

## Stopwatch Autogen

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules . . . . .	1
<b>2</b>	<b>Data Structure Index</b>	<b>3</b>
2.1	Data Structures . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	Utility . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Function Documentation . . . . .	7
4.1.2.1	strencode1digit(char *str, int digit) . . . . .	7
4.1.2.2	strencode2digit(char *str, int digit) . . . . .	8
4.1.2.3	updateScreen(uint8_T om, uint8_T m) . . . . .	8
4.1.2.4	updateTime(uint8_T *oh, uint8_T *om, uint8_T *os, uint8_T *ot, uint8_T oldmode) . . . . .	8
4.2	Interrupt Handler . . . . .	9
4.2.1	Detailed Description . . . . .	9
4.3	Tasks . . . . .	10
4.3.1	Detailed Description . . . . .	10
4.3.2	Function Documentation . . . . .	10
4.3.2.1	main(void) . . . . .	10
4.3.2.2	TASK(TaskLCD) . . . . .	10
4.3.2.3	TASK(TaskClock) . . . . .	10

4.4	Widget	11
4.4.1	Detailed Description	12
4.4.2	Function Documentation	12
4.4.2.1	contains(Widget *w, TPoint *point)	12
4.4.2.2	DrawInit(Widget ws[])	12
4.4.2.3	DrawOff(Widget *w)	13
4.4.2.4	DrawOn(Widget *w)	13
4.4.2.5	OnTouch(const Widget ws[], TPoint *press)	13
4.4.2.6	WPrint(Widget *w, char *s)	14
4.5	Widget Definitions	15
4.5.1	Detailed Description	15
4.5.2	Variable Documentation	15
4.5.2.1	alarm_b	15
4.5.2.2	alarm_exp_i	15
4.5.2.3	backg	16
4.5.2.4	hrs_back	16
4.5.2.5	min_back	16
4.5.2.6	minus_b	16
4.5.2.7	MyWatchScr	16
4.5.2.8	plus_b	17
4.5.2.9	reset_b	17
4.5.2.10	resume_b	17
4.5.2.11	sec_back	17
4.5.2.12	set_b	17
4.5.2.13	start_b	17
4.5.2.14	stop_b	18
4.5.2.15	swatch_b	18
4.5.2.16	timer_b	18
4.5.2.17	timer_exp_i	18
4.5.2.18	tts_back	18
4.5.2.19	txt	18
4.5.2.20	watch_b	18
4.6	Events	19
4.6.1	Detailed Description	19
4.6.2	Macro Definition Documentation	19
4.6.2.1	ClearEvt	19
4.6.2.2	IsEvent	19
4.6.2.3	SetEvt	20

<b>5</b>	<b>Data Structure Documentation</b>	<b>21</b>
5.1	Icon Struct Reference . . . . .	21
5.2	Image Struct Reference . . . . .	21
5.3	Text Struct Reference . . . . .	21
5.4	Widget Struct Reference . . . . .	22
<b>6</b>	<b>File Documentation</b>	<b>23</b>
6.1	code.c File Reference . . . . .	23
6.1.1	Detailed Description . . . . .	24
6.2	Event.c File Reference . . . . .	25
6.2.1	Detailed Description . . . . .	25
6.3	Event.h File Reference . . . . .	25
6.3.1	Detailed Description . . . . .	26
6.4	mypictures.c File Reference . . . . .	26
6.4.1	Detailed Description . . . . .	27
6.5	mypictures.h File Reference . . . . .	27
6.5.1	Detailed Description . . . . .	28
6.6	Widget.c File Reference . . . . .	28
6.6.1	Detailed Description . . . . .	29
6.7	Widget.h File Reference . . . . .	29
6.7.1	Detailed Description . . . . .	31
	<b>Index</b>	<b>33</b>



# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Utility . . . . .	7
Interrupt Handler . . . . .	9
Tasks . . . . .	10
Widget . . . . .	11
Widget Definitions . . . . .	15
Events . . . . .	19





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">Icon</a>	21
<a href="#">Image</a>	21
<a href="#">Text</a>	21
<a href="#">Widget</a>	22



## Chapter 3

# File Index

### 3.1 File List

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

<a href="#">code.c</a>	Contains the body of all tasks and the global variables defined . . . . .	23
<a href="#">Event.c</a>	Contains the event mask definition . . . . .	25
<a href="#">Event.h</a>	Contains the macros used to handle the event masks . . . . .	25
<a href="#">mypictures.c</a>	This file contains the application pictures in RGB565 format . . . . .	26
<a href="#">mypictures.h</a>	Pictures header file . . . . .	27
<a href="#">Widget.c</a>	Contains the functions to manage the widgets on the screen . . . . .	28
<a href="#">Widget.h</a>	Contains the type definitions and the macros used for the screen widgets . . . . .	29



# Chapter 4

## Module Documentation

### 4.1 Utility

#### Functions

- static void [strencode1digit](#) (char \*str, int digit)  
*Converts a one digit integer into a string.*
- static void [strencode2digit](#) (char \*str, int digit)  
*Converts a two digits integer into a string.*
- static void [updateTime](#) (uint8\_T \*oh, uint8\_T \*om, uint8\_T \*os, uint8\_T \*ot, uint8\_T oldmode)  
*Updates the time on the screen.*
- static void [updateScreen](#) (uint8\_T om, uint8\_T m)  
*Updates the screen widgets.*

#### 4.1.1 Detailed Description

#### 4.1.2 Function Documentation

##### 4.1.2.1 static void [strencode1digit](#) ( char \* *str*, int *digit* ) [static]

Converts a one digit integer into a string.

#### Parameters

<i>str</i>	pointer to the returning string.
<i>digit</i>	integer digit to be converted.

#### Return values

<i>None</i>	
-------------	--

#### 4.1.2.2 static void strencode2digit ( char \* *str*, int *digit* ) [static]

Converts a two digits integer into a string.

##### Parameters

<i>str</i>	pointer to the returning string.
<i>digit</i>	integer digits to be converted.

##### Return values

<i>None</i>	
-------------	--

#### 4.1.2.3 static void updateScreen ( uint8\_T *om*, uint8\_T *m* ) [static]

Updates the screen widgets.

##### Parameters

<i>om</i>	Old application mode.
<i>m</i>	New application mode.

##### Return values

<i>None</i>	
-------------	--

#### 4.1.2.4 static void updateTime ( uint8\_T \* *oh*, uint8\_T \* *om*, uint8\_T \* *os*, uint8\_T \* *ot*, uint8\_T *oldmode* ) [static]

Updates the time on the screen.

##### Parameters

<i>oh</i>	Old hours.
<i>om</i>	Old minutes.
<i>os</i>	Old seconds.
<i>ot</i>	Old tenths.
<i>oldmode</i>	Old application mode.

##### Return values

<i>None</i>	
-------------	--

## 4.2 Interrupt Handler

### Functions

- [ISR2](#) (systick\_handler)  
*System Tick interrupt handler.*

### 4.2.1 Detailed Description

## 4.3 Tasks

### Functions

- [TASK](#) (TaskLCD)  
*LDC task body.*
- [TASK](#) (TaskClock)  
*Clock task body.*
- int [main](#) (void)  
*Main task of the application.*

#### 4.3.1 Detailed Description

#### 4.3.2 Function Documentation

##### 4.3.2.1 int main ( void )

Main task of the application.

##### Parameters

<i>None</i>	
-------------	--

##### Return values

<i>None</i>	This function should never return.
-------------	------------------------------------

##### 4.3.2.2 TASK ( TaskLCD )

LDC task body.

This task is periodically activated in order to get the touch events.

##### 4.3.2.3 TASK ( TaskClock )

Clock task body.

This is the most important task. It dispatches events to the state machine and catches the button presses events.



## 4.4 Widget

### Modules

- [Widget Definitions](#)

### Data Structures

- struct [Image](#)
- struct [Icon](#)
- struct [Text](#)
- struct [Widget](#)

### Macros

- `#define NUMWIDGETS 25`
- `#define BAKCG 0`
- `#define BWATCH 1`
- `#define BSWATCH 2`
- `#define BALARM 3`
- `#define BTIMER 4`
- `#define BPLUS 5`
- `#define BMINUS 6`
- `#define BSTART 7`
- `#define BSET 8`
- `#define BRESUME 9`
- `#define BSTOP 10`
- `#define BRESET 11`
- `#define ALARMEXP 12`
- `#define TIMEREXP 13`
- `#define HRSSTR 14`
- `#define MINSTR 15`
- `#define SECSTR 16`
- `#define TTSSTR 17`
- `#define SEP1STR 18`
- `#define SEP2STR 19`
- `#define TTSSEP 20`
- `#define HRSBKG 21`
- `#define MINBKG 22`
- `#define SECBKG 23`
- `#define TTSBKG 24`
- `#define NOEVENT 0x00`
- `#define WATCHBPRESS 0x01`
- `#define SWATCHBPRESS 0x02`
- `#define ALARMBPRESS 0x04`
- `#define TIMERBPRESS 0x08`
- `#define PLUSBPRESS 0x10`
- `#define MINUSBPRESS 0x20`
- `#define STARTBPRESS 0x40`
- `#define STOPBPRESS 0x80`
- `#define WATCHMODE 0`
- `#define SWATCHMODE 1`
- `#define ALARMMODE 2`
- `#define TIMERMODE 3`
- `#define txtinfo(w) ((Text *)((w)->ws))`
- `#define iconinfo(w) ((Icon *)((w)->ws))`
- `#define imginfo(w) ((Image *)((w)->ws))`

## Enumerations

- enum **WidgetType** { **BACKGROUND**, **ICON**, **TEXT**, **IMAGE** }

## Functions

- unsigned char **contains** (**Widget** \*w, **TPoint** \*point)  
*Checks if the touched point is inside a widget.*
- unsigned char **OnTouch** (const **Widget** ws[], **TPoint** \*press)  
*Handles the touch event.*
- void **DrawInit** (**Widget** ws[])  
*Draws the initial GUI of the application.*
- unsigned char **DrawOn** (**Widget** \*w)  
*Draws the 'on' image of a widget.*
- unsigned char **DrawOff** (**Widget** \*w)  
*Draws the 'off' image of a widget.*
- unsigned char **WPrint** (**Widget** \*w, char \*s)  
*Prints a string on the screen.*

### 4.4.1 Detailed Description

### 4.4.2 Function Documentation

#### 4.4.2.1 unsigned char contains ( **Widget** \* w, **TPoint** \* point )

Checks if the touched point is inside a widget.

##### Parameters

<i>w</i>	Pointer to the widget.
<i>point</i>	Pointer to the coordinates data structure.

##### Return values

<i>1</i>	The point is inside the widget.
<i>0</i>	The point is outside the widget.

#### 4.4.2.2 void DrawInit ( **Widget** ws[] )

Draws the initial GUI of the application.

##### Parameters

<i>ws</i>	Pointer to the application widgets array.
-----------	---

## Return values

<i>None.</i>	
--------------	--

## 4.4.2.3 unsigned char DrawOff ( Widget \* w )

Draws the 'off' image of a widget.

## Parameters

<i>w</i>	Pointer to the widget structure.
----------	----------------------------------

## Return values

<i>1</i>	The image was successfully drawn on the screen.
<i>0</i>	Unable to draw the image.

## 4.4.2.4 unsigned char DrawOn ( Widget \* w )

Draws the 'on' image of a widget.

## Parameters

<i>w</i>	Pointer to the widget structure.
----------	----------------------------------

## Return values

<i>1</i>	The image was successfully drawn on the screen.
<i>0</i>	Unable to draw the image.

## 4.4.2.5 unsigned char OnTouch ( const Widget ws[], TPoint \* press )

Handles the touch event.

## Parameters

<i>ws</i>	Pointer to the application widgets array.
<i>press</i>	Pointer to the coordinates data structure.

## Return values

<i>1</i>	The touched point is inside one application widget
<i>0</i>	No widget in the application contains the touched point.

This function scans the entire widget array defined for the application and for each of them checks whether the coordinates of the touched point are inside the widget.

#### 4.4.2.6 unsigned char WPrint ( Widget \* *w*, char \* *s* )

Prints a string on the screen.

##### Parameters

<i>w</i>	Pointer to the widget data structure.
<i>s</i>	Pointer to the string which have to be printed.

## 4.5 Widget Definitions

### Variables

- [Icon](#) `watch_b`
- [Icon](#) `swatch_b`
- [Icon](#) `alarm_b`
- [Icon](#) `timer_b`
- [Icon](#) `plus_b`
- [Icon](#) `minus_b`
- [Icon](#) `start_b`
- [Icon](#) `stop_b`
- [Icon](#) `set_b`
- [Icon](#) `reset_b`
- [Icon](#) `resume_b`
- [Icon](#) `alarm_exp_i`
- [Icon](#) `timer_exp_i`
- [Image](#) `hrs_back`
- [Image](#) `min_back`
- [Image](#) `sec_back`
- [Image](#) `tts_back`
- [Text](#) `txt`
- [Image](#) `backg`
- [Widget](#) `MyWatchScr` [NUMWIDGETS]

*This array contains alle the widgets defined for the application.*

### 4.5.1 Detailed Description

### 4.5.2 Variable Documentation

#### 4.5.2.1 [Icon](#) `alarm_b`

##### Initial value:

```
= {
    b_alarm_on, b_alarm_off, ALARMBPRESS
}
```

#### 4.5.2.2 [Icon](#) `alarm_exp_i`

##### Initial value:

```
= {
    alarm_exp_on, alarm_exp_off, NOEVENT
}
```

#### 4.5.2.3 Image backg

**Initial value:**

```
= {
    bkg
}
```

#### 4.5.2.4 Image hrs\_back

**Initial value:**

```
= {
    hrs_bkg
}
```

#### 4.5.2.5 Image min\_back

**Initial value:**

```
= {
    min_bkg
}
```

#### 4.5.2.6 Icon minus\_b

**Initial value:**

```
= {
    b_minus, hide_minus, MINUSBPRESS
}
```

#### 4.5.2.7 Widget MyWatchScr[NUMWIDGETS]

**Initial value:**

```
= {
    {0, 0, 320, 240, BACKGROUND, (void *)&backg},
    {0, 0, 80, 45, ICON, (void *)&watch_b},
    {80, 0, 80, 45, ICON, (void *)&swatch_b},
    {160, 0, 80, 45, ICON, (void *)&alarm_b},
    {240, 0, 80, 45, ICON, (void *)&timer_b},
    {142, 125, 36, 35, ICON, (void *)&plus_b},
    {142, 196, 36, 35, ICON, (void *)&minus_b},
    {30, 160, 100, 40, ICON, (void *)&start_b},
    {30, 160, 100, 40, ICON, (void *)&set_b},
    {30, 160, 100, 40, ICON, (void *)&resume_b},
    {190, 160, 100, 40, ICON, (void *)&stop_b},
    {190, 160, 100, 40, ICON, (void *)&reset_b},
    {61, 114, 38, 38, ICON, (void *)&alarm_exp_i},
    {221, 114, 38, 38, ICON, (void *)&timer_exp_i},
    {29, 70, 40, 40, TEXT, (void *)&ttxt},
    {99, 70, 40, 40, TEXT, (void *)&ttxt},
    {168, 70, 40, 40, TEXT, (void *)&ttxt},
    {243, 70, 40, 40, TEXT, (void *)&ttxt},
    {80, 66, 40, 40, TEXT, (void *)&ttxt},
    {149, 66, 40, 40, TEXT, (void *)&ttxt},
    {225, 68, 20, 40, TEXT, (void *)&ttxt},
    {29, 70, 62, 42, IMAGE, (void *)&hrs_back},
    {99, 70, 62, 42, IMAGE, (void *)&min_back},
    {168, 70, 62, 42, IMAGE, (void *)&sec_back},
    {230, 70, 62, 42, IMAGE, (void *)&tts_back}
}
```

This array contains alle the widgets defined for the application.

#### 4.5.2.8 Icon plus\_b

**Initial value:**

```
= {  
    b_plus, hide_plus, PLUSBPRESS  
}
```

#### 4.5.2.9 Icon reset\_b

**Initial value:**

```
= {  
    b_reset, hide_stop, STOPBPRESS  
}
```

#### 4.5.2.10 Icon resume\_b

**Initial value:**

```
= {  
    b_resume, hide_start, STARTBPRESS  
}
```

#### 4.5.2.11 Image sec\_back

**Initial value:**

```
= {  
    sec_bkg  
}
```

#### 4.5.2.12 Icon set\_b

**Initial value:**

```
= {  
    b_set, hide_start, STARTBPRESS  
}
```

#### 4.5.2.13 Icon start\_b

**Initial value:**

```
= {  
    b_start, hide_start, STARTBPRESS  
}
```

#### 4.5.2.14 Icon stop\_b

##### Initial value:

```
= {  
    b_stop, hide_stop, STOPBPRESS  
}
```

#### 4.5.2.15 Icon swatch\_b

##### Initial value:

```
= {  
    b_swatch_on, b_swatch_off, SWATCHBPRESS  
}
```

#### 4.5.2.16 Icon timer\_b

##### Initial value:

```
= {  
    b_timer_on, b_timer_off, TIMERBPRESS  
}
```

#### 4.5.2.17 Icon timer\_exp\_i

##### Initial value:

```
= {  
    timer_exp_on, timer_exp_off, NOEVENT  
}
```

#### 4.5.2.18 Image tts\_back

##### Initial value:

```
= {  
    tts_bkg  
}
```

#### 4.5.2.19 Text txt

##### Initial value:

```
= {  
    &Font32x48, White  
}
```

#### 4.5.2.20 Icon watch\_b

##### Initial value:

```
= {  
    b_watch_on, b_watch_off, WATCHBPRESS  
}
```



## 4.6 Events

Event mask declaration.

### Macros

- `#define SetEvt(Event) (evts |= Event)`  
*Sets an event in the event mask.*
- `#define ClearEvt(Event) (evts &= !Event)`  
*Resets an event in the event mask.*
- `#define ClearEvents() (evts = 0)`  
*Resets the event mask.*
- `#define IsEvent(Event) ((unsigned char)(evts & Event))`  
*Checks if an event has been set.*

### Typedefs

- `typedef unsigned char Event`
- `typedef unsigned char Events`

#### 4.6.1 Detailed Description

Event mask declaration.

#### 4.6.2 Macro Definition Documentation

##### 4.6.2.1 `#define ClearEvt( Event ) (evts &= !Event)`

Resets an event in the event mask.

##### Parameters

<i>Event</i>	The event to be reset.
--------------	------------------------

##### 4.6.2.2 `#define IsEvent( Event ) ((unsigned char)(evts & Event))`

Checks if an event has been set.

##### Parameters

<i>Event</i>	The event to be checked in the event mask.
--------------	--

#### 4.6.2.3 `#define SetEvt( Event )(evts |= Event)`

Sets an event in the event mask.

##### Parameters

<i>Event</i>	The event to be set.
--------------	----------------------

## Chapter 5

# Data Structure Documentation

### 5.1 Icon Struct Reference

#### Data Fields

- unsigned char \* **iconp**
- unsigned char \* **iconr**
- Event **onpress**

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

- [Widget.h](#)

### 5.2 Image Struct Reference

#### Data Fields

- unsigned char \* **image**

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

- [Widget.h](#)

### 5.3 Text Struct Reference

#### Data Fields

- sFONT \* **font**
- unsigned short int **color**

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

- [Widget.h](#)

## 5.4 Widget Struct Reference

### Data Fields

- unsigned short int **xl**
- unsigned short int **yt**
- unsigned short int **xw**
- unsigned short int **yh**
- WidgetType **wt**
- void \* **ws**

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

- [Widget.h](#)

## Chapter 6

# File Documentation

### 6.1 code.c File Reference

Contains the body of all tasks and the global variables defined.

```
#include "ee.h"
#include "ee_irq.h"
#include <stdio.h>
#include "stm32f4xx_conf.h"
#include "stm32f4_discovery.h"
#include "stm32f4_discovery_lcd.h"
#include "stm32f4xx.h"
#include "STMPE811QTR.h"
#include "mypictures.h"
#include "Widget.h"
#include "Touch.h"
#include "Event.h"
#include "SWatchFSM.h"
#include "lcd_add.h"
#include "fonts.h"
```

#### Functions

- static void [strencode1digit](#) (char \*str, int digit)  
*Converts a one digit integer into a string.*
- static void [strencode2digit](#) (char \*str, int digit)  
*Converts a two digits integer into a string.*
- static void [updateTime](#) (uint8\_T \*oh, uint8\_T \*om, uint8\_T \*os, uint8\_T \*ot, uint8\_T oldmode)  
*Updates the time on the screen.*
- static void [updateScreen](#) (uint8\_T om, uint8\_T m)  
*Updates the screen widgets.*
- [ISR2](#) (systick\_handler)  
*System Tick interrupt handler.*
- [TASK](#) (TaskLCD)  
*LDC task body.*
- [TASK](#) (TaskClock)  
*Clock task body.*
- int [main](#) (void)  
*Main task of the application.*

## Variables

- RT\_MODEL\_SWatchFSM\_T **SWatch\_state**
- char\_T **errorSig** [6]
- PrevZCX\_SWatchFSM\_T **ZCSig**
- DW\_SWatchFSM\_T **DWork**
- boolean\_T **Bwatch**
- boolean\_T **Bswatch**
- boolean\_T **Balarm**
- boolean\_T **Btimer**
- boolean\_T **Bplus**
- boolean\_T **Bminus**
- boolean\_T **Bstart**
- boolean\_T **Bstop**
- uint8\_T **hours** =0
- uint8\_T **minutes** =0
- uint8\_T **seconds** =0
- uint8\_T **tenths** =0
- uint8\_T **mode**
- uint8\_T **alarm\_status**
- uint8\_T **timer\_exp**
- uint8\_T **swatchrun**
- uint8\_T **watchset**

### 6.1.1 Detailed Description

Contains the body of all tasks and the global variables defined.

#### Author

Paolo Sassi

#### Date

21 January 2016

#### Attention

ERIKA Enterprise - a tiny RTOS for small microcontrollers

Copyright (C) 2002-2013 Evidence Srl

This file is part of ERIKA Enterprise.

ERIKA Enterprise is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License version 2 as published by the Free Software Foundation, (with a special exception described below).

Linking this code statically or dynamically with other modules is making a combined work based on this code. Thus, the terms and conditions of the GNU General Public License cover the whole combination.

As a special exception, the copyright holders of this library give you permission to link this code with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this code, you may extend this exception to your version of the code, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

ERIKA Enterprise is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License version 2 for more details.

You should have received a copy of the GNU General Public License version 2 along with ERIKA Enterprise; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

## 6.2 Event.c File Reference

Contains the event mask definition.

```
#include "Event.h"
```

### Variables

- Events **evts**

### 6.2.1 Detailed Description

Contains the event mask definition.

#### Author

Paolo Sassi

#### Date

22 January 2016

## 6.3 Event.h File Reference

Contains the macros used to handle the event masks.

### Macros

- #define **SetEvt**(Event) (evts |= Event)  
*Sets an event in the event mask.*
- #define **ClearEvt**(Event) (evts &= !Event)  
*Resets an event in the event mask.*
- #define **ClearEvents**() (evts = 0)  
*Resets the event mask.*
- #define **IsEvent**(Event) ((unsigned char)(evts & Event))  
*Checks if an event has been set.*

## Typedefs

- typedef unsigned char **Event**
- typedef unsigned char **Events**

## Variables

- Events **evts**

### 6.3.1 Detailed Description

Contains the macros used to handle the event masks.

#### Author

Paolo Sassi

#### Date

22 January 2016

## 6.4 mypictures.c File Reference

This file contains the application pictures in RGB565 format.

## Variables

- const unsigned char **bkg** [153654]
- const unsigned char **b\_watch\_on** [7254]
- const unsigned char **b\_watch\_off** [7254]
- const unsigned char **b\_swatch\_on** [7254]
- const unsigned char **b\_swatch\_off** [7254]
- const unsigned char **b\_alarm\_on** [7254]
- const unsigned char **b\_alarm\_off** [7254]
- const unsigned char **b\_timer\_on** [7254]
- const unsigned char **b\_timer\_off** [7254]
- const unsigned char **b\_plus** [2646]
- const unsigned char **b\_minus** [2574]
- const unsigned char **b\_start** [8054]
- const unsigned char **b\_stop** [8054]
- const unsigned char **b\_set** [8054]
- const unsigned char **b\_reset** [8054]
- const unsigned char **b\_resume** [8054]
- const unsigned char **hide\_start** [8054]
- const unsigned char **hide\_stop** [8054]
- const unsigned char **hide\_plus** [2574]
- const unsigned char **hide\_minus** [2574]
- const unsigned char **hrs\_bkg** [5262]
- const unsigned char **min\_bkg** [5262]
- const unsigned char **sec\_bkg** [5262]
- const unsigned char **tts\_bkg** [5262]
- const unsigned char **alarm\_exp\_on** [2942]
- const unsigned char **alarm\_exp\_off** [2942]
- const unsigned char **timer\_exp\_on** [2942]
- const unsigned char **timer\_exp\_off** [2942]



### 6.4.1 Detailed Description

This file contains the application pictures in RGB565 format.

#### Author

Paolo Sassi

#### Date

22 January 2016

## 6.5 mypictures.h File Reference

Pictures header file.

### Variables

- const unsigned char **bkg** [153654]
- const unsigned char **b\_watch\_on** [7254]
- const unsigned char **b\_watch\_off** [7254]
- const unsigned char **b\_swatch\_on** [7254]
- const unsigned char **b\_swatch\_off** [7254]
- const unsigned char **b\_alarm\_on** [7254]
- const unsigned char **b\_alarm\_off** [7254]
- const unsigned char **b\_timer\_on** [7254]
- const unsigned char **b\_timer\_off** [7254]
- const unsigned char **b\_plus** [2504]
- const unsigned char **b\_minus** [2504]
- const unsigned char **b\_start** [8054]
- const unsigned char **b\_stop** [8054]
- const unsigned char **b\_set** [8054]
- const unsigned char **b\_reset** [8054]
- const unsigned char **b\_resume** [8054]
- const unsigned char **hide\_start** [8054]
- const unsigned char **hide\_stop** [8054]
- const unsigned char **hide\_plus** [2574]
- const unsigned char **hide\_minus** [2574]
- const unsigned char **hrs\_bkg** [5262]
- const unsigned char **min\_bkg** [5262]
- const unsigned char **sec\_bkg** [5262]
- const unsigned char **tts\_bkg** [5262]
- const unsigned char **alarm\_exp\_on** [2866]
- const unsigned char **alarm\_exp\_off** [2866]
- const unsigned char **timer\_exp\_on** [2942]
- const unsigned char **timer\_exp\_off** [2942]

### 6.5.1 Detailed Description

Pictures header file.

#### Author

Paolo Sassi

#### Date

22 January 2016

## 6.6 Widget.c File Reference

Contains the functions to manage the widgets on the screen.

```
#include "Widget.h"
#include "Event.h"
#include "mypictures.h"
#include <stdio.h>
#include "stm32f4_discovery_lcd.h"
```

### Functions

- unsigned char [contains](#) ([Widget](#) \*w, [TPoint](#) \*point)  
*Checks if the touched point is inside a widget.*
- unsigned char [OnTouch](#) (const [Widget](#) ws[], [TPoint](#) \*press)  
*Handles the touch event.*
- void [DrawInit](#) ([Widget](#) ws[])  
*Draws the initial GUI of the application.*
- unsigned char [DrawOn](#) ([Widget](#) \*w)  
*Draws the 'on' image of a widget.*
- unsigned char [DrawOff](#) ([Widget](#) \*w)  
*Draws the 'off' image of a widget.*
- unsigned char [WPrint](#) ([Widget](#) \*w, char \*s)  
*Prints a string on the screen.*

### Variables

- [Icon](#) [watch\\_b](#)
- [Icon](#) [swatch\\_b](#)
- [Icon](#) [alarm\\_b](#)
- [Icon](#) [timer\\_b](#)
- [Icon](#) [plus\\_b](#)
- [Icon](#) [minus\\_b](#)
- [Icon](#) [start\\_b](#)
- [Icon](#) [stop\\_b](#)
- [Icon](#) [set\\_b](#)
- [Icon](#) [reset\\_b](#)

- [Icon](#) `resume_b`
- [Icon](#) `alarm_exp_i`
- [Icon](#) `timer_exp_i`
- [Image](#) `hrs_back`
- [Image](#) `min_back`
- [Image](#) `sec_back`
- [Image](#) `tts_back`
- [Text](#) `txt`
- [Image](#) `backg`
- [Widget](#) `MyWatchScr` [NUMWIDGETS]

*This array contains alle the widgets defined for the application.*

### 6.6.1 Detailed Description

Contains the functions to manage the widgets on the screen.

#### Author

Paolo Sassi

#### Date

22 January 2016

## 6.7 Widget.h File Reference

Contains the type definitions and the macros used for the screen widgets.

```
#include "Event.h"
#include "Touch.h"
#include "fonts.h"
```

### Data Structures

- struct [Image](#)
- struct [Icon](#)
- struct [Text](#)
- struct [Widget](#)

## Macros

- `#define NUMWIDGETS 25`
- `#define BAKCG 0`
- `#define BWATCH 1`
- `#define BSWATCH 2`
- `#define BALARM 3`
- `#define BTIMER 4`
- `#define BPLUS 5`
- `#define BMINUS 6`
- `#define BSTART 7`
- `#define BSET 8`
- `#define BRESUME 9`
- `#define BSTOP 10`
- `#define BRESET 11`
- `#define ALARMEXP 12`
- `#define TIMEREXP 13`
- `#define HRSSTR 14`
- `#define MINSTR 15`
- `#define SECSTR 16`
- `#define TTSSTR 17`
- `#define SEP1STR 18`
- `#define SEP2STR 19`
- `#define TTSSEP 20`
- `#define HRSBKG 21`
- `#define MINBKG 22`
- `#define SECBKG 23`
- `#define TTSBKG 24`
- `#define NOEVENT 0x00`
- `#define WATCHBPRESS 0x01`
- `#define SWATCHBPRESS 0x02`
- `#define ALARMBPRESS 0x04`
- `#define TIMERBPRESS 0x08`
- `#define PLUSBPRESS 0x10`
- `#define MINUSBPRESS 0x20`
- `#define STARTBPRESS 0x40`
- `#define STOPBPRESS 0x80`
- `#define WATCHMODE 0`
- `#define SWATCHMODE 1`
- `#define ALARMMODE 2`
- `#define TIMERMODE 3`
- `#define txtinfo(w) ((Text *)((w)->ws))`
- `#define iconinfo(w) ((Icon *)((w)->ws))`
- `#define imginfo(w) ((Image *)((w)->ws))`

## Enumerations

- `enum WidgetType { BACKGROUND, ICON, TEXT, IMAGE }`

## Functions

- void `DrawInit` (`Widget` ws[])  
*Draws the initial GUI of the application.*
- unsigned char `OnTouch` (const `Widget` ws[], `TPoint` \*press)  
*Handles the touch event.*
- unsigned char `DrawOn` (`Widget` \*w)  
*Draws the 'on' image of a widget.*
- unsigned char `DrawOff` (`Widget` \*w)  
*Draws the 'off' image of a widget.*
- unsigned char `WPrint` (`Widget` \*w, char \*s)  
*Prints a string on the screen.*

## Variables

- `Widget` `MyWatchScr` []  
*This array contains alle the widgets defined for the application.*

### 6.7.1 Detailed Description

Contains the type definitions and the macros used for the screen widgets.

#### Author

Paolo Sassi

#### Date

22 January 2016



# Index

- alarm\_b
  - Widget Definitions, [15](#)
- alarm\_exp\_i
  - Widget Definitions, [15](#)
- backg
  - Widget Definitions, [15](#)
- ClearEvt
  - Events, [19](#)
- code.c, [23](#)
- contains
  - Widget, [12](#)
- DrawInit
  - Widget, [12](#)
- DrawOff
  - Widget, [13](#)
- DrawOn
  - Widget, [13](#)
- Event.c, [25](#)
- Event.h, [25](#)
- Events, [19](#)
  - ClearEvt, [19](#)
  - IsEvent, [19](#)
  - SetEvt, [19](#)
- hrs\_back
  - Widget Definitions, [16](#)
- Icon, [21](#)
- Image, [21](#)
- Interrupt Handler, [9](#)
- IsEvent
  - Events, [19](#)
- main
  - Tasks, [10](#)
- min\_back
  - Widget Definitions, [16](#)
- minus\_b
  - Widget Definitions, [16](#)
- MyWatchScr
  - Widget Definitions, [16](#)
- mypictures.c, [26](#)
- mypictures.h, [27](#)
- OnTouch
  - Widget, [13](#)
- plus\_b
  - Widget Definitions, [16](#)
- reset\_b
  - Widget Definitions, [17](#)
- resume\_b
  - Widget Definitions, [17](#)
- sec\_back
  - Widget Definitions, [17](#)
- set\_b
  - Widget Definitions, [17](#)
- SetEvt
  - Events, [19](#)
- start\_b
  - Widget Definitions, [17](#)
- stop\_b
  - Widget Definitions, [17](#)
- strencode1digit
  - Utility, [7](#)
- strencode2digit
  - Utility, [7](#)
- swatch\_b
  - Widget Definitions, [18](#)
- TASK
  - Tasks, [10](#)
- Tasks, [10](#)
  - main, [10](#)
  - TASK, [10](#)
- Text, [21](#)
- timer\_b
  - Widget Definitions, [18](#)
- timer\_exp\_i
  - Widget Definitions, [18](#)
- tts\_back
  - Widget Definitions, [18](#)
- txt
  - Widget Definitions, [18](#)
- updateScreen
  - Utility, [8](#)
- updateTime
  - Utility, [8](#)
- Utility, [7](#)
  - strencode1digit, [7](#)
  - strencode2digit, [7](#)
  - updateScreen, [8](#)
  - updateTime, [8](#)
- WPrint
  - Widget, [14](#)

- watch\_b
  - Widget Definitions, [18](#)
- Widget, [11](#), [22](#)
  - contains, [12](#)
  - DrawInit, [12](#)
  - DrawOff, [13](#)
  - DrawOn, [13](#)
  - OnTouch, [13](#)
  - WPrint, [14](#)
- Widget Definitions, [15](#)
  - alarm\_b, [15](#)
  - alarm\_exp\_i, [15](#)
  - backg, [15](#)
  - hrs\_back, [16](#)
  - min\_back, [16](#)
  - minus\_b, [16](#)
  - MyWatchScr, [16](#)
  - plus\_b, [16](#)
  - reset\_b, [17](#)
  - resume\_b, [17](#)
  - sec\_back, [17](#)
  - set\_b, [17](#)
  - start\_b, [17](#)
  - stop\_b, [17](#)
  - swatch\_b, [18](#)
  - timer\_b, [18](#)
  - timer\_exp\_i, [18](#)
  - tts\_back, [18](#)
  - txt, [18](#)
  - watch\_b, [18](#)
- Widget.c, [28](#)
- Widget.h, [29](#)