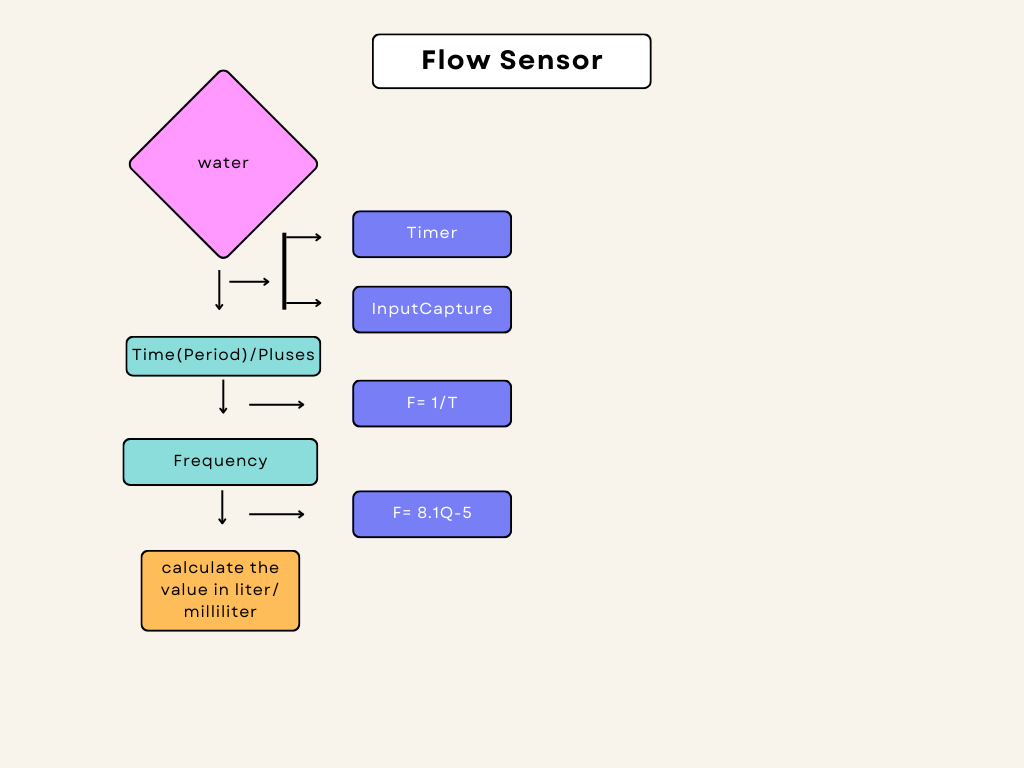
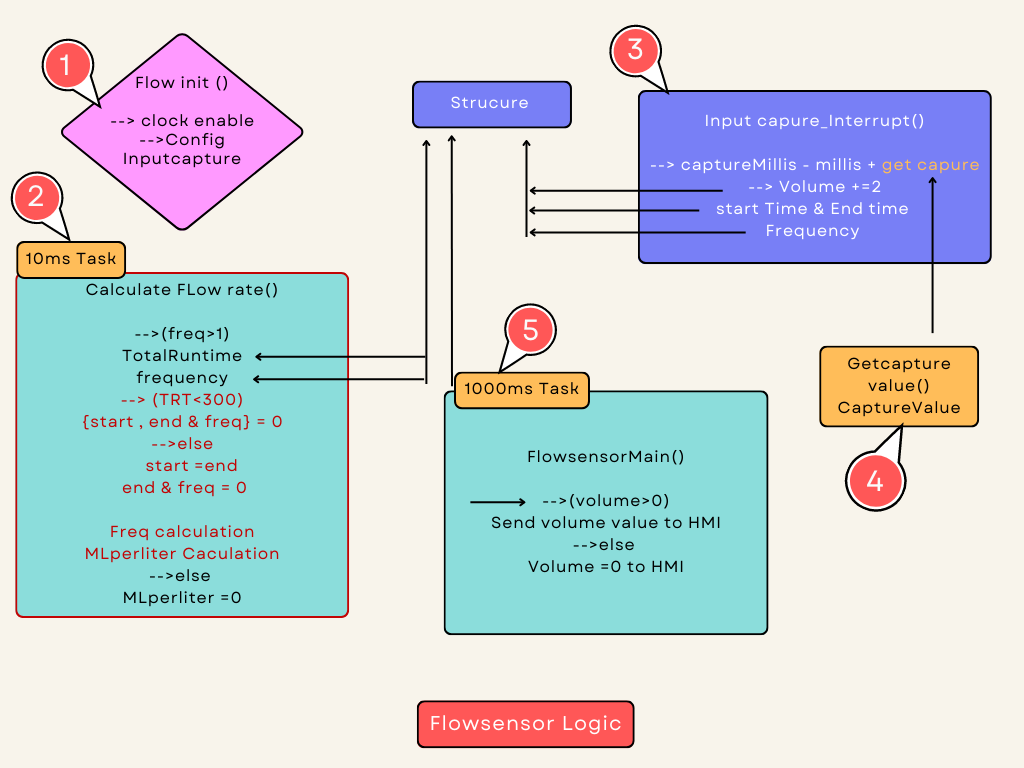
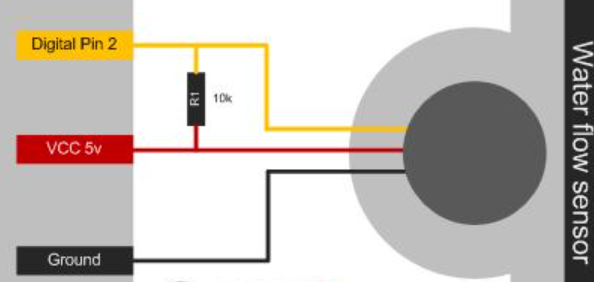
**FLOW Sensor**

**BLOCK DIAGRAM**



Flow sensor is used the find the consumed water level at cooking and washing process.





>> Flow sensor is used here to sense the presense of water level in cooking mode as well as the washing mode.

>> It is also used to find the volume of the water consumption inside the oven and updates in diagnostic page.

>> flow sensor can be used for accurate flow measurement.when water through the flow rotor, the magnetic rotor will rotate and speed will change as the flow change.

SPEC & PIN DETAILS

* its a 3/4 flow sensor 45mm.
* working voltage range is 5v-18v.
* current rating is 5ma(dc5v)
* it has 3pins
* yellow - input
* red- vcc
* black -gnd

**Using Macros**

* **FLOWRATE\_LM\_TO\_MLM\_CONVERTION\_FACTOR**
* This macro is used to convert Litre per minute to Mille litre per minute.
* **FLOWRATE\_MULTIPLICATION\_FACTOR**

**🡪**For using Error corrections

**Using variables**

* **CaptureMillis**🡪 this variable contains the total period of incoming water
* **Millis**🡪 this variable is used to update the **CaptureMillis**variable during every timer overflow.
* **NoWaterIndicatorCount**🡪this variable is used to find the water is incoming or not;
* **WaterInletOffIndicateCounter**🡪 this variable is used to measure the presence of water for every 100 ms to update the NoWaterIndication flag.
* **FrequencyCounter🡪**It is used to contains the no of pulse.
* **TotalRunTime**🡪 it used to contains the total time incoming water.

**Flow of State**

* **Flowsensor\_init()**
* **Calculate Flowrate()(10ms Task)**
* **Inputcapture\_interrupt()**
* **Getcapture\_value()**
* **Flow\_main()(1000ms task)**

**EXPLAINATION**

1. **FlowsensorInit()**

Flow sensor Init() function Enable the clock and Config the Inputcapture in the interrupt method.

1. **Calculate flowrate()(10ms task)**

* This function is called by every 10ms task,
* This is calculates and manages flow rate-related information.
* This function is calculated by the Freq calculation

🡪 It calculates the CH1.freqency by dividing FrequencyCounter by TotalRunTime (scaled to thousands) and rounding to the nearest integer using +0.5.

* MLperMinuteCalculation.

🡪 It calculates CH1.MLPerMinute based on the frequency and some constants (FLOWRATE\_MULTIPLICATION\_FACTOR and (FLOWRATE\_LM\_TO\_MLM\_CONVERTION\_FACTOR).

🡪This calculation seems to convert the frequency into milliliters per minute.

* Totalruntime and freqvalue are get from theST.
* (IsWaterInletOpenFlag == TRUE)

🡪This likely indicates whether the water inlet valve is open.

🡪If the flag is TRUE, it proceeds to further checks related to water flow.

🡪**(CH1.MLPerMinute < 500)**

🡪 If CH1.MLPerMinute is less than 500 milliliters per minute, it increments a counter named **NoWaterIndicatorCount**. This counter likely tracks how long the flow rate has been below the threshold.

🡪**(NoWaterIndicatorCount> 300)**

🡪 If**NoWaterIndicatorCount** exceeds 300 (likely representing 3 seconds of low flow), it takes several actions:

🡪Sets a flag in the MachineStatus structure **(MachineStatus->NWF)** to indicate a "No Water Flow" condition.

🡪Sends a machine status update using **Send\_MachineStatus().**

🡪Reports a diagnostic event, possibly indicating a problem with steam

🡪 Adjusts the **washState** and **steamControl** variables based on certain conditions.

🡪Resets the **NoWaterIndicatorCount** to 0.

🡪**Flow Rate Normalization**

🡪If the flow rate is greater than or equal to 500 milliliters per minute, it resets the **NoWaterIndicatorCount** to 0 and clears the "NWF" flag in MachineStatus.

🡪**Water Inlet Valve Check**

🡪If the water inlet valve flag (IsWaterInletOpenFlag) is FALSE, it increments a counter named WaterInletOffIndicateCounter.

🡪If this counter exceeds 250 (likely representing a certain time threshold), it checks if the flow rate is still high (greater than 400 milliliters per minute).

🡪If the flow rate is high, it may report an error related to the water inlet valve not turning off.

1. **interrupt handler()**

**🡪** This will be used as the TIM Capture/Compare (CC1) interrupt service routine (ISR). It handles tasks related to this interrupt source.

**🡪Update Interrupt Handling**

🡪If the update interrupt flag (TIM\_SR\_UIF) is set, it increments a global variable Millis by 65,000. This variable likely represents the number of milliseconds elapsed.

🡪**Capture/Compare 1 Interrupt Handling**

🡪If the capture/compare 1 interrupt flag (TIM\_SR\_CC1IF) is set, it proceeds with the following actions:

🡪Calculates the **CaptureMillis** by adding the current value of **Millis** to the value captured by TIM1's capture/compare channel 1.

🡪Increases the **CH1.volume by 2**. This incrementation is likely related to counting a specific event (2 ml/pulse).

🡪If **CH1.StartTimeStamp** is 0, it sets **CH1.StartTimeStamp** to the calculated **CaptureMillis**. This is presumably the starting timestamp for measuring time intervals.

🡪Sets **CH1.EndTimeStamp** to the calculated CaptureMillis, effectively capturing the end timestamp for time interval measurement.

🡪Increments CH1.freqencyCounter, which is likely counting the frequency of pulses.

🡪 To get Starttime, Endtime, frequency value and Total volume value. these values are stored in the ST.

1. **Get capture value()**

**🡪**This function is retrieves the captured value from a specific capture/compare channel of a timer.

1. **Flow sensor Main () (1000ms task).**

🡪In this function, the data’s are rearranged (volume (data)) is shared withHMI.

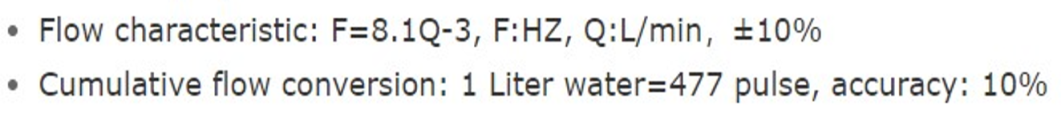
**CALCULATIONS**

It is called in 10 ms task but whenever the value frequency counter is greater than 1 then only the calculation proceded in void calculateFlowRate().

**CH1.freqency=(UL)(((FrequencyCounter\*1000)/TotalRunTime) + 0.5);**

**CH1.MLPerMinute=(UL)((((CH1.freqency+5)\*FLOWRATE\_MULTIPLICATION\_FACTOR)/81)\*FLOWRATE\_LM\_TO\_MLM\_CONVERTION\_FACTOR);**

**T**his part of the code is used to calculate the flow rate in ml per minute.this calculation is done by using the formula that is given below,



for example,

frequency counter value = 5,totalruntime=60

CH1.frequency = (((5\*1000)/1000)+0.5)

= 83.83

CH1.MLPerMinute =(( (83.83+5)\*10)/81)\*1000 🡺 1652.2