Show commands for “walking’ across the Nexus 7k F2 FSM to troubleshoot a MAC related issues

Commands used to walk FSM points in diagrams below

```
module-4# sh hardware internal statistics pktflow ?
  all      Show packetflow boundary stats from all devices
  dropped  Show packetflow dropped stats from all devices
  queued   Show packetflow queued stats from all devices
  rates    Shows rates stats

Using a command as such:
module-4# sh hardware internal statistics device ?
  all       Show stats of all devices
  arb-mux   Arbiter
  fabric    Fabric
  l2lu      L2 Forwarding
  l3lu      L3 Forwarding
  mac       MAC
  phy       PHY
  qengine   Queuing engine
  rewrite   Rewrite engine
  security  Security
  sensor    Sensor

module-4# sh hardware internal statistics device arb-mux congestion ?
  asic-all       Show congestion stats from all ASIC instances
  asic-instance  Show congestion stats from a single ASIC instances
  port           Show congestion stats from a single port
```

```
module-4# sh hardware internal ?
  access-list     Access Control List
  arb-mux         Show arb-mux information
  clf             Show CLF information
  clipper-klm     Show CLF-klm information
  clm             Show CLM information
  clp_l3          Show CLP_L3 information
  ctsdev          Show information about ctsdev
  dev-log         Device error logging commands
  dev-port-map    Show device port map
  dev-version     Show device versions
  eobcsw          Show EOBC switch related information
  errors          Show hardware internal error information
  flow            Netflow Module
  forwarding      Show hardware information for forwarding Asic
  ipc-channel     Show IPC communication channel monitor internal information
  ltl             Read ltl table entries
  mac             Show mac info
  phy             Shows PHY Driver Information
  plog            Persistent logging commands
  qengine         Show queueing engine info
  rewrite_engine  Show rewrite engine info
  sac             Show SAC information
  sensor          Show sensor commands
  sfp_fpga        Show information about sfp_fpga
  sprom           Show sprom internal information
  statistics      Show hardware internal statistics information
  usdex           EUSD menu
  usdsi           USD:SI menu
  version         Show device versions
  xbar-driver     Show HW info of XBAR
```

```
sh hardware fabric-utilization detail
sh hardware queuing drops ingress
sh har clipper queuing drop
sh hardware internal statistics devi l2lu  congestion
sh hard internal stat pktflow rates
sh hardware internal statistics devi l2lu pktflow
sh hard internal stat pktflow dropped
sh hardware internal statistics devi l2lu  errors
sh hard internal eobcsw stats – for sup

sh hardware internal mac gd port
sh fabricpath unicast routes vdc 2
sh hardware internal mac event-history - info -  errors
sh hardware internal arb-mux event-histor errors
```

For tcam

```
sh fib internal error statistics
```

For DRAP

```
sh u2fib internal event-history events
sh u2fib internal info all
```

Using the built in Python interpreter for some future DEVOPS type functions.
Some quick python scripts to run on switches using the CheckPortDiscards() function and the cisco.cli() function.

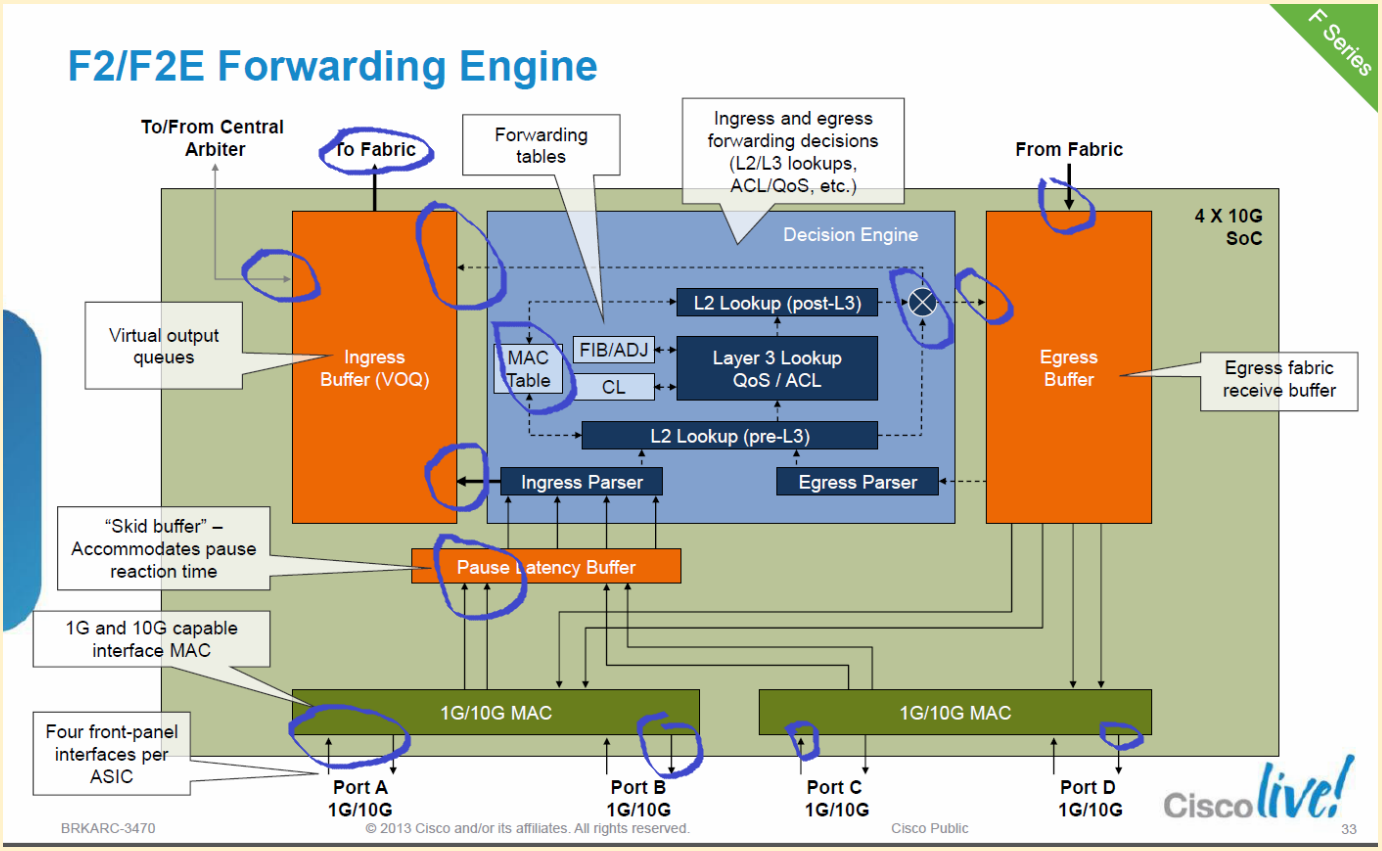
http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus5000/sw/python/api/python_api/API_functions.html#wp1053131

```
import os
import sys
import cisco – your main one
import socket – if you want to use some socket functions for client/server tcp use.
```

Some python pseudo code as an example -   a while loop would be better.

```
def ipinput():
    output = cisco.cli('show hostname')
    print(output)
    ipadd = raw_input('Enter IP address : ')
    clear()
    print(ipadd, "thank you")
    cisco.cli("ping", ipadd)
    yn = raw_input('is this correct(Y/N: ')
          if raw_input = 'y':
             cisco.cli("config t")
             cisco.cli("interface vlan2")
             cisco.cli("ip address", ipadd)
             cisco.cli("end")
             cisco.cli("show interface vlan2")
          elif raw_input = 'n'
              print("Let's try again")
          def ipinput()
```

```
print(time.asctime())

fproute = "show fabricpath route"
cisco.cli(fproute)

cisco.cli("config t")
cisco.cli("ping 1.1.1.1")

for screen clearing
def clear():
    print('\n' * 50)

def cls():
    os.system(['clear','cls'][os.name == 'nt'])

class hostname:
  def hosts():
    host = socket.gethostname()
    print("show hostname using sockets object")
    cisco.cli('show hostname')
    print("show hostname using nexus object")
    print(host)

Test read of a config file
5kDC1# >>> testfile = open('/bootflash/FPAUTHON5k.cfg')
5kDC1# >>> auth = testfile.read()
5kDC1# >>> print(auth)
```



F2/F2E Forwarding Engine

F Series

To/From Central Arbiter

To Fabric

Forwarding tables

Ingress and egress forwarding decisions (L2/L3 lookups, ACL/QoS, etc.)

From Fabric

4 X 10G SoC

Decision Engine

Virtual output queues

Ingress Buffer (VOQ)

L2 Lookup (post-L3)

MAC Table

FIB/ADJ

CL

Layer 3 Lookup QoS / ACL

L2 Lookup (pre-L3)

Ingress Parser

Egress Parser

Egress Buffer

Egress fabric receive buffer

“Skid buffer” – Accommodates pause reaction time

Pause Latency Buffer

1G and 10G capable interface MAC

1G/10G MAC

1G/10G MAC

Four front-panel interfaces per ASIC

Port A 1G/10G

Port B 1G/10G

Port C 1G/10G

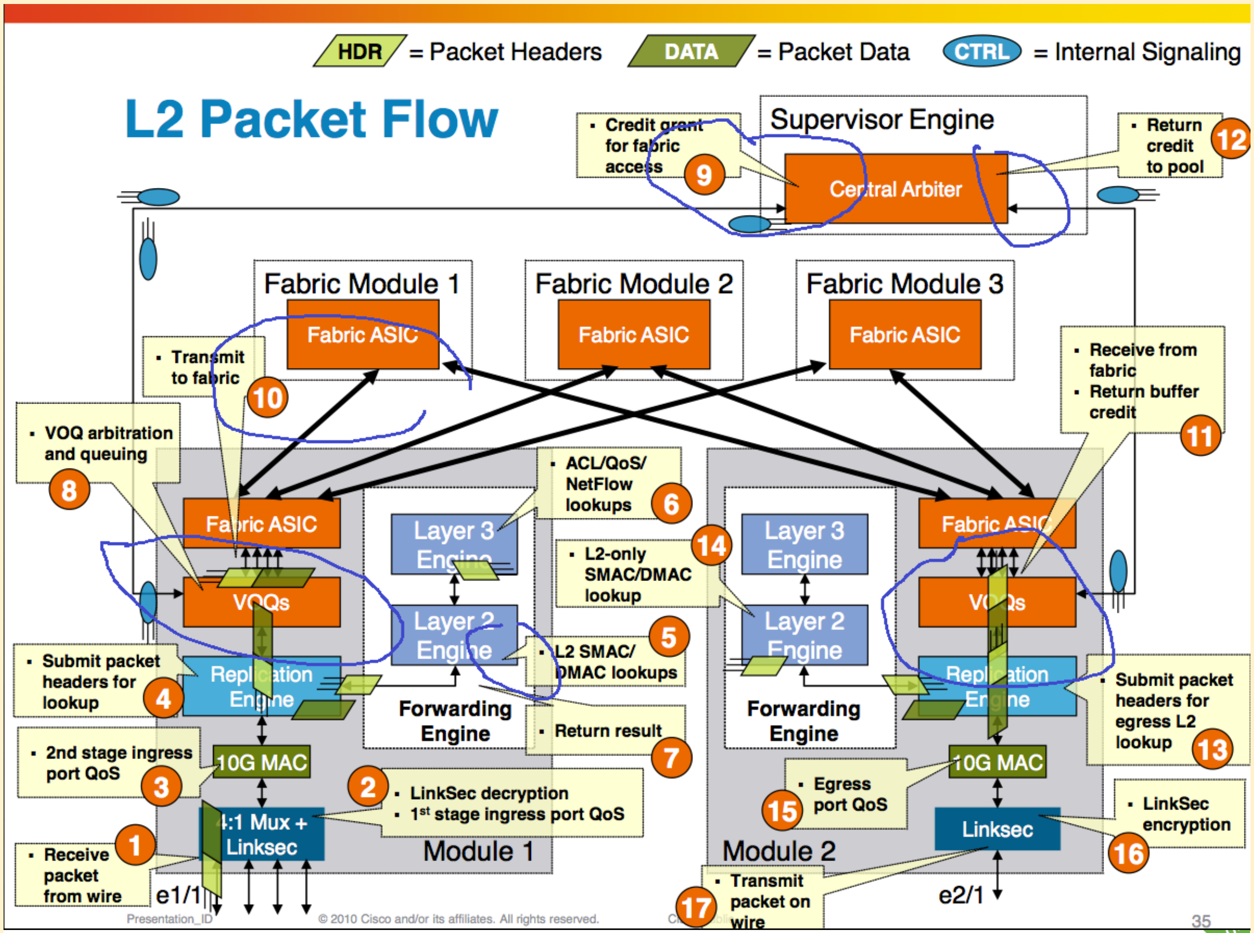
Port D 1G/10G

BRKARC-3470

© 2013 Cisco and/or its affiliates. All rights reserved.

Cisco Public

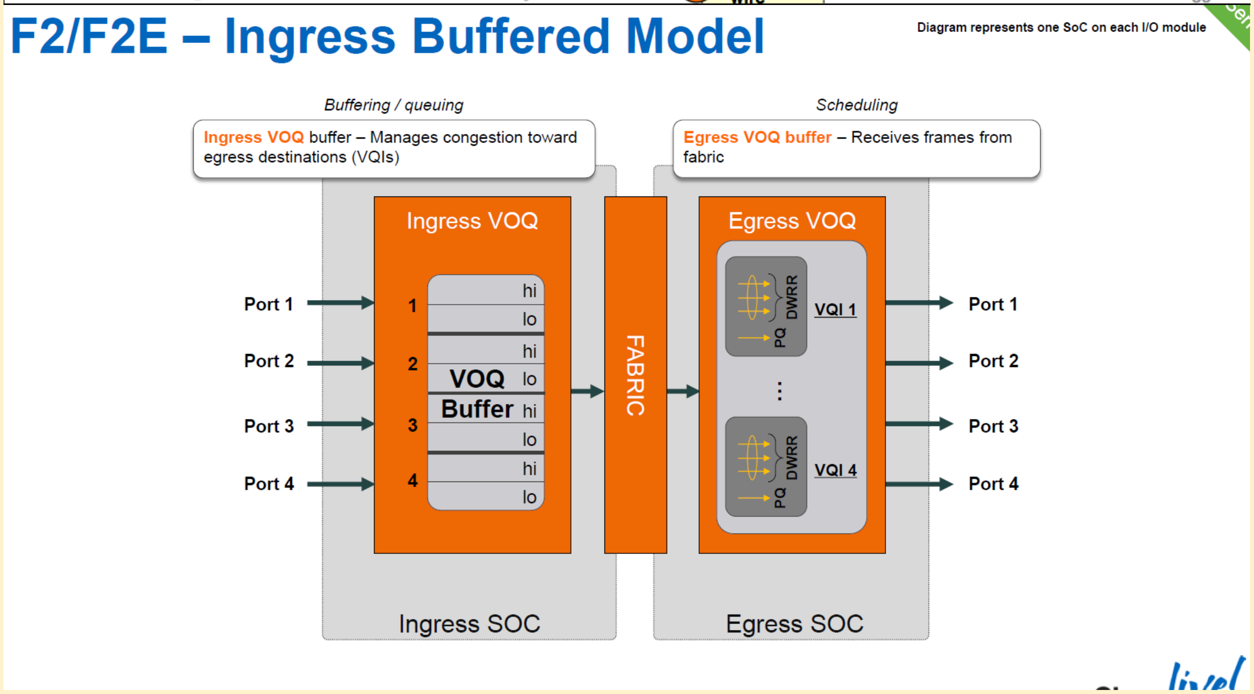
33



L2 Packet Flow

HDR = Packet Headers DATA = Packet Data CTRL = Internal Signaling

1. Receive packet from wire
2. LinkSec decryption; 1st stage ingress port QoS
3. 2nd stage ingress port QoS
4. Submit packet headers for lookup
5. L2 SMAC/DMAC lookups
6. ACL/QoS/NetFlow lookups
7. Return result
8. VOQ arbitration and queuing
9. Credit grant for fabric access
10. Transmit to fabric
11. Receive from fabric; Return buffer credit
12. Return credit to pool
13. Submit packet headers for egress L2 lookup
14. L2-only SMAC/DMAC lookup
15. Egress port QoS
16. LinkSec encryption
17. Transmit packet on wire



F2/F2E – Ingress Buffered Model

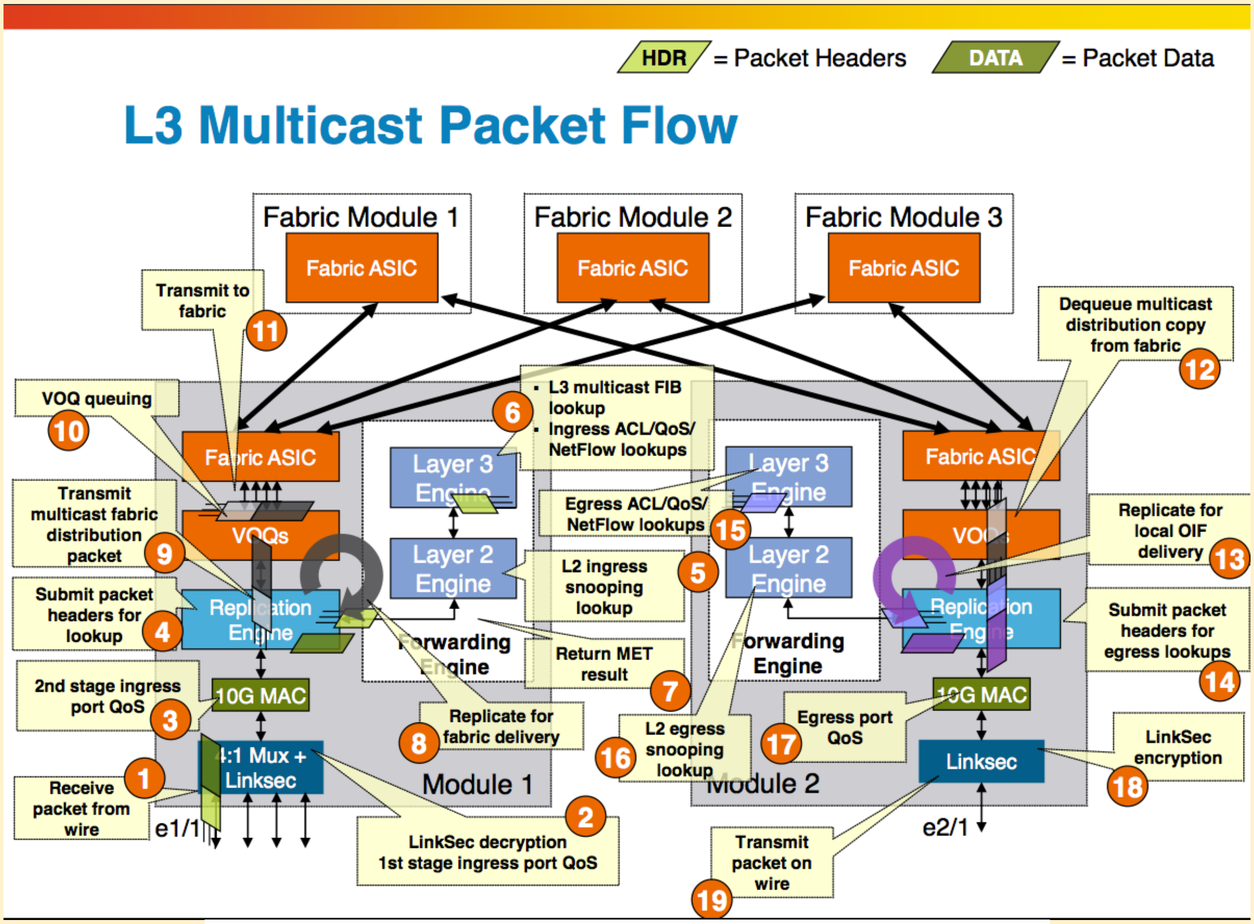
Diagram represents one SoC on each I/O module

Buffering / queuing

Ingress VOQ buffer – Manages congestion toward egress destinations (VQIs)

Scheduling

Egress VOQ buffer – Receives frames from fabric



HDR = Packet Headers   DATA = Packet Data

L3 Multicast Packet Flow

1 Receive packet from wire

2 LinkSec decryption
1st stage ingress port QoS

3 2nd stage ingress port QoS

4 Submit packet headers for lookup

5 L2 ingress snooping lookup

6
- L3 multicast FIB lookup
- Ingress ACL/QoS/ NetFlow lookups

7 Return MET result

8 Replicate for fabric delivery

9 Transmit multicast fabric distribution packet

10 VOQ queuing

11 Transmit to fabric

12 Dequeue multicast distribution copy from fabric

13 Replicate for local OIF delivery

14 Submit packet headers for egress lookups

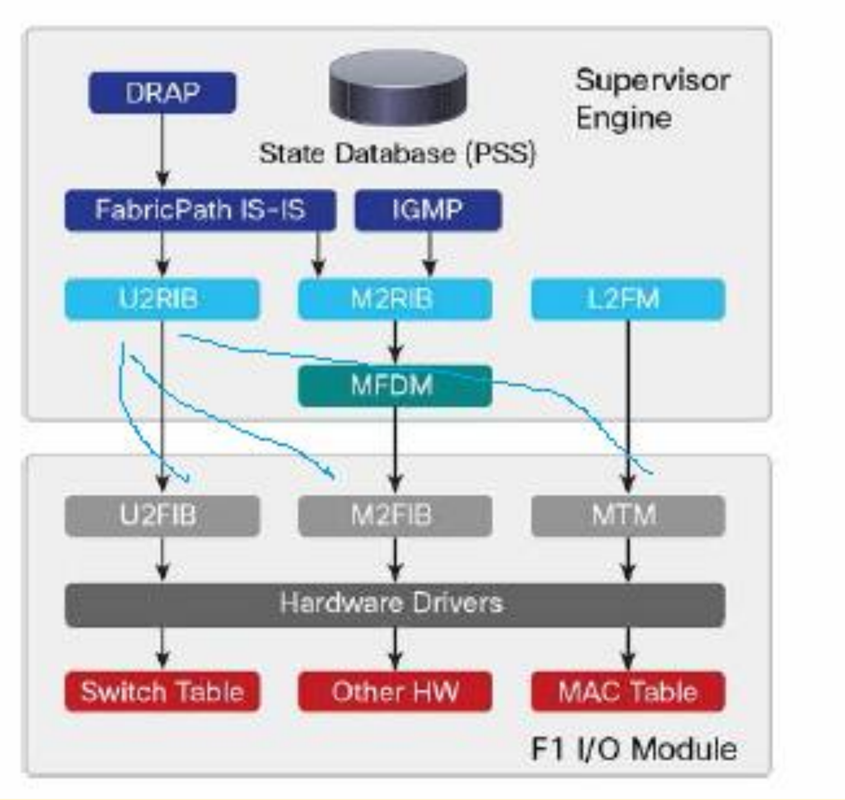
15 Egress ACL/QoS/ NetFlow lookups

16 L2 egress snooping lookup

17 Egress port QoS

18 LinkSec encryption

19 Transmit packet on wire



DRAP
State Database (PSS)
Supervisor Engine
FabricPath IS-IS
IGMP
U2RIB
M2RIB
L2FM
MFDM
U2FIB
M2FIB
MTM
Hardware Drivers
Switch Table
Other HW
MAC Table
F1 I/O Module