



COMPUTADORA INDUSTRIAL
ABIERTA
ARGENTINA



Asociación Civil para la Investigación,
Promoción y Desarrollo de los
Sistemas Electrónicos Embebidos



- Dr. Ing. Ariel Lutenberg
- Profesor en FI-UBA
- Investigador del CONICET
- Iniciador del Proyecto CIAA
- Director Maestría Embebidos UBA
- Coordinador General Simposio “SASE”
- Presidente Asoc. Civil de Sist. Emb. (ACSE)

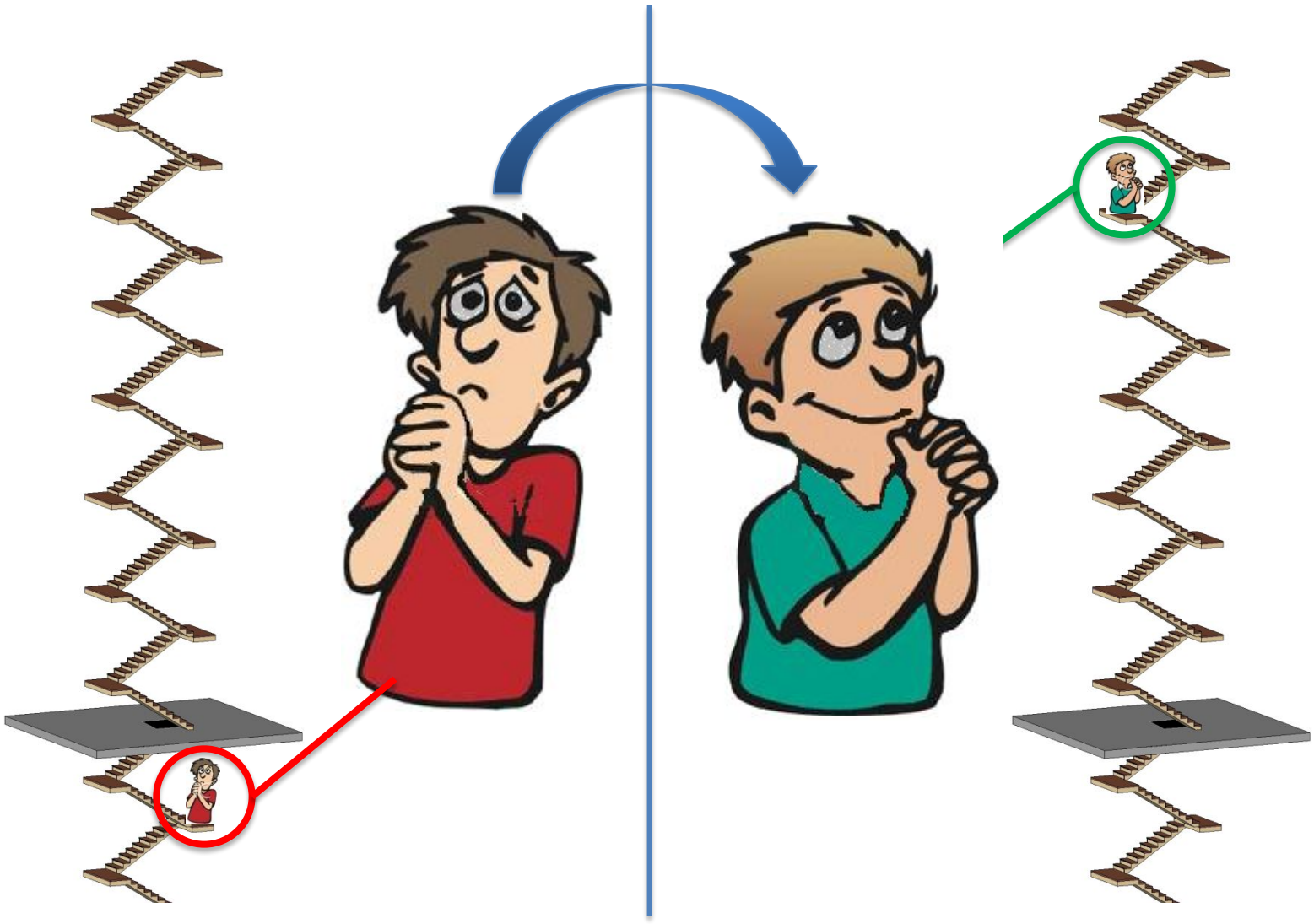


¿Cómo nació el Proyecto CIAA?

En 2013 ACSE y CADIEEL hicimos un análisis de la situación de la industria argentina:

- **No usa Electrónica**
- **Usa Electrónica importada**
- **Usa electrónica propia obsoleta**
 - Usa electrónica propia competitiva

Entonces pensamos en esta idea:



Y así surgió la idea de la “CIAA”

Computadora: que ejecute programas y resuelva problemas (útil para varios *tipos de aplicaciones*).

Computadora

Puertos de
entrada y
de salida

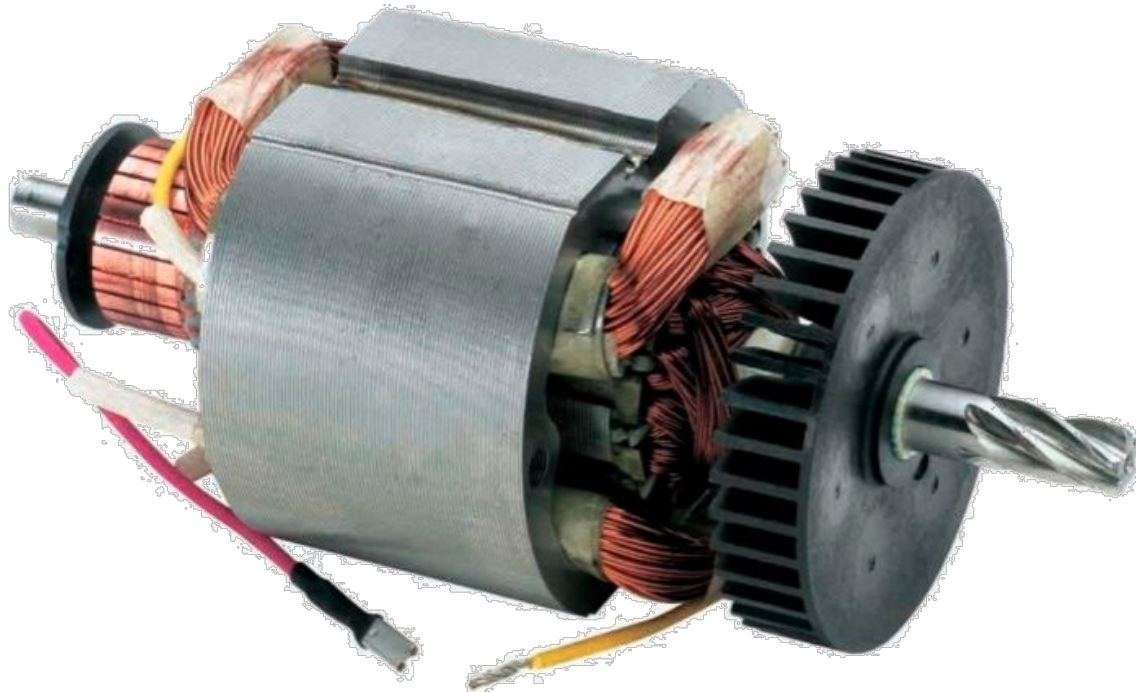
Procesador

Memorias de
datos y de
programas



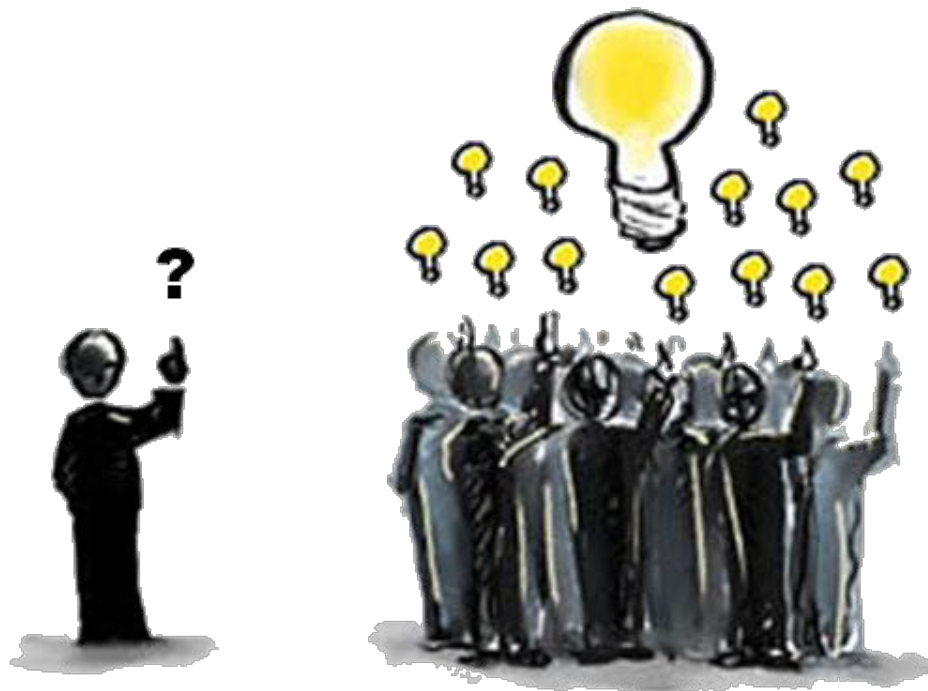
Y así surgió la idea de la “CIAA”

Que sea **Industrial**: que su diseño esté preparado para las exigencias que demandan productos y procesos industriales.



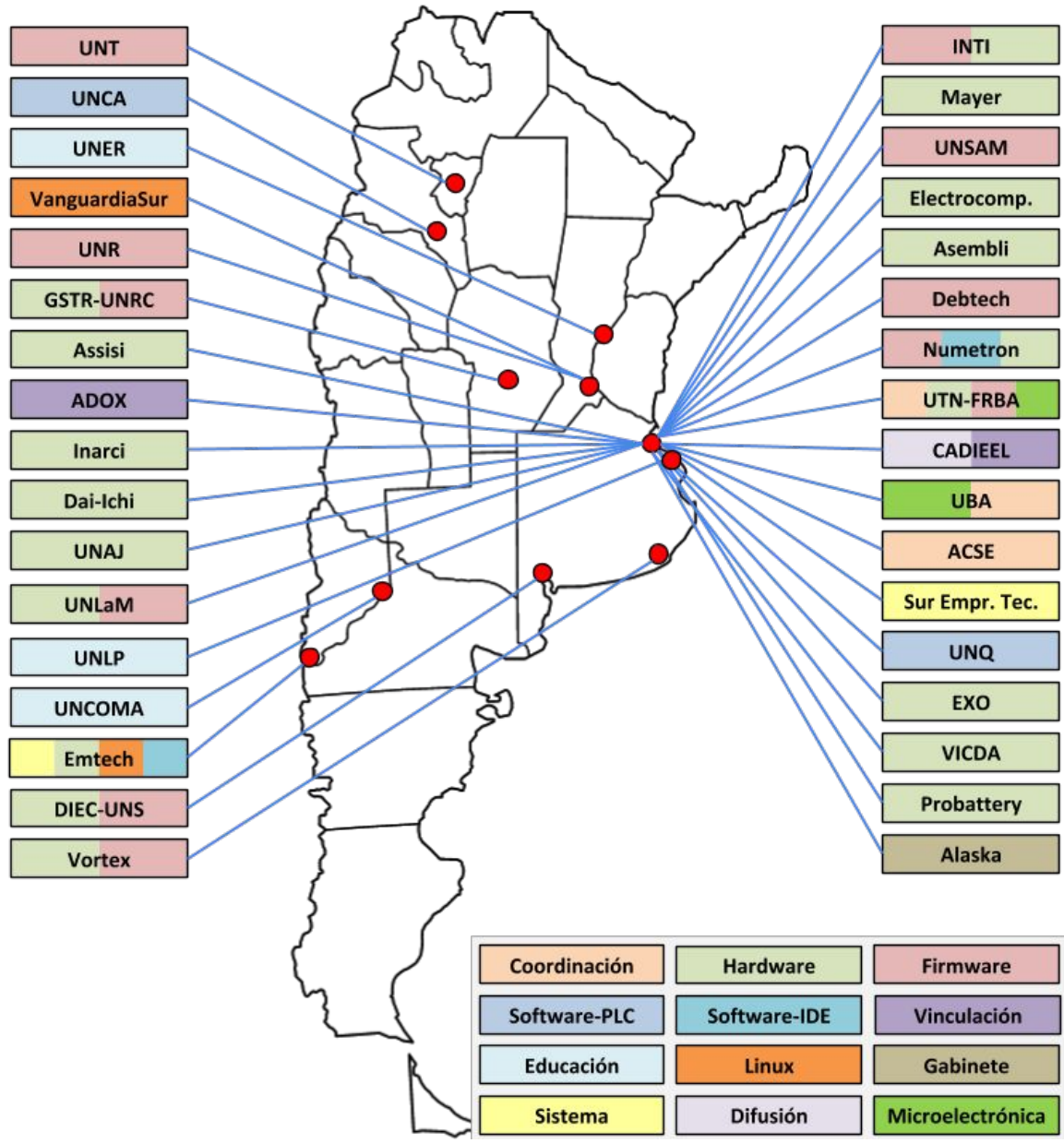
Y así surgió la idea de la “CIAA”

Que sea **Abierta**: que toda la información esté libremente disponible en internet, para que cualquiera la utilice como quiera.





Y la "A" de Argentina es porque...



1. Impulsar el desarrollo tecnológico nacional.
2. Dar visibilidad a la electrónica argentina.
3. Generar cambios estructurales en la forma de generar y utilizar los conocimientos.

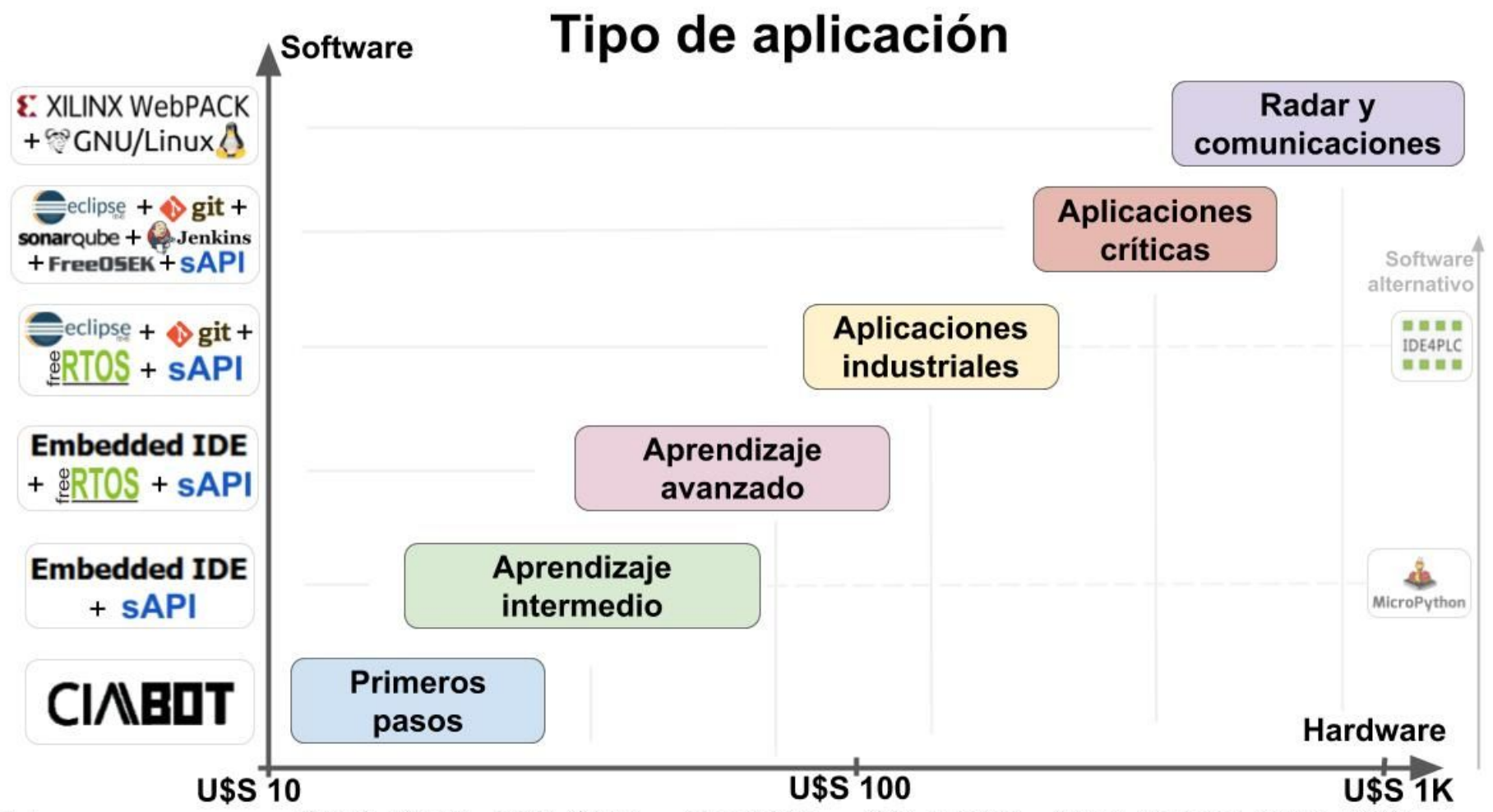


Tipo de aplicación





¿Qué es el Proyecto CIAA?



Notas:

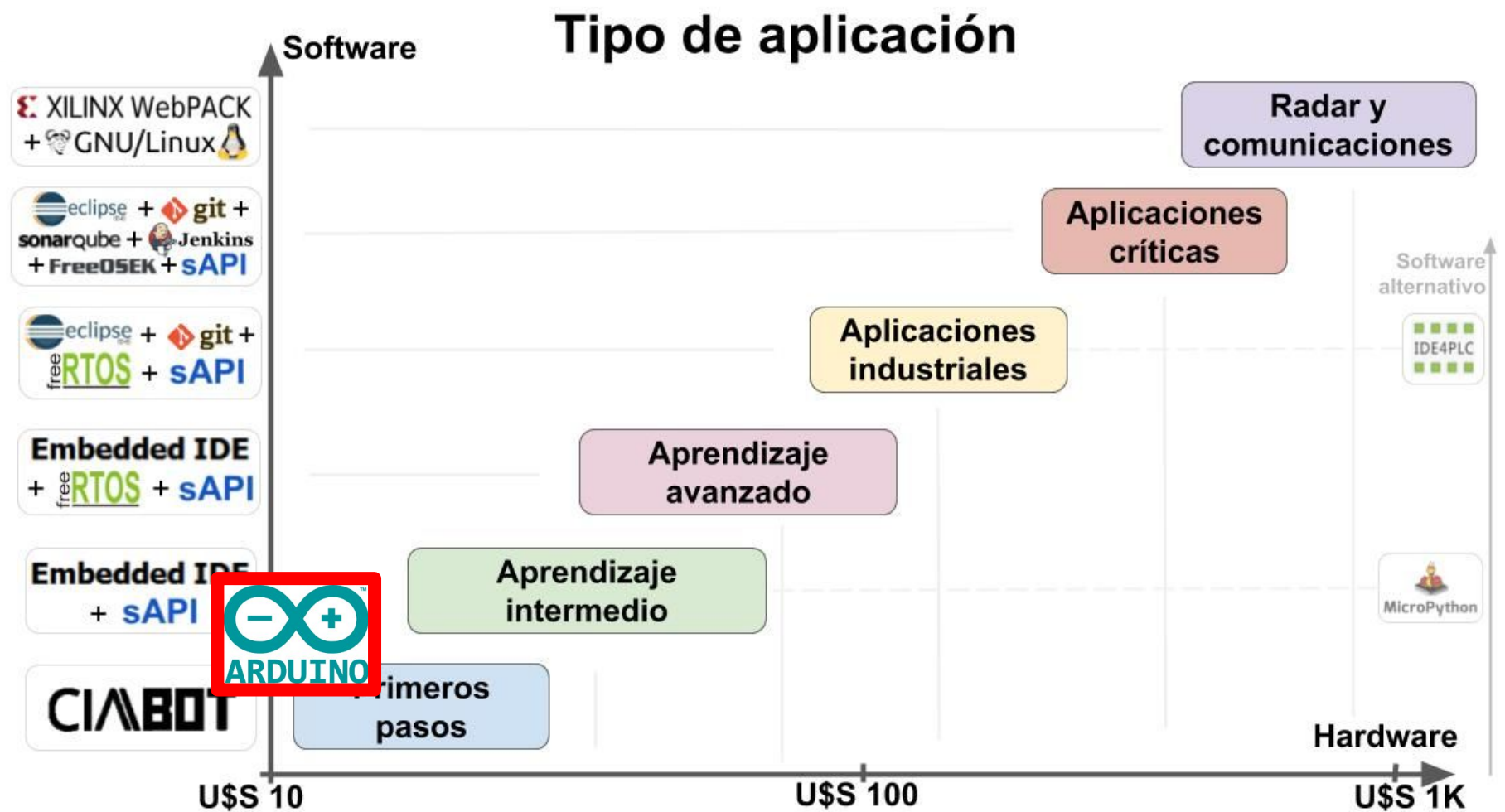
- Este diagrama es orientativo; muchas otras combinaciones de hardware y software son posibles.
- CIAABOT es un entorno de programación visual y sAPI una biblioteca de código C.

CIAA-Z3R0 EDU-CIAA picoCIAA CIAA-NXP CIAA-Safety CIAA-ACC





¿Qué es el Proyecto CIAA?



Notas:

- Este diagrama es orientativo; muchas otras combinaciones de hardware y software son posibles.
- CIAABOT es un entorno de programación visual y sAPI una biblioteca de código C.

CIAA-Z3R0 EDU-CIAA picoCIAA CIAA-NXP CIAA-Safety CIAA-ACC



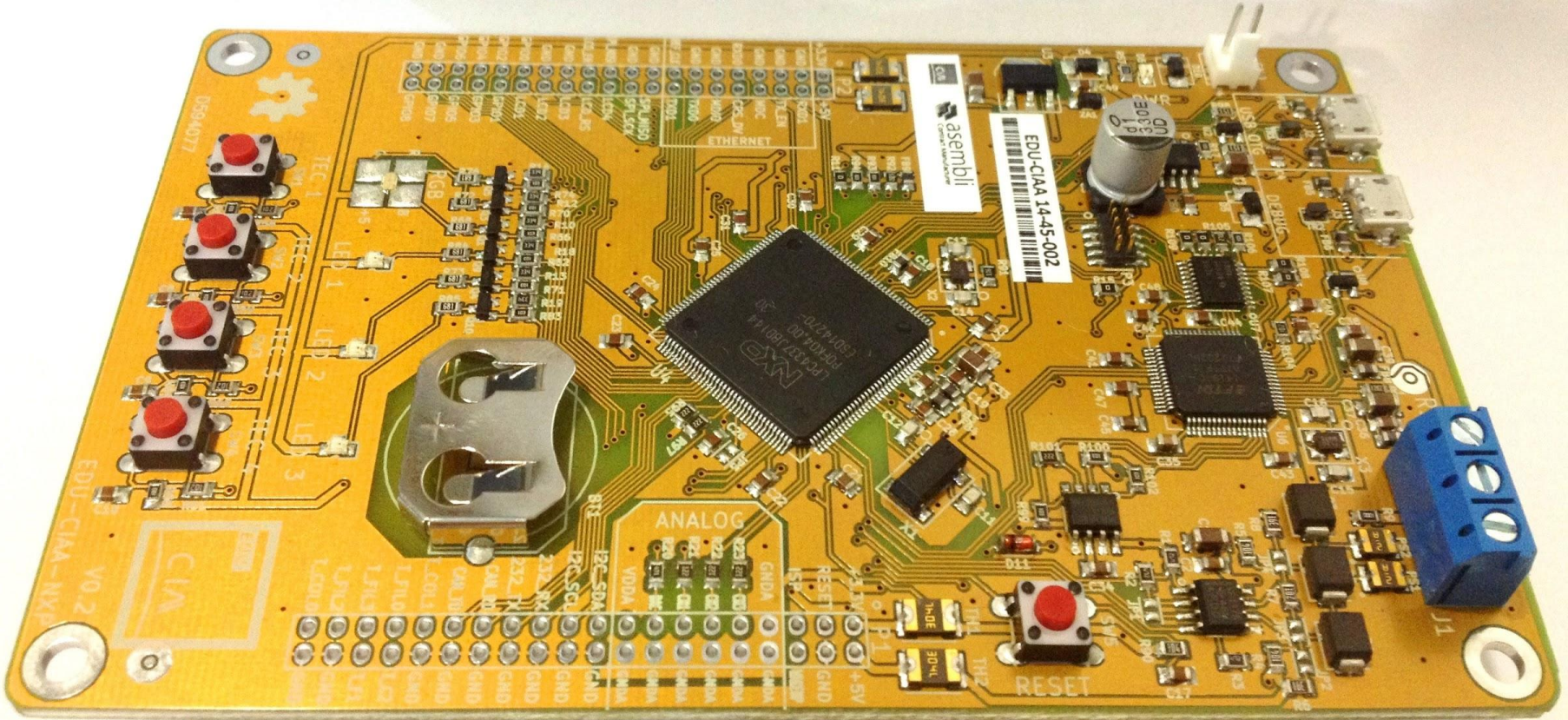


CIAA-Z3R0



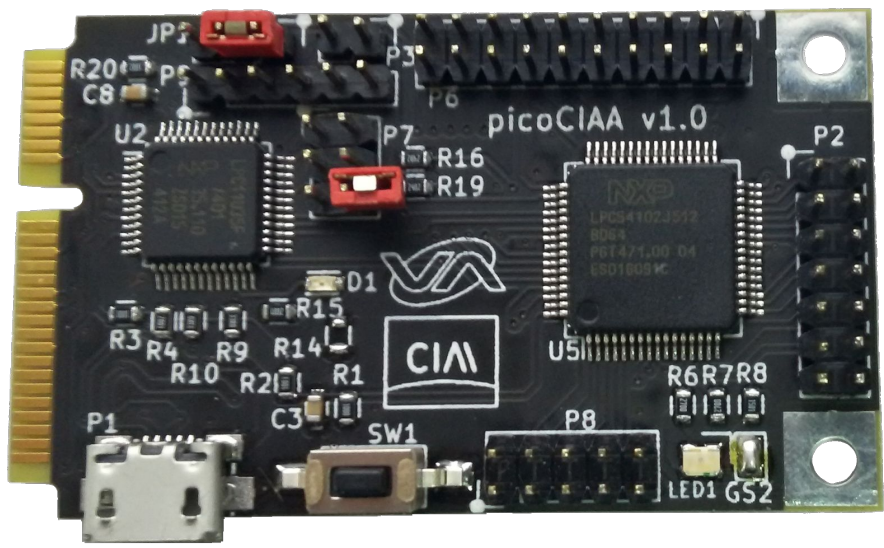


EDU-CIAA



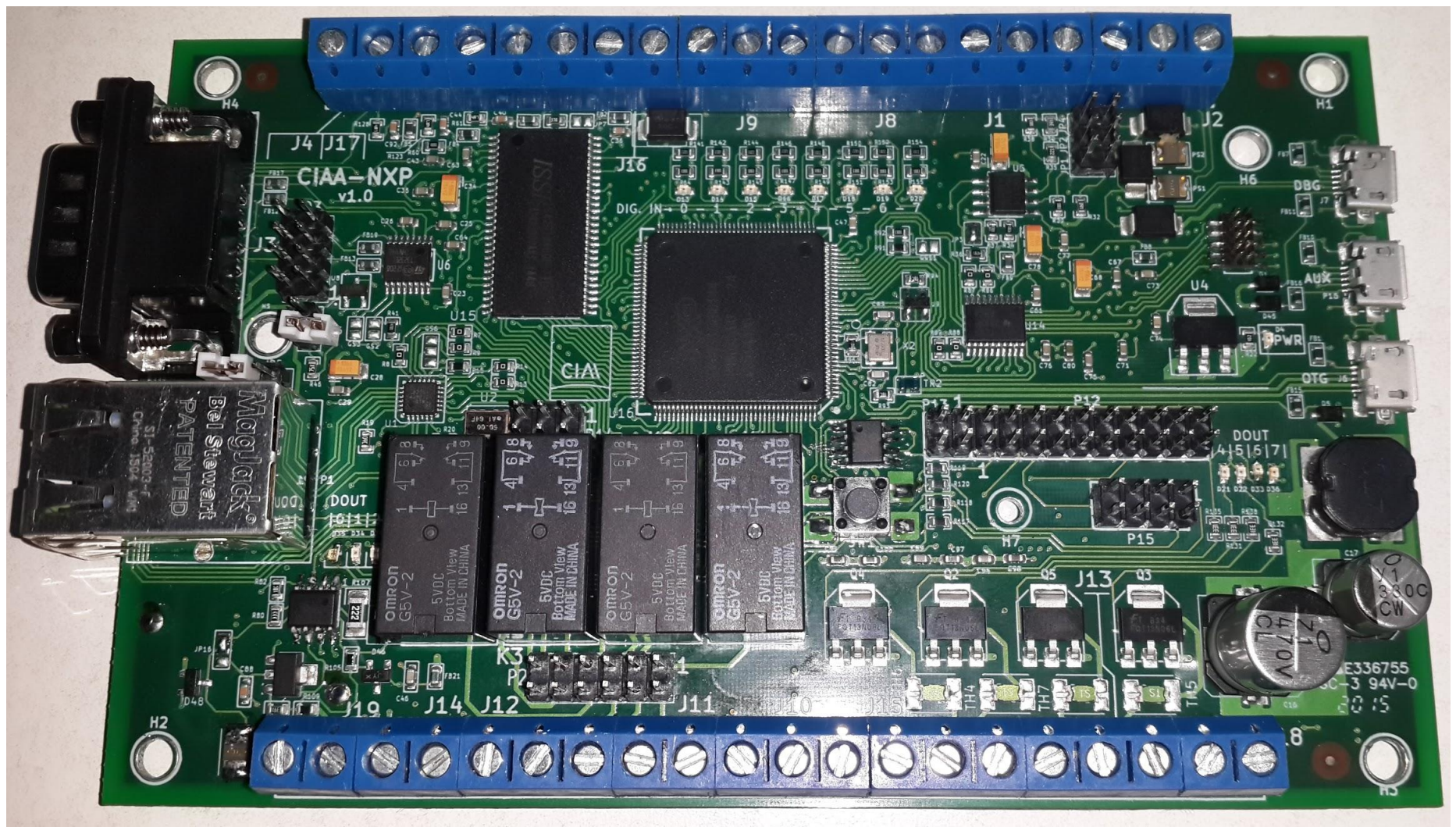


picoCIAA





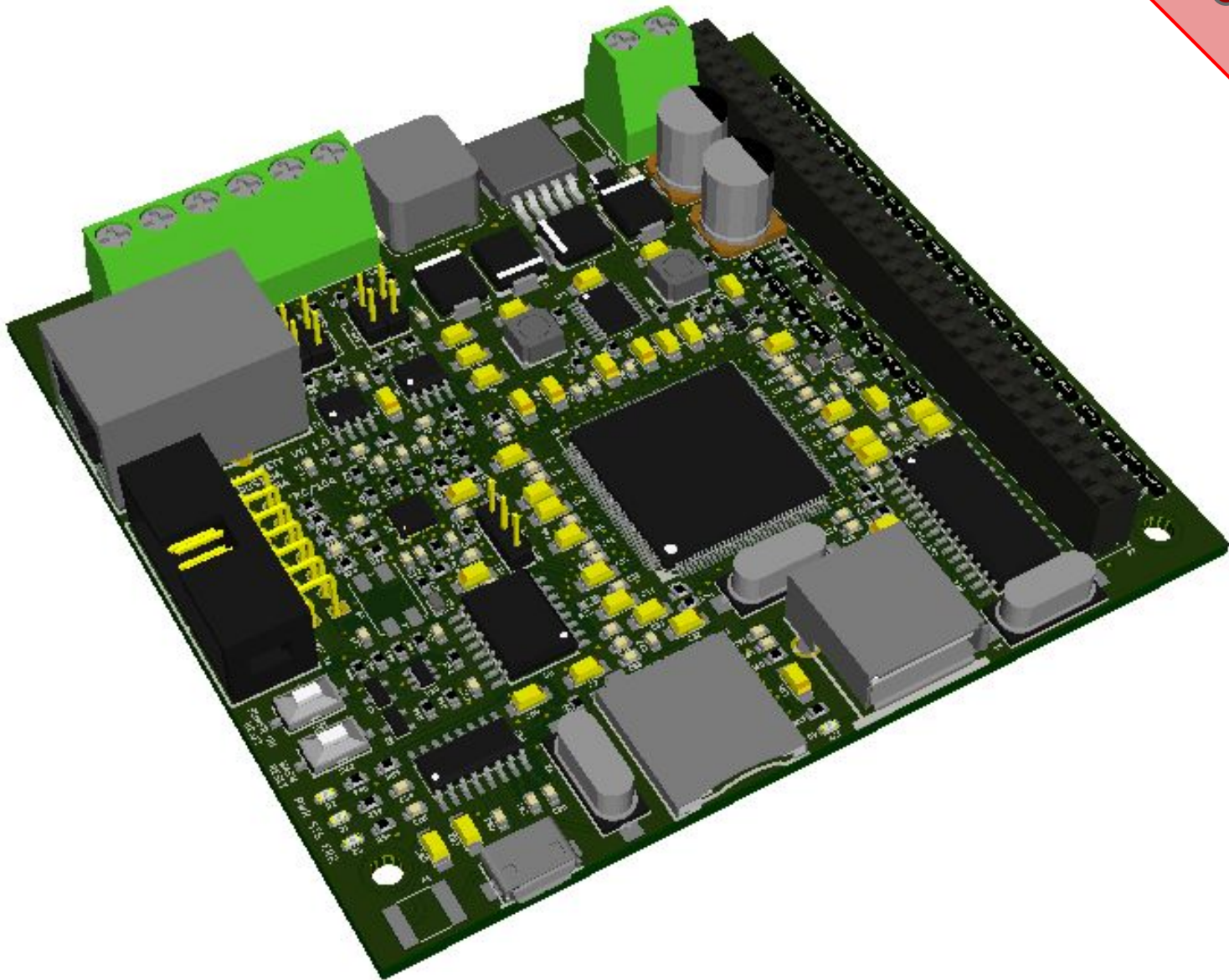
CIAA-NXP





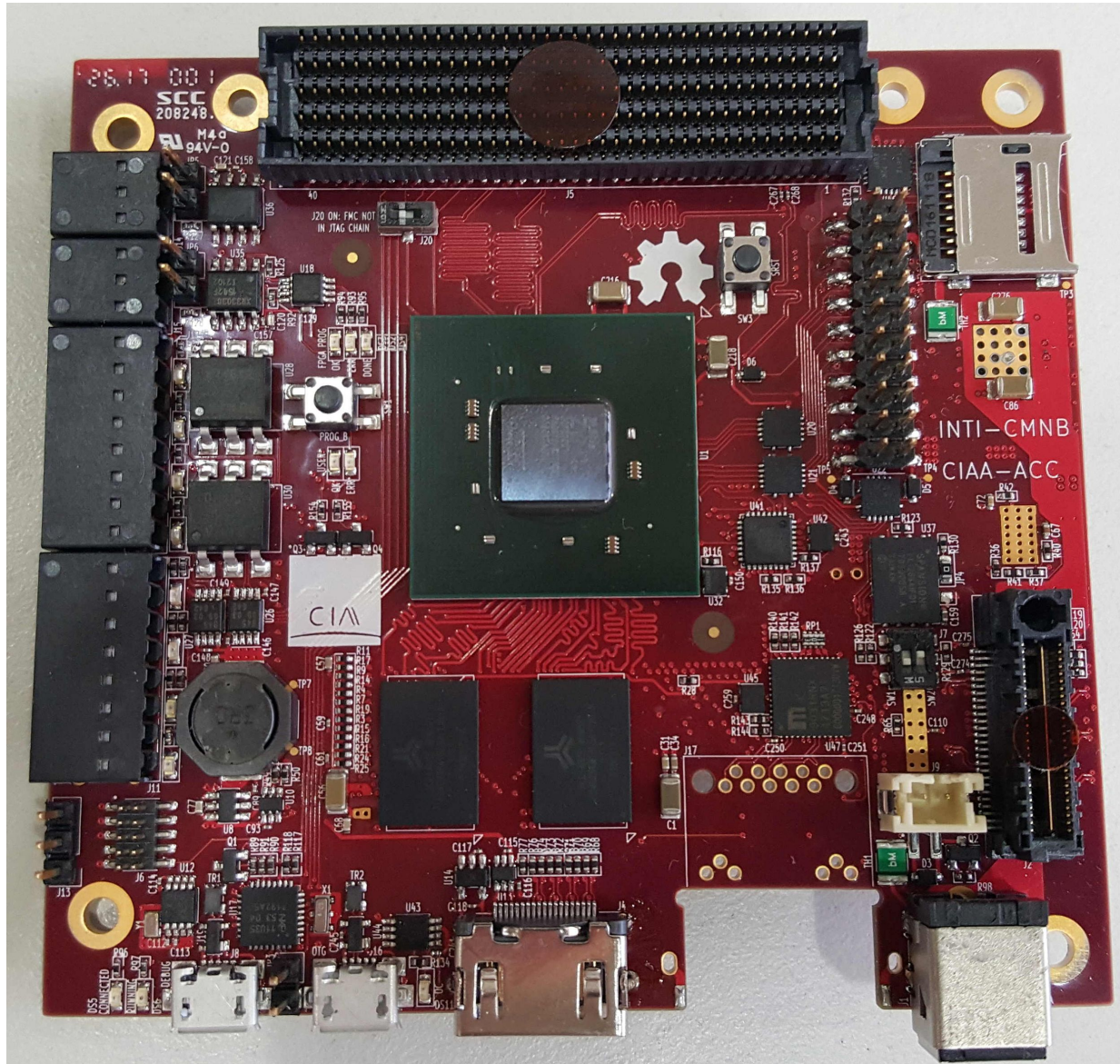
CIAA-Safety

en desarrollo



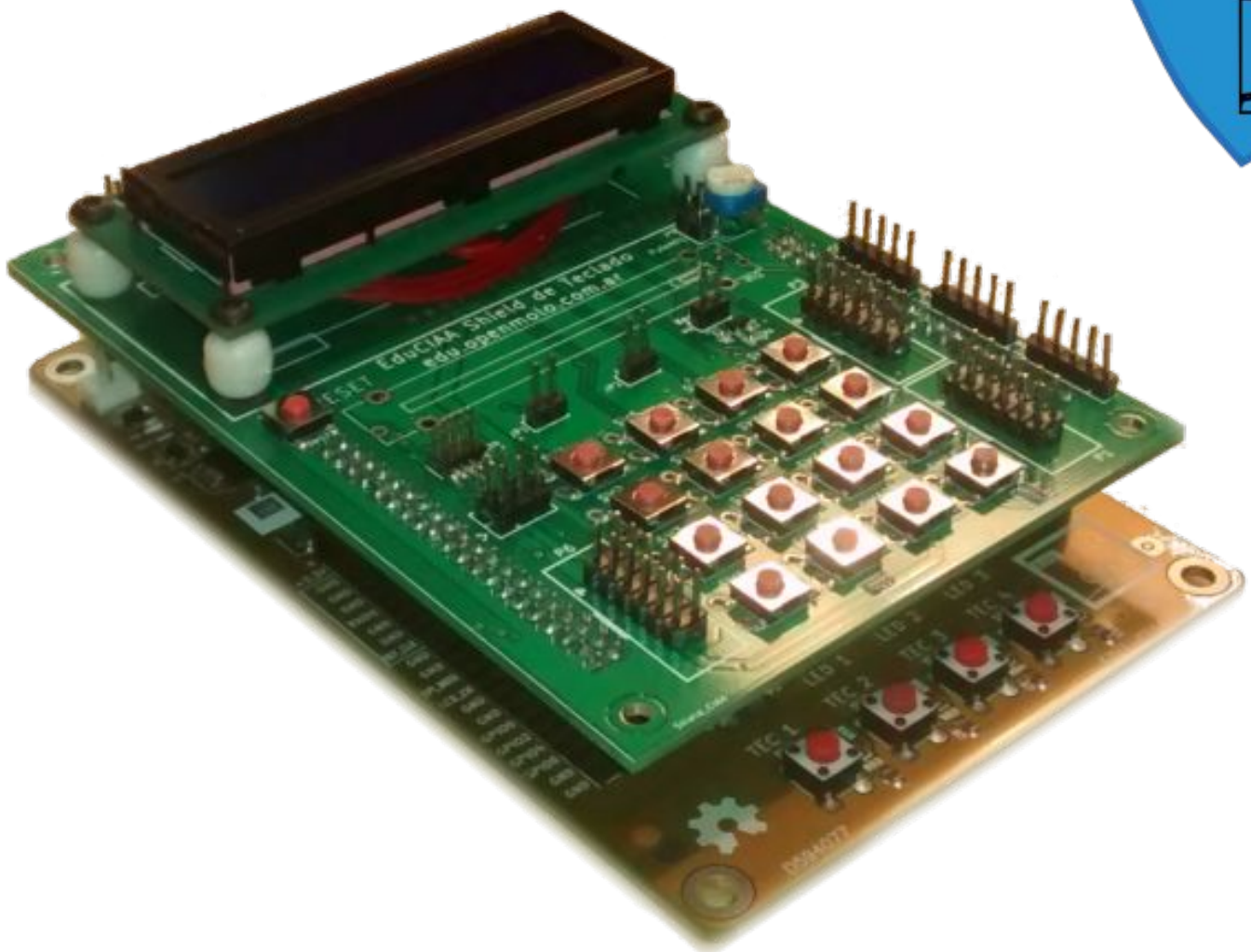


CIAA-ACC





“Ponchos” para prototipado



Ciaabot IDE

Programa compilado
Se ha compilado con éxito ✓

Programa principal

- Repetir para siempre
 - establecer **tiempoinicial** a Leer base de tiempo (ms)
 - establecer **pulsos** a 0
 - Repetir **mientras**
 - Leer base de tiempo (ms) **>** **tiempoinicial** **<** 1000
 - hacer
 - establecer **estadoAnterior** a Leer estado del GPIO Gpio 0
 - Si **Leer estado del GPIO Gpio 0** **y** no **estadoAnterior**
 - hacer
 - establecer **pulsos** a **pulsos** + 1
 - Enviar **número** por UART **pulsos** ¿Agregar Enter?

velocidad_viento * | Usando el Ciaabot G1

The screenshot displays the Embedded IDE interface. The title bar shows the path: `Embedded IDE /home/eric/ciaa-ide/embedded-ide/projects/sAPI_Blink/Makefile`.

File Explorer (Left Panel):

Nombre	Tamaño
app	
inc	
src	
programa.c	2 KB
libs	
config.mk	159 bytes
Makefile	2 KB

Objetivos (Bottom Left):

- Borrar_memoria_flash
- Compilar_proyecto
- Grabar_proyecto_en_flash
- Limpiar_Proyecto
- Uso_de_memoria

Code Editor (Center): `programa.c`

```
1 #include "sapi.h" // <= Biblioteca sAPI
2 int main( void ){
3     // ----- CONFIGURACIONES -----
4     boardConfig(); // Inicializar y configurar la plataforma
5     // ----- REPETIR POR SIEMPRE -----
6     while( TRUE )
7     {
8         // Si se presiona TEC1, enciende el LEDR
9         gpioWrite( LEDR, !(gpioRead(TEC1)) );
10        // Si se presiona TEC2, enciende el LED1
11        gpioWrite( LED1, !(gpioRead(TEC2)) );
12        // Si se presiona TEC3, enciende el LED2
13        gpioWrite( LED2, !(gpioRead(TEC3)) );
14        // Si se presiona TEC4, enciende el LED3
15        gpioWrite( LED3, !(gpioRead(TEC4)) );
16        // Intercambia el valor del LEDB
17        gpioToggle( LEDB );
18        // Retardo bloqueante durante 100ms
19        delay( 100 );
20    }
21    return 0;
22 }
```



Eclipse + git + FreeRTOS + sAPI

The screenshot displays the Eclipse IDE interface for a C/C++ project. The main editor shows the source code for `blinking_echo.c`, which includes the following code:

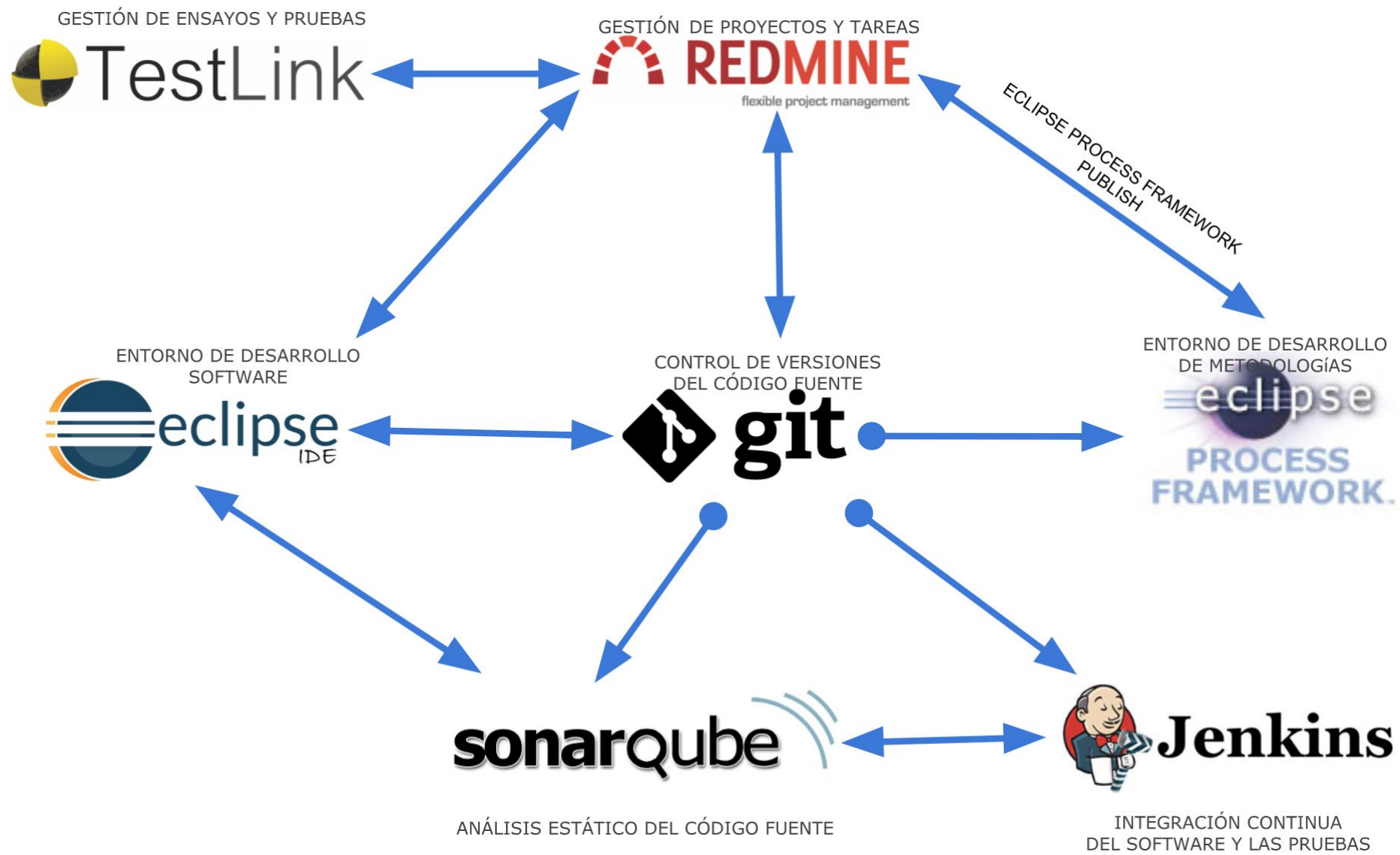
```
165 */
166 TASK(InitTask)
167 {
168     /* init CIAA kernel and devices */
169     ciaa_start();
170
171     ciaaPOSIX_printf("Init Task...\n");
172     /* open CIAA digital inputs */
173     fd_in = ciaaPOSIX_open("/dev/dio/in/0", O_RDONLY);
174
175     /* open CIAA digital outputs */
176     fd_out = ciaaPOSIX_open("/dev/dio/out/0", O_RDWR);
177
178     /* open UART connected to USB bridge (FT2232) */
179     fd_uart1 = ciaaPOSIX_open("/dev/serial/uart/1", O_RDWR);
180
181     /* open UART connected to RS232 connector */
182     fd_uart2 = ciaaPOSIX_open("/dev/serial/uart/2", O_RDWR);
183
184     /* change baud rate for uart usb */
185     ciaaPOSIX_ioctl(fd_uart1, CIAA_IOCTL_SET_BAUDRATE, (void *)CIAA_BAUDRATE_115200);
186
187     /* change FIFO TRIGGER LEVEL for uart usb */
188     ciaaPOSIX_ioctl(fd_uart1, CIAA_IOCTL_SET_FIFO_TRIGGER_LEVEL, (void *)CIAA_FIFO_TRIGGER_LEVEL3);
189 }
```

The Project Explorer on the left shows the project structure, including the `src` directory containing `adc_dac.c`. The Outline view on the right lists the project's files and symbols, such as `os.h`, `ciaaPOSIX_stdio.h`, and `main(void) : int`.

The Console view at the bottom shows the output of the build process:

```
=====  
Post Building blinking_echo  
  
arm-none-eabi-objcopy -v -O binary ./out/bin/bleeping_echo.axf ./out/bin/bleeping_echo.bin  
copy from './out/bin/bleeping_echo.axf' [elf32-littlearm] to './out/bin/bleeping_echo.bin' [binary]  
make[1]: Leaving directory '/home/pablo/ciaa-ide/Firmware'  
  
19:02:14 Build Finished (took 24s.148ms)
```

Entorno para sistemas críticos



[1] <http://www.linsse.com.ar>



IDE4PLC



Board




POU

Board

<== Choose a board

EDU-CIAA-NXP

Picture



POU Variable Declarations Editor

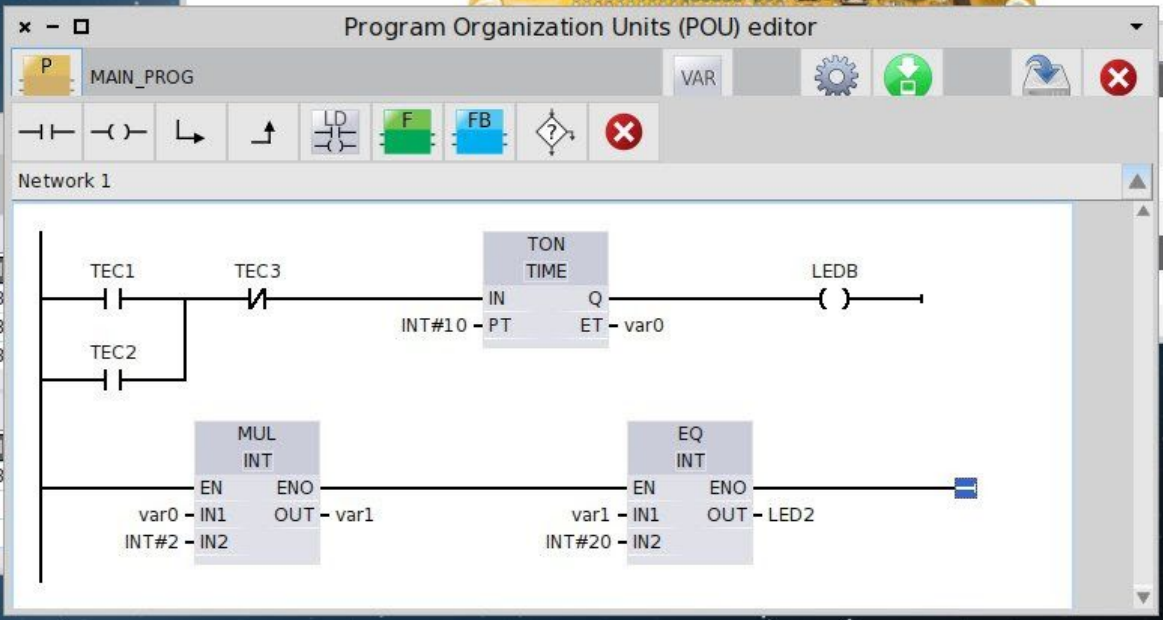
MAIN_PROG

Interface Variables

Category	Name	Datatype
VAR_INPUT	TEC1	BOOL
VAR_INPUT	TEC2	BOOL
VAR_INPUT	TEC3	BOOL

Internal and External Variables

Category	Name	Datatype
VAR	var0	BOOL
VAR	var1	INT
VAR	var2	TIME




```
Main_rgb.py | EDU-CIAA Python Editor
File      EDU-CIAA
New       Open       Save       Load Script  Terminal  Snippets

1 import pyb
2 import math
3
4 print("Test led RGB")
5
6 ledR = pyb.LED(4)
7 ledG = pyb.LED(5)
8 ledB = pyb.LED(6)
9 s0 = bytearray(100)
10 for i in range(0, len(s0)):
11     s0[i] = 8 + int(7 * math.sin(2 * math.pi * i / (len(s0)) ))
12
13 s1 = bytearray(100)
14 for i in range(0, len(s1)):
15     s1[i] = 8 + int(7 * math.sin( (1/3)*math.pi + 2 * math.pi * i / (len(s1)) ))
16
17 s2 = bytearray(100)
18 for i in range(0, len(s2)):
19     s2[i] = 8 + int(7 * math.sin( (2/3)*math.pi + 2 * math.pi * i / (len(s2)) ))
20 t=0
21 while True:
22     pyb.delay(30)
23     val = s0[t]
24     ledR.intensity(val)
25
26     val=s1[t]
27     ledG.intensity(val)
28
29     val=s2[t]
30     ledB.intensity(val)
31
32     t+=1
33     if t==100:
34         t=0
35
```

File: /home/pablo/repos/ciaa/micropython/ciaa-nxp/pyExamples/Main_rgb.py | Encoding: utf-8



C/C++ - CIAA_Uart_HVM/src/ar/edu/unq/embebidos/Main.java - Eclipse Platform

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Resource Java SVN Repository Exploring C/C++ Debug

Project Explorer

- > CIAA_Blinky_HVM
- > CIAA_HolaMundoHVM
- > CIAA_SCJ_Process_HVM
- > CIAA_TecsLeds_HVM
- ▼ CIAA_Uart_HVM
 - ▼ src
 - ▼ ar.edu.unq.embebidos
 - > Led.java
 - > Main.java
 - > Pulsador.java
 - > Uart.java
 - > JRE System Library
 - > icecapSDK.jar
 - > HVM_C_output_files
 - > PC_HVM_FullSCJ
 - > PC_HVM_ScheculerSCJ
 - > PC_HVM_ScheculerSCJ_2

Main.java

```
package ar.edu.unq.embebidos;

public class Main {
    public static void main(String[] args) {

        // Instanciación de los objetos Led
        Led ledR = new Led(0);
        Led led1 = new Led(4);

        // Instanciación de los objetos Pulsador
        Pulsador tec1 = new Pulsador(0);
        Pulsador tec2 = new Pulsador(1);

        // Instanciación del objetos Uart
        Uart serialPort = new Uart();

        // Se configura la UART2
        serialPort.config();
    }
}
```

Problems Tasks Console Properties

No consoles to display at this time.

ar.edu.unq.embebidos.Main.java - CIAA_Uart_HVM/src

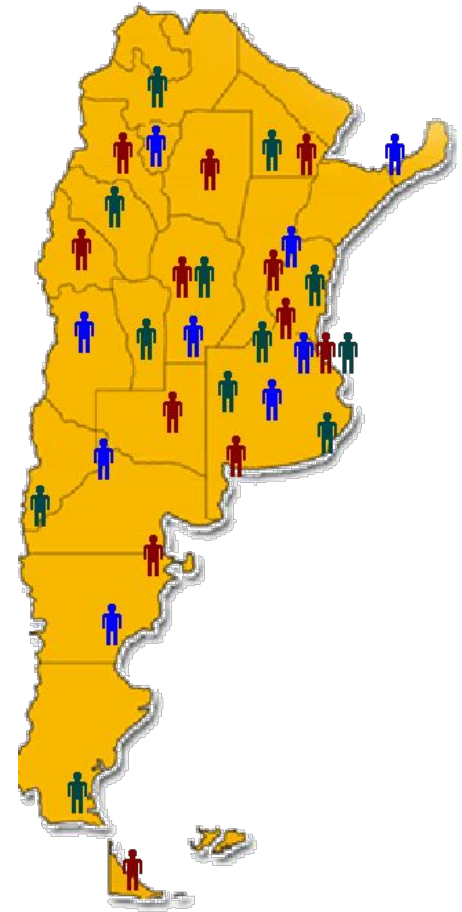
¡Hay miles de EDU-CIAAs en uso en todo el país!

RUSE

Red Universitaria de
Sistemas Embebidos

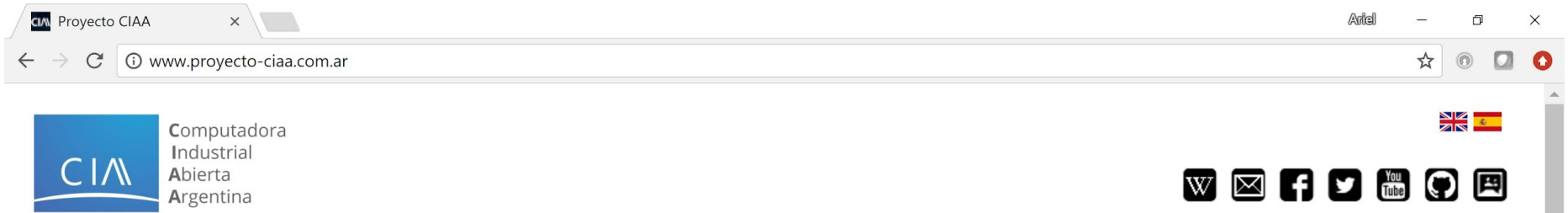


Asociación Civil para la Investigación,
Promoción y Desarrollo de los
Sistemas Electrónicos Embebidos

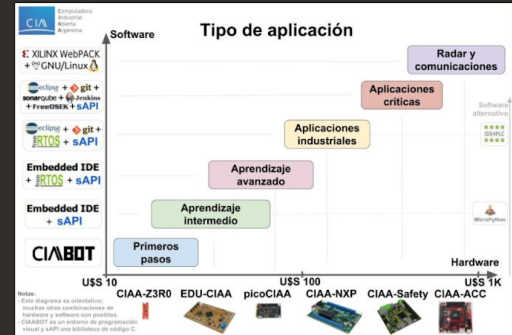




Página web



Inicio



¿Qué es el proyecto CIAA?

¿Qué es una "Computadora"? ¿Qué significa "Industrial" y "Abierta"? ¿Qué busca este Proyecto? **Ejemplos de uso.**

¿Cómo comprar una CIAA?

¿Qué modelos de CIAA existen? ¿Qué son los "Ponchos"? ¿Cuánto cuesta cada CIAA? ¿Cómo puedo comprar una?

¿Cómo programar una CIAA?

¿Qué necesito para empezar? ¿Qué "lenguajes" puedo usar? ¿Dónde puedo aprender? ¿Cómo consulto mis dudas?

- Trabajar en Equipo.
- Decidir por Consenso.
- Reconocer los Aportes.





¡Seguinos!



github.com/ciaa



[@ProyectoCIAA](https://twitter.com/ProyectoCIAA)



[/ProyectoCIAA](https://www.youtube.com/ProyectoCIAA)



www.proyecto-ciaa.com.ar



¡Muchas gracias!