Protocol header

Generated by Doxygen 1.11.0

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 prot_head Struct Reference

Public Attributes

- UINT16 **version1**: 4
  Bitfield for version1 (4 bits)
- UINT16 **version2**: 6
  Bitfield for version2 (6 bits)
- UINT16 **version3**: 6
  Bitfield for version3 (6 bits)
- UINT32 **payload_size**
  The size of the payload in the protocol.
- UCHAR8 **type**
  The type of protocol.
- UINT32 **ID**
  The ID of protocol.

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

- protocol.h

3.2 Protocol_head Struct Reference

This structure of size 11 bytes represent the protocol header which are in packed order.

```
#include <protocol.h>
```

3.2.1 Detailed Description

This structure of size 11 bytes represent the protocol header which are in packed order.

This structure contains bitfields for version information, payload size, type, and ID of the protocol header

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

- protocol.h

Chapter 4

File Documentation

4.1 protocol.c File Reference

This is containing definitions of all functions.

```
#include "protocol.h"
```

Functions

- void main ()
- UINT32 get_packet_id (char *header_pkt)

  this function is used to calculate 4 bytes packet id from bytes 7 to 10 of packet header
- UINT32 get_payload_length (char *header_pkt)

  this function is used to calculate 4 bytes payload length from bytes 2 to 5 of packet header
- void get_protocol_version (char *header_pkt, UINT16 *ver1, UINT16 *ver2, UINT16 *ver3)

  this function is used to calculate all 3 parts of protocol version

Variables

- char **header_pkt** [ ] = "\x0C\x21\x00\x00\x01\x00\x02\x00\x00\x12\x34"

  it is a input header data in Network Order.

4.1.1 Detailed Description

This is containing definitions of all functions.

Author

Shreyas CS

Version

0.1

Date

2024-08-05

4.1.2 Function Documentation

4.1.2.1 get_packet_id()

```
UINT32 get_packet_id (
            char * header_pkt)
```

this function is used to calculate 4 bytes packet id from bytes 7 to 10 of packet header

This function will extract the packet id from the header data.

4.1.2.2 get_payload_length()

```
UINT32 get_payload_length (
            char * header_pkt)
```

this function is used to calculate 4 bytes payload length from bytes 2 to 5 of packet header

This function will extract the payload length from the header data.

4.1.2.3 get_protocol_version()

```
void get_protocol_version (
            char * header_pkt,
            UINT16 * ver1,
            UINT16 * ver2,
            UINT16 * ver3)
```

this function is used to calculate all 3 parts of protocol version

This function will extract the protocol version from the header data.

4.1.2.4 main()

```
void main ()
```

var1 var2 var3 are variables used to store different parts of version

4.2 protocol.h File Reference

It is a headerfile containing declarations of the all functions and stucture.

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

Classes

- struct prot_head

Macros

- #define UINT16 unsigned short int

  Defines a 16-bit unsigned short integer type.
- #define **UINT32** unsigned int
- #define **UCHAR8** unsigned char

Typedefs

- typedef struct prot_head **Protocol_head**

Functions

- UINT32 get_packet_id (char *header_pkt)

  This function will extract the packet id from the header data.
- UINT32 get_payload_length (char *header_pkt)

  This function will extract the payload length from the header data.
- void get_protocol_version (char *header_pkt, UINT16 *ver1, UINT16 *ver2, UINT16 *ver3)

  This function will extract the protocol version from the header data.

4.2.1 Detailed Description

It is a headerfile containing declarations of the all functions and stucture.

Author

Shreyas CS

Version

0.1

Date

2024-08-05

4.2.2 Macro Definition Documentation

4.2.2.1 UINT16

```
#define UINT16 unsigned short int
```

Defines a 16-bit unsigned short integer type.

Defines a 8-bit unsigned char type.

Defines a 32-bit unsigned integer type.

4.2.3 Function Documentation

4.2.3.1 get_packet_id()

```
UINT32 get_packet_id (
            char * header_pkt)
```

This function will extract the packet id from the header data.

Parameters

header_pkt	Pointer to the header data

Returns

Returns 32 bit unsigned integer Packet ID which is present in the header data

This function will extract the packet id from the header data.

4.2.3.2 get_payload_length()

```
UINT32 get_payload_length (
            char * header_pkt)
```

This function will extract the payload length from the header data.

Parameters

header_pkt	Pointer to the header data

Returns

Returns 32 bit unsigned integer payload length which is present in the header data

This function will extract the payload length from the header data.

4.2.3.3 get_protocol_version()

```
void get_protocol_version (
            char * header_pkt,
            UINT16 * ver1,
            UINT16 * ver2,
            UINT16 * ver3)
```

This function will extract the protocol version from the header data.

Parameters

header_pkt	Pointer to the header data.
ver1	Pointer to store version 1 (4 bits).
ver2	Pointer to store version 2 (6 bits).
ver3	Pointer to store version 3 (6 bits).

This function will extract the protocol version from the header data.

4.3 protocol.h

Go to the documentation of this file.

```
00001
00010 #include<stdio.h>
00011 #pragma pack(1)
00016 #define UINT16 unsigned short int
00021 #define UINT32 unsigned int
00026 #define UCHAR8 unsigned char
00027
00035 typedef struct prot_head
00036 {
00040     UINT16 version1 : 4;
00044     UINT16 version2 : 6;
00048     UINT16 version3 : 6;
00052     UINT32 payload_size;
00056     UCHAR8 type;
00060     UINT32 ID;
00061 }Protocol_head;
00062
00069 UINT32 get_packet_id(char *header_pkt);
00076 UINT32 get_payload_length(char *header_pkt);
00086 void get_protocol_version(char *header_pkt,UINT16 *ver1,UINT16 *ver2,UINT16 *ver3);
```