Code documentation

1

Generated by Doxygen 1.8.11

Contents

1 Class Index **1**

1.1 Class List 1

2 File Index **3**

2.1 File List 3

3 Class Documentation **5**

3.1 car Struct Reference 5

3.1.1 Detailed Description 5

3.1.2 Member Data Documentation 5

3.1.2.1 doors 5

3.1.2.2 lights 5

3.1.2.3 r 6

3.1.2.4 seatbelts 6

3.1.2.5 tempEngine 6

3.1.2.6 tempIn 6

3.1.2.7 tempOut 6

4 File Documentation **7**

4.1 FT800.cpp File Reference . . . 7

4.1.1 Detailed Description . . . 7

4.1.2 Function Documentation . . . 7

4.1.2.1 delay_ms(int ms) . . . 7

4.1.2.2 delay_us(int us) . . . 8

4.1.2.3 ft800cmdWrite(unsigned char ftCommand) . . . 8

4.1.2.4 ft800memRead16(unsigned long ftAddress) . . . 8

4.1.2.5 ft800memRead32(unsigned long ftAddress) . . . 8

4.1.2.6 ft800memRead8(unsigned long ftAddress) . . . 9

4.1.2.7 ft800memWrite16(unsigned long ftAddress, unsigned int ftData16) . . . 9

4.1.2.8 ft800memWrite32(unsigned long ftAddress, unsigned long ftData32) . . . 9

4.1.2.9 ft800memWrite8(unsigned long ftAddress, unsigned char ftData8) . . . 9

4.1.2.10 getData() . . . 9

4.1.2.11 incCMDOffset(unsigned int currentOffset, unsigned char commandSize) . . . 10

4.1.2.12 sendData(int data) . . . 10

4.2 FT800.h File Reference . . . 10

4.2.1 Detailed Description . . . 16

4.2.2 Macro Definition Documentation . . . 16

4.2.2.1 BLACK . . . 16

4.2.2.2 BLUE . . . 16

4.2.2.3 FT800_ACTIVE . . . 16

4.2.2.4 FT800_CLK36M . . . 16

4.2.2.5 FT800_CLK48M . . . 16

4.2.2.6 FT800_CLKEXT . . . 16

4.2.2.7 FT800_CORERST . . . 16

4.2.2.8 FT800_PWRDOWN . . . 17

4.2.2.9 FT800_SLEEP . . . 17

4.2.2.10 FT800_STANDBY . . . 17

4.2.2.11 FT_CMD_FIFO_SIZE . . . 17

4.2.2.12 FT_CMD_SIZE . . . 17
4.2.2.13 FT_DL_SIZE . . . 17
4.2.2.14 FTPOINTS . . . 17
4.2.2.15 GREEN . . . 17
4.2.2.16 LCD_QVGA . . . 17
4.2.2.17 MEM_READ . . . 17
4.2.2.18 MEM_WRITE . . . 18
4.2.2.19 RED . . . 18
4.2.2.20 WHITE . . . 18
4.2.2.21 xclock . . . 18
4.2.2.22 xCS . . . 18
4.2.2.23 xPD . . . 18
4.2.2.24 xSDI . . . 18
4.2.2.25 xSDO . . . 18
4.2.3 Function Documentation . . . 18
4.2.3.1 delay_ms(int ms) . . . 18
4.2.3.2 delay_us(int us) . . . 19
4.2.3.3 ft800cmdWrite(unsigned char ftCommand) . . . 19
4.2.3.4 ft800memRead16(unsigned long ftAddress) . . . 19
4.2.3.5 ft800memRead32(unsigned long ftAddress) . . . 19
4.2.3.6 ft800memRead8(unsigned long ftAddress) . . . 19
4.2.3.7 ft800memWrite16(unsigned long ftAddress, unsigned int ftData16) . . . 20
4.2.3.8 ft800memWrite32(unsigned long ftAddress, unsigned long ftData32) . . . 20
4.2.3.9 ft800memWrite8(unsigned long ftAddress, unsigned char ftData8) . . . 20
4.2.3.10 getData() . . . 20
4.2.3.11 incCMDOffset(unsigned int currentOffset, unsigned char commandSize) . . . 21
4.2.3.12 sendData(int data) . . . 21
4.3 FT800api.cpp File Reference . . . 21
4.3.1 Detailed Description . . . 22
4.3.2 Function Documentation . . . 22

4.3.2.1 autko() . . . 22
4.3.2.2 button(int16_t x, int16_t y, int16_t w, int16_t h, int16_t font, uint16_t options, const char *str) . . . 22
4.3.2.3 calibrate() . . . 22
4.3.2.4 dot(unsigned long color, unsigned int point_size, unsigned long point_x, unsigned long point_y) . . . 22
4.3.2.5 initScreen() . . . 23
4.3.2.6 line(unsigned long color, unsigned long line_x1, unsigned long line_y1, unsigned long line_x2, unsigned long line_y2, unsigned long width) . . . 23
4.3.2.7 mainScreen() . . . 23
4.3.2.8 opctionsScreen() . . . 23
4.3.2.9 show() . . . 23
4.3.2.10 smartMirrorScreen() . . . 23
4.3.2.11 spinner(int16_t x, int16_t y, uint16_t style, uint16_t scale) . . . 23
4.3.2.12 start(unsigned long color) . . . 24
4.3.2.13 text(int16_t x, int16_t y, int16_t font, uint16_t options, const char *str) . . . 24
4.4 FT800api.h File Reference . . . 24
4.4.1 Detailed Description . . . 25
4.4.2 Function Documentation . . . 25
4.4.2.1 autko() . . . 25
4.4.2.2 button(int16_t x, int16_t y, int16_t w, int16_t h, int16_t font, uint16_t options, const char *str) . . . 25
4.4.2.3 calibrate() . . . 25
4.4.2.4 dot(unsigned long color, unsigned int point_size, unsigned long point_x, unsigned long point_y) . . . 26
4.4.2.5 initScreen() . . . 27
4.4.2.6 line(unsigned long color, unsigned long line_x1, unsigned long line_y1, unsigned long line_x2, unsigned long line_y2, unsigned long width) . . . 27
4.4.2.7 mainScreen() . . . 27
4.4.2.8 number(int16_t x, int16_t y, int16_t font, uint16_t options, int32_t value) . . . 27
4.4.2.9 opctionsScreen() . . . 27
4.4.2.10 show() . . . 28
4.4.2.11 smartMirrorScreen() . . . 28

4.4.2.12 spinner(int16_t x, int16_t y, uint16_t style, uint16_t scale) . . . 28
4.4.2.13 start(unsigned long color) . . . 28
4.4.2.14 text(int16_t x, int16_t y, int16_t font, uint16_t options, const char *str) . . . 28
4.5 I2C.cpp File Reference . . . 28
4.5.1 Detailed Description . . . 29
4.5.2 Function Documentation . . . 29
4.5.2.1 readPCF(char adres) . . . 29
4.5.3 Variable Documentation . . . 29
4.5.3.1 d . . . 29
4.6 I2C.h File Reference . . . 30
4.6.1 Detailed Description . . . 30
4.6.2 Macro Definition Documentation . . . 30
4.6.2.1 pinInt0 . . . 30
4.6.2.2 scl . . . 30
4.6.2.3 sda . . . 30
4.6.3 Function Documentation . . . 30
4.6.3.1 readPCF(char adres) . . . 30
4.7 simulator.cpp File Reference . . . 31
4.7.1 Detailed Description . . . 31
4.7.2 Function Documentation . . . 31
4.7.2.1 checkChangesDigital() . . . 31
4.7.2.2 printObj(struct car *obj, char *d) . . . 31
4.7.2.3 readData() . . . 32
4.7.2.4 readTemp(int portNumber) . . . 32
4.7.2.5 save(struct car *audi, struct car *tmp) . . . 32
4.7.2.6 sendData() . . . 32
4.8 simulator.h File Reference . . . 32
4.8.1 Detailed Description . . . 33
4.8.2 Function Documentation . . . 33
4.8.2.1 checkChangesAnalog(struct car *audi) . . . 33
4.8.2.2 checkChangesDigital() . . . 33
4.8.2.3 printObj(struct car *obj, char *d) . . . 34
4.8.2.4 readData() . . . 34
4.8.2.5 readTemp(int portNumber) . . . 34
4.8.2.6 save(struct car *audi, struct car *tmp) . . . 34
4.8.2.7 sendData() . . . 34

Index **35**

Chapter 1

Class Index

1.1 Class List

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

car 5

Chapter 2

File Index

2.1 File List

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

FT800.cpp
File containing declarations of all functions required to use with VM800 7
FT800.h
File containing declarations of all functions required to use with VM800 10
FT800api.cpp
File containing declarations of all API functions for VM800 21
FT800api.h
File containing declarations of all API functions for VM800 24
I2C.cpp
File containing declarations of function to read data with using I2C protocol 28
I2C.h
File containing declarations of function to read data with using I2C protocol 30
simulator.cpp
File containing declarations of all functions required to communication with car simulator . . . 31
simulator.h
File containing declarations of all functions required to communication with car simulator . . . 32

Chapter 3

Class Documentation

3.1 car Struct Reference

`#include <simulator.h>`

Public Attributes

- int doors
- int seatbelts
- int lights
- int r
- float tempOut
- float tempIn
- float tempEngine

3.1.1 Detailed Description

Analog ports * A0 - temp Out * A1 - temp In * A2 - temp Engine *

A global car structure *

3.1.2 Member Data Documentation

3.1.2.1 int car::doors

status of doors in car. 1 - open, 0 closed

3.1.2.2 int car::lights

status of lights. 1 -turn on, 0 - turn off

3.1.2.3 int car::r

statu of reverse gear

3.1.2.4 int car::seatbelts

status of seatbelts in car. 1 - open, 0 - closed

3.1.2.5 float car::tempEngine

temperature engine

3.1.2.6 float car::tempIn

temperature inside

3.1.2.7 float car::tempOut

temperature outside

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

- simulator.h

Chapter 4

File Documentation

4.1 FT800.cpp File Reference

File containing declarations of all functions required to use with VM800.

```
#include "FT800.h"
#import <Arduino.h>
```

Functions

- void delay_us (int us)
- void delay_ms (int ms)
- void sendData (int data)
- unsigned char getData ()
- void ft800memWrite8 (unsigned long ftAddress, unsigned char ftData8)
- void ft800memWrite16 (unsigned long ftAddress, unsigned int ftData16)
- void ft800memWrite32 (unsigned long ftAddress, unsigned long ftData32)
- unsigned char ft800memRead8 (unsigned long ftAddress)
- unsigned char ft800memRead16 (unsigned long ftAddress)
- unsigned long ft800memRead32 (unsigned long ftAddress)
- unsigned int incCMDOffset (unsigned int currentOffset, unsigned char commandSize)
- void ft800cmdWrite (unsigned char ftCommand)

4.1.1 Detailed Description

File containing declarations of all functions required to use with VM800.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.1.2 Function Documentation

4.1.2.1 void delay_ms (int *ms*)

Pauses the program for the amount of time (in milisecond) specified as parameter ∗

Parameters

ms	milisecond to delay ∗

4.1.2.2 void delay_us (int *us*)

Pauses the program for the amount of time (in microsecond) specified as parameter ∗

Parameters

us	microseconds to delay ∗

4.1.2.3 void ft800cmdWrite (unsigned char *ftCommand*)

Sends FT800 command ∗

Parameters

ftCommand	command to send to device ∗

4.1.2.4 unsigned char ft800memRead16 (unsigned long *ftAddress*)

Funtion to read 16 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

16 bit data obtained from device ∗

4.1.2.5 unsigned long ft800memRead32 (unsigned long *ftAddress*)

Funtion to read 32 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

32 bit data obtained from device ∗

4.1.2.6 unsigned char ft800memRead8 (unsigned long *ftAddress*)

Funtion to read 8 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

8 bit data obtained from device ∗

4.1.2.7 void ft800memWrite16 (unsigned long *ftAddress,* unsigned int *ftData16*)

Funtion to send 16 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.1.2.8 void ft800memWrite32 (unsigned long *ftAddress,* unsigned long *ftData32*)

Funtion to send 32 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.1.2.9 void ft800memWrite8 (unsigned long *ftAddress,* unsigned char *ftData8*)

Funtion to send 8 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.1.2.10 unsigned char getData ( )

Function getting data from active device with using SPI interface ∗

Returns

8 bit vcalue with obtained value ∗

4.1.2.11 unsigned int incCMDOffset (unsigned int *currentOffset,* unsigned char *commandSize*)

Adds commandSize to the currentOffset. Checks for 4K ring-buffer offset roll-over ∗

Parameters

currentOffset	graphics processor command list pointer ∗
commandSize	number of bytes to increment the offset ∗

Returns

the new ring buffer pointer after adding the command ∗

4.1.2.12 void sendData (int *data*)

Function sending data to active device with using SPI interface ∗

Parameters

data	8 bit value to send to device ∗

4.2 FT800.h File Reference

File containing declarations of all functions required to use with VM800.

```
#import <Arduino.h>
```

Macros

- #define FT_DL_SIZE (8∗1024)
- #define FT_CMD_FIFO_SIZE (4∗1024)
- #define FT_CMD_SIZE (4)
- #define **FT800_VERSION** "1.9.0"
- #define **RAM_CMD** 0x108000UL
- #define **RAM_DL** 0x100000UL
- #define **RAM_G** 0x000000UL
- #define **RAM_PAL** 0x102000UL
- #define **RAM_REG** 0x102400UL
- #define **REG_CLOCK** 0x102408UL
- #define **REG_CMD_DL** 0x1024ecUL
- #define **REG_CMD_READ** 0x1024e4UL

- #define **REG_CMD_WRITE** 0x1024e8UL
- #define **REG_CPURESET** 0x10241cUL
- #define **REG_CSPREAD** 0x102464UL
- #define **REG_DITHER** 0x10245cUL
- #define **REG_DLSWAP** 0x102450UL
- #define **REG_FRAMES** 0x102404UL
- #define **REG_FREQUENCY** 0x10240cUL
- #define **REG_GPIO** 0x102490UL
- #define **REG_GPIO_DIR** 0x10248cUL
- #define **REG_HCYCLE** 0x102428UL
- #define **REG_HOFFSET** 0x10242cUL
- #define **REG_HSIZE** 0x102430UL
- #define **REG_HSYNC0** 0x102434UL
- #define **REG_HSYNC1** 0x102438UL
- #define **REG_ID** 0x102400UL
- #define **REG_INT_EN** 0x10249cUL
- #define **REG_INT_FLAGS** 0x102498UL
- #define **REG_INT_MASK** 0x1024a0UL
- #define **REG_MACRO_0** 0x1024c8UL
- #define **REG_MACRO_1** 0x1024ccUL
- #define **REG_OUTBITS** 0x102458UL
- #define **REG_PCLK** 0x10246cUL
- #define **REG_PCLK_POL** 0x102468UL
- #define **REG_PLAY** 0x102488UL
- #define **REG_PLAYBACK_FORMAT** 0x1024b4UL
- #define **REG_PLAYBACK_FREQ** 0x1024b0UL
- #define **REG_PLAYBACK_LENGTH** 0x1024a8UL
- #define **REG_PLAYBACK_LOOP** 0x1024b8UL
- #define **REG_PLAYBACK_PLAY** 0x1024bcUL
- #define **REG_PLAYBACK_READPTR** 0x1024acUL
- #define **REG_PLAYBACK_START** 0x1024a4UL
- #define **REG_PWM_DUTY** 0x1024c4UL
- #define **REG_PWM_HZ** 0x1024c0UL
- #define **REG_RENDERMODE** 0x102410UL
- #define **REG_ROTATE** 0x102454UL
- #define **REG_SNAPSHOT** 0x102418UL
- #define **REG_SNAPY** 0x102414UL
- #define **REG_SOUND** 0x102484UL
- #define **REG_SWIZZLE** 0x102460UL
- #define **REG_TAG** 0x102478UL
- #define **REG_TAG_X** 0x102470UL
- #define **REG_TAG_Y** 0x102474UL
- #define **REG_TAP_CRC** 0x102420UL
- #define **REG_TAP_MASK** 0x102424UL
- #define **REG_TOUCH_ADC_MODE** 0x1024f4UL
- #define **REG_TOUCH_CHARGE** 0x1024f8UL
- #define **REG_TOUCH_DIRECT_XY** 0x102574UL
- #define **REG_TOUCH_DIRECT_Z1Z2** 0x102578UL
- #define **REG_TOUCH_MODE** 0x1024f0UL
- #define **REG_TOUCH_OVERSAMPLE** 0x102500UL
- #define **REG_TOUCH_RAW_XY** 0x102508UL
- #define **REG_TOUCH_RZ** 0x10250cUL
- #define **REG_TOUCH_RZTHRESH** 0x102504UL
- #define **REG_TOUCH_SCREEN_XY** 0x102510UL
- #define **REG_TOUCH_SETTLE** 0x1024fcUL

- #define **REG_TOUCH_TAG** 0x102518UL
- #define **REG_TOUCH_TAG_XY** 0x102514UL
- #define **REG_TOUCH_TRANSFORM_A** 0x10251cUL
- #define **REG_TOUCH_TRANSFORM_B** 0x102520UL
- #define **REG_TOUCH_TRANSFORM_C** 0x102524UL
- #define **REG_TOUCH_TRANSFORM_D** 0x102528UL
- #define **REG_TOUCH_TRANSFORM_E** 0x10252cUL
- #define **REG_TOUCH_TRANSFORM_F** 0x102530UL
- #define **REG_TRACKER** 0x109000UL
- #define **REG_VCYCLE** 0x10243cUL
- #define **REG_VOFFSET** 0x102440UL
- #define **REG_VOL_PB** 0x10247cUL
- #define **REG_VOL_SOUND** 0x102480UL
- #define **REG_VSIZE** 0x102444UL
- #define **REG_VSYNC0** 0x102448UL
- #define **REG_VSYNC1** 0x10244cUL
- #define **CMDBUF_SIZE** 4096UL
- #define **CMD_APPEND** 0xffffff1eUL
- #define **CMD_BGCOLOR** 0xffffff09UL
- #define **CMD_BUTTON** 0xffffff0dUL
- #define **CMD_CALIBRATE** 0xffffff15UL
- #define **CMD_CLOCK** 0xffffff14UL
- #define **CMD_COLDSTART** 0xffffff32UL
- #define **CMD_DIAL** 0xffffff2dUL
- #define **CMD_DLSTART** 0xffffff00UL
- #define **CMD_FGCOLOR** 0xffffff0aUL
- #define **CMD_GAUGE** 0xffffff13UL
- #define **CMD_GETMATRIX** 0xffffff33UL
- #define **CMD_GETPTR** 0xffffff23UL
- #define **CMD_GRADCOLOR** 0xffffff34UL
- #define **CMD_GRADIENT** 0xffffff0bUL
- #define **CMD_INFLATE** 0xffffff22UL
- #define **CMD_INTERRUPT** 0xffffff02UL
- #define **CMD_KEYS** 0xffffff0eUL
- #define **CMD_LOADIDENTITY** 0xffffff26UL
- #define **CMD_LOADIMAGE** 0xffffff24UL
- #define **CMD_LOGO** 0xffffff31UL
- #define **CMD_MEMCPY** 0xffffff1dUL
- #define **CMD_MEMCRC** 0xffffff18UL
- #define **CMD_MEMSET** 0xffffff1bUL
- #define **CMD_MEMWRITE** 0xffffff1aUL
- #define **CMD_MEMZERO** 0xffffff1cUL
- #define **CMD_NUMBER** 0xffffff2eUL
- #define **CMD_PROGRESS** 0xffffff0fUL
- #define **CMD_REGREAD** 0xffffff19UL
- #define **CMD_ROTATE** 0xffffff29UL
- #define **CMD_SCALE** 0xffffff28UL
- #define **CMD_SCREENSAVER** 0xffffff2fUL
- #define **CMD_SCROLLBAR** 0xffffff11UL
- #define **CMD_SETFONT** 0xffffff2bUL
- #define **CMD_SETMATRIX** 0xffffff2aUL
- #define **CMD_SKETCH** 0xffffff30UL
- #define **CMD_SLIDER** 0xffffff10UL
- #define **CMD_SNAPSHOT** 0xffffff1fUL
- #define **CMD_SPINNER** 0xffffff16UL

- #define **CMD_STOP** 0xffffff17UL
- #define **CMD_SWAP** 0xffffff01UL
- #define **CMD_TEXT** 0xffffff0cUL
- #define **CMD_TOGGLE** 0xffffff12UL
- #define **CMD_TRACK** 0xffffff2cUL
- #define **CMD_TRANSLATE** 0xffffff27UL
- #define **DL_ALPHA_FUNC** 0x09000000UL
- #define **DL_BITMAP_HANDLE** 0x05000000UL
- #define **DL_BITMAP_LAYOUT** 0x07000000UL
- #define **DL_BITMAP_SIZE** 0x08000000UL
- #define **DL_BITMAP_SOURCE** 0x01000000UL
- #define **DL_BITMAP_TFORM_A** 0x15000000UL
- #define **DL_BITMAP_TFORM_B** 0x16000000UL
- #define **DL_BITMAP_TFORM_C** 0x17000000UL
- #define **DL_BITMAP_TFORM_D** 0x18000000UL
- #define **DL_BITMAP_TFORM_E** 0x19000000UL
- #define **DL_BITMAP_TFORM_F** 0x1A000000UL
- #define **DL_BLEND_FUNC** 0x0B000000UL
- #define **DL_BEGIN** 0x1F000000UL
- #define **DL_CALL** 0x1D000000UL
- #define **DL_CLEAR** 0x26000000UL
- #define **DL_CELL** 0x06000000UL
- #define **DL_CLEAR_RGB** 0x02000000UL
- #define **DL_CLEAR_STENCIL** 0x11000000UL
- #define **DL_CLEAR_TAG** 0x12000000UL
- #define **DL_COLOR_A** 0x0F000000UL
- #define **DL_COLOR_MASK** 0x20000000UL
- #define **DL_COLOR_RGB** 0x04000000UL
- #define **DL_DISPLAY** 0x00000000UL
- #define **DL_END** 0x21000000UL
- #define **DL_JUMP** 0x1E000000UL
- #define **DL_LINE_WIDTH** 0x0E000000UL
- #define **DL_MACRO** 0x25000000UL
- #define **DL_POINT_SIZE** 0x0D000000UL
- #define **DL_RESTORE_CONTEXT** 0x23000000UL
- #define **DL_RETURN** 0x24000000UL
- #define **DL_SAVE_CONTEXT** 0x22000000UL
- #define **DL_SCISSOR_SIZE** 0x1C000000UL
- #define **DL_SCISSOR_XY** 0x1B000000UL
- #define **DL_STENCIL_FUNC** 0x0A000000UL
- #define **DL_STENCIL_MASK** 0x13000000UL
- #define **DL_STENCIL_OP** 0x0C000000UL
- #define **DL_TAG** 0x03000000UL
- #define **DL_TAG_MASK** 0x14000000UL
- #define **DL_VERTEX2F** 0x40000000UL
- #define **DL_VERTEX2II** 0x02000000UL
- #define **CLR_COL** 0x4
- #define **CLR_STN** 0x2
- #define **CLR_TAG** 0x1
- #define **DECR** 4UL
- #define **DECR_WRAP** 7UL
- #define **DLSWAP_DONE** 0UL
- #define **DLSWAP_FRAME** 2UL
- #define **DLSWAP_LINE** 1UL
- #define **DST_ALPHA** 3UL

- #define **EDGE_STRIP_A** 7UL
- #define **EDGE_STRIP_B** 8UL
- #define **EDGE_STRIP_L** 6UL
- #define **EDGE_STRIP_R** 5UL
- #define **EQUAL** 5UL
- #define **GEQUAL** 4UL
- #define **GREATER** 3UL
- #define **INCR** 3UL
- #define **INCR_WRAP** 6UL
- #define **INT_CMDEMPTY** 32UL
- #define **INT_CMDFLAG** 64UL
- #define **INT_CONVCOMPLETE** 128UL
- #define **INT_PLAYBACK** 16UL
- #define **INT_SOUND** 8UL
- #define **INT_SWAP** 1UL
- #define **INT_TAG** 4UL
- #define **INT_TOUCH** 2UL
- #define **INVERT** 5UL
- #define **KEEP** 1UL
- #define **L1** 1UL
- #define **L4** 2UL
- #define **L8** 3UL
- #define **LEQUAL** 2UL
- #define **LESS** 1UL
- #define **LINEAR_SAMPLES** 0UL
- #define **LINES** 3UL
- #define **LINE_STRIP** 4UL
- #define **NEAREST** 0UL
- #define **NEVER** 0UL
- #define **NOTEQUAL** 6UL
- #define **ONE** 1UL
- #define **ONE_MINUS_DST_ALPHA** 5UL
- #define **ONE_MINUS_SRC_ALPHA** 4UL
- #define **OPT_CENTER** 1536UL
- #define **OPT_CENTERX** 512UL
- #define **OPT_CENTERY** 1024UL
- #define **OPT_FLAT** 256UL
- #define **OPT_MONO** 1UL
- #define **OPT_NOBACK** 4096UL
- #define **OPT_NODL** 2UL
- #define **OPT_NOHANDS** 49152UL
- #define **OPT_NOHM** 16384UL
- #define **OPT_NOPOINTER** 16384UL
- #define **OPT_NOSECS** 32768UL
- #define **OPT_NOTICKS** 8192UL
- #define **OPT_RIGHTX** 2048UL
- #define **OPT_SIGNED** 256UL
- #define **PALETTED** 8UL
- #define **PLAYCOLOR** 0x00a0a080
- #define FTPOINTS 2UL
- #define **RECTS** 9UL
- #define **REPEAT** 1UL
- #define **REPLACE** 2UL
- #define **RGB332** 4UL
- #define **RGB565** 7UL

- #define **SRC_ALPHA** 2UL
- #define **TEXT8X8** 9UL
- #define **TEXTVGA** 10UL
- #define **TOUCHMODE_CONTINUOUS** 3UL
- #define **TOUCHMODE_FRAME** 2UL
- #define **TOUCHMODE_OFF** 0UL
- #define **TOUCHMODE_ONESHOT** 1UL
- #define **ULAW_SAMPLES** 1UL
- #define **ZERO** 0UL
- #define **RGB**(r, g, b) ((((r) << 16) | (g) << 8) | (b))
- #define **SQ**(v) ((v) * (v))
- #define **MIN**(x, y) ((x) > (y) ? (y) : (x))
- #define **MAX**(x, y) ((x) > (y) ? (x) : (y))
- #define **NOTE**(n, sharp) (((n) - 'C') + ((sharp) * 128))
- #define **F16**(s) (((s) * 65536))
- #define **INVALID_TOUCH_XY** 0x8000
- #define **ABS**(x) ((x) > (0) ? (x) : (-x))
- #define LCD_QVGA
- #define xSDI 8
- #define xSDO 9
- #define xclock 10
- #define xPD 11
- #define xCS 12
- #define FT800_ACTIVE 0x00
- #define FT800_STANDBY 0x41
- #define FT800_SLEEP 0x42
- #define FT800_PWRDOWN 0x50
- #define FT800_CLKEXT 0x44
- #define FT800_CLK48M 0x62
- #define FT800_CLK36M 0x61
- #define FT800_CORERST 0x68
- #define **FT800_GPUACTIVE** 0x40
- #define MEM_WRITE 0x80
- #define MEM_READ 0x00
- #define RED 0xFF0000
- #define GREEN 0x00FF00
- #define BLUE 0x0000FF
- #define WHITE 0xFFFFFF
- #define BLACK 0x000000

Functions

- void delay_us (int us)
- void delay_ms (int ms)
- void sendData (int data)
- unsigned char getData ()
- void ft800memWrite8 (unsigned long ftAddress, unsigned char ftData8)
- void ft800memWrite16 (unsigned long ftAddress, unsigned int ftData16)
- void ft800memWrite32 (unsigned long ftAddress, unsigned long ftData32)
- unsigned char ft800memRead8 (unsigned long ftAddress)
- unsigned char ft800memRead16 (unsigned long ftAddress)
- unsigned long ft800memRead32 (unsigned long ftAddress)
- unsigned int incCMDOffset (unsigned int currentOffset, unsigned char commandSize)
- void ft800cmdWrite (unsigned char ftCommand)

4.2.1 Detailed Description

File containing declarations of all functions required to use with VM800.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.2.2 Macro Definition Documentation

4.2.2.1 #define BLACK 0x000000

Black colour

4.2.2.2 #define BLUE 0x0000FF

Blue colour

4.2.2.3 #define FT800_ACTIVE 0x00

Initializes FT800

4.2.2.4 #define FT800_CLK36M 0x61

Select 36MHz PLL

4.2.2.5 #define FT800_CLK48M 0x62

Select 48MHz PLL

4.2.2.6 #define FT800_CLKEXT 0x44

Select external clock source

4.2.2.7 #define FT800_CORERST 0x68

Reset core - all registers default

4.2.2.8 #define FT800_PWRDOWN 0x50

Place FT800 in Power Down (core off)

4.2.2.9 #define FT800_SLEEP 0x42

Place FT800 in Sleep (clk off)

4.2.2.10 #define FT800_STANDBY 0x41

Place FT800 in Standby (clk running)

4.2.2.11 #define FT_CMD_FIFO_SIZE (4∗1024)

4KB coprocessor Fifo size

4.2.2.12 #define FT_CMD_SIZE (4)

4 byte per coprocessor command of EVE

4.2.2.13 #define FT_DL_SIZE (8∗1024)

8KB Display List buffer size

4.2.2.14 #define FTPOINTS 2UL

"POINTS" is a reserved word

4.2.2.15 #define GREEN 0x00FF00

Green colour

4.2.2.16 #define LCD_QVGA

QVGA = 320 x 240 (VM800B/C 3.5")

4.2.2.17 #define MEM_READ 0x00

FT800 Host Memory Read

4.2.2.18 #define MEM_WRITE 0x80

FT800 Host Memory Write

4.2.2.19 #define RED 0xFF0000

Red colour

4.2.2.20 #define WHITE 0xFFFFFF

White colour

4.2.2.21 #define xclock 10

Clock line - output for Galileo

4.2.2.22 #define xCS 12

Chip Select line for screen - output for Galileo

4.2.2.23 #define xPD 11

PD line for screen - output for Galileo

4.2.2.24 #define xSDI 8

SDI line for SPI interface - input for Galileo

4.2.2.25 #define xSDO 9

SDO line for SPI interface - output for Galileo

4.2.3 Function Documentation

4.2.3.1 void delay_ms (int *ms*)

Pauses the program for the amount of time (in milisecond) specified as parameter ∗

Parameters

ms	milisecond to delay ∗

4.2.3.2 void delay_us (int *us*)

Pauses the program for the amount of time (in microsecond) specified as parameter ∗

Parameters

us	microseconds to delay ∗

4.2.3.3 void ft800cmdWrite (unsigned char *ftCommand*)

Sends FT800 command ∗

Parameters

ftCommand	command to send to device ∗

4.2.3.4 unsigned char ft800memRead16 (unsigned long *ftAddress*)

Funtion to read 16 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

16 bit data obtained from device ∗

4.2.3.5 unsigned long ft800memRead32 (unsigned long *ftAddress*)

Funtion to read 32 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

32 bit data obtained from device ∗

4.2.3.6 unsigned char ft800memRead8 (unsigned long *ftAddress*)

Funtion to read 8 bit value from active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗

Returns

8 bit data obtained from device ∗

4.2.3.7 void ft800memWrite16 (unsigned long *ftAddress,* unsigned int *ftData16*)

Funtion to send 16 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.2.3.8 void ft800memWrite32 (unsigned long *ftAddress,* unsigned long *ftData32*)

Funtion to send 32 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.2.3.9 void ft800memWrite8 (unsigned long *ftAddress,* unsigned char *ftData8*)

Funtion to send 8 bit value to active device with using SPI interface ∗

Parameters

ftAddress	FT800 memory space address (24 bits) ∗
ftData8	a byte to send ∗

4.2.3.10 unsigned char getData ( )

Function getting data from active device with using SPI interface ∗

Returns

8 bit vcalue with obtained value ∗

4.2.3.11 unsigned int incCMDOffset (unsigned int *currentOffset,* unsigned char *commandSize*)

Adds commandSize to the currentOffset. Checks for 4K ring-buffer offset roll-over ∗

Parameters

currentOffset	graphics processor command list pointer ∗
commandSize	number of bytes to increment the offset ∗

Returns

the new ring buffer pointer after adding the command ∗

4.2.3.12 void sendData (int *data*)

Function sending data to active device with using SPI interface ∗

Parameters

data	8 bit value to send to device ∗

4.3 FT800api.cpp File Reference

File containing declarations of all API functions for VM800.

```
#include "FT800api.h"
```

Functions

- void initScreen ()
- void autko ()
- void mainScreen ()
- void smartMirrorScreen ()
- void opctionsScreen ()
- void spinner (int16_t x, int16_t y, uint16_t style, uint16_t scale)
- void button (int16_t x, int16_t y, int16_t w, int16_t h, int16_t font, uint16_t options, const char ∗str)
- void text (int16_t x, int16_t y, int16_t font, uint16_t options, const char ∗str)
- void **number** (int16_t x, int16_t y, int16_t font, uint16_t options, int value)
- void line (unsigned long color, unsigned long line_x1, unsigned long line_y1, unsigned long line_x2, unsigned long line_y2, unsigned long width)
- void dot (unsigned long color, unsigned int point_size, unsigned long point_x, unsigned long point_y)
- void calibrate ()
- void start (unsigned long color)
- void show ()

4.3.1 Detailed Description

File containing declarations of all API functions for VM800.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.3.2 Function Documentation

4.3.2.1 void autko ( )

Function drawing car with proper attributes ∗

4.3.2.2 void button (int16_t *x,* int16_t *y,* int16_t *w,* int16_t *h,* int16_t *font,* uint16_t *options,* const char ∗ *str*)

Function which draw a button on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
w	width for the button ∗
h	height for the button ∗
font	font fort the button text ∗
options	options for the button ∗
str	text to draw inside button ∗

4.3.2.3 void calibrate ( )

Function which calibrate screen ∗

4.3.2.4 void dot (unsigned long *color,* unsigned int *point_size,* unsigned long *point_x,* unsigned long *point_y*)

Function which draw a dot on the screen ∗

Parameters

color	dot color ∗
point_size	size for the dot ∗
point_x	x-coordinate for the dot ∗
point_y	y-coordinate for the dot ∗

4.3.2.5 void initScreen ( )

Function showing init screen durig main screen is loading ∗

4.3.2.6 void line (unsigned long *color,* unsigned long *line_x1,* unsigned long *line_y1,* unsigned long *line_x2,* unsigned long *line_y2,* unsigned long *width*)

Function which draw a line on the screen ∗

Parameters

color	line color ∗
line_x1	x-coordinate for the beginning of the first end ∗
line_y1	y-coordinate for the beginning of the first end ∗
line_x2	x-coordinate for the beginning of the second end ∗
line_y2	y-coordinate for the beginning of the second end ∗
width	line width ∗

4.3.2.7 void mainScreen ( )

Function showing main screen ∗

4.3.2.8 void opctionsScreen ( )

Function showing options screen ∗

4.3.2.9 void show ( )

Function showing prerared screen from buffor ∗

4.3.2.10 void smartMirrorScreen ( )

Function showing smart mirror screen ∗

4.3.2.11 void spinner (int16_t *x,* int16_t *y,* uint16_t *style,* uint16_t *scale*)

Function which draw a spinner on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
style	look swcreen ∗
scale	size of spinner ∗

4.3.2.12 void start (unsigned long *color*)

Function which start inicjalize new screen ∗

Parameters

color	backgroud color ∗

4.3.2.13 void text (int16_t *x*, int16_t *y*, int16_t *font*, uint16_t *options*, const char ∗ *str*)

Function which draw a text on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
font	font for the text ∗
options	options to set for the text ∗
str	text to draw on the screen ∗

4.4 FT800api.h File Reference

File containing declarations of all API functions for VM800.

```
#include "FT800.h"
#include "simulator.h"
#import <Arduino.h>
```

Functions

- void initScreen ()
- void opctionsScreen ()
- void mainScreen ()
- void smartMirrorScreen ()
- void spinner (int16_t x, int16_t y, uint16_t style, uint16_t scale)
- void button (int16_t x, int16_t y, int16_t w, int16_t h, int16_t font, uint16_t options, const char ∗str)
- void text (int16_t x, int16_t y, int16_t font, uint16_t options, const char ∗str)
- void line (unsigned long color, unsigned long line_x1, unsigned long line_y1, unsigned long line_x2, unsigned long line_y2, unsigned long width)
- void dot (unsigned long color, unsigned int point_size, unsigned long point_x, unsigned long point_y)
- void start (unsigned long color)
- void number (int16_t x, int16_t y, int16_t font, uint16_t options, int32_t value)
- void show ()
- void calibrate ()
- void autko ()

Variables

- unsigned int **cmdOffset**
- unsigned int **cmdBufferRd**
- unsigned int **cmdBufferWr**
- struct car * **audi**
- int **timeR**

4.4.1 Detailed Description

File containing declarations of all API functions for VM800.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.4.2 Function Documentation

4.4.2.1 void autko ( )

Function drawing car with proper attributes *

4.4.2.2 void button (int16_t *x*, int16_t *y*, int16_t *w*, int16_t *h*, int16_t *font*, uint16_t *options*, const char * *str*)

Function which draw a button on the screen *

Parameters

x	x-coordinate on the screen *
y	y-coordinate on the screen *
w	width for the button *
h	height for the button *
font	font fort the button text *
options	options for the button *
str	text to draw inside button *

4.4.2.3 void calibrate ( )

Function which calibrate screen *

4.4.2.4 void dot (unsigned long *color,* unsigned int *point_size,* unsigned long *point_x,* unsigned long *point_y*)

Function which draw a dot on the screen ∗

Parameters

color	dot color ∗
point_size	size for the dot ∗
point_x	x-coordinate for the dot ∗
point_y	y-coordinate for the dot ∗

4.4.2.5 void initScreen ( )

Function showing init screen durig main screen is loading ∗

4.4.2.6 void line (unsigned long *color,* unsigned long *line_x1,* unsigned long *line_y1,* unsigned long *line_x2,* unsigned long *line_y2,* unsigned long *width*)

Function which draw a line on the screen ∗

Parameters

color	line color ∗
line_x1	x-coordinate for the beginning of the first end ∗
line_y1	y-coordinate for the beginning of the first end ∗
line_x2	x-coordinate for the beginning of the second end ∗
line_y2	y-coordinate for the beginning of the second end ∗
width	line width ∗

4.4.2.7 void mainScreen ( )

Function showing main screen ∗

4.4.2.8 void number (int16_t *x,* int16_t *y,* int16_t *font,* uint16_t *options,* int32_t *value*)

Function which draw a number on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
font	font for the number ∗
options	options to set for the number ∗
value	value to draw on the screen ∗

4.4.2.9 void opctionsScreen ( )

Function showing options screen ∗

4.4.2.10 void show ( )

Function showing prerared screen from buffor ∗

4.4.2.11 void smartMirrorScreen ( )

Function showing smart mirror screen ∗

4.4.2.12 void spinner (int16_t *x*, int16_t *y*, uint16_t *style*, uint16_t *scale*)

Function which draw a spinner on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
style	look swcreen ∗
scale	size of spinner ∗

4.4.2.13 void start (unsigned long *color*)

Function which start inicjalize new screen ∗

Parameters

color	backgroud color ∗

4.4.2.14 void text (int16_t *x*, int16_t *y*, int16_t *font*, uint16_t *options*, const char ∗ *str*)

Function which draw a text on the screen ∗

Parameters

x	x-coordinate on the screen ∗
y	y-coordinate on the screen ∗
font	font for the text ∗
options	options to set for the text ∗
str	text to draw on the screen ∗

4.5 I2C.cpp File Reference

File containing declarations of function to read data with using I2C protocol.

```
#include "I2C.h"
```

Functions

- int readPCF (char adres)

Variables

- int d = 1

4.5.1 Detailed Description

File containing declarations of function to read data with using I2C protocol.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.5.2 Function Documentation

4.5.2.1 int readPCF (char *adres*)

Reading value from PCF8574N I/O Expander ∗

Parameters

adres	The address of PCF8574N I/O Expander ∗

Returns

Value from the specified PCF8574N I/O Expander ∗

4.5.3 Variable Documentation

4.5.3.1 int d = 1

Delay time - for PCF handing

4.6 I2C.h File Reference

File containing declarations of function to read data with using I2C protocol.

```
#import <Arduino.h>
```

Macros

- #define sda 7
- #define scl 6
- #define pinInt0 2

Functions

- int readPCF (char adres)

4.6.1 Detailed Description

File containing declarations of function to read data with using I2C protocol.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.6.2 Macro Definition Documentation

4.6.2.1 #define pinInt0 2

Interrput port number

4.6.2.2 #define scl 6

SCL port number

4.6.2.3 #define sda 7

SDA port number

4.6.3 Function Documentation

4.6.3.1 int readPCF (char *adres*)

Reading value from PCF8574N I/O Expander ∗

Parameters

adres	The address of PCF8574N I/O Expander ∗

Returns

Value from the specified PCF8574N I/O Expander ∗

4.7 simulator.cpp File Reference

File containing declarations of all functions required to communication with car simulator.

```
#include "simulator.h"
```

Functions

- void printObj (struct car ∗obj, char ∗d)
- int readTemp (int portNumber)
- void save (struct car ∗audi, struct car ∗tmp)
- struct car ∗ readData ()
- void checkChangesDigital ()
- void sendData ()
- void **checkChangesAnalog** ()

4.7.1 Detailed Description

File containing declarations of all functions required to communication with car simulator.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.7.2 Function Documentation

4.7.2.1 void checkChangesDigital ( )

Check if sth on digital ports was changed ∗

4.7.2.2 void printObj (struct car ∗ *obj,* char ∗ *d*)

Debug function to print car structure on a serial monitor ∗ console and to log file on SD car ∗

Parameters

Car	struct to print and save with selected format into file∗
d	actual date ∗

4.7.2.3 struct car∗ readData ( )

Reading data about car status ∗

4.7.2.4 int readTemp (int *portNumber*)

Reading value from analog ports (temperatures) ∗

Parameters

portNumber	The number of the analog input pin to read ∗

Returns

Value from the specified analog pin ∗

4.7.2.5 void save (struct car ∗ *audi*, struct car ∗ *tmp*)

Copying data function from temporary to main struct ∗

Parameters

∗*audi*,∗*tmp*	Structures to and from which data are copied ∗

4.7.2.6 void sendData ( )

Sending actial data to web server ∗

4.8 simulator.h File Reference

File containing declarations of all functions required to communication with car simulator.

```
#import <Arduino.h>
#include "I2C.h"
#include <stdio.h>
#include "FT800api.h"
```

Classes

- struct car

Functions

- void printObj (struct car *obj, char *d)
- void checkChangesAnalog (struct car *audi)
- void checkChangesDigital ()
- struct car * readData ()
- void save (struct car *audi, struct car *tmp)
- int readTemp (int portNumber)
- void sendData ()

Variables

- struct car * **audi**
- int **dataFormat**
- int **saveData**
- short int **screenNR**

4.8.1 Detailed Description

File containing declarations of all functions required to communication with car simulator.

Author

Daniel Sienkiewicz

Date

28 February 2016

4.8.2 Function Documentation

4.8.2.1 void checkChangesAnalog (struct car * *audi*)

Check if sth on analog ports was changed *

Parameters

audi	structure to save dage read from analog sensors *

4.8.2.2 void checkChangesDigital ( )

Check if sth on digital ports was changed *

4.8.2.3 void printObj (struct car * obj, char * d)

Debug function to print car structure on a serial monitor * console and to log file on SD car *

Parameters

Car	struct to print and save with selected format into file*
d	actual date *

4.8.2.4 struct car* readData ( )

Reading data about car status *

4.8.2.5 int readTemp (int portNumber)

Reading value from analog ports (temperatures) *

Parameters

portNumber	The number of the analog input pin to read *

Returns

Value from the specified analog pin *

4.8.2.6 void save (struct car * audi, struct car * tmp)

Copying data function from temporary to main struct *

Parameters

*audi,*tmp*	Structures to and from which data are copied *

4.8.2.7 void sendData ( )

Sending actial data to web server *

Index

autko
    FT800api.cpp, 22
    FT800api.h, 25

BLACK
    FT800.h, 16
BLUE
    FT800.h, 16
button
    FT800api.cpp, 22
    FT800api.h, 25

calibrate
    FT800api.cpp, 22
    FT800api.h, 25
car, 5
    doors, 5
    lights, 5
    r, 5
    seatbelts, 6
    tempEngine, 6
    tempIn, 6
    tempOut, 6
checkChangesAnalog
    simulator.h, 33
checkChangesDigital
    simulator.cpp, 31
    simulator.h, 33

d
    I2C.cpp, 29
delay_ms
    FT800.cpp, 7
    FT800.h, 18
delay_us
    FT800.cpp, 8
    FT800.h, 19
doors
    car, 5
dot
    FT800api.cpp, 22
    FT800api.h, 25

FT800.cpp, 7
    delay_ms, 7
    delay_us, 8
    ft800cmdWrite, 8
    ft800memRead16, 8
    ft800memRead32, 8
    ft800memRead8, 8
    ft800memWrite16, 9
    ft800memWrite32, 9
    ft800memWrite8, 9
    getData, 9
    incCMDOffset, 10
    sendData, 10
FT800.h, 10
    BLACK, 16
    BLUE, 16
    delay_ms, 18
    delay_us, 19
    FT800_ACTIVE, 16
    FT800_CLK36M, 16
    FT800_CLK48M, 16
    FT800_CLKEXT, 16
    FT800_CORERST, 16
    FT800_PWRDOWN, 16
    FT800_SLEEP, 17
    FT800_STANDBY, 17
    FT_CMD_FIFO_SIZE, 17
    FT_CMD_SIZE, 17
    FT_DL_SIZE, 17
    FTPOINTS, 17
    ft800cmdWrite, 19
    ft800memRead16, 19
    ft800memRead32, 19
    ft800memRead8, 19
    ft800memWrite16, 20
    ft800memWrite32, 20
    ft800memWrite8, 20
    GREEN, 17
    getData, 20
    incCMDOffset, 20
    LCD_QVGA, 17
    MEM_READ, 17
    MEM_WRITE, 17
    RED, 18
    sendData, 21
    WHITE, 18
    xCS, 18
    xPD, 18
    xSDI, 18
    xSDO, 18
    xclock, 18
FT800_ACTIVE
    FT800.h, 16
FT800_CLK36M
    FT800.h, 16
FT800_CLK48M

FT800.h, 16
FT800_CLKEXT
FT800.h, 16
FT800_CORERST
FT800.h, 16
FT800_PWRDOWN
FT800.h, 16
FT800_SLEEP
FT800.h, 17
FT800_STANDBY
FT800.h, 17
FT800api.cpp, 21
autko, 22
button, 22
calibrate, 22
dot, 22
initScreen, 23
line, 23
mainScreen, 23
opctionsScreen, 23
show, 23
smartMirrorScreen, 23
spinner, 23
start, 24
text, 24
FT800api.h, 24
autko, 25
button, 25
calibrate, 25
dot, 25
initScreen, 27
line, 27
mainScreen, 27
number, 27
opctionsScreen, 27
show, 27
smartMirrorScreen, 28
spinner, 28
start, 28
text, 28
FT_CMD_FIFO_SIZE
FT800.h, 17
FT_CMD_SIZE
FT800.h, 17
FT_DL_SIZE
FT800.h, 17
FTPOINTS
FT800.h, 17
ft800cmdWrite
FT800.cpp, 8
FT800.h, 19
ft800memRead16
FT800.cpp, 8
FT800.h, 19
ft800memRead32
FT800.cpp, 8
FT800.h, 19
ft800memRead8
FT800.cpp, 8
FT800.h, 19
ft800memWrite16
FT800.cpp, 9
FT800.h, 20
ft800memWrite32
FT800.cpp, 9
FT800.h, 20
ft800memWrite8
FT800.cpp, 9
FT800.h, 20

GREEN
FT800.h, 17
getData
FT800.cpp, 9
FT800.h, 20

I2C.cpp, 28
d, 29
readPCF, 29
I2C.h, 30
pinInt0, 30
readPCF, 30
scl, 30
sda, 30
incCMDOffset
FT800.cpp, 10
FT800.h, 20
initScreen
FT800api.cpp, 23
FT800api.h, 27

LCD_QVGA
FT800.h, 17
lights
car, 5
line
FT800api.cpp, 23
FT800api.h, 27

MEM_READ
FT800.h, 17
MEM_WRITE
FT800.h, 17
mainScreen
FT800api.cpp, 23
FT800api.h, 27

number
FT800api.h, 27

opctionsScreen
FT800api.cpp, 23
FT800api.h, 27

pinInt0
I2C.h, 30
printObj
simulator.cpp, 31

simulator.h, 33

r
car, 5
RED
FT800.h, 18
readData
simulator.cpp, 32
simulator.h, 34
readPCF
I2C.cpp, 29
I2C.h, 30
readTemp
simulator.cpp, 32
simulator.h, 34

save
simulator.cpp, 32
simulator.h, 34
scl
I2C.h, 30
sda
I2C.h, 30
seatbelts
car, 6
sendData
FT800.cpp, 10
FT800.h, 21
simulator.cpp, 32
simulator.h, 34
show
FT800api.cpp, 23
FT800api.h, 27
simulator.cpp, 31
checkChangesDigital, 31
printObj, 31
readData, 32
readTemp, 32
save, 32
sendData, 32
simulator.h, 32
checkChangesAnalog, 33
checkChangesDigital, 33
printObj, 33
readData, 34
readTemp, 34
save, 34
sendData, 34
smartMirrorScreen
FT800api.cpp, 23
FT800api.h, 28
spinner
FT800api.cpp, 23
FT800api.h, 28
start
FT800api.cpp, 24
FT800api.h, 28

tempEngine
car, 6
tempIn
car, 6
tempOut
car, 6
text
FT800api.cpp, 24
FT800api.h, 28

WHITE
FT800.h, 18

xCS
FT800.h, 18
xPD
FT800.h, 18
xSDI
FT800.h, 18
xSDO
FT800.h, 18
xclock
FT800.h, 18