BROADCOM CORPORATION
5300 California Avenue
Irvine, CA 92617
Phone: 949-926-5000
Fax: 949-926-5203

Switch MDK Additional API

Version 0.1

Date 03/15/2010

Revision History

Revision	Date	Description
0.1	03/15/2010	Initial document release.

Page intentionally blank

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

This document (including, without limitation, the Broadcom component(s) identified herein) is not designed, intended, or certified for use in any military, nuclear, medical, mass transportation, aviation, navigations, pollution control, hazardous substances management, or other high risk application. BROADCOM PROVIDES THIS DOCUMENT "AS-IS", WITHOUT WARRANTY OF ANY KIND. BROADCOM DISCLAIMS ALL WARRANTIES, EXPRESSED AND IMPLIED, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT.

Table of Contents

1 Purpose

Document additional switch MDK API supported for the integrated switches in BCA products. Please note that this document is a supplement to the MDK documentation for the integrated switches in BCA products.

2 Port Configuration API

2.1 Egress Tag Remapping (BCM681x only)

Get or Set the egress replacement VLAN tag (32-bit tag including TPID), Match VID, and VLAN tag remapping operations.

API and Data Structures

```
/* Set the selected field of port egress replacement tag */
extern int bmd_port_replace_egress_tag_set(
    int unit,
    int port,
    uint32_t tag);

/* Get the port egress replacement tag */
extern int bmd_port_replace_egress_tag_get(
    int unit,
    int port,
    uint32_t *tag);

/* VLAN tag fields */
typedef enum bmd_tag_sel_e {
    bmdVlanTpid,
    bmdVlan8021p,
    bmdVlanCfi,
    bmdVlanVid,
    bmdVlanMatchVid
} bmd_tag_sel_t;

/* Set the egress tag mangling operations and the match vid */
extern int bmd_port_tag_mangle_set(
    int unit,
    int port,
    bmd_tag_sel_t tag_sel,
    int value);

/* Get the egress tag mangling operations and match vid */
extern int bmd_port_tag_mangle_get(
    int unit,
    int port,
    bmd_tag_sel_t tag_sel,
    int *value);
```

Description

The **bmd_port_replace_egress_tag_set/get** API allow configuration/retrieval of egress VLAN tag used for remapping when the VLAN_ID in the egress packet matches the Match-VID that can be configured/retrieved using the bmd_port_tag_mangle_set/get API.

The **bmd_port_tag_mangle_set/get** API allow configuration/retrieval of which parts of the VLAN tag will be remapped with the corresponding parts in egress remapping tag that can be configured/retrieved using the bmd_port_replace_egress_tag_set/get API.

The **bmd_port_tag_mangle_set/get** API also allow configuration/retrieval of the VLAN ID (referred as Match VID) used for matching with the VLAN ID in the packet in order to determine whether remap the VLAN tag or not.

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.2 Port Pause (Flow Control) Configuration API

Configure or Retrieve the flow-control (pause for LAN ports and side-band mechanisms for other ports) capability of a given port.

API and Data Structures

```
/* Pause capability parameters. */
typedef enum bmd_pause_e {
   bmdPauseNone,
   bmdPauseAuto,
   bmdPauseBoth,
   bmdPauseTx,
   bmdPauseRx
} bmd_pause_t;

/* Set the pause capability of a port */
extern int bmd_port_pause_capability_set(
   int unit,
   int port,
   bmd_pause_t value);

/* Get the pause capability of a port */
extern int bmd_port_pause_capability_get(
   int unit,
   int port,
    bmd_pause_t *value);
```

Description

These API allow configuring and retrieving the flow control capability of a given port. Note that the pause mechanism is used for LAN ports. For IMP port, the flow-control mechanism is a side-band signalling mechanism b/w MIPS and Switch DMA. For MoCA port (switch port that is connected to the integrated MoCA core in BCM6816) also, the flow-control mechanism is a side-band signalling mechanism.

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.3 Port Rate Control Configuration API

Configure or Retrieve the rate-control (both ingress and egress) parameters of a given port.

API and Data Structures

```
/* Set the ingress rate limiting parameters of a port */
extern int bmd_port_rate_ingress_set(
    int unit,
    int port,
    uint32_t kbits_sec,
    uint32_t kbits_burst);

/* Get the ingress rate limiting parameters of a port */
extern int bmd_port_rate_ingress_get(
    int unit,
    int port,
    uint32_t *kbits_sec,
    uint32_t *kbits_burst);

/* Set the egress rate limiting parameters of a port */
extern int bmd_port_rate_egress_set(
    int unit,
    int port,
    uint32_t kbits_sec,
    uint32_t kbits_burst);

/* Get the egress rate limiting parameters of a port */
extern int bmd_port_rate_egress_get(
    int unit,
    int port,
    uint32_t *kbits_sec,
    uint32_t *kbits_burst);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.4 Default VLAN 802.1p priority Configuration API

Configure or Retrieve the 802.1p value of default VLAN of a given port.

API and Data Structures

```
/* Set the 802.1p value of port default tag */
extern int bmd_port_vlan_priority_set(
    int unit,
    int port,
    int priority);

/* Get the 802.1p value of port default tag */
extern int bmd_port_vlan_priority_get(
    int unit,
    int port,
    int *priority);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.5 Port Based VLAN (PBVLAN) Configuration API

Configure or Retrieve the PBVLAN port-map of a given port. A given port can forward packets to only the ports in its PBVLAN port-map.

API and Data Structures

```
/* Set the pbvlan map of a port */
extern int bmd_port_pbvlanmap_set(
    int unit,
    int port,
    uint32_t portmap);
```

```
/* Get the pbvlan map of a port */
extern int bmd_port_pbvlanmap_get(
    int unit,
    int port,
    uint32_t *portmap);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.6 Port Traffic Control Configuration API

Configure or Retrieve the traffic control configuration (enable/disable of rx and/or tx) of a given port.

API and Data Structures

```
/* Port Traffic Control Config. */
typedef enum bmd_traffic_ctrl_e {
    bmdNoRxButTx,
    bmdNoTxButRx,
    bmdNoTxAndRx,
    bmdTxAndRx
} bmd_traffic_ctrl_t;

/* Set the traffic control status of a port */
extern int bmd_port_traffic_control_set(
    int unit,
    int port,
    bmd_traffic_ctrl_t traffic_ctrl);

/* Get the traffic control status of a port */
extern int bmd_port_traffic_control_get(
    int unit,
    int port,
    bmd_traffic_ctrl_t *traffic_ctrl);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

2.7 Jumbo Frame Configuration API

Configure or Retrieve the jumbo frame support of a given port.

API and Data Structures

```
/* Set the jumbo frame status of a port */
extern int bmd_port_jumbo_control_set(
   int unit,
   int port,
   int value);

/* Get the jumbo frame status of a port */
extern int bmd_port_jumbo_control_get(
   int unit,
   int port,
   int *value);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

3 Switch Control and Switch Flow Control (Buffer Mgmt) API

The switch control API support per priority and global switch configuration.

3.1 Buffer Management Overview

The switch supports buffer management and flow control in order to efficiently use the limited number of packet buffer resources. The switch also provides the flexibility to manage buffers across its queues based on the priority. For this, the switch supports several per-queue and total thresholds per each priority and a global pause/drop control register. The pause/drop control register allows enabling or disabling of TXQ drop, TXQ pause, Total drop, and Total Pause buffer control mechanisms.

3.1.1 TXQ Drop and TXQ Pause Mechanisms

The following thresholds are provided for each priority in order to support the TXQ drop and TXQ pause buffer control mechanisms.

1. Priority Q Hysteresis Threshold
2. Priority Q Pause Threshold
3. Priority Q Drop Threshold

The TXQ Pause and TXQ Drop mechanisms use the above 3 thresholds and the pause capability of ingress port to decide on whether an ingress packet can be buffered or not.

If the ingress port is Pause capable, an ingress packet will be dropped or paused as given below.

1. Pause if TXQ Pause is enabled and the buffers consumed by the egress TX queue is more than the Priority Q Pause Threshold of the priority of ingress packet. The Pause ON message will be sent from the ingress port to its link partner upon this Pause condition.
2. Start dropping if TXQ Drop is enabled and the buffers consumed by the egress TX queue is more than the Priority Q Drop Threshold of the priority of ingress packet. The buffering will resume only when the buffers consumed by the egress TX queue become less than the Priority Q Hysteresis Threshold of the priority of ingress packet. The Pause OFF message will be sent from the ingress port to its link partner upon this Pause condition.

If the ingress port is not Pause capable, the drop decision is made as given below.

1. Start dropping if TXQ Pause is enabled and the buffers consumed by the egress TX queue is more than the Priority Q Pause Threshold of the priority of ingress packet. The buffering will resume only when the buffers consumed by the egress TX queue become less than the Priority Q Hysteresis Threshold of the priority of ingress packet.
2. Drop if TXQ Drop is enabled and the buffers consumed by the egress TX queue is more than the Priority Q Drop Threshold of the priority of ingress packet.

3.1.2 Total Drop and Total Pause Mechanisms

The following thresholds are provided for each priority in order to support the Total drop and Total Pause buffer control mechanisms.

4. Priority Q Total Hysteresis Threshold
5. Priority Q Total Pause Threshold
6. Priority Q Total Drop Threshold

The Total Pause and Total Drop mechanisms use the above 3 thresholds and the pause capability of ingress port to decide on whether an ingress packet can be buffered or not.

If the ingress port is Pause capable, an ingress packet will be dropped or paused as given below.

3. Pause if Total Pause is enabled and the total buffers consumed is more than the Total Pause Threshold of the priority of ingress packet and the buffers consumed in the egress TXQ is more than the Rx Reserved count (Rx-Base Reserve Register at page 0Ah and offset 52h). The Pause ON message will be sent from the ingress port to its link partner upon this Pause condition.
4. Start dropping (in other words, stop buffering) if Total Drop is enabled and the total buffers consumed is more than the Total Drop Threshold of the priority of ingress packet. The buffering will resume only when the total buffers consumed becomes less than the Total Hysteresis Threshold of the priority of ingress packet. The Pause OFF message will be sent from the ingress port to its link partner upon this Pause condition.

If the ingress port is not Pause capable, the drop decision is made as given below.

3. Start dropping if Total Pause is enabled and the total buffers consumed is more than the Total Pause Threshold of the priority of ingress packet and the buffers consumed in the egress TXQ is more than

the Rx Reserved count. The buffering will resume only when the total buffers consumed becomes less than the Total Hysteresis Threshold of the priority of ingress packet.

4. Drop if Total Drop is enabled the total buffers consumed is more than the Total Drop Threshold of the priority of ingress packet.

3.1.3 Per Priority Queue Low Threshold Buffer Control (6816 B0 only)

The BCM6816B0 also supports a ‘Priority Q Low Drop Threshold’ per priority that allows the switch to use a minimum number of buffers per each priority queue even when the switch is congested (total consumed buffers has exceeded the total threshold). An ingress packet of a given priority will be buffered in a queue even if the total number of consumed buffers has exceeded total threshold of that priority if that queue has not yet consumed the Low threshold (of that priority) number of buffers and there are unallocated buffers. Note that a queue may still not be able to buffer a packet if there are no unallocated buffers that can be used.

3.2 Switch Control API

Get or Set Per queue Hysteresis, Pause, and Drop buffer thresholds and Total Hysteresis, Pause, and Drop buffer thresholds for each priority level.

API and Data Structures

```
/* Switch config parameters. */
typedef enum bmd_switch_control_e {
   bmdSwitchTotalDropThreshold,
   bmdSwitchTotalPauseThreshold,
   bmdSwitchTotalHysteresisThreshold,
   bmdSwitchTxQHiDropThreshold,
   bmdSwitchTxQHiPauseThreshold,
   bmdSwitchTxQHiHysteresisThreshold,
   bmdSwitchTxQLowDropThreshold,
} bmd_switch_control_t;

/* Set a priority queue config parameter. */
extern int bmd_switch_control_priority_set(
   int unit,
   int priority,
   bmd_switch_control_t type,
   int value);

/* Get a priority queue config parameter. */
extern int bmd_switch_control_priority_get(
   int unit,
   int priority,
   bmd_switch_control_t type,
    int *value);
```

Description

Note: The valid value of type is one of the following.

```
    bmdSwitchTotalDropThreshold,
    bmdSwitchTotalPauseThreshold,
    bmdSwitchTotalHysteresisThreshold,
    bmdSwitchTxQHiDropThreshold,
    bmdSwitchTxQHiPauseThreshold,
    bmdSwitchTxQHiHysteresisThreshold,
    bmdSwitchTxQLowDropThreshold,
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

3.3 Global Switch Control

Enable/Disable or retrieve the configuration of

a) Tx Queue Drop mechanism
b) Tx Queue Pause mechanism
c) Total Drop mechanism
d) Total Pause mechanism
e) 802.1Q VLAN

API and Data Structures

```
/* Switch global config parameters. */
typedef enum bmd_switch_control_e {
    bmdSwitchTXQPauseControl,
    bmdSwitchTXQDropControl,
    bmdSwitchTotalPauseControl,
    bmdSwitchTotalDropControl,
    bmdSwitch8021QControl,
} bmd_switch_control_t;

/* Set a global switch config parameter. */
extern int bmd_switch_control_set(
    int unit,
    bmd_switch_control_t type,
    int value);

/* Get a global switch config parameter. */
```

```
extern int bmd_switch_control_get(
    int unit,
    bmd_switch_control_t type,
    int *value);
```

Description

The valid value of type is one of the following.

```
    bmdSwitchTXQPauseControl,
    bmdSwitchTXQDropControl,
    bmdSwitchTotalPauseControl,
    bmdSwitchTotalDropControl,
    bmdSwitch8021QControl,
```

When the type is bmdSwitchTXQPauseControl, the TXQ Pause mechanism can be enabled/disabled or retrieved.

When the type is bmdSwitchTXQDropControl, the TXQ Drop mechanism can be enabled/disabled or retrieved.

When the type is bmdSwitchTotalPauseControl, the Total Pause mechanism can be enabled/disabled or retrieved.

When the type is bmdSwitchTotalDropControl, the Total Pause mechanism can be enabled/disabled or retrieved.

When the type is bmdSwitch8021QControl, the 802.1Q VLAN support can be enabled/disabled or retrieved.

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4 Class of Service Configuration API

The BCM6816 supports the Strict Priority, Weighted Round Robin, and Combined SP + WRR scheduling policies. These CoS API support configuring the scheduling policy and scheduling parameters for the selected policy. These API also provide support for configuring and retrieving the internal priority to egress queue mapping and egress queue to iuDMA channel mapping.

4.1 Multiple CoS queues

Enable or Disable multiple CoS queues (Egress queues). The BCM6816 supports 8 egress queues per port. The BCM6368/6362/6362 support 4 egress queues per port.

API and Data Structures

```
/* Set the number of egress queues */
extern int bmd_cosq_config_set(
    int unit,
    int numq);

/* Get the number of egress queues */
extern int bmd_cosq_config_get(
    int unit,
    int *numq);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.2 QoS Method Configuration API

Configure/Retrieve how the priority is determined for a given packet.

Note1: bmdComboQoS means the config is Port_Qos_En=0 and QoS_Layer_Sel = 10b. This means DiffServ or 802.1p or MAC in the order depending on availability.

Note2: bmdComboHighestQoS means the config is Port_Qos_En=x and QoS_Layer_Sel = 11b. This means highest available priority of Port Based QoS, 802.1p QoS, Diffserv QoS, and MAC based QoS.

API and Data Structures

```
/* QoS type */
typedef enum bmd_cosq_qos_type_e {
    bmdPortQoS,
    bmdMacQoS,
    bmdPrio8021PQoS,
    bmdDiffServQoS,
    bmdTrafficTypeQoS,
    bmdComboQoS,
    bmdComboHighestQoS,
    bmdNoQoS
} bmd_cosq_qos_type_t;

/* Set the QoS method */
extern int bmd_cosq_priority_method_set(
    int unit,
    bmd_cosq_qos_type_t qos_method);

/* Get the QoS method */
extern int bmd_cosq_priority_method_get(
    int unit,
    bmd_cosq_qos_type_t *qos_method);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.3 Configure the Egress Queue Scheduling Policy

Configure/Retrieve egress queue scheduling mechanism and associated parameters.

API and Data Structures

```
/* Egress Queue Scheduling Method */
typedef enum bmd_cosq_sched_e {
    bmdStrictPriority,
    bmdWeightedRoundRobin,
    bmdSpWrrCombo,
    bmdQosNotEnabled
} bmd_cosq_sched_t;

/* Set the queue scheduling parameters */
extern int bmd_cosq_sched_set(
    int unit,
    bmd_cosq_sched_t sched_type,
    int sp_endq);

/* Get the queue scheduling parameters */
extern int bmd_cosq_sched_get(
    int unit,
    bmd_cosq_sched_t *sched_type,
    int *sp_endq);
```

```
/* Set the WRR weight of a queue */
extern int bmd_cosq_wrr_weight_set(
    int unit,
    int egressq,
    int weight);

/* Get the WRR weight of a queue */
extern int bmd_cosq_wrr_weight_get(
    int unit,
    int egressq,
    int *weight);
```

Description

The **bmd_cosq_sched_set/get** API allow configuration/retrieval of egress queue scheduling method.

The **bmd_cosq_wrr_weight_set/get** API allow configuration/retrieval of weights of the queues that are used when scheduling method is configured as Weighted Round Robin.

The sp_endq parameter is used only when the sched_type is bmdSpWrrCombo. It specifies the queue where SP scheduling ends.

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.4 Configure Internal Priority to CoS queue mapping

Get or Set Internal Priority to CoS queue mapping

API and Data Structures

```
/* Set the internal prioirity to egress queue mapping of a port */
extern int bmd_cosq_port_mapping_set(
    int unit,
    int port,
    int priority,
    int egressq);

/* Get the internal prioirity to egress queue mapping of a port */
extern int bmd_cosq_port_mapping_get(
    int unit,
    int port,
    int priority,
    int *egressq);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.5 Egress queue to iuDMA channel mapping

Get or Set MIPS Egress CoS Queue to iuDMA channel mapping.

API and Data Structures

```
/* Set the egress queue to Rx iuDMA channel (Rx on MIPS) mapping */
extern int bmd_cosq_rxchannel_mapping_set(
    int unit,
    int egressq,
    int channel);

/* Get the egress queue to Rx iuDMA channel (Rx on MIPS) mapping */
extern int bmd_cosq_rxchannel_mapping_get(
    int unit,
    int egressq,
    int *channel);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.6 MIPS Tx priority to CoS Queue mapping

Get or Set MIPS Tx packet priority to Egress CoS Queue mapping.

API and Data Structures

```
/* Set the Tx packet priority (Tx from MIPS) to egress queue mapping */
extern int bmd_cosq_txchannel_mapping_set(
    int unit,
    int channel,
    int egressq);

/* Get the Tx Tx packet priority (Tx from MIPS) to egress queue mapping */
```

```
extern int bmd_cosq_txchannel_mapping_get(
    int unit,
    int channel,
    int *egressq);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.7 DSCP to Priority mapping

Get or Set DSCP to priority mapping.

API and Data Structures

```
/* Set the DSCP to priority mapping */
extern int bmd_cosq_dscp_priority_mapping_set(
    int unit,
    int dscp,
    int priority);

/* Get the DSCP to priority mapping */
extern int bmd_cosq_dscp_priority_mapping_get(
    int unit,
    int dscp,
    int *priority);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

4.8 Priority selection for MIPS Tx packets

Get or Set the method used for selecting the priority for MIPS Tx packets.

API and Data Structures

```
/* Egress Queue selection for MIPS Tx packets */
typedef enum bmd_cosq_txqsel_e {
    bmdUseTxBdPrio,
    bmdUseTxDmaChannel,
    bmdUseNone
} bmd_cosq_txqsel_t;

/* Set the method to specify the egress queue for MIPS Tx packets */
extern int bmd_cosq_txq_selection_set(
    int unit,
    bmd_cosq_txqsel_t txq_sel_method);

/* Get the method to specify the egress queue for MIPS Tx packets */
extern int bmd_cosq_txq_selection_get(
    int unit,
    bmd_cosq_txqsel_t *txq_sel_method);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

5 Other API

5.1 Multicast ARL Table Access API

Read/Write an entry from the additional ARL table.

Note: This additional ARL table is avialable only in BCM6362 and BCM63281.

API and Data Structures

```
/* Configure multicast MAC address in multicast ARL table. */
extern int bmd_mcast_mac_addr_set(
    int unit,
    int entry_id,
    int vlan,
    bmd_mac_addr_t *mac_addr,
    int fwd_portmap,
    int priority,
    int valid);

/* Get multicast MAC address. */
extern int bmd_mcast_mac_addr_get(
    int unit,
    int entry_id,
    int *vlan,
    bmd_mac_addr_t *mac_addr,
    int *fwd_portmap,
    int *priority,
    int *used_bit,
    int *valid_bit);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

5.2 Packet Padding Configuration

Configure/retrieve packet padding enable/disable control and pad length.

Note: This API is avialable only for BCM681x products.

API and Data Structures

```
/* Configure switch padding */
extern int bmd_packet_padding_set(
    int unit,
    int pad_ctrl,
    int length);

/* Get the switch padding parameters */
extern int bmd_packet_padding_get(
    int unit,
    int *pad_status,
    int *length);
```

Returns

BCM_E_NONE	Operation completed successfully.
BCM_E_XXX	Operation failed

6 CLI

Below is the CLI (bmd shell commands) available for testing and/or using the various API described above.

6.1 Entering MDK Shell

```
# /etc/mdk
# MDK.0>
MDK.0> help
```

```
Summary of commands:

reset          Reset chip
init           Basic chip initialization
vlan           Manage VLANs
pvlan          Set or get default VLAN for a port
stp            Set or get spanning tree state
portmode       Set or get port mode
pmac           Add or remove port MAC address
cpumac         Add or remove CPU MAC address
stat           Get or clear statistics counters
swinit         Initialize L2 switching
bmd            Show BMD configuration
phy            Read and write PHY registers
qos            Configure the switch QoS
mmac           Add or Remove Multicast ARL table entries
port           Manage Per Port Configuration
switch         Manage Switch global controls
fc             Manage Flow Control
seti           Modify chip register/memory contents (raw)
geti           Get chip register/memory contents (raw)
debug          Configure debug message output
exit           Quit shell
quit           Quit shell
help           Obtain help for shell commands

For more information about a command, enter 'help command-name'
```

6.2 QoS Shell Commands

```
MDK.0> help qos

 SUMMARY:

   Configure the switch QoS

 USAGE:

   qos multiq [enable|disable]
   qos dscpmap <dscp> [<priority>]
   qos portprimap <ports> <prio> [<queue>]
   qos method [port|mac|8021p|diffserv|traffictype|combo|combohigh]
   qos qtodma <queue> [<dmachannel>]
```

```
    qos dmatoq <dmachannel> [<queue>]
    qos sched [strict|wrr|combo [<strict_endq>]]
    qos txqsel [usebd|usedmaq]
    qos wrr <queue> [<weight>]

    Enable or disable multiple queues (QoS)
    qos multiq enable

    Configure QoS method
    qos method port
    qos method diffserv
    qos method

    Configure DSCP to priority mapping
    qos dscpmap 35 6
    qos dscpmap 35

    Configure port priority to egress queue mapping
    qos portprimap 1-3 3 5
    qos portprimap 2-3 3

    Configure egress queue scheduling
    qos sched strict
    qos sched combo 4
    qos sched

    Configure WRR queue weights (< 0x31)
    qos wrr 0 10

    Configure Queue (<=7 for 6816 and <=3 for others) to DMA channel (<= 3) mapping
    qos qtodma 0 2

    Configure DMA channel (<= 3) to Queue (<=7 for 6816 and <=3 for others) mapping
    qos dmatoq 2 7

    Configure the method to determine egress queue for ingress packets on IMP port
    qos txqsel usebd
```

6.3 Switch Flow/Buffer Control Shell Commands

```
MDK.0> help fc

  SUMMARY:
    Manage Flow Control

  USAGE:

    fc txqpause|txqdrop|totdrop|totpause [enable|disable]
    fc threshold txqhyst|txqdrop|txqpause|txqlowdrop|tothyst|totpause|totdrop <queue> [<value>]
    fc

    Enable or diable the flow control mechanisms and configure queue thresholds
    threshold value is from 0 to toal buffers in the switch (6816:0x400,6368:0x200,6362/6328:0x100)
    Number of queues = 8 for 6816 and 4 for 6328/6362/6368
    Examples:

    fc txqpause enable
    fc totdrop
    fc threshold txqhyst 0 0x30
    fc threshold totdrop 0 0x1F0
```

6.4 Switch Global Control Shell Commands

```
MDK.0> help switch

  SUMMARY:

    Manage Switch global controls

  USAGE:

    switch vlan [enable|disable]
    switch padding disable
    switch padding enable <length>

    Enable or disable 802.1Q VLANs
    Enable or disable packet padding at ingress
    Examples:

    switch vlan enable
    switch vlan
    switch padding enable 100
    switch padding
```

6.5 Switch multicast ARL table access Shell Commands

```
MDK.0> help mmac

  SUMMARY:

     Add or Remove Multicast ARL table entries

  USAGE:

     mmac add <entrynum> <vlan> <macaddr> <ports> <pri>
     mmac del <entrynum>
     mmac show <entrynum>

     The mac address should be specified as xx:xx:xx:xx:xx:xx.
     The entrynum is b/w 0 and 15
     Examples:
     mmac add 2 15 01:5E:00:18:10:04 0xA 1
     mamc del 2
     mmac show 2
```

6.6 Switch Port Configuration Shell Commands

```
MDK.0> help port

  SUMMARY:

     Manage Per Port Configuration

  USAGE:

     port jumbo <ports> [enable|disable]
     port pause <ports> [onlyrx|onlytx|both|none]
     port pbvlan <ports> [<portmap>]
     port irc <ports> [<rateinkbps> <burstinkbits>]
     port erc <ports> [<rateinkbps> <burstinkbits>]
     port remaptag <ports> [<vlan_tag>]
     port remapmatchvid <ports> [<vlan>]
     port remaptagop <ports> [tpid|pid|cid|vid] [enable|disable]
     port traffic <ports> [onlyrx|onlytx|both|none]
     port pvlanpri <ports> [<priority>]


     Enable or disable jumbo packets
     port jumbo 0 enable
port jumbo 0
```

Enable or disable flow control

```
port pause 0-3 onlyrx
port pause 0-8
port pause 8 both
```

Configure rate control

```
port irc 0 100  300
port erc 2 50 100
port erc 2
```

Configure pbvlanmap

```
port pbvlan 0-4 0-6
port pbvlan all
```

Configure egress tag replacment (6816 only)

```
port remaptag 0 0x88740024
port remaptag 0
port remapmatchvid 0 0xfff
port remapmatchvid 0
port remaptagop tpid enable
port remaptagop tpid
```

Configure port traffic control

```
port traffic 0-4 onlyrx
port traffic 0-4
```

Configure port default vlan tag priority

```
port pvlanpri 1-3 5
port pvlanpri 1-3
```