Overview

Link Aggregation Group (LAG) is a method to combine multiple **physical** ports/links that can be used to provide high combined bandwidth and/or redundancy in the network subsystem. It is necessary that all the physical ports of a LAG reside on the same network equipment (e.g. switch, CPE, router etc).

LAG is also sometimes referred as “Port Trunking” in L2 switches.

Linux Kernel refers LAG as “Interface Bonding” and the capability is provided through Kernel Bonding Driver.

Broadcom DSL CPE reference software has provided different ways to aggregate ports starting from 4.16L.01 software releases.

This application note describes those different implementations and finally converges into the Linux Bonding Driver based implementation.

Note: LAG/Interface-Bonding MUST not be confused with DSL Line Bonding.

Static Link Aggregation Support

Multiple Links can be statically aggregated when both LAG components are configured the same way. Static LAG is useful in scenarios where the link status does not change and remain stable (e.g. connecting two chips on the same board).

Static LAN Port Aggregation (Deprecated starting from 5.02L.03 release)

Starting from 4.16L.01 release of Broadcom Reference software, static LAN Port aggregation is supported on BCM63138 and BCM63148 platforms.

BCM63138/63148 platforms’ integrated L2 switch (LSW) supports “Port trunking” capability. Up to 4 ports of switch can be aggregated/trunked into one group and switch supports maximum of two trunk groups.

LAN-to-LAN traffic forwarding

Packet forwarding (Load balancing) from other LAN ports to these aggregated ports is done by LSW based on MAC (DA or SA or DA+SA) hash. MAC hash selection could be changed using CLI command “**ethswctl –c trunk**” or using API **bcm_port_trunk_set().**

WAN/Host/WLAN-to-LAN traffic forwarding

Packet forwarding (Load balancing) from WAN/Host/WLAN to these aggregated ports is done by Ethernet Driver based on MAC (DA+SA) hash. Load balancing logic could be changed by modifying the code in Ethernet Driver.

Usage Details

To enable/disable the “static LAN Port aggregation” feature before building:

1. Use the **make menuconfig** command at the Linux command prompt.
2. $ make menuconfig
3. Navigate to menuconfig→”Ethernet, Switch, and VLAN Selection”
4. Select [ ] “MAC Based Port Trunking (LAG)”
5. Save and exit

In board parameters, specify which LAN ports must be aggregated in which trunk group as below:

1. Two trunk groups are **PORT_FLAG_TRUNK_GROUP_0** and **PORT_FLAG_TRUNK_GROUP_1**
2. Set “bp_ulPortFlags” for the LAN ports to one of the above groups.

Ethernet Driver Logic

During board parameters parsing, Ethernet Driver will configure the LSW accordingly and will create **only ONE** Linux interface to represent all the LAN ports. This way, Linux isn’t aware of the link aggregation. Ethernet driver keeps the Linux interface UP as long as any one of the ports in the group is linked UP and will mark the interface down only when all the ports of the group are linked down.

Packet Acceleration

LAN to LAN traffic is handled by LSW while traffic from Host/WAN/WLAN is accelerated by HW accelerator based on flow-cache learning and the egress LAN port selected by Ethernet Driver based on load balancing algorithm.

Since this is static link aggregation, packet loss if possible when one of the links carrying the traffic goes down and the flows are accelerated by HW. This happens because flow-cache and Linux Kernel is unaware of link down event (as mentioned above).

Static WAN Port Aggregation (Deprecated starting from 5.02L.03 release)

Starting from 5.02L.02 release of Broadcom Reference software, static WAN Port aggregation is supported on BCM63138, BCM63148, BCM4908 and BCM62118 platforms.

These platforms inherently supports **only one** Ethernet WAN port connected to Runner Network processor. In order to aggregate two WAN ports, one of the LSW LAN port may be combined for aggregation purpose.

Notes:

1. Only one LSW LAN port is allowed for aggregation
2. In downstream, traffic could be received from both aggregated ports.
3. In upstream, only WAN port directly connected through Runner network processor can send the traffic.
4. This means, up to 2G downstream and 1G upstream traffic is supported.

LAN-to-LAN traffic forwarding

LAN to LAN forwarding isn’t affected by WAN port aggregation.

LAN/Host/WLAN-to-WAN traffic forwarding

Since only Runner WAN port is used for upstream, no changes will observed in upstream direction.

Dual-WAN-to-LAN/Host/WLAN traffic forwarding

Downstream traffic from LSW port is treated by Runner network processor as if the traffic is received from Runner WAN port. This implies that Runner acceleration logic is not affected except that the traffic towards LAN port will be looped back by Runner (and consequently consumes double the bandwidth between Runner & LSW port).

Usage Details

To enable/disable the “static WAN Port aggregation” feature:

1. Use the **make menuconfig** command at the Linux command prompt.
2. $ make menuconfig
3. Navigate to menuconfig→"Ethernet, Switch, and VLAN Selection"
4. Select [ ] "Static Ethernet WAN Bonding (LAG)"
5. Save and exit
6. Second Ethernet WAN port configuration is stored in NVRAM scratchpad.
7. "pspctl" command line utilities can be used to configure (set/get)
8. scratchpad string is "2ndWAN" and the LSW port value format is 0xYZ (exact 4 letters)
9. Port value layout is – bit: 7 = valid (always set), bit: 6 = unit (always set – currently only usage is LSW Port), bit: 5-0 = physical LSW port number.

Example

In order to set LSW port#2 as second WAN port, do the following from command line, restore default and reboot:

# pspctl set 2ndWAN 0xC2

#pspctl dump 2ndWAN

NVRAM scratchpad is persistent across reboots.

Ethernet Driver Logic

Ethernet Driver will read the NVRAM scratchpad area for 2ndWAN setting and configures the LSW & Runner accordingly. There will be no Linux interface to represent the 2nd WAN port from LSW. This way, Linux isn't aware of the link aggregation. In this configuration, the WAN link is UP only when the Ethernet Port connected through Runner is up.

Linux interface stats will be display as cumulative stats of both the aggregated ports.

Packet Acceleration

Packet acceleration by Runner network processor is transparent to the system in this implementation.

Linux Bonding Driver based Link aggregation Support

Linux bonding driver provides the required flexibility (no board parameters change or NVRAM) and avoids some of the limitations of static link aggregation.

Bonding driver could be used to achieve static or dynamic (LACP) bonding of ports.

General Bonding Driver Operations

1. Bonding driver module is loaded by inserting "bonding.ko"
2. Bonding driver creates Linux network devices (bond0, bond1 …), based on max number of bonding groups supported.
3. Using proc/file-system, interfaces can be added/removed to specific bonding groups (e.g. bond0, bond1 etc).
4. Since bondX interface represents all the aggregated physical ports, network operations should be performed on bonding interface (like routing, bridging etc).

5. In receive direction, bonding driver is “kind-of” pass through but when transmitting the packet, it makes load balancing decisions.
6. Bonding driver could be configured with following transmit policies to use (if applicable for selected mode of operations):
    a. LAYER2 : XOR (MACDA + MACSA + EtherType)
    b. LAYER34: layer 3+4 (IP ^ (TCP || UDP))
    c. LAYER23: layer 2+3 (IP ^ MAC)
    d. ENCAP23: encapsulated layer 2+3
    e. ENCAP34: encapsulated layer 3+4
7. Bonding driver could be configured in several modes to do load balancing (packet transmission) :
    a. Available Modes are:
        - 0-Round-Robin (RR),
        - 1-Active Backup,
        - 2-XOR,
        - 3-Broadcast,
        - 4-802.3AD (LACP),
        - 5-Transmit Load Balancing (TLB),
        - 6-Adaptive Load Balancing (ALB).
    b. From CPE/Gateway perspective, the most interesting modes are RR, XOR and 802.3AD
        - **Round-Robin Mode**: In RR mode, bonding driver will send N number of packets on each link in round robin order. “N” is a configurable parameter and default N=1. When N=0, bonding driver will randomly choose any link for transmission.
        - **XOR / 802.3AD Mode**: In these modes, the packet is transmitted based on selected transmit policy (as mentioned above). The difference between XOR & LACP mode is that LACP provides dynamic bonding option while XOR is static.
            - Important to understand that traffic is not dynamically assigned across member links but rather is distributed using a deterministic hash algorithm, depending on the transmit hash policy configuration.

<u>802.3AD/LACP Notes</u>

Few LACP configuration parameters to be aware of:

- **Mode (Active or Passive)**: In Active mode, ports will start to transmit LACPDUs as soon as link comes up. In Passive mode, the port won’t send LACPDUs until it receives from the partner. In practice, Active mode is the most useful mode and should be configured in Linux Bonding driver by setting “all_slaves_active =1”
- **Aggregation Selection Logic**: In general, LAG must be performed using same type of ports (speed, duplex). Aggregation Selection Logic means if there are multiple ports with different characteristics (speed, duplex), what should be the basis to aggregate the ports. Linux Bonding driver supports three link aggregation criteria: 0=stable (default), 1=bandwidth, 2=count.

For example, if there are three ports in the group with 100FD, 100FD, 1000FD --- Should only two ports w/ 100FD be lagged (based on count), or only 1000FD port should be used (based on bandwidth). Linux Bonding driver can be configured by setting “ad_select=0,1,2”.

- **Link Monitoring**: In order to detect the link failures, speed and duplex; Bonding driver must be able to monitor the link using standard ETHTOOL or MII IOCTLs. Bonding driver must be configured with desired link monitoring interval using “miimon” parameter otherwise the default 100ms is used.

LACP Terms and Parameters (Courtesy above mentioned link)

There are a number of LACP-specific terms and parameter names that must be understood in order to make sense of LACP output and packet traces.

The first and arguably most fundamental concept is that of **actors and partners**.

Actor: The term actor is used to designate the parameters and flags pertaining to the sending node.

Partner: The term partner is used to designate the sending node's view of its peer's parameters and flags. This essentially means that LACP echoes the parameters it receives back to the sender.

System-wide Parameters:

System ID: Each device (CPE) has a LACP System ID. This is a 48 bit value which generally defaults to the device MAC address. The system ID is sent within every LACPDU.

System Priority: 16 bit LACP System Priority is used to decide which system's port priorities are used to decide active / standby in the event that the two peers disagree. Lowest priority wins.

Per LAG Group Parameters:

LACP Key: A unique 16 bit LACP key per LAG group is used to detect cabling faults (cross cabling between different LAG groups).

LACP Flags Bits:

The following flag bits are used to communicate **state** (**Actor State or Partner State**) between systems:

Activity (bit:0) - LACP Active mode (1)/ Passive mode (0).

Timeout (bit:1) – LACPDU transmit rate requested: Fast 1Sec (1)/ Slow 30Sec (0)

Aggregation (bit:2) - Port is configured for aggregation (typically always set)

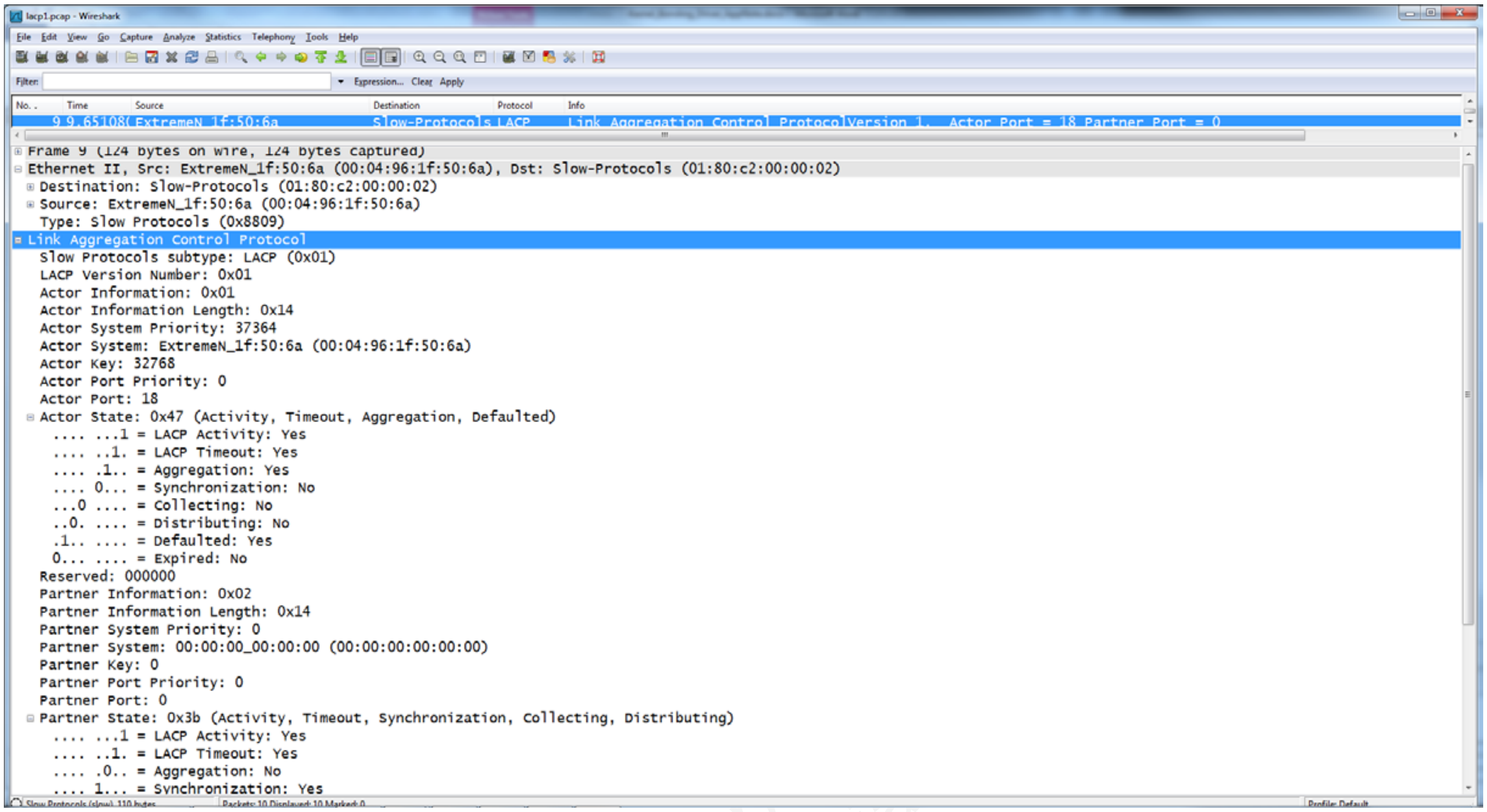
Synchronization (bit:3) - Set to indicate that the system is ready and willing to use this link in the bundle to carry traffic. Cleared to indicate the link is not usable or is in standby mode.

Collecting (bit:4) - Set to indicate that traffic received on this interface will be processed by the device.

Distributing (bit:5) - Set to indicate that the device is using this link to transmit traffic.

Defaulted (bit:6) - When set, indicates that no LACPDUs have been received during the past 6 intervals. Once the defaulted flag transitions to set, any stored partner information is flushed.

Expired (bit:7)- Set to indicate that no LACPDUs have been received by the device during the past 3 intervals.



```
No.   Time       Source                 Destination          Protocol  Info
  9 9.65108( ExtremeN_1f:50:6a          Slow-Protocols LACP            Link Aggregation Control ProtocolVersion 1.  Actor Port = 18 Partner Port = 0
```

```
⊞ Frame 9 (124 bytes on wire, 124 bytes captured)
⊟ Ethernet II, Src: ExtremeN_1f:50:6a (00:04:96:1f:50:6a), Dst: Slow-Protocols (01:80:c2:00:00:02)
  ⊞ Destination: Slow-Protocols (01:80:c2:00:00:02)
  ⊞ Source: ExtremeN_1f:50:6a (00:04:96:1f:50:6a)
    Type: Slow Protocols (0x8809)
⊟ Link Aggregation Control Protocol
    Slow Protocols subtype: LACP (0x01)
    LACP Version Number: 0x01
    Actor Information: 0x01
    Actor Information Length: 0x14
    Actor System Priority: 37364
    Actor System: ExtremeN_1f:50:6a (00:04:96:1f:50:6a)
    Actor Key: 32768
    Actor Port Priority: 0
    Actor Port: 18
  ⊟ Actor State: 0x47 (Activity, Timeout, Aggregation, Defaulted)
      .... ...1 = LACP Activity: Yes
      .... ..1. = LACP Timeout: Yes
      .... .1.. = Aggregation: Yes
      .... 0... = Synchronization: No
      ...0 .... = Collecting: No
      ..0. .... = Distributing: No
      .1.. .... = Defaulted: Yes
      0... .... = Expired: No
    Reserved: 000000
    Partner Information: 0x02
    Partner Information Length: 0x14
    Partner System Priority: 0
    Partner System: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Partner Key: 0
    Partner Port Priority: 0
    Partner Port: 0
  ⊟ Partner State: 0x3b (Activity, Timeout, Synchronization, Collecting, Distributing)
      .... ...1 = LACP Activity: Yes
      .... ..1. = LACP Timeout: Yes
      .... .0.. = Aggregation: No
      .... 1... = Synchronization: Yes
```

Usage Details

To enable/disable the “Kernel Bonding Driver” feature:

1. Use the **make menuconfig** command at the Linux command prompt.
2. $ make menuconfig
3. Navigate to menuconfig→”Kernel Configuration Selection”
4. Select [ ] “Enable Kernel Bonding Driver”
5. Navigate to menuconfig→”Ethernet, Switch, and VLAN Selection”
6. Select [ ] “Ethtool Support”
7. Save, exit and recompile.

Note: “Kernel Bonding Driver” support cannot be enabled if LAN/WAN Static LAG is enabled at compile time.

Ethernet Driver Changes

Ethernet Driver has provided hooks for Linux Bonding driver as below:

1. MAX Bonding group support: By default Ethernet Driver will support two bonding groups (either two LAN groups or one LAN & one WAN).
2. Bonding group change notification: Whenever any member port is added/removed from the group, Linux Bonding driver will notify Ethernet driver to appropriately configure the integrated L2-switch and/or Runner.

Kernel Bonding Driver Changes

Minor changes are done in bonding driver as below:

1. Added Blog hooks to link bonding device for acceleration and/or skip blog.
2. Calls to Ethernet Driver for bonding changes.
3. Provided hook to get the bonding group ID for a given slave interface.

Packet Acceleration

When the traffic is transmitted through bond interface, packet acceleration is only supported in XOR and 802.3AD bonding mode. These are the only two modes that utilizes flow-based transmit policy. In all other bonding modes, packet acceleration will be disabled.

There is no impact on acceleration when traffic is received by bonded ports.

Sample Configuration

# Insert Bonding module in LACP/802.3AD mode

```
insmod lib/modules/4.1.38/kernel/drivers/net/bonding/bonding.ko miimon=1000 mode=4 ad_select=2
xmit_hash_policy=1 all_slaves_active=1
```

# Bring down the ports that will be bonded

```
ifconfig eth3 down
ifconfig eth4 down
```

# Remove the ports from bridge, if needed

```
brctl delif br0 eth3
brctl delif br0 eth4
```

# Add the ports to bonding group bond0

```
echo +eth3 > sys/class/net/bond0/bonding/slaves
echo +eth4 > sys/class/net/bond0/bonding/slaves
```

# Add bond0 interface to bridge

```
brctl addif br0 bond0
```

# Bring all the interfaces up

```
ifconfig eth3 up
ifconfig eth4 up
ifconfig bond0 up
```

Frequently used commands

```
cat sys/class/net/bonding_masters
cat /proc/net/bonding/bond0
echo layer3+4 > sys/class/net/bond0/bonding/xmit_hash_policy
cat sys/class/net/bond0/bonding/mode
```