Uso de RIOT-OS para programación en el ámbito de IoT

Imagen que contiene edificio, cielo

Descripción generada automáticamente

María Ariza Gamero

Alejandro Hernán Luque

Alberto Vázquez Baeza

Infraestructura Avanzada de

Redes de Sensores

Máster en Sistemas Inteligentes en

Energía y Transporte

Contenido

[1. Presentación 2](#_Toc11256964)

[1.1. Mazos que no se usan en el conjunto total de pruebas 2](#_Toc11256965)

[2. RIOT-OS 2](#_Toc11256966)

[3. Git 2](#_Toc11256967)

[4. Docker 2](#_Toc11256968)

[5. Compilar 3](#_Toc11256969)

[6. Flashear 3](#_Toc11256970)

[7. Programa leds 3](#_Toc11256971)

[8. Comunicación 3](#_Toc11256972)

# Presentación

Se desea instalar el sistema operativo RIOT-OS en las placas de Texas Instruments cc2538dk y cc2650stk, conocida como Sensor Tag. Para ello, será necesito el uso de Docker. Una vez ejecutado RIOT-OS, se procederá a comprobar que funciona correctamente empleando un sencillo programa de encendido de LEDs, que primero se compilará desde RIOT-OS y después de flasheará una herramienta nativa propietaria y necesaria de Texas Instrument, Uniflash. Otro flaseador propio de esta misma marca es Flash Programmer, que también se usará. Por último, se procederá a realizar comunicación usando el protocolo 802.15.4.

## Mazos que no se usan en el conjunto total de pruebas

# RIOT-OS

RIOT alimenta el Internet de las Cosas como Linux alimenta el Internet. RIOT-OS es un sistema operativo open source y gratuito desarrollado por una comunidad que reúne compañías, instituciones académicas y aficionados distribuidos por todo el mundo.

RIOT es compatible con la mayoría de los dispositivos IoT de bajo consumo y las arquitecturas de microcontroladores. RIOT tiene como objetivo implementar todos los estándares abiertos relevantes que respaldan un Internet de las Cosas conectado, seguro, duradero y amigable con la privacidad.

La semilla de RIOT fue FeuerWare, un sistema operativo que nace en 2008 para redes de sensores inalámbricos. Surge como parte del proyecto de FeuerWare donde se buscaba la monitorización de los bomberos. Para aumentar la modularidad e incluir nuevos protocolos IETF, desde el repositorio original de FeuerWare se bifurcó µkleos. El soporte para 6LoWPAN, RPL y TCP se integró en los años siguientes. En 2013 RIOT se hace público siendo el sucesor directo de µkleos, promoviendo RIOT a una comunidad mucho más grande.

Actualmente, RIOT se ejecuta en varias plataformas, incluidos dispositivos integrados y PC comunes. El código dependiente del hardware se reduce al mínimo y se abstrae del propio kernel. Esta solución diseñada pensando en IoT posee una arquitectura de microkernel y emplea estándares como 6LoWPAN que facilita su implementación.

# Git

# Guía de uso de Git

### Instalar Git

## Conectarse a este repositorio

### Iniciar el repositorio de git

En la carpeta en la que se desea tener el proyecto se debe realizar un git init. El git init se realiza desde el terminal si se encuentra en linux o en el bash de git si se encuentra en windows

git init

Una vez se ejecuta este comando se crea una carpeta .git en la carpeta de trabajo.

### Conectarse al repositorio

Una vez iniciado el git, se ejecuta la siguiente sentencia para conectarse a este repositorio. Al final de la url tiene que aparecer siempre .git

git remote add origin https://github.com/blalebla/IARS.git

Se suele utilizar origin como etiqueta de carpeta de origen. Se ha conectado con el repositorio pero no se ha descargado nada todavía.

A continuación se ejecuta lo siguiente:

git fetch origin

fetch hace una comprobación del repositorio online, enumero los objetos y las ramas presenters, en este caso solo se tiene la rama master.

### Descarga archivos del repositorio

Para descargar los archivos del repositorio se realiza un pull:

git pull origin master

se ha realizado un pull de la rama master del repositorio online

### Una vez realizado lo anterior ya se tiene acceso de forma local a

los archivos

## Realizar cambios en git

Una vez se ha descargado el repositorio en el que se desea trabajar, ya se pueden realizar modificaciones, crear archivos nuevos, eliminar algunos exitentes, crear carpetas...

### Estructura de Git

Las versiones en git se distribuyen en 3 espacios:

* Working Directory: es el directorio donde se trabaja, todo lo que hay aquí se puede perder.
* Staging: es la zona de preparado
* Commit: la zona donde se guardan directamente

### Pasar de WD al área de Staging

Para pasar del working directory al área de Staging se hace uso del siguiente comando:

gitt add .

El punto lo que hace que agrega al area de staging todos los archivos que se han creado o modificado. También se pueden añadir los archivos uno a uno, o declarandolos de forma consecutiva tras el add.

git add <archivo1.md> <archivo2.bin>

### Commit

Para realizar un commit se realiza la siguiente operación:

git commit -m "comentario del comit"

## Realizar subida de los cambios realizados

Para subir los cambios realizados se debe realizar un push de la siguiente manera:

git push origin master

Una vez ejecutado este comando suele pedir las credenciales de github

# Docker

# Compilar

# Flashear

# Programa leds

# Comunicación