

Robotic

ARDUINO LAB

ARDUINO PARA INICIANTES

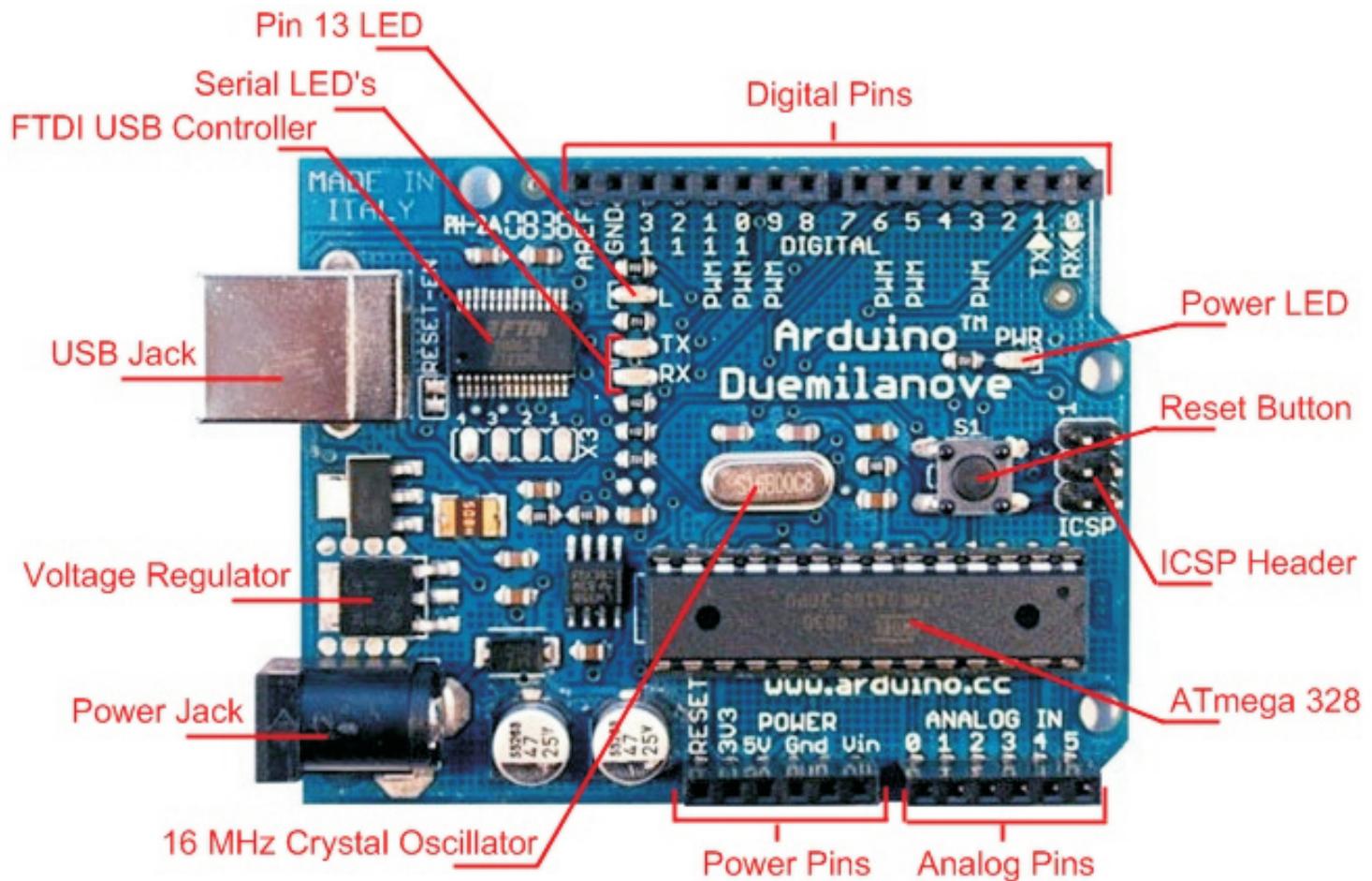
WWW.ROBOTIC.COM.BR

Sumário

LED Piscante	01
Desvanecimento de um LED	02
Buzzer - Melodia	03
Controlando um Servo pelo Serial Monitor	04
Sensor de Luz com LDR - LED	05
Sensor de Luz com LDR - Servo	06
Chave Táctil - Acende um LED	07
Chave Táctil - Servo Motor	08
Potenciômetro - LED	09
Potenciômetro - Servo	10
RTC - Real Time Clock - Serial Monitor	11
RTC - Real Time Clock - Display LCD	12
RTC - Real Time Clock - LED	13
RTC - Real Time Clock - Servo	14
Controle Remoto Códigos - Serial Monitor	15
Controle Remoto - LED	16
Controle Remoto - Servo	17
Controle Remoto - Relé	18
Sensor de Temperatura - Serial Monitor	19
Sensor de Temperatura - Display LCD	20
Sensor de Temperatura - LED	21
Sensor de Temperatura - Servo	22
Sensor de Ultrasom - Serial Monitor	23
Sensor de Ultrasom - Display LCD	24
Sensor de Ultrasom - Servo	25
Alarme de Incêndio com DS18B20	26
Alarme de Intruso com Ultrasom	27
Detector de obstáculos com Infrared	28
Controlando a UMR remotamente pelo notebook	29

Arduino Duemilanove Atmega 328

O Arduino é uma placa simples que contém tudo o que você precisa para começar a trabalhar com eletrônica e programação de microcontroladores. Este diagrama ilustra os principais componentes da placa



Arduino Duemilanove Atmega328

O Arduino Duemilanove é uma placa baseada no microcontrolador Atmega328. "Duemilanove" significa em italiano 2009 foi nomeado após o ano de seu lançamento. O Duemilanove é o último de uma série de placas Arduino USB. Tem 14 pinos de entrada e saída digital (dos quais 6 podem ser utilizados como saídas PWM), 6 entradas analógicas. (Com a biblioteca Mega Servos pode se ligar servos em qualquer saída digital).

Resumo

Microcontrolador ATmega328

Tensão 5V

Tensão de entrada (recomendado) 7-12V

Tensão de entrada (limites) 6-20V

Digital pins de I / O 14 (dos quais 6 oferecem saída PWM)

Analog Input Pins 6

DC atual por I / O Pin 40 mA

Corrente DC 3.3V para Pin 50 mA

Memória Flash 32 KB de que 2 KB utilizado pelo bootloader SRAM 2 KB

EEPROM 1 KB

A velocidade do clock 16 MHz

O Arduino Duemilanove pode ser alimentado através da conexão USB ou com fonte de alimentação externa, adaptador AC para DC ou bateria (7-12v). O plug do adaptador 2,1 mm com centro-positivo. A fonte de energia é selecionado automaticamente.

Os pinos de alimentação são os seguintes:

VIN. A tensão de entrada para a placa Arduino, quando se está usando uma fonte externa de energia (ao contrário de 5 volts a partir da conexão USB). Você pode fornecer a tensão através desse pino.

5V. A fonte de alimentação regulada usada para alimentar o microcontrolador e outros componentes na placa. Isto pode vir tanto de VIN através de um regulador de bordo, ou seja alimentado por USB ou outra fonte de 5V regulados.

3V3. é gerada pelo chip FTDI-on. Máximo consumo de corrente é de 50 mA.

Ground pinos GND..

Memória

O ATmega168 tem 16 KB de memória flash para armazenar o código (de 2 KB que é usado para o gerenciador de inicialização).

ATmega328 tem 32 KB, (também com 2 KB utilizado para o bootloader). O ATmega168 tem 1 KB de SRAM e 512 bytes de EEPROM (que pode ser lido e escrito com a biblioteca EEPROM), o ATmega328 tem 2 KB de SRAM e 1 KB de EEPROM.

Entrada e Saída

Cada um dos 14 pinos digital no Duemilanove pode ser usado como uma entrada ou saída, usando pinMode () , digitalWrite () , e digitalRead () funções. Eles operam em 5 volts. Cada pino pode fornecer ou receber um máximo de 40 mA e tem um resistor pull-up interno (desligado por padrão) de 20-50 kOhms.

Pinos com funções específicas:

Serial: 0 (RX) e 1 (TX). Usado para receber (RX) e transmitir (TX TTL série de dados). Esses pinos são ligados aos pinos correspondentes do chip FTDI USB-to-TTL Serial.

Interrupções externas: 2 e 3. Estes pinos podem ser configurados para disparar uma interrupção em um valor baixo, um aumento ou diminuição de ponta, ou uma alteração no valor. (Veja a função attachInterrupt)

PWM: 3, 5, 6, 9, 10 e 11. Pinos de saída PWM

LED: 13. Há um built-in LED conectado ao pino digital 13.

Quando o pino está em valor alto , o LED está ligado, quando o pino é baixo, ele está desligado

O Duemilanove tem 6 entradas analógicas, cada uma das quais fornecem 10 bits de resolução (ou seja, 1.024 valores diferentes). Por padrão, elas medem do GND para 5 volts, mas é possível mudar a extremidade superior de sua escala com o pino AREF e analogReference () função.

AREF. Tensão de referência para as entradas analógicas. Usado com analogReference () .

Reset. Traga esta linha baixa para redefinir o microcontrolador. Tipicamente usado para adicionar um botão de reset de escudos.

Comunicação

O Duemilanove tem uma série de facilidades para se comunicar com um computador, outro Arduino, ou outros microcontroladores. O ATmega168 e ATmega328 fornecer UART TTL (5V), comunicação serial, que está disponível nos pinos digitais 0 (RX) e 1 (TX). Um FT232RL FTDI os canais de comunicação serial de USB e os drivers FTDI (incluído com o software Arduino) oferecem uma porta virtual com o software no computador. O software inclui um monitor Arduino serial que permite que dados simples de texto a ser enviado e da placa Arduino. O RX e TX LEDs na placa pisca quando dados estão sendo transmitidos através do chip FTDI e conexão USB para o computador (mas não para comunicação serial nos pinos 0 e 1).

A biblioteca Software Serial permite a comunicação serial em qualquer um dos pinos digitais do Duemilanove o. O apoio também ATmega328 I2C (TWI) e comunicação SPI.

O software inclui uma biblioteca Arduino Wire para simplificar o uso do barramento I2C, veja a documentação para mais detalhes. Para usar a comunicação SPI, consulte o datasheet ou ATmega328

Programação

O Duemilanove pode ser programado com o software IDE Arduino. Selecione "Arduino Duemilanove Atmega328" no menu Ferramentas (de acordo com o microcontrolador na placa).

O Arduino Duemilanove ATmega328 vem com um gerenciador (BOOTLOADER) que permite que você faça o upload do novo código para ele sem a utilização de um programador de hardware externo.

Ele se comunica utilizando o protocolo STK500 original (de referência , arquivos de cabeçalho C). Você também pode ignorar o bootloader e programar o microcontrolador através do ICSP (In-Circuit Serial Programming),

O Arduino Duemilanove tem um polyfuse resettable que protege as portas USB do seu computador a partir de shorts e sobrecorrente. Embora a maioria dos computadores fornecem sua própria proteção

Instalando o Programa

1 Faça o download do ambiente de desenvolvimento do Arduino em: www.robotic.com.br

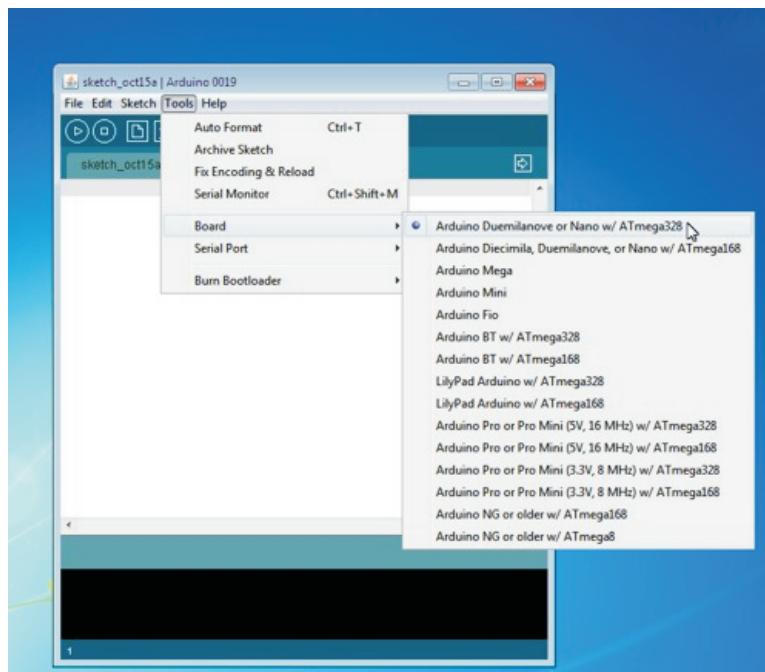
Para programar uma placa Arduino você precisa do ambiente de desenvolvimento.

Quando o download terminar descompacte (unzip) os arquivos. Certifique-se de preservar a estrutura das pastas. Abra a pasta. Dentro você vai encontrar alguns arquivos e sub-pastas. O aplicativo “Arduino” é seu programa ambiente de desenvolvimento.

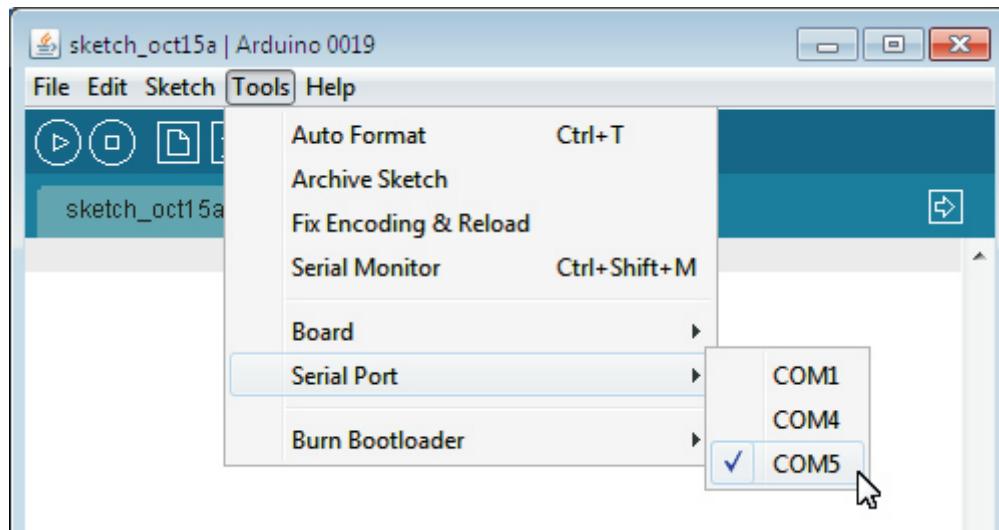
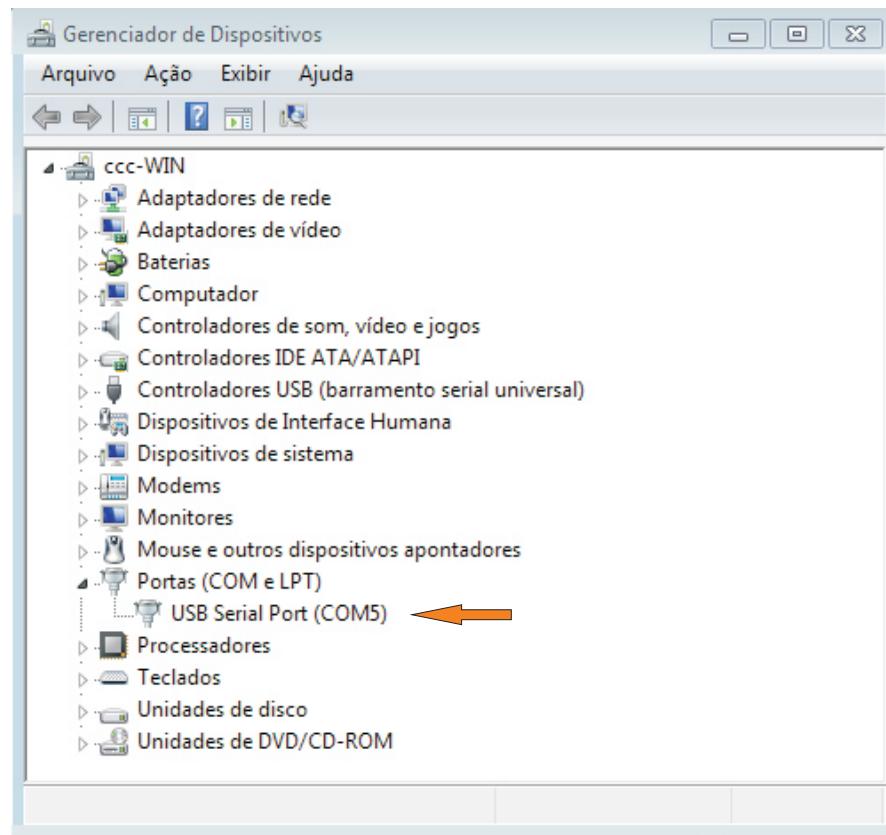
2 Ao conectar a Placa Arduino na Porta USB e caso for solicitado drives localize os drivers USB para o chip FTDI da placa na pasta drivers/FTDI USB Drivers da distribuição do Arduino.

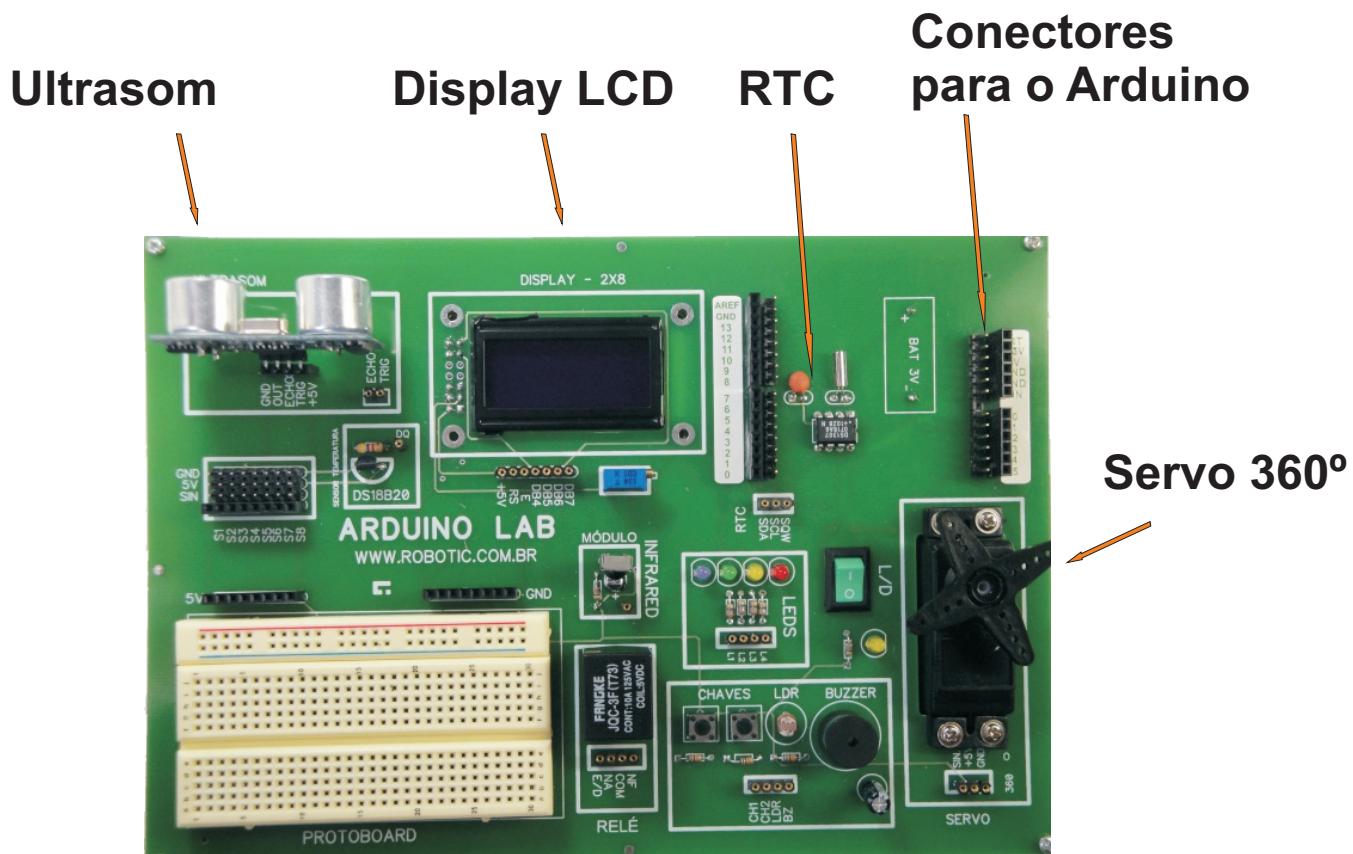
Configurando:

Certifique-se que o "Arduino Duemilanove Atmega 328" está selecionado em: **Tools > Board**



Escolhendo a Porta Serial Selecione a porta serial menu **Tools | Serial Port**. No Windows deveria ser: COM3, COM4 ou COM5 para uma placa USB. Para descobrir qual é abra o Gerenciador de Dispositivos do Windows (na aba Hardware do painel de controle do sistema). Procure por "USB Serial Port" na seção de Portas;





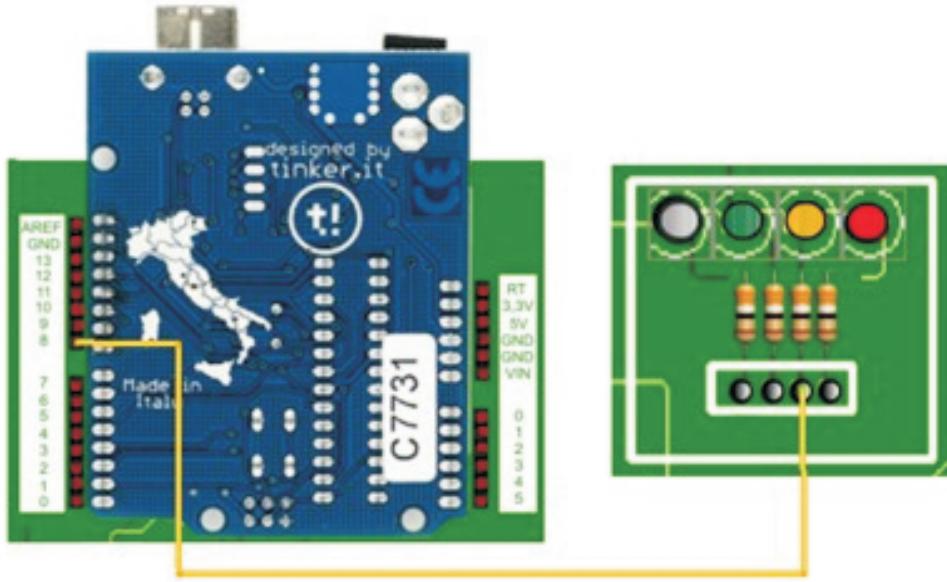
OBS: Para os experimentos deste manual serão necessários as seguintes bibliotecas:

- DallasTemperature
- MegaServo
- LiquidCrystal
- Wire
- Irremote
- Ultrasonic

Verifique no diretório onde foi instalado a IDE do arduino se na pasta LIBRARIES tem estas bibliotecas, caso falte alguma, baixe da pagina de download no site: www.robotic.com.br e descompacte na pasta libraries.

Experimento 1: Led piscante

Faca as ligações conforme abaixo



Conectando os Jumpers

D8 - Pino Digital 8 ligado ao LED

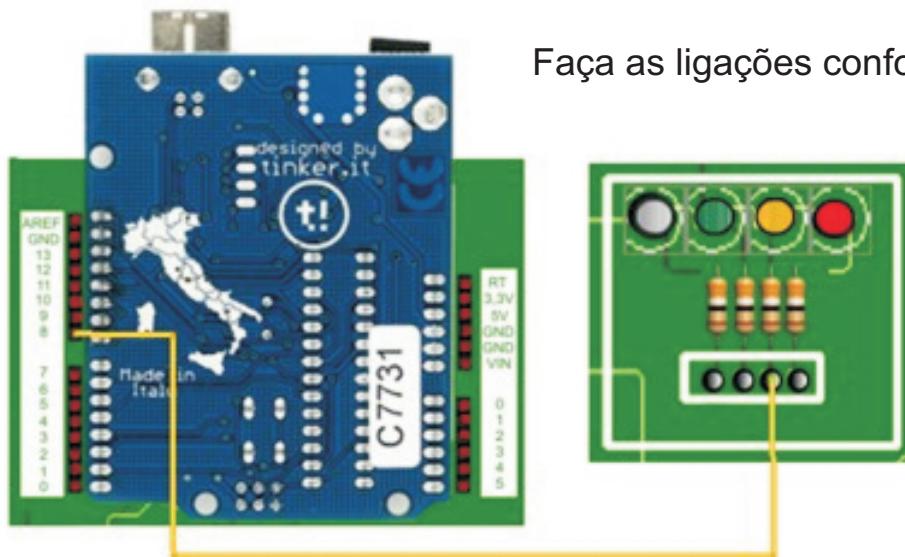
Copie e cole este programa na IDE do arduino

```
/* LED piscante
 *
 * Liga e Desliga um LED conectado a um pino digital 8
 * em intervalos de 1 segundos.
 */
int ledPin = 8;           // LED conectado ao pino digital 8
void setup()
{
    pinMode(ledPin, OUTPUT); // configura pino digital como saída
}
void loop()
{
    digitalWrite(ledPin, HIGH); // liga o LED
    delay(1000);             // temporiza 1 segundo
    digitalWrite(ledPin, LOW); // desliga o LED
    delay(1000);             // aguarda mais um segundo
}
```

Passos:



Experimento 2: Desvanecimento de um Led



Conectando os Jumpers

D8 - Pino Digital 8 ligado ao LED

Copie e cole este programa na IDE do arduino

```
/*
 * Fading (Desvanecimento do Led)
 * Este exemplo mostra como a desvanecer um LED usando a função analogWrite().
 * LED ligado ao pin 9 PWM
 */

int ledPin = 9; // LED no digital pin 9

void setup() { }

void loop() {

    // fade in de min a max em incrementos de 5 pontos:
    for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {

        // define o valor (escala de 0 a 255):
        analogWrite(ledPin, fadeValue);

        // espera por 30 milésimos de segundo para ver o efeito de escurecimento
        delay(30);
    }

    // fade out de Max para min com incrementos de 5 pontos:
    for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {

        // define o valor (escala de 0 a 255):
        analogWrite(ledPin, fadeValue);

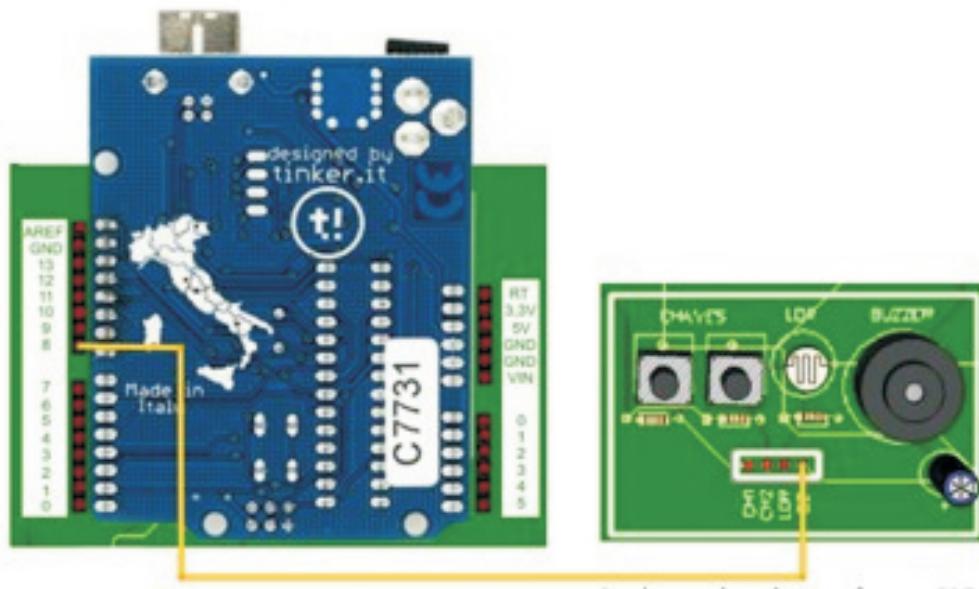
        // espera por 30 milésimos de segundo para ver o efeito de escurecimento
        delay(30);
    }
}
```

Passos:



Experimento 3: Gerando sons - Buzzer

Faça as ligações conforme abaixo



Conectando os Jumpers

D8 - Pino Digital 8 ligado ao BUZZER

Copie e cole este programa na IDE do arduino

```
/*
Melodia
*/
#include "pitches.h"

// notas da melodia:

int melody[] = {
    NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, NOTE_B3, NOTE_C4};

// durações nota: 4 nota = trimestre, nota 8 = oitava, etc:
int noteDurations[] = {
    4, 8, 8, 4, 4, 4, 4};

void setup() {

    // iteração sobre as notas da melodia:
    for (int thisNote = 0; thisNote < 8; thisNote++) {

        // calcular a duração da nota, em um segundo
        // dividido pelo tipo de nota.
        // por exemplo semínima = 1000 / 4, colcheia = 1000 / 8, etc
        int noteDuration = 1000/noteDurations[thisNote];
        tone(8, melody[thisNote],noteDuration);

        // para distinguir as notas, definir um tempo mínimo entre elas.
        // duração da nota + 30% parece funcionar bem:
        int pauseBetweenNotes = noteDuration * 1.30;
        delay(pauseBetweenNotes);

        // parar a reprodução de tom:
        noTone(8);
    }
}

void loop() {
    // para não repetir a melodia.
}
```

Passos:



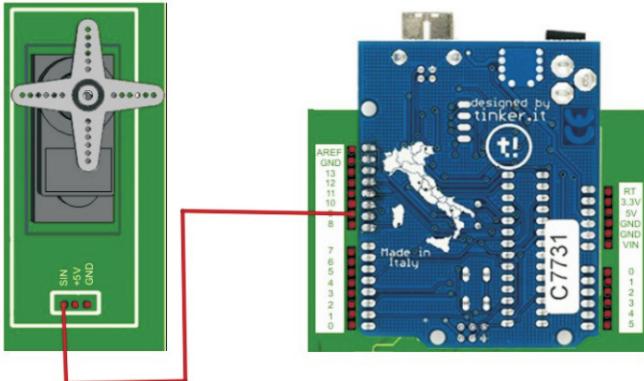
Experimento 4: Controlando um Servo Motor pelo Serial Monitor

Copie e cole este programa na IDE do arduino

Conectando os Jumpers

SIN do Servo ao pino digital 9

Faça as ligações conforme abaixo



Passos:



```
#include <Servo.h>

Servo myservo ;
int pin = 9;
int flag = 0 ;
int i ;
int char_in ;

void setup()
{
    pinMode(pin,OUTPUT);
    Serial.begin(9600) ;
    colocaMenu() ;
}

void loop()
{
    //piscaled() ;
    if(Serial.available() > 0 ) {
        char_in = Serial.read() ;
        if(char_in != -1) {
            Serial.println(char_in,BYTE) ;
        }
    }
    switch (char_in) {
        case '1':
            myservo.attach(pin,880,1510) ;
            myservo.write(180) ;
            Serial.println("Para o servo motor") ;
            break;
        case '2':
            myservo.attach(pin,880,2400) ;
            myservo.write(180) ;
            Serial.println("Anda para TRAZ") ;
            break;
        case '3':
            myservo.attach(pin,880,2400) ;
            myservo.write(0) ;
            Serial.println("ANDA PARA FRENTE") ;
            break;
        default :
            Serial.println("Opcão Invalida") ;
            break;
    }
}

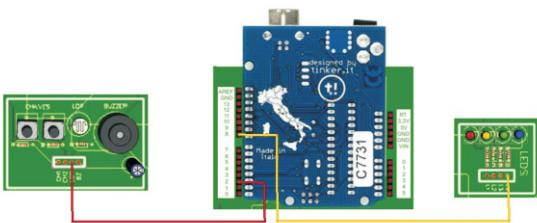
void colocaMenu() {
    Serial.println("Teste de motor servo") ;
    Serial.println("Seleciona :") ;
    Serial.println("1 - PARAR O MOTOR") ;
    Serial.println("2 - ANDA PARA TRAZ") ;
    Serial.println("3 - ANDA PARA FRENTE") ;
    Serial.println("Opcão ? ") ;
}
```

Experimento 5: Controlando um LED pelo sensor de luz LDR

Conectando os Jumpers

D8 - Pino Digital 8 ligado ao LED
D2 - Pino digital 2 Ligado ao LDR

Faça as ligações conforme abaixo



Copie e cole este programa na IDE do arduino

```
//Programa para controle de LED com LDR
int ledpin = 8;
int ldr = 2;
int val;

void setup () {
pinMode (ledpin, OUTPUT );
pinMode (ldr, INPUT );
}

void loop () {
val = digitalRead(ldr);

if (val == LOW ){
digitalWrite (ledpin, LOW );
}

if (val == HIGH ){
digitalWrite (ledpin, HIGH );
}

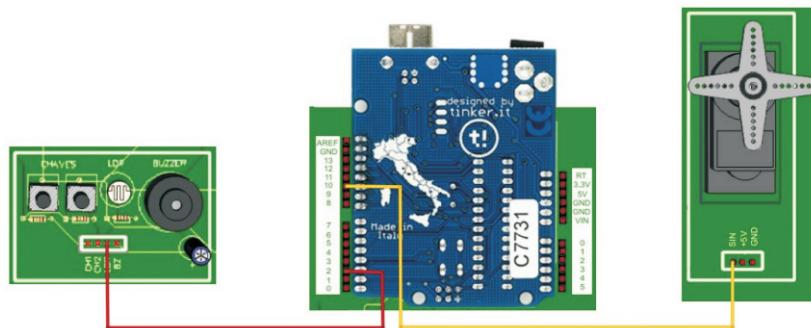
}
```

Passos:



Experimento 6: Controlando um Servo Motor pelo sensor de luz LDR

Faça as ligações conforme abaixo



Conectando os Jumpers

D10 - Pino Digital 10 ligado ao SIN do servo
D2 - Pino digital 2 Ligado ao LDR

Copie e cole este programa na IDE do arduino

```
//Programa para controlar o servo com LDR
#include <Servo.h>
Servo myservo;

int servo = 10;
int ldr = 2;
int val;

void setup(){
    pinMode(servo,OUTPUT);
    pinMode(ldr,INPUT);
    myservo.attach(servo,880,2300) ; //Inicia com o servo parado
    myservo.write(70);
    delay(300);
}

void loop() {
    val = digitalRead(ldr); //verifica o ldr e guarda seu status

    if (val == LOW ) {
        myservo.attach(servo,880,2300) ; //Pára o servo
        myservo.write(70);
        delay(300);
    }

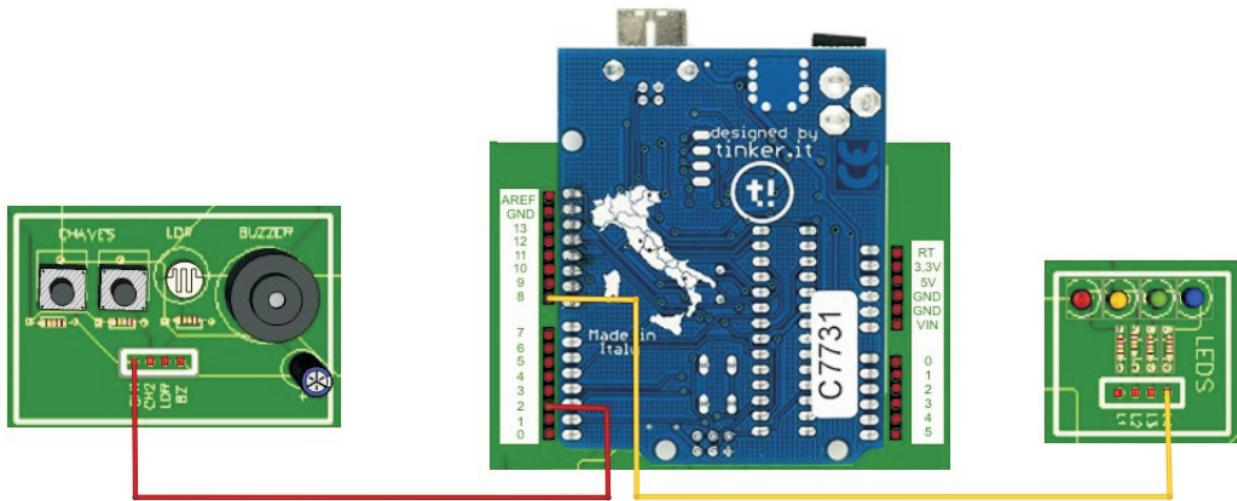
    if (val == HIGH ) {
        myservo.attach(servo,880,2300) ; // aciona o servo
        myservo.write(180) ;
        delay(300);
    }
}
```

Passos:



Experimento 7: Controlando um LED pela Chave Táctil

Faça as ligações conforme abaixo



Conectando os Jumpers

D8 - Pino Digital 8 ligado ao LED
D2 - Pino digital 2 Ligado ao Ch1

Copie e cole este programa na IDE do arduino

```
int chave = 7;      // Chave táctil no pin 7
int led = 8;        // LED no pino 8
int State = 0;      // variável para armazenar o estado da chave

void setup() {
    pinMode(led, OUTPUT);
    pinMode(chave, INPUT);
}

void loop(){
    State = digitalRead(chave);

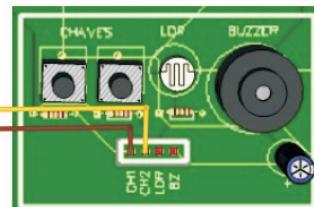
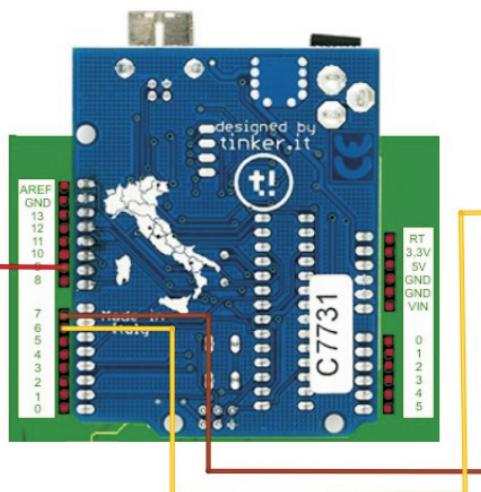
    if (State == HIGH) {
        digitalWrite(led, HIGH);
    }
    else {
        digitalWrite(led, LOW);
    }
}
```

Passos:



Experimento 8: Controlando um Servo Motor pela Chave Táctil

Faça as ligações conforme abaixo



Conectando os Jumpers

* Ligações das Chaves Táctis

PIN 6 - CH2 - Pino digital 6 ligado a Chave 2
PIN 7 - CH1 - Pino digital 7 ligado a Chave 1

* Ligação do Servo Motor

PIN9 - SIN - Pino digital 9 ligado ao pino de sinal do servo

Copie e cole este programa na IDE do arduino

```
#include <Servo.h>

Servo myservo ;
int ST1; //Declara variável ST1
int ST2; //Declara variável ST2
int CHAVE_1 = 7; //chave_1 ligada no pin digital 7
int CHAVE_2 = 6; //chave_2 ligada no pin digital 6
void setup(){
myservo.attach(9,880,2300);
pinMode(CHAVE_1,INPUT); //define as chaves como entrada
pinMode(CHAVE_2,INPUT);
myservo.write(70); //enviando um ângulo de 70 graus, mantém o servo parado
}
void loop() {

ST1 = digitalRead(CHAVE_1); //faz a leitura da chave 1
ST2 = digitalRead(CHAVE_2); //faz a leitura da chave 2

if (ST1 == HIGH and ST2==HIGH){PARA();}
if (ST1 == LOW and ST2==HIGH){GIRA();}
if (ST1 == HIGH and ST2==LOW){VOLTA();}
}

void GIRA() {
myservo.write(180); // gira o servo
delay(1000);
PARA();
}
void PARA() {
myservo.write(70); // pára o servo
delay(5);
}

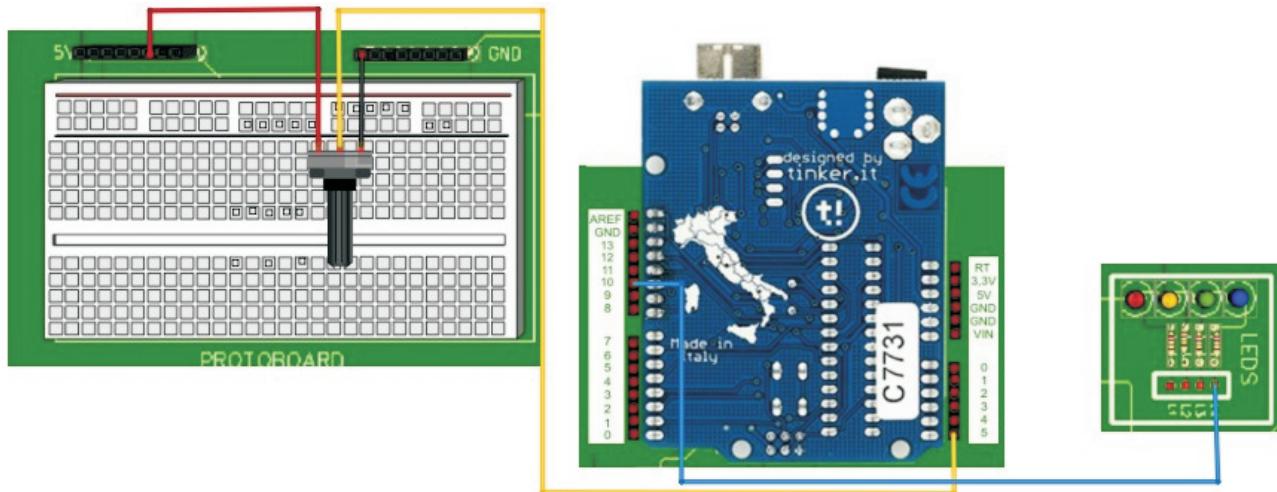
void VOLTA() {
myservo.write(0); // gira o servo sentido contrário
delay(1000);
PARA();
}
```

Passos:



Experimento 9: Controlando um LED através do potenciômetro

Faça as ligações conforme abaixo



Conectando os Jumpers

D10 - Pino Digital 10 ligado a um LED

A5 - Pino 5 analógico do arduino ligado ao pino central do potenciômetro

Copie e cole este programa na IDE do arduino

```
int potPin = 5;      // pino de entrada para o potenciômetro
int ledPin = 10;     // pino de saída para o LED

void setup()
{
    pinMode(ledPin, OUTPUT); // declara ledPin como SAÍDA
}

void loop()
{
    digitalWrite(ledPin, HIGH); // ledPin em on
    delay(analogRead(potPin)); // Espera por um tempo "potPin"

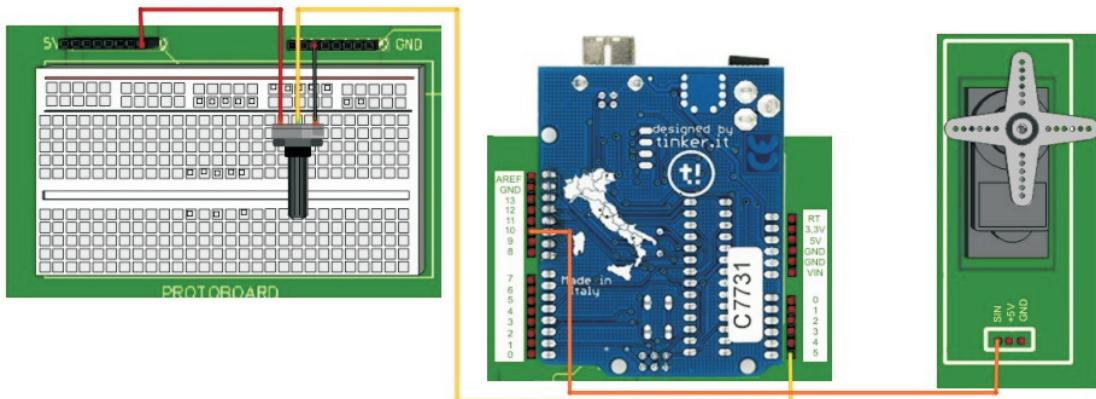
    digitalWrite(ledPin, LOW); // ledPin em off
    delay(analogRead(potPin)); // Espera por um tempo "potPin"
}
```

Passos:



Experimento 10: Controlando um Servo através do potenciômetro

Faça as ligações conforme abaixo



Conectando os Jumpers

SIN - Pino de sinal do Servo ao Pin Digital 10

A5 - Pino 5 analógico do arduino ligado ao pino central do potenciômetro

Copie e cole este programa na IDE do arduino

```
#include <Servo.h>
Servo servo1;
int pinopot = 5;
analogico 5
int valor;
void setup()
{
servo1.attach(10); // Porta onde ira ser ligado o servo
}
void loop()
{
valor = analogRead(pinopot); // leitura do valor do potenciometro e "escrita" desse valor na variavel valor
valor = map(valor, 0, 1023, 0, 179); // Escala para que o valor do potenciometro possa corresponder a um valor de posição no
servo1.write(valor); // Gira o servo de acordo com a escala da linha acima
delay(1); // Espera de 1 milisegunda para que o servo possa atingir a sua posição
}
```

Passos:



Experimento 11: RTC - Real Time Clock

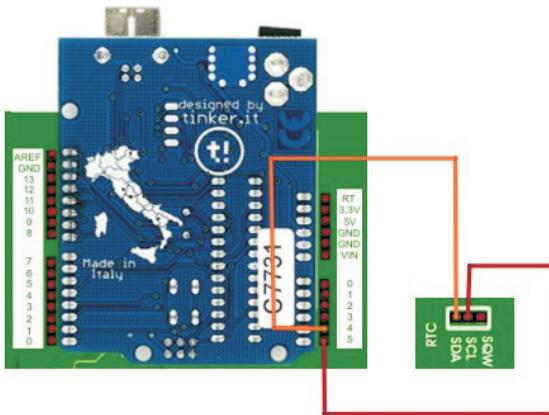
Copie e cole este programa na IDE do arduino

Programando o RTC com a data e horas atuais e visualizando no Serial Monitor.

Conectando os Jumpers

- * A4 -Ligar o pino analógico 4 ao SDA
- * A5 -Ligar o pino analógico 5 ao SCL

Faça as ligações conforme abaixo



Passos:



```
#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68
// Convert normal decimal numbers to binary coded decimal
// Ligar o SDA ao pin4 da entrada analógica e SCL ao pin5 da entrada analógica
byte decToBcd(byte val)
{
    return ( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
    return ( (val/16*10) + (val%16) );
}
void setDateDs1307(byte second, // 0-59
byte minute, // 0-59
byte hour, // 1-23
byte dayOfWeek, // 1-7
byte dayOfMonth, // 1-28/29/30/31
byte month, // 1-12
byte year) // 0-99
{
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.send(decToBcd(second)); // 0 to bit 7 starts the clock
    Wire.send(decToBcd(minute));
    Wire.send(decToBcd(hour)); // If you want 12 hour am/pm you need to set
    // bit 6 (also need to change readDateDs1307)
    Wire.send(decToBcd(dayOfWeek));
    Wire.send(decToBcd(dayOfMonth));
    Wire.send(decToBcd(month));
    Wire.send(decToBcd(year));
    Wire.endTransmission();
}
// Gets the date and time from the ds1307
void getDateDs1307(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
    // Reset the register pointer
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.endTransmission();
    Wire.requestFrom(DS1307_I2C_ADDRESS, 7);
    // A few of these need masks because certain bits are control bits
    *second = bcdToDec(Wire.receive() & 0x7f);
    *minute = bcdToDec(Wire.receive());
    *hour = bcdToDec(Wire.receive() & 0x3f); // Need to change this if 12 hour am/pm
    *dayOfWeek = bcdToDec(Wire.receive());
    *dayOfMonth = bcdToDec(Wire.receive());
    *month = bcdToDec(Wire.receive());
    *year = bcdToDec(Wire.receive());
}
void setup()
{
    byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    Wire.begin();
    Serial.begin(9600);
    //Digite aqui os dados de Data e Hora atuais para configurar o RTC
    second = 45; //Segundos
    minute = 32; //Minutos
    hour = 8; //Hora
    dayOfWeek = 7; //Dia da semana sendo 1 domingo
    dayOfMonth = 30; //Dia do mês
    month = 4; //Mês atual
    year = 11; //Ano atual
    setDateDs1307(second, minute, hour, dayOfWeek, dayOfMonth, month, year);
}
void loop()
{
    byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
    Serial.print("Sao: ");
    Serial.print(hour, DEC);
    Serial.print(":");
    Serial.print(minute, DEC);
    Serial.print(":");
    Serial.print(second, DEC);
    Serial.print(" ");
    Serial.print(dayOfMonth, DEC);
    Serial.print(" ");
    Serial.print(month, DEC);
    Serial.print("/");
    Serial.print(year, DEC);
    switch(dayOfWeek){
        case 1:Serial.print(" Domingo"); break;
        case 2:Serial.print(" Segunda-feira"); break;
        case 3:Serial.print(" Ter a-feira"); break;
        case 4:Serial.print(" Quarta-feira"); break;
        case 5:Serial.print(" Quinta-feira"); break;
        case 6:Serial.print(" Sexta-feira"); break;
        case 7:Serial.print(" Sabado"); break;
        default:Serial.print(" ");
    }
    Serial.println(dayOfWeek);
    delay(1000);
}
```

Experimento 12: RTC - Real Time Clock

Visualizando a data e hora no Display LCD

Conectando os Jumpers

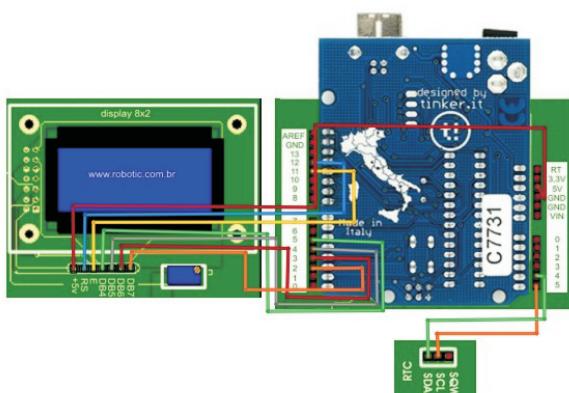
Ligações do Display ao Arduino:

- * RS - Pino digital 12
- * E - Enable pino digital 11
- * D4 - Pino digital 5
- * D5 - Pino digital 4
- * D6 - Pino digital 3
- * D7 - Pino digital 2

Ligações do RTC:

- * A4 -Ligar o pino analógico 4 ao SDA
- * A5 -Ligar o pino analógico 5 ao SCL

Faça as ligações conforme abaixo



Passos:



Copie e cole este programa na IDE do arduino

```
#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68
// Convert normal decimal numbers to binary coded decimal
// Ligar o SDA ao pin4 da entrada analógica e SCL ao pin5 da entrada analógica
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
byte decToBcd(byte val)
{
    return ( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
    return ( (val/16*10) + (val%16) );
}
void setDateDs1307(byte second, // 0-59
byte minute, // 0-59
byte hour, // 1-23
byte dayOfWeek, // 1-7
byte dayOfMonth, // 1-28/29/30/31
byte month, // 1-12
byte year) // 0-99
{
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.send(decToBcd(second)); // 0 to bit 7 starts the clock
    Wire.send(decToBcd(minute));
    Wire.send(decToBcd(hour)); // If you want 12 hour am/pm you need to set
    // bit 6 (also need to change readDateDs1307)
    Wire.send(decToBcd(dayOfWeek));
    Wire.send(decToBcd(dayOfMonth));
    Wire.send(decToBcd(month));
    Wire.send(decToBcd(year));
    Wire.endTransmission();
}
// Gets the date and time from the ds1307
void getDateDs1307(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
    // Reset the register pointer
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.endTransmission();
    Wire.requestFrom(DS1307_I2C_ADDRESS, 7);
    // A few of these need masks because certain bits are control bits
    *second = bcdToDec(Wire.receive() & 0x7f);
    *minute = bcdToDec(Wire.receive());
    *hour = bcdToDec(Wire.receive() & 0x3f); // Need to change this if 12 hour am/pm
    *dayOfWeek = bcdToDec(Wire.receive());
    *dayOfMonth = bcdToDec(Wire.receive());
    *month = bcdToDec(Wire.receive());
    *year = bcdToDec(Wire.receive());
}
void setup() //Registrando a data atual no RTC
{
    lcd.begin(16, 2);
    //byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    Wire.begin();
}
void loop()
{
    byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
    lcd.print(hour, DEC);
    lcd.print(":");
    lcd.print(minute, DEC);
    lcd.print(":");
    lcd.print(second, DEC);
    lcd.setCursor(0,1); //Posiciona cursor na segunda linha
    lcd.print(dayOfMonth, DEC);
    lcd.print(":");
    lcd.print(month, DEC);
    lcd.print(":");
    lcd.print(year, DEC);
    delay(1000);
    lcd.clear();
}
```

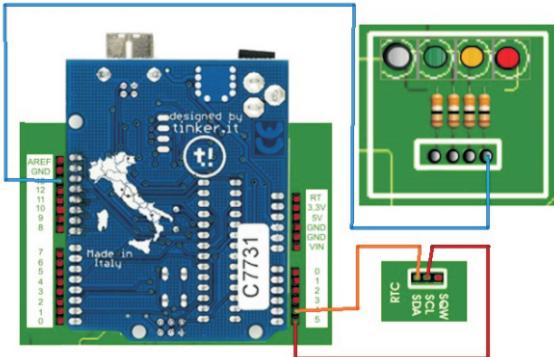
Experimento 13: RTC - Real Time Clock

Demostrativo de alarme - Acender um LED as 8:45h

Conectando os Jumpers

- * A4 -Ligar o pino analógico 4 ao SDA
- * A5 -Ligar o pino analógico 5 ao SCL
- * D13 pino digital 13 ligado ao LED

Faça as ligações conforme abaixo



Passos:



Copie e cole este programa na IDE do arduino

```
#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68
// Convert normal decimal numbers to binary coded decimal
// Ligar o SDA ao pin4 da entrada analógica e SCL ao pin5 da entrada analógica
int LED = 13; //led ligado ao pin 13
byte decToBcd(byte val)
{
    return ( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
    return ( (val/16*10) + (val%16) );
}
void setDateDs1307(byte second, // 0-59
byte minute, // 0-59
byte hour, // 1-23
byte dayOfWeek, // 1-7
byte dayOfMonth, // 1-28/29/30/31
byte month, // 1-12
byte year) // 0-99
{
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.send(decToBcd(second)); // 0 to bit 7 starts the clock
    Wire.send(decToBcd(minute));
    Wire.send(decToBcd(hour)); // If you want 12 hour am/pm you need to set
    // bit 6 (also need to change readDateDs1307)
    Wire.send(decToBcd(dayOfWeek));
    Wire.send(decToBcd(dayOfMonth));
    Wire.send(decToBcd(month));
    Wire.send(decToBcd(year));
    Wire.endTransmission();
}
// Gets the date and time from the ds1307
void getDateDs1307(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
    // Reset the register pointer
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.endTransmission();
    Wire.requestFrom(DS1307_I2C_ADDRESS, 7);
    // A few of these need masks because certain bits are control bits
    *second = bcdToDec(Wire.receive() & 0x7f);
    *minute = bcdToDec(Wire.receive());
    *hour = bcdToDec(Wire.receive() & 0x3f); // Need to change this if 12 hour am/pm
    *dayOfWeek = bcdToDec(Wire.receive());
    *dayOfMonth = bcdToDec(Wire.receive());
    *month = bcdToDec(Wire.receive());
    *year = bcdToDec(Wire.receive());
}
void setup() //Registrando a data atual no RTC
{
    //byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    Wire.begin();
    Serial.begin(9600);
}
void loop()
{
    byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
    Serial.print("Sao: ");
    Serial.print(hour, DEC);
    Serial.print(":");
    Serial.print(minute, DEC);
    Serial.print(":");
    Serial.print(second, DEC);
    Serial.print(" ");
    Serial.print(dayOfMonth, DEC);
    Serial.print("/");
    Serial.print(month, DEC);
    Serial.print("/20");
    Serial.println(year, DEC);
    // As 8:45h o led irá acender
    if (hour == 8 and minute == 45) { digitalWrite(LED, HIGH);}
    //Serial.println(dayOfWeek);
    delay(1000);
}
```

Experimento 14: RTC - Real Time Clock

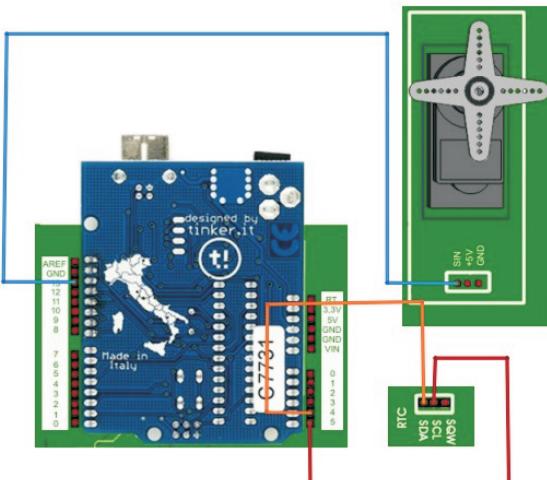
Demonstrativo de acionamento do servo motor de 8:45h às 8:47h

Conectando os Jumpers

Ligações do RTC:

- * A4 -Ligar o pino analógico 4 ao SDA
- * A5 -Ligar o pino analógico 5 ao SCL
- * D13 pino digital 13 ligado ao SIN do servo

Faça as ligações conforme abaixo



Passos:



Copie e cole este programa na IDE do arduino

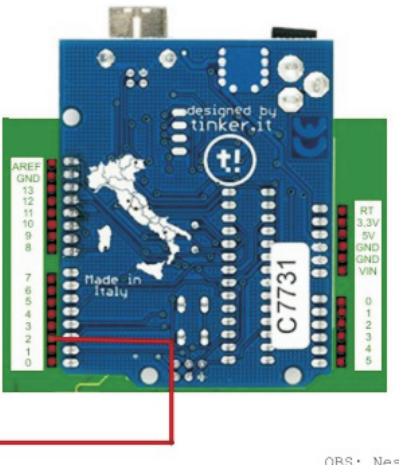
```
#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68
#include <MegaServo.h> //Inclue a biblioteca MegaServos
#define NBR_SERVOS 12 // Numero de Servos ligados. Maximo de 48 arduino MEGA, 12 para outros arduinos
#define FIRST_SERVO_PIN 3
MegaServo Servos[NBR_SERVOS];
// Convert normal decimal numbers to binary coded decimal
// Ligar o SDA ao pin4 da entrada analógica e SCL ao pin5 da entrada analógica
int LED = 13; //led ligado ao pin13
byte decToBcd(byte val)
{
    return ( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
    return ( (val/16*10) + (val%16) );
}
void setDateDs1307(byte second, // 0-59
byte minute, // 0-59
byte hour, // 1-23
byte dayOfWeek, // 1-7
byte dayOfMonth, // 1-28/29/30/31
byte month, // 1-12
byte year) // 0-99
{
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.send(decToBcd(second)); // 0 to bit 7 starts the clock
    Wire.send(decToBcd(minute));
    Wire.send(decToBcd(hour)); // If you want 12 hour am/pm you need to set bit 6 (also need to change setDateDs1307)
    Wire.send(decToBcd(dayOfWeek));
    Wire.send(decToBcd(dayOfMonth));
    Wire.send(decToBcd(month));
    Wire.send(decToBcd(year));
    Wire.endTransmission();
}
// Gets the date and time from the ds1307
void getDateDs1307(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
    // Reset the register pointer
    Wire.beginTransmission(DS1307_I2C_ADDRESS);
    Wire.send(0);
    Wire.endTransmission();
    Wire.requestFrom(DS1307_I2C_ADDRESS, 7);
    // A few of these need masks because certain bits are control bits
    *second = bcdToDec(Wire.receive() & 0x7f);
    *minute = bcdToDec(Wire.receive());
    *hour = bcdToDec(Wire.receive() & 0x3f); // Need to change this if 12 hour am/pm
    *dayOfWeek = bcdToDec(Wire.receive());
    *dayOfMonth = bcdToDec(Wire.receive());
    *month = bcdToDec(Wire.receive());
    *year = bcdToDec(Wire.receive());
}
void setup() //Registrando a data atual no RTC
{
    //byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    Servos[1].attach(3,880,2300); //Servo na porta digital 3
    Servos[1].write(70);
    delay(15);
    Wire.begin();
    Serial.begin(9600);
}
void loop()
{
    byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
    getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
    Serial.print("Sao: ");
    Serial.print(hour, DEC);
    Serial.print(":");
    Serial.print(minute, DEC);
    Serial.print(":");
    Serial.print(second, DEC);
    Serial.print(" ");
    Serial.print(dayOfMonth, DEC);
    Serial.print("/");
    Serial.print(month, DEC);
    Serial.print("/");
    Serial.println(year, DEC);
    // As 8:45h o servo será acionado e as 8:47 desligado
    if (hour == 8 and minute >= 45) { Servos[1].write(180);}
    if (hour == 8 and minute >= 47) { Servos[1].write(70);}
    //Serial.println(dayOfWeek);
    delay(1000);
}
```

Experimento 15: Mostrando os códigos das teclas do controle remoto no serial monitor

Conectando os Jumpers

D2 - Pino Digital 2 ligado ao Módulo Infrared

Faça as ligações conforme abaixo



Copie e cole este programa na IDE do arduino

```
#include <IRremote.h>

const int RECV_PIN = 2;
int LED = 13;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
    Serial.begin(9600);
    pinMode(LED, OUTPUT);
    irrecv.enableIRIn(); // Start the receiver
    irrecv.blink13(true);
}

void loop() {
    if (irrecv.decode(&results)) {
        Serial.println(results.value);

        irrecv.resume(); // Recebe proximo valor
    }
}
```

Passos:



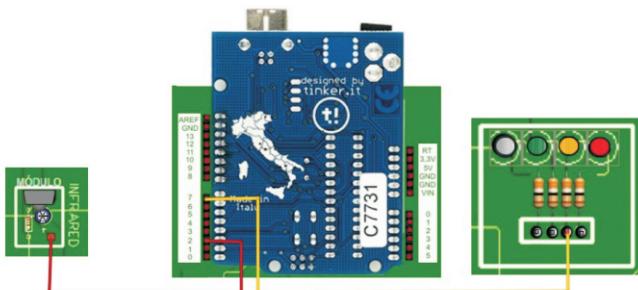
Experimento 16: Comandando um LED pelo controle remoto

Conectando os Jumpers

D7 - Pino Digital 8 ligado a um LED

D2 - Pino Digital 2 ligado ao Módulo Infrared

Faça as ligações conforme abaixo



OBS: Após carregar o programa use as teclas CH+ e CH- do controle para acionar o LED

Copie e cole este programa na IDE do arduino

```
#include <IRremote.h>

const int RECV_PIN = 2;
int LED = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
  irrecv.enableIRIn(); // Start the receiver
  irrecv.blink13(true);
}

void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value);

    if (results.value == 144){
      digitalWrite(LED, HIGH);}

    if (results.value == 2192){
      digitalWrite(LED, LOW); }

    irrecv.resume(); // Recebe proximo valor
  }
}
```

Passos:



Experimento 17: Comandando um servo motor pelo controle remoto

Conectando os Jumpers

INFRARED - Pin Digital 2
SIN - pin digital 9 (servo)

Copie e cole este programa na IDE do arduino

```
#include <IRremote.h> // Inclue a biblioteca IRremote
#include <MegaServo.h> //Inclue a biblioteca MegaServos
#define NBR_SERVOS 12 // Numero de Servos ligados.
#define FIRST_SERVO_PIN 8

MegaServo Servos[NBR_SERVOS] ;
const int RECV_PIN = 2; //Receptor di IR no pino digital 2

IRrecv irrecv(RECV_PIN);
decode_results results;
int servo = 9; //servo no pino digital 9

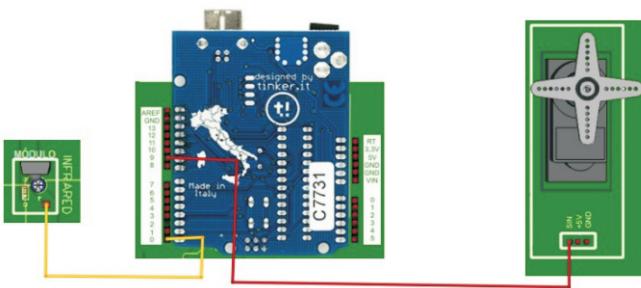
void setup()
{
    irrecv.enableIRIn(); // Inicia o recebimento
    Servos[1].attach(servo,880,2400) ;
    Servos[1].write(70) ;
    delay(15);
}

void loop()
{
    if (irrecv.decode(&results)) {
        if (results.value == 144){ //Tecla CH+ do controle remoto
            Servos[1].write(180) ;
            delay(15); }

        if (results.value == 2192){ //Tecla CH- do controle remoto
            Servos[1].write(0) ;
            delay(15);}

        if (results.value == 1168){ //Tecla VOL+ do controle remoto
            Servos[1].write(70) ;
            delay(15);}

        irrecv.resume(); // Recebe proximo valor
    }
}
```

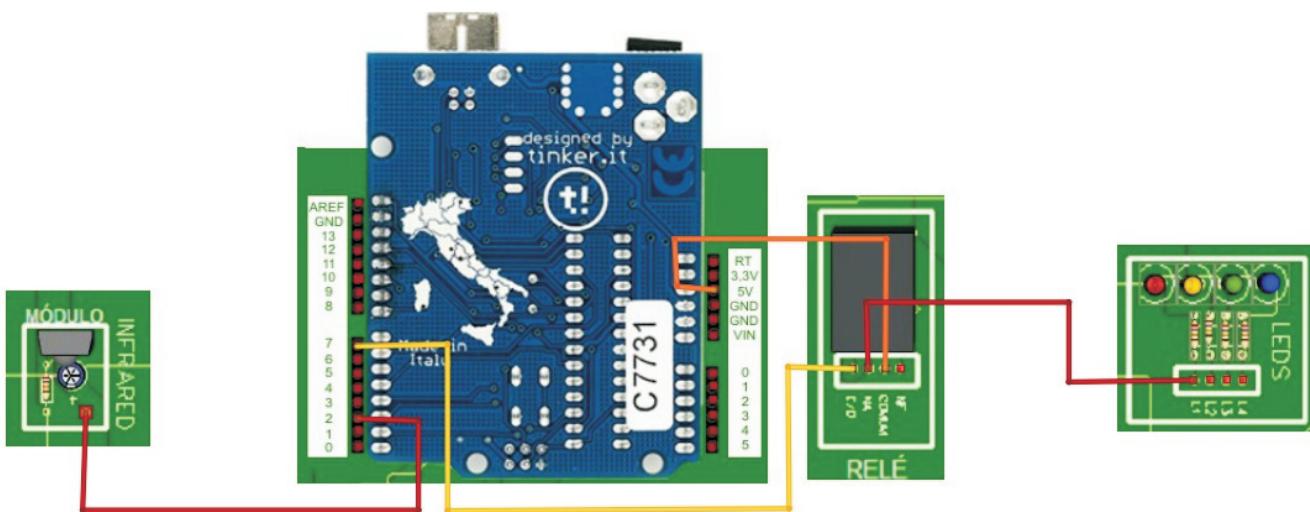


OBS: Numero de Servos ligados. Maximo de 48 para o arduino MEGA, e 12 para outros arduinos

Passos:



Experimento 18: Comandando um RELÉ pelo controle remoto



Conectando os Jumpers

D7 - Pino Digital 8 ligado ao RELÉ
D2 - Pino Digital 2 ligado ao Módulo Infrared

CONEXÕES DO RELÉ

Comum ao +5volts
NA - Ligar a um LED

Copie e cole este programa na IDE do arduino

```
#include <IRremote.h> // Inclue a biblioteca IRremote

const int RECV_PIN = 2; //Receptor di IR no pino digital 2
int RELE = 8; //Relé no pino 8
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
    pinMode(RELE, OUTPUT);
    irrecv.enableIRIn(); // Inicia o recebimento
}

void loop() {
    if (irrecv.decode(&results)) {

        if (results.value == 144){ //Tecla CH+ do controle remoto
            digitalWrite(RELE, LOW);} //Aciona o Relé

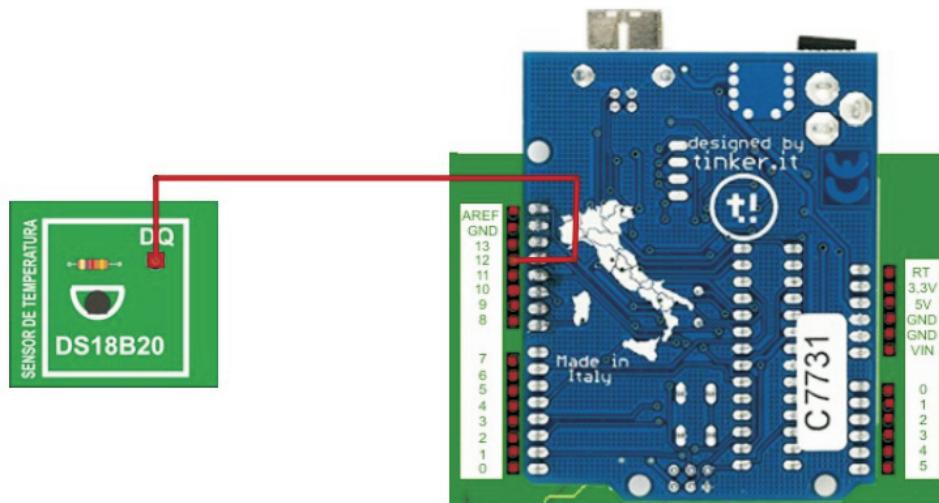
        if (results.value == 2192){ //Tecla CH- do controle remoto
            digitalWrite(RELE, HIGH); } //Desliga o Relé

        irrecv.resume(); // Recebe proximo valor
    }
}
```

Passos:



Experimento 19: Mostrando temperaturas no serial monitor



Conectando os Jumpers

* DQ - Pino digital 12

Faça as ligações conforme abaixo

Copie e cole este programa na IDE do arduino

```
/*
DallasTemperature.CPP - Dallas Temperature Control Library 1.0.0

*/
#include <DallasTemperature.h>
DallasTemperature tempSensor;

void setup(void) {
    // initialize entradas e saídas
    // inicializa a porta serial
    Serial.begin(9600);
    tempSensor.begin(12); // sensor ligado na porta digital 12
    Serial.println("Temperatura Atual: ");
}

void loop(void) {
    // Pega temperatura e envia para o monitor serial.

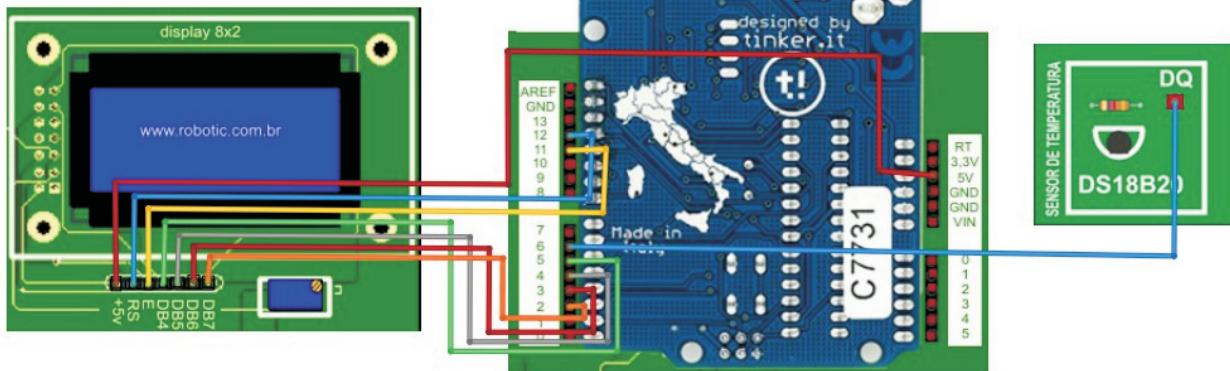
    Serial.print(tempSensor.getTemperature());
    Serial.print("C");
    Serial.println();
}
```

Passos:



Experimento 20: Mostrando temperaturas no display LCD

Conexões do display - Arduino Lab



Conectando os Jumpers

- * RS - Pino digital 12
- * E - Enable pino digital 11
- * D4 - Pino digital 5
- * D5 - Pino digital 4
- * D6 - Pino digital 3
- * D7 - Pino digital 2

Ligações do Sensor DS18B20:

* DQ do Sensor de Temperatura ao pino digital 6

Copie e cole este programa na IDE do arduino

```
/*
Mostrando a temperatura no Display
*/
#include <DallasTemperature.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
DallasTemperature tempSensor;

void setup(void) {
    // initialize entradas e saídas
    // inicializa a porta serial
    lcd.begin(8, 2);
    tempSensor.begin(6); // sensor ligado na porta digital 6
    lcd.print("Temp.:"); // envia a mensagem para o LCD.
}

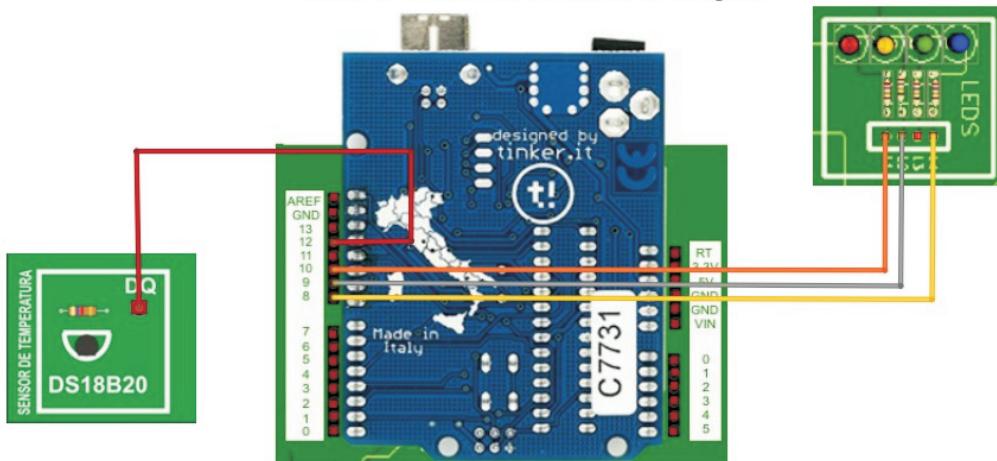
void loop(void) {
    // Pega temperatura e envia para o monitor serial.
    lcd.setCursor(0,1); //Posiciona cursor na segunda linha
    lcd.print(tempSensor.getTemperature());
    lcd.print("C");
}
```

Passos:



Experimento 21: Os leds acendem de acordo com a temperatura

Monte o Circuito Conforme a imagem:



Conectando os Jumpers

- D8 - Pino digital 8 ligado ao LED Azul
- D9 - Pino digital 9 Ligado ao Led Amarelo
- D10 - Pino digital 10 Ligado ao Led Vermelho
- D12 - Pino digital 12 ligado ao DQ do DS18B20

LED Azul acende para temperaturas menores que 25°
 LED Amarelo acende para temperaturas entre 25° e 30°
 LED Vermelho acende para temperaturas maiores que 30°

Passos:



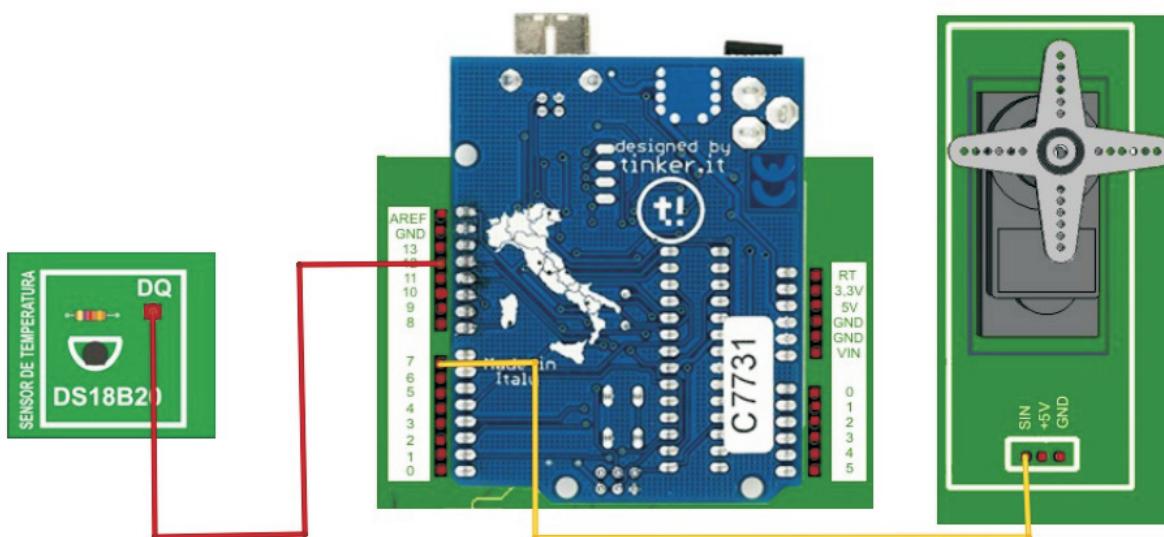
Copie e cole este programa na IDE do arduino

```
/*
SENSOR DE TEMPERATURA
- Este programa acende o led azul para temperatura abaixo de 25 graus
- Acende o led amarelo para temperaturas entre 25 e 30 graus
- Acende o led vermelho para temperaturas maiores que 30 graus
*/
#include <DallasTemperature.h>
DallasTemperature tempSensor;
int led_azul = 8;      //led azul no pino 8
int led_amar = 9;      //led amarelo no pino 9
int led_verm = 10;     //led vermelho no pino 10
int temp;

void setup() { // inicializando entradas e saídas
  pinMode(led_azul, OUTPUT);
  pinMode(led_amar, OUTPUT);
  pinMode(led_verm, OUTPUT);
  Serial.begin(9600); // inicializa a porta serial
  tempSensor.begin(12); // sensor ligado na porta digital 12
  Serial.println("Temperatura Atual: ");
}

void loop() {
  Serial.print(tempSensor.getTemperature()); // envia para o monitor serial.
  Serial.print("C");
  Serial.println();
  Serial.print(tempSensor.getTemperature());
  if (tempSensor.getTemperature() < 25) {
    digitalWrite(led_azul, HIGH);           //verifica se a temperatura é menor que 20 graus
    digitalWrite(led_amar, LOW);
    digitalWrite(led_verm, LOW);
  }
  if (tempSensor.getTemperature() > 25 and tempSensor.getTemperature() < 30) {
    digitalWrite(led_azul, LOW);           //verifica se a temperatura é menor que 20 graus
    digitalWrite(led_amar, HIGH);
    digitalWrite(led_verm, LOW);
  }
  if (tempSensor.getTemperature() > 30) {
    digitalWrite(led_azul, LOW);           //verifica se a temperatura é menor que 20 graus
    digitalWrite(led_amar, LOW);
    digitalWrite(led_verm, HIGH);
  }
}
```

Experimento 22: O servo é acionado de acordo com a temperatura



Conectando os Jumpers

D7 - Pino Digital 7 ligado ao SIN do servo
 D12 - Pino Digital 12 ligado ao DQ do DS18B20

OBS:

Para este experimento serão necessários:

- uma pedra de gelo envolvida por um pano.
 - Um isqueiro (Cuidado somente adultos)
- Ao aproximar o isqueiro do sensor tome os devidos cuidados para não cometer nenhum acidente.

Temperatura menores que 25° servo gira para um lado
 Temperaturas entre 25° e 30° servo se mantém parado
 Temperaturas maiores que 30° servo gira em sentido contrário

Copie e cole este programa na IDE do arduino

```
/*
  SENSOR DE TEMPERATURA
  Este programa aciona um servo de acordo com a temperatura
*/
#include <DallasTemperature.h>
DallasTemperature tempSensor;
#include <MegaServo.h> //Inclue a biblioteca MegaServos
#define NBR_SERVOS 12 // Numero de Servos ligados
#define FIRST_SERVO_PIN 3
MegaServo Servos[NBR_SERVOS] ;
int temp;

void setup() { // inicializando entradas e saídas
  Servos[1].attach(7,880,2300); //Servo na porta digital 7
  Servos[1].write(70);
  delay(15);
  Serial.begin(9600); // inicializa a porta serial
  tempSensor.begin(12); // sensor ligado na porta digital 12
  Serial.println("Temperatura Atual: ");
}

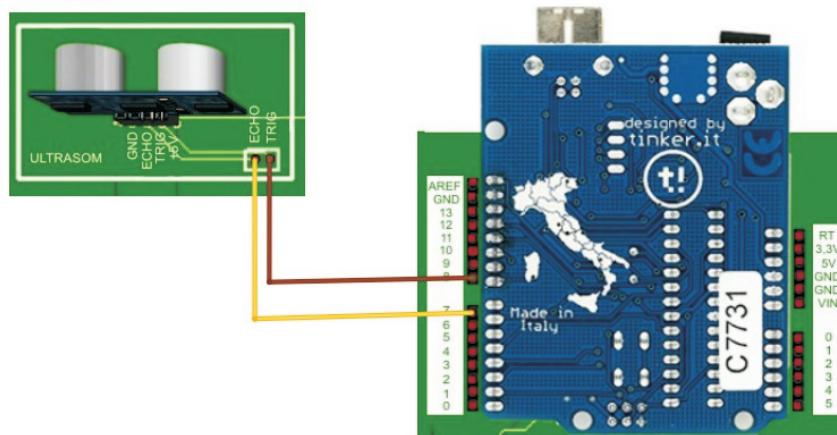
void loop() {
  Serial.print(tempSensor.getTemperature()); // envia para o monitor serial.
  Serial.print("C");
  Serial.println(" ");
  //Serial.println(tempSensor.getTemperature());
  if (tempSensor.getTemperature() < 25) {
    Servos[1].write(0);
    delay(30);
  }
  if (tempSensor.getTemperature() > 25 and tempSensor.getTemperature() < 30) {
    Servos[1].write(70);
    delay(30);
  }
  if (tempSensor.getTemperature() > 30) {
    Servos[1].write(180);
    delay(30);
  }
}
```

Passos:



Experimento 23: Medindo distâncias e mostrando no serial monitor

Monte o Circuito Conforme a imagem:



Conectando os Jumpers

TRIG - Pin Digital 6
ECHO - Pin Digital 7

Copie e cole este programa na IDE do arduino

```
#include "Ultrasonic.h"
#define echoPin 13      //Pino 13 recebe o pulso do echo
#define trigPin 12       //Pino 12 envia o pulso para gerar o echo

//iniciando a função e passando os pinos
Ultrasonic ultrasonic(12,13);

void setup()
{
    Serial.begin(9600);           //inicia a porta serial
    pinMode(echoPin, INPUT);     // define o pino 13 como entrada (recebe)
    pinMode(trigPin, OUTPUT);    // define o pino 12 como saída (envia)
}

void loop()
{
    digitalWrite(trigPin, LOW);  //seta o pino 12 com um pulso baixo "LOW"
    delayMicroseconds(2);        // delay de 2 microsegundos
    digitalWrite(trigPin, HIGH);  //seta o pino 12 com pulso alto "HIGH"
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // função Ranging, faz a conversão do tempo de
    // resposta do echo em centímetros, e armazena
    // na variável distância

    int distancia = (ultrasonic.Ranging(CM));

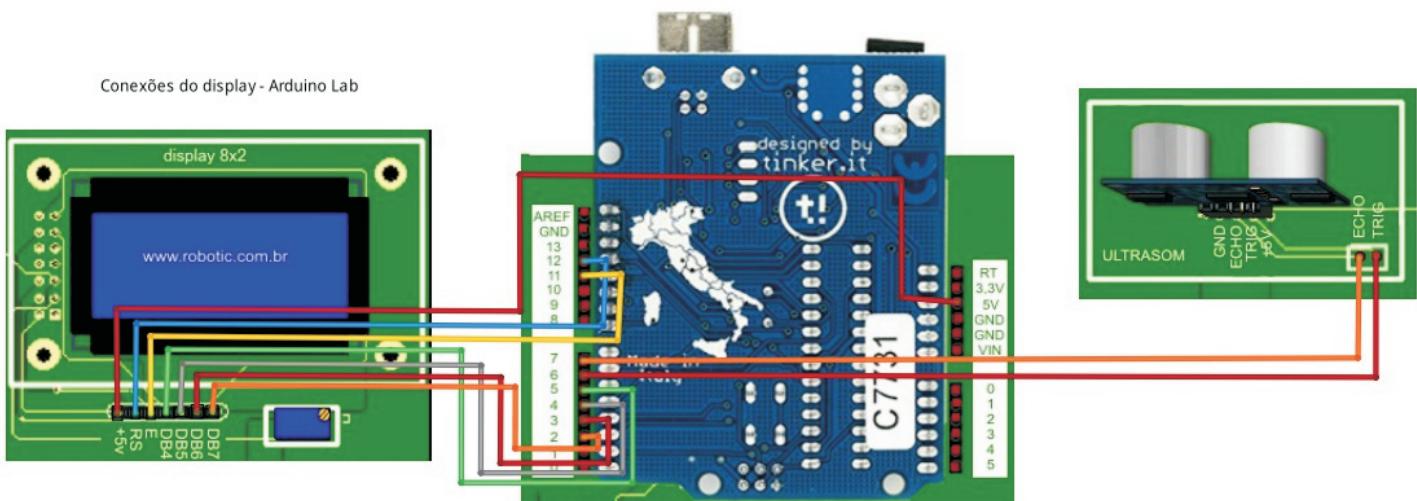
    Serial.print("Distância em CM: ");
    Serial.println(distancia);
    delay(1000); //espera 1 segundo para fazer a leitura novamente
}
```

Passos:



Experimento 24: Medindo distâncias e mostrando no display LCD

Monte o Circuito Conforme a imagem:



Conectando os Jumpers

- * RS - Pino digital 12
- * E - Enable pino digital 11
- * D4 - Pino digital 5
- * D5 - Pino digital 4
- * D6 - Pino digital 3
- * D7 - Pino digital 2
- * R/W - Pino GND
- * ECHO - Pino digital 7
- * TRIG - Trig pino digital 6

Copie e cole este programa na IDE do arduino

```
/*
Medindo distâncias com o Sensor de Ultrasom
e mostrando no display LCD

Ligações do Sensor:
* TRIG - pino digital 6
* ECHO - pino digital 7
*/
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int bip = 13;
int cm;
int trig = 6;
int echo = 7;
long microseconds;
void setup() {
// Serial.begin(9600);
lcd.begin(8, 2);
pinMode(trig, OUTPUT);
pinMode(echo, INPUT);

}

void loop() {

long duration, cm;
digitalWrite(trig, LOW);
delayMicroseconds(2);
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);
delayMicroseconds(2);
duration = pulseIn(echo, HIGH);
cm = microsecondsToCentimeters(duration);
lcd.clear();
lcd.print(cm);

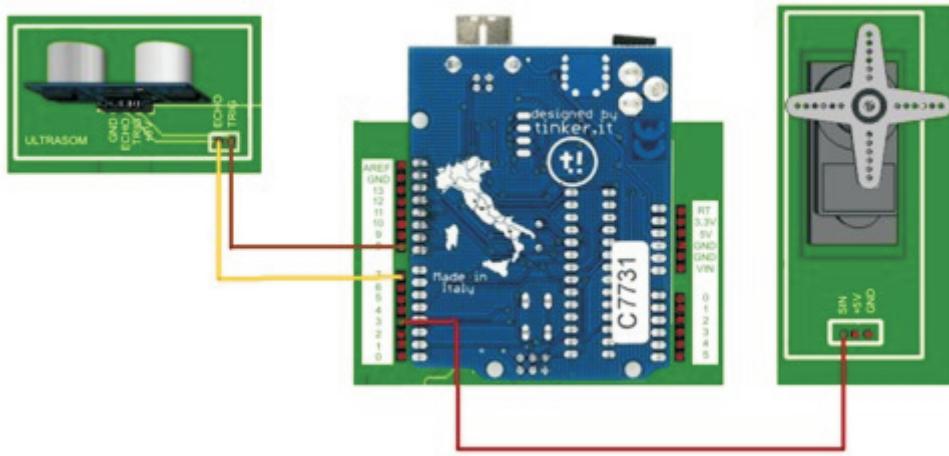
}

long microsecondsToCentimeters(long microseconds) { return
microseconds / 29 / 2;}
```

Passos:



Experimento 25: Medindo distâncias e comandando um servo



Conectando os Jumpers

TRIG - Pin Digital 8
ECHO - Pin Digital 7
SIN - pin digital 3 (servo)

Este programa irá comandar o servo usando a distância de um obstáculo

OBS:

Objeto a menos de 20cm o servo gira em um sentido
Objeto entre 20 e 60cm servo gira em sentido contrário
Objetos a mais de 60cm servo mantém parado

Copie e cole este programa na IDE do arduino

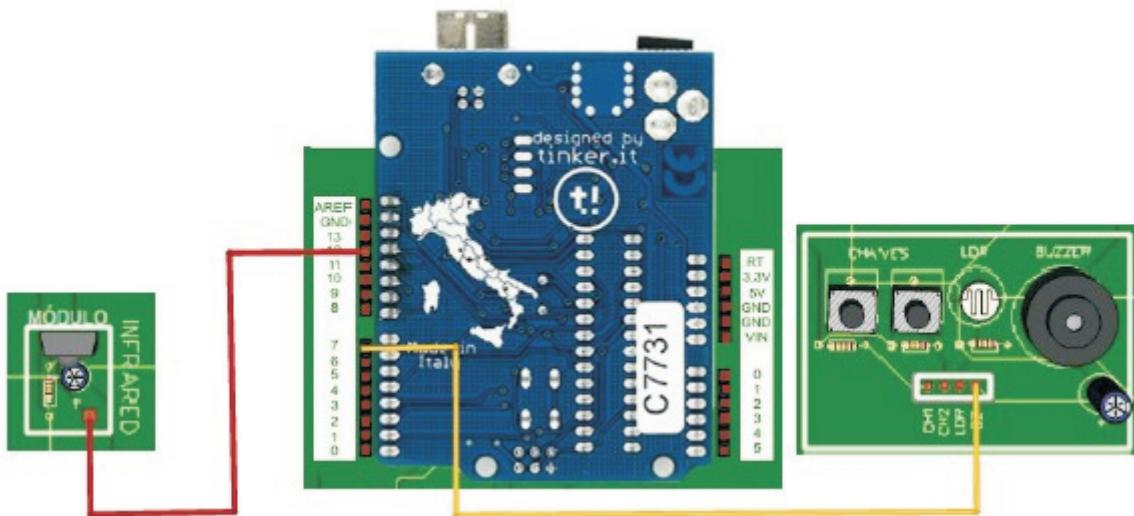
```
#include <MegaServo.h> //Inclue a biblioteca MegaServos
#define FIRST_SERVO_PIN 3 // Numero de Servos ligados. Maximo de 48 arduino MEGA, 12 para outros arduinos
MegaServo Servos[NBR_SERVOS];
// Obs: Se você não tiver esta biblioteca faça o download no site: www.robotic.com.br
int cm;
int trig = 8;
int echo = 7;
long microseconds;
void setup() {
    pinMode(trig, OUTPUT);
    pinMode(echo, INPUT);
    Servos[1].attach(3,880,2400);
}
void loop() {
    long duration, cm;
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    duration = pulseIn(echo, HIGH);
    cm = microsecondsToCentimeters(duration);
    if (cm <= 20){gira();} //Objeto a menos de 20cm o servo gira em um sentido
    if (cm > 20 and cm < 60){volta();} //Objeto entre 20 e 60cm servo gira em sentido contrário
    if (cm >= 60){pare();} // Objetos a mais de 60cm servo mantém parado
}
long microsecondsToCentimeters(long microseconds) { return microseconds / 29 / 2;}
void gira() {
    Servos[1].write(180);
    delay(300);
}
void pare() {
    Servos[1].write(70); // ângulo de 70 graus faz o servo parar
    delay(300);
}
void volta() {
    Servos[1].write(0);
    delay(300);
}
```

Passos:



Experimento 26: Alarme de Incêndio com o sensor DS18B20

Monte o Circuito Conforme a imagem:



Conectando os Jumpers

D8 - Pino Digital 8 ligado ao Buzzer
D12 - Pino Digital 12 ligado ao DQ do DS18B20

Copie e cole este programa na IDE do arduino

```
/*
Alarme de Incêndio com Dallas DS18B20
- Este programa aciona um alarme quando a temperatura
ambiente for maior que 30 graus.
*/
#include <DallasTemperature.h>
DallasTemperature tempSensor;
int temp;
int del = 250; // Duração do Tom
int lowrange = 2000; // Valor mais baixo da freqüencia usada
int highrange = 4000; // Freqüencia mais alta

void setup() { // inicializando entradas e saídas
tempSensor.begin(12); // sensor ligado na porta digital 12
pinMode(8, OUTPUT); // Buzzer porta 8
}

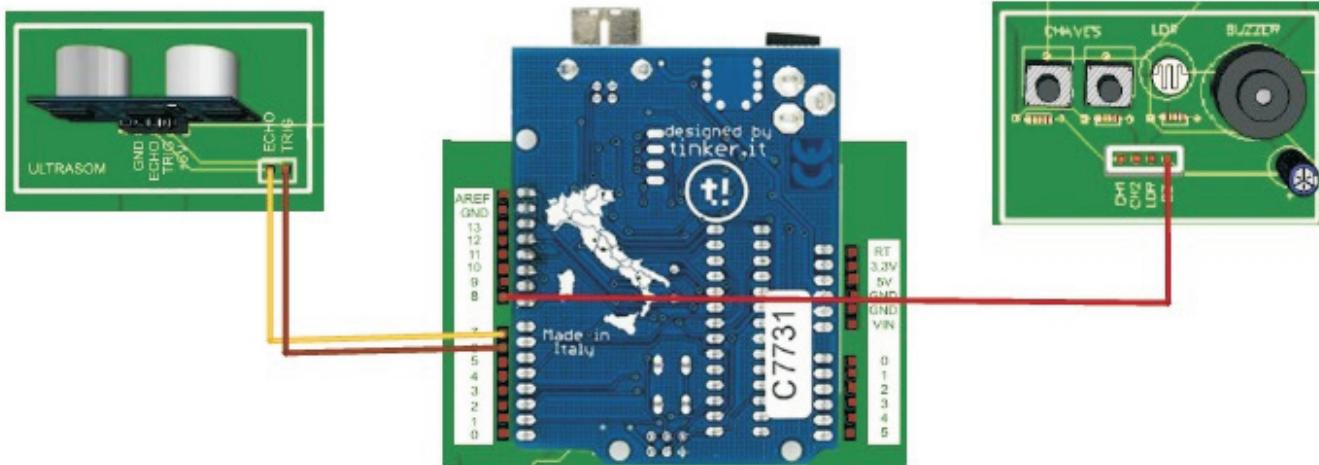
void loop() {
if (tempSensor.getTemperature() > 30){
for (int a = lowrange; a<=highrange; a++){
{
tone (8, a, del);
}
// Redução dos tons
for (int a = highrange; a>=lowrange; a--){
{
tone (8, a, del);
}
}
}
}
```

Passos:



Experimento 27: Alarme de intruso com ultrasom

Monte o Circuito Conforme a imagem:



Conectando os Jumpers

TRIG - Pin Digital 6
ECHO - Pin Digital 7
BUZZER - ligado ao pino digital 8

Copie e cole este programa na IDE do arduino

```
//Alarme de intruso com sensor de ultrasom
int cm;
int trig = 6;
int echo = 7;
long microseconds;
int temp;
int del = 250;           // Duração do Tom
int lowrange = 2000;    // Valor mais baixo da freqüencia usada
int highrange = 4000;   // Freqüencia mais alta
void setup() {
  Serial.begin(9600);

  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(8, OUTPUT); // Buzzer porta 8

}

void loop() {

  long duration, cm;
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  duration = pulseIn(echo, HIGH);
  cm = microsecondsToCentimeters(duration);
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(100);
  if (cm <= 250){alarme();} //Objeto a menos de 250cm
  else {digitalWrite(8,LOW);}

}

long microsecondsToCentimeters(long microseconds) { return microseconds / 29 / 2; }

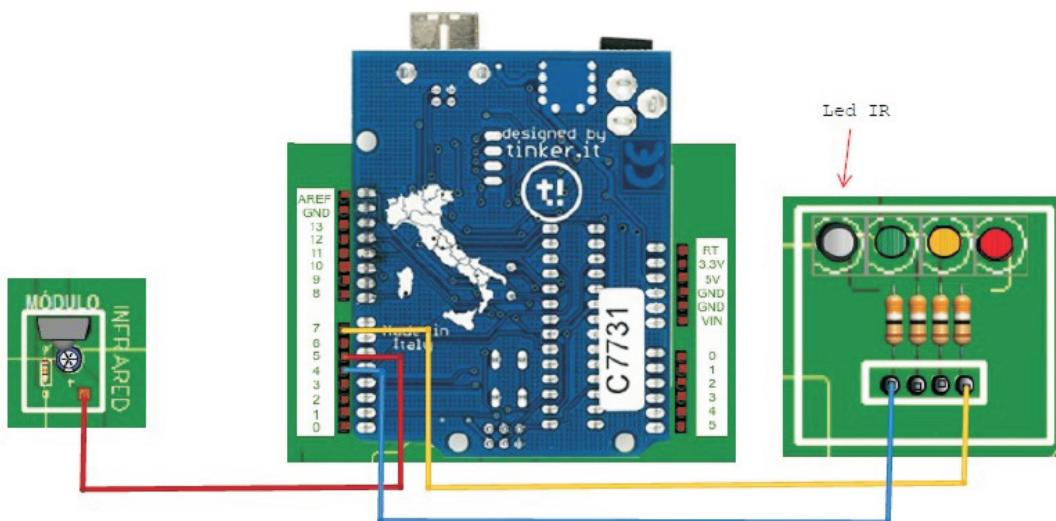
void alarme() {

  for (int a = lowrange; a<=highrange; a++)
  {
    tone (8, a, del);
  }
  // Redução dos tons
  for (int a = highrange; a>=lowrange; a--)
  {
    tone (8, a, del);
  }
}
```

Passos:



Experimento 28: Detectando obstáculos com Infrared



Conectando os Jumpers

D7 - Pino Digital 7 ligado ao LED
 D4 - Pino Digital 4 ligado ao LED INFRAVERMELHO
 D5 - Pino Digital 5 ligado ao Módulo Infrared

Aproxime um obstáculo do sensor de IR

Copie e cole este programa na IDE do arduino

```
//define pinos.
#define irLedPin 4           // LED infrared no pino 4
#define irSensorPin 5        // Sensor infrared no pino 5
int irRead(int readPin, int triggerPin);           //função protótipo
int LED = 7;
void setup()
{
  pinMode(irSensorPin, INPUT);
  pinMode(irLedPin, OUTPUT);
  pinMode(LED, OUTPUT);
  delay(100);
}
void loop()
{
  if (irRead(irSensorPin, irLedPin) == 0){digitalWrite(LED, HIGH);}
  if (irRead(irSensorPin, irLedPin) == 1){digitalWrite(LED, LOW);}

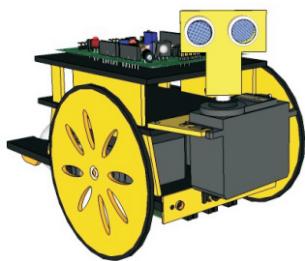
  delay(10); //Aguarda o envio
}
int irRead(int readPin, int triggerPin)
{
  //um período em 38.5khZ é de aproximadamente 26 microsegundos, ciclos int = 38
  int halfPeriod = 13;

  //26 microsegundos * 38 é mais ou menos um milissegundo
  int cycles = 38;
  int i;
  for (i=0; i <=cycles; i++)
  {
    digitalWrite(triggerPin, HIGH);
    delayMicroseconds(halfPeriod);
    digitalWrite(triggerPin, LOW);
    delayMicroseconds(halfPeriod - 1);    // - 1 para compensar a sobrecarga digitalWrite
  }
  return digitalRead(readPin);
}
```

Passos:



Experimento 29: Controlando a Unidade Móvel Robotic Remotamente Pelo Notebook

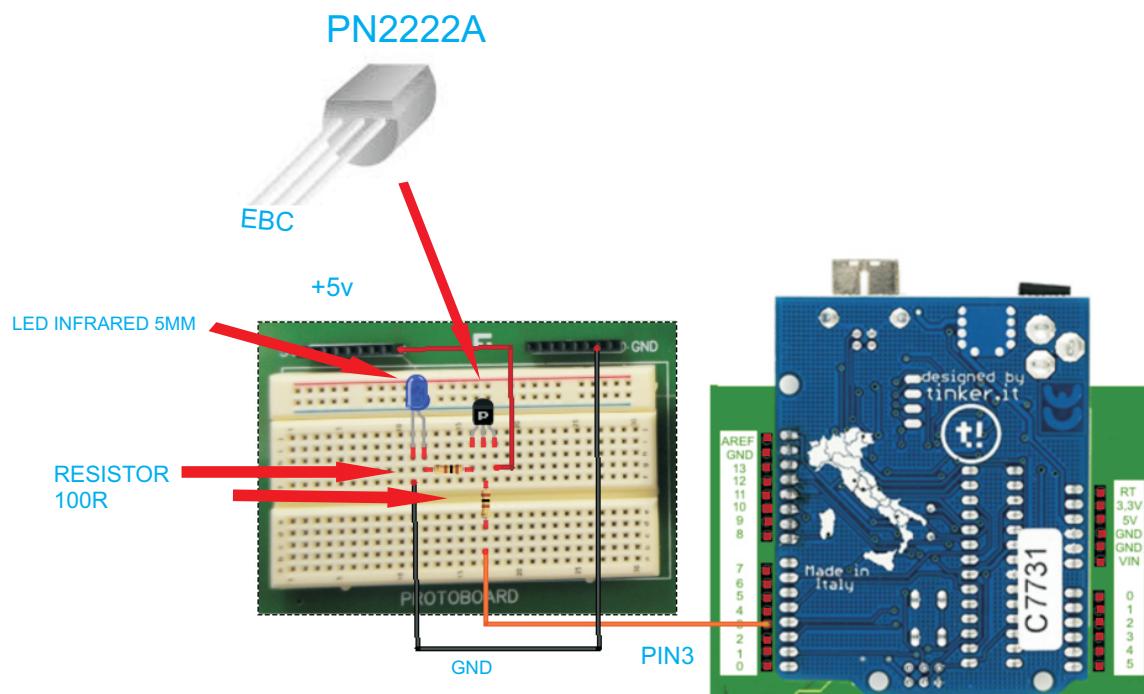


Material Necessário:

- 1 - UMR - Unidade Móvel Robotic
- 1 - Arduino Lab
- 1 - Led Infravermelho de 5mm
- 1 - Transistor PN2222A

A UMR será controlada a partir do Arduino Lab conectado a porta USB. No monitor serial irá mostrar 5 opções de movimentos para UMR. Os sinais serão emitidos pelo LED infravermelho. Para o correto funcionamento teremos que carregar 2 programas um no Arduino Lab (emissor) e o outro na UMR (receptor)

Faça as ligações conforme abaixo (emissor)



PROGRAMA EMISSOR: Copie e cole na IDE do arduino**Este programa deve ser carregado no Arduino Lab que irá emitir os sinais pelo LED IR.**

```
#include <IRremote.h>

IRsend irsend;

int flag = 0 ;
int char_in ;

void setup()
{
    Serial.begin(9600) ;
    colocaMenu() ;

}

void loop() {

//piscaled() ;
if(Serial.available() > 0 ){
    char_in = Serial.read() ;
    if(char_in != -1) {
        Serial.println(char_in,BYTE) ;

        switch (char_in) {
            case '1':
                irsend.sendSony(144, 12);
                Serial.println("frente") ;
                delay(1000);
                break;
            case '2':
                irsend.sendSony(2192, 12);
                Serial.println("Traz") ;
                delay(1000);
                break;
            case '3':
                irsend.sendSony(1168, 12);
                Serial.println("Direita") ;
                delay(1000);
                break;
            case '4':
                irsend.sendSony(3216, 12);
                Serial.println("Esquerda") ;
                delay(1000);
                break;
            case '5':
                irsend.sendSony(3536, 12);
                Serial.println("Parar") ;
                delay(1000);
                break;
            default :
                irsend.sendSony(3536, 12);
                delay(1000);
                break;
        }
    }
}

void colocaMenu() {
    Serial.println("CONTROLANDO A UMR PELA PORTA SERIAL USB") ;
    Serial.println("Selecione :");
    Serial.println("1 - FRENTE") ;
    Serial.println("2 - TRAZ") ;
    Serial.println("3 - DIREITA") ;
    Serial.println("4 - ESQUERDA") ;
    Serial.println("5 - PAPAR") ;
    Serial.println("OPCAO ? ");
}
```

**PROGRAMA RECEPTOR: Copie e cole na IDE do arduino
Este programa deve ser carregado na UMR (Unidade Móvel Robotic).**

```
#include <IRremote.h>
#include <MegaServo.h> //Inclue a biblioteca MegaServos
#define NBR_SERVOS 12 // Numero de Servos ligados. Maximo de 48 arduino MEGA, 12 para outros arduinos
#define FIRST_SERVO_PIN 8
MegaServo Servos[NBR_SERVOS] ;

const int RECV_PIN = 2; //SENSOR DE INFRARED NO PIN 2
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
    Serial.begin(9600);
    Servos[1].attach(8,880,2400) ; //SERVO DIREITO NO PIN 8
    Servos[2].attach(9,880,2400) ; //SERVO ESQUERDO NO PIN 9
    Servos[1].write(70) ;
    Servos[2].write(70) ;
    irrecv.enableIRIn(); // Start the receiver
    irrecv.blink13(true);
    delay(15);
}

void loop() {
    if (irrecv.decode(&results)) {
        Serial.println(results.value);

        if (results.value == 144){ //Chave CH+ frente
            Servos[1].write(180) ;
            Servos[2].write(0) ;
            delay(15);}

        if (results.value == 2192){ //Chave CH- volta
            Servos[1].write(0) ;
            Servos[2].write(180) ;
            delay(15); }

        if (results.value == 1168){ //Chave VOL+ direita
            Servos[1].write(0) ;
            Servos[2].write(0) ;
            delay(15); }

        if (results.value == 3216){ //Chave VOL+ esquerda
            Servos[1].write(180) ;
            Servos[2].write(180) ;
            delay(15); }

        if (results.value >= 3536){ //Chave JUMP PÁRA
            Servos[1].write(70) ;
            Servos[2].write(70) ;
            delay(15); }

        irrecv.resume(); // Recebe proximo valor
    }
}
```

Robotic

UNIDADE MÓVEL ROBOTIC

