# GCP (Generic Communications Protocol) Library 0.1

Generated by Doxygen 1.7.6.1

Wed May 16 2012 07:50:05

## **Contents**

1	Data	Structi	ure Index				1
	1.1	Data S	tructures				1
2	File	Index					3
	2.1	File Lis	st				3
3	Data	Structi	ure Docun	entation			5
	3.1	CRC16	6Params S	ruct Reference			5
		3.1.1	Detailed	Description			5
	3.2	GCPC	onn Struct	Reference			5
		3.2.1	Detailed	Description			6
		3.2.2	Field Doo	umentation			7
			3.2.2.1	send_size			7
4	File	Docum	entation				9
	4.1	crc16.	File Refe	ence			9
		4.1.1	Detailed	Description			9
		4.1.2	Function	Documentation			9
			4.1.2.1	crc16_check			9
			4.1.2.2	crc16_flush			10
			4.1.2.3	crc16_gen			10
			4.1.2.4	crc16_process_byte			10
			4.1.2.5	flip_16bit			11
			4.1.2.6	flip_8bit			11
	4.2	crc16.h	n File Refe	ence			11
		4.2.1	Detailed	Description			12

ii CONTENTS

	100	Eupotion	Documentation	10
	4.2.2		Documentation	
		4.2.2.1	crc16_check	
		4.2.2.2	crc16_flush	
		4.2.2.3	crc16_gen	
		4.2.2.4	crc16_process_byte	
4.3			nce	
	4.3.1		Description	
	4.3.2		Documentation	
		4.3.2.1	gcp_init	14
		4.3.2.2	gcp_recv_byte	15
		4.3.2.3	gcp_send_byte	15
		4.3.2.4	recv_crc1	15
		4.3.2.5	recv_crc2	15
		4.3.2.6	recv_payload	16
		4.3.2.7	recv_preamble1	16
		4.3.2.8	recv_preamble2	16
		4.3.2.9	recv_size1	16
		4.3.2.10	recv_size2	16
		4.3.2.11	send_crc1	17
		4.3.2.12	send_crc2	17
		4.3.2.13	send_payload	17
		4.3.2.14	send_preamble1	17
		4.3.2.15	send_preamble2	18
		4.3.2.16	send_size1	18
		4.3.2.17	send_size2	18
4.4	gcp.h l	File Referer	nce	19
	4.4.1		Description	19
	4.4.2	Enumerat	ion Type Documentation	19
		4.4.2.1	GCPFrameState	
	4.4.3	Function I	Documentation	
	-	4.4.3.1	gcp init	
		4.4.3.2	gcp_recv_byte	
		4.4.3.3	gcp_send_byte	
			80b_000_01	_0

## **Chapter 1**

## **Data Structure Index**

## 1.1 Data Structures

Horo	aro	tho	data	structures	with	hriof	doccri	otione
пеге	are	uie	uala	Structures	WILLI	briei	uescri	บแบบร

CRC16Params	
CRC Parameters	5
GCPConn	
GCP connection parameters and state	5

## Chapter 2

## File Index

## 2.1 File List

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

crc16.c																				ć
crc16.h																				11
gcp.c																				13
gcp.h																				19

4 File Index

## **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 the last bytes in the stream 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 GCPConn Struct Reference

GCP connection parameters and state.

```
#include <gcp.h>
```

#### **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 received from the stream.

• uint16\_t calc\_crc

The calculated CRC checksum of the data received.

• 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 send buffer is being read from and should not be written to.

#### 3.2.1 Detailed Description

GCP connection parameters and state.

## 3.2.2 Field Documentation

### 3.2.2.1 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 crc16.c File Reference

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

#### **Functions**

- static uint8\_t flip\_8bit (uint8\_t val)
  - Reverses the bits in a uint8\_t value.
- static uint16\_t flip\_16bit (uint16\_t val)
  - Reverses the bits in a uint16\_t value.
- uint16\_t crc16\_gen (const uint8\_t \*data, uint16\_t size, const CRC16Params \*params)
  - Generates a CRC code for a block of data.
- int crc16\_check (const uint8\_t \*data, uint16\_t size, const CRC16Params \*params, uint16\_t crc)
  - Validates a CRC code for a block of data.
- uint16\_t crc16\_process\_byte (uint16\_t prev, uint8\_t byte, uint16\_t poly, int msb\_first)
  - Process a single byte in a CRC16 checksum.
- uint16\_t crc16\_flush (uint16\_t prev, uint16\_t poly, int flip)
  - Processes the remaining 16 bits in a CRC checksum.

#### 4.1.1 Detailed Description

#### 4.1.2 Function Documentation

4.1.2.1 int crc16\_check ( const uint8\_t \* data, uint16\_t size, const CRC16Params \* params, uint16\_t crc )

Validates a CRC code for a block of data.

#### **Parameters**

data	The data being checked.
size	The size (in bytes) of the data being checked.
params	A pointer to the CRC parameters.
crc	The CRC being checked.

#### Returns

0 if the CRC code is valid, a non-zero value otherwise.

4.1.2.2 uint16\_t crc16\_flush ( uint16\_t prev, uint16\_t poly, int flip )

Processes the remaining 16 bits in a CRC checksum.

#### **Parameters**

prev	The calculated value before the flush.
poly	The polynomial to use.
flip	If non-zero, reverses the bit order of the output.

#### Returns

The calculated checksum.

4.1.2.3 uint16\_t crc16\_gen ( const uint8\_t \* data, uint16\_t size, const CRC16Params \* params )

Generates a CRC code for a block of data.

#### **Parameters**

data	The data being used to generate the checksum.
size	The size (in bytes) of the data being CRC'd.
params	A pointer to the CRC parameters.

#### Returns

The generated CRC code or 0 on failure.

4.1.2.4 uint16\_t crc16\_process\_byte ( uint16\_t prev, uint8\_t byte, uint16\_t poly, int msb\_first )

Process a single byte in a CRC16 checksum.

#### **Parameters**

ĺ	prev	The previously calculated value.
ĺ	byte	The byte being processed.
ĺ	msb_first	Set to a non-zero value when the most significant bit of the data is to be
		processed first.
ĺ	poly	The polynomial being used for the CRC.

4.1.2.5 uint16\_t flip\_16bit ( uint16\_t val ) [static]

Reverses the bits in a uint16\_t value.

#### **Parameters**

val The value to be flipped.
------------------------------

#### Returns

The flipped value.

4.1.2.6 uint8\_t flip\_8bit ( uint8\_t val ) [static]

Reverses the bits in a uint8\_t value.

#### **Parameters**

val	The value to be flipped.

#### Returns

The flipped value.

### 4.2 crc16.h File Reference

#include <stdint.h>

#### **Data Structures**

• struct CRC16Params

CRC Parameters.

#### **Functions**

 uint16\_t crc16\_gen (const uint8\_t \*data, uint16\_t size, const CRC16Params \*params) Generates a CRC code for a block of data.

 int crc16\_check (const uint8\_t \*data, uint16\_t size, const CRC16Params \*params, uint16\_t crc)

Validates a CRC code for a block of data.

 uint16\_t crc16\_process\_byte (uint16\_t prev, uint8\_t byte, uint16\_t poly, int msb\_first)

Process a single byte in a CRC16 checksum.

• uint16\_t crc16\_flush (uint16\_t prev, uint16\_t poly, int flip)

Processes the remaining 16 bits in a CRC checksum.

#### 4.2.1 Detailed Description

#### 4.2.2 Function Documentation

4.2.2.1 int crc16\_check ( const uint8\_t \* data, uint16\_t size, const CRC16Params \* params, uint16\_t crc )

Validates a CRC code for a block of data.

#### **Parameters**

data	The data being checked.
size	The size (in bytes) of the data being checked.
params	A pointer to the CRC parameters.
crc	The CRC being checked.

### Returns

0 if the CRC code is valid, a non-zero value otherwise.

4.2.2.2 uint16\_t crc16\_flush ( uint16\_t prev, uint16\_t poly, int flip )

Processes the remaining 16 bits in a CRC checksum.

#### **Parameters**

prev	The calculated value before the flush.
poly	The polynomial to use.
flip	If non-zero, reverses the bit order of the output.

#### Returns

The calculated checksum.

4.2.2.3 uint16\_t crc16\_gen ( const uint8\_t \* data, uint16\_t size, const CRC16Params \* params )

Generates a CRC code for a block of data.

#### **Parameters**

data	The data being used to generate the checksum.
size	The size (in bytes) of the data being CRC'd.
params	A pointer to the CRC parameters.

#### Returns

The generated CRC code or 0 on failure.

4.2.2.4 uint16\_t crc16\_process\_byte ( uint16\_t prev, uint8\_t byte, uint16\_t poly, int msb\_first )

Process a single byte in a CRC16 checksum.

#### **Parameters**

prev	The previously calculated value.
byte	The byte being processed.
msb_first	Set to a non-zero value when the most significant bit of the data is to be
	processed first.
poly	The polynomial being used for the CRC.

## 4.3 gcp.c File Reference

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

#### **Defines**

• #define POLY 0x8005

The polynomial used by the GCP protocol for CRC calculation.

#### **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\_recv\_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.

#### 4.3.1 Detailed Description

#### 4.3.2 Function Documentation

4.3.2.1 int gcp\_init ( GCPConn \* c )

Initializes a GCPConn object.

#### **Parameters**

A pointer to the object to be initialized.

#### Returns

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

4.3.2.2 int gcp\_recv\_byte ( GCPConn \* c, uint8\_t b )

Processes a byte from the stream.

#### **Parameters**

С	A pointer to the connection.
b	The byte from the stream to be processed.

#### Returns

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

4.3.2.3 uint8\_t gcp\_send\_byte ( GCPConn \* c )

Calculates the next byte to be sent to the stream.

#### **Parameters**

С	A pointer to the connection.

#### Returns

The next byte (or 0 on failure).

**4.3.2.4** void recv\_crc1 ( GCPConn \* c, uint8\_t b ) [static]

Reads the first byte of the checksum.

#### **Parameters**

Ì	С	A pointer to the GCPConn object.
	b	The byte being read.

4.3.2.5 void recv\_crc2 ( GCPConn \* c, uint8\_t b ) [static]

Reads the second byte of the checksum.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

**4.3.2.6** void recv\_payload ( GCPConn \* c, uint8\_t b ) [static]

Reads the payload data.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

**4.3.2.7 void recv\_preamble1 ( GCPConn \* c, uint8\_t b )** [static]

Reads the first byte of the preamble.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

**4.3.2.8 void recv\_preamble2 ( GCPConn \* c, uint8\_t b )** [static]

Reads the second byte of the preamble.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

**4.3.2.9** void recv\_size1 ( GCPConn \* c, uint8\_t b ) [static]

Reads the first byte of the data size.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

**4.3.2.10** void recv\_size2 ( GCPConn \* c, uint8\_t b ) [static]

Reads the second byte of the data size.

#### **Parameters**

С	A pointer to the GCPConn object.
b	The byte being read.

4.3.2.11 uint8\_t send\_crc1 ( GCPConn \*c ) [static]

Returns the first byte of the checksum to be sent.

#### **Parameters**

С	A pointer to the GCPConn object.

#### Returns

The first byte of the checksum.

**4.3.2.12** uint8\_t send\_crc2 ( GCPConn \*c ) [static]

Returns the second byte of the checksum to be sent.

#### **Parameters**

С	A pointer to the GCPConn object.

#### Returns

The second byte of the checksum.

**4.3.2.13** uint8\_t send\_payload ( GCPConn \* c ) [static]

Returns the next byte of the payload to be sent.

#### **Parameters**

```
c A pointer to the GCPConn object.
```

#### Returns

The next byte of the payload.

4.3.2.14 uint8\_t send\_preamble1 ( GCPConn \*c ) [static]

Returns the first byte of the preamble to be sent.

File Documentation

#### **Parameters**

c A pointer to the GCPConn object.

#### Returns

The first byte of the preamble.

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

Returns the second byte of the preamble to be sent.

#### **Parameters**

```
c A pointer to the GCPConn object.
```

#### Returns

The second byte of the preamble.

```
4.3.2.16 uint8_t send_size1 ( GCPConn * c ) [static]
```

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

### Parameters

```
c A pointer to the GCPConn object.
```

#### Returns

The first byte of the payload size.

```
4.3.2.17 uint8_t send_size2 ( GCPConn * c ) [static]
```

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

#### **Parameters**

c A pointer to the GCPConn object.

#### Returns

The second byte of the payload size.

### 4.4 gcp.h File Reference

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

#### **Data Structures**

struct GCPConn

GCP connection parameters and state.

#### **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\_recv\_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.

#### 4.4.1 Detailed Description

#### 4.4.2 Enumeration Type Documentation

#### 4.4.2.1 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.

File Documentation

gcp\_crc1 Reading first byte of the checksum.

gcp\_crc2 Reading second byte of the checksum.

#### 4.4.3 Function Documentation

```
4.4.3.1 int gcp_init ( GCPConn * c )
```

Initializes a GCPConn object.

#### **Parameters**

20

Α	pointer to the object to be initialized.

#### Returns

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

```
4.4.3.2 int gcp_recv_byte ( GCPConn * c, uint8_t b )
```

Processes a byte from the stream.

#### **Parameters**

С	A pointer to the connection.
b	The byte from the stream to be processed.

#### Returns

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

### 4.4.3.3 uint8\_t gcp\_send\_byte ( GCPConn \* c )

Calculates the next byte to be sent to the stream.

#### **Parameters**

С	A pointer to the connection.

#### Returns

The next byte (or 0 on failure).