[INF1805] Sistemas Reativos

PUC
RIO

Luis Claudio C. Martins
Lucas P Nepomuceno

Objetivo

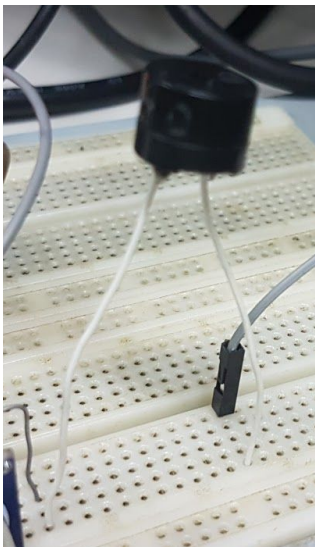
Implementar um sistema de sensor de proximidade que indique a distância e avise o usuário. O objetivo é criar uma prova de conceito para sistemas de automação doméstica (sensor de proximidade para janelas e portas)

O codigo consiste em 3 partes.

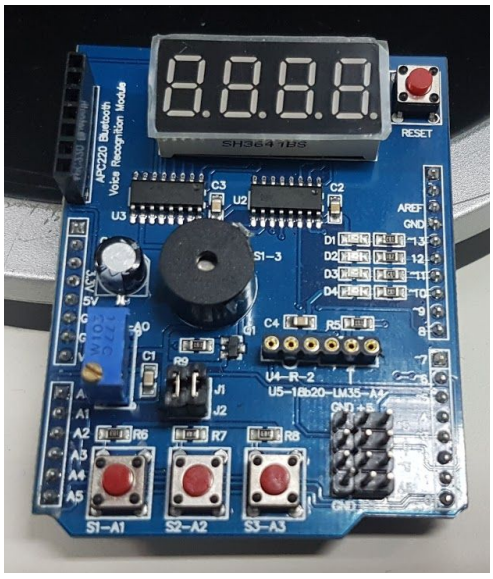
O loop principal recebe a distancia do sensor e armazena.

Após obtido esse valor, uma função mostra ela no display de 7 segmentos. O fabricante garante precisão entre 4cm e 8m, em milimetros, 40mm a 8000mm. Logo, decidimos ignorar qualquer valor acima de 9999 (5 digitos).

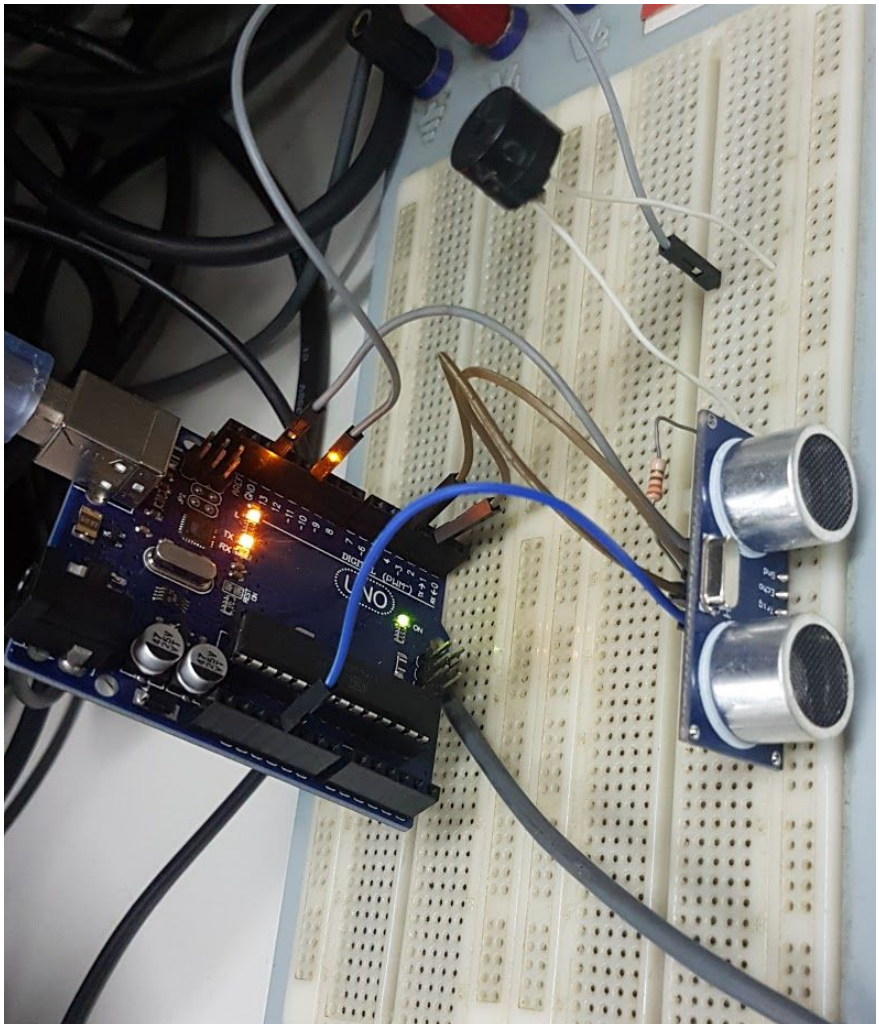
A ultima parte toca um tom no buzzer definido pela variavel melodia, caso a distancia seja menor do que 1000mm.



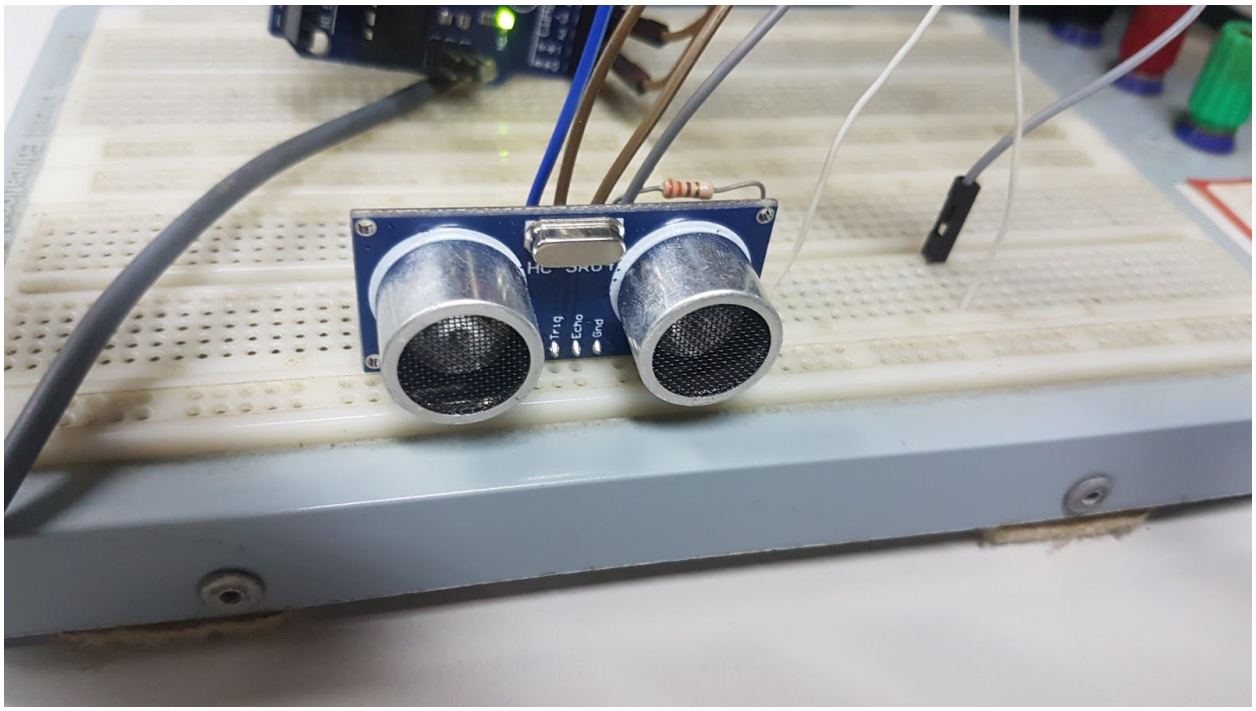
Buzzer



Display



Setup



Sensor

Sensor ultrassônico HC-SR04

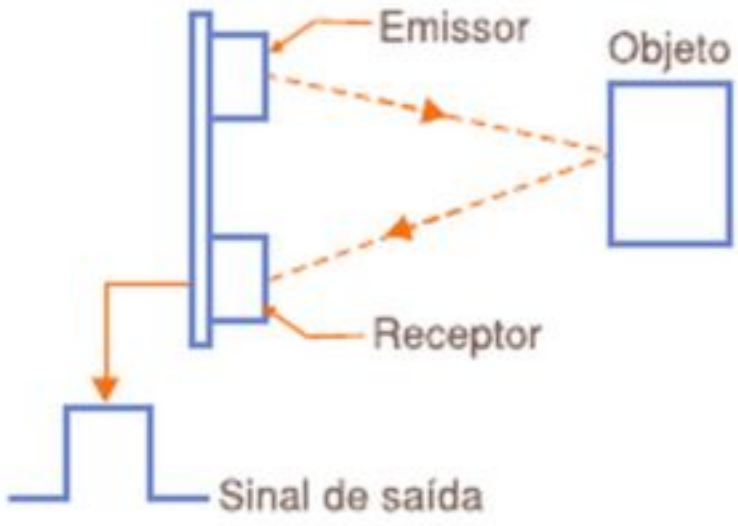
O sensor é capaz de medir distâncias de 2 cm a 4 m com ótima precisão. Este módulo possui 4 pinos (VCC, TRIGGER, ECHO, GND).

Pinos sensor HC-SR04

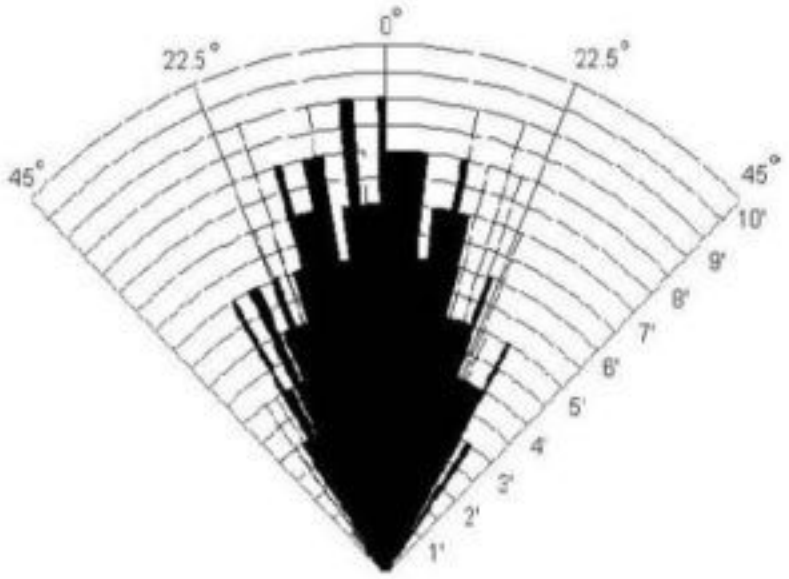
Os pinos TRIGGER e ECHO medem a distância de determinado objeto da seguinte forma: o pino TRIGGER emite ondas sonoras que, ao se depararem com um obstáculo retornam em direção ao sensor. O pino ECHO recebe essas ondas rebatidas e calcula a distância a partir do tempo de retorno.

Essa é a fórmula básica para cálculo de distância: DISTÂNCIA = [Tempo obtido por ECHO * Velocidade do Som] / 2

A velocidade do som poder ser considerada igual a 340 m/s, logo o resultado é obtido em metros.



Pinos Trigger e Echo.



Practical test of performance,
Best in 30 degree angle

Código

```
#include <hcsr04.h>

const byte SEGMENT_MAP[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90};
//BYTE MAP PARA NUMEROS DE 0 A 9
const byte SEGMENT_SELECT[] = {0xF1,0xF2,0xF4,0xF8};                    //BYTE
MAP PARA SELECIONAR O DISPLAY

#define TRIG_PIN 4
#define ECHO_PIN 2
#define buzzer 11
HCSR04 hcsr04(TRIG_PIN, ECHO_PIN);//, 20, 4000);
//Ultrasonic ultrasonic(TRIG_PIN, ECHO_PIN);
unsigned short distance;

#define  c     3830    // 261 Hz
#define  d     3400    // 294 Hz
#define  e     3038    // 329 Hz
#define  f     2864    // 349 Hz
#define  g     2550    // 392 Hz
#define  a     2272    // 440 Hz
#define  b     2028    // 493 Hz
#define  C     1912    // 523 Hz
#define  R     0

/*RELOGIO*/
#define LED1       10
#define LED2       11
#define LED3       12
#define LED4       13
#define BUZZ        3
#define KEY1       A1
#define KEY2       A2
#define KEY3       A3
#define POT        A0
#define LATCH_DIO  4
#define CLK_DIO    7
#define DATA_DIO   8
/*RELOGIO*/


int speakerOut = 11;
// Do we want debugging on serial out? 1 for yes, 0 for no
int DEBUG = 1;


float seno;
int frequencia;

void setup(){
  if (DEBUG) {
```

```
  Serial.begin(9600); // Set serial out if we want debugging
 }
 //Serial.begin(9600);
 pinMode(buzzer, OUTPUT);
}

// MELODY and TIMING  =======================================
//  melody[] is an array of notes, accompanied by beats[],
//  which sets each note's relative length (higher #, longer note)
//int melody[] = {  a,  a,  a,  f,  C,   a,  f,  C,  a,  R, R, R };
int melody[] = {  a,  R,  a,  R,  a,  R,  f,  C,  a,  R,  f,  C,  a,  R};
int beats[]  = {  16, 2,  16, 2,  16, 2,  32, 12, 16, 2,  32, 12,  16, 8};
int MAX_COUNT = sizeof(melody)/2; // Melody length, for looping.
long tempo = 10000;
int pause = 1000;
int rest_count = 100;
int tone_ = 0;
int beat = 0;
long duration  = 0;

// Toca musica
void playTone() {
 long elapsed_time = 0;
 if (tone_ > 0) // Se nao for a nota de descanso
 {
  while (elapsed_time < duration) //Tempo de duração da melodia
  {

   digitalWrite(speakerOut,HIGH);//liga o buzzer
   delayMicroseconds(tone_ / 2);

   digitalWrite(speakerOut, LOW);//desliga o buzzer
   delayMicroseconds(tone_ / 2);

   // Keep track of how long we pulsed
   elapsed_time += (tone_);
  }
 }
 else { // Rest beat; loop times delay
  for (int j = 0; j < rest_count; j++) { // See NOTE on rest_count
   delayMicroseconds(duration);
  }
 }
}


void EscreveNumeroNoDisplay(byte Segment, byte Value)
{
 /*FUNCAO RESPONSAVEL PELA MULTIPLEXAÇÃO DOS DISPLAYS*/
 digitalWrite(LATCH_DIO,LOW);
 shiftOut(DATA_DIO, CLK_DIO, MSBFIRST, SEGMENT_MAP[Value]);
 shiftOut(DATA_DIO, CLK_DIO, MSBFIRST, SEGMENT_SELECT[Segment] );
```

```
  digitalWrite(LATCH_DIO,HIGH);
}

void EscreveNumero(unsigned int primeiro,unsigned int segundo,unsigned int terceiro,unsigned
int quarto)
{
  EscreveNumeroNoDisplay(0,primeiro);
  EscreveNumeroNoDisplay(1,segundo);
  EscreveNumeroNoDisplay(2,terceiro);
  EscreveNumeroNoDisplay(3,quarto);
}

void loop()
{
  distance = hcsr04.distanceInMillimeters();
  /*//Serial.println(hcsr04.ToString());
  Serial.println(distance);
  frequencia = 1000;
  if(distance < 100)
  {
    tone(buzzer,frequencia,200);
  }*/
  //------------------------------------------------
  // Set up a counter to pull from melody[] and beats[]
  for (int i=0; i<MAX_COUNT; i++)
  {
    unsigned short estado = 0;
    distance = hcsr04.distanceInMillimeters();
    Serial.println("--------------------");
    Serial.println(distance);
    unsigned short aux = distance;
    if (aux>9999)
      aux = 0;
    unsigned int primeiro = aux/1000;
    aux -= primeiro*1000;
    unsigned int segundo = aux/100;
    aux -= segundo*100;
    unsigned int terceiro = aux/10;
    aux -= terceiro*10;
    unsigned int quarto = aux;

    Serial.print("primeiro = ");
    Serial.println(primeiro);
    Serial.print("segundo = ");
    Serial.println(segundo);
    Serial.print("terceiro = ");
    Serial.println(terceiro);
    Serial.print("quarto = ");
    Serial.println(quarto);
    Serial.println("--------------------");
    EscreveNumero(primeiro,segundo,terceiro,quarto);
```

```
    frequencia = 1000 * distance;
    tone_ = melody[i];
    beat = beats[i];

    duration = beat * tempo; // Set up timing

    if(distance < 100)
    {
      /*estado = digitalRead(KEY1);
      if(!estado)
      {
          playTone();
      }
      else
      {
        tone(buzzer,frequencia,200);
      }*/
      playTone();
      //tone(buzzer,frequencia,200);
    }
    //playTone();
    // A pause between notes...
    //delayMicroseconds(pause/3);
  }
  delay(100);
}
```

Dificuldades

Encontrar uma biblioteca que se adequa-se ao trabalho proposto

Montagem do equipamento para a utilização

Ambiente de Desenvolvimento, tanto físico como a interface