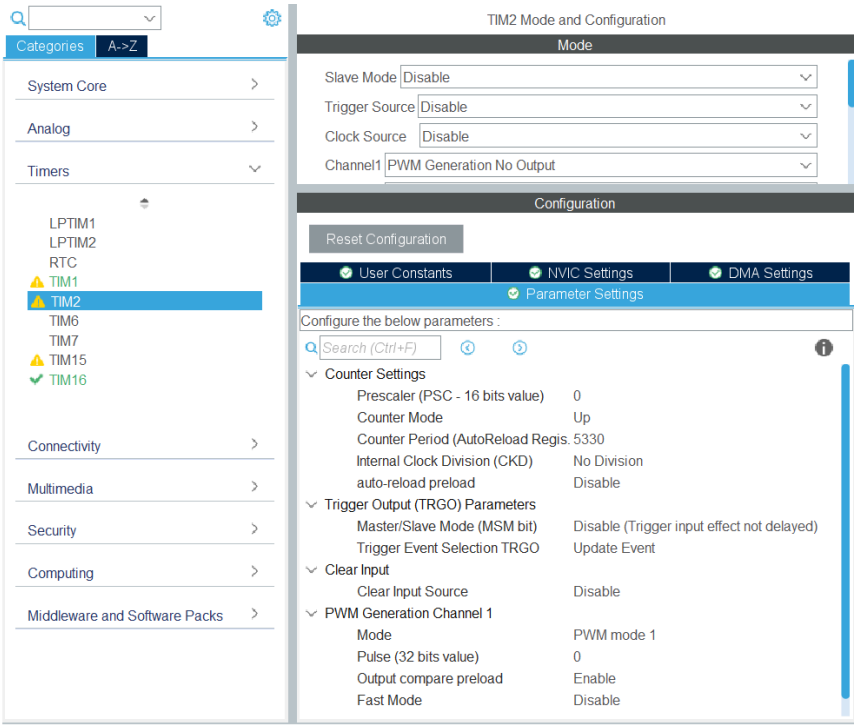
Configured STM32 Documentation

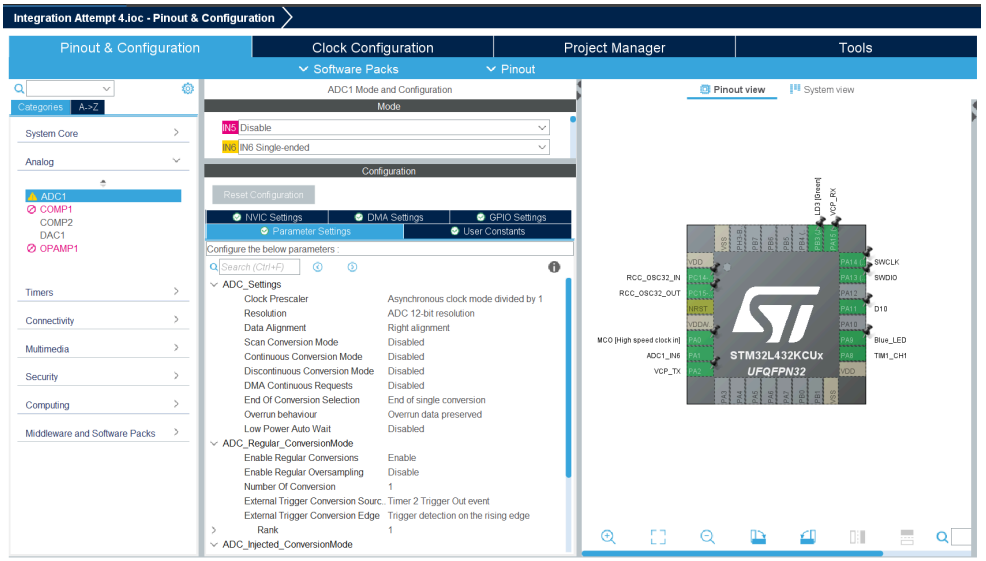
ADC Documentation

The ADC is triggered by TIM2 PWM pulses and is configured as follows:



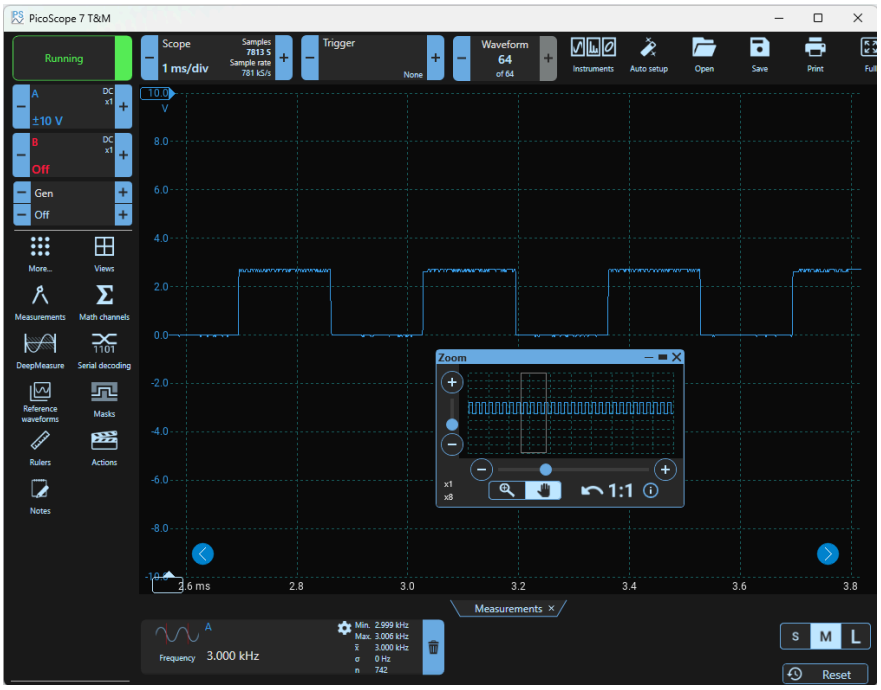
Channel 1 is set to PWM generation with no output, as the PWM signal is not required on an IO pin of the microprocessor. The counter period is set to 5330 which creates a ~6000 Hz signal as SYS_CLK(32MHz) divided by 5330 = 6003.75.

ADC1 is configured in mode IN6 Single-Ended, which reads the analogue value from pin PA1.



Regular conversions are enabled, with the external trigger conversion source set to Timer 2 Trigger Out event, with the trigger caused by a rising edge.

An external blue LED running as a GPIO_output on pin PA11 is configured to toggle on ADC callback conversion, but due to the speed, its flickering is not visible to the human eye, and it is only used to show that callbacks are occurring, since it is set to switch off if the error handler is called. However, when viewing its signal on a scope shows an average frequency of 3000 Hz, meaning the ADC is triggered at twice that, 6000Hz.

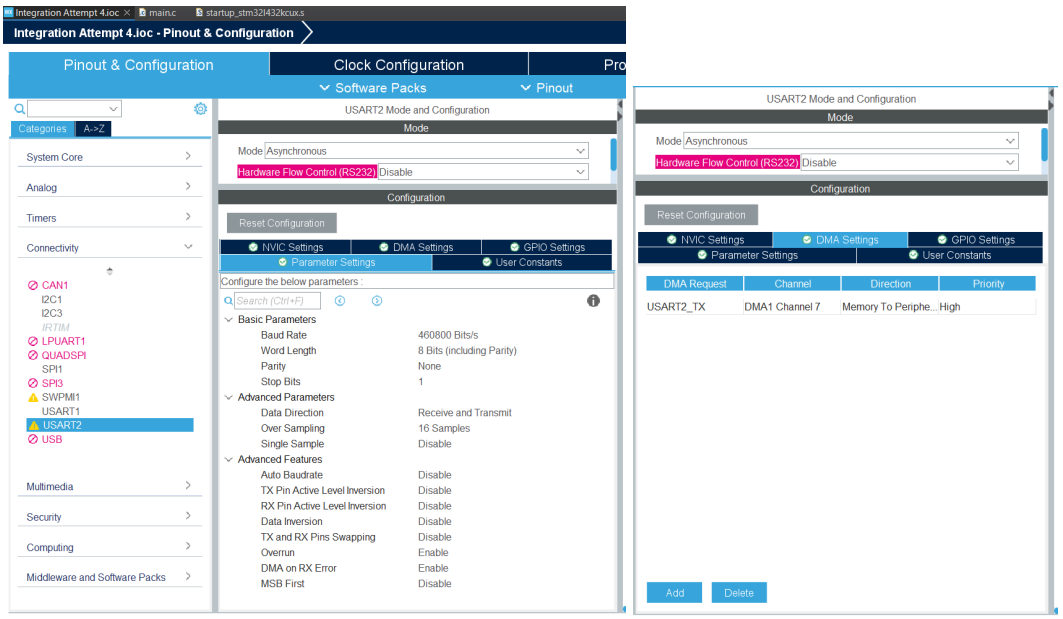


Below, the code for the ADC callback can be seen:

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef *hadc) {
    raw = HAL_ADC_GetValue(&hadc1);

    uint8_t *ADCValPtr = (uint8_t *)(&raw);
    int valLength = sizeof(raw);
    HAL_UART_Transmit_DMA(&huart2, ADCValPtr, valLength);

    HAL_GPIO_TogglePin(Blue_LED_GPIO_Port, Blue_LED_Pin);


}
```

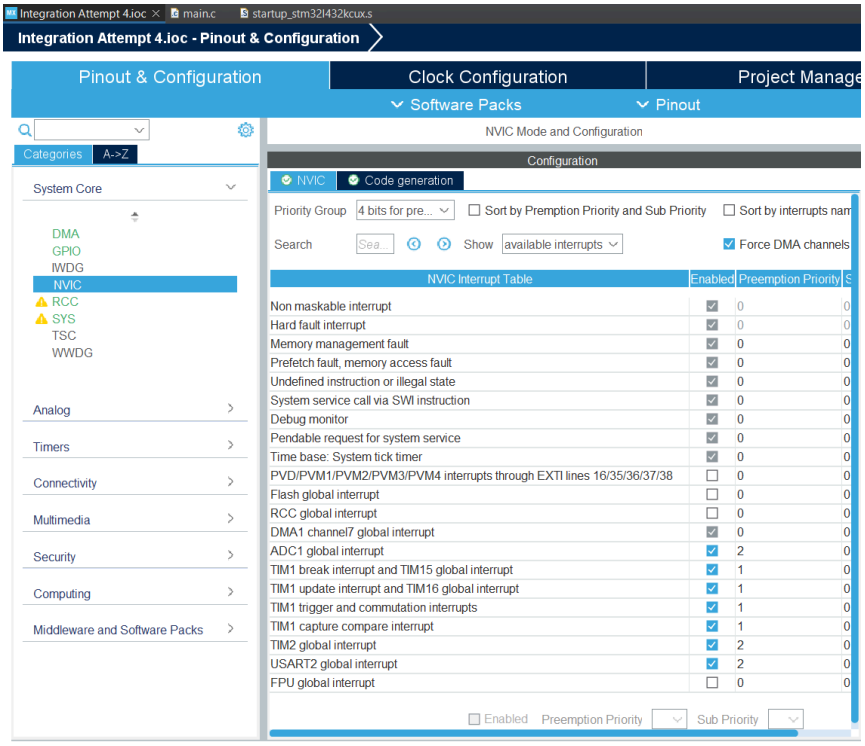
UART Documentation

UART2 is used for serial communication with the CLI (as UART2 utilises the onboard ST-Link to communicate over the micro USB virtual COM port). Its baud rate is increased from the default 115200 bit/s to 460800 bit/s to allow for the highest data throughput possible while not compromising stability. A DMA request is set up on USART2_TX with high priority in the memory to peripheral mode to get the DMA controller to transmit UART data instead of the CPU, freeing it for other tasks.



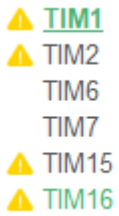
NVIC Interrupt Documentation

Interrupt priorities are configured as shown, with many global interrupts enabled and preemption priorities set. Ultrasonic timing is given the highest priority (1) as the timing must be the most accurate/precise. ADC and UART are given secondary priority (2) as these are not as important.

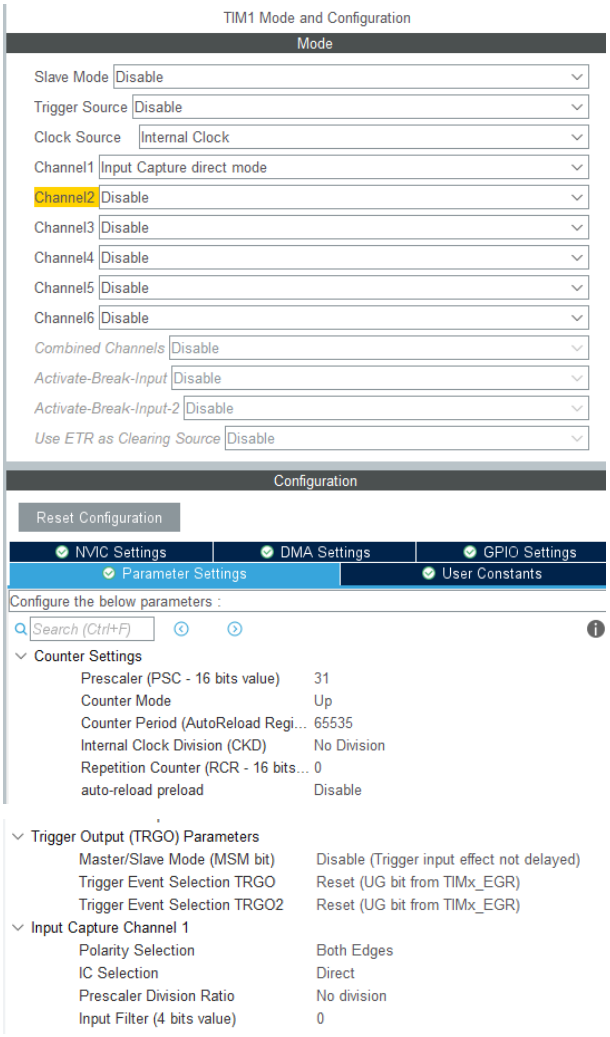


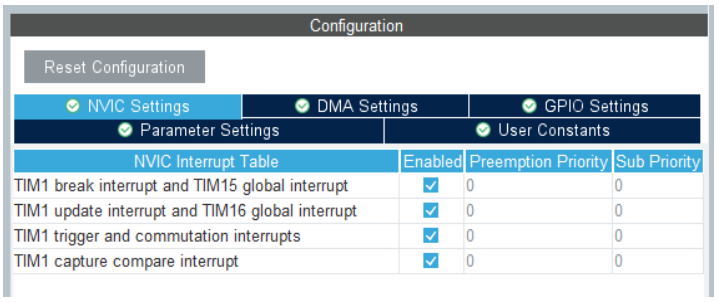
Ultrasonic Documentation

The ultrasonic sensor runs off of two interrupts, configured with **TIM1** and **TIM16**.

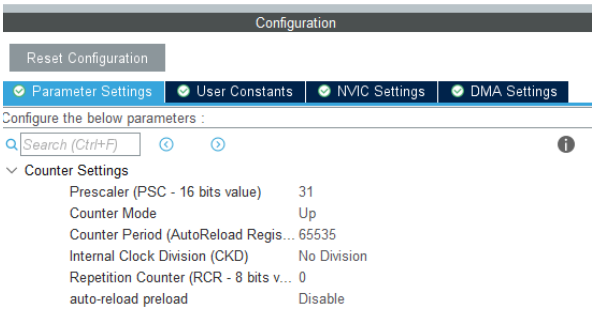


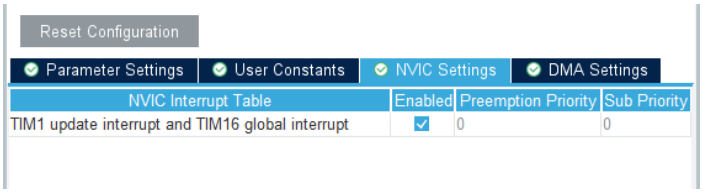
TIM1 works as an Input Capture interrupt and is configured as follows:



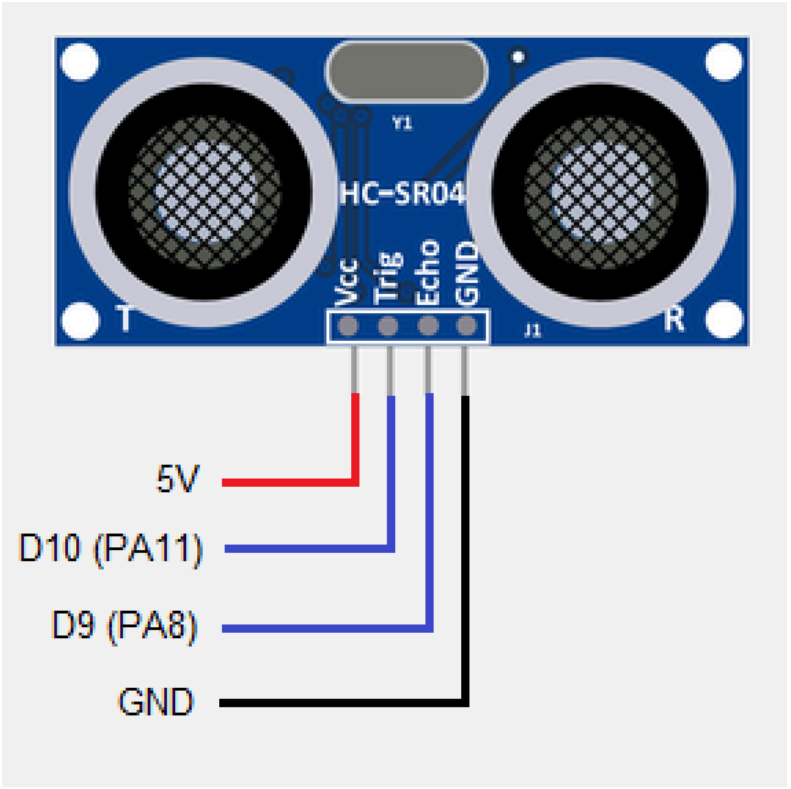


TIM16 works as a timer interrupt and is configured as follows:

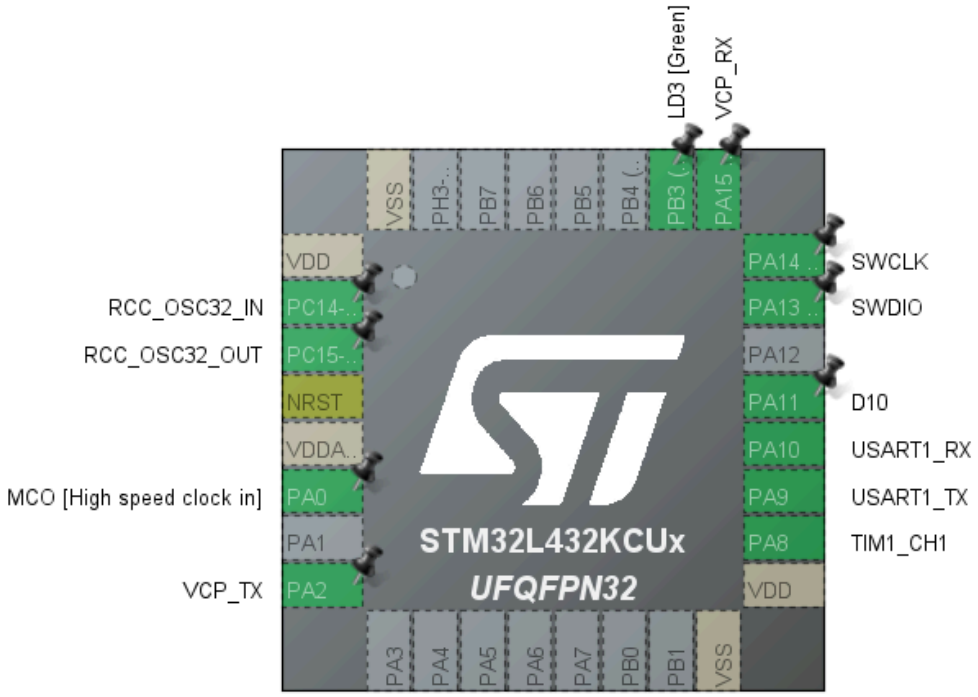




The MCU (STM32) interfaces with the Ultrasonic as described in the following diagram:



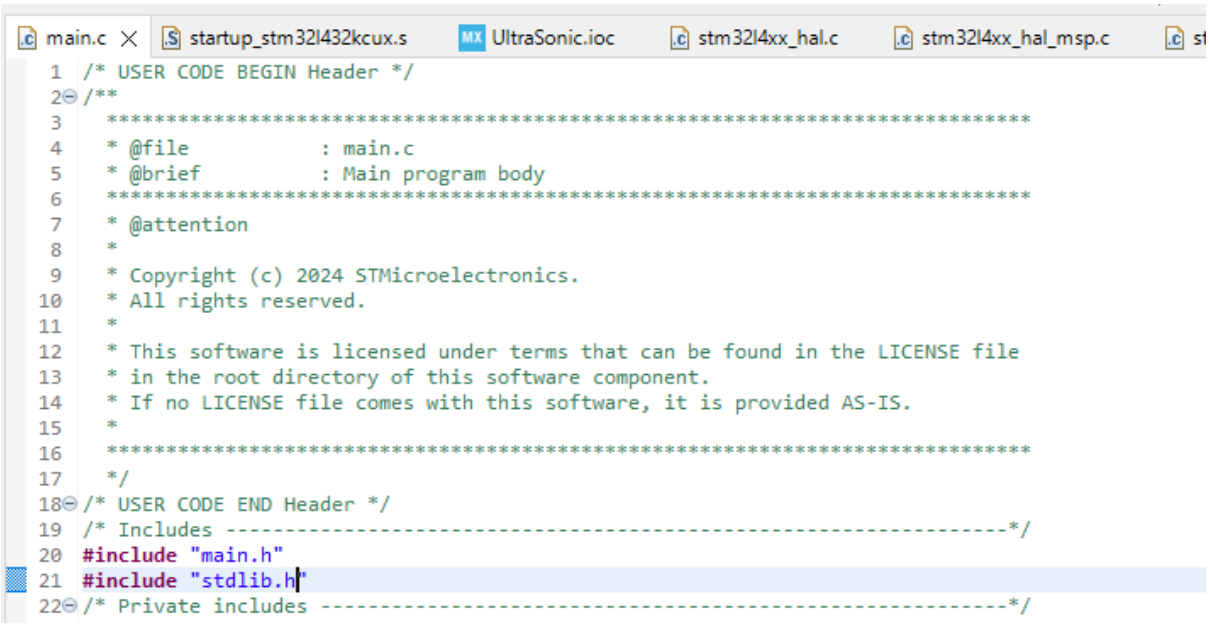
This is the config for the STM32 in CubeIDE (Note: D10 is a GPIO_Output pin)



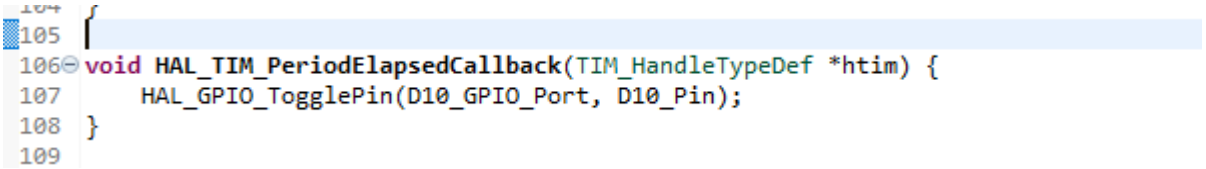
The callback function for **TIM1** is as follows:

```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

int Is_First_Captured = 0;
uint16_t timer2 = 0;
uint16_t timer1 = 0;

uint16_t usTime = 0;

char flag = 'S';
uint8_t * flagpointer = (uint8_t *) &flag;

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim) {

    if (Is_First_Captured==0)
        {
            timer1 = (uint16_t) HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
            Is_First_Captured = 1;
        }

        else

        {
            timer2 = (uint16_t) HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);

            usTime = abs(timer1 - timer2);

            if (usTime > 20000){
                return;
            }
             uint8_t * datapointer = (uint8_t *) &usTime;

            HAL_UART_Transmit(&huart2, flagpointer, 1, 100);
            HAL_UART_Transmit(&huart2, datapointer, 2, 100);


            __HAL_TIM_SET_COUNTER(htim, 0);  // reset the counter
            Is_First_Captured = 0; // set back to false

        }
}

```

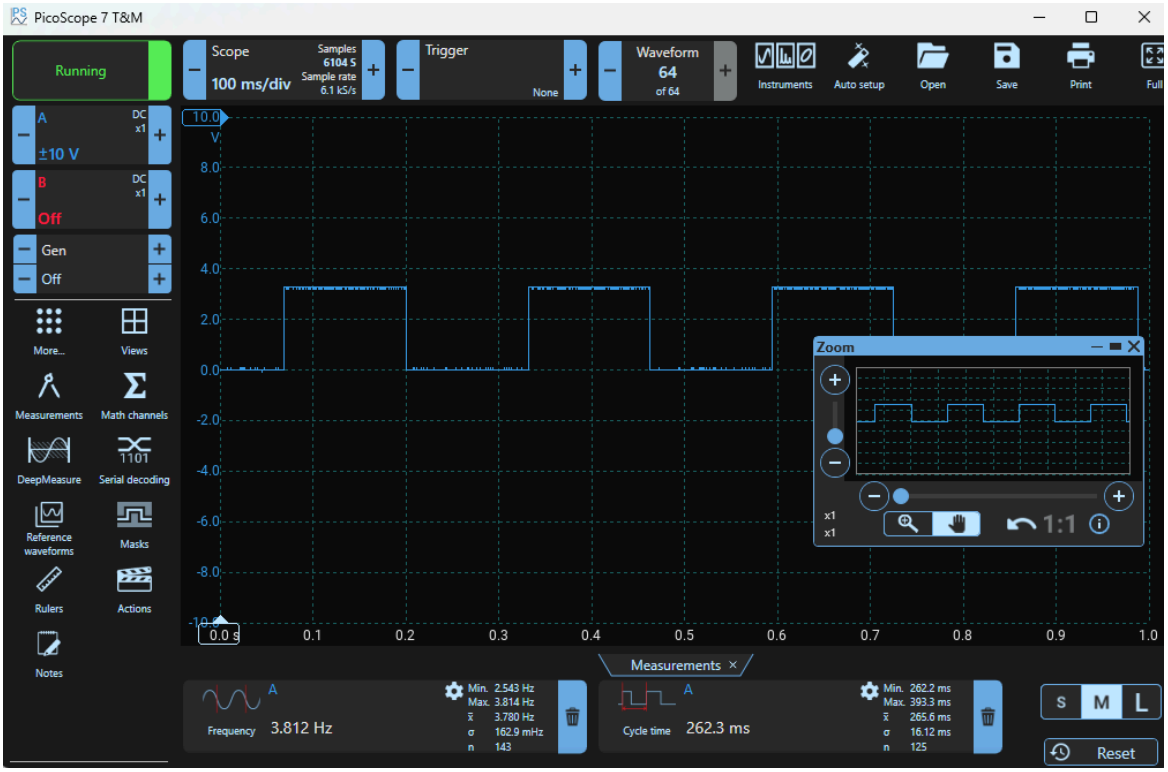
Note: the function “abs” is a stdlib function, you **must include stdlib.h under the “Includes” header as follows:**



The callback function for **TIM16** (Timer interrupt) is as follows:



Explanation:
TIM16 interrupts every time it counts over 65535 microseconds (the counter period), and toggles D10 (the trigger pin) in the callback function. This means that every ~130 milliseconds the trigger pin goes high and a sonic burst is sent from the Ultrasonic Sensor. This can be seen, as onboard LD3 is toggled every time the US callback function is triggered. When connected to a scope we see the following:



Here, the cycle time is measured at 262ms, which is the time taken for two US measurements to be made since the LD3 pin is toggling.

TIM1 interrupts every time D9 (the echo pin) has an edge, i.e. every time D9 goes from low to high or high to low. So, to measure the amount of time between each edge, *HAL_TIM_ReadCapturedValue* is stored at each edge (timer1 and timer2), and the difference between them is taken (usTime)..

There is a chance that the US does not receive a signal back, in which case the value of usTime will be ~65000. To handle this error in the IC function we check if usTime is greater than 20,000, and if it is the function does not transmit the value.

* 20,000 is greater than what the US could receive at its max range. This is why the prescaler value of 32-1 was chosen, we are triggering the US sensor at the maximum frequency possible to allow for the echo to be received at max distance before a new trigger is sent.