UNIVERSIDAD
D
CÓRDOBA

INSTITUTO DE ESTUDIOS DE POSTGRADO

MÁSTER EN ENERGÍAS RENOVABLES DISTRIBUIDAS

SISTEMA SCADA BASADO EN RASPBERRY PI PARA LA MONITORIZACIÓN DE UNA INSTALACIÓN FOTOVOLTAICA

LUIS SEGOVIA GUERRERO

Trabajo para optar al

Máster en Energías Renovables Distribuidas

Profesor Supervisor:

MIGUEL GONZÁLEZ REDONDO

Córdoba, septiembre 2021

DEDICATORIA

A mi Familia y amigos, que me apoyaron enormemente durante la realización del Máster y la ejecución de este proyecto.

AGRADECIMIENTOS

Doy gracias, de forma sincera, por la aceptación y buena acogida que tuve al solicitar este proyecto a mi actual tutor, Miguel. Así como también la buena guía y el gran apoyo que me ha brindado durante la realización de todo el proyecto.

Se agradece de igual manera la flexibilidad que me ha otorgado para realizar el proyecto de forma autónoma y remota en mi ciudad, debido a la complicada situación que hemos seguido viviendo este año. Además, agradecer la siempre disponibilidad y respuesta inmediata a cualquier consulta realizada.

Útil cita dada por mi tutor:

> *Un proyecto no solo es exponer lo desarrollado, es también plasmar el proceso de desarrollo y justificar las elecciones y soluciones*.

INDICE GENERAL

Pág.

INDICE DE TABLAS

INDICE DE FIGURAS

RESUMEN

El presente proyecto se desarrolla en torno a dos temas principales. El primero consiste en el diseño, construcción e instalación de un sistema fotovoltaico de baja potencia. En este se incluyen todos los elementos básicos de una instalación solar fotovoltaica autónoma (panel, controlador de tipo PWM y batería) así como la sensórica y microcontroladores necesarios para la monitorización de este.

El segundo tema de trabajo se basa en el diseño de un sistema de monitorización tipo SCADA. Este se diseña en el entorno Node-RED para ser ejecutado en un dispositivo altamente portátil (Raspberry Pi 4B) y que recibe los datos de un microcontrolador (ESP32) a través del protocolo de comunicación MQTT, pasando por un servidor central denominado bróker (Mosquitto Broker). La característica principal que hace atractivo este proyecto es el bajo coste del sistema de monitorización diseñado. Ideal para aplicaciones en las que no es justificable el coste prohibitivo de ciertos sistemas de monitorización comerciales.

Palabras Clave: Sistema, Fotovoltaico, ESP32, SCADA, Node-RED, Raspberry Pi, FZ0430, ACS712, DHT22, BH1750, Arduino, Monitorización, PWM, DIY, Bajo Coste, Eclipse Mosquitto Broker, MQTT.

ABSTRACT

This project is developed around two main themes. The first consists of the design, construction, and installation of a low power photovoltaic system. It includes all the basic elements of an autonomous photovoltaic solar installation (PV panel, PWM controller, and battery) as well as all the necessary sensors and microcontrollers for monitorization purposes.

The second main topic is based on the design of a SCADA type monitoring system. It is designed around the Node-RED environment to be executed on a highly portable device (Raspberry Pi 4B) and that receives data from a microcontroller (ESP32) through the MQTT communication protocol, passing through a central server called broker (Mosquitto Broker). The main characteristic that makes this project attractive is the low cost of the designed monitoring system. Ideal for applications in which the prohibited cost of certain commercial monitoring systems is not justifiable.

Keywords: PV, system, ESP32, SCADA, Node-RED, Raspberry Pi, FZ0430, ACS712, DHT22, BH1750, Arduino, Monitorization, PWM, DIY, Low Cost, Eclipse Mosquitto Broker, MQTT.

1. INTRODUCCIÓN Y ESTADO DEL ARTE

El control y recopilación de los datos en el mundo ingenieril es algo esencial. Cobra más importancia cuanto más sofisticados y avanzados se hacen los diferentes procesos a controlar. El estudio de los datos obtenidos es de suma importancia para la verificación de que los sistemas están funcionando correctamente o, por otro lado, para detectar y corregir las diversas fallas que puedan surgir.

En el mundo industrial existen numerosas soluciones para este control de los datos, pero a veces, a precios bastante elevados. Dichos costes se han de justificar en función de la envergadura de la instalación a controlar (y su producción y beneficios asociados). Para instalaciones pequeñas o domésticas, estos equipos no podrán ser instalados, por ello, las soluciones de bajo coste son las ideales para estos escenarios (Martínez Cacho & Arizaleta Arteaga, 2018).

En el contexto de las energías renovables, el campo de monitorización y control se hace cada vez más necesario debido al gran crecimiento de éstas en los últimos años. Un punto de gran ayuda en este tipo de instalaciones es la monitorización inalámbrica, de fácil instalación y escalable (Abajo González, et al., 2018).

1.1 SCADA

SCADA es un acrónimo de 'Supervisory Control And Data Acquisition', esto es, un sistema de supervisión, control y adquisición de datos. Un sistema de estas características comprende tanto la parte de hardware como la de software que permiten la supervisión y control de las plantas y sistemas industriales. Con este sistema se pretende integrar el estudio, recogida y procesado de los datos, tanto en tiempo real como el almacenamiento de estos.

Antes de la existencia de SCADA los procesos eran monitoreados y controlados por el personal de las instalaciones de forma manual. Se colocaban instrumentos en el emplazamiento de los sistemas de energías renovables y eran monitoreados por operarios 'in-situ'. Dichos operarios tomaban las medidas de los aparatos allí localizados o tomaban valores con instrumentos portátiles. Los valores eran comunicados a los responsables de la operación para valorar las diferentes medidas que se necesitaran tomar (Dumitru & Gligor, s.f.). Conforme los procesos industriales fueron avanzando, se comenzaron a usar pequeñas formas de automatización como el uso de temporizadores y relés. Todo esto conllevó la aparición del primer sistema SCADA en los 70 bajo el nombre de 'Monolith SCADA' (sistemas locales). Este sistema se crea en un entorno en el que aún no existen las redes. Involucra sistemas completamente independientes. Se utilizaban dos sistemas principales conectados únicamente con unidades terminales remotas (RTUs[1]) a través de una conexión WAN.

Con el avance de los sistemas de comunicación y redes en los 80 y 90 apareció lo que se denomina 'Distributed SCADA' (conexiones tipo LAN). Con esta nueva tecnología se consiguió un abaratamiento del sistema de monitorización además de una reducción del tamaño. La distribución de las tareas y procesos condujo a un aumento de la potencia de

[1] RTU: Unidad Terminal Remota: dispositivo basado en microprocesadores. Permite obtener señales independientes de los procesos y enviar la información a un sitio remoto donde se procese (Wikipedia - La Enciclopedia Libre, 2020).

procesado, de la fiabilidad y la redundancia del sistema. La desventaja de este sistema es que no tenía ninguna capacidad fuera de la red LAN establecida. Además, dichos protocolos LAN solían ser propietarios.

Finalmente, en los 2000, la aparición de arquitecturas de sistema abierto para los diferentes protocolos de comunicación, accesible para todos los fabricantes llevó a la aparición del 'Networked SCADA' y posteriormente el 'IoT SCADA', mejorando considerablemente el control y monitoreo de procesos. El primero involucra un sistema de arquitectura abierta con múltiples sistemas en red que se comunican a través de redes WAN, compartiendo las funciones de la estación principal ('Master SCADA') y utilizando PLCs[2] para el monitoreo. Esta generación de SCADA era muy similar a la segunda generación, pero con una diferencia principal: se podría conectar a Internet y a periféricos de terceras partes a través del protocolo IP. El segundo (4ª generación) es una combinación de los sistemas SCADA convencionales con la arquitectura de nube ('cloud'). El IoT permite una alternativa a los PLCs convencionales y, además, involucra el modelado y tratamiento de datos mejorando la accesibilidad, flexibilidad, disponibilidad, escalabilidad y el coste de los sistemas (Aghenta & Iqbal, 2019) (Oriaghe Aghenta, 2020).

El siguiente esquema recoge toda esta evolución (Figura 1).

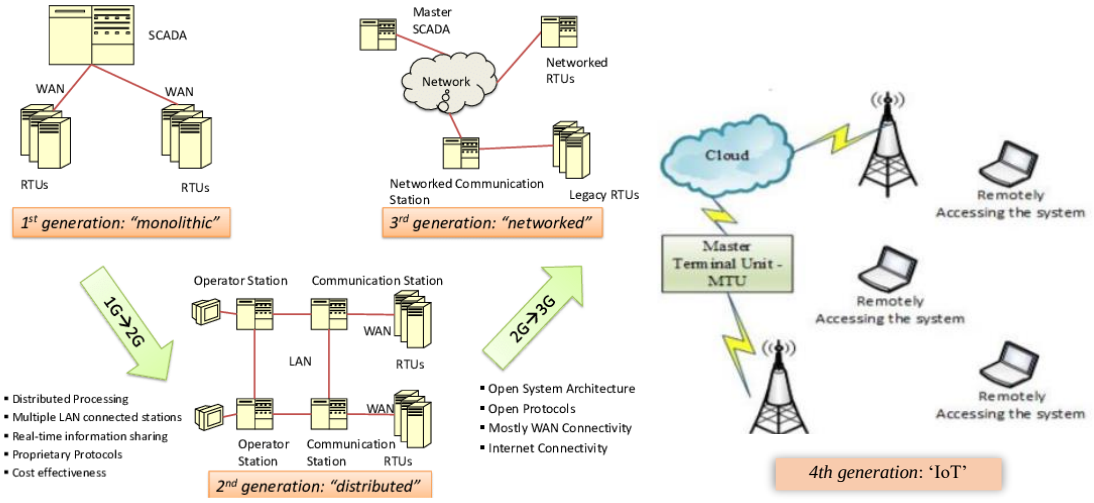


Figura 1. Evolución de los sistemas SCADA. Generaciones 1 – 3 (izquierda) y 4ª (derecha). Imágenes tomadas de: *(Karnouskos & Walter Colombo, 2011)*, *(Nechibvute & Mudzingwa, 2013)* y *(Oriaghe Aghenta, 2020)*

1.1.1 Partes de un sistema SCADA

- Instrumentación de campo (FIDs): incluyen sensores, actuadores, transmisores, etc. Es todo aquello que permita la medida de distintas magnitudes y que están

[2] PLC: Controlador Lógico Programable o Autómata Programable. Es un sistema (computadora) utilizado para monitorear procesos electromecánicos, electroneumáticos, y electrohidráulicos en la ingeniería automática o automatización industrial (Wikipeda - La Enciclopedia Libre, 2021).

conectados y tienen relación directa con los procesos que quieren ser monitoreados.

- Unidades terminales remotas (RTUs): son unidades computarizadas (microprocesadores). Similar a los PLCs. Se prefieren estos últimos por su configuración, flexibilidad, asequibilidad y versatilidad. Se encuentran en el lugar específico donde se desea controlar un proceso. Contribuyen a recoger la información proporcionada por la instrumentación de campo y enviarla a la estación principal de monitoreo y control.
- Unidad principal terminal (MTU) o hosts SCADA: son consolas principales o servidores que funcionan a modo de procesador central para todo el sistema SCADA. Proporciona los comandos necesarios, reúne la información y la almacena, se comunica con los todos los sistemas y proporciona una interfaz humano-máquina (HMI)
- Redes de comunicación: establecen la vía de conexión entre los diversos sistemas, esto es, conectan y transmiten la información de las RTUs/PLCs al SCADA. La plataforma principal SCADA se suele emplazar remotamente y se conecta con la sensórica a través de diversos protocolos como pueden ser el TCP/IP, Ethernet, WiFi, Fieldbus, Modbus, DNP, etc. (Oriaghe Aghenta, 2020).

En la Figura 2 se presenta un esquema de los distintos componentes de un sistema SCADA.

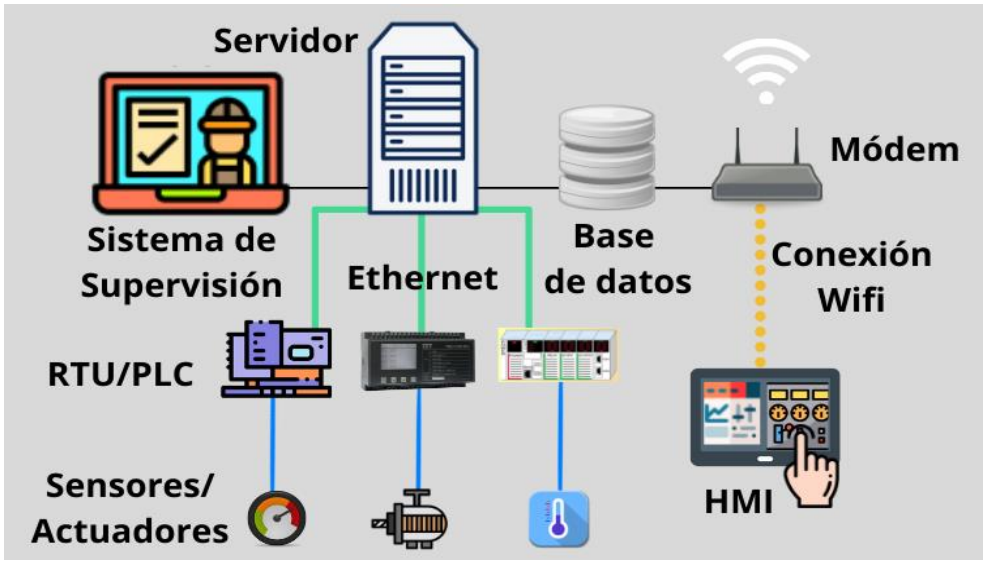


Figura 2. Componentes de un sistema SCADA *(Sicma21, 2021)*.

Además, las características deseables de un sistema SCADA son las siguientes: dinamismo (facilidad de trabajo con nodos), 'retro-instalación' (facilidad de incorporación nuevas tecnologías al sistema existente), facilidad de instalación y uso, redundancia (para aumentar la fiabilidad), bajo consumo de potencia, fiabilidad y disponibilidad y seguridad.

1.2 IoT

Este término, 'IoT', fue acuñado por Kevin Ashton en 1999, fundador del 'MIT Autoidentification Centre'. Más tarde, en 2005, fue oficialmente presentado por la Unión Internacional de Telecomunicación (ITU) Este término empezó a cobrar importancia en los últimos años, con el desarrollo de nuevas tecnologías.

El IoT comprende una gran variedad de definiciones dadas por diversas organizaciones e investigadores. La más común es la definición dada por la ITU en 2012: u*na infraestructura global para la sociedad de la información, permitiendo servicios avanzados mediante la interconexión (física y virtual) de objetos y sistemas basados en las tecnologías de la información y la comunicación interoperables, existentes y en desarrollo.* Además, Guillemin y Friess sugieren una de las definiciones más simples: *El Internet de las Cosas permite a las personas y las cosas estar conectadas en cualquier momento, en cualquier lugar, con cualquier cosa y cualquier persona, idealmente usando cualquier camino o red y cualquier servicio.* Esto se plasma de excelente manera en la Figura *3*. (Atlam, et al., 2018)

Figura 3. IoT. Red *(Atlam, et al., 2018)*

Se puede considerar una infraestructura interconectada dinámica y global que controla diversos objetos de forma inteligente

Para que un sistema pueda considerarse IoT ha de cumplir las siguientes características (no basta con estar conectado a la red Internet):

- Tener una función útil
- Interoperabilidad dentro de un ecosistema donde intervienen varios sistemas y dispositivos interconectados. Esto es, que el dispositivo no sea exclusivo a una sola persona u organización – eso se considera operación máquina-a-máquina.
- Involucrar la sensorización, control y actuación para una determinada aplicación.
- Recolección local de datos
- Capacidad de operar de forma autónoma
- Debe existir una API[3] para la creación de aplicaciones interactivas (Breck, 2020)

Otras características típicas comunes a los sistemas que involucran IoT son:

- Inteligencia – conexión con 'smart-devices'.
- Entorno dinámico
- Gran cantidad de datos
- Heterogeneidad – involucra diferentes dispositivos, plataformas, sistemas operativos, etc., conectados a través de diversos protocolos.
- Bajo consumo de energía

[3] API: Interfaz de programación de aplicaciones. Es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones (Red Hat, 2021).

- Conectividad
- Autoconfiguración – dispositivos y sistemas con mínima intervención humana una vez han sido desplegados

Tecnologías de comunicación IoT: ZigBee, Bluetooth, Z-Wave, NFC y 6LoWPAN entre otras.

Estas tecnologías tienen una gran cantidad de aplicaciones, muchas de ellas con impacto directo en la vida de las personas. Algunas de estas aplicaciones son: 'Healthcare', Ciudades Inteligentes ('Smart Cities'), Casas Inteligentes ('Smart Home'), Industria 4.0, Ventas Inteligente ('Smart Retail'), Vehículos ('Connected Cars'), Parkings interconectados, Energía y Red Inteligente ('Smart Energy and Smart Grid'), monitoreo medioambiental, Agricultura Inteligente ('Smart Agriculture'), 'Wearables'[4].

En definitiva, el IoT es la siguiente etapa de la evolución de Internet, cada vez cobra más importancia en todos los aspectos de la vida diaria, tanto a nivel particular como a gran escala (Atlam, et al., 2018).

1.3 Energía solar fotovoltaica

La energía solar tiene un especial atractivo al ser una de las que tienen una mayor disponibilidad de forma directa y, además, 'gratuita'. Viene utilizándose desde tiempos prehistóricos en todas sus diferentes vertientes. Se puede aprovechar principalmente de dos formas, por la vía térmica y por la vía fotovoltaica. La primera hace uso directo de la radiación recibida para aumentar la temperatura, generalmente, de un fluido caloportador. En la segunda lo que se utiliza es un dispositivo fotovoltaico capaz de convertir la radiación solar en corriente continua. El campo de interés para este trabajo, por la parte física desarrollada, es la segunda, aunque el sistema SCADA sería aplicable en cualquiera de ellas.

Aunque existen diferentes tipos de tecnologías fotovoltaicas, la más extendida debido a criterios de fiabilidad y eficiencia son las basadas en obleas de silicio, capturan alrededor de una cuota del 90 % del mercado. No obstante, las tecnologías se clasifican de acuerdo con el siguiente criterio:

- Primera generación de sistemas fotovoltaicos (PV): silicio cristalino, tanto mono- (sc-Si) como policristalino (mc-Si). Están completamente desarrollados para su venta en el mercado.
- Segunda generación: tecnología de paneles de película fina. Se clasifican en tres familias, a saber: silicio amorfo (a-Si) y microamorfo (a-Si/mc-Si); telururo de cadmio (CdTe); selenuro de cobre e indio (CIS) y diselenuro de cobre indio y galio (CIGS). Están introduciéndose en el mercado.
- Tercera generación: concentradores fotovoltaicos (CPV) y células fotovoltaicas orgánicas. Tecnología no comercial, se encuentra bajo desarrollo.

La producción de los paneles solares de c-Si comenzó en 1963, cuando la compañía Sharp de Japón empezó a producir paneles fotovoltaicos comerciales e instalaron un panel de 242 W en un faro – la instalación comercial más grande hasta el momento. A partir de esa fecha, comenzó la carrera por el desarrollo de esta tecnología emergente (Jana, 2013). En la Figura *4* se presenta la evolución de la energía solar fotovoltaica instalada en todo el mundo en el período 2000-2019. Se puede observar un claro gran aumento en los últimos años, con

[4] Se incluyen generalmente los nombres en inglés debido a que en este campo son ampliamente más conocidos en dicho idioma.

capacidades completamente superiores a las del período 2000-2010 (menos aún existe en el período 1963-2000).

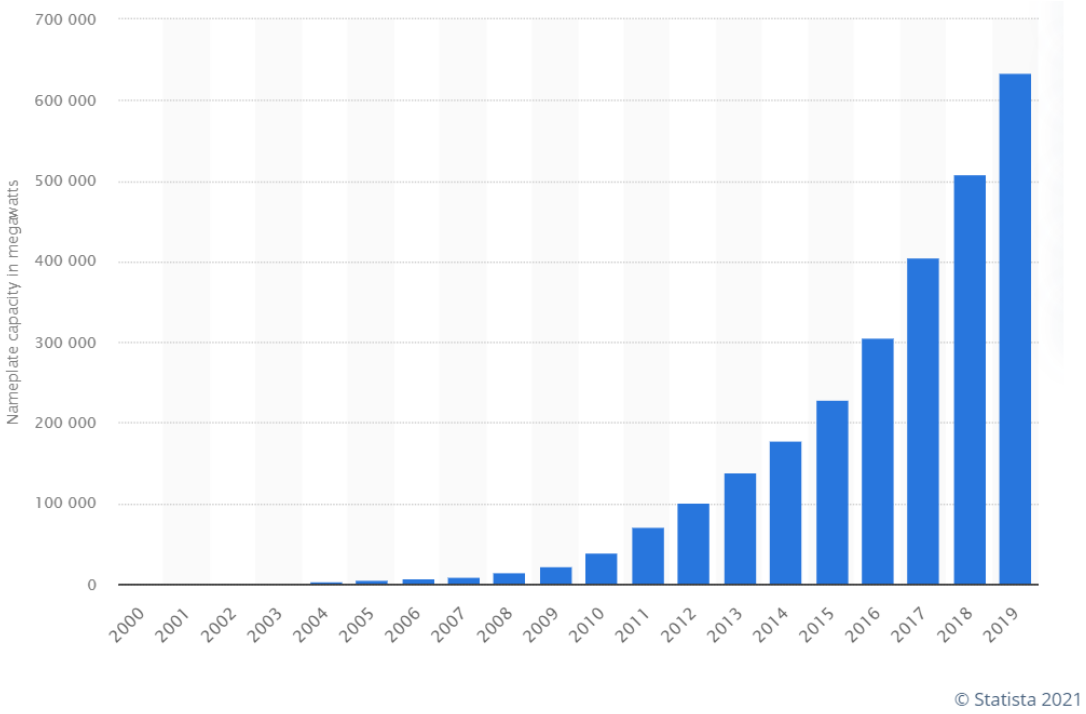


Figura 4. Capacidad solar fotovoltaica instalada a nivel mundial en el período 2000-2019 *(Statista, 2021)*

En la Figura *5* se presenta el esquema de funcionamiento de un panel solar de silicio. Con el que se trabajará en el presente trabajo se trata de un panel fotovoltaico de silicio monocristalino.

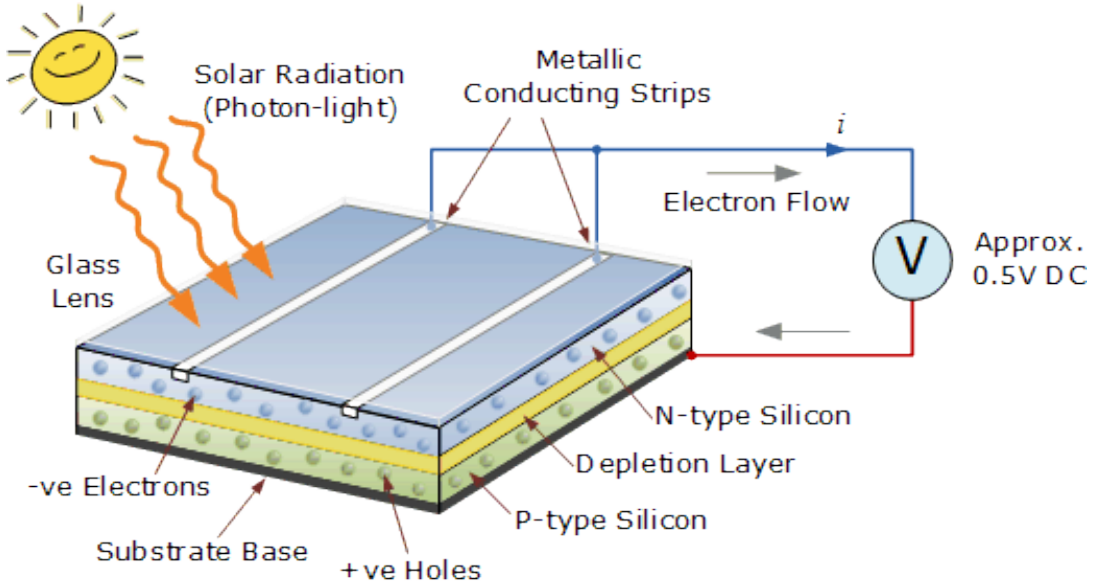


Figura 5. Esquema de funcionamiento panel fotovoltaico de silicio *(Alternative Enery Tutorials, 2019)*.

El material consiste en la unión de dos capas finas consistentes en silicio dopado de distinta manera. La capa superior es de tipo N, esto es, está dopado con átomos con exceso de electrones, como el fósforo. La capa inferior es de tipo P, en este caso se dopa con átomo con deficiencia de electrones como el boro. Cuando ambas capas se unen, se crea el material que es sensible a la radiación solar (célula fotovoltaica) – se establece un campo eléctrico en la región de la unión de ambas capas. La radiación solar incidente (fotones) liberan electrones en el material semiconductor de silicio, por el conocido efecto fotoeléctrico. Cuando ambas capas se unen mediante un circuito eléctrico externo, comienza el movimiento de los electrones (bajo radiación solar), dando lugar a una corriente eléctrica y a una tensión DC fija de entre 0.5 y 0.6 V (la tensión se produce debido al campo eléctrico interno que se establece en la unión PN).

1.4 ESP32

El microcontrolador ESP32 es una evolución del conocido ESP8266. Este último fue lanzado por la compañía Espressif en agosto de 2014. En aquel entonces era aún una desconocida empresa china. Fue un dispositivo lanzado con vistas a servir como puente WiFi entre otros microcontroladores. Pero con el uso, se percataron de que contaba con más poder de procesado y memoria de lo que se pensaba. En noviembre de ese mismo año, Ivan Grokhotkov introdujo la posibilidad de trabajar el ESP8266 desde el IDE de Arduino.

El ESP32 fue lanzado, en septiembre de 2016, para corregir los riesgos de seguridad asociados a su predecesor. Existían infinidad de dispositivos, especialmente instalados en viviendas, conectados mediante el microcontrolador ESP8266. No existía ningún medio de protección, por lo que cualquiera que tuviera acceso físico al dispositivo podía capturar información personal sensible, como las credenciales WiFi.

El ESP32, por tanto, hacía frente a dicho riesgo de seguridad introduciendo la encriptación de hardware y la firma del código. Pero iba más allá, solucionaba otros problemas de su hermano, incluye más GPIOs, ADCs y un DAC. Además, monta un procesador de doble núcleo más potente, Bluetooth y sensores táctiles.

En marzo de 2019 se introdujo un nuevo chip 'Chip7 2-2-A'. Esto dio lugar a nuevo producto: ESP32-S2. Hay algunas mejoras respecto al ESP32 de 2016 pero también algunas desventajas. Cuenta con un procesador de un solo núcleo, pero más potente (LX7 en lugar del LX6). Tiene menos SRAM y ROM, pero cuenta con soporte para memorias externas mayores. La mayor ventaja es el menor consumo de energía. No cuenta con Bluetooth o Ehernet MAC.

Este último dispositivo no viene para sustituir al ESP32, sino más bien como sucesor del ESP8266 en términos de funcionalidades y precio. El objetivo de la compañía Espressif con el dispositivo ESP32-S2 es rellenar el hueco existente entre el ESP8266 y el ESP32. El ESP8266 está destinado a desaparecer debido a las características y riesgos que comporta (Pérez, 2019).

En la Tabla 1 se presenta una comparativa entre los tres microcontroladores mencionados.

Tabla 1. Tabla comparativa ESP8266, ESP32 y ESP32-S2 (Kolipaka, 2019).

Connect and Innovate
连接与创新

ESP8266, ESP32, ESP32-S2

Specifications	ESP8266	ESP32	ESP32-S2
MCU	Single core Xtensa CPU up to 160MHz clock	Dual core Xtensa CPU up to 240MHz clock	Single core Xtensa CPU up to 240MHz clock
Internal Memory (IRAM+DRAM)	160kB	520kB	320kB
Connectivity	Wi-Fi 802.11bgn (HT20)	Wi-Fi 802.11bgn (HT40) Dual mode BT/BLE 4.2 Ethernet	Wi-Fi 802.11bgn (HT40)
External Memory	Up to 16MB SPI flash, (1MB executable XIP)	Up to 16MB SPI flash, (4MB XIP), Up to 4MB SPI RAM	Up to 1GB SPI flash, (7MB XIP, 4MB rodata), Up to 10MB SPI RAM
Peripherals	GPIO, SPI, UART, I2S, ADC	GPIO, SPI, UART, I2C, I2S, PWM, ADC, DAC, RMT, CAN, SD host/slave, low-power coprocessor	GPIO, SPI, UART, I2C, I2S, PWM, ADC, DAC, RMT, USB, low-power coprocessor
Hardware Security Features		SHA2, RSA, AES accelerators Transparent flash decryption Secure boot 1024 bit OTP	SHA2, RSA, AES, HMAC accelerators Transparent flash/RAM en/decryption (AES-XTS) RSA-PSS Secure boot 4096 bit OTP

Espressif Systems EclipseCon Europe 2019

2. OBJETIVOS

Diseño, fabricación y montaje de un sistema fotovoltaico de baja potencia con controladores y sensórica incluidos como sistema completo de referencia para su estudio y monitorización.

Diseño e implementación de un sistema SCADA con pantalla para la monitorización de los parámetros eléctricos y ambientales de una instalación fotovoltaica de pequeña envergadura, sin necesidad de ordenador. Almacenamiento de información en archivos de datos para su posterior descarga y consulta.

El sistema SCADA objetivo será de bajo coste, basado en software y hardware libre.

Caso de aplicación real – aprendizaje tanto de la parte de programación del SCADA como de la integración física de controladores, sensores y sistema fotovoltaico (con controlador, sistema de almacenamiento, paneles solares y cargas asociadas).

3. MATERIAL – HARDWARE Y SOFTWARE

Para trabajar en el proyecto se ha utilizado principalmente la placa Raspberry Pi 4B como herramienta de monitorización y visualización. Para la parte de programación, tanto del ESP32, como del SCADA y para el control remoto de la RPi, se ha utilizado un ordenador marca ASUS modelo ROG Zephyrus G14 GA401IV-HE003 con el sistema operativo Windows 10 Profesional.

3.1 Hardware

En esta subsección se van a recoger todos los materiales necesarios para el montaje físico del sistema fotovoltaico y los elementos para su monitorización.

3.1.1 Monitorización

Raspberry Pi

El equipo principal para la monitorización (recolección, procesado y muestra de datos a través de SCADA) es una Raspberry Pi 4B 8Gb RAM. A esta se le equipa una pantalla táctil LCD de 3,5'' de resolución 320x480 50 FPS[5], con ventilador de refrigeración (Figura 6 y Figura 7).

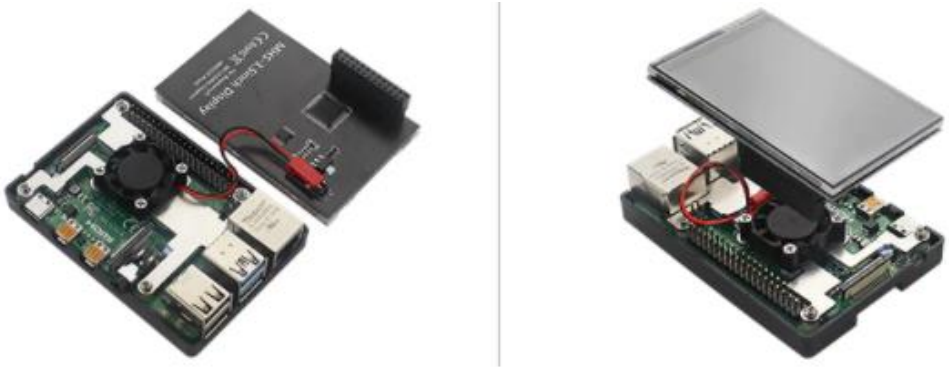


Figura 6. Raspberry PI con la pantalla y ventilador de refrigeración. Imagen tomada del producto comprado en la web.

Cabe mencionar el siguiente párrafo, presentación de la Raspberry Pi 4 en la página oficial: *Your tiny, dual-display, desktop computer…and robots brains, Smart home hub, media centre, networked AI core, factory controller and much more'* (Raspberry Pi, 2021). Recoge la esencia de la Raspberry Pi, útil en ámbitos muy variados, desde un uso como ordenador personal, pasando por unidades centrales de robots hasta como controlador industrial.

Las características de la RPi utilizada se recogen en la Tabla 2.

[5] Más detallado en el apartado 'accesorios' dentro de esta misma sección.

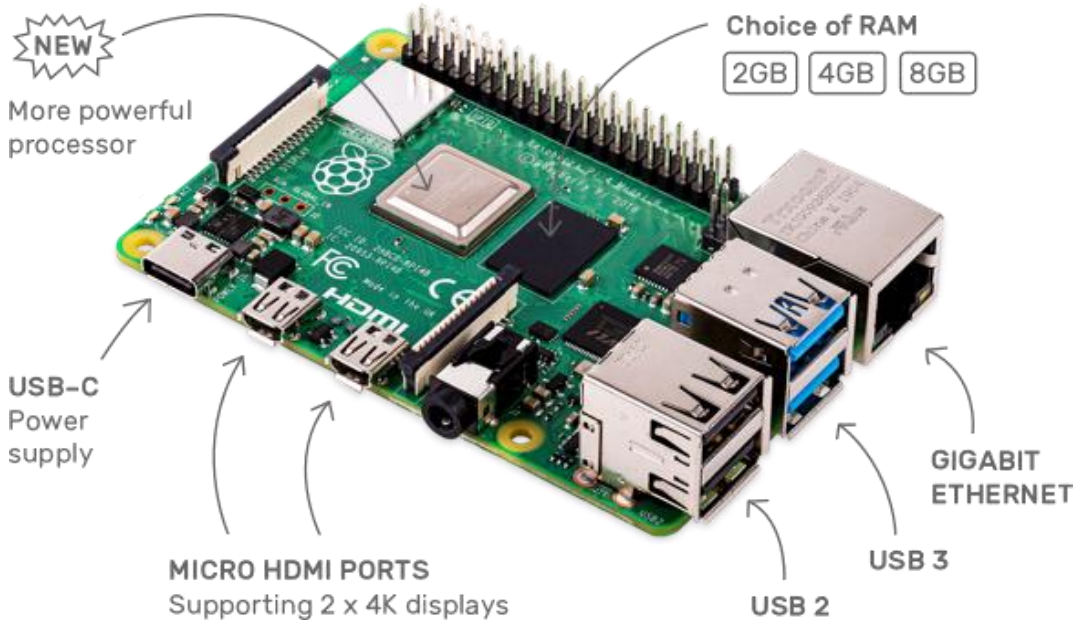


Figura 7. Raspberry Pi 4 Model B.

Tabla 2. Especificaciones RPi 4B.

SoC	Broadcom BCM2711	**Códecs**	H-265 (4kp60 decode) H.264 (1080p60 decode, 1080p30 encode)
CPU	Procesador de cuatro núcleos a 1,5 Cortex-A72 (ARM v8) 64-bit	**Puertos**	2xUSB 3.0 2xUSB 2.0
GPU	VideoCore VI	**Alimentación**	5 V/3 A vía USB-C, 5V / 3 A vía cabezal GPIO
Memoria	8GB LPDDR4-3200 SDRAM	**Expansión**	Cabezal GPIO de 40 pines
Conectividad	802.11ac WiFi/Bluetooth 5.0, Gigabit Ethernet	**Almacenamiento**	Slot Micro-SD
V-ideo y sonido	2x puertos micro HDMI 2.0 (hasta 4K@60Hz) Puerto de pantalla MIPI DSI Puerto de Cámara MIPI CSI Salida estéreo de 4 polos Puerto de vídeo compuesto	**Temperatura de operación**	0-50 ºC ambiente
Otros		OpenGL ES 3.1, Vulkan 1.0 PoE enabled	

ESP32

Es el encargado de recoger los datos de todos los sensores (cableados) y transmitirlos vía WiFi a la Raspberry Pi, donde serán procesados y transmitidos al sistema SCADA a través de un bróker.

Figura 8. ESP32 Dev Module.

Se elige la placa de desarrollo de la marca AZDelivery, modelo Dev Kit C V2 (ESP-WROOM-32). Ver Figura 8.

Esta es la segunda de las placas compradas. Se optó por cambiar de marca frente a una proveniente directamente de China, sin marca conocida, debido a los controles de calidad y posibles problemas de funcionalidad percibidos durante el desarrollo del proyecto. Además, esta placa de desarrollo está diseñada para poder trabajar en una breadboard (como se trabajará en este proyecto). Cuenta con un regulador de voltaje para que pueda ser alimentada a través del puerto USB que proporciona 5V (la alimentación del chip ha de ser 3.3 V[6]).

Las características del dispositivo se recogen en la Tabla 3.

En la Figura 9 se muestra la disposición de los pines de la placa de desarrollo utilizada. Esquema esencial para la correcta realización del proyecto. Estos pines serán referidos posteriormente durante el desarrollo del trabajo.

[6] El voltaje de 5 V no se debe conectar NUNCA a ningún pin del chip del ESP32 (5 V solo a través de la conexión USB).

Tabla 3. Características ESP32 Dev Kit C V2.

Voltaje fuente alimentación USB	5 VDC	**Voltaje entrada/salida**	3,3 VDC
Corriente de funcionamiento	500mA mínimos	**SoC**	ESP32-WROOM 32
CPU	Xtensa® single-dual-core 32-bit LX6	**Rango de frecuencias**	80 MHz/240 MHz
RAM	512 kB	**Memoria flash Externa**	4 Mb
Pines I/O	34	**Canales ADC**	18
Resolución ADC	12-bit	**Canales DAC**	2
Resolución DAC	8-bit	**Interfaces de comunicación**	SPI, I2C, I2S, CAN, UART
Protocolos WiFi	802.11 b/g/n (802.11n hasta 150 Mbps)	**Frecuencia WiFi**	2,4 – 2,5 GHz
Bluetooth	V4.2 – BLE y Bluetooth Classic	**Antena Wireless**	PCB
Dimensiones		56x28x13 mm (2,2x1,1x0,5 in)	

Descripción de los pines:

- Los pines GPIO 34-30 son GPIs (solo de entrada).
- Pines 6-11 conectados directamente a la memoria flash SPI integrada en el chip, no recomendados para otros usos.
- Los 10 sensores táctiles internos del ESP32 se encuentran en los pines 4, 0, 2, 15, 13, 12, 14, 27, 33 y 32 (se pueden utilizar para despertar al ESP32 del 'deep sleep').
- Los pines que pueden ser utilizados como ADC son: 36, 37, 38, 39, 32, 33, 34, 35, 4, 0, 2, 15, 13, 12, 14, 27, 25, 26.
- El DAC se corresponde a los pines 25 y 26.
- Pines reloj en tiempo real: 36, 39, 34, 35, 25, 26, 33, 32, 4, 0, 2, 15, 13, 12, 14 y 27.
- Pines PWM: todos los pines que puedan ser utilizados como salidas (no 34-39).
- Pines interfaz I2C: 21 (SDA) y 22 (SCL).
- Pines interfaz SPI: VSPI – 23 (MOSI), 19 (MISO), 18 (CLK), 5 (CS) y HSPI – 13 (MOSI), 12 (MISO), 14 (CLK) y 15 (CS).
- Pines de conexión, para poner el ESP32 en modo bootloader o flashing: 0, 2, 4, 5, 15 (en 'high' durante el arranque), 12 (en 'low'). Pines no recomendados para pines de conexión 'high' en arranque: 1, 3, 5, 6, 11 (conectados a la memoria flash SPI integrada – por ello no se recomienda su uso), 14 y 15.

El botón EN es el pin de habilitación del regulador de 3,3 V. Tiene estado 'pull-up' y necesita ser conectado a tierra para desactivar el regulador. Puede ser conectado a un pulsador para reiniciar el ESP32.

El puerto de conexión es de tipo microUSB. Se compone de un chip CP21202 (Silicon Laboratories) que permite la comunicación de USB a UART Serie.

La placa de desarrollo tiene también un sensor Hall y un sensor de temperatura, que pueden ser considerados para ciertas aplicaciones.

La corriente máxima absoluta consumida por un GPIO es de 40 mA (AZ-Delivery, 2021).

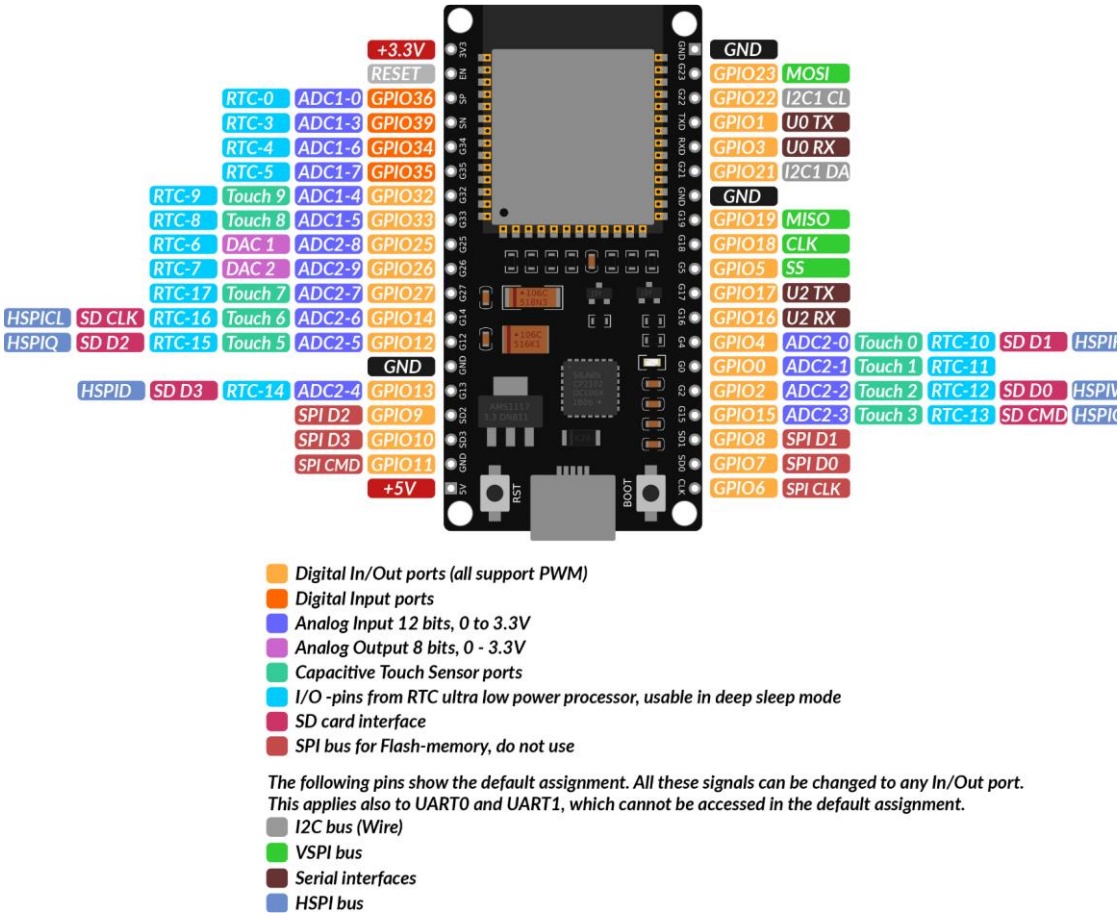


Figura 9. Disposición de los pines placa de desarrollo ESP32 Dev Kit C V2.

Para más información consultar el manual de la placa de desarrollo mencionada o el manual del chip ESP32 de Espressif, disponible en la web oficial (Figura 10).

Figura 10. Logo y web Espressif.

Router

Permitirá implementar la red local para conectar el ESP32 y la Raspberry Pi bajo la misma red y poder realizar la transmisión de los datos. El router utilizado se muestra en la Figura 11.

Es de la marca COMTREND modelo AR-5387un. Se trata de un router inalámbrico ADSL2+802.11n. El punto de acceso WiFi soporta WEP, cifrado inalámbrico WPA/WPA2 y autenticación 802.1x. Es compatible con WDS y utiliza WMM (QoS) para priorizar el tráfico en la red. Tiene la opción de habilitar WPS (Bandaancha.st, 2018).

Características (Tabla 4):

Tabla 4. Características router.

4 puertos LAN Fast Ethernet 10/100 Base-T (R-J45)
Wifi N (802.11n draft 2.0) MIMO 2x2 (hasta 300 Mbps brutos), compatible con 802.11 b/g. Dos antenas externas
Soporta cifrado WPA2
Botón de reset
Puerto USB
Puerto eni
Puerto eno

Figura 11. Router.

3.1.2 Sensores

Para la captación de datos para la monitorización del sistema fotovoltaico de referencia se utilizan distintos sensores, para caracterizar tanto el funcionamiento de la instalación como los parámetros ambientales en la localización de esta.

FZ0430

Figura 12. Sensor FZ0430.

Sensor para la medida de tensiones. Marca WAVGAT modelo FZ0430 (MH-Electronic) (Figura 12).

El principio de funcionamiento es el de un divisor de tensiones. La tensión de entrada se divide entre 5, esto es, si se alimenta con 5V provenientes, en este caso, del ESP32, la máxima tensión que se puede medir en estas condiciones es de $5\ \text{V} \times 5 = 25\ \text{V}$. En el escenario en que se alimentase con 3,3 V, la máxima tensión admisible para ser medida sería 16,5 V.

El ADC del ESP32 es de 12 bits, por lo que la resolución analógica (la salida del sensor es analógica) de este sensor es de $5\ V/4095 = 0{,}0012\ V$. La mínima tensión que puede leer este sensor es por tanto $0{,}0012\ x\ 5 = 0{,}006\ V$ (Llamas, 2016).

Conexionado (Tabla 5):

Tabla 5. Conexionado sensor FZ0430.

VCC	Terminal positivo de la tensión a medir
GND	Terminal negativo (o tierra) de la tensión a medir
+	Alimentación del sensor – positivo – 5 ó 3,3 VDC
-	Alimentación del sensor – negativo (o tierra)
s	Salida del sensor a ser leída por un pin de entrada analógica del ESP32.

ACS 712

Sensor para la medida de corriente. Marca WAVGAT modelo ACS712 30A (ELC 30A).

Se trata de un sensor (Figura 13) con salida analógica. Se alimenta con 5 V. La tensión de salida varía en función de la corriente que pase por él. Si no pasa corriente, la salida es VCC/2 (2,5 V). La salida analógica se corresponde con 100 mV/A.

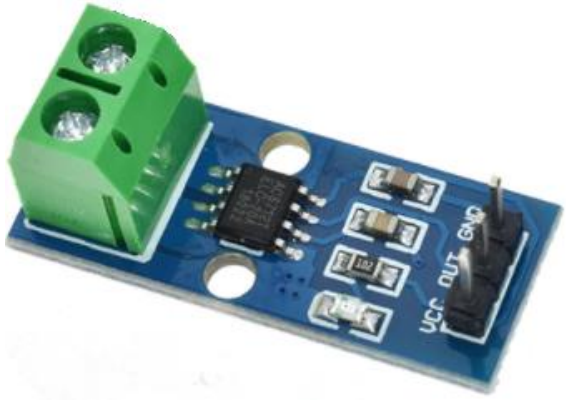


Figura 13. Sensor ACS 712.

Conexionado (Tabla 6):

Tabla 6. Conexionado sensor ACS 712.

Clema – terminal 1	Cable del circuito cuya corriente se quiere medir
Clema - terminal 2	
VCC	+ Alimentación +5 VDC
GND	- Alimentación (tierra)
OUT	Salida analógica del sensor al ESP32

DHT22

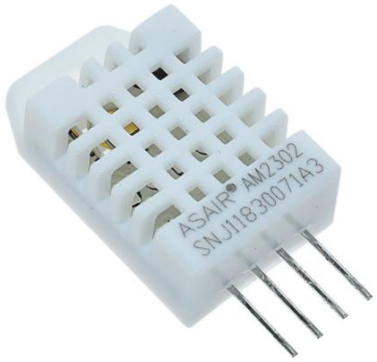


Figura 14. Sensor DHT22.

Sensor para la medida de la humedad y temperatura ambiente. Marca WAVGAT modelo DHT22 (Figura 14).

En este caso se trata de un sensor con salida digital. La medida de humedad la realiza a través de un sensor capacitivo, mientras que la medida de la temperatura ambiente (aire circundante) se realiza mediante un termistor. Las características se muestran en la Tabla 7.

El conexionado de este sensor se presenta en la Tabla 8 (los pines se cuentan de izquierda a derecha mirando el sensor de frente) (Blog by Unit Electronics, 2021).

Tabla 7. Características sensor DHT22.

Rango medida humedad	0-99,9 % (en el rango 0 – 50 ºC)
Error medida humedad	± 2 %
Rango medida temperatura	-40 ~ 80 ºC
Error medida temperatura	± 0,5 ºC (max ± 1 ºC)
Resolución	16 bits
Muestreo	2 s
Tensión de trabajo	3 V – 5,5 V

Tabla 8. Conexionado sensor DHT22.

1 (VCC)	+ Alimentación
2 (DATA)	Pin salida de datos. Conexión a entrada digital del ESP32.
3 (NC)	No usado
4 (GND)	- alimentación (tierra)
Observaciones: se utiliza una resistencia de 4,7 kW (en modo pull-up) entre el pin DATA y el VCC	

BH1750

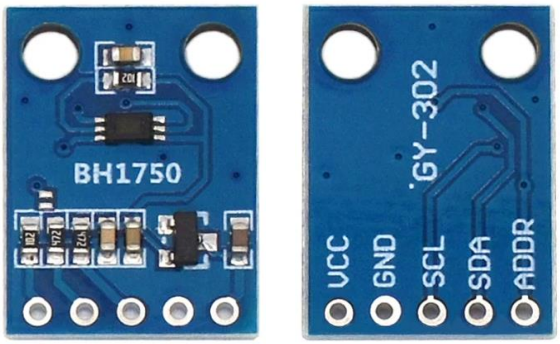


Figura 15. Sensor BH1750.

Sensor para la medida de la iluminancia (flujo lumínico). Marca WAVGAT modelo BH1750 (GY-302) (Figura 15). Incorpora el chip original de la empresa 'Rohm semiconductor' BH1750FVI.

Este sensor tiene un mejor desempeño que un foto-resistor LDR ya que no hay que realizar ninguna conversión para llegar al valor final deseado.

Tiene un conversor interno de 16 bit que entrega la salida digital en formato I2C (bus I2C). Este sensor da la intensidad luminosa en Lux (1 lux = l lumen/m2). Esta unidad se usa en fotometría para medir este parámetro, ya que tiene en cuenta las diferentes longitudes de onda en función de la luminosidad (modelo estándar de la sensibilidad a la luz del ojo humano).

En la Tabla 9 se presenta un rango típico de valores de intensidad luminosa (Lux), para tener una comparativa (Naylamp Mechatronics, 2021) (Llamas, 2016).

Tabla 9. Rangos típicos iluminancias.

Noche	0,001-0,02
Luz de medianoche	0,02-0,3
Interior nublado	5-50
Exterior nublado	50-500
Soleado interior	100-1000
Bajo el sol de verano	~100000
Habitación/salón	150-300
Estándar de vídeo casero	1400
Mesa oficina/lectura	500-700
Supermercados/exposiciones	750-1000
Mesas dibujo/trabajo	1000-1500

Las características de este sensor se recogen en la Tabla 10.

El conexionado del sensor se ha de realizar de acuerdo con la Tabla 11.

Tabla 10. Características sensor BH1750.

Tensión alimentación	3 – 5 VDC
Respuesta espectral	Similar al ojo humano
Rango de medición	1-65535 lux
Rechazo de ruido	a 50/60 Hz
Observaciones: baja dependencia de la medición en función de la fuente de luz (halógeno, led, incandescente, luz de día, etc.)	

Tabla 11. Conexionado sensor BH1750.

GND	- Alimentación (tierra)
ADD	No usado
SDA	Conectado al pin SDA del ESP32
SCL	Conectado al pin SCL del ESP32
VCC	+ Alimentación

NOTA: para información visual sobre el conexionado de los sensores se puede consultar el Anexo A: Esquema Conexiones Fritzing - Figura 53 y Figura 54.

3.1.3 Instalación fotovoltaica

Panel y estructura

El panel empleado en el proyecto para la construcción de la pequeña instalación solar de referencia es de silicio monocristalino, de 50 W y parcialmente flexible (preparado principalmente para ser montado en una instalación de vehículo o barco). Éste se ha

instalado sobre una estructura realizada a medida, en madera, con una inclinación de 20º[7]. Dispone de una conexión rápida hembra de alimentación 5,5 x 2,5 mm. Este conector soporta una corriente de 5 A y un voltaje de 24 VDC (Figura 16).

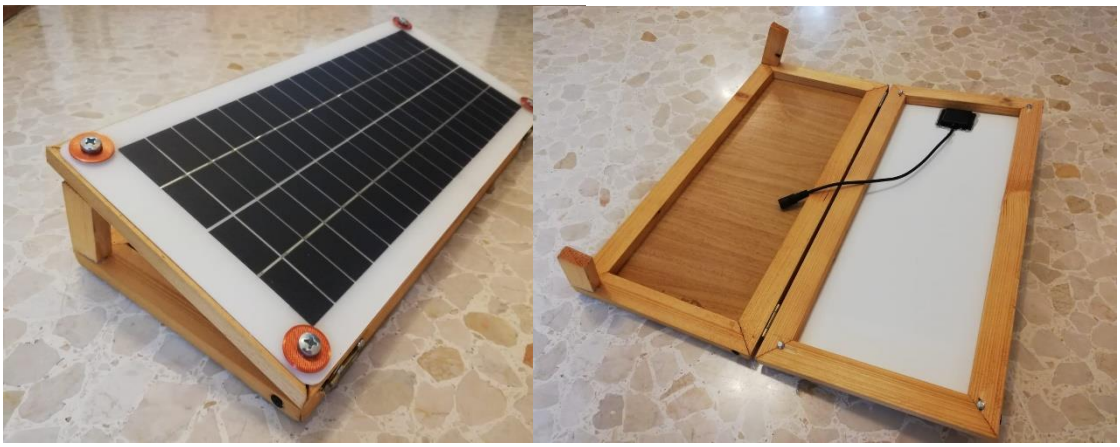


Figura 16. Panel solar con estructura de soporte e inclinación.

Las características del panel se recogen en la Tabla 12.

Tabla 12. Características panel solar.

Potencia	50 W[8]
Voc	20 V
Isc	1,15 A
Coef. temp. Isc	± 0,05 %/ºC
Coef. temp. Voc	- 0,33 %/ºC
Coef. temp. Potencia	- 0,23 %/ºC
NOCT[9]	45 ºC (± 5 ºC)
Salida	18 V – 1,15 A (5 V – 2 A con conversor para USB)
Eficiencia	19,5 %
Dimensiones	435x197 mm

Controlador

El controlador utilizado para regular la potencia proveniente del panel, así como controlar la carga de la batería es de tipo PWM – modelo de 30 A (Figura 17). Este controlador es ampliamente empleado en este tipo de instalaciones de baja potencia[10].

[7] Inclinación óptima para la latitud y la longitud en la que se encuentra la instalación. Dicho valor puede variar ligeramente en las distintas épocas del año.
[8] Potencia según descripción del artículo comprado. Tras ser comprobada y calculada parece que la potencia real dista de ese número – está en torno a 20 W.
[9] Temperatura de operación nominal de la celda
[10] Para instalaciones de mayor potencia es recomendable utilizar un controlador de tipo MPPT. Este controlador permite obtener la máxima potencia de los paneles solares. Lo hacen trabajar en el punto de máxima potencia. El coste del controlador está justificado en este tipo de instalaciones de mayor potencia, ya que obtienen un rendimiento mayor, entre el 20 y el 30%.

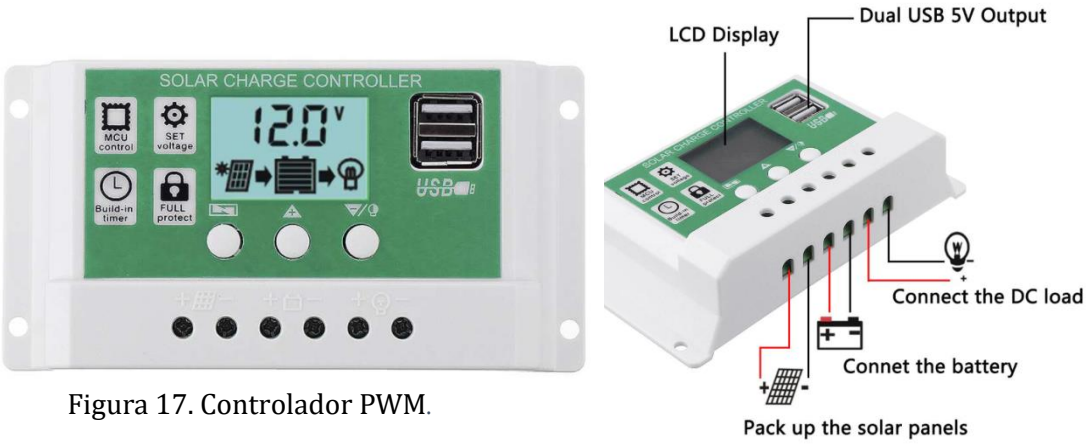


Figura 17. Controlador PWM.

Las características de este controlador se pueden consultar en la Tabla 13. Los apartados voltaje flotante de carga, parada por descarga y reconexión por descarga son ajustables y los valores que aparecen son los que se han establecido para este proyecto. Son directamente dependientes de la batería seleccionada. Ver apartado 'batería' para más información de cómo se han establecido dichos valores.

Tabla 13. Características controlador PWM.

Tensión batería	12/24 V (detección automática)
Corriente de carga	30 A
Corriente de descarga	10 A
Potencia máxima PV	390 W (12 V) – 780 W (24 V)
Tensión máxima paneles	23 V para batería de 12 V
	46 V para batería de 24 V
Tensión de carga de la batería	14,2 V/14,4 V/14,6 V
Voltaje flotante de carga	13,7
Parada por descarga	12,3
Reconexión por descarga	12,6 V
Reconexión para carga	13 V
Consumo propio	< 10 mA
Temperatura de operación	-35-60 ºC
Dimensiones	133,5x70x35 mm

Batería

La batería seleccionada es de la Marca Master U-Power, modelo MU-UP7.2-12F2 (Figura 18). Es de plomo, tecnología AGM, 7,2 Ah y 12 V. Los bornes son de tipo faston F2 6,3 mm.

Al ser una batería monoblock sellada no requiere mantenimiento. Además, es adecuada para usarse en lugares con poca ventilación. El material de fabricación es ABS, por lo que es apto para soportar condiciones de trabajo inclementes.

Las características de la batería se presentan en la Tabla 14.

Figura 18. Batería.

Tabla 14. Características batería.

Voltaje	12 V
Capacidad	7,2 Ah
Dimensiones	151x65x101 mm
Masa	1,99 kg
Máxima corriente de carga	2,16 A
Resistencia interna	18 mΩ
Bornes	Faston 6,3 mm
Profundidad de descarga	60 %
Uso cíclico (‘cycle use’)	14,5-15,0 V
Uso stand-by	13,5-13,8 V

Según el valor de la profundidad de descarga, se puede calcular el voltaje máximo al que puede bajar la batería, en su uso continuo, sin ser dañada. Este valor se corresponde con el del apartado anterior del PWM de ‘parada por descarga’ (a ese valor se le sumó 0,1 V más para estar en el lado de la seguridad y evitar daños en el equipo – trabajaría en torno al 70 %). Para averiguar este valor se utiliza la Tabla 15.

Tabla 15. Estado de carga de una batería en función de su voltaje.

Estado de carga	**Tensión de la batería (12 V)**	**Estado de carga**	**Tensión de la batería (12 V)**
100	12,7	90	12,5
80	12,42	70	12,32
60	12,20	50	12,06
40	11,90	30	11,75
20	11,58	10	11,31
0		10,50	

3.1.4 Accesorios

Pantalla RPi

Se trata de una pantalla táctil de tipo resistivo. Es el modelo MHS35 de 50 FPS. Admite una señal de SPI de hasta 125 MHz[11] (Figura 19).

Figura 19. Pantalla RPi. Parte delantera (izqda.), trasera (dcha.) y ventilador de

Características (Tabla 16):

Tabla 16. Características pantalla táctil.

Resolución	320x480[12]
SKU	MHS3528
Color Display	RGB 65K color
Interfaz LCD	SPI
Tipo LCD	TFT
Tipo panel táctil	Resistivo
Chip panel táctil	XPT2046
Driver IC	ILI9486
Tensión operación	3,3/5 V
Temperatura de operación	-20~70 ºC
Temperatura de almacenamiento	-40~70 ºC
Área activa	48,96x73,44 mm
Dimensiones	85,42x55,60 mm
Consumo	0,16A @ 5 V

[11] También existe el modelo LCD35, de 5 FPS, que admite una señal SPI de hasta 32 MHz. Se optó por la de 50 FPS debido a la mayor calidad de visualización y comportamiento general, para visualizar correctamente el contenido de la RPi, en particular el SCADA diseñado (pantalla estable y sin parpadeo).

[12] Se puede aumentar ligeramente sin dificultar la visualización, mejorando la apariencia del contenido mostrado en pantalla

Carcasa RPi

Carcasa acrílica para la Raspberry Pi 4B con pantalla táctil, para la protección de la pantalla y componentes aledaños (Figura 20).

Figura 20. Carcasa RPi.

Alimentación RPi portátil

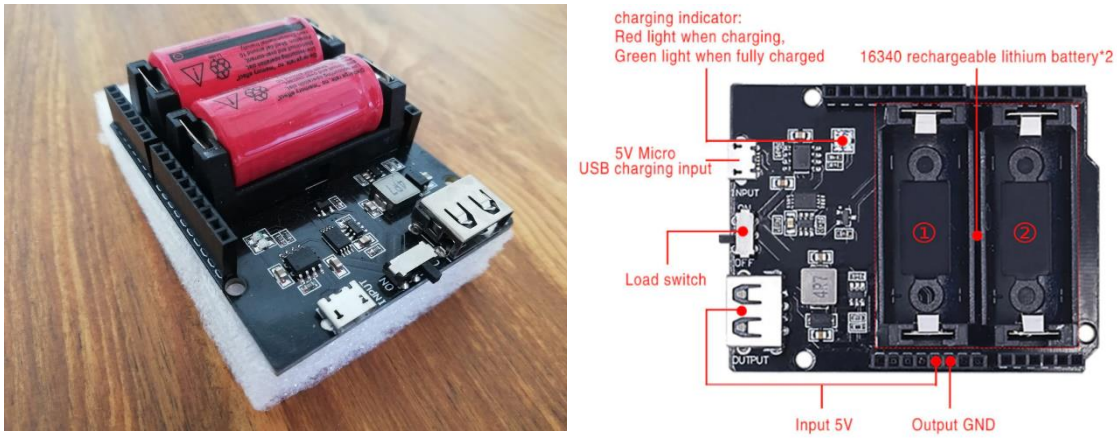


Figura 21. Alimentación portátil.

Es una fuente de alimentación portátil que utiliza 2 baterías de litio 16340 (Figura 21). Se cargan a través del puerto Micro-USB (Input). La salida es a 5 V a través de un puerto USB-A. Lleva incorporado un controlador de carga y un regulador de tensión.

Las baterías utilizadas son dos TR 16340 Li-ion, 3,7 V, 2500 mAh. En total se dispone de 5000 mAh (están en paralelos). La placa también podría funcionar con una sola batería, pero tendría la mitad de la capacidad.

Breadboard

Breadboard marca SunFounder fabricada en plástico ABS con leyenda impresa en negra (Figura 22). Los contactos internos están fabricados en bronce (muelle de cobre de doble hoja). En la base de la placa cuenta con cienta autoadhesiva para su fijación. Cuenta con 830 puntos, 200 en 2 rieles de potencia y 630 en una matriz de 30 x 10. Cuenta con clips de extensión para anclarse a otras placas.

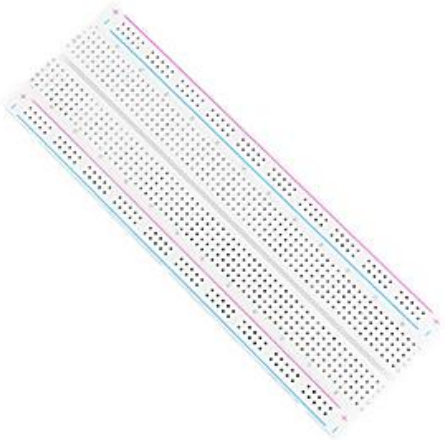


Figura 22. Breadboard.

Alimentación protoboard

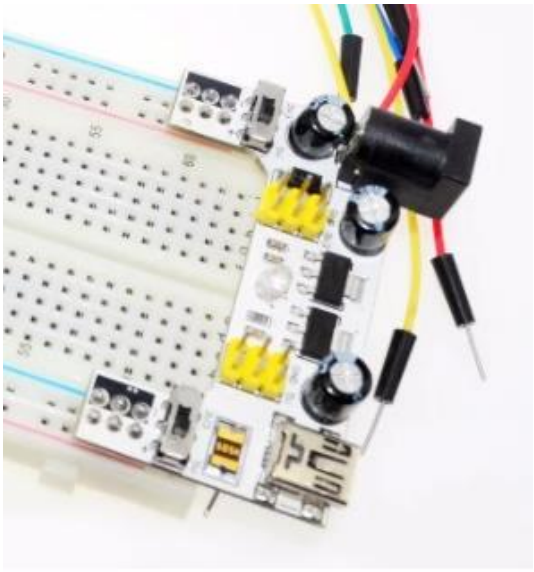


Figura 23. Alimentación protoboard.

Proporciona, mediante cable USB o cable de potencia (conexión hembra), una tensión fija en los dos extremos de la protoboard, seleccionables por separado a 5 v o 3,3 V (tensiones comunes de trabajo para la electrónica considerada). Ver Figura 23.

Cables 'jumper'

Cables marca Neuftech con terminales preparados para trabajar con los diferentes sensores, protoboard y ESP32. Compatibles con matrices de 2,54 mm (breadboard). Figura *24*.

Figura 24. Cables 'jumper'.

Interruptores

Interruptores de activación de los distintos componentes del montaje. Se ha optado por instalar interruptores de 3 pines con luz LED, para saber cuándo está polarizada una parte del circuito. Están certificados para 12 V y 16 A. Tipo SPST (encendido/apagado). Figura *25*.

Figura 25. Interruptor.

3.2 Software

En esta sección se van a exponer todos los recursos informáticos necesarios para el desarrollo del proyecto.

3.2.1 Arduino IDE

Este programa es el 'Integrated Development Environment' de Arduino. Esto es, el conjunto de herramientas de software que permiten a los programadores desarrollar y grabar todo el código necesario para hacer que cierto dispositivo funcione en la forma que se desee (en nuestro caso, el ESP32) (Arduino, 2021).

Se diseñó específicamente para ser usado con los dispositivos Arduino, pero con el tiempo se han ido añadiendo funcionalidades, tanto por parte del soporte oficial de Arduino como de la comunidad, para poder ser utilizado con otros que existan en el mercado, como en este caso el ESP32 con el que se está trabajando en el proyecto.

Figura 26. Arduino IDE (UNO, 2019).

Este IDE permite escribir, depurar, editar y grabar el programa ('sketches') de forma sencilla y con ayudas visuales. Como se puede apreciar en la Figura 26, la interfaz consta de:

- Área de codificación: donde se desarrolla el proyecto. Tiene un código de colores para facilitar la escritura del código.
- Consola de depuración: donde van apareciendo los mensajes relacionados con las acciones sobre el código escrito, así como los posibles errores que pudiera haber dentro del código, indicando la línea en la que se encuentran.
- El botón verificar comprueba el código para ver si hay errores antes de subirlo al dispositivo
- El botón subir manda el código al dispositivo para que se ejecute en el mismo y realice las funciones deseadas.
- Por último, el botón monitor serie abre una ventana donde se van imprimiendo en pantalla lo establecido mediante las funciones de impresión en el código (si las hubiera) – datos transferidos desde y hacia el dispositivo. Además, a veces, permite instruir al dispositivo conectado en tiempo real.

La instalación de este software se puede realizar mediante los comandos:

sudo apt-get update
sudo apt-get dist-upgrade
sudo apt install Arduino-core

Aunque de esa forma se instala una versión antigua del IDE que no incluye algunas funcionalidades necesarias para el ESP32. Una vez comprobado esto (ya no es recomendado este método), se optó por ir a la dirección Arduino.cc/en/software y descargar la versión *Linux ARM 32 bits* del programa.

Además, para la configuración del ESP32 se ha de ir a preferencias/URLs adicionales y añadir la del paquete ESP32. Después, se ha de buscar 'esp32' en herramientas/placas/gestor de tarjetas e instalar el paquete que da soporte a todas las placas (Random Nerd Tutorials, 2018).

3.2.2 Node-RED

Node-RED es una herramienta de programación visual para unir dispositivos, APIs[13] *y servicios online de nuevas e interesantes formas. Proporciona un editor web (basado en el navegador) que simplifica el proceso de unir distintos 'flows' usando un gran número de nodos predefinidos que pueden ser desplegados tan sólo con un doble click* (Node-RED, 2021).

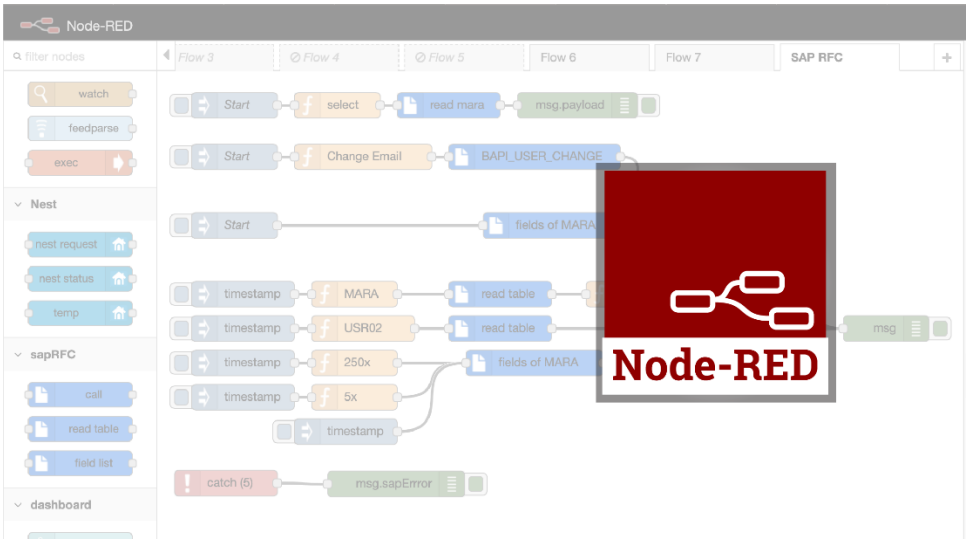


Figura 27. Node-RED.

Esta herramienta lo que hace es mostrar visualmente las relaciones entre funciones. Permite al usuario poder programar sin escribir en líneas, como se puede apreciar en el fondo de la Figura 27.

Se ha elegido este software para la realización del SCADA en el marco de este proyecto debido a la sencillez de aprendizaje y uso, a su robustez y la necesidad de bajos recursos de cómputo. Además, existe una gran cantidad de recursos en línea para poder aprender a utilizarlo y poder desarrollar un proyecto desde cero. Otro factor que tener muy en cuenta es el soporte directo para Raspberry Pi, objetivo de este proyecto, disponible en los repositorios oficiales de la distribución. Esto marca una gran diferencia respecto a otras plataformas disponibles para el desarrollo de una aplicación SCADA.

[13] API: Interfaz de Programación de Aplicaciones ('Application Programming Interface').

Esta herramienta fue creada, en el año 2013, por dos integrantes del grupo de Servicios de Tecnologías Emergentes, de la empresa IBM, Nick O'Leary y Dave Conway-Jones. Apareció como solución a la complejidad de integración del hardware con otros servicios.

Se basa en NodeJS y la librería JavaScript D3.js. El primero proporciona la potencia suficiente para que Node-RED sea escalable y fiable. Permite la programación en JavaScript del lado del servidor.

Se basa en una estructura cuya unidad mínima son los nodos. Son arrastrados desde una paleta lateral directamente al espacio de trabajo. Una vez desplegados en el espacio, se hace doble clic sobre ellos y se abre una ventana de configuración, particular a cada nodo. La siguiente unidad son los flujos ('flows'), que son agrupaciones de nodos, conectados mediante líneas, de manera visual. Finalmente, se puede tener un conjunto de flujos (en distintas páginas gráficas) para poder conseguir el resultado final deseado.

Es una herramienta de código abierto, disponible en GitHub[14] (Aprendiendo Arduino, 2020)[15].

Para instalar Node-RED se puede hacer uso del comando (no recomendado – mejor desde la web) (Arduino, 2020):

sudo apt-get install nodered

Desde la web:

bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)

--node14

Para iniciar el servicio de Node-RED se utiliza:

sudo systemctl start nodered.service

En nuestro caso se ha establecido que se inicie con el inicio del sistema, para que no haya que iniciarlo manualmente:

sudo systemctl enable nodered.service

3.2.3 Mosquitto Broker

Para la transmisión de datos entre el ESP32 y el SCADA se necesita un protocolo y un intermediario (servidor central, (Random Nerd Tutorials, 2017)) que reciba y transmita los datos, desde el ESP32 al SCADA (y viceversa si se diera el caso). Dicho intermediario se denomina bróker.

Como protocolo, se ha optado por el protocolo MQTT, de nuevo, por su versatilidad y gran cantidad de información para su implementación. MQTT proviene de 'Message Queuing Telemetry Transport'. Se trata de un protocolo de comunicación M2M ('machine-to-

[14] https://github.com/node-red/node-red
[15] Ver (Castillo, 2018) y (Programarfacil.com, 2019)

machine'). Se basa en el protocolo TCP/IP como base para la comunicación. Fue creado por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom en 1999 cuya finalidad en aquel entonces era conectar distintos dispositivos utilizados en la industria petrolera. Pasó de ser propietario a ser liberado en 2010 y posteriormente nombrado estándar en 2014 por la OASIS[16].

Las ventajas de este protocolo son:

- Ventajas del patrón pub/sub: escalabilidad, asincronismo y desacoplamiento entre clientes.
- Sencillez y ligereza: interesante para dispositivos IoT, que suelen ser de escasa potencia.
- Menor necesidad de recursos. Esto implica menor consumo de energía. Altamente provechoso para dispositivos que están encendidos 24 horas al día y 7 días a la semana y, además, alimentados por batería (caso del ESP32).
- Poco ancho de banda
- Se pueden implementar medidas de seguridad y calidad del servicio (QoS)
- Protocolo altamente utilizado y comprobado. Robustez y fiabilidad.

El funcionamiento del MQTT se basa, como se ha mencionado anteriormente, en el servicio de mensajería push con método publicar/suscribir (pub/sub) en el que existe un servidor central al que se conectan todos los clientes. Los mensajes son filtrados mediante la disposición de estos en 'topics' organizados jerárquicamente. Si un cliente publica un mensaje dentro de un determinado topic, otros clientes que se conecten (suscriban) a ese topic en particular podrán ver los mensajes publicados.

Los clientes se conectan vía TCP/IP al bróker (éste mantiene un registro de los clientes conectados). Esta conexión no se cierra hasta que el cliente así lo indica. Se emplea el puerto 1883 o 8883 si se utiliza TLS. Para conectarse, el cliente envía un mensaje *connect* con información necesaria para realizar la conexión. El bróker responde con mensaje *connak* con información relativa al resultado de la conexión solicitada. Si el cliente quiere enviar un mensaje utiliza *publish*, el cual contiene el topic y el payload (mensaje). Así mismo, para suscribirse o desuscribirse a un topic se emplea *subscribe* y *unsubscribe* respectivamente. A esto el servidor responde con *suback* y *unsuback*. Finalmente, con motivo de asegurar que la conexión esté activa, los clientes mandan un mensaje *pingreq* al cual el bróker le responde un *pingresp*. Si el cliente se quiere desconectar, envía un *disconnect*.

La estructura de un mensaje MQTT consta de 3 partes bien diferenciadas. Cabecera fija (2-5 bytes), de carácter obligatorio. Contiene un código de control (1 byte) y la longitud del mensaje (1-4 bytes). Cabecera variable, de carácter opcional, agrega información adicional necesaria en ciertos mensajes. Por último, está el contenido (*payload*), también de carácter opcional, es el contenido real del mensaje. Máximo de 256 Mb, pero para una aplicación real, el máximo oscila entre 2 y 4kB (Llamas, 2019). Representación gráfica en la Figura 28.

El bróker elegido para el protocolo seleccionado ha sido 'Eclipse Mosquitto' (Figura 29), por su amplio soporte para Raspberry Pi. Además, es muy ligero y útil para usarse en dispositivos de bajo consumo de potencia.

Este bróker es de código abierto. Es un bróker de mensajes que implementa el protocolo MQTT en las versiones 5.0, 3.1.1 y 3.1. Proporciona un método de transmisión de mensajes en la forma de publicar/suscribir ('Publish/Subscribe'). Esto es lo que lo hace muy útil para

[16] OASIS: 'Organization for the Advancement of Structured Information Standards'.

aplicaciones IoT en las que se requiera envío y recepción de mensajes con dispositivos de baja potencia, como sensores o dispositivos móviles, como teléfonos, microcontroladores (caso del ESP32) u ordenadores embebidos (Eclipse Mosquitto, 2021).

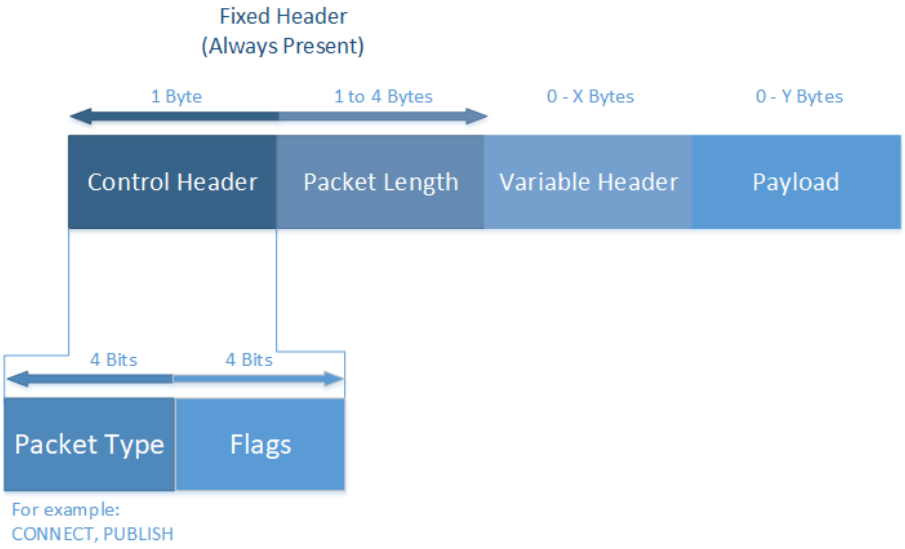


Figura 28. Estructura de un mensaje MQTT (Telit, 2021).

Este bróker puede estar alojado en cualquier parte de la red, hasta incluso en la nube. En el caso que se ocupa en este proyecto, estará ejecutándose en la misma Raspberry Pi en la que se está mostrando el SCADA.

Figura 29. Eclipse Mosquitto.

Para instalar Mosquitto Broker (Random Nerd Tutorials, 2017):

sudo apt update
sudo apt install -y mosquitto mosquitto-clients

Para que se inicie con el sistema automáticamente:

sudo systemctl enable mosquitto.service

Finalmente, para comprobar el estado del bróker se puede hacer uso del comando:

Mosquitto -v

3.2.4 VNC Viewer y Putty

Se trata de dos programas para la conexión remota con la Raspberry Pi. VNC Viewer proporciona un escritorio físico remoto, igual que el de la Raspberry Pi, en tiempo real, para ver lo que está sucediendo y controlarla sin necesidad de tener físicamente un monitor conectado.

Para poder usar VNC Viewer, éste ha de estar instalado en la RPi. Normalmente viene por defecto en el sistema operativo utilizado: RPi OS. En el caso de que no viniera puede ser instalado con los siguientes comandos (Programo ergo sum, 2021):

sudo apt-get update
sudo apt-get install realvnc-vnc-server
sudo apt-get install realvnc-vnc-viewer

Si viene ya preinstalado lo que se debe hacer es activarlo, se habilita en opciones avanzadas, dentro del menú configuración.

También se ha de tener instalado el cliente VNC en el dispositivo desde el que queramos controlar la RPi de forma remota y gráficamente, en este caso, un PC Windows (Figura 30).

Por otra parte, Putty permite el acceso a la RPi tan sólo por línea de comandos. Este programa es esencial tenerlo configurado, tanto en la RPi como en el dispositivo desde el que queramos controlarla, ya que al estar trabajando con una pantalla que debemos configurar, en el caso de que surja algún problema y se quede el sistema operativo sin escritorio, se puede acudir a los comandos remotos para intentar solucionar el problema, evitando así males mayores (como tener que formatear por completo la tarjeta microSD y reinstalar el sistema operativo).

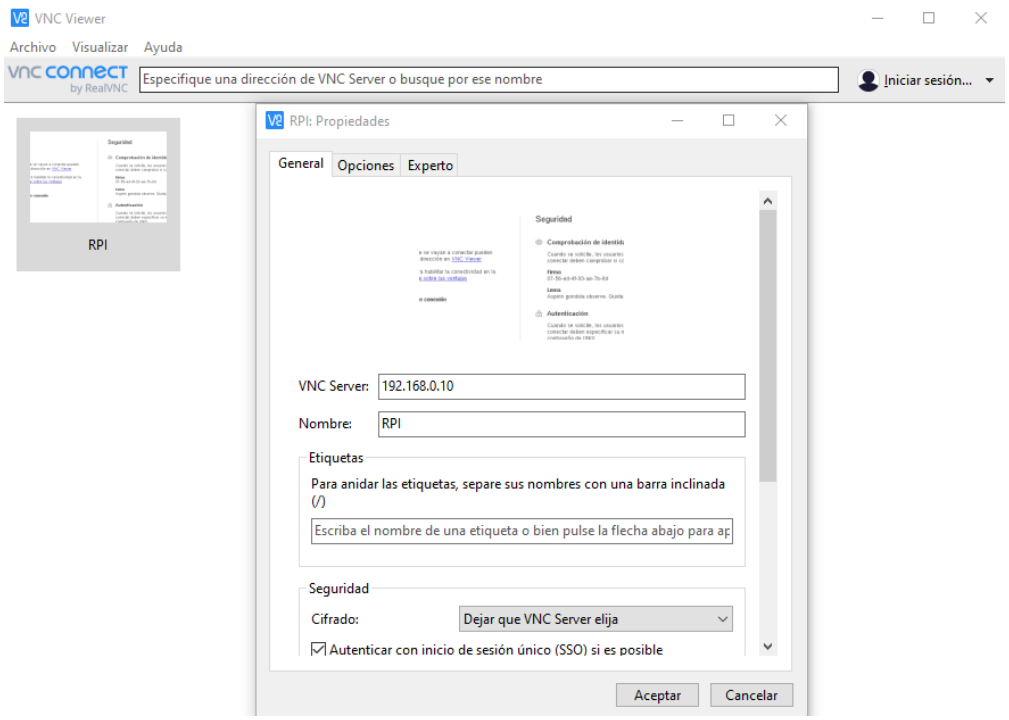


Figura 30. VNC en Windows.

Para poder usar Putty, lo que se ha de tener activado en la RPi es el SSH (igual que activar el VNC). Se activa desde la pestaña de configuración y opciones avanzadas. Una vez activado este servicio, ya nos podemos conectar remotamente a la RPi desde el terminal Windows sobre el que estamos trabajando, instalando este software en el PC (Figura 31).

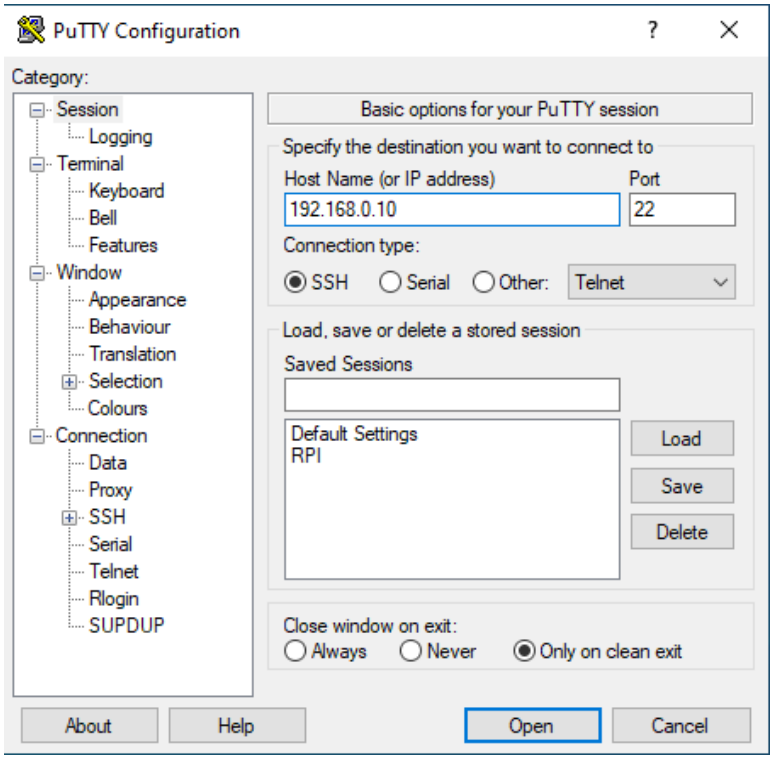


Figura 31. Ventana Putty en Windows.

3.2.5 Raspberry Pi OS

Es una evolución del anterior sistema operativo Raspbian. Se trata de una distribución del sistema operativo GNU/Linux basado en Debian. Es un SO libre. El lanzamiento de Raspbian fue en 2012 por Mike Thompson y Peter green. Está altamente optimizado para ser utilizado en la RPi. Se creó este sistema operativo en particular ya que no había versión de Debian armhf para el procesador que monta la RPi: ARMv6 (Wikipedia - La Enciclopedia Libre, 2021).

Figura 32. Raspberry Pi OS logo.

Es el sistema operativo (Figura 32) que se utilizará en la RPi que está ejecutando el SCADA y que será la herramienta principal del proyecto para la monitorización del sistema fotovoltaico construido.

3.2.6 Fritzing

Software de automatización de diseño electrónico. Es de código libre. Se utiliza en el presente proyecto para la creación de los esquemáticos de los circuitos eléctricos y electrónicos involucrados en el mismo.

Permite pasar desde un proyecto puramente visual, al esquema de circuito eléctrico y PCB sin apenas necesidades de cambio. En la Figura 33 se puede observar las distintas pestañas de trabajo dentro del programa.

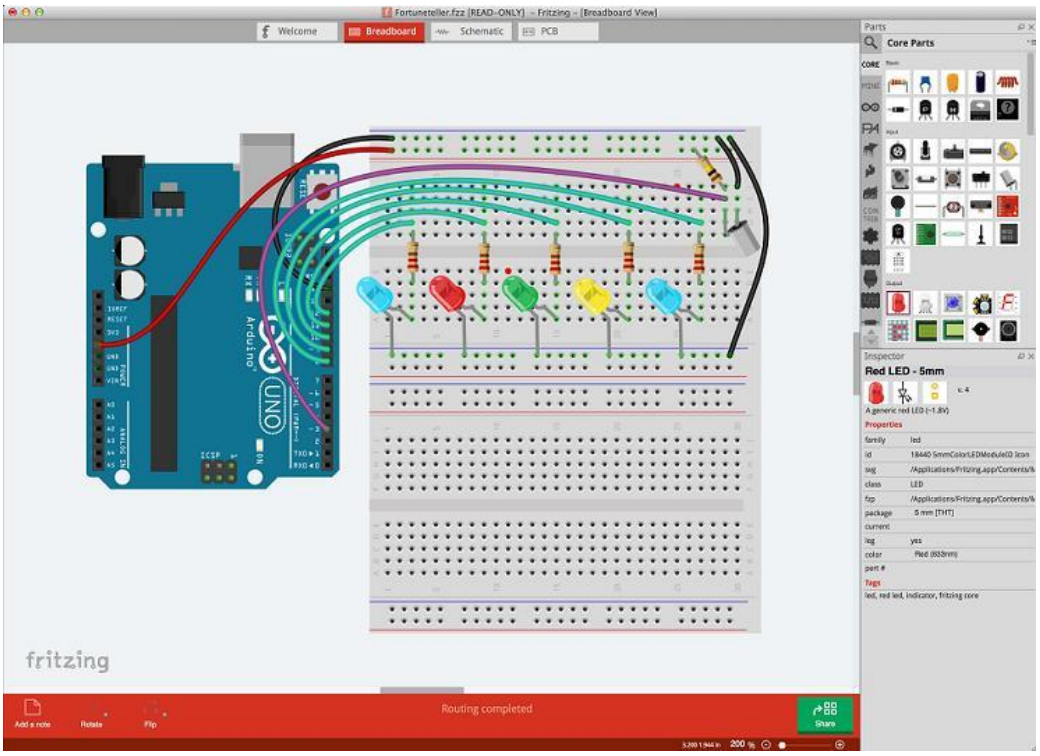


Figura 33. Espacio de trabajo Fritzing (Gobierno de Canarias, 2015).

3.3 Espacio de trabajo. Instalación completa.

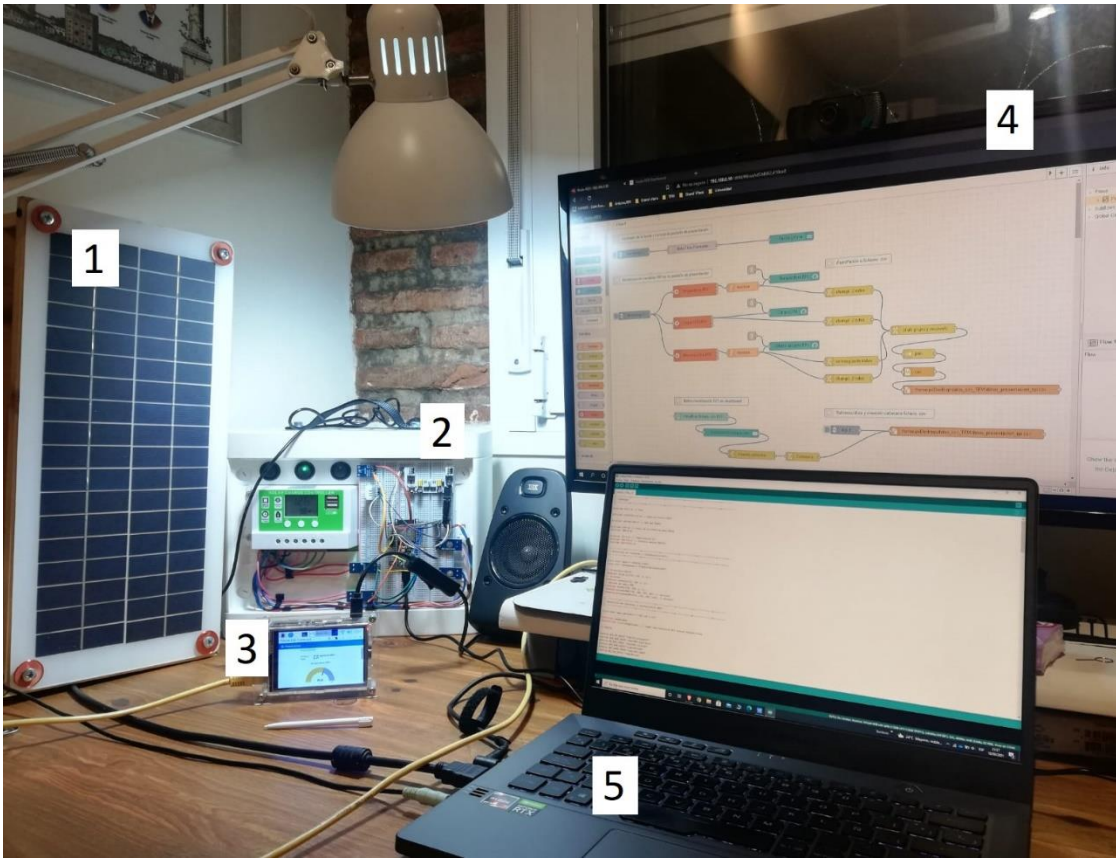


Figura 34. Espacio de trabajo físico.

En la Figura 34 se presenta el espacio de trabajo físico sobre el que se ha realizado el proyecto. Se puede apreciar el panel solar (1), la caja que incluye los elementos de control, monitorización y almacenamiento de la instalación fotovoltaica (2), la Raspberry Pi con pantalla con el sistema SCADA ejecutándose en ella (3), el monitor auxiliar del PC Windows con la ventana de programación visual del SCADA Node-RED (4) y el monitor del PC con la ventana de programación de código para el ESP32 Arduino IDE (5).

En las siguientes subsecciones se irá exponiendo cada uno de los elementos que componen dicho espacio de trabajo, así como la descripción de los códigos de programación diseñados para la ejecución del proyecto.

3.3.1 Instalación fotovoltaica y elementos de monitorización

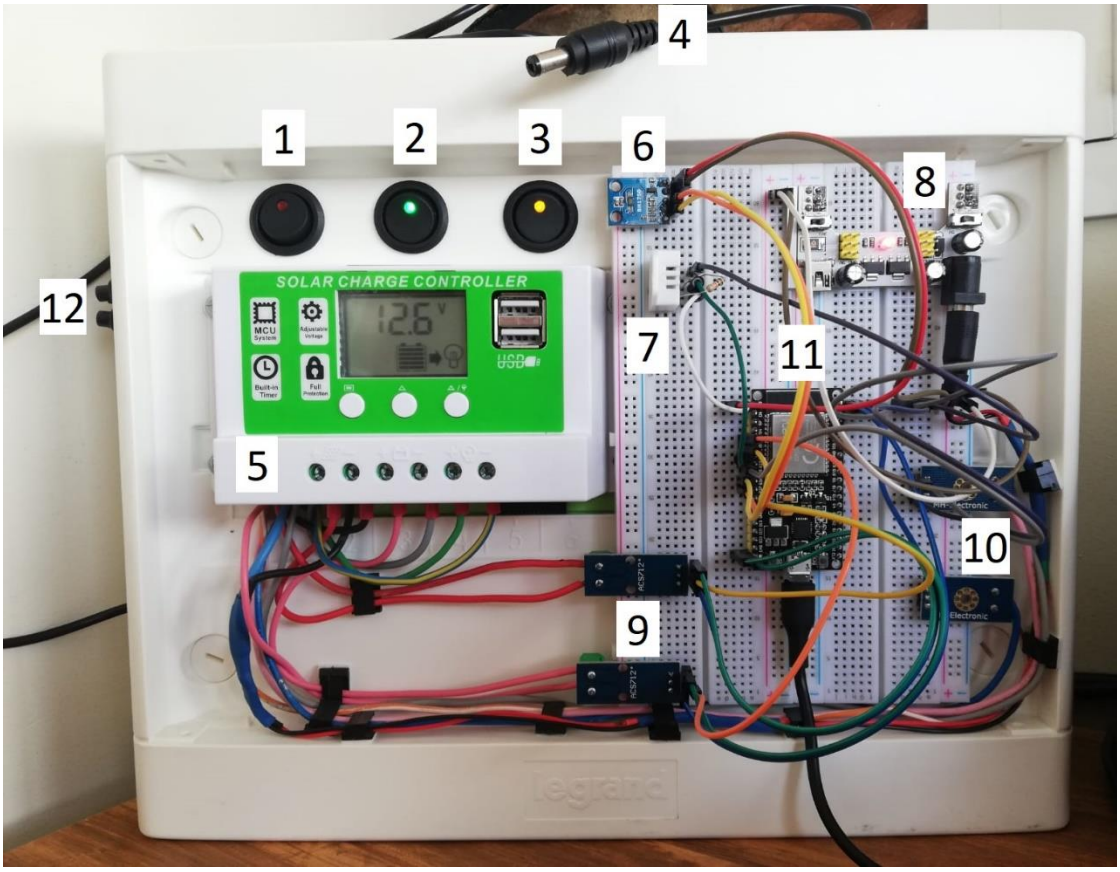


Figura 35. Caja de monitorización - parte delantera.

Todos los elementos de monitorización se han colocados en una caja de contactores térmicos adaptada especialmente para el presente proyecto (Figura 35). Ésta consta de:

1. Interruptor de control de carga
2. Interruptor de activación del controlador (apertura del circuito de la batería)
3. Interruptor de activación de la salida (alimentación de las cargas)
4. Cable de conexión a la placa solar para la alimentación de las cargas asociadas y carga de la batería de almacenamiento
5. Controlador PWM con todos sus cables asociados
6. Sensor BH1750
7. Sensor DHT22
8. Placa reguladora de tensión y alimentación a la breadboard, 3,3 y 5 V
9. Sensores de corriente ACS712
10. Sensores de tensión FZ0430 (solo se utiliza uno de ellos debido a un problema de comunicación de tierras – más explicado en el desarrollo del documento)
11. Placa de desarrollo ESP32 con cable de alimentación micro-USB
12. Terminales de carga de batería externo. Están conectados directamente a la batería para realizar una carga de apoyo en caso de que la disponibilidad de los paneles solares no haya sido suficiente para cargar la batería. Se puede conectar cualquier cargador de baterías de plomo AGM de 12 V.

Todos los sensores, placa de alimentación y placa de desarrollo ESP32 han sido colocados sobre dos breadboards de 830 puntos ancladas entre sí y adheridas a la caja de monitorización mediante la cinta autoadhesiva incorporada en las mismas.

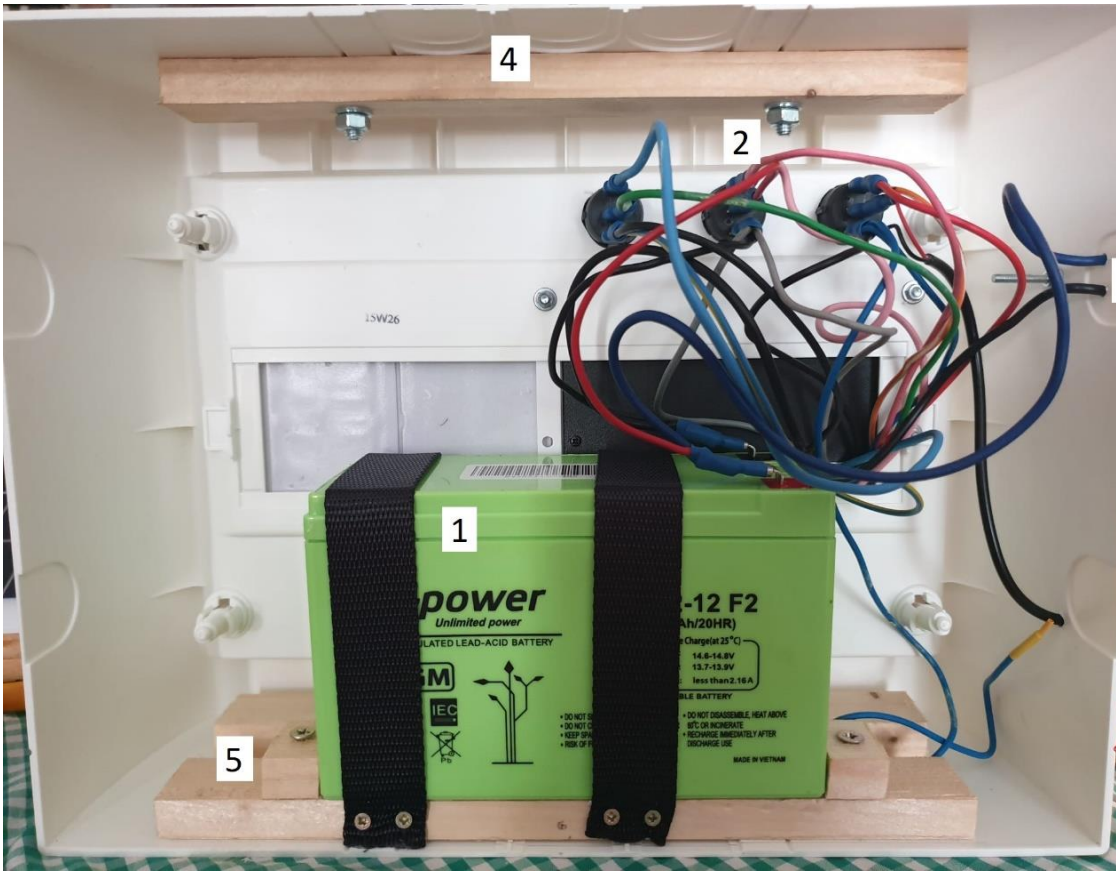


Figura 36. Caja de monitorización - parte trasera.

En la Figura 36 se puede apreciar la parte trasera de la caja de monitorización. En ésta se ha colocado la batería (1) sobre un soporte reforzado para evitar dañar la caja con el peso de esta y para evitar así mismo el movimiento de la batería (reforzado con cinchas de sujeción) (5). Se han realizado todas las conexiones de los interruptores y sensores de tensión (en paralelo) y corriente (en serie) (2). El número (3) es la salida de los cables de carga externa de la batería, conectada directamente a los bornes de ésta. (4) es un listón transversal de refuerzo para el mango de transporte de la caja.

3.3.2 Descripción del montaje de la pantalla y carcasa de la RPi

Finalmente, en la parte de hardware, cabe destacar el resultado final del montaje de la pantalla táctil y la carcasa acrílica sobre la RPi (Figura 37). Se han tenido que adaptar algunos soportes y tornillos del ventilador y carcasa para ajustarlos a las necesidades de la RPi.

La pantalla se instala directamente sobre los pines GPIO de la Raspberry Pi, sobre la que previamente se ha puesto el ventilador de refrigeración.

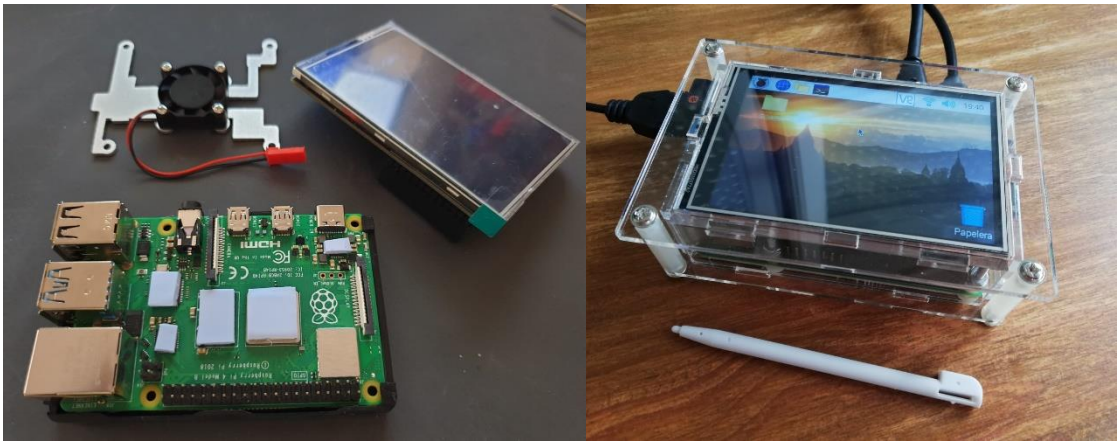


Figura 37. RPi con pantalla y carcasa acrílica.

Para configurar la pantalla, se han de introducir los siguientes comandos:

sudo rm -rf LCD-show

git clone https://github.com/goodtft/LCD-show.git

chmod -R 755 LCD-show

cd LCD-show/

sudo ./MHS35-show

Es importante que sean ejecutados de forma remota a través de SSH sin ninguna pantalla conectada por HDMI, o si, en su defecto, se quiere realizar con un monitor conectado, inmediatamente después de introducir el último comando se ha de desconectar el monitor que esté conectado por HDMI. En caso contrario no se mostraría nada en ninguna de las dos ya que entrarían en conflicto.

Si se quisiera volver a cambiar a un monitor externo, se ha de introducir el comando (y conectar el monitor deseado a través de las salidas HDMI de la placa Raspberry Pi):

sudo ./LCD-hdmi

4. MÉTODOS

4.1 Software:

Se procede a explicar, paso a paso, cada parte de los códigos diseñados, tanto la parte del ESP32 como la del SCADA.

4.1.1 Código ESP32 con Arduino IDE

Librerías

```
#include <WiFi.h> // WiFi

#include <PubSubClient.h> // para protocolo MQTT

#include <driver/adc.h> // ADC del ESP32

#include <DHT.h> // Carga de la librería para DHT22
#include <DHT_U.h>

#include <Wire.h> // Comunicación I2C
#include <BH1750.h> // Librería sensor BH1750
#include <BH1750FVI.h>
```

En este apartado se incluyen todas las librerías necesarias para la correcta ejecución del código. En primer lugar, se añade la librería *WiFi* para poder conectar el ESP32 a la red vía WiFi. A continuación, la librería *PubSubClient* es utilizada para el envío de mensajes a través del protocolo MQTT. La librería *adc* es ara controlar el conversor analógico digital integrado en algunos de los pines del ESP32. Las librerías *DHT* son para gestionar los datos provenientes del sensor de humedad y temperatura DHT22. Finalmente, las librerías *Wire*, *BH1750*, *BH1750FVI* son las encargadas de obtener y gestionar los datos del sensor de iluminancia BH1750.

Definición de variables y configuración WiFi

```
const char* ssid = "JAZZTEL_84E5";
const char* contrasena = "36CA58A7E249A386858F";

// IP estática ESP32
IPAddress local_IP(192, 168, 0, 11);
// IP router
IPAddress gateway(192, 168, 0, 1);
// Máscara de red y DNS
IPAddress subnet(255, 255, 0, 0);
IPAddress primaryDNS(212, 166, 211, 23); // opcional
IPAddress secondaryDNS(212, 166, 132, 104); // opcional
```

En este extracto de código se definen las características de la red WiFi a la que se conectarán los dispositivos. Es la proporcionada por el rúter descrito en la sección 'materiales'. Se define como una cadena de caracteres *const char** (techtutorialsx, 2020).

Es también muy importante definir una IP estática para el ESP32, ya que debemos conocer exactamente la dirección IP en todo momento para poder referirnos a él desde otras aplicaciones (Tutorials, 2018). Esto último se define con la sintaxis propia de la librería *WiFi*.

Definición de variables y configuración MQTT

```
const char* mqtt_servidor = "192.168.0.10";

WiFiClient espClient;
PubSubClient client(espClient); // Crear una instancia del
ciente PubSubClient

// TOPICS:

#define ACS_PV_TOPIC "tfm/PV/corriente"
#define ACS_BAT_TOPIC "tfm/BAT/corriente"
#define FZ_BAT_TOPIC "tfm/BAT/voltaje"
#define DHT_HUM_TOPIC "tfm/DHT/hum"
#define DHT_TEMP_TOPIC "tfm/DHT/temp"
#define BH_LUX_TOPIC "tfm/BH/lux"

long lastMsg = 0;
char msg_1[20];
char msg_2[20];
char msg_3[20];
char msg_4[20];
char msg_5[20];
char msg_6[20];

void receivedCallback(char* topic, byte* payload, unsigned int
length) {
  Serial.print("Message received: ");
  Serial.print(topic);

  Serial.print("payload: ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
   }
   Serial.println();
}

void mqttconnect() {
  // Loop hasta que se reconecta
  while (!client.connected()) {
    Serial.print("MQTT connecting...");
    // ID del cliente
    String clientId = "ESP32Client";
    // Conectarse ahora
    if (client.connect(clientId.c_str())) {
      Serial.println("conectado");
    } else {
      Serial.println("fallido, código de estado = ");
      Serial.print(client.state());
      Serial.println("Intentando de nuevo en 5 segundos");
      // Esperar 5 segundos antes de volver a intentarlo
      delay (5000);
    }
  }
```

```
}
```

En esta parte se configura el ESP32 para la transmisión de los mensajes (datos de los sensores) a través del protocolo MQTT. En primera instancia se define una variable de caracteres con la dirección IP del servidor central, esto es, del dispositivo en el que se está ejecutando el bróker Mosquitto. En este caso es la dirección IP de la Raspberry Pi (IoT Sharing.com, 2017).

A continuación, se crea una instancia de cliente, de nuevo, con la sintaxis propia de la librería *PubSubClient*. Después se definen los 'topics' MQTT a los que habrá que suscribirse desde el sistema SCADA diseñado. Se definen 6 'topics' en total, uno para cada tipo de medida que nos proporcionan los sensores.

En el siguiente párrafo se predefinen variables *msg* que serán usadas posteriormente para la publicación de los mensajes que serán enviados por MQTT.

La sección *void receivedCallBack* define una función que se utiliza para, en el caso que se desee, imprimir los mensajes publicados en el monitor serie, podrá ser llamada posteriormente. Finalmente, la sección *void mqttconnect* define otra función en la que se establece la cadena de mensajes que se publican mientras se realiza la conexión con el bróker MQTT. Se intenta conectar varias veces, con un lapso de 5 segundos entre intentos. Cuando se conecta se obtiene un mensaje 'conectado'. Se invocará en otra parte del código (Tutorials, 2018).

<u>Definición variables sensor ACS712 30A – analógico</u>

```
const int analogIn_ACS_PV = 35; // Pin panel fotovoltaico ACS
712
const int analogIn_ACS_BAT = 34; // Pin batería ACS 712
int mVperAmp = 66; // mV por amperio - característica del sensor
int SensorValue_ACS_PV = 0;
int SensorValue_ACS_BAT = 0;
int ACSoffset = 2538; //mV - característica del sensor.
Teóricamente es 2500, pero midiendo con el ESP32, lo que
obtenemos de media para valores de 0 (sin carga) es ese valor.
double Voltage_ACS_PV = 0;
double Voltage_ACS_BAT = 0;
double AmpsPV = 0;
double AmpsBAT = 0;
```

Se definen variables enteras con el número del pin del ESP32 a los que están conectados los sensores. A continuación, se definen los mV/A, característica del sensor en cuestión considerado, en ese caso es el valor particular para el ACS712 de 30 A. En las dos siguientes líneas, las variables enteras definidas *SensorValue* son los valores que recibe el ESP32 del sensor, no son más que un valor dentro del ADC de 12-bits, esto es, un valor entre 0 y 4095 – esto es común para todos los sensores analógicos empleados. Posteriormente deberá ser convertido a tensión mediante un cálculo matemático, utilizando los valores de referencia aquí definidos.

Después, se define la tensión de referencia cuando no circula corriente por el sensor ACS712, teóricamente son 2500 mV, pero calibrándolo según valores tomados

experimentalmente (con un multímetro externo[17]). Se predefinen las variables de tensión y corriente, en formato *double*, tanto de los sensores conectados en el panel como los conectados en el circuito de la batería.

Definición variables sensor FZ0430 – analógico[18]

```
const int analogIn_FZ_BAT = 33; // Pin Batería - FZ0430
int SensorValue_FZ_PV = 0;
int SensorValue_FZ_BAT = 0;
double Voltage_FZ_PV = 0;
double Voltage_FZ_BAT = 0;
```

Ídem apartado anterior, pero con variables referentes a los sensores de tensión. Posteriormente en el apartado 'loop' se realizarán los cálculos pertinentes para cambiar de valor del ADC a valor de tensión.

Definición variables sensor DHT22 – digital

```
#define DHTTYPE DHT22 // Modelo de sensor
#define DHTPIN 32 // Pin sensor T y pp DHT22
DHT dht(DHTPIN, DHTTYPE, 22);
double Hum = 0;
double Temp = 0;
```

En este caso, la configuración del sensor en la parte de definición del código es algo distinta a los sensores ya descritos debido a que se trata de un sensor digital. Se utiliza la sintaxis propia de la librería DHT. Se ha de definir el tipo de sensor y el pin en el que está conectado del ESP32. Además, se predefinen las variables, tipo *double*, en las que se almacenarán temporalmente la medición de la temperatura y la humedad.

Definición de variables sensor BH1750 – digital[19]

```
const int digitalIN_BH_SDA = 26; // Pin correspondiente
conectado al pin SDA del sensor
const int digitalIN_BH_SCL = 27; // Pin correspondiente
conectado al pin SCL del sensor

BH1750 luxometro;
```

Consideraciones similares en esta subsección a las del sensor digital anterior. Se definen los pines a los que está conectado (dos tipos, SDA y SCL, puertos con configuración especial del ESP32).

---

[17] Multímetro marca Tacklife modelo DM01M. Se han tomado una serie de valores que da el sensor cuando esta vacío, esto es, no circula corriente por él, se ha promediado y sacado el valor de mV que se ajusta a la realidad.

[18] Inicialmente había 2 sensores de este tipo, uno que medía la tensión del panel cuando se conectaba y otro que medía la tensión de la batería. Debido al tipo de conexionado realizado y para evitar complicaciones mayores se optó por eliminar el conectado a los paneles fotovoltaicos porque las tensiones se comunicaban a través de la tierra común del ESP32

[19] Este sensor, a pesar de estar conectados en los pines que tienen también funcionalidad de ADC (ADC #2), no hay problema ya que este sensor no hace uso del ADC, por lo que no habrá problemas con él cuando se active la funcionalidad WiFi.

SETUP

```
void setup() {

  // Baudios
  Serial.begin(9600); // Iniciación comunicación serie

  // WiFI

  WiFi.begin(ssid, contrasena);

  if (!WiFi.config(local_IP, gateway, subnet, primaryDNS,
secondaryDNS)){
    Serial.println("STA configuración fallida");
  }

  while (WiFi.status() != WL_CONNECTED) {
    delay (500);
    Serial.println("Conectando a WiFi...");
  }

  Serial.println("Conectado a la red WiFi");

  Serial.println("");
  Serial.println("Dirección IP: ");
  Serial.println(WiFi.localIP());
  Serial.println("");

  // Configuración de la IP del servidor MQTT y el puerto
  client.setServer(mqtt_servidor, 1883);
  client.setCallback(receivedCallback); // Esta función
"receivedCallback" será invocada cuando reciba el topic
subscrito

  // Configuración del ADC del ESP32:
  analogReadResolution(12); // 12 bits
  analogSetAttenuation(ADC_11db);

  // Definición tipo de PIN
  pinMode(analogIn_ACS_PV, INPUT);
  pinMode(analogIn_ACS_BAT, INPUT);
  // pinMode(analogIn_FZ_PV, INPUT);
  pinMode(analogIn_FZ_BAT, INPUT);

  // Inicialización sensores digitales
  dht.begin(); // Se inicia la comunicación con el DHT22

  Wire.begin(digitalIN_BH_SDA, digitalIN_BH_SCL); // Para
iniciar la comunicación I2C - se configuran los pines del sensor
(en caso contrario no da valores y se obtiene mensaje de
advertencia)
  luxometro.begin(); // Inicialización sensor BH1750 -
resolución por defecto

}
```

La parte del SETUP del programa es donde se ejecutan las primeras funciones, la parte destinada a recoger la configuración.

Con el primer comando *Serial.begin* se inicia la comunicación serie a 9600 baudios. En la parte *WiFi* se inicia la conexión del ESP32 a la red WiFi que se estableció en la sección de definición. Se imprimen mensajes de información en el monitor serie.

A continuación, se configura la IP del servidor MQTT (llamando a la variable IP definida en la sección de definición) y el puerto por el que se va a realizar la comunicación, en este caso el 1883. Se invoca a la función definida anteriormente llamada *receivedCallBack*.

Después se definen los tipos de pines de los sensores de tensión y corriente como de entrada (*INPUT*). Se configura el ADC para que trabaje a 12-bits (el máximo de resolución). Se inician los sensores digitales con los comandos *.begin.* El sensor BH1750 necesita la inicialización, además, de la comunicación I2C.

LOOP

```
void loop() {

// Si el cliente estaba desconectado, intentar conectarlo de
nuevo
  if (!client.connected()) {
    mqttconnect();
  }
// La siguiente función escuhará a los "incoming subscribed
topic-process-invoke receivedCallback"
  client.loop();
```

Se inicia la parte del bucle. Es la parte del código que se ejecuta un número infinito de veces. Al iniciarse el ESP32, una vez cargado el código, se ejecuta el código del setup y luego entra al loop, parte del código que se ejecuta de manera indefinida hasta que se apague o reinicie el microcontrolador (PanamaHitek, 2015). Todo lo que se indica a continuación está dentro de esta parte 'loop' del código – se ha separado con propósitos explicativos.

Loop – Sensores de corriente ACS712 1 & 2.

```
SensorValue_ACS_PV = analogRead(analogIn_ACS_PV);
  Voltage_ACS_PV = (SensorValue_ACS_PV * 3418) / 4095; // mV -
el voltaje de referencia teórico es 3300 mV. Ese valor ha sido
calculado mediante las discrepancias reales con el valor
teórico.
  AmpsPV = ((Voltage_ACS_PV - ACSoffset) / mVperAmp); // amps
  delay(1000);

  Serial.print("Amps PV (A): "); // Se muestra en el monitor
serie el valor de la corriente a través de los terminales del
controlador PWM conectados a batería.
  Serial.println(AmpsPV);
  delay (500);

  snprintf (msg_1, 20, "%1f", AmpsPV);
  // Publicar el mensaje
  client.publish(ACS_PV_TOPIC, msg_1);
  delay(1000);


  SensorValue_ACS_BAT = analogRead(analogIn_ACS_BAT);
```

```
  Voltage_ACS_BAT = (SensorValue_ACS_BAT * 3418) / 4095; // mV -
el voltaje de referencia teórico es 3300 mV. Ese valor ha sido
calculado mediante las discrepancias reales con el valor
teórico.
  AmpsBAT = ((Voltage_ACS_BAT - ACSoffset) / mVperAmp); // amps
  delay(1000);

  Serial.print("Amps BAT (A): "); // Se muestra en el monitor el
valor de la corriente a través de los terminales de salida del
controlador PWM.
  Serial.println(AmpsBAT);
  delay (500);

  snprintf (msg_2, 20, "%1f", AmpsBAT);
  // Publicar el mensaje
  client.publish(ACS_BAT_TOPIC, msg_2);
  delay (1000);
```

En esta parte se ejecuta el código que captura los valores que da el ESP32 de los sensores y calcula, mediante un cálculo matemático con ciertos valores de referencia, el valor de la corriente real que circula por el circuito. Se inicia leyendo mediante el comando *analogRead* el valor proporcionado por el ADC del ESP32. A partir de ese valor, con el valor de referencia de la tensión en vacío, se calcula la tensión que está proporcionando el sensor mediante la ecuación (6.1):

$$Tensión\ ACS = (Valor\ ADC * 3418)\frac{1}{4095}\ mV \tag{6.1}$$

El valor de la tensión de referencia teórico (esto es, el valor máximo de la escala - 3300 mV[20]) ha sido modificado ligeramente según comprobaciones experimentales. Después, para calcular la corriente, se utiliza la ecuación (6.2):

$$Corriente\ ACS = \frac{Tensión\ ACS - Tensión\ ACS\ vacío}{mV/A} \tag{6.2}$$

Donde la tensión del ACS en vacío son los aproximadamente (ligero cambio tras calibración) 2500 mV descritos en la sección de definición del código.

Se escribe un *serial.Print* para mostrar los valores en el monitor serie y además se publica el mensaje por MQTT con *client.publish* (García, 2021). Se realiza el mismo procedimiento para ambos sensores (conectados en pines diferentes del ESP32). Se establece un *delay* de 500 ms entre la impresión en el monitor serie y la publicación y de 1000 ms entre la publicación y el cálculo del siguiente sensor y entre el cálculo y la escritura en el monitor serie.

Loop – Sensor de tensión FZ0430

```
SensorValue_FZ_BAT = analogRead(analogIn_FZ_BAT);
  Voltage_FZ_BAT = (SensorValue_FZ_BAT * 16.5) / 4095; // En
este caso el voltaje de referencia no son 25 V, lo máximo que
```

[20] De nuevo, este valor se ha calculado tomando una serie de valores que provienen del ADC del ESP32, del pin al que está conectado el sensor y datos tomados con el multímetro. Mediante los valores promediados y la ecuación (6.1) se ha calculado el valor que se ajusta a la experiencia.

```
podrá medir son 16,5 debido a que el pin del ESP32 trabaja como
máximo a 3,3 V y no a 5 V. (4095 se corresponderá con 3,3 V).
  delay (1000);

  Serial.print("Voltaje BAT (V): ");
  Serial.println(Voltage_FZ_BAT);
  delay (500);

  snprintf (msg_3, 20, "%1f", Voltage_FZ_BAT);
  // Publicar el mensaje
  client.publish(FZ_BAT_TOPIC, msg_3);
  delay (1000);
```

Procedimiento análogo al caso anterior. Se lee el valor del pin y se realiza un cálculo para averiguar el valor de la tensión medida. A continuación, se imprime el valor en el monitor serie y se publica vía MQTT. La ecuación para el cálculo es la número (6.3):

$$Tensión\ FZ = (Valor\ ADC * 16{,}5)\frac{1}{4095}\ mV \tag{6.3}$$

Donde 16,5 V es la tensión de referencia del sensor y 4095 el número de canales del conversor ADC.

Nota importante: en las ecuaciones (6.1) y (6.3) el orden de las operaciones es esencial. Primero ha de ser realizada la multiplicación y después la división. Aunque sean operaciones matemáticas normalmente conmutables, en este caso no, ya que si se divide el valor del ADC entre 4095 se obtiene un valor muy pequeño que el programa no es capaz de procesar y lo pone como 0, obteniendo así unos valores de tensión iguales a 0, esto es, un resultado erróneo.

Loop- Sensor de humedad y temperatura DHT22

```
 Hum = dht.readHumidity(); // Lectura de la humedad. Asignación
del valor a la variable 'Hum'.
  Temp = dht.readTemperature(); // Lectura de la temperartura.
Asignación del valor a la variable 'Temp'
  delay (1000);

  Serial.print("Humedad (%): ");
  Serial.println(Hum);
  delay (500);

  snprintf (msg_4, 20, "%1f", Hum);
  // Publicar el mensaje
  client.publish(DHT_HUM_TOPIC, msg_4);
  delay (1000);

  Serial.print("Temperatura (oC): ");
  Serial.println(Temp);
  delay (500);

  snprintf (msg_5, 20, "%1f", Temp);
  // Publicar el mensaje
  client.publish(DHT_TEMP_TOPIC, msg_5);
  delay (1000);
```

En este caso la lectura del sensor no es analógica, sino que se realiza de manera digital, con los comandos *dht.readHumidity* y *dht.readTemperature*. Obtenemos directamente el valor en unidades finales (% y ºC respectivamente) sin necesidad de realizar ningún cálculo adicional. Análogamente a los casos anteriores, se imprime el valor en el monitor serie y se publica vía MQTT en el 'topic' correspondiente.

Loop – Sensor de iluminancia BH1750

```
uint16_t lux = luxometro.readLightLevel(); // Lectura del sensor
  delay (1000);

  Serial.print(F("Iluminancia (lx): "));
  Serial.println(lux);
  delay (500);

  double lux_pub = lux; // Cambiamos el tipo de variable para
publicarla

  snprintf (msg_6, 20, "%1f", lux_pub);
  // Publicar el mensaje
  client.publish(BH_LUX_TOPIC, msg_6);
  delay (1000);


}
```

Finalmente se han de leer los valores de iluminancia del sensor BH1750. Se define una unidad de tipo *uint16_t* en la que se guarda el valor de la lectura digital del sensor.

Es importante cambiar el tipo de unidad, una vez leída de *uint16_t* a *double* para que se publique correctamente, si no, no llega nada de información al SCADA y se superpone en la gráfica correspondiente el último valor publicado[21], correspondiente a otro sensor (Random Nerd Tutorials, 2020).

Nota importante: todos los sensores que necesitan de ADC, se han conectado en los pines del ADC #1 del ESP32. Hay dos conversores ADC, el #1 y el #2. Es importante que se hayan conectado a él ya que el ADC #2 no funciona cuando está activada la funcionalidad WiFi y, por tanto, no podríamos medir los valores de los sensores analógicos conectados a él.

Una buena guía para la ejecución de esta parte del proyecto ha sido: (Aprendiendo Arduino, 2020).

En el Anexo B - Figura 55 y Figura 56, se muestra un ejemplo de salida por el monitor serie. El código completo del ESP32 aquí desglosado se incluye completo, sin cortes, en el anexo D, para su facilidad de copiado y pegado en caso de querer reproducir el mismo código.

[21] Para la escritura del código en este documento, en el formato en el que aparece en el lector, se ha de copiar desde el Arduino IDE como HTML, pegar el código en la página https://www.w3schools.com/html/tryit.asp?filename=tryhtml_formatting, darle a 'run' y copiar lo que ahí sale en el documento Word.

4.1.2 SCADA con Node-RED

En esta sección se expondrán los diagramas de bloques realizados en Node-RED y una explicación de la configuración de cada apartado. Los diagramas de bloques se crean y configuran en el Node-RED Developer, la parte de programación visual. Para mostrar los datos en lo que es propiamente el SCADA se utiliza un *addon* particular de Node-RED llamado *Dashboard*, que tiene nodos específicos asociados (principalmente de muestra de gráficos y texto). La muestra del *Dashboard* se recogen en la sección resultados del presente proyecto.

Información interna Raspberry Pi

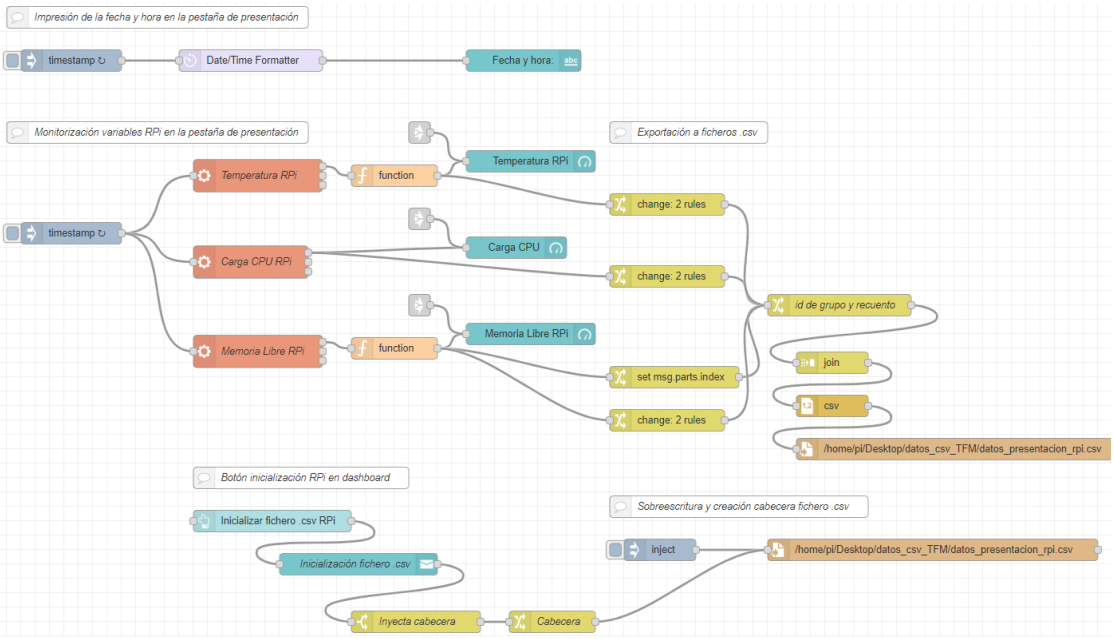


Figura 38. SCADA - Datos RPi

Con estos bloques (Figura 38) se define la pantalla de *Presentación* en el Node-RED Dashboard. En primer lugar, en 'Impresión de la fecha y hora en la pantalla presentación' se imprime la fecha y hora con un nodo *inject* configurado como *timestamp* (en formato de tiempo 'epoch'). Para pasarlo a un formato más conocido y directamente reconocible se utiliza un nodo llamado *Date/Time Formatter* que hay que instalar aparte, desde la sección de gestión de paquetes[22]. Eso se conecta con un bloque de formato *Texto* que se muestra en el *Dashboard* (Tutorials, 2017) (Aprendiendo Arduino, 2020).

En los bloques siguientes, en la sección 'Monitorización variables RPi en la pestaña presentación' se recaban datos internos de la RPi: temperatura, carga del procesador y memoria libre, en primer lugar, a través de un nodo *exec*[23]. La carga del procesador se obtiene directamente, los otros dos pasan a través de una función de acondicionamiento para mostrarse en las unidades adecuadas. Esos datos se conectan con bloques del *Dashboard*, de tipo indicador de aguja (*Gauge*). Los comandos que aparecen en el nodo *exec* son los siguientes:

[22] Se instalan las librerías *Node-Red-Contrib_moment* y *Contrib-Date*, dentro de Node-RED en Manage/Palette/Install.

[23] Ejecuta uno o varios comandos de consola en la Raspberry Pi en la que se está ejecutando Node-RED.

- Temperatura RPi: *vcgencmd measure_temp*
- Carga CPU RPi: *top -d 0.5 -b -n2 | grep "Cpu(s)"|tail -n 1 | awk '{print $2 + $4}'*
- Memoria libre RPi: *free | grep Mem | awk '{print $4/$2 * 100.0}'*

Las funciones para acondicionar los datos anteriores son:

- Temperatura RPi
  - *str = msg.payload*
  - *msg.payload = str.substring(5,9);*
  - *return msg;*
- Memoria libre RPi:
  - *str = msg.payload*
  - *msg.payload = parseInt(str.substring(0,str.length - 1));*
  - *return msg;*

Los bloques de *link* que aparecen en gris (justo antes de los nodos de las gráficas) redirigen a un botón en el *Dashboard* que inyecta un código vacío (del tipo []) para limpiar las gráficas.

El resto de los bloques, en las secciones 'Exportación a ficheros .csv', 'Botón inicialización RPi en dashboard' y 'Sobreescritura y creación cabecera fichero .csv', van enfocados a guardar los datos en un fichero csv. Se utilizan los bloques *change, join, csv* y *file* en la primera sección mencionada.

- El nodo *change* lo que hace es asignar un índice al dato que proviene del nodo al que está conectado para después poder ordenarlos para pasarlos al fichero .csv. Además, en ese mismo nodo se establece otra regla (de ahí que aparezca '2 rules') que convierte el dato tipo cadena de caracteres a formato número.
- El otro nodo *change* que aparece con título 'id de grupo y recuento', tiene de nuevo 2 reglas, una que asigna al conjunto de datos que se quieren que vayan en un fichero .csv un número de grupo y además cuenta el número de datos (columnas) que irán en el fichero.
- El nodo *join* unifica automáticamente todos los datos provenientes de los distintos nodos.
- El nodo *csv* da el formato a los datos, de tipo separados por comas
- Finalmente, el nodo *file* escribe los datos con ese formato a un fichero .csv en la ruta especificada en la cabecera del nodo. Se ha de crear un directorio, local, en este caso en el escritorio de la RPi, donde se van a guardar dichos ficheros.

En la sección de inicialización lo que se establece es un botón del *Dashboard* que al pulsarlo sobreescriba el fichero .csv existente y cree uno nuevo para guardar nuevos datos. Al presionarlo se lanza una notificación que se deberá aceptar, esto es realizado con un nodo *notification*. Este nodo está conectado con un nodo *switch*, que establece que si la entrada es 'Aceptar', inyecta la cabecera que está configurada en el nodo *change* al que está conectado. Finalmente, esto se conecta a un nodo *file* para comenzar un nuevo .csv[24].

En el Anexo C se recogen las distintas ventanas de configuración de los distintos nodos mencionados, para tener una idea de cómo funciona este tipo de programación visual.

[24] El nodo *inject* que aparece en la sección 'sobreescritura y creación cabecera fichero .csv' es simplemente para comprobar el código desde Node-RED developer. Hace la función de un botón del *Dashboard*. Sirve para inyectar manualmente el fichero, sobreescribirlo y poder observarlo desde la consola de depuración del *Developer*.

Sensores de corriente

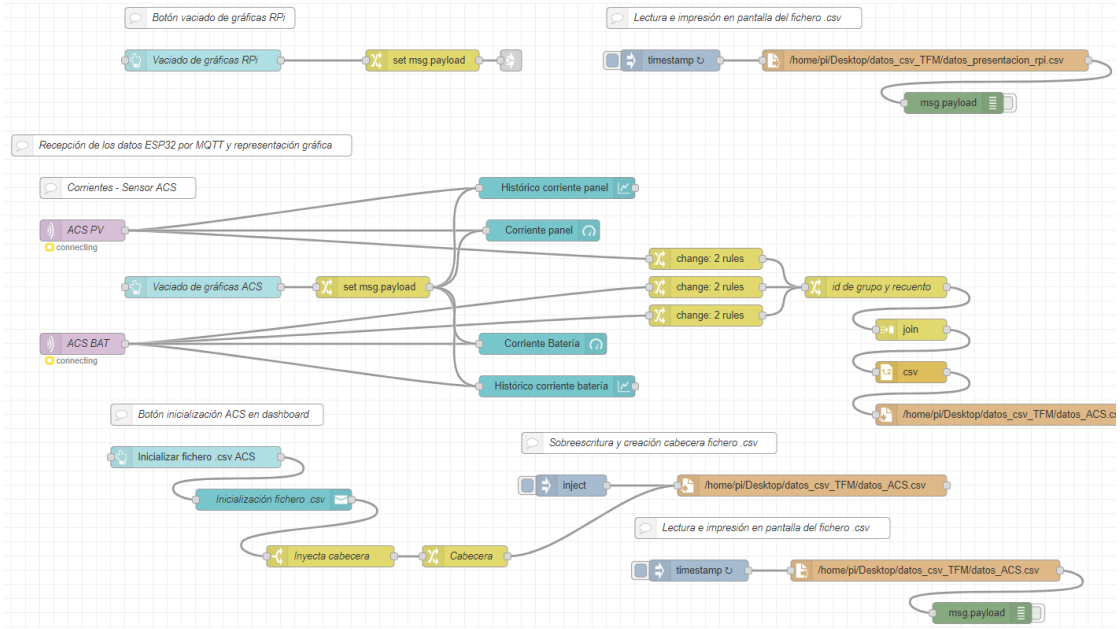


Figura 39. SCADA - Datos sensores de corriente.

En el principio (Figura 39) se pueden apreciar los nodos *link* en gris que conectan con la figura anterior para el vaciado de las gráficas correspondientes (sección 'Botón vaciado de gráficas RPi'). Se encuentra unido a un nodo de *Botón* del *Dashboard* y a otro nodo *change* que es el que proporciona el código vacío para el borrado. En la parte derecha a esta ('Lectura e impresión en pantalla del fichero .csv'), existe un nodo *inject*, tipo *timestamp*, conectado a un nodo *salida de archivo* y a un nodo *debug*. Esto es simplemente para realizar comprobaciones del código en el propio Node-RED Developer, muy útil para verificar qué se está imprimiendo realmente en el archivo .csv en la propia consola Node-RED.

La siguiente parte se trata de la 'Recepción de datos ESP32 por MQTT y representación gráfica'. Estará constituida por el conjunto de nodos destinados a la recolección de los datos provenientes del ESP32 vía MQTT, a la representación gráfica en el SCADA, junto con los botones de operación de estas y al guardado de los datos de los sensores en archivos .csv. Se repite la estructura explicada en el apartado anterior. Tan solo se introduce un nuevo nodo tipo *mqtt in*, que es el que está configurado para conectarse al bróker (IP) y a un *topic* en particular (definido en el código de programación del ESP32), dependiendo del sensor que queramos monitorizar.

La salida de ese nodo ya tiene directamente el valor del sensor proporcionado por el código del ESP32. Lo único que se hace es conectarlo con nodos tipo gráficas del *Dashboard* para mostrarlos visualmente en el SCADA diseñado. Para guardar los datos en el fichero .csv se sigue exactamente el procedimiento arriba explicado. Lo que sí hay que tener en cuenta es que debemos pasar la información proporcionada por MQTT a formato número, ya que lo que se transmite por MQTT viene con formato cadena de caracteres, para poder guardar los ficheros .csv con formato número, con valores separados por comas. Eso se realiza de nuevo con los nodos *change*.

Sensores de tensión

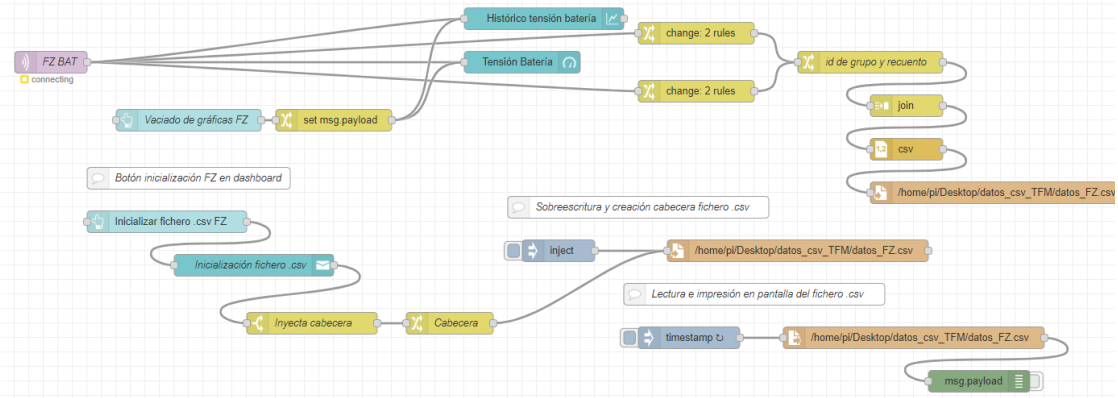


Figura 40. SCADA - Datos sensores de tensión.

Procedimiento completamente análogo a los anteriores (Figura 40).

Sensor de humedad y temperatura

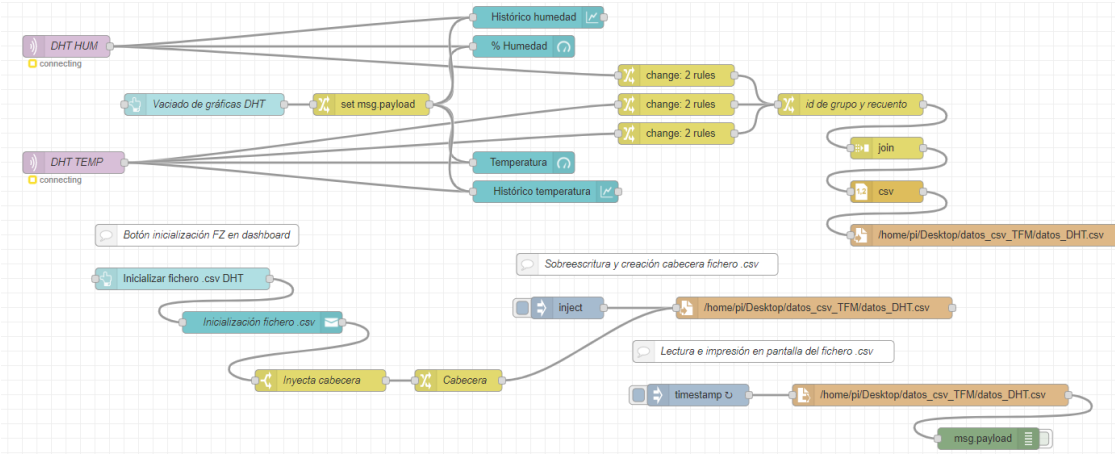


Figura 41. SCADA - Datos sensor de humedad y temperatura.

Procedimiento completamente análogo a los anteriores (Figura 41).

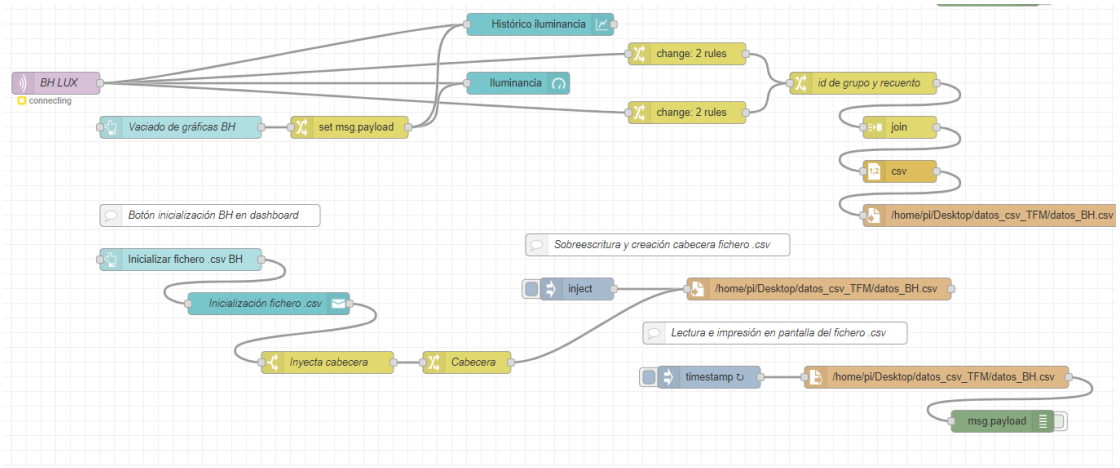


Figura 42. SCADA - Datos sensor de iluminancia

Procedimiento completamente análogo a los anteriores (Figura 42).

En el Anexo E se presenta el código para la importación del SCADA diseñado. Dentro de Node-RED Developer hay que ir al Menú/Import. Copiar y pegar dicho código y presionar el botón 'Import'. Se creará exactamente el mismo flujo de nodos aquí presentado.

5. RESULTADOS

5.1 Verificación del Código del ESP32

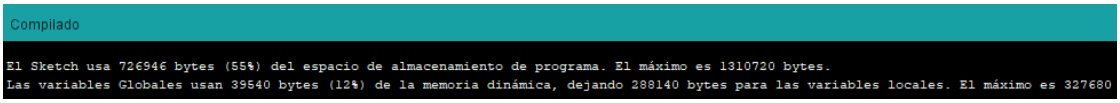


Figura 43. Verificación del código del ESP32.

Compilación del código subido al ESP32 sin errores. Para mandar el código al ESP32, hay que darle al botón 'subir' mantiendo pulsado el botón 'boot' del ESP32 mientras aparece el mensaje 'connecting' en la consola de Arduino IDE, soltarlo cuando conecte.

5.2 Node-RED Dashboard

5.2.1 Escritorio RPi

Figura 44. Escritorio Raspberry Pi.

El resultado final del escritorio de la Raspberry Pi se muestra en la Figura *44*. Tiene 3 elementos principales:

- Node-RED Developer: acceso directo que abre una pestaña de navegador al diagrama de bloques de Node-RED, la parte de programación visual. La dirección a la que apunta es *192.168.0.10:1880*. Esa dirección se corresponde con la IP donde se está ejecutando Node-RED, que en este caso es la Raspberry Pi y además, el puerto 1880.
- Node-RED Dashboard: acceso directo que abre otra pestaña de navegador a la interfaz visual del SCADA asociado al programa diseñado en el *Developer*. Este acceso directo apunta a la dirección *192.168.0.10:1880/ui*. Dirección análoga a la anterior.
- Carpeta datos_csv_TFM: donde se guardan los ficheros .csv del proyecto.

5.2.2 Pestaña Presentación Dashboard

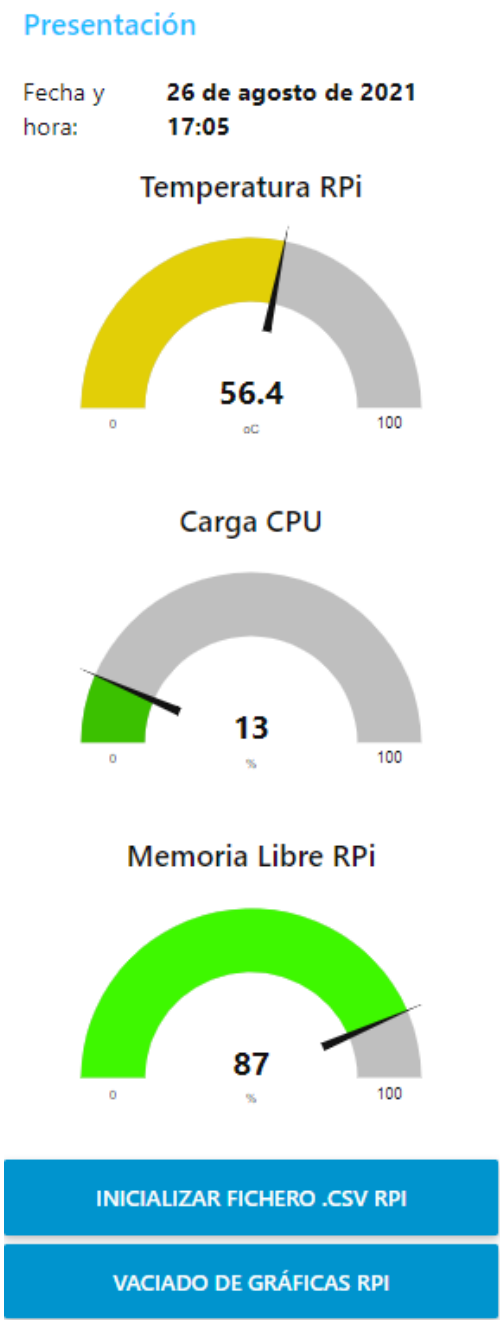


Figura 45. Pestaña presentación SCADA.

En la pestaña 'Presentación' se muestra la fecha y la hora del sistema, así como los datos relativos a la Raspberry Pi (Figura 45), para poder comprobar el estado de ésta y asegurar que está funcionando dentro de los parámetros de operación establecidos por el fabricante.

El estado que se muestra en la Figura 45 es el estado que tiene la Raspberry Pi con el SCADA Dashboard iniciado, esos son los recursos que consume. La temperatura de trabajo es algo alta, pero en concordancia con los datos de temperatura de operación dados por el fabricante, además, hay que tener en cuenta que está siendo operada en un día caluroso de verano.

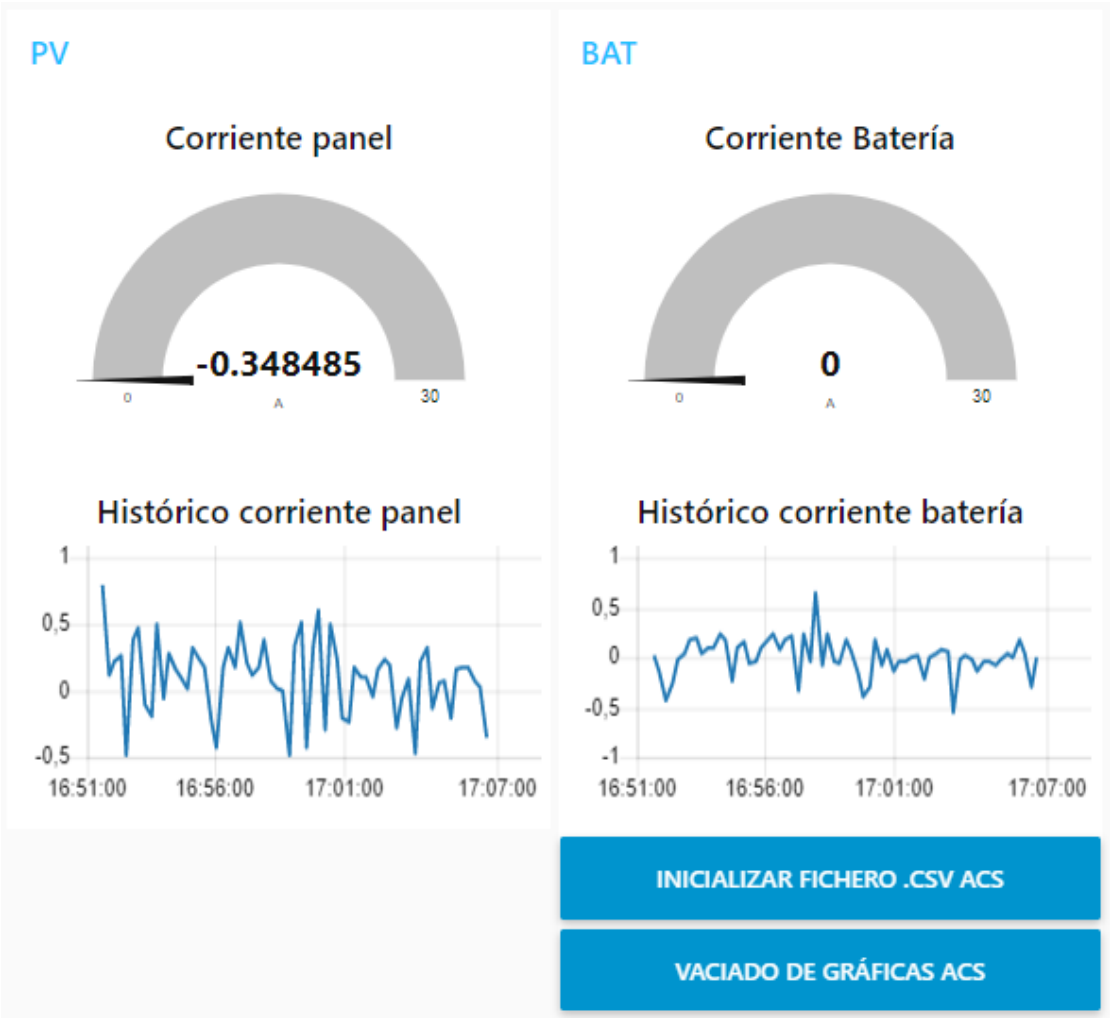


Figura 46. Pestaña ACS SCADA.

En la Figura 46 se presenta la pestaña con los datos de los sensores de corriente. En este se puede observar que las corrientes medidas son muy bajas (consumos típicos del sistema operando). En las gráficas de aguja se presentan los valores actuales, que se van actualizando cada varios segundos. En las gráficas con ejes cartesianos se presenta la evolución temporal de los valores desde que se comenzó a medir (o en su defecto, desde que se presionó el botón 'vaciado de gráficas').

El pico que se observa en la gráfica 'histórico corriente batería' se corresponde con una carga conectada momentáneamente (luz LED a 12 VDC). Todas las demás fluctuaciones que se ven son ruido. Cabe destacar que el sensor elegido ha sido el de 30 A, que tiene menos resolución para corrientes bajas.

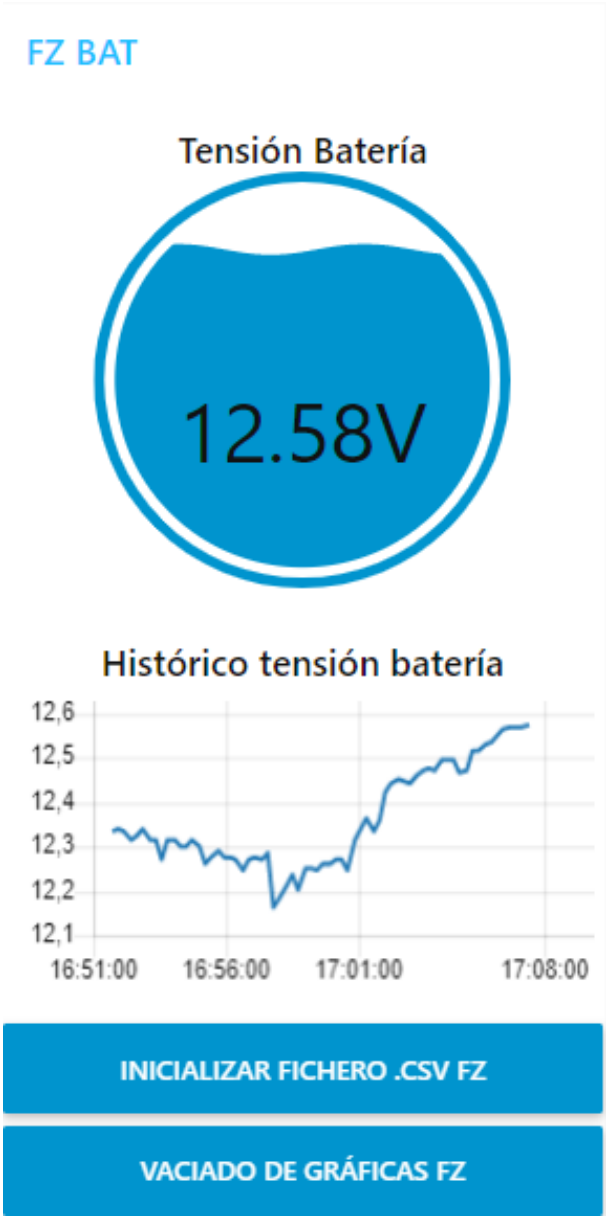


Figura 47. Pestaña FZ SCADA.

En la Figura 47 se muestra la pestaña correspondiente al sensor de tensión conectado a la batería. De nuevo, en el gráfico de burbuja se muestra la tensión de la batería en la medición actual y en el histórico la evolución temporal. Se puede observar en el histórico como al principio comienza a disminuir lentamente la carga, a medida que el sistema se conecta y pone en operación. Aproximadamente a las 17:01:00 h del gráfico, se conecta la placa solar que empieza a cargar la batería. Se puede comprobar que, a partir de ese momento, la tensión de la batería comienza a subir lentamente.

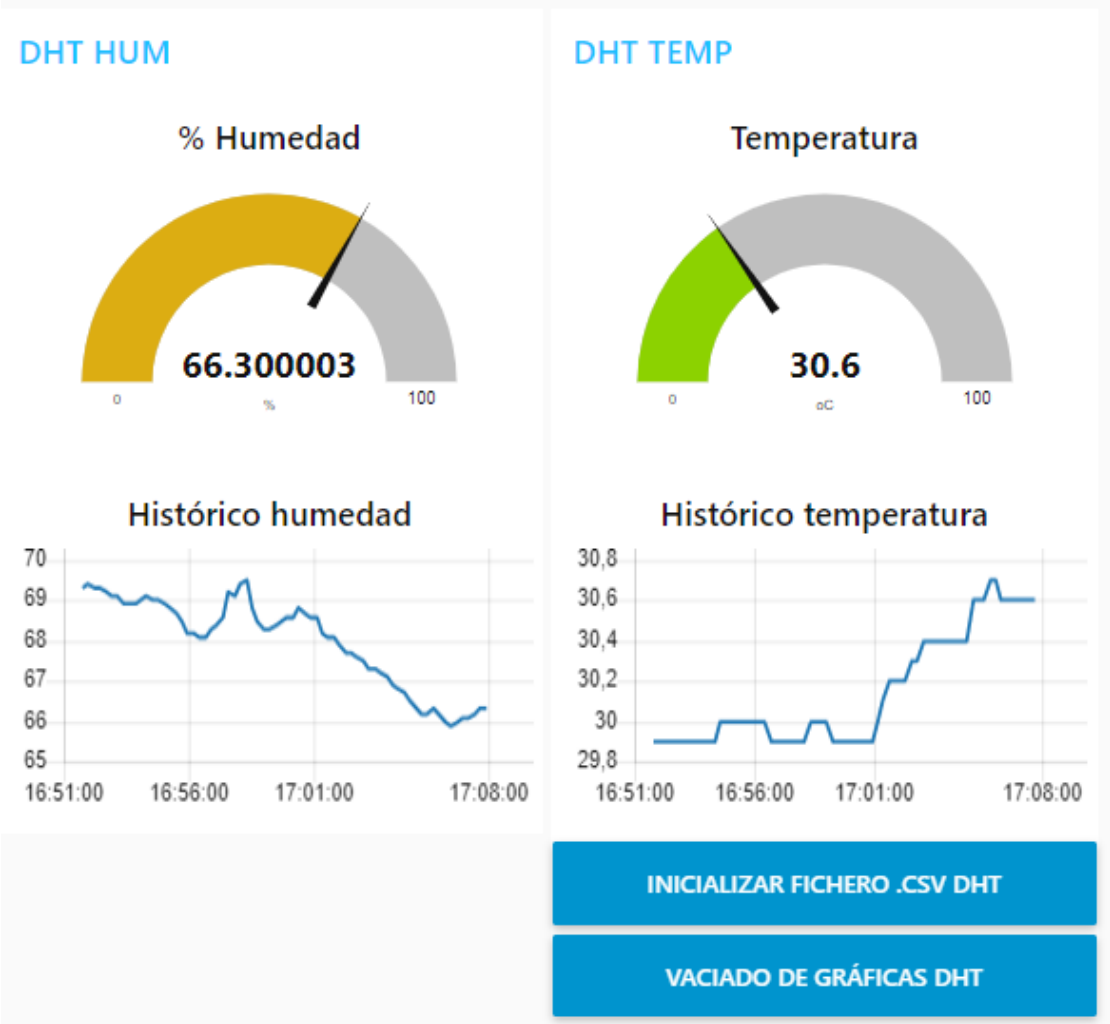


Figura 48. Pestaña DHT SCADA.

En esta Figura 48 se muestran los datos de la pestaña del SCADA correspondiente al sensor de humedad y temperatura. Se utilizan gráficos de aguja (valor actual) y de ejes cartesianos (histórico).

Se puede observar como a partir de las 17:01:00 comienza a bajar la humedad y subir la temperatura. Eso se corresponde con la apertura de una persiana y la ventana en la habitación en la que estaba conectada el sistema, para ver la influencia en la iluminancia y, además, para poner el panel fotovoltaico al sol y cargar la batería.

Figura 49. Pestaña BH SCADA.

La Figura 49 se corresponde con los datos del sensor de iluminancia del SCADA. Se pueden apreciar dos niveles de iluminancia, al principio, unos 500 lux y después unos 700-800 lux. Eso se corresponde con la apertura de la persiana y la ventana descritas en la explicación de la figura anterior. Además, se observa un pico alto de iluminancia, que ronda los 4000 lux. Se corresponde con la iluminación del sensor con la luz led conectada, técnica descrita en la Figura 46 para comprobar la corriente que circulaba por el circuito de la batería.

La estructura de pestañas del SCADA se presenta en la Figura *50*. Seleccionables mediante un desplegable.

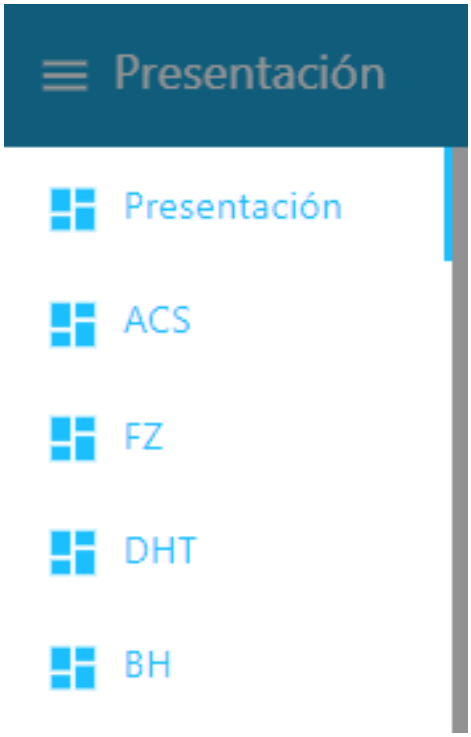
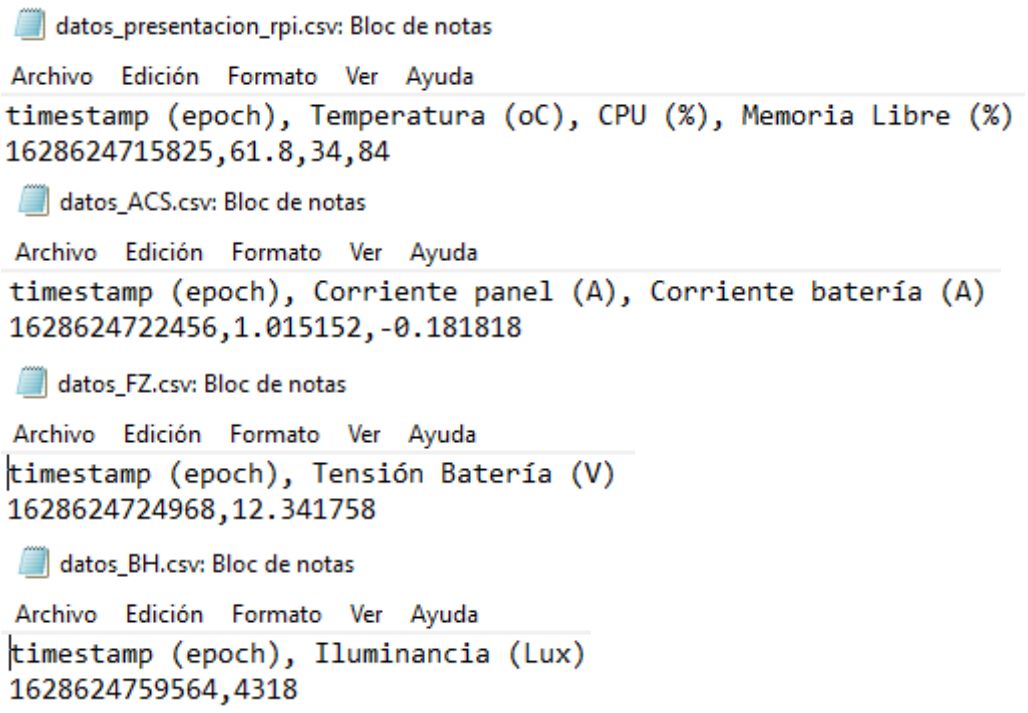


Figura 50. Estructura pestañas SCADA.

5.3 Datos .csv

Los datos recabados se guardan en el formato .csv como se muestra en la Figura 51. Se guardan con una cabecera adaptada a cada fichero de datos. Después de la cabecera, se presentan los datos separados por comas. En dicha figura se muestra una línea de ejemplo para cada fichero de datos reales tomados.



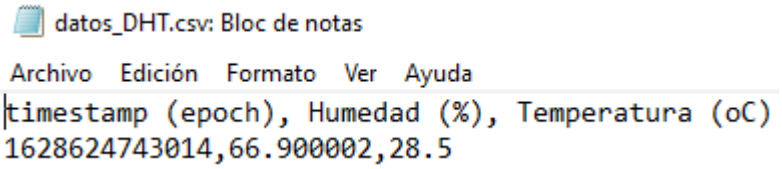


Figura 51. Estructura de datos .csv guardados por el programa SCADA definido.

Los datos se irían alojando con el mismo formato, línea por línea.

6. DISCUSIÓN Y ANÁLISIS DE LOS RESULTADOS

6.1 Resultados

Los resultados obtenidos son los ideales esperados en el comienzo del proyecto. Cabe destacar un par de consideraciones que deberían ser tenidas en cuenta para mejorar el resultado final del proyecto.

En primer lugar, como se ha observado en la Figura 46, el modelo de sensor de corriente ACS 712 elegido para este proyecto no ha sido el más acertado. Con el modelo de 30 A no se tiene buena resolución en la medida de corrientes bajas, como las que se manejan en este proyecto. Principalmente en la medida del consumo propio de la instalación para la alimentación de los sensores, controlador y ESP32, así como la corriente de carga proporcionada por el panel, que es de baja potencia. Como mejora se propone el empleo del modelo de 5 A que funcionará mejor para esta situación particular (no para proyectos de mayor potencia, en los cuales el sensor elegido sería el correcto).

Otra consideración para tener en cuenta es el funcionamiento de la Raspberry Pi para la recolección de datos. Si se quieren almacenar los datos continuamente, la Raspberry Pi debería estar conectada 24/7 al sistema. En este caso, la Raspberry Pi con pantalla. Se podría poner otra Raspberry Pi, con recursos más limitados para tan solo albergar el servidor del SCADA y del bróker y después, con otra con pantalla y/u otro dispositivo, monitorizar la instalación.

6.2 Coste del proyecto

En la Tabla 17 se presenta el coste total del proyecto, incluyendo todos los elementos necesarios para la construcción de la instalación solar. Pero el precio que realmente se quiere considerar es el que costaría el sistema de monitorización, que es el que ha sido anunciado como de bajo coste. El coste del sistema de monitorización completo (incluyendo la RPi utilizada) es de 198,17 €. Con la variación de la Raspberry Pi sugerida en la nota a pie de página 25, ese coste se reduciría hasta aproximadamente **127,18 €.**

Además, si se prescinde de ciertos elementos como la pantalla, tal y como se ha sugerido en la sección 8, para tan sólo correr el bróker y el servidor SCADA en la RPi y después conectarse con cualquier dispositivo móvil a la dirección SCADA, se podrían abaratar más los costes (esto es con el sistema específico diseñado).

Finalmente, alguna reducción adicional de los costes se podría lograr en la producción del prototipo final, donde se eliminarían cables, breadboards, sistemas de alimentación en favor de una placa PCB correctamente diseñada. Los sensores y el microcontrolador ESP32 son los menores de los costes implicados, tienen un coste extremadamente bajo – por lo que se podrían alcanzar precios aún más atractivos con las modificaciones propuestas.

Comparando estos valores con lo que pueden ser sistemas comerciales como *Monsol SCADA Solar*, *Solar-Log* y *Abora Monitor*, entre otros, se puede ver la gran ventaja del sistema diseñado frente a los sistemas comerciales existentes mencionados. Esto es aplicable para instalaciones domésticas y de automóviles de baja y media potencia en las que el coste de las otras es difícilmente justificable. No obstante, decir, que las plataformas comerciales son muy útiles y perfectamente válidas para todos los sistemas, en especial en ambientes

profesionales y/o domésticos de mayor potencia. Y en entornos en los que sea necesaria una mayor precisión, control y funcionalidades 'Off the Shelf'.

Tabla 17. Coste total del proyecto.

ELEMENTO	COSTE TOTAL (€)
Hardware	
Kit Raspberry Pi 4B 8GB	135,99[25]
Pantalla táctil RPi	13,83
Panel solar	15,89
Controlador PWM	6,85
Batería	14,50
Placa de desarrollo ESP32	8,99
ACS712 30A	3,00
FZ0430	1,00
DHT22	2,41
BH1750	0,99
Alimentación breadboard	3,85
Alimentación portátil	2,29
Baterías de litio	4,10
Interruptores	0,92
Breadboards	11,99
Cables 'jumper'	3,99
Cable alimentación ESP32	1,80
Resistencias varias	2,94
Caja de contactores	1,00[26]
Router	0[27]
Multímetro	0
Software	
Raspberry Pi OS	0
Arduino IDE	0
Node-RED	0
Mosquitto Broker	0
VNC Viewer	0
Putty	0
Coste total del proyecto	**236,33[28]**

[25] Este coste es el correspondiente a un kit completo que incluye una carcasa con ventilador, disipadores metálicos, fuente de alimentación y tarjeta microSD de 128Gb. Se podría utilizar una raspberry Pi de menores prestaciones (como la de 4Gb de RAM – en torno a 60 €) y una tarjeta microSD de 32 Gb (5€) y el precio sería aún menor, haciendo el proyecto más atractivo económicamente. Se ha utilizado esta por ser la que tenía disponible para la realización del proyecto.
[26] Este ha sido el coste que ha tenido en mi caso por una compra en una gran superficie. El coste normal de esta caja oscila entre los 15 y los 20 €.
[27] El router y el multímetro ya los tenía disponible antes de realizar el proyecto. No ha sido necesaria su compra para el desarrollo de este.
[28] Este es el coste que ha tenido el proyecto para su realización. No se incluyen piezas y sensores comprados para posible reemplazo, en caso de avería o mal funcionamiento.

7. CONCLUSIONES

Se han cumplido todos los objetivos definidos para la realización del proyecto. Desde la parte física, experimental hasta la parte de programación y sistema de monitorización visual.

Se ha diseñado y construido el sistema fotovoltaico previsto de baja potencia como modelo de referencia para la correcta aplicación de los elementos de monitorización descritos a lo largo del proyecto, en el que se han encontrado obstáculos reales típicos, más o menos difíciles de solventar, durante la realización del proyecto.

Esto último nos ha permitido hacer un trabajo completamente basado en un caso real de aplicación. Por ejemplo, esta instalación, con mínimas modificaciones para la integridad tanto del equipo como del lugar de la instalación, podría ser instalada en un vehículo, barco o incluso en un área doméstica para la alimentación de dispositivos de bajo consumo energético (como por ejemplo un sistema de alarma de seguridad, carga de dispositivos móviles, etc).

Como segunda parte 'central' que comprende este proyecto, se ha diseñado satisfactoriamente un sistema SCADA robusto, fiable y de bajo consumo de recursos que incluye todas las variables mínimas deseadas para la monitorización de un sistema fotovoltaico. Además, este sistema SCADA diseñado sería aplicable en cualquier campo de la industria y doméstico, para cualquier tipo de instalación, proceso o servicio. Tan sólo habría que tener en cuenta las variables que se desean medir y monitorizar y unir hardware (sensores, controladores, actuadores, etc.) y software (SCADA y programación de los microcontroladores) para llevar a cabo el proyecto deseado.

Cabe destacar que, en mayor medida en la parte de programación, tanto en línea como visual, este ha sido un proyecto de iniciación personal en este tipo de tecnologías. Se han podido solventar todos los obstáculos que han aparecido durante la realización del trabajo con constancia, apoyo del tutor e investigación en la Red. Ha constituido un arduo pero interesante camino de aprendizaje, con una reconfortante sensación final de éxito, camino que se plasma de manera extraordinaria en la Figura 52.

Figura 52. Éxito. Imagen de iStock.

BIBLIOGRAFÍA

Abajo González, D., Falcone Lanas, F. J. & Azpilicueta, L., 2018. *Monitorización de una central fotovoltaica mediante sensores inalámbricos con tecnología Zigbee.* Pamplona: Universidad Pública de Navarra.

Aghenta, L. O. & Iqbal, T., 2019. Design and implementation of a low-cost, open source IoT-based SCADA system using ESP32 with OLED, ThingsBoard and MQTT protocol. *Electronics and Electrical Engineering,* pp. 57-86.

Alternative Enery Tutorials, 2019. *Photovoltaics Turn Photons into Electrons.* [En línea] Available at: https://www.alternative-energy-tutorials.com/photovoltaics/photovoltaics-turn-photons-into-electrons.html
[Último acceso: 22 Agosto 2021].

Aprendiendo Arduino, 2020. *Práctica 5: Programar Node-RED, MQTT y Dashboard.* [En línea] Available at: https://aprendiendoarduino.wordpress.com/tag/dashboard-node-red/
[Último acceso: 11 Julio 2021].

Aprendiendo Arduino, 2020. *Qué es Node-RED.* [En línea] Available at: https://aprendiendoarduino.wordpress.com/2020/03/05/que-es-node-red/
[Último acceso: 24 Agosto 2021].

Arduino, 2021. *Software de Arduino.* [En línea] Available at: https://arduino.cl/programacion/
[Último acceso: 21 Agosto 2021].

Arduino, A., 2020. *Instalación Node-RED.* [En línea] Available at: https://aprendiendoarduino.wordpress.com/category/node-red/
[Último acceso: 11 Julio 2021].

Atlam, H. F., Walters, R. J. & Wills, G. B., 2018. Internet of Things: State-of-the-art, Challenges, Applications, and Open Issues. *International Journal of Intelligent Computing Research (IJICR),* Septiembre.9(3).

AZ-Delivery, 2021. *ESP-32 Dev Kit C V2_ES Ebook.* s.l.:s.n.

Bandaancha.st, 2018. *Comtrend AR-5387un.* [En línea] Available at: https://wiki.bandaancha.st/Comtrend_AR-5387un
[Último acceso: 22 Agosto 2021].

Blog by Unit Electronics, 2021. *Cómo programas el DHT22 con Arduino IDE + ESP32.* [En línea]
Available at: https://blog.uelectronics.com/tarjetas-desarrollo/como-programar-el-dht22-con-el-arduino-ide-y-la-placa-esp32/
[Último acceso: 04 Julio 2021].

Breck, C., 2020. *The State of the Art for IoT.* [En línea] Available at: https://blog.colinbreck.com/the-state-of-the-art-for-iot/
[Último acceso: 17 Agosto 2021].

Castillo, D., 2018. *ESP32 con Node-Red.* [En línea]
Available at: https://www.youtube.com/watch?v=XI2i2LqHAew
[Último acceso: 10 Julio 2021].

Dumitru, C.-D. & Gligor, A., s.f. SCADA based software for renewable energy managment system.

Eclipse Mosquitto, 2021. *Eclipse Mosquitto - An open source MQTT broker.* [En línea]
Available at: https://mosquitto.org/
[Último acceso: 24 Agosto 2021].

García, J., 2021. *Making your own MQTT library - Ubidots.* [En línea]
Available at: https://help.ubidots.com/en/articles/750968-making-your-own-mqtt-library
[Último acceso: 18 Julio 2021].

Gobierno de Canarias, 2015. *Diseño electrónico con Fritzing.* [En línea]
Available at: https://www3.gobiernodecanarias.org/medusa/ecoescuela/recursosdigitales/2015/02/10/diseno-electronico-con-fritzing/
[Último acceso: 25 Agosto 2021].

IoT Sharing.com, 2017. *Demo 14: How to use MQTT and Arduino ESP32 to build a simple Smart Home system.* [En línea]
Available at: http://www.iotsharing.com/2017/05/how-to-use-mqtt-to-build-smart-home-arduino-esp32.html
[Último acceso: 11 Julio 2021].

Jana, S., 2013. *State of Art of Solar Photovoltaic Technology*. Bhubaneswar, India, Hindawi Publishing Corporation.

Karnouskos, S. & Walter Colombo, A., 2011. *Architecting the next generation of service-based SCADA/DCS systems of systems.* Alemania, s.n.

Kolipaka, K., 2019. *Slideshare. Eclipse Plugin for ESP-IDF-EclipseCon Europe 2019.*. [En línea]
Available at: https://www.slideshare.net/kondalkolipaka/espidf-eclipse-plugin-eclipsecon2019
[Último acceso: 22 Agosto 2021].

Llamas, L., 2016. *Medir cantidad de luxes con arduino y el luxómetro BH1750.* [En línea]
Available at: https://www.luisllamas.es/medir-cantidad-de-luxes-con-arduino-y-el-luxometro-bh1750/
[Último acceso: 24 Agosto 2021].

Llamas, L., 2016. *Medir voltajes de hasta 25 V con Arduino y FZ0430.* [En línea]
Available at: https://www.luisllamas.es/medir-voltajes-de-hasta-25v-con-arduino-y-fz0430/
[Último acceso: 04 Julio 2021].

Llamas, L., 2019. *¿Qué es MQTT? Su importancia como protocolo IoT.* [En línea]
Available at: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/
[Último acceso: 24 Agosto 2021].

Martínez Cacho, J. & Arizaleta Arteaga, M., 2018. *Monitorización de variables energéticas bajo soporte de Raspberry Pi.* Pamplona: Universidad Pública de Navarra.

Naylamp Mechatronics, 2021. *Módulo sensor de luz digital BH1750.* [En línea]
Available at: https://naylampmechatronics.com/sensores-luz-y-sonido/76-modulo-sensor-de-

luz-digital-bh1750.html
[Último acceso: 24 08 2021].

Nechibvute, A. & Mudzingwa, C., 2013. Wireless Sensor Networks for SCADA and Industrial Control Systems. *International Journal of Engineering and Technology.*

Node-RED, 2021. *Node-RED - Low-code programming for event-driven applications.* [En línea] Available at: https://nodered.org/
[Último acceso: 24 Agosto 2021].

Oriaghe Aghenta, L., 2020. *Open Source SCADA Systems for Small Renewable Power Generation.* St. John's Newfoundland and Labrador: Faculty of Engineering and Applied Science Memorial University of Newfoundland.

PanamaHitek, 2015. *El setup y el loop en Arduino.* [En línea] Available at: http://panamahitek.com/el-setup-y-el-loop-en-arduino/
[Último acceso: 25 Agosto 2021].

Pérez, X., 2019. *Maker Pro. A comparison of the New ES32-S2 to the ESP32.* [En línea] Available at: https://maker.pro/esp8266/tutorial/a-comparison-of-the-new-esp32-s2-to-the-esp32
[Último acceso: 22 Agosto 2021].

Programarfacil.com, L. D. V. H. -., 2019. *Introducción a Node-RED y Raspberry Pi con un sistema de alarma con Arduino.* [En línea] Available at: https://programarfacil.com/blog/raspberry-pi/introduccion-node-red-raspberry-pi/
[Último acceso: 10 Julio 2021].

Programo ergo sum, 2021. *¿Cómoo connfigurar mi Raspberry Pi?.* [En línea] Available at: https://www.programoergosum.com/cursos-online/raspberry-pi/236-acceso-a-raspberry-pi-remota-mediante-vnc/configurar-vnc
[Último acceso: 25 Agosto 2021].

Random Nerd Tutorials, 2017. *ESP8266 and Node-RED with MQTT (Publish and Subscribe).* [En línea] Available at: https://randomnerdtutorials.com/esp8266-and-node-red-with-mqtt/
[Último acceso: 11 Julio 2021].

Random Nerd Tutorials, 2017. *How to Install Mosquitto Broker on Raspberry Pi.* [En línea] Available at: https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/
[Último acceso: 11 Julio 2021].

Random Nerd Tutorials, 2018. *ESP32 Troubleshooting Guide.* [En línea] Available at: https://randomnerdtutorials.com/esp32-troubleshooting-guide/
[Último acceso: 28 Junio 2021].

Random Nerd Tutorials, 2020. *ESP32 MQTT - Publish BME280 Sensor Readings (Arduino IDE).* [En línea] Available at: https://randomnerdtutorials.com/esp32-mqtt-publish-bme280-arduino/
[Último acceso: 18 Julio 2021].

Raspberry Pi, 2021. *Raspberry Pi 4.* [En línea] Available at: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/
[Último acceso: 22 Agosto 2021].

Red Hat, 2021. *¿Qué es una API? Qué son las API y para qué sirven.* [En línea] Available at: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces#:~:text=Una%20API%20es%20un%20conjunto,de%20saber%20c%C3%B3mo%20est%C3%A1n%20implementados.
[Último acceso: 17 Agosto 2021].

Sicma21, 2021. *SCADA: qué es y cómo funciona.* [En línea] Available at: https://www.sicma21.com/scada-que-es-y-como-funciona/
[Último acceso: 17 Agosto 2021].

Statista, 2021. *Cumulative installed solar PV capacity worldwide fron 2000 to 2019.* [En línea] Available at: https://www.statista.com/statistics/280220/global-cumulative-installed-solar-pv-capacity/
[Último acceso: 22 Agosto 2021].

techtutorialsx, 2020. *ESP32: Connecting to a WiFi network.* [En línea] Available at: https://techtutorialsx.com/2017/04/24/esp32-connecting-to-a-wifi-network/
[Último acceso: 10 Julio 2021].

Telit, 2021. *MQTT Control Packets.* [En línea] Available at: https://docs.devicewise.com/Content/Products/IoT_Portal_API_Reference_Guide/MQTT_Interface/MQTT-Control-Packets.htm
[Último acceso: 24 Agosto 2021].

Tutorials, R. N., 2017. *Getting Started with Node-RED Dashboard.* [En línea] Available at: https://randomnerdtutorials.com/getting-started-with-node-red-dashboard/
[Último acceso: 10 Julio 2021].

Tutorials, R. N., 2018. *ESP32 MQTT - Publish and Subscribe with Arduino IDE.* [En línea] Available at: https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/
[Último acceso: 10 Julio 2021].

Tutorials, R. N., 2018. *ESP32 Static/Fixed IP Address.* [En línea] Available at: https://randomnerdtutorials.com/esp32-static-fixed-ip-address-arduino-ide/
[Último acceso: 10 Julio 2021].

UNO, R. -. M., 2019. *Introduccion a Arduino IDE.* [En línea] Available at: https://mecatronicauno.com/introduccion-arduino-ide-tutorial/
[Último acceso: 24 Agosto 2021].

Wikipeda - La Enciclopedia Libre, 2021. *Controlador lógico programable.* [En línea] Available at: https://es.wikipedia.org/wiki/Controlador_l%C3%B3gico_programable
[Último acceso: 17 Agosto 2021].

Wikipedia - La Enciclopedia Libre, 2020. *Unidad Terminal Remota.* [En línea] Available at: https://es.wikipedia.org/wiki/Unidad_Terminal_Remota
[Último acceso: 17 Agosto 2021].

Wikipedia - La Enciclopedia Libre, 2021. *Raspberry Pi OS.* [En línea] Available at: https://es.wikipedia.org/wiki/Raspberry_Pi_OS
[Último acceso: 25 Agosto 2021].

A N E X O S

ANEXO A: ESQUEMA CONEXIONES FRITZING

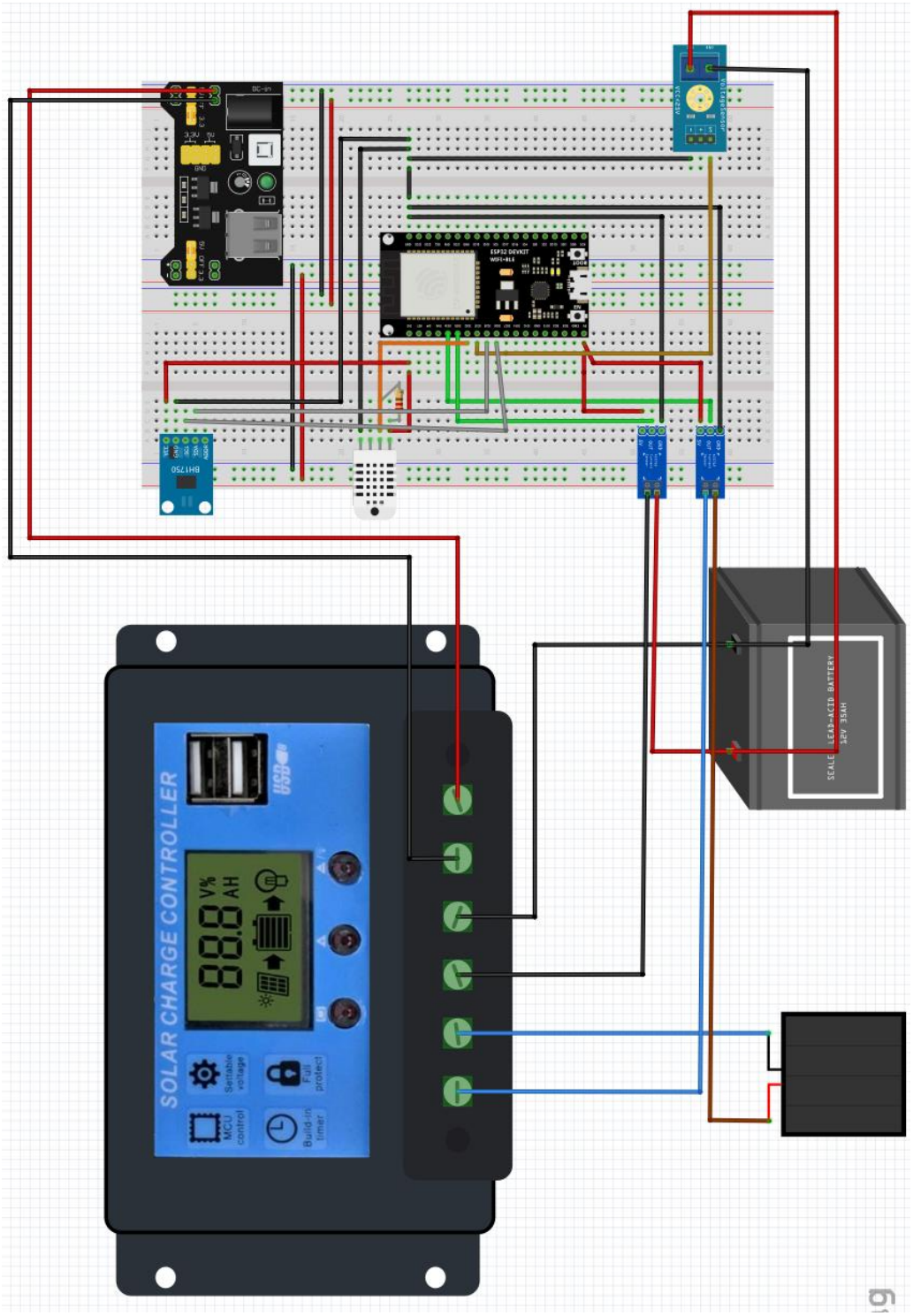


Figura 53.Esquema conexiones Fritzing.

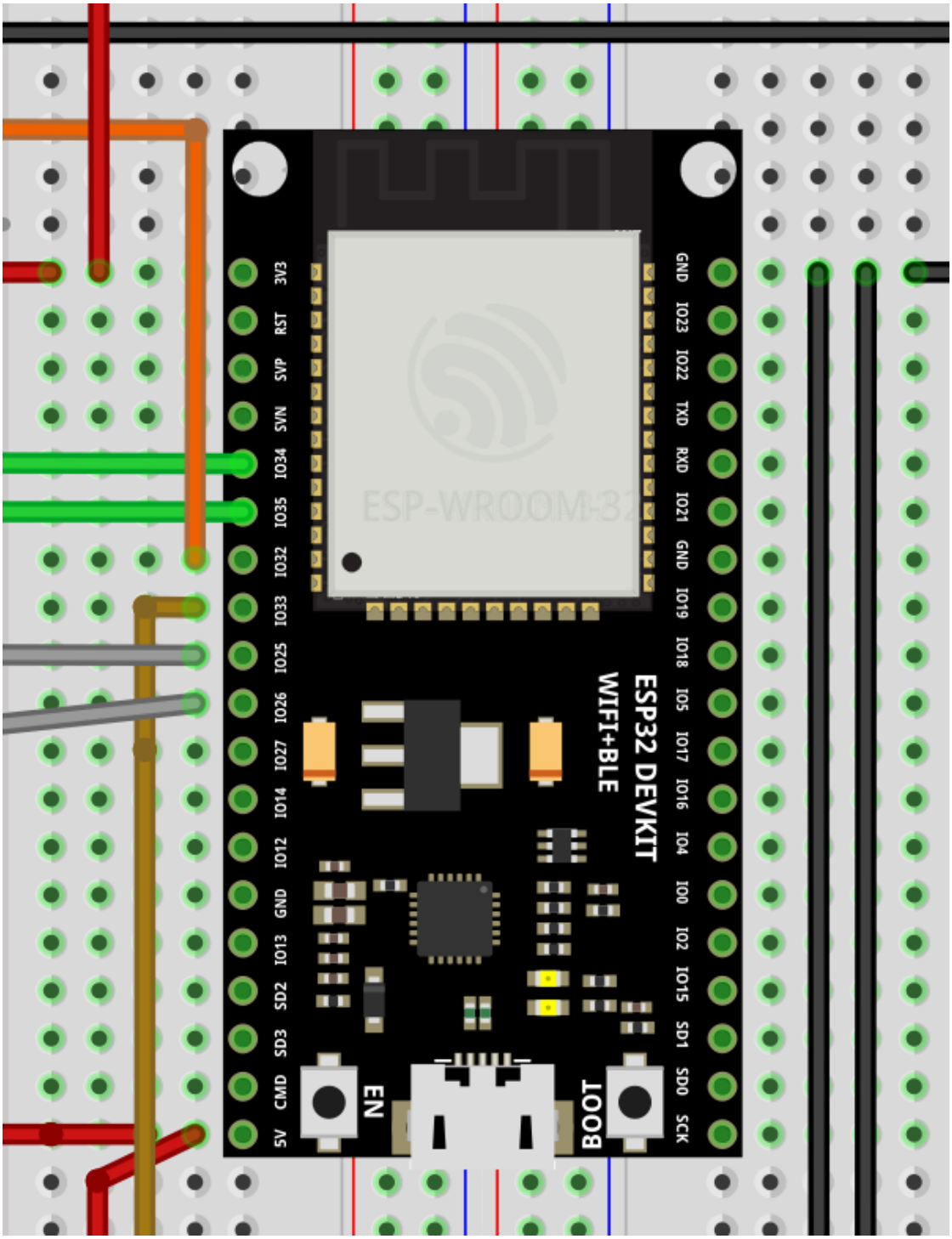


Figura 54. Esquema conexiones. ESP32.

ANEXO B: IMÁGENES MONITOR SERIE

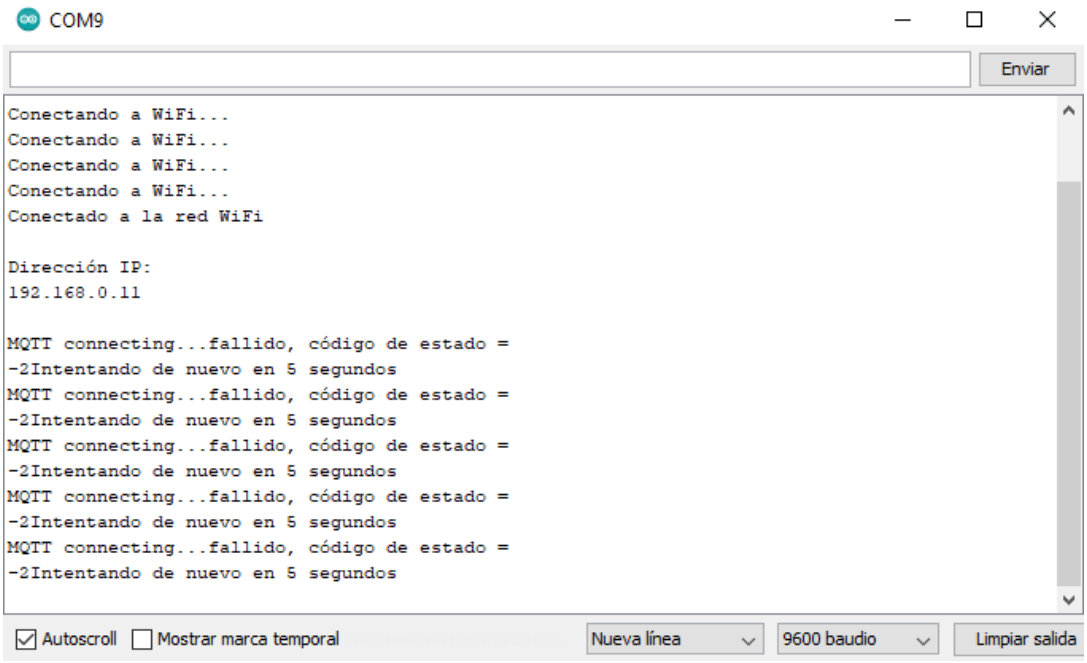


Figura 55. Monitor serie. Conexión WiFi y MQTT.

Nota: en este caso no conectaba porque se tenía conectado un cable Ethernet a la vez que estaba conectado a la red WiFi, no estaba configurado correctamente, aparecía un conflicto de IPs.

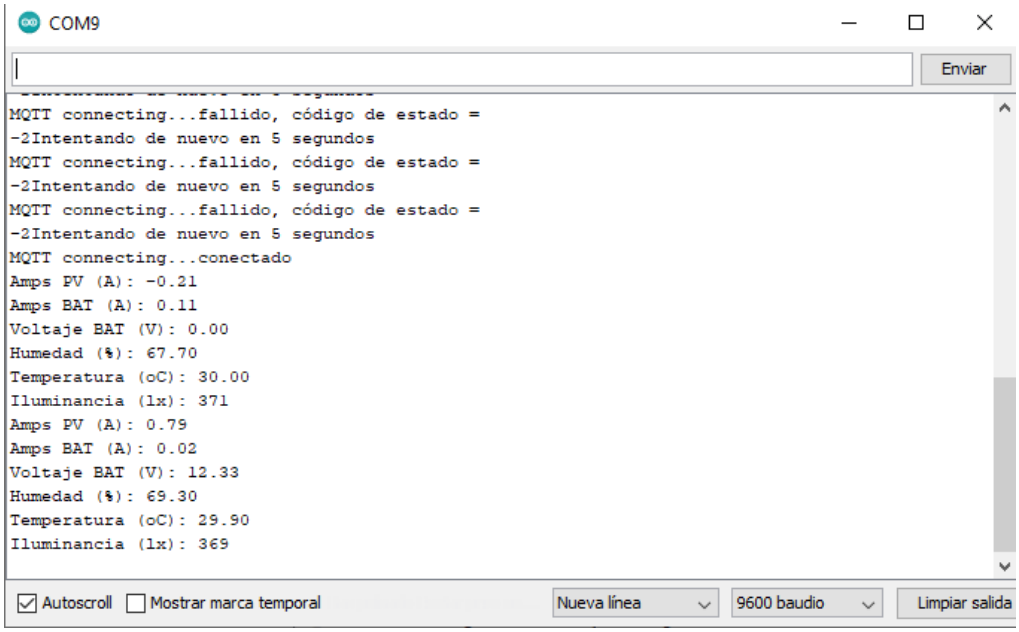


Figura 56. Monitor serie. Conexión WiFi y MQTT. Valores de los sensores.

ANEXO C: VENTANAS DE CONFIGURACIÓN DEL SCADA

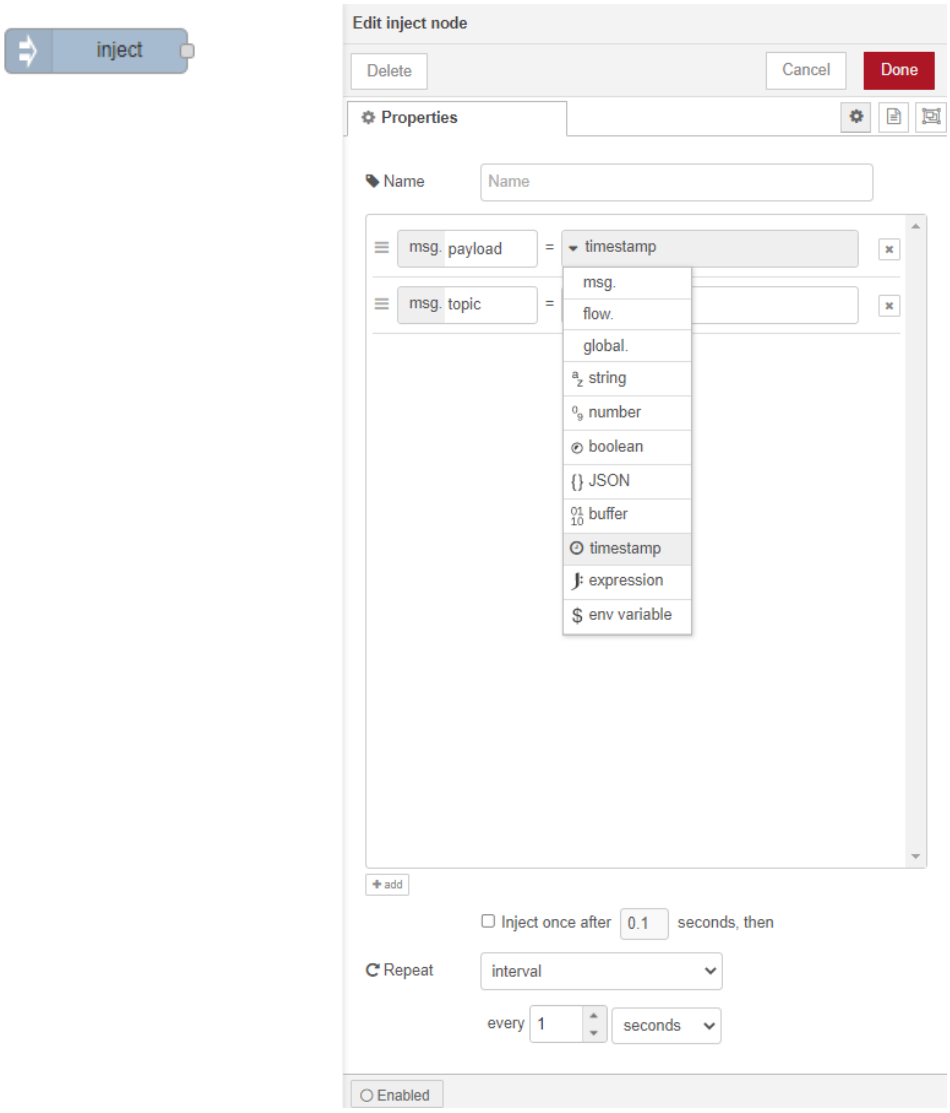


Figura 57. Configuración nodo *inject* tipo *timestamp*

En este ejemplo (Figura 57) se inyecta un *timestamp* (se elige del desplegable) en formato epoch (incluye toda la información de tiempo – fecha y hora). Se hace en intervalos de 1 segundo.

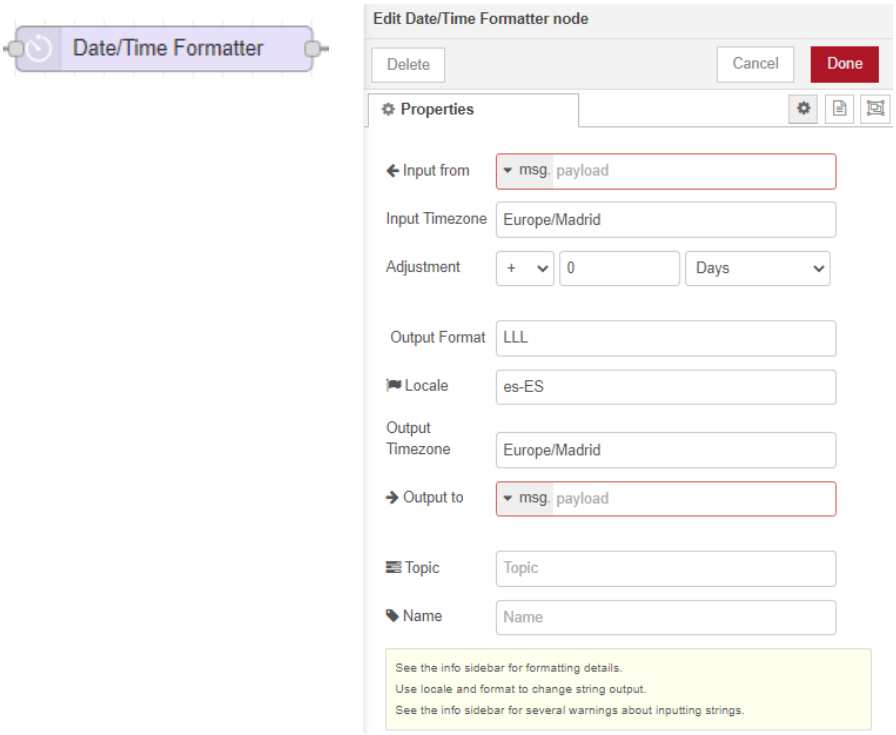


Figura 58. Configuración nodo *Date/Time Formatter.*

Se establece el formato de salida en LLL, y es-ES así como la zona horaria en Europa/Madrid (Figura 58).

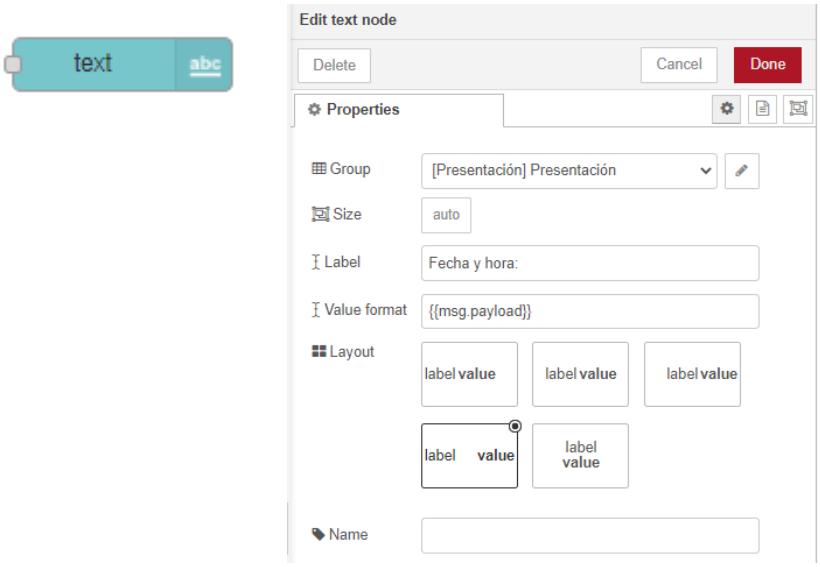


Figura 59. Configuración nodo *text*.

Se nombra el nodo como 'Fecha y hora: ' (Figura 59). Aparece aquí por primera vez la casilla 'Group'. A todo elemento del *Dashboard* se le ha de asignar un grupo y una pestaña. En este caso está asignado a la Pestaña Presentación apartado presentación, que es de la forma que se distribuye después en el SCADA. Se hace lo mismo para cada una de las otras pestañas para los diferentes sensores implementados.

Ver Figura 60. Si se le da al icono del lápiz que aparece al lado de 'Tab', permite modificar y crear otras pestañas con otros nombres.

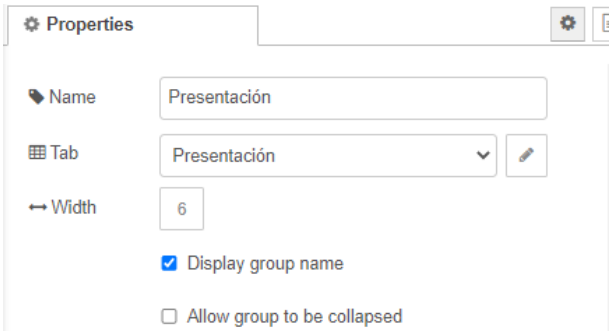


Figura 60. Edición de la pestaña y grupo en el *Dashboard*.

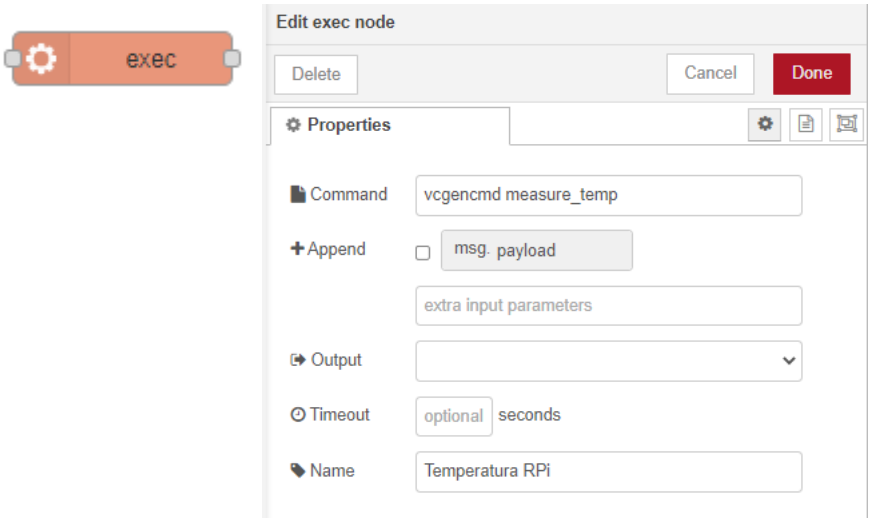


Figura 61. Configuración nodo *exec*.

Se trata de un nodo de ejecución (Figura 61). Lanza y ejecuta uno o varios comandos de consola, en este caso en la Raspberry Pi en la que se está ejecutando el Node-RED. En este caso particular se trata de un comando para recuperar la temperatura de la RPi.

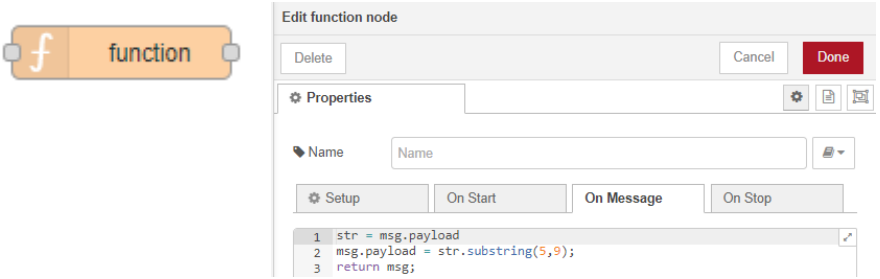


Figura 62. Configuración nodo *function*.

Este nodo (Figura 62) permite realizar una serie de operaciones a datos provenientes de otros nodos mediante una o varias líneas de código. En este caso se trata de un código para

acondicionar los datos que salen del nodo de ejecución que recaba los datos de la temperatura de la Raspberry Pi.

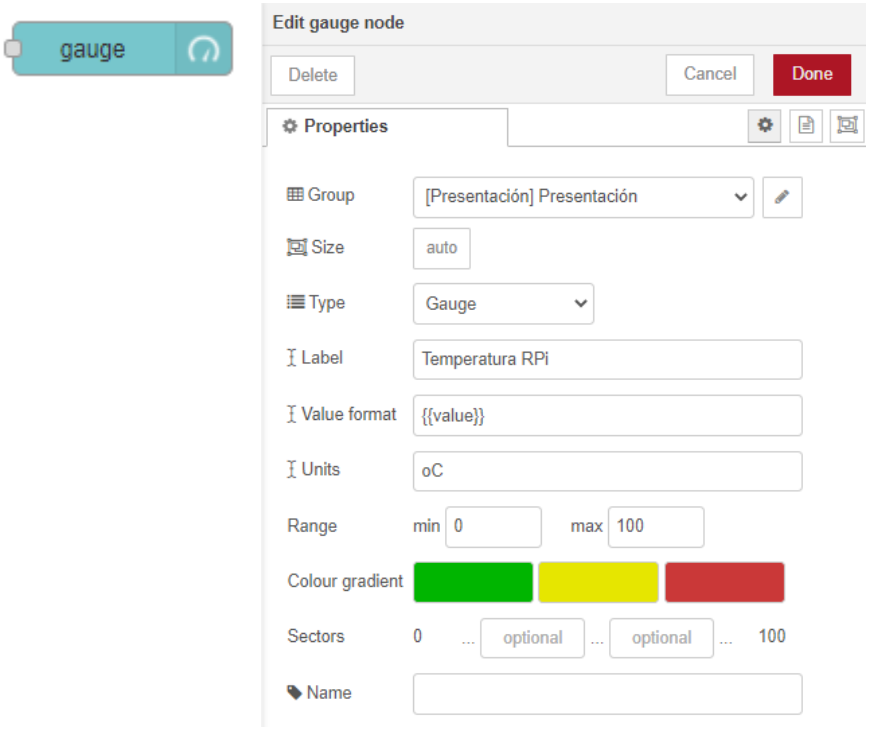


Figura 63. Configuración nodo Gauge del *Dashboard*.

De nuevo, en este nodo (Figura 63), se ha de seleccionar dónde aparecerá en el SCADA (*Dashboard*) en el apartado 'Group'. Se selecciona el tipo de gráfico (en este caso *Gauge*, pero también puede ser *Donut, Compass* y *Level*). Se establecen el nombre de los ejes y las unidades, así como el rango de medición del sensor y el ajuste de colores.

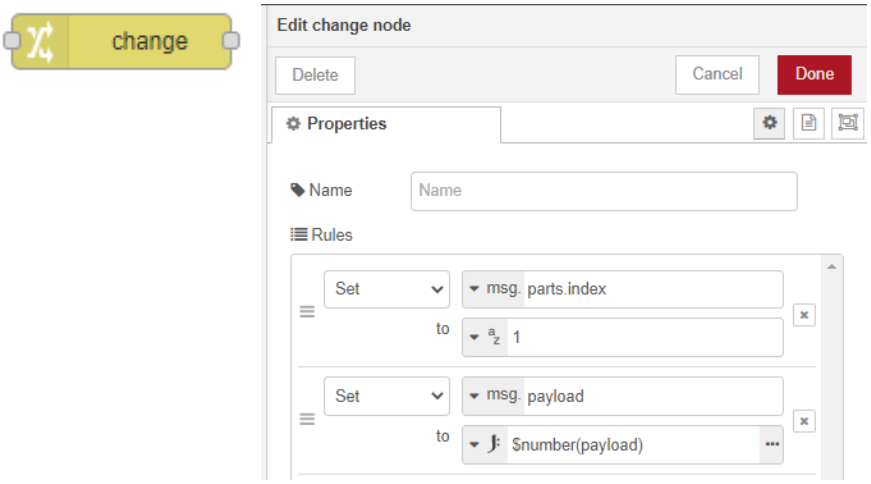


Figura 64. Configuración nodo *change*.

En este caso (Figura 64) se trata del nodo *change* para la asignación de un número de identificación al dato proveniente de otro nodo, aquí el 1 (*parts.index*). Así mismo, este nodo también consta de otra regla que establece el paso del *payload*, de tipo *string* a formato número.

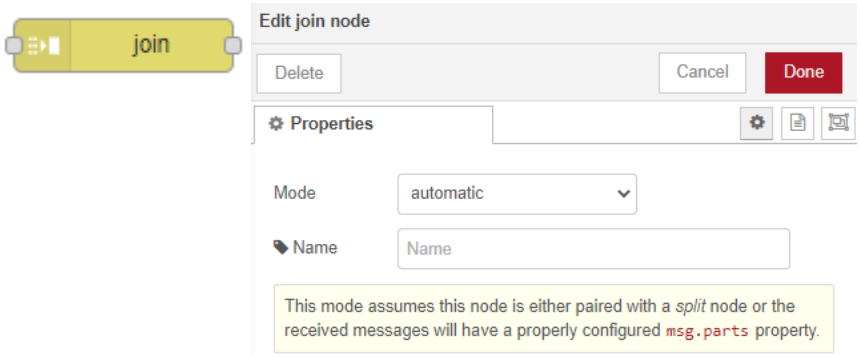


Figura 65. Configuración nodo *join*.

Para este nodo (Figura 65) se deja la configuración por defecto, en automático.

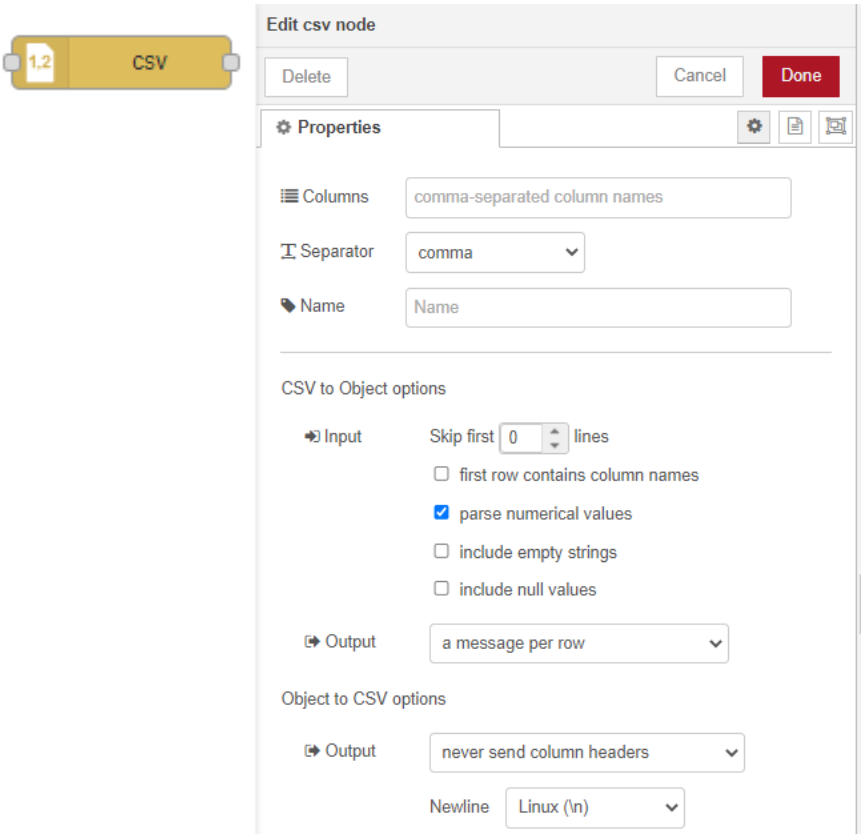


Figura 66. Configuración nodo *csv*.

En este nodo (Figura 66) se establece el separador de los datos como una coma. Además, se fija que la salida sea un mensaje por fila de texto.

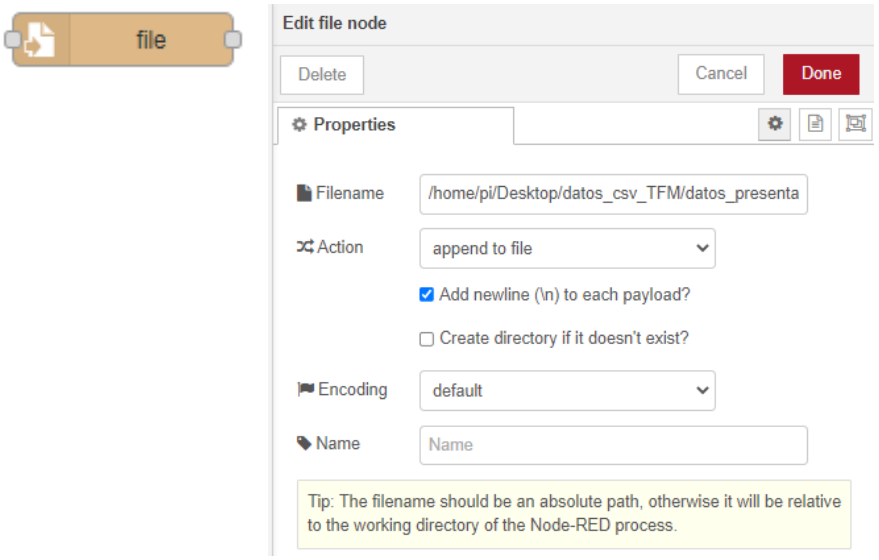


Figura 67. Configuración nodo *file*.

En el nodo *file* (Figura 67) se ha de establecer la ruta absoluta donde queremos guardar el fichero, así como el nombre del fichero acabado en .csv al final de esa ruta. Se establece que se añada una nueva línea por cada mensaje o grupo de mensajes que llega (*payload*).

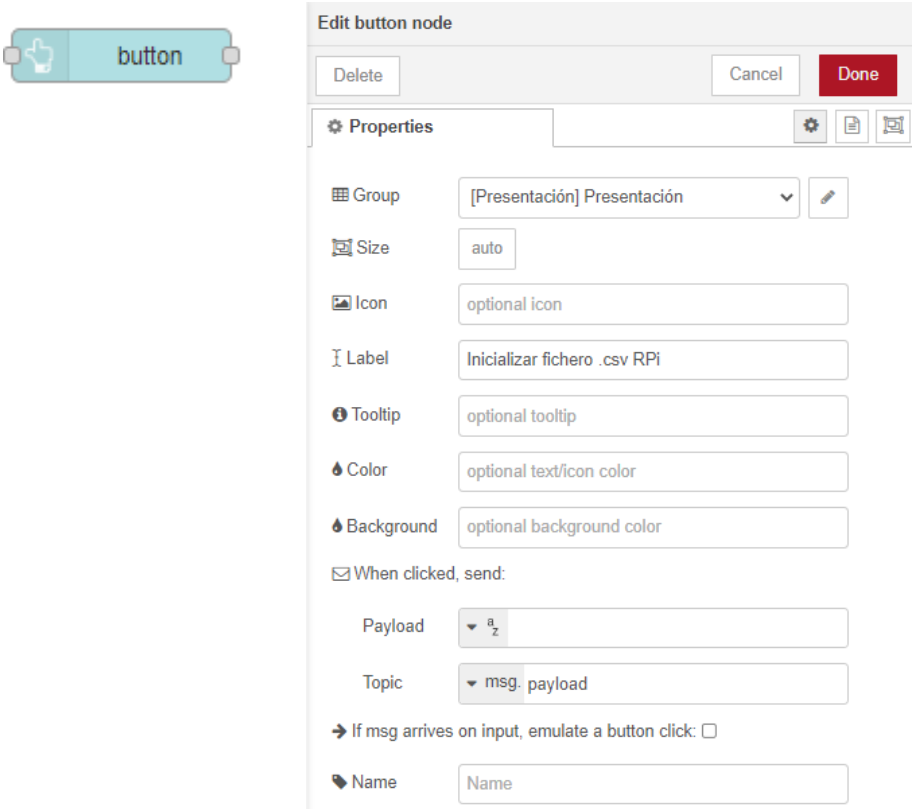


Figura 68. Configuración nodo *button*.

En este nodo (Figura 68) asigna al grupo al que pertenece, así como el nombre que debe aparecer sobre el botón en el *Dashboard*.

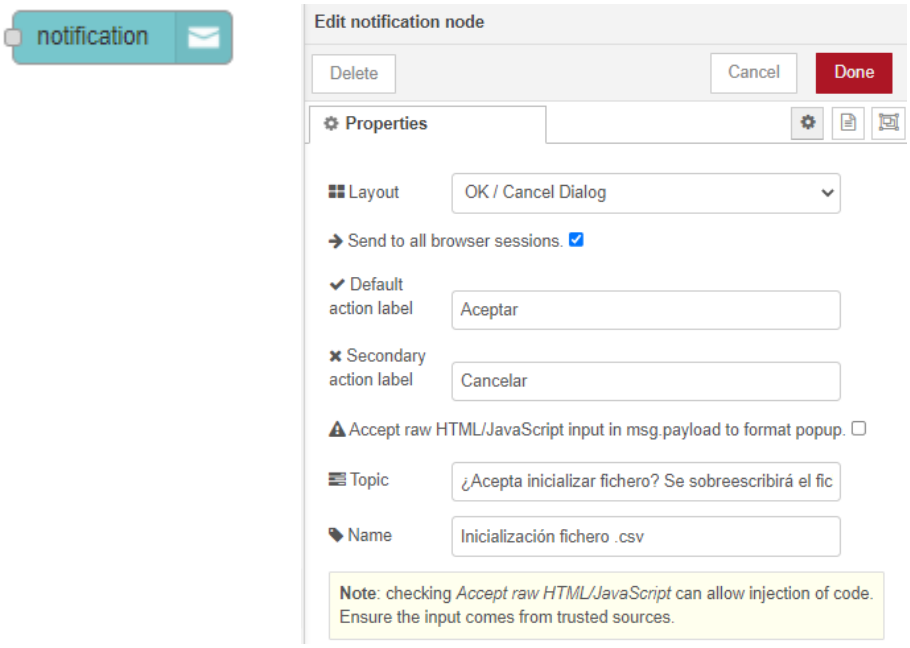


Figura 69. Configuración nodo *notification*.

Se trata de otro nodo del *Dashboard* (Figura 69). Aparecerá la notificación al pulsar un botón. Se ha de establecer el tipo de salida, en este caso OK/Cancel. Se definen ambas opciones, así como el texto que debe aparecer.

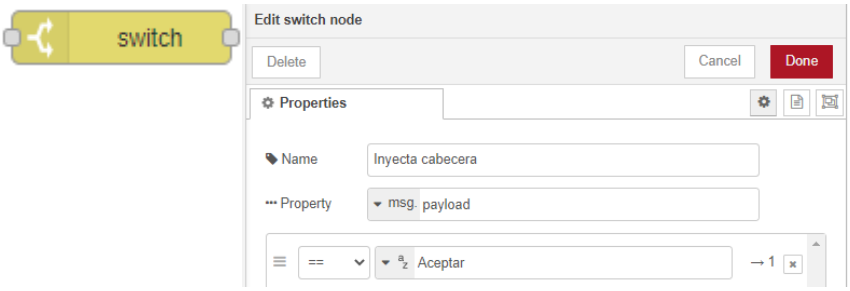


Figura 70. Configuración nodo *switch*.

El nodo switch (Figura 70) establece la condición de que si la entrada (proveniente del nodo que le preceda, en el caso de este proyecto es un nodo de notificación) es igual a Aceptar, continúa con el siguiente nodo.

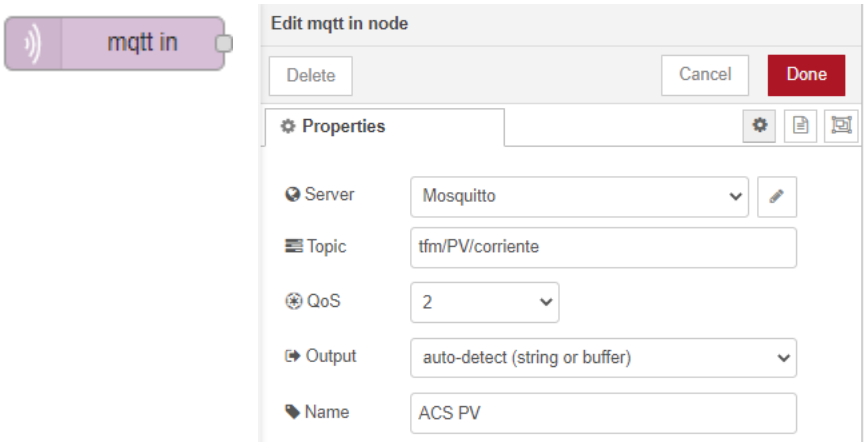


Figura 71. Configuración nodo *mqtt in*.

En el nodo *mqtt* in (Figura 71) se ha de fijar el servidor central (bróker) al que conectarse. Éste se define en el botón del lápiz al lado de la casilla 'Server' (IP y puerto). Se completa la casilla 'Topic' de acuerdo con los mensajes que queramos recibir en este nodo.

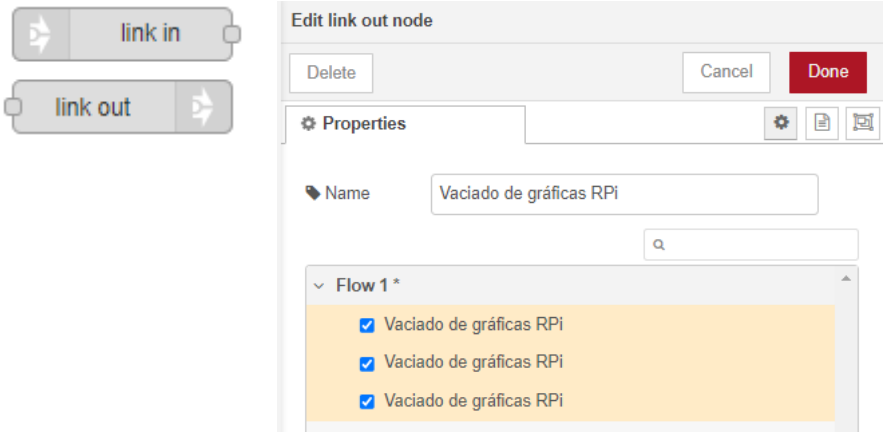


Figura 72. Configuración nodo *link*.

El *link out* se conecta a los nodos que queramos trasladar a otro lugar. Con el *link in* tendremos ese mismo nodo conectado anteriormente en otro lugar. Útil para no tener todo el espacio de trabajo lleno de líneas superpuestas, haciendo la lectura mucho más ardua. Ver Figura 72.

Por último, el nodo *debug* (Figura 73), se utiliza por defecto y sirve para mostrar elementos (los que esté conectados a este nodo) en la consola de depuración de Node-RED Developer.

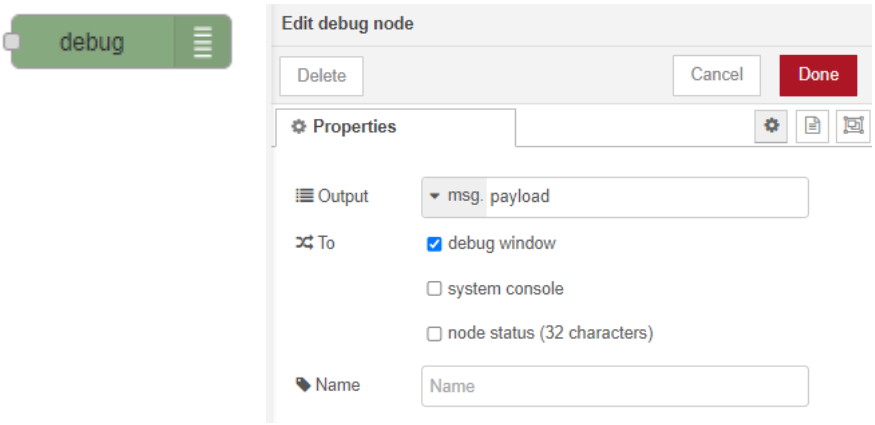


Figura 73. Configuración nodo *debug*.

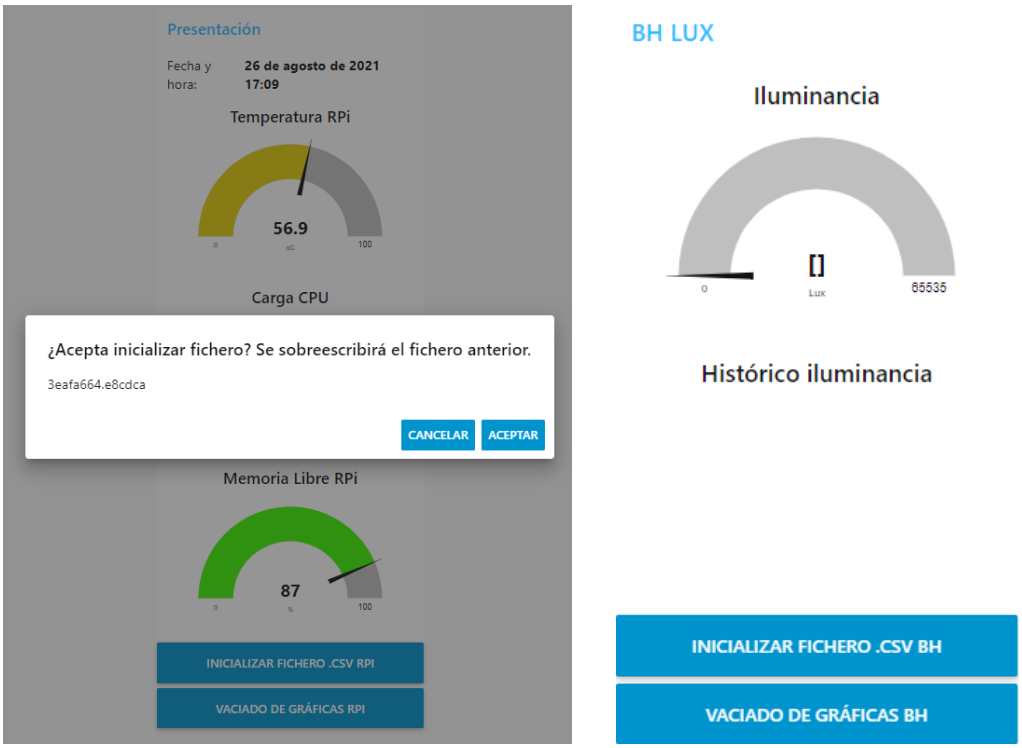


Figura 74. Efecto botones inicialización y vaciado de gráficas.

En la Figura 74 se muestra el efecto que tiene presionar los botones que aparecen en la interfaz de usuario del SCADA diseñado. En el caso del botón de inicialización, al darle a aceptar se borra el fichero existente en la carpeta del escritorio de la RPi de fichero de datos y se comienza uno nuevo. El botón de vaciado de gráficas reinicia las gráficas a su estado inicial, se empiezan a actualizar de nuevo tras llegar de nuevo mensajes con datos en un período de tiempo de unos segundos.

ANEXO D: CÓDIGO COMPLETO ESP32

```
// ----------x-----------x-------------x-------------x-----------
x--------------x---------
// LIBRERÍAS
// ----------x-----------x-------------x-------------x-----------
x--------------x---------

#include <WiFi.h> // WiFi

#include <PubSubClient.h> // para protocolo MQTT

#include <driver/adc.h> // ADC del ESP32

#include <DHT.h> // Carga de la librería para DHT22
#include <DHT_U.h>

#include <Wire.h> // Comunicación I2C
#include <BH1750.h> // Librería sensor BH1750
#include <BH1750FVI.h>

// ----------x-----------x-------------x-------------x-----------
x--------------x---------
// Definición de variables y configuración WiFi
// ----------x-----------x-------------x-------------x-----------
x--------------x---------

const char* ssid = "JAZZTEL_84E5";
const char* contrasena = "36CA58A7E249A386858F";

// IP estática ESP32
IPAddress local_IP(192, 168, 0, 11);
// IP router
IPAddress gateway(192, 168, 0, 1);
// Máscara de red y DNS
IPAddress subnet(255, 255, 0, 0);
IPAddress primaryDNS(212, 166, 211, 23); // opcional
IPAddress secondaryDNS(212, 166, 132, 104); // opcional

// ----------x-----------x-------------x-------------x-----------
x--------------x---------
// Definición de variables y configuración MQTT
// ----------x-----------x-------------x-------------x-----------
x--------------x---------

const char* mqtt_servidor = "192.168.0.10";

WiFiClient espClient;
PubSubClient client(espClient); // Crear una instancia del ciente
PubSubClient

// TOPICS:

#define ACS_PV_TOPIC "tfm/PV/corriente"
#define ACS_BAT_TOPIC "tfm/BAT/corriente"
#define FZ_BAT_TOPIC "tfm/BAT/voltaje"
#define DHT_HUM_TOPIC "tfm/DHT/hum"
#define DHT_TEMP_TOPIC "tfm/DHT/temp"
#define BH_LUX_TOPIC "tfm/BH/lux"
```

```
long lastMsg = 0;
char msg_1[20];
char msg_2[20];
char msg_3[20];
char msg_4[20];
char msg_5[20];
char msg_6[20];

void receivedCallback(char* topic, byte* payload, unsigned int length)
{
  Serial.print("Message received: ");
  Serial.print(topic);

  Serial.print("payload: ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
   }
   Serial.println();
}

void mqttconnect() {
  // Loop hasta que se reconecta
  while (!client.connected()) {
    Serial.print("MQTT connecting...");
    // ID del cliente
    String clientId = "ESP32Client";
    // Conectarse ahora
    if (client.connect(clientId.c_str())) {
      Serial.println("conectado");
    } else {
      Serial.println("fallido, código de estado = ");
      Serial.print(client.state());
      Serial.println("Intentando de nuevo en 5 segundos");
      // Esperar 5 segundos antes de volver a intentarlo
      delay (5000);
    }
  }
}

// ----------x------------x--------------x--------------x------------
x--------------x---------
// Definición de variables sensor ACS712 30 A - Analógico
// ----------x------------x--------------x--------------x------------
x--------------x---------

const int analogIn_ACS_PV = 35; // Pin panel fotovoltaico ACS 712
const int analogIn_ACS_BAT = 34; // Pin batería ACS 712
int mVperAmp = 66; // mV por amperio - característica del sensor
int SensorValue_ACS_PV = 0;
int SensorValue_ACS_BAT = 0;
int ACSoffset = 2538; //mV - característica del sensor. Teóricamente
es 2500, pero midiendo con el ESP32, lo que obtenemos de media para
valores de 0 (sin carga) es ese valor.
double Voltage_ACS_PV = 0;
double Voltage_ACS_BAT = 0;
double AmpsPV = 0;
double AmpsBAT = 0;
```

```
// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// Definición de variables sensor FZ0430 - Analógico
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

const int analogIn_FZ_BAT = 33; // Pin Batería - FZ0430
int SensorValue_FZ_PV = 0;
int SensorValue_FZ_BAT = 0;
double Voltage_FZ_PV = 0;
double Voltage_FZ_BAT = 0;

// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// Definición de variables sensor DHT22 - Digital
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

#define DHTTYPE DHT22 // Modelo de sensor
#define DHTPIN 32 // Pin sensor T y pp DHT22
DHT dht(DHTPIN, DHTTYPE, 22);
double Hum = 0;
double Temp = 0;

// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// Definición de variables sensor BH1750 - Digital
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

const int digitalIN_BH_SDA = 26; // Pin correspondiente conectado al
pin SDA del sensor
const int digitalIN_BH_SCL = 27; // Pin correspondiente conectado al
pin SCL del sensor

BH1750 luxometro;

// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// SETUP SETUP SETUP SETUP SETUP SETUP SETUP
// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

void setup() {

  // Baudios
  Serial.begin(9600); // Iniciación comunicación serie

  // WiFI

  WiFi.begin(ssid, contrasena);

  if (!WiFi.config(local_IP, gateway, subnet, primaryDNS,
secondaryDNS)){
    Serial.println("STA configuración fallida");
```

```
  }

  while (WiFi.status() != WL_CONNECTED) {
    delay (500);
    Serial.println("Conectando a WiFi...");
  }

  Serial.println("Conectado a la red WiFi");

  Serial.println("");
  Serial.println("Dirección IP: ");
  Serial.println(WiFi.localIP());
  Serial.println("");

  // Configuración de la IP del servidor MQTT y el puerto
  client.setServer(mqtt_servidor, 1883);
  client.setCallback(receivedCallback); // Esta función
"receivedCallback" será invocada cuando reciba el topic subscrito

  // Configuración del ADC del ESP32:
  analogReadResolution(12); // 12 bits
  analogSetAttenuation(ADC_11db);

  // Definición tipo de PIN
  pinMode(analogIn_ACS_PV, INPUT);
  pinMode(analogIn_ACS_BAT, INPUT);
  // pinMode(analogIn_FZ_PV, INPUT);
  pinMode(analogIn_FZ_BAT, INPUT);

  // Inicialización sensores digitales
  dht.begin(); // Se inicia la comunicación con el DHT22

  Wire.begin(digitalIN_BH_SDA, digitalIN_BH_SCL); // Para iniciar la
comunicación I2C - se configuran los pines del sensor (en caso
contrario no da valores y se obtiene mensaje de advertencia)
  luxometro.begin(); // Inicialización sensor BH1750 - resolución por
defecto

}

// ----------x------------x--------------x--------------x------------
x--------------x---------
// ----------x------------x--------------x--------------x------------
x--------------x---------
// LOOP LOOP LOOP LOOP LOOP LOOP LOOP
// ----------x------------x--------------x--------------x------------
x--------------x---------
// ----------x------------x--------------x--------------x------------
x--------------x---------

void loop() {

// Si el cliente estaba desconectado, intentar conectarlo de nuevo
  if (!client.connected()) {
    mqttconnect();
  }
// La siguiente función escuhará a los "incoming subscribed topic-
process-invoke receivedCallback"
  client.loop();
```

```
// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// SENSORES DE CORRIENTE - ACS712 1 & 2
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

  SensorValue_ACS_PV = analogRead(analogIn_ACS_PV);
  Voltage_ACS_PV = (SensorValue_ACS_PV * 3418) / 4095; // mV - el
voltaje de referencia teórico es 3300 mV. Ese valor ha sido calculado
mediante las discrepancias reales con el valor teórico.
  AmpsPV = ((Voltage_ACS_PV - ACSoffset) / mVperAmp); // amps
  delay(1000);

  Serial.print("Amps PV (A): "); // Se muestra en el monitor serie el
valor de la corriente a través de los terminales del controlador PWM
conectados a batería.
  Serial.println(AmpsPV);
  delay (500);

  snprintf (msg_1, 20, "%1f", AmpsPV);
  // Publicar el mensaje
  client.publish(ACS_PV_TOPIC, msg_1);
  delay(1000);


  SensorValue_ACS_BAT = analogRead(analogIn_ACS_BAT);
  Voltage_ACS_BAT = (SensorValue_ACS_BAT * 3418) / 4095; // mV - el
voltaje de referencia teórico es 3300 mV. Ese valor ha sido calculado
mediante las discrepancias reales con el valor teórico.
  AmpsBAT = ((Voltage_ACS_BAT - ACSoffset) / mVperAmp); // amps
  delay(1000);

  Serial.print("Amps BAT (A): "); // Se muestra en el monitor el valor
de la corriente a través de los terminales de salida del controlador
PWM.
  Serial.println(AmpsBAT);
  delay (500);

  snprintf (msg_2, 20, "%1f", AmpsBAT);
  // Publicar el mensaje
  client.publish(ACS_BAT_TOPIC, msg_2);
  delay (1000);

// ----------x-----------x-------------x-------------x-----------
x-------------x---------
// SENSORES DE TENSIÓN - FZ0430 1 & 2. Actualización: Sensor V del PV
anulado, interfería mediante el ESP32 con el controlador PWM.
// ----------x-----------x-------------x-------------x-----------
x-------------x---------

  SensorValue_FZ_BAT = analogRead(analogIn_FZ_BAT);
  Voltage_FZ_BAT = (SensorValue_FZ_BAT * 16.5) / 4095; // En este caso
el voltaje de referencia no son 25 V, lo máximo que podrá medir son
16,5 debido a que el pin del ESP32 trabaja como máximo a 3,3 V y no a
5 V. (4095 se corresponderá con 3,3 V).
  delay (1000);

  Serial.print("Voltaje BAT (V): ");
  Serial.println(Voltage_FZ_BAT);
```

```
  delay (500);

  snprintf (msg_3, 20, "%1f", Voltage_FZ_BAT);
  // Publicar el mensaje
  client.publish(FZ_BAT_TOPIC, msg_3);
  delay (1000);

// ----------x------------x--------------x--------------x------------
x--------------x---------
// SENSORES DE HUMEDAD Y TEMPERATURA - DHT22
// ----------x------------x--------------x--------------x------------
x--------------x---------

  Hum = dht.readHumidity(); // Lectura de la humedad. Asignación del
valor a la variable 'Hum'.
  Temp = dht.readTemperature(); // Lectura de la temperartura.
Asignación del valor a la variable 'Temp'
  delay (1000);

  Serial.print("Humedad (%): ");
  Serial.println(Hum);
  delay (500);

  snprintf (msg_4, 20, "%1f", Hum);
  // Publicar el mensaje
  client.publish(DHT_HUM_TOPIC, msg_4);
  delay (1000);

  Serial.print("Temperatura (oC): ");
  Serial.println(Temp);
  delay (500);

  snprintf (msg_5, 20, "%1f", Temp);
  // Publicar el mensaje
  client.publish(DHT_TEMP_TOPIC, msg_5);
  delay (1000);


// ----------x------------x--------------x--------------x------------
x--------------x---------
// SENSOR DE LUZ - BH1750
// ----------x------------x--------------x--------------x------------
x--------------x---------

  uint16_t lux = luxometro.readLightLevel(); // Lectura del sensor
  delay (1000);

  Serial.print(F("Iluminancia (lx): "));
  Serial.println(lux);
  delay (500);

  double lux_pub = lux; // Cambiamos el tipo de variable para
publicarla

  snprintf (msg_6, 20, "%1f", lux_pub);
  // Publicar el mensaje
  client.publish(BH_LUX_TOPIC, msg_6);
  delay (1000);
   }
```

ANEXO E: CÓDIGO DE IMPORTACIÓN DEL SCADA

[{"id":"ef5bf662.410aa8","type":"tab","label":"Flow
1","disabled":false,"info":""},{"id":"c885cb85.c04b48","type":"mqtt
in","z":"ef5bf662.410aa8","name":"ACS
PV","topic":"tfm/PV/corriente","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","nl":f
alse,"rap":true,"rh":0,"x":120,"y":1240,"wires":[["6c0cf0d0.dd338","6ef0b9ff.2c61c8","8a05c5
ef.9843a8"]]},{"id":"6c0cf0d0.dd338","type":"ui_chart","z":"ef5bf662.410aa8","name":"","gro
up":"9404bef5.5e24b","order":0,"width":0,"height":0,"label":"Histórico corriente
panel","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodat
a":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOlder
Unit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7e8","
#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"useDi
fferentColor":false,"x":790,"y":1180,"wires":[[]]},{"id":"7c6e9627.df2098","type":"mqtt
in","z":"ef5bf662.410aa8","name":"ACS
BAT","topic":"tfm/BAT/corriente","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","n
l":false,"rap":true,"rh":0,"x":120,"y":1400,"wires":[["8c36511.2d204b","a0807760.aebc28","e7
969eb9.21c8a","e4b671e6.1e71c"]]},{"id":"8c36511.2d204b","type":"ui_chart","z":"ef5bf662.
410aa8","name":"","group":"7500e219.92f82c","order":2,"width":0,"height":0,"label":"Históri
co corriente
batería","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nod
ata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOld
erUnit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7e8",
"#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"useD
ifferentColor":false,"x":790,"y":1460,"wires":[[]]},{"id":"7b46ed04.073b14","type":"mqtt
in","z":"ef5bf662.410aa8","name":"FZ
BAT","topic":"tfm/BAT/voltaje","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","nl"
:false,"rap":true,"rh":0,"x":110,"y":1900,"wires":[["f58ecbca.fef008","2eeea5b2.55e57a","85e5
4e0d.dcb07","83995526.628b18"]]},{"id":"6449fe1b.192cc","type":"mqtt
in","z":"ef5bf662.410aa8","name":"DHT
HUM","topic":"tfm/DHT/hum","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","nl":f
alse,"rap":true,"rh":0,"x":120,"y":2440,"wires":[["dbb5d54c.09ce78","6133da46.617004","ec5
d0d3.fdb45f"]]},{"id":"60a278ff.fa2678","type":"mqtt
in","z":"ef5bf662.410aa8","name":"DHT
TEMP","topic":"tfm/DHT/temp","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","nl"
:false,"rap":true,"rh":0,"x":130,"y":2600,"wires":[["8a82de3.c789a2","1ba13b4f.abe355","a0af
39b4.db1ca8","82ffed9a.b8983"]]},{"id":"5bfc36be.d36828","type":"mqtt
in","z":"ef5bf662.410aa8","name":"BH
LUX","topic":"tfm/BH/lux","qos":"2","datatype":"auto","broker":"2a091142.00b8ce","nl":fals
e,"rap":true,"rh":0,"x":120,"y":3100,"wires":[["df3fe49c.cd33b8","2dd455e.cf883aa","5da49c2
e.18a9d4","8778fe30.db0c6"]]},{"id":"8a82de3.c789a2","type":"ui_gauge","z":"ef5bf662.410a
a8","name":"","group":"e4e86617.75cda8","order":1,"width":0,"height":0,"gtype":"gage","title
":"Temperatura","label":"oC","format":"{{value}}","min":0,"max":"100","colors":["#00b500",
"#e6e600","#ca3838"],"seg1":"","seg2":"","x":750,"y":2600,"wires":[]},{"id":"dbb5d54c.09ce
78","type":"ui_gauge","z":"ef5bf662.410aa8","name":"","group":"15e0cb08.d66905","order":1
,"width":0,"height":0,"gtype":"gage","title":"%
Humedad","label":"%","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e
600","#ca3838"],"seg1":"","seg2":"","x":750,"y":2440,"wires":[]},{"id":"f58ecbca.fef008","typ
e":"ui_gauge","z":"ef5bf662.410aa8","name":"","group":"44a34dca.01f0a4","order":1,"width":
0,"height":0,"gtype":"wave","title":"Tensión
Batería","label":"V","format":"{{value}}","min":0,"max":"15","colors":["#00b500","#e6e600"
,"#ca3838"],"seg1":"","seg2":"","x":760,"y":1900,"wires":[]},{"id":"df3fe49c.cd33b8","type":"

ui_gauge","z":"ef5bf662.410aa8","name":"","group":"d69ee820.38cd88","order":1,"width":0,"
height":0,"gtype":"gage","title":"Iluminancia","label":"Lux","format":"{{value}}","min":0,"ma
x":"65535","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":750,"y":3100,"
wires":[]},{"id":"fe20dc17.c4916","type":"inject","z":"ef5bf662.410aa8","name":"","props":[{
"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"1","crontab":"","once":false,"onceDelay":0.1
,"topic":"","payload":"","payloadType":"date","x":110,"y":100,"wires":[["378c451.0031bba"]]
},{"id":"1eacb392.06b40c","type":"ui_text","z":"ef5bf662.410aa8","group":"c2ddd6a2.0e2a08
","order":1,"width":0,"height":0,"name":"","label":"Fecha y hora:
","format":"{{msg.payload}}","layout":"row-
spread","x":740,"y":100,"wires":[]},{"id":"378c451.0031bba","type":"moment","z":"ef5bf662.
410aa8","name":"","topic":"","input":"","inputType":"msg","inTz":"Europe/Madrid","adjAmo
unt":0,"adjType":"days","adjDir":"add","format":"LLL","locale":"es-
ES","output":"","outputType":"msg","outTz":"Europe/Madrid","x":360,"y":100,"wires":[["1ea
cb392.06b40c"]]},{"id":"77cc46f.1ce59b8","type":"comment","z":"ef5bf662.410aa8","name":"
Impresión de la fecha y hora en la pestaña de
presentación","info":"","x":230,"y":40,"wires":[]},{"id":"a63004d1.1fe668","type":"inject","z":
"ef5bf662.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10",
"crontab":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":11
0,"y":340,"wires":[["af22b399.12614","408435c9.447fec","8fbcbdf9.549b7"]]},{"id":"af22b39
9.12614","type":"exec","z":"ef5bf662.410aa8","command":"vcgencmd
measure_temp","addpay":false,"append":"","useSpawn":"","timer":"","name":"Temperatura
RPi","x":370,"y":260,"wires":[["14f0a295.59181d"],[],[]]},{"id":"408435c9.447fec","type":"ex
ec","z":"ef5bf662.410aa8","command":"top -d 0.5 -b -n2 | grep \"Cpu(s)\"|tail -n 1 | awk '{print
$2 + $4}'","addpay":false,"append":"","useSpawn":"","timer":"","name":"Carga CPU
RPi","x":360,"y":380,"wires":[["90b76f6a.645a9","e99fdf36.2fa1c"],[],[]]},{"id":"8fbcbdf9.54
9b7","type":"exec","z":"ef5bf662.410aa8","command":"free | grep Mem | awk '{print $4/$2 *
100.0}'","addpay":false,"append":"","useSpawn":"","timer":"","name":"Memoria Libre
RPi","x":370.0000190734863,"y":504.22223567962646,"wires":[["daf6433c.bad81"],[],[]]},{"i
d":"14f0a295.59181d","type":"function","z":"ef5bf662.410aa8","name":"","func":"str =
msg.payload\nmsg.payload = str.substring(5,9);\nreturn
msg;","outputs":1,"noerr":0,"initialize":"","finalize":"","libs":[],"x":560,"y":260,"wires":[["2b1
7af38.98a11","70979b80.a31c34"]]},{"id":"daf6433c.bad81","type":"function","z":"ef5bf662.4
10aa8","name":"","func":"str = msg.payload\nmsg.payload = parseInt(str.substring(0,str.length -
1));\nreturn
msg;","outputs":1,"noerr":0,"initialize":"","finalize":"","libs":[],"x":560,"y":500,"wires":[["9b2
c02d8.2d88e","51d1136a.46e06c","611c14bb.4290ac"]]},{"id":"2b17af38.98a11","type":"ui_g
auge","z":"ef5bf662.410aa8","name":"","group":"c2ddd6a2.0e2a08","order":2,"width":0,"heig
ht":0,"gtype":"gage","title":"Temperatura
RPi","label":"oC","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600",
"#ca3838"],"seg1":"","seg2":"","x":750,"y":240,"wires":[]},{"id":"90b76f6a.645a9","type":"ui
_gauge","z":"ef5bf662.410aa8","name":"","group":"c2ddd6a2.0e2a08","order":3,"width":0,"he
ight":0,"gtype":"gage","title":"Carga
CPU","label":"%","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600",
"#ca3838"],"seg1":"","seg2":"","x":730,"y":360,"wires":[]},{"id":"9b2c02d8.2d88e","type":"ui
_gauge","z":"ef5bf662.410aa8","name":"","group":"c2ddd6a2.0e2a08","order":4,"width":0,"he
ight":0,"gtype":"gage","title":"Memoria Libre
RPi","label":"%","format":"{{value}}","min":0,"max":"100","colors":["#ff0000","#e6e600","#
04ff00"],"seg1":"","seg2":"","x":750,"y":480,"wires":[]},{"id":"9e966127.a8985","type":"com
ment","z":"ef5bf662.410aa8","name":"Monitorización variables RPi en la pestaña de
presentación","info":"","x":230,"y":200,"wires":[]},{"id":"b535689f.551c98","type":"comment
","z":"ef5bf662.410aa8","name":"Recepción de los datos ESP32 por MQTT y representación
gráfica","info":"","x":260,"y":1120,"wires":[]},{"id":"5e94ed3e.3ae154","type":"comment","z"
:"ef5bf662.410aa8","name":"Exportación a ficheros

.csv","info":"","x":970,"y":200,"wires":[]},{"id":"51d1136a.46e06c","type":"change","z":"ef5b
f662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"3","tot":"str"}],"a
ction":"","property":"","from":"","to":"","reg":false,"x":950,"y":540,"wires":[["a15d5a78.4530
e8"]]},{"id":"e99fdf36.2fa1c","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"
set","p":"parts.index","pt":"msg","to":"2","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"
$number($trim(payload))","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":fal
se,"x":940,"y":400,"wires":[["a15d5a78.4530e8"]]},{"id":"70979b80.a31c34","type":"change",
"z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"1","tot":
"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"
","property":"","from":"","to":"","reg":false,"x":940,"y":300,"wires":[["a15d5a78.4530e8"]]},{
"id":"611c14bb.4290ac","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","
p":"payload","pt":"msg","to":"","tot":"date"},{"t":"set","p":"parts.index","pt":"msg","to":"0","t
ot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":940,"y":600,"wires":[["a1
5d5a78.4530e8"]]},{"id":"a15d5a78.4530e8","type":"change","z":"ef5bf662.410aa8","name":"
id de grupo y
recuento","rules":[{"t":"set","p":"parts.id","pt":"msg","to":"grupo1","tot":"str"},{"t":"set","p":"
parts.count","pt":"msg","to":"4","tot":"str"}],"action":"","property":"","from":"","to":"","reg":f
alse,"x":1180,"y":440,"wires":[["bb63012c.b602b"]]},{"id":"bb63012c.b602b","type":"join","z
":"ef5bf662.410aa8","name":"","mode":"auto","build":"string","property":"payload","property
Type":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","c
ount":"","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixup"
:"","x":1170,"y":520,"wires":[["9e603c67.9f9c8"]]},{"id":"9e603c67.9f9c8","type":"csv","z":"
ef5bf662.410aa8","name":"","sep":",","hdrin":"","hdrout":"none","multi":"one","ret":"\\n","tem
p":"","skip":"0","strings":true,"include_empty_strings":"","include_null_values":"","x":1170,"y
":580,"wires":[["9aee3654.4dfae8"]]},{"id":"9aee3654.4dfae8","type":"file","z":"ef5bf662.410
aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_presentacion_rpi.csv","a
ppendNewline":true,"createDir":false,"overwriteFile":"false","encoding":"none","x":1350,"y":
640,"wires":[[]]},{"id":"9cd979f2.128e38","type":"debug","z":"ef5bf662.410aa8","name":"","a
ctive":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","s
tatusType":"auto","x":1350,"y":1060,"wires":[]},{"id":"421330b5.bc92d","type":"inject","z":"e
f5bf662.410aa8","name":"","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"on
ceDelay":0.1,"topic":"","payload":"timestamp (epoch), Temperatura (oC), CPU (%), Memoria
Libre
(%)","payloadType":"str","x":930,"y":780,"wires":[["9a017bd6.1aeb58"]]},{"id":"9a017bd6.1a
eb58","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_
TFM/datos_presentacion_rpi.csv","appendNewline":true,"createDir":false,"overwriteFile":"tru
e","encoding":"none","x":1310,"y":780,"wires":[[]]},{"id":"a056ff6a.7a49b","type":"comment
","z":"ef5bf662.410aa8","name":"Sobreescritura y creación cabecera fichero
.csv","info":"","x":1040,"y":720,"wires":[]},{"id":"8ea5bb50.6df588","type":"comment","z":"e
f5bf662.410aa8","name":"Lectura e impresión en pantalla del fichero
.csv","info":"","x":1040,"y":940,"wires":[]},{"id":"891fc815.4e26d8","type":"inject","z":"ef5b
f662.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10","cront
ab":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":950,"y":
1000,"wires":[["e9a29b04.2eb638"]]},{"id":"e9a29b04.2eb638","type":"file
in","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_pre
sentacion_rpi.csv","format":"utf8","chunk":false,"sendError":false,"encoding":"none","x":1310
,"y":1000,"wires":[["9cd979f2.128e38"]]},{"id":"3eafa664.e8cdca","type":"ui_button","z":"ef5
bf662.410aa8","name":"","group":"c2ddd6a2.0e2a08","order":4,"width":0,"height":0,"passthru
":false,"label":"Inicializar fichero .csv
RPi","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"pa
yload","topicType":"msg","x":390,"y":740,"wires":[["e6a9ac40.26949"]]},{"id":"e6a9ac40.269
49","type":"ui_toast","z":"ef5bf662.410aa8","position":"dialog","displayTime":"3","highlight":
"","sendall":true,"outputs":1,"ok":"Aceptar","cancel":"Cancelar","raw":false,"topic":"¿Acepta

inicializar fichero? Se sobreescribirá el fichero anterior.","name":"Inicialización fichero .csv","x":510,"y":800,"wires":[["f22879d1.2352e8"]]},{"id":"f22879d1.2352e8","type":"switch","z":"ef5bf662.410aa8","name":"Inyecta cabecera","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Aceptar","vt":"str"}],"checkall":"true","repair":false,"outputs":1,"x":590,"y":880,"wires":[["d47ac8ce.0d1e08"]]},{"id":"d47ac8ce.0d1e08","type":"change","z":"ef5bf662.410aa8","name":"Cabecera","rules":[{"t":"set","p":"payload","pt":"msg","to":"timestamp (epoch), Temperatura (oC), CPU (%), Memoria Libre (%)","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":780,"y":880,"wires":[["9a017bd6.1aeb58"]]},{"id":"598efe41.49ec9","type":"comment","z":"ef5bf662.410aa8","name":"Botón inicialización RPi en dashboard","info":"","x":430,"y":680,"wires":[]},{"id":"4b83f31b.08af8c","type":"comment","z":"ef5bf662.410aa8","name":"Botón vaciado de gráficas RPi","info":"","x":300,"y":940,"wires":[]},{"id":"ce0f3691.281ac8","type":"ui_button","z":"ef5bf662.410aa8","name":"Vaciado de gráficas RPi","group":"c2ddd6a2.0e2a08","order":5,"width":0,"height":0,"passthru":false,"label":"Vaciado de gráficas RPi","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":290,"y":1000,"wires":[["4eb02ca1.645744"]]},{"id":"4eb02ca1.645744","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"[]","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":600,"y":1000,"wires":[["17879e06.e228e2"]]},{"id":"17879e06.e228e2","type":"link out","z":"ef5bf662.410aa8","name":"Vaciado de gráficas RPi","links":["7d73a471.02641c","bfadcd2b.e895a","f60c6df5.cd408"],"x":725,"y":1000,"wires":[]},{"id":"bfadcd2b.e895a","type":"link in","z":"ef5bf662.410aa8","name":"Vaciado de gráficas RPi","links":["17879e06.e228e2"],"x":595,"y":200,"wires":[["2b17af38.98a11"]]},{"id":"f60c6df5.cd408","type":"link in","z":"ef5bf662.410aa8","name":"Vaciado de gráficas RPi","links":["17879e06.e228e2"],"x":595,"y":320,"wires":[["90b76f6a.645a9"]]},{"id":"7d73a471.02641c","type":"link in","z":"ef5bf662.410aa8","name":"Vaciado de gráficas RPi","links":["17879e06.e228e2"],"x":595,"y":440,"wires":[["9b2c02d8.2d88e"]]},{"id":"e8a4cdf3.49f6c","type":"comment","z":"ef5bf662.410aa8","name":"Corrientes - Sensor ACS","info":"","x":170,"y":1180,"wires":[]},{"id":"ad96a683.4be768","type":"ui_button","z":"ef5bf662.410aa8","name":"Vaciado de gráficas ACS","group":"7500e219.92f82c","order":4,"width":0,"height":0,"passthru":false,"label":"Vaciado de gráficas ACS","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":290,"y":1320,"wires":[["2d73f000.083ce"]]},{"id":"2d73f000.083ce","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"[]","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":530,"y":1320,"wires":[["6c0cf0d0.dd338","8c36511.2d204b","8a05c5ef.9843a8","e4b671e6.1e71c"]]},{"id":"bd2c9e9d.45cc3","type":"debug","z":"ef5bf662.410aa8","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","statusType":"auto","x":1390,"y":1780,"wires":[]},{"id":"d68e3eba.a8d9d","type":"comment","z":"ef5bf662.410aa8","name":"Lectura e impresión en pantalla del fichero .csv","info":"","x":1080,"y":1660,"wires":[]},{"id":"22272a93.e95776","type":"inject","z":"ef5bf662.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10","crontab":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":990,"y":1720,"wires":[["63341d2f.d7b294"]]},{"id":"63341d2f.d7b294","type":"file in","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_ACS.csv","format":"utf8","chunk":false,"sendError":false,"encoding":"none","x":1310,"y":1720,"wires":[["bd2c9e9d.45cc3"]]},{"id":"6ef0b9ff.2c61c8","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"1","tot":"str"},{"t":"set","p":"p

ayload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":1000,"y":1280,"wires":[["d04da865.658928"]]},{"id":"a0807760.aebc28","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"2","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":1000,"y":1320,"wires":[["d04da865.658928"]]},{"id":"e7969eb9.21c8a","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"","tot":"date"},{"t":"set","p":"parts.index","pt":"msg","to":"0","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":1000,"y":1360,"wires":[["d04da865.658928"]]},{"id":"d04da865.658928","type":"change","z":"ef5bf662.410aa8","name":"id de grupo y recuento","rules":[{"t":"set","p":"parts.id","pt":"msg","to":"grupo2","tot":"str"},{"t":"set","p":"parts.count","pt":"msg","to":"3","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":1240,"y":1320,"wires":[["dc466d44.4cccf"]]},{"id":"dc466d44.4cccf","type":"join","z":"ef5bf662.410aa8","name":"","mode":"auto","build":"string","property":"payload","propertyType":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","count":"","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixup":"","x":1290,"y":1380,"wires":[["a6c504cb.1784f8"]]},{"id":"a6c504cb.1784f8","type":"csv","z":"ef5bf662.410aa8","name":"","sep":",","hdrin":"","hdrout":"none","multi":"one","ret":"\\n","temp":"","skip":"0","strings":true,"include_empty_strings":"","include_null_values":"","x":1290,"y":1440,"wires":[["f3a4ef78.3543b"]]},{"id":"f3a4ef78.3543b","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_ACS.csv","appendNewline":true,"createDir":false,"overwriteFile":"false","encoding":"none","x":1430,"y":1500,"wires":[[]]},{"id":"acaf9681.4a7da8","type":"inject","z":"ef5bf662.410aa8","name":"","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":0.1,"topic":"","payload":"timestamp (epoch), Corriente panel (A), Corriente batería (A)","payloadType":"str","x":810,"y":1600,"wires":[["c9e51566.5b5618"]]},{"id":"c9e51566.5b5618","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_ACS.csv","appendNewline":true,"createDir":false,"overwriteFile":"true","encoding":"none","x":1150,"y":1600,"wires":[[]]},{"id":"29754499.56768c","type":"comment","z":"ef5bf662.410aa8","name":"Sobreescritura y creación cabecera fichero .csv","info":"","x":920,"y":1540,"wires":[]},{"id":"ce7b0e5e.07e4b","type":"ui_button","z":"ef5bf662.410aa8","name":"","group":"7500e219.92f82c","order":3,"width":0,"height":0,"passthru":false,"label":"Inicializar fichero .csv ACS","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"payload","topicType":"msg","x":280,"y":1560,"wires":[["9b19def8.94b53"]]},{"id":"9b19def8.94b53","type":"ui_toast","z":"ef5bf662.410aa8","position":"dialog","displayTime":"3","highlight":"","sendall":true,"outputs":1,"ok":"Aceptar","cancel":"Cancelar","raw":false,"topic":"¿Acepta inicializar fichero? Se sobreescribirá el fichero anterior.","name":"Inicialización fichero .csv","x":390,"y":1620,"wires":[["dffc0515.5e0238"]]},{"id":"dffc0515.5e0238","type":"switch","z":"ef5bf662.410aa8","name":"Inyecta cabecera","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Aceptar","vt":"str"}],"checkall":"true","repair":false,"outputs":1,"x":470,"y":1700,"wires":[["94bfcba1.1fa448"]]},{"id":"94bfcba1.1fa448","type":"change","z":"ef5bf662.410aa8","name":"Cabecera","rules":[{"t":"set","p":"payload","pt":"msg","to":"timestamp (epoch), Corriente panel (A), Corriente batería (A)","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":660,"y":1700,"wires":[["c9e51566.5b5618"]]},{"id":"d472133e.8f7e","type":"comment","z":"ef5bf662.410aa8","name":"Botón inicialización ACS en dashboard","info":"","x":310,"y":1500,"wires":[]},{"id":"2eeea5b2.55e57a","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"1","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":1000,"y":1860,"wires":[["6ade2b68.31ef44"]]},{"id":"85e54e0d.dcb07","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload

","pt":"msg","to":"","tot":"date"},{"t":"set","p":"parts.index","pt":"msg","to":"0","tot":"jsonata
"}],"action":"","property":"","from":"","to":"","reg":false,"x":1000,"y":1940,"wires":[["6ade2b
68.31ef44"]]},{"id":"6ade2b68.31ef44","type":"change","z":"ef5bf662.410aa8","name":"id de
grupo y
recuento","rules":[{"t":"set","p":"parts.id","pt":"msg","to":"grupo3","tot":"str"},{"t":"set","p":"
parts.count","pt":"msg","to":"2","tot":"str"}],"action":"","property":"","from":"","to":"","reg":f
alse,"x":1240,"y":1900,"wires":[["1b82538e.79f79c"]]},{"id":"1b82538e.79f79c","type":"join"
,"z":"ef5bf662.410aa8","name":"","mode":"auto","build":"string","property":"payload","proper
tyType":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","
count":"","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixu
p":"","x":1290,"y":1960,"wires":[["f090a4cf.466d78"]]},{"id":"f090a4cf.466d78","type":"csv",
"z":"ef5bf662.410aa8","name":"","sep":",","hdrin":"","hdrout":"none","multi":"one","ret":"\\n",
"temp":"","skip":"0","strings":true,"include_empty_strings":"","include_null_values":"","x":12
90,"y":2020,"wires":[["2a8d25de.f9b5fa"]]},{"id":"2a8d25de.f9b5fa","type":"file","z":"ef5bf6
62.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_FZ.csv","appendN
ewline":true,"createDir":false,"overwriteFile":"false","encoding":"none","x":1420,"y":2080,"w
ires":[[]]},{"id":"15f0e5d1.d1feca","type":"ui_button","z":"ef5bf662.410aa8","name":"Vaciado
de gráficas
FZ","group":"44a34dca.01f0a4","order":4,"width":0,"height":0,"passthru":false,"label":"Vacia
do de gráficas
FZ","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"top
ic","topicType":"msg","x":300,"y":1980,"wires":[["35bc5e76.3adc72"]]},{"id":"35bc5e76.3adc
72","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"m
sg","to":"[]","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":500,"y"
:1980,"wires":[["f58ecbca.fef008","83995526.628b18"]]},{"id":"a424c4de.84bb98","type":"de
bug","z":"ef5bf662.410aa8","name":"","active":false,"tosidebar":true,"console":false,"tostatus"
:false,"complete":"false","statusVal":"","statusType":"auto","x":1390,"y":2340,"wires":[]},{"id
":"cbe6e6a2.13d4e8","type":"comment","z":"ef5bf662.410aa8","name":"Lectura e impresión en
pantalla del fichero
.csv","info":"","x":1080,"y":2220,"wires":[]},{"id":"be070fa3.cc2d","type":"inject","z":"ef5bf6
62.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10","cronta
b":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":990,"y":2
280,"wires":[["37f98284.f8a92e"]]},{"id":"37f98284.f8a92e","type":"file
in","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_FZ.
csv","format":"utf8","chunk":false,"sendError":false,"encoding":"none","x":1300,"y":2280,"wi
res":[["a424c4de.84bb98"]]},{"id":"446fc79b.ddacc8","type":"inject","z":"ef5bf662.410aa8","
name":"","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":0.1,"top
ic":"","payload":"timestamp (epoch), Tensión batería
(V)","payloadType":"str","x":810,"y":2160,"wires":[["45ddea6d.514f64"]]},{"id":"45ddea6d.5
14f64","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv
_TFM/datos_FZ.csv","appendNewline":true,"createDir":false,"overwriteFile":"true","encoding
":"none","x":1140,"y":2160,"wires":[[]]},{"id":"b1da8761.1f7a28","type":"comment","z":"ef5
bf662.410aa8","name":"Sobreescritura y creación cabecera fichero
.csv","info":"","x":920,"y":2100,"wires":[]},{"id":"1fec73d8.b6dd6c","type":"ui_button","z":"e
f5bf662.410aa8","name":"","group":"44a34dca.01f0a4","order":3,"width":0,"height":0,"passthr
u":false,"label":"Inicializar fichero .csv
FZ","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"pa
yload","topicType":"msg","x":270,"y":2120,"wires":[["6683c992.0450a8"]]},{"id":"6683c992.
0450a8","type":"ui_toast","z":"ef5bf662.410aa8","position":"dialog","displayTime":"3","highli
ght":"","sendall":true,"outputs":1,"ok":"Aceptar","cancel":"Cancelar","raw":false,"topic":"¿Ac
epta inicializar fichero? Se sobreescribirá el fichero anterior.","name":"Inicialización fichero
.csv","x":390,"y":2180,"wires":[["e8b08b23.d56b08"]]},{"id":"e8b08b23.d56b08","type":"swit
ch","z":"ef5bf662.410aa8","name":"Inyecta

cabecera","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Aceptar","vt":"str"}],"checkall":"true","repair":false,"outputs":1,"x":470,"y":2260,"wires":[["2322575a.ff8338"]]},{"id":"2322575a.ff8338","type":"change","z":"ef5bf662.410aa8","name":"Cabecera","rules":[{"t":"set","p":"payload","pt":"msg","to":"timestamp (epoch), Tensión Batería (V)","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":660,"y":2260,"wires":[["45ddea6d.514f64"]]},{"id":"e3368251.65415","type":"comment","z":"ef5bf662.410aa8","name":"Botón inicialización FZ en dashboard","info":"","x":300,"y":2060,"wires":[]},{"id":"6133da46.617004","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"1","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":960,"y":2480,"wires":[["e0968ab7.511538"]]},{"id":"1ba13b4f.abe355","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"2","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":960,"y":2520,"wires":[["e0968ab7.511538"]]},{"id":"a0af39b4.db1ca8","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"","tot":"date"},{"t":"set","p":"parts.index","pt":"msg","to":"0","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":960,"y":2560,"wires":[["e0968ab7.511538"]]},{"id":"e0968ab7.511538","type":"change","z":"ef5bf662.410aa8","name":"id de grupo y recuento","rules":[{"t":"set","p":"parts.id","pt":"msg","to":"grupo4","tot":"str"},{"t":"set","p":"parts.count","pt":"msg","to":"3","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":1200,"y":2520,"wires":[["856b3b8b.1fe618"]]},{"id":"856b3b8b.1fe618","type":"join","z":"ef5bf662.410aa8","name":"","mode":"auto","build":"string","property":"payload","propertyType":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","count":"","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixup":"","x":1250,"y":2580,"wires":[["ca6de15f.56579"]]},{"id":"ca6de15f.56579","type":"csv","z":"ef5bf662.410aa8","name":"","sep":",","hdrin":"","hdrout":"none","multi":"one","ret":"\\n","temp":"","skip":"0","strings":true,"include_empty_strings":"","include_null_values":"","x":1250,"y":2640,"wires":[["6cd91248.53052c"]]},{"id":"6cd91248.53052c","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_DHT.csv","appendNewline":true,"createDir":false,"overwriteFile":"false","encoding":"none","x":1390,"y":2700,"wires":[[]]},{"id":"2e20beb1.901712","type":"ui_button","z":"ef5bf662.410aa8","name":"Vaciado de gráficas DHT","group":"e4e86617.75cda8","order":4,"width":0,"height":0,"passthru":false,"label":"Vaciado de gráficas DHT","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":310,"y":2520,"wires":[["d56bf478.e168b8"]]},{"id":"d56bf478.e168b8","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"[]","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":540,"y":2520,"wires":[["dbb5d54c.09ce78","8a82de3.c789a2","ec5d0d3.fdb45f","82ffed9a.b8983"]]},{"id":"1e958ec7.288681","type":"debug","z":"ef5bf662.410aa8","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","statusType":"auto","x":1390,"y":2980,"wires":[]},{"id":"79d338ff.096308","type":"comment","z":"ef5bf662.410aa8","name":"Lectura e impresión en pantalla del fichero .csv","info":"","x":1080,"y":2860,"wires":[]},{"id":"10a96cd.971bf93","type":"inject","z":"ef5bf662.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10","crontab":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":990,"y":2920,"wires":[["de881f4c.c492"]]},{"id":"de881f4c.c492","type":"file in","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_DHT.csv","format":"utf8","chunk":false,"sendError":false,"encoding":"none","x":1310,"y":2920,"wires":[["1e958ec7.288681"]]},{"id":"e0d1267e.c6aee8","type":"inject","z":"ef5bf662.410aa8","name":"","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":0.1,"topic":"","payload":"timestamp (epoch), Humedad (%), Temperatura

(oC)","payloadType":"str","x":810,"y":2800,"wires":[["13a5865c.e59b6a"]]},{"id":"13a5865c.e59b6a","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_DHT.csv","appendNewline":true,"createDir":false,"overwriteFile":"true","encoding":"none","x":1150,"y":2800,"wires":[[]]},{"id":"1ac0efd4.4daba","type":"comment","z":"ef5bf662.410aa8","name":"Sobreescritura y creación cabecera fichero .csv","info":"","x":920,"y":2740,"wires":[]},{"id":"e21de9d7.0eebe8","type":"ui_button","z":"ef5bf662.410aa8","name":"","group":"e4e86617.75cda8","order":3,"width":0,"height":0,"passthru":false,"label":"Inicializar fichero .csv DHT","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"payload","topicType":"msg","x":280,"y":2760,"wires":[["696fe123.fbc7"]]},{"id":"696fe123.fbc7","type":"ui_toast","z":"ef5bf662.410aa8","position":"dialog","displayTime":"3","highlight":"","sendall":true,"outputs":1,"ok":"Aceptar","cancel":"Cancelar","raw":false,"topic":"¿Acepta inicializar fichero? Se sobreescribirá el fichero anterior.","name":"Inicialización fichero .csv","x":390,"y":2820,"wires":[["224d7b53.5c1e84"]]},{"id":"224d7b53.5c1e84","type":"switch","z":"ef5bf662.410aa8","name":"Inyecta cabecera","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Aceptar","vt":"str"}],"checkall":"true","repair":false,"outputs":1,"x":470,"y":2900,"wires":[["aab807c3.8ab848"]]},{"id":"aab807c3.8ab848","type":"change","z":"ef5bf662.410aa8","name":"Cabecera","rules":[{"t":"set","p":"payload","pt":"msg","to":"timestamp (epoch), Humedad (%), Temperatura (oC)","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":660,"y":2900,"wires":[["13a5865c.e59b6a"]]},{"id":"acef8648.888d88","type":"comment","z":"ef5bf662.410aa8","name":"Botón inicialización FZ en dashboard","info":"","x":300,"y":2700,"wires":[]},{"id":"e3d61834.181458","type":"ui_button","z":"ef5bf662.410aa8","name":"Vaciado de gráficas BH","group":"d69ee820.38cd88","order":4,"width":0,"height":0,"passthru":false,"label":"Vaciado de gráficas BH","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":290,"y":3160,"wires":[["9ced4954.866ab8"]]},{"id":"9ced4954.866ab8","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"[]","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":520,"y":3160,"wires":[["df3fe49c.cd33b8","8778fe30.db0c6"]]},{"id":"2dd455e.cf883aa","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"parts.index","pt":"msg","to":"1","tot":"str"},{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":980,"y":3060,"wires":[["6ae7866b.d1f9d8"]]},{"id":"5da49c2e.18a9d4","type":"change","z":"ef5bf662.410aa8","name":"","rules":[{"t":"set","p":"payload","pt":"msg","to":"","tot":"date"},{"t":"set","p":"parts.index","pt":"msg","to":"0","tot":"jsonata"}],"action":"","property":"","from":"","to":"","reg":false,"x":980,"y":3140,"wires":[["6ae7866b.d1f9d8"]]},{"id":"6ae7866b.d1f9d8","type":"change","z":"ef5bf662.410aa8","name":"id de grupo y recuento","rules":[{"t":"set","p":"parts.id","pt":"msg","to":"grupo5","tot":"str"},{"t":"set","p":"parts.count","pt":"msg","to":"2","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":1220,"y":3100,"wires":[["c51b99e4.ec1498"]]},{"id":"c51b99e4.ec1498","type":"join","z":"ef5bf662.410aa8","name":"","mode":"auto","build":"string","property":"payload","propertyType":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","count":"","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixup":"","x":1270,"y":3160,"wires":[["58c603ef.1bd14c"]]},{"id":"58c603ef.1bd14c","type":"csv","z":"ef5bf662.410aa8","name":"","sep":",","hdrin":"","hdrout":"none","multi":"one","ret":"\\n","temp":"","skip":"0","strings":true,"include_empty_strings":"","include_null_values":"","x":1270,"y":3220,"wires":[["4252fbbf.dc0b94"]]},{"id":"4252fbbf.dc0b94","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_BH.csv","appendNewline":true,"createDir":false,"overwriteFile":"false","encoding":"none","x":1400,"y":3280,"wires":[[]]},{"id":"fab638cc.698608","type":"debug","z":"ef5bf662.410aa8","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","status

Type":"auto","x":1410,"y":3560,"wires":[]},{"id":"ccc829ab.df8118","type":"comment","z":"e
f5bf662.410aa8","name":"Lectura e impresión en pantalla del fichero
.csv","info":"","x":1100,"y":3440,"wires":[]},{"id":"a6972f22.2bb36","type":"inject","z":"ef5b
f662.410aa8","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"10","cront
ab":"","once":false,"onceDelay":0.1,"topic":"","payload":"","payloadType":"date","x":1010,"y"
:3500,"wires":[["210db497.84007c"]]},{"id":"210db497.84007c","type":"file
in","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv_TFM/datos_BH
.csv","format":"utf8","chunk":false,"sendError":false,"encoding":"none","x":1320,"y":3500,"wi
res":[["fab638cc.698608"]]},{"id":"94bb9026.add51","type":"inject","z":"ef5bf662.410aa8","n
ame":"","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":0.1,"topic
":"","payload":"timestamp (epoch), Iluminancia
(Lux)","payloadType":"str","x":830,"y":3380,"wires":[["e91c10b9.6bc1d"]]},{"id":"e91c10b9.
6bc1d","type":"file","z":"ef5bf662.410aa8","name":"","filename":"/home/pi/Desktop/datos_csv
_TFM/datos_BH.csv","appendNewline":true,"createDir":false,"overwriteFile":"true","encoding
":"none","x":1160,"y":3380,"wires":[[]]},{"id":"cd678761.b30a98","type":"comment","z":"ef5
bf662.410aa8","name":"Sobreescritura y creación cabecera fichero
.csv","info":"","x":940,"y":3320,"wires":[]},{"id":"80e69749.b93e98","type":"ui_button","z":"
ef5bf662.410aa8","name":"","group":"d69ee820.38cd88","order":3,"width":0,"height":0,"passt
hru":false,"label":"Inicializar fichero .csv
BH","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"pa
yload","topicType":"msg","x":290,"y":3340,"wires":[["9a31fdaf.c764c"]]},{"id":"9a31fdaf.c76
4c","type":"ui_toast","z":"ef5bf662.410aa8","position":"dialog","displayTime":"3","highlight":
"","sendall":true,"outputs":1,"ok":"Aceptar","cancel":"Cancelar","raw":false,"topic":"¿Acepta
inicializar fichero? Se sobreescribirá el fichero anterior.","name":"Inicialización fichero
.csv","x":410,"y":3400,"wires":[["85162b3f.fd82f8"]]},{"id":"85162b3f.fd82f8","type":"switch
","z":"ef5bf662.410aa8","name":"Inyecta
cabecera","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Aceptar","vt":"st
r"}],"checkall":"true","repair":false,"outputs":1,"x":490,"y":3480,"wires":[["8d779325.131c7"]
]},{"id":"8d779325.131c7","type":"change","z":"ef5bf662.410aa8","name":"Cabecera","rules":
[{"t":"set","p":"payload","pt":"msg","to":"timestamp (epoch), Iluminancia
(Lux)","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":680,"y":3480,"wi
res":[["e91c10b9.6bc1d"]]},{"id":"6340cad6.dface4","type":"comment","z":"ef5bf662.410aa8"
,"name":"Botón inicialización BH en dashboard
","info":"","x":330,"y":3280,"wires":[]},{"id":"83995526.628b18","type":"ui_chart","z":"ef5bf
662.410aa8","name":"","group":"44a34dca.01f0a4","order":2,"width":0,"height":0,"label":"His
tórico tensión
batería","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nod
ata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOld
erUnit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7e8",
"#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"useD
ifferentColor":false,"x":790,"y":1840,"wires":[[]]},{"id":"8a05c5ef.9843a8","type":"ui_gauge",
"z":"ef5bf662.410aa8","name":"","group":"9404bef5.5e24b","order":1,"width":0,"height":0,"gt
ype":"gage","title":"Corriente
panel","label":"A","format":"{{value}}","min":0,"max":"30","colors":["#00b500","#e6e600","
#ca3838"],"seg1":"","seg2":"","x":770,"y":1240,"wires":[]},{"id":"e4b671e6.1e71c","type":"ui
_gauge","z":"ef5bf662.410aa8","name":"","group":"7500e219.92f82c","order":1,"width":0,"hei
ght":0,"gtype":"gage","title":"Corriente
Batería","label":"A","format":"{{value}}","min":0,"max":"30","colors":["#00b500","#e6e600"
,"#ca3838"],"seg1":"","seg2":"","x":770,"y":1400,"wires":[]},{"id":"ec5d0d3.fdb45f","type":"u
i_chart","z":"ef5bf662.410aa8","name":"","group":"15e0cb08.d66905","order":2,"width":0,"he
ight":0,"label":"Histórico
humedad","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","n
odata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeO

lderUnit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7e
8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"us
eDifferentColor":false,"x":770,"y":2400,"wires":[[]]},{"id":"82ffed9a.b8983","type":"ui_chart"
,"z":"ef5bf662.410aa8","name":"","group":"e4e86617.75cda8","order":2,"width":0,"height":0,"
label":"Histórico
temperatura","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear",
"nodata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","remove
OlderUnit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7
e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"u
seDifferentColor":false,"x":780,"y":2640,"wires":[[]]},{"id":"8778fe30.db0c6","type":"ui_char
t","z":"ef5bf662.410aa8","name":"","group":"d69ee820.38cd88","order":2,"width":0,"height":0
,"label":"Histórico
iluminancia","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear",
"nodata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","remove
OlderUnit":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7
e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"outputs":1,"u
seDifferentColor":false,"x":780,"y":3020,"wires":[[]]},{"id":"2a091142.00b8ce","type":"mqtt-
broker","name":"Mosquitto","broker":"192.168.0.10","port":"1883","clientid":"","usetls":false,
"protocolVersion":"4","keepalive":"60","cleansession":true,"birthTopic":"","birthQos":"0","birt
hPayload":"","birthMsg":{},"closeTopic":"","closeQos":"0","closePayload":"","closeMsg":{},"
willTopic":"","willQos":"0","willPayload":"","willMsg":{},"sessionExpiry":""},{"id":"9404bef
5.5e24b","type":"ui_group","name":"PV","tab":"884dc08e.6fb36","order":1,"disp":true,"width
":"6","collapse":false},{"id":"7500e219.92f82c","type":"ui_group","name":"BAT","tab":"884d
c08e.6fb36","order":2,"disp":true,"width":"6","collapse":false},{"id":"e4e86617.75cda8","type
":"ui_group","name":"DHT
TEMP","tab":"a01eed1d.7b846","order":2,"disp":true,"width":"6","collapse":false},{"id":"15e
0cb08.d66905","type":"ui_group","name":"DHT
HUM","tab":"a01eed1d.7b846","order":1,"disp":true,"width":"6","collapse":false},{"id":"44a3
4dca.01f0a4","type":"ui_group","name":"FZ
BAT","tab":"3b8bc467.e278dc","order":1,"disp":true,"width":"6","collapse":false},{"id":"d69e
e820.38cd88","type":"ui_group","name":"BH
LUX","tab":"134cec61.2bcf64","order":1,"disp":true,"width":"6","collapse":false},{"id":"c2dd
d6a2.0e2a08","type":"ui_group","name":"Presentación","tab":"1b11e65f.2cccca","order":1,"dis
p":true,"width":"6","collapse":false},{"id":"884dc08e.6fb36","type":"ui_tab","name":"ACS","i
con":"dashboard","order":2,"disabled":false,"hidden":false},{"id":"a01eed1d.7b846","type":"ui
_tab","name":"DHT","icon":"dashboard","order":4,"disabled":false,"hidden":false},{"id":"3b8
bc467.e278dc","type":"ui_tab","name":"FZ","icon":"dashboard","order":3,"disabled":false,"hid
den":false},{"id":"134cec61.2bcf64","type":"ui_tab","name":"BH","icon":"dashboard","order":
5,"disabled":false,"hidden":false},{"id":"1b11e65f.2cccca","type":"ui_tab","name":"Presentaci
ón","icon":"dashboard","order":1,"disabled":false,"hidden":false}]

EX NIHILO NIHIL FIT

Campus de Rabanales, Edificio Leonardo Da Vinci

14071 CÓRDOBA (España)

Teléfono 957-218373