

Máster Universitario en Ingeniería
de Telecomunicación
Curso 2022-2023

Trabajo Fin de Máster

**“Sistema de monitorización de la
salud de los bosques de quercíneas”**

Manuel Álvarez Herrera

Tutor

José Antonio García Souto

Madrid, 4 de septiembre del 2023



[Incluir en el caso del interés de su publicación en el archivo abierto]
Esta obra se encuentra sujeta a la licencia Creative Commons **Reconocimiento – No
Comercial – Sin Obra Derivada**

Título: Sistema de monitorización de la salud de los bosques de quercíneas

Autor: Manuel Álvarez Herrera

Director: José Antonio García Souto

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Trabajo Fin de Grado el día ____ de _____ del 2023 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO

PRESIDENTE

Agradecimientos

Desde este espacio quiero transmitir mi más sincero afecto, gratitud y cariño a todas las personas que me han dado fuerza y apoyo, las cuales menciono a continuación.

Quiero dar las gracias, en primer lugar, a mi tutor el Dr. José Antonio García Souto, por la forma en que me ha guiado, aconsejado, dedicándome su tiempo y facilitándome la realización de este Trabajo Fin de Máster. Gracias por toda la comprensión y empatía.

Quiero agradecer a la empresa Hydra Space Systems y mis compañeros de trabajo por las facilidades que me han proporcionado para realizar el presente proyecto.

Un especial reconocimiento a la profesión de maestro y a todos los profesionales que, con sus enseñanzas y ejemplo, me han ayudado, formándome como profesional y como ser humano.

Estoy muy agradecido con todos los compañeros del laboratorio de la UC3M que me han sabido guiar y siempre han estado dispuestos a tenderme una mano amiga. He de nombrar a Cindy Báez y Yuliy Moreno por su especial atención e inestimable apoyo.

Quiero mostrar mi gratitud a mis amigos, especial mención a Raúl y Vladimir por su gran amistad e inestimable ayuda al leer el presente TFM con infinita paciencia y aportarme su más sincera opinión, ¡un millón de gracias! Tampoco puedo olvidar a Melchior, Alejandro, Cristian, Alexander, Ulises, ... También quería mencionar a mis compañeros de la UC3M y a mis amigos Juanjo, Guille, Marcos, Javi, Delia, Tomás, Marta... A todos gracias por ser apoyo en momentos difíciles, hemos superado mil obstáculos. Gracias por transmitirme vuestra fuerza, siempre espero contar con vosotros y sabed que siempre contareis conmigo.

He de dedicar este TFM a mi prima Amanda, a mis tíos María Cruz, Amalia y Vicente, a mis abuelos y, en especial, a mis padres que me han dedicado todo su cariño, tiempo y esfuerzo con el objetivo de educarme y formarme lo mejor posible para afrontar la vida, por su paciencia, apoyo y amor incondicional que me guía siempre.

A todos, infinitas gracias.

Resumen

En el presente Trabajo Fin de Máster se ha desarrollado un sistema de monitorización para la obtención de información sobre los efectos que tiene el cambio climático sobre el estado y la salud de los bosques de quercíneas. Esto se ha realizado a través de la instrumentación con múltiples sensores de dos ejemplares de encinas y del ambiente que les rodea.

Para ello se ha realizado un estudio de las distintas magnitudes que puedan ser relevantes para la comprobación del estado de los distintos ejemplares. Se analizan, acondicionan y utilizan distintos sensores que miden las magnitudes estudiadas.

Para la recolección de datos se han analizado distintos tipos de sistemas embebidos que puedan ser útiles para este cometido. Finalmente, se han estudiado algunas técnicas de transmisión, procesamiento y almacenamiento de datos para poder realizar un análisis básico, corroborando la efectividad y buen funcionamiento del sistema implementado, a fin de que estos datos sirvan de información y sean útiles para aplicaciones posteriores.

Palabras clave: cambio climático, sensores, sistema de monitorización, magnitudes arbóreas, magnitudes climáticas, monitorización de bosques, computación frontera, Arduino, Raspberry Pi, Dragino, estación meteorológica, nodo.

Abstract

In this Master Thesis, a monitoring system we have developed to obtain information on the effects of climate change on the state and health of oak forests. This will be done through the sensorization of two specimens of holm oaks and their surrounding environment.

This has been done through the instrumentation with multiple sensors of two specimens of holm oaks and their surrounding environment. Different sensors that measure the magnitudes studied are analysed, conditioned and used.

For data collection, different types of embedded systems that can be useful for this purpose have been analysed. Finally, some data transmission, processing and storage techniques have been studied in order to perform a basic analysis, corroborating the effectiveness and proper functioning of the implemented system, so that these data can be used as information and be useful for later applications.

Keywords: climate change, sensors, monitoring system, tree magnitudes, climate magnitudes, monitoring forest, Edge Computing, Arduino, Raspberry Pi, Dragino, weather station, node.

Abreviaturas

Notación	Significado
C_2H_4	Etileno
CO_2	Dióxido de carbono
DSP	Procesador de señales digitales (Digital Signal Processor)
EC	Conductividad Eléctrica (Electrical Conductivity)
GPIO	Entrada/Salida de Propósito General (General Purpose Input / Output)
GSM	Sistema Global para las comunicaciones Móviles (Global System for Mobile communications)
HW	Hardware
I2C	Círcuito Inter-Integrado (Inter-Integrated Circuit)
IoT	Internet de las Cosas (Internet of Things)
K	Potasio
LoRa	Largo Alcance (Long Range)
N	Nitrógeno
NB	Banda Estrecha (Narrow Band)
NPK	Índice de Nitrógeno, Fósforo y Potasio
O_2	Oxígeno
ONU	Organización de las Naciones Unidas
P	Fósforo
PC	Ordenador personal (Personal Computer)
PCB	Placa de Circuito Impreso (Printed Circuit Board)
pH	Potencial de Hidrógeno (Potential Hydrogen)
PNUMA	Programa de las Naciones Unidas para el Medio Ambiente
PWM	Modulación por Ancho de Pulso (Pulse-Width Modulation)
RH	Humedad Relativa (Relative Humidity)
SCL	System Clock
SDA	System Data
SoC	Sistema en un Chip (System On a Chip)
SW	Software
TFM	Trabajo de Fin de Máster
TTL	Lógica Transistor a Transistor (Transistor-Transistor Logic)
UART	Transmisor-Receptor Asíncrono Universal (Universal Asynchronous Receiver / Transmitter)
UE	Unión Europea

USB	Bus Universal en Serie (Universal Serial Bus)
UVI	Índice de Radiación Ultravioleta (Ultra Violet Index)
μC	Microcontrolador
μP	Microprocesador

Índice general

1. Introducción.....	1
1.1. Motivación	1
1.2. Objetivos	3
1.3. Impacto	4
1.4. Fases de desarrollo y organización del trabajo	5
1.5. Organización de la memoria	8
2. Estado del arte	9
2.1. Quercus: definición, contribución a la economía y localización	9
2.1.1. Definición	9
2.1.2. Contribución a la economía	12
2.1.3. Localización	12
2.2. Cambio climático	17
2.3. Acciones para la sostenibilidad de los bosques	18
2.4. Sensores	19
2.5. Sistemas embebidos.....	21
2.6. Sistemas de monitorización	22
3. Estudio y elección de las magnitudes y de los sensores.....	25
3.1. Estudio de las magnitudes y de los sensores	25
3.2. Elección de las magnitudes y de los sensores	27
4. Implementación del primer nodo sensor.....	32
4.1. Planificación del nodo	32
4.1.1. Datos técnicos, objetivos y soluciones.....	33
4.1.2. Diseño de las posibles soluciones de acondicionamiento	38
4.1.3. Simulación de los circuitos de acondicionamiento	41
4.2. Alimentación del nodo	45
4.3. Conexión entre sensores-Arduino y los circuitos de acondicionamiento	47
4.4. Adquisición de los datos sensor-Arduino y procesado	69
4.5. Comunicación Arduino-Raspberry y almacenamiento de los datos	79
5. Implementación del segundo nodo sensor	83
5.1. Planificación del nodo	83
5.1.1. Modos de funcionamiento	84
5.2. Adquisición de los datos y módulos de conversión	95
5.3. Estudio de la recepción de los datos medidos y preprocesamiento.....	98

5.4.	Almacenamiento de los datos.....	115
5.5.	Alimentación del sistema	118
6.	Integración y resultados	120
6.1.	Entorno de trabajo e instalación del sistema.....	120
6.1.1.	Primer nodo implementado	120
6.1.2.	Segundo nodo implementado.....	122
6.2.	Resultados obtenidos.....	124
7.	Presupuestos	133
7.1.	Gastos asociados al proyecto	133
8.	Conclusiones y trabajo futuro	138
8.1.	Conclusiones.....	138
8.2.	Trabajo futuro	139
Bibliografía.....		140
Apéndice A: Listas de los sensores estudiados		146
Temperatura estomática.....	146	
Humedad de la hoja	147	
Diámetro del tallo, pecíolo y tronco	148	
Tamaño de fruto.....	149	
Sensores de Humedad	149	
Flujo de savia.....	150	
Oquedad del árbol.....	151	
Presión del jugo celular sobre la hoja	151	
Oxígeno en el suelo	152	
Oxígeno en la hoja.....	153	
Potencial hidráulico.....	153	
Dióxido de carbono en la hoja	154	
Dióxido de carbono en el suelo	155	
NPK.....	156	
pH	157	
Humedad del suelo	158	
Estación meteorológica.....	159	
Microcontroladores.....	160	
Apéndice B: Característica de los sensores y propuesta de selección		161
Sensores de Temperatura de Estoma	161	
LT-1M	161	
LT-2M	162	
LT-IRM	162	

LS-40	163
LAT-B3	164
SHT1x.....	165
Sensores de Humedad	166
HM102.....	166
RK300-04	166
Leaf Wetness Sensor de Darrera.....	167
PESSL	167
NMETOS200	168
LWS-L.....	168
HD3901.....	169
PYTHOS31.....	169
Flujo de savia.....	170
SF-G	170
SGEX-16, 29 y 25.....	171
FLOW32A-1k.....	172
SF-L	173
HFD8-100 / HFD8-50	173
Oxígeno en el suelo	173
Oxígeno en el aire/hoja.....	173
SEN0322	174
EZO-O2	174
ME2-02	175
MIX8410-O2	175
Dióxido de carbono en el suelo.....	176
Dióxido de Carbono en hoja/aire	176
SEN0159	177
CO2 Gas Sensor	178
EZO-CO2	178
RK300-03B	179
Wireless Carbon Dioxide CO2 Sensor.....	180
MH-Z19C.....	180
NPK	180
Soil NPK sensor.....	180
Teralytic.....	181
EVTSCAN.....	181
Soil NPK sensor Revalcon	181

Sensor de pH	181
SEN-10972	182
Medidor de pH del suelo 3 en 1	182
SEN0249	183
101990792.....	184
314990620.....	184
PIM520	185
28092.....	185
Estaciones meteorológicas.....	185
Estación meteorológica WLAN Sainlogic.....	186
Bresser Wetterstation Funk	187
Estación meteorológica Sainlogic profi WLAN	187
Sainlogic WS3500	188
Estación meteorológica inalámbrica froggit WH6000 Pro	190
Froggit WH3600 Funk.....	190
Froggit WH3000 SE.....	191
Apéndice C: Códigos de Matlab de las alternativas	192
LAT-B3	192
LT-1M	194
LWS-L.....	195
PYTHOS31.....	195
314990620.....	196
SF-G	197
Apéndice D: Simulación de las alternativas de acondicionamiento	199
Acondicionamiento de los sensores de temperatura de estoma	199
LAT-B3	199
LT-1M	201
Acondicionamiento de los sensores de humedad de hoja	203
PYTHOS-31.....	203
LWS-L.....	206
Acondicionamiento de los sensores de humedad de suelo.....	209
314990620.....	209
Acondicionamiento de los sensores de flujo de savia	211
SF-G	211
Apéndice E: Código Arduino del primer nodo sensor.....	212
Primer modo: Toma de datos continua	212

Primer modo: Toma de datos continua, pero solamente toma cuando le pide la petición la Raspberry Pi	214
Segundo modo: Toma de datos periódico, media winsorizada	217
Segundo modo: Toma de datos periódico, media recortada	220
Apéndice F: Código Raspberry Pi del primer nodo sensor	225
Apéndice G: Comandos AT de Dragino	227
Apéndice H: Código del segundo nodo sensor.....	228
sensores.c.....	228
sensores.h	247
Apéndice I: Formato datos de la estación meteorológica	249

Índice de figuras

Figura 1.2.1 Encina	3
Figura 2.1.1 Fósiles de hojas de quercíneas	9
Figura 2.1.2 Flores masculinas (izquierda) y femeninas (derecha) de la encina	10
Figura 2.1.3 Tronco agrietado y oscurecido de encina adulta	11
Figura 2.1.4 Encina Terrona, la más antigua de la península Ibérica y probablemente de Europa, se localiza en Zarza de Montánchez (Cáceres)	12
Figura 2.1.5 Distribución del <i>quercus robur</i>	13
Figura 2.1.6 Distribución del <i>quercus pirenaica</i> o roble negro	13
Figura 2.1.7 Distribución <i>quercus faginea</i> , conocido como, quejigo, roble carrasqueño	14
Figura 2.1.8 Distribución <i>quercus canariensis</i> conocido como, roble andaluz o quejigo andaluz	14
Figura 2.1.9 Distribución <i>quercus ilex</i> o encina	15
Figura 2.1.10 Distribución <i>quercus suber</i> o alcornoque	15
Figura 2.1.11 Distribución de subespecies <i>quercus ilex</i> por Europa	16
Figura 2.1.12 Distribución <i>quercus petraea</i> por Europa.....	16
Figura 2.3.1 Defoliación media en las parcelas de España (España 2016) y Europa (España 2001).....	19
Figura 2.6.1 Ejemplo de análisis de la evolución del sistema en estudio (curación paciente) (41)	23
Figura 2.6.2 Ejemplo de análisis de los datos para un ámbito general (curación enfermedad) 23	23
Figura 2.6.3 Ejemplo sistema de control de lazo abierto	23
Figura 2.6.4 Ejemplo sistema de control de lazo cerrado	24
Figura 2.6.5 Ejemplo sistema de monitorización, análisis y control de datos	24
Figura 3.1.1 Apertura y cierre de estoma	25
Figura 4.1.1 Esquema conceptual del primer nodo sensor	33
Figura 4.1.2 Tensión promedio de un filtro de una señal PWM	36
Figura 4.1.3 Circuito electrónico de un amplificador de instrumentación	38
Figura 4.1.4 Circuito electrónico de un amplificador diferencial	40
Figura 4.1.5 Circuito electrónico de un amplificador restador	40
Figura 4.1.6 Circuito de acondicionamiento para el sensor LAT-B3	41
Figura 4.1.7 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor LAT-B3	42
Figura 4.1.8 Diagrama de Bode para comprobar el filtrado del sensor LAT-B3	42
Figura 4.1.9 Circuito de acondicionamiento para el sensor PYTHOS31.....	43
Figura 4.1.10 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor PYTHOS31.....	43

Figura 4.1.11 Diagrama de Bode para comprobar el filtrado del sensor PYTHOS31.....	43
Figura 4.1.12 Circuito de acondicionamiento para el sensor 314990620	44
Figura 4.1.13 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor 314990620	45
Figura 4.1.14 Diagrama de Bode para comprobar el filtrado del sensor 314990620.....	45
Figura 4.2.1 Batería usada para alimentar el primer nodo implementado	47
Figura 4.3.1 Alimentación por pines Arduino Uno	48
Figura 4.3.2 Pines de alimentación Raspberry Pi 4 modelo B.....	48
Figura 4.3.3 Hoja jerárquica principal PCB Alimentación	49
Figura 4.3.4 Funte_de_alimentacion.kicad_sch	49
Figura 4.3.5 Placas.kicad_sch.....	49
Figura 4.3.6 Conversores.kicad_sch.....	50
Figura 4.3.7 Diseño PCB Alimentación.....	51
Figura 4.3.8 Cara frontal diseño 3D PCB Alimentación.....	51
Figura 4.3.9 Cara posterior diseño 3D PCB Alimentación	51
Figura 4.3.10 Cara frontal PCB Alimentación sin y con componentes.....	52
Figura 4.3.11 Cara trasera PCB Alimentación sin y con componentes	52
Figura 4.3.12 Hoja jerárquica principal PCB Suelo.....	53
Figura 4.3.13 alimentacion.kicad_sch.....	54
Figura 4.3.14 pines_alimentacion_sensores.kicad_sch.....	54
Figura 4.3.15 pines_IO_sensores.kicad_sch	54
Figura 4.3.16 conversores.kicad_sch con driver NMOS	56
Figura 4.3.17 conversores.kicad_sch con driver PMOS	56
Figura 4.3.18 acondicionamiento_moisture.kicad_sch	56
Figura 4.3.19 Diseño PCB Suelo	57
Figura 4.3.20 Cara frontal diseño 3D PCB Suelo	57
Figura 4.3.21 Cara posterior diseño 3D PCB Suelo	58
Figura 4.3.22 Cara frontal PCB Suelo sin y con componentes	58
Figura 4.3.23 Cara posterior PCB Suelo sin y con componentes	59
Figura 4.3.24 Comprobación de la alimentación al variar el pin de control.....	59
Figura 4.3.25 Hoja jerárquica principal PCB Hoja	60
Figura 4.3.26 alimentacion.kicad_sch.....	60
Figura 4.3.27 pines_alimentacion_sensores.kicad_sch.....	61
Figura 4.3.28 pines_IO_sensores.kicad_sch	61
Figura 4.3.29 conversores.kicad_sch con driver NMOS	62
Figura 4.3.30 conversores.kicad_sch con driver PMOS	62
Figura 4.3.31 referencia_025V.kicad_sch.....	63

Figura 4.3.32 acondicionamiento_temperature.kicad_sch	64
Figura 4.3.33 acondicionamiento_humidity.kicad_sch	64
Figura 4.3.34 amplificacion.kicad_sch	65
Figura 4.3.35 Diseño PCB Hoja	65
Figura 4.3.36 Cara frontal diseño 3D PCB Hoja	66
Figura 4.3.37 Cara posterior diseño 3D PCB Hoja	66
Figura 4.3.38 Cara frontal PCB Hoja sin y con componentes	67
Figura 4.3.39 Cara posterior PCB Hoja sin y con componentes	67
Figura 4.3.40 Módulo MAX485 RS-485 TTL	68
Figura 4.3.41 <i>Esquema de conexiones del primer nodo implementado</i>	69
Figura 4.4.1 Esquema de conexión para comunicación I2C entre Arduino y Raspberry	71
Figura 4.5.1 Comprobación del funcionamiento del primer nodo implementado	82
Figura 5.1.1 Esquema conceptual del segundo nodo de sensado	83
Figura 5.1.2 Primera parte del esquema electrónico del dispositivo NBSN95 de Dragino	85
Figura 5.1.3 Segunda parte del esquema electrónico del dispositivo NBSN95 de Dragino	85
Figura 5.1.4 Primera parte del esquema electrónico del dispositivo LSN50 de Dragino	86
Figura 5.1.5 Segunda parte del esquema electrónico del dispositivo LSN50 de Dragino	86
Figura 5.1.6 Configuración del Keil µVision 5 para el segundo nodo implementado	87
Figura 5.1.7 Configuración del STM32CubeProgrammer para el segundo nodo implementado	87
Figura 5.1.8 Conexión TTL entre cualquiera de los dos dispositivos de Dragino y el PC	88
Figura 5.1.9 Salida del primer modo implementado (modo 7) en el Serial Port Utility	88
Figura 5.1.10 Diagrama conceptual de la asignación de pines del segundo nodo	94
Figura 5.2.1 Esquema de conexiones del modo 7	95
Figura 5.2.2 Esquema de conexiones del modo 8	95
Figura 5.2.3 Esquema de conexiones del modo 9	96
Figura 5.2.4 Esquema de conexiones del modo 10	96
Figura 5.2.5 Esquema de conexiones del modo 11	97
Figura 5.2.6 Esquema de conexiones del modo 12	97
Figura 5.2.7 Esquema de conexiones del modo 13	98
Figura 5.5.1 Batería utilizada para alimentar al segundo nodo	118
Figura 5.5.2 Panel fotovoltaico utilizado para alimentar al segundo nodo	118
Figura 5.5.3 Integración del panel fotovoltaico a la carcasa del dispositivo de Dragino	119
Figura 6.1.1 Montaje previo a la implantación del sensor SEN0249	121
Figura 6.1.2 Colocación de los sensores en los dos ejemplares de encinas	121
Figura 6.1.3 Montaje de la estación meteorológica	123
Figura 6.2.1 Evolución de la humedad de las hojas	125

Figura 6.2.2 Datos de la humedad de hoja de la encina saludable durante el tiempo de medición	125
Figura 6.2.3 Datos de la humedad de hoja de la encina saludable durante un día.....	126
Figura 6.2.4 Datos de la humedad de hoja de la encina seca durante todo el periodo de medición	126
Figura 6.2.5 Datos de la humedad de hoja de la encina seca durante un día	126
Figura 6.2.6 Datos del <i>O₂</i> de hoja de la encina saludable <i>durante todo el periodo de medición</i>	127
Figura 6.2.7 Datos del <i>O₂</i> de hoja de la encina saludable durante un día	127
Figura 6.2.8 Datos del <i>O₂</i> humedad de hoja de la encina seca durante todo el periodo de medición	128
Figura 6.2.9 Datos del <i>O₂</i> humedad de hoja de la encina seca durante un día	128
Figura 6.2.10 Evolución del <i>CO₂</i> del aire en un día en una ciudad y en una zona rural (79) ...	129
Figura 6.2.11 Evolución del <i>CO₂</i> de la encina saludable durante todo el periodo de medición	129
Figura 6.2.12 Evolución del <i>CO₂</i> de la encina saludable durante un día	130
Figura 6.2.13 Evolución del <i>CO₂</i> de la encina seca durante todo el periodo de medición	130
Figura 6.2.14 Evolución del <i>CO₂</i> de la encina seca durante un día	130
Figura 6.2.15 Evolución del pH del suelo en una zona controlada y en otra quemada (81)	131
Figura 6.2.16 Evolución del pH de la encina saludable durante todo el periodo de medición	131
Figura 6.2.17 Evolución del pH de la encina saludable durante un día	132
Figura 6.2.18 Evolución del pH de la encina seca durante todo el periodo de medición.....	132
Figura 6.2.19 Evolución del pH de la encina seca durante un día	132

Índice de tablas

Tabla 1.4.1 Diagrama de Gantt	7
Tabla 3.2.1 Elección de sensores	31
Tabla 4.2.1 Consumo de potencia de los sensores del primer nodo	46
Tabla 4.2.2 Consumo de potencia del Arduino Uno y la Raspberry Pi 4	46
Tabla 4.3.1 Alimentación utilizada para los distintos sensores de suelo.....	55
Tabla 4.3.2 Alimentación utilizada para los distintos sensores de hoja	61
Tabla 5.1.1 Relación entre los sensores y modos de funcionamiento del fabricante / tipo de comunicación.....	89
Tabla 5.1.2 Sensores utilizados por el fabricante	89
Tabla 5.1.3 Sensores pensados para implementar el segundo nodo	90
Tabla 5.1.4 Datos útiles y de configuración que devuelve la estación meteorológica.....	90
Tabla 5.1.5 Datos útiles y de configuración que devuelve el sensor SHT20.....	91
Tabla 5.1.6 Datos útiles y de configuración que devuelve el sensor Sensecap.....	91
Tabla 5.1.7 Datos útiles y de configuración que devuelve el sensor GPS.....	91
Tabla 5.1.8 Datos útiles y de configuración que devuelve el contador de pulsos.....	91
Tabla 5.1.9 Relación entre los sensores y modos de funcionamiento / tipo de comunicación .	92
Tabla 5.1.10 Asignación de los pines a cada sensor y nodo	92
Tabla 5.4.1 Contenido de la trama que se va a transmitir.....	117
Tabla 5.5.1 Consumo de los elementos del sistema.....	119
Tabla 7.1.1 Gastos asociados a los sensores de sendos nodos implementados	133
Tabla 7.1.2 Gastos generales del primer nodo	134
Tabla 7.1.3 Costes de terrenos y bienes naturales	135
Tabla 7.1.4 Costes de transportes	135
Tabla 7.1.5 Costes en sueldos y salarios	135
Tabla 7.1.6 Coste en Seguridad Social	135
Tabla 7.1.7 Costes en recursos y equipos informáticos.....	136
Tabla 7.1.8 Costes de licencias de herramientas de diseño y desarrollo	136
Tabla 7.1.9 Costes de suministros	136
Tabla 7.1.10 Costes de equipo.....	137
Tabla 7.1.11 Costes fijos y variables	137

1. Introducción

La presente introducción se estructura en diferentes apartados, se recogerá la motivación que ha guiado en todo momento la realización de este trabajo. Se fijarán los distintos objetivos que se pretenden alcanzar en el desarrollo del estudio, sin olvidar mencionar las posibles utilidades del presente proyecto.

También se realizará un diagrama de Gantt que mostrará el tiempo necesario para la organización del trabajo. Para finalizar la introducción, se expondrá brevemente la organización de la memoria de este Trabajo Fin de Máster (TFM).

1.1. Motivación

En el ámbito científico y académico se da por cierta las siguientes premisas:

- El calentamiento global ha quedado atrás, ahora estamos en la fase de la ebullición global, esto agrava aún más el cambio climático que tanto daño hace a la naturaleza y al ser humano.
- Hasta ahora, la naturaleza siempre ha sido capaz de recuperarse de todos los daños producidos por cualquier agente, el ser humano ha sido capaz de adaptarse a los diferentes entornos propuestos por la naturaleza y sobrevivir. Sin embargo, es casi imposible adecuarse tan rápido y sobrevivir a un entorno natural degradado en exceso por el continuado cambio climático. (1)

El presente TFM del Máster Habilitante de Ingeniería de Telecomunicación y el posterior trabajo sobre Internet de las Cosas (IoT), fue una propuesta recibida a finales de junio de 2022 por parte de mi tutor y que acepté como un objetivo muy bonito a conseguir, ya que es un fiel reflejo de la educación en valores recibida. Estos trabajos están inspirados en la propuesta MANAGE4FUTURE dentro de la convocatoria de Proyectos de Transición Ecológica y de Transición Digital correspondiente al Plan de Recuperación, Transformación y Resiliencia. Los estudios e implementaciones realizadas son complementarios a las investigaciones correspondientes a dicha propuesta.

Durante todo este tiempo he estado investigando, creando las bases, planificando y estructurando los dos TFM, pero los componentes físicos necesarios para llevar a cabo la implementación completa han ido llegando escalonadamente entre los meses de mayo y julio de 2023. Por lo tanto, esto ha supuesto un reto adicional en el planteamiento del diseño y el desarrollo de los sistemas, así como una limitación para realizar una fase de campaña de medidas.

Las generaciones anteriores de mi familia y hasta mis abuelos, siempre han estado muy ligadas al ritmo que marca la naturaleza, eran recolectores, cultivadores de tierras ajena, criadores de pequeñas unidades de ganado, cazadores, se hacían sus propios útiles, su propia ropa, su propia comida... oficios y habilidades vinculadas con el medio rural. Fueron personas que supieron vivir en el entorno natural, de los medios que proporciona

INTRODUCCIÓN

y estar vinculadas al mismo, tenían muy claro que respetar a la naturaleza era respetarse a sí mismos y al resto de la sociedad, que eran seres naturales.

Esta cultura del respeto y, a la vez, de amor por la naturaleza me la han sabido transmitir, la tengo muy presente y ahora tengo la responsabilidad de que esos valores y cultura no se pierdan, aunque estemos en la era más tecnológica que se ha conocido, aunque no sea mi modo de llevar el pan a la mesa.

Me educaron para tener la casa limpia y ordenada, tanto la propia como la común. Esto implica tanto la posesión física de un techo bajo el que dormir, como el de un techo desde donde ver las estrellas. Una enseñanza que intento aplicar es que nuestro paso por un lugar lo mejore para el siguiente que pase por él.

También, me he acostumbrado a participar tanto de forma familiar como colectiva, en la mejora de la biodiversidad con la plantación de árboles, incluso hemos participado en convertir una escombrera en el parque Manolito Gafotas, en el barrio de Carabanchel Alto en Madrid. Esto sucedió acudiendo a las anuales convocatorias de la asociación de vecinos para plantar árboles, que luego seguimos cuidando, y que, pasados unos quince años, no hay rastro de dicha escombrera, convirtiéndola en un pedacito de naturaleza llena de árboles. Provocando una pequeña mejora del barrio y del medio ambiente, lo que se puede denominar un parque que ahora está en manos del Ayuntamiento.

El instinto de supervivencia del ser humano se puede valorar como el más alto de todas las especies, cosa que nos ha hecho estar en la cúspide de la pirámide y de ponernos al mismo tiempo al borde del abismo. La especie homo sapiens sapiens puede llegar a morir de éxito, dejando paso a la recuperación de la naturaleza por sus propios medios, como ha sucedido a lo largo del tiempo en varias ocasiones.

Este instinto nos tiene que servir de revulsivo, para ayudar de forma activa y fructífera a la rápida recuperación de todo lo que se ha roto en el sistema natural. Todo ello, para que el agua vuelva a su cauce y se recupere el orden natural, pues todo y todos somos Naturaleza, ella puede vivir sin nuestra presencia, cosa que al contrario no es posible.

Me motiva la multitud de conocimientos, métodos y ayudas existentes. Éstas se deben poner en marcha a la vez y a un mismo ritmo, para solucionar los graves problemas creados a la naturaleza y, por ende, al ser humano. Intentando revertirlos y previniéndolos para que no vuelvan a suceder y, de esta forma, convertir la naturaleza en una fuente inagotable de generación de riqueza. Es crucial no matar a la gallina de los huevos de oro, cosa que nos llevaría a nuestro propio suicidio.

Este trabajo me motiva porque puedo hacer otra pequeña aportación a la solución, pues si consigo que alguien aprenda que ayudar a la naturaleza no es un gasto o un coste imposible de asumir por la economía, sino la mejor inversión que la sociedad puede realizar y, además, la más rentable. En conclusión, volveremos a ser seres naturales con las comodidades que nuestro ingenio nos proporciona.

Por último, pero nunca menos importante, me motiva mucho poder ser una diminuta parte activa, a través de este TFM y de otras acciones, en la superación del reto que nos hemos creado. Al poderse calificar éste como el mayor al que se ha enfrentado nuestra especie desde que apareció en este planeta, el cual debemos superar por responsabilidad para con las generaciones futuras, al disponer de los medios y conocimientos necesarios. Siempre hemos sabido superarnos y levantarnos, desde la naturaleza y con la naturaleza.

1.2. Objetivos

El objetivo principal del proyecto es realizar una correcta monitorización de un bosque de quercíneas o quercus (robles, encinas, alcornoque). Para ello es necesario comenzar por aspectos relativos a estos bosques y los ejemplares que los integran, para posteriormente proponer qué variables y sensores serían los más interesantes para este estudio.

Este trabajo se puede abordar desde diferentes puntos de vista, los cuales pueden ser:

- El estudio del bosque como un todo, utilizando teledetección con drones, aviones y satélites que están dotados de cámaras hiperespectrales. Estas sirven para detectar gran parte del espectro electromagnético, tanto el visible como el invisible para el ojo humano, se utilizan junto a procesos de visión artificial, con el objetivo de realizar un análisis en tiempo real sobre los datos físicos y químicos del bosque, pudiendo generar un mapa tridimensional.
- El estudio del bosque a través de cada unidad arbórea (quercus), para ello sería necesario ver qué magnitudes son las más adecuadas y lograr un sistema de monitorización de dichas magnitudes.
- El estudio del bosque a través del clima, de las características meteorológicas y del terreno, estudiando tanto la atmósfera en sus capas más bajas como la morfología y la composición del terreno, todo esto para observar cómo afectan dichos parámetros al desarrollo de estos bosques.



Figura 1.2.1 *Encina*

INTRODUCCIÓN

En un principio, se ha decidido utilizar los dos últimos ejemplos para la realización de este proyecto, debido a que son medios más asequibles y adecuados para el presente estudio, a pesar de que cada uno de los ejemplos descritos podrían dar lugar a un TFM.

Al ser este estudio tan ambicioso y extenso, se ha decidido separarlo en dos proyectos:

- El primer y presente trabajo consiste en la implementación del sistema de monitorización de dos ejemplares de quercíneas para estudiarlos a lo largo del tiempo, ver su estado, su evolución y medir los parámetros medioambientales a través de una estación meteorológica.
- El segundo estudio consistirá en la implementación de un sistema de Internet de las Cosas (IoT) con varios nodos a fin de recoger la información necesaria para el estudio del desarrollo y estado del bosque, a través de una serie de árboles seleccionados al azar y de las condiciones ambientales que les rodea. Este sistema IoT podrá ser escalable, desde solamente dos nodos, hasta los que se deseen.

Al ser estos objetivos muy amplios, se hará un desglose de las tareas necesarias para cumplir satisfactoriamente con el presente proyecto:

- Realizar una búsqueda y estudio previo de la información que se precisará para abordar de forma eficaz el inicio del proyecto.
- Estudiar las magnitudes que indican el estado de salud del bosque de quercíneas y seleccionar aquellas que se desean estudiar.
- Buscar los sensores que miden las magnitudes elegidas y decidir que sensores se adaptan mejor al estudio.
- Buscar distintas estaciones meteorológicas y seleccionar la que más se ajuste al proyecto.
- Realizar el acondicionamiento de las señales provenientes de los sensores que lo necesiten desarrollando placas de circuito impreso (PCB).
- Ejecutar correctamente la integración de los sensores en los sistemas embebidos (placas empotradas) reduciendo riesgos de fallos.
- Comunicar correctamente el nodo, mediante un protocolo de comunicación.
- Recopilar y procesar los datos recogidos de los sensores.
- Graficar los datos procesados y realizar un breve estudio de estos.
- Proporcionar las conclusiones generadas por el presente estudio.

1.3. Impacto

El presente trabajo puede ser útil para comprobar cómo el cambio climático afecta a los bosques de quercíneas y altera el medioambiente. Por otra parte, sus resultados pueden integrarse en varios ámbitos y proyectarse hacia otras ramas del conocimiento:

Las telecomunicaciones: Estudiando las magnitudes físicas a través del procesamiento de las señales recibidas de los distintos sensores que se utilizan para la monitorización del bosque de quercíneas, consiguiendo depurar los datos y extraer conclusiones, logrando que sirvan de base para diferentes disciplinas científicas y medioambientales, que pudieran utilizarlas para mejorar este entorno.

La electrónica: Se hace uso de sensores para medir las distintas magnitudes en estudio, y también, se hacen diferentes circuitos de acondicionamiento de estos sensores. Todo este proceso puede servir de base a futuros trabajos, explicando por qué y cómo se han

INTRODUCCIÓN

utilizado los sensores, que se indicarán en la presente memoria, y exponer su acondicionamiento.

Los sistemas embebidos: Se hará uso de distintas placas empotradas para la captación de la información proporcionada por los sensores y circuitos de acondicionamiento en tiempo real, realizando el sistema de comunicación a través de distintos protocolos.

La agricultura: El proyecto realizado puede ser útil, no solo para un ámbito forestal y medioambiental, sino también para la agricultura en general, comprobando el estado de plantaciones, suelo, ambiente, etc. De esta forma, se pueden prevenir enfermedades y plagas, optimizando los recursos necesarios para una buena cosecha, mejorando su calidad y cantidad.

La biología y la botánica: El trabajo realizado puede ayudar a estas dos disciplinas en su investigación sobre las quercíneas, comprobando los efectos que tiene el cambio climático sobre estos bosques, seleccionando qué especies son las más adecuadas, resistentes y que aporten mayor valor biológico y económico en cada tipo de clima y terreno. La mejora de los bosques conlleva directamente la mejora de la fauna y de la flora que lo rodea.

Las ciencias medioambientales y la ecología: Puede ser de utilidad, puesto que los datos obtenidos pueden servir de base para encontrar soluciones que arreglen lo destruido y, con la información obtenida en tiempo real, prevenir posibles desastres medioambientales. Estos mismos datos sirven para realizar campañas de concienciación para la lucha contra el cambio climático, conservación de los bosques y campañas contra incendios, entre otras.

El IoT: El trabajo indica cómo realizar las estaciones de medición de los quercus, por tanto, estas bases pueden ser los nodos necesarios para la correcta monitorización de este tipo de bosques, pudiendo crear un sistema de IoT a gran escala. Además, también puede servir de base para otros proyectos de IoT.

A parte de los anteriormente nombrados, también existen otros campos en los que la monitorización de ecosistemas es una parte necesaria y fundamental. También se comprueba que en un futuro cercano habrá un aumento de sistemas de monitorización y actuación en los diferentes entornos naturales, que intentarán solucionar las posibles alteraciones que pudieran perjudicar dichos entornos.

1.4. Fases de desarrollo y organización del trabajo

La realización del presente proyecto consta de múltiples tareas o procesos, porque se ha considerado que optimizaba su buen desarrollo:

- **Estudio previo:** Una vez fijada la idea principal, en este primer paso se realiza un análisis previo donde se recopilan y estudian cuales son los datos y medios necesarios para poder abordar correctamente el presente TFM.
- **Búsqueda y selección de las magnitudes:** Una vez establecido claramente el objeto del estudio, se indaga en las posibles magnitudes de

INTRODUCCIÓN

medida que pueden influir en el presente trabajo, posteriormente se seleccionarán las que sean más relevantes.

- **Búsqueda y selección de los sensores, la estación meteorológica y microcontroladores (μC):** A continuación, se buscan los sensores más adecuados capaces de medir las magnitudes seleccionadas en el apartado anterior. También será necesario buscar los μC óptimos para realizar el sistema. Posteriormente, en función de sus características y precios, se procede a seleccionar los que se utilizarán para implementar el sistema.
- **Lanzamiento de compras y espera de suministros:** Una vez se saben todos los elementos necesarios para llevar a cabo todo el sistema, es necesario adquirirlos y esperar a que estos lleguen.
- **Diseño Software:** Se hace necesario, mientras llega el material, ir realizando los códigos que se encargarán de adquirir, procesar y almacenar los datos para los distintos μC que se van a utilizar.
- **Diseño Hardware:** Para poder operar correctamente con los sensores, es necesario comprobar si se pueden conectar directamente al sistema embebido o, en su defecto, es necesario realizar un circuito de acondicionamiento para que se puedan recibir correctamente los datos obtenidos por el sensor. En caso de ser necesario realizar circuitos de acondicionamiento, también se requerirá realizar un estudio de estos, para la correcta elección de los componentes necesarios y su implementación.
- **Recopilar y procesar los datos:** Aquí se analiza la forma de almacenar los datos obtenidos, junto con el tratamiento de estos para poder obtener unos resultados lógicos, fiables e interpretables por especialistas.
- **Redacción de la memoria:** En este apartado final, se han plasmado las ideas, actuaciones, implementaciones y resultados generados a lo largo de todo este trabajo, el conjunto del mismo ha sido reflejado de forma paralela y simultánea a su realización.

Este trabajo contempla una estimación de forma gráfica, mediante el diagrama de Gantt, de los tiempos y duraciones que se prevén para las distintas actividades que lo conforman. El diagrama de Gantt se plasma en la siguiente tabla:

Diagrama de Gantt



Universidad
Carlos III de Madrid

Red IoT para la monitorización de la salud de los bosques de coníferas

Autor: Manuel Álvarez Herrera Tutor: Jose Antonio García Souto

Meses			JUL	AGO	SEP	OCT	NOV	DIC	ENE	FEB	MAR	ABR	MAY	JUN	JUL	AGO
Descripción del hito	Inicio	Días														
Estudio previo, búsqueda y selección																
Estudio previo	04/07/2022	119														
Búsqueda y selección de las magnitudes	04/07/2022	35														
Búsqueda y selección de los sensores y estación meteorológica	08/08/2022	70														
Búsqueda y selección de los microprocesadores	29/08/2022	42														
Compras																
Lanzar compras	17/10/2022	28														
Espesa de suministro	17/10/2022	259														
Desarrollo SW																
Código Arduino	01/09/2022	56														
Código Raspberry Pi	24/10/2022	49														
Código dispositivos Dragino	03/04/2023	77														
Desarrollo HW																
Diseño circuitos de acondicionamiento	06/02/2023	49														
Implementar primer nodo	10/04/2023	84														
Implementar segundo nodo	03/04/2023	52														
Obtención datos																
Recopilar y procesar datos	03/08/2022	56														
Memoria																
Redacción de la memoria	11/08/2022	420														

Tabla 1.4.1 Diagrama de Gantt

1.5. Organización de la memoria

La organización de esta memoria se encuentra dividida en los siguientes apartados:

Apartado 2

Estado del arte: En este apartado se presentará una breve historia sobre los quercus, los bosques, el cambio climático y como este último afecta a los bosques de quercus.

Apartado 3

Estudio y selección de las magnitudes, de los sensores y de las estaciones meteorológicas: En este apartado se recogerán las diferentes magnitudes que afectan a los bosques de quercus, se indicará cuáles de estas se van a utilizar en el presente trabajo. También se clasificarán los diferentes tipos de sensores y estaciones meteorológicas, exponiendo algunos ejemplos de los modelos existentes y especificando cuales de ellos son los que se utilizarán en este proyecto.

Apartado 4

Implementación: En este apartado se explicará todo el proceso para realizar el sistema de monitorización de dos ejemplares de quercus. Primero, se tratará el acondicionamiento de los sensores para la correcta obtención de los datos. Posteriormente, se indicará cuáles son las placas embebidas que se emplearán, tanto para la conexión con los sensores y la estación meteorológica, como para la conexión entre las diferentes placas, indicando como se centralizará la recogida de datos. Finalmente, se expondrá cómo se va a integrar el sistema de monitorización sobre el árbol.

Apartado 5

Datos obtenidos: En este apartado se expondrá un extracto de los datos recogidos a lo largo de un periodo de tiempo, se tratarán estos datos para generar las gráficas correspondientes a la evolución de los quercus y del ambiente en el que viven.

Apartado 6

Conclusiones y trabajo futuro: En este apartado se efectúa la recapitulación y se deducen las conclusiones que emanen de los resultados obtenidos. Finalmente, se enumeran las partes mejorables del trabajo, indicando posibles ejemplos para la utilización, mejora y posible realización de trabajos futuros.

Por último, se indicará la bibliografía, donde se muestra la documentación manejada y consultada para el desarrollo del presente trabajo, junto con los distintos anexos.

2. Estado del arte

La naturaleza, en general, es la esencia de la vida, recoge los hábitats, ambientes y a todos los seres vivos existentes en el planeta Tierra. La deforestación de los bosques cada vez es mayor poniendo en peligro toda la riqueza natural del planeta y nuestra capacidad para afrontar el cambio climático, que hace ya muchos años los científicos vienen avisando. Los bosques, junto con las algas marinas, son los auténticos pulmones de nuestro planeta.

Este trabajo se centra en la monitorización de dos ejemplares de encinas. La elección de este tipo de árboles no es casual, debido a que existen más de quinientas especies (2) de *quercus* en todo el mundo, dado que son árboles de gran importancia ecológica, económica y de gran longevidad.

2.1. Quercus: definición, contribución a la economía y localización

2.1.1. Definición

Los *quercus* son miembros del Reino Plantae, específicamente de la División Magnoliophyta, son árboles y arbustos que se ubican en la clase Magnoliopsida, pertenecen al orden fagales y la familia Fagaceae. Su clasificación llega hasta el género *Quercus*. El origen de esta especie se remonta al final del Cretácico inferior, hace aproximadamente 145 millones de años. (3)



Fossil of *Quercus x hispanica*.
(South of France. Antoine pictures)



Fossil of *Quercus mediterranea*
(Image by Dimitrios Velitzelos et al. 2014)

Figura 2.1.1 Fósiles de hojas de quercíneas (4) (5)

ESTADO DEL ARTE

Los quercus son árboles o arbustos, que manifiestan un crecimiento lento. Sus hojas pueden ser perennes o caducas, poseyendo una disposición alterna a ambos lados del tallo. La variedad de la forma de sus hojas entre distintas especies es considerable, incluso para un mismo ejemplar a lo largo de su vida o según su ubicación puede variar la forma, el color o la textura de la hoja.

Son plantas fanerógamas, es decir, tienen flores unisexuales, pueden coexistir, dentro de un mismo quercus, flores en forma de amento, tanto masculinas como femeninas, con tonalidades que varían entre el verde y el amarillo.

Las flores masculinas siguen un patrón de racimos largos y colgantes, con una estructura de espiga articulada colgante, exhibiendo entre cuatro y diez estambres. A diferencia de las femeninas, que son pequeñas y, en algunos casos, solitarias. Estas muestran tres estigmas y óvulos anátropes que están girados 180° con relación a su base. El fruto es solitario y de origen axil (de brote), con cotiledones planos llamado bellota.



Figura 2.1.2 Flores masculinas (izquierda) y femeninas (derecha) de la encina (6) (7)

La gran mayoría, de los quercus son árboles de hojas perenne, ya que estas hojas tienen una duración de dos a cuatro años, lo que favorece la adaptación de los sensores. Estas hojas son coriáceas, similares al cuero, con una tonalidad verde oscuro en el haz y un tono más claro en el envés. Cuando la planta es joven, sus hojas suelen estar provistas de espinas en el contorno, mientras que, en la edad adulta, las espinas se localizan en las ramas inferiores.

Inicialmente, el tronco de la encina se presenta con una superficie lisa de color verde grisáceo. Conforme el árbol crece, esta tonalidad se oscurece progresivamente. Aproximadamente a los 15 o 20 años de vida, la corteza experimenta grietas y un oscurecimiento, resultando en un tronco prácticamente negro que podría llegar a alcanzar hasta 1 metro de diámetro.



Figura 2.1.3 Tronco agrietado y oscurecido de encina adulta

Los quercus son altamente valorados por la multitud de beneficios que aportan, incluyendo su madera, corteza, frutos, contenido en taninos, capacidad de captura de carbono, contribución a la mitigación del cambio climático, papel en la formación y preservación del suelo (evitando la desertificación), mejora de la calidad del aire y agua.

2.1.2. Contribución a la economía

Las quercíneas son especies vegetales que aportan gran valor tanto al medio ambiente como a la economía. Algunos ejemplos que denotan la importancia de estos árboles son los siguientes:

- **Quercus Ilex:** Conocido por su fruto, la bellota, que es un importante fruto seco, comestible para el ser humano y para el ganado. Este fruto permite la elaboración de harinas, al ser un producto oleaginoso se puede obtener aceite. También se elaboran tanto bebidas alcohólicas como no alcohólicas. Tienen efectos curativos como cicatrización de heridas, regula el azúcar en sangre, alivia el reuma... (8)
- **Quercus Robur:** También conocido como roble blanco o carbollo, su madera es muy valorada para la construcción de muebles, ebanistería y para la construcción de barcos, al poseer gran resistencia a la humedad. También es apreciada para la fabricación de toneles y barricas al proporcionar a distintos licores su color característico como el Whisky o el Brandy. Otras utilidades son que se utiliza para curtir pieles y es un buen combustible. (9)
- **Quercus Suber:** Conocido también como alcornoque o sobreira. Destaca por su producción de corcho, un material ampliamente utilizado en diversos ámbitos como en la apicultura. Tiene un gran valor medioambiental al aprovechar al máximo el agua, ayudan a combatir la erosión de los suelos y permiten que estos almacenen agua. Además, permiten que el desarrollo de otra vegetación bajo ellos junto con hongos como boletus, trufas y rebozuelos, entre muchos otros. (10)

2.1.3. Localización

El ámbito geográfico donde se encuentran distribuidas es, fundamentalmente, el hemisferio norte. Se localizan en una gran diversidad de hábitats que abarcan desde los bosques, las marismas y las sabanas, a las zonas desérticas. Se sitúan en diferentes altitudes que van desde nivel del mar hasta los 4.000 metros de altitud. (11) (12)



Figura 2.1.4 Encina Terrona, la más antigua de la península Ibérica y probablemente de Europa, se localiza en Zarza de Montánchez (Cáceres) (13)

ESTADO DEL ARTE

Las siguientes imágenes exponen la distribución en España de diferentes quercus, resaltadas en verde para indicar su localización. También, se ilustra la distribución de las quercíneas por Europa:



Figura 2.1.5 Distribución del quercus robur (14)



Figura 2.1.6 Distribución del quercus pirenaica o roble negro (15)

ESTADO DEL ARTE



Figura 2.1.7 Distribución *quercus faginea*, conocido como, quejigo, roble carrasqueño (16)



Figura 2.1.8 Distribución *quercus canariensis* conocido como, roble andaluz o quejigo andaluz (17)

ESTADO DEL ARTE

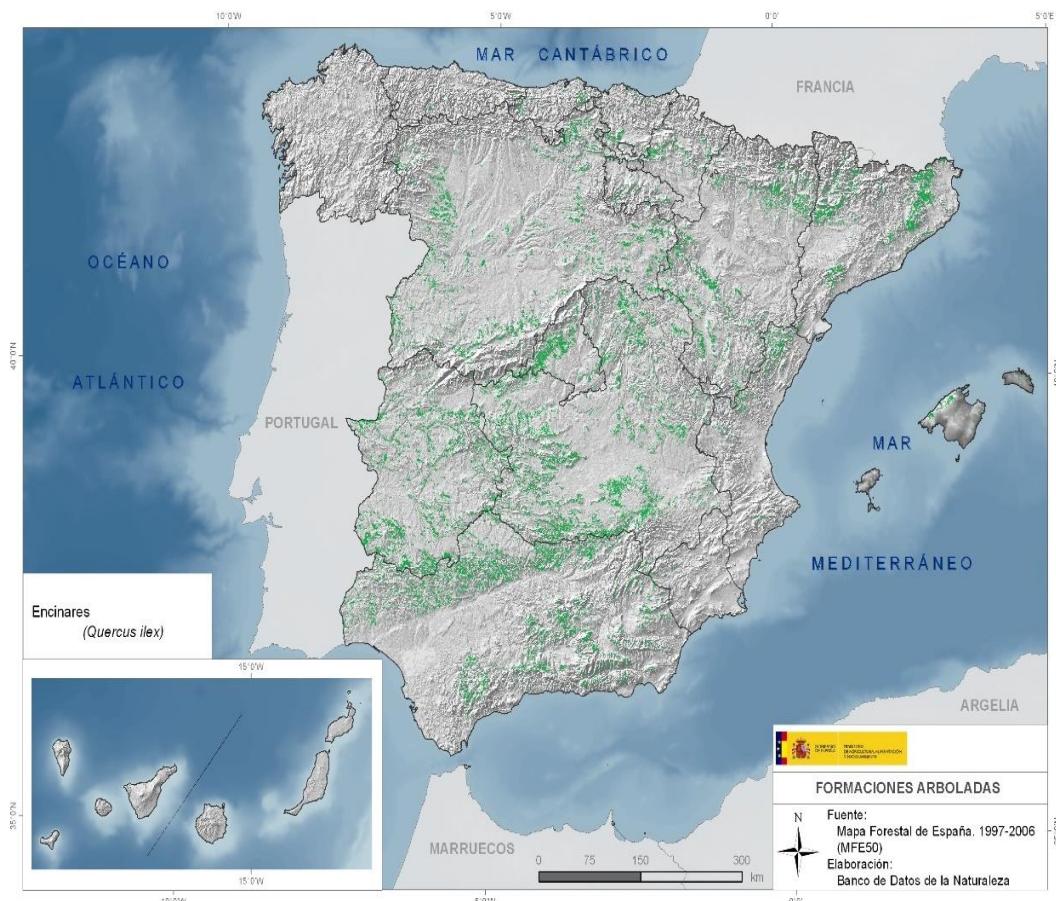
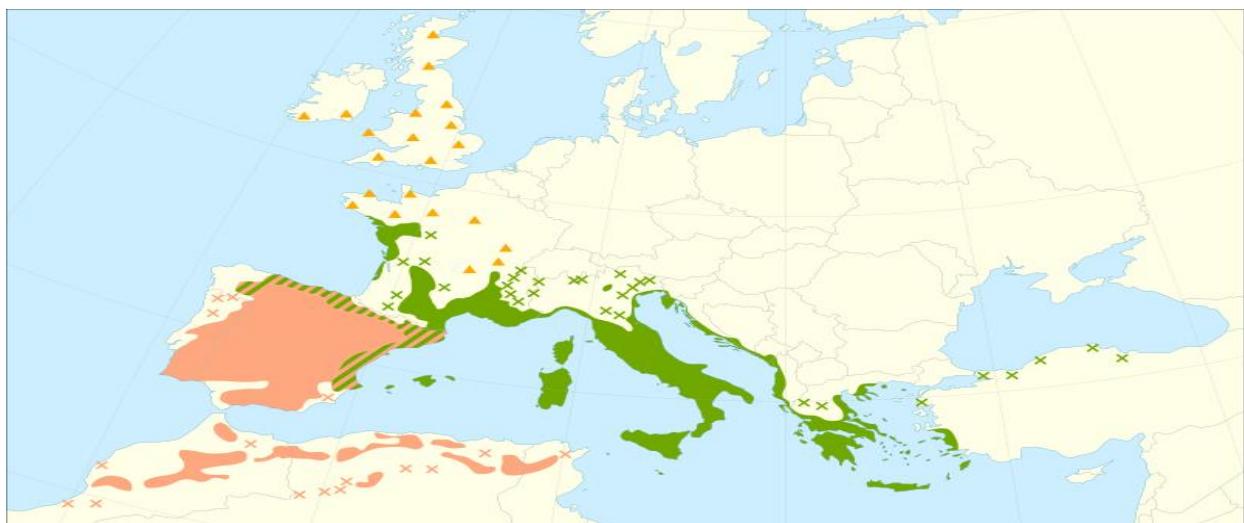


Figura 2.1.9 Distribución quercus ilex o encina (18)



Figura 2.1.10 Distribución quercus suber o alcornoque (19)



Legend of subspecies^[1]:

- *Quercus ilex* subsp. *ilex*.
- *Quercus ilex* subsp. *rotundifolia* (syn. *Q. rotundifolia*, *Q. ballota*).
- ▲ Introduced and naturalised (synanthropic).

Figura 2.1.11 Distribución de subespecies quercus Ilex por Europa (20)

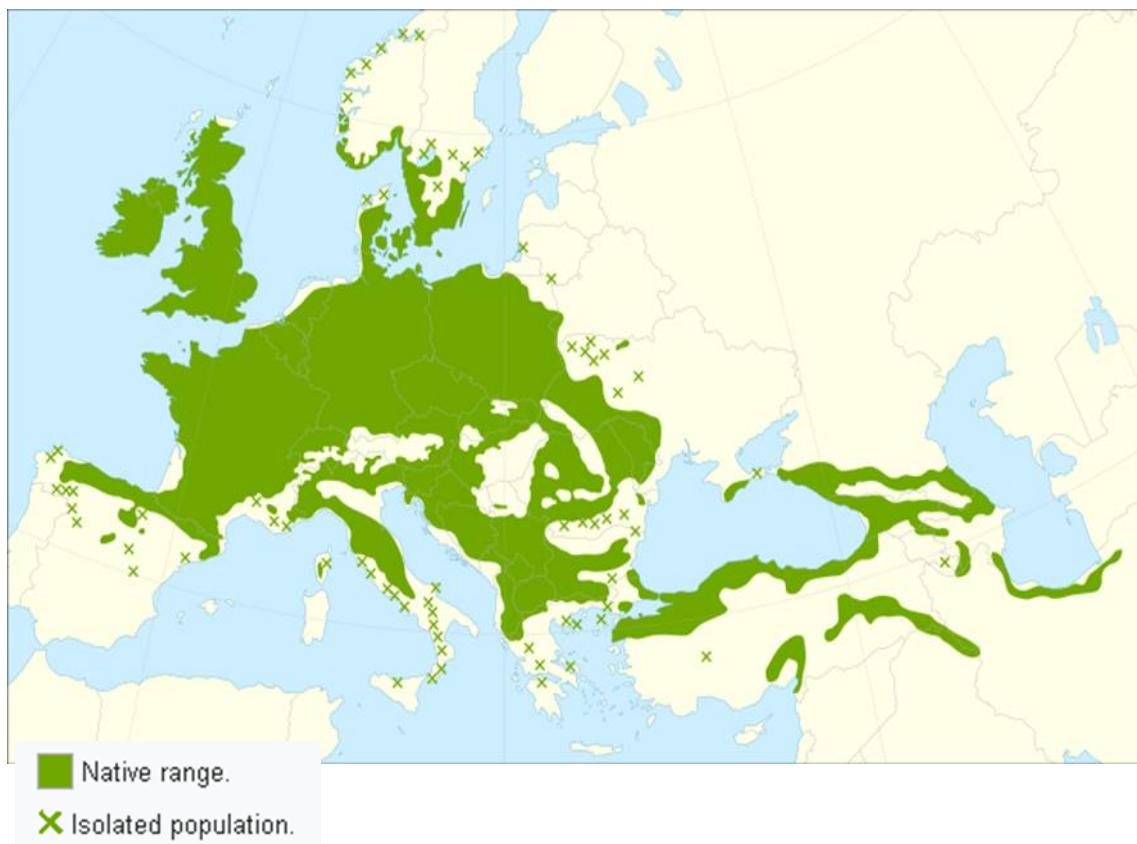


Figura 2.1.12 Distribución quercus petraea por Europa

2.2. Cambio climático

A principios del siglo XIX, por primera vez, se descubrió un efecto invernadero natural, debido a que el clima de la Tierra ha cambiado en múltiples ocasiones y de forma cíclica en los últimos ochocientos mil años. Este efecto es debido a que se producen pequeñas variaciones en la órbita terrestre, recibiendo una mayor o menor energía solar, esto provoca avances o retrocesos de los glaciares (21).

En 1896, el científico sueco Svante Arrhenius, en un artículo que publicó (22), prevé que los cambios en los niveles de dióxido de carbono (CO_2) atmosférico podrían alterar sustancialmente la temperatura de la superficie terrestre, a través del efecto invernadero. Gracias al uso de satélites y otras tecnologías, se ha obtenido mucha información sobre el clima de nuestro planeta y, hoy en día, se sabe que la mayor parte del calentamiento global se debe a la actividad humana, como dijo el físico y climatólogo James Hansen: “El calentamiento global ha alcanzado un nivel tal que podemos atribuir con un alto grado de certeza una relación de causa y efecto entre el efecto invernadero y el calentamiento observado”.

Desde la Edad de Hielo, debido al desarrollo de la sociedad humana han aumentado las emisiones de CO_2 a la atmósfera, esto se produce a una velocidad cada vez mayor que si su origen fuese natural, más concretamente veinticinco veces mayor.

El estudio del clima antiguo o paleoclima indica que el calentamiento en la actualidad está sucediendo de forma más vertiginosa, se estima que unas diez veces más rápido. La temperatura del planeta ha incrementado un grado Celsius desde finales del siglo XIX. El mayor incremento de las temperaturas ha sucedido en los últimos cuarenta años, siendo los últimos los siete años más cálidos que se han documentado.

Los efectos del cambio climático han resultado preocupantes, por eso se realizaron diversos modelos matemáticos para intentar predecir como evolucionaría el planeta ante el aumento del CO_2 y como serían sus consecuencias. Algunos estudios, como el del premio nobel Syukuro Manabe (23), han mostrado que el cambio climático dista de ser uniforme, siendo muy diferente sus consecuencias dependiendo de la parte del planeta en la que se sitúe.

El calentamiento general y el aumento del contenido de humedad del aire, resultantes de un aumento del CO_2 , contribuyen a la gran reducción del gradiente de temperatura meridional en la troposfera inferior del modelo debido al retroceso hacia el polo de la capa de nieve altamente reflectante y al gran aumento del transporte de calor latente hacia el polo.

Otro efecto del cambio climático es visible en los océanos, estos han absorbido el aumento de las temperaturas en 0,33 grados Celsius en su superficie desde finales de la década de los sesenta, los mares han aumentado unos veinte centímetros en los últimos cien años siendo este proceso cada vez más rápido. También se puede ver el efecto que este calentamiento provoca en grandes masas de hielo como las que hay en la Antártida y Groenlandia, comprobando que han disminuido de forma significativa, perdiendo unos 150 mil millones de toneladas y 280 mil millones de toneladas respectivamente en las últimas dos décadas.

ESTADO DEL ARTE

Se tienen bosques cada vez más enfermos que pueden desequilibrar decisivamente la vida en el planeta. En un futuro, pueden pasar de absorber el CO_2 a emitirlo, causando una aceleración sin precedentes del cambio climático.

Se han realizado diversas investigaciones sobre este efecto, una de ellas (24), la cual durante treinta años ha estudiado trescientos mil árboles de más de quinientas setenta selvas y se ha obtenido conclusiones muy inquietantes, los científicos calculan que, sobre mediados del 2030, los bosques pasarán de absorber a emitir CO_2 . Todo esto es debido a los incendios, sequías, acidificación del suelo, aumento de temperatura, plagas, deforestaciones, dinámica forestal y actividades humanas. Según Wannes Hubau: “Los mayores valores de CO_2 aceleran el crecimiento de los árboles; sin embargo, año tras año, este efecto se va contrarrestando por los impactos negativos que ocasiona las temperaturas más altas y las sequías, que reducen su tasa de crecimiento y aumentan su mortalidad”.

La solución ineludible es disminuir la actividad humana, realizar más plantaciones de bosques y un mayor cuidado de ellos, para que así los ecosistemas se regeneren, pero se sabe que cada año desaparece casi cinco millones de hectáreas forestales y la salud de las que quedan está mucho más dañada.

En los últimos tiempos, el ser humano ha contribuido en el deterioro y la destrucción de la naturaleza y, específicamente, en todo tipo de bosques. Por esto, es fundamental que todas las acciones que influyen en los bosques y en la naturaleza valoren siempre los posibles riesgos, ya que es necesario la protección de todos estos entornos, debido a que nuestra vida depende de esto.

2.3. Acciones para la sostenibilidad de los bosques

Con todos los problemas vistos anteriormente, los gobiernos y organismos internacionales han decidido hacer frente al problema y tomar diversas medidas para proteger y mejorar el estado de los bosques. Algunos ejemplos de las acciones tomadas por diferentes instituciones son:

- En Europa Central, algunos países, en especial Alemania, han implantado el “Dauerwald”, que es una forma de gestionar la masa forestal y natural de estos estados. El objetivo de este proyecto es conseguir unos bosques sanos a largo plazo, olvidando la rentabilidad rápida. Esto se logra a través de vigilar el suelo, el clima y el estado del bosque. Estos países decidieron no hacer desmontes para tener una regeneración natural, evitar y reducir los productos químicos, usar tecnologías que dejen la menor huella posible en estos ecosistemas, entre otros.
- La Comunidad Europea crea la Red Europea de Seguimiento de Daños en los Bosques en 1986. Consiste en una red para el estudio del estado de salud de los bosques europeos, lo hace a través de un seguimiento anual con siete mil quinientos puntos de control situados por toda Europa en una cuadrícula de doscientos cincuenta y seis kilómetros cuadrados. Estudia la evolución de la salud de los bosques a partir de factores bióticos y abióticos, los distintos valores obtenidos de estos parámetros sirven para elaborar el informe General de situación de los bosques en Europa. Su objetivo principal es poder aunar y armonizar las

ESTADO DEL ARTE

diferentes Redes de seguimiento de Bosques de los distintos países europeos, integrándolas en una única red.

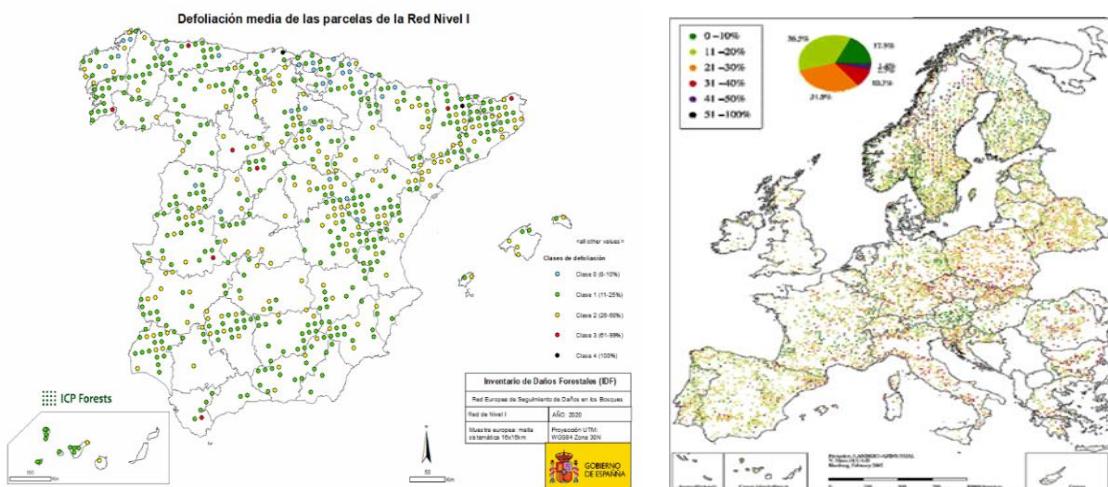


Figura 2.3.1 Defoliación media en las parcelas de España (España 2016) y Europa (España 2001)
(25) (26)

- También la Organización de las Naciones Unidas (ONU), a través del (PNUMA), ha realizado acciones para la protección, educación y divulgación para la sostenibilidad del medioambiente, incluyendo los bosques. Además, la ONU, en la Cumbre sobre el Clima, que se realizó en París, llegó al acuerdo de que, en 2030, se repoblarían trescientas cincuenta millones de hectáreas de terreno degradado.

Para monitorizar el estado de los bosques, y así lograr su conservación, un medio de acercamiento, seguimiento y obtención de información sería a través del uso de los diversos avances tecnológicos. Esta tecnología proporciona un gran volumen de datos e información valiosísima a la hora de la toma de decisiones para la conservación y gestión de recursos naturales. En este trabajo se priorizará el estudio de *quercus*, en concreto, de encinas.

2.4. Sensores

Un sensor es un dispositivo que detecta cambios en el entorno, captando magnitudes químicas o físicas, llamadas variables de instrumentación, se encarga de filtrar, planificar o modificar los estímulos recibidos, transformándolos en magnitudes interpretables por un ordenador, por lo general, en magnitudes eléctricas. A menudo, un sensor no es suficiente para examinar los estímulos exteriores, por ello un mismo equipo suele contener diferentes tipos de sensores.

Como breve historia de los sensores, el ser humano ha tenido la necesidad de controlar el medio que le rodea, comprobando que, para ello, era imprescindible tener instrumentos que midiesen las diferentes magnitudes, temperatura, presión, etc. El primer instrumento reconocido como sensor, fue el ideado para el control de la temperatura (termómetro), este se atribuye al físico e ingeniero polaco Daniel Gabriel Fahrenheit en 1724. La precisión de este dispositivo dependía de conseguir un elemento de amplia expansión térmica, que no se adhiriera al vidrio, que su estado fuera líquido en un amplio rango de temperaturas y fuese fácil de ver, además de conseguir una perforación uniforme del tubo

ESTADO DEL ARTE

de cristal en el que se alojaría el elemento, el metal que cumplía estas características era el mercurio. De esta manera se pudo precisar la cantidad de calor existente, algo que nuestros sentidos no pueden medir, suponiendo un gran avance. (27)

Luego en 1874, un equipo científico francés decidió que sería útil recopilar y dar a conocer a la capital francesa, la temperatura en la cima del Mont Blanc de forma remota, así como la velocidad del viento, su dirección y la altura de la superficie de la nieve, con el tiempo esta estación meteorológica se convirtió en el primer sensor conectado.

Más ejemplos son: gracias a Frederick William Herschel que descubrió los rayos infrarrojos alrededor de 1800 (28), existen los sensores por infrarrojos, posteriormente, en 1878, Samuel Langley crea el bolómetro con dos placas de platino (29), el cual al producirse una elevación de temperatura por radiación infrarroja causaba una variación medible en la resistencia. El primer sensor de proximidad fue creado por Pepperl y Fuchs en 1958 (30), un sensor inductivo que hoy sigue a la vanguardia de la innovación. El primer sensor CCD o también llamado CMOS, utilizado en las cámaras fotográficas digitales fue inventado por Willard Boyle en el año 1969 (31). Honeywell en el año 1969 desarrolla el primer sensor inteligente (32), este se creó para solucionar el problema de compensación de temperatura en los sensores. El desarrollo de estos sensores inteligentes ha permitido aumentar la eficiencia, calidad y velocidad de los procesos industriales, la investigación y el desarrollo científico.

Es decir, un sensor es un tipo de transductor que transforma la magnitud que se quiere medir o controlar, en otra, que facilita su medida. Pueden ser de indicación directa (e.g. un termómetro de mercurio) o pueden estar conectados a un dispositivo adicional (posiblemente a través de un convertidor analógico-digital, un ordenador y un visualizador) de modo que los valores detectados puedan ser leídos por una persona, transmitida para lectura o procesamiento adicional.

Los sensores se pueden clasificar de diversas maneras, algunas de ellas son las siguientes:

- ***En función de la magnitud física o química de medida:*** como, por ejemplo, temperatura, humedad, fuerza, presión, torsión, movimiento, pH, etc.
- ***En función del principio de funcionamiento:*** en este caso hay dos tipos de sensores según cómo generan su señal de salida, ***los pasivos o moduladores*** que necesitan una fuente de alimentación para generarla y ***los activos o generadores*** que no requieren de una fuente de alimentación para generarla. En el caso de los pasivos el estímulo se puede transformar en variaciones de la resistencia eléctrica (e.g. una RTD), de la capacidad (e.g. algunos sensores de humedad), de la inductancia (e.g. un LVDT), entre otros. Algunos ejemplos de los activos serían los termoeléctricos, fotoeléctricos, piezoelectricos, electroquímicos, etc.
- ***En función de cómo se integren los sensores:*** los sensores pueden ser discretos, integrados o inteligentes. En ***los discretos***, la etapa de acondicionamiento está formada por un conjunto de componentes electrónicos independientes que se conectan entre sí junto con el sensor. En el caso de ***los integrados***, como su propio nombre indican, el circuito de acondicionamiento está integrado junto con el sensor, puede ser monolítico o híbrido. ***Los sensores inteligentes*** pueden llevar a cabo operaciones más complejas que los explicados anteriormente, se suelen

ESTADO DEL ARTE

distinguir por realizar algunas funciones como cálculos numéricos, auto calibración, autodiagnóstico, comunicación en red...

- **Según el rango de los datos que generan:** se pueden distinguir dos tipos que serían el llamado **todo o nada (ON-OFF)** que genera una señal de salida con dos posibles estados que se encuentran separados por un umbral. Y el de **medida** que da los valores comprendidos dentro del rango de medida establecido.
- **Según la señal eléctrica que producen:** en este caso se distinguen tres tipos, en **los digitales** la salida está determinada por una serie de valores discretos dentro del rango de medida establecido, esta salida puede ser en serie o en paralelo. En **los analógicos** la salida puede variar de forma continua, pudiendo tomar cualquier valor dentro del rango de medida establecido, la información puede estar en la amplitud de la señal o en otro parámetro. Y en **los temporales** generan la información de salida en función del tiempo, comprobando principalmente los parámetros de frecuencia y fase de las señales obtenidas, las señales generadas pueden ser sinusoidal, triangulares, cuadradas, dientes de sierra...
- **Según como es la relación entre la entrada y la salida:** pudiendo ser de **orden cero, primer orden, segundo orden, tercer orden...** (33) (34) (35)

Los primeros sensores que se utilizaron para el estudio del estado de las plantas fueron los sensores ya existentes para otros fines, fundamentalmente sensores industriales, médicos, bélicos y de otros sectores. Las magnitudes que se necesitaban detectar, derivadas o que influían en las plantas, ya las contemplaban estos sensores, pero era necesario adaptarlos.

Con el tiempo estos sensores fueron evolucionando y mejorándose en todos sus aspectos, para el uso específico que se necesita, es decir, introducirlos en medios más hostiles y mejorando la toma de datos necesarios para el control de lo que se necesita.

Se comprobó que los sensores industriales son muy útiles para la mejora de la producción agrícola, sector fundamental para la sociedad. Actualmente, se ha comprobado que el conjunto de la variedad de la masa forestal es parte fundamental para la solución del cambio climático, tanto por su rendimiento en producto, por ser acumulador de CO₂, como en otros muchos aspectos positivos para la sociedad. Por lo que se han adaptado, mejorado e introducido los sensores agrícolas en la conservación y mejora de la masa forestal.

2.5. Sistemas embebidos

Los sistemas embebidos o empotrados son sistemas de computación que hacen uso de microprocesadores (μP), microcontroladores (μC) o procesadores de señales digitales (DSP) para realizar tareas muy específicas de forma sistematizada, a diferencia de los ordenadores personales (PC) que están diseñados para hacer una amplia gama de tareas. (36) (37)

Una de las características principales es que la mayoría de los componentes se encuentran integrados en la propia placa base y no suelen tener demasiados componentes externos. Estas placas suelen constar de actuadores, sensores y módulos de entrada/salida.

Los sistemas embebidos se suelen catalogar en función de:

- **Su tamaño:** los llamados SES que son de pequeño tamaño y los LES que son de gran tamaño. (38)
- **Su autonomía:** los autónomos, que trabajan de forma independiente sin ningún otro sistema, y los integrados, que forman parte de un sistema mayor que tienen asignadas unas tareas concretas para este sistema.

El primer sistema embebido que se conoce fue el desarrollado por el MIT en 1962 que se utilizó como sistema de guía hacia la Luna de las misiones Apolo. Aunque el primer sistema informático integrado se produjo el año anterior para el sistema de control de aviónica y de guía inercial del misil Minuteman ICBM, este no hacía uso de un circuito integrado ya que utilizaba transistores discretos y puertas lógicas. El primer sistema embebido producido en masa fue el integrado en el Minuteman II.

El primer sistema embebido que incorpora μP se originó en 1970 diseñado por MOS-LSI y se utilizó como ordenador para realizar cálculos de vuelo del F-14 Tomcat. También informaba de datos externos como la velocidad y dirección del viento. Contenía ocho procesadores con chips de memoria.

Los sistemas embebidos salieron del ámbito estatal y militar en la década de los ochenta, fueron elaborados por IBM, donde su comercialización fue generalizada para diversos ámbitos. Esto fue posible gracias a los sistemas en chip (SoC) donde se integra los distintos componentes (μC , μP , memoria, módulos de entrada/salida, etc.) en un solo chip, haciendo que sean más económicos, pero menos flexibles que el PC.

Actualmente, los sistemas embebidos tienen un uso generalizado, siendo utilizados en el ámbito de la salud, entretenimiento, comunicación, parte integral de los sistemas IoT... (39)

2.6. Sistemas de monitorización

Es necesario saber que un sistema de monitorización es un sistema pasivo que se encarga de la vigilancia y el seguimiento del estado del sistema en estudio, a través de distintos sensores que componen la red encargada de recopilar, procesar y almacenar los datos necesarios. Se suele utilizar una interfaz para mostrar o representar los datos obtenidos, pudiendo aparecer a través de la pantalla de un teléfono, de un ordenador... Cuando el sistema en estudio sufre una alteración grave, según lo establecido previamente en este, se puede generar un mecanismo que alerte de esta situación (40). El sistema de monitorización suele venir unido al análisis de datos y al sistema de control.

Normalmente, los datos obtenidos a través del sistema de monitorización se utilizarán para el análisis de la evolución del sistema en estudio o para realizar investigaciones de un ámbito más general, donde no importa el sistema en sí, en las cuales la utilización de esos datos pueda ser valiosa para obtener conclusiones. Un ejemplo podría ser la monitorización de un paciente para obtener sus datos clínicos y, posteriormente, curar a ese paciente (figura 2.6.1). El otro caso sería realizar el estudio de muchos pacientes y obtener sus datos, no para curar solo al paciente, sino para estudiar y curar la enfermedad (figura 2.6.2).



Figura 2.6.1 Ejemplo de análisis de la evolución del sistema en estudio (curación paciente) (41)



Figura 2.6.2 Ejemplo de análisis de los datos para un ámbito general (curación enfermedad) (42) (43)

El sistema de control es un sistema reactivo que, una vez obtenida, procesada y analizada la información del sistema en estudio, selecciona las operaciones más apropiadas de forma automática, en los actuadores, para el correcto funcionamiento del sistema estudiado. Este tipo de sistemas se dividen en dos clases:

- **Lazo abierto:** la salida generada por el sistema no influye en las operaciones del sistema de control. Es decir, no existe una realimentación entre la salida generada por el sistema de control y el controlador, por tanto, la salida no influye en las operaciones del sistema. El mayor inconveniente del lazo abierto es que genera sistemas sensibles a las perturbaciones y cuya exactitud variará en función de la calibración.

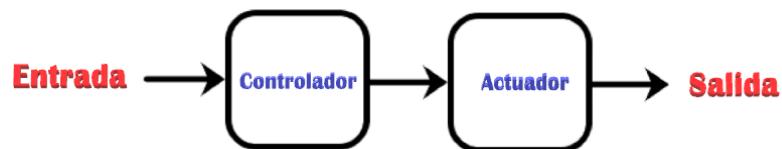


Figura 2.6.3 Ejemplo sistema de control de lazo abierto

ESTADO DEL ARTE

- **Lazo cerrado:** la salida generada por el sistema sí influye en las operaciones del sistema de control. Es decir, existe una realimentación entre la salida generada por el sistema de control y el controlador, por tanto, la salida se compara con la entrada para influir en las operaciones del sistema y así minimizar el error. El lazo cerrado genera sistemas más complejos, pero menos sensibles a las perturbaciones.

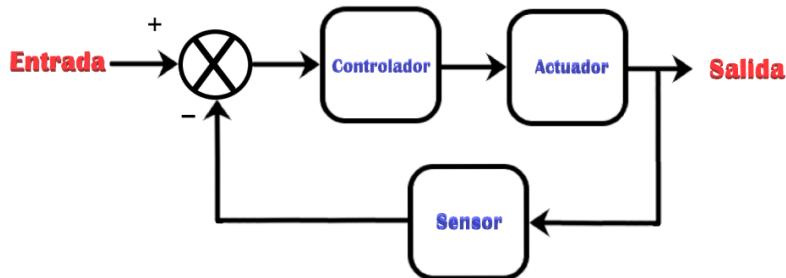


Figura 2.6.4 Ejemplo sistema de control de lazo cerrado

Generalmente, los sistemas más complejos unifican el sistema de monitorización, el análisis de datos y el sistema de control, pudiendo integrarse parte de esta estructura en la nube. Estos sistemas se conectan a través de internet o de una red forman parte del IoT. Un ejemplo se muestra en la figura 2.6.5:

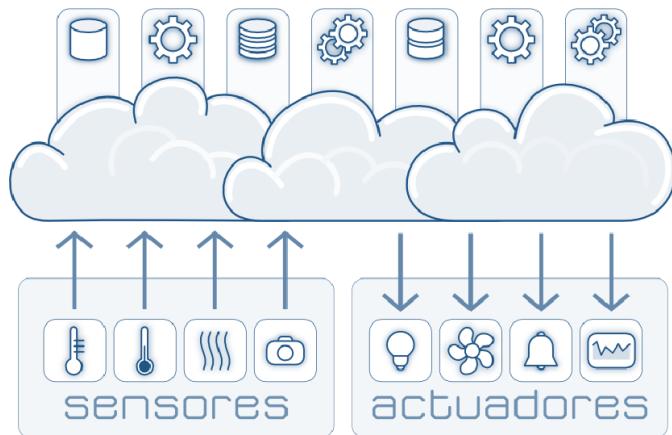


Figura 2.6.5 Ejemplo sistema de monitorización, análisis y control de datos (44)

La monitorización, actualmente, es indispensable para poder gestionar de forma adecuada y automática gran cantidad de información en forma de datos. También es necesario para comprobar el estado de salud de personas, animales, plantas o el correcto funcionamiento de sistemas de producción, informáticos, arquitectónicos...

La monitorización, aunque es relativamente nueva, está experimentando grandes avances, siendo cada vez más eficiente, compleja y extendida.

3. Estudio y elección de las magnitudes y de los sensores

3.1. Estudio de las magnitudes y de los sensores

El objetivo principal de este proyecto es realizar una correcta monitorización de los bosques de quercus a partir de muestras de árboles singulares. Para ello es necesario saber que magnitudes estudiar que aporten información útil.

En este trabajo se va a estudiar las diferentes magnitudes físicas y químicas que definen el estado del árbol, por lo que va a ser primordial saber cuáles se han de estudiar para poder supervisar correctamente la evolución.

La mayor diferencia entre las magnitudes físicas y químicas es que las magnitudes físicas se encargan de estudiar todo tipo de fenómenos y propiedades de los cuerpos independientemente de su naturaleza, mientras que las magnitudes químicas estudian la composición de todas las materias que se encuentran sobre la tierra, de sus transformaciones y de las propiedades que dependen de ella. En el presente proyecto estas dos magnitudes se interrelacionan.

Como se ha dicho anteriormente, el principal objetivo del proyecto es estudiar la evolución del bosque a partir de muestras de árboles singulares, pero para ello es necesario también estudiar las magnitudes ambientales que influyen de forma directa en el desarrollo del bosque. Por tanto, este apartado se subdividirá en otros dos, en el primero se explicarán las magnitudes propias del árbol y en el segundo se explicarán las magnitudes ambientales que influyen sobre los árboles controlados.

Magnitudes propias del árbol:

- **Temperatura en los estomas:** Parámetro fundamental para estudiar la apertura y cierre de los estomas. Sirve para analizar la transpiración del árbol y comprobar la producción de O_2 y CO_2 . Cuando sucede la apertura estomática, el potasio (K) se acumula en las células, atrayendo agua y provoca su expansión, esto produce la apertura de los poros estomáticos. En el cierre estomático, el K se desplaza fuera de las células, llevándose consigo el agua y encogiendo las células, lo que cierra los poros. (45)

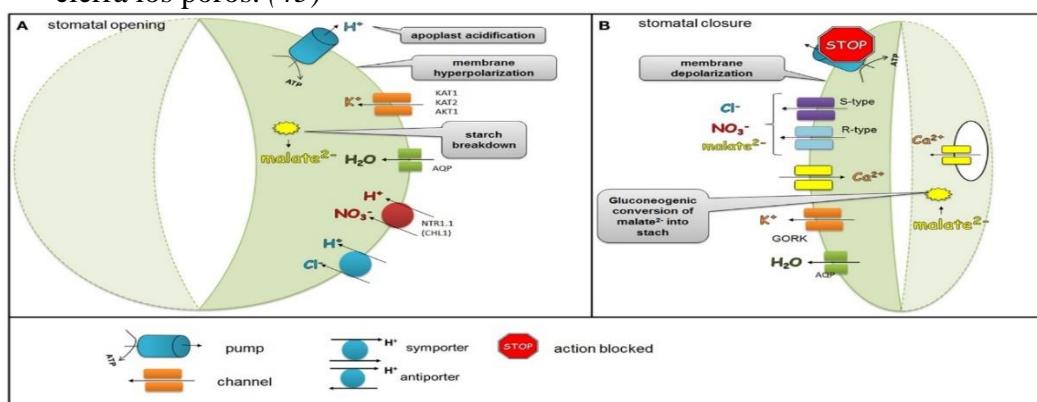


Figura 3.1.1 Apertura y cierre de estoma (46)

- **Humedad emitida por la hoja:** Variable fundamental para ver el estado de salud del ejemplar y ver que esté correctamente hidratado.
- **Diámetro de tallo, peciolo y tronco:** Sirve para estudiar el correcto crecimiento de los árboles y comprobar la resistencia al doblamiento. (47)
- **Tamaño de fruto:** Útil para analizar el proceso de maduración del fruto y comprobar posibles anomalías.
- **Etileno (C_2H_4):** Hormona que afecta al proceso de maduración de los frutos. (48)
- **Flujo de savia:** Se puede comprobar el correcto traslado de hormonas, nutrientes y agua. (49)
- **Oquedad del árbol (A través de sonido percutido):** Sirve para hacer una ecografía del interior del árbol y saber si tiene oquedades producidas por enfermedades o plagas.
- **Resistencia de la madera (A través de corriente eléctrica):** Similar al anterior, pero con impulsos eléctricos.
- **Presión del jugo celular** sobre la hoja: Sirve para estudiar como de hidratada está la hoja.
- **Oxígeno (O_2) en la hoja y en el suelo:** En la hoja sirve para comprobar que la planta está sana, realiza sus funciones básicas fotosintéticas de forma correcta y analizar la apertura de los estomas. En el suelo sirve para comprobar el adecuado funcionamiento de las raíces, al ser parámetro fundamental para la absorción de agua y nutrientes. (50)
- **Dióxido de carbono (CO_2) en la hoja y en el suelo:** En el caso de la hoja, también sirve para vigilar la temperatura estomática y controlar el proceso de transpiración. En el suelo es fundamental para el estudio del cambio climático, indica si el bosque está cumpliendo su objetivo como sumidero de carbono. (51) (52)
- **Densidad de clorofila:** Pigmento fundamental para el metabolismo de la planta al transformar la energía lumínica en energía química. Se puede comprobar el correcto metabolismo y desarrollo del ejemplar. (53)
- **Nitrógeno (N) en el suelo:** Es fundamental para el correcto desarrollo de la planta.
- **Fosforo (P) en el suelo:** Sirve para estudiar el desarrollo de frutos, hojas, ramas y raíces de los árboles.
- **Potasio (K) en el suelo:** Como se ha visto en la temperatura estomática, sirve para regular la humedad de la hoja. Además, de encargarse del flujo del agua y del transporte de nutrientes. (54)
- **Conductividad eléctrica (EC) del suelo:** Al igual que los parámetros anteriores, indica la cantidad de nutrientes que tiene el árbol en su entorno.
- **pH del suelo:** Cada especie prefiere un tipo de suelo con un pH diferente. Es conveniente analizar el pH para comprobar el bienestar del bosque, los nutrientes que dispone e, incluso, puede indicar la existencia de minerales tóxicos. (55)
- **Humedad del suelo:** Sirve para estudiar la absorción de nutrientes y, por ende, el crecimiento del bosque.
- **Observación del estado de la planta:** Haciendo uso de cámaras hiperespectrales para estudiar el estado del bosque entero o de parcelas de este.

ESTUDIO Y ELECCIÓN DE LAS MAGNITUDES Y DE LOS SENSORES

Magnitudes ambientales:

El bosque siempre está influenciado por las características medioambientales externas a los árboles, estas influyen en su crecimiento y desarrollo. Las magnitudes que se van a estudiar son:

- **Viento:** Es una magnitud fundamental para comprobar la correcta evolución de los árboles que se van a estudiar, ya que grandes rachas de vientos pueden ser catastróficas para el desarrollo del árbol, debido a que pueden producir daños mecánicos como el propio vuelco del árbol. Se medirán tanto la dirección como la velocidad y la ráfaga del viento.
- **Temperatura ambiente:** El aumento de las temperaturas y sequías pueden ser determinantes en el desarrollo del bosque, por eso es importante realizar un análisis de la temperatura ambiente, para ver cómo es la evolución del ecosistema.
- **Luz incidente e índice de radiación ultravioleta (UVI):** La luz regula adecuadamente los ciclos fotosintéticos de toda la flora. En cambio, el aumento de la radiación ultravioleta puede ocasionar una disminución de la fotosíntesis y de la producción de biomasa. (56)
- **Humedad del suelo y relativa del aire:** Si la humedad del suelo es baja, la absorción de agua y nutrientes disminuye, lo que se traduce en un crecimiento más lento del bosque. Si el aire es muy húmedo contribuye directamente a los problemas, como enfermedades de las raíces y las hojas. (57) (58)
- **Precipitaciones:** Este parámetro es fundamental, ya que si escasean las precipitaciones el bosque se seca, pero si se inunda provoca la asfixia radicular.
- **Presión atmosférica y del vapor de agua:** Sirve para comprobar cómo es la transpiración de las hojas y la producción de CO_2 , al depender la apertura de los estomas de este parámetro. También influirá en la nutrición del bosque, ya que a baja presión se evapora más fácilmente el agua.

Una vez estudiadas las magnitudes que pueden ser de relevancia en el estudio de los ejemplares, es necesario investigar qué sensores hay en el mercado para realizar correctamente la medición de estas.

Gran parte de este minucioso trabajo se ve reflejado en el [apéndice A](#) del proyecto, donde se pueden observar los distintos sensores que recogen los datos de las magnitudes deseadas.

Estas tablas recogen, para una magnitud en específico, los modelos de los sensores, los precios, si ha sido posible, recogen los tiempos estimados de suministro. Aparte de recoger algunas observaciones que pueden ser de interés, como contactar y el enlace para su adquisición.

3.2. Elección de las magnitudes y de los sensores

Después de haber estudiado las magnitudes y los sensores, es necesario decidir qué tipos de magnitudes y sensores se van a utilizar para la implementación del sistema. Para ello,

ESTUDIO Y ELECCIÓN DE LAS MAGNITUDES Y DE LOS SENSORES

es necesario ser precavidos y estudiar concienzudamente todas las posibles opciones que se han barajado.

En el [apéndice B](#) se recoge el estudio individual de cada sensor, este será de gran relevancia para la elección de los sensores más adecuados para cada magnitud. A continuación, se muestran los parámetros técnicos específicos que más influyen a la hora de comparar los distintos modelos de sensores que miden una misma magnitud:

- **Sensores de temperatura estomática:** Uno de los parámetros más importantes a tener en cuenta, no solo en estos tipos de sensores sino en todos los que vamos a estudiar a continuación, es la temperatura ambiental. Viendo informes meteorológicos y teniendo en cuenta temperaturas extremas, vamos a requerir de sensores que soporten temperaturas entre -36 °C y 48 °C. Para utilizar este sensor será necesario tener en cuenta el tamaño de la hoja, las quercíneas tienen hojas relativamente grandes, en el caso de las encinas, tienen una longitud entre 4 cm y 10 cm y una anchura entre 2 cm y 5 cm.
- **Sensores de humedad emitida por la hoja:** Los parámetros importantes son el rango de temperaturas que puede soportar el sensor, el rango de humedad que es capaz de medir y la forma en que se realiza la medición.
- **Sensores de diámetro de tallo, pecíolo y tronco:** Es importante estudiar la autonomía y la resolución.
- **Sensores de tamaño de fruto:** Similar al caso anterior.
- **Sensores de etileno (C_2H_4):** Es necesario ver cómo y de qué obtiene la medición, el ámbito de uso y el rango de temperaturas que soporta.
- **Sensores de flujo de savia:** Es muy importante la distancia entre las sondas y la forma de colocarlo en el ejemplar de estudio.
- **Sensores de oquedad del árbol (A través de sonido percutido):** Principalmente hay que estudiar la independencia y la autonomía.
- **Sensores de resistencia de la madera (A través de corriente eléctrica):** Lo mismo que en el caso anterior.
- **Sensores de presión del jugo celular sobre la hoja:** Es primordial estudiar la viabilidad a la hora de instalarlo y la durabilidad.
- **Sensores de oxígeno de la hoja (O_2):** En el caso de las hojas es necesario tomar en cuenta el rango de temperatura y humedad que son capaces de soportar. Para medir el O_2 en el suelo es importante ver la forma en la que se realiza la medición.
- **Sensores de dióxido de carbono de la hoja (CO_2):** Similar al caso del O_2 , para las hojas es necesario ver los rangos de humedad y temperatura que son capaces de aguantar. Para el suelo es importante también ver la forma en la que se toma la medida.
- **Sensores de clorofila:** Es necesario comprobar la tecnología que usa, los rangos de temperatura y humedad ambiente que ha de soportar.
- **Sensores de nitrógeno, fósforo y potasio (NPK):** Se ha de tomar en cuenta los rangos de temperatura para que el sensor funcione correctamente.
- **Sensores de conductividad eléctrica (EC) del suelo:** Es imprescindible comprobar el rango de temperaturas que soporta, cómo de resistentes es a la radiación ultravioleta y cómo obtiene la medición.
- **Sensores de pH del suelo:** Hay que tener en cuenta cómo y de qué obtiene la medición, el ámbito de uso y la resistencia a rayos ultravioletas.

ESTUDIO Y ELECCIÓN DE LAS MAGNITUDES Y DE LOS SENSORES

- **Sensores de humedad del suelo:** Es importante tener en cuenta el rango de temperaturas en el que es capaz de trabajar, las resistencias a rayos ultravioletas.
- **Estación meteorológica:** Lo más importante de las estaciones meteorológicas son la cantidad de magnitudes que son capaces de medir ya que están diseñadas para poder soportar ambientes forestales. Sería necesario que se tomasen las magnitudes ambientales expuestas con anterioridad.

La selección de las magnitudes y de los sensores a utilizar están intrínsecamente relacionados, es muy probable que las limitaciones de presupuesto, disponibilidad, dimensiones... de un sensor impidan que en este proyecto se trabaje con determinadas magnitudes. La elección de los sensores está basada en la observación de múltiples parámetros que se estudiarán a continuación:

- **Precio:** Es importante, al disponer de un presupuesto, no solamente tomar en cuenta el precio del sensor, sino su relación con la utilidad que puede aportar al trabajo. Muchos de los sensores son descartados por este parámetro.
- **Distribuidor:** Será necesario conocer el fabricante y el distribuidor del dispositivo, debido a que garantiza la calidad, da confianza en el producto y ofrece garantías.
- **Disponibilidad:** Es de vital importancia que el producto siga en el mercado y no esté descatalogado. De esta forma, si es necesario, se podría replicar el sistema en un futuro.
- **Tiempo de envío:** El tiempo que se tiene para realizar el presente trabajo es limitado, por tanto, es clave contar con los tiempos de envío. Han sido descartados bastantes sensores por este motivo.
- **Precisión:** La precisión en la toma de medidas junto con otros parámetros como la sensibilidad o el error del dispositivo son esenciales a la hora de obtener unos datos fiables.
- **Respuesta temporal:** A la hora de seleccionar el sensor, un parámetro a tener en cuenta en ciertas aplicaciones es el tiempo que tarda el sensor en tomar y transmitir la medida. Al ser el sistema ideado para entornos forestales y evaluar la salud de los árboles en largos períodos de tiempo, el sistema de medición no precisa de mediciones frecuentes ni de baja latencia.
- **Dimensiones:** En próximos apartados se explicará la forma en la que se han de disponer los sensores en el entorno. Es conveniente un análisis previo para saber si dichos sensores son aptos para integrarlos en el ejemplar y en el entorno en función de las características de este.
- **Facilidad a la hora de colocarlo:** Es necesario seleccionar sensores que no requieran de una gran especialización a la hora de situarlos en el ejemplar, ya que podrían dañarlo.
- **Durabilidad:** Al situarse el sistema en un entorno hostil, sin resguardo y de difícil acceso, hay que procurar que los dispositivos que se vayan a utilizar dispongan de las características necesarias para garantizar su objetivo durante un largo periodo de tiempo al no ser fácilmente sustituibles.
- **Empaquetado y protección:** Además, será necesario que dispongan de los recubrimientos necesarios para que estén resguardados de los posibles daños que pudieran recibir tanto del entorno como de la fauna y de la flora.

ESTUDIO Y ELECCIÓN DE LAS MAGNITUDES Y DE LOS SENSORES

- **Tipo de salida:** Para conocer cómo se implementará el sistema, será menester saber el tipo de salida que proporciona el sensor, de esta forma se preverá la necesidad de utilizar sistemas de acondicionamiento entre los sensores y los sistemas embebidos.
- **Protocolo de comunicación:** Ciertos sensores tienen como salida protocolos de comunicación como podría ser I2C, RS232, 1-Wire, entre otros muchos. Será adecuado seleccionar aquellos sensores que mejor se adapten al resto de elementos del sistema.
- **Software (SW) compatible:** De aquellos sensores que tengan la suficiente complejidad para utilizar un SW (Smart Sensors), se seleccionarán aquellos que sean concordantes con el sistema y estén actualizados.
- **Distancia de comunicación:** Aquellos sensores que contengan protocolos de comunicación a distancia, será necesario comprobar el rango de cobertura del que disponen.
- **Consumo:** Al ser un sistema de monitorización en un entorno aislado, será necesario que el sistema tenga una gran autonomía y que logre su propósito a largo plazo sin tener que ser frecuentemente sustituido. Para ello, se precisará de sensores que tengan un bajo consumo de potencia.
- **Tipo de alimentación:** A parte del consumo del sensor, también es relevante saber si utiliza una alimentación “propia” a base de pilas, baterías o si tiene placas solares incluidas. El sistema se alimentará a través de cables con baterías, también se ha hecho un diseño que incluye un panel fotovoltaico. Es necesario saber si son compatibles con el sistema deseado.
- **Neutro para el medio ambiente:** Es recomendable que esté fabricado el dispositivo con materiales que no dañen al entorno en el que se van a situar.

La discusión de cada uno de estos sensores viene recogida en el [apéndice B](#) de esta memoria, es menester recalcar que todo lo expuesto en este apéndice no se ha introducido dentro de la parte principal de la memoria debido a su extensión.

En la siguiente tabla se pueden observar los distintos sensores que se han seleccionado junto con el tipo de magnitudes que miden. Además, se han añadido nuevos sensores para completar el sistema, estos no se han comentado con anterioridad al no medir magnitudes que están implícitamente relacionadas con el estudio del árbol o del ambiente que le influye.

Los nuevos sensores han sido seleccionados al ser considerados de utilidad en un futuro, si se quisiese implementar este sistema como una red interconectada de diversos nodos. Su utilidad principal reside en la localización de los distintos nodos que se tendrían en una gran red. Se ha añadido un sensor de caudalímetro o flujómetro por si fuese necesario un sistema de actuación para el riego y cuidado del bosque, también es útil para cultivos, invernaderos...

Elección de sensores

Modelo	Magnitud	Página web
LAT-B3	Tº estomática	https://ecomatik.de/site/assets/files/21750/usermanual_lat-b3.pdf
PYTHOS-31	Humedad hoja	https://library.metergroup.com/Manuals/20434_PHYTOS31_Manual_Web.pdf
SEN0322	Oxígeno hoja	https://wiki.dfrobot.com/Gravity_I2C_Oxygen_Sensor_SKU_SEN0322
MH-Z19C	CO2 Hoja	https://www.winsen-sensor.com/d/files/mh-z19c-pins%26terminal-type-co2-manual(ver1_2).pdf
Soil NPK Sensor	NPK	https://www.renkeer.com/product/soil-npk-sensor/
SEN0249	pH	https://wiki.dfrobot.com/Gravity__Analog_Spear_Tip_pH_Sensor__Meter_Kit__For_Soil_And_Food_Applications_SKU_SEN0249
314990620	Humedad suelo	https://files.seeedstudio.com/products/101990668/res/RS485%20Soil%20Moisture%20&%20Temperature%20Sensor%20(S-Soil%20MT-02)-Datasheet.pdf
Sensecap	EC, salinidad...	https://files.seeedstudio.com/wiki/Soil_Moisture_Temperature_EC_Sensor/SoilMoisture_Temperature_ECSensorUserManual-S-Temp&VWC&EC-02.pdf
DS18B20	Tº y humedad suelo	https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf
SHT20	Tº y humedad aire	https://wiki.dfrobot.com/SHT20_I2C_Temperature_%26_Humidity_Sensor__Waterproof_Probe_SKU_SEN0227
NEO-6M	GPS	https://www.tme.eu/es/details/neo-6m-0/modulos-gnss-gps/u-blox/neo-6m-0-001/
YF-S201	Caudal	https://www.adafruit.com/product/828
SEN0313	Distancia	https://cdn-shop.adafruit.com/product-files/828/C898+datasheet.pdf
Sainlogic WS 3500 Plus	Estación meteorológica	https://estacionmeteorologaprofesional.com/sainlogic/sainlogic-ws3500-10-en-1/

Tabla 3.2.1 Elección de sensores

4. Implementación del primer nodo sensor

Una vez elegido lo que se quiere medir y los sensores con los que se van a trabajar, se procede a realizar la implementación de los nodos de medición que conformarán el sistema. Estos nodos han de ser escalables, es decir, deben de tener la capacidad de poder añadir más elementos para que sea mejorable y pueda tener una correcta evolución en función de las necesidades futuras.

Se hace necesario explicar que en este trabajo se han realizado dos nodos sensores distintos, ambos utilizan políticas de trabajo diferentes: la forma en la que se realiza la adquisición, procesamiento, almacenamiento de los datos y la forma en la que se alimenta el sistema serán distintos en ambos modelos. Todo esto para realizar un análisis más completo.

En el primer nodo se utilizará un μC Arduino Uno y una Raspberry Pi 4 modelo B, priorizando el uso de sensores analógicos para, de esta forma, desarrollar circuitos de acondicionamiento.

En el segundo nodo se intentará minimizar la cantidad de componentes Hardware (HW) haciendo uso de dos nodos de medición de Dragino que son el NBSN95 y el LSN50. Sendos dispositivos utilizan distintos protocolos de comunicación con el exterior, pero esto no afecta al presente proyecto.

4.1. Planificación del nodo

En este primer apartado se van a exponer todos los pasos que conforman la creación del primer nodo. Se ha de resaltar que el modelo del primer nodo realizado, a su vez puede ser modificado ampliando, reduciendo o cambiando los elementos que lo conforman. También pueden modificarse los modelos de adquisición, preprocesamiento y almacenamiento de los datos obtenidos. Con todo esto se concluye que se trata de un modelo ampliamente transformable y escalable.

A lo largo de este apartado, se expondrá toda la estructura del primer tipo de nodo que se ha diseñado, este constará de diversos sensores que irán a las hojas o al suelo a pie del árbol, para poder tomar correctamente las medidas. Posteriormente, si es necesario, se pasarán los datos que han medido los sensores por circuitos de acondicionamiento que amplifiquen y escalen estos datos.

Ulteriormente, se ha de realizar un SW que recoja y preprocese los datos correctamente en el Arduino. Por último, se transmitirán a la Raspberry Pi a través del protocolo I2C y se almacenarán en un fichero JSON.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

En la figura 4.1.1 se muestra el esquema conceptual de las conexiones, protocolos y alimentación del primer nodo implementado:

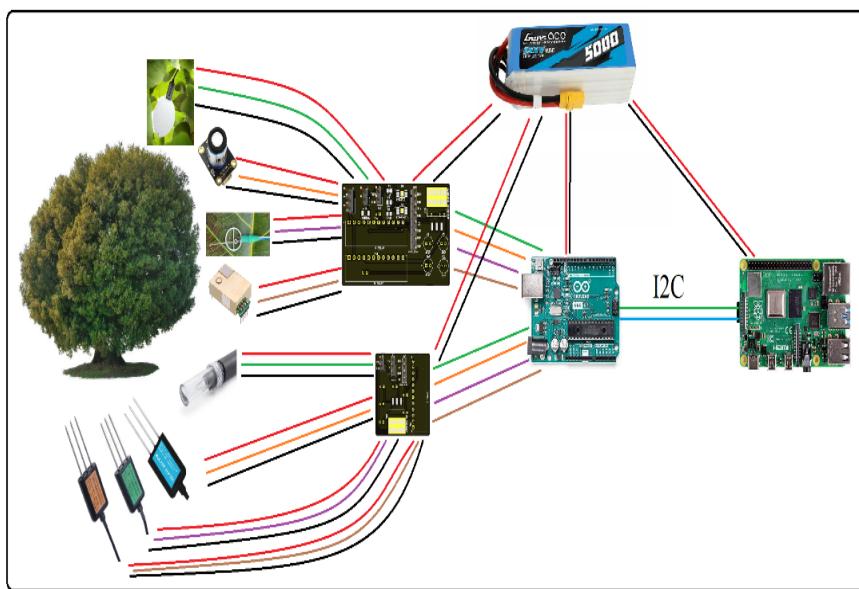


Figura 4.1.1 Esquema conceptual del primer nodo sensor

4.1.1. Datos técnicos, objetivos y soluciones

En primer lugar, es necesario obtener los datos de las magnitudes que se desean medir. Por tanto, se necesitará implementar un sistema que logre recoger los datos proporcionados por los sensores. La herramienta que se utilizará en este primer tipo de nodo para comunicarse con los sensores es un Arduino Uno.

Es necesario realizar circuitos de acondicionamiento para que el Arduino obtenga los datos de los distintos sensores. Comprendiendo el sistema que se quiere implementar, se puede observar que habrá sensores que irán conectados directamente al follaje del árbol, en cambio, otros irán conectados a la tierra en la que este crece. Por tanto, se van a realizar dos circuitos de acondicionamiento, uno para los sensores de hoja y otro para los de suelo, esto facilitará la conexión de los sensores con el Arduino, haciéndolo más versátil y práctico.

A continuación, se van a analizar los distintos requerimientos de cada uno de los sensores, para conocer las necesidades de los mismos y poder generar correctamente dos placas de circuito impreso (PCB) que se adapten lo mejor posible a los consiguientes objetivos:

- **LAT-B3**

- **Datos técnicos:**

- Señal analógica.
 - Alimentación: 2,5 V DC.
 - Rango de medición: -25 °C hasta 70 °C.
 - Rango de salida: 0 V hasta 2,5 V.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

- **Objetivos:**

- Como se ha discutido en el [apartado 3.2](#) de la presente memoria, las temperaturas forestales en España son muy cambiantes en función de la época del año y la zona geográfica, las temperaturas más extremas registradas son de -35,8 °C a 47,6 °C. El caso máximo es aceptable, pero el mínimo es demasiado exigente para los sensores se optará por dejar un margen para ajustar la medición a casos extremos de -15,5 °C a 60,5 °C, principalmente para facilitar la adquisición de componentes. Se tendrá que cambiar de el mínimo del sensor de -25°C a -15,5 °C y el máximo de 70 °C a 60,5 °C. (59)

- **Possibles soluciones:**

- Conectarlo directamente al Arduino Uno, el inconveniente es que el rango de salida del sensor varía entre 0 V y 2,5 V y el Arduino es capaz de medir valores de tensión de entrada entre 0 V y 5 V, por ende, si no se amplifica se perdería mucha resolución.
- Para cambiar el rango, hacer uso de un amplificador de instrumentación con ganancia 2,5 y un offset de -0,625 V, es primordial aclarar que no se va a trabajar con tensiones negativas al complicar en exceso la forma de alimentar las PCB. Además, de que los amplificadores de instrumentación son más caros que los operacionales.
- Para cambiar el rango, hacer uso de un amplificador inversor con ganancia de -2,5 V/V y un offset de 0,625, como la entrada estaría invertida se complicaría un poco la parte de preprocesado de datos, se sustituirá por el circuito que viene a continuación.
- Para cambiar el rango, hacer uso de un circuito diferencial con un amplificador operacional, este tendría una ganancia de 2,5 V/V y un offset de 0,25 V. Esta solución es la que se aplica.

- **PYTHOS-31**

- **Datos técnicos:**

- Señal analógica.
- Alimentación: 2,5 V DC (2mA) hasta 5 V DC (7mA).
- Rango de medición: 0 % RH hasta 100 % RH.
- Rango de salida: 0 V hasta 1,25 V.

- **Objetivos:**

- El objetivo principal es tener una buena resolución para obtener el rango de medición completo.

- **Possibles soluciones:**

- Conectarlo directamente al Arduino Uno, el inconveniente es que el rango de salida del sensor varía entre 0 V y 1,25 V y el Arduino

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

acepta valores de tensión de entrada entre 0 V y 5 V, por ende, si no se amplifica se perdería mucha resolución.

- Usar un amplificador de instrumentación con ganancia 4 V/V y offset de 0 V; es necesario recalcar que los amplificadores de instrumentación son más caros que los operacionales.
- Hacer uso de un circuito diferencial con un amplificador operacional, este tendría una ganancia de 4 V/V y un offset de 0 V. Esta solución es la que se aplica.

- **SEN0322**

- **Datos técnicos:**

- Protocolo I2C.
 - Alimentación: 3,3 V DC hasta 5 V DC.
 - Rango de medición: 0 % Vol hasta 30 % Vol.

- **Objetivos:**

- Obtener la medición lo más cercana a la realidad.

- **Posibles soluciones:**

- Plug & Play: este sensor al ser digital se va a conectar por protocolo I2C directamente al Arduino Uno y se utilizará el código proporcionado por el fabricante.

- **MH-Z19C**

- **Datos técnicos:**

- Señal PWM o UART.
 - Alimentación: 3,6 V DC hasta 5,5 V DC.
 - Rango de medición: 0 ppm hasta 2000 ppm.

- **Objetivos:**

- Obtener la medición lo más cercana a la realidad.

- **Posibles soluciones:**

- Hacer uso de la interfaz a nivel SW de la UART. A pesar de ser una posible solución, al integrarlo con el resto del código genera errores haciéndolo incompatible.
 - Hacer uso de un filtro a nivel HW que promedie el ciclo de trabajo que devuelve la señal PWM. Es decir, hacer un circuito integrador que devuelva un valor de la continua en función del área que genera el pulso cuadrado. Aunque es una solución práctica y útil, que sería la solución con otros dispositivos.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

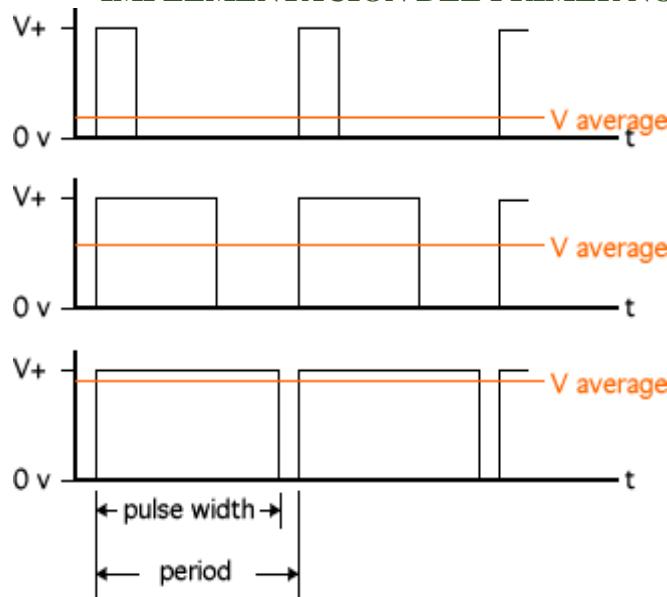


Figura 4.1.2 Tensión promedio de un filtro de una señal PWM (60)

- Hacer uso de un filtro a nivel SW que promedie el ciclo de trabajo que devuelve el sensor. El Arduino Uno tiene pines especializados en generar y recibir señales PWM, junto con el código del fabricante. No sería necesario complicarse en realizar la solución anterior. Esta es la solución que se aplica.

- **Soil NPK Sensor**

- **Datos técnicos:**

- Protocolo RS485.
 - Alimentación: 5 V DC hasta 30 V DC.
 - Rango de medición: 0 mg/kg hasta 1999 mg/kg.

- **Objetivos:**

- Obtener la medición lo más cercana a la realidad.

- **Possibles soluciones:**

- Plug & Play: este sensor al ser digital se va a conectar por protocolo RS485 directamente al Arduino Uno y se utilizará el código proporcionado por el fabricante.

- **SEN0249**

- **Datos técnicos:**

- Señal analógica.
 - Alimentación: 5 V DC.
 - Rango de medición: 0 pH hasta 10 pH.
 - Rango de salida: 0 V hasta 4 V.

- **Objetivos:**

- El objetivo principal es tener una buena resolución para obtener el rango de medición completo.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

- **Possible soluciones:**

- No haría falta el acondicionamiento porque el Arduino Uno tiene resolución de 5 V /1024 (2^{10}) que es muy superior a la del sensor de 4V / 256 (2^8). Por ende, será posible medir los valores del sensor sin la pérdida de información.

- **314990620**

- **Datos técnicos:**

- Sensor analógico o protocolo RS485.
- Alimentación: 3,6 V DC hasta 30 V DC.
- Rango de medición temperatura: -40 °C hasta 80 °C.
- Rango de medición humedad: 0 % hasta 100 %.
- Rango de salida: 0 V hasta 2 V.

- **Objetivos:**

- El objetivo principal es tener una buena resolución para obtener el rango de medición completo.

- **Possible soluciones:**

- Conectarlo directamente al Arduino Uno, el inconveniente es que el rango de salida del sensor varía entre 0 V y 2 V y el Arduino acepta valores de tensión de entrada entre 0 V y 5 V, por ende, si no se amplifica se perdería mucha resolución.
- Hacer uso de un amplificador de instrumentación con ganancia 2,5 V/V y un offset de 0 V, es necesario recalcar que los amplificadores de instrumentación son más caros que los operacionales.
- Hacer uso de un circuito diferencial con un amplificador operacional, este tendría una ganancia de 2,5 V/V y un offset de 0 V. Esta solución se intenta aplicar.
- Plug & Play: este sensor al ser digital se va a conectar por protocolo RS485 directamente al Arduino Uno y se utilizará el código proporcionado por el fabricante. Esta es la solución que finalmente se aplica.

- **Sensecap**

- **Datos técnicos:**

- Protocolo RS485.
- Alimentación: 3,6 V DC hasta 30 V DC.
- Rango de medición temperatura: -40 °C hasta 80 °C.
- Rango de medición humedad: 0 % RH hasta 100 % RH.
- Rango de medición EC: 0 $\mu\text{s}/\text{cm}$ hasta 20000 $\mu\text{s}/\text{cm}$.
- Rango de medición salinidad: 0 hasta 20000 mg/L.
- Rango de medición TDS: 0 hasta 20000 mg/L.
- Rango de medición epsilon: 0 hasta 82 unidades.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

- **Objetivos:**
 - Obtener la medición lo más cercana a la realidad.
- **Posibles soluciones:**
 - Plug & Play: este sensor al ser digital se va a conectar por protocolo RS485 directamente al Arduino Uno y se utilizará el código proporcionado por el fabricante.

4.1.2. Diseño de las posibles soluciones de acondicionamiento

Una vez analizadas las posibles soluciones expuestas se puede comprobar que, para aquellos sensores que no tengan salida analógica, se hará uso de SW para la integración de los sensores en el Arduino Uno. En cambio, para la mayoría de los sensores analógicos se hará uso de un circuito de acondicionamiento que sea capaz de acomodar la señal al rango de entrada del Arduino.

Posteriormente, se procede a realizar el estudio de las distintas soluciones que se han explicado para los sensores con salida analógica, se pueden describir tres tipos de configuraciones que amplifiquen la señal y añadan un offset.

Como el sistema es susceptible a errores y, además, la señal de estos sensores varía muy lentamente a lo largo del tiempo, se hará un filtrado estricto paso bajo para eliminar la mayoría de estos posibles errores y ruidos que afecten a la señal. Por tanto, se realizará un filtro paso bajo para las señales analógicas con una frecuencia de corte de 10 Hz.

Para llevar a cabo un correcto análisis de las distintas soluciones planteadas previamente, se expondrán los distintos circuitos, elementos y ecuaciones que son necesarios para avanzar con el proyecto:

- **Diseño con amplificador de instrumentación:** La primera de las soluciones propuestas, en el caso de los sensores con salidas analógicas, es la que se va a relatar a continuación. El objetivo de este diseño es crear un circuito basado en un amplificador de instrumentación como el que se muestra a continuación:

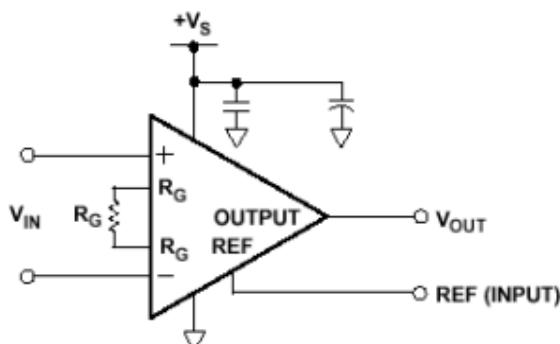


Figura 4.1.3 Circuito electrónico de un amplificador de instrumentación (61)

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

El amplificador de instrumentación es un elemento complejo que consta de dos o más amplificadores operacionales concatenados, ideado para tener gran rechazo al modo común y tener una gran impedancia de entrada. La fórmula que simplifica el funcionamiento de este dispositivo es:

$$V_{OUT} = G \cdot (V_{IN+} - V_{IN-}) + V_{REF} \quad [1]$$

Siendo:

- V_{OUT} : Tensión de salida del amplificador de instrumentación.
- G : Ganancia del amplificador, está dependerá de la resistencia R_G y de la resistencia propia que indique el fabricante del producto. Por ejemplo, la ganancia en el amplificador de instrumentación INA114 es de:

$$G = 1 + \frac{50 \text{ k}\Omega}{R_G} \quad [2]$$

- V_{IN} : Tensión de las entradas del amplificador de instrumentación.
- V_{REF} : Tensión de referencia que se introduce en el terminal REF del dispositivo, este sirve para añadir un offset a la señal de salida del amplificador de instrumentación.

El diseño de este circuito es bastante simple dado que solamente se necesita la resistencia R_G para ajustar la ganancia. La señal se conectaría al terminal positivo del sensor y el negativo a tierra, el offset se ajustaría con el terminal REF en función del sensor.

No se va a utilizar esta solución al precisar de tensiones negativas para ajustar el valor de tensión de referencia. Como se ha establecido anteriormente, se evitará el uso de tensiones negativas para simplificar los circuitos de acondicionamiento a implementar.

- **Diseño con amplificador de instrumentación en configuración inversora:** Esta solución es idéntica a la diseñada en el punto anterior, la diferencia estriba en que se cambiarían la disposición de los pines de entrada, es decir, el pin de entrada positivo iría conectado a tierra y el negativo a la salida del sensor. Por ende, todas las tensiones de referencia utilizarían tensiones positivas, solventando el inconveniente del diseño anterior.

Este diseño tiene dos problemas, el primero es que invertiría el valor de la señal recibida por el Arduino, pero realmente se podría hacer una implementación SW que trabajase correctamente con esta señal y la invirtiera.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

El segundo problema, y por el que se descarta esta solución, es el precio del dispositivo. Es extremadamente simple el circuito diseñado, pero al ser tan caro se pueden hacer diseños ligeramente más complejos que cumplan la misión, pero menos costosos.

- **Diseño con amplificador diferencial:** La última de las soluciones, y la que se va a implementar, es la de un amplificador diferencial con un circuito como el que se muestra a continuación:

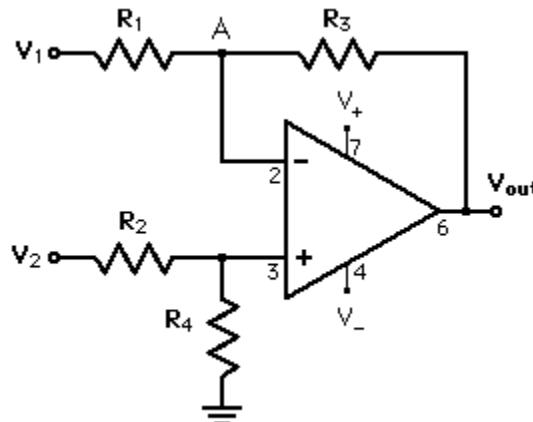


Figura 4.1.4 Circuito electrónico de un amplificador diferencial (62)

Este diseño tiene la siguiente expresión que lo define:

$$V_{OUT} = V_2 \frac{(R_3 + R_1) \cdot R_4}{(R_4 + R_2) \cdot R_1} - V_1 \cdot \frac{R_3}{R_1} \quad [3]$$

Siendo:

- V_{OUT} : Tensión de salida del amplificador de instrumentación.
- R_X : Resistencias que modifican la ganancia y el offset del circuito.
- V_1 : Tensión de referencia.
- V_2 : Tensión proporcionada por el sensor.

En esta solución se simplificará el diseño, igualando la primera resistencia con la segunda y la tercera con la cuarta. A este nuevo diseño se le conoce como amplificador restador, se muestra su circuito junto con la expresión que lo define:

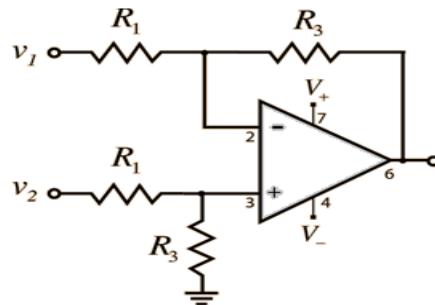


Figura 4.1.5 Circuito electrónico de un amplificador restador

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

$$V_{OUT} = (V_2 - V_1) \frac{R_3}{R_1} \quad [4]$$

A parte de las soluciones vistas, se barajaron otras alternativas, pero para no alargar el proyecto, solamente se han mostrado las más relevantes. El resto requerían de más elementos o su realización era menos plausible.

4.1.3. Simulación de los circuitos de acondicionamiento

Para terminar este subapartado se van a mostrar las simulaciones de los circuitos de acondicionamiento para cada uno de los sensores analógicos utilizados en este tipo de nodo. Graficando la salida generada por estos circuitos respecto a un barrido de tensión utilizando el rango de tensiones de salida propio del sensor, y también se muestra el diagrama de Bode para comprobar que el filtrado se realiza adecuadamente.

- **LAT-B3:** Para diseñar el circuito de la figura 4.1.6 se ha hecho uso de los códigos desarrollados en el [apéndice D](#). En este caso las tensiones que generaría el sensor a la salida para los límites externos de -15,5 °C y 60,5 °C corresponderían con las tensiones de 0,25 V y 2,25 V que se muestra en la figura 4.1.7, comprobando que lo graficado coincide con lo calculado al verse como a 0,25 V de entrada empieza a aumentar la tensión de salida y que llega a los 5 V cuando se tiene una tensión de entrada de 2,25 V, tensiones superiores no las detectaría el Arduino. Además, se ve en la figura 4.1.8 una caída de 3 dB a la frecuencia de 10 Hz, comprobando que el filtrado se realiza correctamente.

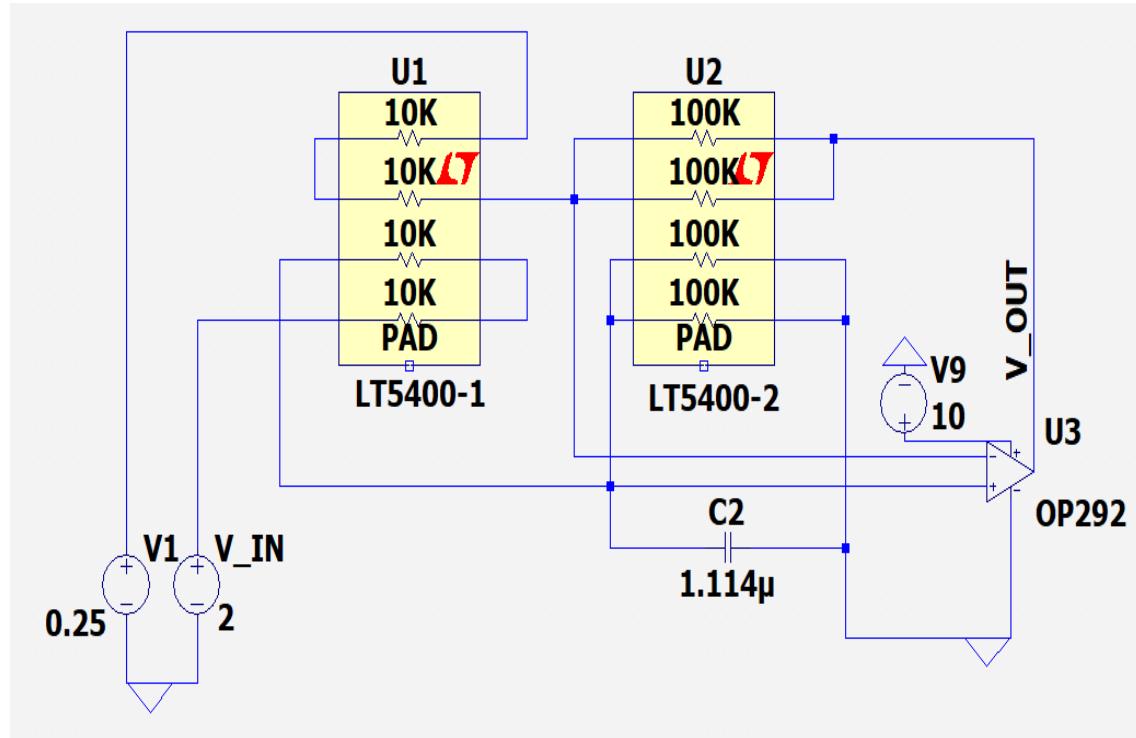


Figura 4.1.6 Circuito de acondicionamiento para el sensor LAT-B3

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

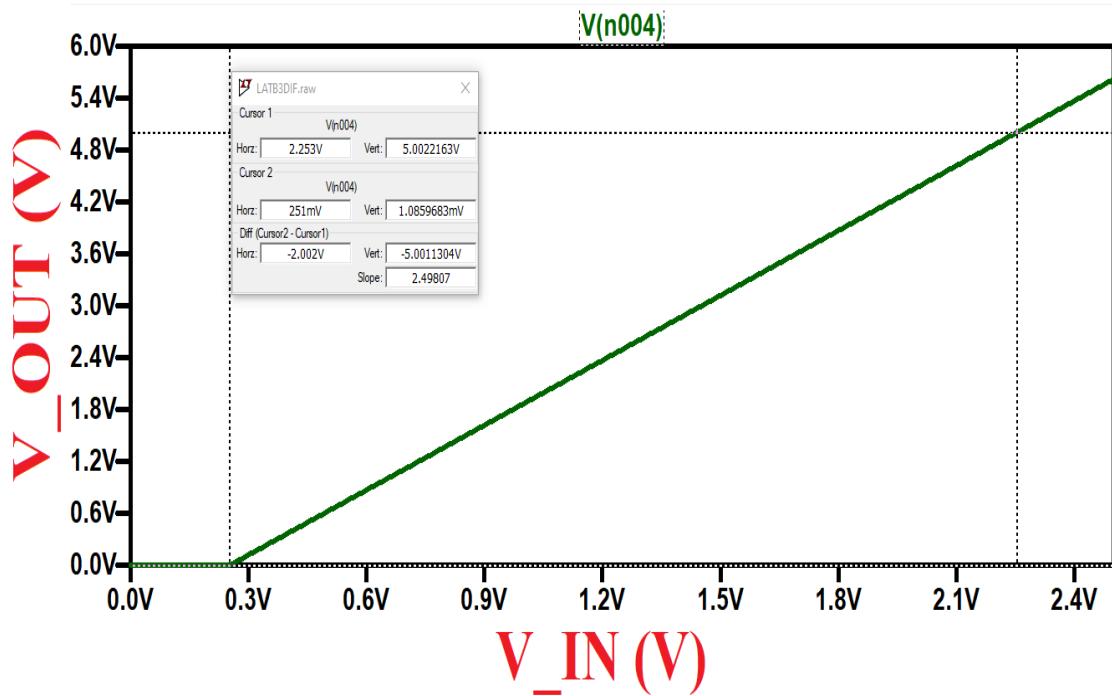


Figura 4.1.7 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor LAT-B3.

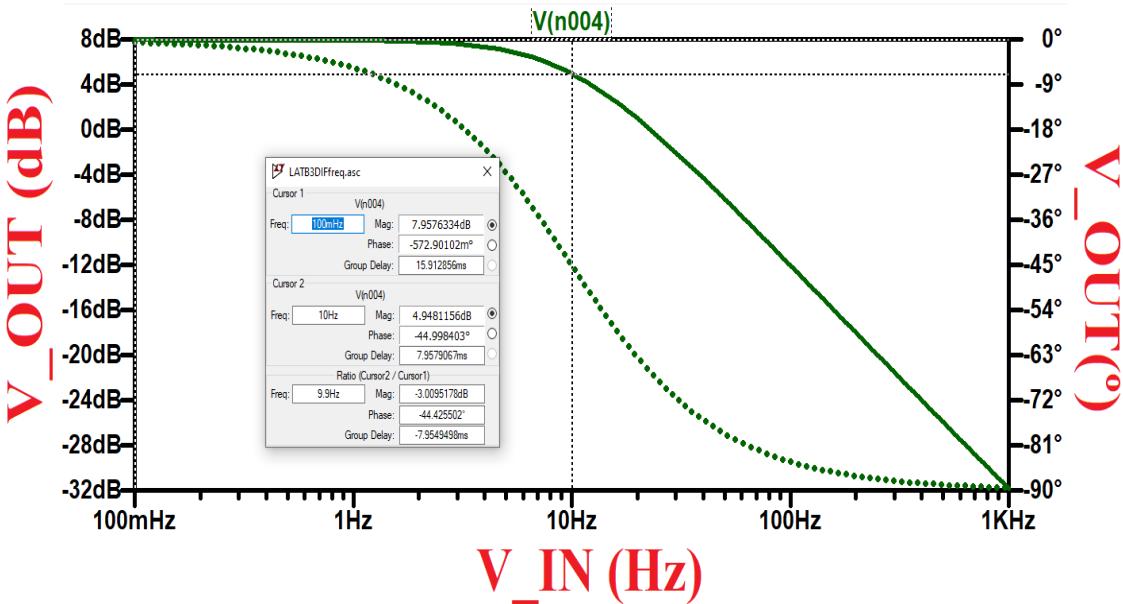


Figura 4.1.8 Diagrama de Bode para comprobar el filtrado del sensor LAT-B3

- **PYTHOS31:** Para diseñar el circuito de la figura 4.1.9 se ha hecho uso de los códigos desarrollados en el [apéndice D](#). En este caso las tensiones que generaría el sensor a la salida para los límites externos de 0 % RH y 100 % RH corresponderían con las tensiones de 0 V y 1,25 V que se muestra en la figura 4.1.10, comprobando que lo graficado coincide con lo calculado al verse como a 0 V de entrada se tienen 0 V de salida y que llega a los 5 V cuando se tiene una tensión de entrada de 1,25 V. Además, se ve en la figura 4.1.11 una caída de 3 dB a la frecuencia de 10 Hz, comprobando que el filtrado se realiza correctamente.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

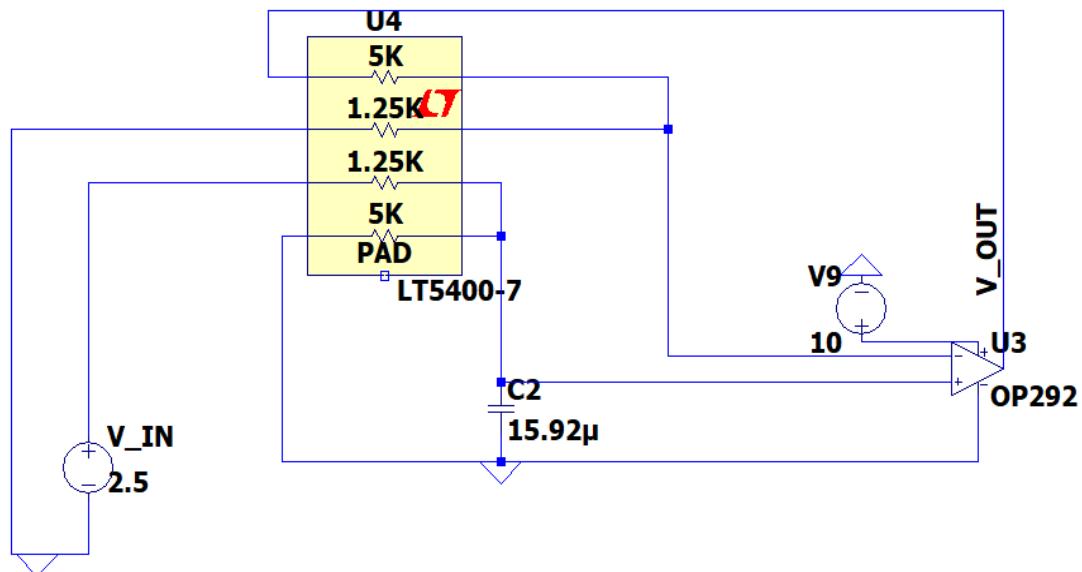


Figura 4.1.9 Circuito de acondicionamiento para el sensor PYTHOS31

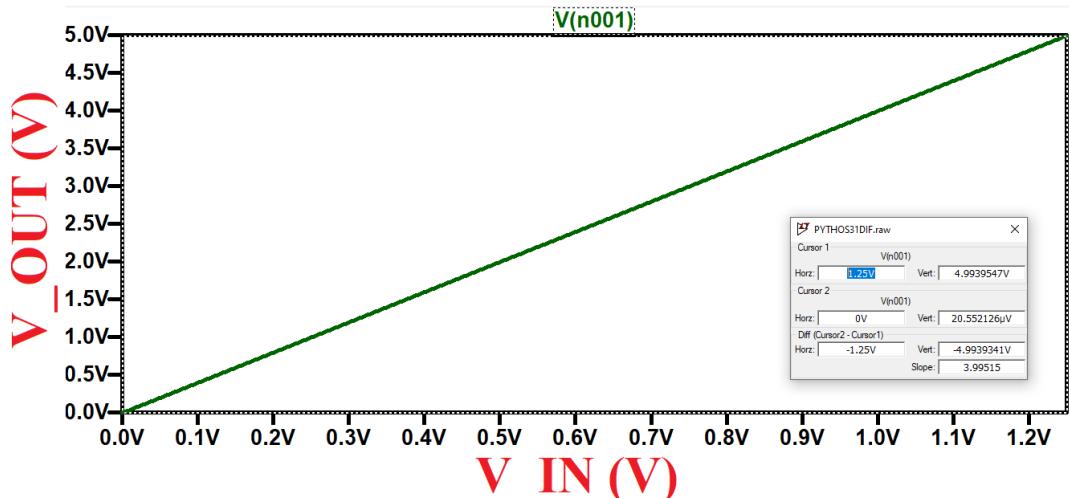


Figura 4.1.10 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor PYTHOS31

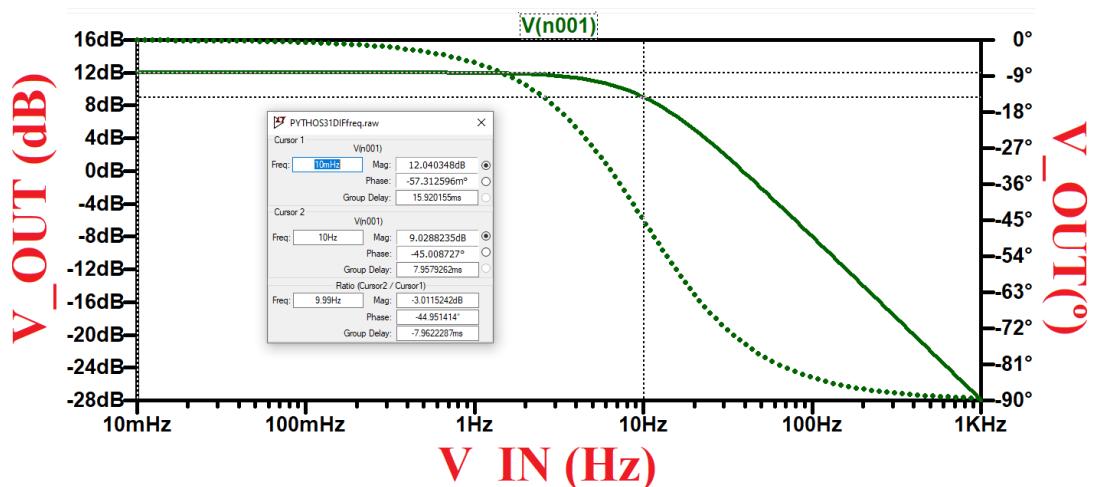


Figura 4.1.11 Diagrama de Bode para comprobar el filtrado del sensor PYTHOS31

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

- 314990620: Al igual que en los casos anteriores, para diseñar el circuito de la figura 4.1.12 se ha hecho uso de los códigos desarrollados en el [apéndice D](#). En este caso las tensiones que generaría el sensor a la salida para los límites externos de 0 % y 100 % de humedad corresponderían con las tensiones de 0 V y 2 V que se muestran en la figura 4.1.13, comprobando la gráfica coincide con lo calculado al verse como a 0 V de entrada se tienen 0 V de salida y que llega a los 5 V cuando se tiene una tensión de entrada de 2 V. Además, se ve en la figura 4.1.14 una caída aproximada de 3 dB a la frecuencia de 10 Hz, comprobando que el filtrado se realiza correctamente.

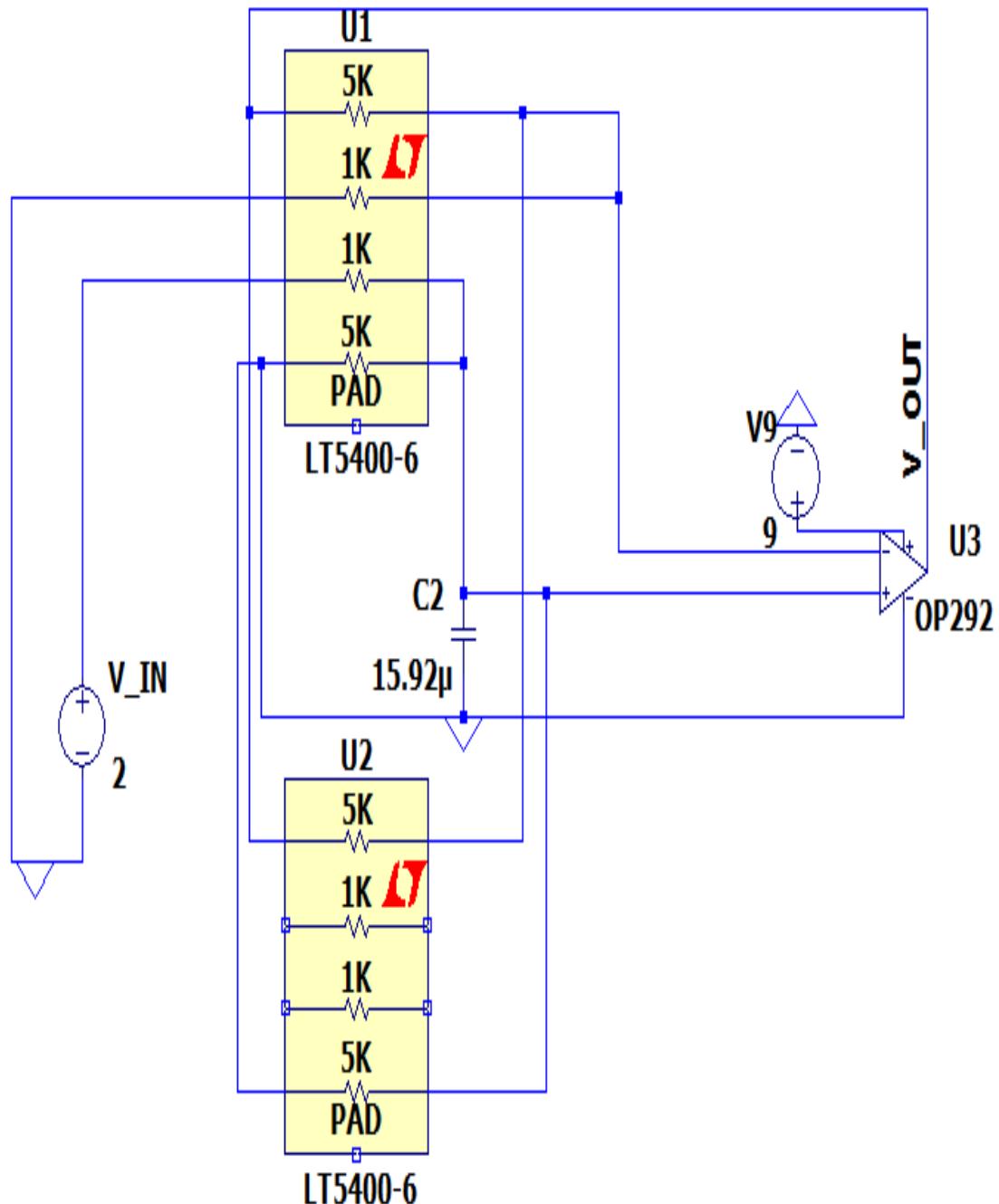


Figura 4.1.12 Circuito de acondicionamiento para el sensor 314990620

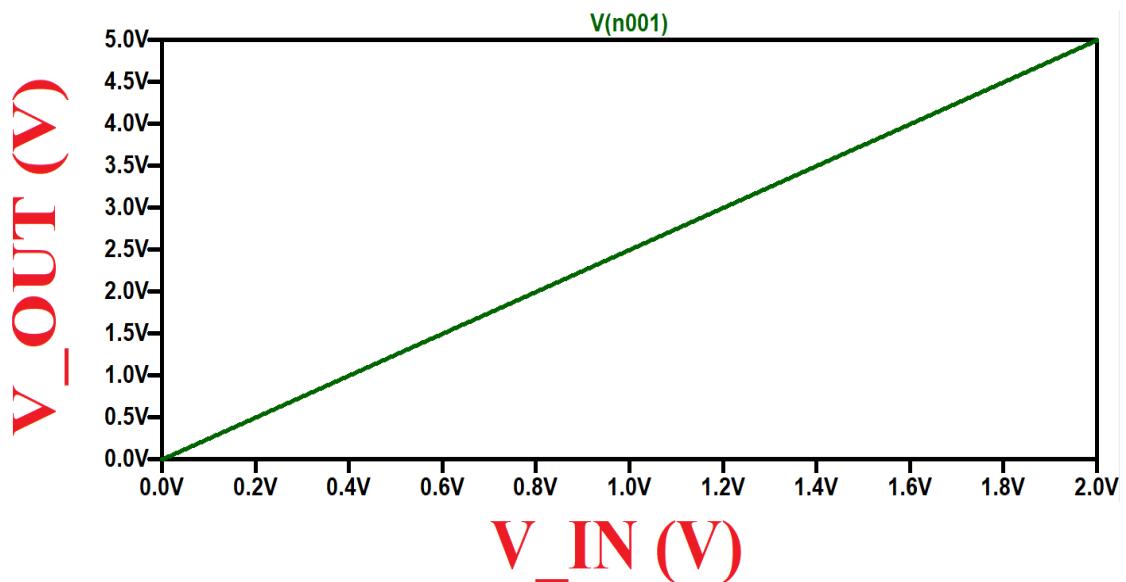


Figura 4.1.13 Barrido de tensión comprobando los rangos de salida respecto los de entrada del circuito de acondicionamiento del sensor 314990620

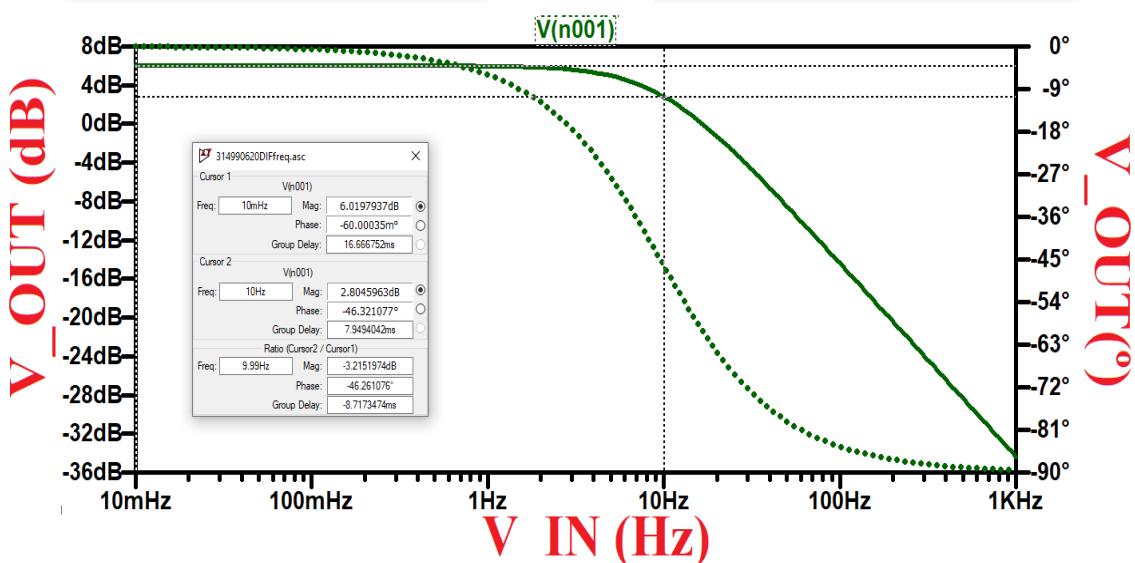


Figura 4.1.14 Diagrama de Bode para comprobar el filtrado del sensor 314990620

Como nota final se ha de decir que en el [apéndice D](#) se recogen el resto de los circuitos que se han planteado en el proyecto, incluso para sensores descartados. Los cálculos de los componentes están indicados en el [apéndice C](#).

4.2. Alimentación del nodo

En este subapartado se va a dilucidar la forma de suministrar energía para el correcto funcionamiento del nodo. A pesar de que esta característica es fundamental para el proyecto, no se profundizará en exceso dada la complejidad y la naturaleza "transversal" del diseño del sistema.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Primero se han de examinar las especificaciones técnicas de los distintos elementos que se van a utilizar en el sistema. Empezando por los sensores, se verifica que estos tienen un bajo consumo energético, de pocos vatios.

Modelo	Magnitud	Consumo (mW)
LAT-B3	Tº estomática	Despreciable
PYTHOS-31	Humedad hoja	40
SENO322	Oxígeno hoja	0,55
MH-Z19C	CO ₂ Hoja	200-625
Soil NPK Sensor	NPK	≤ 150
SENO249	pH	Indefinido
314990620	Humedad suelo	< 960
Sensecap	EC, salinidad...	144

Tabla 4.2.1 Consumo de potencia de los sensores del primer nodo

En el diseño de la placa, el cual se explicará en el siguiente subapartado, se ha intentado reducir la cantidad de componentes que consumen potencia, usando conversores y elementos energéticamente eficientes. También se puede ver el consumo energético del Arduino Uno y la Raspberry Pi 4 en la siguiente tabla:

Placas\Modo de consumo	Consumo en reposo (W)	Consumo promedio (W)	Consumo máximo (W)
Arduino Uno	< 0,45	0,325 – 0,45	2,175 – 2,4
Raspberry Pi 4	0,5 – 3	1,2 – 5,85	2,5 – 6,25

Tabla 4.2.2 Consumo de potencia del Arduino Uno y la Raspberry Pi 4 (63) (64) (65) (66)

Una vez vistos los consumos, es necesario diseñar el sistema de alimentación que se va a utilizar, hay múltiples tipos de baterías dependiendo del sistema objetivo. Como tenemos un sistema estático, en el cual es primordial que esté dotado de una gran autonomía, las dos mejores opciones serían:

- **Alimentación con batería:** Se podría hacer uso de baterías como las de plomo-ácido o las de litio-ferrofosfato, ambas de larga duración, capaces de operar hasta 10 años. A pesar de haber ideado un sistema de bajo consumo, obtener el objetivo de implementar un sistema con larga autonomía puede ser un reto.

Es importante señalar que la vida útil de algunos de los sensores utilizados en el diseño es limitada, de tan solo un par de años. Por tanto, si igualmente hay que modificar los sensores cada “poco” tiempo, se podría hacer uso de baterías con menor capacidad, con duración más limitada y que se cambie a la vez que los sensores.

- **Alimentación con batería y fuente de energía:** En este caso se haría uso de baterías de menor capacidad y vida útil, pero recargables, como las de ion de litio, añadiéndoles una fuente de energía renovable externa, como puede ser un panel

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

fotovoltaico o un pequeño aerogenerador. El precio de este sistema aumentaría considerablemente. En cambio, esta opción, al recoger energía de forma periódica, permitiría una mayor escalabilidad del sistema.

En este proyecto se utilizarán estos dos modelos de alimentación, en este primer nodo se hará uso del primer modelo, sirviéndose de la batería “Gens Ace” de 5000mAh y 25.9V. Esta batería no es de las descritas en el primer modelo, al ser LIPO, pero es la facilitada y suficiente para poderla utilizar durante el tiempo de toma de medidas.



4.3. Conexión entre sensores-Arduino y los circuitos de acondicionamiento

Figura 4.2.1 Batería usada para alimentar el primer nodo implementado (82)

acondicionamiento

Para continuar correctamente con el proyecto, se diseñan unas PCB con un doble propósito, para realizar el acondicionamiento de los sensores analógicos que necesitan de un sistema de amplificación y ajuste, y para alimentar correctamente los sensores y los dispositivos de adquisición (el Arduino y la Raspberry Pi).

Se van a diseñar tres PCB, una para la alimentación del Arduino y la Raspberry y las otras dos para la alimentación de los sensores de hoja y de suelo y su acondicionamiento. Por tanto, se procede a explicar el diseño de los distintos circuitos impresos mientras se explica el porqué de su diseño.

Placa de alimentación

Esta PCB solamente se va a encargar de realizar la alimentación del Arduino Uno y la Raspberry Pi 4. Al investigar la alimentación que necesitan sendos dispositivos, se puede comprobar que es posible alimentarlos desde sus pines, en el caso del Arduino Uno se suministra la alimentación a través de los pines VIN y GND mostrados en la figura 4.3.1. Como se recoge en su hoja de características, la tensión de alimentación puede oscilar entre los 6 V y los 20 V, para este proyecto se utilizará una alimentación de 9 V.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

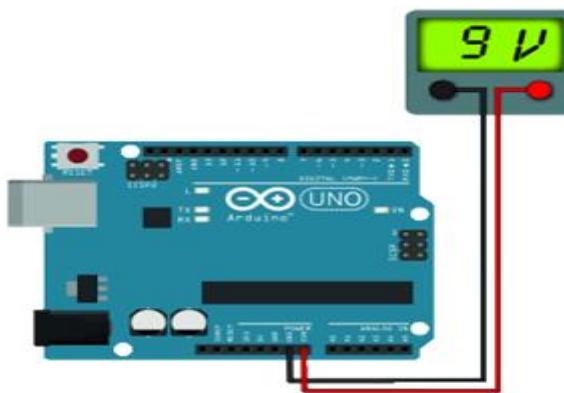


Figura 4.3.1 Alimentación por pines Arduino Uno (67)

En cambio, la Raspberry Pi 4 modelo B tiene una alimentación mucho más restrictiva, siendo menester 5 V con 3A. Los pines para alimentarla son los que se muestran en la figura 4.3.2.

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I ² C)	DC Power 5v	04
05	GPIO03 (SCL1 , I ² C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I ² C ID EEPROM)	(I ² C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Figura 4.3.2 Pines de alimentación Raspberry Pi 4 modelo B

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Conociendo todo lo anteriormente indicado, se procede a diseñar la PCB, para el diseño de esta y las siguientes se hará uso de la herramienta KiCad 6.0.0. Se divide este diseño en tres hojas jerárquicas que son las que se muestran a continuación, en la figura 4.3.3.

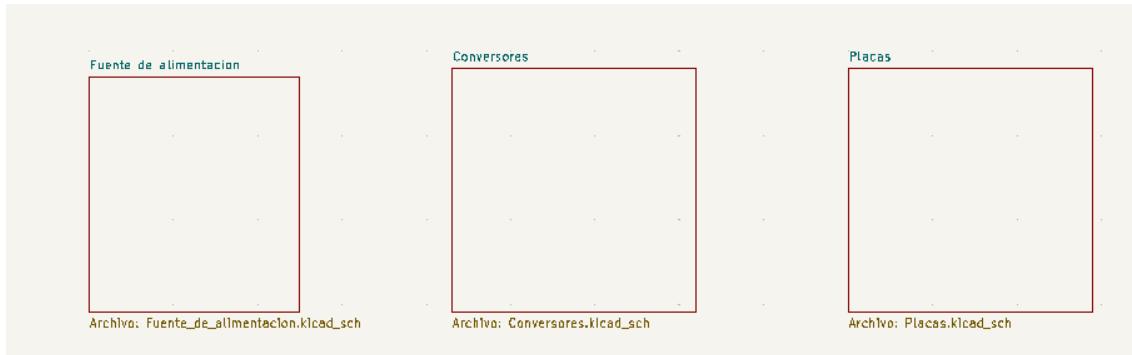


Figura 4.3.3 Hoja jerárquica principal PCB Alimentación

Tanto “Fuente_de_alimentacion.kicad_sch” como “Placas.kicad_sch” contienen los pines a través de los cuales la PCB se conectará con la batería y dará alimentación al Arduino y a la Raspberry respectivamente.

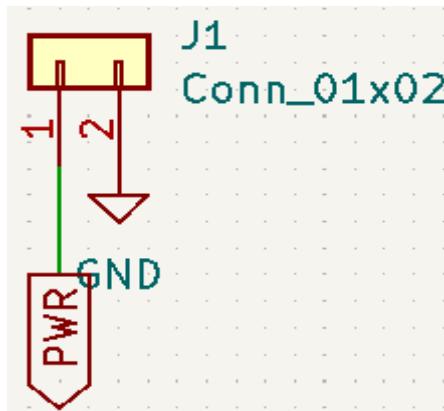


Figura 4.3.4 Funte_de_alimentacion.kicad_sch

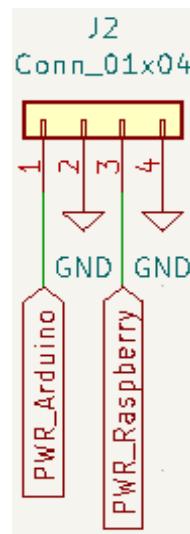


Figura 4.3.5 Placas.kicad_sch

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

En la hoja jerárquica “*Conversores.kicad_sch*” se muestran los dos conversores utilizados para alimentar la placa de Arduino y la Raspberry Pi.

Para alimentar el Arduino de manera adecuada se hace uso del conversor TSR_1 – 2490, que proporciona 9 V con 1 A, es decir, genera 9 W de potencia, más que suficiente para alimentar la placa.

En cambio, para suministrar energía a la Raspberry Pi se hace uso del conversor K7805 – 3AR3. Este proporciona una tensión de salida de 5 V con una corriente de 3 A, justo los 15 W que se necesitan para alimentar la placa.

En ambos casos se añaden condensadores a la entrada y a la salida de los conversores para, de este modo, poder estabilizar la señal de entrada y salida con un filtrado paso bajo.

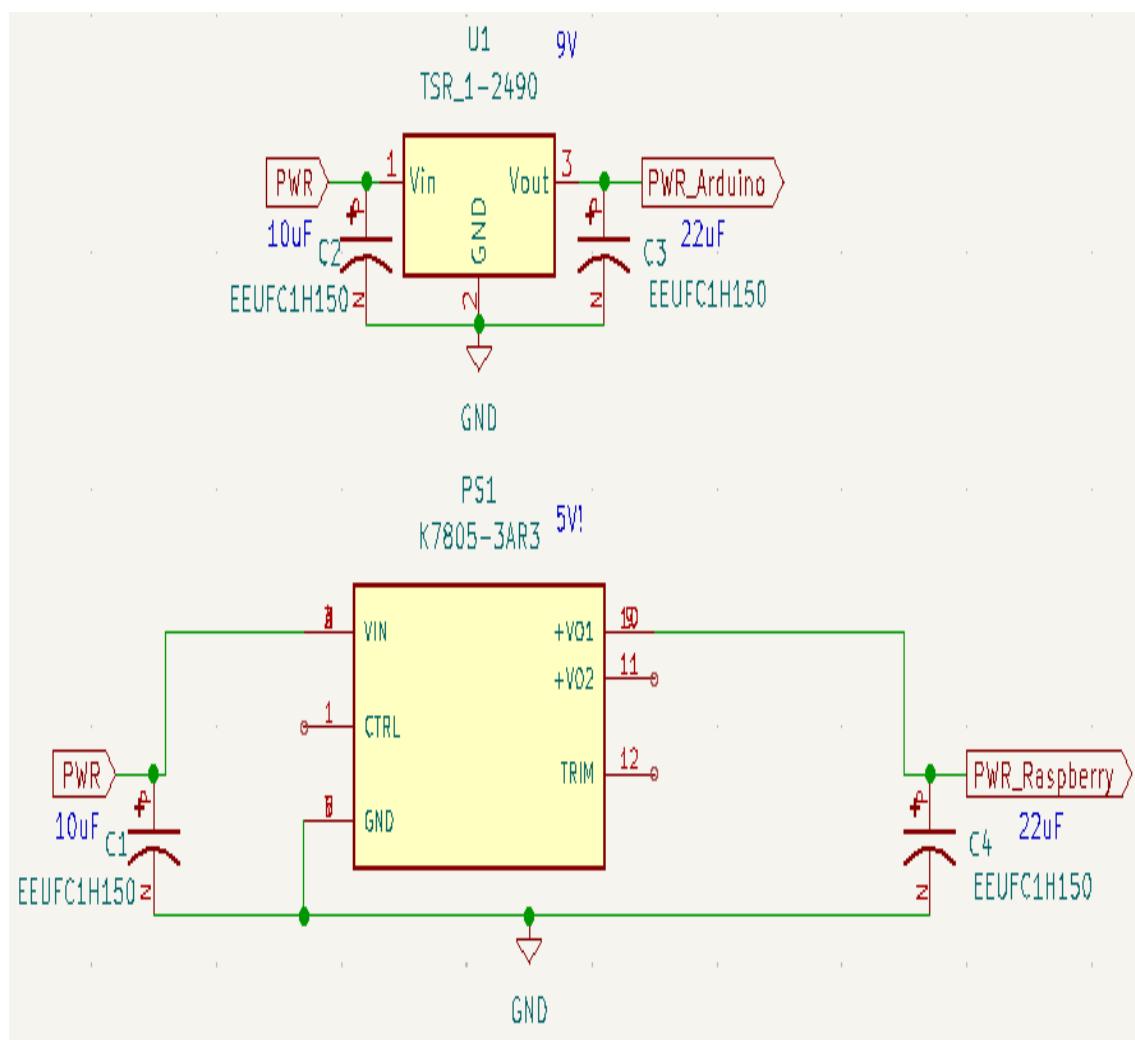


Figura 4.3.6 Conversores.kicad_sch

Posteriormente, se conectarán los distintos elementos de la forma indicada en los esquemas anteriores. El diseño se hace a dos capas y el relleno de estas es masa. El resultado de las conexiones es lo que se muestra a continuación en la figura 4.3.7.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

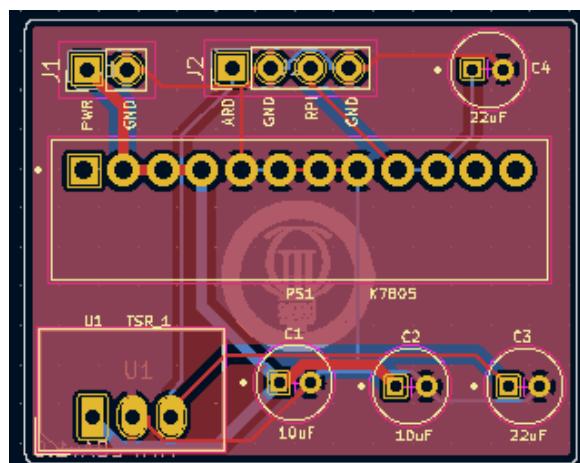


Figura 4.3.7 Diseño PCB Alimentación

También se muestra el diseño 3D de la PCB por las dos caras que genera KiCad. No todos los elementos tenían diseño 3D. Como curiosidad, en la parte inferior de la cara trasera se muestra siempre un código, en este caso es MMPCBAv1.0, la primera M es de máster, la segunda la inicial del nombre del autor, PCB indica que es la placa de circuito impreso, A en este caso por ser la placa de alimentación y v1.0 porque es la primera versión diseñada.

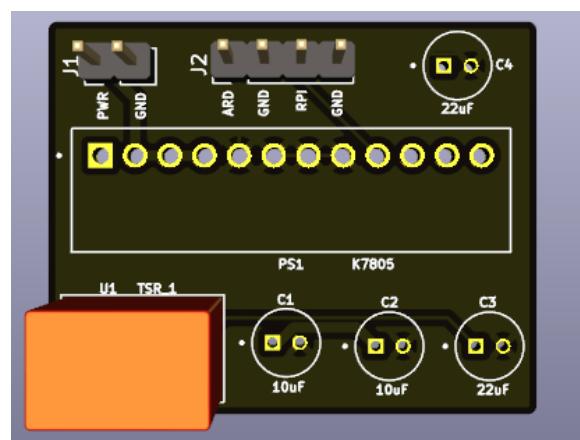


Figura 4.3.8 Cara frontal diseño 3D PCB Alimentación

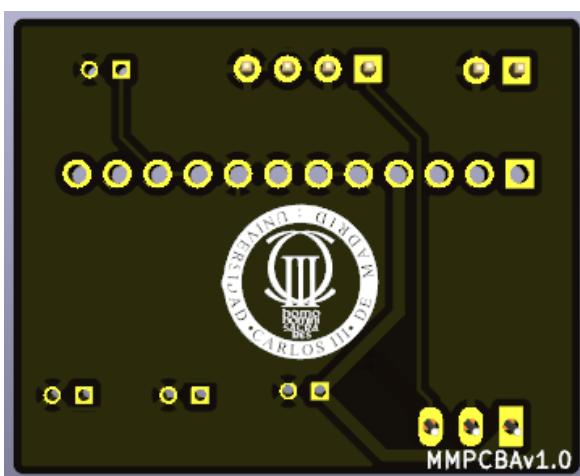


Figura 4.3.9 Cara posterior diseño 3D PCB Alimentación

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Se mandaron a imprimir estas PCB, acto seguido se soldaron a esta el resto de los componentes adquiridos. A continuación, se muestran ambas caras de las PCB sin y con los componentes soldados, sus dimensiones son 36 mm x 29 mm x 22 mm.



Figura 4.3.10 Cara frontal PCB Alimentación sin y con componentes



Figura 4.3.11 Cara trasera PCB Alimentación sin y con componentes

Finalmente, se realizaron las distintas pruebas para comprobar que el diseño implementado era efectivo. Para ello se suministró, por los pines de batería, una tensión de 25 V y se comprobó que las tensiones producidas por los pines de alimentación del Arduino y de la Raspberry fuesen correctos.

En el caso del Arduino se observó que el conversor genera una tensión promedio de 8,97 V, casi los 9 V deseados, comprobando que funciona correctamente. En el caso de la Raspberry Pi supera levemente los 5 V esperados, obteniendo una alimentación promedio de 5,11 V. Es importante recalcar que el rizado del conversor de 5 V es mayor en proporción al conversor de 9 V.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Placa de suelo

Esta placa se va a encargar tanto de alimentar a los distintos sensores como de realizar el acondicionamiento de los sensores analógicos de suelo explicados previamente. Se va a ir explicando el funcionamiento propio de la placa mientras se ven las distintas hojas jerárquicas del esquema implementado en KiCad.



Figura 4.3.12 Hoja jerárquica principal PCB Suelo

Se puede observar en el fichero “*alimentacion.kicad_sch*” los pines que van a estar conectados directamente a la batería para poder alimentar esta parte del sistema.

En el fichero “*pines_alimentacion_sensores.kicad_sch*” se generan los pines que salen de la placa para alimentar los diversos sensores que se utilizan.

El último fichero que trata de pines es el “*pines_IO_sensores.kicad_sch*” que contiene tanto el pin que va conectado a la salida del sensor (IN_MOISTURE) como el pin de la salida del acondicionamiento (OUT_MOISTURE) que irá conectado al Arduino. Además, hay un pin de control (CTR), este es de gran utilidad, porque se utilizará para apagar la placa y, por ende, desactivar el circuito, los sensores y desconectar la batería a través de una señal proveniente de la Raspberry Pi.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

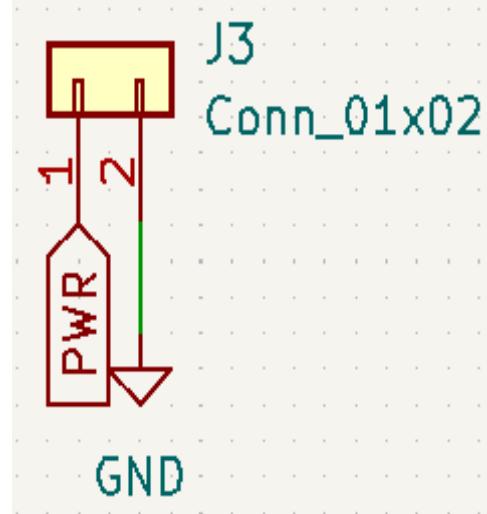


Figura 4.3.13 alimentacion.kicad_sch

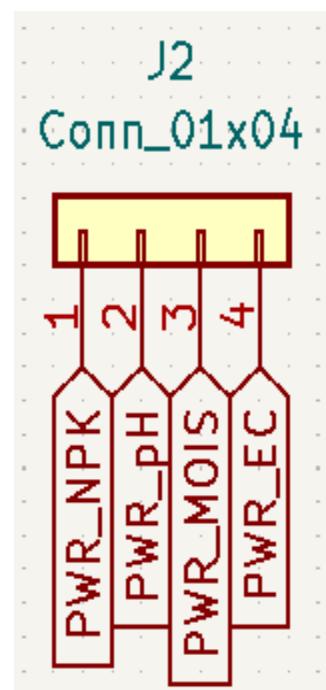


Figura 4.3.14 pines_alimentacion_sensores.kicad_sch

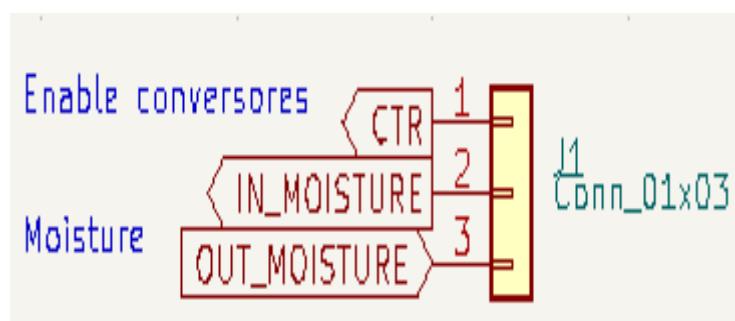


Figura 4.3.15 pines_IO_sensores.kicad_sch

En el fichero “conversores.kicad_sch” se tiene el circuito que se encarga de alimentar a los distintos sensores. Para saber los valores de alimentación de estos, es necesario recurrir a las hojas de datos de los sensores que irán conectados al suelo:

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Modelo del sensor	Magnitud/es	Rangos de alimentación	Alimentación utilizada
Soil NPK Sensor	NPK	5 VDC – 30 VDC	9 VDC
SEN0249	pH	5 VDC	5 VDC
314990620	Humedad del suelo	3,6 VDC – 30 VDC	9 VDC
Soil Moisture & Temperature & EC Sensor	EC	3,6 VDC – 30 VDC	9 VDC

Tabla 4.3.1 Alimentación utilizada para los distintos sensores de suelo

En el siguiente circuito se muestra el conversor R-739.0P que se encarga de proporcionar una tensión de 9 V que servirá para alimentar a los sensores de NPK, de humedad del suelo y de EC, además de alimentar los amplificadores de la fase de acondicionamiento. Para alimentar al sensor de pH se hace uso del regulador LM1085ISX-5.0/NOPB que fija la tensión a los 5 V pedidos por el sensor.

Como se ha explicado, esta parte de la PCB se encarga de la alimentación del circuito de acondicionamiento y de los sensores. Además, se ha utilizado un conversor con entrada de encendido y apagado, se aprovechará la utilidad de este pin para hacer más eficiente al sistema. El conversor estará en funcionamiento cuando la entrada ON/OFF esté en abierto o con una tensión comprendida entre 4,5 V y 28 V y estará apagado cuando tenga una tensión inferior a 0,8 V.

Como los GPIO de la Raspberry Pi alcanzan los 3,3 V, no llegan a los 4,5 V necesarios para activar el conversor. Por tanto, se realiza un driver con un transistor para el encendido y el apagado del sistema. Para hacer más eficiente el sistema, se utilizarán solamente transistores en modo de saturación, es decir, que se activen al recibir corriente para que solamente en los momentos de medida se encienda el sistema y el resto del tiempo permanezca apagado.

En las siguientes imágenes se muestran dos posibles configuraciones en función del tipo de transistor utilizado. A pesar de realizar estos dos diseños, solamente se llevará a cabo el diseño correspondiente al PMOS, esto es porque el PMOS tiene corrientes de fuga en reposo mucho menores que el NMOS y al ser el consumo un parámetro fundamental a tener en cuenta se decide elegir la opción del PMOS.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

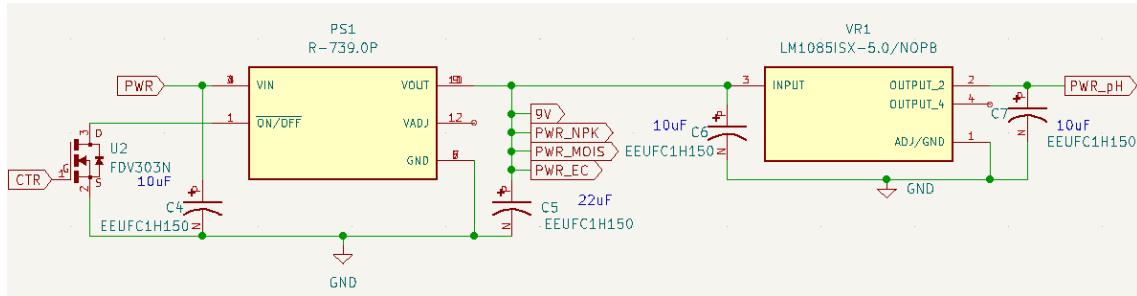


Figura 4.3.16 conversores.kicad_sch con driver NMOS

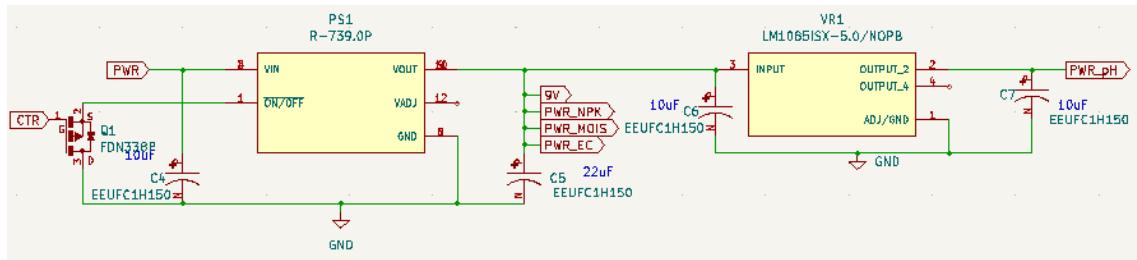


Figura 4.3.17 conversores.kicad_sch con driver PMOS

El siguiente circuito corresponde con “*acondicionamiento_moisture.kicad_sch*”, en el cual se puede observar el acondicionamiento del sensor de humedad de suelo. Tiene tanto la parte de amplificación, como la de filtrado, además de añadirle dos condensadores a la entrada de la alimentación del amplificador para intentar estabilizar la señal y mitigar el ruido.

Para cumplir el requisito de utilizar siempre alimentaciones positivas, se hace uso del amplificador operacional con alimentación asimétrica OP292GSZ – REEL.

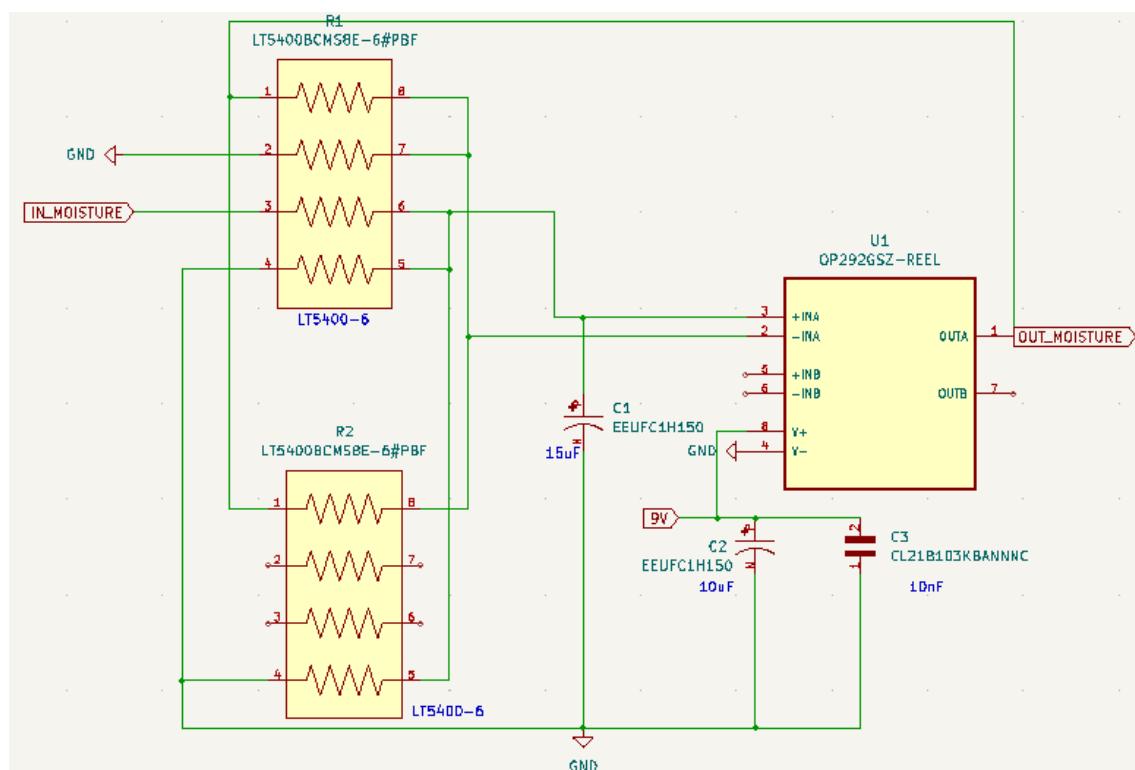


Figura 4.3.18 acondicionamiento_moisture.kicad_sch

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Al igual que en el caso anterior, se conectarán los distintos elementos de la forma indicada en los esquemas anteriores. El diseño se hace a dos capas y el relleno de estas es masa.

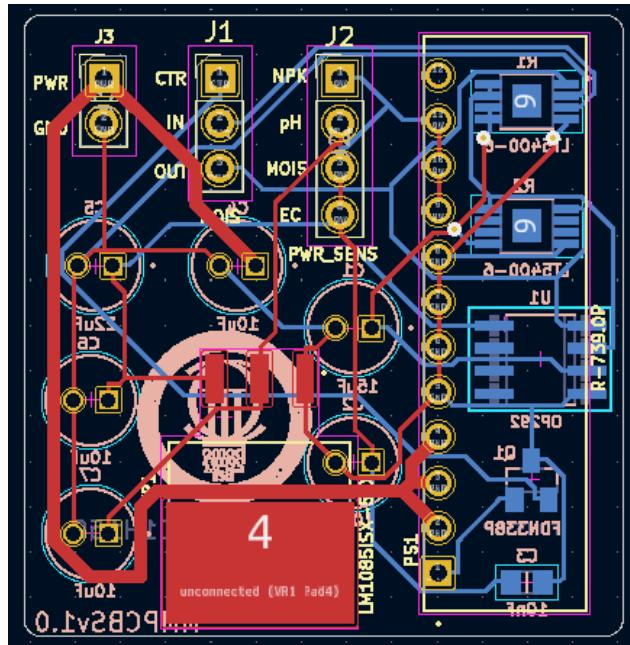


Figura 4.3.19 Diseño PCB Suelo

Se procede a ver el diseño 3D generado por KiCad de la placa de suelo. Al igual que pasaba en la anterior placa, en la parte inferior de la cara trasera se muestra el código MMPCBSv1.0, la sigla S indica que es la placa de suelo, el resto de las siglas ya se han explicado.

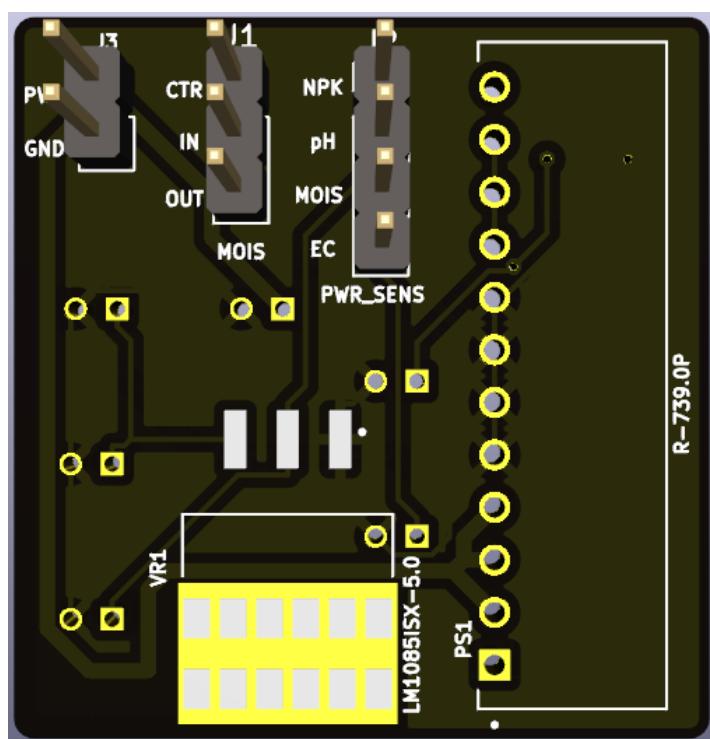


Figura 4.3.20 Cara frontal diseño 3D PCB Suelo

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

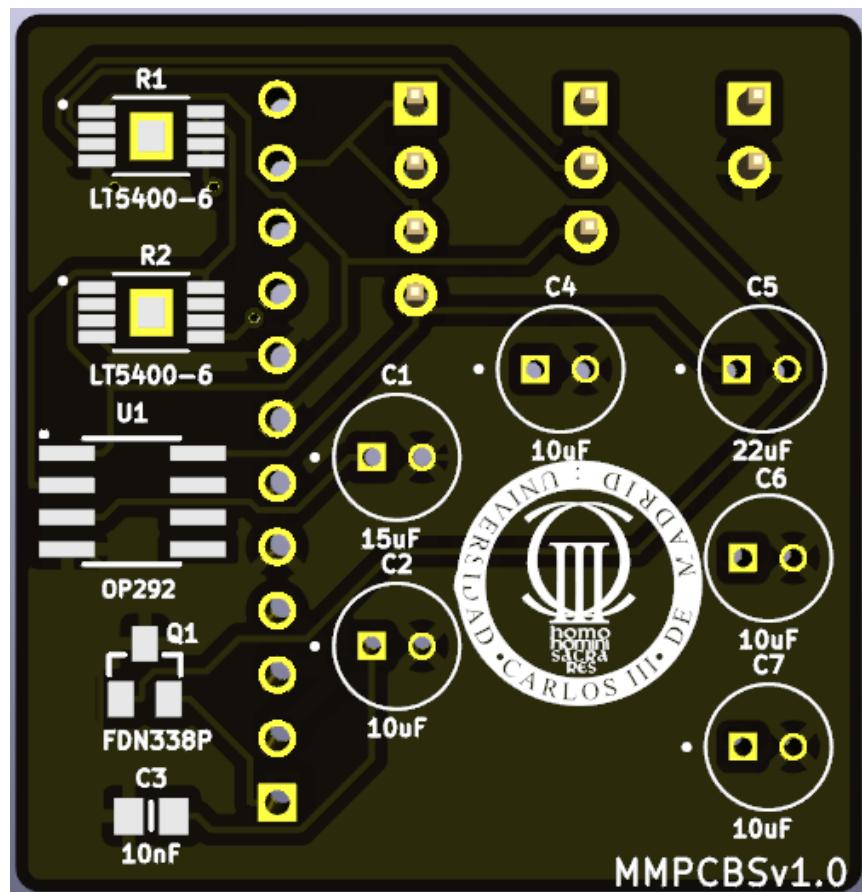


Figura 4.3.21 Cara posterior diseño 3D PCB Suelo

Se imprimieron las PCB y se soldaron los componentes, como se muestra en las siguientes imágenes. Sus dimensiones son 35 mm x 33 mm x 30 mm.

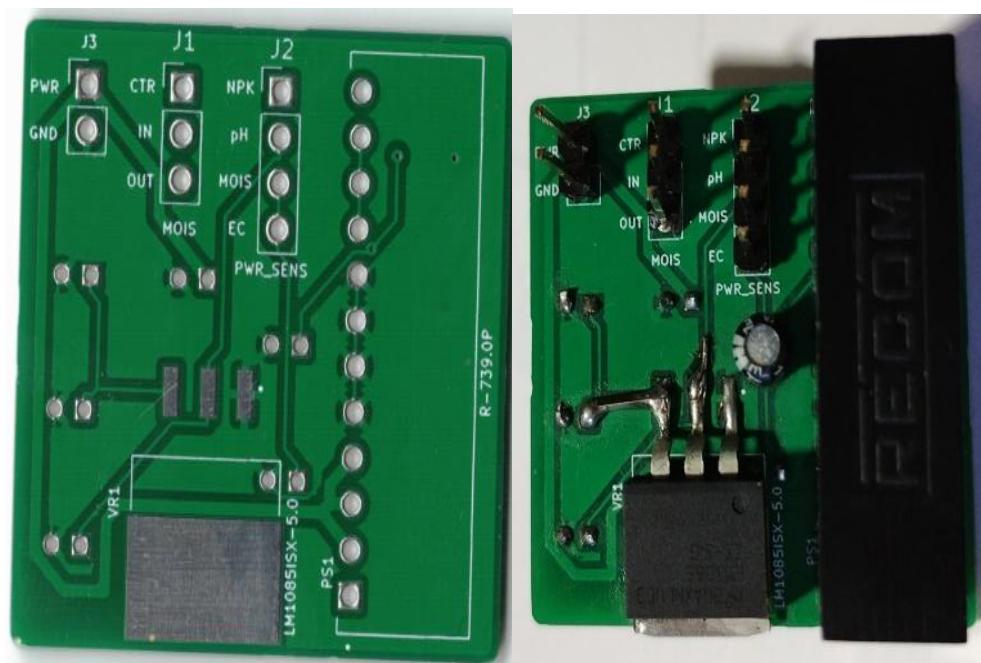


Figura 4.3.22 Cara frontal PCB Suelo sin y con componentes

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

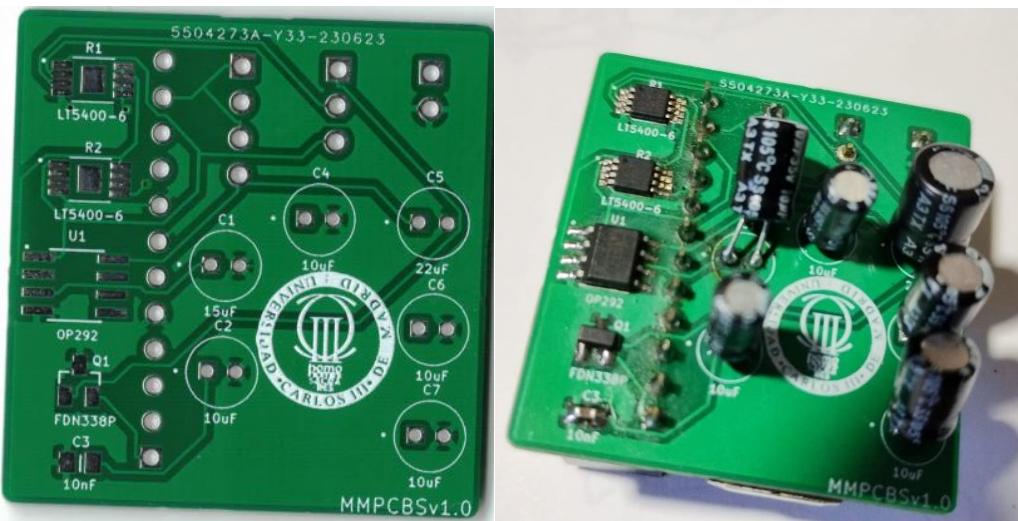


Figura 4.3.23 Cara posterior PCB Suelo sin y con componentes

Para comprobar que la fabricación era correcta y que estaba adecuadamente soldada, se procedió a realizar pruebas en el laboratorio. Primero, se comprobaron que las alimentaciones de los distintos sensores son las esperadas. También, se evidenció el funcionamiento de la parte de acondicionamiento de la PCB del sensor de humedad del suelo, observando la ganancia de 2,5 esperada.

Para finalizar el estudio de esta PCB se muestra como varía la alimentación en función del pin de control, en este caso se observa que se activa el sistema de manera quasi inmediata. En cambio, cuando el pin de control está próximo a los 0 V, tiene una caída más leve debido al tiempo de descarga de los condensadores.



Figura 4.3.24 Comprobación de la alimentación al variar el pin de control

Placa de hoja

La última PCB que se va a estudiar se encarga de efectuar el acondicionamiento y alimentar a los sensores de hoja. Se irá detallando el funcionamiento intrínseco de la placa, al mismo tiempo que se examinan las diversas capas jerárquicas del esquema generado en KiCad.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR



Figura 4.3.25 Hoja jerárquica principal PCB Hoja

De igual manera que en la PCB de suelo, se puede observar en el fichero “*alimentacion.kicad_sch*” los pines que se encargan de alimentar el sistema, debido a que éstos están conectados directamente a la batería.

En el fichero “*pines_alimentacion_sensores.kicad_sch*” se muestran los pines a la salida de los conversores y reguladores, estos tienen como tarea alimentar a los distintos sensores. En este caso, se ha introducido el pin de control (CTR) en esta hoja jerárquica por simplicidad, se recuerda que este es el pin que se utiliza para apagar el circuito y que de esta forma no se consuma potencia.

También está el fichero “*pines_IO_sensores.kicad_sch*” que usa los pines con inicial “IN” para recibir los datos de los sensores y procesarlos a través de los circuitos de acondicionamiento implementados. De igual modo, utiliza los pines “OUT” para unir las salidas de los circuitos de acondicionamiento de los distintos sensores al Arduino Uno.

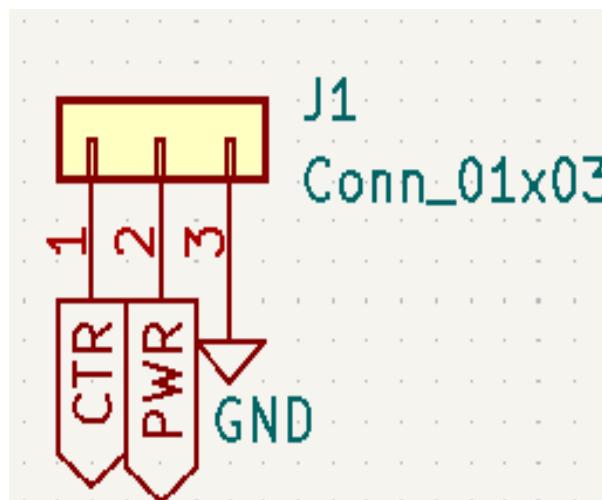


Figura 4.3.26 alimentacion.kicad_sch

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

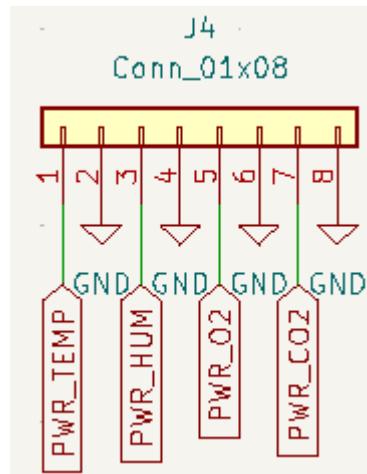


Figura 4.3.27 pines_alimentacion_sensores.kicad_sch



Figura 4.3.28 pines_IO_sensores.kicad_sch

En las figuras 4.3.29 y 4.3.30 se muestra el circuito del fichero “*conversores.kicad_sch*”. En este circuito se observa cómo se transforma la tensión de la batería en las tensiones necesarias para alimentar los distintos sensores y circuitos de acondicionamiento.

Antes de estudiar el circuito, hay que ver cuáles son las alimentaciones que admite cada uno de los sensores que se quieren incorporar al follaje del árbol.

Modelo del sensor	Magnitud/es	Rangos de alimentación	Alimentación utilizada
LAT-B3	Temperatura estomática	2,5 VDC	2,5 VDC
PYTHOS-31	Humectación hoja	2,5 VDC – 5 VDC	5 VDC
SEN0322	O ₂	3,3 VDC – 5 VDC	5 VDC
MH-Z19C	CO ₂	3,6 VDC – 5,5 VDC	5 VDC

Tabla 4.3.2 Alimentación utilizada para los distintos sensores de hoja

En la parte superior del circuito se utiliza un conversor de 9 V (R-739.0P) para alimentar los amplificadores que acondicionan las señales de los sensores. También, se muestra un regulador de 2,5 V (LM317LID) junto con un seguidor de tensión (amplificador

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

operacional LM358DR2G realimentado negativamente) para ajustar la alimentación del sensor de temperatura estomática.

En la parte inferior también se utiliza el mismo tipo de conversor de 9 V para, después, utilizar un regulador de 5 V (LM1085ISX-5.0/N0PB) para alimentar al resto de sensores.

Se muestra tanto el diseño con drivers NMOS como PMOS.

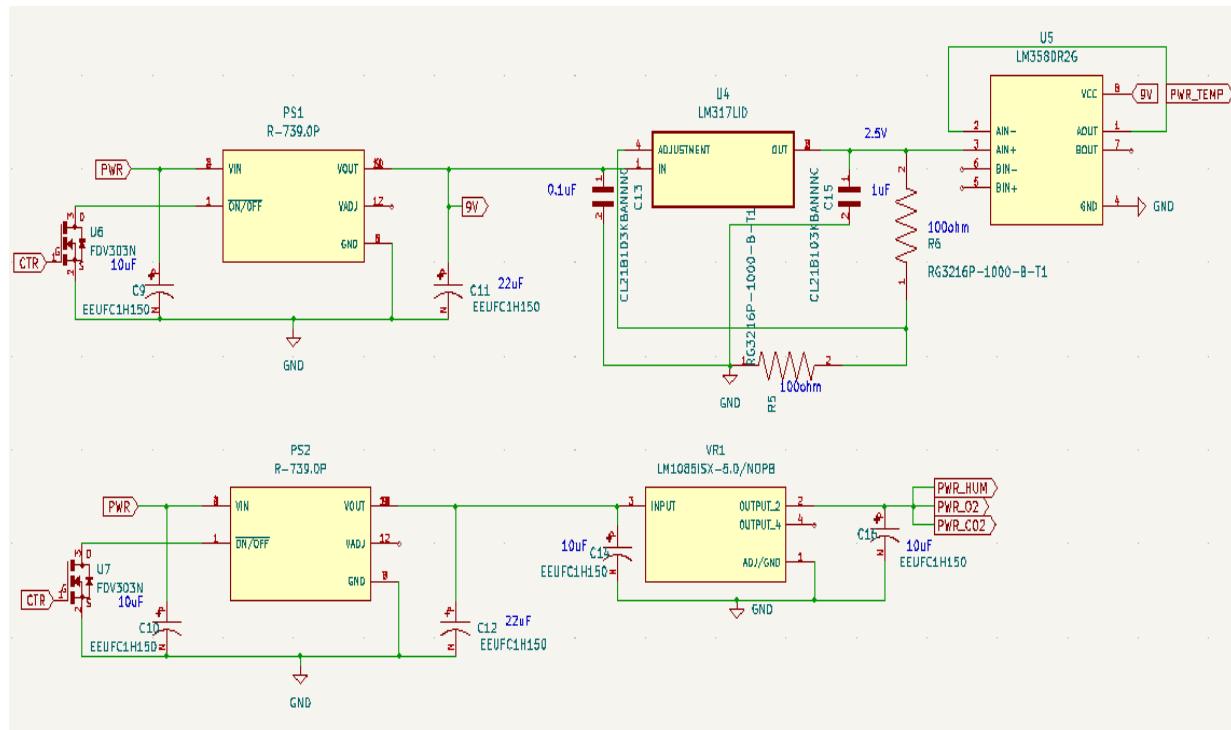


Figura 4.3.29 conversores.kicad_sch con driver NMOS

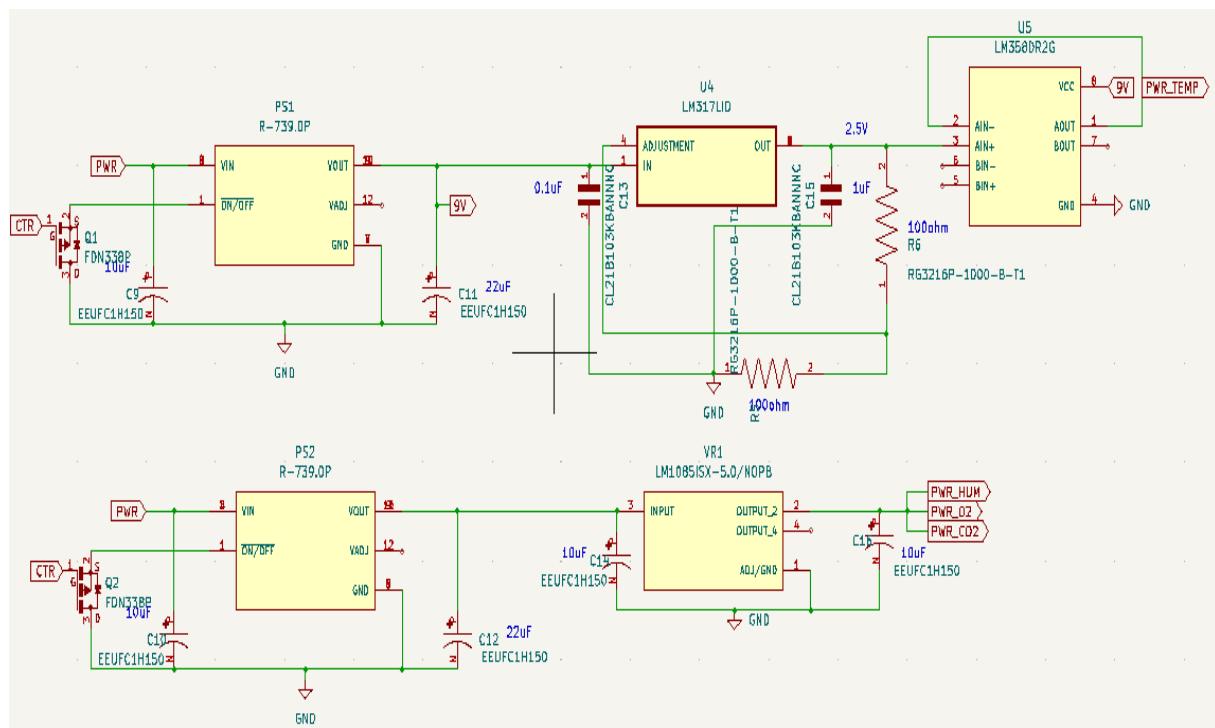


Figura 4.3.30 conversores.kicad_sch con driver PMOS

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Los siguientes circuitos muestran el acondicionamiento de los sensores de temperatura estomática y humedad de hoja. Se han dividido en distintas hojas jerárquicas para facilitar el entendimiento de los esquemáticos.

Para el acondicionamiento del sensor de temperatura estomática, se observó que se necesitaba un offset de 0,25 V. El fichero “referencia_025V.kicad_sch” es el que, a partir de la tensión de 9 V que alimenta a estos circuitos, genera una salida de 1,25V con la referencia de tensión “LM317LID” para, posteriormente, con un divisor de tensión y con un driver seguidor de tensión, generar la referencia de 0,25 V necesaria para paliar el offset de este sensor.

En los ficheros “acondicionamiento_temperature.kicad_sch” y “acondicionamiento_humidity.kicad_sch” se muestran cómo sería la configuración de las resistencias que realimentan al amplificador para obtener las ganancias especificadas en los sensores de temperatura estomática y humedad de hoja respectivamente.

El fichero “amplificacion.kicad_sch” muestra como el amplificador se uniría con las distintas resistencias para el acondicionamiento de los sensores. Se vuelve a recordar que el elemento “OP292GSZ – REEL” es un amplificador con alimentación asimétrica para cumplir el objetivo de solamente hacer uso de tensiones positivas en el sistema.

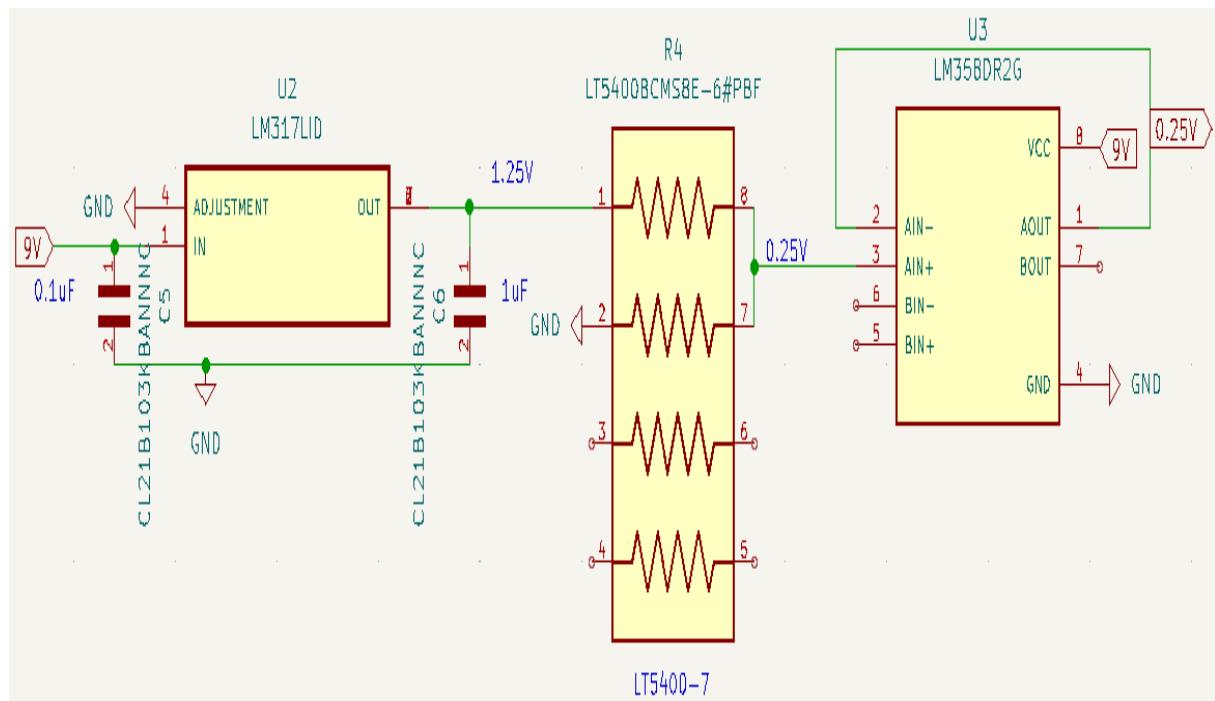


Figura 4.3.31 referencia_025V.kicad_sch

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

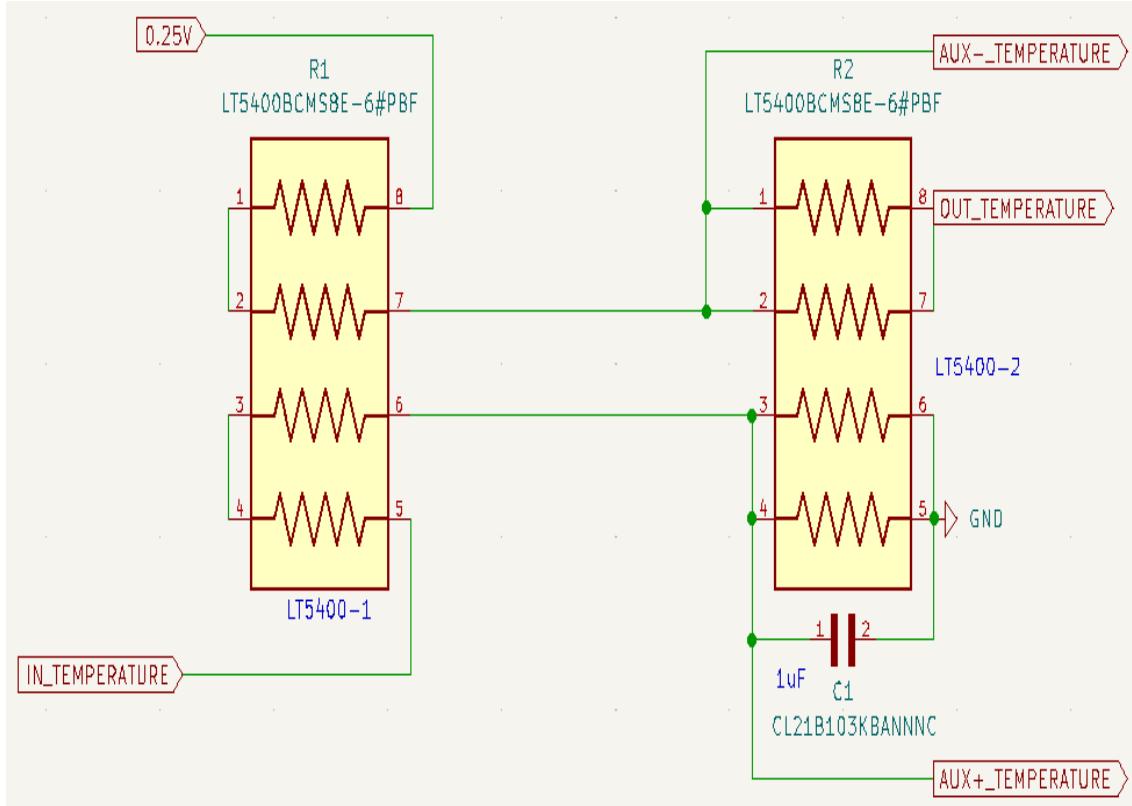


Figura 4.3.32 acondicionamiento_temperature.kicad_sch

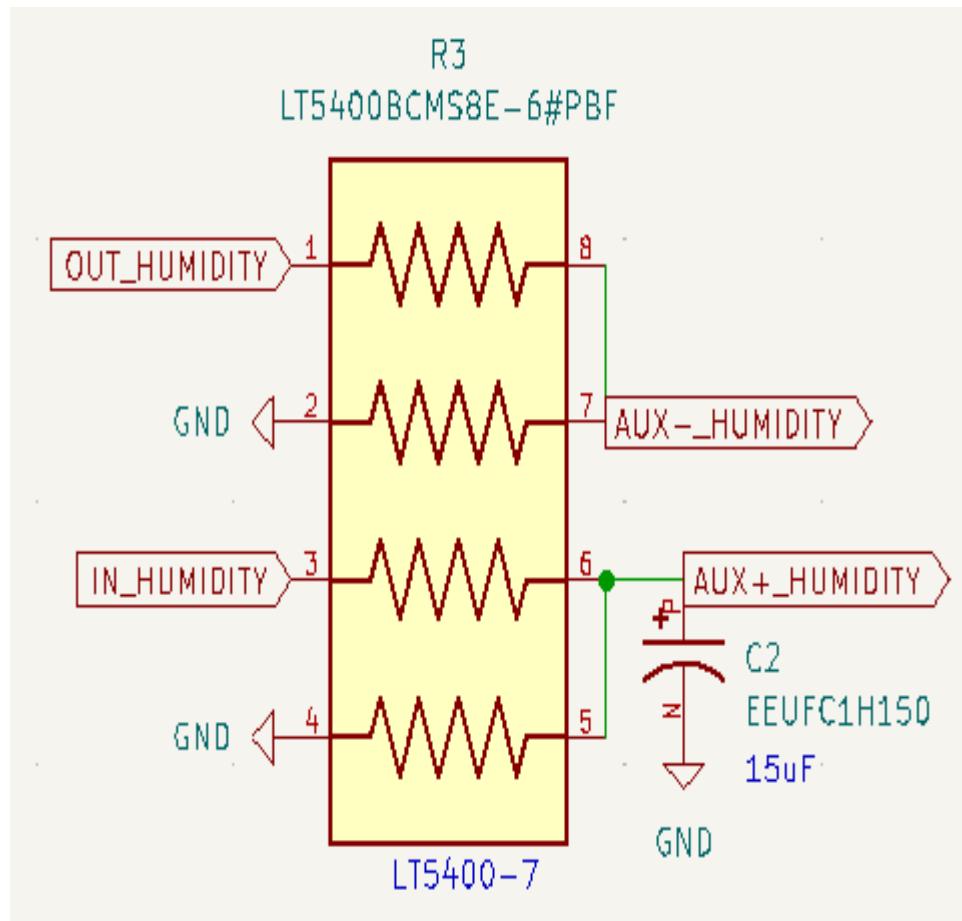


Figura 4.3.33 acondicionamiento_humidity.kicad_sch

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

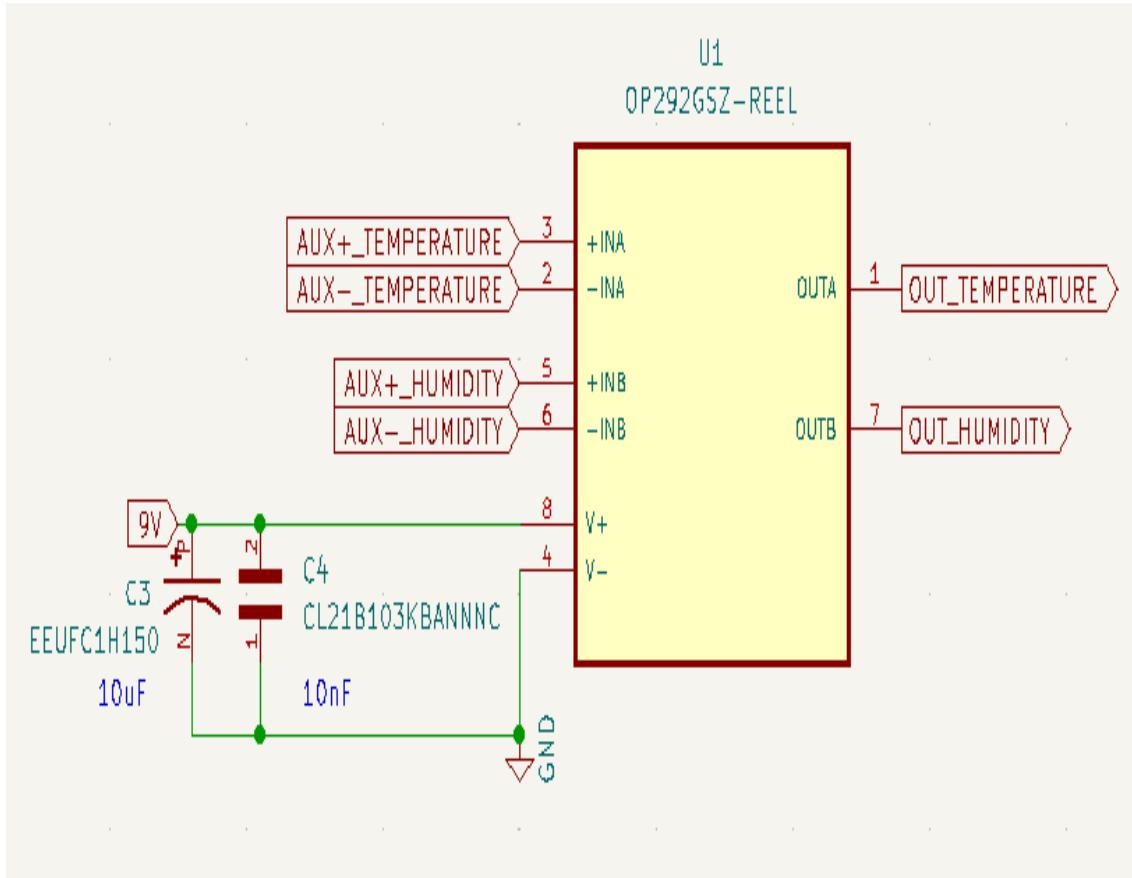


Figura 4.3.34 amplificacion.kicad_sch

Con la herramienta KiCad se cargan los distintos componentes para unirlos, para ello se hace uso de dos capas con relleno masa.

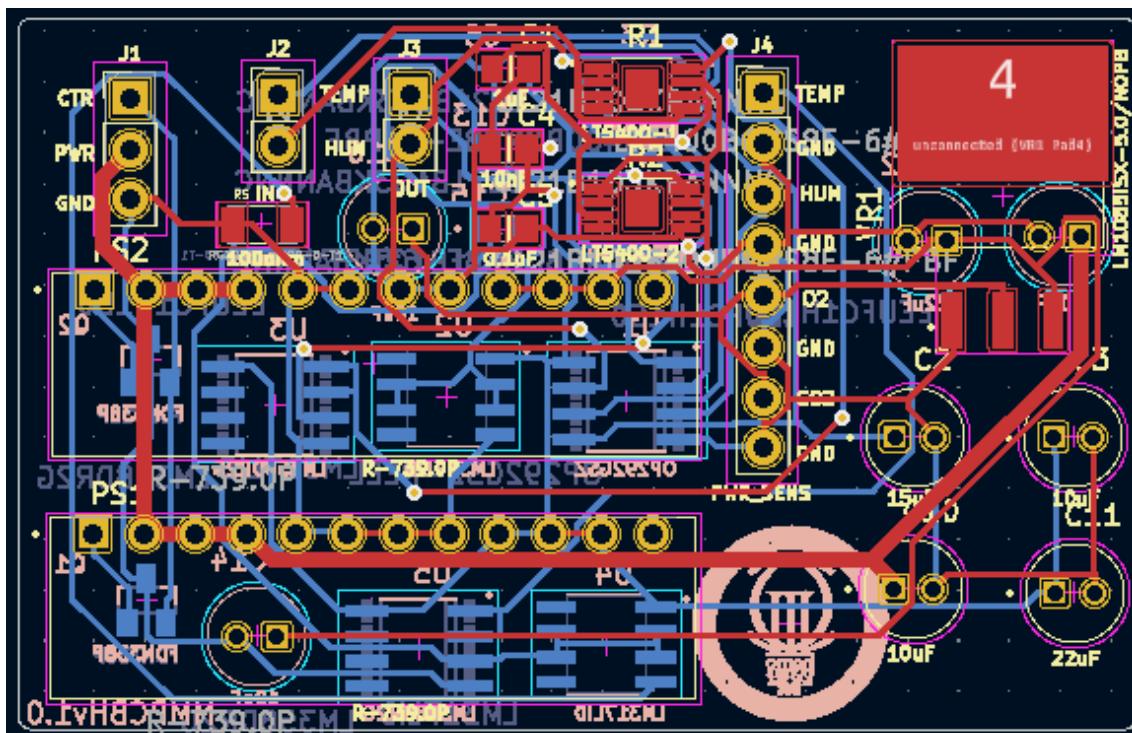


Figura 4.3.35 Diseño PCB Hoja

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Se muestra el diseño 3D del sistema implementado. En la parte inferior de la cara trasera de la placa se muestra el código MMPCBhv1.0, la sigla H indica que es la placa de hoja.

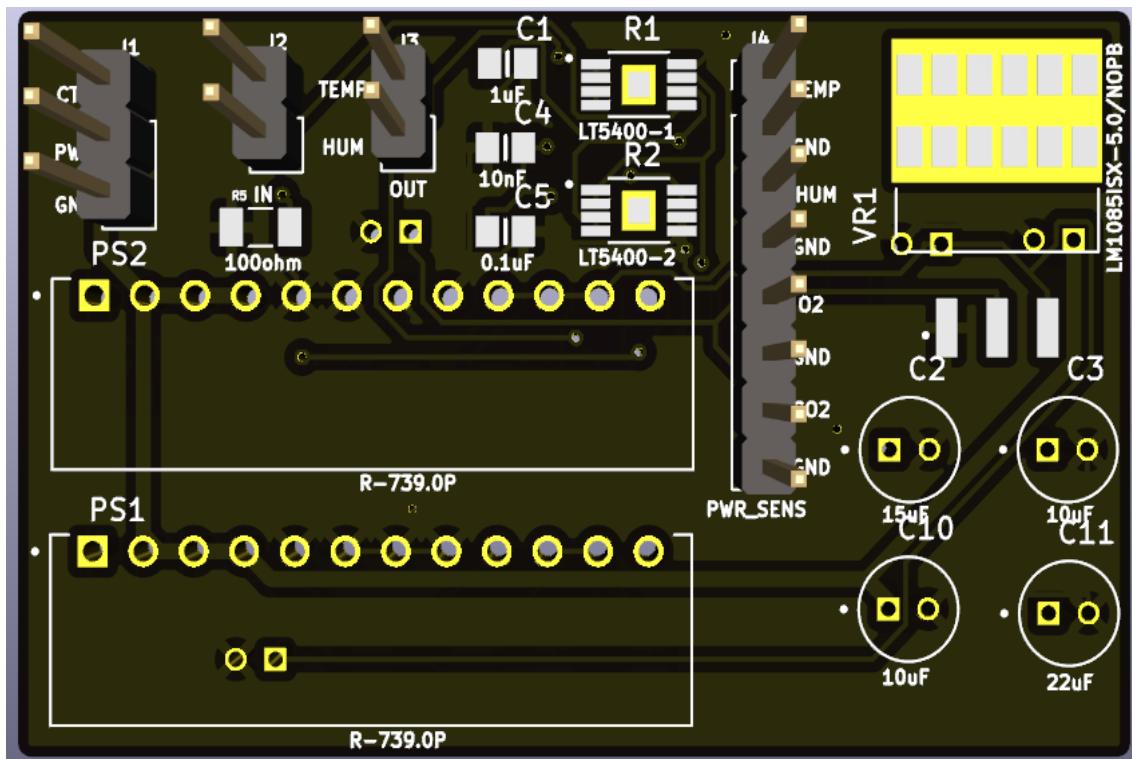


Figura 4.3.36 Cara frontal diseño 3D PCB Hoja

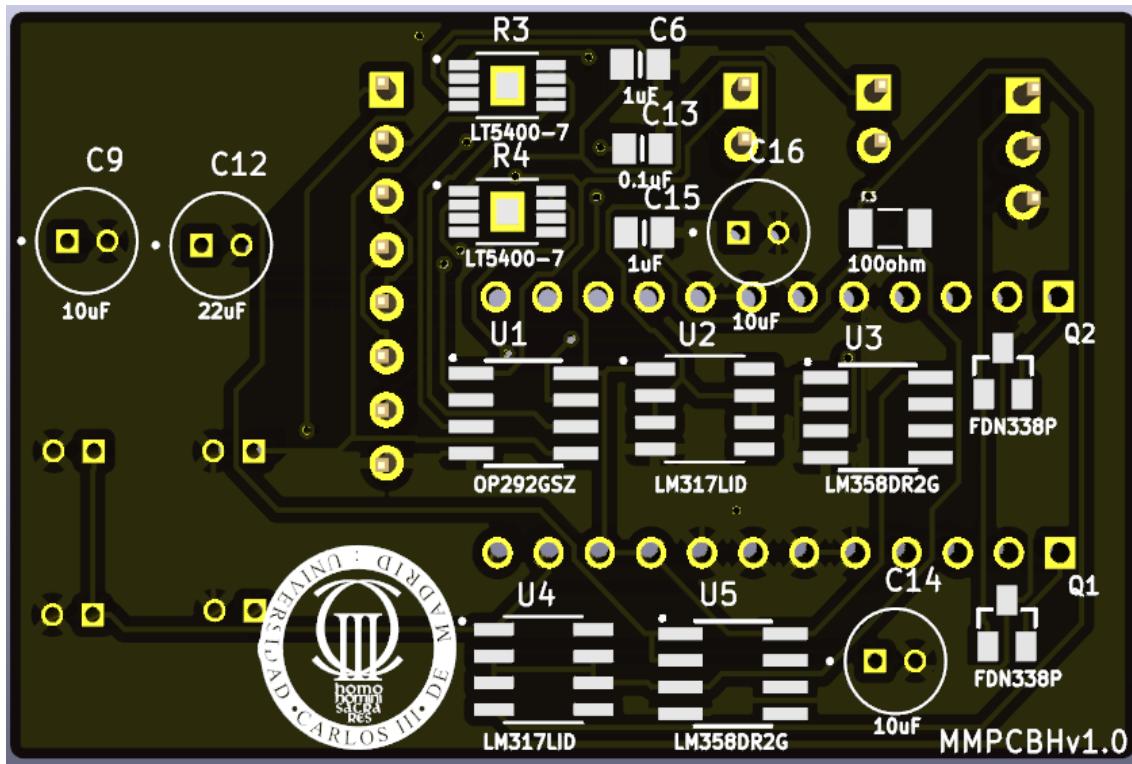


Figura 4.3.37 Cara posterior diseño 3D PCB Hoja

Siguiendo el procedimiento, en las siguientes figuras se muestran las PCB sin y con los componentes soldados, sus dimensiones son 56 mm x 36 mm x 35 mm.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

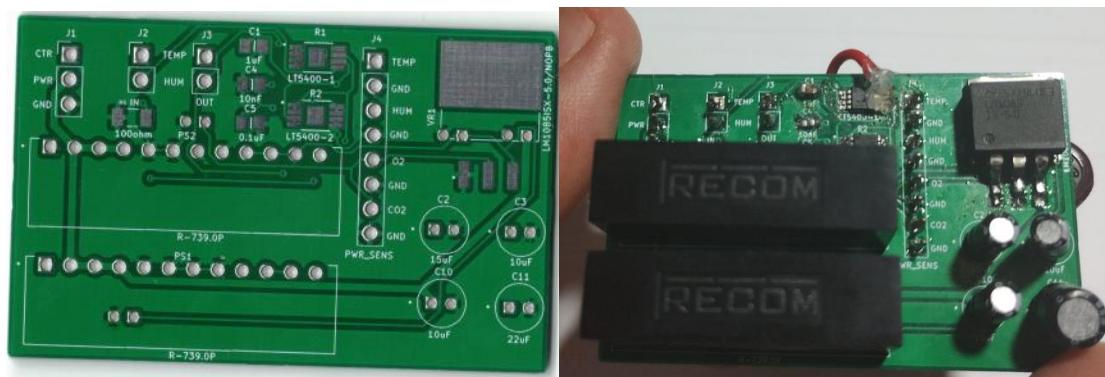


Figura 4.3.38 Cara frontal PCB Hoja sin y con componentes

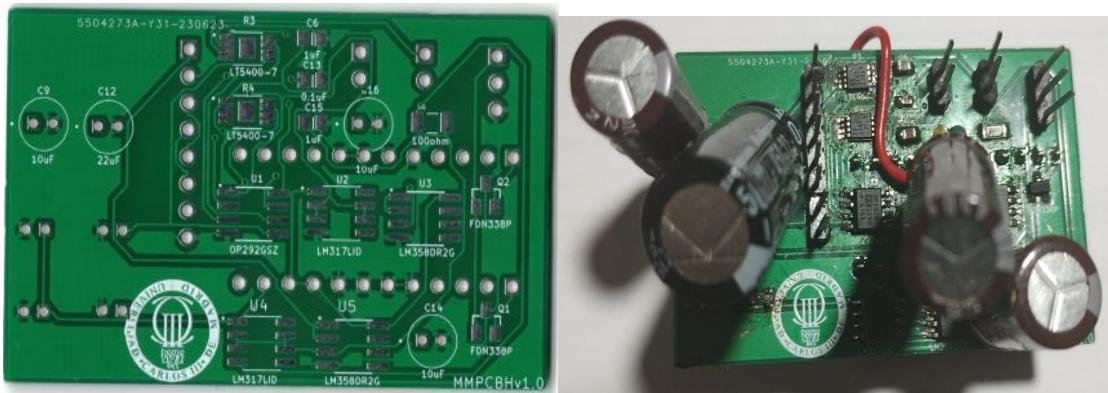


Figura 4.3.39 Cara posterior PCB Hoja sin y con componentes

Con el objetivo de verificar la precisión del proceso de fabricación y asegurarse de que la placa había sido soldada correctamente, se llevaron a cabo una serie de pruebas en el entorno del laboratorio. El primer paso de estas pruebas consistió en la verificación de las tensiones de alimentación suministradas a los diversos sensores observando que, para los sensores de humedad de hoja, O_2 y CO_2 se tienen la alimentación esperada, en cambio, para la temperatura estomática se tiene una tensión de 2,8 V que diverge ligeramente de los 2,5 V esperados.

Se procede a evaluar el funcionamiento de la sección de acondicionamiento de la PCB de hoja. Durante esta evaluación, se observó que el sensor de temperatura estomática alcanzaba la ganancia prevista de 2. Asimismo, en relación con la detección de la humedad de la hoja, se verificó que sí se logra la ganancia deseada de 4.

Con esto se finaliza la parte del diseño y de las pruebas de las PCB utilizadas en este trabajo. Se han comprobado que todos los sensores son correctamente alimentados por estas PCB y que también funciona el acondicionamiento de los sensores analógicos.

Antes de mostrar cómo se realizan las conexiones entre los distintos elementos que conforman este nodo, es menester explicar que los sensores que hacen uso del protocolo de comunicación RS485 precisan de un módulo de conversión para poderlos integrar correctamente en el Arduino Uno al utilizar comunicación TTL. Por esto último, se decide utilizar el módulo de conversión “Módulo MAX485 RS-485 TTL”.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR



Figura 4.3.40 Módulo MAX485 RS-485 TTL (68)

A lo largo del desarrollo de este proyecto, se han enfrentado varios desafíos que han llevado a la modificación del enfoque original planteado. Este planteamiento inicial queda documentado en esta memoria como el punto de partida, se han identificado diversas flaquezas en este enfoque. Sin embargo, debido a limitaciones de tiempo, no se ha tenido la oportunidad de realizar correcciones exhaustivas. Por ende, se ha procedido a realizar ajustes en lo que ya se tenía disponible para poder continuar con el trabajo en curso.

Uno de los mayores inconvenientes ha sido el retraso en el envío de los componentes del sistema. Hasta finales de julio, no se consiguieron dichos componentes, esto provocó la necesidad de trabajar a contra reloj. Es importante resaltar que este retraso no se debe a falta de previsión al tener todo planificado con muchos meses de antelación. Sin embargo, al ser tan específicos estos componentes, conllevó a un suministro tardío, lo cual ha afectado en la ejecución del sistema y en la verificación del funcionamiento de cada parte.

Otro objetivo establecido en el plan era utilizar solamente sensores analógicos, pero para ciertas magnitudes no se consiguieron sensores que proporcionaran una señal analógica de salida, ya sea por su precio o porque no hay en el mercado. Algunos sensores sí que tenían la opción de generar una señal analógica, como el 314990620, pero la falta de existencias de estos sensores requirió su reemplazo por sensores con protocolo de comunicación RS485.

Lamentablemente, el sensor de temperatura estomática sufrió un percance que impidió su inclusión en la versión final del proyecto. Otro problema con los sensores fue que aquellos que utilizaban como protocolo RS485 presentaban interferencias con la comunicación I2C que tiene el Arduino y la Raspberry. Por tanto, no fue posible emplear estos sensores en el marco del tiempo disponible para el proyecto.

Ya se ha terminado de estudiar e implementar la parte HW del primer tipo de nodo. Con esta fase sólidamente establecida, el siguiente paso consiste en exponer de manera detallada los esquemas de conexiones que interrelacionan todos y cada uno de los componentes que, finalmente integran este nodo en cuestión.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

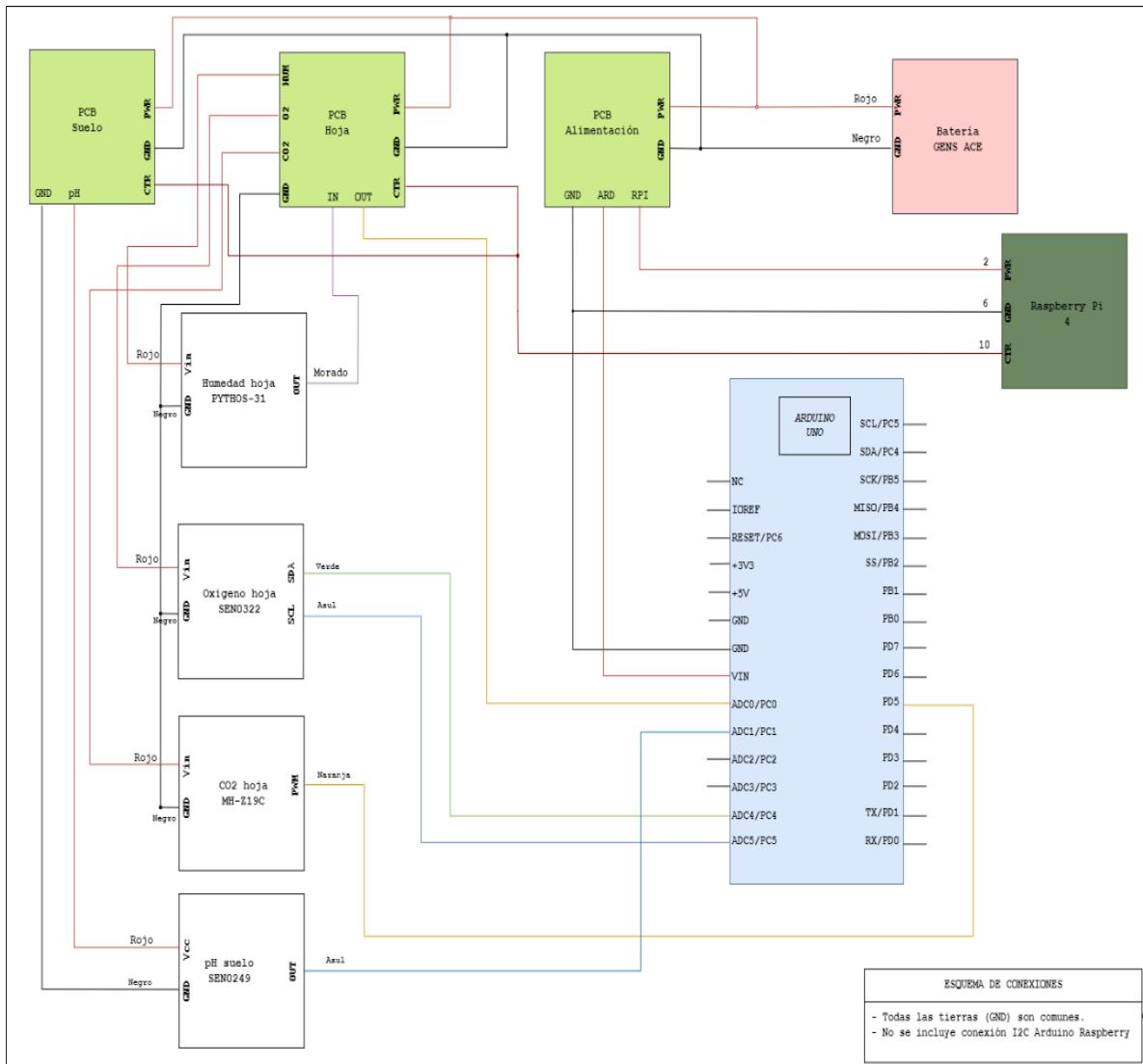


Figura 4.3.41 Esquema de conexiones del primer nodo implementado

4.4. Adquisición de los datos sensor-Arduino y procesado

Gracias a los circuitos de acondicionamiento y los sensores, ya se está en disposición de adquirir los datos de las distintas magnitudes que se desean medir. Va a ser necesario realizar un preprocesamiento básico para, posteriormente, transmitir los datos a la Raspberry Pi, pero se puede complicar en función de los requisitos.

El presente diseño consta de dos modos de operación que puede elegir el usuario en función de sus necesidades. Estos modos se pueden utilizar, también, en función del consumo energético. Los dos modos que se van a utilizar son:

- **Toma continua de datos:** Transmitirá de forma continua los datos, tal y como se obtienen de los sensores o circuitos de acondicionamiento, se considera fundamental no eliminar ningún dato, ya que del procesamiento de estos se encargará otra parte del sistema.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Este modelo puede conllevar un mayor gasto energético debido a la toma y transmisión continua de datos, pero es importante recalcar que solamente toma medidas el dispositivo cuando le llega una petición por parte de la Raspberry Pi. Por tanto, el Arduino podría utilizar este modo de toma continua de datos, pero la Raspberry se encargaría de que no consuma demasiada potencia, haciendo que tome estos datos cada cierto tiempo.

- **Toma periódica de datos:** Establece periodos de tiempo para la toma de datos, además, hace uso del Edge Computing para no saturar el sistema.

El Edge Computing es un término que se refiere al procesamiento de datos realizado cerca de la zona de medida por un sistema para que no se procesen los datos de todos los sensores en el mismo punto, dado el posible amplio volumen de datos que puede tener el sistema final. El propósito es reducir la cantidad de datos transmitidos y prolongar la vida útil de la batería para optimizar el sistema. Con este código se realizará un preprocesamiento que reduzca el número de datos obtenidos, sustituyéndolos por valores que caracterizan toda la información. Para ello se hará uso de herramientas estadísticas que modifiquen, de manera conveniente, todos los datos obtenidos.

Para conseguir el valor significativo de las medidas, se hará uso de la media, el inconveniente es que puede ser muy sensible a los valores atípicos (*outliers*). Para lograr evitar esta distorsión de las medidas, se hará uso de dos técnicas que palían estos efectos.

La media *winsorizada* establece dos límites extremos a partir de una primera media con los datos medidos, posteriormente, da a los *outliers* el valor del extremo más cercano para, a continuación, realizar la media con los nuevos datos.

La media recortada, para evitar los valores extremos, también hace una primera media para establecer unos límites externos, los valores que estén fuera de estos rangos serán eliminados y se procederá a realizar nuevamente la media.

El Arduino, para enviar la información que ha procesado de los sensores, deberá hacer uso de algún protocolo de comunicación, son muchas las opciones posibles. Para realizar una comunicación eficiente con cables entre el Arduino y la Raspberry se podría hacer uso de diversos protocolos de comunicación como 1-Wire, RS485, SPI... Pero, para este trabajo se utilizará el protocolo I2C al tener como características:

- **Hace uso de dos cables:** Esto es una tremenda ventaja, ya que resulta muy simple su conexión y deja más pines libres en el Arduino para poder integrar una mayor cantidad de sensores. Los dos cables que utilizan son una línea de reloj (SCL) y otra de datos (SDA).
- **Gran uso en múltiples dispositivos:** El protocolo I2C es de uso frecuente en la industria para diversas aplicaciones. Al ser tan utilizado dispone de gran cantidad de soportes y librerías para diversos dispositivos, haciendo que este protocolo sea una opción óptima para integrarla en el sistema desarrollado.
- **Velocidad de transmisión limitada:** Este podría ser un gran inconveniente si se precisa de un sistema cuyos datos tengan baja latencia. Sin embargo, en este proyecto, esto no sería para nada un inconveniente, al poder transmitir los datos a una velocidad más que adecuada para las pretensiones del presente trabajo.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

- **Posee sistema de direccionamiento:** Este protocolo tiene siete bits de direccionamiento, por ende, se pueden conectar múltiples dispositivos de forma simultánea sin que esto plantea problemas en la comunicación. Además, posee un sistema de maestro esclavo que será de gran utilidad en este proyecto, la Raspberry se encargará de ser el maestro que vaya haciendo peticiones para que los posibles dispositivos conectados proporcionen sus datos. Es necesario recalcar que, aunque tenga 7 bits de direccionamiento, realmente tiene 120 direcciones útiles.

En la figura 4.4.1 se muestra como se ha de realizar la conexión entre el Arduino Uno y la Raspberry Pi 4.

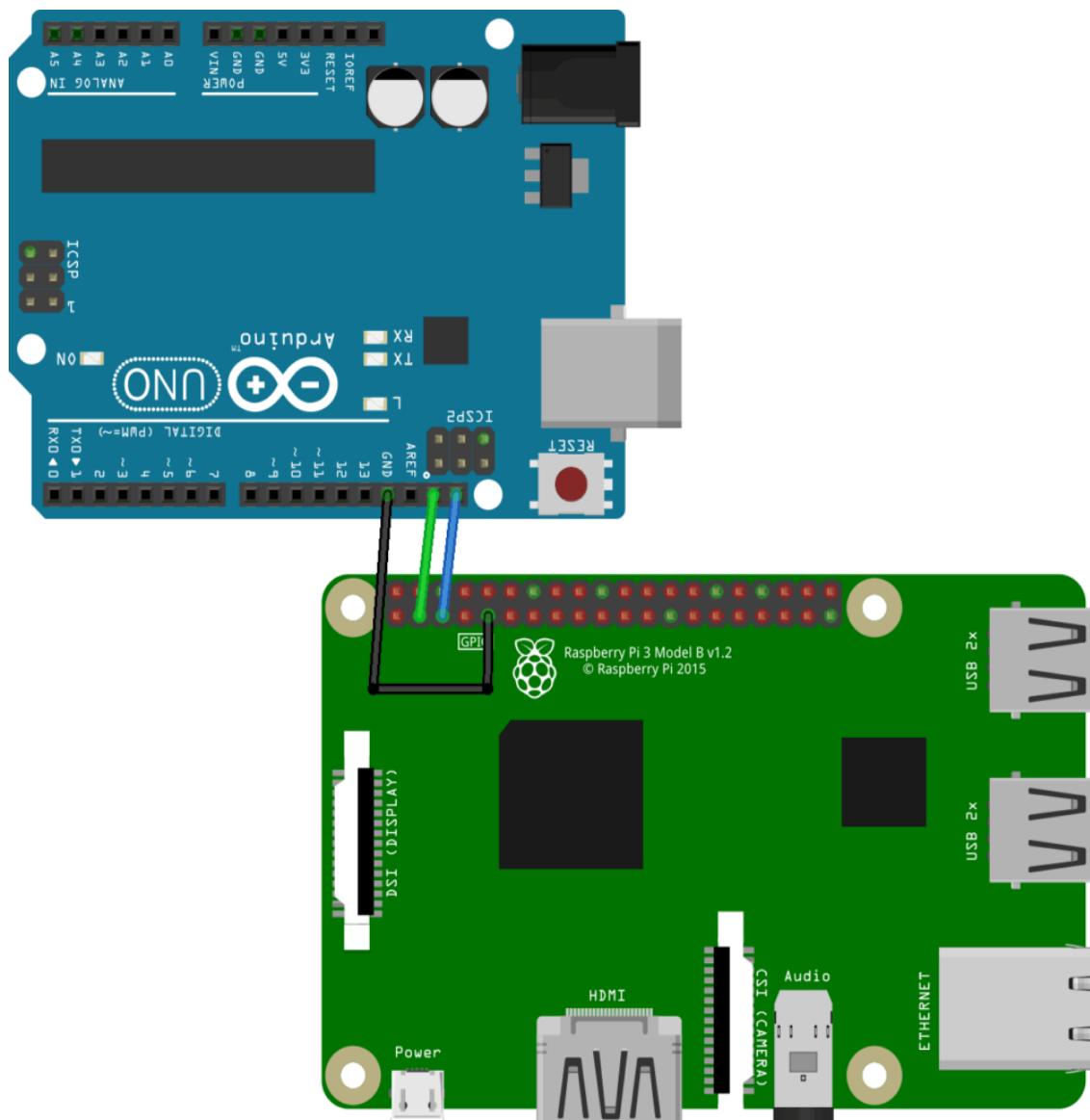


Figura 4.4.1 Esquema de conexión para comunicación I2C entre Arduino y Raspberry (69)

Se procede a explicar los códigos de los dos modos creados y se observará que funcionan de la manera esperada.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Código del primer modo (toma continua de datos)

Se inicializan las librerías para poder hacer uso del protocolo de comunicación I2C con la Raspberry. Además, se utilizará una librería con la que se puede obtener correctamente la información del sensor SEN0322.

```
#include <Wire.h>
#include "DFRobot_OxygenSensor.h"
```

Posteriormente, se tienen que inicializar todos los pines que se van a utilizar para comunicarse con los sensores, además de inicializar la dirección de este Arduino en el protocolo I2C. Asimismo, se crean distintas variables globales para la correcta obtención de los datos de los distintos sensores, es relevante indicar que el sensor SEN0322 utiliza también el protocolo I2C para comunicarse con el Arduino.

```
#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

int prevVal = LOW;
long th, tl, h, l, ppm;

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH

// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor
```

Con la función *setup* se configuran los parámetros previos necesarios para poder trabajar con el presente sistema. De igual forma, se añade un caso de error para indicar los momentos donde se pierde la conexión con el sensor SEN0322. En este caso, la función *loop* no será de mucha utilidad.

```
void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
// Inicialización del sensor de oxígeno
while (!oxygen.begin(Oxygen_IICAddress)) {
    Serial.println("Error al conectar con el sensor de oxígeno.");
    delay(1000);
}
Serial.println("Conexión exitosa con el sensor de oxígeno.");
}

void loop() {

}
```

Toda la toma de medidas y de datos se realiza dentro de la función *requestEvent*, para optimizar el consumo del Arduino. Si se hiciese en la función *loop* estaría todo el rato tomando datos, haciendo que el sistema sea menos eficiente al consumir más energía.

Como se puede observar, primero se realiza la cuenta del tiempo en alto de la señal PWM del sensor MH-Z19C, posteriormente, se aplica la fórmula que viene en su hoja de datos.

Del sensor PYTHOS-31 y SEN0249 se obtiene la señal analógica recogida y se transforma al valor correspondiente de la magnitud medida.

Se hace uso de la función *getOxygenData* para obtener el valor del O_2 del sensor SEN0322, todo ello gracias a la librería incluida.

Por último, se almacenan en la variable *dataToSend* los datos de las magnitudes obtenidas y se transmiten por I2C.

```
void requestEvent() {

    long tt = millis();
    int myVal = digitalRead(pwmPin);

    if (myVal == HIGH) {
        if (myVal != prevVal) {
            h = tt;
            tl = h - 1;
            prevVal = myVal;
        }
    } else {
        if (myVal != prevVal) {
            l = tt;
            th = l - h;
            prevVal = myVal;
            ppm = 2000 * (th - 2) / (th + tl - 4);
        }
    }
}
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
// Lectura de datos del sensor de oxígeno
oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

// Lectura de humedad y conversión a porcentaje
int humRaw = analogRead(humPin);
humidity_percent = map(humRaw, 0, 1023, 0, 100);

// Lectura del sensor de pH
int phRaw = analogRead(phPin);
pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

// Mostrar valores por pantalla
Serial.print("Humedad (%): ");
Serial.println(humidity_percent);
Serial.print("CO2 (ppm): ");
Serial.println(ppm);
Serial.print("Oxígeno (%vol): ");
Serial.println(oxygenData);
Serial.print("pH: ");
Serial.println(pHValue);

// Enviar datos por I2C
byte dataToSend[14];
dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte of humidity
dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte of humidity
dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
dataToSend[5] = ppm & 0xFF; // Low byte of CO2
dataToSend[6] = int(oxygenData); // Valor entero de oxígeno
dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
dataToSend[8] = (int(pHValue) >> 8) & 0xFF; // High byte of pH
dataToSend[9] = int(pHValue) & 0xFF; // Low byte of pH
dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte decimal de pH

Wire.write(dataToSend, sizeof(dataToSend));
}
```

Código del segundo modo (toma periódica de datos)

Al igual que pasa en el primer modo, se incluyen las librerías y variables necesarias. Como se quiere disminuir el consumo, solamente se toman medidas durante cierto tiempo y, después, se pone el Arduino en modo de descanso. Se crea la constante MEAS_TIME para indicar el tiempo que van a durar estas mediciones junto con otras variables para

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

controlar el consumo energético del Arduino. También, se pone el tanto por ciento de los límites para efectuar la media *winsorizada* y truncada.

```
#include <Wire.h>
#include "DFRobot_OxygenSensor.h"

#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH
float ppmAvg = 0.0; // Variable para la media de CO2 durante 30 segundos
unsigned int sampleCount = 0;

// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor

// Tanto por ciento del los límites
#define limPercent 40

// Tiempo en que se toman las medidas en ms
#define MEAS_TIME 30000

unsigned long requestStartTime = 0;
bool isDataCollecting = false;

void requestEvent();
```

Dependiendo de lo que se prefiera utilizar, se crea la función *calculateAverage* para realizar la media *winsorizada*.

```
float calculateAverage(float values[], int count) {
    // Calcular la media inicial
    float sum = 0.0;
    for (int i = 0; i < count; i++) {
        sum += values[i];
    }
    float initialMean = sum / count;

    // Calcular los límites superior e inferior
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
float upperLimit = initialMean * (1.0 + limPercent / 100.0);
float lowerLimit = initialMean * (1.0 - limPercent / 100.0);

// Calcular la nueva media winsorizada
int validCount = 0;
float sumValid = 0.0;
for (int i = 0; i < count; i++) {
    if (data[i] >= lowerLimit && data[i] <= upperLimit) {
        sumValid += data[i];
        validCount++;
    } else if (data[i] < lowerLimit) {
        sumValid += lowerLimit;
        validCount++;
    } else {
        sumValid += upperLimit;
        validCount++;
    }
}
return sumValid / validCount;
}
```

O también, se puede utilizar la media truncada.

```
float calculateAverage(float values[], int count) {
    // Calcular la media inicial
    float sum = 0.0;
    for (int i = 0; i < count; i++) {
        sum += values[i];
    }
    float initialMean = sum / count;

    // Calcular los límites superior e inferior
    float upperLimit = initialMean * (1.0 + limPercent / 100.0);
    float lowerLimit = initialMean * (1.0 - limPercent / 100.0);

    // Calcular la media truncada
    sum = 0.0;
    int validCount = 0;
    for (int i = 0; i < count; i++) {
        if (values[i] >= lowerLimit && values[i] <= upperLimit) {
            sum += values[i];
            validCount++;
        }
    }

    if (validCount > 0) {
        return sum / validCount;
    } else {
        return initialMean;
    }
}
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
}
```

Después de transmitir los datos hay que reiniciar los valores con la función *resetAverages()*.

```
void resetAverages() {
    for (int i = 0; i < sampleCount; i++) {
        ppmValues[i] = 0.0;
        humidityValues[i] = 0.0;
        oxygenValues[i] = 0.0;
        pHValues[i] = 0.0;
    }
    sampleCount = 0;
}
```

La función *setup* prácticamente no se modifica.

```
void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);

    // Inicialización del sensor de oxígeno
    while (!oxygen.begin(Oxygen_IICAddress)) {
        Serial.println("Error al conectar con el sensor de oxígeno.");
        delay(1000);
    }
    Serial.println("Conexión exitosa con el sensor de oxígeno.");
}
```

Se añade a la función *loop* un sistema para que con un *flag* se controle cuando va a estar en modo ocioso y cuando va a estar tomando datos el Arduino, gestionando de esta forma el consumo de batería.

```
void loop() {

    if (isDataCollecting) {
        unsigned long elapsedTime = millis() - requestStartTime;
        long tt = millis();
        int myVal = digitalRead(pwmPin);
        if (myVal == HIGH) {
            if (myVal != prevVal) {
                h = tt;
                tl = h - l;
                prevVal = myVal;
            }
        }
    }
}
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
        }
    } else {
        if (myVal != prevVal) {
            l = tt;
            th = l - h;
            prevVal = myVal;
            ppm = 2000 * (th - 2) / (th + tl - 4);
        }
    }

    // Lectura de datos del sensor de oxígeno
    oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

    // Lectura de humedad y conversión a porcentaje
    int humRaw = analogRead(humPin);
    humidity_percent = map(humRaw, 0, 1023, 0, 100);

    // Lectura del sensor de pH
    int phRaw = analogRead(phPin);
    pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

    if (elapsedTime >= MEAS_TIME) { // 30 segundos
        // Calcular las medias de cada sensor durante 30 segundos
        float avgCO2 = calculateAverage(ppmValues, sampleCount);
        float avgHumi = calculateAverage(humidityValues, sampleCount);
        float avgOxygen = calculateAverage(oxygenValues, sampleCount);
        float avgPH = calculateAverage(pHValues, sampleCount);

        // Enviar medias por I2C
        byte dataToSend[14];

        // Mostrar valores por pantalla
        Serial.print("Humedad (%): ");
        Serial.println(humidity_percent);
        Serial.print("CO2 (ppm): ");
        Serial.println(ppm);
        Serial.print("Oxígeno (%vol): ");
        Serial.println(oxygenData);
        Serial.print("pH: ");
        Serial.println(pHValue);

        // Enviar datos por I2C
        byte dataToSend[14];
        dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte
        of humidity
        dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte
        of humidity
        dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
        dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
        dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
dataToSend[5] = ppm & 0xFF;           // Low byte of CO2
dataToSend[6] = int(oxygenData);       // Valor entero de oxígeno
dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
dataToSend[8] = (int(pHValue) >> 8) & 0xFF;    // High byte of pH
dataToSend[9] = int(pHValue) & 0xFF;          // Low byte of pH
dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte decimal de pH

Wire.write(dataToSend, sizeof(dataToSend));

// Reiniciar variables para la siguiente medición
resetAverages();
isDataCollecting = false;
}

}

}
```

4.5. Comunicación Arduino-Raspberry y almacenamiento de los datos

Como broche final de la implementación de este nodo se va a exponer cómo se ha implementado la comunicación entre el Arduino Uno, que es la placa para el subsistema sensor, con la Raspberry Pi 4 que actuará como receptor de los datos y los almacenará en un archivo con formato JSON.

Se podría realizar tanto una transmisión inalámbrica como por cable. Si se quisiese hacer de manera inalámbrica se tendría que realizar un punto de acceso con la Raspberry trabajando como servidor y el Arduino como cliente que le enviaría de forma periódica los datos de medición. Es una solución factible y totalmente lógica. Pero la distancia entre el Arduino y la Raspberry es relativamente corta, por tanto, hacer la conexión a través de cables es la opción más confiable, práctica y la elegida para este proyecto.

Al igual que pasaba en la transmisión de los datos, también hay diversas formas en las que se pueden almacenar los datos procesados, se podría almacenar en texto plano, binario, hexadecimal, XML, CSV, entre otros tantos. Pero se ha optado por utilizar ficheros JSON por su comodidad, simplicidad y al ser fácilmente legible.

El código utilizado en el maestro (Raspberry Pi 4) está recogido en el [apéndice F](#) para su consulta. A continuación, se pasa a explicar las distintas partes de dicho código:

Primero, se añaden distintas librerías para cumplir con el propósito. Se hace uso de la librería *smbus* para la comunicación por I2C, la librería *time* para trabajar correctamente con el tiempo y *JSON* para el almacenaje de los datos.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
import smbus
import time
import json
```

Se inicializa el *ctr_pin* que es el que se encarga de encender y apagar las PCB de hoja y de suelo. Se generan variables globales para inicializar el bus I2C y otra para indicar la dirección. También, se añade el tiempo para activar el sistema y el tiempo entre mediciones. Las variables anteriores se pueden cambiar en función del caso o, incluso, añadir más direcciones de Arduino si se incluyesen más dispositivos a la comunicación del sistema.

```
# Configuración del pin GPIO para el LED
ctr_pin = 15
GPIO.setmode(GPIO.BCM)
GPIO.setup(ctr_pin, GPIO.OUT)

# Dirección I2C del Arduino
arduino_address = 0x0B

# Inicialización del bus I2C
bus = smbus.SMBus(1)

# Tiempo en segundos entre peticiones en segundos
measurement_time = 1200

# Tiempo en segundos entre activar sistema y toma de medida
wakeup_time = 20
```

Ulteriormente, se muestra la función *read_data* que realiza toda la parte de adquisición de los datos transmitidos por I2C y su posterior almacenaje.

Se lee del terminal toda la trama, para después segmentarla en los datos de las distintas magnitudes, de idéntica forma a como se transmitió por el Arduino. A continuación, se transforman para almacenarlos de forma temporal en un diccionario.

```
# Leer 14 bytes de datos desde el Arduino
data = bus.read_i2c_block_data(arduino_address, 0, 14)

# Decodificar los datos
humidity_high = data[0]
humidity_low = data[1]
co2_high = (data[2] << 24) | (data[3] << 16) | (data[4] << 8) |
data[5]
oxygen_int = data[6]
oxygen_decimal = data[7]
ph_high = data[8]
ph_low = data[9]
ph_decimal = data[10]
```

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

```
# Recuperar los valores
humidity_percent = (humidity_high << 8 | humidity_low)
co2 = co2_high
oxygen = f"{oxygen_int*10}.{oxygen_decimal*10}"
phValue = ((ph_high << 8 | ph_low) + (ph_decimal / 100.0)) * 4

# Crear un diccionario con los datos
data_dict = {

    "timestamp": int(time.time()),
    "humidity": humidity_percent,
    "co2": co2,
    "oxygen": oxygen,
    "pH": pHValue

}
```

Una vez tratados correctamente los datos recibidos y guardados temporalmente en un diccionario, estos se van a almacenar en un archivo con el nombre del mes y año en el que se tomó. De esta forma, se almacena de manera automática cada mes, si ocurriese cualquier imprevisto en el sistema final (en el cual este trabajo es un nodo de una red IoT) se pueden recoger de manera fácil los datos directamente desde el nodo.

```
# Obtener la fecha actual (mes y año)
now = time.localtime()
file_name = f"{now.tm_mon}_{now.tm_year}.json"

# Guardar los datos en el archivo de respaldo JSON
with open(file_name, "a+") as file:
    json.dump(data_dict, file)
    file.write("\n")
```

Para finalizar, se realiza la función principal (*while True*) que se encargará de encender las PCB y sensores, llamar a la función anterior para recibir y almacenar los datos. Posteriormente, se apaga el resto del sistema y espera el tiempo indicado entre peticiones.

```
while True:
    GPIO.output(ctr_pin, GPIO.HIGH)
    time.sleep(wakeup_time)      # Tiempo para despertar al sistema
    read_data()
    time.sleep(1)
    GPIO.output(ctr_pin, GPIO.LOW)
    time.sleep(measurement_time) # Esperar 20 minutos entre cada lectura
```

Si se ejecuta el código por el terminal de la Raspberry Pi mientras el Arduino va tomando los datos, se observa que por el protocolo I2C se reciben correctamente los datos.

IMPLEMENTACIÓN DEL PRIMER NODO SENSOR

Humedad (%): 23
CO₂ (ppm): 152
Oxígeno (%vol): 20.70
pH: 8.4
SDatos guardados correctamente:
Humedad (%): 22
CO₂ (ppm): 331
Oxígeno (%vol): 20.60
pH: 8.24
Datos guardados correctamente:
Humedad (%): 20
CO₂ (ppm): 332
Oxígeno (%vol): 20.70
pH: 8.48
Datos guardados correctamente:
Humedad (%): 19
CO₂ (ppm): 151
Oxígeno (%vol): 20.70
pH: 8.24
S█

Figura 4.5.1 Comprobación del funcionamiento del primer nodo implementado

5. Implementación del segundo nodo sensor

En este apartado, se detallarán los pasos necesarios en la creación del segundo nodo. Al igual que en el primer nodo, se ha desarrollado para aportar flexibilidad ante posibles modificaciones en términos de expansión, reducción o cambio de sus componentes.

Esto supone que el modelo sea altamente adaptable y escalable. La capacidad de transformación se evidencia tanto en la posibilidad de ajustar la composición del nodo como en la capacidad de modificar los procesos asociados con la recopilación y gestión de los datos.

5.1. Planificación del nodo

Este nodo se diferencia porque intenta suprimir la necesidad de un HW externo a los sensores y a los dispositivos de Dragino, constará de distintos modos de funcionamiento, los cuales contienen distintos sensores en cada modo.

La idea original en la toma de datos consiste en realizar directamente las conexiones entre el sensor y la placa sin necesidad de añadir circuitos impresos ni módulos. También se observa que difiere tanto en la política de preprocesamiento de datos, intentando minimizarlo para enviar los datos en crudo de las mediciones obtenidas, como en la forma de alimentación, al utilizar una batería con un pequeño panel fotovoltaico.

Posteriormente, sería necesario implementar un SW que se encargase de la adquisición y del tratamiento de los datos recibidos de los sensores. Se realizará el almacenamiento en la nube, esto queda fuera de los objetivos del presente proyecto, al enfocarse en el nodo de recolección de datos.

Antes de empezar, se realiza un pequeño bosquejo donde se recoge la alimentación y las conexiones entre los sensores y el dispositivo de Dragino.



Figura 5.1.1 Esquema conceptual del segundo nodo sensor

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

5.1.1. Modos de funcionamiento

Antes de comenzar esta sección, es necesario entender que los dispositivos de Dragino están pensados para funcionar correctamente en entornos abiertos al estar diseñado para la toma de datos en entornos rurales, como campos de cultivos, granjas, silos...

Además, posee una carcasa protectora que protege de las inclemencias climáticas y del daño que le pudieran producir los animales al sistema. Con esto se concluye que es una buena elección para poder aplicarlo a diversos entornos forestales como encinares, pinares, hayedos...

En esta sección se van a implementar diversos modos de funcionamiento, cada uno contendrá distintas funcionalidades y hará uso de distintos tipos de sensores. Pero previamente, se analizará el funcionamiento de las placas NBSN95 y LSN50, los modos diseñados por el fabricante y se expondrán los distintos sensores que se van a utilizar y sus características principales.

Funcionamiento de la NBSN95 y LSN50

Empezando por los dispositivos de Dragino, ambos están diseñado para el registro de datos al aire libre para un uso a largo plazo y una transmisión segura de datos. La principal diferencia entre ambos dispositivos es la forma en la que se realiza la comunicación, el nodo NBSN95 hace uso de *NarrowBand-IoT* (NB-IoT) a través del sistema global para las comunicaciones móviles (GSM). En cambio, el dispositivo LSN50 hace uso de la comunicación vía LoRa. Se ha de tener en cuenta que al poseer distintos modos de comunicación estos dispositivos, su implementación SW variará.

Es importante destacar que ambas placas son hermanas, es decir, tienen el μC STM32L072 (70), poseen mismo número de puertos, estos tienen prácticamente las mismas funcionalidades, tienen los mismos modos de funcionamiento de fábrica, entre otras muchas cosas. En consecuencia, gran parte del código desarrollado para el diseño de una de las placas se podrá reutilizar para integrarlo en la otra.

En las siguientes figuras se muestra el esquema electrónico de ambas placas, esto será fundamental para avanzar en el proyecto.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

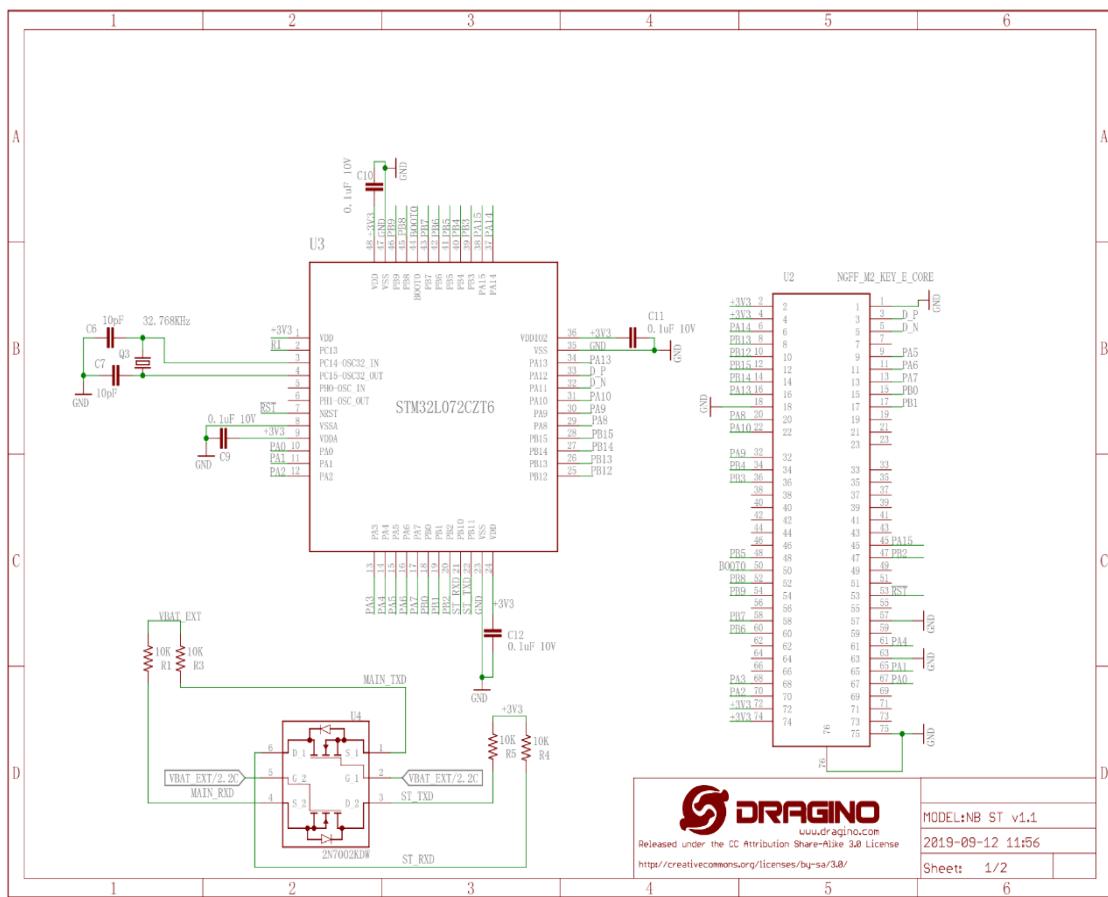


Figura 5.1.2 Primera parte del esquema electrónico del dispositivo NBSN95 de Dragino

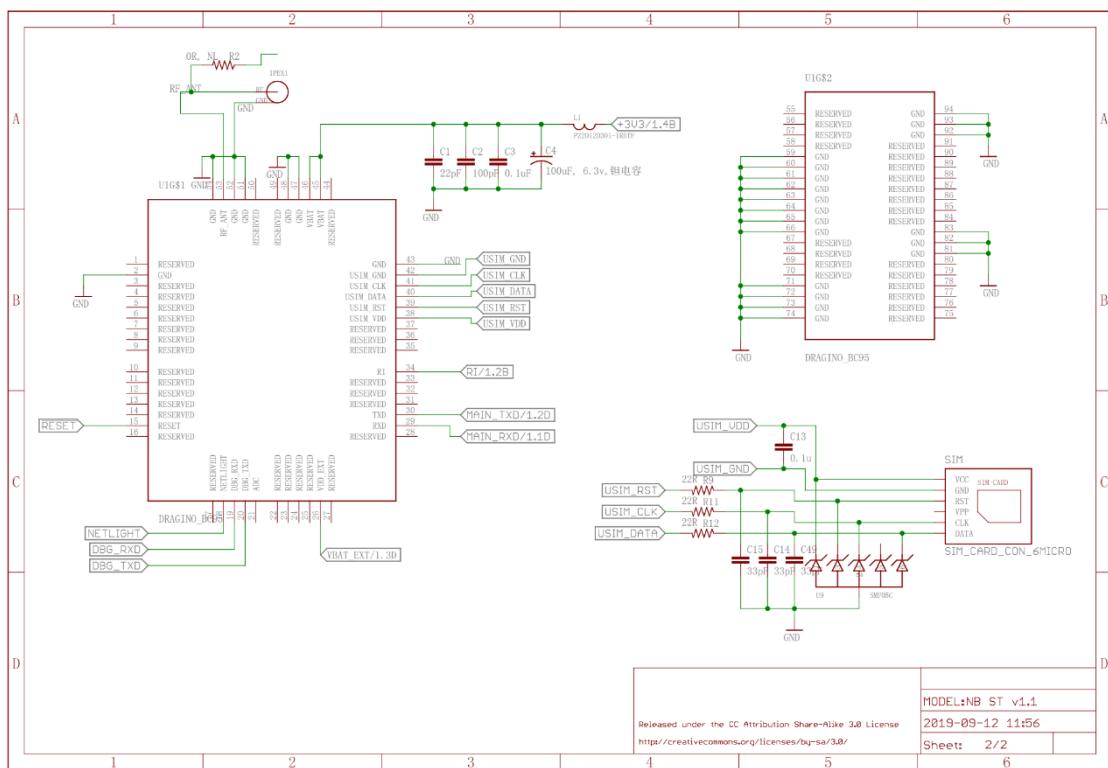


Figura 5.1.3 Segunda parte del esquema electrónico del dispositivo NBSN95 de Dragino (71)

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

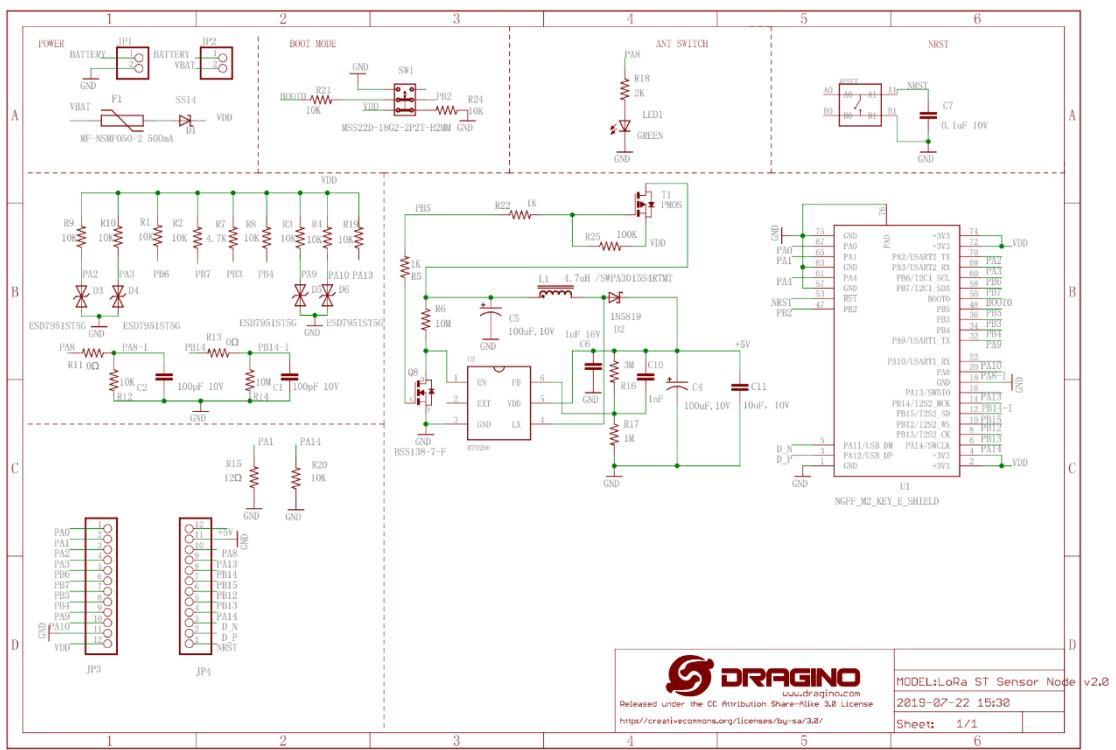


Figura 5.1.4 Primera parte del esquema electrónico del dispositivo LSN50 de Dragino

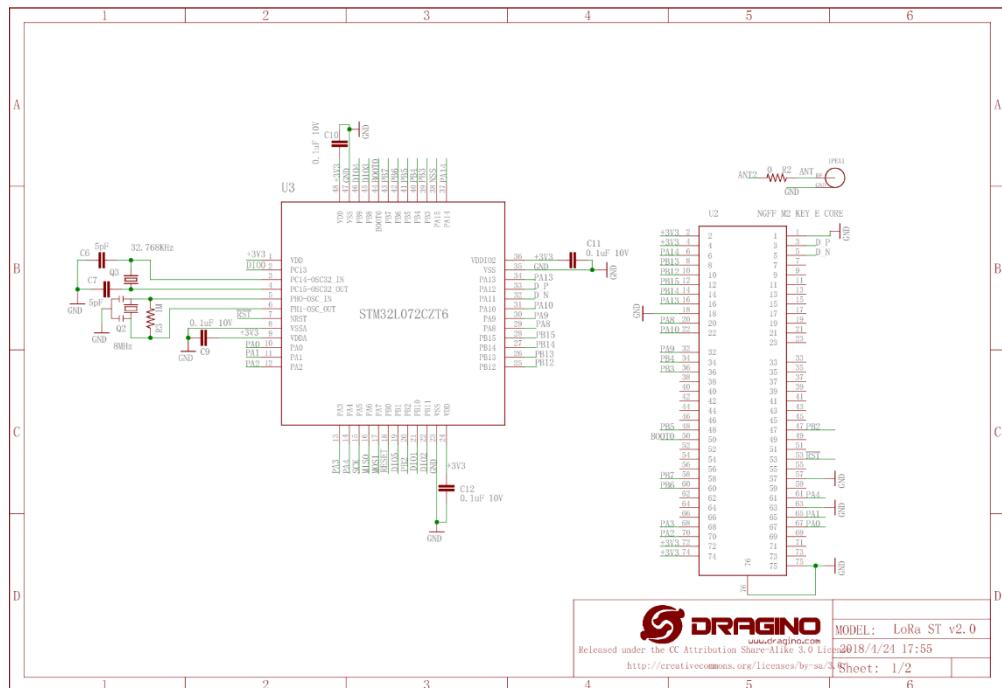


Figura 5.1.5 Segunda parte del esquema electrónico del dispositivo LSN50 de Dragino (72) (73)

Ambos dispositivos vienen con un código previo de fábrica, este se ha de compilar con la herramienta *Keil μVision 5*, pero se ha de instalar tanto el SW necesario para trabajar con el *μC STM32L072* y la quinta versión del compilador, cualquier otra versión no funcionaría o habría que realizar múltiples alteraciones al código original.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

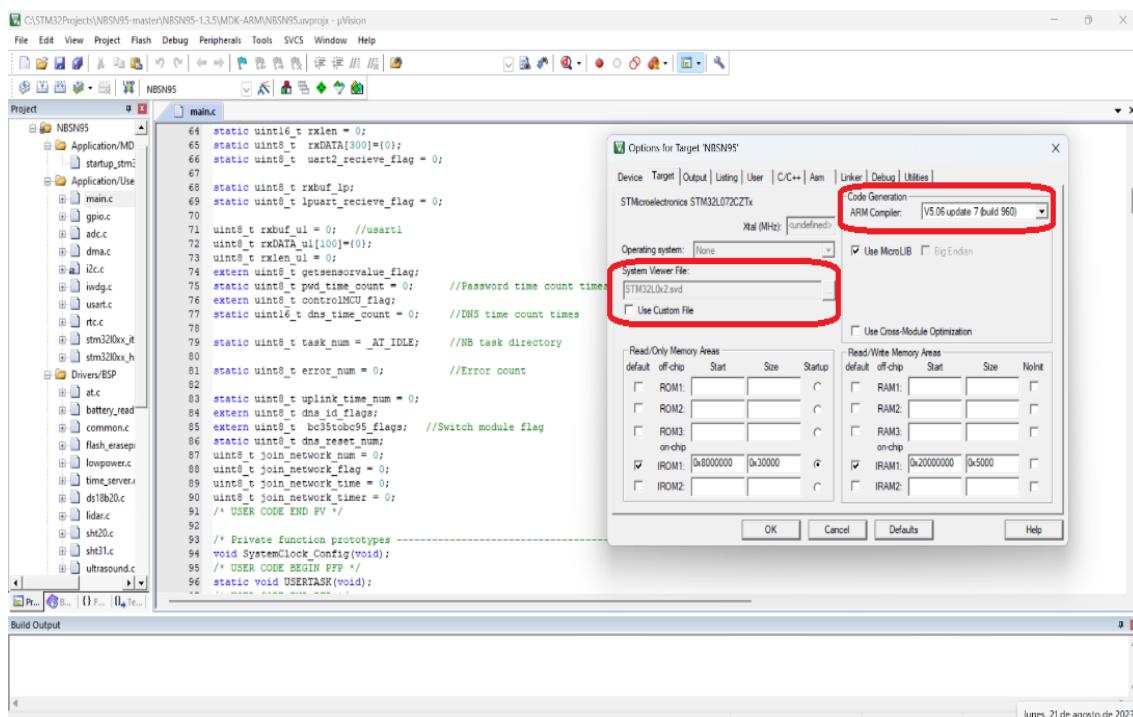


Figura 5.1.6 Configuración del Keil μ Vision 5 para el segundo nodo implementado

Una vez se compila el código, se genera un fichero hexadecimal que se tendrá que subir a la placa utilizando la herramienta *STM32CubeProgrammer*. Para ello hay que indicarle la ruta donde se encuentra el archivo y ponerle los parámetros de configuración que se muestran a la derecha de la figura 5.1.7.

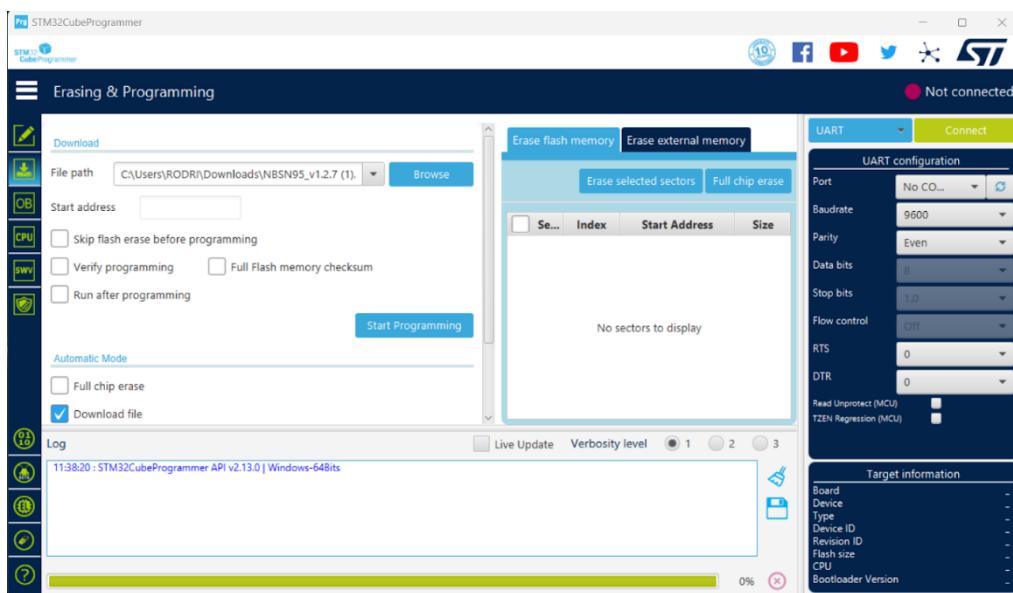


Figura 5.1.7 Configuración del STM32CubeProgrammer para el segundo nodo implementado

Para ver que funciona el sistema correctamente se precisa de un conversor USB2.0 a TTL y con la herramienta *SW Serial Port Utility* se podrá entablar un diálogo con los dispositivos, para ello será necesario poner los parámetros de configuración de la figura 5.1.9 para que se comuniquen correctamente.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

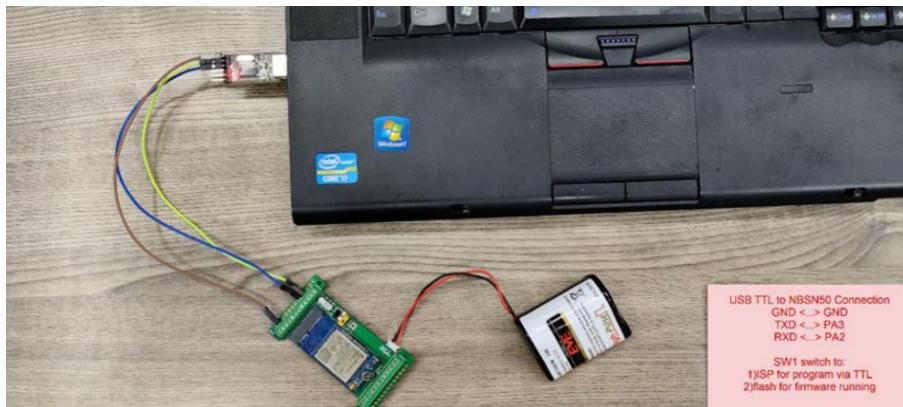


Figura 5.1.8 Conexión TTL entre cualquiera de los dos dispositivos de Dragino y el PC

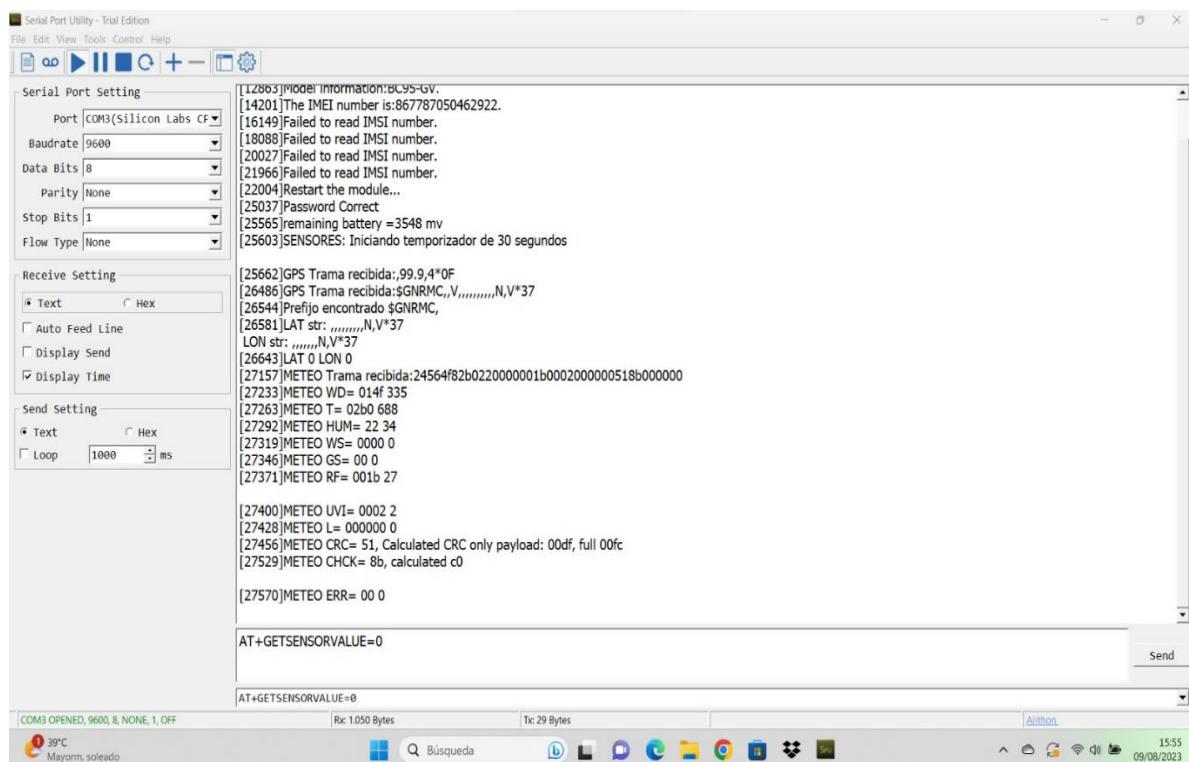


Figura 5.1.9 Salida del primer modo implementado (modo 7) en el *Serial Port Utility*

Modos diseñados por el fabricante

Haciendo uso del manual de usuario (74) (75) de ambas placas, se comprueba que sendos dispositivos poseen unos modos de funcionamiento previos y comunes. Estos modos utilizan distintos tipos de sensores y se pueden intercambiar con el comando AT+CFGMOD = <Modo>.

Es fundamental, para avanzar de forma adecuada, estudiar los modos que vienen ya integrados en los dispositivos proporcionados por Dragino, así se podrán ver cuáles son los puntos en común a tratar y las discrepancias que habrá que solucionar.

En la siguiente tabla se observa el estudio del uso de los sensores en los distintos modos ya implementados por Dragino.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

	DRAGINO	Modos de funcionamiento						Tipo de comunicación					
	Sensor	1	2	3	4	5	6	RS485	RS232	I2C	ADC	GPIO	1-WIRE
	DS18B20	1	1		3	1	1						X
	Sensores ADC	1	1	3	1	1	1						X
	Contador de pulsos						1						X
	SHT31/AHT20		1										X
	Sensores I2C			1/-									X
Uno u otro	Entrada digital	1	1	1	1	1	1						X
	Entrada interrupción	1	1	1	1	1	1						X
Uno u otro	Lidar		1										X
	Ultrasonido		1										X
	HX711/PESO						1						X
	Total	4	4	4/-	5	4	4						

Tabla 5.1.1 Relación entre los sensores y modos de funcionamiento del fabricante / tipo de comunicación

En la tabla 5.1.1 se pueden observar tres tipos de columnas, la que tiene como nombre "Sensor" muestra los distintos sensores que utiliza Dragino en los distintos modos que implementó. En las columnas con nombre "Modos de funcionamiento" se reflejan la cantidad de sensores utilizados en el modo indicado por cada una de las columnas, debajo de estas se muestra la cantidad total de sensores utilizados en cada modo. Las columnas nombradas como "Tipo de comunicación" indican el tipo de comunicación que utiliza el sensor indicado. Es necesario recalcar que a la izquierda se muestran 4 filas con dos colores distintos que pone "Uno u otro", esto indica que solamente se puede utilizar un sensor u otro a la vez.

Sensores

Los sensores que se pueden usar con los modos del fabricante son los que se muestran en la siguiente tabla 5.1.2, es necesario explicar que también tiene preparado modos para utilizar sensores digitales, contadores de pulsos, ADC e interrupciones.

Modelo	Magnitud
DS18B20	Tº y humedad suelo
SHT20	Tº y humedad aire
SEN0313	Distancia

Tabla 5.1.2 Sensores utilizados por el fabricante

Será importante incorporar y volver a emplear sensores que resulten efectivos en la supervisión de entornos forestales. En este sentido, se ha optado por la utilización de los siguientes sensores:

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

Modelo	Magnitud
Sensecap	EC, salinidad...
DS18B20	Tº y humedad suelo
SHT20	Tº y humedad aire
NEO-6M	GPS
YF-S201	Caudal
SEN0313	Distancia
Sainlogic WS 3500 Plus	Estación meteorológica

Tabla 5.1.3 Sensores pensados para implementar el segundo nodo

Una vez visto los sensores que se van a utilizar, es necesario estudiar los datos que devuelve el sensor. En las siguientes tablas se presentan los datos que devuelven los sensores más complejos, dispuestos en el orden en que serán enviados. Aquí se distinguen dos tipos de datos: los datos de configuración o estado, marcados en color rojo, y los datos útiles provenientes de las magnitudes medidas por los sensores, marcados en blanco.

Es relevante mencionar que solo se transmitirán los datos útiles y, por lo tanto, los datos de configuración en color rojo no serán enviados. El orden en que aparecen las distintas magnitudes en esta hoja corresponde al orden en que serán recibidas. Además, se han agregado campos adicionales para indicar si la variable es con signo [s] o sin signo [u], los rangos que indican los límites de medición y las unidades de cada magnitud medida por los sensores.

ESTACION METEOROLÓGICA							
SIGNO	SIZE	SIZE	VARIABLE	DESCRIPCION	RANGO	UM	EQU
		8	FC	Family code			
		8	SC	Security Code			
u	8		WD	Wind direction	0/359	°	
u	1		WD_E	Wind direction error	0/1		
	2		F	FIX			
u	1		LB	Low Battery	0/1		
u	1		Tº_E	Temperature error	0/1		
u	10		Tº	Temperature	-40/60	°	Tº/10-40
u	1		HUM_E	Humidity error	0/1		
u	7		HUM	Humidity	1/99	%	
u	9		WS	Wind Speed		m/s	
u	1		GS_E	Gust Speed error	0/1		
u	7		GS	Gust Speed		m/s	
u	12		RF	Rainfall		in	
u	1		UVI_E	UltraViolet Index error	0/1		
u	15		UVI	UltraViolet Index		tbd	
	10		CRC	Checksum			
u	1		L_E	Light error	0/1		
u	23		L	Light			
u	6		P	Pressure			
SUMA	104	28					

Tabla 5.1.4 Datos útiles y de configuración que devuelve la estación meteorológica

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

SHT20							
SIGNO	SIZE	SIZE	VARIABLE	DESCRIPCION	RANGO	UM	EQU
s	16		T°	TemperaturaAire		°C	
u	16		HUM	HumedadAire		%	
		14	CRC	Checksum			
SUMA	32	14					

Tabla 5.1.5 Datos útiles y de configuración que devuelve el sensor SHT20

SENSACAP							
SIGNO	SIZE	SIZE	VARIABLE	DESCRIPCION	RANGO	UM	
s	14		T°	Temperatura	-40.00/80.00	°C	
u	14		VWC	Volumetric Water Content	0.00/100.00	%	
u	15		EC	Electrical Conductivity	0/20000	us/cm	
u	15		SAL	Salinity	0/20000	mg/L	
u	15		TDS	Total Dissolved Solids	0/20000	mg/L	
u	14		EPS	Dielectric Constant	0.00/82.00	-	
		1	T°U	Degree Unit			
		7	ECTC	EC Temp Coff			
		7	SALC	Salinity Coff			
		7	TDSC	TDS Coff			
		8	SA	Slave Addr			
		3	BR	BaudRate			
		1	PRO	Protocol			
		2	PAR	Parity			
		1	DB	Data Bits			
		1	SB	Stop Bits			
		8	RD	Response Delay			
		8	AOI	Active Output Interval			
SUMA	87	54					

Tabla 5.1.6 Datos útiles y de configuración que devuelve el sensor Sensecap

GPS							
SIGNO	SIZE	SIZE	VARIABLE	DESCRIPCION	RANGO	UM	
		5	SI	String Inicial			
		30	A	Altitud	Rango	Unidades	
s	28		N	Latitud		°	
s	28		W	Longitud		°	
		10	NS/NC	No sé			
		17	NS/NC	No sé			
		20	NS/NC	No sé			
		4	SF	String final			
SUMA	56	86					

Tabla 5.1.7 Datos útiles y de configuración que devuelve el sensor GPS

CONTADOR							
SIGNO	SIZE	SIZE	VARIABLE	DESCRIPCION	RANGO	UM	
u	28		CNT	PULSOS ACUMULADO		pulsos	
u	28		TIM	TIEMPO DE MEDICIÓN		ms	
u	28		MAXP	TAMAÑO MÁXIMO DE PULSO		ms	
u	28		MINP	TAMAÑO MÍNIMO DE PULSO		ms	
Total	112	0					

Tabla 5.1.8 Datos útiles y de configuración que devuelve el contador de pulsos

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

Nuevos modos implementados

Se van a diseñar nuevos modos que no interfieran con los modos dados por el fabricante, todo esto para hacer un sistema más completo donde se puedan utilizar los modos dependiendo del lugar en donde se sitúe el nodo y la función que este desempeñe.

Con base en la hoja de especificaciones del μC , se llevó a cabo una evaluación inicial para determinar la viabilidad de la integración de los sensores previamente vistos en el sistema. Este análisis se realizó con el objetivo de evitar interferencias entre los sensores, así como entre los modos creados por el fabricante y los nuevos modos diseñados.

ID Sensor	Magnitud	Sensor	DISEÑO							Modos de funcionamiento					Tipo de comunicación				
			7	8	9	10	11	12	13	RS485	RS232	I2C	ADC	GPIO	1-WIRE				
A	Tº Suelo	DS18B20						1							X				
B	-	Sensores ADC													X				
C	Caudal	Contador de pulsos				1	1	1	3						X				
D	Tº, hum aire	SHT31/AHT20			1	4			1						X				
E	Posición	GPS		1	1	1	1	1											
F	Lluvia	Pluviómetro			1										X				
G	Tº, hum suelo	Senscap			?	3	3								X				
H	Ambiente	Estación meteorológica (WH24C)	1												X				
Total			2	4/7	5	6	3	3	5										
¿Es viable?			Sí	Sí	Sí	Sí	Sí	Sí	Sí										

Tabla 5.1.9 Relación entre los sensores y modos de funcionamiento / tipo de comunicación

En la tabla superior se muestran los sensores utilizados en los nuevos modos de funcionamiento implementados. Además, se añade una columna extra, donde se indica la magnitud que mide el sensor correspondiente situado a su derecha. En la parte inferior se añade una fila donde se indica la viabilidad de cada uno de los modos.

Posteriormente, se realizó la tabla que se muestra a continuación. En esta se expone la relación entre los pines de la placa con los modos de funcionamiento y los tipos de comunicación utilizados.

J3	J4	ID	PIN	PU	ADC	I2C	SER	X	DRAGINO	M1	M2	M3	M4	M5	M6	X	NEMDRAGINO	M1	M2	M3	M4	M5	M6	int	DISEÑO	M7	M8	M9	M10	M11	M12	M13	
3		PA1		IN1	R4	ADC				X							ADC																
9		PA4			D3	R5	ADC																								ADC		
2		PA0		IN0	T4													ADC															
5		PA3	FU1OK	IN3	RGR8	UART BOOTLOADER				X	X	X	X	X	X																		
4		PA2	FU1OK	IN2	T2R8	UART BOOTLOADER				X	X	X	X	X	X																		
6		PF6	FU1OK	CIA	T1T2	I2C_SHT20		X	X	X							I2C_SHT20	X	X	X					n	I2C_SHT20		SHT20_C2		SHT20_C2		SHT20_C2	
7		PF7	FU1OK	D1A	R1	I2C_SHT20		X	X	X							I2C_SHT20	X	X	X					n	I2C_SHT20		SHT20_C2		SHT20_C2		SHT20_C2	
8		PF8	FU1OK		T5	GPIO-1W		X	X		X	X					GPIO-1W	X	X	X	X											PWR_I2C	1W
10		PA9	FU1OK	C1B	T1	GPIO-1W											UART USUARIO	X	X	X	X	X	X		ny	UART USUARIO		X	X	X	X	X	X
11		PA10	FU1OK	D1B	R1	GPIO-1W											UART USUARIO	X	X	X	X	X	X		ny	UART USUARIO		X	X	X	X	X	X
24		PA12				GPIO_CONTACT		X	X	X	X	X	X				GPIO_CONTACT	X	X	X	X	X			y	POWER	PWR_GPS	PWR_GPS	PWR_GPS	PWR_GPS	PWR_I2C		
19		PB15															GPIO-1W								n	POWER	PWR_GPS	PWR_GPS	PWR_GPS	PWR_GPS	PWR_I2C		
21		PB13			C2												GPIO-1W								n	POWER	PWR_GPS	PWR_GPS	PWR_GPS	PWR_GPS	PWR_I2C		
22		PA14				T2T8																			n	GPIO-ENABLE-RS485	EN485	EN485	EN485	EN485	PWR_I2C		
17		PA13			R8																												
18		PB14			D2	GPIO_PULSOS		X	X	X	X	X	X				GPIO_PULSOS	X	X	X	X	X			y	STLINK-SWDIO	SWDIO	SWDIO	SWDIO	SWDIO	SWDIO		
16		PA8			C3	MODEM LED											GPIO_PULSOS								n	GPIO_PULSOS	PULSOS	PULSOS	PULSOS	PULSOS	PULSOS		
20		PB12				GPIO_HX711/US		X			X						GPIO_HX711/US	X							n	MODEM LED	X	X	X	X	X	X	
23		PA11				GPIO_HX711/US		X			X						GPIO_HX711/US	X		X					y	POWER_I2C		PWR_I2C		PULSOS			
26		PA4			IN4	ADC																											

Tabla 5.1.10 Asignación de los pines a cada sensor y nodo

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

Yendo por columnas, se comprueba que las primeras indican la salida de la regleta de pines, junto con el número de pin correspondiente. Las dos posteriores columnas muestran el nombre o identificador del pin, en las siguientes se muestran todos los modos posibles de comunicación que tienen asignados los correspondientes pines.

También yendo por columnas, se pueden observar tres subtablas donde se indica el tipo de comunicación que se utiliza en cada modo.

- **La primera** muestra el estudio realizado de los pines para los distintos modos de funcionamiento de Dragino ya implementados.
- **La segunda** muestra los pines modificados para los modos de Dragino, este cambio, se hace para que sean compatibles estos modos con los deseados.
- **La tercera** revela la asignación de protocolos de funcionamiento a los pines en función del modo.

Con todo lo observado, se puede comprobar que para el modo 7, 10, 11, 12 y 13 se tienen los pines y las interfaces de comunicación necesarias para implementarlos adecuadamente. Es importante recalcar que para los modos que utilizan protocolos de comunicación I2C (SHT20.C2) no se hace uso de distintos pares de pines para implementarlo si no que solamente se utiliza un par y se usa el *enable* (EN.I2C) para activar el sensor correspondiente.

Se observa, a simple vista, que se puede saber qué sensores son los utilizados en cada pin y en cada modo, pero de igual forma se van a enumerar:

- PULSOS: Se utilizará para los contadores de pulsos
- 1-WIRE (1W): Se utilizará para el sensor DS18B20
- I2C (SHT-20): Se utilizará el canal 2 de comunicación I2C para los sensores SHT20.
- ADC: Se utilizará para los homónimos sensores de ADC.
- UART (232): Se utilizará para el sensor con protocolo de comunicación RS232 como es el GPS (PA2 y PA3).
- UART (485): Se utilizará para los sensores con protocolos de comunicación RS485 como son la estación meteorológica y los sensores Sensecap. Es necesario recalcar que a estos últimos sensores se les puede modificar la dirección, por tanto, se pueden utilizar todos estos sensores en el mismo canal. Al igual que ocurría en el primer modo implementado, para estos sensores se necesita el conversor MAX485.

Para finalizar el apartado, se lleva a cabo un proceso de recopilación de toda la información y conceptos presentados hasta el momento. Esta se realiza en un diagrama de bloques, el cual es una representación gráfica que muestra las diferentes partes o componentes del sistema que se va a implementar.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

El objetivo principal de este diagrama es brindar una visión general y clara del sistema, permitiendo visualizar de manera organizada la implementación de éste.

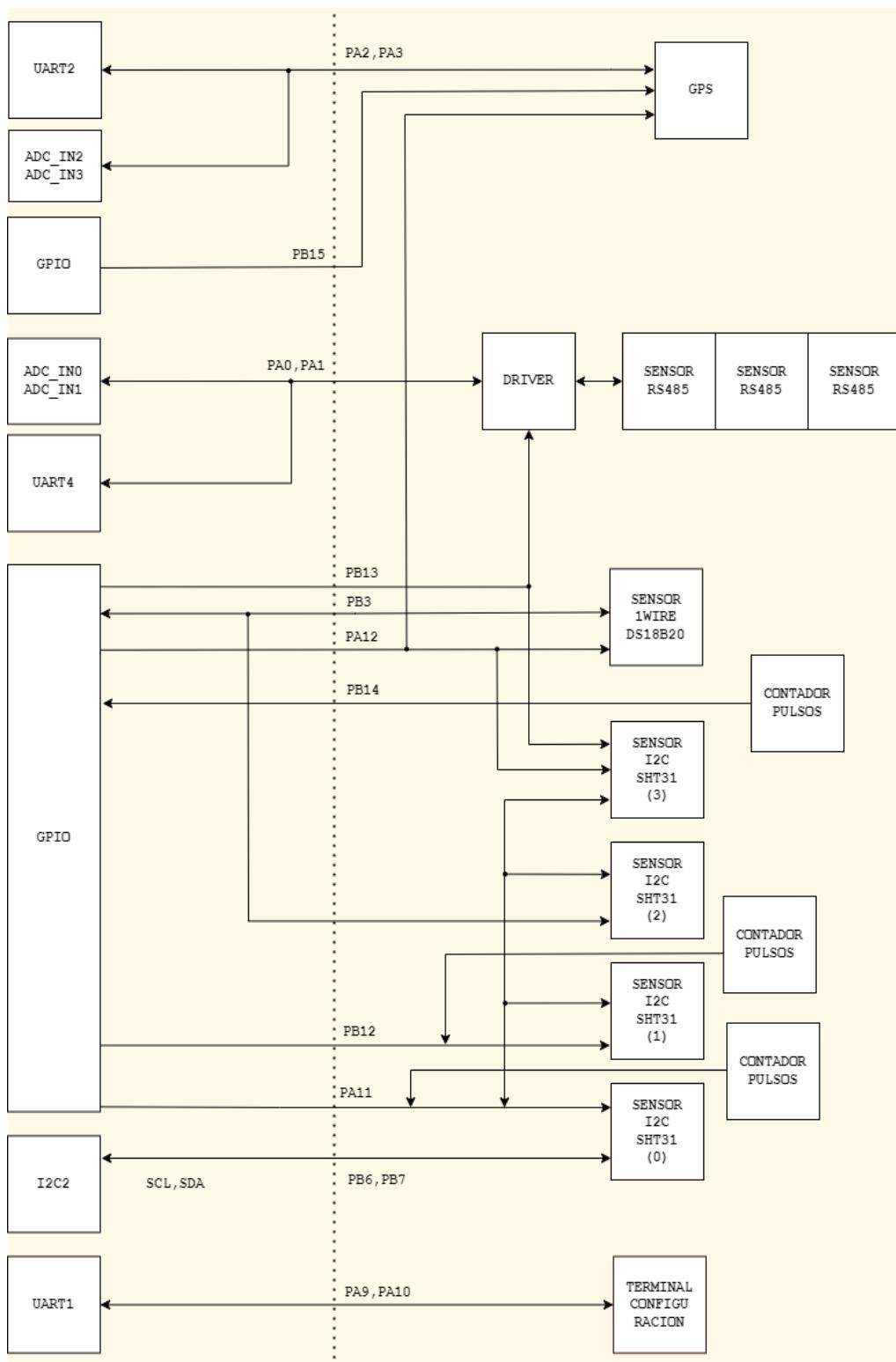


Figura 5.1.10 Diagrama conceptual de la asignación de pines del segundo nodo

Tras haber evaluado todos los aspectos, se verifica que todos los modos ideados operarán sin inconvenientes, contando con el espacio físico y las capacidades de comunicación necesarias. Con esta validación, se está listo para proceder con la implementación de los diversos modos de funcionamiento.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

5.2. Adquisición de los datos y módulos de conversión

Se va a detallar todas las conexiones que precisa el sistema para efectuar la integración de los sensores. Cada uno de los modos implementados vienen recogidos en los siguientes esquemas electrónicos, junto con pequeñas aclaraciones del funcionamiento del modo.

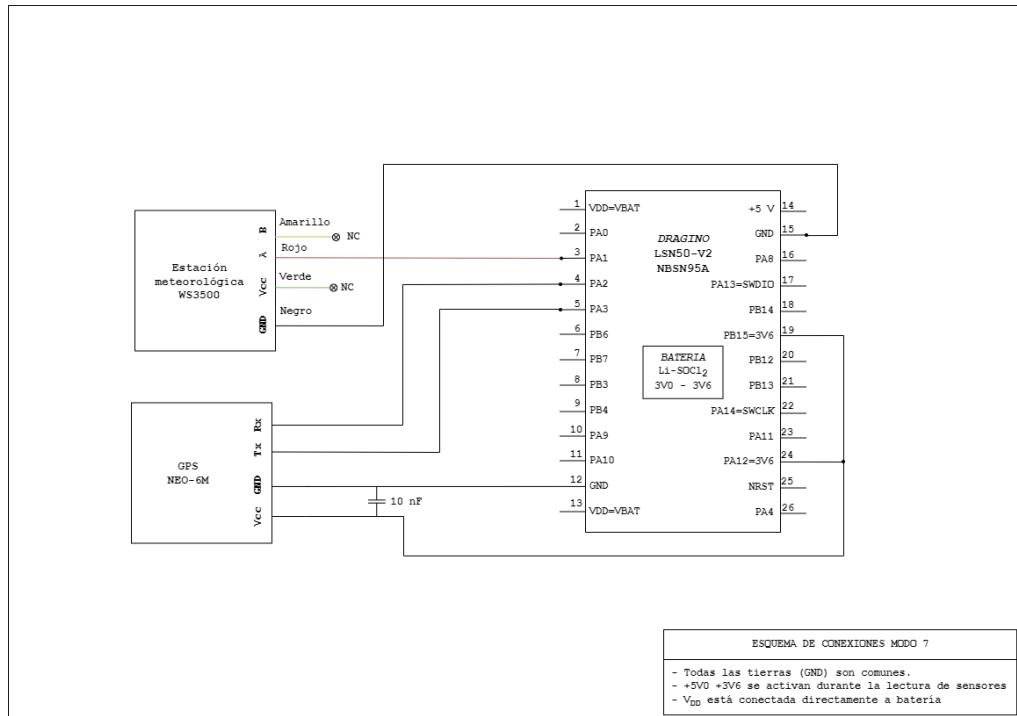


Figura 5.2.1 Esquema de conexiones del modo 7

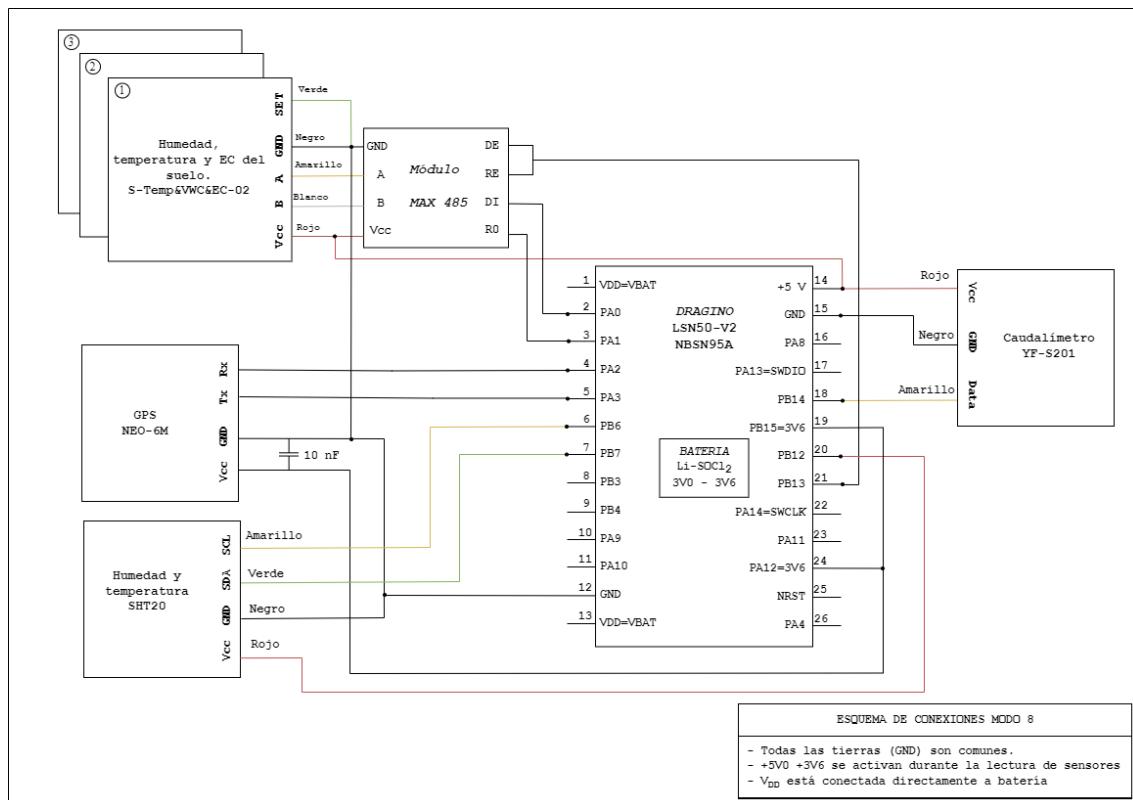


Figura 5.2.2 Esquema de conexiones del modo 8

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

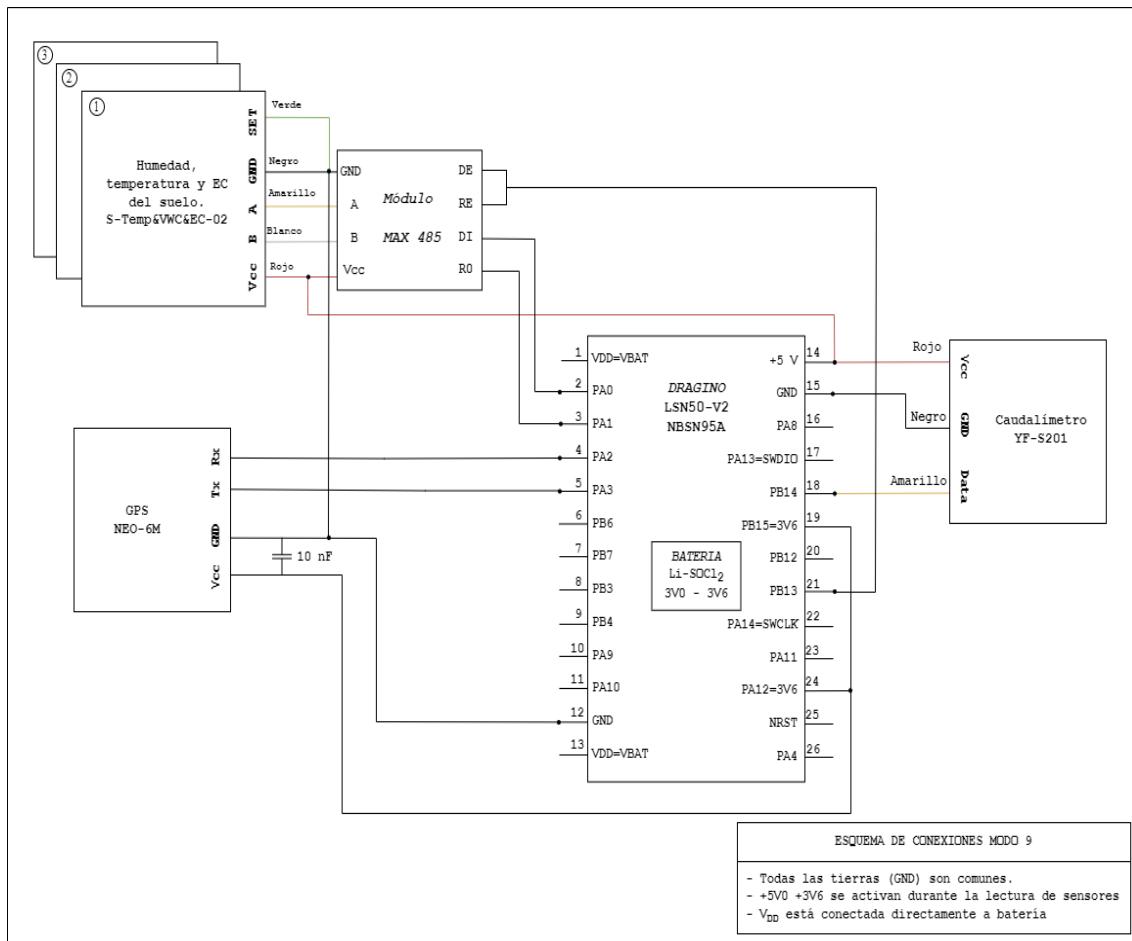


Figura 5.2.3 Esquema de conexiones del modo 9

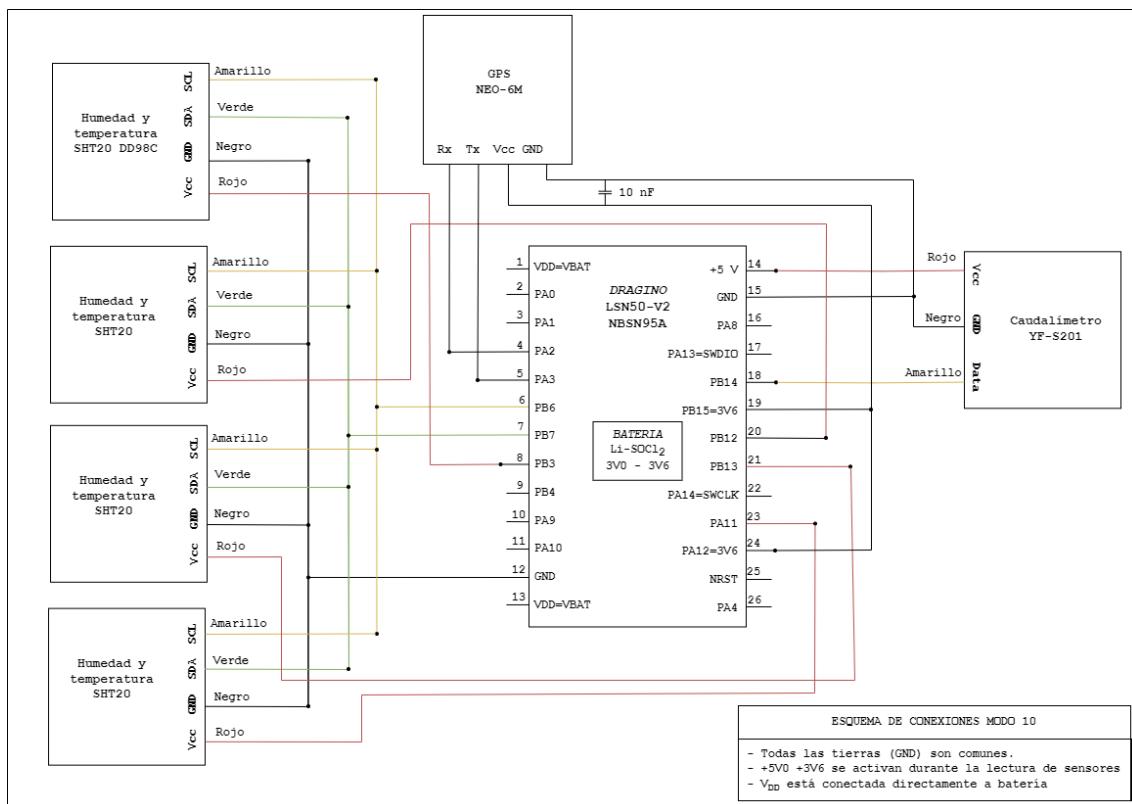


Figura 5.2.4 Esquema de conexiones del modo 10

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

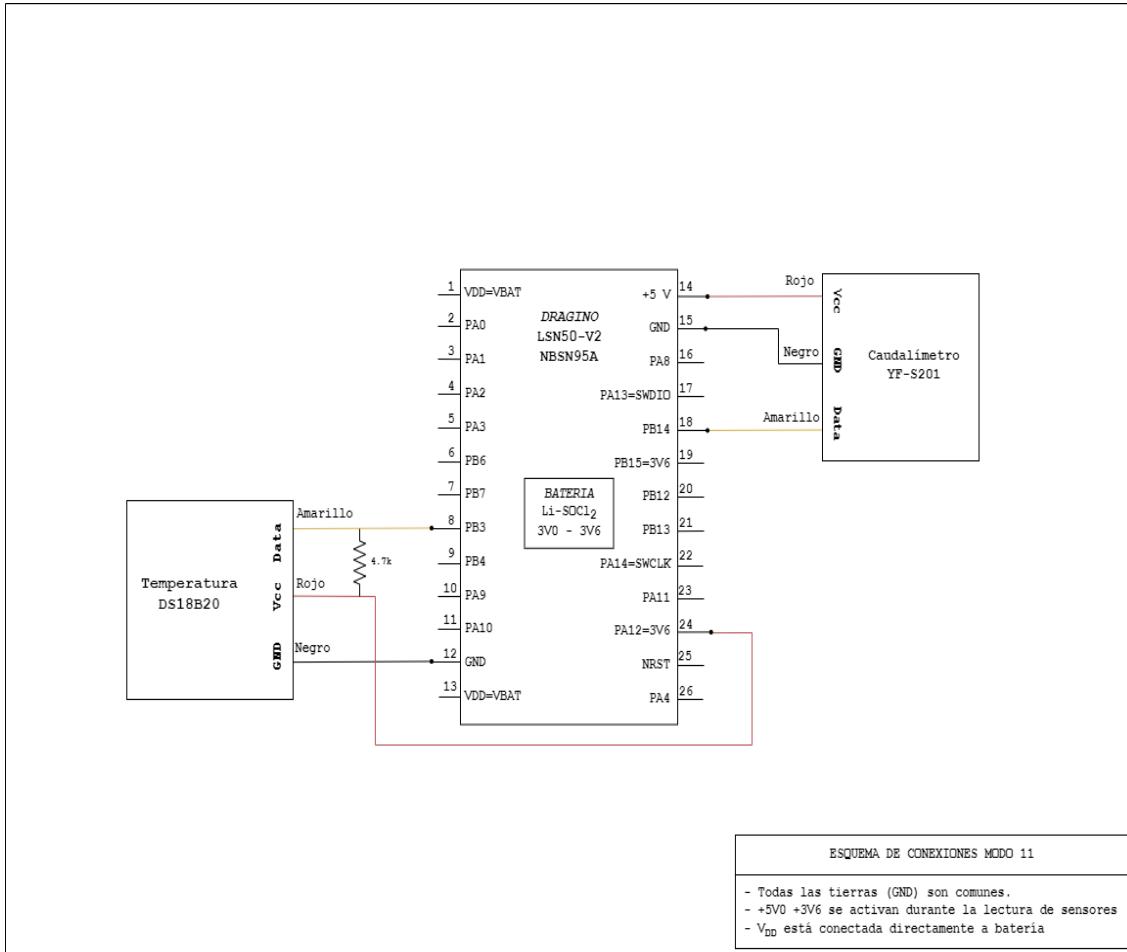


Figura 5.2.5 Esquema de conexiones del modo 11

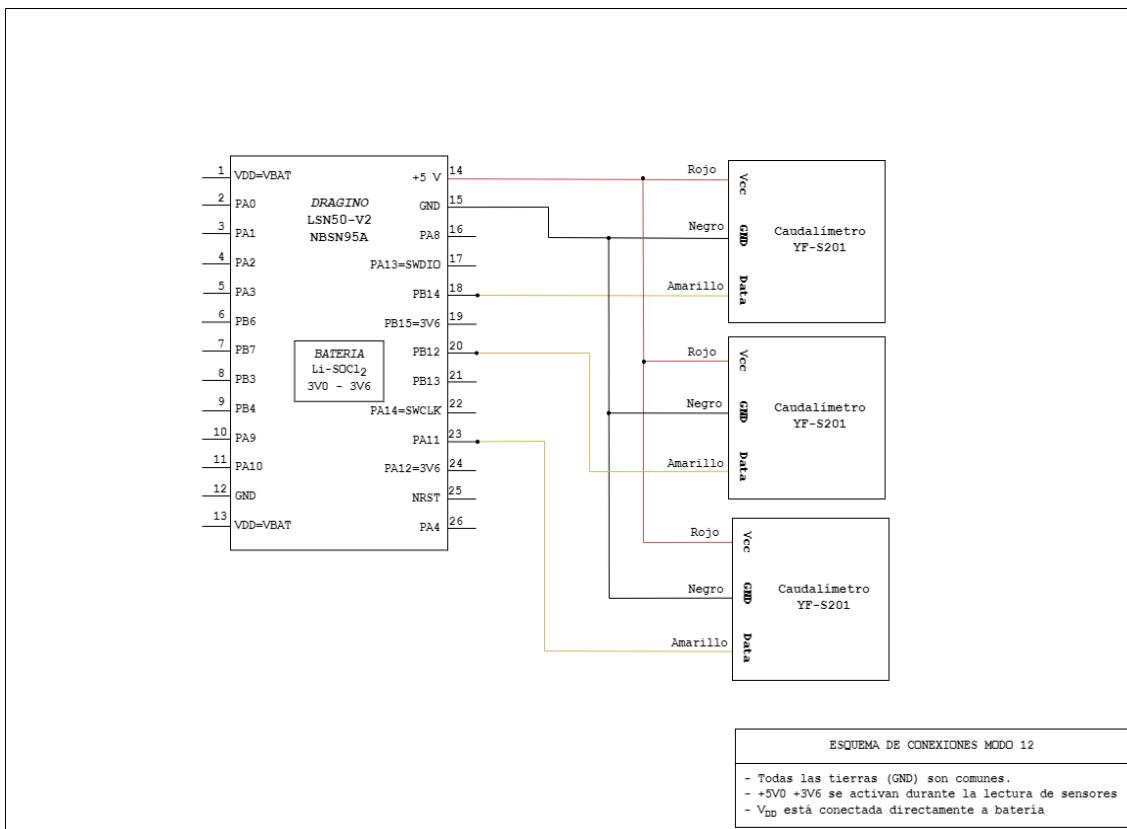


Figura 5.2.6 Esquema de conexiones del modo 12

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

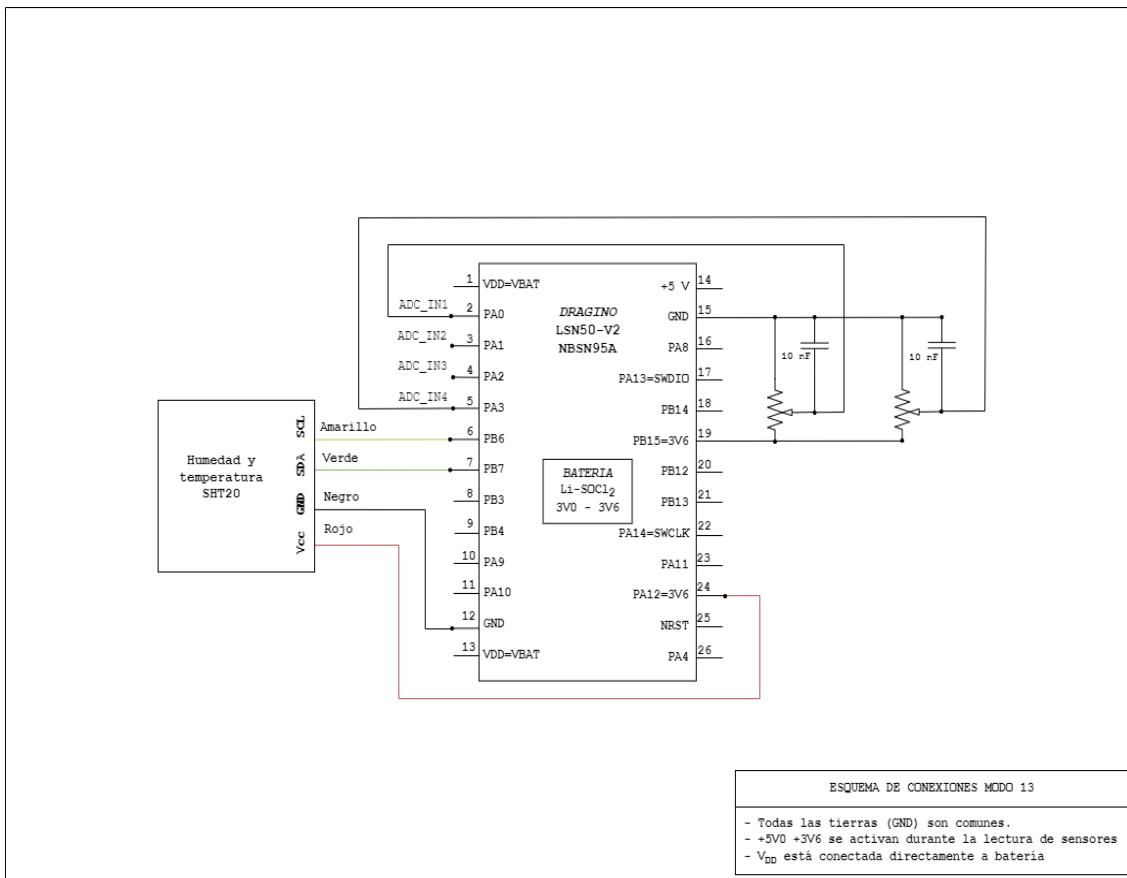


Figura 5.2.7 Esquema de conexiones del modo 13

5.3. Estudio de la recepción de los datos medidos y preprocesamiento

En este apartado se va a explicar la configuración que hay que implementar en los sensores, siempre y cuando sea necesaria, para poder realizar una correcta monitorización del sistema. También se explicarán las funcionalidades del código para el preprocesamiento necesario a la hora de generar las tramas que se van a transmitir, junto con el código efectuado para este propósito. El código completo se puede consultar en el [apéndice H](#).

- **GPS**

En el sensor de GPS se ha implementado un buffer circular donde se van almacenando, byte a byte, todos los datos que envía el sensor. Como se indica en la hoja de datos, al emplear la especificación NMEA 0183, los datos se reciben con la siguiente estructura:

\$GNRMC,092751.000,A,5321.6802,N,00630.3371,W,0.06,31.66,280511,,A*45

Con el fin de garantizar la integridad de los datos y asegurar que al menos una trama esté completamente almacenada, el tamaño del buffer circular es del doble del tamaño de la trama.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

Las funciones implementadas, verifican que ha llegado la trama correctamente, según la especificación NMEA 0183, y la guardan en un buffer para ser tratada. A continuación, se encargan de extraer los datos útiles de latitud y longitud, los cuales serán transmitidos como valores enteros al multiplicarlos por mil. Esta metodología asegura que los datos relevantes sean obtenidos de manera precisa y lista para ser utilizados en el sistema.

```
#define LEN_GPS      8*80*2 //Tamaño del buffer que almacena datos del GPS

//Variables GPS - Buffer circular
char GPS;    //GPS
char latitud[20]; char longitud[20];
char latitudSinPunto[sizeof(latitud) - 1];
char longitudSinPunto[sizeof(longitud) - 1];
uint32_t lat, lon;
char buffer_GPS [LEN_GPS]; //Buffer que lee los datos del buffer circular

//FUNCIONES GESTIÓN TRAMA GPS-----
-----

int shiftCircular(char* trama, int desplazamientos) {
    int longitudTrama = strlen(trama);
    int i, j;
    char temp;

    for (i = 0; i < desplazamientos; i++) {
        temp = trama[0];
        for (j = 0; j < longitudTrama - 1; j++) {
            trama[j] = trama[j + 1];
        }
        trama[longitudTrama - 1] = temp;
    }

    return 0;
}

int ordenarTramaCircular(char* trama) {
    const char* gnrmc = strstr(trama, "$GNRMC");

    if (gnrmc == NULL) return -1;

    int desplazamientos = gnrmc - trama;
    shiftCircular(trama, desplazamientos);

    return 0;
}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
int obtenerLatitudLongitud(const char* trama, char* latitud, char* longitud) {
    char* copiaTrama = strdup(trama);

    if (copiaTrama == NULL) return -1;

    char* token = strtok(copiaTrama, ",");
    int count = 0;
    while (token != NULL) {
        count++;
        if (count == 4) {
            strcpy(latitud, token);
        } else if (count == 6) {
            strcpy(longitud, token);
        }
        token = strtok(NULL, ",");
    }
    free(copiaTrama);

    return 0;
}

void quitarPunto(const char* valor, char* valorSinPunto) {
    int i, j = 0;
    int tamano = strlen(valor);

    for (i = 0; i < tamano; i++) {
        if (valor[i] != '.') {
            valorSinPunto[j] = valor[i];
            j++;
        }
    }
    valorSinPunto[j] = '\0';
}

int read_gps(void){

    memset(&buffer_GPS,0,sizeof(buffer_GPS));

    int empty = fifo1_empty();

    int i_GPS = 0;
    while(!empty){
        buffer_GPS[i_GPS]=fifo1_get();
        i_GPS++;
        if(i_GPS>LEN_GPS) i_GPS = 0;
        empty = fifo1_empty();
    }
}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
lat = 99999999;
lon = 99999999;
if(ordenarTramaCircular(buffer_GPS)==0)
    if(obtenerLatitudLongitud(buffer_GPS,latitud,longitud)==0){
        quitarPunto(latitud, latitudSinPunto);
        quitarPunto(longitud, longitudSinPunto);
        lat = atoi(latitudSinPunto);
        lon = atoi(longitudSinPunto);
    }

// Se almacena en las estructuras
sensores07.gps.lat=lat;
sensores08.gps.lat=lat;
sensores09.gps.lat=lat;
sensores10.gps.lat=lat;
sensores11.gps.lat=lat;

sensores07.gps.lon=lon;
sensores08.gps.lon=lon;
sensores09.gps.lon=lon;
sensores10.gps.lon=lon;
sensores11.gps.lon=lon;

return 0;
}
```

• ESTACIÓN METEOROLÓGICA

Debido al formato de los datos de la estación meteorológica, mostrado en el [apéndice I](#), se implementa una función que realiza la conversión de los datos en formato hexadecimal a bits, facilitando así su manipulación. En la hoja de cálculo de la estación meteorológica, se puede observar que los bits están dispuestos de manera desarreglada.

Al igual que en otros sensores, la trama de datos recibida consta de una sección de datos de configuración o estado y otra de datos útiles. Se han desarrollado funciones específicas para asegurar la fiabilidad de la trama recibida, verificando que los datos de configuración sean recibidos correctamente.

Asimismo, se ha implementado una función para extraer la parte de datos útiles de la trama recibida, con el propósito de obtener de manera organizada los datos de las magnitudes medidas. Para lograrlo, se seleccionan los bits pertinentes de la sección útil de la trama, asegurando una obtención precisa y estructurada de los valores de las magnitudes medidas.

Es necesario destacar que para la transmisión de datos se ha comprimido toda la trama haciendo uso de cada bit para transmitir información útil, de esta forma se reduce el

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

tamaño de la trama enviada para conseguir reducir el consumo de energía, descongestionar el sistema y hacerlo más eficiente.

```

#define LEN_METEOR 25      //Tamaño del buffer que almacena datos de la
estacion meteorologica
#define ID 36    //Identificador estacion meteorologica
#define SC 86    //Secuencia seguridad estacion meteorologica

//Variables estacion metereologico - buffer circular
char meteor;                                //ESTACION METEOROLOGICA
char buffer_meteor[LEN_METEOR];    //Buffer que lee los datos del buffer
circular

//FUNCIONES GESTION TRAMA ESTACION METEOROLOGICA-----
-----

char* convertirABinario(const char* numeros, int longitud) {
    if (longitud <= 0) {
        return NULL;
    }

    int i, j;
    int longitudTotal = longitud * 8; // Longitud total de la cadena
binaria
    char* binario = (char*)malloc((longitudTotal + 1) * sizeof(char));
    if (binario == NULL) {
        return NULL;
    }

    int indice = 0; // Índice en la cadena binaria
    for (i = 0; i < longitud; i++) {
        int num = numeros[i];
        for (j = 7; j >= 0; j--) {
            int bit = (num >> j) & 1;
            binario[indice] = bit ? '1' : '0';
            indice++;
        }
    }
    binario[longitudTotal] = '\0';

    return binario;
}

int verificarIDSC(char* numeros, int longitud) {
    int ID_RX = numeros[0];
    int SC_RX = numeros[1];

    // Verificar desplazamiento circular
    int desplazamiento = 0;
    for (int i = 0; i < longitud; i++) {

```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
if (numeros[i] == ID && numeros[(i + 1) % longitud] == SC) {
    desplazamiento = i;
    break;
}

// Realizar el desplazamiento circular
if (desplazamiento > 0) {
    for (int i = 0; i < desplazamiento; i++) {
        int temp = numeros[0];
        for (int j = 0; j < longitud - 1; j++) {
            numeros[j] = numeros[j + 1];
        }
        numeros[longitud - 1] = temp;
    }
}

// Verificar el ID y el Código de Seguridad
if (ID_RX == ID && SC_RX == SC){
    return 0;
} else {
    return -1;
}

unsigned int obtenerBits(const char* binario, int posicion, int longitud)
{
    unsigned int bits = 0;
    for (int i = 0; i < longitud; i++) { bits = (bits << 1) |
(binario[posicion + i] - '0'); }
    return bits;
}

int read_meteor(void){

    memset(&buffer_meteor,0,sizeof(buffer_meteor));

    int empty = fifo2_empty();

    int i_meteor = 0;
    while(!empty){
        buffer_meteor[i_meteor]=fifo2_get();
        i_meteor++;
        if(i_meteor>LEN_METEOR) i_meteor=0;
        empty = fifo2_empty();
    }

    int longitud = sizeof(buffer_meteor) / sizeof(buffer_meteor[0]);
}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
if(verificarIDSC(buffer_meteor, longitud)) return -1;

char* binario = convertirABinario(buffer_meteor, longitud);

u_int32_t wd      = obtenerBits(binario, 16, 8);
u_int32_t wd_e   = obtenerBits(binario, 24, 1);
u_int32_t lb     = obtenerBits(binario, 28, 1);
u_int32_t t_e    = obtenerBits(binario, 29, 1);
u_int32_t t      = obtenerBits(binario, 30, 10);
u_int32_t hum_e = obtenerBits(binario, 40, 1);
u_int32_t hum   = obtenerBits(binario, 41, 7);
u_int32_t ws    = obtenerBits(binario, 27, 1) * 512 +
obtenerBits(binario, 48, 8);
u_int32_t gs_e  = obtenerBits(binario, 56, 1);
u_int32_t gs    = obtenerBits(binario, 57, 7);
u_int32_t rf    = obtenerBits(binario, 64, 16);
u_int32_t uvi_e = obtenerBits(binario, 80, 1);
u_int32_t uvi   = obtenerBits(binario, 81, 15);
u_int32_t l_e   = obtenerBits(binario, 96, 1);
u_int32_t l     = obtenerBits(binario, 97, 23);
u_int32_t p     = obtenerBits(binario, 130, 6);

// Se almacena en las estructuras
sensores07.est_met.wd=wd;
sensores07.est_met.wd_e=wd_e;
sensores07.est_met.lb=lb;
sensores07.est_met.t_e=t_e;
sensores07.est_met.t=t;
sensores07.est_met.hum_e=hum_e;
sensores07.est_met.hum=hum;
sensores07.est_met.ws=ws;
sensores07.est_met.gs_e=gs_e;
sensores07.est_met.gs=gs;
sensores07.est_met.rf=rf;
sensores07.est_met.uvi_e=uvi_e;
sensores07.est_met.uvi=uvi;
sensores07.est_met.l_e=l_e;
sensores07.est_met.l=l;
sensores07.est_met.p=p;

if (binario != NULL) { free(binario); }

return 0;
}
```

• SENSECAP

En este sensor, es esencial modificar la dirección del dispositivo si se desea utilizar múltiples sensores simultáneamente con las mismas entradas. Esta característica de

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

direcciónamiento es sumamente valiosa, ya que facilita la integración de varios sensores en una única línea de comunicación.

Para llevar a cabo el cambio de dirección del sensor, se requiere emplear una de las siguientes tramas por la línea de comandos:

Dirección 2

01 10 02 00 00 01 04 00 02 E4 50 (El 02 de la antepenúltima pareja se cambia por la dirección que se quiera y el CRC [E4 50] hay que recalcularlo)

Dirección 3

01 10 02 00 00 01 04 00 03 25 90

Dirección 4

01 10 02 00 00 01 04 00 04 64 52

Una vez se le ha pasado al sensor alguna de estas tramas, se ha de reiniciarlo. Para verificar que se ha cambiado exitosamente la dirección, se debe observar por el terminal que devuelve una lista de hexadecimales a los siguientes comandos:

Dirección 2

02 03 02 00 00 01 85 81

Dirección 3

03 03 02 00 00 01 84 50

Dirección 4

04 03 02 00 00 01 85 E7

Cuando el sensor se ha configurado con la dirección adecuada, es crucial enviar un comando a través de la trama para solicitar al sensor correspondiente que proporcione los valores de las seis magnitudes en una respuesta.

La trama de respuesta mencionada previamente contiene tanto datos de configuración o estado como datos útiles. Los datos de configuración son útiles para verificar que el mensaje recibido está bien formulado y no contiene errores. Sin embargo, con el propósito de evitar saturar el sistema, únicamente se transmitirán los datos útiles.

En relación con lo descrito anteriormente, las funciones implementadas en el código están directamente vinculadas a estas tareas. Se ha creado una función para calcular el *checksum* mediante un código de redundancia cíclica, lo que garantiza la integridad de los datos. Asimismo, se genera la trama que será enviada al sensor para verificar que la trama recibida esté correctamente formulada y, por último, se seleccionan los datos útiles para ser transmitidos de manera eficiente. Estas funciones son esenciales para asegurar un flujo de datos confiable y preciso en el sistema de monitorización.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
extern UART_HandleTypeDef huart6;

#define LEN_MSG 8
#define LEN_DATA 17

#define MAX_SENSORES 3
#define INSTRUCCION_MEDICION_SIZE 5

char instruccionMedicion[INSTRUCCION_MEDICION_SIZE] = {0x04, 0x00, 0x00,
0x00, 0x06}; // Instrucción Sensecap
char buffer_sensecap [LEN_DATA];

int RS485Ready;

typedef struct {
    uint8_t numSensor;
} SensorData;

void sendRS485(char *mensaje){
    HAL_GPIO_WritePin(TX_RS485_GPIO_Port, TX_RS485_Pin,
GPIO_PIN_SET);      // Pull DE high to enable TX operation
    HAL_UART_Transmit(&huart6, (uint8_t *)mensaje, LEN_MSG , 1000);
    HAL_GPIO_WritePin(TX_RS485_GPIO_Port, TX_RS485_Pin,
GPIO_PIN_RESET);    // Pull RE Low to enable RX operation
}

// Función calcular CRC16
uint16_t calc_crc16(char *snd, int num) {
    int i, j;
    uint16_t c, crc = 0xFFFF;
    for (i = 0; i < num; i++) {
        c = snd[i] & 0x00FF;
        crc ^= c;
        for (j = 0; j < 8; j++) {
            if (crc & 0x0001) {
                crc >>= 1;
                crc ^= 0xA001;
            }
            else  crc >>= 1;
        }
    }
    return crc;
}

// Función generar la trama que será la instrucción enviada al sensor
void generar_trama(SensorData *sensor, char *trama) {
    trama[0] = sensor->numSensor;
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
for (int i = 0; i < INSTRUCCION_MEDICION_SIZE; i++) {
    trama[i + 1] = instruccionMedicion[i];
}

uint16_t crc = calc_crc16(trama, INSTRUCCION_MEDICION_SIZE + 1);

trama[INSTRUCCION_MEDICION_SIZE + 1] = (char)(crc & 0xFF);
trama[INSTRUCCION_MEDICION_SIZE + 2] = (char)((crc >> 8) & 0xFF);
}

// Función que verifica los datos recibidos por el sensor
int verificar_datos(char *trama, char *buffer_sense, int
buffer_sense_len) {
    if (trama[0] != buffer_sense[0] && trama[1] != buffer_sense[1]) { // Se comprueba que el ID y el modo es igual al enviado en la instrucción
        return -1;
    }

    char buffer_crc[buffer_sense_len - 2];
    for (int i = 0; i < buffer_sense_len - 2; i++) {
        buffer_crc[i] = buffer_sense[i];
    }

    uint16_t crc = calc_crc16(buffer_crc, sizeof(buffer_crc)); // Calcula el CRC del buffer_crc
    if (buffer_sense[buffer_sense_len - 2] != (crc & 0xFF) &&
    buffer_sense[buffer_sense_len - 1] != ((crc >> 8) & 0xFF)) { // Ver por que no va
        return -1;
    }

    int length = buffer_crc[2];
    if (sizeof(buffer_crc) - 3 != length) { // -> -3 (Se quita el ID,
modo y el indicador de tamaño)
        return -1;
    }
    return 0;
}

// Función para seleccionar un entero relacionado con una medición
uint16_t seleccionar_enteros_payload( const char *payload, int inicio,
int fin) {
    uint16_t resultado = 0;
    for (int i = inicio; i <= fin; i++) { resultado = (resultado << 8) |
payload[i]; }
    return resultado;
}

int read_sensecap() {
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
memset(&buffer_sensecap,0,LEN_DATA);

RS485Ready = 0;

SensorData sensores[MAX_SENSORES];
uint16_t t, vwc, ec, sal, tds, eps;

for(int i = 0; i < MAX_SENSORES; i++){
    sensores[i].numSensor = i+1;
}

for (int i = 1; i <= MAX_SENSORES; i++) {
    char trama[INSTRUCCION_MEDICION_SIZE + 3]; // Array de
(INSTRUCCION_MEDICION_SIZE + 3) elementos de 8 bits (el número del
sensor, la instrucción y el CRC)

    generar_trama(&sensores[i], trama);

    sendRS485(trama);

    HAL_Delay(100);

    char payload[buffer_sensecap[2]];
    if(verificar_datos(trama, buffer_sensecap,
sizeof(buffer_sensecap))) { // Existe algún error
        t = 0xFFFF;
        vwc = 0xFFFF;
        ec = 0xFFFF;
        sal = 0xFFFF;
        tds = 0xFFFF;
        eps = 0xFFFF;
    }

    else{
        // Almacenar en payload la carga útil
        for (int i = 0; i < buffer_sensecap[2]; i++) {
            payload[i] = buffer_sensecap[i + 3];
        }

        t = seleccionar_enteros_payload(payload, 0, 1);
        vwc = seleccionar_enteros_payload(payload, 2, 3);
        ec = seleccionar_enteros_payload(payload, 4, 5);
        sal = seleccionar_enteros_payload(payload, 6, 7);
        tds = seleccionar_enteros_payload(payload, 8, 9);
        eps = seleccionar_enteros_payload(payload, 10, 11);

        sensores08.sense[i].t=t;
        sensores08.sense[i].vwc=vwc;
        sensores08.sense[i].ec=ec;
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
    sensores08.sense[i].sal=sal;
    sensores08.sense[i].tds=tds;
    sensores08.sense[i].eps=eps;
    sensores09.sense[i].t=t;
    sensores09.sense[i].vwc=vwc;
    sensores09.sense[i].ec=ec;
    sensores09.sense[i].sal=sal;
    sensores09.sense[i].tds=tds;
    sensores09.sense[i].eps=eps;
}

return 0;
}
```

- **DS18B20**

Se comunica mediante el protocolo en serie *I-Wire*. Para respetar los tiempos de retardo del sensor, detallados en su hoja de características, y obtener el dato de la temperatura con 12 bits de resolución, el máximo, se utilizan las funciones dadas por Dragino.

El dato que se lee del sensor es el mismo que se transmite posteriormente, sin necesidad de procesarlo.

```
//https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf
```

```
void reset_ds18b20(void){
    if(mode!=11) return;

    delay_us_dwt_init();           //Delay para cumplir tiempos del
datasheet
    DS18B20_Reset (mode);         //Reset
}

void read_ds18b20(void){
    if(mode != 11) return;

    int tmp18b20=63488;           //Valor de error
    tmp18b20 = DS18b20_temp();    //Tomar el dato de temperatura

    sensores11.ds18b20=tmp18b20; //Guardar dato estructura

}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

- **SHT20**

Este sensor se comunica mediante el protocolo I2C. Se emplean los comandos proporcionados en el catálogo del sensor, para obtener los datos de temperatura y humedad, así como para hacer un “soft reset” del sensor.

Los datos que se leen del sensor son los mismos que se transmiten posteriormente, sin necesidad de procesarlos.

```
#define SHT2x_I2C_ADDR          0x40
#define SHT2x_HOLD_MASTER         1
#define SHT2x_READ_TEMP_HOLD      0xe3
#define SHT2x_READ_RH_HOLD        0xe5
#define SHT2x_READ_TEMP_NOHOLD    0xf3
#define SHT2x_READ_RH_NOHOLD      0xf5
#define SHT2x_WRITE_REG           0xe6
#define SHT2x_READ_REG            0xe7
#define SHT2x_SOFT_RESET          0xfe
#define SHT2x_TIMEOUT              1000

typedef enum SHT2x_Resolution {
    RES_14_12 = 0x00,
    RES_12_8 = 0x01,
    RES_13_10 = 0x80,
    RES_11_11 = 0x81,
} SHT2x_Resolution;

extern I2C_HandleTypeDef hi2c1;

void reset_sht20(void){
    uint8_t cmd = SHT2x_SOFT_RESET;
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
}

uint16_t SHT2x_GetRaw(uint8_t cmd) {
    uint8_t val[3] = { 0 };
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
    HAL_I2C_Master_Receive (&hi2c1, SHT2x_I2C_ADDR << 1, val, 3,
SHT2x_TIMEOUT);
    return val[0] << 8 | val[1];
}

uint8_t SHT2x_ReadUserReg(void) {
    uint8_t val;
    uint8_t cmd = SHT2x_READ_REG;
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
HAL_I2C_Master_Receive (&hi2c1, SHT2x_I2C_ADDR << 1, &val, 1,
SHT2x_TIMEOUT);
return val;
}

void SHT2x_SetResolution(SHT2x_Resolution res ) {
    uint8_t val = SHT2x_ReadUserReg();
    val = (val & 0x7e) | res;
    uint8_t temp[2] = { SHT2x_WRITE_REG, val };
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, temp, 2,
SHT2x_TIMEOUT);
}

void read_sht20(void){
    for (int i=0;i<4;i++){
        if (i == 0 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_1_Pin ,
GPIO_PIN_SET); //PB12 en STM32F411
        if (i == 1 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_2_Pin ,
GPIO_PIN_SET); //PB13 en STM32F411
        if (i == 2 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_3_Pin ,
GPIO_PIN_SET); //PB14 en STM32F411
        if (i == 3 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_4_Pin ,
GPIO_PIN_SET); //PB15 en STM32F411
        SHT2x_SetResolution(RES_14_12);
        uint16_t temp, hum = 0;
        int hold = 1; //Holding mode, 0 for no hold master, 1 for hold
master.
        char cmd_temp = (hold ? SHT2x_READ_TEMP_HOLD :
SHT2x_READ_TEMP_NOHOLD);
        temp = SHT2x_GetRaw(cmd_temp);
        uint8_t cmd_hum = (hold ? SHT2x_READ_RH_HOLD :
SHT2x_READ_RH_NOHOLD);
        hum = SHT2x_GetRaw(cmd_hum);
        sensores08.sht20.t = temp;
        sensores08.sht20.hum = hum;
        sensores10.sht20[i].t = temp;
        sensores10.sht20[i].hum = hum;
        sensores13.sht20.t = temp;
        sensores13.sht20.hum = hum;

        if (i == 0 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_1_Pin ,
GPIO_PIN_RESET);
        if (i == 1 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_2_Pin ,
GPIO_PIN_RESET);
        if (i == 2 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_3_Pin ,
GPIO_PIN_RESET);
        if (i == 3 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_4_Pin ,
GPIO_PIN_RESET);
    }
}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

- **Contador de pulsos**

El contador de pulsos es un sensor altamente variable, cuyo funcionamiento dependerá del tipo de contador utilizado. A la hora de procesar adecuadamente la información de este sensor, se utilizan los flancos de subida para contar los pulsos.

Los datos transmitidos comprenden la cantidad de pulsos contados, el tiempo de medición, así como el tiempo máximo y mínimo de los pulsos capturados. Para calcular estos últimos valores se emplea la función *HAL_GetTick()*, que proporciona el tiempo de uno de los relojes internos del μC , en milisegundos.

Asimismo, se ha incorporado una funcionalidad de filtrado con el propósito de eliminar posibles ruidos y rebotes que puedan surgir del sensor. Este filtro se modifica mediante la variable filtro, que indica el valor de pulso que no se contabilizará, ya que al ser más pequeño que el mínimo establecido, se identifica como un error. Para eliminar el filtro, se establece un valor de 0, el cual es el valor predeterminado.

Es importante destacar que el filtro actualmente implementado utiliza la función HAL mencionada anteriormente, que opera en milisegundos. Sin embargo, es factible que haya pulsos aún más cortos que no puedan ser filtrados de manera óptima con esta configuración. Aunque se podría emplear un contador para mejorar el filtrado, permitiendo tomar muestras de microsegundos, esto supondría un mayor consumo de energía, poniendo en riesgo la vida útil de la batería del sistema. Por lo tanto, se ha optado por un enfoque que busca un equilibrio entre eficiencia y precisión en la medición de pulsos.

```
#define CONN 3

int contador[CONN] = {0};
uint32_t previousMillis[CONN] = {0};
uint32_t currentMillis [CONN] = {0};
int periodo[CONN] = {0};

int filtro = 2;
int periodoMINval[CONN] = {99999,99999,99999};
int periodoMAXval[CONN] = {0};

void reset_con(void){
    for(int k=0;k!=CONN;k++) contador[k]=0;
}

void read_con(void){
    //Incluir los datos en las estructuras
    //reset_con();
}
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
void interrupt_pa11(void){
    int i = 0;
    currentMillis[i] = HAL_GetTick();
    periodo[i] = currentMillis[i] - previousMillis[i];
    if(periodo[i] > filtro){
        contador[i]++;
        previousMillis[i] = currentMillis[i];
        if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
        if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
    }
    sensores08.contador.cnt      = contador;
    sensores08.contador.time     = contador;
    sensores08.contador.maxp     = periodoMAXval[i];
    sensores08.contador.minp     = periodoMINval[i];
    sensores09.contador.cnt      = contador;
    sensores09.contador.time     = contador;
    sensores09.contador.maxp     = periodoMAXval[i];
    sensores09.contador.minp     = periodoMINval[i];
    sensores10.contador.cnt      = contador;
    sensores10.contador.time     = contador;
    sensores10.contador.maxp     = periodoMAXval[i];
    sensores10.contador.minp     = periodoMINval[i];
    sensores11.contador.cnt      = contador;
    sensores11.contador.time     = contador;
    sensores11.contador.maxp     = periodoMAXval[i];
    sensores11.contador.minp     = periodoMINval[i];
    sensores12.contador[i].cnt   = contador;
    sensores12.contador[i].time  = contador;
    sensores12.contador[i].maxp  = periodoMAXval[i];
    sensores12.contador[i].minp  = periodoMINval[i];
}

void interrupt_pa12(void){
    int i = 1;
    currentMillis[i] = HAL_GetTick();
    periodo[i] = currentMillis[i] - previousMillis[i];
    if(periodo[i] > filtro){
        contador[i]++;
        previousMillis[i] = currentMillis[i];
        if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
        if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
    }
    sensores12.contador[i].cnt   = contador;
    sensores12.contador[i].time  = contador;
    sensores12.contador[i].maxp  = periodoMAXval[i];
    sensores12.contador[i].minp  = periodoMINval[i];
}

void interrupt_pb12(void){
    int i = 2;
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
currentMillis[i] = HAL_GetTick();
periodo[i] = currentMillis[i] - previousMillis[i];
if(periodo[i] > filtro){
    contador[i]++;
    previousMillis[i] = currentMillis[i];
    if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
    if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
}
sensores12.contador[i].cnt = contador;
sensores12.contador[i].time = contador;
sensores12.contador[i].maxp = periodoMAXval[i];
sensores12.contador[i].minp = periodoMINval[i];
}
```

- **ADC**

Se utilizan varios canales del ADC, cada uno asignado a un pin distinto, los valores que se lean por estos canales son lo que se transmitirán.

```
#define NUM_CHANNEL 4

extern ADC_HandleTypeDef hadc1;

int ADC_VAL[NUM_CHANNEL] = {0};

int read_ADC(void){

    memset(&ADC_VAL,0,sizeof(ADC_VAL));
    for (int i=0; i<NUM_CHANNEL; i++){
        ADC_ChannelConfTypeDef sConfig = {0};
        if (i == 0 ) sConfig.Channel = ADC_CHANNEL_0;
        if (i == 1 ) sConfig.Channel = ADC_CHANNEL_1;
        if (i == 2 ) sConfig.Channel = ADC_CHANNEL_2;
        if (i == 3 ) sConfig.Channel = ADC_CHANNEL_3;
        sConfig.Rank = 1;
        sConfig.SamplingTime = ADC_SAMPLETIME_15CYCLES; //480??
        if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK) return -1;
        HAL_ADC_Start(&hadc1);
        HAL_Delay(20);
        ADC_VAL[i] = HAL_ADC_GetValue(&hadc1);
        HAL_ADC_Stop(&hadc1);
    }
    sensores13.adc0=ADC_VAL[0];
    sensores13.adc1=ADC_VAL[1];
    sensores13.adc2=ADC_VAL[2];
    sensores13.adc3=ADC_VAL[3];
    return 0;
}
```

5.4. Almacenamiento de los datos

El ámbito de este proyecto solamente se centra en la adquisición de los datos de los nodos, su pequeño preprocesamiento y almacenamiento dentro del propio nodo. Sin embargo, la política ideada para este nodo es enviar los datos sin almacenarlos, solamente se almacenan en unas estructuras durante un breve periodo de tiempo para guardar los datos del modo de funcionamiento.

```
//Estructuras
// gps
struct gps{
    int32_t lat:28;      // --> latitud
    int32_t lon:28;      // --> longitud
};

// contador de pulsos
struct contador{
    uint32_t cnt:28;     // --> pulsos contados
    uint32_t time:28;    // --> tiempo de medida
    uint32_t maxp:28;   // --> tamaño máximo de pulso
    uint32_t minp:28;   // --> tamaño mínimo de pulso
};

// sensor sensecap
struct sensecap{
    uint16_t t:14;        // --> temperatura del sensor sensecap
    uint16_t vwc:14;      // --> vwc del sensor sensecap
    uint16_t ec:15;       // --> ec del sensor sensecap
    uint16_t sal:15;      // --> salinidad del sensor sensecap
    uint16_t tds:15;      // --> tds del sensor sensecap
    uint16_t eps:14;      // --> epsilon del sensor sensecap
};

// sensor sht20
struct sht20{
    int16_t t:14;          // --> temperatura del sensor sht20
    uint16_t hum:14;       // --> humedad del sensor sht20
};

struct est_met{
    uint32_t wd:9;         // --> dirección de viento
    uint32_t wd_e:1;        // --> bit error dirección de viento
    uint32_t lb:1;          // --> indicador de baja batería
    uint32_t t:11;          // --> temperatura ambiente
    uint32_t t_e:1;         // --> bit error temperatura ambiente
    uint32_t hum:8;         // --> humedad ambiente
    uint32_t hum_e:1;        // --> bit error humedad ambiente
    uint32_t ws:9;          // --> velocidad del viento
    uint32_t gs:8;          // --> velocidad de ráfaga
};
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```
uint32_t gs_e:1;      // --> bit error velocidad de ráfaga
uint32_t rf:12;       // --> lluvia
uint32_t uvi:16;      // --> índice ultravioleta
uint32_t uvi_e:1;     // --> índice ultravioleta
uint32_t l:24;        // --> cantidad de luz
uint32_t l_e:1;       // --> bit error cantidad de luz
uint32_t p:6;         // --> presión
};

struct _sensores07{
    struct est_met est_met;
    struct gps gps;
};

struct _sensores08{
    struct gps gps;
    struct contador contador;
    struct sensecap sense[3];
    struct sht20 sht20;
//uint32_t pluviometro:20;
    uint16_t free:3;
};

struct _sensores09{
    struct gps gps;
    struct contador cont;
    struct sensecap sense[3];
    uint16_t free:3;
};

struct _sensores10{
    struct gps gps;
    struct contador cont;
    struct sht20 sht20[4];
};

struct _sensores11{
    struct gps gps;
    struct contador cont;
    uint16_t ds18b20:16;
};

struct _sensores12{
    struct contador cont[3];
};

struct _sensores13{
    struct sht20 sht20;
// adc
    uint16_t adc0:12;      // --> 1º adc
```

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

```

    uint16_t adc1:12;           // --> 2º adc
    uint16_t adc2:12;           // --> 3º adc
    uint16_t adc3:12;           // --> 4º adc
    uint16_t free:4;
};
```

A pesar de que esto no forme parte del alcance del proyecto, se brindará una breve explicación sobre el método de transmisión de datos.

La siguiente tabla expone el orden secuencial en que los datos serán transmitidos por el nodo. Es perceptible que se enviará una cabecera al inicio, seguida por los datos de los sensores. El tamaño de este paquete de datos variará según el modo en que se encuentre operando, es decir, en función de los sensores que estén involucrados en el modo. Por último, se encuentra una cola conformada por la marca temporal, cerrando así el conjunto de información a ser procesada.

TAMAÑO Y CONTENIDO DE LAS TRAMAS								17/agosto/2023
MODO	7	8	9	10	11	12	13	
Device ID	64	64	64	64	64	64	64	
Ver	16	16	16	16	16	16	16	
BAT	16	16	16	16	16	16	16	
Signal Strength	8	8	8	8	8	8	8	
MOD	8	8	8	8	8	8	8	
gps	56	56	56	56	56			
est_met	104							
cont_1		112	112	112	112	112		
cont_2						112		
cont_3						112		
sensecap_1		87	87				Opcional	
sensecap_2		87	87					
sensecap_3		87	87					
sht20_1		32		32			32	
sht20_2				32				
sht20_3				32				
sht20_4				32				
ds18b20					16			
adc1						12		
adc2						12		
adc3						12		
adc4						12		
fill_zeros		7	3				4	
Timestamp	32	32	32	32	32	32	32	
Total/bits	304	612	576	440	328	480	228	
Total/Bytes	38,0	76,5	72,0	55,0	41,0	60,0	28,5	

Tabla 5.4.1 Contenido de la trama que se va a transmitir

5.5. Alimentación del sistema

A diferencia del primer nodo implementado, este segundo nodo está diseñado para hacer uso de energía renovable para mantenerse a lo largo del tiempo. Como se expuso con anterioridad, se hará uso de un sistema de alimentación formado por una batería y una placa fotovoltaica.

El nodo de Dragino viene incluido con una batería de Li/SOCl₂ (figura 5.5.1), está dará la potencia suficiente para alimentar al sistema durante el tiempo que no reciba energía de la placa solar. Esta placa es la que se muestra en la figura 5.5.2 de 5V, 60 mA y cuyas dimensiones de 68 x 36 mm son ideales para anclarlas a la cara de la carcasa protectora del nodo.



Figura 5.5.1 Batería utilizada para alimentar al segundo nodo (76)



Figura 5.5.2 Panel fotovoltaico utilizado para alimentar al segundo nodo (77)

La idea es tener un sistema que sea alimentado por la noche a través de la batería y el panel fotovoltaico se encargue de suministrar suficiente potencia durante el día para recargar la batería y alimentar a los sensores durante su proceso de medición.

Además, de que el circuito está diseñado para encender los sensores, tomar medidas durante unos pocos segundos (30 segundos en el código) y después esté dormido el sistema hasta que pasen varios minutos o, incluso, horas.

IMPLEMENTACIÓN DEL SEGUNDO NODO SENSOR

Modelo	Magnitud	Consumo (mW)
Sensecap	EC, salinidad...	144
DS18B20	Tº y humedad suelo	5-7,5
SHT20	Tº y humedad aire	< 1.188
NEO-6M	GPS	360
YF-S201	Caudal	75
SENO313	Distancia	≤ 75
Sainlogic WS 3500 Plus	Estación meteorológica	Indefinido

Tabla 5.5.1 Consumo de los elementos del sistema

Con todo lo contado y comprobando la potencia que consumen los sensores, se puede concluir que este sistema logra gran autonomía. Además, como se muestra en la siguiente figura, la placa fotovoltaica se puede integrar perfectamente en la carcasa.



Figura 5.5.3 Integración del panel fotovoltaico a la carcasa del dispositivo de Dragino

6. Integración y resultados

En este apartado, se abordan distintos aspectos relevantes para la correcta instalación del sistema en el entorno previsto, así como los pasos necesarios para inicializar el sistema a fin de recoger los datos de forma adecuada y efectiva. Se expondrán los datos tomados durante unas semanas, junto con un somero análisis de estos.

6.1. Entorno de trabajo e instalación del sistema

A continuación, se va a desarrollar el proceso de instalación de los sensores en el entorno, se expondrá cómo se inicializa el sistema y se describirá dónde se almacenan los datos registrados para cada uno de los nodos.

6.1.1. Primer nodo implementado

Teniendo ya el primer nodo implementado correctamente, será necesario instalar los distintos sensores para la toma de datos.

- **PYTHOS-31:** Este se trata de un sensor de humedad con forma de hoja, que se ha de situar entre el follaje de la encina, para adquirir el valor de la humedad que tiene las hojas del ejemplar. Para su correcta colocación sería necesario hacer uso de algún sistema de sujeción que evitará los movimientos bruscos, en este caso se utilizará una mordaza.
- **SEN0322:** Este sensor de oxígeno se situará, al igual que el sensor anterior, cerca del follaje de la encina, debido a que se quiere medir el O_2 que producen las hojas. Para integrarlo correctamente en los ejemplares también se ha de utilizar un sistema de sujeción, este sensor lo facilita al contar con cuatro orificios por los que se les pasa la mordaza.
- **MH-Z19:** El sensor de CO_2 es muy similar a los sensores anteriores, se ha de situar entre el follaje del ejemplar, al querer medir correctamente la producción de CO_2 de las hojas. Para integrarlo se hizo uso también de una mordaza, pero este sensor sería más complicado de integrar en un ejemplar de mayor tamaño.
- **SEN0249:** Este sensor al medir pH, estará situado a pie del árbol, para tomar mediciones de forma rápida (como se ha hecho en este proyecto) simplemente hay que clavar la punta del sensor en la tierra. En cambio, si se quisiesen realizar mediciones en largos períodos de tiempo, se recomienda enterrar completamente al sensor dejando solo a la vista el cable.

Un factor importante a tener en cuenta es que este sensor viene con su punta protegida en una disolución contenida en un recipiente, es importante girar y tirar hacia abajo del recipiente para intentar conservar la disolución. Posteriormente, se ha de desenroscar una anilla que está enganchada al sensor (es difícil de encontrarla a simple vista) y se introducirá la punta de metal que ayuda a romper el suelo y proteger al sensor. Finalmente, se vuelve a enroscar la anilla con la

INTEGRACIÓN Y RESULTADOS

punta de acero ya puesta, todo este procedimiento viene evidenciado en las siguientes imágenes.



Figura 6.1.1 Montaje previo a la implantación del sensor SEN0249

En la siguiente imagen se muestra la colocación de los distintos sensores en ambos ejemplares de *quercus ilex* (encinas).



Figura 6.1.2 Colocación de los sensores en los dos ejemplares de encinas

INTEGRACIÓN Y RESULTADOS

Una vez integrados los sensores en el entorno, se procede a indicar cómo se han de inicializar los dispositivos de adquisición y preprocesamiento para que tomen los datos de forma correcta.

Primero, se ha de cargar en el Arduino Uno con la herramienta Arduino IDE el código implementado mostrado en el [apéndice E](#). Una vez introducido, cada vez que se encienda el Arduino automáticamente ejecutará el código.

En la Raspberry Pi 4 se ha de acceder a la carpeta donde se haya guardado el código implementado que se muestra en el [apéndice F](#) y ejecutarlo por terminal de comandos con el comando `python3 "NombreCodigo".py`. Como aclaración, sería menester instalar la librería de Python en la Raspberry si no se estuviera previamente instalada, al no venir instalada de serie.

Con estos códigos en funcionamiento, la Raspberry Pi genera de forma automática, en la misma carpeta donde esté guardado el código, un fichero JSON cuyo nombre es el mes y el año de la toma de datos. En su interior estarán guardados el registro mensual de los datos tomados por los sensores.

6.1.2. Segundo nodo implementado

Al igual que ocurría con el primer nodo, se va a comentar como se han de instalar los distintos sensores en el entorno de estudio.

- **DS18B20:** Este sensor se encarga de medir la temperatura que hay en el suelo. Por tanto, para instalarlo hay que clavarlo en el suelo, si se quiere dejar un largo periodo de tiempo tomando medidas se aconseja enterrarlo varios centímetros debajo del suelo.
- **Sensecap:** Se encarga de medir diversas magnitudes relevantes para conocer la composición y el estado del suelo a pie del árbol. Para tomar medidas rápidas se recomienda clavar las agujas del tridente en el suelo. Si se quisiera una instalación permanente de este sensor se ha de enterrar completamente.
- **SHT20:** Este sensor se encarga de medir la temperatura y humedad que hay en el aire, para recoger correctamente los valores de las hojas habría que colocarlo entre el follaje haciendo uso de algún sistema de sujetación.
- **GPS:** Este sensor solamente ha de medir la posición del nodo. Para instalarlo correctamente solamente ha de tener acceso a los satélites de posicionamiento, es decir, no ha de estar cubierto este sensor.
- **Estación meteorológica:** Debe situarse en un lugar abierto, pero cercano al ejemplar que se esté midiendo, todo esto para conocer los factores ambientales que le influyen. A la estación se le ha de conectar la veleta en el soporte metálico mellado y el anemómetro se instala en el soporte metálico que no está mellado dispuesto para este menester. Por último, el pluviómetro se ha de acoplar ajustando las pestañas en los huecos destinados (apretándolo) y se gira para fijarlo. Además, la estación tiene un soporte para acoplarla a un mástil y se ha de

direccionar el norte en la dirección que indica la muesca grabada. Todo esto se muestra en las siguientes figuras.

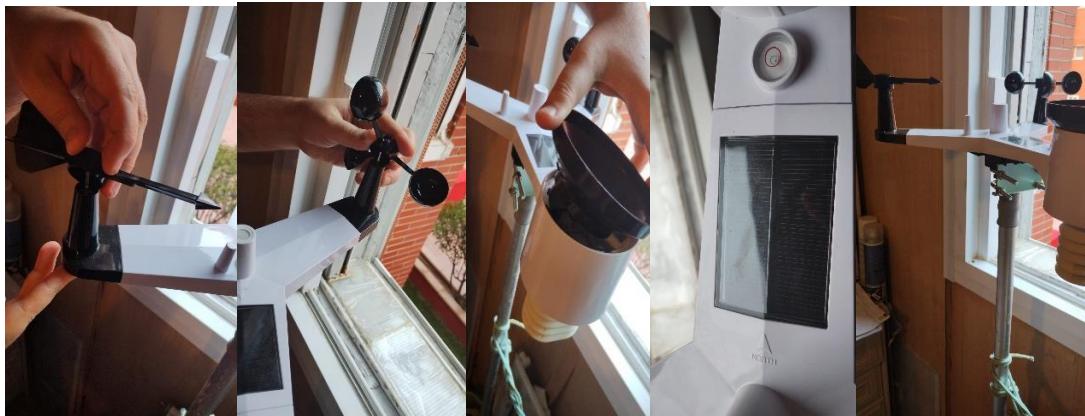


Figura 6.1.3 Montaje de la estación meteorológica

- **Flujómetro:** Este sensor sería utilizado en el caso de que se instale un sistema de riego, esto sería útil para ver cuanta cantidad de agua se le suministra al árbol en estudio. Debe estar instalado en el interior de la tubería, para que funcione correctamente el sensor se ha de direccionar el flujo de agua con la flecha que tiene en la carcasa exterior.

Una vez estudiada la forma de integrar los sensores en el entorno de medición, el siguiente paso sería ver el procedimiento para cargar un fichero, este es algo más complejo que en el nodo anterior:

- Lo primero es compilar el código, esto producirá un fichero hexadecimal.
- Se ha de conectar el dispositivo con el ordenador que contiene el fichero a través del módulo TTL y poner el interruptor blanco en ISP.
- Se abre el programa *STM32CubeProgrammer* y se enciende con el jumper el nodo de Dragino.
- Con la herramienta *STM32CubeProgrammer* se ha de conectar al dispositivo y acceder a la pestaña para subir el código.
- Tras esperar a que se suba el programa, se pone el *switch* blanco en FLASH y con la herramienta *Serial Port Utility* se comprueba que el dispositivo esté funcionando.

Como se ha explicado, el nodo no almacena los datos directamente, solamente se guardan en las estructuras vistas previamente, esperando a que se transmitan los datos.

El funcionamiento propio del nodo se muestra en el siguiente código, primero se alimentan los sensores y se reinician sus valores. Posteriormente, se toman los datos durante 30 segundos y se apagan los sensores.

```

void sensores_loop(){

    power_on();
    reset_con();
    //HAL_TIM_Base_Start_IT(&htim4);    //Inicia el timer de 30 segundos
    while(flag30s == 0) {_WFI();}      //Respetar los tiempos de inicio
de los sensores
    reset_sht20();
    reset_ds18b20();
    read_gps();
    read_meteor();
    read_ADC();
    read_sht20();
    read_ds18b20();
    read_sensecap();
    read_con();
    power_off();
    HAL_TIM_Base_Stop_IT(&htim4); //Para el timer de 30 segundos
    flag30s = 0;

}

```

6.2. Resultados obtenidos

En este apartado se van a examinar los distintos datos obtenidos de cada ejemplar de encina y se procederá a realizar una breve evaluación gráfica de los datos adquiridos.

Se permitió que el sistema tomara datos a lo largo de un amplio período de tiempo como para poder analizar el rendimiento del sistema y poder estudiar adecuadamente a los ejemplares de encina. Se recuerda que los datos no se tomaron de ejemplares en medio de un entorno forestal, sino que se recolectaron en una zona cerrada.

Originalmente, el sistema está concebido para adquirir datos cada varios minutos o incluso horas. Sin embargo, debido a limitaciones temporales, se ajustó el modelo para realizar medidas cada diez segundos y, de esta manera, se obtienen una mayor cantidad de datos para analizar.

Durante el tiempo limitado de este trabajo no se han podido realizar mediciones de las magnitudes en las encinas con el segundo nodo implementado. Por ende, no ha sido posible llevar a cabo un análisis que refleje de forma completa la comparativa entre los datos medidos por sendos nodos.

A continuación, se procede a analizar los datos de los ejemplares con los de otros estudios previos. Este análisis se realizará para cada una de las magnitudes medidas por el primer nodo implementado.

- **Humedad de la hoja**

Para empezar, se estudia la humedad de la hoja. Los estudios realizados se hacen sobre ejemplares en entornos abiertos, esto explica que por la noche la hoja se mantiene más húmeda a causa del rocío y se va secando a lo largo del día como se muestran en la siguiente gráfica. Es importante señalar que este estudio de Nathan Emery (78) puede alertar de posibles enfermedades de los ejemplares.

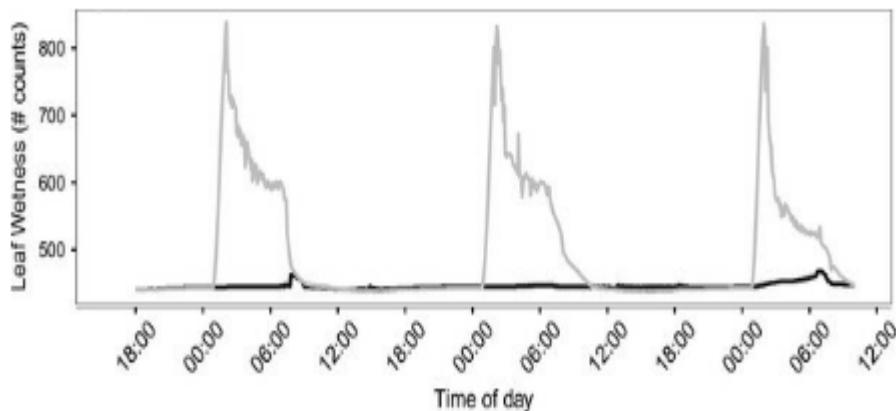


Figura 6.2.1 Evolución de la humedad de las hojas

En la anterior gráfica se observa adecuadamente lo explicado, la humedad empieza a aumentar a partir de las doce de la madrugada y va descendiendo progresivamente hasta las primeras horas de la mañana.

En las siguientes gráficas se observan los datos obtenidos para ambos ejemplares de encinas:



Figura 6.2.2 Datos de la humedad de hoja de la encina saludable durante el tiempo de medición

INTEGRACIÓN Y RESULTADOS



Figura 6.2.3 Datos de la humedad de hoja de la encina saludable durante un día

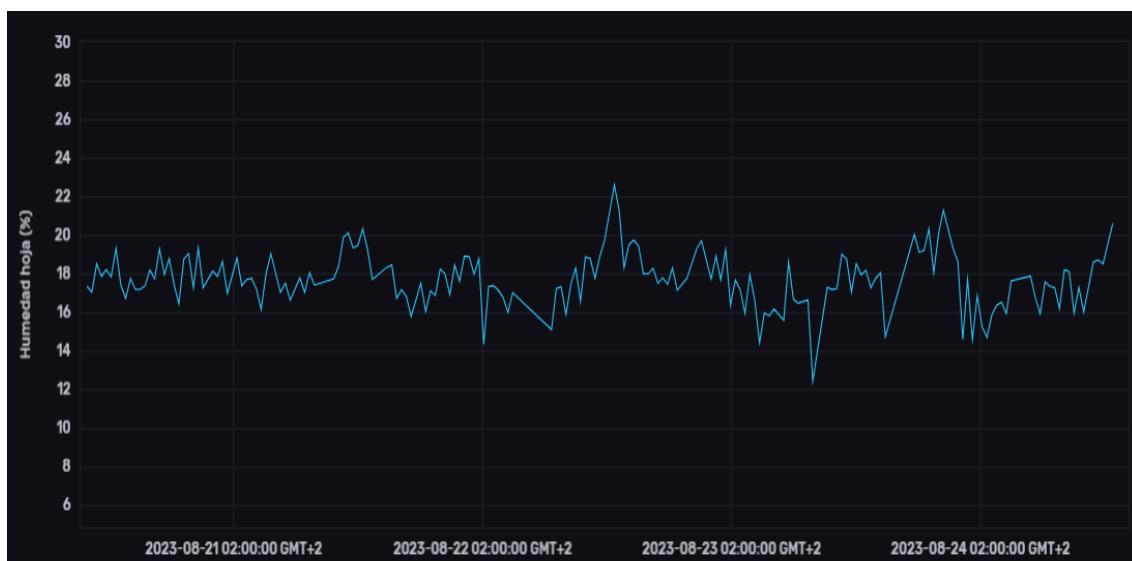


Figura 6.2.4 Datos de la humedad de hoja de la encina seca durante todo el periodo de medición



Figura 6.2.5 Datos de la humedad de hoja de la encina seca durante un día

Se puede observar que el sensor es muy ruidoso, pero se mantiene estable en el rango de medida. Los datos se tomaron durante una ola de calor, podría ser que la humedad no variase tanto como lo esperado debido a este motivo.

También es destacable que los rangos de humedad medidos en la encina saludable, cuando no hay valores atípicos, se mueve entre el 13 % y el 20 %. Mientras que en la encina seca se observan unos rangos de humedad que oscilan entre el 14 % y el 22 %.

- **Oxígeno de la hoja**

A diferencia de la magnitud anterior, este tipo de sensores no están especializados en la medición del O_2 producido por las hojas, si no que se sitúa cerca del follaje para obtener los datos más próximos a la producción realista de O_2 de los ejemplares.

Tampoco se encontraron trabajos donde se estudie el O_2 producido por las hojas de ejemplares de quercus a lo largo del tiempo. En este caso se expondrán los datos comparándolos con la cantidad de O_2 ambiente tomados por estaciones meteorológicas.



Figura 6.2.6 Datos del O_2 de hoja de la encina saludable durante todo el periodo de medición



Figura 6.2.7 Datos del O_2 de hoja de la encina saludable durante un día



Figura 6.2.8 Datos del O_2 humedad de hoja de la encina seca durante todo el periodo de medición



Figura 6.2.9 Datos del O_2 humedad de hoja de la encina seca durante un día

Se puede observar que en ambos casos se mantiene bastante estable, quitando un par de valores atípicos. En la encina saludable los valores de O_2 oscilan entre el 20% y el 21 %. En cambio, para la encina seca estos valores oscilan entre el 19,5 % y 21,3 %.

- **Dióxido de carbono de la hoja**

El dióxido de carbono es una magnitud variable a lo largo del día, principalmente por la fotosíntesis que realiza todo árbol. Los ejemplares producirán más CO_2 por la noche que por la mañana ya que es cuando respira, pero también se ha de recalcar que las mediciones se hacen en un entorno urbano. Por lo cual, la producción de CO_2 de coches y otras máquinas podrían contrarrestar la observación de este fenómeno.

INTEGRACIÓN Y RESULTADOS

Un ejemplo de cómo sería la evolución del CO_2 a lo largo del día en función de si se toma en una ciudad o en una zona rural es la que se muestra a continuación.

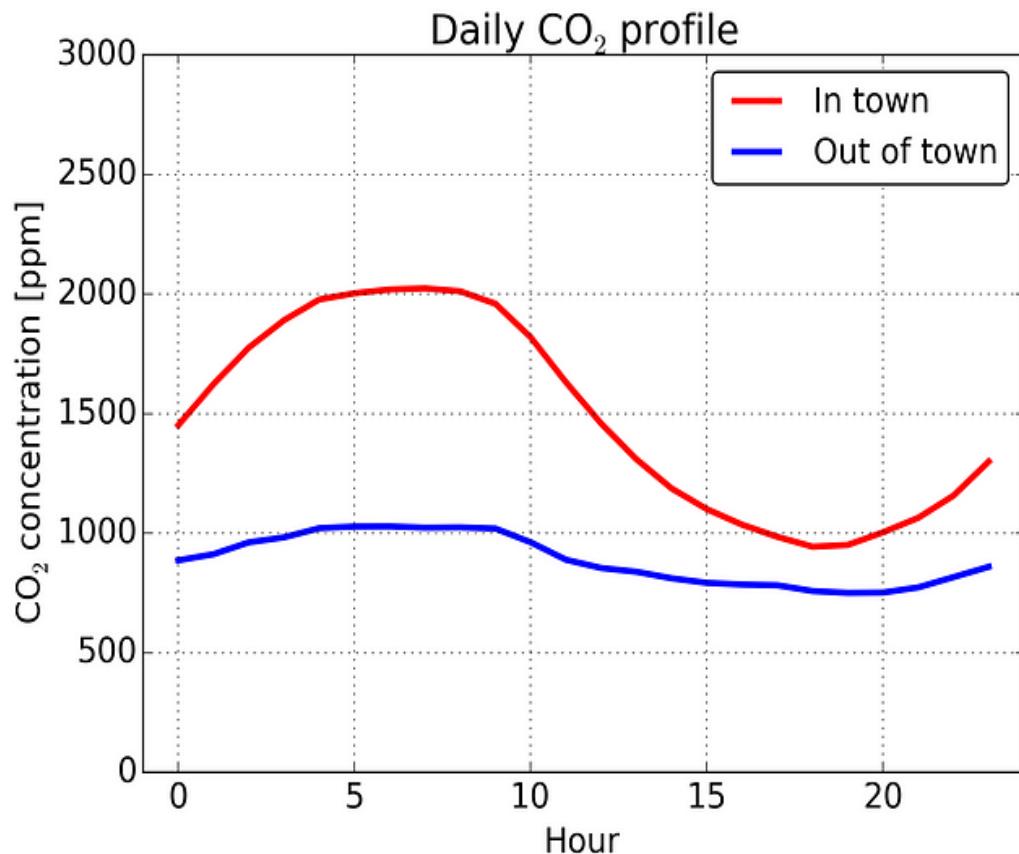


Figura 6.2.10 Evolución del CO_2 del aire en un día en una ciudad y en una zona rural (79)



Figura 6.2.11 Evolución del CO_2 de la encina saludable durante todo el periodo de medición

INTEGRACIÓN Y RESULTADOS

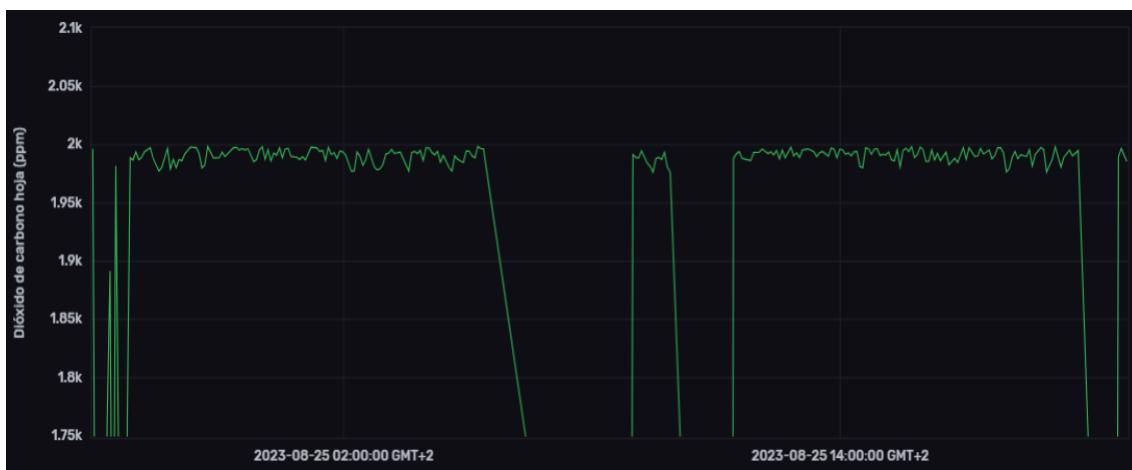


Figura 6.2.12 Evolución del CO_2 de la encina saludable durante un día

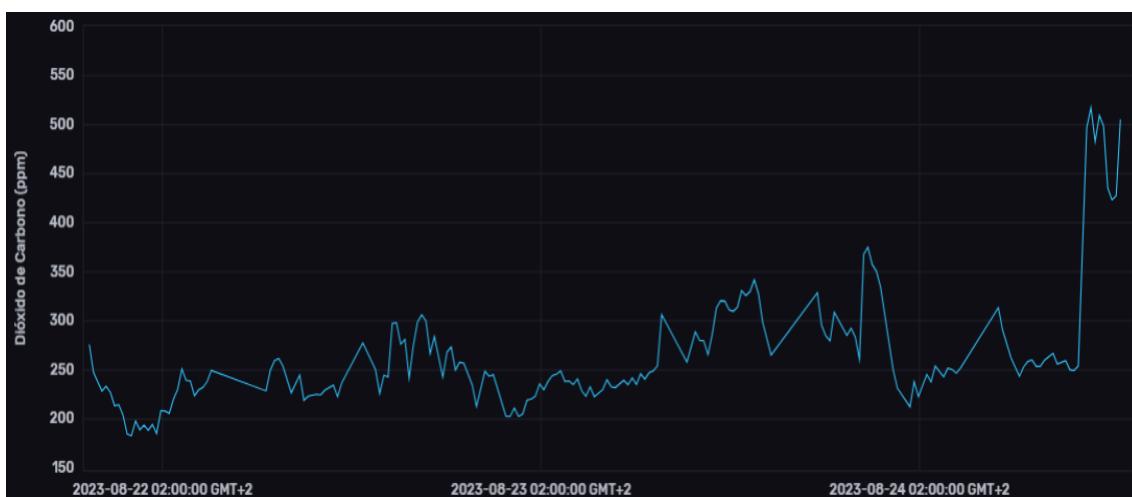


Figura 6.2.13 Evolución del CO_2 de la encina seca durante todo el periodo de medición



Figura 6.2.14 Evolución del CO_2 de la encina seca durante un día

Se puede observar que el CO_2 aumenta por el día y disminuye por la noche. Esto último, a pesar de ser contrario a la fotosíntesis, tiene sentido porque las tomas se tomaron en un entorno urbano y de mucho tráfico, por tanto, el CO_2 producido por los coches interfiere con las medidas tomadas por el sistema. Esto reafirma la idea de que sería necesario realizar estas medidas en entornos naturales, al ser posible, libres de contaminación directa.

- **pH del suelo**

La evolución del pH es más simple de analizar, realmente los valores han de oscilar entorno a los 5,0 y los 7,0 para que este tipo de árboles se desarrollen de forma adecuada. Hay que tener en cuenta que los suelos se pueden dividir en:

- Suelos neutros: Cuentan con un pH entre 6,8 y 7,2.
- Suelos ácidos: Cuentan con un pH inferior a 6,8. Pudiendo ser:
 - o Suelos ligeramente ácidos: Los que cuentan con un pH entre 6,5 y 6,8.
 - o Suelos ácidos: Los que cuentan con un pH entre 5,5 y 6,5.
 - o Suelos muy ácidos: Los que cuentan con un pH entre 4,5 y 5,5.
 - o Suelos extremadamente ácidos: Los que cuentan con un pH inferior a 4,5.

Se puede evidenciar que las encinas sobreviven adecuadamente entre terrenos muy ácidos y neutros. La siguiente figura muestra la evolución de un suelo típico controlado (rombos) y otro quemado (cuadrados). (80)

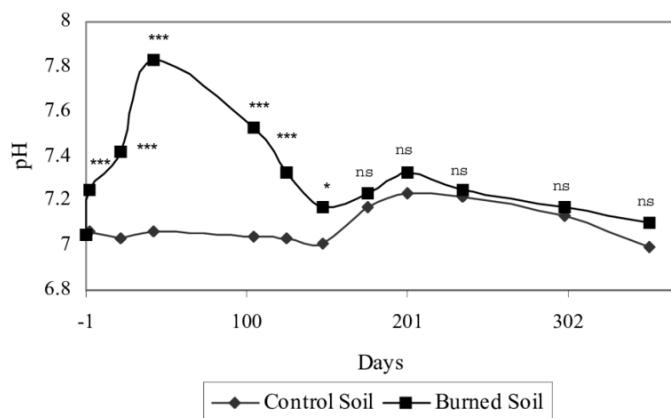


Figura 6.2.15 Evolución del pH del suelo en una zona controlada y en otra quemada (81)

A continuación, se muestra la evolución del pH para las encinas en estudio, hay que tener en cuenta que están en un tiesto en un entorno cerrado.



Figura 6.2.16 Evolución del pH de la encina saludable durante todo el periodo de medición

INTEGRACIÓN Y RESULTADOS



Figura 6.2.17 Evolución del pH de la encina saludable durante un día

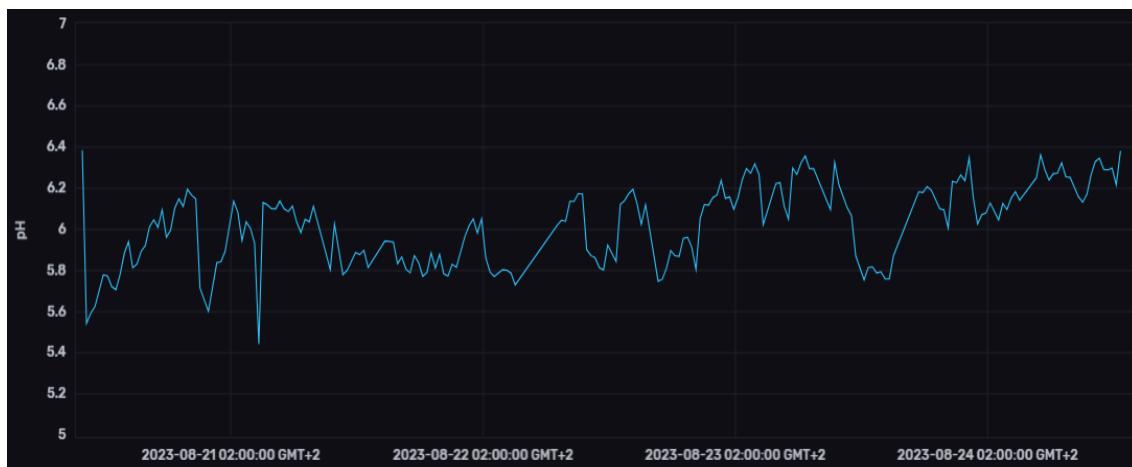


Figura 6.2.18 Evolución del pH de la encina seca durante todo el periodo de medición

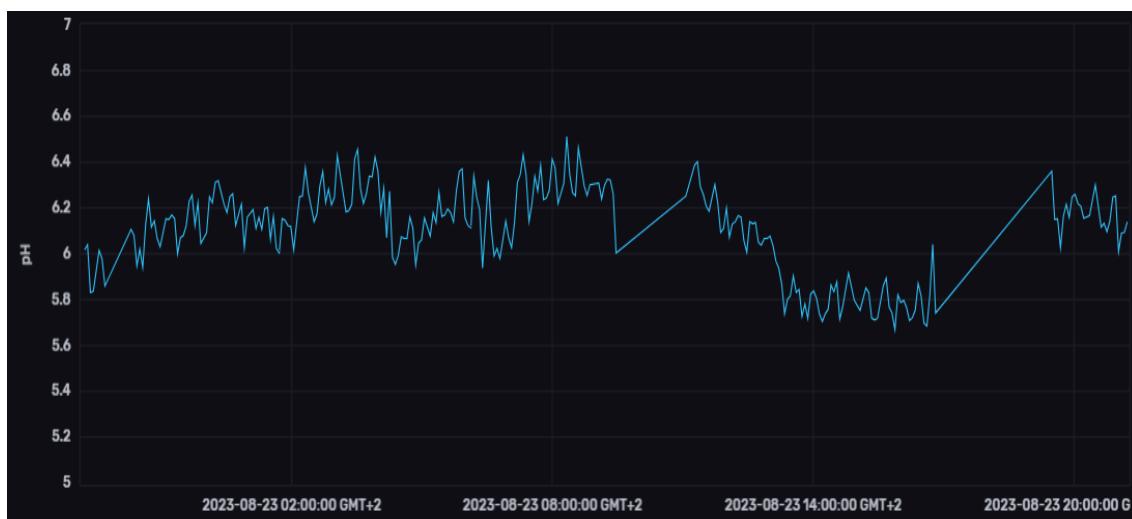


Figura 6.2.19 Evolución del pH de la encina seca durante un día

En este caso, se puede apreciar que el pH de ambos ejemplares se mantiene parcialmente estático a lo largo del tiempo (si no se toman en cuenta valores atípicos) como se esperaba.

Solamente se muestra que el suelo de la encina saludable tiene un pH ligeramente superior al de la encina seca. Mientras que la encina sana su pH oscila entre 6,6 y 7,8, la encina seca oscila entre 5,6 y 6,5.

7. Presupuestos

7.1. Gastos asociados al proyecto

Para completar este apartado, se van a añadir los presupuestos del sistema completo, tomando en cuenta ambos nodos implementados. Se procede a desglosar los distintos gastos producidos por la implementación del proyecto. Se llevará a cabo un análisis donde se recojan los costes de los elementos que conforman al sistema junto con otro tipo de gastos asociados.

Es fundamental, en un principio, tener en cuenta los gastos asociados a los sensores utilizados para la implementación de ambos nodos.

Cantidad	Modelo	IVA	Precio unidad	Precio total
1	LAT-B3	21%	225,00 €	272,25 €
1	Sensecap	21%	81,25 €	98,31 €
1	PYTHOS-31	21%	180,00 €	217,80 €
1	SEN0322	21%	63,99 €	77,43 €
1	MH-Z19C	21%	45,00 €	54,45 €
1	Soil NPK Sensor	21%	52,55 €	63,59 €
1	SEN0249	21%	116,85 €	141,39 €
1	314990620	21%	79,74 €	96,49 €
2	Sainlogic WS 3500 Plus. Estación Meteorológica	21%	185,00 €	447,70 €
1	DS18B20	21%	3,49 €	4,22 €
1	SHT20	21%	8,23 €	9,96 €
1	NEO-6M	21%	18,16 €	21,97 €
1	YF-S201	21%	10,99 €	13,30 €
1	SEN0313	21%	41,75 €	50,52 €
Total				1569,38 €

Tabla 7.1.1 Gastos asociados a los sensores de sendos nodos implementados

También es importante reflejar el coste del resto de elementos que conforman al sistema, teniendo en cuenta tanto el sistema de alimentación como las placas de circuito impreso junto con todos los elementos electrónicos necesarios.

INTEGRACIÓN Y RESULTADOS

Cantidad	Modelo	IVA	Precio unidad	Precio total con descuentos
1	Batería Gens Ace	21 %	114,88 €	139,00 €
1	Batería ES-261520/W	21%	17,00 €	20,57 €
1	Placa solar	21%	0,38 €	0,46 €
5	MMPCBAv1.0	21%	0,38 €	2,30 €
5	MMPCBSv1.0	21%	0,38 €	2,30 €
5	MMPCBhv1.0	21%	0,38 €	2,30 €
2	Raspberry Pi 4	21%	80,00 €	193,60 €
1	Arduino UNO	21%	24,00 €	29,04 €
1	NBSN95	21%	44,80 €	54,21 €
1	LSN50	21 %	49,50 €	59,90 €
5	MAX485	21 %	1,16 €	7,00 €
3	Cables	21%	2,46 €	8,93 €
1	Circuito impreso	21%	1,88 €	2,27 €
10	Amplificador Operacional OP292GSZ-REELCT-ND	21%	6,20 €	55,69 €
15	Condensador 1uF 1276-6470-1-ND	21%	0,10 €	1,11 €
15	Condensador 15 uF 399-C1210C156K8PAC7800CT-ND	21%	0,82 €	9,35 €
10	Resistencia 1 LT5400BCMS8E-1#PBF-ND	21%	9,38 €	87,97 €
10	Resistencia LT5400BCMS8E-2#PBF-ND	21%	9,38 €	87,97 €
10	Resistencia 505-LT5400BCMS8E-6#PBF-ND	21%	9,38 €	87,97 €
15	Convertidor 1470-VR05S05-ND	21%	2,50 €	23,48 €
10	Resistencia LT5400BHMS8E-7#PBF	21%	11,12 €	101,11 €
10	Buffer LM358DR2G	21%	0,42 €	3,40 €
5	Convertidor K7805-3AR3	21%	4,27 €	21,08 €
5	Convertidor R-739.0P	21%	27,11 €	132,38 €
10	Regulador LM317LID	21%	0,88 €	7,85 €
10	Resistencia RG3216P-1000-B-T1	21%	0,56 €	3,98 €
10	PMOS FDN338P	21%	0,41 €	3,50 €
10	Regulador LM1085ISX-5.0/NOPB	21%	1,90 €	17,06 €
10	Resistencia FDV303N	21%	0,39 €	2,91 €
10	NMOS TSR 1-2490	21%	8,48 €	83,52 €
20	Conversor 450LLE22MEFC12.5X25	21%	1,74 €	28,14 €
50	Condensador 22 uF EKXF451ELL100MJ20S	21%	1,00 €	34,27 €
10	Condensador 10 uF C0603X103J4HAC7867	21%	0,18 €	1,18 €
10	Condensador 0,1 uF CL10B104MB8NNNC	21%	0,09 €	0,24 €
TOTAL				1.213,13

Tabla 7.1.2 Gastos generales del primer nodo

INTEGRACIÓN Y RESULTADOS

$$\text{Total materias primas} = \text{Sensores} + \text{Componentes} = 1.569,38 + 1.213,13 = 2.782,51\text{€}$$

En la siguiente tabla, se puede observar los gastos asociados a las encinas necesarias para probar el correcto funcionamiento del sistema.

Cantidad	Modelo	IVA	Precio unidad	Precio total con descuentos (VC)
2	Quercus Ilex (encinas)	10%	27,23 €	59,91 €
TOTAL				59,91€

Tabla 7.1.3 Costes de terrenos y bienes naturales

En la siguiente tabla se resumen los costes del transporte, tanto de los pedidos como los desplazamientos necesarios para desarrollar el proyecto.

Cantidad	Modelo	IVA	Precio unidad	Precio total con descuentos por volumen compra
1	Transportes de componentes y plantas		123,70 €	123,70 €
TOTAL				123,70 €

Tabla 7.1.4 Costes de transportes

A su vez es importante reflejar el coste del resto de elementos que conforman al sistema, teniendo en cuenta tanto el sistema de alimentación, como las placas de circuito impreso junto con todos los elementos electrónicos necesarios.

Categoría	Horas	Coste/hora	Total
Tutor proyecto	45 h	30,00 €	1.350,00 €
Ingeniero Teleco	300 h	19,00 €	5.700,00 €
TOTAL			7.050,00 €

Tabla 7.1.5 Costes en sueldos y salarios

Otro concepto, a parte de los sueldos, es la Seguridad Social que estos llevan asociados.

Categoría	Total (S.S)
S.S del Tutor proyecto	60,75 €
S.S del Ingeniero Teleco	256,50 €
Total	317,25 €

Tabla 7.1.6 Coste en Seguridad Social

INTEGRACIÓN Y RESULTADOS

En la siguiente tabla se muestra los costes de los equipos informáticos necesarios para el proyecto. Se recoge la parte proporcional arreglo al tiempo aproximado empleado en la realización del TFM.

Equipo	Precio	Vida útil (años)	Uso (meses)	Coste
PC	1.900,00 €	8	9	178,13 €
Ordenador portátil	800,00 €	8	5	66,66 €
Teclado	30,00 €	3	9	7,50 €
Ratón	9,00 €	3	9	2,25 €
Total				254,54 €

Tabla 7.1.7 Costes en recursos y equipos informáticos

Se puede ver en la siguiente tabla el coste de las licencias de las herramientas de diseño y desarrollo utilizadas.

Categoría	Total
Arduino IDE	0,00 €
KiCad 7.0	0,00 €
Keil µVision 5	4.987,83 €
MATLAB	262,00 €
Draw.io	0,00 €
Office	30,00 €
Total	5.279,83€

Tabla 7.1.8 Costes de licencias de herramientas de diseño y desarrollo

Se añaden otros costes asociados del proyecto, como los gastos de suministro.

Categoría	Coste
Luz	350,00 €
Internet	351,00 €
Total	701,00 €

Tabla 7.1.9 Costes de suministros

En la siguiente tabla se recoge la parte proporcional de lo gastado en la utilización del equipo necesario para el proyecto.

INTEGRACIÓN Y RESULTADOS

Equipo	Precio	Vida útil (años)	Uso (meses)	Coste
Osciloscopio	2.500,00 €	10	1	20,83 €
Multímetro	100,00 €	10	1	0,83 €
Fuente de tensión	287,00 €	10	1	2,39 €
Soldador	50,00 €	5	1	0,83 €
Total				24,98 €

Tabla 7.1.10 Costes de equipo

Finalmente, se recogen todos los gastos vistos y se dividen en fijos y variables.

Costes fijos	Valor	Costes variables	Valor
Sueldos y salarios personal	7.050,00 €	Materias primas	2.782,51 €
Seguridad Social	317,25 €	Terrenos y bienes naturales	59,91 €
Recursos y equipos informáticos	254,54 €	Transporte	123,70 €
Licencias de herramientas de diseño y desarrollo	5.279,83 €		
Suministros	701,00 €		
Otros gastos	24,98 €		
Total costes fijos	13.627,60 €	Total costes variables	2.966,12 €

Tabla 7.1.11 Costes fijos y variables

Total costes del proyecto = 16.593,72 €

El total de lo gastado en este proyecto es 16.593,72 €. Con la realización de este estudio de los distintos costes y gastos precisados para implementar de forma adecuada los nodos de monitorización, se pretende realizar una estimación aproximada del presupuesto necesario para realizar este proyecto.

8. Conclusiones y trabajo futuro

Una vez finalizado el estudio, la implementación y el análisis del sistema de monitorización de la salud de las encinas desarrollado en este TFM, se procederá a plasmar las conclusiones derivadas del presente trabajo y se explorarán diferentes perspectivas sobre las posibles aplicaciones de este proyecto en un futuro.

8.1. Conclusiones

El objetivo de este proyecto consiste en realizar un sistema de monitorización que sea capaz de medir diferentes magnitudes que afectan a los bosques de quercíneas. Para llevar a cabo este proyecto, se han utilizado dos ejemplares de *quercus ilex* que presentan distintas condiciones: uno claramente saludable y otro, en el que se puede observar a simple vista que sus hojas están secas.

Este trabajo ha logrado desarrollar dos nodos muy distintos que son de gran utilidad para el análisis de la salud de ejemplares de distintos especímenes de encinas. A pesar de que se han desarrollado solamente para las encinas, son aplicables a otras especies vegetales, en muchos casos sin necesidad de adaptar el sistema y, en casos específicos, con una mínima modificación.

El sistema proyectado pretende mejorar la toma de datos obteniéndolos de ejemplares y del entorno en diferentes zonas del bosque; todo ello, para controlar el estado del bosque completo y favorecer la actuación en caso de necesidad y en zonas concretas.

Las medidas tomadas se han realizado sobre especímenes jóvenes confinados en una maceta, por lo que se requiere una segunda fase de desarrollo e instalación para su explotación en campo. Es un sistema escalable en número de nodos y flexible en número y tipo de sensores.

Se han conseguido los objetivos principales planteados en el ámbito de telecomunicación: se han implementado dos sistemas versátiles, económicos y escalables, generando una solución que puede simplificar un problema complejo.

En primera aproximación, se puede observar que con los datos obtenidos se pueden diferenciar los estados de salud de los dos ejemplares estudiados. Ahora bien, para su completa aplicación sería relevante observar una evolución de parámetros a lo largo de un mayor periodo de tiempo, en un entorno forestal realmente abierto y expuesto a inclemencias climáticas.

Los nodos implementados en este proyecto son utilizados en el desarrollo de una red IoT capaz de monitorizar adecuadamente el bosque. La implementación y explotación de dicha red de sensores forma parte de un segundo trabajo. En este primer trabajo se ha recogido la interacción con los sensores, comunicación entre elementos, junto con el procesamiento y almacenamiento de los datos recibidos.

CONCLUSIONES Y TRABAJO FUTURO

Con este trabajo, he logrado ahondar en el estado actual de la monitorización de los bosques y comprender la complejidad de crear y desarrollar un sistema integral de estas características. En este sentido, el proceso seguido me ha proporcionado una visión más amplia de los retos que conlleva un sistema de medida en un entorno forestal. Por otra parte, he adquirido perspectiva sobre aspectos técnicos, ambientales y logísticos.

8.2. Trabajo futuro

Para finalizar, a continuación se desarrollan brevemente algunas de las posibles alternativas y extensiones que tiene el presente proyecto.

Existen diversas formas de afrontar el tema tratado: la opción plasmada en esta memoria es una de esas múltiples posibilidades.

Algunas de las alternativas existentes están centradas en el estudio del bosque de quercus como un todo, utilizando para ello medios aéreos de observación (drones, aviones o satélites) dotados con cámaras hiperespectrales. En este caso, los datos y diagnósticos de árboles concretos con el sistema desarrollado podrían ser complementarios a la observación global. El procedimiento de correlación entre estos dos tipos diferentes de observación es actualmente una línea de trabajo en el campo de la investigación. Para tratar estos datos sería conveniente la utilización de algoritmos de inteligencia artificial.

Como se ha visto reflejado en el proyecto, solamente se ha realizado el estudio de dos ejemplares de quercus, concretamente encinas, a lo largo de un periodo reducido de tiempo. Una posible ampliación del proyecto sería aumentar el número de ejemplares en estudio, realizarlo en campo y ampliar el tiempo de observación.

Por otra parte, partiendo de los diseños realizados en este trabajo sería posible desarrollar nuevos proyectos orientados a la monitorización y control de otras clases de árboles e, incluso otros tipos de plantas, por ejemplo, en estudios agrícolas y del entorno natural.

Como se ha mencionado anteriormente, los datos obtenidos pertenecen a un periodo breve de tiempo, por lo tanto, sería aconsejable extender el tiempo de estudio de cada ejemplar de quercus, viendo su evolución con arreglo a las diferentes estaciones y su comportamiento en periodos de lluvia y sequía. Esta observación se podría complementar con información de la evolución que ha seguido esta especie en estudios previos, a fin de validar los resultados y completar el ciclo de diseño del sistema de monitorización para que sea útil a expertos.

En este trabajo se han tenido presentes las condicionantes de bajo consumo energético y autonomía asociadas a la implementación en campo y en zonas con difícil acceso. Sin embargo, otro estudio interesante sería el rediseño del sistema para disminuir el consumo energético y dotarlo de fuentes propias de energía.

Bibliografía

1. Nations, U. Hottest July ever signals ‘era of global boiling has arrived’ says UN chief | UN News. <https://news.un.org/en/story/2023/07/1139162> (accessed Aug 30, 2023).
2. Anonymous Árboles Ibéricos - Quercus. <https://www.arbolesibericos.es/genre/quercus> (accessed Sep 3, 2023).
3. Acedo, C. TAXONOMÍA DEL GÉNERO QUERCUS L. Especies presentes en la Península Ibérica. *wordpress.com* 2004.
4. Barrón, E.; Averyanova, A.; Kvacek, Z.; Momohara, A.; Pigg, K.; Popova, S.; Postigo-Mijarra, J.; Tiffney, B.; Utescher, T.; Zhou, Z. In *The Fossil History of Quercus; The Fossil History of Quercus*; 2017; pp 39-105.
5. Su, T.; Wilf, P.; Xu, H.; Zhou, Z. Miocene leaves of Elaeagnus (Elaeagnaceae) from the Qinghai-Tibet Plateau, its modern center of diversity and endemism. *American Journal of Botany* 2014, 101, 1350-1361.
6. Unknown La encina y sus adaptaciones al medio: Características de las encinas. 2013.
7. Serbal Quercus ilex (Encina, carrasca o chaparro). 2016.
8. Anais, R. Quercus ilex (Usos tradicionales) ~ Plantas - CONECT-e. <https://conecte.es>
9. Villamayor, M. Quercus, roble y encina, el árbol su madera y curiosidades. <https://www.espiraldevirutas.com/quercus> (accessed Aug 30, 2023).
10. Villaverde, J. Alcornocal, ni te imaginas lo mucho que vale este árbol. Quercus suber. <https://viforsa.es/planta-forestal/para-que-sirve-un-alcornocal-quercus-suber-sobreira/> (accessed Aug 30, 2023).
11. Pérez Ramos, I. M.; Villar, R.; Marañón, T. El fascinante mundo de los Quercus: desde la biología molecular hasta la ecología de comunidades. *ecosistema, REVISTA CIENTÍFICA DE ECOLOGÍA Y MEDIO AMBIENTE* 2014, 23, 1-4.
12. Portillo, A. Roble. Quercus robur L. (fagáceas). *Offarm* 2001, 20, 175.
13. Villanueva, E. La Encina Terrona: el ícono verde de Cáceres. <https://www.elperiodicoextremadura.com/caceres-local/2023/08/20/encina-terrona-icono-verde-caceres-91069544.html> (accessed Aug 30, 2023).
14. España, G. 4RobledalesQuercusRoburQPetraea_tcm30-138145.jpg (2922×2068). https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/4RobledalesQuercusRoburQPetraea_tcm30-138145.jpg (accessed Aug 30, 2023).

BIBLIOGRAFÍA

15. España, G. 15MelojaresQuercuspyrenaica_tcm30-138149.jpg (2922×2068).
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/15MelojaresQuercuspyrenaica_tcm30-138149.jpg (accessed Aug 30, 2023).
16. España, G. 16QuejigaresQuercusfaginea_tcm30-138150.jpg (2922×2068).
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/16QuejigaresQuercusfaginea_tcm30-138150.jpg (accessed Aug 30, 2023).
17. España, G. 17QuejigaresQuercuscanariensis_tcm30-138151.jpg (2922×2068).
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/17QuejigaresQuercuscanariensis_tcm30-138151.jpg (accessed Aug 30, 2023).
18. España, G. 18EncinaresQuercusilex_tcm30-138155.jpg (2922×2068).
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/18EncinaresQuercusilex_tcm30-138155.jpg (accessed Aug 30, 2023).
19. España, G. 19AlcornocalesQuercussuber_tcm30-138156.jpg (2922×2068).
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/servicios/banco-datos-naturaleza/19AlcornocalesQuercussuber_tcm30-138156.jpg (accessed Aug 30, 2023).
20. Caudullo, G.; Welk, E.; San-Miguel-Ayanz, J. Chorological maps for the main European woody species - ScienceDirect. *Elsevier* 2017, 12, 662-666.
21. México, G. ANTECEDENTES HISTÓRICOS.
22. Arrhenius, S. On the Influence of Carbonic Acid in the Air upon the Temperature of the Ground. *Philosophical Magazine and Journal of Science* 1896, 41, 237-276.
23. Manabe, S.; Wetherald, R. On the Distribution of Climate Change Resulting from an Increase in CO₂ Content of the Atmosphere. *Journal of Atmospheric Sciences* 1979, 37, 99-118.
24. Zhu, Z.; Piao, S.; Myneni, R. B.; Huang, M.; Zeng, Z.; Canadell, J. G.; Ciais, P.; Sitch, S.; Friedlingstein, P.; Arneth, A.; Cao, C.; Cheng, L.; Kato, E.; Koven, C.; Li, Y.; Lian, X.; Liu, Y.; Liu, R.; Mao, J.; Pan, Y.; Peng, S.; Peñuelas, J.; Poulter, B.; Pugh, T. A. M.; Stocker, B. D.; Viovy, N.; Wang, X.; Wang, Y.; Xiao, Z.; Yang, H.; Zaehle, S.; Zeng, N. Greening of the Earth and its drivers. *Nature Clim Change* 2016, 6, 791-795.
25. Anonymous INVENTARIO DE DAÑOS FORESTALES (IDF) EN ESPAÑA. *miteco.gob.es* 2019, 20.
26. Anonymous Díptico.
https://www.miteco.gob.es/content/dam/miteco/es/biodiversidad/temas/inventarios-nacionales/Res%C3%BAmen%20IDF%20202011_tcm30-154440.pdf (accessed Aug 30, 2023).

BIBLIOGRAFÍA

27. Martínez, M. ¿Sabes cuál fue el primer sensor conectado de la historia?
<https://www.nobbot.com/primer-sensor-conectado/> (accessed Sep 3, 2023).
28. Sarola, J. Sir Frederick William Herschel, descubridor de la luz infrarroja | NIRS Research. *NIRS Research* 2020.
29. Fernández, T.; Elena, T. Biografia de Samuel Pierpont Langley.
<https://www.biografiasyvidas.com/biografia/l/langley.htm> (accessed Sep 3, 2023).
30. AnonymousPepperl+Fuchs | Historia de Pepperl+Fuchs, legado de Pepperl+Fuchs.
<https://www.pepperl-fuchs.com/spain/es/518.htm> (accessed Sep 3, 2023).
31. Gregersen, E. Willard Boyle | Nobel Prize, inventor, imaging technology | Britannica. <https://www.britannica.com> 2023.
32. Custodio Ruiz, A.; Bragós Bardía, R.; Pallàs Areny, R. SENSORES INTELIGENTES:
UNA HISTORIA CON FUTURO. <https://upcommons.upc.edu> 1999.
33. de Paz Menéndez, A. DESARROLLO DE UN SISTEMA DE SENSORES PARA LA DETECCIÓN DE SUSTANCIAS PELIGROSAS, UNIVERSIDAD AUTÓNOMA DE MADRID, 2015.
34. Jesús, O. SENSORES: CLASIFICACIÓN DE LOS SENSORES. 2010.
35. Lourdes, L. Sensores. 2022.
36. Úbeda Miñarro, B. Apuntes de: Sistemas embebidos.
<https://www.um.es/documents/4874468/19345367/ssee-t01.pdf/4ea71f56-2950-4c3f-acbe-e7699e490f4e> (accessed Aug 30, 2023).
37. AnonymousSistemas empotrados > Información, Biografía, Archivo, Historia.
<https://es.wikidat.com> (accessed Aug 30, 2023).
38. Virgam ¿Qué son los sistemas embebidos?. - Tecnología, ciencia y educación. 2015.
39. seabrookewindows La Historia de los Sistemas Embebidos.
<https://www.seabrookewindows.com/JM7p4x2Ww/>.
40. Wikipedia Sistemas de monitorización y control. 2019.
41. Limited, A. El médico lleva mascarilla facial. Mascarilla médica de protección contra virus. Médico o enfermera en el hospital. Tratamiento de salud. Dibujado a mano. Dibujos animados de stickman. Fideos Imagen Vector de stock - Alamy.
<https://www.alamy.es/el-medico-lleva-mascarilla-facial-mascarilla-medica-de-proteccion-contra-virus-medico-o-enfermera-en-el-hospital-tratamiento-de-salud-dibujado-a-mano-dibujos-animados-de-stickman-fideos-image351041113.html> (accessed Sep 28, 2022).
42. 123RF Hombre Del Palillo De La Historieta Que Dibuja El Ejemplo Conceptual Del Doctor Que Sostiene Dos Píldoras. Concepto De Salud Y Abuso De Drogas.

BIBLIOGRAFÍA

- Ilustraciones Svg, Vectoriales, Clip Art Vectorizado Libre De Derechos. Image 99563366. https://es.123rf.com/photo_99563366_hombre-del-palillo-de-la-historieta-que-dibuja-el-ejemplo-conceptual-del-doctor-que-sostiene-dos-pil.html (accessed Sep 28, 2022).
43. dreamstime Coronavirus Icono Palo Figura Símbolo Conjunto Enfermo Hombre Aislado Siluetas Ilustración del Vector - Ilustración de gente, respiratorio: 174669567. <https://es.dreamstime.com/coronavirus-icono-palo-figura-símbolo-conjunto-enfermo-hombre-aislado-siluetas-fondo-blanco-image174669567> (accessed Sep 28, 2022).
44. Ventura, V. Qué es la Internet de las Cosas (IoT). 2017.
45. Lim, C. W.; Baek, W.; Jung, J.; Kim, J.; Lee, S. C. Function of ABA in Stomatal Defense against Biotic and Drought Stresses. *Int J Mol Sci* 2015, *16*, 15251-15270.
46. Daszkowska-Golec, A.; Szarejko, I. Open or close the gate – stomata action under the control of phytohormones in drought stress conditions. *frontiers* 2013, *4*, 138.
47. Lagunes-Fortiz, E.; Villanueva-Verduzco, C.; Lagunes-Fortiz, E. R.; Zamora-Macorra, E. J.; Ávila-Alistac, N.; Villanueva-Sánchez, E. La densidad de siembra en el crecimiento de la verdolaga . <https://cienciasagricolas.inifap.gob.mx/index.php/agricolas/article/download/2848/3944?inline=1#:~:text=El%20diámetro%20del%20tallo%20es,el%20viento%20o%20la%20lluvia> (accessed Aug 30, 2023).
48. Luján Basile, S. M.; Ríssola, M. G.; Chindamo, M.; Manfreda, V. T. Efectos de la hormona vegetal etileno en diversas especies vegetales | UNICEN. <https://www.unicen.edu.ar/content/efectos-de-la-hormona-vegetal-etileno-en-diversas-especies-vegetales> (accessed Aug 30, 2023).
49. AnonymousNuevo Sensor de flujo de savia IMPLEXX. <https://blog.biofisicaambiental.com/sensor-de-flujo-de-savia-implexx/> (accessed Aug 30, 2023).
50. AnonymousRespiración radicular: la importancia del oxígeno para las plantas. <https://www.moleaer.com/es/blog/respiracion-radicular> (accessed Aug 30, 2023).
51. San Diego, U. Researchers Identify Elusive Carbon Dioxide Sensor in Plants that Controls Water Loss. <https://today.ucsd.edu/story/researchers-identify-elusive-carbon-dioxide-sensor-in-plants-that-controls-water-loss> (accessed Aug 30, 2023).
52. Fernández-Getino, A. P.; Ruiz-Peinado, R.; Montero González, G.; Sánchez Palomares, O. ESTIMACIÓN DEL CO₂ FIJADO EN SUELOS Y ÁRBOLES DE LOS REBOLLARES DE QUERCUS PYRENAICA EN LA PROVINCIA DE OURENSE. *Sociedad Española de Ciencias Forestales* 2008, *25*, 179-184.
53. Ruiz Santiago, F. L.; Ruiz Velázquez, J. A.; Hernández Becerra, J. A.; García Jiménez, R.; Valadez Villareal, A. Extracción y cuantificación de clorofila en hojas comestibles del estado de Tabasco. *Investigación y Desarrollo en Ciencia y Tecnología de Alimentos* .

BIBLIOGRAFÍA

54. Castro-Luna, I.; Gavi-Reyes, F.; Peña-Cabriales, J. J.; Núñez-Escobar, R.; Etchevers-Barra, J. D. EFICIENCIA DE RECUPERACIÓN DE N Y K DE TRES FERTILIZANTES DE LENTA LIBERACIÓN. *Terra Latinoamericana* 2006, 24, 277-282.
55. 16, Factsheet | HGIC 1650S | Updated: Oct; Print, 2. |. Cambiando el pH del Suelo. <https://hgic.clemson.edu/factsheet/cambiando-el-ph-del-suelo/> (accessed Aug 30, 2023).
56. Carrasco-Ríos, L. EFECTO DE LA RADIACIÓN ULTRAVIOLETA-B EN PLANTAS. *Idesia (Arica)* 2009, 27, 59-76.
57. Nafziger, E. Corn Roots, Wet Soils, and Nitrogen. 2013.
58. Chen Lopez, J. ¿Cómo influye la humedad en la calidad de los cultivos? | PROMIX. <https://www.pthorticulture.com/es/centro-de-formacion/como-influye-la-humedad-en-la-calidad-de-los-cultivos/> (accessed Aug 30, 2023).
59. AnonymousAnexo:Récords meteorológicos en España. 2023.
60. Nostradamux Meteorología y Electrónica: PWM: Una manera sencilla de controlar un nivel de tensión. 2010.
61. Moreno Velasco, I. EL AMPLIFICADOR DE INSTRUMENTACIÓN. <http://www.unet.edu.ve> .
62. AnonymousOp-amp Varieties. <http://hyperphysics.phy-astr.gsu.edu/hbasees/Electronic/opampvar7.html#c1> (accessed Aug 30, 2023).
63. AnonymousConsumo de energía Arduino Uno versus Arduino Nano | Pi Productora.
64. Velasco, R. Esta es la energía que consume cada modelo de Raspberry Pi. <https://www.redeszone.net/2017/03/02/energia-consumida-raspberry-pi/> (accessed Aug 30, 2023).
65. Arduino® UNO R3. <https://docs.arduino.cc> , 5.
66. Raspberry Pi Raspberry Pi 4 Model B. *raspberrypi.com* 2019, 5.
67. Arduino, T. Alimentar el Arduino: La guía definitiva. 2017.
68. AnonymousMódulo Conversor TTL a RS485 MAX485.
69. AnonymousRaspberry Pi (master) Arduino (slave) I2C communication with WiringPi. 2019.
70. AnonymousSTM32L072x8 STM32L072xB STM32L072xZ. *st.com* .
71. AnonymousNB ST v1.1.sch.pdf. (accessed Aug 30, 2023).

BIBLIOGRAFÍA

72. AnonymousLoRa ST v2.0 sch.pdf .
<https://github.com/dragino/Lora/blob/master/LoRaST/v2.0/LoRa%20ST%20v2.0%20sch.pdf> (accessed Aug 30, 2023).
73. AnonymousLoRa ST Sensor Node v2.0.sch.pdf.
<https://github.com/dragino/Lora/blob/master/LSN50/v2.0/LoRa%20ST%20Sensor%20Node%20v2.0.sch.pdf> (accessed Aug 30, 2023).
74. Xiaoling NBSN95_NBSN95A NB-IoT Sensor Node User Manual - DRAGINO.
http://wiki.dragino.com/xwiki/bin/view/Main/User%20Manual%20for%20LoRaWA%20End%20Nodes/NBSN95_NBSN95A%20NB-IoT%20Sensor%20Node%20User%20Manual/ (accessed Aug 30, 2023).
75. Xiaoling LSN50 & LSN50-V2 - LoRaWAN Sensor Node User Manual - DRAGINO.
<http://wiki.dragino.com/xwiki/bin/view/Main/User%20Manual%20for%20LoRaWA%20End%20Nodes/LSN50%20%26%20LSN50-V2%20-%20LoRaWAN%20Sensor%20Node%20User%20Manual/> (accessed Aug 30, 2023).
76. AnonymousEVE Li/SOCl2 battery ES-261520/W.
<https://shop.molukas.com/es/accesorios/83-eve-lisocl2-battery-es-261520w.html> (accessed Aug 30, 2023).
77. Anonymous2.23RON 32% de DESCUENTO|Panel Solar de 5V y 60mA, minisistema Solar para baterías, cargadores de teléfonos móviles portátiles, 68x36MM, gran oferta, 1 unidad| - AliExpress.
[//es.aliexpress.com/item/1005005096477333.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp=](http://es.aliexpress.com/item/1005005096477333.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp=) (accessed Aug 30, 2023).
78. Emery, N. Foliar uptake of fog in coastal California shrub species. *Oecologia* 2016, 182.
79. Jean, J. I'm living in a carbon bubble. Literally. | by Joel Jean | Medium. 2016.
80. Masats, J. El pH del suelo para el cultivo de las plantas. <https://www.botanical-online.com/cultivo/suelo-tipos-ph> (accessed Aug 30, 2023).
81. Mataix-Solera, J.; Navarro-Pedreño, J.; Guerrero, C.; Gómez Lucas, I.; Marco, B.; Mataix, J. In *Effects of an experimental fire on soil microbial populations in a Mediterranean environment*; Man and Soil at the Third Millennium; Geoforma Ediciones: 2002; pp 1607-1614.
82. AnonymousGEA50007S45X9. <https://aeromodelismoserpa.es/gens-ace/18421-gens-ace-5000mah-259v-45c-7s1p-lipo-battery-pack-with-xt90.html> (accessed Aug 30, 2023).

Apéndice A: Listas de los sensores estudiados

En este primer apéndice se muestran algunos de los distintos sensores, placas y estaciones meteorológicas que se han buscado e investigado. Se muestra el nombre del dispositivo, el precio, el tiempo que tardarían en servirlo, algunas observaciones importantes, el contacto proporcionado por la empresa y el enlace donde se puede adquirir u observar el producto.

Temperatura estomática

Temperatura estoma					
Sensor	Precio	Tiempo	Observaciones	Contacto	Enlace
LT-1M	275 €	3-6 semanas		info@phyto-sensor.com	http://phyto-sensor.com/LT-1M#:~:text=The%20LT-1M%20sensor,is%20about%20millimeter%20in%20diameter
LT-2M				solfranc@solfra.com	http://www.solfranc.com/productos/wp-content/uploads/2014/11/lt_2m_sensor_de_temperatura_de_hoja_solfranc.es_.pdf
LT-IRM				https://edaphic.com.au/contact-us/	https://edaphic.com.au/temperature/infrared-temperature-sensor/
LS-31	295 €		DESCATALOGADO		https://www.alphaomega-electronics.com/en/agricultura/3967-ls-31-broad-leaf-temperature-sensor-for-broad-leaves-weight-2-gr.html
Ls-40	330 €				https://www.alphaomega-electronics.com/en/agricultura/3967-ls-31-broad-leaf-temperature-sensor-for-broad-leaves-weight-2-gr.html
LAT-B3	225 €	1-2 semanas	Se puede utilizar un datalogger DL18 precio 188€	info@ecomatik.de	https://ecomatik.de/en/products/leaf-sensors/leaf-temperature/

Humedad de la hoja

Humedad emitida por la hoja higrómetro

Sensor	Precio	Tiempo	Observaciones	Contacto	Enlace
HM102	230€ + IVA		s-40		http://www.gisiberica.com/Higr%F3metros%20especiales/HM102.html
Higrómetro de punto de rocío WP4C	DESCATALOGADO		DESCATALOGADO		https://www.lab-ferrer.com/higrometro-de-punto-de-rocio-wp4c/
LWS			CONTACTO PROFESOR		https://www.evvos.com/product/leaf-wetness-sensor/
RK300-04	5 Uds. =421,28€	7-9días laborables	Mínimo 5 unidades 340\$+ 78\$ gastos envió =421,28€	https://www.rikasensor.com/inquire/cart	https://www.rikasensor.com/rk300-04-leaf-wetness-sensor-humidity-sensor.html?gclid=EA1aIQobChMlrlLmx_WC-QIV6gyLCh0w2g79EAAYAiAAEgKkBvD_BwE
Leaf Wetness Sensor	140 €				https://www.darrera.com/wp/en/product/6420-leaf-wetness-sensor/
PESSL	339,12 €		sugieren el NMETOS200 con sensores temperatura,humedad,pluviometro precio 943,78€	sales@metos.at	https://metos.at/es/portfolio/leaf-wetness/
NMETOS200	943,78 €		sugerencia empresa	sales@metos.at	https://metos.at/es/portfolio/leaf-wetness/
LWS-L	192 €			https://www.campbellsci.cc/request-quote	https://www.campbellsci.cc/lws
HD3901	250 €	10-12 semanas		https://www.deltaohm.com/product/hd3901-leaf-wetness-sensor/	https://www.deltaohm.com/product/hd3901-leaf-wetness-sensor/
PYTHOS31	180,00 €		Lo lleva LabFerrer en España	https://www.metergroup.com/en/meter-environment/products/phytos-31-leaf-wetness-sensor	https://www.metergroup.com/en/meter-environment/products/phytos-31-leaf-wetness-sensor

Diámetro del tallo, peciolo y tronco

Diámetro de tallo, peciolo y tronco				
Sensor	Precio	Observaciones	Contacto	Enlace
Dendrómetros DS25	308 €			http://www.gisiberica.com/dendr%F3metro/manual%20ds21.htm
Dendrómetro DS26	529 €			http://www.gisiberica.com/dendr%F3metro/MANUAL%20DS26.htm
Dendrómetro MMM-Tech	DESCATALOGADO	DESCATALOGADO		https://www.mmm-tech.de/de/downloads?task=download.send&id=88&catid=3&m=0
DD-S	350 €			https://www.mmm-tech.de/de/downloads?task=download.send&id=88&catid=3&m=0
DF	365 €			https://www.mmm-tech.de/de/downloads?task=download.send&id=88&catid=3&m=0
DC-2	375 €			https://www.mmm-tech.de/de/downloads?task=download.send&id=88&catid=3&m=0
DL-18	425 €			https://www.mmm-tech.de/de/downloads?task=download.send&id=88&catid=3&m=0
Dendrómetro SD-5(6,10)T			https://edaphic.com.au/contact-us/	https://www.alphaomega-electronics.com/es/sensores-estaciones/1568-dendrometro-para-frutas-y-verduras-diametro-0-11-cm-df-11.html
Dendrómetro SDI12			https://edaphic.com.au/contact-us/	https://www.implexx.io/wp-content/uploads/2021/05/Impléxx-SD-Stem-Dendrometer-Manual.pdf
LVDT Solartron Metrology, recorrido ±25mm	492,37 €			https://es.rs-online.com/web/p/sensores-de-control-de-movimiento/7271329
LVDT Solartron Metrology, recorrido ±100mm	1.058 €			https://es.rs-online.com/web/p/sensores-de-control-de-movimiento/7271357
LVDT Analog Devices, recorrido ±2.5 → ±50.8mm	147,31 €			https://es.rs-online.com/web/p/sensores-de-control-de-movimiento/8031591

Tamaño de fruto

Tamaño de fruto		
Sensor	Precio	Enlace
Dendrómetro DS50	497 €	http://www.gisiberica.com/dend%F3metro/DS50.html
Dendrómetro DS50A	481 €	http://gisiberica.com/CODIGOS/DS50.HTM

Sensores de Humedad

Clorofila				
Sensor	Precio	Observaciones	Contacto	Enlace
Cyclops-7 de Turner Designs	3.000€-5.000€		https://www.ott.com/es-es/productos/sensores-179/sensor-de-clorofila-a-347/	https://www.ott.com/es-es/productos/sensores-179/sensor-de-clorofila-a-347/
MC-100	2.385 €			https://www.alphamega-electronics.com/es/concentracion-clorofila/1365-mc-100-medidor-de-concentracion-de-clorofila-mol-m-2-con-gps-interno.html
Micro PAM	4 cabezales 15.675€	4 cabezales sin sistema de Adquisición	https://www.walz.com/support/contact_company.html	https://www.walz.com/products/chl_p700/micro-pam/introduction.html
Micro PAM	3cabazales 21.340€ 4cabezas 24.490€	versión en línea 3cabazales17830€ 4 cabezas 21665€	https://www.walz.com/support/contact_company.html	https://www.walz.com/products/chl_p700/micro-pam/introduction.html

Flujo de savia

Flujo de savia					
Sensor	Precio	Tiempo	Observaciones	Contacto	Enlace
Sensor de Flujo de Savia SFM1				https://www.lab-ferrer.com/contactar/	http://www.lapacacr.com/images/ICT/Sensor-Flujo-Savia-SFM-Espanol.pdf
Sensor de Flujo de Savia HRM				https://www.lab-ferrer.com/contactar/	https://www.lab-ferrer.com/sensor-de-flujo-de-savia-sfm1-y-hrm/#:~:text=El%20sensor%20SFM1%20es%20capaz,calor%20atendiendo%20al%20m%C3%A9todo%20HRM
SF-G Sensor Flujo de Savia Ecomatik (2 agujas)	339 €				https://www.alphaomega-electronics.com/es/saviaflujo-de-savia/815-sf-g-sensor-flujo-de-savia-ecomatik-2-agujas.html
East30 Sensors				https://www.lab-ferrer.com/contactar/	https://www.east30sensors.com
SGEX-16	436 €		CABLES DE EXTENSIÓN 15 m precio 200€ + batería y placa solar sin precio	https://www.idelsur.com/?page_id=1889 (gema@lab-ferrer.com)	https://www.idelsur.com/?cat=269
SGEX-19	442 €		CABLES DE EXTENSIÓN 22 m precio 250€ + batería y placa solar sin precio	https://www.idelsur.com/?page_id=1889 (gema@lab-ferrer.com)	https://www.idelsur.com/?cat=269
SGEX-25	455 €		CABLES DE EXTENSIÓN 28 m precio 300€ + batería y placa solar sin precio	https://www.idelsur.com/?page_id=1889 (gema@lab-ferrer.com)	https://www.idelsur.com/?cat=269
Sensores de Flujo de Savia EXO-Skin FLOW32A-1K	9.800 €		TODO INCLUIDO	https://www.idelsur.com/?page_id=1889 (gema@lab-ferrer.com)	https://www.idelsur.com/?cat=269
SF-L	580 €				https://www.alphaomega-electronics.com/es/sensores-y-sondas/816-sf-l-sensor-de-flujo-de-savia-ecomatik-4-agujas.html
FRUITION	ERROR	ERROR	ERROR Envío solicitud	https://fruitionsciences.com/es/sap-flow-irrigation-sensors	https://fruitionsciences.com/es/sap-flow-irrigation-sensors
HFD (HFD8-50)	0 €		imagino precio erróneo consultar	https://www.cotecno.cl/solicitar-cotizacion/	https://www.cotecno.cl/sensor-de-flujo-de-savia-hfd-hfd8-100-hfd8-50/
HFD (HFD8-100)	0 €		imagino precio erróneo consultar	https://www.cotecno.cl/solicitar-cotizacion/	https://www.cotecno.cl/sensor-de-flujo-de-savia-hfd-hfd8-100-hfd8-50/

Oquedad del árbol

Oquedad del árbol (A través de sonido percutido o resistencia eléctrica de la madera)					
Sensor	Precio	Tiempo	Observaciones	Contacto	Enlace
PICUS Sonic	ERROR	ERROR	ERROR Envío solicitud	info@argus-electronic.de	https://www.agriexpo.online/es/prod/argus-electronic-gmbh/product-182221-61045.html
ARBOTOM	8.000 €		mínimo 8 unidades a 1.000€/Ud.	info@rinntech.com	https://www.agriexpo.online/es/prod/rinntech/product-182572-64702.html
ArborSonic 3D				office@fakopp.com	https://www.agriexpo.online/es/prod/fakopp-bt/product-182571-64678.html
Tomografía sónica	650\$- 654,39€				https://www.newdayarborist.com/tree-diagnostic-services/sonic-tomography/#:~:text=We%20are%20proud%20to%20be,per%20tree%20plus%20travel%20time.

Presión del jugo celular sobre la hoja

Presión del jugo celular sobre la hoja			
Sensor	Precio	Contacto	Enlace
R16 SENSOR DE TURGENCIA DE HOJA	6 Ud. 6.200 €	https://docs.google.com/forms/d/e/1FAIpQLSfTxqnFGh1kItckYk0mGog6HLoPfEy5X_inkyLWhQeTCxg/viewform	https://apgefert.greenfield.farm/wp-content/uploads/2020/09/R16.pdf
Sensor de Agua Yara		marcelodip@seedmech.com	https://www.seedmech.com/producto/sensor-de-agua-yara/

Oxígeno en el suelo

Oxígeno en suelo					
Sensor	Precio	Tiempo	Observaciones	Contacto	Enlace
Sensor de oxígeno gas en el aire y medios porosos Modelo SO de Apogee Instruments				https://www.lab-ferrer.com/contactar/	https://www.lab-ferrer.com/sensor-de-oxigeno-gas-en-el-aire-y-medios-porosos-so/
SOM1 Soil Oxygen Meter	1650\$-1657,19€			https://ictinternational.com/products/som1/som1-soil-oxygen-meter/	https://ictinternational.com/products/som1/som1-soil-oxygen-meter/
MIJ-03	960\$-964,18€(ICT)			https://edaphic.com.au/contact-us/	https://edaphic.com.au/oxygen/es-o2-soil-oxygen-sensor/
RK520-02				https://www.rikasensor.com/inquire/cart	https://www.rikasensor.com/professional-soil-oxygen-sensor-industry-for-soil-monitoring.html
Soil oxygen content analysis system	Instalado 5.000€	< 1 mes	Instalación fija	https://www.royaleijkelkamp.com/our-company/contact/ TLF:648312373	https://www.royaleijkelkamp.com/products/field-measuring-equipment/soil-oxygen/analysis/soil-oxygen-content-analysis-system/
ICTO2	ERROR	ERROR	ERROR Envío solicitud	https://www.hfspereu.com.pe/ict-international/sensor-de-oxigeno-del-suelo-ict2-ict-international?page=5&limit=75	https://geomor.com.pl/wp-content/uploads/2020/04/ICTO2_Soil_Oxygen_Sensor.pdf
CS511-L	NO VTA EUROPA		No se distribuye en Europa	https://www.campbellsci.es/questions	https://www.campbellsci.com/cs511-l

Oxígeno en la hoja

Oxígeno en hoja/aire:				
Sensores	Precio	Tiempo	Observaciones	Enlace
Gravity: I2C Oxygen Sensor (Raspberry Pi)	63,99 €			https://www.pi-shop.ch/gravity-i2c-oxygen-sensor
Seeed Grove - Oxígeno/Oxygen Sensor (MIX8410)	39,93 €			https://www.seeedstudio.com/Grove-Oxygen-Sensor-MIX8410-p-4697.html
EZO-O2™ Embedded Oxygen Sensor	100,99\$-101,47€		No valido, útil solo industria	https://atlas-scientific.com/probes/oxygen-sensor/#
o Sensor de oxígeno ME2-O2	35 €	1-5 días		https://es.aliexpress.com/item/32336030733.html?UTABTest=aliabtest300081_422161&_randl_currency=EUR&_randl_shipto=ES&src=google
MIX8410-O2	21\$-21,10€	5 días		https://spanish.alibaba.com/product-detail/MIX8410-O2-O2-oxygen-sensor-oxygen-1600221990224.html

Potencial hidráulico

Potencial hidráulico			
Sensor	Precio	Contacto	Enlace
PSY1 Psicrómetro	3390\$-3404,76€	https://ictinternational.com/products/som1/som1-soil-oxygen-meter/	https://ictinternational.com/products/psy1/psy1-stem-psychrometer/

Dióxido de carbono en la hoja

Dióxido de carbono en la hoja/aire				
Sensor	Precio	Observaciones	Contacto	Enlace
SEN0159	56,00 €			https://es.farnell.com/dfrobot/sen0159/sensor-anal-gico-gas-co2-placa/dp/3517879?cjevent=3f2e73c50b3e11ed814e03550a18050f&cjdata=MXxZfDB8WXww&CMP=AFC-CJ-ES-1765328&gross_price=true&source=CJ#anchorTechnicalDOCS
CO2 Gas Sensor	377\$-378,78€			https://www.vernier.com/product/co2-gas-sensor/
				https://pubs.acs.org/doi/10.1021/acs.jchmed.8b00473
EZO-CO2	166,99\$-167,78€			https://atlas-scientific.com/probes/co2-sensor/
RK300-03B	276.78€	Mínimo 2 unidades 138\$ unidas	https://www.rikasensor.com/inquire/cart	https://www.rikasensor.com/rk300-03-co2-concentration-sensor.html?gclid=Cj0KCQjw2_OWBhDqARIsAAUNTFUhG2x3CsUkHPF9gru_EoQsYQYddnUTTXntSFVA0u-HKst9SvAz5saAnPIEALw_wcB
Wireless Carbon Dioxide CO2 Sensor	257,71 €			https://heliot.ca/wireless-carbon-dioxide-co2-sensor/
MH-Z19C NDIR CO2 Sensor	45 €			https://www.winsen-sensor.com/sensors/co2-sensor/mh-z19c.html

Dióxido de carbono en el suelo

Dióxido de carbono en el suelo			
Sensor	Precio	Contacto	Enlace
SRS1000T		https://www.lab-ferrer.com/contactar/	https://www.lab-ferrer.com/sistema-para-medir-la-respiracion-del-suelo-srs/
SRS2000T		https://www.lab-ferrer.com/contactar/	https://www.lab-ferrer.com/sistema-para-medir-la-respiracion-del-suelo-srs/
eosGP CO2 sensor		https://edaphic.com.au/contact-us/	https://edaphic.com.au/products/soils/forerunner-gp-sensor/
eosFD	3.750 €	https://eosense.com/products/eosfd-soil-co2-sensor/#	https://eosense.com/products/eosfd-soil-co2-sensor/
SRC-2		https://coltein.com/producto/camara-de-respiracion-del-suelo-src-2/	https://coltein.com/producto/camara-de-respiracion-del-suelo-src-2/
LI-COR (CO2, CH4, and N2O)		https://www.licor.com/env/products/soil_flux/quote	https://www.licor.com/env/products/soil_flux/
LI-COR: 8250, 8250-01, 8200-104, 8200-104C		https://www.licor.com/env/products/soil_flux/quote	https://www.licor.com/env/products/soil_flux/long-term?gclid=EAialQobChMI2KS86fmT-QlVaoODBx1u7Qi7EAAYASAAEgLpEvD_BwE
CFLUX-1		https://ppsystems.com/quote-request-pp-systems/	http://www.hansatech-instruments.com/product/cflux-1/
MH-Z16		https://www.winsen-sensor.com/sensors/co2-sensor/mh-z16.html	https://www.winsen-sensor.com/sensors/co2-sensor/mh-z16.html

NPK

NPK (para saber fertilidad del suelo)				
Sensor	Precio	Tiempo	Contacto	Enlace
RS485 npk tester	169.78\$-170,58€			https://www.amazon.com/-/es/Taidacent-Sensores-Detector-Nitrógeno-nitrógeno/dp/B08MXXSP59
Soil NPK Sensor	52,3\$-52,55€			https://www.renkeer.com/product/soil-npk-sensor/
Replacement Sensor for Horiba LAQUAtwin NO3 Meters	160\$-160,76€			https://www.agriculturesolutions.com/replacement-sensor-for-horiba-laqua-no3-meters
Teralytic	1.200\$-1.205,10€	PEDIDO 2023		https://teralytic.com/index.html
EVTSCAN Sensor NPK	129,99 €			www.amazon.es/EVTSCAN-Fertilizante-Inteligente-nutrientes-precisión/dp/B0957874WZ/ref=sr_1_5
Soil NPK Sensor Revalcon			info@revalcon.com	https://revalcon.com/soil-npk-sensor/
All in One Soil Sensor NPK pH Ec Sensor for Agriculture Monitoring	190\$-190,90€			https://rainbowsensor.en.made-in-china.com/product/CFiApvTzhRYe/China-All-in-One-Soil-Sensor-NPK-pH-Ec-Sensor-for-Agriculture-Monitoring.html

pH

pH del suelo				
Sensor	Precio	Tiempo	Observaciones	Enlace
SEN-10972	DESCATALOGADO		DESCATALOGADO	https://www.digikey.es/es/products/detail/sparkfun-electronics/SEN-10972/5766911
SEN0161	34,82 €	DESCARTADO	MIDE PH SOLO AGUA	es.farnell.com/dfrobot/sen0161/kit-sensor-medidor-anal-gico-ph/dp/2946120
3 en 1, medidor luz solar, humedad del suelo, Detector de acidez, Monitor de PH	7,09 €			es.aliexpress.com/item/1005004063776178.html
Sonda de pH de alta precisión	26,27 €			www.amazon.es/adquisición-Suministros-industrial-profesional-hidropónicas/dp/B08S3N4212/ref=asc_df_B08S3N4212/
Soil detector	4,45 €			https://es.aliexpress.com/item/1005004300424170.html
SEN0249	116,85 €			es.farnell.com/dfrobot/sen0249/kit-sensor-ph-punta-lanza-anal/dp/3517936
DFRobot	84,75 €		DESCARTADO	es.rs-online.com/web/p/placas-y-kits-compatibles-con-arduino/2163779
SEN0169	67,17 €		DESCARTADO	https://es.farnell.com/dfrobot/sen0169/kit-sensor-anal-gico-ph-medidor/dp/3517876?gross_price=true
Sonda de pH ProMinent 150702	281,71 €		DESCARTADO	es.rs-online.com/web/p/electrodos-de-ph/0531292

Humedad del suelo

Humedad del suelo			
Sensor	Precio	Observaciones	Enlace
HPP809A033	143,32 €	DESCARTADO	https://www.digikey.es/es/products/detail/te-connectivity-measurement-specialties/HPP809A033/223-1589-ND/5277325
101990792	220,10 €		https://www.digikey.es/es/products/detail/seeed-technology-co-ltd/101990792/14672153
314990620	79,74 €		www.digikey.es/es/products/detail/seeed-technology-co-ltd/314990620/16570933
PIM520	13,97 €		https://www.digikey.es/es/products/detail/pimoroni-ltd/PIM520/13537122
28092	5.37€		https://www.digikey.es/es/products/detail/parallax-inc/28092/10453796

Estación meteorológica

Estación meteorológica		
Sensor	Precio	Enlace
WLAN Sainlogic	169 €	https://www.amazon.es/sainlogic-WS3500-Estaci%C3%B3n-meteorol%C3%B3gica-inal%C3%A1mbrica/dp/B081F1VTYT/ref=asc_df_B081F1VTYT/
Bresser Wetterstation Funk mit Außensensor Wetter Center 5-in-	49 €	https://www.amazon.es/Bresser-Centro-meteorol%C3%B3gico-Estaci%C3%B3n-meteorol%C3%B3gica/dp/B07DW23KWF/ref=asc_df_B07DW23KWF/
Estación meteorológica Sainlogic profi WLAN, estación meteorológica inteligente de Internet WiFi	206,11 €	https://www.amazon.es/sainlogic-FT0300-Estaci%C3%B3n-meteorol%C3%B3gica-Negro/dp/B0836FVVYZ/ref=asc_df_B0836FVVYZ/
Sainlogic WS 3500 Plus. Estación Meteorológica Inalámbrica Profesional 10	185 €	https://estacionmeteorologicaproyecto.com/sainlogic/sainlogic-ws3500-10-en-1/
froggit Estación meteorológica inalámbrica WH6000 Pro con conexión por Internet, incluye sensor exterior combinado 7 en 1	260 €	https://www.amazon.es/dp/B08T1571GL
Estación Meteorológica WLAN Sainlogic Profesional, Estación Meteorológica WiFi con Sensor Para Exteriores Alimentada Por Energía Solar, Medidor de Lluvia, Pronóstico del Tiempo	185 €	https://www.amazon.es/dp/B07TVMQ2CK
Froggit WH3600 Funk Estación meteorológica WiFi WLAN	150 €	https://www.amazon.es/dp/B08H8H1MHQ
Froggit Estación meteorológica WH3000 SE (Edition 2018) a Internet Wi-Fi, App	149,99 €	https://www.amazon.es/dp/B06X6JPJ6Q
Sainlogic Estación meteorológica inalámbrica con 3 sensores exteriores, previsión meteorológica, pantalla a color, termómetro inalámbrico con sensor exterior.	65,37 €	https://www.amazon.es/dp/B07V6FYTKL

Microcontroladores

Placas			
Placas	Precio	Observaciones	Enlace
TelosB	333,70 € + IVA		https://es.aliexpress.com/item/32858705651.html
Wasp mote	14.50€	LIBELIUM WASPMOTE V1.1 + EXPANSION RADIO BOARD (WITHOUT BOX)	www.ebay.es/itm/154862060704
Arduino uno	29 €		es.farnell.com/arduino/a000066/arduino-uno-placa-de-evaluaci/dp/2075382
Arduino mega R3	50,99 €		https://www.amazon.es/Arduino-Mega-2560-R3-Microcontrolador/dp/B0046AMGW0/ref=asc_df_B0046AMGW0/
Raspberry 4B	74,10 €		https://www.kubii.es/les-cartes-raspberry-pi/2772-nouveau-raspberry-pi-4-modele-b-4gb-kubii-0765756931182.html?src=raspberrypi

Apéndice B: Características de los sensores y propuesta de selección

Sensores de Temperatura de Estoma

Hay que tener en cuenta que los rangos máximos y mínimos de temperatura, los registros en España fueron -36 °C y 48 °C.

LT-1M

Specifications

Measurement range	0 to 50 °C
Instrumental accuracy	< 0.15 °C
Output	
LT-1M	0 to 2 VDC
LT-1Mi	4 to 20 mA
Supply voltage	10 to 30 VDC
Power	
LT-1M	1.5 W max
LT-1Mi	2 W max
Probe dimensions, mm	50 W × 20 H × 10 D
Probe weight	1.6 g
Contact area of thermistor	About 1 mm ²
Standard calibration	$T(°C) = f(V, Volt)$
Best fit:	$T = 1.8649V^3 - 4.5048V^2 + 26.542V - 0.0099$
Approximation error	< ±0.06 °C
Linear fit:	$T = 24.477V - 0.1352$
Approximation error	< ±1 °C
Tolerance range	±0.08 °C
Cable length between probe and signal conditioner	1 m

El mayor problema de este sensor son las dimensiones, se puede observar que la estructura que envuelve al sensor es más grande que el ancho de la hoja, lo bueno es que el área de contacto del termistor es aproximadamente el ancho de la hoja, así que si le hacemos unas modificaciones podría funcionar, además que es el que tiene una precisión mayor de los que hay nombrados en esta hoja (quitando el SHT1x). Otro problema es que el rango de medida puede no funcionar en días muy fríos. Recomendable

LT-2M

Especificaciones	
Rango de medición	5 - 50 °C
Precisión instrumental	± 0.2 °C
Estabilidad	± 0.5 °C más de un año
Salida	0-2 V o 4-20 mA (opcional)
Dimensiones totales	35 x 15 x 15 mm ³
Adaptador de alimentación (opcional)	De 10 a 30 Vdc
Tensión de alimentación	+15 Vdc ± 10%

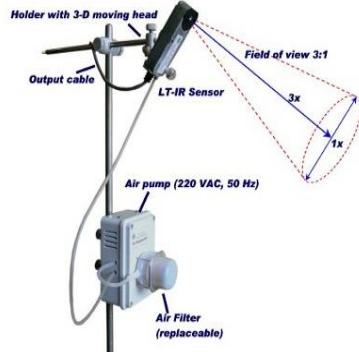
Este tiene un menor rango de temperatura que el anterior, haciendo que no se pueda tomar medidas con precisión en días de invierno, además que la precisión es menor que la anterior. En la parte de la alimentación debe ser algo superior al LT-1M. Tiene dimensiones superiores al anterior, pero el problema es que, al ser de plástico, no creo que pueda ser tan moldeable que los alambres del LT-1M. En mi opinión, sería mejor el LT-1M.

No recomendable

LT-IRM

Specifications

Measurement range	0 to 100 °C
Absolute accuracy	±1.0 °C
Repeatability	±0.1 °C
Output	
Standard	0 to 2 VDC, linear
Optional	4 to 20 or 0 to 20 mA
Supply voltage	12 to 24 VDC
Field of view	3:1
Update time	Approx. 250 ms
Emissivity setting	0.9
Spectral response	5.5 to 20 µm
Ambient temperature	0 to 70 °C
Relative humidity	95% non condensing
Probe dimensions (w/cable), mm	94.3 H x 34.9 W x 19.4 L
Probe weight	230 g
Protection index with air on	IP 54



Mayor rango de temperatura que el Lt-1M, pero en altas temperaturas, así que es despreciable este parámetro. La precisión es mucho peor al Lt-1M. Necesita más alimentación que el Lt-1M, pero menos que el Lt-2M. El principal problema es que, al ser un sensor infrarrojo, se va a tener que acercar mucho a la hoja para que no mida el

suelo o los alrededores de la hoja, además que el viento hará que se mueva las hojas y que no tome buenas medidas. Yo lo descartaría. **No recomendable**

LS-40

LS 40 - LAT-C Leaf temperature sensor for conifer needles.

Extremely lightweight. Direct, highly precise and continuous measurements of needle AND air temperature, range between -20 and +70°C. Spatial signal integration via multiple measurement points at the surface of several needles. Suitable for needle sizes > 3 mm length.

Analogue sensor outputs (mV):

1) thermopile: 1x differential channel, - 4.2 to 4.2 mV for a temperature difference between needle and air from -10 to +10 °C

2) thermistor: 1x single ended channel, @ typical excitation voltage of 2500 mV -> analogue output 0-2500 mV.

Operating Environment:

25 to 70 °C; 0 to 100 % relative humidity.

standard configuration:

Complete with pluggable 5 m sensor cable extension, extension may be up to 20 m.

Cable configuration: bare end

Lo bueno de este es que está diseñado específicamente para las coníferas. El rango de temperatura lo cumple, pero solo opera en el ambiente de 25 a 70°C. **Descartado**

LAT-B3

Technical Specifications

Name	LAT-B3 : Leaf-&-Air Temperature Sensor, broadleaf type
Application positioning	Mounting position: Leaf surface Dual-probe spacing: User-configurable distance between T_{leaf} and T_{air} probes max. 35 mm
Suitable leaf size and thickness	Standard sensor size for leaves of: Length > 1.4 cm length Width between 0.8 to 20 cm (for larger leaf widths on request). Stable magnet-mounting possible for leaf thickness < 0.7 mm
Measurement range	-25 to + 70°C
Accuracy	Sensor dependent: Tolerance of Absolute Tair & Tleaf: $\pm 0.4^{\circ}\text{C}$ in temperature range between +5°C to+40°C $\pm 0.8^{\circ}\text{C}$ in temperature range between -25°C to+70°C Tolerance of leaf-to-air temperature difference ($\Delta T_{leaf-air}$): $\pm 0.2^{\circ}$ in temperature range between -25°C to+70°C Logger dependent, @ 25 °C: e.g. CR300 series: $\pm 0.01^{\circ}\text{C}$ e.g. DL18 Logger: $\pm 0.03^{\circ}\text{C}$
Resolution	Logger dependent, @ 25 °C: e.g. CR300 series: $0.25 \cdot 10^{-4}^{\circ}\text{C}$ e.g. DL18 Logger: $0.35 \cdot 10^{-3}^{\circ}\text{C}$
Size and weight	Diameter 12 mm, weight ca. 0.9 g
Output signal type	Supplied with 2500 mV, output signal is between 0 to 2500mV
Power supply	Excitation voltage Vex usually switched 2500 mV, power up 100ms max. Power consumption negligible.
Operating conditions	Air temperature: -25 to 70 °C, air humidity: 0 to 100%
Cable length	0.5m + 4.5m plug-in extension, plug-in extension up to max. 50 m possible

Se puede observar que el rango de medida de la temperatura lo cumple. También las dimensiones de la hoja. La alimentación es la más baja de las que se han visto. [El recomendado](#)

SHT1x

Relative Humidity

Parameter	Condition	min	typ	max	Units
Resolution ¹		0.4	0.05	0.05	%RH
		8	12	12	bit
Accuracy ² SHT10	typical		±4.5		%RH
	maximal		see Figure 2		
Accuracy ² SHT11	typical		±3.0		%RH
	maximal		see Figure 2		
Accuracy ² SHT15	typical		±2.0		%RH
	maximal		see Figure 2		
Repeatability			±0.1		%RH
Hysteresis			±1		%RH
Non-linearity	linearized		<<1		%RH
Response time ³	τ (63%)		8		s
Operating Range		0		100	%RH
Long term drift ⁴	normal		< 0.5		%RH/yr

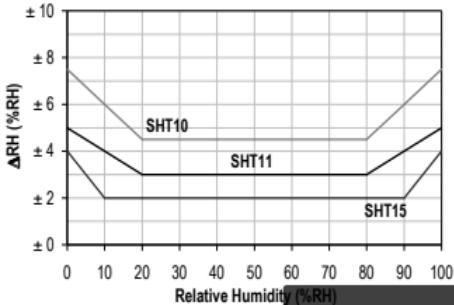


Figure 2: Maximal RH-tolerance at 25°C per sensor type.

Temperature

Parameter	Condition	min	typ	max	Units
Resolution ¹		0.04	0.01	0.01	°C
		12	14	14	bit
Accuracy ² SHT10	typical		±0.5		°C
	maximal		see Figure 3		
Accuracy ² SHT11	typical		±0.4		°C
	maximal		see Figure 3		
Accuracy ² SHT15	typical		±0.3		°C
	maximal		see Figure 3		
Repeatability			±0.1		°C
Operating Range		-40		123.8	°C
		-40		254.9	°F
Response Time ⁶	τ (63%)	5		30	s
Long term drift			< 0.04		°C/yr

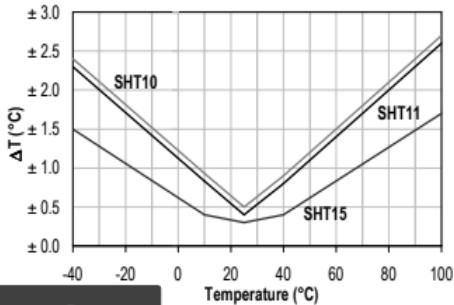
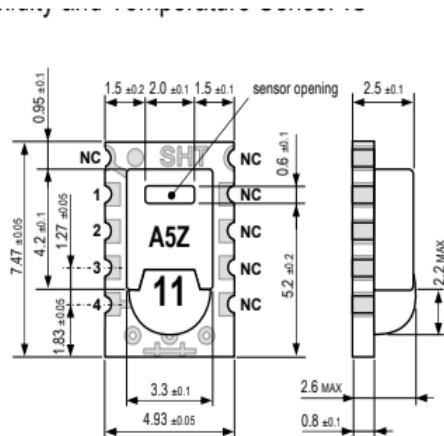


Figure 3: Maximal T-tolerance per sensor type.



Electrical and General Items

Parameter	Condition	min	typ	max	Units
Source Voltage		2.4	3.3	5.5	V
Power Consumption ⁵	sleep		2	5	µW
	measuring		3		mW
	average		90		µW
Communication	digital 2-wire interface, see Communication				
Storage	10 – 50°C (0 – 125°C peak), 20 – 60%RH				

Este podría ser tremadamente útil para nuestro trabajo, cumple el rango de temperatura especificado, también el de humedad. Uno de los problemas menores es que el ancho de la apertura del sensor es mayor la anchura de la hoja, entonces se tendría que girar el chip. Pero el mayor problema que tiene es que en el apartado 1.3 te indica que el sensor no se puede exponer demasiado tiempo, sino que es para tomar medidas más esporádicas, por tanto, no podríamos implantarlo sobre la planta para la toma de medidas a largo plazo.

Descartado

Sensores de Humedad

En el caso de los sensores de humedad, sería conveniente que tuviesen un rango de medición entre el 0 % RH y el 100 % RH, además de poder tomar mediciones de forma adecuada en los rangos de temperaturas visto anteriormente. Sería conveniente que tuviese un mayor tiempo de vida y que fuese fácilmente integrable en el sistema.

HM102

DATOS TÉCNICOS	
PRINCIPIO DE MEDICIÓN	Capacitivo
CAMPO DE MEDICIÓN	0...100% de humedad de la hoja
PRECISIÓN	± 5%
CONSUMO	< 1 mA
SALIDA	Analógica 0,5...3 V
TEMPERATURA DE TRABAJO	-30...+60 °C
CABLE	Cables abiertos con 4 polos, 5 o 10 m de longitud a definir cuando se va a pedir
PROTECCIÓN	IP67
ALIMENTACIÓN	4 pilas de botón LR44, 1,5 V
PESO	Acerca de 100 g (incluyendo el cable da 5 m)
MEDIDAS (AxHxP)	61 x 115 x 11 mm (sin cable). Espesor del sensor 1,6 mm

Este modelo cumple las condiciones de temperatura y humedad. Además, se podría colocar en la parte exterior del follaje del árbol con una inclinación de 30-45° respecto al suelo. Lo malo es que utiliza pilas de botón y a lo mejor hay que modificarlo.
Recomendable

RK300-04

SPECIFICATIONS

Item	Technical Specification
Range	Wetness: 0-100% Temperature: -40-+80°C
Accuracy	Wetness: ±3%(0-50%),±5%(>50%) Temperature: ±0.5°C
Repeatability	<±3%fs
Temperature Drift	≤0.2%FS/°C
Supply	5VDC,12-24VDC
Output	4-20mA,0-5V,0-2V,RS485
IP Rating	IP65
Operating Temperature	-40°C-+80°C
Dimension	65*13*145mm
Storage	-40-70°C@20%-90%RH

Las características se ajustan a la perfección observando que tiene una mayor precisión en su medición. El problema estriba en la necesidad de comprar un mínimo de 5 unidades, aunque en realidad es el más barato por unidad siendo este valor de 84,26 €. La ventaja es que la alimentación no va por pilas. **Descartado**

Leaf Wetness Sensor de Darrera

Sensor specifications

- Range: 0 to 15 leaf wetness index
- Accuracy: ± 0.5 leaf wetness index

Electrical specifications

- Power supply voltage: 3 VDC
- Power consumption: 100 μ A
- Output signal: 2.5 to 3 VDC

Mechanical specifications

- Material: fiberglass
- Dimensions: 58 x 102 x 58 mm
- Weight: 370 g

Se puede observar que no dice nada del rango de temperaturas que soporta, además que el de humedad utiliza su propia escala (índice de humedad de hoja), lo bueno es que es el que menos consume y es de los más barato de los que hay. **Descartado**

PESSL

TECHNICAL SPECIFICATIONS

Supply voltage	4.75 V - 5.25 V
Supply current	max. 1500 μ A
Short circuit protection	Infinite (within supply voltage range)
Dry / Wet threshold	220 - 390 kOhm
Output	Dry: max. 0.4 VDC Wet: min. VCC-0.4 VDC
Electronic	Totally plastic encapsulated - SMD
Dimensions	42 mm x 78 mm x 15 mm
Cable length	5 m

Se observa que no dice nada del rango de temperaturas que soporta, la alimentación es similar a las anteriores, el precio es elevado, además que es rara la forma de ponerlo en el árbol. **No recomendado**

NMETOS200

NMETOS200	700228 (HL7800)	NMETOS con sensor de temperatura, humedad relativa, pluviómetro y humectación de hoja. Calcula el punto de rocío, DPV y deltaT. (ENFERMEDADES)	1	967,98 €	2,5	943,78 €
BEGINNER INTERMEDIATE ADVANCED SPECIFIC						
Precipitation	✓	✓	✓	✓		
Air Temperature		✓	✓	✓	✓	
Relative Humidity		✓	✓	✓	✓	
Volumetric Water Content			✓			
Soil Temperature			✓			
Soil Water Tension			✓			
Leaf Wetness		calculated	calculated	✓		
Dew Point*		calculated	calculated	calculated		
VPD*		calculated	calculated	calculated		
Delta T*		calculated	calculated	calculated		

Este se podría utilizar como estación meteorológica, pero es un tanto caro.

No recomendable

LWS-L

Measurement Description	Dry, frosted, wet
Signal Type/Output	Analog voltage
Measurement Time	10 ms
Power	2.5 Vdc @ 2 mA to 5 Vdc @ 7 mA
Output	250 to 1500 mV (millivolt reading relates to moisture state)
Operating Temperature Range	-40° to +60°C
Life Expectancy	2+ years (continuous use)
Painting	Does not require painting.
Dimensions	12.0 x 5.8 x 0.8 cm (4.7 x 2.3 x 0.3 in.)
Weight	0.14 kg (5 oz) with 4.57 m (15 ft) cable

Tiene muy buenas características. Recomendable, compararlo con el PYTHOS (Son casi idénticos, solo que tiene el cable más corto, pero más rango de salida)

HD3901

Measurement Principle	Capacitive
Measurement Range	0...100% of leaf area wetness
Accuracy	± 5%
Power Supply	5...18 Vdc
Consumption	< 1 mA
Output	Analog 0.5...3 V
Operating Temperature	-30...+60 °C
Dimensions	61 x 115 x 11 mm (excluding cable). Width of the sensor 1.6 mm
Cable	4 poles with open wires at the end, length 5 or 10 m to be defined when ordering
Weight	About 100 g (including the 5 m cable)
Protection Degree	IP67

Cumple las especificaciones, no utiliza pilas, tiene una precisión normalita al igual que la alimentación. **Recomendable, pero un poco caro.**

PYTHOS31

Dimensions	
Length	12.0 cm (4.7 in)
Width	5.8 cm (2.3 in)
Height	0.8 cm (0.3 in)
Operating Temperature Range	
Minimum	-40 °C
Typical	NA
Maximum	+60 °C
NOTE: Sensors may be used at higher temperatures under certain conditions; contact Customer Support for assistance.	
Cable Length	
5 m (standard) 40 m (maximum custom cable length)	
NOTE: Contact Customer Support if a nonstandard cable length is needed.	
Cable Diameter	
0.165 ± 0.004 (4.20 ± .10 mm) with min. jacket of 0.030 (.76 mm)	
Connector Types	
3.5-mm stereo plug connector or stripped and tinned wires	
Stereo Plug Connector Diameter	
3.50 mm	
Conductor Gauge	
22 AWG /24 AWG drain wire	
Supply Voltage	
Minimum	2.5 VDC
Typical	NA
Maximum	5.0 VDC
Settling Time	
10 ms	
Output	
300–1,250 mV (depends on excitation voltage)	
Data Logger Compatibility	
Data acquisition systems capable of switched 2.5–5.0 VDC excitation and single-ended voltage measurement at greater than or equal to 12-bit resolution.	

No tiene malas dimensiones, el rango de temperatura es bueno, tiene un cable de 5 m que no viene mal, la alimentación es muy buena. Lo malo es que no dice nada de la precisión, pero supongo que será del 95% como casi todos. **Muy recomendable**

Flujo de savia

El sensor de flujo de savia está diseñado para tomar medidas en entornos abiertos, por tanto, podrá trabajar en las condiciones de humedad y temperaturas antes vistas. El mayor inconveniente de este tipo de dispositivos es su complejidad a la hora de integrarlo en el ejemplar.

SF-G

Nombre	Sap Flow Sensor Type SF-G
Tamaño Aguja	Diameterr 1.5 mm, Length 33, 43, 63 mm
Longitud Cable	5 m, extendable to 20 m
Adecuado para dimensiones árbol	Diameter> 2 cm
Alimentación	Una Fuente CCS es suficiente para alimentar hasta 3 sensores SF-G y consume 1 W (12 V x 84 mA DC, estabilizado)
Salida	0 µV to 1000 µV DC

Opciones / Información para Pedidos:

- Cable de extensión (please specify in meters, but max. 20 m)
- CCS Fuente Alimentación (Una Fuente CCS es suficiente para alimentar hasta 3 sensores SF-G)
- Herramientas de Instalación
- Data Logger

Está bien ya que no consume demasiado y precio económico.

El recomendado

SGEX-16, 29 y 25

Especificaciones mecánicas

Model	Plant Diameter (mm)	Min Dia. (mm)	Max Dia. (mm)	Min Installed Length (Axial) (mm)	Typ. Installed Length-Shield & Insulation (Axial) (mm)
SGEX-9	9	8	10	70	150-350
SGEX-10	10	9.5	13	70	170-360
SGEX-13	13	12	16	80	190-380
SGEX-16	16	15	19	90	210-400
SGEX-19	19	18	23	100	240-450
SGEX-25	25	25	29	120	280-500

Especificaciones eléctricas

Model	Heater (Ohms)	Typical Voltage (dc)	Max Voltage	Typical Power (W)	Max Power (W)
SGEX-9	120	4.0	5.0	.13	.21
SGEX-10	140	4.5	5.0	.15	.18
SGEX-13	120	4.5	5.0	.17	.21
SGEX-16	100	4.5	5.0	.20	.25
SGEX-19	60	4.5	5.0	.34	.42
SGEX-25	42	4.5	5.0	.48	.60

Especificaciones ambientales

	Min (°C)	Max (°C)	
Operating Temp	0	50	Below freezing plants and sensor will not have / respond to transpiration.
Installation Temp	15	35	At temps below min, EXO skin is brittle, may sustain damage during installation.
Storage Temp	0	60	Store clean and dry, away from direct heat.

Son muy caros, pero al ser del Lab-Ferrer a lo mejor hacen precio si cogemos muchos.
Recomendado

FLOW32A-1k

VARINST	Flow32A-1K	1,00	9.800,000	9.800,000
Flow32A Sap Flow System				
Datalogger	CR1000x logger with built-in sap flow calculator			
Base Inputs	8 Differential Channels - Analog, SDI-12			
Channel Expansion	AM16/32 Relay Multiplexer			
Expanded Inputs	32 Differential Channels - Analog			
Sensor Capacity	(8) Dynagages up to (32) sensors with expansion			
Range	+/-200 to 5000 mV			
Resolution	0.05 to 0.88 uV			
Voltage Regulation	AVRD Dual Voltage, 1.5 - 10 V, 5 A each			
Base Memory	4 MB			
Hourly data - 1 year				
Daily data - 1 year				
Sap flow calculation - 8 months for 8 gages				

Datalogger	CR1000x logger with built-in sap flow calculator
Base Inputs	8 Differential Channels - Analog, SDI-12
Channel Expansion	AM16/32 Relay Multiplexer
Expanded Inputs	32 Differential Channels - Analog
Sensor Capacity	(8) Dynagages up to (32) sensors with expansion
Range	+/-200 to 5000 mV
Resolution	0.05 to 0.88 uV
Voltage Regulation	AVRD Dual Voltage, 1.5 - 10 V, 5 A each
Base Memory	4 MB Hourly data - 1 year Daily data - 1 year Sap flow calculation - 8 months for 8 gages
Expanded Memory	Removable microSD flash memory, up to 16 GB
Communications	9-PIN Male RS-232 Serial Cable, 15 ft (5m)
Battery	7 Ahr / 12 V Sealed Lead Acid
Charger	120 V AC, 6 A 220 V AC, 4.5 A
Sensor Cables	8 x 7.6 m (25 ft) with Connectors
Enclosure	White fiberglass, NEMA 4X, with pole mounts, lockable, 17 x 14 x 6.5" (43 x 35 x 16 cm)
System Weight	11.5 kg

Muy caro. No recomendable

SF-L

Nombre	Sap Flow Sensor Type SF-L (SF 10)
Tamaño de la Aguja	Diameterr 1.5 mm, Length 33, 43, 63 mm
Longitud del Cable	5 m, extendable to 20 m
Adecuado para tamaños de árbol	Diameter> 15 cm
Alimentación	Una Fuente CCS es suficiente para alimentar hasta 3 sensores SF-G y consume 1 W (12 V x 84 mA DC, estabilizado)
Salida	0 µV a 1000 µV DC

Opciones / Información para Pedidos:

- Cable de extensión (please specify in meters, but max. 20 m)
- [CCS](#) Fuente Alimentación (Una Fuente CCS es suficiente para alimentar hasta 3 sensores SF-G)
- Herramientas de Instalación
- Data Logger

Similar al SF-G, pero más caro, principal diferencia el diámetro del árbol.
No recomendable

HFD8-100 / HFD8-50

No hay datos ni precio. **No recomendable**

Oxígeno en el suelo

Los precios son muy abusivos, así que a lo sumo podríamos utilizar el MIJ-03, además que la mayoría te ofrecen un servicio de instalación, se podría buscar algún estudio que haya hecho la medición del oxígeno en el suelo de una manera más económica.

Oxígeno en el aire/hoja

Al ser más económicos se puede realizar un mejor estudio de comparación.

SEN0322

FEATURES

- > High sensitivity
- > I2C Interface
- > Compatible with both 3.3V and 5V micro-controllers
- > Polarity protection

SPECIFICATION

- > Detection of Gas: oxygen
- > Operating Voltage: 3.3 to 5.5V DC
- > Output Signal: I2C output
- > Measurement Range: 0~25%Vol
- > Maximum Measurement limit: 30%Vol
- > Resolution: 0.15%Vol
- > Sensitivity: (0.10±0.05) mA (in the air)
- > Stability: <2% (Every month)
- > Repeatability: <2%
- > Response Time: ≤15 seconds
- > Operating Temperature: -20 ~ 50°C
- > Operating humidity: 0 ~ 99%RH (no condensation)
- > Pressure Range: standard atmospheric pressure ±10%
- > Lifetime:> 2 years (in the air)
- > Dimension(L x W x H): 37 * 27 * 24.5 mm/1.46 * 1.06 * 0.97 inches
- > Weight:0.037kg

Tiene buen rango de alimentación, de temperatura, soporta bien la humedad, tiene buen error, dura dos años y utiliza protocolo I2C. **Recomendable**

EZO-02

Reads	Gaseous O2
Range	0 – 42% (2x atmospheric O2 levels)
Calibration	Factory calibrated
Response time	1 reading per second
Resolution	0.01
Accuracy	+/- 0.01% (0.02 PPT)
Connector	5 lead data cable
Cable length	1 meter
Data protocol	UART & I2C
Default I2C Address	108 (0x6c)
Data format	ASCII
Operating voltage	3.3V – 5V
Life expectancy	~3.5 years

Tiene mejor característica en la medida del oxígeno, condiciones de alimentación similares, comunicación I2C, pero es para industria. **No Válido**

ME2-02

Artículo	Parámetro
Gas de detección	O2
Rango de medición	0 ~ 25% Vol
Concentración máxima de detección	30% Vol
Sensibilidad	(0,1 ~ 0,3) mA (en aire)
Tiempo de respuesta (T90)	≤ 15S
Resistencia de carga (recomendada)	100R
Repetibilidad	Valor de salida inferior al 2%
Estabilidad (/mes)	Menos de 2%
Deriva cero (-20 °C ~ 40 °C)	≤ 0.1% vol
Temperatura de almacenamiento	-20 °C ~ 50 °C
Almacenamiento de humedad	0% ~ 99% RH
Rango de presión	Atmósfera normal ± 10%
Vida útil esperada	2 años

Similar al primero, pero tiene peor sensibilidad.

Precio Recomendable

MIX8410-O2

Specification	Details
Input Voltage	3.3/5V
Interface	Analog
Measurement range	0-25%
Maximum Overload	30%
Sensitivity	0.1±0.03 mA
Repeatability	±2%
Response Time	<10S
Working Temperature Range	-20 - 50°C
Storage Temperature Range	0 ~ 50°C
Operating Humidity Range	15 ~ 90% RH, no-condensing
Power on aging time	30min

Item	Specification
Target Gas	Oxygen
Measurement Range	0 ~ 25%
Maximum Overload	30%
Sensitivity	$0.1 \pm 0.03 \text{ mA}$
Repeatability	$\pm 2\%$
Response Time (T90)	< 10 Seconds
Recommend Resistor	100Ω
Long Term drift	< 5% per year
Life Expectancy(in air)	2 Year
Operating Temperature Range	-20 ~ 50°C
Storage Temperature Range	0 ~ 50°C
Operating Humidity Range	15 ~ 90% RH, no-condensing

Características muy similares al anterior, pero con mejor sensibilidad y trabaja en un rango menor de humedad. Es el más económico. [El recomendado](#)

Los dos que más recomiendo son el [Gravity I2C](#) al conectarse directamente a la Rasp/Arduino, tener mejor características. Pero el [MIX8410-O2](#) también es muy bueno, al tener un precio tan económico y no tener muy malas características.

Dióxido de carbono en el suelo

Los precios de estos sensores son muy abusivos o no hay por la red demasiada información de estos, por tanto, no se estudiarán.

Dióxido de Carbono en hoja/aire

La mayoría de estos sensores están diseñados para tomar datos medioambientales, pero se estudiarán y se aplicarán para la toma del dióxido del entorno al follaje del árbol. Es necesario que sobrevivan a las condiciones de humedad y temperatura impuestas.

SENO159

Atributo del producto	Valor del atributo	Buscar similar
Fabricante:	DFRobot	<input type="checkbox"/>
Categoría de producto:	Herramientas de desarrollo de sensor de diversas funciones	<input checked="" type="checkbox"/>
RoHS:	 Detalles	
Producto:	Plug-In Modules	<input type="checkbox"/>
Tipo:	CO2 Sensor	<input type="checkbox"/>
La herramienta es para la evaluación de :	MG-811	<input type="checkbox"/>
Voltaje operativo de suministro:	0 V to 5 V	<input type="checkbox"/>
Empaquetado:	Bulk	<input type="checkbox"/>
Marca:	DFRobot	
Tipo de interfaz:	Analog	
Tipo de producto:	Multiple Function Sensor Development Tools	
Serie:	Gravity	
Cantidad del paquete de fábrica:	1	
Subcategoría:	Development Tools	
Peso unitario:	35 g	

Symbol	Parameter Name	Technical	Remarks
V_H	Heating Voltage	6.0 ± 0.1 V	AC or DC
R_H	Heating Resistor	$30.0 \pm 5\%$ Ω	Room Temperature
I_H	Heating Current	@200mA	
P_H	Heating Power	@1200mW	
Tao	Operating Temperature	-20—50	
Tas	Storage Temperature	-20—70	
? E?M F	Output	30—50mV	350—10000ppmCO2

No tiene malas características, además que cumple los rangos de temperaturas estipulados. Es *plug and play*, así que es más fácil de utilizar. Recomendable

CO₂ Gas Sensor

Specifications

- Response time: 95% of full-scale reading in 120 seconds
- Warm-up time: 90 seconds
- Pressure effect: 0.19% of reading/mm of Hg from standard pressure
- Output signal range: 0-4.0 V
- Gas sampling mode: diffusion
- Normal operating temperature range: 25°C (±5°C)
- Operating humidity range: 5–95% (non-condensing)
- Storage temperature range: -40 to 65°C
- Measurement Range:
 - Low range: 0 to 10,000 ppm CO₂
 - High range: 0 to 100,000 ppm CO₂
- Typical Accuracy (at standard pressure, 1 atm):
 - Low range:
 - 0 to 1,000 ppm: ±100 ppm
 - 1,000 to 10,000 ppm: ±10% of reading
 - High range:
 - 0 to 1,000 ppm: ±200 ppm
 - 1,000 to 100,000 ppm: ±20% of reading
- Resolution:
 - 0 to 10,000 ppm CO₂: 3ppm
 - 0 to 100,000 ppm CO₂: 30ppm

Mejores características que el anterior, pero muy caro. No recomendable

EZO-CO₂

Reads	Gaseous CO ₂
Range	0 – 10,000 ppm
Calibration	Factory calibrated
Response time	1 reading per second
Resolution	1 ppm
Accuracy	(+/- 5%) + (+/- 50 ppm)
Connector	5 lead data cable
Warmup time	10 seconds
Cable length	1 meter
Data protocol	UART & I2C
Default I2C address	105 (0x69)
Data format	ASCII
Operating voltage	3.3V – 5V
Life expectancy	~5.5 years

Power consumption Absolute max ratings

	LED	MAX	SLEEP	Parameter	MIN	TYP	MAX
5V	ON	45 mA	3.4 mA	Storage temperature	-65 °C		75 °C
	OFF	44 mA		Operational temperature	-20 °C	25 °C	50 °C
3.3V	ON	42 mA	3.0 mA	VCC	3.3V	3.3V	5.5V
	OFF	41 mA		Humidity Range 0 to 95% rh non-condensing			

Mejores características que el primero, pero algo más caro. Descartado

RK300-03B

Item	Technical Specification
Range(concentration)	0-5000ppm,0-10000ppm
Accuracy	$\pm 5\text{ppm} + 2\% \text{rdg}$ @25°C
Supply	5VDC,12-24VDC
Output	4-20mA,0-5V,RS485
Power Consumption	<0.25w
Warm Up Time	3min
Response Time	<20s
Temperature Drift	$\leq 0.2\% \text{FS}/^\circ\text{C}$
Stability	$\pm 40\text{ppm}/\text{year}$
Repeatability	$\pm 1\%\text{fs}$
Operating Temperature	-20°C-+60°C@15-80%RH
Storage	-40-70°C@20%-90%RH
Shell Material	ABS

No tiene malas especificaciones, aguanta la temperatura especificada, pero la humedad no es tan buena. Habría que comprar 2 sí o sí encareciéndolo demasiado.

No recomendable

Wireless Carbon Dioxide CO2 Sensor

Este sería el ideal, la cosa es que es bastante caro y ya está diseñado para ser un nodo de una red LoRa.

Descartado

MH-Z19C

Model No.	MH-Z19C
Detection Gas	CO2
Working voltage	5.0±0.1V DC
Average current	< 40mA (@5V power supply)
Peak current	125mA (@5V power supply)
Interface level	3.3 V (Compatible with 5V)
Detection Range	400~10000ppm(optional)
Output signal	Serial Port (UART) (TTL level 3.3V) PWM
Preheat time	1 min
Response Time	T ₉₀ < 120 s
Working temperature	-10 ~ 50 °C
Working humidity	0 ~ 95% RH (No condensation)
Storage temperature	-20~60 °C
Weight	5 g
Lifespan	> 10 years

Realmente es el que tiene mejor relación calidad precio, además que tiene una vida útil muy elevada, cumpliendo las características medioambientales. El recomendado

NPK

Lo malo de estos sensores es que muchos se venden en amazon o aliexpress, pero no tiene catálogo, además de no ser muy fiables

Soil NPK sensor

Power supply: 5-30VDC
Maximum power consumption: ≤0.15W
Operating temperature: -40~80°C
NPK parameters:
 Range: 0-1999 mg/kg(mg/L)
 Resolution: 1 mg/kg(mg/L)
 Precision: ±2%FS
Response time: ≤1S
Protection grade: IP68
Probe material: 316 stainless steel
Sealing material: Black flame-retardant epoxy resin
Default cable length: 2 meters, cable length can be customized
Dimensions: 45*15*123mm
Output signal: RS485/4-20ma/0-5v/0-10v

Cumple especificaciones de temperatura, tiene buen rango y resolución, buena precisión y buen precio. **Recomendable**

Teralytic

Caro y no disponible hasta 2023, aunque es muy buen sensor. **No recomendable**

EVTSCAN

Especificación:

New and old models are shipped randomly



New Type

Old Type

- Material: ABS
 - Color: como se muestra en la imagen
 - Longitud (no incluye cable): aprox. 14 cm/5,5 pulgadas
 - Ancho: Approx.4cm/1.6in
 - Peso: aprox. 203 g/7,2 oz
 - Rango de medición: 0-1999 mg/kg
 - Humedad de trabajo: 5 a 95% (humedad relativa), sin condensación
 - Precisión de medición: $\pm 2\%$ F.s

 - Resolución: 1mg/kg (mg/l)
 - Tasa de baudios: 2400/4800/9600
 - Tiempo de respuesta (T90, segundos): menos de 10
 - Puerto de comunicación: RS485
 - Temperatura de Trabajo: 5 a 45°C
- Fuente de alimentación: 12 V-24 V CC
 - Clase de protección: IP68

No cumple especificaciones de temperatura, pide mucha tensión de alimentación, rango de medición y resolución similar al Soil NPK Sensor. **No recomendable**

Soil NPK sensor Revalcon

No se ha conseguido información. **No recomendable**

Sensor de pH

Los rangos de acidez que tiene que tomar suelen ser entre los 5 y los 7 de pH, pero todos los sensores cumplen esta característica de sobra. Por tanto, lo fundamental va a ser que puedan trabajar en condiciones óptimas en las condiciones de temperatura y humedad. Además, de tener en cuenta el precio.

SEN-10972

Features

- Full range pH reading from .001 to 14.000
- Accurate pH readings down to the thousands place (**+/- 0.02**)
- Temperature dependent or temperature independent readings
- Flexible calibration protocol supports single point, 2 point, or 3 point calibration
- Calibration required only once per year with Atlas Scientific pH probe
- Single reading or continuous reading modes
- **Data format is ASCII**

Two data protocols

- UART asynchronous serial connectivity
- (RX/TX voltage swing 0-VCC)
- I²C (default I²C address 0x63)
- Compatible with any microprocessor that supports UART, or I²C protocol
- Operating voltage: 3.3V to 5V
- Works with any off-the-shelf pH probe

Sleep mode power consumption

- 0.995mA at 3.3V

Descatalogado. **No recomendable**

Medidor de pH del suelo 3 en 1

Descripción:

Científico preciso: fácil y preciso de leer los niveles de humedad, pH y luz, promueve la salud de las plantas.

Rango de humedad: 1-10 (1-3 seco, 4-7 ni, 8-10 mojado); Luz correspondiente: 0-2000 Lux (0-200 bajo, 200-500LOW +, 500-1000 Ni 1000-2000 HGH); Rango de pH: 3,5-8 pH (3,5-6,5 ácido, 7 in, 7-8 alcalino).

Solo necesitas enchufar y leer. Es muy fácil de usar. Si las plantas están muy secas, el puntero del medidor de tierra no rotará. Significa que tus plantas necesitan agua en este momento.

Compacto para uso en interiores o exteriores: saber cuándo regar, ajustar el pH o cambiar la iluminación de su jardín, flores, plantas tanto en interiores como en exteriores.

Mide la humedad a nivel de la raíz. No requiere batería, es fácil y fácil de usar. Inserta el medidor en el suelo, cambia a la configuración que quieras medir y lee la escala.

Especificaciones:

Material: aleación de aluminio, plástico.

Tamaño: 50x38x290mm.

Longitud de la sonda: 20,2 cm.

Color: Como se muestra en las imágenes.

Rango de PH: 3,5-8,0 (3,5-6,5: ácido, 7: Ni, 7-8: alcalino).

Rango de humedad: 1-10 (1-3: seco, 4-7: Ni, 8-10: mojado).

Rango de luz: 0-2000.

Notas:

Debido a la luz y a la diferencia de ajuste de la pantalla, el color del artículo puede ser ligeramente diferente al de las imágenes.

Permite una ligera diferencia de tamaño debido a las diferentes mediciones manuales.

Para evitar dañar el electrodo, no utilizar en el suelo con piedras e insertar a la fuerza, limpiar el electrodo después de cada uso.

El paquete incluye:

Medidor de suelo 1*3 en 1.

Hay que modificarlo para quedarse solamente con el sensor y descartar la parte de la medición. **Descartado**

SEN0249

Specification

Spear Tip pH Probe

- Measuring Range: 0~10pH
- Accuracy: $\pm 0.1\text{pH}$
- Operating Temperature: 5~60 °C
- Response Time : ≤2min(in standard buffer solution)
- Salt Bridge Material: Poly Tetra Fluoro Ethylene (PTFE)
- Shell Material: Polyoxymethylene (POM)
- Filling Solution: Can NOT be Filled
- Wiring Connector: BNC
- Wire Length: 850mm (BNC connector included)

Signal Transmitter Board

- Input Power : 5.00V

https://www.dfrobot.com/wiki/index.php?title=Gravity:_Analog_Spear_Tip_pH_Sensor___Meter_Kit___For_Soil_Ar



- Output Channel: A/D
- Output signal: V = 4.0V
- Accuracy : < $\pm 0.2\text{pH}$ (25 °C)
- Probe Connector: BNC
- Module Connector: Gravity:PH2.0-3Pin
- Dimension: 43mm×32mm(1.69inch*1.26inch)

Este sensor tiene buenas características, el único fallo es el rango de temperaturas que no lo cumple. **Recomendable**

Humedad del suelo

Los rangos de medición deben comprender entre el 0 % al 100 % de humedad para tomar adecuadamente las medidas. Es recomendable que puedan trabajar en las condiciones de temperatura y humedad ambiente vistas con anterioridad. Es relevante ver precio y tiempo de vida de estos sensores.

101990792

Soil Temperature	
Range	-30 °C to +70 °C
Accuracy	±0.5 °C
Resolution	0.1 °C
Soil Moisture	
Range	From completely dry to fully saturated (from 0% to 100% of saturation)
Accuracy	±2% (0 to 50 %) ±3% (50 to 100 %)
Resolution	0.03%(0 to 50 %); 1%(50 to 100 %)
General Parameters	
Product Model	LoRa-S-868/915/AU915/923-Soil MT-01
Microcontroller	Ultra-low-power MCU
Support Protocol	Based on LoRaWAN v1.0.2 protocol
LoRa Channel Plan	EU868 / US915 / AU915 / AS923
LoRa Power Output	16 dBm (EIRP)
Sensitivity	EU868: -137.5dBm(SF12, BW125KHz) US915/AU915/AS923: -136.5dBm(SF12, BW125KHz)
Current Consumption	5 µA (sleep mode) 120 mA max(active mode)
Communication Distance	2 to 10 km (depending on different antennas and environments)
Measuring Area	A cylinder area (with the probe as the center, diameter: 7cm, height: 7cm)
Battery Life	≥ 3 year (upload data once per hour)
Battery Voltage	3.6V
Battery Capacity	19Ah (Non-rechargeable)
IP Rating	IP66
UV Resistance	anti-aging (from rain/sun exposure): UL746C F1
Enclosure Material	PC
Operating Temperature	-30 °C to +70 °C
Operating Humidity	0 to 100 %RH (non-condensing)
Device Weight	415g

Cumple características de temperatura y humedad, tiene buenas características. Este está ya diseñado para una red LoRa. **Recomendable**

314990620

General Parameters	
Product Model	S-Soil MT-02A
Interface	RS-485
Protocol	MODBUS-RTU RS485
Power Supply	3.6 ~ 30V DC
Current Consumption	Max 40mA@24V DC
Operating Temperature	-40 °C to +85 °C
Storage Temperature	-40 °C to +85 °C
Response Time	Less than 1 second
Measuring Area	A cylinder area (with the probe as the center, diameter: 7cm, height: 7cm)
The material of the probe	Food grade stainless steel
Sealing Material	The black flame retardant epoxy resin
IP Rating	IP68
Cable Length	5 meters
Installation	All buried or probe into all of the measured medium
Device Weight	270g

Soil Temperature	
Range	-40 °C to +80 °C
Accuracy	±0.5 °C
Resolution	0.1 °C

Soil Moisture	
Range	From completely dry to fully saturated (from 0% to 100% of saturation)
Accuracy	±3% (0~53%) ±5% (53~100%)
Resolution	1 %

Wiring Diagram	

Buena alimentación, buen rango de temperatura. Tiene buenas características, pero peores que el anterior. Mide también temperatura del suelo. **El recomendado**

PIM520

TIPO	DESCRIPCIÓN	SELECCIONAR
Categoría	Sensores y transductores Humedad, sensores de humedad	<input type="radio"/>
Fabricante	Pimoroni Ltd	<input type="checkbox"/>
Serie	-	<input type="checkbox"/>
Paquete	Granel ?	<input type="checkbox"/>
Estado del producto	Activo	<input type="checkbox"/>
Tipo de sensor	Humedad	<input type="checkbox"/>
Rango de humedad	-	<input type="checkbox"/>
Tipo de salida	frecuencia	<input type="checkbox"/>
Salida	-	<input type="checkbox"/>
Precisión	-	<input type="checkbox"/>
Sensibilidad	-	<input type="checkbox"/>
Voltaje de la fuente	-	<input type="checkbox"/>
Tipo de montaje	Definido por el usuario	<input type="checkbox"/>
Temperatura de funcionamiento	-	<input type="checkbox"/>
Paquete / Caja (carcasa)	Módulo	<input type="checkbox"/>
Paquete del dispositivo del proveedor	-	<input type="checkbox"/>

No tiene características, no se puede enchufar. Pero es muy bueno si quieras cuidar una planta. **No recomendable en este trabajo, pero sí en tu casa**

28092

Operating voltage	2.0V-5.0V
Output type	Analog output
Detectable depth	38mm
Dimensions	20.0mm*51.0mm
Fixing hole size	2.0mm

No es un sensor profesional, sino que sirve para iniciarse en el estudio.

No recomendable

Estaciones meteorológicas

El problema es que al no tener catálogo hay que fiarse de lo que dice el anunciante. En este caso es más recomendable las Sainlogic, ya que proporcionan funcionalidades para conectarlos a dispositivos como la Raspberry Pi 4.

Estación meteorológica WLAN Sainlogic

Acerca de este producto

- **【WLAN】**: el sensor está conectado a la consola a través de RF (915 MHz) y la consola está conectada a su router (2,4 GHz). Verifique las condiciones climáticas en tiempo real, los datos históricos y las advertencias en su teléfono inteligente, tablet, laptop o computadora de escritorio.
- **【WEATHER UNDERGROUND】**: con la opción de conexión WLAN extendida, su estación puede transmitir de forma inalámbrica sus datos a la red de estaciones meteorológicas personales más grande del mundo, Weather Underground. Experimente la conveniencia de llevar su información meteorológica personal con usted mientras viaja.
- **【PANTALLA GRANDE A COLOR】**: La pantalla grande a color permite ver todas las funciones y las informaciones importantes. Las áreas separadas de manera diferente favorecen aun más la visibilidad de la pantalla. La pantalla LCD de alta calidad se ilumina continuamente y se puede configurar en casi cualquier espacio para ahorrar espacio. Controle fácilmente las condiciones climáticas en su casa, en el jardín y sus alrededores con la pantalla brillante LCD a color fácil de leer. Obtenga fácilmente temperatura ; más información.
- **【SENSOR AL AIRE LIBRE】**: El sensor para exteriores le muestra información actual sobre la temperatura, la humedad del aire, la velocidad y la dirección del viento, así como la cantidad de precipitación y la radiación UV.
- **【PRONÓSTICO DEL TIEMPO EXACTO】**: Experimente la comodidad de su información meteorológica personal. La estación meteorológica mide la velocidad del viento, la dirección del viento, la precipitación, la temperatura exterior, la luz solar y la radiación UV, y ofrece un pronóstico del tiempo. Una función de alarma con función de repetición también es parte del rango de funciones.
- **【FUNCIÓN DE ALARMA】**: puede configurar fácilmente una alarma para que siempre se le avise si las temperaturas alcanzan un área crítica.



- Método de control/tipo de controlador: Control remoto
- Protocolo de conectividad: Radiofrecuencia
- Tecnología de conectividad: Wifi
- Tipo de enchufe: No se requiere
- Dispositivos compatibles: Teléfono inteligente
- Actualización de software compatible: 2021.9
- Requisitos mínimos del sistema: No se requiere
- Idioma del manual: Inglés, Alemán
- Mostrar opciones de idioma: Inglés
- Aplicaciones compatibles: Sitio web del servidor meteorológico
- Sistema operativo: No se requiere
- Velocidad del procesador: No se requiere
- Fabricante tarjeta gráfica: No se requiere
- Plataforma de hardware: No se requiere
- Servicios Internet compatibles: Clima subterráneo, clima en la nube (Weather Undergound, Weather Cloud)

Tiene las magnitudes que se desean medir, además de ser de Sainlogic.

Recomendable

Bresser Wetterstation Funk

Acerca de este producto

- Estación meteorológica radio controlada con multisensor 5 en 1
- Medida de temperatura, velocidad del aire, presión atmosférica,
- Humedad ambiental, probabilidad de precipitación, memoria histórica
- 120x190x22 mm; 310g y 344x394x136 mm; 682 g
- Incluido: Estación base, multisensor y material de montaje

Detalles del producto

Pilas : 6 AA (Tipo de pila necesaria)

Dimensiones del producto : 38.8 x 15 x 28 cm; 992 gramos

Producto en Amazon.es desde : 28 junio 2018

Fabricante : Bresser

ASIN : B07DW23KWF

Número de modelo del producto : 7002511

País de origen : China

Clasificación en los más vendidos de Amazon: nº125 en Jardín (Ver el Top 100 en Jardín)
nº1 en Estaciones meteorológicas para exterior

Opiniones de los clientes: ★★★★☆ ~ 600 valoraciones

Al no ser Sainlogic no tiene la capacidad de comunicarse de forma tan eficiente con la Raspbeery Pi 4.

No recomendable

Estación meteorológica Sainlogic profi WLAN

Acerca de este producto

- **【WIFI + Weather Underground】** Con la opción avanzada de conexión Wi-Fi, su estación meteorológica puede transferir de forma inalámbrica sus datos a la red personal más grande del mundo weather underground. Compruebe las condiciones meteorológicas en tiempo real, los datos históricos y las advertencias en sus teléfonos inteligentes, laptops, tablets o computadoras de escritorio.
- **【Pantalla grande a color】** La pantalla grande a color puede mostrar todas las funciones e información importantes. Las áreas separadas permiten una visibilidad que se ha mostrado claramente.
- **【Sistema de transferencias inalámbricas】** La estación meteorológica ofrece hasta 8 canales diferentes, lo que significa que puede conectar hasta 8 sensores adicionales para monitorizar la información meteorológica desde varios lugares.
- **【SENSOR PARA EXTERIORES】** Los sensores para exteriores muestran información actual sobre la temperatura, la humedad del aire, la presión del aire, la velocidad y la dirección del viento así como la precipitación y la radiación ultravioleta, incluidos los paneles solares.
- **【Pronóstico Meteorológico Preciso】** Experimente la cómoda información meteorológica personal. La estación meteorológica mide la velocidad del viento, la dirección del viento, la precipitación, la temperatura exterior, la radiación solar y la radiación ultravioleta, y proporciona pronóstico del tiempo y fase lunar.
- **01.** Cuando los clientes usan el producto para medir la velocidad del viento, deben evitar obstáculos como edificios, árboles, vehículos, líneas de alto voltaje, etc., para no afectar la precisión de la medición. Solicite a los clientes que se comuniquen con nosotros directamente. Les enviaremos el reemplazo.
- **02.** Nuestros productos pueden alcanzar distancias de comunicación por radio de hasta 300 pies entre el receptor y el transmisor en campo abierto sin interferir con obstáculos como edificios, árboles, vehículos, líneas de alto voltaje, etc. Las señales inalámbricas no penetran en los edificios metálicos. La mayoría de las aplicaciones están limitadas a 100 pies debido a obstrucciones de edificios, paredes e interferencias.
- **03.** Compruebe si la red está conectada, si hay interferencias alrededor del sensor y si la distancia está demasiado lejos del límite.



Especificaciones:

Método de control:	Control remoto	Requisitos mínimos sistema:	No es necesario
Protocolo de conectividad:	Radiofrecuencia	Idioma del manual:	Inglés, alemán
Tecnología de conectividad:	Wifi	Opciones de idioma:	Inglés
Tipo de enchufe:	No requerido	Aplicaciones compatibles:	Sitios web de servidores meteorológicos
Dispositivos compatibles:	Teléfono inteligente	Servicios de Internet:	Weather Underground, Weather Cloud
Actualización de software:	2021.9		

Tiene las magnitudes que se desean medir, además de ser de Sainlogic.
Recomendable

Sainlogic WS3500

Acerca de este producto

- Con esta estación meteorológica profesional, puede controlar las condiciones climáticas en el hogar o en el jardín a través de una pantalla LCD a color clara y legible.
- La conexión WiFi permite que su estación meteorológica transmita los datos meteorológicos a la estación meteorológica personal más grande del mundo, WeatherUnderground.com. Puede consultar los datos actuales e históricos en cualquier lugar y en cualquier momento.
- La alarma del reloj es automática y se sincroniza con la red en tiempo real. Puede configurar fácilmente la alarma para que se active cuando la temperatura alcance un área crítica.
- Esta estación meteorológica puede medir la velocidad del viento, la dirección del viento, la precipitación, la temperatura y humedad exterior, la radiación solar y la radiación ultravioleta.
- La pantalla muestra temperatura, humedad y presión del aire. Además, calcula la temperatura del punto de condensación, el índice de calor, con varios colores.

Detalles del producto

Pilas : 2 AA (Tipo de pila necesaria)

Is Discontinued By Manufacturer : No

Dimensiones del producto : 32 x 15 x 37 cm; 1 gramos

Producto en Amazon.es desde : 15 abril 2018

Fabricante : Sainlogic

ASIN : B07F2GQDW2

Número de modelo del producto : WS3500

País de origen : China

Departamento : Men's

Clasificación en los más vendidos de Amazon: nº16,901 en Jardín (Ver el Top 100 en Jardín)

nº108 en Estaciones meteorológicas para exterior

Opiniones de los clientes: ★★★★★ 707 valoraciones



Método de control/tipo de controlador: Control remoto

Protocolo de conectividad: Radiofrecuencia

Tecnología de conectividad: Wifi

Tipo de enchufe: No se requiere

Dispositivos compatibles: Teléfono inteligente

Actualización de software compatible: 2021.9

Requisitos mínimos del sistema: No se requiere

Idioma del manual: Inglés, Alemán

Mostrar opciones de idioma: Inglés

Aplicaciones compatibles: Sitio web del servidor meteorológico

Sistema operativo: No se requiere

Velocidad del procesador: No se requiere

Fabricante tarjeta gráfica: No se requiere

Plataforma de hardware: No se requiere

Servicios Internet compatibles: Clima subterráneo, clima en la nube (Weather Undergound, Weather Cloud)

Tiene las magnitudes que se desean medir, además de ser de Sainlogic.

Recomendable

Estación meteorológica inalámbrica froggit WH6000 Pro

Acerca de este producto

- La estación meteorológica inalámbrica WH6000 PRO de Froggit convence no solo por sus materiales de alta calidad, sino sobre todo por sus valores de medición precisos y la instalación rápida y sencilla.
- Gran pantalla LCD a color de 8 pulgadas con retroiluminación permanente y continua. Color negro brillante con acabado de laca de piano.
- Carga de datos meteorológicos (datos exteriores) a un servidor meteorológico Wunderground y WeatherCloud para la observación mundial del propio microclima.
- Unidad de exterior multifunción 7 en 1 con soporte solar (temperatura, humedad, precipitación, dirección del viento, velocidad del viento, rayos UV, intensidad de luz). Ventilador integrado para reducir el calor solar.
- Sensor interior externo para medir la temperatura interior y la humedad del aire para el interior. Ampliable: hasta un total de 7 sensores de radio termo-higrómetro (se venden por separado).
- Pantalla clara de todos los valores medidos, además de función de despertador/alarma, alarma de heladas/hielo, fases lunares, visualización exacta de hora y fecha vía Internet, sincronización de tiempo, índice de calor, previsión meteorológica gráfica de 12 horas, indicador de temperatura ambiente mediante símbolo.
- Incluye: 1 consola de pantalla WH6000 Pro, incluye fuente de alimentación, 1 sensor interior WH6000 Pro, 7 en 1, sensor exterior combinado WH6000 PRO.

Especificaciones para este producto

EAN	0046382351996
Nombre de la marca	froggit
Número de pieza	650
Tipo de fuente de alimentación	Energía solar , Batería
Tipo de pantalla	LCD
UPC	046382351996

Al no ser Sainlogic no tiene la capacidad de comunicarse de forma tan eficiente con la Raspberry Pi 4.

No recomendable

Froggit WH3600 Funk

Acerca de este producto

- El WH3600 combina la última tecnología de medición meteorológica con la última tecnología. Gracias al módulo WiFi integrado, tus datos meteorológicos locales serán globales, ya que puedes cargar en tiempo real tus datos meteorológicos en servidores meteorológicos, como Weather Underground, Ecowitt, Weathercloud. Gracias a la conexión a Internet tendrás una visión general de tus datos meteorológicos en todo el mundo.
- No necesitas un ordenador, ya que todos los datos se muestran directamente en la pantalla a todo color. Nueva aplicación de configuración: WS-View, con la que puedes configurar la estación meteorológica paso a paso. ¡No puede ser más fácil!
- La unidad exterior en forma de Y mide la temperatura exterior, la humedad relativa, la radiación solar, el valor UV, la precipitación, la velocidad del viento y la dirección del viento (cálculo actualizado cada 16 segundos) y envía todos los datos de medición a la unidad de visualización.
- La unidad de pantalla tiene un termo-higrómetro incorporado, con el que se mide permanentemente la temperatura interior, la humedad relativa y la presión del aire. Todos los datos recopilados se evalúan automáticamente y te muestran la previsión meteorológica, la tasa de lluvia, el punto de rocío, así como las tendencias de la presión del aire, la temperatura interior y la humedad.
- Incluye: Unidad de visualización WH3600, unidad exterior inalámbrica, niveles de viento, adaptador de 5 V CC, tornillo de arco con pinzas de fijación, manual de usuario en alemán/inglés (idioma español no garantizado).

Al no ser Sainlogic no tiene la capacidad de comunicarse de forma tan eficiente con la Raspberry Pi 4.

No recomendable

Froggit WH3000 SE

Acerca de este producto

- Nuestro nuevo miembro de las filas de estaciones meteorológicas WiFi. El WH3000 SE combina la última tecnología de medición de tiempo con la última tecnología. El módulo WiFi integrado Sus datos meteorológicos locales son globales, como se puede cargar sus datos del tiempo Weather Server como el tiempo en tiempo real. Gracias a la conexión a Internet que tiene por lo tanto una visión general de la información sobre el clima global.
- No es necesario un PC, ya que se muestran todos los datos directamente en la pantalla a todo color. Además, ahora hay una aplicación de configuración con la que se puede ajustar el paso de estación meteorológica a paso. Es así de fácil!
- El novedoso diseñado unidad de Y-exterior mide la temperatura exterior, humedad relativa, radiación solar, el valor UV, la precipitación y la velocidad del viento, dirección del viento (cálculo actualizado cada 16 seg.) Y envía todos los datos a la unidad de visualización.
- La unidad de visualización tiene una termo-higrómetro construido, con el permanente la temperatura interna, se mide la humedad relativa y la presión de aire. Todos los datos recogidos serán analizados automáticamente y le dan un pronóstico de un tiempo, la intensidad de la lluvia, el punto de rocío, así como las tendencias de la presión del aire, la temperatura y la humedad interna.
- "Todo en uno" unidad de radio al aire libre con integrada termo-higrómetro en el nuevo Y-diseño (temperatura, humedad, velocidad del viento, dirección del viento, precipitación, UV y la luz del sol) - Módulo incorporado WiFi en la unidad de visualización - Android / iOS para dispositivos WiFi fácil y la sumisión al servidor de tiempo (Wunderground) - transmisión de datos en tiempo real al servidor de tiempo (Wunderground)

Al no ser Sainlogic no tiene la capacidad de comunicarse de forma tan eficiente con la Raspbeery Pi 4.

No recomendable

Apéndice C: Códigos de Matlab de las alternativas

En este apéndice se recogen los códigos de Matlab creados para obtener los valores de las ganancias, offset y filtrados para el acondicionamiento de los sensores de temperatura estomática, flujo de savia y humedad de hoja y de suelo.

Se verán cuatro tipos de código:

- **El primer tipo** es para calcular la ganancia y el offset en un amplificador de instrumentación en función del rango de salida del sensor y el rango de entrada que le quieras introducir al Arduino. Para exemplificarlo, a lo mejor se quiere que el rango de entrada que recibe el Arduino sea menor que el rango de medición propio del sensor, al ser los valores extremos casos extraordinarios y que no se dan con frecuencia.
- **El segundo tipo** idéntico que el anterior, pero haciendo uso de una ganancia negativa.
- **El tercer tipo** de código refleja lo mismo que el tipo anterior, pero para amplificadores diferenciales (montados con amplificadores operacionales).
- **El cuarto tipo** es para calcular los valores de las resistencias y condensadores para obtener la ganancia requerida y el filtrado de 10 Hz estipulado.

LAT-B3

```
%% Amplificador de instrumentación
syms G Vref;
rangorimin = -25; % Rango mínimo del sensor
rangorimax = 70; % Rango máximo del sensor
rangfinmin = -10; % Rango mínimo que se quiere medir
rangfinmax = 55; % Rango máximo que se quiere medir
Vorimin = 0; % Tensión mínima a la salida del sensor
Vorimax = 2.5; % Tensión máxima a la salida del sensor
Vfinmin = 0; % Tensión mínima a la salida del AI
Vfinmax = 5; % Tensión máxima a la salida del AI

% Cálculo de las tensiones de entrada al AI
Vinsmin = (Vorimax-Vorimin) / (rangorimax-rangorimin) * (rangfinmin-rangorimin);
Vinsmax = (Vorimax-Vorimin) / (rangorimax-rangorimin) * (rangfinmax-rangorimin);

% Obtención de la G y Vref
ec1 = G*Vinsmin+Vref-Vfinmin;
ec2 = G*Vinsmax+Vref-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```

%% Inversor
syms G Vref;
rangorimin = -25; % Rango mínimo del sensor
rangorimax = 70; % Rango máximo del sensor
rangfinmin = -10; % Rango mínimo que se quiere medir
rangfinmax = 55; % Rango máximo que se quiere medir
Vorimin = 0; % Tensión mínima a la salida del sensor
Vorimax = 2.5; % Tensión máxima a la salida del sensor
Vfinmin = 0; % Tensión mínima a la salida del AI
Vfinmax = -5; % Tensión máxima a la salida del AI

% Cálculo de las tensiones de entrada al AI
Vinsmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-rangorimin);
Vinsmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-rangorimin);

% Obtención de la G y Vref
ec1 = G*Vinsmin+Vref-Vfinmin;
ec2 = G*Vinsmax+Vref-Vfinmax;
[G, Vref] = solve(ec1,ec2)

```

```

%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = -25; % Rango mínimo del sensor
rangorimax = 70; % Rango máximo del sensor
rangfinmin = -15.5; % Rango mínimo que se quiere medir
rangfinmax = 60.5; % Rango máximo que se quiere medir
Vorimin = 0; % Tensión mínima a la salida del sensor
Vorimax = 2.5; % Tensión máxima a la salida del sensor
Vfinmin = 0; % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5; % Tensión máxima a la salida del amplificador
diferencial

% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)

```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 20*10^3; % La resistencia uno va a tener que ser prefijada
fc = 10;       % Frecuencia de corte en hercios para el filtrado paso
bajo
R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2) / (R1+R2)))
```

LT-1M

```
%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = 0;      % Rango mínimo del sensor
rangorimax = 50;     % Rango máximo del sensor
rangfinmin = 0;       % Rango mínimo que se quiere medir
rangfinmax = 50;     % Rango máximo que se quiere medir
Vorimin = 0;          % Tensión mínima a la salida del sensor
Vorimax = 2;          % Tensión máxima a la salida del sensor
Vfinmin = 0;          % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5;          % Tensión máxima a la salida del amplificador
diferencial

% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 1*10^3;    % La resistencia uno va a tener que ser prefijada
fc = 10;         % Frecuencia de corte en hercios para el filtrado
paso bajo

R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2) / (R1+R2)))

%% Circuito no inversor LT-1M
R = 1*10^3;      % Resistencia del filtro paso bajo
R1 = 1*10^3;     % La resistencia una va a tener que ser prefijada
fc = 10;          % Frecuencia de corte en hercios para el filtrado
paso bajo

C = 1/(2*pi*fc*R)
R2 = (G-1) * R1
```

LWS-L

```
%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = 0;           % Rango mínimo del sensor
rangorimax = 100;          % Rango máximo del sensor
rangfinmin = 0;            % Rango mínimo que se quiere medir
rangfinmax = 100;          % Rango máximo que se quiere medir
Vorimin = 0;               % Tensión mínima a la salida del sensor
Vorimax = 1.25;             % Tensión máxima a la salida del sensor
Vfinmin = 0;                % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5;                 % Tensión máxima a la salida del amplificador
diferencial

% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 1.25*10^3; % La resistencia uno va a tener que ser prefijada
fc = 10;          % Frecuencia de corte en hercios para el filtrado
paso bajo

R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2)/(R1+R2)))
```

PYTHOS31

```
%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = 0;           % Rango mínimo del sensor
rangorimax = 100;          % Rango máximo del sensor
rangfinmin = 0;            % Rango mínimo que se quiere medir
rangfinmax = 100;          % Rango máximo que se quiere medir
Vorimin = 0;               % Tensión mínima a la salida del sensor
Vorimax = 1.25;             % Tensión máxima a la salida del sensor
Vfinmin = 0;                % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5;                 % Tensión máxima a la salida del amplificador
diferencial
```

```
% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 1.25*10^3; % La resistencia uno va a tener que ser prefijada
fc = 10; % Frecuencia de corte en hercios para el filtrado
paso bajo

R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2)/(R1+R2)))
```

314990620

```
%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = 0; % Rango mínimo del sensor
rangorimax = 100; % Rango máximo del sensor
rangfinmin = 0; % Rango mínimo que se quiere medir
rangfinmax = 100; % Rango máximo que se quiere medir
Vorimin = 0; % Tensión mínima a la salida del sensor
Vorimax = 2; % Tensión máxima a la salida del sensor
Vfinmin = 0; % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5; % Tensión máxima a la salida del amplificador
diferencial

% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 1*10^3; % La resistencia uno va a tener que ser prefijada
fc = 10; % Frecuencia de corte en hercios para el filtrado
paso bajo
```

```
R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2)/(R1+R2)))
```

SF-G

```
%% SF-G
syms G Vref;
Vorimin = 100*10^-6; % Tensión mínima a la salida del sensor
Vorimax = 800*10^-6; % Tensión máxima a la salida del sensor
Vfinmin = 0;           % Tensión mínima a la salida del AI
Vfinmax = 5;           % Tensión máxima a la salida del AI

% Obtención de la G y Vref
ec1 = G*Vorimin+Vref-Vfinmin;
ec2 = G*Vorimax+Vref-Vfinmax;
[G, Vref] = solve(ec1,ec2)

Rg = 24200/(G-1)
```

```
%% Inversor
syms G Vref;
rangorimin = 2.5;    % Rango mínimo del sensor
rangorimax = 20;     % Rango máximo del sensor
rangfinmin = 2.5;    % Rango mínimo que se quiere medir
rangfinmax = 20;     % Rango máximo que se quiere medir
Vorimin = 100*10^-6; % Tensión mínima a la salida del sensor
Vorimax = 800*10^-6; % Tensión máxima a la salida del sensor
Vfinmin = 0;          % Tensión mínima a la salida del AI
Vfinmax = -5;         % Tensión máxima a la salida del AI

% Cálculo de las tensiones de entrada al AI
Vinsmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vinsmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*Vinsmin+Vref-Vfinmin;
ec2 = G*Vinsmax+Vref-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Circuito diferencial
clear all
clc

syms G Vref;
rangorimin = -25;    % Rango mínimo del sensor
rangorimax = 70;     % Rango máximo del sensor
rangfinmin = -15.5;  % Rango mínimo que se quiere medir
rangfinmax = 60.5;   % Rango máximo que se quiere medir
Vorimin = 0;          % Tensión mínima a la salida del sensor
Vorimax = 2.5;         % Tensión máxima a la salida del sensor
```

```
Vfinmin = 0; % Tensión mínima a la salida del amplificador
diferencial
Vfinmax = 5; % Tensión máxima a la salida del amplificador
diferencial

% Cálculo de las tensiones de salida de los sensores
Vdifmin = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmin-
rangorimin);
Vdifmax = (Vorimax-Vorimin)/(rangorimax-rangorimin)*(rangfinmax-
rangorimin);

% Obtención de la G y Vref
ec1 = G*(Vdifmin-Vref)-Vfinmin;
ec2 = G*(Vdifmax-Vref)-Vfinmax;
[G, Vref] = solve(ec1,ec2)
```

```
%% Cálculo de resistencias y condensadores para ganancia y filtrado
R1 = 20*10^3;% La resistencia uno va a tener que ser prefijada
fc = 10; % Frecuencia de corte en hercios para el filtrado paso
bajo

R2 = G * R1
C1 = 1/(2*pi*fc*((R1*R2) / (R1+R2)))
```

Apéndice D: Simulación de las alternativas de acondicionamiento

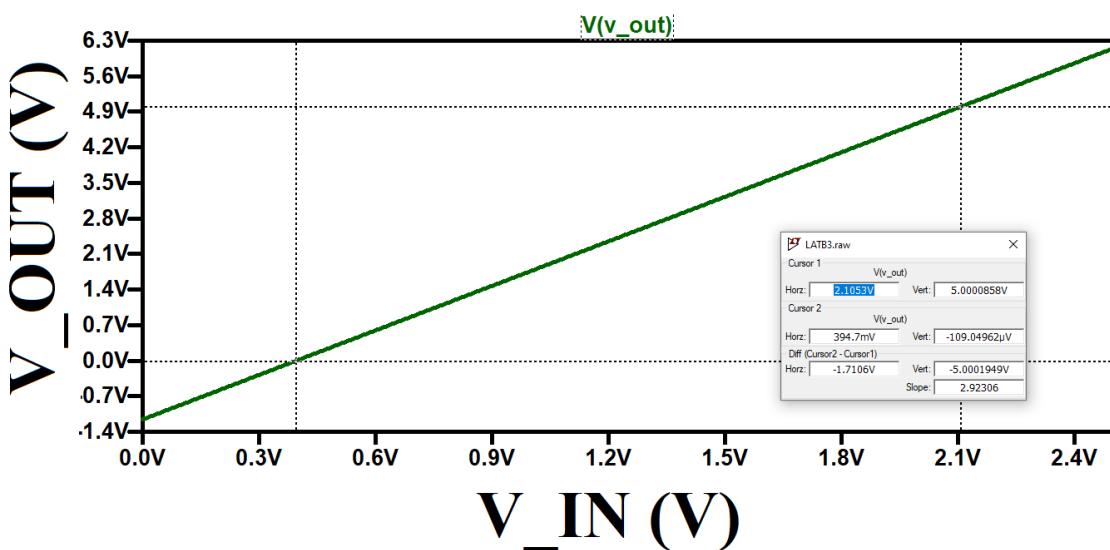
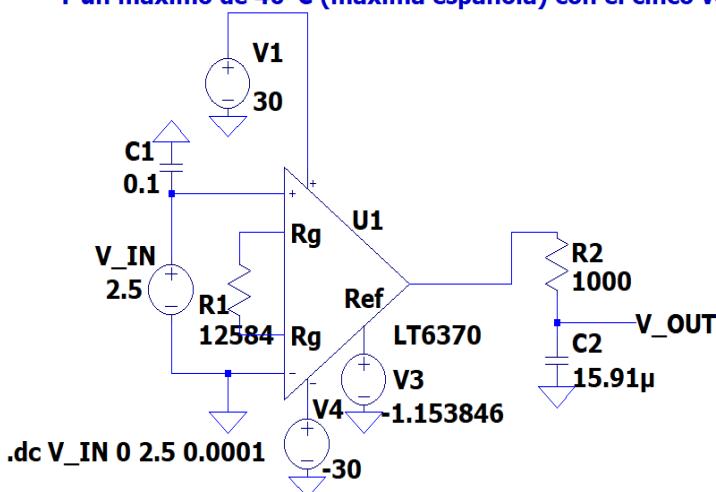
En este apéndice se recogen tanto los circuitos de acondicionamiento realizados para distintos sensores usados y no usados como distintos diseños que al final no se utilizaron. Todo esto para mostrar otras posibles configuraciones que podrían ser de utilidad para otros diseños o trabajos futuros.

Acondicionamiento de los sensores de temperatura de estoma

LAT-B3

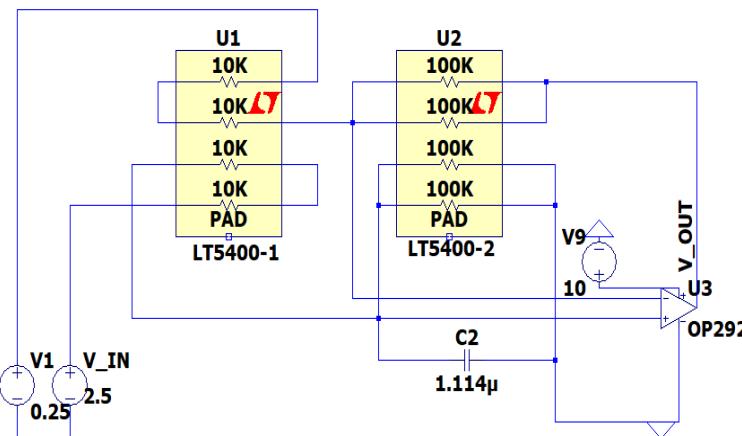
Amplificador operacional

Círcuito de acondicionamiento LATB3 con amplificador de instrumentación (referencia negativa)
Se corresponde con un rango de -6°C (mínima española) con el cero en tensión
Y un máximo de 40°C (máxima española) con el cinco voltios en tensión

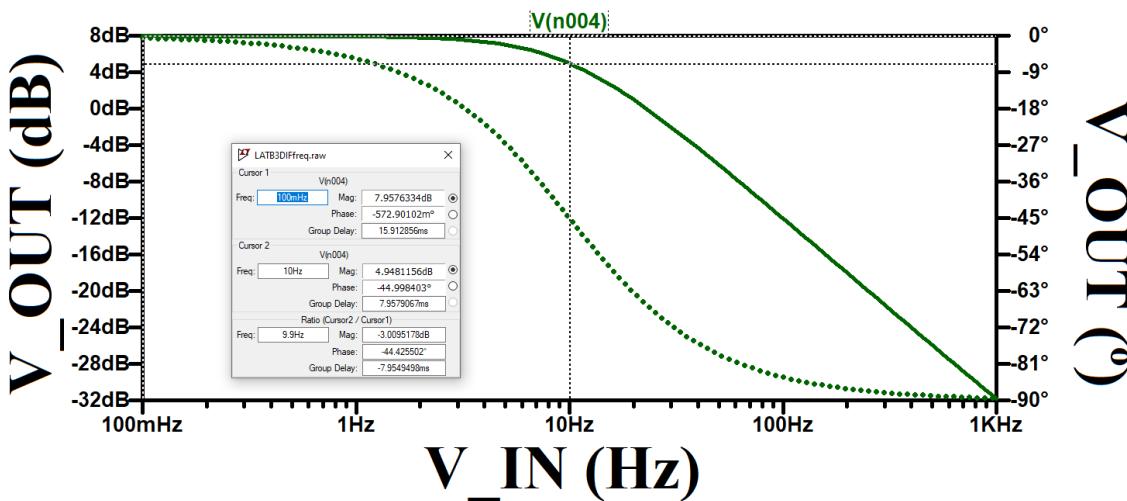
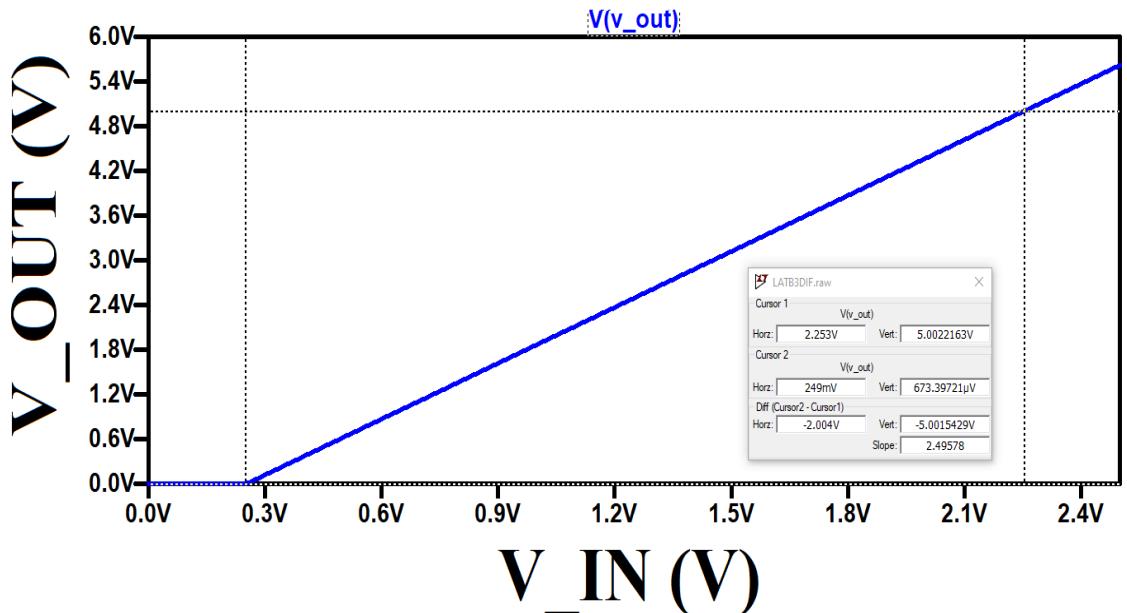


Amplificador diferencial

Círcuito de acondicionamiento del sensor LATB3 con amplificadores operacionales (filtrado y utilizando referencia)
El rango de salida tiene un mínimo de 0V cuando hay -15.5°C y en máximo de 5V cuando hay 60.5°C



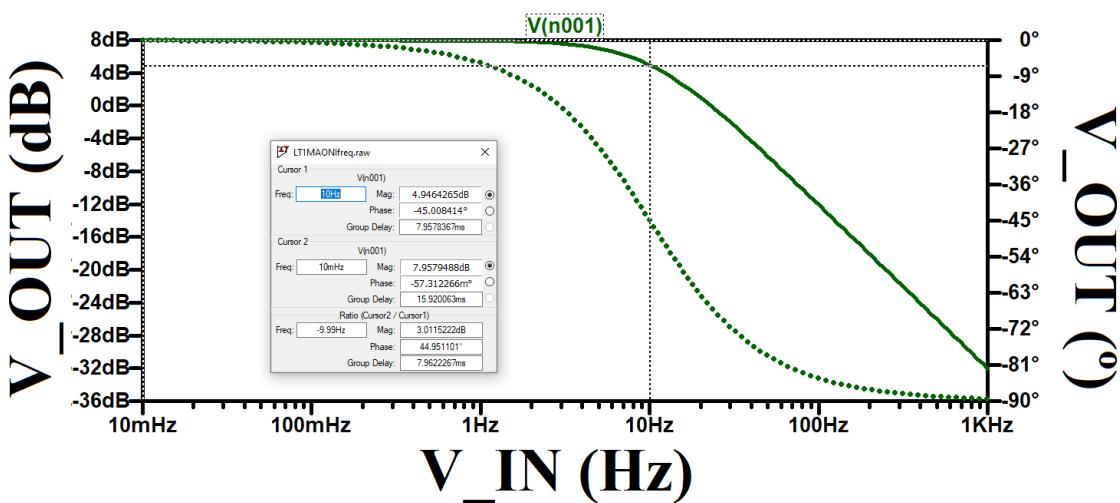
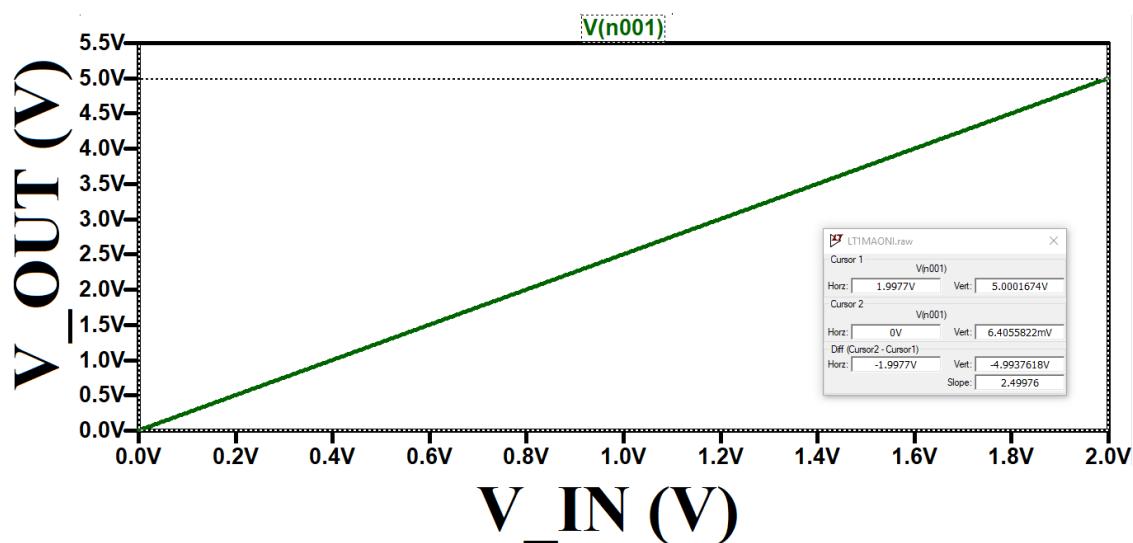
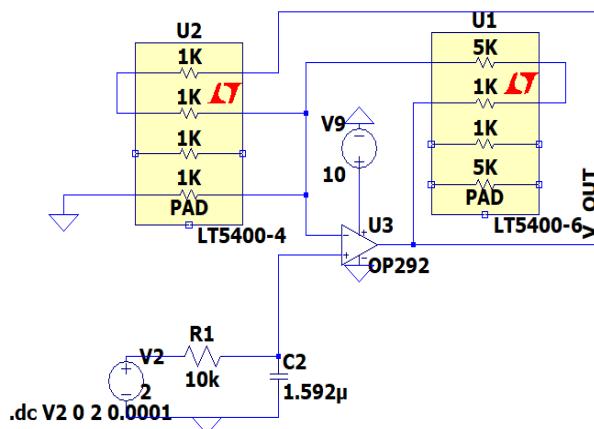
.dc V_IN 0 2.5 0.001 El cálculo del condensador es el paralelo de las resistencias 2000 y 5000 y 10 Hz



LT-1M

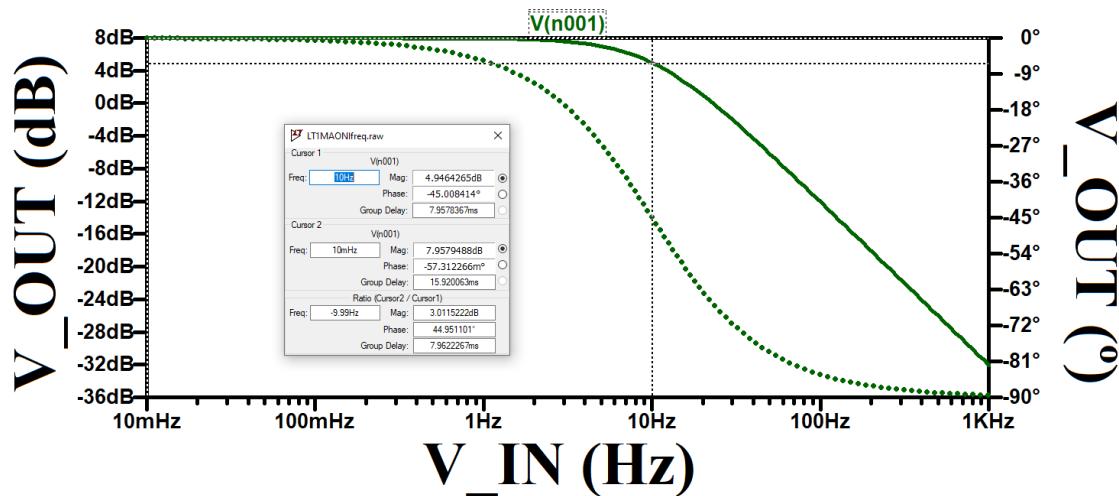
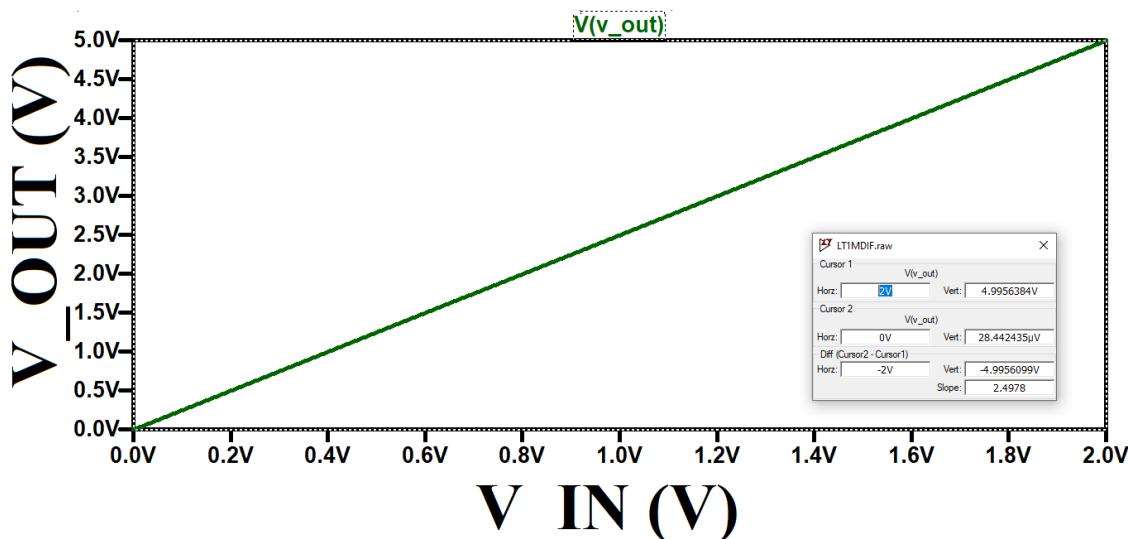
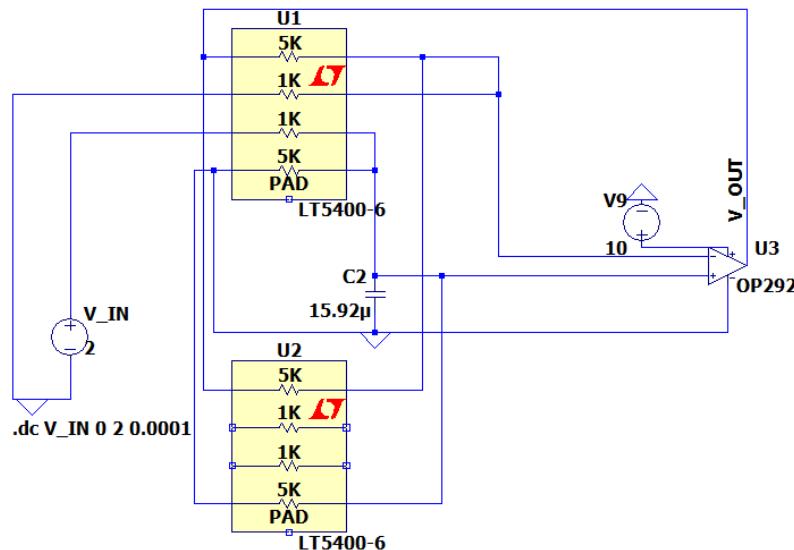
Amplificador operacional no inverter

Circuito de acondicionamiento del sensor LT-1M con amplificador operacional no inverter (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0° y en máximo de 5V cuando hay 50°



Amplificador diferencial

Circuito de acondicionamiento del sensor LT-1M con amplificador diferencial (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0° y en máximo de 5V cuando hay 50°



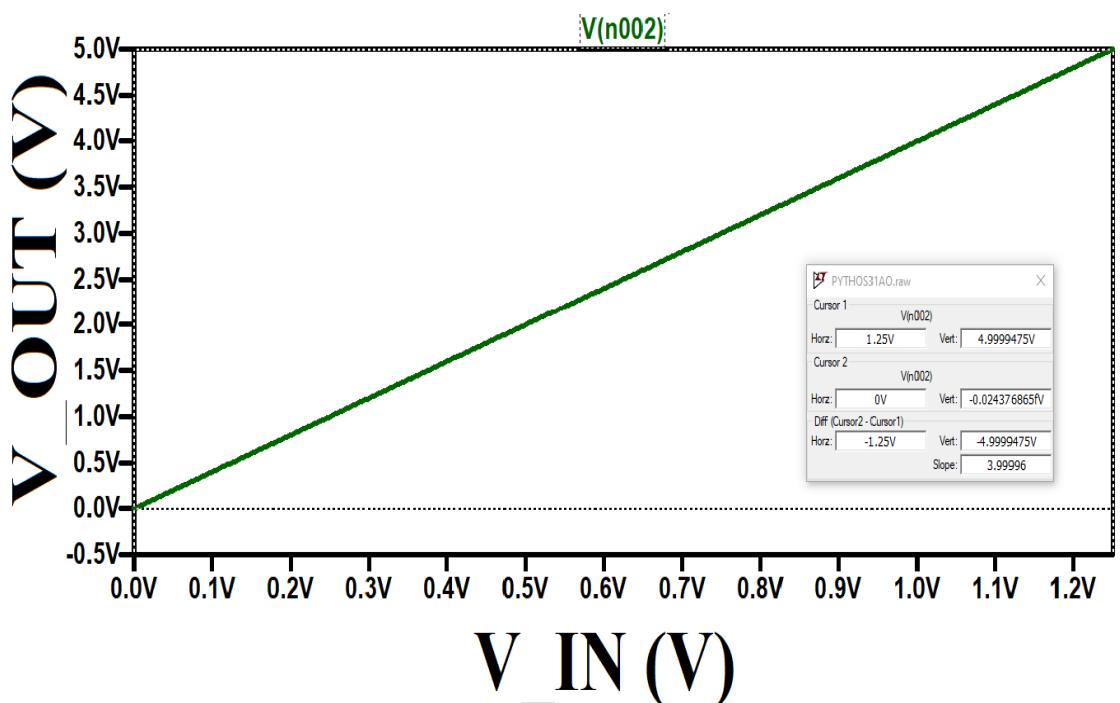
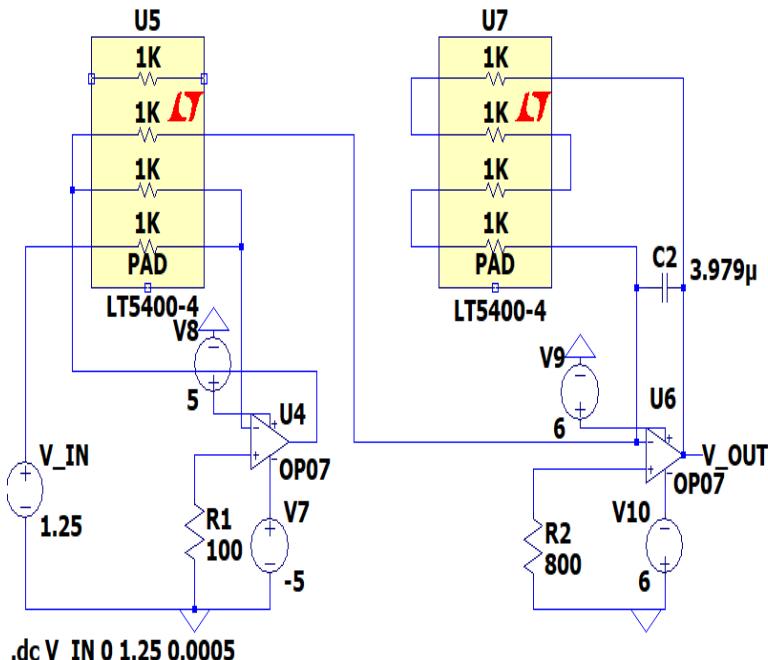
Acondicionamiento de los sensores de humedad de hoja

PYTHOS-31

Dos amplificadores operacionales

Círculo de acondicionamiento del sensor PYTHOS31 con amplificadores operacionales (filtrado y sin modificar referencia)

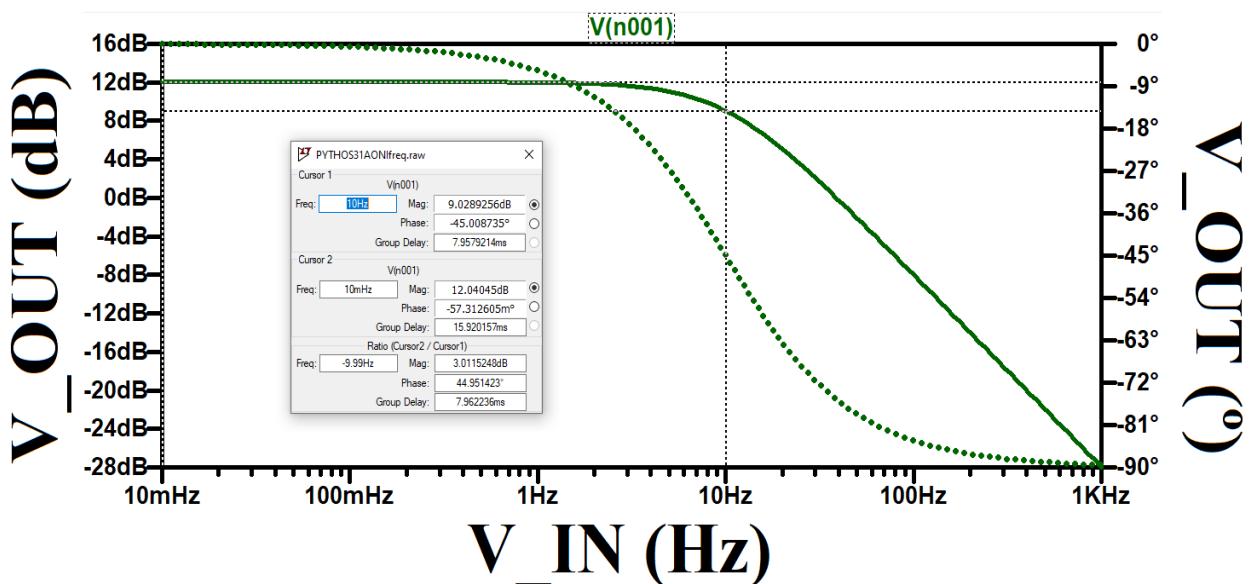
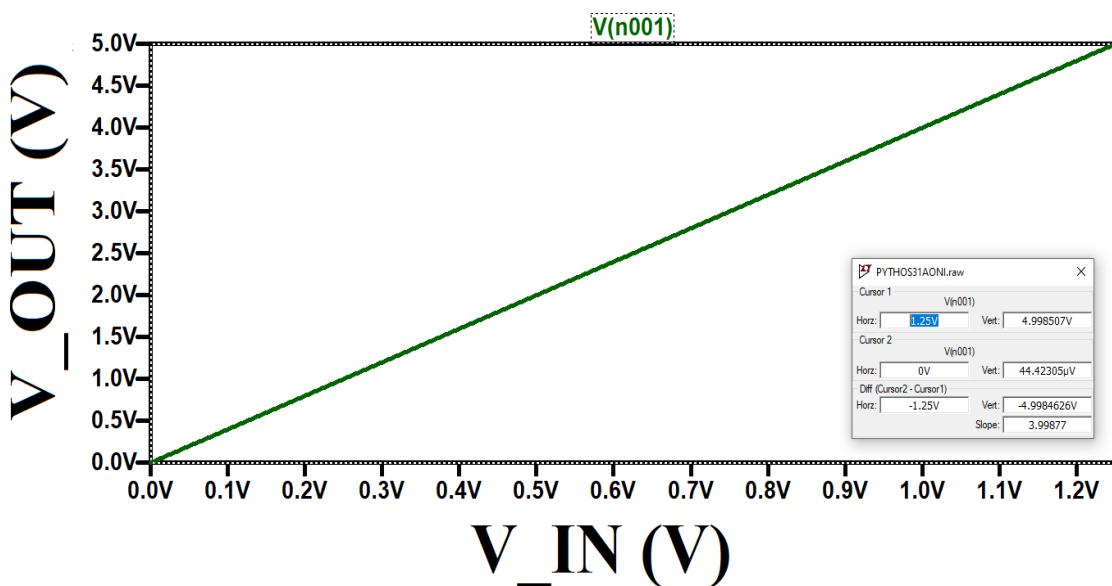
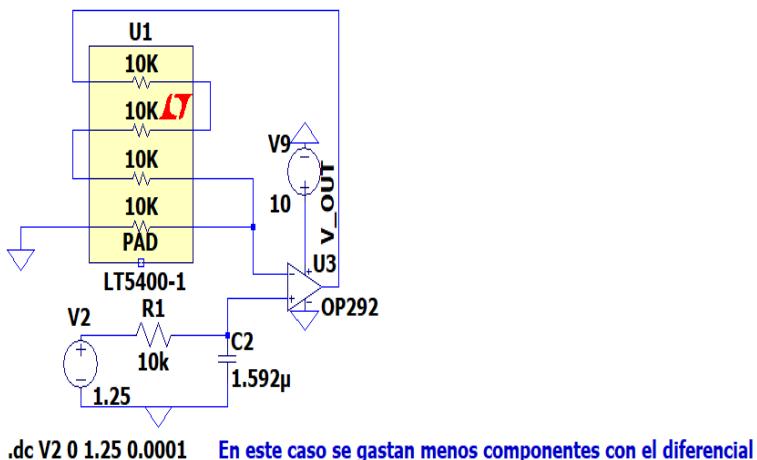
El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH



Amplificador operacional no inveror

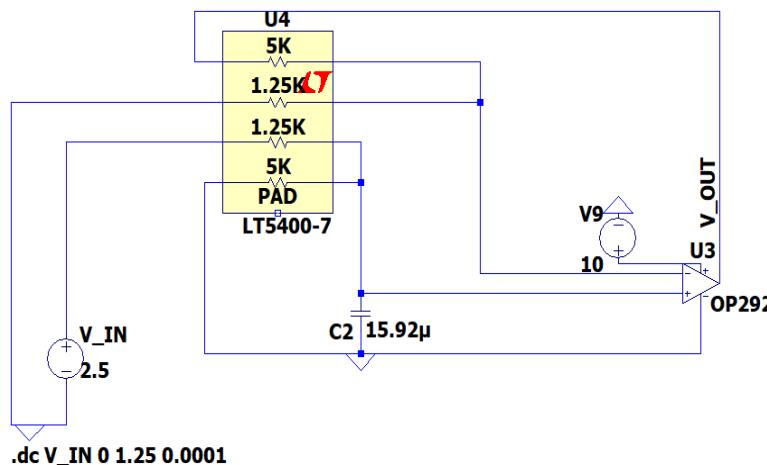
Circuito de acondicionamiento del sensor PYTHOS31 con amplificador operacional no inveror (filtrado y sin modificar referencia)

El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH

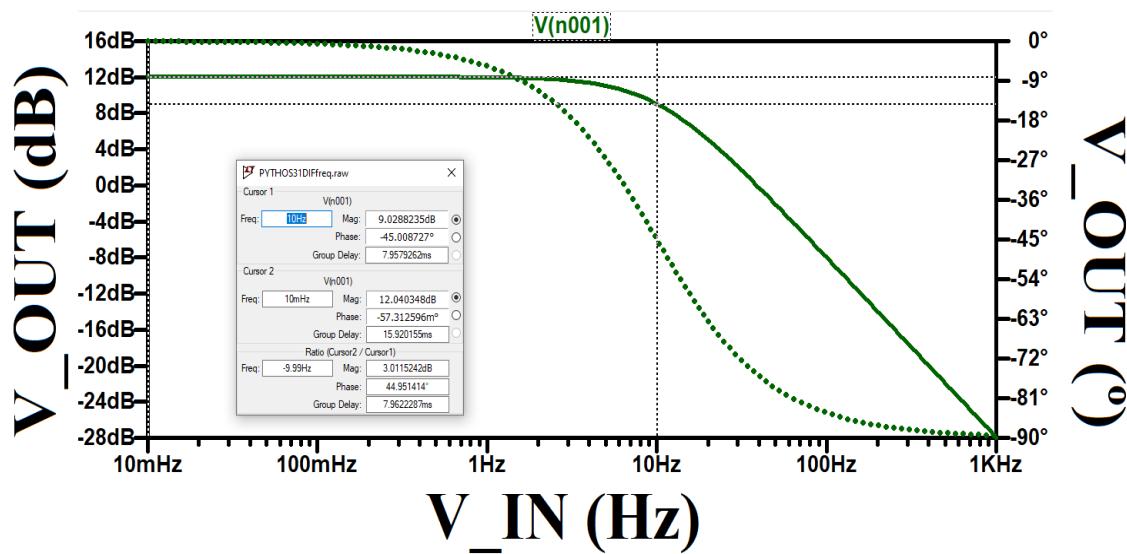
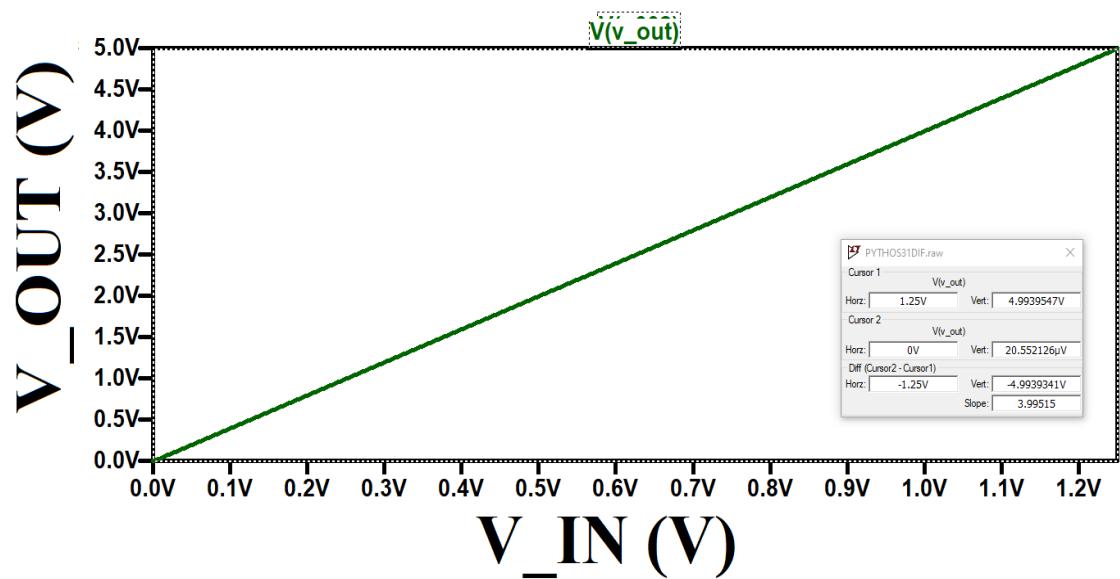


Amplificador diferencial

Circuito de acondicionamiento del sensor LWS-L con amplificador diferencial (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH



.dc V_IN 0 1.25 0.0001

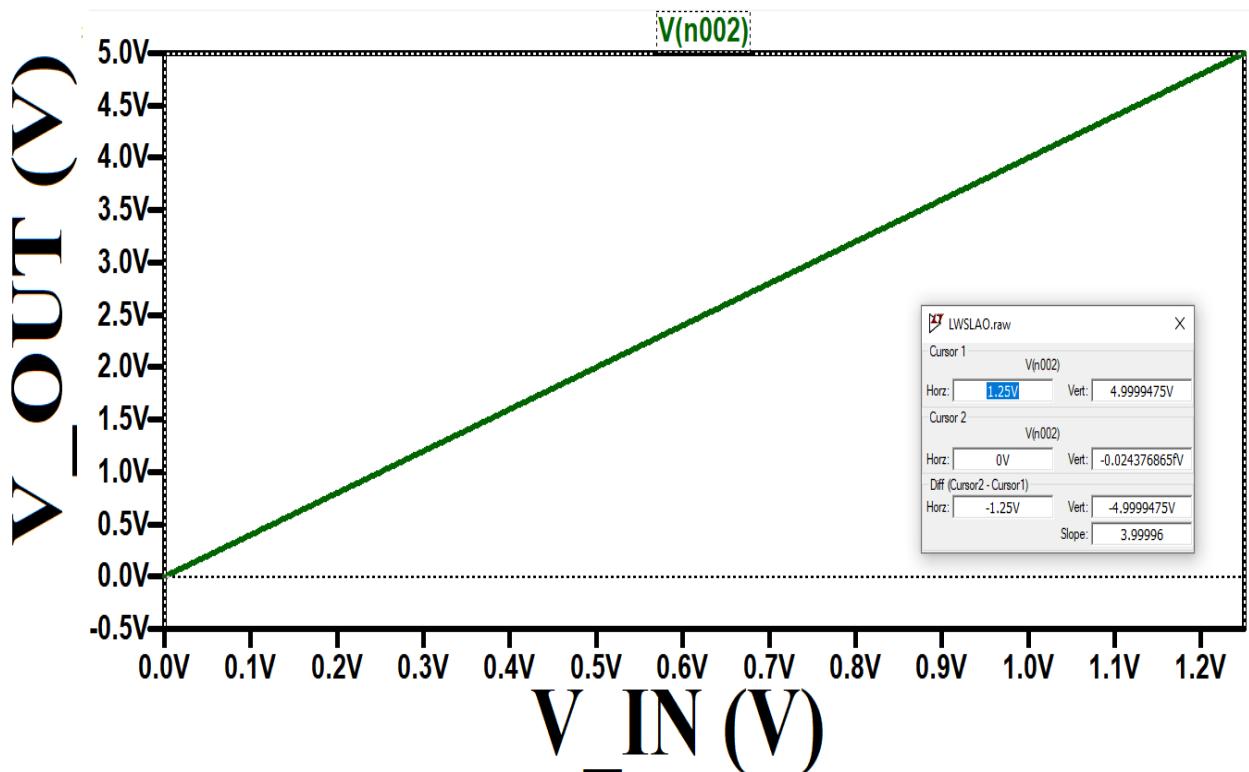
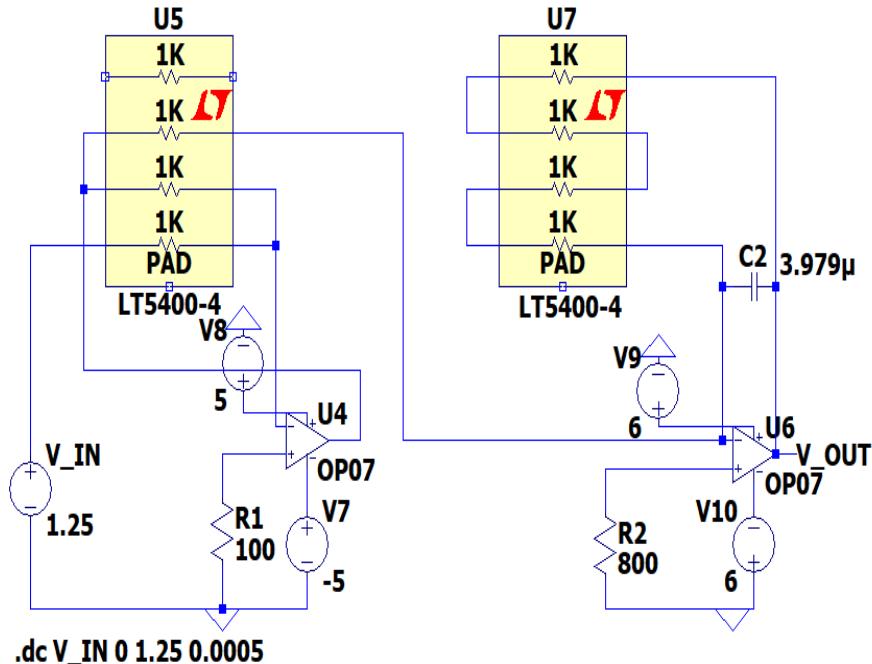


LWS-L

Dos amplificadores operacionales

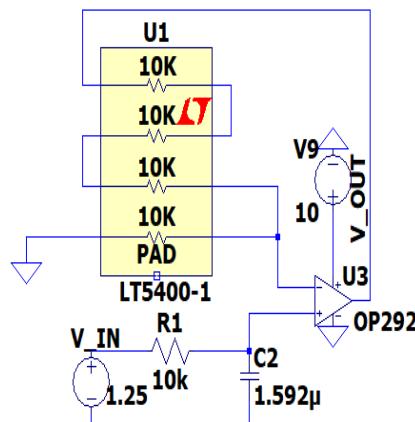
Círculo de acondicionamiento del sensor LWS-L con amplificadores operacionales (filtrado y sin modificar referencia)

El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH

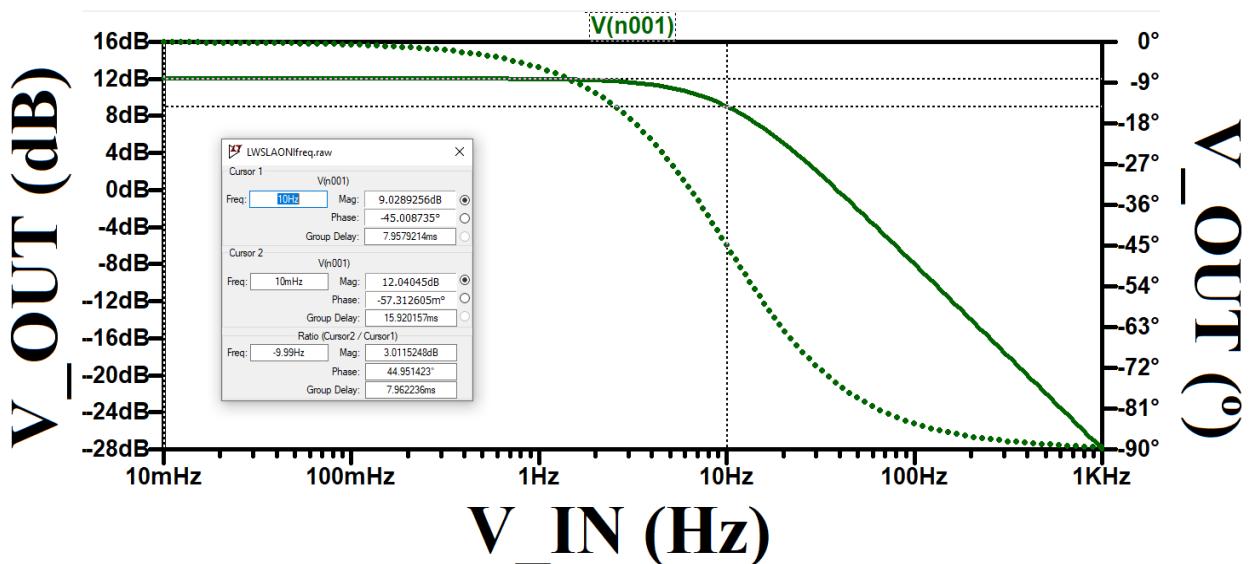
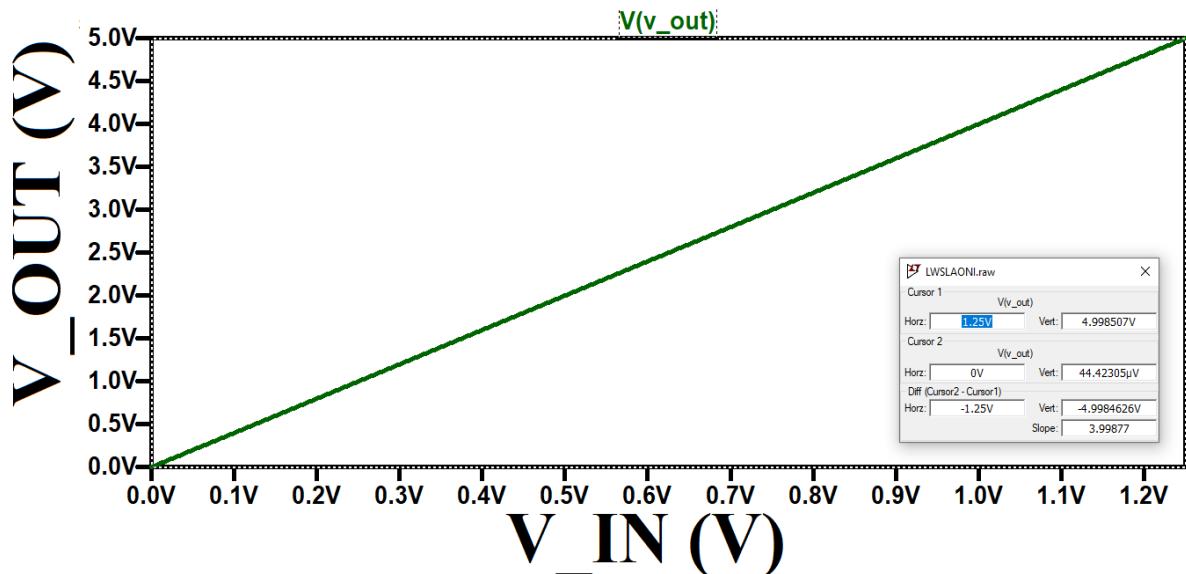


Amplificador operacional no inversor

Círculo de acondicionamiento del sensor LWS-L con amplificador operacional no inversor (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH

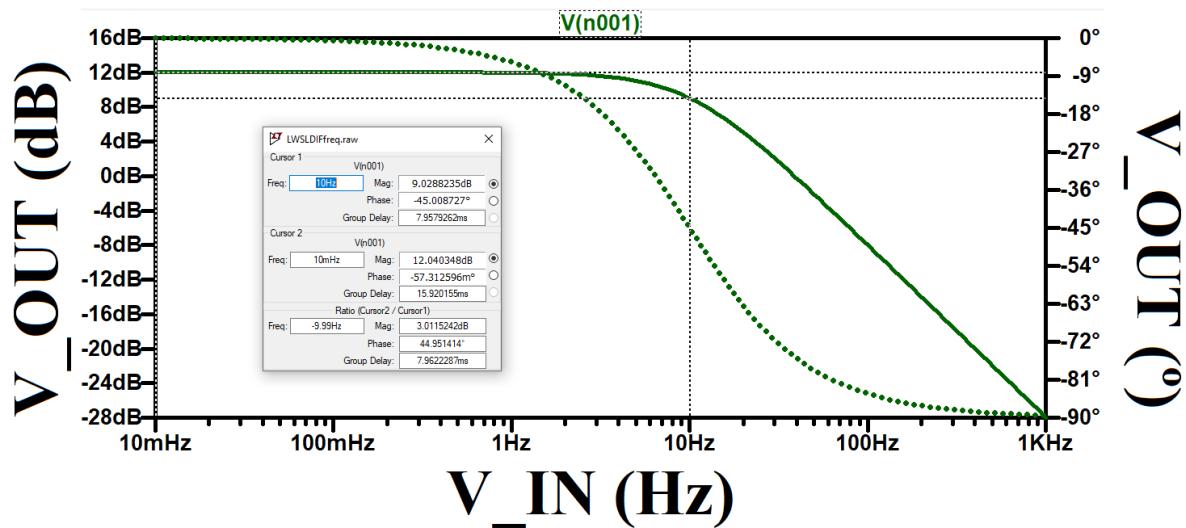
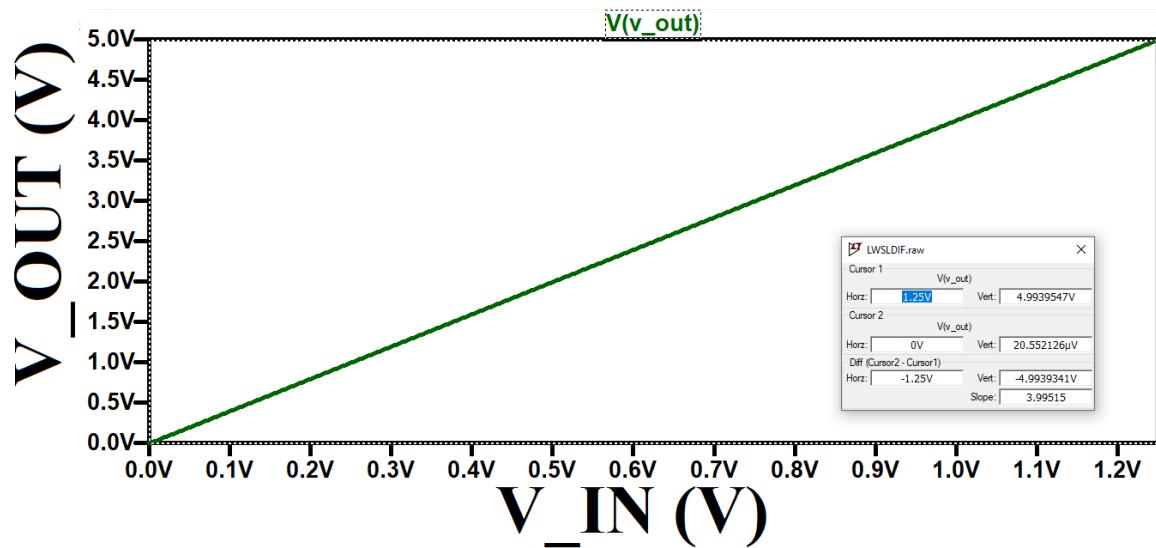
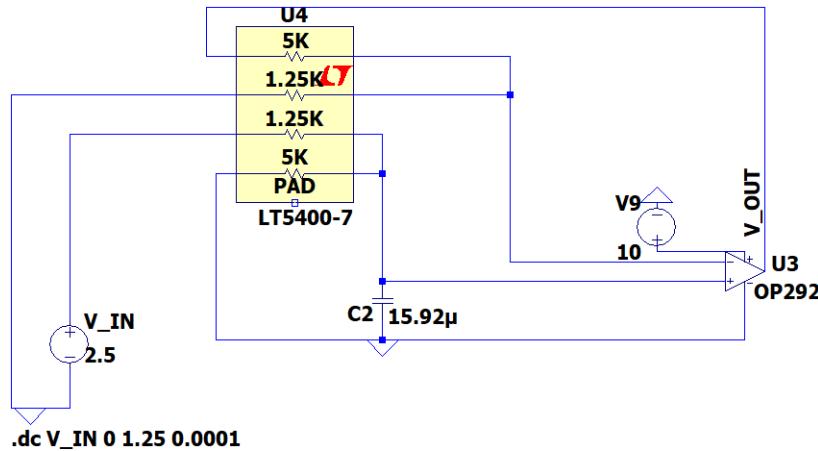


.dc V_IN 0 1.25 0.0001 En este caso se gastan menos componentes con el diferencial



Amplificador diferencial

Circuito de acondicionamiento del sensor LWS-L con amplificador diferencial (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH



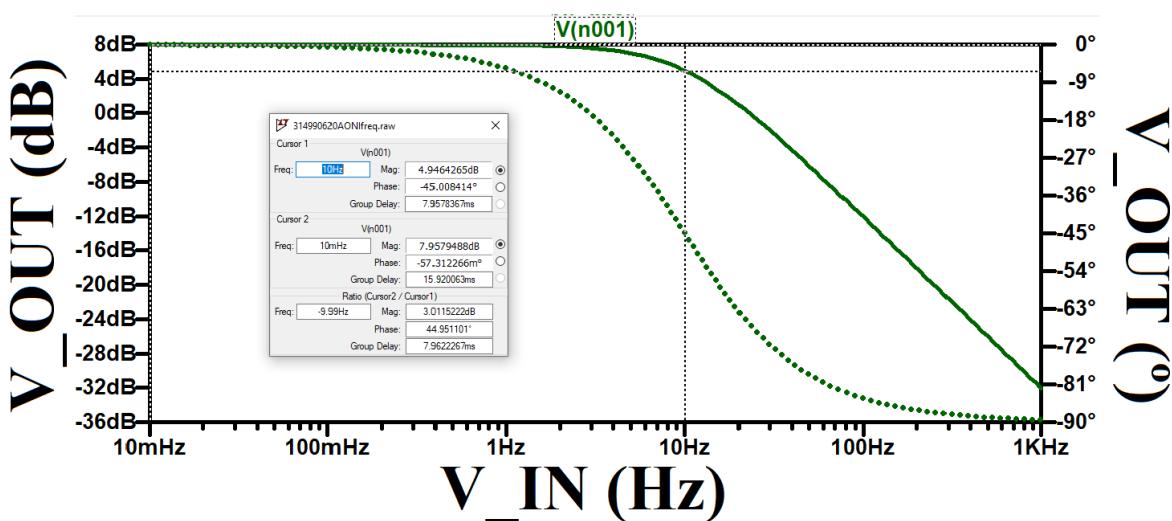
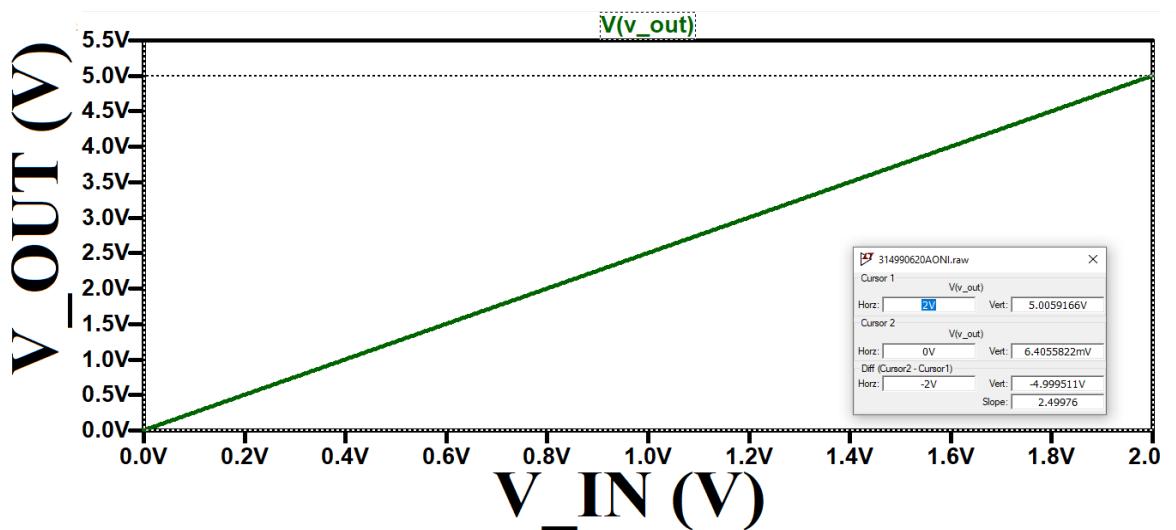
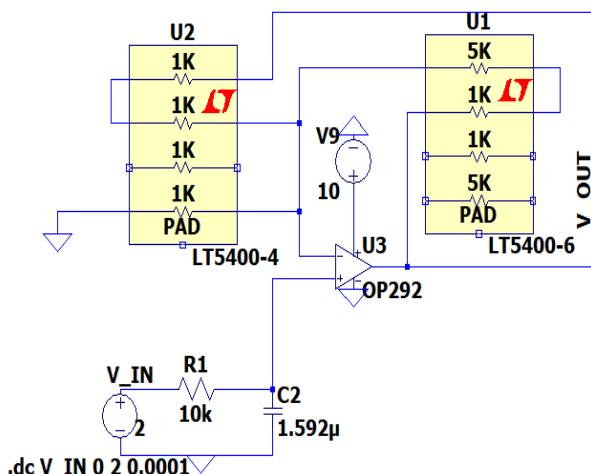
Acondicionamiento de los sensores de humedad de suelo

314990620

Amplificador operacional no inversor

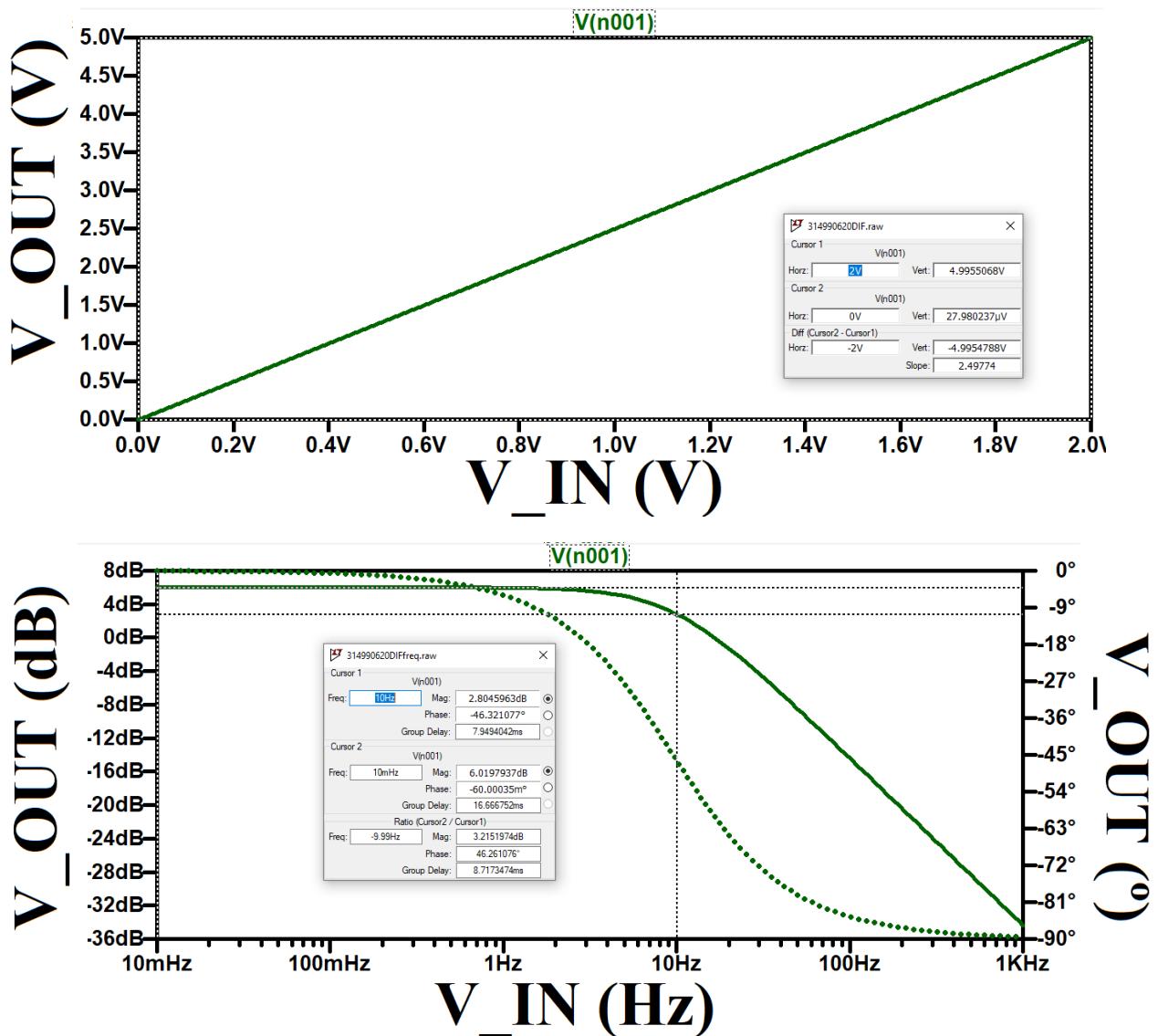
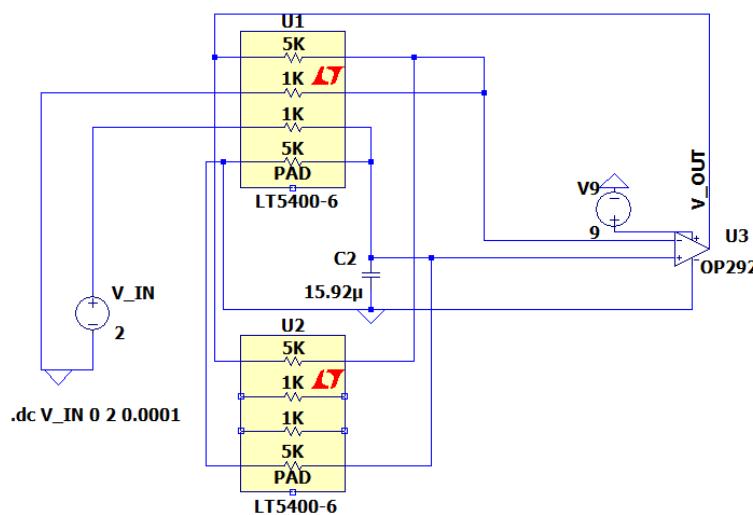
Circuito de acondicionamiento del sensor 314990620 con amplificador operacional no inversor (filtrado y sin modificar referencia)

El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH



Amplificador diferencial

Circuito de acondicionamiento del sensor 314990620 con amplificador diferencial (filtrado y sin modificar referencia)
El rango de salida tiene un mínimo de 0V cuando hay 0% RH y en máximo de 5V cuando hay 100% RH



Acondicionamiento de los sensores de flujo de savia

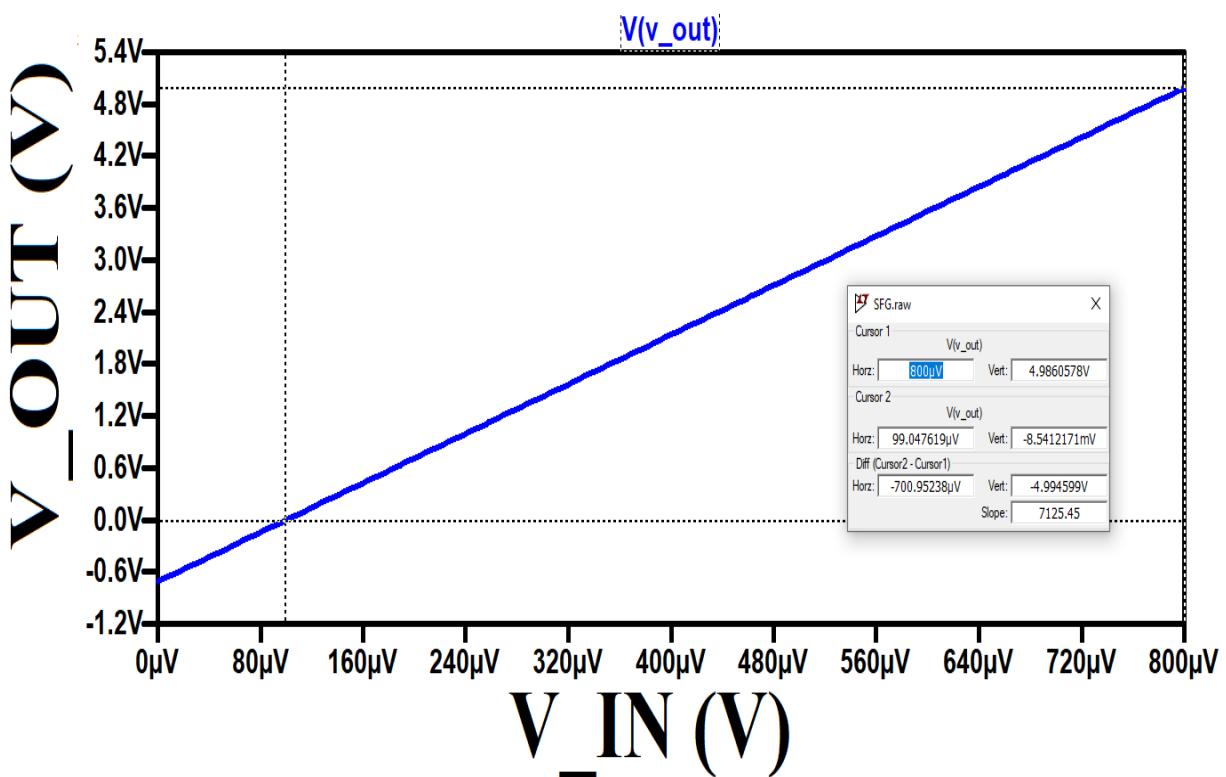
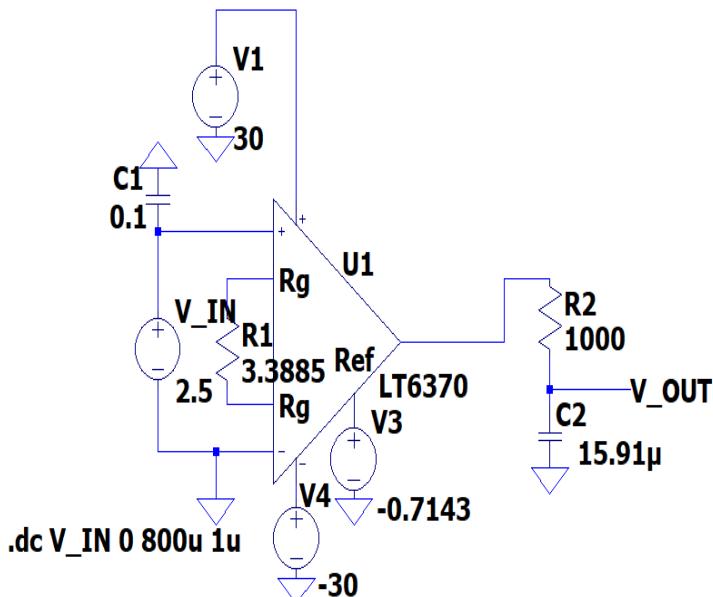
SF-G

Amplificador de instrumentación

Circuito de acondicionamiento SF-G con amplificador de instrumentación (referencia negativa)

Se corresponde con un rango de 2.5°C con el cero en tensión

Y un máximo de 20°C con el cinco voltios en tensión



Apéndice E: Código Arduino del primer nodo sensor

En este código se muestran los SW implementados para el Arduino UNO del primer nodo implementado.

Primer modo: Toma de datos continua

```
#include <Wire.h>
#include "DFRobot_OxygenSensor.h"

#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

int prevVal = LOW;
long th, tl, h, l, ppm;

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH

// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor

void requestEvent();

void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);

    // Inicialización del sensor de oxígeno
    while (!oxygen.begin(Oxygen_IICAddress)) {
        Serial.println("Error al conectar con el sensor de oxígeno.");
        delay(1000);
    }
}
```

```
Serial.println("Conexión exitosa con el sensor de oxígeno.");
}

void loop() {
    long tt = millis();
    int myVal = digitalRead(pwmPin);

    if (myVal == HIGH) {
        if (myVal != prevVal) {
            h = tt;
            tl = h - l;
            prevVal = myVal;
        }
    } else {
        if (myVal != prevVal) {
            l = tt;
            th = l - h;
            prevVal = myVal;
            ppm = 2000 * (th - 2) / (th + tl - 4);
        }
    }

    // Lectura de datos del sensor de oxígeno
    oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

    // Lectura de humedad y conversión a porcentaje
    int humRaw = analogRead(humPin);
    humidity_percent = map(humRaw, 0, 1023, 0, 100);

    // Lectura del sensor de pH
    int phRaw = analogRead(phPin);
    pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

    // Mostrar valores por pantalla
    Serial.print("Humedad (%): ");
    Serial.println(humidity_percent);
    Serial.print("CO2 (ppm): ");
    Serial.println(ppm);
    Serial.print("Oxígeno (%vol): ");
    Serial.println(oxygenData);
    Serial.print("pH: ");
    Serial.println(pHValue);

}

void requestEvent() {

    // Enviar datos por I2C
```

```

byte dataToSend[14];
dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte of
humidity
dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte of
humidity
dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
dataToSend[5] = ppm & 0xFF; // Low byte of CO2
dataToSend[6] = int(oxygenData); // Valor entero de oxígeno
dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
dataToSend[8] = (int(pHValue) >> 8) & 0xFF; // High byte of pH
dataToSend[9] = int(pHValue) & 0xFF; // Low byte of pH
dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte decimal
de pH

Wire.write(dataToSend, sizeof(dataToSend));
}

```

Primer modo: Toma de datos continua, pero solamente toma cuando le pide la petición la Raspberry Pi

```

#include <Wire.h>
#include "DFRobot_OxygenSensor.h"

#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

int prevVal = LOW;
long th, tl, h, l, ppm;

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH

// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

```

```
// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor

void requestEvent();

void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);

    // Inicialización del sensor de oxígeno
    while (!oxygen.begin(Oxygen_IICAddress)) {
        Serial.println("Error al conectar con el sensor de oxígeno.");
        delay(1000);
    }
    Serial.println("Conexión exitosa con el sensor de oxígeno.");
}

void loop() {

}

void requestEvent() {

    long tt = millis();
    int myVal = digitalRead(pwmPin);

    if (myVal == HIGH) {
        if (myVal != prevVal) {
            h = tt;
            tl = h - 1;
            prevVal = myVal;
        }
    } else {
        if (myVal != prevVal) {
            l = tt;
            th = l - h;
            prevVal = myVal;
            ppm = 2000 * (th - 2) / (th + tl - 4);
        }
    }

    // Lectura de datos del sensor de oxígeno
    oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

    // Lectura de humedad y conversión a porcentaje
}
```

```

int humRaw = analogRead(humPin);
humidity_percent = map(humRaw, 0, 1023, 0, 100);

// Lectura del sensor de pH
int phRaw = analogRead(phPin);
pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

// Mostrar valores por pantalla
Serial.print("Humedad (%): ");
Serial.println(humidity_percent);
Serial.print("CO2 (ppm): ");
Serial.println(ppm);
Serial.print("Oxígeno (%vol): ");
Serial.println(oxygenData);
Serial.print("pH: ");
Serial.println(pHValue);

// Enviar datos por I2C
byte dataToSend[14];
dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte of humidity
dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte of humidity
dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
dataToSend[5] = ppm & 0xFF; // Low byte of CO2
dataToSend[6] = int(oxygenData); // Valor entero de oxígeno
dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
dataToSend[8] = (int(pHValue) >> 8) & 0xFF; // High byte of pH
dataToSend[9] = int(pHValue) & 0xFF; // Low byte of pH
dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte decimal de pH

Wire.write(dataToSend, sizeof(dataToSend));
}

```

Segundo modo: Toma de datos periódico, media winsorizada

```
#include <Wire.h>
#include "DFRobot_OxygenSensor.h"

#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH
float ppmAvg = 0.0; // Variable para la media de CO2 durante 30 segundos
unsigned int sampleCount = 0;

// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor

// Tanto por ciento del los límites
#define limPercent 40

// Tiempo en que se toman las medidas en ms
#define MEAS_TIME 30000

unsigned long requestStartTime = 0;
bool isDataCollecting = false;

void requestEvent();

float calculateAverage(float values[], int count) {
    // Calcular la media inicial
    float sum = 0.0;
    for (int i = 0; i < count; i++) {
        sum += values[i];
    }
    float initialMean = sum / count;

    // Calcular los límites superior e inferior
```

```

float upperLimit = initialMean * (1.0 + limPercent / 100.0);
float lowerLimit = initialMean * (1.0 - limPercent / 100.0);

// Calcular la nueva media winsorizada
int validCount = 0;
float sumValid = 0.0;
for (int i = 0; i < count; i++) {
    if (data[i] >= lowerLimit && data[i] <= upperLimit) {
        sumValid += data[i];
        validCount++;
    } else if (data[i] < lowerLimit) {
        sumValid += lowerLimit;
        validCount++;
    } else {
        sumValid += upperLimit;
        validCount++;
    }
}
return sumValid / validCount;
}

void resetAverages() {
    for (int i = 0; i < sampleCount; i++) {
        ppmValues[i] = 0.0;
        humidityValues[i] = 0.0;
        oxygenValues[i] = 0.0;
        pHValues[i] = 0.0;
    }
    sampleCount = 0;
}

void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);

    // Inicialización del sensor de oxígeno
    while (!oxygen.begin(Oxygen_IICAddress)) {
        Serial.println("Error al conectar con el sensor de oxígeno.");
        delay(1000);
    }
    Serial.println("Conexión exitosa con el sensor de oxígeno.");
}

void loop() {

    if (isDataCollecting) {

```

```

unsigned long elapsedTime = millis() - requestStartTime;

long tt = millis();
int myVal = digitalRead(pwmPin);

if (myVal == HIGH) {
    if (myVal != prevVal) {
        h = tt;
        tl = h - 1;
        prevVal = myVal;
    }
} else {
    if (myVal != prevVal) {
        l = tt;
        th = l - h;
        prevVal = myVal;
        ppm = 2000 * (th - 2) / (th + tl - 4);
    }
}

// Lectura de datos del sensor de oxígeno
oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

// Lectura de humedad y conversión a porcentaje
int humRaw = analogRead(humPin);
humidity_percent = map(humRaw, 0, 1023, 0, 100);

// Lectura del sensor de pH
int phRaw = analogRead(phPin);
pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

if (elapsedTime >= MEAS_TIME) { // 30 segundos
    // Calcular las medias de cada sensor durante 30 segundos
    float avgCO2 = calculateAverage(ppmValues, sampleCount);
    float avgHumi = calculateAverage(humidityValues, sampleCount);
    float avgOxygen = calculateAverage(oxygenValues, sampleCount);
    float avgPH = calculateAverage(pHValues, sampleCount);

    // Enviar medias por I2C
    byte dataToSend[14];

    // Mostrar valores por pantalla
    Serial.print("Humedad (%): ");
    Serial.println(humidity_percent);
    Serial.print("CO2 (ppm): ");
    Serial.println(ppm);
    Serial.print("Oxígeno (%vol): ");
    Serial.println(oxygenData);
}

```

```

Serial.print("pH: ");
Serial.println(pHValue);

// Enviar datos por I2C
byte dataToSend[14];
dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte
of humidity
dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte
of humidity
dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
dataToSend[5] = ppm & 0xFF; // Low byte of CO2
dataToSend[6] = int(oxygenData); // Valor entero de oxígeno
dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
dataToSend[8] = (int(pHValue) >> 8) & 0xFF; // High byte of pH
dataToSend[9] = int(pHValue) & 0xFF; // Low byte of pH
dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte decimal de pH

Wire.write(dataToSend, sizeof(dataToSend));

// Reiniciar variables para la siguiente medición
resetAverages();
isDataCollecting = false;
}

}
}

```

Segundo modo: Toma de datos periódico, media recortada

```
#include <Wire.h>
#include "DFRobot_OxygenSensor.h"

#define I2C_SLAVE_ADDRESS 0x0B
#define pwmPin 5
#define humPin A0
#define phPin A1

// Variables globales para mantener los valores actualizados
float oxygenData = 0.0;
float humidity_percent = 0.0; // Variable para la humedad relativa en %
float pHValue = 0.0; // Variable para el valor de pH
float ppmAvg = 0.0; // Variable para la media de CO2 durante 30 segundos
unsigned int sampleCount = 0;
```

```
// Configuración del sensor de oxígeno
#define Oxygen_IICAddress ADDRESS_3
#define COLLECT_NUMBER 10
DFRobot_OxygenSensor oxygen;

// OFFSET para el sensor de pH
float OFFSET = 0.0; // Ajusta esto según las especificaciones del sensor

// Tanto por ciento del los límites
#define limPercent 40

// Tiempo en que se toman las medidas en ms
#define MEAS_TIME 30000

unsigned long requestStartTime = 0;
bool isDataCollecting = false;

void requestEvent();

float calculateAverage(float values[], int count) {
    // Calcular la media inicial
    float sum = 0.0;
    for (int i = 0; i < count; i++) {
        sum += values[i];
    }
    float initialMean = sum / count;

    // Calcular los límites superior e inferior
    float upperLimit = initialMean * (1.0 + limPercent / 100.0);
    float lowerLimit = initialMean * (1.0 - limPercent / 100.0);

    // Calcular la media truncada
    sum = 0.0;
    int validCount = 0;
    for (int i = 0; i < count; i++) {
        if (values[i] >= lowerLimit && values[i] <= upperLimit) {
            sum += values[i];
            validCount++;
        }
    }

    if (validCount > 0) {
        return sum / validCount;
    } else {
        return initialMean;
    }
}
```

```

void resetAverages() {
    for (int i = 0; i < sampleCount; i++) {
        ppmValues[i] = 0.0;
        humidityValues[i] = 0.0;
        oxygenValues[i] = 0.0;
        pHValues[i] = 0.0;
    }
    sampleCount = 0;
}

void setup() {
    Serial.begin(9600);
    pinMode(pwmPin, INPUT);
    Wire.begin(I2C_SLAVE_ADDRESS);
    Wire.onRequest(requestEvent);

    // Inicialización del sensor de oxígeno
    while (!oxygen.begin(Oxygen_IICAddress)) {
        Serial.println("Error al conectar con el sensor de oxígeno.");
        delay(1000);
    }
    Serial.println("Conexión exitosa con el sensor de oxígeno.");
}

void loop() {

    if (isDataCollecting) {
        unsigned long elapsedTime = millis() - requestStartTime;

        long tt = millis();
        int myVal = digitalRead(pwmPin);

        if (myVal == HIGH) {
            if (myVal != prevVal) {
                h = tt;
                tl = h - l;
                prevVal = myVal;
            }
        } else {
            if (myVal != prevVal) {
                l = tt;
                th = l - h;
                prevVal = myVal;
                ppm = 2000 * (th - 2) / (th + tl - 4);
            }
        }
    }
}

```

```

// Lectura de datos del sensor de oxígeno
oxygenData = oxygen.getOxygenData(COLLECT_NUMBER);

// Lectura de humedad y conversión a porcentaje
int humRaw = analogRead(humPin);
humidity_percent = map(humRaw, 0, 1023, 0, 100);

// Lectura del sensor de pH
int phRaw = analogRead(phPin);
pHValue = 3.5 * (phRaw / 1023.0) + OFFSET;

if (elapsedTime >= MEAS_TIME) { // 30 segundos
    // Calcular las medias de cada sensor durante 30 segundos
    float avgCO2 = calculateAverage(ppmValues, sampleCount);
    float avgHumi = calculateAverage(humidityValues, sampleCount);
    float avgOxygen = calculateAverage(oxygenValues, sampleCount);
    float avgPH = calculateAverage(pHValues, sampleCount);

    // Enviar medias por I2C
    byte dataToSend[14];

    // Mostrar valores por pantalla
    Serial.print("Humedad (%): ");
    Serial.println(humidity_percent);
    Serial.print("CO2 (ppm): ");
    Serial.println(ppm);
    Serial.print("Oxígeno (%vol): ");
    Serial.println(oxygenData);
    Serial.print("pH: ");
    Serial.println(pHValue);

    // Enviar datos por I2C
    byte dataToSend[14];
    dataToSend[0] = (int(humidity_percent) >> 8) & 0xFF; // High byte
    of humidity
    dataToSend[1] = int(humidity_percent) & 0xFF; // Low byte
    of humidity
    dataToSend[2] = (ppm >> 24) & 0xFF; // Highest byte of CO2
    dataToSend[3] = (ppm >> 16) & 0xFF; // High byte of CO2
    dataToSend[4] = (ppm >> 8) & 0xFF; // Middle byte of CO2
    dataToSend[5] = ppm & 0xFF; // Low byte of CO2
    dataToSend[6] = int(oxygenData); // Valor entero de oxígeno
    dataToSend[7] = int((oxygenData * 10) - int(oxygenData) * 10); // Decimal de oxígeno
    dataToSend[8] = (int(pHValue) >> 8) & 0xFF; // High byte of pH
    dataToSend[9] = int(pHValue) & 0xFF; // Low byte of pH
}

```

```
    dataToSend[10] = int((pHValue - int(pHValue)) * 100); // Parte
decimal de pH

    Wire.write(dataToSend, sizeof(dataToSend));

    // Reiniciar variables para la siguiente medición
    resetAverages();
    isDataCollecting = false;
}
}
}
```

Apéndice F: Código Raspberry Pi del primer nodo sensor

En este apéndice se muestra el código utilizado en la Raspberry Pi 4 para recibir los datos medidos por los sensores y preprocesados por el Arduino Uno.

```
import smbus
import time
import json

import RPi.GPIO as GPIO

# Configuración del pin GPIO para el LED
ctr_pin = 15
GPIO.setmode(GPIO.BCM)
GPIO.setup(ctr_pin, GPIO.OUT)

# Dirección I2C del Arduino
arduino_address = 0x0B

# Inicialización del bus I2C
bus = smbus.SMBus(1)

# Tiempo en segundos entre peticiones
measurement_time = 10

# Tiempo en segundos entre activar sistema y toma de medida
wakeup_time = 20

def read_data():
    try:
        # Leer 14 bytes de datos desde el Arduino
        data = bus.read_i2c_block_data(arduino_address, 0, 14)

        # Decodificar los datos
        humidity_high = data[0]
        humidity_low = data[1]
        co2_high = (data[2] << 24) | (data[3] << 16) | (data[4] << 8) |
data[5]
        oxygen_int = data[6]
        oxygen_decimal = data[7]
        ph_high = data[8]
        ph_low = data[9]
        ph_decimal = data[10]
```

```

# Recuperar los valores
humidity_percent = (humidity_high << 8 | humidity_low)
co2 = co2_high
oxygen = f"{oxygen_int*10}.{oxygen_decimal*10}"
phValue = ((ph_high << 8 | ph_low) + (ph_decimal / 100.0)) * 4

# Crear un diccionario con los datos
data_dict = {
    "timestamp": int(time.time()),
    "humidity": humidity_percent,
    "co2": co2,
    "oxygen": oxygen,
    "pH": phValue
}

# Obtener la fecha actual (mes y año)
now = time.localtime()
file_name = f"{now.tm_mon}_{now.tm_year}.json"

# Guardar los datos en el archivo de respaldo JSON
with open(file_name, "a+") as file:
    json.dump(data_dict, file)
    file.write("\n")

# Guardar los datos a transmitir
tx_data_file = "tx_data.json"
with open(tx_data_file, "w") as tx_file:
    json.dump(data_dict, tx_file)

print("Datos guardados correctamente:")
print("Humedad (%):", humidity_percent)
print("CO2 (ppm):", co2)
print("Oxígeno (%vol):", oxygen)
print("pH:", phValue)

except Exception as e:
    print("Error al leer datos:", e)

while True:
    GPIO.output(ctr_pin, GPIO.HIGH)
    time.sleep(wakeup_time)      # Tiempo para despertar al sistema
    read_data()
    time.sleep(1)
    GPIO.output(ctr_pin, GPIO.LOW)
    time.sleep(measurement_time) # Esperar 20 minutos entre cada lectura

```

Apéndice G: Comandos AT de Dragino

Se muestran todos los comandos AT, a través de los cuales se puede interactuar con los dispositivos de Dragino.

AT+<CMD>?	: Help on <CMD>
AT+<CMD>	: Run <CMD>
AT+<CMD>=<value>	: Set the value
AT+<CMD>=?	: Get the value
General Commands	
AT	: Attention
AT?	: Short Help
ATZ	: MCU Reset
AT+TDC	: Application Data Transmission Interval
AT+CFG	: Print all configurations
AT+CFGMOD	: Working mode selection
AT+INTMOD	: Set the trigger interrupt mode
AT+5VT	: Set extend the time of 5V power
AT+PRO	: Choose agreement
AT+RXDL	: Extend the sending and receiving time
AT+SERVADDR	: Server Address
AT+APN	: Get or set the APN
AT+FBAND	: Get or Set whether to automatically modify the frequency band
AT+DNSCFG	: Get or Set DNS Server
AT+GETSENSORVALUE	: Returns the current sensor measurement
AT+TR	: Get or Set record time"
AT+NOUD	: Get or Set the number of data to be uploaded
AT+CDP	: Read or Clear cached data
AT+TEMPALARM	: Get or Set alarm of temp
AT+PHALARM	: Get or Set alarm of PH
AT+PHCAL	: calibrate PH value
COAP Management	
AT+URI	: Resource parameters
UDP Management	
AT+CFM	: Upload confirmation mode (only valid for UDP)

MQTT Management	
AT+CLIENT	: Get or Set MQTT client
AT+UNAME	: Get or Set MQTT Username
AT+PWD	: Get or Set MQTT password
AT+PUBTOPIC	: Get or Set MQTT publish topic
AT+SUBTOPIC	: Get or Set MQTT subscription topic
Information	
AT+FDR	: Factory Data Reset
AT+PWD	: Serial Access Password

Apéndice H: Código del segundo nodo sensor

A continuación, se muestran los códigos *sensores.c* y *sensores.h* para la adquisición y preprocesamiento de datos para el segundo nodo implementado.

sensores.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "sensores.h"
#include "onewire.h"

#include "main.h"

int mode;

//Estructuras
// gps
struct gps{
    int32_t lat:28;      // --> latitud
    int32_t lon:28;      // --> longitud
};

// contador de pulsos
struct contador{
    uint32_t cnt:28;     // --> pulsos contados
    uint32_t time:28;    // --> tiempo de medida
    uint32_t maxp:28;    // --> tamaño máximo de pulso
    uint32_t minp:28;    // --> tamaño mínimo de pulso
};

// sensor sensecap
struct sensecap{
    uint16_t t:14;        // --> temperatura del sensor sensecap
    uint16_t vwc:14;       // --> vwc del sensor sensecap
    uint16_t ec:15;        // --> ec del sensor sensecap
    uint16_t sal:15;       // --> salinidad del sensor sensecap
    uint16_t tds:15;       // --> tds del sensor sensecap
    uint16_t eps:14;       // --> epsilon del sensor sensecap
};

// sensor sht20
struct sht20{
    int16_t t:14;          // --> temperatura del sensor sht20
```

```

    uint16_t hum:14;      // --> humedad del sensor sht20
};

struct est_met{
    uint32_t wd:9;        // --> dirección de viento
    uint32_t wd_e:1;       // --> bit error dirección de viento
    uint32_t lb:1;         // --> indicador de baja batería
    uint32_t t:11;         // --> temperatura ambiente
    uint32_t t_e:1;        // --> bit error temperatura ambiente
    uint32_t hum:8;        // --> humedad ambiente
    uint32_t hum_e:1;       // --> bit error humedad ambiente
    uint32_t ws:9;         // --> velocidad del viento
    uint32_t gs:8;         // --> velocidad de ráfaga
    uint32_t gs_e:1;        // --> bit error velocidad de ráfaga
    uint32_t rf:12;        // --> lluvia
    uint32_t uvi:16;        // --> índice ultravioleta
    uint32_t uvi_e:1;       // --> índice ultravioleta
    uint32_t l:24;          // --> cantidad de luz
    uint32_t l_e:1;         // --> bit error cantidad de luz
    uint32_t p:6;           // --> presión
};

struct _sensores07{
    struct est_met est_met;
    struct gps gps;
};

struct _sensores08{
    struct gps gps;
    struct contador contador;
    struct senscap sense[3];
    struct sht20 sht20;
    //uint32_t pluviometro:20;
    uint16_t free:3;
};

struct _sensores09{
    struct gps gps;
    struct contador cont;
    struct senscap sense[3];
    uint16_t free:3;
};

struct _sensores10{
    struct gps gps;
    struct contador cont;
    struct sht20 sht20[4];
};

```

```

struct _sensores11{
    struct gps gps;
    struct contador cont;
    uint16_t ds18b20:16;
};

struct _sensores12{
    struct contador cont[3];
};

struct _sensores13{
    struct sht20 sht20;
    // adc
    uint16_t adc0:12;           // --> 1º adc
    uint16_t adc1:12;           // --> 2º adc
    uint16_t adc2:12;           // --> 3º adc
    uint16_t adc3:12;           // --> 4º adc
    uint16_t free:4;
};

#define PACK __attribute__((packed, scalar_storage_order("little-endian")))
struct PACK _sensores07 sensores07;
struct PACK _sensores08 sensores08;
struct PACK _sensores09 sensores09;
struct PACK _sensores10 sensores10;
struct PACK _sensores11 sensores11;
struct PACK _sensores12 sensores12;
struct PACK _sensores13 sensores13;

///////////////////////////////
#define FIFO_SIZE1 8*80*2
char a[FIFO_SIZE1];
int w1=0,r1=0;

//int IsFull() { return((wI+1)%FIFO_SIZE == rI); } //Devuelve cero si el Buffer está lleno
int fifo1_empty() {
    return(w1 == r1);    //Devuelve cero si el Buffer está vacío, se cumple rI = wI
}
void fifo1_add(char val) {
    a[w1]=val;
    w1++;
    if(w1>FIFO_SIZE1-1)w1=0;
}

```

```

char fifo1_get(void) {
    char b;
    b = a[r1];
    r1++;
    if(r1>FIFO_SIZE1-1)r1=0;
    return b;
}

#define FIFO_SIZE2 25
char b[FIFO_SIZE2];
int w2=0,r2=0;

//int IsFull() { return((wI+1)%FIFO_SIZE == rI); } //Devuelve cero si el
Buffer está lleno
int fifo2_empty() {
    return(w2 == r2); //Devuelve cero si el Buffer está vacío, se
cumple rI = wI
}
void fifo2_add(char val) {
    a[w2]=val;
    w2++;
    if(w2>FIFO_SIZE2-1)w2=0;
}
char fifo2_get(void) {
    char b;
    b = a[r2];
    r2++;
    if(r2>FIFO_SIZE2-1)r2=0;
    return b;
}
///////////////////////////////
/*
//Funcion leer todos los sensores
void read_sensor(void);
void buffercircular_GPS      (void);
void buffercircular_estacion (void);

//Funciones gestion trama GPS
int read_gps(void);
int shiftCircular(char* trama, int desplazamientos);
int ordenarTramaCircular(char* trama);
int obtenerLatitudLongitud(const char* trama, char* latitud, char*
longitud);

//Funciones gestion trama estacion meteorologica

```

```

int read_meteor(void);
char* convertirABinario(const char* numeros, int longitud);
int verificarIDSC(char * numeros, int longitud);
unsigned int obtenerBits(const char* binario, int posicion, int
longitud);
*/

#define LEN_GPS      8*80*2 //Tamaño del buffer que almacena datos del
GPS
#define LEN_METEOR   25      //Tamaño del buffer que almacena datos de la
estacion meteorologica
#define ID 36      //Identificador estacion meteorologica
#define SC 86      //Secuencia seguridad estacion meteorologica

//Variables GPS - Buffer circular
char GPS;    //GPS
char latitud[20]; char longitud[20];
char latitudSinPunto[sizeof(latitud) - 1];
char longitudSinPunto[sizeof(longitud) - 1];
uint32_t lat, lon;
char buffer_GPS [LEN_GPS]; //Buffer que lee los datos del buffer
circular

//Variables estacion metereologico - buffer circular
char meteor;                      //ESTACION METEOROLOGICA
char buffer_meteor[LEN_METEOR];     //Buffer que lee los datos del buffer
circular

//FUNCIONES GESTION TRAMA GPS-----
-----
-----
int shiftCircular(char* trama, int desplazamientos) {
    int longitudTrama = strlen(trama);
    int i, j;
    char temp;

    for (i = 0; i < desplazamientos; i++) {
        temp = trama[0];
        for (j = 0; j < longitudTrama - 1; j++) {
            trama[j] = trama[j + 1];
        }
        trama[longitudTrama - 1] = temp;
    }
}

```

```
    return 0;
}

int ordenarTramaCircular(char* trama) {
    const char* gnrmc = strstr(trama, "$GNRMC");

    if (gnrmc == NULL) return -1;

    int desplazamientos = gnrmc - trama;
    shiftCircular(trama, desplazamientos);

    return 0;
}

int obtenerLatitudLongitud(const char* trama, char* latitud, char*
longitud) {
    char* copiaTrama = strdup(trama);

    if (copiaTrama == NULL) return -1;

    char* token = strtok(copiaTrama, ",");
    int count = 0;
    while (token != NULL) {
        count++;
        if (count == 4) {
            strcpy(latitud, token);
        } else if (count == 6) {
            strcpy(longitud, token);
        }
        token = strtok(NULL, ",");
    }
    free(copiaTrama);

    return 0;
}

void quitarPunto(const char* valor, char* valorSinPunto) {
    int i, j = 0;
    int tamano = strlen(valor);

    for (i = 0; i < tamano; i++) {
        if (valor[i] != '.') {
            valorSinPunto[j] = valor[i];
            j++;
        }
    }
    valorSinPunto[j] = '\0';
}
```

```
}
```



```
// FUNCIONES GESTIÓN TRAMA ESTACIÓN METEOROLÓGICA-----
```

```
-----
```

```
-----
```

```
char* convertirABinario(const char* numeros, int longitud) {
```

```
    if (longitud <= 0) {
```

```
        return NULL;
```

```
    }
```



```
    int i, j;
```

```
    int longitudTotal = longitud * 8; // Longitud total de la cadena
```

```
binaria
```

```
    char* binario = (char*)malloc((longitudTotal + 1) * sizeof(char));
```

```
    if (binario == NULL) {
```

```
        return NULL;
```

```
    }
```



```
    int indice = 0; // Índice en la cadena binaria
```

```
    for (i = 0; i < longitud; i++) {
```

```
        int num = numeros[i];
```

```
        for (j = 7; j >= 0; j--) {
```

```
            int bit = (num >> j) & 1;
```

```
            binario[indice] = bit ? '1' : '0';
```

```
            indice++;
```

```
        }
```

```
    }
```

```
    binario[longitudTotal] = '\0';
```



```
    return binario;
```

```
}
```



```
int verificarIDSC(char* numeros, int longitud) {
```

```
    int ID_RX = numeros[0];
```

```
    int SC_RX = numeros[1];
```



```
    // Verificar desplazamiento circular
```

```
    int desplazamiento = 0;
```

```
    for (int i = 0; i < longitud; i++) {
```

```
        if (numeros[i] == ID && numeros[(i + 1) % longitud] == SC) {
```

```
            desplazamiento = i;
```

```
            break;
```

```
        }
```

```
    }
```



```
    // Realizar el desplazamiento circular
```

```

if (desplazamiento > 0) {
    for (int i = 0; i < desplazamiento; i++) {
        int temp = numeros[0];
        for (int j = 0; j < longitud - 1; j++) {
            numeros[j] = numeros[j + 1];
        }
        numeros[longitud - 1] = temp;
    }
}

// Verificar el ID y el Código de Seguridad
if (ID_RX == ID && SC_RX == SC){
    return 0;
} else {
    return -1;
}

unsigned int obtenerBits(const char* binario, int posicion, int longitud)
{
    unsigned int bits = 0;
    for (int i = 0; i < longitud; i++) { bits = (bits << 1) |
(binario[posicion + i] - '0'); }
    return bits;
}

int read_gps(void){

memset(&buffer_GPS,0,sizeof(buffer_GPS));

int empty = fifo1_empty();

int i_GPS = 0;
while(!empty){
    buffer_GPS[i_GPS]=fifo1_get();
    i_GPS++;
    if(i_GPS>LEN_GPS) i_GPS = 0;
    empty = fifo1_empty();
}

/*
if( ordenarTramaCircular(buffer_GPS)){
lat = 99999999;
lon = 99999999;
} else{ if( obtenerLatitudLongitud (buffer_GPS, latitud, longitud)){
lat = 99999999;
}
}
}

```

```

lon = 99999999;
} else {
quitarPunto(latitud, latitudSinPunto);
quitarPunto(longitud, longitudSinPunto);
lat = atoi(latitudSinPunto);
lon = atoi(longitudSinPunto);
}
}
*/
lat = 99999999;
lon = 99999999;
if(ordenarTramaCircular(buffer_GPS)==0)
    if(obtenerLatitudLongitud(buffer_GPS,latitud,longitud)==0){
        quitarPunto(latitud, latitudSinPunto);
        quitarPunto(longitud, longitudSinPunto);
        lat = atoi(latitudSinPunto);
        lon = atoi(longitudSinPunto);
    }

// Se almacena en las estructuras
sensores07.gps.lat=lat;
sensores08.gps.lat=lat;
sensores09.gps.lat=lat;
sensores10.gps.lat=lat;
sensores11.gps.lat=lat;

sensores07.gps.lon=lon;
sensores08.gps.lon=lon;
sensores09.gps.lon=lon;
sensores10.gps.lon=lon;
sensores11.gps.lon=lon;

return 0;
}

int read_meteor(void){

memset(&buffer_meteor,0,sizeof(buffer_meteor));

int empty = fifo2_empty();

int i_meteor = 0;
while(!empty){
    buffer_meteor[i_meteor]=fifo2_get();
    i_meteor++;
}
}

```

```

    if(i_meteor>LEN_METEOR) i_meteor=0;
    empty = fifo2_empty();
}

int longitud = sizeof(buffer_meteor) / sizeof(buffer_meteor[0]);

if(verificarIDSC(buffer_meteor, longitud)) return -1;

char* binario = convertirABinario(buffer_meteor, longitud);

u_int32_t wd      = obtenerBits(binario, 16, 8);
u_int32_t wd_e   = obtenerBits(binario, 24, 1);
u_int32_t lb     = obtenerBits(binario, 28, 1);
u_int32_t t_e    = obtenerBits(binario, 29, 1);
u_int32_t t      = obtenerBits(binario, 30, 10);
u_int32_t hum_e  = obtenerBits(binario, 40, 1);
u_int32_t hum    = obtenerBits(binario, 41, 7);
u_int32_t ws     = obtenerBits(binario, 27, 1) * 512 +
obtenerBits(binario, 48, 8);
u_int32_t gs_e   = obtenerBits(binario, 56, 1);
u_int32_t gs     = obtenerBits(binario, 57, 7);
u_int32_t rf     = obtenerBits(binario, 64, 16);
u_int32_t uvi_e  = obtenerBits(binario, 80, 1);
u_int32_t uvi   = obtenerBits(binario, 81, 15);
u_int32_t l_e    = obtenerBits(binario, 96, 1);
u_int32_t l     = obtenerBits(binario, 97, 23);
u_int32_t p      = obtenerBits(binario, 130, 6);

// Se almacena en las estructuras
sensores07.est_met.wd=wd;
sensores07.est_met.wd_e=wd_e;
sensores07.est_met.lb=lb;
sensores07.est_met.t_e=t_e;
sensores07.est_met.t=t;
sensores07.est_met.hum_e=hum_e;
sensores07.est_met.hum=hum;
sensores07.est_met.ws=ws;
sensores07.est_met.gs_e=gs_e;
sensores07.est_met.gs=gs;
sensores07.est_met.rf=rf;
sensores07.est_met.uvi_e=uvi_e;
sensores07.est_met.uvi=uvi;
sensores07.est_met.l_e=l_e;
sensores07.est_met.l=l;
sensores07.est_met.p=p;

if (binario != NULL) { free(binario); }

```

```

    return 0;
}

///////////////////////////////
/////////////////////////////
//https://www.analog.com/media/en/technical-documentation/data-
sheets/ds18b20.pdf

void reset_ds18b20(void){
    if(mode!=11) return;

    delay_us_dwt_init();           //Delay para cumplir tiempos del
datasheet
    DS18B20_Reset (mode);         //Reset
}

void read_ds18b20(void){
    if(mode != 11) return;

    int tmp18b20=63488;           //Valor de error
    tmp18b20 = DS18b20_temp();    //Tomar el dato de temperatura

    sensores11.ds18b20=tmp18b20; //Guardar dato estructura

}

///////////////////////////////
/////////////////////////////


extern UART_HandleTypeDef huart6;

#define LEN_MSG 8
#define LEN_DATA 17

#define MAX_SENDORES 3
#define INSTRUCCION_MEDICION_SIZE 5

char instruccionMedicion[INSTRUCCION_MEDICION_SIZE] = {0x04, 0x00, 0x00,
0x00, 0x06}; // Instrucción Sensecap
char buffer_sensecap [LEN_DATA];

int RS485Ready;

```

```

typedef struct {
    uint8_t numSensor;
} SensorData;

void sendRS485(char *mensage){
    HAL_GPIO_WritePin(TX_RS485_GPIO_Port, TX_RS485_Pin,
GPIO_PIN_SET); // Pull DE high to enable TX operation
    HAL_UART_Transmit(&huart6, (uint8_t *)mensage, LEN_MSG , 1000);
    HAL_GPIO_WritePin(TX_RS485_GPIO_Port, TX_RS485_Pin,
GPIO_PIN_RESET); // Pull RE Low to enable RX operation
}

// Función calcular CRC16
uint16_t calc_crc16(char *snd, int num) {
    int i, j;
    uint16_t c, crc = 0xFFFF;
    for (i = 0; i < num; i++) {
        c = snd[i] & 0x00FF;
        crc ^= c;
        for (j = 0; j < 8; j++) {
            if (crc & 0x0001) {
                crc >>= 1;
                crc ^= 0xA001;
            }
            else  crc >>= 1;
        }
    }
    return crc;
}

// Función generar la trama que será la instrucción enviada al sensor
void generar_trama(SensorData *sensor, char *trama) {
    trama[0] = sensor->numSensor;

    for (int i = 0; i < INSTRUCCION_MEDICION_SIZE; i++) {
        trama[i + 1] = instruccionMedicion[i];
    }

    uint16_t crc = calc_crc16(trama, INSTRUCCION_MEDICION_SIZE + 1);

    trama[INSTRUCCION_MEDICION_SIZE + 1] = (char)(crc & 0xFF);
    trama[INSTRUCCION_MEDICION_SIZE + 2] = (char)((crc >> 8) & 0xFF);
}

// Función que verifica los datos recibidos por el sensor
int verificar_datos(char *trama, char *buffer_sense, int
buffer_sense_len) {

```

```

if (trama[0] != buffer_sense[0] && trama[1] != buffer_sense[1]) { //  

Se comprueba que el ID y el modo es igual al enviado en la instrucción  

    return -1;  

}

char buffer_crc[buffer_sense_len - 2];  

for (int i = 0; i < buffer_sense_len - 2; i++) {  

    buffer_crc[i] = buffer_sense[i];  

}

uint16_t crc = calc_crc16(buffer_crc, sizeof(buffer_crc)); //  

Calcula el CRC del buffer_crc  

if (buffer_sense[buffer_sense_len - 2] != (crc & 0xFF) &&  

buffer_sense[buffer_sense_len - 1] != ((crc >> 8) & 0xFF)) { // Ver por  

que no va  

    return -1;  

}

int length = buffer_crc[2];  

if (sizeof(buffer_crc) - 3 != length) { // -> -3 (Se quita el ID,  

modo y el indicador de tamaño)  

    return -1;  

}
return 0;
}

// Función para seleccionar un entero relacionado con una medición
uint16_t seleccionar_enteros_payload( const char *payload, int inicio,
int fin) {
    uint16_t resultado = 0;
    for (int i = inicio; i <= fin; i++) { resultado = (resultado << 8) |  

payload[i]; }
    return resultado;
}

int read_sensecap() {

    memset(&buffer_sensecap, 0, LEN_DATA);

    RS485Ready = 0;

    SensorData sensores[MAX_SENSORES];
    uint16_t t, vwc, ec, sal, tds, eps;

    for(int i = 0; i < MAX_SENSORES; i++){
        sensores[i].numSensor = i+1;
    }
}

```

```
for (int i = 1; i <= MAX_SENORES; i++) {
    char trama[INSTRUCCION_MEDICION_SIZE + 3]; // Array de
(INSTRUCCION_MEDICION_SIZE + 3) elementos de 8 bits (el número del
sensor, la instrucción y el CRC)

    generar_trama(&sensores[i], trama);

    sendRS485(trama);

    HAL_Delay(100);

    char payload[buffer_sensecap[2]];
    if(verificar_datos(trama, buffer_sensecap,
sizeof(buffer_sensecap))) { // Existe algún error
        t = 0xFFFF;
        vwc = 0xFFFF;
        ec = 0xFFFF;
        sal = 0xFFFF;
        tds = 0xFFFF;
        eps = 0xFFFF;
    }

else{
    // Almacenar en payload la carga útil
    for (int i = 0; i < buffer_sensecap[2]; i++) {
        payload[i] = buffer_sensecap[i + 3];
    }

    t = seleccionar_enteros_payload(payload, 0, 1);
    vwc = seleccionar_enteros_payload(payload, 2, 3);
    ec = seleccionar_enteros_payload(payload, 4, 5);
    sal = seleccionar_enteros_payload(payload, 6, 7);
    tds = seleccionar_enteros_payload(payload, 8, 9);
    eps = seleccionar_enteros_payload(payload, 10, 11);

    sensores08.sense[i].t=t;
    sensores08.sense[i].vwc=vwc;
    sensores08.sense[i].ec=ec;
    sensores08.sense[i].sal=sal;
    sensores08.sense[i].tds=tds;
    sensores08.sense[i].eps=eps;
    sensores09.sense[i].t=t;
    sensores09.sense[i].vwc=vwc;
    sensores09.sense[i].ec=ec;
    sensores09.sense[i].sal=sal;
    sensores09.sense[i].tds=tds;
    sensores09.sense[i].eps=eps;
```

```

    }

    return 0;
}

///////////////////////////////
/////////////////////////////
#define CONN 3

int contador[CONN] = {0};
uint32_t previousMillis[CONN] = {0};
uint32_t currentMillis [CONN] = {0};
int periodo[CONN] = {0};

int filtro = 2;
int periodoMINval[CONN] = {99999,99999,99999};
int periodoMAXval[CONN] = {0};

void reset_con(void){
    for(int k=0;k!=CONN;k++)contador[k]=0;
}

void read_con(void){
    //Incluir los datos en las estructuras
    //reset_con();
}

void interrupt_pa11(void){
    int i = 0;
    currentMillis[i] = HAL_GetTick();
    periodo[i] = currentMillis[i] - previousMillis[i];
    if(periodo[i] > filtro){
        contador[i]++;
        previousMillis[i] = currentMillis[i];
        if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
        if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
    }
    sensores08.contador.cnt      = contador;
    sensores08.contador.time     = contador;
    sensores08.contador.maxp     = periodoMAXval[i];
    sensores08.contador.minp     = periodoMINval[i];
    sensores09.contador.cnt      = contador;
}

```

```

sensores09.contador.time      = contador;
sensores09.contador.maxp     = periodoMAXval[i];
sensores09.contador.minp     = periodoMINval[i];
sensores10.contador.cnt      = contador;
sensores10.contador.time      = contador;
sensores10.contador.maxp     = periodoMAXval[i];
sensores10.contador.minp     = periodoMINval[i];
sensores11.contador.cnt      = contador;
sensores11.contador.time      = contador;
sensores11.contador.maxp     = periodoMAXval[i];
sensores11.contador.minp     = periodoMINval[i];
sensores12.contador[i].cnt    = contador;
sensores12.contador[i].time    = contador;
sensores12.contador[i].maxp    = periodoMAXval[i];
sensores12.contador[i].minp    = periodoMINval[i];
}

void interrupt_pa12(void){
    int i = 1;
    currentMillis[i] = HAL_GetTick();
    periodo[i] = currentMillis[i] - previousMillis[i];
    if(periodo[i] > filtro){
        contador[i]++;
        previousMillis[i] = currentMillis[i];
        if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
        if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
    }
    sensores12.contador[i].cnt    = contador;
    sensores12.contador[i].time    = contador;
    sensores12.contador[i].maxp    = periodoMAXval[i];
    sensores12.contador[i].minp    = periodoMINval[i];
}

void interrupt_pb12(void){
    int i = 2;
    currentMillis[i] = HAL_GetTick();
    periodo[i] = currentMillis[i] - previousMillis[i];
    if(periodo[i] > filtro){
        contador[i]++;
        previousMillis[i] = currentMillis[i];
        if(periodo[i] < periodoMINval[i]) periodoMINval[i] = periodo[i];
        if(periodo[i] > periodoMAXval[i]) periodoMAXval[i] = periodo[i];
    }
    sensores12.contador[i].cnt    = contador;
    sensores12.contador[i].time    = contador;
    sensores12.contador[i].maxp    = periodoMAXval[i];
    sensores12.contador[i].minp    = periodoMINval[i];
}

```

```
//////////  

//////////  

#define NUM_CHANNEL 4  

extern ADC_HandleTypeDef hadc1;  

int ADC_VAL[NUM_CHANNEL] = {0};  

int read_ADC(void){  

    memset(&ADC_VAL, 0, sizeof(ADC_VAL));  

    for (int i=0; i<NUM_CHANNEL; i++){  

        ADC_ChannelConfTypeDef sConfig = {0};  

        if (i == 0 ) sConfig.Channel = ADC_CHANNEL_0;  

        if (i == 1 ) sConfig.Channel = ADC_CHANNEL_1;  

        if (i == 2 ) sConfig.Channel = ADC_CHANNEL_2;  

        if (i == 3 ) sConfig.Channel = ADC_CHANNEL_3;  

        sConfig.Rank = 1;  

        sConfig.SamplingTime = ADC_SAMPLETIME_15CYCLES; //480??  

        if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK) return -1;  

        HAL_ADC_Start(&hadc1);  

        HAL_Delay(20);  

        ADC_VAL[i] = HAL_ADC_GetValue(&hadc1);  

        HAL_ADC_Stop(&hadc1);  

    }  

    sensores13.adc0=ADC_VAL[0];  

    sensores13.adc1=ADC_VAL[1];  

    sensores13.adc2=ADC_VAL[2];  

    sensores13.adc3=ADC_VAL[3];  

    return 0;  

}  

//////////  

//////////  

#define SHT2x_I2C_ADDR          0x40
#define SHT2x_HOLD_MASTER        1
#define SHT2x_READ_TEMP_HOLD     0xe3
#define SHT2x_READ_RH_HOLD       0xe5
#define SHT2x_READ_TEMP_NOHOLD   0xf3
#define SHT2x_READ_RH_NOHOLD     0xf5
#define SHT2x_WRITE_REG          0xe6
#define SHT2x_READ_REG           0xe7
#define SHT2x_SOFT_RESET          0xfe
```

```

#define SHT2x_TIMEOUT          1000

typedef enum SHT2x_Resolution {
    RES_14_12 = 0x00,
    RES_12_8 = 0x01,
    RES_13_10 = 0x80,
    RES_11_11 = 0x81,
} SHT2x_Resolution;

extern I2C_HandleTypeDef hi2c1;

void reset_sht20(void){
    uint8_t cmd = SHT2x_SOFT_RESET;
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
}

uint16_t SHT2x_GetRaw(uint8_t cmd) {
    uint8_t val[3] = { 0 };
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
    HAL_I2C_Master_Receive (&hi2c1, SHT2x_I2C_ADDR << 1, val, 3,
SHT2x_TIMEOUT);
    return val[0] << 8 | val[1];
}

uint8_t SHT2x_ReadUserReg(void) {
    uint8_t val;
    uint8_t cmd = SHT2x_READ_REG;
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, &cmd, 1,
SHT2x_TIMEOUT);
    HAL_I2C_Master_Receive (&hi2c1, SHT2x_I2C_ADDR << 1, &val, 1,
SHT2x_TIMEOUT);
    return val;
}

void SHT2x_SetResolution(SHT2x_Resolution res ) {
    uint8_t val = SHT2x_ReadUserReg();
    val = (val & 0x7e) | res;
    uint8_t temp[2] = { SHT2x_WRITE_REG, val };
    HAL_I2C_Master_Transmit(&hi2c1, SHT2x_I2C_ADDR << 1, temp, 2,
SHT2x_TIMEOUT);
}

void read_sht20(void){
    for (int i=0;i<4;i++){

```

```

        if (i == 0 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_1_Pin ,
GPIO_PIN_SET); //PB12 en STM32F411
            if (i == 1 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_2_Pin ,
GPIO_PIN_SET); //PB13 en STM32F411
                if (i == 2 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_3_Pin ,
GPIO_PIN_SET); //PB14 en STM32F411
                    if (i == 3 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_4_Pin ,
GPIO_PIN_SET); //PB15 en STM32F411

                    SHT2x_SetResolution(RES_14_12);
                    uint16_t temp, hum = 0;
                    int hold = 1; //Holding mode, 0 for no hold master, 1 for hold
master.
                    char cmd_temp = (hold ? SHT2x_READ_TEMP_HOLD :
SHT2x_READ_TEMP_NOHOLD);
                    temp = SHT2x_GetRaw(cmd_temp);
                    uint8_t cmd_hum = (hold ? SHT2x_READ_RH_HOLD :
SHT2x_READ_RH_NOHOLD);
                    hum = SHT2x_GetRaw(cmd_hum);
                    sensores08.sht20.t = temp;
                    sensores08.sht20.hum = hum;
                    sensores10.sht20[i].t = temp;
                    sensores10.sht20[i].hum = hum;
                    sensores13.sht20.t = temp;
                    sensores13.sht20.hum = hum;

                    if (i == 0 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_1_Pin ,
GPIO_PIN_RESET);
                        if (i == 1 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_2_Pin ,
GPIO_PIN_RESET);
                            if (i == 2 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_3_Pin ,
GPIO_PIN_RESET);
                                if (i == 3 ) HAL_GPIO_WritePin(GPIOB, GPIO_I2C_4_Pin ,
GPIO_PIN_RESET);
                            }
}
}

///////////////////////////////
/////////////////////////////
extern int flag30s;
extern TIM_HandleTypeDef htim4;

void sensores_loop(){
    power_on();
    reset_con();
    //HAL_TIM_Base_Start_IT(&htim4);    //Inicia el timer de 30 segundos
    //while(flag30s == 0) {__WFI();}    //Respetar los tiempos de inicio
de los sensores

```

```

reset_sht20();
reset_ds18b20();
read_gps();
read_meteor();
read_ADC();
read_sht20();
read_ds18b20();
read_sensecap();
read_con();
power_off();
//HAL_TIM_Base_Stop_IT(&htim4); //Para el timer de 30 segundos
//flag30s = 0;
}

//segun modo genera la cadena a ser enviada
int sensores_data(int mode,char *p){
    switch(mode){
        case 7: memcpy(p,&sensores07,sizeof(sensores07)); return
        sizeof(sensores07); break;
        case 8: memcpy(p,&sensores08,sizeof(sensores08)); return
        sizeof(sensores08); break;
        case 9: memcpy(p,&sensores09,sizeof(sensores09)); return
        sizeof(sensores09); break;
        case 10: memcpy(p,&sensores10,sizeof(sensores10)); return
        sizeof(sensores10); break;
        case 11: memcpy(p,&sensores11,sizeof(sensores11)); return
        sizeof(sensores11); break;
        case 12: memcpy(p,&sensores12,sizeof(sensores12)); return
        sizeof(sensores12); break;
        case 13: memcpy(p,&sensores13,sizeof(sensores13)); return
        sizeof(sensores13); break;
    }
    return 0;
}

```

sensores.h

```

void sensores_reset(int mode);
void sensores_loop ();
int sensores_data (int mode,char *p);

void fifo1_add(char);
char fifo1_get(void);
int fifo1_empty(void);

```

```
void fifo2_add(char);
char fifo2_get(void);
int fifo2_empty(void);

//sensores_ll
void power_on(void);
void power_off(void);
///int adc_read(int channel);
```

Apéndice I: Formato de los datos de la estación meteorológica

Se muestra la forma en la que se reciben los datos de la estación meteorológica.

Nibble No.	Name	Bit No.	Bit Name	Function	Value	Description	
1	FC_H	3	FC_7	identify tx type	0	family code=24H	0
		2	FC_6		0		1
		1	FC_5		1		2
		0	FC_4		0		3
	FC_L	3	FC_3		0		4
		2	FC_2		1		5
		1	FC_1		0		6
		0	FC_0		0		7
2	SC_H	3	SC_7	security code	0		8
		2	SC_6		1		9
		1	SC_5		0		10
		0	SC_4		1		11
	SC_L	3	SC_3		0		12
		2	SC_2		1		13
		1	SC_1		1		14
		0	SC_0		0		15
3	DIR_M	3	DIR_7	Wind direction		value in hex (Range: 0° - 359°) If invalid fill with 0x1ff	16
		2	DIR_6				17
		1	DIR_5				18
		0	DIR_4				19
	DIR_L	3	DIR_3				20
		2	DIR_2				21
		1	DIR_1				22
		0	DIR_0				23
4	DIR_H	3	DIR_8				24
		2	1	FIX_1			25
		1	1	FIX_1			26
		0		WSP_8		wind speed ninth bit	27
	TMP_H	3	low battery			=1 low battery	28
		2	TMP_10	temperature		value in hex 10.5 C = 1F9h -10.5 C = 127h with 400 offset added (Range: -40.0C -> 60.0C) If invalid fill with 0x7ff	29
		1	TMP_9				30
		0	TMP_8				31
5	TMP_M	3	TMP_7				32
		2	TMP_6				33
		1	TMP_5				34
		0	TMP_4				35

BIBLIOGRAFÍA

		TMP_L	3	TMP_3							36
			2	TMP_2							37
			1	TMP_1							38
			0	TMP_0							39
6	HM_H	3	HM_7	humidity							40
		2	HM_6								41
		1	HM_5								42
		0	HM_4								43
	HM_L	3	HM_3								44
		2	HM_2								45
		1	HM_1								46
		0	HM_0								47
7	WIND_H	3	WSP_7	wind speed							48
		2	WSP_6								49
		1	WSP_5								50
		0	WSP_4								51
	WIND_L	3	WSP_3								52
		2	WSP_2								53
		1	WSP_1								54
		0	WSP_0								55
8	GUST_H	3	GUST_7	gust speed							56
		2	GUST_6								57
		1	GUST_5								58
		0	GUST_4								59
	GUST_L	3	GUST_3								60
		2	GUST_2								61
		1	GUST_1								62
		0	GUST_0								63
9	RAIN_HH	3	RAIN_15	rainfall							64
		2	RAIN_14								65
		1	RAIN_13								66
		0	RAIN_12								67
	RAIN_HL	3	RAIN_11								68
		2	RAIN_10								69
		1	RAIN_9								70
		0	RAIN_8								71
10	RAIN_LH	3	RAIN_7	rain counter value in hex (rain factor:0.3mm)							72
		2	RAIN_6								73
		1	RAIN_5								74
		0	RAIN_4								75
	RAIN_LL	3	RAIN_3								76
		2	RAIN_2								77
		1	RAIN_1								78
		0	RAIN_0								79
11	UVI_HH	3	UVI_15	UVI							80
		2	UVI_14								81

		1	UVI_13				82
		0	UVI_12				83
12	UVI_HL	3	UVI_11				84
		2	UVI_10				85
		1	UVI_9				86
		0	UVI_8				87
		3	UVI_7			value in hex (Range: 0 -> 20000) If invalid fill with 0xff	88
13	UVI_LH	2	UVI_6				89
		1	UVI_5				90
		0	UVI_4				91
		3	UVI_3				92
	UVI_LL	2	UVI_2				93
		1	UVI_1				94
		0	UVI_0				95
		3	LIGHT_23				96
14	LIGHT_HH	2	LIGHT_22				97
		1	LIGHT_21				98
		0	LIGHT_20				99
		3	LIGHT_19				100
	LIGHT_HL	2	LIGHT_18				101
		1	LIGHT_17				102
		0	LIGHT_16				103
		3	LIGHT_15				104
15	LIGHT_MH	2	LIGHT_14				105
		1	LIGHT_13				106
		0	LIGHT_12				107
	LIGHT_ML	3	LIGHT_11				108
		2	LIGHT_10				109
		1	LIGHT_9				110
		0	LIGHT_8				111
16	LIGHT_LH	3	LIGHT_7				112
		2	LIGHT_6				113
		1	LIGHT_5				114
		0	LIGHT_4				115
	LIGHT_LL	3	LIGHT_3				116
		2	LIGHT_2				117
		1	LIGHT_1				118
		0	LIGHT_0				119
	CRC_H	3	CRC_7				120
		2	CRC_6				121
		1	CRC_5				122
		0	CRC_4				123
	CRC_L	3	CRC_3			crc checksum value	124
		2	CRC_2				125
		1	CRC_1				126
		0	CRC_0				127

BIBLIOGRAFÍA

17	checksum_h					the sum of previous 16byte w/o carry	128
	checksum_h						129
35	pressure_h_h					unit: hpa H2D(pressure_h:m:l)/100 eg: 0x018A9E for 1010.22hpa	130
36	pressure_h_l						131
37	pressure_m_h						132
38	pressure_m_l						133
39	pressure_l_h						134
40	pressure_l_l						135
							136
uart baud rate:	9600						
stop bit	1						
UV value	UVI						
0 -> 432	0						
433 -> 851	1						
852 -> 1210	2						
1211 -> 1570	3						
1571 -> 2017	4						
2018 -> 2450	5						
2451 -> 2761	6						
2762 -> 3100	7						
3101 -> 3512	8						
3513 -> 3918	9						
3919 -> 4277	10						
4278 -> 4650	11						
4651 -> 5029	12						
>=5230	13						