

quectel_bc66_drv

Generated by Doxygen 1.9.1

1 Data Structure Index	1
1 Data Structure Index	1
1.1 Data Structures	1
2 File Index	1
2.1 File List	1
3 Data Structure Documentation	2
3.1 bc66_at_cmd_t Struct Reference	2
3.1.1 Detailed Description	2
3.1.2 Field Documentation	2
3.2 bc66_obj_t Struct Reference	3
3.2.1 Field Documentation	3
4 File Documentation	5
4.1 /Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.c File Reference	5
4.1.1 Macro Definition Documentation	6
4.1.2 Enumeration Type Documentation	6
4.1.3 Function Documentation	7
4.1.4 Variable Documentation	8
4.2 /Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.h File Reference	9
4.2.1 Detailed Description	10
4.2.2 Enumeration Type Documentation	11
4.2.3 Function Documentation	12
Index	15

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

bc66_at_cmd_t	
BC66 Command struct	2
bc66_obj_t	3

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

/Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.c	5
--	----------

3 Data Structure Documentation

3.1 bc66_at_cmd_t Struct Reference

BC66 Command struct.

Data Fields

- const char * [cmd](#)
at command sentence
- [cmd_flg_t](#) [cmd_flags](#)
flags for command implementation (see
- char * [cmd_rsp](#)
expected command response
- uint32_t [rsp_timeout](#)
response timeout [ms]

3.1.1 Detailed Description

BC66 Command struct.

3.1.2 Field Documentation

3.1.2.1 **cmd** const char* cmd

at command sentence

3.1.2.2 **cmd_flags** [cmd_flg_t](#) cmd_flags

flags for command implementation (see
[flags](#) [enum](#))

3.1.2.3 **cmd_rsp** char* cmd_rsp

expected command response

3.1.2.4 `rsp_timeout` `uint32_t` `rsp_timeout`

response timeout [ms]

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

- `/Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.c`

3.2 `bc66_obj_t` Struct Reference

```
#include <bc66_drv.h>
```

Data Fields

- `const void(* func_init_ptr)()`
uart initialize function pointer
- `const void(* func_delay)(uint32_t t)`
delay function pointer
- `int(* func_w_bytes_ptr)(uint8_t *txc, uint8_t size)`
write bytes function pointer
- `int(* func_r_bytes_ptr)(uint8_t *rxc)`
read one-byte function pointer
- `struct {`
 - `void(* MDM_PSM_EINT_N)(size_t pin_value)`
delay function pointer
 - `void(* MDM_PWRKEY_N)(size_t pin_value)`
modem power key function pointer
 - `void(* MDM_RESET_N)(size_t pin_value)`
modem reset function pointer
 - `void(* MDM_RI)()`
modem ring interrupt function pointer
- `} control_lines`

3.2.1 **Field Documentation****3.2.1.1** `struct { ... } control_lines`**3.2.1.2** `func_delay` `const void(* func_delay) (uint32_t t)`

delay function pointer

3.2.1.3 func_init_ptr `const void(* func_init_ptr) ()`

uart initialize function pointer

3.2.1.4 func_r_bytes_ptr `int(* func_r_bytes_ptr) (uint8_t *rxc)`

read one-byte function pointer

3.2.1.5 func_w_bytes_ptr `int(* func_w_bytes_ptr) (uint8_t *txc, uint8_t size)`

write bytes function pointer

3.2.1.6 MDM_PSM_EINT_N `void(* MDM_PSM_EINT_N) (size_t pin_value)`

delay function pointer

3.2.1.7 MDM_PWRKEY_N `void(* MDM_PWRKEY_N) (size_t pin_value)`

modem power key function pointer

3.2.1.8 MDM_RESET_N `void(* MDM_RESET_N) (size_t pin_value)`

modem reset function pointer

3.2.1.9 MDM_RI `void(* MDM_RI) ()`

modem ring interrupt function pointer

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

- /Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.h

4 File Documentation

4.1 /Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdarg.h>
#include "bc66_drv.h"
```

Data Structures

- struct [bc66_at_cmd_t](#)
BC66 Command struct.

Macros

- #define [CMD_END_LINE](#) "\r\n"
End of line command chars.
- #define [RSP_OK](#) "\r\nOK\r\n"
Ok response.
- #define [RSP_ERROR](#) "\r\nERROR\r\n"
Error response.
- #define [RSP_END_OF_LINE](#) "\r\n"
End of line response chars.
- #define [RSP_TIMEOUT](#) "BC66_TIMEOUT\r\n"
Answer when a timeout is occurred.
- #define [RSP_NO_CMD_IMPLEMENTED](#) "BC66_NO_CMD\r\n"
The command is not implemented.
- #define [MAX_RSP_SIZE](#) 64
Max AT response size.

Enumerations

- enum [cmd_flg_t](#) { [TEST](#) = 0x1 , [READ](#) = 0x2 , [WRITE](#) = 0x4 , [EXE](#) = 0x8 }
Command possibilities indicator flags.

Functions

- void [bc66_init](#) ([bc66_obj_t](#) *bc66_obj)
Function to initialize bc66 object.
- void [bc66_deinit](#) ([bc66_obj_t](#) *bc66_obj)
Function to initialize bc66 object.
- char * [bc66_send_at_command](#) ([bc66_cmd_type_t](#) cmd_type, const [bc66_cmd_list_t](#) cmd_lst, const char *exp_rsp, const char *arg_fmt,...)
Function to send at command sentence to bc66 module through an external function communication.
- char * [bc66_get_at_response](#) (char *rsp)
Function to get any response stored in the RX buffer.
- void [bc66_reset](#) (void)
Reset the module when PIN is low.
- void [bc66_power_on](#) ()
Pull down PWRKEY to turn on the module.
- void [bc66_power_off](#) ()
Pull up PWRKEY to turn off the module.

Variables

- const `bc66_at_cmd_t bc66_cmds_list []`

Define AT commands list: order must be equal to commands definition enum `bc66_cmd_list_t`.

4.1.1 Macro Definition Documentation

4.1.1.1 `CMD_END_LINE` `#define CMD_END_LINE "\r\n"`

End of line command chars.

4.1.1.2 `MAX_RSP_SIZE` `#define MAX_RSP_SIZE 64`

Max AT response size.

4.1.1.3 `RSP_END_OF_LINE` `#define RSP_END_OF_LINE "\r\n"`

End of line response chars.

4.1.1.4 `RSP_ERROR` `#define RSP_ERROR "\r\nERROR\r\n"`

Error response.

4.1.1.5 `RSP_NO_CMD_IMPEMENTED` `#define RSP_NO_CMD_IMPEMENTED "BC66_NO_CMD\r\n"`

The command is not implemented.

4.1.1.6 `RSP_OK` `#define RSP_OK "\r\nOK\r\n"`

Ok response.

4.1.1.7 `RSP_TIMEOUT` `#define RSP_TIMEOUT "BC66_TIMEOUT\r\n"`

Answer when a timeout is occurred.

4.1.2 Enumeration Type Documentation

4.1.2.1 `cmd_flg_t` `enum cmd_flg_t`

Command possibilities indicator flags.

Enumerator

TEST	Command has test possibility.
READ	Command has read possibility.
WRITE	Command has write possibility.
EXE	Command has execute possibility.

4.1.3 Function Documentation

4.1.3.1 bc66_deinit() `void bc66_deinit (`
`bc66_obj_t * bc66_obj)`

Function to initialize bc66 object.

Parameters

<i>bc66_obj</i>	
-----------------	--

4.1.3.2 bc66_get_at_response() `char* bc66_get_at_response (`
`char * rsp)`

Function to get any response stored in the RX buffer.

Parameters

<i>rsp</i>	: response to get
------------	-------------------

Returns

Response if found, NULL otherwise

4.1.3.3 bc66_init() `void bc66_init (`
`bc66_obj_t * bc66_obj)`

Function to initialize bc66 object.

Parameters

<i>bc66_obj</i>	
-----------------	--

4.1.3.4 **bc66_power_off()** `void bc66_power_off ()`

Pull up PWRKEY to turn off the module.

4.1.3.5 **bc66_power_on()** `void bc66_power_on ()`

Pull down PWRKEY to turn on the module.

4.1.3.6 **bc66_reset()** `void bc66_reset (` `void)`

Reset the module when PIN is low.

4.1.3.7 **bc66_send_at_command()** `char* bc66_send_at_command (` `bc66_cmd_type_t cmd_type,` `const bc66_cmd_list_t cmd_lst,` `const char * exp_rsp,` `const char * arg_fmt,` `...)`

Function to send at command sentence to bc66 module through an external function communication.

Parameters

<i>cmd_type</i>	: BC66_CMD_TEST, BC66_CMD_READ, BC66_CMD_WRITE or BC66_CMD_EXE type.
<i>cmd_lst</i>	: command to send (see command list).
<i>rsp</i>	: pointer to expected response text.
<i>arg_fmt</i>	: arguments format (like printf function) and must be send all arguments too.

Returns

- Command answer text
- OK
- ERROR
- TIMEOUT

4.1.4 Variable Documentation

4.1.4.1 bc66_cmds_list `const bc66_at_cmd_t bc66_cmds_list[]`

Define AT commands list: order must be equal to commands definition enum `bc66_cmd_list_t`.

4.2 /Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.h File Reference

MIT License.

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

Data Structures

- struct `bc66_obj_t`

Enumerations

- enum `bc66_cmd_type_t` { `BC66_CMD_TEST` , `BC66_CMD_READ` , `BC66_CMD_WRITE` , `BC66_CMD_EXE` }
- AT command possibility. Erch command can test and/or read and/or write and/or execute. Use with `bc66_send_at_command()` function.
- enum `bc66_cmd_list_t` {
`bc66_cmd_list_AT` , `bc66_cmd_list_ATI` , `bc66_cmd_list_ATE` , `bc66_cmd_list_CEREG` ,
`bc66_cmd_list_CESQ` , `bc66_cmd_list_CGATT` , `bc66_cmd_list_CGPADDR` , `bc66_cmd_list_QCGDEFCONT`
 ,
`bc66_cmd_list_CIMI` , `bc66_cmd_list_CPIN` , `bc66_cmd_list_CPSMS` , `bc66_cmd_list_QNBIOTEVENT` ,
`bc66_cmd_list_QMTCFG` , `bc66_cmd_list_QMTOPE` , `bc66_cmd_list_QMTCLOSE` , `bc66_cmd_list_QMTCONN`
 ,
`bc66_cmd_list_QMTDISC` , `bc66_cmd_list_QMTSUB` , `bc66_cmd_list_QMTUNS` , `bc66_cmd_list_QMTPUB`
 ,
`bc66_cmd_list_size` }
 This is the commands implemented list.

Functions

- void `bc66_init` (`bc66_obj_t` *`bc66_obj`)
 Function to initialize bc66 object.
- char * `bc66_get_at_response` (char *`rsp`)
 Function to get any response stored in the RX buffer.
- char * `bc66_send_at_command` (`bc66_cmd_type_t` `cmd_type`, const `bc66_cmd_list_t` `cmd_lst`, const char *`exp_rsp`, const char *`arg_fmt`,...)
 Function to send at command sentence to bc66 module through an external function communication.
- void `bc66_reset` (void)
 Reset the module when PIN is low.
- void `bc66_power_on` ()
 Pull down PWRKEY to turn on the module.
- void `bc66_power_off` ()
 Pull up PWRKEY to turn off the module.

4.2.1 Detailed Description

MIT License.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright

Juan Cruz Becerra

BC66 NB-IoT modem driver. (<https://www.quectel.com/product/bc66.htm>)

AT Command Syntax The `AT` or `AT+` prefix must be set at the beginning of each command line. Entering `<CR>` will terminate a command line. Commands are usually followed by a response that includes `<CR><LF><response><CR><LF>`. Throughout this document, only the responses are presented, `<CR><LF>` are omitted intentionally.

Types of AT Commands and Responses

- Test Command `AT+<x>=?`
- Read Command `AT+<x>?`
- Write Command `AT+<x>=<value>`
- Execution Command `AT+<x>`

Date

15 de marzo de 2021, 10:06

Author

Eng. Juan Cruz Becerra

Version

1.0.0

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright

Juan Cruz Becerra

Date

15 de marzo de 2021, 10:06

Author

Eng. Juan Cruz Becerra

Version

1.0.0

4.2.2 Enumeration Type Documentation**4.2.2.1 bc66_cmd_list_t** enum bc66_cmd_list_t

This is the commands implemented list.

Enumerator

bc66_cmd_list_AT	AT command. Use to sync baud rate.
bc66_cmd_list_ATI	Display Product Identification Information.
bc66_cmd_list_ATE	Set Command Echo Mode.
bc66_cmd_list_CEREG	EPS Network Registration Status.
bc66_cmd_list_CESQ	Extended Signal Quality.
bc66_cmd_list_CGATT	PS Attachment or Detachment.
bc66_cmd_list_CGPADDR	Show PDP Addresses.
bc66_cmd_list_QCGDEFCONT	Set Default PSD Connection Settings.
bc66_cmd_list_CIMI	Request International Mobile Subscriber Identity.
bc66_cmd_list_CPIN	Enter PIN.
bc66_cmd_list_CPSMS	Power Saving Mode Setting.
bc66_cmd_list_QNBIOTEVENT	Enable/Disable NB-IoT Related Event Report.
bc66_cmd_list_QMTCFG	Configure Optional Parameters of MQTT.
bc66_cmd_list_QMTOOPEN	Open a Network for MQTT Client.
bc66_cmd_list_QMTCLOSE	Close a Network for MQTT Client.
bc66_cmd_list_QMTCONN	Connect a Client to MQTT Server.
bc66_cmd_list_QMTDISC	Disconnect a Client from MQTT Server.
bc66_cmd_list_QMTSUB	Subscribe to Topics.
bc66_cmd_list_QMTUNS	Unsubscribe from Topics.
bc66_cmd_list_QMTPUB	Publish Messages.
bc66_cmd_list_size	Is not a command. Only to know commands quantity.

4.2.2.2 bc66_cmd_type_t enum bc66_cmd_type_t

AT command possibility. Erch command can test and/or read and/or write and/or execute. Use with `bc66_send_at_command()` function.

Enumerator

BC66_CMD_TEST	Send AT TEST command.
---------------	-----------------------

Enumerator

BC66_CMD_READ	Send AT READ command.
BC66_CMD_WRITE	Send AT WRITE command.
BC66_CMD_EXE	Send AT TEST command.

4.2.3 Function Documentation

4.2.3.1 bc66_get_at_response() `char* bc66_get_at_response (`
`char * rsp)`

Function to get any response stored in the RX buffer.

Parameters

<i>rsp</i>	: response to get
------------	-------------------

Returns

Response if found, NULL otherwise

4.2.3.2 bc66_init() `void bc66_init (`
`bc66_obj_t * bc66_obj)`

Function to initialize bc66 object.

Parameters

<i>bc66_obj</i>	
-----------------	--

4.2.3.3 bc66_power_off() `void bc66_power_off ()`
Pull up PWRKEY to turn off the module.

4.2.3.4 bc66_power_on() `void bc66_power_on ()`
Pull down PWRKEY to turn on the module.

4.2.3.5 bc66_reset() `void bc66_reset (`
`void)`
Reset the module when PIN is low.

4.2.3.6 bc66_send_at_command() `char* bc66_send_at_command (`
`bc66_cmd_type_t cmd_type,`
`const bc66_cmd_list_t cmd_lst,`
`const char * exp_rsp,`
`const char * arg_fmt,`
`...)`

Function to send at command sentence to bc66 module through an external function communication.

Parameters

<i>cmd_type</i>	: BC66_CMD_TEST, BC66_CMD_READ, BC66_CMD_WRITE or BC66_CMD_EXE type.
<i>cmd_lst</i>	: command to send (see command list).
<i>rsp</i>	: pointer to expected response text.
<i>arg_fmt</i>	: arguments format (like printf function) and must be sended all arguments too.

Returns

- Command aswer text
- OK
- ERROR
- TIMEOUT

Index

[/Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.c](#), [bc66_drv.h](#), [11](#)
[5](#)
[BC66_CMD_READ](#)
[/Users/jcbecerra/dev/fw/iot/quectel_bc66_driver/src/bc66_drv.h](#), [bc66_drv.h](#), [12](#)
[9](#)
[BC66_CMD_TEST](#)
[bc66_drv.h](#), [11](#)
[bc66_cmd_type_t](#)
[bc66_drv.h](#), [11](#)
[BC66_CMD_WRITE](#)
[bc66_drv.h](#), [12](#)
[bc66_cmds_list](#)
[bc66_drv.c](#), [8](#)
[bc66_deinit](#)
[bc66_drv.c](#), [7](#)
[bc66_drv.c](#)
[bc66_cmds_list](#), [8](#)
[bc66_deinit](#), [7](#)
[bc66_get_at_response](#), [7](#)
[bc66_init](#), [7](#)
[bc66_power_off](#), [8](#)
[bc66_power_on](#), [8](#)
[bc66_reset](#), [8](#)
[bc66_send_at_command](#), [8](#)
[CMD_END_LINE](#), [6](#)
[cmd_flg_t](#), [6](#)
[EXE](#), [7](#)
[MAX_RSP_SIZE](#), [6](#)
[READ](#), [7](#)
[RSP_END_OF_LINE](#), [6](#)
[RSP_ERROR](#), [6](#)
[RSP_NO_CMD_IMPEMENTED](#), [6](#)
[RSP_OK](#), [6](#)
[RSP_TIMEOUT](#), [6](#)
[TEST](#), [7](#)
[WRITE](#), [7](#)
[bc66_drv.h](#)
[BC66_CMD_EXE](#), [12](#)
[bc66_cmd_list_AT](#), [11](#)
[bc66_cmd_list_ATE](#), [11](#)
[bc66_cmd_list_ATI](#), [11](#)
[bc66_cmd_list_CEREG](#), [11](#)
[bc66_cmd_list_CESQ](#), [11](#)
[bc66_cmd_list_CGATT](#), [11](#)
[bc66_cmd_list_CGPADDR](#), [11](#)
[bc66_cmd_list_CIMI](#), [11](#)
[bc66_cmd_list_CPIN](#), [11](#)
[bc66_cmd_list_CPSMS](#), [11](#)
[bc66_cmd_list_QCGDEFCONT](#), [11](#)
[bc66_cmd_list_QMTCFG](#), [11](#)
[bc66_cmd_list_QMTCLOSE](#), [11](#)
[bc66_cmd_list_QMTCONN](#), [11](#)
[bc66_cmd_list_QMTDISC](#), [11](#)
[bc66_cmd_list_QMTOPEN](#), [11](#)
[bc66_cmd_list_QMTPUB](#), [11](#)
[bc66_cmd_list_QMTSUB](#), [11](#)
[bc66_cmd_list_QMTUNS](#), [11](#)
[bc66_cmd_list_QNBIOTEVENT](#), [11](#)
[bc66_cmd_list_size](#), [11](#)
[bc66_cmd_list_t](#)

- bc66_cmd_list_QNBIOTEVENT, 11
- bc66_cmd_list_size, 11
- bc66_cmd_list_t, 11
- BC66_CMD_READ, 12
- BC66_CMD_TEST, 11
- bc66_cmd_type_t, 11
- BC66_CMD_WRITE, 12
- bc66_get_at_response, 12
- bc66_init, 12
- bc66_power_off, 12
- bc66_power_on, 12
- bc66_reset, 12
- bc66_send_at_command, 12
- bc66_get_at_response
 - bc66_drv.c, 7
 - bc66_drv.h, 12
- bc66_init
 - bc66_drv.c, 7
 - bc66_drv.h, 12
- bc66_obj_t, 3
 - control_lines, 3
 - func_delay, 3
 - func_init_ptr, 3
 - func_r_bytes_ptr, 4
 - func_w_bytes_ptr, 4
 - MDM_PSM_EINT_N, 4
 - MDM_PWRKEY_N, 4
 - MDM_RESET_N, 4
 - MDM_RI, 4
- bc66_power_off
 - bc66_drv.c, 8
 - bc66_drv.h, 12
- bc66_power_on
 - bc66_drv.c, 8
 - bc66_drv.h, 12
- bc66_reset
 - bc66_drv.c, 8
 - bc66_drv.h, 12
- bc66_send_at_command
 - bc66_drv.c, 8
 - bc66_drv.h, 12
- cmd
 - bc66_at_cmd_t, 2
- CMD_END_LINE
 - bc66_drv.c, 6
- cmd_flags
 - bc66_at_cmd_t, 2
- cmd_flg_t
 - bc66_drv.c, 6
- cmd_rsp
 - bc66_at_cmd_t, 2
- control_lines
 - bc66_obj_t, 3
- EXE
 - bc66_drv.c, 7
- func_delay
 - bc66_obj_t, 3
- func_init_ptr
 - bc66_obj_t, 3
- func_r_bytes_ptr
 - bc66_obj_t, 4
- func_w_bytes_ptr
 - bc66_obj_t, 4
- MAX_RSP_SIZE
 - bc66_drv.c, 6
- MDM_PSM_EINT_N
 - bc66_obj_t, 4
- MDM_PWRKEY_N
 - bc66_obj_t, 4
- MDM_RESET_N
 - bc66_obj_t, 4
- MDM_RI
 - bc66_obj_t, 4
- READ
 - bc66_drv.c, 7
- RSP_END_OF_LINE
 - bc66_drv.c, 6
- RSP_ERROR
 - bc66_drv.c, 6
- RSP_NO_CMD_IMPEMENTED
 - bc66_drv.c, 6
- RSP_OK
 - bc66_drv.c, 6
- RSP_TIMEOUT
 - bc66_drv.c, 6
- rsp_timeout
 - bc66_at_cmd_t, 2
- TEST
 - bc66_drv.c, 7
- WRITE
 - bc66_drv.c, 7