TUTORIAL: Digital Out

Name: Yechan Kim ID: 22100153

# I. Introduction

In this lab, we will learn how to control the Digital Output of GPIOs of the MCU board to turn on/off an LED.

The objectives of this lab are to learn how to

* Read and configure registers of digital GPIO of MCU
* Program firmware to control digital input/output pins

### Hardware

NUCLEO -F411RE

### Software

Keil uVision IDE, CMSIS, EC\_HAL

### Documentation

[STM32 Reference Manual](https://ykkim.gitbook.io/ec/stm32-m4-programming/hardware/nucleo-f411re#manual-documentation)

# II. Basics of GPIO Out

## A. Bit Operation

Fill in the blanks. You should write the answer in both hexa-decimal and binary number.

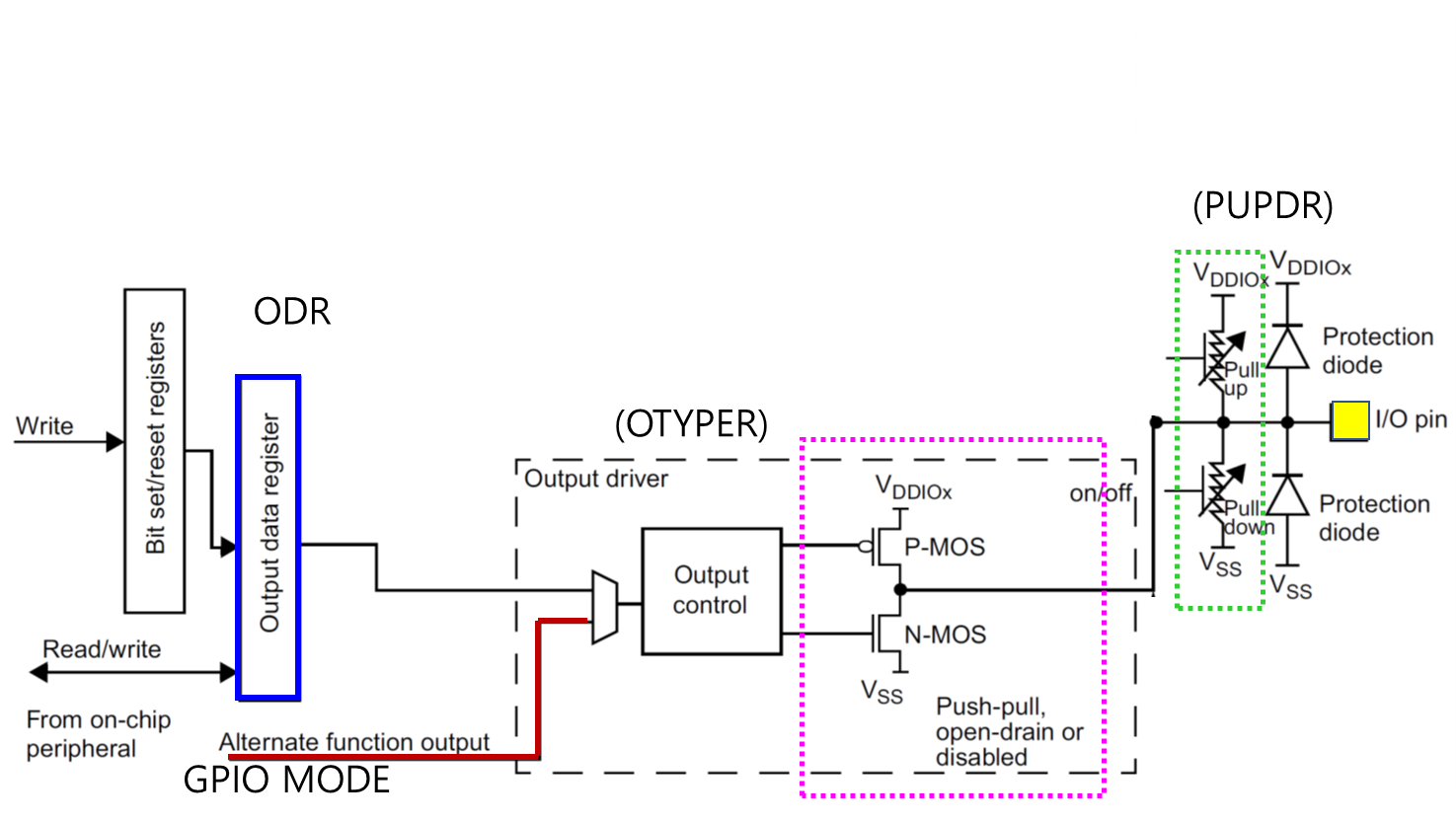
|  |  |  |  |
| --- | --- | --- | --- |
|  | Bit operation | Description | Answer |
|  | 0xA & 0x2 | AND | 0x2 |
|  | 0x3 | 0xC | OR | 0xF |
|  | ~0x7 | NOT | 0x8 |
|  | 0xF ^ 0x5 | XOR(Toggle) | 0xA |
|  | 0x1 << 3 | Shift left | 0x8 |
|  | 0xC >> 2 | Shift right | 0x3 |
|  | 0x11 |= 1<<3 | Set bit | 0x19 |
|  | 0xFF &= ~1<<4 | Clear bit | 0xEF |
|  | val=0x0F &1<<3 | Read bit | 0x0F |

## B. GPIO Digital Out Register

List GPIO registers for this LAB

|  |  |  |
| --- | --- | --- |
| Type | Register Name | Description |
| GPIO | GPIOx\_MODER | Mode: Output/Input/Analog |
|  | GPIOx\_OTYPER | Output Type: Opendrain/Push-Pull |
|  | GPIOx\_OSPEEDR | Output Speed: |
|  | GPIOx\_PUPDR | Pull-Up Pull-Down: |
|  | GPIOx\_ODR | Output Data Register |

Schematic



Process of GPIOx register initiation

|  |
| --- |
| 0. Enable Peripheral Clock (**AHB1ENR**)  1. Configure as Digital Output (**MODER**)  2. Configure pull-up/down resistors (**PUPDR**)  3. For Output: Configure Output Type (**OTYPE**)  4. For Output: Configure Output Speed (**OSPEEDR**)  5. Output Data **(ODR)** |

# III. Tutorial

## A. Register Configuration

**1. GPIO: Digital Out - Pin Initialization & Set LED**

Port A Pin 5 / Output / Push-Pull / No Pull-Up & No Pull-Down / Default Speed

use **#define LED\_PIN 5**

* **MODER:** Output (MODER5[1:0]=01)

|  |
| --- |
| GPIOA->MODER & = ~(3<<( LED\_PIN \*2)); // clear bits at both [10] and [11]  GPIOA->MODER | = 1<<(LED\_PIN \*2); // set bit at [10] |

|  |
| --- |
| 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진  자동 생성된 설명 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Register** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Initial** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| **Logic** | Bitwise OR | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **mask** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Value** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 0 | 1 | x | x | x | x | x | x | x | x | x | x |

* **OTYPER:** Push-Pull (OT5=0)

|  |
| --- |
| GPIOA->OTYPER &= ~(1<< LED\_PIN) |

|  |
| --- |
| 텍스트, 폰트, 스크린샷, 라인이(가) 표시된 사진  자동 생성된 설명 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Register** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Initial** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| **Logic** | Bitwise AND | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **mask** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **1** | **1** | **1** | **1** | **1** |
| **Value** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **0** | **x** | **x** | **x** | **x** | **x** |

* **PUPDR:** no pull-up no pull-down (PUPDR5[1:0]=00)

|  |
| --- |
| GPIOA->PUPDR &= ~(3<<( LED\_PIN \*2));  GPIOA->PUPDR |= 0<< ( LED\_PIN \*2) |

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Register** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Initial** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| **Logic** | Bitwise OR | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **mask** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Value** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 0 | 0 | x | x | x | x | x | x | x | x | x | x |

* **ODR:** Set LED (ODR5=1)

|  |
| --- |
| GPIOA->ODR |= 1<<LED\_PIN; |

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Register** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** | |
| **Initial** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | |
| **Logic** | Bitwise OR | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **mask** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | |
| **Value** | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 1 | x | x | x | x | x | |

## B. Programming

**Preparation**

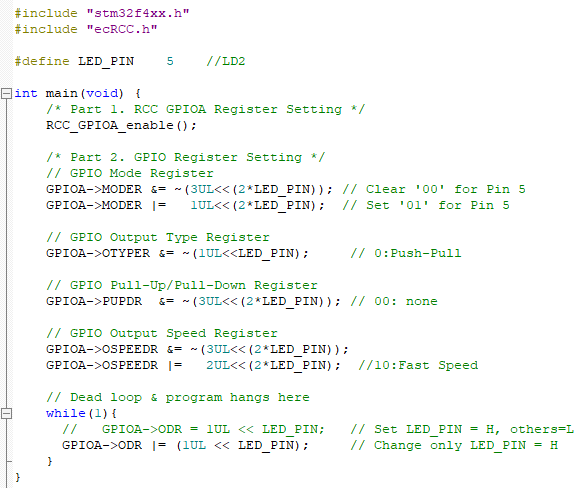
* + Open the program ‘Keil uVision5’ and create a new project.
  + “ /repos/EC/Tutorial/TU\_GPIO\_Digital\_Out\_LED/”
  + Name the project as ‘**TU\_GPIO\_Digital\_Out\_LED**’.
  + Create a new item(file) and name it as ‘**TU\_GPIO\_Digital\_Out\_LED.c’**
  + Copy and paste from the source code ‘[**TU\_GPIO\_Digital\_Out\_LED\_student.c**](https://github.com/ykkimhgu/EC-student/tree/main/tutorial/tutorial-student)’.
  + Download [ecRCC\_student.h and ecRCC\_student.c](https://github.com/ykkimhgu/EC-student/tree/main/include/lib-student). Then, change the file names as **ecRCC2.h** and **ecRCC2.c**.
  + Save them in \lib folder: “**repos/EC/include**”

**Exercise**

This is an example code of turning ON LED *without* button input. Do not worry if you do not understand what this code means yet. You will learn one by one in the following few weeks.

Fill in the empty spaces in the code. Then, compile(F7) and flash(F8) the source code onto the MCU board. Verify the program by checking if the LED is turned ON.

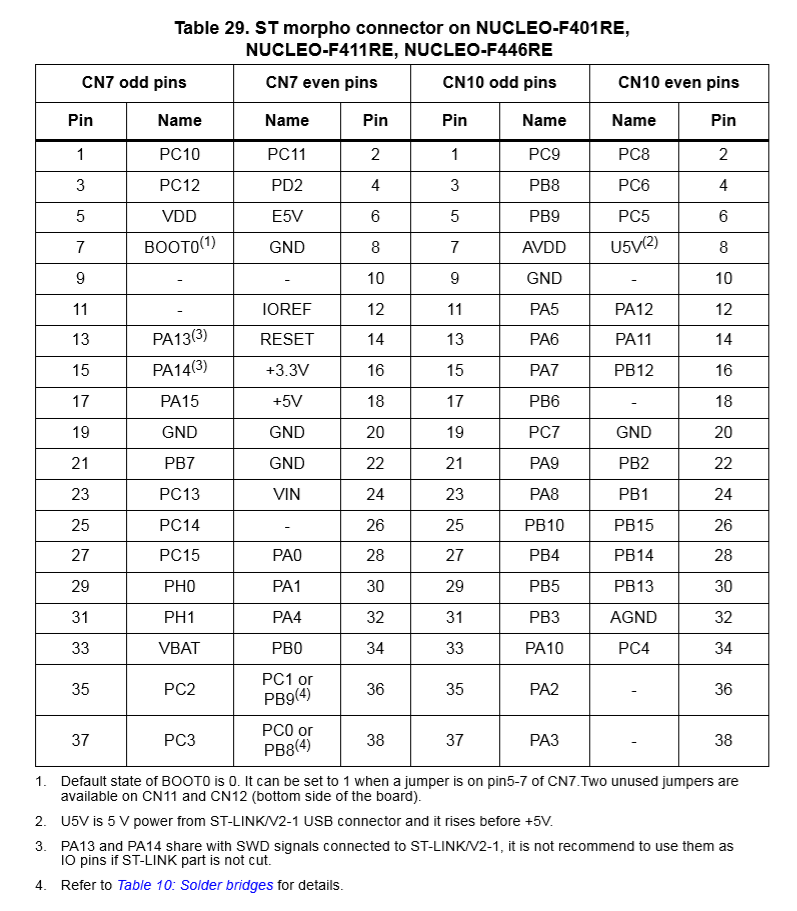
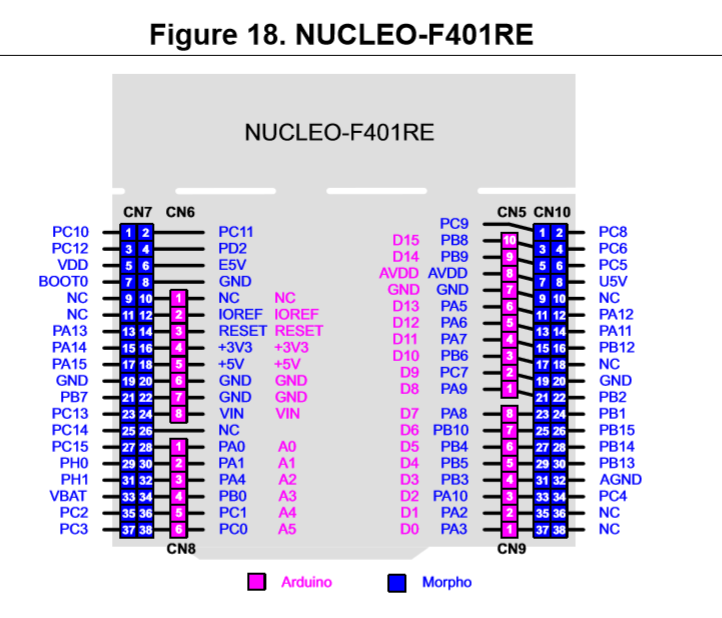
**Solution**



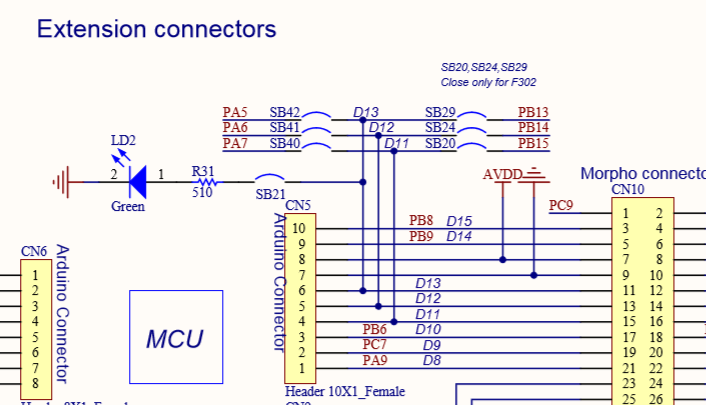
## Appendix

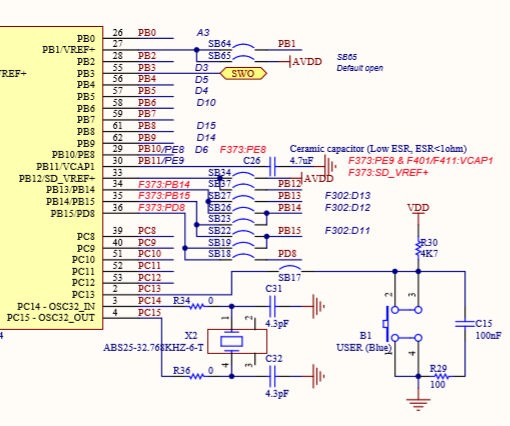
[See here for MCU resources](https://ykkim.gitbook.io/ec/resource/nucleo-f411re)

1. Pin Configuration of NUCLE-F401RE

****

1. LED/Button Circuit Diagram





## Results

**Source Code**

/\*\*

  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

  \* @author  SSSLAB

    \* @Mod       2024-8-23 by YKKIM

  \* @brief   Embedded Controller:  Tutorial Digital Out

  \*                  - Turn on LED LD2

  \*

  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

// GPIO Mode             : Input(00), Output(01), AlterFunc(10), Analog(11, reset)

// GPIO Speed            : Low speed (00), Medium speed (01), Fast speed (10), High speed (11)

// GPIO Output Type: Output push-pull (0, reset), Output open drain (1)

// GPIO Push-Pull    : No pull-up, pull-down (00), Pull-up (01), Pull-down (10), Reserved (11)

#include "stm32f4xx.h"

#include "ecRCC2.h"

//#define LED\_PIN    PA\_5       //LD2

#define LED\_PIN 5

int main(void) {

        /\* Part 1. RCC GPIOA Register Setting \*/

        RCC\_HSI\_init();

        RCC\_GPIOA\_enable();

        /\* Part 2. GPIO Register Setting \*/

        // GPIO Mode Register

        GPIOA->MODER &=  ~(3<<( LED\_PIN \*2));                                           // Clear '00' for Pin 5

        GPIOA->MODER |=  1<<(LED\_PIN \*2);                                           // Set '01' for Pin 5

        // GPIO Output Type Register

        GPIOA->OTYPER &=~(1<<LED\_PIN) ;     Clear '00'

                                      //

                                                                                    // 0:Push-Pull

        // GPIO Pull-Up/Pull-Down Register

        GPIOA->PUPDR  &= ~(3<<( LED\_PIN \*2));                                           // 00: none

        // GPIO Output Speed Register

        GPIOA->OSPEEDR &=  ~(3<<( LED\_PIN \*2));

        GPIOA->OSPEEDR |= 2<<( LED\_PIN \*2);                                     //10:Fast Speed

        // Dead loop & program hangs here

        while(1){

            GPIOA->ODR |= (1 << LED\_PIN);  // Set LED\_PIN = H, others=L

        }

        return 0;

}

**Screen Shot**

