NAME

qat - Intel (R) QuickAssist Technology (QAT) driver

SYNOPSIS

To load the driver call:

kldload qat

In order to load the driver on boot add these lines to loader.conf(5) selecting firmware(s) suitable for installed device(s)

qat_200xx_fw_load="YES"
qat_c3xxx_fw_load="YES"
qat_c4xxx_fw_load="YES"
qat_c62x_fw_load="YES"
qat_dh895xcc_fw_load="YES"
qat_4xxx_fw_load="YES"
qat_load="YES"

DESCRIPTION

The **qat** driver supports cryptography and compression acceleration of the Intel (R) QuickAssist Technology (QAT) devices.

The **qat** driver is intended for platforms that contain:

- Intel (R) C62x Chipset
- Intel (R) Atom C3000 processor product family
- Intel (R) QuickAssist Adapter 8960/Intel (R) QuickAssist Adapter 8970 (formerly known as "Lewis Hill")
- Intel (R) Communications Chipset 8925 to 8955 Series
- Intel (R) Atom P5300 processor product family
- Intel (R) QAT 4xxx Series

The **qat** driver supports cryptography and compression acceleration. A complete API for offloading these operations is exposed in the kernel and may be used by any other entity directly. For details of usage and supported operations and algorithms refer to the following documentation available from Intel Download Center **https://downloadcenter.intel.com**:

- Intel (R), *QuickAssist Technology API Programmer's Guide*.
- Intel (R), *QuickAssist Technology Cryptographic API Reference Manual*.
- Intel (R), *QuickAssist Technology Data Compression API Reference Manual*.
- Intel (R), *QuickAssist Technology Performance Optimization Guide*.

In addition to exposing complete kernel API for offloading cryptography and compression operations, the **qat** driver also integrates with crypto(4), allowing offloading supported cryptography operations to Intel (R) QuickAssist Technology (QAT) devices. For details of usage and supported operations and algorithms refer to the documentation mentioned above and *SEE ALSO* section.

SYSCTL_VARIABLES

Following variables may be used to reconfigure the QAT device. For configuration persistence those variables may be set before loading the driver, either via kenv(1) or loader.conf(5). The device specific configuration options are prefixed with *dev.qat.X.* where X is the device number. The specific device needs to be in "down" state before changing the configuration.

state Show current state of the device. Override the device state. Possible values: "down", "up".

NOTE: If the symmetric services are used for device the qat_ocf driver needs to be disabled prior the device reconfiguration. Following variable may be used to enable/disable the QAT cryptographic framework connectivity *dev.qat_ocf.0.enable.* Enabled by default.

cfg_services

Override the device services enabled: symmetric, asymmetric, data compression. Possible values: "sym", "asym", "dc", "sym;dc", "asym;dc", "sym;asym". Default services configured are "sym;asym" for even and "dc" for odd devices.

cfg_mode

Override the device mode configuration for kernel space and user space instances. Possible values: "ks", "us", "ks;us". Default value "ks;us".

num_user_processes

Override the number of uio user space processes that can connect to the QAT device. Default: 2

The following sysctl(8) variables are read-only:

frequency

QAT device frequency value.

mmp_version

QAT MMP Library revision number.

hw_version

QAT hardware revision number.

fw_version
: QAT firmware revision number.

dev_cfg
: Summary of device specific configuration.

heartbeat
: QAT device heartbeat status. Value ’1’ indicates that the device is operational.

heartbeat_failed
: Number of QAT heartbeat failures received.

heartbeat_sent
: Number of QAT heartbeat requests sent.

COMPATIBILITY

The **qat** driver replaced previous implementation introduced in FreeBSD 13.0. Current version, in addition to crypto(4) integration, supports also data compression and exposes a complete API for offloading data compression and cryptography operations.

SEE ALSO

crypto(4), ipsec(4), pci(4), crypto(7), crypto(9)

HISTORY

This **qat** driver was introduced in FreeBSD 14.0. FreeBSD 13.0 included a different version of **qat** driver.

AUTHORS

The **qat** driver was written by Intel (R) Corporation.