Seminario de Electrónica: Sistemas Embebidos – Instalación de Herramientas

1. **IDE** (Integrated Development Environment) **MCUXpresso** (p/Linux) **o LPCXpresso** (p/Windows)

   a. Instale **OpenOCD 0.10.0** desde el siguiente sitio (seleccione el instalador según su Sistema Operativo):
   https://github.com/gnuarmeclipse/openocd/releases/tag/gae-0.10.0-20160110

   b. Instale **Git** desde el siguiente sitio (seleccione el instalador según su Sistema Operativo, usaremos Git por línea de comando):
   https://git-scm.com/ (https://git-scm.com/book/en/v2/Getting-Started-Installing-Git)

   c. Registrarse, Descargar, Instalar, Ejecutar y Licenciar **MCUXpresso IDE v10.1.1** (o posterior) o **LPCXpresso IDE v8.2.0** (o posterior)
   https://www.nxp.com/support/developer-resources/software-development-tools/mcuxpresso-software-and-tools/mcuxpresso-integrated-development-environment-ide:MCUXpresso-IDE?tab=Design_Tools_Tab

      i. Dentro de MCUXpresso o LPCXpresso, agregue el plug-in **OpenOCD Debugging**
      Menú **Help → Install New Software …** Work with: http://gnuarmeclipse.sourceforge.net/updates
      Seleccione el plug-in y luego siga las instrucciones del asistente (**GNU ARM C/C++ OpenOCD Debugging**)
      Instalar VCP driver del chip **U6** FT2232-H (conversor USB-Serie) **DEBUG** y configurarlo como se explica en:
      http://proyecto-ciaa.com.ar/devwiki/lib/exe/fetch.php?media=docu:fw:bm:ide:installciaa_ide_windows_v1.0.pdf o
      http://proyecto-ciaa.com.ar/devwiki/lib/exe/fetch.php?media=docu:fw:bm:ide:installciaa_ide_linux_v1.0.pdf

      ii. Antes de ejecutar conectar la placa **EDU-CIAA-NXP** a su PC

         1. Usando la línea de comandos, clone el repositorio de trabajo y cree una copia del archivo **project.mk.template** llamada **project.mk**:
         git clone **https://github.com/ciaa/firmware_v2.git**
         cd **firmware_v2**
         git status -s
         git checkout **master**
         cp **project.mk.template project.mk**
         En el archivo **project.mk** podrá configurar el **proyecto**, el **procesador** y la **placa** a utilizar, por ejemplo:
            PROJECT = **sapi_examples/edu-ciaa-nxp/bare_metal/gpio/gpio_02_blinky**
            TARGET = **lpc4337_m4**
            BOARD = **edu_ciaa_nxp**

         2. Seleccionar como nombre de Workspace: **workspace-SE-2019-TPs**

         3. MCUXpresso/LPCXpresso ha abierto su vista **Develop**, en ella editaremos y compilaremos fuentes C (.h & .c), procedamos a abrir firmware_v2 del proyecto CIAA.
         Mediante **Fiel -> New -> Other -> C/C++ -> Makefile Project with Existing Code …**

            a. En **Existing Code Location Browse…** busque la carpeta **firmware_v2** (clonada en 1). Si no eligió una ubicación en particular, por defecto debería estar en directorio del usuario (/home/usuario).

            b. **Destilde** la opción **C++**

            c. **Seleccione** la opción **NXP MCU Tools**

         4. Haga clic en **firmware_v2** y a continuación en **Build firmware_v2 [Debug]**. Debería observar el proceso de compilación que finalizará con una salida **Console** similar a la siguiente (el directorio de salida varía según el microcontrolador elegido):

```
*** linking project gpio_02_blinky ***
   text    data     bss     dec     hex    filename
   8252       8      36    8296    2068
out/lpc4337_m4/gpio_02_blinky.axf
copy from `out/lpc4337_m4/gpio_02_blinky.axf' [elf32-littlearm] to
`out/lpc4337_m4/gpio_02_blinky.bin' [binary]
*** post-build ***
make[1]: Leaving directory '<USER_PATH>/firmware_v2'

Build complete
```

         5. Configuración de **Debug**:

            a. Clic derecho en workspace → Debug As → **Debug Configurations…**

b. Doble clic en **GDB OpenOCD Debugging**

c. Clic en **Search Project…** seleccione el archivo **out/lpc4337_m4/gpio_02_blinky.axf** (si no aparece automáticamente)
Otra opción es hacer **clic** en Browser y seleccionar **firmware_v2**, automaticamente debe seleccionar **out/lpc4337_m4/gpio_02_blinky.axf** como C/C++ Application

d. Pestaña **Debugger**, sección OpenOCD Setup, haga clic en **Browse…** para navegar a la carpeta de instalación de OpenOCD, luego a la carpeta bin y finalmente al ejecutable **openocd.exe**

e. **Config options** ingrese el siguiente texto: **-f etc/openocd/lpc4337.cfg**

f. Sección **GDB Client Setup**, en el campo **Executable** escriba: **arm-none-eabi-gdb**

g. A partir de este punto es necesario tener la **EDU-CIAA-NXP** conectada a la PC a través de la interfaz **Debug**

h. Clic en **Apply** , luego en **Debug**, debería comenzar la sesión de Debug con un **breakpoint** en la primer línea de la función **main()**

i. MCUXpresso ha abierto su vista **Debug**, allí ejecutaremos **gpio_02_blinky** (ejemplo de aplicación)

iii. Dentro de **MCUXpresso/LPCXpresso**, agregar el plug-in **eGit** (Git for Eclipse)
Menú **Help → Install New Software …** Work with: **eGit** - https://download.eclipse.org/egit/updates
→ Add... → OK … Seleccione "**Git integration for Eclipse** & **Java implementation for Git**" **→ Next …**
→ Finish (siga instrucciones del asistente)

iv. Dentro de **MCUXpresso/LPCXpresso**, agregar el plug-in **Yakindu StateChart Tools**
Menú **Help → Install New Software …** Work with: http://updates.yakindu.org/sct/mars/releases/
Seleccione el plug-in y luego siga las instrucciones del asistente (**Yakindu SCT**)

v. Antes de ejecutar asegúrese tener conectada la placa **EDU-CIAA-NXP** a su PC (recuerde conectarla **siempre al mismo puerto USB**) a través de la interfaz **Debug**

1. Seleccionar como nombre de Workspace: **workspace-SE-2019-TPs** (el mismo que utilizó para el **1.c.2.ii.2**)

2. En el archivo **project.mk** podrá configurar el **proyecto**, el **procesador** y la **placa** a utilizar, por ejemplo:
PROJECT = **sapi_examples/edu-ciaa-nxp/statecharts/statecharts_bare_metal**
TARGET = **lpc4337_m4**
BOARD = **edu_ciaa_nxp**

3. Verifique tener en la carpeta **sapi_examples/edu-ciaa-nxp/statcharts/statecharts_bare_metal/gen/** los archivos:
   a. **prefix.sct** Yakindu SCT **Statechart Model** file
      i. De no encontrar el archivo **prefix.sct** => copiar y pegar Blinky.-sct y renombrar como: **prefix.sct**

   b. **pregix.sgen** Yakindu SCT **Code Generator Model** file

4. Para Editar el modelo: Doble clic sobre **prefix.sct**

5. Para Simular el modelo: Clic derecho sobre **prefix.sct** -> **Run Us** -> **1 Satechart Simulation**

6. Para Editar la generación de código: Doble clic sobre **pregix.sgen**

7. Para Generar el código del modelo: Clic derecho sobre **pregix.sgen** -> **Generate Code Artifacts** (Artifacts => **Prefix.c**, **Prefix.h**, **PrefixRequired.h** y **sc_types.h**)

8. Compilación de **firmware_v2**: Idem **1.c.2.ii.3**

9. Configuración de **Debug**: Idem **1.c.2.ii.3**

10. Prueba de **Debug**: Idem **1.c.2.ii.3**