



REALTEK

RTL9600

API Document

Ver 1.0.3

Jan. 14, 2013



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

© 2010 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

This document is intended for use by the system engineer when integrating with Realtek Switch Software SDK. 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.

REVISION HISTORY

Revision	Date	Description	Author
1.0.3	2013/01/14	Update New version	Realtek
1.0.2	2012/04/28	Add rate/led/dot1x API Modify pon mac API	Realtek
1.0.1	2012/03/12	Add GPON API	Realtek
1.0.0	2012/02/17	First Release	Realtek

RTL9600	1
API Document	1
Ver 1.0.3	1
COPYRIGHT	2
TRADEMARKS	2
DISCLAIMER	2
USING THIS DOCUMENT	2
REVISION HISTORY	3
1. Overview	1
2. Module ACL API	2
2.1. List of Symbols	2
2.2. rtk_acl_init	3
2.3. rtk_acl_template_set	3
2.4. rtk_acl_template_get	3
2.5. rtk_acl_fieldSelect_set	4
2.6. rtk_acl_fieldSelect_get	4
2.7. rtk_acl_igrRuleEntry_get	5
2.8. rtk_acl_igrRuleField_add	5
2.9. rtk_acl_igrRuleEntry_add	6
2.10. rtk_acl_igrRuleEntry_del	6
2.11. rtk_acl_igrRuleEntry_delAll	7
2.12. rtk_acl_igrUnmatchAction_set	7
2.13. rtk_acl_igrUnmatchAction_get	8
2.14. rtk_acl_igrState_set	8
2.15. rtk_acl_igrState_get	9
2.16. rtk_acl_ipRange_set	9
2.17. rtk_acl_ipRange_get	10

2.18.	rtk_acl_vidRange_set.....	10
2.19.	rtk_acl_vidRange_get	10
2.20.	rtk_acl_portRange_set	11
2.21.	rtk_acl_portRange_get.....	11
2.22.	rtk_acl_packetLengthRange_set.....	12
2.23.	rtk_acl_packetLengthRange_get.....	12
2.24.	rtk_acl_igrRuleMode_set.....	13
2.25.	rtk_acl_igrRuleMode_get	13
2.26.	rtk_acl_igrPermitState_set.....	14
2.27.	rtk_acl_igrPermitState_get	14
3.	Module Classification API	15
3.1.	List of Symbols	15
3.2.	rtk_classify_init	15
3.3.	rtk_classify_cfgEntry_add	16
3.4.	rtk_classify_cfgEntry_get	16
3.5.	rtk_classify_cfgEntry_del	17
3.6.	rtk_classify_field_add.....	17
3.7.	rtk_classify_unmatchAction_set.....	18
3.8.	rtk_classify_unmatchAction_get	18
3.9.	rtk_classify_portRange_set.....	19
3.10.	rtk_classify_portRange_get	19
3.11.	rtk_classify_ipRange_set.....	20
3.12.	rtk_classify_ipRange_get.....	20
3.13.	rtk_classify_cfSel_set	21
3.14.	rtk_classify_cfSel_get.....	21
3.15.	rtk_classify_cfPri2Dscp_set	22
3.16.	rtk_classify_cfPri2Dscp_get	22
3.17.	rtk_classify_permit_sel_get	23

3.18. rtk_classify_permit_sel_set.....	23
3.19. rtk_classify_us1pRemarkPrior_set	23
3.20. rtk_classify_us1pRemarkPrior_get.....	24
4. Module CPU Tag API	24
4.1. List of Symbols.....	24
4.2. rtk_cpu_init.....	25
4.3. rtk_cpu_awarePortMask_set.....	25
4.4. rtk_cpu_awarePortMask_get	26
4.5. rtk_cpu_tagFormat_set	26
4.6. rtk_cpu_tagFormat_get.....	26
4.7. rtk_cpu_trapInsertTag_set	27
4.8. rtk_cpu_trapInsertTag_get.....	27
5. Module 802.1x API	28
5.1. List of Symbols.....	28
5.2. rtk_dot1x_init.....	29
5.3. rtk_dot1x_unauthPacketOper_get.....	29
5.4. rtk_dot1x_unauthPacketOper_set	30
5.5. rtk_dot1x_portBasedEnable_get.....	30
5.6. rtk_dot1x_portBasedEnable_set	31
5.7. rtk_dot1x_portBasedAuthStatus_get	31
5.8. rtk_dot1x_portBasedAuthStatus_set.....	32
5.9. rtk_dot1x_portBasedDirection_get.....	33
5.10. rtk_dot1x_portBasedDirection_set	33
5.11. rtk_dot1x_macBasedEnable_get.....	34
5.12. rtk_dot1x_macBasedEnable_set	34
5.13. rtk_dot1x_macBasedDirection_get.....	35
5.14. rtk_dot1x_macBasedDirection_set	36
5.15. rtk_dot1x_guestVlan_get.....	36

5.16. rtk_dot1x_guestVlan_set	37
5.17. rtk_dot1x_guestVlanBehavior_get	37
5.18. rtk_dot1x_guestVlanBehavior_set.....	38
5.19. rtk_dot1x_trapPri_get	38
5.20. rtk_dot1x_trapPri_set.....	39
6. Module GPIO API.....	39
6.1. List of Symbols.....	39
6.2. rtk_gpio_init	40
6.3. rtk_gpio_state_set.....	40
6.4. rtk_gpio_state_get.....	40
6.5. rtk_gpio_mode_set.....	41
6.6. rtk_gpio_mode_get	41
6.7. rtk_gpio_databit_get.....	42
6.8. rtk_gpio_databit_set.....	42
6.9. rtk_gpio_intr_set.....	43
6.10. rtk_gpio_intr_get	43
7. Module GPON Mac API	43
7.1. List of Symbols.....	44
7.2. rtk_gpon_driver_initialize	46
7.3. rtk_gpon_driver_deInitialize.....	46
7.4. rtk_gpon_device_initialize.....	46
7.5. rtk_gpon_device_deInitialize.....	47
7.6. rtk_gpon_evtHdlStateChange_reg.....	47
7.7. rtk_gpon_evtHdlDsFecChange_reg.....	48
7.8. rtk_gpon_evtHdlUsFecChange_reg.....	48
7.9. rtk_gpon_evtHdlUsPloamUrgEmpty_reg.....	49
7.10. rtk_gpon_evtHdlUsPloamNrmEmpty_reg.....	49
7.11. rtk_gpon_evtHdlPloam_reg.....	50

7.12. rtk_gpon_evtHdlOmci_reg	50
7.13. rtk_gpon_callbackQueryAesKey_reg	50
7.14. rtk_gpon_evtHdlAlarm_reg	51
7.15. rtk_gpon_serialNumber_set	51
7.16. rtk_gpon_serialNumber_get	52
7.17. rtk_gpon_password_set	52
7.18. rtk_gpon_password_get	53
7.19. rtk_gpon_parameter_set	53
7.20. rtk_gpon_parameter_get	53
7.21. rtk_gpon_activate	54
7.22. rtk_gpon_deActivate	54
7.23. rtk_gpon_ponStatus_get	55
7.24. rtk_gpon_isr_entry	55
7.25. rtk_gpon_tcont_create	55
7.26. rtk_gpon_tcont_destroy	56
7.27. rtk_gpon_tcont_get	56
7.28. rtk_gpon_dsFlow_set	57
7.29. rtk_gpon_dsFlow_get	57
7.30. rtk_gpon_usFlow_set	58
7.31. rtk_gpon_usFlow_get	58
7.32. rtk_gpon_ploam_send	58
7.33. rtk_gpon_broadcastPass_set	59
7.34. rtk_gpon_broadcastPass_get	59
7.35. rtk_gpon_nonMcastPass_set	60
7.36. rtk_gpon_nonMcastPass_get	60
7.37. rtk_gpon_multicastAddrCheck_set	60
7.38. rtk_gpon_multicastAddrCheck_get	61
7.39. rtk_gpon_macFilterMode_set	61

7.40. rtk_gpon_macFilterMode_get.....	62
7.41. rtk_gpon_mcForceMode_set	62
7.42. rtk_gpon_mcForceMode_get.....	63
7.43. rtk_gpon_macEntry_add.....	63
7.44. rtk_gpon_macEntry_del.....	63
7.45. rtk_gpon_macEntry_get.....	64
7.46. rtk_gpon_rdi_set.....	64
7.47. rtk_gpon_rdi_get.....	65
7.48. rtk_gpon_powerLevel_set	65
7.49. rtk_gpon_powerLevel_get	65
7.50. rtk_gpon_alarmStatus_get	66
7.51. rtk_gpon_globalCounter_get	66
7.52. rtk_gpon_tcontCounter_get	67
7.53. rtk_gpon_flowCounter_get.....	67
7.54. rtk_gpon_version_get	68
7.55. rtk_gpon_txForceLaser_set.....	68
7.56. rtk_gpon_txForceLaser_get	69
7.57. rtk_gpon_txForceIdle_set	69
7.58. rtk_gpon_txForceIdle_get.....	69
7.59. rtk_gpon_dsFecSts_get.....	70
7.60. rtk_gpon_version_show.....	70
7.61. rtk_gpon_devInfo_show	71
7.62. rtk_gpon_gtc_show.....	71
7.63. rtk_gpon_tcont_show.....	71
7.64. rtk_gpon_dsFlow_show.....	72
7.65. rtk_gpon_usFlow_show.....	72
7.66. rtk_gpon_macTable_show.....	73
7.67. rtk_gpon_globalCounter_show.....	73

7.68.	rtk_gpon_tcontCounter_show.....	73
7.69.	rtk_gpon_flowCounter_show	74
7.70.	rtk_gpon_omci_tx.....	74
7.71.	rtk_gpon_omci_rx.....	75
7.72.	rtk_gpon_test_set.....	75
7.73.	rtk_gpon_test_get.....	76
7.74.	rtk_gpon_unit_test.....	76
7.75.	rtk_gpon_initial.....	76
7.76.	rtk_gpon_deinitial.....	77
7.77.	rtk_gpon_debug_set.....	77
7.78.	rtk_gpon_autoTcont_set	78
7.79.	rtk_gpon_autoTcont_get	78
7.80.	rtk_gpon_autoBoh_set	78
7.81.	rtk_gpon_autoBoh_get.....	79
7.82.	rtk_gpon_eqdOffset_set.....	79
7.83.	rtk_gpon_eqdOffset_get	80
7.84.	rtk_gpon_pktGen_cfg_set.....	80
7.85.	rtk_gpon_pktGen_buf_set.....	81
8.	Module I2C API	81
8.1.	List of Symbols.....	81
8.2.	rtk_i2c_init.....	82
8.3.	rtk_i2c_enable_set	82
8.4.	rtk_i2c_enable_get.....	83
8.5.	rtk_i2c_width_set.....	83
8.6.	rtk_i2c_width_get	84
8.7.	rtk_i2c_write	84
8.8.	rtk_i2c_read	85
9.	Module Init API.....	85

9.1.	List of Symbols.....	85
9.2.	rtk_init.....	86
9.3.	rtk_deinit.....	86
10.	Module Interrupt APIs.....	87
10.1.	List of Symbols.....	87
10.2.	rtk_intr_init.....	87
10.3.	rtk_intr_polarity_set.....	88
10.4.	rtk_intr_polarity_get.....	88
10.5.	rtk_intr_imr_set	89
10.6.	rtk_intr_imr_get.....	89
10.7.	rtk_intr_ims_get.....	90
10.8.	rtk_intr_ims_clear.....	90
10.9.	rtk_intr_speedChangeStatus_get.....	91
10.10.	rtk_intr_speedChangeStatus_clear.....	91
10.11.	rtk_intr_linkupStatus_get.....	91
10.12.	rtk_intr_linkupStatus_clear.....	92
10.13.	rtk_intr_linkdownStatus_get.....	92
10.14.	rtk_intr_linkdownStatus_clear.....	93
10.15.	rtk_intr_gphyStatus_get.....	93
10.16.	rtk_intr_gphyStatus_clear	93
10.17.	rtk_intr_imr_restore	94
11.	Module IRQ API	94
11.1.	List of Symbols.....	94
11.2.	rtk_switch_irq_init.....	95
11.3.	rtk_switch_irq_exit	95
11.4.	rtk_irq_isr_register.....	95
11.5.	rtk_irq_isr_unregister.....	96
12.	Module L2 API.....	96

12.1.	List of Symbols	96
12.2.	rtk_l2_init	98
12.3.	rtk_l2_flushLinkDownPortAddrEnable_get	98
12.4.	rtk_l2_flushLinkDownPortAddrEnable_set	99
12.5.	rtk_l2_unicastAddr_flush.....	99
12.6.	rtk_l2_table_clear	100
12.7.	rtk_l2_limitLearningOverStatus_get	100
12.8.	rtk_l2_limitLearningOverStatus_clear.....	101
12.9.	rtk_l2_learningCnt_get	101
12.10.	rtk_l2_limitLearningCnt_get	102
12.11.	rtk_l2_limitLearningCnt_set.....	102
12.12.	rtk_l2_limitLearningCntAction_get.....	103
12.13.	rtk_l2_limitLearningCntAction_set.....	103
12.14.	rtk_l2_portLimitLearningOverStatus_get.....	104
12.15.	rtk_l2_portLimitLearningOverStatus_clear.....	104
12.16.	rtk_l2_portLearningCnt_get	105
12.17.	rtk_l2_portLimitLearningCnt_get.....	105
12.18.	rtk_l2_portLimitLearningCnt_set	106
12.19.	rtk_l2_portLimitLearningCntAction_get.....	106
12.20.	rtk_l2_portLimitLearningCntAction_set	107
12.21.	rtk_l2_aging_get	107
12.22.	rtk_l2_aging_set.....	108
12.23.	rtk_l2_portAgingEnable_get.....	108
12.24.	rtk_l2_portAgingEnable_set	109
12.25.	rtk_l2_lookupMissAction_get	109
12.26.	rtk_l2_lookupMissAction_set.....	110
12.27.	rtk_l2_portLookupMissAction_get.....	111
12.28.	rtk_l2_portLookupMissAction_set	112

12.29. rtk_l2_lookupMissFloodPortMask_get	112
12.30. rtk_l2_lookupMissFloodPortMask_set.....	113
12.31. rtk_l2_lookupMissFloodPortMask_add	114
12.32. rtk_l2_lookupMissFloodPortMask_del	114
12.33. rtk_l2_newMacOp_get	115
12.34. rtk_l2_newMacOp_set.....	115
12.35. rtk_l2_nextValidAddr_get.....	116
12.36. rtk_l2_nextValidAddrOnPort_get.....	117
12.37. rtk_l2_nextValidMcastAddr_get	118
12.38. rtk_l2_nextValidIpMcastAddr_get	118
12.39. rtk_l2_nextValidEntry_get	119
12.40. rtk_l2_addr_add	119
12.41. rtk_l2_addr_del.....	120
12.42. rtk_l2_addr_get.....	120
12.43. rtk_l2_addr_delAll.....	121
12.44. rtk_l2_mcastAddr_add.....	121
12.45. rtk_l2_mcastAddr_del	122
12.46. rtk_l2_mcastAddr_get	122
12.47. rtk_l2_illegalPortMoveAction_get	123
12.48. rtk_l2_illegalPortMoveAction_set.....	123
12.49. rtk_l2_ipmcMode_get.....	124
12.50. rtk_l2_ipmcMode_set	125
12.51. rtk_l2_ipmcGroupLookupMissHash_get.....	125
12.52. rtk_l2_ipmcGroupLookupMissHash_set.....	126
12.53. rtk_l2_ipmcGroup_add.....	126
12.54. rtk_l2_ipmcGroup_del.....	127
12.55. rtk_l2_ipmcGroup_get.....	127
12.56. rtk_l2_portIpmcAction_get.....	128

12.57. rtk_l2_portIpmcAction_set	128
12.58. rtk_l2_ipMcastAddr_add	129
12.59. rtk_l2_ipMcastAddr_del	129
12.60. rtk_l2_ipMcastAddr_get	130
12.61. rtk_l2_srcPortEgrFilterMask_get	130
12.62. rtk_l2_srcPortEgrFilterMask_set	131
12.63. rtk_l2_extPortEgrFilterMask_get	131
12.64. rtk_l2_extPortEgrFilterMask_set	132
13. Module LED APIs	132
13.1. List of Symbols	132
13.2. rtk_led_init	133
13.3. rtk_led_operation_get	133
13.4. rtk_led_operation_set	134
13.5. rtk_led_serialMode_get	134
13.6. rtk_led_serialMode_set	134
13.7. rtk_led_blinkRate_get	135
13.8. rtk_led_blinkRate_set	135
13.9. rtk_led_config_set	136
13.10. rtk_led_config_get	136
13.11. rtk_led_modeForce_get	137
13.12. rtk_led_modeForce_set	138
13.13. rtk_led_parallelEnable_get	138
13.14. rtk_led_parallelEnable_set	139
14. Module Mirror API	139
14.1. List of Symbols	139
14.2. rtk_mirror_init	140
14.3. rtk_mirror_portBased_set	140
14.4. rtk_mirror_portBased_get	141

14.5.	rtk_mirror_portIso_set.....	141
14.6.	rtk_mirror_portIso_get.....	142
15.	Module OAM API.....	142
15.1.	List of Symbols.....	142
15.2.	rtk_oam_init.....	143
15.3.	rtk_oam_parserAction_set.....	143
15.4.	rtk_oam_parserAction_get.....	143
15.5.	rtk_oam_multiplexerAction_set	144
15.6.	rtk_oam_multiplexerAction_get.....	144
16.	Module PON MAC API.....	145
16.1.	List of Symbols.....	145
16.2.	rtk_ponmac_init.....	145
16.3.	rtk_ponmac_queue_add	146
16.4.	rtk_ponmac_queue_get.....	146
16.5.	rtk_ponmac_queue_del.....	147
16.6.	rtk_ponmac_flow2Queue_set	147
16.7.	rtk_ponmac_flow2Queue_get.....	148
16.8.	rtk_ponmac_mode_set.....	148
16.9.	rtk_ponmac_mode_get.....	149
17.	Module Port API.....	149
17.1.	List of Symbols.....	150
17.2.	rtk_port_init	151
17.3.	rtk_port_link_get	152
17.4.	rtk_port_speedDuplex_get.....	152
17.5.	rtk_port_flowctrl_get.....	153
17.6.	rtk_port_phyAutoNegoEnable_get.....	154
17.7.	rtk_port_phyAutoNegoEnable_set	154
17.8.	rtk_port_phyAutoNegoAbility_get.....	155

17.9. rtk_port_phyAutoNegoAbility_set	155
17.10. rtk_port_phyForceModeAbility_get	156
17.11. rtk_port_phyForceModeAbility_set.....	156
17.12. rtk_port_phyReg_get	157
17.13. rtk_port_phyReg_set.....	158
17.14. rtk_port_phyMasterSlave_get.....	159
17.15. rtk_port_phyMasterSlave_set	159
17.16. rtk_port_phyTestMode_get.....	160
17.17. rtk_port_phyTestMode_set.....	160
17.18. rtk_port_cpuPortId_get.....	161
17.19. rtk_port_isolation_get.....	161
17.20. rtk_port_isolation_set	162
17.21. rtk_port_isolationExt_get	162
17.22. rtk_port_isolationExt_set.....	163
17.23. rtk_port_isolationL34_get.....	164
17.24. rtk_port_isolationL34_set	164
17.25. rtk_port_isolationExtL34_get	165
17.26. rtk_port_isolationExtL34_set	165
17.27. rtk_port_isolationEntry_get	166
17.28. rtk_port_isolationEntry_set.....	167
17.29. rtk_port_isolationEntryExt_get.....	167
17.30. rtk_port_isolationEntryExt_set	168
17.31. rtk_port_isolationCtagPktConfig_get	169
17.32. rtk_port_isolationCtagPktConfig_set.....	169
17.33. rtk_port_isolationL34PktConfig_get	169
17.34. rtk_port_isolationL34PktConfig_set.....	170
17.35. rtk_port_isolationIpmcLeaky_get.....	170
17.36. rtk_port_isolationIpmcLeaky_set	171

17.37. rtk_port_isolationPortLeaky_get	171
17.38. rtk_port_isolationPortLeaky_set.....	172
17.39. rtk_port_isolationLeaky_get	172
17.40. rtk_port_isolationLeaky_set	173
17.41. rtk_port_macRemoteLoopbackEnable_get.....	173
17.42. rtk_port_macRemoteLoopbackEnable_set.....	174
17.43. rtk_port_macLocalLoopbackEnable_get.....	175
17.44. rtk_port_macLocalLoopbackEnable_set	175
17.45. rtk_port_adminEnable_get.....	176
17.46. rtk_port_adminEnable_set	176
17.47. rtk_port_specialCongest_get.....	177
17.48. rtk_port_specialCongest_set.....	177
17.49. rtk_port_specialCongestStatus_get.....	178
17.50. rtk_port_specialCongestStatus_clear.....	178
17.51. rtk_port_greenEnable_get.....	179
17.52. rtk_port_greenEnable_set	179
17.53. rtk_port_phyCrossOverMode_get	180
17.54. rtk_port_phyCrossOverMode_set.....	180
17.55. rtk_port_enhancedFid_get	181
17.56. rtk_port_enhancedFid_set.....	181
17.57. rtk_port_rtctResult_get	182
17.58. rtk_port_rtct_start	182
17.59. rtk_port_macForceAbility_set	183
17.60. rtk_port_macForceAbility_get.....	183
17.61. rtk_port_macForceAbilityState_set	184
17.62. rtk_port_macForceAbilityState_get.....	184
17.63. rtk_port_macExtMode_set.....	185
17.64. rtk_port_macExtMode_get	185

17.65. rtk_port_macExtRgmiiDelay_set	186
17.66. rtk_port_macExtRgmiiDelay_get	186
18. Module QoS API	187
18.1. List of Symbols	187
18.2. rtk_qos_init	188
18.3. rtk_qos_priSelGroup_get	189
18.4. rtk_qos_priSelGroup_set	189
18.5. rtk_qos_portPri_get	190
18.6. rtk_qos_portPri_set	190
18.7. rtk_qos_dscpPriRemapGroup_get	191
18.8. rtk_qos_dscpPriRemapGroup_set	191
18.9. rtk_qos_1pPriRemapGroup_get	192
18.10. rtk_qos_1pPriRemapGroup_set	193
18.11. rtk_qos_priMap_set	193
18.12. rtk_qos_priMap_get	194
18.13. rtk_qos_portPriMap_get	195
18.14. rtk_qos_portPriMap_set	195
18.15. rtk_qos_1pRemarkEnable_get	196
18.16. rtk_qos_1pRemarkEnable_set	196
18.17. rtk_qos_1pRemarkGroup_get	197
18.18. rtk_qos_1pRemarkGroup_set	198
18.19. rtk_qos_dscpRemarkEnable_get	198
18.20. rtk_qos_dscpRemarkEnable_set	199
18.21. rtk_qos_dscpRemarkGroup_get	199
18.22. rtk_qos_dscpRemarkGroup_set	200
18.23. rtk_qos_portDscpRemarkSrcSel_get	201
18.24. rtk_qos_portDscpRemarkSrcSel_set	201
18.25. rtk_qos_dscp2DscpRemarkGroup_get	202

18.26. rtk_qos_dscp2DscpRemarkGroup_set.....	203
18.27. rtk_qos_fwd2CpuPriRemap_get.....	203
18.28. rtk_qos_fwd2CpuPriRemap_set	204
18.29. rtk_qos_schedulingQueue_get.....	204
18.30. rtk_qos_schedulingQueue_set	205
18.31. rtk_qos_portPriSelGroup_get	206
18.32. rtk_qos_portPriSelGroup_set.....	206
19. Module Rate API.....	207
19.1. List of Symbols.....	207
19.2. rtk_rate_init.....	208
19.3. rtk_rate_portIgrBandwidthCtrlRate_get	208
19.4. rtk_rate_portIgrBandwidthCtrlRate_set.....	209
19.5. rtk_rate_portIgrBandwidthCtrlIncludeIfg_get.....	209
19.6. rtk_rate_portIgrBandwidthCtrlIncludeIfg_set	210
19.7. rtk_rate_portEgrBandwidthCtrlRate_get	210
19.8. rtk_rate_portEgrBandwidthCtrlRate_set	211
19.9. rtk_rate_egrBandwidthCtrlIncludeIfg_get.....	211
19.10. rtk_rate_egrBandwidthCtrlIncludeIfg_set	212
19.11. rtk_rate_portEgrBandwidthCtrlIncludeIfg_get.....	212
19.12. rtk_rate_portEgrBandwidthCtrlIncludeIfg_set	213
19.13. rtk_rate_egrQueueBwCtrlEnable_get.....	214
19.14. rtk_rate_egrQueueBwCtrlEnable_set	214
19.15. rtk_rate_egrQueueBwCtrlMeterIdx_get	215
19.16. rtk_rate_egrQueueBwCtrlMeterIdx_set	216
19.17. rtk_rate_stormControlMeterIdx_get	216
19.18. rtk_rate_stormControlMeterIdx_set	217
19.19. rtk_rate_stormControlPortEnable_get	218
19.20. rtk_rate_stormControlPortEnable_set.....	219

19.21. rtk_rate_stormControlEnable_get.....	219
19.22. rtk_rate_stormControlEnable_set	220
19.23. rtk_rate_stormBypass_set.....	221
19.24. rtk_rate_stormBypass_get.....	222
19.25. rtk_rate_shareMeter_set.....	224
19.26. rtk_rate_shareMeter_get	225
19.27. rtk_rate_shareMeterBucket_set	225
19.28. rtk_rate_shareMeterBucket_get	226
19.29. rtk_rate_shareMeterExceed_get	226
19.30. rtk_rate_shareMeterExceed_clear.....	227
20. Module RLDP and RLPP API.....	227
20.1. List of Symbols.....	227
20.2. rtk_rldp_init	228
20.3. rtk_rldp_config_set.....	228
20.4. rtk_rldp_config_get	229
20.5. rtk_rldp_portConfig_set.....	229
20.6. rtk_rldp_portConfig_get	230
20.7. rtk_rldp_status_get.....	230
20.8. rtk_rldp_portStatus_get.....	231
20.9. rtk_rldp_portStatus_clear.....	231
20.10. rtk_rlpp_init	232
20.11. rtk_rlpp_trapType_set.....	232
20.12. rtk_rlpp_trapType_get	232
21. Module Security API.....	233
21.1. List of Symbols.....	233
21.2. rtk_sec_init.....	233
21.3. rtk_sec_portAttackPreventState_get.....	234
21.4. rtk_sec_portAttackPreventState_set	234

21.5.	rtk_sec_attackPrevent_get	235
21.6.	rtk_sec_attackPrevent_set.....	235
21.7.	rtk_sec_attackFloodThresh_get	236
21.8.	rtk_sec_attackFloodThresh_set.....	237
22.	Module Statistic API	237
22.1.	List of Symbols.....	237
22.2.	rtk_stat_init	238
22.3.	rtk_stat_global_reset.....	239
22.4.	rtk_stat_port_reset	239
22.5.	rtk_stat_log_reset.....	240
22.6.	rtk_stat_rstCntValue_set.....	240
22.7.	rtk_stat_rstCntValue_get	240
22.8.	rtk_stat_global_get.....	241
22.9.	rtk_stat_global_getAll	241
22.10.	rtk_stat_port_get	242
22.11.	rtk_stat_port_getAll	243
22.12.	rtk_stat_log_get	243
22.13.	rtk_stat_logCtrl_set.....	244
22.14.	rtk_stat_logCtrl_get	244
22.15.	rtk_stat_mibCntMode_get	245
22.16.	rtk_stat_mibCntMode_set.....	245
22.17.	rtk_stat_mibLatchTimer_get.....	245
22.18.	rtk_stat_mibLatchTimer_set	246
22.19.	rtk_stat_mibSyncMode_get	246
22.20.	rtk_stat_mibSyncMode_set	247
22.21.	rtk_stat_mibCntTagLen_get	247
22.22.	rtk_stat_mibCntTagLen_set.....	248
22.23.	rtk_stat_pktInfo_get.....	248

23.	Module STP APIs.....	249
23.1.	List of Symbols.....	249
23.2.	rtk_stp_init.....	249
23.3.	rtk_stp_mstpState_get.....	249
23.4.	rtk_stp_mstpState_set	250
24.	Module SVLAN API.....	251
24.1.	List of Symbols.....	251
24.2.	rtk_svlan_init	252
24.3.	rtk_svlan_create.....	253
24.4.	rtk_svlan_destroy.....	253
24.5.	rtk_svlan_portSvid_get.....	254
24.6.	rtk_svlan_portSvid_set	254
24.7.	rtk_svlan_servicePort_get.....	255
24.8.	rtk_svlan_servicePort_set	255
24.9.	rtk_svlan_memberPort_set	256
24.10.	rtk_svlan_memberPort_get.....	256
24.11.	rtk_svlan_tpidEntry_get.....	257
24.12.	rtk_svlan_tpidEntry_set	257
24.13.	rtk_svlan_priorityRef_set	258
24.14.	rtk_svlan_priorityRef_get	258
24.15.	rtk_svlan_memberPortEntry_set.....	259
24.16.	rtk_svlan_memberPortEntry_get	259
24.17.	rtk_svlan_ipmc2s_add	260
24.18.	rtk_svlan_ipmc2s_del	261
24.19.	rtk_svlan_ipmc2s_get	261
24.20.	rtk_svlan_l2mc2s_add	262
24.21.	rtk_svlan_l2mc2s_del	262
24.22.	rtk_svlan_l2mc2s_get	263

24.23. rtk_svlan_sp2c_add	264
24.24. rtk_svlan_sp2c_get	264
24.25. rtk_svlan_sp2c_del	265
24.26. rtk_svlan_dmacVidSelState_set	265
24.27. rtk_svlan_dmacVidSelState_get	266
24.28. rtk_svlan_unmatchAction_set.....	266
24.29. rtk_svlan_unmatchAction_get.....	267
24.30. rtk_svlan_untagAction_set	268
24.31. rtk_svlan_untagAction_get.....	268
24.32. rtk_svlan_c2s_add	269
24.33. rtk_svlan_c2s_del	270
24.34. rtk_svlan_c2s_get	270
24.35. rtk_svlan_trapPri_get.....	271
24.36. rtk_svlan_trapPri_set	271
24.37. rtk_svlan_deiKeepState_get	272
24.38. rtk_svlan_deiKeepState_set.....	272
24.39. rtk_svlan_lookupType_get	273
24.40. rtk_svlan_lookupType_set.....	273
24.41. rtk_svlan_sp2cUnmatchCtagging_get	274
24.42. rtk_svlan_sp2cUnmatchCtagging_set.....	274
24.43. rtk_svlan_priority_get.....	274
24.44. rtk_svlan_priority_set	275
24.45. rtk_svlan_fid_get	275
24.46. rtk_svlan_fid_set.....	276
24.47. rtk_svlan_fidEnable_get	277
24.48. rtk_svlan_fidEnable_set.....	277
24.49. rtk_svlan_enhancedFid_get	278
24.50. rtk_svlan_enhancedFid_set.....	278

24.51. rtk_svlan_enhancedFidEnable_get	279
24.52. rtk_svlan_enhancedFidEnable_set.....	279
24.53. rtk_svlan_dmacVidSelForcedState_set	280
24.54. rtk_svlan_dmacVidSelForcedState_get.....	280
25. Module Switch Global API	281
25.1. List of Symbols	281
25.2. rtk_switch_init	282
25.3. rtk_switch_deviceInfo_get.....	282
25.4. rtk_switch_phyPortId_get.....	282
25.5. rtk_switch_logicalPort_get.....	283
25.6. rtk_switch_port2PortMask_set	283
25.7. rtk_switch_port2PortMask_clear	284
25.8. rtk_switch_portIdInMask_check	284
25.9. rtk_switch_portMask_Clear.....	285
25.10. rtk_switch_allPortMask_set.....	285
25.11. rtk_switch_allExtPortMask_set	286
25.12. rtk_switch_nextPortInMask_get	286
25.13. rtk_switch_maxPktLenLinkSpeed_get	287
25.14. rtk_switch_maxPktLenLinkSpeed_set.....	287
25.15. rtk_switch_mgmtMacAddr_get	288
25.16. rtk_switch_mgmtMacAddr_set.....	288
25.17. rtk_switch_chip_reset	289
26. Module TIME API.....	289
26.1. List of Symbols	289
26.2. rtk_time_portTransparentEnable_set.....	290
26.3. rtk_time_portTransparentEnable_get.....	290
26.4. rtk_time_init.....	291
26.5. rtk_time_portPtpEnable_get	291

26.6. rtk_time_portPtpEnable_set.....	292
26.7. rtk_time_curTime_get	292
26.8. rtk_time_curTime_latch.....	293
26.9. rtk_time_refTime_get	293
26.10. rtk_time_refTime_set.....	294
26.11. rtk_time_frequency_set	294
26.12. rtk_time_frequency_get.....	294
26.13. rtk_time_ptpIgrMsgAction_set.....	295
26.14. rtk_time_ptpIgrMsgAction_get	295
26.15. rtk_time_ptpEgrMsgAction_set.....	296
26.16. rtk_time_ptpEgrMsgAction_get	296
26.17. rtk_time_meanPathDelay_set	297
26.18. rtk_time_meanPathDelay_get.....	297
26.19. rtk_time_rxTime_set.....	298
26.20. rtk_time_rxTime_get	298
27. Module TRAP API.....	299
27.1. List of Symbols.....	299
27.2. rtk_trap_init	300
27.3. rtk_trap_reasonTrapToCpuPriority_get.....	300
27.4. rtk_trap_reasonTrapToCpuPriority_set	301
27.5. rtk_trap_igmpCtrlPkt2CpuEnable_get	301
27.6. rtk_trap_igmpCtrlPkt2CpuEnable_set.....	302
27.7. rtk_trap_mldCtrlPkt2CpuEnable_get	302
27.8. rtk_trap_mldCtrlPkt2CpuEnable_set.....	303
27.9. rtk_trap_portIgmpMldCtrlPktAction_get	304
27.10. rtk_trap_portIgmpMldCtrlPktAction_set.....	304
27.11. rtk_trap_ipMcastPkt2CpuEnable_get	305
27.12. rtk_trap_ipMcastPkt2CpuEnable_set	305

27.13. rtk_trap_l2McastPkt2CpuEnable_get	306
27.14. rtk_trap_l2McastPkt2CpuEnable_set	306
27.15. rtk_trap_rmaAction_get	307
27.16. rtk_trap_rmaAction_set	307
27.17. rtk_trap_rmaPri_get	309
27.18. rtk_trap_rmaPri_set	309
27.19. rtk_trap_oamPduAction_get	310
27.20. rtk_trap_oamPduAction_set	310
27.21. rtk_trap_oamPduPri_get	311
27.22. rtk_trap_oamPduPri_set	311
28. Module TRUNK API	312
28.1. List of Symbols	312
28.2. rtk_trunk_init	312
28.3. rtk_trunk_distributionAlgorithm_get	313
28.4. rtk_trunk_distributionAlgorithm_set	314
28.5. rtk_trunk_port_get	314
28.6. rtk_trunk_port_set	315
28.7. rtk_trunk_hashMappingTable_get	315
28.8. rtk_trunk_hashMappingTable_set	316
28.9. rtk_trunk_mode_get	317
28.10. rtk_trunk_mode_set	317
28.11. rtk_trunk_trafficSeparate_get	318
28.12. rtk_trunk_trafficSeparate_set	318
28.13. rtk_trunk_portQueueEmpty_get	319
28.14. rtk_trunk_trafficPause_get	319
28.15. rtk_trunk_trafficPause_set	320
29. Module VLAN API	320
29.1. List of Symbols	320

29.2. rtk_vlan_init.....	322
29.3. rtk_vlan_create.....	322
29.4. rtk_vlan_destroy	323
29.5. rtk_vlan_destroyAll	323
29.6. rtk_vlan_fid_get.....	324
29.7. rtk_vlan_fid_set	324
29.8. rtk_vlan_fidMode_get	325
29.9. rtk_vlan_fidMode_set	326
29.10. rtk_vlan_port_get.....	326
29.11. rtk_vlan_port_set	327
29.12. rtk_vlan_extPort_get.....	328
29.13. rtk_vlan_extPort_set	328
29.14. rtk_vlan_stg_get.....	329
29.15. rtk_vlan_stg_set	329
29.16. rtk_vlan_priority_get	330
29.17. rtk_vlan_priority_set.....	330
29.18. rtk_vlan_priorityEnable_get	331
29.19. rtk_vlan_priorityEnable_set.....	331
29.20. rtk_vlan_policingEnable_get	332
29.21. rtk_vlan_policingEnable_set.....	332
29.22. rtk_vlan_policingMeterIdx_get	333
29.23. rtk_vlan_policingMeterIdx_set	334
29.24. rtk_vlan_portAcceptFrameType_get	334
29.25. rtk_vlan_portAcceptFrameType_set	335
29.26. rtk_vlan_vlanFunctionEnable_get	335
29.27. rtk_vlan_vlanFunctionEnable_set	336
29.28. rtk_vlan_portIgrFilterEnable_get	336
29.29. rtk_vlan_portIgrFilterEnable_set	337

29.30. rtk_vlan_leaky_get.....	338
29.31. rtk_vlan_leaky_set	339
29.32. rtk_vlan_portLeaky_get.....	341
29.33. rtk_vlan_portLeaky_set	342
29.34. rtk_vlan_keepType_get.....	342
29.35. rtk_vlan_keepType_set.....	343
29.36. rtk_vlan_portPvid_get	343
29.37. rtk_vlan_portPvid_set.....	344
29.38. rtk_vlan_extPortPvid_get	344
29.39. rtk_vlan_extPortPvid_set.....	345
29.40. rtk_vlan_protoGroup_get.....	345
29.41. rtk_vlan_protoGroup_set	346
29.42. rtk_vlan_portProtoVlan_get	347
29.43. rtk_vlan_portProtoVlan_set.....	347
29.44. rtk_vlan_tagMode_get	348
29.45. rtk_vlan_tagMode_set.....	348
29.46. rtk_vlan_portFid_get	349
29.47. rtk_vlan_portFid_set	350
29.48. rtk_vlan_portPriority_get.....	350
29.49. rtk_vlan_portPriority_set	351
29.50. rtk_vlan_portEgrTagKeepType_get	351
29.51. rtk_vlan_portEgrTagKeepType_set.....	352
29.52. rtk_vlan_transparentEnable_get	352
29.53. rtk_vlan_transparentEnable_set.....	353
29.54. rtk_vlan_cfiKeepEnable_get.....	353
29.55. rtk_vlan_cfiKeepEnable_set.....	354
29.56. rtk_vlan_reservedVidAction_get	354
29.57. rtk_vlan_reservedVidAction_set	355

29.58. rtk_vlan_tagModeIp4mc_get.....	355
29.59. rtk_vlan_tagModeIp4mc_set	356
29.60. rtk_vlan_tagModeIp6mc_get.....	357
29.61. rtk_vlan_tagModeIp6mc_set	357
30. Module EPON MAC APIs	358
30.1. List of Symbols.....	358
30.2. rtk_epon_init.....	359
30.3. rtk_epon_intrMask_get.....	359
30.4. rtk_epon_intrMask_set	360
30.5. rtk_epon_intr_get.....	360
30.6. rtk_epon_intr_disableAll	361
30.7. rtk_epon_llid_entry_set	361
30.8. rtk_epon_llid_entry_get.....	361
30.9. rtk_epon_forceLaserState_set.....	362
30.10. rtk_epon_forceLaserState_get	362
30.11. rtk_epon_laserTime_set.....	363
30.12. rtk_epon_laserTime_get	363
30.13. rtk_epon_syncTime_get.....	363
30.14. rtk_epon_registerReq_get.....	364
30.15. rtk_epon_registerReq_set	364
30.16. rtk_epon_churningKey_set.....	365
30.17. rtk_epon_churningKey_get.....	365
30.18. rtk_epon_usFecState_get	365
30.19. rtk_epon_usFecState_set	366
30.20. rtk_epon_dsFecState_get	366
30.21. rtk_epon_dsFecState_set	367
30.22. rtk_epon_mibCounter_get	367
30.23. rtk_epon_mibGlobal_reset.....	367

30.24. rtk_epon_mibLlidIdx_reset.....	368
30.25. rtk_epon_losState_get.....	368
31. Module L34 API.....	369
31.1. List of Symbols.....	369
31.2. rtk_l34_init.....	370
31.3. rtk_l34_netifTable_set.....	371
31.4. rtk_l34_netifTable_get.....	371
31.5. rtk_l34_arpTable_set	372
31.6. rtk_l34_arpTable_get.....	372
31.7. rtk_l34_arpTable_del.....	373
31.8. rtk_l34_pppoeTable_set.....	373
31.9. rtk_l34_pppoeTable_get	374
31.10. rtk_l34_routingTable_set.....	374
31.11. rtk_l34_routingTable_get	375
31.12. rtk_l34_routingTable_del	375
31.13. rtk_l34_nexthopTable_set.....	376
31.14. rtk_l34_nexthopTable_get	376
31.15. rtk_l34_extIntIPTable_set.....	377
31.16. rtk_l34_extIntIPTable_get	377
31.17. rtk_l34_extIntIPTable_del	378
31.18. rtk_l34_naptInboundTable_set	378
31.19. rtk_l34_naptInboundTable_get	379
31.20. rtk_l34_naptOutboundTable_set.....	379
31.21. rtk_l34_naptOutboundTable_get	380
31.22. rtk_l34_ipmcTransTable_set	380
31.23. rtk_l34_ipmcTransTable_get	381
31.24. rtk_l34_table_reset.....	381
31.25. rtk_l34_ipv6RoutingTable_set	382

31.26. rtk_l34_ipv6RoutingTable_get	382
31.27. rtk_l34_ipv6NeighborTable_set	383
31.28. rtk_l34_ipv6NeighborTable_get	383
31.29. rtk_l34_hsabMode_set	384
31.30. rtk_l34_hsabMode_get	384
31.31. rtk_l34_hsaData_get	385
31.32. rtk_l34_hsbData_get	385
31.33. rtk_l34_hsdState_set	385
31.34. rtk_l34_hsdState_get	386
31.35. rtk_l34_hsdState_get	386
31.36. rtk_l34_hwL4TrfWrkTbl_set	387
31.37. rtk_l34_hwL4TrfWrkTbl_get	387
31.38. rtk_l34_l4TrfTb_get	388
31.39. rtk_l34_hwL4TrfWrkTbl_Clear	388
31.40. rtk_l34_hwArpTrfWrkTbl_set	388
31.41. rtk_l34_hwArpTrfWrkTbl_get	389
31.42. rtk_l34_arpTrfTb_get	389
31.43. rtk_l34_hwArpTrfWrkTbl_Clear	390
32. Module L34 Lite API	390
32.1. List of Symbols	390
32.2. rtk_l34_lite_init	391
32.3. rtk_l34_netif_create	392
32.4. rtk_l34_netifPPPoE_set	392
32.5. rtk_l34_netifVlan_set	393
32.6. rtk_l34_netifRoutingState_set	393
32.7. rtk_l34_netifMtu_set	394
32.8. rtk_l34_netifIpaddr_set	394
32.9. rtk_l34_netifNat_set	395

32.10. rtk_l34_netifState_set	395
32.11. rtk_l34_netif_get.....	396
32.12. rtk_l34_netifGateway_set.....	396
32.13. rtk_l34_netif_set.....	397
32.14. rtk_l34_netif_del.....	397
32.15. rtk_l34_arp_add.....	397
32.16. rtk_l34_arp_get.....	398
32.17. rtk_l34_arp_del.....	398
32.18. rtk_l34_route_add.....	399
32.19. rtk_l34_route_del.....	399
32.20. rtk_l34_connectTrack_add	400
32.21. rtk_l34_connectTrack_get	400
32.22. rtk_l34_connectTrack_del	401
32.23. rtk_l34_globalCfg_get	401
32.24. rtk_l34_route6_add	401
32.25. rtk_l34_route6_del.....	402
32.26. rtk_l34_route6_get.....	402
32.27. rtk_l34_neigh6_add	403
32.28. rtk_l34_neigh6_del	403
32.29. rtk_l34_neigh6_get	404
32.30. rtk_l34_netifIp6addr_add	404
32.31. rtk_l34_netifIp6addr_del	405
32.32. rtk_l34_netifMac_set	405
33. Module L34 Binding API.....	406
33.1. List of Symbols.....	406
33.2. rtk_l34_bindingTable_set	407
33.3. rtk_l34_bindingTable_get.....	407
33.4. rtk_l34_bindingAction_set	408

33.5. rtk_l34_bindingAction_get.....	408
33.6. rtk_l34_wanTypeTable_set.....	409
33.7. rtk_l34_wanTypeTable_get	409
33.8. rtk_l34_portWanMap_set	410
33.9. rtk_l34_portWanMap_get.....	410
33.10. rtk_l34_globalState_set	411
33.11. rtk_l34_globalState_get.....	411
33.12. rtk_l34_lookupMode_set	412
33.13. rtk_l34_lookupMode_get.....	412
33.14. rtk_l34_lookupPortMap_set	413
33.15. rtk_l34_lookupPortMap_get	413
33.16. rtk_l34_wanRoutMode_set.....	414
33.17. rtk_l34_wanRoutMode_get	414
33.18. rtk_l34_arpTrfIndicator_get	415
33.19. rtk_l34_naptTrfIndicator_get.....	415
33.20. rtk_l34_pppTrfIndicator_get.....	416
33.21. rtk_l34_neighTrfIndicator_get.....	416
33.22. rtk_l34_naptTrfIndicator_get_all.....	417
33.23. rtk_l34_arpTrfIndicator_get_all	417

1. Overview

Description The file includes all high-layer API definition

Copyright © 2012 Realtek™ Semiconductor Corp. All rights reserved.

Realtek
CONFIDENTIAL
for Loso Technology, Inc

2. Module ACL API

Filename: acl.h

Description

The file includes the following modules and sub-modules

(1) ACL rule action configure and modification

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

2.1. List of Symbols

Here is a list of all functions and variables in this module

acl.h - Definition of ACL API
rtk_acl_init
rtk_acl_template_set
rtk_acl_template_get
rtk_acl_fieldSelect_set
rtk_acl_fieldSelect_get
rtk_acl_igrRuleEntry_get
rtk_acl_igrRuleField_add
rtk_acl_igrRuleEntry_add
rtk_acl_igrRuleEntry_del
rtk_acl_igrRuleEntry_delAll
rtk_acl_igrUnmatchAction_set
rtk_acl_igrUnmatchAction_get
rtk_acl_igrState_set
rtk_acl_igrState_get
rtk_acl_ipRange_set
rtk_acl_ipRange_get
rtk_acl_vidRange_set
rtk_acl_vidRange_get
rtk_acl_portRange_set
rtk_acl_portRange_get
rtk_acl_packetLengthRange_set
rtk_acl_packetLengthRange_get
rtk_acl_igrRuleMode_set
rtk_acl_igrRuleMode_get
rtk_acl_igrPermitState_set
rtk_acl_igrPermitState_get

2.2. rtk_acl_init

`int32 rtk_acl_init(void)`

Initialize ACL module.

Defined in: acl.h

Parameters

`void`

Comments

Must initialize ACL module before calling any ACL APIs. Apollo init acl mode as ACL_IGR_RULE_MODE_0

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

2.3. rtk_acl_template_set

`int32 rtk_acl_template_set(rtk_acl_template_t *aclTemplate)`

Set template of ingress ACL.

Defined in: acl.h

Parameters

`*aclTemplate`

Ingress ACL template

Comments

This function set ACL template.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_INPUT`

Invalid input parameters.

2.4. rtk_acl_template_get

`int32 rtk_acl_template_get(rtk_acl_template_t *aclTemplate)`

Get template of ingress ACL.

Defined in: acl.h

Parameters	<i>*aclTemplate</i>
	Ingress ACL template
Comments	This function get ACL template.
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

2.5. rtk_acl_fieldSelect_set

`int32 rtk_acl_fieldSelect_set(rtk_acl_field_entry_t *pFieldEntry)`

Set user defined field selectors in HSB

Defined in: acl.h

Parameters	<i>*pFieldEntry</i>
	pointer of field selector entry
Comments	System support 16 user defined field selectors. Each selector can be enabled or disable. User can defined retrieving 16-bits in many predefiend standard l2/l3/l4 payload.
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed

2.6. rtk_acl_fieldSelect_get

`int32 rtk_acl_fieldSelect_get(rtk_acl_field_entry_t *pFieldEntry)`

Get user defined field selectors in HSB

Defined in: acl.h

Parameters	<i>*pFieldEntry</i>
	pointer of field selector entry
Comments	None.
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed

2.7. rtk_acl_igrRuleEntry_get

`int32 rtk_acl_igrRuleEntry_get(rt_k_acl_ingress_entry_t *pAclRule)`

Get an ACL entry from ASIC

Defined in: acl.h

Parameters

**pAclRule*

The ACL configuration that this function will add comparison rule

Comments

use this API to get rule entry the field data will return in raw format raw data is return in pAclRule->field.readField

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NULL_POINTER

Pointer pAclRule point to NULL.

RT_ERR_INPUT

Invalid input parameters.

2.8. rtk_acl_igrRuleField_add

`int32 rtk_acl_igrRuleField_add(rt_k_acl_ingress_entry_t *pAclRule,
rt_k_acl_field_t *pAclField)`

Add comparison rule to an ACL configuration

Defined in: acl.h

Parameters

**pAclRule*

The ACL configuration that this function will add comparison rule

**pAclField*

The comparison rule that will be added.

Comments

This function add a comparison rule (*pAclField) to an ACL configuration (*pAclEntry). Pointer pFilter_cfg points to an ACL configuration structure, this structure keeps multiple ACL comparison rules by means of linked list. Pointer pAclField will be added to linked list kepted by structure that pAclEntry points to. caller should not free (*pAclField) before rtk_acl_igrRuleEntry_add is called

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NULL_POINTER

Pointer pFilter_field or pFilter_cfg point to NULL.

RT_ERR_INPUT	Invalid input parameters.
--------------	---------------------------

2.9. rtk_acl_igrRuleEntry_add

int32 rtk_acl_igrRuleEntry_add(rtk_acl_ingress_entry_t *pAclRule)

Add an ACL configuration to ASIC

Defined in: acl.h

Parameters

**pAclRule*
ACL ingress filter rule configuration.

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Pointer pAclrule point to NULL.
RT_ERR_INPUT	Invalid input parameters.
RT_ERR_ENTRY_INDEX	Invalid entryIdx .

2.10. rtk_acl_igrRuleEntry_del

int32 rtk_acl_igrRuleEntry_del(uint32 index)

Delete an ACL configuration from ASIC

Defined in: acl.h

Parameters

index
ACL ingress filter rule configuration.

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_ENTRY_INDEX	Invalid entryIdx .

2.11. rtk_acl_igrRuleEntry_delAll

`int32 rtk_acl_igrRuleEntry_delAll(void)`

Delete all ACL configuration from ASIC

Defined in: acl.h

Parameters *void*

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

2.12. rtk_acl_igrUnmatchAction_set

`int32 rtk_acl_igrUnmatchAction_set(rtk_port_t port,
rtk_filter_unmatch_action_type_t action)`

Apply action to packets when no ACL configuration match

Defined in: acl.h

Parameters *port*

Port id.

action

Action.

Comments This function gets action of packets when no ACL configuration matches.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id.
	RT_ERR_INPUT	Invalid input parameters.

2.13. rtk_acl_igrUnmatchAction_get

```
int32 rtk_acl_igrUnmatchAction_get(rtk_port_t port,
                                    rtk_filter_unmatch_action_type_t *pAction)
```

Get action to packets when no ACL configuration match

Defined in: acl.h

Parameters

port
Port id.

**pAction*
Action.

Comments

This function gets action of packets when no ACL configuration matches.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

2.14. rtk_acl_igrState_set

```
int32 rtk_acl_igrState_set(rtk_port_t port, rtk_enable_t state)
```

Set state of ingress ACL.

Defined in: acl.h

Parameters

port
Port id.

state
Ingress ACL state.

Comments

This function gets action of packets when no ACL configuration matches.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

2.15. rtk_acl_igrState_get

`int32 rtk_acl_igrState_get(rtk_port_t port, rtk_enable_t *pState)`

Get state of ingress ACL.

Defined in: acl.h

Parameters

port
Port id.

**pState*
Ingress ACL state.

Comments

This function gets action of packets when no ACL configuration matches.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

2.16. rtk_acl_ipRange_set

`int32 rtk_acl_ipRange_set(rtk_acl_rangeCheck_ip_t *pRangeEntry)`

Set IP Range check

Defined in: acl.h

Parameters

**pRangeEntry*
IP Range entry

Comments

upper Ip must be larger or equal than lowerIp.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

2.17. rtk_acl_ipRange_get

`int32 rtk_acl_ipRange_get(rtk_acl_rangeCheck_ip_t *pRangeEntry)`

Set IP Range check

Defined in: acl.h

Parameters `*pRangeEntry`
IP Range entry

Comments None.

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_OUT_OF_RANGE</code>	The parameter is out of range

2.18. rtk_acl_vidRange_set

`int32 rtk_acl_vidRange_set(rtk_acl_rangeCheck_vid_t *pRangeEntry)`

Set VID Range check

Defined in: acl.h

Parameters `*pRangeEntry`
VLAN id Range entry

Comments upper Vid must be larger or equal than lowerVid.

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_OUT_OF_RANGE</code>	The parameter is out of range
	<code>RT_ERR_INPUT</code>	Input error

2.19. rtk_acl_vidRange_get

`int32 rtk_acl_vidRange_get(rtk_acl_rangeCheck_vid_t *pRangeEntry)`

Get VID Range check

	Defined in: acl.h
Parameters	* <i>pRangeEntry</i> VLAN id Range entry
Comments	None.
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_OUT_OF_RANGE
	ok failed The parameter is out of range

2.20. rtk_acl_portRange_set

int32 rtk_acl_portRange_set(rtk_acl_rangeCheck_l4Port_t **pRangeEntry*)

Set Port Range check

Defined in: acl.h

	* <i>pRangeEntry</i> L4 Port Range entry
Comments	upper Port must be larger or equal than lowerPort.
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_OUT_OF_RANGE RT_ERR_INPUT
	ok failed The parameter is out of range Input error

2.21. rtk_acl_portRange_get

int32 rtk_acl_portRange_get(rtk_acl_rangeCheck_l4Port_t **pRangeEntry*)

Set Port Range check

Defined in: acl.h

	* <i>pRangeEntry</i> L4 Port Range entry
Comments	None.
Return Codes	RT_ERR_OK
	ok

RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

2.22. rtk_acl_packetLengthRange_set

`int32 rtk_acl_packetLengthRange_set(rtk_acl_rangeCheck_pktLength_t *pRangeEntry)`

Set packet length Range check

Defined in: acl.h

Parameters

`*pRangeEntry`
packet length range entry

Comments

upper length must be larger or equal than lower length.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

2.23. rtk_acl_packetLengthRange_get

`int32 rtk_acl_packetLengthRange_get(rtk_acl_rangeCheck_pktLength_t *pRangeEntry)`

Set packet length Range check

Defined in: acl.h

Parameters

`*pRangeEntry`
packet length range entry

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

2.24. rtk_acl_igrRuleMode_set

`int32 rtk_acl_igrRuleMode_set(rtk_acl_igr_rule_mode_t mode)`

Set ingress ACL rule mode

Defined in: acl.h

Parameters

mode
ingress ACL rule mode

Comments

ACL_IGR_RULE_MODE_0, 64 rules, the size each rule is 16x8 bits
 - ACL_IGR_RULE_MODE_1, 128 rules,
 - the size each rule is 16x4 bits(entry 0~63)
 - the size each rule is 16x3 bits(entry 64~127)

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	Input error

2.25. rtk_acl_igrRuleMode_get

`int32 rtk_acl_igrRuleMode_get(rtk_acl_igr_rule_mode_t *pMode)`

Get ingress ACL rule mode

Defined in: acl.h

Parameters

**pMode*
ingress ACL rule mode

Comments

ACL_IGR_RULE_MODE_0, 64 rules, the size each rule is 16x8 bits
 - ACL_IGR_RULE_MODE_1, 128 rules,
 - the size of each rule is 16x4 bits(entry 0~63)
 - the size of each rule is 16x3 bits(entry 64~127) Mode chaged
 all template/rule will be cleared

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	Input error

2.26. rtk_acl_igrPermitState_set

`int32 rtk_acl_igrPermitState_set(rtk_port_t port, rtk_enable_t state)`

Set permit state of ingress ACL.

Defined in: acl.h

Parameters

port
Port id.

state
Ingress ACL state.

Comments

This function set action of packets when no ACL configuration matches.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

2.27. rtk_acl_igrPermitState_get

`int32 rtk_acl_igrPermitState_get(rtk_port_t port, rtk_enable_t *pState)`

Get state of ingress ACL.

Defined in: acl.h

Parameters

port
Port id.
**pState*
Ingress ACL state.

Comments

This function gets action of packets when no ACL configuration matches.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

3. Module Classification API

Filename: classify.h

Description The file includes the following modules and sub-modules
(1) classification rule add/delete/get

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

3.1. List of Symbols

Here is a list of all functions and variables in this module

classify.h - Definition of Classification API
rtk_classify_init
rtk_classify_cfgEntry_add
rtk_classify_cfgEntry_get
rtk_classify_cfgEntry_del
rtk_classify_field_add
rtk_classify_unmatchAction_set
rtk_classify_unmatchAction_get
rtk_classify_portRange_set
rtk_classify_portRange_get
rtk_classify_ipRange_set
rtk_classify_ipRange_get
rtk_classify_cfSel_set
rtk_classify_cfSel_get
rtk_classify_cfPri2Dscp_set
rtk_classify_cfPri2Dscp_get
rtk_classify_permit_sel_get
rtk_classify_permit_sel_set
rtk_classify_us1pRemarkPrior_set
rtk_classify_us1pRemarkPrior_get

3.2. rtk_classify_init

int32 rtk_classify_init(void)

Initialize classification module.

Defined in: classify.h

Parameters *void*

Comments Must initialize classification module before calling any classification APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

3.3. rtk_classify_cfgEntry_add

`int32 rtk_classify_cfgEntry_add(rtk_classify_cfg_t *pClassifyCfg)`

Add an classification entry to ASIC

Defined in: classify.h

Parameters `*pClassifyCfg`
index of classification entry.

Comments None.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	Pointer pClassifyCfg point to NULL.
	RT_ERR_INPUT	Invalid input parameters.

3.4. rtk_classify_cfgEntry_get

`int32 rtk_classify_cfgEntry_get(rtk_classify_cfg_t *pClassifyCfg)`

Gdd an classification entry from ASIC

Defined in: classify.h

Parameters `*pClassifyCfg`
The classification configuration that this function will add comparison rule

Comments None.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	Pointer pClassifyCfg point to NULL.
	RT_ERR_INPUT	Invalid input parameters.

3.5. rtk_classify_cfgEntry_del

`int32 rtk_classify_cfgEntry_del(uint32 entryIdx)`

Delete an classification configuration from ASIC

Defined in: classify.h

Parameters

entryIdx
index of classification entry.

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_ENTRY_INDEX</code>	Invalid classification index .

3.6. rtk_classify_field_add

`int32 rtk_classify_field_add(rtk_classify_cfg_t *pClassifyEntry,
rtk_classify_field_t *pClassifyField)`

Add comparison field to an classification configuration

Defined in: classify.h

Parameters

**pClassifyEntry*
The classification configuration that this function will add comparison rule
**pClassifyField*
The comparison rule that will be added.

Comments

This function add a comparison rule (**pClassifyField*) to an ACL configuration (**pClassifyEntry*). Pointer *pFilter_cfg* points to an ACL configuration structure, this structure keeps multiple ACL comparison rules by means of linked list. Pointer *pAclField* will be added to linked list kepted by structure that *pAclEntry* points to.
- caller should not free (**pClassifyField*) before `rtk_classify_cfgEntry_add` is called

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	Pointer <i>pFilter_field</i> or <i>pFilter_cfg</i> point to NULL.
<code>RT_ERR_INPUT</code>	Invalid input parameters.

3.7. rtk_classify_unmatchAction_set

`int32 rtk_classify_unmatchAction_set(rtk_classify_unmatch_action_t action)`

Apply action to packets when no classification configuration match

Defined in: classify.h

Parameters

action

unmatch action.

Comments

This function gets action of packets when no classification configuration matches.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

Invalid input parameters.

3.8. rtk_classify_unmatchAction_get

`int32 rtk_classify_unmatchAction_get(rtk_classify_unmatch_action_t *pAction)`

Get action to packets when no classification configuration match

Defined in: classify.h

Parameters

**pAction*

Action.

Comments

This function gets action of packets when no classification configuration matches.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NULL_POINTER

Pointer pAction point to NULL.

RT_ERR_INPUT

Invalid input parameters.

3.9. rtk_classify_portRange_set

```
int32 rtk_classify_portRange_set(rtk_classify_rangeCheck_l4Port_t
*pRangeEntry)
```

Set Port Range check

Defined in: classify.h

Parameters

**pRangeEntry*
L4 Port Range entry

Comments

UpperPort must be larger or equal than lowerPort. This function is not supported in Test chip.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

3.10. rtk_classify_portRange_get

```
int32 rtk_classify_portRange_get(rtk_classify_rangeCheck_l4Port_t
*pRangeEntry)
```

Set Port Range check

Defined in: classify.h

Parameters

**pRangeEntry*
L4 Port Range entry

Comments

This function is not supported in Test chip.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_INPUT	Input error

3.11. rtk_classify_ipRange_set

`int32 rtk_classify_ipRange_set(rtk_classify_rangeCheck_ip_t *pRangeEntry)`

Set IP Range check

Defined in: classify.h

Parameters

`*pRangeEntry`
IP Range entry

Comments

UpperIp must be larger or equal than lowerIp. This function is not supported in Test chip.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_SMI</code>	SMI access error
<code>RT_ERR_OUT_OF_RANGE</code>	The parameter is out of range
<code>RT_ERR_INPUT</code>	Input error

3.12. rtk_classify_ipRange_get

`int32 rtk_classify_ipRange_get(rtk_classify_rangeCheck_ip_t *pRangeEntry)`

Set IP Range check

Defined in: classify.h

Parameters

`*pRangeEntry`
IP Range entry

Comments

This function is not supported in Test chip.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_SMI</code>	SMI access error
<code>RT_ERR_OUT_OF_RANGE</code>	The parameter is out of range

3.13. rtk_classify_cfSel_set

`int32 rtk_classify_cfSel_set(rtk_port_t port, rtk_classify_cf_sel_t cfSel)`

Set CF port selection, only pon port and RGMII port can be set

Defined in: classify.h

Parameters

port

port id, only pon port and RGMII port can be set.

cfSel

CF port selection.

Comments

Only accept pon port and RGMII port. This function is not supported in Test chip.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_SMI

SMI access error

RT_ERR_OUT_OF_RANGE

The parameter is out of range

3.14. rtk_classify_cfSel_get

`int32 rtk_classify_cfSel_get(rtk_port_t port, rtk_classify_cf_sel_t *pCfSel)`

Get CF port selection, only pon port and RGMII port can be get

Defined in: classify.h

Parameters

port

port id, only pon port and RGMII port can be get.

**pCfSel*

pointer of CF port selection.

Comments

Only accept pon port and RGMII port. This function is not supported in Test chip.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_SMI

SMI access error

RT_ERR_OUT_OF_RANGE

The parameter is out of range

RT_ERR_NULL_POINTER

Pointer pClassifyCfg point to NULL.

3.15. rtk_classify_cfPri2Dscp_set

`int32 rtk_classify_cfPri2Dscp_set(rtk_pri_t pri, rtk_dscp_t dscp)`

Set CF priority to DSCP value mapping

Defined in: classify.h

Parameters

pri
priority value

dscp
DSCP value.

Comments

This function is not supported in Test chip.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_OUT_OF_RANGE	The parameter is out of range

3.16. rtk_classify_cfPri2Dscp_get

`int32 rtk_classify_cfPri2Dscp_get(rtk_pri_t pri, rtk_dscp_t *pDscp)`

Get CF priority to DSCP value mapping

Defined in: classify.h

Parameters

pri
priority value
**pDscp*
pointer of DSCP value.

Comments

This function is not supported in Test chip.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_OUT_OF_RANGE	The parameter is out of range
RT_ERR_NULL_POINTER	Pointer pClassifyCfg point to NULL.

3.17. rtk_classify_permit_sel_get

`int32 rtk_classify_permit_sel_get(rtk_classify_permit_sel_t *pPermitSel)`

Set classification permit selection, from 0 to 511 or from 64 to 511

Defined in: classify.h

Parameters

`*pPermitSel`
point of CF permit selection

Comments

Only accept from 0 to 511 or from 64 to 511

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

3.18. rtk_classify_permit_sel_set

`int32 rtk_classify_permit_sel_set(rtk_classify_permit_sel_t permitSel)`

Set classification permit selection, from 0 to 511 or from 64 to 511

Defined in: classify.h

Parameters

`permitSel`
CF permit selection

Comments

Only accept from 0 to 511 or from 64 to 511

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

3.19. rtk_classify_us1pRemarkPrior_set

`int32 rtk_classify_us1pRemarkPrior_set(rtk_classify_us_1premark_prior_t prior)`

Set classification U/S 1p remark is prior than ACL U/S 1p remarking

Defined in: classify.h

Parameters	<i>prior</i>	CF US 1p remarking is prior than ACL or not
Comments		
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

3.20. rtk_classify_us1pRemarkPrior_get

```
int32 rtk_classify_us1pRemarkPrior_get(rtk_classify_us_1premark_prior_t  
*pPrior)
```

Get classification U/S 1p remark is prior than ACL U/S 1p remarking or not

Defined in: classify.h

Parameters	<i>*pPrior</i>	CF US 1p remarking is prior than ACL or not
Comments		
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

4. Module CPU Tag API

Filename: cpu.h

Description The file includes the following modules and sub-modules
(1) CPU tag functions set/get

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

4.1. List of Symbols

Here is a list of all functions and variables in this module

cpu.h - Definition of CPU Tag API
rtk_cpu_init
rtk_cpu_awarePortMask_set
rtk_cpu_awarePortMask_get

rtk_cpu_tagFormat_set
rtk_cpu_tagFormat_get
rtk_cpu_trapInsertTag_set
rtk_cpu_trapInsertTag_get

4.2. rtk_cpu_init

int32 rtk_cpu_init(void)

Initialize cpu tag module.

Defined in: cpu.h

Parameters

void

Comments

Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

4.3. rtk_cpu_awarePortMask_set

int32 rtk_cpu_awarePortMask_set(rt_k_portmask_t *port_mask*)

Set CPU aware port mask.

Defined in: cpu.h

Parameters

port_mask
CPU aware port mask

Comments

Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	

4.4. rtk_cpu_awarePortMask_get

`int32 rtk_cpu_awarePortMask_get(rtk_portmask_t *pPort_mask)`

Get CPU aware port mask.

Defined in: cpu.h

Parameters

`*pPort_mask`
the pointer of CPU aware port mask

Comments

Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	

4.5. rtk_cpu_tagFormat_set

`int32 rtk_cpu_tagFormat_set(rtk_cpu_tag_fmt_t mode)`

Set CPU tag format.

Defined in: cpu.h

Parameters

`mode`
CPU tag format mode

Comments

Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_OUT_OF_RANGE</code>	

4.6. rtk_cpu_tagFormat_get

`int32 rtk_cpu_tagFormat_get(rtk_cpu_tag_fmt_t *pMode)`

Get CPU tag format.

Defined in: cpu.h

Parameters **pMode*
 the pointer of CPU tag format mode

Comments Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	

4.7. rtk_cpu_trapInsertTag_set

`int32 rtk_cpu_trapInsertTag_set(rtk_enable_t state)`

Set trap CPU insert tag state.

Defined in: cpu.h

Parameters *state*
 insert CPU tag state

Comments Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_OUT_OF_RANGE	

4.8. rtk_cpu_trapInsertTag_get

`int32 rtk_cpu_trapInsertTag_get(rtk_enable_t *pState)`

Get trap CPU insert tag state.

Defined in: cpu.h

Parameters **pState*
 the pointer of insert CPU tag state

Comments Must initialize cpu tag module before calling any cpu tag APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

RT_ERR_NULL_POINTER

5. Module 802.1x API

Filename: dot1x.h

Description

The file includes the following modules and sub-modules
(1) Unauth packet handling
(2) 802.1X port-based NAC
(3) 802.1X MAC-based NAC
(4) 802.1X parameter
(5) Parameter for trapped packets

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

5.1. List of Symbols

Here is a list of all functions and variables in this module

dot1x.h - Definition of 802.1x API
rtk_dot1x_init
rtk_dot1x_unauthPacketOper_get
rtk_dot1x_unauthPacketOper_set
rtk_dot1x_portBasedEnable_get
rtk_dot1x_portBasedEnable_set
rtk_dot1x_portBasedAuthStatus_get
rtk_dot1x_portBasedAuthStatus_set
rtk_dot1x_portBasedDirection_get
rtk_dot1x_portBasedDirection_set
rtk_dot1x_macBasedEnable_get
rtk_dot1x_macBasedEnable_set
rtk_dot1x_macBasedDirection_get
rtk_dot1x_macBasedDirection_set
rtk_dot1x_guestVlan_get
rtk_dot1x_guestVlan_set
rtk_dot1x_guestVlanBehavior_get
rtk_dot1x_guestVlanBehavior_set
rtk_dot1x_trapPri_get
rtk_dot1x_trapPri_set

5.2. rtk_dot1x_init

`int32 rtk_dot1x_init(void)`

Initial the dot1x module of the specified device..

Defined in: dot1x.h

Parameters

`void`

Comments

Must initialize dot1x module before calling any dot1x APIs.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

5.3. rtk_dot1x_unauthPacketOper_get

`int32 rtk_dot1x_unauthPacketOper_get(rtk_port_t port, rtk_action_t *pUnauthAction)`

Get the configuration of unauthorized behavior for both 802.1x port and mac based network access control on specified port.

Defined in: dot1x.h

Parameters

`port`

port id

`*pUnauthAction`

The action of how to handle unauthorized packet

Comments

Forwarding action for unauth packet is as following

- DOT1X_ACTION_DROP
- DOT1X_ACTION_TRAP2CPU
- DOT1X_ACTION_TO_GUEST_VLAN

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_NOT_INIT`

The module is not initial

`RT_ERR_NULL_POINTER`

NULL pointer

5.4. rtk_dot1x_unauthPacketOper_set

```
int32 rtk_dot1x_unauthPacketOper_set(rtk_port_t port, rtk_action_t
unauthAction)
```

Set the configuration of unauthorized behavior for both 802.1x port and mac based network access control on specified port.

Defined in: dot1x.h

Parameters

port
port id

unauthAction

The action of how to handle unauthorized packet

Comments

Forwarding action for unauth packet is as following

- DOT1X_ACTION_DROP
- DOT1X_ACTION_TRAP2CPU
- DOT1X_ACTION_TO_GUEST_VLAN

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

5.5. rtk_dot1x_portBasedEnable_get

```
int32 rtk_dot1x_portBasedEnable_get(rtk_port_t port, rtk_enable_t
*pEnable)
```

Get the status of 802.1x port-based network access control on a specific port.

Defined in: dot1x.h

Parameters

port
port id

**pEnable*

The status of 802.1x port

Comments

(1) If a port is 802.1x port based network access control "enabled", it should be authenticated so packets from that port wont be dropped or trapped to CPU.

(2) The status of 802.1x port-based network access control is as following:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_NULL_POINTER	NULL pointer

5.6. rtk_dot1x_portBasedEnable_set

`int32 rtk_dot1x_portBasedEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set the status of 802.1x port-based network access control on a specific port

Defined in: dot1x.h

Parameters	<i>port</i>
	port id
	<i>enable</i>

The status of 802.1x port

Comments
(1) If a port is 802.1x port based network access control "enabled", it should be authenticated so packets from that port wont be dropped or trapped to CPU.

(2) The status of 802.1x port-based network access control is as following:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	Invalid port id

5.7. rtk_dot1x_portBasedAuthStatus_get

`int32 rtk_dot1x_portBasedAuthStatus_get(rtk_port_t port,
rtk_dot1x_auth_status_t *pPort_auth)`

Get the authenticated status of 802.1x port-based network access control on a specific port.

Defined in: dot1x.h

Parameters

port
port id
**pPort_auth*
The status of 802.1x port

Comments

The authenticated status of 802.1x port-based network access control is as following:
- UNAUTH
- AUTH

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	Invalid port id
RT_ERR_NULL_POINTER	NUL pointer

5.8. rtk_dot1x_portBasedAuthStatus_set

```
int32 rtk_dot1x_portBasedAuthStatus_set(rtk_port_t port,  
rtk_dot1x_auth_status_t port_auth)
```

port.

Defined in: dot1x.h

Parameters

port
port id
port_auth
The status of 802.1x port

Comments

The authenticated status of 802.1x port-based network access control is as following:
- UNAUTH
- AUTH

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	Invalid port id

5.9. rtk_dot1x_portBasedDirection_get

```
int32 rtk_dot1x_portBasedDirection_get(rtk_port_t port,
                                         rtk_dot1x_direction_t *pPort_direction)
```

port.

Defined in: dot1x.h

Parameters

port

port id

**pPort_direction*

The status of 802.1x port

Comments

The operate controlled direction of 802.1x port-based network access control is as following:

- BOTH
- IN

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_PORT_ID

Invalid port id

5.10. rtk_dot1x_portBasedDirection_set

```
int32 rtk_dot1x_portBasedDirection_set(rtk_port_t port,
                                         rtk_dot1x_direction_t port_direction)
```

port.

Defined in: dot1x.h

Parameters

port

port id

port_direction

The controlled direction of 802.1x port

Comments

The operate controlled direction of 802.1x port-based network access control is as following:

- BOTH
- IN

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	Invalid port id

5.11. rtk_dot1x_macBasedEnable_get

```
int32 rtk_dot1x_macBasedEnable_get(rtk_port_t port, rtk_enable_t
*pEnable)
```

Get the status of 802.1x MAC-based network access control on a specific port.

Defined in: dot1x.h

Parameters	<i>port</i> port id <i>*pEnable</i> The status of 802.1x MAC
-------------------	---

Comments
(1) If a port is 802.1x MAC based network access control "enabled", the incoming packets should be authenticated so packets from that port wont be dropped or trapped to CPU.

(2) The status of 802.1x MAC-based network access control is as following:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_NULL_POINTER	NULL pointer

5.12. rtk_dot1x_macBasedEnable_set

```
int32 rtk_dot1x_macBasedEnable_set(rtk_port_t port, rtk_enable_t enable)
```

		Set the status of 802.1x MAC-based network access control on a specific port.
		Defined in: dot1x.h
Parameters	<i>port</i>	port id
	<i>enable</i>	The status of 802.1x MAC
Comments	(1) If a port is 802.1x MAC based network access control "enabled", the incoming packets should be authenticated so packets from that port wont be dropped or trapped to CPU.	
	(2) The status of 802.1x MAC-based network access control is as following: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	Invalid port id

5.13. rtk_dot1x_macBasedDirection_get

```
int32 rtk_dot1x_macBasedDirection_get(rtk_dot1x_direction_t
*pMac_direction)
```

Get the operate controlled direction 802.1x mac-based network access control on system.

Defined in: dot1x.h

Parameters	<i>*pMac_direction</i>	port id
Comments	The operate controlled direction of 802.1x mac-based network access control is as following: - BOTH - IN	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial

5.14. rtk_dot1x_macBasedDirection_set

```
int32 rtk_dot1x_macBasedDirection_set(rtk_dot1x_direction_t  
mac_direction)
```

Set the operate controlled direction 802.1x mac-based network access control on system.

Defined in: dot1x.h

Parameters

mac_direction

The controlled direction of 802.1x mac

Comments

The operate controlled direction of 802.1x mac-based network access control is as following:

- BOTH
- IN

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

5.15. rtk_dot1x_guestVlan_get

```
int32 rtk_dot1x_guestVlan_get(rtk_vlan_t *pGuest_vlan)
```

Get guest vlan on specified port.

Defined in: dot1x.h

Parameters

**pGuest_vlan*

The controlled direction of 802.1x mac

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

5.16. rtk_dot1x_guestVlan_set

`int32 rtk_dot1x_guestVlan_set(rtk_vlan_t guest_vlan)`

Set guest vlan

Defined in: dot1x.h

Parameters

guest_vlan
guest vlan id

Comments

(1) Configure the port's forwarding guest vlan setting.
 (2) When the packet is unauthenticated and action is DOT1X_ACTION_TO_GUEST_VLAN, the packet will based on the guest vlan to forward.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_VLAN_VID</code>	invalid vlan id

5.17. rtk_dot1x_guestVlanBehavior_get

`int32 rtk_dot1x_guestVlanBehavior_get(rtk_dot1x_guestVlanBehavior_t *pBehavior)`

Get forwarding behavior for host in guest vlan.

Defined in: dot1x.h

Parameters

**pBehavior*
guest vlan id

Comments

Forwarding behavior is as following
 - DISALLOW_TO_AUTH_DA
 - ALLOW_TO_AUTH_DA

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

5.18. rtk_dot1x_guestVlanBehavior_set

`int32 rtk_dot1x_guestVlanBehavior_set(rtk_dot1x_guestVlanBehavior_t behavior)`

Set forwarding behavior for host in guest vlan.

Defined in: dot1x.h

Parameters

behavior

Forwarding behavior

Comments

Forwarding behavior is as following

- DISALLOW_TO_AUTH_DA
- ALLOW_TO_AUTH_DA

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_INPUT

invalid input parameter

5.19. rtk_dot1x_trapPri_get

`int32 rtk_dot1x_trapPri_get(rtk_pri_t *pPriority)`

Get priority of trapped dot1x packets.

Defined in: dot1x.h

Parameters

**pPriority*

Forwarding behavior

Comments

(1) Get the dot1x trap to cpu priority and valid range is 0-7.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

5.20. rtk_dot1x_trapPri_set

`int32 rtk_dot1x_trapPri_set(rtk_pri_t priority)`

Set priority of trapped dot1x packet.

Defined in: dot1x.h

Parameters

priority
priority

Comments

(1) Config the dot1x trap to cpu priority and valid range is 0-7.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_PRIORITY</code>	invalid priority value

6. Module GPIO API

Filename: gpio.h

Description

Provide the APIs to enable and configure GPIO

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

6.1. List of Symbols

Here is a list of all functions and variables in this module

- gpio.h - Definition of GPIO API
- rtk_gpio_init
- rtk_gpio_state_set
- rtk_gpio_state_get
- rtk_gpio_mode_set
- rtk_gpio_mode_get
- rtk_gpio_databit_get
- rtk_gpio_databit_set
- rtk_gpio_intr_set
- rtk_gpio_intr_get

6.2. rtk_gpio_init

`int32 rtk_gpio_init(void)`

gpio init function

Defined in: gpio.h

Parameters

`void`

Comments

6.3. rtk_gpio_state_set

`int32 rtk_gpio_state_set(uint32 gpioId, rtk_enable_t enable)`

enable or disable gpio function

Defined in: gpio.h

Parameters

`gpioId`

gpio id from 0~71

`enable`

enable or disable

Comments

None.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

6.4. rtk_gpio_state_get

`int32 rtk_gpio_state_get(uint32 gpioId, rtk_enable_t *enable)`

enable or disable gpio function

Defined in: gpio.h

Parameters

`gpioId`

gpio id from 0~71

	<i>*enable</i>	point for get enable or disable
Comments	None.	
Return Codes	RT_ERR_OK RT_ERR_FAILED	ok failed

6.5. rtk_gpio_mode_set

	int32 rtk_gpio_mode_set(uint32 <i>gpioId</i> , rtk_gpio_mode_t <i>mode</i>)	
	set gpio to input or output mode	
	Defined in: gpio.h	
Parameters	<i>gpioId</i>	gpio id from 0 to 71
	<i>mode</i>	gpio mode, input or output mode
Comments	None.	
Return Codes	RT_ERR_OK RT_ERR_FAILED	ok failed

6.6. rtk_gpio_mode_get

	int32 rtk_gpio_mode_get(uint32 <i>gpioId</i> , rtk_gpio_mode_t * <i>mode</i>)	
	set gpio to input or output mode	
	Defined in: gpio.h	
Parameters	<i>gpioId</i>	gpio id from 0 to 71
	<i>*mode</i>	point for get gpio mode
Comments	None.	
Return Codes	RT_ERR_OK	ok

RT_ERR_FAILED	failed
---------------	--------

6.7. rtk_gpio_databit_get

`int32 rtk_gpio_databit_get(uint32 gpioId, uint32 *data)`

read gpio data

Defined in: gpio.h

Parameters

gpioId
 gpio id from 0 to 71

**data*
 point for read data from gpio

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

6.8. rtk_gpio_databit_set

`int32 rtk_gpio_databit_set(uint32 gpioId, uint32 data)`

write data to gpio

Defined in: gpio.h

Parameters

gpioId
 gpio id from 0 to 71

data
 point for write data to gpio

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

6.9. rtk_gpio_intr_set

`int32 rtk_gpio_intr_set(uint32 intrId, rtk_enable_t state)`

write data to gpio

Defined in: gpio.h

Parameters

intrId
gpio id from 0 to 71

state
point for write data to gpio

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

6.10. rtk_gpio_intr_get

`int32 rtk_gpio_intr_get(uint32 intrId, rtk_enable_t *state)`

write data to gpio

Defined in: gpio.h

Parameters

intrId
gpio id from 0 to 71

**state*
point for write data to gpio

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

7. Module GPON Mac API

Filename: gpon.h

Description Provide the APIs to access GPON MAC

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

7.1. List of Symbols

Here is a list of all functions and variables in this module

gpon.h - Definition of GPON Mac API
rtk_gpon_driver_initialize
rtk_gpon_driver_deInitialize
rtk_gpon_device_initialize
rtk_gpon_device_deInitialize
rtk_gpon_evtHdlStateChange_reg
rtk_gpon_evtHdlDsFecChange_reg
rtk_gpon_evtHdlUsFecChange_reg
rtk_gpon_evtHdlUsPloamUrgEmpty_reg
rtk_gpon_evtHdlUsPloamNrmEmpty_reg
rtk_gpon_evtHdlPloam_reg
rtk_gpon_evtHdlOmci_reg
rtk_gpon_callbackQueryAesKey_reg
rtk_gpon_evtHdlAlarm_reg
rtk_gpon_serialNumber_set
rtk_gpon_serialNumber_get
rtk_gpon_password_set
rtk_gpon_password_get
rtk_gpon_parameter_set
rtk_gpon_parameter_get
rtk_gpon_activate
rtk_gpon_deActivate
rtk_gpon_ponStatus_get
rtk_gpon_isr_entry
rtk_gpon_tcont_create
rtk_gpon_tcont_destroy
rtk_gpon_tcont_get
rtk_gpon_dsFlow_set
rtk_gpon_dsFlow_get
rtk_gpon_usFlow_set
rtk_gpon_usFlow_get
rtk_gpon_ploam_send
rtk_gpon_broadcastPass_set
rtk_gpon_broadcastPass_get
rtk_gpon_nonMcastPass_set
rtk_gpon_nonMcastPass_get
rtk_gpon_multicastAddrCheck_set
rtk_gpon_multicastAddrCheck_get

```
rtk_gpon_macFilterMode_set  
rtk_gpon_macFilterMode_get  
rtk_gpon_mcForceMode_set  
rtk_gpon_mcForceMode_get  
rtk_gpon_macEntry_add  
rtk_gpon_macEntry_del  
rtk_gpon_macEntry_get  
rtk_gpon_rdi_set  
rtk_gpon_rdi_get  
rtk_gpon_powerLevel_set  
rtk_gpon_powerLevel_get  
rtk_gpon_alarmStatus_get  
rtk_gpon_globalCounter_get  
rtk_gpon_tcontCounter_get  
rtk_gpon_flowCounter_get  
rtk_gpon_version_get  
rtk_gpon_txForceLaser_set  
rtk_gpon_txForceLaser_get  
rtk_gpon_txForceIdle_set  
rtk_gpon_txForceIdle_get  
rtk_gpon_dsFecSts_get  
rtk_gpon_version_show  
rtk_gpon_devInfo_show  
rtk_gpon_gtc_show  
rtk_gpon_tcont_show  
rtk_gpon_dsFlow_show  
rtk_gpon_usFlow_show  
rtk_gpon_macTable_show  
rtk_gpon_globalCounter_show  
rtk_gpon_tcontCounter_show  
rtk_gpon_flowCounter_show  
rtk_gpon_omci_tx  
rtk_gpon_omci_rx  
rtk_gpon_test_set  
rtk_gpon_test_get  
rtk_gpon_unit_test  
rtk_gpon_initial  
rtk_gpon_deinitial  
rtk_gpon_debug_set  
rtk_gpon_autoTcont_set  
rtk_gpon_autoTcont_get  
rtk_gpon_autoBoh_set  
rtk_gpon_autoBoh_get  
rtk_gpon_eqdOffset_set  
rtk_gpon_eqdOffset_get
```

rtk_gpon_pktGen_cfg_set
rtk_gpon_pktGen_buf_set

7.2. rtk_gpon_driver_initialize

int32 rtk_gpon_driver_initialize(void)

GPON Mac Drv Initialization. To start the GPON Mac Drv.

Defined in: gpon.h

Parameters

void

Comments

Must initialize classification module before calling any GPON APIs.

Return Codes

RT_ERR_OK	ok
others	fail

7.3. rtk_gpon_driver_deInitialize

int32 rtk_gpon_driver_deInitialize(void)

GPON Mac Drv De-Initialization. To Stop the GPON Mac Drv. The last function call for GPON Mac Drv.

Defined in: gpon.h

Parameters

void

Comments

The Device should be de-initialized before this function call.

Return Codes

RT_ERR_OK	ok
others	fail

7.4. rtk_gpon_device_initialize

int32 rtk_gpon_device_initialize(void)

GPON Mac Device Initialization. To start the device of the GPON Mac.

Defined in: gpon.h

Parameters *void*

Comments This function should be called after the Drv Initialization and before any other operation. It should be called after Device is de-initialized but the Drv is not be de-initialized.

Return Codes	RT_ERR_OK	ok
	others	fail

7.5. rtk_gpon_device_deInitialize

int32 rtk_gpon_device_deInitialize(void)

GPON Mac Device De-Initialization. To stop the device of the GPON Mac.

Defined in: gpon.h

Parameters *void*

Comments It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.6. rtk_gpon_evtHdlStateChange_reg

**int32
rtk_gpon_evtHdlStateChange_reg(rtk_gpon_eventHandleFunc_stateChange
_tfunc)**

This function is called to register the callback function of the State Change.

Defined in: gpon.h

Parameters *func*
the callback function to be registered

Comments It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.7. rtk_gpon_evtHdlDsFecChange_reg

int32
rtk_gpon_evtHdlDsFecChange_reg(rtк_gpon_eventHandleFunc_fecChange_t func)

This function is called to register the callback function of the D/S FEC Change.

Defined in: gpon.h

Parameters *func*
 the callback function to be registered

Comments It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.8. rtk_gpon_evtHdlUsFecChange_reg

int32
rtk_gpon_evtHdlUsFecChange_reg(rtк_gpon_eventHandleFunc_fecChange_t func)

This function is called to register the callback function of the U/S FEC Change.

Defined in: gpon.h

Parameters *func*
 the callback function to be registered

Comments It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes	RT_ERR_OK	ok
---------------------	-----------	----

others	fail
--------	------

7.9. rtk_gpon_evtHdlUsPloamUrgEmpty_reg

int32
**rtk_gpon_evtHdlUsPloamUrgEmpty_reg(rtк_gpon_eventHandleFunc_usPlo
amEmpty_t *func*)**

This function is called to register the callback function of the U/S PLOAM urgent queue is empty.

Defined in: gpon.h

Parameters

func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.10. rtk_gpon_evtHdlUsPloamNrmEmpty_reg

int32
**rtk_gpon_evtHdlUsPloamNrmEmpty_reg(rtк_gpon_eventHandleFunc_usPlo
amEmpty_t *func*)**

This function is called to register the callback function of the U/S PLOAM normal queue is empty.

Defined in: gpon.h

Parameters

func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.11. rtk_gpon_evtHdlPloam_reg

`int32 rtk_gpon_evtHdlPloam_reg(rtk_gpon_eventHandleFunc_ploam_t func)`

This function is called to register the event handler of the Rx PLOAM.

Defined in: gpon.h

Parameters

func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.12. rtk_gpon_evtHdlOmci_reg

`int32 rtk_gpon_evtHdlOmci_reg(rtk_gpon_eventHandleFunc_omci_t func)`

This function is called to register the event handler of the Rx OMCI.

Defined in: gpon.h

Parameters

func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.13. rtk_gpon_callbackQueryAesKey_reg

`int32
rtk_gpon_callbackQueryAesKey_reg(rtk_gpon_callbackFunc_queryAesKey_t func)`

This function is called to register the callback function of the AES Key Query.

Defined in: gpon.h

Parameters

func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.14. rtk_gpon_evtHdlAlarm_reg

```
int32 rtk_gpon_evtHdlAlarm_reg(rtk_gpon_alarm_type_t alarmType,
                               rtk_gpon_eventHandleFunc_fault_t func)
```

This function is called to register the alarm event handler of the alarm.

Defined in: gpon.h

Parameters

alarmType
the alarm type
func
the callback function to be registered

Comments

It should be called before the Drv is de-initialized and the GPON Mac is not activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.15. rtk_gpon_serialNumber_set

```
int32 rtk_gpon_serialNumber_set(rtk_gpon_serialNumber_t *pSN)
```

GPON MAC Set Serial Number.

Defined in: gpon.h

Parameters

**pSN*
the pointer of Serial Number

Comments It should be called before the GPON MAC is activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.16. rtk_gpon_serialNumber_get

`int32 rtk_gpon_serialNumber_get(rtk_gpon_serialNumber_t *pSN)`

GPON MAC get Serial Number.

Defined in: gpon.h

Parameters `*pSN`
the pointer of Serial Number

Comments It should be called before the GPON MAC is activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.17. rtk_gpon_password_set

`int32 rtk_gpon_password_set(rtk_gpon_password_t *pPwd)`

GPON MAC set Password.

Defined in: gpon.h

Parameters `*pPwd`
the pointer of Password

Comments It should be called before the GPON MAC is activated.

Return Codes	RT_ERR_OK	ok
	others	fail

7.18. rtk_gpon_password_get

`int32 rtk_gpon_password_get(rtk_gpon_password_t *pPwd)`

GPON MAC get Password.

Defined in: gpon.h

Parameters

`*pPwd`
the pointer of Password

Comments

It should be called before the GPON MAC is activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.19. rtk_gpon_parameter_set

`int32 rtk_gpon_parameter_set(rtk_gpon_parameter_type_t type, void *pPara)`

GPON MAC set parameters.

Defined in: gpon.h

Parameters

`type`
the parameter type
`*pPara`
the pointer of Parameter

Comments

It should be called before the GPON MAC is activated.

Return Codes

RT_ERR_OK	ok
others	fail

7.20. rtk_gpon_parameter_get

`int32 rtk_gpon_parameter_get(rtk_gpon_parameter_type_t type, void *pPara)`

GPON MAC get parameters, which is set by rtk_gpon_parameter_set.

Defined in: gpon.h

Parameters	<i>type</i> the parameter type <i>*pPara</i> the pointer of Parameter	
Comments		
Return Codes	RT_ERR_OK others	ok fail

7.21. rtk_gpon_activate

`int32 rtk_gpon_activate(rtk_gpon_initialState_t initState)`

GPON MAC Activating.

Defined in: gpon.h

Parameters	<i>initState</i> the initial state when ONU active	
Comments		
Return Codes	RT_ERR_OK others	ok fail

7.22. rtk_gpon_deActivate

`int32 rtk_gpon_deActivate(void)`

GPON MAC de-Activate.

Defined in: gpon.h

Parameters	<i>void</i>
Comments	The GPON MAC is out of work now.
Return Codes	

RT_ERR_OK	ok
others	fail

7.23. rtk_gpon_ponStatus_get

`int32 rtk_gpon_ponStatus_get(rt_k_gpon_fsm_status_t *pStatus)`

GPON MAC Get PON Status.

Defined in: gpon.h

Parameters

`*pStatus`
pointer of status

Comments

If the device is not activated, an unknown status is returned.

Return Codes

RT_ERR_OK	ok
others	fail

7.24. rtk_gpon_isr_entry

`void rtk_gpon_isr_entry(void)`

GPON MAC ISR entry

Defined in: gpon.h

Parameters

`void`

Comments

It should be called in interrupt process or a polling thread

7.25. rtk_gpon_tcont_create

`int32 rtk_gpon_tcont_create(rt_k_gpon_tcont_ind_t *pInd,
rt_k_gpon_tcont_attr_t *pAttr)`

GPON MAC Create a TCont by assigning an alloc id.

Defined in: gpon.h

Parameters	* <i>pInd</i> the pointer of ALLOC_id
	* <i>pAttr</i> the pointer of tcont attribute(TCont id)
Comments	A TCont ID is returned in pAttr.
Return Codes	RT_ERR_OK others
	ok fail

7.26. rtk_gpon_tcont_destroy

`int32 rtk_gpon_tcont_destroy(rtk_gpon_tcont_ind_t *pInd)`

GPON MAC Remove a TCont.

Defined in: gpon.h

Parameters	* <i>pInd</i> the pointer of ALLOC_id
Comments	
Return Codes	RT_ERR_OK others
	ok fail

7.27. rtk_gpon_tcont_get

`int32 rtk_gpon_tcont_get(rtk_gpon_tcont_ind_t *pInd, rtk_gpon_tcont_attr_t *pAttr)`

GPON MAC Get a TCont with an alloc id.

Defined in: gpon.h

Parameters	* <i>pInd</i> the pointer of ALLOC_id
	* <i>pAttr</i> the pointer of tcont attribute(TCont id)
Comments	The TCont ID is returned in pAttr.
Return Codes	

RT_ERR_OK	ok
others	fail

7.28. rtk_gpon_dsFlow_set

`int32 rtk_gpon_dsFlow_set(uint32 flowId, rtk_gpon_dsFlow_attr_t *pAttr)`

GPON MAC set a D/S flow.

Defined in: gpon.h

Parameters

flowId
the flow id
**pAttr*
the pointer of flow attribute(Gem port id,...)

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.29. rtk_gpon_dsFlow_get

`int32 rtk_gpon_dsFlow_get(uint32 flowId, rtk_gpon_dsFlow_attr_t *pAttr)`

GPON MAC get a D/S flow.

Defined in: gpon.h

Parameters

flowId
the flow id
**pAttr*
the pointer of flow attribute(Gem port id,...)

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.30. rtk_gpon_usFlow_set

`int32 rtk_gpon_usFlow_set(uint32 flowId, rtk_gpon_usFlow_attr_t *pAttr)`

GPON MAC set a U/S flow.

Defined in: gpon.h

Parameters

flowId
the flow id
**pAttr*
the pointer of flow attribute(Gem port id,...)

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.31. rtk_gpon_usFlow_get

`int32 rtk_gpon_usFlow_get(uint32 flowId, rtk_gpon_usFlow_attr_t *pAttr)`

GPON MAC get a U/S flow.

Defined in: gpon.h

Parameters

flowId
the flow id
**pAttr*
the pointer of flow attribute(Gem port id,...)

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.32. rtk_gpon_ploam_send

`int32 rtk_gpon_ploam_send(int32 urgent, rtk_gpon_ploam_t *pPloam)`

GPON MAC Send a PLOAM in upstream.

Defined in: gpon.h

Parameters

urgent
specify it is a urgent(1) or normal(0) PLOAM message

**pPloam*
the pointer of PLOAM message

Comments

A error is returned if the PLOAM is not sent.

Return Codes

RT_ERR_OK	ok
others	fail

7.33. rtk_gpon_broadcastPass_set

`int32 rtk_gpon_broadcastPass_set(int32 mode)`

GPON MAC set the broadcast pass mode.

Defined in: gpon.h

Parameters

mode
turn on(1) or off(0) the broadcast pass mode.

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.34. rtk_gpon_broadcastPass_get

`int32 rtk_gpon_broadcastPass_get(int32 *pMode)`

GPON MAC get the broadcast pass mode.

Defined in: gpon.h

Parameters

**pMode*
the pointer of broadcast pass mode: turn on(1) or off(0).

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.35. rtk_gpon_nonMcastPass_set

`int32 rtk_gpon_nonMcastPass_set(int32 mode)`

GPON MAC set the non-multicast pass mode.

Defined in: gpon.h

Parameters

mode
turn on(1) or off(0) the non

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.36. rtk_gpon_nonMcastPass_get

`int32 rtk_gpon_nonMcastPass_get(int32 *pMode)`

GPON MAC get the non-multicast pass mode.

Defined in: gpon.h

Parameters

**pMode*
the pointer of non

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.37. rtk_gpon_multicastAddrCheck_set

`int32 rtk_gpon_multicastAddrCheck_set(uint32 ipv4_pattern, uint32
ipv6_pattern)`

GPON MAC set the address pattern.

Defined in: gpon.h

Parameters

ipv4_pattern

Address pattern of DA[47:24] for IPv4 packets.

ipv6_pattern

Address pattern of DA[47:32] for IPv6 packets.

Comments

Return Codes

RT ERR OK

ok

fail

7.38. rtk_gpon_multicastAddrCheck_get

int32 rtk_gpon_multicastAddrCheck_get(uint32 *pIpv4_Pattern, uint32 *pIpv6_Pattern)

GPON MAC get the address pattern.

Defined in: gpon.h

**pIpv4_Pattern*

Address patterns

Ipv6_Pattern

Address pattern of DA[47:24] for IPv6 packets

Comments

Return Codes

RT ERR OK

ok

fail

7.39. rtk_gpon_macFilterMode_set

```
int32 rtk_gpon_macFilterMode_set(rt_k_gpon_macTable_exclude_mode_t  
mode)
```

GPON MAC set the mac filter mode.

Defined in: gpon.h

Parameters *mode*
MAC table filter mode.

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.40. rtk_gpon_macFilterMode_get

```
int32 rtk_gpon_macFilterMode_get(rt_k_gpon_macTable_exclude_mode_t  
*pMode)
```

GPON MAC get the mac filter mode.

Defined in: gpon.h

Parameters **pMode*
pointer of MAC filter table filter mode.

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.41. rtk_gpon_mcForceMode_set

```
int32 rtk_gpon_mcForceMode_set(rt_k_gpon_mc_force_mode_t ipv4,  
rt_k_gpon_mc_force_mode_t ipv6)
```

GPON MAC set the multicast force mode.

Defined in: gpon.h

Parameters *ipv4*
IPv4 multicast force mode.
ipv6
IPv6 multicast force mode.

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.42. rtk_gpon_mcForceMode_get

```
int32 rtk_gpon_mcForceMode_get(rtk_gpon_mc_force_mode_t *pIpv4,
                                rtk_gpon_mc_force_mode_t *pIpv6)
```

GPON MAC get the multicast force mode.

Defined in: gpon.h

Parameters

*pIpv4	The pointer of IPv4 multicast force mode.
*pIpv6	The pointer of IPv6 multicast force mode.

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.43. rtk_gpon_macEntry_add

```
int32 rtk_gpon_macEntry_add(rtk_gpon_macTable_entry_t *pEntry)
```

GPON MAC Add a MAC entry by the MAC Address.

Defined in: gpon.h

Parameters

*pEntry	pointer of MAC filter table entry.
---------	------------------------------------

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.44. rtk_gpon_macEntry_del

```
int32 rtk_gpon_macEntry_del(rtk_gpon_macTable_entry_t *pEntry)
```

GPON MAC Remove a MAC entry by the MAC Address.

Defined in: gpon.h

Parameters **pEntry*
 pointer of MAC filter table entry.

Comments

Return Codes RT_ERR_OK ok
 others fail

7.45. rtk_gpon_macEntry_get

```
int32 rtk_gpon_macEntry_get(uint32 index, rtk_gpon_macTable_entry_t
    *pEntry)
```

GPON MAC Get a MAC entry by the table index.

Defined in: gpon.h

Parameters *index*
 index of MAC filter table entry.
 **pEntry*
 pointer of MAC filter table entry.

Comments

Return Codes RT_ERR_OK ok
 others fail

7.46. rtk_gpon_rdi_set

```
int32 rtk_gpon_rdi_set(int32 enable)
```

GPON MAC set the RDI indicator in upstream.

Defined in: gpon.h

Parameters *enable*
 specify to turn on/off RDI.

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.47. rtk_gpon_rdi_get

`int32 rtk_gpon_rdi_get(int32 *pEnable)`

GPON MAC get the RDI indicator in upstream.

Defined in: gpon.h

Parameters	<i>*pEnable</i> the pointer of RDI indicator.
-------------------	--

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.48. rtk_gpon_powerLevel_set

`int32 rtk_gpon_powerLevel_set(uint8 level)`

Serial_Number_ONU PLOAMu message.

Defined in: gpon.h

Parameters	<i>level</i> the power lever.
-------------------	----------------------------------

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.49. rtk_gpon_powerLevel_get

`int32 rtk_gpon_powerLevel_get(uint8 *pLevel)`

GPON MAC get ONU power level.

Defined in: gpon.h

Parameters

**pLevel*
the pointer of power lever.

Comments

Return Codes

RT_ERR_OK
others

ok
fail

7.50. rtk_gpon_alarmStatus_get

```
int32 rtk_gpon_alarmStatus_get(rtk_gpon_alarm_type_t alarm, int32
*pStatus)
```

GPON MAC get the alarm status.

Defined in: gpon.h

Parameters

alarm
the alarm type.
**pStatus*
the pointer of alarm status

Comments

Return Codes

RT_ERR_OK
others

ok
fail

7.51. rtk_gpon_globalCounter_get

```
int32 rtk_gpon_globalCounter_get(rtk_gpon_global_performance_type_t
type, rtk_gpon_global_counter_t *pPara)
```

GPON MAC get global performance counter.

Defined in: gpon.h

Parameters

type
the PM type.

**pPara*
the pointer of counter data

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.52. rtk_gpon_tcontCounter_get

```
int32 rtk_gpon_tcontCounter_get(uint32 tcontId,
                                rtk_gpon_tcont_performance_type_t type, rtk_gpon_tcont_counter_t *pPara)
```

GPON MAC get Tcont performance counter.

Defined in: gpon.h

Parameters

tcontId
the TCont id

type
the PM type.

**pPara*
the pointer of counter data

Comments

Return Codes	RT_ERR_OK	ok
	others	fail

7.53. rtk_gpon_flowCounter_get

```
int32 rtk_gpon_flowCounter_get(uint32 flowId,
                                rtk_gpon_flow_performance_type_t type, rtk_gpon_flow_counter_t *pPara)
```

GPON MAC get Flow performance counter.

Defined in: gpon.h

Parameters

flowId
the flow id

type
the PM type.
**pPara*
the pointer of counter data

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.54. rtk_gpon_version_get

```
int32 rtk_gpon_version_get(rtk_gpon_device_ver_t *pHver,  
                           rtk_gpon_driver_ver_t *pSver)
```

GPON MAC get the version infomation for debug.

Defined in: gpon.h

Parameters

**pHver*
the pointer of Hardware versiotsn
**pSver*
the pointer of Software versiotsn

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.55. rtk_gpon_txForceLaser_set

```
int32 rtk_gpon_txForceLaser_set(rtk_gpon_laser_status_t status)
```

GPON MAC set the Laser status.

Defined in: gpon.h

Parameters

status
specify to force turn on/off laser

Comments**Return Codes**

RT_ERR_OK	ok
others	fail

7.56. rtk_gpon_txForceLaser_get

`int32 rtk_gpon_txForceLaser_get(rtk_gpon_laser_status_t *pStatus)`

GPON MAC get the Laser status.

Defined in: gpon.h

Parameters

`*pStatus`
pointer of force laser status

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.57. rtk_gpon_txForceIdle_set

`int32 rtk_gpon_txForceIdle_set(int32 on)`

GPON MAC set to force insert the idle in upstream.

Defined in: gpon.h

Parameters

`on`
specify to force send Idle

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.58. rtk_gpon_txForceIdle_get

`int32 rtk_gpon_txForceIdle_get(int32 *pOn)`

GPON MAC get the status to force insert the idle in upstream.

Defined in: gpon.h

Parameters **pOn*
pointer of force Idle

Comments

Return Codes RT_ERR_OK
others ok fail

7.59. rtk_gpon_dsFecSts_get

`int32 rtk_gpon_dsFecSts_get(int32* pEn)`

GPON MAC get the status to FEC in downstream from Ident field.

Defined in: gpon.h

Parameters *pEn*
pointer of D/S FEC status

Comments

Return Codes RT_ERR_OK
others ok fail

7.60. rtk_gpon_version_show

`int32 rtk_gpon_version_show(void)`

GPON MAC show version infomation in COM port.

Defined in: gpon.h

Parameters *void*

Comments

Return Codes RT_ERR_OK
others ok fail

7.61. rtk_gpon_devInfo_show

`int32 rtk_gpon_devInfo_show(void)`

GPON MAC show the whole driver infomation in COM port.

Defined in: gpon.h

Parameters

`void`

Comments

Return Codes

`RT_ERR_OK`

ok

others

fail

7.62. rtk_gpon_gtc_show

`int32 rtk_gpon_gtc_show(void)`

GPON MAC show the whole GTC infomation in COM port.

Defined in: gpon.h

Parameters

`void`

Comments

Return Codes

`RT_ERR_OK`

ok

others

fail

7.63. rtk_gpon_tcont_show

`int32 rtk_gpon_tcont_show(uint32 tcont)`

GPON MAC show the TCont infomation in COM port.

Defined in: gpon.h

Parameters

tcont
pointer of D/S FEC status

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.64. rtk_gpon_dsFlow_show

`int32 rtk_gpon_dsFlow_show(uint32 flow)`

GPON MAC show the D/S flow infomation in COM port.

Defined in: gpon.h

Parameters	<i>flow</i> pointer of D/S FEC status
-------------------	--

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.65. rtk_gpon_usFlow_show

`int32 rtk_gpon_usFlow_show(uint32 flow)`

GPON MAC show the U/S flow infomation in COM port.

Defined in: gpon.h

Parameters	<i>flow</i> pointer of D/S FEC status
-------------------	--

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.66. rtk_gpon_macTable_show

`int32 rtk_gpon_macTable_show(void)`

GPON MAC show Ethernet Mac Table in COM port.

Defined in: gpon.h

Parameters

`void`

Comments

Return Codes

`RT_ERR_OK`

ok

others

fail

7.67. rtk_gpon_globalCounter_show

`int32 rtk_gpon_globalCounter_show(rtk_gpon_global_performance_type_t type)`

GPON MAC show Global Counter in COM port.

Defined in: gpon.h

Parameters

`type`

counter type

Comments

Return Codes

`RT_ERR_OK`

ok

others

fail

7.68. rtk_gpon_tcontCounter_show

`int32 rtk_gpon_tcontCounter_show(uint32 idx,
rtk_gpon_tcont_performance_type_t type)`

GPON MAC show TCont Counter in COM port.

Defined in: gpon.h

Parameters	<i>idx</i> TCont index
	<i>type</i> counter type

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.69. rtk_gpon_flowCounter_show

```
int32 rtk_gpon_flowCounter_show(uint32 idx,
                                rtk_gpon_flow_performance_type_t type)
```

GPON MAC show Flow Counter in COM port.

Defined in: gpon.h

Parameters	<i>idx</i> Flow index
	<i>type</i> counter type

Comments

Return Codes	RT_ERR_OK others	ok fail
---------------------	---------------------	------------

7.70. rtk_gpon_omci_tx

```
int32 rtk_gpon_omci_tx(rtk_gpon_omci_msg_t *omci)
```

Transmit OMCI message.

Defined in: gpon.h

Parameters	<i>*omci</i> pointer of OMCI message data
-------------------	--

Comments**Return Codes**

RT_ERR_OK	ok
others	fail

7.71. rtk_gpon_omci_rx

`int32 rtk_gpon_omci_rx(uint8 *buf, uint32 len)`

Receive OMCI message.

Defined in: gpon.h

Parameters

<i>*buf</i>	pointer of received data
<i>len</i>	received data length

Comments

Return Codes

RT_ERR_OK	ok
others	fail

7.72. rtk_gpon_test_set

`int32 rtk_gpon_test_set(uint32 data)`

set GPON MAC test register

Defined in: gpon.h

Parameters

<i>data</i>	register data
-------------	---------------

Comments

For debug used in user space.

Return Codes

RT_ERR_OK	ok
others	fail

7.73. rtk_gpon_test_get

`int32 rtk_gpon_test_get(uint32 *pData)`

get GPON MAC test register

Defined in: gpon.h

Parameters

`*pData`
returned register data

Comments

For debug used in user space.

Return Codes

RT_ERR_OK	ok
others	fail

7.74. rtk_gpon_unit_test

`int32 rtk_gpon_unit_test(uint32 id)`

gpon unit test

Defined in: gpon.h

Parameters

`id`
test id

Comments

For debug used in user space.

Return Codes

RT_ERR_OK	ok
others	fail

7.75. rtk_gpon_initial

`int32 rtk_gpon_initial(uint32 data)`

GPON initial, including driver init, device init and irq register.

Defined in: gpon.h

Parameters

	<i>data</i>	user data
Comments	Only used in user space.	
Return Codes	RT_ERR_OK others	ok fail

7.76. rtk_gpon_deinitial

`int32 rtk_gpon_deinitial(void)`

GPON deinitial, including driver deinit, device deinit and irq deregister.

Defined in: gpon.h

Parameters	<i>void</i>
Comments	Only used in user space.
Return Codes	RT_ERR_OK others

7.77. rtk_gpon_debug_set

`int32 rtk_gpon_debug_set(int32 enable)`

Turn on/off gpon debug print, for debug used.

Defined in: gpon.h

Parameters	<i>enable</i> turn on/off debug print
Comments	Only used in user space.
Return Codes	RT_ERR_OK others

7.78. rtk_gpon_autoTcont_set

`int32 rtk_gpon_autoTcont_set(int32 state)`

enable/disable tcont auto learning function

Defined in: gpon.h

Parameters

`state`
enable(1)/disable(0) state

Comments

It is used for debug in user space.

Return Codes

<code>RT_ERR_OK</code>	ok
others	fail

7.79. rtk_gpon_autoTcont_get

`int32 rtk_gpon_autoTcont_get(int32 *pState)`

get tcont auto learning state

Defined in: gpon.h

Parameters

`*pState`
enable(1)/disable(0) state

Comments

It is used for debug in user space.

Return Codes

<code>RT_ERR_OK</code>	ok
others	fail

7.80. rtk_gpon_autoBoh_set

`int32 rtk_gpon_autoBoh_set(int32 state)`

enable/disable BOH auto configure function

Defined in: gpon.h

Parameters

	<i>state</i>	enable(1)/disable(0) state
Comments	It is used for debug in user space.	
Return Codes	RT_ERR_OK	ok
	others	fail

7.81. rtk_gpon_autoBoh_get

	<code>int32 rtk_gpon_autoBoh_get(int32 *pState)</code>	
	get BOH auto configure state	
	Defined in: gpon.h	
Parameters	<i>*pState</i>	
	enable(1)/disable(0) state	
Comments	It is used for debug in user space.	
Return Codes	RT_ERR_OK	ok
	others	fail

7.82. rtk_gpon_eqdOffset_set

	<code>int32 rtk_gpon_eqdOffset_set(int32 offset)</code>	
	configure eqd offset	
	Defined in: gpon.h	
Parameters	<i>offset</i>	
	offset value	
Comments	It is used for debug in user space.	
Return Codes	RT_ERR_OK	ok
	others	fail

7.83. rtk_gpon_eqdOffset_get

`int32 rtk_gpon_eqdOffset_get(int32 *pOffset)`

get eqd offset

Defined in: gpon.h

Parameters

`*pOffset`
offset value

Comments

It is used for debug in user space.

Return Codes

RT_ERR_OK	ok
others	fail

7.84. rtk_gpon_pktGen_cfg_set

`int32 rtk_gpon_pktGen_cfg_set(uint32 item, uint32 tcont, uint32 len, uint32 gem, int32 omci)`

packet generator configure.

Defined in: gpon.h

Parameters

`item`
item number
`tcont`
tcont id
`len`
packet length
`gem`
gem port
`omci`
omci flag

Comments

Only for debug used

Return Codes

RT_ERR_OK	ok
others	fail

7.85. rtk_gpon_pktGen_buf_set

`int32 rtk_gpon_pktGen_buf_set(uint32 item, uint8 *buf, uint32 len)`

packet generator buffer.

Defined in: gpon.h

Parameters

`item`

item number

`*buf`

packet buffer

`len`

packet length

Comments

Only for debug used

Return Codes

`RT_ERR_OK`

ok

others

fail

8. Module I2C API

Filename: i2c.h

Description

The file includes the following modules and sub-modules

- (1) I2C control
- (2) I2C read/write

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

8.1. List of Symbols

Here is a list of all functions and variables in this module

i2c.h - Definition of I2C API
`rtk_i2c_init`
`rtk_i2c_enable_set`
`rtk_i2c_enable_get`
`rtk_i2c_width_set`
`rtk_i2c_width_get`

rtk_i2c_write
rtk_i2c_read

8.2. rtk_i2c_init

`int32 rtk_i2c_init(rtк_i2c_port_t i2cPort)`

Initialize i2c interface.

Defined in: i2c.h

Parameters

i2cPort

I2C port interface

Comments

Must initialize before calling any other APIs.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

8.3. rtk_i2c_enable_set

`int32 rtk_i2c_enable_set(rtк_i2c_port_t i2cPort, rtк_enable_t enable)`

Enable/Disable I2C interface.

Defined in: i2c.h

Parameters

i2cPort

I2C port interface

enable

enable/disable state

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

8.4. rtk_i2c_enable_get

`int32 rtk_i2c_enable_get(rtk_i2c_port_t i2cPort, rtk_enable_t *pEnable)`

Get I2C interface state.

Defined in: i2c.h

Parameters

i2cPort
I2C port interface

**pEnable*
the pointer of enable/disable state

Comments

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	

8.5. rtk_i2c_width_set

`int32 rtk_i2c_width_set(rtk_i2c_port_t i2cPort, rtk_i2c_width_t width)`

Set the data and address width of I2C interface.

Defined in: i2c.h

Parameters

i2cPort
I2C port interface

width
8

Comments

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	

8.6. rtk_i2c_width_get

`int32 rtk_i2c_width_get(rtk_i2c_port_t i2cPort, rtk_i2c_width_t *pWidth)`

Get the data and address width of I2C interface.

Defined in: i2c.h

Parameters

i2cPort
I2C port interface
**pWidth*
the pointer of width

Comments

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	

8.7. rtk_i2c_write

`int32 rtk_i2c_write(rtk_i2c_port_t i2cPort, uint32 devID, uint32 regAddr, uint32 data)`

I2c write data.

Defined in: i2c.h

Parameters

i2cPort
I2C port interface
devID
the device ID
regAddr
register address
data
data value

Comments

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	

8.8. rtk_i2c_read

```
int32 rtk_i2c_read(rtk_i2c_port_t i2cPort, uint32 devID, uint32 regAddr,
                    uint32 *pData)
```

I2c read data.

Defined in: i2c.h

Parameters	<i>i2cPort</i> I2C port interface <i>devID</i> the device ID <i>regAddr</i> register address <i>*pData</i> the pointer of returned data
-------------------	--

Comments

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT RT_ERR_NULL_POINTER	ok failed
---------------------	---	--------------

9. Module Init API

Filename: init.h

Description Initialize All Layers of RTK Module

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

9.1. List of Symbols

Here is a list of all functions and variables in this module

init.h - Definition of Init API

rtk_init
rtk_deinit

9.2. rtk_init

int32 rtk_init(void)

Initialize the specified device

Defined in: init.h

Parameters

void

Comments

INIT must be initialized before using all of APIs in each modules

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_UNIT_ID

invalid unit id

9.3. rtk_deinit

int32 rtk_deinit(void)

De-Initialize the driver, release irq

Defined in: init.h

Parameters

void

Comments

INIT must be initialized before using all of APIs in each modules

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

10. Module Interrupt APIs

Filename: intr.h

Description

The file have include the following module and sub-modules
(1) Interrupt parameter settings

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

10.1. List of Symbols

Here is a list of all functions and variables in this module

intr.h - Definition those public Interrupt APIs and its data type in the SDK.
rtk_intr_init
rtk_intr_polarity_set
rtk_intr_polarity_get
rtk_intr_imr_set
rtk_intr_imr_get
rtk_intr_ims_get
rtk_intr_ims_clear
rtk_intr_speedChangeStatus_get
rtk_intr_speedChangeStatus_clear
rtk_intr_linkupStatus_get
rtk_intr_linkupStatus_clear
rtk_intr_linkdownStatus_get
rtk_intr_linkdownStatus_clear
rtk_intr_gphyStatus_get
rtk_intr_gphyStatus_clear
rtk_intr_imr_restore

10.2. rtk_intr_init

`int32 rtk_intr_init(void)`

Initialize interrupt module.

Defined in: intr.h

Parameters

`void`

Comments Must initialize interrupt module before calling any interrupt APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

10.3. rtk_intr_polarity_set

`int32 rtk_intr_polarity_set(rtk_intr_polarity_t polar)`

Set interrupt polarity mode

Defined in: intr.h

Parameters *polar*
Interrupt polarity mode.

Comments The API can set Interrupt polarity mode. The modes that can be set as following:
- INTR_POLAR_HIGH
- INTR_POLAR_LOW

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

10.4. rtk_intr_polarity_get

`int32 rtk_intr_polarity_get(rtk_intr_polarity_t *pPolar)`

Get Interrupt polarity mode

Defined in: intr.h

Parameters **pPolar*
Interrupt polarity mode

Comments The API can get Interrupt polarity mode. The modes that can be got as following:
- INTR_POLAR_HIGH
- INTR_POLAR_LOW

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

10.5. rtk_intr_imr_set

`int32 rtk_intr_imr_set(rtk_intr_type_t intr, rtk_enable_t enable)`

Set interrupt mask.

Defined in: intr.h

Parameters

intr
interrupt type

enable
interrupt status

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_OUT_OF_RANGE</code>	Invalid input parameters.

10.6. rtk_intr_imr_get

`int32 rtk_intr_imr_get(rtk_intr_type_t intr, rtk_enable_t *pEnable)`

Get interrupt mask.

Defined in: intr.h

Parameters

intr
interrupt type
**pEnable*
pointer of return status

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	Invalid input parameters.
<code>RT_ERR_OUT_OF_RANGE</code>	

10.7. rtk_intr_ims_get

`int32 rtk_intr_ims_get(rtk_intr_type_t intr, rtk_enable_t *pStatus)`

Get interrupt status.

Defined in: intr.h

Parameters

intr
interrupt type

**pStatus*
pointer of return status of mask (for SW_INTR_TYPE_ALL) or

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Invalid input parameters.
RT_ERR_OUT_OF_RANGE	

10.8. rtk_intr_ims_clear

`int32 rtk_intr_ims_clear(rtk_intr_type_t intr)`

Clear interrupt status.

Defined in: intr.h

Parameters

intr
interrupt type

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_OUT_OF_RANGE	Invalid input parameters.

10.9. rtk_intr_speedChangeStatus_get

`int32 rtk_intr_speedChangeStatus_get(rtk_portmask_t *pPortMask)`

Get interrupt status of speed change.

Defined in: intr.h

Parameters

`*pPortMask`
pointer of return port status (bitmask)

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	Invalid input parameters.

10.10.rtk_intr_speedChangeStatus_clear

`int32 rtk_intr_speedChangeStatus_clear(void)`

Clear interrupt status of port speed change.

Defined in: intr.h

Parameters

`void`

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

10.11.rtk_intr_linkupStatus_get

`int32 rtk_intr_linkupStatus_get(rtk_portmask_t *pPortMask)`

Get interrupt status of linkup.

Defined in: intr.h

Parameters	<i>*pPortMask</i> pointer of return status
Comments	None.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NULL_POINTER Invalid input parameters.

10.12.rtk_intr_linkupStatus_clear

`int32 rtk_intr_linkupStatus_clear(void)`

Clear interrupt status of linkup.

Defined in: intr.h

Parameters	<i>void</i>
Comments	None.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed

10.13.rtk_intr_linkdownStatus_get

`int32 rtk_intr_linkdownStatus_get(rtk_portmask_t *pPortMask)`

Get interrupt status of linkdown.

Defined in: intr.h

Parameters	<i>*pPortMask</i> pointer of return status
Comments	None.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NULL_POINTER Invalid input parameters.

10.14.rtk_intr_linkdownStatus_clear

`int32 rtk_intr_linkdownStatus_clear(void)`

Clear interrupt status of linkdown.

Defined in: intr.h

Parameters `void`

Comments None.

Return Codes `RT_ERR_OK` ok
`RT_ERR_FAILED` failed

10.15.rtk_intr_gphyStatus_get

`int32 rtk_intr_gphyStatus_get(rt_k_portmask_t *pPortMask)`

Get interrupt status of GPHY.

Defined in: intr.h

Parameters `*pPortMask`
pointer of return status

Comments None.

Return Codes `RT_ERR_OK` ok
`RT_ERR_FAILED` failed
`RT_ERR_NULL_POINTER` Invalid input parameters.

10.16.rtk_intr_gphyStatus_clear

`int32 rtk_intr_gphyStatus_clear(void)`

Clear interrupt status of GPHY.

Defined in: intr.h

Parameters	<i>void</i>				
Comments	None.				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

10.17.rtk_intr_imr_restore

`int32 rtk_intr_imr_restore(uint32 imrValue)`

set imr mask from input value

Defined in: intr.h

Parameters	<i>imrValue</i> pointer of return status				
Comments	None.				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

11. Module IRQ API

Filename: irq.h

Description	Provide the APIs to register/deregisger switch IRQ
	Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

11.1. List of Symbols

Here is a list of all functions and variables in this module

irq.h - Definition of IRQ API
 rtk_switch_irq_init
 rtk_switch_irq_exit
 rtk_irq_isr_register
 rtk_irq_isr_unregister

11.2. rtk_switch_irq_init

`int32 rtk_switch_irq_init(uint32 irq_id)`

IRQ register

Defined in: irq.h

Parameters

irq_id
IRQ ID

Comments

11.3. rtk_switch_irq_exit

`int32 rtk_switch_irq_exit(void)`

IRQ deregister

Defined in: irq.h

Parameters

void

Comments

11.4. rtk_irq_isr_register

`int32 rtk_irq_isr_register(rtk_intr_type_t intr, void *fun)`

Register isr handler

Defined in: irq.h

Parameters

intr
interrupt type

**fun*
function pointer of isr hander

Comments

None.

Return Codes

`RT_ERR_OK`

ok

RT_ERR_FAILED	failed
---------------	--------

11.5. rtk_irq_isr_unregister

`int32 rtk_irq_isr_unregister(rtk_intr_type_t intr)`

Unregister isr handler

Defined in: irq.h

Parameters

intr
interrupt type

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

12. Module L2 API

Filename: l2.h

Description

The file includes the following modules and sub-modules

- (1) Mac address flush
- (2) Address learning limit
- (3) Parameter for L2 lookup and learning engine
- (4) Unicast address
- (5) L2 multicast
- (6) IP multicast
- (7) Multicast forwarding table
- (8) CPU mac
- (9) Port move
- (10) Parameter for lookup miss
- (11) Parameter for MISC

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

12.1. List of Symbols

Here is a list of all functions and variables in this module

l2.h - Definition of L2 API
rtk_l2_init
rtk_l2_flushLinkDownPortAddrEnable_get
rtk_l2_flushLinkDownPortAddrEnable_set
rtk_l2_unicastAddr_flush
rtk_l2_table_clear
rtk_l2_limitLearningOverStatus_get
rtk_l2_limitLearningOverStatus_clear
rtk_l2_learningCnt_get
rtk_l2_limitLearningCnt_get
rtk_l2_limitLearningCnt_set
rtk_l2_limitLearningCntAction_get
rtk_l2_limitLearningCntAction_set
rtk_l2_portLimitLearningOverStatus_get
rtk_l2_portLimitLearningOverStatus_clear
rtk_l2_portLearningCnt_get
rtk_l2_portLimitLearningCnt_get
rtk_l2_portLimitLearningCnt_set
rtk_l2_portLimitLearningCntAction_get
rtk_l2_portLimitLearningCntAction_set
rtk_l2_aging_get
rtk_l2_aging_set
rtk_l2_portAgingEnable_get
rtk_l2_portAgingEnable_set
rtk_l2_lookupMissAction_get
rtk_l2_lookupMissAction_set
rtk_l2_portLookupMissAction_get
rtk_l2_portLookupMissAction_set
rtk_l2_lookupMissFloodPortMask_get
rtk_l2_lookupMissFloodPortMask_set
rtk_l2_lookupMissFloodPortMask_add
rtk_l2_lookupMissFloodPortMask_del
rtk_l2_newMacOp_get
rtk_l2_newMacOp_set
rtk_l2_nextValidAddr_get
rtk_l2_nextValidAddrOnPort_get
rtk_l2_nextValidMcastAddr_get
rtk_l2_nextValidIpMcastAddr_get
rtk_l2_nextValidEntry_get
rtk_l2_addr_add
rtk_l2_addr_del
rtk_l2_addr_get
rtk_l2_addr_delAll
rtk_l2_mcastAddr_add
rtk_l2_mcastAddr_del

```

rtk_l2_mcastAddr_get
rtk_l2_illegalPortMoveAction_get
rtk_l2_illegalPortMoveAction_set
rtk_l2_ipmcMode_get
rtk_l2_ipmcMode_set
rtk_l2_ipmcGroupLookupMissHash_get
rtk_l2_ipmcGroupLookupMissHash_set
rtk_l2_ipmcGroup_add
rtk_l2_ipmcGroup_del
rtk_l2_ipmcGroup_get
rtk_l2_portIpmcAction_get
rtk_l2_portIpmcAction_set
rtk_l2_ipMcastAddr_add
rtk_l2_ipMcastAddr_del
rtk_l2_ipMcastAddr_get
rtk_l2_srcPortEgrFilterMask_get
rtk_l2_srcPortEgrFilterMask_set
rtk_l2_extPortEgrFilterMask_get
rtk_l2_extPortEgrFilterMask_set

```

12.2. rtk_l2_init

`int32 rtk_l2_init(void)`

Initialize l2 module of the specified device.

Defined in: l2.h

Parameters

`void`

Comments

Must initialize l2 module before calling any l2 APIs.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

12.3. rtk_l2_flushLinkDownPortAddrEnable_get

`int32 rtk_l2_flushLinkDownPortAddrEnable_get(rtk_enable_t *pEnable)`

Get HW flush linkdown port mac configuration.

	Defined in: l2.h
Parameters	<i>*pEnable</i> pointer buffer of state of HW clear linkdown port mac
Comments	(1) Make sure chip have supported the function before using the API. (2) The API is apply to whole system. (3) The status of flush linkdown port address is as following: - DISABLED - ENABLED
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NULL_POINTER input parameter may be null pointer

12.4. rtk_l2_flushLinkDownPortAddrEnable_set

`int32 rtk_l2_flushLinkDownPortAddrEnable_set(rt_k_enable_t enable)`

Set HW flush linkdown port mac configuration.

	Defined in: l2.h
Parameters	<i>enable</i> configure value
Comments	(1) Make sure chip have supported the function before using the API. (2) The API is apply to whole system. (3) The status of flush linkdown port address is as following: - DISABLED - ENABLED
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_INPUT invalid input parameter

12.5. rtk_l2_ucastAddr_flush

`int32 rtk_l2_ucastAddr_flush(rt_k_flushCfg_t *pConfig)`

Flush unicast address

Defined in: l2.h

Parameters	<i>*pConfig</i> flush config								
Comments	None								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>invalid input parameter</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_INPUT	invalid input parameter
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_NOT_INIT	The module is not initial								
RT_ERR_INPUT	invalid input parameter								

12.6. rtk_l2_table_clear

`int32 rtk_l2_table_clear(void)`

All the entries (static and dynamic) (L2 and L3) will be deleted.

Defined in: l2.h

Parameters	<i>void</i>						
Comments	None						
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_NOT_INIT	The module is not initial						

12.7. rtk_l2_limitLearningOverStatus_get

`int32 rtk_l2_limitLearningOverStatus_get(uint32 *pStatus)`

Get the system learning over status

Defined in: l2.h

Parameters	<i>*pStatus</i> 1: learning over, 0: not learning over		
Comments			
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> </table>	RT_ERR_OK	ok
RT_ERR_OK	ok		

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.8. rtk_l2_limitLearningOverStatus_clear

`int32 rtk_l2_limitLearningOverStatus_clear(void)`

Clear the system learning over status

Defined in: l2.h

Parameters `void`

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id

12.9. rtk_l2_learningCnt_get

`int32 rtk_l2_learningCnt_get(uint32 *pMacCnt)`

Get the total mac learning counts of the whole specified device.

Defined in: l2.h

Parameters `*pMacCnt`
pointer buffer of mac learning counts of the port

Comments (1) The mac learning counts only calculate dynamic mac numbers.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.10.rtk_l2_limitLearningCnt_get

`int32 rtk_l2_limitLearningCnt_get(uint32 *pMacCnt)`

Get the maximum mac learning counts of the specified device.

Defined in: l2.h

Parameters

`*pMacCnt`
pointer buffer of maximum mac learning counts

Comments

(1) The maximum mac learning counts only limit for dynamic learning mac address, not apply to static mac address.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

12.11.rtk_l2_limitLearningCnt_set

`int32 rtk_l2_limitLearningCnt_set(uint32 macCnt)`

Set the maximum mac learning counts of the specified device.

Defined in: l2.h

Parameters

`macCnt`
maximum mac learning counts

Comments

(1) The maximum mac learning counts only limit for dynamic learning mac address, not apply to static mac address.
(2) Set the macCnt to 0 mean disable learning in the system.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_LIMITED_L2ENTRY_NUM</code>	invalid limited L2 entry number

12.12.rtk_l2_limitLearningCntAction_get

```
int32 rtk_l2_limitLearningCntAction_get(rtk_l2_limitLearnCntAction_t
                                         *pLearningAction)
```

Get the action when over learning maximum mac counts of the specified device.

Defined in: l2.h

Parameters

**pLearningAction*

pointer buffer of action when over learning maximum mac counts

Comments

(1) The action symbol as following

- LIMIT_LEARN_CNT_ACTION_DROP
- LIMIT_LEARN_CNT_ACTION_FORWARD
- LIMIT_LEARN_CNT_ACTION_TO_CPU
- LIMIT_LEARN_CNT_ACTION_COPY_TO_CPU

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
---------------	--------

RT_ERR_NULL_POINTER	input parameter may be null pointer
---------------------	-------------------------------------

12.13.rtk_l2_limitLearningCntAction_set

```
int32 rtk_l2_limitLearningCntAction_set(rtk_l2_limitLearnCntAction_t
                                         learningAction)
```

Set the action when over learning maximum mac counts of the specified device.

Defined in: l2.h

Parameters

learningAction

action when over learning maximum mac counts

Comments

(1) The action symbol as following

- LIMIT_LEARN_CNT_ACTION_DROP
- LIMIT_LEARN_CNT_ACTION_FORWARD
- LIMIT_LEARN_CNT_ACTION_TO_CPU
- LIMIT_LEARN_CNT_ACTION_COPY_TO_CPU

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
---------------	--------

12.14.rtk_l2_portLimitLearningOverStatus_get

```
int32 rtk_l2_portLimitLearningOverStatus_get(rtk_port_t port, uint32
*pStatus)
```

Get the port learning over status

Defined in: l2.h

Parameters

port
Port ID

**pStatus*
1: learning over, 0: not learning over

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.15.rtk_l2_portLimitLearningOverStatus_clear

```
int32 rtk_l2_portLimitLearningOverStatus_clear(rtk_port_t port)
```

Clear the port learning over status

Defined in: l2.h

Parameters

port
Port ID

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id

12.16.rtk_l2_portLearningCnt_get

`int32 rtk_l2_portLearningCnt_get(rtk_port_t port, uint32 *pMacCnt)`

Get the mac learning counts of the port.

Defined in: l2.h

Parameters

port
port id
**pMacCnt*
pointer buffer of mac learning counts of the port

Comments

(1) The mac learning counts only calculate dynamic mac numbers.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_PORT_ID</code>	invalid port id
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

12.17.rtk_l2_portLimitLearningCnt_get

`int32 rtk_l2_portLimitLearningCnt_get(rtk_port_t port, uint32 *pMacCnt)`

Get the maximum mac learning counts of the port.

Defined in: l2.h

Parameters

port
port id
**pMacCnt*
pointer buffer of maximum mac learning counts

Comments

(1) The maximum mac learning counts only limit for dynamic learning mac address, not apply to static mac address.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_PORT_ID</code>	invalid port id
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

12.18.rtk_l2_portLimitLearningCnt_set

`int32 rtk_l2_portLimitLearningCnt_set(rtk_port_t port, uint32 macCnt)`

Set the maximum mac learning counts of the port.

Defined in: l2.h

Parameters

port
port id

macCnt
maximum mac learning counts

Comments

- (1) The maximum mac learning counts only limit for dynamic learning mac address, not apply to static mac address.
- (2) Set the macCnt to 0 mean disable learning in the port.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_PORT_ID</code>	invalid port id
<code>RT_ERR_LIMITED_L2ENTRY_NUM</code>	invalid limited L2 entry number

12.19.rtk_l2_portLimitLearningCntAction_get

`int32 rtk_l2_portLimitLearningCntAction_get(rtk_port_t port,
rtk_l2_limitLearnCntAction_t *pLearningAction)`

Get the action when over learning maximum mac counts of the port.

Defined in: l2.h

Parameters

port
port id

**pLearningAction*
pointer buffer of action when over learning maximum mac counts

Comments

The action symbol as following

- LIMIT_LEARN_CNT_ACTION_DROP
- LIMIT_LEARN_CNT_ACTION_FORWARD
- LIMIT_LEARN_CNT_ACTION_TO_CPU
- LIMIT_LEARN_CNT_ACTION_COPY_CPU

Return Codes

<code>RT_ERR_OK</code>	ok
------------------------	----

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.20.rtk_l2_portLimitLearningCntAction_set

`int32 rtk_l2_portLimitLearningCntAction_set(rtk_port_t port,
rtk_l2_limitLearnCntAction_t learningAction)`

Set the action when over learning maximum mac counts of the port.

Defined in: l2.h

Parameters

port

port id

learningAction

action when over learning maximum mac counts

Comments

The action symbol as following

- LIMIT_LEARN_CNT_ACTION_DROP
- LIMIT_LEARN_CNT_ACTION_FORWARD
- LIMIT_LEARN_CNT_ACTION_TO_CPU
- LIMIT_LEARN_CNT_ACTION_COPY_CPU

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_PORT_ID

invalid port id

12.21.rtk_l2_aging_get

`int32 rtk_l2_aging_get(uint32 *pAgingTime)`

Get the dynamic address aging time.

Defined in: l2.h

Parameters

**pAgingTime*

pointer buffer of aging time

Comments

Get aging_time as 0 mean disable aging mechanism. (0.1sec)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.22.rtk_l2_aging_set

`int32 rtk_l2_aging_set(uint32 agingTime)`

Set the dynamic address aging time.

Defined in: l2.h

Parameters *agingTime*
 aging time

Comments (1) RTL8329/RTL8389 aging time is not configurable.
(2) apply aging_time as 0 mean disable aging mechanism.

Return Codes RT_ERR_OK ok
 RT_ERR_FAILED failed
 RT_ERR_INPUT invalid input parameter

12.23.rtk_l2_portAgingEnable_get

`int32 rtk_l2_portAgingEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get the dynamic address aging out setting of the specified port.

Defined in: l2.h

Parameters *port*
 port id
 **pEnable*
 pointer to enable status of Age

Comments None

Return Codes RT_ERR_OK ok
 RT_ERR_FAILED failed
 RT_ERR_PORT_ID invalid port id
 RT_ERR_NULL_POINTER input parameter may be null pointer

12.24.rtk_l2_portAgingEnable_set

`int32 rtk_l2_portAgingEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set the dynamic address aging out configuration of the specified port

Defined in: l2.h

Parameters

port
port id
enable
enable status of Age

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	invalid input parameter

12.25.rtk_l2_lookupMissAction_get

`int32 rtk_l2_lookupMissAction_get(rtk_l2_lookupMissType_t type, rtk_action_t *pAction)`

Get forwarding action when destination address lookup miss.

Defined in: l2.h

Parameters

type
type of lookup miss
**pAction*
pointer to forwarding action

Comments

Type of lookup missis as following:
 - DLF_TYPE_IPMC
 - DLF_TYPE_UCAST
 - DLF_TYPE_BCAST
 - DLF_TYPE_MCAST

Forwarding action is as following:

- ACTION_DROP
- ACTION_TRAP2CPU
- ACTION_FLOOD_IN_VLAN
- ACTION_FLOOD_IN_ALL_PORT (only for DLF_TYPE_MCAST)
- ACTION_FLOOD_IN_ROUTER_PORTS (only for DLF_TYPE_IPMC)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid type of lookup miss
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.26.rtk_l2_lookupMissAction_set

```
int32 rtk_l2_lookupMissAction_set(rtk_l2_lookupMissType_t type,
                                  rtk_action_t action)
```

Set forwarding action when destination address lookup miss.

Defined in: l2.h

Parameters

type
type of lookup miss

action
forwarding action

Comments

Type of lookup missis as following:

- DLF_TYPE_IPMC
- DLF_TYPE_UCAST
- DLF_TYPE_BCAST
- DLF_TYPE_MCAST

Forwarding action is as following:

- ACTION_FORWARD
- ACTION_DROP
- ACTION_TRAP2CPU
- ACTION_COPY2CPU (only for DLF_TYPE_UCAST)
- ACTION_FLOOD_IN_ALL_PORT (only for DLF_TYPE_MCAST)
- ACTION_FLOOD_IN_ROUTER_PORTS (only for DLF_TYPE_IPMC)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid type of lookup miss
RT_ERR_FWD_ACTION	invalid forwarding action

12.27.rtk_l2_portLookupMissAction_get

`int32 rtk_l2_portLookupMissAction_get(rtk_port_t port,
rtk_l2_lookupMissType_t type, rtk_action_t *pAction)`

Get forwarding action of specified port when destination address lookup miss.

Defined in: l2.h

Parameters

port
port id
type
type of lookup miss

**pAction*
pointer to forwarding action

Comments

Type of lookup miss is as following:

- DLF_TYPE_IPMC
- DLF_TYPE_UCAST
- DLF_TYPE_MCAST
- DLF_TYPE_IP6MC

Forwarding action is as following:

- ACTION_DROP
- ACTION_TRAP2CPU
- ACTION_FORWARD
- ACTION_DROP_EXCLUDE_RMA (Only for DLF_TYPE_MCAST)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid type of lookup miss
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.28.rtk_l2_portLookupMissAction_set

```
int32 rtk_l2_portLookupMissAction_set(rtk_port_t port,
                                      rtk_l2_lookupMissType_t type, rtk_action_t action)
```

Set forwarding action of specified port when destination address lookup miss.

Defined in: l2.h

Parameters

port
port id

type
type of lookup miss

action
forwarding action

Comments

Type of lookup miss is as following:

- DLF_TYPE_IPMC
- DLF_TYPE_UCAST
- DLF_TYPE_MCAST
- DLF_TYPE_IP6MC

Forwarding action is as following:

- ACTION_DROP
- ACTION_TRAP2CPU
- ACTION_FORWARD
- ACTION_DROP_EXCLUDE_RMA (Only for DLF_TYPE_MCAST)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initialized
RT_ERR_INPUT	invalid type of lookup miss
RT_ERR_PORT_ID	invalid port id
RT_ERR_FWD_ACTION	invalid forwarding action

12.29.rtk_l2_lookupMissFloodPortMask_get

```
int32 rtk_l2_lookupMissFloodPortMask_get(rtk_l2_lookupMissType_t type,
                                         rtk_portmask_t *pFlood_portmask)
```

Get flooding port mask when unicast or multicast address lookup missed in L2 table.

Defined in: l2.h

Parameters	<i>type</i> type of lookup miss <i>*pFlood_portmask</i> flooding port mask configuration when unicast/multicast lookup missed.
Comments	DLF_TYPE_IPMC, DLF_TYPE_IP6MC & DLF_TYPE_MCAST shares the same configuration.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NOT_INIT The module is not initial RT_ERR_NULL_POINTER input parameter may be null pointer

12.30.rtk_l2_lookupMissFloodPortMask_set

```
int32 rtk_l2_lookupMissFloodPortMask_set(rtk_l2_lookupMissType_t type,  
rtk_portmask_t *pFlood_portmask)
```

Set flooding port mask when unicast or multicast address lookup missed in L2 table.

Defined in: l2.h

Parameters	<i>type</i> type of lookup miss <i>*pFlood_portmask</i> flooding port mask configuration when unicast/multicast lookup missed.
Comments	DLF_TYPE_IPMC, DLF_TYPE_IP6MC & DLF_TYPE_MCAST shares the same configuration.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NOT_INIT The module is not initial RT_ERR_NULL_POINTER input parameter may be null pointer

12.31.rtk_l2_lookupMissFloodPortMask_add

```
int32 rtk_l2_lookupMissFloodPortMask_add(rtk_l2_lookupMissType_t type,
                                         rtk_port_t flood_port)
```

Add one port member to flooding port mask when unicast or multicast address lookup missed in L2 table.

Defined in: l2.h

Parameters

type
type of lookup miss

flood_port
port id that is going to be added in flooding port mask.

Comments

DLF_TYPE_IPMC & DLF_TYPE_MCAST shares the same configuration.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.32.rtk_l2_lookupMissFloodPortMask_del

```
int32 rtk_l2_lookupMissFloodPortMask_del(rtk_l2_lookupMissType_t type,
                                         rtk_port_t flood_port)
```

Del one port member in flooding port mask when unicast or multicast address lookup missed in L2 table.

Defined in: l2.h

Parameters

type
type of lookup miss

flood_port
port id that is going to be added in flooding port mask.

Comments

DLF_TYPE_IPMC & DLF_TYPE_MCAST shares the same configuration..

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

RT_ERR_NULL_POINTER	input parameter may be null pointer
---------------------	-------------------------------------

12.33.rtk_l2_newMacOp_get

```
int32 rtk_l2_newMacOp_get(rtk_port_t port, rtk_l2_newMacLrnMode_t
    *pLrnMode, rtk_action_t *pFwdAction)
```

Get learning mode and forwarding action of new learned address on specified port.

Defined in: l2.h

Parameters

<i>port</i>	port id
<i>*pLrnMode</i>	pointer to learning mode
<i>*pFwdAction</i>	pointer to forwarding action

Comments

Forwarding action is as following
 - ACTION_FORWARD
 - ACTION_DROP
 - ACTION_TRAP2CPU
 - ACTION_COPY2CPU

Learning mode is as following
 - HARDWARE_LEARNING
 - SOFTWARE_LEARNING
 - NOT_LEARNING

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.34.rtk_l2_newMacOp_set

```
int32 rtk_l2_newMacOp_set(rtk_port_t port, rtk_l2_newMacLrnMode_t
    lrnMode, rtk_action_t fwdAction)
```

	Set learning mode and forwarding action of new learned address on specified port.													
	Defined in: l2.h													
Parameters	<p><i>port</i> port id</p> <p><i>lrnMode</i> learning mode</p> <p><i>fwdAction</i> forwarding action</p>													
Comments	<p>Forwarding action is as following</p> <ul style="list-style-type: none"> - ACTION_FORWARD - ACTION_DROP - ACTION_TRAP2CPU - ACTION_COPY2CPU <p>Learning mode is as following</p> <ul style="list-style-type: none"> - HARDWARE_LEARNING - SOFTWARE_LEARNING - NOT_LEARNING 													
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>invalid port id</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>invalid input parameter</td> </tr> <tr> <td>RT_ERR_FWD_ACTION</td> <td>invalid forwarding action</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_PORT_ID	invalid port id	RT_ERR_INPUT	invalid input parameter	RT_ERR_FWD_ACTION	invalid forwarding action
RT_ERR_OK	ok													
RT_ERR_FAILED	failed													
RT_ERR_NOT_INIT	The module is not initial													
RT_ERR_PORT_ID	invalid port id													
RT_ERR_INPUT	invalid input parameter													
RT_ERR_FWD_ACTION	invalid forwarding action													

12.35.rtk_l2_nextValidAddr_get

```
int32 rtk_l2_nextValidAddr_get(int32 *pScanIdx, rtk_l2_unicastAddr_t
*pL2UcastData)
```

Get next valid L2 unicast address entry from the specified device.

Defined in: l2.h

Parameters	* <i>pScanIdx</i> currently scan index of l2 table to get next.
	* <i>pL2UcastData</i> the get type, include static mac or not.

Comments

- (1) The function will skip valid l2 multicast and ip multicast entry and reply next valid L2 unicast address is based on index order of l2 table.
- (2) Please input 0 for get the first entry of l2 table.
- (3) The pScanIdx is the input and also is the output argument.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_MAC	invalid mac address
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_L2_ENTRY_NOTFOUND	specified entry not found

12.36.rtk_l2_nextValidAddrOnPort_get

```
int32 rtk_l2_nextValidAddrOnPort_get(rtk_port_t port, int32 *pScanIdx,
                                     rtk_l2_ucastAddr_t *pL2UcastData)
```

Get next valid L2 unicast address entry from specify port.

Defined in: l2.h

Parameters	<i>port</i>	currently scan index of l2 table to get next.
	<i>*pScanIdx</i>	structure of l2 address data
	<i>*pL2UcastData</i>	structure of l2 address data

Comments	(1) The function will skip valid l2 multicast and ip multicast entry and reply next valid L2 unicast address is based on index order of l2 table.
	(2) Please input 0 for get the first entry of l2 table.
	(3) The pScanIdx is the input and also is the output argument.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_MAC	invalid mac address
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_L2_ENTRY_NOTFOUND	specified entry not found

12.37.rtk_l2_nextValidMcastAddr_get

```
int32 rtk_l2_nextValidMcastAddr_get(int32 *pScanIdx, rtk_l2_mcastAddr_t
*pL2McastData)
```

Get next valid L2 multicast address entry from the specified device.

Defined in: l2.h

Parameters

**pScanIdx*
currently scan index of l2 table to get next.

**pL2McastData*
structure of l2 address data

Comments

- (1) The function will skip valid l2 unicast and ip multicast entry and reply next valid L2 multicast address is based on index order of l2 table.
- (2) Please input 0 for get the first entry of l2 table.
- (3) The pScan_idx is the input and also is the output argument.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_VLAN_VID	invalid vid
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_L2_ENTRY_NOTFOUND	specified entry not found

12.38.rtk_l2_nextValidIpMcastAddr_get

```
int32 rtk_l2_nextValidIpMcastAddr_get(int32 *pScanIdx,
rtk_l2_ipMcastAddr_t *pIpMcastData)
```

Get next valid L2 ip multicast address entry from the specified device.

Defined in: l2.h

Parameters

**pScanIdx*
currently scan index of l2 table to get next.

**pIpMcastData*
structure of l2 address data

Comments

- (1) The function will skip valid l2 unicast and multicast entry and reply next valid L2 ip multicast address is based on index order of l2 table.

- (2) Please input 0 for get the first entry of l2 table.
 (3) The pScan_idx is the input and also is the output argument.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_L2_ENTRY_NOTFOUND	specified entry not found

12.39.rtk_l2_nextValidEntry_get

```
int32 rtk_l2_nextValidEntry_get(int32 *pScanIdx, rtk_l2_addr_table_t
*pL2Entry)
```

Get LUT next valid entry.

Defined in: l2.h

Parameters	* <i>pScanIdx</i>
	Index field in the structure.

* <i>pL2Entry</i>
entry content

Comments This API is used to get next valid LUT entry.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_L2_EMPTY_ENTRY	Empty LUT entry.
	RT_ERR_INPUT	Invalid input parameters.

12.40.rtk_l2_addr_add

```
int32 rtk_l2_addr_add(rtk_l2_ucastAddr_t *pL2Addr)
```

Add L2 entry to ASIC.

Defined in: l2.h

Parameters	* <i>pL2Addr</i>
	L2 entry

Comments Need to initialize L2 entry before add it.

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_VLAN_VID RT_ERR_MAC RT_ERR_NULL_POINTER RT_ERR_INPUT	ok failed The module is not initial invalid vlan id invalid mac address input parameter may be null pointer invalid input parameter
---------------------	---	---

12.41.rtk_l2_addr_del

`int32 rtk_l2_addr_del(rtk_l2_unicastAddr_t *pL2Addr)`

Delete a L2 unicast address entry.

Defined in: l2.h

Parameters
`*pL2Addr`
L2 entry

Comments
If the mac has existed in the LUT, it will be deleted. Otherwise, it will return RT_ERR_L2_ENTRY_NOTFOUND.

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_VLAN_VID RT_ERR_MAC RT_ERR_L2_ENTRY_NOTFOUND	ok failed invalid vid invalid mac address specified entry not found
---------------------	---	---

12.42.rtk_l2_addr_get

`int32 rtk_l2_addr_get(rtk_l2_unicastAddr_t *pL2Addr)`

Get L2 entry based on specified vid and MAC address

Defined in: l2.h

Parameters
`*pL2Addr`
pointer to L2 entry

Comments

If the unicast mac address existed in LUT, it will return the port and fid where the mac is learned. Otherwise, it will return a RT_ERR_L2_ENTRY_NOTFOUND error.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vlan id
	RT_ERR_MAC	invalid mac address
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_L2_ENTRY_NOTFOUND	specified entry not found

12.43.rtk_l2_addr_delAll

`int32 rtk_l2_addr_delAll(uint32 includeStatic)`

Delete all L2 unicast address entry.

Defined in: l2.h

Parameters	<i>includeStatic</i>
	include static mac or not?

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

12.44.rtk_l2_mcastAddr_add

`int32 rtk_l2_mcastAddr_add(rtk_l2_mcastAddr_t *pMcastAddr)`

Add L2 multicast entry to ASIC.

Defined in: l2.h

Parameters	<i>*pMcastAddr</i>
	L2 multicast entry

Comments	Need to initialize L2 multicast entry before add it.
-----------------	--

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vlan id
RT_ERR_MAC	invalid mac address
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	invalid input parameter

12.45.rtk_l2_mcastAddr_del

`int32 rtk_l2_mcastAddr_del(rtk_l2_mcastAddr_t *pMcastAddr)`

Delete a L2 multicast address entry.

Defined in: l2.h

Parameters

`*pMcastAddr`
L2 multicast entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_L2_HASH_KEY	invalid L2 Hash key
RT_ERR_L2_EMPTY_ENTRY	the entry is empty(invalid)

12.46.rtk_l2_mcastAddr_get

`int32 rtk_l2_mcastAddr_get(rtk_l2_mcastAddr_t *pMcastAddr)`

Update content of L2 multicast entry.

Defined in: l2.h

Parameters

`*pMcastAddr`
L2 multicast entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vlan id
RT_ERR_MAC	invalid mac address
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	invalid input parameter

12.47.rtk_l2_illegalPortMoveAction_get

```
int32 rtk_l2_illegalPortMoveAction_get(rtk_port_t port, rtk_action_t
*pFwdAction)
```

Get forwarding action when illegal port moving happen on specified port.

Defined in: l2.h

Parameters

port
port id
**pFwdAction*
pointer to forwarding action

Comments

Forwarding action is as following
 - ACTION_FORWARD
 - ACTION_DROP
 - ACTION_TRAP2CPU
 - ACTION_COPY2CPU

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.48.rtk_l2_illegalPortMoveAction_set

```
int32 rtk_l2_illegalPortMoveAction_set(rtk_port_t port, rtk_action_t
fwdAction)
```

Set forwarding action when illegal port moving happen on specified port.

	Defined in: l2.h	
Parameters	<i>port</i>	port id
	<i>fwdAction</i>	forwarding action
Comments	Forwarding action is as following - ACTION_FORWARD - ACTION_DROP - ACTION_TRAP2CPU - ACTION_COPY2CPU	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_FWD_ACTION	invalid forwarding action

12.49.rtk_l2_ipmcMode_get

```
int32 rtk_l2_ipmcMode_get(rtk_l2_ipmcMode_t *pMode)
```

Get lookup mode of layer2 ip multicast switching.

Defined in: l2.h

Parameters	<i>*pMode</i>	pointer to lookup mode of layer2 ip multicast switching
Comments	Lookup mode of layer2 ip multicast switching is as following - LOOKUP_ON_DIP_AND_SIP - LOOKUP_ON_MAC_AND_VID_FID - LOOKUP_ON_DPI_AND_VID	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

12.50.rtk_l2_ipmcMode_set

`int32 rtk_l2_ipmcMode_set(rtk_l2_ipmcMode_t mode)`

Set lookup mode of layer2 ip multicast switching.

Defined in: l2.h

Parameters

mode
lookup mode of layer2 ip multicast switching

Comments

Lookup mode of layer2 ip multicast switching is as following
 - LOOKUP_ON_DIP_AND_SIP
 - LOOKUP_ON_MAC_AND_VID_FID
 - LOOKUP_ON_DPI_AND_VID

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid input parameter

12.51.rtk_l2_ipmcGroupLookupMissHash_get

`int32 rtk_l2_ipmcGroupLookupMissHash_get(rtk_l2_ipmcHashOp_t *pIpmcHash)`

Get Hash operation of IPv4 multicast packet which is not in IPMC Group Table.

Defined in: l2.h

Parameters

**pIpmcHash*
pointer of Hash operation of IPv4 multicast packet which is not in IPMC Group Table

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.52.rtk_l2_ipmcGroupLookupMissHash_set

`int32 rtk_l2_ipmcGroupLookupMissHash_set(rtk_l2_ipmcHashOp_t
ipmcHash)`

Set Hash operation of IPv4 multicast packet which is not in IPMC Group Table.

Defined in: l2.h

Parameters

ipmcHash

Hash operation of IPv4 multicast packet which is not in IPMC Group Table

Comments

None.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

12.53.rtk_l2_ipmcGroup_add

`int32 rtk_l2_ipmcGroup_add(ipaddr_t gip, rtk_portmask_t *pPortmask)`

Add an entry to IPMC Group Table.

Defined in: l2.h

Parameters

gip

Group IP

**pPortmask*

Group member port mask

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

RT_ERR_ENTRY_FULL

entry is full

12.54.rtk_l2_ipmcGroup_del

`int32 rtk_l2_ipmcGroup_del(ipaddr_t gip)`

Delete an entry from IPMC Group Table.

Defined in: l2.h

Parameters

gip
Group IP

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer
<code>RT_ERR_ENTRY_NOTFOUND</code>	specified entry not found

12.55.rtk_l2_ipmcGroup_get

`int32 rtk_l2_ipmcGroup_get(ipaddr_t gip, rtk_portmask_t *pPortmask)`

Get an entry from IPMC Group Table.

Defined in: l2.h

Parameters

gip
Group IP
**pPortmask*
Group member port mask

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer
<code>RT_ERR_ENTRY_NOTFOUND</code>	specified entry not found

12.56.rtk_l2_portIpmcAction_get

`int32 rtk_l2_portIpmcAction_get(rtk_port_t port, rtk_action_t *pAction)`

Get the Action of IPMC packet per ingress port.

Defined in: l2.h

Parameters

port
Ingress port number

**pAction*
IPMC packet action (ACTION_FORWARD or ACTION_DROP)

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

12.57.rtk_l2_portIpmcAction_set

`int32 rtk_l2_portIpmcAction_set(rtk_port_t port, rtk_action_t action)`

Set the Action of IPMC packet per ingress port.

Defined in: l2.h

Parameters

port
Ingress port number

action
IPMC packet action (ACTION_FORWARD or ACTION_DROP)

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	Invalid input parameter

12.58.rtk_l2_ipMcastAddr_add

`int32 rtk_l2_ipMcastAddr_add(rtk_l2_ipMcastAddr_t *pIpMcastAddr)`

Add IP multicast entry to ASIC.

Defined in: l2.h

Parameters `*pIpMcastAddr`
 IP multicast entry

Comments Need to initialize IP multicast entry before add it.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_IPV4_ADDRESS	Invalid IPv4 address
	RT_ERR_VLAN_VID	invalid vlan id
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_INPUT	invalid input parameter

12.59.rtk_l2_ipMcastAddr_del

`int32 rtk_l2_ipMcastAddr_del(rtk_l2_ipMcastAddr_t *pIpMcastAddr)`

Delete a L2 ip multicast address entry from the specified device.

Defined in: l2.h

Parameters `*pIpMcastAddr`
 IP multicast entry

Comments (1) In vlan unaware mode (SVL), the vid will be ignore, suggest to input vid=0 in vlan unaware mode.
 (2) In vlan aware mode (IVL), the vid will be care.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_L2_HASH_KEY	invalid L2 Hash key
	RT_ERR_L2_EMPTY_ENTRY	the entry is empty(invalid)

12.60.rtk_l2_ipMcastAddr_get

`int32 rtk_l2_ipMcastAddr_get(rtk_l2_ipMcastAddr_t *pIpMcastAddr)`

Get IP multicast entry on specified dip and sip.

Defined in: l2.h

Parameters

`*pIpMcastAddr`
IP multicast entry

Comments

Need to initialize IP multicast entry before add it.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_IPV4_ADDRESS</code>	Invalid IPv4 address
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

12.61.rtk_l2_srcPortEgrFilterMask_get

`int32 rtk_l2_srcPortEgrFilterMask_get(rtk_portmask_t *pFilter_portmask)`

when packet egress.

Defined in: l2.h

Parameters

`*pFilter_portmask`
source port egress filtering configuration when packet egress.

Comments

May be used when wireless device connected. Get permission status for frames if its source port is equal to destination port.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

12.62.rtk_l2_srcPortEgrFilterMask_set

`int32 rtk_l2_srcPortEgrFilterMask_set(rtk_portmask_t *pFilter_portmask)`

when packet egress.

Defined in: l2.h

Parameters

`*pFilter_portmask`

source port egress filtering configuration when packet egress.

Comments

May be used when wireless device connected

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_NOT_INIT`

The module is not initial

`RT_ERR_NULL_POINTER`

input parameter may be null pointer

12.63.rtk_l2_extPortEgrFilterMask_get

`int32 rtk_l2_extPortEgrFilterMask_get(rtk_portmask_t *pExt_portmask)`

when packet egress.

Defined in: l2.h

Parameters

`*pExt_portmask`

extension port egress filtering configuration when packet egress.

Comments

May be used when wireless device connected. Get permission status for frames if its source port is equal to destination port.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_NOT_INIT`

The module is not initial

`RT_ERR_NULL_POINTER`

input parameter may be null pointer

12.64.rtk_l2_extPortEgrFilterMask_set

`int32 rtk_l2_extPortEgrFilterMask_set(rtk_portmask_t *pExt_portmask)`

when packet egress.

Defined in: l2.h

Parameters

`*pExt_portmask`
extension port egress filtering configuration when packet egress.

Comments

May be used when wireless device connected

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

13. Module LED APIs

Filename: led.h

Description

The file have include the following module and sub-modules

(1) LED parameter settings

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

13.1. List of Symbols

Here is a list of all functions and variables in this module

led.h - Definition those public LED APIs and its data type in the SDK.
`rtk_led_init`
`rtk_led_operation_get`
`rtk_led_operation_set`
`rtk_led_serialMode_get`
`rtk_led_serialMode_set`
`rtk_led_blinkRate_get`
`rtk_led_blinkRate_set`
`rtk_led_config_set`
`rtk_led_config_get`

```
rtk_led_modeForce_get
rtk_led_modeForce_set
rtk_led_parallelEnable_get
rtk_led_parallelEnable_set
```

13.2. rtk_led_init

int32 rtk_led_init(void)

Initialize led module.

Defined in: led.h

Parameters

void

Comments

Must initialize led module before calling any led APIs.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

13.3. rtk_led_operation_get

int32 rtk_led_operation_get(rt_kit_led_operation_t *pMode)

Get Led operation mode

Defined in: led.h

Parameters

<i>*pMode</i>	LED operation mode.
---------------	---------------------

Comments

The API can set Led operation mode. The modes that can be set are as following:

- LED_OP_PARALLEL 17 led
- LED_OP_SERIAL 32 led

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

13.4. rtk_led_operation_set

`int32 rtk_led_operation_set(rtk_led_operation_t mode)`

Set Led operation mode

Defined in: led.h

Parameters

mode
LED operation mode.

Comments

The API can set Led operation mode. The modes that can be set are as following:

- LED_OP_PARALLEL 17 led
- LED_OP_SERIAL 32 led

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	Invalid input parameters.

13.5. rtk_led_serialMode_get

`int32 rtk_led_serialMode_get(rtk_led_active_t *pActive)`

Set Led serial mode active congiuration

Defined in: led.h

Parameters

**pActive*
high low active mode.

Comments

The API can set LED serial mode active congiuration.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	Invalid input parameters.

13.6. rtk_led_serialMode_set

`int32 rtk_led_serialMode_set(rtk_led_active_t active)`

Set Led serial mode active congiuration

Defined in: led.h

Parameters

active
high low active mode.

Comments

The API can set LED serial mode active congiuration.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

13.7. rtk_led_blinkRate_get

```
int32 rtk_led_blinkRate_get(rtk_led_blinkGroup_t group,
                           rtk_led_blink_rate_t *pBlinkRate)
```

Get LED blinking rate at mode 0 to mode 3

Defined in: led.h

Parameters

<i>group</i>	led blinking group
<i>*pBlinkRate</i>	blinking rate.

Comments

There are 8 types of LED blinking rates at 32ms, 48ms, 64ms, 96ms, 128ms, 256ms, 512ms, and 1024ms.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

13.8. rtk_led_blinkRate_set

```
int32 rtk_led_blinkRate_set(rtk_led_blinkGroup_t group,
                           rtk_led_blink_rate_t blinkRate)
```

Get LED blinking rate

Defined in: led.h

Parameters	<i>group</i> led blinking group <i>blinkRate</i> blinking rate.
Comments	There are 8 types of LED blinking rates at 32ms, 48ms, 64ms, 96ms, 128ms, 256ms, 512ms, and 1024ms.
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT

ok
failed
Invalid input parameters.

13.9. rtk_led_config_set

```
int32 rtk_led_config_set(uint32 ledIdx, rtk_led_type_t type, rtk_led_config_t *pConfig)
```

Set per group Led to configuration mode

Defined in: led.h

Parameters	<i>ledIdx</i> LED index id. <i>type</i> LED type <i>*pConfig</i> LED configuration
-------------------	---

Comments There are 8 types of LED blinking rates at 32ms, 48ms, 64ms, 96ms, 128ms, 256ms, 512ms, and 1024ms.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

13.10.rtk_led_config_get

```
int32 rtk_led_config_get(uint32 ledIdx, rtk_led_type_t *pType,  
rtk_led_config_t *pConfig)
```

	Set per group Led to configuration mode							
	Defined in: led.h							
Parameters	<p><i>ledIdx</i> LED index id.</p> <p><i>*pType</i> LED type</p> <p><i>*pConfig</i> LED configuration</p>							
Comments	There are 8 types of LED blinking rates at 32ms, 48ms, 64ms, 96ms, 128ms, 256ms, 512ms, and 1024ms.							
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok							
RT_ERR_FAILED	failed							
RT_ERR_INPUT	Invalid input parameters.							

13.11.rtk_led_modeForce_get

```
int32 rtk_led_modeForce_get(uint32 ledIdx, rtk_led_force_mode_t *pMode)
```

Get Led group to configuration force mode

Defined in: led.h

Parameters	<i>ledIdx</i> LED index id.	<i>*pMode</i> LED force mode.						
Comments	The API can get forced Led group mode. The force modes that can be set are as following: - LED_FORCE_BLINK, - LED_FORCE_OFF, - LED_FORCE_ON. For LED_OP_SERIAL the max led index is 31 For LED_OP_PARALLEL the max led index is 16							
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok							
RT_ERR_FAILED	failed							
RT_ERR_INPUT	Invalid input parameters.							

13.12.rtk_led_modeForce_set

`int32 rtk_led_modeForce_set(uint32 ledIdx, rtk_led_force_mode_t mode)`

Set Led group to configuration force mode

Defined in: led.h

Parameters

ledIdx

LED index id.

mode

LED force mode.

Comments

The API can get forced Led group mode. The force modes that can be set are as following:

- LED_FORCE_BLINK,
- LED_FORCE_OFF,
- LED_FORCE_ON. For LED_OP_SERIAL the max led index is 31 For LED_OP_PARALLEL the max led index is 16

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

Invalid input parameters.

13.13.rtk_led_parallelEnable_get

`int32 rtk_led_parallelEnable_get(uint32 ledIdx, rtk_enable_t *pState)`

Get Led group enable status for parallel mode

Defined in: led.h

Parameters

ledIdx

LED index id.

**pState*

LED parallel enable status.

Comments

The API can get forced Led group mode. The force modes that can be set are as following:

- LED_FORCE_BLINK,
- LED_FORCE_OFF,
- LED_FORCE_ON. For LED_OP_SERIAL the max led index is 31 For LED_OP_PARALLEL the max led index is 16

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

13.14.rtk_led_parallelEnable_set

`int32 rtk_led_parallelEnable_set(uint32 ledIdx, rtk_enable_t state)`

Set Led group enable status for parallel mode

Defined in: led.h

Parameters	<i>ledIdx</i>	LED index id.
	<i>state</i>	LED parallel enable status.

Comments
The API can get forced Led group mode. The force modes that can be set are as following:

- LED_FORCE_BLINK,
- LED_FORCE_OFF,
- LED_FORCE_ON. For LED_OP_SERIAL the max led index is 31 For LED_OP_PARALLEL the max led index is 16

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

14. Module Mirror API

Filename: mirror.h

Description
The file includes the following modules and sub-modules
(1) Port-based mirror

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

14.1. List of Symbols

Here is a list of all functions and variables in this module

mirror.h - Definition of Mirror API
 rtk_mirror_init
 rtk_mirror_portBased_set
 rtk_mirror_portBased_get
 rtk_mirror_portIso_set
 rtk_mirror_portIso_get

14.2. rtk_mirror_init

int32 rtk_mirror_init(void)

Initialize the mirroring database.

Defined in: mirror.h

Parameters

void

Comments

Must initialize Mirror module before calling any Mirror APIs.

Return Codes

RT_ERR_OK

ok

14.3. rtk_mirror_portBased_set

int32 rtk_mirror_portBased_set(rt_k_port_t *mirroringPort*, rt_k_portmask_t **pMirroredRxPortmask*, rt_k_portmask_t **pMirroredTxPortmask*)

Set port mirror function.

Defined in: mirror.h

Parameters

mirroringPort

Monitor port.

**pMirroredRxPortmask*

Rx mirror port mask.

**pMirroredTxPortmask*

Tx mirror port mask.

Comments

The API is to set mirror function of source port and mirror port. The mirror port can only be set to one port and the TX and RX mirror ports should be identical.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port number
RT_ERR_PORT_MASK	Invalid portmask.

14.4. rtk_mirror_portBased_get

`int32 rtk_mirror_portBased_get(rt_k_port_t *pMirroringPort, rt_k_portmask_t *pMirroredRxPortmask, rt_k_portmask_t *pMirroredTxPortmask)`

Get port mirror function.

Defined in: mirror.h

Parameters

`*pMirroringPort`
Monitor port.
`*pMirroredRxPortmask`
Rx mirror port mask.
`*pMirroredTxPortmask`
Tx mirror port mask.

Comments

The API is to get mirror function of source port and mirror port.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

14.5. rtk_mirror_portIso_set

`int32 rtk_mirror_portIso_set(rt_k_enable_t enable)`

Set mirror port isolation.

Defined in: mirror.h

Parameters

`enable`
Monitor port.

Comments

The API is to set mirror isolation function that prevent normal forwarding packets to mirror port.

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_ENABLE	Invalid enable input

14.6. rtk_mirror_portIso_get

`int32 rtk_mirror_portIso_get(rtk_enable_t *pEnable)`

Get mirror port isolation.

Defined in: mirror.h

Parameters

`*pEnable`

Monitor port.

Comments

The API is to get mirror isolation status.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

Invalid input parameters.

15. Module OAM API

Filename: oam.h

Description

The file includes the following modules and sub-modules

(1) OAM (802.3ah) configuration

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

15.1. List of Symbols

Here is a list of all functions and variables in this module

oam.h - Definition of OAM API
 rtk_oam_init
 rtk_oam_parserAction_set
 rtk_oam_parserAction_get
 rtk_oam_multiplexerAction_set
 rtk_oam_multiplexerAction_get

15.2. rtk_oam_init

`int32 rtk_oam_init(void)`

Initialize oam module.

Defined in: oam.h

Parameters

`void`

Comments

Must initialize oam module before calling any oam APIs.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

15.3. rtk_oam_parserAction_set

`int32 rtk_oam_parserAction_set(rtk_port_t port, rtk_oam_parser_act_t action)`

Set OAM parser action

Defined in: oam.h

Parameters

`port`

port id

`action`

parser action

Comments

None

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_PORT_ID`

invalid port id

15.4. rtk_oam_parserAction_get

`int32 rtk_oam_parserAction_get(rtk_port_t port, rtk_oam_parser_act_t *pAction)`

	Get OAM parser action						
	Defined in: oam.h						
Parameters	<p><i>port</i> port id</p> <p><i>*pAction</i> parser action</p>						
Comments	None						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>invalid port id</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	invalid port id
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_PORT_ID	invalid port id						

15.5. rtk_oam_multiplexerAction_set

```
int32 rtk_oam_multiplexerAction_set(rtk_port_t port,  
rtk_oam_multiplexer_act_t action)
```

Set OAM multiplexer action

Defined in: oam.h

	Defined in: oam.h						
Parameters	<p><i>port</i> port id</p> <p><i>action</i> parser action</p>						
Comments	None						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>invalid port id</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	invalid port id
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_PORT_ID	invalid port id						

15.6. rtk_oam_multiplexerAction_get

```
int32 rtk_oam_multiplexerAction_get(rtk_port_t port,  
rtk_oam_multiplexer_act_t *pAction)
```

Get OAM multiplexer action

	Defined in: oam.h
Parameters	<p><i>port</i> port id</p> <p><i>*pAction</i> parser action</p>
Comments	None
Return Codes	<p>RT_ERR_OK ok</p> <p>RT_ERR_FAILED failed</p> <p>RT_ERR_PORT_ID invalid port id</p>

16. Module PON MAC API

Filename: ponmac.h

Description The file includes the following modules and sub-modules
 (1) queue configuration (PIR/CIR/Queue schuedule type)
 (2) flow and queue mapping

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

16.1. List of Symbols

Here is a list of all functions and variables in this module

ponmac.h - Definition of PON MAC API
 rtk_ponmac_init
 rtk_ponmac_queue_add
 rtk_ponmac_queue_get
 rtk_ponmac_queue_del
 rtk_ponmac_flow2Queue_set
 rtk_ponmac_flow2Queue_get
 rtk_ponmac_mode_set
 rtk_ponmac_mode_get

16.2. rtk_ponmac_init

`int32 rtk_ponmac_init(void)`

Configure PON MAC initial settings

Defined in: ponmac.h

Parameters *void*

Comments

Return Codes RT_ERR_OK
RT_ERR_FAILED

ok
failed

16.3. rtk_ponmac_queue_add

```
int32 rtk_ponmac_queue_add(rtk_ponmac_queue_t *pQueue,
                           rtk_ponmac_queueCfg_t *pQueueCfg)
```

Add queue to given scheduler id and apply queue setting

Defined in: ponmac.h

Parameters **pQueue*
queue id and scheduler id for ths queue.
**pQueueCfg*
queue configuration

Comments None

Return Codes RT_ERR_OK
RT_ERR_FAILED
RT_ERR_NULL_POINTER
RT_ERR_INPUT

ok
failed
Pointer pQueueList point to NULL.
Invalid input parameters.

16.4. rtk_ponmac_queue_get

```
int32 rtk_ponmac_queue_get(rtk_ponmac_queue_t *pQueue,
                           rtk_ponmac_queueCfg_t *pQueueCfg)
```

get queue setting

Defined in: ponmac.h

Parameters	<p><i>*pQueue</i> queue id and scheduler id for ths queue.</p> <p><i>*pQueueCfg</i> queue configuration</p>								
Comments	None								
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>Pointer pQueueList point to NULL.</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NULL_POINTER	Pointer pQueueList point to NULL.	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_NULL_POINTER	Pointer pQueueList point to NULL.								
RT_ERR_INPUT	Invalid input parameters.								

16.5. rtk_ponmac_queue_del

`int32 rtk_ponmac_queue_del(rtk_ponmac_queue_t *pQueue)`

delete queue from given scheduler id

Defined in: ponmac.h

Parameters	<p><i>*pQueue</i> queue id and scheduler id for ths queue.</p>						
Comments	None						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_INPUT	Invalid input parameters.						

16.6. rtk_ponmac_flow2Queue_set

`int32 rtk_ponmac_flow2Queue_set(uint32 flow, rtk_ponmac_queue_t *pQueue)`

mapping flow to given queue

Defined in: ponmac.h

Parameters	<p><i>flow</i> flow id.</p>
-------------------	---------------------------------

	<i>*pQueue</i>	queue id.
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT	ok failed Invalid input parameters.

16.7. rtk_ponmac_flow2Queue_get

`int32 rtk_ponmac_flow2Queue_get(uint32 flow, rtk_ponmac_queue_t *pQueue)`

get queue id for this flow

Defined in: ponmac.h

Parameters	<i>flow</i>	flow id.
	<i>*pQueue</i>	queue id.
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER RT_ERR_INPUT	ok failed Pointer pQueue point to NULL. Invalid input parameters.

16.8. rtk_ponmac_mode_set

`int32 rtk_ponmac_mode_set(rtk_ponmac_mode_t mode, rtk_enable_t state)`

set pon mac mode

Defined in: ponmac.h

Parameters	<i>mode</i>	
	pon mode, epon or gpon	

	<i>state</i>	disable or enable
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER RT_ERR_INPUT	ok failed Pointer pQueue point to NULL. Invalid input parameters.

16.9. rtk_ponmac_mode_get

`int32 rtk_ponmac_mode_get(rtk_ponmac_mode_t mode, rtk_enable_t *pState)`

set pon mac mode

Defined in: ponmac.h

Parameters	<i>mode</i>	pon mode, epon or gpon
	<i>*pState</i>	point of state, enable or disable
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER RT_ERR_INPUT	ok failed Pointer pQueue point to NULL. Invalid input parameters.

17. Module Port API

Filename: port.h

Description	The file have include the following module and sub-modules (1) Parameter settings for the port-based view (2) RTCT
--------------------	--

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

17.1. List of Symbols

Here is a list of all functions and variables in this module

port.h - Definition of Port API
rtk_port_init
rtk_port_link_get
rtk_port_speedDuplex_get
rtk_port_flowctrl_get
rtk_port_phyAutoNegoEnable_get
rtk_port_phyAutoNegoEnable_set
rtk_port_phyAutoNegoAbility_get
rtk_port_phyAutoNegoAbility_set
rtk_port_phyForceModeAbility_get
rtk_port_phyForceModeAbility_set
rtk_port_phyReg_get
rtk_port_phyReg_set
rtk_port_phyMasterSlave_get
rtk_port_phyMasterSlave_set
rtk_port_phyTestMode_get
rtk_port_phyTestMode_set
rtk_port_cpuPortId_get
rtk_port_isolation_get
rtk_port_isolation_set
rtk_port_isolationExt_get
rtk_port_isolationExt_set
rtk_port_isolationL34_get
rtk_port_isolationL34_set
rtk_port_isolationExtL34_get
rtk_port_isolationExtL34_set
rtk_port_isolationEntry_get
rtk_port_isolationEntry_set
rtk_port_isolationEntryExt_get
rtk_port_isolationEntryExt_set
rtk_port_isolationCtagPktConfig_get
rtk_port_isolationCtagPktConfig_set
rtk_port_isolationL34PktConfig_get
rtk_port_isolationL34PktConfig_set
rtk_port_isolationIpmcLeaky_get
rtk_port_isolationIpmcLeaky_set
rtk_port_isolationPortLeaky_get
rtk_port_isolationPortLeaky_set
rtk_port_isolationLeaky_get
rtk_port_isolationLeaky_set
rtk_port_macRemoteLoopbackEnable_get
rtk_port_macRemoteLoopbackEnable_set

rtk_port_macLocalLoopbackEnable_get
rtk_port_macLocalLoopbackEnable_set
rtk_port_adminEnable_get
rtk_port_adminEnable_set
rtk_port_specialCongest_get
rtk_port_specialCongest_set
rtk_port_specialCongestStatus_get
rtk_port_specialCongestStatus_clear
rtk_port_greenEnable_get
rtk_port_greenEnable_set
rtk_port_phyCrossOverMode_get
rtk_port_phyCrossOverMode_set
rtk_port_enhancedFid_get
rtk_port_enhancedFid_set
rtk_port_rtctResult_get
rtk_port_rtct_start
rtk_port_macForceAbility_set
rtk_port_macForceAbility_get
rtk_port_macForceAbilityState_set
rtk_port_macForceAbilityState_get
rtk_port_macExtMode_set
rtk_port_macExtMode_get
rtk_port_macExtRgmiiDelay_set
rtk_port_macExtRgmiiDelay_get

17.2. rtk_port_init

`int32 rtk_port_init(void)`

Initialize port module of the specified device.

Defined in: port.h

Parameters `void`

Comments Module must be initialized before using all of APIs in this module

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

17.3. rtk_port_link_get

`int32 rtk_port_link_get(rtk_port_t port, rtk_port_linkStatus_t *pLinkStatus)`

Get the link status of the specific port

Defined in: port.h

Parameters

port
port id

**pLinkStatus*
pointer to the link status

Comments

The link status of the port is as following:

- LINKDOWN
- LINKUP

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_PORT_ID</code>	invalid port id
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

17.4. rtk_port_speedDuplex_get

`int32 rtk_port_speedDuplex_get(rtk_port_t port, rtk_port_speed_t *pSpeed,
rtk_port_duplex_t *pDuplex)`

Get the negotiated port speed and duplex status of the specific port

Defined in: port.h

Parameters

port
port id

**pSpeed*
pointer to the port speed

**pDuplex*
pointer to the port duplex

Comments

(1) The speed type of the port is as following:

- PORT_SPEED_10M
- PORT_SPEED_100M
- PORT_SPEED_1000M

(2) The duplex mode of the port is as following:

- HALF_DUPLEX
- FULL_DUPLEX

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_UNIT_ID	invalid unit id
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_PORT_LINKDOWN	link down port status

17.5. rtk_port_flowctrl_get

```
int32 rtk_port_flowctrl_get(rtk_port_t port, uint32 *pTxStatus, uint32
*pRxStatus)
```

Get the negotiated flow control status of the specific port

Defined in: port.h

Parameters	<i>port</i>	port id
	<i>*pTxStatus</i>	pointer to the negotiation result of the Tx flow control
	<i>*pRxStatus</i>	pointer to the negotiation result of the Rx flow control

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer
	RT_ERR_PORT_LINKDOWN	link down port status

17.6. rtk_port_phyAutoNegoEnable_get

```
int32 rtk_port_phyAutoNegoEnable_get(rtk_port_t port, rtk_enable_t
*pEnable)
```

Get PHY ability of the specific port

Defined in: port.h

Parameters

port
port id
**pEnable*
pointer to PHY auto negotiation status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.7. rtk_port_phyAutoNegoEnable_set

```
int32 rtk_port_phyAutoNegoEnable_set(rtk_port_t port, rtk_enable_t enable)
```

Set PHY ability of the specific port

Defined in: port.h

Parameters

port
port id
enable
enable PHY auto negotiation

Comments

ENABLED : switch to PHY auto negotiation mode
 - DISABLED: switch to PHY force mode
 - Once the abilities of both auto-nego and force mode are set, you can freely
 switch the mode without calling ability setting API again

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	invalid unit id

RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	input parameter out of range

17.8. rtk_port_phyAutoNegoAbility_get

`int32 rtk_port_phyAutoNegoAbility_get(rtk_port_t port,
rtk_port_phy_ability_t *pAbility)`

Get PHY auto negotiation ability of the specific port

Defined in: port.h

Parameters

port
port id
**pAbility*
pointer to the PHY ability

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.9. rtk_port_phyAutoNegoAbility_set

`int32 rtk_port_phyAutoNegoAbility_set(rtk_port_t port,
rtk_port_phy_ability_t *pAbility)`

Set PHY auto negotiation ability of the specific port

Defined in: port.h

Parameters

port
port id
**pAbility*
pointer to the PHY ability

Comments

You can set these abilities no matter which mode PHY currently stays on

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.10.rtk_port_phyForceModeAbility_get

`int32 rtk_port_phyForceModeAbility_get(rtk_port_t port, rtk_port_speed_t *pSpeed, rtk_port_duplex_t *pDuplex, rtk_enable_t *pFlowControl)`

Get PHY ability status of the specific port

Defined in: port.h

Parameters

port
port id
**pSpeed*
pointer to the port speed
**pDuplex*
pointer to the port duplex
**pFlowControl*
pointer to the flow control enable status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.11.rtk_port_phyForceModeAbility_set

`int32 rtk_port_phyForceModeAbility_set(rtk_port_t port, rtk_port_speed_t speed, rtk_port_duplex_t duplex, rtk_enable_t flowControl)`

Set the port speed/duplex mode/pause/asy_pause in the PHY force mode

Defined in: port.h

Parameters

port
port id

speed
 port speed
duplex
 port duplex mode
flowControl
 enable flow control

Comments

(1) You can set these abilities no matter which mode PHY currently stays on

(2) The speed type of the port is as following:

- PORT_SPEED_10M
- PORT_SPEED_100M

(3) The duplex mode of the port is as following:

- HALF_DUPLEX
- FULL_DUPLEX

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_PHY_SPEED	invalid PHY speed setting
RT_ERR_PHY_DUPLEX	invalid PHY duplex setting
RT_ERR_INPUT	invalid input parameter

17.12.rtk_port_phyReg_get

```
int32 rtk_port_phyReg_get(rtk_port_t port, uint32 page, rtk_port_phy_reg_t
reg, uint32 *pData)
```

Get PHY register data of the specific port

Defined in: port.h

Parameters

port
 port id
page
 page id
reg
 reg id

**pData*
pointer to the PHY reg data

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_PHY_PAGE_ID	invalid page id
	RT_ERR_PHY_REG_ID	invalid reg id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.13.rtk_port_phyReg_set

`int32 rtk_port_phyReg_set(rtk_port_t port, uint32 page, rtk_port_phy_reg_t reg, uint32 data)`

Set PHY register data of the specific port

Defined in: port.h

Parameters

<i>port</i>	port id
<i>page</i>	page id
<i>reg</i>	reg id
<i>data</i>	reg data

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_PHY_PAGE_ID	invalid page id
	RT_ERR_PHY_REG_ID	invalid reg id

17.14.rtk_port_phyMasterSlave_get

```
int32 rtk_port_phyMasterSlave_get(rtk_port_t port, rtk_port_masterSlave_t
*pMasterSlave)
```

Get PHY configuration of master/slave mode of the specific port

Defined in: port.h

Parameters

<i>port</i>	port id
<i>*pMasterSlave</i>	pointer to the PHY master slave configuration

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.15.rtk_port_phyMasterSlave_set

```
int32 rtk_port_phyMasterSlave_set(rtk_port_t port, rtk_port_masterSlave_t
masterSlave)
```

Set PHY configuration of master/slave mode of the specific port

Defined in: port.h

Parameters

<i>port</i>	port id
<i>masterSlave</i>	PHY master slave configuration

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	RT_ERR_INPUT

17.16.rtk_port_phyTestMode_get

```
int32 rtk_port_phyTestMode_get(rtk_port_t port, rtk_port_phy_test_mode_t
*pTestMode)
```

Get PHY in which test mode.

Defined in: port.h

Parameters

port
Port id.

**pTestMode*

PHY test mode 0: normal 1: test mode 1 2: test mode 2 3: test mode 3 4: test mode 4 5~7: reserved

Comments

Get test mode of PHY from register setting 9.15 to 9.13.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_PORT_ID	Invalid port number.
RT_ERR_INPUT	Invalid input parameters.
RT_ERR_BUSYWAIT_TIMEOUT	PHY access busy

17.17.rtk_port_phyTestMode_set

```
int32 rtk_port_phyTestMode_set(rtk_port_t port, rtk_port_phy_test_mode_t
testMode)
```

Set PHY in test mode.

Defined in: port.h

Parameters

port
port id.

testMode

PHY test mode 0: normal 1: test mode 1 2: test mode 2 3: test mode 3 4: test mode 4 5~7: reserved

Comments

None.

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_PORT_ID	Invalid port number.
RT_ERR_BUSYWAIT_TIMEOUT	PHY access busy

17.18.rtk_port_cpuPortId_get

`int32 rtk_port_cpuPortId_get(rtk_port_t *pPort)`

Get CPU port id of the specific unit

Defined in: port.h

Parameters

`*pPort`
pointer to CPU port id

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.19.rtk_port_isolation_get

`int32 rtk_port_isolation_get(rtk_port_t port, rtk_portmask_t *pPortmask,
rtk_portmask_t *pExtPortmask)`

Get the portmask of the port isolation

Defined in: port.h

Parameters

`port`
port id
`*pPortmask`
pointer to the portmask
`*pExtPortmask`
pointer to extension portmask

Comments

(1) Default value of each port is 1
(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.20.rtk_port_isolation_set

`int32 rtk_port_isolation_set(rtk_port_t port, rtk_portmask_t *pPortmask,
rtk_portmask_t *pExtPortmask)`

Set the portmask of the port isolation

Defined in: port.h

Parameters

port
port id
**pPortmask*
pointer to the portmask
**pExtPortmask*
pointer to extension portmask

Comments

(1) Default value of each port is 1
(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_PORT_MASK	invalid port mask

17.21.rtk_port_isolationExt_get

`int32 rtk_port_isolationExt_get(rtk_port_t port, rtk_portmask_t *pPortmask,
rtk_portmask_t *pExtPortmask)`

Get the portmask of 2nd port isolation configuration

Defined in: port.h

Parameters

port
port id

	<i>*pPortmask</i> pointer to the portmask
	<i>*pExtPortmask</i> pointer to extension portmask
Comments	(1) Default value of each port is 1 (2) Enable port isolation in the certain ports if relative portmask bits are set to 1
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_NULL_POINTER
	ok failed invalid port id input parameter may be null pointer

17.22.rtk_port_isolationExt_set

`int32 rtk_port_isolationExt_set(rtk_port_t port, rtk_portmask_t *pPortmask, rtk_portmask_t *pExtPortmask)`

Set the portmask of the 2nd port isolation configuration

Defined in: port.h

Parameters

<i>port</i>	port id
<i>*pPortmask</i>	pointer to the portmask
<i>*pExtPortmask</i>	pointer to extension portmask

Comments

(1) Default value of each port is 1
(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_PORT_MASK	invalid port mask

17.23.rtk_port_isolationL34_get

```
int32 rtk_port_isolationL34_get(rtk_port_t port, rtk_portmask_t *pPortmask,
                                rtk_portmask_t *pExtPortmask)
```

Get the portmask of the port isolation

Defined in: port.h

Parameters

port
port id

**pPortmask*
pointer to the portmask

**pExtPortmask*
pointer to extension portmask

Comments

(1) Default value of each port is 1

(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.24.rtk_port_isolationL34_set

```
int32 rtk_port_isolationL34_set(rtk_port_t port, rtk_portmask_t *pPortmask,
                                rtk_portmask_t *pExtPortmask)
```

Set the portmask of the port isolation

Defined in: port.h

Parameters

port
port id

**pPortmask*
pointer to the portmask

**pExtPortmask*
pointer to extension portmask

Comments

(1) Default value of each port is 1

(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_PORT_MASK	invalid port mask

17.25.rtk_port_isolationExtL34_get

```
int32 rtk_port_isolationExtL34_get(rtk_port_t port, rtk_portmask_t
*pPortmask, rtk_portmask_t *pExtPortmask)
```

Get the portmask of 2nd port isolation configuration

Defined in: port.h

Parameters

port
port id
**pPortmask*
pointer to the portmask
**pExtPortmask*
pointer to extension portmask

Comments

(1) Default value of each port is 1
(2) Enable port isolation in the certain ports if relative portmask bits are set to 1

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.26.rtk_port_isolationExtL34_set

```
int32 rtk_port_isolationExtL34_set(rtk_port_t port, rtk_portmask_t
*pPortmask, rtk_portmask_t *pExtPortmask)
```

Set the portmask of the 2nd port isolation configuration

Defined in: port.h

Parameters

port
port id

	<i>*pPortmask</i> pointer to the portmask
	<i>*pExtPortmask</i> pointer to extension portmask
Comments	(1) Default value of each port is 1 (2) Enable port isolation in the certain ports if relative portmask bits are set to 1
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_PORT_MASK
	ok failed invalid port id invalid port mask

17.27.rtk_port_isolationEntry_get

```
int32 rtk_port_isolationEntry_get(rtk_port_isoConfig_t mode, rtk_port_t
port, rtk_portmask_t *pPortmask, rtk_portmask_t *pExtPortmask)
```

Get Port isolation portmask

Defined in: port.h

Parameters

mode
Configuration 0 or 1

port
Ingress port

**pPortmask*
Isolation portmask for specified ingress port.

**pExtPortmask*
Isolation extension portmask for specified ingress port.

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	invalid port id

17.28.rtk_port_isolationEntry_set

`int32 rtk_port_isolationEntry_set(rtk_port_isoConfig_t mode, rtk_port_t port, rtk_portmask_t *pPortmask, rtk_portmask_t *pExtPortmask)`

Set Port isolation portmask

Defined in: port.h

Parameters

mode

Configuration 0 or 1

port

Ingress port

**pPortmask*

Isolation portmask for specified ingress port.

**pExtPortmask*

Isolation extension portmask for specified ingress port.

Comments

pExtPortmask is the extension egress portmask toward CPU port. If users specify an empty extension portmask and CPU port is set in pPortmask, the packets will be restricted to be forwarded to CPU. Likewise, If users specify an non-empty extension portmask and CPU port is not set in pPortmask, the packets will be restricted to be forwarded to CPU. too.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_NULL_POINTER`

invalid port id

17.29.rtk_port_isolationEntryExt_get

`int32 rtk_port_isolationEntryExt_get(rtk_port_isoConfig_t mode, rtk_port_t port, rtk_portmask_t *pPortmask, rtk_portmask_t *pExtPortmask)`

Get Port isolation portmask

Defined in: port.h

Parameters

mode

Configuration 0 or 1

port

Ingress port

**pPortmask*
Isolation portmask for specified ingress port.
**pExtPortmask*
Isolation extension portmask for specified ingress port.

Comments None.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	invalid port id

17.30.rtk_port_isolationEntryExt_set

```
int32 rtk_port_isolationEntryExt_set(rtk_port_isoConfig_t mode, rtk_port_t port, rtk_portmask_t *pPortmask, rtk_portmask_t *pExtPortmask)
```

Set Port isolation portmask

Defined in: port.h

Parameters

<i>mode</i>	Configuration 0 or 1
<i>port</i>	Ingress port
<i>*pPortmask</i>	Isolation portmask for specified ingress port.
<i>*pExtPortmask</i>	Isolation extension portmask for specified ingress port.

Comments pExtPortmask is the extension egress portmask toward CPU port. If users specify an empty extension portmask and CPU port is set in pPortmask, the packets will be restricted to be forwarded to CPU. Likewise, If users specify an non-empty extension portmask and CPU port is not set in pPortmask, the packets will be restricted to be forwarded to CPU. too.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	invalid port id

17.31.rtk_port_isolationCtagPktConfig_get

`int32 rtk_port_isolationCtagPktConfig_get(rtk_port_isoConfig_t *pMode)`

Isolation configuration selection for ingress Ctag packets

Defined in: port.h

Parameters

**pMode*
Isolation configuration selection

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	invalid port id

17.32.rtk_port_isolationCtagPktConfig_set

`int32 rtk_port_isolationCtagPktConfig_set(rtk_port_isoConfig_t mode)`

Isolation configuration selection for ingress Ctag packets

Defined in: port.h

Parameters

mode
Isolation configuration selection

Comments

None.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

17.33.rtk_port_isolationL34PktConfig_get

`int32 rtk_port_isolationL34PktConfig_get(rtk_port_isoConfig_t *pMode)`

Isolation configuration selection for ingress Ctag packets

Defined in: port.h

Parameters	<i>*pMode</i>
	Isolation configuration selection
Comments	None.
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed
	RT_ERR_NULL_POINTER invalid port id

17.34.rtk_port_isolationL34PktConfig_set

`int32 rtk_port_isolationL34PktConfig_set(rtk_port_isoConfig_t mode)`

Isolation configuration selection for ingress Ctag packets

Defined in: port.h

Parameters	<i>mode</i>
	Isolation configuration selection
Comments	None.
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed

17.35.rtk_port_isolationIpmcLeaky_get

`int32 rtk_port_isolationIpmcLeaky_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get the ip multicast leaky state of the port isolation

Defined in: port.h

Parameters	<i>port</i>
	port id
	<i>*pEnable</i>
	status of port isolation leaky for ip multicast packets
Comments	none
Return Codes	
	RT_ERR_OK ok

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.36.rtk_port_isolationIpmcLeaky_set

`int32 rtk_port_isolationIpmcLeaky_set(rtk_port_t port, rtk_enable_t enable)`

Set the ip multicast leaky state of the port isolation

Defined in: port.h

Parameters

<i>port</i>	port id
<i>enable</i>	status of port isolation leaky for ip multicast packets

Comments

none

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.37.rtk_port_isolationPortLeaky_get

`int32 rtk_port_isolationPortLeaky_get(rtk_port_t port, rtk_leaky_type_t type, rtk_enable_t *pEnable)`

Get the per port isolation leaky state for given type

Defined in: port.h

Parameters

<i>port</i>	port id
<i>type</i>	Packet type for isolation leaky.
<i>*pEnable</i>	status of port isolation leaky for given leaky type

Comments	none
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.38.rtk_port_isolationPortLeaky_set

```
int32 rtk_port_isolationPortLeaky_set(rtk_port_t port, rtk_leaky_type_t type,
                                      rtk_enable_t enable)
```

Set the per port isolation leaky state for given type

Defined in: port.h

Parameters	
<i>port</i>	port id
<i>type</i>	Packet type for isolation leaky.
<i>enable</i>	status of port isolation leaky for given leaky type
Comments	none
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.39.rtk_port_isolationLeaky_get

```
int32 rtk_port_isolationLeaky_get(rtk_leaky_type_t type, rtk_enable_t
*pEnable)
```

Get the per port isolation leaky state for given type

Defined in: port.h

Parameters

Comments	none						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_NULL_POINTER	input parameter may be null pointer						

17.40.rtk_port_isolationLeaky_set

```
int32 rtk_port_isolationLeaky_set(rtk_leaky_type_t type, rtk_enable_t enable)
```

Set the per port isolation leaky state for given type

Defined in: port.h

Parameters	<i>type</i>	Packet type for isolation leaky.
	<i>enable</i>	status of port isolation leaky for given leaky type

Comments	none
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.41.rtk_port_macRemoteLoopbackEnable_get

```
int32 rtk_port_macRemoteLoopbackEnable_get(rtk_port_t port, rtk_enable_t *pEnable)
```

Get the mac remote loopback enable status of the specific port

Defined in: port.h

Parameters	
-------------------	--

	<i>port</i>	port id
	<i>*pEnable</i>	pointer to the enable status of mac remote loopback
Comments		
(1) The mac remote loopback enable status of the port is as following: - DISABLE - ENABLE		
(2) Remote loopback is used to loopback packet RX to switch core back to the outer interface.		
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

17.42.rtk_port_macRemoteLoopbackEnable_set

```
int32 rtk_port_macRemoteLoopbackEnable_set(rtk_port_t port,
                                          rtk_enable_t enable)
```

Set the mac remote loopback enable status of the specific port

Defined in: port.h

	<i>port</i>	port id
	<i>enable</i>	enable status of mac remote loopback
Comments		
(1) The mac remote loopback enable status of the port is as following: - DISABLE - ENABLE		
(2) Remote loopback is used to loopback packet RX to switch core back to the outer interface.		
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

17.43.rtk_port_macLocalLoopbackEnable_get

`int32 rtk_port_macLocalLoopbackEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get the mac local loopback enable status of the specific port

Defined in: port.h

Parameters

port

port id

**pEnable*

pointer to the enable status of mac local loopback

Comments

(1) The mac local loopback enable status of the port is as following:

- DISABLE
- ENABLE

(2) Local loopback is used to loopback packet TX from switch core back to switch core.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_PORT_ID

invalid port id

RT_ERR_INPUT

invalid input parameter

17.44.rtk_port_macLocalLoopbackEnable_set

`int32 rtk_port_macLocalLoopbackEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set the mac local loopback enable status of the specific port

Defined in: port.h

Parameters

port

port id

enable

enable status of mac local loopback

Comments

(1) The mac local loopback enable status of the port is as following:
 - DISABLE

- ENABLE
 (2) Local loopback is used to loopback packet TX from switch core back to switch core.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

17.45.rtk_port_adminEnable_get

`int32 rtk_port_adminEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get port admin status of the specific port

Defined in: port.h

Parameters	<i>port</i>	port id
	<i>*pEnable</i>	pointer to the port admin status

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.46.rtk_port_adminEnable_set

`int32 rtk_port_adminEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set port admin status of the specific port

Defined in: port.h

Parameters	<i>port</i>	port id
-------------------	-------------	---------

	<i>enable</i>	
	port admin status	
Comments	None	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id

17.47.rtk_port_specialCongest_get

`int32 rtk_port_specialCongest_get(rtk_port_t port, uint32 *pSecond)`

Set the congest seconds of the specific port

Defined in: port.h

Parameters	<i>port</i>	
	port id	
	<i>*pSecond</i>	congest timer (seconds)

Comments	None	
-----------------	------	--

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.48.rtk_port_specialCongest_set

`int32 rtk_port_specialCongest_set(rtk_port_t port, uint32 second)`

Set the congest seconds of the specific port

Defined in: port.h

Parameters	<i>port</i>	
	port id	

	<i>second</i>	congest timer (seconds)
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_INPUT	ok failed invalid port id invalid input parameter

17.49.rtk_port_specialCongestStatus_get

`int32 rtk_port_specialCongestStatus_get(rtk_port_t port, uint32 *pStatus)`

Get the congest status of the specific port

Defined in: port.h

Parameters	<i>port</i> port id <i>*pStatus</i> Congest status	
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_INPUT	ok failed invalid port id invalid input parameter

17.50.rtk_port_specialCongestStatus_clear

`int32 rtk_port_specialCongestStatus_clear(rtk_port_t port)`

Get the congest status of the specific port

Defined in: port.h

Parameters	<i>port</i> port id	
Comments	None	

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

17.51.rtk_port_greenEnable_get

`int32 rtk_port_greenEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get the statue of green feature of the specific port in the specific unit

Defined in: port.h

Parameters

<i>port</i>	port id
<i>*pEnable</i>	pointer to status of green feature

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.52.rtk_port_greenEnable_set

`int32 rtk_port_greenEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set the statue of green feature of the specific port in the specific unit

Defined in: port.h

Parameters

<i>port</i>	port id
<i>enable</i>	status of green feature

Comments

None

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id

17.53.rtk_port_phyCrossOverMode_get

**int32 rtk_port_phyCrossOverMode_get(rtk_port_t *port*,
 rtk_port_crossOver_mode_t **pCrossoverMode*)**

Get cross over mode in the specified port.

Defined in: port.h

Parameters

port
 port id
**pCrossoverMode*
 pointer to cross over mode

Comments

Following value is valid

- PORT_CROSSOVER_MODE_AUTO
- PORT_CROSSOVER_MODE_MDI
- PORT_CROSSOVER_MODE_MDIX

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.54.rtk_port_phyCrossOverMode_set

**int32 rtk_port_phyCrossOverMode_set(rtk_port_t *port*,
 rtk_port_crossOver_mode_t *crossoverMode*)**

Set cross over mode in the specified port.

Defined in: port.h

Parameters

port
 port id
crossoverMode
 cross over mode

Comments	Following value is valid - PORT_CROSSOVER_MODE_AUTO - PORT_CROSSOVER_MODE_MDI - PORT_CROSSOVER_MODE_MDIX	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

17.55.rtk_port_enhancedFid_get

`int32 rtk_port_enhancedFid_get(rtk_port_t port, rtk_efid_t *pEfId)`

Get port EFID

Defined in: port.h

Parameters	<i>port</i>	port id
	<i>*pEfId</i>	EFID
Comments	none	
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

17.56.rtk_port_enhancedFid_set

`int32 rtk_port_enhancedFid_set(rtk_port_t port, rtk_efid_t efid)`

Set port EFID

Defined in: port.h

Parameters	<i>port</i>	port id
-------------------	-------------	---------

	<i>efid</i>	EFID
Comments	none	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID	ok failed invalid port id

17.57.rtk_port_rtctResult_get

`int32 rtk_port_rtctResult_get(rtk_port_t port, rtk_rtctResult_t *pResult)`

Get test result of RTCT.

Defined in: port.h

Parameters	<i>port</i>	port id
	<i>*pResult</i>	Test Result

Comments
If linkType is PORT_SPEED_1000M, test result will be stored in ge_result. If linkType is PORT_SPEED_10M or PORT_SPEED_100M, test result will be stored in fe_result.

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_NULL_POINTER	ok failed invalid port id input parameter may be null pointer
---------------------	---	--

17.58.rtk_port_rtct_start

`int32 rtk_port_rtct_start(rtk_portmask_t *pPortmask)`

When enable RTCT, the port won't transmit and receive normal traffic.

Defined in: port.h

Parameters	<i>*pPortmask</i>
	the ports for RTCT test

Comments	none	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id

17.59.rtk_port_macForceAbility_set

`int32 rtk_port_macForceAbility_set(rtk_port_t port, rtk_port_macAbility_t macAbility)`

Set MAC forece ability

Defined in: port.h

Parameters

port the ports for set ability

macAbility mac ability value

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_UNIT_ID	invalid unit id
RT_ERR_NULL_POINTER	input parameter may be null pointer

17.60.rtk_port_macForceAbility_get

`int32 rtk_port_macForceAbility_get(rtk_port_t port, rtk_port_macAbility_t *pMacAbility)`

Set MAC forece ability

Defined in: port.h

Parameters

port the ports for get ability

<i>*pMacAbility</i>	mac ability value										
Comments	None										
Return Codes	<table> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_UNIT_ID</td><td>invalid unit id</td></tr> <tr> <td>RT_ERR_NULL_POINTER</td><td>input parameter may be null pointer</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_UNIT_ID	invalid unit id	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_NOT_INIT	The module is not initial										
RT_ERR_UNIT_ID	invalid unit id										
RT_ERR_NULL_POINTER	input parameter may be null pointer										

17.61.rtk_port_macForceAbilityState_set

`int32 rtk_port_macForceAbilityState_set(rtk_port_t port, rtk_enable_t state)`

Set MAC force ability state

Defined in: port.h

Parameters	<code>port</code> the ports for set ability <code>state</code> mac ability state										
Comments	None										
Return Codes	<table> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_UNIT_ID</td><td>invalid unit id</td></tr> <tr> <td>RT_ERR_NULL_POINTER</td><td>input parameter may be null pointer</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_UNIT_ID	invalid unit id	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_NOT_INIT	The module is not initial										
RT_ERR_UNIT_ID	invalid unit id										
RT_ERR_NULL_POINTER	input parameter may be null pointer										

17.62.rtk_port_macForceAbilityState_get

`int32 rtk_port_macForceAbilityState_get(rtk_port_t port, rtk_enable_t *pState)`

Get MAC force ability state

Defined in: port.h

Parameters	<i>port</i> the ports for get ability <i>*pState</i> mac ability state
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_UNIT_ID RT_ERR_NULL_POINTER

17.63.rtk_port_macExtMode_set

<code>int32 rtk_port_macExtMode_set(rtk_port_t port, rtk_port_ext_mode_t ext_mode)</code>		
Set extension MAC mode		
Defined in: port.h		
Parameters	<i>port</i> the ports number <i>ext_mode</i> the mode setting	
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT	ok failed The module is not initial

17.64.rtk_port_macExtMode_get

<code>int32 rtk_port_macExtMode_get(rtk_port_t port, rtk_port_ext_mode_t *pExt_mode)</code>		
Get extension MAC mode		
Defined in: port.h		

Parameters	<i>port</i> the ports number <i>*pExt_mode</i> the mode setting
Comments	None
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NOT_INIT The module is not initial RT_ERR_NULL_POINTER input parameter may be null pointer

17.65.rtk_port_macExtRgmiiDelay_set

int32 rtk_port_macExtRgmiiDelay_set(rtk_port_t port, uint32 txDelay, uint32 rxDelay)

Set RGMII TX/RX delay

Defined in: port.h

Parameters	<i>port</i> the ports number <i>txDelay</i> the TX delay (0 ~ 1) <i>rxDelay</i> the RX delay (0 ~ 7)
Comments	None
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NOT_INIT The module is not initial

17.66.rtk_port_macExtRgmiiDelay_get

**int32 rtk_port_macExtRgmiiDelay_get(rtk_port_t port, uint32 *pTxDelay,
uint32 *pRxDelay)**

Get RGMII TX/RX delay

	Defined in: port.h								
Parameters	<p><i>port</i> the ports number</p> <p><i>*pTxDelay</i> the TX delay (0 ~ 1)</p> <p><i>*pRxDelay</i> the RX delay (0 ~ 7)</p>								
Comments	None								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_NOT_INIT	The module is not initial								
RT_ERR_NULL_POINTER	input parameter may be null pointer								

18. Module QoS API

Filename: qos.h

Description	The file includes the following modules and sub-modules (1) Ingress Priority Decision (2) Egress Remarking (3) Queue Scheduling (4) Congestion avoidance
--------------------	--

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

18.1. List of Symbols

Here is a list of all functions and variables in this module

qos.h - Definition of QoS API
 rtk_qos_init
 rtk_qos_priSelGroup_get
 rtk_qos_priSelGroup_set
 rtk_qos_portPri_get
 rtk_qos_portPri_set
 rtk_qos_dscpPriRemapGroup_get
 rtk_qos_dscpPriRemapGroup_set
 rtk_qos_1pPriRemapGroup_get
 rtk_qos_1pPriRemapGroup_set

```

rtk_qos_priMap_set
rtk_qos_priMap_get
rtk_qos_portPriMap_get
rtk_qos_portPriMap_set
rtk_qos_1pRemarkEnable_get
rtk_qos_1pRemarkEnable_set
rtk_qos_1pRemarkGroup_get
rtk_qos_1pRemarkGroup_set
rtk_qos_dscpRemarkEnable_get
rtk_qos_dscpRemarkEnable_set
rtk_qos_dscpRemarkGroup_get
rtk_qos_dscpRemarkGroup_set
rtk_qos_portDscpRemarkSrcSel_get
rtk_qos_portDscpRemarkSrcSel_set
rtk_qos_dscp2DscpRemarkGroup_get
rtk_qos_dscp2DscpRemarkGroup_set
rtk_qos_fwd2CpuPriRemap_get
rtk_qos_fwd2CpuPriRemap_set
rtk_qos_schedulingQueue_get
rtk_qos_schedulingQueue_set
rtk_qos_portPriSelGroup_get
rtk_qos_portPriSelGroup_set

```

18.2. rtk_qos_init

`int32 rtk_qos_init(void)`

Configure QoS initial settings

Defined in: qos.h

Parameters

`void`

Comments

The initialization does the following actions:

- set input bandwidth control parameters to default values
- set priority decision parameters
- set scheduling parameters
- disable port remark ability
- CPU port init 8 using priority to queue mapping index 0
- Other port init 1 queue using priority to queue mapping index 1

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

RT_ERR_QUEUE_NUM	Invalid queue number
------------------	----------------------

18.3. rtk_qos_priSelGroup_get

```
int32 rtk_qos_priSelGroup_get(uint32 grpIdx, rtk_qos_priSelWeight_t
    *pWeightOfPriSel)
```

Get weight of each priority assignment on specified priority selection group.

Defined in: qos.h

Parameters

grpIdx
index of priority selection group

**pWeightOfPriSel*
pointer to weight of each priority assignment

Comments

Apollo only support group 0

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid input parameter
RT_ERR_NULL_POINTER	input parameter may be null pointer

18.4. rtk_qos_priSelGroup_set

```
int32 rtk_qos_priSelGroup_set(uint32 grpIdx, rtk_qos_priSelWeight_t
    *pWeightOfPriSel)
```

Set weight of each priority assignment on specified priority selection group.

Defined in: qos.h

Parameters

grpIdx
index of priority selection group

**pWeightOfPriSel*
weight of each priority assignment

Comments

Apollo only support group 0

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	invalid input parameter
RT_ERR_NULL_POINTER	input parameter may be null pointer

18.5. rtk_qos_portPri_get

`int32 rtk_qos_portPri_get(rt_k_port_t port, rt_k_pri_t *pIntPri)`

Get internal priority of one port.

Defined in: qos.h

Parameters

port
port id
**pIntPri*

Priorities assigment for specific port. (range from 0 ~ 7, 7 is

Comments

None

Return Codes

RT_ERR_OK
RT_ERR_FAILED
RT_ERR_PORT_ID
RT_ERR_NULL_POINTER

ok
failed
Invalid port id
NULL pointer

18.6. rtk_qos_portPri_set

`int32 rtk_qos_portPri_set(rt_k_port_t port, rt_k_pri_t intPri)`

Get internal priority of one port.

Defined in: qos.h

Parameters

port
port id
intPri

Priorities assigment for specific port. (range from 0 ~ 7, 7 is

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id
RT_ERR_NULL_POINTER	NULL pointer

18.7. rtk_qos_dscpPriRemapGroup_get

`int32 rtk_qos_dscpPriRemapGroup_get(uint32 grpIdx, uint32 dscp, rtk_pri_t *pIntPri, uint32 *pDp)`

Get remapped internal priority of DSCP on specified DSCP remapping group.

Defined in: qos.h

Parameters

grpIdx
index of dscp remapping group

dscp
DSCP

**pIntPri*
pointer to internal priority

**pDp*
pointer to drop precedence

Comments

Apollo only support group 0

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_QOS_DSCP_VALUE	invalid DSCP value
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	invalid input parameter

18.8. rtk_qos_dscpPriRemapGroup_set

`int32 rtk_qos_dscpPriRemapGroup_set(uint32 grpIdx, uint32 dscp, rtk_pri_t intPri, uint32 dp)`

Set remapped internal priority of DSCP on specified DSCP remapping group.

Defined in: qos.h

Parameters	<i>grpIdx</i> index of dscp remapping group														
	<i>dscp</i> DSCP														
	<i>intPri</i> internal priority														
	<i>dp</i> drop precedence														
Comments	Apollo only support group 0														
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_QOS_DSCP_VALUE</td><td>invalid DSCP value</td></tr> <tr> <td>RT_ERR_QOS_INT_PRIORITY</td><td>invalid internal priority</td></tr> <tr> <td>RT_ERR_DROP_PRECEDENCE</td><td>invalid drop precedence</td></tr> <tr> <td>RT_ERR_INPUT</td><td>invalid input parameter</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_QOS_DSCP_VALUE	invalid DSCP value	RT_ERR_QOS_INT_PRIORITY	invalid internal priority	RT_ERR_DROP_PRECEDENCE	invalid drop precedence	RT_ERR_INPUT	invalid input parameter
RT_ERR_OK	ok														
RT_ERR_FAILED	failed														
RT_ERR_NOT_INIT	The module is not initial														
RT_ERR_QOS_DSCP_VALUE	invalid DSCP value														
RT_ERR_QOS_INT_PRIORITY	invalid internal priority														
RT_ERR_DROP_PRECEDENCE	invalid drop precedence														
RT_ERR_INPUT	invalid input parameter														

18.9. rtk_qos_1pPriRemapGroup_get

```
int32 rtk_qos_1pPriRemapGroup_get(uint32 grpIdx, rtk_pri_t dot1pPri,
rtk_pri_t *pIntPri, uint32 *pDp)
```

Get remapped internal priority of dot1p priority on specified dot1p priority remapping group.

Defined in: qos.h

Parameters	<i>grpIdx</i> index of outer dot1p remapping group				
	<i>dot1pPri</i> dot1p priority				
	<i>*pIntPri</i> pointer to internal priority				
	<i>*pDp</i> pointer to drop precedence				
Comments	Apollo only support group 0				
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_QOS_1P_PRIORITY	invalid dot1p priority
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	invalid input parameter

18.10.rtk_qos_1pPriRemapGroup_set

`int32 rtk_qos_1pPriRemapGroup_set(uint32 grpIdx, rtk_pri_t dot1pPri, rtk_pri_t intPri, uint32 dp)`

Set remapped internal priority of dot1p priority on specified dot1p priority remapping group.

Defined in: qos.h

Parameters

grpIdx
index of dot1p remapping group
dot1pPri
dot1p priority
intPri
internal priority
dp
drop precedence

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_QOS_1P_PRIORITY	invalid dot1p priority
RT_ERR_QOS_INT_PRIORITY	invalid internal priority
RT_ERR_DROP_PRECEDENCE	invalid drop precedence
RT_ERR_INPUT	invalid input parameter

18.11.rtk_qos_priMap_set

`int32 rtk_qos_priMap_set(uint32 group, rtk_qos_pri2queue_t *pPri2qid)`

Set the entry of internal priority to QID mapping table.

Defined in: qos.h

Parameters

group
the group of priority to Queue id map(0~3).

**pPri2qid*
array of internal priority on a queue

Comments

Below is an example of internal priority to QID mapping table.

group	Priority							
	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0	0
2	0	0	0	0	6	6	6	6
3	0	0	0	1	1	2	2	3 -for table index 0
pPri2qid[0] = 0	internal priority 0 map to queue 0							
pPri2qid[1] = 1	internal priority 1 map to queue 1							
pPri2qid[2] = 2	internal priority 2 map to queue 2							
pPri2qid[3] = 3	internal priority 3 map to queue 3							
pPri2qid[4] = 4	internal priority 4 map to queue 4							
pPri2qid[5] = 5	internal priority 5 map to queue 5							
pPri2qid[6] = 6	internal priority 6 map to queue 6							
pPri2qid[7] = 7	internal priority 7 map to queue 7							

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_ENTRY_INDEX	Invalid group index
RT_ERR_NULL_POINTER	input parameter may be null pointer

18.12.rtk_qos_priMap_get

`int32 rtk_qos_priMap_get(uint32 group, rtk_qos_pri2queue_t *pPri2qid)`

Get the entry of internal priority to QID mapping table.

Defined in: qos.h

Parameters

group
the group of priority to Queue id map(0~3).

	<i>*pPri2qid</i>	array of internal priority on a queue
Comments	None	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_ENTRY_INDEX	Invalid group index
	RT_ERR_NULL_POINTER	input parameter may be null pointer

18.13.rtk_qos_portPriMap_get

`int32 rtk_qos_portPriMap_get(rtk_port_t port, uint32 *pGroup)`

Get the value of internal priority to QID mapping table on specified port.

Defined in: qos.h

Parameters	<i>port</i>	port id
	<i>*pGroup</i>	Priority to queue mapping group
Comments	None	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

18.14.rtk_qos_portPriMap_set

`int32 rtk_qos_portPriMap_set(rtk_port_t port, uint32 group)`

Set the value of internal priority to QID mapping table on specified port.

Defined in: qos.h

Parameters	<i>port</i>	port id
-------------------	-------------	---------

	<i>group</i>	index to priority to queue table
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID	ok failed The module is not initial invalid port id

18.15.rtk_qos_1pRemarkEnable_get

`int32 rtk_qos_1pRemarkEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get 802.1p remark status for a port

Defined in: qos.h

Parameters	<i>port</i>	port id
	<i>*pEnable</i>	status of 802.1p remark

Comments	The status of 802.1p remark: - DISABLED - ENABLED
-----------------	---

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_NULL_POINTER	ok failed Invalid port id NULL pointer
---------------------	---	---

18.16.rtk_qos_1pRemarkEnable_set

`int32 rtk_qos_1pRemarkEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set 802.1p remark status for a port

Defined in: qos.h

Parameters	
-------------------	--

	<i>port</i>	port id.
	<i>enable</i>	status of 802.1p remark
Comments	The status of 802.1p remark:	
	- DISABLED	
	- ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id

18.17.rtk_qos_1pRemarkGroup_get

```
int32 rtk_qos_1pRemarkGroup_get(uint32 grpIdx, rtk_pri_t intPri, uint32 dp,
                                rtk_pri_t *pDot1pPri)
```

Get remarked dot1p priority of internal priority on specified dot1p remark group.

Defined in: qos.h

Parameters	<i>grpIdx</i>	index of dot1p remark group
	<i>intPri</i>	internal priority
	<i>dp</i>	drop precedence
	<i>*pDot1pPri</i>	pointer to dot1p priority

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_QOS_INT_PRIORITY	invalid internal priority
	RT_ERR_DROP_PRECEDENCE	invalid drop precedence
	RT_ERR_INPUT	invalid input parameter
	RT_ERR_NULL_POINTER	input parameter may be null pointer

18.18.rtk_qos_1pRemarkGroup_set

```
int32 rtk_qos_1pRemarkGroup_set(uint32 grpIdx, rtk_pri_t intPri, uint32 dp,
                                rtk_pri_t dot1pPri)
```

Set remarked dot1p priority of internal priority on specified dot1p remark group.

Defined in: qos.h

Parameters

<i>grpIdx</i>	index of dot1p remark group
<i>intPri</i>	internal priority
<i>dp</i>	drop precedence
<i>dot1pPri</i>	dot1p priority

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_QOS_1P_PRIORITY	invalid dot1p priority
RT_ERR_QOS_INT_PRIORITY	invalid internal priority
RT_ERR_DROP_PRECEDENCE	invalid drop precedence
RT_ERR_INPUT	invalid input parameter

18.19.rtk_qos_dscpRemarkEnable_get

```
int32 rtk_qos_dscpRemarkEnable_get(rtk_port_t port, rtk_enable_t
                                    *pEnable)
```

Get DSCP remark status for a port

Defined in: qos.h

Parameters

<i>port</i>	port id
<i>*pEnable</i>	status of DSCP remark

Comments	The status of DSCP remark: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_NULL_POINTER	NULL pointer

18.20.rtk_qos_dscpRemarkEnable_set

`int32 rtk_qos_dscpRemarkEnable_set(rt_k_port_t port, rt_k_enable_t enable)`

Set DSCP remark status for a port

Defined in: qos.h

Parameters	<i>port</i>	port id
	<i>enable</i>	status of DSCP remark
Comments	The status of DSCP remark: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id

18.21.rtk_qos_dscpRemarkGroup_get

`int32 rtk_qos_dscpRemarkGroup_get(uint32 grpIdx, rt_k_pri_t intPri, uint32 dp, uint32 *pDscp)`

Get remarked DSCP of internal priority on specified dscp remark group.

Defined in: qos.h

Parameters	<i>grpIdx</i>	index of dot1p remapping group
-------------------	---------------	--------------------------------

<i>intPri</i>	internal priority														
<i>dp</i>	drop precedence														
<i>*pDscp</i>	pointer to DSCP														
Comments	(1) The valid range of grp_idx is 0 for apollo														
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_QOS_INT_PRIORITY</td><td>invalid internal priority</td></tr> <tr> <td>RT_ERR_DROP_PRECEDENCE</td><td>invalid drop precedence</td></tr> <tr> <td>RT_ERR_INPUT</td><td>invalid input parameter</td></tr> <tr> <td>RT_ERR_NULL_POINTER</td><td>input parameter may be null pointer</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_QOS_INT_PRIORITY	invalid internal priority	RT_ERR_DROP_PRECEDENCE	invalid drop precedence	RT_ERR_INPUT	invalid input parameter	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok														
RT_ERR_FAILED	failed														
RT_ERR_NOT_INIT	The module is not initial														
RT_ERR_QOS_INT_PRIORITY	invalid internal priority														
RT_ERR_DROP_PRECEDENCE	invalid drop precedence														
RT_ERR_INPUT	invalid input parameter														
RT_ERR_NULL_POINTER	input parameter may be null pointer														

18.22.rtk_qos_dscpRemarkGroup_set

```
int32 rtk_qos_dscpRemarkGroup_set(uint32 grpIdx, rtk_pri_t intPri, uint32
dp, uint32 dscp)
```

Set remarked DSCP of internal priority on specified dscp remark group.

Defined in: qos.h

Parameters	<i>grpIdx</i> index of dot1p remarking group								
	<i>intPri</i> internal priority value (range from 0 ~ 7)								
	<i>dp</i> drop precedence								
	<i>dscp</i> DSCP								
Comments	(1) The valid range of grp_idx is 0 for apollo (2) dp(drop precedence) is not implement in Apollo								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_QOS_DSCP_VALUE</td><td>invalid DSCP value</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_QOS_DSCP_VALUE	invalid DSCP value
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_NOT_INIT	The module is not initial								
RT_ERR_QOS_DSCP_VALUE	invalid DSCP value								

RT_ERR_QOS_INT_PRIORITY	invalid internal priority
RT_ERR_DROP_PRECEDENCE	invalid drop precedence
RT_ERR_INPUT	invalid input parameter

18.23.rtk_qos_portDscpRemarkSrcSel_get

```
int32 rtk_qos_portDscpRemarkSrcSel_get(rtk_port_t port,
                                         rtk_qos_dscpRmkSrc_t *pType)
```

Get remarking source of DSCP remarking.

Defined in: qos.h

Parameters

<i>port</i>	port id
<i>*pType</i>	remarking source

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	invalid input parameter
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	

18.24.rtk_qos_portDscpRemarkSrcSel_set

```
int32 rtk_qos_portDscpRemarkSrcSel_set(rtk_port_t port,
                                         rtk_qos_dscpRmkSrc_t type)
```

Set remarking source of DSCP remarking.

Defined in: qos.h

Parameters

<i>port</i>	port id
-------------	---------

	<i>type</i>	remarking source
Comments	The API can configure DSCP remark functionality to map original DSCP value or internal priority to TX DSCP value.	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_INPUT RT_ERR_NULL_POINTER	ok failed The module is not initial invalid port id invalid input parameter

18.25.rtk_qos_dscp2DscpRemarkGroup_get

```
int32 rtk_qos_dscp2DscpRemarkGroup_get(uint32 grpIdx, uint32 dscp,
                                         uint32 *pDscp)
```

Get DSCP to remarked DSCP mapping.

Defined in: qos.h

Parameters	<i>grpIdx</i> DSCP value
	<i>dscp</i> group index
	<i>*pDscp</i> remarked DSCP value

Comments None.

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_QOS_DSCP_VALUE RT_ERR_NULL_POINTER	ok failed Invalid dscp value NULL pointer
---------------------	--	--

18.26.rtk_qos_dscp2DscpRemarkGroup_set

```
int32 rtk_qos_dscp2DscpRemarkGroup_set(uint32 grpIdx, uint32 dscp,
                                         uint32 rmkDscp)
```

Set DSCP to remarked DSCP mapping.

Defined in: qos.h

Parameters

grpIdx
group index

dscp
DSCP value

rmkDscp
remarked DSCP value

Comments

dscp parameter can be DSCP value or internal priority according to configuration of API `dal_apollo_qos_dscpRemarkSrcSel_set()`, because DSCP remark functionality can map original DSCP value or internal priority to TX DSCP value.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_QOS_DSCP_VALUE</code>	Invalid dscp value

18.27.rtk_qos_fwd2CpuPriRemap_get

```
int32 rtk_qos_fwd2CpuPriRemap_get(rtk_pri_t intPri, rtk_pri_t *pRempPri)
```

Get forward to CPU port remapped priority for internal priority.

Defined in: qos.h

Parameters

intPri
internal priority

**pRempPri*
pointer to remapping priority

Comments

dscp parameter can be DSCP value or internal priority according to configuration of API `dal_apollo_qos_dscpRemarkSrcSel_set()`, because DSCP remark functionality can map original DSCP value or internal priority to TX DSCP value.

Return Codes

<code>RT_ERR_OK</code>	ok
------------------------	----

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_QOS_INT_PRIORITY	invalid internal priority
RT_ERR_INPUT	invalid input parameter

18.28.rtk_qos_fwd2CpuPriRemap_set

`int32 rtk_qos_fwd2CpuPriRemap_set(rtk_pri_t intPri, rtk_pri_t rempPri)`

Set remapped internal priority of DSCP on specified DSCP remapping group.

Defined in: qos.h

Parameters

intPri
internal priority

rmpPri
remapping priority

Comments

dscp parameter can be DSCP value or internal priority according to configuration of API `dal_apollo_qos_dscpRemarkSrcSel_set()`, because DSCP remark functionality can map original DSCP value or internal priority to TX DSCP value.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_QOS_INT_PRIORITY	invalid internal priority
RT_ERR_INPUT	invalid input parameter

18.29.rtk_qos_schedulingQueue_get

`int32 rtk_qos_schedulingQueue_get(rtk_port_t port,
rtk_qos_queue_weights_t *pQweights)`

Get the scheduling types and weights of queues on specific port in egress scheduling.

Defined in: qos.h

Parameters

	<i>port</i>	port id
	<i>*pQweights</i>	the array of weights for WRR/WFQ queue (valid:1~128, 0 for STRICT_PRIORITY queue)
Comments	The types of queue are: WFQ_WRR_PRIORITY or STRICT_PRIORITY. If the weight is 0 then the type is STRICT_PRIORITY, else the type is WFQ_WRR_PRIORITY.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_NULL_POINTER	NULL pointer

18.30.rtk_qos_schedulingQueue_set

```
int32 rtk_qos_schedulingQueue_set(rtk_port_t port,
                                  rtk_qos_queue_weights_t *pQweights)
```

Set the scheduling types and weights of queues on specific port in egress scheduling.

Defined in: qos.h

	<i>port</i>	port id
	<i>*pQweights</i>	the array of weights for WRR/WFQ queue (valid:1~128, 0 for STRICT_PRIORITY queue)

Comments The types of queue are: WFQ_WRR_PRIORITY or STRICT_PRIORITY. If the weight is 0 then the type is STRICT_PRIORITY, else the type is WFQ_WRR_PRIORITY.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_QOS_QUEUE_WEIGHT	Invalid queue weight

18.31.rtk_qos_portPriSelGroup_get

`int32 rtk_qos_portPriSelGroup_get(rt_k_port_t port, uint32 *pPriSelGrpIdx)`

Get priority selection group for specified port.

Defined in: qos.h

Parameters

port
port id

**pPriSelGrpIdx*
pointer to index of priority selection group

Comments

The types of queue are: WFQ_WRR_PRIORITY or STRICT_PRIORITY. If the weight is 0 then the type is STRICT_PRIORITY, else the type is WFQ_WRR_PRIORITY.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_PORT_ID</code>	invalid port id
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

18.32.rtk_qos_portPriSelGroup_set

`int32 rtk_qos_portPriSelGroup_set(rt_k_port_t port, uint32 priSelGrpIdx)`

Set priority selection group for specified port.

Defined in: qos.h

Parameters

port
port id

priSelGrpIdx
index of priority selection group

Comments

The types of queue are: WFQ_WRR_PRIORITY or STRICT_PRIORITY. If the weight is 0 then the type is STRICT_PRIORITY, else the type is WFQ_WRR_PRIORITY.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	invalid input parameter

19. Module Rate API

Filename: rate.h

Description

The file includes the following modules and sub-modules

- (1) Configuration of Ingress Port Bandwidth Control [Ingress Rate Limit]
- (2) Configuration of Egress Port Bandwidth Control [Egress Rate Limit]
- (3) Configuration of Storm Control
- (3) Configuration of meter

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

19.1. List of Symbols

Here is a list of all functions and variables in this module

rate.h - Definition of Port Bandwidth Control and Storm Control API
 rtk_rate_init

rtk_rate_portIgrBandwidthCtrlRate_get
 rtk_rate_portIgrBandwidthCtrlRate_set
 rtk_rate_portIgrBandwidthCtrlIncludeIfg_get
 rtk_rate_portIgrBandwidthCtrlIncludeIfg_set
 rtk_rate_portEgrBandwidthCtrlRate_get
 rtk_rate_portEgrBandwidthCtrlRate_set
 rtk_rate_egrBandwidthCtrlIncludeIfg_get
 rtk_rate_egrBandwidthCtrlIncludeIfg_set
 rtk_rate_portEgrBandwidthCtrlIncludeIfg_get
 rtk_rate_portEgrBandwidthCtrlIncludeIfg_set
 rtk_rate_egrQueueBwCtrlEnable_get
 rtk_rate_egrQueueBwCtrlEnable_set
 rtk_rate_egrQueueBwCtrlMeterIdx_get
 rtk_rate_egrQueueBwCtrlMeterIdx_set
 rtk_rate_stormControlMeterIdx_get
 rtk_rate_stormControlMeterIdx_set
 rtk_rate_stormControlPortEnable_get
 rtk_rate_stormControlPortEnable_set
 rtk_rate_stormControlEnable_get
 rtk_rate_stormControlEnable_set

```

rtk_rate_stormBypass_set
rtk_rate_stormBypass_get
rtk_rate_shareMeter_set
rtk_rate_shareMeter_get
rtk_rate_shareMeterBucket_set
rtk_rate_shareMeterBucket_get
rtk_rate_shareMeterExceed_get
rtk_rate_shareMeterExceed_clear

```

19.2. rtk_rate_init

int32 rtk_rate_init(void)

Initial the rate module.

Defined in: rate.h

Parameters

void

Comments

None.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

19.3. rtk_rate_portIgrBandwidthCtrlRate_get

int32 rtk_rate_portIgrBandwidthCtrlRate_get(rtк_port_t *port*, uint32 **pRate*)

Get the ingress bandwidth control rate.

Defined in: rate.h

Parameters

port
port id

**pRate*
ingress bandwidth control rate

Comments

(1) The actual rate is "rate * chip granularity".
(2) The unit of granularity in apollo is 8Kbps.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_NULL_POINTER	NULL pointer

19.4. rtk_rate_portIgrBandwidthCtrlRate_set

`int32 rtk_rate_portIgrBandwidthCtrlRate_set(rtk_port_t port, uint32 rate)`

Set the ingress bandwidth control rate.

Defined in: rate.h

Parameters

port
port id

rate
ingress bandwidth control rate

Comments

- (1) The actual rate is "rate * chip granularity".
- (2) The unit of granularity in apollo is 8Kbps.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id
RT_ERR_RATE	Invalid input rate

19.5. rtk_rate_portIgrBandwidthCtrlIncludeIfg_get

`int32 rtk_rate_portIgrBandwidthCtrlIncludeIfg_get(rtk_port_t port, rtk_enable_t *pIfgInclude)`

Get the status of ingress bandwidth control includes IFG or not.

Defined in: rate.h

Parameters

port
port id
**pIfgInclude*
include IFG or not

Comments

(1) Ingress bandwidth control includes/excludes the Preamble & IFG (20 Bytes).

(2) The status of ifg_include:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_UNIT_ID	Invalid unit id
	RT_ERR_INPUT	Invalid input parameter

19.6. rtk_rate_portIgrBandwidthCtrlIncludeIfg_set

```
int32 rtk_rate_portIgrBandwidthCtrlIncludeIfg_set(rtk_port_t port,
rtk_enable_t ifgInclude)
```

Set the status of ingress bandwidth control includes IFG or not.

Defined in: rate.h

Parameters	<i>port</i> port id	
	<i>ifgInclude</i> include IFG or not	
Comments	(1) Ingress bandwidth control includes/excludes the Preamble & IFG (20 Bytes).	
	(2) The status of ifgInclude:	
	- DISABLED	
	- ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameter

19.7. rtk_rate_portEgrBandwidthCtrlRate_get

```
int32 rtk_rate_portEgrBandwidthCtrlRate_get(rtk_port_t port, uint32
*pRate)
```

Get the egress bandwidth control rate.

	Defined in: rate.h								
Parameters	<p><i>port</i> port id</p> <p><i>*pRate</i> egress bandwidth control rate</p>								
Comments	(1) The actual rate is "rate * chip granularity". (2) The unit of granularity in Apollo is 8Kbps.								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>Invalid port id</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>NULL pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	Invalid port id	RT_ERR_NULL_POINTER	NULL pointer
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_PORT_ID	Invalid port id								
RT_ERR_NULL_POINTER	NULL pointer								

19.8. rtk_rate_portEgrBandwidthCtrlRate_set

```
int32 rtk_rate_portEgrBandwidthCtrlRate_set(rtk_port_t port, uint32 rate)
```

Set the egress bandwidth control rate.

	Defined in: rate.h								
Parameters	<p><i>port</i> port id</p> <p><i>rate</i> egress bandwidth control rate</p>								
Comments	(1) The actual rate is "rate * chip granularity". (2) The unit of granularity in Apollo is 8Kbps.								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>Invalid port id</td> </tr> <tr> <td>RT_ERR_RATE</td> <td>Invalid input rate</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	Invalid port id	RT_ERR_RATE	Invalid input rate
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_PORT_ID	Invalid port id								
RT_ERR_RATE	Invalid input rate								

19.9. rtk_rate_egrBandwidthCtrlIncludeIfg_get

```
int32 rtk_rate_egrBandwidthCtrlIncludeIfg_get(rtk_enable_t *pIfgInclude)
```

Get the status of egress bandwidth control includes IFG or not.

	Defined in: rate.h	
Parameters	* <i>pIfgInclude</i> include IFG or not	
Comments	(1) Egress bandwidth control includes/excludes the Preamble & IFG (20 Bytes). (2) The status of ifg_include: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT	ok failed Invalid input parameter

19.10.rtk_rate_egxBandwidthCtrlIncludeIfg_set

`int32 rtk_rate_egxBandwidthCtrlIncludeIfg_set(rtk_enable_t ifgInclude)`

Set the status of egress bandwidth control includes IFG or not.

Defined in: rate.h

Parameters	<i>ifgInclude</i> include IFG or not	
Comments	(1) Egress bandwidth control includes/excludes the Preamble & IFG (20 Bytes). (2) The status of ifg_include: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT	ok failed Invalid input parameter

19.11.rtk_rate_portEgrBandwidthCtrlIncludeIfg_get

`int32 rtk_rate_portEgrBandwidthCtrlIncludeIfg_get(rtk_port_t port,
 rtk_enable_t *pIfgInclude)`

Per port get the status of egress bandwidth control includes IFG or not.

Defined in: rate.h

Parameters	<i>port</i> include IFG or not <i>*pIfgInclude</i> egress bandwidth control rate
Comments	(1) Egress bandwidth control includes/excludes the Preamble & IFG (20 Bytes). (2) The status of ifg_include: - DISABLED - ENABLED
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_PORT_ID invalid port id RT_ERR_INPUT Invalid input parameter

19.12.rtk_rate_portEgrBandwidthCtrlIncludeIfg_set

```
int32 rtk_rate_portEgrBandwidthCtrlIncludeIfg_set(rtk_port_t port,
rtk_enable_t ifgInclude)
```

Per port set the status of egress bandwidth control includes IFG or not.

Defined in: rate.h

Parameters	<i>port</i> port id <i>ifgInclude</i> include IFG or not
Comments	(1) Egress bandwidth control includes/excludes the Preamble & IFG (20 Bytes). (2) The status of ifg_include: - DISABLED - ENABLED
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_PORT_ID invalid port id RT_ERR_INPUT Invalid input parameter

19.13.rtk_rate_egressQueueBwCtrlEnable_get

```
int32 rtk_rate_egressQueueBwCtrlEnable_get(rtk_port_t port, rtk_qid_t queue,
                                           rtk_enable_t *pEnable)
```

Get enable status of egress bandwidth control on specified queue.

Defined in: rate.h

Parameters

port
port id

queue
queue id

**pEnable*
Pointer to enable status of egress queue bandwidth control

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_QUEUE_ID	invalid queue id
RT_ERR_NULL_POINTER	input parameter may be null pointer

19.14.rtk_rate_egressQueueBwCtrlEnable_set

```
int32 rtk_rate_egressQueueBwCtrlEnable_set(rtk_port_t port, rtk_qid_t queue,
                                            rtk_enable_t enable)
```

Set enable status of egress bandwidth control on specified queue.

Defined in: rate.h

Parameters

port
port id

queue
queue id

enable
enable status of egress queue bandwidth control

Comments	None												
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>invalid port id</td> </tr> <tr> <td>RT_ERR_QUEUE_ID</td> <td>invalid queue id</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>invalid input parameter</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_PORT_ID	invalid port id	RT_ERR_QUEUE_ID	invalid queue id	RT_ERR_INPUT	invalid input parameter
RT_ERR_OK	ok												
RT_ERR_FAILED	failed												
RT_ERR_NOT_INIT	The module is not initial												
RT_ERR_PORT_ID	invalid port id												
RT_ERR_QUEUE_ID	invalid queue id												
RT_ERR_INPUT	invalid input parameter												

19.15.rtk_rate_egrQueueBwCtrlMeterIdx_get

```
int32 rtk_rate_egrQueueBwCtrlMeterIdx_get(rtk_port_t port, rtk_qid_t queue, uint32 *pMeterIndex)
```

Get rate of egress bandwidth control on specified queue.

Defined in: rate.h

Parameters	<i>port</i> port id <i>queue</i> queue id <i>*pMeterIndex</i> meter index
-------------------	--

Comments	The actual rate is "rate * chip granularity". The unit of granularity in Apollo is 8Kbps.
-----------------	---

Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_PORT_ID</td><td>invalid port id</td></tr> <tr> <td>RT_ERR_QUEUE_ID</td><td>invalid queue id</td></tr> <tr> <td>RT_ERR_NULL_POINTER</td><td>input parameter may be null pointer</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_PORT_ID	invalid port id	RT_ERR_QUEUE_ID	invalid queue id	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok												
RT_ERR_FAILED	failed												
RT_ERR_NOT_INIT	The module is not initial												
RT_ERR_PORT_ID	invalid port id												
RT_ERR_QUEUE_ID	invalid queue id												
RT_ERR_NULL_POINTER	input parameter may be null pointer												

19.16.rtk_rate_egressQueueBwCtrlMeterIdx_set

int32 rtk_rate_egressQueueBwCtrlMeterIdx_set(rtk_port_t port, rtk_qid_t queue, uint32 meterIndex)

Set rate of egress bandwidth control on specified queue.

Defined in: rate.h

Parameters

port
port id

queue
queue id

meterIndex
meter index

Comments

The actual rate is "rate * chip granularity". The unit of granularity in Apollo is 8Kbps.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initialized
RT_ERR_PORT_ID	invalid port id
RT_ERR_QUEUE_ID	invalid queue id
RT_ERR_FILTER_METER_ID	Invalid meter

19.17.rtk_rate_stormControlMeterIdx_get

int32 rtk_rate_stormControlMeterIdx_get(rtk_port_t port, rtk_rate_storm_group_t stormType, uint32 *pIndex)

Get the storm control meter index.

Defined in: rate.h

Parameters

port
port id

stormType
storm group type

**pIndex*
storm control meter index.

Comments	The storm group types are as following: - STORM_GROUP_UNKNOWN_UNICAST - STORM_GROUP_UNKNOWN_MULTICAST - STORM_GROUP_MULTICAST - STORM_GROUP_BROADCAST - STORM_GROUP_DHCP - STORM_GROUP_ARP - STORM_GROUP_IGMP_MLD - Before call this API must make sure the global strom gruop for given group is enabled, otherwise this API will return RT_ERR_ENTRY_NOTFOUND											
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>Invalid port id</td> </tr> <tr> <td>RT_ERR_ENTRY_NOTFOUND</td> <td>The global strom group is not enable for this group</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>NULL pointer</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	Invalid port id	RT_ERR_ENTRY_NOTFOUND	The global strom group is not enable for this group	RT_ERR_NULL_POINTER	NULL pointer
RT_ERR_OK	ok											
RT_ERR_FAILED	failed											
RT_ERR_PORT_ID	Invalid port id											
RT_ERR_ENTRY_NOTFOUND	The global strom group is not enable for this group											
RT_ERR_NULL_POINTER	NULL pointer											

19.18.rtk_rate_stormControlMeterIdx_set

```
int32 rtk_rate_stormControlMeterIdx_set(rt_k_port_t port,
                                         rt_k_rate_storm_group_t stormType, uint32 index)
```

Set the storm control meter index.

Defined in: rate.h

Parameters	<i>port</i> port id <i>stormType</i> storm group type <i>index</i> storm control meter index.
Comments	The storm group types are as following: - STORM_GROUP_UNKNOWN_UNICAST - STORM_GROUP_UNKNOWN_MULTICAST - STORM_GROUP_MULTICAST - STORM_GROUP_BROADCAST - STORM_GROUP_DHCP - STORM_GROUP_ARP - STORM_GROUP_IGMP_MLD - Before call this API must make sure the global strom gruop for given group is enabled, otherwise this API will return RT_ERR_ENTRY_NOTFOUND

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id
	RT_ERR_ENTRY_NOTFOUND	The global strom group is not enable for this group
	RT_ERR_FILTER_METER_ID	Invalid meter
	RT_ERR_RATE	Invalid input bandwidth

19.19.rtk_rate_stormControlPortEnable_get

```
int32 rtk_rate_stormControlPortEnable_get(rt_k_port_t port,
                                         rtk_rate_storm_group_t stormType, rtk_enable_t *pEnable)
```

Get enable status of storm control on specified port.

Defined in: rate.h

Parameters

port
port id
stormType
storm group type
**pEnable*
pointer to enable status of storm control

Comments

The storm group types are as following:
 - STORM_GROUP_UNKNOWN_UNICAST
 - STORM_GROUP_UNKNOWN_MULTICAST
 - STORM_GROUP_MULTICAST
 - STORM_GROUP_BROADCAST
 - STORM_GROUP_DHCP
 - STORM_GROUP_ARP
 - STORM_GROUP_IGMP_MLD
 - When global strom group for given strom type is disabled, API will return DISABLED

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_SFC_UNKNOWN_GROUP	Unknown storm group
RT_ERR_NULL_POINTER	input parameter may be null pointer

19.20.rtk_rate_stormControlPortEnable_set

```
int32 rtk_rate_stormControlPortEnable_set(rtk_port_t port,
                                         rtk_rate_storm_group_t stormType, rtk_enable_t enable)
```

Set enable status of storm control on specified port.

Defined in: rate.h

Parameters

port

port id

stormType

storm group type

enable

enable status of storm control

Comments

The storm group types are as following:

- STORM_GROUP_UNKNOWN_UNICAST
- STORM_GROUP_UNKNOWN_MULTICAST
- STORM_GROUP_MULTICAST
- STORM_GROUP_BROADCAST
- STORM_GROUP_DHCP
- STORM_GROUP_ARP
- STORM_GROUP_IGMP_MLD

- Before call this API must make sure the global strom gruop for given group is enabled, otherwise this API will return RT_ERR_ENTRY_NOTFOUND

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_SFC_UNKNOWN_GROUP	Unknown storm group
RT_ERR_ENTRY_NOTFOUND	The global strom group is not enable for this group
RT_ERR_INPUT	invalid input parameter

19.21.rtk_rate_stormControlEnable_get

```
int32 rtk_rate_stormControlEnable_get(rtk_rate_storm_group_ctrl_t
                                       *stormCtrl)
```

Get enable status of storm control on specified port.

	Defined in: rate.h
Parameters	* <i>stormCtrl</i> storm group type enable control
Comments	The storm group types are as following: - STORM_GROUP_UNKNOWN_UNICAST - STORM_GROUP_UNKNOWN_MULTICAST - STORM_GROUP_MULTICAST - STORM_GROUP_BROADCAST - STORM_GROUP_DHCP - STORM_GROUP_ARP - STORM_GROUP_IGMP_MLD
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_NOT_INIT The module is not initial RT_ERR_PORT_ID invalid port id RT_ERR_NULL_POINTER input parameter may be null pointer

19.22.rtk_rate_stormControlEnable_set

```
int32 rtk_rate_stormControlEnable_set(rtk_rate_storm_group_ctrl_t
*stormCtrl)
```

Set enable status of storm control on specified port.

Defined in: rate.h

Parameters	* <i>stormCtrl</i> storm group type enable control
Comments	The storm group types are as following: - STORM_GROUP_UNKNOWN_UNICAST - STORM_GROUP_UNKNOWN_MULTICAST - STORM_GROUP_MULTICAST - STORM_GROUP_BROADCAST - STORM_GROUP_DHCP - STORM_GROUP_ARP - STORM_GROUP_IGMP_MLD total 4 storm type can be enabled. - if total enable group exceed 4 system will return RT_ERR_ENTRY_FULL - when global storm type set to disable the per port setting for this storm type will also set to disable for all port.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_INPUT	invalid input parameter

19.23.rtk_rate_stormBypass_set

`int32 rtk_rate_stormBypass_set(rtk_storm_bypass_t type, rtk_enable_t enable)`

Set bypass storm filter control configuration.

Defined in: rate.h

Parameters

type
Bypass storm filter control type.
enable
Bypass status.

Comments

This API can set per-port bypass storm filter control frame type including RMA and igmp. The bypass frame type is as following:

- BYPASS_BRG_GROUP,
- BYPASS_FD_PAUSE,
- BYPASS_SP_MCAST,
- BYPASS_1X_PAE,
- BYPASS_UNDEF_BRG_04,
- BYPASS_UNDEF_BRG_05,
- BYPASS_UNDEF_BRG_06,
- BYPASS_UNDEF_BRG_07,
- BYPASS_PROVIDER_BRIDGE_GROUP_ADDRESS,
- BYPASS_UNDEF_BRG_09,
- BYPASS_UNDEF_BRG_0A,
- BYPASS_UNDEF_BRG_0B,
- BYPASS_UNDEF_BRG_0C,
- BYPASS_PROVIDER_BRIDGE_GVRP_ADDRESS,
- BYPASS_8021AB,
- BYPASS_UNDEF_BRG_0F,
- BYPASS_BRG_MNGEMENT,
- BYPASS_UNDEFINED_11,
- BYPASS_UNDEFINED_12,
- BYPASS_UNDEFINED_13,
- BYPASS_UNDEFINED_14,
- BYPASS_UNDEFINED_15,

- BYPASS_UNDEFINED_16,
- BYPASS_UNDEFINED_17,
- BYPASS_UNDEFINED_18,
- BYPASS_UNDEFINED_19,
- BYPASS_UNDEFINED_1A,
- BYPASS_UNDEFINED_1B,
- BYPASS_UNDEFINED_1C,
- BYPASS_UNDEFINED_1D,
- BYPASS_UNDEFINED_1E,
- BYPASS_UNDEFINED_1F,
- BYPASS_GMRP,
- BYPASS_GVRP,
- BYPASS_UNDEF_GARP_22,
- BYPASS_UNDEF_GARP_23,
- BYPASS_UNDEF_GARP_24,
- BYPASS_UNDEF_GARP_25,
- BYPASS_UNDEF_GARP_26,
- BYPASS_UNDEF_GARP_27,
- BYPASS_UNDEF_GARP_28,
- BYPASS_UNDEF_GARP_29,
- BYPASS_UNDEF_GARP_2A,
- BYPASS_UNDEF_GARP_2B,
- BYPASS_UNDEF_GARP_2C,
- BYPASS_UNDEF_GARP_2D,
- BYPASS_UNDEF_GARP_2E,
- BYPASS_UNDEF_GARP_2F,
- BYPASS_IGMP.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.
RT_ERR_ENABLE	Invalid IFG parameter

19.24.rtk_rate_stormBypass_get

```
int32 rtk_rate_stormBypass_get(rtk_storm_bypass_t type, rtk_enable_t
*pEnable)
```

Get bypass storm filter control configuration.

Defined in: rate.h

Parameters

	<i>type</i> Bypass storm filter control type.
	<i>*pEnable</i> Bypass status.
Comments	This API can get per-port bypass storm filter control frame type including RMA and igmp. The bypass frame type is as following: - BYPASS_BRG_GROUP, - BYPASS_FD_PAUSE, - BYPASS_SP_MCAST, - BYPASS_1X_PAE, - BYPASS_UNDEF_BRG_04, - BYPASS_UNDEF_BRG_05, - BYPASS_UNDEF_BRG_06, - BYPASS_UNDEF_BRG_07, - BYPASS_PROVIDER_BRIDGE_GROUP_ADDRESS, - BYPASS_UNDEF_BRG_09, - BYPASS_UNDEF_BRG_0A, - BYPASS_UNDEF_BRG_0B, - BYPASS_UNDEF_BRG_0C, - BYPASS_PROVIDER_BRIDGE_GVRP_ADDRESS, - BYPASS_8021AB, - BYPASS_UNDEF_BRG_0F, - BYPASS_BRG_MNGEMENT, - BYPASS_UNDEFINED_11, - BYPASS_UNDEFINED_12, - BYPASS_UNDEFINED_13, - BYPASS_UNDEFINED_14, - BYPASS_UNDEFINED_15, - BYPASS_UNDEFINED_16, - BYPASS_UNDEFINED_17, - BYPASS_UNDEFINED_18, - BYPASS_UNDEFINED_19, - BYPASS_UNDEFINED_1A, - BYPASS_UNDEFINED_1B, - BYPASS_UNDEFINED_1C, - BYPASS_UNDEFINED_1D, - BYPASS_UNDEFINED_1E, - BYPASS_UNDEFINED_1F, - BYPASS_GMRP, - BYPASS_GVRP, - BYPASS_UNDEF_GARP_22, - BYPASS_UNDEF_GARP_23, - BYPASS_UNDEF_GARP_24, - BYPASS_UNDEF_GARP_25, - BYPASS_UNDEF_GARP_26,

- BYPASS_UNDEF_GARP_27,
- BYPASS_UNDEF_GARP_28,
- BYPASS_UNDEF_GARP_29,
- BYPASS_UNDEF_GARP_2A,
- BYPASS_UNDEF_GARP_2B,
- BYPASS_UNDEF_GARP_2C,
- BYPASS_UNDEF_GARP_2D,
- BYPASS_UNDEF_GARP_2E,
- BYPASS_UNDEF_GARP_2F,
- BYPASS_IGMP.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.

19.25.rtk_rate_shareMeter_set

```
int32 rtk_rate_shareMeter_set(uint32 index, uint32 rate, rtk_enable_t ifgInclude)
```

Set meter configuration

Defined in: rate.h

Parameters

index
shared meter index

rate
rate of share meter

ifgInclude
include IFG or not, ENABLE:include DISABLE:exclude

Comments

The API can set shared meter rate and ifg include for each meter. The rate unit is 1 kbps and the range is from 8k to 1048568k. The granularity of rate is 8 kbps. The ifg_include parameter is used for rate calculation with/without inter-frame-gap and preamble.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_FILTER_METER_ID	Invalid meter
RT_ERR_RATE	Invalid rate
RT_ERR_INPUT	Invalid input parameters

19.26.rtk_rate_shareMeter_get

```
int32 rtk_rate_shareMeter_get(uint32 index, uint32 *pRate, rtk_enable_t
    *pIfgInclude)
```

Get meter configuration

Defined in: rate.h

Parameters

index
shared meter index

**pRate*
pointer of rate of share meter

**pIfgInclude*
include IFG or not, ENABLE:include DISABLE:exclude

Comments

The API can get shared meter rate and ifg include for each meter. The rate unit is 1 kbps and the granularity of rate is 8 kbps. The ifg_include parameter is used for rate calculation with/without inter-frame-gap and preamble

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_FILTER_METER_ID	Invalid meter

19.27.rtk_rate_shareMeterBucket_set

```
int32 rtk_rate_shareMeterBucket_set(uint32 index, uint32 bucketSize)
```

Set meter Bucket Size

Defined in: rate.h

Parameters

index
shared meter index

bucketSize
Bucket Size

Comments

The API can set shared meter bucket size.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Error Input

RT_ERR_FILTER_METER_ID	Invalid meter
------------------------	---------------

19.28.rtk_rate_shareMeterBucket_get

int32 rtk_rate_shareMeterBucket_get(uint32 index, uint32 *pBucketSize)

Get meter Bucket Size

Defined in: rate.h

Parameters

index
shared meter index

**pBucketSize*
Bucket Size

Comments

The API can get shared meter bucket size.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_FILTER_METER_ID	Invalid meter

19.29.rtk_rate_shareMeterExceed_get

int32 rtk_rate_shareMeterExceed_get(uint32 index, uint32 *pIsExceed)

Get exceed meter status.

Defined in: rate.h

Parameters

index
shared meter index
**pIsExceed*
pointer to exceed status

Comments

Exceed status is as following

- TRUE - rate is more than configured rate.
- FALSE - rate is not over then configured rate.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

RT_ERR_NULL_POINTER	input parameter may be null pointer
---------------------	-------------------------------------

19.30.rtk_rate_shareMeterExceed_clear

int32 rtk_rate_shareMeterExceed_clear(uint32 index)

Clear share meter exceed status.

Defined in: rate.h

Parameters

index
shared meter index

Comments

Exceed status is as following
 - TRUE - rate is more than configured rate.
 - FALSE - rate is not over then configured rate.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

20. Module RLDP and RLPP API

Filename: rldp.h

Description

The file have include the following module and sub-modules 1) RLDP and RLPP configuration and status

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

20.1. List of Symbols

Here is a list of all functions and variables in this module

- rldp.h - Declaration of RLDP and RLPP API
- rtk_rldp_init
- rtk_rldp_config_set
- rtk_rldp_config_get
- rtk_rldp_portConfig_set
- rtk_rldp_portConfig_get

```
rtk_rldp_status_get
rtk_rldp_portStatus_get
rtk_rldp_portStatus_clear
rtk_rlpp_init
rtk_rlpp_trapType_set
rtk_rlpp_trapType_get
```

20.2. rtk_rldp_init

`int32 rtk_rldp_init(void)`

Initialize rldp module.

Defined in: rldp.h

Parameters

`void`

Comments Must initialize rldp module before calling any rldp APIs.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

20.3. rtk_rldp_config_set

`int32 rtk_rldp_config_set(rtk_rldp_config_t *pConfig)`

Set RLDP module configuration

Defined in: rldp.h

Parameters

`*pConfig`

configuration structure of RLDP

Comments None

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_INPUT`

`RT_ERR_NULL_POINTER`

20.4. rtk_rldp_config_get

`int32 rtk_rldp_config_get(rtk_rldp_config_t *pConfig)`

Get RLDP module configuration

Defined in: rldp.h

Parameters `*pConfig`
 configuration structure of RLDP

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	
	RT_ERR_NULL_POINTER	

20.5. rtk_rldp_portConfig_set

`int32 rtk_rldp_portConfig_set(rtk_port_t port, rtk_rldp_portConfig_t *pPortConfig)`

Set per port RLDP module configuration

Defined in: rldp.h

Parameters `port`
 port number to be configured

`*pPortConfig`
 per port configuration structure of RLDP

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	
	RT_ERR_NULL_POINTER	

20.6. rtk_rldp_portConfig_get

```
int32 rtk_rldp_portConfig_get(rtk_port_t port, rtk_rldp_portConfig_t
*pPortConfig)
```

Get per port RLDP module configuration

Defined in: rldp.h

Parameters

port
port number to be get

**pPortConfig*
per port configuration structure of RLDP

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	
RT_ERR_NULL_POINTER	

20.7. rtk_rldp_status_get

```
int32 rtk_rldp_status_get(rtk_rldp_status_t *pStatus)
```

Get RLDP module status

Defined in: rldp.h

Parameters

**pStatus*
status structure of RLDP

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	

20.8. rtk_rldp_portStatus_get

```
int32 rtk_rldp_portStatus_get(rtk_port_t port, rtk_rldp_portStatus_t
    *pPortStatus)
```

Get RLDP module status

Defined in: rldp.h

Parameters

port
port number to be get

**pPortStatus*
per port status structure of RLDP

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	
RT_ERR_NULL_POINTER	

20.9. rtk_rldp_portStatus_clear

```
int32 rtk_rldp_portStatus_clear(rtk_port_t port, rtk_rldp_portStatus_t
    *pPortStatus)
```

Clear RLDP module status

Defined in: rldp.h

Parameters

port
port number to be clear
**pPortStatus*
per port status structure of RLDP

Comments

Clear operation effect loop_enter and loop_leave only, other field in the structure are don't care

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	
RT_ERR_NULL_POINTER	

20.10.rtk_rlpp_init

`int32 rtk_rlpp_init(void)`

Initialize rlpp module.

Defined in: rldp.h

Parameters

void

Comments

Must initialize rlpp module before calling any rlpp APIs.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

20.11.rtk_rlpp_trapType_set

`int32 rtk_rlpp_trapType_set(rtk_rlpp_trapType_t type)`

Set RLPP trap to cpu operation, trap or not trap

Defined in: rldp.h

Parameters

type

RLPP trap operation type

Comments

Trap the RLPP packet to CPU for software processing

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

20.12.rtk_rlpp_trapType_get

`int32 rtk_rlpp_trapType_get(rtk_rlpp_trapType_t *pType)`

Get RLPP trap to cpu operation, trap or not trap

	Defined in: rldp.h
Parameters	<i>*pType</i> RLPP trap operation type
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER

21. Module Security API

Filename: sec.h

Description The file includes the following modules and sub-modules
(1) attack prevention

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

21.1. List of Symbols

Here is a list of all functions and variables in this module

sec.h - Definition of Security API
 rtk_sec_init
 rtk_sec_portAttackPreventState_get
 rtk_sec_portAttackPreventState_set
 rtk_sec_attackPrevent_get
 rtk_sec_attackPrevent_set
 rtk_sec_attackFloodThresh_get
 rtk_sec_attackFloodThresh_set

21.2. rtk_sec_init

int32 rtk_sec_init(void)

Initialize security module.

Defined in: sec.h

Parameters	<i>void</i>				
Comments	Must initialize security module before calling any sec APIs.				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

21.3. rtk_sec_portAttackPreventState_get

```
int32 rtk_sec_portAttackPreventState_get(rtk_port_t port, rtk_enable_t
                                         *pEnable)
```

Per port get attack prevention confi state

Defined in: sec.h

Parameters	<i>port</i> port id <i>*pEnable</i> status attack prevention								
Comments	The status attack prevention: - DISABLED - ENABLED								
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>Invalid port id</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>NULL pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_PORT_ID	Invalid port id	RT_ERR_NULL_POINTER	NULL pointer
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_PORT_ID	Invalid port id								
RT_ERR_NULL_POINTER	NULL pointer								

21.4. rtk_sec_portAttackPreventState_set

```
int32 rtk_sec_portAttackPreventState_set(rtk_port_t port, rtk_enable_t
                                         enable)
```

Per port set attack prevention confi state

Defined in: sec.h

Parameters	<i>port</i> port id.
-------------------	-------------------------

	<i>enable</i>	
	status attack prevention	
Comments	The status attack prevention:	
	- DISABLED	
	- ENABLED	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port id

21.5. rtk_sec_attackPrevent_get

```
int32 rtk_sec_attackPrevent_get(rtk_sec_attackType_t attackType,  
                               rtk_action_t *pAction)
```

Get action for each kind of attack on specified port.

Defined in: sec.h

Parameters	<i>attackType</i>	
	type of attack	
	* <i>pAction</i>	pointer to action for attack

Comments	Action is as following:	
	- ACTION_TRAP2CPU	
	- ACTION_DROP	
	- ACTION_FORWARD	

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_INPUT	invalid input parameter
	RT_ERR_NULL_POINTER	input parameter may be null pointer

21.6. rtk_sec_attackPrevent_set

```
int32 rtk_sec_attackPrevent_set(rtk_sec_attackType_t attackType,  
                               rtk_action_t action)
```

	Set action for each kind of attack.										
	Defined in: sec.h										
Parameters	<p><i>attackType</i> type of attack</p> <p><i>action</i> action for attack</p>										
Comments	Action is as following: - ACTION_TRAP2CPU - ACTION_DROP - ACTION_FORWARD										
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_FWD_ACTION</td> <td>invalid forwarding action</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>invalid input parameter</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_FWD_ACTION	invalid forwarding action	RT_ERR_INPUT	invalid input parameter
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_NOT_INIT	The module is not initial										
RT_ERR_FWD_ACTION	invalid forwarding action										
RT_ERR_INPUT	invalid input parameter										

21.7. rtk_sec_attackFloodThresh_get

```
int32 rtk_sec_attackFloodThresh_get(rtk_sec_attackFloodType_t type,
                                     uint32 *pFloodThresh)
```

Get flood threshold, time unit 1ms.

Defined in: sec.h

Parameters	<p><i>type</i> pointer to flood threshold</p> <p><i>*pFloodThresh</i> action for attack</p>								
Comments	Flood type is as following: - SEC_ICMPFLOOD - SEC_SYNCFLOOD - SEC_FINFLOOD								
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_NOT_INIT	The module is not initial								
RT_ERR_NULL_POINTER	input parameter may be null pointer								

21.8. rtk_sec_attackFloodThresh_set

```
int32 rtk_sec_attackFloodThresh_set(rtk_sec_attackFloodType_t type, uint32
floodThresh)
```

Set flood threshold, time unit 1ms.

Defined in: sec.h

Parameters

type
flood threshold

floodThresh
action for attack

Comments

Flood type is as following:
- SEC_ICMPFLOOD
- SEC_SYNCFLOOD
- SEC_FINFLOOD

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

22. Module Statistic API

Filename: stat.h

Description

The file includes the following modules and sub-modules
(1) Statistic Counter Reset
(2) Statistic Counter Get

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

22.1. List of Symbols

Here is a list of all functions and variables in this module

stat.h - Definition of Statistic API
rtk_stat_init
rtk_stat_global_reset

```
rtk_stat_port_reset  
rtk_stat_log_reset  
rtk_stat_rstCntValue_set  
rtk_stat_rstCntValue_get  
rtk_stat_global_get  
rtk_stat_global_getAll  
rtk_stat_port_get  
rtk_stat_port_getAll  
rtk_stat_log_get  
rtk_stat_logCtrl_set  
rtk_stat_logCtrl_get  
rtk_stat_mibCntMode_get  
rtk_stat_mibCntMode_set  
rtk_stat_mibLatchTimer_get  
rtk_stat_mibLatchTimer_set  
rtk_stat_mibSyncMode_get  
rtk_stat_mibSyncMode_set  
rtk_stat_mibCntTagLen_get  
rtk_stat_mibCntTagLen_set  
rtk_stat_pktInfo_get
```

22.2. rtk_stat_init

`int32 rtk_stat_init(void)`

Initialize stat module of the specified device.

Defined in: stat.h

Parameters

`void`

Comments

Must initialize stat module before calling any stat APIs.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_STAT_GLOBAL_CNTR_FAI</code>	Could not retrieve/reset Global Counter
<code>RT_ERR_STAT_PORT_CNTR_FAIL</code>	Could not retrieve/reset Port Counter

22.3. rtk_stat_global_reset

`int32 rtk_stat_global_reset(void)`

Reset the global counters.

Defined in: stat.h

Parameters *void*

Comments None

Return Codes RT_ERR_OK ok
RT_ERR_FAILED failed
RT_ERR_STAT_GLOBAL_CNTR_FAI L
RT_ERR_STAT_GLOBAL_CNTR_FAI L Could not retrieve/reset Global Counter

22.4. rtk_stat_port_reset

`int32 rtk_stat_port_reset(rtk_port_t port)`

Reset the specified port counters in the specified device.

Defined in: stat.h

Parameters *port*
port id

Comments None

Return Codes RT_ERR_OK ok
RT_ERR_FAILED failed
RT_ERR_PORT_ID invalid port id
RT_ERR_STAT_PORT_CNTR_FAIL Could not retrieve/reset Port Counter

22.5. rtk_stat_log_reset

`int32 rtk_stat_log_reset(uint32 index)`

Reset the specified ACL log counters.

Defined in: stat.h

Parameters

index
log index

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	invalid port id

22.6. rtk_stat_rstCntValue_set

`int32 rtk_stat_rstCntValue_set(rtk_mib_RST_value_t rstValue)`

Set the counter value after reset

Defined in: stat.h

Parameters

rstValue
the counter value after reset

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	invalid port id

22.7. rtk_stat_rstCntValue_get

`int32 rtk_stat_rstCntValue_get(rtk_mib_RST_value_t *pRstValue)`

Get the counter value after reset

	Defined in: stat.h
Parameters	* <i>pRstValue</i> pointer buffer of value
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT RT_ERR_NULL_POINTER
	ok failed invalid port id Could not retrieve/reset Port Counter

22.8. rtk_stat_global_get

`int32 rtk_stat_global_get(rtk_stat_global_type_t cntrIdx, uint64 *pCntr)`

Get one specified global counter in the specified device.

	Defined in: stat.h
Parameters	<i>cntrIdx</i> specified global counter index <i>*pCntr</i> pointer buffer of counter value
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_UNIT_ID RT_ERR_NULL_POINTER RT_ERR_STAT_GLOBAL_CNTR_FAILED RT_ERR_STAT_INVALID_GLOBAL_CNTR
	ok failed invalid unit id input parameter may be null pointer Could not retrieve/reset Global Counter Invalid Global Counter

22.9. rtk_stat_global_getAll

`int32 rtk_stat_global_getAll(rtk_stat_global_cntr_t *pGlobalCntrs)`

Get all global counters in the specified device.

Defined in: stat.h

Parameters	<i>*pGlobalCntrs</i>	pointer buffer of global counter structure
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER RT_ERR_STAT_GLOBAL_CNTR_FAIL	ok failed input parameter may be null pointer Could not retrieve/reset Global Counter
	RT_ERR_STAT_INVALID_GLOBAL_CNTR	Invalid Global Counter

22.10.rtk_stat_port_get

```
int32 rtk_stat_port_get(rtk_port_t port, rtk_stat_port_type_t cntrIdx, uint64
                        *pCntr)
```

Get one specified port counter.

Defined in: stat.h

Parameters	<i>port</i>	port id
	<i>cntrIdx</i>	specified port counter index
	<i>*pCntr</i>	pointer buffer of counter value
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_PORT_ID RT_ERR_NULL_POINTER RT_ERR_STAT_PORT_CNTR_FAIL	ok failed invalid port id input parameter may be null pointer Could not retrieve/reset Port Counter

22.11.rtk_stat_port_getAll

`int32 rtk_stat_port_getAll(rtk_port_t port, rtk_stat_port_cntr_t *pPortCntrs)`

Get all counters of one specified port in the specified device.

Defined in: stat.h

Parameters

port
port id
**pPortCntrs*
pointer buffer of counter value

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_STAT_PORT_CNTR_FAIL	Could not retrieve/reset Port Counter

22.12.rtk_stat_log_get

`int32 rtk_stat_log_get(uint32 index, uint64 *pCntr)`

Get ACL logging counter.

Defined in: stat.h

Parameters

index
logging index
**pCntr*
pointer buffer of counter value

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	invalid index
RT_ERR_NULL_POINTER	input parameter may be null pointer

22.13.rtk_stat_logCtrl_set

`int32 rtk_stat_logCtrl_set(uint32 index, rtk_stat_log_ctrl_t ctrl)`

set the acl log counters type for packet or byte counter

Defined in: stat.h

Parameters

index

index of ACL log counter

ctrl

log counter control setting

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

invalid index

22.14.rtk_stat_logCtrl_get

`int32 rtk_stat_logCtrl_get(uint32 index, rtk_stat_log_ctrl_t *pCtrl)`

get the acl log counters type for packet or byte counter

Defined in: stat.h

Parameters

index

index of ACL log counter

**pCtrl*

log counter control setting

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

invalid index

22.15.rtk_stat_mibCntMode_get

`int32 rtk_stat_mibCntMode_get(rt_k_mib_count_mode_t *pCnt_mode)`

Get the MIB data update mode

Defined in: stat.h

Parameters `*pCnt_mode`
 pointer buffer of MIB data update mode

Comments None

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_INPUT</code>	invalid index
	<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

22.16.rtk_stat_mibCntMode_set

`int32 rtk_stat_mibCntMode_set(rt_k_mib_count_mode_t cnt_mode)`

Set MIB data update mode

Defined in: stat.h

Parameters `cnt_mode`
 MIB counter update mode

Comments None

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_INPUT</code>	invalid index

22.17.rtk_stat_mibLatchTimer_get

`int32 rtk_stat_mibLatchTimer_get(uint32 *pTimer)`

Get the MIB latch timer

Defined in: stat.h

Parameters	<i>*pTimer</i>
	pointer buffer of MIB latch timer
Comments	None
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed
	RT_ERR_INPUT invalid index
	RT_ERR_NULL_POINTER input parameter may be null pointer

22.18.rtk_stat_mibLatchTimer_set

`int32 rtk_stat_mibLatchTimer_set(uint32 timer)`

Set MIB data update mode

Defined in: stat.h

Parameters	<i>timer</i>
	MIB latch timer
Comments	None
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed
	RT_ERR_INPUT invalid index

22.19.rtk_stat_mibSyncMode_get

`int32 rtk_stat_mibSyncMode_get(rtk_mib_sync_mode_t *pSync_mode)`

Get the MIB register data update mode

Defined in: stat.h

Parameters	<i>*pSync_mode</i>
	pointer buffer of MIB register data update mode
Comments	None
Return Codes	

RT_ERR_FAILED	failed
RT_ERR_INPUT	invalid index
RT_ERR_NULL_POINTER	input parameter may be null pointer

22.20.rtk_stat_mibSyncMode_set

`int32 rtk_stat_mibSyncMode_set(rtk_mib_sync_mode_t sync_mode)`

Set MIB register data update mode

Defined in: stat.h

Parameters

sync_mode
MIB register data update mode

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	invalid index

22.21.rtk_stat_mibCntTagLen_get

`int32 rtk_stat_mibCntTagLen_get(rtk_mib_tag_cnt_dir_t direction,
rtk_mib_tag_cnt_state_t *pState)`

Get counting Tag length state in tx/rx packet

Defined in: stat.h

Parameters

direction
count tx or rx tag length
**pState*
pointer buffer of count tx/rx tag length state

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	invalid index
RT_ERR_NULL_POINTER	input parameter may be null pointer

22.22.rtk_stat_mibCntTagLen_set

```
int32 rtk_stat_mibCntTagLen_set(rtk_mib_tag_cnt_dir_t direction,
                                rtk_mib_tag_cnt_state_t state)
```

Set counting length including Ctag length or excluding Ctag length for tx/rx packet

Defined in: stat.h

Parameters

direction
count tx or rx tag length

state
count tag length state

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	invalid index

22.23.rtk_stat_pktInfo_get

```
int32 rtk_stat_pktInfo_get(rtk_port_t port, uint32 *pCode)
```

Get the newest packet trap/drop reason

Defined in: stat.h

Parameters

port
port index
**pCode*
the newest packet trap/drop reason code

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid index

23. Module STP APIs

Filename: stp.h

Description The file have include the following module and sub-modules 1) spanning tree (1D, 1w and 1s)

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

23.1. List of Symbols

Here is a list of all functions and variables in this module

stp.h - Definition those public STP APIs and its data type in the SDK.
rtk_stp_init
rtk_stp_mstpState_get
rtk_stp_mstpState_set

23.2. rtk_stp_init

`int32 rtk_stp_init(void)`

Initialize stp module of the specified device.

Defined in: stp.h

Parameters `void`

Comments Must initialize stp module before calling any stp APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

23.3. rtk_stp_mstpState_get

`int32 rtk_stp_mstpState_get(uint32 msti, rtk_port_t port, rtk_stp_state_t *pStpState)`

	Get port spanning tree state of the msti from the specified device.													
	Defined in: stp.h													
Parameters	<p><i>msti</i> multiple spanning tree instance</p> <p><i>port</i> port id</p> <p><i>*pStpState</i> pointer buffer of spanning tree state</p>													
Comments	1. For single spanning tree mode, input CIST0 (msti=0). 2. Spanning tree state as following - STP_STATE_DISABLED - STP_STATE_BLOCKING - STP_STATE_LEARNING - STP_STATE_FORWARDING													
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_MSTI</td> <td>invalid msti</td> </tr> <tr> <td>RT_ERR_PORT_ID</td> <td>invalid port id</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_MSTI	invalid msti	RT_ERR_PORT_ID	invalid port id	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok													
RT_ERR_FAILED	failed													
RT_ERR_NOT_INIT	The module is not initial													
RT_ERR_MSTI	invalid msti													
RT_ERR_PORT_ID	invalid port id													
RT_ERR_NULL_POINTER	input parameter may be null pointer													

23.4. rtk_stp_mstpState_set

```
int32 rtk_stp_mstpState_set(uint32 msti, rtk_port_t port, rtk_stp_state_t
                           stpState)
```

Set port spanning tree state of the msti to the specified device.

Defined in: stp.h

Parameters	<i>msti</i> multiple spanning tree instance
	<i>port</i> port id
	<i>stpState</i> spanning tree state

Comments 1. For single spanning tree mode, input CIST0 (msti=0). 2. Spanning tree state as following

	- STP_STATE_DISABLED - STP_STATE_BLOCKING - STP_STATE_LEARNING - STP_STATE_FORWARDING	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_MSTI	invalid msti
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_MSTP_STATE	invalid spanning tree status

24. Module SVLAN API

Filename: svlan.h

Description The file includes the following modules and sub-modules
(1) 802.1ad, SVLAN [VLAN Stacking]

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

24.1. List of Symbols

Here is a list of all functions and variables in this module

svlan.h - Definition of SVLAN API
 rtk_svlan_init
 rtk_svlan_create
 rtk_svlan_destroy
 rtk_svlan_portSvid_get
 rtk_svlan_portSvid_set
 rtk_svlan_servicePort_get
 rtk_svlan_servicePort_set
 rtk_svlan_memberPort_set
 rtk_svlan_memberPort_get
 rtk_svlan_tpidEntry_get
 rtk_svlan_tpidEntry_set
 rtk_svlan_priorityRef_set
 rtk_svlan_priorityRef_get
 rtk_svlan_memberPortEntry_set
 rtk_svlan_memberPortEntry_get

```
rtk_svlan_ipmc2s_add
rtk_svlan_ipmc2s_del
rtk_svlan_ipmc2s_get
rtk_svlan_l2mc2s_add
rtk_svlan_l2mc2s_del
rtk_svlan_l2mc2s_get
rtk_svlan_sp2c_add
rtk_svlan_sp2c_get
rtk_svlan_sp2c_del
rtk_svlan_dmacVidSelState_set
rtk_svlan_dmacVidSelState_get
rtk_svlan_unmatchAction_set
rtk_svlan_unmatchAction_get
rtk_svlan_untagAction_set
rtk_svlan_untagAction_get
rtk_svlan_c2s_add
rtk_svlan_c2s_del
rtk_svlan_c2s_get
rtk_svlan_trapPri_get
rtk_svlan_trapPri_set
rtk_svlan_deiKeepState_get
rtk_svlan_deiKeepState_set
rtk_svlan_lookupType_get
rtk_svlan_lookupType_set
rtk_svlan_sp2cUnmatchCtagging_get
rtk_svlan_sp2cUnmatchCtagging_set
rtk_svlan_priority_get
rtk_svlan_priority_set
rtk_svlan_fid_get
rtk_svlan_fid_set
rtk_svlan_fidEnable_get
rtk_svlan_fidEnable_set
rtk_svlan_enhancedFid_get
rtk_svlan_enhancedFid_set
rtk_svlan_enhancedFidEnable_get
rtk_svlan_enhancedFidEnable_set
rtk_svlan_dmacVidSelForcedState_set
rtk_svlan_dmacVidSelForcedState_get
```

24.2. rtk_svlan_init

`int32 rtk_svlan_init(void)`

Initialize svlan module.

Defined in: svlan.h

Parameters *void*

Comments Must initialize svlan module before calling any svlan APIs.

Return Codes RT_ERR_OK ok
RT_ERR_FAILED failed

24.3. rtk_svlan_create

`int32 rtk_svlan_create(rtk_vlan_t svid)`

Create the svlan.

Defined in: svlan.h

Parameters *svid*
svlan id to be created

Comments None

Return Codes RT_ERR_OK ok
RT_ERR_FAILED failed
RT_ERR_SVLAN_EXIST SVLAN entry is exist

24.4. rtk_svlan_destroy

`int32 rtk_svlan_destroy(rtk_vlan_t svid)`

Destroy the svlan.

Defined in: svlan.h

Parameters *svid*
svlan id to be destroyed

Comments None

Return Codes RT_ERR_OK ok

RT_ERR_FAILED	failed
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found

24.5. rtk_svlan_portSvid_get

int32 rtk_svlan_portSvid_get(rtk_port_t port, rtk_vlan_t *pSvid)

Get port default svlan id.

Defined in: svlan.h

Parameters	<i>port</i> port id	
	<i>*pSvid</i> pointer buffer of port default svlan id	
Comments	None	
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

24.6. rtk_svlan_portSvid_set

int32 rtk_svlan_portSvid_set(rtk_port_t port, rtk_vlan_t svid)

Set port default svlan id.

Defined in: svlan.h

Parameters	<i>port</i> port id	
	<i>svid</i> port default svlan id	
Comments	None	
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

RT_ERR_PORT_ID	invalid port id
----------------	-----------------

24.7. rtk_svlan_servicePort_get

`int32 rtk_svlan_servicePort_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get service ports from the specified device.

Defined in: svlan.h

Parameters

port
port id
**pEnable*
status of service port

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

24.8. rtk_svlan_servicePort_set

`int32 rtk_svlan_servicePort_set(rtk_port_t port, rtk_enable_t enable)`

Set service ports to the specified device.

Defined in: svlan.h

Parameters

port
port id
enable
status of service port

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

24.9. rtk_svlan_memberPort_set

```
int32 rtk_svlan_memberPort_set(rtk_vlan_t svvid, rtk_portmask_t
    *pSvlanPortmask, rtk_portmask_t *pSvlanUntagPortmask)
```

Replace the svlan members.

Defined in: svlan.h

Parameters

<i>svvid</i>	svlan id
<i>*pSvlanPortmask</i>	svlan member ports
<i>*pSvlanUntagPortmask</i>	svlan untag member ports

Comments

(1) Don't care the original svlan members and replace with new configure directly.
(2) svlan portmask only for svlan ingress filter checking

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SVLAN_ENTRY_INDEX	invalid svid entry no
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found

24.10.rtk_svlan_memberPort_get

```
int32 rtk_svlan_memberPort_get(rtk_vlan_t svvid, rtk_portmask_t
    *pSvlanPortmask, rtk_portmask_t *pSvlanUntagPortmask)
```

Get the svlan members.

Defined in: svlan.h

Parameters

<i>svvid</i>	svlan id
<i>*pSvlanPortmask</i>	svlan member ports
<i>*pSvlanUntagPortmask</i>	svlan untag member ports

Comments

- (1) Don't care the original svlan members and replace with new configure directly.
- (2) svlan portmask only for svlan ingress filter checking

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_SVLAN_ENTRY_INDEX	invalid svid entry no
	RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found

24.11.rtk_svlan_tpidEntry_get

`int32 rtk_svlan_tpidEntry_get(uint32 svlanIndex, uint32 *pSvlanTagId)`

Get the svlan TPID.

Defined in: svlan.h

Parameters

svlanIndex
index of svlan table

**pSvlanTagId*
pointer buffer of svlan TPID

Comments

Only support pSvlanTagId 0 in Apollo.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	invalid unit id
RT_ERR_NULL_POINTER	input parameter may be null pointer

24.12.rtk_svlan_tpidEntry_set

`int32 rtk_svlan_tpidEntry_set(uint32 svlan_index, uint32 svlan_tag_id)`

Set the svlan TPID.

Defined in: svlan.h

Parameters

svlan_index
index of svlan table

svlan_tag_id
svlan TPID

Comments Only support pSvlan_tag_id 0 in Apollo.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

24.13.rtk_svlan_priorityRef_set

`int32 rtk_svlan_priorityRef_set(rtk_svlan_pri_ref_t ref)`

Set S-VLAN upstream priority reference setting.

Defined in: svlan.h

Parameters
ref reference selection parameter.

Comments The API can set the upstream SVLAN tag priority reference source. The related priority sources are as following:

- REF_INTERNAL_PRI,
- REF_CTAG_PRI,
- REF_SVLAN_PRI.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameter.

24.14.rtk_svlan_priorityRef_get

`int32 rtk_svlan_priorityRef_get(rtk_svlan_pri_ref_t *pRef)`

Get S-VLAN upstream priority reference setting.

Defined in: svlan.h

Parameters
**pRef* reference selection parameter.

Comments The API can get the upstream SVLAN tag priority reference source. The related priority sources are as following:
- REF_INTERNAL_PRI,

- REF_CTAG_PRI,
- REF_SVLAN_PRI.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_SMI	SMI access error

24.15.rtk_svlan_memberPortEntry_set

`int32 rtk_svlan_memberPortEntry_set(rtk_svlan_memberCfg_t *pSvlan_cfg)`

Configure system SVLAN member content

Defined in: svlan.h

Parameters	<code>*pSvlan_cfg</code>
	SVLAN member configuration

Comments	The API can set system 64 accepted s-tag frame format. Only 64 SVID S-tag frame will be accepted to receiving from uplink ports. Other SVID S-tag frame or S-untagged frame will be dropped by default setup. - rtk_svlan_memberCfg_t->svid is SVID of SVLAN member configuration. - rtk_svlan_memberCfg_t->memberport is member port mask of SVLAN member configuration. - rtk_svlan_memberCfg_t->fid is filtering database of SVLAN member configuration. - rtk_svlan_memberCfg_t->priority is priority of SVLAN member configuration.
-----------------	--

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameter.
	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
	RT_ERR_PORT_MASK	Invalid portmask.
	RT_ERR_SVLAN_TABLE_FULL	SVLAN configuration is full.

24.16.rtk_svlan_memberPortEntry_get

`int32 rtk_svlan_memberPortEntry_get(rtk_svlan_memberCfg_t *pSvlan_cfg)`

Get SVLAN member Configure.

	Defined in: svlan.h
Parameters	* <i>pSvlan_cfg</i> SVLAN member configuration
Comments	The API can get system 64 accepted s-tag frame format. Only 64 SVID S-tag frame will be accepted to receiving from uplink ports. Other SVID S-tag frame or S-untagged frame will be dropped.
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_SVLAN_ENTRY_NOT_FOU specified svlan entry not found. ND RT_ERR_INPUT Invalid input parameters.

24.17.rtk_svlan_ipmc2s_add

```
int32 rtk_svlan_ipmc2s_add(ipaddr_t ipmc, ipaddr_t ipmcMsk, rtk_vlan_t svid)
```

add ip multicast address to SVLAN

Defined in: svlan.h
Parameters
<i>ipmc</i> ip multicast address
<i>ipmcMsk</i> ip multicast address mask
<i>svid</i> SVLAN VID

Comments	The API can set IP multicast to SVID configuration. If upstream packet is IPv4 multicast packet and DIP is matched MC2S configuration, ASIC will assign egress SVID to the packet. There are 8 SVLAN multicast configurations for IP and L2 multicast.
-----------------	--

Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed RT_ERR_SMI SMI access error RT_ERR_SVLAN_VID Invalid SVLAN VID parameter. RT_ERR_SVLAN_ENTRY_NOT_FOU specified svlan entry not found. ND RT_ERR_OUT_OF_RANGE input out of range.
---------------------	--

RT_ERR_INPUT

Invalid input parameters.

24.18.rtk_svlan_ipmc2s_del

int32 rtk_svlan_ipmc2s_del(ipaddr_t ipmc, ipaddr_t ipmcMsk)

delete ip multicast address to SVLAN

Defined in: svlan.h

Parameters*ipmc*

ip multicast address

ipmcMsk

ip multicast address mask

Comments

The API can delete IP multicast to SVID configuration. There are 8 SVLAN multicast configurations for IP and L2 multicast.

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_SMI

SMI access error

RT_ERR_SVLAN_VID

Invalid SVLAN VID parameter.

RT_ERR_OUT_OF_RANGE

input out of range.

24.19.rtk_svlan_ipmc2s_get

int32 rtk_svlan_ipmc2s_get(ipaddr_t ipmc, ipaddr_t ipmcMsk, rtk_vlan_t *pSvid)

Get ip multicast address to SVLAN

Defined in: svlan.h

Parameters*ipmc*

ip multicast address

ipmcMsk

ip multicast address mask

**pSvid*

SVLAN VID

Comments

The API can get IP multicast to SVID configuration. There are 8 SVLAN multicast configurations for IP and L2 multicast.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_OUT_OF_RANGE	input out of range.

24.20.rtk_svlan_l2mc2s_add

int32 rtk_svlan_l2mc2s_add(rtk_mac_t mac, rtk_mac_t macMsk, rtk_vlan_t svid)

Add L2 multicast address to SVLAN

Defined in: svlan.h

Parameters

<i>mac</i>	L2 multicast address
<i>macMsk</i>	L2 multicast address
<i>svid</i>	SVLAN VID

Comments

The API can set L2 Multicast to SVID configuration. If upstream packet is L2 multicast packet and DMAC is matched, ASIC will assign egress SVID to the packet. There are 32 SVLAN multicast configurations for IP and L2 multicast.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
RT_ERR_SVLAN_ENTRY_NOT_FOUND	specified svlan entry not found.
RT_ERR_OUT_OF_RANGE	input out of range.
RT_ERR_INPUT	Invalid input parameters.

24.21.rtk_svlan_l2mc2s_del

int32 rtk_svlan_l2mc2s_del(rtk_mac_t mac, rtk_mac_t macMsk)

	delete L2 multicast address to SVLAN									
	Defined in: svlan.h									
Parameters	<p><i>mac</i> L2 multicast address</p>									
	<p><i>macMsk</i> L2 multicast address</p>									
Comments	The API can delete Multicast to SVID configuration. There are 8 SVLAN multicast configurations for IP and L2 multicast.									
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_SVLAN_VID</td> <td>Invalid SVLAN VID parameter.</td> </tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td> <td>input out of range.</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.	RT_ERR_OUT_OF_RANGE	input out of range.
RT_ERR_OK	ok									
RT_ERR_FAILED	failed									
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.									
RT_ERR_OUT_OF_RANGE	input out of range.									

24.22.rtk_svlan_l2mc2s_get

	<code>int32 rtk_svlan_l2mc2s_get(rtk_mac_t mac, rtk_mac_t macMsk, rtk_vlan_t *pSvid)</code>									
	Get L2 multicast address to SVLAN									
	Defined in: svlan.h									
Parameters	<p><i>mac</i> L2 multicast address</p> <p><i>macMsk</i> L2 multicast address</p> <p><i>*pSvid</i> SVLAN VID</p>									
Comments	The API can get L2 mutlicast to SVID configuration. There are 32 SVLAN multicast configurations for IP and L2 multicast.									
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td> <td>input out of range.</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_INPUT	Invalid input parameters.	RT_ERR_OUT_OF_RANGE	input out of range.
RT_ERR_OK	ok									
RT_ERR_FAILED	failed									
RT_ERR_INPUT	Invalid input parameters.									
RT_ERR_OUT_OF_RANGE	input out of range.									

24.23.rtk_svlan_sp2c_add

`int32 rtk_svlan_sp2c_add(rtk_vlan_t svnid, rtk_port_t dstPort, rtk_vlan_t cvid)`

Add system SP2C configuration

Defined in: svlan.h

Parameters

svnid
VLAN ID

dstPort
Destination port of SVLAN to CVLAN configuration

cvid
SVLAN VID

Comments

The API can add SVID & Destination Port to CVLAN configuration. The downstream frames with assigned SVID will be add C-tag with assigned CVID if the output port is the assigned destination port. There are 128 SP2C configurations.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_PORT_ID</code>	Invalid port number.
<code>RT_ERR_SVLAN_VID</code>	Invalid SVLAN VID parameter.
<code>RT_ERR_VLAN_VID</code>	Invalid VID parameter.
<code>RT_ERR_OUT_OF_RANGE</code>	input out of range.
<code>RT_ERR_INPUT</code>	Invalid input parameters.

24.24.rtk_svlan_sp2c_get

`int32 rtk_svlan_sp2c_get(rtk_vlan_t svnid, rtk_port_t dstPort, rtk_vlan_t *pCvid)`

Get configure system SP2C content

Defined in: svlan.h

Parameters

svnid
SVLAN VID

dstPort
Destination port of SVLAN to CVLAN configuration

	<i>*pCvid</i>	VLAN ID
Comments	The API can get SVID & Destination Port to CVLAN configuration. There are 128 SP2C configurations.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_OUT_OF_RANGE	input out of range.
	RT_ERR_PORT_ID	Invalid port number.
	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.

24.25.rtk_svlan_sp2c_del

`int32 rtk_svlan_sp2c_del(rtk_vlan_t svid, rtk_port_t dstPort)`

Delete system SP2C configuration

Defined in: svlan.h

Parameters	<i>svid</i>	SVLAN VID
	<i>dstPort</i>	Destination port of SVLAN to CVLAN configuration
Comments	The API can delete SVID & Destination Port to CVLAN configuration. There are 128 SP2C configurations.	
Return Codes		
	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid port number.
	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
	RT_ERR_OUT_OF_RANGE	input out of range.

24.26.rtk_svlan_dmacVidSelState_set

`int32 rtk_svlan_dmacVidSelState_set(rtk_port_t port, rtk_enable_t enable)`

Set DMAC CVID selection status

Defined in: svlan.h

Parameters	<i>port</i> Port
	<i>enable</i> state of DMAC CVID Selection
Comments	This API can set DMAC CVID Selection state
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed
	RT_ERR_PORT_ID Invalid port id.
	RT_ERR_INPUT Invalid input parameters.

24.27.rtk_svlan_dmacVidSelState_get

```
int32 rtk_svlan_dmacVidSelState_get(rtk_port_t port, rtk_enable_t
*pEnable)
```

Get DMAC CVID selection status

Defined in: svlan.h

Parameters	<i>port</i> Port
	<i>*pEnable</i> state of DMAC CVID Selection
Comments	This API can get DMAC CVID Selection state
Return Codes	
	RT_ERR_OK ok
	RT_ERR_FAILED failed
	RT_ERR_PORT_ID Invalid port id.
	RT_ERR_INPUT Invalid input parameters.

24.28.rtk_svlan_unmatchAction_set

```
int32 rtk_svlan_unmatchAction_set(rtk_svlan_action_t action, rtk_vlan_t
svvid)
```

Configure Action of downstream Unmatch packet

	Defined in: svlan.h												
Parameters	<p><i>action</i> Action for Unmatch</p> <p><i>svid</i> The SVID assigned to Unmatch packet</p>												
Comments	The API can configure action of downstream Un-match packet. A SVID assigned to the un-match is also supported by this API. The parameter add svid is only referenced when the action is set to UNMATCH_ASSIGN												
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_SVLAN_VID</td> <td>Invalid SVLAN VID parameter.</td> </tr> <tr> <td>RT_ERR_SVLAN_ENTRY_NOT_FOU ND</td> <td>specified svlan entry not found.</td> </tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td> <td>input out of range.</td> </tr> <tr> <td>RT_ERR_INPUT</td> <td>Invalid input parameters.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.	RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.	RT_ERR_OUT_OF_RANGE	input out of range.	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok												
RT_ERR_FAILED	failed												
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.												
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.												
RT_ERR_OUT_OF_RANGE	input out of range.												
RT_ERR_INPUT	Invalid input parameters.												

24.29.rtk_svlan_unmatchAction_get

```
int32 rtk_svlan_unmatchAction_get(rtk_svlan_action_t *pAction, rtk_vlan_t  
*pSvid)
```

Get Action of downstream Unmatch packet

Defined in: svlan.h

Parameters	<p><i>*pAction</i> Action for Unmatch</p> <p><i>*pSvid</i> The SVID assigned to Unmatch packet</p>										
Comments	The API can Get action of downstream Un-match packet. A SVID assigned to the un-match is also retrieved by this API. The parameter pSvid is only referenced when the action is UNMATCH_ASSIGN										
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_SVLAN_VID</td> <td>Invalid SVLAN VID parameter.</td> </tr> <tr> <td>RT_ERR_SVLAN_ENTRY_NOT_FOU ND</td> <td>specified svlan entry not found.</td> </tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td> <td>input out of range.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.	RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.	RT_ERR_OUT_OF_RANGE	input out of range.
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.										
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.										
RT_ERR_OUT_OF_RANGE	input out of range.										

RT_ERR_INPUT	Invalid input parameters.
--------------	---------------------------

24.30.rtk_svlan_untagAction_set

int32 rtk_svlan_untagAction_set(rtk_svlan_action_t *action*, rtk_vlan_t *svid*)

Configure Action of downstream UnStag packet

Defined in: svlan.h

Parameters

action

Action for UnStag

svid

The SVID assigned to UnStag packet

Comments

The API can configure action of downstream Un-Stag packet. A SVID assigned to the un-stag is also supported by this API. The parameter of svid is only referenced when the action is set to UNTAG_ASSIGN

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
RT_ERR_SVLAN_ENTRY_NOT_FOUND	specified svlan entry not found.
RT_ERR_OUT_OF_RANGE	input out of range.
RT_ERR_INPUT	Invalid input parameters.

24.31.rtk_svlan_untagAction_get

int32 rtk_svlan_untagAction_get(rtk_svlan_action_t **pAction*, rtk_vlan_t **pSvid*)

Get Action of downstream UnStag packet

Defined in: svlan.h

Parameters

**pAction*

Action for UnStag

<i>*pSvid</i>	The SVID assigned to UnStag packet														
Comments	The API can Get action of downstream Un-Stag packet. A SVID assigned to the un-stag is also retrieved by this API. The parameter pSvid is only referenced when the action is UNTAG_ASSIGN														
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_SMI</td><td>SMI access error</td></tr> <tr> <td>RT_ERR_SVLAN_VID</td><td>Invalid SVLAN VID parameter.</td></tr> <tr> <td>RT_ERR_SVLAN_ENTRY_NOT_FOU ND</td><td>specified svlan entry not found.</td></tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td><td>input out of range.</td></tr> <tr> <td>RT_ERR_INPUT</td><td>Invalid input parameters.</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_SMI	SMI access error	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.	RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.	RT_ERR_OUT_OF_RANGE	input out of range.	RT_ERR_INPUT	Invalid input parameters.
RT_ERR_OK	ok														
RT_ERR_FAILED	failed														
RT_ERR_SMI	SMI access error														
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.														
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.														
RT_ERR_OUT_OF_RANGE	input out of range.														
RT_ERR_INPUT	Invalid input parameters.														

24.32.rtk_svlan_c2s_add

`int32 rtk_svlan_c2s_add(rtk_vlan_t cvid, rtk_port_t port, rtk_vlan_t svid)`

add CVID and ingress Port to SVLAN

Defined in: svlan.h

Parameters

cvid
CVLAN VID

port
port id

svid
SVLAN VID

Comments

The API can set upstream packet CVID and ingress port to SVID configuration. There are 128 SVLAN configurations for CVID and ingress port. If CVID and SVID of configured entry are matched with configuration parameter, then different ingress port will share the same configuration entry.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified svlan entry not found.

RT_ERR_OUT_OF_RANGE	input out of range.
---------------------	---------------------

24.33.rtk_svlan_c2s_del

int32 rtk_svlan_c2s_del(rtk_vlan_t cvid, rtk_port_t port, rtk_vlan_t svid)

delete CVID and ingress Port to SVLAN

Defined in: svlan.h

Parameters

cvid
CVLAN VID

port
port id

svid
SVLAN VID

Comments

The API can delet upstream packet CVID and ingress port to SVID configuration.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_SMI	SMI access error
RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
RT_ERR_SVLAN_ENTRY_NOT_FOUND	specified sylan entry not found.
RT_ERR_OUT_OF_RANGE	input out of range.

24.34.rtk_svlan_c2s_get

int32 rtk_svlan_c2s_get(rtk_vlan_t cvid, rtk_port_t port, rtk_vlan_t *pSvid)

Get CVID and ingress Port to SVLAN

Defined in: svlan.h

Parameters

cvid
CVLAN VID

port
port id

	<i>*pSvid</i>	SVLAN VID
Comments	The API can delet upstream packet CVID and ingress port to SVID configuration.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_SVLAN_VID	Invalid SVLAN VID parameter.
	RT_ERR_SVLAN_ENTRY_NOT_FOU ND	specified sylvan entry not found.
	RT_ERR_OUT_OF_RANGE	input out of range.

24.35.rtk_svlan_trapPri_get

`int32 rtk_svlan_trapPri_get(rtk_pri_t *pPriority)`

Get svlan trap priority

Defined in: svlan.h

Parameters	<i>*pPriority</i>	priority for trap packets
-------------------	-------------------	---------------------------

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	input parameter may be null pointer

24.36.rtk_svlan_trapPri_set

`int32 rtk_svlan_trapPri_set(rtk_pri_t priority)`

Set svlan trap priority

Defined in: svlan.h

Parameters	<i>priority</i>	priority for trap packets
-------------------	-----------------	---------------------------

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_QOS_INT_PRIORITY	input parameter may be null pointer

24.37.rtk_svlan_deiKeepState_get

`int32 rtk_svlan_deiKeepState_get(rtk_enable_t *pEnable)`

Get svlan dei keep state

Defined in: svlan.h

Parameters	<i>*pEnable</i>
	state of keep dei

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	input parameter may be null pointer

24.38.rtk_svlan_deiKeepState_set

`int32 rtk_svlan_deiKeepState_set(rtk_enable_t enable)`

Set svlan dei keep state

Defined in: svlan.h

Parameters	<i>enable</i>
	state of DMAC CVID Selection

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

24.39.rtk_svlan_lookupType_get

`int32 rtk_svlan_lookupType_get(rtk_svlan_lookupType_t *pType)`

Get lookup type of SVLAN

Defined in: svlan.h

Parameters

`*pType`
lookup type

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	Invalid input parameters.
<code>RT_ERR_CHIP_NOT_SUPPORTED</code>	Invalid SVLAN VID parameter.

24.40.rtk_svlan_lookupType_set

`int32 rtk_svlan_lookupType_set(rtk_svlan_lookupType_t type)`

Set lookup type of SVLAN

Defined in: svlan.h

Parameters

`type`
lookup type

Comments

Must call this API after rtl_svlan_init. Otherwise, there will be some unexpected switch behaviors. Set lookup type to SVLAN_LOOKUP_C4KVLAN and must create vlan by using API rtk_vlan_create. In the SVLAN_LOOKUP_C4KVLAN config, rtk_svlan_create and rtk_svlan_destroy will return failed.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_CHIP_NOT_SUPPORTED</code>	Invalid input parameters.

24.41.rtk_svlan_sp2cUnmatchCtagging_get

`int32 rtk_svlan_sp2cUnmatchCtagging_get(rtk_enable_t *pState)`

Get unmatch sp2c egress ctagging state

Defined in: svlan.h

Parameters `*pState`
unmatch cvlan tagging state

Comments None

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_NULL_POINTER</code>	Invalid input parameters.

24.42.rtk_svlan_sp2cUnmatchCtagging_set

`int32 rtk_svlan_sp2cUnmatchCtagging_set(rtk_enable_t state)`

Set unmatch sp2c egress ctagging state

Defined in: svlan.h

Parameters `state`
unmatch cvlan tagging state

Comments None

Return Codes	<code>RT_ERR_OK</code>	ok
	<code>RT_ERR_FAILED</code>	failed
	<code>RT_ERR_NULL_POINTER</code>	Invalid input parameters.

24.43.rtk_svlan_priority_get

`int32 rtk_svlan_priority_get(rtk_vlan_t svid, rtk_pri_t *pPriority)`

Get SVLAN priority for each SVID.

	Defined in: svlan.h												
Parameters	<p><i>svid</i> svlan id</p> <p><i>*pPriority</i> priority assigned for the SVID.</p>												
Comments	None												
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_SMI</td> <td>SMI access error</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>Invalid SVLAN VID parameter.</td> </tr> <tr> <td>RT_ERR_VLAN_VID</td> <td>specified svlan entry not found.</td> </tr> <tr> <td>RT_ERR_SVLAN_NOT_EXIST</td> <td>input out of range.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_SMI	SMI access error	RT_ERR_NULL_POINTER	Invalid SVLAN VID parameter.	RT_ERR_VLAN_VID	specified svlan entry not found.	RT_ERR_SVLAN_NOT_EXIST	input out of range.
RT_ERR_OK	ok												
RT_ERR_FAILED	failed												
RT_ERR_SMI	SMI access error												
RT_ERR_NULL_POINTER	Invalid SVLAN VID parameter.												
RT_ERR_VLAN_VID	specified svlan entry not found.												
RT_ERR_SVLAN_NOT_EXIST	input out of range.												

24.44.rtk_svlan_priority_set

```
int32 rtk_svlan_priority_set(rtk_vlan_t svid, rtk_pri_t priority)
```

Set SVLAN priority for each SVID.

Defined in: svlan.h

Parameters	<i>svid</i> svlan id										
	<i>priority</i> priority assigned for the SVID.										
Comments	This API is used to set priority per SVLAN.										
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_VLAN_PRIORITY</td> <td>Invalid priority.</td> </tr> <tr> <td>RT_ERR_VLAN_VID</td> <td>Invalid SVLAN VID parameter.</td> </tr> <tr> <td>RT_ERR_SVLAN_NOT_EXIST</td> <td>specified svlan entry not found.</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_VLAN_PRIORITY	Invalid priority.	RT_ERR_VLAN_VID	Invalid SVLAN VID parameter.	RT_ERR_SVLAN_NOT_EXIST	specified svlan entry not found.
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_VLAN_PRIORITY	Invalid priority.										
RT_ERR_VLAN_VID	Invalid SVLAN VID parameter.										
RT_ERR_SVLAN_NOT_EXIST	specified svlan entry not found.										

24.45.rtk_svlan_fid_get

```
int32 rtk_svlan_fid_get(rtk_vlan_t svid, rtk_fid_t *pFid)
```

	Get the filter id of the vlan.
	Defined in: svlan.h
Parameters	<p><i>svid</i> svlan id</p> <p><i>*pFid</i> pointer buffer of filter id</p>
Comments	None
Return Codes	<p>RT_ERR_OK ok</p> <p>RT_ERR_FAILED failed</p> <p>RT_ERR_NOT_INIT The module is not initial</p> <p>RT_ERR_VLAN_VID Invalid SVLAN VID parameter.</p> <p>RT_ERR_SVLAN_NOT_EXIST specified svlan entry not found.</p> <p>RT_ERR_NULL_POINTER input parameter may be null pointer</p>

24.46.rtk_svlan_fid_set

```
int32 rtk_svlan_fid_set(rtk_vlan_t svid, rtk_fid_t fid)
```

Set the filter id of the svlan.

Defined in: svlan.h

Parameters	<p><i>svid</i> svlan id</p> <p><i>fid</i> filter id</p>
Comments	The FID is effective only in VLAN SVL mode.
Return Codes	<p>RT_ERR_OK ok</p> <p>RT_ERR_FAILED failed</p> <p>RT_ERR_NOT_INIT The module is not initial</p> <p>RT_ERR_OUT_OF_RANGE input parameter out of range</p> <p>RT_ERR_VLAN_VID specified svlan entry not found.</p> <p>RT_ERR_SVLAN_NOT_EXIST input parameter may be null pointer</p>

24.47.rtk_svlan_fidEnable_get

`int32 rtk_svlan_fidEnable_get(rtk_vlan_t svvid, rtk_enable_t *pEnable)`

Get svlan based fid assignment status.

Defined in: svlan.h

Parameters

svvid
svlan id
**pEnable*
pointer to svlan based fid assignment status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	input parameter out of range
RT_ERR_SVLAN_NOT_EXIST	specified svlan entry not found.
RT_ERR_NULL_POINTER	input parameter may be null pointer

24.48.rtk_svlan_fidEnable_set

`int32 rtk_svlan_fidEnable_set(rtk_vlan_t svvid, rtk_enable_t enable)`

Set svlan based fid assignment status.

Defined in: svlan.h

Parameters

svvid
svlan id
enable
svlan based fid assignment status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_OUT_OF_RANGE	input parameter out of range

RT_ERR_VLAN_VID	specified svlan entry not found.
RT_ERR_SVLAN_NOT_EXIST	input parameter may be null pointer

24.49.rtk_svlan_enhancedFid_get

`int32 rtk_svlan_enhancedFid_get(rtk_vlan_t svid, rtk_efid_t *pEfid)`

Get the enhanced filter id of the vlan.

Defined in: svlan.h

Parameters

<i>svid</i>	svlan id
<i>*pEfid</i>	pointer buffer of enhanced filter id

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	input parameter out of range
RT_ERR_SVLAN_NOT_EXIST	specified svlan entry not found.
RT_ERR_NULL_POINTER	input parameter may be null pointer

24.50.rtk_svlan_enhancedFid_set

`int32 rtk_svlan_enhancedFid_set(rtk_vlan_t svid, rtk_efid_t efid)`

Set the enhanced filter id of the svlan.

Defined in: svlan.h

Parameters

<i>svid</i>	svlan id
<i>efid</i>	enhanced filter id

Comments

The EFID is effective only in VLAN SVL mode.

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_OUT_OF_RANGE	input parameter out of range
RT_ERR_VLAN_VID	specified svlan entry not found.
RT_ERR_SVLAN_NOT_EXIST	input parameter may be null pointer

24.51.rtk_svlan_enhancedFidEnable_get

```
int32 rtk_svlan_enhancedFidEnable_get(rtk_vlan_t svvid, rtk_enable_t *pEnable)
```

Get svlan based fid assignment status.

Defined in: svlan.h

Parameters

svvid
svlan id
**pEnable*
pointer to svlan based efid assignment status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	input parameter out of range
RT_ERR_SVLAN_NOT_EXIST	specified svlan entry not found.
RT_ERR_NULL_POINTER	input parameter may be null pointer

24.52.rtk_svlan_enhancedFidEnable_set

```
int32 rtk_svlan_enhancedFidEnable_set(rtk_vlan_t svvid, rtk_enable_t enable)
```

Set svlan based efid assignment status.

Defined in: svlan.h

Parameters

svvid
svlan id

	<i>enable</i>	svlan based efid assignment status
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_OUT_OF_RANGE RT_ERR_VLAN_VID RT_ERR_SVLAN_NOT_EXIST	ok failed The module is not initial input parameter out of range specified svlan entry not found. input parameter may be null pointer

24.53.rtk_svlan_dmacVidSelForcedState_set

`int32 rtk_svlan_dmacVidSelForcedState_set(rtk_enable_t enable)`

Set DMAC CVID selection status

Defined in: svlan.h

Parameters	<i>enable</i>	Port
-------------------	---------------	------

Comments This API can set DMAC CVID Selection forced state

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT	ok failed Invalid input parameters.
---------------------	--	---

24.54.rtk_svlan_dmacVidSelForcedState_get

`int32 rtk_svlan_dmacVidSelForcedState_get(rtk_enable_t *pEnable)`

Get DMAC CVID selection forced status

Defined in: svlan.h

Parameters	<i>*pEnable</i>	state of DMAC CVID Selection
-------------------	-----------------	------------------------------

Comments This API can get DMAC CVID Selection forced state

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	input parameter may be null pointer

25. Module Switch Global API

Filename: switch.h

Description	The file have include the following module and sub-modules (1) Switch parameter settings (2) Management address and vlan configuration.
--------------------	---

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

25.1. List of Symbols

Here is a list of all functions and variables in this module

```

switch.h - Definition of Switch Global API
rtk_switch_init
rtk_switch_deviceInfo_get
rtk_switch_phyPortId_get
rtk_switch_logicalPort_get
rtk_switch_port2PortMask_set
rtk_switch_port2PortMask_clear
rtk_switch_portIdInMask_check
rtk_switch_portMask_Clear
rtk_switch_allPortMask_set
rtk_switch_allExtPortMask_set
rtk_switch_nextPortInMask_get
rtk_switch_maxPktLenLinkSpeed_get
rtk_switch_maxPktLenLinkSpeed_set
rtk_switch_mgmtMacAddr_get
rtk_switch_mgmtMacAddr_set
rtk_switch_chip_reset

```

25.2. rtk_switch_init

`int32 rtk_switch_init(void)`

Initialize switch module of the specified device.

Defined in: switch.h

Parameters

`void`

Comments

Module must be initialized before using all of APIs in this module

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

25.3. rtk_switchDeviceInfo_get

`int32 rtk_switchDeviceInfo_get(rtk_switch_devInfo_t *pDevInfo)`

Get device information of the specific unit

Defined in: switch.h

Parameters

`*pDevInfo`

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

25.4. rtk_switch_phyPortId_get

`int32 rtk_switch_phyPortId_get(rtk_switch_port_name_t portName, int32 *pPortId)`

Get physical port id from logical port name

Defined in: switch.h

Parameters	<i>portName</i> logical port name
	<i>*pPortId</i> pointer to the physical port id
Comments	Call RTK API the port ID must get from this API
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER
	ok failed input parameter may be null pointer

25.5. rtk_switch_logicalPort_get

`int32 rtk_switch_logicalPort_get(int32 portId, rtk_switch_port_name_t *pPortName)`

Get logical port name from physical port id

Defined in: switch.h

Parameters	<i>portId</i> physical port id
	<i>*pPortName</i> pointer to logical port name

Comments Call RTK API the port ID must get from this API

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER	ok failed input parameter may be null pointer
---------------------	---	---

25.6. rtk_switch_port2PortMask_set

`int32 rtk_switch_port2PortMask_set(rtk_portmask_t *pPortMask,
rtk_switch_port_name_t portName)`

Set port id to the portlist

Defined in: switch.h

Parameters

<i>*pPortMask</i>	port mask
<i>portName</i>	logical port name
Comments	Call RTK API the port mask must set by this API
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

25.7. rtk_switch_port2PortMask_clear

```
int32 rtk_switch_port2PortMask_clear(rtk_portmask_t *pPortMask,
                                     rtk_switch_port_name_t portName)
```

Set port id to the portlist

Defined in: switch.h

Parameters	<i>*pPortMask</i>	port mask
	<i>portName</i>	logical port name
Comments	Call RTK API the port mask must set by this API	
Return Codes		
RT_ERR_OK	ok	
RT_ERR_FAILED	failed	
RT_ERR_NULL_POINTER	input parameter may be null pointer	

25.8. rtk_switch_portIdInMask_check

```
int32 rtk_switch_portIdInMask_check(rtk_portmask_t *pPortMask,
                                     rtk_switch_port_name_t portName)
```

Check if given port is in port list

Defined in: switch.h

Parameters

Parameters	<i>*pPortMask</i> port mask						
	<i>portName</i> logical port name						
Comments	Call RTK API the port mask must set by this API						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_NULL_POINTER	input parameter may be null pointer						

25.9. rtk_switch_portMask_Clear

`int32 rtk_switch_portMask_Clear(rt_k_portmask_t *pPortMask)`

Clear all port mask

Defined in: switch.h

Parameters	<i>*pPortMask</i> port mask						
Comments	Call RTK API the port mask must set by this API						
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_NULL_POINTER	input parameter may be null pointer						

25.10.rtk_switch_allPortMask_set

`int32 rtk_switch_allPortMask_set(rt_k_portmask_t *pPortMask)`

Set all switch port to mask

Defined in: switch.h

Parameters	<i>*pPortMask</i> port mask
Comments	Call RTK API the port mask must set by this API
Return Codes	RT_ERR_OK ok

RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

25.11.rtk_switch_allExtPortMask_set

`int32 rtk_switch_allExtPortMask_set(rtk_portmask_t *pPortMask)`

Set all extention port to mask

Defined in: switch.h

Parameters

`*pPortMask`
port mask

Comments

Call RTK API the port mask must set by this API

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

25.12.rtk_switch_nextPortInMask_get

`int32 rtk_switch_nextPortInMask_get(rtk_portmask_t *pPortMask, int32 *pPortId)`

Get next port id in the port mask

Defined in: switch.h

Parameters

`*pPortMask`
port mask
`*pPortId`
given port id to get the next port in the port mask

Comments

-RT_ERR_OK means the pPortId record the next port, others do not have any port for given port id next port -Set *pPortId to RTK_SWITCH_GET_FIRST_PORT to get the first port

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

25.13.rtk_switch_maxPktLenLinkSpeed_get

```
int32
rtk_switch_maxPktLenLinkSpeed_get(rtk_switch_maxPktLen_linkSpeed_t
speed, uint32 *pLen)
```

Get the max packet length setting of the specific speed type

Defined in: switch.h

Parameters

speed
speed type

**pLen*
pointer to the max packet length

Comments

-RT_ERR_OK means the pPortId record the next port, others do not have any port for given port id next port
-Set *pPortId to RTK_SWITCH_GET_FIRST_PORT to get the first port

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_INPUT	invalid enum speed type

25.14.rtk_switch_maxPktLenLinkSpeed_set

```
int32
rtk_switch_maxPktLenLinkSpeed_set(rtk_switch_maxPktLen_linkSpeed_t
speed, uint32 len)
```

Set the max packet length of the specific speed type

Defined in: switch.h

Parameters

speed
speed type

len
max packet length

Comments

-RT_ERR_OK means the pPortId record the next port, others do not have any port for given port id next port -Set *pPortId to RTK_SWITCH_GET_FIRST_PORT to get the first port

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	invalid enum speed type

25.15.rtk_switch_mgmtMacAddr_get

`int32 rtk_switch_mgmtMacAddr_get(rtk_mac_t *pMac)`

Get MAC address of switch.

Defined in: switch.h

Parameters	<code>*pMac</code>	pointer to MAC address
-------------------	--------------------	------------------------

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

25.16.rtk_switch_mgmtMacAddr_set

`int32 rtk_switch_mgmtMacAddr_set(rtk_mac_t *pMac)`

Set MAC address of switch.

Defined in: switch.h

Parameters	<code>*pMac</code>	MAC address
-------------------	--------------------	-------------

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

25.17.rtk_switch_chip_reset

`int32 rtk_switch_chip_reset(void)`

Reset switch chip

Defined in: switch.h

Parameters `void`

Comments None

Return Codes `RT_ERR_OK` ok

`RT_ERR_FAILED` failed

`RT_ERR_NOT_INIT` The module is not initial

`RT_ERR_NULL_POINTER` input parameter may be null pointer

26. Module TIME API

Filename: time.h

Description The file includes the following modules and sub-modules
 (1) IEEE 1588

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

26.1. List of Symbols

Here is a list of all functions and variables in this module

time.h - Definition of TIME API
`rtk_time_portTransparentEnable_set`
`rtk_time_portTransparentEnable_get`
`rtk_time_init`
`rtk_time_portPtpEnable_get`
`rtk_time_portPtpEnable_set`

```

rtk_time_curTime_get
rtk_time_curTime_latch
rtk_time_refTime_get
rtk_time_refTime_set
rtk_time_frequency_set
rtk_time_frequency_get
rtk_time_ptpIgrMsgAction_set
rtk_time_ptpIgrMsgAction_get
rtk_time_ptpEgrMsgAction_set
rtk_time_ptpEgrMsgAction_get
rtk_time_meanPathDelay_set
rtk_time_meanPathDelay_get
rtk_time_rxTime_set
rtk_time_rxTime_get

```

26.2. rtk_time_portTransparentEnable_set

`int32 rtk_time_portTransparentEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set transparent ports to the specified device.

Defined in: time.h

Parameters

<i>port</i>	ports
<i>enable</i>	enable status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

26.3. rtk_time_portTransparentEnable_get

`int32 rtk_time_portTransparentEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get transparent ports to the specified device.

	Defined in: time.h
Parameters	<p><i>port</i> ports</p> <p><i>*pEnable</i> enable status</p>
Comments	None
Return Codes	<p>RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER</p>
	ok failed Pointer enable point to NULL.

26.4. rtk_time_init

`int32 rtk_time_init(void)`

Initialize Time module.

Defined in: time.h

Parameters	<i>void</i>
Comments	Must initialize Time module before calling any Time APIs.
Return Codes	<p>RT_ERR_OK RT_ERR_FAILED</p>
	ok failed

26.5. rtk_time_portPtpEnable_get

`int32 rtk_time_portPtpEnable_get(rtk_port_t port, rtk_enable_t *pEnable)`

Get PTP status of the specified port.

Defined in: time.h

Parameters	<p><i>port</i> port id</p> <p><i>*pEnable</i> status</p>
-------------------	--

Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	invalid port id
RT_ERR_INPUT	Invalid input parameters.

26.6. rtk_time_portPtpEnable_set

`int32 rtk_time_portPtpEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set PTP status of the specified port.

Defined in: time.h

Parameters	
<i>port</i>	port id
<i>enable</i>	status
Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid port id.
RT_ERR_INPUT	Invalid input parameters.

26.7. rtk_time_curTime_get

`int32 rtk_time_curTime_get(rtk_time_timeStamp_t *pTimeStamp)`

Get the current time.

Defined in: time.h

Parameters	<i>*pTimeStamp</i> pointer buffer of the current time
Comments	None
Return Codes	

RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Pointer pTimeStamp point to NULL.

26.8. rtk_time_curTime_latch

`int32 rtk_time_curTime_latch(void)`

Latch the current time.

Defined in: time.h

Parameters `void`

Comments None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

26.9. rtk_time_refTime_get

`int32 rtk_time_refTime_get(uint32 *pSign, rtk_time_timeStamp_t *pTimeStamp)`

Get the reference time.

Defined in: time.h

Parameters

<code>*pSign</code>	pointer buffer of sign
<code>*pTimeStamp</code>	pointer buffer of the reference time

Comments None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Pointer pTimeStamp/pSign point to NULL.

26.10.rtk_time_refTime_set

`int32 rtk_time_refTime_set(uint32 sign, rtk_time_timeStamp_t timeStamp)`

Set the reference time.

Defined in: time.h

Parameters

sign
significant

timeStamp
reference timestamp value

Comments

sign=0 for positive adjustment, sign=1 for negative adjustment.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	invalid input parameter

26.11.rtk_time_frequency_set

`int32 rtk_time_frequency_set(uint32 freq)`

Set frequency of PTP system time.

Defined in: time.h

Parameters

freq
reference timestamp value

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_INPUT</code>	invalid input parameter

26.12.rtk_time_frequency_get

`int32 rtk_time_frequency_get(uint32 *freq)`

Set frequency of PTP system time.

Defined in: time.h

Parameters **freq*
 reference timestamp value

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	invalid input parameter

26.13.rtk_time_ptpIgrMsgAction_set

```
int32 rtk_time_ptpIgrMsgAction_set(rtk_time_ptpMsgType_t type,
                                    rtk_time_ptpIgrMsg_action_t igr_action)
```

Set ingress action configuration for PTP message.

Defined in: time.h

Parameters *type*
 PTP message type
igr_action
 ingress action.

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_ALLOWED	Invalid action.
	RT_ERR_INPUT	Invalid input parameters.

26.14.rtk_time_ptpIgrMsgAction_get

```
int32 rtk_time_ptpIgrMsgAction_get(rtk_time_ptpMsgType_t type,
                                    rtk_time_ptpIgrMsg_action_t *igr_action)
```

Get ingress action configuration for PTP message.

Defined in: time.h

Parameters	<i>type</i> PTP message type <i>*igr_action</i> ingress action.
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER

26.15.rtk_time_ptpEgrMsgAction_set

```
int32 rtk_time_ptpEgrMsgAction_set(rtk_time_ptpMsgType_t type,  
rtk_time_ptpEgrMsg_action_t egr_action)
```

Set egress action configuration for PTP message.

Defined in: time.h

Parameters	<i>type</i> PTP message type <i>egr_action</i> egress action.
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_ALLOWED RT_ERR_INPUT
	ok failed Invalid action. Invalid input parameters.

26.16.rtk_time_ptpEgrMsgAction_get

```
int32 rtk_time_ptpEgrMsgAction_get(rtk_time_ptpMsgType_t type,  
rtk_time_ptpEgrMsg_action_t *egr_action)
```

Get egress action configuration for PTP message.

Defined in: time.h

Parameters	<i>type</i> PTP message type <i>*egr_action</i> egress action.
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NULL_POINTER ok failed Pointer egr_action point to NULL.

26.17.rtk_time_meanPathDelay_set

`int32 rtk_time_meanPathDelay_set(uint32 delay)`

Set rtk_time_meanPathDelay_set of PTP system time.

Defined in: time.h

Parameters	<i>delay</i> mean path delay value
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT ok failed invalid input parameter

26.18.rtk_time_meanPathDelay_get

`int32 rtk_time_meanPathDelay_get(uint32 *delay)`

Get rtk_time_meanPathDelay_get of PTP system time.

Defined in: time.h

Parameters	<i>*delay</i> mean path delay.
Comments	None
Return Codes	RT_ERR_OK ok

RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	

26.19.rtk_time_rxTime_set

`int32 rtk_time_rxTime_set(rtk_time_timeStamp_t timeStamp)`

Set the RX time.

Defined in: time.h

Parameters

timeStamp
RX timestamp.

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

26.20.rtk_time_rxTime_get

`int32 rtk_time_rxTime_get(rtk_time_timeStamp_t *pTimeStamp)`

Get the RX time.

Defined in: time.h

Parameters

**pTimeStamp*
pointer buffer of the RX time.

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Pointer pTimeStamp point to NULL.

27. Module TRAP API

Filename: trap.h

Description

The file includes the following modules and sub-modules
(1) Configuration for trapping packet to CPU
(2) RMA
(3) User defined RMA
(4) System-wise management frame
(5) System-wise user defined management frame
(6) Per port user defined management frame
(7) Packet with special flag or option
(8) CFM and OAM packet

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

27.1. List of Symbols

Here is a list of all functions and variables in this module

trap.h - Definition of TRAP API
rtk_trap_init
rtk_trap_reasonTrapToCpuPriority_get
rtk_trap_reasonTrapToCpuPriority_set
rtk_trap_igmpCtrlPkt2CpuEnable_get
rtk_trap_igmpCtrlPkt2CpuEnable_set
rtk_trap_mldCtrlPkt2CpuEnable_get
rtk_trap_mldCtrlPkt2CpuEnable_set
rtk_trap_portIgmpMldCtrlPktAction_get
rtk_trap_portIgmpMldCtrlPktAction_set
rtk_trap_ipMcastPkt2CpuEnable_get
rtk_trap_ipMcastPkt2CpuEnable_set
rtk_trap_l2McastPkt2CpuEnable_get
rtk_trap_l2McastPkt2CpuEnable_set
rtk_trap_rmaAction_get
rtk_trap_rmaAction_set
rtk_trap_rmaPri_get
rtk_trap_rmaPri_set
rtk_trap_oamPduAction_get
rtk_trap_oamPduAction_set
rtk_trap_oamPduPri_get
rtk_trap_oamPduPri_set

27.2. rtk_trap_init

`int32 rtk_trap_init(void)`

Initial the trap module of the specified device..

Defined in: trap.h

Parameters

`void`

Comments

None

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

27.3. rtk_trap_reasonTrapToCpuPriority_get

`int32 rtk_trap_reasonTrapToCpuPriority_get(rtk_trap_reason_type_t type,
rtk_pri_t *pPriority)`

Get priority value of a packet that trapped to CPU port according to specific reason.

Defined in: trap.h

Parameters

`type`

reason that trap to CPU port.

`*pPriority`

configured internal priority for such reason.

Comments

Currently the trap reason that supported are listed as follows:

- TRAP_REASON_RMA
- TRAP_REASON_IPV4IGMP
- TRAP_REASON_IPV6MLD
- TRAP_REASON_1XEAPOL
- TRAP_REASON_VLANERR
- TRAP_REASON_SLPCHANGE
- TRAP_REASON_MULTICASTDLF
- TRAP_REASON_CFI
- TRAP_REASON_1XUNAUTH

Return Codes

`RT_ERR_OK`

ok

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	Invalid input parameter
RT_ERR_NULL_POINTER	NULL pointer

27.4. rtk_trap_reasonTrapToCpuPriority_set

`int32 rtk_trap_reasonTrapToCpuPriority_set(rtk_trap_reason_type_t type,
rtk_pri_t priority)`

Set priority value of a packet that trapped to CPU port according to specific reason.

Defined in: trap.h

Parameters

type
reason that trap to CPU port.

priority
internal priority that is going to be set for specific trap reason.

Comments

Currently the trap reason that supported are listed as follows:

- TRAP_REASON_RMA
- TRAP_REASON_IPV4IGMP
- TRAP_REASON_IPV6MLD
- TRAP_REASON_1XEAPOL
- TRAP_REASON_VLANERR
- TRAP_REASON_SLPCHANGE
- TRAP_REASON_MULTICASTDLF
- TRAP_REASON_CFI
- TRAP_REASON_1XUNAUTH

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_INPUT	Invalid input parameter

27.5. rtk_trap_igmpCtrlPkt2CpuEnable_get

`int32 rtk_trap_igmpCtrlPkt2CpuEnable_get(rtk_enable_t *pEnable)`

Get the configuration about whether IGMP control packets need be trapped to CPU.

Defined in: trap.h

Parameters

**pEnable*
status of IGMP control packet trap to CPU

Comments

The status of IGMP control packet trap to CPU:

- DISABLED
- ENABLED

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	Invalid unit id
RT_ERR_NULL_POINTER	NULL pointer

27.6. rtk_trap_igmpCtrlPkt2CpuEnable_set

`int32 rtk_trap_igmpCtrlPkt2CpuEnable_set(rtk_enable_t enable)`

Set the configuration about whether IGMP control packets need be trapped to CPU.

Defined in: trap.h

Parameters

enable
unit id

Comments

The status of IGMP control packet trap to CPU:

- DISABLED
- ENABLED

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	Invalid unit id
RT_ERR_INPUT	Invalid input parameter

27.7. rtk_trap_mldCtrlPkt2CpuEnable_get

`int32 rtk_trap_mldCtrlPkt2CpuEnable_get(rtk_enable_t *pEnable)`

Get the configuration about whether MLD control packets need be trapped to CPU.

Defined in: trap.h

Parameters	<i>*pEnable</i>
	status of MLD control packet trap to CPU
Comments	The status of MLD control packet trap to CPU: - DISABLED - ENABLED
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	Invalid unit id
RT_ERR_NULL_POINTER	NULL pointer

27.8. rtk_trap_mldCtrlPkt2CpuEnable_set

`int32 rtk_trap_mldCtrlPkt2CpuEnable_set(rtk_enable_t enable)`

Set the configuration about whether MLD control packets need be trapped to CPU.

Defined in: trap.h

Parameters	<i>enable</i>
	status of MLD control packet trap to CPU
Comments	The status of MLD control packet trap to CPU: - DISABLED - ENABLED
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	Invalid unit id
RT_ERR_INPUT	Invalid input parameter

27.9. rtk_trap_portIgmpMldCtrlPktAction_get

```
int32 rtk_trap_portIgmpMldCtrlPktAction_get(rtk_port_t port,
                                             rtk_trap_igmpMld_type_t igmpMldType, rtk_action_t *pAction)
```

Get the configuration about MLD control packets Action

Defined in: trap.h

Parameters

port

The ingress port ID.

igmpMldType

IGMP/MLD protocol type;

**pAction*

Action of IGMP/MLD control packet

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_PORT_ID	Invalid unit id
RT_ERR_INPUT	Invalid input parameter
RT_ERR_NULL_POINTER	NULL pointer

27.10.rtk_trap_portIgmpMldCtrlPktAction_set

```
int32 rtk_trap_portIgmpMldCtrlPktAction_set(rtk_port_t port,
                                             rtk_trap_igmpMld_type_t igmpMldType, rtk_action_t action)
```

Set the configuration about MLD control packets Action

Defined in: trap.h

Parameters

port

The ingress port ID.

igmpMldType

IGMP/MLD protocol type;

action

Action of IGMP/MLD control packet

Comments

None.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_PORT_ID	Invalid unit id
	RT_ERR_INPUT	Invalid input parameter

27.11.rtk_trap_ipMcastPkt2CpuEnable_get

`int32 rtk_trap_ipMcastPkt2CpuEnable_get(rtk_enable_t *pEnable)`

Get the configuration about whether IP multicast packet lookup miss need be trapped to CPU.

Defined in: trap.h

Parameters

**pEnable*
status of IP multicast packet trap to CPU

Comments

The status of IP multicast packet trap to CPU:

- DISABLED
- ENABLED

Return Codes

RT_ERR_OK	ok	
RT_ERR_FAILED	failed	
RT_ERR_UNIT_ID	Invalid unit id	
RT_ERR_NULL_POINTER	NULL pointer	

27.12.rtk_trap_ipMcastPkt2CpuEnable_set

`int32 rtk_trap_ipMcastPkt2CpuEnable_set(rtk_enable_t enable)`

Set the configuration about whether IP multicast packet lookup miss need be trapped to CPU.

Defined in: trap.h

Parameters

enable
status of IP multicast packet trap to CPU

Comments

The status of IP multicast packet trap to CPU:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_UNIT_ID	Invalid unit id
	RT_ERR_INPUT	Invalid input parameter

27.13.rtk_trap_l2McastPkt2CpuEnable_get

`int32 rtk_trap_l2McastPkt2CpuEnable_get(rtk_enable_t *pEnable)`

Get the configuration about whether L2 multicast packets lookup miss need be trapped to CPU.

Defined in: trap.h

Parameters

**pEnable*
status of L2 multicast packet trap to CPU

Comments

The status of L2 multicast packet trap to CPU:

- DISABLED
- ENABLED

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	Invalid unit id
RT_ERR_NULL_POINTER	NULL pointer

27.14.rtk_trap_l2McastPkt2CpuEnable_set

`int32 rtk_trap_l2McastPkt2CpuEnable_set(rtk_enable_t enable)`

Set the configuration about whether L2 multicast packets lookup miss need be trapped to CPU.

Defined in: trap.h

Parameters

enable
status of L2 multicast packet trap to CPU

Comments

The status of L2 multicast packet trap to CPU:

- DISABLED
- ENABLED

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_UNIT_ID	Invalid unit id
	RT_ERR_INPUT	Invalid input parameter

27.15.rtk_trap_rmaAction_get

`int32 rtk_trap_rmaAction_get(rtk_mac_t *pRmaFrame,
rtk_trap_rma_action_t *pRmaAction)`

Get action of reserved multicast address(RMA) frame.

Defined in: trap.h

Parameters	<i>*pRmaFrame</i>	Reserved multicast address.
	<i>*pRmaAction</i>	RMA action

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameter
	RT_ERR_NULL_POINTER	NULL pointer

27.16.rtk_trap_rmaAction_set

`int32 rtk_trap_rmaAction_set(rtk_mac_t *pRmaFrame,
rtk_trap_rma_action_t rmaAction)`

Set action of reserved multicast address(RMA) frame.

Defined in: trap.h

Parameters	<i>*pRmaFrame</i>	Reserved multicast address.
	<i>rmaAction</i>	RMA action

Comments	
-----------------	--

The supported Reserved Multicast Address frame: Assignment
Address RMA_BRG_GROUP (Bridge Group Address)
01-80-C2-00-00-00 RMA_FD_PAUSE (IEEE Std 802.3, 1988 Edition, Full
Duplex PAUSE operation) 01-80-C2-00-00-01 RMA_SP_MCAST (IEEE Std
802.3ad Slow Protocols-Multicast address) 01-80-C2-00-00-02
RMA_1X_PAE (IEEE Std 802.1X PAE address)
01-80-C2-00-00-03 RMA_RESERVED04 (Reserved)
01-80-C2-00-00-04 RMA_MEDIA_ACCESS_USE (Media Access Method
Specific Use) 01-80-C2-00-00-05 RMA_RESERVED06
(Reserved)
01-80-C2-00-00-06 RMA_RESERVED07 (Reserved)
01-80-C2-00-00-07 RMA_PVD_BRG_GROUP (Provider Bridge Group Address)
01-80-C2-00-00-08 RMA_RESERVED09 (Reserved)
01-80-C2-00-00-09 RMA_RESERVED0A (Reserved)
01-80-C2-00-00-0A RMA_RESERVED0B (Reserved)
01-80-C2-00-00-0B RMA_RESERVED0C (Reserved)
01-80-C2-00-00-0C RMA_MVRP (Provider Bridge MVRP Address)
01-80-C2-00-00-0D RMA_1ab_LL_DISCOVERY (802.1ab Link Layer Discover
Protocol Address) 01-80-C2-00-00-0E RMA_RESERVED0F
(Reserved)
01-80-C2-00-00-0F RMA_BRG_MNGEMENT (All LANs Bridge Management
Group Address) 01-80-C2-00-00-10
RMA_LOAD_SERV_GENERIC_ADDR (Load Server Generic Address)
01-80-C2-00-00-11 RMA_LOAD_DEV_GENERIC_ADDR (Loadable Device
Generic Address) 01-80-C2-00-00-12 RMA_RESERVED13
(Reserved)
01-80-C2-00-00-13 RMA_RESERVED14 (Reserved)
01-80-C2-00-00-14 RMA_RESERVED15 (Reserved)
01-80-C2-00-00-15 RMA_RESERVED16 (Reserved)
01-80-C2-00-00-16 RMA_RESERVED17 (Reserved)
01-80-C2-00-00-17 RMA_MANAGER_STA_GENERIC_ADDR (Generic
Address for All Manager Stations) 01-80-C2-00-00-18
RMA_RESERVED19 (Reserved)
01-80-C2-00-00-19 RMA_AGENT_STA_GENERIC_ADDR (Generic Address
for All Agent Stations) 01-80-C2-00-00-1A RMA_RESERVED1B
(Reserved)
01-80-C2-00-00-1B RMA_RESERVED1C (Reserved)
01-80-C2-00-00-1C RMA_RESERVED1D (Reserved)
01-80-C2-00-00-1D RMA_RESERVED1E (Reserved)
01-80-C2-00-00-1E RMA_RESERVED1F (Reserved)
01-80-C2-00-00-1F RMA_GMRP (GMRP Address)
01-80-C2-00-00-20 RMA_GVRP (GVRP address)
01-80-C2-00-00-21 RMA_UNDEF_GARP22~2F (Undefined GARP address)
01-80-C2-00-00-22 ~ 01-80-C2-00-00-2F CDP

01-00-0C-CC-CC-CC CDP
01-00-0C-CC-CC-CD

The supported Reserved Multicast Address action:

- RMA_ACTION_FORWARD
- RMA_ACTION_DROP
- RMA_ACTION_TRAP2CPU

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameter
	RT_ERR_RMA_ACTION	Invalid RMA action

27.17.rtk_trap_rmaPri_get

`int32 rtk_trap_rmaPri_get(rtk_pri_t *pPriority)`

Get priority of packets trapped to CPU.

Defined in: trap.h

Parameters	<code>*pPriority</code>
	pointer to priority

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

27.18.rtk_trap_rmaPri_set

`int32 rtk_trap_rmaPri_set(rtk_pri_t priority)`

Set priority of packets trapped to CPU.

Defined in: trap.h

Parameters	<code>priority</code>
	priority

Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PRIORITY	invalid priority value

27.19.rtk_trap_oamPduAction_get

`int32 rtk_trap_oamPduAction_get(rtk_action_t *pAction)`

Get forwarding action of trapped oam PDU on specified port.

Defined in: trap.h

Parameters	<i>*pAction</i>
	pointer to forwarding action of trapped oam PDU

Comments	Forwarding action is as following - ACTION_FORWARD - ACTION_TRAP2CPU
-----------------	--

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

27.20.rtk_trap_oamPduAction_set

`int32 rtk_trap_oamPduAction_set(rtk_action_t action)`

Set forwarding action of trapped oam PDU on specified port.

Defined in: trap.h

Parameters	<i>action</i>
	forwarding action of trapped oam PDU

Comments

Forwarding action is as following

- ACTION_FORWARD
- ACTION_TRAP2CPU

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_FWD_ACTION	invalid forwarding action

27.21.rtk_trap_oamPduPri_get

`int32 rtk_trap_oamPduPri_get(rtk_pri_t *pPriority)`

Get priority of trapped OAM PDU.

Defined in: trap.h

Parameters	<i>*pPriority</i>
	pointer to priority

Comments	None
-----------------	------

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

27.22.rtk_trap_oamPduPri_set

`int32 rtk_trap_oamPduPri_set(rtk_pri_t priority)`

Set priority of trapped OAM PDU.

Defined in: trap.h

Parameters	<i>priority</i>
	priority

Comments	None
-----------------	------

Return Codes	
---------------------	--

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PRIORITY	invalid priority value

28. Module TRUNK API

Filename: trunk.h

Description

The file includes the following modules and sub-modules

(1) User Configuration Trunk

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

28.1. List of Symbols

Here is a list of all functions and variables in this module

trunk.h - Definition of TRUNK API
 rtk_trunk_init
 rtk_trunk_distributionAlgorithm_get
 rtk_trunk_distributionAlgorithm_set
 rtk_trunk_port_get
 rtk_trunk_port_set
 rtk_trunk_hashMappingTable_get
 rtk_trunk_hashMappingTable_set
 rtk_trunk_mode_get
 rtk_trunk_mode_set
 rtk_trunk_trafficSeparate_get
 rtk_trunk_trafficSeparate_set
 rtk_trunk_portQueueEmpty_get
 rtk_trunk_trafficPause_get
 rtk_trunk_trafficPause_set

28.2. rtk_trunk_init

int32 rtk_trunk_init(void)

Initialize trunk module of the specified device.

	Defined in: trunk.h				
Parameters	<i>void</i>				
Comments	Must initialize trunk module before calling any trunk APIs.				
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

28.3. rtk_trunk_distributionAlgorithm_get

```
int32 rtk_trunk_distributionAlgorithm_get(uint32 trkGid, uint32
                                         *pAlgoBitmask)
```

Get the distribution algorithm of the trunk group id from the specified device.

Defined in: trunk.h

Parameters	<trkgid> trunk group id *pAlgoBitmask pointer buffer of bitmask of the distribution algorithm </trkgid>										
Comments	You can use OR operations in following bits to decide your algorithm. - TRUNK_DISTRIBUTION_ALGO_SPA_BIT (source port) - TRUNK_DISTRIBUTION_ALGO_SMAC_BIT (source mac) - TRUNK_DISTRIBUTION_ALGO_DMAC_BIT (destination mac) - TRUNK_DISTRIBUTION_ALGO_SIP_BIT (source ip) - TRUNK_DISTRIBUTION_ALGO_DIP_BIT (destination ip) - TRUNK_DISTRIBUTION_ALGO_SRC_L4PORT_BIT (source layer4 port) - TRUNK_DISTRIBUTION_ALGO_DST_L4PORT_BIT (destination layer4 port)										
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_UNIT_ID</td> <td>invalid unit id</td> </tr> <tr> <td>RT_ERR_LA_TRUNK_ID</td> <td>invalid trunk ID</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_UNIT_ID	invalid unit id	RT_ERR_LA_TRUNK_ID	invalid trunk ID	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok										
RT_ERR_FAILED	failed										
RT_ERR_UNIT_ID	invalid unit id										
RT_ERR_LA_TRUNK_ID	invalid trunk ID										
RT_ERR_NULL_POINTER	input parameter may be null pointer										

28.4. rtk_trunk_distributionAlgorithm_set

`int32 rtk_trunk_distributionAlgorithm_set(uint32 trkGid, uint32 algoBitmask)`

Set the distribution algorithm of the trunk group id from the specified device.

Defined in: trunk.h

Parameters

trkGid

trunk group id

algoBitmask

bitmask of the distribution algorithm

Comments

You can use OR operations in following bits to decide your algorithm.

- TRUNK_DISTRIBUTION_ALGO_SPA_BIT (source port)
- TRUNK_DISTRIBUTION_ALGO_SMAC_BIT (source mac)
- TRUNK_DISTRIBUTION_ALGO_DMAC_BIT (destination mac)
- TRUNK_DISTRIBUTION_ALGO_SIP_BIT (source ip)
- TRUNK_DISTRIBUTION_ALGO_DIP_BIT (destination ip)
- TRUNK_DISTRIBUTION_ALGO_SRC_L4PORT_BIT (source layer4 port)
- TRUNK_DISTRIBUTION_ALGO_DST_L4PORT_BIT (destination layer4 port)

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_LA_TRUNK_ID`

invalid trunk ID

`RT_ERR_LA_HASHMASK`

invalid hash mask

28.5. rtk_trunk_port_get

`int32 rtk_trunk_port_get(uint32 trkGid, rtk_portmask_t *pTrunkMemberPortmask)`

Get the members of the trunk id from the specified device.

Defined in: trunk.h

Parameters

trkGid

trunk group id

**pTrunkMemberPortmask*

pointer buffer of trunk member ports

Comments	None								
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_LA_TRUNK_ID</td> <td>invalid trunk ID</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_LA_TRUNK_ID	invalid trunk ID	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok								
RT_ERR_FAILED	failed								
RT_ERR_LA_TRUNK_ID	invalid trunk ID								
RT_ERR_NULL_POINTER	input parameter may be null pointer								

28.6. rtk_trunk_port_set

```
int32 rtk_trunk_port_set(uint32 trkGid, rtk_portmask_t
    *pTrunkMemberPortmask)
```

Set the members of the trunk id to the specified device.

Defined in: trunk.h

Parameters

trkGid
 trunk group id
**pTrunkMemberPortmask*
 trunk member ports

Comments

Return Codes

None	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_LA_TRUNK_ID	invalid trunk ID
RT_ERR_LA_MEMBER_OVERLAP	the specified port mask is overlapped with other group
RT_ERR_LA_PORTNUM_DUMB	it can only aggregate at most four ports when 802.1ad dumb mode
RT_ERR_LA_PORTNUM_NORMAL	it can only aggregate at most eight ports when 802.1ad normal mode

28.7. rtk_trunk_hashMappingTable_get

```
int32 rtk_trunk_hashMappingTable_get(uint32 trk_gid,
    rtk_trunk_hashVal2Port_t *pHash2Port_array)
```

Get hash value to port array in the trunk group id from the specified device.

Defined in: trunk.h

Parameters	<i>trk_gid</i> trunk group id <i>*pHash2Port_array</i> pointer buffer of ports associate with the hash value
-------------------	---

Comments	(1) The valid trk_gid is 0 in APOLLO
-----------------	--------------------------------------

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_UNIT_ID RT_ERR_LA_TRUNK_ID RT_ERR_NULL_POINTER	ok failed invalid unit id invalid trunk ID input parameter may be null pointer
---------------------	---	--

28.8. rtk_trunk_hashMappingTable_set

```
int32 rtk_trunk_hashMappingTable_set(uint32 trk_gid,
                                     rtk_trunk_hashVal2Port_t *pHash2Port_array)
```

Set hash value to port array in the trunk group id from the specified device.

Defined in: trunk.h

Parameters	<i>trk_gid</i> trunk group id <i>*pHash2Port_array</i> ports associate with the hash value
-------------------	---

Comments	(1) The valid trk_gid is 0 in APOLLO
-----------------	--------------------------------------

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_UNIT_ID RT_ERR_LA_TRUNK_ID RT_ERR_NULL_POINTER RT_ERR_LA_TRUNK_NOT_EXIST RT_ERR_LA_NOT_MEMBER_PORT RT_ERR_LA_CPUPORT	ok failed invalid unit id invalid trunk ID input parameter may be null pointer the trunk doesn't exist the port is not a member port of the trunk CPU port can not be aggregated port
---------------------	--	--

28.9. rtk_trunk_mode_get

`int32 rtk_trunk_mode_get(rtk_trunk_mode_t *pMode)`

Get the trunk mode from the specified device.

Defined in: trunk.h

Parameters

`*pMode`
pointer buffer of trunk mode

Comments

The enum of the trunk mode as following

- TRUNK_MODE_NORMAL
- TRUNK_MODE_DUMB

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_UNIT_ID</code>	invalid unit id
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

28.10.rtk_trunk_mode_set

`int32 rtk_trunk_mode_set(rtk_trunk_mode_t mode)`

Set the trunk mode to the specified device.

Defined in: trunk.h

Parameters

`mode`
trunk mode

Comments

The enum of the trunk mode as following

- TRUNK_MODE_NORMAL
- TRUNK_MODE_DUMB

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_UNIT_ID</code>	invalid unit id
<code>RT_ERR_INPUT</code>	invalid input parameter

28.11.rtk_trunk_trafficSeparate_get

```
int32 rtk_trunk_trafficSeparate_get(uint32 trk_gid,
rtk_trunk_separateType_t *pSeparateType)
```

Get the traffic separation setting of a trunk group from the specified device.

Defined in: trunk.h

Parameters

trk_gid

trunk group id

**pSeparateType*

pointer separated traffic type

Comments

SEPARATE_NONE: disable traffic separation

SEPARATE_KNOWN_MULTI_AND_FLOOD: trunk MSB link up port is dedicated to TX known multicast and flooding (L2 lookup miss) traffic

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	invalid unit id
RT_ERR_LA_TRUNK_ID	invalid trunk ID
RT_ERR_NULL_POINTER	input parameter may be null pointer

28.12.rtk_trunk_trafficSeparate_set

```
int32 rtk_trunk_trafficSeparate_set(uint32 trk_gid,
rtk_trunk_separateType_t separateType)
```

Set the traffic separation setting of a trunk group from the specified device.

Defined in: trunk.h

Parameters

trk_gid

trunk group id

separateType

traffic separation setting

Comments

SEPARATE_NONE: disable traffic separation

SEPARATE_KNOWN_MULTI_AND_FLOOD: trunk MSB link up port is dedicated to TX known multicast and flooding (L2 lookup miss) traffic

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_UNIT_ID	invalid unit id
RT_ERR_LA_TRUNK_ID	invalid trunk ID
RT_ERR_LA_HASHMASK	invalid hash mask

28.13.rtk_trunk_portQueueEmpty_get

`int32 rtk_trunk_portQueueEmpty_get(rtk_portmask_t *pEmpty_portmask)`

Get the port mask which all queues are empty.

Defined in: trunk.h

Parameters

`*pEmpty_portmask`
pointer empty port mask

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	input parameter may be null pointer

28.14.rtk_trunk_trafficPause_get

`int32 rtk_trunk_trafficPause_get(uint32 trk_gid, rtk_enable_t *pEnable)`

Get the traffic pause setting of a trunk group.

Defined in: trunk.h

Parameters

`trk_gid`
trunk group id
`*pEnable`
pointer of traffic pause state.

Comments

None.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_LA_TRUNK_ID	invalid trunk ID

RT_ERR_NULL_POINTER

input parameter may be null pointer

28.15.rtk_trunk_trafficPause_set

int32 rtk_trunk_trafficPause_set(uint32 trk_gid, rtk_enable_t enable)

Set the traffic separation setting of a trunk group from the specified device.

Defined in: trunk.h

Parameters

trk_gid

trunk group id

enable

traffic pause state

Comments

None.

Return Codes

RT_ERR_OK ok

RT_ERR_FAILED failed

RT_ERR_LA_TRUNK_ID invalid trunk ID

29. Module VLAN API

Filename: vlan.h

Description

The file includes the following modules and sub-modules

- (1) Vlan table configure and modification
- (2) Accept frame type
- (3) Vlan ingress/egress filter
- (4) Port based and protocol based vlan
- (5) TPID configuration
- (6) Ingress tag handling
- (7) Tag format handling

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

29.1. List of Symbols

Here is a list of all functions and variables in this module

vlan.h - Definition of VLAN API
rtk_vlan_init
rtk_vlan_create
rtk_vlan_destroy
rtk_vlan_destroyAll
rtk_vlan_fid_get
rtk_vlan_fid_set
rtk_vlan_fidMode_get
rtk_vlan_fidMode_set
rtk_vlan_port_get
rtk_vlan_port_set
rtk_vlan_extPort_get
rtk_vlan_extPort_set
rtk_vlan_stg_get
rtk_vlan_stg_set
rtk_vlan_priority_get
rtk_vlan_priority_set
rtk_vlan_priorityEnable_get
rtk_vlan_priorityEnable_set
rtk_vlan_policingEnable_get
rtk_vlan_policingEnable_set
rtk_vlan_policingMeterIdx_get
rtk_vlan_policingMeterIdx_set
rtk_vlan_portAcceptFrameType_get
rtk_vlan_portAcceptFrameType_set
rtk_vlan_vlanFunctionEnable_get
rtk_vlan_vlanFunctionEnable_set
rtk_vlan_portIgrFilterEnable_get
rtk_vlan_portIgrFilterEnable_set
rtk_vlan_leaky_get
rtk_vlan_leaky_set
rtk_vlan_portLeaky_get
rtk_vlan_portLeaky_set
rtk_vlan_keepType_get
rtk_vlan_keepType_set
rtk_vlan_portPvid_get
rtk_vlan_portPvid_set
rtk_vlan_extPortPvid_get
rtk_vlan_extPortPvid_set
rtk_vlan_protoGroup_get
rtk_vlan_protoGroup_set
rtk_vlan_portProtoVlan_get
rtk_vlan_portProtoVlan_set
rtk_vlan_tagMode_get
rtk_vlan_tagMode_set

```

rtk_vlan_portFid_get
rtk_vlan_portFid_set
rtk_vlan_portPriority_get
rtk_vlan_portPriority_set
rtk_vlan_portEgrTagKeepType_get
rtk_vlan_portEgrTagKeepType_set
rtk_vlan_transparentEnable_get
rtk_vlan_transparentEnable_set
rtk_vlan_cfiKeepEnable_get
rtk_vlan_cfiKeepEnable_set
rtk_vlan_reservedVidAction_get
rtk_vlan_reservedVidAction_set
rtk_vlan_tagModeIp4mc_get
rtk_vlan_tagModeIp4mc_set
rtk_vlan_tagModeIp6mc_get
rtk_vlan_tagModeIp6mc_set

```

29.2. rtk_vlan_init

int32 rtk_vlan_init(void)

Initialize vlan module.

Defined in: vlan.h

Parameters

void

Comments

Must initialize vlan module before calling any vlan APIs.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

29.3. rtk_vlan_create

int32 rtk_vlan_create(rtк_vlan_t vid)

Create the vlan in the specified device.

Defined in: vlan.h

Parameters

	<i>vid</i>	vlan id to be created
Comments	Must initialize vlan module before calling any vlan APIs.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_EXIST	vlan is exist

29.4. rtk_vlan_destroy

`int32 rtk_vlan_destroy(rt_k_vlan_t vid)`

Destroy the vlan.

Defined in: vlan.h

Parameters	<i>vid</i>	vlan id to be destroyed
Comments	None	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found

29.5. rtk_vlan_destroyAll

`int32 rtk_vlan_destroyAll(uint32 restoreDefaultVlan)`

Destroy all vlans except default vlan.

Defined in: vlan.h

Parameters	<i>restoreDefaultVlan</i>	keep and restore default vlan id or not?
-------------------	---------------------------	--

Comments	The restore argument is permit following value: - 0: remove default vlan - 1: restore default vlan	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_UNIT_ID	invalid unit id

29.6. rtk_vlan_fid_get		
int32 rtk_vlan_fid_get(rtk_vlan_t vid, rtk_fid_t *pFid)		
Get the filter id of the vlan.		
Defined in: vlan.h		
Parameters		
<i>vid</i>	vlan id	
<i>*pFid</i>	pointer buffer of filter id	
Comments	(1) In IVL mode, vid is equal with fid after vlan create. (2) You don't need to care fid when you use the IVL mode. (3) The API should be used for SVL mode.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.7. rtk_vlan_fid_set

int32 rtk_vlan_fid_set(rtk_vlan_t vid, rtk_fid_t fid)

Set the filter id of the vlan.

	Defined in: vlan.h	
Parameters	<i>vid</i>	vlan id
	<i>fid</i>	filter id
Comments	(1) In IVL mode, vid is equal with fid after vlan create. (2) You don't need to care fid when you use the IVL mode. (3) The API should be used for SVL mode.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
	RT_ERR_OUT_OF_RANGE	input parameter out of range

29.8. rtk_vlan_fidMode_get

`int32 rtk_vlan_fidMode_get(rtk_vlan_t vid, rtk_fidMode_t *pMode)`

Get the filter id mode of the vlan.

Defined in: vlan.h

Parameters	<i>vid</i>	vlan id
	<i>*pMode</i>	pointer buffer of filter id mode
Comments	mode can be: -VLAN_FID_IVL -VLAN_FID_SVL	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.9. rtk_vlan_fidMode_set

`int32 rtk_vlan_fidMode_set(rtk_vlan_t vid, rtk_fidMode_t mode)`

Set the filter id mode of the vlan.

Defined in: vlan.h

Parameters

vid
 vlan id

mode
 filter id mode

Comments

mode can be: -VLAN_FID_IVL -VLAN_FID_SVL

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vid
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
D	
RT_ERR_OUT_OF_RANGE	input parameter out of range

29.10.rtk_vlan_port_get

`int32 rtk_vlan_port_get(rtk_vlan_t vid, rtk_portmask_t *pMemberPortmask,
 rtk_portmask_t *pUntagPortmask)`

Get the vlan members.

Defined in: vlan.h

Parameters

vid
 vlan id

**pMemberPortmask*
 pointer buffer of member ports

**pUntagPortmask*
 pointer buffer of untagged member ports

Comments

None

Return Codes

RT_ERR_OK	ok
-----------	----

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vid
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.11.rtk_vlan_port_set

`int32 rtk_vlan_port_set(rtk_vlan_t vid, rtk_portmask_t *pMember_portmask, rtk_portmask_t *pUntag_portmask)`

Replace the vlan members.

Defined in: `vlan.h`

Parameters

vid

vlan id

**pMember_portmask*

member ports

**pUntag_portmask*

untagged member ports

Comments

Don't care the original vlan members and replace with new configure directly. If users specify an empty extension portmask and CPU port is set in pMember_portmask, the packets will be restricted to be forwarded to CPU. Likewise, If users specify an non-empty extension portmask and CPU port is not set in pMember_portmask, the packets will be restricted to be forwarded to CPU. too.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vid
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.12.rtk_vlan_extPort_get

`int32 rtk_vlan_extPort_get(rtk_vlan_t vid, rtk_portmask_t *pExt_portmask)`

Get the vlan extension members.

Defined in: vlan.h

Parameters

vid
 vlan id
**pExt_portmask*
 pointer buffer of extension member ports

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_VLAN_VID</code>	invalid vid
<code>RT_ERR_VLAN_ENTRY_NOT_FOUND</code>	specified vlan entry not found
<code>D</code>	
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

29.13.rtk_vlan_extPort_set

`int32 rtk_vlan_extPort_set(rtk_vlan_t vid, rtk_portmask_t *pExt_portmask)`

Replace the vlan extension members.

Defined in: vlan.h

Parameters

vid
 vlan id
**pExt_portmask*
 extension member ports

Comments

Don't care the original vlan members and replace with new configure directly. If users specify an empty extension portmask and CPU port is set in pMember_portmask, the packets will be restricted to be forwarded to CPU. Likewise, If users specify an non-empty extension portmask and CPU port is not set in pMember_portmask, the packets will be restricted to be forwarded to CPU. too.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_VLAN_VID	invalid vid
	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.14.rtk_vlan_stg_get

`int32 rtk_vlan_stg_get(rtk_vlan_t vid, rtk_stg_t *pStg)`

Get spanning tree group instance of the vlan from the specified device.

Defined in: `vlan.h`

Parameters

<i>vid</i>	vlan id
<i>*pStg</i>	pointer buffer of spanning tree group instance

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vid
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.15.rtk_vlan_stg_set

`int32 rtk_vlan_stg_set(rtk_vlan_t vid, rtk_stg_t stg)`

Set spanning tree group instance of the vlan.

Defined in: `vlan.h`

Parameters

<i>vid</i>	vlan id
<i>stg</i>	spanning tree group instance
Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_VLAN_VID	invalid vid
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found
RT_ERR_MSTI	invalid msti

29.16.rtk_vlan_priority_get

```
int32 rtk_vlan_priority_get(rtk_vlan_t vid, rtk_pri_t *pPriority)
```

Get VLAN priority for each CVLAN.

Defined in: wlan.h

Parameters	<i>vid</i>
	vlan id
	<i>*pPriority</i>

802.1p priority for the PVID.

Comments	This API is used to set priority per VLAN.
-----------------	--

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_SMI	SMI access error
	RT_ERR_VLAN_VID	Invalid VID parameter.
	RT_ERR_PORT_ID	Invalid port number.

29.17.rtk_vlan_priority_set

```
int32 rtk_vlan_priority_set(rtk_vlan_t vid, rtk_pri_t priority)
```

Set VLAN priority for each CVLAN.

Defined in: vlan.h

Parameters

vid
 vlan id

priority
 802.1p priority for the PVID.

Comments

This API is used to set priority per VLAN.

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_VLAN_VID	Invalid VID parameter.
RT_ERR_VLAN_PRIORITY	Invalid priority.

29.18.rtk_vlan_priorityEnable_get

`int32 rtk_vlan_priorityEnable_get(rtk_vlan_t vid, rtk_enable_t *pEnable)`

Get vlan based priority assignment status.

Defined in: vlan.h

Parameters

vid
 vlan id

**pEnable*
 pointer to vlan based priority assignment status

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.19.rtk_vlan_priorityEnable_set

`int32 rtk_vlan_priorityEnable_set(rtk_vlan_t vid, rtk_enable_t enable)`

Set vlan based priority assignment status.

	Defined in: vlan.h						
Parameters	<p><i>vid</i> vlan id</p> <p><i>enable</i> vlan based priority assignment status</p>						
Comments	None						
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial
RT_ERR_OK	ok						
RT_ERR_FAILED	failed						
RT_ERR_NOT_INIT	The module is not initial						

29.20.rtk_vlan_policingEnable_get

```
int32 rtk_vlan_policingEnable_get(rtk_vlan_t vid, rtk_enable_t *pEnable)
```

Get the policing mode of the vlan.

Defined in: vlan.h

Parameters	<p><i>vid</i> vlan id</p> <p><i>*pEnable</i> pointer buffer of filter id mode</p>												
Comments	None												
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_FAILED</td><td>failed</td></tr> <tr> <td>RT_ERR_NOT_INIT</td><td>The module is not initial</td></tr> <tr> <td>RT_ERR_VLAN_VID</td><td>invalid vid</td></tr> <tr> <td>RT_ERR_VLAN_ENTRY_NOT_FOUND</td><td>specified vlan entry not found</td></tr> <tr> <td>RT_ERR_NULL_POINTER</td><td>input parameter may be null pointer</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_VLAN_VID	invalid vid	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok												
RT_ERR_FAILED	failed												
RT_ERR_NOT_INIT	The module is not initial												
RT_ERR_VLAN_VID	invalid vid												
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found												
RT_ERR_NULL_POINTER	input parameter may be null pointer												

29.21.rtk_vlan_policingEnable_set

```
int32 rtk_vlan_policingEnable_set(rtk_vlan_t vid, rtk_enable_t enable)
```

	Set the policing mode of the vlan.													
	Defined in: vlan.h													
Parameters	<p><i>vid</i> vlan id</p> <p><i>enable</i> State of Policing.</p>													
Comments	None													
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_VLAN_VID</td> <td>invalid vid</td> </tr> <tr> <td>RT_ERR_VLAN_ENTRY_NOT_FOUND</td> <td>specified vlan entry not found</td> </tr> <tr> <td>RT_ERR_OUT_OF_RANGE</td> <td>input parameter out of range</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_VLAN_VID	invalid vid	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found	RT_ERR_OUT_OF_RANGE	input parameter out of range
RT_ERR_OK	ok													
RT_ERR_FAILED	failed													
RT_ERR_NOT_INIT	The module is not initial													
RT_ERR_VLAN_VID	invalid vid													
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found													
RT_ERR_OUT_OF_RANGE	input parameter out of range													

29.22.rtk_vlan_policingMeterIdx_get

```
int32 rtk_vlan_policingMeterIdx_get(rtk_vlan_t vid, uint32 *pIndex)
```

Get the policing mode of the vlan.

Defined in: vlan.h

Parameters	<i>vid</i> vlan id													
	<i>*pIndex</i> pointer of meter index													
Comments	None													
Return Codes	<table border="0"> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> <tr> <td>RT_ERR_NOT_INIT</td> <td>The module is not initial</td> </tr> <tr> <td>RT_ERR_VLAN_VID</td> <td>invalid vid</td> </tr> <tr> <td>RT_ERR_VLAN_ENTRY_NOT_FOUND</td> <td>specified vlan entry not found</td> </tr> <tr> <td>RT_ERR_NULL_POINTER</td> <td>input parameter may be null pointer</td> </tr> </table>		RT_ERR_OK	ok	RT_ERR_FAILED	failed	RT_ERR_NOT_INIT	The module is not initial	RT_ERR_VLAN_VID	invalid vid	RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found	RT_ERR_NULL_POINTER	input parameter may be null pointer
RT_ERR_OK	ok													
RT_ERR_FAILED	failed													
RT_ERR_NOT_INIT	The module is not initial													
RT_ERR_VLAN_VID	invalid vid													
RT_ERR_VLAN_ENTRY_NOT_FOUND	specified vlan entry not found													
RT_ERR_NULL_POINTER	input parameter may be null pointer													

29.23.rtk_vlan_policingMeterIdx_set

`int32 rtk_vlan_policingMeterIdx_set(rtk_vlan_t vid, uint32 index)`

Set the policing mode of the vlan.

Defined in: `vlan.h`

Parameters

vid
 vlan id

index
 Meter index.

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_VLAN_VID</code>	invalid vid
<code>RT_ERR_VLAN_ENTRY_NOT_FOUND</code>	specified vlan entry not found
<code>D</code>	
<code>RT_ERR_OUT_OF_RANGE</code>	input parameter out of range

29.24.rtk_vlan_portAcceptFrameType_get

`int32 rtk_vlan_portAcceptFrameType_get(rtk_port_t port,
rtk_vlan_acceptFrameType_t *pAcceptFrameType)`

Get vlan accept frame type of the port.

Defined in: `vlan.h`

Parameters

port
 port id
**pAcceptFrameType*
 pointer buffer of accept frame type

Comments

The accept frame type as following:

- `ACCEPT_FRAME_TYPE_ALL`
- `ACCEPT_FRAME_TYPE_TAG_ONLY`
- `ACCEPT_FRAME_TYPE_UNTAG_ONLY`

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.25.rtk_vlan_portAcceptFrameType_set

`int32 rtk_vlan_portAcceptFrameType_set(rtk_port_t port,
rtk_vlan_acceptFrameType_t acceptFrameType)`

Set vlan accept frame type of the port.

Defined in: vlan.h

Parameters

port

port id

acceptFrameType

accept frame type

Comments

The accept frame type as following:

- ACCEPT_FRAME_TYPE_ALL
- ACCEPT_FRAME_TYPE_TAG_ONLY
- ACCEPT_FRAME_TYPE_UNTAG_ONLY
- ACCEPT_FRAME_TYPE_1P_1Q_TAG_ONLY

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_VLAN_ACCEPT_FRAME_TYPE	invalid accept frame type
RT_ERR_CHIP_NOT_SUPPORTED	functions not supported by this chip model

29.26.rtk_vlan_vlanFunctionEnable_get

`int32 rtk_vlan_vlanFunctionEnable_get(rtk_enable_t *pEnable)`

Get the VLAN enable status.

	Defined in: vlan.h	
Parameters	* <i>pEnable</i> enable status	
Comments	The status of vlan function is as following: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT	ok failed The module is not initial

29.27.rtk_vlan_vlanFunctionEnable_set

```
int32 rtk_vlan_vlanFunctionEnable_set(rtk_enable_t enable)
```

Set the VLAN enable status.

Defined in: vlan.h

Parameters	<i>enable</i> enable status	
Comments	The status of vlan function is as following: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT	ok failed The module is not initial

29.28.rtk_vlan_portIgrFilterEnable_get

```
int32 rtk_vlan_portIgrFilterEnable_get(rtk_port_t port, rtk_enable_t
*pEnable)
```

Get vlan ingress filter status of the port.

Defined in: vlan.h

Parameters

	<i>port</i>	port id
	<i>*pEnable</i>	pointer buffer of ingress filter status
Comments	The status of vlan function is as following: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_NULL_POINTER	
	ok failed The module is not initial invalid port id input parameter may be null pointer	

29.29.rtk_vlan_portIgrFilterEnable_set

`int32 rtk_vlan_portIgrFilterEnable_set(rtk_port_t port, rtk_enable_t enable)`

Set vlan ingress filter status of the port to the specified device.

Defined in: `vlan.h`

Parameters	<i>port</i>	port id
	<i>enable</i>	ingress filter configure
Comments	The status of vlan ingress filter is as following: - DISABLED - ENABLED	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_INPUT	
	ok failed The module is not initial invalid port id invalid input parameter	

29.30.rtk_vlan_leaky_get

`int32 rtk_vlan_leaky_get(rtk_leaky_type_t leakyType, rtk_enable_t *pEnable)`

Get VLAN leaky.

Defined in: wlan.h

Parameters

leakyType
Packet type for VLAN leaky.

**pEnable*
Leaky status.

Comments

This API can get VLAN leaky status for RMA and IGMP/MLD packets. The leaky frame types are as following:

- LEAKY_BRG_GROUP,
- LEAKY_FD_PAUSE,
- LEAKY_SP_MCAST,
- LEAKY_1X_PAE,
- LEAKY_UNDEF_BRG_04,
- LEAKY_UNDEF_BRG_05,
- LEAKY_UNDEF_BRG_06,
- LEAKY_UNDEF_BRG_07,
- LEAKY_PROVIDER_BRIDGE_GROUP_ADDRESS,
- LEAKY_UNDEF_BRG_09,
- LEAKY_UNDEF_BRG_0A,
- LEAKY_UNDEF_BRG_0B,
- LEAKY_UNDEF_BRG_0C,
- LEAKY_PROVIDER_BRIDGE_GVRP_ADDRESS,
- LEAKY_8021AB,
- LEAKY_UNDEF_BRG_0F,
- LEAKY_BRG_MNGEMENT,
- LEAKY_UNDEFINED_11,
- LEAKY_UNDEFINED_12,
- LEAKY_UNDEFINED_13,
- LEAKY_UNDEFINED_14,
- LEAKY_UNDEFINED_15,
- LEAKY_UNDEFINED_16,
- LEAKY_UNDEFINED_17,
- LEAKY_UNDEFINED_18,
- LEAKY_UNDEFINED_19,
- LEAKY_UNDEFINED_1A,
- LEAKY_UNDEFINED_1B,
- LEAKY_UNDEFINED_1C,
- LEAKY_UNDEFINED_1D,

- LEAKY_UNDEFINED_1E,
- LEAKY_UNDEFINED_1F,
- LEAKY_GMRP,
- LEAKY_GVRP,
- LEAKY_UNDEF_GARP_22,
- LEAKY_UNDEF_GARP_23,
- LEAKY_UNDEF_GARP_24,
- LEAKY_UNDEF_GARP_25,
- LEAKY_UNDEF_GARP_26,
- LEAKY_UNDEF_GARP_27,
- LEAKY_UNDEF_GARP_28,
- LEAKY_UNDEF_GARP_29,
- LEAKY_UNDEF_GARP_2A,
- LEAKY_UNDEF_GARP_2B,
- LEAKY_UNDEF_GARP_2C,
- LEAKY_UNDEF_GARP_2D,
- LEAKY_UNDEF_GARP_2E,
- LEAKY_UNDEF_GARP_2F,
- LEAKY_IGMP,
- LEAKY_CDP,
- LEAKY_SSTP,

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.

29.31.rtk_vlan_leaky_set

`int32 rtk_vlan_leaky_set(rtk_leaky_type_t leakyType, rtk_enable_t enable)`

Set VLAN leaky.

Defined in: vlan.h

Parameters	<i>leakyType</i> Packet type for VLAN leaky.
	<i>enable</i> Leaky status.

Comments	This API can set VLAN leaky for RMA and IGMP/MLD packets. The leaky frame types are as following: <ul style="list-style-type: none"> - LEAKY_BRG_GROUP, - LEAKY_FD_PAUSE,
-----------------	--

- LEAKY_SP_MCAST,
- LEAKY_1X_PAE,
- LEAKY_UNDEF_BRG_04,
- LEAKY_UNDEF_BRG_05,
- LEAKY_UNDEF_BRG_06,
- LEAKY_UNDEF_BRG_07,
- LEAKY_PROVIDER_BRIDGE_GROUP_ADDRESS,
- LEAKY_UNDEF_BRG_09,
- LEAKY_UNDEF_BRG_0A,
- LEAKY_UNDEF_BRG_0B,
- LEAKY_UNDEF_BRG_0C,
- LEAKY_PROVIDER_BRIDGE_GVRP_ADDRESS,
- LEAKY_8021AB,
- LEAKY_UNDEF_BRG_0F,
- LEAKY_BRG_MNGEMENT,
- LEAKY_UNDEFINED_11,
- LEAKY_UNDEFINED_12,
- LEAKY_UNDEFINED_13,
- LEAKY_UNDEFINED_14,
- LEAKY_UNDEFINED_15,
- LEAKY_UNDEFINED_16,
- LEAKY_UNDEFINED_17,
- LEAKY_UNDEFINED_18,
- LEAKY_UNDEFINED_19,
- LEAKY_UNDEFINED_1A,
- LEAKY_UNDEFINED_1B,
- LEAKY_UNDEFINED_1C,
- LEAKY_UNDEFINED_1D,
- LEAKY_UNDEFINED_1E,
- LEAKY_UNDEFINED_1F,
- LEAKY_GMRP,
- LEAKY_GVRP,
- LEAKY_UNDEF_GARP_22,
- LEAKY_UNDEF_GARP_23,
- LEAKY_UNDEF_GARP_24,
- LEAKY_UNDEF_GARP_25,
- LEAKY_UNDEF_GARP_26,
- LEAKY_UNDEF_GARP_27,
- LEAKY_UNDEF_GARP_28,
- LEAKY_UNDEF_GARP_29,
- LEAKY_UNDEF_GARP_2A,
- LEAKY_UNDEF_GARP_2B,
- LEAKY_UNDEF_GARP_2C,
- LEAKY_UNDEF_GARP_2D,
- LEAKY_UNDEF_GARP_2E,

- LEAKY_UNDEF_GARP_2F,
- LEAKY_IGMP,
- LEAKY_CDP,
- LEAKY_SSTP,

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_ENABLE	Invalid enable input

29.32.rtk_vlan_portLeaky_get

```
int32 rtk_vlan_portLeaky_get(rtk_port_t port, rtk_leaky_type_t leakyType,
                           rtk_enable_t *pEnable)
```

Get VLAN port-based leaky.

Defined in: vlan.h

Parameters	<i>port</i>	port ID
	<i>leakyType</i>	Packet type for VLAN leaky.
	<i>*pEnable</i>	Leaky status.

Comments	This API can set VLAN leaky for RMA and IGMP/MLD packets. The leaky frame types are as following:	
	- LEAKY_IPMULTICAST	

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_ENABLE	Invalid enable input
	RT_ERR_NULL_POINTER	NULL Pointer

29.33.rtk_vlan_portLeaky_set

```
int32 rtk_vlan_portLeaky_set(rtk_port_t port, rtk_leaky_type_t leakyType,
                             rtk_enable_t enable)
```

Set VLAN port-based leaky.

Defined in: vlan.h

Parameters

port
port ID

leakyType
Packet type for VLAN leaky.

enable
Leaky status.

Comments

This API can set VLAN leaky for RMA and IGMP/MLD packets. The leaky frame types are as following:
- LEAKY_IPMULTICAST

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.
RT_ERR_ENABLE	Invalid enable input

29.34.rtk_vlan_keepType_get

```
int32 rtk_vlan_keepType_get(rtk_vlan_keep_type_t keepType, rtk_enable_t
                           *pEnable)
```

Get VLAN keep format setting.

Defined in: vlan.h

Parameters

keepType
Packet type for VLAN keep format.

**pEnable*
Leaky status.

Comments

This API can set VLAN leaky for RMA and IGMP/MLD packets. The leaky frame types are as following:
- LEAKY_IPMULTICAST

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameters.
	RT_ERR_NULL_POINTER	NULL pointer

29.35.rtk_vlan_keepType_set

`int32 rtk_vlan_keepType_set(rtk_vlan_keep_type_t keepType, rtk_enable_t enable)`

Set VLAN keep format setting.

Defined in: `vlan.h`

Parameters

keepType

Packet type for VLAN keep format.

enable

Leaky status.

Comments

This API can set VLAN leaky for RMA and IGMP/MLD packets. The leaky frame types are as following:

- LEAKY_IPMULTICAST

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

Invalid input parameters.

29.36.rtk_vlan_portPvid_get

`int32 rtk_vlan_portPvid_get(rtk_port_t port, uint32 *pPvid)`

Get port default vlan id.

Defined in: `vlan.h`

Parameters

port

port id

**pPvid*

pointer buffer of port default vlan id

Comments

None

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_NULL_POINTER	ok failed The module is not initial invalid port id input parameter may be null pointer
---------------------	--	---

29.37.rtk_vlan_portPvid_set

`int32 rtk_vlan_portPvid_set(rtk_port_t port, uint32 pvid)`

Set port default vlan id.

Defined in: `vlan.h`

Parameters

port
port id
pvid
port default vlan id

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_VLAN_VID	invalid vid

29.38.rtk_vlan_extPortPvid_get

`int32 rtk_vlan_extPortPvid_get(uint32 extPort, uint32 *pPvid)`

Get extension port default vlan id.

Defined in: `vlan.h`

Parameters

extPort
Extension port id
**pPvid*
pointer buffer of port default vlan id

Comments	Configuration on EXT port 0 will be get from physical port 6.	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.39.rtk_vlan_extPortPvid_set

`int32 rtk_vlan_extPortPvid_set(uint32 extPort, uint32 pvid)`

Set extension port default vlan id.

Defined in: `vlan.h`

Parameters	<i>extPort</i>	extension port id
	<i>pvid</i>	extension port default vlan id
Comments		
Configuration on EXT port 0 will be applied to physical port 6.		

Comments

Configuration on EXT port 0 will be applied to physical port 6.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_VLAN_VID	invalid vid

29.40.rtk_vlan_protoGroup_get

`int32 rtk_vlan_protoGroup_get(uint32 protoGroupIdx,
rtk_vlan_protoGroup_t *pProtoGroup)`

Get protocol group for protocol based vlan.

Defined in: `vlan.h`

Parameters	<i>protoGroupIdx</i>	protocol group index
-------------------	----------------------	----------------------

	<i>*pProtoGroup</i>	pointer to protocol group
Comments	None	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_OUT_OF_RANGE RT_ERR_NULL_POINTER	ok failed The module is not initial protocol group index is out of range input parameter may be null pointer

29.41.rtk_vlan_protoGroup_set

```
int32 rtk_vlan_protoGroup_set(uint32 protoGroupIdx,  
                           rtk_vlan_protoGroup_t *pProtoGroup)
```

Set protocol group for protocol based vlan.

Defined in: wlan.h

Parameters	<i>protoGroupIdx</i> protocol group index <i>*pProtoGroup</i> protocol group
-------------------	---

Comments	Frame type is as following: - FRAME_TYPE_ETHERNET - FRAME_TYPE_RFC1042 (SNAP) - FRAME_TYPE_LLCOOTHER
-----------------	---

Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_VLAN_FRAME_TYPE RT_ERR_OUT_OF_RANGE RT_ERR_INPUT	ok failed The module is not initial invalid frame type protocol group index is out of range invalid input parameter
---------------------	--	--

29.42.rtk_vlan_portProtoVlan_get

```
int32 rtk_vlan_portProtoVlan_get(rtk_port_t port, uint32 protoGroupIdx,
                                  rtk_vlan_protoVlanCfg_t *pVlanCfg)
```

Get vlan of specified protocol group on specified port.

Defined in: wlan.h

Parameters

<i>port</i>	port id
<i>protoGroupIdx</i>	protocol group index
<i>*pVlanCfg</i>	pointer to vlan configuration of protocol group

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_OUT_OF_RANGE	protocol group index is out of range
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.43.rtk_vlan_portProtoVlan_set

```
int32 rtk_vlan_portProtoVlan_set(rtk_port_t port, uint32 protoGroupIdx,
                                  rtk_vlan_protoVlanCfg_t *pVlanCfg)
```

Set vlan of specified protocol group on specified port.

Defined in: wlan.h

Parameters

<i>port</i>	port id
<i>protoGroupIdx</i>	protocol group index
<i>*pVlanCfg</i>	vlan configuration of protocol group

Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_OUT_OF_RANGE	protocol group index is out of range
RT_ERR_INPUT	invalid input parameter
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.44.rtk_vlan_tagMode_get

```
int32 rtk_vlan_tagMode_get(rtk_port_t port, rtk_vlan_tagMode_t
                           *pTagMode)
```

Get vlan tagged mode of the port.

Defined in: vlan.h

Parameters	<i>port</i> port id
	<i>*pTagMode</i> pointer buffer of vlan tagged mode

Comments	The vlan tagged mode as following:
	- VLAN_TAG_MODE_ORIGINAL (depend on chip normal decision)
	- VLAN_TAG_MODE_KEEP_FORMAT (keep ingress format to egress)
	- VLAN_TAG_MODE_PRI (always priority tag out)

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.45.rtk_vlan_tagMode_set

```
int32 rtk_vlan_tagMode_set(rtk_port_t port, rtk_vlan_tagMode_t tagMode)
```

	Set vlan tagged mode of the port.	
	Defined in: vlan.h	
Parameters	<p><i>port</i> port id</p> <p><i>tagMode</i> vlan tagged mode</p>	
Comments	<p>The vlan tagged mode as following:</p> <ul style="list-style-type: none"> - VLAN_TAG_MODE_ORIGINAL (depend on chip normal decision) - VLAN_TAG_MODE_KEEP_FORMAT (keep ingress format to egress) - VLAN_TAG_MODE_PRI (always priority tag out) 	
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_INPUT	ok failed The module is not initial invalid port id invalid input parameter

29.46.rtk_vlan_portFid_get

```
int32 rtk_vlan_portFid_get(rtk_port_t *port, rtk_enable_t *pEnable, rtk_fid_t *pFid)
```

Get port-based filtering database

Defined in: vlan.h

Parameters	<i>port</i> Port id. <i>*pEnable</i> enable port <i>*pFid</i> Specified filtering database.
Comments	The API can get port-based filtering database status. If the function is enabled, all input packets will be assigned to the port-based fid regardless vlan tag.
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_INPUT RT_ERR_PORT_ID

29.47.rtk_vlan_portFid_set

`int32 rtk_vlan_portFid_set(rtk_port_t port, rtk_enable_t enable, rtk_fid_t fid)`

Set port-based filtering database

Defined in: `vlan.h`

Parameters

port

Port id.

enable

enable port

fid

Specified filtering database.

Comments

The API can set port-based filtering database. If the function is enabled, all input packets will be assigned to the port-based fid regardless vlan tag.

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

`RT_ERR_L2_FID`

Invalid fid.

`RT_ERR_INPUT`

Invalid input parameter.

`RT_ERR_PORT_ID`

Invalid port ID.

29.48.rtk_vlan_portPriority_get

`int32 rtk_vlan_portPriority_get(rtk_port_t port, rtk_pri_t *pPriority)`

Get port-based priority

Defined in: `vlan.h`

Parameters

port

Port id.

**pPriority*

port

Comments

Return Codes

`RT_ERR_OK`

ok

RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameters.
RT_ERR_PORT_ID	Invalid port ID.

29.49.rtk_vlan_portPriority_set

`int32 rtk_vlan_portPriority_set(rtk_port_t port, rtk_pri_t priority)`

Set port-based priority

Defined in: vlan.h

Parameters

port
Port id.
priority
VLAN port

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameter.
RT_ERR_PORT_ID	Invalid port ID.

29.50.rtk_vlan_portEgrTagKeepType_get

`int32 rtk_vlan_portEgrTagKeepType_get(rtk_port_t egr_port, rtk_portmask_t *pIgr_portmask, rtk_vlan_tagKeepType_t *pType)`

Get egress tag keep type

Defined in: vlan.h

Parameters

egr_port
Egress port id.
**pIgr_portmask*
Pointer of Ingress portmask
**pType*
Pointer of tag keep type

Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameter.
RT_ERR_PORT_ID	Invalid port ID.
RT_ERR_NULL_POINTER	Null pointer

29.51.rtk_vlan_portEgrTagKeepType_set

```
int32 rtk_vlan_portEgrTagKeepType_set(rtk_port_t egr_port,
                                       rtk_portmask_t *pIgr_portmask, rtk_vlan_tagKeepType_t type)
```

Set egress tag keep type

Defined in: vlan.h

Parameters	<i>egr_port</i> Egress port id.
	<i>*pIgr_portmask</i> Pointer of Ingress portmask
	<i>type</i> Tag keep type

Comments	None
Return Codes	
RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameter.
RT_ERR_PORT_ID	Invalid port ID.
RT_ERR_NULL_POINTER	Null pointer
RT_ERR_PORT_MASK	Invalid port mask.

29.52.rtk_vlan_transparentEnable_get

```
int32 rtk_vlan_transparentEnable_get(rtk_enable_t *pEnable)
```

Get state of VLAN transparent

Defined in: vlan.h

Parameters **pEnable*
 Pointer of VLAN transparent function

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NULL_POINTER	Null pointer

29.53.rtk_vlan_transparentEnable_set

`int32 rtk_vlan_transparentEnable_set(rtk_enable_t enable)`

Set state of VLAN transparent

Defined in: vlan.h

Parameters *enable*
 VLAN transparent function.

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_INPUT	Invalid input parameter.

29.54.rtk_vlan_cfiKeepEnable_get

`int32 rtk_vlan_cfiKeepEnable_get(rtk_enable_t *pEnable)`

Get state of CFI keep

Defined in: vlan.h

Parameters **pEnable*
 Pointer of CFI Keep

Comments ENABLED: Keep original CFI value DISABLED: Always output VLAN tag with CFI = 0

Return Codes	RT_ERR_OK	ok
---------------------	-----------	----

RT_ERR_FAILED	failed
RT_ERR_NULL_POINTER	Null pointer

29.55.rtk_vlan_cfiKeepEnable_set

int32 rtk_vlan_cfiKeepEnable_set(rtk_enable_t enable)

Set state of CFI keep

Defined in: vlan.h

Parameters

enable
state of CFI KEEP

Comments

ENABLED: Keep original CFI value
DISABLED: Always output VLAN tag with
CFI = 0

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_INPUT	Invalid input parameter.

29.56.rtk_vlan_reservedVidAction_get

**int32 rtk_vlan_reservedVidAction_get(rtk_vlan_resVidAction_t
*pAction_vid0, rtk_vlan_resVidAction_t *pAction_vid4095)**

Get the action of VID 0 and VID 4095 packet

Defined in: vlan.h

Parameters

**pAction_vid0*
Pointer of VID 0 action
**pAction_vid4095*
Pointer of VID 4095 action

Comments

RESVID_ACTION_UNTAG: VID 0 or VID 4095 tagged packets will be treated
as untagged packets
RESVID_ACTION_UNTAG: VID 0 or VID 4095 tagged
packets will be treated as tagged packets

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

RT_ERR_NULL_POINTER	Null pointer
---------------------	--------------

29.57.rtk_vlan_reservedVidAction_set

`int32 rtk_vlan_reservedVidAction_set(rtk_vlan_resVidAction_t action_vid0,
rtk_vlan_resVidAction_t action_vid4095)`

Set the action of VID 0 and VID 4095 packet

Defined in: vlan.h

Parameters

action_vid0

Pointer of VID 0 action

action_vid4095

Pointer of VID 4095 action

Comments

RESVID_ACTION_UNTAG: VID 0 or VID 4095 tagged packets will be treated as untagged packets
 RESVID_ACTION_UNTAG: VID 0 or VID 4095 tagged packets will be treated as tagged packets

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_INPUT

Error Input

29.58.rtk_vlan_tagModeIp4mc_get

`int32 rtk_vlan_tagModeIp4mc_get(rtk_port_t port, rtk_vlan_tagModeIpmc_t
*pTagMode)`

Get vlan tagged mode for ipv4 multicast packet of the port.

Defined in: vlan.h

Parameters

port

port id

**pTagMode*

pointer buffer of vlan tagged mode

Comments

The ipv4 multicast vlan tagged mode as following:

- VLAN_TAG_MODE_IPMC_ORIGINAL (depend on chip normal decision)

- VLAN_TAG_MODE_IPMC_KEEP_FORMAT (keep ingress format to egress)
- VLAN_TAG_MODE_IPMC_PRI (always priority tag out)
- VLAN_TAG_MODE_IPMC_DEFUAL (as default tag mode setting)

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_NULL_POINTER	input parameter may be null pointer

29.59.rtk_vlan_tagModeIp4mc_set

```
int32 rtk_vlan_tagModeIp4mc_set(rtk_port_t port, rtk_vlan_tagModeIpmc_t tagMode)
```

Set vlan tagged mode for ipv4 multicast packet of the port.

Defined in: wlan.h

Parameters	<i>port</i> port id	<i>tagMode</i> vlan tagged mode
Comments		
	The ipv4 multicast vlan tagged mode as following:	
	- VLAN_TAG_MODE_IPMC_ORIGINAL (depend on chip normal decision)	
	- VLAN_TAG_MODE_IPMC_KEEP_FORMAT (keep ingress format to egress)	(keep ingress format to egress)
	- VLAN_TAG_MODE_IPMC_PRI (always priority tag out)	
	- VLAN_TAG_MODE_IPMC_DEFUAL (as default tag mode setting)	

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_PORT_ID	invalid port id
	RT_ERR_INPUT	invalid input parameter

29.60.rtk_vlan_tagModeIp6mc_get

```
int32 rtk_vlan_tagModeIp6mc_get(rtk_port_t port, rtk_vlan_tagModeIpmc_t
*pTagMode)
```

Get vlan tagged mode for ipv6 multicast packet of the port.

Defined in: vlan.h

Parameters

<i>port</i>	port id
<i>*pTagMode</i>	pointer buffer of vlan tagged mode

Comments

The ipv4 multicast vlan tagged mode as following:

- VLAN_TAG_MODE_IPMC_ORIGINAL (depend on chip normal decision)
- VLAN_TAG_MODE_IPMC_KEEP_FORMAT (keep ingress format to egress)
- VLAN_TAG_MODE_IPMC_PRI (always priority tag out)
- VLAN_TAG_MODE_IPMC_DEFAULT (as default tag mode setting)

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_PORT_ID	invalid port id
RT_ERR_NULL_POINTER	input parameter may be null pointer

29.61.rtk_vlan_tagModeIp6mc_set

```
int32 rtk_vlan_tagModeIp6mc_set(rtk_port_t port, rtk_vlan_tagModeIpmc_t
tagMode)
```

Set vlan tagged mode for ipv6 multicast packet of the port.

Defined in: vlan.h

Parameters

<i>port</i>	port id
<i>tagMode</i>	vlan tagged mode

Comments	The ipv4 multicast vlan tagged mode as following: - VLAN_TAG_MODE_IPMC_ORIGINAL (depend on chip normal decision) - VLAN_TAG_MODE_IPMC_KEEP_FORMAT (keep ingress format to egress) - VLAN_TAG_MODE_IPMC_PRI - VLAN_TAG_MODE_IPMC_DEFAULT (always priority tag out) (as default tag mode setting)
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_PORT_ID RT_ERR_INPUT

30. Module EPON MAC APIs

Filename: epon.h

Description Provide the APIs to access EPON MAC

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

30.1. List of Symbols

Here is a list of all functions and variables in this module

epon.h - EPON MAC register access APIs
 rtk_epon_init
 rtk_epon_intrMask_get
 rtk_epon_intrMask_set
 rtk_epon_intr_get
 rtk_epon_intr_disableAll
 rtk_epon_llid_entry_set
 rtk_epon_llid_entry_get
 rtk_epon_forceLaserState_set
 rtk_epon_forceLaserState_get
 rtk_epon_laserTime_set
 rtk_epon_laserTime_get
 rtk_epon_syncTime_get
 rtk_epon_registerReq_get
 rtk_epon_registerReq_set

rtk_epon_churningKey_set
 rtk_epon_churningKey_get
 rtk_epon_usFecState_get
 rtk_epon_usFecState_set
 rtk_epon_dsFecState_get
 rtk_epon_dsFecState_set
 rtk_epon_mibCounter_get
 rtk_epon_mibGlobal_reset
 rtk_epon_mibLlidIdx_reset
 rtk_epon_losState_get

30.2. rtk_epon_init

int32 rtk_epon_init(void)

epon register level initial function

Defined in: epon.h

Parameters

void

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

30.3. rtk_epon_intrMask_get

int32 rtk_epon_intrMask_get(rtk_epon_intrType_t *intrType*, rtk_enable_t **pState*)

Get EPON interrupt mask

Defined in: epon.h

Parameters

intrType

**pState*

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.4. rtk_epon_intrMask_set

`int32 rtk_epon_intrMask_set(rtk_epon_intrType_t intrType, rtk_enable_t state)`

Set EPON interrupt mask

Defined in: epon.h

Parameters

intrType
state

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.5. rtk_epon_intr_get

`int32 rtk_epon_intr_get(rtk_epon_intrType_t intrType, rtk_enable_t *pState)`

Set EPON interrupt state

Defined in: epon.h

Parameters

intrType
**pState*

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.6. rtk_epon_intr_disableAll

`int32 rtk_epon_intr_disableAll(void)`

Disable all of top interrupt for EPON

Defined in: epon.h

Parameters

`void`

Comments

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

30.7. rtk_epon_llid_entry_set

`int32 rtk_epon_llid_entry_set(rtk_epon_llid_entry_t *pLlidEntry)`

Set llid entry

Defined in: epon.h

Parameters

`*pLlidEntry`

Comments

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

30.8. rtk_epon_llid_entry_get

`int32 rtk_epon_llid_entry_get(rtk_epon_llid_entry_t *pLlidEntry)`

Get llid entry

Defined in: epon.ha

Parameters

**pLlidEntry*

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.9. rtk_epon_forceLaserState_set

`int32 rtk_epon_forceLaserState_set(rtk_epon_laser_status_t laserStatus)`

Set Force Laser status

Defined in: epon.h

Parameters	<i>laserStatus</i>
-------------------	--------------------

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.10.rtk_epon_forceLaserState_get

`int32 rtk_epon_forceLaserState_get(rtk_epon_laser_status_t *pLaserStatus)`

Get Force Laser status

Defined in: epon.h

Parameters	<i>*pLaserStatus</i>
-------------------	----------------------

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.11.rtk_epon_laserTime_set

`int32 rtk_epon_laserTime_set(uint8 laserOnTime, uint8 laserOffTime)`

Set laserTime value

Defined in: epon.h

Parameters

laserOnTime

laserOffTime

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

30.12.rtk_epon_laserTime_get

`int32 rtk_epon_laserTime_get(uint8 *pLaserOnTime, uint8 *pLaserOffTime)`

Get laser Time value

Defined in: epon.h

Parameters

**pLaserOnTime*

**pLaserOffTime*

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

30.13.rtk_epon_syncTime_get

`int32 rtk_epon_syncTime_get(uint8 *pSyncTime)`

Get sync Time value

Defined in: epon.h

Parameters **pSyncTime*

Comments

Return Codes RT_ERR_OK
RT_ERR_FAILED

ok
failed

30.14.rtk_epon_registerReq_get

`int32 rtk_epon_registerReq_get(rtk_epon_regReq_t *pRegEntry)`

Get register request relative parameter

Defined in: epon.h

Parameters **pRegEntry*

Comments

Return Codes RT_ERR_OK
RT_ERR_FAILED

ok
failed

30.15.rtk_epon_registerReq_set

`int32 rtk_epon_registerReq_set(rtk_epon_regReq_t *pRegEntry)`

Set register request relative parameter

Defined in: epon.h

Parameters **pRegEntry*

Comments

Return Codes RT_ERR_OK
RT_ERR_FAILED

ok
failed

30.16.rtk_epon_churningKey_set

`int32 rtk_epon_churningKey_set(rtk_epon_churningKeyEntry_t *pEntry)`

Set churning key entry

Defined in: epon.h

Parameters

`*pEntry`

Comments

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

30.17.rtk_epon_churningKey_get

`int32 rtk_epon_churningKey_get(rtk_epon_churningKeyEntry_t *pEntry)`

Get churning key entry

Defined in: epon.h

Parameters

`*pEntry`

Comments

Return Codes

`RT_ERR_OK`

ok

`RT_ERR_FAILED`

failed

30.18.rtk_epon_usFecState_get

`int32 rtk_epon_usFecState_get(rtk_enable_t *pState)`

Get upstream fec state

Defined in: epon.h

Parameters

**pState*

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.19.rtk_epon_usFecState_set

`int32 rtk_epon_usFecState_set(rtk_enable_t state)`

Set upstream fec state

Defined in: epon.h

Parameters

state

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.20.rtk_epon_dsFecState_get

`int32 rtk_epon_dsFecState_get(rtk_enable_t *pState)`

Get down-stream fec state

Defined in: epon.h

Parameters

**pState*
stream FEC state

Comments

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

30.21.rtk_epon_dsFecState_set

`int32 rtk_epon_dsFecState_set(rtk_enable_t state)`

Set down-stream fec state

Defined in: epon.h

Parameters

state
stream FEC state

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

30.22.rtk_epon_mibCounter_get

`int32 rtk_epon_mibCounter_get(rtk_epon_counter_t *pCounter)`

Set down-stream fec state

Defined in: epon.h

Parameters

**pCounter*
stream FEC state

Comments

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

30.23.rtk_epon_mibGlobal_reset

`int32 rtk_epon_mibGlobal_reset(void)`

Reset EPON global counters.

Defined in: epon.h

Parameters

void

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_STAT_GLOBAL_CNTR_FAI	Could not reset Global Counter L

30.24.rtk_epon_mibLlidIdx_reset

int32 rtk_epon_mibLlidIdx_reset(uint8 llidIdx)

Reset the specified LLID index counters.

Defined in: epon.h

Parameters

llidIdx
LLID table index

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

30.25.rtk_epon_losState_get

int32 rtk_epon_losState_get(rtk_enable_t *pState)

Get laser lose of signal state.

Defined in: epon.h

Parameters

**pState*
LLID table index

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

31. Module L34 API

Filename: l34.h

Description

The file includes the following modules and sub-modules

- (1) L34 Networking Interface configuration
- (2) L34 Routing Table configuration
- (3) L34 ARP Table configuration
- (4) L34 NAPT connection configuration
- (5) L34 System configuration
- (6) L34 NAPTR configuration
- (7) L34 NEXT-HOP configuration
- (8) L34 External_Internal IP configuration
- (9) L34 Binding configuration
- (10) L34 IPv6 configuration

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

31.1. List of Symbols

Here is a list of all functions and variables in this module

l34.h - Definition of L34 API
rtk_l34_init
rtk_l34_netifTable_set
rtk_l34_netifTable_get
rtk_l34_arpTable_set
rtk_l34_arpTable_get
rtk_l34_arpTable_del
rtk_l34_pppoeTable_set
rtk_l34_pppoeTable_get
rtk_l34_routingTable_set
rtk_l34_routingTable_get
rtk_l34_routingTable_del
rtk_l34_nexthopTable_set
rtk_l34_nexthopTable_get
rtk_l34_extIntIPTable_set
rtk_l34_extIntIPTable_get
rtk_l34_extIntIPTable_del
rtk_l34_naptInboundTable_set
rtk_l34_naptInboundTable_get
rtk_l34_naptOutboundTable_set
rtk_l34_naptOutboundTable_get
rtk_l34_ipmcTransTable_set

```

rtk_l34_ipmcTransTable_get
rtk_l34_table_reset
rtk_l34_ipv6RoutingTable_set
rtk_l34_ipv6RoutingTable_get
rtk_l34_ipv6NeighborTable_set
rtk_l34_ipv6NeighborTable_get
rtk_l34_hsabMode_set
rtk_l34_hsabMode_get
rtk_l34_hsaData_get
rtk_l34_hsbData_get
rtk_l34_hsdState_set
rtk_l34_hsdState_get
rtk_l34_hsdState_get
rtk_l34_hwL4TrfWrkTbl_set
rtk_l34_hwL4TrfWrkTbl_get
rtk_l34_l4TrfTb_get
rtk_l34_hwL4TrfWrkTbl_Clear
rtk_l34_hwArpTrfWrkTbl_set
rtk_l34_hwArpTrfWrkTbl_get
rtk_l34_arpTrfTb_get
rtk_l34_hwArpTrfWrkTbl_Clear

```

31.2. rtk_l34_init

`int32 rtk_l34_init(void)`

Initialize l34 module.

Defined in: l34.h

Parameters

`void`

Comments

Must initialize l34 module before calling any l34 APIs.

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

31.3. rtk_l34_netifTable_set

`int32 rtk_l34_netifTable_set(uint32 idx, rtk_l34_netif_entry_t *netifEntry)`

Set netif table entry

Defined in: l34.h

Parameters

idx

index of netif table

**netifEntry*

point of netif entry

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

31.4. rtk_l34_netifTable_get

`int32 rtk_l34_netifTable_get(uint32 idx, rtk_l34_netif_entry_t *netifEntry)`

Get netif table entry

Defined in: l34.h

Parameters

idx

index of netif table

**netifEntry*

point of netif entry result

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

31.5. rtk_l34_arpTable_set

`int32 rtk_l34_arpTable_set(uint32 idx, rtk_l34_arp_entry_t *arpEntry)`

Set arp table entry

Defined in: l34.h

Parameters

idx
index of arp table
**arpEntry*
point of arp entry

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

31.6. rtk_l34_arpTable_get

`int32 rtk_l34_arpTable_get(uint32 idx, rtk_l34_arp_entry_t *arpEntry)`

Get arp table entry

Defined in: l34.h

Parameters

idx
index of arp table
**arpEntry*
point of arp entry result

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

31.7. rtk_l34_arpTable_del

`int32 rtk_l34_arpTable_del(uint32 idx)`

Delete arp table entry

Defined in: l34.h

Parameters

idx
index of arp table

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

31.8. rtk_l34_pppoeTable_set

`int32 rtk_l34_pppoeTable_set(uint32 idx, rtk_l34_pppoe_entry_t *pppEntry)`

Set PPPoE table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of PPPoE table
<i>*pppEntry</i>	point of PPPoE entry

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

31.9. rtk_l34_pppoeTable_get

```
int32 rtk_l34_pppoeTable_get(uint32 idx, rtk_l34_pppoe_entry_t *pppEntry)
```

Get PPPoE table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of PPPoE table
<i>*pppEntry</i>	point of PPPoE entry result

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.10.rtk_l34_routingTable_set

```
int32 rtk_l34_routingTable_set(uint32 idx, rtk_l34_routing_entry_t
*routeEntry)
```

Set Routing table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of Routing table
<i>*routeEntry</i>	point of Routing entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.11.rtk_l34_routingTable_get

```
int32 rtk_l34_routingTable_get(uint32 idx, rtk_l34_routing_entry_t
    *routeEntry)
```

Get Routing table entry

Defined in: l34.h

Parameters

idx
index of Routing table

**routeEntry*
point of Routing entry result

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.12.rtk_l34_routingTable_del

```
int32 rtk_l34_routingTable_del(uint32 idx)
```

Delete arp Routing entry

Defined in: l34.h

Parameters

idx
index of Routing table

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.13.rtk_l34_nexthopTable_set

```
int32 rtk_l34_nexthopTable_set(uint32 idx, rtk_l34_nexthop_entry_t
*nextHopEntry)
```

Set Next-Hop table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of Next
<i>*nextHopEntry</i>	point of Next

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.14.rtk_l34_nexthopTable_get

```
int32 rtk_l34_nexthopTable_get(uint32 idx, rtk_l34_nexthop_entry_t
*nextHopEntry)
```

Get Next-Hop table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of Next
<i>*nextHopEntry</i>	point of Next

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.15.rtk_l34_extIntIPTable_set

```
int32 rtk_l34_extIntIPTable_set(uint32 idx, rtk_l34_ext_intip_entry_t
*extIpEntry)
```

Set External_Internal IP table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of External_Internal IP table
------------	-------------------------------------

<i>*extIpEntry</i>	point of External_Internal IP entry
--------------------	-------------------------------------

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.16.rtk_l34_extIntIPTable_get

```
int32 rtk_l34_extIntIPTable_get(uint32 idx, rtk_l34_ext_intip_entry_t
*extIpEntry)
```

Get External_Internal IP table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of External_Internal IP table
------------	-------------------------------------

<i>*extIpEntry</i>	point of External_Internal IP entry result
--------------------	--

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.17.rtk_l34_extIntIPTable_del

int32 rtk_l34_extIntIPTable_del(uint32 idx)

Delete arp External_Internal entry

Defined in: l34.h

Parameters *idx*
index of External_Internal table

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

31.18.rtk_l34_naptInboundTable_set

**int32 rtk_l34_naptInboundTable_set(int8 forced, uint32 idx,
rtk_l34_naptInbound_entry_t *naptrEntry)**

Set NAPTR table entry

Defined in: l34.h

Parameters *forced*
force set to NAPTR table
idx
value of NAPTR table entry
**naptrEntry*

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

31.19.rtk_l34_naptInboundTable_get

```
int32 rtk_l34_naptInboundTable_get(uint32 idx,
    rtk_l34_naptInbound_entry_t *naptrEntry)
```

Get napt table entry

Defined in: l34.h

Parameters

idx
index of NAPTR table

**naptrEntry*
point of NAPTR entry result

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.20.rtk_l34_naptOutboundTable_set

```
int32 rtk_l34_naptOutboundTable_set(int8 forced, uint32 idx,
    rtk_l34_naptOutbound_entry_t *naptEntry)
```

Set napt table entry

Defined in: l34.h

Parameters

forced
force set to napt table

idx
value of napt table entry

**naptEntry*

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

RT_ERR_NULL_POINTER	input parameter may be null pointer
---------------------	-------------------------------------

31.21.rtk_l34_naptOutboundTable_get

`int32 rtk_l34_naptOutboundTable_get(uint32 idx,
rtk_l34_naptOutbound_entry_t *naptEntry)`

Get napt table entry

Defined in: l34.h

Parameters

idx
index of napt table

**naptEntry*
point of napt entry result

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.22.rtk_l34_ipmcTransTable_set

`int32 rtk_l34_ipmcTransTable_set(uint32 idx, rtk_l34_ipmcTrans_entry_t
*ipmcEntry)`

Set IPMC Transfer table entry

Defined in: l34.h

Parameters

idx
force set to IPMC Transfer table

**ipmcEntry*
value of IPMC Transfer entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.23.rtk_l34_ipmcTransTable_get

`int32 rtk_l34_ipmcTransTable_get(uint32 idx, rtk_l34_ipmcTrans_entry_t *ipmcEntry)`

Get IPMC Transfer table entry

Defined in: l34.h

Parameters

<i>idx</i>	index of IPMC Transfer table
<i>*ipmcEntry</i>	point of IPMC Transfer entry result

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.24.rtk_l34_table_reset

`int32 rtk_l34_table_reset(rtk_l34_table_type_t type)`

Reset a specific L34 table entries

Defined in: l34.h

Parameters

<i>type</i>	L34 Table type
-------------	----------------

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.25.rtk_l34_ipv6RoutingTable_set

```
int32 rtk_l34_ipv6RoutingTable_set(uint32 idx, rtk_ipv6Routing_entry_t
*ipv6RoutEntry)
```

Set a IPv6 routing entry by idx.

Defined in: l34.h

Parameters

<i>idx</i>	index of ipv6 routing entry
<i>*ipv6RoutEntry</i>	point of ipv6 routing table

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.26.rtk_l34_ipv6RoutingTable_get

```
int32 rtk_l34_ipv6RoutingTable_get(uint32 idx, rtk_ipv6Routing_entry_t
*ipv6RoutEntry)
```

Get a IPv6 routing entry by idx.

Defined in: l34.h

Parameters

<i>idx</i>	index of ipv6 routing entry
<i>*ipv6RoutEntry</i>	point of ipv6 routing table

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial

RT_ERR_NULL_POINTER	input parameter may be null pointer
---------------------	-------------------------------------

31.27.rtk_l34_ipv6NeighborTable_set

`int32 rtk_l34_ipv6NeighborTable_set(uint32 idx, rtk_ipv6Neighbor_entry_t *ipv6NeighborEntry)`

Set neighbor table

Defined in: l34.h

Parameters

idx
index of neighbor table
**ipv6NeighborEntry*
point of neighbor data

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.28.rtk_l34_ipv6NeighborTable_get

`int32 rtk_l34_ipv6NeighborTable_get(uint32 idx, rtk_ipv6Neighbor_entry_t *ipv6NeighborEntry)`

Get neighbor table

Defined in: l34.h

Parameters

idx
index of neighbor table
**ipv6NeighborEntry*
point of neighbor data

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

31.29.rtk_l34_hsabMode_set

`int32 rtk_l34_hsabMode_set(rtk_l34_hsba_mode_t hsabMode)`

Set L34 hsab mode

Defined in: l34.h

Parameters

hsabMode

L34 hsab

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

31.30.rtk_l34_hsabMode_get

`int32 rtk_l34_hsabMode_get(rtk_l34_hsba_mode_t *pHsabMode)`

Get L34 hsab mode

Defined in: l34.h

Parameters

**pHsabMode*

point of L34 hsab

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

RT_ERR_NOT_INIT

The module is not initial

RT_ERR_NULL_POINTER

input parameter may be null pointer

31.31.rtk_l34_hsaData_get

`int32 rtk_l34_hsaData_get(rtk_l34_hsa_t *pHsaData)`

Get L34 hsa data

Defined in: l34.h

Parameters `*pHsaData`
 point of hsa data

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

31.32.rtk_l34_hsbData_get

`int32 rtk_l34_hsbData_get(rtk_l34_hsb_t *pHsbData)`

Get L34 hsab mode

Defined in: l34.h

Parameters `*pHsbData`
 point of hsa data

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

31.33.rtk_l34_hsdState_set

`int32 rtk_l34_hsdState_set(rtk_enable_t hsdState)`

	Set L34 hsd state
	Defined in: l34.h
Parameters	<i>hsdState</i> L34 hsd state
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_NULL_POINTER
	ok failed The module is not initial input parameter may be null pointer

31.34.rtk_l34_hsdState_get

```
int32 rtk_l34_hsdState_get(rtk_enable_t *phsdState)
```

Get L34 hsab mode

Defined in: l34.h

Parameters	<i>*phsdState</i> point of hsd state
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED RT_ERR_NOT_INIT RT_ERR_NULL_POINTER
	ok failed The module is not initial input parameter may be null pointer

31.35.rtk_l34_hsdState_get

```
int32 rtk_l34_hsdState_get(rtk_enable_t *phsdState)
```

Get L34 hsab mode

Defined in: l34.h

Parameters	<i>*phsdState</i> point of hsd state
-------------------	---

Comments	None	
Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed
	RT_ERR_NOT_INIT	The module is not initial
	RT_ERR_NULL_POINTER	input parameter may be null pointer

31.36.rtk_l34_hwL4TrfWrkTbl_set

`int32 rtk_l34_hwL4TrfWrkTbl_set(rtk_l34_l4_trf_t *l4TrfTable)`

Set HW working table id for L4 trf.

Defined in: l34.h

Parameters	<i>l4TrfTable</i>	
	point of hsd state	

Comments	None	
Return Codes	RT_ERR_OK	ok

	RT_ERR_SMI	SMI access error
--	------------	------------------

31.37.rtk_l34_hwL4TrfWrkTbl_get

`int32 rtk_l34_hwL4TrfWrkTbl_get(rtk_l34_l4_trf_t *pl4TrfTable)`

Get HW working table id for L4 trf.

Defined in: l34.h

Parameters	<i>*pl4TrfTable</i>	
	point of hsd state	

Comments	None	
Return Codes	RT_ERR_OK	ok

	RT_ERR_SMI	SMI access error
--	------------	------------------

31.38.rtk_l34_l4TrfTb_get

```
int32 rtk_l34_l4TrfTb_get(rtk_l34_l4_trf_t l4TrfTable, uint32 l4EntryIndex,
                           rtk_enable_t *pIndicator)
```

Get HW working table id for L4 trf.

Defined in: l34.h

Parameters

l4TrfTable
table index

l4EntryIndex
index of l4 table that went to get

**pIndicator*
indicator for result of state

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_SMI	SMI access error

31.39.rtk_l34_hwL4TrfWrkTbl_Clear

```
int32 rtk_l34_hwL4TrfWrkTbl_Clear(rtk_l34_l4_trf_t l4TrfTable)
```

Clear HW working table id for ARP trf.

Defined in: l34.h

Parameters

l4TrfTable
table index

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_SMI	SMI access error

31.40.rtk_l34_hwArpTrfWrkTbl_set

```
int32 rtk_l34_hwArpTrfWrkTbl_set(rtk_l34_arp_trf_t arpTrfTable)
```

Set HW working table id for ARP trf.

Defined in: l34.h

Parameters *arpTrfTable*
table index

Comments None

Return Codes RT_ERR_OK ok
RT_ERR_SMI SMI access error

31.41.rtk_l34_hwArpTrfWrkTbl_get

`int32 rtk_l34_hwArpTrfWrkTbl_get(rtk_l34_arp_trf_t *pArpTrfTable)`

Get HW working table id for ARP trf.

Defined in: l34.h

Parameters **pArpTrfTable*
table index

Comments None

Return Codes RT_ERR_OK ok
RT_ERR_SMI SMI access error

31.42.rtk_l34_arpTrfTb_get

`int32 rtk_l34_arpTrfTb_get(rtk_l34_arp_trf_t arpTrfTable, uint32 arpEntryIndex, rtk_enable_t *pIndicator)`

Get HW working table id for ARP trf.

Defined in: l34.h

Parameters *arpTrfTable*
table index

arpEntryIndex
index of l4 table that went to get

<i>*pIndicator</i>	indicator for result of state				
Comments	None				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_SMI</td> <td>SMI access error</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_SMI	SMI access error
RT_ERR_OK	ok				
RT_ERR_SMI	SMI access error				

31.43.rtk_l34_hwArpTrfWrkTbl_Clear

`int32 rtk_l34_hwArpTrfWrkTbl_Clear(rtк_l34_arp_trf_t arpTrfTable)`

Clear HW working table id for ARP trf.

Defined in: l34.h

Parameters	<i>arpTrfTable</i> table index
-------------------	-----------------------------------

Comments	None
-----------------	------

Return Codes	<table> <tr> <td>RT_ERR_OK</td><td>ok</td></tr> <tr> <td>RT_ERR_SMI</td><td>SMI access error</td></tr> </table>	RT_ERR_OK	ok	RT_ERR_SMI	SMI access error
RT_ERR_OK	ok				
RT_ERR_SMI	SMI access error				

32. Module L34 Lite API

Filename: l34lite.h

Description	The file includes the following modules and sub-modules (1) L34 Networking Interface configuration (2) L34 Routing Table configuration (3) L34 ARP Table configuration (4) L34 NAPT connection configuration (5) L34 Global configuration (6) L34 IPv6 Routing Table (7) L34 IPv6 Neighbor Table
--------------------	--

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

32.1. List of Symbols

Here is a list of all functions and variables in this module

l34lite.h - Definition of L34 Lite API
rtk_l34_lite_init
rtk_l34_netif_create
rtk_l34_netifPPPoE_set
rtk_l34_netifVlan_set
rtk_l34_netifRoutingState_set
rtk_l34_netifMtu_set
rtk_l34_netifIpaddr_set
rtk_l34_netifNat_set
rtk_l34_netifState_set
rtk_l34_netif_get
rtk_l34_netifGateway_set
rtk_l34_netif_set
rtk_l34_netif_del
rtk_l34_arp_add
rtk_l34_arp_get
rtk_l34_arp_del
rtk_l34_route_add
rtk_l34_route_del
rtk_l34_connectTrack_add
rtk_l34_connectTrack_get
rtk_l34_connectTrack_del
rtk_l34_globalCfg_get
rtk_l34_route6_add
rtk_l34_route6_del
rtk_l34_route6_get
rtk_l34_neigh6_add
rtk_l34_neigh6_del
rtk_l34_neigh6_get
rtk_l34_netifIp6addr_add
rtk_l34_netifIp6addr_del
rtk_l34_netifMac_set

32.2. rtk_l34_lite_init

int32 rtk_l34_lite_init(void)

Initialize l34 lite module.

Defined in: l34lite.h

Parameters *void*

Comments Must initialize l34 lite module before calling any l34 lite APIs.

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

32.3. rtk_l34_netif_create

```
int32 rtk_l34_netif_create(uint32 netifId, rtk_l34_netifType_t netifType,
                           rtk_mac_t ifmac)
```

create a new interface

Defined in: l34lite.h

Parameters

<i>netifId</i>	interface index
----------------	-----------------

<i>netifType</i>	WAN or LAN
------------------	------------

<i>ifmac</i>	MAC address for this interface
--------------	--------------------------------

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

32.4. rtk_l34_netifPPPoE_set

```
int32 rtk_l34_netifPPPoE_set(uint32 netifId, rtk_enable_t pppState, uint32
                             sessionId, rtk_enable_t passThroughState)
```

set pppoe relation parameters

Defined in: l34lite.h

Parameters

<i>netifId</i>	interface index
<i>pppState</i>	enable or disable pppState

<i>sessionId</i>	PPPoE session id
<i>passThroughState</i>	enable or disable PPPoE pass through function in this interface.
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.5. rtk_l34_netifVlan_set

`int32 rtk_l34_netifVlan_set(uint32 netifId, rtk_vlan_t vid, rtk_pri_t defaultPri)`

set interface vlan configure

Defined in: l34lite.h

Parameters	<i>netifId</i> interface index
	<i>vid</i> vlan id for interface
	<i>defaultPri</i> default priority for interface, value 8 means disable
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.6. rtk_l34_netifRoutingState_set

`int32 rtk_l34_netifRoutingState_set(uint32 netifId, rtk_enable_t routingState)`

set interface routing state

Defined in: l34lite.h

Parameters	<i>netifId</i> interface id
-------------------	--------------------------------

Comments	None				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

32.7. rtk_l34_netifMtu_set

`int32 rtk_l34_netifMtu_set(uint32 netifId, uint32 mtu)`

set interface mtu size

Defined in: l34lite.h

Parameters	<i>netifId</i> interface id
	<i>mtu</i> interface mtu size

Comments	None				
Return Codes	<table> <tr> <td>RT_ERR_OK</td> <td>ok</td> </tr> <tr> <td>RT_ERR_FAILED</td> <td>failed</td> </tr> </table>	RT_ERR_OK	ok	RT_ERR_FAILED	failed
RT_ERR_OK	ok				
RT_ERR_FAILED	failed				

32.8. rtk_l34_netifIpaddr_set

`int32 rtk_l34_netifIpaddr_set(uint32 netifId, rtk_ip_addr_t ipaddr, uint32 ipmask)`

set interface ipaddress

Defined in: l34lite.h

Parameters	<i>netifId</i> interface id
	<i>ipaddr</i> interface ip address
	<i>ipmask</i> ip mask length

Comments	None
Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed

32.9. rtk_l34_netifNat_set

`int32 rtk_l34_netifNat_set(uint32 netifId, rtk_l34_natType_t natType,
rtk_ip_addr_t internalp)`

set interface nat related configuration

Defined in: l34lite.h

Parameters	<i>netifId</i> interface id
	<i>natType</i> interface nat type for connections
	<i>internalp</i> for nat mode, interanl ip address

Comments None

Return Codes	RT_ERR_OK ok RT_ERR_FAILED failed
---------------------	--------------------------------------

32.10.rtk_l34_netifState_set

`int32 rtk_l34_netifState_set(uint32 netifId, rtk_enable_t ifState)`

set interface status

Defined in: l34lite.h

Parameters	<i>netifId</i> interface id
	<i>ifState</i> interface status, up/down

Comments None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.11.rtk_l34_netif_get

`int32 rtk_l34_netif_get(uint32 netifId, rtk_l34_netifInfo_t *netifInfo)`

get interface information

Defined in: l34lite.h

Parameters

netifId
interface id
**netifInfo*
networking interface information

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.12.rtk_l34_netifGateway_set

`int32 rtk_l34_netifGateway_set(uint32 netifId, rtk_mac_t gatewayMac)`

set net interface gateway configuration

Defined in: l34lite.h

Parameters

netifId
interface id
gatewayMac
MAC address for net interface gateway

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.13.rtk_l34_netif_set

`int32 rtk_l34_netif_set(uint32 netifId, rtk_l34_netifInfo_t netifInfo)`

set all net interface configuration

Defined in: l34lite.h

Parameters

netifId
interface id

netifInfo
net interface information

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

32.14.rtk_l34_netif_del

`int32 rtk_l34_netif_del(uint32 netifId)`

delete interface

Defined in: l34lite.h

Parameters

netifId
interface id

Comments

None

Return Codes

RT_ERR_OK

ok

RT_ERR_FAILED

failed

32.15.rtk_l34_arp_add

`int32 rtk_l34_arp_add(rtk_ip_addr_t ipaddr, rtk_mac_t mac)`

add an arp entry

Defined in: l34lite.h

Parameters	<i>ipaddr</i> ip address
	<i>mac</i> mac address for ipaddr
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.16.rtk_l34_arp_get

`int32 rtk_l34_arp_get(rtk_ip_addr_t ipaddr, rtk_l34_arpInfo_t *arpInfo)`

get mac address from ip address

Defined in: l34lite.h

Parameters	<i>ipaddr</i> ip address for get arp entry
	<i>*arpInfo</i> arp related information
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.17.rtk_l34_arp_del

`int32 rtk_l34_arp_del(rtk_ip_addr_t ipaddr)`

delete an arp entry

Defined in: l34lite.h

Parameters	<i>ipaddr</i> ip address for delete arp entry
Comments	None
Return Codes	RT_ERR_OK
	ok

RT_ERR_FAILED	failed
---------------	--------

32.18.rtk_l34_route_add

`int32 rtk_l34_route_add(rtk_l34_routeType_t routeType,
rtk_l34_routeTable_t routeTable)`

add a routing table entry

Defined in: l34lite.h

Parameters

routeType
type of this routing table
routeTable
routing table necessary information

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.19.rtk_l34_route_del

`int32 rtk_l34_route_del(rtk_ip_addr_t ipaddr, uint32 ipmask)`

get a routing table entry

Defined in: l34lite.h

Parameters

ipaddr
routing table index for get routing information
ipmask
point of routing information

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.20.rtk_l34_connectTrack_add

```
int32 rtk_l34_connectTrack_add(rtk_l34_direct_t dir, rtk_l34_tuple_t tuple,
                               rtk_ip_addr_t natIp, uint16 natport)
```

add a connection tracking table

Defined in: l34lite.h

Parameters

dir
direction of connection

tuple
five tuple for connection

natIp
new source ip address

natport
new source port

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.21.rtk_l34_connectTrack_get

```
int32 rtk_l34_connectTrack_get(rtk_l34_tuple_t tuple, rtk_l34_connectInfo_t
                               *connectInfo)
```

get a connection information

Defined in: l34lite.h

Parameters

tuple
five tuple for connection

**connectInfo*
point of connection information

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.22.rtk_l34_connectTrack_del

`int32 rtk_l34_connectTrack_del(rtk_l34_tuple_t tuple)`

delete a connection

Defined in: l34lite.h

Parameters

tuple
five tuple for connection

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

32.23.rtk_l34_globalCfg_get

`int32 rtk_l34_globalCfg_get(rtk_l34_global_cfg_t *globalCfg)`

get l34 global configuration

Defined in: l34lite.h

Parameters

**globalCfg*
five tuple for connection

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed

32.24.rtk_l34_route6_add

`int32 rtk_l34_route6_add(rtk_l34_ipv6RouteType_t route6Type,
rtk_l34_route6Table_t route6Table)`

add l34 IPv6 routing table

Defined in: l34lite.h

Parameters	<i>route6Type</i> IPv6 route type
	<i>route6Table</i> IPv6 route value
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.25.rtk_l34_route6_del

`int32 rtk_l34_route6_del(rtk_ipv6_addr_t ip6addr, uint32 prefixLen)`

delete l34 IPv6 routing table

Defined in: l34lite.h

Parameters	<i>ip6addr</i> IPv6 address
	<i>prefixLen</i> prefix length
Comments	None
Return Codes	RT_ERR_OK RT_ERR_FAILED
	ok failed

32.26.rtk_l34_route6_get

`int32 rtk_l34_route6_get(uint32 index, rtk_l34_route6Info_t *pRoute6Info)`

Get l34 IPv6 routing table

Defined in: l34lite.h

Parameters	<i>index</i> index of ipv6 information
	<i>*pRoute6Info</i> point of route 6 information
Comments	None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

32.27.rtk_l34_neigh6_add

`int32 rtk_l34_neigh6_add(rtk_ipv6_addr_t ip6addr, rtk_mac_t mac)`

add l34 IPv6 neighbor entry

Defined in: l34lite.h

Parameters	<i>ip6addr</i> ipv6 address	<i>mac</i> mac address for ipaddr
-------------------	--------------------------------	--------------------------------------

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

32.28.rtk_l34_neigh6_del

`int32 rtk_l34_neigh6_del(rtk_ipv6_addr_t ip6addr)`

delete l34 IPv6 routing table

Defined in: l34lite.h

Parameters	<i>ip6addr</i> IPv6 address
-------------------	--------------------------------

Comments None

Return Codes	RT_ERR_OK	ok
	RT_ERR_FAILED	failed

32.29.rtk_l34_neigh6_get

```
int32 rtk_l34_neigh6_get(rtk_ipv6_addr_t ip6addr, rtk_l34_neigh6Info_t
    *pNeigh6Info)
```

Get l34 IPv6 routing table

Defined in: l34lite.h

Parameters

<i>ip6addr</i>	ipv6 address for get
<i>*pNeigh6Info</i>	point of neighbor information

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.30.rtk_l34_netifIp6addr_add

```
int32 rtk_l34_netifIp6addr_add(uint32 netifId, rtk_ipv6_addr_t ip6addr,
    uint32 prefixLen)
```

add interface ipv6 address

Defined in: l34lite.h

Parameters

<i>netifId</i>	interface id
<i>ip6addr</i>	interface ip address
<i>prefixLen</i>	ipv6 prefix length

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.31.rtk_l34_netifIp6addr_del

```
int32 rtk_l34_netifIp6addr_del(uint32 netifId, rtk_ipv6_addr_t ip6addr,
                                uint32 prefixLen)
```

delete interface ipv6 address

Defined in: l34lite.h

Parameters

<i>netifId</i>	interface id
<i>ip6addr</i>	interface ip address
<i>prefixLen</i>	ipv6 prefix length

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

32.32.rtk_l34_netifMac_set

```
int32 rtk_l34_netifMac_set(uint32 netifId, rtk_mac_t hwAddr)
```

set netif MAC address

Defined in: l34lite.h

Parameters

<i>netifId</i>	interface id
<i>hwAddr</i>	interface HW MAC address

Comments

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed

33. Module L34 Binding API

Filename: l34_bind_config.h

Description

The file includes the following modules and sub-modules

- (1) L34 Binding configuration
- (2) L34 Port & WAN mapping
- (3) L34 System Configuration
- (4) L34 Traffic Indicator

Copyright © 2009 Realtek™ Semiconductor Corp. All rights reserved.

33.1. List of Symbols

Here is a list of all functions and variables in this module

l34_bind_config.h - Definition of L34 API
rtk_l34_bindingTable_set
rtk_l34_bindingTable_get
rtk_l34_bindingAction_set
rtk_l34_bindingAction_get
rtk_l34_wanTypeTable_set
rtk_l34_wanTypeTable_get
rtk_l34_portWanMap_set
rtk_l34_portWanMap_get
rtk_l34_globalState_set
rtk_l34_globalState_get
rtk_l34_lookupMode_set
rtk_l34_lookupMode_get
rtk_l34_lookupPortMap_set
rtk_l34_lookupPortMap_get
rtk_l34_wanRoutMode_set
rtk_l34_wanRoutMode_get
rtk_l34_arpTrfIndicator_get
rtk_l34_naptTrfIndicator_get
rtk_l34_pppTrfIndicator_get
rtk_l34_neighTrfIndicator_get
rtk_l34_naptTrfIndicator_get_all
rtk_l34_arpTrfIndicator_get_all

33.2. rtk_l34_bindingTable_set

`int32 rtk_l34_bindingTable_set(uint32 idx, rtk_binding_entry_t *bindEntry)`

Set binding table

Defined in: l34_bind_config.h

Parameters

idx
index of binding table

**bindEntry*
point of binding data

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.3. rtk_l34_bindingTable_get

`int32 rtk_l34_bindingTable_get(uint32 idx, rtk_binding_entry_t *bindEntry)`

Get binding table

Defined in: l34_bind_config.h

Parameters

idx
index of binding table

**bindEntry*
point of binding data

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.4. rtk_l34_bindingAction_set

```
int32 rtk_l34_bindingAction_set(rtk_l34_bindType_t bindType,
                                rtk_l34_bindAct_t bindAction)
```

Set binding action

Defined in: l34_bind_config.h

Parameters

bindType
binding type
bindAction
binding action

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.5. rtk_l34_bindingAction_get

```
int32 rtk_l34_bindingAction_get(rtk_l34_bindType_t bindType,
                                rtk_l34_bindAct_t *bindAction)
```

Get binding action

Defined in: l34_bind_config.h

Parameters

bindType
binding type
**bindAction*
binding action

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.6. rtk_l34_wanTypeTable_set

```
int32 rtk_l34_wanTypeTable_set(uint32 idx, rtk_wanType_entry_t
    *wanTypeEntry)
```

Set WAN type entry by idx.

Defined in: l34_bind_config.h

Parameters

idx
index of wan type table for binding

**wanTypeEntry*
point of wan type table entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.7. rtk_l34_wanTypeTable_get

```
int32 rtk_l34_wanTypeTable_get(uint32 idx, rtk_wanType_entry_t
    *wanTypeEntry)
```

Get WAN type entry by idx.

Defined in: l34_bind_config.h

Parameters

idx
index of wan type table for binding

**wanTypeEntry*
point of wan type table entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.8. rtk_l34_portWanMap_set

```
int32 rtk_l34_portWanMap_set(rtk_l34_portWanMapType_t
    portWanMapType, rtk_l34_portWanMap_entry_t portWanMapEntry)
```

Set L34 WAN interface port mapping

Defined in: l34_bind_config.h

Parameters

portWanMapType
port wan mapping type
portWanMapEntry
port wan mapping entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.9. rtk_l34_portWanMap_get

```
int32 rtk_l34_portWanMap_get(rtk_l34_portWanMapType_t
    portWanMapType, rtk_l34_portWanMap_entry_t *pPortWanMapEntry)
```

Get L34 WAN interface port mapping

Defined in: l34_bind_config.h

Parameters

portWanMapType
port wan mapping type
**pPortWanMapEntry*
point of port wan mapping entry

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.10.rtk_l34_globalState_set

```
int32 rtk_l34_globalState_set(rtk_l34_globalStateType_t stateType,
                               rtk_enable_t state)
```

get l34 global status

Defined in: l34_bind_config.h

Parameters

<i>stateType</i>	status type
<i>state</i>	status of state type

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.11.rtk_l34_globalState_get

```
int32 rtk_l34_globalState_get(rtk_l34_globalStateType_t stateType,
                               rtk_enable_t *pState)
```

set l34 global status

Defined in: l34_bind_config.h

Parameters

<i>stateType</i>	status type
<i>*pState</i>	status of state type

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.12.rtk_l34_lookupMode_set

`int32 rtk_l34_lookupMode_set(rtk_l34_lookupMode_t lookupMode)`

configure l34 lookup mode selection

Defined in: l34_bind_config.h

Parameters

lookupMode
mode of l34 lookup method

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.13.rtk_l34_lookupMode_get

`int32 rtk_l34_lookupMode_get(rtk_l34_lookupMode_t *pLookupMode)`

get l34 lookup mode selection

Defined in: l34_bind_config.h

Parameters

**pLookupMode*
point of mode of l34 lookup method

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.14.rtk_l34_lookupPortMap_set

`int32 rtk_l34_lookupPortMap_set(rtk_l34_portType_t portType, uint32 portId,
uint32 wanIdx)`

configure l34 port base mapping

Defined in: l34_bind_config.h

Parameters

portType
port type, mac port/ext port/vc port

portId
port identity

wanIdx
port based to wanIdx

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.15.rtk_l34_lookupPortMap_get

`int32 rtk_l34_lookupPortMap_get(rtk_l34_portType_t portType, uint32
portId, uint32 *pWanIdx)`

configure l34 port base mapping

Defined in: l34_bind_config.h

Parameters

portType
port type, mac port/ext port/vc port

portId
port identity

**pWanIdx*
port based to wanIdx

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
------------------------	----

RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.16.rtk_l34_wanRoutMode_set

int32 rtk_l34_wanRoutMode_set(rtk_l34_wanRouteMode_t wanRouteMode)

set wan route mode

Defined in: l34_bind_config.h

Parameters

wanRouteMode
mode of wan routed

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.17.rtk_l34_wanRoutMode_get

**int32 rtk_l34_wanRoutMode_get(rtk_l34_wanRouteMode_t
*pWanRouteMode)**

get wan route mode

Defined in: l34_bind_config.h

Parameters

**pWanRouteMode*
point of mode of wan routed

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.18.rtk_l34_arpTrfIndicator_get

`int32 rtk_l34_arpTrfIndicator_get(uint32 index, rtk_enable_t *pArpIndicator)`

get arp entry traffic indicator by index

Defined in: l34_bind_config.h

Parameters

index
traffic table index

**pArpIndicator*
point of traffic indicator for arp

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.19.rtk_l34_naptTrfIndicator_get

`int32 rtk_l34_naptTrfIndicator_get(uint32 index, rtk_enable_t *pNaptIndicator)`

get napt entry traffic indicator by index

Defined in: l34_bind_config.h

Parameters

index
traffic table index

**pNaptIndicator*
point of traffic indicator for arp

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.20.rtk_l34_pppTrfIndicator_get

```
int32 rtk_l34_pppTrfIndicator_get(uint32 index, rtk_enable_t
*pPppIndicator)
```

get ppp entry traffic indicator by index

Defined in: l34_bind_config.h

Parameters

index
traffic table index

**pPppIndicator*
point of traffic indicator for pppoe table

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.21.rtk_l34_neighTrfIndicator_get

```
int32 rtk_l34_neighTrfIndicator_get(uint32 index, rtk_enable_t
*pNeighIndicator)
```

get ipv6 neighbor entry traffic indicator by index

Defined in: l34_bind_config.h

Parameters

index
traffic table index
**pNeighIndicator*
point of traffic indicator for neighbor

Comments

None

Return Codes

RT_ERR_OK	ok
RT_ERR_FAILED	failed
RT_ERR_NOT_INIT	The module is not initial
RT_ERR_NULL_POINTER	input parameter may be null pointer

33.22.rtk_l34_naptTrfIndicator_get_all

`int32 rtk_l34_naptTrfIndicator_get_all(uint32 *pNaptMaps)`

get napt entry traffic indicator

Defined in: l34_bind_config.h

Parameters

`*pNaptMaps`
point of traffic indicator for mask (64)

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer

33.23.rtk_l34_arpTrfIndicator_get_all

`int32 rtk_l34_arpTrfIndicator_get_all(uint32 *pArpMaps)`

get all arp entry traffic indicator

Defined in: l34_bind_config.h

Parameters

`*pArpMaps`
point of traffic indicator for arp

Comments

None

Return Codes

<code>RT_ERR_OK</code>	ok
<code>RT_ERR_FAILED</code>	failed
<code>RT_ERR_NOT_INIT</code>	The module is not initial
<code>RT_ERR_NULL_POINTER</code>	input parameter may be null pointer