

Desarrollo de sistemas embebidos en framework multiplataforma



Autor: Mg. Lic. Santiago Germino.



VSCodium logo author: Christophe Larsonneur (MIT License)



embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

Disertante

Mg. Lic. Santiago Germino (UBA/CONICET)

<https://www.linkedin.com/in/royconejo>

- De donde vengo.
- A que me dedico.



Introducción

Sistemas embebidos
Framework multiplataforma



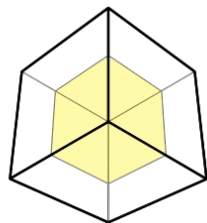
Sistemas embebidos

- Conjunto de hardware y software.
- Resuelven un problema o necesidad puntual.
- Generalmente restringidos en costo, potencia y/o consumo.
- Eficiencia altamente valorada.
- Sistemas críticos embebidos (safety, mission, business, security).

Framework multiplataforma

Breve historia:

- Origen.
- Necesidad.
- Objetivos.



embedul.ar



embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](#))



Primeros trabajos



RETRO-CIAA - Defensa Trabajo Final del Posgrado en Sistemas Embebidos (CESE FI-UBA) - 18/12/2018



<https://youtu.be/Z3VP16FbQA>

Implementación de un adaptador de video por software en microcontrolador de doble núcleo



<http://elektron.fi.uba.ar/index.php/elektron/article/view/91>



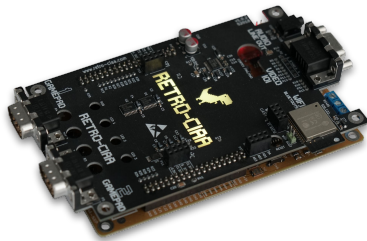
embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

Hardware inicial



EDU-CIAA-NXP

<http://www.proyecto-ciaa.com.ar/>

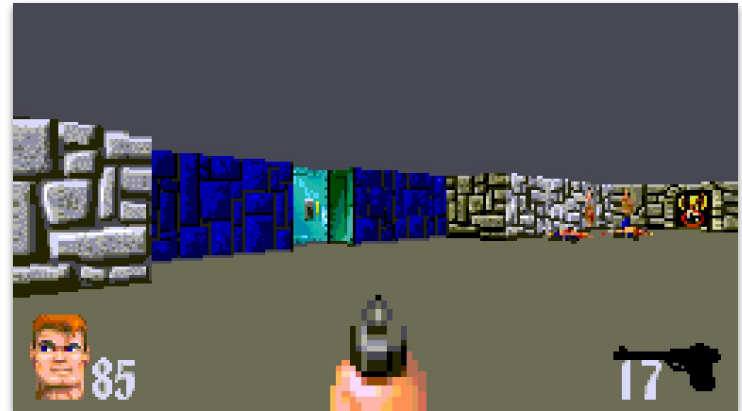


Poncho con conector de video, DAC de sonido y joysticks.

<https://www.retro-ciaa.com>

Primer caso de uso (2019)

- Aplicación compleja.
- Altos requerimientos de memoria, almacenamiento y procesamiento.
- En el límite de lo posible.





Entorno de desarrollo



- Linux.
- VSCodium.
Versión libre de Visual Studio Code.
- GCC. Arm[®] GCC Toolchain. LLVM/Clang.
- Python 3.

Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Entorno de desarrollo

VSCodium
Arm[®] GCC Toolchain
Herramientas

Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Entorno de desarrollo VSCodium



VSCodium logo author: Christophe Larssonneur (MIT License)

Visual Studio Code editor showing C code for the embedded systems framework. The interface includes a file explorer on the left, a central code editor with three files open, and a terminal at the bottom.

File Explorer (Left):

- embedular > source > core > variant.h
- embedular > source > core > manager > C > storage.c
- embedular > source > core > manager > C > log.c

Variant.h (Left Editor):

```
#define VARIANT_ARGS_16(a,...)    VARIANT_SpawnAuto(16), \
                                  VARIANT_ARGS_15(a,...)
#define VARIANT_ARGS_15(a,...)   VARIANT_SpawnAuto(15), \
                                  VARIANT_ARGS_14(a,...)
#define VARIANT_ARGS_14(a,...)   VARIANT_SpawnAuto(14), \
                                  VARIANT_ARGS_13(a,...)
#define VARIANT_ARGS_13(a,...)   VARIANT_SpawnAuto(13), \
                                  VARIANT_ARGS_12(a,...)
#define VARIANT_ARGS_12(a,...)   VARIANT_SpawnAuto(12), \
                                  VARIANT_ARGS_11(a,...)
#define VARIANT_ARGS_11(a,...)   VARIANT_SpawnAuto(11), \
                                  VARIANT_ARGS_10(a,...)
#define VARIANT_ARGS_10(a,...)   VARIANT_SpawnAuto(10), \
                                  VARIANT_ARGS_9(a,...)
#define VARIANT_ARGS_9(a,...)    VARIANT_SpawnAuto(9), \
                                  VARIANT_ARGS_8(a,...)
#define VARIANT_ARGS_8(a,...)    VARIANT_SpawnAuto(8), \
                                  VARIANT_ARGS_7(a,...)
#define VARIANT_ARGS_7(a,...)    VARIANT_SpawnAuto(7), \
                                  VARIANT_ARGS_6(a,...)
#define VARIANT_ARGS_6(a,...)    VARIANT_SpawnAuto(6), \
                                  VARIANT_ARGS_5(a,...)
#define VARIANT_ARGS_5(a,...)    VARIANT_SpawnAuto(5), \
                                  VARIANT_ARGS_4(a,...)
#define VARIANT_ARGS_4(a,...)    VARIANT_SpawnAuto(4), \
                                  VARIANT_ARGS_3(a,...)
#define VARIANT_ARGS_3(a,...)    VARIANT_SpawnAuto(3), \
                                  VARIANT_ARGS_2(a,...)
#define VARIANT_ARGS_2(a,...)    VARIANT_SpawnAuto(2), \
                                  VARIANT_ARGS_1(a,...)
#define VARIANT_ARGS_1(a,...)    VARIANT_SpawnAuto(1), \
                                  ...
```

Storage.c (Middle Editor):

```
static RAMSTOR_Status_Result drvRead (const enum STORAGE_Role Role,
                                      uint8_t *data, uint32_t sector,
                                      uint32_t count)
{
    const struct STORAGE_Volume * vol = &s->volume[Role];
    RAMSTOR_Status_Result R =
        RAMSTOR_MediaRead (vol->driver, data, sector, count);
    if (R != RAMSTOR_Status_Result_OK)
        ++ s->failedRequests.read;
    return R;
}

static RAMSTOR_Status_Result drvWrite (const enum STORAGE_Role Role,
                                       const uint8_t *data,
                                       uint32_t sector, uint32_t count)
{
    const struct STORAGE_Volume * vol = &s->volume[Role];
    s->failedRequests.write += count;
    const RAMSTOR_Status_Result R =
        RAMSTOR_MediaWrite (vol->driver, data, sector, count);
    if (R != RAMSTOR_Status_Result_OK)
        ++ s->failedRequests.write;
```

Log.c (Right Editor):

```
void LOG_Init (struct LOG *const L, struct BOARD *const Board,
              struct STREAM *const DebugStream)
{
    BOARD_AssertState (!s.L);
    BOARD_AssertParams (L && STREAM_IsValid(DebugStream));
    OBJECT_Clear (L);
    L->debugStream = DebugStream;
    L->logTableStyle = &s.DefaultLogTableStyle;
    L->logItemsStyle = &s.DefaultLogItemsStyle;
    L->logProgressStyle = &s.DefaultLogProgressStyle;
    s.L = L;
    LOG_ContextBegin (Board, LANG_BOARD_INIT_SEQUENCE);
    LOG_ContextBegin (L, LANG_INIT);
    {
        LOG_Items (2,
                  LANG_DEBUG_STREAM,    STREAM_Description(DebugStream),
                  LANG_MAX_LOG_ITEMS,  (uint32_t)LOG_ITEMS_MAX);
    }
    LOG_ContextEnd ();
}

inline static void outStr (struct STREAM *const S, const char *const Str)
{
    STREAM_In_FromString (S, Str);
}
```

Terminal (Bottom):

```
bash: /usr/bin/ls: cannot access 'ls': No such file or directory
Run [HOSTED] wolf3d.ar
```

Footer:

embedded systems framework
© 2018-2022 Santiago Perinot
http://embedul.ar
2022-09-18T12:17:03:00 GCC 10.2.1 - 03

Instalación

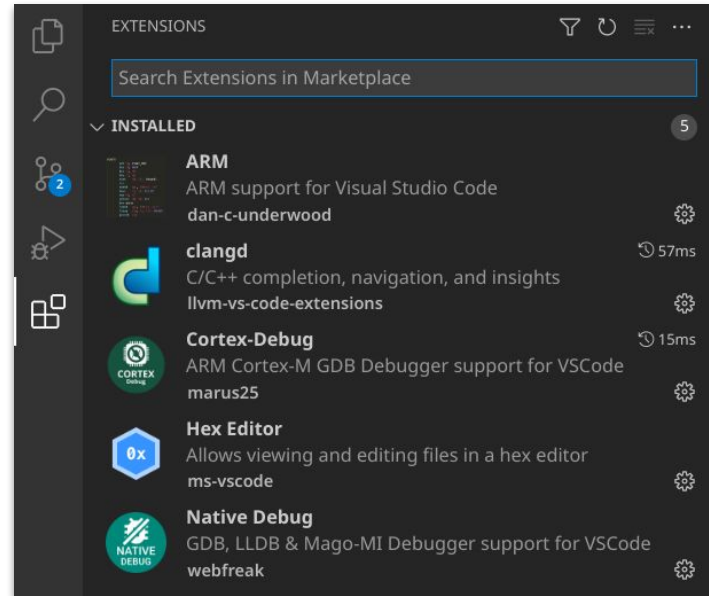
<https://vscodium.com/>

Seguir instrucciones en “Debian / Ubuntu (deb package)”.



Requerimientos

- Distribución Linux actualizada.
Debian 11, Ubuntu 22.04 LTS
- Extensiones: ARM, clangd, Cortex-Debug, Hex Editor, Native Debug.



Entorno de desarrollo

Arm[®] GCC Toolchain

arm Developer

Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.



Arm[®] GNU Toolchain

arm Developer

<https://developer.arm.com/downloads/-/arm-gnu-toolchain-downloads>

x86_64 Linux hosted cross toolchains
AArch32 bare-metal target (arm-none-eabi)

- Descomprimir.
`sudo tar -xvf arm-gnu-toolchain-{}.tar.xz -C /opt`
- Agregar ruta al directorio `bin` de `arm-none-eabi-gcc`.
`export PATH=$PATH:/opt/arm-gnu-toolchain-{} /bin`

Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Entorno de desarrollo

Herramientas





Herramientas

Para build, flash y debug:

- apt install `git` `make` `bear` `gcc` `openocd`
openocd versión 0.11 en adelante.

Para generar documentación:

- apt install `python3-pip`
- pip3 install `sphinx` `hawkmoth` `sphinx_rtd_theme`

VSCodium en sistemas embebidos



VSCodium logo author: Christophe Larssonneur (MIT License)

Conceptos básicos

- Cada proyecto *es una carpeta* (directorio).
- Cada carpeta contiene:
 - Archivo **Makefile** del proyecto.
 - Archivos de código.
 - Carpeta oculta **.vscode**.



Conceptos básicos (cont'd)

VSCoode debe **invocar al Makefile** del proyecto.

¿Como se logra?



Subcarpeta .vscode

Contiene al menos tres archivos en formato **JSON**.

- tasks.json
- launch.json
- settings.json



tasks.json

- Específica **tareas** a ejecutar.
- Generalmente se invocan **programas**.
*Por ejemplo, **make**.*
- Posibilidad de **parámetros** opcionales.
Target hardware, nivel de optimización, etc.



task.json (cont'd)

```
tasks.json M x
embedul.ar > .vscode > {} tasks.json > [ ] tasks > {} 3 > group
1
2 // See https://go.microsoft.com/fwlink/?LinkId=733558
3 // for the documentation about the tasks.json format
4 // -----
5 // embedul.ar build/run tasks.
6 "version": "2.0.0",
7 "tasks": [
8   {
9     "label": "Debug",
10    "detail": "Build w/debug symbols and no optimization.",
11    "type": "shell",
12    "command": "intercept-build make BUILD_TARGET=${input:target} OLEVEL=0 DEBUG=yes FLASH_TOOL=${input:gdb-flash-tool}",
13    "problemMatcher": [],
14    "group": "build"
15  },
16  {
17    "label": "Release",
18    "detail": "Build w/optimizations.",
19    "type": "shell",
20    "command": "intercept-build make BUILD_TARGET=${input:target} OLEVEL=${input:optimization} DEBUG=no FLASH_TOOL=${input:gdb-flash-tool}",
21    "problemMatcher": [],
22    "group": "build"
23  },
24  {
25    "label": "Documentation",
26    "detail": "Rebuild documentation.",
27    "type": "shell",
28    "command": "cd ./documentation && make html",
29    "problemMatcher": [],
30    "group": "build"
31  }
32 ]
```

launch.json

- Configura sesiones de depuración.
- Se especifica tipo de sesión.
GDB, cortex-debug...
- Parámetros según tipo de sesión.
Por ejemplo en cortex-debug, servidor GDB: openocd, JLink...
- Ejecutable.



Launch.json (cont'd)

```
{} launch.json M x
embedul.ar > .vscode > {} launch.json > ...
1  {
2    "version": "0.2.0",
3    "configurations": [
4      {
5        "name": "[GDB] host",
6        "cwd": "${workspaceFolder}",
7        "request": "launch",
8        "type": "gdb",
9        "target": "${workspaceFolder}/build/hosted/gcc/debug_generic_00/${workspaceFolderBasename}.elf"
10     },
11     {
12       "name": "[JLink] edu_ciaa_retro_poncho",
13       "cwd": "${workspaceFolder}",
14       "executable": "${workspaceFolder}/build/edu_ciaa_retro_poncho/arm-none-eabi-gcc/debug_cortex-m4_00/${workspaceFolderBasename}.elf",
15       "request": "launch",
16       "type": "cortex-debug",
17       "servertype": "jlink"
18     },
19     {
20       "name": "[OpenOCD] edu_ciaa_retro_poncho-m4",
21       "cwd": "${workspaceFolder}",
22       "executable": "${workspaceFolder}/build/edu_ciaa_retro_poncho/arm-none-eabi-gcc/debug_cortex-m4_00/${workspaceFolderBasename}.elf",
23       "request": "launch",
24       "type": "cortex-debug",
25       "servertype": "openocd",
26       "numberOfProcessors": 2,
27       "targetProcessor": 0,
28       "configFiles": [
29         "${env:LIB_EMBEDULAR_PATH}/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/openocd/edu_ciaa-ftdi.cfg"
30       ],
31       "svdFile": "${env:LIB_EMBEDULAR_PATH}/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/svd/LPC43xx_43Sxx.svd"
32     },
33   ]
34 }
```

settings.json

- Preferencias del editor.

Por ejemplo, el código de embedul.ar sigue un formato de 80 columnas y 4 espacios por tab.

- Configuración de las extensiones.



¿Qué es un *workspace*?

- Se pueden abrir **varios** proyectos (varias carpetas) en el mismo *workspace*.
- Archivos abiertos en el editor.
- Configuración de la vista.



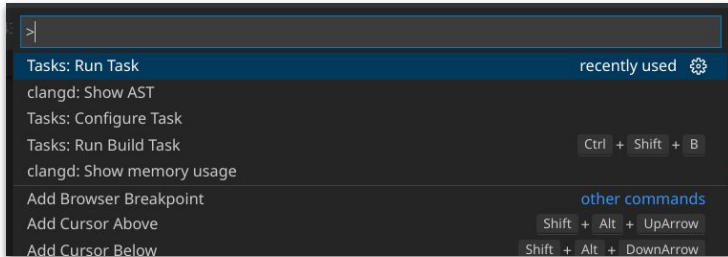
VSCodium en sistemas embebidos

Ejecutar tareas y depuración

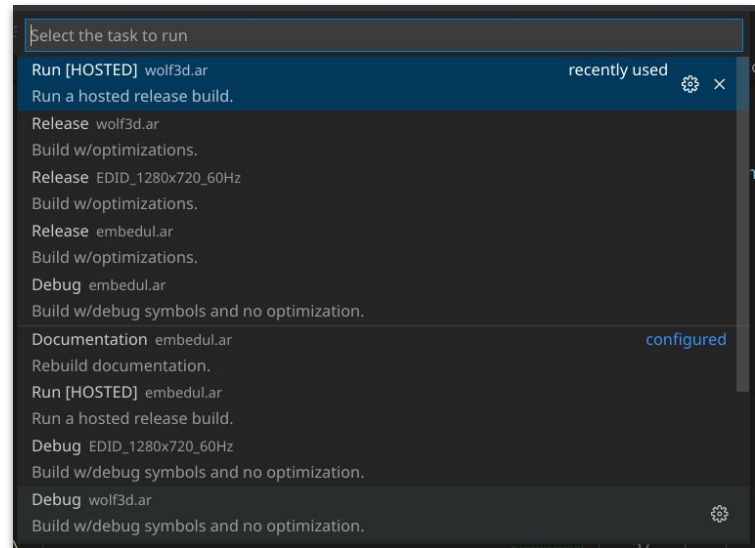


Ejecutar tareas

Command palette: **Ctrl+Shift+P**

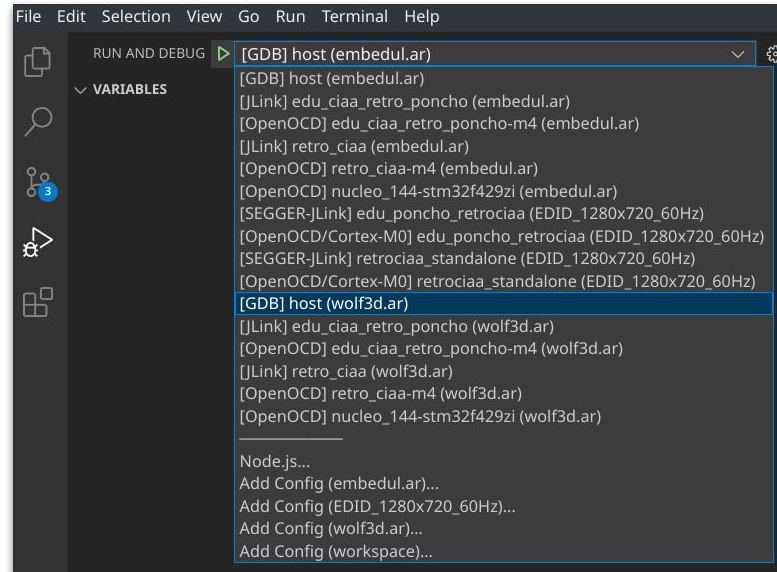


Build menu: **Ctrl+Shift+B**

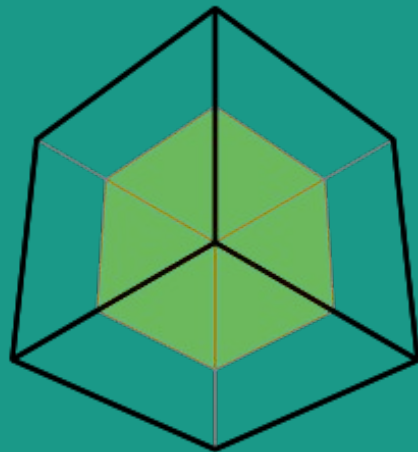


Ejecutar sesión de depuración

- Ctrl+Shift+D
- Elegir la configuración **deseada** en el proyecto **correcto**.
¡Difícil en Workspaces con varios proyectos abiertos!



embedul.ar



emb



embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](#))

embedul.ar



- Framework **open source** para sistemas embebidos.
- Multiplataforma, **targets diversos**.
(Nativo o cross-compiled para microcontroladores, web, etc...)
- Diseño modular **orientado a objetos**.
- Lenguaje de programación: **C11**.

<http://github.com/royconejo/embedul.ar>



embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

embedul.ar (cont'd)



- Capa de abstracción **transparente para la aplicación.**
Compila sin modificaciones en cualquier target.
- **Licencia zlib/libpng.**
Considerada “free software” por la Debian Free Software Guidelines, Open Source Initiative y Free Software Foundation.
- Sistema de **build anidado**, construido sobre GNU Make.



embedul.ar logo author: Santiago Germino ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

embedul.ar (cont'd)



- Solo **Stack**. No utiliza memoria **Heap**.
No malloc(), free(), etc.
- Soporta codificación **UTF-8** en todas sus funciones.
こんにちは世界!
- Programación defensiva.
Por ejemplo, Bounds checking en cada función.



embedul.ar (cont'd)



- Apunta a cumplir rigurosos estándares de **sistemas críticos**.
MISRA-C, CERT C.
- Contenedores básicos funcionan sobre **memoria estática**.
- Tipos de dato “Variant”.
- Funciones con **cantidad automática** y **tipo automático** de parámetros, validados en tiempo de compilación (sí, en “C”).




embedul.ar (cont'd)



- **Documentación.**



embedul.ar

Sistema de build



Sistema de build

```
[MK] (0|Makefile)
[MK] (1|system.mk) Welcome to http://embedul.ar build system
[MK] (1|system.mk) Date/time is '2022-08-19T06:01-03:00'
[MK] (1|system.mk) Build system ready
[MK] (1|system.mk) EOF
[MK] (0|Makefile) Including 'build.mk'
[MK] (1|build.mk) Build not configured yet
[MK] (1|build.mk) Including 'config.mk'
[MK] (2|config.mk) App. name 'wolf3d.ar'
[MK] (2|config.mk) Requested target shortcut 'retro_ciaa-m4'
[MK] (2|config.mk) Using makefiles 'target/shortcut/retro_ciaa-m4 lib/3rd_party/fatfs vcs/git toolchain/gcc'
[MK] (2|config.mk) Including 'target/shortcut/retro_ciaa-m4.mk'
[MK] (3|retro_ciaa-m4.mk) Including 'lib/arch/arm-cortex/lpc/board/retro_ciaa-m4.mk'
[MK] (4|retro_ciaa-m4.mk) Including 'lib/arch/arm-cortex/lpc/shared/lpc4337-m4.mk'
[MK] (5|lpc4337-m4.mk) Including 'cpu/arch/arm-cortex/lpc/lpc4337jbd144_m4.mk'
[MK] (6|lpc4337jbd144_m4.mk) Declaring chip manufacturer 'lpc', family '43', model '37', variant 'jbd144'
[MK] (6|lpc4337jbd144_m4.mk) Including 'cpu/arch/arm-cortex/m4f.mk'
[MK] (7|m4f.mk) Including 'cpu/arch/arm-cortex/m4f.mk'
[MK] (8|m4f.mk) EOF
[MK] (7|m4f.mk) EOF
[MK] (6|lpc4337jbd144_m4.mk) EOF
[MK] (5|lpc4337-m4.mk) Including 'lib/arch/arm-cortex/lpc/shared/lpc4337.mk'
[MK] (6|lpc4337.mk) Including 'target/check.mk'
[MK] (7|check.mk) Using target 'retro_ciaa'
[MK] (7|check.mk) EOF
[MK] (6|lpc4337-m4.mk) EOF
[MK] (5|lpc4337-m4.mk) EOF
[MK] (4|retro_ciaa-m4.mk) Including 'lib/embedul.ar.mk'
[MK] (5|embedul.ar.mk) Declaring library 'embedul.ar'
[MK] (5|embedul.ar.mk) Locate 'embedul.ar' in '/home/royconejo/projects/software/embedded/embedul.ar/source/core'
[MK] (5|embedul.ar.mk) Visual boot enabled
[MK] (5|embedul.ar.mk) Using CORE modules 'anim array bitfield cyclic device main object sequence utf8 variant'
[MK] (5|embedul.ar.mk) Using SUBSYSTEMS 'sound video'
[MK] (5|embedul.ar.mk) Using FatFs custom disks and config
[MK] (5|embedul.ar.mk) Exporting '2' user defined, '3' default config params
[MK] (5|embedul.ar.mk) Config param 'VISUAL_BOOT=1'
[MK] (5|embedul.ar.mk) Config param 'INPUT_SWITCH_ACTION=0'
[MK] (5|embedul.ar.mk) Config param 'SPLASH_THEME L1=lock glyph'
[MK] (5|embedul.ar.mk) Config param 'SPLASH_THEME L2=none'
[MK] (5|embedul.ar.mk) Config param 'SPLASH_THEME L3=c64'
[MK] (5|embedul.ar.mk) EOF
```

Sistema de build (cont'd)

```
[MK] (5|embedul.ar.mk) EOF
[MK] (4|retro_ciaa-m4.mk) Including 'lib/arch/arm-cortex/lpc/lpcopen.mk'
[MK] (5|lpcopen.mk) Locate 'NXP LPCOpen' in '/home/royconejo/projects/software/embedded/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/lpcopen'
[MK] (5|lpcopen.mk) Using chip drivers 'adc chip clock evrt gpdma gpio i2c i2cm i2s rtc sdif sdmmc ssp sysinit uart'
[MK] (5|lpcopen.mk) EOF
[MK] (4|retro_ciaa-m4.mk) Including 'lib/3rd_party/sfmt.mk'
[MK] (5|sfmt.mk) Locate 'Fast Mersenne Twister' in '/home/royconejo/projects/software/embedded/embedul.ar/source/3rd_party/sfmt-1.5.1'
[MK] (5|sfmt.mk) Using Fast Mersenne Twister with period = 607
[MK] (5|sfmt.mk) EOF
[MK] (4|retro_ciaa-m4.mk) EOF
[MK] (3|retro_ciaa-m4.mk) EOF
[MK] (2|config.mk) Including 'lib/3rd_party/fatfs.mk'
[MK] (3|fatfs.mk) Locate 'ChaN FatFS' in '/home/royconejo/projects/software/embedded/embedul.ar/source/3rd_party/fatfs'
[MK] (3|fatfs.mk) EOF
[MK] (2|config.mk) Including 'vcs/git.mk'
[MK] (3|git.mk) Application does not use GIT
[MK] (3|git.mk) Framework version: 'master/4daa64c2'
[MK] (3|git.mk) EOF
[MK] (2|config.mk) Including 'toolchain/gcc.mk'
[MK] (3|gcc.mk) GNU toolchain selected
[MK] (3|gcc.mk) CHIP is 'lpc4337jbd144'
[MK] (3|gcc.mk) Appending 16 CHIP-based CFLAGS directives
[MK] (3|gcc.mk) CPU model is 'cortex-m4'
[MK] (3|gcc.mk) Including 'toolchain/gcc/cross-cortex-m.mk'
[MK] (4|cross-cortex-m.mk) TARGET 'retro_ciaa' is cross-compiled
[MK] (4|cross-cortex-m.mk) Cross compiler toolchain prefix 'arm-none-eabi-'
[MK] (4|cross-cortex-m.mk) Freestanding (bare metal) target
[MK] (4|cross-cortex-m.mk) EOF
[MK] (3|gcc.mk) Toolchain version '10.3.1'
[MK] (3|gcc.mk) DEBUG 'no', OLEVEL '3'
[MK] (3|gcc.mk) EOF
[MK] (2|config.mk) EOF
[MK] (1|build.mk) Build root './build/retro_ciaa/arm-none-eabi-gcc/cortex-m4_03'
[MK] (1|build.mk) Flash tool 'no'
[MK] (1|build.mk) Including 'flash/no.mk'
[MK] (2|no.mk) EOF
[MK] (1|build.mk) EOF
[MK] (0|Makefile) EOF
[RM] ./build/retro_ciaa/arm-none-eabi-gcc/cortex-m4_03/embedul.ar/source/arch/arm-cortex/shared/systick.o
[RM] ./build/retro_ciaa/arm-none-eabi-gcc/cortex-m4_03/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/lpcopen/boards/retro_ciaa/board.o
[RM] ./build/retro_ciaa/arm-none-eabi-gcc/cortex-m4_03/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/lpcopen/boards/retro_ciaa/board_sysinit.o
[RM] ./build/retro_ciaa/arm-none-eabi-gcc/cortex-m4_03/embedul.ar/source/arch/arm-cortex/lpc/18xx_43xx/lpcopen/shared/i2cm.o
```




Makefile simple

```
embedul.ar > M Makefile
1 include $(LIB_EMBEDULAR_PATH)/embedul.ar/makefiles/system.mk
2
3 LIB_EMBEDULAR_CONFIG := VISUAL_BOOT=1 \
4                         SPLASH_THEME_L3=nyan_cat
5
6 BUILD_LIBS += 3rd_party/fatfs
7
8 APP_OBJS += \
9            ./testing/main.o
10
11 $(BUILD)
12
```

Makefile “no tan simple”

```
wolf3d.ar > M Makefile
1 include $(LIB_EMBEDULAR_PATH)/embedul.ar/makefiles/system.mk
2
3 LIB_EMBEDULAR_CONFIG := VISUAL_BOOT=1 \
4     INPUT_SWITCH_ACTION=0
5
6 BUILD_LIBS += 3rd_party/fatfs
7
8 # CFLAGS += -DSPANISH
9
10 lpc4337jbd144_CFLAGS += -DBOARD_RETRO_CIAA_DISABLE_TCP_SERVER
11
12 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_GRCHUNK_PIC_332='CC_Section(".rodata.$$Flash2")'
13 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_MAPSEGS_PLANE_0='CC_Section(".rodata.$$Flash2")'
14 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_MAPSEGS_PLANE_1='CC_Section(".rodata.$$Flash2")'
15 #lpc4337jbd144_CFLAGS += -DRODATA_SECTION_AUDIOSEGS_MUSIC_ADLIB='CC_Section(".rodata.$$Flash2")'
16 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_AUDIOSEGS_SOUND_ADLIB='CC_Section(".rodata.$$Flash2")'
17 #lpc4337jbd144_CFLAGS += -DRODATA_SECTION_TEXTURE_PAGE_332='CC_Section(".rodata.$$Flash2")'
18 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_SPRITE_PAGE_332='CC_Section(".rodata.$$Flash2")'
19 #lpc4337jbd144_CFLAGS += -DBSS_SECTION_OBJLIST='CC_Section(".bss.$$RamAHB_ETB16")'
20 lpc4337jbd144_CFLAGS += -DBSS_SECTION_TILEMAP='CC_Section(".bss.$$RamAHB_ETB16")'
21 lpc4337jbd144_CFLAGS += -DBSS_SECTION_ACTORAT='CC_Section(".bss.$$RamLoc40")'
22 lpc4337jbd144_CFLAGS += -DBSS_SECTION_STATOBJLIST='CC_Section(".bss.$$RamAHB16")'
23 # lpc4337jbd144_CFLAGS += -DBSS_SECTION_VISLIST='CC_Section(".bss.$$RamAHB16")'
24 #lpc4337jbd144_CFLAGS += -DBSS_SECTION_AREACONNECT='CC_Section(".bss.$$RamAHB16")'
25 #lpc4337jbd144_CFLAGS += -DBSS_SECTION_WALLHEIGHT='CC_Section(".bss.$$RamAHB16")'
26 #lpc4337jbd144_CFLAGS += -DBSS_SECTION_D00ROBJLIST='CC_Section(".bss.$$RamAHB16")'
27 #lpc4337jbd144_CFLAGS += -DBSS_SECTION_SPOTVIS='CC_Section(".bss.$$RamAHB16")'
28 #lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_TILE_hud='CC_Section(".rodata.$$Flash2")'
29 #lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_TILEMAP_hud='CC_Section(".rodata.$$Flash2")'
```

Makefile “no tan simple” (cont'd)

```
30 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_SFX_SoundChunks_0='CC_Section( ".rodata.$$Flash2" )'
31 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_SFX_SoundChunks_1='CC_Section( ".rodata.$$Flash2" )'
32 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_SFX_SoundChunks_2='CC_Section( ".rodata.$$Flash2" )'
33 lpc4337jbd144_CFLAGS += -DBSS_SECTION_RETROCIAA_SYSTEM='CC_Section( ".bss.$$RamAHB16" )'
34 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_RETROCIAA_SPLASH_THEME='CC_Section( ".rodata.$$Flash2" )'
35 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_TILEMAP_nyancat='CC_Section( ".rodata.$$Flash2" )'
36 lpc4337jbd144_CFLAGS += -DRODATA_SECTION_ASSETS_TILE_nyancat='CC_Section( ".rodata.$$Flash2" )'
37
38
39 # Project files to build
40 APP_OBJS += \
41 ./retroport.o \
42 ./retroport_main.o \
43 ./id_ca.o \
44 ./id_in.o \
45 ./id_pm.o \
46 ./id_sd.o \
47 ./id_us_1.o \
48 ./id_vh.o \
49 ./id_vl.o \
50 ./wl_act1.o \
51 ./wl_act2.o \
52 ./wl_agent.o \
53 ./wl_debug.o \
54 ./wl_draw.o \
55 ./wl_game.o \
56 ./wl_inter.o \
57 ./wl_main.o \
58 ./wl_menu.o \
59 ./wl_play.o \
60 ./wl_state.o \
61 ./wl_text.o \
62 ./wl_def.o \
63 ./rom/wl1/rom.o
64
65 $(BUILD)
```

embedul.ar

Diseño modular orientado a objetos





Diseño modular orientado a objetos

- Objetos conceptualmente *devices* o dispositivos.
- El framework define la clase base y una interfaz (abstract base class).
- Los *drivers* implementan la interfaz privada del *device*.
- La implementación del *driver* se invoca a través del *device* (inheritance).



Diseño modular orientado a objetos (cont'd)

- Las implementaciones dependientes de la plataforma, o las dependientes de librerías puntuales, están encapsuladas en sus propias definiciones, -y además-
- La clase base de cualquier **device** debe ser tratada como estructuras opacas (encapsulation).

En "C" hay formas de volverlas realmente opacas, pero no pregunten.



Diseño modular orientado a objetos (cont'd)

- Mediante un *pointer type cast* de un `driver` a su tipo de `device`, se accede igualmente a los servicios de ese objeto. También funciona al revés (polymorphism).
- Un tipo de `device` puede tener varias instancias, o una sola (singleton).
- Cualquier `device` puede tener múltiples implementaciones.

embedul.ar

Listado de devices





BOARD

Abstracción del sistema.

“El sistema” puede ser un microcontrolador, una aplicación corriendo en un sistema operativo o en una página web, etc.

*Gracias a esto, embedul.ar soporta **targets** **diversos**.*



PACKET

Abstracción de transmisión o recepción de **datos estructurados**.

Los “datos estructurados” pueden ser una comunicación por TCP/IP, SPI, I2C, etc.

*Es decir, embedul.ar **abstrae** tanto protocolos de comunicación de software y de hardware, como SPI e I2C.*



RAWSTOR

Abstracción de lectura y escritura de unidades de información (**sectores**) en dispositivos de **almacenamiento**.

Los “dispositivos de almacenamiento” pueden ser cualquier cosa.

embedul.ar abstrae el acceso a cualquier dispositivo de almacenamiento, incluso a través de fatfs.



SOUND

Abstracción de dispositivo de **sonido**.

Soporta efectos de sonido y música BGM
leída mediante streaming desde un
RAWSTOR.

*No importa que el dispositivo implementado sea
un DAC o la librería SDL.*



VIDEO

Abstracción de dispositivo de **video**.

Además del driver dependiente de la plataforma, embedul.ar incluye:

- Funciones para convertir y dibujar sprites animados y con transparencias.
- Áreas de clipping de pantalla.
- Dibujo de texto UTF-8.
- Lo básico, líneas, rectángulos...

Todo esto **independiente** de la plataforma.



IO

Abstracción de dispositivo de
entrada/salida.

Dispositivos con cualquier cantidad de entradas y/o salidas, digitales o analógicas. Se pueden implementar los **clásicos**:

- Teclado.
- Mouse.
- Joysticks.

Pero también cualquier tipo de **sensores**:

- Acelerómetro.
- Giróscopo.
- Temperatura.
- Humedad.
- Luz.

O **actuadores**:

- Motores.
- Luces LED, incluso SSL.

Todo en el mismo o en diferentes drivers.



RANDOM

Abstracción de dispositivo de **generador de números aleatorios**.

Por hardware o software, dependiente o no de la arquitectura o de librerías externas.

Como driver independiente de la plataforma, embedul.ar implementa el dispositivo "RANDOM_SFMT" (SIMD Fast Mersenne Twister).



STREAM

Abstracción de dispositivo de envío y recepción de datos **serie byte a byte**.

Puede ser “stdout/stdin” en sistemas hosted, o una implementación de una UART en sistemas embebidos.

*De este modo, embedul.ar **abstrae** toda comunicación serie.*

embedul.ar

Managers





Managers

- Se ocupan de gestionar los *drivers* de cada sistema.
- Durante la secuencia de inicio, el sistema o *board* registra sus *drivers* en los managers correspondientes.
- La aplicación invoca a los managers para obtener servicios de los *drivers*, generalmente de manera *indirecta*.



Managers

¿De manera indirecta?

- El board registra un driver y lo vincula a una etiqueta genérica de salida. Por ejemplo, la salida `IO_BOARD_OUTB_LED_RGB_RED` del driver `IO_BOARD` en la etiqueta `OUTPUT_Bit_Warning` del OUTPUT manager.
- La aplicación controla `OUTPUT_Bit_Warning` sin importar en qué sistema se ejecute.

embedul.ar

Listado de managers





INPUT

Input manager.

Gestiona las entradas del sistema (dispositivos IO).

Brinda a la aplicación información sobre eventos de press, click, doubleclick, hold, release para entradas digitales.

La aplicación puede obtener un valor guardado (buffered) hasta que el driver correspondiente se actualice, o uno inmediato, es decir, fuerza al driver a actualizarlo.



OUTPUT

Output manager.

Gestiona las salidas del sistema (dispositivos IO).

Brinda a la aplicación información sobre eventos de press, click, doubleclick, hold, release para entradas digitales.

Igual que Input, permite bufferear cambios en las salidas hasta que el driver mismo actualice (deferred) o efectivizarse al instante.



COMM

Communication manager.

Gestiona la comunicación del sistema (dispositivos STREAM y PACKET).

Al igual que INPUT y OUTPUT, el board registra y vincula los dispositivos de comunicación a una etiqueta.

Por ejemplo:

- `COMM_Stream_Log`
- `COMM_Packet_RS485`
- `COMM_Packet_SPI_Local`



STORAGE

Storage manager.

Gestiona el acceso a los dispositivos de almacenamiento RAWSTOR.

El board registra y vincula los dispositivos de almacenamiento asignándoles una etiqueta (rol). También puede “presentar” un dispositivo al manager y este determine el tipo (raw, fatfs) detectando el MBR y las particiones.

Además, el manager brinda un servicio de “cached elements”: archivos convertidos a almacenamiento RAW contiguo para acelerar la lectura.



LOG

Logging manager.

Mensajes del sistema a través de un STREAM registrado por la implementación del objeto board.

Mensajes de log



```

      embedded systems framework
      © 2018-2022 santiago germino
      http://embedul.ar
      2022-08-18T09:20:03:00 GCC 10.3.1 -0s
      master/4a4d07c6

      board
RETRO-CIAA

      retro-ciaa board
      http://www.retro-ciaa.com
      nxp lpc4337fet - dualcore arm cortex-m4/m0 - 204 mhz
      software video adapter - 720p hdmi output
      stack port expansion

      application name
      wolf3d.ar
      No GIT

      200| dev:board:retro-ciaa: board init sequence.
      209| manager:log: init.
      3  | * debug stream:"lpcopen usart" • max log items:6,
      10|x
```

Mensajes de log (cont'd)

```
14|
378| communication drivers.
387| dev:packet:lpc43xx i2c controller: init.
398| dev:packet:lpc43xx i2c controller: packet command.
4| @ set:"s|speed" * value:"400000"
14| dev:packet:lpc43xx i2c controller: command value set.
21|
38|
52|
441| storage drivers.
450| dev:rawstor:lpcopen sdmmc: init.
658| dev:rawstor:lpcopen sdmmc: storage media init.
382| @ media status:"ready"
387|
602|
1069| io level 2 drivers.
1078| dev:board:retro-ciaa: available expansion modules.
3| dev:board:retro-ciaa: x genesis module... ✓
1102| dev:io:lp5036 36-channel led driver: init.
1|
1119| dev:io:Huewheel-based marquee: init.
0|
1135| dev:io:dual genesis on pca9673: init.
0|
65| dev:board:retro-ciaa: x ssl module... ✗
73|
89|
1169| output manager summary.
```

devices		
id	driver	source
0	retro-ciaa board io	0
1	Huewheel-based marquee	0

mappings				
type	code	device	driver id	name

embedul.ar

Programación defensiva



Programación defensiva

- Modo paranóico.
- Detecta errores y bugs causados por entradas inesperadas.

Última línea de defensa, antes validación y testing!

```
32
33 void INPUT_Init (struct INPUT *const I)
34 {
35     BOARD_AssertState (!s_i);
36     BOARD_AssertParams (I);
37
38     OBJECT_Clear (I);
39
40     LOG_ContextBegin (I, LANG_INIT);
41     {
```

```
72 * :param CodePoint: Unicode code point from the basic multilingual plan
73 */
74 uint8_t UTF8_SetCodePoint (uint8_t *const Data, const uint32_t Octets,
75                             const uint16_t CodePoint)
76 {
77     BOARD_AssertParams (Data && Octets);
78
79     // UTF-8 one octet, same as 7-bit ASCII.
80     if (CodePoint < 128)
81     {
```

embedul.ar

C11





C11

- Se utiliza ampliamente C11.
- Los autoparams tienen un costo
_Generic, _Static_assert...
- Compila como C11, no como C++ estándar.
- Sin embargo, se puede invocar desde “C++”.

¿Por qué “C”?

- Indiscutido en **bajo nivel** (o sistemas con severas restricciones).
- Multiplataforma facil.
Creado para portar sistemas operativos.
- Gente dedicada a bajo nivel lo entiende “bien”.
- Casi nadie usa **C11**.
- Se lleva mejor con **bare metal**, incluso sin *standard libraries*.
- Sus datos se importan o exportan fácil a cualquier otro lenguaje; muy fácil hacer bindings.

¿Y “C++”?

```
1  std::map<int, std::string> myMap;
2
3  if (auto [iter, succeeded] = myMap.insert(value); succeeded) {
4      useIter(iter);
5  }
```

Evalué que “C” era la herramienta **ideal**. Pero más allá de eso:

- Malentendidos sobre qué es o cómo aplicar “C++”.
¿**Superset** de “C”? Hace 40 años, tal vez. Hoy son lenguajes **distintos**.
https://www.stroustrup.com/bs_faq.html#C-is-subset
- C++17 son jeroglíficos para programador “C”.



¿Y “C++”? (cont'd)

- Overpowered para esta aplicación.
- En embedular no se usa Heap.
¿Qué hacer con el operador new?
- Escasa gente experta.
tampoco abundan expertos en puro “C”, pero es manejable.

¿Preguntas?

iMuchas gracias!



Auspiciantes

**SAMCRO
REMERAS**

