

Sensirion SPS30

Sensor de PM2.5

Contenidos

- Descripción y casos de uso
- Principio de funcionamiento
- Emplazamiento
- Conectividad y Conexionado
- Software
- Fuentes

Descripción y casos de uso

El **SPS30** es un sensor óptico que se usa para medir la presencia de **PM2.5** y **PM10** para el control de calidad del aire y acondicionamiento del aire.

Las principales ventajas son su **reducido tamaño** y **extensa vida útil**.



**PM2.5 Sensor for
HVAC and air quality
applications SPS30**

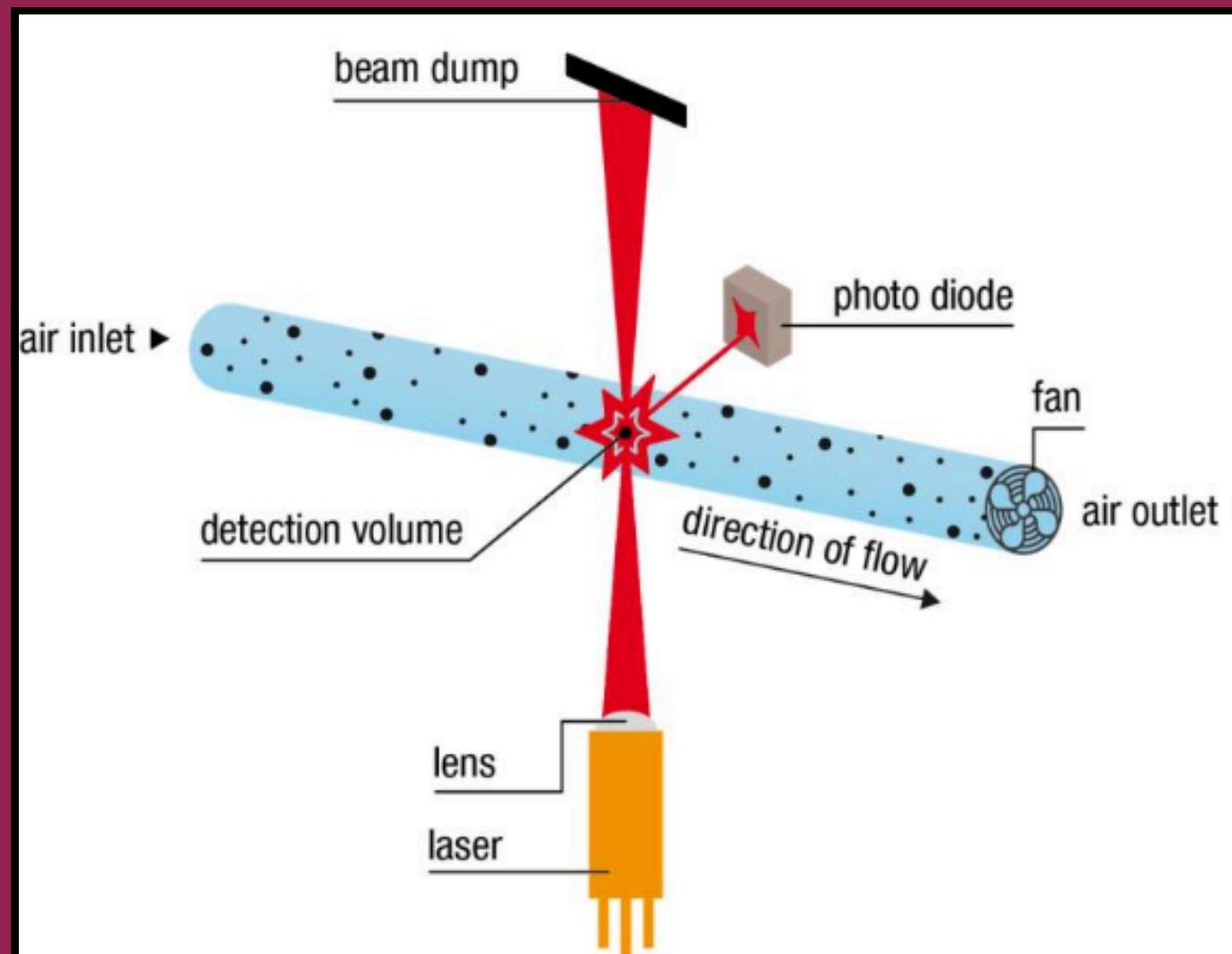


Figura 1: Medición de **volumen de partículas** en un **flujo de aire** mediante **láser**

Principio de funcionamiento

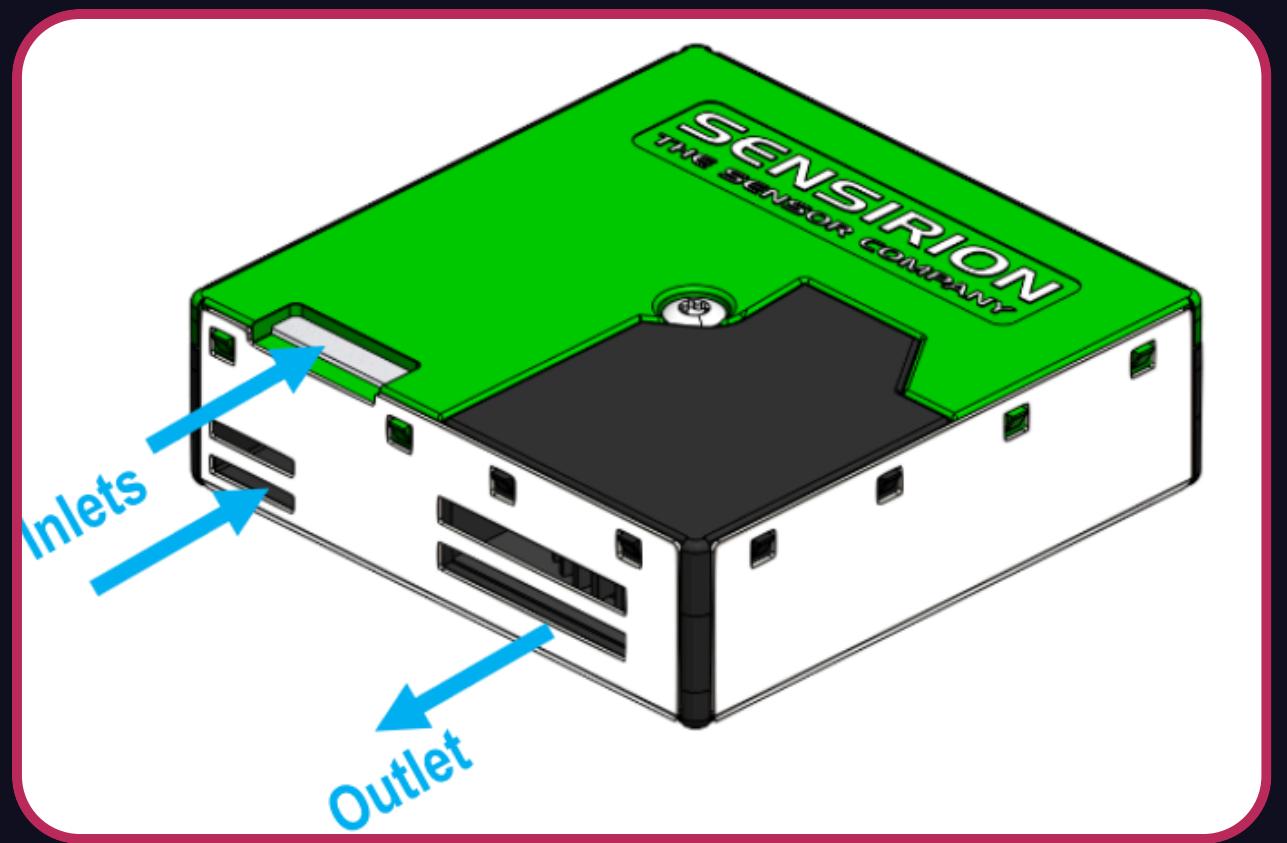
Es un **sensor óptico**. Funciona midiendo la cantidad de **partículas de materia** de distintos diámetros en un flujo de aire mediante **scattering** de un haz láser.

La medida más fiable es la de PM2.5. Las partículas de PM10 son demasiado pocas para medirlas con este tipo de sensores.

Si se decide ignorar este hecho: las medidas de un diámetro determinado *incluyen las de menor diámetro*, por lo tanto para que la medida sea correcta hay que **corregir por software**.

$$\text{PM10}_{\text{real}} = \text{PM10}_{\text{medido}} - \text{PM2.5}_{\text{medido}}$$

$$\text{PM2.5}_{\text{real}} = \text{PM2.5}_{\text{medido}}$$



Emplazamiento

Es necesario que el sensor tenga acceso a una **entrada** y una **salida de aire** para conseguir el correcto **flujo de aire**.

El fabricante recomienda que estas cavidades sean **independientes** entre sí para evitar que el flujo saliente contamine al entrante.

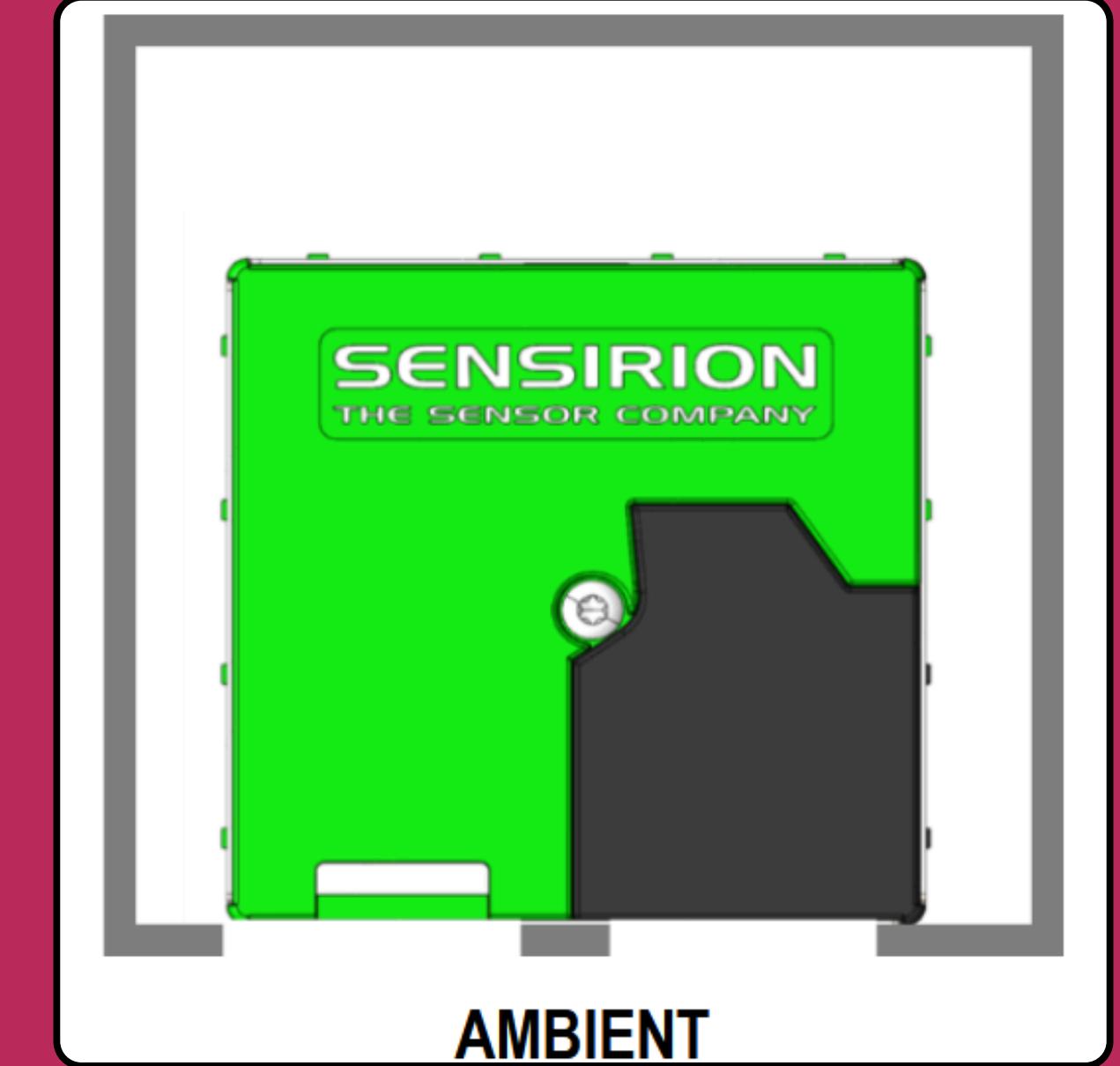


Figura 2: Entrada y salida de aire independientes.

El **SPS30** acepta **2** tipos de conexiones:

I2C: para distancias cortas (<20 cm) y entornos sin interferencias. Ha de conectarse el pin **SEL** a masa (**GND**).

Es el modo soportado en el **driver de Arduino**.

UART: para distancias más largas, más inmune a interferencias.

Se selecciona con el pin **SEL** volátil (**Sin conectar**).

Es el usado por el driver **embedded-sps** **UART** y los **script de python** (USB-TTL)

Más información en la hoja de características (*datasheet*)

Conectividad y Conexionado

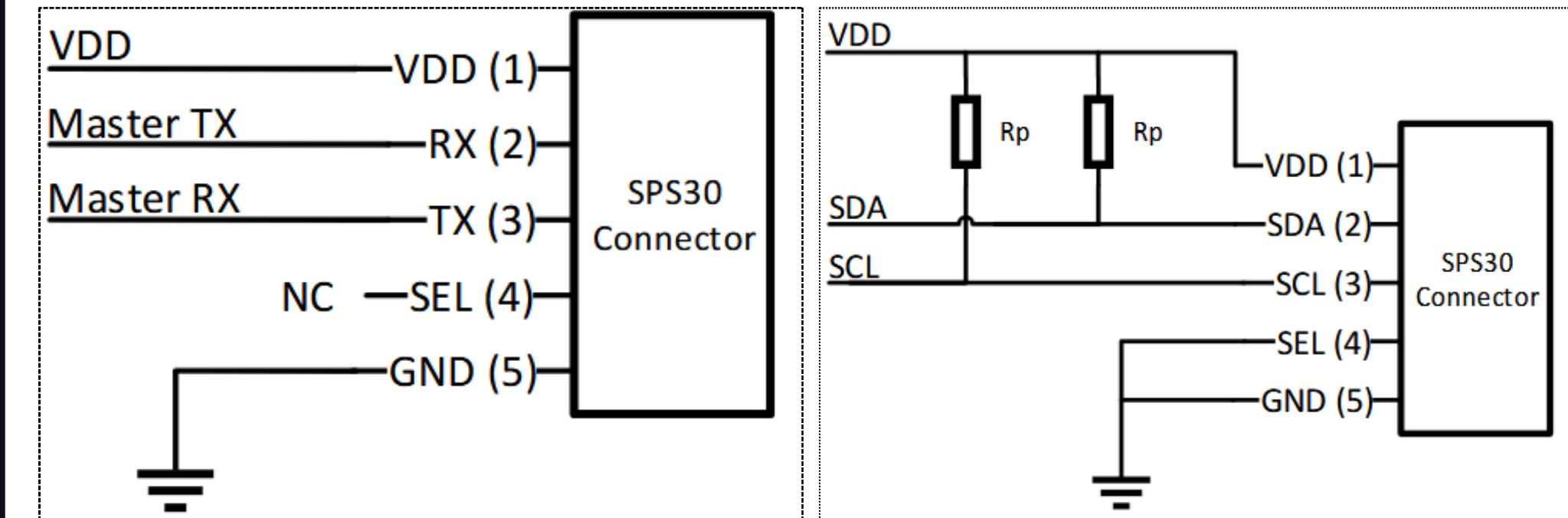


Figura 3: Conexionado **UART** (izq) e **I2C** (derecha)

Existen varios **drivers** ya implementados de forma oficial por el fabricante (**Sensirion**).
En orden de “complejidad”:

- **Arduino-sps**: Librería de **I2C** para comunicarse con el sensor **SPS30**.
enlace: <https://github.com/Sensirion/arduino-sps>
- **embedded-sps**: Librería de **I2C** de más bajo nivel en la que está basada la implementación de *arduino-sps*.
enlace: <https://github.com/Sensirion/embedded-sps>
- **embedded-uart-sps**: Similar a la anterior (de bajo nivel) pero para comunicación **UART**.
enlace: <https://github.com/Sensirion/embedded-uart-sps>
- **sps30**: este driver **no es oficial del fabricante**, pero tiene la posibilidad de adaptarse tanto a **UART** como a **I2C**.
enlace: <https://github.com/paulvha/sps30>
- **Sensirion_SPS30**: script de python **no oficial** para conectarse al sensor **SPS30** mediante un **adaptador USB a TTL** (CP2102). **Más instrucciones en el repositorio de MediaLab de diagnóstico de sensores**.
enlace: https://github.com/binh-bk/Sensirion_SPS30

Fuentes

- Catálogo: <https://sensirion.com/products/catalog/SPS30>
- Datasheet: [Enlace datasheet](#)
- Sensor specification statement: [Enlace spec. statement](#)
- Mechanical design and assembly: [Enlace mech. design & assembly](#)
- Repositorio Arduino: <https://github.com/Sensirion/arduino-sps>