LAB: Input Capture - Ultrasonic

Date: 2023-10-31

Author: Gyeonheal An

Github: https://github.com/AnGyeonheal/Embedded_Control_GH

Demo Video: https://www.youtube.com/shorts/gBAAXINBI0E

I. Introduction

In this lab, you are required to create a simple program that uses input capture mode to measure the distance using an ultrasonic distance sensor. The sensor also needs trigger pulses that can be generated by using the timer output.

i. Requirement

Hardware

- MCU
  - NUCLEO-F411RE
- Actuator/Sensor/Others:
  - HC-SR04
  - breadboard

Software

- Keil uVision, CMSIS, EC_HAL library
- Clion(with PlatformIO core plugin)
  Library: STM32Cube library package(Official), EC_HAL library
  Compiler: GNU Arm Embedded Toolchain
  Debugger: ST-Link

II. Problem 1: Create HAL library

i. Create HAL library

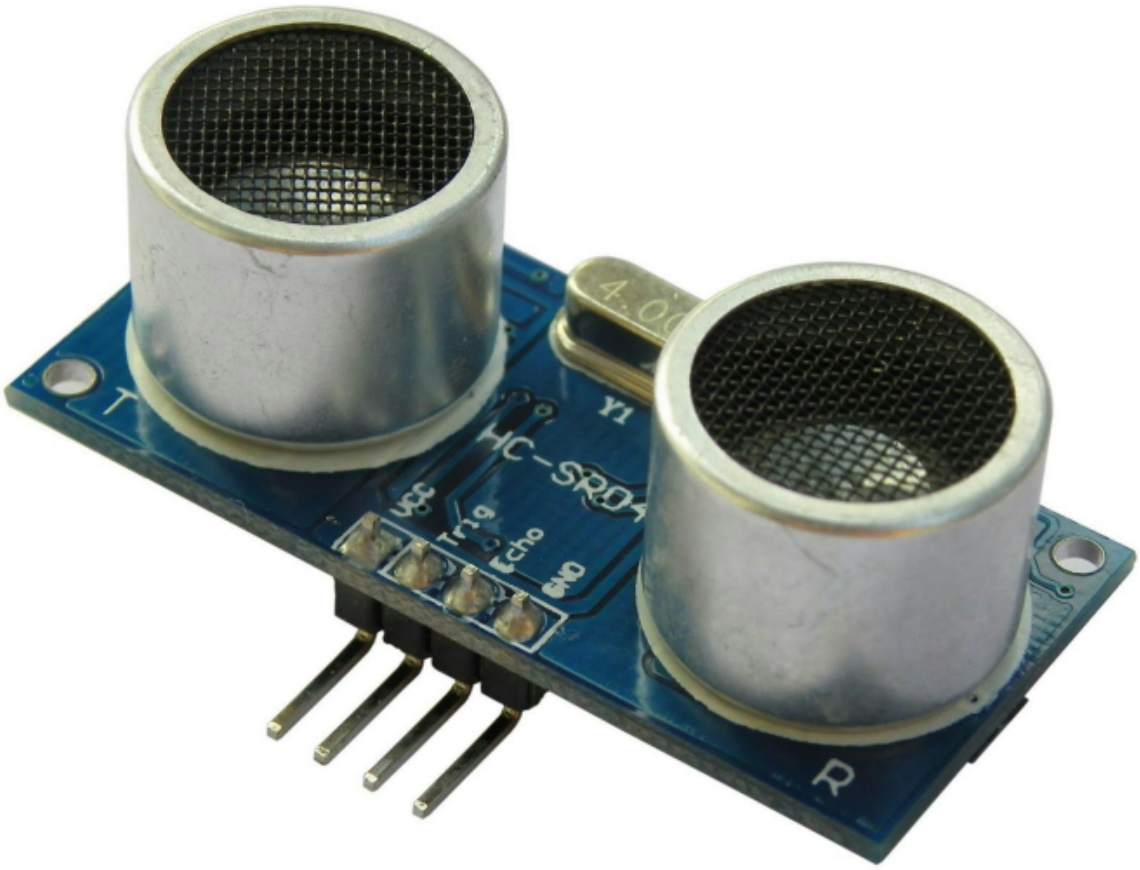
ecTIM.h

```
/* Input Capture*/
// ICn selection according to CHn
#define FIRST 1
#define SECOND 2

// Edge Type
#define IC_RISE 0
#define IC_FALL 1
#define IC_BOTH 2

// IC Number
#define IC_1 1
#define IC_2 2
```

```
#define IC_3 3
#define IC_4 4

void ICAP_pinmap(PinName_t pinName, TIM_TypeDef **TIMx, int *chN);
void ICAP_init(PinName_t pinName);
void ICAP_setup(PinName_t pinName, int ICn, int edge_type);
void ICAP_counter_us(PinName_t pinName, int usec);
uint32_t ICAP_capture(TIM_TypeDef* TIMx, uint32_t ICn);
uint32_t ICAP_read(TIM_TypeDef *TIMx);
```

III. Problem 2: Ultrasonic Distance Sensor (HC-SR04)

The HC-SR04 ultrasonic distance sensor. This economical sensor provides 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm. Each HC-SR04 module includes an ultrasonic transmitter, a receiver and a control circuit.



The HC-SR04 Ultrasonic Range Sensor Features:

- Input Voltage: 5V
- Current Draw: 20mA (Max)
- Digital Output: 5V
- Digital Output: 0V (Low)
- Sensing Angle: 30° Cone
- Angle of Effect: 15° Cone
- Ultrasonic Frequency: 40kHz
- Range: 2cm - 400cm

i. Procedure

1. Create a new project under the directory
   `\repos\EC\LAB\LAB_Timer_InputCaputre_Ultrasonic`

- The project name is "**LAB_Timer_InputCaputre_Ultrasonic".**

2. Create a new source file named as "**LAB_Timer_InputCaputre_Ultrasonic.c"**

- **ecGPIO.h, ecGPIO.c**
- **ecRCC.h, ecRCC.c**
- **ecTIM.h, ecTIM.c**
- **ecPWM.h, ecPWM.c**
- **ecSysTick.h, ecSysTick.c**
- **ecUART_simple.h, ecUART_simple.c**

3. Connect the HC-SR04 ultrasonic distance sensor to MCU pins(PA6 - trigger, PB6 - echo), VCC and GND

ii. Measurement of Distance

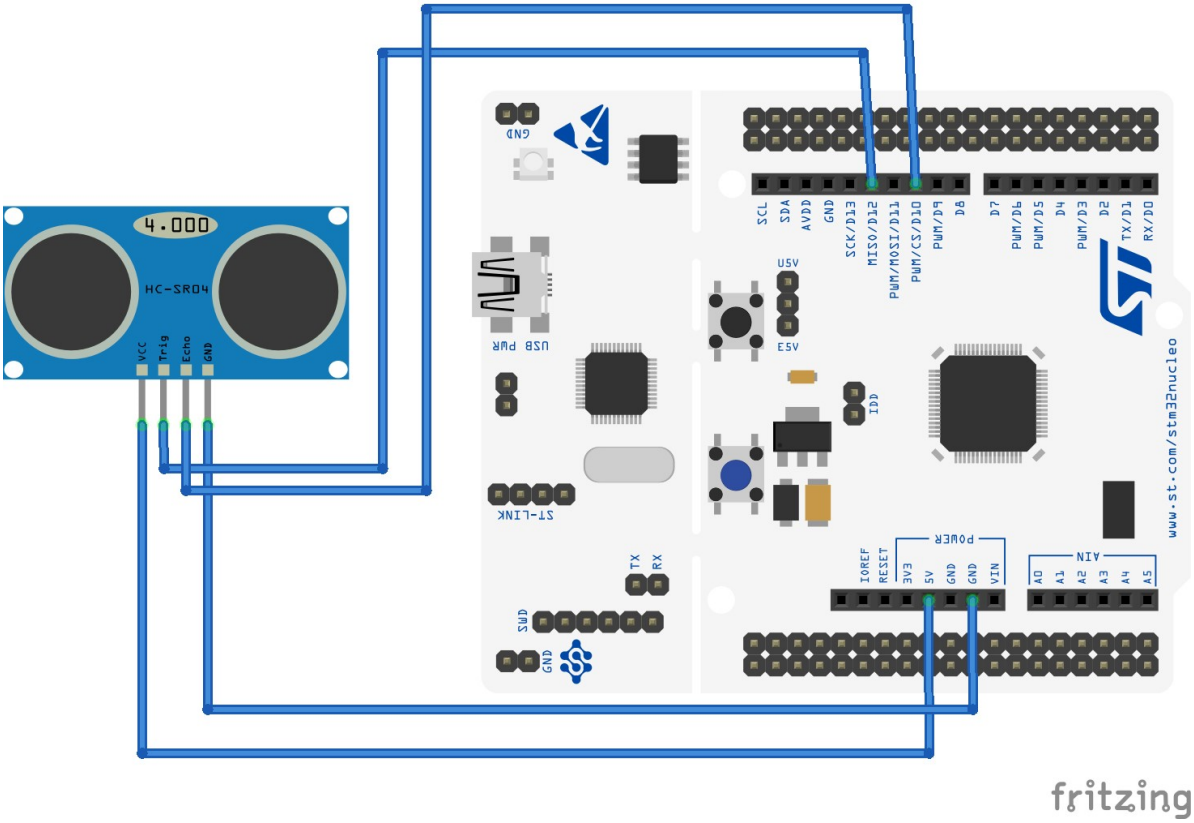
The program needs to

- Generate a trigger pulse as PWM to the sensor.
- Receive echo pulses from the ultrasonic sensor
- Measure the distance by calculating pulse-width of the echo pulse.
- Display measured distance in [cm] on serial monitor of Tera-Term for
  (a) 10mm (b) 50mm (c) 100mm

iii. Configuration

System Clock	PWM	Input Capture
PLL (84MHz)	PA6 (TIM3_CH1)	PB6 (TIM4_CH1)
	AF, Push-Pull, No Pull-up Pull-down, Fast	AF, No Pull-up Pull-down
	PWM period: 50msec pulse width: 10usec	Counter Clock : 0.1MHz (10us) TI4 -> IC1 (rising edge) TI4 -> IC2 (falling edge)

iv. Circuit Diagram



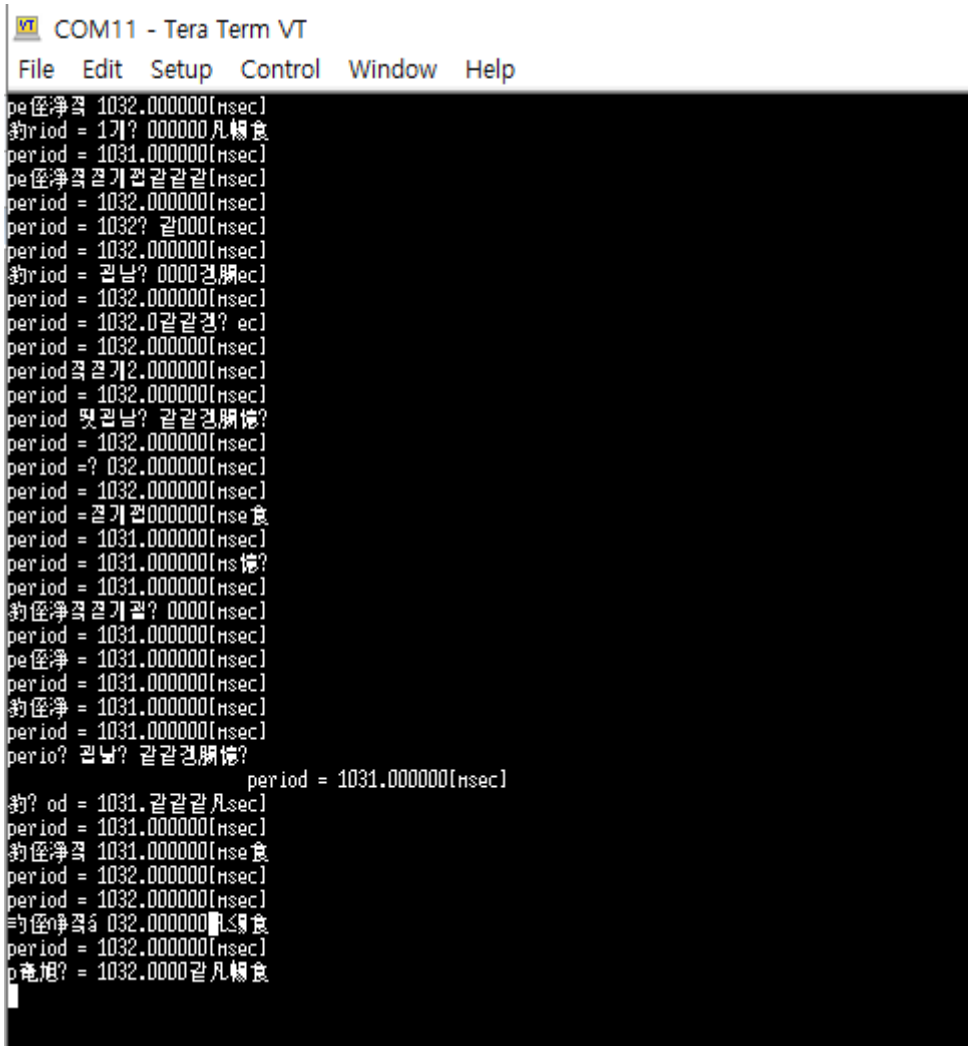
v. Discussion

1. **There can be an over-capture case, when a new capture interrupt occurs before reading the CCR value. When does it occur and how can you calculate the time span accurately between two captures?**

   If the MCU board reads and records the first Input Capture Value before processing, and then the second Input Capture Value is recorded and processed before the first one, data may be lost due to the time interval between the first and second events. This can occur when the process of storing and calculating the data received from the first event is complex and time-consuming or when the time interval between the two events is shorter than the Timer Interrupt's time interval.

   Therefore, to address this, data processing should be simplified by taking into account the two capture times, and the Timer Interrupt's time interval should be adjusted to be longer, considering the time span of the Input Capture.

2. **In the tutorial, what is the accuracy when measuring the period of 1Hz square wave? Show your result.**



$$T = \frac{1}{f} = \frac{1}{1Hz} = 1sec = 1000ms$$

$$accuracy = \frac{1031.2 - 1000}{1000} \times 100\% = 3.12\%$$

3.

vi. Code

```
#include "stm32f411xe.h"
#include "math.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecTIM.h"
#include "ecPWM.h"
#include "ecUART_simple.h"
#include "ecSysTIck.h"
#include "ecICAP.h"

uint32_t ovf_cnt = 0;
float distance = 0;
float timeInterval = 0;
float time1 = 0;
float time2 = 0;

#define TRIG PA_6
#define ECHO PB_6
```

```
void setup(void);
```

```
int main(void){

    setup();

    while(1){
        distance = (float) timeInterval * 340.0 / 2.0 / 10.0;   // [mm] -> [cm]
        printf("%f cm\r\n", distance);
        delay_ms(500);
    }
}
```

This is main code which displays distances on Tera Term in real time. It converts mm to cm by time interval. Since the speed of ultrasonic waves from the ultrasonic sensor is constant, the time difference between the output ultrasonic wave and the recognized

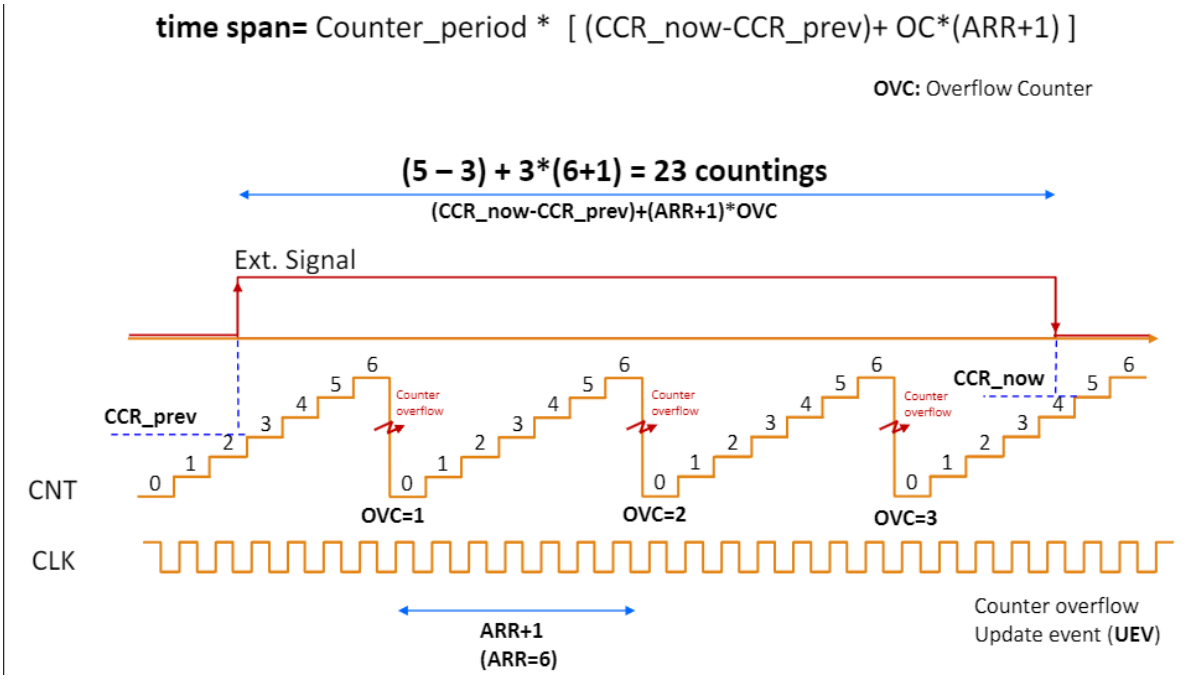
```
void TIM4_IRQHandler(void){
    if(is_UIF(TIM4)){                     // Update interrupt
        ovf_cnt++;                                              // overflow
count
        clear_UIF(TIM4);                                  // clear update
interrupt flag
    }
    if(is_CCIF(TIM4, 1)){                                 // TIM4_Ch1 (IC1)
Capture Flag. Rising Edge Detect
        time1 = TIM4->CCR1;                                  // Capture TimeStart
        clear_CCIF(TIM4, 1);                // clear capture/compare interrupt
flag
    }
    else if(is_CCIF(TIM4, 2)){                                    // TIM4_Ch2
(IC2) Capture Flag. Falling Edge Detect
        time2 = TIM4->CCR2;                                  // Capture TimeEnd
        timeInterval = ((time2 - time1) + (TIM4->ARR+1) * ovf_cnt) * 0.01;  //
(10us * counter pulse -> [msec] unit) Total time of echo pulse
        ovf_cnt = 0;                          // overflow reset
        clear_CCIF(TIM4,2);                                  // clear
capture/compare interrupt flag
    }
}
```

We can calculate the overflow count by Timer Interrupt which counting up every 1ms.

The starting captured value saves at time1 and ending captured value saves at time2.

And we can find the time span through the equation below.

Finally we convert 10 us to 1 msec.

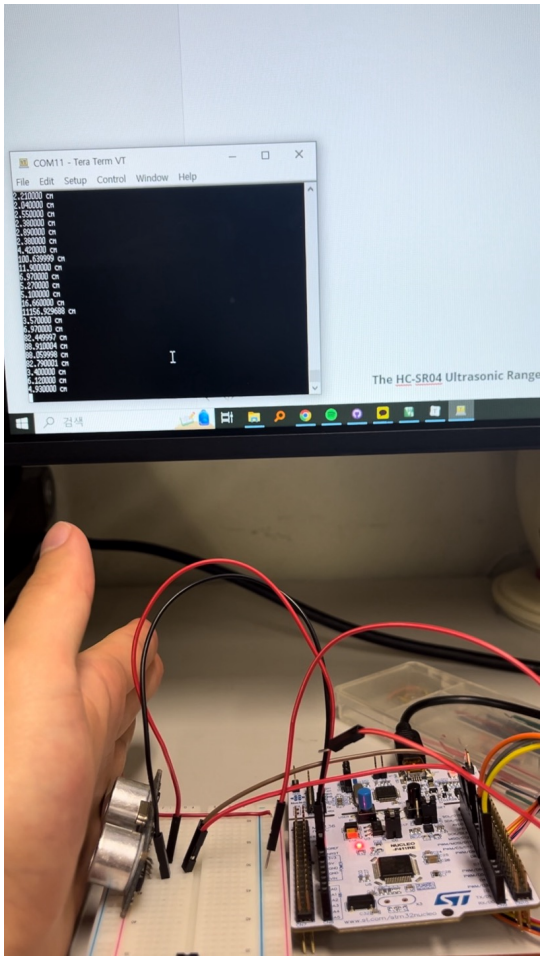


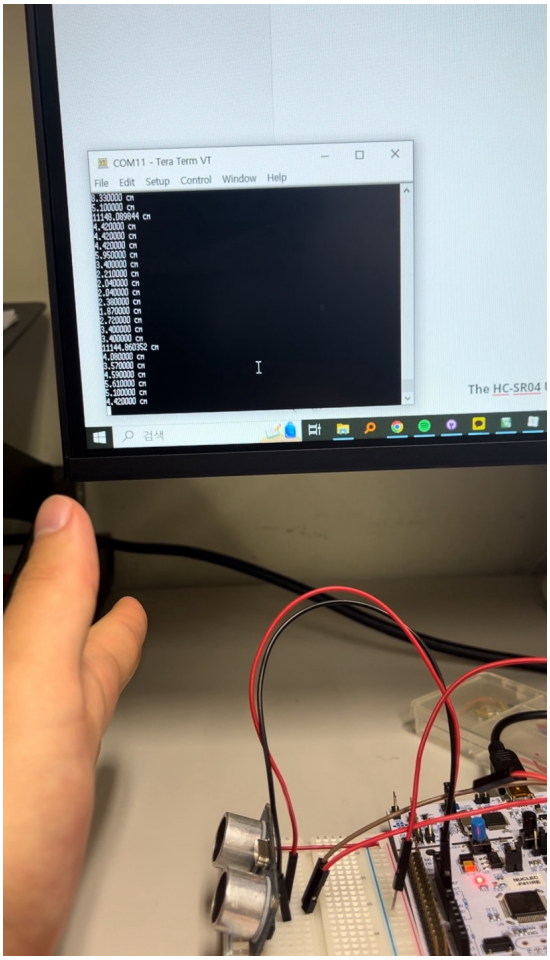

```
void setup(){

    RCC_PLL_init();
    SysTick_init();
    UART2_init();

// PWM configuration ---------------------------------------------------------
    PWM_init(TRIG);          // PA_6: Ultrasonic trig pulse
    PWM_period_us(TRIG, 50000);     // PWM of 50ms period. Use period_us()
    PWM_pulsewidth_us(TRIG, 10);    // PWM pulse width of 10us


// Input Capture configuration -----------------------------------------------------
    ICAP_init(ECHO);         // PB_6 as input caputre
    ICAP_counter_us(ECHO, 10);       // ICAP counter step time as 10us
    ICAP_setup(ECHO, 1, IC_RISE);  // TIM4_CH1 as IC1 , rising edge detect
    ICAP_setup(ECHO, 2, IC_FALL);  // TIM4_CH2 as IC2 , falling edge detect

}
```

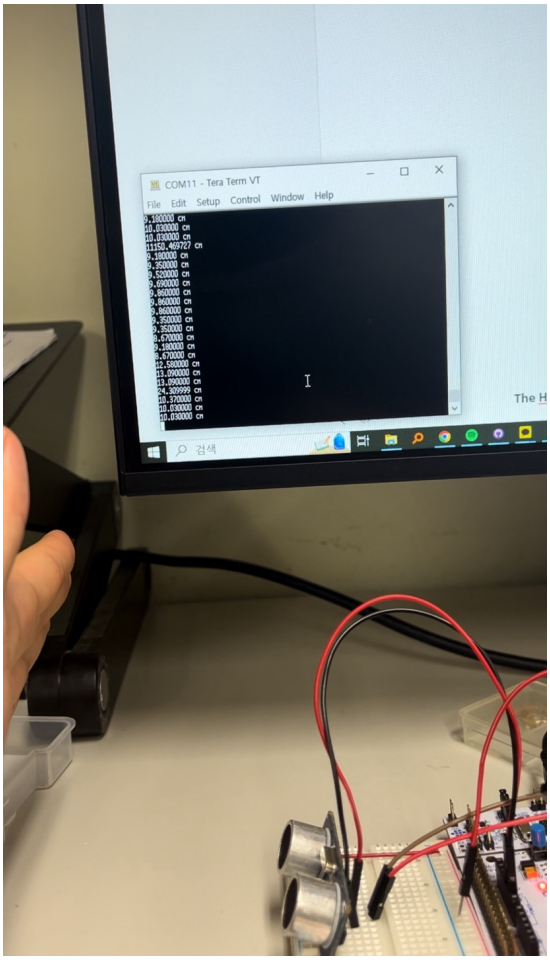
This is initializing System Clock, Systick, UART2, Trigger and Echo.

TRIG pin initialized in PWM and ECHO pin initialized in ICAP.

vii. Results


COM11 - Tera Term VT
File Edit Setup Control Window Help
The HC-SR04 Ultrasonic Range


COM11 - Tera Term VT
File Edit Setup Control Window Help
The HC-SR04



10mm was couldn't detected because this ultrasonic sensor model can detect 25cm to 400cm.

demo video link

IV. Reference

https://ykkim.gitbook.io/ec/ec-course/lab/lab-input-capture-ultrasonic

V. Troubleshooting