MinixTimeKeeper

Generated by Doxygen 1.9.1

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 date_struct Struct Reference

Data structure that holds the date data.

```
#include <utils.h>
```

Data Fields

- uint8_t day

  Day.
- uint8_t month

  Month.
- uint16_t year

  Year.
- uint8_t dayNumber

  Day of the Week.

3.1.1 Detailed Description

Data structure that holds the date data.

Definition at line 34 of file utils.h.

3.1.2 Field Documentation

3.1.2.1 day

```
uint8_t day
```

Day.

Definition at line 35 of file utils.h.

3.1.2.2 dayNumber

```
uint8_t dayNumber
```

Day of the Week.

Definition at line 38 of file utils.h.

3.1.2.3 month

```
uint8_t month
```

Month.

Definition at line 36 of file utils.h.

3.1.2.4 year

```
uint16_t year
```

Year.

Definition at line 37 of file utils.h.

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

- /home/neves/reps/LCOM/shared/proj/src/controller/utils.h

3.2 MouseInfo Struct Reference

Data structure that holds the information about the mouse.

```
#include <utils.h>
```

Data Fields

- int16_t x
- int16_t y

  Mouse X and Y Coordinates.
- uint8_t lb
- uint8_t rb

  Mouse Left and Right Button States.

3.2.1 Detailed Description

Data structure that holds the information about the mouse.

Definition at line 17 of file utils.h.

3.2.2 Field Documentation

3.2.2.1 lb

```
uint8_t lb
```

Definition at line 19 of file utils.h.

3.2.2.2 rb

```
uint8_t rb
```

Mouse Left and Right Button States.

Definition at line 19 of file utils.h.

3.2.2.3 x

```
int16_t x
```

Definition at line 18 of file utils.h.

3.2.2.4 y

```
int16_t y
```

Mouse X and Y Coordinates.

Definition at line 18 of file utils.h.

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

- /home/neves/reps/LCOM/shared/proj/src/controller/utils.h

3.3 Sprite Struct Reference

```
#include <sprite.h>
```

Data Fields

- uint16_t height
- uint16_t width
- uint16_t x
- uint16_t y
- uint32_t * colors
- uint32_t color
- bool pressed

3.3.1 Detailed Description

Definition at line 6 of file sprite.h.

3.3.2 Field Documentation

3.3.2.1 color

```
uint32_t color
```

Definition at line 9 of file sprite.h.

3.3.2.2 colors

```
uint32_t* colors
```

Definition at line 8 of file sprite.h.

3.3.2.3 height

```
uint16_t height
```

Definition at line 7 of file sprite.h.

3.3.2.4 pressed

```
bool pressed
```

Definition at line 10 of file sprite.h.

3.3.2.5 width

```
uint16_t width
```

Definition at line 7 of file sprite.h.

3.3.2.6 x

```
uint16_t x
```

Definition at line 7 of file sprite.h.

3.3.2.7 y

```
uint16_t y
```

Definition at line 7 of file sprite.h.

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

- /home/neves/reps/LCOM/shared/proj/src/model/sprite.h

3.4 time_struct Struct Reference

Data structure that holds the time data.

```
#include <utils.h>
```

Data Fields

- uint8_t hours

  Hours.
- uint8_t minutes

  Minutes.
- uint8_t seconds

  Seconds.

3.4.1 Detailed Description

Data structure that holds the time data.

Definition at line 25 of file utils.h.

3.4.2 Field Documentation

3.4.2.1 hours

```
uint8_t hours
```

Hours.

Definition at line 26 of file utils.h.

3.4.2.2 minutes

```
uint8_t minutes
```

Minutes.

Definition at line 27 of file utils.h.

3.4.2.3 seconds

```
uint8_t seconds
```

Seconds.

Definition at line 28 of file utils.h.

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

- /home/neves/reps/LCOM/shared/proj/src/controller/utils.h

Chapter 4

File Documentation

4.1 /home/neves/reps/LCOM/shared/proj/src/config.h File Reference

Macros

- #define GAME_FREQUENCY 30
- #define VIDEO_MODE 0x115
- #define DOUBLE_BUFFER 1
- #define RED 0xFF0000
- #define GREEN 0x00FF00
- #define BLUE 0x0000FF
- #define DARKBLUE 0x00008B
- #define YELLOW 0xFFFF00
- #define ORANGE 0xFFA500
- #define BLACK 0x000000
- #define WHITE 0xFFFFFF
- #define TRANSPARENT 0xFFFFFFFF
- #define PRESSED 0x888888
- #define Q_KEY 0x10
- #define S_KEY 0x1F
- #define G_KEY 0x22
- #define E_KEY 0x12
- #define C_KEY 0x2E
- #define T_KEY 0x14
- #define BACKSPACE_KEY 0x0E
- #define ONE_KEY 0x02
- #define TWO_KEY 0x03
- #define THREE_KEY 0x04
- #define FOUR_KEY 0x05
- #define FIVE_KEY 0x06
- #define SIX_KEY 0x07
- #define SEVEN_KEY 0x08
- #define EIGHT_KEY 0x09
- #define NINE_KEY 0x0A
- #define ZERO_KEY 0x0B

4.1.1 Macro Definition Documentation

4.1.1.1 BACKSPACE_KEY

```
#define BACKSPACE_KEY 0x0E
```

Definition at line 31 of file config.h.

4.1.1.2 BLACK

```
#define BLACK 0x000000
```

Definition at line 18 of file config.h.

4.1.1.3 BLUE

```
#define BLUE 0x0000FF
```

Definition at line 14 of file config.h.

4.1.1.4 C_KEY

```
#define C_KEY 0x2E
```

Definition at line 29 of file config.h.

4.1.1.5 DARKBLUE

```
#define DARKBLUE 0x00008B
```

Definition at line 15 of file config.h.

4.1.1.6 DOUBLE_BUFFER

```
#define DOUBLE_BUFFER 1
```

Definition at line 8 of file config.h.

4.1.1.7 E_KEY

```
#define E_KEY 0x12
```

Definition at line 28 of file config.h.

4.1.1.8 EIGHT_KEY

```
#define EIGHT_KEY 0x09
```

Definition at line 42 of file config.h.

4.1.1.9 FIVE_KEY

```
#define FIVE_KEY 0x06
```

Definition at line 39 of file config.h.

4.1.1.10 FOUR_KEY

```
#define FOUR_KEY 0x05
```

Definition at line 38 of file config.h.

4.1.1.11 G_KEY

```
#define G_KEY 0x22
```

Definition at line 27 of file config.h.

4.1.1.12 GAME_FREQUENCY

```
#define GAME_FREQUENCY 30
```

Definition at line 6 of file config.h.

4.1.1.13 GREEN

```
#define GREEN 0x00FF00
```

Definition at line 13 of file config.h.

4.1.1.14 NINE_KEY

```
#define NINE_KEY 0x0A
```

Definition at line 43 of file config.h.

4.1.1.15 ONE_KEY

```
#define ONE_KEY 0x02
```

Definition at line 35 of file config.h.

4.1.1.16 ORANGE

```
#define ORANGE 0xFFA500
```

Definition at line 17 of file config.h.

4.1.1.17 PRESSED

```
#define PRESSED 0x888888
```

Definition at line 21 of file config.h.

4.1.1.18 Q_KEY

```
#define Q_KEY 0x10
```

Definition at line 25 of file config.h.

4.1.1.19 RED

```
#define RED 0xFF0000
```

Definition at line 12 of file config.h.

4.1.1.20 S_KEY

```
#define S_KEY 0x1F
```

Definition at line 26 of file config.h.

4.1.1.21 SEVEN_KEY

```
#define SEVEN_KEY 0x08
```

Definition at line 41 of file config.h.

4.1.1.22 SIX_KEY

```
#define SIX_KEY 0x07
```

Definition at line 40 of file config.h.

4.1.1.23 T_KEY

```
#define T_KEY 0x14
```

Definition at line 30 of file config.h.

4.1.1.24 THREE_KEY

```
#define THREE_KEY 0x04
```

Definition at line 37 of file config.h.

4.1.1.25 TRANSPARENT

```
#define TRANSPARENT 0xFFFFFFFF
```

Definition at line 20 of file config.h.

4.1.1.26 TWO_KEY

```
#define TWO_KEY 0x03
```

Definition at line 36 of file config.h.

4.1.1.27 VIDEO_MODE

```
#define VIDEO_MODE 0x115
```

Definition at line 7 of file config.h.

4.1.1.28 WHITE

```
#define WHITE 0xFFFFFF
```

Definition at line 19 of file config.h.

4.1.1.29 YELLOW

```
#define YELLOW 0xFFFF00
```

Definition at line 16 of file config.h.

4.1.1.30 ZERO_KEY

```
#define ZERO_KEY 0x0B
```

Definition at line 44 of file config.h.

4.2 /home/neves/reps/LCOM/shared/proj/src/controller/KBC.c File Reference

```
#include "KBC.h"
```

Functions

- int() read_KBC_output (uint8_t ∗byte, uint8_t mouse)

  Reads the KBC output buffer.
- int() write_to_KBC (uint8_t port, uint8_t command)

  Writes a command to the KBC.

4.2.1 Function Documentation

4.2.1.1 read_KBC_output()

```
int() read_KBC_output (
            uint8_t ∗ byte,
            uint8_t mouse )
```

Reads the KBC output buffer.

Parameters

byte	Pointer to the byte to store the read value
mouse	Indicates if the data is from the mouse (1) or keyboard (0)

Returns

Return 0 upon success and non-zero otherwise

Definition at line 3 of file KBC.c.

4.2.1.2 write_to_KBC()

```
int() write_to_KBC (
            uint8_t port,
            uint8_t command )
```

Writes a command to the KBC.

Parameters

port	The port to write to
command	The command to write

Returns

Return 0 upon success and non-zero otherwise

Definition at line 39 of file KBC.c.

4.3 /home/neves/reps/LCOM/shared/proj/src/controller/KBC.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "utils.h"
```

Macros

- #define KB_IRQ 1

  Keyboard IRQ Line.
- #define MOUSE_IRQ 12

  Mouse IRQ Line.
- #define KB_MASK BIT(1)

  Keyboard IRQ Mask.
- #define MOUSE_MASK BIT(2)

  Mouse IRQ Mask.
- #define KB_DELAY 20000

  Delay Between each Attempt.
- #define MAX_ATTEMPTS 5

  Maximum Number of Attempts.
- #define ESC_BREAKCODE 0x81

  Breakcode for the ESC key.
- #define BREAK_CODE_BIT BIT(7)

  Bit that indicates if a key is being pressed or released.
- #define KBC_STATUS_REG 0x64

  Status Register Register.
- #define KBC_IN_BUF 0x64

Input Buffer Register.

- #define KBC_IN_BUF_ARG 0x60

  Input Buffer Register.
- #define KBC_OUT_BUF 0x60

  Output Buffer Register.
- #define PARITY_ERROR BIT(7)

  Parity Error Bit.
- #define TIMEOUT_ERROR BIT(6)

  Timeout Error Bit.
- #define MOUSE_DATA_BIT BIT(5)

  Mouse Data Available Bit.
- #define FULL_IN_BUF BIT(1)

  Full Input Buffer Bit.
- #define FULL_OUT_BUF BIT(0)

  Full Output Buffer Bit.
- #define KBC_READ_CMD 0x20

  Read Command for the KBC.
- #define KBC_WRITE_CMD 0x60

  Write Command for the KBC.
- #define KBC_WRITE_MOUSE 0xD4

  Write Command for the Mouse.
- #define KBC_KB_INT BIT(0)

  Keyboard Interrupt Enable Bit.
- #define KBC_MOUSE_INT BIT(1)

  Mouse Interrupt Enable Bit.
- #define MOUSE_ACK 0xFA

  Mouse Acknowledgment Byte.
- #define MOUSE_NACK 0xFE

  Mouse Error Byte.
- #define MOUSE_DATA_STREAM_MODE 0xEA

  Set Stream Mode.
- #define MOUSE_DATA_REPORT_ENABLE 0xF4

  Enable Data Reporting.
- #define MOUSE_DATA_REPORT_DISABLE 0xF5

  Disable Data Reporting.
- #define MOUSE_OVERFLOW_Y BIT(7)

  Y Overflow.
- #define MOUSE_OVERFLOW_X BIT(6)

  X Overflow.
- #define MOUSE_SIGNAL_Y BIT(5)

  Y Sign.
- #define MOUSE_SIGNAL_X BIT(4)

  X Sign.
- #define FIRST_BYTE BIT(3)

  First Byte.
- #define MOUSE_MB BIT(2)

  Middle Button.
- #define MOUSE_RB BIT(1)

  Right Button.
- #define MOUSE_LB BIT(0)

  Left Button.

Functions

- int() read_KBC_output (uint8_t ∗byte, uint8_t mouse)

  Reads the KBC output buffer.
- int() write_to_KBC (uint8_t port, uint8_t command)

  Writes a command to the KBC.

4.3.1 Macro Definition Documentation

4.3.1.1 BREAK_CODE_BIT

```
#define BREAK_CODE_BIT BIT(7)
```

Bit that indicates if a key is being pressed or released.

Definition at line 24 of file KBC.h.

4.3.1.2 ESC_BREAKCODE

```
#define ESC_BREAKCODE 0x81
```

Breakcode for the ESC key.

Definition at line 23 of file KBC.h.

4.3.1.3 FIRST_BYTE

```
#define FIRST_BYTE BIT(3)
```

First Byte.

Definition at line 65 of file KBC.h.

4.3.1.4 FULL_IN_BUF

```
#define FULL_IN_BUF BIT(1)
```

Full Input Buffer Bit.

Definition at line 38 of file KBC.h.

4.3.1.5 FULL_OUT_BUF

```
#define FULL_OUT_BUF BIT(0)
```

Full Output Buffer Bit.

Definition at line 39 of file KBC.h.

4.3.1.6 KB_DELAY

```
#define KB_DELAY 20000
```

Delay Between each Attempt.

Definition at line 20 of file KBC.h.

4.3.1.7 KB_IRQ

```
#define KB_IRQ 1
```

Keyboard IRQ Line.

Definition at line 14 of file KBC.h.

4.3.1.8 KB_MASK

```
#define KB_MASK BIT(1)
```

Keyboard IRQ Mask.

Definition at line 17 of file KBC.h.

4.3.1.9 KBC_IN_BUF

```
#define KBC_IN_BUF 0x64
```

Input Buffer Register.

Definition at line 29 of file KBC.h.

4.3.1.10 KBC_IN_BUF_ARG

```
#define KBC_IN_BUF_ARG 0x60
```

Input Buffer Register.

Definition at line 30 of file KBC.h.

4.3.1.11 KBC_KB_INT

```
#define KBC_KB_INT BIT(0)
```

Keyboard Interrupt Enable Bit.

Definition at line 49 of file KBC.h.

4.3.1.12 KBC_MOUSE_INT

```
#define KBC_MOUSE_INT BIT(1)
```

Mouse Interrupt Enable Bit.

Definition at line 50 of file KBC.h.

4.3.1.13 KBC_OUT_BUF

```
#define KBC_OUT_BUF 0x60
```

Output Buffer Register.

Definition at line 31 of file KBC.h.

4.3.1.14 KBC_READ_CMD

```
#define KBC_READ_CMD 0x20
```

Read Command for the KBC.

Definition at line 43 of file KBC.h.

4.3.1.15 KBC_STATUS_REG

```
#define KBC_STATUS_REG 0x64
```

Status Register Register.

Definition at line 28 of file KBC.h.

4.3.1.16 KBC_WRITE_CMD

```
#define KBC_WRITE_CMD 0x60
```

Write Command for the KBC.

Definition at line 44 of file KBC.h.

4.3.1.17 KBC_WRITE_MOUSE

```
#define KBC_WRITE_MOUSE 0xD4
```

Write Command for the Mouse.

Definition at line 45 of file KBC.h.

4.3.1.18 MAX_ATTEMPTS

```
#define MAX_ATTEMPTS 5
```

Maximum Number of Attempts.

Definition at line 21 of file KBC.h.

4.3.1.19 MOUSE_ACK

```
#define MOUSE_ACK 0xFA
```

Mouse Acknowledgment Byte.

Definition at line 54 of file KBC.h.

4.3.1.20 MOUSE_DATA_BIT

```
#define MOUSE_DATA_BIT BIT(5)
```

Mouse Data Available Bit.

Definition at line 37 of file KBC.h.

4.3.1.21 MOUSE_DATA_REPORT_DISABLE

```
#define MOUSE_DATA_REPORT_DISABLE 0xF5
```

Disable Data Reporting.

Definition at line 59 of file KBC.h.

4.3.1.22 MOUSE_DATA_REPORT_ENABLE

```
#define MOUSE_DATA_REPORT_ENABLE 0xF4
```

Enable Data Reporting.

Definition at line 58 of file KBC.h.

4.3.1.23 MOUSE_DATA_STREAM_MODE

```
#define MOUSE_DATA_STREAM_MODE 0xEA
```

Set Stream Mode.

Definition at line 57 of file KBC.h.

4.3.1.24 MOUSE_IRQ

```
#define MOUSE_IRQ 12
```

Mouse IRQ Line.

Definition at line 15 of file KBC.h.

4.3.1.25 MOUSE_LB

```
#define MOUSE_LB BIT(0)
```

Left Button.

Definition at line 68 of file KBC.h.

4.3.1.26 MOUSE_MASK

```
#define MOUSE_MASK BIT(2)
```

Mouse IRQ Mask.

Definition at line 18 of file KBC.h.

4.3.1.27 MOUSE_MB

```
#define MOUSE_MB BIT(2)
```

Middle Button.

Definition at line 66 of file KBC.h.

4.3.1.28 MOUSE_NACK

```
#define MOUSE_NACK 0xFE
```

Mouse Error Byte.

Definition at line 55 of file KBC.h.

4.3.1.29 MOUSE_OVERFLOW_X

```
#define MOUSE_OVERFLOW_X BIT(6)
```

X Overflow.

Definition at line 62 of file KBC.h.

4.3.1.30 MOUSE_OVERFLOW_Y

```
#define MOUSE_OVERFLOW_Y BIT(7)
```

Y Overflow.

Definition at line 61 of file KBC.h.

4.3.1.31 MOUSE_RB

```
#define MOUSE_RB BIT(1)
```

Right Button.

Definition at line 67 of file KBC.h.

4.3.1.32 MOUSE_SIGNAL_X

```
#define MOUSE_SIGNAL_X BIT(4)
```

X Sign.

Definition at line 64 of file KBC.h.

4.3.1.33 MOUSE_SIGNAL_Y

```
#define MOUSE_SIGNAL_Y BIT(5)
```

Y Sign.

Definition at line 63 of file KBC.h.

4.3.1.34 PARITY_ERROR

```
#define PARITY_ERROR BIT(7)
```

Parity Error Bit.

Definition at line 35 of file KBC.h.

4.3.1.35 TIMEOUT_ERROR

```
#define TIMEOUT_ERROR BIT(6)
```

Timeout Error Bit.

Definition at line 36 of file KBC.h.

4.3.2 Function Documentation

4.3.2.1 read_KBC_output()

```
int() read_KBC_output (
            uint8_t * byte,
            uint8_t mouse )
```

Reads the KBC output buffer.

Parameters

byte	Pointer to the byte to store the read value
mouse	Indicates if the data is from the mouse (1) or keyboard (0)

Returns

Return 0 upon success and non-zero otherwise

Definition at line 3 of file KBC.c.

4.3.2.2 write_to_KBC()

```
int() write_to_KBC (
            uint8_t port,
            uint8_t command )
```

Writes a command to the KBC.

Parameters

port	The port to write to
command	The command to write

Returns

Return 0 upon success and non-zero otherwise

Definition at line 39 of file KBC.c.

4.4 /home/neves/reps/↩ LCOM/shared/proj/src/controller/keyboard/keyboard.c File Reference

```
#include "keyboard.h"
```

Functions

- int() keyboard_subscribe_int ()

  Subscribes the Keyboard interrupts.
- int() keyboard_unsubscribe_int ()

  Unsubscribes the Keyboard interrupts.
- void() kbc_ih ()

  Keyboard Interrupt Handler.

Variables

- int kb_hook_id = 1
- uint8_t scancode

4.4.1 Function Documentation

4.4.1.1 kbc_ih()

```
void() kbc_ih ( )
```

Keyboard Interrupt Handler.

Reads the KBC's output buffer and stores the scancode

Definition at line 16 of file keyboard.c.

4.4.1.2 keyboard_subscribe_int()

```
int() keyboard_subscribe_int ( )
```

Subscribes the Keyboard interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 6 of file keyboard.c.

4.4.1.3 keyboard_unsubscribe_int()

```
int() keyboard_unsubscribe_int ( )
```

Unsubscribes the Keyboard interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file keyboard.c.

4.4.2 Variable Documentation

4.4.2.1 kb_hook_id

```
int kb_hook_id = 1
```

Definition at line 3 of file keyboard.c.

4.4.2.2 scancode

```
uint8_t scancode
```

Definition at line 4 of file keyboard.c.

4.5 /home/neves/reps/↩ LCOM/shared/proj/src/controller/keyboard/keyboard.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "../KBC.h"
```

Functions

- int() keyboard_subscribe_int ()

  Subscribes the Keyboard interrupts.
- int() keyboard_unsubscribe_int ()

  Unsubscribes the Keyboard interrupts.
- void() kbc_ih ()

  Keyboard Interrupt Handler.

4.5.1 Function Documentation

4.5.1.1 kbc_ih()

```
void() kbc_ih ( )
```

Keyboard Interrupt Handler.

Reads the KBC's output buffer and stores the scancode

Definition at line 16 of file keyboard.c.

4.5.1.2 keyboard_subscribe_int()

```
int() keyboard_subscribe_int ( )
```

Subscribes the Keyboard interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 6 of file keyboard.c.

4.5.1.3 keyboard_unsubscribe_int()

```
int() keyboard_unsubscribe_int ( )
```

Unsubscribes the Keyboard interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file keyboard.c.

4.6 /home/neves/reps/LCOM/shared/proj/src/controller/mouse/mouse.c File Reference

```
#include "mouse.h"
```

Functions

- int() mouse_subscribe_int ()

  Subscribes the Mouse interrupts.
- int() mouse_unsubscribe_int ()

  Unsubscribes the Mouse interrupts.
- void() mouse_ih ()

  Mouse Interrupt Handler.
- void() mouse_sync ()

  Synchronizes the mouse data.
- void() update_mouse_info ()

  Builds the MouseInfo structure.
- int() mouse_write_command (uint8_t command)

  Writes a command to the mouse.

Variables

- int mouse_hook_id = 2
- uint8_t byte_index = 0
- uint8_t mouse_byte
- uint8_t mouse_data [3]
- MouseInfo mouse_info
- vbe_mode_info_t vbe_info

4.6.1 Function Documentation

4.6.1.1 mouse_ih()

```
void() mouse_ih ( )
```

Mouse Interrupt Handler.

Reads the KBC's output buffer and stores the scancode

Definition at line 20 of file mouse.c.

4.6.1.2 mouse_subscribe_int()

```
int() mouse_subscribe_int ( )
```

Subscribes the Mouse interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 10 of file mouse.c.

4.6.1.3 mouse_sync()

```
void() mouse_sync ( )
```

Synchronizes the mouse data.

Stores the mouse data in the mouse_data array

Definition at line 25 of file mouse.c.

4.6.1.4 mouse_unsubscribe_int()

```
int() mouse_unsubscribe_int ( )
```

Unsubscribes the Mouse interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 15 of file mouse.c.

4.6.1.5 mouse_write_command()

```
int() mouse_write_command (
             uint8_t command )
```

Writes a command to the mouse.

Parameters

command	The command to be sent to the mouse

Returns

Return 0 upon success and non-zero otherwise

Definition at line 52 of file mouse.c.

4.6.1.6 update_mouse_info()

```
void() update_mouse_info ( )
```

Builds the MouseInfo structure.

Builds the MouseInfo structure based on the mouse data

Definition at line 35 of file mouse.c.

4.6.2 Variable Documentation

4.6.2.1 byte_index

```
uint8_t byte_index = 0
```

Definition at line 4 of file mouse.c.

4.6.2.2 mouse_byte

```
uint8_t mouse_byte
```

Definition at line 5 of file mouse.c.

4.6.2.3 mouse_data

```
uint8_t mouse_data[3]
```

Definition at line 6 of file mouse.c.

4.6.2.4 mouse_hook_id

```
int mouse_hook_id = 2
```

Definition at line 3 of file mouse.c.

4.6.2.5 mouse_info

```
MouseInfo mouse_info
```

Definition at line 7 of file mouse.c.

4.6.2.6 vbe_info

```
vbe_mode_info_t vbe_info  [extern]
```

Definition at line 3 of file graphic.c.

4.7 /home/neves/reps/LCOM/shared/proj/src/controller/mouse/mouse.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "../KBC.h"
#include "../video/graphic.h"
```

Functions

- int() mouse_subscribe_int ()

  Subscribes the Mouse interrupts.
- int() mouse_unsubscribe_int ()

  Unsubscribes the Mouse interrupts.
- void() mouse_ih ()

  Mouse Interrupt Handler.
- void() mouse_sync ()

  Synchronizes the mouse data.
- void() update_mouse_info ()

  Builds the MouseInfo structure.
- int() mouse_write_command (uint8_t command)

  Writes a command to the mouse.

4.7.1 Function Documentation

4.7.1.1 mouse_ih()

```
void() mouse_ih ( )
```

Mouse Interrupt Handler.

Reads the KBC's output buffer and stores the scancode

Definition at line 20 of file mouse.c.

4.7.1.2 mouse_subscribe_int()

```
int() mouse_subscribe_int ( )
```

Subscribes the Mouse interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 10 of file mouse.c.

4.7.1.3 mouse_sync()

```
void() mouse_sync ( )
```

Synchronizes the mouse data.

Stores the mouse data in the mouse_data array

Definition at line 25 of file mouse.c.

4.7.1.4 mouse_unsubscribe_int()

```
int() mouse_unsubscribe_int ( )
```

Unsubscribes the Mouse interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 15 of file mouse.c.

4.7.1.5 mouse_write_command()

```
int() mouse_write_command (
             uint8_t command )
```

Writes a command to the mouse.

Parameters

command	The command to be sent to the mouse

Returns

Return 0 upon success and non-zero otherwise

Definition at line 52 of file mouse.c.

4.7.1.6 update_mouse_info()

```
void() update_mouse_info ( )
```

Builds the MouseInfo structure.

Builds the MouseInfo structure based on the mouse data

Definition at line 35 of file mouse.c.

4.8 /home/neves/reps/LCOM/shared/proj/src/controller/rtc/rtc.c File Reference

```
#include "rtc.h"
```

Functions

- int() rtc_subscribe_int ()
  Subscribes the RTC's interrupts.
- int() rtc_unsubscribe_int ()
  Unsubscribes the RTC's interrupts.
- void() rtc_ih ()
  RTC Interrupt Handler.
- int() rtc_start ()
  Starts the RTC.
- int() rtc_stop ()
  Stops the RTC.
- int() rtc_update ()
  Updates the RTC's data structures.
- int() rtc_set_alarm (time_struct alarm_time)
  Sets the RTC's alarm.
- int() rtc_disable_alarm ()
  Disables the RTC's alarm.
- int() rtc_read (uint8_t port, uint8_t ∗output)
  Reads a RTC register.
- int() rtc_write (uint8_t port, uint8_t value)
  Writes to a RTC register.
- uint8_t bcd_to_bin (uint8_t bcd)
  Converts a BCD number to binary.
- uint8_t bin_to_bcd (uint8_t bin)
  Converts a binary number to BCD.
- void convert_to_24h ()
  Converts the RTC's time data to 24h format.

Variables

- int rtc_hook_id = 3
- uint8_t rtc_mode
- uint8_t rtc_time_format
- uint8_t rtc_int_cause
- uint8_t rtc_original_config
- time_struct rtc_time
- date_struct rtc_date

4.8.1 Function Documentation

4.8.1.1 bcd_to_bin()

```
uint8_t bcd_to_bin (
             uint8_t bcd )
```

Converts a BCD number to binary.

Parameters

bcd	BCD number to convert

Returns

Converted binary number

Definition at line 167 of file rtc.c.

4.8.1.2 bin_to_bcd()

```
uint8_t bin_to_bcd (
             uint8_t bin )
```

Converts a binary number to BCD.

Parameters

bin	Binary number to convert

Returns

Converted BCD number

Definition at line 172 of file rtc.c.

4.8.1.3 convert_to_24h()

```
void convert_to_24h ( )
```

Converts the RTC's time data to 24h format.

Definition at line 177 of file rtc.c.

4.8.1.4 rtc_disable_alarm()

```
int() rtc_disable_alarm ( )
```

Disables the RTC's alarm.

Disables the RTC's alarm interrupt

Returns

Return 0 upon success and non-zero otherwise

Definition at line 137 of file rtc.c.

4.8.1.5 rtc_ih()

```
void() rtc_ih ( )
```

RTC Interrupt Handler.

Reads the RTC's C Register and performs the necessary actions given the interrupt cause

Definition at line 21 of file rtc.c.

4.8.1.6 rtc_read()

```
int() rtc_read (
             uint8_t port,
             uint8_t * output )
```

Reads a RTC register.

Reads a given RTC register and stores the output in the given pointer

Parameters

port	Address to be read from
output	Pointer to store the output

Returns

Return 0 upon success and non-zero otherwise

Definition at line 149 of file rtc.c.

4.8.1.7 rtc_set_alarm()

```
int() rtc_set_alarm (
            time_struct alarm_time )
```

Sets the RTC's alarm.

Enables the RTC's alarm interrupt and sets the alarm time

Returns

Return 0 upon success and non-zero otherwise

Definition at line 99 of file rtc.c.

4.8.1.8 rtc_start()

```
int() rtc_start ( )
```

Starts the RTC.

Reads the RTC's modes and sets the RTC's data structures

Returns

Return 0 upon success and non-zero otherwise

Definition at line 32 of file rtc.c.

4.8.1.9 rtc_stop()

```
int() rtc_stop ( )
```

Stops the RTC.

Restores the RTC's original configuration

Returns

Return 0 upon success and non-zero otherwise

Definition at line 48 of file rtc.c.

4.8.1.10 rtc_subscribe_int()

```
int() rtc_subscribe_int ( )
```

Subscribes the RTC's interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file rtc.c.

4.8.1.11 rtc_unsubscribe_int()

```
int() rtc_unsubscribe_int ( )
```

Unsubscribes the RTC's interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 16 of file rtc.c.

4.8.1.12 rtc_update()

```
int() rtc_update ( )
```

Updates the RTC's data structures.

Reads the RTC's date and time and updates the RTC structures

Returns

Return 0 upon success and non-zero otherwise

Definition at line 53 of file rtc.c.

4.8.1.13 rtc_write()

```
int() rtc_write (
             uint8_t port,
             uint8_t value )
```

Writes to a RTC register.

Writes a byte to a given RTC register

Parameters

port	Address to be written to
value	Value to be written

Returns

Return 0 upon success and non-zero otherwise

Definition at line 158 of file rtc.c.

4.8.2 Variable Documentation

4.8.2.1 rtc_date

```
date_struct rtc_date
```

Definition at line 9 of file rtc.c.

4.8.2.2 rtc_hook_id

```
int rtc_hook_id = 3
```

Definition at line 3 of file rtc.c.

4.8.2.3 rtc_int_cause

```
uint8_t rtc_int_cause
```

Definition at line 6 of file rtc.c.

4.8.2.4 rtc_mode

```
uint8_t rtc_mode
```

Definition at line 4 of file rtc.c.

4.8.2.5 rtc_original_config

```
uint8_t rtc_original_config
```

Definition at line 7 of file rtc.c.

4.8.2.6 rtc_time

```
time_struct rtc_time
```

Definition at line 8 of file rtc.c.

4.8.2.7 rtc_time_format

```
uint8_t rtc_time_format
```

Definition at line 5 of file rtc.c.

4.9 /home/neves/reps/LCOM/shared/proj/src/controller/rtc/rtc.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "../utils.h"
```

Macros

- #define RTC_IRQ 8

  RTC IRQ Line.
- #define RTC_MASK BIT(3)

  RTC Interrupt Mask.
- #define RTC_DELAY 1000

  Standard Delay Value.
- #define RTC_MAX_ATTEMPTS 5

  Maximum Number of Attempts.
- #define RTC_ADDR_REG 0x70

  RTC Address Register Address.
- #define RTC_DATA_REG 0x71

  RTC Data Input/Output Register Address.
- #define RTC_SECONDS 0

  RTC Seconds Register Number.
- #define RTC_SECONDS_ALARM 1

  RTC Seconds Alarm Register Number.

- #define RTC_MINUTES 2

  RTC Minutes Register Number.
- #define RTC_MINUTES_ALARM 3

  RTC Minutes Alarm Register Number.
- #define RTC_HOURS 4

  RTC Hours Register Number.
- #define RTC_HOURS_ALARM 5

  RTC Hours Alarm Register Number.
- #define RTC_DAY_OF_WEEK 6

  RTC Day of Week Register Number.
- #define RTC_DAY_OF_MONTH 7

  RTC Day of Month Register Number.
- #define RTC_MONTH 8

  RTC Month Register Number.
- #define RTC_YEAR 9

  RTC Year Register Number.
- #define RTC_REG_A 10

  RTC Register A Number.
- #define RTC_REG_B 11

  RTC Register B Number.
- #define RTC_REG_C 12

  RTC Register C Number.
- #define RTC_REG_D 13

  RTC Register D Number.
- #define RTC_UIP BIT(7)

  RTC Update In Progress Bit.
- #define RTC_SET BIT(7)

  RTC Inhibit Update Bit.
- #define RTC_PIE BIT(6)

  RTC Periodic Interrupt Enable Bit.
- #define RTC_AIE BIT(5)

  RTC Alarm Interrupt Enable Bit.
- #define RTC_UIE BIT(4)

  RTC Update Interrupt Enable Bit.
- #define RTC_DM BIT(2)

  RTC Data Mode Bit.
- #define RTC_24HR BIT(1)

  RTC 24 Hour Mode Bit.
- #define RTC_IRQF BIT(7)

  RTC Interrupt Flag Bit.
- #define RTC_PF BIT(6)

  RTC Periodic Flag Bit.
- #define RTC_AF BIT(5)

  RTC Alarm Flag Bit.
- #define RTC_UF BIT(4)

  RTC Update Flag Bit.

Functions

- int() rtc_subscribe_int ()
  Subscribes the RTC's interrupts.
- int() rtc_unsubscribe_int ()
  Unsubscribes the RTC's interrupts.
- void() rtc_ih ()
  RTC Interrupt Handler.
- int() rtc_start ()
  Starts the RTC.
- int() rtc_stop ()
  Stops the RTC.
- int() rtc_update ()
  Updates the RTC's data structures.
- int() rtc_set_alarm (time_struct alarm_time)
  Sets the RTC's alarm.
- int() rtc_disable_alarm ()
  Disables the RTC's alarm.
- int() rtc_read (uint8_t port, uint8_t ∗output)
  Reads a RTC register.
- int() rtc_write (uint8_t port, uint8_t value)
  Writes to a RTC register.
- uint8_t bcd_to_bin (uint8_t bcd)
  Converts a BCD number to binary.
- uint8_t bin_to_bcd (uint8_t bin)
  Converts a binary number to BCD.
- void convert_to_24h ()
  Converts the RTC's time data to 24h format.

4.9.1 Macro Definition Documentation

4.9.1.1 RTC_24HR

```
#define RTC_24HR BIT(1)
```

RTC 24 Hour Mode Bit.

Definition at line 54 of file rtc.h.

4.9.1.2 RTC_ADDR_REG

```
#define RTC_ADDR_REG 0x70
```

RTC Address Register Address.

Definition at line 22 of file rtc.h.

4.9.1.3 RTC_AF

```
#define RTC_AF BIT(5)
```

RTC Alarm Flag Bit.

Definition at line 59 of file rtc.h.

4.9.1.4 RTC_AIE

```
#define RTC_AIE BIT(5)
```

RTC Alarm Interrupt Enable Bit.

Definition at line 51 of file rtc.h.

4.9.1.5 RTC_DATA_REG

```
#define RTC_DATA_REG 0x71
```

RTC Data Input/Output Register Address.

Definition at line 23 of file rtc.h.

4.9.1.6 RTC_DAY_OF_MONTH

```
#define RTC_DAY_OF_MONTH 7
```

RTC Day of Month Register Number.

Definition at line 34 of file rtc.h.

4.9.1.7 RTC_DAY_OF_WEEK

```
#define RTC_DAY_OF_WEEK 6
```

RTC Day of Week Register Number.

Definition at line 33 of file rtc.h.

4.9.1.8 RTC_DELAY

```
#define RTC_DELAY 1000
```

Standard Delay Value.

Definition at line 17 of file rtc.h.

4.9.1.9 RTC_DM

```
#define RTC_DM BIT(2)
```

RTC Data Mode Bit.

Definition at line 53 of file rtc.h.

4.9.1.10 RTC_HOURS

```
#define RTC_HOURS 4
```

RTC Hours Register Number.

Definition at line 31 of file rtc.h.

4.9.1.11 RTC_HOURS_ALARM

```
#define RTC_HOURS_ALARM 5
```

RTC Hours Alarm Register Number.

Definition at line 32 of file rtc.h.

4.9.1.12 RTC_IRQ

```
#define RTC_IRQ 8
```

RTC IRQ Line.

Definition at line 14 of file rtc.h.

4.9.1.13 RTC_IRQF

```
#define RTC_IRQF BIT(7)
```

RTC Interrupt Flag Bit.

Definition at line 57 of file rtc.h.

4.9.1.14 RTC_MASK

```
#define RTC_MASK BIT(3)
```

RTC Interrupt Mask.

Definition at line 15 of file rtc.h.

4.9.1.15 RTC_MAX_ATTEMPTS

```
#define RTC_MAX_ATTEMPTS 5
```

Maximum Number of Attempts.

Definition at line 18 of file rtc.h.

4.9.1.16 RTC_MINUTES

```
#define RTC_MINUTES 2
```

RTC Minutes Register Number.

Definition at line 29 of file rtc.h.

4.9.1.17 RTC_MINUTES_ALARM

```
#define RTC_MINUTES_ALARM 3
```

RTC Minutes Alarm Register Number.

Definition at line 30 of file rtc.h.

4.9.1.18 RTC_MONTH

```
#define RTC_MONTH 8
```

RTC Month Register Number.

Definition at line 35 of file rtc.h.

4.9.1.19 RTC_PF

```
#define RTC_PF BIT(6)
```

RTC Periodic Flag Bit.

Definition at line 58 of file rtc.h.

4.9.1.20 RTC_PIE

```
#define RTC_PIE BIT(6)
```

RTC Periodic Interrupt Enable Bit.

Definition at line 50 of file rtc.h.

4.9.1.21 RTC_REG_A

```
#define RTC_REG_A 10
```

RTC Register A Number.

Definition at line 38 of file rtc.h.

4.9.1.22 RTC_REG_B

```
#define RTC_REG_B 11
```

RTC Register B Number.

Definition at line 39 of file rtc.h.

4.9.1.23 RTC_REG_C

```
#define RTC_REG_C 12
```

RTC Register C Number.

Definition at line 40 of file rtc.h.

4.9.1.24 RTC_REG_D

```
#define RTC_REG_D 13
```

RTC Register D Number.

Definition at line 41 of file rtc.h.

4.9.1.25 RTC_SECONDS

```
#define RTC_SECONDS 0
```

RTC Seconds Register Number.

Definition at line 27 of file rtc.h.

4.9.1.26 RTC_SECONDS_ALARM

```
#define RTC_SECONDS_ALARM 1
```

RTC Seconds Alarm Register Number.

Definition at line 28 of file rtc.h.

4.9.1.27 RTC_SET

```
#define RTC_SET BIT(7)
```

RTC Inhibit Update Bit.

Definition at line 49 of file rtc.h.

4.9.1.28 RTC_UF

```
#define RTC_UF BIT(4)
```

RTC Update Flag Bit.

Definition at line 60 of file rtc.h.

4.9.1.29 RTC_UIE

```
#define RTC_UIE BIT(4)
```

RTC Update Interrupt Enable Bit.

Definition at line 52 of file rtc.h.

4.9.1.30 RTC_UIP

```
#define RTC_UIP BIT(7)
```

RTC Update In Progress Bit.

Definition at line 46 of file rtc.h.

4.9.1.31 RTC_YEAR

```
#define RTC_YEAR 9
```

RTC Year Register Number.

Definition at line 36 of file rtc.h.

4.9.2 Function Documentation

4.9.2.1 bcd_to_bin()

```
uint8_t bcd_to_bin (
            uint8_t bcd )
```

Converts a BCD number to binary.

Parameters

bcd	BCD number to convert

Returns

Converted binary number

Definition at line 167 of file rtc.c.

4.9.2.2 bin_to_bcd()

```
uint8_t bin_to_bcd (
            uint8_t bin )
```

Converts a binary number to BCD.

Parameters

bin	Binary number to convert

Returns

Converted BCD number

Definition at line 172 of file rtc.c.

4.9.2.3 convert_to_24h()

```
void convert_to_24h ( )
```

Converts the RTC's time data to 24h format.

Definition at line 177 of file rtc.c.

4.9.2.4 rtc_disable_alarm()

```
int() rtc_disable_alarm ( )
```

Disables the RTC's alarm.

Disables the RTC's alarm interrupt

Returns

Return 0 upon success and non-zero otherwise

Definition at line 137 of file rtc.c.

4.9.2.5 rtc_ih()

```
void() rtc_ih ( )
```

RTC Interrupt Handler.

Reads the RTC's C Register and performs the necessary actions given the interrupt cause

Definition at line 21 of file rtc.c.

4.9.2.6 rtc_read()

```
int() rtc_read (
            uint8_t port,
            uint8_t * output )
```

Reads a RTC register.

Reads a given RTC register and stores the output in the given pointer

Parameters

port	Address to be read from
output	Pointer to store the output

Returns

Return 0 upon success and non-zero otherwise

Definition at line 149 of file rtc.c.

4.9.2.7 rtc_set_alarm()

```
int() rtc_set_alarm (
            time_struct alarm_time )
```

Sets the RTC's alarm.

Enables the RTC's alarm interrupt and sets the alarm time

Returns

Return 0 upon success and non-zero otherwise

Definition at line 99 of file rtc.c.

4.9.2.8 rtc_start()

```
int() rtc_start ( )
```

Starts the RTC.

Reads the RTC's modes and sets the RTC's data structures

Returns

Return 0 upon success and non-zero otherwise

Definition at line 32 of file rtc.c.

4.9.2.9 rtc_stop()

```
int() rtc_stop ( )
```

Stops the RTC.

Restores the RTC's original configuration

Returns

Return 0 upon success and non-zero otherwise

Definition at line 48 of file rtc.c.

4.9.2.10 rtc_subscribe_int()

```
int() rtc_subscribe_int ( )
```

Subscribes the RTC's interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file rtc.c.

4.9.2.11 rtc_unsubscribe_int()

```
int() rtc_unsubscribe_int ( )
```

Unsubscribes the RTC's interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 16 of file rtc.c.

4.9.2.12 rtc_update()

```
int() rtc_update ( )
```

Updates the RTC's data structures.

Reads the RTC's date and time and updates the RTC structures

Returns

Return 0 upon success and non-zero otherwise

Definition at line 53 of file rtc.c.

4.9.2.13 rtc_write()

```
int() rtc_write (
             uint8_t port,
             uint8_t value )
```

Writes to a RTC register.

Writes a byte to a given RTC register

Parameters

port	Address to be written to
value	Value to be written

Returns

Return 0 upon success and non-zero otherwise

Definition at line 158 of file rtc.c.

4.10 /home/neves/reps/LCOM/shared/proj/src/controller/timer/timer.c File Reference

```
#include "timer.h"
```

Functions

- int() timer_subscribe_ints ()

  Subscribes the Timer interrupts.
- int() timer_unsubscribe_int ()

  Unsubscribes the Timer interrupts.
- void() timer_ih ()

  Timer Interrupt Handler.
- int() timer_get_conf (uint8_t timer, uint8_t ∗st)

  Gets the configuration of the timer.
- int() timer_set_frequency (uint8_t timer, uint32_t freq)

  Sets the frequency of the timer.

Variables

- int timer_hook_id = 0
- int timer_counter = 0

4.10.1 Function Documentation

4.10.1.1 timer_get_conf()

```
int() timer_get_conf (
            uint8_t timer,
            uint8_t * st )
```

Gets the configuration of the timer.

Parameters

timer	The timer number (0, 1, or 2)
st	Pointer to store the configuration

Returns

Return 0 upon success and non-zero otherwise

Definition at line 21 of file timer.c.

4.10.1.2 timer_ih()

```
void() timer_ih ( )
```

Timer Interrupt Handler.

Increments the timer_counter variable

Definition at line 16 of file timer.c.

4.10.1.3 timer_set_frequency()

```
int() timer_set_frequency (
            uint8_t timer,
            uint32_t freq )
```

Sets the frequency of the timer.

Parameters

timer	The timer number (0, 1, or 2)
freq	The frequency to set

Returns

Return 0 upon success and non-zero otherwise

Definition at line 34 of file timer.c.

4.10.1.4 timer_subscribe_ints()

```
int() timer_subscribe_ints ( )
```

Subscribes the Timer interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 6 of file timer.c.

4.10.1.5 timer_unsubscribe_int()

```
int() timer_unsubscribe_int ( )
```

Unsubscribes the Timer interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file timer.c.

4.10.2 Variable Documentation

4.10.2.1 timer_counter

```
int timer_counter = 0
```

Definition at line 4 of file timer.c.

4.10.2.2 timer_hook_id

```
int timer_hook_id = 0
```

Definition at line 3 of file timer.c.

4.11 /home/neves/reps/LCOM/shared/proj/src/controller/timer/timer.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "../utils.h"
```

Macros

- #define TIMER_IRQ 0

  Timer IRQ Line.
- #define TIMER_MASK BIT(0)

  Timer IRQ Mask.
- #define TIMER_FREQ 1193182

  Default Timer Frequency.
- #define TIMER_0 0x40

  Timer 0 Register.
- #define TIMER_1 0x41

  Timer 1 Register.
- #define TIMER_2 0x42

  Timer 2 Register.
- #define TIMER_CTRL 0x43

  Control Register.
- #define TIMER_SEL0 0x00

  Control Word for Timer 0.
- #define TIMER_SEL1 BIT(6)

  Control Word for Timer 1.
- #define TIMER_SEL2 BIT(7)

  Control Word for Timer 2.
- #define TIMER_LSB BIT(4)

  Initialize Counter LSB only.
- #define TIMER_MSB BIT(5)

  Initialize Counter MSB only.
- #define TIMER_LSB_MSB (TIMER_LSB | TIMER_MSB)

  Initialize LSB first and MSB afterwards.
- #define TIMER_SQR_WAVE (BIT(2) | BIT(1))

  Mode 3: Square Wave Generator Mode.
- #define TIMER_RATE_GEN BIT(2)

  Mode 2: Rate Generator Mode.
- #define TIMER_BCD 0x01

  BCD Mode.
- #define TIMER_BIN 0x00

  Binary Mode.
- #define TIMER_RB_COUNT_ BIT(5)

  Read Count Value.
- #define TIMER_RB_STATUS_ BIT(4)

  Read Status Value.
- #define TIMER_RB_SEL(n) BIT((n) + 1)

  Select Timer for Read-Back.
- #define TIMER_RB_CMD (BIT(7) | BIT(6))

  Read Back Command.

Functions

- int() timer_subscribe_ints ()

  Subscribes the Timer interrupts.
- int() timer_unsubscribe_int ()

  Unsubscribes the Timer interrupts.
- void() timer_ih ()

  Timer Interrupt Handler.
- int() timer_get_conf (uint8_t timer, uint8_t ∗st)

  Gets the configuration of the timer.
- int() timer_set_frequency (uint8_t timer, uint32_t freq)

  Sets the frequency of the timer.

4.11.1 Macro Definition Documentation

4.11.1.1 TIMER_0

```
#define TIMER_0 0x40
```

Timer 0 Register.

Definition at line 20 of file timer.h.

4.11.1.2 TIMER_1

```
#define TIMER_1 0x41
```

Timer 1 Register.

Definition at line 21 of file timer.h.

4.11.1.3 TIMER_2

```
#define TIMER_2 0x42
```

Timer 2 Register.

Definition at line 22 of file timer.h.

4.11.1.4 TIMER_BCD

```
#define TIMER_BCD 0x01
```

BCD Mode.

Definition at line 46 of file timer.h.

4.11.1.5 TIMER_BIN

```
#define TIMER_BIN 0x00
```

Binary Mode.

Definition at line 47 of file timer.h.

4.11.1.6 TIMER_CTRL

```
#define TIMER_CTRL 0x43
```

Control Register.

Definition at line 23 of file timer.h.

4.11.1.7 TIMER_FREQ

```
#define TIMER_FREQ 1193182
```

Default Timer Frequency.

Definition at line 16 of file timer.h.

4.11.1.8 TIMER_IRQ

```
#define TIMER_IRQ 0
```

Timer IRQ Line.

Definition at line 14 of file timer.h.

4.11.1.9 TIMER_LSB

```
#define TIMER_LSB BIT(4)
```

Initialize Counter LSB only.

Definition at line 35 of file timer.h.

4.11.1.10 TIMER_LSB_MSB

```
#define TIMER_LSB_MSB (TIMER_LSB | TIMER_MSB)
```

Initialize LSB first and MSB afterwards.

Definition at line 37 of file timer.h.

4.11.1.11 TIMER_MASK

```
#define TIMER_MASK BIT(0)
```

Timer IRQ Mask.

Definition at line 15 of file timer.h.

4.11.1.12 TIMER_MSB

```
#define TIMER_MSB BIT(5)
```

Initialize Counter MSB only.

Definition at line 36 of file timer.h.

4.11.1.13 TIMER_RATE_GEN

```
#define TIMER_RATE_GEN BIT(2)
```

Mode 2: Rate Generator Mode.

Definition at line 42 of file timer.h.

4.11.1.14 TIMER_RB_CMD

```
#define TIMER_RB_CMD (BIT(7) | BIT(6))
```

Read Back Command.

Definition at line 54 of file timer.h.

4.11.1.15 TIMER_RB_COUNT_

```
#define TIMER_RB_COUNT_ BIT(5)
```

Read Count Value.

Definition at line 51 of file timer.h.

4.11.1.16 TIMER_RB_SEL

```
#define TIMER_RB_SEL(
              n ) BIT((n) + 1)
```

Select Timer for Read-Back.

Definition at line 53 of file timer.h.

4.11.1.17 TIMER_RB_STATUS_

```
#define TIMER_RB_STATUS_ BIT(4)
```

Read Status Value.

Definition at line 52 of file timer.h.

4.11.1.18 TIMER_SEL0

```
#define TIMER_SEL0 0x00
```

Control Word for Timer 0.

Definition at line 29 of file timer.h.

4.11.1.19 TIMER_SEL1

```
#define TIMER_SEL1 BIT(6)
```

Control Word for Timer 1.

Definition at line 30 of file timer.h.

4.11.1.20 TIMER_SEL2

```
#define TIMER_SEL2 BIT(7)
```

Control Word for Timer 2.

Definition at line 31 of file timer.h.

4.11.1.21 TIMER_SQR_WAVE

```
#define TIMER_SQR_WAVE (BIT(2) | BIT(1))
```

Mode 3: Square Wave Generator Mode.

Definition at line 41 of file timer.h.

4.11.2 Function Documentation

4.11.2.1 timer_get_conf()

```
int() timer_get_conf (
            uint8_t timer,
            uint8_t * st )
```

Gets the configuration of the timer.

Parameters

timer	The timer number (0, 1, or 2)
st	Pointer to store the configuration

Returns

Return 0 upon success and non-zero otherwise

Definition at line 21 of file timer.c.

4.11.2.2 timer_ih()

```
void() timer_ih ( )
```

Timer Interrupt Handler.

Increments the timer_counter variable

Definition at line 16 of file timer.c.

4.11.2.3 timer_set_frequency()

```
int() timer_set_frequency (
             uint8_t timer,
             uint32_t freq )
```

Sets the frequency of the timer.

Parameters

timer	The timer number (0, 1, or 2)
freq	The frequency to set

Returns

Return 0 upon success and non-zero otherwise

Definition at line 34 of file timer.c.

4.11.2.4 timer_subscribe_ints()

```
int() timer_subscribe_ints ( )
```

Subscribes the Timer interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 6 of file timer.c.

4.11.2.5 timer_unsubscribe_int()

```
int() timer_unsubscribe_int ( )
```

Unsubscribes the Timer interrupts.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 11 of file timer.c.

4.12 /home/neves/reps/LCOM/shared/proj/src/controller/utils.c File Reference

```
#include "utils.h"
```

Functions

- int() util_get_LSB (uint16_t val, uint8_t ∗lsb)

  Retrieves the least significant byte of a 16-bit value.
- int() util_get_MSB (uint16_t val, uint8_t ∗msb)

  Retrieves the most significant byte of a 16-bit value.
- int() util_sys_inb (int port, uint8_t ∗value)

  Reads a byte from a specified port and converts it to an 8-bit value.

4.12.1 Function Documentation

4.12.1.1 util_get_LSB()

```
int() util_get_LSB (
            uint16_t val,
            uint8_t ∗ lsb )
```

Retrieves the least significant byte of a 16-bit value.

Parameters

val	16-bit value to be processed
lsb	Address of memory to store the least significant byte

Returns

Return 0 upon success and non-zero otherwise

Definition at line 3 of file utils.c.

4.12.1.2 util_get_MSB()

```
int() util_get_MSB (
            uint16_t val,
            uint8_t * msb )
```

Retrieves the most significant byte of a 16-bit value.

Parameters

val	16-bit value to be processed
msb	Address of memory to store the most significant byte

Returns

Return 0 upon success and non-zero otherwise

Definition at line 12 of file utils.c.

4.12.1.3 util_sys_inb()

```
int() util_sys_inb (
            int port,
            uint8_t * value )
```

Reads a byte from a specified port and converts it to an 8-bit value.

Parameters

port	Port to read from
value	Address of memory to store the processed value

Returns

Return 0 upon success and non-zero otherwise

Definition at line 21 of file utils.c.

4.13 /home/neves/reps/LCOM/shared/proj/src/controller/utils.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
```

Data Structures

- struct MouseInfo

  Data structure that holds the information about the mouse.
- struct time_struct

  Data structure that holds the time data.
- struct date_struct

  Data structure that holds the date data.

Macros

- #define BIT(n) (1 << (n))

Functions

- int() util_get_LSB (uint16_t val, uint8_t ∗lsb)

  Retrieves the least significant byte of a 16-bit value.
- int() util_get_MSB (uint16_t val, uint8_t ∗msb)

  Retrieves the most significant byte of a 16-bit value.
- int() util_sys_inb (int port, uint8_t ∗value)

  Reads a byte from a specified port and converts it to an 8-bit value.

4.13.1 Macro Definition Documentation

4.13.1.1 BIT

```
#define BIT(
                n ) (1 << (n))
```

Definition at line 9 of file utils.h.

4.13.2 Function Documentation

4.13.2.1 util_get_LSB()

```
int() util_get_LSB (
               uint16_t val,
               uint8_t * lsb )
```

Retrieves the least significant byte of a 16-bit value.

Parameters

val	16-bit value to be processed
lsb	Address of memory to store the least significant byte

Returns

Return 0 upon success and non-zero otherwise

Definition at line 3 of file utils.c.

4.13.2.2 util_get_MSB()

```
int() util_get_MSB (
              uint16_t val,
              uint8_t * msb )
```

Retrieves the most significant byte of a 16-bit value.

Parameters

val	16-bit value to be processed
msb	Address of memory to store the most significant byte

Returns

Return 0 upon success and non-zero otherwise

Definition at line 12 of file utils.c.

4.13.2.3 util_sys_inb()

```
int() util_sys_inb (
              int port,
              uint8_t * value )
```

Reads a byte from a specified port and converts it to an 8-bit value.

Parameters

port	Port to read from
value	Address of memory to store the processed value

Returns

Return 0 upon success and non-zero otherwise

Definition at line 21 of file utils.c.

4.14 /home/neves/reps/LCOM/shared/proj/src/controller/video/graphic.c File Reference

```
#include "graphic.h"
```

Functions

- int() set_graphic_mode (uint16_t submode)

  Sets the graphics mode using VESA BIOS Extensions.
- int() set_text_mode ()

  Sets the text mode using VESA BIOS Extensions.
- int() set_frame_buffer (uint16_t mode, uint8_t ∗∗frame_buffer)

  Sets the frame buffer for the specified mode.
- int() draw_pixel (uint16_t x, uint16_t y, uint32_t color, uint8_t ∗buffer)

  Draws a pixel on a buffer.
- int() draw_line (uint16_t x, uint16_t y, uint16_t len, uint32_t color, uint8_t ∗buffer)

  Draws a line of pixels on a buffer.
- int() draw_rectangle (uint16_t x, uint16_t y, uint16_t width, uint16_t height, uint32_t color, uint8_t ∗buffer)

  Draws a rectangle on a buffer.
- int() draw_XPM (xpm_map_t xpm, uint16_t x, uint16_t y, uint8_t ∗buffer)

  Draws a XPM image on the screen.

Variables

- vbe_mode_info_t vbe_info

4.14.1 Function Documentation

4.14.1.1 draw_line()

```
int() draw_line (
            uint16_t x,
            uint16_t y,
            uint16_t len,
            uint32_t color,
            uint8_t * buffer )
```

Draws a line of pixels on a buffer.

Parameters

x	The X coordinate of the starting pixel.
y	The Y coordinate of the starting pixel.
len	The length of the line in pixels.
color	The color of the line.
buffer	The buffer where the line will be drawn.

Returns

> Return 0 upon success and non-zero otherwise

Definition at line 66 of file graphic.c.

4.14.1.2 draw_pixel()

```
int() draw_pixel (
            uint16_t x,
            uint16_t y,
            uint32_t color,
            uint8_t * buffer )
```

Draws a pixel on a buffer.

Parameters

x	The X coordinate of the pixel.
y	The Y coordinate of the pixel.
color	The color of the pixel.
buffer	The buffer where the pixel will be drawn.

Returns

> Return 0 upon success and non-zero otherwise

Definition at line 54 of file graphic.c.

4.14.1.3 draw_rectangle()

```
int() draw_rectangle (
            uint16_t x,
            uint16_t y,
            uint16_t width,
            uint16_t height,
            uint32_t color,
            uint8_t * buffer )
```

Draws a rectangle on a buffer.

Parameters

x	The X coordinate of the top-left corner.
y	The Y coordinate of the top-left corner.
width	The width of the rectangle.
height	The height of the rectangle.
color	The color of the rectangle.
buffer	The buffer where the rectangle will be drawn.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 73 of file graphic.c.

4.14.1.4 draw_XPM()

```
int() draw_XPM (
            xpm_map_t xpm,
            uint16_t x,
            uint16_t y,
            uint8_t * buffer )
```

Draws a XPM image on the screen.

Parameters

xpm	The XPM image to draw.
x	The x coordinate of the top-left corner.
y	The y coordinate of the top-left corner.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 80 of file graphic.c.

4.14.1.5 set_frame_buffer()

```
int() set_frame_buffer (
            uint16_t mode,
            uint8_t ** frame_buffer )
```

Sets the frame buffer for the specified mode.

Parameters

mode	The mode to set.
frame_buffer	The frame buffer pointer to be used.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 35 of file graphic.c.

4.14.1.6 set_graphic_mode()

```
int() set_graphic_mode (
            uint16_t submode )
```

Sets the graphics mode using VESA BIOS Extensions.

Parameters

submode	The mode to set.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 5 of file graphic.c.

4.14.1.7 set_text_mode()

```
int() set_text_mode ( )
```

Sets the text mode using VESA BIOS Extensions.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 20 of file graphic.c.

4.14.2 Variable Documentation

4.14.2.1 vbe_info

```
vbe_mode_info_t vbe_info
```

Definition at line 3 of file graphic.c.

4.15 /home/neves/reps/LCOM/shared/proj/src/controller/video/graphic.h File Reference

```
#include <lcom/lcf.h>
#include <stdint.h>
#include "../utils.h"
```

Macros

- #define VBE_1024x768_INDEXED 0x105

  VBE mode for 1024x768 resolution.
- #define VBE_640x480 0x110

  VBE mode for 640x480 resolution.
- #define VBE_800x600 0x115

  VBE mode for 800x600 resolution.
- #define VBE_1280x1024 0x11A

  VBE mode for 1280x1024 resolution.
- #define VBE_1152x864 0x14C

  VBE mode for 1152x864 resolution.

Functions

- int() set_graphic_mode (uint16_t submode)

  Sets the graphics mode using VESA BIOS Extensions.
- int() set_text_mode ()

  Sets the text mode using VESA BIOS Extensions.
- int() set_frame_buffer (uint16_t mode, uint8_t ∗∗frame_buffer)

  Sets the frame buffer for the specified mode.
- int() draw_pixel (uint16_t x, uint16_t y, uint32_t color, uint8_t ∗buffer)

  Draws a pixel on a buffer.
- int() draw_line (uint16_t x, uint16_t y, uint16_t len, uint32_t color, uint8_t ∗buffer)

  Draws a line of pixels on a buffer.
- int() draw_rectangle (uint16_t x, uint16_t y, uint16_t width, uint16_t height, uint32_t color, uint8_t ∗buffer)

  Draws a rectangle on a buffer.
- int() draw_XPM (xpm_map_t xpm, uint16_t x, uint16_t y, uint8_t ∗buffer)

  Draws a XPM image on the screen.

4.15.1 Macro Definition Documentation

4.15.1.1 VBE_1024x768_INDEXED

```
#define VBE_1024x768_INDEXED 0x105
```

VBE mode for 1024x768 resolution.

Definition at line 12 of file graphic.h.

4.15.1.2 VBE_1152x864

```
#define VBE_1152x864 0x14C
```

VBE mode for 1152x864 resolution.

Definition at line 16 of file graphic.h.

4.15.1.3 VBE_1280x1024

```
#define VBE_1280x1024 0x11A
```

VBE mode for 1280x1024 resolution.

Definition at line 15 of file graphic.h.

4.15.1.4 VBE_640x480

```
#define VBE_640x480 0x110
```

VBE mode for 640x480 resolution.

Definition at line 13 of file graphic.h.

4.15.1.5 VBE_800x600

```
#define VBE_800x600 0x115
```

VBE mode for 800x600 resolution.

Definition at line 14 of file graphic.h.

4.15.2 Function Documentation

4.15.2.1 draw_line()

```
int() draw_line (
            uint16_t x,
            uint16_t y,
            uint16_t len,
            uint32_t color,
            uint8_t * buffer )
```

Draws a line of pixels on a buffer.

Parameters

x	The X coordinate of the starting pixel.
y	The Y coordinate of the starting pixel.
len	The length of the line in pixels.
color	The color of the line.
buffer	The buffer where the line will be drawn.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 66 of file graphic.c.

4.15.2.2 draw_pixel()

```
int() draw_pixel (
            uint16_t x,
            uint16_t y,
            uint32_t color,
            uint8_t * buffer )
```

Draws a pixel on a buffer.

Parameters

x	The X coordinate of the pixel.
y	The Y coordinate of the pixel.
color	The color of the pixel.
buffer	The buffer where the pixel will be drawn.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 54 of file graphic.c.

4.15.2.3 draw_rectangle()

```
int() draw_rectangle (
            uint16_t x,
            uint16_t y,
            uint16_t width,
            uint16_t height,
            uint32_t color,
            uint8_t * buffer )
```

Draws a rectangle on a buffer.

Parameters

x	The X coordinate of the top-left corner.
y	The Y coordinate of the top-left corner.
width	The width of the rectangle.
height	The height of the rectangle.
color	The color of the rectangle.
buffer	The buffer where the rectangle will be drawn.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 73 of file graphic.c.

4.15.2.4 draw_XPM()

```
int() draw_XPM (
            xpm_map_t xpm,
            uint16_t x,
            uint16_t y,
            uint8_t * buffer )
```

Draws a XPM image on the screen.

Parameters

xpm	The XPM image to draw.
x	The x coordinate of the top-left corner.
y	The y coordinate of the top-left corner.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 80 of file graphic.c.

4.15.2.5 set_frame_buffer()

```
int() set_frame_buffer (
            uint16_t mode,
            uint8_t ** frame_buffer )
```

Sets the frame buffer for the specified mode.

Parameters

mode	The mode to set.
frame_buffer	The frame buffer pointer to be used.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 35 of file graphic.c.

4.15.2.6 set_graphic_mode()

```
int() set_graphic_mode (
            uint16_t submode )
```

Sets the graphics mode using VESA BIOS Extensions.

Parameters

submode	The mode to set.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 5 of file graphic.c.

4.15.2.7 set_text_mode()

```
int() set_text_mode ( )
```

Sets the text mode using VESA BIOS Extensions.

Returns

Return 0 upon success and non-zero otherwise

Definition at line 20 of file graphic.c.

4.16 /home/neves/reps/LCOM/shared/proj/src/main.c File Reference

```
#include <lcom/lcf.h>
#include "model/model.h"
#include "view/view.h"
```

Functions

- int() main (int argc, char ∗argv[ ])
- int setup ()
- int cleanup ()
- int() proj_main_loop (int argc, char ∗argv[ ])

Variables

- SystemState systemState

4.16.1 Function Documentation

4.16.1.1 cleanup()

```
int cleanup ( )
```

Definition at line 53 of file main.c.

4.16.1.2 main()

```
int() main (
             int argc,
             char * argv[ ] )
```

Definition at line 8 of file main.c.

4.16.1.3 proj_main_loop()

```
int() proj_main_loop (
             int argc,
             char * argv[ ] )
```

Definition at line 76 of file main.c.

4.16.1.4 setup()

```
int setup ( )
```

Definition at line 23 of file main.c.

4.16.2 Variable Documentation

4.16.2.1 systemState

```
SystemState systemState [extern]
```

Definition at line 5 of file model.c.

4.17 /home/neves/reps/LCOM/shared/proj/src/model/model.c File Reference

```
#include "model.h"
#include "sprite.h"
```

Functions

- void setup_sprites ()
- void destroy_sprites ()
- void update_timer_state ()
- void update_keyboard_state ()
- void insert_new_input (int new_input)
- void delete_last_input ()
- void update_mouse_state ()
- void update_chrono_buttons ()
- void update_timer_buttons ()
- void update_toolbar_buttons ()
- bool is_mouse_over_button (Sprite ∗button)

Variables

- SystemState systemState = RUNNING
- MenuState menuState = RUNNING_CLOCK
- ChronoState chronoState = OFF
- ChronoState timerState = OFF
- int chrono_seconds = 0
- int timer_seconds = 0
- int timer_input [6] = {0, 0, 0, 0, 0, 0}
- int timer_input_length = 0
- Sprite ∗ mouse
- Sprite ∗ colon
- Sprite ∗ slash
- Sprite ∗ block
- Sprite ∗ days_of_week [7]
- Sprite ∗ digits [10]
- Sprite ∗ toolbar_buttons [3]
- Sprite ∗ chrono_buttons [3]
- int timer_counter
- uint8_t scancode
- uint8_t byte_index
- MouseInfo mouse_info
- vbe_mode_info_t vbe_info

4.17.1 Function Documentation

4.17.1.1 delete_last_input()

```
void delete_last_input ( )
```

Definition at line 203 of file model.c.

4.17.1.2 destroy_sprites()

```
void destroy_sprites ( )
```

Definition at line 92 of file model.c.

4.17.1.3 insert_new_input()

```
void insert_new_input (
             int new_input )
```

Definition at line 191 of file model.c.

4.17.1.4 is_mouse_over_button()

```
bool is_mouse_over_button (
             Sprite * button )
```

Definition at line 289 of file model.c.

4.17.1.5 setup_sprites()

```
void setup_sprites ( )
```

Definition at line 38 of file model.c.

4.17.1.6 update_chrono_buttons()

```
void update_chrono_buttons ( )
```

Definition at line 235 of file model.c.

4.17.1.7 update_keyboard_state()

```
void update_keyboard_state ( )
```

Definition at line 144 of file model.c.

4.17.1.8 update_mouse_state()

```
void update_mouse_state ( )
```

Definition at line 220 of file model.c.

4.17.1.9 update_timer_buttons()

```
void update_timer_buttons ( )
```

Definition at line 250 of file model.c.

4.17.1.10 update_timer_state()

```
void update_timer_state ( )
```

Definition at line 109 of file model.c.

4.17.1.11 update_toolbar_buttons()

```
void update_toolbar_buttons ( )
```

Definition at line 275 of file model.c.

4.17.2 Variable Documentation

4.17.2.1 block

```
Sprite* block
```

Definition at line 18 of file model.c.

4.17.2.2 byte_index

```
uint8_t byte_index  [extern]
```

Definition at line 4 of file mouse.c.

4.17.2.3 chrono_buttons

```
Sprite* chrono_buttons[3]
```

Definition at line 22 of file model.c.

4.17.2.4 chrono_seconds

```
int chrono_seconds = 0
```

Definition at line 9 of file model.c.

4.17.2.5 chronoState

```
ChronoState chronoState = OFF
```

Definition at line 7 of file model.c.

4.17.2.6 colon

```
Sprite* colon
```

Definition at line 16 of file model.c.

4.17.2.7 days_of_week

```
Sprite* days_of_week[7]
```

Definition at line 19 of file model.c.

4.17.2.8 digits

```
Sprite* digits[10]
```

Definition at line 20 of file model.c.

4.17.2.9 menuState

```
MenuState menuState = RUNNING_CLOCK
```

Definition at line 6 of file model.c.

4.17.2.10 mouse

```
Sprite* mouse
```

Definition at line 15 of file model.c.

4.17.2.11 mouse_info

```
MouseInfo mouse_info [extern]
```

Definition at line 7 of file mouse.c.

4.17.2.12 scancode

```
uint8_t scancode  [extern]
```

Definition at line 4 of file keyboard.c.

4.17.2.13 slash

```
Sprite* slash
```

Definition at line 17 of file model.c.

4.17.2.14 systemState

```
SystemState systemState = RUNNING
```

Definition at line 5 of file model.c.

4.17.2.15 timer_counter

```
int timer_counter  [extern]
```

Definition at line 4 of file timer.c.

4.17.2.16 timer_input

```
int timer_input[6] = {0, 0, 0, 0, 0, 0}
```

Definition at line 11 of file model.c.

4.17.2.17 timer_input_length

```
int timer_input_length = 0
```

Definition at line 12 of file model.c.

4.17.2.18 timer_seconds

```
int timer_seconds = 0
```

Definition at line 10 of file model.c.

4.17.2.19 timerState

```
ChronoState timerState = OFF
```

Definition at line 8 of file model.c.

4.17.2.20 toolbar_buttons

```
Sprite* toolbar_buttons[3]
```

Definition at line 21 of file model.c.

4.17.2.21 vbe_info

```
vbe_mode_info_t vbe_info [extern]
```

Definition at line 3 of file graphic.c.

4.18 /home/neves/reps/LCOM/shared/proj/src/model/model.h File Reference

```
#include <minix/sysutil.h>
#include <lcom/lcf.h>
#include "controller/timer/timer.h"
#include "controller/keyboard/keyboard.h"
#include "controller/mouse/mouse.h"
#include "controller/video/graphic.h"
#include "controller/rtc/rtc.h"
#include "view/view.h"
#include "model/sprite.h"
#include "config.h"
#include "xpm/colon.xpm"
#include "xpm/mouse.xpm"
#include "xpm/num_0.xpm"
#include "xpm/num_1.xpm"
#include "xpm/num_2.xpm"
#include "xpm/num_3.xpm"
#include "xpm/num_4.xpm"
```

```
#include "xpm/num_5.xpm"
#include "xpm/num_6.xpm"
#include "xpm/num_7.xpm"
#include "xpm/num_8.xpm"
#include "xpm/num_9.xpm"
#include "xpm/slash.xpm"
#include "xpm/timer.xpm"
#include "xpm/chrono.xpm"
#include "xpm/clock.xpm"
#include "xpm/sun.xpm"
#include "xpm/mon.xpm"
#include "xpm/tue.xpm"
#include "xpm/wed.xpm"
#include "xpm/thu.xpm"
#include "xpm/fri.xpm"
#include "xpm/sat.xpm"
#include "xpm/start.xpm"
#include "xpm/pause.xpm"
#include "xpm/reset.xpm"
```

Enumerations

- enum SystemState { RUNNING , EXIT }
- enum MenuState { RUNNING_CLOCK , CHRONO , TIMER }
- enum ChronoState { ON , OFF }

Functions

- void update_timer_state ()
- void update_keyboard_state ()
- void update_mouse_state ()
- void update_chrono_buttons ()
- void update_toolbar_buttons ()
- void update_timer_buttons ()
- void setup_sprites ()
- void destroy_sprites ()
- void insert_new_input (int new_input)
- void delete_last_input ()
- bool is_mouse_over_button (Sprite ∗button)

Variables

- Sprite ∗ mouse
- Sprite ∗ colon
- Sprite ∗ slash
- Sprite ∗ block
- Sprite ∗ days_of_week [7]
- Sprite ∗ digits [10]
- Sprite ∗ toolbar_buttons [3]
- Sprite ∗ chrono_buttons [3]
- int chrono_seconds
- int timer_seconds

- int timer_input [6]
- int timer_input_length
- SystemState systemState
- MenuState menuState
- ChronoState chronoState
- ChronoState timerState

4.18.1 Enumeration Type Documentation

4.18.1.1 ChronoState

```
enum ChronoState
```

Enumerator

ON	
OFF	

Definition at line 70 of file model.h.

4.18.1.2 MenuState

```
enum MenuState
```

Enumerator

RUNNING_CLOCK	
CHRONO	
TIMER	

Definition at line 64 of file model.h.

4.18.1.3 SystemState

```
enum SystemState
```

Enumerator

RUNNING	
EXIT	

Definition at line 59 of file model.h.

4.18.2 Function Documentation

4.18.2.1 delete_last_input()

```
void delete_last_input ( )
```

Definition at line 203 of file model.c.

4.18.2.2 destroy_sprites()

```
void destroy_sprites ( )
```

Definition at line 92 of file model.c.

4.18.2.3 insert_new_input()

```
void insert_new_input (
             int new_input )
```

Definition at line 191 of file model.c.

4.18.2.4 is_mouse_over_button()

```
bool is_mouse_over_button (
             Sprite * button )
```

Definition at line 289 of file model.c.

4.18.2.5 setup_sprites()

```
void setup_sprites ( )
```

Definition at line 38 of file model.c.

4.18.2.6 update_chrono_buttons()

```
void update_chrono_buttons ( )
```

Definition at line 235 of file model.c.

4.18.2.7 update_keyboard_state()

```
void update_keyboard_state ( )
```

Definition at line 144 of file model.c.

4.18.2.8 update_mouse_state()

```
void update_mouse_state ( )
```

Definition at line 220 of file model.c.

4.18.2.9 update_timer_buttons()

```
void update_timer_buttons ( )
```

Definition at line 250 of file model.c.

4.18.2.10 update_timer_state()

```
void update_timer_state ( )
```

Definition at line 109 of file model.c.

4.18.2.11 update_toolbar_buttons()

```
void update_toolbar_buttons ( )
```

Definition at line 275 of file model.c.

4.18.3 Variable Documentation

4.18.3.1 block

```
Sprite* block  [extern]
```

Definition at line 18 of file model.c.

4.18.3.2 chrono_buttons

```
Sprite* chrono_buttons[3]  [extern]
```

Definition at line 22 of file model.c.

4.18.3.3 chrono_seconds

```
int chrono_seconds  [extern]
```

Definition at line 9 of file model.c.

4.18.3.4 chronoState

```
ChronoState chronoState  [extern]
```

Definition at line 7 of file model.c.

4.18.3.5 colon

```
Sprite* colon  [extern]
```

Definition at line 16 of file model.c.

4.18.3.6 days_of_week

```
Sprite* days_of_week[7]  [extern]
```

Definition at line 19 of file model.c.

4.18.3.7 digits

```
Sprite* digits[10]  [extern]
```

Definition at line 20 of file model.c.

4.18.3.8 menuState

```
MenuState menuState  [extern]
```

Definition at line 6 of file model.c.

4.18.3.9 mouse

```
Sprite* mouse  [extern]
```

Definition at line 15 of file model.c.

4.18.3.10 slash

```
Sprite* slash  [extern]
```

Definition at line 17 of file model.c.

4.18.3.11 systemState

```
SystemState systemState  [extern]
```

Definition at line 5 of file model.c.

4.18.3.12 timer_input

```
int timer_input[6] [extern]
```

Definition at line 11 of file model.c.

4.18.3.13 timer_input_length

```
int timer_input_length [extern]
```

Definition at line 12 of file model.c.

4.18.3.14 timer_seconds

```
int timer_seconds [extern]
```

Definition at line 10 of file model.c.

4.18.3.15 timerState

```
ChronoState timerState [extern]
```

Definition at line 8 of file model.c.

4.18.3.16 toolbar_buttons

```
Sprite* toolbar_buttons[3] [extern]
```

Definition at line 21 of file model.c.

4.19 /home/neves/reps/LCOM/shared/proj/src/model/sprite.c File Reference

```
#include "sprite.h"
```

Functions

- Sprite ∗ create_sprite_xpm (xpm_map_t sprite)
- Sprite ∗ create_sprite_button (uint16_t width, uint16_t height, uint32_t color)
- void destroy_sprite (Sprite ∗sprite)

4.19.1 Function Documentation

4.19.1.1 create_sprite_button()

```
Sprite* create_sprite_button (
            uint16_t width,
            uint16_t height,
            uint32_t color )
```

Definition at line 24 of file sprite.c.

4.19.1.2 create_sprite_xpm()

```
Sprite* create_sprite_xpm (
            xpm_map_t sprite )
```

Definition at line 5 of file sprite.c.

4.19.1.3 destroy_sprite()

```
void destroy_sprite (
            Sprite * sprite )
```

Definition at line 35 of file sprite.c.

4.20 /home/neves/reps/LCOM/shared/proj/src/model/sprite.h File Reference

```
#include "controller/video/graphic.h"
```

Data Structures

- struct Sprite

Functions

- Sprite ∗ create_sprite_xpm (xpm_map_t sprite)
- Sprite ∗ create_sprite_button (uint16_t width, uint16_t height, uint32_t color)
- void destroy_sprite (Sprite ∗sprite)

4.20.1 Function Documentation

4.20.1.1 create_sprite_button()

```
Sprite* create_sprite_button (
             uint16_t width,
             uint16_t height,
             uint32_t color )
```

Definition at line 24 of file sprite.c.

4.20.1.2 create_sprite_xpm()

```
Sprite* create_sprite_xpm (
             xpm_map_t sprite )
```

Definition at line 5 of file sprite.c.

4.20.1.3 destroy_sprite()

```
void destroy_sprite (
             Sprite * sprite )
```

Definition at line 35 of file sprite.c.

4.21 /home/neves/reps/LCOM/shared/proj/src/view/view.c File Reference

```
#include "view.h"
```

Functions

- int set_frame_buffers (uint16_t mode)
- void swap_buffers ()
- void draw_new_frame ()
- void draw_time ()
- void draw_chrono_menu ()
- void draw_chrono_buttons ()
- void draw_timer_menu ()
- void draw_blocks ()
- void draw_timer_input ()
- void draw_mouse ()
- void display_real_time ()
- void draw_toolbar ()
- int draw_sprite_xpm (Sprite ∗sprite, int x, int y)
- int draw_sprite_button (Sprite ∗sprite, int x, int y)

Variables

- uint8_t ∗ main_frame_buffer
- uint8_t ∗ secondary_frame_buffer
- uint8_t ∗ drawing_frame_buffer
- uint32_t frame_buffer_size
- MouseInfo mouse_info
- vbe_mode_info_t vbe_info
- time_struct rtc_time
- date_struct rtc_date

4.21.1 Function Documentation

4.21.1.1 display_real_time()

```
void display_real_time ( )
```

Definition at line 143 of file view.c.

4.21.1.2 draw_blocks()

```
void draw_blocks ( )
```

Definition at line 105 of file view.c.

4.21.1.3 draw_chrono_buttons()

```
void draw_chrono_buttons ( )
```

Definition at line 77 of file view.c.

4.21.1.4 draw_chrono_menu()

```
void draw_chrono_menu ( )
```

Definition at line 57 of file view.c.

4.21.1.5 draw_mouse()

```
void draw_mouse ( )
```

Definition at line 137 of file view.c.

4.21.1.6 draw_new_frame()

```
void draw_new_frame ( )
```

Definition at line 35 of file view.c.

4.21.1.7 draw_sprite_button()

```
int draw_sprite_button (
             Sprite * sprite,
             int x,
             int y )
```

Definition at line 234 of file view.c.

4.21.1.8 draw_sprite_xpm()

```
int draw_sprite_xpm (
             Sprite * sprite,
             int x,
             int y )
```

Definition at line 213 of file view.c.

4.21.1.9 draw_time()

```
void draw_time ( )
```

Definition at line 51 of file view.c.

4.21.1.10 draw_timer_input()

```
void draw_timer_input ( )
```

Definition at line 117 of file view.c.

4.21.1.11 draw_timer_menu()

```
void draw_timer_menu ( )
```

Definition at line 94 of file view.c.

4.21.1.12 draw_toolbar()

```
void draw_toolbar ( )
```

Definition at line 204 of file view.c.

4.21.1.13 set_frame_buffers()

```
int set_frame_buffers (
             uint16_t mode )
```

Definition at line 19 of file view.c.

4.21.1.14 swap_buffers()

```
void swap_buffers ( )
```

Definition at line 31 of file view.c.

4.21.2 Variable Documentation

4.21.2.1 drawing_frame_buffer

```
uint8_t* drawing_frame_buffer
```

Definition at line 6 of file view.c.

4.21.2.2 frame_buffer_size

```
uint32_t frame_buffer_size
```

Definition at line 7 of file view.c.

4.21.2.3 main_frame_buffer

```
uint8_t* main_frame_buffer
```

Definition at line 4 of file view.c.

4.21.2.4 mouse_info

```
MouseInfo mouse_info  [extern]
```

Definition at line 7 of file mouse.c.

4.21.2.5 rtc_date

```
date_struct rtc_date  [extern]
```

Definition at line 9 of file rtc.c.

4.21.2.6 rtc_time

```
time_struct rtc_time  [extern]
```

Definition at line 8 of file rtc.c.

4.21.2.7 secondary_frame_buffer

```
uint8_t* secondary_frame_buffer
```

Definition at line 5 of file view.c.

4.21.2.8 vbe_info

```
vbe_mode_info_t vbe_info  [extern]
```

Definition at line 3 of file graphic.c.

4.22 /home/neves/reps/LCOM/shared/proj/src/view/view.h File Reference

```
#include <minix/sysutil.h>
#include <lcom/lcf.h>
#include "config.h"
#include "controller/video/graphic.h"
#include "controller/mouse/mouse.h"
#include "controller/rtc/rtc.h"
#include "controller/utils.h"
#include "model/sprite.h"
#include "model/model.h"
```

Functions

- void draw_new_frame ()
- void draw_time ()
- void draw_chrono_menu ()
- void draw_chrono_buttons ()
- void draw_timer_menu ()
- void draw_blocks ()
- void draw_timer_input ()
- void draw_timer ()
- void draw_toolbar ()
- void draw_mouse ()
- void swap_buffers ()
- void display_real_time ()
- int draw_sprite_xpm (Sprite ∗sprite, int x, int y)
- int draw_sprite_button (Sprite ∗sprite, int x, int y)
- int set_frame_buffers (uint16_t mode)

Variables

- uint8_t ∗ drawing_frame_buffer
- uint8_t ∗ main_frame_buffer

4.22.1 Function Documentation

4.22.1.1 display_real_time()

```
void display_real_time ( )
```

Definition at line 143 of file view.c.

4.22.1.2 draw_blocks()

```
void draw_blocks ( )
```

Definition at line 105 of file view.c.

4.22.1.3 draw_chrono_buttons()

```
void draw_chrono_buttons ( )
```

Definition at line 77 of file view.c.

4.22.1.4 draw_chrono_menu()

```
void draw_chrono_menu ( )
```

Definition at line 57 of file view.c.

4.22.1.5 draw_mouse()

```
void draw_mouse ( )
```

Definition at line 137 of file view.c.

4.22.1.6 draw_new_frame()

```
void draw_new_frame ( )
```

Definition at line 35 of file view.c.

4.22.1.7 draw_sprite_button()

```
int draw_sprite_button (
             Sprite * sprite,
             int x,
             int y )
```

Definition at line 234 of file view.c.

4.22.1.8 draw_sprite_xpm()

```
int draw_sprite_xpm (
             Sprite * sprite,
             int x,
             int y )
```

Definition at line 213 of file view.c.

4.22.1.9 draw_time()

```
void draw_time ( )
```

Definition at line 51 of file view.c.

4.22.1.10 draw_timer()

```
void draw_timer ( )
```

4.22.1.11 draw_timer_input()

```
void draw_timer_input ( )
```

Definition at line 117 of file view.c.

4.22.1.12 draw_timer_menu()

```
void draw_timer_menu ( )
```

Definition at line 94 of file view.c.

4.22.1.13 draw_toolbar()

```
void draw_toolbar ( )
```

Definition at line 204 of file view.c.

4.22.1.14 set_frame_buffers()

```
int set_frame_buffers (
             uint16_t mode )
```

Definition at line 19 of file view.c.

4.22.1.15 swap_buffers()

```
void swap_buffers ( )
```

Definition at line 31 of file view.c.

4.22.2 Variable Documentation

4.22.2.1 drawing_frame_buffer

```
uint8_t* drawing_frame_buffer  [extern]
```

Definition at line 6 of file view.c.

4.22.2.2 main_frame_buffer

```
uint8_t* main_frame_buffer  [extern]
```

Definition at line 4 of file view.c.

Index