

TALLER DE PROYECTO I 2019

EDU-CIAA

+

Eclipse IDE

PRIMEROS PASOS

PRIMEROS PASOS

- Instalación de las herramientas de trabajo (Disponibles en Lab. Barcala)
 - IDE Eclipse + ARM GCC + OPENOC Debugger + otros...
- En WIN
 - Descargar el paquete de instalación *Setup_CIAA_IDE_Suite_v1.2.2.exe*
 - <http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:software-ide>
 - Seguir los pasos de la guía de instalación *Instructivo de instalación en WIN*
 - <http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=docu:fw:bm:ide:install>
- En Linux / Ubuntu
 - Seguir los pasos de la guía de instalación *Instructivo de instalación en LINUX*
 - *Ubuntu 14*
 - <http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=docu:fw:bm:ide:install>
 - *Ubuntu 16*
 - http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=docu:fw:bm:ide:install_linux
 - *Ubuntu 18*
 - https://docs.google.com/document/d/1iTK27sv_Vc2F9clWshdBXy524J5BLITPsYd5mECSwrk/edit
 - https://docs.google.com/document/d/1iVhpiYWKoNsVariiURx_DLUO1nSq_9mUkWfENwboQWM/edit#heading=h.xi23fuxhnyxv

PRIMEROS PASOS

Nuevas Herramientas (2019):

- Software: <https://github.com/ciaa/software/>
- Paquete de herramientas listas para usar para la programación de las plataformas del Proyecto CIAA. Herramientas contenidas:
 - Lanzador de aplicaciones.
 - Entornos de programación en lenguaje C/C++:
 - Embedded IDE.
 - GNU MCU Eclipse.
 - IDE4PLC. Entorno de programación en lenguaje Ladder Diagram (IEC 61131-3).
 - CIAABOT IDE. Entorno de programación en lenguaje CIAABOT (basado en Blockly).
 - GNU ARM Embedded. Toolchain.
 - OpenOCD. Programación y depuración.
 - Zenity. Generación de interfaces gráficas.

PRIMEROS PASOS

Descarga del Proyecto a utilizar como base en el TP1

- Firmware v3 : https://github.com/ciaa/firmware_v3/

Notar que a diferencia de la version anterior Firmware_V2, el archivo project.mk se reemplaza por estos dos según indica el readme en el github oficial:

Select a target board to compile

- Create a *board.mk* text file inside this folder.
 - `BOARD = edu_ciaa_nxp`

Select a program to compile

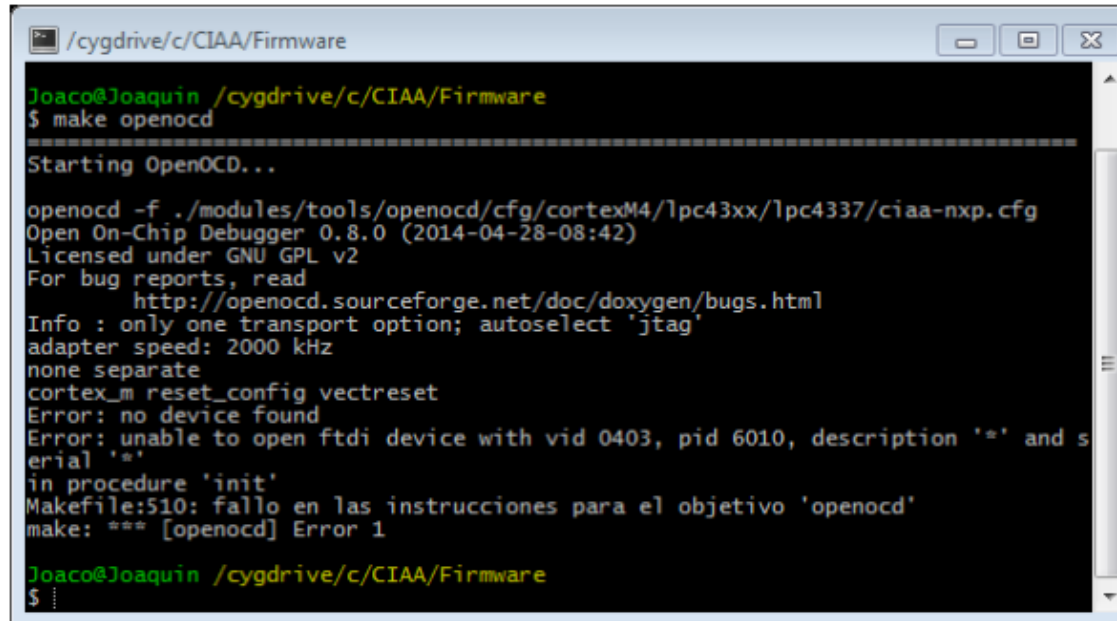
- Create a *program.mk* text file inside this folder.
 - `PROGRAM_PATH = examples/c`
 - `PROGRAM_NAME = app`

PRIMEROS PASOS

- Verificar si las herramientas se instalaron correctamente utilizando los comandos de consola (Linux o Cygwin)
- Abrir una consola y ubicarse en el directorio del proyecto
- Ejecutar:
 `make clean`
 `make`
 `make download`
- Si todo esta correcto, el programa compilado se bajará y ejecutará en la placa EDU-CIAA

PRIMEROS PASOS

- Si aparecer el siguiente error al descargar o depurar, significa que conectamos la placa en un puerto donde no está instalado el driver

A screenshot of a terminal window titled "/cygdrive/c/CIAA/Firmware". The prompt is "Joaco@Joaquin /cygdrive/c/CIAA/Firmware". The user has entered "\$ make openocd". The output shows "Starting OpenOCD..." followed by the OpenOCD command line: "openocd -f ./modules/tools/openocd/cfg/cortexM4/lpc43xx/lpc4337/ciaa-nxp.cfg". It then displays version and license information, the transport option 'jtag', and the adapter speed. The error message is: "Error: no device found", "Error: unable to open ftdi device with vid 0403, pid 6010, description '*' and serial '*'", and "in procedure 'init'". The final output is "Makefile:510: fallo en las instrucciones para el objetivo 'openocd'" and "make: *** [openocd] Error 1".

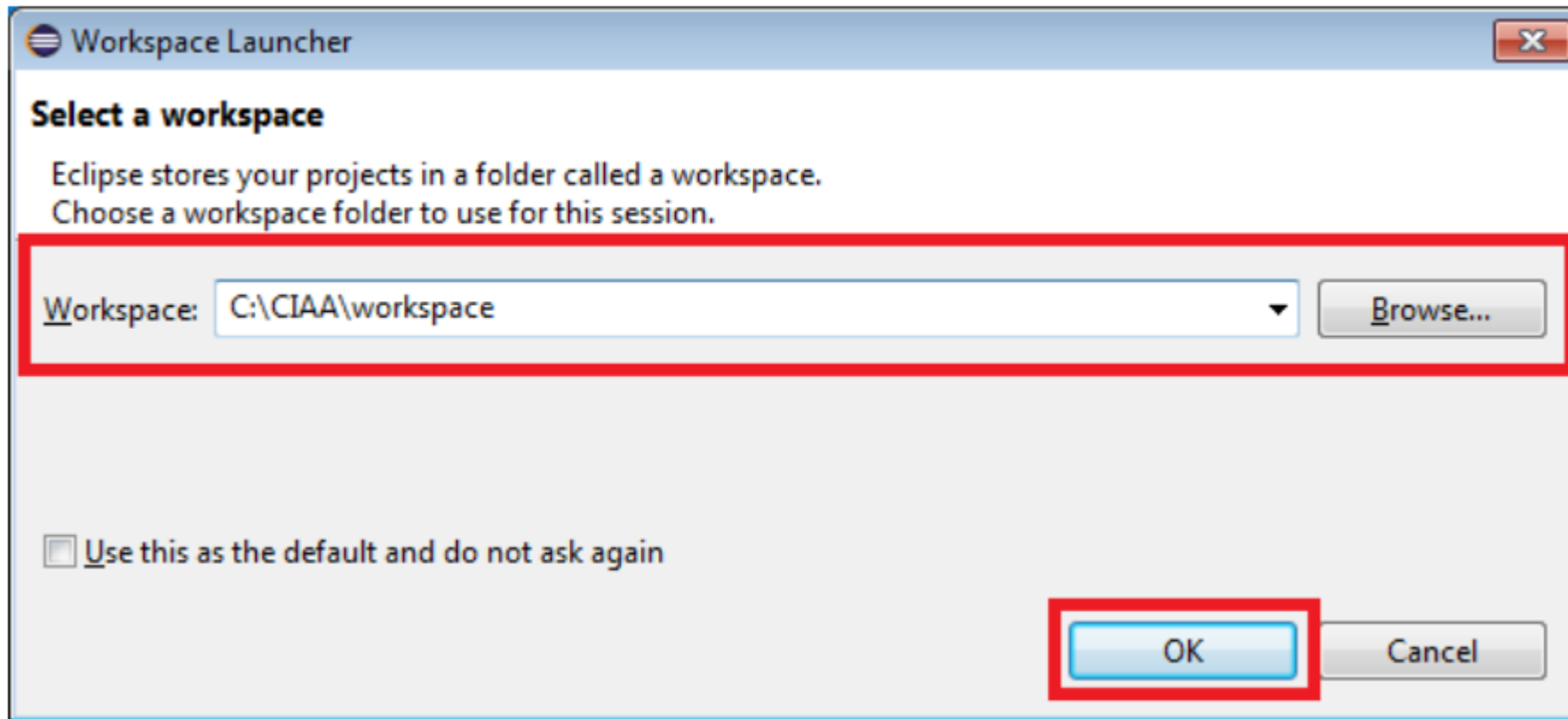
```
Joaco@Joaquin /cygdrive/c/CIAA/Firmware
$ make openocd
=====
Starting OpenOCD...
openocd -f ./modules/tools/openocd/cfg/cortexM4/lpc43xx/lpc4337/ciaa-nxp.cfg
Open On-Chip Debugger 0.8.0 (2014-04-28-08:42)
Licensed under GNU GPL v2
For bug reports, read
  http://openocd.sourceforge.net/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'jtag'
adapter speed: 2000 kHz
none separate
cortex_m reset_config vectreset
Error: no device found
Error: unable to open ftdi device with vid 0403, pid 6010, description '*' and serial '*'
in procedure 'init'
Makefile:510: fallo en las instrucciones para el objetivo 'openocd'
make: *** [openocd] Error 1

Joaco@Joaquin /cygdrive/c/CIAA/Firmware
$
```

Corregir el driver con Zadig.exe (disponible en el paquete de herramientas)
Verificar cargando un ejemplo tipo “blinky” en la placa

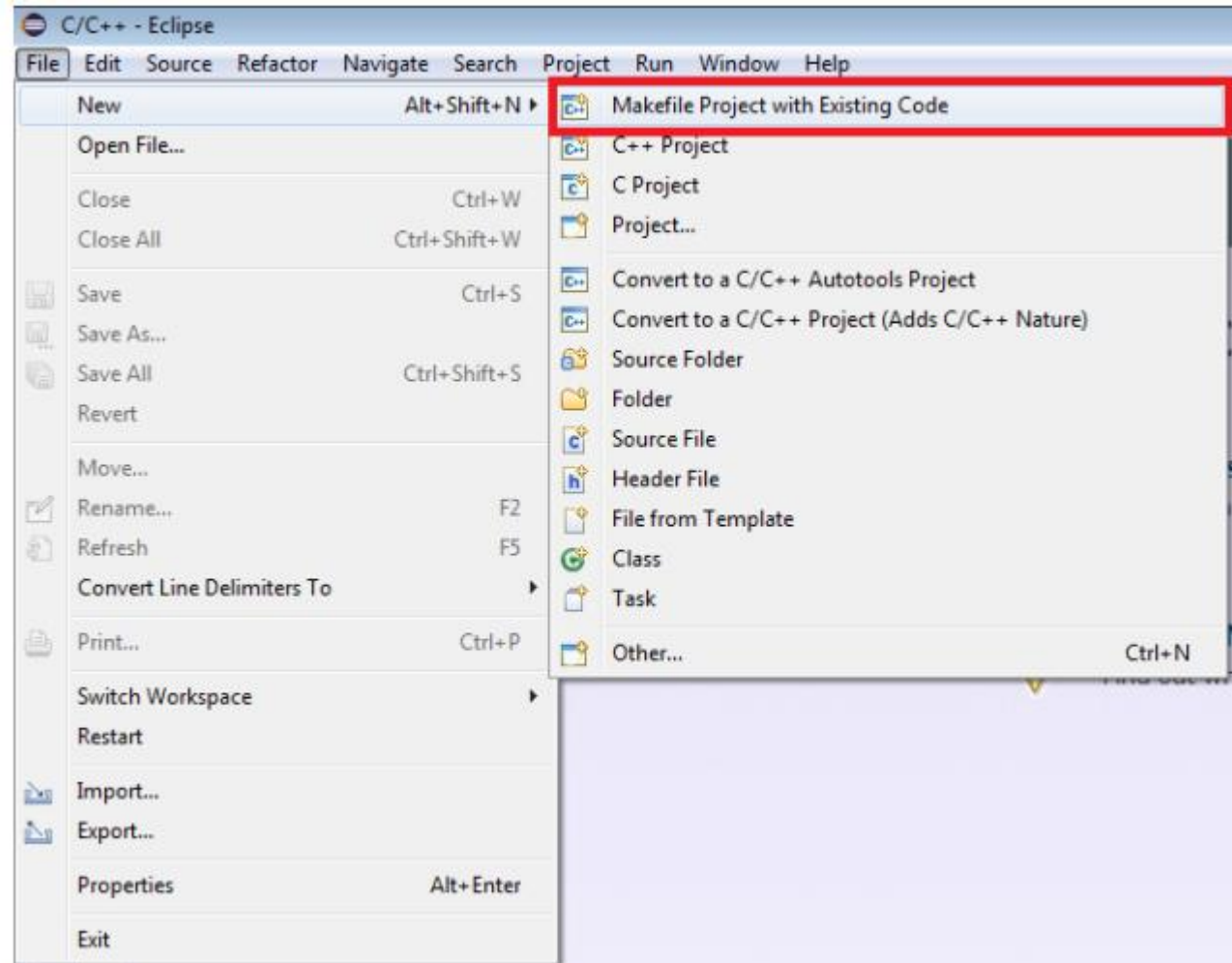
Abrir CIAA Eclipse

- Ejecutar Eclipse
- Seleccionar el WORKSPACE

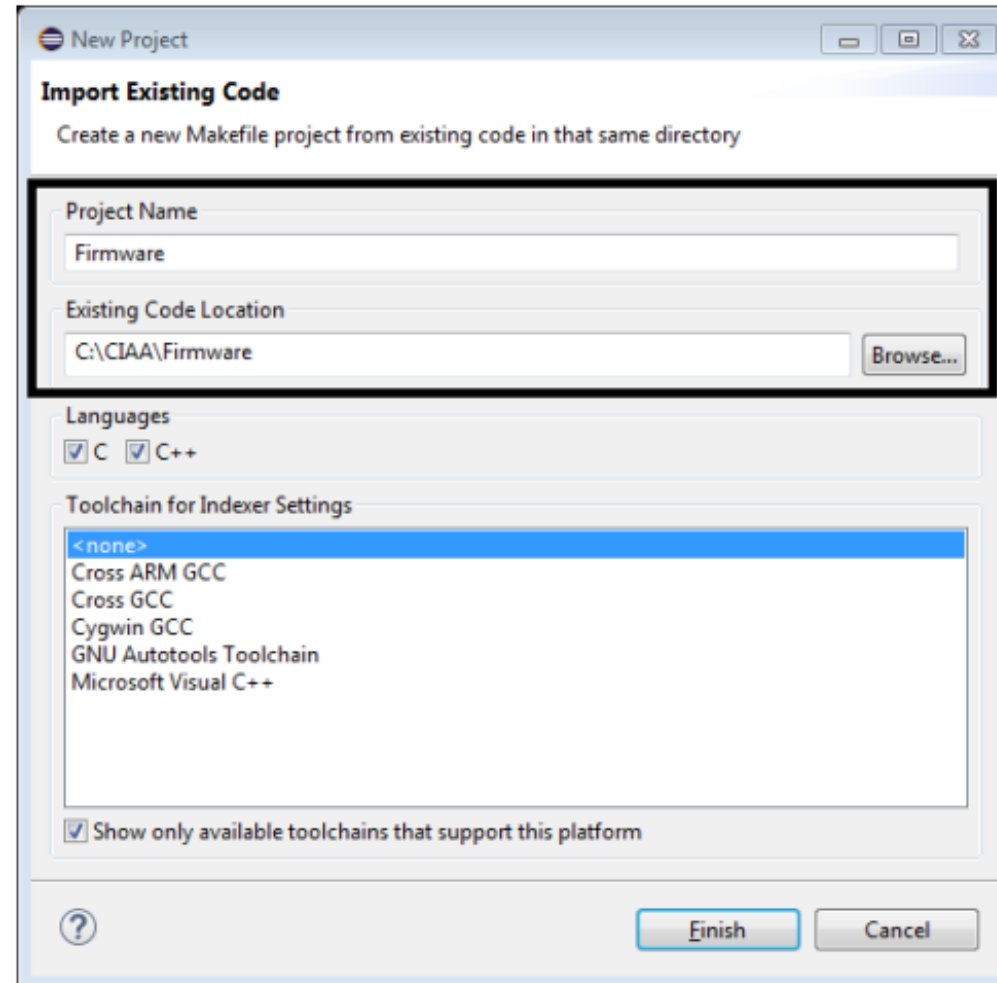


Abrir un nuevo proyecto

- Utilizaremos como ejemplo de proyecto base: Firmware (V2 o V3)



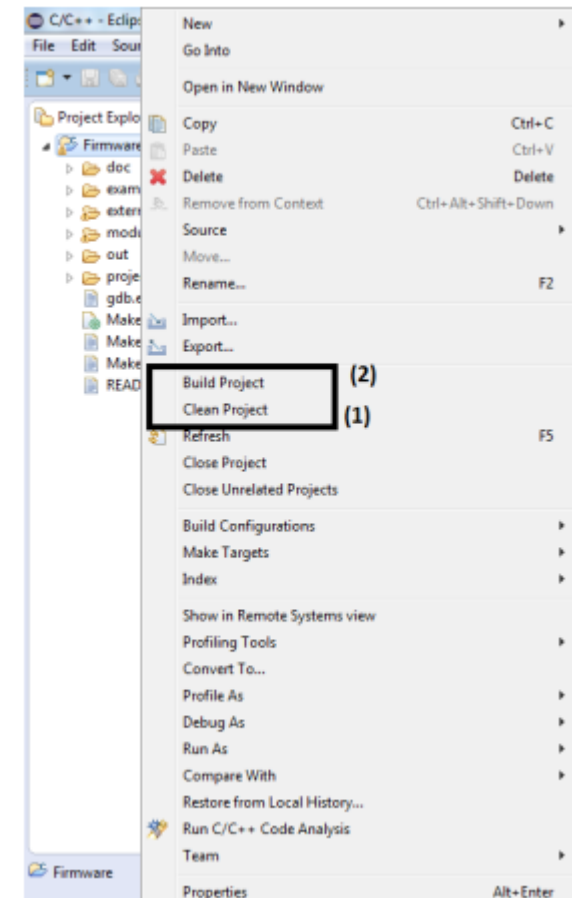
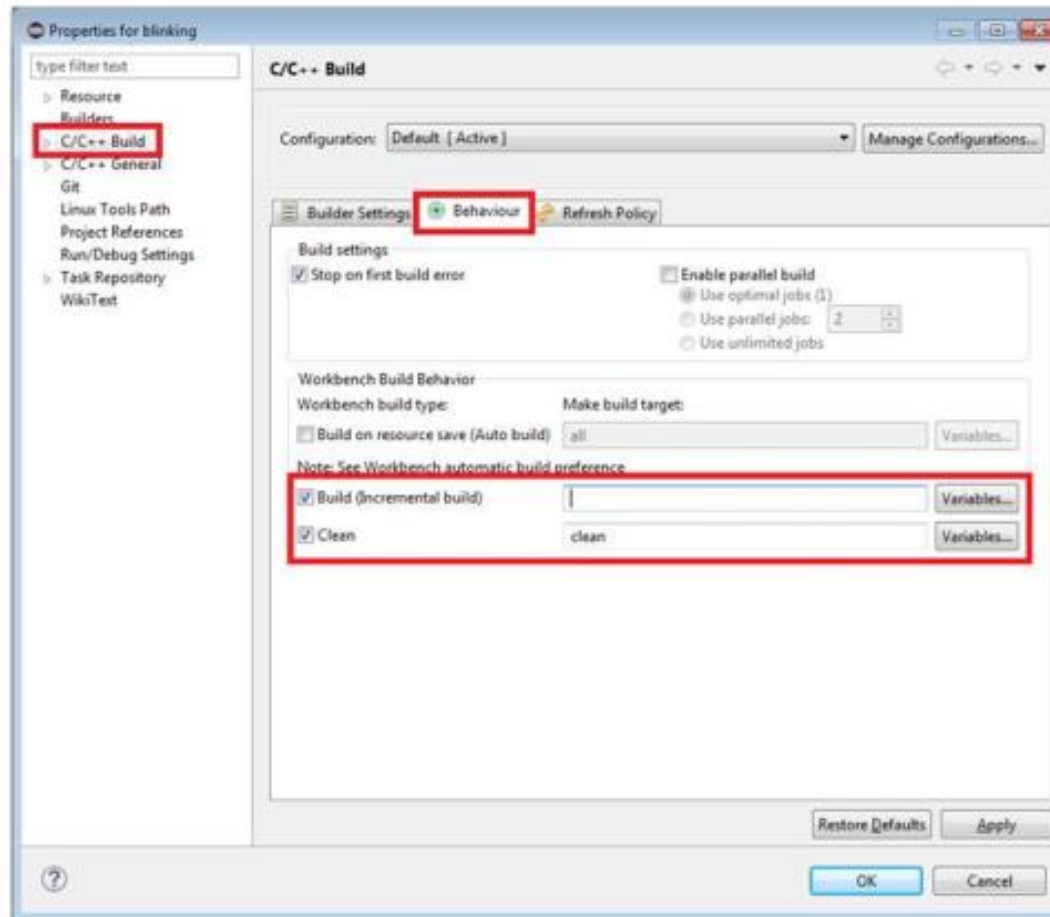
Abrir el Proyecto descargado



Elegir FirmwareV3

Configurar las propiedades del Proyecto

- Botón derecho sobre la carpeta del proyecto, seleccionar *Properties*
- Seleccionar C/C++Build para configurar los comandos CLEAN y BUILD



Utilizando el proyecto Firmware V3

Estructura del Proyecto

- DOCUMENTATION (Documentación completa de la CIAA,EDU-CIAA,PICO-CIAA y micros)
- EXAMPLES (ejemplos diversos aportados al proyecto-ciaa por colaboradores)
- LIBS (carpetas que incluyen todas las bibliotecas utilizadas en los ejemplos, por ejemplo:)
 - lpc_Open (archivos .h y .c que incluyen las bibliotecas LPCOpen)
 - FreeRTOS (archivos .h y .c que incluyen bibliotecas del sistema operativo de tiempo real FreeRTOS)
 - SAPI (archivos .h y .c de la biblioteca SAPI)
- SCRIPTS (archivos para el debugger y test)
- board.mk (archivo de configuración del MAKEFILE para especificar y BOARD=)
- program.mk (archivo de configuración del MAKEFILE para especificar NAME=, PATH=)
- Makefile (scrip para la construcción (BUILD) de los binarios y los archivos necesarios para depuración)

Utilizando el proyecto Firmware V3

- 1) Documentación de la biblioteca sAPI (simple API)
...\\FirmwareV3\\libs\\sapi\\documentation
- 2) Documentación de la EDU-CIAA
...\\FirmwareV3\\documentation\\CIAA_Boards\\NXP_LPC4337\\EDU-CIAA-NXP

Utilizando el proyecto Firmware V3

compilar

```
// INICIALIZACIONES
8
9 #include "app.h"           // <= Su propia cabecera
10 #include "sapi.h"          // <= Biblioteca sAPI
11
12 // FUNCION PRINCIPAL, PUNTO DE ENTRADA AL PROGRAMA LUEGO DE ENCENDIDO O RESET.
13 int main( void )
14 {
15     // ----- CONFIGURACIONES -----
16
17     // Inicializar y configurar la plataforma
18     boardConfig();
19
20     // Crear varias variables del tipo booleano
21     bool_t buttonValue = OFF;
22     bool_t ledValue     = OFF;
23     // Crear variable del tipo tick_t para contar tiempo
24     tick_t timeCount    = 0;
25
26     // ----- REPETIR POR SIEMPRE -----
27     while( TRUE ) {
28
29         /* Retardo bloqueante durante 100ms */
30
31         delay( 100 );
32
33         /* Si pasaron 10 segundos comienza a funcionar el programa que copia las
34            configuraciones de BOARD al LED. Mientras espera titila el LED. */
```

Configurar board.mk
BOARD = edu_ciaa_nxp

Configurar program.mk
PROGRAM_PATH = examples/c
PROGRAM_NAME = app

Properties Progress

Writable Smart Insert 19: 1

13:33 26/08/2019

Utilizando el proyecto Firmware V3

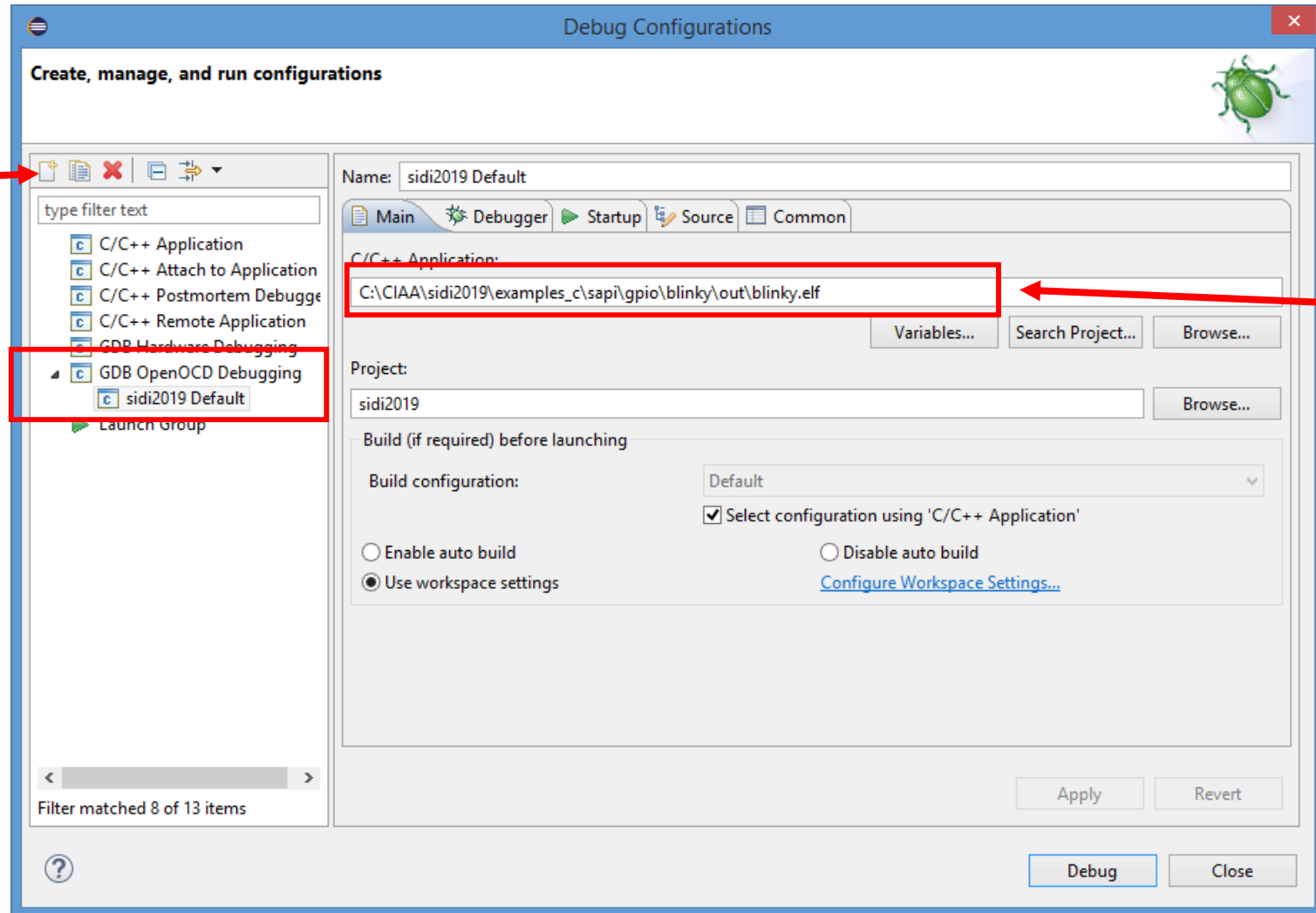
- Para descargar y depurar el programa en la placa se debe configurar el eclipse como se muestra a continuación:

configuración de OpenOCD para Debug:

- Durante la instalación del CIAA-IDE, se instalan las herramientas para Debug. Aquí se utilizará OpenOCD
- Para configurar esta herramienta, desde el menú ***'Run→Debug Configurations...'***
- doble click sobre ***'GDB OpenOCD Debugging'*** para crear un módulo nuevo de configuración del tipo ***'GDB OpenOCD Debugging'***.
- configurar como lo indica la figura siguiente:

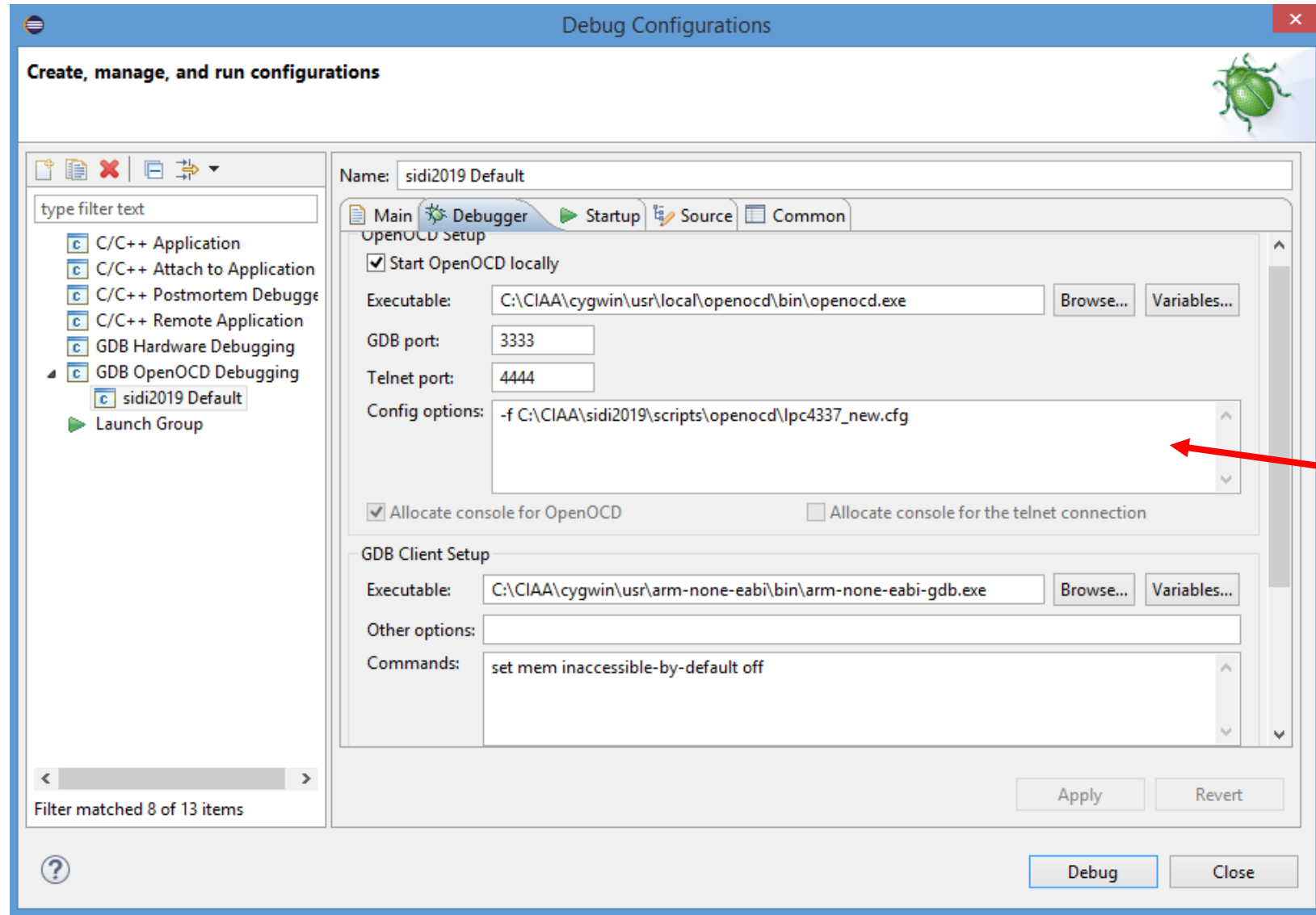
Utilizando el proyecto Firmware V3

Presionar New
Para crear un perfil
De debug



Establecer la ruta del
archivo se desea
depurar
Extensión .elf

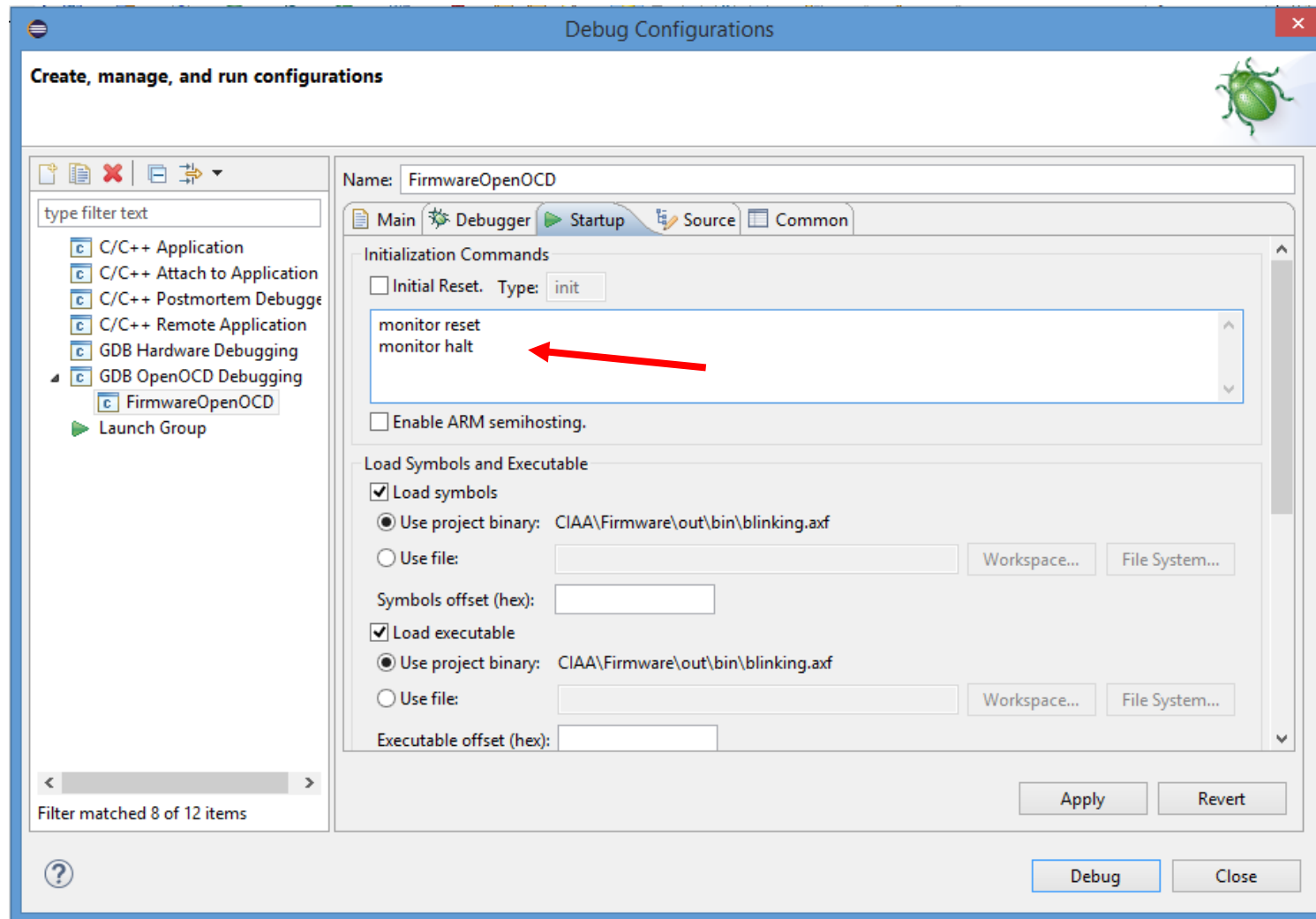
Utilizando el proyecto Firmware V3



Configurar
La pestaña
Debugger

Indicar la ruta del
archivo
Lpc4337_new.cfg

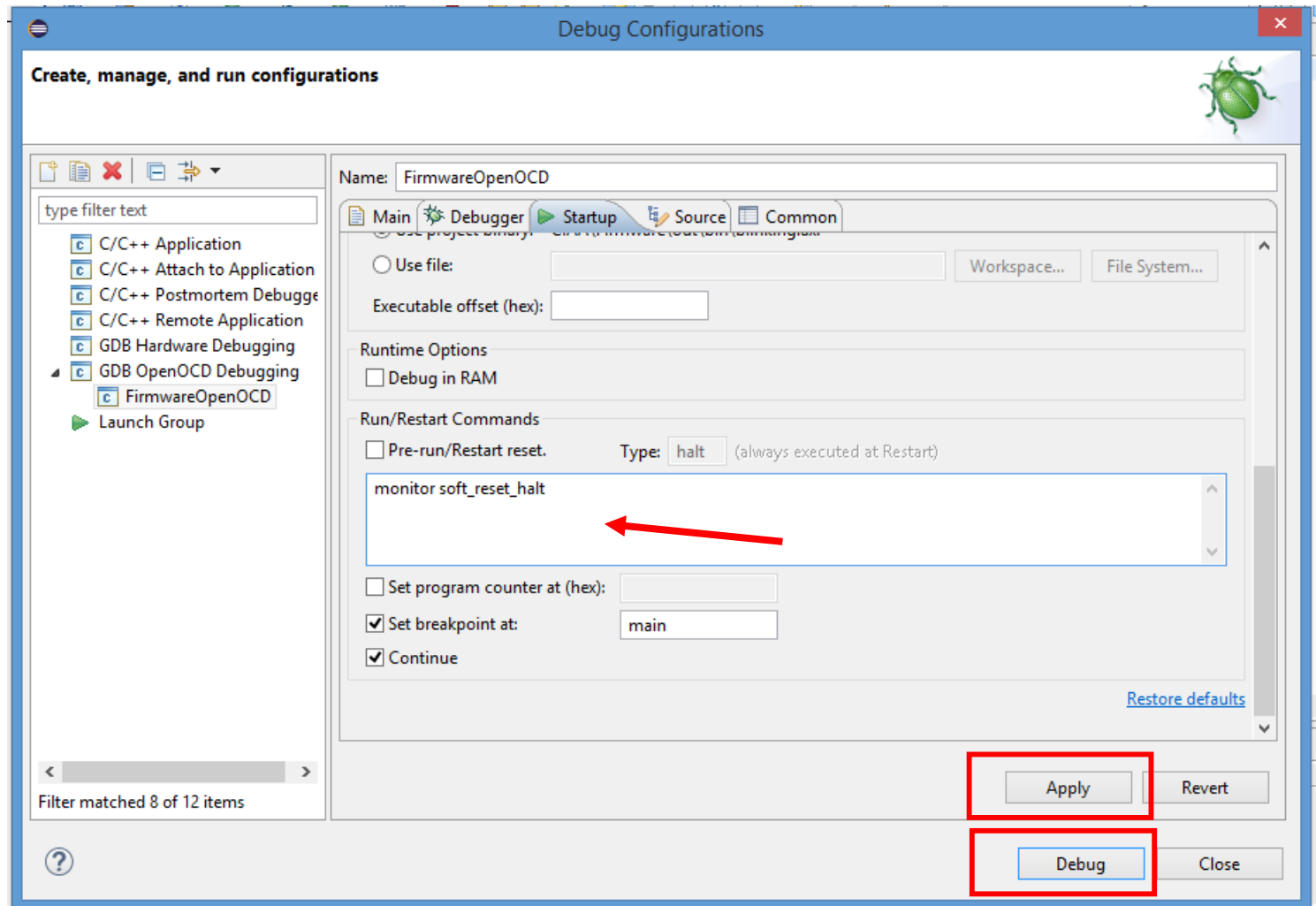
Utilizando el proyecto Firmware V3



Configurar la pestaña
Startup



Utilizando el proyecto Firmware V3



Utilizando el proyecto Firmware V3

Debug - firmware_v2-master/sapi_examples/edu-ciaa-nxp/bare_metal/gpio/gpio_02_blinky/src/blinky.c - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access C/C++ Debug

Debug

firmware_v2-master Default [GDB OpenOCD Debugging]

gpio_02_blinky.axf

Thread [1] (Suspended : Breakpoint)

main() at blinky.c:63 0x1a000304

arm-none-eabi-gdb.exe

start stop step Debug

(x)= Variables Breakpoints Registers Peripherals Modules

Name	Type	Value

project.mk (gdb[0].proc[42000].threadGroup[i1],gdb[0].proc[42000].OSthread[1].thread[1].frame[0] blinky.c

```
63 boardConfig();
64
65 /* ----- REPETIR POR SIEMPRE ----- */
66 while(1) {
67
68     /* Prendo el led azul */
69     gpioWrite( LEDB, ON );
70
71     delay(500);
72
73     /* Apago el led azul */
74     gpioWrite( LEDB, OFF );
75
76     delay(500);
77 }
```

Outline

- sapi.h
- main(void) : int

Console

firmware_v2-master Default [GDB OpenOCD Debugging] arm-none-eabi-gdb.exe

\$1 = 0xff

The target endianness is set automatically (currently little endian)

Temporary breakpoint 1, main () at sapi_examples/edu-ciaa-nxp/bare_metal/gpio/gpio_02_blinky/src/blinky.c:63

63 boardConfig();