

# **Cómo crear un único bitstream con el SoftConsole para una SmartFusion2**

Creador: David Rubio G.

Entrada: <https://soceame.wordpress.com/2024/12/05/como-crear-un-unico-bitstream-con-el-softconsole-para-una-smartfusion2/>

Blog: <https://soceame.wordpress.com/>

GitHub: <https://github.com/DRubioG>

Fecha última modificación: 23/02/2025

En esta entrada te voy a explicar cómo generar un único bitstream con el bitstream de Libero y el binario que genera el SoftConsole.

Esto es muy útil cuando quieres grabar muchos SoCs con un mismo bitstream, como por ejemplo cuando estás fabricando PCBs.

Para ello primero necesitas conocer la base de funcionamiento de Libero y del SoftConsole.

<https://soceame.wordpress.com/2024/12/01/proyecto-basico-con-libero/>

<https://soceame.wordpress.com/2024/12/02/como-crear-un-proyecto-para-una-smartfusion2/>

Y también tienes que tener claro como se configuran los perfiles de compilación del SoftConsole.

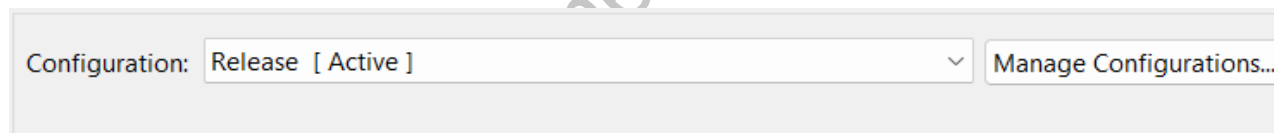
<https://soceame.wordpress.com/2024/12/04/como-compilar-y-depurar-en-softconsole-para-smartfusion2/>

## Primera parte

Primero tiene que tener un proyecto desarrollado en SoftConsole, para ello necesitas haber creado un proyecto en Libero dónde configuras el MSS.

Una vez tengas el proyecto en el SoftConsole probado, depurado y quieras unificarlo con el bitstream de Libero podemos comenzar a generarlo.

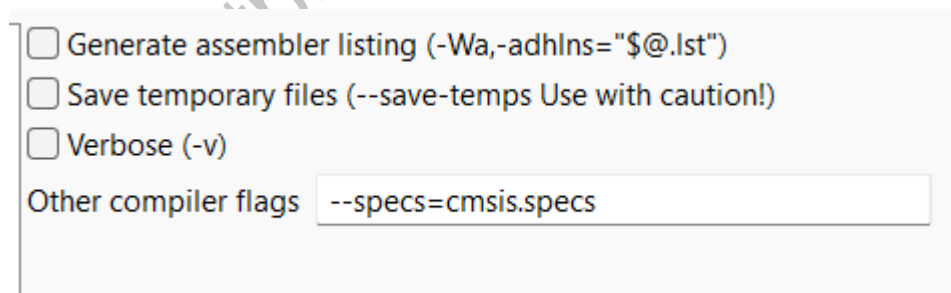
Lo primero que hay que hacer es cambiar el perfil de compilación de depuración a **Release**.



Configuration: Release [ Active ] Manage Configurations...

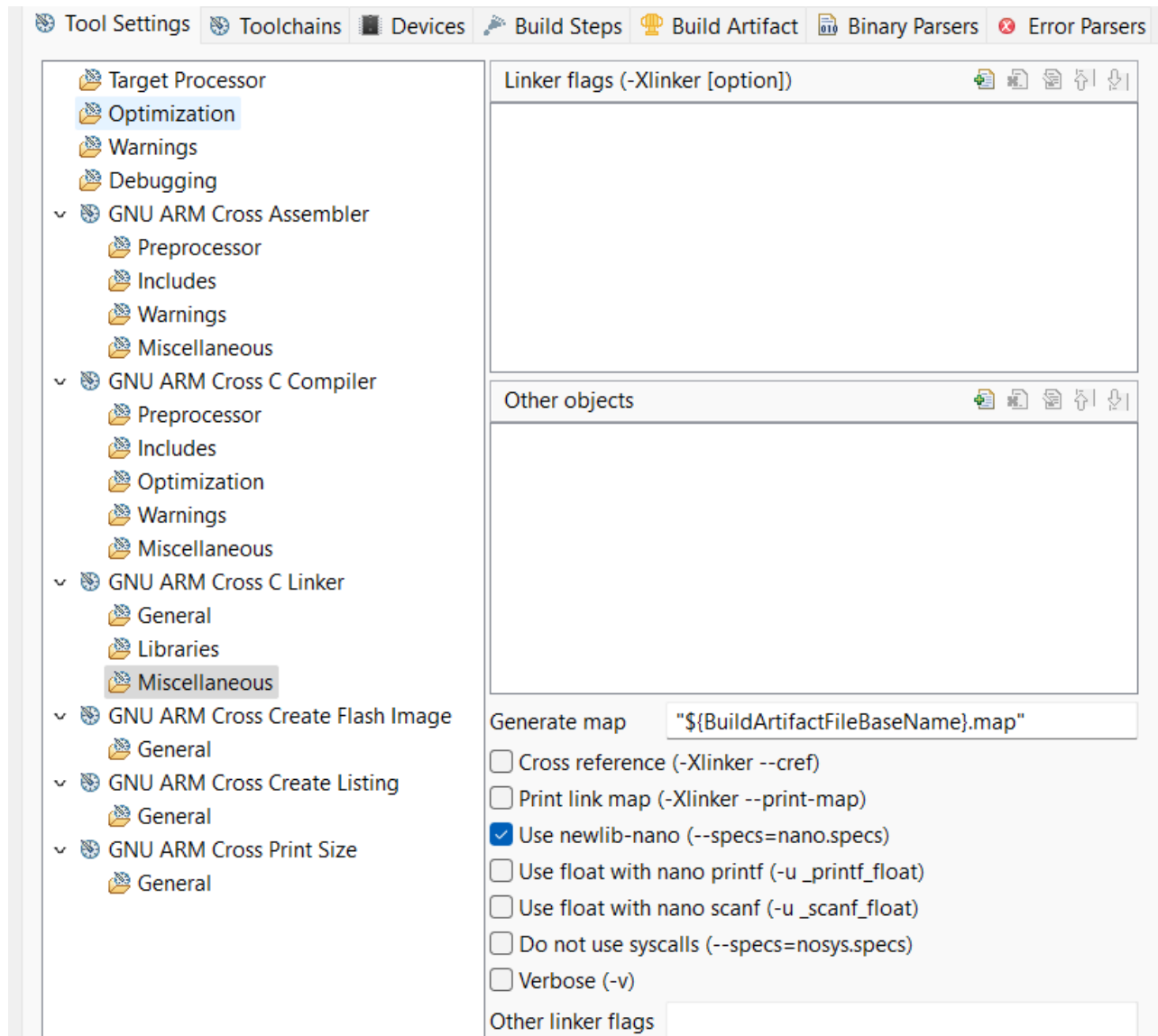
Después tienes que configurar lo mismo que se configura para depuración salvo el *Script files*. Esto se hace de la siguiente forma:

- configurar el *compiler flags*

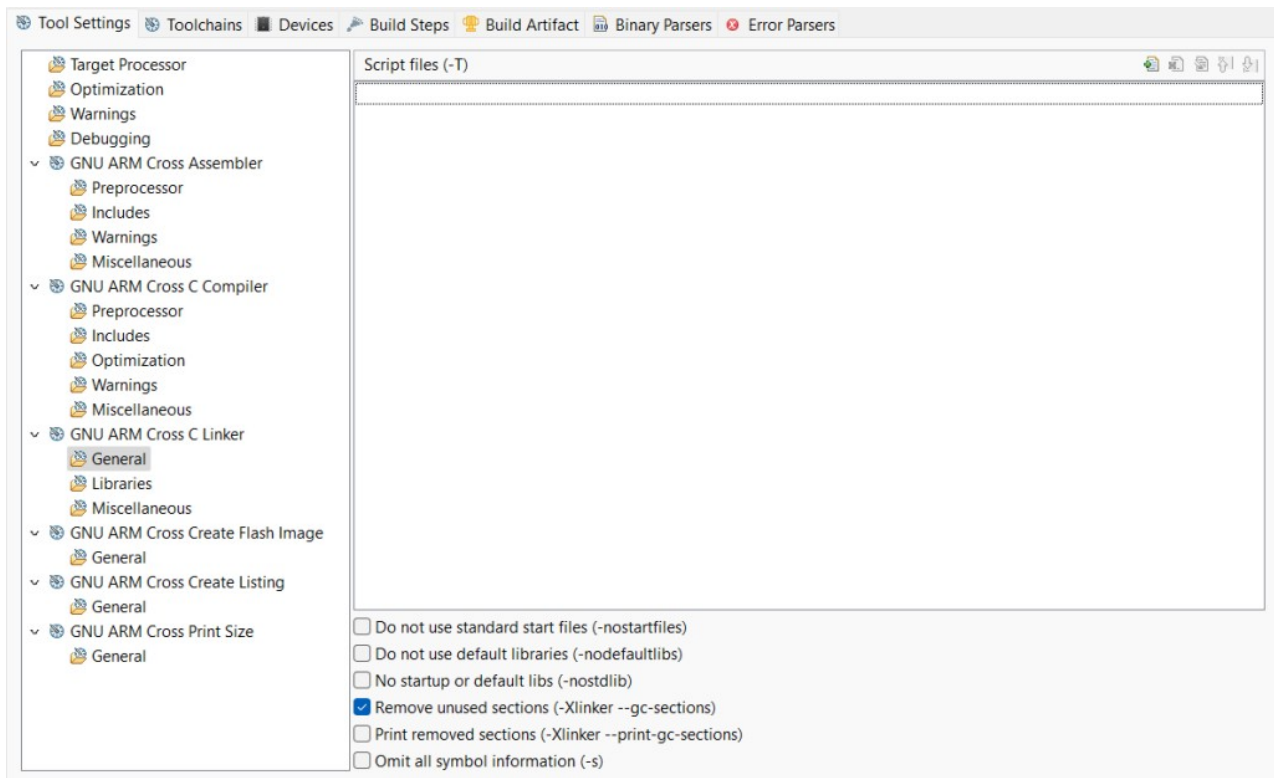


☐ Generate assembler listing (-Wa,-adhlns=\"\$@.lst\")  
☐ Save temporary files (--save-temps Use with caution!)  
☐ Verbose (-v)  
Other compiler flags --specs=cmsis.specs

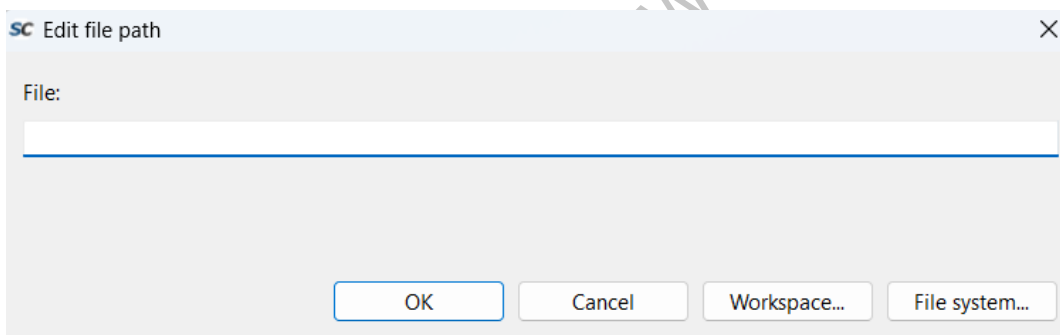
- Y el `--specs=nano.specs`.



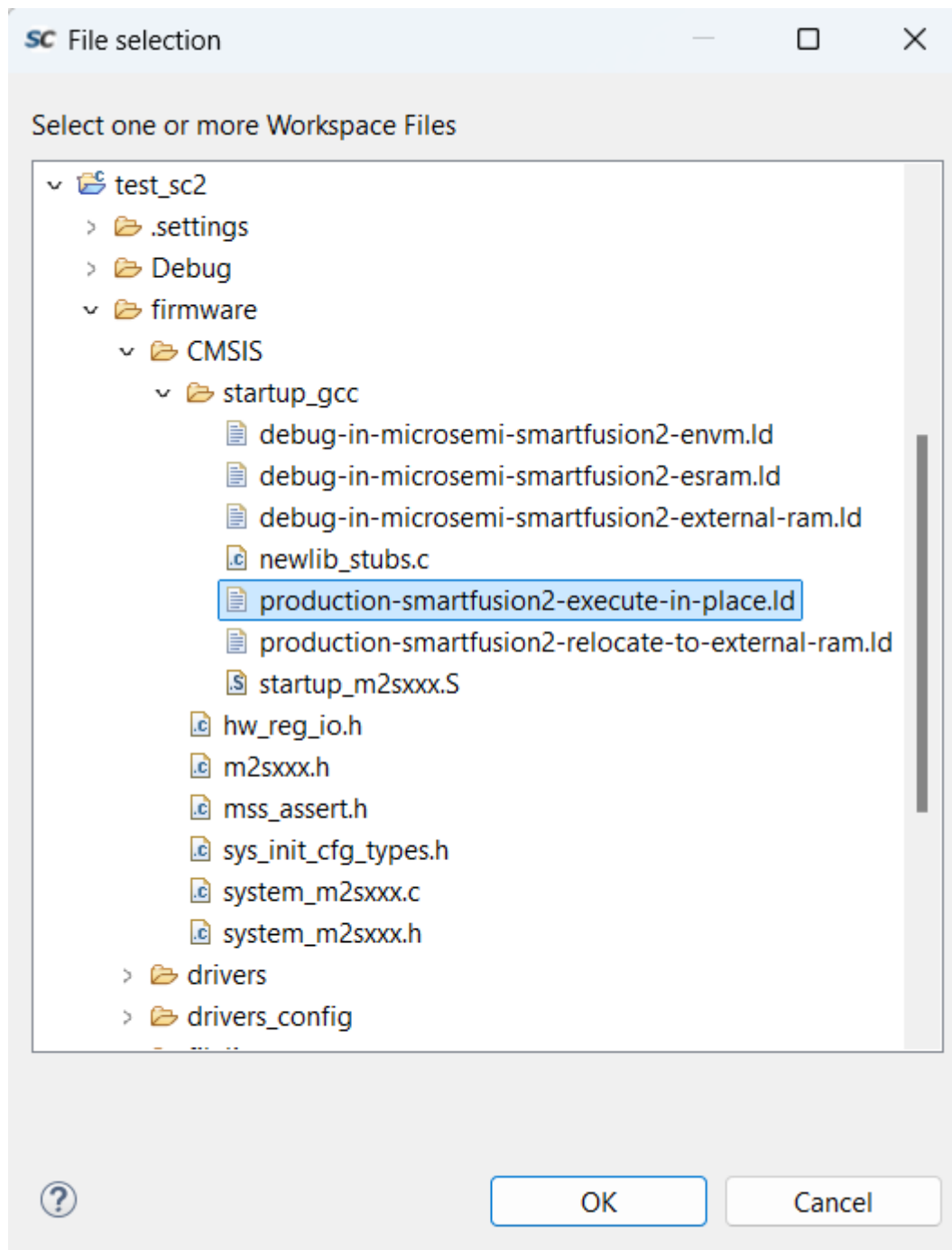
Y lo que tenemos que cambiar es el *Script File* en *GNU ARM Cross C Linker*.



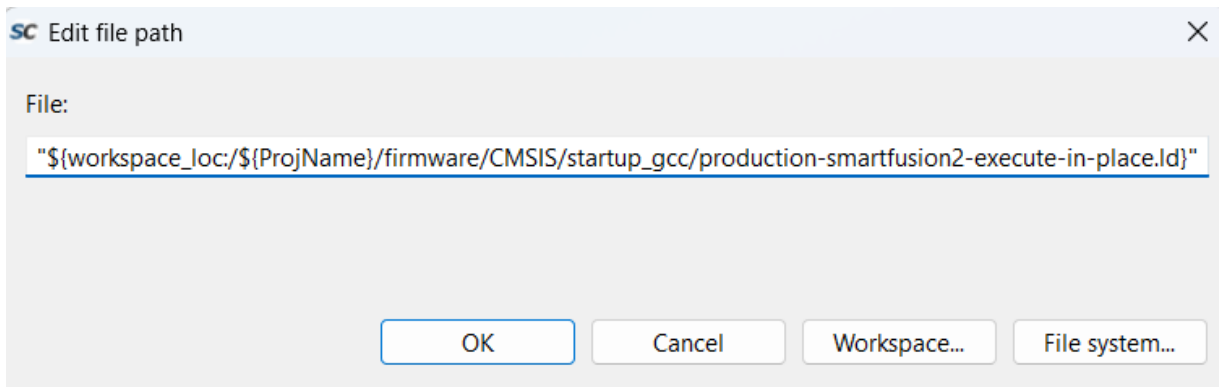
Añadimos un nuevo *Script file* desde el *Workspace*.



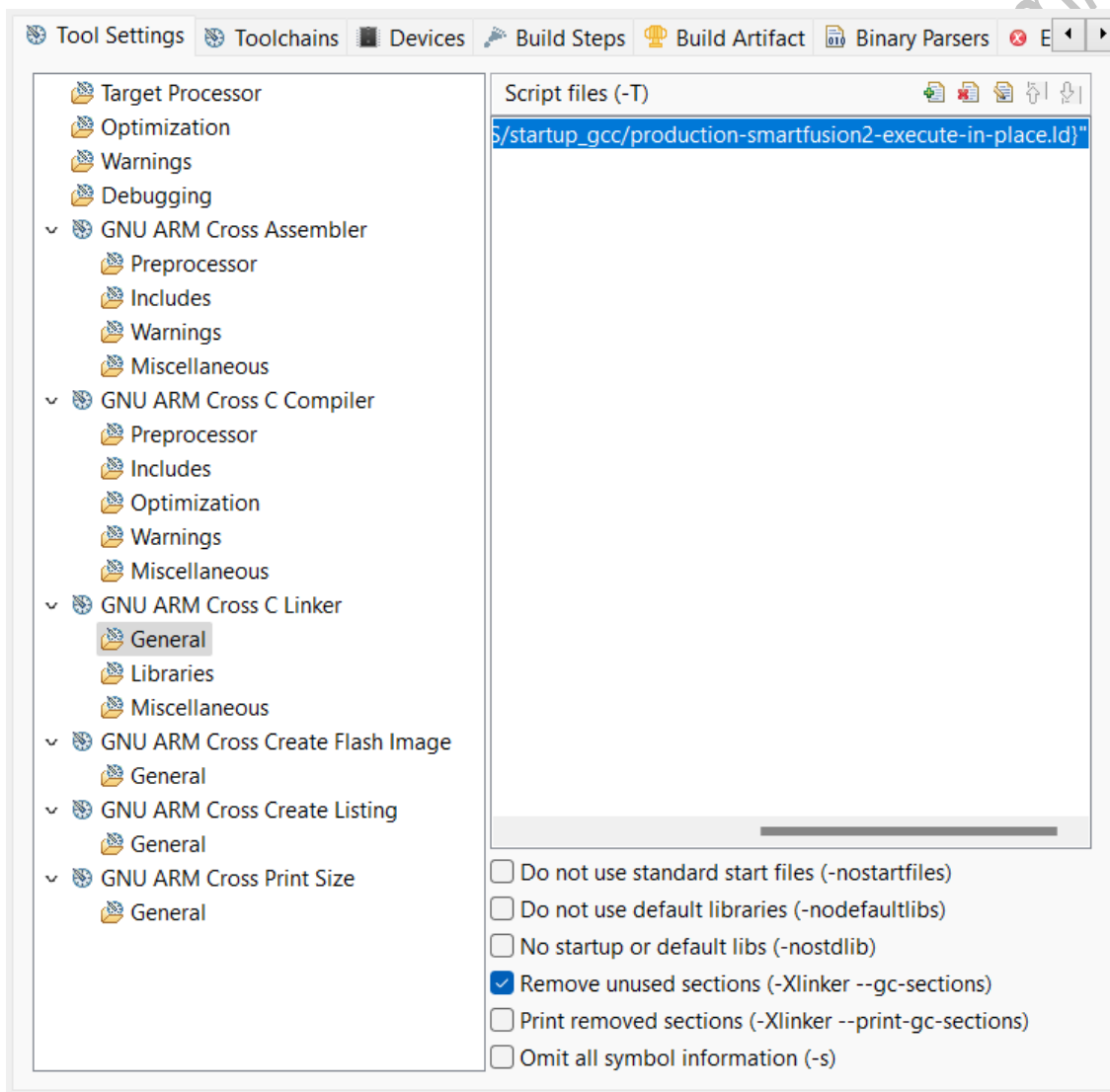
Y ahora cambiamos el fichero de uno de *depuración* a uno de *producción*. Nos aparecen dos opciones de producción, para guardarlo interna o externamente. En nuestro caso elegimos internamente (*production-smartfusion2-executable-in-place.ld*).



Ahora aparece seleccionado.

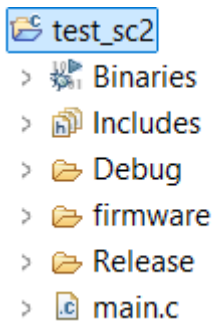


El fichero aparece en la interfaz.

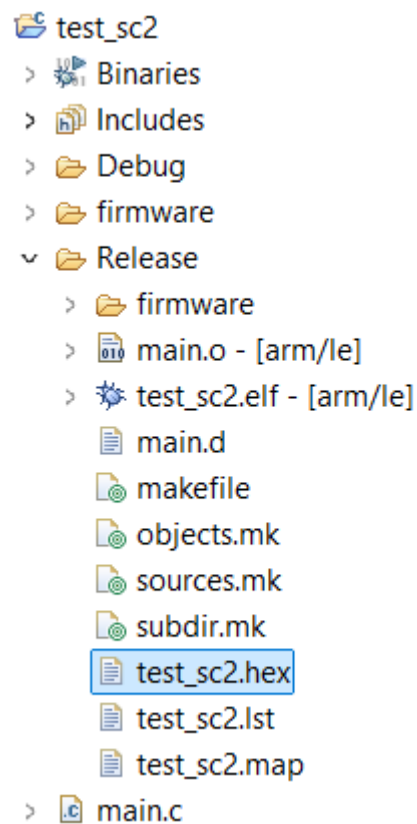


Bien una vez hemos hecho todo esto, podemos volver a compilar el proyecto.

Al hacerlo aparece una nueva carpeta que se llama *Release*.



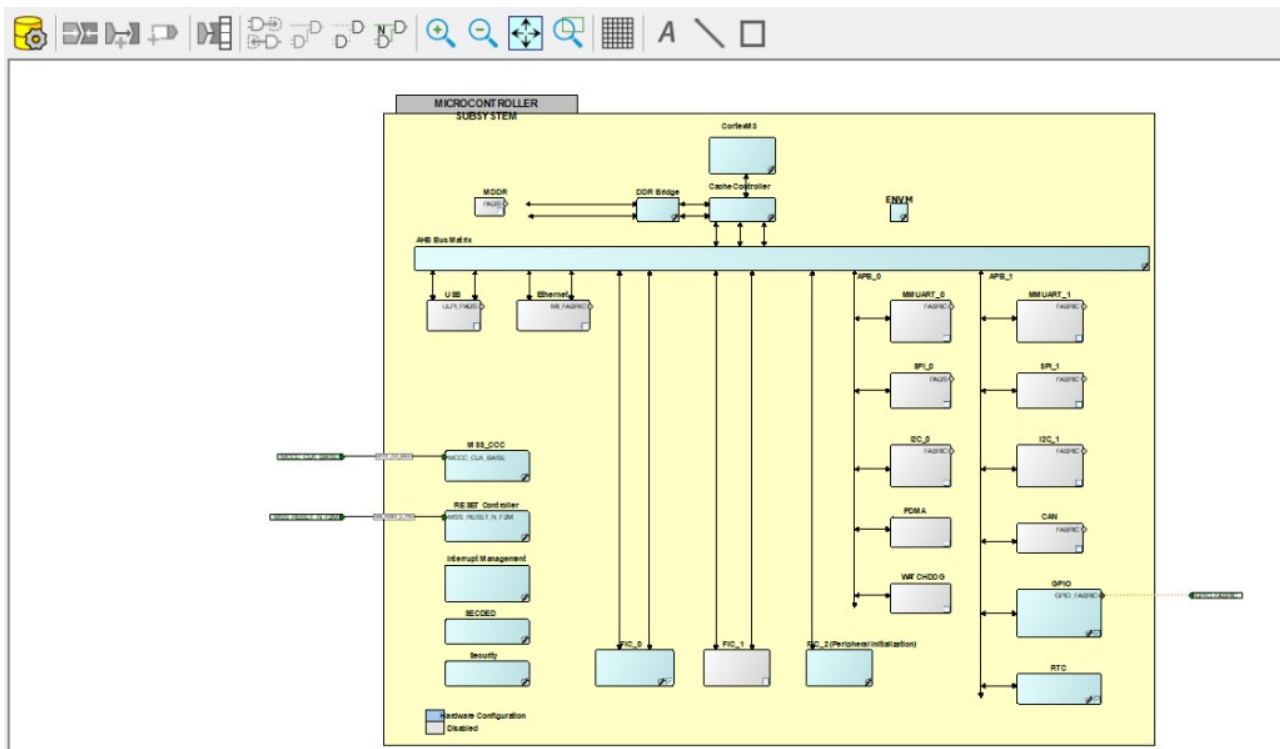
Esta carpeta tiene dentro el fichero que necesitamos para crear el bitstream único, es el fichero **.hex**. Este fichero es el que nos tenemos que llevar a Libero.



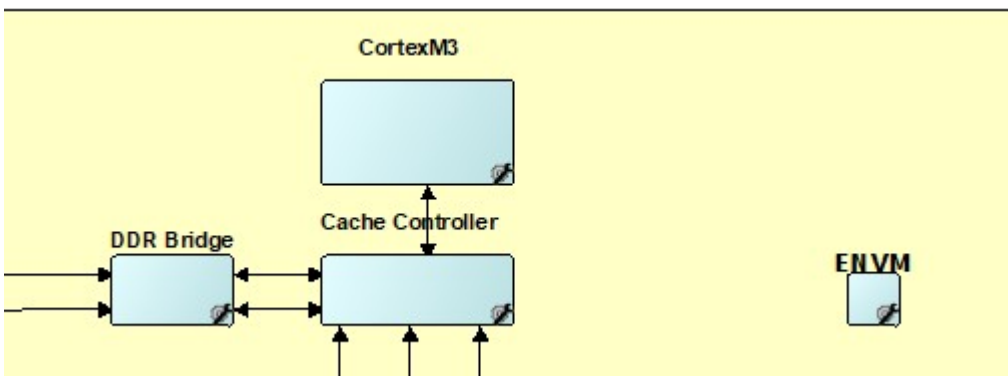
## Segunda parte

Una vez hemos generado el .hex en Release, **NO en Debug**, porque en Debug Libero **no** permite utilizar ese .hex, volvemos al proyecto inicial de Libero.

Volvemos a esta pestaña de configuración del MSS.

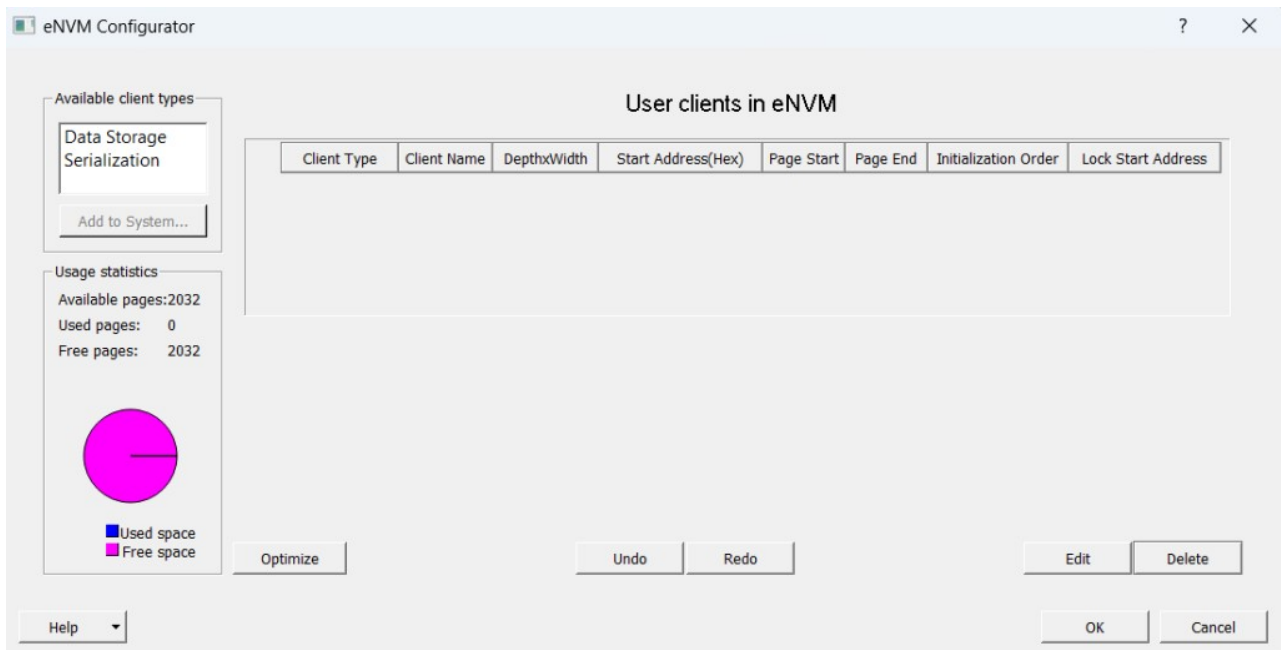


Y tenemos que irnos a la opción **ENVM**, que es la que permite grabar dentro del MSS determinados ficheros en memoria no-volátil.

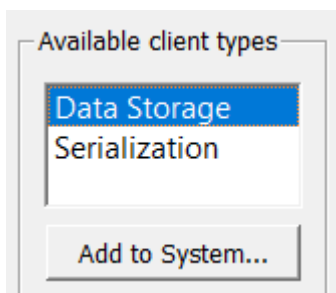


Al pincharle se abre una pestaña como esta, vacía.





Ahora le damos a *Data Storage* y a *Add to System*.



Se nos abre una pestaña como la siguiente, que nos pide dos cosas, darle un nombre a la estructura de memoria y un fichero .hex válido, que es el que hemos generado anteriormente.

**Add Data Storage Client**

Client name:

**eNVM**

☒ Content from file:  ...

Format:

☐ Use absolute addressing ⓘ

☐ Content filled with 0s

☐ No content (client is a placeholder and will not be programmed)

Start address: 0x

Size of word:  Bits

Number of words:  Decimal

☐ Use as ROM ⓘ

☐ Use content for simulation

Help

La damos un nombre y un .hex válido de producción. También marcamos la casilla de *Use as ROM*.

**Add Data Storage Client**

Client name:

**eNVM**

☒ Content from file:  ...

**i** Imported Memory file location : C:  test\_sc2.hex

Format:  ▼

☐ Use absolute addressing **i**

☐ Content filled with 0s

☐ No content (client is a placeholder and will not be programmed)

Start address: 0x  ▼

Size of word:  ▼ Bits

Number of words:  Decimal

☒ Use as ROM **i**

☐ Use content for simulation

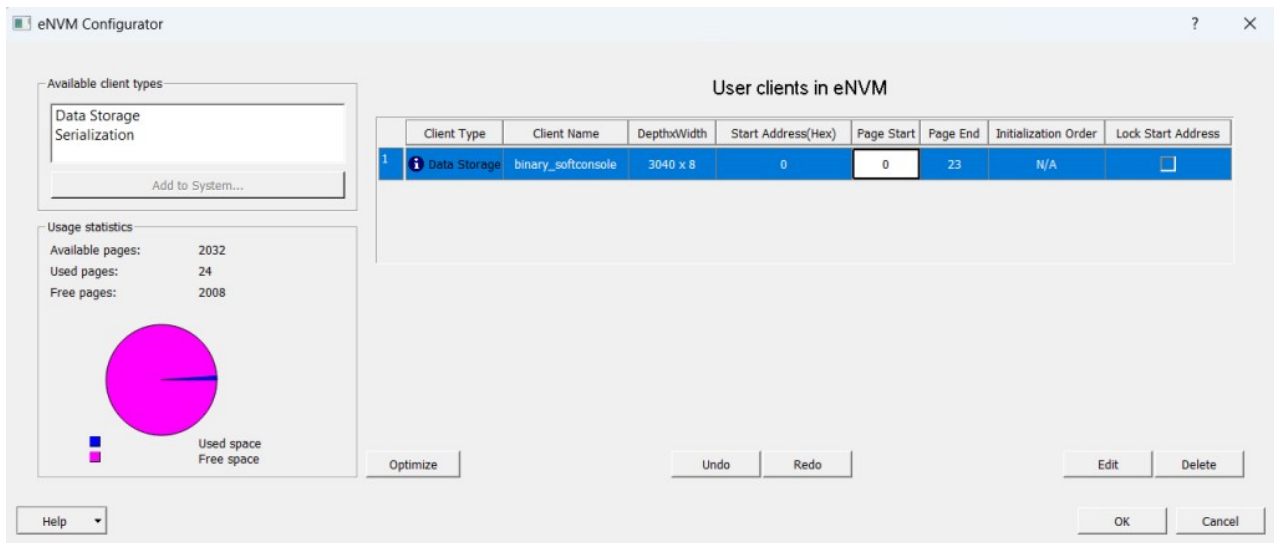
Help ▼ OK Cancel

El *Use as ROM* es para que todos los maestros la puedan leer, pero no modificar.

☒ Use as ROM **i**

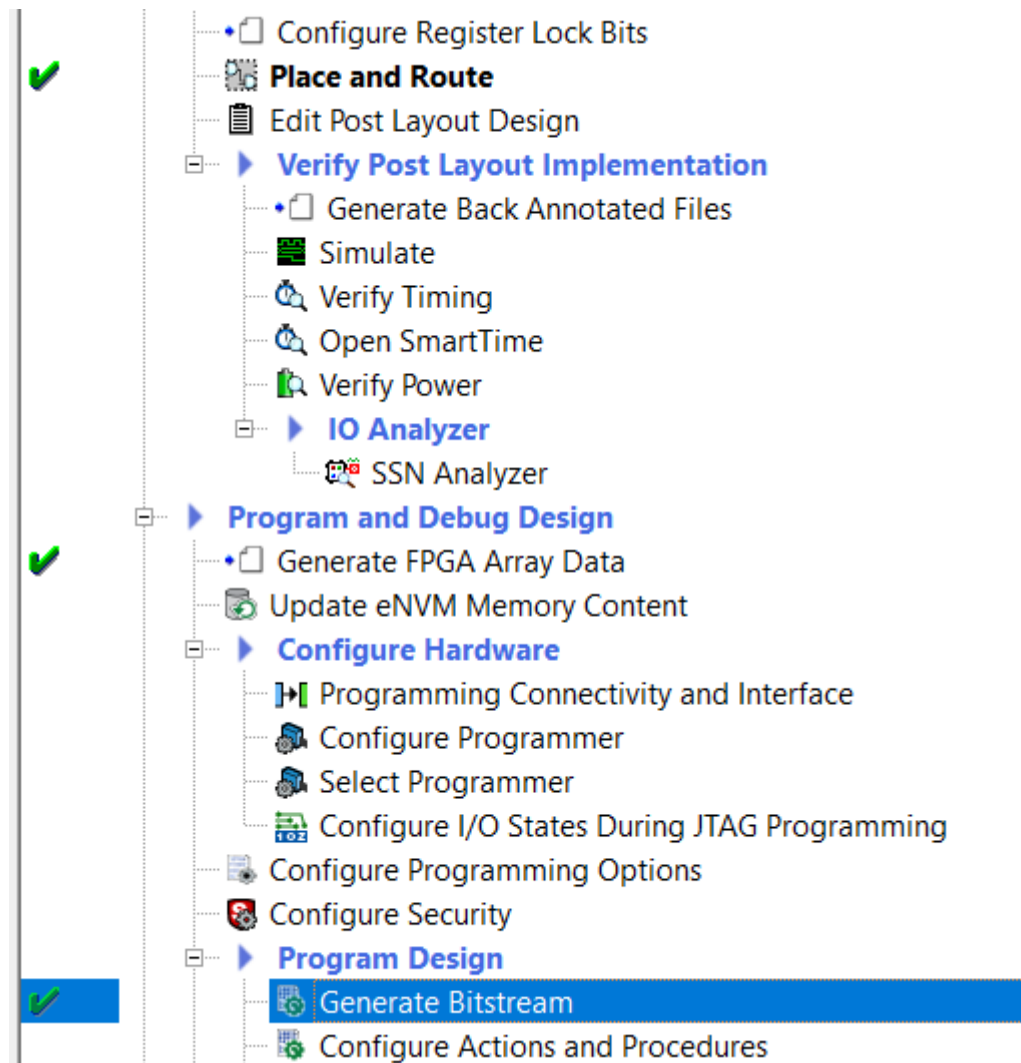
On - All masters have read-only access to this eNVM client.  
Off - All masters have access to this eNVM client.

Después de añadir la nueva capa de memoria aparece en la pestaña.



Ahora lo único que tenemos que hacer es darle a los iconos amarillos con el engranaje y al botón *Build Hierarchy*.

Una vez tengamos el proyecto ya reconstruido y listo, lo sintetizamos y generamos el bitstream.



Este bitstream que hemos generado contiene dentro el .hex que hemos creado. Ahora solo hace falta exportarlo para grabarlo con el *FlashPro Express*.

En esta entrada te dejo cómo instalar y utilizar el *FlashPro Express*.

<https://soceame.wordpress.com/2024/11/24/tutorial-sobre-el-flashpro-express-instalacion-y-ejecucion/>

**NOTA:** la opción *Update eNVM Memory Content* nos lleva a la pestaña donde hemos introducido el binario del SoftConsole (.hex).

