REALTEK

RTL9607C
SINGLE-CHIP PON

Realtek confidential for tenda

L2 Table Application Note
(CONFIDENTIAL: Development Partners Only)

Rev. 1.0.0
2 June 2017

REALTEK
Realtek Semiconductor Corp.
No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan
Tel.: +886-3-578-0211 Fax: +886-3-577-6047
www.realtek.com

COPYRIGHT

©2017 Realtek Semiconductor Corp. All rights reserved. No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form or by any means without the written permission of Realtek Semiconductor Corp.

TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

DISCLAIMER

Realtek provides this document “as is”, without warranty of any kind, neither expressed nor implied, including, but not limited to, the particular purpose. Realtek may make improvements and/or changes in this document or in the product described in this document at any time. This document could include technical inaccuracies or typographical errors.

USING THIS DOCUMENT

Though every effort has been made to assure that this document is current and accurate, more information may have become available subsequent to the production of this guide. In that event, please contact your Realtek representative for additional information that may help in the development process.

CONFIDENTIALITY

This document is confidential and should not be provided to a third-party without the permission of Realtek Semiconductor Corporation.

Realtek confidential for tenda

REVISION HISTORY

Revision	Release Date	Summary
1.0.0	2017/06/02	First Release

Table of Contents

Realtek confidential for tenda

List of Tables

List of Figures

Realtek confidential for tenda

1. Overview

L2 table in RTL9607C is a 2K entries table. This table is operated by a 4 way hash mechanism. Each entry in L2 table is 80 bits long and can be either L2 or L3 entry. The following figure gives all 3 types of L2 table entry.

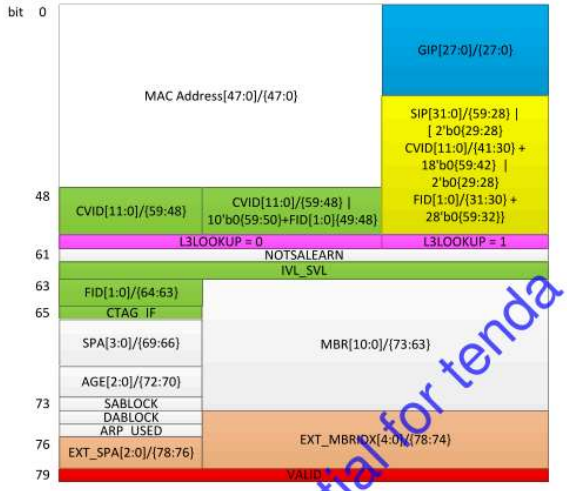


Figure 1. L2 Table entry format

Type 0 is L2 unicast type which can be created by users or dynamic learning by RTL9607C.

Type 1 is used to support L2 multicast. This type should be always static entry and only be created / deleted by software.

Type 2 is IPv4 multicast type. This type should be always static entry and only be created / deleted by software.

RTL9607C uses the following mechanism to determinate the type of entry. After knowing the type of entry, RTL9607C can extract each field in entry for further usage.

Realtek confidential for tenda

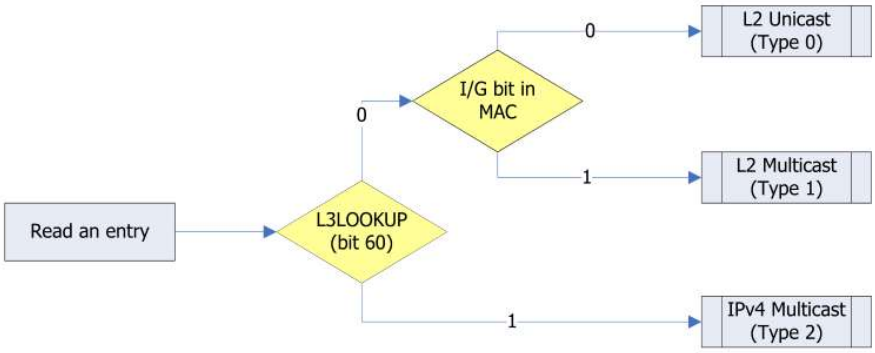


Figure 2. L2 Table entry decode flow

By knowing type of L2 entry, RTL9607C can extract the following fields from entry. All fields in L2 Table entry is listed as following.

Table 1. L2 Table Field

Field	Description	Bits
MAC	MAC address	48
FID	FID	2
L3LOOKUP	L3 or L2 lookup	1
MBR	Port list	11
EXT_MBRIDX	Index for Extension port list of EXT 17-0	5
NOTSALEARN	Not ASIC SA auto learning entry 1. unicast entry is valid if NOTSALREAN=1 or AGE!=0 2. L2/L3 multicast entry is valid if NOTSALREAN=1	1
SABLOCK	Source MAC blocking	1
DABLOCK	Destination MAC blocking	1
AGE	Aging Time	3
SPA	Source port of learning address	4
EXT_SPA	Source extension of learning address	3
L2CVID	Learning Ctag VID(SVL and Ctagging) or Ingress CVLAN VID(IVL or untagging)	12
IVL_SVL	L2 hash with VID(IVL_SVL=1) or FID(IVL_SVL=0)	1
ARP_USAGE	Unicast address for ARP usage	1
CTAG_IF	C-tagging or not for ingress CVLAN learning	1

Realtek confidential for tendo

2. L2 Table Learning

When RTL9607C receives a packet, it will try to lookup the source MAC address of packet in L2 table. If this is a new MAC address, RTL9607C will try to write this MAC into L2 Table. If this is MAC address already exists in L2 Table, RTL9607C will update it port number information and refresh ageing timer field in this entry.

RTL9607C supports IVL (independent VLAN Learning), SVL (Shared VLAN Learning) and Hybrid (Both IVL & SVL) mode. The IVL and SVL mode selection in learning is dependent on VLAN setting. Thus, users can configure some VLANs in IVL and other in SVL. If a VLAN is configured in IVL mode, all the packets belongs to this VLAN will be learned in IVL mode and vice versa. The entire learning procedure in IVL and SAL is given in the following section.

All the dynamic entries learned by RTL9607C are always type 0 which is defined in Figure 1.

2.1. Independent & Shared VLAN Learning

Realtek confidential for tenda

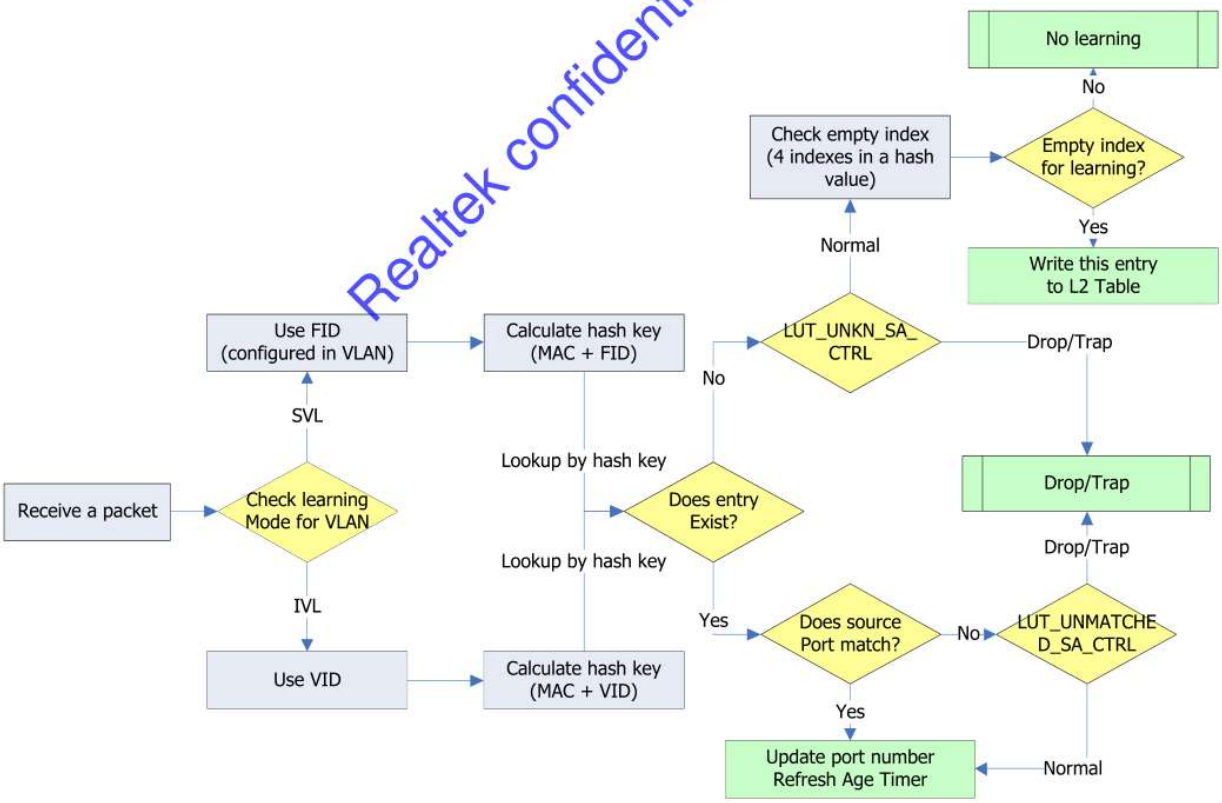


Figure 3. L2 Table learning flow

When the learning process begins, RTL9607C will try to get the learning mode of this packet from VLAN table. This is because that RTL9607C needs to calculate the hash key before trying to locate an index in L2 Table. If learning mode is SVL, RTL9607C will use MAC + FID to calculate hash key. If learning mode is IVL, RTL9607C will use MAC + VID to calculate hash key. The hash key calculation algorithm for L2 unicast entry is given as following.

L2 Unicast (SVL):

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus FID_{01}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus FID_{00}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45}$$

L2 Unicast (IVL):

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44} \oplus VID_{05}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43} \oplus VID_{04}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42} \oplus VID_{03}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41} \oplus VID_{02} \oplus VID_{11}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus VID_{01} \oplus VID_{10}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus VID_{00} \oplus VID_{09}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47} \oplus VID_{08}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46} \oplus VID_{07}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45} \oplus VID_{06}$$

Realtek confidential for tenda

No matter which learning mode is, RTL9607C will get a 9 bits hash key (0~511) after the calculation. Every 4 indexes in L2 Tables share the same hash key. In other words, RTL9607C only needs to lookup 4 entries and it can know a specified entry is in L2 Table or not.

By having the hash key, RTL9607C would look up the specified MAC in these 4 entries. If the MAC address is found, RTL9607C will compare the source port information between L2 entry and packet received port number. If the source port number is matched, it will refresh the ageing time field in L2 entry. If the source port number is not matched, the action is controlled by LUT_UNMATCHED_SA_CTRL. Only when the action of LUT_UNMATCHED_SA_CTRL is setting

as “Normal”, RTL9607C would update the source port number in L2 entry and refresh the Ageing timer field. All other settings will not change the exist entry in L2 Table.

If the MAC address is not found in L2 Table, this action is controlled by LUT_UNKN_SA_CTRL. Just like LUT_UNMATCHED_SA_CTRL, when the action of LUT_UNKN_SA_CTRL is setting as “Normal”, RTL9607C would update the source port number in L2 entry and refresh the Ageing timer field. Otherwise, the original entry will not be touched.

All 4 actions of LUT_UNMATCHED_SA_CTRL & LUT_UNKN_SA_CTRL are listed:

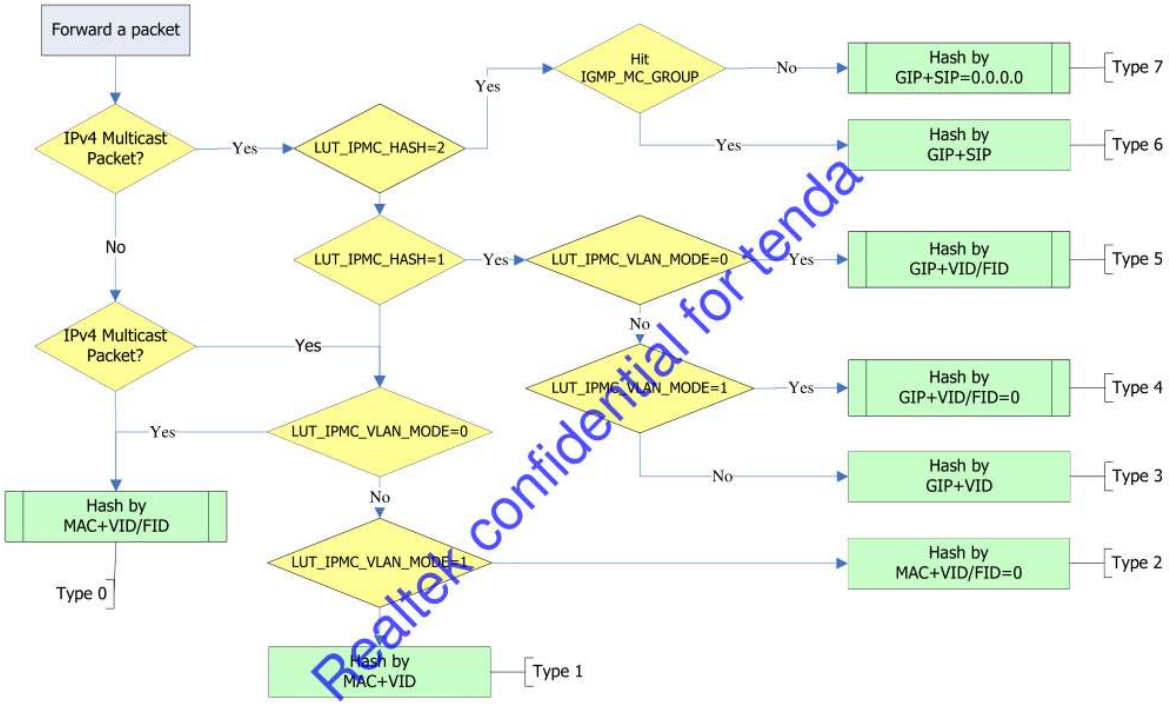
Table 2. Learning action setting

Field Name	Bits	Description
LUT_UNKN_SA_CTRL	2	Per-port drop/trap packet if SA is unknown 0b00: normal 0b01: drop packet & disable learning 0b10: trap to CPU & disable learning 0b11: reserved
LUT_UNMATCHED_SA_CTRL	2	Per-port drop/trap packet if SA is not from the same source port as L2 SPA. If SPA is CPU port, then need to check EXT_SPA. 0b00: normal 0b01: drop packet & disable learning 0b10: trap to CPU & disable learning 0b11: reserved

Realtek confidential for tenda

3. L2 Table Lookup & Forwarding

When RTL9607C needs to forward a packet out, it will do a lookup in L2 table and forward the packet to the port/port mask in L2 entry. RTL9607C supports L2 unicast, L2 multicast & L3 multicast forwarding. That is, RTL9607C needs to know to calculate the hash key before lookup. The following figure shows the decision of hash key calculation.



Realtek confidential for tenda

Figure 4. Lookup decision flow

3.1. Independent & Shared VLAN Forwarding

If the decision of lookup is Type 0, RTL9607C will use MAC + VID/FID to calculate the hash key. Like dynamic address learning, RTL9607C needs to know IVL or SVL before calculate. By having this information, RTL9607C can calculate a hash key and lookup into L2 Table. The hash key calculation algorithm for Type 0 is as same as dynamic address learning in section 2.1. For Type 0, the hash key calculation algorithm is as following:

L2 Multicast (SVL):

key [08] = $MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44}$
key [07] = $MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43}$
key [06] = $MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42}$
key [05] = $MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41}$
key [04] = $MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus FID_{01}$
key [03] = $MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus FID_{00}$
key [02] = $MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47}$
key [01] = $MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46}$
key [00] = $MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45}$

L2 Multicast (IVL):

key [08] = $MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44} \oplus VID_{05}$
key [07] = $MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43} \oplus VID_{04}$
key [06] = $MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42} \oplus VID_{03}$
key [05] = $MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41} \oplus VID_{02} \oplus VID_{11}$
key [04] = $MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus VID_{01} \oplus VID_{10}$
key [03] = $MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus VID_{00} \oplus VID_{09}$
key [02] = $MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47} \oplus VID_{08}$
key [01] = $MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46} \oplus VID_{07}$
key [00] = $MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45} \oplus VID_{06}$

Realtek confidential for tenda

If the entry is found in L2 Table, the packet will be forwarded to port / port mask according to the SPA or MBR/EXT_MBR field of the entry. If the entry is not found, this packet will follow unknown packet control action. We will discuss this at Section 3.4.

3.2. IPv4 Multicast Lookup

If the decision of lookup is Type 3 to 7 that means this packet is an IPv4 multicast packet. RTL9607C will do the IPv4 multicast lookup and forwarding. RTL9607C will use GIP + SIP + VID/FID to calculate the hash key. The hash key calculation is as following:

IPv4 Multicast while LUT_IPMC_VLAN_MODE=2 && LUT_IPMC_HASH=1 (Type3):

key [08] = $GIP_{08} \oplus GIP_{17} \oplus GIP_{26} \oplus VID_{05}$
key [07] = $GIP_{07} \oplus GIP_{16} \oplus GIP_{25} \oplus VID_{04}$
key [06] = $GIP_{06} \oplus GIP_{15} \oplus GIP_{24} \oplus VID_{03}$
key [05] = $GIP_{05} \oplus GIP_{14} \oplus GIP_{23} \oplus VID_{02} \oplus VID_{11}$
key [04] = $GIP_{04} \oplus GIP_{13} \oplus GIP_{22} \oplus VID_{01} \oplus VID_{10}$

key [03] = $GIP_{03} \oplus GIP_{12} \oplus GIP_{21} \oplus VID_{00} \oplus VID_{09}$
key [02] = $GIP_{02} \oplus GIP_{11} \oplus GIP_{20} \oplus VID_{08}$
key [01] = $GIP_{01} \oplus GIP_{10} \oplus GIP_{19} \oplus VID_{07}$
key [00] = $GIP_{00} \oplus GIP_{09} \oplus GIP_{18} \oplus GIP_{27} \oplus VID_{06}$

IPv4 Multicast while LUT_IPMC_VLAN_MODE=2 && LUT_IPMC_HASH=1 (Type4):

key [08] = $GIP_{08} \oplus GIP_{17} \oplus GIP_{26}$
key [07] = $GIP_{07} \oplus GIP_{16} \oplus GIP_{25}$
key [06] = $GIP_{06} \oplus GIP_{15} \oplus GIP_{24}$
key [05] = $GIP_{05} \oplus GIP_{14} \oplus GIP_{23}$
key [04] = $GIP_{04} \oplus GIP_{13} \oplus GIP_{22}$
key [03] = $GIP_{03} \oplus GIP_{12} \oplus GIP_{21}$
key [02] = $GIP_{02} \oplus GIP_{11} \oplus GIP_{20}$
key [01] = $GIP_{01} \oplus GIP_{10} \oplus GIP_{19}$
key [00] = $GIP_{00} \oplus GIP_{09} \oplus GIP_{18} \oplus GIP_{27}$

IPv4 Multicast while LUT_IPMC_VLAN_MODE=0 && LUT_IPMC_HASH=1 (Type5):

key [08] = $GIP_{08} \oplus GIP_{17} \oplus GIP_{26}$
key [07] = $GIP_{07} \oplus GIP_{16} \oplus GIP_{25}$
key [06] = $GIP_{06} \oplus GIP_{15} \oplus GIP_{24}$
key [05] = $GIP_{05} \oplus GIP_{14} \oplus GIP_{23}$
key [04] = $GIP_{04} \oplus GIP_{13} \oplus GIP_{22} \oplus FID_{01}$
key [03] = $GIP_{03} \oplus GIP_{12} \oplus GIP_{21} \oplus FID_{00}$
key [02] = $GIP_{02} \oplus GIP_{11} \oplus GIP_{20}$
key [01] = $GIP_{01} \oplus GIP_{10} \oplus GIP_{19}$
key [00] = $GIP_{00} \oplus GIP_{09} \oplus GIP_{18} \oplus GIP_{27}$

Realtek confidential for tenda

IPv4 Multicast while LUT_IPMC_VLAN_MODE=2 && LUT_IPMC_HASH=2 && Hit IGMP_MC_GROUP (Type6):

key [08] = $GIP_{08} \oplus GIP_{17} \oplus GIP_{26} \oplus SIP_{07} \oplus SIP_{16} \oplus SIP_{25}$
key [07] = $GIP_{07} \oplus GIP_{16} \oplus GIP_{25} \oplus SIP_{06} \oplus SIP_{15} \oplus SIP_{24}$
key [06] = $GIP_{06} \oplus GIP_{15} \oplus GIP_{24} \oplus SIP_{05} \oplus SIP_{14} \oplus SIP_{23}$
key [05] = $GIP_{05} \oplus GIP_{14} \oplus GIP_{23} \oplus SIP_{04} \oplus SIP_{13} \oplus SIP_{22} \oplus SIP_{31}$
key [04] = $GIP_{04} \oplus GIP_{13} \oplus GIP_{22} \oplus SIP_{03} \oplus SIP_{12} \oplus SIP_{21} \oplus SIP_{30}$
key [03] = $GIP_{03} \oplus GIP_{12} \oplus GIP_{21} \oplus SIP_{02} \oplus SIP_{11} \oplus SIP_{20} \oplus SIP_{29}$
key [02] = $GIP_{02} \oplus GIP_{11} \oplus GIP_{20} \oplus SIP_{01} \oplus SIP_{10} \oplus SIP_{19} \oplus SIP_{28}$
key [01] = $GIP_{01} \oplus GIP_{10} \oplus GIP_{19} \oplus SIP_{00} \oplus SIP_{09} \oplus SIP_{18} \oplus SIP_{27}$
key [00] = $GIP_{00} \oplus GIP_{09} \oplus GIP_{18} \oplus GIP_{27} \oplus SIP_{08} \oplus SIP_{16} \oplus SIP_{26}$

IPv4 Multicast while LUT_IPMC_VLAN_MODE=2 && LUT_IPMC_HASH=2 && NO Hit IGMP_MC_GROUP (Type7):

key [08] = $GIP_{08} \oplus GIP_{17} \oplus GIP_{26}$
key [07] = $GIP_{07} \oplus GIP_{16} \oplus GIP_{25}$
key [06] = $GIP_{06} \oplus GIP_{15} \oplus GIP_{24}$
key [05] = $GIP_{05} \oplus GIP_{14} \oplus GIP_{23}$
key [04] = $GIP_{04} \oplus GIP_{13} \oplus GIP_{22}$
key [03] = $GIP_{03} \oplus GIP_{12} \oplus GIP_{21}$
key [02] = $GIP_{02} \oplus GIP_{11} \oplus GIP_{20}$
key [01] = $GIP_{01} \oplus GIP_{10} \oplus GIP_{19}$
key [00] = $GIP_{00} \oplus GIP_{09} \oplus GIP_{18} \oplus GIP_{27}$

If the entry is not found, this packet will follow unknown packet control action. We will discuss this at Section 3.4

For supporting include/exclude mode forwarding which is defined in IGMPv3. RTL9607C uses L2 Table and IGMP_MC_GROUP to offer this functionality. IGMP_MC_GROUP is a 64 entries table and given as following:

Table 3. IGMP_SIP_FILTER_TBL

Field	Description	Bits
Valid	Valid entry	1
GIPn	IP Multicast Group Address	28
MBR	Port list	11
EXT_MBRIDX	Index for Extension port list of EXT 17-0	5

Realtek confidential for tenda

GIP is multicast group IP. The multicast IP address is always class D address. When RTL9607C try to forward an IPv4 multicast packet and LUT_IPMC_HASH is 2, ASIC will lookup IGMP_MC_GROUP first. If the GIP hit an entry in IGMP_MC_GROUP, then ASIC will hash by GIP+SIP. If the result of lookup IGMP_MC_GROUP table is miss, ASIC will only use GIP to look entry up in L2 table. After L2 table hash, ASIC will forward the packet according to the MBR in IGMP_MC_GROUP or L2 Table.

If users need an IGMPv3 Include mode group setting, users should create an entry in IGMP_MC_GROUP and add an associated {GIP, SIP, MBR != 0} entry in L2 Table. Thus, ASIC will refer the L2 table MBR as forward port mask.

If users need an Exclude mode group setting, users should set the SIP entry as {GIP, SIP, MBR=0} in L2 Table, and create an associated IGMP_MC_GROUP entry with correct MBR or EXT_MBRIDX. That is, if the packet could not hit any entry in L2 table but it get hit in IGMP_MC_GROUP, ASIC would forward it to MBR or EXT_MBRIDX in IPMC_GROUP_TABLE.

3.3. IPv6 Multicast Lookup

In RTL9607C, the IPv6 multicast packet might be classified to Type0 to Type3. RTL9607C will do the L2 table hash only by MAC+FID/VID. The hash key calculation is as following:

Type0 SVL:

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus FID_{01}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus FID_{00}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45}$$

Type0 IVL:

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44} \oplus VID_{05}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43} \oplus VID_{04}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42} \oplus VID_{03}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41} \oplus VID_{02} \oplus VID_{11}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus VID_{01} \oplus VID_{10}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus VID_{00} \oplus VID_{09}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47} \oplus VID_{08}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46} \oplus VID_{07}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45} \oplus VID_{06}$$

Type1:

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44} \oplus VID_{05}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43} \oplus VID_{04}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42} \oplus VID_{03}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41} \oplus VID_{02} \oplus VID_{11}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40} \oplus VID_{01} \oplus VID_{10}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39} \oplus VID_{00} \oplus VID_{09}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47} \oplus VID_{08}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46} \oplus VID_{07}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45} \oplus VID_{06}$$

Realtek confidential for tenda

Type2:

$$key[08] = MAC_{08} \oplus MAC_{17} \oplus MAC_{26} \oplus MAC_{35} \oplus MAC_{44}$$
$$key[07] = MAC_{07} \oplus MAC_{16} \oplus MAC_{25} \oplus MAC_{34} \oplus MAC_{43}$$
$$key[06] = MAC_{06} \oplus MAC_{15} \oplus MAC_{24} \oplus MAC_{33} \oplus MAC_{42}$$
$$key[05] = MAC_{05} \oplus MAC_{14} \oplus MAC_{23} \oplus MAC_{32} \oplus MAC_{41}$$
$$key[04] = MAC_{04} \oplus MAC_{13} \oplus MAC_{22} \oplus MAC_{31} \oplus MAC_{40}$$
$$key[03] = MAC_{03} \oplus MAC_{12} \oplus MAC_{21} \oplus MAC_{30} \oplus MAC_{39}$$
$$key[02] = MAC_{02} \oplus MAC_{11} \oplus MAC_{20} \oplus MAC_{29} \oplus MAC_{38} \oplus MAC_{47}$$
$$key[01] = MAC_{01} \oplus MAC_{10} \oplus MAC_{19} \oplus MAC_{28} \oplus MAC_{37} \oplus MAC_{46}$$
$$key[00] = MAC_{00} \oplus MAC_{09} \oplus MAC_{18} \oplus MAC_{27} \oplus MAC_{36} \oplus MAC_{45}$$

If the entry is not found, this packet will follow unknown packet control action. We will discuss this at Section 3.4

3.4. Unknown Packet Control

If the decision of lookup for a packet is Type 0, that also means the packet is an L2 unicast packet. If its destination MAC address can not be founded in L2 Table, it will be classified as an “Unknown unicast” packet. This kind of packet is controlled by LUT_UNKN_UC_DA_CTRL. The possible action of this control can be Flooding / Drop / Trap to CPU. If users configure this action as “Flooding”, users can also specify the “Flooding port mask for unknown unicast”.

If the decision of lookup for a packet is not Type 0, this packet is a multicast packet. (Either L2 or L3) If its destination MAC or GIP address can not be founded in L2 Table, it will be classified as an “Unknown Multicast” packet. 3 registers control the behavior of this kind of packet dependent on packet format.

Realtek confidential for tenda

Table 4. Unknown Multicast Control

Field Name	Bits	Description
UNKN_IP4_MC_ACT	2	Per port unknown IPv4 multicast frame behaviour 0b00: normal flooding, exclude IGMP packets 0b01: drop packet, exclude IP 224.0.0.x only when UNKN_RESERVED_IP4_ACT != 0 and IGMP packets 0b10: trap to CPU, exclude IP 224.0.0.x only when UNKN_RESERVED_IP4_ACT != 0 and IGMP packets 0b11: Reserved
UNKN_IP6_MC_ACT L	2	Per port unknown Ipv6 multicast frame behaviour 0b00: normal flooding, exclude MLD packets 0b01: drop packet, exclude UNKN_MC_IP6_RSV_ADDR configured addressonly when UNKN_RESERVED_IP6_ACT != 0 and MLD packets 0b10: trap to CPU, exclude UNKN_MC_IP6_RSV_ADDR configured address only when UNKN_RESERVED_IP6_ACT != 0 and MLD packets 0b11: Reserved
UNKN_L2_MC_ACT	2	Per port unknown L2 multicast frame behaviour 0b00: normal flooding exclude IGMP/MLD

	0b01: drop packet exclude IGMP/MLD 0b10: trap to CPU exclude IGMP/MLD 0b11: drop packet exclude RMA(01-80-C2-00-00-xx)/IGMP/MLD

If users configure above configuration as “Flooding”, users can also configure it flooding port mask.

3.5. Source Port Filtering

No matter the packet type and the lookup result (known or unknown), the packet will not be forwarded to the port which RTL9607C receive it. Even if the source port is set in the MBR/EXT_MBR, RTL9607C will mask out this port when egress. However, some application may need RTL9607C forward the packet back, the following configuration give the ability to permit/forbid this kind of forwarding behavior.

Table 5. Source Port Filtering Control

Field Name	Bits	Description
L2_SRC_PORT_PERMIT	1	Per-port don’t filter packet if source port and destination port are the same 0b0: normal, filtering source port in forwarding portmask 0b1: just forward packet
L2_SRC_EXT_PERMIT	1	Per-EXT don’t filter packet if source port and destination port are the same 0b0: normal, filtering source port in forwarding portmask 0b1: just forward packet

Realtek confidential for tenda

4. L2 Table Configuration

This section gives some features discussion of L2 Table. RTL9607C supports automatically dynamic learning for L2 unicast address. An Ageing timer mechanism is also provided to collect entry space from unused entry. Ageing timer is configurable and will be discussed at section 4.1.

Another feature if learning limitation, users can specify a per-port or per-system learning limitation. Learning over action for those packets which can not be learned due to this limitation can be configured also. This feature is discussed at Section 4.2.

Once a port is link down, users can request to clear all dynamic entries. This configuration is give at Section 4.3.

4.1. Ageing Timer

RTL9607C provides the following configuration to control the ageing mechanism.

Table 6. Ageing Timer Control

Field Name	Bits	Description
LUT_AGEOUT_CTRL	1	Per-port L2 LUT age update setting 0b0: disable age update 0b1: enable age update
AGE_SPD	21	L2 lookup table aging speed/period for each 2K entries, unit 0.1 sec 0b0:reserved

By default, LUT_AGEOUT_CTRL is 0x1. That means the ageing mechanism is enabled. AGE_SPEED is used for controlling the time period which a dynamic L2 entry can exist in L2 Table. When a dynamic L2 entry is learned or refreshed, the Age field in L2 entry will be set to 7. This field will be decreased by Aging mechanism. If this field is decreased to 0, the L2 entry is deleted.

The time period of decreasing Age field from 7 to 0 is AGE_SPEED.

Realtek confidential for tenda

4.2. Learning Limitation

The size of L2 Table in RTL9607C is 2K entries. However, RTL9607C also provides the following control register to configure a system-based and port-based limitation.

Table 7. Learning Limitation Control

Field Name	Bits	Description
LUT_LRN_LIMITNO	12	System and per-port L2 entries limitation number
LUT_LEARN_OVER_ACT	2	Per-port auto leaning number exceed behavior

		0b00: normal forwarding 0b01: drop packet 0b10: trap to CPU 0b11: Reserved(as 0b00)
LUT_SYS_LRN_LIMITNO	12	System-based L2 entries limitation number
LUT_SYS_LRN_OVER_CTRL	2	System-based auto leaning number exceed behavior 0b00: normal forwarding 0b01: drop packet 0b10: trap to CPU 0b11: Reserved(as 0b00)
LUT_SYS_LRN_LIMIT_EN	1	Per port learning count is counted as system-based L2 learning count and need to check system-based learning limit 0b0:disble 0b1:enable
LUT_ENTRY_FULL_ACT	1	SMAC(unicast only) can not be learned because LUT 4 way entries full (without CAM) or LUT 4 way + CAM full(with CAM) 0b0:Forward 0b1:Trap to CPU

Per-port leaning counters and system learning counter are used to count the number of dynamic entries. Before a new dynamic entry is learned, the learning counters will be compared to LUT_LRN_LIMITNO. A dynamic address can be learned only when the port-based and system-based limitation is not reached. User can set not using system learning counter by LUT_SYS_LRN_LIMIT_EN.

Only dynamic L2 entry will be counted in learning counters. Static entries will not affect the behavior of learning limitation function.

It dynamic entry learned by ASIC can't find empty entry, we can decide this entry to forward or trap to CPU.

Realtek confidential for tenda

4.3. Link down Flush

If a port of RTL9607C is link down, RTL9607C will flush all the dynamic entries in that port by default. The following configuration can be used to control this behavior.

Table 8. Learning Limitation Control

Field Name	Bits	Description
LUT_FLUSH_DYNAMIC	1	Flush dymanic entries(both ARP_USAGE=0 and NOTSALEARN=0) 0b1: enable
LUT_FLUSH_STATIC	1	Flush static entries(include ARP_USAGE=1 or NOTSALEARN=1) 0b1: enable
LINKDOWN_AGEOUT	1	Link down port aging out setting 0b0:disable aging out 0b1:enable force aging out all L2 lookup entries belong to link down ports

If LINKDOWN_AGEOUT is set to 1, RTL9607C will flush entries when port is link down. The type of entries which will be flushed is controlled by LUT_FLUSH_DYNAMIC and LUT_FLUSH_STATIC.

Realtek confidential for tenda

5. L2 Table Entry operation

Some fields in L2 entry are used to perform additional feature. The following section would discuss these fields and their functionality.

5.1. Block

The easiest way to configure RTL9607C dropping a packet with specified destination MAC address or a source MAC address is to use DA_BLOCK and SA_BLOCK field in a L2 entry. When RTL9607C receives a packet, it will do a source MAC address lookup in L2 Table for learning. It L2 entry is found and SA_BLOCK of the L2 entry is set, this packet will be dropped.

Like SA_BLOCK, when RTL9607C do the destination MAC address lookup and L2 entry with DA_BLOCK setting exists, this packet will be dropped.

SA_BLOCK function works on all types of packets due to RTL9607C will perform source MAC lookup for every incoming packet. However, if the lookup decision is not L2 unicast, there is no DA_BLOCK action will be perform due to this field is only in Type 0.

Realtek confidential for tenda

6. Sample code

This chapter gives some sample code to configure the functionality of L2 Table via SDK.

6.1. Initialization

API “rtk_l2_init” can be used to initialize entire L2 module. This API can also be used as reset function. When this function is called, L2 module will be reset.

```
/* Initialize L2 Module */
rtk_l2_init();
```

6.2. Add & Delete an entry

All 4 types of L2 entry can be added and deleted by SDK. This section gives the sample for adding and deleting a L2 entry in L2 Table.

```
/* Add an IVL L2 unicast entry with MAC: 00:01:02:03:04:05 and VID = 1 */
/* source port = 2, source address priority = 4 */

rtk_l2_ucastAddr_t l2Addr;

osal_memset(&l2Addr, 0x00, sizeof(rtk_l2_ucastAddr_t));
l2Addr.vid  = 1;
L2Addr.mac.octet[0] = 0x00;
L2Addr.mac.octet[1] = 0x01;
L2Addr.mac.octet[2] = 0x02;
L2Addr.mac.octet[3] = 0x03;
L2Addr.mac.octet[4] = 0x04;
L2Addr.mac.octet[5] = 0x05;
L2Addr.port = 2;
L2Addr.priority = 4;
L2Addr.flags = (RTK_L2_UCAST_FLAG_LOOKUP_PRI |
                RTK_L2_UCAST_FLAG_STATIC |
                RTK_L2_UCAST_FLAG_IVL );
```

Realtek confidential for tenda

```
rtk_l2_addr_add(&l2Addr);
```

```
/* Delete an IVL L2 unicast entry with MAC: 00:01:02:03:04:05, VID = 1 */

rtk_l2_ucastAddr_t l2Addr;

osal_memset(&l2Addr, 0x00, sizeof(rtk_l2_ucastAddr_t));
l2Addr.vid  = 1;
L2Addr.mac.octet[0] = 0x00;
L2Addr.mac.octet[1] = 0x01;
L2Addr.mac.octet[2] = 0x02;
L2Addr.mac.octet[3] = 0x03;
L2Addr.mac.octet[4] = 0x04;
L2Addr.mac.octet[5] = 0x05;
L2Addr.flags = (RTK_L2_UCAST_FLAG_STATIC |
                RTK_L2_UCAST_FLAG_IVL );

rtk_l2_addr_del(&l2Addr);
```

Realtek confidential for tenda

```
/* Add an SVL L2 multicast entry with MAC: 01:80:C2:11:22:33, FID = 3 */
/* port mask = UTP port 0~3 */

rtk_l2_mcastAddr_t l2Mcast;

osal_memset(&l2Mcast, 0x00, sizeof(rtk_l2_mcastAddr_t));
l2Mcast.fid = 3;
l2Mcast.mac.octet[0] = 0x01;
l2Mcast.mac.octet[1] = 0x80;
l2Mcast.mac.octet[2] = 0xC2;
l2Mcast.mac.octet[3] = 0x11;
l2Mcast.mac.octet[4] = 0x22;
l2Mcast.mac.octet[5] = 0x33;
RTK_PORTMASK_RESET(&ipmcAddr.portmask);
```

```
RTK_PORTMASK_PORT_SET(&ipmcAddr.portmask, RTK_PORT_UTP0);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP1);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP2);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP3);

rtk_l2_mcastAddr_add(&l2Mcast);
```

```
/* Delete an SVL L2 multicast entry with MAC: 01:80:C2:11:22:33, FID = 3
*/

tk_l2_mcastAddr_t l2Mcast;

osal_memset(&l2Mcast, 0x00, sizeof(rtk_l2_mcastAddr_t));
l2Mcast.fid = 3;
l2Mcast.mac.octet[0] = 0x01;
l2Mcast.mac.octet[1] = 0x80;
l2Mcast.mac.octet[2] = 0xC2;
l2Mcast.mac.octet[3] = 0x11;
l2Mcast.mac.octet[4] = 0x22;
l2Mcast.mac.octet[5] = 0x33;

rtk_l2_mcastAddr_del(&l2Mcast);
```

Realtek confidential for tenda

```
/* Add an IP multicast entry with DIP = 224.1.2.3, SIP = 10.1.1.1 */
/* port mask = UTP port 0~3 */

rtk_l2_ipMcastAddr_t ipmcAddr;

osal_memset(&ipmcAddr, 0x00, sizeof(rtk_l2_ipMcastAddr_t));
ipmcAddr.dip = 0xE0010203;
ipmcAddr.sip = 0x0A010101;
ipmcAddr.flags |= RTK_L2_IPMCAST_FLAG_STATIC;
```

```
RTK_PORTMASK_RESET(&ipmcAddr.portmask);
RTK_PORTMASK_PORT_SET(&ipmcAddr.portmask, RTK_PORT_UTP0);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP1);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP2);
RTK_PORTMASK_OR (&ipmcAddr.portmask, RTK_PORT_UTP3);

rtk_l2_ipMcastAddr_add(&l2Mcast);
```

```
/* Delete an IP multicast entry with DIP = 224.1.2.3, SIP = 10.1.1.1 */

rtk_l2_ipMcastAddr_t ipmcAddr;

osal_memset(&ipmcAddr, 0x00, sizeof(rtk_l2_ipMcastAddr_t));
ipmcAddr.dip = 0xE0010203;
ipmcAddr.sip = 0x0A010101;
ipmcAddr.flags |= RTK_L2_IPMCAST_FLAG_STATIC;

rtk_l2_ipMcastAddr_del(&l2Mcast);
```

Realtek confidential for tenda

6.3. Get an Exist Entry

Like Addition and deletion, RTL9607C SDK also provides API to get an exist entry in L2 Table.

```
/* Get an IVL L2 unicast entry with MAC: 00:01:02:03:04:05, VID = 1 */

int32 ret;
rtk_l2_ucastAddr_t l2Addr;

osal_memset(&l2Addr, 0x00, sizeof(rtk_l2_ucastAddr_t));
l2Addr.vid  = 1;
L2Addr.mac.octet[0] = 0x00;
L2Addr.mac.octet[1] = 0x01;
L2Addr.mac.octet[2] = 0x02;
L2Addr.mac.octet[3] = 0x03;
```

```
L2Addr.mac.octet[4] = 0x04;
L2Addr.mac.octet[5] = 0x05;
L2Addr.flags = (RTK_L2_UCAST_FLAG_STATIC |
                RTK_L2_UCAST_FLAG_IVL );

ret = rtk_l2_addr_get(&l2Addr);
if(ret == RT_ERR_OK)
    /* Entry found */
else
    /* Entry not found */
```

```
/* Get an SVL L2 multicast entry with MAC: 01:80:C2:11:22:33, FID = 3 */

int32 ret;
rtk_l2_mcastAddr_t l2Mcast;

osal_memset(&l2Mcast, 0x00, sizeof(rtk_l2_mcastAddr_t));
l2Mcast.fid = 3;
l2Mcast.mac.octet[0] = 0x01;
l2Mcast.mac.octet[1] = 0x80;
l2Mcast.mac.octet[2] = 0xC2;
l2Mcast.mac.octet[3] = 0x11;
l2Mcast.mac.octet[4] = 0x22;
l2Mcast.mac.octet[5] = 0x33;

ret = rtk_l2_mcastAddr_get(&l2Mcast);
if(ret == RT_ERR_OK)
    /* Entry found */
else
    /* Entry not found */
```

Realtek confidential for tenda

```
/* Get an IP multicast entry with DIP = 224.1.2.3, SIP = 10.1.1.1 */

int32 ret;
```

```
rtk_l2_ipMcastAddr_t ipmcAddr;

osal_memset(&ipmcAddr, 0x00, sizeof(rtk_l2_ipMcastAddr_t));
ipmcAddr.dip = 0xE0010203;
ipmcAddr.sip = 0x0A010101;
ipmcAddr.flags |= RTK_L2_IPMCAST_FLAG_STATIC;

ret = rtk_l2_ipMcastAddr_get(&l2Mcast);
if(ret == RT_ERR_OK)
    /* Entry found */
else
    /* Entry not found */
```

6.4. Dump entire L2 Table

Section 6.3 gives the sample code for getting s specified entry. However, if users don't know the MAC/IP address of a L2 entry or they just want dump entire L2 table, the following sample can be used to do this.

Each entry in L2 tabled will be assigned an "address" which is from 0 to 2047. This address represents the physical position of an entry in L2 table. When user use API to dump L2 table, RTL9607C SDK API will request users to input an address and the API will return the next valid entry in L2 Table and its physical address. Users can use this returned address as next address input until the address returned by SDK API is smaller than input.

Realtek confidential for tenda

```
/* Dump L2 unicast entry */

int32 ret;
int32 address = 0;
int32 pre_addr;
rtk_l2_ucastAddr_t l2Addr;

osal_memset(&l2Addr, 0x00, sizeof(rtk_l2_ucastAddr_t));

do
{
   pre_addr = address;
    ret = rtk_l2_nextValidAddr_get(&address, &l2Addr), ret);
```

```
    if(ret != RT_ERR_OK)
        break;

    if(address < pre_addr)
        break;
}while(1)
```

```
/* Dump L2 multicast entry */

int32 ret;
int32 address = 0;
int32 pre_addr;
rtk_l2_mcastAddr_t l2Mcast;

osal_memset(&l2Mcast, 0x00, sizeof(rtk_l2_mcastAddr_t));

do
{
   pre_addr = address;
    ret = rtk_l2_nextValidMcastAddr_get(&address, &l2Mcast), ret);
    if(ret != RT_ERR_OK)
        break;

    if(address < pre_addr)
        break;
}while(1)
```

Realtek confidential for tenda

```
/* Dump L3 multicast entry */

int32 ret;
int32 address = 0;
int32 pre_addr;
rtk_l2_ipMcastAddr_t ipmcAddr;
```

```
osal_memset(&ipmcAddr, 0x00, sizeof(rtk_l2_ipMcastAddr_t));


do
{
   pre_addr = address;
    ret = rtk_l2_nextValidIpMcastAddr_get(&address, &ipmcAddr), ret);
    if(ret != RT_ERR_OK)
        break;

    if(address < pre_addr)
        break;
}while(1)
```

6.5. Learning Limitation & Action

RTL9607C provides port-based and system-based learning limitation. Learning over action can be configurable for both kinds of limitations.

```
/* Set port 2 learning limtation to 20 entries */
/* Set port 2 learning over action as drop */

rtk_l2_portLimitLearningCnt_set(2, 20);
rtk_l2_portLimitLearningCntAction_set(2, LIMIT_LEARN_CNT_ACTION_DROP);
```

```
/* Set system learning limtation to 1000 entries */
/* Set system learning over action as forward */

rtk_l2_limitLearningCnt_set(1000);
rtk_l2_limitLearningCntAction_set(LIMIT_LEARN_CNT_ACTION_FORWARD);
```

Realtek confidential for tenda

6.6. Ageing Timer

The ageing timer is the time period that a used dynamic entry can exist in L2 Table. When a dynamic L2 entry doesn't be referenced in ageing timer period, it may be deleted by RTL9607C.
The ageing timer configuration sample code is as following

```
/* Set Ageing timer to 300 seconds */

rtk_l2_aging_set(300);
```

6.7. Flush Control

RTL9607C SDK can flush dynamic & static entry in L2 Table.

```
/* Flush all dynamic entry at UTP port 0 */
rtk_l2_flushCfg_t cfg;
uint32 phyPortNum;

rtk_switch_phyPortId_get(RTK_PORT_UTP0, &phyPortNum);

cfg.flushByPort = 1;
cfg.port = phyPortNum;
cfg.flushDynamicAddr = 1;
cfg.flushStaticAddr = 0;
rtk_l2_ucastAddr_flush(&cfg);
```

Realtek confidential for tenda

6.8. Source Port Filtering

By default, RTL9607C will not forward packet to the port which it receives this packet. This function is called "source port filtering". RTL9607C SDK API provides API to enable/disable this function.

```

```

```
/* Disable UTP Port 2 & 3 source port filtering */

rtk_portmask_t portmask ;

RTK_PORTMASK_RESET(portmask);
RTK_PORTMASK_PORT_SET (&portmask, RTK_PORT_UTP2);
RTK_PORTMASK_PORT_SET (&portmask, RTK_PORT_UTP3);
rtk_l2_srcPortEgrFilterMask_set(&portmask);
```

Realtek confidential for tenda

Hsinchu 300, Taiwan, R.O.C.
Tel: 886-3-5780211 Fax: 886-3-5776047
www.realtek.com

Realtek confidential for tenda

全文阅读已结束，下载本文需要使用

300 积分

下载此文档

阅读了该文档的用户还阅读了这些文档

RTL9607C_L2_Table

发表评论

验证码： 换一张

匿名评论

提交

道客巴巴网站 版权所有 | ©2008-2025 | 网站备案：京ICP备18056798号-1 京公网安备11010802036365号