

GCP

0.1

Generated by Doxygen 1.7.6.1

Tue May 15 2012 10:08:45

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	CRC16Params Struct Reference	5
3.1.1	Detailed Description	5
3.2	GCPConn Struct Reference	5
3.2.1	Field Documentation	6
3.2.1.1	send_size	6
4	File Documentation	7
4.1	gcp.c File Reference	7
4.1.1	Detailed Description	8
4.1.2	Function Documentation	8
4.1.2.1	gcp_init	8
4.1.2.2	gcp_rcv_byte	8
4.1.2.3	gcp_send_byte	9
4.1.2.4	rcv_crc1	9
4.1.2.5	rcv_crc2	9
4.1.2.6	rcv_payload	9
4.1.2.7	rcv_preamble1	10
4.1.2.8	rcv_preamble2	10
4.1.2.9	rcv_size1	10

4.1.2.10	recv_size2	10
4.1.2.11	send_crc1	10
4.1.2.12	send_crc2	11
4.1.2.13	send_payload	11
4.1.2.14	send_preamble1	11
4.1.2.15	send_preamble2	11
4.1.2.16	send_size1	12
4.1.2.17	send_size2	12
4.2	gcp.h File Reference	12
4.2.1	Detailed Description	13
4.2.2	Enumeration Type Documentation	13
4.2.2.1	GCPFrameState	13
4.2.3	Function Documentation	13
4.2.3.1	gcp_init	13
4.2.3.2	gcp_recv_byte	13
4.2.3.3	gcp_send_byte	14

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

CRC16Params	
CRC Parameters	5
GCPCConn	5

Chapter 2

File Index

2.1 File List

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

crc16.h	??
gcp.c	7
gcp.h	12

Chapter 3

Data Structure Documentation

3.1 CRC16Params Struct Reference

CRC Parameters.

```
#include <crc16.h>
```

Data Fields

- uint16_t [prefix](#)
Prefix to be added to the data.
- uint16_t [poly](#)
Polynomial to be used.
- unsigned [flip_bits](#): 1
Process the most significant bit of each byte first.
- unsigned [flip_bytes](#): 1
Process bytes at the highest index in the array first.
- unsigned [flip_output](#): 1
Reverse the bits in the output after calculating.

3.1.1 Detailed Description

CRC Parameters.

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

- [crc16.h](#)

3.2 GCPCConn Struct Reference

Data Fields

- `uint8_t * recv_buf`
Receive buffer.
- `uint8_t * send_buf`
Send buffer.
- `uint16_t recv_size`
Receive buffer size.
- `uint16_t send_size`
Send buffer size.
- `uint16_t data_size`
Size of the data in the receive buffer.
- `uint16_t bytes_rcvd`
Number of payload bytes received.
- `uint16_t bytes_sent`
Number of payload bytes sent.
- `uint16_t recv_crc`
The crc checksum of the received data.
- `uint16_t send_crc`
The crc checksum of the data being sent.
- `GCPFrameState recv_state`
The receive state.
- `GCPFrameState send_state`
The send state.
- unsigned `recv_lock`: 1
When true, indicates that the receive buffer is being written to and should not be read from.
- unsigned `send_lock`: 1
When true, indicates that the receive buffer is being read from and should not be written to.

3.2.1 Field Documentation

3.2.1.1 `uint16_t GCPConn::send_size`

Send buffer size.

Note

This is the size of the data in the send buffer, not the size of the buffer itself.

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

- `gcp.h`

Chapter 4

File Documentation

4.1 gcp.c File Reference

```
#include <config.h> #include "gcp.h" #include "crc16.h"
```

Functions

- static void [recv_preamble1](#) (GCPConn *c, uint8_t b)
Reads the first byte of the preamble.
- static void [recv_preamble2](#) (GCPConn *c, uint8_t b)
Reads the second byte of the preamble.
- static void [recv_size1](#) (GCPConn *c, uint8_t b)
Reads the first byte of the data size.
- static void [recv_size2](#) (GCPConn *c, uint8_t b)
Reads the second byte of the data size.
- static void [recv_payload](#) (GCPConn *c, uint8_t b)
Reads the payload data.
- static void [recv_crc1](#) (GCPConn *c, uint8_t b)
Reads the first byte of the checksum.
- static void [recv_crc2](#) (GCPConn *c, uint8_t b)
Reads the second byte of the checksum.
- static uint8_t [send_preamble1](#) (GCPConn *c)
Returns the first byte of the preamble to be sent.
- static uint8_t [send_preamble2](#) (GCPConn *c)
Returns the second byte of the preamble to be sent.
- static uint8_t [send_size1](#) (GCPConn *c)
Returns the first byte of the payload size to be sent.
- static uint8_t [send_size2](#) (GCPConn *c)
Returns the second byte of the payload size to be sent.

- static uint8_t [send_payload](#) (GCPConn *c)
Returns the next byte of the payload to be sent.
- static uint8_t [send_crc1](#) (GCPConn *c)
Returns the first byte of the checksum to be sent.
- static uint8_t [send_crc2](#) (GCPConn *c)
Returns the second byte of the checksum to be sent.
- int [gcp_init](#) (GCPConn *c)
Initializes a GCPConn object.
- int [gcp_rcv_byte](#) (GCPConn *c, uint8_t b)
Processes a byte from the stream.
- uint8_t [gcp_send_byte](#) (GCPConn *c)
Calculates the next byte to be sent to the stream.

Variables

- const CRC16Params [gcp_crc_params](#) = { 0, 0x8005, 0, 0, 1 }
The parameters used by the GCP protocol.

4.1.1 Detailed Description

4.1.2 Function Documentation

4.1.2.1 int [gcp_init](#) (GCPConn * c)

Initializes a [GCPConn](#) object.

Parameters

<i>A</i>	pointer to the object to be initialized.
----------	--

Returns

0 on success; a non-zero value on failure.

4.1.2.2 int [gcp_rcv_byte](#) (GCPConn * c, uint8_t b)

Processes a byte from the stream.

Parameters

<i>c</i>	A pointer to the connection.
<i>b</i>	The byte from the stream to be processed.

Returns

0 on success; a non-zero value on failure.

4.1.2.3 uint8_t gcp_send_byte (GCPCConn * c)

Calculates the next byte to be sent to the stream.

Parameters

<i>c</i>	A pointer to the connection.
----------	------------------------------

Returns

The next byte (or 0 on failure).

4.1.2.4 void recv_crc1 (GCPCConn * c, uint8_t b) [static]

Reads the first byte of the checksum.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.5 void recv_crc2 (GCPCConn * c, uint8_t b) [static]

Reads the second byte of the checksum.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.6 void recv_payload (GCPCConn * c, uint8_t b) [static]

Reads the payload data.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.7 void recv_preamble1 (GCPCConn * *c*, uint8_t *b*) [static]

Reads the first byte of the preamble.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.8 void recv_preamble2 (GCPCConn * *c*, uint8_t *b*) [static]

Reads the second byte of the preamble.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.9 void recv_size1 (GCPCConn * *c*, uint8_t *b*) [static]

Reads the first byte of the data size.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.10 void recv_size2 (GCPCConn * *c*, uint8_t *b*) [static]

Reads the second byte of the data size.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
<i>b</i>	The byte being read.

4.1.2.11 uint8_t send_crc1 (GCPCConn * *c*) [static]

Returns the first byte of the checksum to be sent.

Parameters

<i>c</i>	A pointer to the GCPCConn object.
----------	---

Returns

The first byte of the checksum.

4.1.2.12 `uint8_t send_crc2 (GCPConn * c) [static]`

Returns the second byte of the checksum to be sent.

Parameters

<code>c</code>	A pointer to the GCPConn object.
----------------	--

Returns

The second byte of the checksum.

4.1.2.13 `uint8_t send_payload (GCPConn * c) [static]`

Returns the next byte of the payload to be sent.

Parameters

<code>c</code>	A pointer to the GCPConn object.
----------------	--

Returns

The next byte of the payload.

4.1.2.14 `uint8_t send_preamble1 (GCPConn * c) [static]`

Returns the first byte of the preamble to be sent.

Parameters

<code>c</code>	A pointer to the GCPConn object.
----------------	--

Returns

The first byte of the preamble.

4.1.2.15 `uint8_t send_preamble2 (GCPConn * c) [static]`

Returns the second byte of the preamble to be sent.

Parameters

<code>c</code>	A pointer to the GCPCConn object.
----------------	---

Returns

The second byte of the preamble.

4.1.2.16 `uint8_t send_size1 (GCPCConn * c) [static]`

Returns the first byte of the payload size to be sent.

Parameters

<code>c</code>	A pointer to the GCPCConn object.
----------------	---

Returns

The first byte of the payload size.

4.1.2.17 `uint8_t send_size2 (GCPCConn * c) [static]`

Returns the second byte of the payload size to be sent.

Parameters

<code>c</code>	A pointer to the GCPCConn object.
----------------	---

Returns

The second byte of the payload size.

4.2 gcp.h File Reference

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

Data Structures

- struct [GCPCConn](#)

Enumerations

- enum [GCPFrameState](#) { [gcp_preamble1](#), [gcp_preamble2](#), [gcp_size1](#), [gcp_size2](#), [gcp_payload](#), [gcp_crc1](#), [gcp_crc2](#) }
Communication state.

Functions

- int [gcp_init](#) ([GCPConn](#) *c)
Initializes a [GCPConn](#) object.
- int [gcp_rcv_byte](#) ([GCPConn](#) *c, uint8_t b)
Processes a byte from the stream.
- uint8_t [gcp_send_byte](#) ([GCPConn](#) *c)
Calculates the next byte to be sent to the stream.

4.2.1 Detailed Description

4.2.2 Enumeration Type Documentation

4.2.2.1 enum GCPFrameState

Communication state.

Enumerator:

- [gcp_preamble1](#)*** Reading first byte of the preamble.
- [gcp_preamble2](#)*** Reading second byte of the preamble.
- [gcp_size1](#)*** Reading first byte of the payload size.
- [gcp_size2](#)*** Reading second byte of the payload size.
- [gcp_payload](#)*** Reading payload data.
- [gcp_crc1](#)*** Reading first byte of the checksum.
- [gcp_crc2](#)*** Reading second byte of the checksum.

4.2.3 Function Documentation

4.2.3.1 int [gcp_init](#) ([GCPConn](#) * c)

Initializes a [GCPConn](#) object.

Parameters

<i>A</i>	pointer to the object to be initialized.
-----------------	--

Returns

0 on success; a non-zero value on failure.

4.2.3.2 int [gcp_rcv_byte](#) ([GCPConn](#) * c, uint8_t b)

Processes a byte from the stream.

Parameters

<i>c</i>	A pointer to the connection.
<i>b</i>	The byte from the stream to be processed.

Returns

0 on success; a non-zero value on failure.

4.2.3.3 uint8_t gcp_send_byte (GCPCConn * c)

Calculates the next byte to be sent to the stream.

Parameters

<i>c</i>	A pointer to the connection.
----------	------------------------------

Returns

The next byte (or 0 on failure).