QMI CTL 1.11

QMI Control Svc Spec

80-VB816-3 J

July 30, 2013

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Contents

List of Figures

List of Tables

Revision History

Revision	Date	Description
A	April 2006	Initial release.
B	Oct 2007	Updated Table 3-1, added Section 3.4
C	May 2008	Updated Section 3.2.2 to include an optional TLV to QMI_CTL_GET_VERSION_INFO to return addendum information, if any
D	Jul 2009	Updates for this revision include minor version 3. Added a new value to the mandatory TLV in the request and added new request and response TLVs to the QMI_CTL_SET_DATA_FORMAT message.
E	Feb 2010	Updates for this revision include minor version 4. Updated Chapter 3 with new configuration messages related to synchronization and Power Save mode. Added two new appendixes about indication filtering use and synchronization states. Corrected: • Field length for QMI_CTL_GET_CLIENT_ID_RESP (Section 3.3.2) • Data format size for QMI_CTL_SET_DATA_FORMAT_REQ (Section 3.7.1)
F	Jan 2011	Updates for this revision include minor version 5. Updated Sections 3.7.1 to 3.7.3. Added *Qualcomm USB Vendor Features* to Table 1-1 References (Section 1.4).
G	Apr 2012	Updates for this revision include minor version 6. Added the following information to the TLV tables: • Version first introduced • Field type Renamed TLVs: • Uplink TLP Setting renamed to Uplink Data Aggregation Protocol • Configured Uplink TLP Setting renamed to Configured Uplink Data Aggregation Protocol Updated: • Power Save Normal State Report (name change: added Normal) • Section numbering corrected: current Section 3.9.4 was Section 3.10 in Revision F

Revision	Date	Description
H	Apr 2013	Updates for this revision include minor version 7 and minor version 8. Updated version table (Section 2.3.3). Added Result Code TLV: • QMI_CTL_SET_INSTANCE_ID_RESP (Section 3.1.2) • QMI_CTL_SET_DATA_FORMAT_RESP (Section 3.7.2) • QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_RESP (Section 3.10.2) Added version introduced to: • Power Save Normal State Report (Section 3.9.1) • Power Save Descriptor (Section 3.10.1) • Permitted Indication Set (Section 3.10.1) Corrected field types for: • QMUX Service Version List (Section 3.2.2) – service_version_list_len • Addendum Version List (Section 3.2.2) – addendum_label_len – addendum_version_list_len Updated field value in optional TLV: Permitted Indication Set (Section 3.10.1).
J	Jul 2013	Updates for this revision include minor version 9 through minor version 11. Added the State Variables Per Control Point table (Section 2.5.1) Updated Description of QMI_CTL_SET_PWR_SAVE_MODE REQ/RESP (Section 3.11.3) Added the Response Reporting During Power Save TLV (Section 3.12.2) Added new commands: • QMI_CTL_CONFIGURE_RESPONSE_FILTERING_IN_PWR_SAVE (Section 3.13) • QMI_CTL_GET_RESPONSE_FILTERING_SETTING_IN_PWR_SAVE (Section 3.14) • QMI_CTL_SET_SVC_AVAIL_LIST (Section 3.15) • QMI_CTL_GET_SVC_AVAIL_LIST (Section 3.16) • QMI_CTL_SET_EVENT_REPORT (Section 3.17) • QMI_CTL_SVC_AVAIL_IND (Section 3.18) • QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT (Section 3.19)

Note: There is no Rev. I, O, Q, S, X, or Z per Mil. standards.

1 Introduction

1.1 Purpose

This specification documents Major Version 1 of the Qualcomm Messaging Interface (QMI) for the Control Service (QMI_CTL).

QMI_CTL is a QMI service within the QMI framework defined in [Q2]. QMI_CTL messages are transported over the QMUX Control Message Transport Protocol.

QMI_CTL provides the QMUX layer on the Terminal Equipment (TE), e.g., the host driver, commands related to the QMUX link, and client management:

- QMUX link identification
- QMI service version identification
- QMI service client ID allocation and deallocation/revocation

It is expected that the QMI function-level driver and/or QMUX layer implementation on the TE will use QMI_CTL to access this functionality on the MSM® device.

1.2 Scope

This document is intended for QMI control points to exchange information related to QMI services and QMI client management using QMI_CTL on Qualcomm MSM devices.

This document provides the following details about QMI_CTL:

- Theory of operation – Chapter 2 provides the theory of operation of QMI_CTL. The chapter includes messaging conventions, assigned QMI service type, fundamental service concepts, and state variables related to the service.
- Message formats, syntax, and semantics – Chapter 3 provides the specific syntax and semantics of messages included in this version of the QMI_CTL specification.
- Additional information – Appendix A describes QMI indication filtering use cases and Appendix B provides QMI_CTL SYNC diagrams.

1.3 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font. For example, `#include`.

Parameter types are indicated by arrows:

- → Designates an input parameter
- ← Designates an output parameter
- ↔ Designates a parameter used for both input and output

1.4 References

Reference documents are listed in Table 1-1. Reference documents that are no longer applicable are deleted from this table; therefore, reference numbers may not be sequential.

Table 1-1 Reference documents and standards

Ref.	Document	
Qualcomm Technologies		
Q1	*Application Note: Software Glossary for Customers*	CL93-V3077-1
Q2	*Qualcomm MSM Interface (QMI) Architecture*	80-VB816-1
Q3	*Application Note: Qualcomm USB Vendor Features*	80-VK053-1
Standards		
S1	*3rd Generation Partnership Project; Technical Specification Group Terminals; AT Command Set for User Equipment (UE)*	3GPP TS 27.007 (1999)
S2	*Data Service Options for Spread Spectrum Systems: AT Command Processing and the Rm Interface*	3GPP2 C.S0017-003-A
S3	*Data Transmission Systems and Equipment: Extensions to Serial Asynchronous Dialing and Control*	TIA/EIA/IS-131

1.5 Technical Assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies at https://support.cdmatech.com.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

1.6 Acronyms

For definitions of terms and abbreviations, refer to [Q1]. Table 1-2 lists terms that are specific to this document.

Table 1-2 Acronyms

Acronym	Definition
PDU	protocol data unit
QMI	Qualcomm messaging interface
QMUX	QMI Multiplexing Protocol
QOS	quality of service
SDU	service data unit
TE	terminal equipment
TLP	Thin Layer Protocol

2 Theory of Operation

2.1 Generalized QMI Service Compliance

The QMI_CTL service complies with the generalized QMI service specification, including the rules for messages, indications and responses, byte ordering, arbitration, constants, result, and error code values described in [Q2]. Compliance exceptions include:

- The QMI_CTL PDU format differs from the Generalized QMI Service PDU format in that the transaction ID is a single byte in length.
- Multiple QMI_CTL messages (SDUs) cannot be transmitted (bundled) in a single QMUX PDU.

Extensions to the generalized QMI service theory of operation are noted in subsequent sections of this chapter.

2.2 CTL Service Type

CTL is assigned QMI service type 0x00.

2.3 Message Definition Template

2.3.1 Byte Ordering

Numeric values in QMI_CTL messages are encoded in little-endian format.

String values in QMI_CTL messages are sent from the first to the last character (i.e., the same order that is stored in memory for most architectures).

2.3.2 QMI_CTL PDU

QMI_CTL messages consist of a short PDU header that is followed by the QMI_CTL message, as illustrated in Figure 2-1.

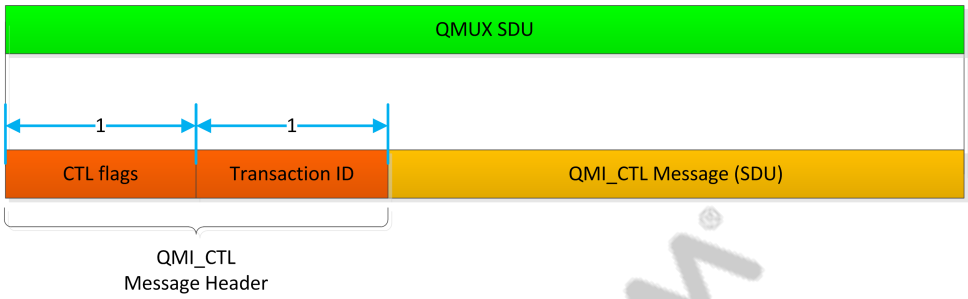

Figure 2-1 QMI_CTL PDU format

Figure 2-1 also illustrates how QMI_CTL messages are carried in a QMUX SDU.

Note: The QMI_CTL PDU must contain only one QMI_CTL message.

Table 2-1 provides details regarding the fields in the QMI_CTL header.

Table 2-1 QMI_CTL header format

Header field	Bit number	Description
Control flags	0 to 1 (least-significant bits)	Type of QMI_CTL message (SDU) following the header. Valid values: • 00 – Request • 01 – Response • 10 – Indication • 11 – Reserved
	2 to 7	Reserved (must be set to 0).
Tx ID	0 to 7	Transaction ID that must be incremented each time the control point issues a new message; used to associate a response with the corresponding request.

The QMI_CTL message (SDU) conforms to the QMI Generalized Service Message (SDU) format described in [Q2].

2.3.3 Response Message Result TLV

This Type-Length-Value (TLV) is present in all Response messages defined in this document. It is not present in the Indication messages.

Name	Version introduced	Version last modified
Result Code	Corresponding response's *Version introduced*	Corresponding response's *Version last modified*

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x02			1	Result Code
Length	4			2	
Value	→	uint16	qmi_result	2	Result code • QMI_RESULT_SUCCESS • QMI_RESULT_FAILURE
		uint16	qmi_error	2	Error code – Possible error code values are described in the error codes section of each message definition

2.4 QMI_CTL Fundamental Concepts

2.4.1 QMI_CTL Control Point

The QMUX endpoints on the TE (e.g., host driver) and the device use QMI_CTL to exchange information related to QMI services and to QMI client management within those services.

Client ID 0x00 is implicitly assigned to the host driver for the purpose of exchanging QMI_CTL messages; therefore, the client_id field of all QMI_CTL messages it sends and receives is 0x00. This is required, since QMI_CTL provides the client ID management function and cannot assign a client ID to itself.

2.4.2 QMI_CTL Service Version

QMI_CTL provides a means to learn the versions of each QMI service supported by the MSM device.

The QMI_CTL control point on the TE (e.g., driver) should first verify that the QMI_CTL service version is compatible before it performs client ID management operations.

2.4.3 QMI Link ID

A QMI-enabled MSM device can support multiple logical QMI connections to a TE. Each is capable of exchanging QMI messages and is referred to as a QMI link.

In cases where the QMI-enabled devices need to be distinguished on the host (e.g., connecting two such devices to the same TE), a QMI link ID can be assigned to each QMI link using primitives provided by this service. It is good practice to do so, in case this scenario arises.

2.4.4 Client ID Management

QMI_CTL defines procedures for assigning unique client IDs to allow other QMI services (besides QMI_CTL) on the MSM device to serve multiple control points. For example, the TE driver can use QMI_CTL to request multiple client IDs from a QMI service on the MSM, and can assign these client IDs to control points on the TE on request.

Unique client IDs enable a resource manager on the MSM to enforce arbitration policies when messages are processed from different QMI control points. When applications request a client ID for any QMI service, the QMI_CTL control point may provide the service version supported by the device. This enables the application to identify the extent of service-specific functionality that is supported.

2.5 Service State Variables

2.5.1 State Variables Per Control Point

Name	Description	Possible values	Default value
report_svc_available	Indicates whether available QMI services are to be reported.	• 0 – Do not report • 1 – Report available QMI services	0

3 QMI_CTL Messages

Table 3-1 QMI_CTL messages

Command	ID	Description
QMI_CTL_SET_INSTANCE_ID	0x0020	Generates a unique ID to distinguish the QMI link over which the message is sent.
QMI_CTL_GET_VERSION_INFO	0x0021	Queries the versions of all QMI services supported by the device.
QMI_CTL_GET_CLIENT_ID	0x0022	Requests a client ID for the specified QMI service type.
QMI_CTL_RELEASE_CLIENT_ID	0x0023	Releases a previously assigned client ID.
QMI_CTL_REVOKE_CLIENT_ID_IND	0x0024	Indicates that a client ID has been revoked by the service.
QMI_CTL_INVALID_CLIENT_ID_IND	0x0025	Indicates that a client ID/service type pair specified in a QMUX header is invalid.
QMI_CTL_SET_DATA_FORMAT	0x0026	Indicates the MSM device of the data format used by the driver.
QMI_CTL_SYNC	0x0027	Synchronizes the service provider and service consumer.
QMI_CTL_SYNC_IND	0x0027	Synchronizes the service provider and service consumer.
QMI_CTL_REG_PWR_SAVE_MODE	0x0028	Sets the Control Power Save mode state reporting conditions for the requesting control point.
QMI_CTL_PWR_SAVE_MODE_IND	0x0028	Indicates a Power Save mode state change update was sent to the control point.
QMI_CTL_CONFIG_PWR_SAVE_SETTINGS	0x0029	Configures the event indication filter by the Power Save state for each QMI service.
QMI_CTL_SET_PWR_SAVE_MODE	0x002A	Sets the QMI framework Power Save mode.
QMI_CTL_GET_PWR_SAVE_MODE	0x002B	Gets the current QMI framework Power Save mode.
QMI_CTL_CONFIGURE_RESPONSE_ FILTERING_IN_PWR_SAVE	0x002C	Configures filtering or reporting of QMI responses for Power Save states.
QMI_CTL_GET_RESPONSE_FILTERING_ SETTING_IN_PWR_SAVE	0x002D	Gets the response reporting settings of all configured power states.

Table 3-1 QMI_CTL messages (cont.)

Command	ID	Description
QMI_CTL_SET_SVC_AVAIL_LIST	0x002E	Configures the list of QMI services to be reported to the control point when the services become available.
QMI_CTL_GET_SVC_AVAIL_LIST	0x002F	Queries the list of QMI services to be reported to the TE when the service is available.
QMI_CTL_SET_EVENT_REPORT	0x0030	Sets the control service reporting conditions for the requesting control point.
QMI_CTL_SVC_AVAIL_IND	0x0031	Indicates that a service in the QMI Service Available list is available.
QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT	0x0032	Configures the event indication filter by the Power Save state for each QMI service.

3.1 QMI_CTL_SET_INSTANCE_ID

Generates a unique ID to distinguish the QMI link over which the message is sent.

CTL message ID

0x0020

Version introduced

Major - 1, Minor - 0

3.1.1 Request - QMI_CTL_SET_INSTANCE_ID_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Host Driver Instance	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Host Driver Instance
Length	1			2	
Value	→	uint8	host_driver_instance	1	Host-unique QMI instance for this device driver.

Optional TLVs

None

3.1.2 Response - QMI_CTL_SET_INSTANCE_ID_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response. This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
QMI Link ID	1.0	1.0
Result Code	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	QMI Link ID
Length	2			2	
Value	→	uint16	qmi_id	2	Unique QMI link ID assigned to the link over which the message is exchanged. The upper byte is assigned by the QMI_CTL service and the lower byte is assigned by the host (the value passed in the request).

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request

3.1.3 Description of QMI_CTL_SET_INSTANCE_ID REQ/RESP

This command facilitates the assignment of a unique QMI link ID to the physical channel carrying the QMUX messages.

The QMI_CTL control point on the TE is required to send this message when there is a need to distinguish QMI links. This occurs in the following interconnection configurations:

- Multiple QMI-enabled MSM devices connected to a single TE
- Multiple TEs connected to a single QMI-enabled MSM device

The returned QMI link ID is the concatenation of the host identifier byte with the device identifier byte.

3.2 QMI_CTL_GET_VERSION_INFO

Queries the versions of all QMI services supported by the device.

CTL message ID

0x0021

Version introduced

Major - 1, Minor - 0

3.2.1 Request - QMI_CTL_GET_VERSION_INFO_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.2.2 Response - QMI_CTL_GET_VERSION_INFO_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response. This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
QMUX Service Version List	1.0	1.0
Result Code	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	QMUX Service Version List
Length	Var			2	
Value	→	uint8	service_version_list_len	1	Number of sets of the following elements: • qmi_svc_type • major_ver • minor_ver
		uint8	qmi_svc_type	1	QMI service type, as defined in [Q2].
		uint16	major_ver	2	Major version number of the QMI service specified by qmi_svc_type.
		uint16	minor_ver	2	Minor version number of the QMI service specified by qmi_svc_type.

Optional TLVs

Name	Version introduced	Version last modified
Addendum Version List	1.2	1.2

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Addendum Version List
Length	Var			2	
Value	→	uint8	addendum_label_len	1	Number of sets of the following elements: • addendum_label
		string	addendum_label	Var	Label describing the addendum.
		uint8	addendum_version_list_len	1	Number of sets of the following elements: • qmi_svc_type • addendum_major_ver • addendum_minor_ver
		uint8	qmi_svc_type	1	QMI service type, as defined in [Q2].
		uint16	addendum_major_ver	2	Addendum major version number of the QMI service specified by qmi_svc_type.
		uint16	addendum_minor_ver	2	Addendum minor version number of the QMI service specified by qmi_svc_type.

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response

3.2.3 Description of QMI_CTL_GET_VERSION_INFO REQ/RESP

This command queries the major and minor version numbers of all QMI services, including QMI_CTL, supported by the QMI-enabled device.

Each QMI service version can be represented as a base version (Addendum version).

Every QMI service has a base version that is represented by major and minor version numbers given by the QMI service version list mandatory TLV, described in Section 3.2.2.

When a QMI service wants to advertise additional functionality supported on top of the base version, the service has an addendum version given by the Addendum Version optional TLV.

The addendum label name is a text string that is a label for the overall QMI addendum. An addendum major and minor version is present for each QMI service that wants to advertise additional functionality. Addendum major and minor versions are returned only for services that have additional functionality to advertise on top of their base version.

3.3 QMI_CTL_GET_CLIENT_ID

Requests a client ID for the specified QMI service type.

CTL message ID

0x0022

Version introduced

Major - 1, Minor - 0

3.3.1 Request - QMI_CTL_GET_CLIENT_ID_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
QMI Service Type	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	QMI Service Type
Length	1			2	
Value	→	uint8	qmi_svc_type	1	QMI service type for which a client ID is requested.

Optional TLVs

None

3.3.2 Response - QMI_CTL_GET_CLIENT_ID_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response. This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
Assigned Client ID	1.0	1.0
Result Code	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Assigned Client ID
Length	2			2	
Value	→	uint8	qmi_svc_type	1	QMI Service type.
		uint8	client_id	1	Client ID.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_CLIENT_IDS_ EXHAUSTED	Maximum number of concurrently assignable client IDs have already been allocated by the service
QMI_ERR_INVALID_SERVICE_TYPE	QMI service type is not supported by the device, or the QMI service does not assign client IDs dynamically

3.3.3 Description of QMI_CTL_GET_CLIENT_ID REQ/RESP

This command obtains a client ID from the specified QMI service. The client ID is assigned by the issuer of this request to a specific control point (application).

The service type cannot be QMI_CTL. There is only a single control point (the QMI driver on the TE) for the QMI_CTL service.

3.4 QMI_CTL_RELEASE_CLIENT_ID

Releases a previously assigned client ID.

CTL message ID

0x0023

Version introduced

Major - 1, Minor - 0

3.4.1 Request - QMI_CTL_RELEASE_CLIENT_ID_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Client ID to Release	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Client ID to Release
Length	2			2	
Value	→	uint8	qmi_svc_type	1	QMI Service type.
		uint8	client_id	1	Client ID.

Optional TLVs

None

3.4.2 Response - QMI_CTL_RELEASE_CLIENT_ID_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response. This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
Released Client ID	1.0	1.0
Result Code	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Released Client ID
Length	2			2	
Value	→	uint8	qmi_svc_type	1	QMI Service type.
		uint8	client_id	1	Client ID.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_CLIENT_ID	Client ID to be released was not allocated by the specified QMI service
QMI_ERR_INVALID_SERVICE_TYPE	QMI service type is not supported by the device, or the QMI service does not assign client IDs dynamically

3.4.3 Description of QMI_CTL_RELEASE_CLIENT_ID REQ/RESP

This command releases a client ID that was previously assigned by the specified QMI service.

3.5 QMI_CTL_REVOKE_CLIENT_ID_IND

Indicates that a client ID has been revoked by the service.

CTL message ID

0x0024

Version introduced

Major - 1, Minor - 0

3.5.1 Indication - QMI_CTL_REVOKE_CLIENT_ID_IND

Message type

Indication

Sender

Service

Indication scope

Unicast (per control point)

Mandatory TLVs

Name	Version introduced	Version last modified
Revoked Client ID	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Revoked Client ID
Length	2			2	
Value	→	uint8	qmi_svc_type	1	QMI Service type.
		uint8	client_id	1	Client ID.

Optional TLVs

None

3.5.2 Description of QMI_CTL_REVOKE_CLIENT_ID_IND

This indication is sent if the service determines that it needs to revoke an assigned client ID.

The receiver should ensure that no further messages are sent using the revoked client ID, unless it has been subsequently reassigned via a QMI_CTL_GET_CLIENT_ID request.

3.6 QMI_CTL_INVALID_CLIENT_ID_IND

Indicates that a client ID/service type pair specified in a QMUX header is invalid.

CTL message ID

0x0025

Version introduced

Major - 1, Minor - 0

3.6.1 Indication - QMI_CTL_INVALID_CLIENT_ID_IND

Message type

Indication

Sender

Service

Indication scope

Unicast (per control point)

Mandatory TLVs

Name	Version introduced	Version last modified
Invalid Client ID	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Invalid Client ID
Length	2			2	
Value	→	uint8	qmi_svc_type	1	QMI Service type.
		uint8	client_id	1	Client ID.

Optional TLVs

None

3.6.2 Description of QMI_CTL_INVALID_CLIENT_ID_IND

This indication may be generated when a QMI service message is received with an invalid client ID specified.

Note: There is no guarantee that this message will be generated. The QMI_CTL service may limit the generation rate of this indication to avoid overflowing the control path with frequent indications such as these.

3.7 QMI_CTL_SET_DATA_FORMAT

Indicates the MSM device of the data format used by the driver.

CTL message ID

0x0026

Version introduced

Major - 1, Minor - 1

3.7.1 Request - QMI_CTL_SET_DATA_FORMAT_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Data Format	Unknown	1.3

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Data Format
Length	1			2	
Value	→	enum8	data_format	1	Data format used by the driver. Values: • 0 – No QOS flow header • 1 – QOS flow header present

Optional TLVs

Name	Version introduced	Version last modified
Underlying Link Layer Protocol	Unknown	1.3
Uplink Data Aggregation Protocol	Unknown	1.6

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Underlying Link Layer Protocol
Length	2			2	

Field	Field value	Field type	Parameter	Size (byte)	Description
Value	→	mask16	link_prot	2	Bitmask of the link protocols supported by the driver. If multiple protocols are supported, they are OR'ed together as a mask. Values: • 0x1 – 802.3 • 0x2 – IP
Type	0x11			1	Uplink Data Aggregation Protocol
Length	1			2	
Value	→	enum8	ul_data_agg_setting	1	Data aggregation protocol to be used for uplink data transfer. Values: • 0x0 – Disable data aggregation • 0x1 – TLP (Thin Layer Protocol)

3.7.2 Response - QMI_CTL_SET_DATA_FORMAT_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.0	1.3

Optional TLVs

Name	Version introduced	Version last modified
Underlying Link Layer Protocol	Unknown	1.3
Configured Uplink Data Aggregation Protocol	Unknown	1.6

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Underlying Link Layer Protocol
Length	2			2	
Value	→	mask16	link_prot	2	Link protocol used by the driver. Only one protocol in the response indicates the mode to be used. Values: • 0x1 – 802.3 • 0x2 – IP

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x11			1	Configured Uplink Data Aggregation Protocol
Length	1			2	
Value	→	enum8	ul_data_agg_setting	1	Data aggregation protocol configured on the device. Values: • 0x0 – Disabled • 0x1 – TLP (Thin Layer Protocol)

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_DATA_FORMAT	Value of the data format was incorrect

3.7.3 Description of QMI_CTL_SET_DATA_FORMAT REQ/RESP

Any change in data format should be performed only when the driver is initializing. The format should not be changed dynamically.

When the data format in the mandatory Data Format TLV is set to 1, the driver must add a 6-byte QOS flow header to the start of the data packet. This header is useful only if you want to access QOS when you are using a device that supports the QMI_QOS service. In the absence of QMI_QOS, the driver should not use this message. Instead, the driver should use the default data format (i.e., the QOS flow header is not present) or should set the data format to 0 if the driver needs to use this message for other TLVs.

If the driver does not receive a response or it receives an error response, the driver should assume that this data format is not supported. The default data format should be used (i.e., send the data packets without the QOS flow header).

Figure 3-1 illustrates the fields of the QOS flow header.

Field	Value
Version	1
Resvd	0 (for future use)
Flow_id	4-byte flow identifier indicating the flow to which the packet belongs; must be set to 0 if the packet belongs to the default (best effort) flow

Figure 3-1 QOS flow header

Note: The QOS flow header is present only in the up (reverse) link direction. It is not present in the down (forward) link direction.

If QOS in not used, the data format in the Mandatory Data Format TLV in the request must be set to a value of 0.

The default underlying link layer protocol is 802.3. To change the protocol to use another mode, such as Raw IP, the underlying link layer protocol optional TLV in the request must be used to specify the modes supported by the driver. The device then chooses the protocol and uses the underlying link layer protocol-optional TLV in the response to indicate the mode which is to be used by the driver.

The default data aggregation protocol setting is disabled by default on the device. To change the setting, the Uplink Data Aggregation Protocol optional TLV must be included in the request. The device then uses the Configured Uplink Data Aggregation Protocol optional TLV in the response to reflect whether the setting took effect.

Note: In the future, this interface will be deprecated and will be replaced by a new API.

3.8 QMI_CTL_SYNC

Synchronizes the service provider and service consumer.

CTL message ID

0x0027

Version introduced

Major - 1, Minor - 4

3.8.1 Request - QMI_CTL_SYNC_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.8.2 Response - QMI_CTL_SYNC_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.4	1.4

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response

3.8.3 Indication - QMI_CTL_SYNC_IND

Message type

Indication

Sender

Service

Indication scope

Unicast (per control point)

Mandatory TLVs

None

Optional TLVs

None

3.8.4 Description of QMI_CTL_SYNC_IND

This indication synchronizes service and control points. The QMI_CTL_SYNC_IND command is only sent at modem bootup, including modem restarts. The indication is delivered using a backoff schedule of 0s, 1s, 2s, 4s, 8s, 16s, 32s, and 64s, after which the service stops sending indications. See Appendix B for state diagrams that explain the behavior of service and control points.

3.9 QMI_CTL_REG_PWR_SAVE_MODE

Sets the Control Power Save mode state reporting conditions for the requesting control point.

CTL message ID

0x0028

Version introduced

Major - 1, Minor - 4

3.9.1 Request - QMI_CTL_REG_PWR_SAVE_MODE_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Power Save Normal State Report	1.4	1.4

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save Normal State Report
Length	1			2	
Value	→	boolean	report_pwrsnormal_state	1	Reports when the Power Save state changes to NORMAL. Values: • 0 – Do not report • 1 – Report when Power Save state changes to NORMAL

Optional TLVs

None

3.9.2 Response - QMI_CTL_REG_PWR_SAVE_MODE_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.4	1.4

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request

3.9.3 Description of QMI_CTL_REG_PWR_SAVE_MODE REQ/RESP

The QMI_CTL_PWR_SAVE_MODE_IND indication is sent on return to NORMAL/Full Power mode if the control point is registered for the Power Save Normal State Report TLV.

3.9.4 Indication - QMI_CTL_PWR_SAVE_MODE_IND

Message type

Indication

Sender

Service

Indication scope

Unicast (per control point)

Mandatory TLVs

None

Optional TLVs

Name	Version introduced	Version last modified
Power Save State Report	1.4	1.4

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Power Save State Report
Length	8			2	
Value	→	uint32	current_state	4	Handle specified by the control point representing the Power Save state; indicates the state after transition.
		uint32	previous_state	4	Handle specified by the control point representing the Power Save state; indicates the state prior to transition.

3.9.5 Description of QMI_CTL_PWR_SAVE_MODE_IND

Interested control points must previously register for the reporting using the QMI_CTL_REG_PWR_SAVE_MODE_REQ message.

3.10 QMI_CTL_CONFIG_PWR_SAVE_SETTINGS

Configures the event indication filter by the Power Save state for each QMI service.

CTL message ID

0x0029

Version introduced

Major - 1, Minor - 4

3.10.1 Request - QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Power Save Descriptor	1.4	1.4

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save Descriptor
Length	5			2	
Value	→	uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_SET_PWR_SAVE_MODE_REQ. Reserved values: 0x00 through 0xFF.
		uint8	qmi_service	1	Valid QMI service identifier.

Optional TLVs

Name	Version introduced	Version last modified
Permitted Indication Set	1.4	1.8

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x11			1	Permitted Indication Set
Length	Var			2	
Value	→	uint16	indication_set	Var	Sequence of indication message identifiers (2 bytes each).

3.10.2 Response - QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.4	1.4

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_ARG	Value exceeds the allowed range

3.10.3 Description of QMI_CTL_CONFIG_PWR_SAVE_SETTINGS REQ/RESP

The control point's Power Save indication filter internal state variables are modified according to the settings specified in the TLVs included in the request message.

The command configures the settings for a specific Power state handle for a specified QMI service. The device allows configuration for up to four power state handles. Any internal or vendor-specific QMI service can be specified for a given Power Save state handle. Predefined handle values are:

- 0x00 – NORMAL – Full-power mode; no indication filtering performed
- 0x01 – SUSPEND – Low-power mode
- 0x02 – POWERDOWN – Low-power mode
- 0x03 – Reserved for future use

Modem event indications to be forwarded to a control point during the specified Power Save descriptor are to be specified in the indication_set field. Any indication not present is filtered and dropped. The absence of the indication_set value dictates all indications are filtered for the power state descriptor.

Note: This command is used only for configuring the device with the allowable indication set for a specific power state. The filter is set only when the specified power state takes effect.

See Appendix A for details about indication filtering.

3.11 QMI_CTL_SET_PWR_SAVE_MODE

Sets the QMI framework Power Save mode.

CTL message ID

0x002A

Version introduced

Major - 1, Minor - 4

3.11.1 Request - QMI_CTL_SET_PWR_SAVE_MODE_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Power Save State	1.4	1.4

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save State
Length	4			2	
Value	→	uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_CONFIG_ PWR_SAVE_SAVINGS.

Optional TLVs

None

3.11.2 Response - QMI_CTL_SET_ PWR_SAVE_MODE_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.4	1.4

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_ARG	Value exceeds the allowed range

3.11.3 Description of QMI_CTL_SET_PWR_SAVE_MODE REQ/RESP

This command sets the QMI framework Power Save mode and is used for direct power state notification applications.

The control point's Power Save indication filter that matches the specified Power Save state is installed for each configured service. Only those indications specified in the Indication Set TLV are forwarded to the control point. Any pending transaction response is discarded upon entering the Power Save state to avoid waking up the client processor. The QMI_CTL_CONFIGURE_RESPONSE_FILTERING_IN_PWR_SAVE_REQ message can override this response suppression behavior while in Power Save mode and enable the control point to receive response messages.

Control points are responsible to ensure all transactions are complete before setting the Power Save mode to avoid loss of pending responses.

If the received Power Save state is not previously configured through QMI_CTL_CONFIG_PWR_SAVE_SETTINGS, it is created with default settings and becomes one of the four device-configurable Power Save handles. When four Power Save handles have been previously configured, the request is rejected and the device continues to operate in the previously set Power Save state.

To return to Full Power mode, the control point must specify the NORMAL predefined handle value described in Section 3.10.3.

See Appendix A for details about indication filtering.

3.12 QMI_CTL_GET_PWR_SAVE_MODE

Gets the current QMI framework Power Save mode.

CTL message ID

0x002B

Version introduced

Major - 1, Minor - 4

3.12.1 Request - QMI_CTL_GET_PWR_SAVE_MODE_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.12.2 Response - QMI_CTL_GET_PWR_SAVE_MODE_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response. This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
Power Save State	1.4	1.4
Result Code	1.4	1.4

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save State
Length	4			2	
Value	→	uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state.

Optional TLVs

This TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
Response Reporting During Power Save	1.9	1.9

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Response Reporting During Power Save
Length	1			2	
Value	→	boolean	response_reporting	1	Indicates whether QMI response messages are reported for the current Power Save state. Values: • 0 – Do not report responses • 1 – Report responses even when Power Save state is not set to NORMAL

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response

3.12.3 Description of QMI_CTL_GET_PWR_SAVE_MODE REQ/RESP

This command queries the control point's current Power Save state. The predefined handle value NORMAL, described in Section 3.10.3, indicates that the Power Save mode has not been set.

See Appendix A for details about indication filtering.

3.13 QMI_CTL_CONFIGURE_RESPONSE_FILTERING_IN_-PWR_SAVE

Configures filtering or reporting of QMI responses for Power Save states.

CTL message ID

0x002C

Version introduced

Major - 1, Minor - 9

3.13.1 Request - QMI_CTL_CONFIGURE_RESPONSE_FILTERING_IN_-PWR_SAVE_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Power Save State Response Reporting	1.9	1.9

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save State Response Reporting
Length	Var			2	
Value	→	uint8	pwr_state_list_len	1	Number of sets of the following elements: • pwrsave_state • response_reporting
		uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_CONFIG_PWR_SAVE_SETTINGS or QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT. Reserved values: 0x00 through 0xFF.

Field	Field value	Field type	Parameter	Size (byte)	Description
		boolean	response_reporting	1	Indicates whether QMI response messages are to be reported when the Power Save state is not set to NORMAL. Values: • 0 – Do not report responses • 1 – Report responses even when Power Save state is not set to NORMAL

Optional TLVs

None

3.13.2 Response - QMI_CTL_CONFIGURE_RESPONSE_FILTERING_IN_-PWR_SAVE_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.9	1.9

Optional TLVs

Name	Version introduced	Version last modified
Power Save State Response Reporting	1.9	1.9

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Power Save State Response Reporting
Length	Var			2	
Value	→	uint8	pwr_state_list_len	1	Number of sets of the following elements: • pwrsave_state • response_reporting

Field	Field value	Field type	Parameter	Size (byte)	Description
		uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_CONFIG_ PWR_SAVE_SETTINGS or QMI_CTL_CONFIG_PWR_SAVE_ SETTINGS_EXT. Reserved values: 0x00 through 0xFF.
		boolean	response_reporting	1	Indicates whether QMI response messages are to be reported when the Power Save state is not set to NORMAL. Values: • 0 – Do not report responses • 1 – Report responses even when Power Save state is not set to NORMAL

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_ARG	Value exceeds the allowed range
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response

3.13.3 Description of QMI_CTL_CONFIGURE_RESPONSE_FILTERING_-IN_PWR_SAVE REQ/RESP

This command configures whether QMI responses are to be filtered for each of the specified Power Save states. This command allows the configuration of this setting for a list of Power Save states.

The default behavior of the modem in Power Save mode is to filter out all the response messages. This command overrides the default behavior and enables the control point to receive response messages. If the Response Reporting During Power Save TLV (in QMI_CTL_GET_PWR_SAVE_MODE) is set to 1, any pending transaction response is sent out regardless of the Power Save state and wakes up the client processor.

If a control point does not send this request to configure the response filtering setting, the device uses the default setting that filters any QMI response message. In this case, the control points are responsible for ensuring that all transactions are complete before configuring the Power Save mode, which avoids loss of pending responses.

If the received Power Save state is not previously configured through QMI_CTL_CONFIG_PWR_SAVE_SETTINGS or QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT, it is created with default settings and becomes one of the four device-configurable Power Save handles.

3.14 QMI_CTL_GET_RESPONSE_FILTERING_SETTING_IN_-PWR_SAVE

Gets the response reporting settings of all configured power states.

CTL message ID

0x002D

Version introduced

Major - 1, Minor - 9

3.14.1 Request - QMI_CTL_GET_RESPONSE_FILTERING_SETTING_IN_-PWR_SAVE_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.14.2 Response - QMI_CTL_GET_RESPONSE_FILTERING_SETTING_-IN_PWR_SAVE_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.9	1.9

Optional TLVs

Name	Version introduced	Version last modified
Power Save State Response Reporting	1.9	1.9

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Power Save State Response Reporting
Length	Var			2	
Value	→	uint8	pwr_state_list_len	1	Number of sets of the following elements: • pwrsave_state • response_reporting
		uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_CONFIG_PWR_SAVE_SETTINGS or QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT. Reserved values: 0x00 through 0xFF.
		boolean	response_reporting	1	Indicates whether QMI response messages are to be reported when the Power Save state is not set to NORMAL. Values: • 0 – Do not report responses • 1 – Report responses even when Power Save state is not set to NORMAL

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INFO_UNAVAILABLE	Power Save Mode information is unavailable

3.14.3 Description of QMI_CTL_GET_RESPONSE_FILTERING_SETTING_IN_PWR_SAVE REQ/RESP

This command queries the settings of the response_reporting field for all power states configured on the device. It returns a list of all Power Save modes that have been configured and whether the modem is allowed to send responses during the corresponding Power Save mode.

3.15 QMI_CTL_SET_SVC_AVAIL_LIST

Configures the list of QMI services to be reported to the control point when the services become available.

CTL message ID

0x002E

Version introduced

Major - 1, Minor - 10

3.15.1 Request - QMI_CTL_SET_SVC_AVAIL_LIST_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
QMI Service Availability List	1.10	1.10

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	QMI Service Availability List
Length	Var			2	
Value	→	uint8	qmi_service_len	1	Number of sets of the following elements: • qmi_service
		uint8	qmi_service	Var	Valid QMI service identifier.

Optional TLVs

None

3.15.2 Response - QMI_CTL_SET_SVC_AVAIL_LIST_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.10	1.10

Optional TLVs

Name	Version introduced	Version last modified
QMI Service Availability List	1.10	1.10

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	QMI Service Availability List
Length	Var			2	
Value	→	uint8	qmi_service_len	1	Number of sets of the following elements: • qmi_service
		uint8	qmi_service	Var	Valid QMI service identifier.

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_ARG	Value exceeds the allowed range
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_ARG_TOO_LONG	Argument passed in a TLV is larger than the available storage in the device

3.15.3 Description of QMI_CTL_SET_SVC_AVAIL_LIST REQ/RESP

This command is used by the control point to configure the list of QMI services to determine the availability of services on the device. This list is overwritten each time a new request is sent by the control point. A maximum of 255 services can be specified in the list.

The control point registers for service availablity notification using QMI_CTL_SET_EVENT_REPORT. This notifies the control point of any service in the configured list that becomes available on the device.

Once the list is configured, the control point receives a QMI_CTL_SVC_AVAIL_IND indication whenever a service in the list boots up. The list of configured services is returned by a QMI_CTL_SET_SVC_AVAIL_LIST_RESP message.

If a QMI_CTL_SET_SVC_AVAIL_LIST_RESP message returns QMI_ERR_NONE, it also returns the list of QMI services that were accepted by the modem. This can be a subset of services provided in a QMI_CTL_SET_SVC_AVAIL_LIST_REQ message if, for example, one of the services requested is not recognized as a valid service by the modem.

3.16 QMI_CTL_GET_SVC_AVAIL_LIST

Queries the list of QMI services to be reported to the TE when the service is available.

CTL message ID

0x002F

Version introduced

Major - 1, Minor - 10

3.16.1 Request - QMI_CTL_GET_SVC_AVAIL_LIST_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.16.2 Response - QMI_CTL_GET_SVC_AVAIL_LIST_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.10	1.10

Optional TLVs

Name	Version introduced	Version last modified
QMI Service Availability List	1.10	1.10

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	QMI Service Availability List
Length	Var			2	
Value	→	uint8	qmi_service_len	1	Number of sets of the following elements: • qmi_service
		uint8	qmi_service	Var	Valid QMI service identifier.

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INFO_UNAVAILABLE	Information is not available

3.16.3 Description of QMI_CTL_GET_SVC_AVAIL_LIST REQ/RESP

This command queries the service available list configured by the control point using QMI_CTL_SET_SVC_AVAIL_LIST. A request sent before the list is configured on the device elicits a QMI_ERR_INFO_UNAVAILABLE error.

3.17 QMI_CTL_SET_EVENT_REPORT

Sets the control service reporting conditions for the requesting control point.

CTL message ID

0x0030

Version introduced

Major - 1, Minor - 10

3.17.1 Request - QMI_CTL_SET_EVENT_REPORT_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

Name	Version introduced	Version last modified
Service Available Reporting	1.10	1.10

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Service Available Reporting
Length	1			2	
Value	→	boolean	report_svc_available	1	Indicates whether available QMI services are to be reported. Values: • 0 – Do not report • 1 – Report available QMI services

3.17.2 Response - QMI_CTL_SET_EVENT_REPORT_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Optional TLVs

Name	Version introduced	Version last modified
Service Available Reporting	1.10	1.10

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Service Available Reporting
Length	1			2	
Value	→	boolean	report_svc_available	1	Indicates whether available QMI services are to be reported. Values: • 0 – Do not report • 1 – Report available QMI services

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request

3.17.3 Description of QMI_CTL_SET_EVENT_REPORT REQ/RESP

The state variables for control point event reporting are modified to reflect the settings indicated in the TLVs present in the request message. The service maintains a separate set of state variables for each control point. See Section 2.5.1 for details about control point state variables.

The Service Available Reporting TLV enables the control point to receive notifications when a service becomes available on the device. Services configured using the QMI_CTL_SET_SVC_AVAIL_LIST command are reported when this TLV is set to 1.

3.18 QMI_CTL_SVC_AVAIL_IND

Indicates that a service in the QMI Service Available list is available.

CTL message ID

0x0031

Version introduced

Major - 1, Minor - 10

3.18.1 Indication - QMI_CTL_SVC_AVAIL_IND

Message type

Indication

Sender

Service

Indication scope

Unicast (per control point)

Mandatory TLVs

None

Optional TLVs

None

3.18.2 Description of QMI_CTL_SVC_AVAIL_IND

This indication is generated when a control point's report_svc_available state variable is set using QMI_CTL_SET_EVENT_REPORT_REQ and a service in the configured Service Available list becomes available on the device. The list is created using QMI_CTL_SET_SVC_AVAIL_LIST.

The control point must have previously used QMI_CTL_SET_EVENT_REPORT_REQ to register for the corresponding state change to be reported.

The QMI_CTL_SVC_AVAIL_IND indication is sent to the control point after the registration using QMI_CTL_SET_EVENT_REPORT is successful.

3.19 QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT

Configures the event indication filter by the Power Save state for each QMI service.

CTL message ID

0x0032

Version introduced

Major - 1, Minor - 11

3.19.1 Request - QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
Power Save Descriptor	1.11	1.11

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Power Save Descriptor
Length	5			2	
Value	→	uint32	pwrsave_state	4	Handle specified by the control point representing the Power Save state. The client-specified value is to match the value used in QMI_CTL_SET_PWR_SAVE_MODE_REQ. Reserved values: 0x00 through 0xFF.
		uint8	qmi_service	1	Valid QMI service identifier.

Optional TLVs

Name	Version introduced	Version last modified
Permitted Indication Set	1.11	1.11

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	Permitted Indication Set
Length	Var			2	

Field	Field value	Field type	Parameter	Size (byte)	Description
Value	→	uint8	indication_set_len	1	Number of sets of the following elements: • indication_set
		uint16	indication_set	Var	Sequence of indication message identifiers (2 bytes each).

3.19.2 Response - QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT_-RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.3) is always present in the response.

Name	Version introduced	Version last modified
Result Code	1.11	1.11

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_MISSING_ARG	One or more required TLVs were missing in the request
QMI_ERR_INVALID_ARG	Value exceeds the allowed range
QMI_ERR_ARG_TOO_LONG	Argument passed in a TLV is larger than the available storage in the device

3.19.3 Description of QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_EXT REQ/RESP

The internal state variables of the control point Power Save indication filter are modified according to the settings specified in the TLVs included in the request message.

The command configures the settings for a specific Power state handle for a specified QMI service. The device allows configuration for up to four power state handles. Any internal or vendor-specific QMI service can be specified for a given Power Save state handle. Predefined handle values are:

- 0x00 – NORMAL – Full-power mode; no indication filtering performed
- 0x01 – SUSPEND – Low-power mode
- 0x02 – POWERDOWN – Low-power mode
- 0x03 – Reserved for future use

Modem event indications to be forwarded to a control point during the specified Power Save descriptor are to be specified in the indication_set field. Any indication not present is filtered and dropped. The absence of an indication_set value dictates that all indications are to be filtered for the power state descriptor.

Note: This command is used only for configuring a device with the allowable indication set for a specific power state. The filter is set only when the specified power state takes effect. This command supports the configuration of up to 255 indications at one time.

See Appendix A for details about indication filtering.

A QMI Indication Filtering Use Cases

The QMI_CTL service has only one control point (i.e., client) per port. QMI_CTL service permits control points to register for indications based on modem processor events. Previously, no differentiation was made between events since they are posted to control points unconditionally following a successful registration.

In multiprocessor environments, the application processor may prioritize an event indication based on its power management state. It is undesirable to wakeup from a Low Power state to process a meaningless notification because doing so needlessly consumes power resources. However, other indications may justify processor wake-up. For example, in a Low Power state, a change in modem RSSI might be ignored, while an incoming call should be processed normally.

The QMI indication filtering mechanism permits the control point to specify the desired modem event notifications based on the application processor power management state. Once enabled, QMI will block any future modem event not listed in the permitted Indication set. There may be multiple power management states, each with their own Indication set. Power management states without a registered Indication set will be assumed to have a complete filter, allowing no events to be posted to the QMI control point.

On return to Normal/Full Power mode, the client may need to query current information to update its modem state.

The use cases for QMI indication filtering are:

- Direct power state notification – The control point explicitly sets the QMI Filtering mode using the QMI_CTL_SET_PWR_SAVE_REQ message, depending on the application processor power state.
- Indirect power state notification – The application processor power management state is conveyed indirectly via a modem processor entity, which sets the QMI Filtering mode.

A.1 Direct Power State Notification

Using direct notification, the application processor control point explicitly informs QMI of the processor Power Save state, thereby installing the previously registered indication filter for all subsequent modem events.

Figure A-1 illustrates the process sequence for direct power state notification.

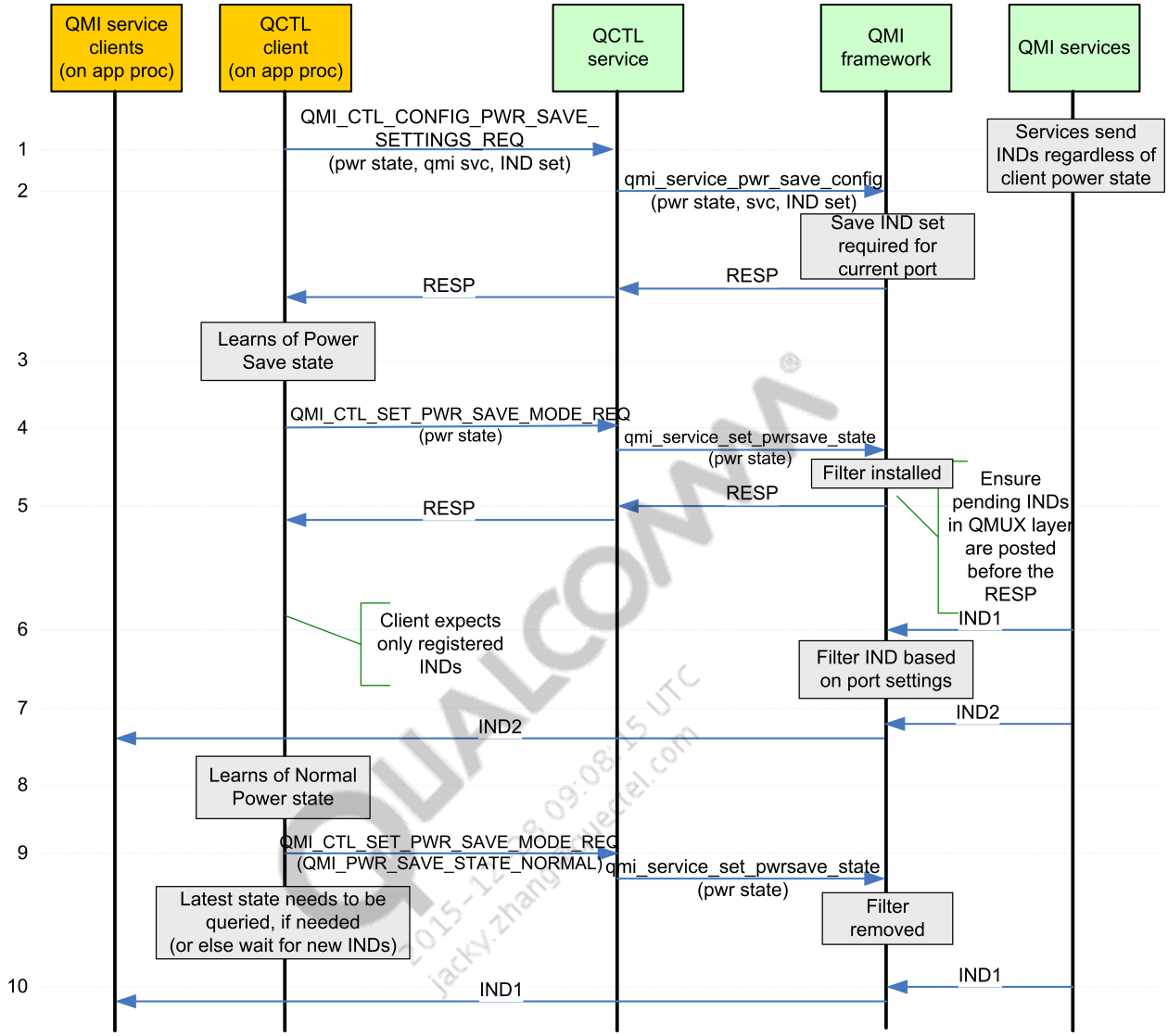

Figure A-1 Direct Power State Notification

The use case is described as follows:

1. The QMI_CTL control point on the application processor registers the Indication set to be filtered for a power state and QMI service combination.
2. The QMI_CTLservice registers a Power state, QMI service, Indication set tuple with the QMI framework. The tuple is stored on a per-port basis such that indication filters are independent for each QMI port.
3. The QMI_CTL control point learns of the power management state change on the application processor. The details of this notification are outside the scope of this document.
4. The QMI_CTL control point sends a QMI_CTL_SET_PWR_SAVE_MODE_REQ message to the QMI_CTL service, indicating the new power state. The message is propagated to the QMI framework, which installs the required indication filter.

5. The QMI framework generates a response for processing the power save start request. After the response is received, the client can expect to receive indications as per the configured filter settings.
6. A QMI service generates a modem event indication (IND1). The QMI framework applies an indication filter for the specific port. For indications that do not match the filter specification, the message is dropped and not propagated to the QMI_CTL control point.
7. A QMI service generates a modem event indication (IND2). The QMI framework applies an indication filter for the specific port. For indications matching the filter specification, the message is propagated to the QMI_CTL control point.
8. The QMI_CTL control point on the application processor is notified of power management state change, entering Normal/Full Power state. The details of this notification are outside the scope of this document.
9. The QMI_CTL control point sends a QMI_CTL_SET_PWR_SAVE_MODE_REQ message to the QMI_CTL service, indicating the new power state. The message is propagated to the QMI framework, which resets the indication filter to the default.
10. A QMI service generates a modem event indication (IND1). The QMI framework bypasses indication filtering, and the message is propagated to the QMI_CTL control point as normal.

A.2 Indirect Power State Notification

Using indirect notification, the application processor power state management is known by an entity on the modem processor, which generates indirect notification of the application processor power management state. The modem entity then informs QMI of the power save state, thereby installing the previously registered indication filter for all subsequent modem events. This use case applies to hardware configurations where the application processor is tightly coupled to the modem processor.

Figure A-2 illustrates the process sequence for indirect power state notification.

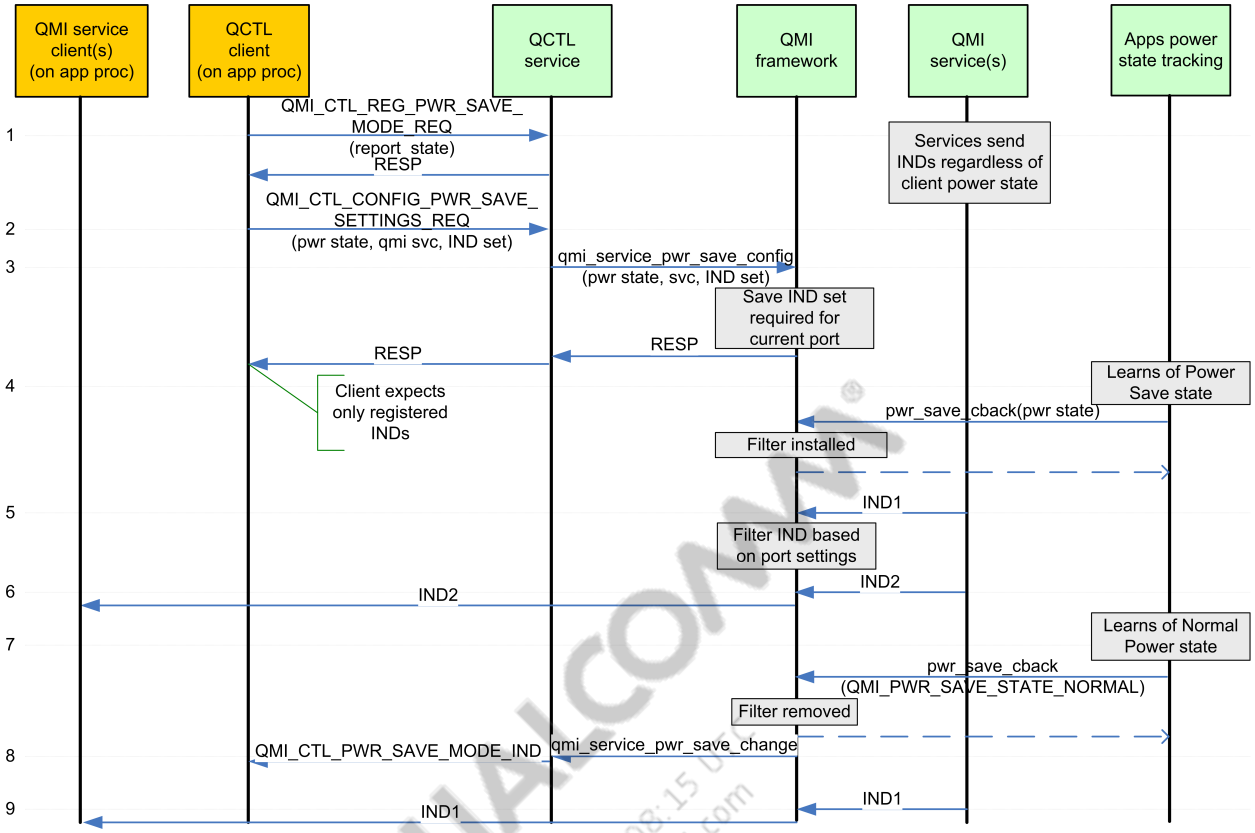

Figure A-2 Indirect Power State Notification

The use case is described as follows:

1. The QMI_CTL control point on the application processor registers for Power Save mode state transition reporting.
2. The QMI_CTL control point on the application processor registers the Indication set to be filtered for a power state and QMI service combination.
3. The QMI_CTL service registers a Power state, QMI service, Indication set tuple with the QMI framework. The tuple is stored on a per-port basis so indication filters are independent for each QMI port.
4. The modem entity is notified of an application processor power management state change, indicating a Conservation Power state. The details of this notification are outside the scope of this document. The modem entity invokes the QMI power save callback to propagate the power state change notification. The QMI framework installs the required indication filter for the power save state on each active port.
5. A QMI service generates a modem event indication (IND1). The QMI framework applies an indication filter for the specific port. For indications not matching the filter specification, the message is dropped and not propagated to the QMI_CTL control point.
6. A QMI service generates a modem event indication (IND2). The QMI framework applies an indication filter for the specific port. For indications matching the filter specification, the message is propagated to the QMI_CTL control point.

7. The modem entity is notified of an application processor power management state change, indicating Full/Normal Power state. The details of this notification are outside the scope of this document. The modem entity invokes the QMI power save callback to propagate the power state change notification. The QMI framework resets the indication filter to the default on each active port.
8. The QMI service generates a QMI_CTL_POWER_SAVE_MODE_IND message to report the Power Save mode state transition. This event is only generated if the control point previously registered for it. The indication is sent only on return to Normal/Full Power mode.
9. A QMI service generates a modem event indication (IND1). The QMI framework bypasses indication filtering, and the message is propagated to the QMI_CTL control point as normal.

A.2.1 Notification Mechanism

QMI publishes a callback function for an external subsystem to provide indirect notification of a power state change on the application processor. The new power state applies to all QMI control points. The function prototype is specified as follows:

```
/*===========================================================================
  FUNCTION QMI_SVC_PWR_SAVE_CBACK
===========================================================================*/

/**
  Notifies the QMI of power management state change. Used to install the
  Modem event indication filter previously registered by the control point.

  PARAMETERS
  @param[ ] qmi_instance  QMI instance.
  @param[ ] new_state     New power save state.

  @return
  None.

  @dependencies
  None.

  @sideeffects
  May change the indication filter installed for the QMI service(s).
*/
extern void qmi_svc_pwr_save_cback
(
  uint8                          qmi_instance,
  qmi_svc_pwr_state_handle_type  new_state
);
```

The new_state parameter can be any allowable power save handle value, as long as it is consistent with the event filters definitions specified using QMI_CTL_CONFIG_PWR_SAVE_SETTINGS_REQ. To return to Full Power mode, use the NORMAL predefined handle value described in Section 3.10.3.

Note: The definition of the callback might change in the future, if necessary. If changes occur, customers will be expected to change any of their code that calls into this callback function.

B QMI_CTL SYNC Diagrams

Figure B-1 illustrates the client state and service state synchronization processes.

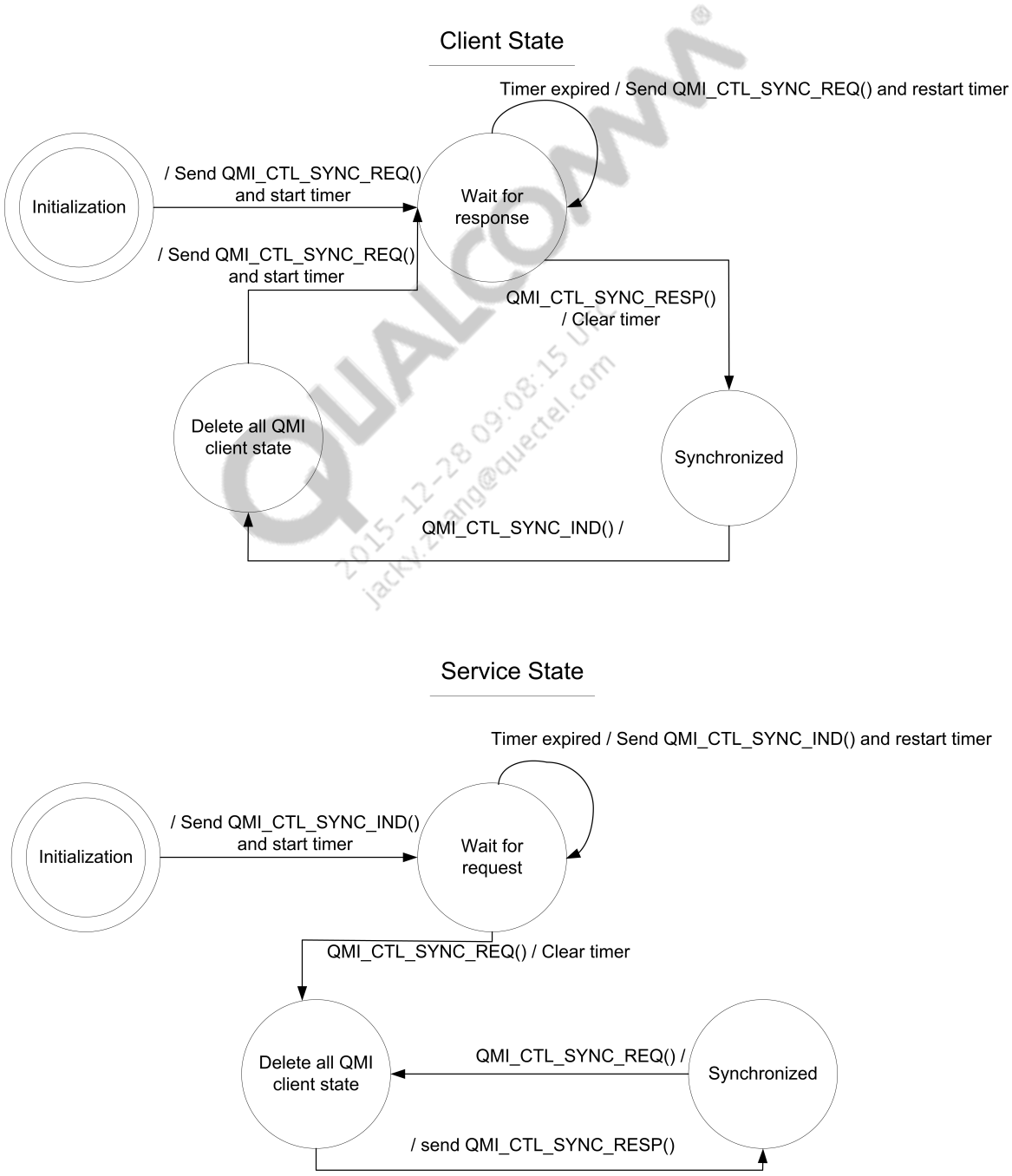

Figure B-1 QMI_CTL SYNC diagrams