

\*\*\*\*\*

## PREMIER PAS VERS LE DEVELOPPEMENT AUTOUR DU STM32

Guetting\_Started

\*\*\*\*\*

## Présentation de la carte Nucleo stm32L476rg

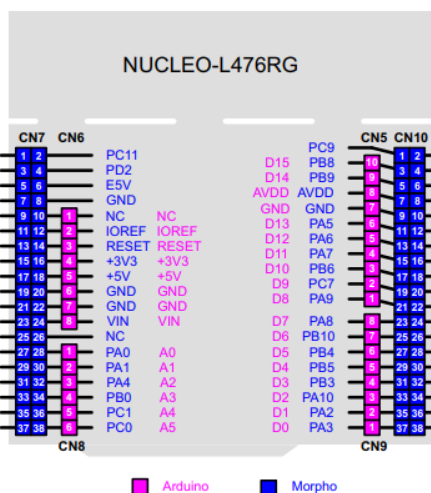
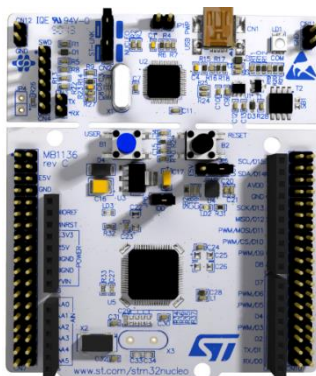
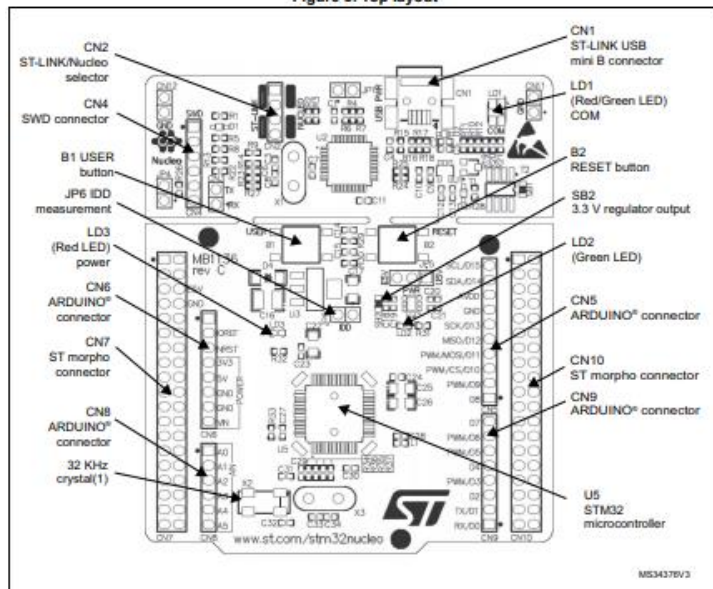


Figure 3. Top layout



### LEDS

La LED LD1 tricolore (vert, orange, rouge) LD1 (COM) renseigne sur ST-LINK état des communications. La couleur par défaut de LD1 est le rouge. LD1 devient vert pour indiquer que communication est en cours entre le PC et le ST-LINK/V2 :

- Clignotement lent rouge/éteint : à la mise sous tension avant l'initialisation USB
- Clignotement rapide rouge/éteint : après la première communication correcte entre le PC et ST-LINK/V2-1 (énumération)
- LED rouge allumée : lorsque l'initialisation entre le PC et ST-LINK/V2-1 est terminée
- LED verte allumée : après une initialisation réussie de la communication cible
- Clignotant Rouge/Vert : pendant la communication avec la cible
- Vert allumé : communication terminée et réussie
- Orange allumé : échec de la communication

La LED Utilisateur LD2 : la LED verte est une LED utilisateur connectée au signal ARDUINO® D13 correspondant vers STM32 I/O PA5 (broche 21) ou PB13 (broche 34) selon la cible STM32. Faire référence à Tableau 11 à Tableau 23 lorsque :

- L'E/S est à la valeur HAUTE, la LED est allumée
- L'I/O est LOW, la LED est éteinte

LD3 PWR : la LED rouge indique que la partie STM32 est alimentée et que l'alimentation +5V est disponible

### PUSH-BUTTON

B1 USER : le bouton utilisateur est connecté à l'I/O PC13 (broche 2) du STM32 microcontrôleur.

B2 RESET : ce bouton poussoir est connecté à NRST, et est utilisé pour le RESET le STM32 microcontrôleur.

Avant de commencer à développer il est primordial d'aller chercher nous même les informations.

Pour se repérer dans le STM32, il nous faut 2 documentations essentielles :

- Le datasheet (organisatin interne du micro, spécifications électrique, etc)  
<https://www.st.com/resource/en/datasheet/stm32l476rg.pdf>
- Le reference manual (fonctionnement détaillé de chaque périphérique et registres associés)  
[https://www.st.com/resource/en/reference\\_manual/dm00083560-stm32l4x5-and-stm32l4x6-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf](https://www.st.com/resource/en/reference_manual/dm00083560-stm32l4x5-and-stm32l4x6-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf)
- Bien sur si l'on développe sur carte nucleo pour nos tests il faut la petite documentation  
<https://www.st.com/en/evaluation-tools/nucleo-l476rg.html#documentation>

La connexion : Nous avons plusieurs choix possibles :

Pour une première connexion il nous faut le driver STLink/V2: stsw-link009\_v2.0.2.

<https://www.st.com/en/development-tools/stsw-link009.html>

La programmation : Nous avons plusieurs choix possibles :

Nous allons choisir Atollic de TrueStudio car il permet de prendre en compte la compilation arm et la possibilité de programmer en C/C++ et d'accéder au registre et d'intégrer une architecture personnalisée.

Nous allons aussi utiliser CubeMX pour comprendre en cas de difficulté de fonctionnement puisqu'il utilise les dernières bibliothèques et permet de déboguer certaines fois, de plus cela permet de compenser les faibles tutos sur Atollic.

Mais pour débiter facilement sans prise de tête nous allons utiliser Keil : MDK-ARM

<https://www.keil.com/demo/eval/arm.htm#!#DOWNLOAD>

Mais pour débiter et gagner en compréhension et se familiariser avec les bibliothèques :

STM32CUBEMX et CUBEIDE

<https://www.st.com/en/development-tools/stm32cubemx.html>

<https://www.st.com/en/development-tools/stm32cubeide.html>

## Premier Test pour la mise en place de l'environnement de travail

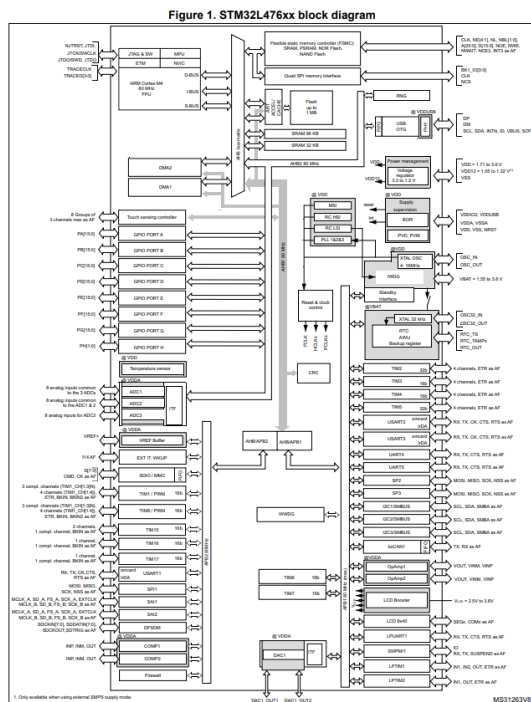
Faire clignoter la Led (LD2) reliée sur le port PA5 de la carte Nucleo.

Il nous faut la documentation

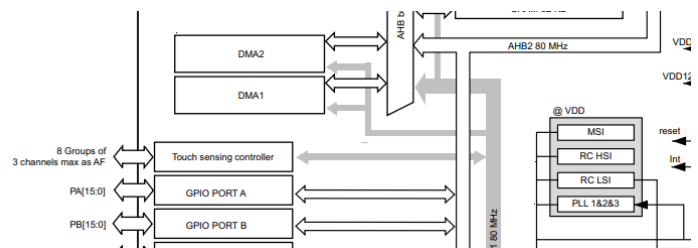
- datasheet [stm32l476rg.pdf]
- reference\_manual (dm00083560-stm32l4x5-and-stm32l4x6-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf) [RM0351]
- Manual-Board carte nucleo-34 board [UM1724]

Sur les microcontrôleurs 8 et 16 bits (MSP430, ATME1, PIC, etc) on configure les périphériques et c'est parti, Mais sur ARM il faut : Activer (AHB2ENR) et régler au besoin l'horloge du périphérique + Activer le registre GPIOA (MODER)+ Activer le port pin (1 ou 0) (ODR).

1- il faut savoir sur quel bus d'horloge est connecté le dit périphérique GPIO\_PORT\_A du port PA5 [datasheet]



Note: AF: alternate function on I/O pins.



Le PortA PA[15:0] est sur le bus AHB2 80MHZ.

Maintenant il faut aller voir dans [reference\_manual] ce que raconte le bus AHB2 ?

C'est un registre classé dans la fonctionnalité RCC (Reset and Clock Control).

D'où le pointeur RCC->AHB2xxx et ce registre AHB2 doit être enable (AHB2ENR) ou reset (AHB2RSTR).

Il est toujours intéressant d'aller lire ce que fait la fonctionnalité RCC.

[reference\_manual]

2- La bus AHB2 est rattaché au registre : manual\_register\_stm32L476xx.pdf

6.4	RCC registers	223
6.4.1	Clock control register (RCC_CR)	223
6.4.2	Internal clock sources calibration register (RCC_ICSCR)	226
6.4.3	Clock configuration register (RCC_CFGR)	227
6.4.4	PLL configuration register (RCC_PLLCFGR)	229
6.4.5	PLLSAI1 configuration register (RCC_PLLSAI1CFGR)	232
6.4.6	PLLSAI2 configuration register (RCC_PLLSAI2CFGR)	235
6.4.7	Clock interrupt enable register (RCC_CIER)	236
6.4.8	Clock interrupt flag register (RCC_CIFR)	238
6.4.9	Clock interrupt clear register (RCC_CICR)	239
6.4.11	AHB2 peripheral reset register (RCC_AHB2RSTR)	242
6.4.12	AHB3 peripheral reset register (RCC_AHB3RSTR)	244
6.4.13	APB1 peripheral reset register 1 (RCC_APB1RSTR1)	244
6.4.14	APB1 peripheral reset register 2 (RCC_APB1RSTR2)	247
6.4.15	APB2 peripheral reset register (RCC_APB2RSTR)	248
6.4.16	AHB1 peripheral clock enable register (RCC_AHB1ENR)	249
6.4.17	AHB2 peripheral clock enable register (RCC_AHB2ENR)	251
6.4.18	AHB3 peripheral clock enable register (RCC_AHB3ENR)	252

## AHB2 peripheral clock enable register (RCC\_AHB2ENR) p251

**RM0351** Reset and clock control (RCC)

### 6.4.17 AHB2 peripheral clock enable register (RCC\_AHB2ENR)

Address offset: 0x4C

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access

**Note:** When the peripheral clock is not active, the peripheral registers read or write access is not supported.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNG EN	HASH EN	AESEN
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	DCME N	ADCCEN	OTGFS EN	Res.	Res.	Res.	GPIOE N	GPIOH EN	GPIOG EN	GPIOF EN	GPIOE EN	GPIOD EN	GPIOC EN	GPIOB EN	GPIOA EN
	FW	FW	FW				FW	FW	FW	FW	FW	FW	FW	FW	FW

Bits 31:19 Reserved, must be kept at reset value.

Bit 18 **RNGEN**: Random Number Generator clock enable

Set and cleared by software.

0: Random Number Generator clock disabled

1: Random Number Generator clock enabled

Bit 1 **GPIOBEN**: IO port B clock enable

Set and cleared by software.

0: IO port B clock disabled

1: IO port B clock enabled

Bit 0 **GPIOAEN**: IO port A clock enable

Set and cleared by software.

0: IO port A clock disabled

1: IO port A clock enabled

On voit qu'il faut mettre le bit 0 du registre à 1.

Pour activer l'horloge du port A (on a l'utilité de chaque bit p252 GPIOAEN).

### 3- Concernant le registre GPIOA (MODER) il faut l'activer

#### 8.5 GPIO registers

For a summary of register bits, register address offsets and reset values, refer to [Table 40](#).  
The peripheral registers can be written in word, half word or byte mode.

##### 8.5.1 GPIO port mode register (GPIOx\_MODER) (x = A to I)

Address offset: 0x00

Reset value: 0xABFF FFFF (for port A)

Reset value: 0xFFFF FEBF (for port B)

On STM32L47x/L48x:

Reset value: 0xFFFF FFFF (for ports C..G)

Reset value: 0x0000 000F (for port H)

On STM32L49x/L4Ax:

Reset value: 0xFFFF FFFF (for ports C..H)

Reset value: 0x00FF FFFF (for port I)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Bits 31:0 **MODE[15:0][1:0]**: Port x configuration I/O pin y (y = 15 to 0)

These bits are written by software to configure the I/O mode.

00: Input mode

01: General purpose output mode

10: Alternate function mode

11: Analog mode (reset state)

Concernant le registre 32 bits MODER de la fonctionnalité GPIOA il s'agit de mettre le bit 10 à 1.

#### 4- Activer la sortie PA5 (Activer le port pin (1 ou 0) (ODR).

##### 8.5.6 GPIO port output data register (GPIOx\_ODR) (x = A to I)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

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

Bits 15:0 **OD[15:0]**: Port output data I/O pin y (y = 15 to 0)

These bits can be read and written by software.

Note: For atomic bit set/reset, the OD bits can be individually set and/or reset by writing to the **GPIOx\_BSRR** register (x = A..FA to H).

Et enfin on met à 1 ou 0 le bit de pA5 via le registre ODR de la fonctionnalité GPIOA

Si tu te demandes comment connaître les registres nécessaires à telle ou telle fonctionnalités, c'est écrit dans le Reference Manual, regarde le chapitre 8 General-purpose I/Os (GPIO) tout est écrit.

#### 8.5.7 GPIO port bit set/reset register (GPIOx\_BSRR) (x = A to I)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:16 **BR[15:0]**: Port x reset I/O pin y (y = 15 to 0)

These bits are write-only. A read to these bits returns the value 0x0000.

0: No action on the corresponding ODx bit

1: Resets the corresponding ODx bit

*Note: If both BSx and BRx are set, BSx has priority.*

Bits 15:0 **BS[15:0]**: Port x set I/O pin y (y = 15 to 0)

These bits are write-only. A read to these bits returns the value 0x0000.

0: No action on the corresponding ODx bit

1: Sets the corresponding ODx bit

Donc le programme est

```
#include "stm32l4xx.h"

int main(void)
{
    volatile unsigned int i =0;
    RCC -> AHB2ENR|= 1<<0;
    //GPIOA->MODER |= 1 << 10;//erreur
    GPIOA->MODER &= 0xFFFFF7FF;
    while(1)
    {
        for (i= 0; i < 50000; i++)
            GPIOA -> ODR= (1<<5); //on affecte 1 au bit du rang 5
            //GPIOA -> BSRR= (1<<5);

        for (i = 0; i < 50000; i++)
            GPIOA -> ODR= (0<<5); //PA5 à 0  ///GPIOA -> BSRR= (1<<21);}
    }
}
```

Nous avons compris que :

[Il faut Activer l'horloge et si besoin la régler.](#)

Le PortA PA[15 :0] est sur le bus AHB2 80MHZ. [datasheet] organisation interne du stm32.

AHB2 est un registre classé dans la fonctionnalité RCC (Reset and Clock Control), d'où le pointeur RCC->AHB2xxx et ce registre AHB2 doit être enable (AHB2ENR)

RM0351

Reset and clock control (RCC)

#### 6.4.17 AHB2 peripheral clock enable register (RCC\_AHB2ENR)

Address offset: 0x4C

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access

Note: When the peripheral clock is not active, the peripheral registers read or write access is not supported.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RNG EN	HASH EN	AESEN
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	DCMI EN	ADCEN	OTGFS EN	Res.	Res.	Res.	GPIOE EN	GPIOH EN	GPIOG EN	GPIOF EN	GPIOE EN	GPIOD EN	GPIOC EN	GPIOB EN	GPIOA EN
	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit 0 GPIOAEN: IO port A clock enable  
Set and cleared by software.  
0: IO port A clock disabled  
1: IO port A clock enabled

RCC -> AHB2ENR |= 1 << 0;

...

On met à 1 le bit 0 du registre AHB2ENR

0000 0000 0000 0000 0000 0000 0000 0000

OU 0000 0000 0000 0000 0000 0000 0000 0001

= 0000 0000 0000 0000 0000 0000 0000 0001

## Il faut activer le port GPIOA avec MODER

### 8.5 GPIO registers

For a summary of register bits, register address offsets and reset values, refer to [Table 40](#).  
The peripheral registers can be written in word, half word or byte mode.

#### 8.5.1 GPIO port mode register (GPIOx\_MODER) (x = A to I)

Address offset: 0x00

Reset value: 0xABFF FFFF (for port A)

Reset value: 0xFFFF FEBF (for port B)

On STM32L47x/L48x:

Reset value: 0xFFFF FFFF (for ports C..G)

Reset value: 0x0000 000F (for port H)

On STM32L49x/L4Ax:

Reset value: 0xFFFF FFFF (for ports C..H)

Reset value: 0x00FF FFFF (for port I)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Bits 31:0 **MODE[15:0][1:0]**: Port x configuration I/O pin y (y = 15 to 0)

These bits are written by software to configure the I/O mode.

00: Input mode

01: General purpose output mode

10: Alternate function mode

11: Analog mode (reset state)

Table 40. GPIO register map and reset values

Offset	Register name	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
0x00	GPIOA_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]		MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
	Reset value	1	0	1	0	1	0	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

Concernant le registre 32 bits MODER de la fonctionnalité GPIOA 5 il s'agit de mettre MODE5[01].

```
GPIOA->MODER &= 0xFFFF7FF; //GPIOA->MODER |= 1 << 10; //erreur
```

Reset Value : 0XABFF FFFF

1010 1011 1111 1111 1111 1111 1111 1111 : 0XABFF FFFF

OU 0000 0000 0000 0000 0000 0100 0000 0000

1010 1011 1111 1111 1111 1111 1111 1111 = GPIOA->MODER |= 1 << 10 **ERREUR BIT 10 à 1**

Ici tout est à 1 en entrée, nous on veut MODE5[0:1] output mode5 [01 : General purpose output mode]

Soit le bit 10 à 1 et le bit 11 à 0 pour que PA5 soit une sortie

1010 1011 1111 1111 1111 1111 1111 1111 : 0XABFF FFFF

& 1111 1111 1111 1111 1111 0111 1111 1111

1010 1011 1111 1111 1111 0111 1111 1111 = GPIOA->MODER &= 0xFFFF7FF

GPIOA->MODER &= 0<<11

Voici pourquoi il y a l'erreur :

Pour une question d'écriture et de préférence :

GPIOA->ODR |= (1 << 5); /\* mise à 1 du bit n°5 \*/ moi je préfère

GPIOA->ODR |= 0x20; /\* mise à 1 du bit n°5 \*/ ce qui revient exactement au même.

OU 0000 0000

0x20 = 0010 0000 = 0010 0000

GPIOA->ODR &= ~(1 << 5); /\* mise à 0 du bit n°5 \*/ moi je préfère

GPIOA->ODR &= 0xDF; /\* mise à 0 du bit n°5 \*/ ce qui revient exactement au même.

ET 1111 1111

0xDF = 1101 1111 = 1101 1111



## Il faut actionner la sortie

### 8.5.6 GPIO port output data register (GPIOx\_ODR) (x = A to I)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OD[31]	OD[30]	OD[29]	OD[28]	OD[27]	OD[26]	OD[25]	OD[24]	OD[23]	OD[22]	OD[21]	OD[20]	OD[19]	OD[18]	OD[17]	OD[16]
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD[15]	OD[14]	OD[13]	OD[12]	OD[11]	OD[10]	OD[9]	OD[8]	OD[7]	OD[6]	OD[5]	OD[4]	OD[3]	OD[2]	OD[1]	OD[0]
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

Bits 15:0 **OD[15:0]**: Port output data I/O pin y (y = 15 to 0)

These bits can be read and written by software.

Note: For atomic bit set/reset, the OD bits can be individually set and/or reset by writing to the **GPIOx\_BSRR** register (x = A..FA to H).

Et enfin on met à 1 ou 0 le bit de pA5 via le registre ODR de la fonctionnalité GPIOA

GPIOA -> ODR= (0<<5)                      GPIOA -> ODR= (1<<5)

The screenshot shows the STM32CubeIDE interface. The top bar indicates the project is 'stm32l475'. The left sidebar shows the 'Packs' view for 'Device: STM32L475'. The main window is divided into two panes. The left pane lists installed packs for the selected device, including ARMv7-M, CMSIS, and various HAL and LL packs. The right pane shows the 'Packs' list with columns for Pack, Action, and Description. It lists packs like STM32L475 selected, STM32L475S4 Series Device Support, and various generic and CMSIS packs.

Pack	Action	Description
Device Specific	2 Packs	STM32L475 selected
STM32L475 selected	1 Pack	STM32L475S4 Series Device Support
Generic	71 Packs	STM32L475S4 Series Device Support
ARM-Packs: PC117	Install	GAUSS PC117 Cryptographic Token Interf.
ARM-Packs: Unity	Install	Unit Testing for C (Especially Embedded So
ARMv7-M	Install	Software components for inter processor co
ARMv7-M	Install	CMSIS (Common Microcontroller Software
ARMv7-M	Install	CMSIS Drivers for external devices
ARMv7-M	Install	CMSIS Driver Validation
ARMv7-M	Install	Bundle of KeilTools for Cortex-M and Certe
ARMv7-M	Install	CMSIS-DSP Validation
ARMv7-M	Install	Device Driver for the ARMv7 (Rho)TM/U-N
ARMv7-M	Install	ARM mbed Client for Cortex-M devices
ARMv7-M	Install	ARM mbed Cryptographic library
ARMv7-M	Install	ARM mbed Cryptographic and TLS/SSL lib
ARMv7-M	Install	GNU Scheduler for Cortex-M3/M4
ARMv7-M	Install	PLB (Platform Security Architecture)

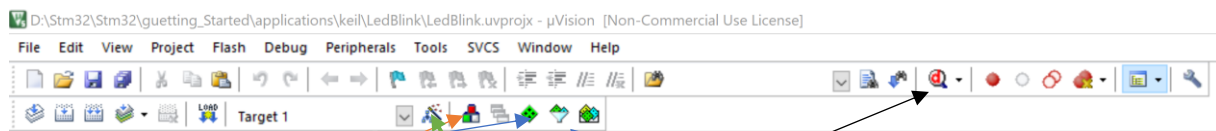
Fermer

 **Manage Run-Time Environment**

Software Component	Sel.	Variant	Version	Description
(+) Board Support		NUCLEO-L476RG	1.0.0	STMicroelectronics NUCLEO-L476RG Development Board
(+) CMSIS				Cortex_Microcontroller Software Interface Components
(+ CORE)	<input checked="" type="checkbox"/>		5.5.0	CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, ARMv8.1-M
(+ DSP)	<input type="checkbox"/>	Source	19.0-dev	CMSIS-DSP Library for Cortex-M, SC000, and SC300
(+ NN Lib)	<input type="checkbox"/>		3.0.0	CMSIS-NN Neural Network Library
(+) RTOS (API)			1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300
(+) RTOS2 (API)			2.1.3	CMSIS-RTOS2 API for Cortex-M, SC000, and SC300
(+) CMSIS Driver				Unified Device Drivers compliant to CMSIS-Driver Specifications
(+) Compiler		ARM Compiler	16.0	Compiler Extensions for ARM Compiler 5 and ARM Compiler 6
(+) Data Exchange				Data exchange or data formatter
(+) Device				Startup, System Setup
(+ Startup	<input checked="" type="checkbox"/>		1.3.1	System Startup for STMicroelectronics STM32L4 Series
(+ STM32Cube Framework (API)			1.1.0	STM32Cube Framework
(+ STM32Cube HAL				STM32L4xx Hardware Abstraction Layer (HAL) Drivers
(+ STM32Cube LL				
(+) File System		MDK-Plus	6.14.4	File Access on various storage devices
(+) Network		MDK-Plus	7.16.0	IPv4 Networking using Ethernet or Serial protocols
(+) USB		MDK-Plus	6.15.4	USB Communication with various device classes

Project: LedBlink

- Target 1
  - Source Group 1
    - CMSIS
    - Device
      - startup\_stm32i476xx.s (Startup)
      - system\_stm32i4xx.c (Startup)



Charger le programme dans la cible

10

Alors on installe stm32CubeMX avec ses nombreux tutos du net on voit dans les détails des registres en mode debug mais aussi cela nous approche d'un peu plus des bibliothèques pour prendre de la hauteur.

Pour se familiariser avec le stm32 avec une interface conviviale : stm32CUBEMX

Téléchargement : <https://www.st.com/en/development-tools/stm32cubemx.html>

<https://www.st.com/en/development-tools/stm32cubeide.html>

Il manque plus que choisir sa carte et c'est partie, nous pouvons générer pour plusieurs plateformes.

## GPIO NUCLEO PIN

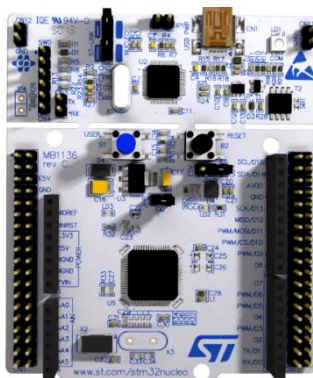
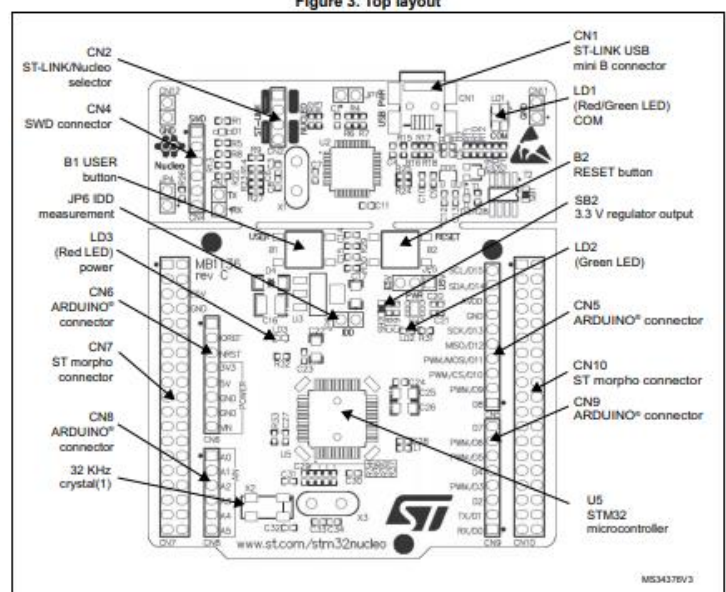
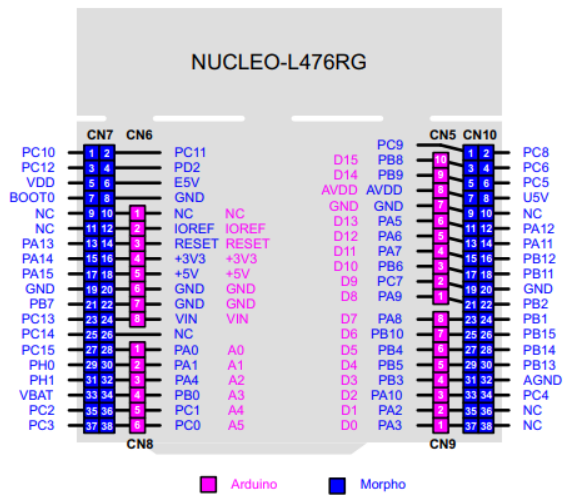


Figure 3. Top layout





**Table 23. ARDUINO® connectors on NUCLEO-L476RG**

Connector	Pin	Pin name	STM32 pin	Function
Left connectors				
CN6 power	1	NC	-	-
	2	IOREF	-	3.3V Ref
	3	RESET	NRST	RESET
	4	+3.3V	-	3.3V input/output
	5	+5V	-	5V output
	6	GND	-	ground
	7	GND	-	ground
	8	VIN	-	Power input
CN8 analog	1	A0	PA0	ADC12_IN5
	2	A1	PA1	ADC12_IN6
	3	A2	PA4	ADC12_IN9
	4	A3	PB0	ADC12_IN15
	5	A4	PC1 or PB9 <sup>(1)</sup>	ADC123_IN2 (PC1) or I2C1_SDA (PB9)
	6	A5	PC0 or PB8 <sup>(1)</sup>	ADC123_IN1 (PC0) or I2C1_SCL (PB8)
Right connectors				
CN5 digital	10	D15	PB8	I2C1_SCL
	9	D14	PB9	I2C1_SDA
	8	AREF	-	AVDD
	7	GND	-	ground
	6	D13	PA5	SPI1_SCK
	5	D12	PA6	SPI1_MISO
	4	D11	PA7	TIM17_CH1 or SPI1_MOSI
	3	D10	PB6	TIM4_CH1 or SPI1_CS

Connector	Pin	Pin name	STM32 pin	Function
CN5 digital	2	D9	PC7	TIM3_CH2
	1	D8	PA9	-
CN9 digital	8	D7	PA8	-
	7	D6	PB10	TIM2_CH3
	6	D5	PB4	TIM3_CH1
	5	D4	PB5	-
	4	D3	PB3	TIM2_CH2
	3	D2	PA10	-
	2	D1	PA2	USART2_TX
	1	D0	PA3	USART2_RX

1. Refer to [Table 10: Solder bridges](#) for details.

