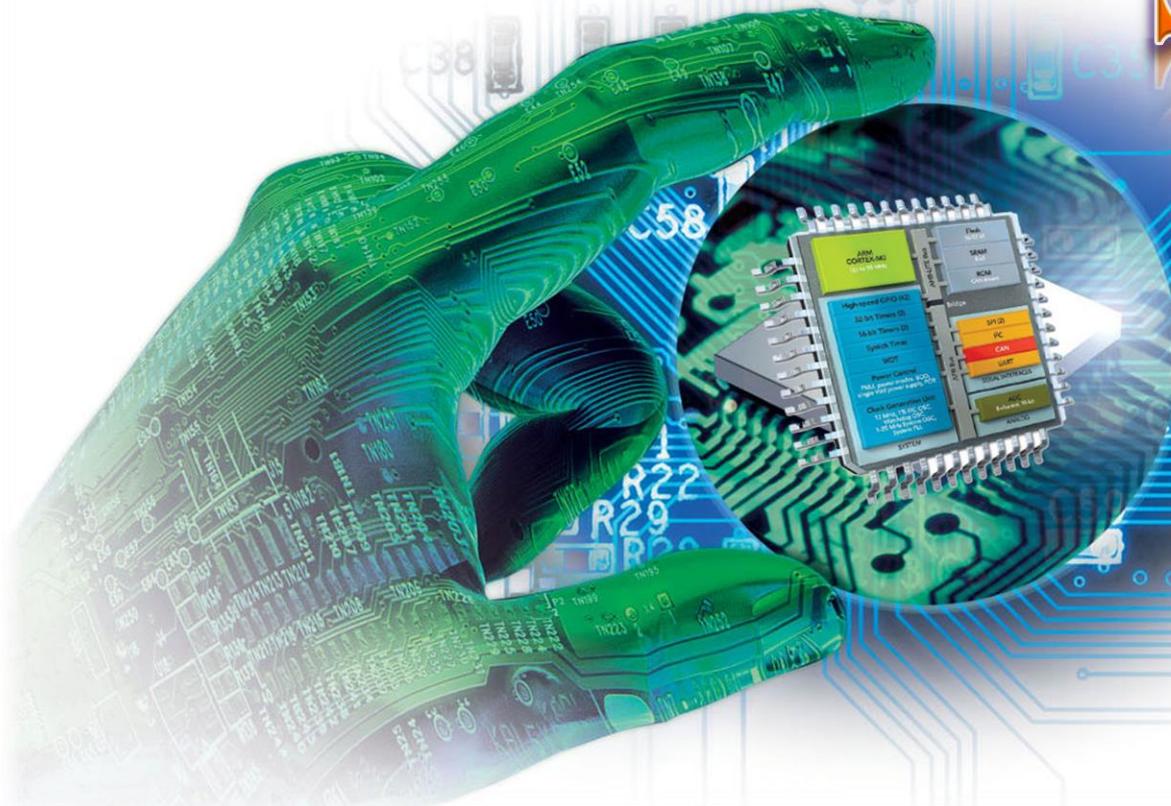


# مُتَحَكِّمَات

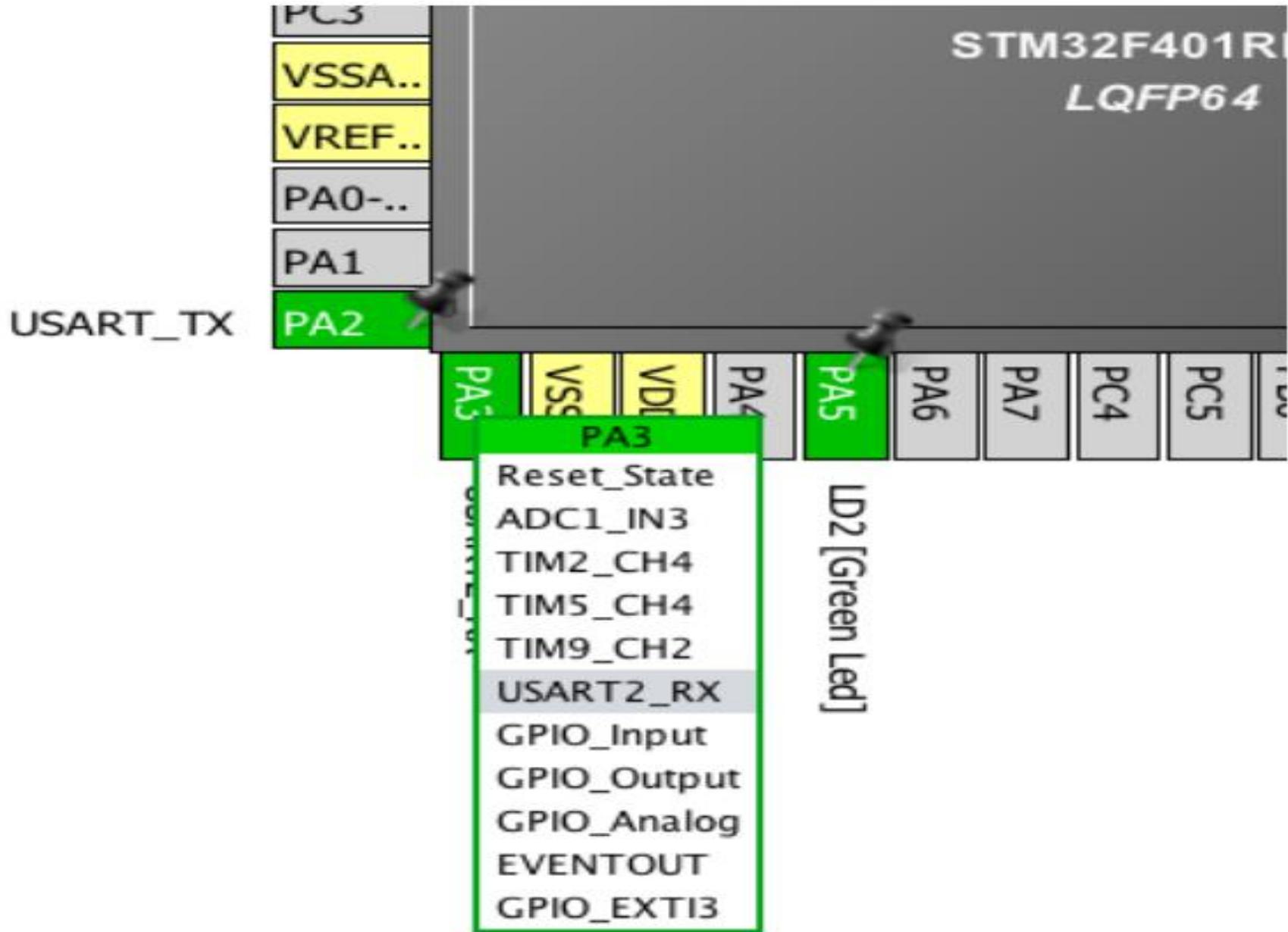
# STM32

## 2



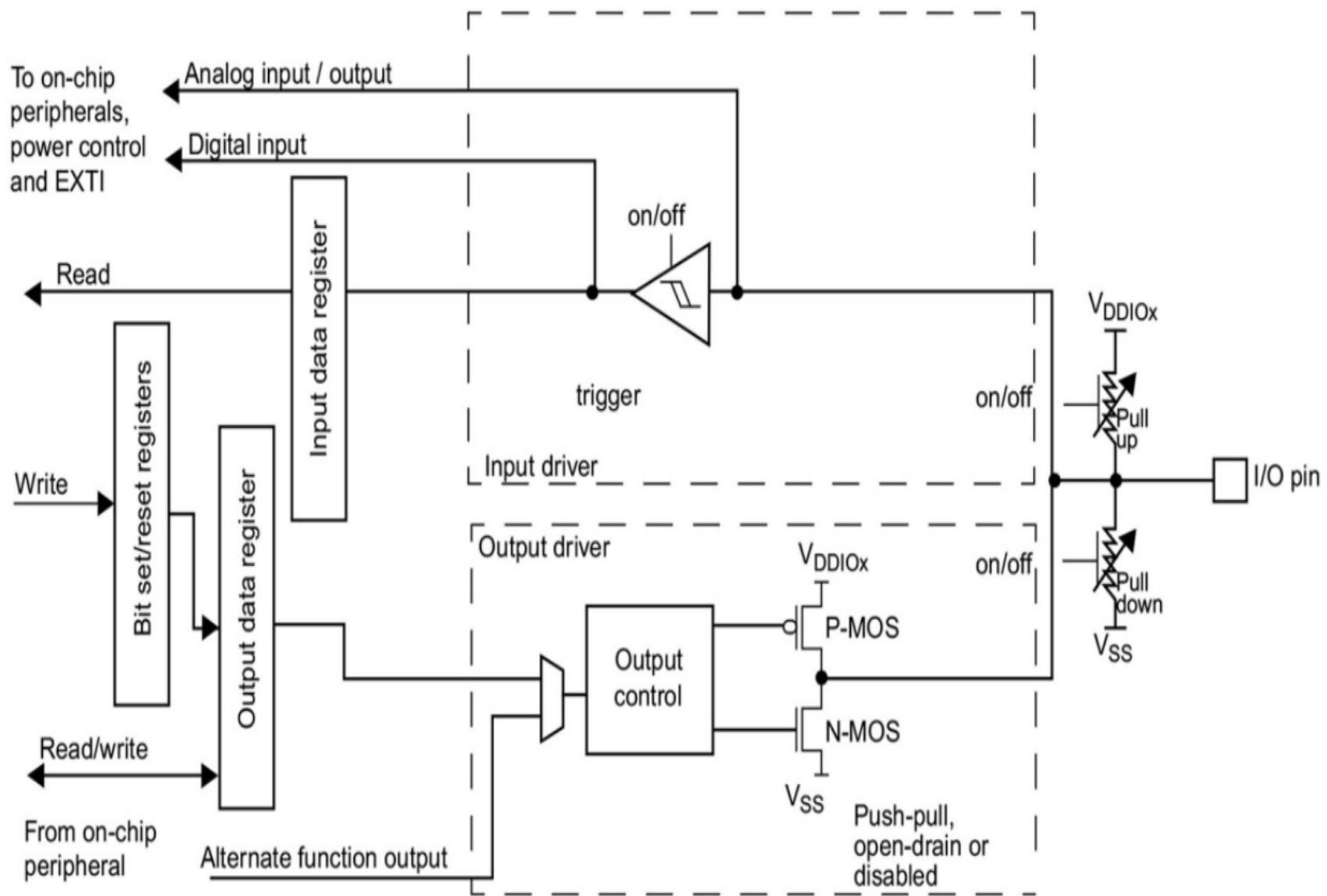
- أنماط عمل أقطاب المتحكم GPIO Mode
- برمجة أقطاب الخرج في متحكمات stm32
- التوابع المستخدمة من مكتبة HAL للتحكم بالمخارج الرقمية في متحكم STM32.
- بناء أول تطبيق لإضاءة ليد باستخدام متحكمات stm32 و مكتبة HAL
- برمجة أقطاب الدخل في متحكمات stm32
- التوابع المستخدمة من مكتبة HAL للتحكم بالمداخل الرقمية في متحكم STM32.
- بناء تطبيق لإضاءة ليد من خلال مفتاح لحظي باستخدام متحكمات stm32 و مكتبة HAL

# أنماط عمل أقطاب المتحكم .1 GPIO Mode

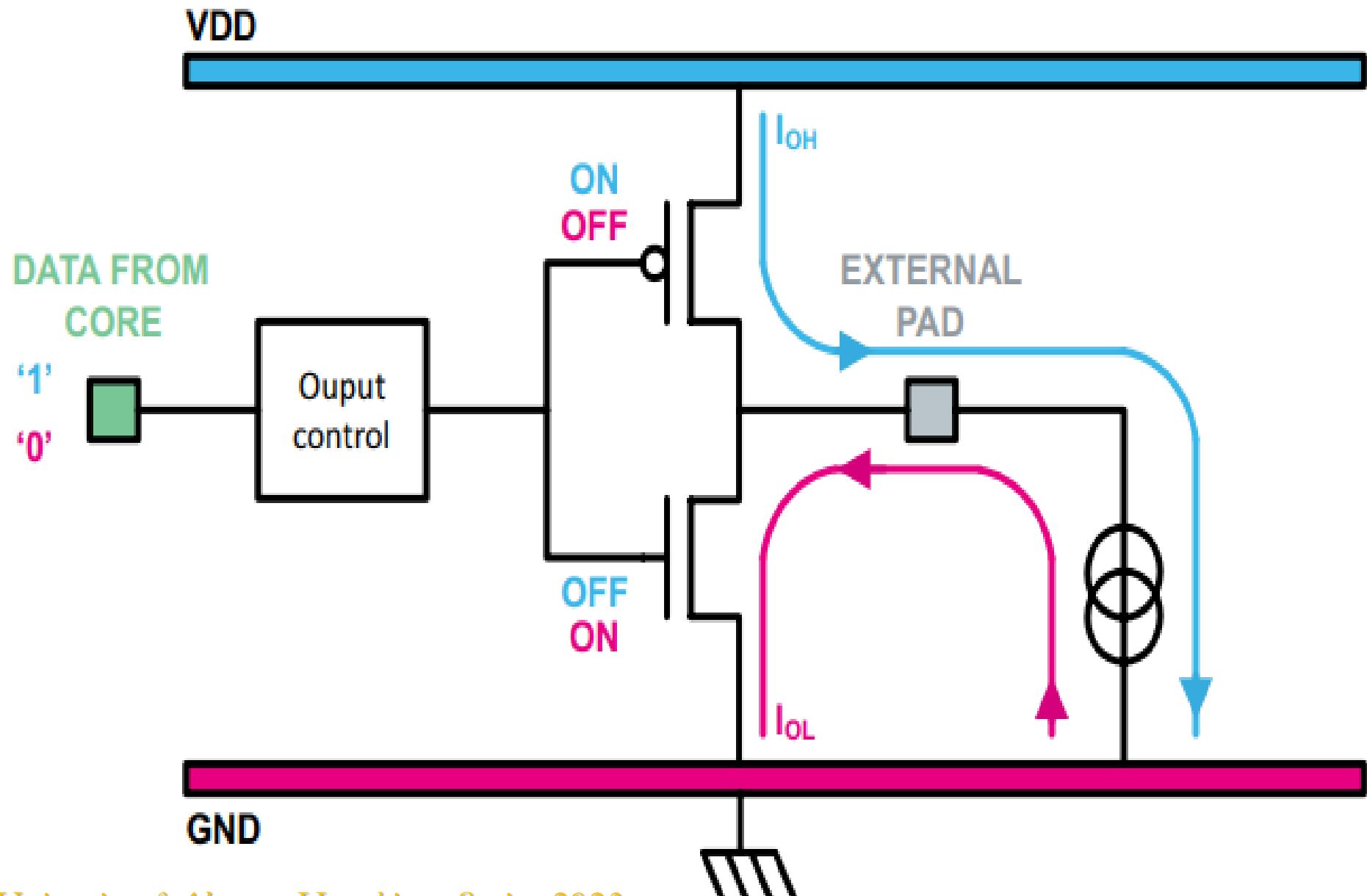


# أنماط عمل أقطاب المتحكم .1

## GPIO Mode



## برمجة أقطاب الخرج في متحكمات stm32 .2



## 2. برمجة أقطاب الخرج في متحكمات stm32

نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

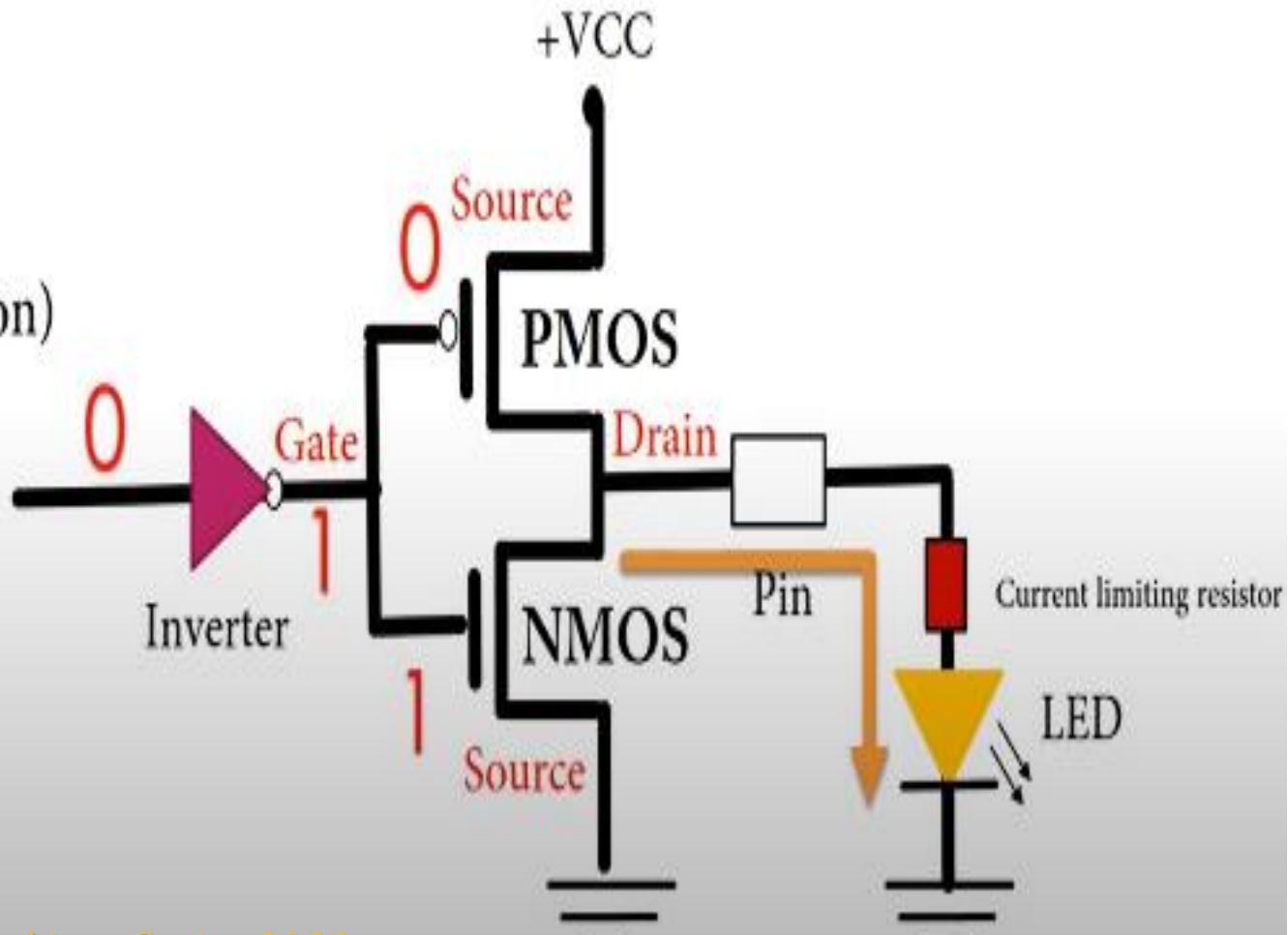
□ **نمط PUSH-PULL:** فبفرض تم وصل مصعد ليد مع قطب الإخراج للمتحكم المصغر، فعند تطبيق واحد منطقي على القطب يصبح الترانزستور PMOS بحالة تشغيل on وبالتالي يتم تطبيق جهد الـ VCC على مصعد الليد ويضيء الليد، أما عند تطبيق صفر منطقي على القطب يصبح الترانزستور NMOS بحالة تشغيل on وبالتالي يتم تطبيق جهد الأرضي GND على مصعد الليد ويطفأ الليد كما هو موضح بالشكل التالي:

نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

**نط** PUSH-PULL

## Push-Pull

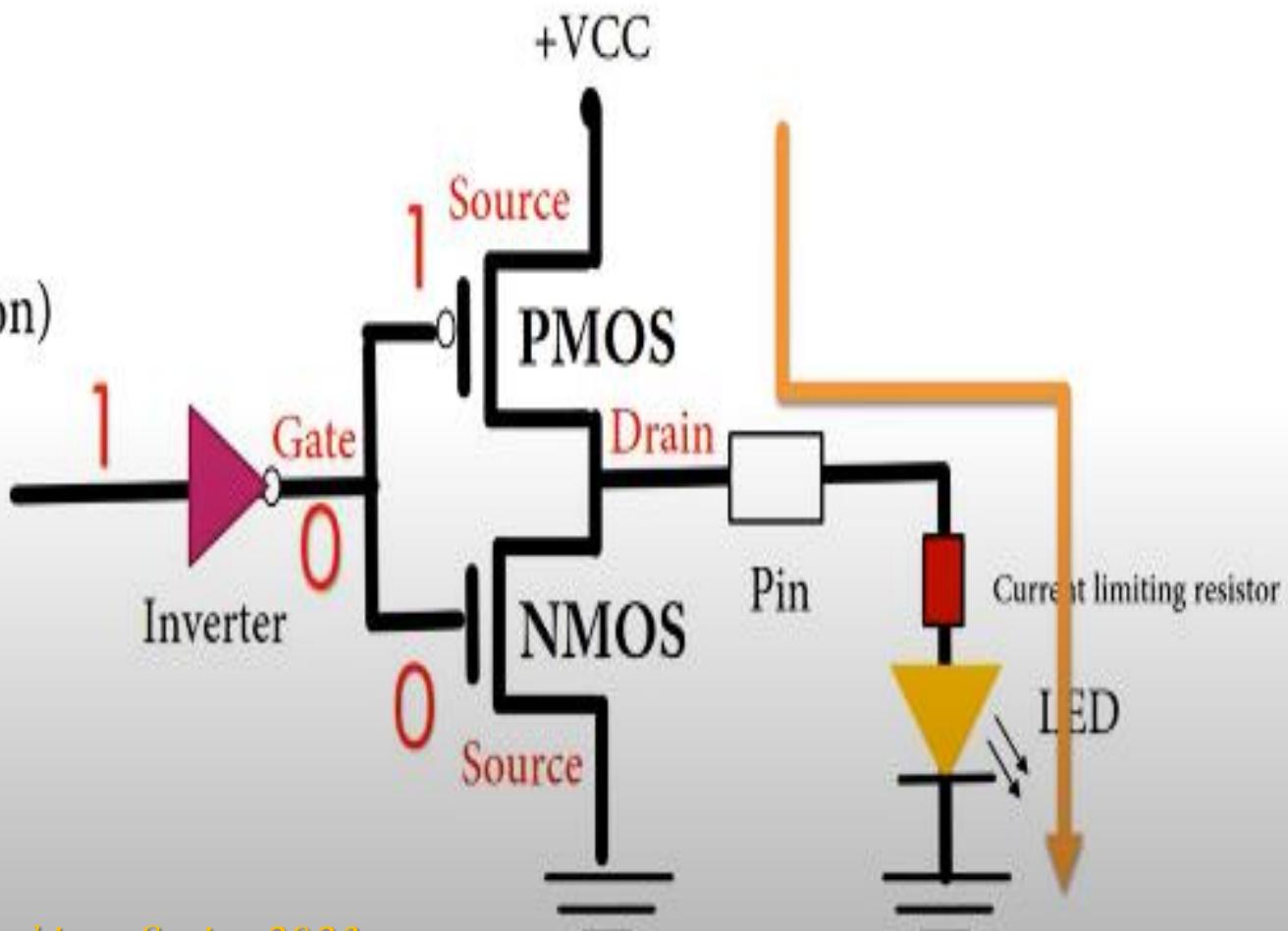
(Default Configuration)



نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO  
نط **PUSH-PULL**

## Push-Pull

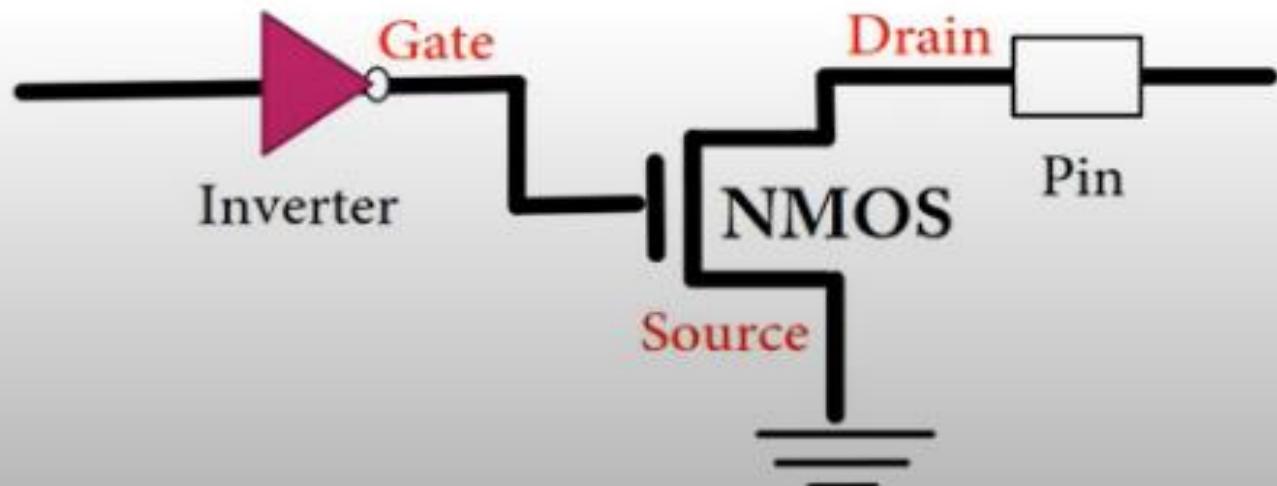
(Default Configuration)



نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO:

- **نمط Open-Drain** في هذا النمط يمكن للمتحكم أن يعمل كصرف للتيار sink فقط، فيتم قيادة قطب الخرج من خلال ترانزستور واحد من نوع NMOS كما هو موضح في الأشكال التالية.

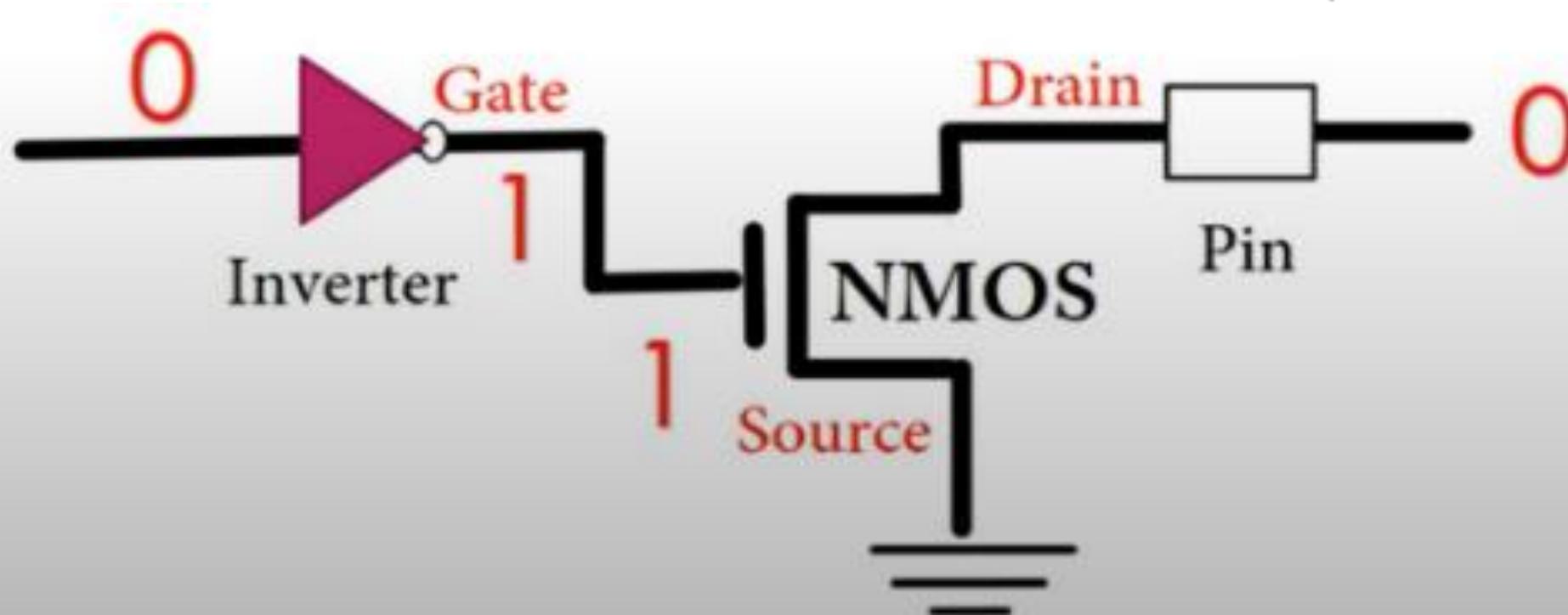
### Open Drain



## 2. برمجة أقطاب الخرج في متحكمات stm32

نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

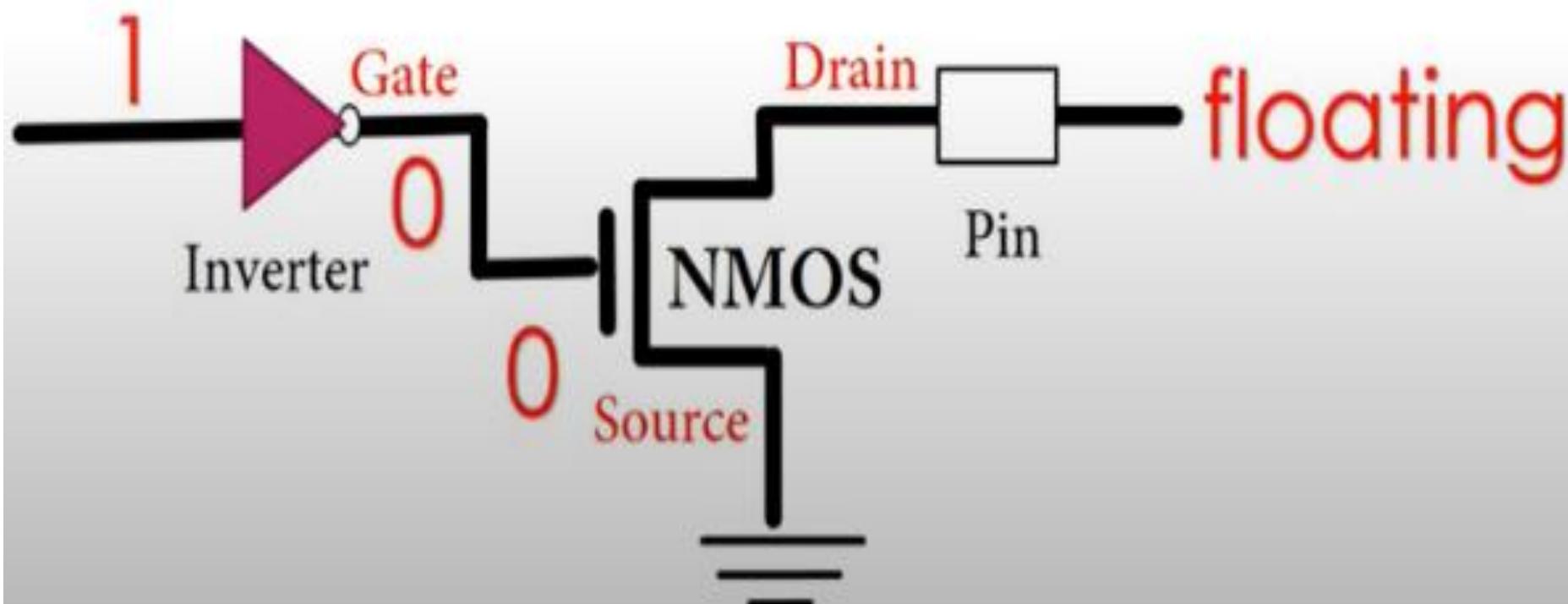
□ **نمط Open-Drain** في هذا النمط يمكن للمتحكم أن يعمل كصرف للتيار sink فقط، فيتم قيادة قطب الخرج من خلال ترانزستور واحد من نوع NMOS كما هو موضح في الأشكال التالية:



## 2. برمجة أقطاب الخرج في متحكمات stm32

نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO:

□ **نمط Open-Drain** في هذا النمط يمكن للمتحكم أن يعمل كصرف للتيار sink فقط، فيتم قيادة قطب الخرج من خلال ترانزستور واحد من نوع NMOS كما هو موضح في الأشكال التالية:



## 2. برمجة أقطاب الخرج في متحكمات stm32

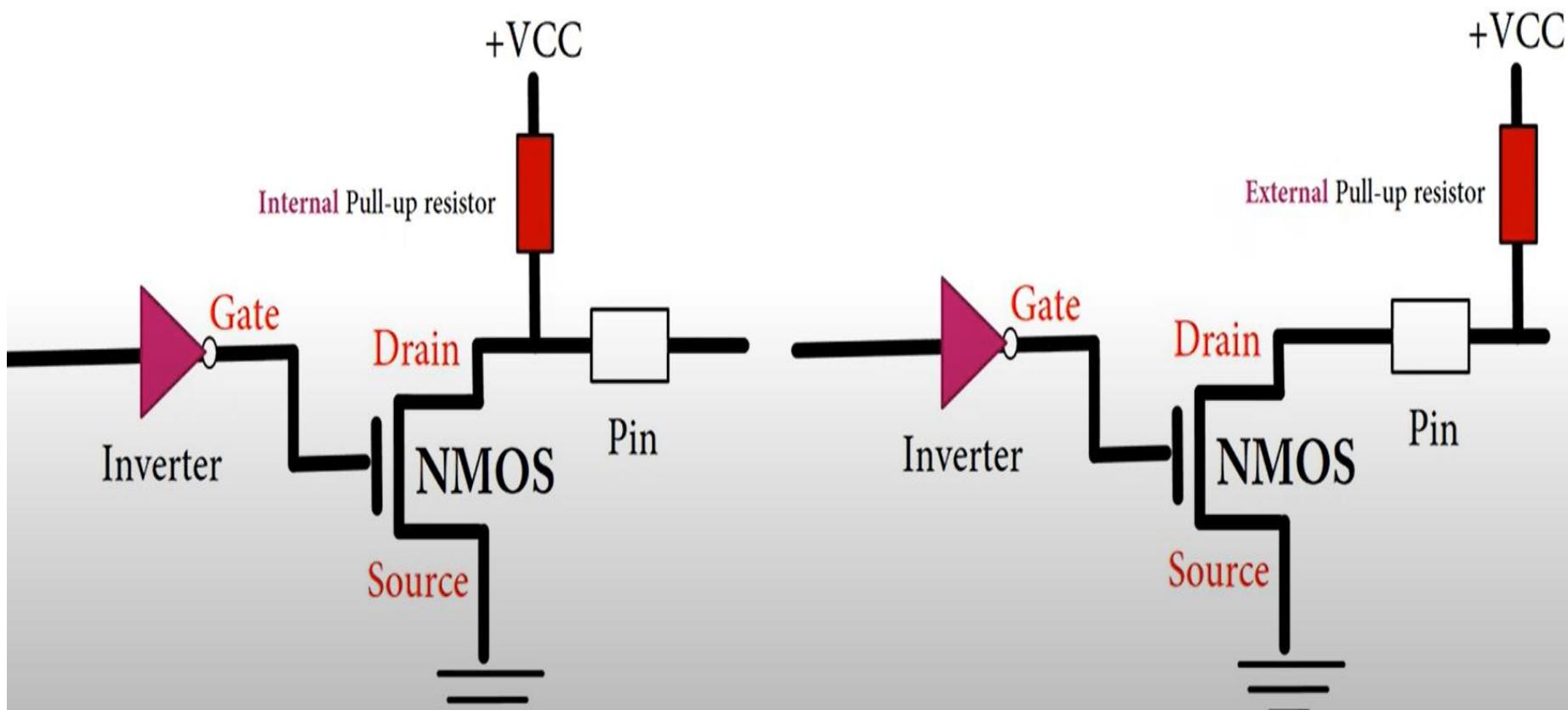
نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

□ **نمط Open-Drain** نلاحظ من الأشكال السابقة أنه عند تطبيق صفر منطقي على قطب المتحكم يصبح الترانزستور بحالة توصيل on وبالتالي يتم توصيل الحمل مع الأرضي GND، أما في حالة تطبيق واحد منطقي على قطب المتحكم يصبح الترانزستور بحالة فصل off وبالتالي يصبح الجهد المطبق على الحمل عائماً غير محدد floating لذا ولحل هذه المشكلة لابد من توصيل مقاومة رفع مع قطب المتحكم ، يمكن توصيل هذه المقاومة خارجياً أو يمكن تفعيل مقاومة الرفع الداخلية الموجودة ضمن المتحكم كما في الشكل التالي:

نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

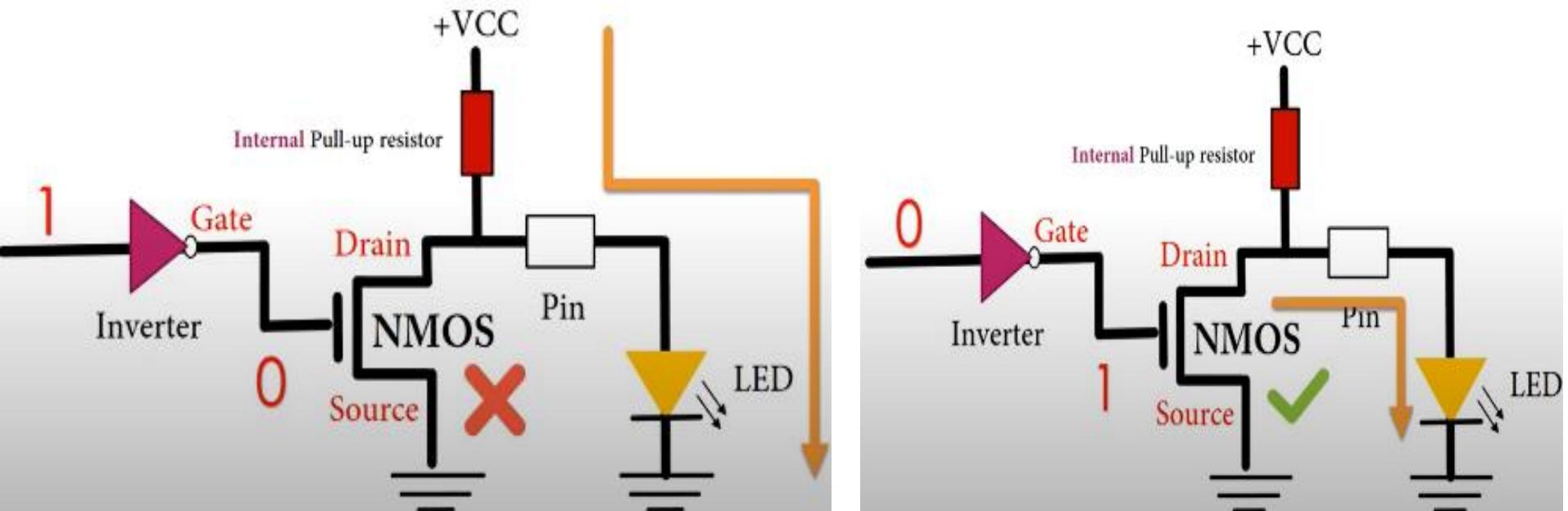
نمط  Open-Drain

## Open Drain with Pull-up Resistor -



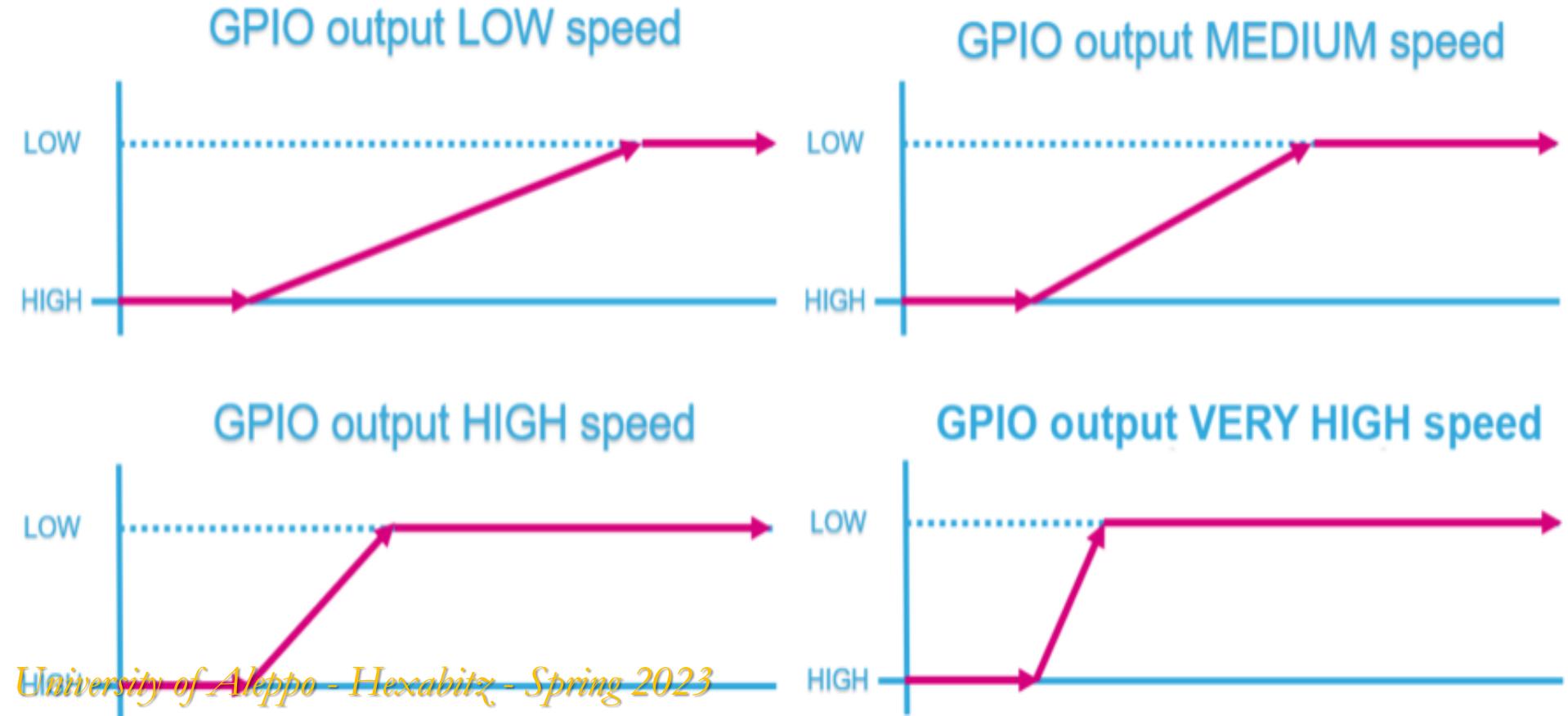
نميز نمطي عمل لكل قطب خرج من أقطاب المتحكم :GPIO

## نمط Open-Drain



# برمجة أقطاب الخرج في متحكمات stm32

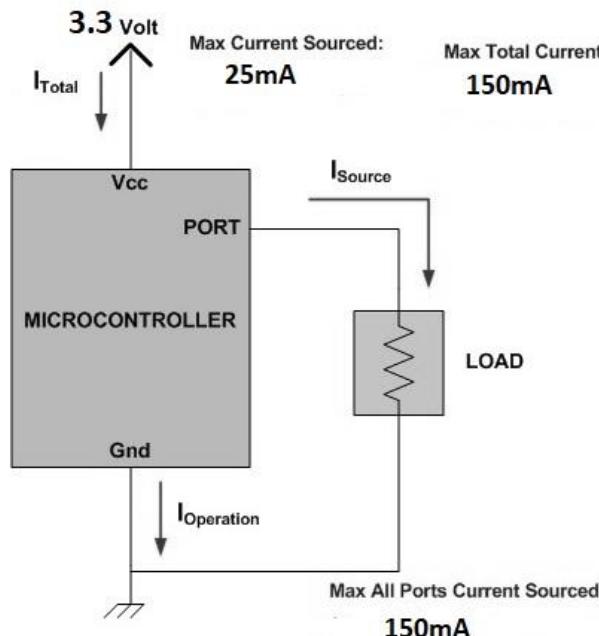
: يعني سرعة انتقال الإشارة من الحالة المنخفضة low إلى الحالة المرتفعة HIGH وبالعكس وهو ما يسمى بزمن الصعود rise time وزمن الهبوط fall time كما هو موضح بالشكل التالي:



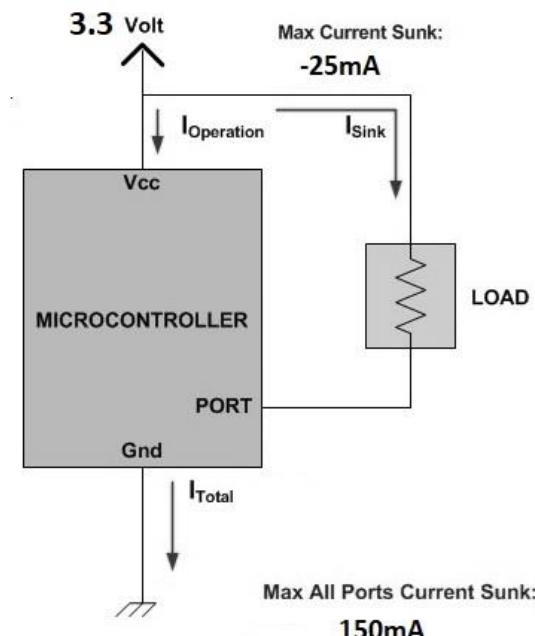
# ربط الأحمال مع مخارج المتحكم

إن وصل الأحمال مع أقطاب المتحكم يكون بطريقتين:

- A. يعمل القطب كمنبع لتيار تشغيل الحمل (Source).
- B. يعمل القطب كمصرف لتيار تشغيل الحمل (Sink).



A. Microcontroller's Port is used as a Current Source



B. Microcontroller's Port is used to Sink Current

## ربط الأحمال مع مخارج المتحكم

أهم الاعتبارات التي يجب أن تؤخذ بعين الاعتبار عن ربط أقطاب المتحكم إلى الأحمال هو:

- التيار الأعظمي المستهلك من قطب المتحكم (Vcc to Gnd) (Vcc to Gnd) الذي يمكن سحبه أو تصريفه عن طريق المتحكم بشكل كلي هو 150mA وفق المواصفات الكهربائية لعائلة متحكمات STM32.
- قيمة التيار التي يمكن سحبها أو تصريفها لقطب خرج من أقطاب المتحكم تتراوح عادةً من 25mA حسب المواصفات الكهربائية للمتحكم المصغر STM32.

## ربط الأحمال مع مخارج المتحكم

Symbol	Ratings	Max.	Unit
$I_{VDD}$	Total current into $V_{DD}/V_{DDA}$ power lines (source) <sup>(1)</sup>	150	
$I_{VSS}$	Total current out of $V_{SS}$ ground lines (sink) <sup>(1)</sup>	150	
$I_O$	Output current sunk by any I/O and control pin	25	
	Output current source by any I/Os and control pin	-25	mA

- لإعطاء واحد منطقي set أو صفر منطقي reset لقطب محدد من أي منفذ من منافذ المتحكم يقوم باستخدام التابع التالي من مكتبة HAL.

```
void HAL_GPIO_WritePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState);
```

# اسم المنفذ المراد التحكم بأحد أقطابه على سبيل المثال

لإعطاء واحد منطقي نكتب رقم القطب المراد التحكم به على سبيل المثال **GPIO\_PIN\_SET** وإعطاء صفر منطقي نكتب **GPIO\_PIN\_RESET**

# مُوَابِعِ الْمُسَكَّنِ مِنْ حَبْ - HAL - STM32 مُتَحَكِّم

مثال 1: لكتابة واحد منطقى على القطب رقم 10 من المنفذ D نستخدم  
التابع التالي:

**HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_10,  
GPIO\_PIN\_SET);**

مثال 2: لكتابة صفر منطقى على القطب رقم 5 من المنفذ A نستخدم  
التابع التالي:

**HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5,  
GPIO\_PIN\_RESET);**

□ لعكس الحالة المنطقية لأحد الأقطاب نستخدم التابع التالي:

```
HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx,  
                      uint16_t GPIO_Pin);
```

مثال: لعكس الحالة المنطقية للقطب رقم 5 من المنفذ A نستخدم التابع التالي:

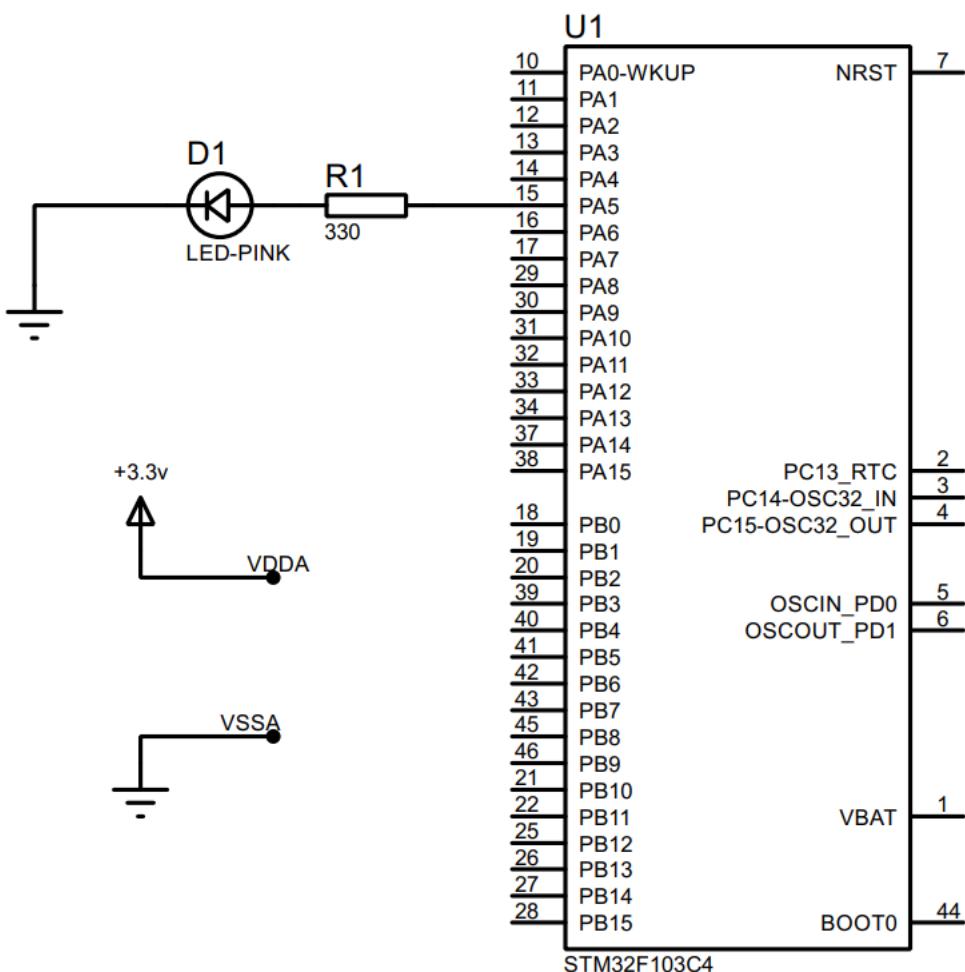
```
HAL_GPIO_TogglePin(GPIOA,GPIO_PIN_5);
```

□ لإضافة تأخير زمني بالمياللي ثانية نستخدم التابع التالي:

```
HAL_DelayMilliseconds)
```

# HAL و STM32 و مكتبة

سنقوم بتصميم تطبيق يقوم بعمل toggle لليد المتصل بالقطب PA5



# STM32 HAL و مكتبة STM32

نقوم بضبط إعدادات القطب PA5 كقطب خرج



Screenshot of the STM32CubeMX software interface showing the configuration of the PA5 pin.

The left sidebar shows categories: System Core, DMA, GPIO (selected), IWDG, NVIC, RCC, SYS, and WWDG. The main window has tabs: Pinout & Configuration, Clock Configuration, Project Manager, and Tools. The Pinout & Configuration tab is active.

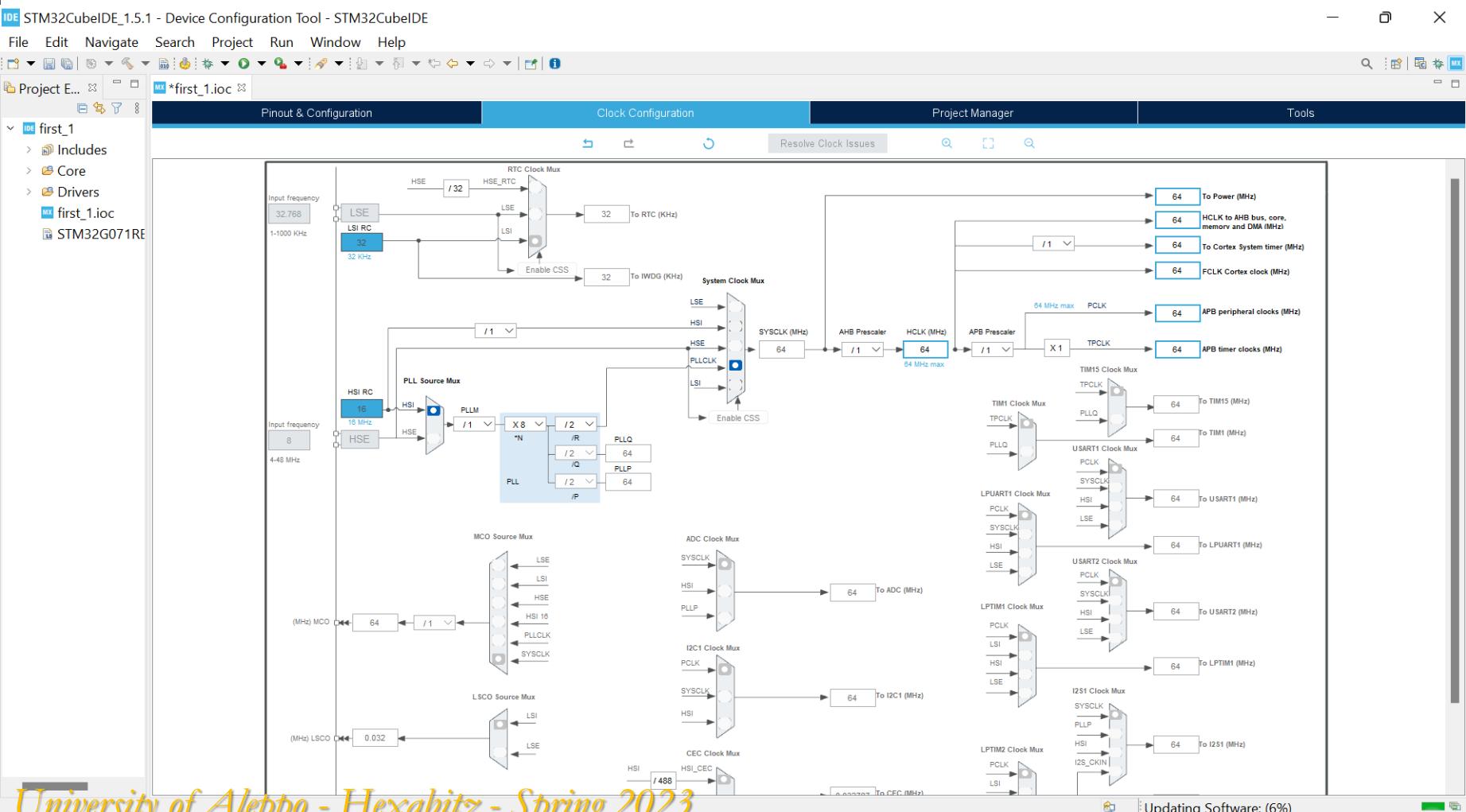
The GPIO Mode and Configuration panel shows the following settings for PA5:

Pin Name	Signal on	GPIO output level	GPIO mode	GPIO Pull	Maximum output speed	User Label	Modified
PA5	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low		<input type="checkbox"/>

The right side displays the STM32F103C4Tx LQFP48 package pinout diagram. The PA5 pin is highlighted in green at the bottom center. The diagram also shows other pins like VDD, VSS, PB0, PB1, PB2, PB3, PB4, PB5, PB6, PB7, PB8, PB9, PC13, PC14, PC15, PD0, PD1, NRST, VSSA, VDDA, PA0, PA1, PA2, PA3, PA4, PA6, PA7, PB0, PB1, PB2, PB10, PB11, VSS, and VDD.

# STM32 HAL و مكتبة HAL و STM32

نقوم بضبط تردد الساعة للمتحكم



## HAL و مكتبة STM32

□ نقوم بالضغط على **Project...Generate** أو من **Ctrl+s** أو من **code**، ليتم حفظ المشروع وتوليد الكود ثم نقوم بإضافة الجزء التالي:

```
#include "main.h"  
void SystemClock_Config(void);  
static void MX_GPIO_Init(void);  
int main(void) {  
    HAL_Init();  
    SystemClock_Config();  
    MX_GPIO_Init();
```

```
while (1)
{
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    HAL_Delay(500);
}
```

HAL:

□ هناك مجموعة من المسجلات تستخدم للتحكم بالمخارج الرقمية لمتحكم STM32Senkifi فقط بذكر المسجل المسؤول عن عمل set/reset للمنفذ أو لأحد الأقطاب الموجودة فيه

#### 7.4.6 GPIO port output data register (GPIOx\_ODR) (x = A..H)

Address offset: 0x14

Reset value: 0x0000 0000

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODRy**: Port output data ( $y = 0..15$ )

These bits can be read and written by software.

## HAL:

- لكتابه واحد منطقى على القطب رقم 5 من المنفذ A نكتب:
- GPIOA->ODR |= 1<<5; // Set the Pin PA5**
- و لكتابه صفر منطقى عليه نكتب:
- GPIOA->ODR &= ~(1<<5); // Reset the Pin PA5**
- كما يمكن جعل المنفذ بالكامل بحالة set من خلال كتابة :
- GPIOA->ODR = 0xFFFF; // Set the PORTA HIGH**
- كما يمكن جعل المنفذ بالكامل بحالة reset من خلال كتابة :
- GPIOA->ODR = 0x00; // Reset the PORTA**

```
#include "main.h"
```

```
void SystemClock_Config(void);  
static void MX_GPIO_Init(void);
```

```
int main(void)
```

```
{
```

```
    HAL_Init();
```

```
    SystemClock_Config();
```

```
    MX_GPIO_Init();
```

```
while (1)
```

```
{
```

```
    GPIOA->ODR = 1<<5;
```

```
    HAL_Delay(500);
```

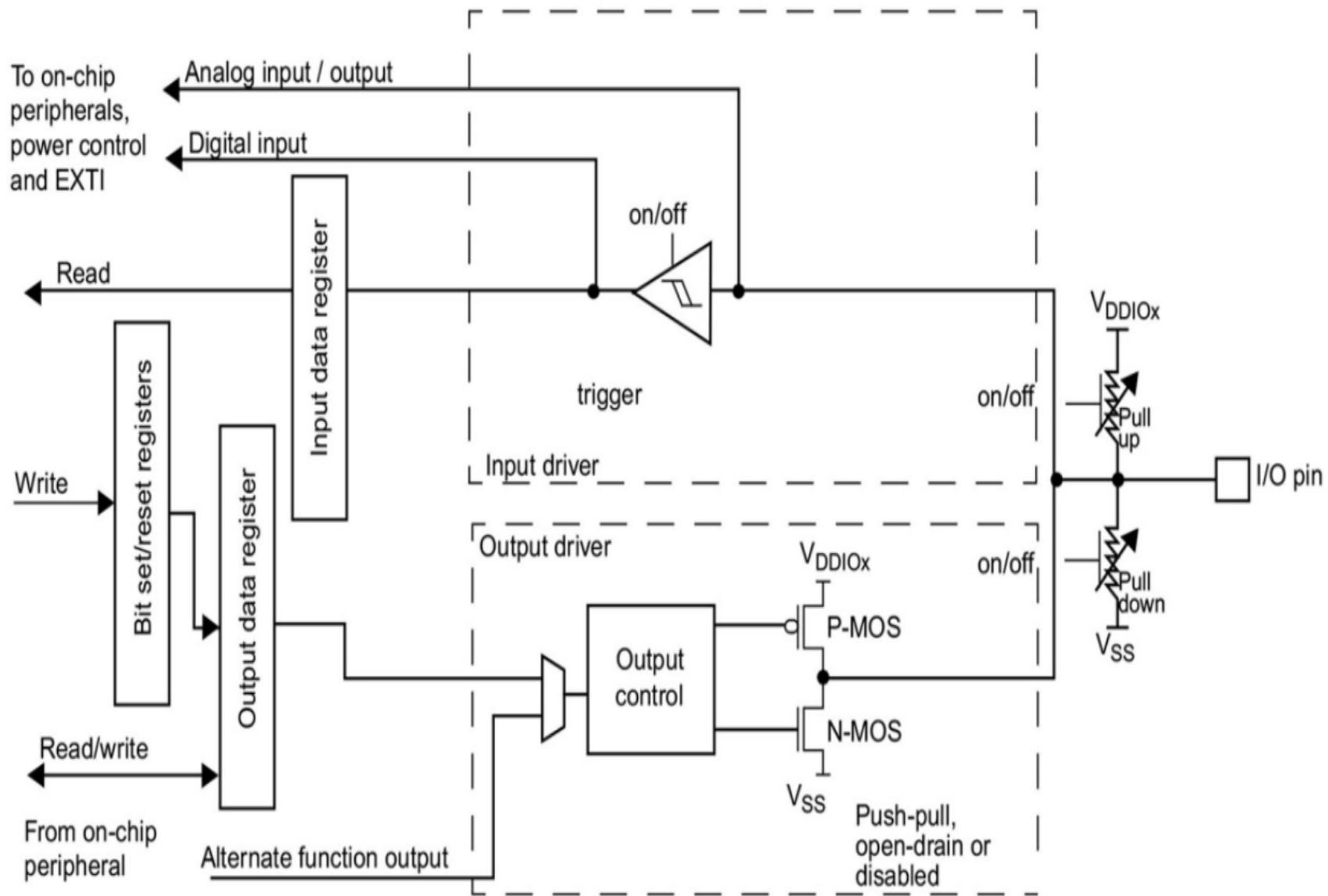
```
    GPIOA->ODR &= ~(1<<5);
```

```
    HAL_Delay(500);
```

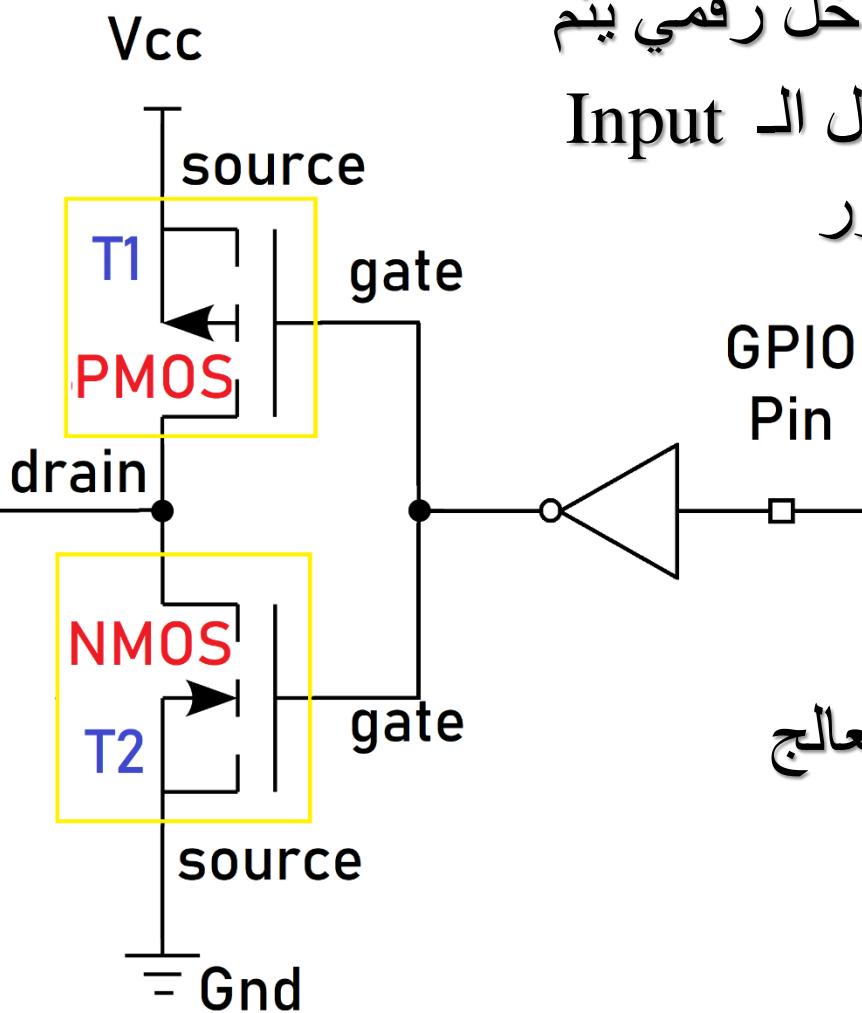
```
}
```

```
}
```

# برمجة أقطاب الدخل في متحكمات stm32



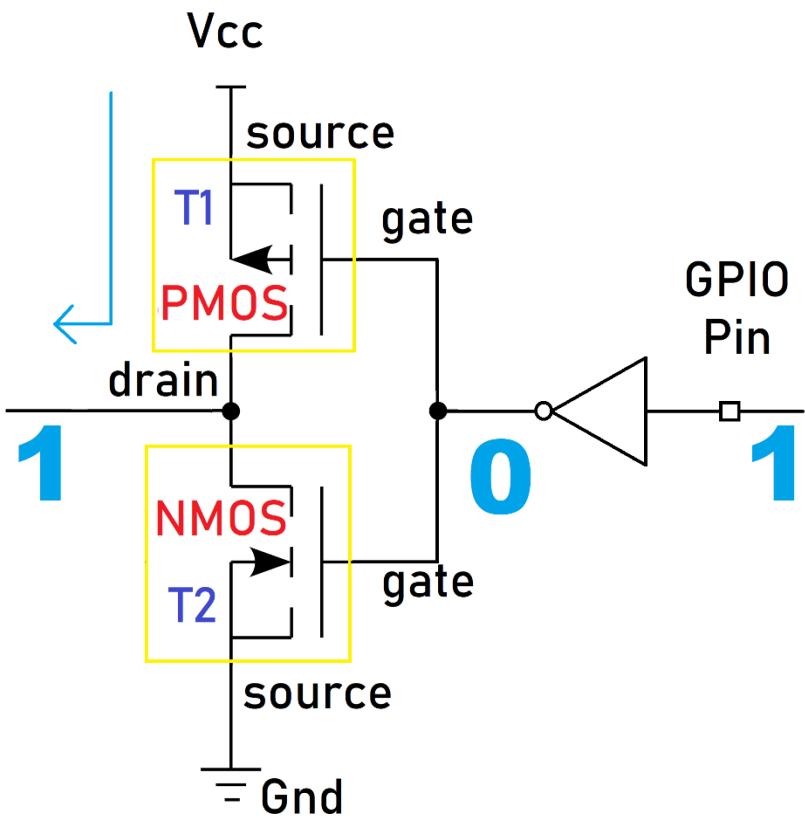
# برمجة أقطاب الدخل في متحكمات stm32



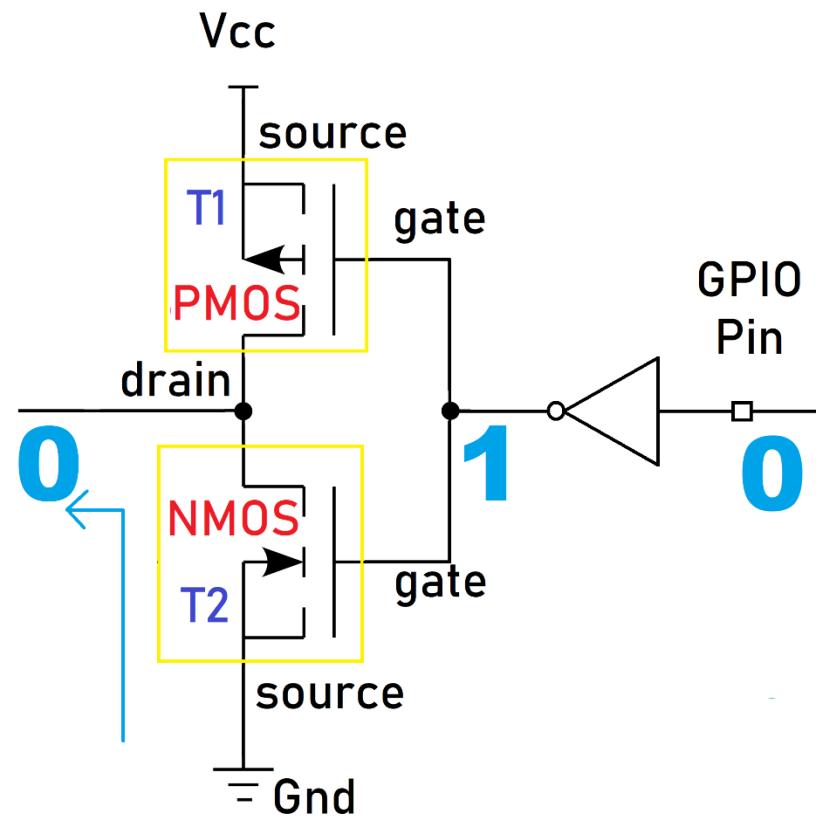
عند استخدام قطب المتحكم كقطب دخل رقمي يتم إلغاء تفعيل Input buffer وتفعيل الـ Output buffer والذي يتكون من ترانزستور من نوع NMOS وترانزستور من نوع PMOS حيث تتصل بوابة الترانزستور مع قطب الـ GPIO و قطب المصرف للترانزستور متصل بالمعالج

## Input Buffer

# برمجة أقطاب الدخل في متحكمات stm32

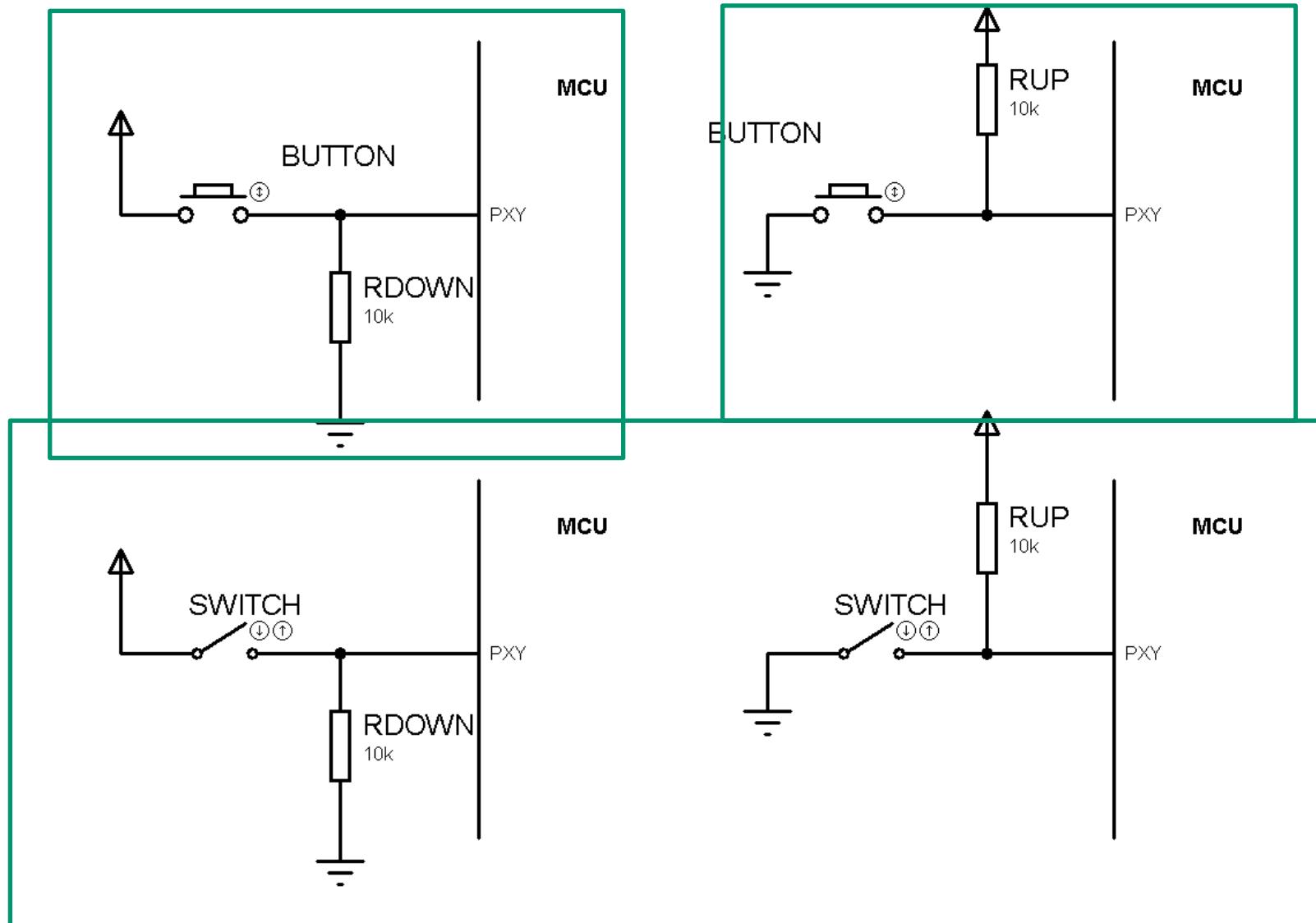


**Input Buffer** reads 1



**Input Buffer** reads 0

# ربط المفاتيح وكبسات اللحظية مع مداخل المتحكم



# الأنماط المختلفة لقطب الدخل الرقمي

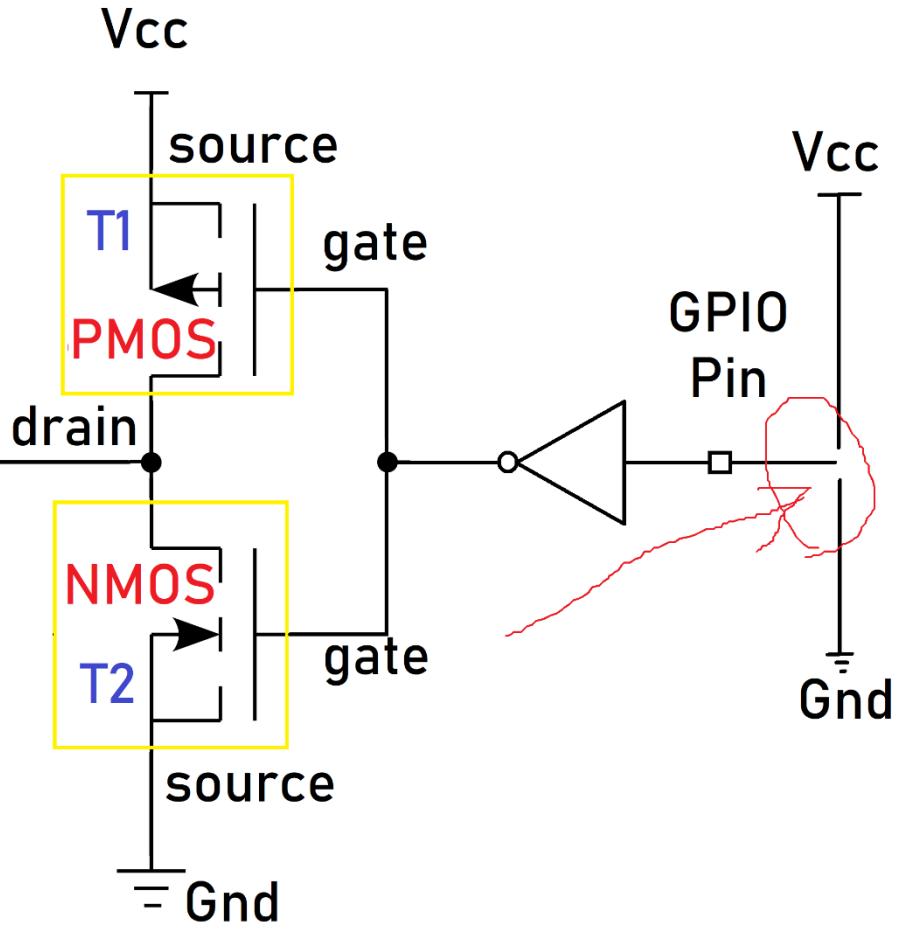
نميز ثلاث أنماط مختلفة لقطب الدخل الرقمي للمتحكم :

**High impedance of Floating**  نمط الـ

**PULL-Up**  نمط الـ

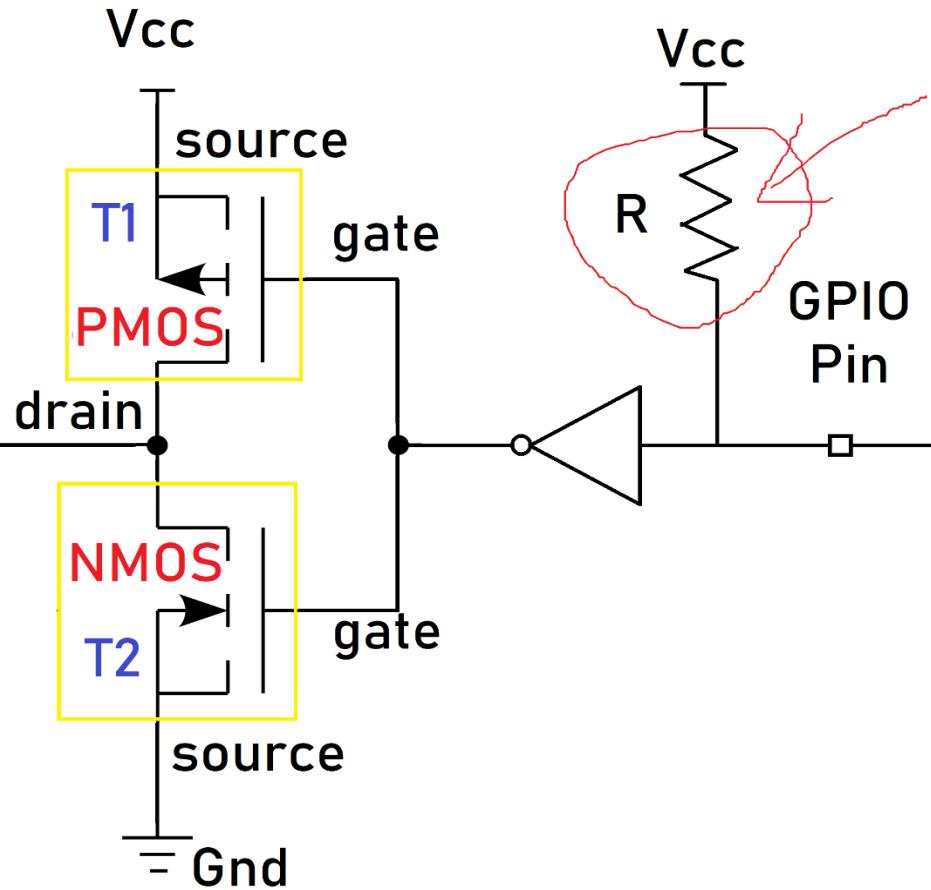
**Pull-Down**  نمط الـ

## نمط High impedance of Floating



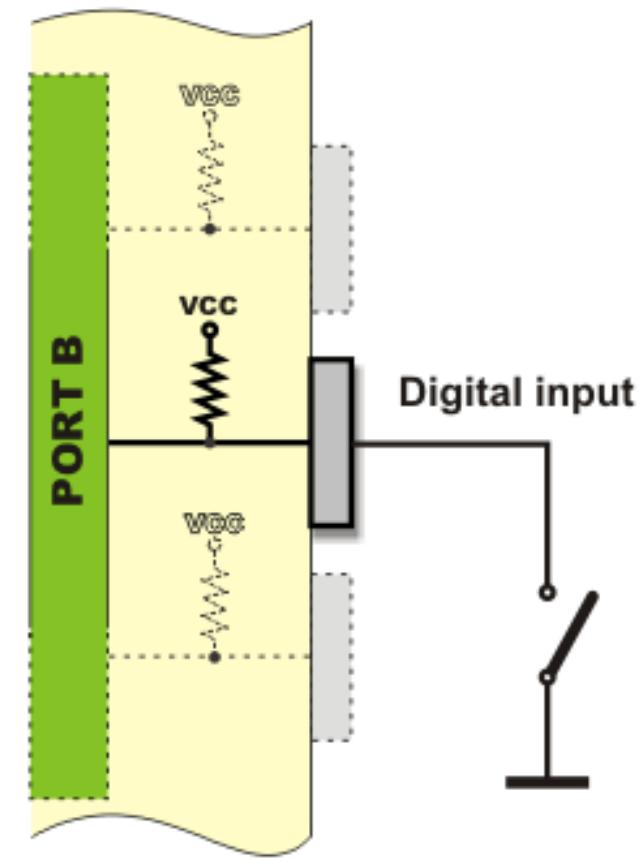
# Input Buffer

## نط PULL-UP □

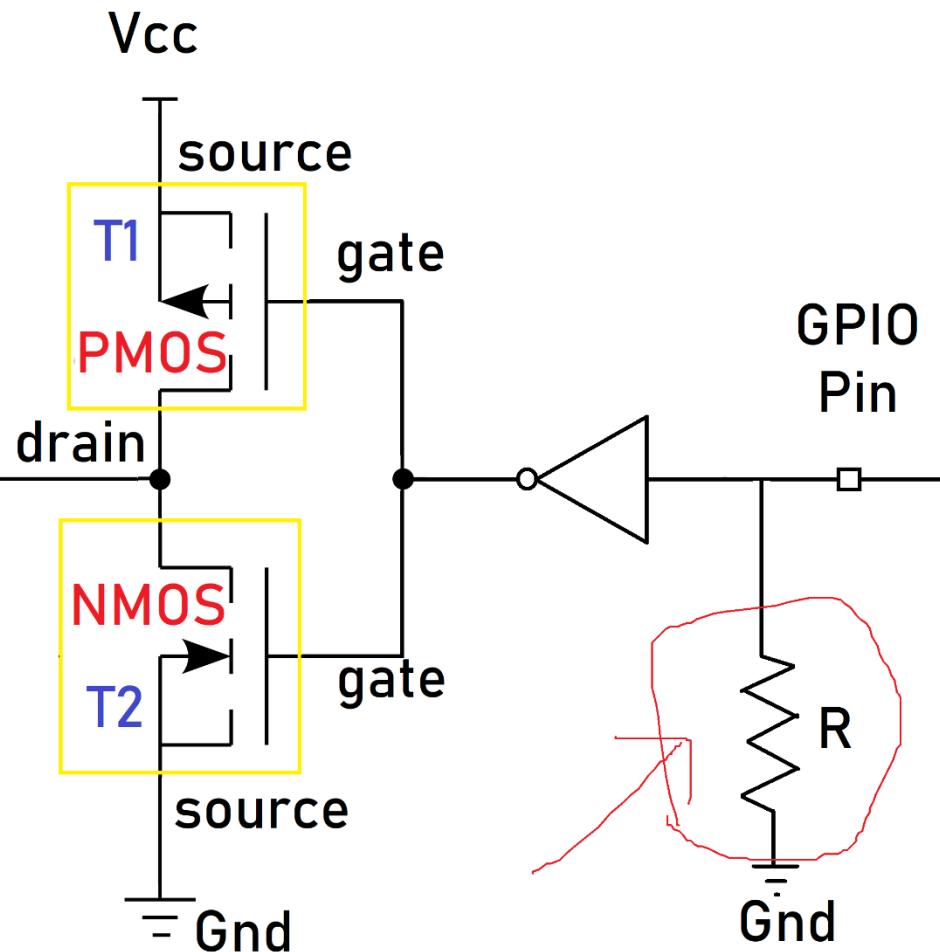


## Input Buffer

Pin with pull-up resistor



## نمط PULL-DOWN



## Input Buffer

# برمجة أقطاب الدخل في متحكمات stm32

**Table 20. Port bit configuration table**

Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01		0 or 1
	Open-drain		1			0 or 1
Alternate Function output	Push-pull	1	0	11	see <a href="#">Table 21</a>	Don't care
	Open-drain		1			Don't care
Input	Analog	0	0	00		Don't care
	Input floating		1			Don't care
	Input pull-down	1	0			0
	Input pull-up					1

# التابع المستخدمة من مكتبة HAL للتحكم بالمخارج الرقمية في متحكم STM32

- نستخدم التابع التالي لمعرفة حالة الدخل الرقمي على أحد أقطاب المتحكم

```
HAL_GPIO_ReadPin(GPIO_TypeDef* GPIOx, uint16_t  
GPIO_Pin);
```

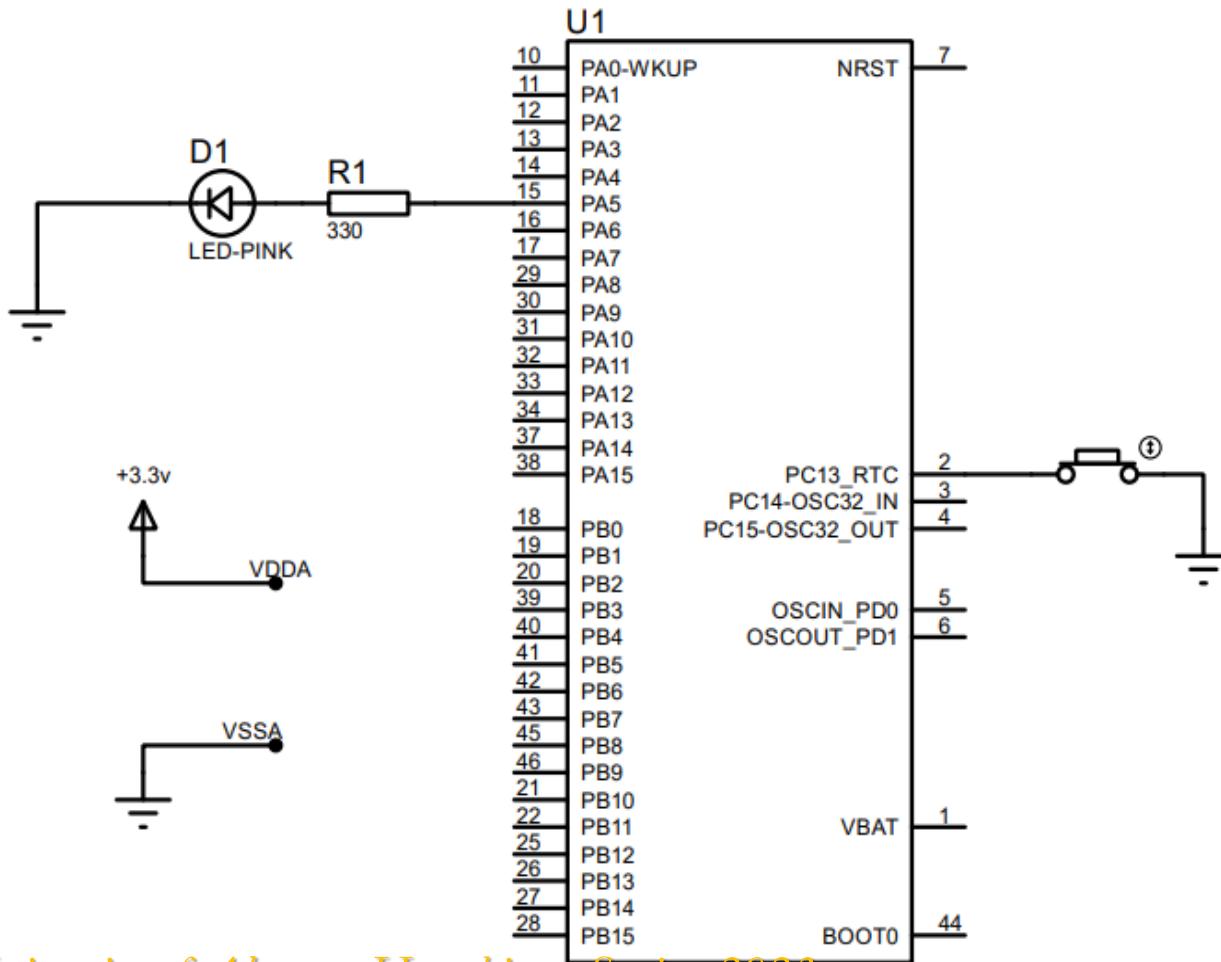
حيث يعيد هذا التابع `GPIO_PIN_RESET` في حال كانت الحالة المنطقية للقطب في حالة جهد منخفض ، ويعد `GPIO_PIN_SET` في حال كانت الحالة المنطقية للقطب في حالة جهد مرتفع.

مثال: لقراءة حالة الدخل الرقمي على القطب رقم 13 من المنفذ E نستخدم التابع التالي:

```
HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_13);
```

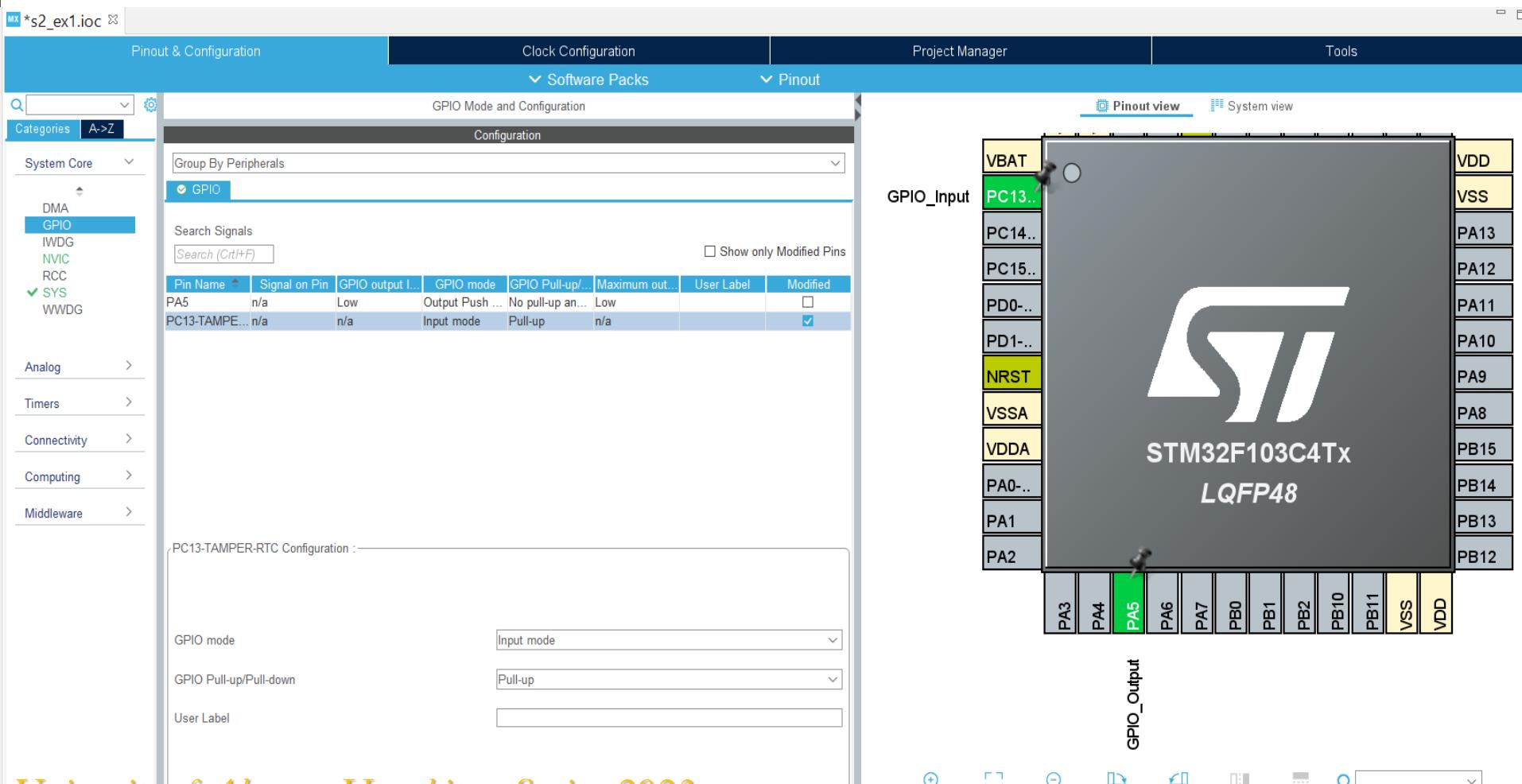
# التطبيق الأول: إضاءة ليد من خلال مفتاح لحظي باستخدام متحكمات HAL و مكتبة STM32

□ بناء تطبيق لإضاءة ليد من خلال مفتاح لحظي باستخدام متحكمات HAL و مكتبة stm32



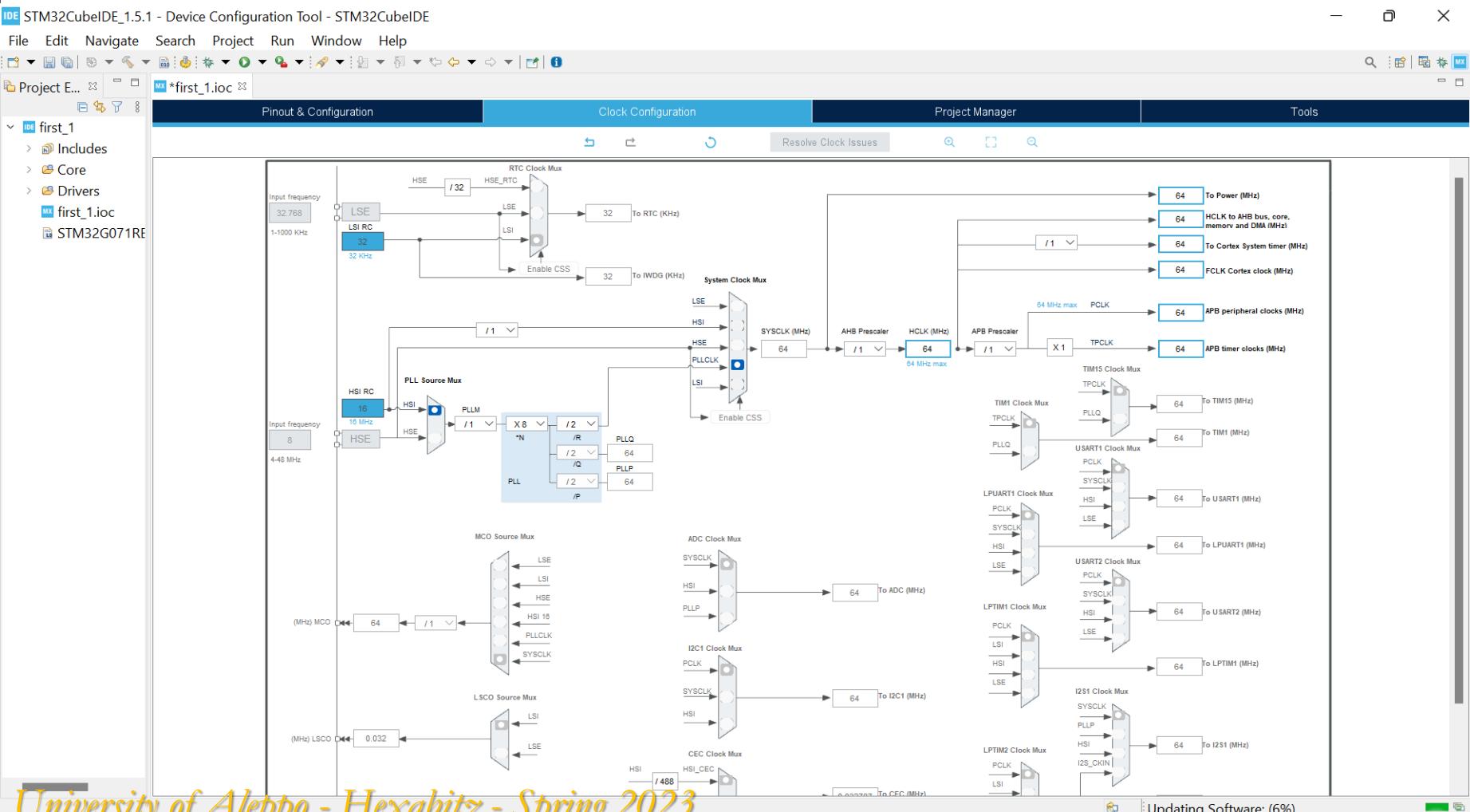
# التطبيق الأول: إضاءة لمبة وإطفاؤه كل 0.5 sec باستخدام متحكمات HAL ومتيبة STM32

نقوم بضبط إعدادات القطب خرج PA5 القطب والقطب PC13  
قطب دخل



# التطبيق الأول: إضاءة ليد وإطفاؤه كل sec 0.5 باستخدام متحكمات HAL ومتيبة STM32

نقوم بضبط تردد الساعة للمتحكم



# التطبيق الأول: إضاءة لمبة وإطفاؤه كل 0.5 sec باستخدام متحكمات HAL و مكتبة STM32

نقوم بالضغط على **Ctrl+s** أو من **Project...Generate**، ليتم حفظ المشروع وتوليد الكود ثم نقوم بإضافة الجزء التالي:

```
#include "main.h"

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
```

# التطبيق الأول: إضاءة ليد وإطفاؤه كل 0.5 sec باستخدام متحكمات HAL و مكتبة STM32

```
while (1){  
if(HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13)==  
GPIO_PIN_RESET)  
{  
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5,  
GPIO_PIN_SET);  
}  
else  
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5,  
GPIO_PIN_RESET);  
}  
}
```

# للتحكم بالمداخل الرقمية للمتحكم STM32 دون استخدام مكتبة HAL:

□ هناك مجموعة من المسجلات تستخدم للتحكم بالمدخلات الرقمية لمتحكم STM32 سنكتفي فقط بذكر المسجل المسؤول عن قراءة حالة المنفذ أو أحد الأقطاب الموجودة فيه ويدعى Input data register (IDR) له الشكل التالي:

## 7.4.5 GPIO port input data register (GPIOx\_IDR) (x = A..H)

Address offset: 0x10

Reset value: 0x0000 XXXX (where X means undefined)

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
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 IDRy: Port input data (y = 0..15)

These bits are read-only and can be accessed in word mode only. They contain the input value of the corresponding I/O port.

# للحكم بالمخارج الرقمية للمتحكم STM32 دون استخدام مكتبة HAL:

□ والمسجل IDR هو مسجل للقراءة فقط ، البات 16:31 غير مستخدمين، أما البات 0:15 فيعبر كل بت عن حالة القطب المقابل له، ففي حال تفعيل مقاومة الرفع الداخلية للقطب عندها سيكون القطب في حالة HIGH بشكل دائم، وعند ضغط المفتاح الموصول معه سيصبح القطب في حالة LOW، لذا يجب مراقبة حالة القطب بشكل مستمر لحين يصبح القطب في حالة LOW عندها يكون المفتاح الموصول معه مضغوط.

□ فلفحص حالة القطب الأول من المنفذ A نستخدم جملة الشرط التالية:

if (!(GPIOA->IDR &(1<<1))) □

```
#include "main.h"

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

int main(void)
{
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
```

```
while (1)
{
    if (!(GPIOC->IDR &(1<<13)))
    {
        GPIOA->ODR = 1<<5;
    }
    else
        GPIOA->ODR &= ~(1<<5);
}
```

Thank you for listening