## **Integrantes:** Fego Ramiro.

## **Laboratorio:** I.M.Y.R.S.C 7°1.

**INTRODUCCIÓN**

Descripción del Proyecto

**Eco Caudal** tiene como objetivo principal generar conciencia sobre el consumo de agua en diversos entornos, desde hogares hasta empresas. La implementación de este sistema permite monitorear de manera eficiente y efectiva la cantidad de agua utilizada, promoviendo tanto un uso consciente en el ámbito doméstico como una gestión eficiente en entornos comerciales.

En el ámbito residencial, el proyecto busca no solo fomentar hábitos más sostenibles, sino también brindar a los usuarios la capacidad de verificar que el pago por los servicios de agua sea justo y evitar posibles irregularidades. Además, la aplicación de esta tecnología puede extenderse a contextos más técnicos, como en empresas que requieren un control preciso del caudal para optimizar sus procesos y recursos hídricos.

En resumen, esta iniciativa fusiona la innovación tecnológica con la conciencia ambiental, proponiendo una solución práctica y accesible para abordar el uso responsable del agua en diversas áreas de la sociedad.

**METODOLOGÍA DE TRABAJO**

¿Qué es una metodología y por qué la utilizamos?

Una metodología de trabajo es todo conjunto de principios, prácticas, procesos y herramientas que facilitan la forma en que se lleva a cabo una tarea o proyecto en general. Todo esto nos da la posibilidad de tener un trabajo organizado, más eficiente y con mayor calidad.

Metodología Utilizada

La metodología utilizada para este proyecto fue la conocida como **SCRUM**. Fué elegida por su complejo de tener tareas organizadas, para desarrollo ágil trabajando colaborativamente y en equipo obteniendo así el mejor resultado en proyectos.

# **FASES DEL PROYECTO**

Fase 1

En esta fase definimos los roles del equipo para ejecutar el proyecto. Es primordial reconocer las cualidades de cada integrante, como también implementar salir de nuestra zona de confort para poder así obtener un desarrollo mejor y más grande, junto con el crecimiento del propio equipo.

Fase 2

En esta fase se planean las distintas tareas y su realización. Además de delimitar hasta dónde llegará el equipo, para evitar el desperdicio de recursos. Se hace una estimación de tiempo y esfuerzo necesarios para finalizar y asignarle un valor.

Fase 3

En esta fase se desarrolla el entregable de las tareas hechas anteriormente, haciendo una revisión en el equipo de desarrollo, seguido de cuestiones como que se hará hoy, que hice ayer y como puedo cumplir con mis tareas de surgir inconvenientes.

Fase 4

En esta fase se analiza y se comprueba el entregable para medir y mejorar la eficiencia del equipo. Formulando soluciones y oportunidades para mejorar los procesos de las próximas tareas a realizar.

Fase 5

Por último, en esta fase se hace la entrega final. Presentando entregables al cliente y retrospectiva del proyecto para dar a conocer que sucedió y cómo se puede mejorar.

# **ETAPAS DEL PROYECTO**

**5/12:** Comenzamos por definir los roles de cada integrante, visualizando un perfecto desarrollo de el proyecto. Para luego crear las distintas tareas que se realizarían durante el transcurso de las semanas, siendo estas cortas y organizadas para mayor eficiencia a la hora de trabajar.

**7/12:** Se puso en marcha la realización de las tareas establecidas, empezando por bocetos del logo, creando algo acorde al proyecto y sobre todo llamativo. Describir el proyecto enfatizando el objetivo principal, entre otras tareas.

**12/12:** Seguimos con tareas, pasando a la parte práctica de las mismas. Pasamos el boceto de el logo para crearlo de manera digital. Investigamos acerca de los componentes para poder empezar a desarrollar el código de su funcionamiento.

**14/12:** Continuamos con el código del cual faltaban partes, precisamente hacer los cálculos de medición, realizando el paso a paso de la conversión a pulsos. Para ello investigamos un poco más acerca del sensor y arduino para su conexión. Por último simulamos la conexión en Fritzing capturando el esquema de la misma.

**19/12:** Llegando a la etapa final, buscamos en internet ejemplos prácticos de el proyecto, como se aplica en un entorno real y todos los demás componentes necesarios. Para ello copiamos las configuraciones y lo agregamos a la documentación como complemento.

**21/12:** Por último recopilamos toda la información, como los esquemas, componentes, códigos y procesos, para concretar la documentación. Para finalmente realizar la entrega del proyecto.

# **COMPONENTES NECESARIOS CON PRESUPUESTO**

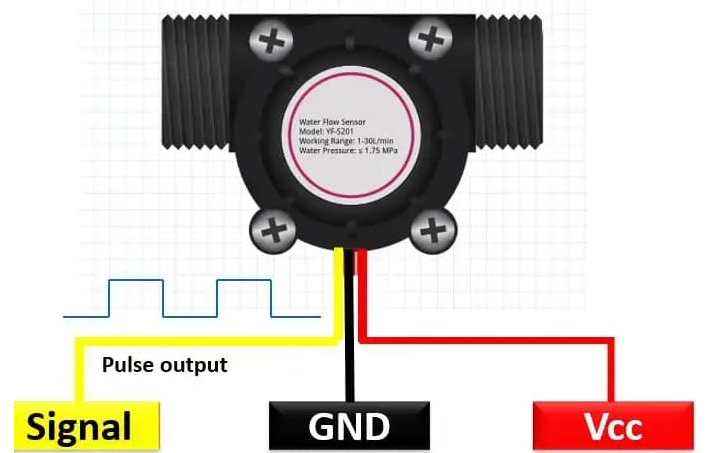
* Arduino Mega. **$34600**
* Sensor de flujo de agua YF-S201. **$8000**
* Bomba de agua de corriente continua. **- TOTAL DE COMPONENTES:**
* Driver de potencia BTS7960. **$16000 $58.600**
* Tanque con geometría esférica. **-**
* Tubo para transportar el agua. **-**

# **SENSOR DE FLUJO DE AGUA YF-S201**

Funcionamiento del YF-S201

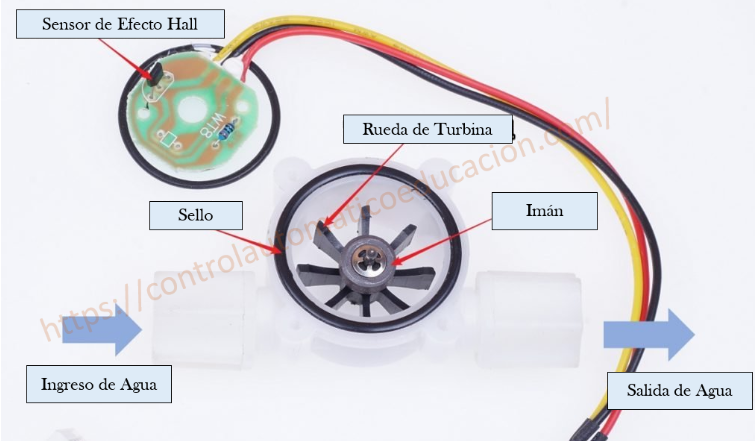
**¿Dónde se utiliza un sensor de flujo de agua como el YF-S201?** En industrias de automatización, especialmente en sectores como el de bebidas, la necesidad de monitorear con precisión la cantidad de líquido que se distribuye.Allí una línea de producción de refrescos: es esencial asegurar que cada botella se llene con la cantidad exacta de bebida. También encontramos estos sensores en máquinas de café, sistemas de riego inteligente, dispensadores automáticos de agua, entre otros.

El siguiente diagrama muestra el pinout del sensor de flujo de agua YF-S201. Está compuesto por tres pines:



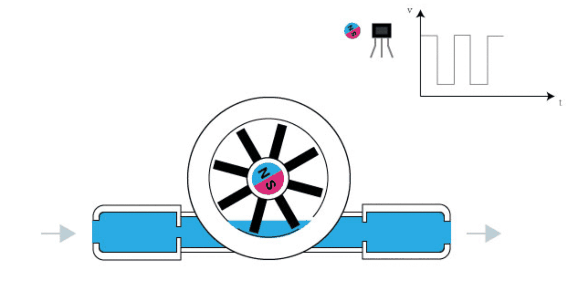
Componentes internos del sensor

Este sensor de flujo de agua integra dos componentes clave para su funcionamiento: un rotor (o rueda de turbina) y un sensor de efecto Hall.

En los componentes internos del sensor podemos encontrar una pequeña turbina. Cuando el agua entra, fluye por allí causando que gire. La velocidad de la turbina está directamente relacionada con la velocidad de flujo de agua que se transmite al sensor.

¿Cómo opera el sensor?

El sensor YF-S201 opera mediante el **Efecto** **Hall**. Se basa en la generación de u n voltaje cuando un conductor eléctrico atraviesa un campo magnético. En el caso del sensor, la turbina es la que contiene un pequeño imán integrado, que cuando gira altera el campo magnético.

El sensor detecta estas alteraciones que a cada revolución completada por la turbina, genera un pulso en el pin que corresponde a la salida. Cada pulso es directamente proporcional a la velocidad de rotación de la turbina, además del flujo del agua.

Características del sensor YF-S201

* Voltaje Operacional: 4.5 – 24 Volts
* Voltaje Normal: 5 – 18 Volts
* Corriente Máxima: 15 mA @ 5V
* Capacidad de Carga: ≤10 mA @ 5V
* Tasa de Flujo: 1 -30 L/min
* Temperatura Operacional: ≤ 80°C
* Temperatura operativa del fluido: ≤ 1200C
* Rango de humedad: 35% – 90% HR
* Resistencia de aislamiento: ≥ 100 Ohmios
* Sensor de flujo de agua que interactúa con Arduino

# **MEDICIÓN DE CAUDAL YF-S201 Y ARDUINO**

Para medir el caudal se necesita realizar distintas operaciones, es un proceso intuitivo. A medida que fluye el líquido a través, se deben contar los pulsos que son generados por el sensor. Esto puede ser contabilizado con pines de interrupción de **Arduino** para permitir medir la tasa de flujo en tiempo real.

Conversión de Pulsos

Cuando se utilizan sensores de flujo de agua como el YF-S201, una parte esencial es convertir los pulsos generados por el sensor en una medición de flujo de agua. Esta sección te guiará paso a paso a través de este proceso.

**PASO 1: La relación entre pulsos y volúmen**

Cada pulso generado por el sensor YF-S201 es equivalente a 2.25 ml de agua. Si conoces la cantidad de pulsos en un período de tiempo, puedes calcular el volumen de agua.

**PASO 2: Convertir pulsos en frecuencia (L/min)**

Para calcular el flujo de agua en litros por minuto ***(L/min)***, puedes utilizar la fórmula dada en la hoja de datos del sensor:

Esto significa que puedes calcular el flujo en litros por minuto usando un conteo de pulsos. Por ejemplo, si tu conteo de pulsos se encuentra en la variable ***pulseCount*** y tu flujo en la variable ***flowRate*** sería:

**PASO 3: Convertir litros por hora (L/h)**

Si prefieres tener la medición en ***litros*** por ***hora***, simplemente multiplica el resultado por 60:

En tu código de Arduino, esto se verá así:

1. flowRate1 = (pulseCount1 \* 60.0) / 7.5;

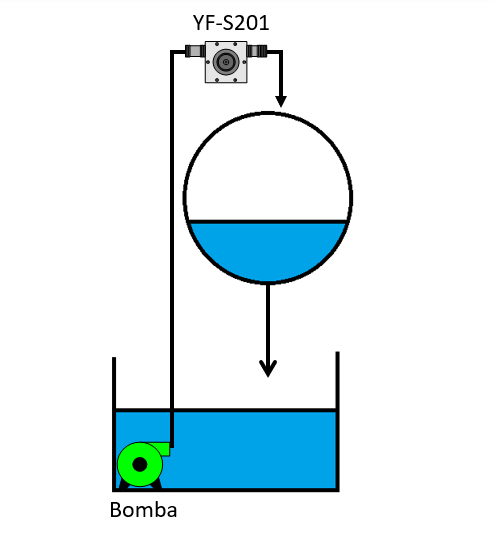
**¿Por qué 7.5?**

Te estarás preguntando de dónde viene el número 7.5 en estas ecuaciones. Este valor proviene directamente de las características del sensor y es la relación entre la frecuencia de pulsos y el flujo de agua en L/min. Es un valor específico para el sensor YF-S201 y es esencial para la conversión precisa.

# **Ejemplo práctico: Control de una bomba de corriente CC con Arduino y medición de caudal**

Objetivo

Modular una bomba de corriente continua (CC) para transportar agua desde un reservorio hasta un tanque con geometría esférica, midiendo la tasa de caudal con el sensor YF-S201 y mostrando el resultado a través del puerto serial.



Descripción

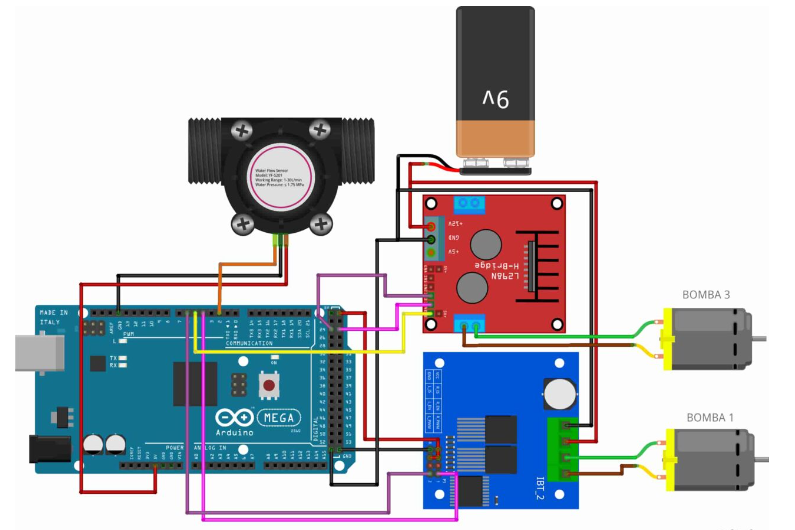
1. **Configuración Inicial:**

* Conectaremos la bomba de agua CC al driver de potencia BTS7960. El BTS7960 permitirá modular la bomba mediante PWM (Modulación por Ancho de Pulso) controlado por el Arduino.
* También explicaremos cómo realizar esta conexión con el driver L298, cuyo código de modulación es muy similar.

1. **Medición del Caudal:**

* Colocaremos el sensor YF-S201 en la tubería de ingreso al tanque esférico.
* Conectaremos el sensor al Arduino según las instrucciones previamente mencionadas

Circuito Fritzing



Código de Arduino

// BTS7960 - 1 (Bomba 1)

byte RPWM1 = 4;

byte LPWM1 = 6;

// l298 - 1 (Bomba 3)

byte IN3\_L1 = 24;

byte IN4\_L1 = 25;

byte BEN\_L1 = 5;

// Sensor YF-S201

byte flowSensor1Pin = 2;

volatile unsigned **long** pulseCount1 = 0;

**float** flowRate1 = 0.0;

unsigned **long** oldTime = 0;

//Estructura da Bomba L298

**typedef** **struct**{

byte enPin1;

byte enPin2;

byte pwmPin;

}BombaL;

//Estructura da Bomba BTS7960

**typedef** **struct**{

byte pwmA;

byte pwmB;

}BombaBTS;

//Crear las bombas con las estructuras

const BombaBTS bomba1 = {RPWM1, LPWM1};

const BombaL bomba3 = {IN3\_L1, IN4\_L1, BEN\_L1};

**float** inputValue = 0;

**int** velocity = 0;

String inputString = ""; // Una cadena para mantener los datos recibidos

boolean stringComplete = **false**; // Ya se recibió toda la cadena?

**void** setup() {

Serial.begin(9600); //Comunicación Serial

inputString.reserve(200); // Reserva espacio para la cadena

pinMode(flowSensor1Pin, INPUT\_PULLUP);

attachInterrupt(digitalPinToInterrupt(flowSensor1Pin), flowSensor1Interrupt, RISING);

}

**void** loop() {

//Transforma el valor de porcentaje a velocidad

velocity = map(inputValue,0,100,0,255);

// Ativar bomba1 (BombaBTS)

digitalWrite(bomba1.pwmA, LOW);

analogWrite(bomba1.pwmB, velocity);

/\*

// Ativar bomba3 (BombaL)

digitalWrite(bomba3.enPin1, HIGH);

digitalWrite(bomba3.enPin2, LOW);

analogWrite(bomba3.pwmPin, velocity);

\*/

//Calcular flujo cada 1segundo

**if** ((millis() - oldTime) > 1000) {

detachInterrupt(digitalPinToInterrupt(flowSensor1Pin));

flowRate1 = (pulseCount1 \* 60.0) / 7.5; // L/h (YF-S201: 1 pulso = 2.25 mL)

// Reiniciar conteo de pulsos y tiempo

pulseCount1 = 0;

oldTime = millis();

attachInterrupt(digitalPinToInterrupt(flowSensor1Pin), flowSensor1Interrupt, RISING);

// Mostrar la tasa de vazão

Serial.print("Flujo en el tanque 1: ");

Serial.print(flowRate1);

Serial.println(" L/h \n\n");

}

}

// Esta función se llama cada vez que el puerto serie recibe un byte:

**void** serialEvent() {

**while** (Serial.available()) {

**char** inChar = (**char**)Serial.read(); // Obtiene el nuevo byte

inputString += inChar; // Añade el byte a la cadena

**if** (inChar == '\n') { // Si es un byte de nueva línea, la entrada está completa

inputValue = inputString.toInt(); // Convierte la cadena a int

// Si inputValue está en el rango de 0 a 100

**if**(inputValue > 100){inputValue = 100;}

**if**(inputValue < 0){inputValue = 0;}

inputString = ""; // Limpia la cadena

}

}

}

//Función de la interrupción

**void** flowSensor1Interrupt() {

pulseCount1++;

}