My Project

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

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
```

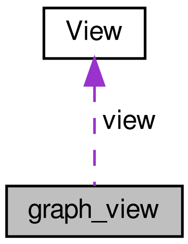
The documentation for this struct was generated from the following file:

- 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
```

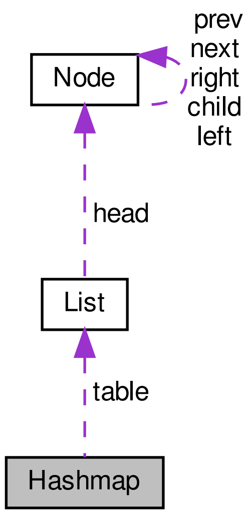
The documentation for this struct was generated from the following file:

- 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) ()
```

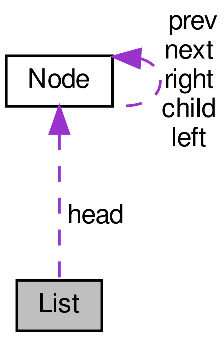
The documentation for this struct was generated from the following file:

- 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 doublely_linked

```
bool List::doublely_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
```

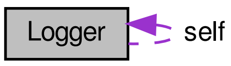
The documentation for this struct was generated from the following file:

- 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
```

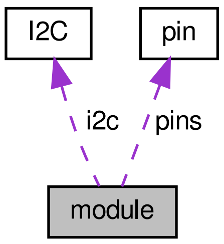
The documentation for this struct was generated from the following file:

- 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
```

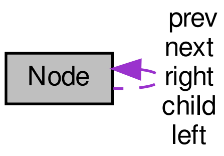
The documentation for this struct was generated from the following file:

- 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
```

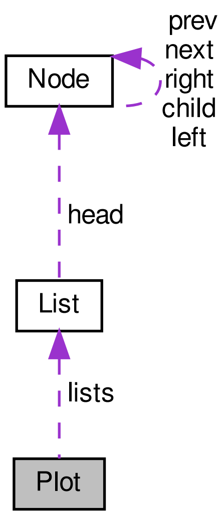
The documentation for this struct was generated from the following file:

- 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
```

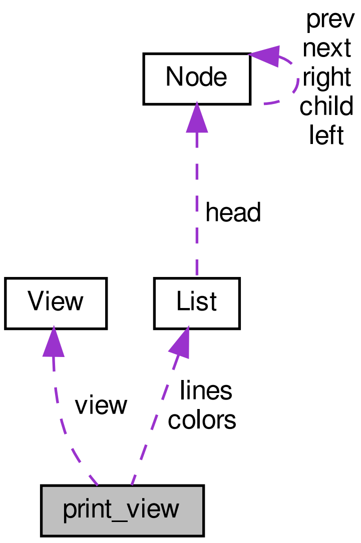
The documentation for this struct was generated from the following file:

- 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
```

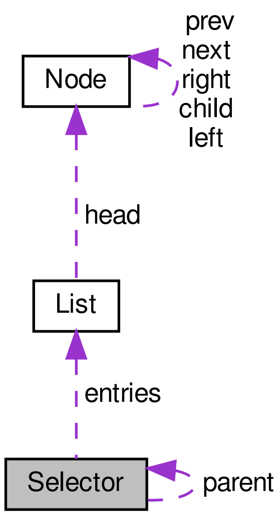
The documentation for this struct was generated from the following file:

- 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
```

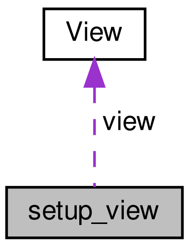
The documentation for this struct was generated from the following file:

- 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:

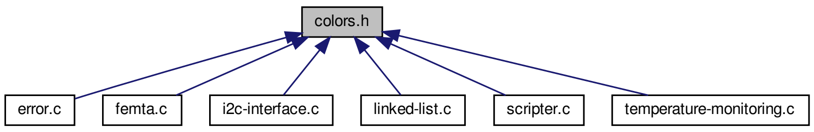
- graphics.h

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"
```

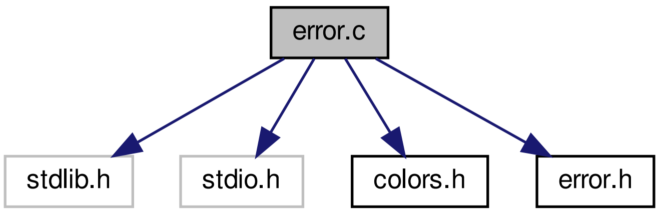
4.1.1.17 YELLOW

```
#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

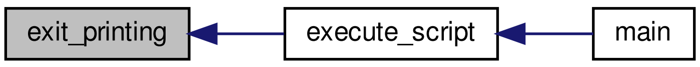
- void exit_printing (char ∗message, char code)

4.2.1 Function Documentation

4.2.1.1 exit_printing()

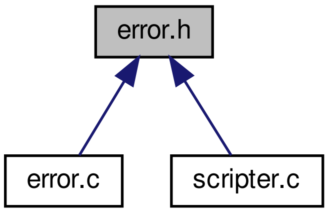
```
void exit_printing (
             char ∗ message,
             char code )
```

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

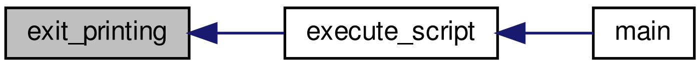
```
#define ERROR_PROGRAMMER 1
```

4.3.2 Function Documentation

4.3.2.1 exit_printing()

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

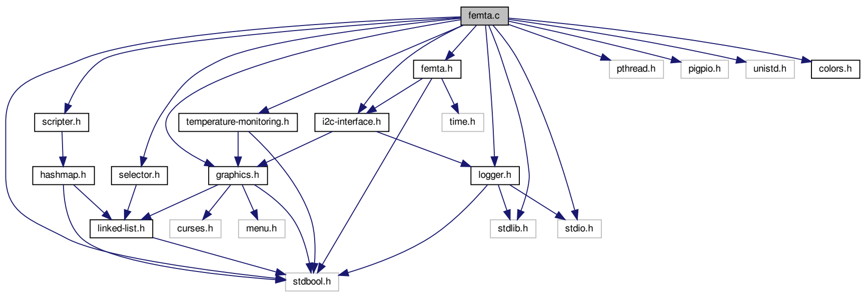
Here is the caller graph for this function:



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

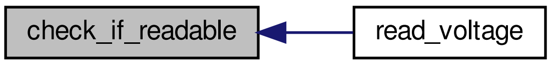
```
#define UART_STATE 3
```

4.4.2 Function Documentation

4.4.2.1 check_if_readable()

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

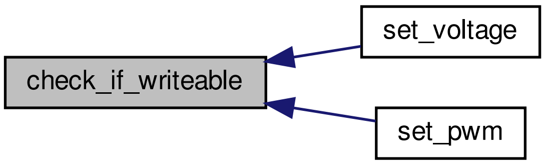
Here is the caller graph for this function:



4.4.2.2 check_if_writeable()

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

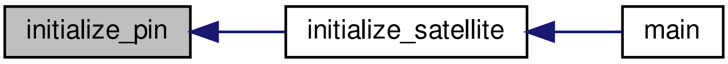
Here is the caller graph for this function:



4.4.2.3 initialize_pin()

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

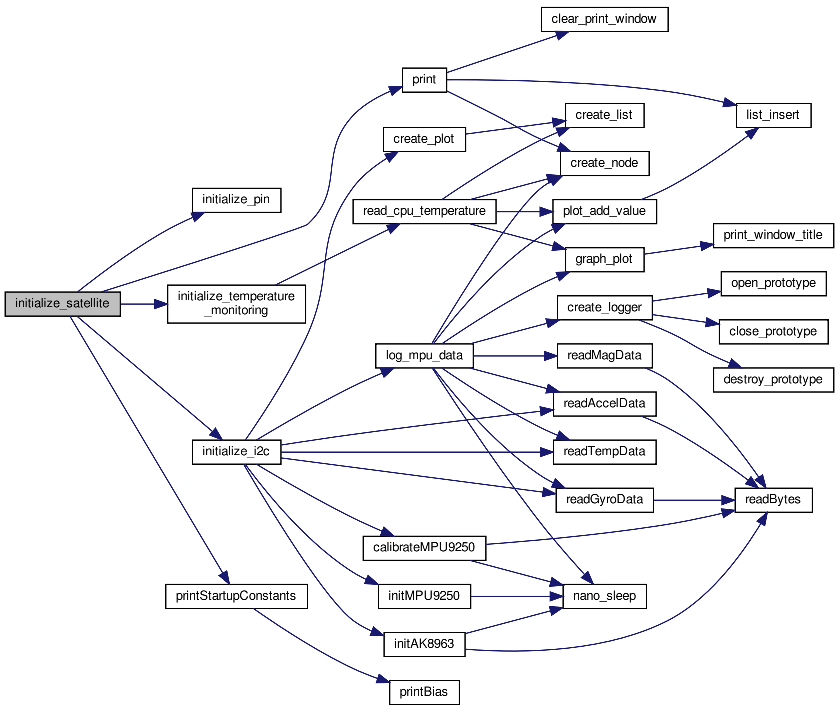
Here is the caller graph for this function:



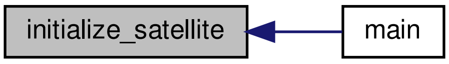
4.4.2.4 initialize_satellite()

```
void initialize_satellite ( )
```

Here is the call graph for this function:



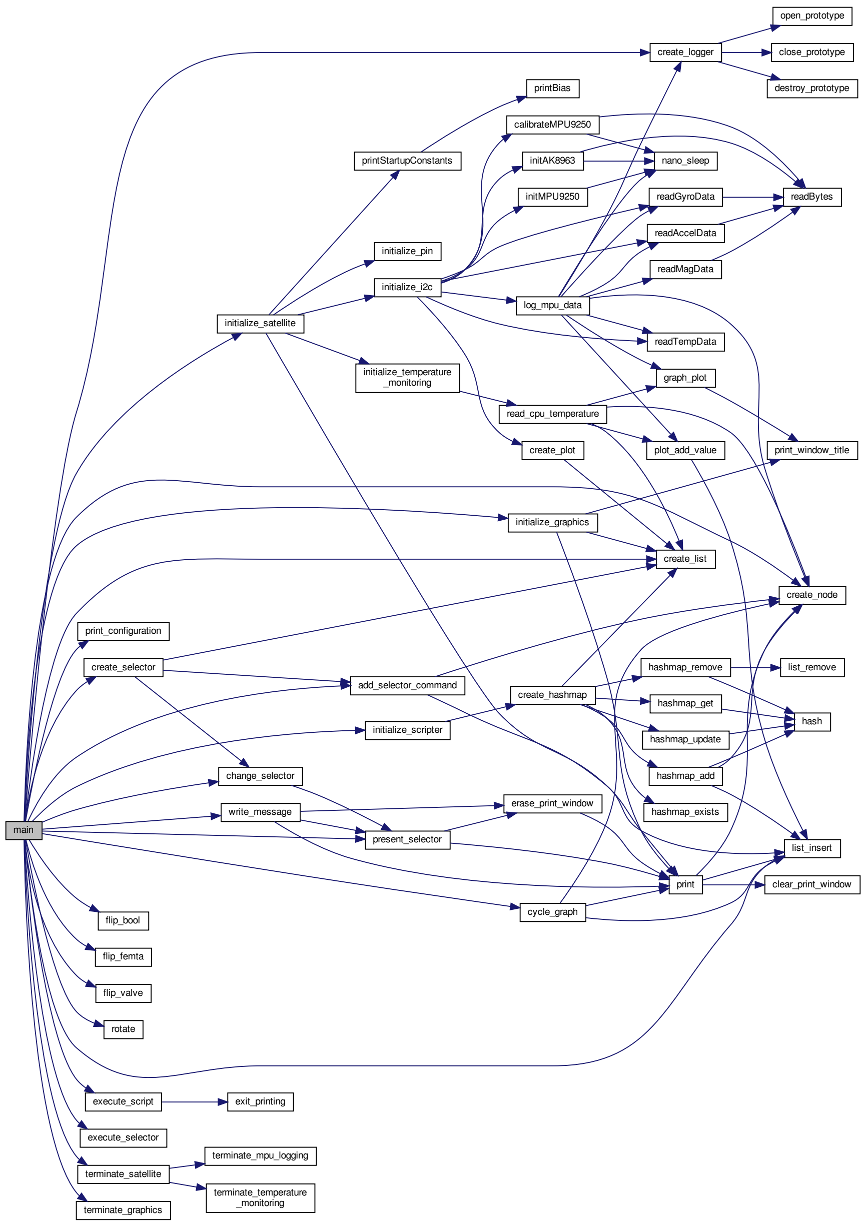
Here is the caller graph for this function:



4.4.2.5 main()

```
int main ( )
```

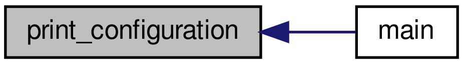
Here is the call graph for this function:



4.4.2.6 print_configuration()

```
void print_configuration ( )
```

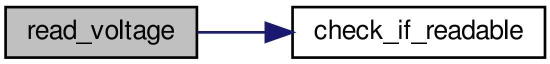
Here is the caller graph for this function:



4.4.2.7 read_voltage()

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

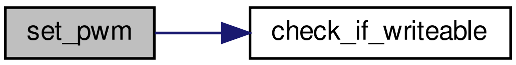
Here is the call graph for this function:



4.4.2.8 set_pwm()

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

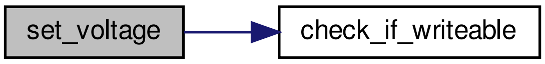
Here is the call graph for this function:



4.4.2.9 set_voltage()

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

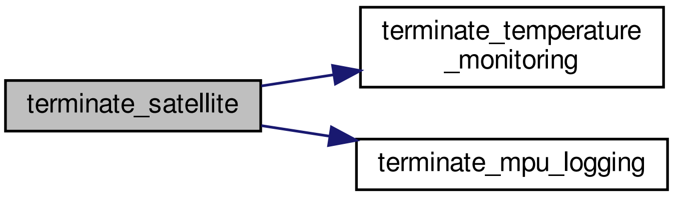
Here is the call graph for this function:



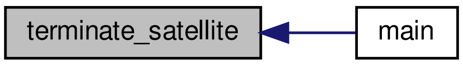
4.4.2.10 terminate_satellite()

```
void terminate_satellite ( )
```

Here is the call graph for this function:



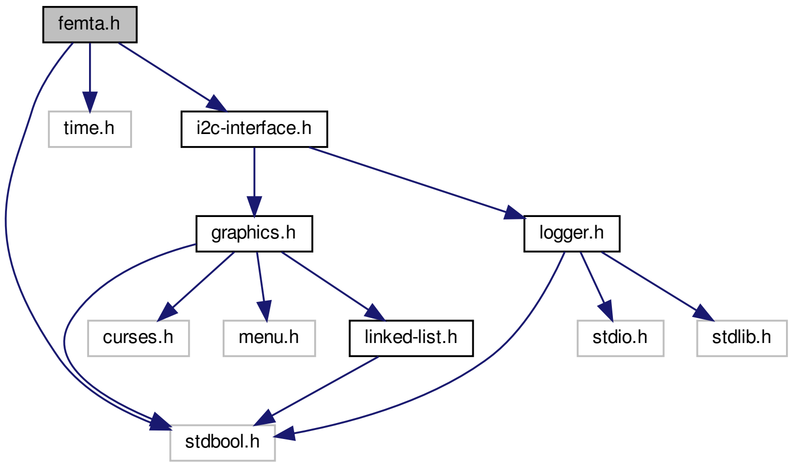
Here is the caller graph for this function:



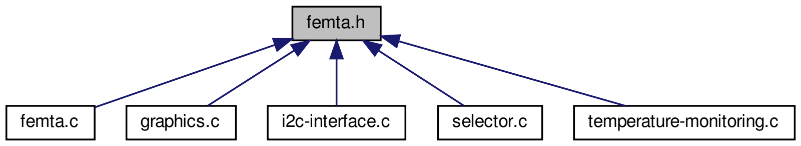
4.5 femta.h File Reference

```
#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

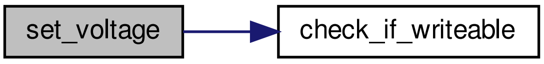
```
typedef struct UART UART
```

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
```

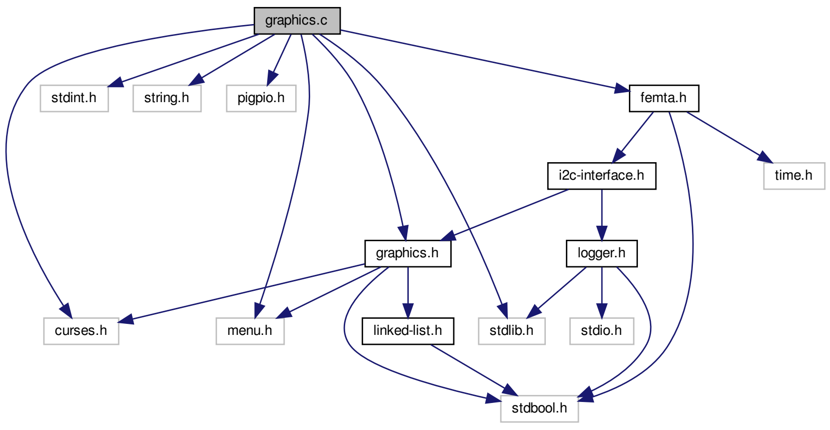
4.5.3.5 Valve

```
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

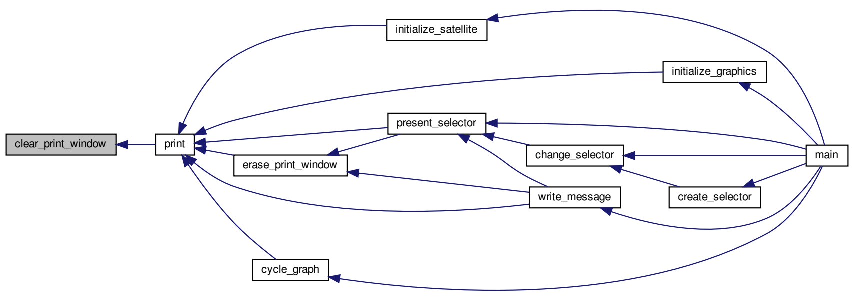
```
#define UART_STATE 3
```

4.6.2 Function Documentation

4.6.2.1 clear_print_window()

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

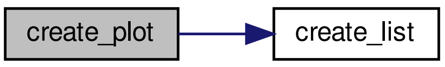
Here is the caller graph for this function:



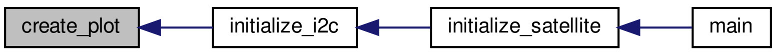
4.6.2.2 create_plot()

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

Here is the call graph for this function:



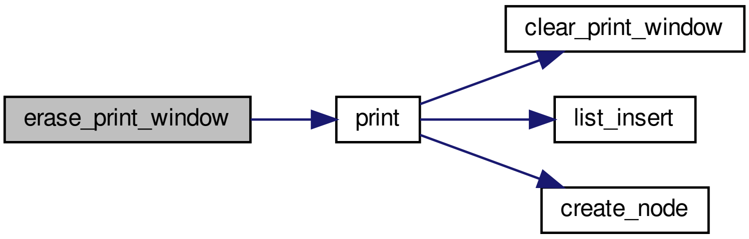
Here is the caller graph for this function:



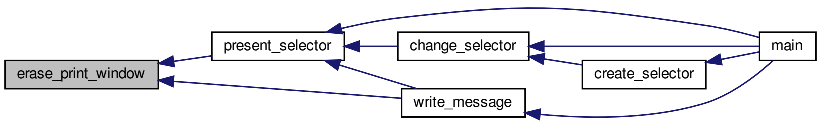
4.6.2.3 erase_print_window()

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

Here is the call graph for this function:



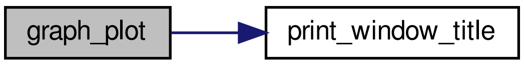
Here is the caller graph for this function:



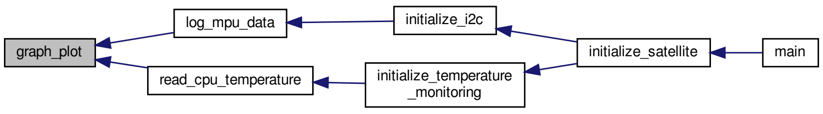
4.6.2.4 graph_plot()

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

Here is the call graph for this function:



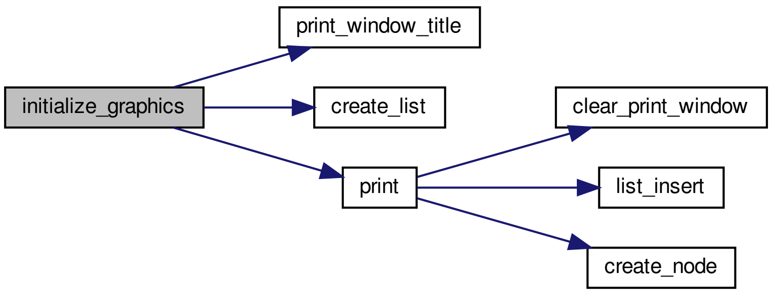
Here is the caller graph for this function:



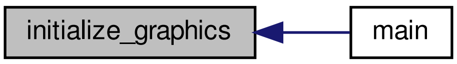
4.6.2.5 initialize_graphics()

```
void initialize_graphics ( )
```

Here is the call graph for this function:



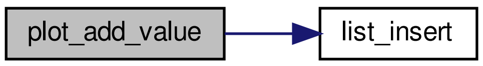
Here is the caller graph for this function:



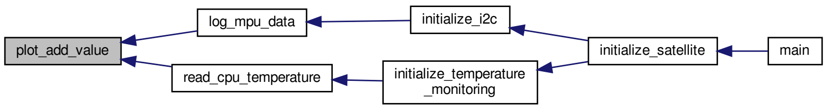
4.6.2.6 plot_add_value()

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

Here is the call graph for this function:



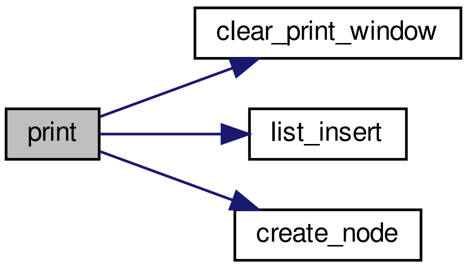
Here is the caller graph for this function:



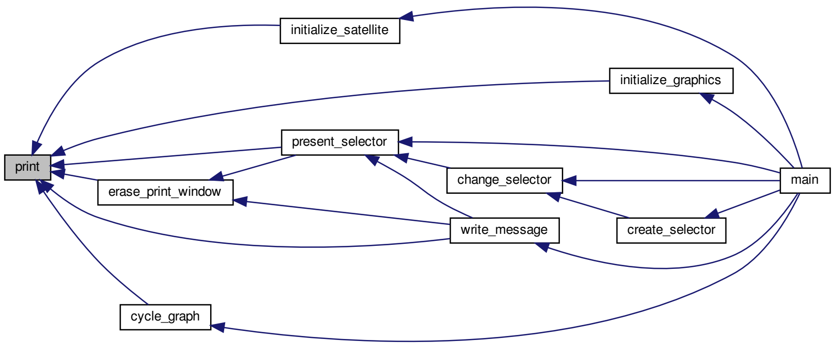
4.6.2.7 print()

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

Here is the call graph for this function:



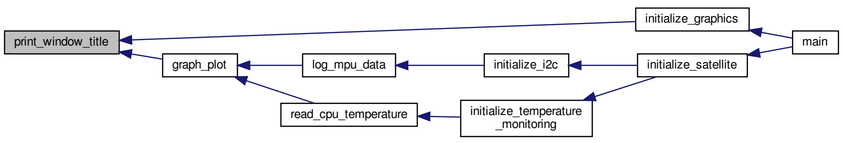
Here is the caller graph for this function:



4.6.2.8 print_window_title() [1/2]

```
void print_window_title ( )
```

Here is the caller graph for this function:



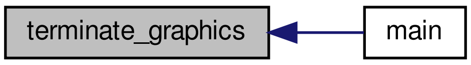
4.6.2.9 print_window_title() [2/2]

```
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
```

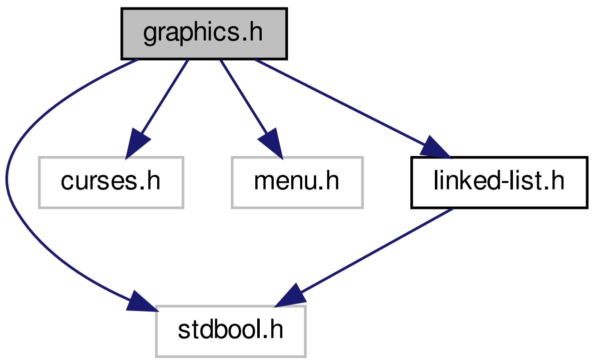
4.6.3.4 setup_views

```
setup_view** setup_views
```

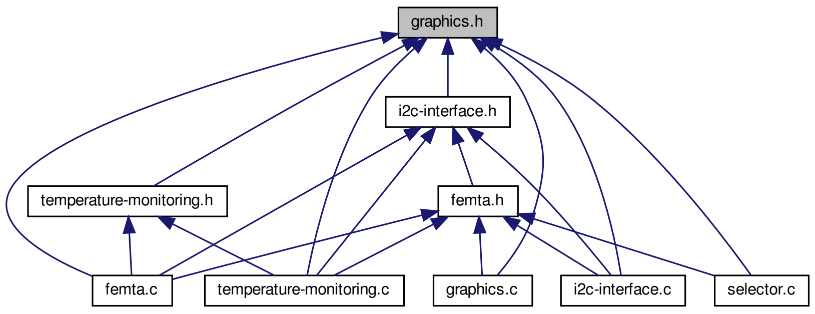
4.7 graphics.h File Reference

```
#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

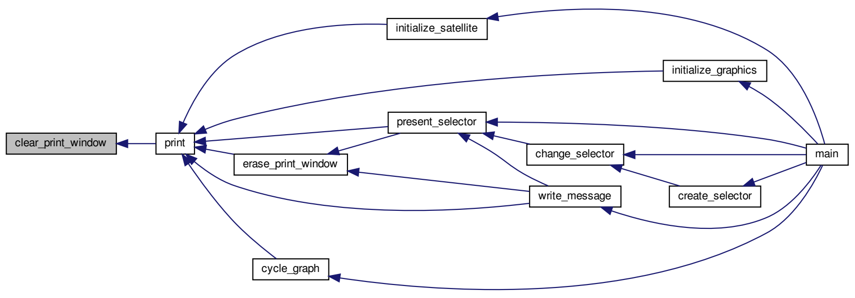
```
typedef struct View View
```

4.7.3 Function Documentation

4.7.3.1 clear_print_window()

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

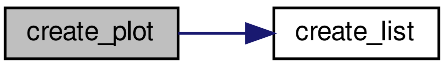
Here is the caller graph for this function:



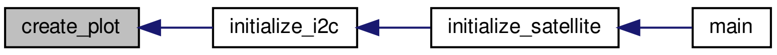
4.7.3.2 create_plot()

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

Here is the call graph for this function:



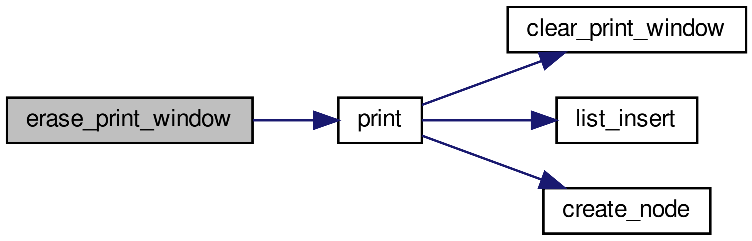
Here is the caller graph for this function:



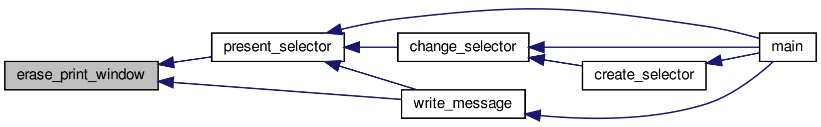
4.7.3.3 erase_print_window()

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

Here is the call graph for this function:



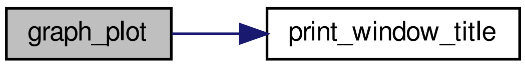
Here is the caller graph for this function:



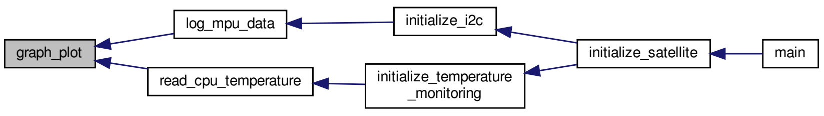
4.7.3.4 graph_plot()

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

Here is the call graph for this function:



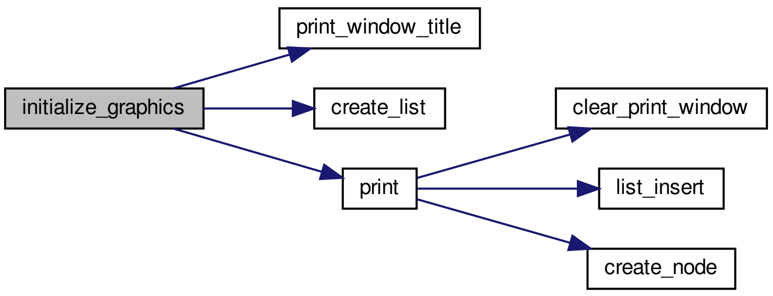
Here is the caller graph for this function:



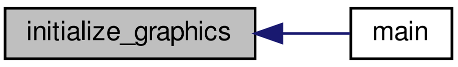
4.7.3.5 initialize_graphics()

```
void initialize_graphics ( )
```

Here is the call graph for this function:



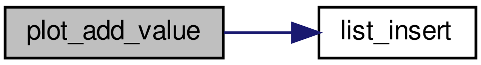
Here is the caller graph for this function:



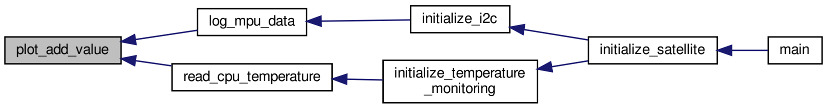
4.7.3.6 plot_add_value()

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

Here is the call graph for this function:



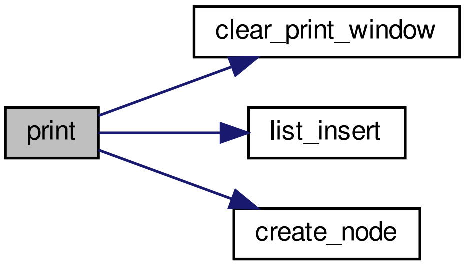
Here is the caller graph for this function:



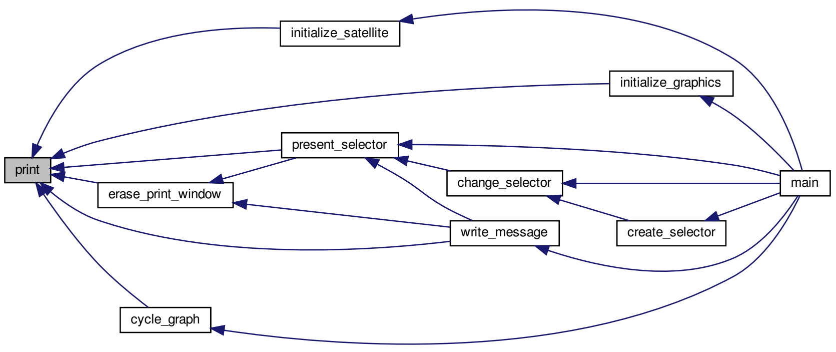
4.7.3.7 print()

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

Here is the call graph for this function:



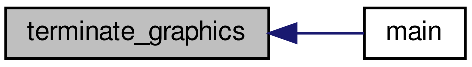
Here is the caller graph for this function:



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
```

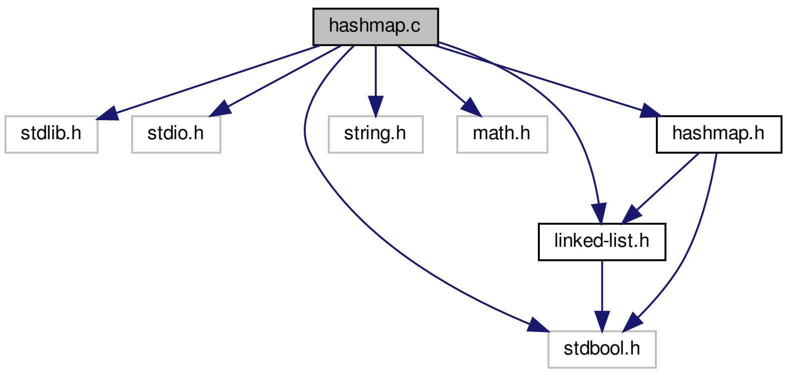
4.7.4.5 owner_index_list

```
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

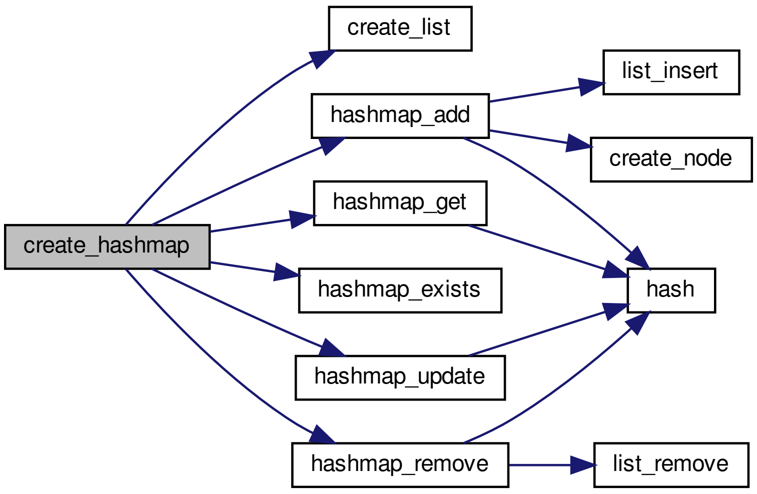
- void ∗ hashmap_get (Hashmap ∗this, char ∗string)
- void hashmap_add (Hashmap ∗this, char ∗string, void ∗datum)
- void hashmap_update (Hashmap ∗this, char ∗string, void ∗datum)
- bool hashmap_exists (Hashmap ∗this, char ∗string)
- void hashmap_remove (Hashmap ∗this, char ∗string)
- Hashmap ∗ create_hashmap (int expected_size)
- int hash (char ∗string, int upper_bound)

4.8.1 Function Documentation

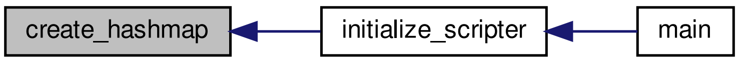
4.8.1.1 create_hashmap()

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

Here is the call graph for this function:



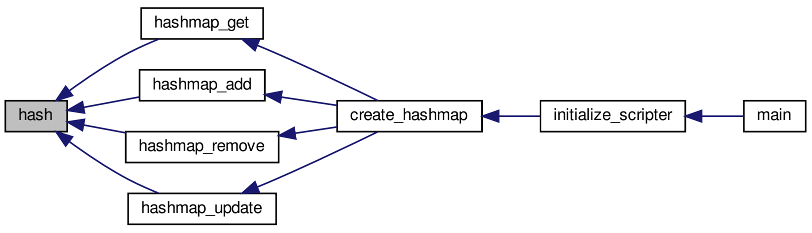
Here is the caller graph for this function:



4.8.1.2 hash()

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

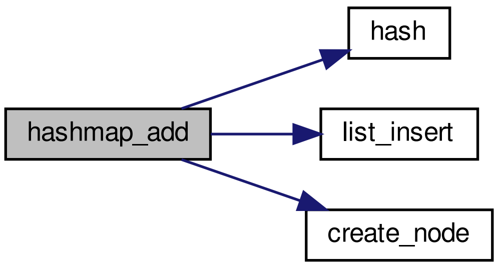
Here is the caller graph for this function:



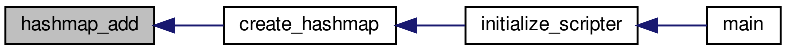
4.8.1.3 hashmap_add()

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

Here is the call graph for this function:



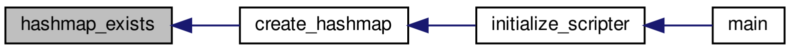
Here is the caller graph for this function:



4.8.1.4 hashmap_exists()

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

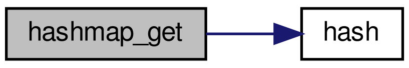
Here is the caller graph for this function:



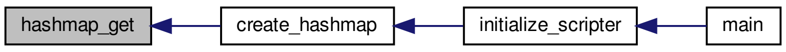
4.8.1.5 hashmap_get()

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

Here is the call graph for this function:



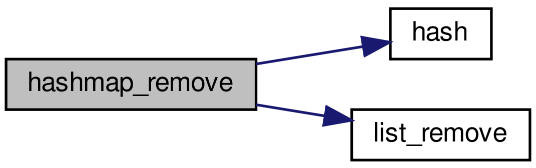
Here is the caller graph for this function:



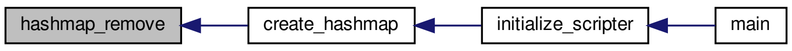
4.8.1.6 hashmap_remove()

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

Here is the call graph for this function:



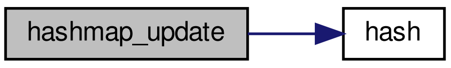
Here is the caller graph for this function:



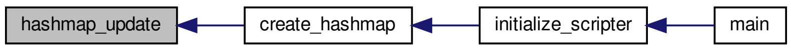
4.8.1.7 hashmap_update()

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

Here is the call graph for this function:



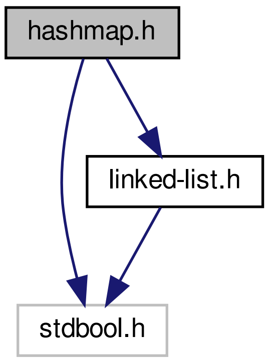
Here is the caller graph for this function:



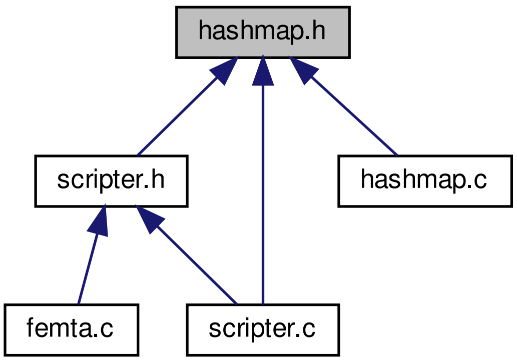
4.9 hashmap.h File Reference

```
#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

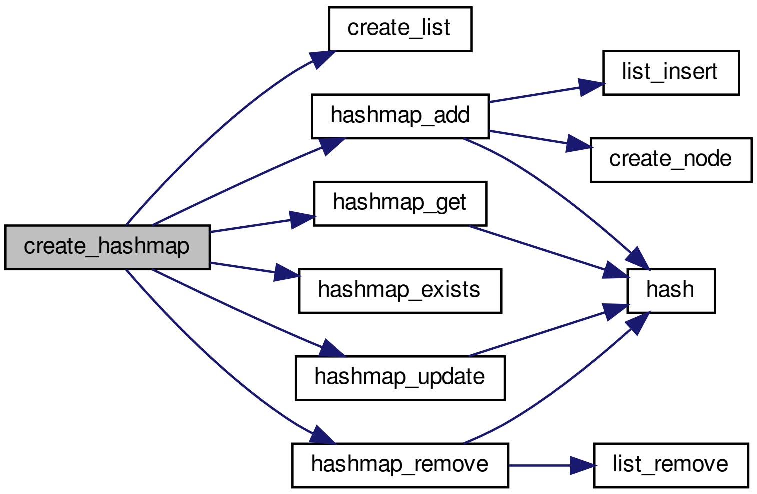
```
typedef struct HashmapElement HashmapElement
```

4.9.3 Function Documentation

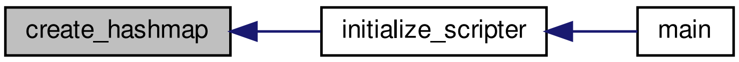
4.9.3.1 create_hashmap()

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

Here is the call graph for this function:



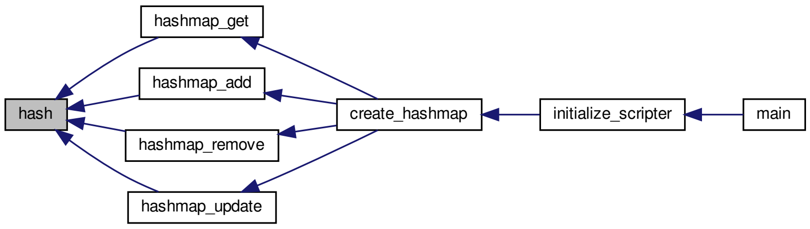
Here is the caller graph for this function:



4.9.3.2 hash()

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

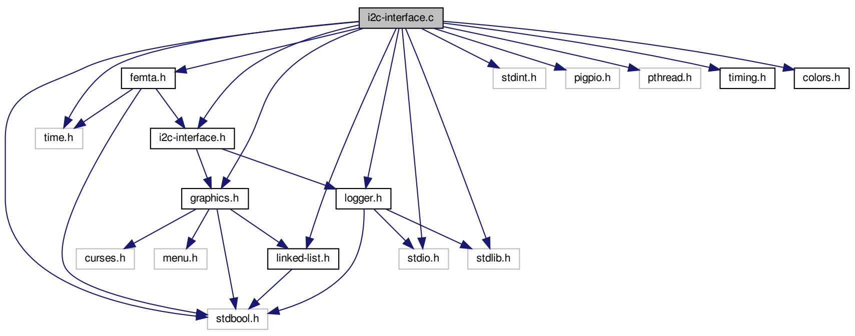
Here is the caller graph for this function:



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/MP↩U9250/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 requirments, 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

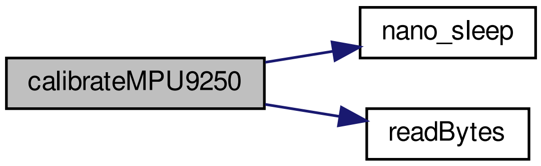
MFS_14BITS	
MFS_16BITS	

4.10.3 Function Documentation

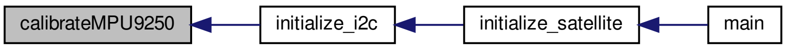
4.10.3.1 calibrateMPU9250()

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

Here is the call graph for this function:



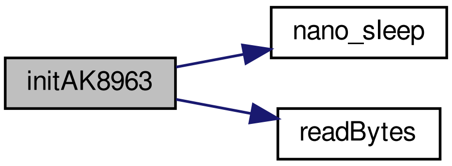
Here is the caller graph for this function:



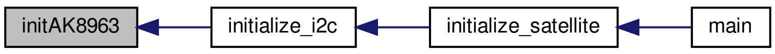
4.10.3.2 initAK8963()

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

Here is the call graph for this function:



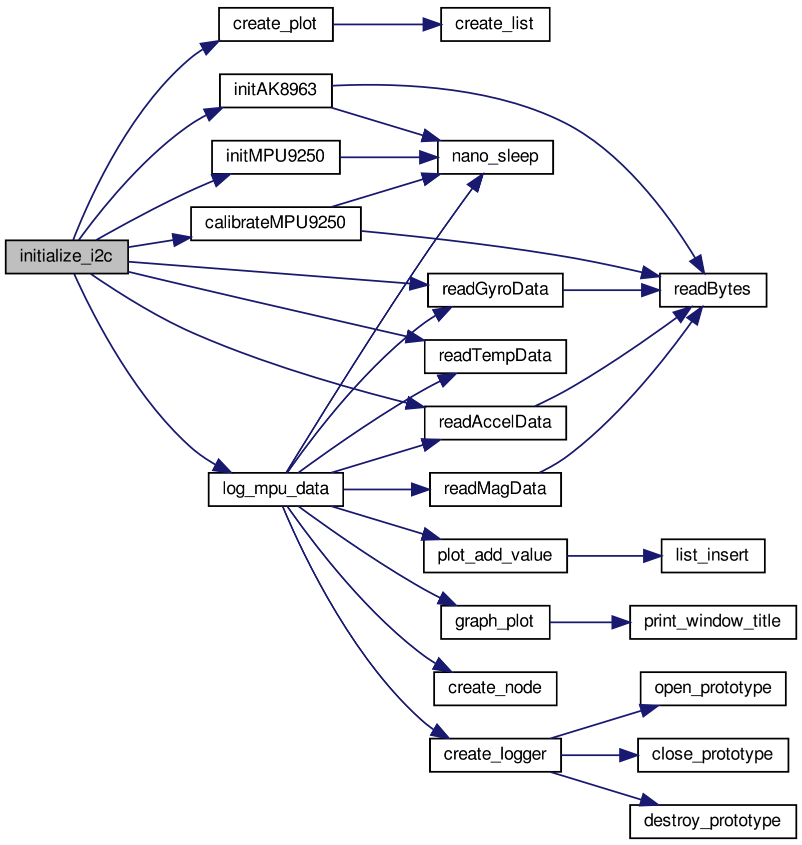
Here is the caller graph for this function:



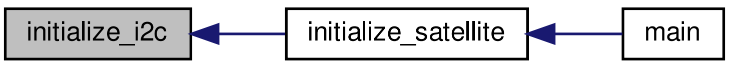
4.10.3.3 initialize_i2c()

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

Here is the call graph for this function:



Here is the caller graph for this function:

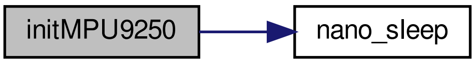


4.10.3.4 initMPU9250()

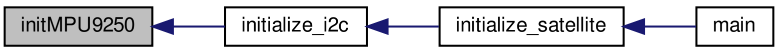
```
void initMPU9250 ( )
```

?!?!?

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



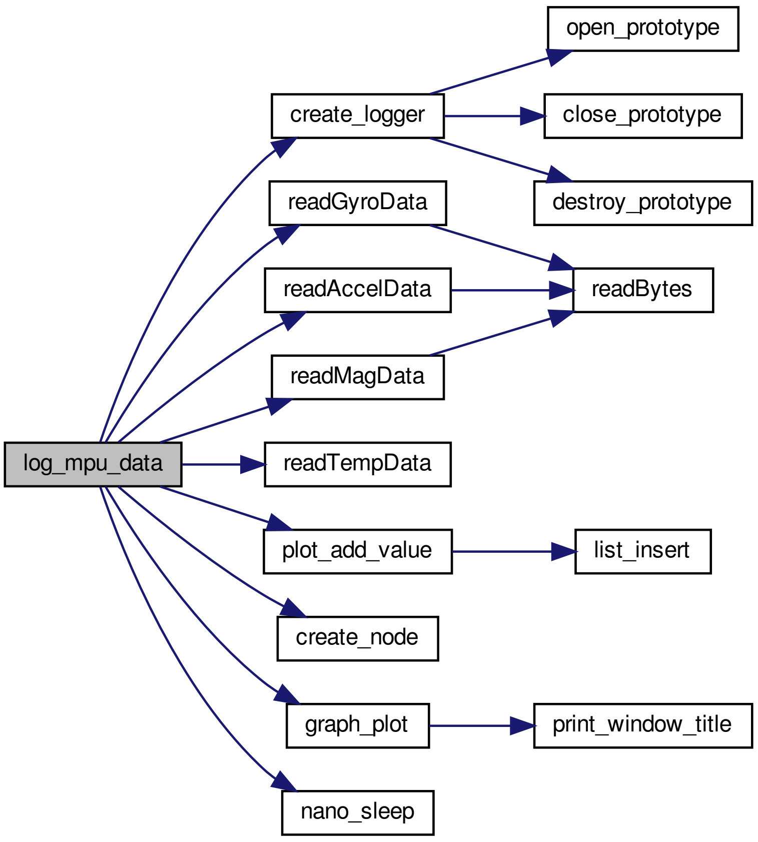
Here is the caller graph for this function:



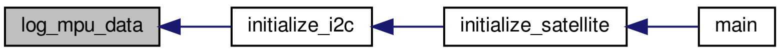
4.10.3.5 log_mpu_data()

```
void* log_mpu_data ( )
```

Here is the call graph for this function:



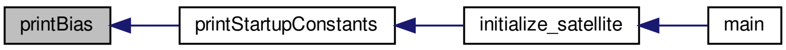
Here is the caller graph for this function:



4.10.3.6 printBias()

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

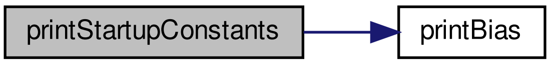
Here is the caller graph for this function:



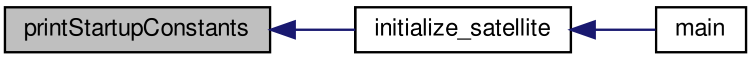
4.10.3.7 printStartupConstants()

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

Here is the call graph for this function:



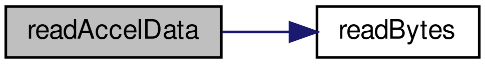
Here is the caller graph for this function:



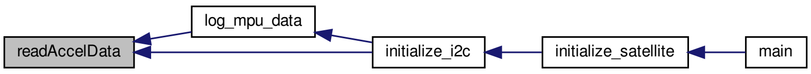
4.10.3.8 readAccelData()

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

Here is the call graph for this function:



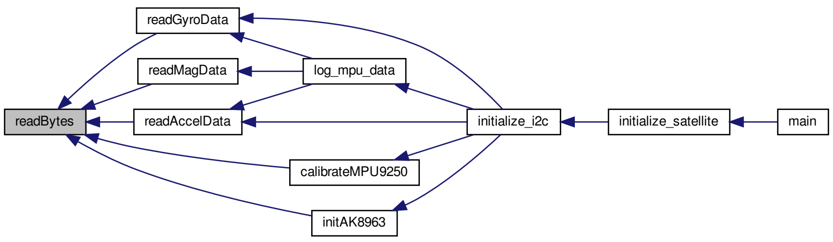
Here is the caller graph for this function:



4.10.3.9 readBytes()

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

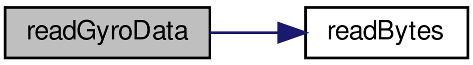
Here is the caller graph for this function:



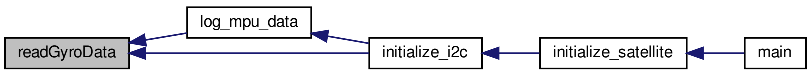
4.10.3.10 readGyroData()

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

Here is the call graph for this function:



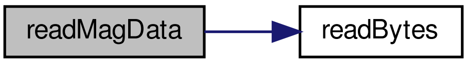
Here is the caller graph for this function:



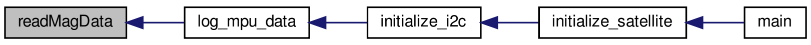
4.10.3.11 readMagData()

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

Here is the call graph for this function:



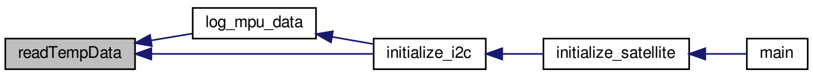
Here is the caller graph for this function:



4.10.3.12 readTempData()

```
float readTempData ( )
```

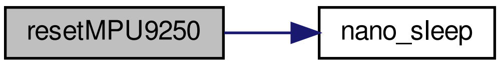
Here is the caller graph for this function:



4.10.3.13 resetMPU9250()

```
void resetMPU9250 ( )
```

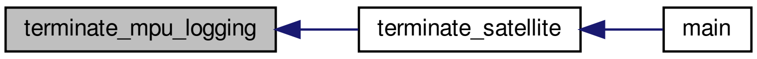
Here is the call graph for this function:



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
```

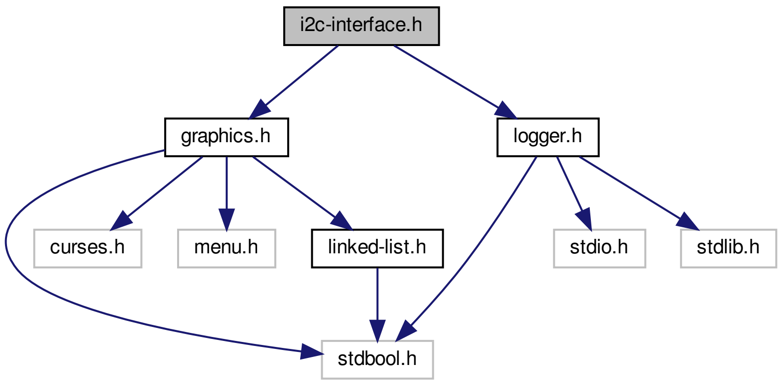
4.10.4.18 newMagData

```
bool newMagData = false
```

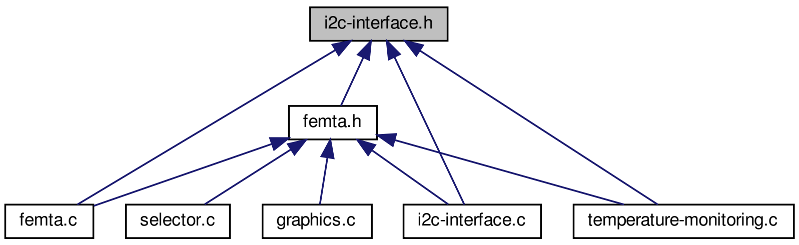
4.11 i2c-interface.h File Reference

```
#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

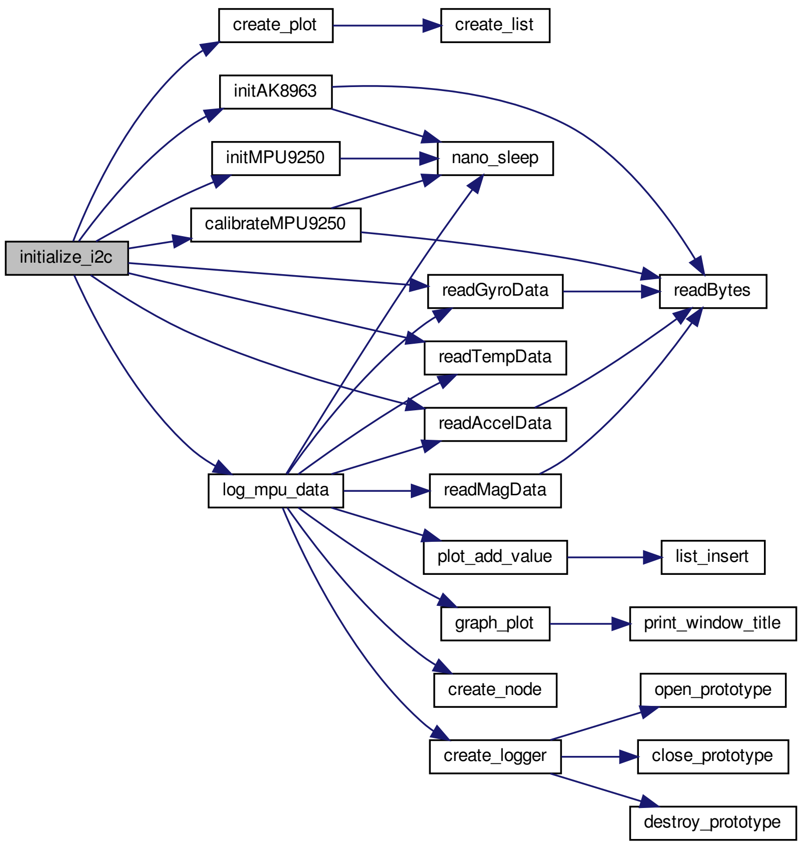
```
typedef struct module module
```

4.11.2 Function Documentation

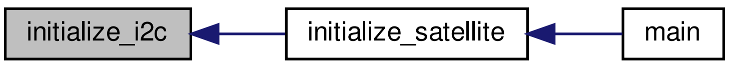
4.11.2.1 initialize_i2c()

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

Here is the call graph for this function:



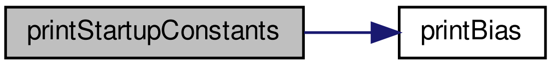
Here is the caller graph for this function:



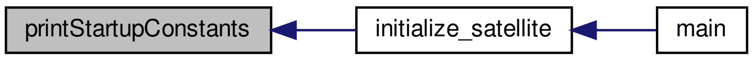
4.11.2.2 printStartupConstants()

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

Here is the call graph for this function:



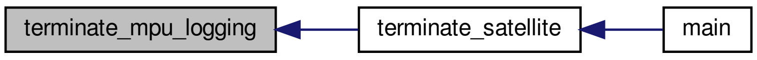
Here is the caller graph for this function:



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
```

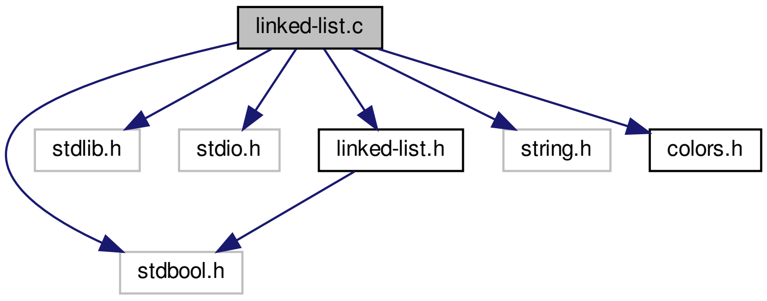
4.11.3.5 mpu_magn_plot

```
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

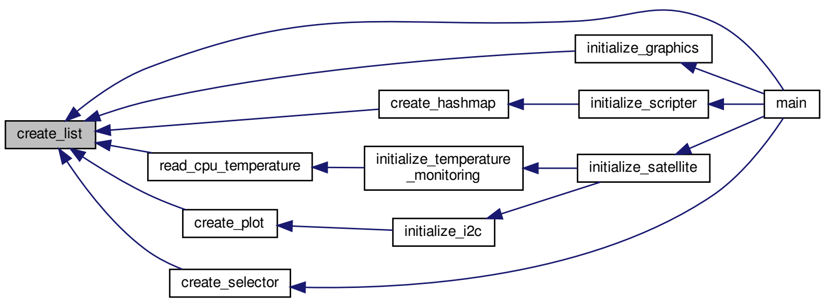
- Node ∗ create_node (void ∗value)
- List ∗ create_list (unsigned int limit, bool doublely_linked)
- void list_insert (List ∗list, Node ∗node)
- void list_remove (List ∗list, Node ∗node)

4.12.1 Function Documentation

4.12.1.1 create_list()

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

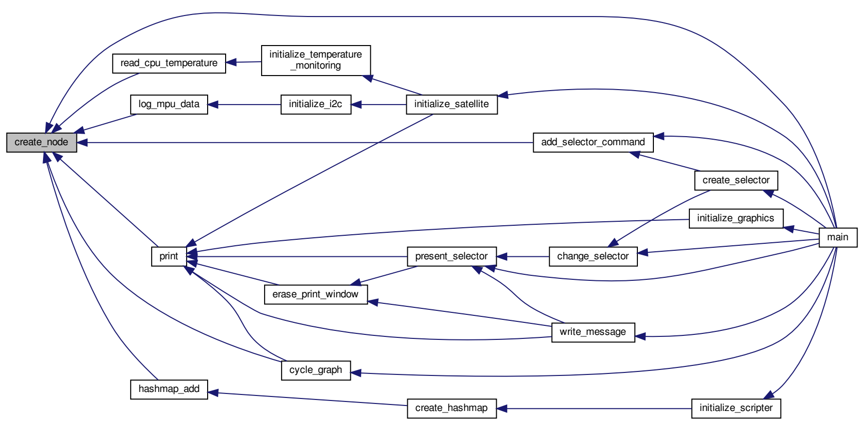
Here is the caller graph for this function:



4.12.1.2 create_node()

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

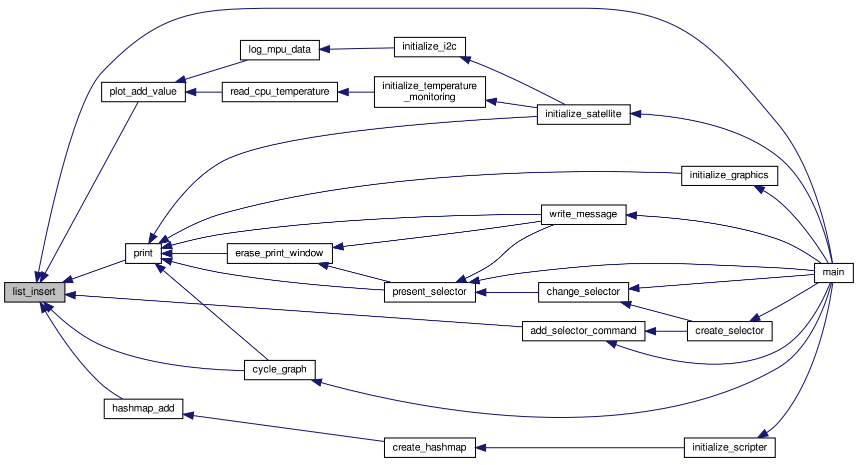
Here is the caller graph for this function:



4.12.1.3 list_insert()

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

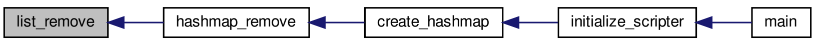
Here is the caller graph for this function:



4.12.1.4 list_remove()

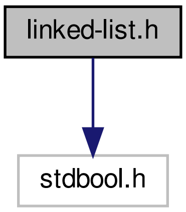
```
void list_remove (
            List * list,
            Node * node )
```

Here is the caller graph for this function:

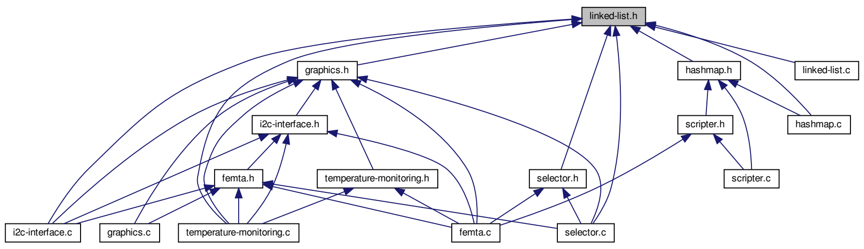


4.13 linked-list.h File Reference

```
#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

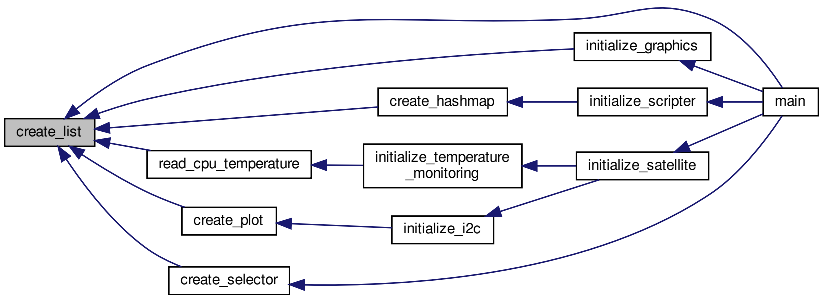
```
typedef struct Node Node
```

4.13.3 Function Documentation

4.13.3.1 create_list()

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

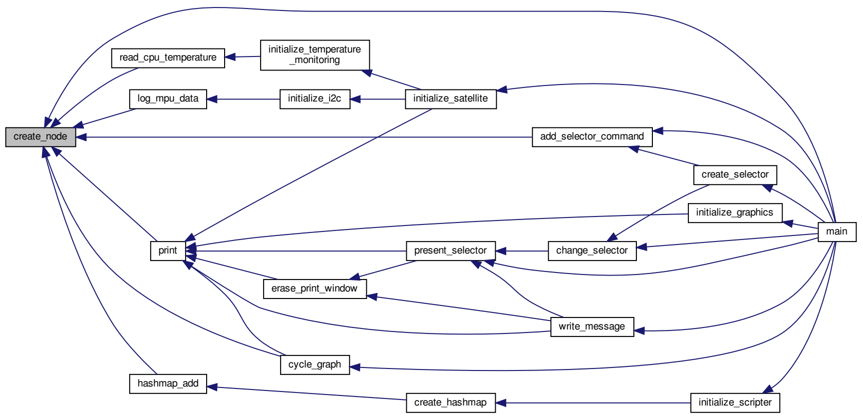
Here is the caller graph for this function:



4.13.3.2 create_node()

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

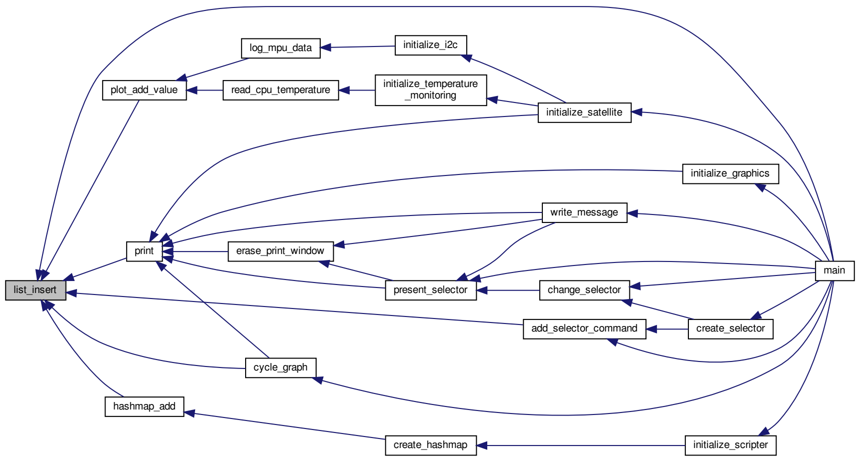
Here is the caller graph for this function:



4.13.3.3 list_insert()

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

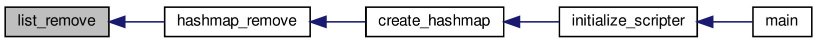
Here is the caller graph for this function:



4.13.3.4 list_remove()

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

Here is the caller graph for this function:

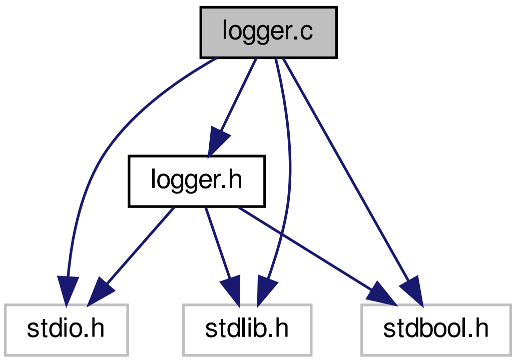


4.14 logger.c File Reference

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

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

Include dependency graph for logger.c:



Functions

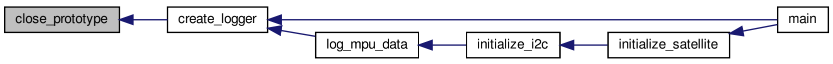
- bool open_prototype (Logger ∗self)
- bool close_prototype (Logger ∗self)
- void destroy_prototype (Logger ∗self)
- Logger ∗ create_logger (char ∗filename)

4.14.1 Function Documentation

4.14.1.1 close_prototype()

```
bool close_prototype (
            Logger ∗ self )
```

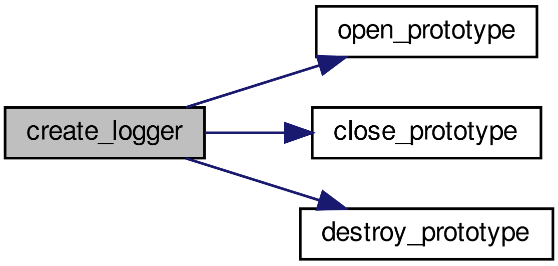
Here is the caller graph for this function:



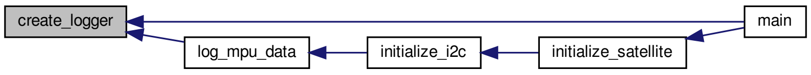
4.14.1.2 create_logger()

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

Here is the call graph for this function:



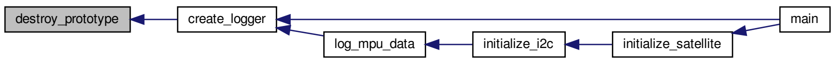
Here is the caller graph for this function:



4.14.1.3 destroy_prototype()

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

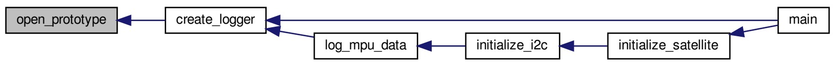
Here is the caller graph for this function:



4.14.1.4 open_prototype()

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

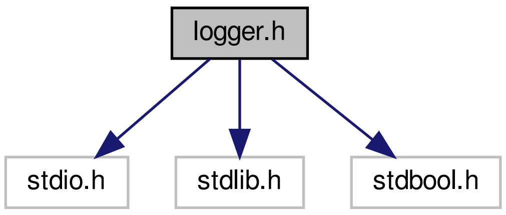
Here is the caller graph for this function:



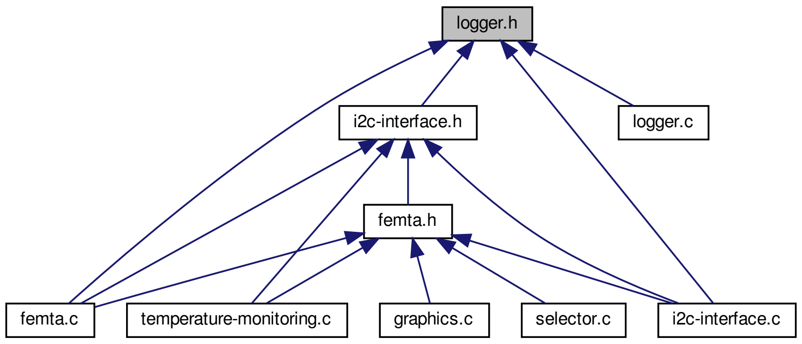
4.15 logger.h File Reference

```
#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

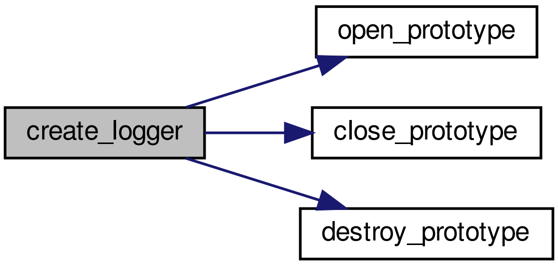
```
typedef struct Logger Logger
```

4.15.2 Function Documentation

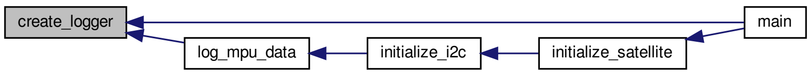
4.15.2.1 create_logger()

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

Here is the call graph for this function:



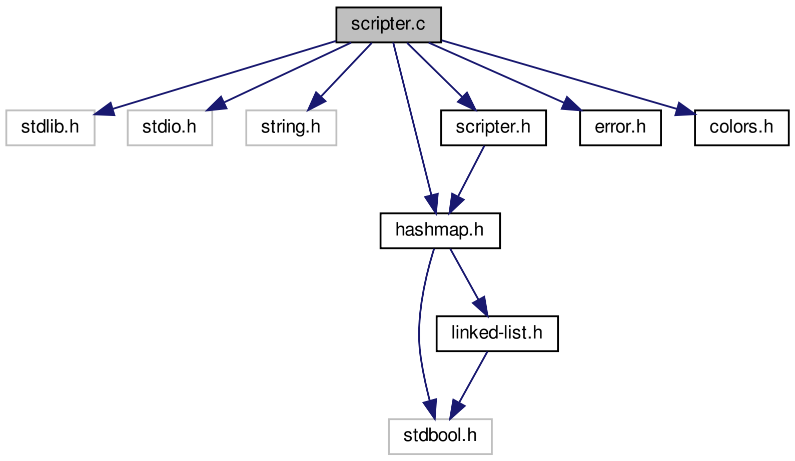
Here is the caller graph for this function:



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

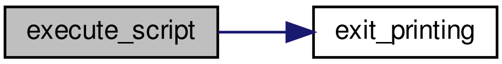
4.16.1.1 define_script_action()

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

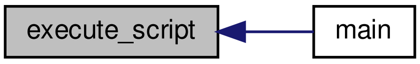
4.16.1.2 execute_script()

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

Here is the call graph for this function:



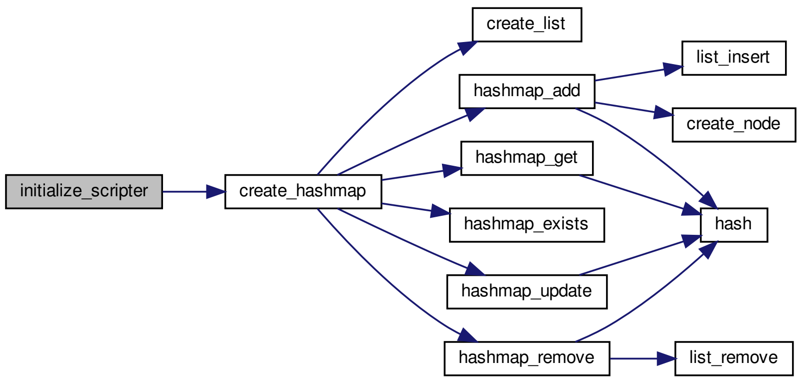
Here is the caller graph for this function:



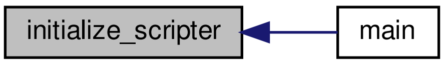
4.16.1.3 initialize_scripter()

```
void initialize_scripter ( )
```

Here is the call graph for this function:



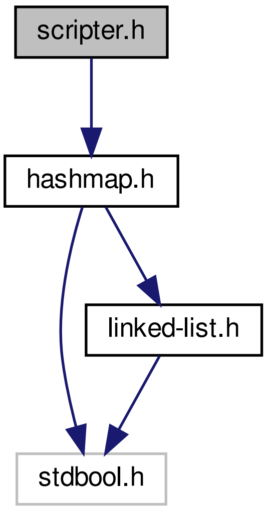
Here is the caller graph for this function:



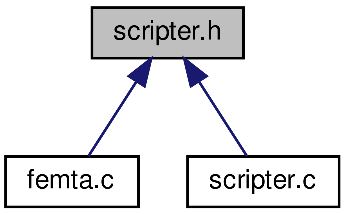
4.17 scripter.h File Reference

```
#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

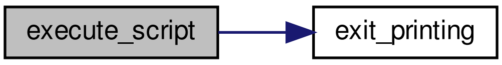
4.17.2.1 define_script_action()

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

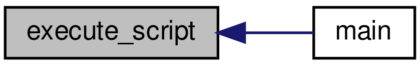
4.17.2.2 execute_script()

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

Here is the call graph for this function:



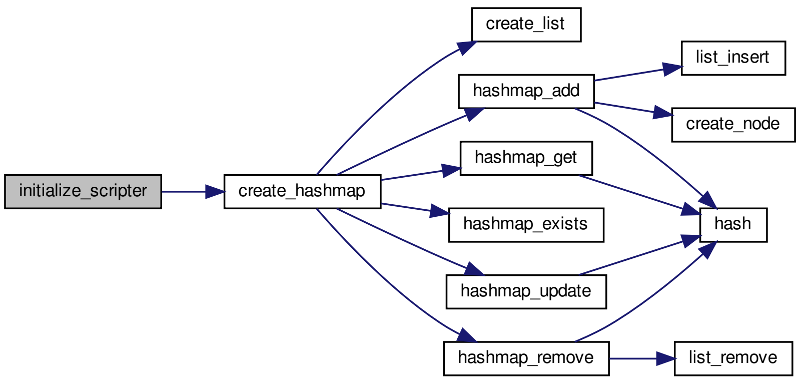
Here is the caller graph for this function:



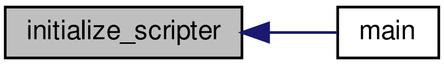
4.17.2.3 initialize_scripter()

```
void initialize_scripter ( )
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.17.3 Variable Documentation

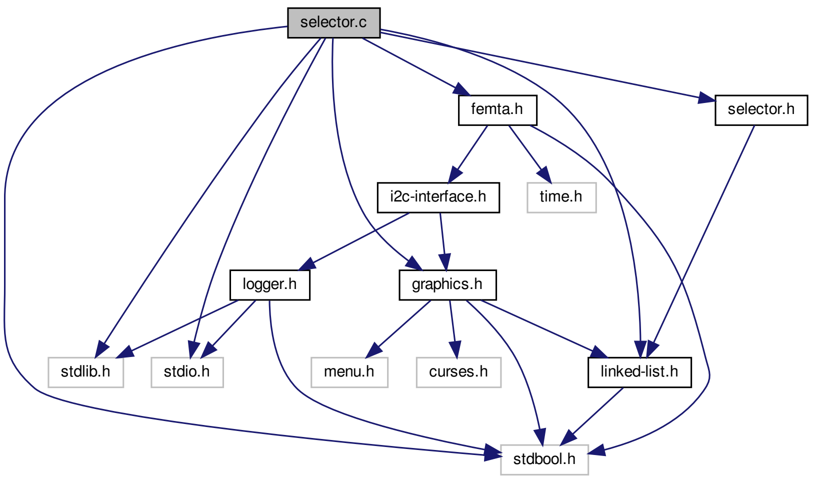
4.17.3.1 action_table

```
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

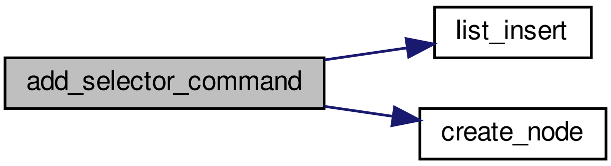
- void change_selector (void ∗selector)
- Selector ∗ create_selector (Selector ∗parent)
- void add_selector_command (Selector ∗selector, char key, char ∗text, lambda action, void ∗argument)
- void execute_selector (Selector ∗selector, char key)
- void present_selector (Selector ∗selector)
- void flip_bool (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

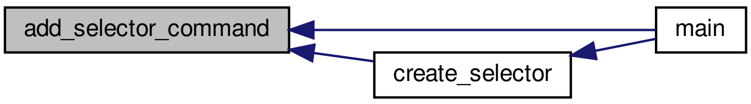
4.18.1.1 add_selector_command()

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

Here is the call graph for this function:



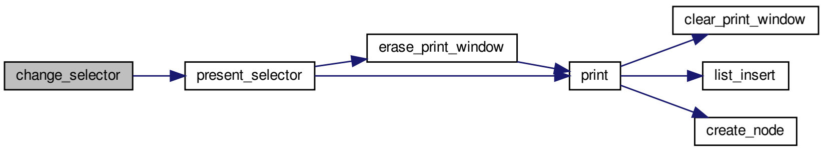
Here is the caller graph for this function:



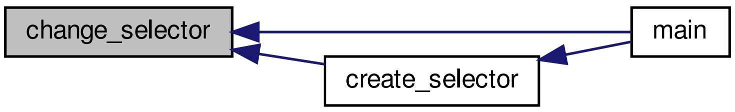
4.18.1.2 change_selector()

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

Here is the call graph for this function:



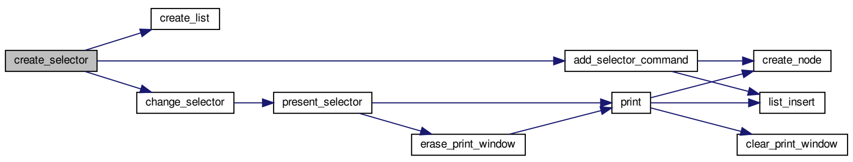
Here is the caller graph for this function:



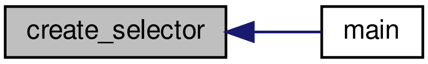
4.18.1.3 create_selector()

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

Here is the call graph for this function:



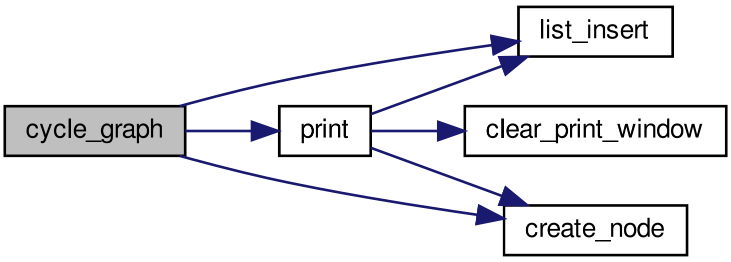
Here is the caller graph for this function:



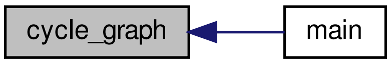
4.18.1.4 cycle_graph()

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

Here is the call graph for this function:



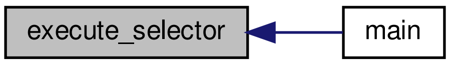
Here is the caller graph for this function:



4.18.1.5 execute_selector()

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

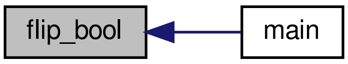
Here is the caller graph for this function:



4.18.1.6 flip_bool()

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

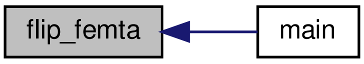
Here is the caller graph for this function:



4.18.1.7 flip_femta()

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

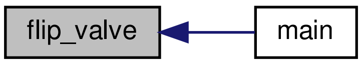
Here is the caller graph for this function:



4.18.1.8 flip_valve()

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

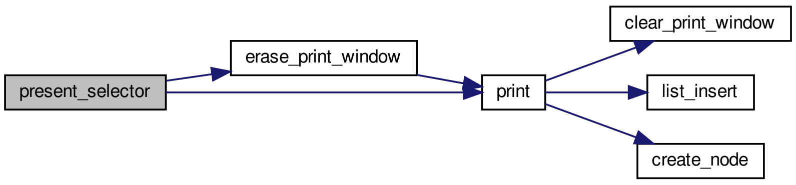
Here is the caller graph for this function:



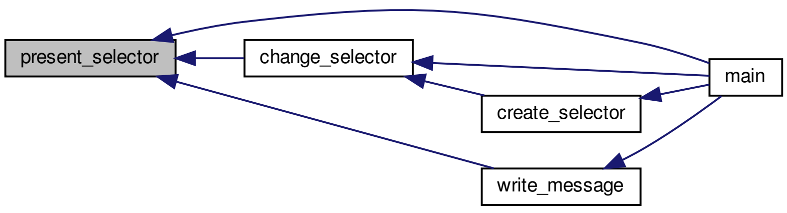
4.18.1.9 present_selector()

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

Here is the call graph for this function:



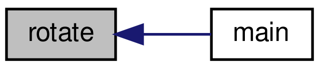
Here is the caller graph for this function:



4.18.1.10 rotate()

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

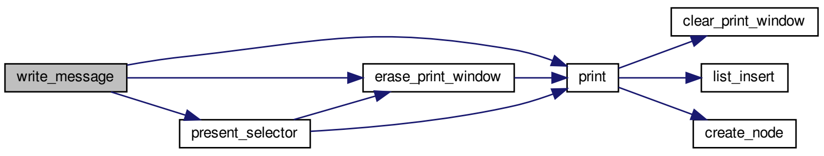
Here is the caller graph for this function:



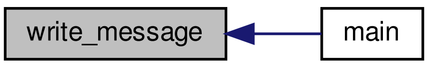
4.18.1.11 write_message()

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

Here is the call graph for this function:

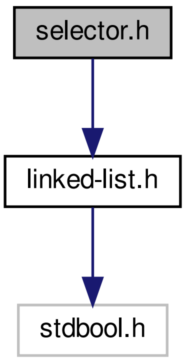


Here is the caller graph for this function:

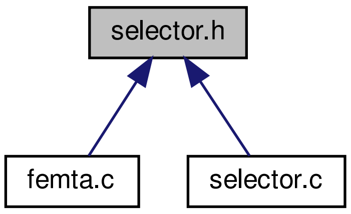


4.19 selector.h File Reference

```
#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

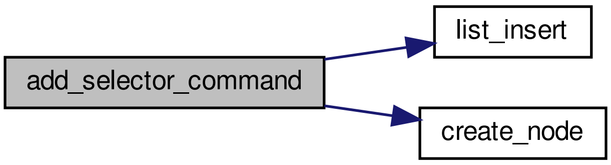
```
typedef struct Selector Selector
```

4.19.2 Function Documentation

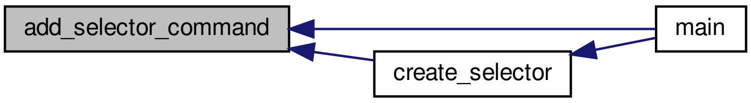
4.19.2.1 add_selector_command()

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

Here is the call graph for this function:



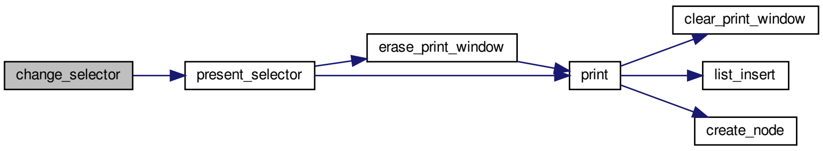
Here is the caller graph for this function:



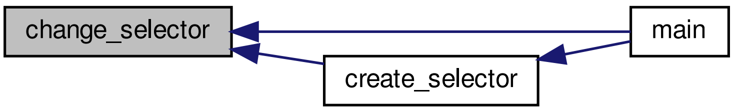
4.19.2.2 change_selector()

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

Here is the call graph for this function:



Here is the caller graph for this function:



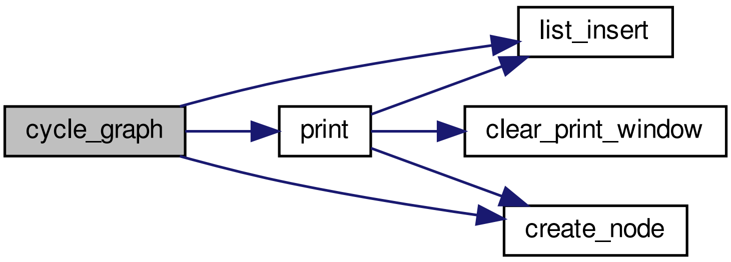
4.19.2.3 create_selector()

```
Selector* create_selector ( )
```

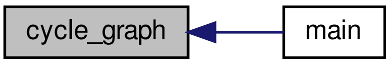
4.19.2.4 cycle_graph()

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

Here is the call graph for this function:



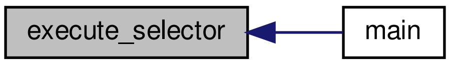
Here is the caller graph for this function:



4.19.2.5 execute_selector()

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

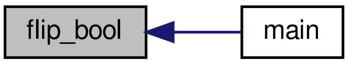
Here is the caller graph for this function:



4.19.2.6 flip_bool()

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

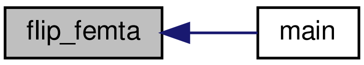
Here is the caller graph for this function:



4.19.2.7 flip_femta()

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

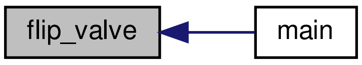
Here is the caller graph for this function:



4.19.2.8 flip_valve()

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

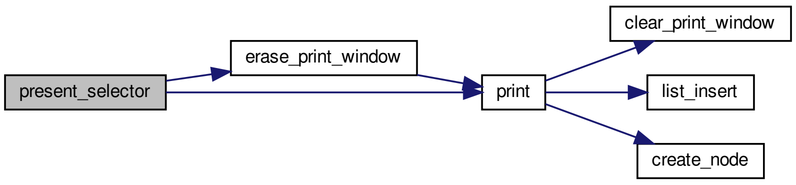
Here is the caller graph for this function:



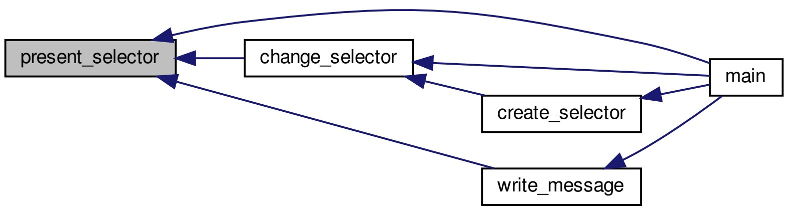
4.19.2.9 present_selector()

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

Here is the call graph for this function:



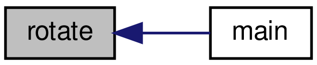
Here is the caller graph for this function:



4.19.2.10 rotate()

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

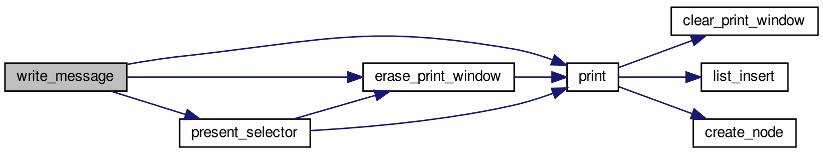
Here is the caller graph for this function:



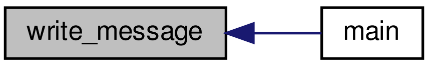
4.19.2.11 write_message()

```
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

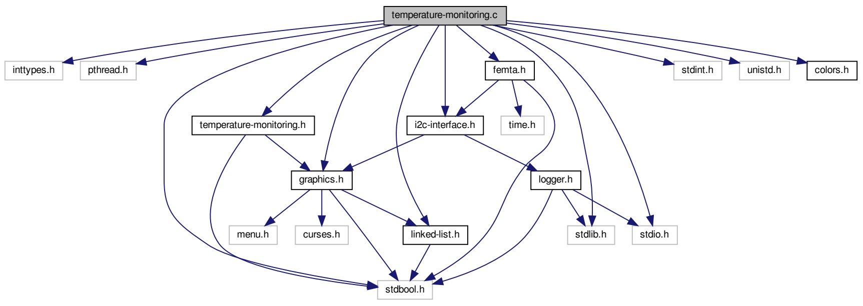
4.19.3.1 visible_selector

```
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

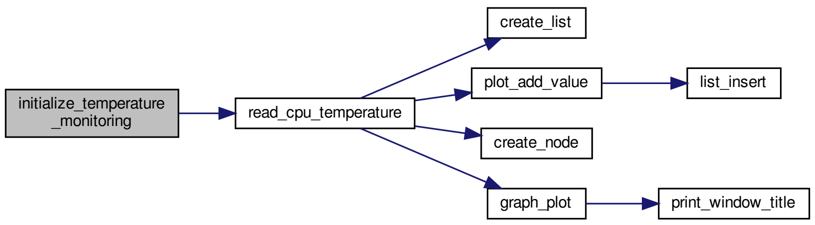
- FILE ∗ cpu_temperature_log_file
- char ∗ temperature_log_filename = "./logs/cpu-temperature-log.txt"
- pthread_t cpu_temperature_thread
- bool termination_signal
- int values_read = 0

4.20.1 Function Documentation

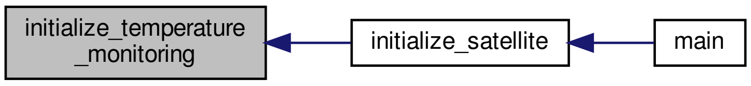
4.20.1.1 initialize_temperature_monitoring()

```
bool initialize_temperature_monitoring ( )
```

Here is the call graph for this function:



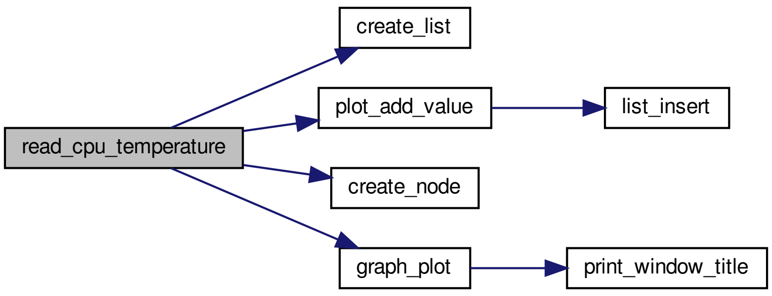
Here is the caller graph for this function:



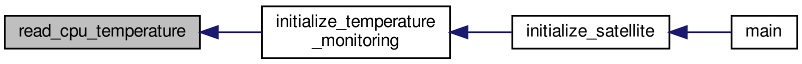
4.20.1.2 read_cpu_temperature()

```
void* read_cpu_temperature ( )
```

Here is the call graph for this function:



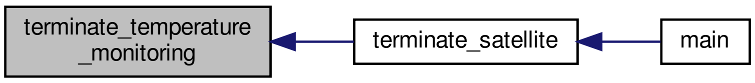
Here is the caller graph for this function:



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
```

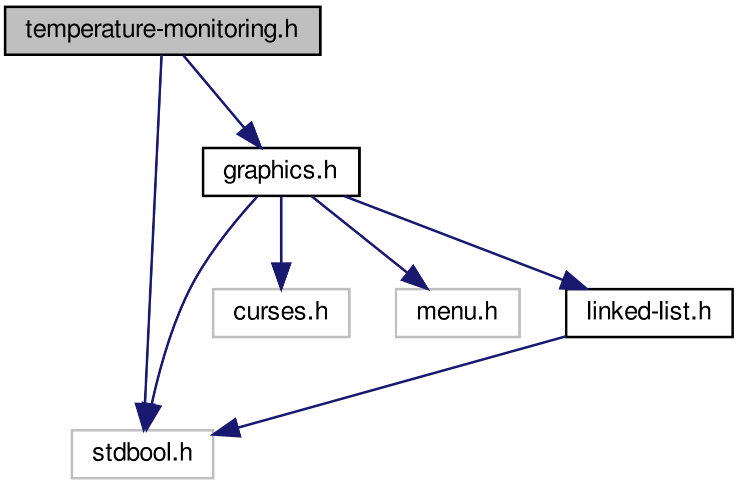
4.20.2.5 values_read

```
int values_read = 0
```

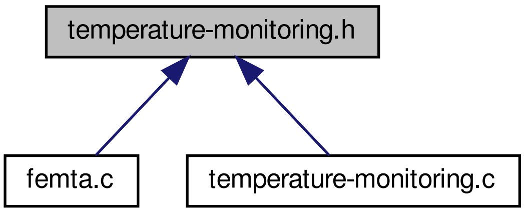
4.21 temperature-monitoring.h File Reference

```
#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

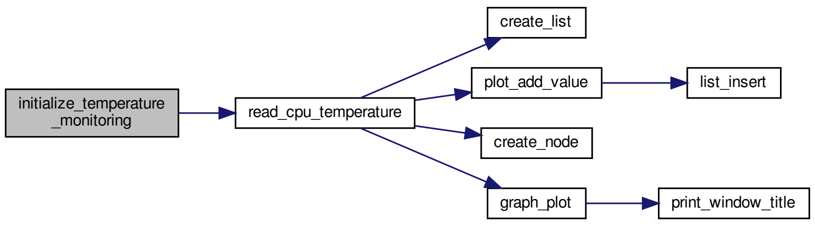
- Plot ∗ temperature_plot

4.21.1 Function Documentation

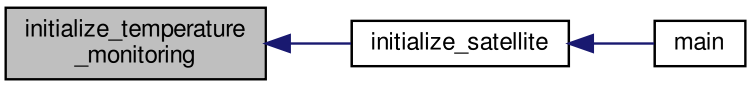
4.21.1.1 initialize_temperature_monitoring()

```
bool initialize_temperature_monitoring ( )
```

Here is the call graph for this function:



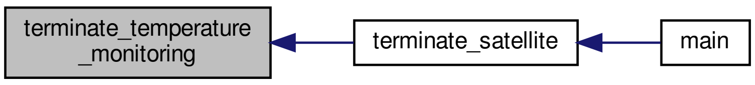
Here is the caller graph for this function:



4.21.1.2 terminate_temperature_monitoring()

```
void terminate_temperature_monitoring ( )
```

Here is the caller graph for this function:



4.21.2 Variable Documentation

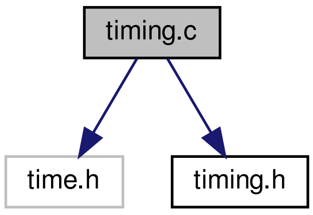
4.21.2.1 temperature_plot

```
Plot* temperature_plot
```

4.22 timing.c File Reference

```
#include <time.h>
#include "timing.h"
```

Include dependency graph for timing.c:



Functions

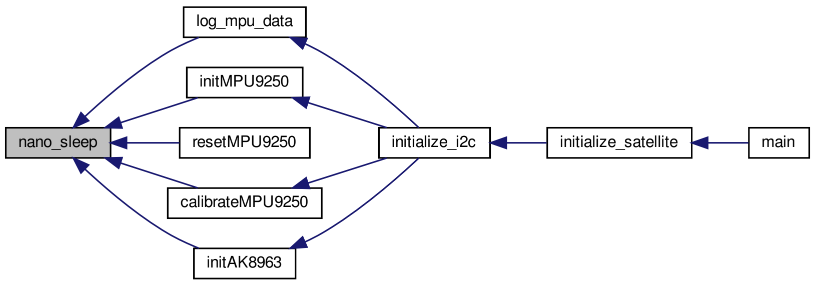
- void nano_sleep (long duration)

4.22.1 Function Documentation

4.22.1.1 nano_sleep()

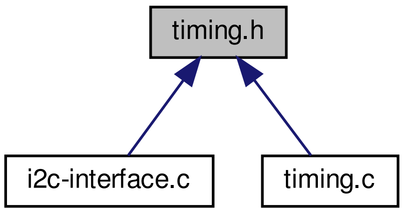
```
void nano_sleep (
             long duration )
```

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

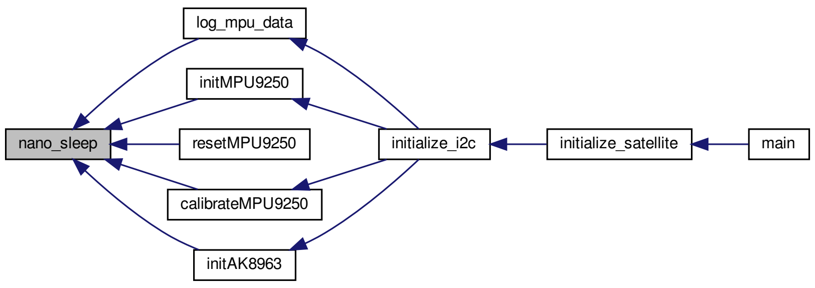
- void nano_sleep (long duration)

4.23.1 Function Documentation

4.23.1.1 nano_sleep()

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

Here is the caller graph for this function:


log_mpu_data
initMPU9250
nano_sleep
resetMPU9250
initialize_i2c
initialize_satellite
main
calibrateMPU9250
initAK8963