Host Communication Protocol

2.0

Generated by Doxygen 1.8.11

Contents

Chapter 1

Main

Welcome to the documentation for the Host Communication Protocol (HCP).

The first part covers the physical method of sending messages and the second part covers the specification of the different command flows.

1.1 FPC embedded stack and HCP

- FPC embedded stack
- HCP frame format

1.2 Command flow specification

- Biometrics
  - Capture
  - Extract
  - Enroll
  - Identify
- Image handling
- Template handling
- Storage handling
- Sensor operations
- Device operations

Chapter 2

FPC embedded stack

The communication stack implemented on the embedded devices by FPC follows the following specification.

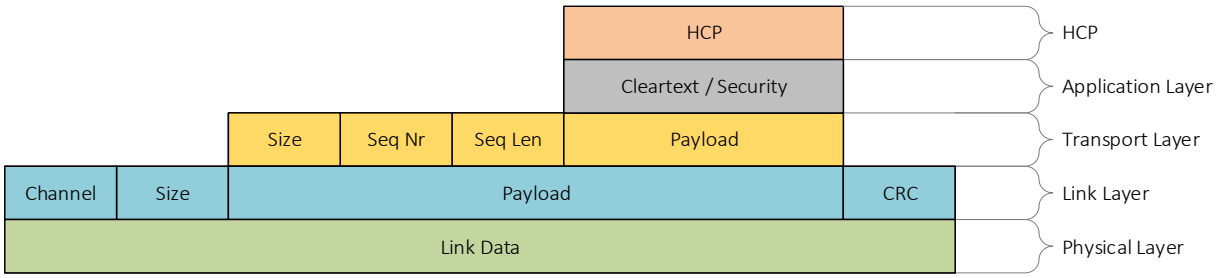


Figure 2.1 HCP embedded stack

2.1 Physical

The physical layer have a fixed size buffer of 256 bytes.

2.2 Link

The link layer handles packet consistency.

Each packet received is acknowledged on the link layer, if an error occurs no retransmission is done on this level, instead the error is propagated upwards.

Channel	Size	Payload	CRC
2 bytes	2 bytes	size bytes	4 bytes

All fields are using unsigned data types.

2.3 Transport

The transport layer handles packet segmentation.

As the PHY MTU is 256 bytes the maximum payload per segment is 242 bytes.

Errors are propagated upwards.

Size	Seq Nr	Seq Len	Payload
2 bytes	2 bytes	2 bytes	size bytes

All fields are using unsigned data types.

2.4 Application

The application layer is a optional security layer, the default implementation is clear text (unsecure).

If a security solution is used it will be part of that products documentation.

2.5 HCP

The HCP frame is described in the hcpf HCP frame format section.

Chapter 3

HCP frame format

The Host Communication Protocol (HCP) describes a general way of sending commands and information between devices.

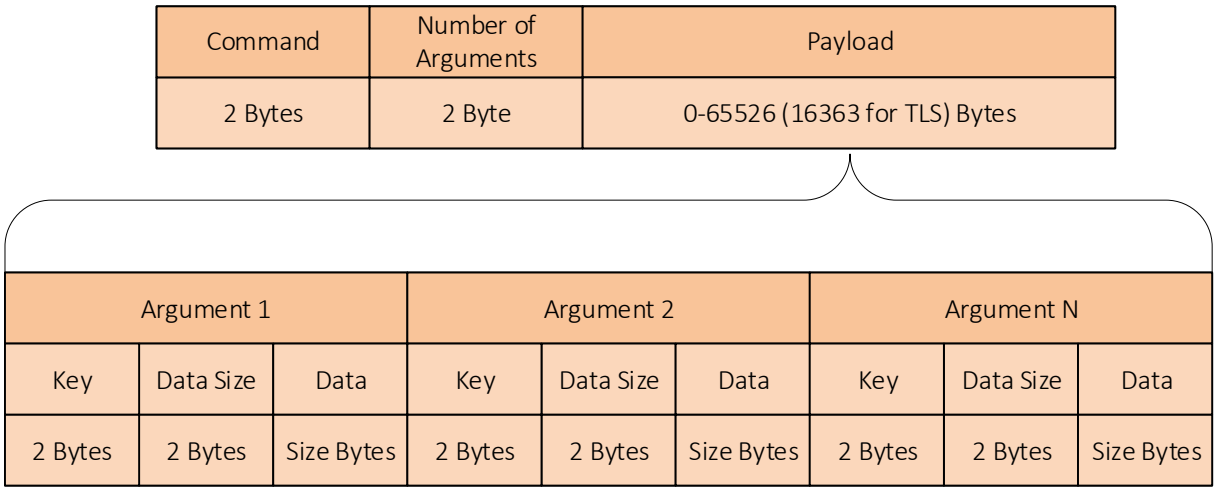


Figure 3.1 HCP frame format

3.1 Command

The Commands define the general action that is going to be executed. However, each command can have several Arguments each with data attached.

CMD	Num Args	Payload
2 bytes	2 bytes	xx bytes

All fields are using unsigned data types.

3.2 Argument

The Argument is used as a complement to the command if it is needed and can contain arbitrary data.

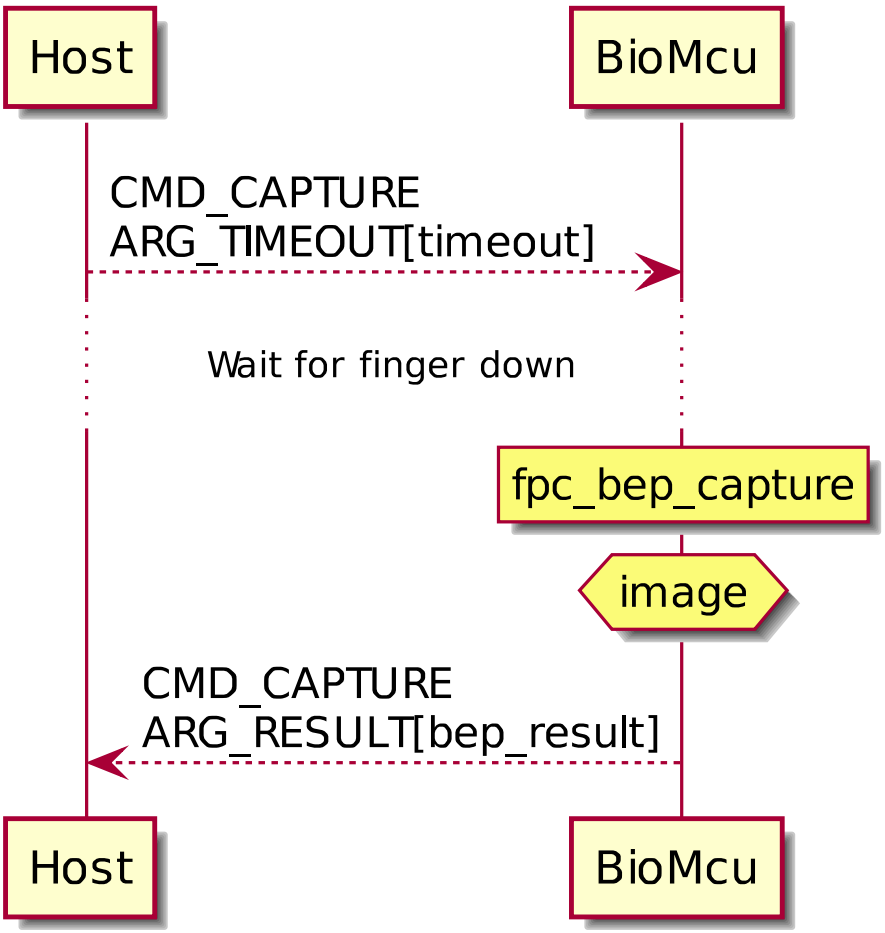
ARG	Size	Data
2 bytes	2 bytes	size bytes

All fields are using unsigned data types.

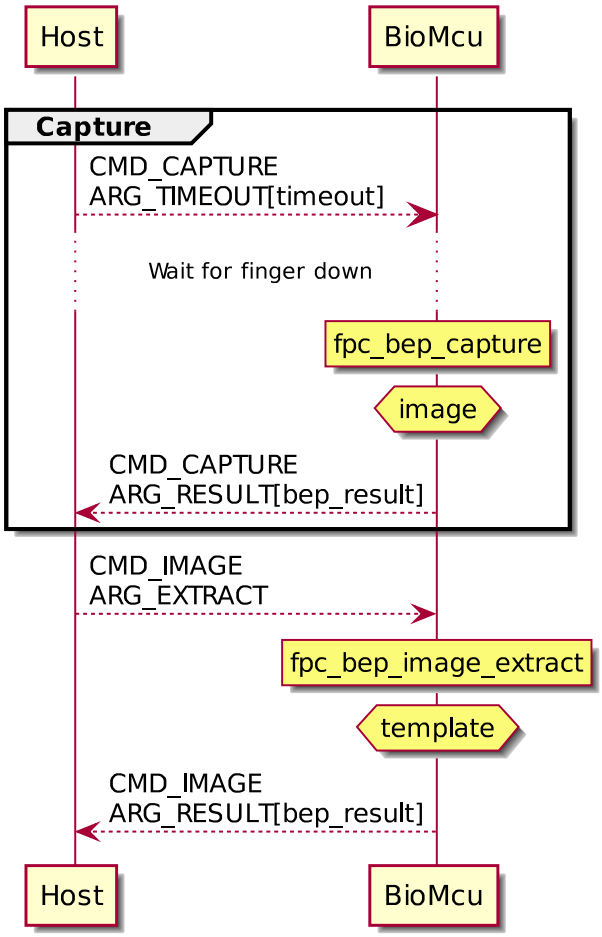
Chapter 4

Biometrics

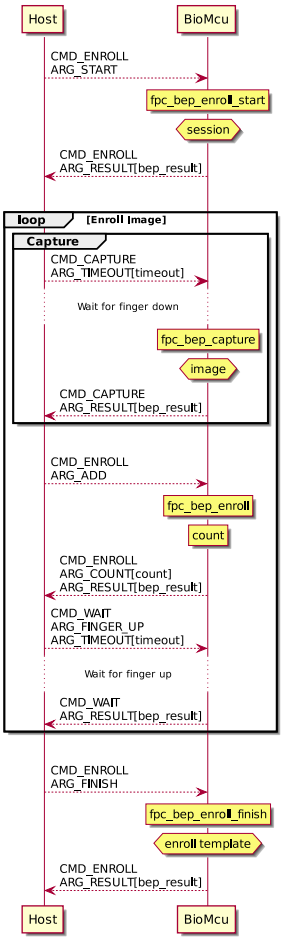
4.1 Capture



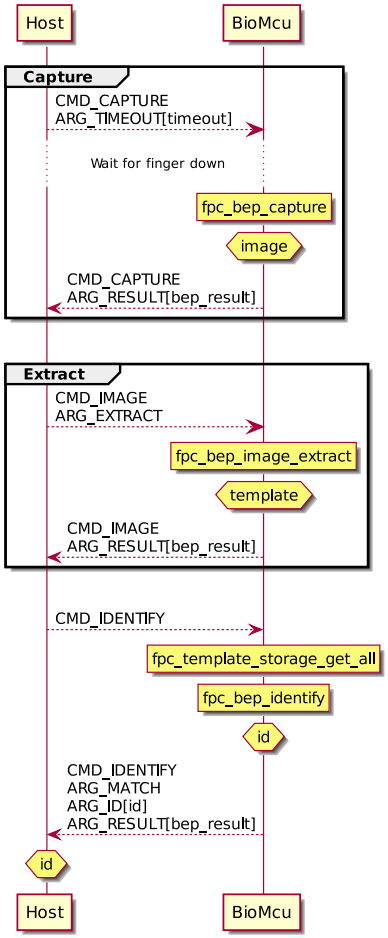
4.2 Extract


Host
BioMcu
Capture
CMD_CAPTURE
ARG_TIMEOUT[timeout]
Wait for finger down
fpc_bep_capture
image
CMD_CAPTURE
ARG_RESULT[bep_result]
CMD_IMAGE
ARG_EXTRACT
fpc_bep_image_extract
template
CMD_IMAGE
ARG_RESULT[bep_result]
Host
BioMcu

4.3 Enroll


Host
BioMcu
CMD_ENROLL
ARG_START
fpc_bep_enroll_start
session
CMD_ENROLL
ARG_RESULT[bep_result]
loop
[Enroll Image]
Capture
CMD_CAPTURE
ARG_TIMEOUT[timeout]
Wait for finger down
fpc_bep_capture
image
CMD_CAPTURE
ARG_RESULT[bep_result]
CMD_ENROLL
ARG_ADD
fpc_bep_enroll
count
CMD_ENROLL
ARG_COUNT[count]
ARG_RESULT[bep_result]
CMD_WAIT
ARG_FINGER_UP
ARG_TIMEOUT[timeout]
Wait for finger up
CMD_WAIT
ARG_RESULT[bep_result]
CMD_ENROLL
ARG_FINISH
fpc_bep_enroll_finish
enroll template
CMD_ENROLL
ARG_RESULT[bep_result]
Host
BioMcu

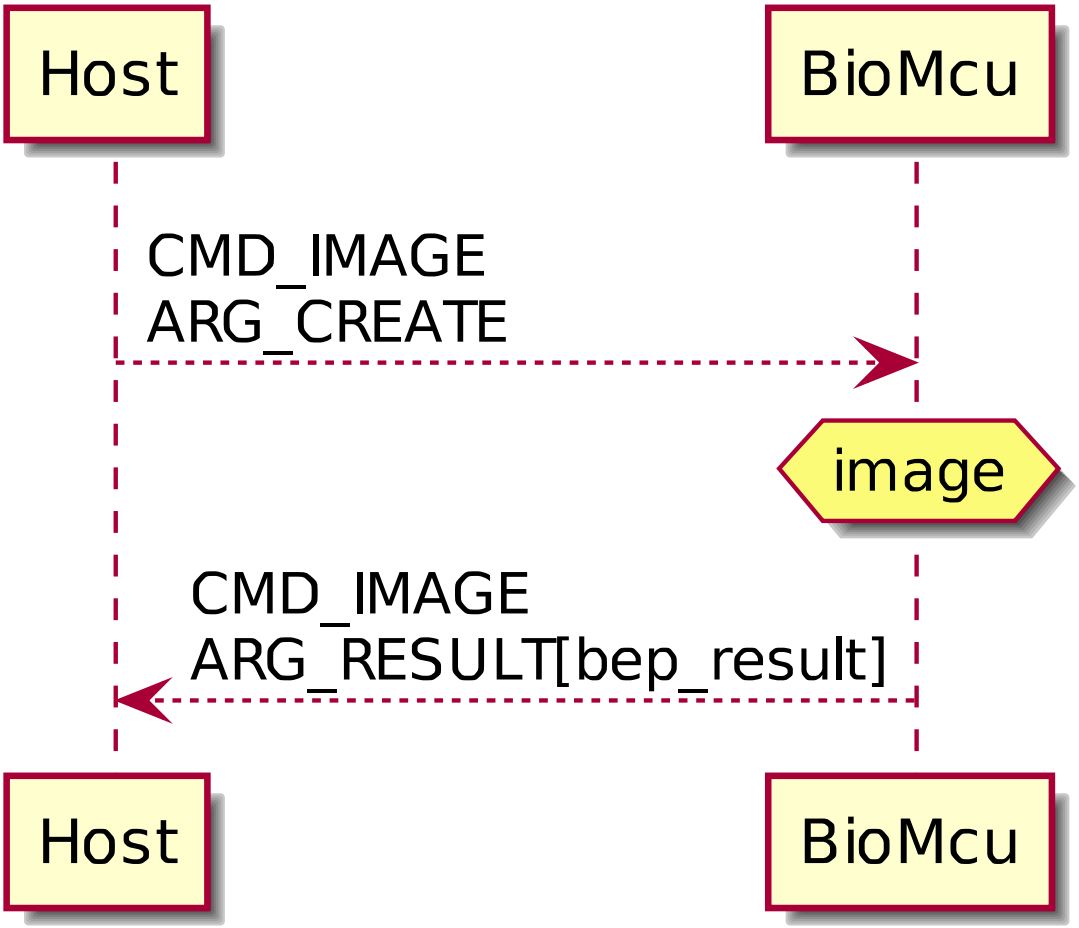
4.4 Identify


Host
BioMcu
Capture
CMD_CAPTURE
ARG_TIMEOUT[timeout]
Wait for finger down
fpc_bep_capture
image
CMD_CAPTURE
ARG_RESULT[bep_result]
Extract
CMD_IMAGE
ARG_EXTRACT
fpc_bep_image_extract
template
CMD_IMAGE
ARG_RESULT[bep_result]
CMD_IDENTIFY
fpc_template_storage_get_all
fpc_bep_identify
id
CMD_IDENTIFY
ARG_MATCH
ARG_ID[id]
ARG_RESULT[bep_result]
id
Host
BioMcu

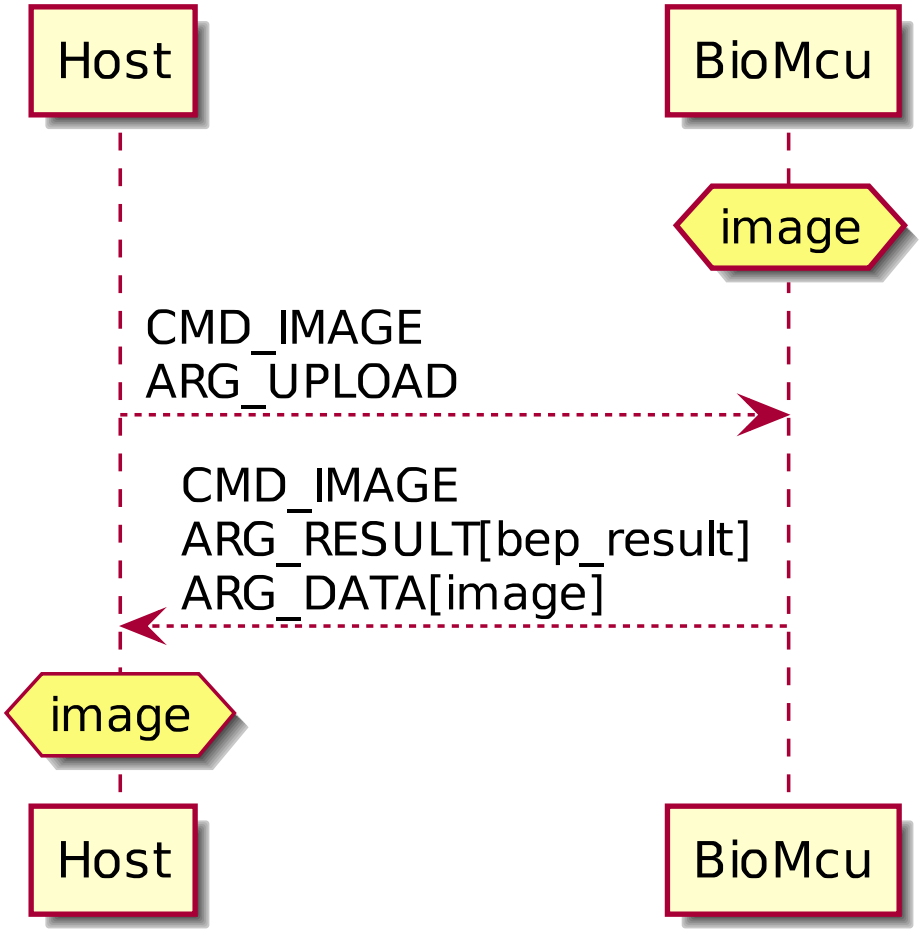
Chapter 5

Image handling

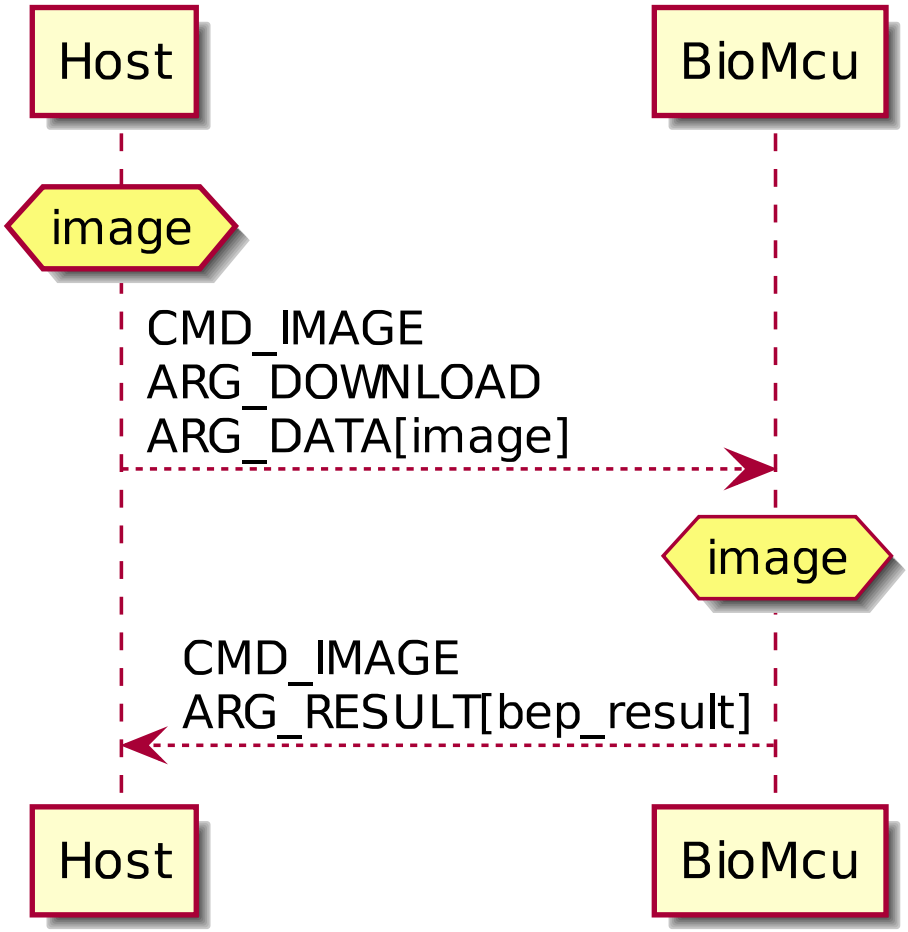
5.1 Create



5.2 Upload


Host
BioMcu
image
CMD_IMAGE
ARG_UPLOAD
CMD_IMAGE
ARG_RESULT[bep_result]
ARG_DATA[image]
image
Host
BioMcu

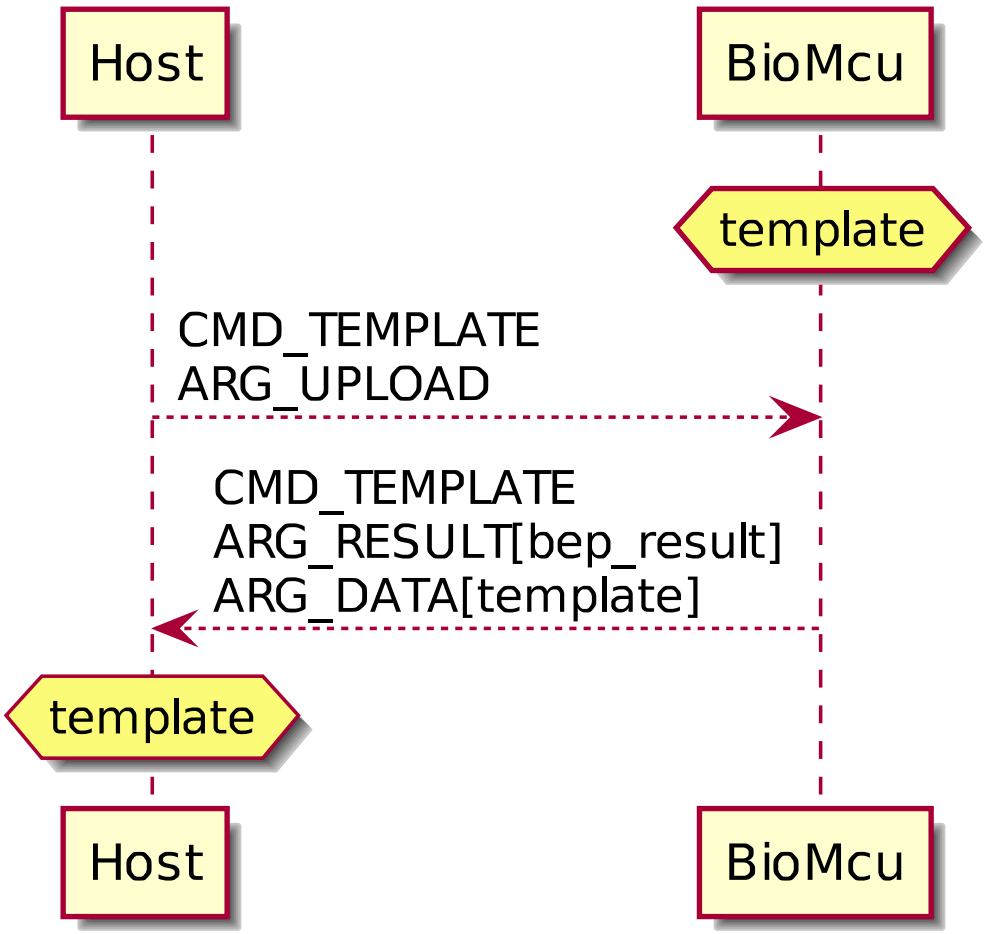
5.3 Download


Host
BioMcu
image
CMD_IMAGE
ARG_DOWNLOAD
ARG_DATA[image]
image
CMD_IMAGE
ARG_RESULT[bep_result]
Host
BioMcu

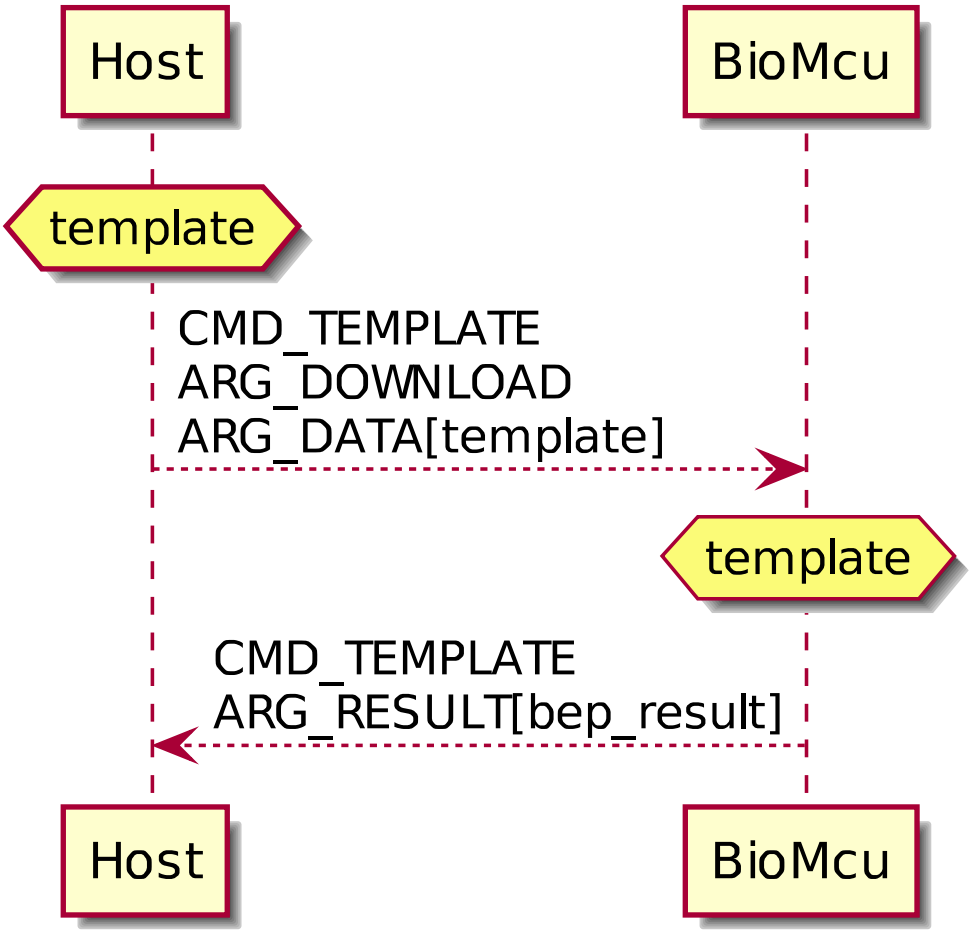
Chapter 6

Template handling

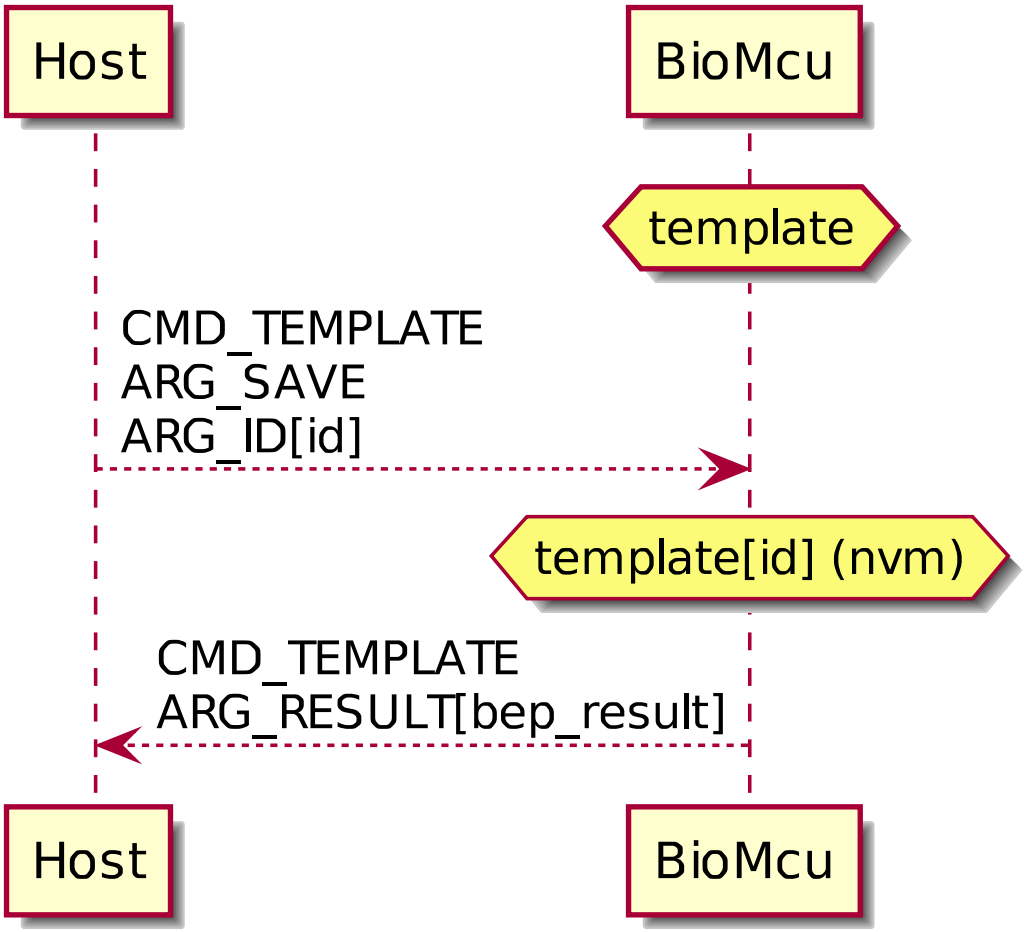
6.1 Upload



6.2 Download


Host
BioMcu
template
CMD_TEMPLATE
ARG_DOWNLOAD
ARG_DATA[template]
template
CMD_TEMPLATE
ARG_RESULT[bep_result]
Host
BioMcu

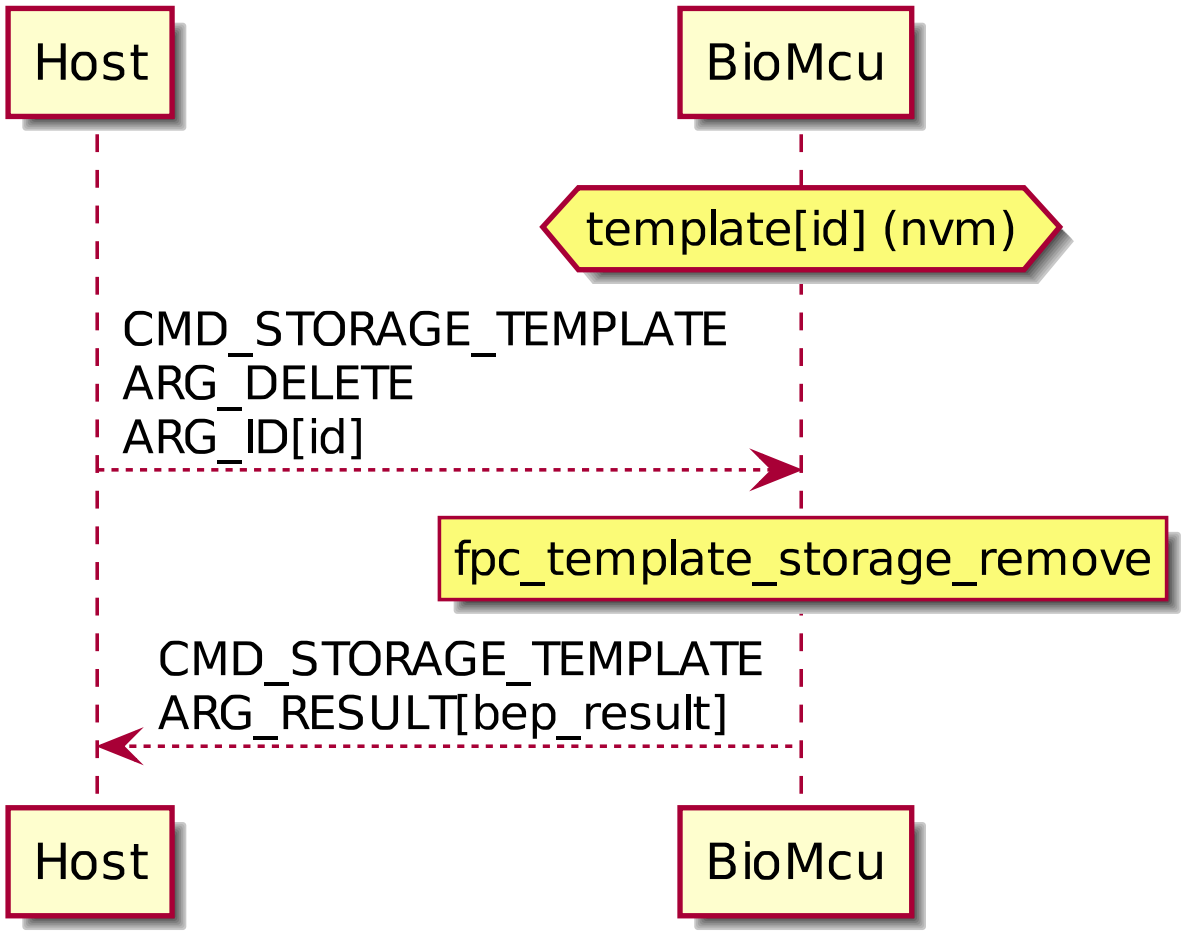
6.3 Save


Host
BioMcu
template
CMD_TEMPLATE
ARG_SAVE
ARG_ID[id]
template[id] (nvm)
CMD_TEMPLATE
ARG_RESULT[bep_result]
Host
BioMcu

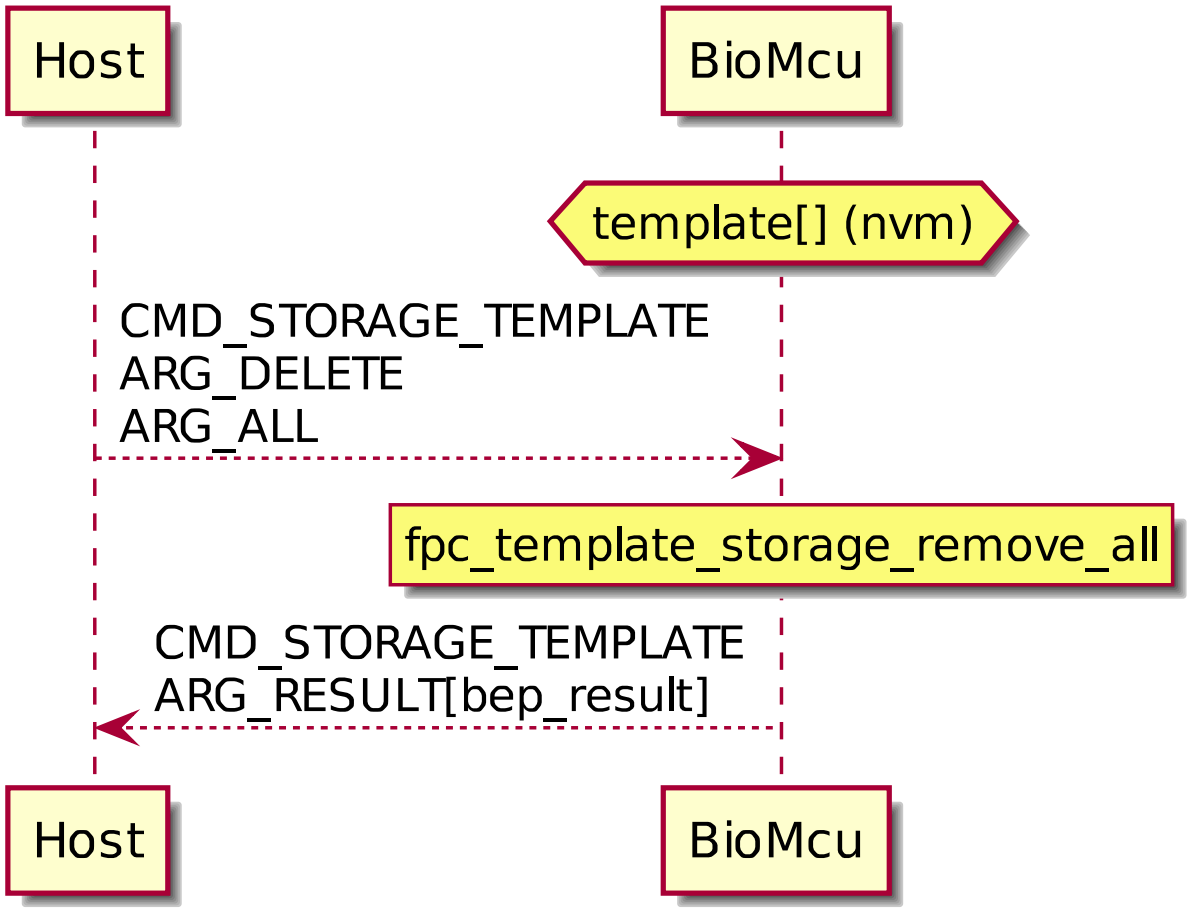
Chapter 7

Storage handling

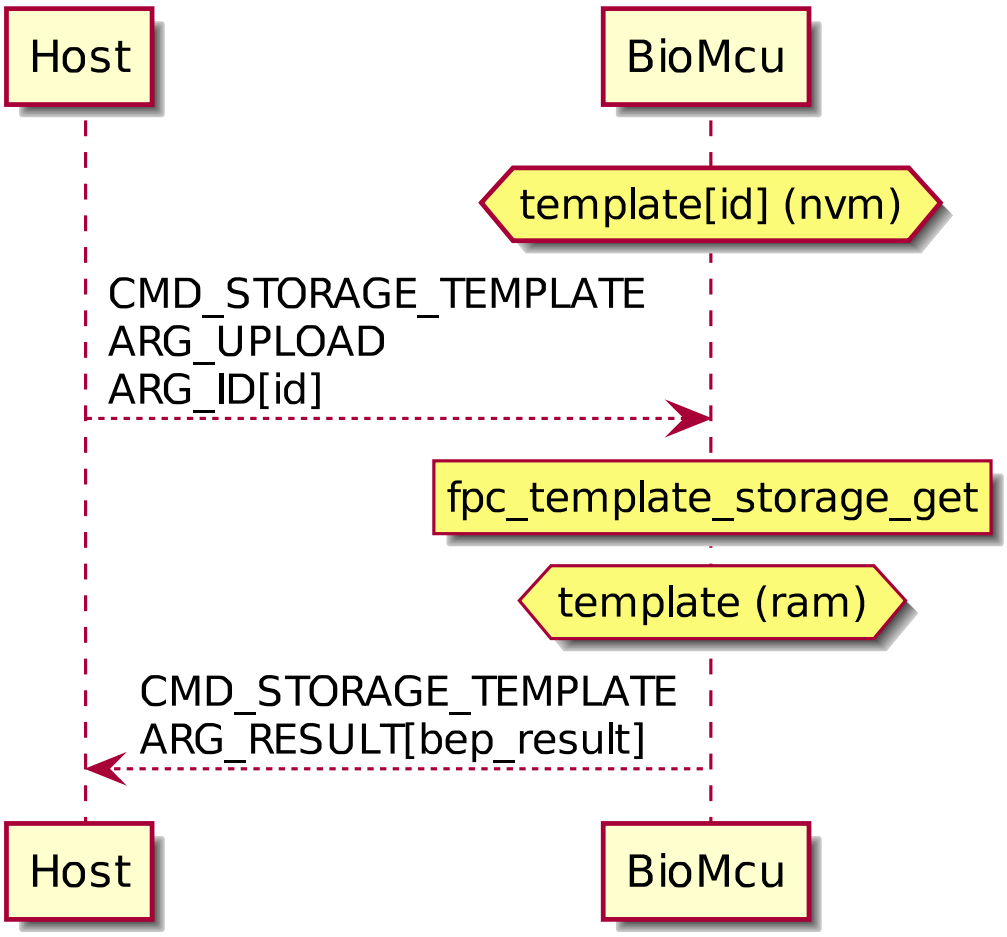
7.1 Delete ID



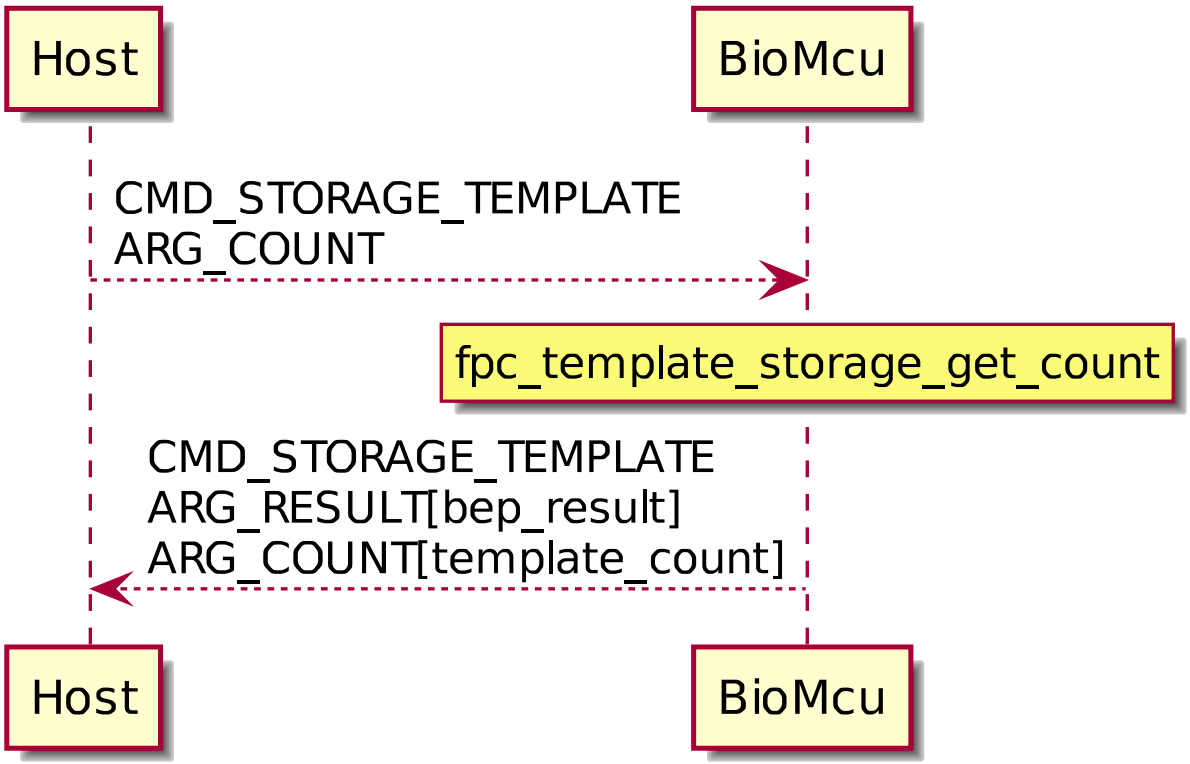
7.2 Delete All



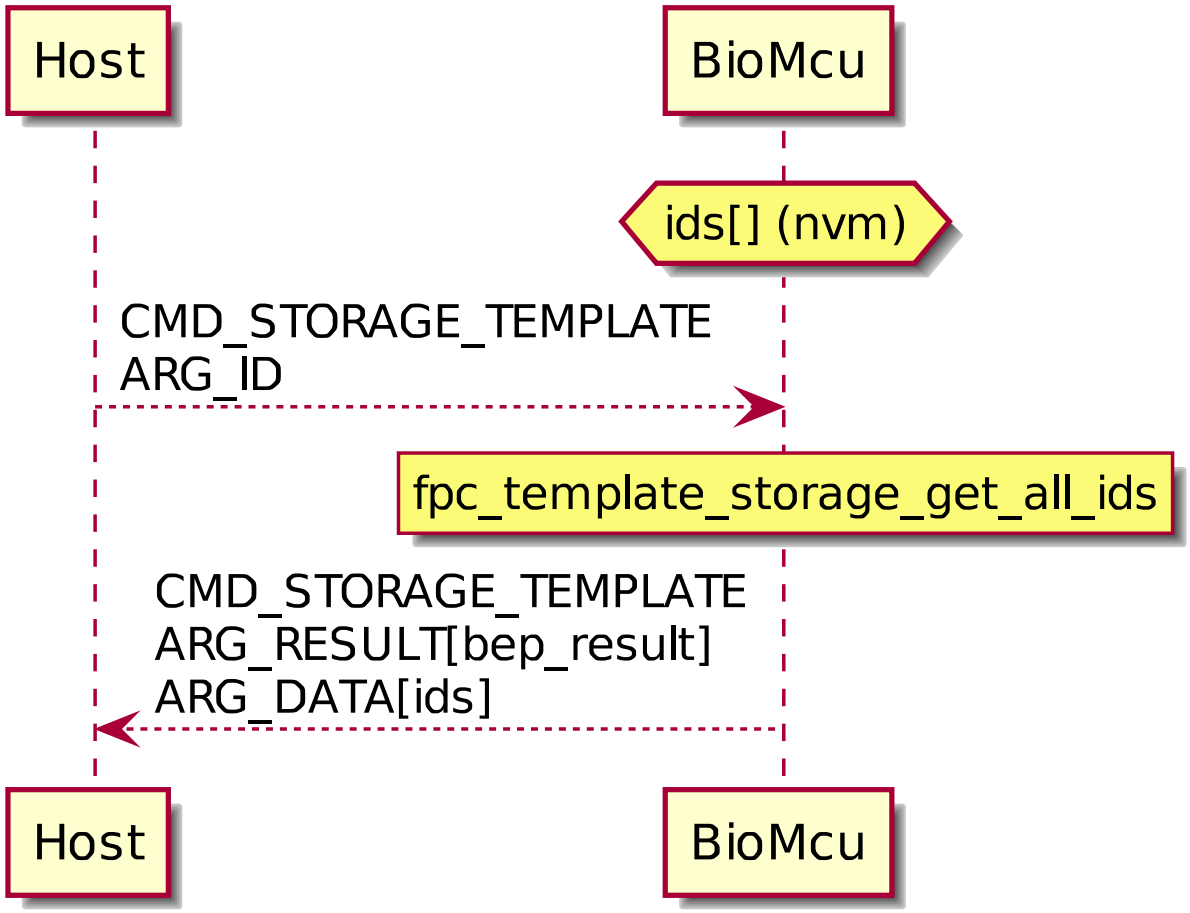
7.3 Upload


Host
BioMcu
template[id] (nvm)
CMD_STORAGE_TEMPLATE
ARG_UPLOAD
ARG_ID[id]
fpc_template_storage_get
template (ram)
CMD_STORAGE_TEMPLATE
ARG_RESULT[bep_result]
Host
BioMcu

7.4 Count


Host
BioMcu
CMD_STORAGE_TEMPLATE
ARG_COUNT
fpc_template_storage_get_count
CMD_STORAGE_TEMPLATE
ARG_RESULT[bep_result]
ARG_COUNT[template_count]
Host
BioMcu

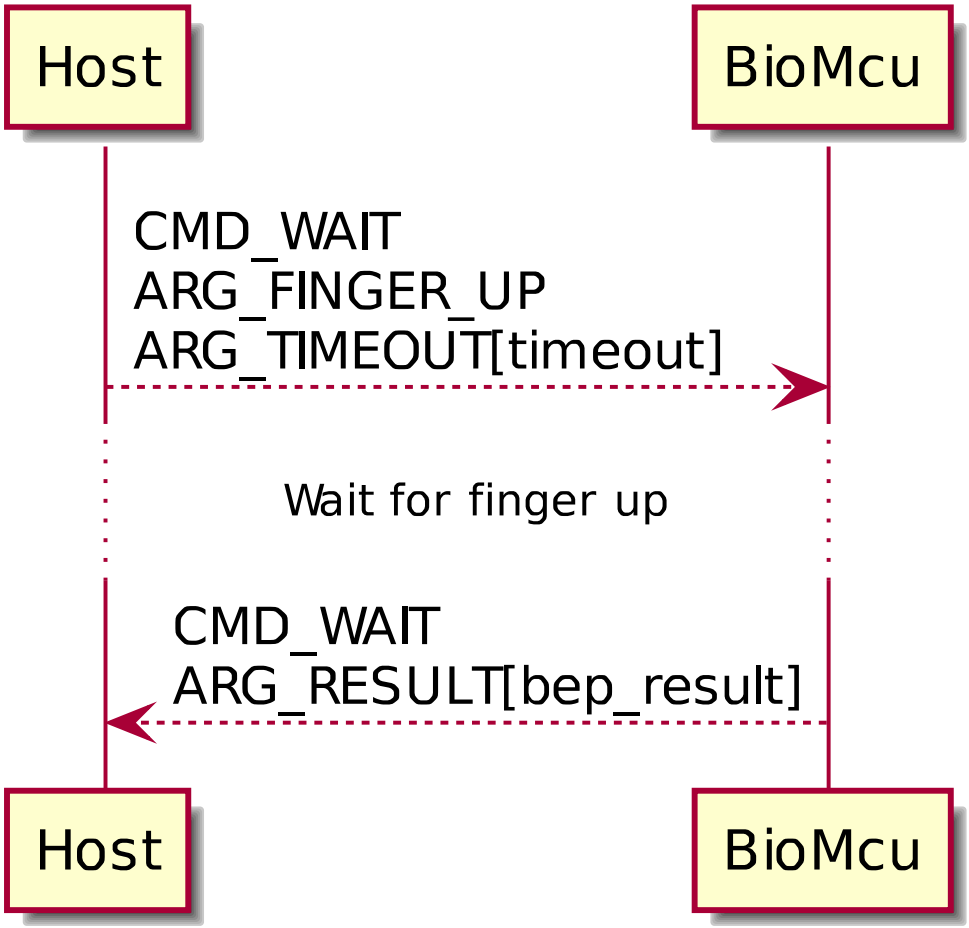
7.5 Get IDs


Host
BioMcu
ids[] (nvm)
CMD_STORAGE_TEMPLATE
ARG_ID
fpc_template_storage_get_all_ids
CMD_STORAGE_TEMPLATE
ARG_RESULT[bep_result]
ARG_DATA[ids]
Host
BioMcu

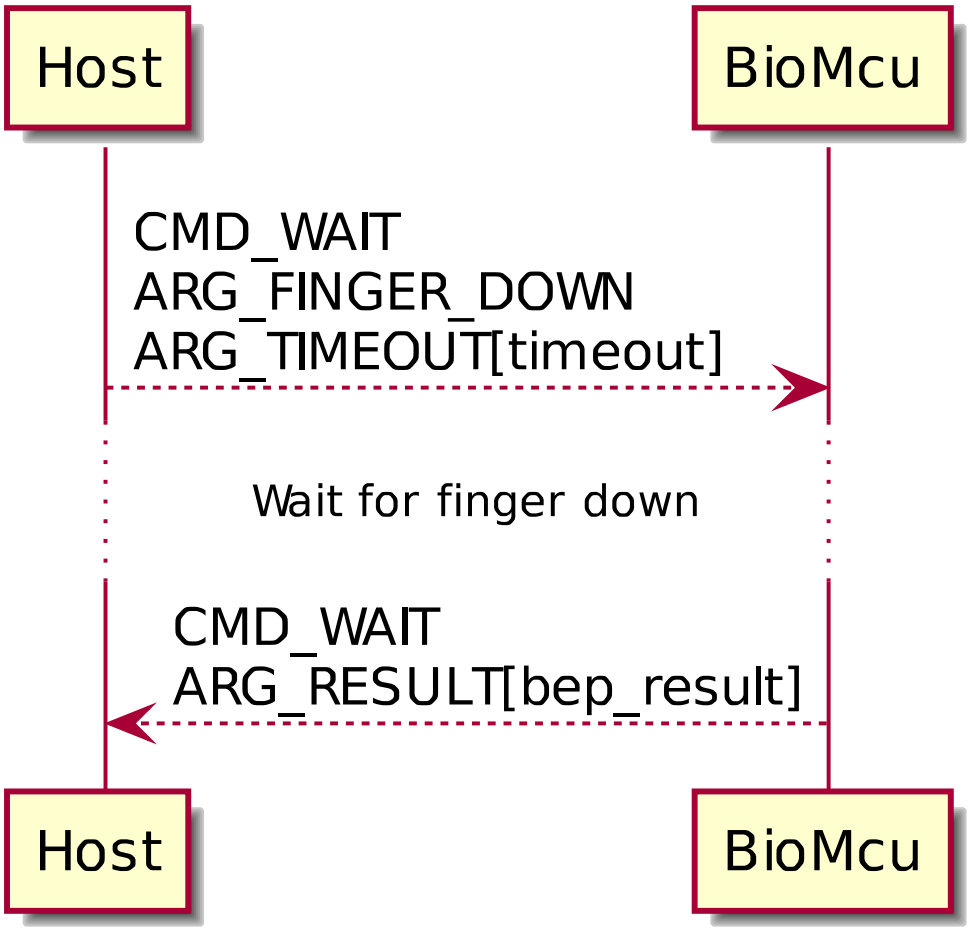
Chapter 8

Sensor operations

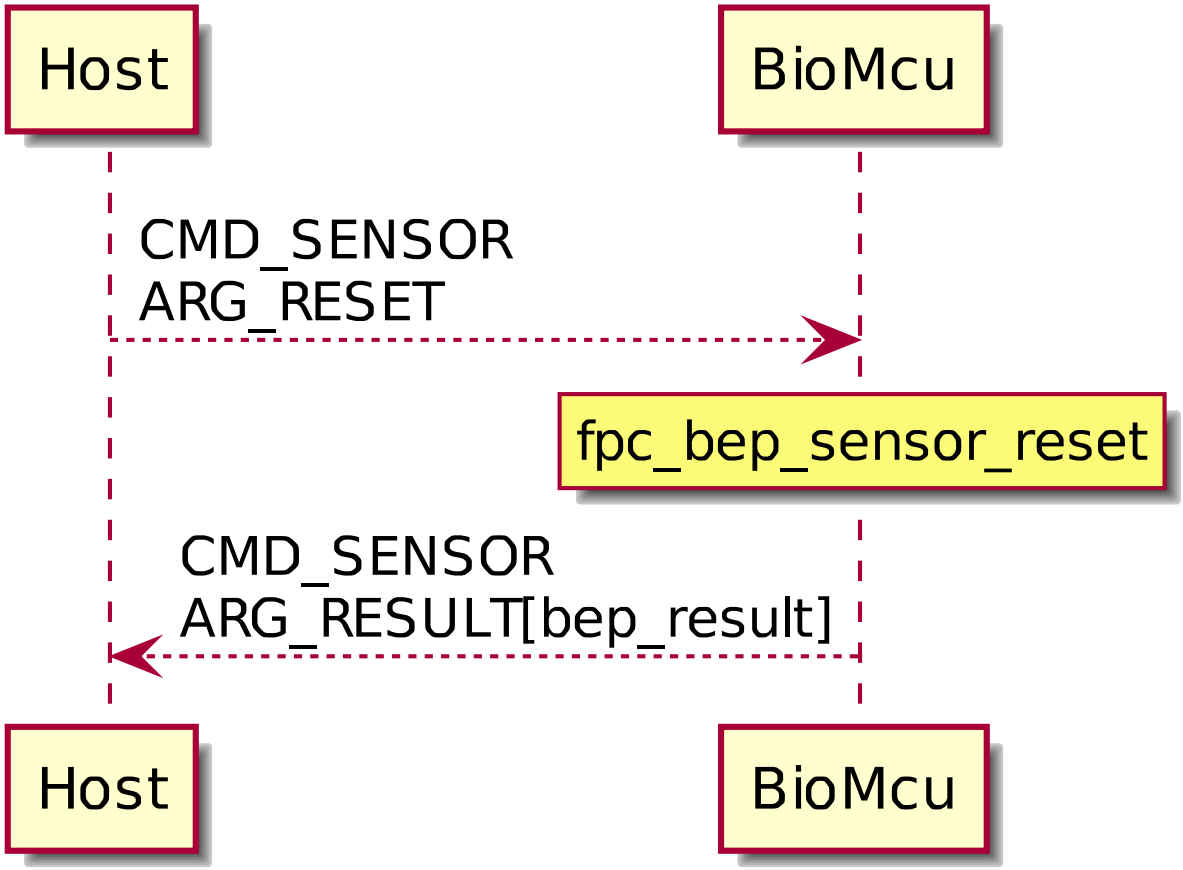
8.1 Wait for finger up



8.2 Wait for finger down



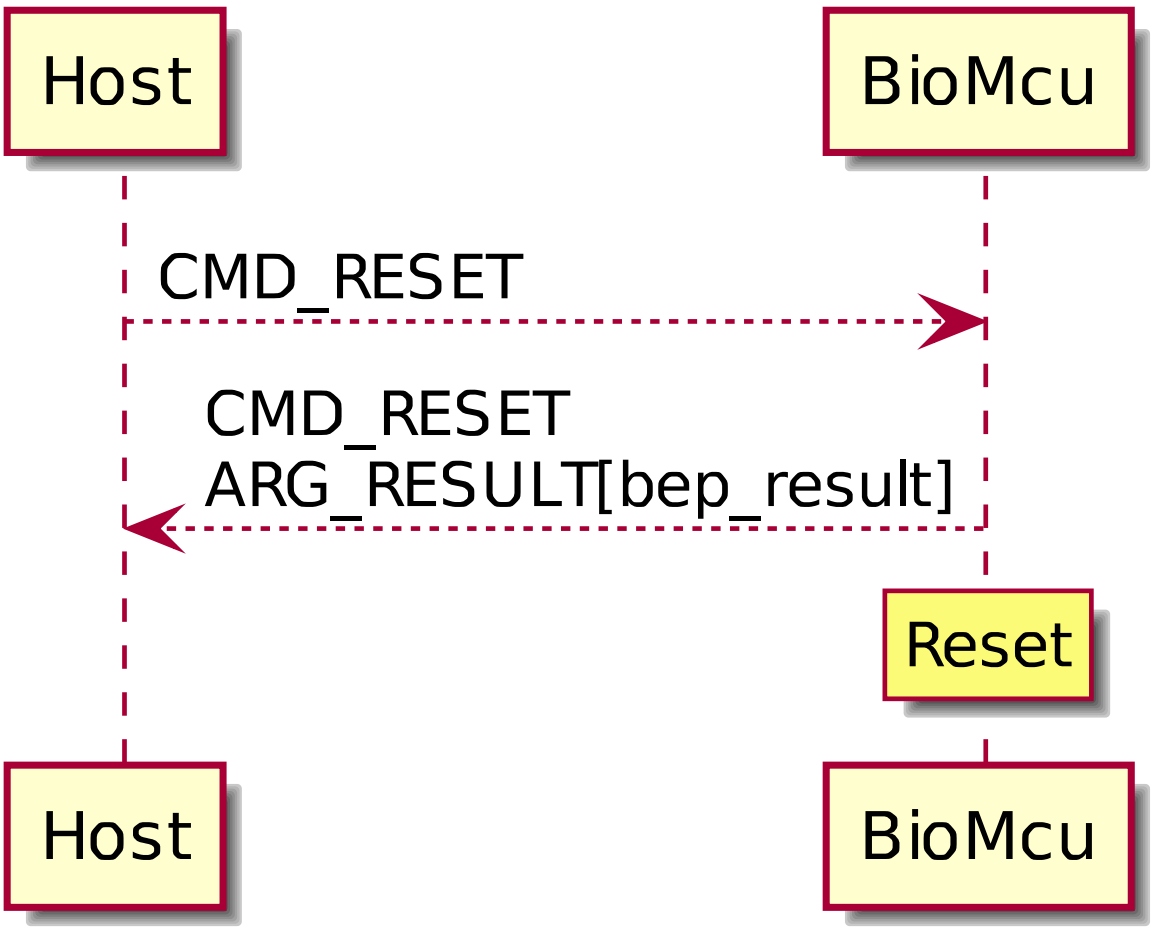
8.3 Reset sensor



Chapter 9

Device operations

9.1 Reset device



Chapter 10

Data Structure Index

10.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 11

File Index

11.1 File List

Here is a list of all files with brief descriptions:

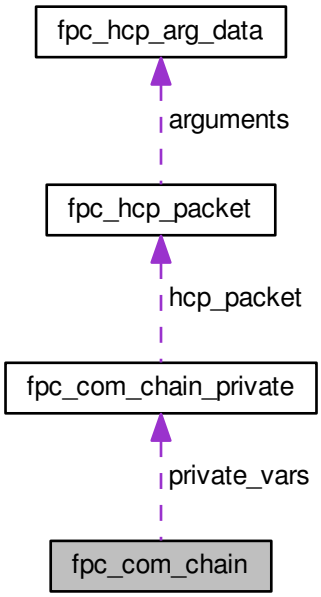
Chapter 12

Data Structure Documentation

12.1 fpc_com_chain Struct Reference

`#include <fpc_com_chain.h>`

Collaboration diagram for fpc_com_chain:



Data Fields

- bool initialized
- uint32_t(∗ crc_calc)(uint32_t start, const void ∗data, uint32_t size)
- fpc_com_chain_private_t private_vars

- void * session

  User session pointer. User private stuff, to be able to pass necessary info from the layer that calls hcp down to the user's TX and RX functions (phy_tx/rx), to enable multi threaded applications at the host side.
- void * context

  User context pointer. User private stuff, to be able to pass nessecary context to argument_allocator and argument_free.

HCP Layer

- void *(* argument_allocator)(fpc_hcp_cmd_t cmd, fpc_hcp_arg_t arg, uint16_t size, bool *free_data, void *context)
- void(* argument_free)(fpc_hcp_cmd_t cmd, fpc_hcp_arg_data_t *arg_data, void *context)

Application Layer

- fpc_com_result_t(* app_tx)(fpc_com_chain_t *chain)
- fpc_com_result_t(* app_rx)(fpc_com_chain_t *chain)
- uint16_t(* app_overhead_get)(uint16_t *offset)
- uint16_t app_packet_size [2]
- uint16_t app_mtu_size [2]
- uint8_t * app_mtu_buffer [2]

Transport Layer

- fpc_com_result_t(* tsp_tx)(fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
- fpc_com_result_t(* tsp_rx)(fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
- uint16_t(* tsp_overhead_get)(uint16_t *offset)

Link Layer

- uint16_t(* link_overhead_get)(uint16_t *offset)
- fpc_com_channel_t channel

Physical Layer

- fpc_com_result_t(* phy_tx)(uint16_t size, const uint8_t *buffer, uint32_t timeout, void *session)
- fpc_com_result_t(* phy_rx)(uint16_t size, uint8_t *buffer, uint32_t timeout, void *session)
- uint16_t phy_mtu_size [2]
- uint8_t * phy_mtu_buffer [2]
- uint32_t phy_timeout_tx
- uint32_t phy_timeout_rx

12.1.1 Detailed Description

Communication chain struct

Definition at line 50 of file fpc_com_chain.h.

12.1.2 Field Documentation

12.1.2.1 uint8_t* fpc_com_chain::app_mtu_buffer[2]

Application MTU buffers

Definition at line 83 of file fpc_com_chain.h.

12.1.2.2 uint16_t fpc_com_chain::app_mtu_size[2]

Application MTU sizes

Definition at line 81 of file fpc_com_chain.h.

12.1.2.3 uint16_t(∗ fpc_com_chain::app_overhead_get) (uint16_t ∗offset)

Application layer overhead get interface function

Definition at line 77 of file fpc_com_chain.h.

12.1.2.4 uint16_t fpc_com_chain::app_packet_size[2]

Application packet sizes

Definition at line 79 of file fpc_com_chain.h.

12.1.2.5 fpc_com_result_t(∗ fpc_com_chain::app_rx) (fpc_com_chain_t ∗chain)

Application layer receive interface function

Definition at line 75 of file fpc_com_chain.h.

12.1.2.6 fpc_com_result_t(∗ fpc_com_chain::app_tx) (fpc_com_chain_t ∗chain)

Application layer transmit interface function

Definition at line 73 of file fpc_com_chain.h.

12.1.2.7 void∗(∗ fpc_com_chain::argument_allocator) (fpc_hcp_cmd_t cmd, fpc_hcp_arg_t arg, uint16_t size, bool ∗free_data, void ∗context)

Argument allocator interface function

Definition at line 59 of file fpc_com_chain.h.

12.1.2.8 void(∗ fpc_com_chain::argument_free) (fpc_hcp_cmd_t cmd, fpc_hcp_arg_data_t ∗arg_data, void ∗context)

Argument free interface function

Definition at line 62 of file fpc_com_chain.h.

12.1.2.9 fpc_com_channel_t fpc_com_chain::channel

Communication channel

Definition at line 105 of file fpc_com_chain.h.

12.1.2.10 void* fpc_com_chain::context

User context pointer. User private stuff, to be able to pass nessecary context to argument_allocator and argument↩
_free.

Definition at line 143 of file fpc_com_chain.h.

12.1.2.11 uint32_t(* fpc_com_chain::crc_calc) (uint32_t start, const void *data, uint32_t size)

CRC calculation interface function

Definition at line 66 of file fpc_com_chain.h.

12.1.2.12 bool fpc_com_chain::initialized

Initialization status

Definition at line 52 of file fpc_com_chain.h.

12.1.2.13 uint16_t(* fpc_com_chain::link_overhead_get) (uint16_t *offset)

Link layer overhead get interface function

Definition at line 103 of file fpc_com_chain.h.

12.1.2.14 uint8_t* fpc_com_chain::phy_mtu_buffer[2]

Physical MTU buffers

Definition at line 121 of file fpc_com_chain.h.

12.1.2.15 uint16_t fpc_com_chain::phy_mtu_size[2]

Physical MTU sizes

Definition at line 119 of file fpc_com_chain.h.

12.1.2.16 fpc_com_result_t(∗ fpc_com_chain::phy_rx) (uint16_t size, uint8_t ∗buffer, uint32_t timeout, void ∗session)

Physical layer receive interface function

Definition at line 116 of file fpc_com_chain.h.

12.1.2.17 uint32_t fpc_com_chain::phy_timeout_rx

Physical receive timeout

Definition at line 125 of file fpc_com_chain.h.

12.1.2.18 uint32_t fpc_com_chain::phy_timeout_tx

Physical transmit timeout

Definition at line 123 of file fpc_com_chain.h.

12.1.2.19 fpc_com_result_t(∗ fpc_com_chain::phy_tx) (uint16_t size, const uint8_t ∗buffer, uint32_t timeout, void ∗session)

Physical layer transmit interface function

Definition at line 113 of file fpc_com_chain.h.

12.1.2.20 fpc_com_chain_private_t fpc_com_chain::private_vars

Communication change private variables

Definition at line 129 of file fpc_com_chain.h.

12.1.2.21 void∗ fpc_com_chain::session

User session pointer. User private stuff, to be able to pass necessary info from the layer that calls hcp down to the user's TX and RX functions (phy_tx/rx), to enable multi threaded applications at the host side.

Definition at line 137 of file fpc_com_chain.h.

12.1.2.22 uint16_t(∗ fpc_com_chain::tsp_overhead_get) (uint16_t ∗offset)

Transport layer overhead get interface function

Definition at line 95 of file fpc_com_chain.h.

12.1.2.23 fpc_com_result_t(∗ fpc_com_chain::tsp_rx) (fpc_com_packet_tsp_t ∗packet, fpc_com_chain_t ∗chain)

Transport layer receive interface function

Definition at line 93 of file fpc_com_chain.h.

12.1.2.24 fpc_com_result_t(∗ fpc_com_chain::tsp_tx) (fpc_com_packet_tsp_t ∗packet, fpc_com_chain_t ∗chain)

Transport layer transmit interface function

Definition at line 91 of file fpc_com_chain.h.

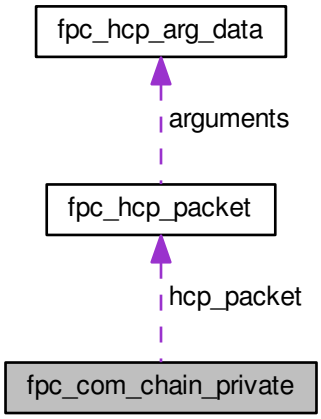
The documentation for this struct was generated from the following file:

- inc/fpc_com_chain.h

12.2 fpc_com_chain_private Struct Reference

`#include <fpc_com_chain.h>`

Collaboration diagram for fpc_com_chain_private:



Data Fields

- fpc_hcp_packet_t ∗ hcp_packet
- uint16_t hcp_seq_len
- uint16_t hcp_seq_nr

12.2.1 Detailed Description

Communication chain private struct

Definition at line 36 of file fpc_com_chain.h.

12.2.2 Field Documentation

12.2.2.1 fpc_hcp_packet_t* fpc_com_chain_private::hcp_packet

HCP packet

Definition at line 38 of file fpc_com_chain.h.

12.2.2.2 uint16_t fpc_com_chain_private::hcp_seq_len

HCP sequence length

Definition at line 40 of file fpc_com_chain.h.

12.2.2.3 uint16_t fpc_com_chain_private::hcp_seq_nr

HCP sequence number

Definition at line 42 of file fpc_com_chain.h.

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

- inc/fpc_com_chain.h

12.3 fpc_com_packet_link Struct Reference

```
#include <fpc_com_packets.h>
```

Data Fields

- fpc_com_channel_t channel
- uint16_t size
- uint8_t * data
- uint32_t crc

12.3.1 Detailed Description

Link layer packet

Definition at line 61 of file fpc_com_packets.h.

12.3.2 Field Documentation

12.3.2.1 fpc_com_channel_t fpc_com_packet_link::channel

Communication channel

Definition at line 63 of file fpc_com_packets.h.

12.3.2.2 uint32_t fpc_com_packet_link::crc

CRC of data

Definition at line 69 of file fpc_com_packets.h.

12.3.2.3 uint8_t* fpc_com_packet_link::data

Packet data

Definition at line 67 of file fpc_com_packets.h.

12.3.2.4 uint16_t fpc_com_packet_link::size

Size of packet

Definition at line 65 of file fpc_com_packets.h.

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

- inc/fpc_com_packets.h

12.4 fpc_com_packet_transport Struct Reference

```
#include <fpc_com_packets.h>
```

Data Fields

- uint16_t size
- uint16_t seq_len
- uint16_t seq_nr
- uint8_t * data

12.4.1 Detailed Description

Transport layer packet.

Definition at line 37 of file fpc_com_packets.h.

12.4.2 Field Documentation

12.4.2.1 uint8_t* fpc_com_packet_transport::data

Packet data

Definition at line 45 of file fpc_com_packets.h.

12.4.2.2 uint16_t fpc_com_packet_transport::seq_len

Sequence length

Definition at line 41 of file fpc_com_packets.h.

12.4.2.3 uint16_t fpc_com_packet_transport::seq_nr

Sequence number

Definition at line 43 of file fpc_com_packets.h.

12.4.2.4 uint16_t fpc_com_packet_transport::size

Size of packet

Definition at line 39 of file fpc_com_packets.h.

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

- inc/fpc_com_packets.h

12.5 fpc_hcp_arg_data Struct Reference

Command Argument.

```
#include <fpc_hcp_common.h>
```

Data Fields

- fpc_hcp_arg_t arg
- uint16_t size
- bool free_data
- uint8_t * data

12.5.1 Detailed Description

Command Argument.

Definition at line 196 of file fpc_hcp_common.h.

12.5.2 Field Documentation

12.5.2.1 fpc_hcp_arg_t fpc_hcp_arg_data::arg

Argument

Definition at line 198 of file fpc_hcp_common.h.

12.5.2.2 uint8_t* fpc_hcp_arg_data::data

Pointer to data

Definition at line 204 of file fpc_hcp_common.h.

12.5.2.3 bool fpc_hcp_arg_data::free_data

Free data inside HCP

Definition at line 202 of file fpc_hcp_common.h.

12.5.2.4 uint16_t fpc_hcp_arg_data::size

Size of data

Definition at line 200 of file fpc_hcp_common.h.

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

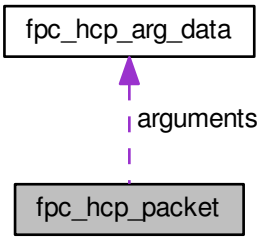
- inc/fpc_hcp_common.h

12.6 fpc_hcp_packet Struct Reference

Application Command Packet.

`#include <fpc_hcp_common.h>`

Collaboration diagram for fpc_hcp_packet:



Data Fields

- fpc_hcp_cmd_t id
- uint16_t num_args
- fpc_hcp_arg_data_t * arguments

12.6.1 Detailed Description

Application Command Packet.

Definition at line 210 of file fpc_hcp_common.h.

12.6.2 Field Documentation

12.6.2.1 fpc_hcp_arg_data_t* fpc_hcp_packet::arguments

Pointer to argument data

Definition at line 216 of file fpc_hcp_common.h.

12.6.2.2 fpc_hcp_cmd_t fpc_hcp_packet::id

Command ID

Definition at line 212 of file fpc_hcp_common.h.

12.6.2.3 uint16_t fpc_hcp_packet::num_args

Number of arguments

Definition at line 214 of file fpc_hcp_common.h.

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

- inc/fpc_hcp_common.h

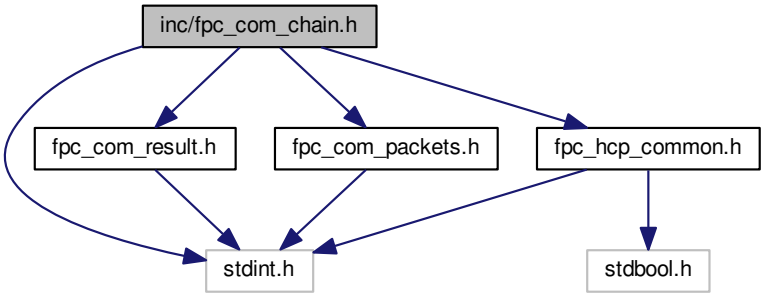
Chapter 13

File Documentation

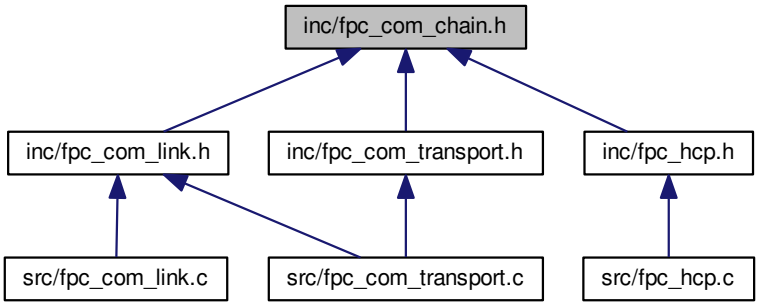
Communication chain type definitions.

```
#include <stdint.h>
#include "fpc_com_result.h"
#include "fpc_hcp_common.h"
#include "fpc_com_packets.h"
```

Include dependency graph for fpc_com_chain.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct fpc_com_chain_private
- struct fpc_com_chain

Typedefs

- typedef struct fpc_com_chain_private fpc_com_chain_private_t

  Communication chain private variables.
- typedef struct fpc_com_chain fpc_com_chain_t

  Communication chain.

Enumerations

- enum fpc_com_chain_dir_t {
  FPC_COM_CHAIN_TX = 0,
  FPC_COM_CHAIN_RX = 1 }

  Communication chain direction type.

13.10.1 Detailed Description

Communication chain type definitions.

13.10.2 Typedef Documentation

13.10.2.1 typedef struct fpc_com_chain_private fpc_com_chain_private_t

Communication chain private variables.

Definition at line 34 of file fpc_com_chain.h.

13.10.2.2 typedef struct fpc_com_chain fpc_com_chain_t

Communication chain.

Definition at line 48 of file fpc_com_chain.h.

13.10.3 Enumeration Type Documentation

13.10.3.1 enum fpc_com_chain_dir_t

Communication chain direction type.

Enumerator

FPC_COM_CHAIN_TX

FPC_COM_CHAIN_RX

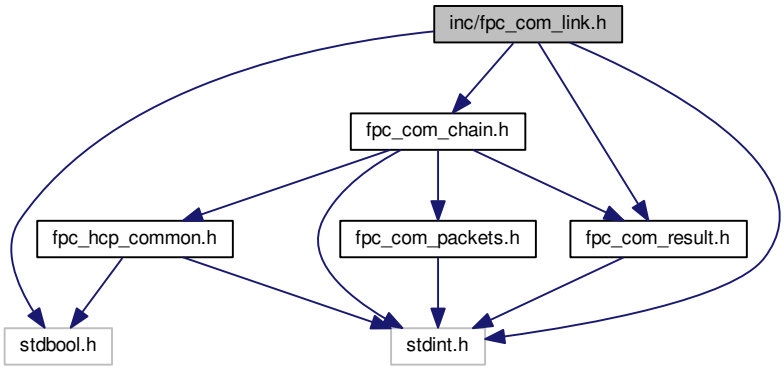
Definition at line 149 of file fpc_com_chain.h.

13.11 inc/fpc_com_link.h File Reference

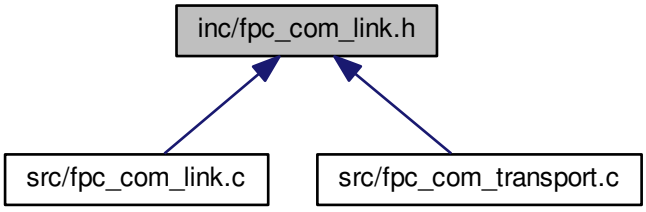
Communication link interface.

```
#include <stdbool.h>
#include <stdint.h>
#include "fpc_com_result.h"
#include "fpc_com_chain.h"
```

Include dependency graph for fpc_com_link.h:



This graph shows which files directly or indirectly include this file:



Functions

- fpc_com_result_t fpc_com_link_transmit (fpc_com_packet_link_t *packet, fpc_com_chain_t *chain)

  Sends a packet over the physical link in blocking mode.
- fpc_com_result_t fpc_com_link_receive (fpc_com_packet_link_t *packet, fpc_com_chain_t *chain)

  Receives a packet from the physical link.
- uint16_t fpc_com_link_get_overhead (uint16_t *offset)

  Returns the overhead of the layer.

13.11.1 Detailed Description

Communication link interface.

13.11.2 Function Documentation

13.11.2.1 uint16_t fpc_com_link_get_overhead (uint16_t * *offset*)

Returns the overhead of the layer.

Parameters

out	*offset*	The offset to the packet data.

Returns

Overhead size in bytes.

Definition at line 126 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, and fpc_com_packet_link::size.

13.11.2.2 fpc_com_result_t fpc_com_link_receive (fpc_com_packet_link_t * *packet*, fpc_com_chain_t * *chain*)

Receives a packet from the physical link.

Parameters

in,out	*packet*	Packet to populate.
in	*chain*	The communication chain to use.

Returns

fpc_com_result_t

Definition at line 73 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, fpc_com_chain::crc_calc, fpc_com_packet_link::data, FPC_COM_ACK, FPC_COM_CHAIN_RX, FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_IO_ERROR, FPC_COM_RESULT_OK, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain::phy_rx, fpc_com_chain::phy_timeout_rx, fpc_com_chain::phy_timeout_tx, fpc_com_chain::phy_tx, fpc_com_chain::session, and fpc_com_packet_link::size.

13.11.2.3 fpc_com_result_t fpc_com_link_transmit (fpc_com_packet_link_t * *packet*, fpc_com_chain_t * *chain*)

Sends a packet over the physical link in blocking mode.

Parameters

in	*packet*	Packet to transmit.
in	*chain*	The communication chain to use.

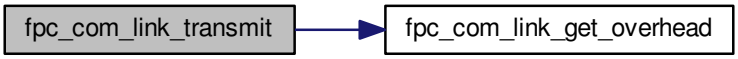
Returns

fpc_com_result_t

Definition at line 27 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, fpc_com_chain::crc_calc, fpc_com_packet_link::data, FPC_COM_ACK, FPC_COM_CHAIN_TX, fpc_com_link_get_overhead(), FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_IO_ERROR, FPC_COM_RESULT_OK, FPC_COM_RESULT_TIMEOUT, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_rx, fpc_com_chain::phy_timeout_rx, fpc_com_chain::phy_timeout_tx, fpc_com_chain::phy_tx, fpc_com_chain::session, and fpc_com_packet_link::size.

Here is the call graph for this function:

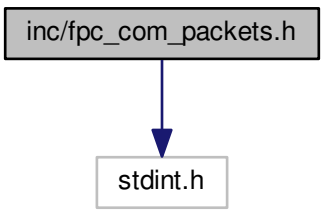


13.12 inc/fpc_com_packets.h File Reference

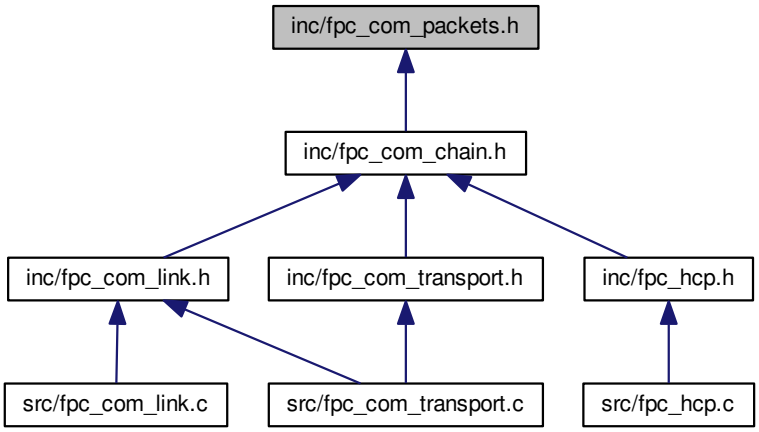
Communication packet type definitions.

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

Include dependency graph for fpc_com_packets.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct fpc_com_packet_transport
- struct fpc_com_packet_link

Macros

- #define FPC_COM_ACK 0x7f01ff7f

Typedefs

- typedef struct fpc_com_packet_transport fpc_com_packet_tsp_t
- typedef uint16_t fpc_com_channel_t
- typedef struct fpc_com_packet_link fpc_com_packet_link_t

Enumerations

- enum fpc_com_channel {
  FPC_COM_CHANNEL_NONE = 0x00,
  FPC_COM_CHANNEL_CLEAR = 0x01,
  FPC_COM_CHANNEL_TLS = 0x02,
  FPC_COM_CHANNEL_END = 0xFF }

13.12.1 Detailed Description

Communication packet type definitions.

13.12.2 Macro Definition Documentation

13.12.2.1 #define FPC_COM_ACK 0x7f01ff7f

Communication acknowledge definition

Definition at line 32 of file fpc_com_packets.h.

13.12.3 Typedef Documentation

13.12.3.1 typedef uint16_t fpc_com_channel_t

Communication channel type

Definition at line 58 of file fpc_com_packets.h.

13.12.3.2 typedef struct fpc_com_packet_link fpc_com_packet_link_t

Link layer packet

13.12.3.3 typedef struct fpc_com_packet_transport fpc_com_packet_tsp_t

Transport layer packet.

13.12.4 Enumeration Type Documentation

13.12.4.1 enum fpc_com_channel

Transport packet channels.

Enumerator

FPC_COM_CHANNEL_NONE

FPC_COM_CHANNEL_CLEAR

FPC_COM_CHANNEL_TLS

FPC_COM_CHANNEL_END

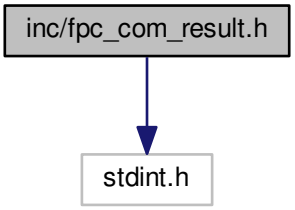
Definition at line 51 of file fpc_com_packets.h.

13.13 inc/fpc_com_result.h File Reference

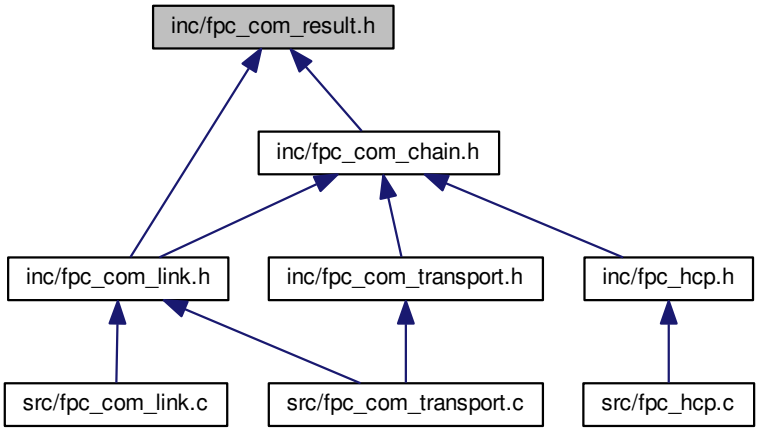
Communication result type definitions.

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

Include dependency graph for fpc_com_result.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef uint8_t fpc_com_result_t

Enumerations

- enum fpc_com_result {
  FPC_COM_RESULT_OK,
  FPC_COM_RESULT_NO_MEMORY,
  FPC_COM_RESULT_INVALID_ARGUMENT,
  FPC_COM_RESULT_NOT_IMPLEMENTED,
  FPC_COM_RESULT_IO_ERROR,
  FPC_COM_RESULT_TIMEOUT }

13.13.1 Detailed Description

Communication result type definitions.

13.13.2 Typedef Documentation

13.13.2.1 typedef uint8_t fpc_com_result_t

Communication result type

Definition at line 41 of file fpc_com_result.h.

13.13.3 Enumeration Type Documentation

13.13.3.1 enum fpc_com_result

Communication result codes

Enumerator

FPC_COM_RESULT_OK
FPC_COM_RESULT_NO_MEMORY
FPC_COM_RESULT_INVALID_ARGUMENT
FPC_COM_RESULT_NOT_IMPLEMENTED
FPC_COM_RESULT_IO_ERROR
FPC_COM_RESULT_TIMEOUT

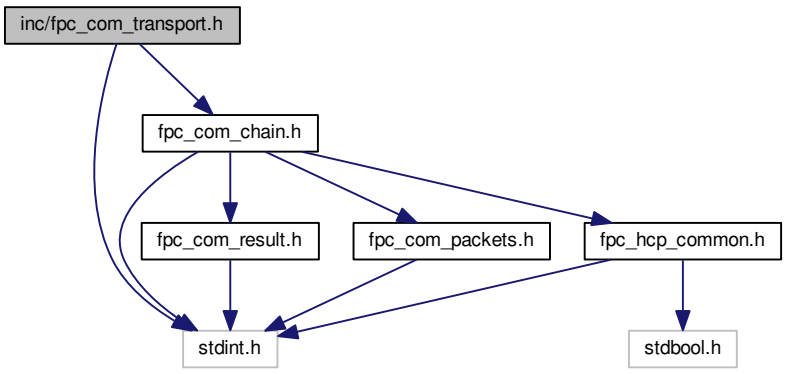
Definition at line 32 of file fpc_com_result.h.

13.14 inc/fpc_com_transport.h File Reference

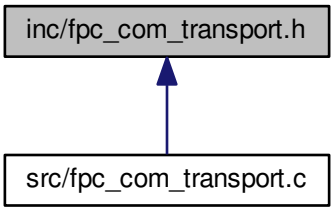
Communication transport interface.

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

Include dependency graph for fpc_com_transport.h:



This graph shows which files directly or indirectly include this file:



Functions

- fpc_com_result_t fpc_com_transport_transmit (fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
  Transmit a transport layer packet.
- fpc_com_result_t fpc_com_transport_receive (fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
  Receive a transport layer packet.
- uint16_t fpc_com_transport_get_overhead (uint16_t *offset)
  Returns the overhead of the layer.

13.14.1 Detailed Description

Communication transport interface.

13.14.2 Function Documentation

13.14.2.1 uint16_t fpc_com_transport_get_overhead (uint16_t * *offset*)

Returns the overhead of the layer.

Parameters

out	*offset*	The offset to the packet data.

Returns

Overhead size in bytes.

Definition at line 88 of file fpc_com_transport.c.

References fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, and fpc_com_packet_↩ transport::size.

13.14.2.2 fpc_com_result_t fpc_com_transport_receive (fpc_com_packet_tsp_t * *packet*, fpc_com_chain_t * *chain*)

Receive a transport layer packet.

Parameters

in,out	*packet*	The packet to populate.
in	*chain*	The chain to use.

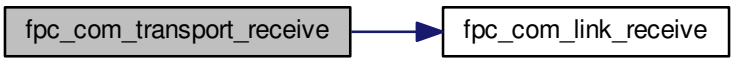
Returns

fpc_com_result_t

Definition at line 60 of file fpc_com_transport.c.

References fpc_com_packet_transport::data, fpc_com_packet_link::data, fpc_com_link_receive(), FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_OK, fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, and fpc_com_packet_transport::size.

Here is the call graph for this function:



13.14.2.3 fpc_com_result_t fpc_com_transport_transmit (fpc_com_packet_tsp_t * *packet*, fpc_com_chain_t * *chain*)

Transmit a transport layer packet.

Parameters

in	*packet*	The packet to transmit.
in	*chain*	The chain to use.

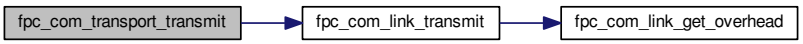
Returns

fpc_com_result_t

Definition at line 28 of file fpc_com_transport.c.

References fpc_com_packet_link::channel, fpc_com_chain::channel, fpc_com_packet_link::data, FPC_COM_CHAIN_TX, fpc_com_link_transmit(), FPC_COM_RESULT_INVALID_ARGUMENT, fpc_com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, fpc_com_packet_transport::size, fpc_com_packet_link::size, and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:

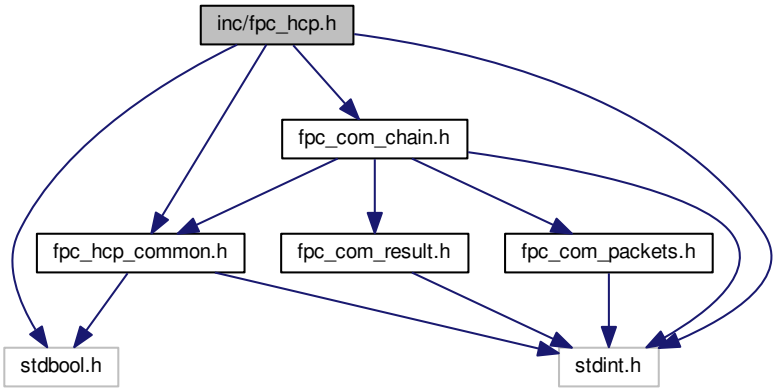


13.15 inc/fpc_hcp.h File Reference

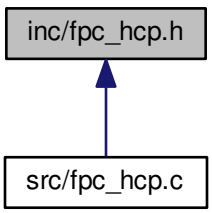
Host Communication Protocol interface.

```
#include <stdbool.h>
#include <stdint.h>
#include "fpc_hcp_common.h"
#include "fpc_com_chain.h"
```

Include dependency graph for fpc_hcp.h:



This graph shows which files directly or indirectly include this file:



Functions

- fpc_com_result_t fpc_hcp_transmit (fpc_hcp_packet_t *packet, fpc_com_chain_t *chain)

  Transmits an application packet through the supplied transmit chain.
- fpc_com_result_t fpc_hcp_receive (fpc_hcp_packet_t *packet, fpc_com_chain_t *chain)

  Receives an application packet through the supplied transmit chain.
- bool fpc_hcp_arg_add (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg, uint16_t size, bool free_data, void *data)

  Add argument to packet.
- bool fpc_hcp_arg_check (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg)

  Check if command contains selected argument key.
- fpc_hcp_arg_data_t * fpc_hcp_arg_get (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg)

  Get Argument with specified key.
- bool fpc_hcp_arg_copy_data (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg, uint16_t data_size, uint8_t *data)

  Copy data from an argument with specified key.
- void fpc_hcp_free (fpc_com_chain_t *chain, fpc_hcp_packet_t *packet)

  Frees the resources held by the packet i.e. the dynamic data held in the arguments.
- uint16_t fpc_hcp_get_size (fpc_hcp_packet_t *packet, uint16_t *num_args)

  Calculate serialized packet size.

13.15.1 Detailed Description

Host Communication Protocol interface.

13.15.2 Function Documentation

13.15.2.1 bool fpc_hcp_arg_add (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*, uint16_t *size*, bool *free_data*, void * *data*)

Add argument to packet.

Note

This function does not allocate any memory, it will only set the argument variables.

Parameters

in	*packet*	Packet to add to.
in	*arg*	Argument id.
in	*size*	Size of argument data.
in	*free_data*	Set to true if data should be owned by the argument, false if user still owns data.
in	*data*	Pointer to argument data.

Returns

true = success, false = failure.

Definition at line 145 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, fpc_hcp_packet::arguments, fpc_hcp_arg_data::data, fpc_hcp↩_arg_data::free_data, fpc_hcp_packet::num_args, and fpc_hcp_arg_data::size.

13.15.2.2 bool fpc_hcp_arg_check (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*)

Check if command contains selected argument key.

Parameters

in	*packet*	The packet to scan.
in	*arg*	Argument to look for.

Returns

true if found, false if not found.

Definition at line 169 of file fpc_hcp.c.

References fpc_hcp_arg_get().

Here is the call graph for this function:

13.15.2.3 bool fpc_hcp_arg_copy_data (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*, uint16_t *data_size*, uint8_t * *data*)

Copy data from an argument with specified key.

Argument data will be copied to specified data buffer. Remaining bytes in data will be cleared if the argument data size is less than data size when the argument contains data.

Parameters

in	*packet*	The packet to operate on.
in	*arg*	The arg to retrieve data from.
in	*data_size*	Number of bytes to copy.
in,out	*data*	Pointer to data buffer.

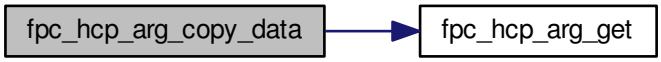
Returns

True if argument found, false if not found.

Definition at line 183 of file fpc_hcp.c.

References fpc_hcp_arg_data::data, fpc_hcp_arg_get(), and fpc_hcp_arg_data::size.

Here is the call graph for this function:



13.15.2.4 fpc_hcp_arg_data_t* fpc_hcp_arg_get (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*)

Get Argument with specified key.

Parameters

in	*packet*	The packet to operate on.
in	*arg*	The arg to retrieve.

Returns

Pointer to fpc_hcp_arg_data_t is successful, otherwise NULL.

Definition at line 173 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, fpc_hcp_packet::arguments, and fpc_hcp_packet::num_args.

13.15.2.5 void fpc_hcp_free (fpc_com_chain_t * *chain*, fpc_hcp_packet_t * *packet*)

Frees the resources held by the packet i.e. the dynamic data held in the arguments.

Parameters

in	*chain*	Pointer to the communication chain used to retrieve the packet.
in	*packet*	Pointer to packet.

Definition at line 198 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, fpc_com_chain::argument_free, fpc_hcp_packet::arguments, CMD_NONE, fpc_com_chain::context, fpc_hcp_packet::id, and fpc_hcp_packet::num_args.

13.15.2.6 uint16_t fpc_hcp_get_size (fpc_hcp_packet_t * *packet*, uint16_t * *num_args*)

Calculate serialized packet size.

Parameters

in	*packet*	Packet to calculate.
in,out	*num_args*	Will return number of arguments held by the command can be set to NULL.

Returns

Serialized size.

Definition at line 64 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, ARGUMENT_HEADER_SIZE, fpc_hcp_packet::arguments, fpc_hcp_packet::num_args, PACKET_HEADER_SIZE, and fpc_hcp_arg_data::size.

13.15.2.7 fpc_com_result_t fpc_hcp_receive (fpc_hcp_packet_t * *packet*, fpc_com_chain_t * *chain*)

Receives an application packet through the supplied transmit chain.

Parameters

in,out	*packet*	Pointer to pre-allocated packet struct.
in	*chain*	The chain to use.

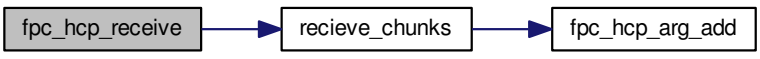
Returns

fpc_com_result_t

Definition at line 117 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, FPC_COM_CHAIN_RX, FPC_COM_RESULT_INVALID_ARGUMENT, fpc_com_chain_private::hcp_packet, fpc_com_chain::initialized, fpc_com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain::private_vars, recieve_chunks(), and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:



13.15.2.8 fpc_com_result_t fpc_hcp_transmit (fpc_hcp_packet_t * *packet*, fpc_com_chain_t * *chain*)

Transmits an application packet through the supplied transmit chain.

Parameters

in	*packet*	Application packet to send.
in	*chain*	The chain to use.

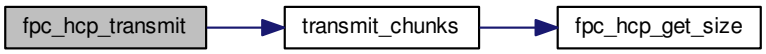
Returns

fpc_com_result_t

Definition at line 89 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, FPC_COM_CHAIN_TX, FPC_↩ COM_RESULT_INVALID_ARGUMENT, fpc_com_chain_private::hcp_packet, fpc_com_chain::initialized, fpc_↩ com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain↩ ::private_vars, transmit_chunks(), and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:

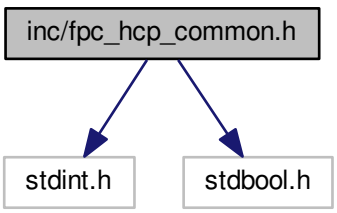


13.16 inc/fpc_hcp_common.h File Reference

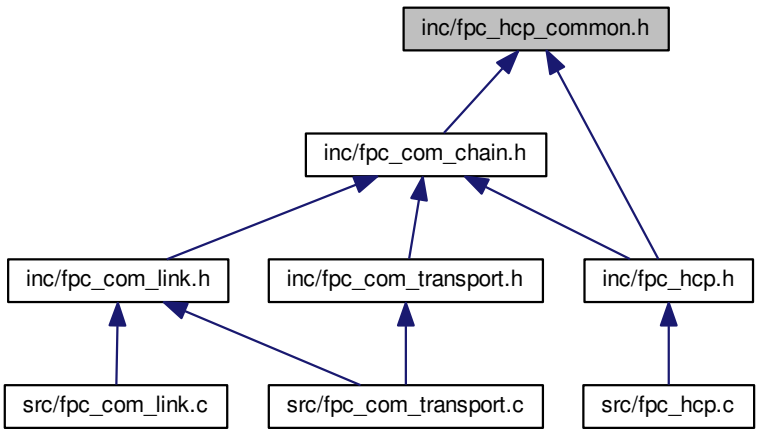
Host Communication Protocol common type definitions.

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

Include dependency graph for fpc_hcp_common.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct fpc_hcp_arg_data

  Command Argument.
- struct fpc_hcp_packet

  Application Command Packet.

Macros

- #define HCP_MIN(x, y) (((x) < (y)) ? (x) : (y))
- #define CMD_APP_BASE_VAL 0xE000
- #define ARG_APP_BASE_VAL 0x7000

Typedefs

- typedef uint16_t fpc_hcp_cmd_t
- typedef uint16_t fpc_hcp_arg_t
- typedef struct fpc_hcp_arg_data fpc_hcp_arg_data_t

  Command Argument.
- typedef struct fpc_hcp_packet fpc_hcp_packet_t

  Application Command Packet.

Enumerations

- enum fpc_hcp_cmd {
  CMD_NONE = 0x0000,
  CMD_CAPTURE = 0x0001,
  CMD_ENROLL = 0x0002,
  CMD_IDENTIFY = 0x0003,
  CMD_MATCH = 0x0004,
  CMD_IMAGE = 0x0005,
  CMD_TEMPLATE = 0x0006,
  CMD_WAIT = 0x0007,
  CMD_SETTINGS = 0x0008,
  CMD_NAVIGATE = 0x1001,
  CMD_SENSOR = 0x1002,
  CMD_DEADPIXELS = 0x1003,
  CMD_CONNECT = 0x2001,
  CMD_RECONNECT = 0x2002,
  CMD_RESET = 0x3002,
  CMD_CANCEL = 0x3003,
  CMD_INFO = 0x3004,
  CMD_STORAGE_TEMPLATE = 0x4002,
  CMD_STORAGE_CALIBRATION = 0x4003,
  CMD_STORAGE_LOG = 0x4004,
  CMD_STORAGE_SETTINGS = 0x4005,
  CMD_TEST = 0x5001,
  CMD_MCU = 0x5002,
  CMD_GPIO = 0x5003,
  CMD_COMMUNICATION = 0x6001,
  CMD_APP_BASE = CMD_APP_BASE_VAL,
  CMD_DIAG = 0xF003,
  CMD_FFFF = 0xFFFF }

- enum fpc_hcp_arg {

ARG_NONE = 0x0000,
ARG_FINGER_DOWN = 0x0001,
ARG_FINGER_UP = 0x0002,
ARG_START = 0x0003,
ARG_ADD = 0x0004,
ARG_FINISH = 0x0005,
ARG_ID = 0x0006,
ARG_ALL = 0x0007,
ARG_EXTRACT = 0x0008,
ARG_MATCH_IMAGE = 0x0009,
ARG_MATCH = 0x000A,
ARG_ACQUIRE = 0x1001,
ARG_RELEASE = 0x1002,
ARG_SET = 0x1003,
ARG_GET = 0x1004,
ARG_UPLOAD = 0x1005,
ARG_DOWNLOAD = 0x1006,
ARG_CREATE = 0x1007,
ARG_SAVE = 0x1008,
ARG_DELETE = 0x1009,
ARG_DATA = 0x100A,
ARG_UPDATE = 0x100B,
ARG_SEQ_NR = 0x100C,
ARG_SEQ_LEN = 0x100D,
ARG_RESULT = 0x2001,
ARG_COUNT = 0x2002,
ARG_SIZE = 0x2003,
ARG_LEVEL = 0x2004,
ARG_FORMAT = 0x2005,
ARG_FLAG = 0x2006,
ARG_PROPERTIES = 0x2007,
ARG_SPEED = 0x2008,
ARG_PROD_TEST = 0x2009,
ARG_SENSOR_TYPE = 0x3001,
ARG_WIDTH = 0x3002,
ARG_HEIGHT = 0x3003,
ARG_RESET = 0x3004,
ARG_DPI = 0x3005,
ARG_MAX_SPI_CLOCK = 0x3006,
ARG_NUM_SUB_AREAS_WIDTH = 0x3007,
ARG_NUM_SUB_AREAS_HEIGHT = 0x3008,
ARG_IRQ_STATUS = 0x3009,
ARG_RESET_HARD = 0x300A,
ARG_IDLE = 0x4001,
ARG_SLEEP = 0x4002,
ARG_DEEP_SLEEP = 0x4003,
ARG_POWER_MODE = 0x4004,
ARG_BUSY_WAIT = 0x4005,
ARG_TIMEOUT = 0x5001,
ARG_DONE = 0x5002,
ARG_BOOT = 0x6001,
ARG_STATUS = 0x6002,
ARG_VERSION = 0x6003,
ARG_UNIQUE_ID = 0x6004,
ARG_APP_BASE = ARG_APP_BASE_VAL,
ARG_NONCE = 0x8001,
ARG_MAC = 0x8002,
ARG_RANDOM = 0x8003,
ARG_CLAIM = 0x8004,
ARG_PUBLIC_KEY = 0x8005,
ARG_CIPHERTEXT = 0x8006,
ARG_MTU = 0x9001,
ARG_STACK = 0xE001,
ARG_FILL = 0xE002,

ARG_FFFF = 0xFFFF }

13.16.1 Detailed Description

Host Communication Protocol common type definitions.

13.16.2 Macro Definition Documentation

13.16.2.1 #define ARG_APP_BASE_VAL 0x7000

Program specific arguments base number

Definition at line 39 of file fpc_hcp_common.h.

13.16.2.2 #define CMD_APP_BASE_VAL 0xE000

Program specific commands base number

Definition at line 36 of file fpc_hcp_common.h.

13.16.2.3 #define HCP_MIN(*x*, *y*) (((x) < (y)) ? (x) : (y))

Returns the smallest of two values.

Definition at line 33 of file fpc_hcp_common.h.

13.16.3 Typedef Documentation

13.16.3.1 typedef struct fpc_hcp_arg_data fpc_hcp_arg_t

Command Argument.

13.16.3.2 typedef uint16_t fpc_hcp_arg_t

HCP Argument type

Definition at line 191 of file fpc_hcp_common.h.

13.16.3.3 typedef uint16_t fpc_hcp_cmd_t

HCP Command type

Definition at line 92 of file fpc_hcp_common.h.

13.16.3.4 typedef struct fpc_hcp_packet fpc_hcp_packet_t

Application Command Packet.

13.16.4 Enumeration Type Documentation

13.16.4.1 enum fpc_hcp_arg

HCP Argument definitions

Enumerator

ARG_NONE

ARG_FINGER_DOWN

ARG_FINGER_UP

ARG_START

ARG_ADD

ARG_FINISH

ARG_ID

ARG_ALL

ARG_EXTRACT

ARG_MATCH_IMAGE

ARG_MATCH

ARG_ACQUIRE

ARG_RELEASE

ARG_SET

ARG_GET

ARG_UPLOAD

ARG_DOWNLOAD

ARG_CREATE

ARG_SAVE

ARG_DELETE

ARG_DATA

ARG_UPDATE

ARG_SEQ_NR

ARG_SEQ_LEN

ARG_RESULT

ARG_COUNT

ARG_SIZE

ARG_LEVEL

ARG_FORMAT

ARG_FLAG

ARG_PROPERTIES

ARG_SPEED

ARG_PROD_TEST

ARG_SENSOR_TYPE
ARG_WIDTH
ARG_HEIGHT
ARG_RESET
ARG_DPI
ARG_MAX_SPI_CLOCK
ARG_NUM_SUB_AREAS_WIDTH
ARG_NUM_SUB_AREAS_HEIGHT
ARG_IRQ_STATUS
ARG_RESET_HARD
ARG_IDLE
ARG_SLEEP
ARG_DEEP_SLEEP
ARG_POWER_MODE
ARG_BUSY_WAIT
ARG_TIMEOUT
ARG_DONE
ARG_BOOT
ARG_STATUS
ARG_VERSION
ARG_UNIQUE_ID
ARG_APP_BASE
ARG_NONCE
ARG_MAC
ARG_RANDOM
ARG_CLAIM
ARG_PUBLIC_KEY
ARG_CIPHERTEXT
ARG_MTU
ARG_STACK
ARG_FILL
ARG_HEAP
ARG_MODE
ARG_DEBUG
ARG_FFFF

Definition at line 95 of file fpc_hcp_common.h.

13.16.4.2 enum fpc_hcp_cmd

HCP Command definitions

Enumerator

CMD_NONE
CMD_CAPTURE

CMD_ENROLL

CMD_IDENTIFY

CMD_MATCH

CMD_IMAGE

CMD_TEMPLATE

CMD_WAIT

CMD_SETTINGS

CMD_NAVIGATE

CMD_SENSOR

CMD_DEADPIXELS

CMD_CONNECT

CMD_RECONNECT

CMD_RESET

CMD_CANCEL

CMD_INFO

CMD_STORAGE_TEMPLATE

CMD_STORAGE_CALIBRATION

CMD_STORAGE_LOG

CMD_STORAGE_SETTINGS

CMD_TEST

CMD_MCU

CMD_GPIO

CMD_COMMUNICATION

CMD_APP_BASE

CMD_DIAG

CMD_FFFF

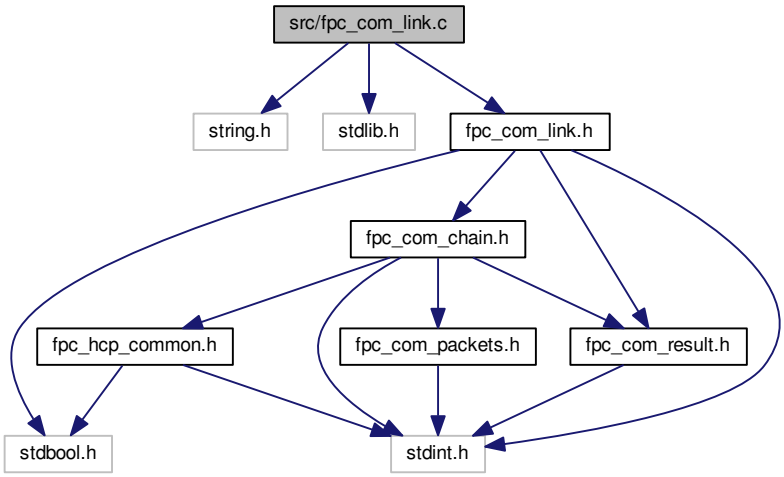
Definition at line 42 of file fpc_hcp_common.h.

13.17 src/fpc_com_link.c File Reference

Communication link layer implementation.

```
#include <string.h>
#include <stdlib.h>
#include "fpc_com_link.h"
```

Include dependency graph for fpc_com_link.c:



Functions

- fpc_com_result_t fpc_com_link_transmit (fpc_com_packet_link_t *packet, fpc_com_chain_t *chain)

  Sends a packet over the physical link in blocking mode.
- fpc_com_result_t fpc_com_link_receive (fpc_com_packet_link_t *packet, fpc_com_chain_t *chain)

  Receives a packet from the physical link.
- uint16_t fpc_com_link_get_overhead (uint16_t *offset)

  Returns the overhead of the layer.

13.17.1 Detailed Description

Communication link layer implementation.

13.17.2 Function Documentation

13.17.2.1 uint16_t fpc_com_link_get_overhead (uint16_t * *offset*)

Returns the overhead of the layer.

Parameters

out	*offset*	The offset to the packet data.

Returns

Overhead size in bytes.

Definition at line 126 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, and fpc_com_packet_link::size.

13.17.2.2 fpc_com_result_t fpc_com_link_receive (fpc_com_packet_link_t * *packet*, fpc_com_chain_t * *chain*)

Receives a packet from the physical link.

Parameters

in,out	*packet*	Packet to populate.
in	*chain*	The communication chain to use.

Returns

fpc_com_result_t

Definition at line 73 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, fpc_com_chain::crc_calc, fpc_com_packet_link::data, FPC_COM_ACK, FPC_COM_CHAIN_RX, FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_IO_ERROR, FPC_COM_RESULT_OK, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain::phy_rx, fpc_com_chain::phy_timeout_rx, fpc_com_chain::phy_timeout_tx, fpc_com_chain::phy_tx, fpc_com_chain::session, and fpc_com_packet_link::size.

13.17.2.3 fpc_com_result_t fpc_com_link_transmit (fpc_com_packet_link_t * *packet*, fpc_com_chain_t * *chain*)

Sends a packet over the physical link in blocking mode.

Parameters

in	*packet*	Packet to transmit.
in	*chain*	The communication chain to use.

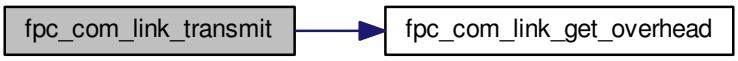
Returns

fpc_com_result_t

Definition at line 27 of file fpc_com_link.c.

References fpc_com_packet_link::channel, fpc_com_packet_link::crc, fpc_com_chain::crc_calc, fpc_com_packet_link::data, FPC_COM_ACK, FPC_COM_CHAIN_TX, fpc_com_link_get_overhead(), FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_IO_ERROR, FPC_COM_RESULT_OK, FPC_COM_RESULT_TIMEOUT, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_rx, fpc_com_chain::phy_timeout_rx, fpc_com_chain::phy_timeout_tx, fpc_com_chain::phy_tx, fpc_com_chain::session, and fpc_com_packet_link::size.

Here is the call graph for this function:

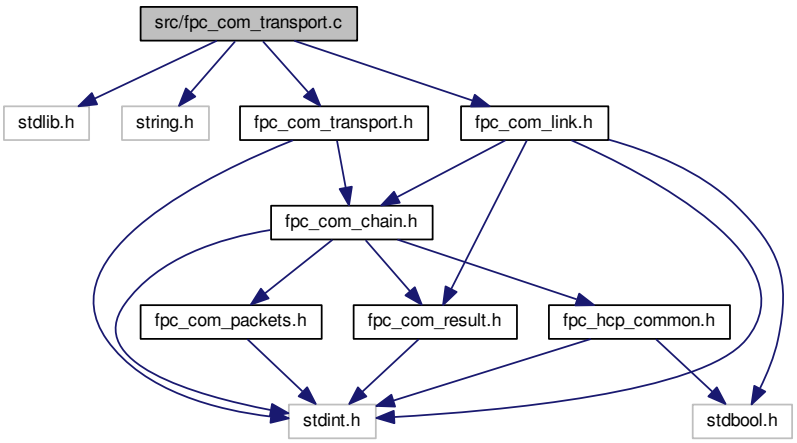


13.18 src/fpc_com_transport.c File Reference

Communication transport layer implementation.

```
#include <stdlib.h>
#include <string.h>
#include "fpc_com_link.h"
#include "fpc_com_transport.h"
```

Include dependency graph for fpc_com_transport.c:



Functions

- fpc_com_result_t fpc_com_transport_transmit (fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
  Transmit a transport layer packet.
- fpc_com_result_t fpc_com_transport_receive (fpc_com_packet_tsp_t *packet, fpc_com_chain_t *chain)
  Receive a transport layer packet.
- uint16_t fpc_com_transport_get_overhead (uint16_t *offset)
  Returns the overhead of the layer.

13.18.1 Detailed Description

Communication transport layer implementation.

13.18.2 Function Documentation

13.18.2.1 uint16_t fpc_com_transport_get_overhead (uint16_t * *offset*)

Returns the overhead of the layer.

Parameters

out	*offset*	The offset to the packet data.

Returns

Overhead size in bytes.

Definition at line 88 of file fpc_com_transport.c.

References fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, and fpc_com_packet_transport::size.

13.18.2.2 fpc_com_result_t fpc_com_transport_receive (fpc_com_packet_tsp_t * *packet*, fpc_com_chain_t * *chain*)

Receive a transport layer packet.

Parameters

in,out	*packet*	The packet to populate.
in	*chain*	The chain to use.

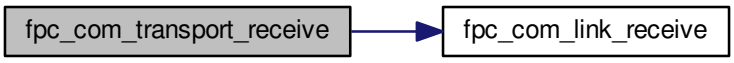
Returns

fpc_com_result_t

Definition at line 60 of file fpc_com_transport.c.

References fpc_com_packet_transport::data, fpc_com_packet_link::data, fpc_com_link_receive(), FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_OK, fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, and fpc_com_packet_transport::size.

Here is the call graph for this function:



13.18.2.3 fpc_com_result_t fpc_com_transport_transmit (fpc_com_packet_tsp_t * *packet*, fpc_com_chain_t * *chain*)

Transmit a transport layer packet.

Parameters

in	*packet*	The packet to transmit.
in	*chain*	The chain to use.

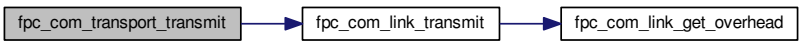
Returns

fpc_com_result_t

Definition at line 28 of file fpc_com_transport.c.

References fpc_com_packet_link::channel, fpc_com_chain::channel, fpc_com_packet_link::data, FPC_COM_CHAIN_TX, fpc_com_link_transmit(), FPC_COM_RESULT_INVALID_ARGUMENT, fpc_com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_packet_transport::seq_len, fpc_com_packet_transport::seq_nr, fpc_com_packet_transport::size, fpc_com_packet_link::size, and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:

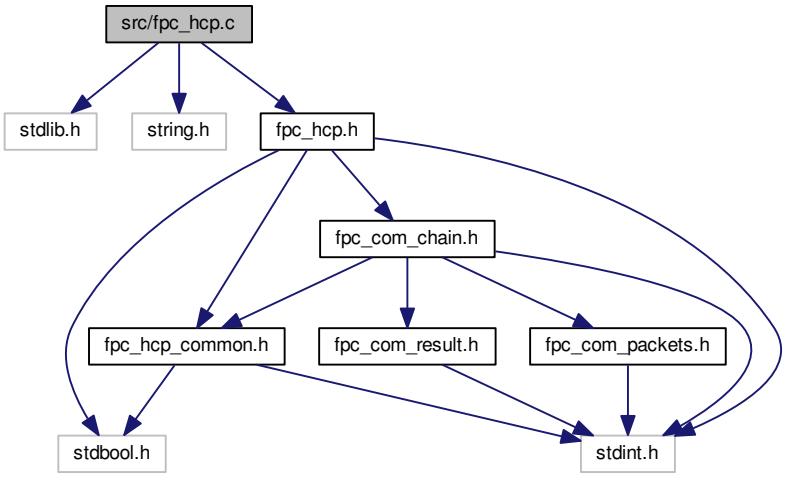


13.19 src/fpc_hcp.c File Reference

Host Communication Protocol implementation.

```
#include <stdlib.h>
#include <string.h>
#include "fpc_hcp.h"
```

Include dependency graph for fpc_hcp.c:



Macros

HCP Packet Member Sizes

Macros for packet member sizes.

- #define PACKET_ID_SIZE sizeof(((fpc_hcp_packet_t*)0)->id)
- #define PACKET_NUM_ARGS_SIZE sizeof(((fpc_hcp_packet_t*)0)->num_args)
- #define PACKET_HEADER_SIZE (PACKET_ID_SIZE + PACKET_NUM_ARGS_SIZE)

HCP Argument Member Sizes

Macros for argument member sizes.

- #define ARGUMENT_ARG_SIZE sizeof(((fpc_hcp_arg_data_t*)0)->arg)
- #define ARGUMENT_SIZE_SIZE sizeof(((fpc_hcp_arg_data_t*)0)->size)
- #define ARGUMENT_HEADER_SIZE (ARGUMENT_ARG_SIZE + ARGUMENT_SIZE_SIZE)

Functions

- static fpc_com_result_t recieve_chunks (fpc_com_chain_t *chain)

  Handle receive chunks.
- static fpc_com_result_t transmit_chunks (fpc_com_chain_t *chain)

  Handle transmit chunks.
- uint16_t fpc_hcp_get_size (fpc_hcp_packet_t *packet, uint16_t *num_args)

  Calculate serialized packet size.
- fpc_com_result_t fpc_hcp_transmit (fpc_hcp_packet_t *packet, fpc_com_chain_t *chain)

  Transmits an application packet through the supplied transmit chain.
- fpc_com_result_t fpc_hcp_receive (fpc_hcp_packet_t *packet, fpc_com_chain_t *chain)

  Receives an application packet through the supplied transmit chain.
- bool fpc_hcp_arg_add (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg, uint16_t size, bool free_data, void *data)

Add argument to packet.

- bool fpc_hcp_arg_check (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg)

  Check if command contains selected argument key.
- fpc_hcp_arg_data_t * fpc_hcp_arg_get (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg)

  Get Argument with specified key.
- bool fpc_hcp_arg_copy_data (fpc_hcp_packet_t *packet, fpc_hcp_arg_t arg, uint16_t data_size, uint8_t *data)

  Copy data from an argument with specified key.
- void fpc_hcp_free (fpc_com_chain_t *chain, fpc_hcp_packet_t *packet)

  Frees the resources held by the packet i.e. the dynamic data held in the arguments.

13.19.1 Detailed Description

Host Communication Protocol implementation.

13.19.2 Macro Definition Documentation

13.19.2.1 #define ARGUMENT_ARG_SIZE sizeof(((fpc_hcp_arg_data_t*)0)->arg)

Definition at line 44 of file fpc_hcp.c.

13.19.2.2 #define ARGUMENT_HEADER_SIZE (ARGUMENT_ARG_SIZE + ARGUMENT_SIZE_SIZE)

Definition at line 46 of file fpc_hcp.c.

13.19.2.3 #define ARGUMENT_SIZE_SIZE sizeof(((fpc_hcp_arg_data_t*)0)->size)

Definition at line 45 of file fpc_hcp.c.

13.19.2.4 #define PACKET_HEADER_SIZE (PACKET_ID_SIZE + PACKET_NUM_ARGS_SIZE)

Definition at line 35 of file fpc_hcp.c.

13.19.2.5 #define PACKET_ID_SIZE sizeof(((fpc_hcp_packet_t*)0)->id)

Definition at line 33 of file fpc_hcp.c.

13.19.2.6 #define PACKET_NUM_ARGS_SIZE sizeof(((fpc_hcp_packet_t*)0)->num_args)

Definition at line 34 of file fpc_hcp.c.

13.19.3 Function Documentation

13.19.3.1 bool fpc_hcp_arg_add (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*, uint16_t *size*, bool *free_data*, void * *data*)

Add argument to packet.

Note

This function does not allocate any memory, it will only set the argument variables.

Parameters

in	*packet*	Packet to add to.
in	*arg*	Argument id.
in	*size*	Size of argument data.
in	*free_data*	Set to true if data should be owned by the argument, false if user still owns data.
in	*data*	Pointer to argument data.

Returns

true = success, false = failure.

Definition at line 145 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, fpc_hcp_packet::arguments, fpc_hcp_arg_data::data, fpc_hcp_arg_data::free_data, fpc_hcp_packet::num_args, and fpc_hcp_arg_data::size.

13.19.3.2 bool fpc_hcp_arg_check (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*)

Check if command contains selected argument key.

Parameters

in	*packet*	The packet to scan.
in	*arg*	Argument to look for.

Returns

true if found, false if not found.

Definition at line 169 of file fpc_hcp.c.

References fpc_hcp_arg_get().

Here is the call graph for this function:

13.19.3.3 bool fpc_hcp_arg_copy_data (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*, uint16_t *data_size*, uint8_t * *data*)

Copy data from an argument with specified key.

Argument data will be copied to specified data buffer. Remaining bytes in data will be cleared if the argument data size is less than data size when the argument contains data.

Parameters

in	*packet*	The packet to operate on.
in	*arg*	The arg to retrieve data from.
in	*data_size*	Number of bytes to copy.
in,out	*data*	Pointer to data buffer.

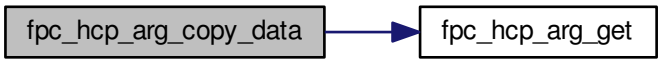
Returns

True if argument found, false if not found.

Definition at line 183 of file fpc_hcp.c.

References fpc_hcp_arg_data::data, fpc_hcp_arg_get(), and fpc_hcp_arg_data::size.

Here is the call graph for this function:



13.19.3.4 fpc_hcp_arg_data_t* fpc_hcp_arg_get (fpc_hcp_packet_t * *packet*, fpc_hcp_arg_t *arg*)

Get Argument with specified key.

Parameters

in	*packet*	The packet to operate on.
in	*arg*	The arg to retrieve.

Returns

Pointer to fpc_hcp_arg_data_t is successful, otherwise NULL.

Definition at line 173 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, fpc_hcp_packet::arguments, and fpc_hcp_packet::num_args.

13.19.3.5 void fpc_hcp_free (fpc_com_chain_t * *chain*, fpc_hcp_packet_t * *packet*)

Frees the resources held by the packet i.e. the dynamic data held in the arguments.

Parameters

in	*chain*	Pointer to the communication chain used to retrieve the packet.
in	*packet*	Pointer to packet.

Definition at line 198 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, fpc_com_chain::argument_free, fpc_hcp_packet::arguments, CMD_NONE, fpc_com_chain::context, fpc_hcp_packet::id, and fpc_hcp_packet::num_args.

13.19.3.6 uint16_t fpc_hcp_get_size (fpc_hcp_packet_t * *packet*, uint16_t * *num_args*)

Calculate serialized packet size.

Parameters

in	*packet*	Packet to calculate.
in,out	*num_args*	Will return number of arguments held by the command can be set to NULL.

Returns

Serialized size.

Definition at line 64 of file fpc_hcp.c.

References fpc_hcp_arg_data::arg, ARG_NONE, ARGUMENT_HEADER_SIZE, fpc_hcp_packet::arguments, fpc_hcp_packet::num_args, PACKET_HEADER_SIZE, and fpc_hcp_arg_data::size.

13.19.3.7 fpc_com_result_t fpc_hcp_receive (fpc_hcp_packet_t * *packet*, fpc_com_chain_t * *chain*)

Receives an application packet through the supplied transmit chain.

Parameters

in,out	*packet*	Pointer to pre-allocated packet struct.
in	*chain*	The chain to use.

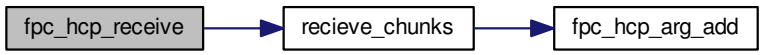
Returns

fpc_com_result_t

Definition at line 117 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, FPC_COM_CHAIN_RX, FPC_↩ COM_RESULT_INVALID_ARGUMENT, fpc_com_chain_private::hcp_packet, fpc_com_chain::initialized, fpc_↩ com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain↩ ::private_vars, recieve_chunks(), and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:



13.19.3.8 fpc_com_result_t fpc_hcp_transmit (fpc_hcp_packet_t * *packet*, fpc_com_chain_t * *chain*)

Transmits an application packet through the supplied transmit chain.

Parameters

in	*packet*	Application packet to send.
in	*chain*	The chain to use.

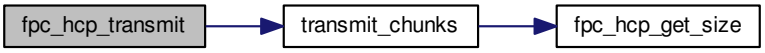
Returns

fpc_com_result_t

Definition at line 89 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, FPC_COM_CHAIN_TX, FPC_↩ COM_RESULT_INVALID_ARGUMENT, fpc_com_chain_private::hcp_packet, fpc_com_chain::initialized, fpc_↩ com_chain::link_overhead_get, fpc_com_chain::phy_mtu_buffer, fpc_com_chain::phy_mtu_size, fpc_com_chain↩ ::private_vars, transmit_chunks(), and fpc_com_chain::tsp_overhead_get.

Here is the call graph for this function:



13.19.3.9 static fpc_com_result_t recieve_chunks (fpc_com_chain_t * *chain*) [static]

Handle receive chunks.

Parameters

chain	Comminucation chain. return fpc_com_result_t

Definition at line 211 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, fpc_com_chain::app_overhead_get, fpc_com_chain::app_packet_size, fpc_com_chain::app_rx, fpc_hcp_arg_data::arg, fpc_com_chain::argument_allocator, ARGUMENT_ARG_SIZE, fpc_com_chain::argument_free, ARGUMENT_SIZE_SIZE, fpc_com_chain::context, fpc_hcp_arg_data::data, FPC_COM_CHAIN_RX, FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_NO_MEMORY, FPC_COM_RESULT_OK, fpc_hcp_arg_add(), fpc_hcp_arg_data::free_data, HCP_MIN, fpc_com_chain_private::hcp_packet, fpc_hcp_packet::id, PACKET_HEADER_SIZE, PACKET_ID_SIZE, PACKET_NUM_ARGS_SIZE, fpc_com_chain::private_vars, and fpc_hcp_arg_data::size.

Here is the call graph for this function:

13.19.3.10 static fpc_com_result_t transmit_chunks (fpc_com_chain_t * *chain*) [static]

Handle transmit chunks.

Parameters

chain	Comminucation chain. return fpc_com_result_t

Definition at line 355 of file fpc_hcp.c.

References fpc_com_chain::app_mtu_buffer, fpc_com_chain::app_mtu_size, fpc_com_chain::app_overhead_get, fpc_com_chain::app_packet_size, fpc_com_chain::app_tx, fpc_hcp_arg_data::arg, ARGUMENT_ARG_SIZE, ARGUMENT_HEADER_SIZE, ARGUMENT_SIZE_SIZE, fpc_hcp_packet::arguments, fpc_hcp_arg_data::data, FPC_COM_CHAIN_TX, FPC_COM_RESULT_INVALID_ARGUMENT, FPC_COM_RESULT_OK, fpc_hcp_get_size(), HCP_MIN, fpc_com_chain_private::hcp_packet, fpc_com_chain_private::hcp_seq_len, fpc_com_chain_private::hcp_seq_nr, fpc_hcp_packet::id, PACKET_HEADER_SIZE, PACKET_ID_SIZE, PACKET_NUM_ARGS_SIZE, fpc_com_chain::phy_mtu_size, fpc_com_chain::private_vars, and fpc_hcp_arg_data::size.

Here is the call graph for this function:

Index