wrsw_rtud

v1.0

Generated by Doxygen 1.7.2

Thu Feb 3 2011 17:09:03

Contents

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

3 Data Structure Documentation

3.1 fd_handle Struct Reference

Filtering Database entry handle.

Data Fields

- int mem_type
- uint16_t addr
- struct filtering_entry * entry_ptr

3.1.1 Detailed Description

Filtering Database entry handle.

3.1.2 Field Documentation

3.1.2.1 uint16_t fd_handle::addr

3.1.2.2 struct filtering_entry* fd_handle::entry_ptr

3.1.2.3 int fd_handle::mem_type

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

- software/wrsw_rtud/rtu_fd.c

3.2 filtering_entry Struct Reference

RTU Filtering Database Entry Object.

`#include <rtu.h>`

Data Fields

- int valid
- int end_of_bucket
- int is_bpdu
- uint8_t mac [ETH_ALEN]
- uint8_t fid
- uint32_t port_mask_src
- uint32_t port_mask_dst
- int drop_when_source
- int drop_when_dest
- int drop_unmatched_src_ports
- uint32_t last_access_t
- uint8_t prio_src
- int has_prio_src
- int prio_override_src
- uint8_t prio_dst
- int has_prio_dst
- int prio_override_dst
- int go_to_cam
- uint16_t cam_addr
- int dynamic

3.2.1 Detailed Description

RTU Filtering Database Entry Object.

3.2.2 Field Documentation

3.2.2.1 uint16_t filtering_entry::cam_addr

3.2.2.2 int filtering_entry::drop_unmatched_src_ports

3.2.2.3 int filtering_entry::drop_when_dest

3.2.2.4 int filtering_entry::drop_when_source

3.2.2.5 int filtering_entry::dynamic

3.2.2.6 int filtering_entry::end_of_bucket

3.2.2.7 uint8_t filtering_entry::fid

3.2.2.8 int filtering_entry::go_to_cam

3.2.2.9 int filtering_entry::has_prio_dst

3.2.2.10 int filtering_entry::has_prio_src

3.2.2.11 int filtering_entry::is_bpdu

3.2.2.12 uint32_t filtering_entry::last_access_t

3.2.2.13 uint8_t filtering_entry::mac[ETH_ALEN]

3.2.2.14 uint32_t filtering_entry::port_mask_dst

3.2.2.15 uint32_t filtering_entry::port_mask_src

3.2.2.16 uint8_t filtering_entry::prio_dst

3.2.2.17 int filtering_entry::prio_override_dst

3.2.2.18 int filtering_entry::prio_override_src

3.2.2.19 uint8_t filtering_entry::prio_src

3.2.2.20 int filtering_entry::valid

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

- software/wrsw_rtud/rtu.h

3.3 hw_req Struct Reference

HW (HTAB or HCAM) write request.

Data Fields

- int type
- struct fd_handle handle
- struct hw_req * next

3.3.1 Detailed Description

HW (HTAB or HCAM) write request.

3.3.2 Field Documentation

3.3.2.1 struct fd_handle hw_req::handle

3.3.2.2 struct hw_req* hw_req::next

3.3.2.3 int hw_req::type

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

- software/wrsw_rtud/rtu_fd.c

3.4 rtu_request Struct Reference

RTU request: input for the RTU.

```
#include <rtu.h>
```

Data Fields

- int port_id
- uint8_t src [ETH_ALEN]
- uint8_t dst [ETH_ALEN]
- uint16_t vid
- int has_vid
- uint8_t prio
- int has_prio

3.4.1 Detailed Description

RTU request: input for the RTU.

3.4.2 Field Documentation

3.4.2.1 uint8_t rtu_request::dst[ETH_ALEN]

3.4.2.2 int rtu_request::has_prio

3.4.2.3 int rtu_request::has_vid

3.4.2.4 int rtu_request::port_id

3.4.2.5 uint8_t rtu_request::prio

3.4.2.6 uint8_t rtu_request::src[ETH_ALEN]

3.4.2.7 uint16_t rtu_request::vid

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

- software/wrsw_rtud/rtu.h

3.5 rtudexp_fd_entry_t Struct Reference

```
#include <rtud_exports.h>
```

Data Fields

- uint8_t mac [8]
- uint32_t dpm
- uint32_t spm
- uint8_t priority
- int dynamic

3.5.1 Field Documentation

3.5.1.1 uint32_t rtudexp_fd_entry_t::dpm

3.5.1.2 int rtudexp_fd_entry_t::dynamic

3.5.1.3 uint8_t rtudexp_fd_entry_t::mac[8]

3.5.1.4 uint8_t rtudexp_fd_entry_t::priority

3.5.1.5 uint32_t rtudexp_fd_entry_t::spm

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

- software/wrsw_rtud/rtud_exports.h

3.6 rtudexp_fd_list_t Struct Reference

`#include <rtud_exports.h>`

Data Fields

- rtudexp_fd_entry_t list [8]
- int num_rules
- int next

3.6.1 Field Documentation

3.6.1.1 rtudexp_fd_entry_t rtudexp_fd_list_t::list[8]

3.6.1.2 int rtudexp_fd_list_t::next

3.6.1.3 int rtudexp_fd_list_t::num_rules

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

- software/wrsw_rtud/rtud_exports.h

3.7 vlan_table_entry Struct Reference

RTU VLAN registration entry object.

`#include <rtu.h>`

Data Fields

- uint32_t port_mask
- uint8_t fid
- uint8_t prio
- int has_prio
- int prio_override
- int drop

3.7.1 Detailed Description

RTU VLAN registration entry object.

3.7.2 Field Documentation

3.7.2.1 int vlan_table_entry::drop

3.7.2.2 uint8_t vlan_table_entry::fid

3.7.2.3 int vlan_table_entry::has_prio

3.7.2.4 uint32_t vlan_table_entry::port_mask

3.7.2.5 uint8_t vlan_table_entry::prio

3.7.2.6 int vlan_table_entry::prio_override

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

- software/wrsw_rtud/rtu.h

4 File Documentation

4.1 software/wrsw_rtud/mac.c File Reference

```
#include "mac.h"
#include <stdio.h>
```

Functions

- char * mac_to_string (uint8_t mac[ETH_ALEN])
  Helper function to convert mac address into a string.

4.1.1 Function Documentation

4.1.1.1 char* mac_to_string (uint8_t *mac[ETH_ALEN]*)

Helper function to convert mac address into a string.

4.2 software/wrsw_rtud/mac.h File Reference

```
#include <stdint.h>
#include <string.h>
#include <linux/if_ether.h>
```

Functions

- static int mac_equal (uint8_t a[ETH_ALEN], uint8_t b[ETH_ALEN])
  Check whether two mac addresses are equal.
- static uint8_t * mac_copy (uint8_t dst[ETH_ALEN], uint8_t src[ETH_ALEN])
  copies src mac address into dst mac address.
- static uint8_t * mac_clean (uint8_t mac[ETH_ALEN])
  Set MAC address to 00:00:00:00:00:00.
- char * mac_to_string (uint8_t mac[ETH_ALEN])
  Helper function to convert mac address into a string.

4.2.1 Function Documentation

4.2.1.1 static uint8_t* mac_clean (uint8_t *mac[ETH_ALEN]*) [inline, static]

Set MAC address to 00:00:00:00:00:00.

Returns

pointer to mac address

4.2.1.2 static uint8_t* mac_copy (uint8_t *dst[ETH_ALEN]*, uint8_t *src[ETH_ALEN]*) [inline, static]

copies src mac address into dst mac address.

Returns

pointer to dst mac address

4.2.1.3 static int mac_equal (uint8_t *a[ETH_ALEN]*, uint8_t *b[ETH_ALEN]*) [inline, static]

Check whether two mac addresses are equal.

Returns

1 if both addresses are equal. 0 otherwise.

4.2.1.4 char* mac_to_string (uint8_t *mac[ETH_ALEN]*)

Helper function to convert mac address into a string.

4.3 software/wrsw_rtud/rtu.h File Reference

```
#include <stdint.h>
#include <errno.h>
#include <string.h>
#include <time.h>
#include <linux/if_ether.h>
#include <stdio.h>
#include "mac.h"
```

Data Structures

- struct rtu_request

  RTU request: input for the RTU.

- struct filtering_entry

RTU Filtering Database Entry Object.

- struct vlan_table_entry

  RTU VLAN registration entry object.

Defines

- #define RTU_BANKS 2
- #define RTU_BUCKETS 4
- #define LAST_RTU_BUCKET ((RTU_BUCKETS)-1)
- #define RTU_ENTRIES (16384/(RTU_BANKS))
- #define HTAB_ENTRIES ((RTU_ENTRIES)/(RTU_BUCKETS))
- #define LAST_HTAB_ENTRY ((HTAB_ENTRIES)-1)
- #define ENTRY_WORDS 8
- #define CAM_ENTRIES (((RTU_HCAM_WORDS)/(ENTRY_WORDS))/(RTU_BANKS))
- #define LAST_CAM_ENTRY ((CAM_ENTRIES)-1)
- #define MIN_PORT 0
- #define MAX_PORT 9
- #define NIC_PORT 10
- #define NUM_VLANS 4096
- #define NUM_RESERVED_ADDR 16
- #define DEFAULT_AGING_TIME 300
- #define MIN_AGING_TIME 10
- #define MAX_AGING_TIME 10000
- #define DEFAULT_AGING_RES 20
- #define time_after(a, b) ((long)(b) - (long)(a) < 0)
- #define TRACE_DBG(...)

Functions

- static struct filtering_entry * rtu_fe_copy (struct filtering_entry *dst, struct filtering_entry *src)

  Copies src filtering entry body into dst filtering entry body.

- static struct filtering_entry * rtu_fe_clean (struct filtering_entry *ent)

  Cleans MAC entry from mirror MAC table. All entry fields are reset.

- static unsigned long now ()

  Returns number of seconds since the epoch.

- int rtud_init_exports ()
- void rtud_handle_wripc ()

4.3.1 Define Documentation

4.3.1.1 #define CAM_ENTRIES (((RTU_HCAM_WORDS)/(ENTRY_WORDS))/(RTU_BANKS))

4.3.1.2 #define DEFAULT_AGING_RES 20

4.3.1.3 #define DEFAULT_AGING_TIME 300

4.3.1.4 #define ENTRY_WORDS 8

4.3.1.5 #define HTAB_ENTRIES ((RTU_ENTRIES)/(RTU_BUCKETS))

4.3.1.6 #define LAST_CAM_ENTRY ((CAM_ENTRIES)-1)

4.3.1.7 #define LAST_HTAB_ENTRY ((HTAB_ENTRIES)-1)

4.3.1.8 #define LAST_RTU_BUCKET ((RTU_BUCKETS)-1)

4.3.1.9 #define MAX_AGING_TIME 10000

4.3.1.10 #define MAX_PORT 9

4.3.1.11 #define MIN_AGING_TIME 10

4.3.1.12 #define MIN_PORT 0

4.3.1.13 #define NIC_PORT 10

4.3.1.14 #define NUM_RESERVED_ADDR 16

4.3.1.15 #define NUM_VLANS 4096

4.3.1.16 #define RTU_BANKS 2

4.3.1.17 #define RTU_BUCKETS 4

4.3.1.18 #define RTU_ENTRIES (16384/(RTU_BANKS))

4.3.1.19 #define time_after(*a*, *b*) ((long)(b) - (long)(a) < 0)

4.3.1.20 #define TRACE_DBG(...)

4.3.2 Function Documentation

4.3.2.1 static unsigned long now ( ) [inline, static]

Returns number of seconds since the epoch.

4.3.2.2 static struct filtering_entry* rtu_fe_clean (struct filtering_entry * *ent*) [static, read]

Cleans MAC entry from mirror MAC table. All entry fields are reset.

Parameters

ent	pointer to entry to clean (either in HCAM or HTAB)

Returns

pointer to filtering entry that was cleaned

4.3.2.3 static struct filtering_entry* rtu_fe_copy (struct filtering_entry * *dst*, struct filtering_entry * *src*) [static, read]

Copies src filtering entry body into dst filtering entry body.

Returns

pointer to dst filtering entry.

4.3.2.4 void rtud_handle_wripc ( )

4.3.2.5 int rtud_init_exports ( )

4.4 software/wrsw_rtud/rtu_drv.c File Reference

```
#include <unistd.h>
#include <fcntl.h>
```

```
#include <sys/ioctl.h>
#include <hw/switch_hw.h>
#include <hw/wrsw_rtu_wb.h>
#include <hal_client.h>
#include "rtu_drv.h"
#include "wr_rtu_irq.h"
```

Functions

- int shw_fpga_mmap_init ()
- static void write_mfifo_addr (uint32_t zbt_addr)
- static void write_mfifo_data (uint32_t word)
- static uint32_t mac_entry_word0_w (struct filtering_entry *ent)
- static uint32_t mac_entry_word1_w (struct filtering_entry *ent)
- static uint32_t mac_entry_word2_w (struct filtering_entry *ent)
- static uint32_t mac_entry_word3_w (struct filtering_entry *ent)
- static uint32_t mac_entry_word4_w (struct filtering_entry *ent)
- static void mac_entry_word0_r (uint32_t word, struct filtering_entry *ent)
- static void mac_entry_word1_r (uint32_t word, struct filtering_entry *ent)
- static void mac_entry_word2_r (uint32_t word, struct filtering_entry *ent)
- static void mac_entry_word3_r (uint32_t word, struct filtering_entry *ent)
- static void mac_entry_word4_r (uint32_t word, struct filtering_entry *ent)
- static uint32_t vlan_entry_word0_w (struct vlan_table_entry *ent)
- static uint32_t fpga_rtu_pcr_addr (int port)
- int rtu_init (void)

  Initialize HW RTU memory map.

- void rtu_exit (void)
- int rtu_ufifo_is_empty (void)

  Returns the UFIFO empty flag.

- int rtu_read_learning_queue_cnt (void)

  Returns the current learning queue length.

- int rtu_read_learning_queue (struct rtu_request *req)

  Reads unrecognised request from UFIFO. Blocks on read if learning queue is empty.

- int rtu_read_mfifo_cnt (void)

  Returns the current main hashtable CPU access FIFO (MFIFO) length.

- int rtu_mfifo_is_full (void)

  Checks whether the main hashtable CPU access FIFO (MFIFO) is full.

- int rtu_mfifo_is_empty (void)

Checks whether the main hashtable CPU access FIFO (MFIFO) is empty.

- void rtu_clean_mfifo (void)

  Cleans MFIFO.

- void rtu_write_htab_entry (uint16_t zbt_addr, struct filtering_entry *ent)

  Writes one MAC entry into main hash table at the given address.

- void rtu_clean_htab_entry (uint16_t zbt_addr)

  Cleans MAC entry in main hash table at the given address.

- void rtu_clean_htab (void)

  Cleans main hash table. Cleans all entries in HTAB inactive bank.

- void rtu_read_hcam_entry (uint16_t cam_addr, struct filtering_entry *ent)

  Reads MAC entry from HCAM Hash collisions memory.

- void rtu_write_hcam_entry (uint16_t cam_addr, struct filtering_entry *ent)

  Writes MAC entry to HCAM Hash collisions memory.

- void rtu_clean_hcam_entry (uint8_t cam_addr)

  Cleans MAC entry in HCAM Hash collisions memory.

- void rtu_clean_hcam (void)

  Cleans HCAM. Cleans all entries in HCAM inactive bank.

- uint32_t rtu_read_agr_htab (uint32_t addr)

  Read word from aging HTAB. Aging RAM Size: 256 32-bit words.

- void rtu_clean_agr_htab (void)

  Clears aging bitmap for HTAB.

- uint32_t rtu_read_agr_hcam (void)

  Read aging register for HCAM. Each bit corresponds to one MAC entry in HCAM memory.

- void rtu_clean_agr_hcam (void)

  Clears aging register for HCAM.

- void rtu_write_vlan_entry (uint32_t addr, struct vlan_table_entry *ent)

  Writes entry to vlan table. VLAN table size: 4096 32-bit words.

- void rtu_clean_vlan_entry (uint32_t addr)

  Cleans VLAN entry in VLAN table.

- void rtu_clean_vlan (void)

Cleans VLAN table (drop bit is set to 1)

- void rtu_enable (void)

  Enables RTU operation.

- void rtu_disable (void)

  Disables RTU operation.

- uint16_t rtu_read_hash_poly (void)

  Gets the polynomial used for hash computation in RTU at HW.

- void rtu_write_hash_poly (uint16_t hash_poly)

  Sets the polynomial used for hash computation in RTU at HW.

- void rtu_set_active_htab_bank (uint8_t bank)

  Set active ZBT bank.

- void rtu_set_active_hcam_bank (uint8_t bank)

  Set active CAM bank.

- void rtu_set_active_bank (uint8_t bank)

  Set active ZBT and CAM banks at once.

- int rtu_set_fixed_prio_on_port (int port, uint8_t prio)

  Sets fixed priority of value 'prio' on indicated port. It overrides the priority coming form the endpoint.

- int rtu_unset_fixed_prio_on_port (int port)

  Unsets fixed priority on indicated port. Orders to use priority from the endpoint instead.

- int rtu_learn_enable_on_port (int port, int flag)

  Sets the LEARN_EN flag on indicated port.

- int rtu_pass_bpdu_on_port (int port, int flag)

  Sets the PASS_BPDU flag on indicated port.

- int rtu_pass_all_on_port (int port, int flag)

  Sets the PASS_ALL flag on indicated port.

- int rtu_set_unrecognised_behaviour_on_port (int port, int flag)

  Sets the B_UNREC flag on indicated port.

- void rtu_enable_irq (void)
- void rtu_disable_irq (void)
- void rtu_clear_irq (void)

Variables

- static int fd

4.4.1 Function Documentation

4.4.1.1 static uint32_t fpga_rtu_pcr_addr (int *port*) [static]

4.4.1.2 static void mac_entry_word0_r (uint32_t *word,* struct filtering_entry * *ent*) [static]

4.4.1.3 static uint32_t mac_entry_word0_w (struct filtering_entry * *ent*) [static]

4.4.1.4 static void mac_entry_word1_r (uint32_t *word,* struct filtering_entry * *ent*) [static]

4.4.1.5 static uint32_t mac_entry_word1_w (struct filtering_entry * *ent*) [static]

4.4.1.6 static void mac_entry_word2_r (uint32_t *word,* struct filtering_entry * *ent*) [static]

4.4.1.7 static uint32_t mac_entry_word2_w (struct filtering_entry * *ent*) [static]

4.4.1.8 static void mac_entry_word3_r (uint32_t *word,* struct filtering_entry * *ent*) [static]

4.4.1.9 static uint32_t mac_entry_word3_w (struct filtering_entry * *ent*) [static]

4.4.1.10 static void mac_entry_word4_r (uint32_t *word,* struct filtering_entry * *ent*) [static]

4.4.1.11 static uint32_t mac_entry_word4_w (struct filtering_entry * *ent*) [static]

4.4.1.12 void rtu_clean_agr_hcam (void)

Clears aging register for HCAM.

4.4.1.13 void rtu_clean_agr_htab (void)

Clears aging bitmap for HTAB.

4.4.1.14 void rtu_clean_hcam (void)

Cleans HCAM. Cleans all entries in HCAM inactive bank.

4.4.1.15 void rtu_clean_hcam_entry (uint8_t *cam_addr*)

Cleans MAC entry in HCAM Hash collisions memory.

Parameters

addr	memory address which shoud be cleaned.

4.4.1.16 void rtu_clean_htab (void)

Cleans main hash table. Cleans all entries in HTAB inactive bank.

4.4.1.17 void rtu_clean_htab_entry (uint16_t *zbt_addr*)

Cleans MAC entry in main hash table at the given address.

Parameters

zbt_addr	memory address which shoud be cleaned.

4.4.1.18 void rtu_clean_mfifo (void)

Cleans MFIFO.

4.4.1.19 void rtu_clean_vlan (void)

Cleans VLAN table (drop bit is set to 1)

4.4.1.20 void rtu_clean_vlan_entry (uint32_t *addr*)

Cleans VLAN entry in VLAN table.

Parameters

addr	memory address which shoud be cleaned.

4.4.1.21 void rtu_clear_irq (void)

4.4.1.22 void rtu_disable (void)

Disables RTU operation.

4.4.1.23 void rtu_disable_irq (void)

4.4.1.24 void rtu_enable (void)

Enables RTU operation.

4.4.1.25 void rtu_enable_irq (void)

4.4.1.26 void rtu_exit (void)

4.4.1.27 int rtu_init (void)

Initialize HW RTU memory map.

4.4.1.28 int rtu_learn_enable_on_port (int *port,* int *flag*)

Sets the LEARN_EN flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0 disables learning. Otherwise: enables learning porcess on this port.

Returns

error code.

4.4.1.29 int rtu_mfifo_is_empty (void)

Checks whether the main hashtable CPU access FIFO (MFIFO) is empty.

Returns

1 if MFIFO is empty. 0 otherwise.

4.4.1.30 int rtu_mfifo_is_full (void)

Checks whether the main hashtable CPU access FIFO (MFIFO) is full.

Returns

1 if MFIFO is full. 0 otherwise.

4.4.1.31 int rtu_pass_all_on_port (int *port,* int *flag*)

Sets the PASS_ALL flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: all packets are dropped. Otherwise: all packets are passed.

Returns

error code.

4.4.1.32 int rtu_pass_bpdu_on_port (int *port,* int *flag*)

Sets the PASS_BPDU flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: BPDU packets are passed RTU rules only if PASS_ALL is set. Otherwise: BPDU packets are passed according to RTU rules.

Returns

error code.

4.4.1.33 uint32_t rtu_read_agr_hcam (void)

Read aging register for HCAM. Each bit corresponds to one MAC entry in HCAM memory.

4.4.1.34 uint32_t rtu_read_agr_htab (uint32_t *addr*)

Read word from aging HTAB. Aging RAM Size: 256 32-bit words.

4.4.1.35 uint16_t rtu_read_hash_poly (void)

Gets the polynomial used for hash computation in RTU at HW.

Returns

hash_poly hex representation of polynomial

4.4.1.36 void rtu_read_hcam_entry (uint16_t *cam_addr,* struct filtering_entry * *ent*)

Reads MAC entry from HCAM Hash collisions memory.

Parameters

ent	used to store the entry read. Memory should be handled by callee.
cam_addr	memory address which shoud be read.

4.4.1.37 int rtu_read_learning_queue (struct rtu_request * *req*)

Reads unrecognised request from UFIFO. Blocks on read if learning queue is empty.

Parameters

req	pointer to unrecognised request data. Memory handled by callee.

Returns

error code

4.4.1.38 int rtu_read_learning_queue_cnt (void)

Returns the current learning queue length.

Returns

Number of unrecognised requests currently in the learning queue.

4.4.1.39 int rtu_read_mfifo_cnt (void)

Returns the current main hashtable CPU access FIFO (MFIFO) length.

Returns

Number of MAC entries currently in the MFIFO.

4.4.1.40 void rtu_set_active_bank (uint8_t *bank*)

Set active ZBT and CAM banks at once.

Parameters

bank	active ZBT and CAM bank (0 or 1). Other values will be evaluated as 1.

4.4.1.41 void rtu_set_active_hcam_bank (uint8_t *bank*)

Set active CAM bank.

Parameters

bank	active CAM bank (0 or 1). Other values will be evaluated as 1.

4.4.1.42 void rtu_set_active_htab_bank (uint8_t *bank*)

Set active ZBT bank.

Parameters

bank	active ZBT bank (0 or 1). Other values will be evaluated as 1.

4.4.1.43 int rtu_set_fixed_prio_on_port (int *port,* uint8_t *prio*)

Sets fixed priority of value 'prio' on indicated port. It overrides the priority coming form the endpoint.

Parameters

port	port number (0 to 9)
prio	priority value

Returns

error code.

4.4.1.44 int rtu_set_unrecognised_behaviour_on_port (int *port,* int *flag*)

Sets the B_UNREC flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: packet is dropped. Otherwise: packet is broadcast.

Returns

error code.

4.4.1.45 int rtu_ufifo_is_empty (void)

Returns the UFIFO empty flag.

Returns

Value of UFIFO empty flag.

4.4.1.46 int rtu_unset_fixed_prio_on_port (int *port*)

Unsets fixed priority on indicated port. Orders to use priority from the endpoint instead.

Parameters

port	port number (0 to 9)

Returns

error code.

4.4.1.47 void rtu_write_hash_poly (uint16_t *hash_poly*)

Sets the polynomial used for hash computation in RTU at HW.

Parameters

hash_poly	hex representation of polynomial

4.4.1.48 void rtu_write_hcam_entry (uint16_t *cam_addr,* struct filtering_entry * *ent*)

Writes MAC entry to HCAM Hash collisions memory.

Parameters

ent	MAC table entry to be written to HCAM.
cam_addr	memory address in which MAC entry shoud be added.

4.4.1.49 void rtu_write_htab_entry (uint16_t *zbt_addr,* struct filtering_entry * *ent*)

Writes one MAC entry into main hash table at the given address.

Parameters

ent	MAC table entry to be written to MFIFO.
zbt_addr	ZBT SRAM memory address in which MAC entry shoud be added.

4.4.1.50 void rtu_write_vlan_entry (uint32_t *addr,* struct vlan_table_entry * *ent*)

Writes entry to vlan table. VLAN table size: 4096 32-bit words.

Parameters

addr	entry memory address

4.4.1.51 int shw_fpga_mmap_init ( )

4.4.1.52 static uint32_t vlan_entry_word0_w (struct vlan_table_entry * *ent*) [static]

4.4.1.53 static void write_mfifo_addr (uint32_t *zbt_addr*) [static]

4.4.1.54 static void write_mfifo_data (uint32_t *word*) [static]

4.4.2 Variable Documentation

4.4.2.1 int fd [static]

4.5 software/wrsw_rtud/rtu_drv.h File Reference

```
#include "rtu.h"
```

Defines

- #define RTU_HCAM 0x4000
- #define RTU_ARAM_MAIN 0x8000
- #define RTU_VLAN_TAB 0xc000
- #define RTU_MFIFO_R0_DATA_SEL 0x00000000
- #define RTU_MFIFO_R1_ADDR_MASK 0x0007FFFF
- #define RTU_DEVNAME "/dev/wr_rtu"

Functions

- int rtu_init (void) __attribute__((warn_unused_result))

  Initialize HW RTU memory map.

- void rtu_exit (void)
- int rtu_ufifo_is_empty (void)

  Returns the UFIFO empty flag.

- int rtu_read_learning_queue_cnt (void)

  Returns the current learning queue length.

- int rtu_read_learning_queue (struct rtu_request *req)

  Reads unrecognised request from UFIFO. Blocks on read if learning queue is empty.

- int rtu_read_mfifo_cnt (void)

  Returns the current main hashtable CPU access FIFO (MFIFO) length.

- int rtu_mfifo_is_full (void)

  Checks whether the main hashtable CPU access FIFO (MFIFO) is full.

- int rtu_mfifo_is_empty (void)

  Checks whether the main hashtable CPU access FIFO (MFIFO) is empty.

- void rtu_clean_mfifo (void)

  Cleans MFIFO.

- void rtu_write_htab_entry (uint16_t zbt_addr, struct filtering_entry *ent)

  Writes one MAC entry into main hash table at the given address.

- void rtu_clean_htab_entry (uint16_t zbt_addr)

  Cleans MAC entry in main hash table at the given address.

- void rtu_clean_htab (void)

  Cleans main hash table. Cleans all entries in HTAB inactive bank.

- void rtu_read_hcam_entry (uint16_t cam_addr, struct filtering_entry *ent)

  Reads MAC entry from HCAM Hash collisions memory.

- void rtu_write_hcam_entry (uint16_t cam_addr, struct filtering_entry *ent)

  Writes MAC entry to HCAM Hash collisions memory.

- void rtu_clean_hcam_entry (uint8_t cam_addr)

  Cleans MAC entry in HCAM Hash collisions memory.

- void rtu_clean_hcam (void)

  Cleans HCAM. Cleans all entries in HCAM inactive bank.

- uint32_t rtu_read_agr_htab (uint32_t addr)

  Read word from aging HTAB. Aging RAM Size: 256 32-bit words.

- void rtu_clean_agr_htab (void)

  Clears aging bitmap for HTAB.

- uint32_t rtu_read_agr_hcam (void)

  Read aging register for HCAM. Each bit corresponds to one MAC entry in HCAM memory.

- void rtu_clean_agr_hcam (void)

  Clears aging register for HCAM.

- void rtu_write_vlan_entry (uint32_t addr, struct vlan_table_entry *ent)

  Writes entry to vlan table. VLAN table size: 4096 32-bit words.

- void rtu_clean_vlan_entry (uint32_t addr)

  Cleans VLAN entry in VLAN table.

- void rtu_clean_vlan (void)

  Cleans VLAN table (drop bit is set to 1)

- void rtu_enable (void)

  Enables RTU operation.

- void rtu_disable (void)

  Disables RTU operation.

- void rtu_write_hash_poly (uint16_t hash_poly)

  Sets the polynomial used for hash computation in RTU at HW.

- uint16_t rtu_read_hash_poly (void)

  Gets the polynomial used for hash computation in RTU at HW.

- void rtu_set_active_htab_bank (uint8_t bank)

  Set active ZBT bank.

- void rtu_set_active_hcam_bank (uint8_t bank)

  Set active CAM bank.

- void rtu_set_active_bank (uint8_t bank)

  Set active ZBT and CAM banks at once.

- int rtu_set_fixed_prio_on_port (int port, uint8_t prio) __attribute__((warn_unused_result))

  Sets fixed priority of value 'prio' on indicated port. It overrides the priority coming form the endpoint.

- int rtu_unset_fixed_prio_on_port (int port) __attribute__((warn_unused_result))

  Unsets fixed priority on indicated port. Orders to use priority from the endpoint instead.

- int rtu_learn_enable_on_port (int port, int flag) __attribute__((warn_unused_result))

  Sets the LEARN_EN flag on indicated port.

- int rtu_pass_bpdu_on_port (int port, int flag) __attribute__((warn_unused_result))

  Sets the PASS_BPDU flag on indicated port.

- int rtu_pass_all_on_port (int port, int pass_all) __attribute__((warn_unused_result))

Sets the PASS_ALL flag on indicated port.

- int rtu_set_unrecognised_behaviour_on_port (int port, int flag) __attribute__((warn_unused_result))

  Sets the B_UNREC flag on indicated port.

- void rtu_enable_irq (void)
- void rtu_disable_irq (void)
- void rtu_clear_irq (void)

4.5.1 Define Documentation

4.5.1.1 #define RTU_ARAM_MAIN 0x8000

4.5.1.2 #define RTU_DEVNAME "/dev/wr_rtu"

4.5.1.3 #define RTU_HCAM 0x4000

4.5.1.4 #define RTU_MFIFO_R0_DATA_SEL 0x00000000

4.5.1.5 #define RTU_MFIFO_R1_ADDR_MASK 0x0007FFFF

4.5.1.6 #define RTU_VLAN_TAB 0xc000

4.5.2 Function Documentation

4.5.2.1 void rtu_clean_agr_hcam (void)

Clears aging register for HCAM.

4.5.2.2 void rtu_clean_agr_htab (void)

Clears aging bitmap for HTAB.

4.5.2.3 void rtu_clean_hcam (void)

Cleans HCAM. Cleans all entries in HCAM inactive bank.

4.5.2.4 void rtu_clean_hcam_entry (uint8_t *cam_addr*)

Cleans MAC entry in HCAM Hash collisions memory.

Parameters

addr	memory address which shoud be cleaned.

4.5.2.5 void rtu_clean_htab (void)

Cleans main hash table. Cleans all entries in HTAB inactive bank.

4.5.2.6 void rtu_clean_htab_entry (uint16_t *zbt_addr*)

Cleans MAC entry in main hash table at the given address.

Parameters

zbt_addr	memory address which shoud be cleaned.

4.5.2.7 void rtu_clean_mfifo (void)

Cleans MFIFO.

4.5.2.8 void rtu_clean_vlan (void)

Cleans VLAN table (drop bit is set to 1)

4.5.2.9 void rtu_clean_vlan_entry (uint32_t *addr*)

Cleans VLAN entry in VLAN table.

Parameters

addr	memory address which shoud be cleaned.

4.5.2.10 void rtu_clear_irq (void)

4.5.2.11 void rtu_disable (void)

Disables RTU operation.

4.5.2.12 void rtu_disable_irq (void)

4.5.2.13 void rtu_enable (void)

Enables RTU operation.

4.5.2.14 void rtu_enable_irq (void)

4.5.2.15 void rtu_exit (void)

4.5.2.16 int rtu_init (void)

Initialize HW RTU memory map.

4.5.2.17 int rtu_learn_enable_on_port (int *port,* int *flag*)

Sets the LEARN_EN flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0 disables learning. Otherwise: enables learning porcess on this port.

Returns

error code.

4.5.2.18 int rtu_mfifo_is_empty (void)

Checks whether the main hashtable CPU access FIFO (MFIFO) is empty.

Returns

1 if MFIFO is empty. 0 otherwise.

4.5.2.19 int rtu_mfifo_is_full (void)

Checks whether the main hashtable CPU access FIFO (MFIFO) is full.

Returns

1 if MFIFO is full. 0 otherwise.

4.5.2.20 int rtu_pass_all_on_port (int *port,* int *flag*)

Sets the PASS_ALL flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: all packets are dropped. Otherwise: all packets are passed.

Returns

error code.

4.5.2.21 int rtu_pass_bpdu_on_port (int *port,* int *flag*)

Sets the PASS_BPDU flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: BPDU packets are passed RTU rules only if PASS_ALL is set. Otherwise: BPDU packets are passed according to RTU rules.

Returns

error code.

4.5.2.22 uint32_t rtu_read_agr_hcam (void)

Read aging register for HCAM. Each bit corresponds to one MAC entry in HCAM memory.

4.5.2.23 uint32_t rtu_read_agr_htab (uint32_t *addr*)

Read word from aging HTAB. Aging RAM Size: 256 32-bit words.

4.5.2.24 uint16_t rtu_read_hash_poly (void)

Gets the polynomial used for hash computation in RTU at HW.

Returns

hash_poly hex representation of polynomial

4.5.2.25 void rtu_read_hcam_entry (uint16_t *cam_addr,* struct filtering_entry * *ent*)

Reads MAC entry from HCAM Hash collisions memory.

Parameters

ent	used to store the entry read. Memory should be handled by callee.
cam_addr	memory address which shoud be read.

4.5.2.26 int rtu_read_learning_queue (struct rtu_request * *req*)

Reads unrecognised request from UFIFO. Blocks on read if learning queue is empty.

Parameters

req	pointer to unrecognised request data. Memory handled by callee.

Returns

error code

4.5.2.27 int rtu_read_learning_queue_cnt (void)

Returns the current learning queue length.

Returns

Number of unrecognised requests currently in the learning queue.

4.5.2.28 int rtu_read_mfifo_cnt (void)

Returns the current main hashtable CPU access FIFO (MFIFO) length.

Returns

Number of MAC entries currently in the MFIFO.

4.5.2.29 void rtu_set_active_bank (uint8_t *bank*)

Set active ZBT and CAM banks at once.

Parameters

bank	active ZBT and CAM bank (0 or 1). Other values will be evaluated as 1.

4.5.2.30 void rtu_set_active_hcam_bank (uint8_t *bank*)

Set active CAM bank.

Parameters

bank	active CAM bank (0 or 1). Other values will be evaluated as 1.

4.5.2.31 void rtu_set_active_htab_bank (uint8_t *bank*)

Set active ZBT bank.

Parameters

bank	active ZBT bank (0 or 1). Other values will be evaluated as 1.

4.5.2.32 int rtu_set_fixed_prio_on_port (int *port,* uint8_t *prio*)

Sets fixed priority of value 'prio' on indicated port. It overrides the priority coming form the endpoint.

Parameters

port	port number (0 to 9)
prio	priority value

Returns

error code.

4.5.2.33 int rtu_set_unrecognised_behaviour_on_port (int *port,* int *flag*)

Sets the B_UNREC flag on indicated port.

Parameters

port	port number (0 to 9)
flag	0: packet is dropped. Otherwise: packet is broadcast.

Returns

error code.

4.5.2.34 int rtu_ufifo_is_empty (void)

Returns the UFIFO empty flag.

Returns

Value of UFIFO empty flag.

4.5.2.35 int rtu_unset_fixed_prio_on_port (int *port*)

Unsets fixed priority on indicated port. Orders to use priority from the endpoint instead.

Parameters

port	port number (0 to 9)

Returns

error code.

4.5.2.36 void rtu_write_hash_poly (uint16_t *hash_poly*)

Sets the polynomial used for hash computation in RTU at HW.

Parameters

hash_poly	hex representation of polynomial

4.5.2.37 void rtu_write_hcam_entry (uint16_t *cam_addr,* struct filtering_entry * *ent*)

Writes MAC entry to HCAM Hash collisions memory.

Parameters

ent	MAC table entry to be written to HCAM.
cam_addr	memory address in which MAC entry shoud be added.

4.5.2.38 void rtu_write_htab_entry (uint16_t *zbt_addr,* struct filtering_entry * *ent*)

Writes one MAC entry into main hash table at the given address.

Parameters

ent	MAC table entry to be written to MFIFO.
zbt_addr	ZBT SRAM memory address in which MAC entry shoud be added.

4.5.2.39 void rtu_write_vlan_entry (uint32_t *addr,* struct vlan_table_entry * *ent*)

Writes entry to vlan table. VLAN table size: 4096 32-bit words.

Parameters

addr	entry memory address

4.6 software/wrsw_rtud/rtu_fd.c File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
#include <hw/wrsw_rtu_wb.h>
#include <hw/trace.h>
#include "rtu_fd.h"
#include "rtu_drv.h"
#include "rtu_hash.h"
```

Data Structures

- struct fd_handle

  Filtering Database entry handle.

- struct hw_req

  HW (HTAB or HCAM) write request.

Defines

- #define HTAB 0
- #define HCAM 1
- #define HW_WRITE_REQ 0
- #define HW_CLEAN_REQ 1

- #define FOUND 1
- #define NOT_FOUND 0
- #define NOT_FOUND_AND_FULL -1
- #define NOT_FOUND_AND_FIRST -1

Functions

- static struct hw_req * tail (struct hw_req *head)
- static void clean_list (struct hw_req *head)
- static int add_hw_req (int type, int mem, uint16_t addr, struct filtering_entry *ent)

  Adds a new HW request at the end of the main hw request list.

- static int write_htab_entry (uint16_t addr, struct filtering_entry *e)
- static int write_hcam_entry (uint16_t addr, struct filtering_entry *e)
- static int clean_htab_entry (uint16_t addr)
- static int clean_hcam_entry (uint16_t addr)
- static uint16_t zbt_addr (uint16_t hash, int bucket)
- static uint16_t cam_addr (int bucket)
- static int cam_bucket (uint16_t cam_addr)
- static int matched (uint32_t word, int offset)
- static int htab_contains (uint8_t mac[ETH_ALEN], uint8_t fid, int *bucket, struct filtering_entry **ent)

  Checks whether a given pair (mac,fid) is at HTAB.

- static int hcam_contains (uint8_t mac[ETH_ALEN], uint8_t fid, int *bucket, struct filtering_entry **ent)
- static int find_empty_bucket_in_hcam (void)

  Find the most appropriate empty bucket to insert new hash collision list. The algorithm first finds the fragment which contains the max number of consecutive empty positions. Then divides this fragment into two parts: first block is still available for possible increment of any existing list; The second block will be available for the new list. The algorithm keeps a fair and uniform distribution of fragments space.

- static void set_active_bank (int b)

  Set the filtering database active bank both in software and hardware. Note both HTAB and HCAM active banks are switched at once. Bank switching is delayed until MFIFO is empty (method remains blocked meanwhile).

- static void clean_fd (void)
- static void clean_vd (void)
- static void clean_aging_map (void)

  Clean HCAM aging register and HTAB aging bitmap.

- static void update_aging_map (void)

  Update aging map cache with contents read from aging registers at HW.

- static void rtu_hw_commit (void)

  Read changes from hw_req_list and invoke RTU driver to efectively write or clean the entry.

- static void rtu_fd_commit (void)

  Commits entry changes at software to hardware HTAB and HCAM.

- static void shift_htab_entries (uint16_t hash, int bucket)

  Shifts HTAB list one position, starting at bucket.

- static int shift_hcam_entries (int bucket)

  Shifts HCAM list one position, starting at bucket. If entry to remove is end of bucket, marks previous one (if exists) as the new end of bucket.

- static void delete_htab_entry (uint16_t hash, int bucket)

  Deletes HTAB entry by shifting HTAB list. If HCAM is used, it also copies first HCAM entry to last HTAB bucket.

- static void delete_hcam_entry (int bucket)

  Deletes HCAM entry by shifting HCAM list. Updates HTAB last entry if neccessary.

- static void rtu_fd_age_out (void)
- static void rtu_fd_age_update (void)

  Updates the age of filtering entries accessed in the last period.

- int rtu_fd_init (uint16_t poly, unsigned long aging)

  Initializes the RTU filtering database.

- int rtu_fd_create_entry (uint8_t mac[ETH_ALEN], uint16_t vid, uint32_t port_map, int dynamic)

  Creates or updates a filtering entry in the filtering database.

- void rtu_fd_set_hash_poly (uint16_t poly)

  Set the polynomial used for hash calculation. Changing the hash polynomial requires removing any existing entry from RTU table. Note in case RTU table becomes full, this function may be used to change hash polynomial (thus leading to a different hash distribution).

- int rtu_fd_set_aging_time (unsigned long t)

  Sets the aging time for dynamic filtering entries.

- void rtu_fd_flush (void)

  Deletes old filtering entries from filtering database to support changes in active topology.

- struct filtering_entry * rtu_fd_lookup_htab_entry (int index)

Variables

- struct hw_req * hw_req_list

  HW write and clean requests list. Used to temporarily store entry changes performed at SW.

- static struct filtering_entry rtu_htab [HTAB_ENTRIES][RTU_BUCKETS]

  Mirror of ZBT SRAM memory MAC address table. Main filtering table organized as hash table with 4-entry buckets. Note both banks have the same content. Therefore SW only mirrors one bank.

- static struct filtering_entry rtu_hcam [CAM_ENTRIES]

  Mirror of CAM lookup table. For RTU entries with more than 4 matches.

- static uint8_t bank

  Table bank to write entries to. HTAB and HCAM banks will be handled according to this single bank value.

- static uint32_t rtu_agr_htab [RTU_ARAM_MAIN_WORDS]

  Mirror of Aging RAM.

- static uint32_t rtu_agr_hcam
- static unsigned long aging_time = DEFAULT_AGING_TIME

  Max time that a dynamic MAC entry can remain in the MAC table after being used. [seconds].

- static struct vlan_table_entry vlan_tab [NUM_VLANS]
- static pthread_mutex_t fd_mutex

  Mutex used to synchronise concurrent access to the filtering database.

4.6.1 Define Documentation

4.6.1.1 #define FOUND 1

4.6.1.2 #define HCAM 1

4.6.1.3 #define HTAB 0

4.6.1.4 #define HW_CLEAN_REQ 1

4.6.1.5 #define HW_WRITE_REQ 0

4.6.1.6 #define NOT_FOUND 0

4.6.1.7 #define NOT_FOUND_AND_FIRST -1

4.6.1.8 #define NOT_FOUND_AND_FULL -1

4.6.2 Function Documentation

4.6.2.1 static int add_hw_req (int *type,* int *mem,* uint16_t *addr,* struct filtering_entry * *ent*) [static]

Adds a new HW request at the end of the main hw request list.

4.6.2.2 static uint16_t cam_addr (int *bucket*) [inline, static]

4.6.2.3 static int cam_bucket (uint16_t *cam_addr*) [inline, static]

4.6.2.4 static void clean_aging_map (void) [static]

Clean HCAM aging register and HTAB aging bitmap.

4.6.2.5 static void clean_fd (void) [static]

Filtering database initialisation.

4.6.2.6 static int clean_hcam_entry (uint16_t *addr*) [inline, static]

4.6.2.7 static int clean_htab_entry (uint16_t *addr*) [inline, static]

4.6.2.8 static void clean_list (struct hw_req * *head*) [static]

Removes all elements from the hw_req_list

4.6.2.9 static void clean_vd (void) [static]

VLAN database initialisation. VLANs are initially marked as disabled.

4.6.2.10 static void delete_hcam_entry (int *bucket*) [static]

Deletes HCAM entry by shifting HCAM list. Updates HTAB last entry if neccessary.

Parameters

bucket	CAM entry address

4.6.2.11 static void delete_htab_entry (uint16_t *hash*, int *bucket*) [static]

Deletes HTAB entry by shifting HTAB list. If HCAM is used, it also copies first HCAM entry to last HTAB bucket.

Parameters

hash	hashcode for entry to remove.
bucket	HTAB bucket for entry to remove

4.6.2.12 static int find_empty_bucket_in_hcam (void) [static]

Find the most appropriate empty bucket to insert new hash collision list. The algorithm first finds the fragment which contains the max number of consecutive empty positions. Then divides this fragment into two parts: first block is still available for possible increment of any existing list; The second block will be available for the new list. The algorithm keeps a fair and uniform distribution of fragments space.

Returns

bucket index or -1 if the HCAM table is full.

4.6.2.13 static int hcam_contains (uint8_t *mac[ETH_ALEN],* uint8_t *fid,* int * *bucket,* struct filtering_entry ** *ent*) [static]

Checks whether a given pair (mac,fid) is at HCAM

Parameters

mac	mac address
fid	filtering database identifier
bucket	inout param.Returns the bucket number where the entry was found
ent	pointer to entry found.

Returns

0 if entry was not found. 1 if entry was found. -1 if entry was not found and the end of bucket was reached. -ENOMEM if no more entries after the end of bucket. -EINVAL if bucket >= CAM_ENTRIES or HCAM inconsistent.

4.6.2.14 static int htab_contains (uint8_t *mac[ETH_ALEN],* uint8_t *fid,* int * *bucket,* struct filtering_entry ** *ent*) [static]

Checks whether a given pair (mac,fid) is at HTAB.

Parameters

mac	mac address
fid	filtering database identifier
bucket	inout param.Returns the bucket number where the entry was found
ent	pointer to entry found.

Returns

0 if entry was not found. 1 if entry was found. -1 if not found and HTAB was full for the corresponding hash. -EINVAL if bucket >= RTU_BUCKETS

4.6.2.15 static int matched (uint32_t *word,* int *offset*) [inline, static]

4.6.2.16 static void rtu_fd_age_out (void) [static]

For each filtering entry in the filtering database, this method checks its last access time and removes it in case entry is older than the aging time.

4.6.2.17 static void rtu_fd_age_update (void) [static]

Updates the age of filtering entries accessed in the last period.

4.6.2.18 static void rtu_fd_commit (void) [static]

Commits entry changes at software to hardware HTAB and HCAM.

4.6.2.19 int rtu_fd_create_entry (uint8_t *mac[ETH_ALEN],* uint16_t *vid,* uint32_t *port_map,* int *dynamic*)

Creates or updates a filtering entry in the filtering database.

Parameters

mac	MAC address specification
vid	VLAN identifier
port_map	a port map specification with a control element for each outbound port to specify filtering for that MAC address specification and VID
dynamic	it indicates whether it's a dynamic entry

Returns

0 if entry was created or updated. -ENOMEM if no space is available.

4.6.2.20 void rtu_fd_flush (void)

Deletes old filtering entries from filtering database to support changes in active topology.

4.6.2.21 int rtu_fd_init (uint16_t *poly*, unsigned long *aging*)

Initializes the RTU filtering database.

Parameters

poly	hash polinomial.
aging	aging time

4.6.2.22 struct filtering_entry* rtu_fd_lookup_htab_entry (int *index*) [read]

4.6.2.23 int rtu_fd_set_aging_time (unsigned long *t*)

Sets the aging time for dynamic filtering entries.

Parameters

t	new aging time value [seconds].

Returns

-EINVAL if t $<$ 10 or t $>$ 1000000 (802.1Q, Table 8.3); 0 otherwise.

4.6.2.24 void rtu_fd_set_hash_poly (uint16_t *poly*)

Set the polynomial used for hash calculation. Changing the hash polynomial requires removing any existing entry from RTU table. Note in case RTU table becomes full, this function may be used to change hash polynomial (thus leading to a different hash distribution).

Parameters

poly	binary polynomial representation. CRC-16-CCITT -> 1+x^5+x^12+x^16 -> 0x1021 CRC-16-IBM -> 1+x^2+x^15+x^16 -> 0x8005 CRC-16-DECT -> 1+x^3+x^7+x^8+x^10+x^16 -> 0x0589

4.6.2.25 static void rtu_hw_commit (void) [static]

Read changes from hw_req_list and invoke RTU driver to efectively write or clean the entry.

4.6.2.26 static void set_active_bank (int *n*) [static]

Set the filtering database active bank both in software and hardware. Note both HTAB and HCAM active banks are switched at once. Bank switching is delayed until MFIFO is empty (method remains blocked meanwhile).

4.6.2.27 static int shift_hcam_entries (int *bucket*) [static]

Shifts HCAM list one position, starting at bucket. If entry to remove is end of bucket, marks previous one (if exists) as the new end of bucket.

Returns

-1 if more entries remain in HCAM. Otherwise, returns the hash for entry, in order to help modifying the last HTAB entry

4.6.2.28 static void shift_htab_entries (uint16_t *hash,* int *bucket*) [static]

Shifts HTAB list one position, starting at bucket.

4.6.2.29 static struct hw_req * tail (struct hw_req * *head*) [static, read]

Returns pointer to last element in hw_req_list.

4.6.2.30 static void update_aging_map (void) [static]

Update aging map cache with contents read from aging registers at HW.

4.6.2.31 static int write_hcam_entry (uint16_t *addr,* struct filtering_entry * *e*) [inline, static]

4.6.2.32 static int write_htab_entry (uint16_t *addr,* struct filtering_entry * *e*) [inline, static]

4.6.2.33 static uint16_t zbt_addr (uint16_t *hash,* int *bucket*) [inline, static]

4.6.3 Variable Documentation

4.6.3.1 unsigned long aging_time = DEFAULT_AGING_TIME [static]

Max time that a dynamic MAC entry can remain in the MAC table after being used. [seconds].

4.6.3.2 uint8_t bank [static]

Table bank to write entries to. HTAB and HCAM banks will be handled according to this single bank value.

4.6.3.3 pthread_mutex_t fd_mutex [static]

Mutex used to synchronise concurrent access to the filtering database.

4.6.3.4 struct hw_req* hw_req_list

HW write and clean requests list. Used to temporarily store entry changes performed at SW.

4.6.3.5 uint32_t rtu_agr_hcam [static]

4.6.3.6 uint32_t rtu_agr_htab[RTU_ARAM_MAIN_WORDS] [static]

Mirror of Aging RAM.

4.6.3.7 struct filtering_entry rtu_hcam[CAM_ENTRIES] [static]

Mirror of CAM lookup table. For RTU entries with more than 4 matches.

4.6.3.8 struct filtering_entry rtu_htab[HTAB_ENTRIES][RTU_BUCKETS] [static]

Mirror of ZBT SRAM memory MAC address table. Main filtering table organized as hash table with 4-entry buckets. Note both banks have the same content. Therefore SW only mirrors one bank.

4.6.3.9 struct vlan_table_entry vlan_tab[NUM_VLANS] [static]

Mirror of VLAN table

4.7 software/wrsw_rtud/rtu_fd.h File Reference

```
#include "rtu.h"
```

Defines

- #define STATIC 0
- #define DYNAMIC 1

Functions

- int rtu_fd_init (uint16_t poly, unsigned long aging) __attribute__((warn_unused_result))

  Initializes the RTU filtering database.

- int rtu_fd_create_entry (uint8_t mac[ETH_ALEN], uint16_t vid, uint32_t port_map, int dynamic) __attribute__((warn_unused_result))

  Creates or updates a filtering entry in the filtering database.

- int rtu_fd_set_aging_time (unsigned long t) __attribute__((warn_unused_result))

  Sets the aging time for dynamic filtering entries.

- void rtu_fd_set_hash_poly (uint16_t poly)

  Set the polynomial used for hash calculation. Changing the hash polynomial requires removing any existing entry from RTU table. Note in case RTU table becomes full, this function may be used to change hash polynomial (thus leading to a different hash distribution).

- void rtu_fd_flush (void)

  Deletes old filtering entries from filtering database to support changes in active topology.

- struct filtering_entry * rtu_fd_lookup_htab_entry (int index)

4.7.1 Define Documentation

4.7.1.1 #define DYNAMIC 1

4.7.1.2 #define STATIC 0

4.7.2 Function Documentation

4.7.2.1 int rtu_fd_create_entry (uint8_t *mac[ETH_ALEN],* uint16_t *vid,* uint32_t *port_map,* int *dynamic*)

Creates or updates a filtering entry in the filtering database.

Parameters

mac	MAC address specification
vid	VLAN identifier
port_map	a port map specification with a control element for each outbound port to specify filtering for that MAC address specification and VID
dynamic	it indicates whether it's a dynamic entry

Returns

0 if entry was created or updated. -ENOMEM if no space is available.

4.7.2.2 void rtu_fd_flush (void)

Deletes old filtering entries from filtering database to support changes in active topology.

4.7.2.3 int rtu_fd_init (uint16_t *poly,* unsigned long *aging*)

Initializes the RTU filtering database.

Parameters

poly	hash polinomial.
aging	aging time

4.7.2.4 struct filtering_entry* rtu_fd_lookup_htab_entry (int *index*) [read]

4.7.2.5 int rtu_fd_set_aging_time (unsigned long *t*)

Sets the aging time for dynamic filtering entries.

Parameters

t	new aging time value [seconds].

Returns

-EINVAL if t $<$ 10 or t $>$ 1000000 (802.1Q, Table 8.3); 0 otherwise.

4.7.2.6 void rtu_fd_set_hash_poly (uint16_t *poly*)

Set the polynomial used for hash calculation. Changing the hash polynomial requires removing any existing entry from RTU table. Note in case RTU table becomes full, this function may be used to change hash polynomial (thus leading to a different hash distribution).

Parameters

poly	binary polynomial representation. CRC-16-CCITT -> 1+x^5+x^12+x^16 -> 0x1021 CRC-16-IBM -> 1+x^2+x^15+x^16 -> 0x8005 CRC-16-DECT -> 1+x^3+x^7+x^8+x^10+x^16 -> 0x0589

4.8 software/wrsw_rtud/rtu_hash.c File Reference

```
#include <hw/trace.h>
#include "rtu_hash.h"
```

Functions

- static uint16_t crc16 (uint16_t const init_crc, uint16_t const message)
- void rtu_hash_set_poly (uint16_t poly)
- uint16_t rtu_hash (uint8_t mac[ETH_ALEN], uint8_t fid)

Variables

- static uint32_t hash_poly

  Polynomial used to calculate the MAC entry hash code.

4.8.1 Function Documentation

4.8.1.1 static uint16_t crc16 (uint16_t const *init_crc,* uint16_t const *message*) [static]

4.8.1.2 uint16_t rtu_hash (uint8_t *mac[ETH_ALEN],* uint8_t *fid*)

4.8.1.3 void rtu_hash_set_poly (uint16_t *poly*)

4.8.2 Variable Documentation

4.8.2.1 uint32_t hash_poly [static]

Polynomial used to calculate the MAC entry hash code.

4.9 software/wrsw_rtud/rtu_hash.h File Reference

```
#include "mac.h"
```

Defines

- #define HW_POLYNOMIAL_CCITT 0x1021
- #define HW_POLYNOMIAL_IBM 0x8005
- #define HW_POLYNOMIAL_DECT 0x0589

Functions

- void rtu_hash_set_poly (uint16_t poly)
- uint16_t rtu_hash (uint8_t mac[ETH_ALEN], uint8_t fid)

4.9.1 Define Documentation

4.9.1.1 #define HW_POLYNOMIAL_CCITT 0x1021

4.9.1.2 #define HW_POLYNOMIAL_DECT 0x0589

4.9.1.3 #define HW_POLYNOMIAL_IBM 0x8005

4.9.2 Function Documentation

4.9.2.1 uint16_t rtu_hash (uint8_t *mac[ETH_ALEN],* uint8_t *fid*)

4.9.2.2 void rtu_hash_set_poly (uint16_t *poly*)

4.10 software/wrsw_rtud/rtud.c File Reference

```
#include <unistd.h>
#include <stdlib.h>
#include <pthread.h>
#include <signal.h>
#include <hw/switch_hw.h>
#include <hal_client.h>
#include "rtu.h"
#include "mac.h"
#include "rtu_fd.h"
```

```
#include "rtu_drv.h"
#include "rtu_hash.h"
#include "utils.h"
```

Functions

- static int rtu_create_static_entries ()

  Creates the static entries in the filtering database.

- static void * rtu_daemon_aging_process (void *arg)

  Periodically removes the filtering database old entries.

- static void * rtu_daemon_wripc_process (void *arg)

  Handles WRIPC requests. Currently used to dump the filtering database contents when requested by external processes.

- static int rtu_daemon_learning_process ()

  Handles the learning process.

- static int rtu_daemon_init (uint16_t poly, unsigned long aging_time)

  RTU set up. Initialises routing table cache and RTU at hardware.

- static void rtu_daemon_destroy ()

  RTU shutdown.

- void sigint (int signum)
- int main (int argc, char **argv)

  Starts up the learning and aging processes.

Variables

- static pthread_t aging_process
- static pthread_t wripc_process

4.10.1 Function Documentation

4.10.1.1 int main (int *argc,* char ** *argv*)

Starts up the learning and aging processes.

4.10.1.2 static int rtu_create_static_entries ( ) [static]

Creates the static entries in the filtering database.

Returns

error code

4.10.1.3 static void* rtu_daemon_aging_process (void * *arg*) [static]

Periodically removes the filtering database old entries.

4.10.1.4 static void rtu_daemon_destroy ( ) [static]

RTU shutdown.

4.10.1.5 static int rtu_daemon_init (uint16_t *poly,* unsigned long *aging_time*) [static]

RTU set up. Initialises routing table cache and RTU at hardware.

Parameters

poly	hash polinomial.
aging_time	Aging time in seconds.

Returns

error code.

4.10.1.6 static int rtu_daemon_learning_process ( ) [static]

Handles the learning process.

Returns

error code

4.10.1.7 static void* rtu_daemon_wripc_process (void * *arg*) [static]

Handles WRIPC requests. Currently used to dump the filtering database contents when requested by external processes.

4.10.1.8 void sigint (int *signum*)

4.10.2 Variable Documentation

4.10.2.1 pthread_t aging_process [static]

4.10.2.2 pthread_t wripc_process [static]

4.11 software/wrsw_rtud/rtud_exports.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <errno.h>
#include <wr_ipc.h>
#include <hw/trace.h>
#include "rtu.h"
#include "rtu_fd.h"
#include "rtud_exports.h"
#include "mac.h"
```

Functions

- int rtud_init_exports ()
- void rtudexp_get_fd_list (rtudexp_fd_list_t *list, int start_from)
- void rtud_handle_wripc ()

Variables

- static int rtud_ipc

4.11.1 Function Documentation

4.11.1.1 void rtud_handle_wripc ( )

4.11.1.2 int rtud_init_exports ( )

4.11.1.3 void rtudexp_get_fd_list (rtudexp_fd_list_t * *list,* int *start_from*)

4.11.2 Variable Documentation

4.11.2.1 int rtud_ipc [static]

4.12 software/wrsw_rtud/rtud_exports.h File Reference

```
#include <stdint.h>
```

Data Structures

- struct rtudexp_fd_entry_t
- struct rtudexp_fd_list_t

Functions

- void rtudexp_get_fd_list (rtudexp_fd_list_t *list, int start_from)

4.12.1 Function Documentation

4.12.1.1 void rtudexp_get_fd_list (rtudexp_fd_list_t * *list,* int *start_from*)

4.13 software/wrsw_rtud/utils.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
```

Functions

- void daemonize (void)
- void usage (char *name)

4.13.1 Function Documentation

4.13.1.1 void daemonize (void)

4.13.1.2 void usage (char * *name*)

4.14 software/wrsw_rtud/utils.h File Reference

Functions

- void daemonize (void)
- void usage (char *name)

4.14.1 Function Documentation

4.14.1.1 void daemonize (void)

4.14.1.2 void usage (char * *name*)

4.15 software/wrsw_rtud/wr_rtu.h File Reference

Defines

- #define __WR_IOC_MAGIC '4'
- #define WR_RTU_IRQWAIT _IO(__WR_IOC_MAGIC, 4)
- #define WR_RTU_IRQENA _IO(__WR_IOC_MAGIC, 5)

4.15.1 Define Documentation

4.15.1.1 #define __WR_IOC_MAGIC '4'

4.15.1.2 #define WR_RTU_IRQENA _IO(__WR_IOC_MAGIC, 5)

4.15.1.3 #define WR_RTU_IRQWAIT _IO(__WR_IOC_MAGIC, 4)