Using the GPIOs and interrupt controller to drive LEDs on STM8 Nucleo-64 boards

Introduction

The NUCLEO-8S208RB (built around the STM8S208RBT6 device) and the NUCLEO-8L152R8 (built around the STM8L152R8T6 device) are boards that allow the evaluation of the main features of all the STM8S Series and STM8L Series microcontrollers.

This application note provides a short description on how to use the GPIOs and the interrupt controllers on the NUCLEO-8S208RB and on the NUCLEO-8L152R8 to drive a set of LEDs.

Once the microcontroller (STM8S208RBT6 or STM8L152R8T6 in this example) has been powered-up through a USB cable connected to the host PC, the LD2 and LD5 LEDs (not on-board LEDs) start blinking. Each time that the "push" button is pressed, the interrupt controller assets an interrupt that is used to control the I/Os, and changes the LED behavior.

Table 1. Applicable products

Type	Part number
Evaluation boards	NUCLEO-8S208RB
	NUCLEO-8L152R8

Reference documents

- *STM8 Nucleo-64 boards* data brief (DB3591)
- *STM8L152R8T6 Nucleo-64 board* user manual (UM2351)
- *STM8S208RBT6 Nucleo-64 board* user manual (UM2364)

1 Application description

This section describes the hardward requirements, the application's schematics and the application's principle to use the GPIOs and the interrupt controller to drive a set of LEDs on the NUCLEO-8S208RB or the NUCLEO-8L152R8 boards.

1.1 Hardware requirements

No on-board resources are required other than the user push-button.

The external components required by the application are listed on the table below.

Table 2. External hardware components required

External components	Value	Comments
LD2, LD3, LD4, LD5	-	Standards LEDs
R2, R3, R4, R5	510 Ω	Protective resistors

1.2 Application schematics

The figure below shows how to interface the LEDs and the push button with the NUCLEO-8S208RB or the NUCLEO-8L152R8 boards. For details on NUCLEO-8S208RB or NUCLEO-8L152R8 boards implementation, refer to the board schematics provided in the corresponding user manual (UM2351 or UM2364).

Protective resistors, R2, R3, R4, and R5, are mandatory to limit the current to a value that does not harm the LEDs. The push button requires a debounce filter (RC) and a pull-up resistor (R1) to avoid triggering several interrupts due to the mechanical bouncing of the button.

Figure 1. STM8S Series application schematic

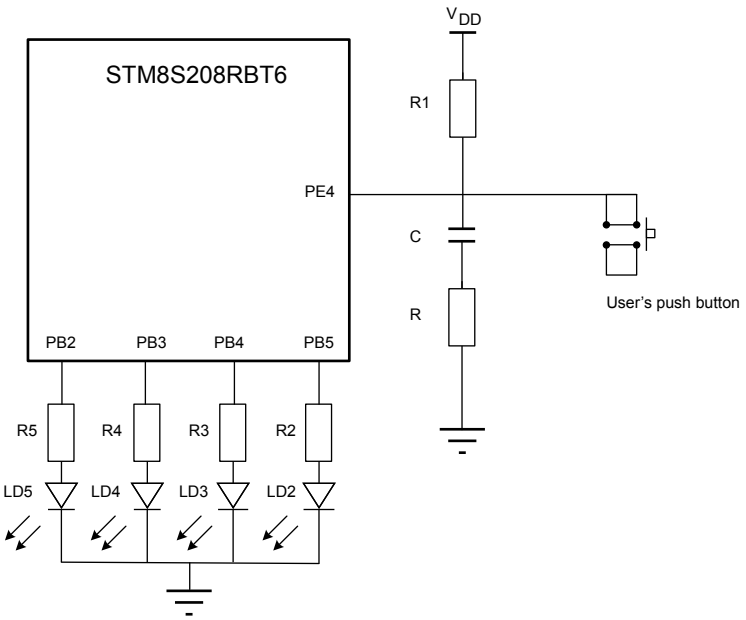
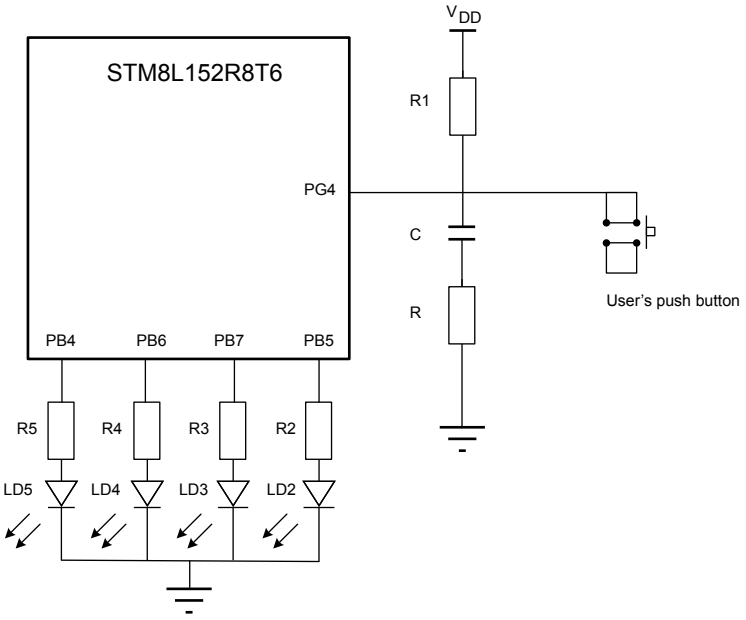


Figure 2. **STM8L Series application schematic**



1.3 Application principle

At startup, LD2 and LD5 start blinking meaning that the Flash memory of the STM8S208RBT6 or STM8L152R8T6 device has been successfully programmed.

Pressing the push button generates an interrupt which is handled by the application software to drive the LEDs.

Only one of the two pairs of LEDs, LD2/LD5 and LD3/LD4 blink at a time. A button event triggers the blinking of the other pair while switching off the first one. The LEDs blinking conditions are described in the table below.

Table 3. **LEDs configuration**

Application	LED state
At startup	LD2 and LD5 blink
On button event	The blinking LED pairs are swapped

2 Software description

The application software uses STM8S Series and STM8L Series standard firmware libraries to control the general purpose features described in the peripheral configuration section of this application note.

2.1 STM8S Series and STM8L Series peripherals configuration

2.1.1 GPIOs configuration

The application drives the MCU I/Os to interface the microcontroller with external hardware components. The GPIO_Init() function configures the button (PE4 for STM8S208RBT6 or PG4 for STM8L152R8T6) as floating input with interrupt to detect push button events, and PB2/PB3/PB4/PB5 for STM8S Series or PB4/PB5/PB6/PB7 for STM8L Series as output push-pull to control the LEDs.

2.1.2 EXTI configuration

The external interrupt controller is configured through the EXTI_SetExtIntSensitivity() function to handle the external interrupts on the push button (PE4 for STM8S208RBT6 or PG4 for STM8L152R8T6).

The external interrupt sensitivity is configured to trigger an interrupt each time a falling edge, and only a falling edge, is detected on the push button (PE4 for STM8S208RBT6 or PG4 for STM8L152R8T6).

2.2 STM8 standard firmware library configuration

2.2.1 STM8S Series standard firmware library

The *stm8s_conf.h* file of the STM8S Series standard firmware library is used to configure the library by enabling the peripheral functions used by the application.

The following define-statements must be present:

```
#define _GPIO 1 /* enables the GPIOs */
#define _EXTI 1 /* enables the EXTI */
```

2.2.2 STM8L Series standard firmware library

The *stm8l15x_conf.h* file of the STM8S Series standard firmware library is used to configure the library by enabling the peripheral functions used by the application.

The following define-statements must be present:

```
#include "stm8l15x_gpio.h"
#include "stm8l15x_exti.h"
```

2.3 Application software flowcharts

This section contains the main loop flowchart and the interrupt function flowchart.

2.3.1 Main loop flowchart

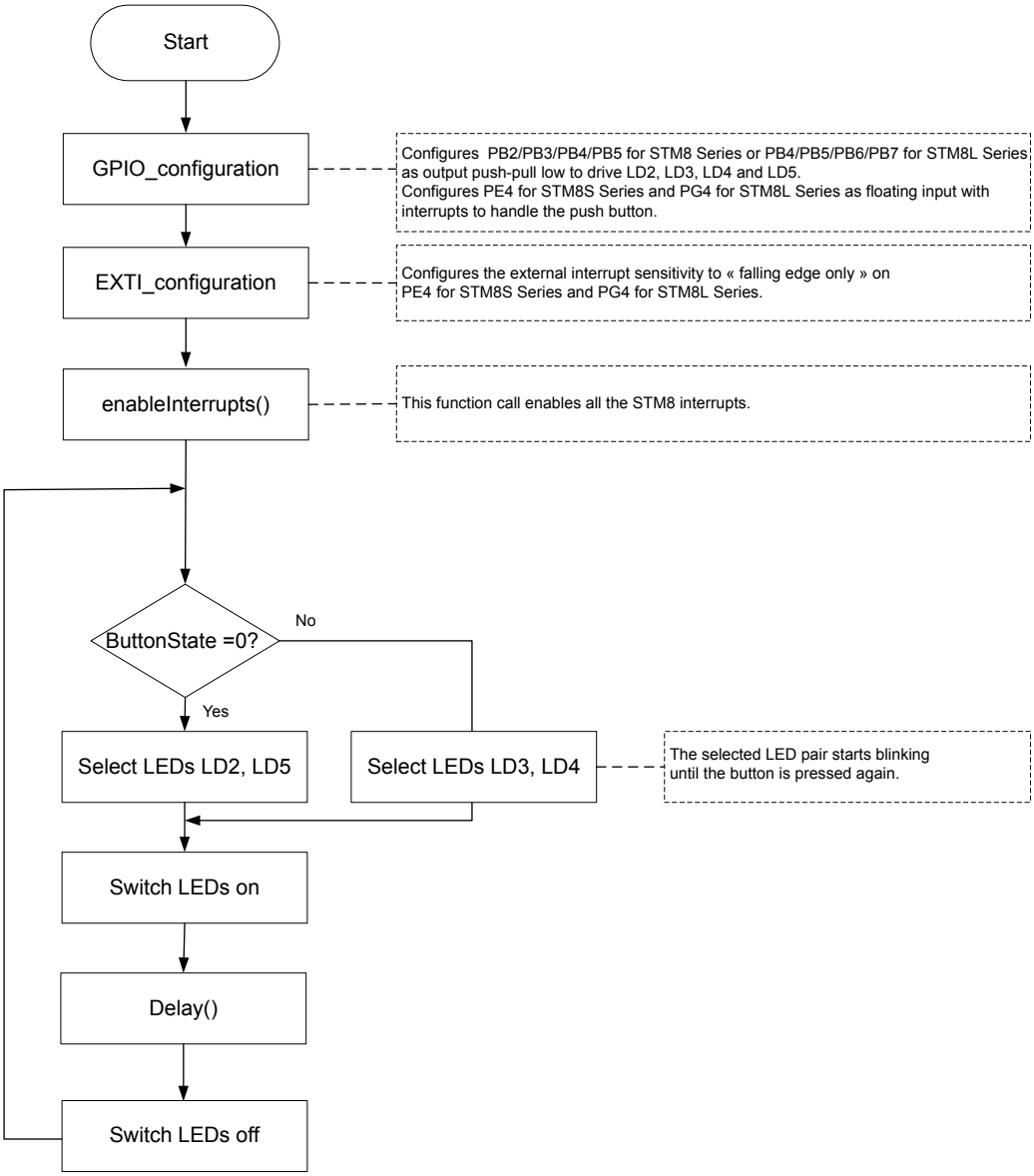
The code main loop implements the algorithm that controls the LEDs according to push button events. The blinking LED pair is selected by setting the *ButtonState* flag.

Each time the push button is pressed, an interrupt is triggered and *ButtonState* is complemented (see Section 2.3.2 Interrupt function flowchart . The main loop code tests *ButtonState* and selects the blinking LED pair according to its value (see Section 1.3 Application principle).

The Delay() function generates a delay between the LED *on* and *off* states so that we can see them blink.

The figure below shows the flowchart of the application software main loop.

Figure 3. Main loop flowchart



2.3.2 Interrupt function flowchart

Each time an interrupt is asserted, a sepcific function complements the *ButtonState* flag and the main loop behaves accordingly (see Section 1.3 Application principle). This function is:

- EXTI_PORTE_IRQhandler() for STM8S Series
- EXTI_PORTG_IRQhandler() for STM8L Series

The figures below shows the flowchart of the EXTI_PORTE_IRQhandler() and the EXTI_PORTG_IRQhandler() interrupt function.

Figure 4. EXTI_PORTE_IRQhandler() function flowchart for STM8S Series

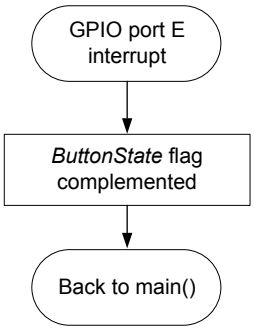
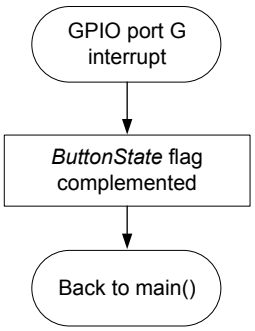


Figure 5. EXTI_PORTG_IRQhandler() function flowchart for STM8L Series



Revision history

Table 4. **Document revision history**

Date	Version	Changes
29-Jun-2018	1	Initial release.

Contents

List of tables

List of figures