

ZebOS-XP RMON SMI Reference
IP Infusion Inc.

Generated by Doxygen 1.6.1

Wed Dec 16 12:33:33 2015

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	smi_alarm_entry Struct Reference	5
3.2	smi_event_indices Struct Reference	6
3.3	smi_msg_rmon Struct Reference	7
3.4	smi_rmon_ifstats Struct Reference	8
4	File Documentation	9
4.1	smi_rmon.h File Reference	9
4.1.1	Detailed Description	15
4.1.2	Function Documentation	16
4.1.2.1	smi_get_rmon_alarm_interval	16
4.1.2.2	smi_get_rmon_alarm_rising_event_index	16
4.1.2.3	smi_get_rmon_alarm_rising_threshold	16
4.1.2.4	smi_get_rmon_alarm_sample_type	17
4.1.2.5	smi_get_rmon_alarm_start_up	17
4.1.2.6	smi_get_rmon_alarm_variable	18
4.1.2.7	smi_get_rmon_coll_history_bucket	18
4.1.2.8	smi_get_rmon_coll_history_index	18
4.1.2.9	smi_get_rmon_coll_history_interval	19
4.1.2.10	smi_get_rmon_coll_history_owner	19

4.1.2.11	smi_get_rmon_coll_history_status	19
4.1.2.12	smi_rmon_alarm_index_remove	20
4.1.2.13	smi_rmon_coll_history_bucket_set	20
4.1.2.14	smi_rmon_coll_history_datasource_set	20
4.1.2.15	smi_rmon_coll_history_index_add_new	21
4.1.2.16	smi_rmon_coll_history_index_remove	21
4.1.2.17	smi_rmon_coll_history_index_set	21
4.1.2.18	smi_rmon_coll_history_interval_set	22
4.1.2.19	smi_rmon_coll_history_owner_set	22
4.1.2.20	smi_rmon_coll_history_set_active	22
4.1.2.21	smi_rmon_coll_history_set_inactive	23
4.1.2.22	smi_rmon_coll_history_validate	23
4.1.2.23	smi_rmon_coll_stats_validate	23
4.1.2.24	smi_rmon_collection_stat_entry_add	24
4.1.2.25	smi_rmon_collection_stat_entry_remove	24
4.1.2.26	smi_rmon_event_index_remove	25
4.1.2.27	smi_rmon_event_type_get	25
4.1.2.28	smi_rmon_event_type_set	25
4.1.2.29	smi_rmon_get_alarm_entry	26
4.1.2.30	smi_rmon_get_alarm_falling_event_index	26
4.1.2.31	smi_rmon_get_alarm_falling_threshold	27
4.1.2.32	smi_rmon_get_alarm_owner	27
4.1.2.33	smi_rmon_get_event_comm	27
4.1.2.34	smi_rmon_get_event_description	28
4.1.2.35	smi_rmon_get_event_index	28
4.1.2.36	smi_rmon_get_event_owner	28
4.1.2.37	smi_rmon_get_event_status	29
4.1.2.38	smi_rmon_get_if_counter	29
4.1.2.39	smi_rmon_get_if_stats	30
4.1.2.40	smi_rmon_set_alarm_entry	30
4.1.2.41	smi_rmon_set_alarm_falling_event_index	31
4.1.2.42	smi_rmon_set_alarm_falling_threshold	31
4.1.2.43	smi_rmon_set_alarm_interval	31
4.1.2.44	smi_rmon_set_alarm_owner	32

4.1.2.45	smi_rmon_set_alarm_rising_event_index	32
4.1.2.46	smi_rmon_set_alarm_rising_threshold	32
4.1.2.47	smi_rmon_set_alarm_sample_type	33
4.1.2.48	smi_rmon_set_alarm_start_up	33
4.1.2.49	smi_rmon_set_alarm_status	34
4.1.2.50	smi_rmon_set_alarm_variable	34
4.1.2.51	smi_rmon_set_event_active	34
4.1.2.52	smi_rmon_set_event_comm	35
4.1.2.53	smi_rmon_set_event_description	35
4.1.2.54	smi_rmon_set_event_index	35
4.1.2.55	smi_rmon_set_event_owner	36
4.1.2.56	smi_rmon_set_event_status	36
4.1.2.57	smi_rmon_snmp_get_ether_stats_status	37
4.1.2.58	smi_rmon_snmp_get_event_community	37
4.1.2.59	smi_rmon_snmp_get_event_description	37
4.1.2.60	smi_rmon_snmp_get_event_owner	38
4.1.2.61	smi_rmon_snmp_get_event_type	38
4.1.2.62	smi_rmon_snmp_set_ether_stats_status	39
4.1.2.63	smi_rmon_snmp_set_event_community	39
4.1.2.64	smi_rmon_snmp_set_event_description	39
4.1.2.65	smi_rmon_snmp_set_event_owner	40
4.1.2.66	smi_rmon_snmp_set_event_type	40
4.1.2.67	smi_rmon_stats_flush_all_port	41
4.1.2.68	smi_rmon_stats_flush_port	41
4.2	smi_rmon_msg.h File Reference	42
4.2.1	Detailed Description	43

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

smi_alarm_entry	5
smi_event_indices	6
smi_msg_rmon	7
smi_rmon_ifstats	8

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

smi_rmon.h (Remote network monitoring devices, often called monitors or probes, are instruments that exist for the purpose of managing a network. Often these remote probes are stand-alone devices and devote significant internal resources for the sole purpose of managing a network. An organization may employ many of these devices, one per network segment, to manage its internet. In addition, these devices may be used for a network management service provider to access a client network, often geographically remote)	9
smi_rmon_msg.h (Defines data structures used by Remote Monitoring SMI APIs)	42

Chapter 3

Data Structure Documentation

3.1 smi_alarm_entry Struct Reference

Data Fields

- `u_int32_t interval`
- `u_int32_t alarmSampleType`
- `st_int64_t risingValue`
- `st_int64_t fallingValue`
- `u_int32_t rising_event`
- `u_int32_t falling_event`
- `char ownername [SMI_RMON_OWNER_NAME_SIZE]`
- `char alarmVariableWord [SMI_RMON_ALARM_VAR_WORD_LENGTH+1]`
- `u_int8_t AlarmStartUpAlarm`

The documentation for this struct was generated from the following file:

- [smi_rmon_msg.h](#)

3.2 smi_event_indices Struct Reference

Data Fields

- `u_int32_t event_count`
- `u_int32_t event_last_index`
- `u_int32_t event_list [SMI_MAX_EVENT_NUM]`

The documentation for this struct was generated from the following file:

- [smi_rmon_msg.h](#)

3.3 smi_msg_rmon Struct Reference

Data Fields

- smi_cindex_t **cindex**
- smi_cindex_t **extended_cindex_1**
- char **if_name** [INTERFACE_NAMSIZ]
- u_int32_t **history_index**
- u_int32_t **history_interval**
- char **history_owner** [SMI_RMON_OWNER_NAME_SIZE+1]
- u_int32_t **alarm_interval**
- oid **alarm_oid** [MAX_OID_LEN]
- char **alarmVariableWord** [SMI_RMON_ALARM_VAR_WORD_LENGTH]
- u_int32_t **sample_type**
- u_int32_t **alarm_startup**
- s_int32_t **rising_threshold**
- s_int32_t **falling_threshold**
- u_int32_t **rising_event_indx**
- u_int32_t **falling_event_indx**
- char **alarmOwner** [SMI_RMON_OWNER_NAME_SIZE]
- u_int32_t **alarmIndex**
- u_int32_t **alarmStatus**
- u_int32_t **eventIndex**
- u_int32_t **eventStatus**
- char **eventCommunity** [SMI_RMON_COMM_LENGTH+1]
- char **eventDescription** [SMI_RMON_DESCR_LENGTH+1]
- char **eventOwner** [SMI_RMON_OWNER_NAME_SIZE+1]
- int **history_status**
- u_int32_t **index**
- u_int32_t **bucket**
- u_int32_t **if_index**
- u_int32_t **event_type**
- u_int32_t **ether_status**
- u_int32_t **snmp_event_type**
- u_int32_t **statsdata**
- char **comm** [SMI_RMON_COMM_LENGTH]
- char **descr** [SMI_RMON_DESCR_LENGTH]
- char **ownername** [SMI_RMON_OWNER_NAME_SIZE]
- u_int8_t **AlarmStartUpAlarm**
- u_int32_t **alarmSampleType**
- enum smi_rmon_stats_counter **rmon_if_counter**
- struct [smi_rmon_ifstats](#) **if_stats**
- struct [smi_alarm_entry](#) **alarmentry**
- struct [smi_event_indices](#) **event_indices**

The documentation for this struct was generated from the following file:

- [smi_rmon_msg.h](#)

3.4 smi_rmon_ifstats Struct Reference

Data Fields

- u_int32_t **ifindex**
- ut_int64_t **good_octets_rcv**
- ut_int64_t **bad_octets_rcv**
- ut_int64_t **mac_transmit_err**
- ut_int64_t **good_pkts_rcv**
- ut_int64_t **bad_pkts_rcv**
- ut_int64_t **brdc_pkts_rcv**
- ut_int64_t **mc_pkts_rcv**
- ut_int64_t **pkts_64_octets**
- ut_int64_t **pkts_65_127_octets**
- ut_int64_t **pkts_128_255_octets**
- ut_int64_t **pkts_256_511_octets**
- ut_int64_t **pkts_512_1023_octets**
- ut_int64_t **pkts_1024_max_octets**
- ut_int64_t **good_octets_sent**
- ut_int64_t **good_pkts_sent**
- ut_int64_t **excessive_collisions**
- ut_int64_t **mc_pkts_sent**
- ut_int64_t **brdc_pkts_sent**
- ut_int64_t **unrecog_mac_cntr_rcv**
- ut_int64_t **fc_sent**
- ut_int64_t **good_fc_rcv**
- ut_int64_t **drop_events**
- ut_int64_t **undersize_pkts**
- ut_int64_t **fragments_pkts**
- ut_int64_t **oversize_pkts**
- ut_int64_t **jabber_pkts**
- ut_int64_t **mac_rcv_error**
- ut_int64_t **bad_crc**
- ut_int64_t **collisions**
- ut_int64_t **late_collisions**
- ut_int64_t **bad_fc_rcv**

The documentation for this struct was generated from the following file:

- [smi_rmon_msg.h](#)

Chapter 4

File Documentation

4.1 smi_rmon.h File Reference

Remote network monitoring devices, often called monitors or probes, are instruments that exist for the purpose of managing a network. Often these remote probes are stand-alone devices and devote significant internal resources for the sole purpose of managing a network. An organization may employ many of these devices, one per network segment, to manage its internet. In addition, these devices may be used for a network management service provider to access a client network, often geographically remote.

Functions

- int [smi_rmon_get_if_stats](#) (struct smiclient_globals *azg, char *ifname, struct [smi_rmon_ifstats](#) *ifstats)

This function gets statistics measured by the probe for each monitored Ethernet interface.

- s_int32_t [smi_rmon_coll_stats_validate](#) (struct smiclient_globals *azg, u_int32_t index, char *ifname)

This function checks if the collection is already enabled on the interface.

- s_int32_t [smi_rmon_collection_stat_entry_add](#) (struct smiclient_globals *azg, char *ifname, u_int32_t index, char *ownername)

This function adds a collection statistics entry on an interface.

- s_int32_t [smi_rmon_collection_stat_entry_remove](#) (struct smiclient_globals *azg, char *ifname, u_int32_t index)

This function removes a collection statistics entry on an interface.

- s_int32_t [smi_rmon_coll_history_validate](#) (struct smiclient_globals *azg, u_int32_t index, char *ifname)

This function checks if the history control parameters are already set on this interface.

- s_int32_t [smi_rmon_coll_history_set_active](#) (struct smiclient_globals *azg, u_int32_t index)

This function sets the history control entry to active status.

- s_int32_t [smi_get_rmon_coll_history_status](#) (struct smiclient_globals *azg, u_int32_t index, int *stat)

This function gets the status of this historyControl entry.

- s_int32_t [smi_rmon_coll_history_set_inactive](#) (struct smiclient_globals *azg, u_int32_t index)

This function sets the history control entry to inactive status.

- s_int32_t [smi_rmon_coll_history_index_add_new](#) (struct smiclient_globals *azg, u_int32_t index, char *ifname)

This function sets the history control entry to inactive status.

- s_int32_t [smi_rmon_coll_history_datasource_set](#) (struct smiclient_globals *azg, u_int32_t index, char *ifname)

This function sets the history control entry to inactive status.

- s_int32_t [smi_rmon_coll_history_bucket_set](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t bucket, char *ifname)

This function sets the buckets requested for a history control entry on an interface.

- s_int32_t [smi_get_rmon_coll_history_bucket](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *bucket)

This function gets the buckets requested for a history control entry on an interface.

- s_int32_t [smi_rmon_coll_history_index_set](#) (struct smiclient_globals *azg, u_int32_t index, char *ifname)

This function adds a collection history control entry on an interface.

- s_int32_t [smi_get_rmon_coll_history_index](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *if_index)

This function gets an historical sample of Ethernet statistics on a particular Ethernet interface.

- s_int32_t [smi_rmon_event_type_set](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t event_type)

This function sets the type of the event entry.

- s_int32_t [smi_rmon_get_event_index](#) (struct smiclient_globals *azg, u_int32_t *event_indices)

This function gets an index that uniquely identifies an entry in the event table. Each such entry defines one event that is to be generated when the appropriate conditions occur.

- s_int32_t [smi_rmon_event_type_get](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *event_type)
This function gets the type of the event entry.
- s_int32_t [smi_rmon_snmp_set_event_type](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t snmp_event_type)
This function sets the type of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_get_event_type](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *snmp_event_type)
This function gets the type of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_set_event_community](#) (struct smiclient_globals *azg, u_int32_t index, char *comm)
This function sets the type of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_get_event_community](#) (struct smiclient_globals *azg, u_int32_t index, char *comm)
This function gets the type of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_set_event_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *ownername)
This function sets the owner name of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_get_event_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *ownername)
This function gets the owner name of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_set_ether_stats_status](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t ether_status)
This function sets the status of an entry in the etherStatsTable.
- s_int32_t [smi_rmon_snmp_get_ether_stats_status](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *ether_status)
This function gets the status of an entry in the etherStatsTable.
- s_int32_t [smi_rmon_snmp_set_event_description](#) (struct smiclient_globals *azg, u_int32_t index, char *descr)
This function sets the description of an entry in the eventTable.
- s_int32_t [smi_rmon_snmp_get_event_description](#) (struct smiclient_globals *azg, u_int32_t index, char *descr)
This function gets the description of an entry in the eventTable.
- s_int32_t [smi_rmon_get_if_counter](#) (struct smiclient_globals *azg, char *ifname, enum smi_rmon_stats_counter rmon_counter, u_int32_t *statsdata)

This function gets the statistics for Ethernet interfaces.

- int [smi_rmon_stats_flush_port](#) (struct smiclient_globals *azg, char *ifname)

This function clears the statistics(counter) for a specific interface.

- int [smi_rmon_stats_flush_all_port](#) (struct smiclient_globals *azg, u_int32_t index)

This function clears the statistics(counter) for all interface.

- int [smi_rmon_coll_history_interval_set](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t interval, char *ifname)

This function sets the interval of the history control entry on an interface.

- s_int32_t [smi_get_rmon_coll_history_interval](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *interval)

This function gets the interval of the history control entry on an interface.

- int [smi_rmon_coll_history_owner_set](#) (struct smiclient_globals *azg, u_int32_t index, char *ownerName, char *ifname)

This function sets the owner of the history control entry on an interface.

- int [smi_get_rmon_coll_history_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *ownerName)

This function gets the owner of the history control entry on an interface.

- s_int32_t [smi_rmon_coll_history_index_remove](#) (struct smiclient_globals *azg, u_int32_t index)

This function removes an entry from the history control table.

- s_int32_t [smi_rmon_set_alarm_interval](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t interval)

This function sets the alarm polling interval.

- s_int32_t [smi_get_rmon_alarm_interval](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *interval)

This function gets the alarm polling interval.

- s_int32_t [smi_rmon_set_alarm_variable](#) (struct smiclient_globals *azg, u_int32_t index, oid *oidname)

This function sets the variable of the alarm entry.

- s_int32_t [smi_get_rmon_alarm_variable](#) (struct smiclient_globals *azg, u_int32_t index, oid *name)

This function gets the variable of the alarm entry.

- s_int32_t [smi_rmon_set_alarm_sample_type](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t sample_type)

This function sets the sample type of the alarm entry.

- `s_int32_t smi_get_rmon_alarm_sample_type` (struct smiclient_globals *azg, u_int32_t index, u_int32_t *sample_type)

This function gets the sample type of the alarm entry.

- `s_int32_t smi_rmon_set_alarm_start_up` (struct smiclient_globals *azg, u_int32_t index, u_int32_t startup)

This function sets the alarm start-up type of the alarm entry.

- `s_int32_t smi_get_rmon_alarm_start_up` (struct smiclient_globals *azg, u_int32_t index, u_int32_t *startup)

This function gets the alarm start-up type of the alarm entry.

- `s_int32_t smi_rmon_set_alarm_rising_threshold` (struct smiclient_globals *azg, u_int32_t index, s_int32_t rising_th)

This function sets the rising threshold value of the alarm entry.

- `s_int32_t smi_get_rmon_alarm_rising_threshold` (struct smiclient_globals *azg, u_int32_t index, s_int32_t *rising_th)

This function gets the rising threshold value of the alarm entry.

- `s_int32_t smi_rmon_set_alarm_falling_threshold` (struct smiclient_globals *azg, u_int32_t index, s_int32_t falling_th)

This function sets the falling threshold value of the alarm entry.

- `s_int32_t smi_rmon_get_alarm_falling_threshold` (struct smiclient_globals *azg, u_int32_t index, s_int32_t *falling_th)

This function gets the falling threshold value of the alarm entry.

- `s_int32_t smi_rmon_set_alarm_rising_event_index` (struct smiclient_globals *azg, u_int32_t index, u_int32_t event_ix)

This function sets the event corresponding to crossing the rising threshold value of the alarm entry.

- `s_int32_t smi_get_rmon_alarm_rising_event_index` (struct smiclient_globals *azg, u_int32_t index, u_int32_t *event_ix)

This function gets the event corresponding to crossing the rising threshold value of the alarm entry.

- `s_int32_t smi_rmon_set_alarm_falling_event_index` (struct smiclient_globals *azg, u_int32_t index, u_int32_t event_ix)

This function sets the event corresponding to crossing the falling threshold value of the alarm entry.

- `s_int32_t smi_rmon_get_alarm_falling_event_index` (struct smiclient_globals *azg, u_int32_t index, u_int32_t *event_ix)

This function gets the event corresponding to crossing the falling threshold value of the alarm entry.

- s_int32_t [smi_rmon_set_alarm_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *owner)

This function sets the owner of the alarm.

- s_int32_t [smi_rmon_get_alarm_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *ownerName)

This function gets the owner of the alarm.

- s_int32_t [smi_rmon_set_alarm_entry](#) (struct smiclient_globals *azg, u_int32_t alarm_index, char *variable, u_int32_t interval, u_int32_t alarmSampleType, s_int32_t rising_value, s_int32_t falling_value, u_int32_t rising_event, u_int32_t falling_event, char *owner_name, u_int8_t AlarmStartUpAlarm)

This function sets a list of parameters that set up a periodic checking for alarm conditions.

- s_int32_t [smi_rmon_get_alarm_entry](#) (struct smiclient_globals *azg, u_int32_t index, struct [smi_alarm_entry](#) *alarm_entry)

This function sets a list of parameters that set up a periodic checking for alarm conditions.

- s_int32_t [smi_rmon_alarm_index_remove](#) (struct smiclient_globals *azg, u_int32_t index)

This function removes an alarm entry.

- int32_t [smi_rmon_set_alarm_status](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t status)

This function sets the status of the alarm entry.

- s_int32_t [smi_rmon_event_index_remove](#) (struct smiclient_globals *azg, u_int32_t index)

This function removes an event entry.

- s_int32_t [smi_rmon_set_event_index](#) (struct smiclient_globals *azg, u_int32_t index)

This function sets an index that uniquely identifies an entry in the event table. Each such entry defines one event that is to be generated when the appropriate conditions occur.

- s_int32_t [smi_rmon_set_event_active](#) (struct smiclient_globals *azg, u_int32_t index)

This function activates the event once certain condition are met.

- s_int32_t [smi_rmon_set_event_status](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t status)

This function sets the status of event entry.

- s_int32_t [smi_rmon_get_event_status](#) (struct smiclient_globals *azg, u_int32_t index, u_int32_t *status)

This function gets the status of event entry.

- s_int32_t [smi_rmon_set_event_comm](#) (struct smiclient_globals *azg, u_int32_t index, char *community)

This function sets the community of event entry. If an SNMP trap is to be sent, it will be sent to the SNMP community specified by this octet string.

- s_int32_t [smi_rmon_get_event_comm](#) (struct smiclient_globals *azg, u_int32_t index, char *community)

This function gets the community of event entry. If an SNMP trap is to be sent, it will be sent to the SNMP community specified by this octet string.

- s_int32_t [smi_rmon_set_event_description](#) (struct smiclient_globals *azg, u_int32_t index, char *description)

This function sets a comment describing this event entry.

- s_int32_t [smi_rmon_get_event_description](#) (struct smiclient_globals *azg, u_int32_t index, char *description)

This function gets a comment describing this event entry.

- s_int32_t [smi_rmon_set_event_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *owner)

This function sets the entity that configured event entry and is therefore using the resources assigned to it.

- s_int32_t [smi_rmon_get_event_owner](#) (struct smiclient_globals *azg, u_int32_t index, char *owner)

This function gets the entity that configured event entry and is therefore using the resources assigned to it.

4.1.1 Detailed Description

Remote network monitoring devices, often called monitors or probes, are instruments that exist for the purpose of managing a network. Often these remote probes are stand-alone devices and devote significant internal resources for the sole purpose of managing a network. An organization may employ many of these devices, one per network segment, to manage its internet. In addition, these devices may be used for a network management service provider to access a client network, often geographically remote.

4.1.2 Function Documentation

4.1.2.1 `s_int32_t smi_get_rmon_alarm_interval (struct smiclient_globals * azg, u_int32_t index, u_int32_t * interval)`

This function gets the alarm polling interval. `smi_get_rmon_alarm_interval`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *interval* Polling interval

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.2 `s_int32_t smi_get_rmon_alarm_rising_event_index (struct smiclient_globals * azg, u_int32_t index, u_int32_t * event_ix)`

This function gets the event corresponding to crossing the rising threshold value of the alarm entry. `smi_get_rmon_alarm_rising_event_index`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *event_ix* Rising event value

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.3 `s_int32_t smi_get_rmon_alarm_rising_threshold (struct smiclient_globals * azg, u_int32_t index, s_int32_t * rising_th)`

This function gets the rising threshold value of the alarm entry. `smi_get_rmon_alarm_rising_threshold`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index

→ *rising_th* Rising threshold value

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.4 s_int32_t smi_get_rmon_alarm_sample_type (struct smiclient_globals * azg, u_int32_t index, u_int32_t * sample_type)

This function gets the sample type of the alarm entry. smi_get_rmon_alarm_sample_type

Parameters:

← *azg* Pointer to the SMI client global structure
← *index* Alarm entry index
→ *sample_type* Alarm sample type (Absolute -1, Delta -2)

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.5 s_int32_t smi_get_rmon_alarm_start_up (struct smiclient_globals * azg, u_int32_t index, u_int32_t * startup)

This function gets the alarm start-up type of the alarm entry. smi_get_rmon_alarm_start_up

Parameters:

← *azg* Pointer to the SMI client global structure
← *index* Alarm entry index
→ *startup* Start-up alarm type (rising alarm -1, falling alarm - 2, rising or falling alarm - 3)

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.6 `s_int32_t smi_get_rmon_alarm_variable (struct smiclient_globals * azg, u_int32_t index, oid * name)`

This function gets the variable of the alarm entry. `smi_get_rmon_alarm_variable`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *name* Variable object identifier (OID)

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.7 `s_int32_t smi_get_rmon_coll_history_bucket (struct smiclient_globals * azg, u_int32_t index, u_int32_t * bucket)`

This function gets the buckets requested for a history control entry on an interface. `smi_get_rmon_coll_history_bucket`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- *bucket* Number of buckets

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.8 `s_int32_t smi_get_rmon_coll_history_index (struct smiclient_globals * azg, u_int32_t index, u_int32_t * if_index)`

This function gets an historical sample of Ethernet statistics on a particular Ethernet interface. `smi_get_rmon_coll_history_index`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- *if_index* The interface index

Returns:

RESULT_OK on success, otherwise one of the following error codes
 RESULT_ERROR

4.1.2.9 s_int32_t smi_get_rmon_coll_history_interval (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t * *interval*)

This function gets the interval of the history control entry on an interface. smi_get_rmon_coll_history_interval

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- *interval* Polling interval

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.10 int smi_get_rmon_coll_history_owner (struct smiclient_globals * *azg*, u_int32_t *index*, char * *ownerName*)

This function gets the owner of the history control entry on an interface. smi_get_rmon_coll_history_owner

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- *ownername* Owner name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.11 s_int32_t smi_get_rmon_coll_history_status (struct smiclient_globals * *azg*, u_int32_t *index*, int * *stat*)

This function gets the status of this historyControl entry. smi_get_rmon_coll_history_status

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- *stat* Status record

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.12 `s_int32_t smi_rmon_alarm_index_remove (struct smiclient_globals * azg, u_int32_t index)`

This function removes an alarm entry. `smi_rmon_alarm_index_remove`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.13 `s_int32_t smi_rmon_coll_history_bucket_set (struct smiclient_globals * azg, u_int32_t index, u_int32_t bucket, char * ifname)`

This function sets the buckets requested for a history control entry on an interface. `smi_rmon_coll_history_bucket_set`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *bucket* Number of buckets
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.14 `s_int32_t smi_rmon_coll_history_datasource_set (struct smiclient_globals * azg, u_int32_t index, char * ifname)`

This function sets the history control entry to inactive status. `smi_rmon_coll_history_datasource_set`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.15 s_int32_t smi_rmon_coll_history_index_add_new (struct smiclient_globals * *azg*, u_int32_t *index*, char * *ifname*)

This function sets the history control entry to inactive status. smi_rmon_coll_history_index_add_new

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.16 s_int32_t smi_rmon_coll_history_index_remove (struct smiclient_globals * *azg*, u_int32_t *index*)

This function removes an entry from the history control table. smi_rmon_coll_history_index_remove

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.17 s_int32_t smi_rmon_coll_history_index_set (struct smiclient_globals * *azg*, u_int32_t *index*, char * *ifname*)

This function adds a collection history control entry on an interface. smi_rmon_coll_history_index_set

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.18 `int smi_rmon_coll_history_interval_set (struct smiclient_globals * azg, u_int32_t index, u_int32_t interval, char * ifname)`

This function sets the interval of the history control entry on an interface. `smi_rmon_coll_history_interval_set`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *interval* Polling interval
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.19 `int smi_rmon_coll_history_owner_set (struct smiclient_globals * azg, u_int32_t index, char * ownerName, char * ifname)`

This function sets the owner of the history control entry on an interface. `smi_rmon_coll_history_owner_set`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index
- ← *ownername* Owner name
- ← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.20 `s_int32_t smi_rmon_coll_history_set_active (struct smiclient_globals * azg, u_int32_t index)`

This function sets the history control entry to active status. `smi_rmon_coll_history_set_active`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* History control entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.21 s_int32_t smi_rmon_coll_history_set_inactive (struct smiclient_globals * azg, u_int32_t index)

This function sets the history control entry to inactive status. smi_rmon_coll_history_set_inactive

Parameters:

← *azg* Pointer to the SMI client global structure
← *index* History control entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.22 s_int32_t smi_rmon_coll_history_validate (struct smiclient_globals * azg, u_int32_t index, char * ifname)

This function checks if the history control parameters are already set on this interface. smi_rmon_coll_history_validate

Parameters:

← *azg* Pointer to the SMI client global structure
← *index* History control entry index
← *ifname* The Interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.23 s_int32_t smi_rmon_coll_stats_validate (struct smiclient_globals * azg, u_int32_t index, char * ifname)

This function checks if the collection is already enabled on the interface. smi_rmon_coll_stats_validate

Parameters:

← *azg* Pointer to the SMI client global structure

← *index* Etherstats entry index

← *ifname* The interface name

Returns:

0 on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.24 s_int32_t smi_rmon_collection_stat_entry_add (struct smiclient_globals * azg, char * ifname, u_int32_t index, char * ownername)

This function adds a collection statistics entry on an interface. smi_rmon_collection_stat_entry_add

Parameters:

← *azg* Pointer to the SMI client global structure

← *ifname* The Interface name

← *index* Etherstats entry index

← *ownername* Owner name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.25 s_int32_t smi_rmon_collection_stat_entry_remove (struct smiclient_globals * azg, char * ifname, u_int32_t index)

This function removes a collection statistics entry on an interface. smi_rmon_collection_stat_entry_remove

Parameters:

← *azg* Pointer to the SMI client global structure

← *ifname* The Interface name

← *index* Etherstats entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.26 `s_int32_t smi_rmon_event_index_remove (struct smiclient_globals *
azg, u_int32_t index)`

This function removes an event entry. `smi_rmon_event_index_remove`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.27 `s_int32_t smi_rmon_event_type_get (struct smiclient_globals * azg,
u_int32_t index, u_int32_t * event_type)`

This function gets the type of the event entry. `smi_rmon_event_type_get`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- *event_type* Type:
 - None (1)
 - Log (2)
 - SNMP trap (3)
 - Log and trap (4)

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error
codes
RMON_API_GET_FAILURE

4.1.2.28 `s_int32_t smi_rmon_event_type_set (struct smiclient_globals * azg,
u_int32_t index, u_int32_t event_type)`

This function sets the type of the event entry. `smi_rmon_event_type_set`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *event_type* Type:

- None (1)
- Log (2)
- SNMP trap (3)
- Log and trap (4)

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.29 `s_int32_t smi_rmon_get_alarm_entry (struct smiclient_globals * azg, u_int32_t index, struct smi_alarm_entry * alarm_entry)`

This function sets a list of parameters that set up a periodic checking for alarm conditions. `smi_rmon_get_alarm_entry`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *alarm_entry* List of Alarm parameters : \ ref struct [smi_alarm_entry](#)

Returns:

RESULT_OK on success, otherwise one of the following error codes
 RESULT_ERROR
 RMON_API_SET_FAILURE

4.1.2.30 `s_int32_t smi_rmon_get_alarm_falling_event_index (struct smiclient_globals * azg, u_int32_t index, u_int32_t * event_ix)`

This function gets the event corresponding to crossing the falling threshold value of the alarm entry. `smi_rmon_get_alarm_falling_event_index`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *event_ix* Falling event value

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.31 s_int32_t smi_rmon_get_alarm_falling_threshold (struct smiclient_globals * azg, u_int32_t index, s_int32_t * falling_th)

This function gets the falling threshold value of the alarm entry. smi_rmon_get_alarm_falling_threshold

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *falling_th* Falling threshold value

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.32 s_int32_t smi_rmon_get_alarm_owner (struct smiclient_globals * azg, u_int32_t index, char * ownerName)

This function gets the owner of the alarm. smi_rmon_get_alarm_owner

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- *ownerName* Owner name

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.33 s_int32_t smi_rmon_get_event_comm (struct smiclient_globals * azg, u_int32_t index, char * community)

This function gets the community of event entry. If an SNMP trap is to be sent, it will be sent to the SNMP community specified by this octet string. smi_rmon_get_event_comm

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- *community* Event entry community

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.34 `s_int32_t smi_rmon_get_event_description (struct smiclient_globals * azg, u_int32_t index, char * description)`

This function gets a comment describing this event entry. `smi_rmon_get_event_description`

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Event entry index
 → *description* Event entry description

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.35 `s_int32_t smi_rmon_get_event_index (struct smiclient_globals * azg, u_int32_t * event_indices)`

This function gets an index that uniquely identifies an entry in the event table. Each such entry defines one event that is to be generated when the appropriate conditions occur. `smi_rmon_get_event_index`

Parameters:

← *azg* Pointer to the SMI client global structure
 → *event_indices* The event index

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.36 `s_int32_t smi_rmon_get_event_owner (struct smiclient_globals * azg, u_int32_t index, char * owner)`

This function gets the entity that configured event entry and is therefore using the resources assigned to it. `smi_rmon_get_event_owner`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- *owner* Owner name

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.37 s_int32_t smi_rmon_get_event_status (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t * *status*)

This function gets the status of event entry. smi_rmon_get_event_status

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- *status* Event entry status : VALID_STATUS 1
 CREATE_REQ_STATUS 2
 UNDER_CREATION_STATUS 3
 INVALID_STATUS 4

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.38 s_int32_t smi_rmon_get_if_counter (struct smiclient_globals * *azg*, char * *ifname*, enum smi_rmon_stats_counter *rmon_counter*, u_int32_t * *statsdata*)

This function gets the statistics for Ethernet interfaces. smi_rmon_get_if_counter

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *ifname* The interface name
- ← *rmon_counter* Counters for statistics measurement
- *statsdata* Statistic data

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.39 `int smi_rmon_get_if_stats (struct smiclient_globals * azg, char * ifname, struct smi_rmon_ifstats * ifstats)`

This function gets statistics measured by the probe for each monitored Ethernet interface. `smi_rmon_get_if_stats`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *ifname* The interface name
- *ifstats* Statistics measured for monitored Ethernet interface

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.40 `s_int32_t smi_rmon_set_alarm_entry (struct smiclient_globals * azg, u_int32_t alarm_index, char * variable, u_int32_t interval, u_int32_t alarmSampleType, s_int32_t rising_value, s_int32_t falling_value, u_int32_t rising_event, u_int32_t falling_event, char * owner_name, u_int8_t AlarmStartupAlarm)`

This function sets a list of parameters that set up a periodic checking for alarm conditions. `smi_rmon_set_alarm_entry`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *alarm_index* Alarm entry index
- ← *variable* Alarm variable
- ← *interval* Polling interval
- ← *alarmSampleType* Alarm Sample Type
- ← *rising_value* Value of the statistic during the rising sampling period
- ← *falling_value* Value of the statistic during the falling sampling period
- ← *rising_event* Statistic during the rising sampling period
- ← *falling_event* Statistic during the falling sampling period
- ← *ownerName* Owner name
- ← *AlarmStartupAlarm* Alarm Startup Alarm

Returns:

RESULT_OK on success, otherwise one of the following error codes
 RESULT_ERROR
 RMON_API_SET_FAILURE

4.1.2.41 `s_int32_t smi_rmon_set_alarm_falling_event_index (struct smiclient_globals * azg, u_int32_t index, u_int32_t event_ix)`

This function sets the event corresponding to crossing the falling threshold value of the alarm entry. `smi_rmon_set_alarm_falling_event_index`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *event_ix* Falling event value

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.42 `s_int32_t smi_rmon_set_alarm_falling_threshold (struct smiclient_globals * azg, u_int32_t index, s_int32_t falling_th)`

This function sets the falling threshold value of the alarm entry. `smi_rmon_set_alarm_falling_threshold`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *falling_th* Falling threshold value

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.43 `s_int32_t smi_rmon_set_alarm_interval (struct smiclient_globals * azg, u_int32_t index, u_int32_t interval)`

This function sets the alarm polling interval. `smi_rmon_set_alarm_interval`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *interval* Polling interval

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.44 `s_int32_t smi_rmon_set_alarm_owner (struct smiclient_globals * azg, u_int32_t index, char * owner)`

This function sets the owner of the alarm. `smi_rmon_set_alarm_owner`

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Alarm entry index
 ← *owner* Owner

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.45 `s_int32_t smi_rmon_set_alarm_rising_event_index (struct smiclient_globals * azg, u_int32_t index, u_int32_t event_ix)`

This function sets the event corresponding to crossing the rising threshold value of the alarm entry. `smi_rmon_set_alarm_rising_event_index`

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Alarm entry index
 ← *event_ix* Rising event value

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.46 `s_int32_t smi_rmon_set_alarm_rising_threshold (struct smiclient_globals * azg, u_int32_t index, s_int32_t rising_th)`

This function sets the rising threshold value of the alarm entry. `smi_rmon_set_alarm_rising_threshold`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *rising_th* Rising threshold value

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.47 s_int32_t smi_rmon_set_alarm_sample_type (struct smiclient_globals * azg, u_int32_t index, u_int32_t sample_type)

This function sets the sample type of the alarm entry. smi_rmon_set_alarm_sample_type

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *sample_type* Alarm sample type (Absolute -1, Delta -2)

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.48 s_int32_t smi_rmon_set_alarm_start_up (struct smiclient_globals * azg, u_int32_t index, u_int32_t startup)

This function sets the alarm start-up type of the alarm entry. smi_rmon_set_alarm_start_up

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *startup* Start-up alarm type (rising alarm -1, falling alarm - 2, rising or falling alarm - 3)

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.49 `int32_t smi_rmon_set_alarm_status (struct smiclient_globals * azg, u_int32_t index, u_int32_t status)`

This function sets the status of the alarm entry. `smi_rmon_set_alarm_status`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *status* Alarm entry status

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.50 `s_int32_t smi_rmon_set_alarm_variable (struct smiclient_globals * azg, u_int32_t index, oid * oidname)`

This function sets the variable of the alarm entry. `smi_rmon_set_alarm_variable`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Alarm entry index
- ← *oidname* Variable object identifier (OID)

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error
codes
RMON_API_SET_FAILURE

4.1.2.51 `s_int32_t smi_rmon_set_event_active (struct smiclient_globals * azg, u_int32_t index)`

This function activates the event once certain condition are met. `smi_rmon_set_event_active`

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error
codes
RMON_API_SET_FAILURE

**4.1.2.52 s_int32_t smi_rmon_set_event_comm (struct smiclient_globals * *azg*,
u_int32_t *index*, char * *community*)**

This function sets the community of event entry. If an SNMP trap is to be sent, it will be sent to the SNMP community specified by this octet string. smi_rmon_set_event_comm

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *community* Event entry community

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

**4.1.2.53 s_int32_t smi_rmon_set_event_description (struct smiclient_globals *
azg, u_int32_t *index*, char * *description*)**

This function sets a comment describing this event entry. smi_rmon_set_event_description

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *description* Event entry description

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

**4.1.2.54 s_int32_t smi_rmon_set_event_index (struct smiclient_globals * *azg*,
u_int32_t *index*)**

This function sets an index that uniquely identifies an entry in the event table. Each such entry defines one event that is to be generated when the appropriate conditions occur. smi_rmon_set_event_index

Parameters:

- ← *azg* Pointer to the SMI client global structure

← *index* Event entry index

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

**4.1.2.55 s_int32_t smi_rmon_set_event_owner (struct smiclient_globals * *azg*,
 u_int32_t *index*, char * *owner*)**

This function sets the entity that configured event entry and is therefore using the resources assigned to it. smi_rmon_set_event_owner

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Event entry index
 ← *owner* Owner name

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

**4.1.2.56 s_int32_t smi_rmon_set_event_status (struct smiclient_globals * *azg*,
 u_int32_t *index*, u_int32_t *status*)**

This function sets the status of event entry. smi_rmon_set_event_status

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Event entry index
 ← *status* Event entry status

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.57 s_int32_t smi_rmon_snmp_get_ether_stats_status (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t * *ether_status*)

This function gets the status of an entry in the etherStatsTable. smi_rmon_snmp_get_ether_stats_status

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Status entry index
- *ether_status* Ethernet Status

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.58 s_int32_t smi_rmon_snmp_get_event_community (struct smiclient_globals * *azg*, u_int32_t *index*, char * *comm*)

This function gets the type of an entry in the eventTable. smi_rmon_snmp_get_event_community

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- *comm* Community name

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
RMON_API_GET_FAILURE

4.1.2.59 s_int32_t smi_rmon_snmp_get_event_description (struct smiclient_globals * *azg*, u_int32_t *index*, char * *descr*)

This function gets the description of an entry in the eventTable. smi_rmon_snmp_get_event_description

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Status entry index
- *descr* Event description

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.60 s_int32_t smi_rmon_snmp_get_event_owner (struct smiclient_globals * *azg*, u_int32_t *index*, char * *ownername*)

This function gets the owner name of an entry in the eventTable. smi_rmon_snmp_get_event_owner

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Event entry index
 → *ownername* Owner name

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.61 s_int32_t smi_rmon_snmp_get_event_type (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t * *snmp_event_type*)

This function gets the type of an entry in the eventTable. smi_rmon_snmp_get_event_type

Parameters:

← *azg* Pointer to the SMI client global structure
 ← *index* Event entry index
 → *snmp_event_type* Type of event:

- None (1)
- Log (2)
- SNMP trap (3)
- Log and trap (4)

Returns:

RMON_API_GET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_GET_FAILURE

4.1.2.62 s_int32_t smi_rmon_snmp_set_ether_stats_status (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t *ether_status*)

This function sets the status of an entry in the etherStatsTable. smi_rmon_snmp_set_ether_stats_status

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Status entry index
- ← *ether_status* Ethernet Status

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.63 s_int32_t smi_rmon_snmp_set_event_community (struct smiclient_globals * *azg*, u_int32_t *index*, char * *comm*)

This function sets the type of an entry in the eventTable. smi_rmon_snmp_set_event_community

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *comm* Community name

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
RMON_API_SET_FAILURE

4.1.2.64 s_int32_t smi_rmon_snmp_set_event_description (struct smiclient_globals * *azg*, u_int32_t *index*, char * *descr*)

This function sets the description of an entry in the eventTable. smi_rmon_snmp_set_event_description

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Status entry index
- ← *descr* Event description

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.65 s_int32_t smi_rmon_snmp_set_event_owner (struct smiclient_globals * *azg*, u_int32_t *index*, char * *ownername*)

This function sets the owner name of an entry in the eventTable. smi_rmon_snmp_set_event_owner

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *ownername* Owner name

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.66 s_int32_t smi_rmon_snmp_set_event_type (struct smiclient_globals * *azg*, u_int32_t *index*, u_int32_t *snmp_event_type*)

This function sets the type of an entry in the eventTable. smi_rmon_snmp_set_event_type

Parameters:

- ← *azg* Pointer to the SMI client global structure
- ← *index* Event entry index
- ← *snmp_event_type* Type of event:
 - None (1)
 - Log (2)
 - SNMP trap (3)
 - Log and trap (4)

Returns:

RMON_API_SET_SUCCESS on success, otherwise one of the following error codes
 RMON_API_SET_FAILURE

4.1.2.67 `int smi_rmon_stats_flush_all_port (struct smiclient_globals * azg,
u_int32_t index)`

This function clears the statistics(counter) for all interface. `smi_rmon_stats_flush_all_port`

Parameters:

← *azg* Pointer to the SMI client global structure

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.1.2.68 `int smi_rmon_stats_flush_port (struct smiclient_globals * azg, char *
ifname)`

This function clears the statistics(counter) for a specific interface. `smi_rmon_stats_flush_port`

Parameters:

← *azg* Pointer to the SMI client global structure

← *ifname* The interface name

Returns:

RESULT_OK on success, otherwise one of the following error codes
RESULT_ERROR

4.2 smi_rmon_msg.h File Reference

Defines data structures used by Remote Monitoring SMI APIs. #include "smi_message.h"

```
#include "asn1.h"
```

Data Structures

- struct [smi_rmon_ifstats](#)
- struct [smi_alarm_entry](#)
- struct [smi_event_indices](#)
- struct [smi_msg_rmon](#)

Defines

- #define SMI_MSG_RMON_SIZE 4
- #define SMI_RMON_OWNER_NAME_SIZE 127
- #define ALARM_OID_SIZE 10
- #define SMI_RMON_COMM_LENGTH 127
- #define SMI_RMON_DESCR_LENGTH 255
- #define SMI_MAX_EVENT_NUM 5
- #define SMI_RMON_INDEX_MIN 1
- #define SMI_RMON_INDEX_MAX 65535
- #define SMI_RMON_POLLINTVAL_MIN 1
- #define SMI_RMON_POLLINTVAL_MAX 3600
- #define SMI_RMON_ALARM_AB 1
- #define SMI_RMON_ALARM_DELTA 2
- #define SMI_RMON_ALARM_RISING 1
- #define SMI_RMON_ALARM_FALLING 2
- #define SMI_RMON_ALARM_RISORFALL 3
- #define SMI_RMON_EVENTTYPE_MAX 3
- #define SMI_RMON_VALID_STATUS 1
- #define SMI_RMON_INVALID_STATUS 4
- #define SMI_RMON_CTYPE_HISTORYINDEX 0
- #define SMI_RMON_CTYPE_HISTORYINTERVAL 1
- #define SMI_RMON_CTYPE_HISTORYOWNER 2
- #define SMI_RMON_CTYPE_IFNAME 3
- #define SMI_RMON_CTYPE_ALARMINTERVAL 4
- #define SMI_RMON_CTYPE_ALARMROID 5
- #define SMI_RMON_CTYPE_SAMPLETYPE 6
- #define SMI_RMON_CTYPE_ALARMSTARTUP 7
- #define SMI_RMON_CTYPE_RISINGTHRESHOLD 8
- #define SMI_RMON_CTYPE_FALLINGTHRESHOLD 9
- #define SMI_RMON_CTYPE_RISINGEVTINDX 10
- #define SMI_RMON_CTYPE_FALLINGEVTINDX 11

- #define **SMI_RMON_CTYPE_ALARMOWNER** 12
- #define **SMI_RMON_CTYPE_ALARMVARIABLEWORD** 13
- #define **SMI_RMON_CTYPE_ALARMINDEX** 14
- #define **SMI_RMON_CTYPE_EVENTINDEX** 15
- #define **SMI_RMON_CTYPE_EVENTSTATUS** 16
- #define **SMI_RMON_CTYPE_EVENTCOMM** 17
- #define **SMI_RMON_CTYPE_EVENTDESCRIPT** 18
- #define **SMI_RMON_CTYPE_EVENTOWNER** 19
- #define **SMI_RMON_CTYPE_INDEX** 20
- #define **SMI_RMON_CTYPE_IFINDEX** 21
- #define **SMI_RMON_CTYPE_ADDSTAT** 22
- #define **SMI_RMON_CTYPE_STATUS** 23
- #define **SMI_RMON_CTYPE_BUCKET** 24
- #define **SMI_RMON_CTYPE_EVENTTYPE** 25
- #define **SMI_RMON_CTYPE_SNMPEVTYPE** 26
- #define **SMI_RMON_CTYPE_SCOMMUNITY** 27
- #define **SMI_RMON_CTYPE_SEVENTOWNER** 28
- #define **SMI_RMON_CTYPE_ETHERSTATUS** 29
- #define **SMI_RMON_CTYPE_DESCRIPTION** 30
- #define **SMI_RMON_CTYPE_EXTENDED_TYPE** 31
- #define **SMI_RMON_CTYPE_IFSTATS** 0
- #define **SMI_RMON_CTYPE_IFCOUNTER** 1
- #define **SMI_RMON_CTYPE_COUNTERDATA** 2
- #define **SMI_RMON_CTYPE_ALARMSTRUCT** 3
- #define **SMI_RMON_CTYPE_ALARM_STATUS** 4
- #define **SMI_RMON_CTYPE_EVENTINDICES** 5
- #define **SMI_RMON_CTYPE_ALARMSTARTUPALARM** 6
- #define **SMI_RMON_CTYPE_ALARMSAMPLETYPE** 7

Enumerations

- enum **smi_rmon_stats_counter** { **SMI_COUNTER** }

Functions

- void **smi_rmon_dump** (struct lib_globals *zg, struct [smi_msg_rmon](#) *msg)
- int **smi_encode_rmonmsg** (u_char **pnt, u_int16_t *size, struct [smi_msg_rmon](#) *msg)
- int **smi_decode_rmonmsg** (u_char **pnt, u_int16_t *size, struct [smi_msg_rmon](#) *msg)
- int **smi_parse_rmon** (u_char **pnt, u_int16_t *size, struct smi_msg_header *header, void *arg, SMI_CALLBACK callback)

4.2.1 Detailed Description

Defines data structures used by Remote Monitoring SMI APIs.

Index

smi_alarm_entry, [5](#)
smi_event_indices, [6](#)
smi_get_rmon_alarm_interval
 smi_rmon.h, [16](#)
smi_get_rmon_alarm_rising_event_index
 smi_rmon.h, [16](#)
smi_get_rmon_alarm_rising_threshold
 smi_rmon.h, [16](#)
smi_get_rmon_alarm_sample_type
 smi_rmon.h, [17](#)
smi_get_rmon_alarm_start_up
 smi_rmon.h, [17](#)
smi_get_rmon_alarm_variable
 smi_rmon.h, [17](#)
smi_get_rmon_coll_history_bucket
 smi_rmon.h, [18](#)
smi_get_rmon_coll_history_index
 smi_rmon.h, [18](#)
smi_get_rmon_coll_history_interval
 smi_rmon.h, [18](#)
smi_get_rmon_coll_history_owner
 smi_rmon.h, [19](#)
smi_get_rmon_coll_history_status
 smi_rmon.h, [19](#)
smi_msg_rmon, [7](#)
smi_rmon.h, [9](#)
 smi_get_rmon_alarm_interval, [16](#)
 smi_get_rmon_alarm_rising_
 event_index, [16](#)
 smi_get_rmon_alarm_rising_
 threshold, [16](#)
 smi_get_rmon_alarm_sample_type,
 [17](#)
 smi_get_rmon_alarm_start_up, [17](#)
 smi_get_rmon_alarm_variable, [17](#)
 smi_get_rmon_coll_history_bucket,
 [18](#)
 smi_get_rmon_coll_history_index,
 [18](#)
 smi_get_rmon_coll_history_
 interval, [18](#)
 smi_get_rmon_coll_history_owner,
 [19](#)
 smi_get_rmon_coll_history_status,
 [19](#)
 smi_rmon_alarm_index_remove, [19](#)
 smi_rmon_coll_history_bucket_set,
 [20](#)
 smi_rmon_coll_history_
 datasource_set, [20](#)
 smi_rmon_coll_history_index_
 add_new, [20](#)
 smi_rmon_coll_history_index_
 remove, [21](#)
 smi_rmon_coll_history_index_set,
 [21](#)
 smi_rmon_coll_history_interval_set,
 [21](#)
 smi_rmon_coll_history_owner_set,
 [22](#)
 smi_rmon_coll_history_set_active,
 [22](#)
 smi_rmon_coll_history_set_
 inactive, [23](#)
 smi_rmon_coll_history_validate, [23](#)
 smi_rmon_coll_stats_validate, [23](#)
 smi_rmon_collection_stat_entry_
 add, [24](#)
 smi_rmon_collection_stat_entry_
 remove, [24](#)
 smi_rmon_event_index_remove, [24](#)
 smi_rmon_event_type_get, [25](#)
 smi_rmon_event_type_set, [25](#)
 smi_rmon_get_alarm_entry, [26](#)
 smi_rmon_get_alarm_falling_
 event_index, [26](#)
 smi_rmon_get_alarm_falling_
 threshold, [26](#)
 smi_rmon_get_alarm_owner, [27](#)
 smi_rmon_get_event_comm, [27](#)
 smi_rmon_get_event_description,
 [28](#)

- smi_rmon_get_event_index, [28](#)
- smi_rmon_get_event_owner, [28](#)
- smi_rmon_get_event_status, [29](#)
- smi_rmon_get_if_counter, [29](#)
- smi_rmon_get_if_stats, [29](#)
- smi_rmon_set_alarm_entry, [30](#)
- smi_rmon_set_alarm_falling_event_index, [30](#)
- smi_rmon_set_alarm_falling_threshold, [31](#)
- smi_rmon_set_alarm_interval, [31](#)
- smi_rmon_set_alarm_owner, [32](#)
- smi_rmon_set_alarm_rising_event_index, [32](#)
- smi_rmon_set_alarm_rising_threshold, [32](#)
- smi_rmon_set_alarm_sample_type, [33](#)
- smi_rmon_set_alarm_start_up, [33](#)
- smi_rmon_set_alarm_status, [33](#)
- smi_rmon_set_alarm_variable, [34](#)
- smi_rmon_set_event_active, [34](#)
- smi_rmon_set_event_comm, [34](#)
- smi_rmon_set_event_description, [35](#)
- smi_rmon_set_event_index, [35](#)
- smi_rmon_set_event_owner, [36](#)
- smi_rmon_set_event_status, [36](#)
- smi_rmon_snmp_get_ether_stats_status, [36](#)
- smi_rmon_snmp_get_event_community, [37](#)
- smi_rmon_snmp_get_event_description, [37](#)
- smi_rmon_snmp_get_event_owner, [38](#)
- smi_rmon_snmp_get_event_type, [38](#)
- smi_rmon_snmp_set_ether_stats_status, [38](#)
- smi_rmon_snmp_set_event_community, [39](#)
- smi_rmon_snmp_set_event_description, [39](#)
- smi_rmon_snmp_set_event_owner, [40](#)
- smi_rmon_snmp_set_event_type, [40](#)
- smi_rmon_stats_flush_all_port, [40](#)
- smi_rmon_stats_flush_port, [41](#)
- smi_rmon_alarm_index_remove
 - smi_rmon.h, [19](#)
- smi_rmon_coll_history_bucket_set
 - smi_rmon.h, [20](#)
- smi_rmon_coll_history_datasource_set
 - smi_rmon.h, [20](#)
- smi_rmon_coll_history_index_add_new
 - smi_rmon.h, [20](#)
- smi_rmon_coll_history_index_remove
 - smi_rmon.h, [21](#)
- smi_rmon_coll_history_index_set
 - smi_rmon.h, [21](#)
- smi_rmon_coll_history_interval_set
 - smi_rmon.h, [21](#)
- smi_rmon_coll_history_owner_set
 - smi_rmon.h, [22](#)
- smi_rmon_coll_history_set_active
 - smi_rmon.h, [22](#)
- smi_rmon_coll_history_set_inactive
 - smi_rmon.h, [23](#)
- smi_rmon_coll_history_validate
 - smi_rmon.h, [23](#)
- smi_rmon_coll_stats_validate
 - smi_rmon.h, [23](#)
- smi_rmon_collection_stat_entry_add
 - smi_rmon.h, [24](#)
- smi_rmon_collection_stat_entry_remove
 - smi_rmon.h, [24](#)
- smi_rmon_event_index_remove
 - smi_rmon.h, [24](#)
- smi_rmon_event_type_get
 - smi_rmon.h, [25](#)
- smi_rmon_event_type_set
 - smi_rmon.h, [25](#)
- smi_rmon_get_alarm_entry
 - smi_rmon.h, [26](#)
- smi_rmon_get_alarm_falling_event_index
 - smi_rmon.h, [26](#)
- smi_rmon_get_alarm_falling_threshold
 - smi_rmon.h, [26](#)
- smi_rmon_get_alarm_owner
 - smi_rmon.h, [27](#)
- smi_rmon_get_event_comm
 - smi_rmon.h, [27](#)
- smi_rmon_get_event_description
 - smi_rmon.h, [28](#)
- smi_rmon_get_event_index
 - smi_rmon.h, [28](#)
- smi_rmon_get_event_owner
 - smi_rmon.h, [28](#)
- smi_rmon_get_event_status
 - smi_rmon.h, [29](#)

- smi_rmon_get_if_counter
 - smi_rmon.h, [29](#)
- smi_rmon_get_if_stats
 - smi_rmon.h, [29](#)
- smi_rmon_ifstats, [8](#)
- smi_rmon_msg.h, [42](#)
- smi_rmon_set_alarm_entry
 - smi_rmon.h, [30](#)
- smi_rmon_set_alarm_falling_event_index
 - smi_rmon.h, [30](#)
- smi_rmon_set_alarm_falling_threshold
 - smi_rmon.h, [31](#)
- smi_rmon_set_alarm_interval
 - smi_rmon.h, [31](#)
- smi_rmon_set_alarm_owner
 - smi_rmon.h, [32](#)
- smi_rmon_set_alarm_rising_event_index
 - smi_rmon.h, [32](#)
- smi_rmon_set_alarm_rising_threshold
 - smi_rmon.h, [32](#)
- smi_rmon_set_alarm_sample_type
 - smi_rmon.h, [33](#)
- smi_rmon_set_alarm_start_up
 - smi_rmon.h, [33](#)
- smi_rmon_set_alarm_status
 - smi_rmon.h, [33](#)
- smi_rmon_set_alarm_variable
 - smi_rmon.h, [34](#)
- smi_rmon_set_event_active
 - smi_rmon.h, [34](#)
- smi_rmon_set_event_comm
 - smi_rmon.h, [34](#)
- smi_rmon_set_event_description
 - smi_rmon.h, [35](#)
- smi_rmon_set_event_index
 - smi_rmon.h, [35](#)
- smi_rmon_set_event_owner
 - smi_rmon.h, [36](#)
- smi_rmon_set_event_status
 - smi_rmon.h, [36](#)
- smi_rmon_snmp_get_ether_stats_status
 - smi_rmon.h, [36](#)
- smi_rmon_snmp_get_event_community
 - smi_rmon.h, [37](#)
- smi_rmon_snmp_get_event_description
 - smi_rmon.h, [37](#)
- smi_rmon_snmp_get_event_owner
 - smi_rmon.h, [38](#)
- smi_rmon_snmp_get_event_type
 - smi_rmon.h, [38](#)
- smi_rmon_snmp_set_ether_stats_status
 - smi_rmon.h, [38](#)
- smi_rmon_snmp_set_event_community
 - smi_rmon.h, [39](#)
- smi_rmon_snmp_set_event_description
 - smi_rmon.h, [39](#)
- smi_rmon_snmp_set_event_owner
 - smi_rmon.h, [40](#)
- smi_rmon_snmp_set_event_type
 - smi_rmon.h, [40](#)
- smi_rmon_stats_flush_all_port
 - smi_rmon.h, [40](#)
- smi_rmon_stats_flush_port
 - smi_rmon.h, [41](#)