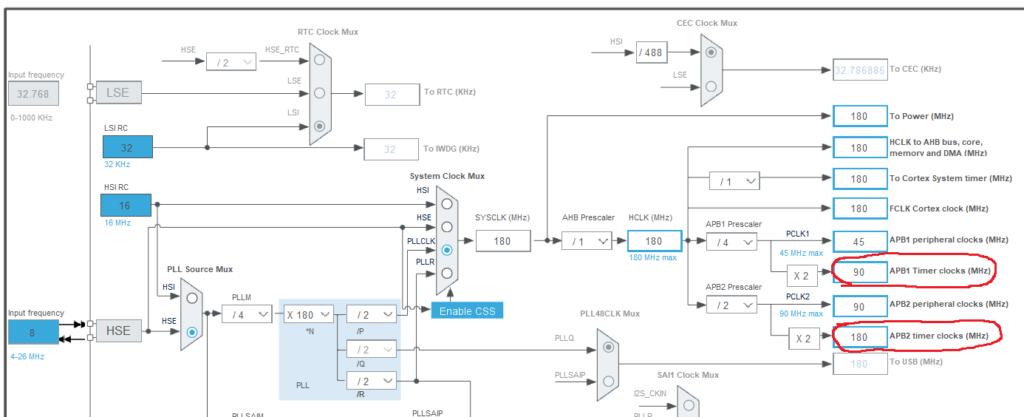
**Input Capture in STM32**

This is yet another tutorial in the Timer Series in STM32, and today we will see how to use Input Capture in STM32. We will use the input capture to measure the Frequency and width of the input signal. Let’s see the Cube MX setup first.

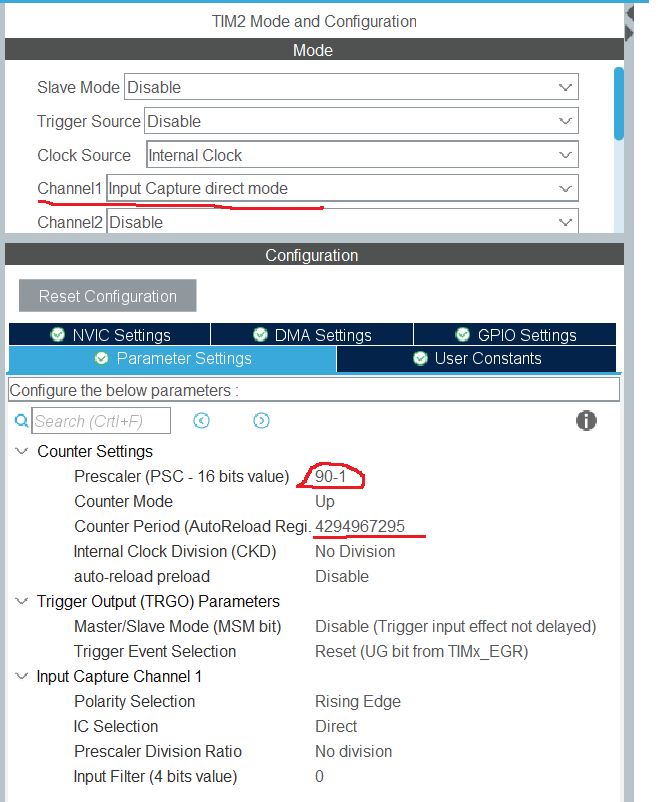
**CubeMX Setup**

[](https://controllerstech.com/wp-content/uploads/2021/09/IC_1.png)

CLOCK CFG

* Here I am going to use two timers, i.e Timer 1 for the PWM output and Timer 2 for the Input Capture
* Timer 1 is connected to the APB2 clock, which is running at 180 MHz
* Timer 2 is connected to the APB1 clock, which is running at 90 MHz

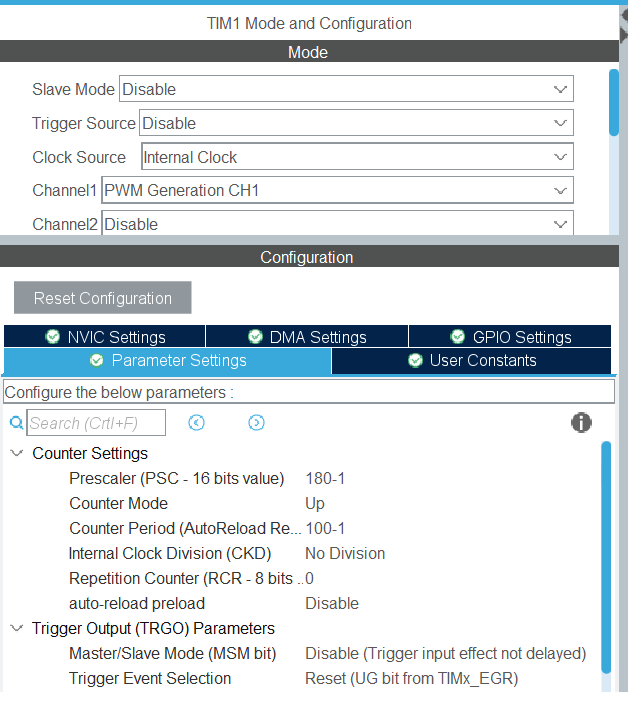
This part is very important. The minimum frequency, which the device can read will depend on it. Below is the configuration for the TIMER 2

[](https://controllerstech.com/wp-content/uploads/2021/09/IC_2.png)

TIM2 CFG

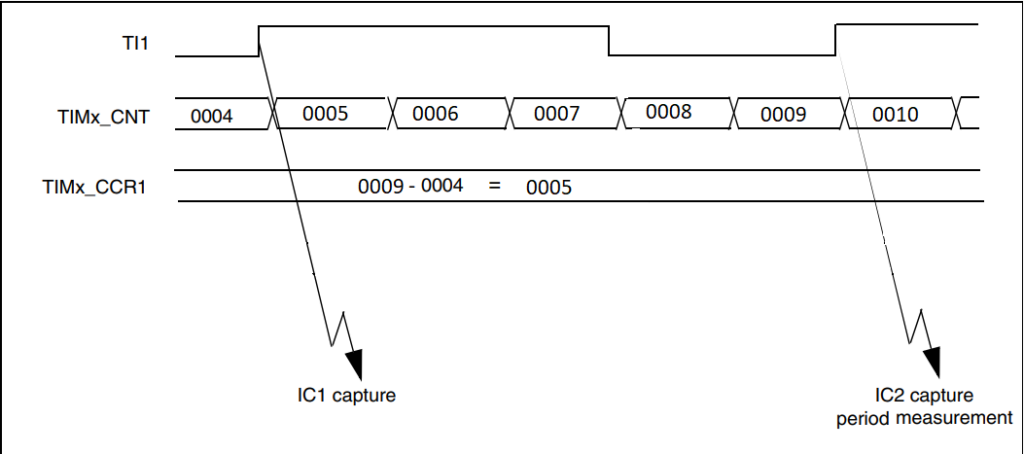
* I have enabled the Input capture Direct Mode for channel 1
* The Prescalar is set o 90, which would divide the APB2 clock by 90, making the Timer 2 clock = 1 MHz
* I am leaving the ARR to 0xffffffff (Max for 32 bit Timer)
* The minimum frequency that the Timer can read is equal to (TIMx CLOCK/ARR). In our case it will be (1MHz/0xffffffff) Hz.

The TIMER1 is configures to give a PWM output signal. If you want to know more about PWM output, check out the tutorial [PWM (Pulse Width Modulation) in STM32 – Controllerstech.com](https://controllerstech.com/pwm-in-stm32/)



**Measuring Frequency**

In order to measure the Frequency of the input signal, we need to measure the time between the 2 rising edges, or between 2 falling edges. Below is the figure explaining the same

[](https://controllerstech.com/wp-content/uploads/2021/09/IC_7.png)

* When the first Rising edge occurs, the counter value is recorded.
* Another counter value is recorded after the second rising edge occurs.
* Now the difference between these 2 counter values is calculated.
* The Difference in the counter values will give us the frequency.
* This entire process is shown below

#define TIMCLOCK 90000000

#define PRESCALAR 90

uint32\_t IC\_Val1 = 0;

uint32\_t IC\_Val2 = 0;

uint32\_t Difference = 0;

int Is\_First\_Captured = 0;

/\* Measure Frequency \*/

float frequency = 0;

void HAL\_TIM\_IC\_CaptureCallback(TIM\_HandleTypeDef \*htim)

{

if (htim->Channel == HAL\_TIM\_ACTIVE\_CHANNEL\_1)

{

if (Is\_First\_Captured==0) // if the first rising edge is not captured

{

IC\_Val1 = HAL\_TIM\_ReadCapturedValue(htim, TIM\_CHANNEL\_1); // read the first value

Is\_First\_Captured = 1; // set the first captured as true

}

else // If the first rising edge is captured, now we will capture the second edge

{

IC\_Val2 = HAL\_TIM\_ReadCapturedValue(htim, TIM\_CHANNEL\_1); // read second value

if (IC\_Val2 > IC\_Val1)

{

Difference = IC\_Val2-IC\_Val1;

}

else if (IC\_Val1 > IC\_Val2)

{

Difference = (0xffffffff - IC\_Val1) + IC\_Val2;

}

float refClock = TIMCLOCK/(PRESCALAR);

frequency = refClock/Difference;

\_\_HAL\_TIM\_SET\_COUNTER(htim, 0); // reset the counter

Is\_First\_Captured = 0; // set it back to false

}

}

}

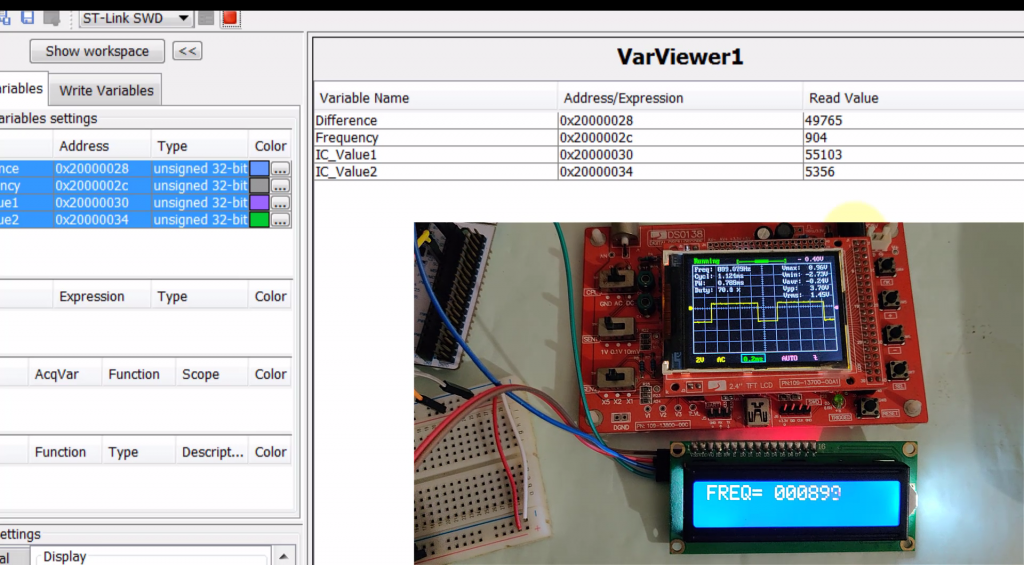
* The above callback function is called whenever the rising edge is detected.
* When called first time, ***Is\_First\_Captured***was 0 so the hence the ***IC\_Val1***will be recorded.
* When called after the second rising edge, the ***Is\_First\_Captured*** is 1 now so IC\_Val2 will be recorded.
* We will then calculate the Difference between the 2 values.
* Reference clock is calculated based on the setup we have done for our timer.
* Frequency is equal to the (Reference clock / Difference).

In the main function, we have to start the TIMER in the Input capture interrupt mode. I am also starting the Timer 1 in the PWM mode, so to provide the signal for the Timer2.

TIM1->CCR1 = 50;

HAL\_TIM\_PWM\_Start(&htim1, TIM\_CHANNEL\_1);

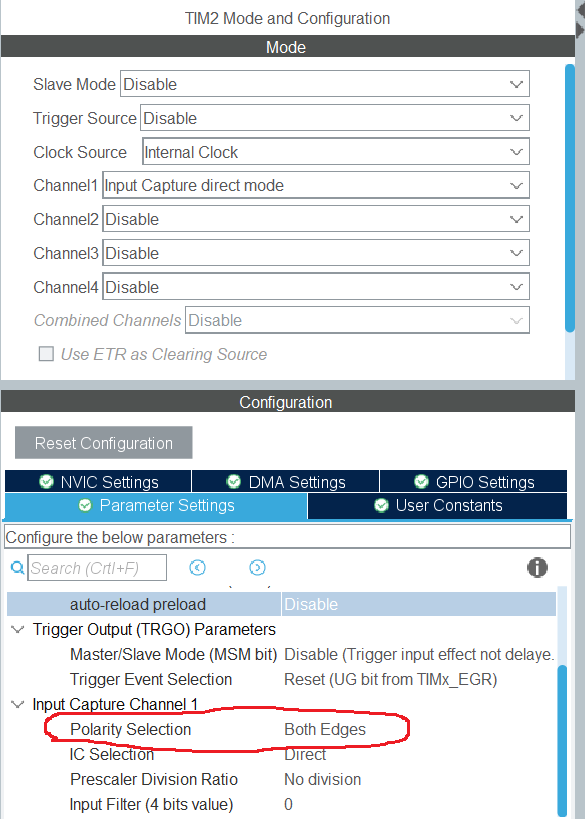
HAL\_TIM\_IC\_Start\_IT(&htim2, TIM\_CHANNEL\_1);

[](https://controllerstech.com/wp-content/uploads/2019/05/pic.png)

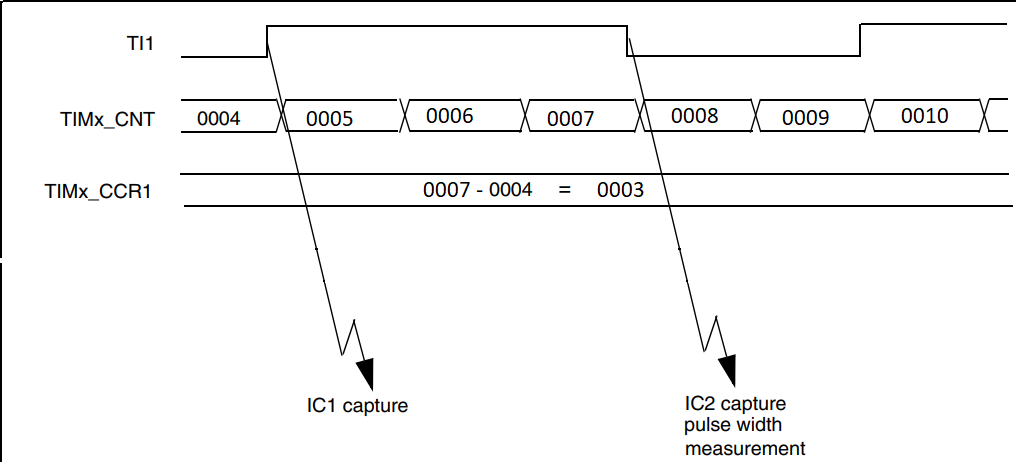
Frequency measured using input capture

**Measuring Pulse Width**

In order to measure the Pulse width, we want the interrupt to trigger on both the edges of the incoming signal. To do so, we need to modify the CubeMX as shown below.



As you can see in the figure above, I have switched the Polarity to both the edges. This will make the interrupt to trigger on both the edges of the incoming signal

[](https://controllerstech.com/wp-content/uploads/2021/09/IC_8.png)

* When the first rising edge occurs, The counter value is stored in the ICVal 1
* The next interrupt will occur at the falling edge, and the counter value is stored in the IC val 2
* The Pulse width can be calculated using this counter value.
* The process for the same is shown below

/\* Measure Width \*/

uint32\_t usWidth = 0;

void HAL\_TIM\_IC\_CaptureCallback(TIM\_HandleTypeDef \*htim)

{

if (htim->Channel == HAL\_TIM\_ACTIVE\_CHANNEL\_1) // if the interrupt source is channel1

{

if (Is\_First\_Captured==0) // if the first value is not captured

{

IC\_Val1 = HAL\_TIM\_ReadCapturedValue(htim, TIM\_CHANNEL\_1); // read the first value

Is\_First\_Captured = 1; // set the first captured as true

}

else // if the first is already captured

{

IC\_Val2 = HAL\_TIM\_ReadCapturedValue(htim, TIM\_CHANNEL\_1); // read second value

if (IC\_Val2 > IC\_Val1)

{

Difference = IC\_Val2-IC\_Val1;

}

else if (IC\_Val1 > IC\_Val2)

{

Difference = (0xffffffff - IC\_Val1) + IC\_Val2;

}

float refClock = TIMCLOCK/(PRESCALAR);

float mFactor = 1000000/refClock;

usWidth = Difference\*mFactor;

\_\_HAL\_TIM\_SET\_COUNTER(htim, 0); // reset the counter

Is\_First\_Captured = 0; // set it back to false

}

}

}

* The above callback function is called when either the Rising or Falling edge is detected.
* When called for the rising edge, ***Is\_First\_Captured***was 0 so the hence the ***IC\_Val1***will be recorded.
* When called for the falling edge, the ***Is\_First\_Captured*** is 1 now so IC\_Val2 will be recorded.
* We will then calculate the Difference between the 2 values.
* This difference is the time for which the Pulse is high, and this time will depend on the timer configuration.
* To measure this time in microseconds, we have to use the reference clock.
* Reference clock is calculated based on the setup we have done for our timer.
* And finally, the usWidth will be calculated.