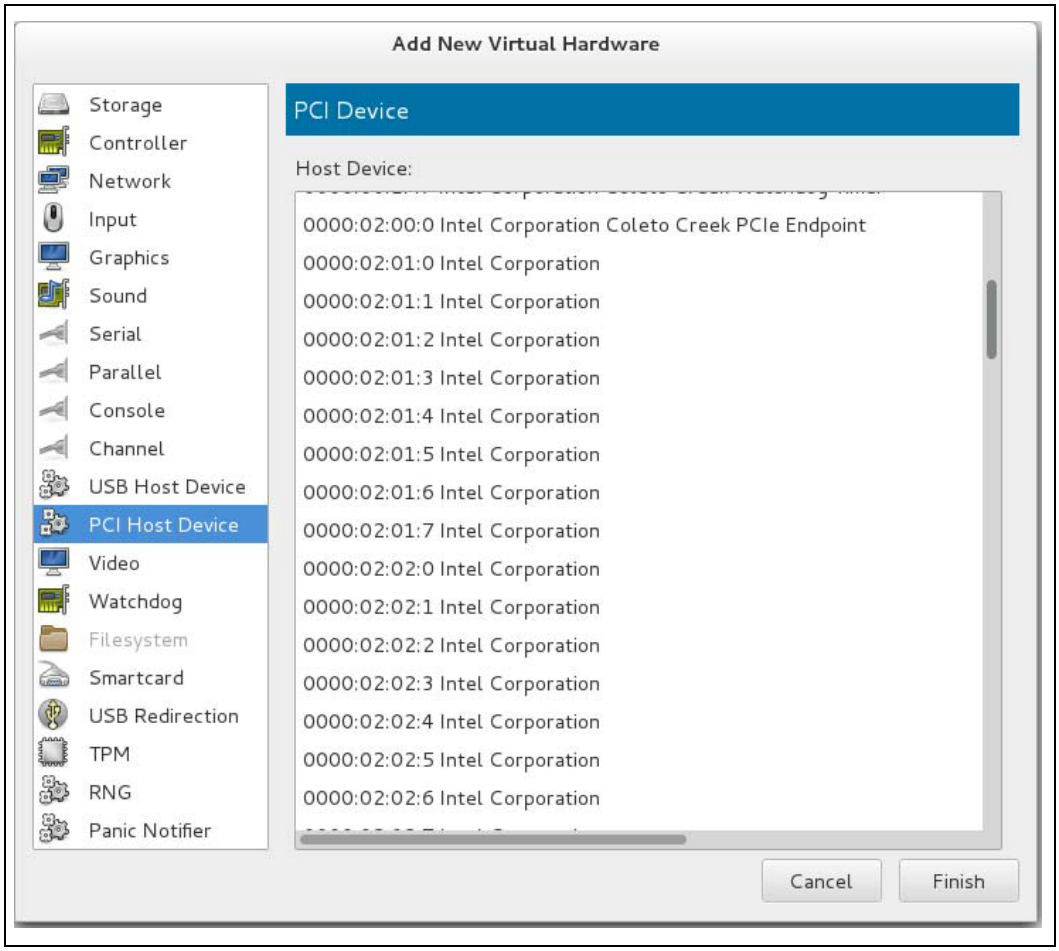
Figure 2. View VM Details



3. Configure the processor, memory, boot options, and virtual hardware for the guest.

4. To add co-processor VFs (refer to Table 3 for supported devices and their device IDs) or GigE ports, select **Add Hardware** in the bottom-left corner and click **PCI Host Device**.

Figure 3. Add New Virtual Hardware



5. Select the appropriate PCI device (for instance, in Figure 3, `02:01:1` is one of the 0443 devices) to attach to Guest and click **Finish**. This newly added device should appear in the left column of details for the Guest.

Note: This action will internally unbind the PCI device from the Host driver currently being used and bind it to vfio-pci (CentOS* v7.1). If using a CLI, a similar sequence is: `virsh-detach <pci_func>` and `virsh-attach <domain> <pci_func>`.

6. **Optional**: To detach a PCI device from the guest, click the PCI device to be detached from the details page left column and click **Remove** (bottom row).

Note: You can add and remove some PCI devices while the guest is running.

7. To run the guest, go to **Virtual Machine > Run** or click **Play Radio** on the Menu bar.
8. To view the guest console, go to **View > Console**.

2.4.4 Installing Intel® QuickAssist Technology Software on Guest

1. In the Guest OS, verify that the appropriate device has been passed through (see Section 2.4.3), as evidenced by the `lspci` command. Refer to Table 3 for the VF device IDs.
2. Install the Intel® QAT Software package on the Guest.
3. Enable the SR-IOV build on the host by using:

   ```
   # ./configure --enable-icp-sriov=guest
   ```

4. Install the QAT software:

   ```
   # make install
   ```

Note: This configuration file supports a limited number of service instances. Specifically, the limitations are a budget of 16 rings per VF. Refer to the relevant *Programmer's Guide* (Table 1) for more information on the configuration file formats. More devices can be passed-through if more service instances are required.

§

3 FAQ

3.1 Q: How can I pass through the Intel® QAT PF to a guest?

Intel® QAT Hardware 1.6 and 1.7 devices are not fully compliant to PCI specs. For this reason, when a Function Level Reset (FLR) is done on the device by a driver different than the Intel® QAT driver (e.g. `vfio-pci`), the value of Maximum Payload Size (MPS) is restored to the reset value, rather than the previous value. This has an impact on full direct pass-through.

Note: This example uses the C62x Chipset and its <device_ID> of is **37c8**, change this for your device ID, referencing What's the Quickassist (QAT) device ID for my hardware? below. Thus, the command referenced would be "`echo 8086 37c8 > /sys/bus/pci/drivers/vfio-pci/new_id`". All instances are **bolded** below.

Follow this procedure to assign a device to a guest using full direct passthrough:

On the host:

1. Ensure `intel_iommu=on`.
2. If using an existing guest VM, verify that the targeted guest VM is not running – if so, perform a shutdown.
3. Do not install the QAT driver – if currently installed, perform the uninstall command:
   ```
   # make uninstall
   ```
4. If the libvirtd service is not running, start it by using the following commands:
   ```
   # chkconfig libvirtd on
   # service libvirtd start
   ```
5. Load the `vfio-pci` driver using:
   ```
   # modprobe vfio-pci
   ```
6. Bind the `vfio` driver to Intel® QAT devices using:
   ```
   # echo 8086 37c8 > /sys/bus/pci/drivers/vfio-pci/new_id
   ```
7. Read the device MPS using:
   ```
   # lspci -vvvnd 8086:37c8 | grep "MaxPayload [1-9]* bytes, Max"
   ```
8. Pass the required parameters and instantiate the guest VM using the following command:
   ```
   # qemu-system-x86_64 -enable-kvm -hda <path to your HD image> -m
   <memory in MB>M -device vfio-pci,host=<BDF of your QAT PF device>
   ```
   For example:
   ```
   # qemu-system-x86_64 -enable-kvm -hda
   /var/lib/libvirt/images/f24.qcow2 -m 2048M -smp
   16,cores=8,threads=1,sockets=2,maxcpus=16 -device vfio-
   pci,host=03:00.0
   ```

Note: On more recent kernels, it may be required to have the vfio-pci module inserted with "disable_denylist=1". Note that this can be done once with "sudo modprobe vfio-pci disable_denylist=1" or persistently by adding the option "options vfio-pci disable_denylist=1" to /etc/modprobe.d/vfio-pci.conf.

Note: The KVM Virtual Machine Manager GUI can also be used to instantiate the guest VM using the QAT PF BDF, if so desired.

9. Set the MPS to its original value. For example, if the MPS in the upstream bridge is equal to 256, enter the command:

```
# setpci -d 8086:37c8 0x7c.b=0x37
```

Note: If the guest is rebooted, the MPS will be changed.

On the guest:

Configure and install QAT driver.

Note: Unlike when using a VF device, QAT is **NOT** configured using the guest SR-IOV flag: `# ./configure --enable-icp-sriov=guest`. Simply use `# ./configure` plus any additionally desired flags.

3.2 Q: What's the QuickAssist (QAT) device ID for my hardware?

Note: This information can also be found in <QAT_package_dir>\quickassist\build_system\build_files\qat_service\.

QAT HW version	Name	PF DID	VF DID
1.5	Intel® Communications Chipset 8900 to 8920 Series	0434	0442
1.5	Intel Atom® Processor C2000 Product Family for Communications Infrastructure	1f18	1f19
1.6	Intel® Communications Chipset 8925 to 8955 Series	0435	0443
1.7	Intel Xeon® Processor D1500	6f54	6f55
1.7	Intel® C62x Chipset	37c8	37c9
1.7	Intel® Xeon® Processor D-2100	37c8	37c9
1.7	Intel Atom® Processor C3000	19e2	19e3
1.8	Intel Atom® Processor P5300	18a0	18a1
1.8	Intel Atom® Processor P5700	18a0	18a1
1.8	Intel Atom® Processor P5900	18a0	18a1
1.7	Intel Xeon® Processor D1700	18ee	18ef

§