Sensor Ultrassônico HC-SR04

1 - Descrição do Sensor

O sensor HC-SR04 é muito popular e comumente usado em aplicações simples de automação. Tem uma faixa de alcance de 2 cm a 400 cm e é interessante para diversas aplicações simples.

Figura 1 - Sensor HC-SR04

O sensor ultrassônico estará em modo inativo enquanto não houver sinal no seu pino de Trigger. O sensor começa a operar quando um um pulso, que deve ter duração maior ou igual a 2us, é aplicado no pino Trig. Depois de ativado o sensor envia 8 pulsos de onda sonora em uma frequência ultrassônica de 40kHz. O som viaja em linha reta até atingir um objeto e então reflete de volta ao módulo. O módulo por sua vez envia um pulso digital no pino de Echo que tem uma largura igual ao tempo de viagem do som indo e voltando entre o módulo e o objeto detectado.

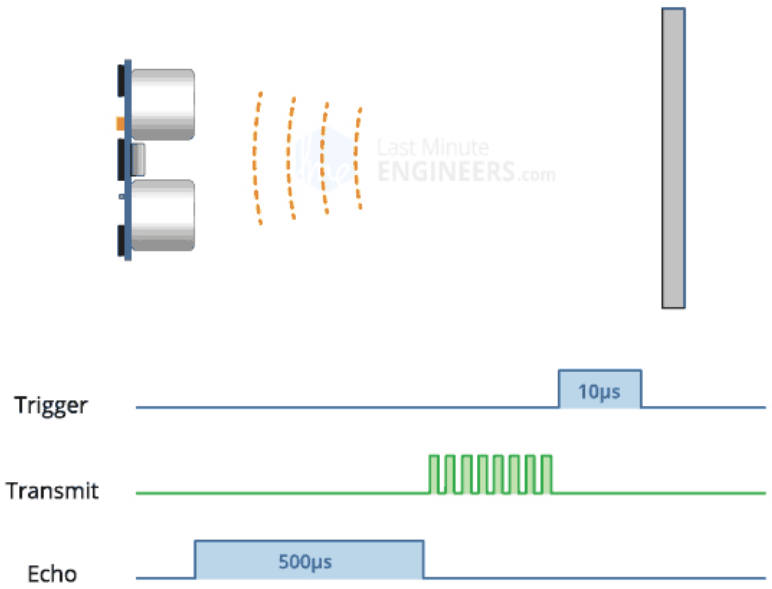


Figura 2 - Sinais no sensor

Para calcular a distância entre o sensor e o objeto de interesse basta medir a largura do pulso de echo gerado no sensor e dividi-lo por 2 (pois o tempo é de ida e volta) e multiplicar o valor pela velocidade de propagação do som no ar que é constante e igual a 340 m/s. Ou seja:

$$d = (V_{som} * T_{echo})/2$$

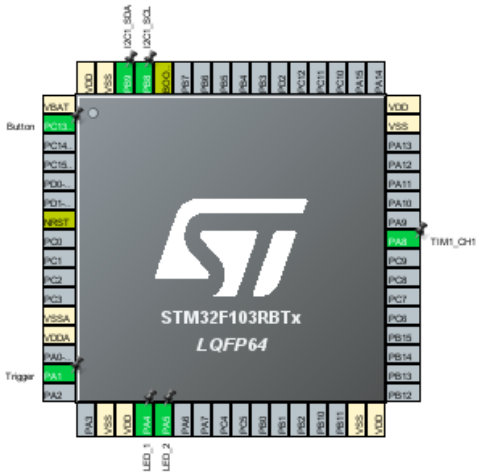
2 - Estratégia de implementação

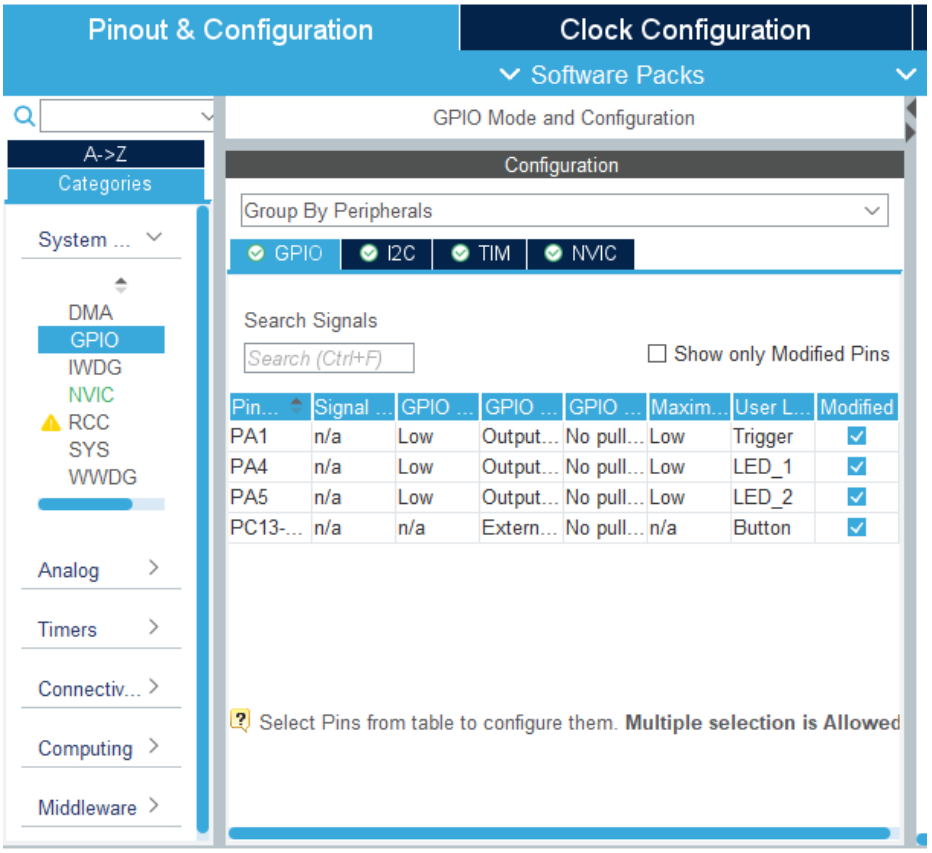
Para desenvolver a API para este sensor, a estratégia adotada foi a de utilizar a unidade de captura de entrada (ICU) dos módulos temporizadores para capturar o tempo em que o pulso de echo vai para nível alto e nível baixo e calcular a diferença entre os dois tempos. O código abaixo descreve o comportamento mencionado:

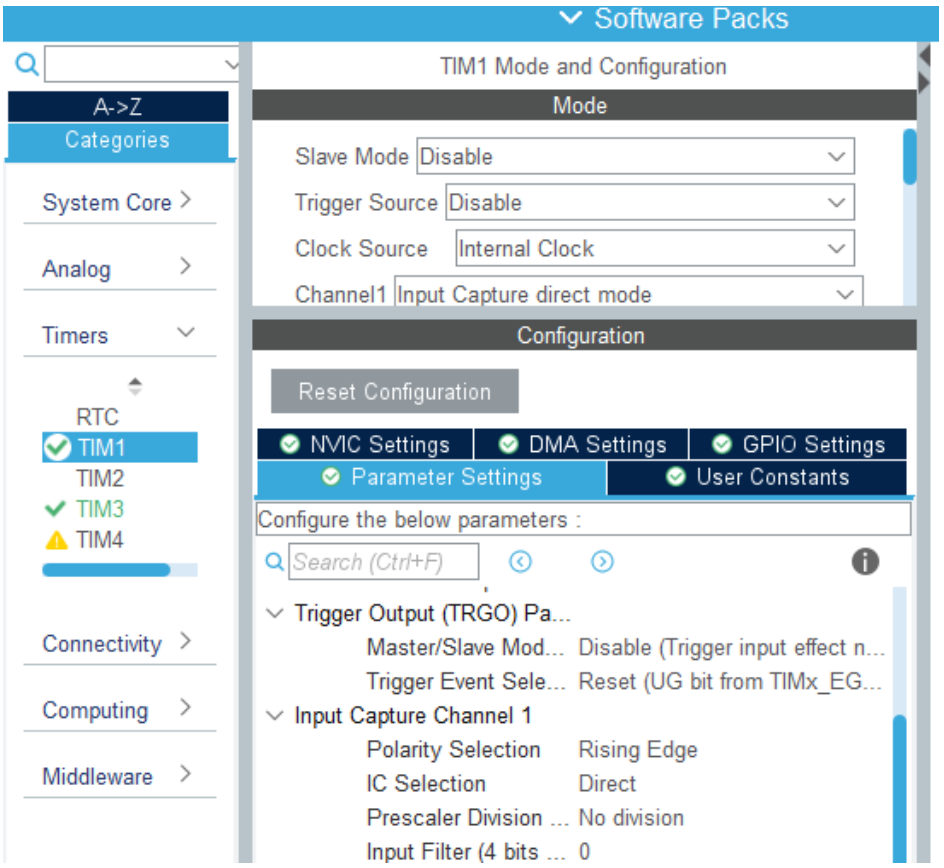
```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim) //Esta função
serve para capturar o tempo de que Echo fica em nível lógico alto
{
    if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)
    {
        if (Primeira_Captura==0) //Se a variavel for igual a zero
significa que o sinal subiu para nivel lógico alto
        {
            Time1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1); //O
momento de subida é armazenado na variavel time1
            Primeira_Captura = 1;


            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1,
TIM_INPUTCHANNELPOLARITY_FALLING); //Muda a configuração para ler o
momento em que o sinal esteja descendo
        }

        else if (Primeira_Captura==1) ////Se a variavel for igual a um
significa que o sinal desceu para nivel lógico baixo
        {
            Time2 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);//
//O momento de descida é armazenado na variavel time1
            __HAL_TIM_SET_COUNTER(htim, 0); //Reinicia o contador do
timer1 para zero

            if (Time2 > Time1)// Se time2 for maior que time1 o
resultado do tempo é igual a diferenca
            {
                Diferenca = Time2-Time1;
            }
```

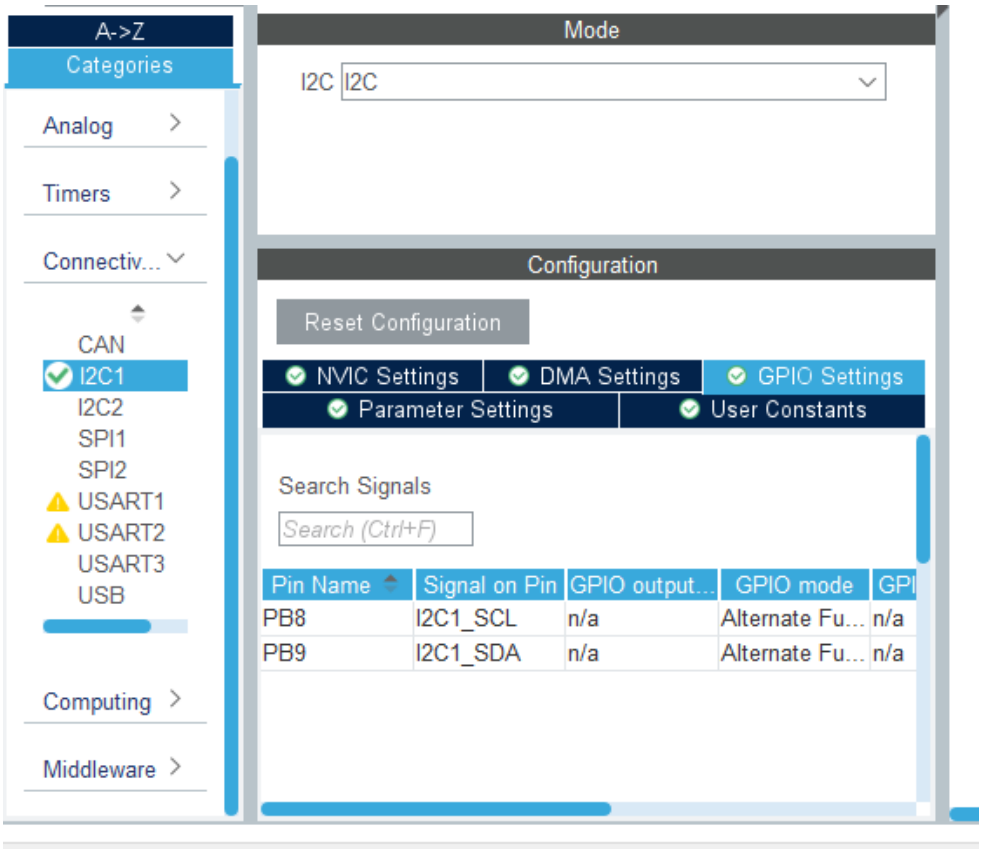
```
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1,
TIM_INPUTCHANNELPOLARITY_RISING); //Muda a configuração para ler o
momento em que o sinal esteja subindo
            __HAL_TIM_DISABLE_IT(&htim1, TIM_IT_CC1); //A interrupção é
desabilitada para ser chamada apenas quando necessário
        }
    }
}
```

3 - Configurações do ambiente

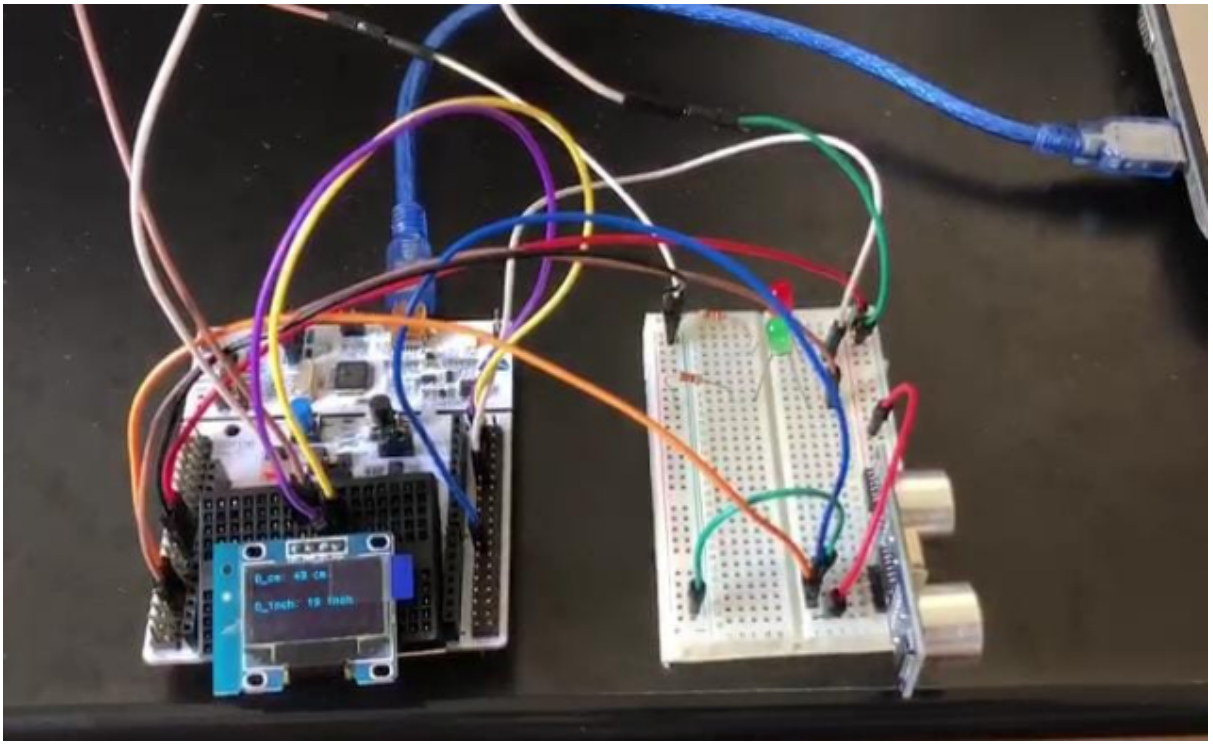





Software Packs
TIM1 Mode and Configuration
A->Z
Categories
System Core
Analog
Timers
RTC
TIM1
TIM2
TIM3
TIM4
Connectivity
Computing
Middleware
Mode
Slave Mode Disable
Trigger Source Disable
Clock Source Internal Clock
Channel1 Input Capture direct mode
Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
Trigger Output (TRGO) Pa...
Master/Slave Mod... Disable (Trigger input effect n...
Trigger Event Sele... Reset (UG bit from TIMx_EG...
Input Capture Channel 1
Polarity Selection Rising Edge
IC Selection Direct
Prescaler Division ... No division
Input Filter (4 bits ... 0


A->Z
Categories
Analog
Timers
Connectiv...
CAN
I2C1
I2C2
SPI1
SPI2
USART1
USART2
USART3
USB
Computing
Middleware
Mode
I2C I2C
Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Search Signals
Search (Ctrl+F)
Pin Name
Signal on Pin
GPIO output...
GPIO mode
GPI
PB8 I2C1_SCL n/a Alternate Fu... n/a
PB9 I2C1_SDA n/a Alternate Fu... n/a

4 - Conexões



LED1 -> PA4

LED2 -> PA5

Trigger -> PA1

Echo -> PA8

VCC -> 3.3V

GND -> VSS

OLED

SCL -> PB8

SDA -> PB9

Repositório: https://github.com/cristovaoeustaquio/API_Sensor_Ultrasonico_HCSR04

5 - Referências

[1] UM1850 - Description of STM32F1 HAL and Low-layer drivers

[2] UM2609 - STM32CubeIDE user guide

[3] https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/