UM0588
User manual

Multilayer C library for LED dimming used on systems with SPI and DMA capabilities

1 Introduction

The purpose of this user manual is to describe how to use the C library dedicated to perform LED dimming. The document describes all the library functions and their link with the standard peripheral library of a given microcontroller.

This C library can be used without any change on any STM32F10xxx microcontroller.

The library contains a demonstration firmware running on STMicroelectronics evaluation kit STEVAL-ILL015V1.

Contents

List of tables

2 Document and library rules

2.1 Predefined values and structures

2.1.1 Value: ERROR and SUCCESS

```
typedef enum
{
ERROR = 0,
SUCCESS = !ERROR
} ErrorStatus;
```

These values return the result of the functions. The value is ERROR if the function failed and SUCCESS otherwise.

2.1.2 Structure: buttonsAndADC

```
struct buttonsAndADC
{
  unsigned char buttonLeft;
  unsigned char buttonRight;
  unsigned char buttonCenter;
  unsigned char (*buttonKnob)(void);
};
```

This global structure is defined in board_control.c, and declared in board_control.h. It provides the status of the STEVAL-ILL015V1 buttons and knob.

Variables linked to the buttons are set to ‘1’ when the button is pressed. It remains ‘1’ until the user resets it.

Calling buttonKnob() returns the knob current value (variable resistor connected to the ADC).

The following commands must be executed to be able to use this structure. The functions are declared in board_control.h (BC prefix stands for Board Control):

```
BC_GPIO_Configuration();
BC_EXTI_Configuration();
BC_NVIC_Configuration();
BC_ADC_Configuration();
```

2.1.3 Structure environment

```
struct environment
{
  unsigned char columns;            //size of display
  unsigned char rows;               //size of display
  unsigned char colors;             //1 - one color, 3 RGB leds.
  unsigned char randomNumber;       //some games need random number
  unsigned char gameFinished;       //when game ends
  unsigned char setupGame;          //set flag when started first.
The flag is cleared automatically
  unsigned char refreshRequest;     //game changed data to display
};
```

This structure is declared in led_demonstration.h. It is used to pass large amounts of parameters to the demonstration functions. The members of this structure define the logical arrangement of the LED display and help to drive demonstration algorithms.

3 Description of the firmware library

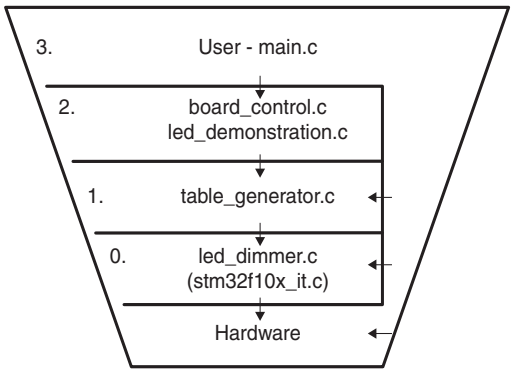
3.1 Summary of firmware library files

Table 1. Firmware library files

L[1]	Filename	Description
0	led_dimmer.c (.h)	Basic dimmer functionality, configuration and commands- hardware related
0	stm32f10x_it.c	Interrupt routines dedicated to service dimming functionality - hardware related
1	table_generator.c (h)	Important tool for generation of the data to be send by interrupt routines
2	board_control.c (h)	Extension enabling simple usage of buttons and ADC on evaluation kit - hardware related
2	led_demonstration.c (.h)	Extension showing functionalities, demonstrations and modes of the evaluation kit
3	main.c	Example application using led_demonstration.c and board_control.c

1. The L column groups the library files by level, from user to hardware (*Figure 1*).

Figure 1. Firmware library files



All the source files use libraries dedicated to the microcontroller peripherals. They are listed in *Table 2*.

Table 2. External libraries and functions

L[1]	Filename	Description
0	stm32f10x_gpio.h	library used to configure the general purpose i/o of the STM32F10xxx
0	stm32f10x_systick.h	library used to configure system timer of STM32F10xxx
0	stm32f10x_dma.h	library used to configure DMA of STM32F10xxx
0	stm32f10x_nvic.h	library used to configure the interrupt controller of STM32F10xxx
0	stm32f10x_lib.h	library containing the definitions, addresses of the STM32F10xxx
1	stdlib.h	provides the fuction "malloc" for dynamic memory allocation
2	stm32f10x_exti.h	library used to configure the external interrupts that are connected to the buttons

1. The L column groups the library files by level, from user to hardware (*Figure 1*).

3.2 LEVEL 0 - led_dimmer.c (.h)

- **File constants and functions**

  PWM_8_bit 256
  PWM_9_bit 512
  PWM_10_bit 1024
  PWM_11_bit 2048
  PWM_12_bit 4096
  RCC_Configuration
  NVIC_Configuration
  SPI_DMA_Initsetup
  start_dimmer
  stop_dimmer
  LED_driver_OutEnable
  LED_driver_OutDisable
  LED_driver_OutEnable_All_On
  LED_driver_OutDisable_All_Off
  error_detection_DM
  error_detection_LE_OE
  SPI_CLK_LOW
  SPI_CLK_HIGH
  use_new_table_immediately
  use_new_table_carefully
  new_table_not_changed_yet
  generateReducedLookUpTable
  generateFullLookUpTable
  removeTableLookUpTable
  removeLookUpTableInUse
  removeLookUpTableReleased

- **External functions and types called from the file**

  free(), SPI_SendData(), SPI_ReceiveData(), SPI_GetFlagStatus() and group of functions declared in libraries dedicated to NVIC, SysTick, GPIO, RCC, DMA.

- **Detailed function description**: see *Chapter 3.2.1* to *3.2.17*.

3.2.1 Functions: RCC_Configuration, NVIC_Configuration, SPI_DMA_Init

Table 3. RCC_Configuration, NVIC_Configuration, SPI_DMA_Init functions

Filename	Description
Function name	RCC_Configuration, NVIC_Configuration, SPI_DMA_Init functions
Function prototype	void RCC_Configuration(), void NVIC_Configuration(), void SPI_DMA_Init()
Behavior description	These three functions configure the system timer, interrupt system, and DMA
Preconditions	—
Called functions	STM32F10xxx library functions

Example

```
RCC_Configuration();
NVIC_Configuration();
SPI_DMA_Init();
```

3.2.2 Function: setup

Table 4. Setup function

Filename	Description
Function name	setup
Function prototype	int setup (int LEDcount, int PWMdepth, int requestedRefreshRate)
Behavior description	configures timers and internal variables for requested dimming parameters
Input parameter1	LEDcount: indicates, how many single LEDs are connected to the system
Input parameter2	PWMdepth: one of those: PWM_8_bit, PWM_9_bit, PWM_10_bit, PWM_11_bit
Input parameter3	requestedRefreshRate: requested refresh rate
Output parameter	returns SUCCESS if requested parameters are achievable
Preconditions	—
Called functions	STM32F10xxx library functions for Systimer, DMA, NVIC

Example

```
/* We have 8 RGB LEDs connected, need 8bit PWM depth, refresh
100Hz*/
setup(3*8, PWM_8_bit, 100);
```

3.2.3 Function: start_dimmer

Table 5. Start_dimmer function

Filename	Description
Function name	start_dimmer
Function prototype	void start_dimmer(void)
Behavior description	Starts dimming. The function LED_drive_OutEnable must then be called in order to make driver outputs active, to make dimming visible on LEDs.
Preconditions	The following functions have to be called before running start_dimmer: RCC_Configuration, NVIC_Configuration, SPI_DMA_Init, setup, generateReducedLookUpTable, use_new_table_immediately
Called functions	SysTick_CounterCmd(SysTick_Counter_Enable)

Example

```
  RCC_Configuration();
  NVIC_Configuration();
  SPI_DMA_Init();

  setup(32, PWM_8_bit, 100);

  if(!generateReducedLookUpTable(LEDmap, 32, PWM_8_bit, &tablePtr,
&tableKeyPtr, &tableSize))
    while(1);
  use_new_table_immediately(tablePtr, tableKeyPtr, tableSize);

  LED_driver_OutEnable(1);
  start_dimmer();
```

3.2.4 Function: stop_dimmer

Table 6. Stop_dimmer function

Filename	Description
Function name	stop_dimmer
Function prototype	void stop_dimmer(void)
Behavior description	stops dimming
Preconditions	—
Called functions	SysTick_CounterCmd(SysTick_Counter_Disable)

3.2.5 Function: LED_driver_OutEnable

Table 7. LED_driver_OutEnable function

Filename	Description
Function name	LED_driver_OutEnable
Function prototype	void LED_driver_OutEnable(unsigned char which, unsigned char RGBchannel)
Behavior description	Makes driver outputs active, makes dimming visible on LEDs.
Input parameter1	which: selects which driver output in a row will be enabled. Its value can be 1 or 2. RGBchannel: selects the color of the LEDs to be enabled.
Preconditions	SPI_DMA_Init
Called functions	STM32F10xxx library functions for GPIO

Example

```
/* We would like to enable RED LEDs of the second driver out of two
only.*/
LED_driver_OutEnable(2, REDchannel);
```

3.2.6 Function: LED_driver_OutDisable

Table 8. LED_driver_OutDisable function

Filename	Description
Function name	LED_driver_OutDisable
Function prototype	void LED_driver_OutDisable(unsigned char which, unsigned char RGBchannel)
Behavior description	switches off the LED panel, so that data loaded and latched into the LED driver are no more visible
Input parameter1	which: selects which driver output in a row will be enabled. Its value can be 1 or 2. RGBchannel: selects the color of the LEDs to be disabled.
Preconditions	SPI_DMA_Init
Called functions	STM32F10xxx library functions for GPIO

Example

```
/* We would like to disable GREEN LEDs of the first driver out of
two only.*/
LED_driver_OutDisable(1, GREENchannel);
```

3.2.7 Function: LED_driver_OutEnable_All_On

Table 9. LED_driver_OutEnable function

Filename	Description
Function name	LED_driver_OutEnable_All_On
Function prototype	void LED_driver_OutEnable_All_On (void)
Behavior description	Switches on the LEDS according to data loaded and latched into the all LED drivers
Input parameter1	—
Preconditions	SPI_DMA_Init
Called functions	STM32F10xxx library functions for GPIO.

Example

```
/* We would like to enable all the drivers*/
       LED_driver_OutEnable_All_On();
```

3.2.8 Function: LED_driver_OutDisable_All_Off

Table 10. LED_driver_OutDisable function

Filename	Description
Function name	LED_driver_OutDisable_All_Off
Function prototype	void LED_driver_OutDisable_All_Off (void))
Behavior description	switches off the LED panel so that data loaded and latched into all LED drivers are no more visible
Input parameter1	—
Preconditions	SPI_DMA_Init
Called functions	STM32F10xxx library functions for GPIO

Example

```
/* We would like to disable all two drivers*/
       LED_driver_OutDisable_All_Off();
```

3.2.9 Function: error_detection_DM

Table 11. Error_detection_DM function

Filename	Description
Function name	error_detection_DM
Function prototype	unsigned char error_detection_DM(unsigned char *LEDmapIN, int LEDcount)
Behavior description	detects defective LEDs. The defective LED position is indicated by zeroes in LEDmapIN. returns ‘1’ if an error occurred uses DM signal to enter the test mode
Input parameter1	*LEDmapIN: array of bytes, the length of the array is LEDcount
Input parameter2	LEDcount: number of LEDs connected to the system
Output parameter	It returns ‘1’ if an error occurred. The defective LED position is indicated by zeroes in LEDmapIN. It uses DM signal to enter the test mode.
Preconditions	SPI_DMA_Init
Called functions	LED_driver_OutDisable_All_Off(), GPIO, SPI, LED_driver_OutEnable_All_On()

Example

```
unsigned char LEDmapClear[48];
error_detection_DM(LEDmapClear, 48);
```

3.2.10 Function: error_detection_LE_OE

Table 12. Error_detection_LE_OE function

Filename	Description
Function name	error_detection_LE_OE
Function prototype	unsigned char error_detection_LE_OE(unsigned char *LEDmapIN, int LEDcount)
Behavior description	detects defective LEDs
Input parameter1	*LEDmapIN: array of bytes, the length of the array is LEDcount
Input parameter2	LEDcount: number of LEDs connected to the system
Output parameter	It returns ‘1’ if an error occurred. The defective LED position is indicated by zeroes in LEDmapIN. It uses LE and OE signals to enter the test mode
Preconditions	SPI_DMA_Init
Called functions	called functionsLED_driver_OutDisable_All_Off(), GPIO functions, SPI functions, LED_driver_OutEnable_All_On(), SPI_CLK_HIGH(), SPI_CLK_LOW()

Example

```
unsigned char LEDmapClear[48];
error_detection_LE_OE(LEDmapClear, 48);
```

3.2.11 Function: SPI_CLK_LOW, SPI_CLK_HIGH

Table 13. SPI_CLK_LOW, SPI_CLK_HIGH function

Filename	Description
Function name	SPI_CLK_LOW, SPI_CLK_HIGH
Function prototype	void SPI_CLK_LOW(void); void SPI_CLK_HIGH(void)
Behavior description	Changes the SPI clock output to the desired logical and electrical level
Preconditions	the SPI pins have to be reconfigured to the GPIO Output Push Pull mode
Called functions	GPIO functions

Example

```
/* Simple generation of CLK pulse edge and intra data change*/
  GPIO_SetBits(GPIOA, GPIO_Pin_4);
  SPI_CLK_HIGH();
  SPI_CLK_LOW();
  GPIO_ResetBits(GPIOA, GPIO_Pin_4);
  SPI_CLK_HIGH();
  SPI_CLK_LOW();
```

3.2.12 Function: use_new_table_immediately

Table 14. Use_new_table_immediately function

Filename	Description
Function name	use_new_table_immediately
Function prototype	void use_new_table_immediately(unsigned char *table, unsigned char *tableKey, int table_size)
Behavior description	It exchanges table (SPI data) without preventing glitch
Input parameter1	*table: table with SPI data generated by generateReducedLookUpTable
Input parameter2	*tableKey: indexes for the table generated by generateReducedLookUpTable
Input parameter3	table_size: size of the table returned by generateReducedLookUpTable
Preconditions	generateReducedLookUpTable
Called functions	none

Example

```
/* Mainly used for the first generation of the table or similar*/
int tableSize;
  unsigned char *tablePtr, *tableKeyPtr;

  setup(48, PWM_8_bit, 100);
  if(!generateReducedLookUpTable(LEDmapClear, 48, PWM_8_bit,
  &tablePtr, &tableKeyPtr, &tableSize))
    while(1);
  use_new_table_immediately(tablePtr, tableKeyPtr, tableSize);
```

3.2.13 Function: use_new_table_carefully

Table 15. Use_new_table_carefully function

Filename	Description
Function name	use_new_table_carefully
Function prototype	void use_new_table_carefully(unsigned char *table, unsigned char *tableKey, int table_size)
Behavior description	It exchange table (SPI data) at appropriate time to prevent glitch. It takes a new table and replaces the pointers as soon as it is the right time
Input parameter1	*table: table with SPI data generated by generateReducedLookUpTable
Input parameter2	*tableKey: indexes for the table generated by generateReducedLookUpTable
Input parameter3	table_size: size of the table returned by generateReducedLookUpTable
Preconditions	generateReducedLookUpTable
Called functions	none

Example

```
/* Mainly used during slight change of the table, in main loops.*/
   ...
   while(1){
    tetris_B_and_W(LEDmapT, &buttonsADC, &env, &(userData[0]));
   if(env.refreshRequest){
     if(!generateReducedLookUpTable(LEDmapT, 32, PWM_8_bit,
   &tablePtr, &tableKeyPtr, &tableSize))
       while(1);
     use_new_table_carefully(tablePtr, tableKeyPtr, tableSize);
     while(new_table_not_changed_yet());
     removeLookUpTableReleased();
   }
     }
```

3.2.14 Function: new_table_not_changed_yet

Table 16. New_table_not_changed_yet function

Filename	Description
Function name	new_table_not_changed_yet
Function prototype	int new_table_not_changed_yet(void)
Behavior description	it returns TRUE untill appropriate time for table exchange is reached
Output parameter	remains TRUE (‘1’) untill the new table pointers are used and exchanged with the old one
Preconditions	use_new_table_carefully
Called functions	none

Example

```
use_new_table_carefully(tablePtr, tableKeyPtr, tableSize);
while(new_table_not_changed_yet());
removeLookUpTableReleased();
```

3.2.15 Function: generateReducedLookUpTable

Table 17. GenerateReducedLookUpTable function

Filename	Description
Function name	generateReducedLookUpTable
Function prototype	int generateReducedLookUpTable(unsigned char *LEDmapIN, int LEDcount, int PWMdepth, unsigned char **tableIN, unsigned char **tableKeyIN, int *table_size)
Behavior description	generate tables. Allocates memory and updates table_size value. Returns ‘1’ if successful.
Input parameter1	*LEDmapIN: array of bytes, the array length is LEDcount. User defines here the brightness of every LED.
Input parameter2	LEDcount: number of LEDs connected to the system
Input parameter3	PWMdepth: used resolution of the PWM
Input parameter4	**tableIN: the function creates a table containing SPI data and updates this pointer
Input parameter5	**tableKeyIN: the function creates indexes for the table and updates this pointer
Input parameter6	*table_size: the function returns in this variable the size of the created table
Input parameter7	requestedRefreshRate: requested refresh dimming rate
Output parameter	return ‘1’, if allocation and generation completed successfully
Preconditions	enough of free RAM
Called functions	performGenerateReducedLookUpTable()

Example

```
  unsigned char LEDmapT[32] = {0x00, 0x00, 0x00, 0x00,
                               0x00, 0x00, 0x00, 0x00,
                               0x00, 0x00, 0x00, 0x00,
                               0x00, 0x00, 0x20, 0x00,
                               0x00, 0x00, 0x20, 0x00,
                               0xe0, 0x00, 0x00, 0xf0,
                               0xc0, 0x00, 0x00, 0x60,
                               0x60, 0x00, 0x00, 0x50};
       if(!generateReducedLookUpTable(LEDmapT, 32, PWM_8_bit,
&tablePtr, &tableKeyPtr, &tableSize))
  while(1); //an endless loop if the table generation fails
       //now use use_new_table function to use generated data
```

3.2.16 Function: removeLookUpTableInUse

Table 18. RemoveLookUpTableInUse function

Filename	Description
Function name	removeLookUpTableInUse
Function prototype	void removeLookUpTableInUse()
Behavior description	releases tableIN, and tableKeyIN created in the system by use_new_table_X
Preconditions	stop_dimmer, new_table_not_changed_yet, use_new_table_X
Called functions	free()

```
/* It is used to clean up the memory after the application
finishes.*/
```

3.2.17 Function: removeLookUpTableReleased

Table 19. RemoveLookUpTableReleased function

Filename	Description
Function name	removeLookUpTableReleased
Function prototype	void removeLookUpTableReleased()
Behavior description	releases tableIN, and tableKeyIN created in the system by use_new_table_X
Preconditions	GenerateReducedLookUpTable followed by one of these two: use_new_table_immediately or use_new_table_carefully. Enough free RAM space.
Called functions	free()

Example

```
if(!generateReducedLookUpTable(LEDmapT, 32, PWM_8_bit,
&tablePtr,  &tableKeyPtr, &tableSize))
while(1);
use_new_table_carefully(tablePtr, tableKeyPtr, tableSize);
while(new_table_not_changed_yet());
removeLookUpTableReleased();
```

3.3 LEVEL 0 - stm32f10x_it.c

- **File functions**

  This project uses the following interrupt handlers:

  SysTickHandler

  DMAChannel3_IRQHandler

  SPI1_IRQHandler

  EXTI15_10_IRQHandler

  USB_LP_CAN_RX0_IRQHandler

- **Detailed function description**

  All these functions are called by the microcontroller when an interrupt occurs. They cannot be called directly by the user. These interrupt handlers can be found in STM32f10x_it.c.

3.4 LEVEL 1 - table_generator.c (h)

- **File constants and functions**

  performGenerateReducedLookUpTable

- **External functions and types called from this file:**

  malloc();

- **Detailed function description:** see *Chapter 3.4.1*.

3.4.1 Function: performGenerateReducedLookUpTable

Table 20. PerformGenerateReducedLookUpTable function

Filename	Description
Function name	performGenerateReducedLookUpTable
Function prototype	int performGenerateReducedLookUpTable(unsigned char *LEDmapIN, int LEDcount, int PWMdepth, unsigned char **tableIN, unsigned char **tableKeyIN, int *table_size)
Behavior description	Generates the tables for the function generateReducedLookUpTable. It should not be called directly by user (contradicts section 3.3 “they cannot be called’).
Preconditions	—
Called functions	malloc(), sizeof()

Example

```
/*This function is called by function generateReducedLookUpTable
from the led_dimmer.c file. The function is not intended to be
called directly by user*/
```

3.5 LEVEL 2 - board_control.c (h)

- **File constants and functions**
  void BC_ADC_Configuration(void);
  void BC_GPIO_Configuration(void);
  void BC_EXTI_Configuration(void);
  void BC_NVIC_Configuration(void);
  struct buttonsAndADC;
- **External functions and types called from this file**
  ADC, GPIO, EXTI, NVIC functions.
- **Detailed function description:** see *Chapter 3.5.1*.

3.5.1 Functions: BC_ADC_Configuration, BC_GPIO_Configuration, BC_EXTI_Configuration, BC_NVIC_Configuration

Table 21. Setup function

Filename	Description
Function name	BC_ADC_Configuration, BC_GPIO_Configuration, BC_EXTI_Configuration, BC_NVIC_Configuration
Function prototype	void BC_ADC_Configuration(void), void BC_GPIO_Configuration(void), void BC_EXTI_Configuration(void), void BC_NVIC_Configuration(void)
Behavior description	This four functions configure the ADC for knob, the GPIO for buttons, the external interrupts for buttons, and the interrupt controller for buttons. The whole configuration is given for the STEVAL-ILL015V1 evaluation kit.
Preconditions	—
Called functions	see source code

Example

```
/* BC_ADC_Configuration must be called before using structure:
unsigned char buttonsAndADC.(*buttonKnob)(void);*/
```

3.6 LEVEL 2 - led_demonstration.c (.h)

- **File constants and functions**
  modeSelect
  tetris_color
  solid_color_demo
  wave_color_demo
  error_demo
  writeCharacter
- **External functions and types called from this file:**
  Structures used: environment, buttonsAndADC.
  No hardware related functions are called from this file.
- **Detailed function description:** see *Chapter 3.6.1* to *3.6.3*.

3.6.1 Function: modeSelect

Table 22. ModeSelect function

Filename	Description
Function name	modeSelect
Function prototype	void modeSelect(unsigned char *LEDmap, struct buttonsAndADC *btnADC, struct environment *env, signed int *userData)
Behavior description	Provides a user interface. Using buttons, the user can choose one letter among many in order to select the preferred application.
Input parameter1	*LEDmap: input and output brightness display map
Input parameter2	*btnADC: global variable buttonsADC
Input parameter3	*env: struct environment (described at the beginning of this document)
Input parameter4	*userData: array of signed integers, length of 11
Output parameter	userData[0] returns index relevant to the chosen letter. userData[0] returns ‘100’ if no letter has been chosen yet.
Preconditions	structStruct environment, *env: must be filled with correct data. BC_ADC_Configuration must, BC_GPIO_Configuration, BC_EXTI_Configuration, BC_NVIC_Configuration should be called before using of this function.
Called functions	buttonsAndADC->buttonKnob(), writeCharacter()

Example

```
/* See the complete code in the chapter: LEVEL 3 - main.c */
```

3.6.2 Function: writeCharacter

Table 23. WriteCharacter function

Filename	Description
Function name	writeCharacter
Function prototype	void writeCharacter(unsigned char *LEDmap, unsigned char *mask, unsigned short brightnessR, unsigned short brightnessG, unsigned short brightnessB, unsigned char Rotate)
Behavior description	It sets bytes in the *LEDmap array according to the requested character to be displayed.
Input parameter1	*LEDmap: an output brightness display map
Input parameter2	*mask: table with the character shape
Input parameter3,4,5	brightnessR, brightnessG, brightnessB: defines the color of the generated character
Input parameter4	rotate: ‘0’: no rotation ‘1’: rotation: change character orientation upside down
Preconditions	—
Called functions	—

Example

```
/* .. .. ..
writeCharacter(LEDmap, (unsigned char *) number_mask[n100], 0,
0x50, 0, ROTATE);
.. .. ..*/
```

3.6.3 Functions: tetris_color, solid_color_demo, wave_color_demo, error_demo

Table 24. Tetris_color, solid_color_demo, wave_color_demo, error_demo functions

Filename	Description
Function name	tetris_color, solid_color_demo, wave_color_demo, error_demo
Function prototype	void tetris_color(unsigned char *LEDmap, struct buttonsAndADC *btnADC, struct environment *env, signed int *userData) void solid_color_demo(unsigned char *LEDmap, struct buttonsAndADC *btnADC, struct environment *env, signed int *userData) void wave_color_demo(unsigned char *LEDmap, struct buttonsAndADC *btnADC, struct environment *env, signed int *userData) void error_demo(unsigned char *LEDmap, struct buttonsAndADC *btnADC, struct environment *env, signed int *userData)
Behavior description	Shows different types of LED demonstrations. These functions never ask for termination.
Input parameter1	*LEDmap: input and output brightness display map
Input parameter2	*btnADC: global variable buttonsADC
Input parameter3	*env: struct environment (described at the beginning of this document)
Input parameter4	*userData: array of signed integers, length of 11.
Preconditions	struct environment, *env: must be filled with correct data. BC_ADC_Configuration must, BC_GPIO_Configuration, BC_EXTI_Configuration, BC_NVIC_Configuration should be called before using of this function.
Called functions	buttonsAndADC->buttonKnob(), writeCharacter()

Example

```
/* See the complete code in the chapter: LEVEL 3 - main.c */
```

3.7 LEVEL 3 - main.c

- **File functions**

  waitAbit(), main().

- **Functions called from LEVEL 2, 1, 0**

  All functions from level 2, a few from level 1, and occasionally from level 0, are called from this unit. It is recommended to use the highest level functions possible while solving any task.

- **External functions that are called from this file**

  These functions are not directly related to the hardware. It can be any function from level 0 to level 2.

- **Detailed function description**

```
int main(void){
 RCC_Configuration();
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
 RCC_USBCLKConfig(RCC_USBCLKSource_PLLCLK_Div1);
 RCC_APB1PeriphClockCmd(RCC_APB1Periph_USB, ENABLE);
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);
 NVIC_Configuration();
 SPI_DMA_Init();

 setup(48, PWM_8_bit, 100);

 if(!generateReducedLookUpTable(LEDmapClear, 48, PWM_8_bit,
&tablePtr, &tableKeyPtr, &tableSize))
    while(1);
 use_new_table_immediately(tablePtr, tableKeyPtr, tableSize);

 start_dimmer();

 BC_GPIO_Configuration();
 BC_EXTI_Configuration();
 BC_NVIC_Configuration();
 BC_ADC_Configuration();

 LED_driver_OutEnable(1);
 LED_driver_OutEnable(2);

 #define modes 4;
 modeSelector = 100;
 userData[1] = modes;
 env.columns = 3;
 env.rows = 5;
 env.setupGame = 1;

 while(1){
    switch(modeSelector){

    case 0: tetris_color(LEDmapClear, &buttonsADC, &env,
&(userData[0]));break;            //A-tetris
    case 1: wave_color_demo(LEDmapClear, &buttonsADC, &env,
&(userData[0]));break; //B-wavedemo
    case 2: solid_color_demo(LEDmapClear, &buttonsADC, &env,
&(userData[0]));break; //C-solid color
    case 3: error_demo(LEDmapClear, &buttonsADC, &env,
&(userData[0]));               /D - error detection
              if(env.gameFinished){
                error_detection_DMX(LEDmapClear, 48);
                start_dimmer();
                LED_driver_OutEnable(1);
                LED_driver_OutEnable(2);
              }
          break;
```

```
    case 100: modeSelect(LEDmapClear, &buttonsADC, &env,
&(userData[0]));
              if(env.gameFinished){
                env.setupGame = 1;
                env.gameFinished = 0;
                modeSelector = userData[0];
                break;
                }
    default: modeSelector = 100;break;
    }

    if(env.refreshRequest){
      if(!generateReducedLookUpTable(LEDmapClear, 48, PWM_8_bit,
&tablePtr, &tableKeyPtr, &tableSize))
        while(1);            //endless loop if memory or generation
fails

      use_new_table_carefully(tablePtr, tableKeyPtr, tableSize);
      while(new_table_not_changed_yet());
      removeLookUpTableReleased();
      env.refreshRequest = 0;
    }
 }
}//main
```

4 Revision history

Table 25. Document revision history

Date	Revision	Changes
17-Sep-2008	1	Initial release.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com