LAB: Input Capture - Ultrasonic

Embedded Controller Lab Report

LAB: Input Capture - Ultrasonic

Date: 2023.11.07

Author: Hanmin Kim

Github:https://github.com/ansterdaz/HKim/tree/main

Demo video: https://youtube.com/shorts/GMdHbS_3HJI

Introduction

In this lab, a simple program is required to create 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.

Requirement

Hardware

MCU

- NUCLEO-F411RE

Actuator/Sensor/Others:

- HC-SR04
- breadboard

Software

Keil uVision, CMSIS, EC_HAL library

Problem 1: Crear HAL library

Create HAL library

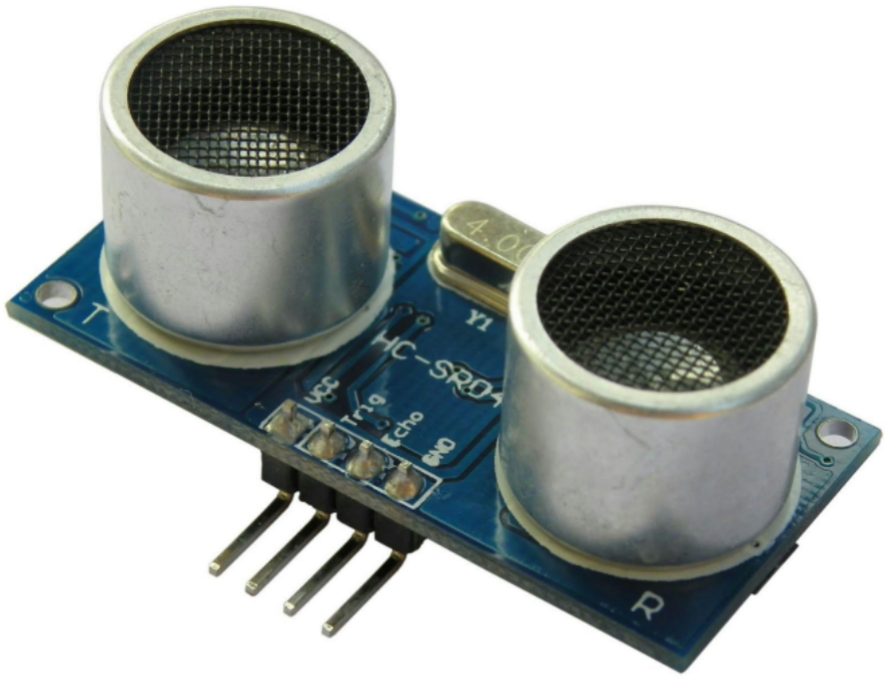
Declare and Define the following functions in your library. You must update your header files located in the directory `EC \lib\`.

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);
```

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.



HC-SR04

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

Procedure

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

- The project name is "**LAB_Timer_InputCaputre_Ultrasonic".**
- Create a new source file named as "**LAB_Timer_InputCaputre_Ultrasonic.c"**

2. Include your updated library in `\repos\EC\lib\` to your project.

- **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

Measurement of Distance

The program needs to

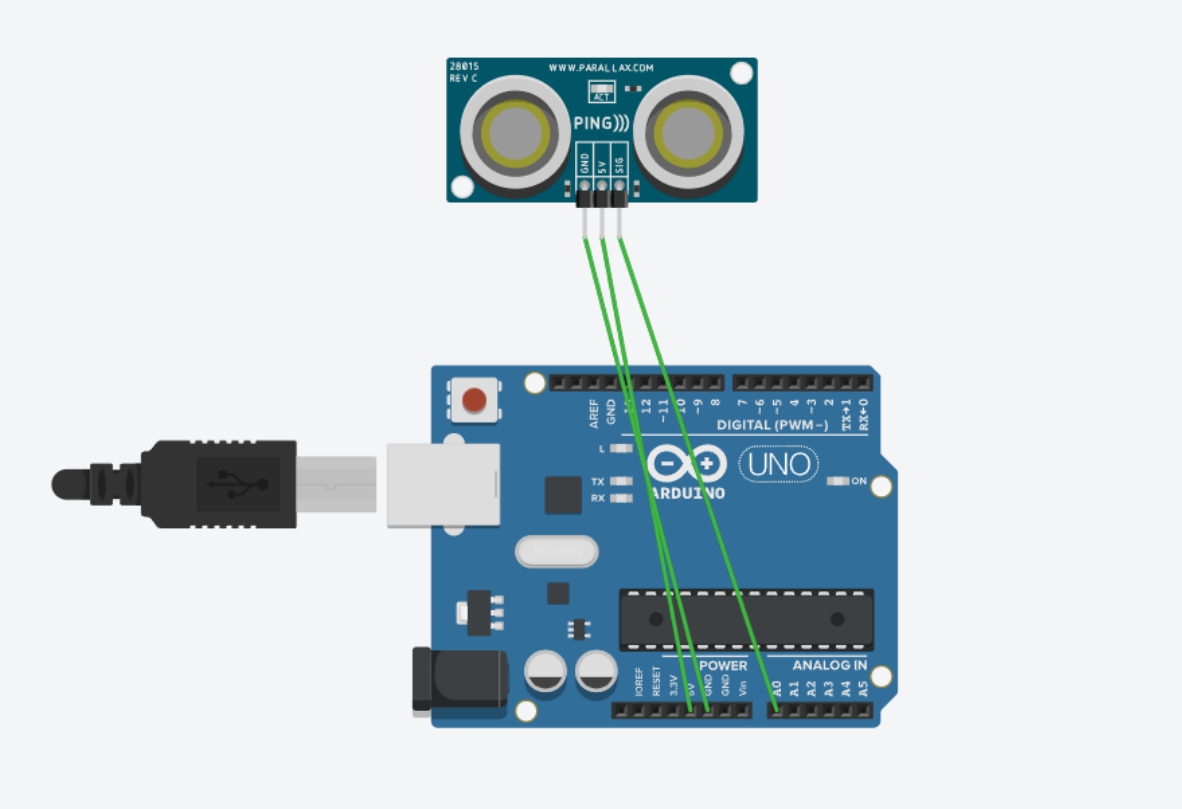
- Generate a trigger pulse as PWM to the sensor.
- Receive echo pulses from the ultrasonic sensor
- Measure the distance by calculating pulse-width of the echo pulse.
- Display measured distance in [cm] on serial monitor of Tera-Term for

  (a) 10mm (b) 50mm (c) 100mm

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)

Circuit Diagram



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?

   An overcapture case may occur when the timeinterval is not defined accurately. The time interval can be calculated through the difference between the CCR values received from the other two channels.

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

Code

```
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);
  }
}

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  ,ARR
    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    ,ARR
    timeInterval = (time2-time1 + ovf_cnt*((TIM4->ARR)+1))/100;   // (10us * counter pulse -> [msec] unit) Total time of echo pulse
    ovf_cnt = 0;                        // overflow reset
    clear_CCIF(TIM4,2);                 // clear capture/compare interrupt flag
  }
}
```

Wrote a code that calculates the time interval by receiving a signal going out at the CCR value of channel 1 and returning a signal at the CCR value of channel 2.

Result

Through Teratum, confirmed that the distance is measured when placing a hand on the sensor.

Video Link: https://youtube.com/shorts/GMdHbS_3HJI

Reference

https://github.com/ykkimhgu

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

Appendix

Main Code

```c
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);
  }
}

  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.
      time1 =  TIM4->CCR1;                  // Capture TimeStart  ,ARR
      clear_CCIF(TIM4, 1);                  // clear capture/compare interru
    }

    else if(is_CCIF(TIM4, 2)){                  // TIM4_Ch2 (IC2) Capture
      time2 = TIM4->CCR2;                   // Capture TimeEnd    ,ARR
      timeInterval = (time2-time1 + ovf_cnt*((TIM4->ARR)+1))/100;   // (1(
      ovf_cnt = 0;                          // overflow reset
      clear_CCIF(TIM4,2);                   // clear capture/compare interru
    }
  }

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

}
```