



Entorno de programación educativo en lenguaje Python para la EDU-CIAA-NXP

- **Autor:** Ing. Ernesto Gigliotti. UTN-FRA
- **Director:** Esp. Ing. Eric Pernia
- **Jurados:**
 - Dr. Ing. Pablo Gomez
 - Ing. Alejandro Permingeat
 - Esp. Ing. Pablo Ridolfi



**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

INTRODUCCIÓN



Dificultades en la enseñanza de programación

- **Lenguaje elegido**
- **Sintaxis**
- **Tipos de datos**
- **Sentencias condicionales**
- **Bucles**
- **Referencias, punteros**
- **IDE**



Dificultades en la enseñanza de sistemas embebidos



Dificultades en la enseñanza de sistemas embebidos

Debugger

Registers

Makefile

UART

Drivers

GPIOs

GDB OpenOCD

Stack

Cross-compiling

Interrupts

GCC ASM ISP

Von Neumann

Programmer

Harvard

JTAG

FLASH

RAM

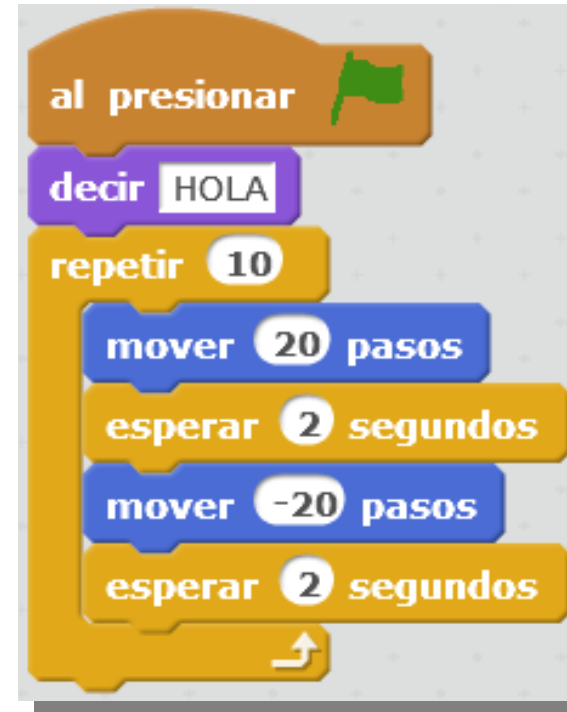




Herramientas



Herramientas



Scratch



Herramientas

**LEGO
WeDo**



MakeBlock



Scratch



```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
```

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is
  delay(1000);                     // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by makin
  delay(1000);                     // wait for a second
}
```

Herramientas



C/C++ simplificado



LEGO
WeDo



MakeBlock



Scratch



Plataforma educativa propuesta en este trabajo



Lenguaje: Python

- Sintaxis simple y clara
- Ideal como primer lenguaje
- Adoptado por muchas universidades





Lenguaje: Python

- Sintaxis simple y clara
- Ideal como primer lenguaje
- Adoptado por muchas universidades



```
while True:  
    led.on()  
    pyb.delay(500)  
    led.off()  
    pyb.delay(500)
```



Hardware: EDU-CIAA-NXP

- Bajo costo
- Ideal como primer hardware
- Adoptado por muchas universidades
- Comunidad Proyecto CIAA





Entorno de desarrollo

```
1 #LED Example
2 import pyb
3 led1 = pyb.LED(1)
4 while True:
5     led1.on()
6     pyb.delay(1000)
7     led1.off()
8     pyb.delay(1000)
9 #
```

- Fácil de instalar
- Fácil de configurar
- Graba en la placa el código Python
- Snippets de código
- Terminal integrada



Documentación



- Documentación de bibliotecas
- Ejemplos



Plataforma educativa



**Placa y
firmware**

+



IDE

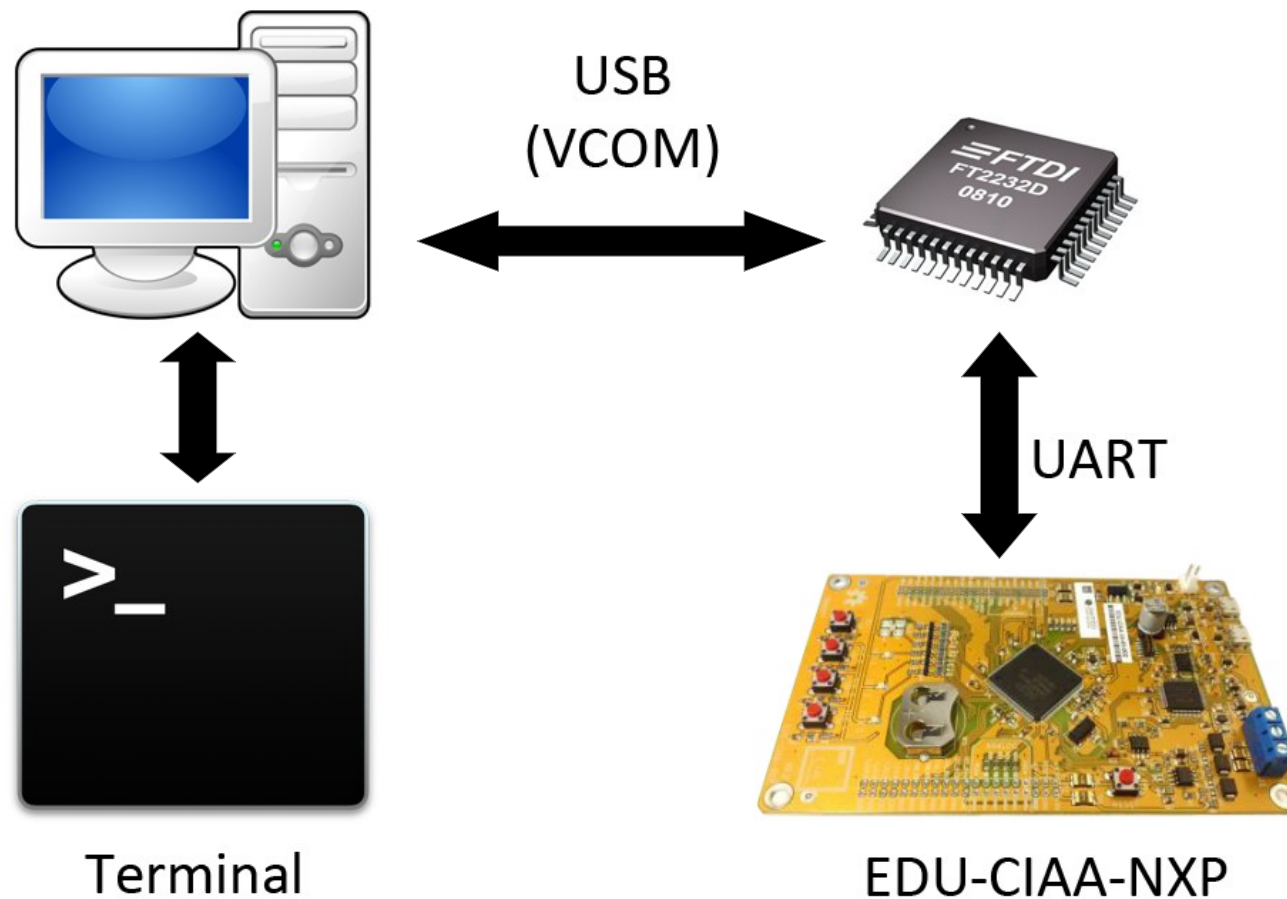
+



Ejemplos



Conexión





**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

DEMO



Punto de partida

- Port de Micropython para la EDU-CIAA-NXP^[1]:
- Intérprete.
- Garbage Collector.
- Filesystem FAT12.
- Sin soporte de periféricos.

[1] Port de micropython realizado por Martin Ribelotta. <https://github.com/martinribelotta/micropython>



Punto de partida

- Proyecto EDILE:
 - Open Source.
 - Procesador de texto.
 - Syntax highlight.
 - Python.



Requerimientos

- Manejo de hardware desde Python:
 - Leds que dispone la placa.
 - Pulsadores.
 - GPIO.
 - UART.
 - Interface RS485.
 - Entradas ADC.
 - Salida DAC.
 - La EEPROM interna.
 - Timers.



Requerimientos

- Entorno de desarrollo:
 - Multiplataforma.
 - Instalación simple.
 - No cambiar firmware de la placa.
 - Comunicación por USB.
 - Terminal serie.
 - Snippets.
 - Syntax highlight.
 - 1 archivo con script de python.



Requerimientos

- **Proyectos de ejemplo:**
 - **Inicial.**
 - **Intermedio.**
 - **Avanzado.**
- **Explicaciones detalladas.**
- **Documentación de las bibliotecas.**



**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

DISEÑO E IMPLEMENTACIÓN



Arquitectura Firmware

Script Python programado por el usuario

Intérprete
uPython

Módulos Python
(Fronzen)

Módulos Python
de periféricos

uPython HAL

Board Support Package

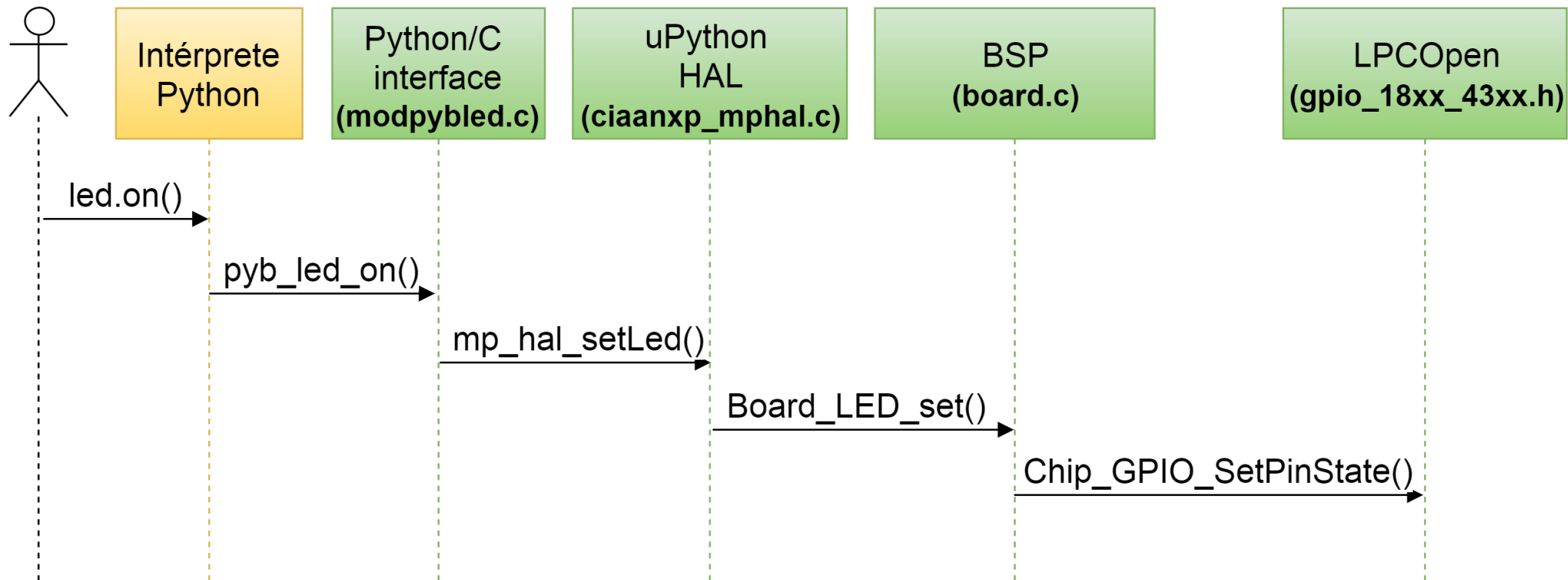
EDU-CIAA NXP

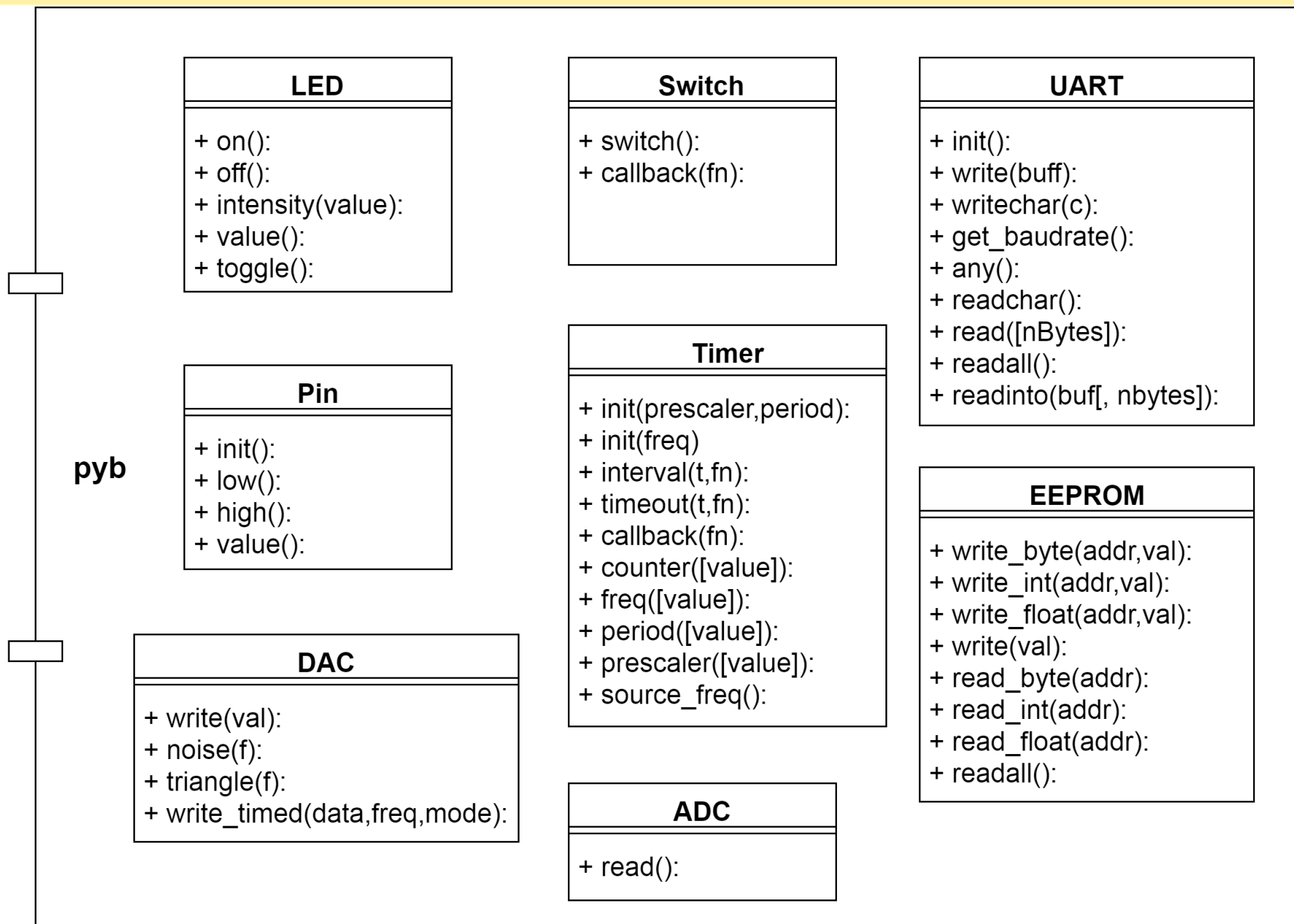


```
import pyb
```

```
led = pyb.LED(1)
```

```
led.on()
```







Arquitectura IDE

Archivo .py donde se escribe el código Python

Editor de archivo .py

Sistema de Plug-ins

Menú para la
EDU-CIAA-NXP
(Configuración, Grabar, Snippets, Terminal)

Lógica del
editor
(Guardar,
Nuevo,
Guardar
como, etc.)

Ventana de
Configuración

Ventana de
Grabación

Ventana de
Snippets

Ventana
Terminal
serie

Lógica
Configuración

Lógica
Grabación

Lógica
Snippets

Lógica
Terminal



**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

ENSAYOS Y RESULTADOS



Tests Unitarios uPython HAL

mainTest.c

testsLeds.c

testsDAC.c

testsTimers.c

testsSwitches.c

testsADC.c

tests485.c

testsUart.c

testsGPIO.c

testsEEPROM.c

utest.c

utest.h



Tests Unitarios uPython HAL

mainTest.c

testsLeds.c

testsDAC.c

testsTimers.c

testsSwitches.c

testsADC.c

tests485.c

testsUart.c

testsGPIO.c

testsEEPROM.c

utest.c

utest.h

58



Tests Unitarios clases Python

TestCase

+ testCounter
+ testOKCounter

+ setUp():
+ tearDown():
+ assertTrue(v,msg):
+ assertFalse(v,msg):
+ assertEquals(v1,v2,msg):
+ assertNotEqual(v1,v2,msg):
+ assertIsNone(v,msg):
+ assertIsNotNone(v,msg):
+ assertIsInstance(obj,cls,msg):
+ assertIsNotInstance(obj,cls,msg):
+ assertGT(v1,v2,msg):
+ run(obj):
+ printStatistics():

TestException

- **TestLeds**
- **TestSwitches**
- **TestUart**
- **TestEEPROM**
- **TestDAC**
- **TestADC**
- **TestGPIO**
- **TestRS485**
- **TestTimers**



TestXXX

+ test_1():
+ test_2():
+ test_n():



Tests Unitarios clases Python

TestCase

+ testCounter
+ testOKCounter

+ setUp():
+ tearDown():
+ assertTrue(v,msg):
+ assertFalse(v,msg):
+ assertEquals(v1,v2,msg):
+ assertNotEqual(v1,v2,msg):
+ assertIsNone(v,msg):
+ assertIsNotNone(v,msg):
+ assertIsInstance(obj,cls,msg):
+ assertIsNotInstance(obj,cls,msg):
+ assertGT(v1,v2,msg):
+ run(obj):
+ printStatistics():

TestException

69

TestXXX

+ test_1():
+ test_2():
+ test_n():

- TestLeds
- TestSwitches
- TestUart
- TestEEPROM
- TestDAC
- TestADC
- TestGPIO
- TestRS485
- TestTimers



**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

Tests funcionales



Tests funcionales

- Clases Python



Tests funcionales

- Clases Python
- Uso del IDE



Tests funcionales

- Clases Python
- Uso del IDE
- Matriz trazabilidad



CONCLUSIONES



**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

Pasos a seguir



Pasos a seguir

- **Interrupciones**
- **PWM**
- **Keyboard y LCD**
- **SPI**
- **I2C**
- **RTC**



Pasos a seguir

- **Interrupciones**
- **PWM**
- **Keyboard y LCD**
- **SPI**
- **I2C**
- **RTC**

- **Modbus**
- **time**
- **Core M0**
- **CAN**
- **Ethernet**




Pasos a seguir

- Interrupciones
- PWM
- Keyboard y LCD
- SPI
- I2C
- RTC

- Modbus
- time
- Core M0
- CAN
- Ethernet

```
1 import pyb
2
3 led = pyb.
```

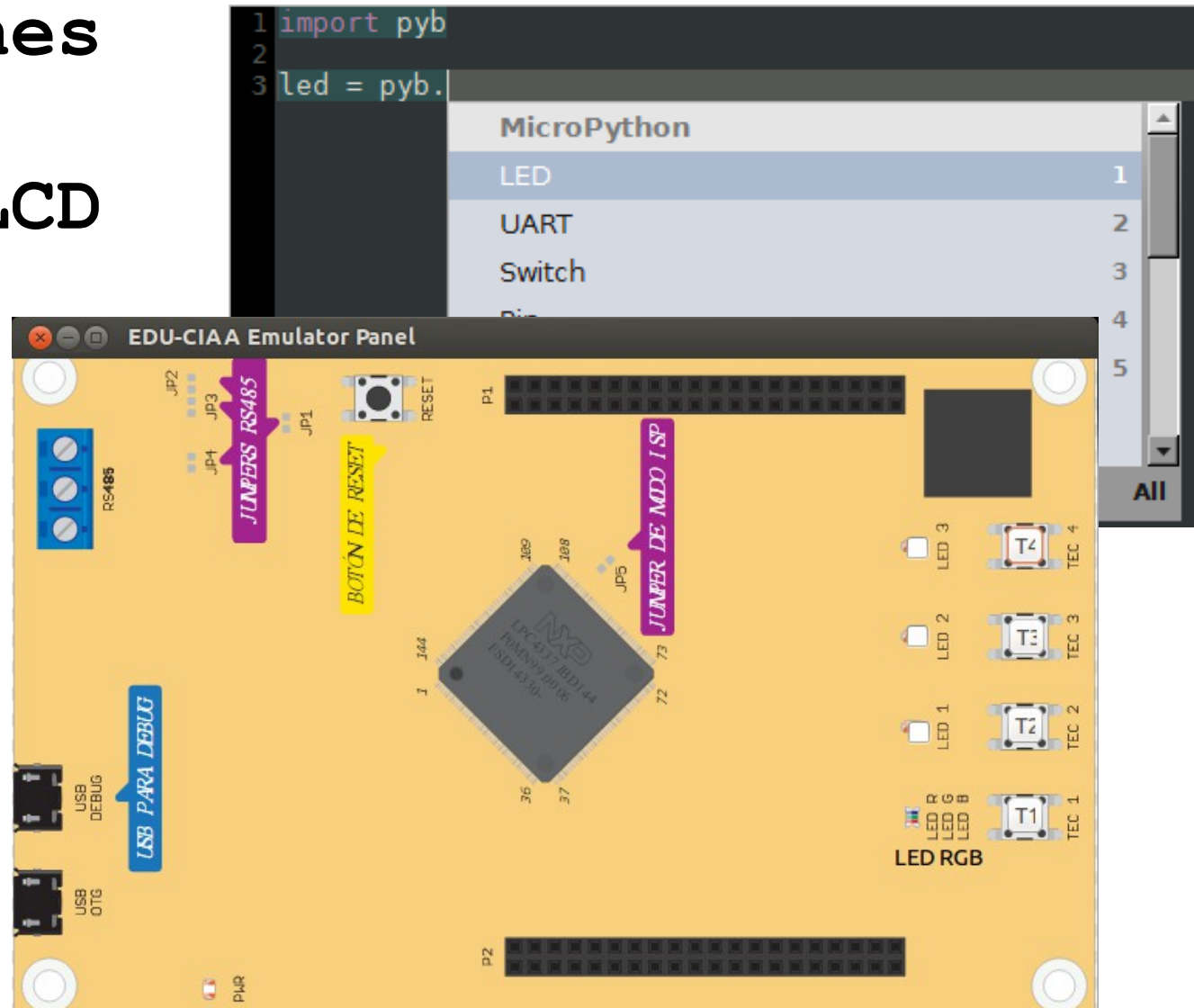




Pasos a seguir

- Interrupciones
- PWM
- Keyboard y LCD
- SPI
- I2C
- RTC

- Modbus
- time
- Core M0
- CAN
- Ethernet





**FACULTAD
DE INGENIERIA**

Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

Ing. Ernesto Gigliotti

PREGUNTAS