FlowStats Feature

CPE-AN202-R

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

April 26, 2013

Revision History

Revision	*Date*	*Change Description*
CPE-AN202-R	04/26/13	**Updated:** • Figure 1: "FlwStats Theory of Operation," on page 5 • Bullets in "FlwStatsQueryInfo_t Structure" on page 6 • flwStats features code in "fc Utility Program" on page 9
CPE-AN201-R	04/15/13	**General:** Added IP version queries. Made query strings follow standard naming practices. **Updated:** • "FlwStatsQueryInfo_t Structure" on page 6 • "fc Utility Program" on page 9
CPE-AN200-R	03/29/13	Initial release

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

Table of Contents

About This Document

Purpose and Audience

FlowStats (flwStats) is a statistics package used to retrieve and aggregate data flow statistics for flows that are accelerated by Broadcom® accelerators (software and hardware).

This document details the following:

- The basic Theory of Operation for the flwStats package.
- The flwStats user API.
- flwStats features in the fc command line utility, which can be used as a debug and test tool for this API and also serves as a sample implementation of the API.

Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

For a comprehensive list of acronyms and other terms used in Broadcom documents, go to: http://www.broadcom.com/press/glossary.php.

Document Conventions

The following conventions may be used in this document:

Convention	*Description*
Bold	User input and actions: for example, type **exit**, click **OK,** press **Alt+C**
`Monospace`	Code: `#include <iostream>` HTML: `<td rowspan = 3>` Command line commands and parameters: `wl [-l] <command>`
< >	Placeholders for *required* elements: enter your <username> or `wl <command>`
[ ]	Indicates *optional* command-line parameters: `wl [-l]` Indicates bit and byte ranges (inclusive): [0:3] or [7:0]

Technical Support

Broadcom provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates through its customer support portal (https://support.broadcom.com). For a CSP account, contact your Sales or Engineering support representative.

In addition, Broadcom provides other product support through its Downloads and Support site (http://www.broadcom.com/support/).

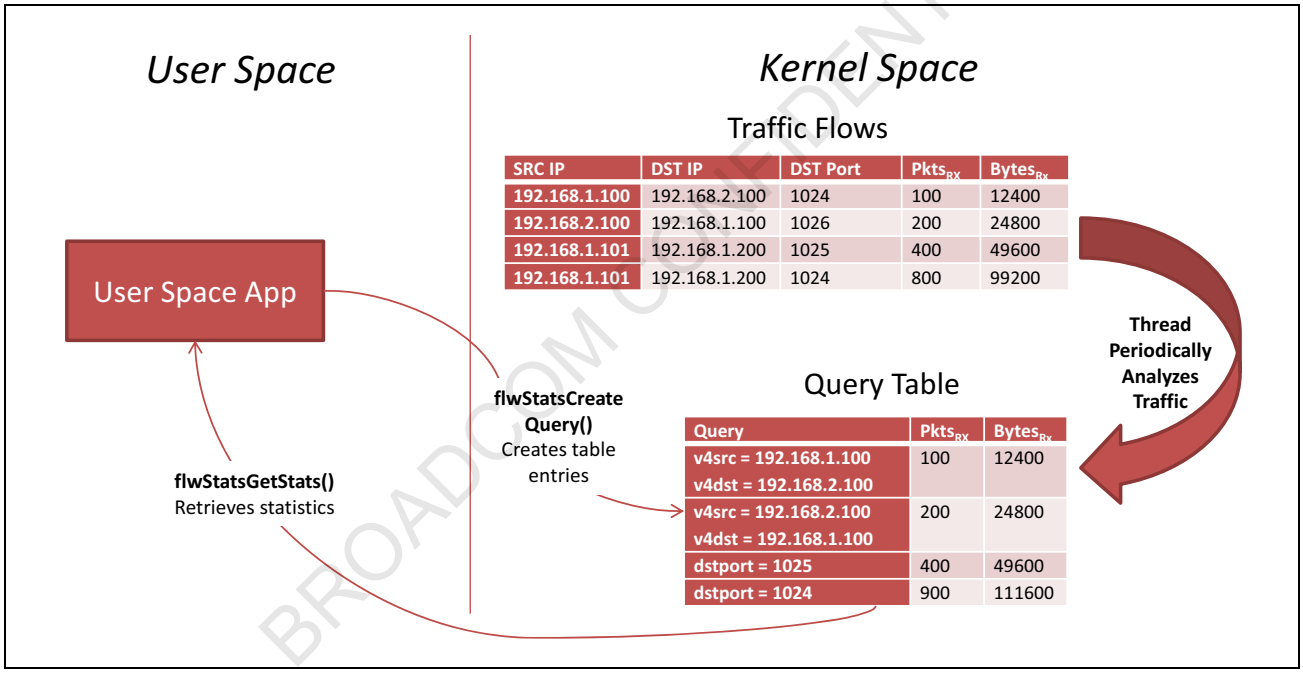
Theory of Operation

For a network comprising two devices on LAN ports with IP addresses of 192.168.1.100 and 192.168.1.101 and a single device on the WAN port with an IP address of 192.168.2.100, the following statistics are important:

- All data flowing from 192.168.1.100 to 192.168.2.100
- All data flowing in the reverse direction
- All data sent from IP port 1024
- All data sent from IP port 1025

These can all be viewed as a series of queries, each specifying one or more parameters on which the flows are periodically searched and the data recorded. A user space application registers these queries via a **fcCtlCreateFlwStatsQuery()** call. Later, it can retrieve the accumulated data via a **fcCtlGetFlwStatsQuery()** call. Figure 1 shows the interaction of the components in our example.

Figure 1: FlwStats Theory of Operation



As can be seen, the statistics data is aggregated periodically except for the first time a query is created. Once a query is created, the query is made active after one complete iteration of stats collection so that the stats are normalized to the next stats collection period. This can be modified periodicity at compile time by changing the macro **BCM_FLWSTATS_GRANULARITY_IN_SEC** to the number of seconds desired for processing to finish a complete cycle of all flows and all queries (the default is 30 seconds). Broadcom recommends keeping the poll granularity at 30 seconds or higher to keep the polling to a minimum. The periodic polling is performed as a low-priority task to minimize any impact to system performance.

Userspace API

The interface between the user space application and the statistics package is defined by a set of library routines and a dedicated data structure.

FlwStatsQueryInfo_t Structure

Input and output from user space to the statistics package uses the **FlwStatsQueryInfo_t** structure as shown below:

```
// FlwStatsQueryInfo_t is used on all APIs
typedef union {
    FlwStatsCreateQueryInfo_t  create;
    FlwStatsGetQueryInfo_t     get;
  FlwStatsClearQueryInfo_t   clear;
    FlwStatsDeleteQueryInfo_t  delete;
}FlwStatsQueryInfo_t;

typedef struct {
    IN  FlwStatsQueryTuple_t  queryTuple;
    OUT uint32_t              handle;
}FlwStatsCreateQueryInfo_t;

typedef struct {
    IN  uint32_t              handle;
    OUT FlwStats_t            flwSt;
}FlwStatsGetQueryInfo_t;

typedef struct {
    IN  uint32_t              handle;
}FlwStatsDeleteQueryInfo_t;

typedef struct {
    IN  uint32_t              handle;
}FlwStatsClearQueryInfo_t;
```


- **FlwStatsQueryTuple_t** —Holds the fields flagging the data for the flows to be used in collecting statistics. An input parameter on **fcCtlCreateFlwStatsQuery()**.
- If the user is interested in collecting stats for flows with single VLAN, “innervid” must be filled in the query tuple.
- l4srcport and l4dstport CANNOT be used to filter for multicast flows. Flowcache tracks the multicast flows by the destination group address. So, when creating flwstats commands, for multicast flows, flwstats users must use v4dst (DST IP address/multicast group address) to uniquely identify the multicast flow. v4src can also be used to track the flwstats based on the multicast source address. Using l4srcport and l4dstport as a filter results in incorrect flow statistics.

```
typedef struct
    FlwStatsQueryIpVer_t srcipver;              /* src ip version */
#define FLWSTATS_QUERYMASK_SRCIPVER  0x00004000 /* src ip version mask */
    FlwStatsQueryIpVer_t dstipver;              /* dst ip version */
#define FLWSTATS_QUERYMASK_DSTIPVER  0x00002000 /* dst ip version mask */
{   uint32_t        srcv4addr;                  /* IPV4  src addr */
#define FLWSTATS_QUERYMASK_SRCV4     0x00001000 /* IPV4  src mask */
    uint32_t        dstv4addr;                  /* IPV4  dst addr */
#define FLWSTATS_QUERYMASK_DSTV4     0x00000800 /* IPV4  dst mask */
    uint32_t        srcv6addr[4];               /* IPV6  src addr */
#define FLWSTATS_QUERYMASK_SRCV6     0x00000400 /* IPV6  src mask */
    uint32_t        dstv6addr[4];               /* IPV6  dst addr */
#define FLWSTATS_QUERYMASK_DSTV6     0x00000200 /* IPV6  dst mask */
    uint16_t        l4srcport;                  /* L4    src port */
#define FLWSTATS_QUERYMASK_L4SRCPRT  0x00000100 /* L4    src mask */
    uint16_t        l4dstport;                  /* L4    dst port */
#define FLWSTATS_QUERYMASK_L4DSTPRT  0x00000080 /* L4    dst mask */
    uint8_t         ipproto;                    /* ip    protocol */
#define FLWSTATS_QUERYMASK_IPPROTO   0x00000040 /* ip protocol mask */
    uint16_t        innervid;                   /* Inner VLAN */
#define FLWSTATS_QUERYMASK_INVID     0x00000020 /* Inner VLAN mask */
    uint16_t        outervid;                   /* Outer VLAN */
#define FLWSTATS_QUERYMASK_OUTVID    0x00000010 /* Outer VLAN mask */
    uint8_t         macSA[ETH_ALEN];            /* Source MAC */
#define FLWSTATS_QUERYMASK_SRCMAC    0x00000008 /* Source MAC mask */
    uint8_t         macDA[ETH_ALEN];            /* Destination MAC */
#define FLWSTATS_QUERYMASK_DSTMAC    0x00000004 /* Destination MAC mask */
    uint8_t         srcphy[FLWSTATS_PHY_LEN];   /* Source Phy e.g. "eth0"*/
#define FLWSTATS_QUERYMASK_SRCPHY    0x00000002 /* Source MAC mask */
    uint8_t         dstphy[FLWSTATS_PHY_LEN];   /* Destination Phy */
#define FLWSTATS_QUERYMASK_DSTPHY    0x00000001 /* Destination MAC mask */
    uint16_t        mask;                       /* Mask */
}FlwStatsQueryTuple_t;
```

  - **Mask** — This field in **FlwStatsQueryTuple_t** requires special attention. It is a bitwise mask indicating which of the above fields should be used for selecting queries, with a set bit indicating the parameter that should be used. For example, 0b000000000100 indicates that only the **Dst MAC** field should be used to for determining flows to match, while 0b101010000000 indicates which **Src IP**, **Src Port**, and **Protocol** should be used.

- **FlwStats_t** — Actual statistics returned from the flowcache. This serves as an output parameter on **fcCtlGetFlwStatsQuery()**.
  - **unsigned long int rxPktCount** — Cumulative count of number of packets for flows matching the query.
  - **unsigned long int rxBytes** — Cumulative count of number of bytes matching the query.
- **uint32_t handle** — An unsigned integer value used to identify a previously created FlowStats query. An output parameter on **fcCtlCreateFlwStatsQuery()** and an input parameter on **fcCtlGetFlwStatsQuery()**, **fcCtlDeleteFlwStatsQuery()**, and **fcCtlClearFlwStatsQuery()**.

FlwStats Library Calls

The **flwStats** API presents the following calls to applications:

- **int fcCtlCreateFlwStatsQuery (FlwStatsQueryInfo_t *pQueryInfo)** — Creates a statistics query for later use. The **FlwStatsCreateQueryInfo_t** field in pQueryInfo union is used by this Create API. A maximum of 128 queries can be created at this time. The max limit can be modified by changing the macro **MAX_FLW_STATS_QUERIES**.
  - **Input** — **FlwStatsQueryTuple_t** structure within **FlwStatsCreateQueryInfo_t** that defines what flows should be included in statistics calculations.
  - **Output** — **handle** field within **FlwStatsCreateQueryInfo_t** that returns a unique ID handle used to identify the new query node in kernel space for later use by **fcCtlGetFlwStatsQuery()**, **fcCtlDeleteFlwStatsQuery()**, or **fcCtlClearFlwStatsQuery()**.
  - Returns zero on success or an error code on failure.
- **int fcCtlGetFlwStatsQuery(FlwStatsQueryInfo_t *pQueryInfo)** — Retrieves statistics for a query previously set up by a call to **fcCtlCreateFlwStatsQuery()**. **FlwStatsGetQueryInfo_t** field in pQueryInfo union is used by this Get API.
  - **Input** — **handle** field within **FlwStatsGetQueryInfo_t** that holds the unique identifier returned by earlier **fcCtlCreateFlwStatsQuery()** call to identify the previously created query. This must identify a specific query.
  - **Output** — **FlwStats_t** structure within **FlwStatsGetQueryInfo_t** that holds the statistics data.
  - Returns zero on success or an error code on failure.
- **int fcCtlDeleteFlwStatsQuery (FlwStatsQueryInfo_t *pQueryInfo)** — Deletes one or more queries previously set up by a call to **fcCtlCreateFlwStatsQuery()**. **FlwStatsDeleteQueryInfo_t** field in pQueryInfo union is used by this Delete API.
  - **Input** — **handle** field within **FlwStatsDeleteQueryInfo_t** that holds the unique identifier returned by earlier **fcCtlCreateFlwStatsQuery()** call to identify the previously created query. The special macro value **ALL_STATS_QUERIES_HANDLE** may be used to delete all queries.
  - **Output** — None.
  - Returns zero on success or an error code on failure.
- **int fcCtlClearFlwStatsQuery (FlwStatsQueryInfo_t *pQueryInfo)** — Zeroes out the counters for one or more queries previously set up by a call to **fcCtlCreateFlwStatsQuery()**. **FlwStatsClearQueryInfo_t** field in pQueryInfo union is used by this Delete API.
  - **Input** — **handle** field within **FlwStatsClearQueryInfo_t** that holds the unique identifier returned by earlier **fcCtlCreateFlwStatsQuery()** call to identify the previously created query. The special macro value **ALL_STATS_QUERIES_HANDLE** may be used to clear the counters for all queries.
  - **Output** — None.
  - Returns zero on success or an error code on failure.

Here are some possible errors associated with the API. This is not a definitive list, and other errors may be found in the **flwstats.h** header file.

- FLWSTAT_ERR_QUERY_NOT_FOUND — Indicates query identified was not found.
- FLWSTAT_ERR_BAD_PARAMS — Indicates an illegal parameter passed into the routine.
- FLWSTAT_ERR_TOO_MANY_QUERIES — Returned by **fcCtlCreateFlwStatsQuery()**. An attempt was made to create more queries than are supported.

fc Utility Program

The **flwStats** parameter of the **fc** command line utility ("**fc flwstats**") can be used to debug and test the FlwStats package. Also, the code itself serves as a sample implementation of the API.

Here is a detailed description of the **flwstats** operations:

- **fc flwstats n <field> <value> {<field> <value>}** — Commands **flwstats** to begin tracking statistics for traffic that matches the indicated parameters. One or more field/value pairs may be used. Here are the valid field keywords and their meanings. Unless otherwise indicated, **<field**> is always a decimal number.
  - **srcv4 <ip4addr>** — Uses the source address to indicate what traffic to track. **<ipv4addr>** must be in IPv4 dotted decimal format.
  - **dstv4 <ip4addr>** — Matches the address against traffic destination address.
  - **srcv6 <ip6addr>** — Matches the IPv6 address to the source of traffic. **<ip6addr>** must be in colon notation (a series of 1 to 8 hexadecimal numbers separated by colons with omitted numbers treated as zeroes). Dotted quad notation is not supported.
  - **dstv6 <ip6addr>** — Matches the IPv6 address to the destination of traffic.
  - **srcport <field>** or **dstport <field>** — Matches the TCP source or destination port against traffic.
  - **proto <field>** — Matches IP protocol field.
  - **invid <field>** — Matches packets using a VLAN with a VID identifier of **<field>**.
  - **outvid <field>** — If a VLAN is nested within another VLAN, this keyword matches the VID of the outer VLAN. **invid** is used instead when a single, unnested VLAN is configured.
  - **srcmac <macAddr>** — Matches packets against the indicated source Ethernet MAC address, expressed as 6 hexadecimal numbers separated by colons.
  - **dstmac <macAddr>** — Matches packets against the indicated destination Ethernet MAC address.
  - **srcphy <phy>** — Matches against the indicated source PHY device, expressed as a string (**eth3**).
  - **dstphy <phy>** — Matches against the indicated destination PHY device.
  - **srcipver <version>** — Matches packets where the IP version is IPV4 if <version> is "v4" or IPV6 if <version> is "v6."
  - **dstipver <version>** — Matches packets where the IP version is IPV4 or IPV6 as indicated.

Upon success, an integer identifier is returned to the console for later use or retrieval (the **<id>** field in the commands below).

- **fc flwstats g <id>** — Gets statistics results for the traffic matching parameters specified by a previous call as indicated by **<id>**.
- **fc flwstats d <id>** — Stops tracking (delete) statistics results for the traffic matching parameters specified by a previous call as indicated by **<id>**. The keyword "**all**" may be used instead of an identifier to stop tracking any statistics.
- **fc flwstats c <id>** — Clears statistics counters results for the traffic matching parameters specified by a previous call as indicated by **<id>**. The keyword "**all**" may be used instead of an identifier to zero out counters for all **flwStats** statistics.
- **fc flwstats p** — Prints debug data.

Below is an example session using the **flwStats** features in **fc** that duplicates the scenario from "Theory of Operation" on page 5".

```
# fcctl flwstats n dstport 1024              User sets up a query…
  New query created.  ID=0
# fcctl flwstats n dstport 1025              …and another one.
  New query created.  ID=1
# fcctl flwstats n dstv4 192.168.2.100 srcv4 192.168.1.100
  New query created.  ID=2                   …and another one.
# fcctl flwstats n srcv4 192.168.2.100 dstv4 192.168.1.100
  New query created.  ID=3                   …and one last one.
#

A series of 124 byte UDP packets are then accelerated through the target board:
 100 packets from 192.168.1.100 to 192.168.2.100 on dst port 1024
 200 packets from 192.168.2.100 to 192.168.1.100 on dst port 1026
 400 packets from 192.168.1.101 to 192.168.2.100 on dst port 1025
 800 packets from 192.168.1.101 to 192.168.2.100 on dst port 1024
#
#
# fcctl flwstats p          User prints statistics results (spacing changed for legibility)
  Query ID: Packets: Bytes: Tuple
  --------:--------:------:----------
         3:     100: 12400: srcv4=192.168.2.100 dstv4=192.168.1.100
         2:     200: 24800: srcv4=192.168.1.100 dstv4=192.168.2.100
         1:     900:111600: dstport=1025
         0:     400: 49600: dstport=1024
#
#
```

Connecting everything®

BROADCOM CORPORATION
5300 California Avenue
Irvine, CA 92617

Phone: 949-926-5000
Fax: 949-926-5203
E-mail: info@broadcom.com
Web: www.broadcom.com