

Laboratorio III

*Nota: Trabajo práctico para la unidad curricular Sistemas Embebidos

Tomás Eduardo Conti

Ing. Mecatrónica

Universidad Tecnológica del Uruguay

Fray Bentos, Río Negro, Uruguay

tomas.conti@estudiantes.utec.edu.uy

Pablo Kevin Pereira

Ing. Mecatrónica

Universidad Tecnológica del Uruguay

Fray Bentos, Río Negro, Uruguay

pablo.pereira.p@estudiantes.utec.edu.uy

Resumen—Se presenta la implementación modular de un sistema embebido de riego automatizado, desarrollada íntegramente sobre plataforma ESP32 utilizando FreeRTOS como sistema operativo en tiempo real. La programación fue realizada en base a una implementación física previamente desarrollada, replicando el comportamiento observado, reorganizando la disposición de pines (por diferencia de microcontrolador), la lógica de tareas y los requerimientos funcionales del sistema original.

El sistema fue diseñado mediante la separación de tareas concurrentes, cada una con prioridades específicas según su criticidad funcional: lectura de sensores, control de riego, gestión de interfaz y lógica de usuario. La implementación resultante permite evaluar el comportamiento del sistema bajo condiciones de multitarea.

Este trabajo demuestra cómo una arquitectura basada en RTOS permite encapsular funciones críticas y preparar firmware robusto, confiable y listo para ser desplegado sobre hardware real sin necesidad de modificaciones estructurales.

Index Terms—Sistemas embebidos, riego automático, tiempo real, ESP32, instrumentación, FreeRTOS.

I. INTRODUCCIÓN

Los sistemas embebidos modernos demandan una gestión eficiente de recursos y tareas, especialmente cuando se busca mantener un comportamiento determinista ante eventos asincrónicos. FreeRTOS, como sistema operativo en tiempo real, provee una solución robusta para gestionar concurrencia, asignar prioridades y facilitar el diseño de arquitecturas diversas.

En el presente trabajo se implementó un sistema de riego automatizado utilizando la plataforma ESP32 con FreeRTOS, tomando como referencia directa una implementación física previa sobre microcontroladores convencionales. Se replicaron las funciones y tiempos medidos en el sistema original, adaptándolos a un modelo multitarea estructurado.

Cada funcionalidad —como la lectura de humedad, control de riego, actualización de pantalla y gestión de botones— fue encapsulada en tareas independientes, respetando las prioridades y frecuencias definidas en el análisis experimental previo. La separación de responsabilidades y la planificación de tareas permiten validar la transición hacia un modelo de programación en tiempo real, apto para ser desplegado directamente en hardware físico.

Este laboratorio se centró en la programación y estructuración del sistema bajo FreeRTOS, evaluando su comporta-

miento lógico y su compatibilidad con el diseño previamente implementado en condiciones reales.

II. MATERIALES Y MÉTODOS

El presente trabajo consistió exclusivamente en el desarrollo del firmware para un sistema de riego automatizado, basado en tareas concurrentes gestionadas mediante FreeRTOS. La implementación fue realizada íntegramente en entorno de simulación y desarrollo, respetando la estructura lógica, la disposición de pines y la planificación funcional derivada de una versión previamente implementada en hardware físico.

El sistema fue desarrollado utilizando los siguientes elementos:

- 1 × Microcontrolador ESP32
- 1 × Laptop con sistema operativo Linux Ubuntu
- Visual Studio Code con extensión para ESP-IDF
- 1 × Cable USB–MicroUSB para conexión y carga del firmware

Para el desarrollo del firmware se utilizó el entorno **ESP-IDF (Espressif IoT Development Framework)**, instalado y configurado en **Visual Studio Code** con extensiones específicas para programación, depuración y monitorización en ESP32. El código fue organizado en tareas independientes que modelan el comportamiento lógico del sistema original, y compilado utilizando las herramientas `idf.py build`, `idf.py flash` y `idf.py monitor`.

Cada tarea fue diseñada para representar funcionalidades reales del sistema físico previamente analizado, con el objetivo de obtener un firmware completamente operativo y listo para ser desplegado en condiciones reales sin necesidad de modificaciones adicionales.

III. METODOLOGÍA EXPERIMENTAL

El presente laboratorio se centró en la estructuración lógica y programación multitarea de un sistema embebido de riego automatizado utilizando FreeRTOS sobre ESP32. La metodología adoptada consistió en diseñar, implementar y probar el firmware completo basado en tareas concurrentes, siguiendo la planificación funcional y temporal desarrollada en el Laboratorio II.

El desarrollo se organizó en módulos independientes, cada uno implementado como una tarea de FreeRTOS, y todos integrados dentro del archivo `Lab3_main.cpp`. Las tareas

fueron diseñadas con distintas prioridades según su criticidad, y se ejecutaron sobre el núcleo principal (core 0) del ESP32, utilizando el puerto SMP por defecto.

III-A. Estructura del proyecto

El proyecto fue organizado en un repositorio con estructura modular, siguiendo las convenciones del entorno de desarrollo ESP-IDF. La jerarquía de carpetas y archivos es la siguiente:

Lab3_RTOS/	Raíz del repositorio del proyecto
.vscode/	Configuraciones de Visual Studio Code
c_cpp_properties.json	Ajustes IntelliSense
launch.json	Configuración de depuración
settings.json	Preferencias de workspace
tasks.json	Tareas de build, flash y monitor
build/	Salida de compilación (auto-generada)
CMakeLists.txt	CMake principal (plantilla ESP-IDF)
sdkconfig	Archivo de configuración del proyecto
main/	Carpeta principal: firmware y librerías
CMakeLists.txt	Define los fuentes de main y sus dependencias
Lab3_main.cpp	Código principal con <code>app_main()</code> y tareas
LCD.hpp	Interfaz de la librería de control de LCD
LCD.cpp	Implementación de las funciones de la librería LCD

Cuadro 1

ESTRUCTURA DEL PROYECTO LAB3_RTOS CON ESP-IDF Y FREERTOS

El directorio `main/` contiene la lógica principal del sistema embebido. Allí se definen tanto la función `app_main()` como las tareas concurrentes, junto con una librería desacoplada para el control de la pantalla LCD.

El archivo `sdkconfig` almacena los parámetros seleccionados mediante `menuconfig`, incluyendo configuraciones de FreeRTOS, UART, periféricos y otras opciones del sistema.

La carpeta `main/` contiene la lógica principal del sistema. El archivo `Lab3_main.cpp` define las tareas, inicializa los periféricos y coordina la ejecución del sistema.

III-A1. Contenido del directorio `main/`: La carpeta `main/` contiene el firmware central del sistema embebido. En ella residen tanto el archivo principal `Lab3_main.cpp` como la librería desacoplada de pantalla LCD. A continuación se resumen los archivos clave:

- `CMakeLists.txt`: declara los archivos fuente (`Lab3_main.cpp`, `LCD.cpp`), especifica los directorios de cabeceras (`INCLUDE_DIRS`) y las dependencias requeridas (`driver`, `freertos`, `esp_adc`, `vfs`).
- `Lab3_main.cpp`: contiene la función `app_main()` y la lógica general del sistema. Inicializa periféricos (ADC, GPIO), y crea las tareas `sensor_task`, `switch_task`, `control_task` y `lcd_task`.

- `LCD.hpp`: define la interfaz pública de la pantalla LCD, incluyendo funciones como `lcdInit()`, `lcdClear()`, `lcdSetCursor()`, `lcdPrint()`, entre otras. Este diseño desacopla completamente la lógica de visualización del código principal.
- `LCD.cpp`: implementa internamente las funciones declaradas en `LCD.hpp`, permitiendo una organización más clara y mantenible del sistema.

III-B. Identificación del puerto USB y configuración del monitor serie

La conexión entre la laptop y el microcontrolador ESP32 se realizó a través de un puerto USB estándar, utilizando el convertidor USB-UART integrado en el chip CP210x. Para verificar que el dispositivo fue correctamente reconocido por el sistema, se utilizó el comando `lsusb`, observándose el identificador correspondiente a *Silicon Labs CP210x UART Bridge*.

Posteriormente, se utilizó el comando `ls /dev/ttyUSB*` para identificar el puerto serie asignado por el sistema operativo. En este caso, se detectó correctamente como `/dev/ttyUSB0`. La conexión al puerto se puede ver en la Figura 1.

En el entorno Visual Studio Code, Figura 2, la extensión ESP-IDF permitió seleccionar dicho puerto tanto para la carga del firmware como para la apertura del monitor serie. La identificación completa se mostró como `/dev/ttyUSB0 ESP32-D0WD-V3 (revision v3.1)`.

Esta configuración fue necesaria para habilitar las funciones de `idf.py flash` y `idf.py monitor`, fundamentales para la carga y depuración del firmware durante el desarrollo del sistema.

```

pablo_kevin@Apolo-Nitro-PK: ~
$ lsusb
Command 'lsusb' not found, did you mean:
  command 'lsusb' from deb usbutils (1:017-1)
Try: sudo apt install <deb name>
pablo_kevin@Apolo-Nitro-PK: ~
$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 002: ID 046d:c52b Logitech, Inc. Unifying Receiver
Bus 003 Device 003: ID 0408:403a Quanta Computer, Inc. ACER HD User Facing
Bus 003 Device 004: ID 8087:0026 Intel Corp. AX201 Bluetooth
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
pablo_kevin@Apolo-Nitro-PK: ~
$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 002: ID 046d:c52b Logitech, Inc. Unifying Receiver
Bus 003 Device 003: ID 0408:403a Quanta Computer, Inc. ACER HD User Facing
Bus 003 Device 004: ID 8087:0026 Intel Corp. AX201 Bluetooth
Bus 003 Device 007: ID 10c4:ea60 Silicon Labs CP210x UART Bridge
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
pablo_kevin@Apolo-Nitro-PK: ~
$ ls /dev/ttyUSB* /dev/ttyACM*
ls: cannot access '/dev/ttyACM*': No such file or directory
/dev/ttyUSB0
pablo_kevin@Apolo-Nitro-PK: ~

```

Figura 1. Identificación del puerto USB en terminal

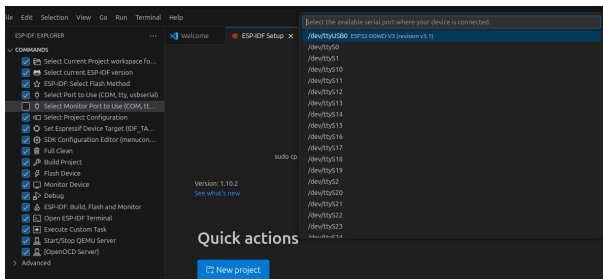


Figura 2. Identificación del puerto USB en Visual Studio Code

III-B1. Ejecución en núcleo (Core): El microcontrolador ESP32 dispone de dos núcleos de CPU:

- **PRO CPU (core 0):** núcleo principal, donde se ejecutan por defecto la función `app_main()` y las tareas creadas con `xTaskCreate()`.
- **APP CPU (core 1):** núcleo secundario, disponible para tareas explícitamente asignadas mediante `xTaskCreatePinnedToCore()`.

El framework ESP-IDF utiliza un puerto de multiprocesamiento simétrico (SMP), lo que habilita el uso de ambos núcleos para distribuir la carga de trabajo. No obstante, en este laboratorio no fue necesario explotar esa capacidad, ya que las tareas definidas presentan una carga muy liviana y no requieren garantías estrictas de tiempo real.

Por esta razón, todas las tareas fueron creadas mediante `xTaskCreate()`, ejecutándose por defecto sobre el **core 0 (PRO CPU)**. El núcleo 1 permanece libre y sin tareas asignadas durante toda la ejecución.

Se considera que el core 1 podría ser utilizado en futuras extensiones del sistema, por ejemplo para implementar tareas de conectividad IoT o manejo de comunicación inalámbrica, operando en paralelo al sistema de riego.

Esta arquitectura modular permite escalar el sistema sin modificar las tareas ya existentes ni comprometer su desempeño.

III-C. Descripción del archivo `Lab3_main.cpp`

El archivo `Lab3_main.cpp` contiene la lógica principal del firmware desarrollado para ESP32 bajo FreeRTOS. En él se definen las cabeceras necesarias, las variables compartidas entre tareas, la inicialización de periféricos, y la creación de tareas concurrentes que conforman el sistema de riego automatizado.

III-C1. Inclusión de cabeceras: Al inicio del archivo se incluyen todas las cabeceras necesarias para el funcionamiento del sistema:

- `FreeRTOS.h`, `task.h`: proveen las funciones básicas del sistema operativo en tiempo real, necesarias para la creación y manejo de tareas.
- `gpio.h`, `adc_oneshot.h`, `uart.h`, `esp_vfs_dev.h`: permiten configurar y utilizar los periféricos del ESP32 como entradas/salidas digitales, conversión analógica, puerto serie y sistema de archivos virtual.
- `LCD.hpp`: define la interfaz pública de la librería encargada de controlar la pantalla LCD.

- `stdio.h`: cabecera estándar de C, utilizada para funciones como `printf()`.

Estas cabeceras encapsulan tanto las funcionalidades específicas del hardware como los servicios del sistema operativo, y permiten construir una aplicación modular, portable y estructurada.

III-C2. Definición de pines y variables globales: El archivo `Lab3_main.cpp` define un conjunto de constantes y variables globales necesarias para el funcionamiento del sistema. Estas definiciones establecen la configuración de hardware virtual sobre la que opera el firmware y permiten el intercambio de información entre tareas.

III-C2a. Pines de entrada/salida: Se utilizaron constantes del tipo `constexpr` para definir los pines correspondientes a cada función:

- `MOISTURE_PIN`: pin ADC asignado al sensor de humedad.
- `RELAY_PIN`: salida digital para activar o desactivar la bomba de riego.
- `MODE_BTN`, `PERIOD_BTN`, `THRESH_BTN`: entradas digitales con pull-up asociadas a los botones de configuración del usuario.
- `LCD_RS`, `LCD_EN`, `LCD_D4-D7`: pines asignados a la pantalla LCD en modo 4 bits.

III-C2b. Variables globales: Se declararon variables globales `volatile` para garantizar que las operaciones realizadas desde distintas tareas se reflejen correctamente a nivel de compilador. Estas variables mantienen el estado compartido del sistema:

- `mode`: modo operativo del sistema (manual o automático).
- `irrigationPeriod`: periodo configurado de riego automático.
- `irrigationTime`: duración del tiempo de riego por ciclo.
- `moistureThreshold`: umbral de humedad utilizado para la decisión de activación.
- `currMoisture`: última lectura del sensor de humedad.

III-C2c. Handle del ADC: Se define un identificador global `adc_handle` del tipo `adc_oneshot_unit_handle_t`, utilizado para realizar las lecturas únicas de conversión analógica desde el periférico ADC del ESP32.

Esta sección del código permite centralizar la configuración de pines y estados, facilitando su modificación y mantención, al tiempo que prepara el contexto necesario para el correcto funcionamiento multitarea del sistema.

III-C3. Función `init_uart()`: La función `init_uart()` se encarga de configurar el canal de comunicación UART0 del ESP32 para depuración y monitoreo a través del puerto serie. Esta configuración es fundamental para visualizar los mensajes de `printf()` o `ESP_LOGx` durante el desarrollo.

La función realiza los siguientes pasos:

- Configura la velocidad del puerto serie a 115200 baudios.

- Asocia el controlador UART al sistema de archivos virtual de ESP-IDF (VFS), permitiendo redirigir la salida estándar al monitor serie.
- Instala el driver UART utilizando `uart_driver_install()` con una cola de tamaño predeterminado.

Este mecanismo es utilizado a lo largo del sistema para imprimir por consola información útil como los valores de humedad, los estados de la bomba, los cambios de modo y otras variables internas. Facilita la depuración y el análisis de funcionamiento sin necesidad de hardware adicional, y resulta especialmente valioso durante la etapa de pruebas y validación.

III-C4. Función `app_main()`: La función `app_main()` constituye el punto de arranque del sistema en el entorno ESP-IDF. En ella se realiza la inicialización de todos los periféricos y estructuras necesarias, seguida de la creación de tareas FreeRTOS que ejecutan la lógica concurrente del sistema.

Las acciones principales realizadas en esta función se detallan a continuación:

III-C4a. Inicialización del GPIO de la bomba: Se configura el pin asignado al relé que controla la bomba de riego como salida digital, y se inicializa en estado bajo para asegurar que la válvula permanezca cerrada al inicio del programa.

III-C4b. Inicialización de botones: Mediante una función lambda llamada `init_button()`, se configura cada uno de los pines de entrada correspondientes a los botones del usuario con resistencias de pull-up internas activadas. Esta abstracción permite reutilizar el mismo procedimiento para los tres botones del sistema.

III-C4c. Inicialización del ADC (modo one-shot): Se inicializa el ADC del ESP32 utilizando el modo `one-shot`, lo que permite realizar lecturas puntuales bajo demanda. El procedimiento incluye:

- Creación de una unidad de ADC mediante `adc_oneshot_new_unit()`.
- Configuración del canal correspondiente al pin del sensor de humedad, estableciendo atenuación y resolución deseadas.

El identificador del ADC se almacena en la variable global `adc_handle` para ser utilizada posteriormente desde la tarea de adquisición de humedad.

III-C4d. Inicialización de la pantalla LCD: Se llama a la función `lcdInit(...)` definida en la librería `LCD.hpp`, la cual configura el bus de datos en modo 4 bits y posiciona el cursor, dejando la pantalla lista para mostrar información durante la ejecución.

III-C4e. Creación de tareas FreeRTOS: Finalmente, se crean cuatro tareas concurrentes mediante la función `xTaskCreate()`. Cada tarea es asignada con una prioridad específica, definida en base a su criticidad funcional:

- `sensor_task`: prioridad 7 (lectura del sensor de humedad cada 4 s).
- `switch_task`: prioridad 6 (detección de pulsaciones y modificación de parámetros).

- `control_task`: prioridad 5 (decisión de activación de la bomba).
- `lcd_task`: prioridad 4 (actualización de la pantalla LCD cada 3 s).

Todas las tareas se ejecutan sobre el núcleo 0 (core 0), ya que no se utilizó `xTaskCreatePinnedToCore()`. Esto resulta suficiente dada la baja carga de procesamiento y la ausencia de tareas críticas con restricciones temporales estrictas.

Esta función resume el inicio del sistema y permite que cada módulo opere de forma desacoplada bajo el modelo de ejecución multitarea provisto por FreeRTOS.

III-C5. Implementación de tareas: Cada tarea definida en el sistema cumple una función específica del sistema de riego automatizado. A continuación, se detalla la lógica de funcionamiento y las interacciones principales de cada una.

III-C5a. `sensor_task`: Esta tarea se ejecuta periódicamente cada 4 segundos. Utiliza el ADC configurado en modo one-shot para adquirir el valor de humedad del suelo. La lectura se realiza mediante la función `adc_oneshot_read()`, y el valor obtenido se almacena en la variable global `currMoisture`.

La tarea incorpora una demora mediante `vTaskDelay()` para espaciar las lecturas, y actúa como fuente de datos para el sistema.

III-C5b. `switch_task`: La tarea `switch_task` monitorea el estado de tres botones físicos conectados a pines con pull-up interno. Permite cambiar parámetros del sistema como:

- Modo de riego (manual / automático)
- Periodo de riego automático
- Umbral de humedad para activación

Para evitar lecturas erróneas por rebote, se utilizan retardos con `vTaskDelay()` entre lecturas. Los botones son escaneados de forma cíclica, y las variables globales son actualizadas en función de las pulsaciones detectadas.

III-C5c. `control_task`: Esta tarea es el núcleo de la lógica de riego. Según el modo de funcionamiento seleccionado y los valores actuales de humedad, decide si activar o desactivar el relé que controla la bomba. En modo automático, mantiene un contador interno que gestiona la temporización del riego.

El relé es activado o desactivado mediante la escritura digital sobre el pin configurado como salida, respetando el ciclo de riego y los umbrales definidos.

III-C5d. `lcd_task`: La tarea `lcd_task` actualiza el contenido de la pantalla LCD con la información del sistema: modo actual, humedad leída, umbral y temporizador de riego. La actualización se realiza cada 3 segundos utilizando funciones de la librería `LCD.cpp`.

Para evitar parpadeos o demoras innecesarias, se utiliza una estrategia de actualización optimizada, donde solo se sobrescriben los campos modificados. Esto reduce el tiempo total de escritura y permite mantener fluidez en la interfaz sin comprometer el rendimiento del sistema.

Cada tarea opera de manera desacoplada, compartiendo únicamente las variables necesarias. Gracias al diseño coope-

rativo y al uso controlado de prioridades, no se requiere sincronización explícita mediante semáforos o colas, dado que no existen condiciones de carrera ni accesos simultáneos problemáticos.

III-C5e. Nota sobre temporización en FreeRTOS: En el contexto de FreeRTOS, un *tick* es la unidad básica de tiempo utilizada por el sistema operativo para medir retardos, gestionar la planificación y administrar temporizadores. Cada tick es generado por una interrupción periódica de hardware, típicamente mediante el *SysTick timer*, a una frecuencia definida por la constante `configTICK_RATE_HZ` (por ejemplo, 1000 Hz = 1 ms por tick).

Cuando una tarea invoca `vTaskDelay(pdMS_TO_TICKS(500))`, el RTOS convierte ese retardo de milisegundos a ticks. Por ejemplo, con una frecuencia de tick de 1 kHz, 500 ms equivalen a 500 ticks.

El tick también es utilizado por el planificador para evaluar el estado de tareas bloqueadas, manejar timeouts en colas y semáforos, y activar temporizadores de software.

En resumen, el tick actúa como un “pulso” de referencia temporal constante, que permite al sistema operativo mantener una planificación precisa y determinista.

III-C5f. Consideración sobre el uso de `vTaskDelay()` y estabilidad del sistema: Durante el desarrollo se observó que, en ciertas configuraciones, la placa ESP32 se reiniciaba de forma inesperada. Tras analizar el comportamiento, se identificó que la causa era la ausencia de retardos en algunas tareas: al no incluir ningún `vTaskDelay()`, el procesador permanecía ocupado permanentemente en bucles activos, impidiendo que el sistema operativo liberara CPU o reiniciara el temporizador del watchdog.

Este comportamiento provoca que el watchdog se desborde, y como medida de protección, el ESP32 reinicia automáticamente el sistema. Por tanto, se concluye que en FreeRTOS (y otros RTOS similares) es esencial que toda tarea incluya, al menos, algún mecanismo de espera —por breve que sea— como `vTaskDelay()`, `vTaskDelayUntil()` o bloqueo sobre semáforos/colas. Esto no solo permite al planificador alternar entre tareas, sino que además evita que el sistema entre en condiciones de fallo por falta de respiro.

Este aprendizaje destaca la importancia de diseñar tareas que cedan tiempo de CPU explícitamente, aún en sistemas de baja carga.

IV. RESULTADOS

En esta sección se documentan los resultados obtenidos durante la ejecución del firmware desarrollado, validando el comportamiento funcional de las tareas y periféricos configurados. Dado que el sistema fue implementado íntegramente en entorno de programación, se utilizaron herramientas de depuración por consola y elementos integrados del hardware para verificar el funcionamiento esperado.

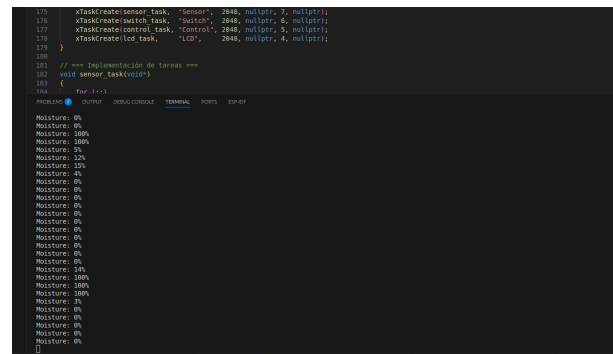
IV-A. Verificación de lectura del sensor de humedad

Se validó la funcionalidad del ADC en modo one-shot a través de la tarea `sensor_task`, imprimiendo el valor de humedad cada 4 segundos mediante la consola serie. La lectura fue realizada sin ningún sensor conectado, tocando manualmente el pin analógico con el dedo, lo cual permitió observar variaciones en la conductividad y cambios en la lectura.

A continuación se presenta un fragmento de la salida obtenida en el terminal:

```
Moisture: 100%
Moisture: 5%
Moisture: 12%
Moisture: 0%
Moisture: 14%
Moisture: 43%
Moisture: 22%
Moisture: 0%
```

Este comportamiento evidencia que el sistema es capaz de adquirir una señal analógica, procesarla correctamente y reflejarla en consola. En la Figura 3 se muestra la salida obtenida.



```
170 xTaskCreate(sensor_task, "Sensor", 2048, NULL, 1, &sensor_task);
171 xTaskCreate(switch_task, "Switch", 2048, NULL, 1, &switch_task);
172 xTaskCreate(control_task, "Control", 2048, NULL, 1, &control_task);
173 xTaskCreate lcd_task, "LCD", 2048, NULL, 1, &lcd_task);
174 }
175 // FreeRTOS configuration of tasks over
176 void main(void)
177 {
178     // Run
179     for (;;)
180     {
181         // Run
182         // Run
183         // Run
184         // Run
185         // Run
186         // Run
187         // Run
188         // Run
189         // Run
190         // Run
191         // Run
192         // Run
193         // Run
194         // Run
195         // Run
196         // Run
197         // Run
198         // Run
199         // Run
200         // Run
201         // Run
202         // Run
203         // Run
204         // Run
205         // Run
206         // Run
207         // Run
208         // Run
209         // Run
210         // Run
211         // Run
212         // Run
213         // Run
214         // Run
215         // Run
216         // Run
217         // Run
218         // Run
219         // Run
220         // Run
221         // Run
222         // Run
223         // Run
224         // Run
225         // Run
226         // Run
227         // Run
228         // Run
229         // Run
230         // Run
231         // Run
232         // Run
233         // Run
234         // Run
235         // Run
236         // Run
237         // Run
238         // Run
239         // Run
240         // Run
241         // Run
242         // Run
243         // Run
244         // Run
245         // Run
246         // Run
247         // Run
248         // Run
249         // Run
250         // Run
251         // Run
252         // Run
253         // Run
254         // Run
255         // Run
256         // Run
257         // Run
258         // Run
259         // Run
260         // Run
261         // Run
262         // Run
263         // Run
264         // Run
265         // Run
266         // Run
267         // Run
268         // Run
269         // Run
270         // Run
271         // Run
272         // Run
273         // Run
274         // Run
275         // Run
276         // Run
277         // Run
278         // Run
279         // Run
280         // Run
281         // Run
282         // Run
283         // Run
284         // Run
285         // Run
286         // Run
287         // Run
288         // Run
289         // Run
290         // Run
291         // Run
292         // Run
293         // Run
294         // Run
295         // Run
296         // Run
297         // Run
298         // Run
299         // Run
300         // Run
301         // Run
302         // Run
303         // Run
304         // Run
305         // Run
306         // Run
307         // Run
308         // Run
309         // Run
310         // Run
311         // Run
312         // Run
313         // Run
314         // Run
315         // Run
316         // Run
317         // Run
318         // Run
319         // Run
320         // Run
321         // Run
322         // Run
323         // Run
324         // Run
325         // Run
326         // Run
327         // Run
328         // Run
329         // Run
330         // Run
331         // Run
332         // Run
333         // Run
334         // Run
335         // Run
336         // Run
337         // Run
338         // Run
339         // Run
340         // Run
341         // Run
342         // Run
343         // Run
344         // Run
345         // Run
346         // Run
347         // Run
348         // Run
349         // Run
350         // Run
351         // Run
352         // Run
353         // Run
354         // Run
355         // Run
356         // Run
357         // Run
358         // Run
359         // Run
360         // Run
361         // Run
362         // Run
363         // Run
364         // Run
365         // Run
366         // Run
367         // Run
368         // Run
369         // Run
370         // Run
371         // Run
372         // Run
373         // Run
374         // Run
375         // Run
376         // Run
377         // Run
378         // Run
379         // Run
380         // Run
381         // Run
382         // Run
383         // Run
384         // Run
385         // Run
386         // Run
387         // Run
388         // Run
389         // Run
390         // Run
391         // Run
392         // Run
393         // Run
394         // Run
395         // Run
396         // Run
397         // Run
398         // Run
399         // Run
400         // Run
401         // Run
402         // Run
403         // Run
404         // Run
405         // Run
406         // Run
407         // Run
408         // Run
409         // Run
410         // Run
411         // Run
412         // Run
413         // Run
414         // Run
415         // Run
416         // Run
417         // Run
418         // Run
419         // Run
420         // Run
421         // Run
422         // Run
423         // Run
424         // Run
425         // Run
426         // Run
427         // Run
428         // Run
429         // Run
430         // Run
431         // Run
432         // Run
433         // Run
434         // Run
435         // Run
436         // Run
437         // Run
438         // Run
439         // Run
440         // Run
441         // Run
442         // Run
443         // Run
444         // Run
445         // Run
446         // Run
447         // Run
448         // Run
449         // Run
450         // Run
451         // Run
452         // Run
453         // Run
454         // Run
455         // Run
456         // Run
457         // Run
458         // Run
459         // Run
460         // Run
461         // Run
462         // Run
463         // Run
464         // Run
465         // Run
466         // Run
467         // Run
468         // Run
469         // Run
470         // Run
471         // Run
472         // Run
473         // Run
474         // Run
475         // Run
476         // Run
477         // Run
478         // Run
479         // Run
480         // Run
481         // Run
482         // Run
483         // Run
484         // Run
485         // Run
486         // Run
487         // Run
488         // Run
489         // Run
490         // Run
491         // Run
492         // Run
493         // Run
494         // Run
495         // Run
496         // Run
497         // Run
498         // Run
499         // Run
500         // Run
501         // Run
502         // Run
503         // Run
504         // Run
505         // Run
506         // Run
507         // Run
508         // Run
509         // Run
510         // Run
511         // Run
512         // Run
513         // Run
514         // Run
515         // Run
516         // Run
517         // Run
518         // Run
519         // Run
520         // Run
521         // Run
522         // Run
523         // Run
524         // Run
525         // Run
526         // Run
527         // Run
528         // Run
529         // Run
530         // Run
531         // Run
532         // Run
533         // Run
534         // Run
535         // Run
536         // Run
537         // Run
538         // Run
539         // Run
540         // Run
541         // Run
542         // Run
543         // Run
544         // Run
545         // Run
546         // Run
547         // Run
548         // Run
549         // Run
550         // Run
551         // Run
552         // Run
553         // Run
554         // Run
555         // Run
556         // Run
557         // Run
558         // Run
559         // Run
560         // Run
561         // Run
562         // Run
563         // Run
564         // Run
565         // Run
566         // Run
567         // Run
568         // Run
569         // Run
570         // Run
571         // Run
572         // Run
573         // Run
574         // Run
575         // Run
576         // Run
577         // Run
578         // Run
579         // Run
580         // Run
581         // Run
582         // Run
583         // Run
584         // Run
585         // Run
586         // Run
587         // Run
588         // Run
589         // Run
590         // Run
591         // Run
592         // Run
593         // Run
594         // Run
595         // Run
596         // Run
597         // Run
598         // Run
599         // Run
600         // Run
601         // Run
602         // Run
603         // Run
604         // Run
605         // Run
606         // Run
607         // Run
608         // Run
609         // Run
610         // Run
611         // Run
612         // Run
613         // Run
614         // Run
615         // Run
616         // Run
617         // Run
618         // Run
619         // Run
620         // Run
621         // Run
622         // Run
623         // Run
624         // Run
625         // Run
626         // Run
627         // Run
628         // Run
629         // Run
630         // Run
631         // Run
632         // Run
633         // Run
634         // Run
635         // Run
636         // Run
637         // Run
638         // Run
639         // Run
640         // Run
641         // Run
642         // Run
643         // Run
644         // Run
645         // Run
646         // Run
647         // Run
648         // Run
649         // Run
650         // Run
651         // Run
652         // Run
653         // Run
654         // Run
655         // Run
656         // Run
657         // Run
658         // Run
659         // Run
660         // Run
661         // Run
662         // Run
663         // Run
664         // Run
665         // Run
666         // Run
667         // Run
668         // Run
669         // Run
670         // Run
671         // Run
672         // Run
673         // Run
674         // Run
675         // Run
676         // Run
677         // Run
678         // Run
679         // Run
680         // Run
681         // Run
682         // Run
683         // Run
684         // Run
685         // Run
686         // Run
687         // Run
688         // Run
689         // Run
690         // Run
691         // Run
692         // Run
693         // Run
694         // Run
695         // Run
696         // Run
697         // Run
698         // Run
699         // Run
700         // Run
701         // Run
702         // Run
703         // Run
704         // Run
705         // Run
706         // Run
707         // Run
708         // Run
709         // Run
710         // Run
711         // Run
712         // Run
713         // Run
714         // Run
715         // Run
716         // Run
717         // Run
718         // Run
719         // Run
720         // Run
721         // Run
722         // Run
723         // Run
724         // Run
725         // Run
726         // Run
727         // Run
728         // Run
729         // Run
730         // Run
731         // Run
732         // Run
733         // Run
734         // Run
735         // Run
736         // Run
737         // Run
738         // Run
739         // Run
740         // Run
741         // Run
742         // Run
743         // Run
744         // Run
745         // Run
746         // Run
747         // Run
748         // Run
749         // Run
750         // Run
751         // Run
752         // Run
753         // Run
754         // Run
755         // Run
756         // Run
757         // Run
758         // Run
759         // Run
760         // Run
761         // Run
762         // Run
763         // Run
764         // Run
765         // Run
766         // Run
767         // Run
768         // Run
769         // Run
770         // Run
771         // Run
772         // Run
773         // Run
774         // Run
775         // Run
776         // Run
777         // Run
778         // Run
779         // Run
780         // Run
781         // Run
782         // Run
783         // Run
784         // Run
785         // Run
786         // Run
787         // Run
788         // Run
789         // Run
790         // Run
791         // Run
792         // Run
793         // Run
794         // Run
795         // Run
796         // Run
797         // Run
798         // Run
799         // Run
800         // Run
801         // Run
802         // Run
803         // Run
804         // Run
805         // Run
806         // Run
807         // Run
808         // Run
809         // Run
810         // Run
811         // Run
812         // Run
813         // Run
814         // Run
815         // Run
816         // Run
817         // Run
818         // Run
819         // Run
820         // Run
821         // Run
822         // Run
823         // Run
824         // Run
825         // Run
826         // Run
827         // Run
828         // Run
829         // Run
830         // Run
831         // Run
832         // Run
833         // Run
834         // Run
835         // Run
836         // Run
837         // Run
838         // Run
839         // Run
840         // Run
841         // Run
842         // Run
843         // Run
844         // Run
845         // Run
846         // Run
847         // Run
848         // Run
849         // Run
850         // Run
851         // Run
852         // Run
853         // Run
854         // Run
855         // Run
856         // Run
857         // Run
858         // Run
859         // Run
860         // Run
861         // Run
862         // Run
863         // Run
864         // Run
865         // Run
866         // Run
867         // Run
868         // Run
869         // Run
870         // Run
871         // Run
872         // Run
873         // Run
874         // Run
875         // Run
876         // Run
877         // Run
878         // Run
879         // Run
880         // Run
881         // Run
882         // Run
883         // Run
884         // Run
885         // Run
886         // Run
887         // Run
888         // Run
889         // Run
890         // Run
891         // Run
892         // Run
893         // Run
894         // Run
895         // Run
896         // Run
897         // Run
898         // Run
899         // Run
900         // Run
901         // Run
902         // Run
903         // Run
904         // Run
905         // Run
906         // Run
907         // Run
908         // Run
909         // Run
910         // Run
911         // Run
912         // Run
913         // Run
914         // Run
915         // Run
916         // Run
917         // Run
918         // Run
919         // Run
920         // Run
921         // Run
922         // Run
923         // Run
924         // Run
925         // Run
926         // Run
927         // Run
928         // Run
929         // Run
930         // Run
931         // Run
932         // Run
933         // Run
934         // Run
935         // Run
936         // Run
937         // Run
938         // Run
939         // Run
940         // Run
941         // Run
942         // Run
943         // Run
944         // Run
945         // Run
946         // Run
947         // Run
948         // Run
949         // Run
950         // Run
951         // Run
952         // Run
953         // Run
954         // Run
955         // Run
956         // Run
957         // Run
958         // Run
959         // Run
960         // Run
961         // Run
962         // Run
963         // Run
964         // Run
965         // Run
966         // Run
967         // Run
968         // Run
969         // Run
970         // Run
971         // Run
972         // Run
973         // Run
974         // Run
975         // Run
976         // Run
977         // Run
978         // Run
979         // Run
980         // Run
981         // Run
982         // Run
983         // Run
984         // Run
985         // Run
986         // Run
987         // Run
988         // Run
989         // Run
990         // Run
991         // Run
992         // Run
993         // Run
994         // Run
995         // Run
996         // Run
997         // Run
998         // Run
999         // Run
1000        // Run
1001        // Run
1002        // Run
1003        // Run
1004        // Run
1005        // Run
1006        // Run
1007        // Run
1008        // Run
1009        // Run
1010        // Run
1011        // Run
1012        // Run
1013        // Run
1014        // Run
1015        // Run
1016        // Run
1017        // Run
1018        // Run
1019        // Run
1020        // Run
1021        // Run
1022        // Run
1023        // Run
1024        // Run
1025        // Run
1026        // Run
1027        // Run
1028        // Run
1029        // Run
1030        // Run
1031        // Run
1032        // Run
1033        // Run
1034        // Run
1035        // Run
1036        // Run
1037        // Run
1038        // Run
1039        // Run
1040        // Run
1041        // Run
1042        // Run
1043        // Run
1044        // Run
1045        // Run
1046        // Run
1047        // Run
1048        // Run
1049        // Run
1050        // Run
1051        // Run
1052        // Run
1053        // Run
1054        // Run
1055        // Run
1056        // Run
1057        // Run
1058        // Run
1059        // Run
1060        // Run
1061        // Run
1062        // Run
1063        // Run
1064        // Run
1065        // Run
1066        // Run
1067        // Run
1068        // Run
1069        // Run
1070        // Run
1071        // Run
1072        // Run
1073        // Run
1074        // Run
1075        // Run
1076        // Run
1077        // Run
1078        // Run
1079        // Run
1080        // Run
1081        // Run
1082        // Run
1083        // Run
1084        // Run
1085        // Run
1086        // Run
1087        // Run
1088        // Run
1089        // Run
1090        // Run
1091        // Run
1092        // Run
1093        // Run
1094        // Run
1095        // Run
1096        // Run
1097        // Run
1098        // Run
1099        // Run
1100        // Run
1101        // Run
1102        // Run
1103        // Run
1104        // Run
1105        // Run
1106        // Run
1107        // Run
1108        // Run
1109        // Run
1110        // Run
1111        // Run
1112        // Run
1113        // Run
1114        // Run
1115        // Run
1116        // Run
1117        // Run
1118        // Run
1119        // Run
1120        // Run
1121        // Run
1122        // Run
1123        // Run
1124        // Run
1125        // Run
1126        // Run
1127        // Run
1128        // Run
1129        // Run
1130        // Run
1131        // Run
1132        // Run
1133        // Run
1134        // Run
1135        // Run
1136        // Run
1137        // Run
1138        // Run
1139        // Run
1140        // Run
1141        // Run
1142        // Run
1143        // Run
1144        // Run
1145        // Run
1146        // Run
1147        // Run
1148        // Run
1149        // Run
1150        // Run
1151        // Run
1152        // Run
1153        // Run
1154        // Run
1155        // Run
1156        // Run
1157        // Run
1158        // Run
1159        // Run
1160        // Run
1161        // Run
1162        // Run
1163        // Run
1164        // Run
1165        // Run
1166        // Run
1167        // Run
1168        // Run
1169        // Run
1170        // Run
1171        // Run
1172        // Run
1173        // Run
1174        // Run
1175        // Run
1176        // Run
1177        // Run
1178        // Run
1179        // Run
1180        // Run
1181        // Run
1182        // Run
1183        // Run
1184        // Run
1185        // Run
1186        // Run
1187        // Run
1188        // Run
1189        // Run
1190        // Run
1191        // Run
1192        // Run
1193        // Run
1194        // Run
1195        // Run
1196        // Run
1197        // Run
1198        // Run
1199        // Run
1200        // Run
1201        // Run
1202        // Run
1203        // Run
1204        // Run
1205        // Run
1206        // Run
1207        // Run
1208        // Run
1209        // Run
1210        // Run
1211        // Run
1212        // Run
1213        // Run
1214        // Run
1215        // Run
1216        // Run
1217        // Run
1218        // Run
1219        // Run
1220        // Run
1221        // Run
1222        // Run
1223        // Run
1224        // Run
1225        // Run
1226        // Run
1227        // Run
1228        // Run
1229        // Run
1230        // Run
1231        // Run
1232        // Run
1233        // Run
1234        // Run
1235        // Run
1236        // Run
1237        // Run
1238        // Run
1239        // Run
1240        // Run
1241        // Run
1242        // Run
1243        // Run
1244        // Run
1245        // Run
1246        // Run
1247        // Run
1248        // Run
1249        // Run
1250        // Run
1251        // Run
1252        // Run
1253        // Run
1254        // Run
1255        // Run
1256        // Run
1257        // Run
1258        // Run
1259        // Run
1260        // Run
1261        // Run
1262        // Run
1263        // Run
1264        // Run
1265        // Run
1266        // Run
1267        // Run
1268        // Run
1269        // Run
1270        // Run
1271        // Run
1272        // Run
1273        // Run
1274        // Run
1275        // Run
1276        // Run
1277        // Run
1278        // Run
1279        // Run
1280        // Run
1281        // Run
1282        // Run
1283        // Run
1284        // Run
1285        // Run
1286        // Run
1287        // Run
1288        // Run
1289        // Run
1290        // Run
1291        // Run
1292        // Run
1293        // Run
1294        // Run
1295        // Run
1296        // Run
1297        // Run
1298        // Run
1299        // Run
1300        // Run
1301        // Run
1302        // Run
1303        // Run
1304        // Run
1305        // Run
1306        // Run
1307        // Run
1308        // Run
1309        // Run
1310        // Run
1311        // Run
1312        // Run
1313        // Run
1314        // Run
1315        // Run
1316        // Run
1317        // Run
1318        // Run
1319        // Run
1320        // Run
1321        // Run
1322        // Run
1323        // Run
1324        // Run
1325        // Run
1326        // Run
1327        // Run
1328        // Run
1329        // Run
1330        // Run
1331        // Run
1332        // Run
1333        // Run
1334        // Run
1335        // Run
1336        // Run
1337        // Run
1338        // Run
1339        // Run
1340        // Run
1341        // Run
1342        // Run
1343        // Run
1344        // Run
1345        // Run
1346        // Run
1347        // Run
1348        // Run
1349        // Run
1350        // Run
1351        // Run
1352        // Run
1353        // Run
1354        // Run
1355        // Run
1356        // Run
1357        // Run
1358        // Run
1359        // Run
1360        // Run
1361        // Run
1362        // Run
1363        // Run
1364        // Run
1365        // Run
1366        // Run
1367        // Run
1368        // Run
1369        // Run
1370        // Run
1371        // Run
1372        // Run
1373        // Run
1374        // Run
1375        // Run
1376        // Run
1377        // Run
1378        // Run
1379        // Run
1380        // Run
1381        // Run
1382        // Run
1383        // Run
1384        // Run
1385        // Run
1386        // Run
1387        // Run
1388        // Run
1389        // Run
1390        // Run
1391        // Run
1392        // Run
1393        // Run
1394        // Run
1395        // Run
1396        // Run
1397        // Run
1398        // Run
1399        // Run
1400        // Run
1401        // Run
1402        // Run
1403        // Run
1404        // Run
1405        // Run
1406        // Run
1407        // Run
1408        // Run
1409        // Run
1410        // Run
1411        // Run
1412        // Run
1413        // Run
1414        // Run
1415        // Run
1416        // Run
1417        // Run
1418        // Run
1419        // Run
1420        // Run
1421        // Run
1422        // Run
1423        // Run
1424        // Run
1425        // Run
1426        // Run
1427        // Run
1428        // Run
1429        // Run
1430        // Run
1431        // Run
1432        // Run
1433        // Run
1434        // Run
1435        // Run
1436        // Run
1437        // Run
1438        // Run
1439        // Run
1440        // Run
1441        // Run
1442        // Run
1443        // Run
1444        // Run
1445        // Run
1446        // Run
1447        // Run
1448        // Run
1449        // Run
1450        // Run
1451        // Run
1452        // Run
1453        // Run
1454        // Run
1455        // Run
1456        // Run
1457        // Run
1458        // Run
1459        // Run
1460        // Run
1461        // Run
1462        // Run
1463        // Run
1464        // Run
1465        // Run
1466        // Run
1467        // Run
1468        // Run
1469        // Run
1470        // Run
1471        // Run
1472        // Run
1473        // Run
1474        // Run
1475        // Run
1476        // Run
1477        // Run
1478        // Run
1479        // Run
1480        // Run
1481        // Run
1482        // Run
1483        // Run
1484        // Run
1485        // Run
1486        // Run
1487        // Run
1488        // Run
1489        // Run
1490        // Run
1491        // Run
1492        // Run
1493        // Run
1494        // Run
1495        // Run
1496        // Run
1497        // Run
1498        // Run
1499        // Run
1500        // Run
1501        // Run
1502        // Run
1503        // Run
1504        // Run
1505        // Run
1506        // Run
1507        // Run
1508        // Run
1509        // Run
1510        // Run
1511        // Run
1512        // Run
1513        // Run
1514        // Run
1515        // Run
1516        // Run
1517        // Run
1518        // Run
1519        // Run
1520        // Run
1521        // Run
1522        // Run
1523        // Run
1524        // Run
1525        // Run
1526        // Run
1527        // Run
1528        // Run
1529        // Run
1530        // Run
1531        // Run
1532        // Run
1533        // Run
1534        // Run
1535        // Run
1536        // Run
1537        // Run
1538        // Run
1539        // Run
1540        // Run
1541        // Run
1542        // Run
1543        // Run
1544        // Run
1545        // Run
1546        // Run
1547        // Run
1548        // Run
1549        // Run
1550        // Run
1551        // Run
1552        // Run
1553        // Run
1554        // Run
1555        // Run
1556        // Run
1557        // Run
1558        // Run
1559        // Run
1560        // Run
1561        // Run
1562        // Run
1563        // Run
1564        // Run
1565        // Run
1566        // Run
1567        // Run
1568        // Run
1569        // Run
1570        // Run
1571        // Run
1572        // Run
1573        // Run
1574        // Run
1575        // Run
1576        // Run
1577        // Run
1578        // Run
1579        // Run
1580        // Run
1581        // Run
1582        // Run
1583        // Run
1584        // Run
1585        // Run
1586        // Run
1587        // Run
1588        // Run
1589        // Run
1590        // Run
1591        // Run
1592        // Run
1593        // Run
1594        // Run
1595        // Run
1596        // Run
1597        // Run
1598        // Run
1599        // Run
1600        // Run
1601        // Run
1602        // Run
1603        // Run
1604        // Run
1605        // Run
1606        // Run
1607        // Run
1608        // Run
1609        // Run
1610        // Run
1611        // Run
1612        // Run
1613        // Run
1614        // Run
1615        // Run
1616        // Run
1617        // Run
1618        // Run
1619
```

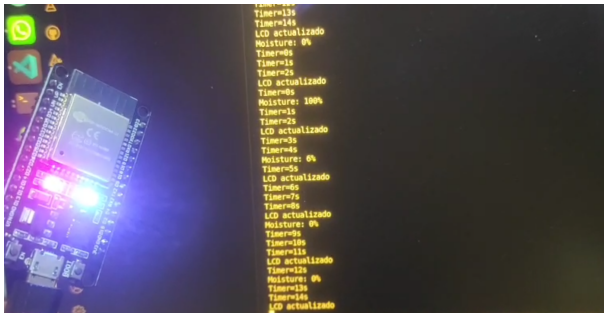



Figura 4. Demostración de funcionamiento y visualización en consola

IV-C. Memoria utilizada

En la Figura 5 se muestra un resumen del uso de memoria generado automáticamente por ESP-IDF durante la compilación del proyecto:

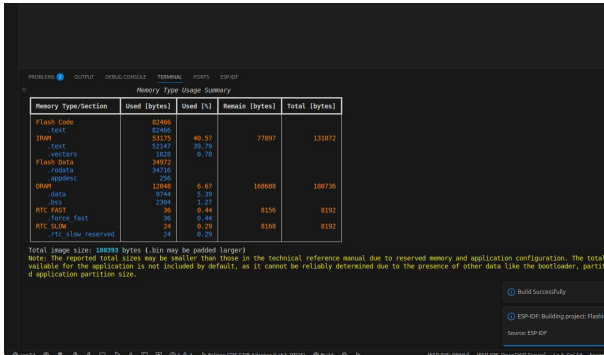


Figura 5. Sumario de uso de memoria

El sistema presenta una utilización moderada de IRAM (40.57) y mínima en DRAM y RTC, lo cual indica que la implementación es eficiente y deja amplio margen para futuras expansiones. El siguiente enlace lleva a un repositorio en GitHub con los archivos multimedia y código propio del proyecto. https://github.com/PabloKevin/Monitor_de_Humedad_y_Riego_Automatico/tree/main

REFERENCIAS

[1] Quiroga, D. (2025). Sistemas embebidos (Séptimo semestre IMEC, 2025–1S). Universidad Tecnológica del Uruguay, Instituto Tecnológico Regional Suroeste.