Practical 4

Introduction to Microcontrollers - General Purpose Input Output

Practical 4 has a number of sub-tasks, please complete the tasks in order.
Read and revise:
Topic 3 on VULA lessons.
Practical instructions

1. Read the practical instructions.
2. Download and install a suitable ARM toolchain for the STM32F0 series of microcontrolelrs (and IDE) for your PC/Mac. Possible options include but are not limited to:
   (a) **Atollic TrueSTUDIO** (Windows 32-bit/64-bit; Linux 32-bit/64-bit)
   (b) **Eclipse IDE for C/C++ Developers with ARM tool chain extension** (Windows 32-bit/64-bit; Linux 32-bit/64-bit; Mac OS X 32-bit/64-bit (Intel/M processors)).
   (c) Command line based tools such as ARM-GCC, GDB, Openocd, ST-LINK etc.
3. Complete the online pre-practical quiz on VULA
4. Complete your code at home
5. Complete the online post-practical quiz on VULA
6. Attend and demonstrate your practical in the last week of term

You will need the following components for this practical (see Appendix **??***):*

- The STM32F0 Discovery Board which will be used as a regulated +5V power supply.
- A USB Type A to Mini-B Cable

4.1 Introduction

The aim of this practical is to introduce to the general purpose output (GPIO) capability of your STMicroelectronics STM32F0 Microcontroller a. You will also learn how to read the microcontroller reference manuals to identify which registers are needed to control the system's operation.

This practical contains both online practical quizzes (pre- and post-) and physical testing. Please ensure that you submit your quizzes on VULA by the due dates. There will be an in person practical demonstration session in the last week of term where you will be demonstrating the operation of your circuit to a TA or tutor.

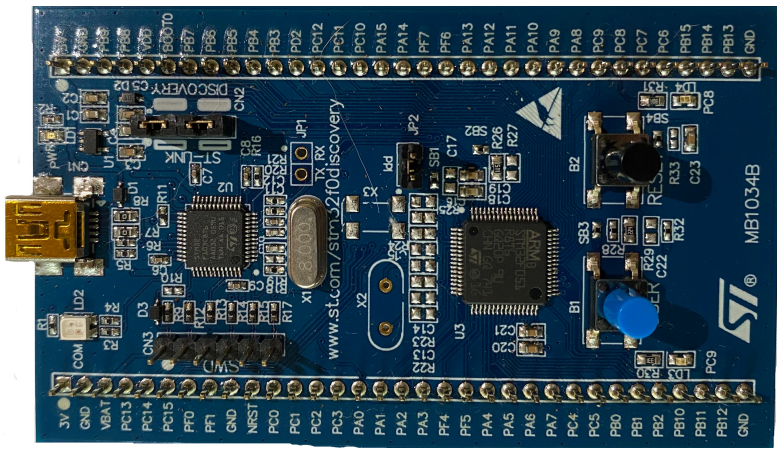


Figure 4.1: The STM32F0 Discovery Board.

4.2 Background: General purpose input output

In this practical you will learn how to configure and use the *General Purpose Input/Output* ports (GPIO) of your STMicroelectronics STM32F051R8T6 microcontroller. Microcontrollers contain a number of *General Purpose Input Output* (GPIO) pins which act as a generic physical interface or connection to external devices. The behaviour and the functionality of these pins can be modified by the programmer in firmware and must be defined before use.

Input and *output* (I/O) signals can be either digital or analogue and this depends on the type of external device which is connected to the pin.

Microcontroller GPIO pins are grouped together in a parallel interface known as a *port*. These ports are normally labelled with a letter such as port A, port B etc. and often contain 8 to 16 pins per port.

All microcontrollers contain GPIO ports, however their physical implementation and functionality varies between manufacturers and families. The 64-pin STM32F051R8T6 microcontroller contains 51 GPIO pins, a number of which are multiplexed with internal peripheral modules built into the microcontroller chip. The pin names and numbers can be seen in the figure 4.2.

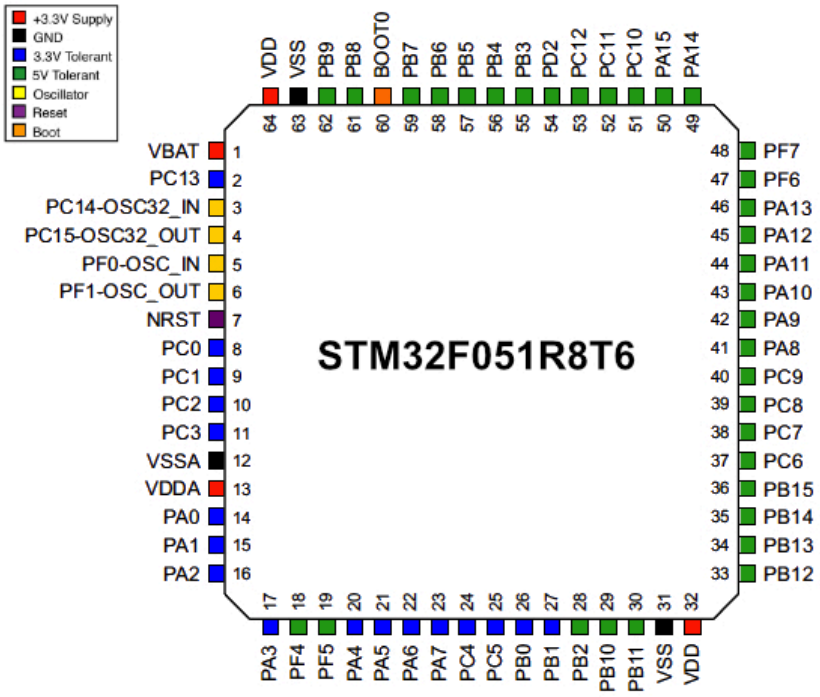


Figure 4.2: The pinouts of the STM32F051R8T6 microcontroller. The colour of the pin represents the pin type and I/O structure.

Each GPIO port's operation is controlled by a number of GPIO peripheral registers as well as a *Reset and Clock Control* peripheral register (**AHBENR**) which define the following:

- The AHB bus clock connections to the GPIO port.
- The register's mode (Digital input, digital output, alternative function such as SPI, I^2C etc. or analogue mode)
- GPIO port's output type (Output enabled as push/pull or enabled as open-drain)
- GPIO port's output switching speed (low, medium or high)
- GPIO port's internal pull up or pull down resistor enable
- Used to read GPIO port input data register
- GPIO port output data register
- GPIO port set or reset the output data register
- Registers to select an assigned alternative function for a port pin

The internal structure of a single pin in a port can be seen in figure 4.3.

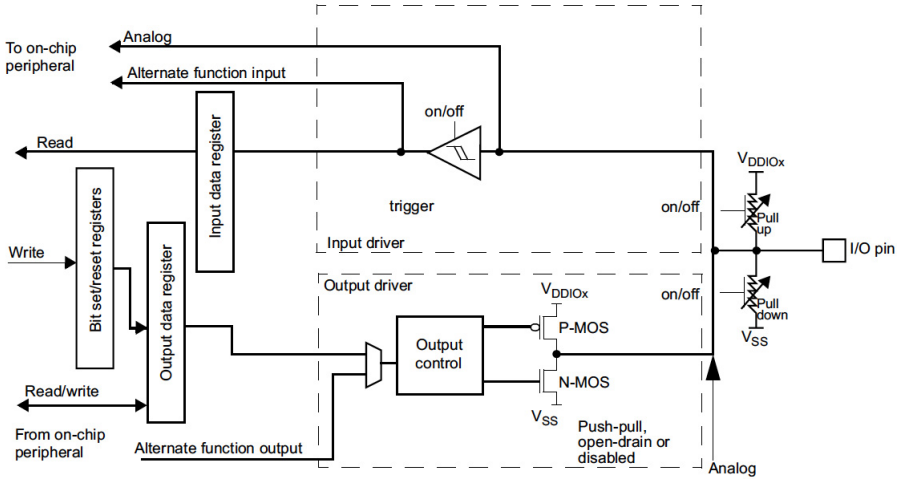


Figure 4.3: The internal structure of a GPIO pin in the STM32F0 microcontroller. (Taken from [4])

The layout of the external digital input output devices on the STM32F0 Discovery board can be seen in figure 4.4.

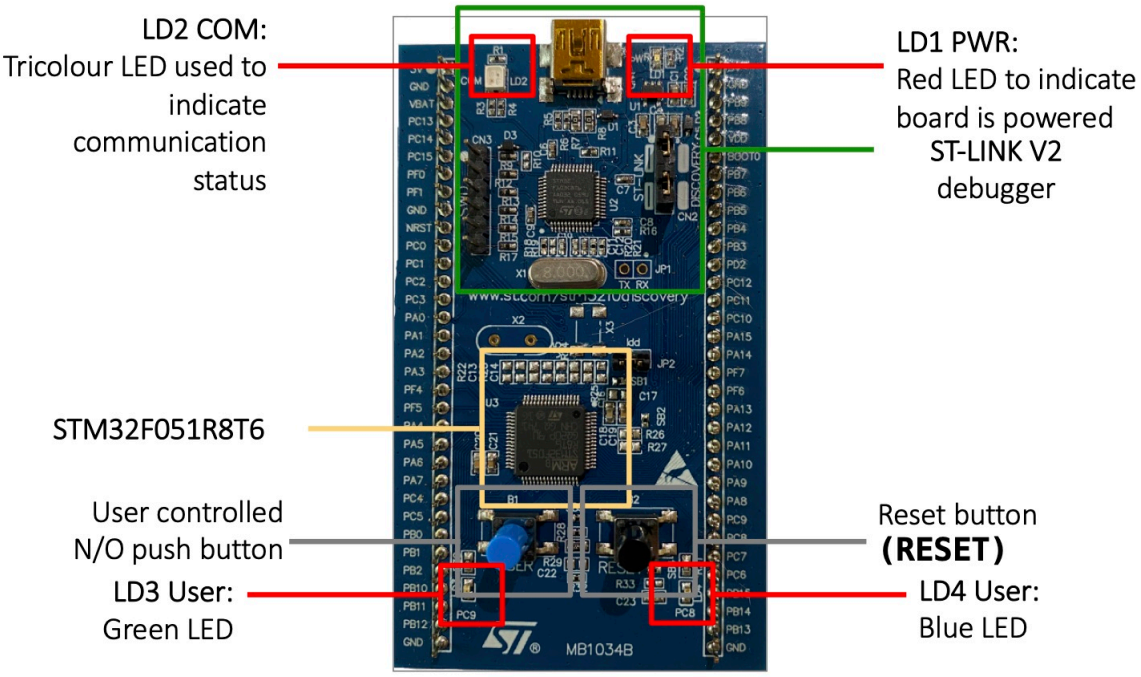


Figure 4.4: The simple input/output devices on the STM32F0 Discovery Board The basic I/O devices on the development board have been labelled as follows: **red** - digital outputs which have been connected to LEDS, **gray** - digital inputs which have been connected to normally open push buttons, with external pull-down resistors.

4.3 Task 4.1: Online pre-practical quiz

Please answer the following questions in your pre-practical Q&A write-up **before the start of the practical**. This will be used as a reference document during the practical.

(a) Using the schematic for the STM32F0 Discovery Board at the end of the practical sheet, identify which General Purpose Input/Output port(s) **AND** pins the user controlled LEDS are connected to on the STM32F0 Discovery Board.

(b) Name all the registers which are used to configure the digital output operation of the ports connected to the user controlled LEDs on the STM32F0 Discovery Board.

(c) Using the schematic for the STM32F0 Discovery Board at the end of the practical sheet, identify which port(s) the user controlled Normally Open (N.O.) push button is connected to on the STM32F0 Discovery Board.

(d) Name all the registers which are used to configure the digital input operation of the port pin connected to the user controlled push button on the STM32F0 Discovery Board.

(e) Explain in detail what this C statement is shorthand for `GPIOC->MODER.`

(f) Explain what each of the following lines of code do when run on a the STM32F0 Discovery Board, using the Reference sheet at the end of this manual for guidance.

(i) `RCC->AHBENR |= 1<<19;`

(ii) `GPIOC->MODER|= 0x00000500;`

(iii) `GPIOC->ODR = 0b0000001100000000;`

(g) Write **`RCC->AHBENR |= 1<<19;`** in two alternative ways which produce the same effect on the STM32F051 microcontroller.

(h) Read through the questions in Section 4.4 and draw a flow chart which describes the **FULL** final program operation. Upload a pdf version of this to the pre-practical quiz on VULA.

We will now start our program design, which we are going to complete step by step.

4.4 Coding

Before opening your IDE, download the following files from VULA → *EEE2046F* → *Resources* → *Practicals* → *Coctemplate.c*

1. Open your IDE on your computer
2. Create a new STM32F0 project, using the instructions on the VULA lesson 3.2.
3. Delete the contents of your main.c file in your new project and copy and paste the code from the ctemplate into this file
4. Connect your STM32F0 discovery board to your computer to the PC using the USB cable. You are now ready to begin your task.

In your **.c** file start to write your program step by step. Build, debug and run your code after each question and make sure it behaves as expected, troubleshooting as you progress. Ensure that you provide detailed comments in each section. Use your flow chart written in your pre-practical write up to help with the coding in this practical. **NOTE:** Clear, concise and well commented coding is encouraged. Please ensure that you explain all steps in your code and follow good coding practice.

(a) Write a C function in your **main.c** file called **`void initGPIO(void),`** which initialises the GPIO ports as described in pre-practical. i.e. intialises the port pins connected to LEDs **LD3** and **LD4** and pushbutton **B1**. The LEDs must be OFF by default.

Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

(b) Create a function called **`void delay(void)`** which uses a for loop to create an approximate 1 second delay.

Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

(c) Write code in your main **`for`** loop to turn on **LD3**. Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

(d) Write code in your main **`for`** loop to turn on **LD4** (i.e. **LD3** = OFF, **LD4** = ON). Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

(e) Write code in your main **`for`** loop which does the following:

   (a) **LD3** = ON, **LD4** = OFF
   (b) delay of approximately 1 second

(c) **LD3** = OFF, **LD4** = ON

(d) delay of approximately 1 second

Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

(f) Now create a function called **`void blinky(void)`** implements the code written under in the previous question and run this in your main **`for`** loop in place of that code.

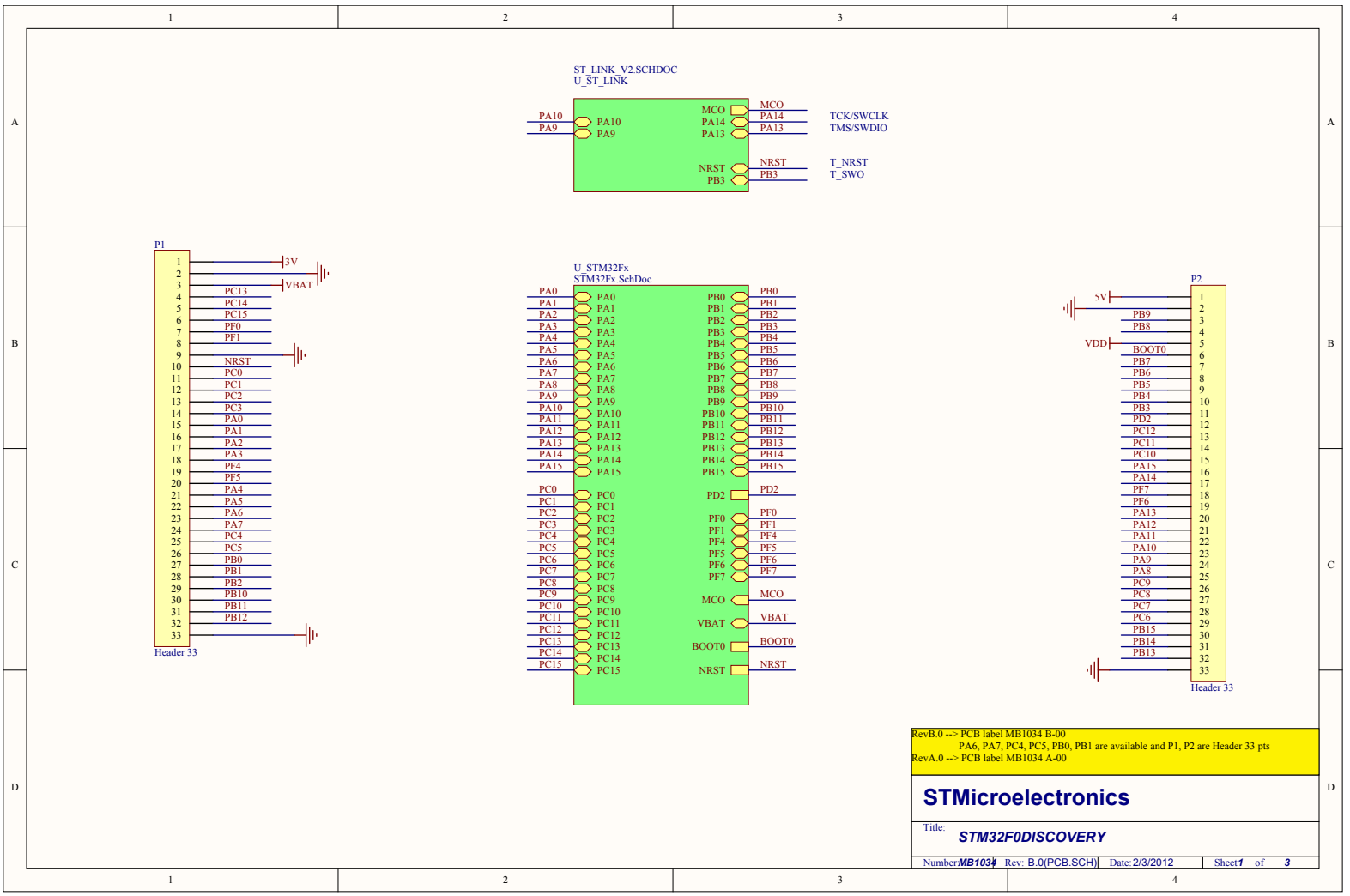
Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

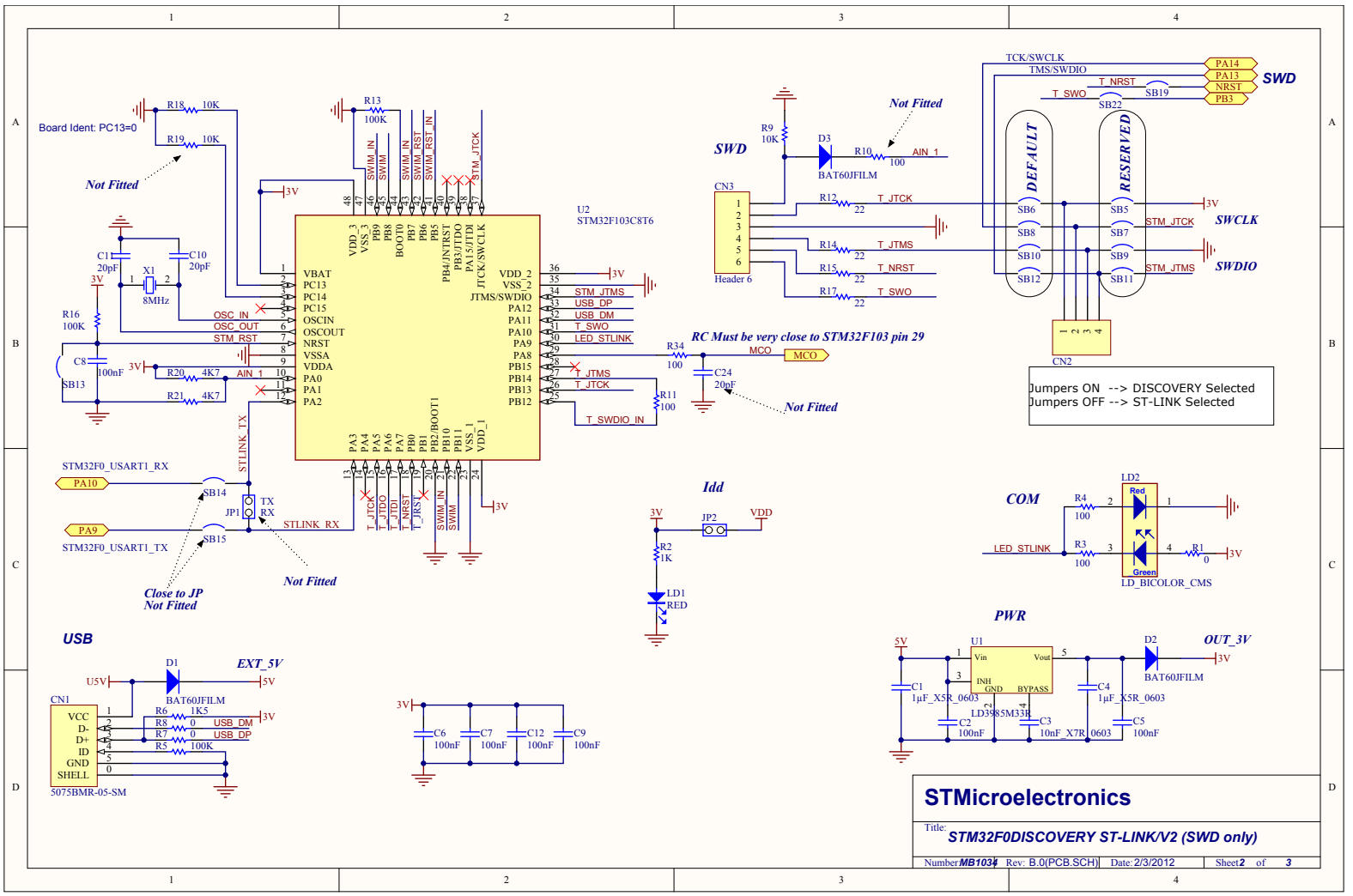
(g) Write code that only starts the blinky routine once push button **B1** has been pressed.

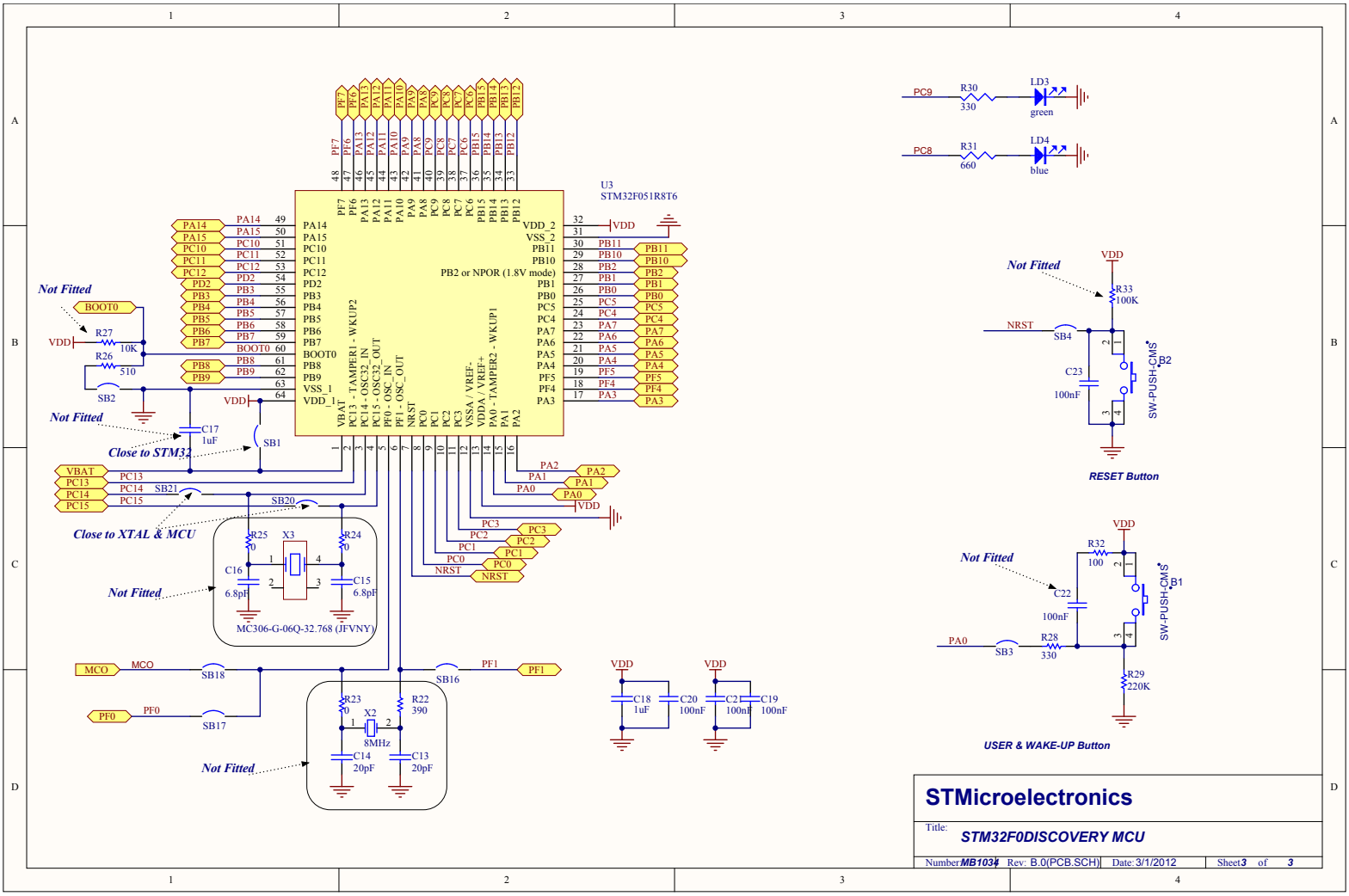
Copy this function and paste it in your report. Copy this code segment and upload it to the post-practical quiz on VULA.

4.5 Task 4.2: Online post-practical quiz

1. Upload your code segments to the text boxes on VULA as described above.
2. **Ensure that your main.c file is well commented and upload it to VULA. It will be compiled, run and evaluated by the tutors.**
3. Take a short video clip of you board showing your program's final operation. (if not possible please email me)


ST_LINK_V2.SCHDOC
U_ST_LINK
PA10
PA9
MCO
PA14
PA13
NRST
PB3
TCK/SWCLK
TMS/SWDIO
T_NRST
T_SWO
P1
3V
VBAT
NRST
Header 33
U_STM32Fx
STM32Fx.SchDoc
BOOT0
P2
5V
VDD
Header 33
RevB.0 --> PCB label MB1034 B-00
PA6, PA7, PC4, PC5, PB0, PB1 are available and P1, P2 are Header 33 pts
RevA.0 --> PCB label MB1034 A-00
STMicroelectronics
Title:
STM32F0DISCOVERY
Number:MB1034
Rev: B.0(PCB.SCH)
Date: 2/3/2012
Sheet 1 of 3


Board Ident: PC13=0
Not Fitted
U2
STM32F103C8T6
SWD
CN3
Header 6
BAT60JFILM
DEFAULT
RESERVED
SWCLK
SWDIO
CN2
Jumpers ON --> DISCOVERY Selected
Jumpers OFF --> ST-LINK Selected
RC Must be very close to STM32F103 pin 29
Idd
COM
LD_BICOLOR_CMS
PWR
USB
EXT_5V
5075BMR-05-SM
Close to JP
Not Fitted
STMicroelectronics
STM32F0DISCOVERY ST-LINK/V2 (SWD only)
Number:MB1034
Rev: B.0(PCB.SCH)
Date: 2/3/2012
Sheet 2 of 3


U3
STM32F051R8T6
PB2 or NPOR (1.8V mode)
Not Fitted
Close to STM32
Close to XTAL & MCU
MC306-G-06Q-32.768 (JFVNY)
8MHz
LD3
green
LD4
blue
RESET Button
USER & WAKE-UP Button
SW-PUSH-CMS
STMicroelectronics
STM32F0DISCOVERY MCU
MB1034
B.0(PCB.SCH)
3/1/2012

Bibliography

[1] Randal E. Bryant and David R. O'Hallaron. *Computer systems: A programmers perspective*. Prentice Hall, second edition edition, 2011.

[2] cplusplus.com. sprintf. `http://www.cplusplus.com/reference/cstdio/sprintf/`, 2015.

[3] nexperia. HEF4011B quad 2-input NAND gate. Datasheet Rev. 6-10 December 2015, 2017.

[4] STMicroelectronics. STM32F0x1/STM32F0x2/STM32F0x8 advanced ARM-based 32-bit MCUs reference manual. Reference Manual RM0091, DocID018940 Rev. 5, 2014.