

## My Project

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	Command Struct Reference . . . . .	5
3.1.1	Member Data Documentation . . . . .	5
3.1.1.1	action . . . . .	5
3.1.1.2	argument . . . . .	5
3.1.1.3	key . . . . .	5
3.1.1.4	text . . . . .	6
3.2	graph_view Struct Reference . . . . .	6
3.2.1	Member Data Documentation . . . . .	6
3.2.1.1	horizontal_tick_marks . . . . .	6
3.2.1.2	vertical_tick_marks . . . . .	7
3.2.1.3	view . . . . .	7
3.3	Hashmap Struct Reference . . . . .	7
3.3.1	Member Data Documentation . . . . .	8
3.3.1.1	add . . . . .	8
3.3.1.2	elements . . . . .	8
3.3.1.3	exists . . . . .	8
3.3.1.4	get . . . . .	8

3.3.1.5	<a href="#">remove</a>	8
3.3.1.6	<a href="#">size</a>	8
3.3.1.7	<a href="#">table</a>	8
3.3.1.8	<a href="#">update</a>	9
3.4	<a href="#">HashMapElement Struct Reference</a>	9
3.4.1	<a href="#">Member Data Documentation</a>	9
3.4.1.1	<a href="#">datum</a>	9
3.4.1.2	<a href="#">key</a>	9
3.5	<a href="#">I2C Struct Reference</a>	9
3.5.1	<a href="#">Member Data Documentation</a>	10
3.5.1.1	<a href="#">accelerometers</a>	10
3.5.1.2	<a href="#">gyros</a>	10
3.5.1.3	<a href="#">i2c_address</a>	10
3.5.1.4	<a href="#">i2c_slave_address</a>	10
3.5.1.5	<a href="#">magnetometers</a>	10
3.5.1.6	<a href="#">registers</a>	11
3.5.1.7	<a href="#">temperature</a>	11
3.6	<a href="#">List Struct Reference</a>	11
3.6.1	<a href="#">Member Data Documentation</a>	11
3.6.1.1	<a href="#">doublely_linked</a>	12
3.6.1.2	<a href="#">elements</a>	12
3.6.1.3	<a href="#">elements_limit</a>	12
3.6.1.4	<a href="#">head</a>	12
3.7	<a href="#">Logger Struct Reference</a>	12
3.7.1	<a href="#">Member Data Documentation</a>	13
3.7.1.1	<a href="#">close</a>	13
3.7.1.2	<a href="#">destroy</a>	13
3.7.1.3	<a href="#">file</a>	13
3.7.1.4	<a href="#">filename</a>	13
3.7.1.5	<a href="#">open</a>	13

3.7.1.6	self	14
3.7.1.7	termination_signal	14
3.7.1.8	thread	14
3.7.1.9	values_read	14
3.8	module Struct Reference	14
3.8.1	Member Data Documentation	15
3.8.1.1	i2c	15
3.8.1.2	identifier	15
3.8.1.3	initialized	15
3.8.1.4	loaded	15
3.8.1.5	n_pins	15
3.8.1.6	pins	16
3.8.1.7	uart	16
3.9	Node Struct Reference	16
3.9.1	Member Data Documentation	16
3.9.1.1	"@3	17
3.9.1.2	"@5	17
3.9.1.3	child	17
3.9.1.4	left	17
3.9.1.5	next	17
3.9.1.6	prev	17
3.9.1.7	right	17
3.9.1.8	value	18
3.10	pin Struct Reference	18
3.10.1	Member Data Documentation	18
3.10.1.1	"@1	18
3.10.1.2	duty_cycle	18
3.10.1.3	logical	18
3.10.1.4	physical	19
3.10.1.5	state	19

3.10.1.6	voltage	19
3.11	Plot Struct Reference	19
3.11.1	Member Data Documentation	20
3.11.1.1	has_data	20
3.11.1.2	lists	20
3.11.1.3	max_value	20
3.11.1.4	min_value	20
3.11.1.5	name	20
3.11.1.6	number_of_lists	21
3.12	print_view Struct Reference	21
3.12.1	Member Data Documentation	21
3.12.1.1	colors	22
3.12.1.2	current_view_line	22
3.12.1.3	lines	22
3.12.1.4	number_lines_printed	22
3.12.1.5	number_of_lines	22
3.12.1.6	view	22
3.13	Selector Struct Reference	23
3.13.1	Member Data Documentation	23
3.13.1.1	entries	23
3.13.1.2	parent	23
3.13.1.3	title	24
3.14	setup_view Struct Reference	24
3.14.1	Member Data Documentation	24
3.14.1.1	view	24
3.15	View Struct Reference	25
3.15.1	Member Data Documentation	25
3.15.1.1	inner_height	25
3.15.1.2	inner_width	25
3.15.1.3	outer_height	25
3.15.1.4	outer_width	25
3.15.1.5	window	25

<b>4 File Documentation</b>	<b>27</b>
4.1 colors.h File Reference	27
4.1.1 Macro Definition Documentation	27
4.1.1.1 BLUE	28
4.1.1.2 CONSOLE_BLUE	28
4.1.1.3 CONSOLE_CYAN	28
4.1.1.4 CONSOLE_GRAY	28
4.1.1.5 CONSOLE_GREEN	28
4.1.1.6 CONSOLE_MAGENTA	28
4.1.1.7 CONSOLE_RED	28
4.1.1.8 CONSOLE_RESET	28
4.1.1.9 CONSOLE_YELLOW	29
4.1.1.10 DIM	29
4.1.1.11 GREEN	29
4.1.1.12 GREY	29
4.1.1.13 PURPLE	29
4.1.1.14 RED	29
4.1.1.15 RESET	29
4.1.1.16 UNDIM	29
4.1.1.17 YELLOW	30
4.2 error.c File Reference	30
4.2.1 Function Documentation	30
4.2.1.1 exit_printing()	30
4.3 error.h File Reference	31
4.3.1 Macro Definition Documentation	31
4.3.1.1 ERROR_LIBRARY_FAILURE	31
4.3.1.2 ERROR_OS_FAILURE	31
4.3.1.3 ERROR_PROGRAMMER	31
4.3.2 Function Documentation	32
4.3.2.1 exit_printing()	32

4.4	femta.c File Reference	32
4.4.1	Macro Definition Documentation	33
4.4.1.1	I2C_STATE	33
4.4.1.2	NUMBER_OF_MODULES	33
4.4.1.3	UART_STATE	33
4.4.2	Function Documentation	33
4.4.2.1	check_if_readable()	34
4.4.2.2	check_if_writeable()	34
4.4.2.3	initialize_pin()	34
4.4.2.4	initialize_satellite()	35
4.4.2.5	main()	36
4.4.2.6	print_configuration()	36
4.4.2.7	read_voltage()	37
4.4.2.8	set_pwm()	37
4.4.2.9	set_voltage()	38
4.4.2.10	terminate_satellite()	38
4.5	femta.h File Reference	39
4.5.1	Typedef Documentation	40
4.5.1.1	I2C	40
4.5.1.2	module	40
4.5.1.3	pin	40
4.5.1.4	UART	40
4.5.2	Function Documentation	40
4.5.2.1	set_voltage()	41
4.5.3	Variable Documentation	41
4.5.3.1	FEMTA	41
4.5.3.2	modules	41
4.5.3.3	MPU	41
4.5.3.4	start_time	41
4.5.3.5	Valve	42



4.6	graphics.c File Reference	42
4.6.1	Macro Definition Documentation	43
4.6.1.1	I2C_STATE	43
4.6.1.2	NUMBER_OF_GRAPH_VIEWS	43
4.6.1.3	NUMBER_OF_MODULES	43
4.6.1.4	NUMBER_OF_PRINT_VIEWS	43
4.6.1.5	NUMBER_OF_SETUP_VIEWS	43
4.6.1.6	UART_STATE	43
4.6.2	Function Documentation	44
4.6.2.1	clear_print_window()	44
4.6.2.2	create_plot()	44
4.6.2.3	erase_print_window()	45
4.6.2.4	graph_plot()	45
4.6.2.5	initialize_graphics()	46
4.6.2.6	plot_add_value()	47
4.6.2.7	print()	47
4.6.2.8	print_window_title() [1/2]	48
4.6.2.9	print_window_title() [2/2]	48
4.6.2.10	terminate_graphics()	49
4.6.2.11	update_state_graphic()	49
4.6.3	Variable Documentation	49
4.6.3.1	graph_views	49
4.6.3.2	print_views	49
4.6.3.3	ready_to_graph	49
4.6.3.4	setup_views	50
4.7	graphics.h File Reference	50
4.7.1	Macro Definition Documentation	51
4.7.1.1	CONTROL_WINDOW	51
4.7.1.2	GENERAL_WINDOW	51
4.7.1.3	OPERATE_WINDOW	52

4.7.2	Typedef Documentation . . . . .	52
4.7.2.1	graph_view . . . . .	52
4.7.2.2	Plot . . . . .	52
4.7.2.3	print_view . . . . .	52
4.7.2.4	setup_view . . . . .	52
4.7.2.5	View . . . . .	52
4.7.3	Function Documentation . . . . .	52
4.7.3.1	clear_print_window() . . . . .	53
4.7.3.2	create_plot() . . . . .	53
4.7.3.3	erase_print_window() . . . . .	54
4.7.3.4	graph_plot() . . . . .	54
4.7.3.5	initialize_graphics() . . . . .	55
4.7.3.6	plot_add_value() . . . . .	56
4.7.3.7	print() . . . . .	56
4.7.3.8	terminate_graphics() . . . . .	57
4.7.3.9	update_state_graphic() . . . . .	57
4.7.4	Variable Documentation . . . . .	57
4.7.4.1	all_possible_owners . . . . .	58
4.7.4.2	graph_owner . . . . .	58
4.7.4.3	graph_owner_index_node . . . . .	58
4.7.4.4	number_of_data_points_plottable . . . . .	58
4.7.4.5	owner_index_list . . . . .	58
4.8	hashmap.c File Reference . . . . .	58
4.8.1	Function Documentation . . . . .	59
4.8.1.1	create_hashmap() . . . . .	59
4.8.1.2	hash() . . . . .	60
4.8.1.3	hashmap_add() . . . . .	60
4.8.1.4	hashmap_exists() . . . . .	61
4.8.1.5	hashmap_get() . . . . .	61
4.8.1.6	hashmap_remove() . . . . .	62

4.8.1.7	hashmap_update()	62
4.9	hashmap.h File Reference	63
4.9.1	Macro Definition Documentation	64
4.9.1.1	HASHMAP_DEFAULT_SIZE	64
4.9.1.2	HASHMAP_THRESHOLD	64
4.9.2	Typedef Documentation	64
4.9.2.1	Hashmap	64
4.9.2.2	HashmapElement	65
4.9.3	Function Documentation	65
4.9.3.1	create_hashmap()	65
4.9.3.2	hash()	66
4.10	i2c-interface.c File Reference	66
4.10.1	Macro Definition Documentation	68
4.10.1.1	ACCEL_CONFIG	68
4.10.1.2	ACCEL_CONFIG2	68
4.10.1.3	ACCEL_XOUT_H	68
4.10.1.4	AK8963_ADDRESS	68
4.10.1.5	AK8963_ASAX	69
4.10.1.6	AK8963_CNTL	69
4.10.1.7	AK8963_ST1	69
4.10.1.8	AK8963_XOUT_L	69
4.10.1.9	CONFIG	69
4.10.1.10	FIFO_COUNTH	69
4.10.1.11	FIFO_EN	69
4.10.1.12	FIFO_R_W	70
4.10.1.13	GYRO_CONFIG	70
4.10.1.14	GYRO_XOUT_H	70
4.10.1.15	I2C_MST_CTRL	70
4.10.1.16	INT_ENABLE	70
4.10.1.17	INT_PIN_CFG	70

4.10.1.18 MPU9250_ADDRESS . . . . .	70
4.10.1.19 PWR_MGMT_1 . . . . .	70
4.10.1.20 PWR_MGMT_2 . . . . .	71
4.10.1.21 SMPLRT_DIV . . . . .	71
4.10.1.22 TEMP_OUT_H . . . . .	71
4.10.1.23 TEMP_OUT_L . . . . .	71
4.10.1.24 USER_CTRL . . . . .	71
4.10.1.25 XA_OFFSET_H . . . . .	71
4.10.1.26 YA_OFFSET_H . . . . .	71
4.10.1.27 ZA_OFFSET_H . . . . .	71
4.10.2 Enumeration Type Documentation . . . . .	71
4.10.2.1 Ascale . . . . .	71
4.10.2.2 Gscale . . . . .	72
4.10.2.3 Mscale . . . . .	72
4.10.3 Function Documentation . . . . .	72
4.10.3.1 calibrateMPU9250() . . . . .	72
4.10.3.2 initAK8963() . . . . .	73
4.10.3.3 initialize_i2c() . . . . .	74
4.10.3.4 initMPU9250() . . . . .	75
4.10.3.5 log_mpu_data() . . . . .	75
4.10.3.6 printBias() . . . . .	76
4.10.3.7 printStartupConstants() . . . . .	77
4.10.3.8 readAccelData() . . . . .	77
4.10.3.9 readBytes() . . . . .	78
4.10.3.10 readGyroData() . . . . .	79
4.10.3.11 readMagData() . . . . .	79
4.10.3.12 readTempData() . . . . .	80
4.10.3.13 resetMPU9250() . . . . .	80
4.10.3.14 terminate_mpu_logging() . . . . .	80
4.10.4 Variable Documentation . . . . .	81

4.10.4.1	accelBias	81
4.10.4.2	aRes	81
4.10.4.3	Ascale	81
4.10.4.4	gRes	81
4.10.4.5	Gscale	81
4.10.4.6	gyroBias	81
4.10.4.7	magBias	81
4.10.4.8	magCalibration	82
4.10.4.9	magScale	82
4.10.4.10	Mmode	82
4.10.4.11	mpu_log_file	82
4.10.4.12	mpu_log_file_name	82
4.10.4.13	mpu_termination_signal	82
4.10.4.14	mpu_thread	82
4.10.4.15	mpu_values_read	82
4.10.4.16	mRes	83
4.10.4.17	Mscale	83
4.10.4.18	newMagData	83
4.11	i2c-interface.h File Reference	83
4.11.1	Typedef Documentation	84
4.11.1.1	I2C	84
4.11.1.2	module	84
4.11.2	Function Documentation	84
4.11.2.1	initialize_i2c()	85
4.11.2.2	printStartupConstants()	86
4.11.2.3	terminate_mpu_logging()	86
4.11.3	Variable Documentation	86
4.11.3.1	i2c_device	87
4.11.3.2	mpu_acel_plot	87
4.11.3.3	mpu_gyro_plot	87

4.11.3.4	<a href="#">mpu_logger</a>	87
4.11.3.5	<a href="#">mpu_magn_plot</a>	87
4.12	<a href="#">linked-list.c File Reference</a>	87
4.12.1	<a href="#">Function Documentation</a>	88
4.12.1.1	<a href="#">create_list()</a>	88
4.12.1.2	<a href="#">create_node()</a>	88
4.12.1.3	<a href="#">list_insert()</a>	89
4.12.1.4	<a href="#">list_remove()</a>	89
4.13	<a href="#">linked-list.h File Reference</a>	90
4.13.1	<a href="#">Macro Definition Documentation</a>	91
4.13.1.1	<a href="#">FLOAT_NODE</a>	91
4.13.1.2	<a href="#">INTEGER_NODE</a>	91
4.13.1.3	<a href="#">STRING_NODE</a>	91
4.13.2	<a href="#">Typedef Documentation</a>	91
4.13.2.1	<a href="#">List</a>	91
4.13.2.2	<a href="#">Node</a>	91
4.13.3	<a href="#">Function Documentation</a>	91
4.13.3.1	<a href="#">create_list()</a>	92
4.13.3.2	<a href="#">create_node()</a>	92
4.13.3.3	<a href="#">list_insert()</a>	93
4.13.3.4	<a href="#">list_remove()</a>	93
4.14	<a href="#">logger.c File Reference</a>	93
4.14.1	<a href="#">Function Documentation</a>	94
4.14.1.1	<a href="#">close_prototype()</a>	94
4.14.1.2	<a href="#">create_logger()</a>	95
4.14.1.3	<a href="#">destroy_prototype()</a>	95
4.14.1.4	<a href="#">open_prototype()</a>	96
4.15	<a href="#">logger.h File Reference</a>	96
4.15.1	<a href="#">Typedef Documentation</a>	97
4.15.1.1	<a href="#">Logger</a>	97

4.15.2	Function Documentation	97
4.15.2.1	create_logger()	97
4.16	scripter.c File Reference	98
4.16.1	Function Documentation	98
4.16.1.1	define_script_action()	98
4.16.1.2	execute_script()	99
4.16.1.3	initialize_scripter()	99
4.17	scripter.h File Reference	100
4.17.1	Typedef Documentation	101
4.17.1.1	lambda	101
4.17.2	Function Documentation	101
4.17.2.1	define_script_action()	101
4.17.2.2	execute_script()	102
4.17.2.3	initialize_scripter()	102
4.17.3	Variable Documentation	103
4.17.3.1	action_table	103
4.18	selector.c File Reference	103
4.18.1	Function Documentation	104
4.18.1.1	add_selector_command()	104
4.18.1.2	change_selector()	105
4.18.1.3	create_selector()	105
4.18.1.4	cycle_graph()	106
4.18.1.5	execute_selector()	107
4.18.1.6	flip_bool()	107
4.18.1.7	flip_femta()	107
4.18.1.8	flip_valve()	108
4.18.1.9	present_selector()	108
4.18.1.10	rotate()	109
4.18.1.11	write_message()	109
4.19	selector.h File Reference	110

4.19.1	Typedef Documentation . . . . .	111
4.19.1.1	Command . . . . .	111
4.19.1.2	lambda . . . . .	111
4.19.1.3	Selector . . . . .	111
4.19.2	Function Documentation . . . . .	111
4.19.2.1	add_selector_command() . . . . .	112
4.19.2.2	change_selector() . . . . .	112
4.19.2.3	create_selector() . . . . .	113
4.19.2.4	cycle_graph() . . . . .	113
4.19.2.5	execute_selector() . . . . .	114
4.19.2.6	flip_bool() . . . . .	114
4.19.2.7	flip_femta() . . . . .	114
4.19.2.8	flip_valve() . . . . .	115
4.19.2.9	present_selector() . . . . .	115
4.19.2.10	rotate() . . . . .	116
4.19.2.11	write_message() . . . . .	116
4.19.3	Variable Documentation . . . . .	116
4.19.3.1	visible_selector . . . . .	117
4.20	temperature-monitoring.c File Reference . . . . .	117
4.20.1	Function Documentation . . . . .	117
4.20.1.1	initialize_temperature_monitoring() . . . . .	118
4.20.1.2	read_cpu_temperature() . . . . .	118
4.20.1.3	terminate_temperature_monitoring() . . . . .	119
4.20.2	Variable Documentation . . . . .	119
4.20.2.1	cpu_temperature_log_file . . . . .	119
4.20.2.2	cpu_temperature_thread . . . . .	119
4.20.2.3	temperature_log_filename . . . . .	119
4.20.2.4	termination_signal . . . . .	120
4.20.2.5	values_read . . . . .	120
4.21	temperature-monitoring.h File Reference . . . . .	120
4.21.1	Function Documentation . . . . .	121
4.21.1.1	initialize_temperature_monitoring() . . . . .	121
4.21.1.2	terminate_temperature_monitoring() . . . . .	122
4.21.2	Variable Documentation . . . . .	122
4.21.2.1	temperature_plot . . . . .	122
4.22	timing.c File Reference . . . . .	122
4.22.1	Function Documentation . . . . .	123
4.22.1.1	nano_sleep() . . . . .	123
4.23	timing.h File Reference . . . . .	123
4.23.1	Function Documentation . . . . .	123
4.23.1.1	nano_sleep() . . . . .	124



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">Command</a>	5
<a href="#">graph_view</a>	6
<a href="#">Hashmap</a>	7
<a href="#">HashmapElement</a>	9
<a href="#">I2C</a>	9
<a href="#">List</a>	11
<a href="#">Logger</a>	12
<a href="#">module</a>	14
<a href="#">Node</a>	16
<a href="#">pin</a>	18
<a href="#">Plot</a>	19
<a href="#">print_view</a>	21
<a href="#">Selector</a>	23
<a href="#">setup_view</a>	24
<a href="#">View</a>	25



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

colors.h	27
error.c	30
error.h	31
femta.c	32
femta.h	39
graphics.c	42
graphics.h	50
hashmap.c	58
hashmap.h	63
i2c-interface.c	66
i2c-interface.h	83
linked-list.c	87
linked-list.h	90
logger.c	93
logger.h	96
scripter.c	98
scripter.h	100
selector.c	103
selector.h	110
temperature-monitoring.c	117
temperature-monitoring.h	120
timing.c	122
timing.h	123



## Chapter 3

# Class Documentation

### 3.1 Command Struct Reference

```
#include <selector.h>
```

#### Public Attributes

- char [key](#)
- char \* [text](#)
- [lambda](#) [action](#)
- void \* [argument](#)

#### 3.1.1 Member Data Documentation

##### 3.1.1.1 [action](#)

```
lambda Command::action
```

##### 3.1.1.2 [argument](#)

```
void* Command::argument
```

##### 3.1.1.3 [key](#)

```
char Command::key
```

#### 3.1.1.4 text

```
char* Command::text
```

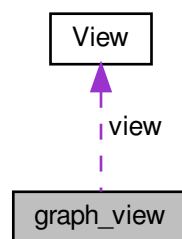
The documentation for this struct was generated from the following file:

- [selector.h](#)

## 3.2 graph\_view Struct Reference

```
#include <graphics.h>
```

Collaboration diagram for graph\_view:



### Public Attributes

- [View](#) \* [view](#)
- unsigned char [vertical\\_tick\\_marks](#)
- unsigned char [horizontal\\_tick\\_marks](#)

### 3.2.1 Member Data Documentation

#### 3.2.1.1 horizontal\_tick\_marks

```
unsigned char graph_view::horizontal_tick_marks
```

## 3.2.1.2 vertical\_tick\_marks

```
unsigned char graph_view::vertical_tick_marks
```

## 3.2.1.3 view

```
View* graph_view::view
```

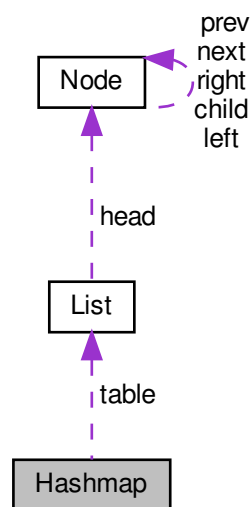
The documentation for this struct was generated from the following file:

- [graphics.h](#)

## 3.3 Hashmap Struct Reference

```
#include <hashmap.h>
```

Collaboration diagram for Hashmap:



### Public Attributes

- unsigned int [elements](#)
- unsigned int [size](#)
- [List](#) \*\* [table](#)
- void (\* [get](#) )(Hashmap \*this, char \*string)
- void (\* [add](#) )(Hashmap \*this, char \*string, void \*datum)
- void (\* [remove](#) )(Hashmap \*this, char \*string)
- bool (\* [exists](#) )(Hashmap \*this, char \*string)
- void (\* [update](#) )(Hashmap \*this, char \*string, void \*datum)

### 3.3.1 Member Data Documentation

#### 3.3.1.1 add

```
void(* Hashmap::add) (Hashmap *this, char *string, void *datum)
```

#### 3.3.1.2 elements

```
unsigned int Hashmap::elements
```

#### 3.3.1.3 exists

```
bool(* Hashmap::exists) (Hashmap *this, char *string)
```

#### 3.3.1.4 get

```
void*(* Hashmap::get) (Hashmap *this, char *string)
```

#### 3.3.1.5 remove

```
void(* Hashmap::remove) (Hashmap *this, char *string)
```

#### 3.3.1.6 size

```
unsigned int Hashmap::size
```

#### 3.3.1.7 table

```
List** Hashmap::table
```



### 3.3.1.8 update

```
void(* Hashmap::update) (Hashmap *this, char *string, void *datum)
```

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

- [hashmap.h](#)

## 3.4 HashmapElement Struct Reference

```
#include <hashmap.h>
```

### Public Attributes

- char \* [key](#)
- void \* [datum](#)

### 3.4.1 Member Data Documentation

#### 3.4.1.1 datum

```
void* HashmapElement::datum
```

#### 3.4.1.2 key

```
char* HashmapElement::key
```

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

- [hashmap.h](#)

## 3.5 I2C Struct Reference

```
#include <i2c-interface.h>
```

## Public Attributes

- unsigned char [i2c\\_address](#)
- unsigned char [i2c\\_slave\\_address](#)
- short \* [registers](#)
- void(\* [gyros](#) )(float \*axes)
- void(\* [accelerometers](#) )(float \*axes)
- void(\* [magnetometers](#) )(float \*axes)
- float(\* [temperature](#) )()

## 3.5.1 Member Data Documentation

### 3.5.1.1 accelerometers

```
void(* I2C::accelerometers) (float *axes)
```

### 3.5.1.2 gyros

```
void(* I2C::gyros) (float *axes)
```

### 3.5.1.3 i2c\_address

```
unsigned char I2C::i2c_address
```

### 3.5.1.4 i2c\_slave\_address

```
unsigned char I2C::i2c_slave_address
```

### 3.5.1.5 magnetometers

```
void(* I2C::magnetometers) (float *axes)
```

### 3.5.1.6 registers

```
short* I2C::registers
```

### 3.5.1.7 temperature

```
float(* I2C::temperature) ()
```

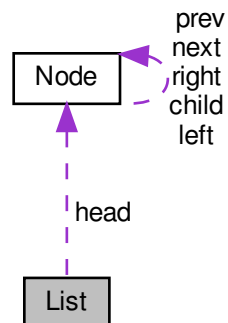
The documentation for this struct was generated from the following file:

- [i2c-interface.h](#)

## 3.6 List Struct Reference

```
#include <linked-list.h>
```

Collaboration diagram for List:



### Public Attributes

- [Node \\* head](#)
- unsigned int [elements](#)
- unsigned int [elements\\_limit](#)
- bool [doublely\\_linked](#)

### 3.6.1 Member Data Documentation

### 3.6.1.1 doubly\_linked

```
bool List::doubly_linked
```

### 3.6.1.2 elements

```
unsigned int List::elements
```

### 3.6.1.3 elements\_limit

```
unsigned int List::elements_limit
```

### 3.6.1.4 head

```
Node* List::head
```

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

- [linked-list.h](#)

## 3.7 Logger Struct Reference

```
#include <logger.h>
```

Collaboration diagram for Logger:



## Public Attributes

- `Logger * self`
- `FILE * file`
- `char * filename`
- `pthread_t thread`
- `bool termination_signal`
- `int values_read`
- `bool(* open)(Logger *self)`
- `bool(* close)(Logger *self)`
- `void(* destroy)(Logger *self)`

### 3.7.1 Member Data Documentation

#### 3.7.1.1 close

```
bool(* Logger::close) (Logger *self)
```

#### 3.7.1.2 destroy

```
void(* Logger::destroy) (Logger *self)
```

#### 3.7.1.3 file

```
FILE* Logger::file
```

#### 3.7.1.4 filename

```
char* Logger::filename
```

#### 3.7.1.5 open

```
bool(* Logger::open) (Logger *self)
```

### 3.7.1.6 self

```
Logger* Logger::self
```

### 3.7.1.7 termination\_signal

```
bool Logger::termination_signal
```

### 3.7.1.8 thread

```
pthread_t Logger::thread
```

### 3.7.1.9 values\_read

```
int Logger::values_read
```

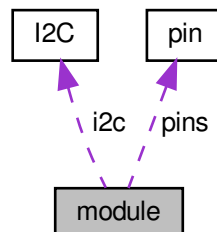
The documentation for this struct was generated from the following file:

- [logger.h](#)

## 3.8 module Struct Reference

```
#include <femta.h>
```

Collaboration diagram for module:



## Public Attributes

- char \* [identifier](#)
- [pin](#) \* pins
- char [n\\_pins](#)
- [I2C](#) \* [i2c](#)
- [UART](#) \* [uart](#)
- bool [initialized](#)
- bool [loaded](#)

## 3.8.1 Member Data Documentation

### 3.8.1.1 [i2c](#)

[I2C](#)\* module::i2c

### 3.8.1.2 [identifier](#)

char\* module::identifier

### 3.8.1.3 [initialized](#)

bool module::initialized

### 3.8.1.4 [loaded](#)

bool module::loaded

### 3.8.1.5 [n\\_pins](#)

char module::n\_pins

### 3.8.1.6 pins

```
pin* module::pins
```

### 3.8.1.7 uart

```
UART* module::uart
```

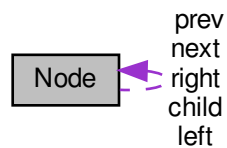
The documentation for this struct was generated from the following file:

- [femta.h](#)

## 3.9 Node Struct Reference

```
#include <linked-list.h>
```

Collaboration diagram for Node:



### Public Attributes

- union {  
    [Node](#) \* [next](#)  
    [Node](#) \* [right](#)  
};
- union {  
    [Node](#) \* [prev](#)  
    [Node](#) \* [left](#)  
    [Node](#) \* [child](#)  
};
- void \* [value](#)

### 3.9.1 Member Data Documentation



#### 3.9.1.1 "@3

```
union { ... }
```

#### 3.9.1.2 "@5

```
union { ... }
```

#### 3.9.1.3 child

```
Node* Node::child
```

#### 3.9.1.4 left

```
Node* Node::left
```

#### 3.9.1.5 next

```
Node* Node::next
```

#### 3.9.1.6 prev

```
Node* Node::prev
```

#### 3.9.1.7 right

```
Node* Node::right
```

### 3.9.1.8 value

```
void* Node::value
```

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

- [linked-list.h](#)

## 3.10 pin Struct Reference

```
#include <femta.h>
```

### Public Attributes

- char [state](#)
  - char [logical](#)
  - char [physical](#)
  - union {
    - char [voltage](#)
    - unsigned char [duty\\_cycle](#)
- ```
};
```

### 3.10.1 Member Data Documentation

#### 3.10.1.1 "@1

```
union { ... }
```

#### 3.10.1.2 duty\_cycle

```
unsigned char pin::duty_cycle
```

#### 3.10.1.3 logical

```
char pin::logical
```

#### 3.10.1.4 physical

```
char pin::physical
```

#### 3.10.1.5 state

```
char pin::state
```

#### 3.10.1.6 voltage

```
char pin::voltage
```

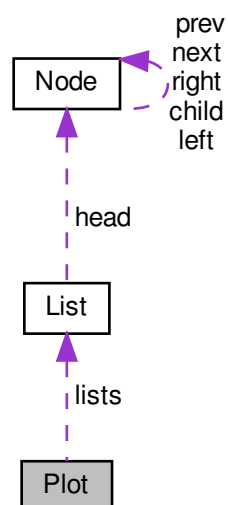
The documentation for this struct was generated from the following file:

- [femta.h](#)

## 3.11 Plot Struct Reference

```
#include <graphics.h>
```

Collaboration diagram for Plot:



## Public Attributes

- char \* [name](#)
- [List](#) \*\* [lists](#)
- unsigned char [number\\_of\\_lists](#)
- float [min\\_value](#)
- float [max\\_value](#)
- bool [has\\_data](#)

### 3.11.1 Member Data Documentation

#### 3.11.1.1 [has\\_data](#)

```
bool Plot::has_data
```

#### 3.11.1.2 [lists](#)

```
List** Plot::lists
```

#### 3.11.1.3 [max\\_value](#)

```
float Plot::max_value
```

#### 3.11.1.4 [min\\_value](#)

```
float Plot::min_value
```

#### 3.11.1.5 [name](#)

```
char* Plot::name
```

## 3.11.1.6 number\_of\_lists

```
unsigned char Plot::number_of_lists
```

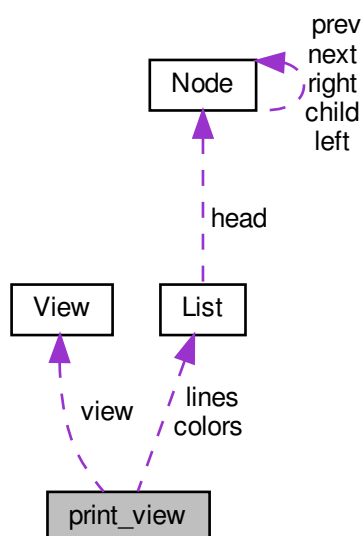
The documentation for this struct was generated from the following file:

- [graphics.h](#)

## 3.12 print\_view Struct Reference

```
#include <graphics.h>
```

Collaboration diagram for print\_view:



## Public Attributes

- [View](#) \* `view`
- [List](#) \* `lines`
- [List](#) \* `colors`
- unsigned char `number_lines_printed`
- unsigned char `current_view_line`
- unsigned char `number_of_lines`

## 3.12.1 Member Data Documentation

### 3.12.1.1 colors

`List* print_view::colors`

### 3.12.1.2 current\_view\_line

`unsigned char print_view::current_view_line`

### 3.12.1.3 lines

`List* print_view::lines`

### 3.12.1.4 number\_lines\_printed

`unsigned char print_view::number_lines_printed`

### 3.12.1.5 number\_of\_lines

`unsigned char print_view::number_of_lines`

### 3.12.1.6 view

`View* print_view::view`

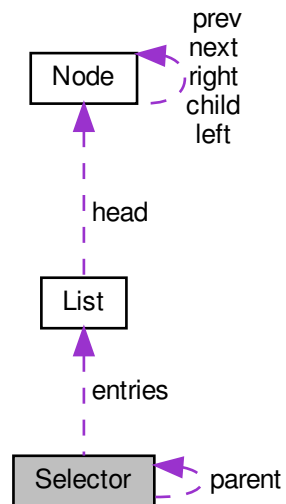
The documentation for this struct was generated from the following file:

- [graphics.h](#)

## 3.13 Selector Struct Reference

```
#include <selector.h>
```

Collaboration diagram for Selector:



### Public Attributes

- char \* [title](#)
- List \* [entries](#)
- Selector \* [parent](#)

#### 3.13.1 Member Data Documentation

##### 3.13.1.1 entries

```
List* Selector::entries
```

##### 3.13.1.2 parent

```
Selector* Selector::parent
```

### 3.13.1.3 title

```
char* Selector::title
```

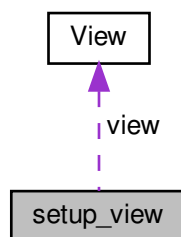
The documentation for this struct was generated from the following file:

- [selector.h](#)

## 3.14 setup\_view Struct Reference

```
#include <graphics.h>
```

Collaboration diagram for setup\_view:



### Public Attributes

- [View](#) \* [view](#)

### 3.14.1 Member Data Documentation

#### 3.14.1.1 view

```
View* setup_view::view
```

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

- [graphics.h](#)



## 3.15 View Struct Reference

```
#include <graphics.h>
```

### Public Attributes

- WINDOW \* [window](#)
- unsigned char [inner\\_width](#)
- unsigned char [inner\\_height](#)
- unsigned char [outer\\_width](#)
- unsigned char [outer\\_height](#)

### 3.15.1 Member Data Documentation

#### 3.15.1.1 inner\_height

```
unsigned char View::inner_height
```

#### 3.15.1.2 inner\_width

```
unsigned char View::inner_width
```

#### 3.15.1.3 outer\_height

```
unsigned char View::outer_height
```

#### 3.15.1.4 outer\_width

```
unsigned char View::outer_width
```

#### 3.15.1.5 window

```
WINDOW* View::window
```

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

- [graphics.h](#)

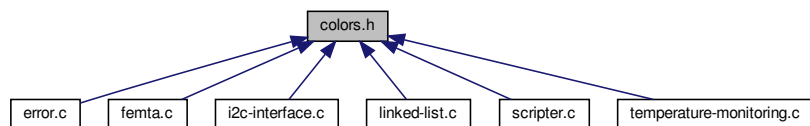


## Chapter 4

# File Documentation

### 4.1 colors.h File Reference

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



#### Macros

- `#define RED "\e[0;31m"`
- `#define GREY "\e[0;35m"`
- `#define BLUE "\e[0;34m"`
- `#define GREEN "\e[0;32m"`
- `#define PURPLE "\e[0;35m"`
- `#define YELLOW "\e[0;33m"`
- `#define RESET "\e[0m"`
- `#define DIM "\e[2m"`
- `#define UNDIM "\e[22m"`
- `#define CONSOLE_RED "\e[31m"`
- `#define CONSOLE_GREEN "\e[32m"`
- `#define CONSOLE_YELLOW "\e[33m"`
- `#define CONSOLE_BLUE "\e[34m"`
- `#define CONSOLE_MAGENTA "\e[35m"`
- `#define CONSOLE_CYAN "\e[36m"`
- `#define CONSOLE_GRAY "\e[37m"`
- `#define CONSOLE_RESET "\e[39m"`

#### 4.1.1 Macro Definition Documentation

#### 4.1.1.1 BLUE

```
#define BLUE "\e[0;34m"
```

#### 4.1.1.2 CONSOLE\_BLUE

```
#define CONSOLE_BLUE "\e[34m"
```

#### 4.1.1.3 CONSOLE\_CYAN

```
#define CONSOLE_CYAN "\e[36m"
```

#### 4.1.1.4 CONSOLE\_GRAY

```
#define CONSOLE_GRAY "\e[37m"
```

#### 4.1.1.5 CONSOLE\_GREEN

```
#define CONSOLE_GREEN "\e[32m"
```

#### 4.1.1.6 CONSOLE\_MAGENTA

```
#define CONSOLE_MAGENTA "\e[35m"
```

#### 4.1.1.7 CONSOLE\_RED

```
#define CONSOLE_RED "\e[31m"
```

#### 4.1.1.8 CONSOLE\_RESET

```
#define CONSOLE_RESET "\e[39m"
```

#### 4.1.1.9 CONSOLE\_YELLOW

```
#define CONSOLE_YELLOW "\e[33m"
```

#### 4.1.1.10 DIM

```
#define DIM "\e[2m"
```

#### 4.1.1.11 GREEN

```
#define GREEN "\e[0;32m"
```

#### 4.1.1.12 GREY

```
#define GREY "\e[0;35m"
```

#### 4.1.1.13 PURPLE

```
#define PURPLE "\e[0;35m"
```

#### 4.1.1.14 RED

```
#define RED "\e[0;31m"
```

#### 4.1.1.15 RESET

```
#define RESET "\e[0m"
```

#### 4.1.1.16 UNDIM

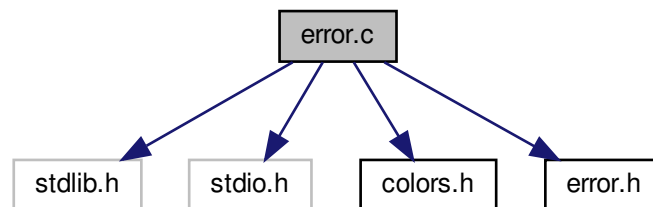
```
#define UNDIM "\e[22m"
```

#### 4.1.1.17 YELLOW

```
#define YELLOW "\e[0;33m"
```

## 4.2 error.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "colors.h"
#include "error.h"
Include dependency graph for error.c:
```



### Functions

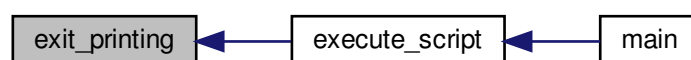
- void `exit_printing` (char \*message, char code)

#### 4.2.1 Function Documentation

##### 4.2.1.1 exit\_printing()

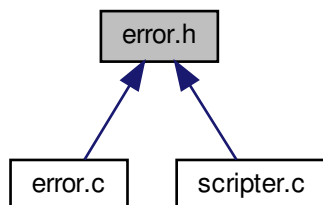
```
void exit_printing (
    char * message,
    char code )
```

Here is the caller graph for this function:



## 4.3 error.h File Reference

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



### Macros

- `#define ERROR_PROGRAMMER 1`
- `#define ERROR_OS_FAILURE 2`
- `#define ERROR_LIBRARY_FAILURE 3`

### Functions

- `void exit_printing (char *message, char code)`

#### 4.3.1 Macro Definition Documentation

##### 4.3.1.1 ERROR\_LIBRARY\_FAILURE

```
#define ERROR_LIBRARY_FAILURE 3
```

##### 4.3.1.2 ERROR\_OS\_FAILURE

```
#define ERROR_OS_FAILURE 2
```

##### 4.3.1.3 ERROR\_PROGRAMMER

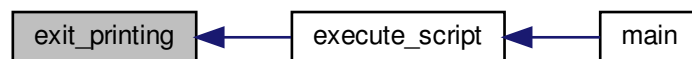
```
#define ERROR_PROGRAMMER 1
```

## 4.3.2 Function Documentation

### 4.3.2.1 exit\_printing()

```
void exit_printing (
    char * message,
    char code )
```

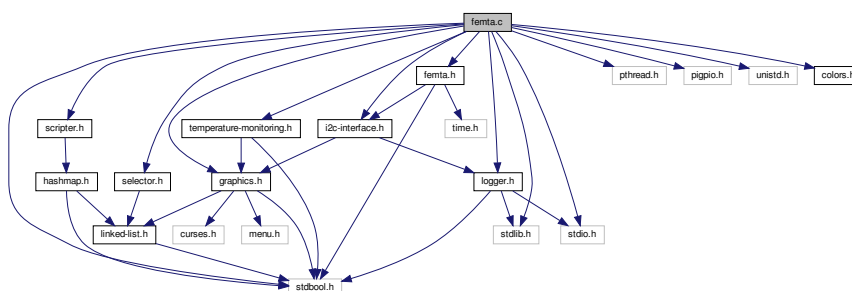
Here is the caller graph for this function:



## 4.4 femta.c File Reference

```
#include <stdbool.h>
#include <pthread.h>
#include <stdlib.h>
#include <pigpio.h>
#include <unistd.h>
#include <stdio.h>
#include "femta.h"
#include "i2c-interface.h"
#include "temperature-monitoring.h"
#include "graphics.h"
#include "selector.h"
#include "scripter.h"
#include "logger.h"
#include "colors.h"
```

Include dependency graph for femta.c:





## Macros

- #define `NUMBER_OF_MODULES` 3
- #define `I2C_STATE` 2
- #define `UART_STATE` 3

## Functions

- void `initialize_pin` (`pin` \*initialent, char logical, char physical, short state)
- void `initialize_satellite` ()
- void `print_configuration` ()
- void `terminate_satellite` ()
- void `check_if_writeable` (`pin` \*p)
- void `check_if_readable` (`pin` \*p)
- char `read_voltage` (`pin` \*p)
- void `set_voltage` (`pin` \*p, char voltage)
- void `set_pwm` (`pin` \*p, unsigned char duty\_cycle)
- int `main` ()

### 4.4.1 Macro Definition Documentation

#### 4.4.1.1 I2C\_STATE

```
#define I2C_STATE 2
```

#### 4.4.1.2 NUMBER\_OF\_MODULES

```
#define NUMBER_OF_MODULES 3
```

#### 4.4.1.3 UART\_STATE

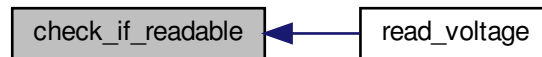
```
#define UART_STATE 3
```

### 4.4.2 Function Documentation

#### 4.4.2.1 check\_if\_readable()

```
void check_if_readable (
    pin * p )
```

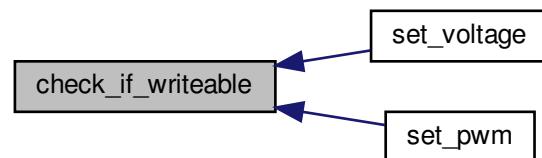
Here is the caller graph for this function:



#### 4.4.2.2 check\_if\_writeable()

```
void check_if_writeable (
    pin * p )
```

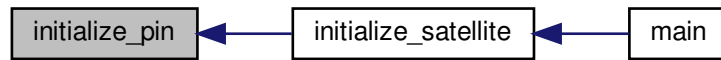
Here is the caller graph for this function:



#### 4.4.2.3 initialize\_pin()

```
void initialize_pin (
    pin * initialent,
    char logical,
    char physical,
    short state )
```

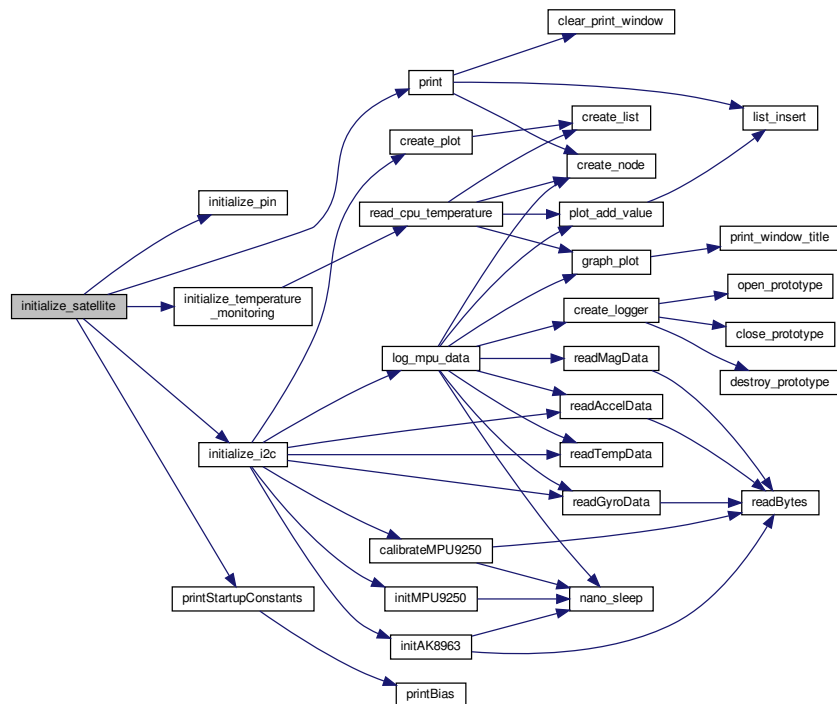
Here is the caller graph for this function:



#### 4.4.2.4 initialize\_satellite()

```
void initialize_satellite ( )
```

Here is the call graph for this function:

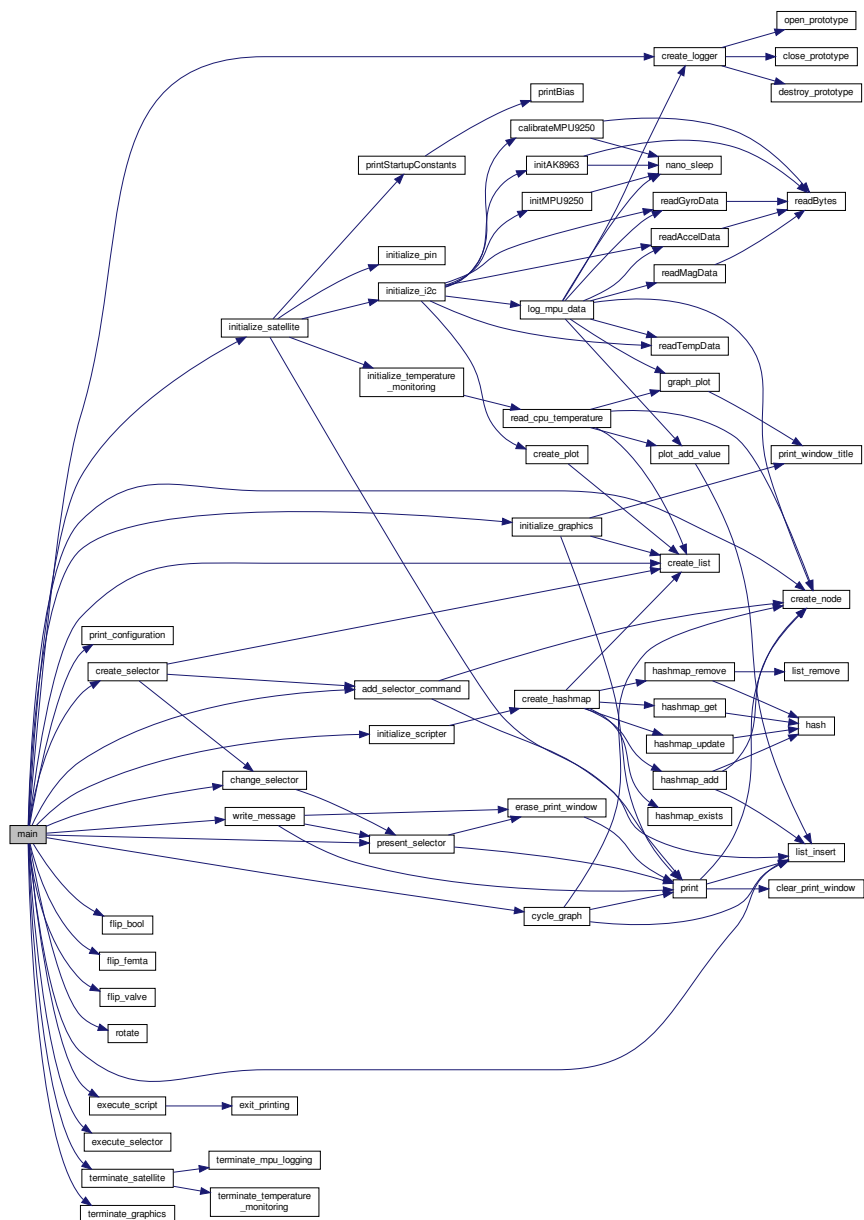


Here is the caller graph for this function:



```
int main ( )
```

Here is the call graph for this function:



## 4.4.2.6 print\_configuration()

```
void print_configuration ( )
```

Here is the caller graph for this function:



## 4.4.2.7 read\_voltage()

```
char read_voltage (
    pin * p )
```

Here is the call graph for this function:



## 4.4.2.8 set\_pwm()

```
void set_pwm (
    pin * p,
    unsigned char duty_cycle )
```

Here is the call graph for this function:



#### 4.4.2.9 set\_voltage()

```
void set_voltage (
    pin * p,
    char voltage )
```

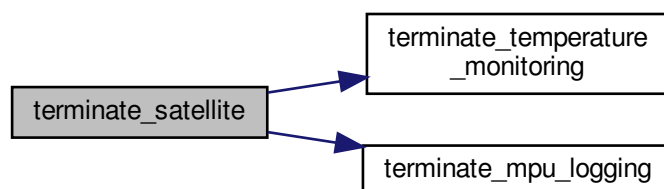
Here is the call graph for this function:



#### 4.4.2.10 terminate\_satellite()

```
void terminate_satellite ( )
```

Here is the call graph for this function:

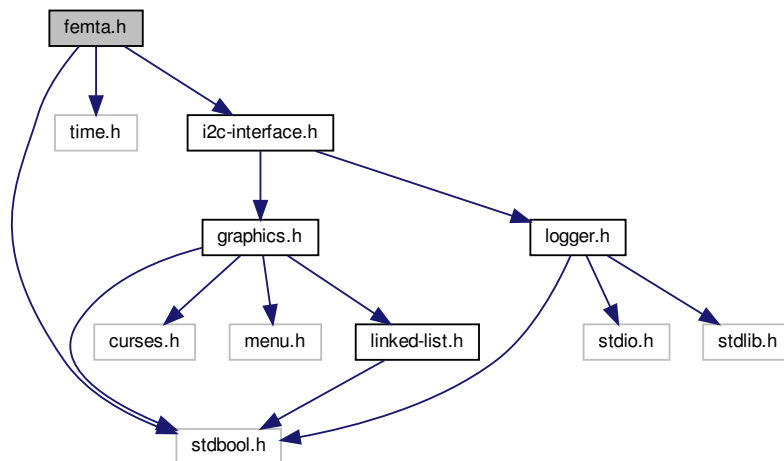


Here is the caller graph for this function:

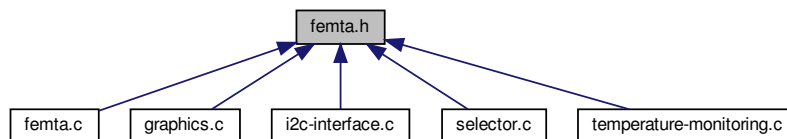


## 4.5 femta.h File Reference

```
#include <stdbool.h>
#include <time.h>
#include "i2c-interface.h"
Include dependency graph for femta.h:
```



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



### Classes

- struct [pin](#)
- struct [module](#)

### Typedefs

- typedef struct [pin](#) pin
- typedef struct [I2C](#) I2C
- typedef struct [UART](#) UART
- typedef struct [module](#) module

## Functions

- void `set_voltage` (`pin` \*p, char voltage)

## Variables

- `module` \*\* `modules`
- `module` \* `MPU`
- `module` \* `Valve`
- `module` \* `FEMTA`
- `time_t` `start_time`

## 4.5.1 Typedef Documentation

### 4.5.1.1 I2C

```
typedef struct I2C I2C
```

### 4.5.1.2 module

```
typedef struct module module
```

### 4.5.1.3 pin

```
typedef struct pin pin
```

### 4.5.1.4 UART

```
typedef struct UART UART
```

## 4.5.2 Function Documentation



#### 4.5.2.1 set\_voltage()

```
void set_voltage (
    pin * p,
    char voltage )
```

Here is the call graph for this function:



### 4.5.3 Variable Documentation

#### 4.5.3.1 FEMTA

```
module * FEMTA
```

#### 4.5.3.2 modules

```
module** modules
```

#### 4.5.3.3 MPU

```
module * MPU
```

#### 4.5.3.4 start\_time

```
time_t start_time
```

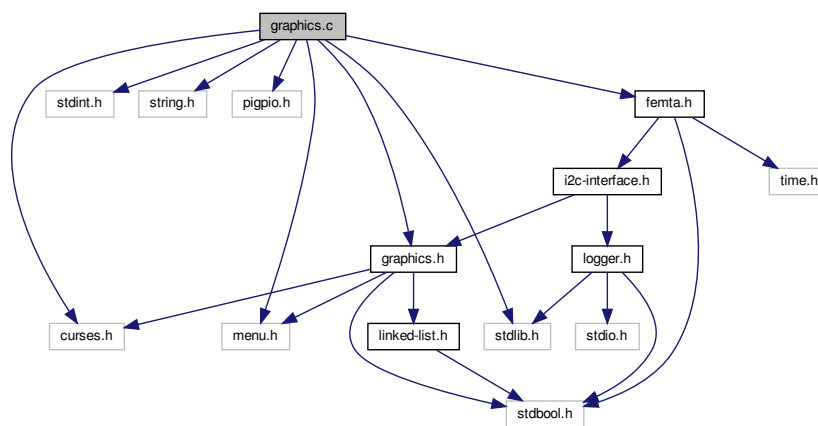
#### 4.5.3.5 Valve

module \* Valve

## 4.6 graphics.c File Reference

```
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <pigpio.h>
#include <curses.h>
#include <menu.h>
#include "graphics.h"
#include "femta.h"
```

Include dependency graph for graphics.c:



### Macros

- `#define` `NUMBER_OF_MODULES` 3
- `#define` `I2C_STATE` 2
- `#define` `UART_STATE` 3
- `#define` `NUMBER_OF_PRINT_VIEWS` 3
- `#define` `NUMBER_OF_GRAPH_VIEWS` 1
- `#define` `NUMBER_OF_SETUP_VIEWS` 1

### Functions

- void `print_window_title` ()
- void `initialize_graphics` ()
- void `terminate_graphics` ()
- void `print_window_title` (WINDOW \*win, int starty, int startx, int width, char \*string, chtype color)
- `Plot` \* `create_plot` (char \*name, unsigned char number\_of\_lists)
- void `clear_print_window` (unsigned char window\_number)
- void `print` (unsigned char window\_number, char \*string, unsigned int color)
- void `erase_print_window` (unsigned char window\_number)
- void `update_state_graphic` (unsigned char line, bool state)
- void `plot_add_value` (Plot \*plot, List \*list, Node \*node)
- void `graph_plot` (Plot \*plot)

## Variables

- bool `ready_to_graph` = false
- `print_view` \*\* `print_views`
- `graph_view` \*\* `graph_views`
- `setup_view` \*\* `setup_views`

## 4.6.1 Macro Definition Documentation

### 4.6.1.1 I2C\_STATE

```
#define I2C_STATE 2
```

### 4.6.1.2 NUMBER\_OF\_GRAPH\_VIEWS

```
#define NUMBER_OF_GRAPH_VIEWS 1
```

### 4.6.1.3 NUMBER\_OF\_MODULES

```
#define NUMBER_OF_MODULES 3
```

### 4.6.1.4 NUMBER\_OF\_PRINT\_VIEWS

```
#define NUMBER_OF_PRINT_VIEWS 3
```

### 4.6.1.5 NUMBER\_OF\_SETUP\_VIEWS

```
#define NUMBER_OF_SETUP_VIEWS 1
```

### 4.6.1.6 UART\_STATE

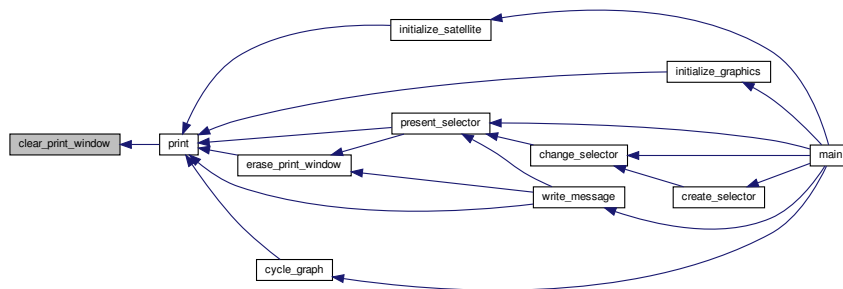
```
#define UART_STATE 3
```

## 4.6.2 Function Documentation

### 4.6.2.1 `clear_print_window()`

```
void clear_print_window (
    unsigned char window_number )
```

Here is the caller graph for this function:



### 4.6.2.2 `create_plot()`

```
Plot* create_plot (
    char * name,
    unsigned char number_of_lists )
```

Here is the call graph for this function:



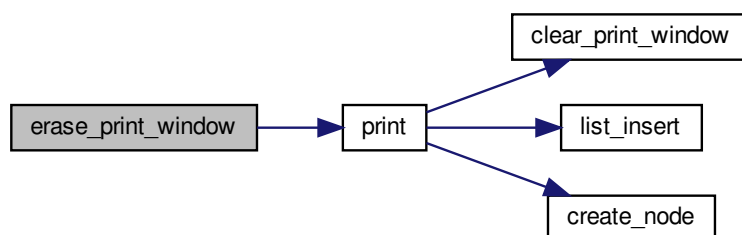
Here is the caller graph for this function:



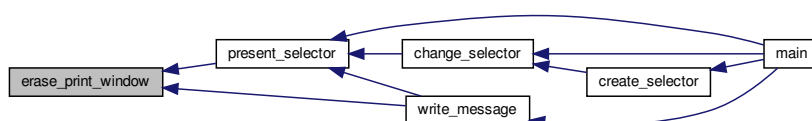
## 4.6.2.3 erase\_print\_window()

```
void erase_print_window (  
    unsigned char window_number )
```

Here is the call graph for this function:



Here is the caller graph for this function:



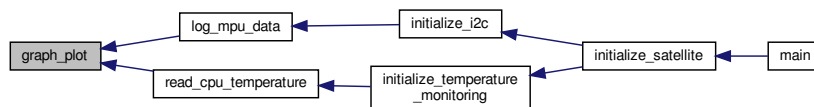
## 4.6.2.4 graph\_plot()

```
void graph_plot (  
    Plot * plot )
```

Here is the call graph for this function:



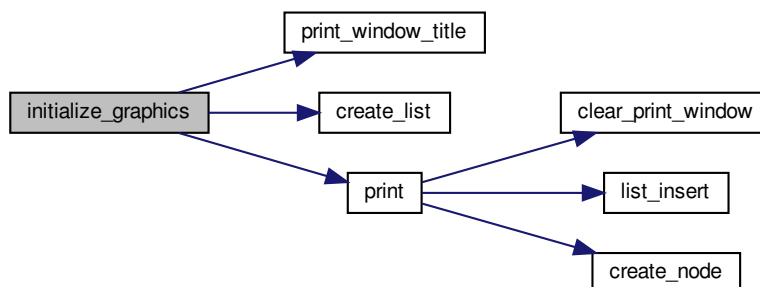
Here is the caller graph for this function:



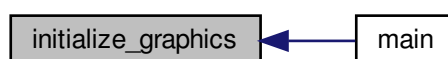
#### 4.6.2.5 initialize\_graphics()

```
void initialize_graphics ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



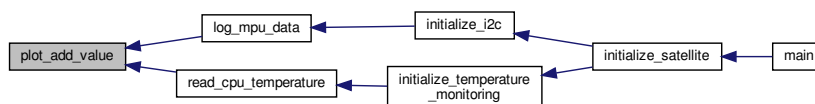
## 4.6.2.6 plot\_add\_value()

```
void plot_add_value (
    Plot * plot,
    List * list,
    Node * node )
```

Here is the call graph for this function:



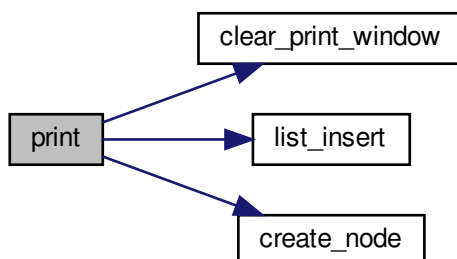
Here is the caller graph for this function:



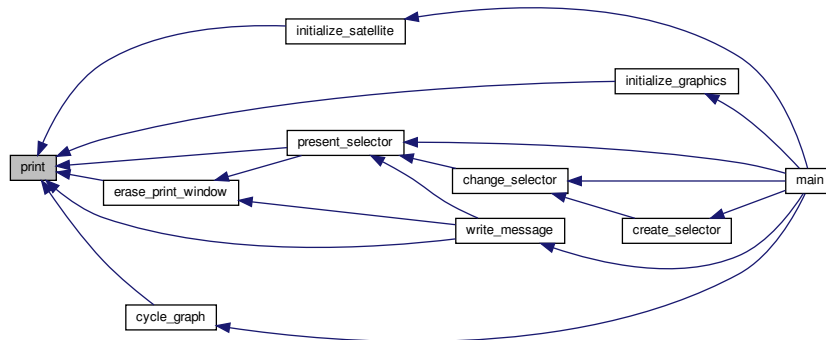
## 4.6.2.7 print()

```
void print (
    unsigned char window_number,
    char * string,
    unsigned int color )
```

Here is the call graph for this function:



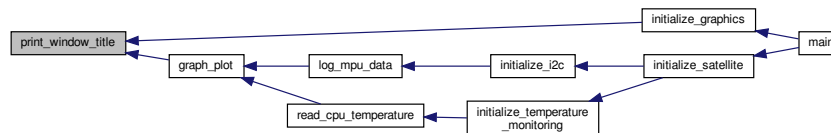
Here is the caller graph for this function:



#### 4.6.2.8 print\_window\_title() [1/2]

```
void print_window_title ( )
```

Here is the caller graph for this function:



#### 4.6.2.9 print\_window\_title() [2/2]

```
void print_window_title (
    WINDOW * win,
    int starty,
    int startx,
    int width,
    char * string,
    chtype color )
```



#### 4.6.2.10 terminate\_graphics()

```
void terminate_graphics ( )
```

Here is the caller graph for this function:



#### 4.6.2.11 update\_state\_graphic()

```
void update_state_graphic (
    unsigned char line,
    bool state )
```

### 4.6.3 Variable Documentation

#### 4.6.3.1 graph\_views

```
graph_view** graph_views
```

#### 4.6.3.2 print\_views

```
print_view** print_views
```

#### 4.6.3.3 ready\_to\_graph

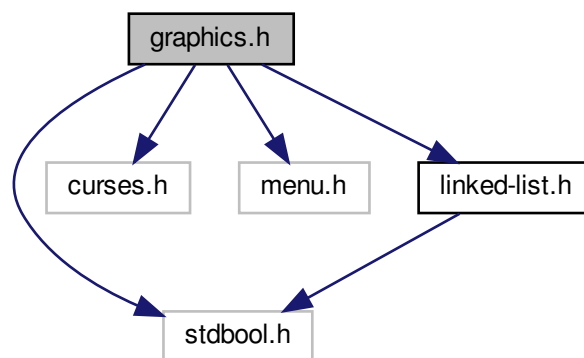
```
bool ready_to_graph = false
```

#### 4.6.3.4 setup\_views

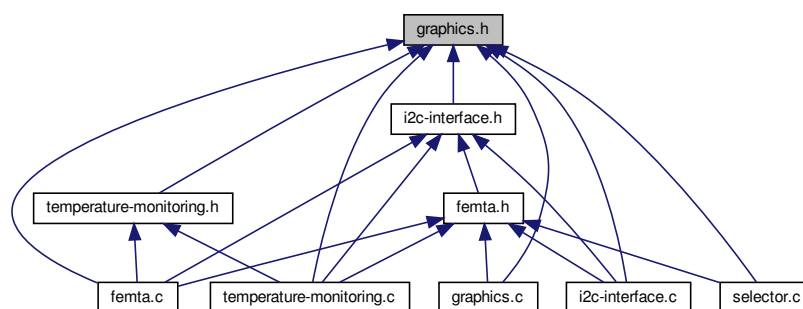
```
setup_view** setup_views
```

## 4.7 graphics.h File Reference

```
#include <stdbool.h>
#include <curses.h>
#include <menu.h>
#include "linked-list.h"
Include dependency graph for graphics.h:
```



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



## Classes

- struct [View](#)
- struct [Plot](#)
- struct [print\\_view](#)
- struct [graph\\_view](#)
- struct [setup\\_view](#)

## Macros

- `#define GENERAL_WINDOW 0`
- `#define CONTROL_WINDOW 1`
- `#define OPERATE_WINDOW 2`

## Typedefs

- `typedef struct View View`
- `typedef struct Plot Plot`
- `typedef struct print_view print_view`
- `typedef struct graph_view graph_view`
- `typedef struct setup_view setup_view`

## Functions

- `void initialize_graphics ()`
- `void terminate_graphics ()`
- `void print (unsigned char window_number, char *string, unsigned int color)`
- `void clear_print_window (unsigned char window_number)`
- `void erase_print_window (unsigned char window_number)`
- `void update_state_graphic (unsigned char line, bool state)`
- `void graph_plot (Plot *plot)`
- `void plot_add_value (Plot *plot, List *list, Node *node)`
- `Plot * create_plot (char *name, unsigned char number_of_lists)`

## Variables

- `unsigned char number_of_data_points_plottable`
- `Plot * graph_owner`
- `Plot ** all_possible_owners`
- `List * owner_index_list`
- `Node * graph_owner_index_node`

### 4.7.1 Macro Definition Documentation

#### 4.7.1.1 CONTROL\_WINDOW

```
#define CONTROL_WINDOW 1
```

#### 4.7.1.2 GENERAL\_WINDOW

```
#define GENERAL_WINDOW 0
```

#### 4.7.1.3 OPERATE\_WINDOW

```
#define OPERATE_WINDOW 2
```

### 4.7.2 Typedef Documentation

#### 4.7.2.1 graph\_view

```
typedef struct graph_view graph_view
```

#### 4.7.2.2 Plot

```
typedef struct Plot Plot
```

#### 4.7.2.3 print\_view

```
typedef struct print_view print_view
```

#### 4.7.2.4 setup\_view

```
typedef struct setup_view setup_view
```

#### 4.7.2.5 View

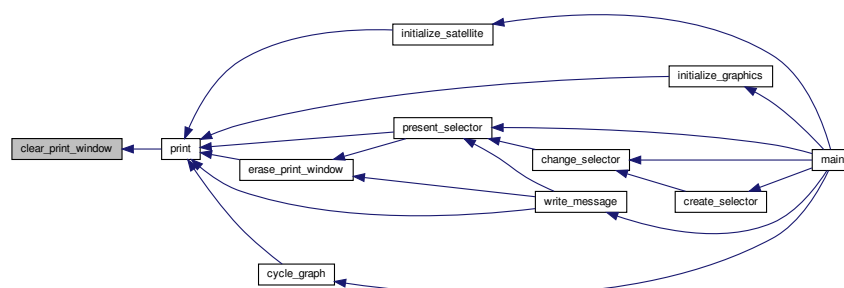
```
typedef struct View View
```

### 4.7.3 Function Documentation

## 4.7.3.1 clear\_print\_window()

```
void clear_print_window (
    unsigned char window_number )
```

Here is the caller graph for this function:



## 4.7.3.2 create\_plot()

```
Plot* create_plot (
    char * name,
    unsigned char number_of_lists )
```

Here is the call graph for this function:



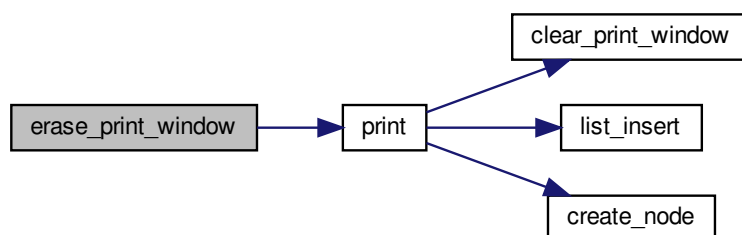
Here is the caller graph for this function:



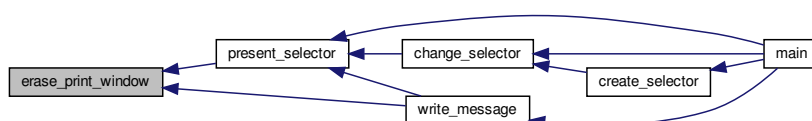
#### 4.7.3.3 erase\_print\_window()

```
void erase_print_window (
    unsigned char window_number )
```

Here is the call graph for this function:



Here is the caller graph for this function:



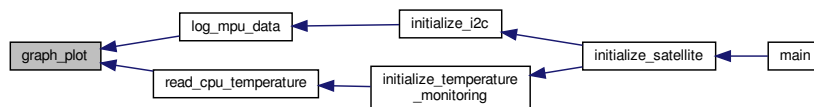
#### 4.7.3.4 graph\_plot()

```
void graph_plot (
    Plot * plot )
```

Here is the call graph for this function:



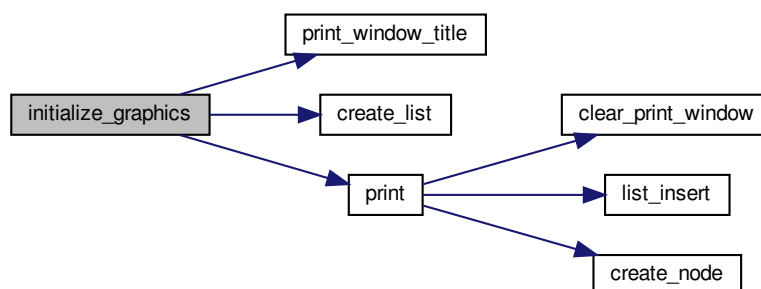
Here is the caller graph for this function:



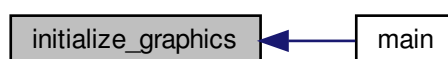
#### 4.7.3.5 initialize\_graphics()

```
void initialize_graphics ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



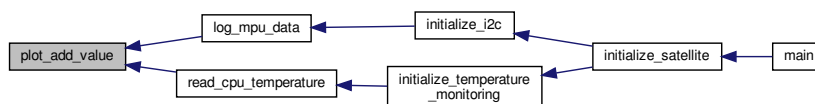
#### 4.7.3.6 plot\_add\_value()

```
void plot_add_value (
    Plot * plot,
    List * list,
    Node * node )
```

Here is the call graph for this function:



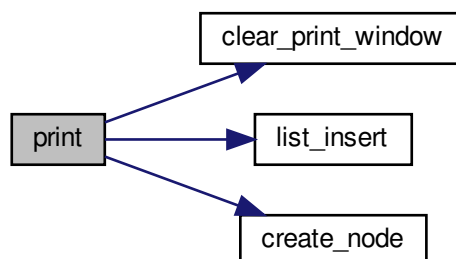
Here is the caller graph for this function:



#### 4.7.3.7 print()

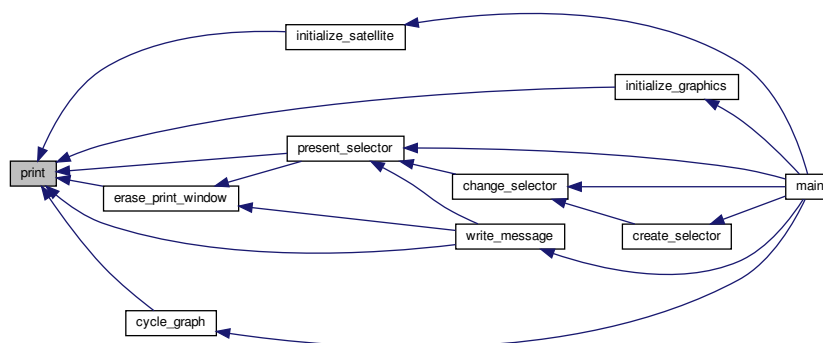
```
void print (
    unsigned char window_number,
    char * string,
    unsigned int color )
```

Here is the call graph for this function:





Here is the caller graph for this function:



#### 4.7.3.8 terminate\_graphics()

```
void terminate_graphics ( )
```

Here is the caller graph for this function:



#### 4.7.3.9 update\_state\_graphic()

```
void update_state_graphic (
    unsigned char line,
    bool state )
```

### 4.7.4 Variable Documentation

#### 4.7.4.1 all\_possible\_owners

```
Plot** all_possible_owners
```

#### 4.7.4.2 graph\_owner

```
Plot* graph_owner
```

#### 4.7.4.3 graph\_owner\_index\_node

```
Node* graph_owner_index_node
```

#### 4.7.4.4 number\_of\_data\_points\_plottable

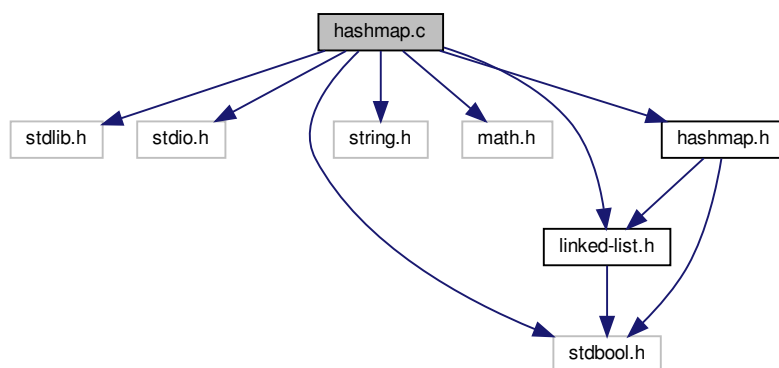
```
unsigned char number_of_data_points_plottable
```

#### 4.7.4.5 owner\_index\_list

```
List* owner_index_list
```

## 4.8 hashmap.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>
#include <string.h>
#include <math.h>
#include "linked-list.h"
#include "hashmap.h"
Include dependency graph for hashmap.c:
```



## Functions

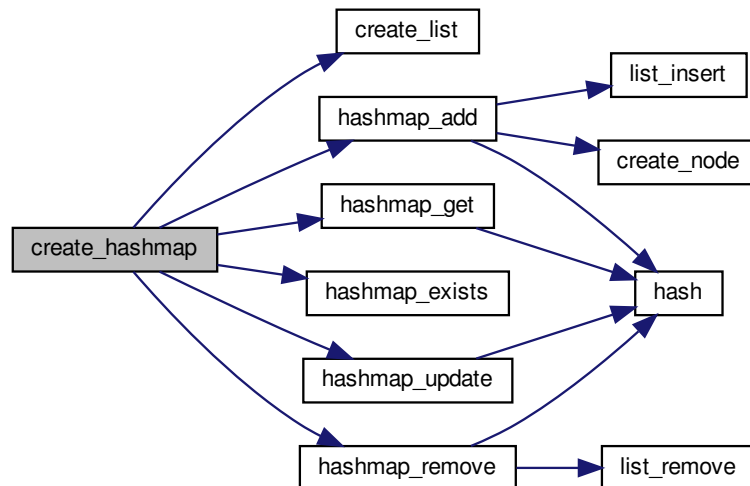
- void \* [hashmap\\_get](#) (Hashmap \*this, char \*string)
- void [hashmap\\_add](#) (Hashmap \*this, char \*string, void \*datum)
- void [hashmap\\_update](#) (Hashmap \*this, char \*string, void \*datum)
- bool [hashmap\\_exists](#) (Hashmap \*this, char \*string)
- void [hashmap\\_remove](#) (Hashmap \*this, char \*string)
- Hashmap \* [create\\_hashmap](#) (int expected\_size)
- int [hash](#) (char \*string, int upper\_bound)

### 4.8.1 Function Documentation

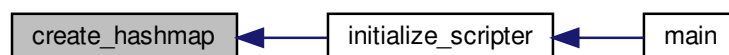
#### 4.8.1.1 create\_hashmap()

```
Hashmap* create_hashmap (
    int expected_size )
```

Here is the call graph for this function:



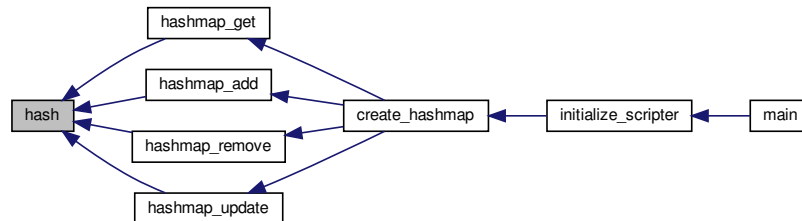
Here is the caller graph for this function:



#### 4.8.1.2 hash()

```
int hash (
    char * string,
    int upper_bound )
```

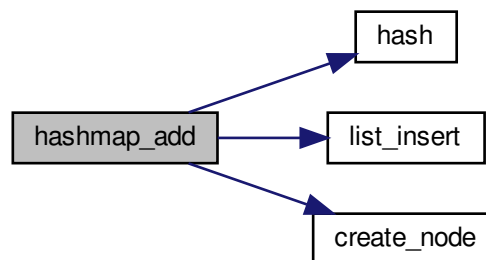
Here is the caller graph for this function:



#### 4.8.1.3 hashmap\_add()

```
void hashmap_add (
    Hashmap * this,
    char * string,
    void * datum )
```

Here is the call graph for this function:



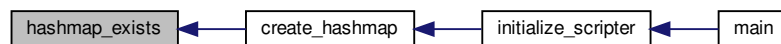
Here is the caller graph for this function:



#### 4.8.1.4 hashmap\_exists()

```
bool hashmap_exists (  
    Hashmap * this,  
    char * string )
```

Here is the caller graph for this function:



#### 4.8.1.5 hashmap\_get()

```
void * hashmap_get (  
    Hashmap * this,  
    char * string )
```

Here is the call graph for this function:



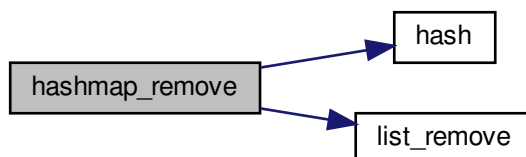
Here is the caller graph for this function:



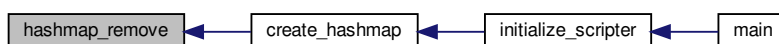
#### 4.8.1.6 hashmap\_remove()

```
void hashmap_remove (
    Hashmap * this,
    char * string )
```

Here is the call graph for this function:



Here is the caller graph for this function:



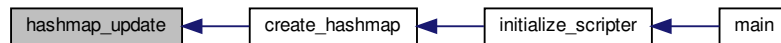
#### 4.8.1.7 hashmap\_update()

```
void hashmap_update (
    Hashmap * this,
    char * string,
    void * datum )
```

Here is the call graph for this function:



Here is the caller graph for this function:

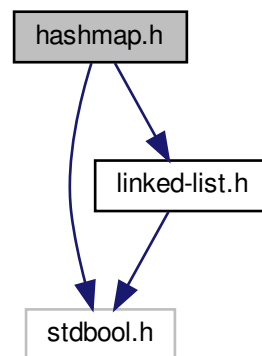


## 4.9 hashmap.h File Reference

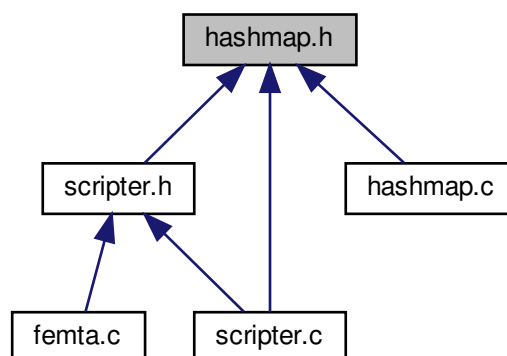
```
#include "stdbool.h"
```

```
#include "linked-list.h"
```

Include dependency graph for `hashmap.h`:



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



## Classes

- struct [HashmapElement](#)
- struct [Hashmap](#)

## Macros

- `#define` [HASHMAP\\_THRESHOLD](#) .6
- `#define` [HASHMAP\\_DEFAULT\\_SIZE](#) 64

## Typedefs

- `typedef struct` [HashmapElement](#) [HashmapElement](#)
- `typedef struct` [Hashmap](#) [Hashmap](#)

## Functions

- [Hashmap](#) \* [create\\_hashmap](#) (int expected\_size)
- int [hash](#) (char \*string, int upper\_bound)

### 4.9.1 Macro Definition Documentation

#### 4.9.1.1 HASHMAP\_DEFAULT\_SIZE

```
#define HASHMAP_DEFAULT_SIZE 64
```

#### 4.9.1.2 HASHMAP\_THRESHOLD

```
#define HASHMAP_THRESHOLD .6
```

### 4.9.2 Typedef Documentation

#### 4.9.2.1 Hashmap

```
typedef struct Hashmap Hashmap
```



#### 4.9.2.2 HashmapElement

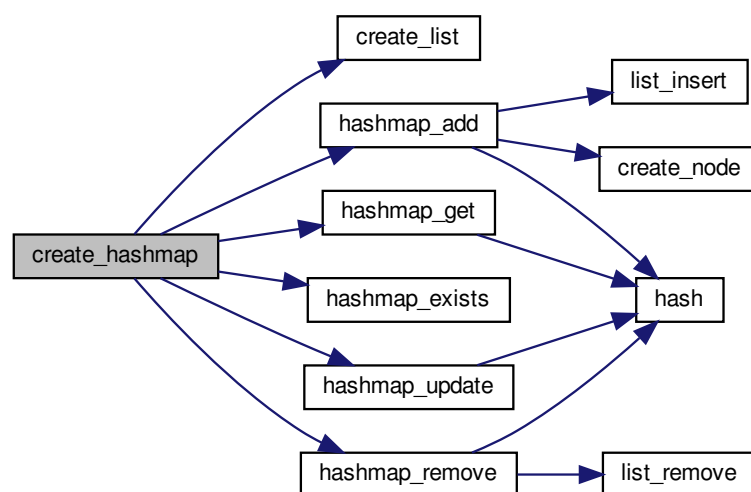
```
typedef struct HashmapElement HashmapElement
```

### 4.9.3 Function Documentation

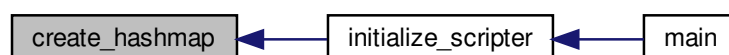
#### 4.9.3.1 create\_hashmap()

```
Hashmap* create_hashmap (  
    int expected_size )
```

Here is the call graph for this function:



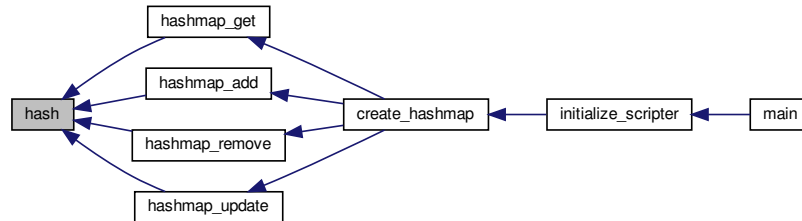
Here is the caller graph for this function:



#### 4.9.3.2 hash()

```
int hash (
    char * string,
    int upper_bound )
```

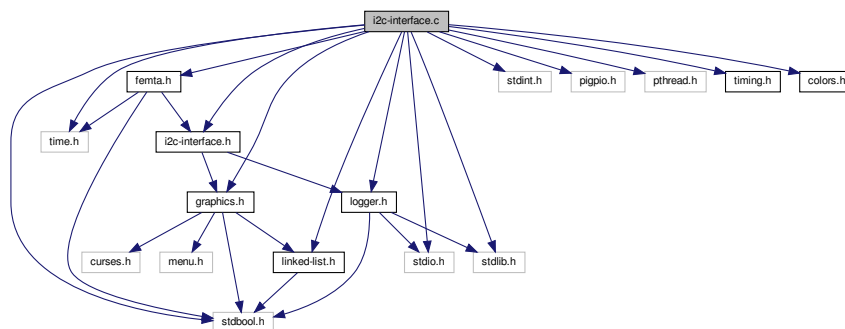
Here is the caller graph for this function:



## 4.10 i2c-interface.c File Reference

```
#include <time.h>
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <pigpio.h>
#include <pthread.h>
#include <stdbool.h>
#include "femta.h"
#include "i2c-interface.h"
#include "linked-list.h"
#include "graphics.h"
#include "timing.h"
#include "logger.h"
#include "colors.h"
```

Include dependency graph for `i2c-interface.c`:



## Macros

- `#define AK8963_ST1 0x02`
- `#define AK8963_XOUT_L 0x03`
- `#define AK8963_CNTL 0x0A`
- `#define AK8963_ADDRESS 0x0C`
- `#define AK8963_ASAX 0x10`
- `#define SMPLRT_DIV 0x19`
- `#define CONFIG 0x1A`
- `#define GYRO_CONFIG 0x1B`
- `#define ACCEL_CONFIG 0x1C`
- `#define ACCEL_CONFIG2 0x1D`
- `#define FIFO_EN 0x23`
- `#define I2C_MST_CTRL 0x24`
- `#define INT_PIN_CFG 0x37`
- `#define INT_ENABLE 0x38`
- `#define ACCEL_XOUT_H 0x3B`
- `#define TEMP_OUT_H 0x41`
- `#define TEMP_OUT_L 0x42`
- `#define GYRO_XOUT_H 0x43`
- `#define USER_CTRL 0x6A`
- `#define PWR_MGMT_1 0x6B`
- `#define PWR_MGMT_2 0x6C`
- `#define MPU9250_ADDRESS 0x68`
- `#define FIFO_COUNTH 0x72`
- `#define FIFO_R_W 0x74`
- `#define XA_OFFSET_H 0x77`
- `#define YA_OFFSET_H 0x7A`
- `#define ZA_OFFSET_H 0x7D`

## Enumerations

- `enum Ascale { AFS_2G = 0, AFS_4G, AFS_8G, AFS_16G }`
- `enum Gscale { GFS_250DPS = 0, GFS_500DPS, GFS_1000DPS, GFS_2000DPS }`
- `enum Mscale { MFS_14BITS = 0, MFS_16BITS }`

## Functions

- `void printBias (char *offset, char axis, float value)`
- `void printStartupConstants (char *offset)`
- `void readBytes (uint8_t address, uint8_t location, uint8_t number, uint8_t *data)`
- `float readTempData ()`
- `void readGyroData (float *axes)`
- `void readAccelData (float *axes)`
- `void readMagData (float *axes)`
- `void * log_mpu_data ()`
- `void initMPU9250 ()`
- `void resetMPU9250 ()`
- `void calibrateMPU9250 (float *dest1, float *dest2)`
- `void initAK8963 (float *destination)`
- `bool initialize_i2c (module *initialent)`
- `void terminate_mpu_logging ()`

## Variables

- FILE \* `mpu_log_file`
- char \* `mpu_log_file_name` = `"./logs/mpu-log.txt"`
- pthread\_t `mpu_thread`
- bool `mpu_termination_signal`
- int `mpu_values_read` = 0
- float `gyroBias` [3] = {0, 0, 0}
- float `accelBias` [3] = {0, 0, 0}
- float `magBias` [3] = {0, 0, 0}
- float `magScale` [3] = {1, 1, 1}
- float `magCalibration` [3] = {0, 0, 0}
- uint8\_t `Ascale` = `AFS_2G`
- uint8\_t `Gscale` = `GFS_250DPS`
- uint8\_t `Mscale` = `MFS_16BITS`
- uint8\_t `Mmode` = 0x02
- float `aRes` = 2.0 / 32768.0
- float `gRes` = 250.0 / 32768.0
- float `mRes` = 10. \* 4912. / 32760.0
- bool `newMagData` = false

## 4.10.1 Macro Definition Documentation

### 4.10.1.1 ACCEL\_CONFIG

```
#define ACCEL_CONFIG 0x1C
```

### 4.10.1.2 ACCEL\_CONFIG2

```
#define ACCEL_CONFIG2 0x1D
```

### 4.10.1.3 ACCEL\_XOUT\_H

```
#define ACCEL_XOUT_H 0x3B
```

### 4.10.1.4 AK8963\_ADDRESS

```
#define AK8963_ADDRESS 0x0C
```

#### 4.10.1.5 AK8963\_ASAX

```
#define AK8963_ASAX 0x10
```

#### 4.10.1.6 AK8963\_CNTL

```
#define AK8963_CNTL 0x0A
```

#### 4.10.1.7 AK8963\_ST1

```
#define AK8963_ST1 0x02
```

The following program is a C port of the code located at [https://github.com/kriswiner/MPU9250/blob/master/MPU9250\\_MS5637\\_AHRS\\_t3.ino](https://github.com/kriswiner/MPU9250/blob/master/MPU9250_MS5637_AHRS_t3.ino).

Alterations have been made by Noah Franks to integrate the file into the FEMTA Cubesat program. Additional code exists for specific use within FEMTA's project requirements, but many of the functions can be copied as they are over to future projects involving communication with the MPU 9250 over [I2C](#).

#### 4.10.1.8 AK8963\_XOUT\_L

```
#define AK8963_XOUT_L 0x03
```

#### 4.10.1.9 CONFIG

```
#define CONFIG 0x1A
```

#### 4.10.1.10 FIFO\_COUNTH

```
#define FIFO_COUNTH 0x72
```

#### 4.10.1.11 FIFO\_EN

```
#define FIFO_EN 0x23
```

#### 4.10.1.12 FIFO\_R\_W

```
#define FIFO_R_W 0x74
```

#### 4.10.1.13 GYRO\_CONFIG

```
#define GYRO_CONFIG 0x1B
```

#### 4.10.1.14 GYRO\_XOUT\_H

```
#define GYRO_XOUT_H 0x43
```

#### 4.10.1.15 I2C\_MST\_CTRL

```
#define I2C_MST_CTRL 0x24
```

#### 4.10.1.16 INT\_ENABLE

```
#define INT_ENABLE 0x38
```

#### 4.10.1.17 INT\_PIN\_CFG

```
#define INT_PIN_CFG 0x37
```

#### 4.10.1.18 MPU9250\_ADDRESS

```
#define MPU9250_ADDRESS 0x68
```

#### 4.10.1.19 PWR\_MGMT\_1

```
#define PWR_MGMT_1 0x6B
```

#### 4.10.1.20 PWR\_MGMT\_2

```
#define PWR_MGMT_2 0x6C
```

#### 4.10.1.21 SMPLRT\_DIV

```
#define SMPLRT_DIV 0x19
```

#### 4.10.1.22 TEMP\_OUT\_H

```
#define TEMP_OUT_H 0x41
```

#### 4.10.1.23 TEMP\_OUT\_L

```
#define TEMP_OUT_L 0x42
```

#### 4.10.1.24 USER\_CTRL

```
#define USER_CTRL 0x6A
```

#### 4.10.1.25 XA\_OFFSET\_H

```
#define XA_OFFSET_H 0x77
```

#### 4.10.1.26 YA\_OFFSET\_H

```
#define YA_OFFSET_H 0x7A
```

#### 4.10.1.27 ZA\_OFFSET\_H

```
#define ZA_OFFSET_H 0x7D
```

### 4.10.2 Enumeration Type Documentation

#### 4.10.2.1 Ascale

```
enum Ascale
```

**Enumerator**

|         |  |
|---------|--|
| AFS_2G  |  |
| AFS_4G  |  |
| AFS_8G  |  |
| AFS_16G |  |

**4.10.2.2 Gscale**

enum [Gscale](#)

**Enumerator**

|             |  |
|-------------|--|
| GFS_250DPS  |  |
| GFS_500DPS  |  |
| GFS_1000DPS |  |
| GFS_2000DPS |  |

**4.10.2.3 Mscale**

enum [Mscale](#)

**Enumerator**

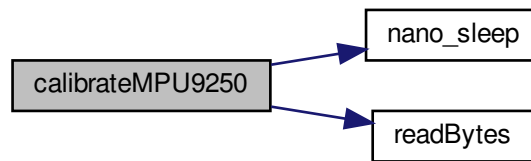
|            |  |
|------------|--|
| MFS_14BITS |  |
| MFS_16BITS |  |

**4.10.3 Function Documentation****4.10.3.1 calibrateMPU9250()**

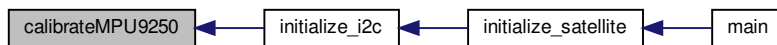
```
void calibrateMPU9250 (
    float * dest1,
    float * dest2 )
```



Here is the call graph for this function:



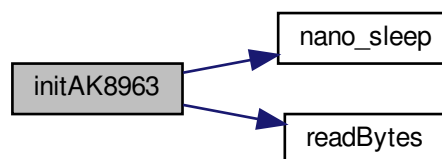
Here is the caller graph for this function:



#### 4.10.3.2 `initAK8963()`

```
void initAK8963 (  
    float * destination )
```

Here is the call graph for this function:



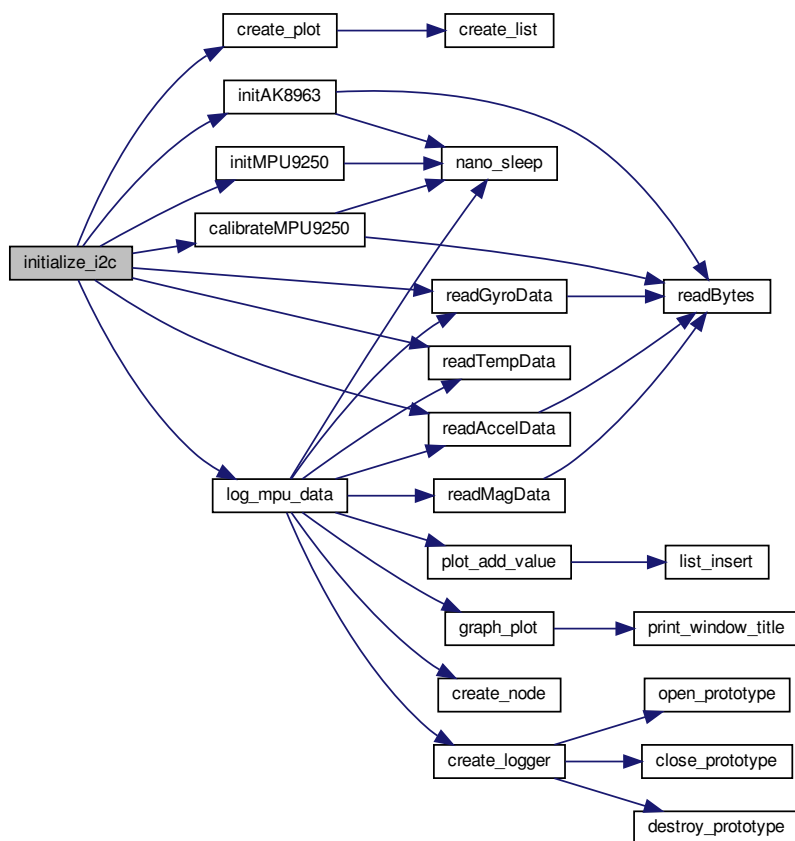
Here is the caller graph for this function:



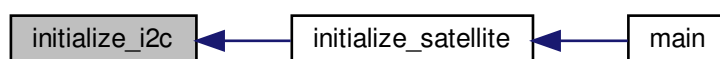
#### 4.10.3.3 initialize\_i2c()

```
bool initialize_i2c (
    module * initialent )
```

Here is the call graph for this function:



Here is the caller graph for this function:



## 4.10.3.4 initMPU9250()

```
void initMPU9250 ( )
```

?!?!?

?!?!? Here is the call graph for this function:



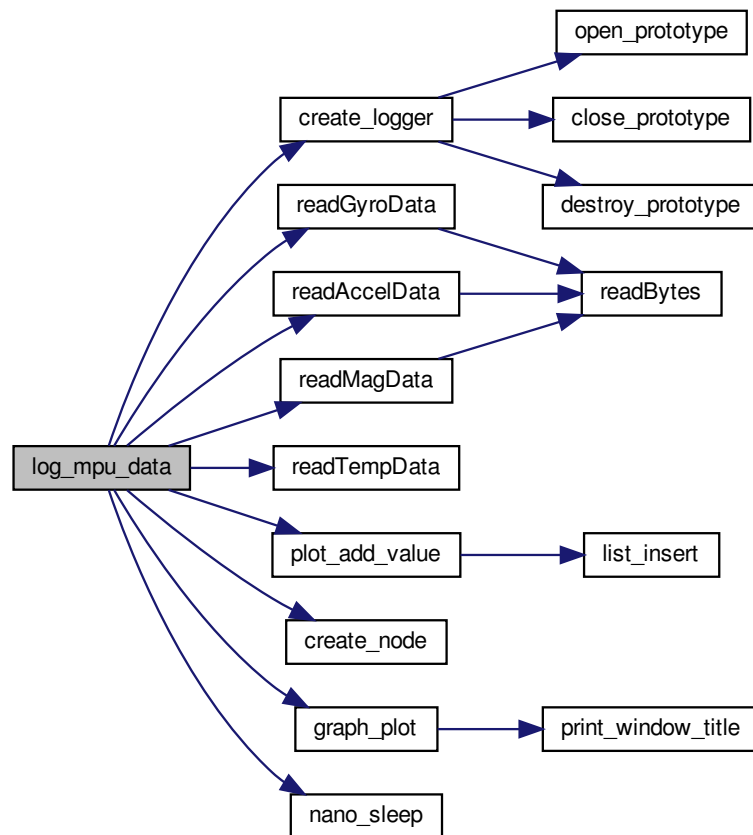
Here is the caller graph for this function:



## 4.10.3.5 log\_mpu\_data()

```
void* log_mpu_data ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 4.10.3.6 printBias()

```

void printBias (
    char * offset,
    char axis,
    float value )
  
```

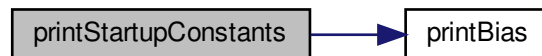
Here is the caller graph for this function:



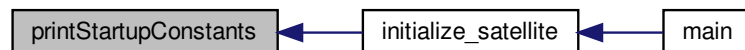
#### 4.10.3.7 printStartupConstants()

```
void printStartupConstants (  
    char * offset )
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 4.10.3.8 readAccelData()

```
void readAccelData (  
    float * axes )
```

Here is the call graph for this function:



Here is the caller graph for this function:

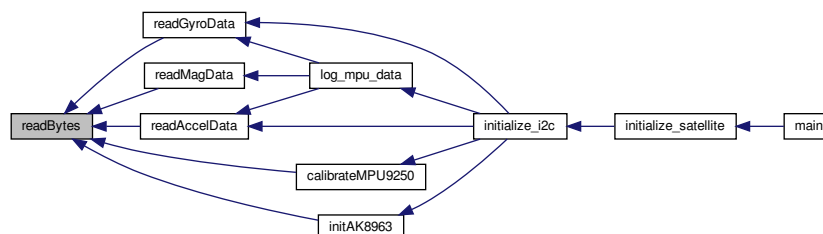


#### 4.10.3.9 readBytes()

```

void readBytes (
    uint8_t address,
    uint8_t location,
    uint8_t number,
    uint8_t * data )
  
```

Here is the caller graph for this function:



## 4.10.3.10 readGyroData()

```
void readGyroData (
    float * axes )
```

Here is the call graph for this function:



Here is the caller graph for this function:



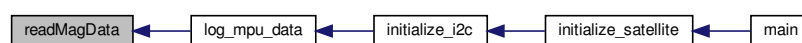
## 4.10.3.11 readMagData()

```
void readMagData (
    float * axes )
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 4.10.3.12 readTempData()

```
float readTempData ( )
```

Here is the caller graph for this function:



#### 4.10.3.13 resetMPU9250()

```
void resetMPU9250 ( )
```

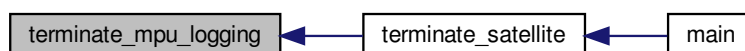
Here is the call graph for this function:



#### 4.10.3.14 terminate\_mpu\_logging()

```
void terminate_mpu_logging ( )
```

Here is the caller graph for this function:





## 4.10.4 Variable Documentation

### 4.10.4.1 accelBias

```
float accelBias[3] = {0, 0, 0}
```

### 4.10.4.2 aRes

```
float aRes = 2.0 / 32768.0
```

### 4.10.4.3 Ascale

```
uint8_t Ascale = AFS_2G
```

### 4.10.4.4 gRes

```
float gRes = 250.0 / 32768.0
```

### 4.10.4.5 Gscale

```
uint8_t Gscale = GFS_250DPS
```

### 4.10.4.6 gyroBias

```
float gyroBias[3] = {0, 0, 0}
```

### 4.10.4.7 magBias

```
float magBias[3] = {0, 0, 0}
```

#### 4.10.4.8 magCalibration

```
float magCalibration[3] = {0, 0, 0}
```

#### 4.10.4.9 magScale

```
float magScale[3] = {1, 1, 1}
```

#### 4.10.4.10 Mmode

```
uint8_t Mmode = 0x02
```

#### 4.10.4.11 mpu\_log\_file

```
FILE* mpu_log_file
```

#### 4.10.4.12 mpu\_log\_file\_name

```
char* mpu_log_file_name = "./logs/mpu-log.txt"
```

#### 4.10.4.13 mpu\_termination\_signal

```
bool mpu_termination_signal
```

#### 4.10.4.14 mpu\_thread

```
pthread_t mpu_thread
```

#### 4.10.4.15 mpu\_values\_read

```
int mpu_values_read = 0
```

## 4.10.4.16 mRes

```
float mRes = 10. * 4912. / 32760.0
```

## 4.10.4.17 Mscale

```
uint8_t Mscale = MFS_16BITS
```

## 4.10.4.18 newMagData

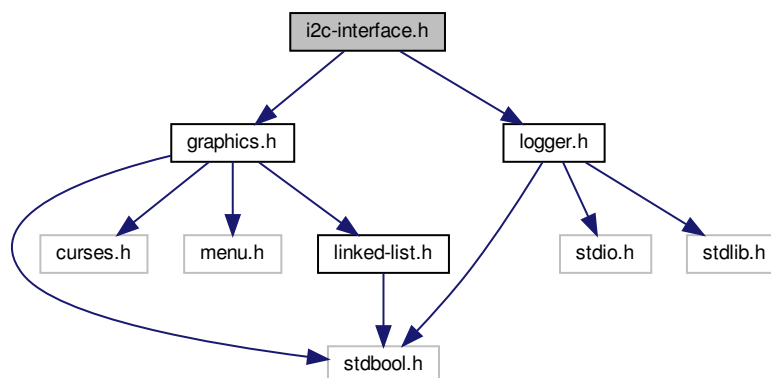
```
bool newMagData = false
```

## 4.11 i2c-interface.h File Reference

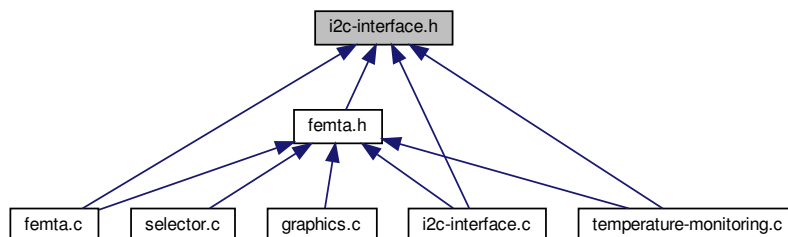
```
#include "graphics.h"
```

```
#include "logger.h"
```

Include dependency graph for i2c-interface.h:



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



## Classes

- struct [I2C](#)

## Typedefs

- typedef struct [I2C](#) [I2C](#)
- typedef struct [module](#) [module](#)

## Functions

- bool [initialize\\_i2c](#) ([module](#) \*initialent)
- void [printStartupConstants](#) (char \*offset)
- void [terminate\\_mpu\\_logging](#) ()

## Variables

- [module](#) \* [i2c\\_device](#)
- [Plot](#) \* [mpu\\_gyro\\_plot](#)
- [Plot](#) \* [mpu\\_acel\\_plot](#)
- [Plot](#) \* [mpu\\_magn\\_plot](#)
- [Logger](#) \* [mpu\\_logger](#)

### 4.11.1 Typedef Documentation

#### 4.11.1.1 I2C

```
typedef struct I2C I2C
```

#### 4.11.1.2 module

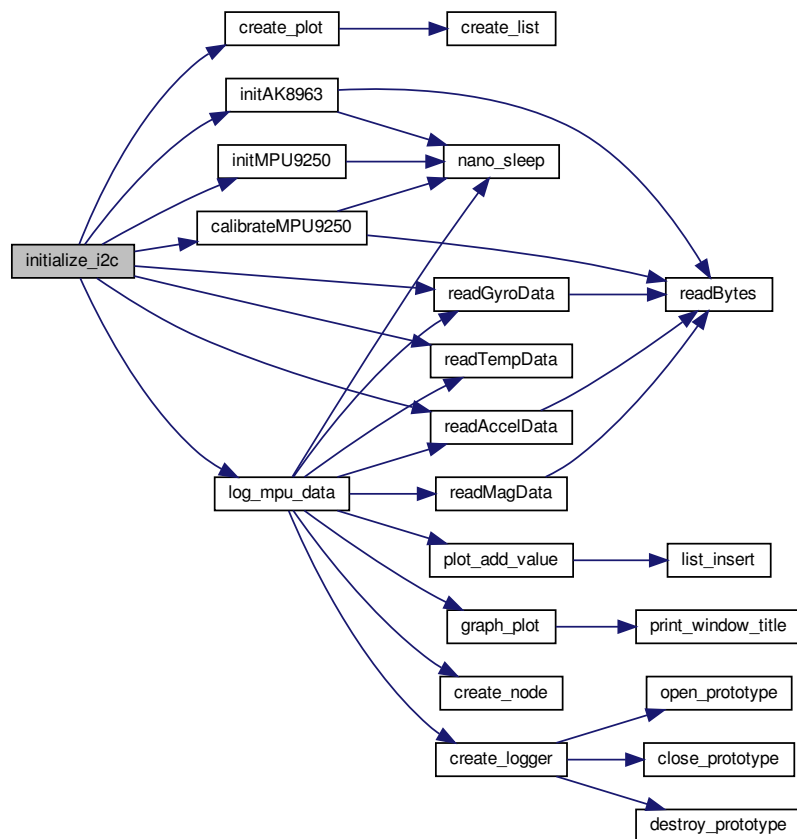
```
typedef struct module module
```

### 4.11.2 Function Documentation

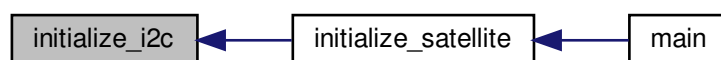
## 4.11.2.1 initialize\_i2c()

```
bool initialize_i2c (
    module * initialent )
```

Here is the call graph for this function:



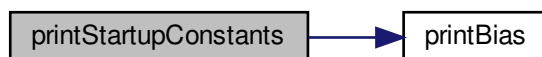
Here is the caller graph for this function:



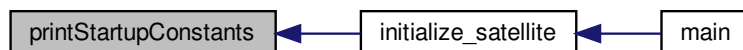
#### 4.11.2.2 printStartupConstants()

```
void printStartupConstants (
    char * offset )
```

Here is the call graph for this function:



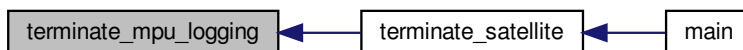
Here is the caller graph for this function:



#### 4.11.2.3 terminate\_mpu\_logging()

```
void terminate_mpu_logging ( )
```

Here is the caller graph for this function:



### 4.11.3 Variable Documentation

## 4.11.3.1 i2c\_device

```
module* i2c_device
```

## 4.11.3.2 mpu\_acel\_plot

```
Plot* mpu_acel_plot
```

## 4.11.3.3 mpu\_gyro\_plot

```
Plot* mpu_gyro_plot
```

## 4.11.3.4 mpu\_logger

```
Logger* mpu_logger
```

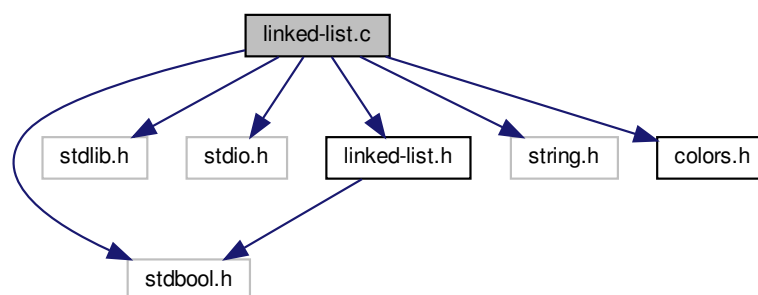
## 4.11.3.5 mpu\_magn\_plot

```
Plot* mpu_magn_plot
```

## 4.12 linked-list.c File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <stdio.h>
#include "linked-list.h"
#include "string.h"
#include "colors.h"
```

Include dependency graph for linked-list.c:



## Functions

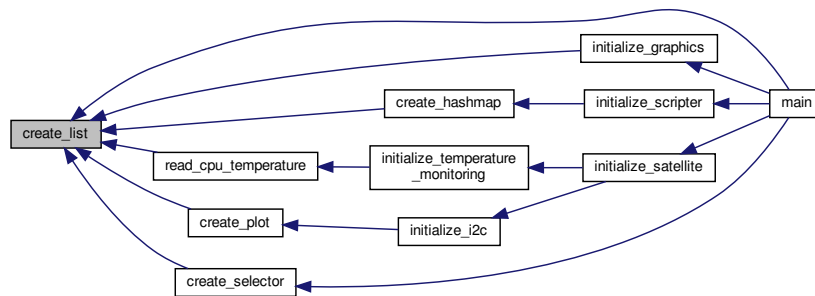
- `Node * create_node (void *value)`
- `List * create_list (unsigned int limit, bool doubly_linked)`
- `void list_insert (List *list, Node *node)`
- `void list_remove (List *list, Node *node)`

### 4.12.1 Function Documentation

#### 4.12.1.1 create\_list()

```
List* create_list (
    unsigned int limit,
    bool doubly_linked )
```

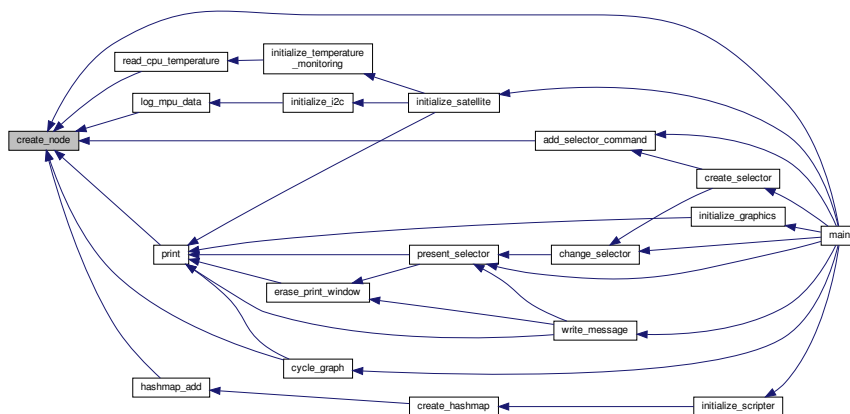
Here is the caller graph for this function:



#### 4.12.1.2 create\_node()

```
Node* create_node (
    void * value )
```

Here is the caller graph for this function:

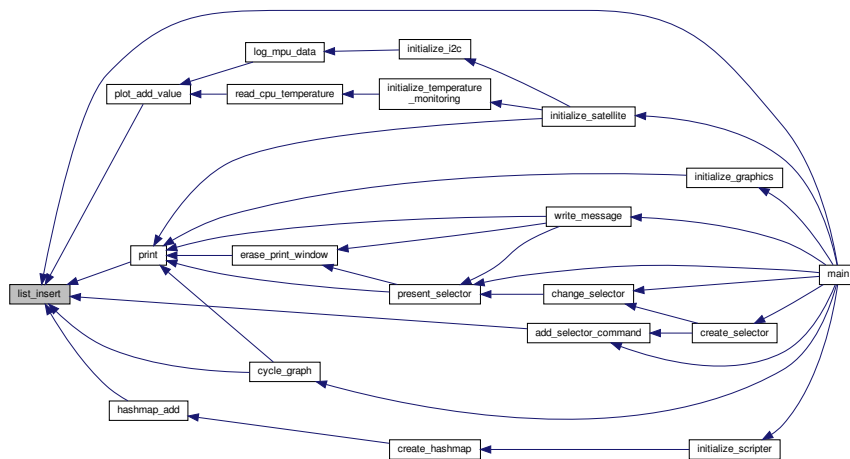




## 4.12.1.3 list\_insert()

```
void list_insert (
    List * list,
    Node * node )
```

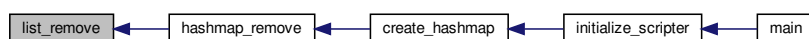
Here is the caller graph for this function:



## 4.12.1.4 list\_remove()

```
void list_remove (
    List * list,
    Node * node )
```

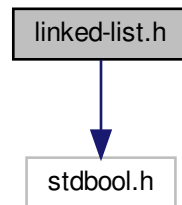
Here is the caller graph for this function:



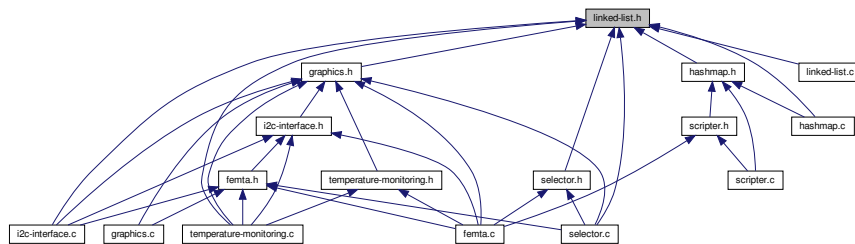
## 4.13 linked-list.h File Reference

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

Include dependency graph for linked-list.h:



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



### Classes

- struct [Node](#)
- struct [List](#)

### Macros

- `#define` [INTEGER\\_NODE](#) 0
- `#define` [FLOAT\\_NODE](#) 1
- `#define` [STRING\\_NODE](#) 2

### Typedefs

- `typedef struct` [Node](#) [Node](#)
- `typedef struct` [List](#) [List](#)

## Functions

- [Node](#) \* [create\\_node](#) (void \*value)
- [List](#) \* [create\\_list](#) (unsigned int limit, bool doublely\_linked)
- void [list\\_insert](#) ([List](#) \*list, [Node](#) \*node)
- void [list\\_remove](#) ([List](#) \*list, [Node](#) \*node)

### 4.13.1 Macro Definition Documentation

#### 4.13.1.1 FLOAT\_NODE

```
#define FLOAT_NODE 1
```

#### 4.13.1.2 INTEGER\_NODE

```
#define INTEGER_NODE 0
```

#### 4.13.1.3 STRING\_NODE

```
#define STRING_NODE 2
```

### 4.13.2 Typedef Documentation

#### 4.13.2.1 List

```
typedef struct List List
```

#### 4.13.2.2 Node

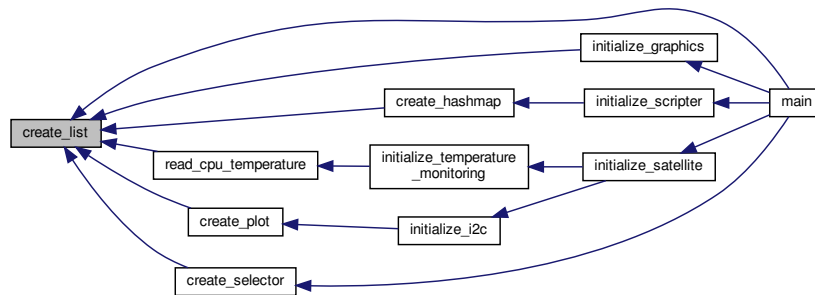
```
typedef struct Node Node
```

### 4.13.3 Function Documentation

#### 4.13.3.1 create\_list()

```
List* create_list (
    unsigned int limit,
    bool doublely_linked )
```

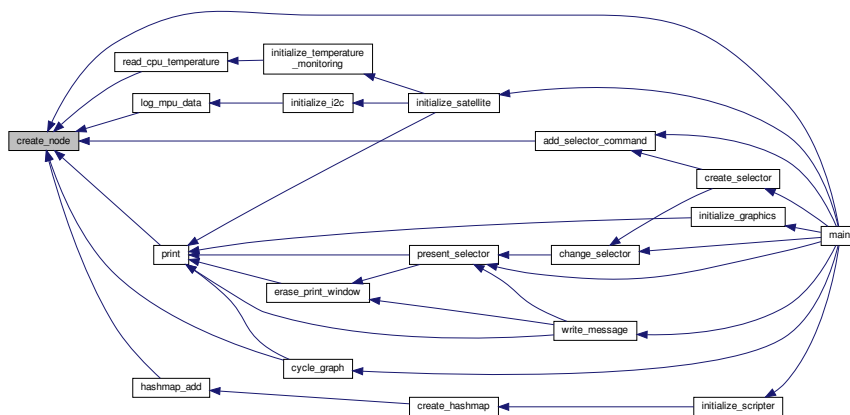
Here is the caller graph for this function:



#### 4.13.3.2 create\_node()

```
Node* create_node (
    void * value )
```

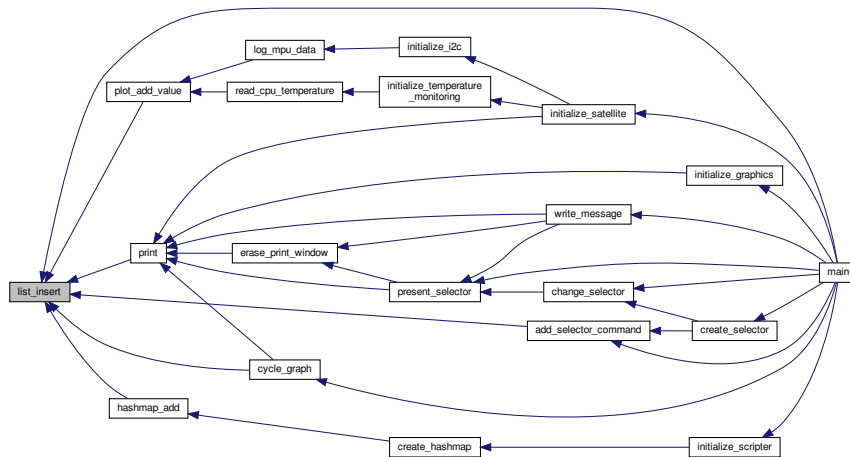
Here is the caller graph for this function:



## 4.13.3.3 list\_insert()

```
void list_insert (
    List * list,
    Node * node )
```

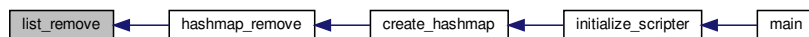
Here is the caller graph for this function:



## 4.13.3.4 list\_remove()

```
void list_remove (
    List * list,
    Node * node )
```

Here is the caller graph for this function:

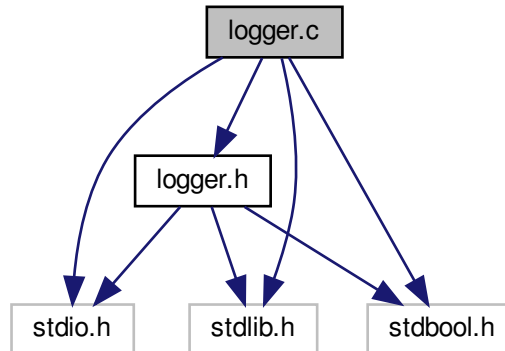


## 4.14 logger.c File Reference

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

```
#include "logger.h"
```

Include dependency graph for logger.c:



## Functions

- bool `open_prototype (Logger *self)`
- bool `close_prototype (Logger *self)`
- void `destroy_prototype (Logger *self)`
- `Logger *` `create_logger (char *filename)`

### 4.14.1 Function Documentation

#### 4.14.1.1 close\_prototype()

```
bool close_prototype (
    Logger * self )
```

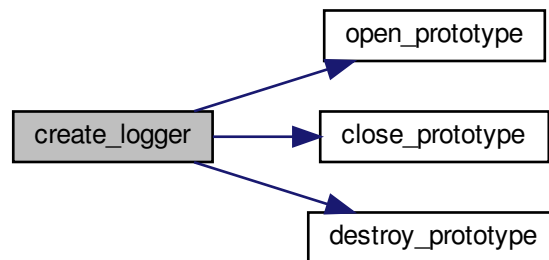
Here is the caller graph for this function:



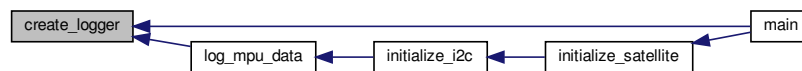
## 4.14.1.2 create\_logger()

```
Logger* create_logger (  
    char * filename )
```

Here is the call graph for this function:



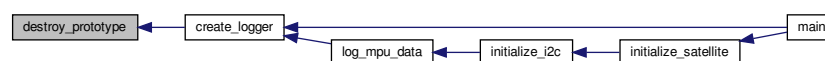
Here is the caller graph for this function:



## 4.14.1.3 destroy\_prototype()

```
void destroy_prototype (  
    Logger * self )
```

Here is the caller graph for this function:



#### 4.14.1.4 open\_prototype()

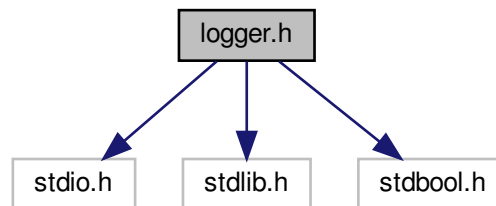
```
bool open_prototype (
    Logger * self )
```

Here is the caller graph for this function:

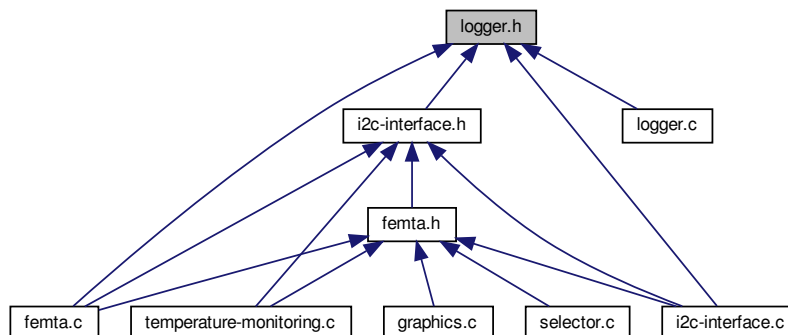


### 4.15 logger.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
Include dependency graph for logger.h:
```



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





## Classes

- struct [Logger](#)

## Typedefs

- typedef struct [Logger](#) [Logger](#)

## Functions

- [Logger](#) \* [create\\_logger](#) (char \*log\_file\_name)

### 4.15.1 Typedef Documentation

#### 4.15.1.1 Logger

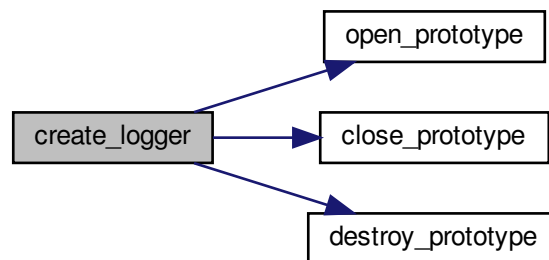
```
typedef struct Logger Logger
```

### 4.15.2 Function Documentation

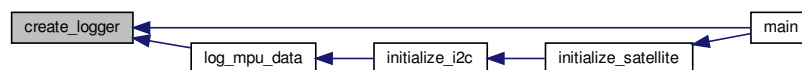
#### 4.15.2.1 create\_logger()

```
Logger* create_logger (  
    char * log_file_name )
```

Here is the call graph for this function:



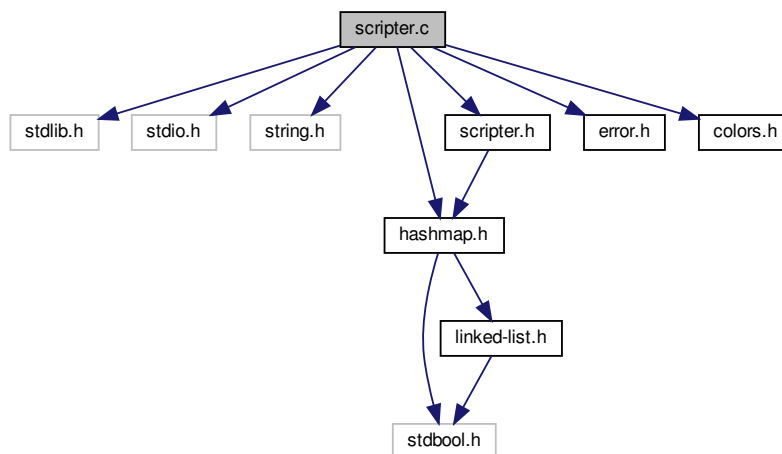
Here is the caller graph for this function:



## 4.16 scripter.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "hashmap.h"
#include "scripter.h"
#include "error.h"
#include "colors.h"
```

Include dependency graph for scripter.c:



### Functions

- void [initialize\\_scripter](#) ()
- void [define\\_script\\_action](#) (char \*symbol, [lambda](#) action)
- void [execute\\_script](#) (char \*filename)

#### 4.16.1 Function Documentation

##### 4.16.1.1 [define\\_script\\_action](#)()

```
void define_script_action (
    char * symbol,
    lambda action )
```

## 4.16.1.2 execute\_script()

```
void execute_script (
    char * filename )
```

Here is the call graph for this function:



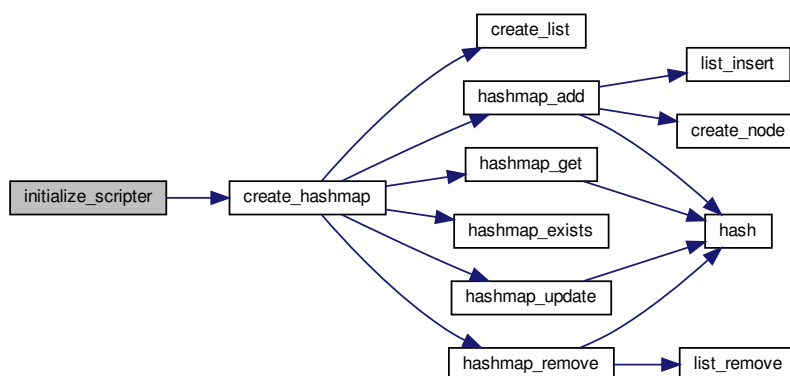
Here is the caller graph for this function:



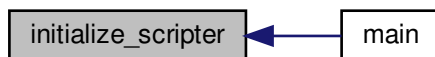
## 4.16.1.3 initialize\_scripter()

```
void initialize_scripter ( )
```

Here is the call graph for this function:



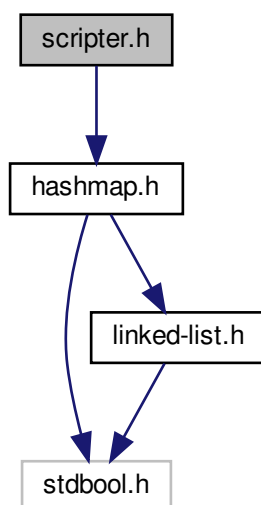
Here is the caller graph for this function:



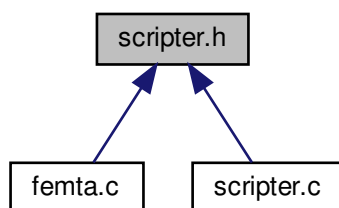
## 4.17 scripter.h File Reference

```
#include "hashmap.h"
```

Include dependency graph for `scripter.h`:



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



## Typedefs

- typedef void(\* [lambda](#)) (void \*)

## Functions

- void [initialize\\_scripter](#) ()
- void [define\\_script\\_action](#) (char \*symbol, [lambda](#) action)
- void [execute\\_script](#) (char \*filename)

## Variables

- [HashMap](#) \* [action\\_table](#)

### 4.17.1 Typedef Documentation

#### 4.17.1.1 [lambda](#)

```
typedef void(* lambda) (void *)
```

### 4.17.2 Function Documentation

#### 4.17.2.1 [define\\_script\\_action\(\)](#)

```
void define_script_action (  
    char * symbol,  
    lambda action )
```

#### 4.17.2.2 execute\_script()

```
void execute_script (
    char * filename )
```

Here is the call graph for this function:



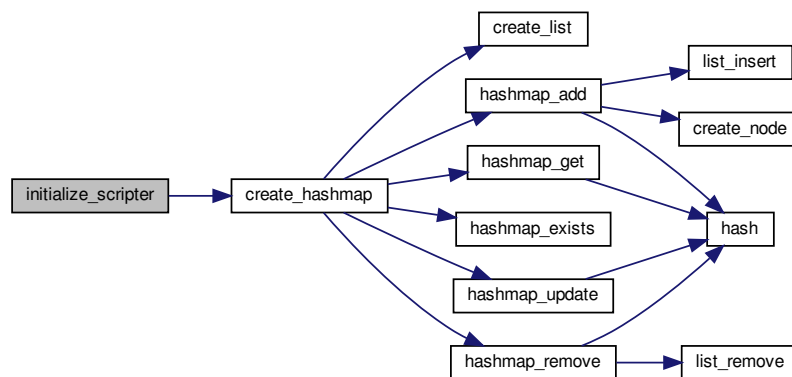
Here is the caller graph for this function:



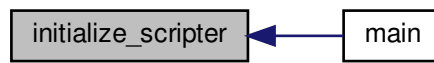
#### 4.17.2.3 initialize\_scripter()

```
void initialize_scripter ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 4.17.3 Variable Documentation

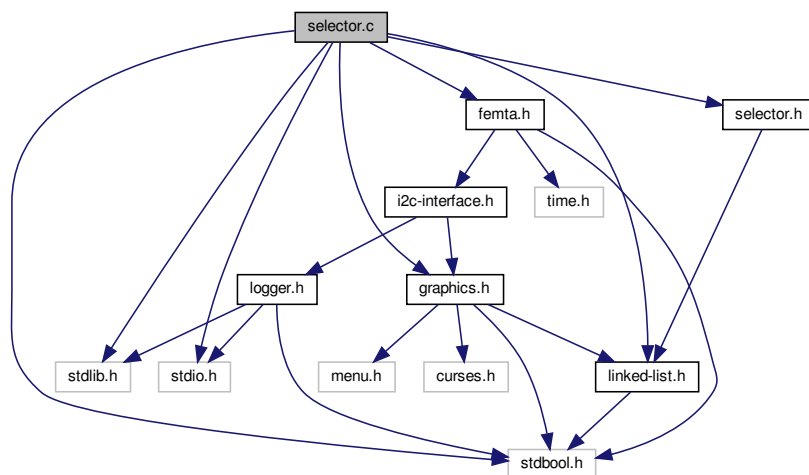
#### 4.17.3.1 action\_table

Hashmap\* action\_table

## 4.18 selector.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>
#include "linked-list.h"
#include "selector.h"
#include "graphics.h"
#include "femta.h"
```

Include dependency graph for selector.c:



## Functions

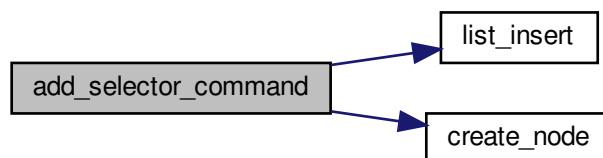
- void `change_selector` (void \*selector)
- `Selector` \* `create_selector` (`Selector` \*parent)
- void `add_selector_command` (`Selector` \*selector, char key, char \*text, `lambda` action, void \*argument)
- void `execute_selector` (`Selector` \*selector, char key)
- void `present_selector` (`Selector` \*selector)
- void `flip_bool` (void \*pointer)
- void `cycle_graph` (void \*nil)
- void `flip_femta` (void \*number)
- void `flip_valve` (void \*nil)
- void `rotate` (void \*nil)
- void `write_message` (void \*logger)

### 4.18.1 Function Documentation

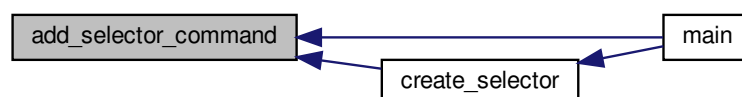
#### 4.18.1.1 `add_selector_command()`

```
void add_selector_command (
    Selector * selector,
    char key,
    char * text,
    lambda action,
    void * argument )
```

Here is the call graph for this function:



Here is the caller graph for this function:

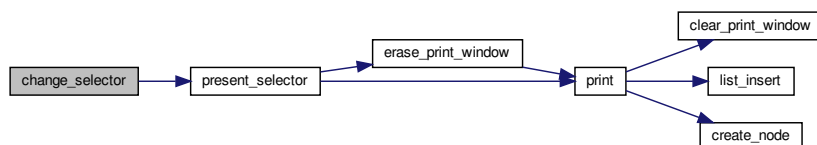




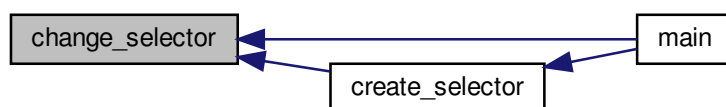
## 4.18.1.2 change\_selector()

```
void change_selector (
    void * selector )
```

Here is the call graph for this function:



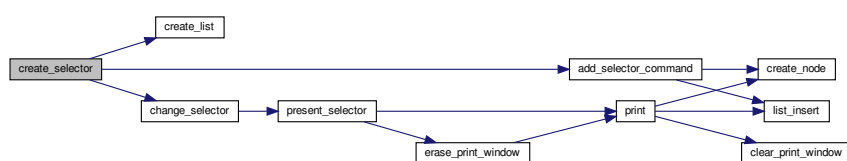
Here is the caller graph for this function:



## 4.18.1.3 create\_selector()

```
Selector* create_selector (
    Selector * parent )
```

Here is the call graph for this function:



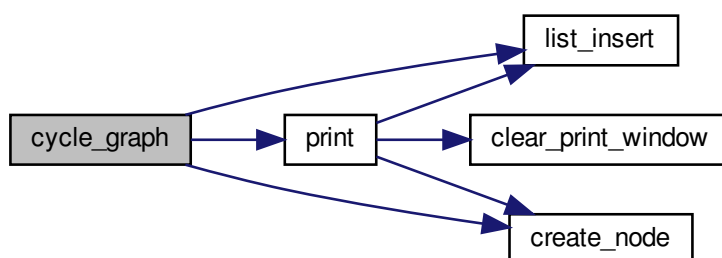
Here is the caller graph for this function:



#### 4.18.1.4 cycle\_graph()

```
void cycle_graph (  
    void * nil )
```

Here is the call graph for this function:



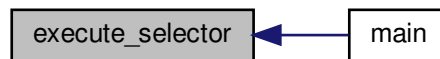
Here is the caller graph for this function:



## 4.18.1.5 execute\_selector()

```
void execute_selector (
    Selector * selector,
    char key )
```

Here is the caller graph for this function:



## 4.18.1.6 flip\_bool()

```
void flip_bool (
    void * pointer )
```

Here is the caller graph for this function:



## 4.18.1.7 flip\_femta()

```
void flip_femta (
    void * number )
```

Here is the caller graph for this function:



#### 4.18.1.8 flip\_valve()

```
void flip_valve (
    void * nil )
```

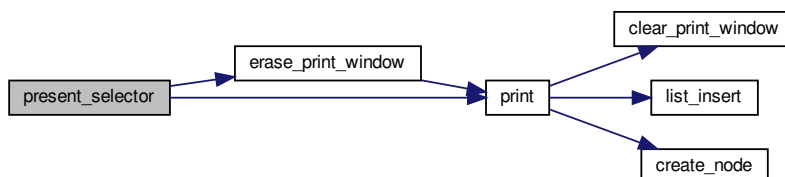
Here is the caller graph for this function:



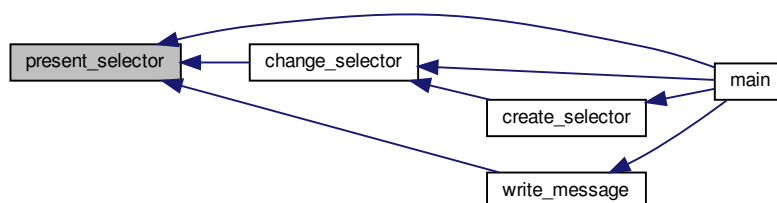
#### 4.18.1.9 present\_selector()

```
void present_selector (
    Selector * selector )
```

Here is the call graph for this function:



Here is the caller graph for this function:



## 4.18.1.10 rotate()

```
void rotate (  
    void * nil )
```

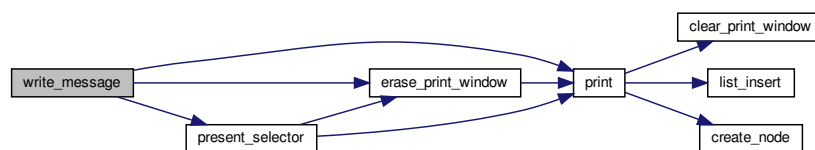
Here is the caller graph for this function:



## 4.18.1.11 write\_message()

```
void write_message (  
    void * logger )
```

Here is the call graph for this function:



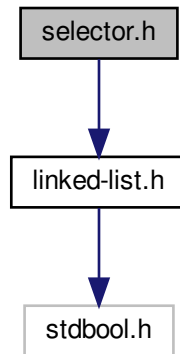
Here is the caller graph for this function:



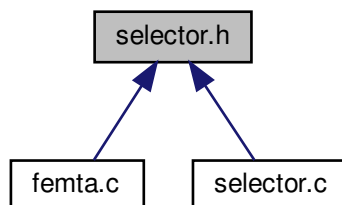
## 4.19 selector.h File Reference

```
#include "linked-list.h"
```

Include dependency graph for selector.h:



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



### Classes

- struct [Command](#)
- struct [Selector](#)

### Typedefs

- typedef void(\* [lambda](#)) (void \*)
- typedef struct [Command](#) [Command](#)
- typedef struct [Selector](#) [Selector](#)

## Functions

- [Selector](#) \* [create\\_selector](#) ()
- void [add\\_selector\\_command](#) ([Selector](#) \*selector, char key, char \*text, [lambda](#) action, void \*argument)
- void [execute\\_selector](#) ([Selector](#) \*selector, char key)
- void [present\\_selector](#) ([Selector](#) \*selector)
- void [change\\_selector](#) (void \*selector)
- void [flip\\_bool](#) (void \*pointer)
- void [cycle\\_graph](#) (void \*nil)
- void [flip\\_femta](#) (void \*number)
- void [flip\\_valve](#) (void \*nil)
- void [rotate](#) (void \*nil)
- void [write\\_message](#) (void \*nil)

## Variables

- [Selector](#) \* [visible\\_selector](#)

### 4.19.1 Typedef Documentation

#### 4.19.1.1 Command

```
typedef struct Command Command
```

#### 4.19.1.2 lambda

```
typedef void(* lambda) (void *)
```

#### 4.19.1.3 Selector

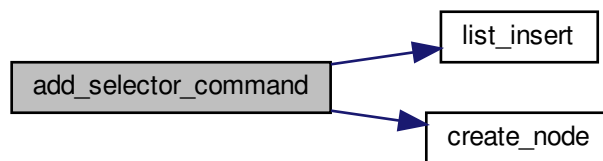
```
typedef struct Selector Selector
```

### 4.19.2 Function Documentation

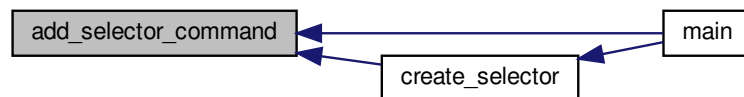
#### 4.19.2.1 add\_selector\_command()

```
void add_selector_command (  
    Selector * selector,  
    char key,  
    char * text,  
    lambda action,  
    void * argument )
```

Here is the call graph for this function:



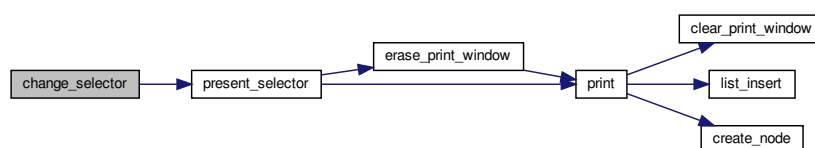
Here is the caller graph for this function:



#### 4.19.2.2 change\_selector()

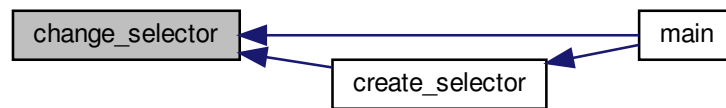
```
void change_selector (  
    void * selector )
```

Here is the call graph for this function:





Here is the caller graph for this function:



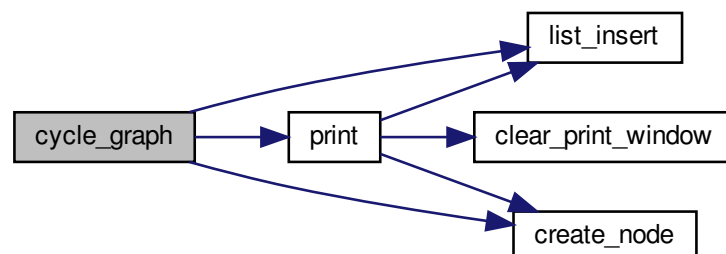
#### 4.19.2.3 `create_selector()`

```
Selector* create_selector ( )
```

#### 4.19.2.4 `cycle_graph()`

```
void cycle_graph (
    void * nil )
```

Here is the call graph for this function:



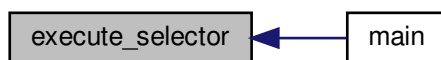
Here is the caller graph for this function:



#### 4.19.2.5 execute\_selector()

```
void execute_selector (
    Selector * selector,
    char key )
```

Here is the caller graph for this function:



#### 4.19.2.6 flip\_bool()

```
void flip_bool (
    void * pointer )
```

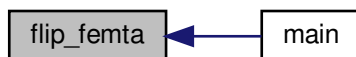
Here is the caller graph for this function:



#### 4.19.2.7 flip\_femta()

```
void flip_femta (
    void * number )
```

Here is the caller graph for this function:



## 4.19.2.8 flip\_valve()

```
void flip_valve (  
    void * nil )
```

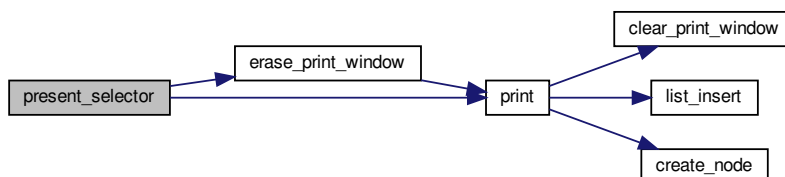
Here is the caller graph for this function:



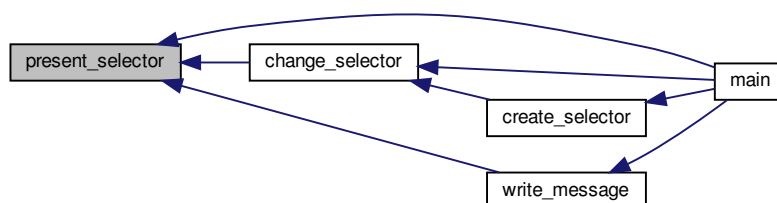
## 4.19.2.9 present\_selector()

```
void present_selector (  
    Selector * selector )
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 4.19.2.10 rotate()

```
void rotate (  
    void * nil )
```

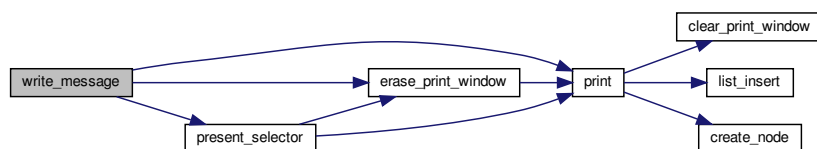
Here is the caller graph for this function:



#### 4.19.2.11 write\_message()

```
void write_message (  
    void * nil )
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 4.19.3 Variable Documentation

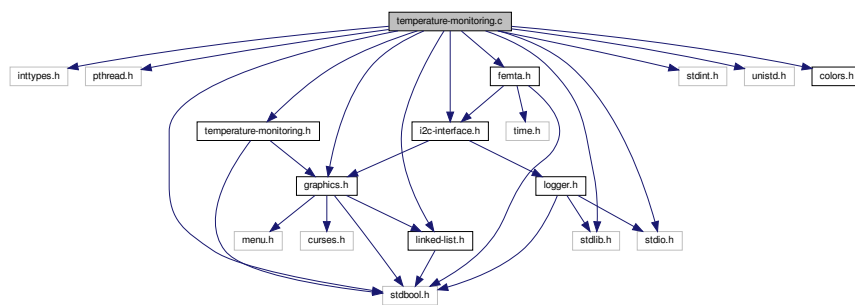
## 4.19.3.1 visible\_selector

```
Selector* visible_selector
```

## 4.20 temperature-monitoring.c File Reference

```
#include <inttypes.h>
#include <pthread.h>
#include <stdbool.h>
#include <stdint.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include "femta.h"
#include "i2c-interface.h"
#include "temperature-monitoring.h"
#include "linked-list.h"
#include "graphics.h"
#include "colors.h"
```

Include dependency graph for temperature-monitoring.c:



## Functions

- void \* [read\\_cpu\\_temperature](#) ()
- bool [initialize\\_temperature\\_monitoring](#) ()
- void [terminate\\_temperature\\_monitoring](#) ()

## Variables

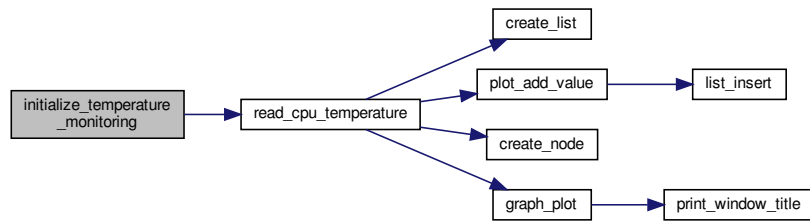
- FILE \* [cpu\\_temperature\\_log\\_file](#)
- char \* [temperature\\_log\\_filename](#) = `"/logs/cpu-temperature-log.txt"`
- pthread\_t [cpu\\_temperature\\_thread](#)
- bool [termination\\_signal](#)
- int [values\\_read](#) = 0

## 4.20.1 Function Documentation

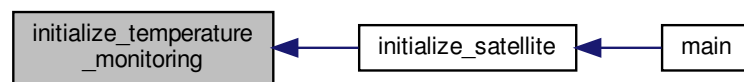
#### 4.20.1.1 initialize\_temperature\_monitoring()

```
bool initialize_temperature_monitoring ( )
```

Here is the call graph for this function:



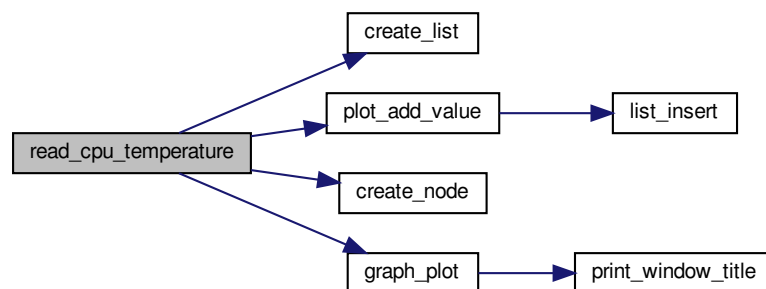
Here is the caller graph for this function:



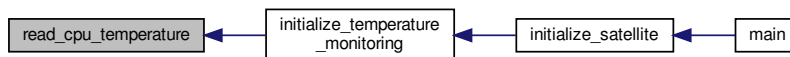
#### 4.20.1.2 read\_cpu\_temperature()

```
void* read_cpu_temperature ( )
```

Here is the call graph for this function:



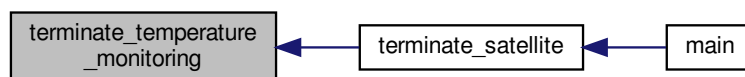
Here is the caller graph for this function:



#### 4.20.1.3 `terminate_temperature_monitoring()`

```
void terminate_temperature_monitoring ( )
```

Here is the caller graph for this function:



### 4.20.2 Variable Documentation

#### 4.20.2.1 `cpu_temperature_log_file`

```
FILE* cpu_temperature_log_file
```

#### 4.20.2.2 `cpu_temperature_thread`

```
pthread_t cpu_temperature_thread
```

#### 4.20.2.3 `temperature_log_filename`

```
char* temperature_log_filename = "./logs/cpu-temperature-log.txt"
```

#### 4.20.2.4 termination\_signal

```
bool termination_signal
```

#### 4.20.2.5 values\_read

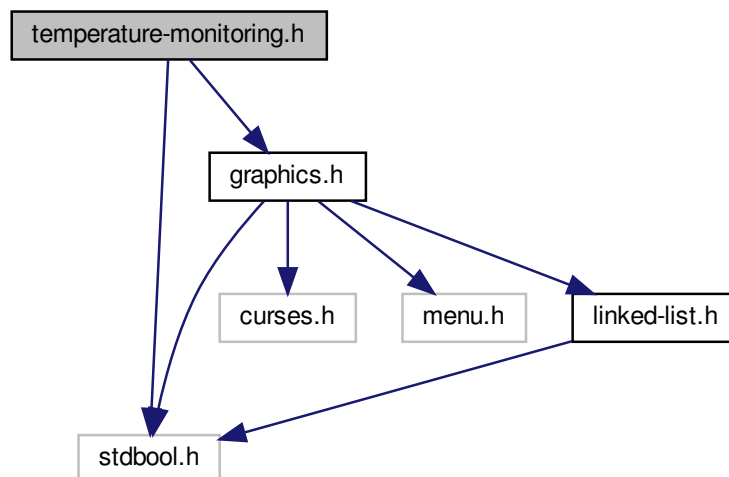
```
int values_read = 0
```

### 4.21 temperature-monitoring.h File Reference

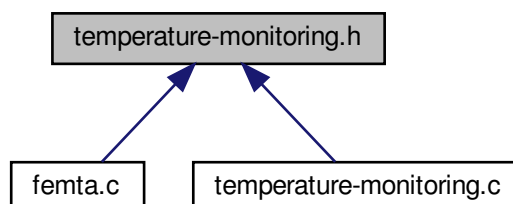
```
#include <stdbool.h>
```

```
#include "graphics.h"
```

Include dependency graph for temperature-monitoring.h:



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





## Functions

- bool [initialize\\_temperature\\_monitoring](#) ()
- void [terminate\\_temperature\\_monitoring](#) ()

## Variables

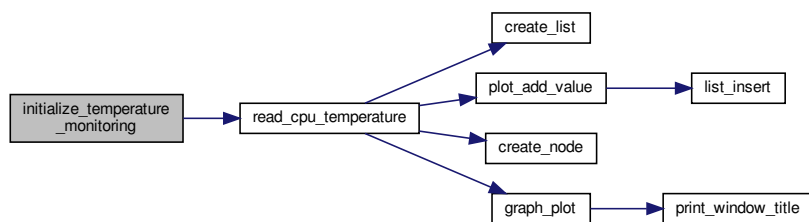
- [Plot](#) \* [temperature\\_plot](#)

### 4.21.1 Function Documentation

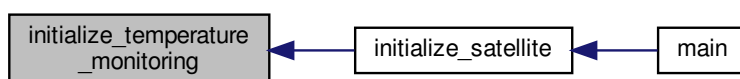
#### 4.21.1.1 initialize\_temperature\_monitoring()

```
bool initialize_temperature_monitoring ( )
```

Here is the call graph for this function:



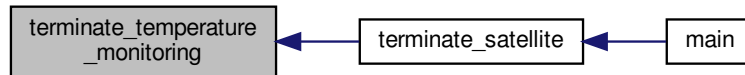
Here is the caller graph for this function:



#### 4.21.1.2 terminate\_temperature\_monitoring()

```
void terminate_temperature_monitoring ( )
```

Here is the caller graph for this function:



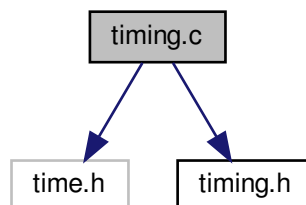
### 4.21.2 Variable Documentation

#### 4.21.2.1 temperature\_plot

```
Plot* temperature_plot
```

## 4.22 timing.c File Reference

```
#include <time.h>
#include "timing.h"
Include dependency graph for timing.c:
```



### Functions

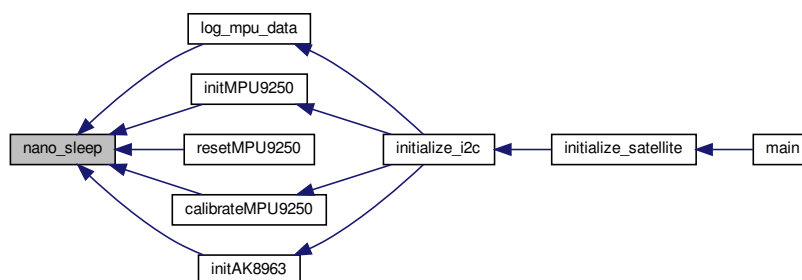
- void `nano_sleep` (long duration)

### 4.22.1 Function Documentation

#### 4.22.1.1 nano\_sleep()

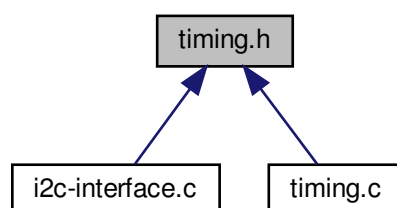
```
void nano_sleep (
    long duration )
```

Here is the caller graph for this function:



## 4.23 timing.h File Reference

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



### Functions

- void `nano_sleep` (long duration)

### 4.23.1 Function Documentation

#### 4.23.1.1 nano\_sleep()

```
void nano_sleep (
    long duration )
```

Here is the caller graph for this function:

