

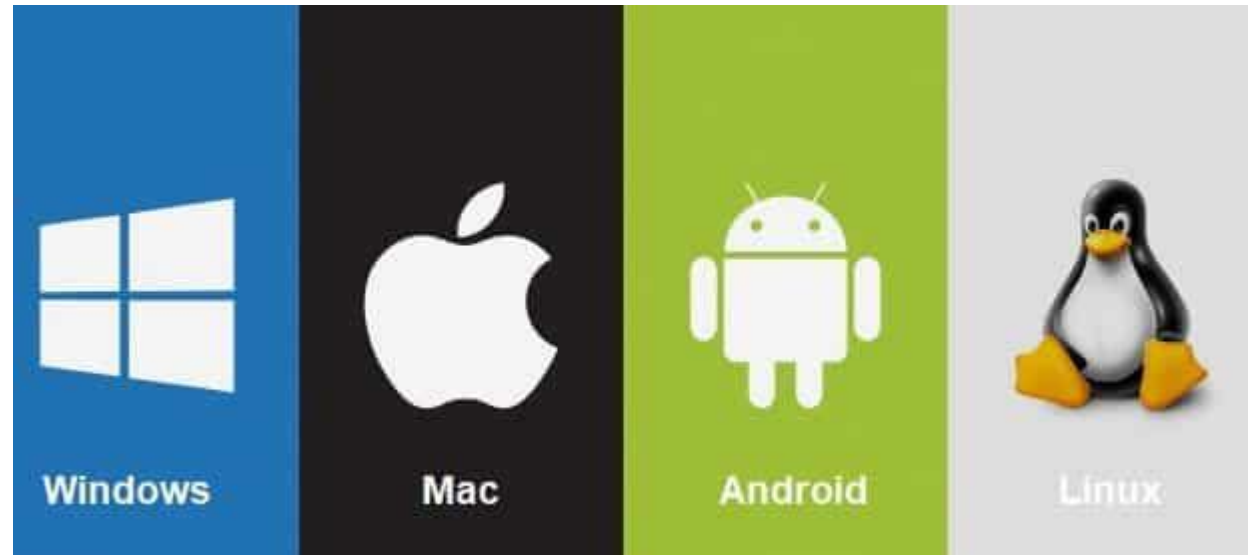
SISTEMAS OPERATIVOS

Un sistema operativo (SO) es el software fundamental que gestiona el hardware de una computadora y proporciona servicios básicos a otras aplicaciones y programas. Actúa como un intermediario entre el hardware y el software, permitiendo que diferentes programas se ejecuten y utilicen los recursos de la computadora de manera eficiente.



SISTEMAS OPERATIVOS

Un sistema operativo (SO) es el software fundamental que gestiona el hardware de una computadora y proporciona servicios básicos a otras aplicaciones y programas. Actúa como un intermediario entre el hardware y el software, permitiendo que diferentes programas se ejecuten y utilicen los recursos de la computadora de manera eficiente.



Características de RTOS

Gestión de procesos

Gestión de memoria

Gestión de archivos

Gestión de dispositivos

Interfaz de usuario

Seguridad y acceso

SISTEMAS OPERATIVOS EN MCU

Conocidos como **sistemas operativos en tiempo real** (RTOS, por sus siglas en inglés), son una categoría especial de sistemas operativos diseñados para gestionar los recursos de microcontroladores en aplicaciones embebidas

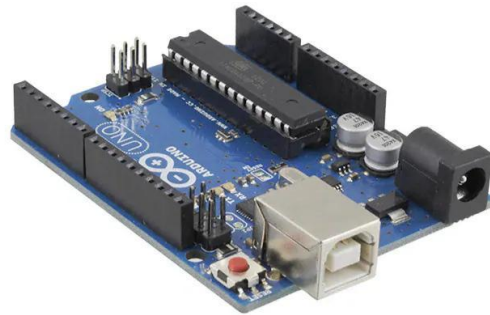


Best RTOS for Embedded System



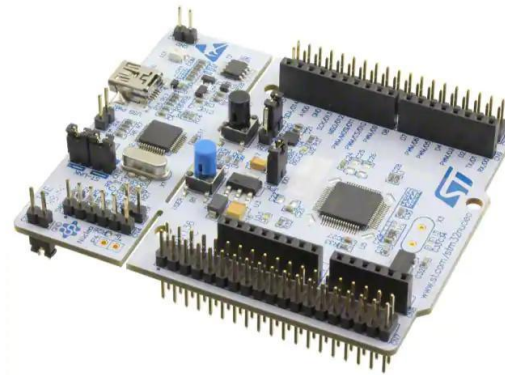
SISTEMAS OPERATIVOS EN MCU

A diferencia de los sistemas operativos para computadoras personales o servidores, los RTOS están optimizados para entornos con recursos limitados y donde el cumplimiento de tiempos específicos es crucial.



ATmega 328p

- 16 MHz
- 32 kB flash
- 2 kB RAM



STM32L476RG

- 80 MHz
- 1 MB flash
- 128 kB RAM



ESP-WROOM-32

- 240 MHz (dual core)
- 4 MB flash
- 520 kB RAM

Super Loop



RTOS

Características

Tiempo Real

Uso Eficiente de Recursos

Gestión de Tareas

Multitarea

Determinismo

Interrupciones

SISTEMAS OPERATIVOS EN MCU



Best RTOS for Embedded System



SUPER LOOP Y RTOS

Typical Programming

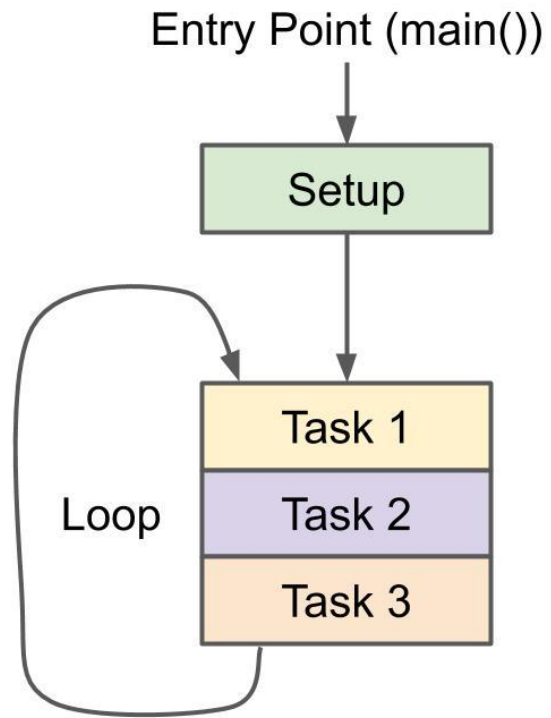


Bare Metal Programming

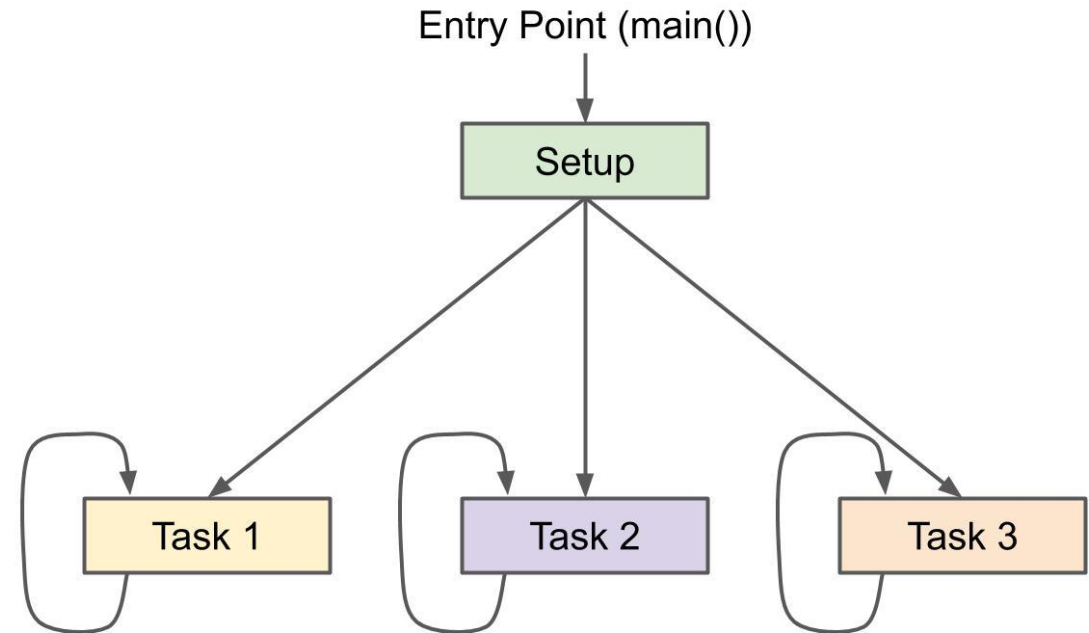


SUPER LOOP Y RTOS

Super Loop



RTOS



Funciones principales

vTaskDelay : Retrasar una tarea durante un número determinado de ticks.

```
void vTaskDelay( const TickType_t xTicksToDelay );
```

Parámetros:

- **xTicksToDelay** : Es la cantidad de tiempo, en períodos de tic, que la tarea debe bloquearse.

Un *tick* es una unidad básica de tiempo en un sistema operativo en tiempo real (RTOS). Representa un intervalo de tiempo fijo que el sistema utiliza para medir y controlar el tiempo, especialmente en la planificación de tareas. En un RTOS, el *tick* se usa para programar cuándo deben ejecutarse las tareas. Por ejemplo, cuando una tarea utiliza una función como `vTaskDelay()` para esperar un número específico de *ticks*, el RTOS bloquea la tarea hasta que hayan transcurrido esos *ticks*, permitiendo que otras tareas se ejecuten mientras tanto. Un tick es generado por un temporizador de hardware

Funciones principales

xTaskCreate : Crea una nueva tarea y la agrega a la lista de tareas que están listas para ejecutarse.

```
static inline IRAM_ATTR BaseType_t xTaskCreate(  
    TaskFunction_t pvTaskCode,  
    const char * const pcName,      /*lint !e971 Unqualified char types are allowed for strings and single cl  
    const uint32_t usStackDepth,  
    void * const pvParameters,  
    UBaseType_t uxPriority,  
    TaskHandle_t * const pxCreatedTask) PRIVILEGED_FUNCTION  
{  
    return xTaskCreatePinnedToCore( pvTaskCode, pcName, usStackDepth, pvParameters, uxPriority, pxCreatedTask, tskNO_AFFINITY )  
}  
/*
```

Parámetros:

- **pvTaskCode**: Puntero a la función de entrada de la tarea (solo el nombre de la función).
- **pcName** : Un nombre descriptivo para la tarea. Se utiliza principalmente para facilitar la depuración, pero también se puede utilizar para obtener un identificador de tarea.
- **uxStackDepth**: La cantidad de palabras (;no bytes!) que se asignarán para usar como pila de la tarea.
- **pvParameters**: Un valor que se pasa como argumento a la tarea creada.
- **uxPriority**: La prioridad con la que se ejecutará la tarea creada.
- **pxCreatedTask**: Se utiliza para pasar un identificador a la tarea creada fuera de la función xTaskCreate(). pxCreatedTask es opcional y se puede establecer en NULL.