IMPLEMENTACIÓN DE SISTEMA COMPACTO DE MEDIDA PARA SENSORES DE GAS CONDUCTOMÉTRICOS

Proyecto Fin de Grado presentado por

Aizpea Eceiza Alday

Bajo la supervisión de

Dra. Gemma García Mandayo

D. Jurgi Gonzalez de Chavarri Ostolaza

Para optar al Grado en Ingeniería en Electrónica Industrial

Donostia-San Sebastián, julio 2014

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABLAS

1. INTRODUCCIÓN

1.1 Motivación

Las viviendas, oficinas, tiendas y otros edificios tanto del sector público como del privado suponen cerca del 40% del consumo de energía en la Unión Europea [1]. El 50% de este consumo está destinado a sistemas de calefacción [2] y la reducción de esta demanda energética es la estrategia clave para avanzar en el desarrollo de edificios de bajo consumo energético. Esta estrategia se basa en el control del flujo de aire que circula por los edificios, maximizando la recirculación del aire y minimizando la cantidad de aire fresco. Esta acción eleva el riesgo de que la calidad del aire interior se empobrezca.

La continua exposición a ambientes con una pobre calidad de aire es un problema de salud pública (ver *Figura 1-1*) que concierne, sobre todo, a países desarrollados y a países en vías de desarrollo. La contaminación del aire del interior de los edificios causa 1.5 millones de muertes al año en el mundo [3] y se calcula que la salud de más de la mitad de la población mundial puede verse afectada por la baja calidad del aire interior [4]. En los últimos 30 años los casos de asma y alergias han aumentado considerablemente en las regiones desarrolladas. El corto periodo en el que se ha dado este incremento implica que esta tendencia no es debida a cambios genéticos, sino a la exposición a ambientes con una mala calidad del aire. Además, las enfermedades crónicas asociadas a dichas exposiciones tienen un gran impacto en las economías internacionales. Se estima que las bajas laborales que pueden estar relacionadas con la pobre calidad del aire representan una pérdida de más de 80.000 millones de euros anuales [5].

Las acciones a realizar para mejorar la calidad del aire del interior de los edificios son a menudo sencillas. En unos casos, basta con incrementar el flujo de aire a través de los sistemas de ventilación o aumentar la proporción de aire fresco en el aire que recircula para mejorar notablemente la calidad del aire. En otros casos, son necesarios sistemas de purificación de aire y tecnologías de desintoxicación que reducen la concentración de contaminantes específicos. Para poder usar estas tecnologías de una manera eficiente se requiere un conocimiento detallado de la naturaleza de los contaminantes.

DISEASE	POLLUTANT	IMPACT	DIRECT MEDICAL COSTS (EUROS)
Bronchial asthma in children/ teenagers	Allergens (dust mites, moulds, animal dander)	> 160,000 Prevalent cases/yr	>80 millions
Lung cancer	Radon	1500-6000 deaths per year	26-105 millions
Bronchial asthma in children/ teenagers	Environmental Tobacco Smoke	>30000 cases / yr	> 15 millions
Acute airways infections		>50000 new cases year	> 12 millions
Lung cancer		> 500 deaths / yr	> 9 millions
Acute heart infraction		> 900 deaths / yr	> 8 millions
Leukaemia	Benzene	36 -190 cases / yr	0.5 - 4 millions
Acute poisoning	CO	> 200 deaths / yr	1 million
Total			**> 152 - 234 millions**

Tabla 1-1. Impacto de la baja calidad del aire en la salud

Existen tecnologías capaces de medir concentraciones de los principales contaminantes y existen redes de medida de calidad del aire exterior que informan a los ciudadanos de los potenciales riesgos. Los sistemas de monitorización y el equipamiento analítico usado en esas redes es aparatoso, caro y no proporciona la monitorización en tiempo real requerida. Se han dedicado muchos recursos para medir y monitorizar la calidad del aire del ambiente exterior, pero teniendo en cuenta de que la población de muchas regiones pasa más de un 90% del tiempo en el interior de edificios, es de alta prioridad la implementación de un sistema rentable de monitorización del aire interior con el fin de proteger la salud humana.

Este proyecto se ha realizado en el contexto del proyecto INTASENSE (INTegrated Air quality SENSor for Energy efficient environment control). *La idea principal de INTASENSE (ver Figura 1-1) es integrar un número de micro- y nano-tecnologías en una plataforma de detección común para producir un sistema miniaturizado de bajo coste que pueda medir la calidad del aire e identificar la naturaleza y la forma de los contaminantes [6].* Este dispositivo permitirá el control inteligente de los sistemas de tratamiento de aire para reducir la exposición de la población a ambientes interiores con una pobre calidad del aire. Además, la capacidad de usar este tipo de sistemas de manera más efectiva y económica puede derivar en una amplia implementación de sistemas de purificación de aire. Esto conllevará a una mejora general de la salud pública y de la calidad de vida de los ciudadanos.

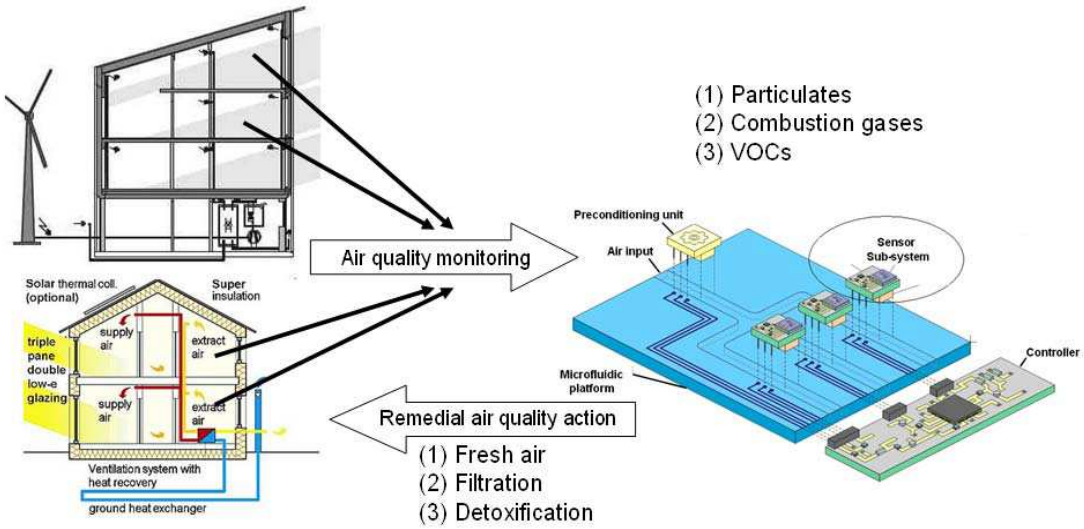


Figura 1-1. Representación de la idea principal de INTASENSE

Este proyecto se centra en el diseño y la implementación de un sistema de acondicionamiento de señal compacto para los sensores de gas conductométricos resultantes del proyecto INTASENSE y para los sensores utilizados para medir humedad y temperatura en el sistema, parámetros que se monitorizan para poder observar cómo afecta su variación a las respuesta del sensor ante diferentes contaminantes. Asimismo, se ha diseñado un sistema que simula un ambiente lo más cercano posible al ambiente real en el que va a trabajar el sensor, para poder obtener así unos resultados más próximos al campo real que los alcanzados en los ensayos realizados en laboratorio.

1.2 Objetivos

Los principales objetivos de este proyecto son:

- El diseño e implementación de una plataforma de comunicaciones de bajo coste y bajo consumo que recoja los datos de cada sensor, el de temperatura y humedad relativa y los sensores de gas, para enviarlos a una interfaz de usuario donde se tratan y muestran los datos obtenidos.

- Ensayo de la plataforma diseñada conectada al prototipo INTASENSE, que integra los sensores con la micro-fluídica, en una atmosfera de laboratorio con gases contaminantes para establecer su comportamiento y comparar los datos obtenidos con los medidos por los equipos de medida del laboratorio.

- El diseño e implementación de un banco de prueba que contenga el prototipo INTASENSE y la electrónica de control diseñada para realizar ensayos en condiciones lo más próximas posible a las aplicaciones finales.

1.3 Estructura del proyecto

Para poder llevar a cabo los objetivos del proyecto, se ha seguido el procedimiento explicado a continuación.

En primer lugar, se ha programado el sensor digital de temperatura y humedad relativa haciendo uso de una plataforma comercial de bajo coste con una interfaz de dispositivo USB2.0 que permite mostrar las lecturas realizadas en un monitor serie. Una vez el sensor funciona como es requerido, se ha diseñado una interfaz de usuario donde se muestran las lecturas de temperatura y humedad relativa de una manera más gráfica pudiendo registrar los datos obtenidos para un posterior análisis de los resultados.

A continuación, se ha diseñado un pequeño circuito para poder estimar la resistencia del sensor de gas y se han actualizado tanto el software como la interfaz de usuario de manera que también se obtengan y registren las lecturas de la resistencia.

Finalmente, se ha implementado un sistema que se asemeja a un ambiente de trabajo real del sensor y así demostrar cómo funcionaría el dispositivo completo, incluyendo la electrónica de control, en una aplicación final.

2. DISEÑO HARDWARE

En este capítulo se presentan las bases teóricas más fundamentales de los sistemas electrónicos y de los microcontroladores. Posteriormente se procede a la presentación de los componentes que han sido necesarios para la implementación del sistema de medida.

2.1 Definición de sistema electrónico

Un sistema electrónico es un conjunto de circuitos que interactúan entre sí para obtener un resultado. Es un conjunto de sensores, circuitos de procesamiento y control, actuadores y fuente de alimentación. Un sistema electrónico se puede distinguir claramente en tres bloques:

- **Bloque de entrada** a través del cual se introduce la orden o la señal, bien a través de un accionador o bien a través de sensores que obtienen información del mundo físico externo y lo transforman en una señal eléctrica manipulable por la circuitería de control interna.

- **Bloque de proceso** que consta de circuitos de procesamiento y control capaces de manipular la señal eléctrica convenientemente. Dependiendo tanto del diseño de los componentes hardware del sistema como del conjunto de instrucciones que tenga programado convierte una señal de entrada en una señal capaz de activar el bloque de salida.

- **Bloque de salida** formada por actuadores que transforman la señal eléctrica, resultado del procesamiento, en energía que actúa sobre el mundo físico externo.

En la *Figura 2-1* se muestra de forma gráfica el diagrama básico de un sistema electrónico con los tres bloques principales claramente diferenciados.

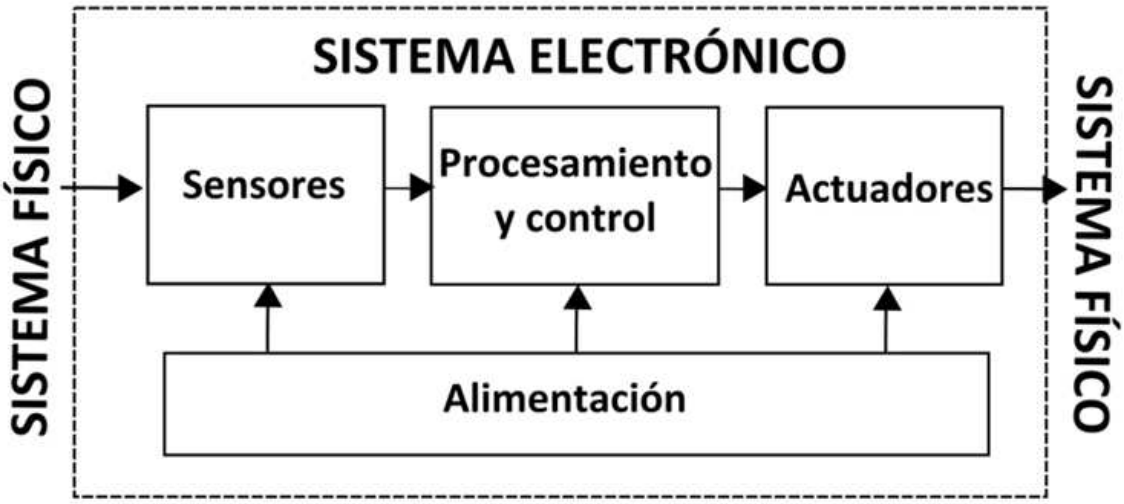


Figura 2-1. Diagrama de bloques de sistema electrónico

En el sistema electrónico implementado en el proyecto, se identifican como bloque de entrada el sensor de temperatura y humedad relativa y el sensor conductométrico de gas. El bloque de proceso está formado por la plataforma comercial de bajo coste que se emplea para la adquisición y procesamiento de los datos. Finalmente, en este proyecto, no se dispone de un bloque de salida formado por actuadores, ya que ese no es el objetivo del proyecto.

2.1.1 ¿Qué es un microcontrolador?

El microcontrolador es el componente más importante de la circuitería de procesamiento y control. Es un computador completo, integrado en un solo chip, capaz de ejecutar constantemente una serie de instrucciones definidas por el programador.

Por definición, el microcontrolador debe tener en su interior tres elementos básicos:

- **CPU (Central Processing Unit/Unidad Central de Proceso)***:* es la parte del microcontrolador encargada de leer cada instrucción, de ejecutarla y de controlar que la ejecución se ha dado correctamente. Estas instrucciones hacen uso de datos disponibles previamente y generan otros datos diferentes que podrán ser

utilizados por las siguientes instrucciones. En la siguiente figura se muestra la arquitectura básica de una CPU.

- **Diferentes tipos de memoria***:* alojan tanto las instrucciones como los datos, de manera que estos estén siempre disponibles para que la CPU pueda acceder y trabajar con ellas. Generalmente, se distinguen dos tipos de memorias. Por un lado, las llamadas persistentes o no volátiles, tipo ROM, en las que el contenido se almacena de forma permanente incluso tras cortes de alimentación eléctrica. Por otro lado, están las volátiles, tipo RAM, en las cuales el contenido se pierde tras un corte de alimentación. Habitualmente, las instrucciones de programa se almacenan en la memoria no volátil y la memoria volátil se destina a guardar las variables y datos.

- **Patillas de E/S (entrada/salida)**: son las encargadas de comunicar el microcontrolador con el exterior. Se pueden conectar los sensores en las patillas de entrada haciendo que el microcontrolador reciba las señales requeridas del mundo físico exterior. Los actuadores se acoplan a las patillas de salida de manera que éstas interactúan con el mundo exterior según las señales proporcionadas por el microcontrolador. Cabe destacar que la mayoría de las patillas pueden ser utilizados indistintamente como entradas o como salidas.

En este proyecto, para implementar el sistema de medida de sensores se ha hecho uso de una plataforma comercial que se basa en el microcontrolador LM4F120 sobre la cual se profundizara más adelante.

2.2 Componentes empleados para la implementación del sistema

2.2.1 Stellaris® LM4F120 Launchpad

El *Stellaris® LM4F120 Launchpad Evaluation Board* es una plataforma de evaluación de bajo coste para microcontroladores basados en ARM® Cortex™ -M4F. El Stellaris Launchpad dispone de botones programables de usuario, un LED RGB para aplicaciones personalizadas y pines del Stellaris LM4F120 Launchpad Boosterpack XL Interface que permiten conectar a la plataforma diferentes periféricos y sensores. En el diseño destaca la interfaz de dispositivo USB2.0 que permite comunicar la placa con

otros dispositivos, normalmente un PC, y así transferir datos. Además, a través de la interfaz USB2.0, la plataforma recibe la tensión de alimentación necesaria, alrededor de 5V. La *Figura 2-2* muestra una foto del Stellaris Launchpad.

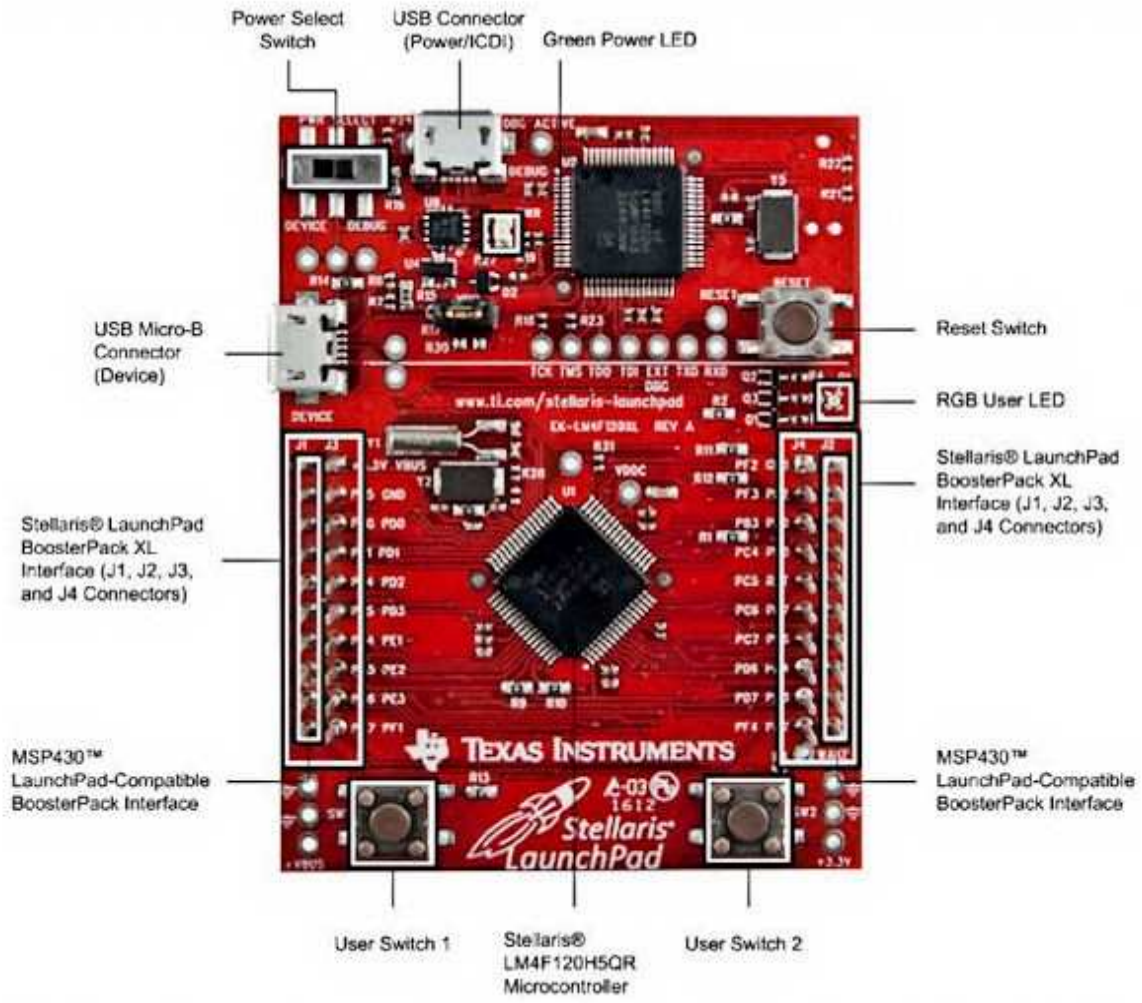


Figura 2-2. Stellaris LM4F120 Launchpad Evaluation Board

Una de las características más importantes de la plataforma es la presencia de pines E/S tanto digitales como analógicos (el “pin map” está disponible en [7]). Las entradas o salidas digitales se emplean para la comunicación con dispositivos que proporcionan una señal digital, como es el caso del sensor de temperatura y humedad relativa que se utiliza en este proyecto. Los pines analógicos, en cambio, reciben una señal de tensión que necesita ser convertida a un número discreto digital para que el

microcontrolador pueda procesar la información. El Stellaris Launchpad dispone de dos convertidores analógico digital (ADC) idénticos que transforman la señal eléctrica en código digital.

La resolución de las entradas analógicas depende directamente del número de bits que usa el convertidor analógico digital. En el caso del Stellaris Launchpad éste dispone de 12 bits, lo que significa que el número digital discreto tomará valores entre 0 y 4095. Teniendo en cuenta que las tensiones que puede proporcionar el circuito diseñado para estimar la resistencia del sensor conductométrico de gas fluctúan entre 0 y 3.3 V, se concluye que la resolución de las entradas analógicas es de 0.8 mV.

Para implementar un sistema de medida de sensores de gas conductométricos que estime la resistencia del sensor con la mayor exactitud posible, es necesario tener una alta resolución. La resolución proporcionada por el convertidor analógico digital del Stellaris Launchpad es más elevada que la proporcionada por otras plataformas comerciales del mercado, como las placas Arduino, es por ello que se ha optado por el uso de esta plataforma.

El resto de características del *Stellaris® LM4F120 Launchpad Evaluation Board* se pueden consultar en el User Manual [8].

2.2.2 Sensor de temperatura y humedad relativa, SHT71

Este sensor de temperatura y humedad relativa se caracteriza por su alta fiabilidad, su bajo consumo y su alta estabilidad a largo plazo. Además, proporciona una salida digital totalmente calibrada.

Para las mediciones de la humedad relativa se usa un sensor capacitivo. La temperatura, en cambio, se mide con un sensor “band-gap”. Ambos sensores se encuentran acoplados a un convertidor analógico digital de 14 bits y a un circuito de interfaz serie que dan lugar a una señal de calidad superior, un rápido tiempo de respuesta e insensibilidad a perturbaciones externas. La interfaz consta de cuatro pines definidos en la *Figura 2-3*.

Pin	Name	Comment
1	SCK	Serial Clock, input only
2	VDD	Source Voltage
3	GND	Ground
4	DATA	Serial Data, bidirectional

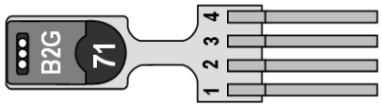


Figura 2-3. Especificaciones de la interfaz

Los pines de alimentación y tierra se han conectado a los 3.3 V y a la tierra proporcionados por el *Stellaris Launchpad*. Los pines de reloj y datos se han conectado a dos pines digitales de la plataforma (ver *Figura 2-4*). Cabe destacar que es necesario el uso de una resistencia "pull-up" entre la alimentación y el pin de datos para evitar que en ausencia de señal los valores binarios recibidos no fluctúen sin sentido. Se ha empleado una resistencia de 10kΩ siguiendo las recomendaciones de la hoja de especificaciones.

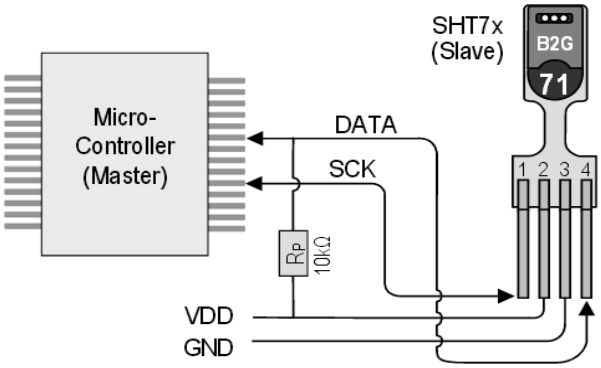


Figura 2-4. Circuito de aplicación habitual del SHT71

En cuanto a la comunicación con el sensor, ésta se detallará más adelante a la hora de profundizar en el diseño del software. Igualmente, se recomienda acudir a la hoja de especificaciones que se encuentra en el Anexo I para adquirir mayor conocimiento acerca tanto de la comunicación con el sensor como del resto de las características de éste.

2.2.3 Sensor de gas conductométrico

Los sensores de gas son dispositivos que detectan moléculas de gas y emiten una señal eléctrica en función de la cantidad de moléculas detectadas. Los sensores conductométricos de gas son la opción más barata en el mercado de los sensores de gas.

El principio de funcionamiento de este tipo de sensores se basa en el cambio en la conductancia del material sensible en función de la concentración de gas. En primer lugar, se calienta el material hasta la temperatura de trabajo de manera que las moléculas de oxígeno son absorbidas en la superficie. Estas moléculas atrapan electrones de la banda de conducción del semiconductor. Durante estos procesos la resistencia del material se estabiliza y se mantiene en la línea base. Manteniendo la misma temperatura e introduciendo un pulso de gas, las moléculas de gas reaccionan con el oxígeno dando lugar a una variación en la densidad de electrones. Por lo tanto, se da un incremento en la resistencia del material sensible. En el momento en el que se retira el gas la resistencia recupera la línea base. Este proceso se puede ver gráficamente en la *Figura 2-5*.

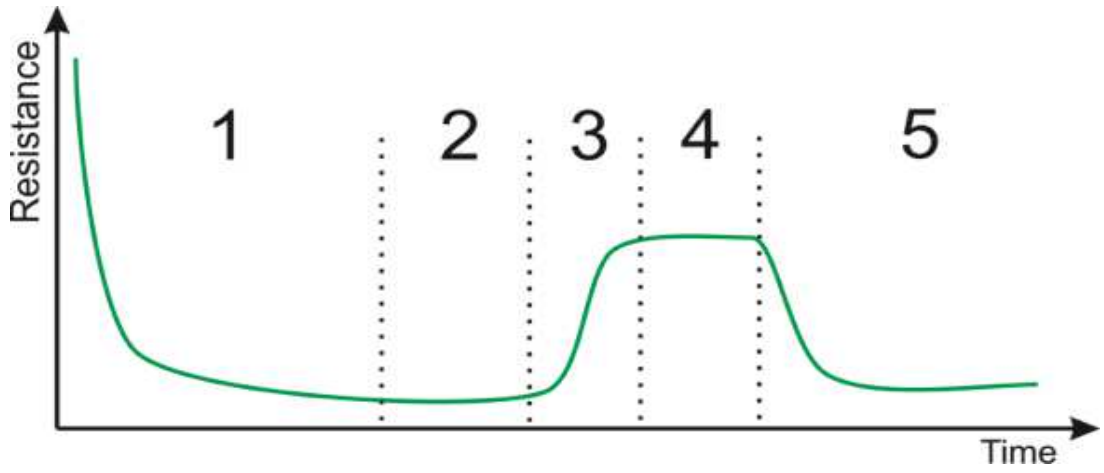


Figura 2-5. Respuesta del sensor a un pulso de gas

Para más detalles, en la referencia [9] se desarrolla de manera exhaustiva el diseño y la caracterización de los sensores conductométricos de gas.

En este proyecto se dispone de un sensor cuyo material sensible es el óxido de níquel. El sensor está soldado a una estructura con cuatro pines. Dos de ellos son los pines calefactores que se conectan a una fuente de alimentación que proporciona al sensor entre la tensión necesaria para poder así alcanzar la temperatura de trabajo. Entre los otros dos pines se puede obtener el valor de la resistencia del sensor.

Para poder estimar la resistencia del sensor se ha implementado un divisor de tensión.

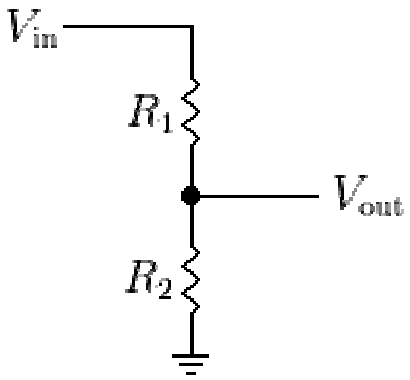


Figura 2-6. Divisor de tensión

Se sabe que la resistencia base del sensor de óxido de níquel es del orden de los 100 kΩ y es representada por la R_1 de la *Figura 2-6*. En base a ese dato, se ha optado por una R_2 de 107 kΩ. V_{out} toma un valor de entre 0 y 3.3 V dependiendo de la caída de tensión en función de la resistencia del sensor. Conocido V_{out}, conectando un pin analógico del Stellaris Launchpad a ese punto del divisor, y conocida R_2 con una simple aplicación de la Ley de Ohm se obtiene la corriente que circula por el divisor resistivo:

$$V_{out} = I * R_2$$

$$I = \frac{V_{out}}{R_2}$$

Conocida la corriente se aplica de nuevo la Ley de Ohm para así obtener una estimación de la resistencia del sensor:

$$R_1 = \frac{V_{in} - V_{out}}{I}$$

2.2.4 Bomba rotativa de paletas, SP135FZ

Esta es una bomba rotativa de paletas para aire o gas impulsado por motor eléctrico con un mínimo consumo de corriente. Además, tiene unas características muy lineales. Por su diseño, presenta unas vibraciones reducidas, muy bajo ruido y un tamaño de bolsillo extremadamente pequeño. Esta bomba está disponible en varias tensiones de funcionamiento, pero para este proyecto se ha optado por la que opera a 3V, ya que se tiene como objetivo crear un sistema de medida de bajo consumo y bajo coste.

En la *Figura 2-7* se muestra la curva de rendimiento. La bomba que se ha utilizado en este proyecto viene representada por la línea roja. Esta curva y otras características se encuentran explicadas en profundidad en la hoja de especificaciones del componente que se puede consultar en el Anexo II.

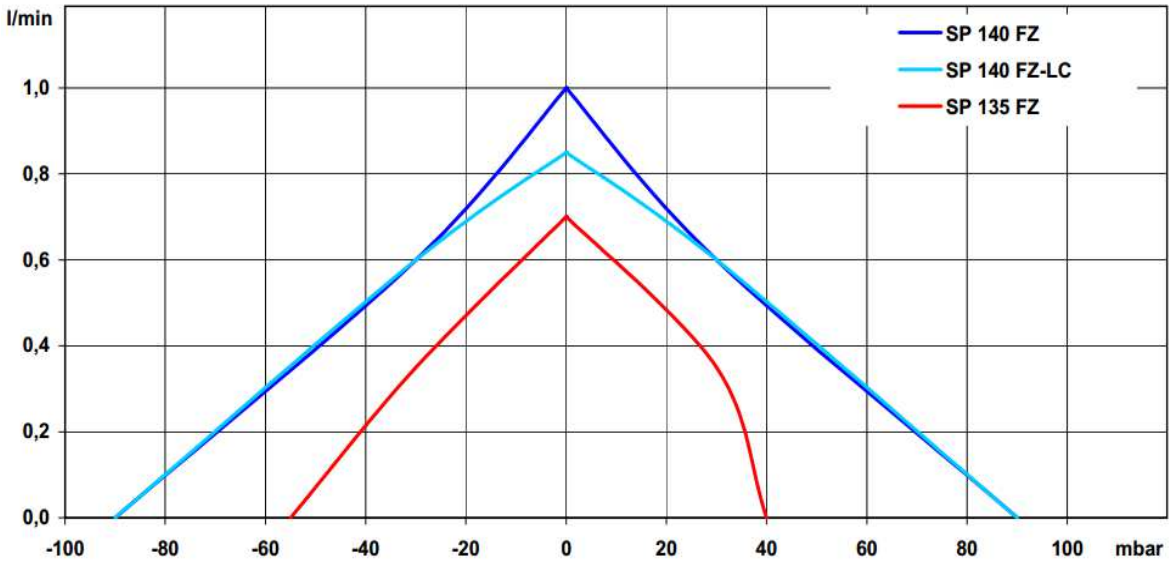


Figura 2-7. Curva de rendimiento SP135

3. DISEÑO SOFTWARE

En este capítulo se introduce el entorno Energia, el cual se ha empleado para diseñar e implementar el programa que el Stellaris Launchpad ejecuta. Se introduce también, el lenguaje de programación gráfico LabVIEW que se ha empleado para diseñar e implementar la interfaz de usuario. Asimismo, se procede a la explicación de la estructura y función del programa y la interfaz de usuario.

3.1 Introducción al lenguaje Energía

Energia es una plataforma abierta de prototipado electrónico iniciado por Robert Wessels en Enero de 2012 [10] con el fin de habilitar los entornos de Wiring y Arduino para los Launchpad basados en MSP430 y los Stellarpad de Texas Instruments. Al igual que Arduino, tanto el entorno de desarrollo como el lenguaje de programación Energia están basados en otro entorno y lenguaje preexistentes, Processing, un lenguaje que surgió en 2001 y se usa actualmente para aprender las bases de la programación, para el prototipado y para la producción.

3.1.1 Entorno de Desarrollo Integrado

El Entorno de Desarrollo Integrado, también llamado sketch, es un conjunto de herramientas software que permite escribir y editar los programas, comprobar que no se ha cometido ningún error y permite grabar los programas en la memoria del microcontrolador del Stellaris Launchpad para que este se convierta en el ejecutor autónomo de los programas.

Antes de empezar a trabajar en el entorno, es necesario configurarlo. Para ello, en la pestaña “tools” (ver *Figura 3-1*) se define cuál de las placas de Texas Instruments que soporta Energia es la que está conectada. El Stellaris Launchpad empleado para la implementación del sistema de medida se identifica en el entorno Energia como *StellarPad w/ lm4f120h5qr (80 MHz).* Asimismo, se determina a qué puerto serie está conectada esa placa para que la transmisión de datos se dé correctamente.

Tal y como se observa en la *Figura 3-1*, un sketch siempre se compone de tres secciones:

Figura 3-1. Entorno de Desarrollo Integrado

- **Sección de declaración de variables globales y funciones**: se ubica al principio del sketch y su inicio y final no están delimitados por ningún símbolo. En ella se declaran algunas variables y se definen las funciones que implementan algunas acciones a realizar por el programa con el fin de simplificar el diseño del programa principal.

- **Sección "void setup()"**: delimitada por llaves de apertura y cierre, las instrucciones escritas en esta sección se ejecutan una única vez al encender o resetear el Stellaris Launchpad. Por ello, las instrucciones de esta sección sirven para realizar alguna preconfiguración inicial. En este proyecto que se ha

aprovechado esta sección para inicializar y configurar la comunicación a través del puerto serie.

- **Sección "void loop()"**: delimitada también por llaves de apertura y cierre, las instrucciones escritas en esta sección se ejecutan después de las de la sección "void setup()" infinitas veces permitiendo que el programa responda hasta que la placa se apague o resetee. Es decir, las instrucciones que forman parte de esta sección son el programa principal que se ejecuta continuamente.

3.1.2 Variables

Uno de los conceptos más importantes en relación con las variables es el ámbito de una variable. Una variable puede ser global o local.

- **Variables globales**: se declaran tal y como indica el nombre en la sección de declaración de variables globales y funciones. Son variables que pueden ser accedidas desde cualquier punto del sketch. Todas las instrucciones del programa, sin importar dónde estén escritas estas, pueden utilizar y cambiar el valor de estas variables.

  En el programa implementado en este proyecto, se han declarado como variables globales los pines del Stellaris Launchpad conectados al sensor de temperatura y humedad relativa y al sensor de gas conductométrico, ya que es necesario poder acceder a los pines desde diferentes funciones y diferentes secciones del sketch.

- **Variables locales**: son las que se declaran en el interior de alguna de las secciones del sketch y pueden ser accedidas sólo por las instrucciones escritas dentro de esa misma sección.

  En el programa implementado se han utilizado variables locales dentro de las funciones definidas en la primera sección del sketch para poder realizar operaciones internas y dentro del de la sección "void loop()" para poder mandar los datos obtenidos al puerto serie.

Otro de los conceptos más relevante en cuanto a las variables es el tipo de una variable. Los tipos de variable que el lenguaje Energia admite son: void, boolean, char, byte, int, word, long, float, double, string y array. A continuación se explican en

profundidad los tipos int y char, que son los tipos que se han empleado en el programa diseñado. Se puede encontrar una amplia información sobre el resto de tipos en la referencia [9].

- **int**: es el principal tipo de variable para almacenar números enteros. Toma 2 bytes de memoria, lo que significa que el rango de valores va de -32768 a 32767 (de -2^15 a 2^15).

  Todas las variables del programa que toman un valor entero se han declarado como variables de tipo int. También se han declarado como tipo int los pines de E/S del Stellaris Launchpad que se utilizan para conectar el sensor digital de temperatura y humedad y el sensor conductométrico de gas. Para ser estrictos, los pines se han declarado como *const int*, es decir, son variables sólo de lectura y su valor no se puede cambiar.

- **char**: es un tipo de variable que toma 1 byte de memoria y almacena un solo carácter. Dichos caracteres se almacenan como números mediante la codificación ASCII, tomando valores de -128 a 127 (de -2^7 a 2^7).

  Se han declarado como tipo char las variables que reciben un carácter de la interfaz de usuario. Más adelante, se explica la función que tienen estos caracteres en el programa principal.

3.1.3 Comunicación serie

Como se ha mencionado en anteriores apartados, el Stellaris Launchpad dispone de una interfaz de dispositivo USB2.0 que permite comunicar la plataforma con el PC mediante un puerto serie, permitiendo transferir los datos proporcionados por los sensores.

Dentro del sketch, se pueden enviar datos al microcontrolador y recibirlos de él haciendo uso del elemento *Serial* del lenguaje Energia. *Serial* es un objeto del lenguaje que contiene instrucciones que permiten manipular la comunicación serie.

A continuación, se explica la sintaxis, funcionamiento y utilidad de las instrucciones incluidas en el objeto Serial que se han empleado en la programación.

- **Serial.begin()**: es la instrucción que habilita el canal serie para que pueda dar comienzo la comunicación por él. Por tanto, la ejecución de esta instrucción se tiene que dar antes de realizar cualquier transmisión por dicho canal. Es por ello, que se escribe dentro de la sección “void setup()”. Además, mediante su único parámetro se ha definido la velocidad en bits/s a la que se produce la transferencia serie de datos. En el caso del Stellaris Launchpad esta conexión está limitada a 9600 baud.

- **Serial.print()**: envía a través del puerto serie un dato (definido como parámetro) desde el microcontrolador hacia el exterior. Ese dato puede ser de cualquier tipo y se puede especificar a través de una variable o explícitamente. En ese último caso, el dato se escribe entre comillas simples si es un carácter y entre comillas dobles si es una cadena. Esta instrucción tiene un valor de retorno, que puede no ser utilizado en el programa, que especifica el número de bytes enviados.

- **Serial.available()**: devuelve el número de bytes disponibles para ser leídos que provienen del exterior a través del canal serie. Son bytes que ya han llegado al microcontrolador y permanecen almacenados temporalmente en un buffer hasta que son procesados mediante la instrucción *Serial.read()*. Si no hay bytes para leer esta instrucción devuelve 0. Por ello, en el programa implementado se comprueba si Serial.available()>0, que significa que han llegado bytes al microcontrolador, y en ese caso se procede a la lectura de los datos. Esta instrucción no tiene parámetros.

- **Serial.read()**: devuelve el primer byte no leído de los que están almacenados en el buffer y al leerlo lo elimina de ese buffer. Para leer el siguiente byte, se ha de volver a ejecutar *Serial.read()* y así sucesivamente hasta que se hayan leído todos los bytes. Cuando ya no quedan más bytes disponibles, esta instrucción devuelve un -1. Esta instrucción no tiene parámetros.

En la *Tabla 3-1* se muestra un resumen de todas las instrucciones explicadas anteriormente.

3.1.4 E/S digitales y analógicas

El Stellaris Launchpad interacciona con los sensores mediante pines E/S digitales y analógicos. Para ello, el lenguaje Energia dispone de varias funciones que tratan señales de tipo digital y de tipo analógico.

Las funciones que ofrece el lenguaje Energia para trabajar con entradas y salidas digitales, que en este proyecto están conectados a la interfaz serie del SHT71, se definen a continuación.

- **pinMode()**: configura un pin digital, cuyo número se especifica como primer parámetro, como entrada o como salida, dependiendo de si el valor que toma el segundo parámetro es la constante predefinida INPUT o OUTPUT, respectivamente. No tiene valor de retorno.

- **digitalWrite()**: escribe un valor HIGH (3 V) o un valor LOW (0 V) en un pin digital. Para ello, el pin digital en el cual se escribe ha tenido que ser definido mediante la función *pinMode()* como OUTPUT. Esta función tiene dos parámetros, el primero es el pin al que se le envía la señal y mediante el segundo parámetro se define si en ese pin se quiere escribir un HIGH o un LOW. No tiene valor de retorno.

- **digitalRead()**: devuelve el valor leído de un pin digital que anteriormente ha tenido que ser configurado mediante la función *pinMode()* como INPUT. Esta función tiene como único parámetro el pin del cual se quiere leer el valor. Su valor de retorno es una variable de tipo int que puede tomar dos valores: HIGH (1) o LOW (0).

En cuanto a la gestión de las entradas y salidas analógicas, cabe destacar que los pines analógicos funcionan siempre como entrada, por lo que no es necesario configurarlas mediante *pinMode()*. Es posible generar una salida analógica PWM configurando un pin digital específico (no todas los pines digitales pueden generar una señal PWM) como OUTPUT y haciendo uso de la función *analogWrite()*. El Stellaris Launchpad no dispone de pines digitales que puedan generar señales PWM.

Así pues, el lenguaje Energia dispone de la siguiente función para gestionar las entradas analógicas, conectadas en este proyecto al sensor conductométrico de gas:

- **analogRead()**: devuelve el valor leído del pin de entrada analógico que se ha especificado como parámetro. Este valor se obtiene, tal y como se explica en el apartado 2.2.1, mapeando la entrada analógica obtenida (entre 0 y 3.3 V) a un valor entero entre 0 y 4095 (ADC de 12 bits), tal que la resolución de lectura es de 3.3 V/4095, es decir, de 0.8 mV.

En la *Tabla 3-1* se puede ver un resumen de las instrucciones que gobiernan las E/S digitales y analógicas.

INSTRUCCIÓN	PARÁMETRO	VALOR DE RETORNO
Serial.begin()	baud rate	___
Serial.print()	dato	___
Serial.available()	___	número de bytes
Serial.read()	___	byte leído
pinMode()	pin digital	___
digitalWrite()	pin digital, HIGH/LOW	___
digitalRead()	pin digital	variable int
analogRead()	pin analógico	valor leído

Tabla 3-1. Resumen de instrucciones empleadas en la implementación del software

3.2 Implementación del programa con Energía

3.2.1 Programación del sensor SHT71

Para poder obtener una lectura correcta de temperatura o humedad relativa se sigue la secuencia de medida descrita en la *Figura 3-2*. Las secuencias de medida empiezan siempre habilitando la transmisión para después mandar al sensor un comando que varía de valor dependiendo de si se requiere una medida de temperatura o una de humedad relativa. Una vez se ha verificado mediante un acknowledge que el comando se ha recibido correctamente, es necesario un tiempo de espera para que se complete la

medida. Dos bytes de datos de medida se transmiten a continuación. Es necesario que el microcontrolador detecte un ACK después de la transmisión de cada byte.

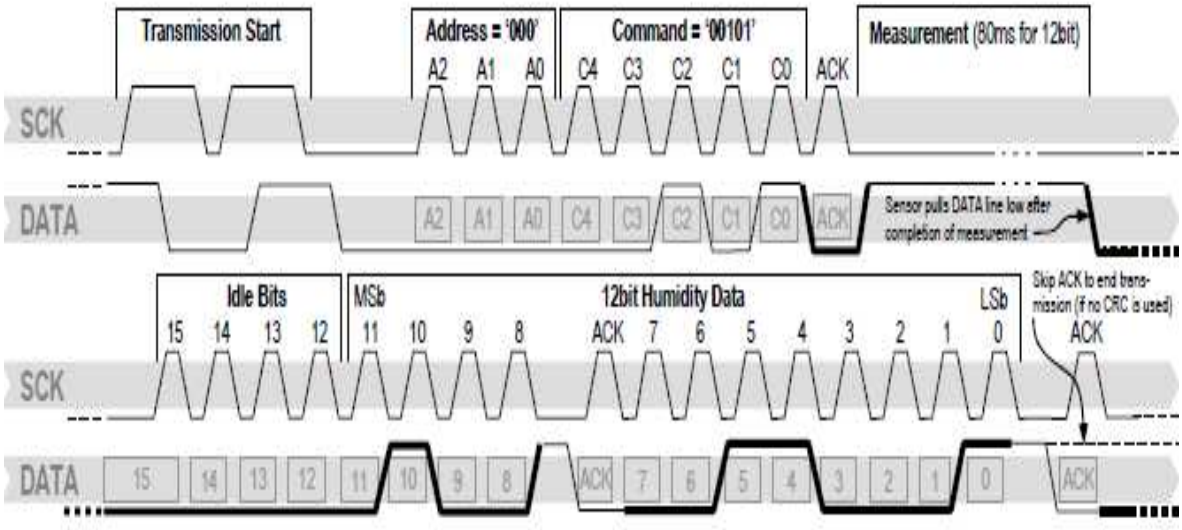


Figura 3-2. Ejemplo de secuencia de medida de HR

A la hora de programar la secuencia de medida, se han definido varias funciones que implementan las distintas acciones que se deben realizar para obtener una lectura:

- **void Reset()**: a pesar de que en la secuencia definida en la *Figura 3-2* no se determina la necesidad de resetear el sensor para obtener una medida correcta, se ha implementado esta función para evitar los errores de comunicación con el sensor. La secuencia de reset se basa en conmutar el SCK nueve o más veces mientras DATA está en alto. Esta secuencia resetea sólo la interfaz, el registro de estado preserva su contenido.

- **void Start_Transmision()**: es la función que implementa la secuencia de comienzo de transmisión. Tal y cómo se aprecia en la *Figura 3-2*, consiste en poner DATA en bajo mientras SCK está en alto, seguido de un pulso en bajo de SCK y finalmente se pone DATA en alto mientras SCK sigue en alto. Cabe destacar, que está función espera 20 ms antes de comenzar la secuencia, ya que es necesario que el sensor se encuentre en “sleep mode” antes de mandar cualquier comando.

- **void ACK()**: es la función que implementa la secuencia que indica que la recepción de comandos y la transmisión de los datos ha sido correcta. Se ejecuta después de transmitir cada byte de información para comprobar que no ha habido ningún error. Consiste en poner DATA en bajo después del flanco de bajada del octavo pulso de SCK.

- **int Medida()**: es la función que realiza una medida de temperatura. Antes de realizar cada lectura se resetea la interfaz del sensor ejecutando la función Reset() y seguido se ejecuta la función Start_Transmision() de manera que ya puede dar comienzo la transmisión.

  A continuación, se manda un comando, un byte, de valor 00000011. Al recibir este comando, el sensor interpreta que tiene que realizar una medida de temperatura. Una vez ejecutada la función ACK() que indica que la transmisión ha finalizado con éxito, es necesario esperar 320ms para que al sensor le dé tiempo de realizar una medida de 14 bits de resolución. Se pueden obtener medidas de 8 y 12 bits siendo el tiempo de espera de 20 y 80 ms respectivamente. Una vez finalizado el tiempo de espera, el sensor transmite dos bytes de información por la línea de datos. De nuevo, es necesario ejecutar la función ACK() después de la transmisión de cada byte.

  Finalmente, los bits transmitidos se almacenan en un array para luego convertirlos en una variable de tipo int, que es la variable de retorno de la función. Realizando ciertas operaciones de conversión indicadas en la hoja de especificaciones del sensor podremos obtener el valor de la temperatura en ºC. Esta última operación se realiza mediante LabVIEW.

- **int Medida2()**: es una función con la misma estructura que la función int Medida(). La principal diferencia respecto a dicha función es que el comando que se manda al sensor es un byte de valor 00000101. Al recibir este comando, el sensor interpreta que tiene que realizar una medida de humedad relativa. La medida de humedad relativa tiene una resolución de 12 bits.

Haciendo uso de estas funciones, se obtienen las medidas de temperatura y humedad relativa a falta de realizar las conversiones requeridas para obtener valores en ºC y %, respectivamente. Más adelante, se explica cómo se han incorporado estas funciones al programa principal.

El código completo se encuentra en el Anexo III.

3.2.2 Programación de la lectura de la caída de tensión en el divisor resistivo

Tal y como se describe en el apartado 2.2.3, se necesita conocer la tensión de salida del divisor resistivo implementado para poder estimar la resistencia del sensor de gas. Para ello, se dispone de un pin analógico del Stellaris Launchpad conectado al punto de tensión de salida.

Para obtener el valor de la tensión se ha realizado, dentro del programa principal, un *analogRead()* del pin analógico conectado, obteniendo así un valor discreto entre 0 y 4095. Después, haciendo uso de la función *Serial.print()*, se ha escrito dicho valor en el puerto serie.

Las operaciones necesarias, definidas en el apartado 2.2.3, para estimar la resistencia del sensor no se han programado con Energia. Tomando el valor discreto escrito en el puerto serie, esas operaciones se han realizado mediante LabVIEW.

3.2.3 Programa principal

El programa principal (ver *Figura 3-3*) se ha implementado dentro de la sección “void loop()” del sketch de manera que se ejecuta continuamente. Se comprueba primero si en el puerto serie hay algún byte disponible para ser leído. En ese caso se procede a la lectura del byte, que en este programa es un carácter transferido al puerto serie a través de la interfaz de usuario, tal y como se explica más adelante. Si el carácter leído es una ‘T’ se procede a realizar una medida de temperatura. Si, en cambio, el carácter recibido es una ‘H’ se realiza una medida de humedad relativa. Si no se cumple ninguno de los dos casos anteriores, se lee el valor de tensión del pin de entrada analógico para estimar luego mediante LabVIEW® la resistencia del sensor de gas.

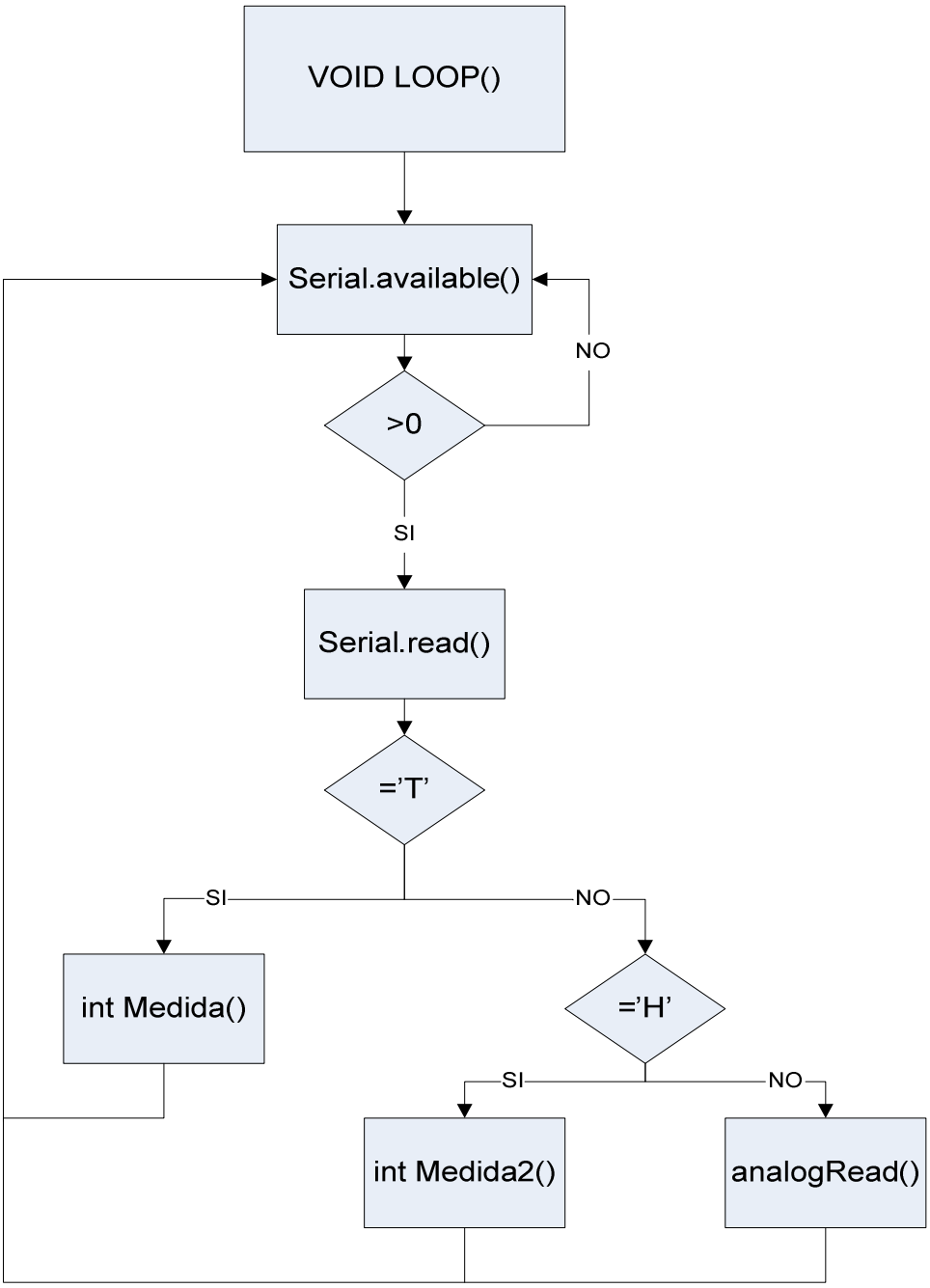


Figura 3-3. Diagrama de flujo del programa principal

3.3 Introducción a LabVIEW®

LabVIEW, Laboratory Virtual Instrument Engineering Workbench, es una plataforma y un entorno de desarrollo para un lenguaje de programación gráfico creado por National Instruments en 1986. Se emplea para la adquisición de datos, el control de instrumentos y la automatización industrial.

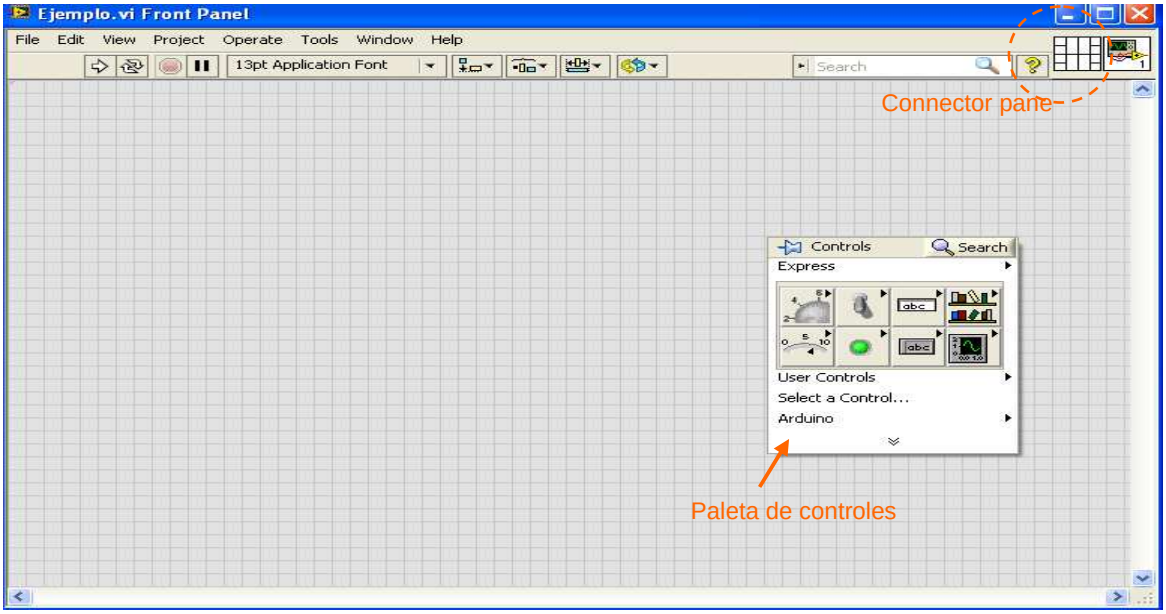


Figura 3-4. Front Panel

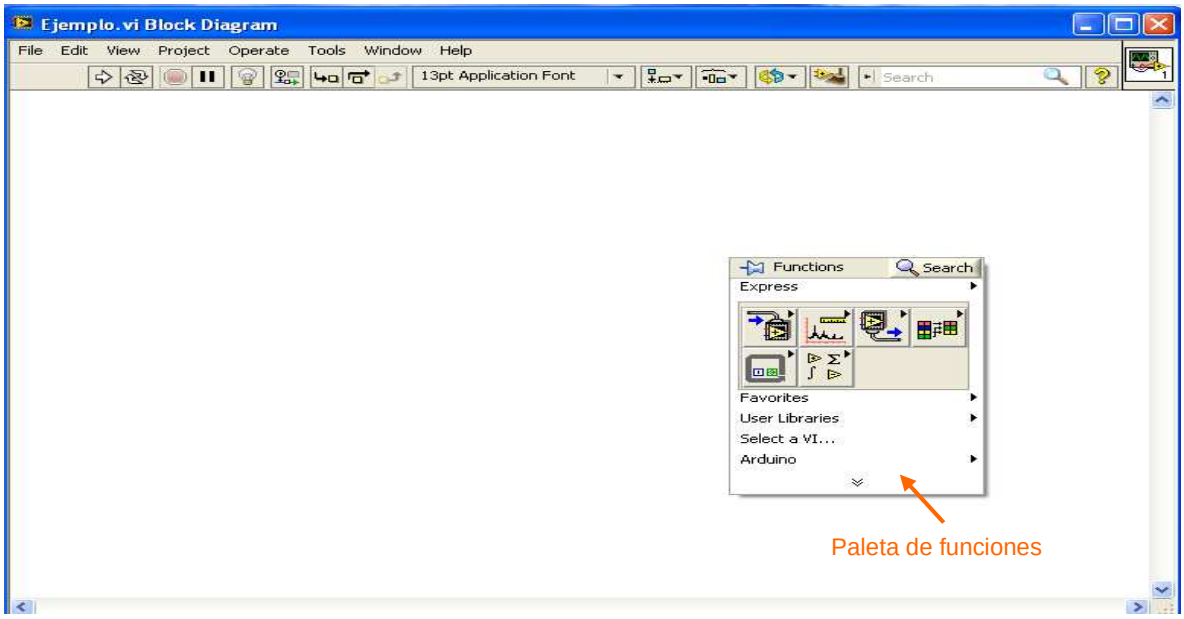


Figura 3-5. Block Diagram

Los programas de LabVIEW están basados en el concepto de “Virtual Instrument”, o VI. Cada VI puede ser un programa ejecutado por sí mismo o también una función que puede ser llamada por otros VIs. Cuando un VI se ejecuta como un programa independiente, el usuario interactúa con una Interfaz Gráfica de Usuario (ver *Figura 3-4*) llamada ‘Front Panel’. El código fuente que controla el programa se especifica en el ‘Block Diagram’ (ver *Figura 3-5*). En el ‘Front Panel’ se dispone de una paleta de controles que permite colocar controles e indicadores. Se dispone, también, de una paleta de funciones en el ‘Block Diagram’ para la implementación del programa.

Los VIs que son llamados por otros VIs pueden ser arrastrados dentro del ‘Block Diagram’ como “subVI”s y conectados al resto de la estructura de flujo de datos haciendo uso del ‘Connector pane’ (esquina superior derecha de la *Figura 3-4*). A través de este panel se definen cómo se pasan los datos exteriores al “subVI”.

3.3.1 Módulos de LabVIEW®

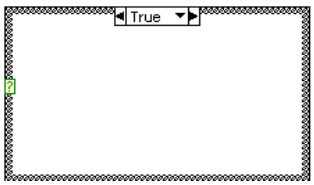
A continuación, se explican los diferentes módulos o funciones de LabVIEW que se han empleado para la implementación del software desarrollado. Por un lado, se presentan los módulos que se han usado en el ‘Block Diagram’ para implementar el código y, por otro lado, los módulos empleados en el ‘Front Panel’ para diseñar la interfaz gráfica de usuario.

- **‘Block Diagram’:**

  - ESTRUCTURAS

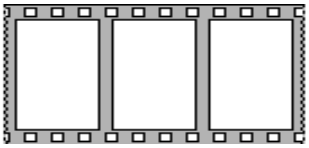
    Las estructuras controlan el flujo del programa.

    “Case Structure”



Esta estructura tiene uno o más subdiagramas o casos. La estructura ejecuta un solo caso cada vez que ejecuta. El valor conectado al terminal de selección determina qué caso se ejecuta, pudiendo ser ese valor de tipo boolean, string, integer o enumerated.

"Flat sequence structure"



Consiste en uno o más subdiagramas que se ejecutan secuencialmente. Los subdiagramas se ejecutan de izquierda a derecha cuando todos los valores conectados a un subdiagrama estén disponibles. Las entradas de un subdiagrama pueden depender de la salida de otro.

"While loop"

Similar a un Ciclo Do o un Ciclo Repeat-Until en lenguajes de programación basados en texto. Repite el subdiagrama que contiene dentro continuamente hasta que la terminal condicional recibe un valor booleano particular que hace cumplir una condición de salida.

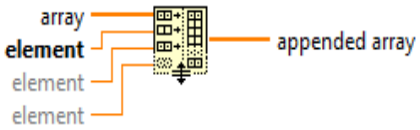
"Feedback Node"

Almacena datos de la ejecución de un VI o de una iteración de bucle para la siguiente ejecución o iteración.

- ARRAY

"Build array"

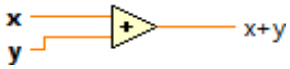
Concatena múltiples arrays o añade elementos a un array n-dimensional.



- NUMERIC

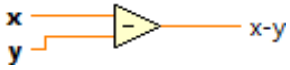
"Add"

Calcula la suma de las entradas.



“Subtract”

Calcula la resta de las entradas.



“Multiply”

Devuelve el producto de las entradas.

“Divide”

Calcula el cociente de las entradas.

“Numeric Constant”

Se emplea para pasar un valor numérico al programa implementado.

“DBL Numeric Constant”

Se emplea para pasar un valor numérico de coma flotante al programa implementado.

- COMPARISON

“Equal”

Devuelve TRUE si *x* es igual a *y*. En caso contrario, devuelve FALSE.

"Less"

Devuelve TRUE si *x* es menor que *y*. En caso contrario, devuelve FALSE.

- BOOLEANS

"Boolean Constant"

Proporciona un valor TRUE o FALSE al programa implementado.

- STRING

"String Constant"

Proporciona un string de texto constante al programa implementado.

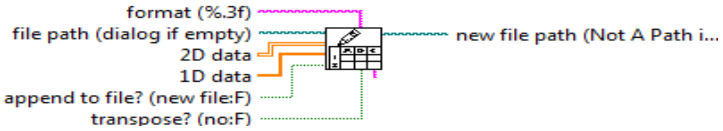
"Fract/Exp String To Number"

Interpreta los caracteres de 0 a 9, plus, minus, e, E y la coma decimal que puedan formar parte de un string como un número de coma flotante, exponencial o fraccional y devuelve dicho número.

- FILE I/O

"Write to Spreadsheet File.vi"

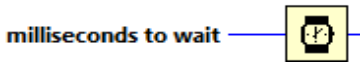
Convierte arrays 1D o 2D en un string de texto y lo escribe en un archivo nuevo o lo añade a un archivo ya existente.



➢ TIMING

"Wait(ms)"

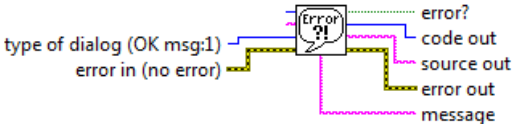
Espera el número de milisegundos especificado y devuelve el valor del temporizador de milisegundos.



➢ DIALOG & USER INTERFACE

"Simple Error Handler.vi"

Indica si se ha producido algún error. Si se da el caso, este VI devuelve una descripción del error.



- **'Front panel':**

➢ CONTROLES

Se emplean para introducir datos para el control de la aplicación y los parámetros de medida. Existen controles de tipo numérico y booleano (ver *Figura 3-6*). En este proyecto se han empleado sólo los de tipo numérico.

- INDICADORES

Se emplean para mostrar datos de diferentes maneras (ver *Figura 3-6*). Pueden ser pequeños monitores que muestran valores numéricos, luces… e incluso gráficos (aunque LabVIEW no los clasifica como indicadores cumplen esa función).

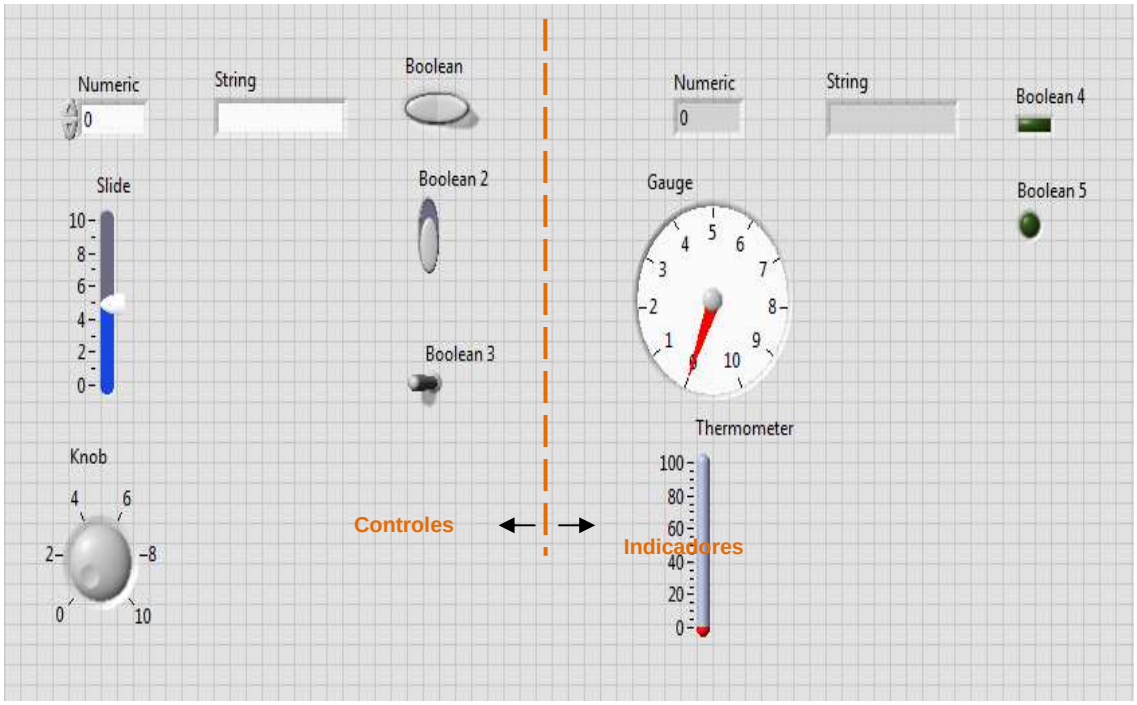


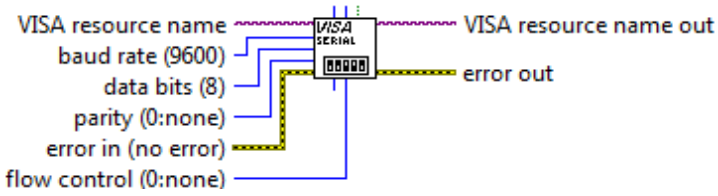
Figura 3-6. Controles e Indicadores

3.3.2 Módulos de NI-VISA

Virtual Instrument Software Architecture (VISA), es un estándar para la configuración y programación de sistemas de instrumentación que comprenden interfaces GPIB, VXI, PXI, Serie, Ethernet y/o USB. NI-VISA proporciona la interfaz de programación entre el hardware y el entorno de LabVIEW. A continuación, se definen los módulos empleados en el programa para adquirir los datos proporcionados por el Stellaris Launchpad a través del puerto serie.

"VISA Configure Serial Port"

Inicializa el puerto serie especificado a través de *VISA resource name* según la configuración fijada por el resto de parámetros.

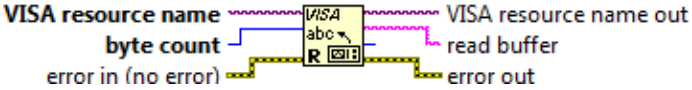


"VISA Write"

Escribe la información disponible en *write buffer* en el puerto serie especificado por *VISA resource name*.

"VISA Read"

Lee un número de bytes, especificados por *byte count*, del puerto serie definido por *VISA resource name* y devuelve la información en *read buffer*.

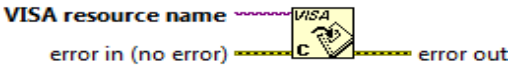


"VISA Bytes at Serial Port"

Devuelve el número de bytes que hay en el buffer de entrada del puerto serie especificado.

"VISA Close"

Cierra la comunicación a través del puerto serie, el cual se define a través de *VISA resource name*.



3.4 Diseño de la interfaz de usuario con LabVIEW®

El software implementado con LabVIEW trata, muestra y registra los datos de medida de los sensores proporcionados por el Stellaris Launchpad a través del puerto serie.

A continuación, se detalla la estructura general del programa final. En primer lugar, se explica el trabajo realizado en el 'Block Diagram' que es donde se ha implementado la funcionalidad de la aplicación. En segundo lugar, se muestra el diseño del 'Front Panel' y cómo el usuario puede interaccionar con él.

3.4.1 'Block Diagram'

Hasta el momento, se tiene un Stellaris Launchpad que ejecuta un programa, implementado con Energia, que toma la información proporcionada por los sensores y la transfiere a través de un puerto serie. Para poder obtener esos datos y tratarlos, LabVIEW se conecta a ese mismo puerto serie tal y como se muestra en la *Figura 3-7*.

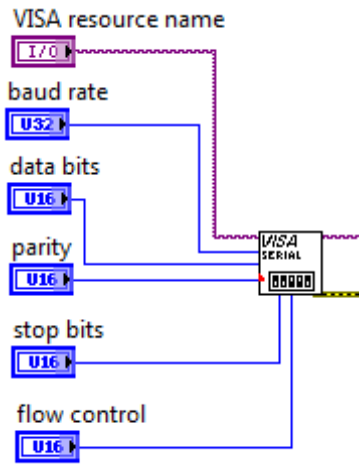


Figura 3-7. Inicialización y configuración del puerto serie

Se ha adelantado en el apartado 3.2.3 que por el diseño del programa implementado con Energia es necesario recibir un carácter mediante el puerto serie para obtener las medidas de temperatura, humedad relativa o tensión. Por ello, después de inicializar el puerto serie, se transfiere dicho carácter de la aplicación de LabVIEW al puerto serie.

Una vez se haya transferido una ‘T’, una ‘H’ o una ‘R’, el programa espera 380 ms para que los sensores y el Stellaris Launchpad tengan tiempo de reaccionar y realizar las medidas oportunas. A continuación, con el modulo NI-VISA adecuado se lee del puerto serie la información proporcionada por los sensores. Dependiendo del carácter que se haya mandado al puerto serie a través de LabVIEW, los datos recibidos se han tratado de diferente manera. Así pues se diferencian tres casos distintos:

- **Si se ha escrito una ‘T’ en el puerto serie**, los datos que el Stellaris Launchpad transfiere y que LabVIEW lee son los correspondientes a la medida de temperatura. Para obtener la temperatura en ºC se ha realizado una conversión de los datos haciendo uso de los módulos numéricos que proporciona LabVIEW y siguiendo la fórmula de conversión proporcionada por la hoja de especificaciones del SHT71 (ver *Figura 3-8*). Finalmente, se registran los datos en un archivo.

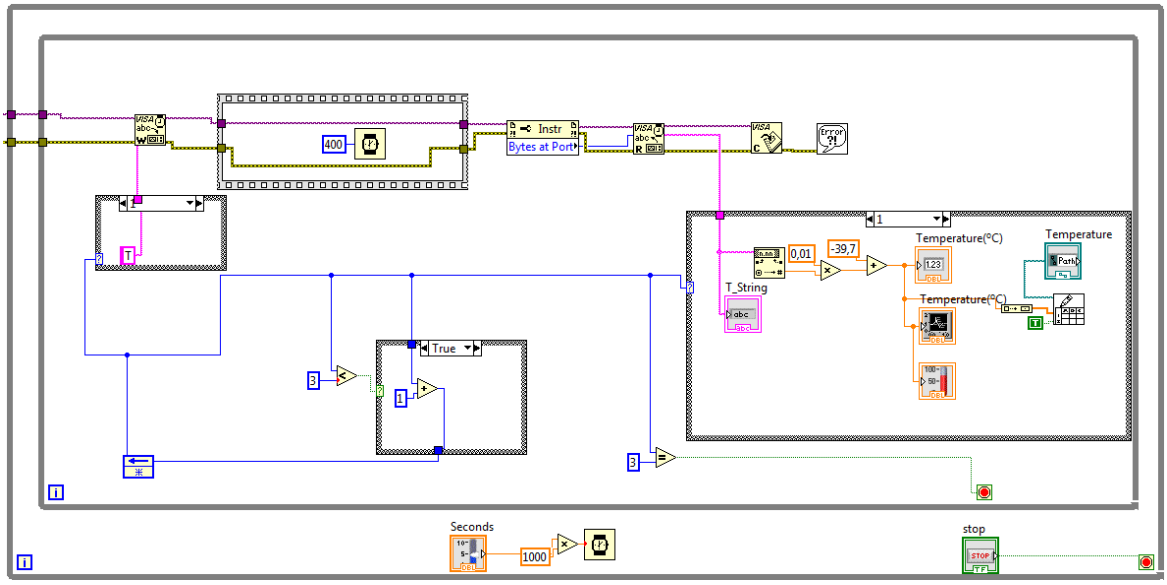


Figura 3-8. Caso 1 del “case estructure”, tratamiento de los datos de temperatura

- **Si se ha escrito una ‘H’ en el puerto serie**, los datos que el Stellaris Launchpad transfiere y que LabVIEW lee son los correspondientes a la medida de humedad relativa. Para obtener la humedad relativa en % se ha llevado a cabo un procedimiento igual al empleado en el caso anterior, pero implementado la fórmula de conversión adecuada para la humedad relativa (ver *Figura 3-9*). Finalmente, se registran los datos en un archivo.

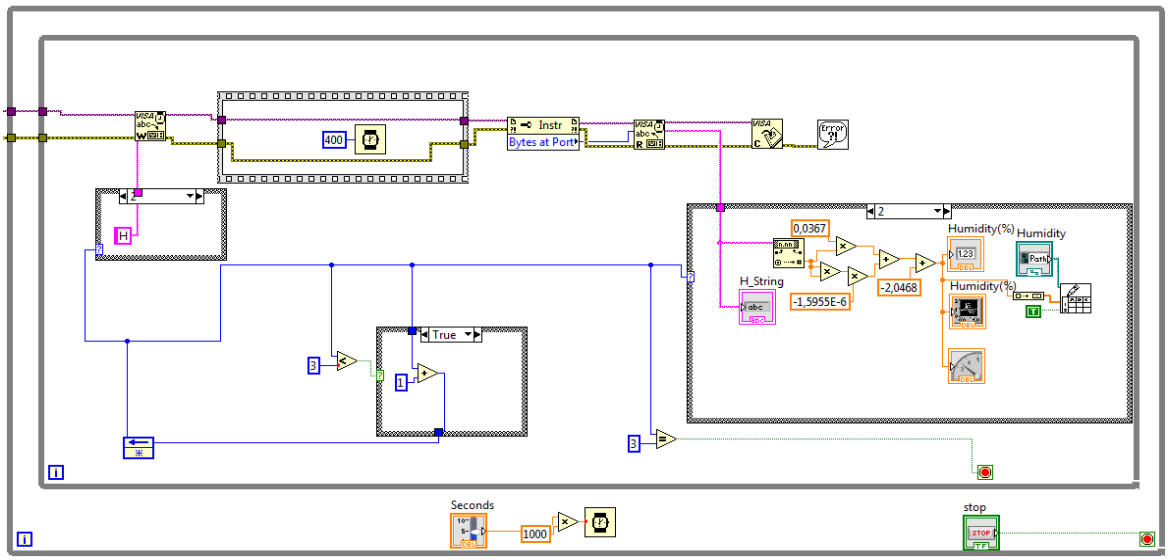


Figura 3-9. Caso 2 del "case estructure", tratamiento de los datos de humedad relativa

- **Si se ha escrito una 'R' en el puerto serie**, los datos que el Stellaris Launchpad transfiere y que LabVIEW lee son los correspondientes a la tensión del pin analógico conectado al divisor resistivo imp lementado con el fin de estimar la resistencia del sensor conductométrico de gas. Partiendo del dato de tensión, se estima la resistencia realizando las operaciones especificadas en el apartado 2.2.3 (ver *Figura 3-10*). Finalmente, se registran los datos en un archivo.

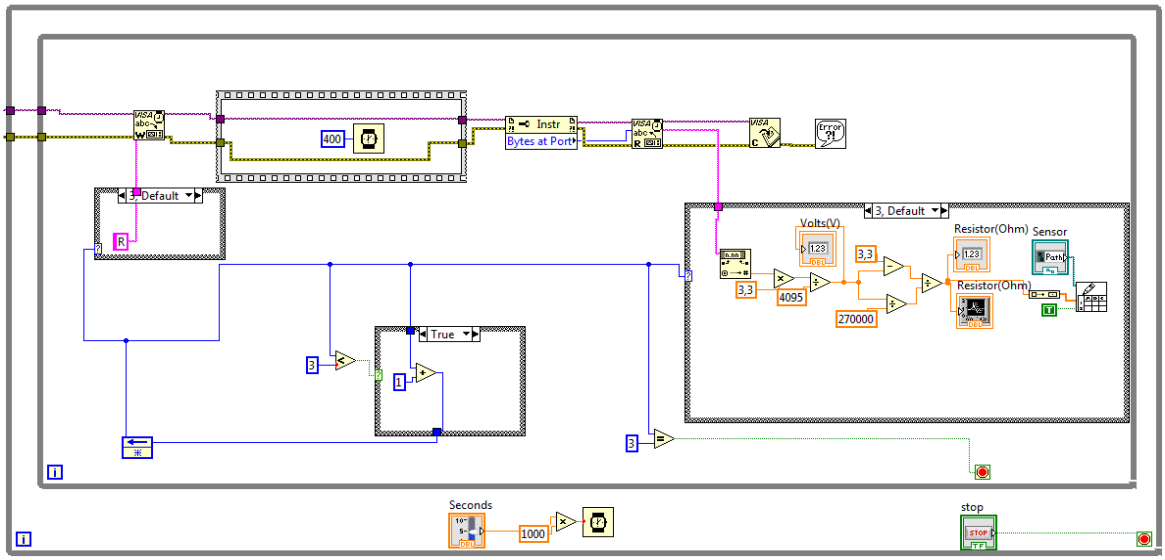


Figura 3-10. Caso 3 del "case estructure", estimación de la resistencia del sensor de gas

En las figuras anteriores, se aprecia como el programa principal está dentro de dos “While Loop”. Con el primer “While Loop” que alberga el otro en su interior, se consigue que las medidas de temperatura, humedad relativa y resistencia se realicen cada cierto tiempo establecido por el usuario. El segundo “While Loop” hace que cada vez que se tengan que realizar las medidas, estas se efectúen las 3 a la vez.

3.4.2 ‘Front Panel’

La Interfaz Gráfica de Usuario se ha diseñado tal y como se muestra en la *Figura 3-11*. Se dispone de gráficos e indicadores numéricos en los cuales el usuario puede ver los datos obtenidos. Además, hay controles para configurar el puerto serie, para determinar el tiempo de espera entre medidas y para especificar los archivos donde se quieren guardar los datos obtenidos.

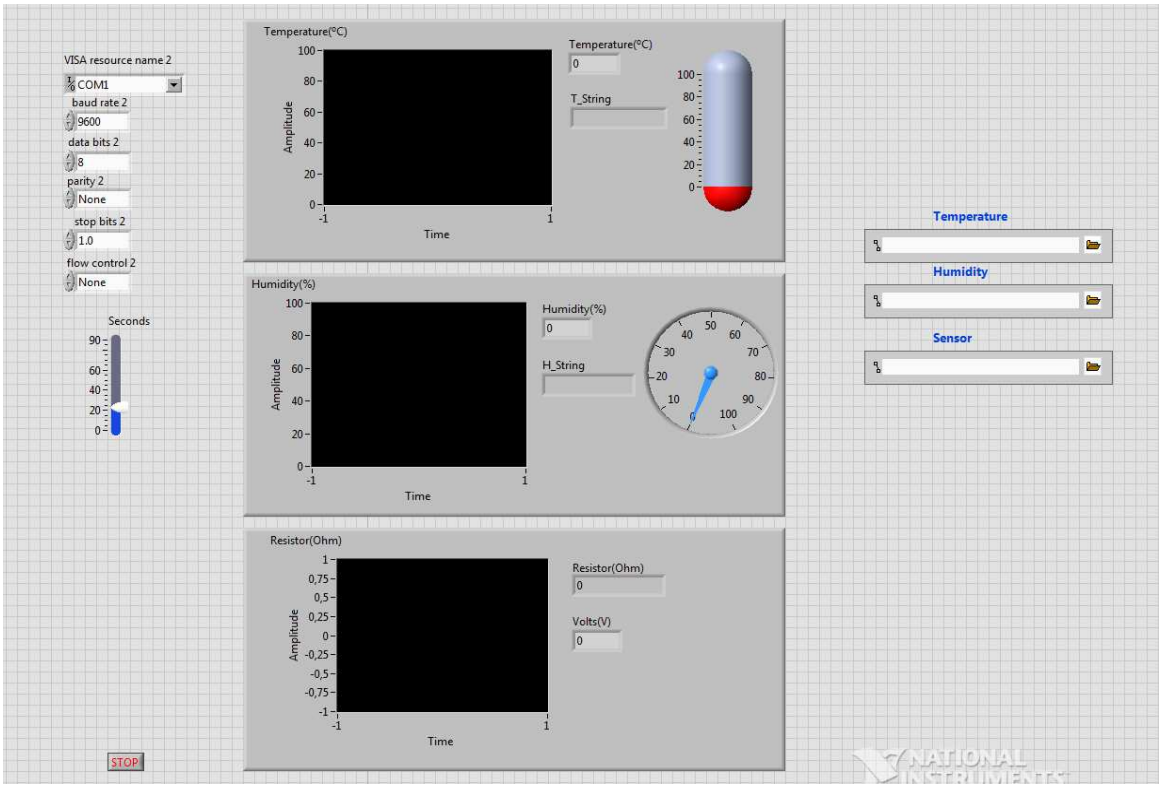


Figura 3-11. Interfaz Gráfica de Usuario

4. MEDIDAS Y RESULTADOS

En este capítulo se explica el procedimiento seguido en las medidas, junto con los resultados obtenidos para cada ambiente en el que se han realizado los ensayos.

4.1 Medidas en atmosfera de laboratorio

4.1.1 Procedimiento

Para las medidas en atmosfera de laboratorio, se tiene, por un lado, una plataforma resultante del proyecto INTASENSE que integra los sensores con la micro-fluídica (ver *Figura 4-1*). Por otro lado, se dispone del Stellaris Launchpad y del PC, conectados entre sí por un cable USB, y de la fuente de alimentación. Para conectar la plataforma con el Stellaris Launchpad y la fuente de alimentación se ha diseñado mediante EAGLE 6.5.0 una placa con diferentes conectores dispuestos tal y como se aprecia en la *Figura 4-2*. En este diseño se han incluido también la resistencia de "pull-up" necesaria para el correcto funcionamiento del SHT71 y la resistencia de 107 kΩ que forma parte del divisor resistivo.

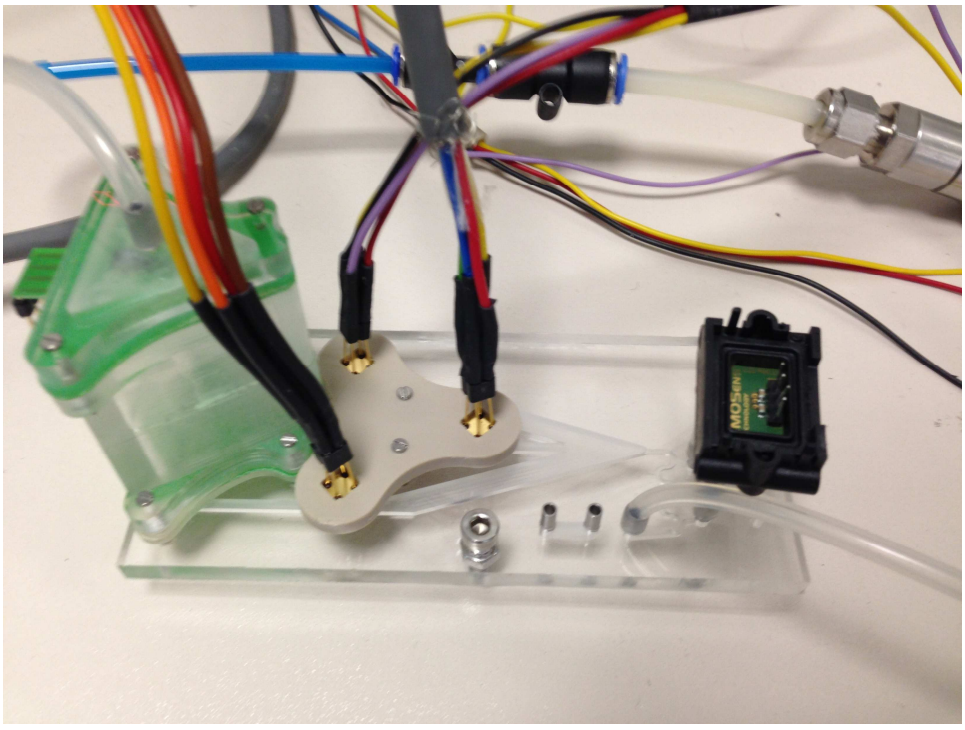


Figura 4-1. Plataforma de INTASENSE que integra los sensores con la micro-fluídica

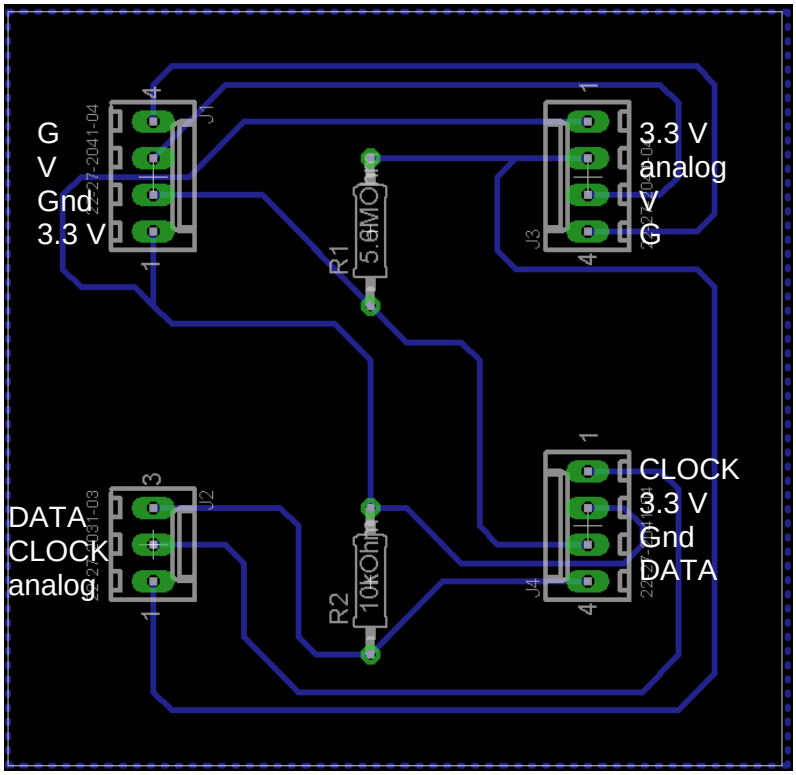


Figura 4-2. Layout de la placa diseñada mediante EAGLE

4.1.2 Resultados obtenidos del estudio de humedad relativa

En la *Figura 4-1* se aprecia que la plataforma de INTASENSE dispone de una pequeña cámara que contiene en su interior un material (gel de sílice) que filtra la humedad del fluido que se introduce. Tiene, también, dos sensores SHT71 colocados de manera que uno mide la temperatura y la humedad relativa del fluido entrante antes de pasar por el filtro y el otro mide los mismos parámetros a la salida. Así, empleando un segundo Stellaris Launchpad que ejecuta el mismo programa y duplicando el código implementado con LabVIEW para tomar datos de dos puertos serie (ver *Figura 4-3*), ha sido posible realizar un estudio de humedad relativa que determina la capacidad del gel de sílice de filtrar la humedad y cómo afectan los cambios en la humedad relativa al comportamiento del sensor de gas conductométrico.

Los cambios en la humedad relativa del aire a la salida del filtro debido a los cambios en la entrada no son apreciables hasta que el material del filtro no empieza a saturar. Por ello, antes de comenzar el ensayo, se ha introducido a la plataforma un 80% aire húmedo hasta que se ha empezado a apreciar un cambio en la humedad relativa del aire de salida.

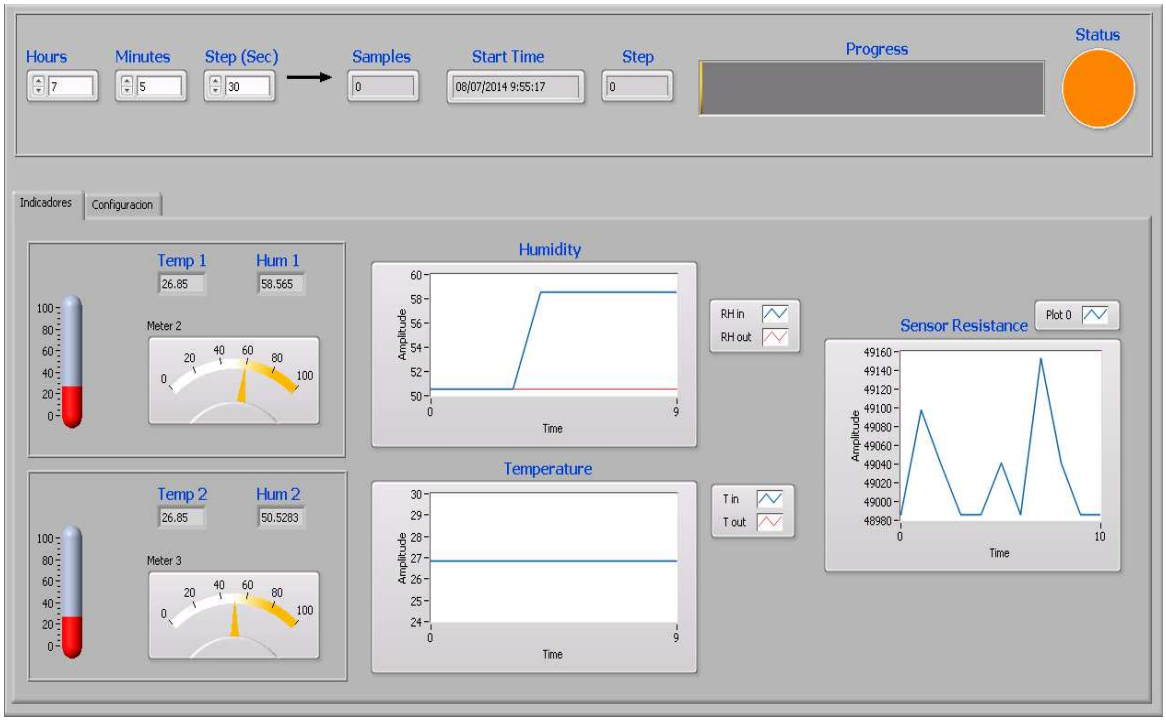


Figura 4-3. 'Front Panel' modificado para realizar el estudio de humedad relativa

El ensayo se basa en introducir a la plataforma pulsos de aire de 12 horas de duración con diferentes valores de humedad relativa. La secuencia realizada se define en la *Tabla 4-1*.

PULSOS	AIRE SECO	AIRE HÚMEDO	DURACIÓN
1	60%	40%	10 min (para estabilizar)
2	40%	60%	12 h (720 min)
3	80%	20%	12 h (720 min)
4	20%	80%	12 h (720 min)
5	100%	0%	12 h (720 min)
6	0%	100%	12 h (720 min)
7	60%	40%	12 h (720 min)

Tabla 4-1. Secuencia de pulsos de aire

En la *Figura 4-3* se pueden observar las medidas de humedad relativa realizadas por los sensores SHT71 a la entrada y a la salida a lo largo del ensayo. Al principio del ensayo la humedad del aire del interior de la cámara es menor que la del aire entrante. Por ello, las esferas de gel de sílice absorben parte de la humedad entrante haciendo que a la salida se reduzca el porcentaje de humedad relativa. Al reducir el porcentaje de aire húmedo entrante al 20%, se reduce la humedad relativa a la salida pero en menor medida, ya que al ser la humedad del interior mayor que la del exterior el gel de sílice expulsa parte de la humedad absorbida.

Al aumentar el porcentaje de aire húmedo entrante al 80%, el cambio en el porcentaje de humedad relativa a la salida se da con mucho retardo. Esto se debe a que en el pulso anterior se ha extraído humedad del gel de sílice y, por ello, el filtro está lejos de estar saturado. Una vez el gel de sílice está suficientemente saturado se aprecia el cambio en la humedad relativa a la salida.

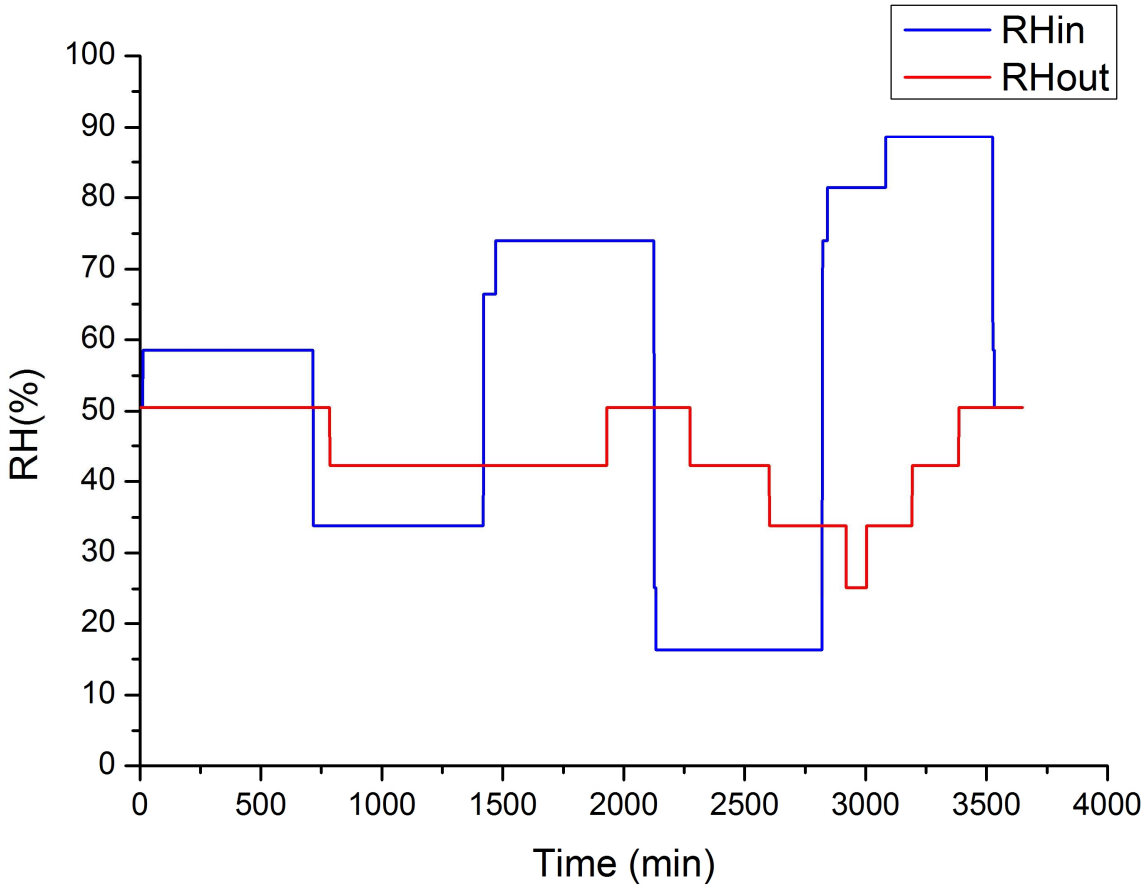


Figura 4-4. Medidas de humedad relativa a la entrada y a la salida de la plataforma

En el siguiente pulso se ha introducido a la plataforma un 100% de aire seco. Eso supone que la humedad relativa del aire de entrada debería de ser de 0%, pero debido a factores desconocidos se observa una humedad relativa de alrededor del 15%. Con esta bajada de la humedad relativa a la entrada, se reduce también, en menor medida, la humedad relativa a la salida. Se sospecha que aumentando la duración del pulso, los valores de humedad relativa a la entrada y a la salida se igualarían.

Finalmente, se ha inyectado a la plataforma un 100% de aire húmedo. De manera escalonada, la humedad relativa a la salida ha ido aumentando hasta alcanzar un valor de alrededor del 50%. Este valor se ha mantenido al bajar al 40% la cantidad de aire húmedo entrante.

En cuanto a los cambios que se dan en la resistencia del sensor de óxido de níquel debido a los cambios en la humedad relativa del aire que lo rodea, se puede concluir que, las variaciones de humedad relativa no provocan variaciones relevantes en la resistencia, ya que a pesar de los pequeños picos que se producen el sensor siempre recupera el valor inicial de la resistencia.

4.1.3 Resultados obtenidos del ensayo con gas

Para determinar cómo reacciona el sensor de óxido de níquel ante gases que influyen en la mala calidad del aire se ha realizado un ensayo introduciendo NO_2 a la plataforma de INTASENSE. En este ensayo, para obtener las medidas de temperatura, humedad relativa y resistencia del sensor, se han empleado los programas de Energia y LabVIEW originales.

En la *Figura 4-5* se muestran los valores que ha ido tomando la resistencia del sensor durante el ensayo. Se han introducido 7 pulsos de gas que se definen en la *Tabla 4-2*. La temperatura y la humedad relativa, proporcionados por el sensor SHT71, se han mantenido constantes durante todo el ensayo a 29.4 ºC y 7.2 %, respectivamente.

pulso 1	20 ppm
pulso 2	10 ppm
pulso 3	5 ppm
pulso 4	1 ppm
pulso 5	500 ppb
pulso 6	100 ppb
pulso 7	50 ppb

Tabla 4-2. Pulso de gas durante el ensayo

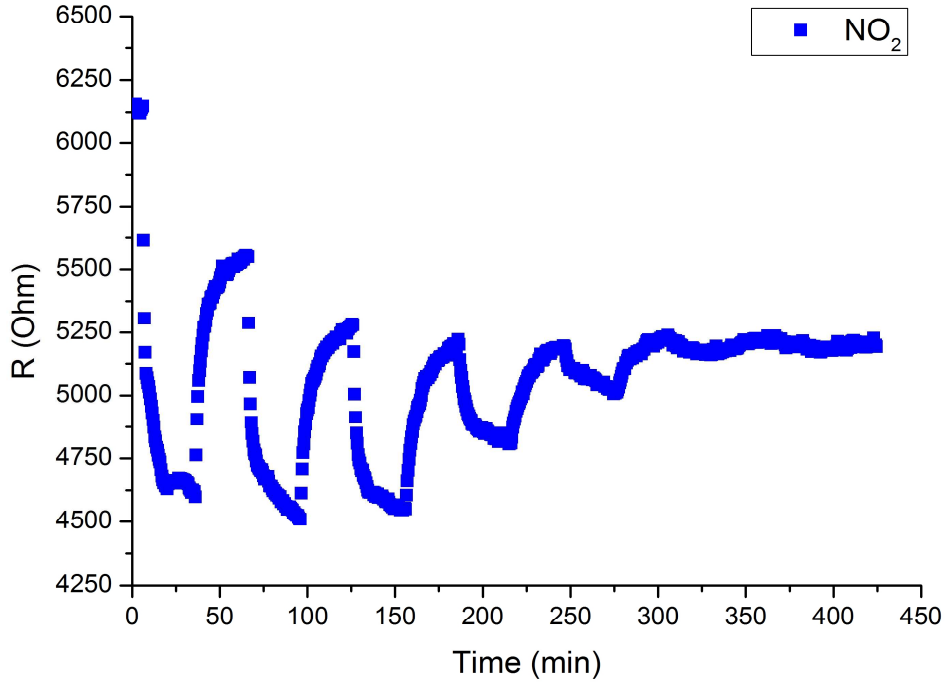


Figura 4-5. Variaciones de la resistencia del sensor de óxido de níquel en presencia de NO2

Se aprecia cómo al detectar cada pulso de gas la resistencia del sensor de óxido de níquel baja. En ausencia de gas el valor de la resistencia tiende a recuperar sus valores habituales para volver a bajar al detectar el siguiente pulso de gas. A medida que las concentraciones disminuyen, la recuperación de la línea base se va haciendo más costosa.

4.2 Medidas en condiciones representativas de aplicaciones finales

4.2.1 Diseño y simulación de un ambiente de trabajo real

Para simular un ambiente lo más cercano posible al ambiente en el que va a trabajar el sensor de gas conductométrico, se quiere introducir la plataforma que integra los sensores y la micro-fluídica dentro de una cámara. La cámara rectangular, con unas dimensiones de 380x240x130 mm, simula una habitación pequeña. La placa de conectores y el Stellaris Launchpad se sitúan fuera.

En primer lugar, es imprescindible crear una entrada y una salida para el flujo de aire. Para ello, y con el fin de obtener un flujo de aire bien repartido por toda la cámara para que los resultados del sistema no sean alterados, se han simulado mediante COMSOL varias disposiciones de la entrada y la salida. En estas simulaciones se ha asumido que el flujo de aire es de 400 cm^3/min y el radio de los tubos disponibles de 6 mm. A continuación, se muestran dos de los resultados más significativos.

- **Entrada y salida en la superficie superior**: tanto la entrada como la salida se encuentran sobre el eje central y a 40 mm del borde. Introduciendo un flujo de aire desde el exterior, en la *Figura 4-2* se puede ver cómo se reparte ese aire dentro de la cámara.

  Con esta disposición se tienen grandes turbulencias a la entrada y el flujo de aire en el resto de la cámara es muy débil. Por ello, se considera que no es una disposición de la entrada y la salida adecuada para obtener unos resultados válidos.

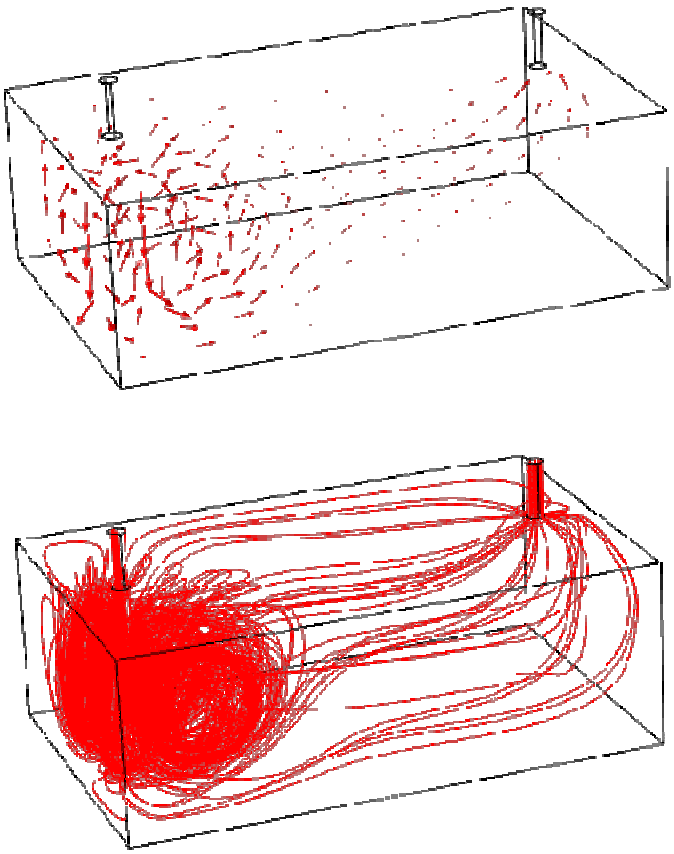


Figura 4-6. Flujo de aire dentro de la cámara con entrada y salida en la superficie superior

- **Entrada en una de las superficies laterales y salida en la opuesta**: la entrada se encuentra en el centro de una de las superficies laterales y la salida está centrada con ella en la superficie opuesta. Con esta disposición, en la *Figura 4-3* se ve cómo se reparte el flujo de aire dentro de la cámara.

  Se consigue un flujo bastante lineal que se reparte de manera aceptable por todo el volumen de la cámara. Por ello, se considera esta disposición de la entrada y la salida apropiada para realizar las medidas con el sensor.

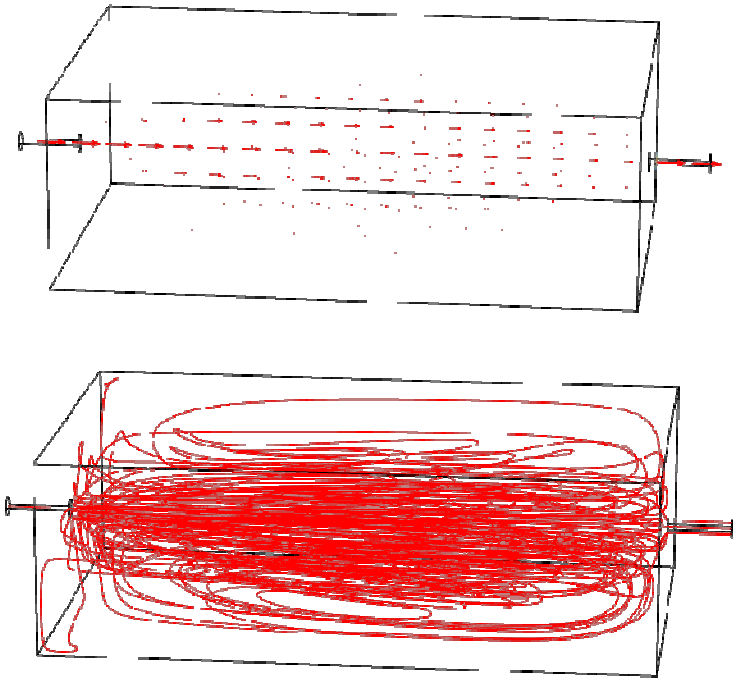


Figura 4-7. Flujo de aire dentro de la cámara con entrada y salida en las superficies laterales

En segundo lugar, es necesario crear unas entradas para los cables eléctricos que van conectados a los sensores. Se requieren dos entradas, una para el cable con el conector que se conecta al sensor de gas conductométrico y otro para el cable con el conector que se conecta al sensor de temperatura y humedad relativa. Se ha decidido colocar estas entradas en la misma superficie frontal a 3 cm del borde inferior y con una separación entre ellas de 18 cm.

4.2.2 Resultados obtenidos

La cámara, de aproximadamente 12 litros de volumen, tarda en llenarse con un flujo máximo de 400 cm^3/min alrededor de 30 minutos. Esto supone que para poder ver cambios en la resistencia del sensor provocados por el fluido entrante los ensayos tiene que ser de muy larga duración. A causa de ello y al no disponer del tiempo suficiente, se ha decidido que en este proyecto no se van a realizar medidas dentro de la cámara.

5. CONCLUSIONES Y FUTUROS TRABAJOS

En este proyecto se ha desarrollado un sistema compacto de medida para sensores de gas conductométricos y a lo largo de esta memoria se han presentado detalladamente tanto el hardware como el software implementados.

En este apartado se presentan las conclusiones derivadas de los resultados obtenidos en los ensayos y los trabajos que en un futuro se pueden realizar para mejorar el sistema de medida.

5.1 Conclusiones

En esta memoria desarrollada para el proyecto de implementación de sistema compacto de medida para sensores de gas conductométricos, se ha desarrollado primero un análisis de los componentes hardware empleados en el sistema con el fin de entender la razón por la que se ha optado por ellas para llevar a cabo el proyecto.

Sobre el software desarrollado con Energia se puede concluir que se ha desarrollado en un lenguaje de programación libre que permite modificar y trabajar con el programa en cualquier sistema informático. Además, su similitud con el lenguaje Arduino permite ejecutar, con unas ligeras modificaciones, el mismo programa en una placa Arduino.

El software implementado en LabVIEW ha sido optimizado lo máximo posible con el fin de crear una interfaz clara y simple para el usuario.

El sistema completo ha sido testeado y validado con ensayos en ambientes de laboratorio. Con los ensayos realizados se puede concluir que el sistema de medida funciona correctamente. El tratamiento, la muestra y el registro de las medidas han sido correctos en todo momento. En cuanto a la recogida de datos, han surgido ciertos problemas debidos al hardware, sobre todo debido a la placa de conectores diseñada mediante EAGLE 6.5.0, pero estos problemas se han podido solucionar.

Resumiendo, con el trabajo realizado se han cumplido los objetivos de este proyecto. Se ha diseñado e implementado una plataforma de comunicaciones de bajo coste y bajo consumo que recoja las medidas realizadas por los sensores y se han realizado ensayos con la plataforma prototipo de INTASENSE. También, se ha diseñado

y simulado un demostrable para realizar ensayos en condiciones representativas de aplicaciones finales, aunque no se han podido realizar estos ensayos.

5.2 Futuros trabajos

Actualmente, el Stellaris Launchpad está programado para obtener el valor de la resistencia de un solo sensor de gas conductométrico. El principal trabajo futuro es lograr medir la resistencia de varios sensores de gas de materiales sensibles diferentes con una misma plataforma comercial, como puede ser el Stellaris Laucnhpad.

Para ello, se pueden programar varios pines analógicos del Stellaris Launchpad para que midan las tensiones de salida de los diferentes divisores resistivos implementados. Las resistencias que además de los sensores forman parte del divisor han de tomar valores diferentes dependiendo de la resistencia base del sensor.

Por otra parte, interesa crear en un futuro un sistema más flexible que sea capaz de medir la resistencia de sensores de diferentes materiales sensibles utilizando un mismo pin analógico. Tal y como se observa en la *Figura 5-1* se implementa un sistema con multiplexor que elije, dependiendo de la tensión de salida, la resistencia que mejor se adapta al sensor de gas presente.

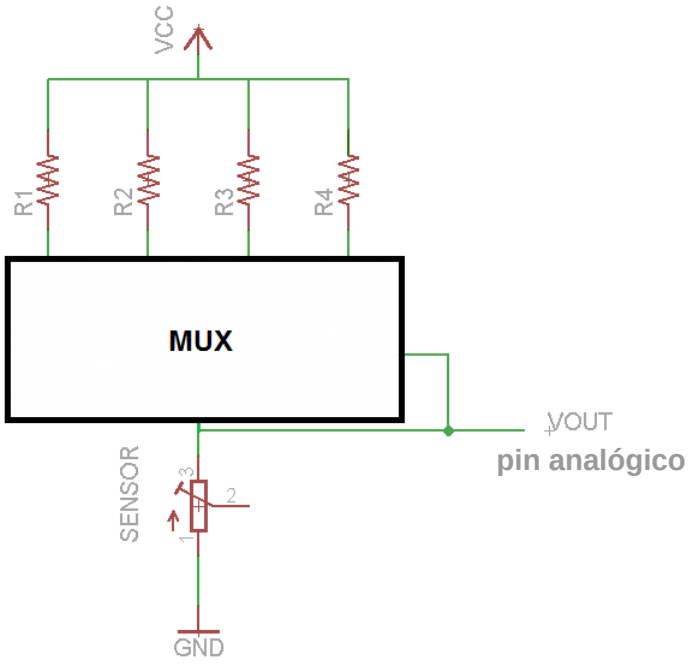


Figura 5-1. Sistema de medida de diferentes sensores de gas conductométricos con multiplexor

El software implementado está optimizado para el sistema de medida implementado para este proyecto. Otra futura línea de trabajo se basaría en expandirlo y reprogramarlo con el fin de obtener un software más estándar, modificable y comercial.

Finalmente, es necesario realizar en un futuro medidas en condiciones representativas de aplicaciones finales del sensor de gas conductométrico empleando la cámara que se ha diseñado y simulado. A través de estos ensayos se podrá definir el funcionamiento del sensor de una manera más concreta.

REFERENCIAS

[1] Paulou, J., Lonsdale, J., Jamieson, M., Neuweg, I., Trucco, P., Maio, P., Blom, M. and Warringa, G., *Financing the energy renovation of buildings with Cohesion Policy funding*, European Comission, 2014.

[2] Janssen, R., *Towards Energy Efficient Buildings in Europe,* EUROACE, 2004.

[3] *Estimated deaths and DALYs attributable to selected environmental risk factors, by WHO member state*, WHO 2002. Disponible en: http://www.who.int/entity/quantifying_ehimpacts/countryprofilesebd.xls

[4] *ibid*, WHO 2002.

[5] Hansen, C. and Selte, H., *Air pollution and Sick-leaves*, Env. and Resource Economics, Vol. 16, pp31-50, 2000

[6] UE project, grant no. 285037, INTATSENSE, *INTegrated Air quality SENSor for Energy efficient environment control* under the program EeB.ENV.2011.3.1.5-1.

[7] http://energia.nu/Guide_StellarisLaunchPad.html

[8] Stellaris® LM4F120 Launchpad Evaluation Kit User's Manual. Disponible en:

http://www.ti.com/tool/EK-LM4F120XL

[9] Gonzalez-Chavarri, J., *Sensor based on advanced micro and nano technologies for safety air quality applications*, TECNUN-CEIT.

[10] http://energia.nu/

ANEXO I

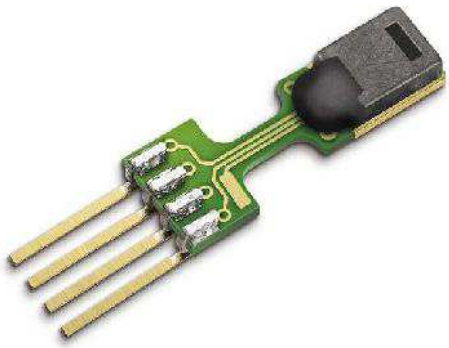
Hoja de especificaciones SHT71

SENSIRION
THE SENSOR COMPANY

Datasheet SHT7x (SHT71, SHT75)

Humidity and Temperature Sensor IC

- Fully calibrated
- Digital output
- Low power consumption
- Excellent long term stability
- Pin type package – easy integration



Product Summary

SHT7x (including SHT71 and SHT75) is Sensirion's family of relative humidity and temperature sensors with pins. The sensors integrate sensor elements plus signal processing in compact format and provide a fully calibrated digital output. A unique capacitive sensor element is used for measuring relative humidity while temperature is measured by a band-gap sensor. The applied CMOSens® technology guarantees excellent reliability and long term stability. Both sensors are seamlessly coupled to a 14bit analog to digital converter and a serial interface circuit. This results in superior signal quality, a fast response time and insensitivity to external disturbances (EMC).

Each SHT7x is individually calibrated in a precision humidity chamber. The calibration coefficients are programmed into an OTP memory on the chip. These coefficients are used to internally calibrate the signals from the sensors. The 2-wire serial interface and internal voltage regulation allows for easy and fast system integration. The small size and low power consumption makes SHT7x the ultimate choice for even the most demanding applications.

SHT7x is supplied on FR4 with pins which allows for easy integration or replacement. The same sensor is also available as surface mountable packaging (SHT1x) or on flex print (SHTA1).

Dimensions

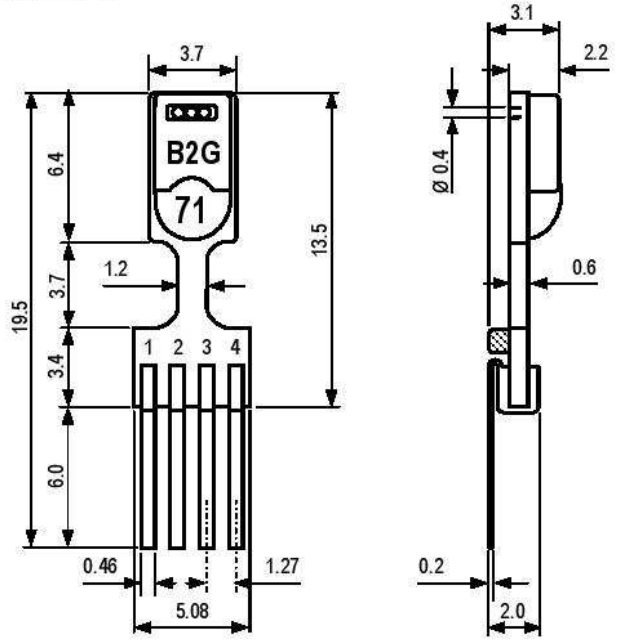


Figure 1: Drawing of SHT7x (applies to SHT71 and SHT75) sensor packaging, dimensions in mm (1mm = 0.039inch). Contact assignment: 1: SCK, 2: VDD, 3: GND, 4: DATA. Hatched item on backside of PCB is a 100nF capacitor – see Section 2.1 for more information.

Sensor Chip

SHT7x V4 – for which this datasheet applies – features a version 4 Silicon sensor chip. Besides a humidity and a temperature sensor the chip contains an amplifier, A/D converter, OTP memory and a digital interface. V4 sensors can be identified by the alpha-numeric traceability code on the sensor cap – see example "B2G" code on Figure 1.

Material Contents

While the sensor is made of a CMOS chip the sensor housing consists of an LCP cap with epoxy glob top on an FR4 substrate. Pins are made of a Cu/Be alloy coated with Ni and Au. The device is fully RoHS and WEEE compliant, thus it is free of Pb, Cd, Hg, Cr(6+), PBB and PBDE.

Evaluation Kits

For sensor trial measurements, for qualification of the sensor or even experimental application (data logging) of the sensor there is an evaluation kit *EK-H4* available including SHT71 (same sensor chip as SHT1x) and 4 sensor channels, hard and software to interface with a computer. For other evaluation kits please check www.sensirion.com/humidity.

Sensor Performance

Relative Humidity

Parameter	Condition	min	typ	max	Units
Resolution[1]		0.4	0.05	0.05	%RH
		8	12	12	bit
Accuracy[2] SHT71	typ		±3.0		%RH
	max	see Figure 2			
Accuracy[2] SHT75	typ		±1.8		%RH
	max	see Figure 2			
Repeatability			±0.1		%RH
Hysteresis			±1		%RH
Nonlinearity	raw data		±3		%RH
	linearized		<<1		%RH
Response time[3]	tau 63%		8		s
Operating Range		0		100	%RH
Long term drift[4]	normal		< 0.5		%RH/yr

Temperature

Parameter	Condition	min	typ	max	Units
Resolution[1]		0.04	0.01	0.01	°C
		12	14	14	bit
Accuracy[2] SHT71	typ		±0.4		°C
	max	see Figure 3			
Accuracy[2] SHT75	typ		±0.3		°C
	max	see Figure 3			
Repeatability			±0.1		°C
Operating Range		-40		123.8	°C
		-40		254.9	°F
Response Time [6]	tau 63%	5		30	s
Long term drift			< 0.04		°C/yr

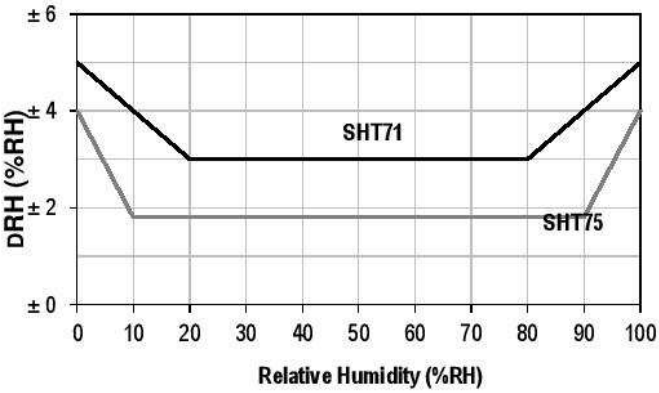


Figure 2: Maximal RH-tolerance at 25°C per sensor type.

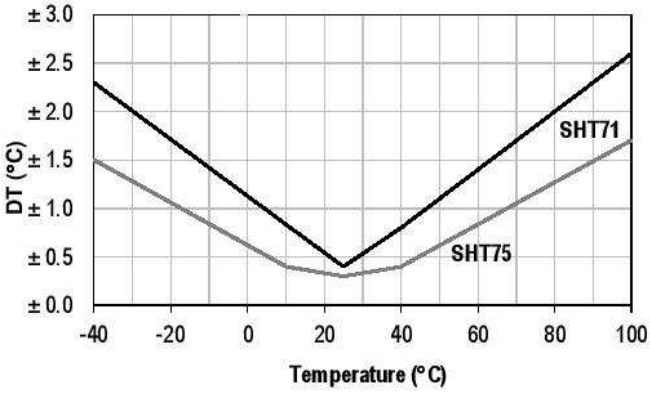


Figure 3: Maximal T-tolerance per sensor type.

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 – 80°C peak), 20 – 60%RH				

Packaging Information

Sensor Type	Packaging	Quantity	Order Number
SHT71	Tape Stripes	50	1-100092-04
SHT75	Tape Stripes	50	1-100071-04

This datasheet is subject to change and may be amended without prior notice.

[1] The default measurement resolution of is 14bit for temperature and 12bit for humidity. It can be reduced to 12/8bit by command to status register.

[2] Accuracies are tested at Outgoing Quality Control at 25°C (77°F) and 3.3V. Values exclude hysteresis and are only applicable to non-condensing environments.

[3] Time for reaching 63% of a step function, valid at 25°C and 1 m/s airflow.

[4] Value may be higher in environments with high contents of volatile organic compounds. See Section 1.3 of Users Guide.

[5] Values for VDD=3.3V at 25°C, average value at one 12bit measurement per second.

[6] Response time depends on heat capacity of and thermal resistance to sensor substrate.

Users Guide SHT7x

1 Application Information

1.1 Operating Conditions

Sensor works stable within recommended normal range – see Figure 4. Long term exposures to conditions outside normal range, especially at humidity >80%RH, may temporarily offset the RH signal (+3 %RH after 60h). After return to normal range it will slowly return towards calibration state by itself. See Section 1.4 "Reconditioning Procedure" to accelerate eliminating the offset. Prolonged exposure to extreme conditions may accelerate ageing.

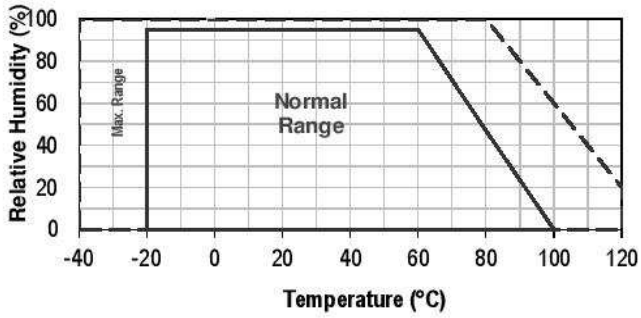


Figure 4: Operating Conditions

1.2 Sockets and Soldering

For maintain high accuracy specifications the sensor shall not be soldered. Sockets may be used such as "Preci-dip / Mill-Max R851-83-004-20-001" or similar.

Standard wave soldering ovens may be used at maximum 235°C for 20 seconds. For manual soldering contact time must be limited to 5 seconds at up to 350°C[7].

After wave soldering the devices should be stored at >75%RH for at least 12h to allow the polymer to re-hydrate. Alternatively the re-hydration process may be performed at ambient conditions (>40%RH) during more than 5 days.

In no case, neither after manual nor wave soldering, a board wash shall be applied. In case of application with exposure of the sensor to corrosive gases the soldering pads of pins and PCB shall be sealed to prevent loose contacts or short cuts.

1.3 Storage Conditions and Handling Instructions

It is of great importance to understand that a humidity sensor is not a normal electronic component and needs to be handled with care. Chemical vapors at high concentration in combination with long exposure times may offset the sensor reading.

For these reasons it is recommended to store the sensors in original packaging including the sealed ESD bag at following conditions: Temperature shall be in the range of 10°C – 50°C (0 – 80°C for limited time) and humidity at 20 – 60%RH (sensors that are not stored in ESD bags). For sensors that have been removed from the original packaging we recommend to store them in ESD bags made of metal-in PE-HD[8].

In manufacturing and transport the sensors shall be prevented of high concentration of chemical solvents and long exposure times. Out-gassing of glues, adhesive tapes and stickers or out-gassing packaging material such as bubble foils, foams, etc. shall be avoided. Manufacturing area shall be well ventilated.

For more detailed information please consult the document *"Handling Instructions"* or contact Sensirion.

1.4 Reconditioning Procedure

As stated above extreme conditions or exposure to solvent vapors may offset the sensor. The following reconditioning procedure may bring the sensor back to calibration state:

Baking: 100 – 105°C at < 5%RH for 10h
Re-Hydration: 20 – 30°C at ~ 75%RH for 12h [9].

1.5 Temperature Effects

Relative humidity reading strongly depends on temperature. Therefore, it is essential to keep humidity sensors at the same temperature as the air of which the relative humidity is to be measured. In case of testing or qualification the reference sensor and test sensor must show equal temperature to allow for comparing humidity readings.

The packaging of SHT7x is designed for minimal heat transfer from the pins to the sensor. Still, if the SHT7x shares a PCB with electronic components that produce heat it should be mounted in a way that prevents heat transfer or keeps it as low as possible.

Furthermore, there are self-heating effects in case the measurement frequency is too high. Please refer to Section 3.3 for detailed information.

[7] 235°C corresponds to 455°F, 350°C corresponds to 662°F

[8] For example, 3M antistatic bag, product "1910" with zipper .

[9] 75%RH can conveniently be generated with saturated NaCl solution. 100 – 105°C correspond to 212 – 221°F, 20 – 30°C correspond to 68 – 86°F

1.6 Light

The SHT7x is not light sensitive. Prolonged direct exposure to sunshine or strong UV radiation may age the housing.

1.7 Materials Used for Sealing / Mounting

Many materials absorb humidity and will act as a buffer increasing response times and hysteresis. Materials in the vicinity of the sensor must therefore be carefully chosen. Recommended materials are: Any metals, LCP, POM (Delrin), PTFE (Teflon), PE, PEEK, PP, PB, PPS, PSU, PVDF, PVF.

For sealing and gluing (use sparingly): Use high filled epoxy for electronic packaging (e.g. glob top, underfill), and Silicone. Out-gassing of these materials may also contaminate the SHT7x (see Section 1.3). Therefore try to add the sensor as a last manufacturing step to the assembly, store the assembly well ventilated after manufacturing or bake at 50°C for 24h to outgas contaminants before packing.

1.8 Wiring Considerations and Signal Integrity

SHT7x are often applied using wires. Carrying the SCK and DATA signal parallel and in close proximity more than 10cm may result in cross talk and loss of communication. This may be resolved by routing VDD and/or GND between the two data signals and/or using shielded cables. Furthermore, slowing down SCK frequency will possibly improve signal integrity.

Please see the Application Note "ESD, Latch-up and EMC" for more information.

1.9 ESD (Electrostatic Discharge)

ESD immunity is qualified according to MIL STD 883E, method 3015 (Human Body Model at ±2 kV).

Latch-up immunity is provided at a force current of ±100mA with T_{amb} = 80°C according to JEDEC78A. See Application Note "ESD, Latch-up and EMC" for more information.

2 Interface Specifications

Pin	Name	Comment
1	SCK	Serial Clock, input only
2	VDD	Source Voltage
3	GND	Ground
4	DATA	Serial Data, bidirectional

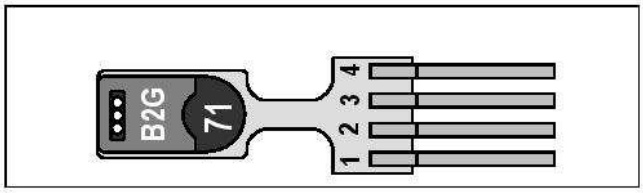


Table 1: SHT7x pin assignment.

2.1 Power Pins (VDD, GND)

The supply voltage of SHT7x must be in the range of 2.4 and 5.5V, recommended supply voltage is 3.3V. Decoupling of VDD and GND by a 100nF capacitor is integrated on the backside of the sensor packaging.

The serial interface of the SHT7x is optimized for sensor readout and effective power consumption. The sensor cannot be addressed by I²C protocol, however, the sensor can be connected to an I²C bus without interference with other devices connected to the bus. Microcontroller must switch between protocols.

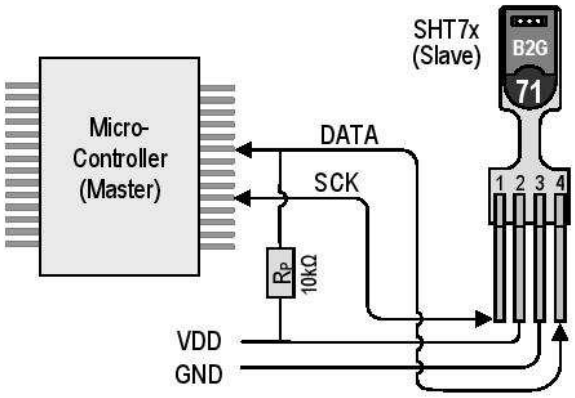


Figure 5: Typical application circuit, including pull up resistor R_P.

2.2 Serial clock input (SCK)

SCK is used to synchronize the communication between microcontroller and SHT7x. Since the interface consists of fully static logic there is no minimum SCK frequency.

2.3 Serial data (DATA)

The DATA tri-state pin is used to transfer data in and out of the sensor. For sending a command to the sensor, DATA is valid on the rising edge of the serial clock (SCK) and must remain stable while SCK is high. After the falling edge of SCK the DATA value may be changed. For safe communication DATA valid shall be extended T_{SU} and T_{HO} before the rising and after the falling edge of SCK, respectively – see Figure 6. For reading data from the sensor, DATA is valid T_V after SCK has gone low and remains valid until the next falling edge of SCK.

To avoid signal contention the microcontroller must only drive DATA low. An external pull-up resistor (e.g. 10 kΩ) is required to pull the signal high – it should be noted that pull-up resistors may be included in I/O circuits of

microcontrollers. See Table 2 for detailed I/O characteristic of the sensor.

2.4 Electrical Characteristics

The electrical characteristics such as power consumption, low and high level, input and output voltages depend on the supply voltage. Table 2 gives electrical characteristics of SHT7x with the assumption of 5V supply voltage if not stated otherwise. For proper communication with the sensor it is essential to make sure that signal design is strictly within the limits given in Table 3 and Figure 6. Absolute maximum ratings for VDD versus GND are +7V and -0.3V. Exposure to absolute maximum rating conditions for extended periods may affect the sensor reliability (e.g. hot carrier degradation, oxide breakdown).

Parameter	Conditions	min	typ	max	Units
Power supply DC[10]		2.4	3.3	5.5	V
Supply current	measuring		0.55	1	mA
	average[11]	2	28		µA
	sleep		0.3	1.5	µA
Low level output voltage	I_{OL} < 4 mA	0		250	mV
High level output voltage	R_P < 25 kΩ	90%		100%	VDD
Low level input voltage	Negative going	0%		20%	VDD
High level input voltage	Positive going	80%		100%	VDD
Input current on pads				1	µA
Output current	on			4	mA
	Tri-stated (off)		10	20	µA

Table 2: SHT7x DC characteristics. R_P stands for pull up resistor, while I_{OL} is low level output current.

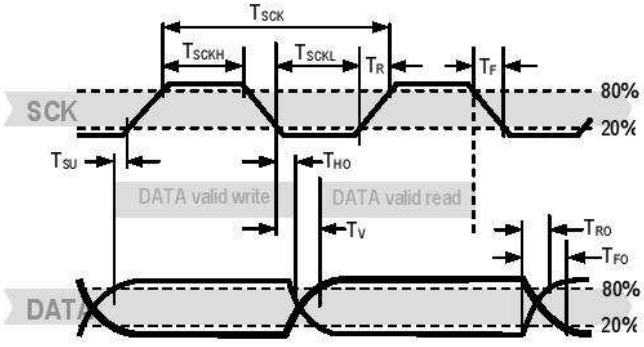


Figure 6: Timing Diagram, abbreviations are explained in Table 3. Bold DATA line is controlled by the sensor, plain DATA line is controlled by the micro-controller. Note that DATA valid read time is triggered by falling edge of anterior toggle.

[10] Recommended voltage supply for highest accuracy is 3.3V, due to sensor calibration.

[11] Minimum value with one measurement of 8 bit resolution without OTP reload per second, typical value with one measurement of 12bit resolution per second.

	Parameter	Conditions	min	typ	max	Units
F_{SCK}	SCK Frequency	VDD > 4.5V	0	0.1	5	MHz
		VDD < 4.5V	0	0.1	1	MHz
T_{SCKx}	SCK hi/low time		100			ns
T_R/T_F	SCK rise/fall time		1	200	*	ns
T_{FO}	DATA fall time	OL = 5pF	3.5	10	20	ns
		OL = 100pF	30	40	200	ns
T_{RO}	DATA rise time		**	**	**	ns
T_V	DATA valid time		200	250	***	ns
T_{SU}	DATA setup time		100	150	***	ns
T_{HO}	DATA hold time		10	15	****	ns

* $T_{R\_max} + T_{F\_max} = (F_{SCK})^{-1} - T_{SCKH} - T_{SCKL}$

** T_{R0} is determined by the R_P*C_{bus} time-constant at DATA line

*** T_{V_max} and T_{SU_max} depend on external pull-up resistor (R_P) and total bus line capacitance (Cbus) at DATA line

**** $T_{H0\_max} < T_V - \max(T_{R0}, T_{F0})$

Table 3: SHT7x I/O signal characteristics, OL stands for Output Load, entities are displayed in Figure 6.

3 Communication with Sensor

3.1 Start up Sensor

As a first step the sensor is powered up to chosen supply voltage VDD. The slew rate during power up shall not fall below 1V/ms. After power-up the sensor needs 11ms to get to Sleep State. No commands must be sent before that time.

3.2 Sending a Command

To initiate a transmission, a Transmission Start sequence has to be issued. It consists of a lowering of the DATA line while SCK is high, followed by a low pulse on SCK and raising DATA again while SCK is still high – see Figure 7.

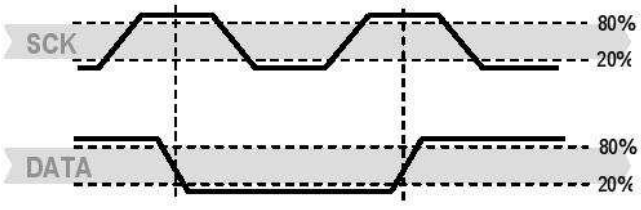


Figure 7: "Transmission Start" sequence

The subsequent command consists of three address bits (only '000' is supported) and five command bits. The SHT7x indicates the proper reception of a command by pulling the DATA pin low (ACK bit) after the falling edge of the 8th SCK clock. The DATA line is released (and goes high) after the falling edge of the 9th SCK clock.

Command	Code
Reserved	0000x
Measure Temperature	**00011**
Measure Relative Humidity	**00101**
Read Status Register	00111
Write Status Register	00110
Reserved	0101x-1110x
Soft reset, resets the interface, clears the status register to default values. Wait minimum 11 ms before next command	**11110**

Table 4: SHT7x list of commands

3.3 Measurement of RH and T

After issuing a measurement command ('00000101' for relative humidity, '00000011' for temperature) the controller has to wait for the measurement to complete. This takes a maximum of 20/80/320 ms for a 8/12/14bit measurement. The time varies with the speed of the internal oscillator and can be lower by up to 30%. To signal the completion of a measurement, the SHT7x pulls data line low and enters Idle Mode. The controller must wait for this Data Ready signal before restarting SCK to readout the data. Measurement data is stored until readout, therefore the controller can continue with other tasks and readout at its convenience.

Two bytes of measurement data and one byte of CRC checksum (optional) will then be transmitted. The micro controller must acknowledge each byte by pulling the DATA line low. All values are MSB first, right justified (e.g. the 5th SCK is MSB for a 12bit value, for a 8bit result the first byte is not used).

Communication terminates after the acknowledge bit of the CRC data. If CRC-8 checksum is not used the controller may terminate the communication after the measurement data LSB by keeping ACK high. The device automatically returns to Sleep Mode after measurement and communication are completed.

Important: To keep self heating below 0.1°C, SHT7x should not be active for more than 10% of the time – e.g. maximum one measurement per second at 12bit accuracy shall be made.

3.4 Connection reset sequence

If communication with the device is lost the following signal sequence will reset the serial interface: While leaving DATA high, toggle SCK nine or more times – see Figure 8. This must be followed by a Transmission Start sequence preceding the next command. This sequence resets the interface only. The status register preserves its content.

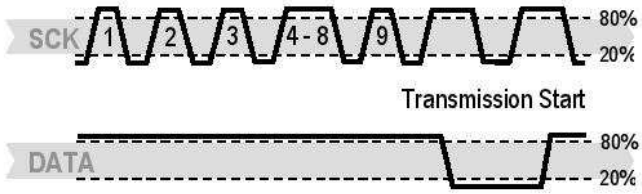


Figure 8: Connection Reset Sequence

3.5 CRC-8 Checksum calculation

The whole digital transmission is secured by an 8bit checksum. It ensures that any wrong data can be detected and eliminated. As described above this is an additional feature of which may be used or abandoned. Please consult Application Note "CRC Checksum" for information on how to calculate the CRC.

3.6 Status Register

Some of the advanced functions of the SHT7x such as selecting measurement resolution, end-of-battery notice, use of OTP reload or using the heater may be activated by sending a command to the status register. The following section gives a brief overview of these features.

After the command Status Register Read or Status Register Write – see Table 4 – the content of 8 bits of the status register may be read out or written. For the communication compare Figure 9 and Figure 10 – the assignation of the bits is displayed in Table 5.

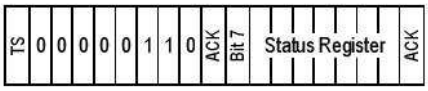


Figure 9: Status Register Write

Figure 10: Status Register Read

Examples of full communication cycle are displayed in Figure 11 and Figure 12.

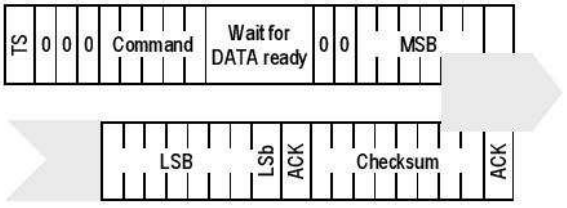


Figure 11: Overview of Measurement Sequence. TS = Transmission Start, MSB = Most Significant Byte, LSB = Last Significant Byte, LSb = Last Significant Bit.

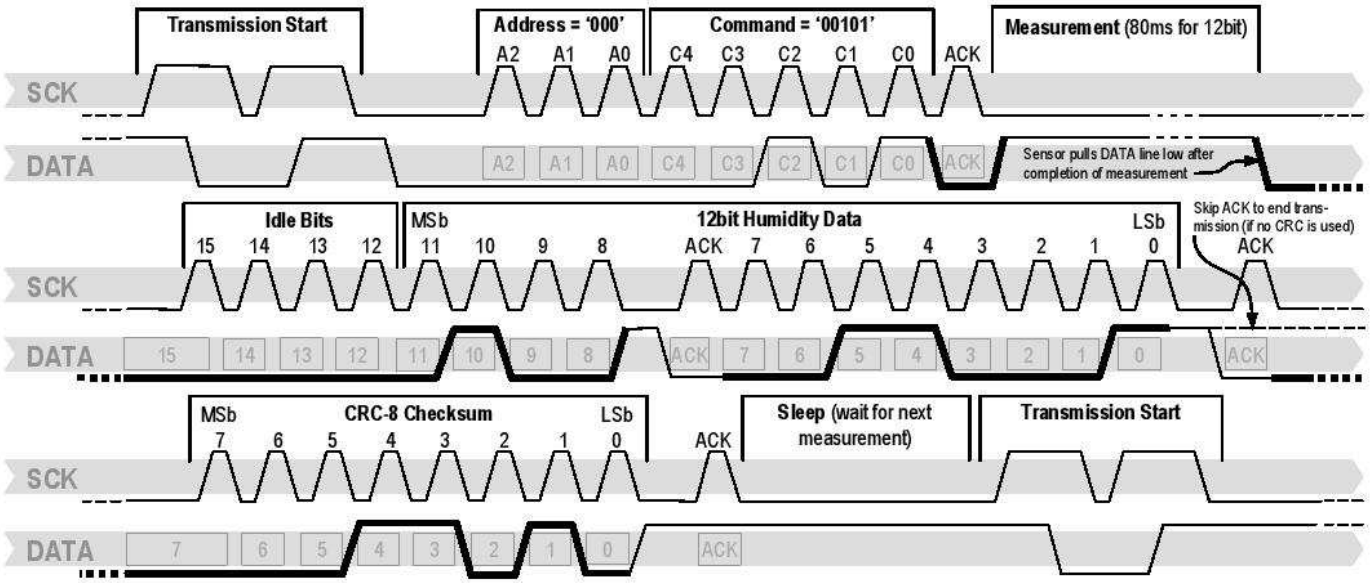


Figure 12: Example RH measurement sequence for value "0000'0100"0011'0001" = 1073 = 35.50%RH (without temperature compensation). DATA valid times are given and referenced in boxes on DATA line. Bold DATA lines are controlled by sensor while plain lines are controlled by the micro-controller.

Bit	Type	Description	Default	
7		reserved	0	
6	R	End of Battery (low voltage detection) '0' for VDD > 2.47 '1' for VDD < 2.47	X	No default value, bit is only updated after a measurement
5		reserved	0	
4		reserved	0	
3		For Testing only, do not use	0	
2	R/W	Heater	0	off
1	R/W	no reload from OTP	0	reload
0	R/W	'1' = 8bit RH / 12bit Temp. resolution '0' = 12bit RH / 14bit Temp. resolution	0	12bit RH 14bit Temp.

Table 5: Status Register Bits

Measurement resolution: The default measurement resolution of 14bit (temperature) and 12bit (humidity) can be reduced to 12 and 8bit. This is especially useful in high speed or extreme low power applications.

End of Battery function detects and notifies VDD voltages below 2.47 V. Accuracy is ±0.05 V.

Heater: An on chip heating element can be addressed by writing a command into status register. The heater may increase the temperature of the sensor by 5 – 10°C[12] beyond ambient temperature. The heater draws roughly 8mA @ 5V supply voltage.

For example the heater can be helpful for functionality analysis: Humidity and temperature readings before and after applying the heater are compared. Temperature shall increase while relative humidity decreases at the same time. Dew point shall remain the same.

Please note: The temperature reading will display the temperature of the heated sensor element and not ambient temperature. Furthermore, the sensor is not qualified for continuous application of the heater.

OTP reload: With this operation the calibration data is uploaded to the register before each measurement. This may be deactivated for reducing measurement time by about 10ms.

4 Conversion of Signal Output

4.1 Relative Humidity

For compensating non-linearity of the humidity sensor – see Figure 13 – and for obtaining the full accuracy of the sensor it is recommended to convert the humidity readout (SO_{RH}) with the following formula with coefficients given in Table 6:

$$RH_{linear} = c_1 + c_2 \cdot SO_{RH} + c_3 \cdot SO_{RH}^2 \ (\%RH)$$

SO_{RH}	c_1	c_2	c_3
12 bit	-2.0468	0.0367	-1.5955E-6
8 bit	-2.0468	0.5872	-4.0845E-4

Table 6: Humidity conversion coefficients

[12] Corresponds to 9 – 18°F

Values higher than 99%RH indicate fully saturated air and must be processed and displayed as 100%RH[13]. Please note that the humidity sensor has no significant voltage dependency.

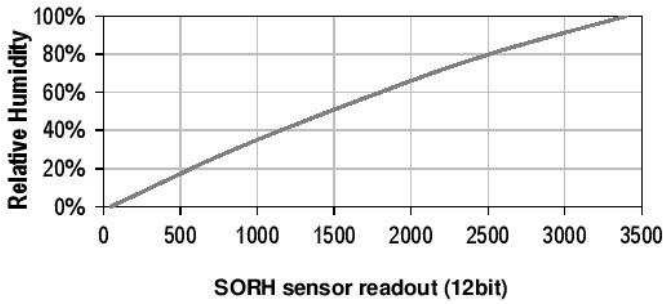


Figure 13: Conversion from SO_{RH} to relative humidity

4.2 Temperature compensation of Humidity Signal

For temperatures significantly different from 25°C (~77°F) the humidity signal requires temperature compensation. The temperature correction corresponds roughly to 0.12%RH/°C @ 50%RH. Coefficients for the temperature compensation are given in Table 8.

$$RH_{true} = (T_{°C} - 25) \cdot (t_1 + t_2 \cdot SO_{RH}) + RH_{linear}$$

SO_{RH}	t_1	t_2
12 bit	0.01	0.00008
8 bit	0.01	0.00128

Table 7: Temperature compensation coefficients

4.3 Temperature

The band-gap PTAT (Proportional To Absolute Temperature) temperature sensor is very linear by design. Use the following formula to convert digital readout (SO_T) to temperature value, with coefficients given in Table 9:

$$T = d_1 + d_2 \cdot SO_T$$

VDD	d_1 (°C)	d_1 (°F)
5V	-40.1	-40.2
4V	-39.8	-39.6
3.5V	-39.7	-39.5
3V	-39.6	-39.3
2.5V	-39.4	-38.9

SO_T	d_2 (°C)	d_2 (°F)
14bit	0.01	0.018
12bit	0.04	0.072

Table 8: Temperature conversion coefficients.

4.4 Dew Point

SHT7x is not measuring dew point directly, however dew point can be derived from humidity and temperature readings. Since humidity and temperature are both measured on the same monolithic chip, the SHT7x allows superb dew point measurements.

For dew point (T_d) calculations there are various formulas to be applied, most of them quite complicated. For the temperature range of -40 – 50°C the following approximation provides good accuracy with parameters given in Table 10:

$$T_d(RH,T) = T_n \cdot \frac{\ln\left(\frac{RH}{100\%}\right) + \frac{m \cdot T}{T_n + T}}{m - \ln\left(\frac{RH}{100\%}\right) - \frac{m \cdot T}{T_n + T}}$$

Temperature Range	Tn (°C)	m
Above water, 0 – 50°C	243.12	17.62
Above ice, -40 – 0°C	272.62	22.46

Table 9: Parameters for dew point (T_d) calculation.

Please note that "ln(...)" denotes the natural logarithm. For RH and T the linearized and compensated values for relative humidity and temperature shall be applied.

For more information on dew point calculation see Application Note "Introduction to Humidity".

5 Environmental Stability

If sensors are qualified for assemblies or devices, please make sure that they experience same conditions as the reference sensor. It should be taken into account that response times in assemblies may be longer, hence enough dwell time for the measurement shall be granted. For detailed information please consult Application Note "Testing Guide".

SHT7x have been tested according to the test conditions given in Table 11. Sensor performance under other test conditions cannot be guaranteed and is not part of the sensor specifications. Especially, no guarantee can be given for sensor performance in the field or for customer's specific application.

Please contact Sensirion for detailed information.

[13] If wetted excessively (strong condensation of water on sensor surface), sensor output signal can drop below 100%RH (even below 0%RH in some cases), but the sensor will recover completely when water droplets evaporate. The sensor is not damaged by water immersion or condensation.

Environment	Standard	Results [14]
HTOL	125°C, 1000 h	Within specifications
TC	-40°C - 125°C, 500 cycles Acc. JESD22-A104-C	Within specifications
THU	85°C / 85%RH, 1000h	Within specifications
ESD immunity	MIL STD 883E, method 3015 (Human Body Model at ±2kV)	Qualified
Latch-up	force current of ±100mA with T_{amb} = 80°C, acc. JEDEC 17	Qualified

Table 10: Qualification tests: HTSL = High Temperature Storage Lifetime, TC = Temperature Cycles, UHST = Unbiased Highly accelerated temperature and humidity Test, THU = Temperature humidity unbiased

6 Packaging

6.1 Packaging type

The device is supplied in a single-in-line pin type package. The sensor housing consists of a Liquid Crystal Polymer (LCP) cap with epoxy glob top on a standard 0.6 mm FR4 substrate. The sensor head is connected to the pins, by a small bridge to minimize heat conduction and response times. The pins are made of Cu/Be alloy coated with 1.3µm Ni and 0.5µm Au, which are soldered to the FR4 substrate by lead-free solder paste. The gold plated back side of the sensor head is connected to the GND pin. A 100nF capacitor is mounted on the back side between VDD and GND. The device is fully RoHS and WEEE compliant – thus it is free of of Pb, Cd, Hg, Cr(6+), PBB and PBDE.

Size including pins is 19.5 x 5.08 x 3.1mm. Total weight: 168 mg, weight of sensor head: 73 mg.

All pins are Au plated to avoid corrosion. They can be soldered or mate with most 1.27 mm (0.05") sockets, for example: Preci-dip / Mill-Max R851-83-004-20-001 or similar.

6.2 Traceability Information

All SHT7x are marked with an alphanumeric, three digit code on the chip cap – see "B2G" on Figure 1. The lot numbers allow full traceability through production, calibration and testing. No information can be derived from the code directly, respective data is stored at Sensirion.

Labels on the reels are displayed in Figure 14 and Figure 15, they both give traceability information.

Figure 14: First label on reel: XX = Sensor Type (71 for SHT71), NN = Chip Version (04 for V4), Y = last digit of year, RRR = number of sensors on reel divided by 10 (200 for 2000 units), TTTTT = Traceability Code.

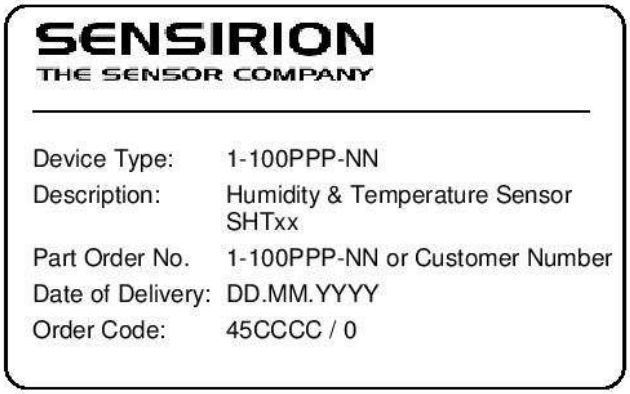


Figure 15: Second label on reel: For Device Type and Part Order Number please refer to Table 12, Delivery Date (also Date Code) is date of packaging of sensors (DD = day, MM = month, YYYY = year), CCCC = number of order.

6.3 Shipping Package

SHT7x are shipped in 32mm tape at 50pcs each – for details see Figure 16 and Table 12. Reels are individually labeled with barcode and human readable labels, see section 6.2.

Sensor Type	Packaging	Quantity	Order Number
SHT71	Tape Stripes	50	1-100092-04
SHT75	Tape Stripes	50	1-100071-04

Table 11: Packaging types per sensor type.

Dimensions of packaging tape are given in Figure 16. All tapes have a 7 pockets empty leader tape (first pockets of the tape) and a 7 pockets empty trailer tape (last pockets of the tape).

[14] According to accuracy and long term drift specification given on Page 2.

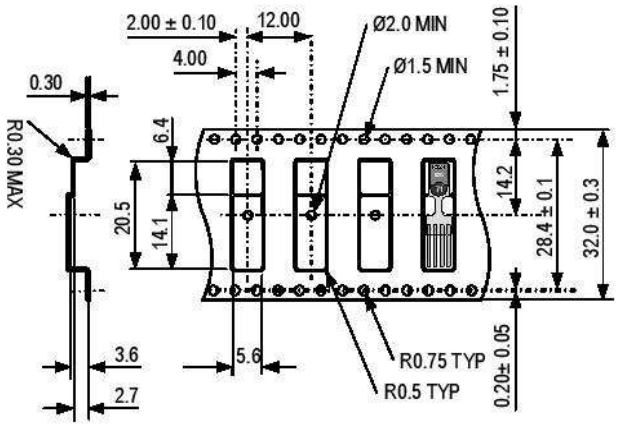


Figure 16: Tape configuration and unit orientation within tape, dimensions in mm (1mm = 0.039inch). Leader tape is to the right of the figure, trailer tape to the left.

Revision History

Date	Version	Page(s)	Changes
March 2007	3.0	1 – 10	Data sheet valid for SHTxx-V4 and SHTxx-V3
July 2008	4.0	1 – 10	New release, rework of datasheet
April 2009	4.2	2, 7	Amended foot note 2, communication diagram changed (Figure 12)
May 2010	4.3	1 – 11	Errors eliminated, information added – for details please ask for change protocol.
December 2011	5	1, 7-9	References to V3 sensors eliminated.

Important Notices

Warning, Personal Injury

Do not use this product as safety or emergency stop devices or in any other application where failure of the product could result in personal injury. Do not use this product for applications other than its intended and authorized use. Before installing, handling, using or servicing this product, please consult the data sheet and application notes. Failure to comply with these instructions could result in death or serious injury.

If the Buyer shall purchase or use SENSIRION products for any unintended or unauthorized application, Buyer shall defend, indemnify and hold harmless SENSIRION and its officers, employees, subsidiaries, affiliates and distributors against all claims, costs, damages and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if SENSIRION shall be allegedly negligent with respect to the design or the manufacture of the product.

ESD Precautions

The inherent design of this component causes it to be sensitive to electrostatic discharge (ESD). To prevent ESD-induced damage and/or degradation, take customary and statutory ESD precautions when handling this product.
See application note "ESD, Latchup and EMC" for more information.

Warranty

SENSIRION warrants solely to the original purchaser of this product for a period of 12 months (one year) from the date of delivery that this product shall be of the quality, material and workmanship defined in SENSIRION's published specifications of the product. Within such period, if proven to be defective, SENSIRION shall repair and/or replace this product, in SENSIRION's discretion, free of charge to the Buyer, provided that:

- notice in writing describing the defects shall be given to SENSIRION within fourteen (14) days after their appearance;
- such defects shall be found, to SENSIRION's reasonable satisfaction, to have arisen from SENSIRION's faulty design, material, or workmanship;
- the defective product shall be returned to SENSIRION's factory at the Buyer's expense; and
- the warranty period for any repaired or replaced product shall be limited to the unexpired portion of the original period.

This warranty does not apply to any equipment which has not been installed and used within the specifications recommended by SENSIRION for the intended and proper use of the equipment. EXCEPT FOR THE WARRANTIES EXPRESSLY SET FORTH HEREIN, SENSIRION MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THE PRODUCT. ANY AND ALL WARRANTIES, INCLUDING WITHOUT LIMITATION, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ARE EXPRESSLY EXCLUDED AND DECLINED.
SENSIRION is only liable for defects of this product arising under the conditions of operation provided for in the data sheet and proper use of the goods. SENSIRION explicitly disclaims all warranties, express or implied, for any period during which the goods are operated or stored not in accordance with the technical specifications.
SENSIRION does not assume any liability arising out of any application or use of any product or circuit and specifically disclaims any and all liability, including without limitation consequential or incidental damages. All operating parameters, including without limitation recommended parameters, must be validated for each customer's applications by customer's technical experts. Recommended parameters can and do vary in different applications.
SENSIRION reserves the right, without further notice, (i) to change the product specifications and/or the information in this document and (ii) to improve reliability, functions and design of this product.

Headquarters and Subsidiaries

SENSIRION AG
Laubisruetistr. 50
CH-8712 Staefa ZH
Switzerland

phone: +41 44 306 40 00
fax: +41 44 306 40 30
info@sensirion.com
www.sensirion.com

Sensirion AG (Germany)
phone: +41 44 927 11 66
info@sensirion.com
www.sensirion.com

Sensirion Inc., USA
phone: +1 805 409 4900
info_us@sensirion.com
www.sensirion.com

Sensirion Japan Co. Ltd.
phone: +81 3 3444 4940
info@sensirion.co.jp
www.sensirion.co.jp

Sensirion Korea Co. Ltd.
phone: +82 31 345 0031 3
info@sensirion.co.kr
www.sensirion.co.kr

Sensirion China Co. Ltd.
phone: +86 755 8252 1501
info@sensirion.com.cn
www.sensirion.com.cn

To find your local representative, please visit www.sensirion.com/contact

ANEXO II

Hoja de especificaciones SP135FZ

SP 135 FZ / 140 FZ

Smallest Vane Pump World Wide!

SP 135 FZ
SP 140 FZ

Product Features

- micro vane pump for air and gas
- applicable as vacuum pump or pressure pump
- electric motor driven
- available in various voltages
- minimal current consumption
- linear characteristic
- oil free / maintenance free
- reduced vibrations
- low noise
- extremely small pocket size
- lightweight (13,5 g)

Applications

- Gas sampler / gas measurement / gas metering
- Portable gas detector pump for safety devices
- Patient monitoring / patient analysers
- Personal Gas Detector / Gas Monitor
- Exhaust gas analyser
- Pneumatics
- Pipetting devices
- Fuel cells

Schwarzer Precision

Am Lichtbogen 7 • 45141 Essen • Germany
Phone: +49 201 31697-0 • Fax: +49 201 31697-29
e-mail: info@schwarzer.com • http://www.schwarzer.com

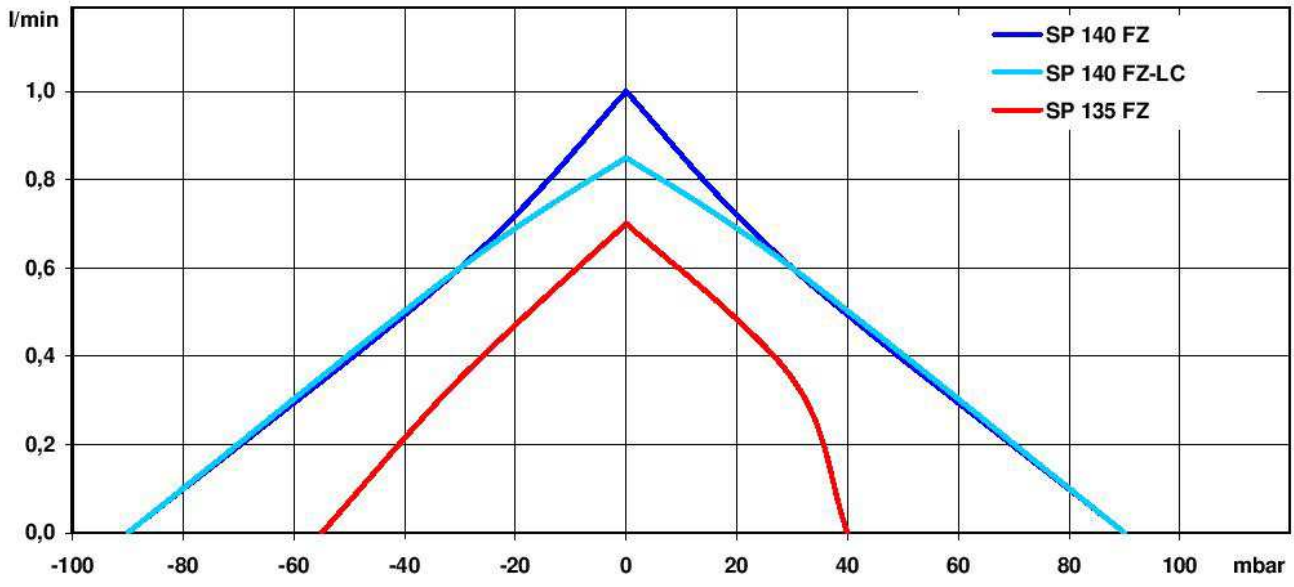
SP 135 FZ / 140 FZ

These pumps are our standard models. For special requirements we are offering customised versions. Please contact us!

Designation	SP 135 FZ		SP 140 FZ-LC		SP 140 FZ
Operating Voltage Art.-No.	3 Vdc 7s60044	4,5 Vdc 7s60045	3 Vdc 7s60040	4,5 Vdc 7s60041	4,5 - 12 Vdc on request
Pneumatic Performance *					
Free Flow [l/min]	0,7	0,7	0,85	0,85	1,0
flow at:					
30 mbar [l/min]	0,35	0,35	0,6	0,6	0,60
max. Pressure [mbar]	40	55	90	90	90
flow at:					
-30 mbar [l/min]	0,35	0,35	0,6	0,6	0,60
max. Vacuum [mbar]	-55	-55	-90	-90	-90
Motor / Power Consumption					
Motor Type	Skew Wound DC		Iron Core		Skew Wound DC
max. Current [mA]	130 mA	90 mA	250 mA	130 mA	
Construction					
Body and moving parts	Resin bound Graphite and Tecapeek				
Weight [g]	12,3		12,3		13,5

*** Note:** 10 µm filtration required at pump inlet for optimal performance!

Performance Curve

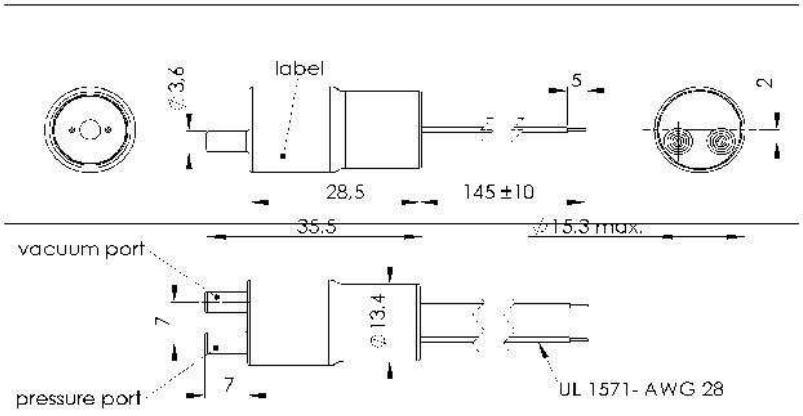


Schwarzer Precision

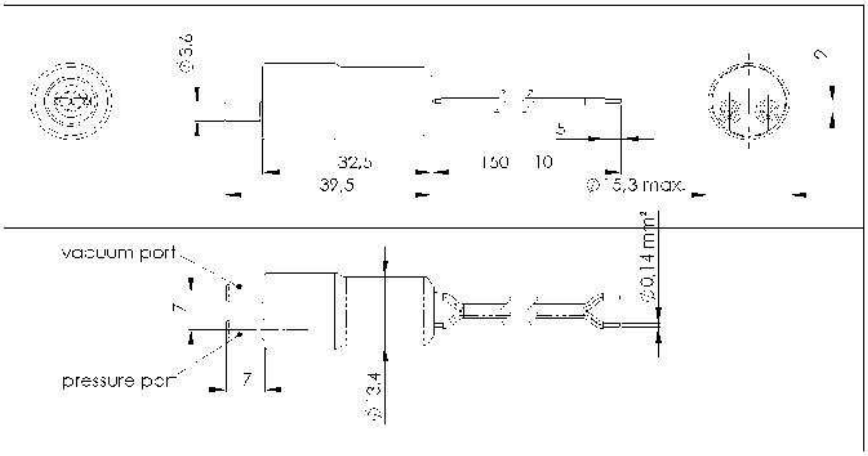
Am Lichtbogen 7 • 45141 Essen • Germany
Phone: +49 201 31697-0 • Fax: +49 201 31697-29
e-mail: info@schwarzer.com • http://www.schwarzer.com

SP 135 FZ / 140 FZ

SP 135 FZ / 140 FZ



SP 140 FZ-LC



Schwarzer Precision

Am Lichtbogen 7 • 45141 Essen • Germany
Phone: +49 201 31697-0 • Fax: +49 201 31697-29
e-mail: info@schwarzer.com • http://www.schwarzer.com

ANEXO III

Código completo implementado con Energia

```
//SECCIÓN DE DECLARACIÓN DE VARIABLES GLOBALES Y FUNCIONES

//Variables globales

 const int clock=PB_1;    //Se asigna el reloj al pin digital PB1
 const int data=PD_1;     //Se asigna los datos al pin digital PD1
 const int analogIn=A10;  //Se asigna la salida analógica al pin A10(PB_4)


//Funciones

 //RESET: SOLO INTERFAZ, EL REGISTRO SE MANTIENE
 void Reset()
 {
  pinMode(data,OUTPUT);    //data como salida para poder escribir en el pin
  digitalWrite(data,HIGH);
  digitalWrite(clock,LOW);
  for (int i=1;i<=10;i++)  //mantener el data en HIGH durante 9 pulsos de reloj o mas
  {
   digitalWrite(clock,HIGH);
   delayMicroseconds(10);
   digitalWrite(clock,LOW);
   delayMicroseconds(10);
  }
 }


 //SECUENCIA DE COMIENZO DE LA TRANMISIÓN
 void Start_Transmision()
 {
  pinMode(data, OUTPUT); //data como salida para poder escribir en el pin
  delay(20);                    //esperar a alcanzar el 'sleep state' (al menos 11 ms)
  digitalWrite(data,HIGH);    //data en LOW mientras reloj HIGH
  digitalWrite(clock,HIGH);   // y volviendo a poner data en HIGH mientras reloj HIGH
  delayMicroseconds(10);
  digitalWrite(data,LOW);
  delayMicroseconds(10);
  digitalWrite(clock,LOW);
```

```
  delayMicroseconds(10);
  digitalWrite(clock,HIGH);
  delayMicroseconds(10);
  digitalWrite(data,HIGH);
  delayMicroseconds(10);
  digitalWrite(clock,LOW);
}

//SECUENCIA QUE INDICA QUE LA TRANMISIÓN HA SIDO CORRECTA
void ACK()
{
  pinMode(data,OUTPUT);  //data como salida para poder escribir en el pin
  digitalWrite(clock,HIGH);   //mantener data en LOW durante un pulso de reloj
  digitalWrite(data,LOW);
  delayMicroseconds(10);
  digitalWrite(clock,LOW);
  delayMicroseconds(10);
  pinMode(data,INPUT);      //data como entrada para poder leer el pin
}

//FUNCIÓN QUE REALIZA UNA MEDIDA DE TEMPERATURA
int Medida()
{
  Reset();
  Start_Transmision();
  pinMode(data,OUTPUT); //como salida para poder escribir el comando que se manda
  digitalWrite(clock,LOW);  //en cada pulso de reloj se manda un bit del comando

  //Enviar comando
  int comandoT[]={0, 0, 0, 0, 0, 0, 1, 1};  //es el comando para medir la temperatura
  for (int i=0;i<8;i++)  //se recorren todos los bits del comando
  {
    if (comandoT[i]==1)  //si el bit es un 1, escribir HIGH en el pin
    {
      digitalWrite(data,HIGH);
    }
```

```
  else              //sino, escribir LOW
  {
    digitalWrite(data,LOW);
  }
  delayMicroseconds(10);
  digitalWrite(clock,HIGH);
  delayMicroseconds(10);
  digitalWrite(clock,LOW);
}

ACK();
delay(320);

//Se leen la información que transmite el sensor
pinMode(data,INPUT);  //como entrada para leer la medida que ha hecho el sensor
int value[16];        //variable de 16 bits donde se guarda lo medida

for (int k=0;k<16;k++)
{
  digitalWrite(clock,HIGH);  //en cada pulso de reloj leer lo que hay en data
  int d=digitalRead(data);
  if (d==HIGH)              //si data es HIGH escribir 1 en value
  {
    value[k] = 1;
  }
  else                    //sino, escribir un cero
  {
    value[k]=0;
  }
  digitalWrite(clock,LOW);
  if (k==8)                //después de 8 pulsos mandar un ACK
  {
    ACK();
  }
}
```

```
  //se convierten los 16 bits del array en un int
  int valor = 0;
  for (int i=0;i<16;i++)
  {
    if (value[i] == 1)
    {
      valor = valor + (1 << (15-i));
    }
  }
 return valor;  //es el valor de retorno de la función
}

//FUNCIÓN QUE REALIZA UNA MEDIDA DE HUMEDAD RELATIVA
int Medida2()
{
  Reset();
  Start_Transmision();
  pinMode(data,OUTPUT);   //como salida para poder escribir el comando que se manda
  digitalWrite(clock,LOW);  //en cada pulso de reloj mandar un bit del comando

  //Enviar comando
  int comandoT[]={0, 0, 0, 0, 0, 1, 0, 1};  //es el comando para medir la humedad relativa
  for (int i=0;i<8;i++)  //recorre todos los bits del comando
  {
   if (comandoT[i]==1)  //si el bit es un 1, escribir HIGH en el pin
    {
      digitalWrite(data,HIGH);
    }
    else              //sino, escribir LOW
    {
      digitalWrite(data,LOW);
    }
    delayMicroseconds(10);
    digitalWrite(clock,HIGH);
    delayMicroseconds(10);
    digitalWrite(clock,LOW);
  }
```

```
  ACK();
  delay(320); //Se espera a que el sensor realice la medida

  //Se leen la información que transmite el sensor
  pinMode(data,INPUT);      //como entrada para leer la medida que ha hecho el sensor
  int value2[16];           //variable de 16bits donde se guarda lo medida

  for (int k=0;k<16;k++)
  {
    digitalWrite(clock,HIGH);  //en cada pulso de reloj leer lo que hay en data
    int d=digitalRead(data);
    if (d==HIGH)              //si data es HIGH escribir 1 en value
    {
      value2[k] = 1;
    }
    else                      //sino, escribir un cero
    {
      value2[k]=0;
    }
    digitalWrite(clock,LOW);
    if (k==8)                 //despues de 8 pulsos mandar un ACK
    {
      ACK();
    }

  //se convierten los 16 bits del array en un int
  int valor2 = 0;
  for (int i=0;i<16;i++)
  {
    if (value2[i] == 1)
    {
      valor2 = valor2 + (1 << (15-i));
    }
  }
 return valor2;  //es el valor de retorno de la función
}
```

```
//SECCIÓN "VOID SETUP()"

 void setup()
 {
  Serial.begin(9600);          //Inicialización y configuración del puerto serie
  pinMode(clock, OUTPUT);//clock siempre como salida, en ese pin solo se escribe
 }

//SECCIÓN "VOID LOOP()"

 void loop()
 {
  if (Serial.available()>0)     //Se comprueba si hay algún byte disponible para ser leído
  {
    char ind=Serial.read();    //Se lee la información disponible en el puerto serie

    if (ind=='T')
    {
      //La información recibida corresponde a la temperatura
      int tem=Medida();
      Serial.print(tem);
    }
    else if (ind=='H')
    {
      //La información recibida corresponde a la humedad relativa
      int hum=Medida2();
      Serial.print(hum);
    }
    else
    {
      //La información recibida corresponde a la caida de tensión en el divisor resistivo
      int voltaje= analogRead(analogIn);
      Serial.print(voltaje);
    }
  }
 }
```