MICROS 32 BITS
STM - GPIO

ROBINSON JIMENEZ MORENO

No es seguro | www2.keil.com/mdk5/uvision/

ones | publish Significado... | Introduction to Con... | bibliograph | Robin | DAVINCI Sla... | New Courses | MIT... | MDK Version 5 | Olguer Morales - Y... | Understanding the...

arm KEIL

Products | Download | Events | Support | Search Keil.com... | Go

Home / MDK / µVision IDE

Learning Platform

µVision® IDE

The µVision IDE combines project management, run-time environment, build facilities, source code editing, and program debugging in a single powerful environment. µVision is easy-to-use and accelerates your embedded software development. µVision supports multiple screens and allows you to create individual window layouts anywhere on the visual surface.

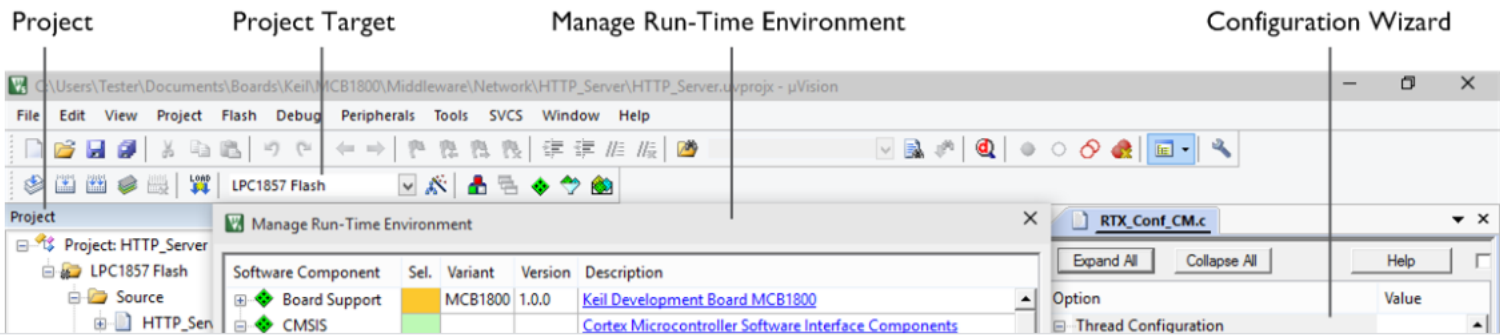
The µVision Debugger provides a single environment in which you may test, verify, and optimize your application code. The debugger includes traditional features like simple and complex breakpoints, watch windows, and execution control and provides full visibility to device peripherals.

Quick Links

- µVision Debugger
- µVision User's Guide
- Online manuals for MDK
- Verification Tools

µVision Project Manager and Run-Time Environment

With the µVision Project Manager and Run-Time Environment you create software application using pre-build software components and device support from Software Packs. The software components contain libraries, source modules, configuration files, source code templates, and documentation. Software components can be generic to support a wide range of devices and applications.



ENTORNO DE PROGRAMACION KEIL

Abrir proyecto

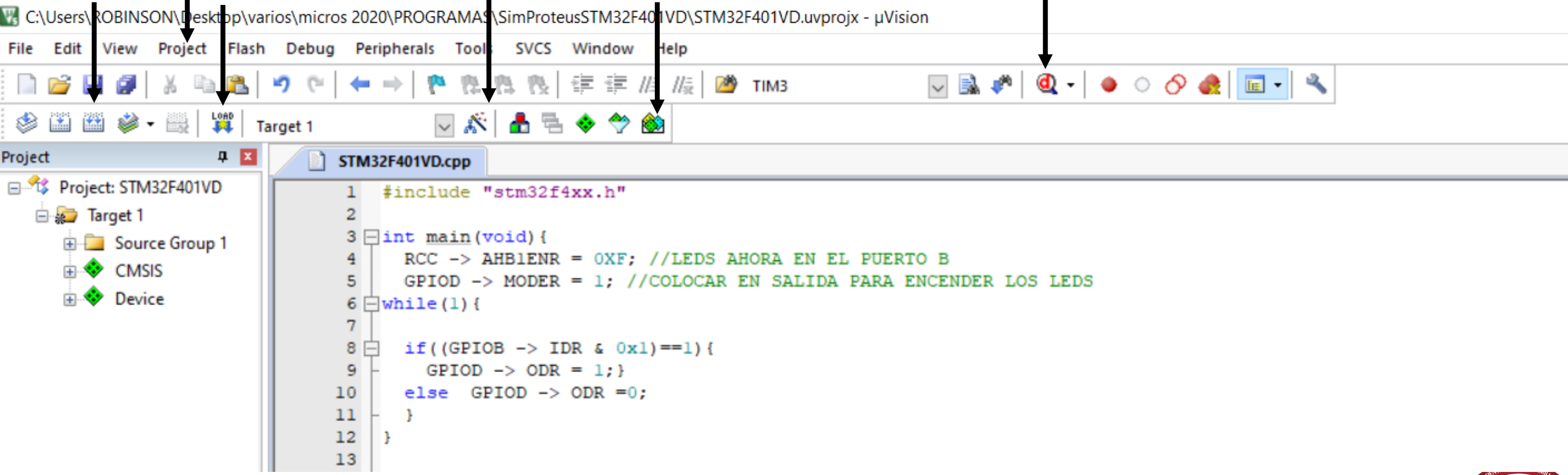
Cuadros de dialogo para programación en la tarjeta

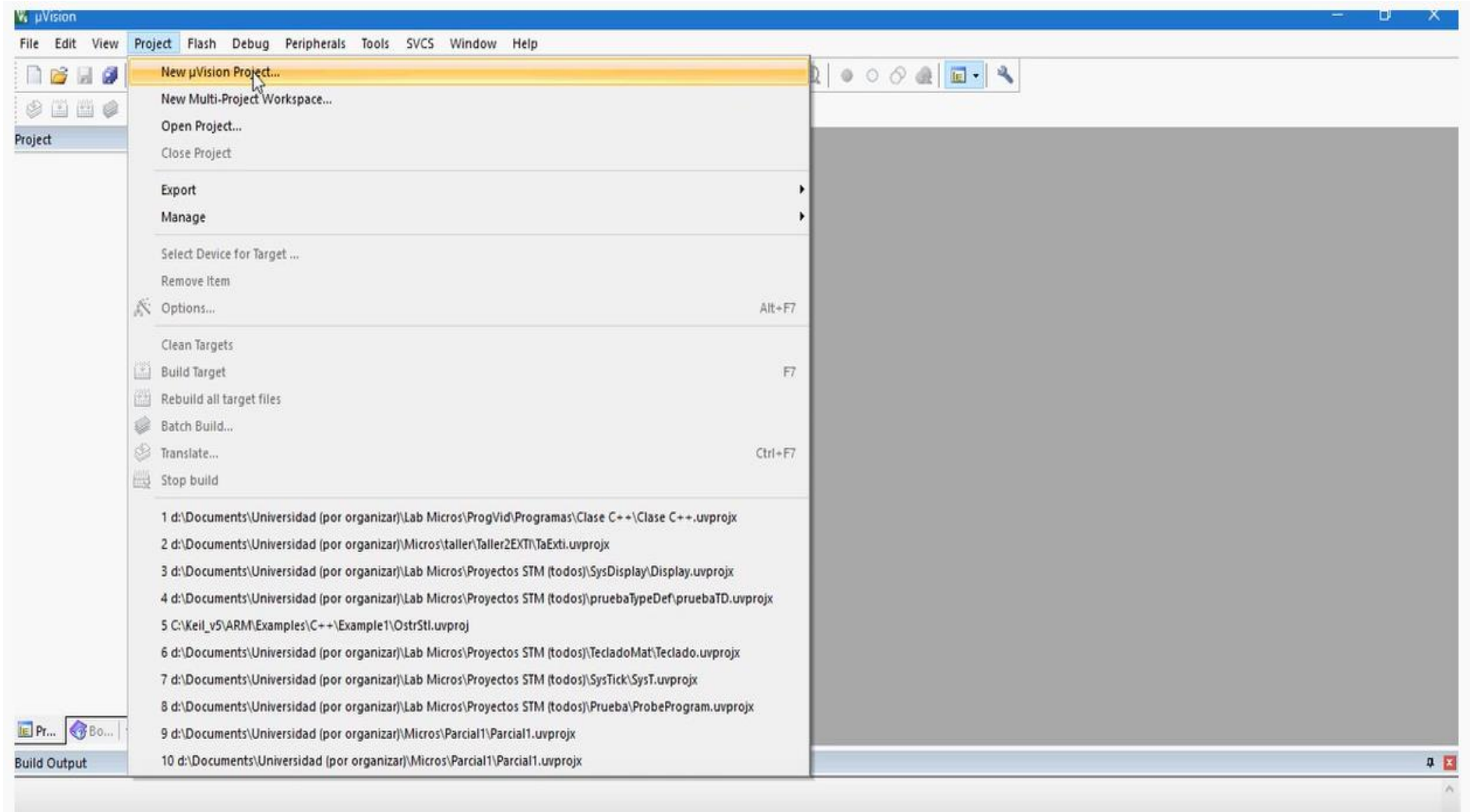
Activación/desactivación simulación de código

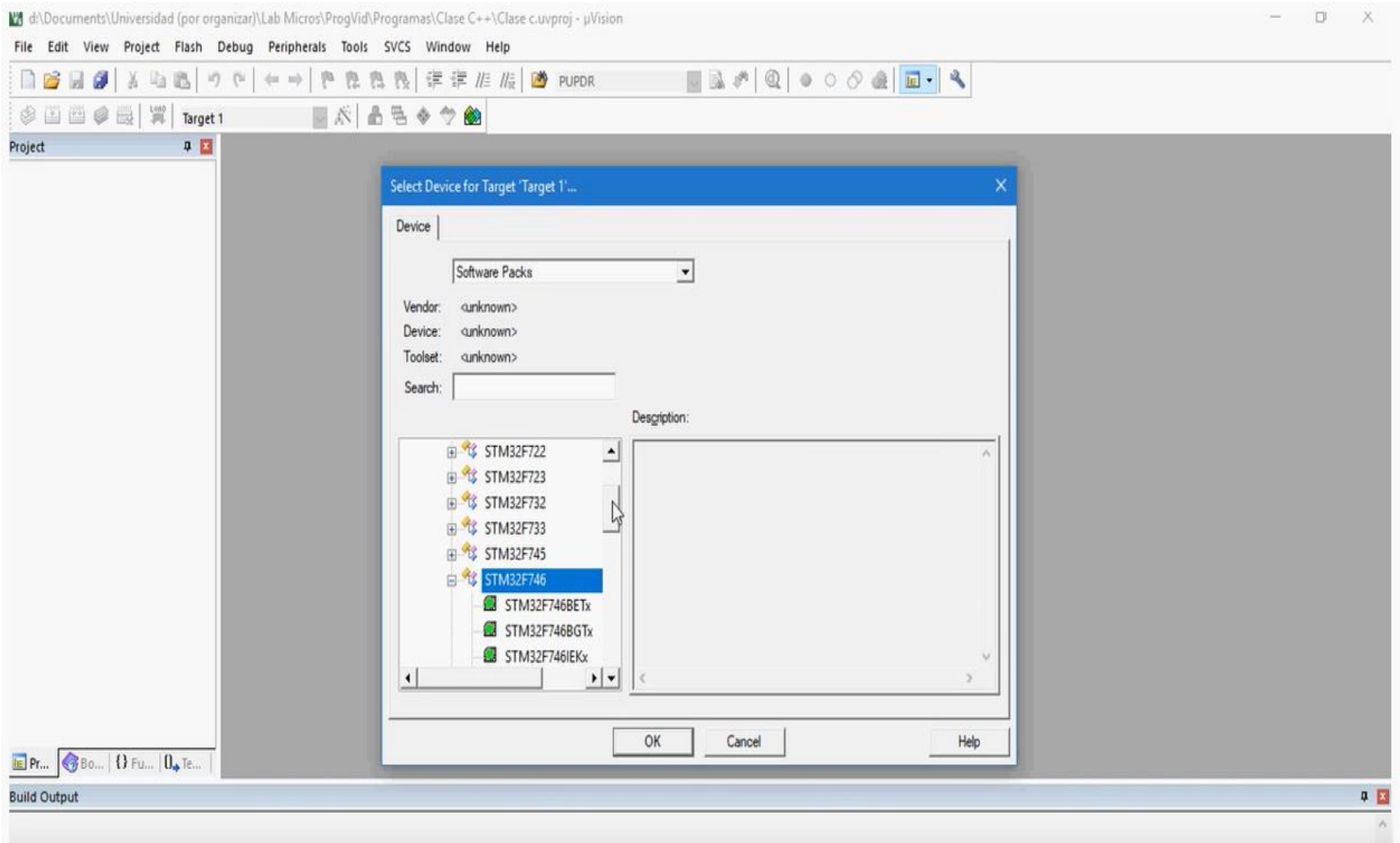
Compilar

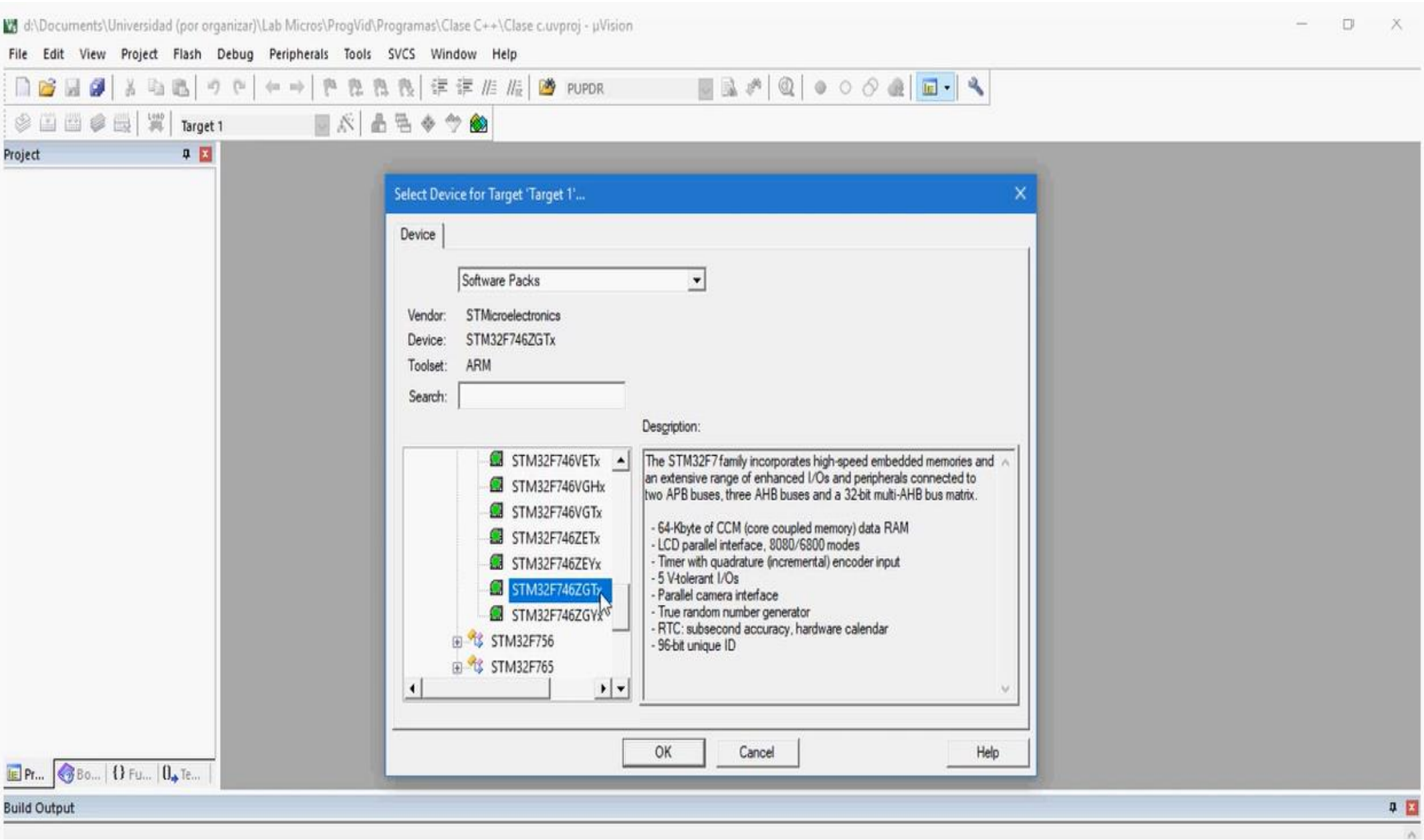
Programar tarjeta STM

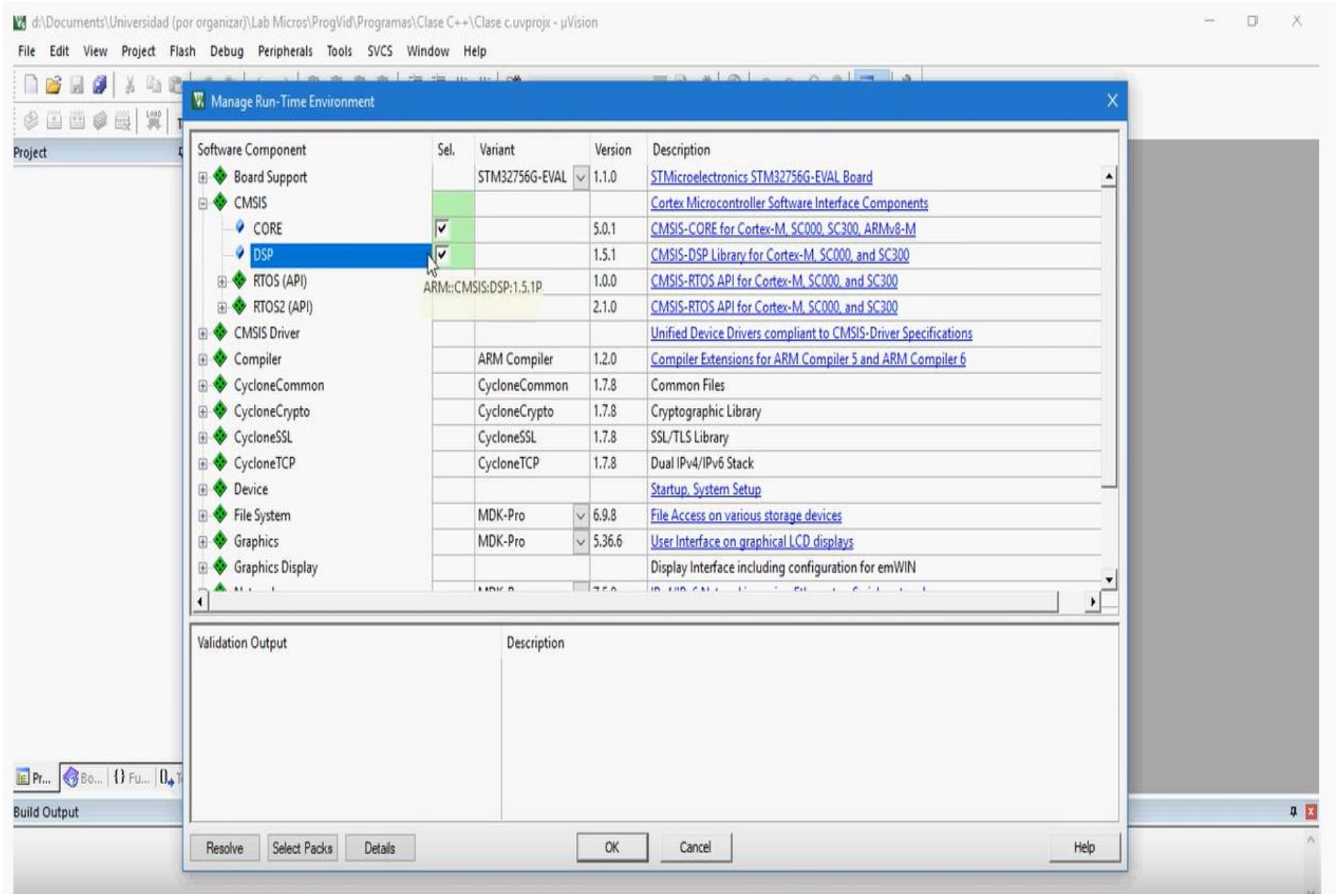
Instalador de paquetes stm32 F7 y stm32f4

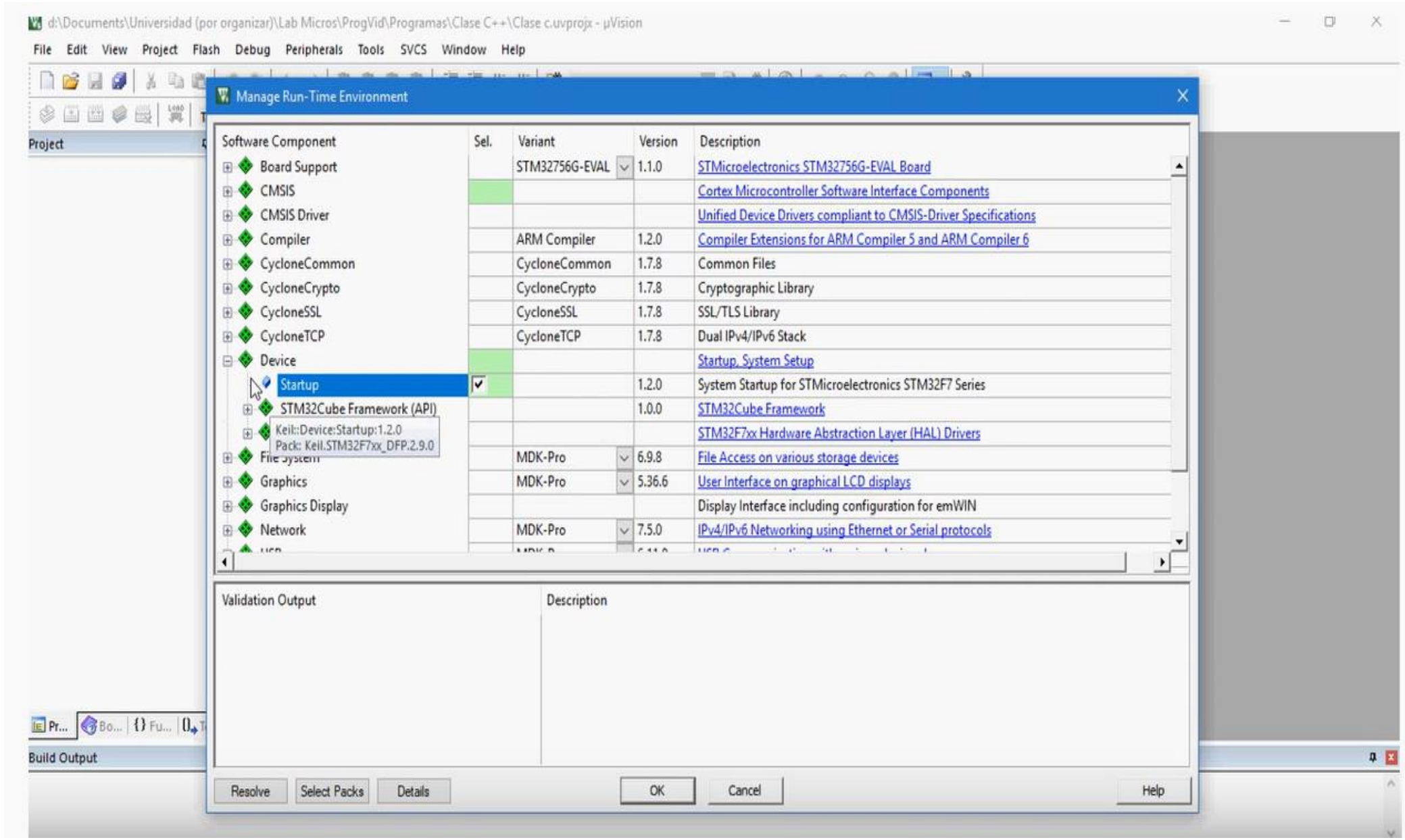



µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
New µVision Project...
New Multi-Project Workspace...
Open Project...
Close Project
Export
Manage
Select Device for Target ...
Remove Item
Options... Alt+F7
Clean Targets
Build Target F7
Rebuild all target files
Batch Build...
Translate... Ctrl+F7
Stop build
1 d:\Documents\Universidad (por organizar)\Lab Micros\ProgVid\Programas\Clase C++\Clase C++.uvprojx
2 d:\Documents\Universidad (por organizar)\Micros\taller\Taller2EXTI\TaExti.uvprojx
3 d:\Documents\Universidad (por organizar)\Lab Micros\Proyectos STM (todos)\SysDisplay\Display.uvprojx
4 d:\Documents\Universidad (por organizar)\Lab Micros\Proyectos STM (todos)\pruebaTypeDef\pruebaTD.uvprojx
5 C:\Keil_v5\ARM\Examples\C++\Example1\OstrStl.uvproj
6 d:\Documents\Universidad (por organizar)\Lab Micros\Proyectos STM (todos)\TecladoMat\Teclado.uvprojx
7 d:\Documents\Universidad (por organizar)\Lab Micros\Proyectos STM (todos)\SysTick\SysT.uvprojx
8 d:\Documents\Universidad (por organizar)\Lab Micros\Proyectos STM (todos)\Prueba\ProbeProgram.uvprojx
9 d:\Documents\Universidad (por organizar)\Micros\Parcial1\Parcial1.uvprojx
10 d:\Documents\Universidad (por organizar)\Micros\Parcial1\Parcial1.uvprojx
Project
Build Output


d:\Documents\Universidad (por organizar)\Lab Micros\ProgVid\Programas\Clase C++\Clase c.uvproj - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
PUPDR
Target 1
Project
Select Device for Target 'Target 1'...
Device
Software Packs
Vendor: <unknown>
Device: <unknown>
Toolset: <unknown>
Search:
Description:
STM32F722
STM32F723
STM32F732
STM32F733
STM32F745
STM32F746
STM32F746BETx
STM32F746BGTx
STM32F746IEKx
OK
Cancel
Help
Build Output

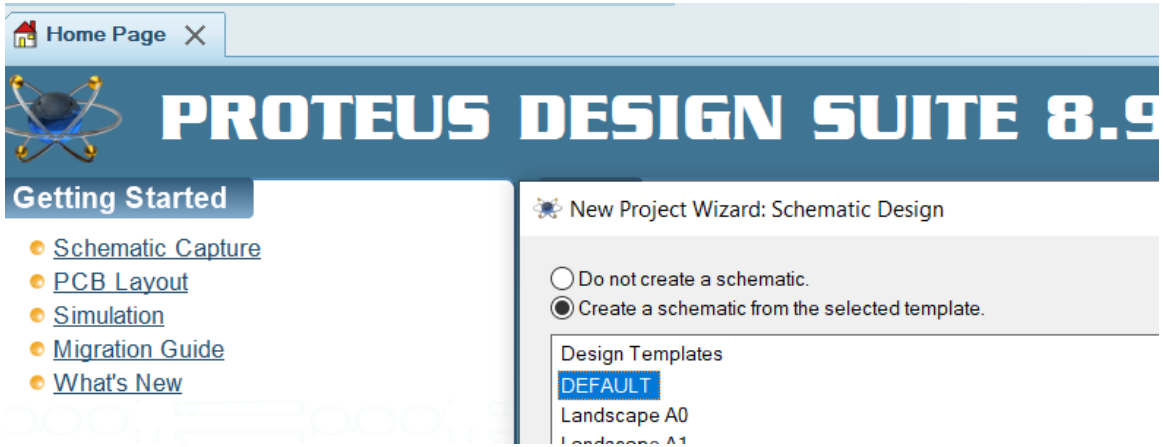

d:\Documents\Universidad (por organizar)\Lab Micros\ProgVid\Programas\Clase C++\Clase c.uvproj - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
PUPDR
Target 1
Project
Select Device for Target 'Target 1'...
Device
Software Packs
Vendor: STMicroelectronics
Device: STM32F746ZGTx
Toolset: ARM
Search:
Description:
STM32F746VETx
STM32F746VGHx
STM32F746VGTx
STM32F746ZETx
STM32F746ZEYx
STM32F746ZGTx
STM32F746ZGYx
STM32F756
STM32F765
The STM32F7 family incorporates high-speed embedded memories and an extensive range of enhanced I/Os and peripherals connected to two APB buses, three AHB buses and a 32-bit multi-AHB bus matrix.
- 64-Kbyte of CCM (core coupled memory) data RAM
- LCD parallel interface, 8080/6800 modes
- Timer with quadrature (incremental) encoder input
- 5 V-tolerant I/Os
- Parallel camera interface
- True random number generator
- RTC: subsecond accuracy, hardware calendar
- 96-bit unique ID
OK
Cancel
Help
Build Output

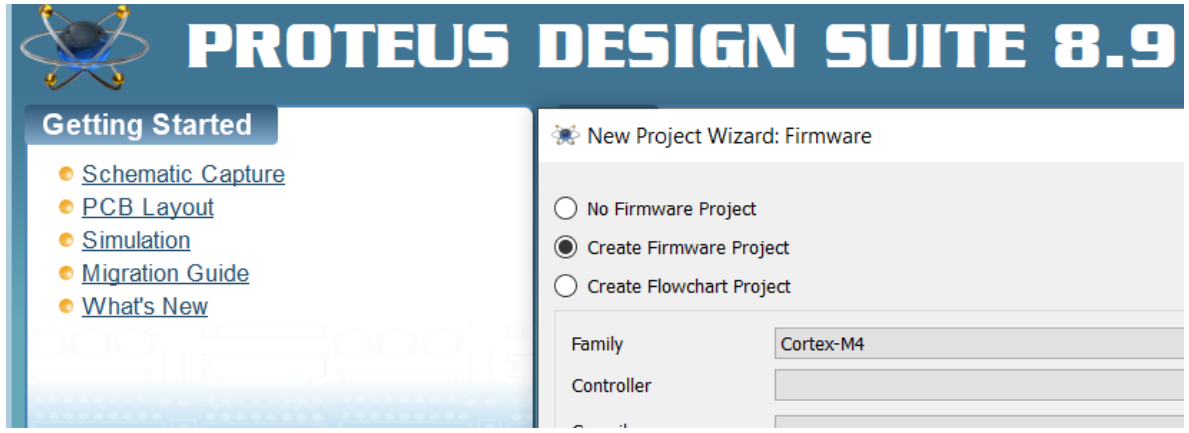


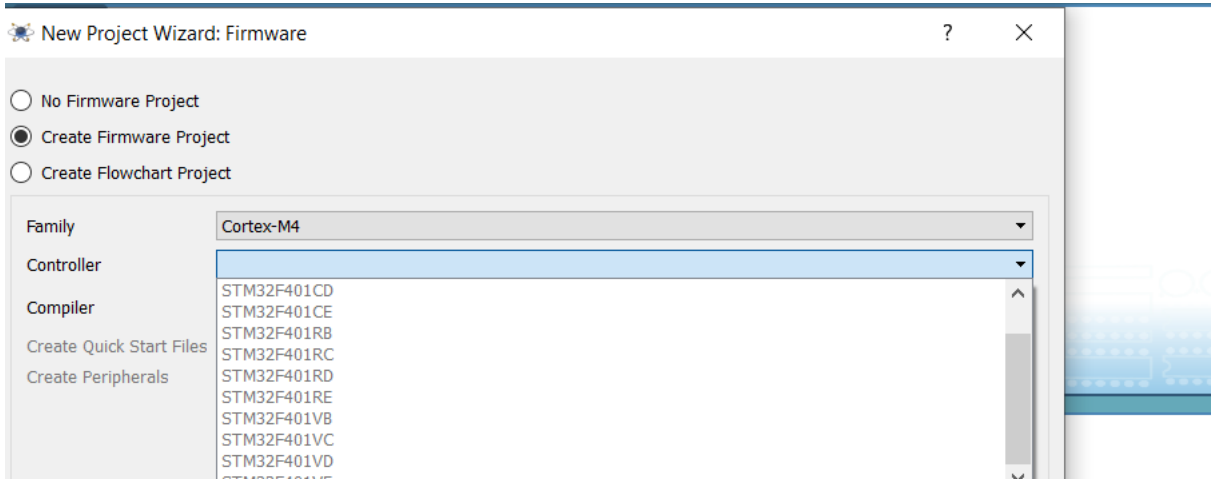


INGENIERÍA MECATRÓNICA UMNG

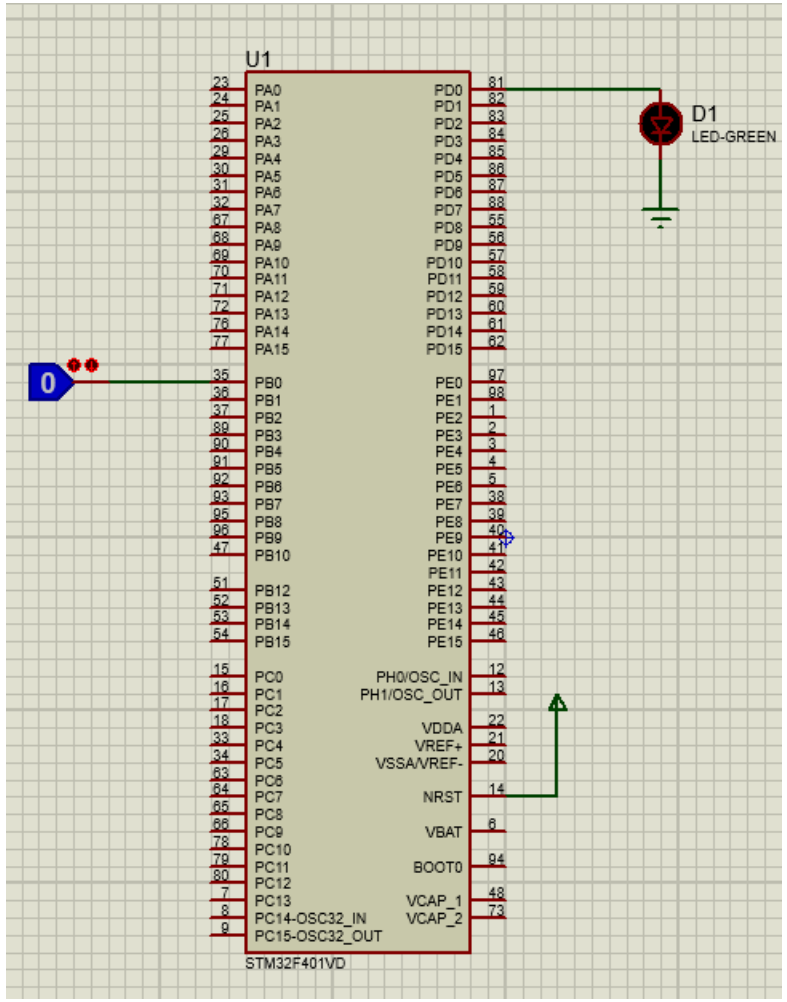
ENTORNO DE SIMULACION

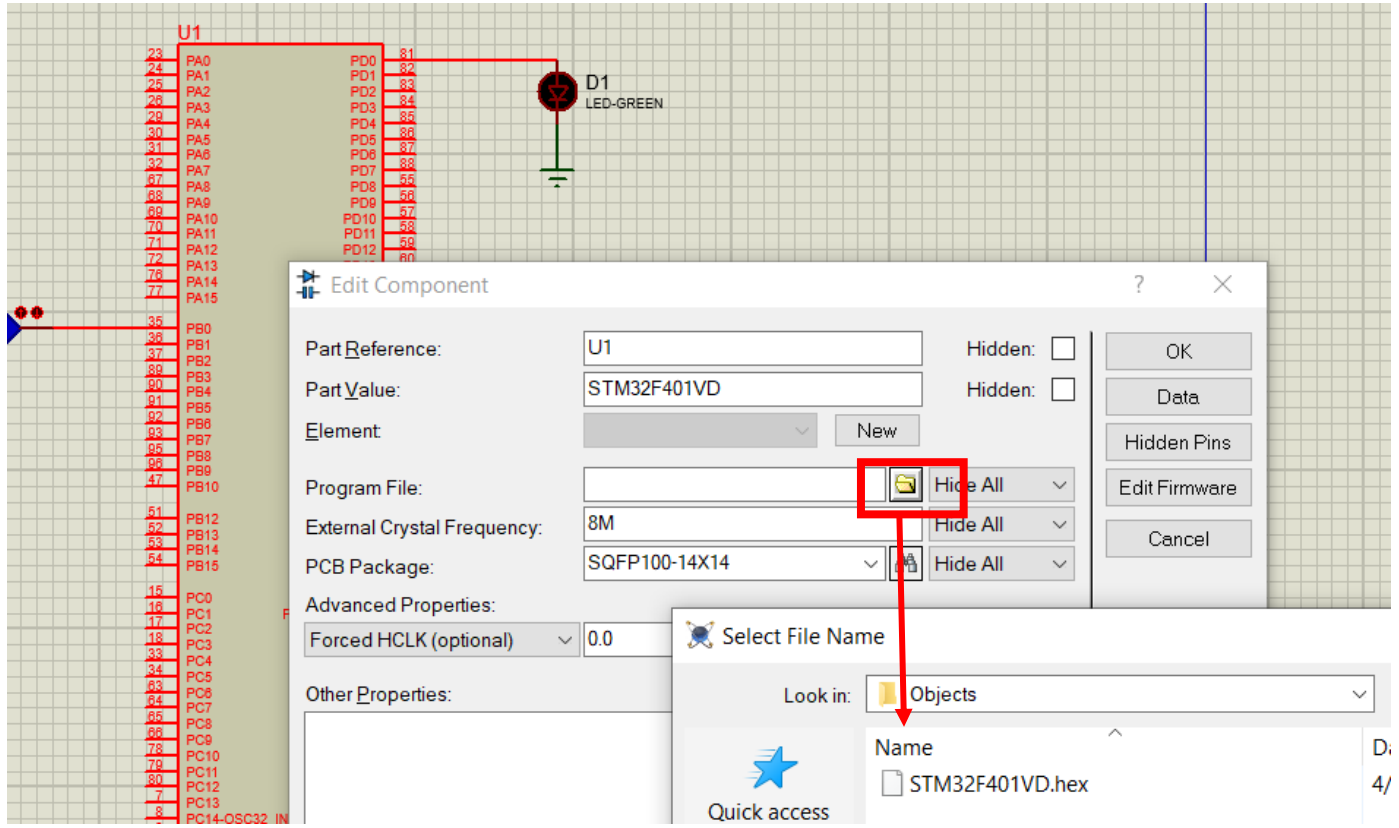


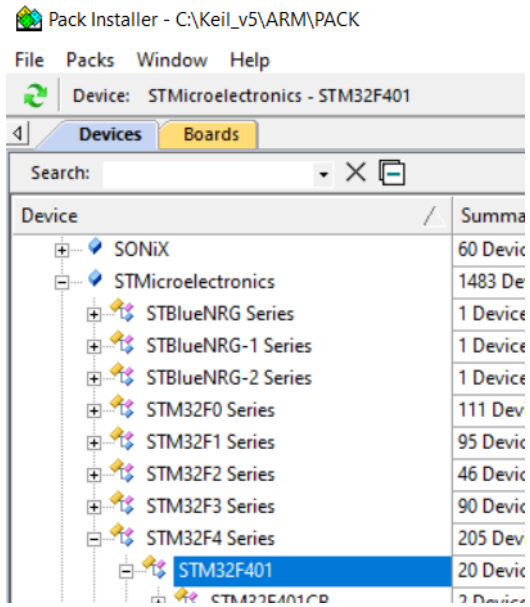


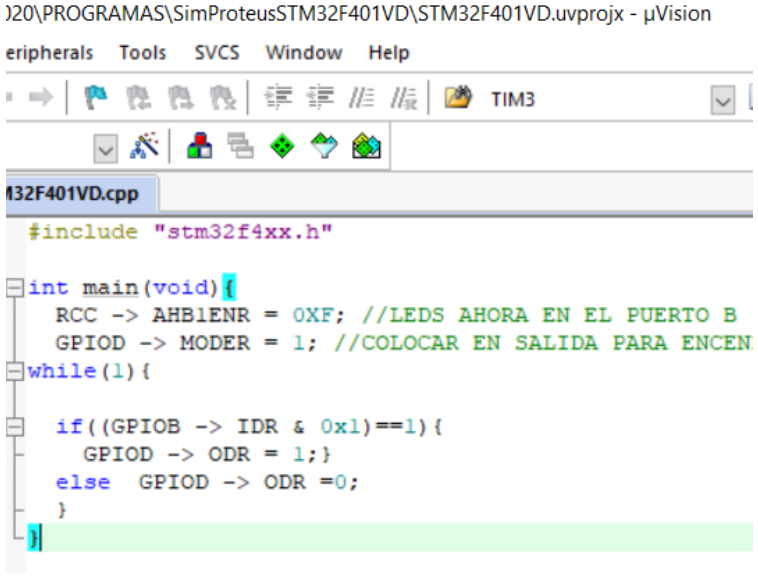


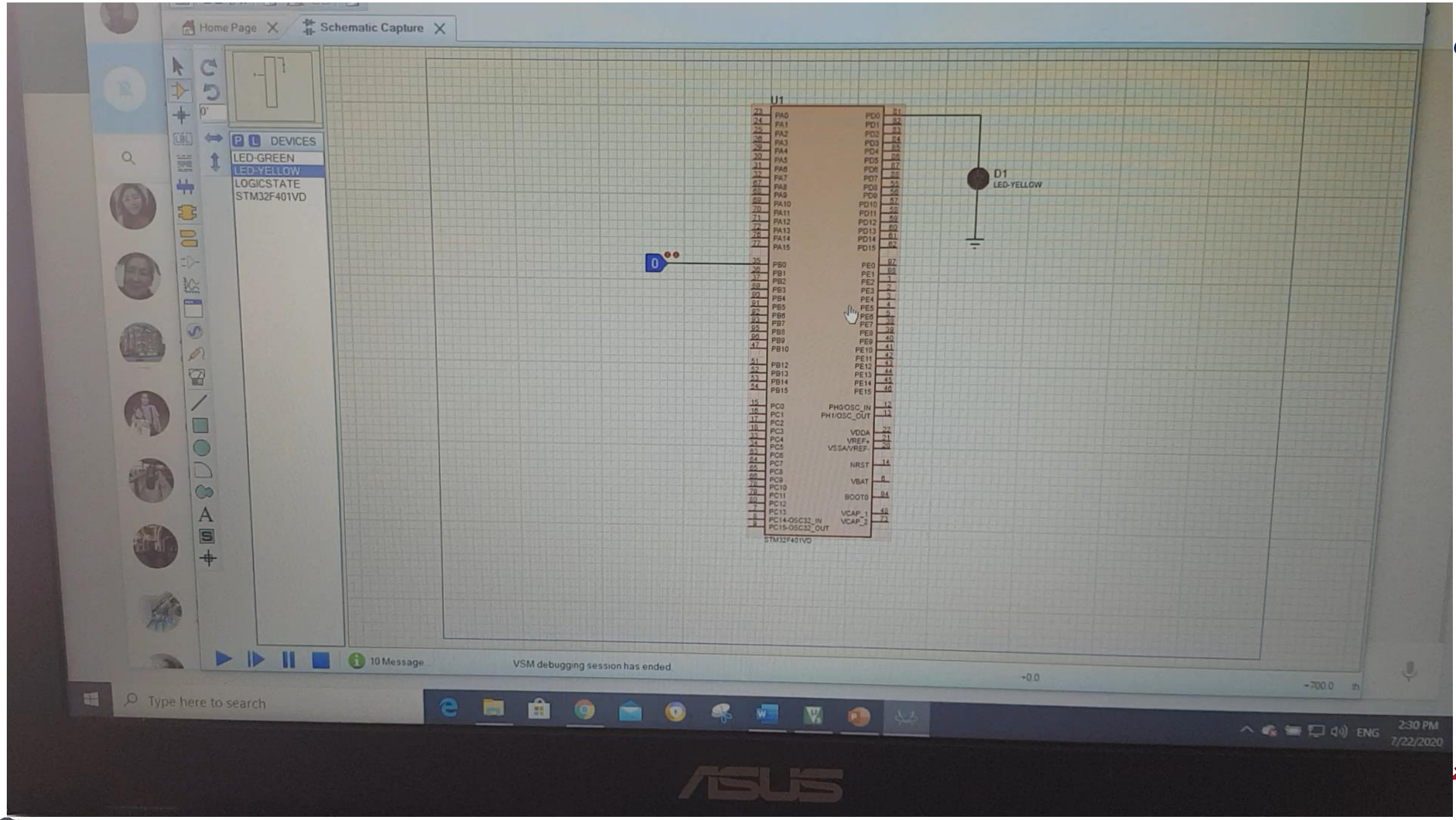
UNIVERSIDAD MILITAR NUEVA GRANADA


U1
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PA8
PA9
PA10
PA11
PA12
PA13
PA14
PA15
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PB8
PB9
PB10
PB12
PB13
PB14
PB15
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PC8
PC9
PC10
PC11
PC12
PC13
PC14-OSC32_IN
PC15-OSC32_OUT
PD0
PD1
PD2
PD3
PD4
PD5
PD6
PD7
PD8
PD9
PD10
PD11
PD12
PD13
PD14
PD15
PE0
PE1
PE2
PE3
PE4
PE5
PE6
PE7
PE8
PE9
PE10
PE11
PE12
PE13
PE14
PE15
PH0/OSC_IN
PH1/OSC_OUT
VDDA
VREF+
VSSA/VREF-
NRST
VBAT
BOOT0
VCAP_1
VCAP_2
STM32F401VD
D1
LED-GREEN
0


U1
D1
LED-GREEN
Edit Component
Part Reference:
U1
Hidden:
Part Value:
STM32F401VD
Hidden:
Element:
New
Program File:
Hide All
External Crystal Frequency:
8M
Hide All
PCB Package:
SQFP100-14X14
Hide All
Advanced Properties:
Forced HCLK (optional)
0.0
Other Properties:
OK
Data
Hidden Pins
Edit Firmware
Cancel
Select File Name
Look in:
Objects
Name
STM32F401VD.hex
Quick access


Pack Installer - C:\Keil_v5\ARM\PACK
File
Packs
Window
Help
Device: STMicroelectronics - STM32F401
Devices
Boards
Search:
Device
Summa
SONiX
60 Devic
STMicroelectronics
1483 De
STBlueNRG Series
1 Device
STBlueNRG-1 Series
1 Device
STBlueNRG-2 Series
1 Device
STM32F0 Series
111 Dev
STM32F1 Series
95 Devic
STM32F2 Series
46 Devic
STM32F3 Series
90 Devic
STM32F4 Series
205 Dev
STM32F401
20 Devic


020\PROGRAMAS\SimProteusSTM32F401VD\STM32F401VD.uvprojx - µVision
eripherals
Tools
SVCS
Window
Help
TIM3
32F401VD.cpp
#include "stm32f4xx.h"
int main(void){
RCC -> AHB1ENR = 0XF; //LEDS AHORA EN EL PUERTO B
GPIOD -> MODER = 1; //COLOCAR EN SALIDA PARA ENCEN
while(1){
if((GPIOB -> IDR & 0x1)==1){
GPIOD -> ODR = 1;}
else GPIOD -> ODR =0;
}
}



INGENIERÍA MECATRÓNICA UMNG

UNIVERSIDAD MILITAR NUEVA GRANADA

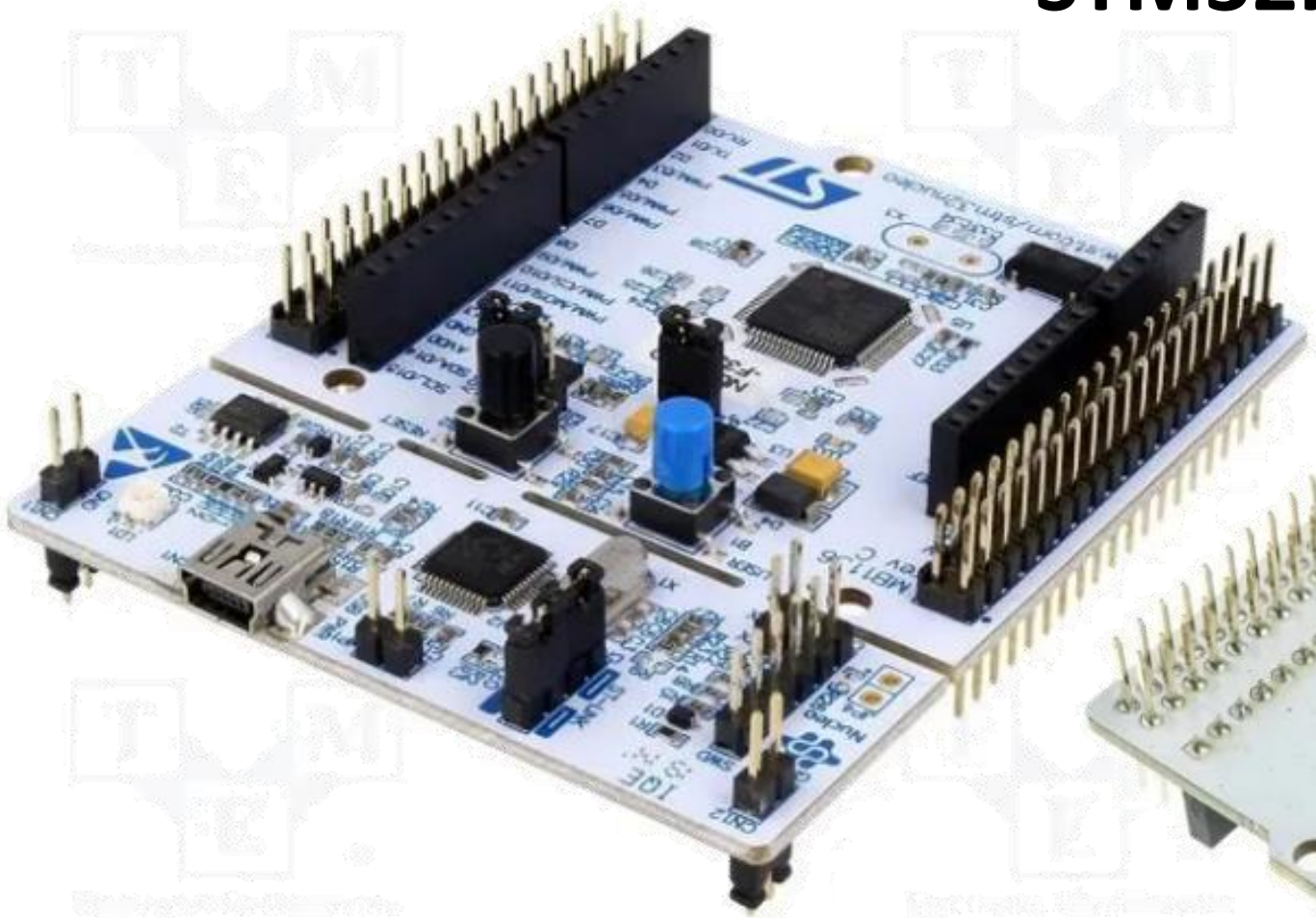
Arquitectura de un sistema embebido

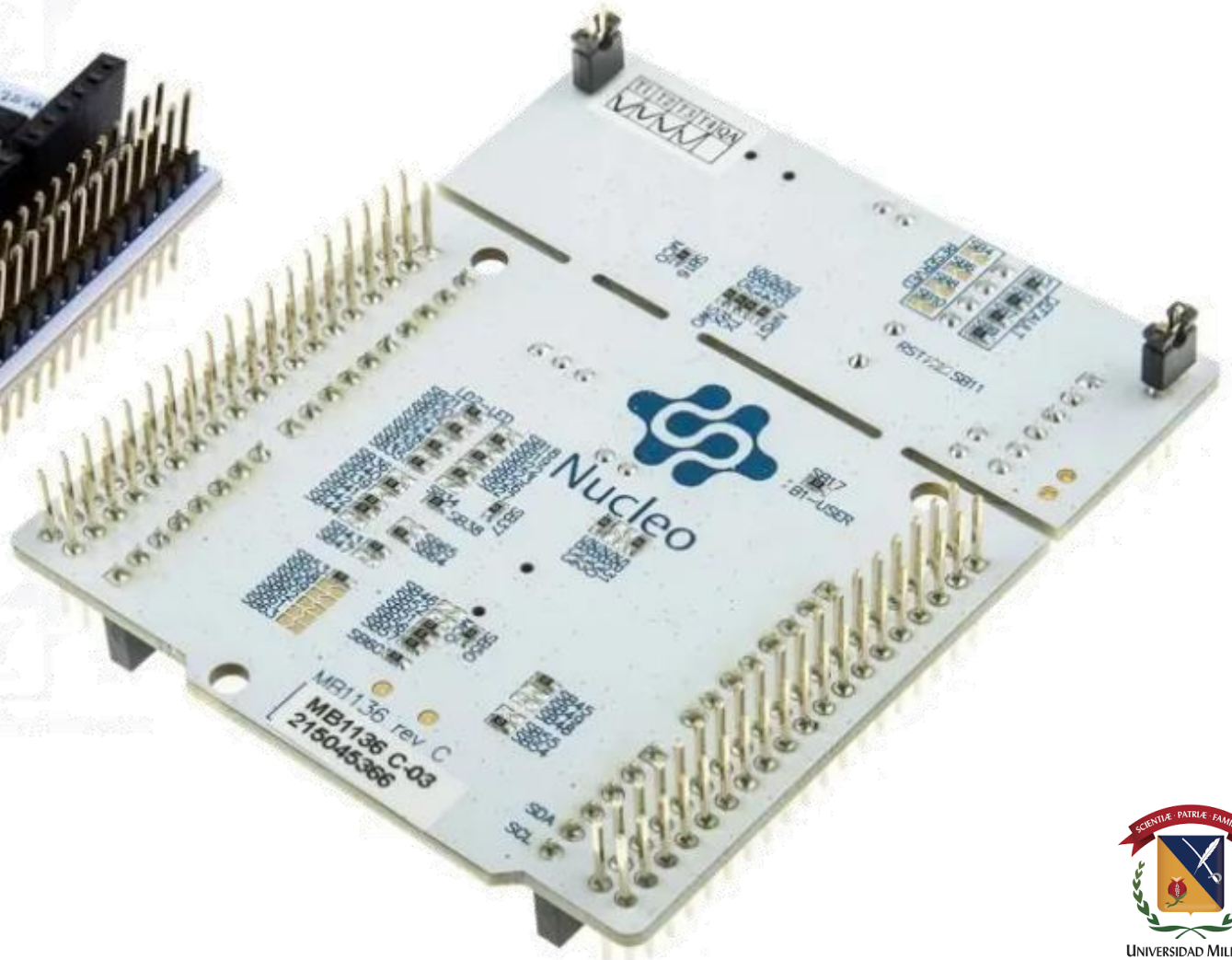
- Para un sistema embebido específico, como la STM32F4, se encuentran componentes como:

  - Microcontrolador ARM 32-bit $Cortex^{TM}$
  - Oscilador de 168 MHZ
  - *Memory Flash* de 1 Mbyte
  - SRAM de 192+4 Kbyte
  - CCM (Core Coupled Memory) de 64 Kbyte
  - Interfaz paralela de LCD
  - Interfaz paralela a 8-14 bits de câmara a 54 Mbytes/s
  - Unidad de cálculo CRC
  - RTC: Precisión inferior a segundos y Calendario Hardware
  - 3 conversores A/D de 12 bits
  - 2 conversores D/A de 12 bits
  - 17 temporizadores, 15 de estos a 16 bits y 2 a 32 bits, con funciones IC/OC/PWM
  - 140 I/O con capacidad de interrupción
  - 15 interfaces de comunicación
  - 4 LEDs de uso general
  - Conexiones para periféricos de entrada y salida

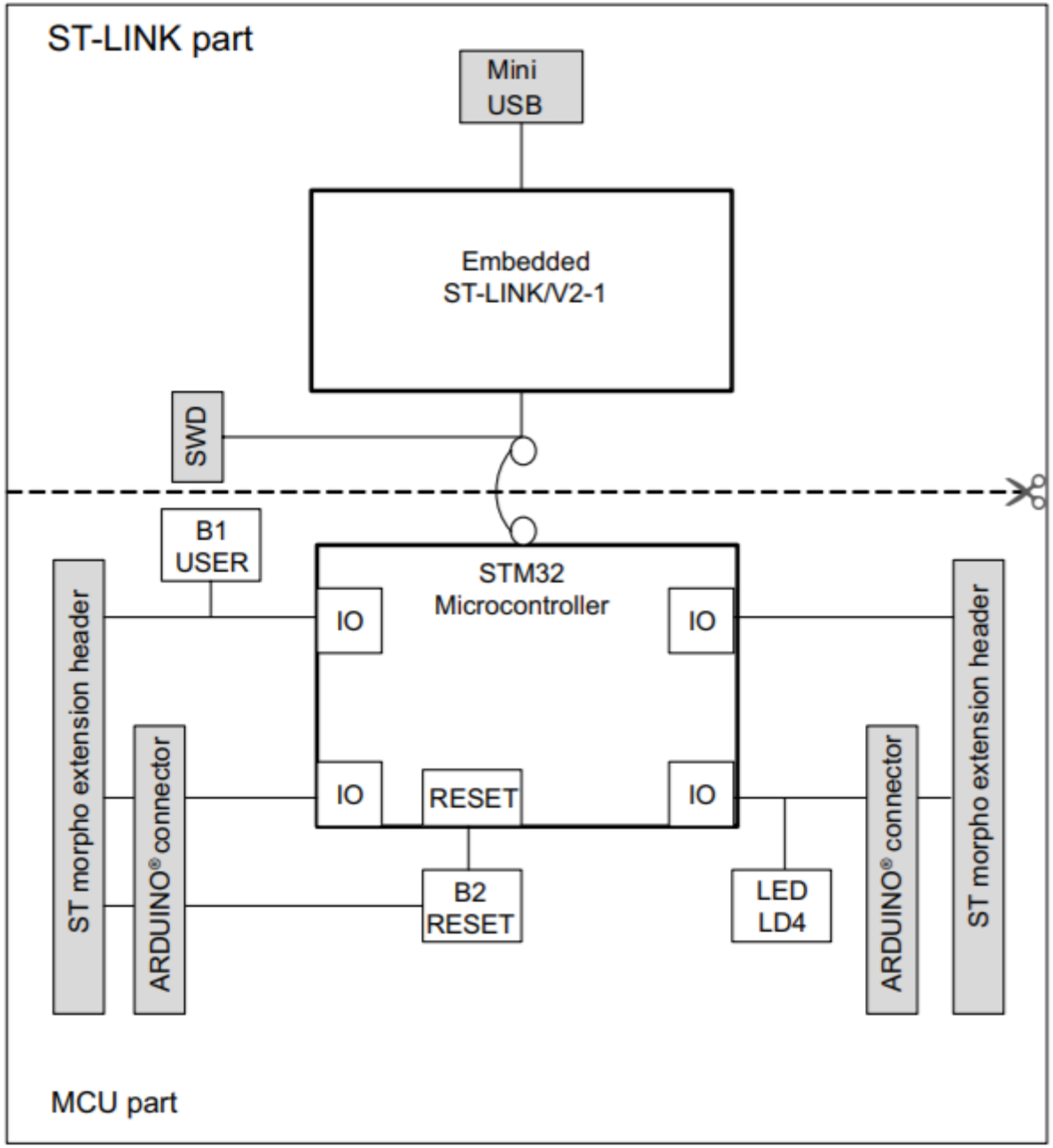
INGENIERÍA MECATRÓNICA UMNG

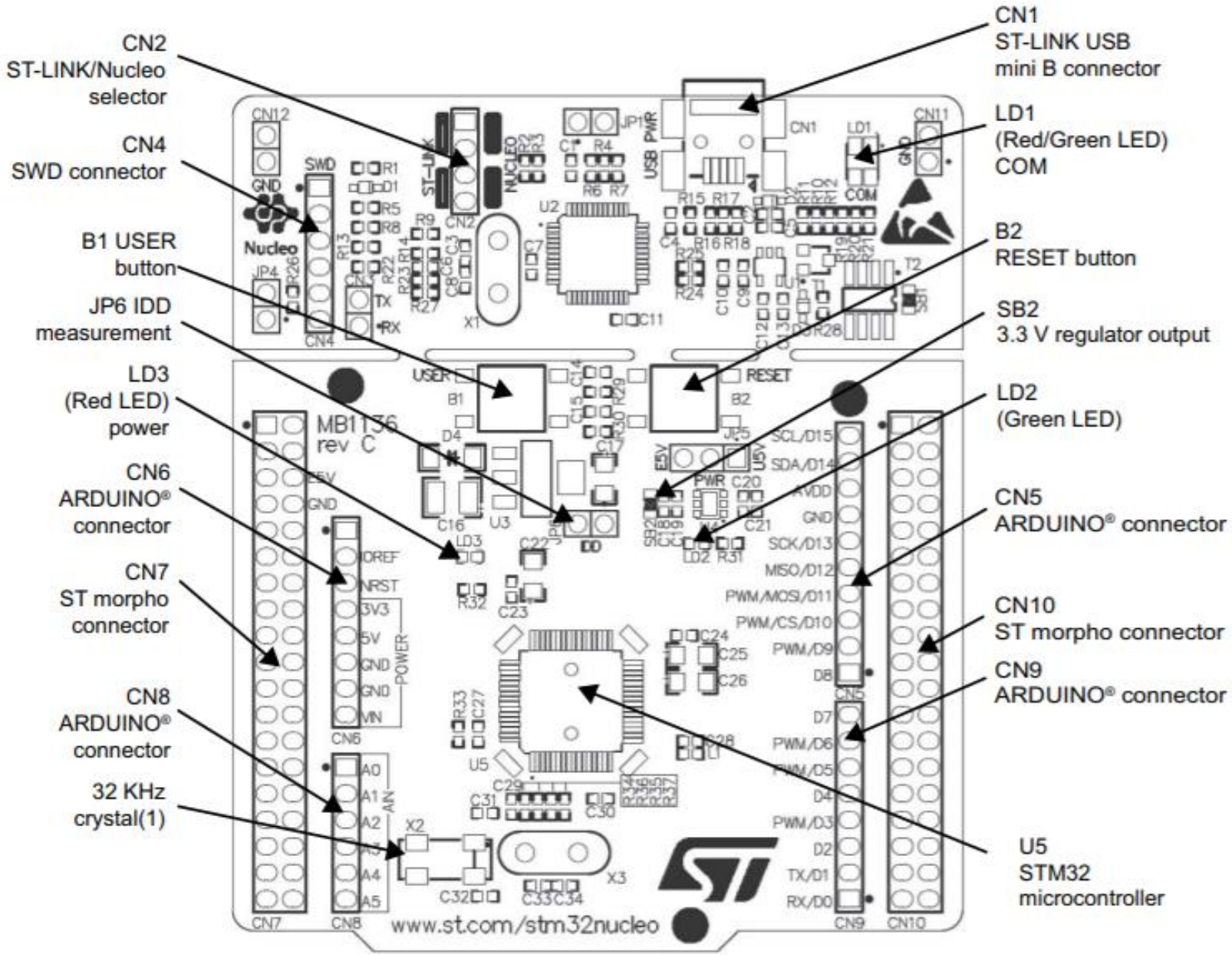
STM32F4

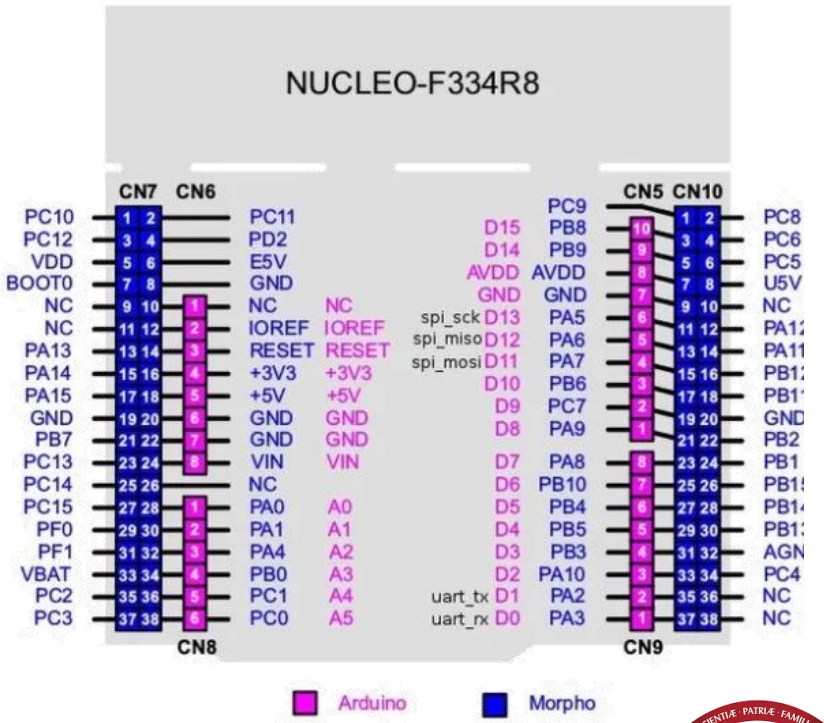


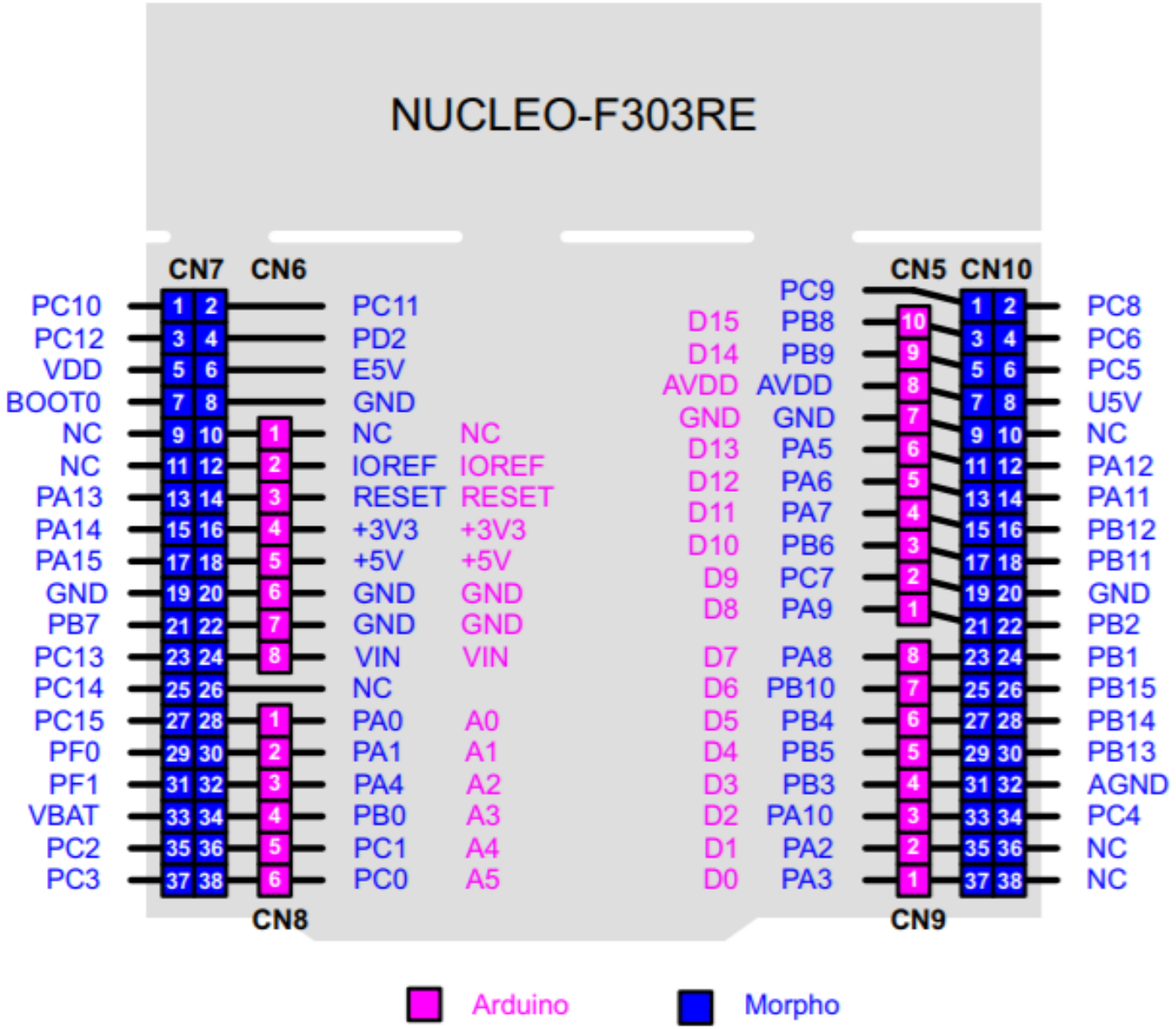


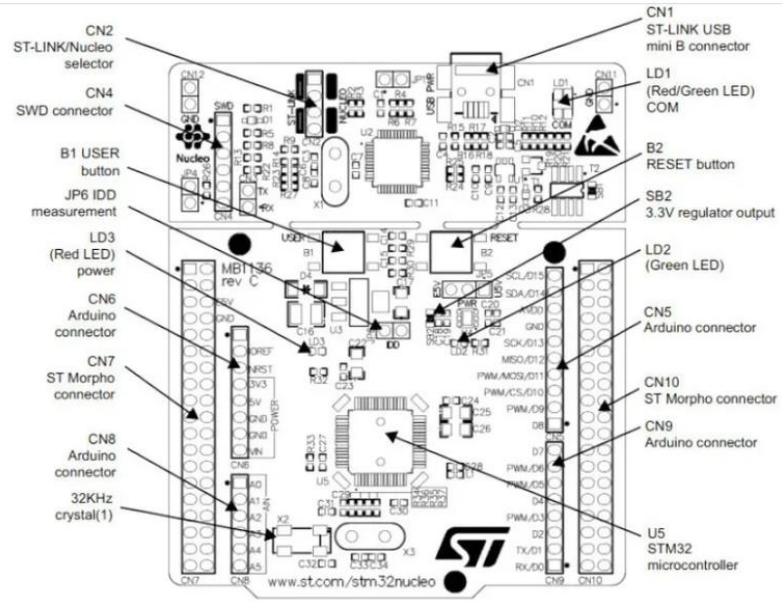
UNIVERSIDAD MILITAR NUEVA GRANADA


ST-LINK part
Mini USB
Embedded ST-LINK/V2-1
SWD
B1 USER
STM32 Microcontroller
IO
IO
IO
IO
RESET
B2 RESET
LED LD4
ST morpho extension header
ARDUINO® connector
ARDUINO® connector
ST morpho extension header
MCU part


CN2
ST-LINK/Nucleo
selector
CN4
SWD connector
B1 USER
button
JP6 IDD
measurement
LD3
(Red LED)
power
CN6
ARDUINO®
connector
CN7
ST morpho
connector
CN8
ARDUINO®
connector
32 KHz
crystal(1)
CN1
ST-LINK USB
mini B connector
LD1
(Red/Green LED)
COM
B2
RESET button
SB2
3.3 V regulator output
LD2
(Green LED)
CN5
ARDUINO® connector
CN10
ST morpho connector
CN9
ARDUINO® connector
U5
STM32
microcontroller
MB1136
rev C
www.st.com/stm32nucleo


NUCLEO-F334R8
CN7 CN6
CN5 CN10
CN8
CN9
Arduino
Morpho


NUCLEO-F303RE
CN7
CN6
PC10 1 2 PC11
PC12 3 4 PD2
VDD 5 6 E5V
BOOT0 7 8 GND
NC 9 10 NC NC
NC 11 12 IOREF IOREF
PA13 13 14 RESET RESET
PA14 15 16 +3V3 +3V3
PA15 17 18 +5V +5V
GND 19 20 GND GND
PB7 21 22 GND GND
PC13 23 24 VIN VIN
PC14 25 26 NC
PC15 27 28 PA0 A0
PF0 29 30 PA1 A1
PF1 31 32 PA4 A2
VBAT 33 34 PB0 A3
PC2 35 36 PC1 A4
PC3 37 38 PC0 A5
CN8
CN5
CN10
PC9
D15 PB8
D14 PB9
AVDD AVDD
GND GND
D13 PA5
D12 PA6
D11 PA7
D10 PB6
D9 PC7
D8 PA9
D7 PA8
D6 PB10
D5 PB4
D4 PB5
D3 PB3
D2 PA10
D1 PA2
D0 PA3
1 2 PC8
3 4 PC6
5 6 PC5
7 8 U5V
9 10 NC
11 12 PA12
13 14 PA11
15 16 PB12
17 18 PB11
19 20 GND
21 22 PB2
23 24 PB1
25 26 PB15
27 28 PB14
29 30 PB13
31 32 AGND
33 34 PC4
35 36 NC
37 38 NC
CN9
Arduino
Morpho


CN2
ST-LINK/Nucleo selector
CN4
SWD connector
B1 USER button
JP6 IDD measurement
LD3 (Red LED) power
CN6 Arduino connector
CN7 ST Morpho connector
CN8 Arduino connector
32KHz crystal(1)
CN1 ST-LINK USB mini B connector
LD1 (Red/Green LED) COM
B2 RESET button
SB2 3.3V regulator output
LD2 (Green LED)
CN5 Arduino connector
CN10 ST Morpho connector
CN9 Arduino connector
U5 STM32 microcontroller
www.st.com/stm32nucleo

LEDs

The tricolor LED (green, orange, red) LD1 (COM) provides information about ST-LINK communication status. LD1 default color is red. LD1 turns to green to indicate that communication is in progress between the PC and the ST-LINK/V2-1, with the following setup:

- Slow blinking Red/Off: at power-on before USB initialization
- Fast blinking Red/Off: after the first correct communication between the PC and ST-LINK/V2-1 (enumeration)
- Red LED On: when the initialization between the PC and ST-LINK/V2-1 is complete
- Green LED On: after a successful target communication initialization
- Blinking Red/Green: during communication with the target
- Green On: communication finished and successful
- Orange On: Communication failure

User LD2: the green LED is a user LED connected to ARDUINO® signal D13 corresponding to STM32 I/O PA5 (pin 21) or PB13 (pin 34) depending on the STM32 target. Refer to *Table 11* to *Table 23* when:

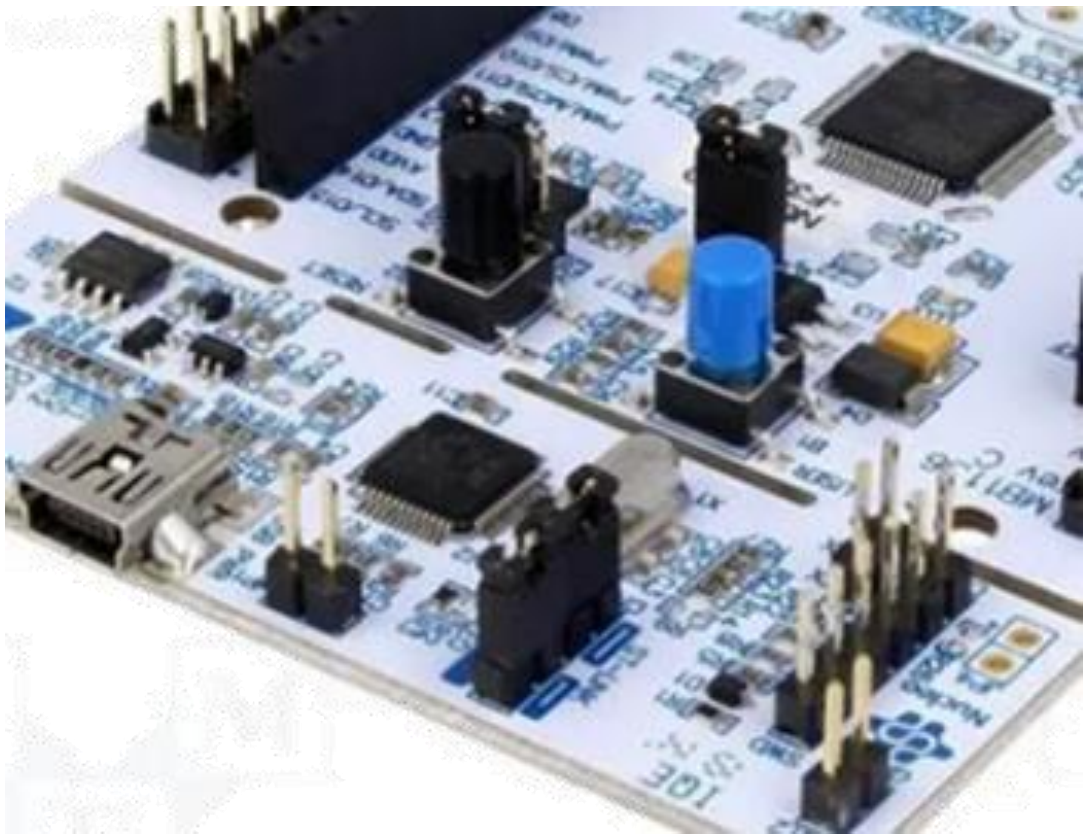
- the I/O is HIGH value, the LED is on
- the I/O is LOW, the LED is off

LD3 PWR: the red LED indicates that the STM32 part is powered and +5V power is available.

Push-buttons

B1 USER: the user button is connected to the I/O PC13 (pin 2) of the STM32 microcontroller.

B2 RESET: this push-button is connected to NRST, and is used to RESET the STM32 microcontroller.



```
#include "STM32F7xx.h"

//BASICO LED 1 EJEMPLO ENTRADA SALIDA DIGITAL.

int main(void)
{
  //**************************************************************
  //CONFIGURACION "CLOCK"
  RCC->AHB1ENR |= 6;    //PRENDER EL CLOCK DEL PTB

  //**************************************************************
  //CONFIGURACION DE PINES
  GPIOB->MODER =1;  //PTB  -> OUTPUT
  GPIOC->MODER =0;           //PTC  -> INPUT
  //**************************************************************

  while(true){        //bucle infinito
    //Lee pulsador enciende LED
     while((GPIOC->IDR & 0X2000)==0X2000 ){ GPIOB->ODR |= 1; }
     GPIOB->ODR = 0;
  }//cierra while

}//cierra main
```

GPIO functional description

Subject to the specific hardware characteristics of each I/O port listed in the datasheet, each port bit of the general-purpose I/O (GPIO) ports can be individually configured by software in several modes:

- Input floating
- Input pull-up
- Input-pull-down
- Analog
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Alternate function push-pull with pull-up or pull-down capability
- Alternate function open-drain with pull-up or pull-down capability

Each I/O port bit is freely programmable, however the I/O port registers have to be accessed as 32-bit words, half-words or bytes. The purpose of the GPIOx_BSRR and GPIOx_BRR registers is to allow atomic read/modify accesses to any of the GPIOx_ODR registers. In this way, there is no risk of an IRQ occurring between the read and the modify access.

General-purpose I/O (GPIO)

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in input floating mode.

The debug pins are in AF pull-up/pull-down after reset:

- PA15: JTDI in pull-up
- PA14: JTCK/SWCLK in pull-down
- PA13: JTMS/SWDAT in pull-up
- PB4: NJTRST in pull-up
- PB3: JTDO in floating state

```
GPIOA->MODER |= 0x
```

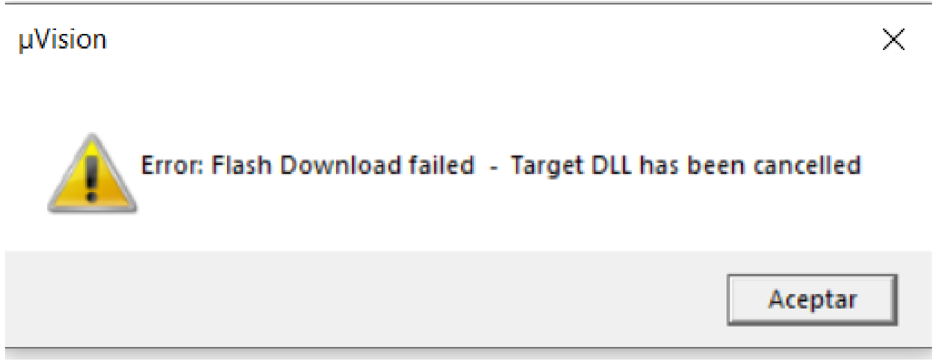
Enmascaramiento necesario

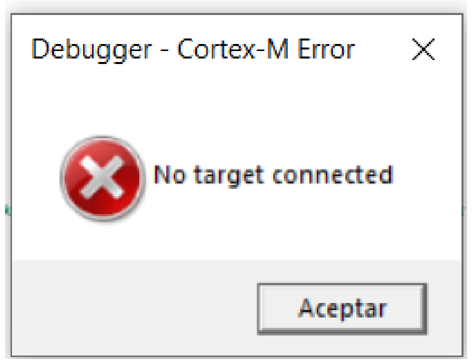
When the pin is configured as output, the value written to the output data register (GPIOx_ODR) is output on the I/O pin. It is possible to use the output driver in push-pull mode or open-drain mode (only the low level is driven, high level is HI-Z).

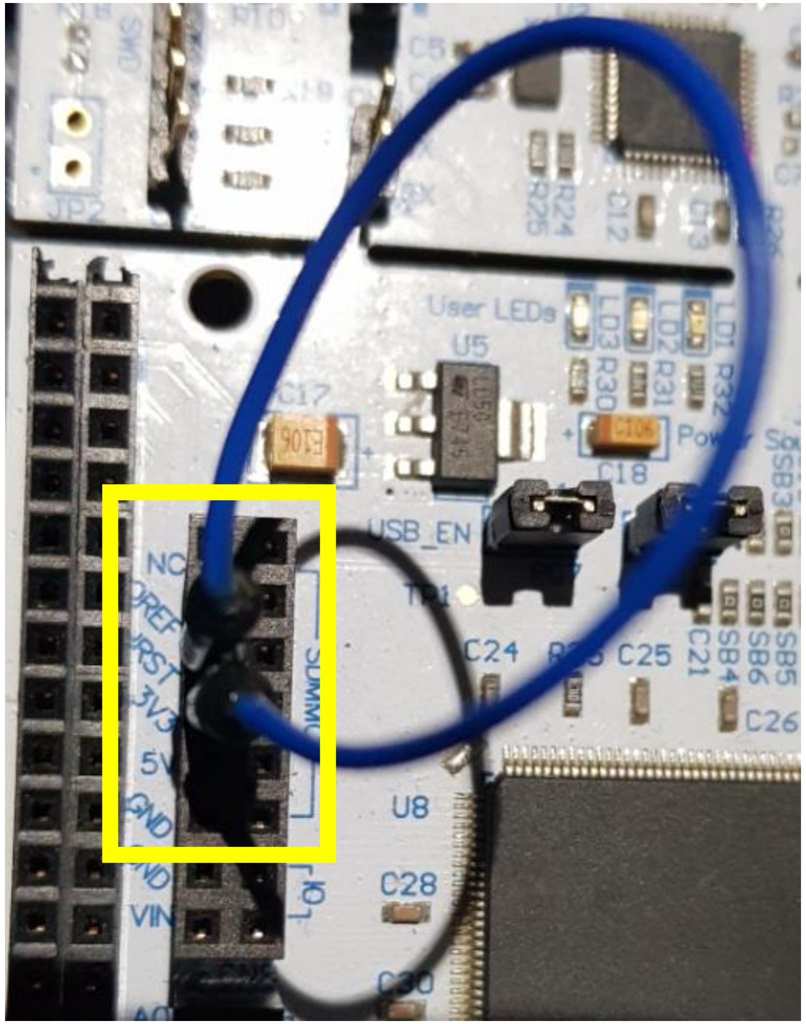
The input data register (GPIOx_IDR) captures the data present on the I/O pin at every AHB clock cycle.

All GPIO pins have weak internal pull-up and pull-down resistors, which can be activated or not depending on the value in the GPIOx_PUPDR register.

RESET

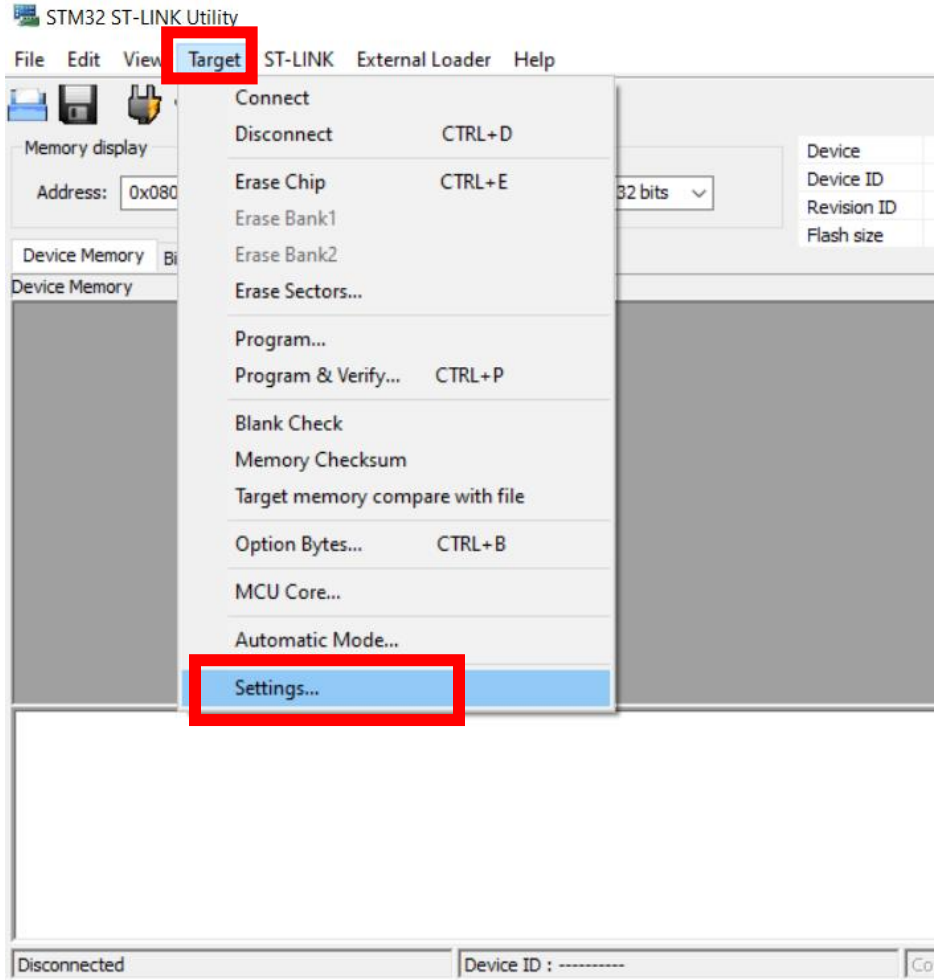

µVision
Error: Flash Download failed - Target DLL has been cancelled
Aceptar

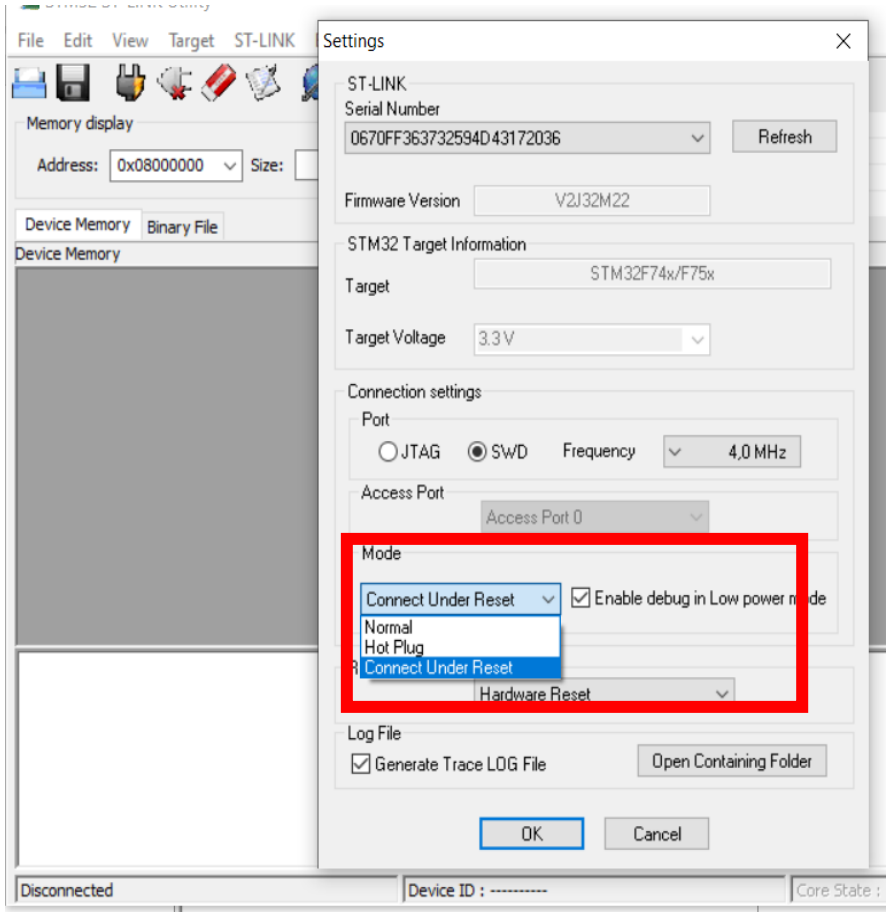

Debugger - Cortex-M Error
No target connected
Aceptar

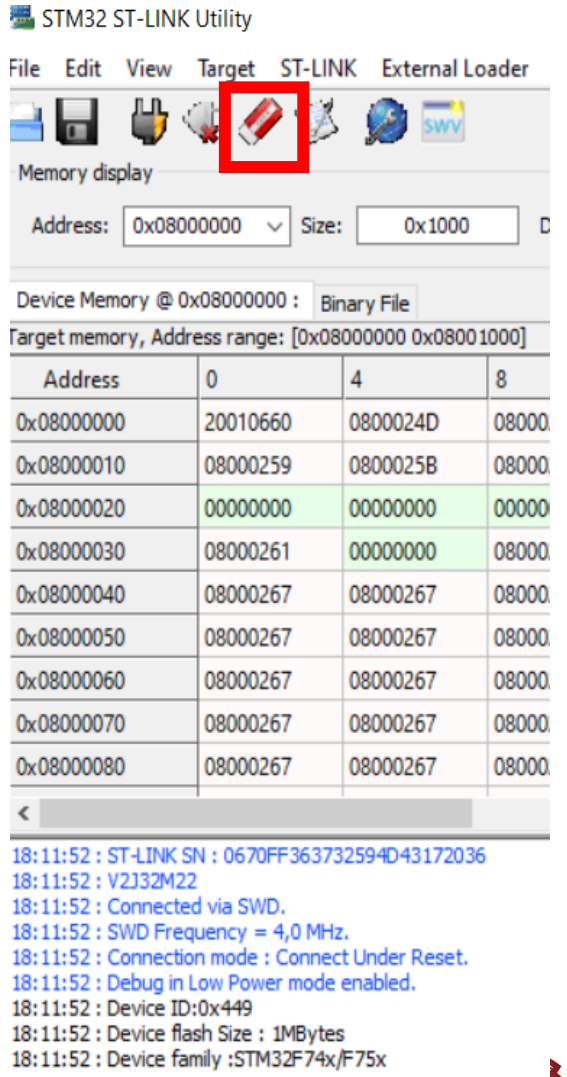

User LEDs
U5
C17
USB_EN
C24
C25
C26
U8
C28
NC
VIN


ST-LINK Utility
STM32 ST-LINK Utility

RESET


STM32 ST-LINK Utility
File Edit View Target ST-LINK External Loader Help
Memory display
Address: 0x080
Device Memory
Connect
Disconnect CTRL+D
Erase Chip CTRL+E
Erase Bank1
Erase Bank2
Erase Sectors...
Program...
Program & Verify... CTRL+P
Blank Check
Memory Checksum
Target memory compare with file
Option Bytes... CTRL+B
MCU Core...
Automatic Mode...
Settings...
Device
Device ID
Revision ID
Flash size
Disconnected
Device ID : ----------


Settings
ST-LINK
Serial Number
0670FF363732594D43172036
Refresh
Firmware Version
V2J32M22
STM32 Target Information
Target
STM32F74x/F75x
Target Voltage
3.3 V
Connection settings
Port
JTAG SWD Frequency 4,0 MHz
Access Port
Access Port 0
Mode
Connect Under Reset
Enable debug in Low power mode
Normal
Hot Plug
Connect Under Reset
Hardware Reset
Log File
Generate Trace LOG File
Open Containing Folder
OK
Cancel
Memory display
Address: 0x08000000
Size:
Device Memory
Binary File
Disconnected
Device ID : ----------


STM32 ST-LINK Utility
File Edit View Target ST-LINK External Loader
Memory display
Address: 0x08000000
Size: 0x1000
Device Memory @ 0x08000000 :
Binary File
Target memory, Address range: [0x08000000 0x08001000]
Address 0 4 8
0x08000000 20010660 0800024D
0x08000010 08000259 0800025B
0x08000020 00000000 00000000
0x08000030 08000261 00000000
0x08000040 08000267 08000267
0x08000050 08000267 08000267
0x08000060 08000267 08000267
0x08000070 08000267 08000267
0x08000080 08000267 08000267
18:11:52 : ST-LINK SN : 0670FF363732594D43172036
18:11:52 : V2J32M22
18:11:52 : Connected via SWD.
18:11:52 : SWD Frequency = 4,0 MHz.
18:11:52 : Connection mode : Connect Under Reset.
18:11:52 : Debug in Low Power mode enabled.
18:11:52 : Device ID:0x449
18:11:52 : Device flash Size : 1MBytes
18:11:52 : Device family :STM32F74x/F75x

Clocks

Three different clock sources can be used to drive the system clock (SYSCLK):

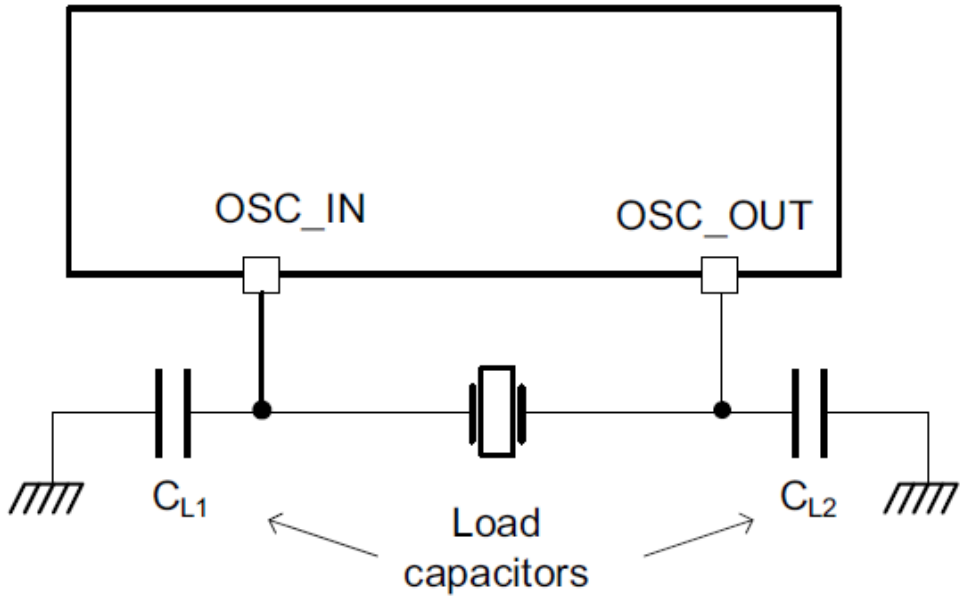
- HSI oscillator clock =**16 MHz**
- HSE oscillator clock =**8MHz**
- Main PLL (PLL) clock

The devices have the two following secondary clock sources:

- 32 kHz low-speed internal RC (LSI RC) which drives the independent watchdog and, optionally, the RTC used for Auto-wakeup from the Stop/Standby mode.
- 32.768 kHz low-speed external crystal (LSE crystal) which optionally drives the RTC clock (RTCCLK)

Each clock source can be switched on or off independently when it is not used, to optimize power consumption.

Crystal/ceramic resonators



PLL

The devices feature three PLLs:

- A main PLL (PLL) clocked by the HSE or HSI oscillator and featuring two different output clocks:
  - The first output is used to generate the high speed system clock (up to 216 MHz)
  - The second output is used to generate 48MHz clock for the USB OTG FS, SDMMC and RNG.

HSI clock

The HSI clock signal is generated from an internal 16 MHz RC oscillator and can be used directly as a system clock, or used as PLL input.

I/O port control registers

Each of the GPIO ports has four 32-bit memory-mapped control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) to configure up to 16 I/Os. The GPIOx_MODER register is used to select the I/O mode (input, output, AF, analog). The GPIOx_OTYPER and GPIOx_OSPEEDR registers are used to select the output type (push-pull or open-drain) and speed. The GPIOx_PUPDR register is used to select the pull-up/pull-down whatever the I/O direction.

I/O port data registers

Each GPIO has two 16-bit memory-mapped data registers: input and output data registers (GPIOx_IDR and GPIOx_ODR). GPIOx_ODR stores the data to be output, it is read/write accessible. The data input through the I/O are stored into the input data register (GPIOx_IDR), a read-only register.

General-purpose I/Os (GPIO)

- Output states: push-pull or open drain + pull-up/down
- Output data from output data register (GPIOx_ODR) or peripheral (alternate function
- output)
- Speed selection for each I/O
- Input states: floating, pull-up/down, analog
- Input data to input data register (GPIOx_IDR) or peripheral (alternate function input)
- Bit set and reset register (GPIOx_ BSRR) for bitwise write access to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configurations
- Analog function
- Alternate function selection registers
- Fast toggle capable of changing every two clock cycles
- Highly flexible pin multiplexing allows the use of I/O pins as GPIOs or as one of several
- peripheral functions

5.3.10 RCC AHB1 peripheral clock register (RCC_AHB1ENR)

Address offset: 0x30

Reset value: 0x0010 0000

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

```
RCC -> AHB1ENR =
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	OTGHSULPIEN	OTGHSEN	ETHMACPTPEN	ETHMACRXEN	ETHMACTXEN	ETHMACEN	Res.	DMA2DEN	DMA2EN	DMA1EN	DTCMRAMEN	Res.	BKPSRAMEN	Res.	Res.
	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw		rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCEN	Res.	GPIOKEN	GPIOJEN	GPIOIEN	GPIOHEN	GPIOGEN	GPIOFEN	GPIOEEN	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
			rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 2 **GPIOCEN:** IO port C clock enable

This bit is set and cleared by software.

0: IO port C clock disabled

1: IO port C clock enabled

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

This bit is 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

This bit is set and cleared by software.

0: IO port A clock disabled

1: IO port A clock enabled

5.3.10 RCC AHB1 peripheral clock register (RCC_AHB1ENR)

```
RCC -> AHB1ENR =
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	OTGHSULPIEN	OTGHSEN	ETHMACPTPEN	ETHMACRXEN	ETHMACTXEN	ETHMACEN	Res.	DMA2DEN	DMA2EN	DMA1EN	DTCMRAMEN	Res.	BKPSRAMEN	Res.	Res.
	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw		rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRCEN	Res.	GPIOKEN	GPIOJEN	GPIOIEN	GPIOHEN	GPIOGEN	GPIOFEN	GPIOEEN	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
			rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Que puertos se activan con las siguientes instrucciones:

```
RCC -> AHB1ENR = 16;
```

```
RCC -> AHB1ENR = 0x10;
```

Escribir la intrucción para activar el o los puertos indicados:

1. Puerto B
2. Puerto A y C
3. Puerto A,B,C y E
4. Puerto D y G

6.4.1 GPIO port mode register (GPIOx_MODER) (x =A..K)

Address offset:0x00

Reset values:

- 0xA800 0000 for port A
- 0x0000 0280 for port B
- 0x0000 0000 for other ports

```
GPIOB -> MODER =
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y+1:2y **MODERy[1:0]:** Port x configuration bits (y = 0..15)

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

00: Input mode (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

6.4.1 GPIO port mode register (GPIOx_MODER) (x =A..K)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y+1:2y **MODERy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O mode.
00: Input mode (reset state)
01: General purpose output mode

```
GPIOA -> MODER = 0;
```

Todos los bits del Puerto A de entrada

Que bits y en que modo se configuran con las siguientes instrucciones:

```
GPIOD -> MODER = 1;    GPIOD -> MODER = 16;
```

Escribir la intrucción para activar el o los bits indicados:

1. Puerto B, pines pares salida, impares entrada
2. Puerto C pines impares salida, pares entrada
3. Puerto E parte alta salida, parte baja entrada

6.4.6 GPIO port output data register (GPIOx_ODR) (x = A..K)

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

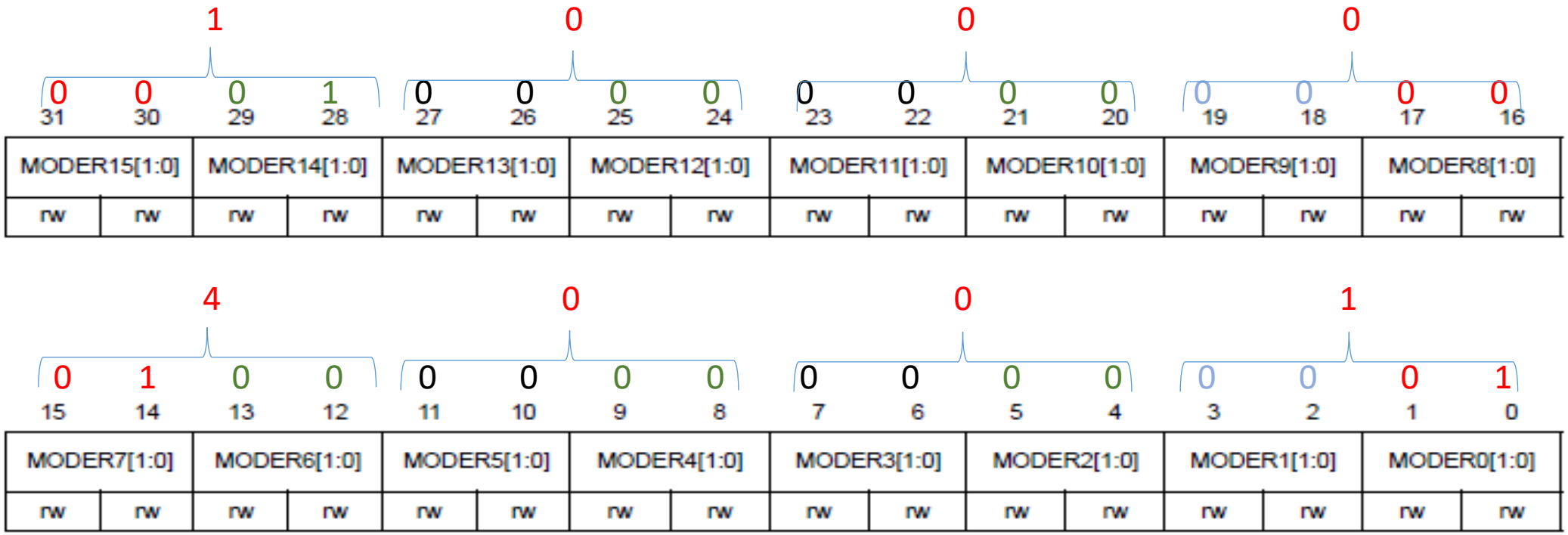
These bits can be read and written by software.

Note: For atomic bit set/reset, the ODR bits can be individually set and/or reset by writing to the GPIOx_BSRR or GPIOx_BRR registers (x = A..F).

```
GPIOD -> ODR = a;
```

6.4.1 GPIO port mode register (GPIOx_MODER) (x =A..K)

```
GPIOB -> MODER = 0X10004001; //LEDS TARJETA
```



Bits 2y+1:2y **MODERy[1:0]:** Port x configuration bits (y = 0..15)

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

00: Input mode (reset state)

01: General purpose output mode

10: Alternate function mode

11: Analog mode

6.4.6 GPIO port output data register (GPIOx_ODR) (x = A..K)

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
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

BIN	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
HEX			4				0				8				1	

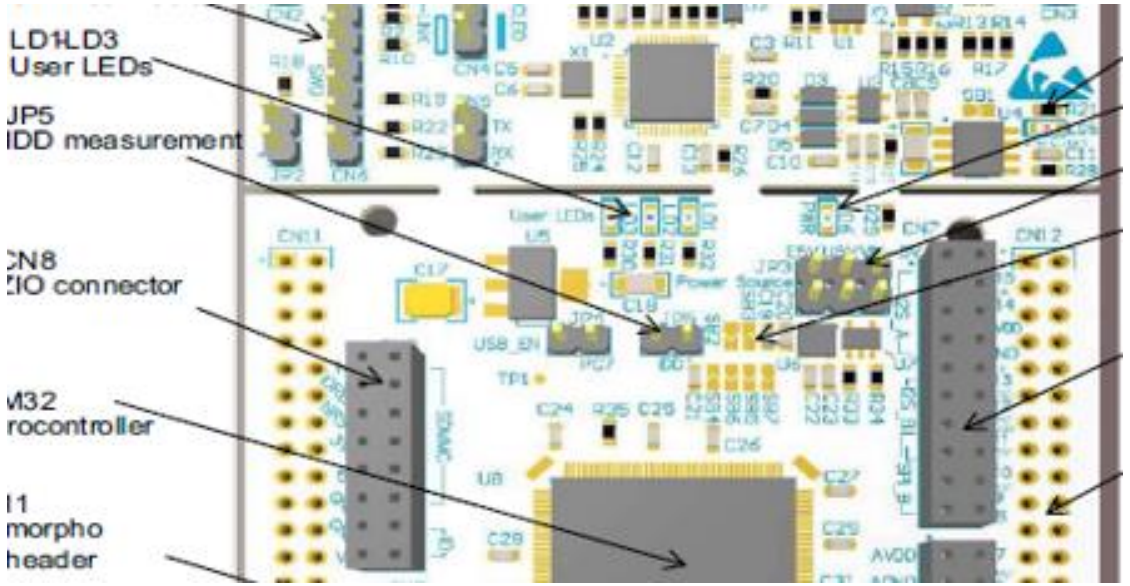
```
GPIOB -> ODR = 0X4081; //ENCENDER LOS 3 LEDS
```

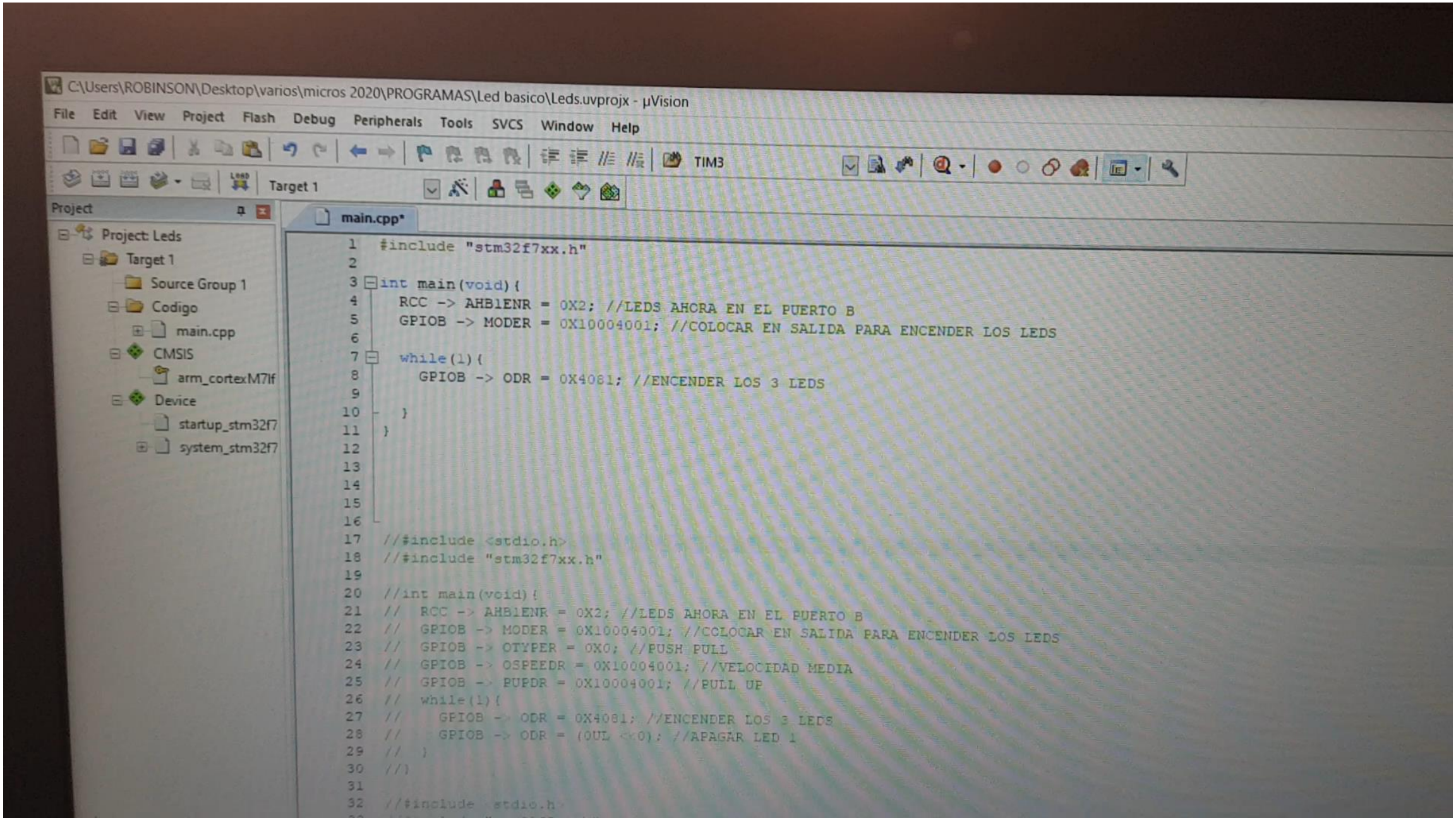
LEDs

User LD1: a green user LED is connected to the STM32 I/O PB0 OFF) or PA5 (SB119 ON and SB120 OFF) corresponding to the S

User LD2: a blue user LED is connected to PB7

User LD3: a red user LED is connected to PB14.





6.4.5 GPIO port input data register (GPIOx_IDR) (x = A..K)

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
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
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

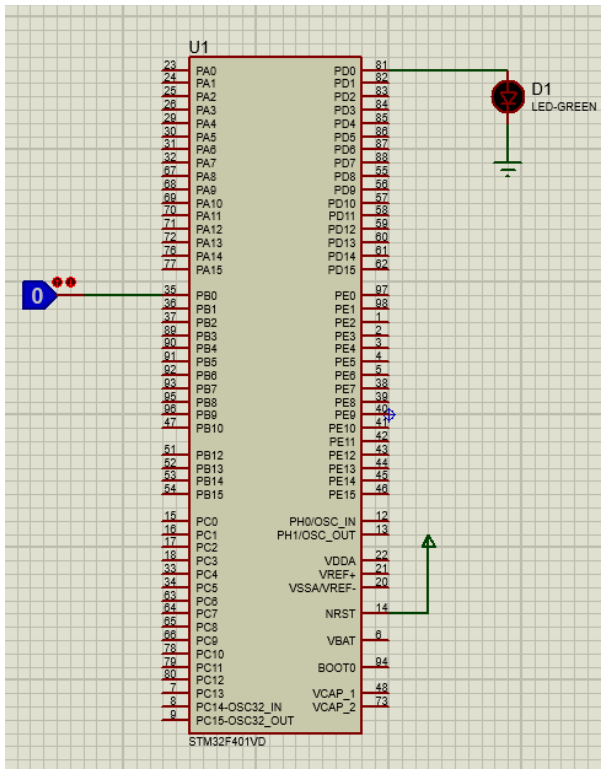
Bits 31:16 Reserved, must be kept at reset value.

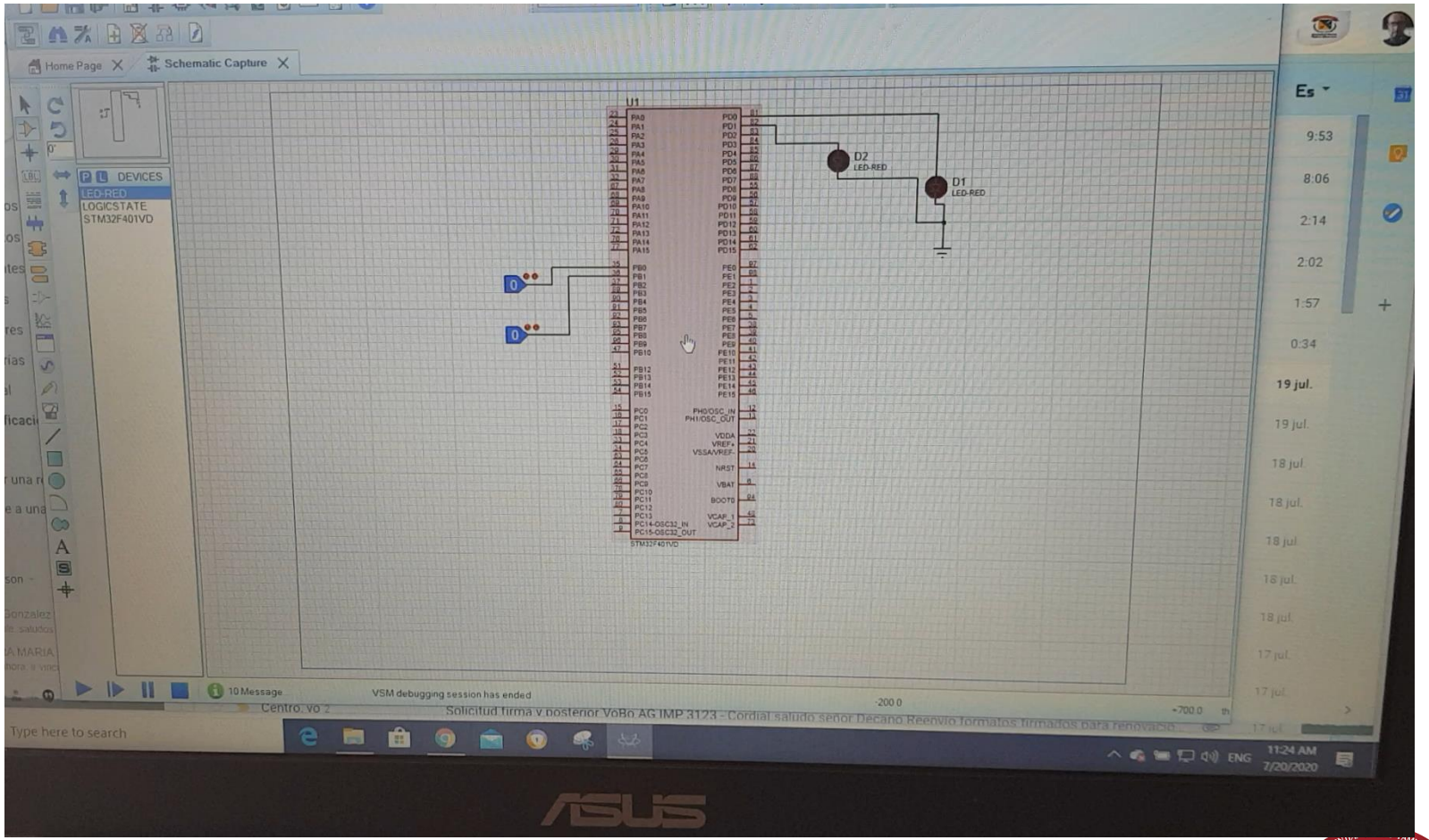
Bits 15:0 **IDRy:** Port input data bit (y = 0..15)
These bits are read-only. They contain the input value of the corresponding I/O port.

```
a= GPIOB -> IDR;
```

M32F401VD.cpp

```
#include "stm32f4xx.h"

int main(void){
  RCC -> AHB1ENR = 0XF; //LEDS AHORA
  GPIOD -> MODER = 1; //COLOCAR EN S
while(1){

  if((GPIOB -> IDR & 0x1)==1){
    GPIOD -> ODR = 1;}
  else  GPIOD -> ODR =0;
  }
}
```





6.4.2 GPIO port output type register (GPIOx_OTYPER) (x = A..K)

Address offset: 0x04

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
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
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 **OTy:** Port x configuration bits (y = 0..15)

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

0: Output push-pull (reset state)

1: Output open-drain

6.4.3 GPIO port output speed register (GPIOx_OSPEEDR) (x = A..K)

Address offset: 0x08

Reset value:

- 0x0C00 0000 for port A
- 0x0000 00C0 for port B
- 0x0000 0000 for other ports

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEEDR15 [1:0]		OSPEEDR14 [1:0]		OSPEEDR13 [1:0]		OSPEEDR12 [1:0]		OSPEEDR11 [1:0]		OSPEEDR10 [1:0]		OSPEEDR9 [1:0]		OSPEEDR8 [1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSPEEDR7 [1:0]		OSPEEDR6 [1:0]		OSPEEDR5 [1:0]		OSPEEDR4 [1:0]		OSPEEDR3 [1:0]		OSPEEDR2 [1:0]		OSPEEDR1 [1:0]		OSPEEDR0 [1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y+1:2y **OSPEEDRy[1:0]**: Port x configuration bits (y = 0..15)

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

00: Low speed
01: Medium speed
10: High speed
11: Very high speed

Note: Refer to the product datasheets for the values of OSPEEDRy bits versus V_{DD} range and external load.

DIOSPEEDRy[1:0]: Port x configuration bits (y = 0..15) 00: 2 MHz Low speed 01: 25 MHz Medium speed 10: 50 MHz Fast speed 11: 100 MHz High speed on 30 pF (80 MHz Output max speed on 15 pF)

Reset values: 0x0000 00C0 for port B 0x0000 0000 for other ports

6.4.4 GPIO port pull-up/pull-down register (GPIOx_PUPDR) (x = A..K)

Address offset: 0x0C

Reset values:

- 0x6400 0000 for port A
- 0x0000 0100 for port B
- 0x0000 0000 for other ports

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y+1:2y **PUPDRy[1:0]:** Port x configuration bits (y = 0..15)

These bits are written by software to configure the I/O pull-up or pull-down

00: No pull-up, pull-down
01: Pull-up
10: Pull-down
11: Reserved

6.4.9 GPIO alternate function low register (GPIOx_AFRL) (x = A..K)

Address offset: 0x20

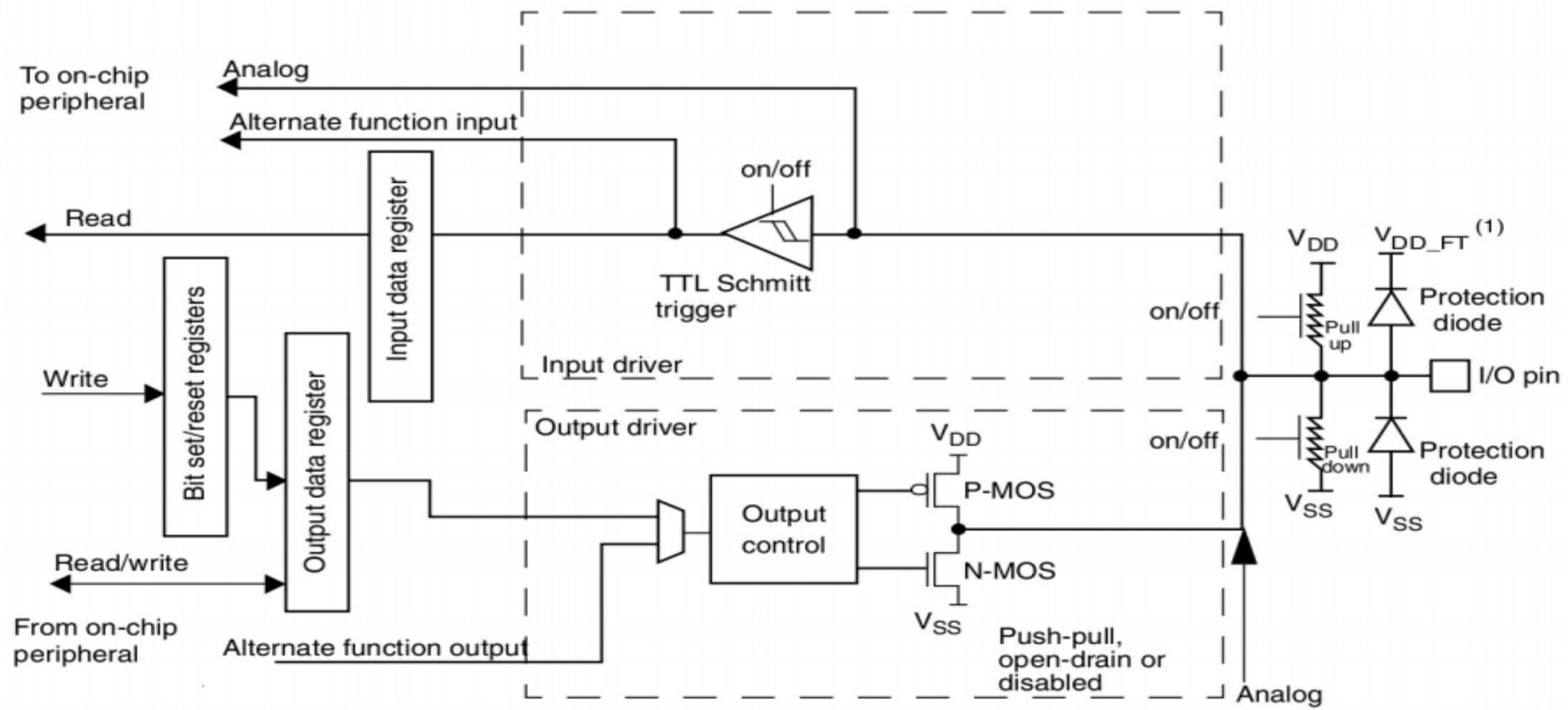
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 **AFRy[3:0]:** Alternate function selection for port x pin y (y = 0..7)
These bits are written by software to configure alternate function I/Os

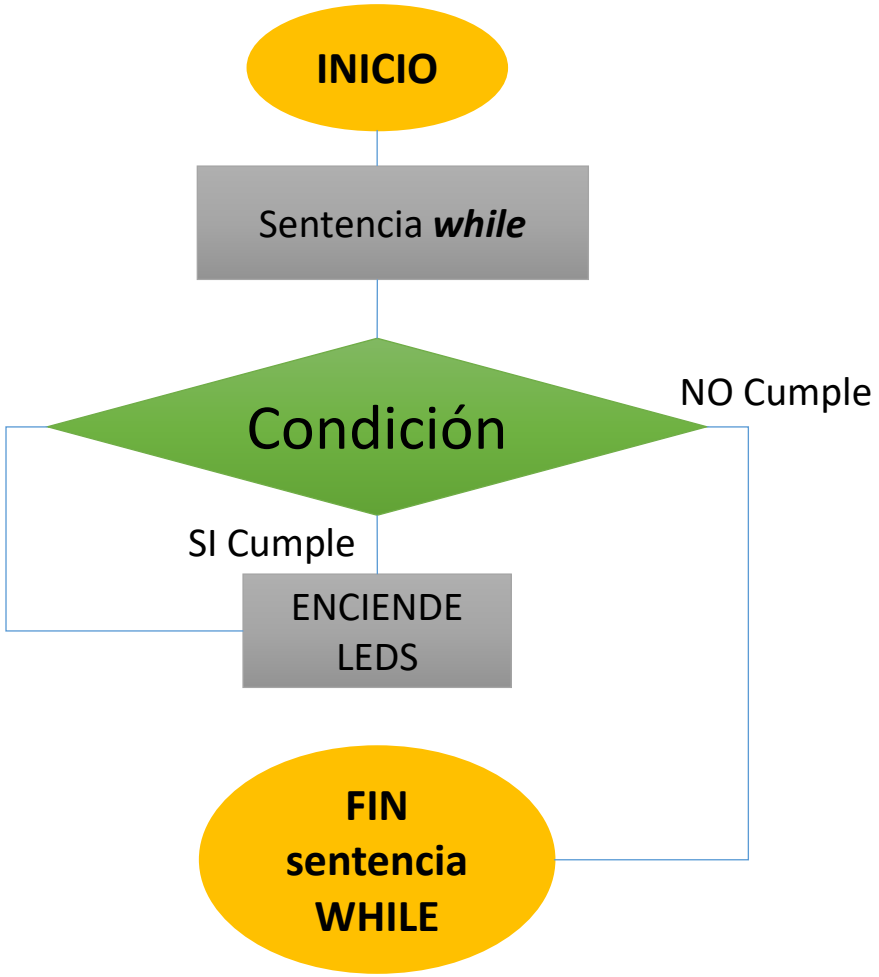
AFSELy selection:

0000: AF0
0001: AF1
0010: AF2
0011: AF3
0100: AF4
0101: AF5
0110: AF6
0111: AF7
1000: AF8
1001: AF9
1010: AF10
1011: AF11
1100: AF12
1101: AF13
1110: AF14
1111: AF15

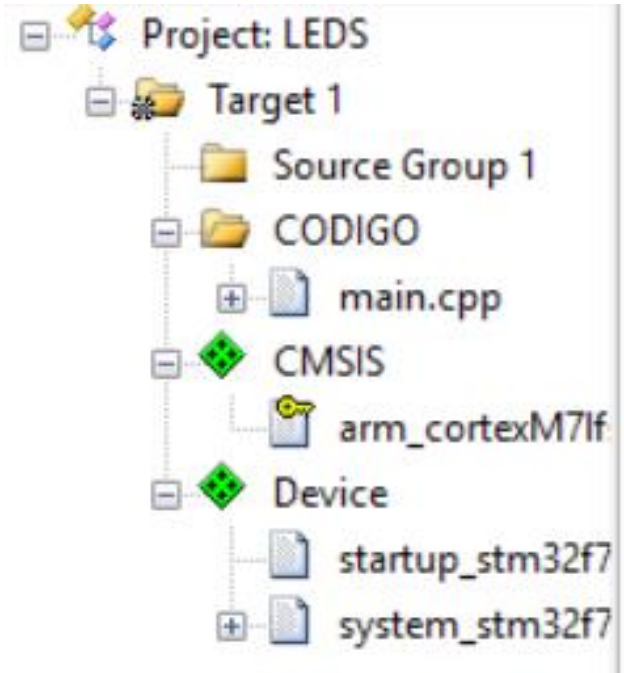

To on-chip peripheral
Analog
Alternate function input
on/off
Read
Input data register
TTL Schmitt trigger
Write
Bit set/reset registers
Output data register
Input driver
Output driver
V_{DD}
V_{DD_FT} (1)
Protection diode
I/O pin
Pull up
Pull down
V_{SS}
P-MOS
N-MOS
Output control
Read/write
From on-chip peripheral
Alternate function output
Push-pull, open-drain or disabled
Analog

Ejercicio: analizar el siguiente código y explicarlo

```
////////////////////////////////////
#include "stm32f7xx.h"
int x,y,z;
int main(void){
  RCC -> AHB1ENR = 1; //Pto A activo.
  while (1){
  if ((GPIOA->IDR&0X01)==1 && y==0){
  y=1;x=0;
   }
  if ((GPIOA->IDR&0X01)==1 && x==0){
  y=0;x=1;
   }
  }
}
```




```
#include "stm32f7xx.h"
int x,y;
int main(void){
   RCC -> AHB1ENR = 9; //Pto A y D activos.
   while (1){
   if ((GPIOA->IDR&0X01)==1 && y==0){
   y=1;
   GPIOD->ODR=~GPIOD->ODR;
    }
   if ((GPIOA->IDR&0X02)==2 && y==1){
   y=0;
    }
   }
}
```

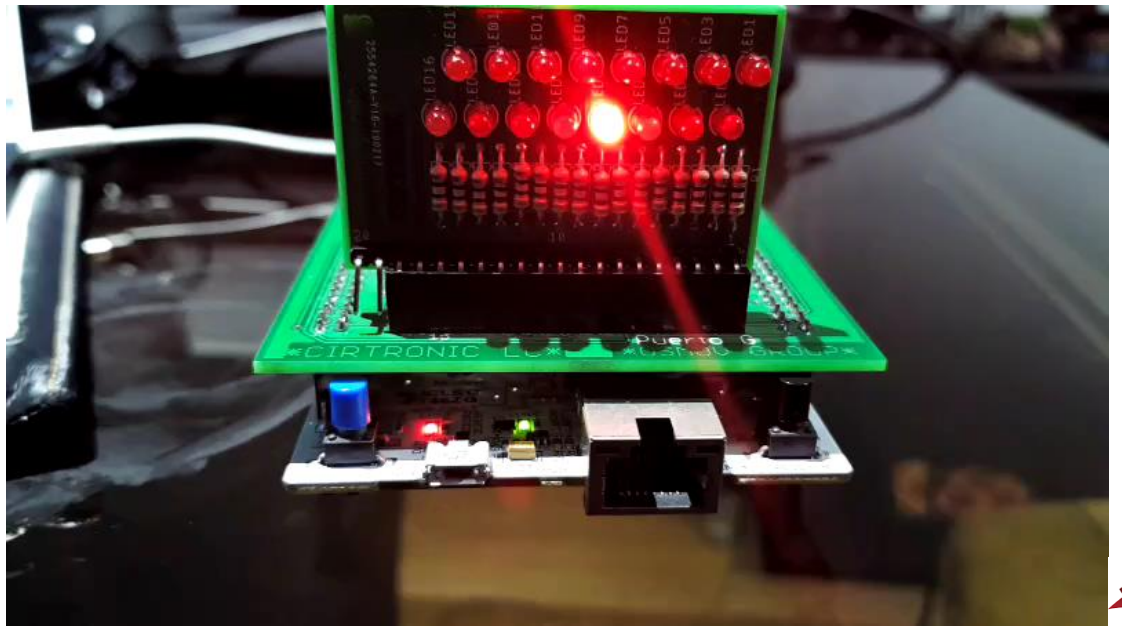



```
1  #include "stm32f7xx.h"
2
3
4  int time=50000;
5
6  int main (void)
7  {
8  int i=0;
9
10   RCC->AHB1ENR |=(1UL<<1);
11   GPIOB->MODER=0X10004001;
12   GPIOB->OTYPER=0;
13   GPIOB->OSPEEDR=0X10004001;
14   GPIOB->PUPDR=0X10004001;
15
16   while(true){
17   GPIOB->ODR |=(1UL<<14);
18   for(i=0;i<time;i++);
19     GPIOB->ODR &= ~(1UL<<14);
20   for(i=0;i<time;i++);
21   }
22
23 }
```

```
//////////////SECUENCIA LEDS  2//////////////

#include <stdio.h>
#include "stm32f7xx.h"

int time=0;
int sec[16]={1,2,4,8,16,32,64,128,256,512,1024,2048,4096,8192,16384,32768};

int main(void){
  RCC -> AHB1ENR = 0xff; //LEDS AHORA EN EL PUERTO B
  GPIOG -> MODER = 0X55555555; //COLOCAR EN SALIDA PARA ENCENDER LOS LEDS
  GPIOG -> OSPEEDR = 1; //VELOCIDAD MEDIA
  GPIOG -> PUPDR = 0; // NO PULL UP
  while(1){

    for(int a=0;a<17;a++){
    GPIOG -> ODR = sec[a];  //
    for(time=0;time<50000;time++);}
    }
    }
```

Ejercicio:

Reproducir la secuencia del video.



TAREA:

Hacer un análisis de un articulo del IEEE Xplore (base de datos de la universidad) en donde se evidencie el uso de microcontroladores, subir tanto el artículo como el análisis en formato IEEE (ambos pdf) al aula virtual,. El link es https://stadiumm.unimilitar.edu.co/ dar click en la letra I.