Coletânea de Tutoriais LdG

Sumário

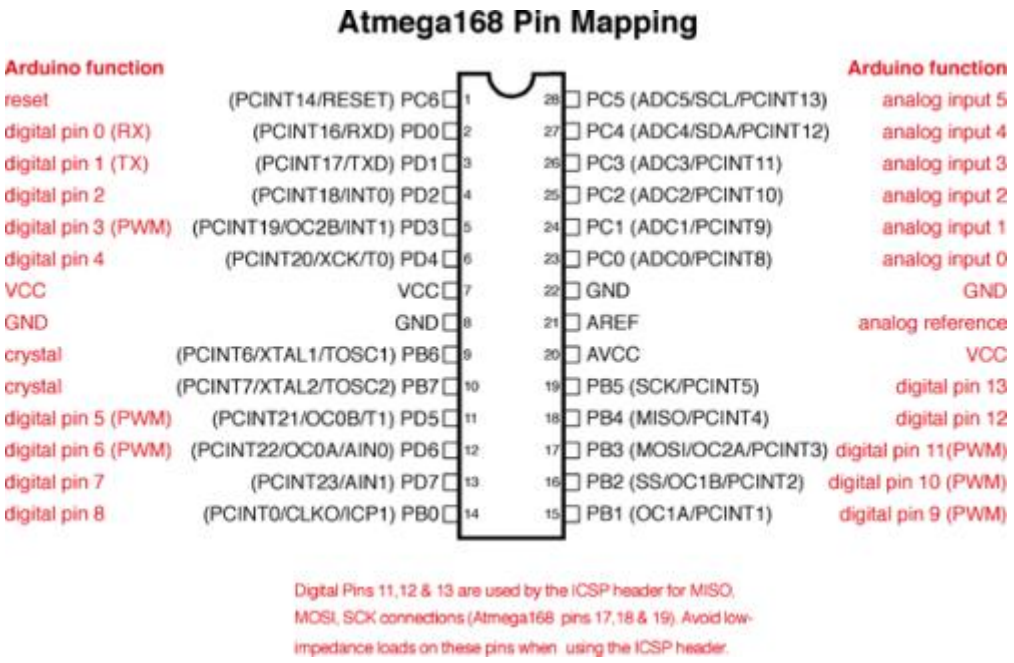
1. Tutorial: Como montar o Arduino na protoboard

A placa Arduino é uma placa de desenvolvimento open source (código aberto), isto é, qualquer pessoa pode montar a placa, mudar o Nome da placa (já que Arduino é uma marca registrada), usar para uso próprio e/ou vender. A placa Arduino pode ser utilizada para automação, controle de processos, comunicação, etc.

A placa Arduino vem com um microcontrolador da série atmega da marca ATMEL. O microcontrolador utilizado na placa Arduino UNO e Duemilanove é o atmega328P-PU. Este microcontrolador funciona com 5V de alimentação, memória flash de 32Kbytes, SRAM de 2Kbytes, EEPROM de 1Kbyte e velocidade de clock de 16Mhz.

Como o código é aberto, pode-se montar a placa na protoboard. A seguir mostraremos quais os componentes necessários e como montar o Arduino Duemilanove, pois é mais simples e utiliza comunicação por FTDI e não pelo atmega16U2 como o do UNO.

Abaixo mostra a pinagem do atmega328P-PU(atmega168 tem a mesma pinagem que o 328P-PU) referente a placa Arduino:



Para montar a placa Arduino Duemilanove na protoboard é necessário os seguintes componentes:

1x Atmega328P-PU com bootloader para Arduino Duemilanove

1x Breakout RS232 para USB Serial(aqui utilizamos este breakout para comunicação, pode-se usar outros)

2x capacitores de 22pF

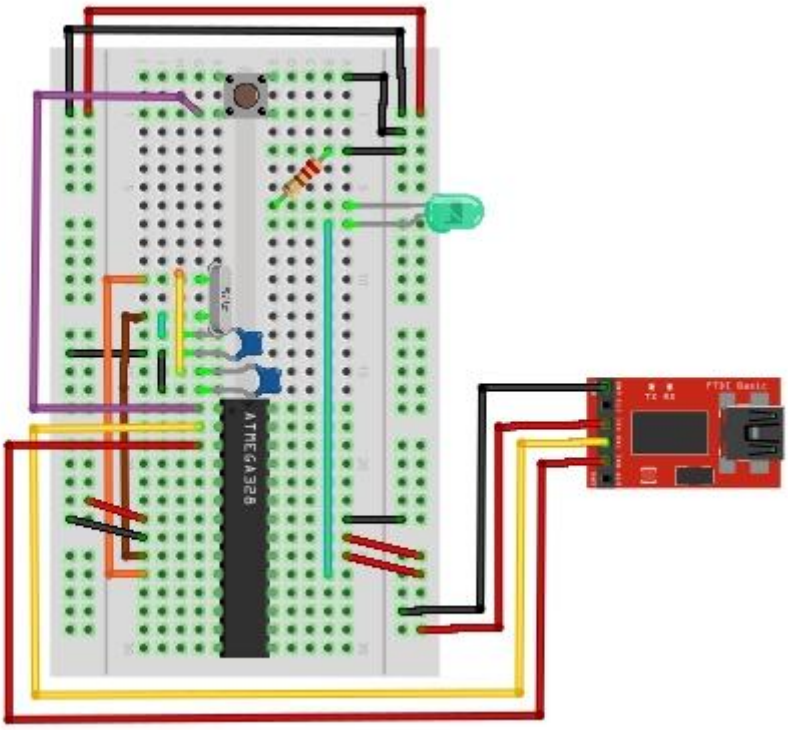
1x Cristal de 16Mhz

1x pushbutton

1x resistor de 220 ohm

1x Led

Seguindo a pinagem do atmega acima, primeiro coloque o atmega328P-PU na protoboard. Em seguida os capacitores e o cristal nos pino 9 e 10. O TXO do FTDI no RX (pino 2) do atmega, o RXI do FTDI no TX (pino 3) do atmega. Conecte o pushbutton entre o RESET (pino 1) do atmega e o GND. Conecte o pino 19 do atmega na perna maior do Led, e depois a perna menor do Led em uma das pontas do resistor e a outra ponta no GND. Conecte a alimentação e conecte todos os GNDs. A figura abaixo mostra a montagem final.



Pronto! Seu arduino montado na protoboard está pronta! Agora é só diversão!

Lembrando novamente que para funcionar, você tem que gravar o bootloader no atmega328P-PU. Para este caso, o bootloader utilizado é para a placa Arduino Duemilanove. Então configure a IDE do Arduino para Arduino Duemilanove.

Referências:

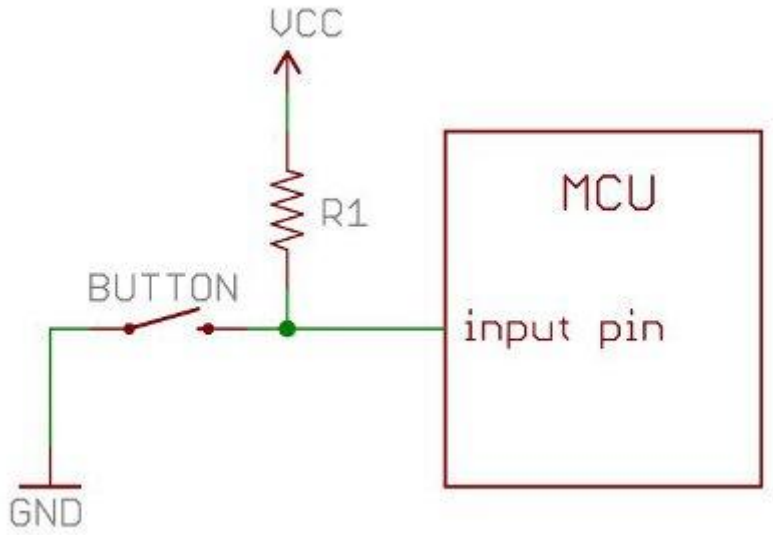
http://arduino.cc/en/Main/Standalone
http://labdegarag1.lojatemporaria.com/
http://labdegaragem.com/

2. Para que servem resistores Pull-Up/Pull-Down e como implementá-los

Muitas vezes ouvimos: Para que serve resistores de Pull-Up/Pull-Down e como implementar em microcontroladores(PIC, Atmega, etc).

Os Pull-Ups/Pull-Down são utilizados para evitar flutuação em pinos configurados como entradas(INPUT). Em geral, é necessário implementar externamente, mas muitas vezes há Pull-Ups implementados internamente em alguns pinos do microcontrolador. No caso dos microcontroladores Atmegas, já existem Pull-Ups internos em todos os pinos digitais e analógicos (OBS: Só use Pull-Up nos pinos analógicos caso utilizar estes como digitais), portanto não há necessidade de implementar Pull-Up externamente.

Caso necessite de utilizar Pull-Ups externamente, abaixo está mostrado como implementar:



Para escolher o resistor de Pull-Up é necessário satisfazer duas condições:

1. Quando o botão é pressionado, o pino de entrada vai para LOW. O resistor R1 limita a corrente do VCC que passa pelo botão e vai pro GND.
2. Quando o botão não é pressionado, o pino de entrada vai para HIGH. O resistor R1 limita a tensão no pino de entrada.

Para essas condições o resistor não pode ser de valor muito baixo, senão passará muita energia para o pino de entrada. E o resistor não pode ser muito alto senão não passará tensão necessária para o pino de entrada e/ou afetará na velocidade de transmissão.

Em geral, o resistor R1 deve ser um décimo menor que a impedância do pino de entrada, mas geralmente a impedância de entrada varia entre 100Kohm e 1Mohm. (OBS: Procure no datasheet para informações mais detalhadas)

Mas suponha que queira limitar a corrente do pino de entrada para 1mA(0.001A). Fazendo o cálculo pela Lei de Ohm:

V=RxI

Sendo, V=5V(tensão de alimentação)

I=1mA(corrente através do resistor e chegando no pino de entrada)

R=? (resistor de Pull-Up)

Resolvendo o cálculo:

5 = Rx0.001

Portanto: R = 5000 ohm para o resistor de Pull-Up

E é isso aí! Caso tenha dúvidas, poste um comentário aqui no blog!

Referências:

http://www.sparkfun.com/tutorials/218

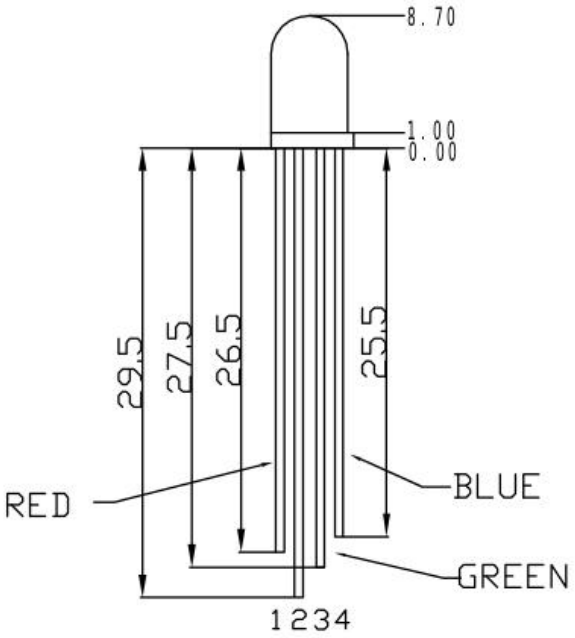
http://pt.wikipedia.org/wiki/Resistores_pull-up

3. Tutorial LED RGB com Arduino

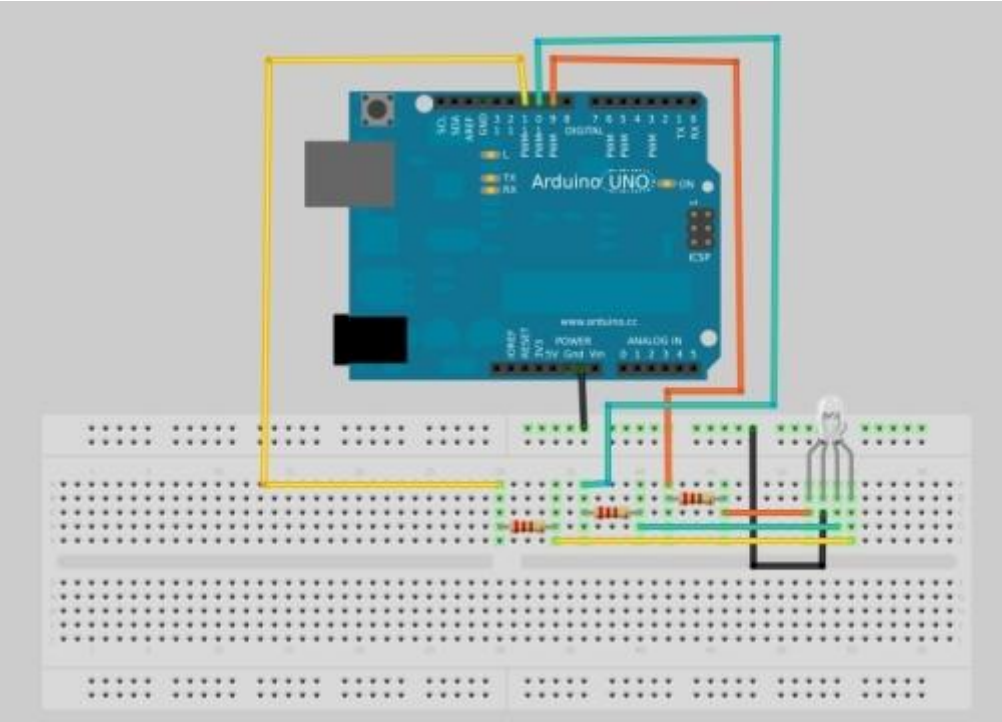
Com o LED RGB pode-se acender as três cores primárias: Vermelho(Red), Verde(Green) e Azul(Blue). Além das cores primárias pode-se misturar essas cores para criar outras cores. Por exemplo: Amarelo(Vermelho e Verde), Ciano(Verde e Azul) e Magenta(Azul e Vermelho), Branco(Vermelho, Azul e Verde).

Com o Arduino, pode-se variar a intensidade de cada cor para adquirir a cor desejada.
Neste link, tem uma tabela RGB.

Neste Tutorial, fizemos apenas mostrando as cores primárias e as cores citadas no exemplo acima. Na figura abaixo mostra a pinagem necessária para fazer as ligações com o Arduino. Caso queira ver o datasheet, clique aqui.



Agora que sabemos a pinagem do LED RGB, podemos fazer as ligações:



Utilizando os pinos digitais 9,10 e 11 (Pinos PWMs referênciados com '~' na frente) do Arduino, a programação feita está demonstrada abaixo:

```
int redpin=9;      //Pin 9
int greenpin=10;    //Pin 10
int bluepin=11;      //Pin 11
int var=0;
int var1=0;


void setup()
{
}

void loop()
{
  for(var=250;var<255;var++)
  {
  analogWrite(redpin,var);  //RED
  analogWrite(greenpin,0);
  delay(500);
  analogWrite(redpin,0);    //GREEN
  analogWrite(greenpin,var);
  delay(500);
  analogWrite(greenpin,0);   //BLUE
  analogWrite(bluepin,var);
  delay(500);
  analogWrite(bluepin,0);
  delay(500);
  }

  for(var1=250;var1<255;var1++)
  {
  analogWrite(redpin,var1);   //YELLOW
  analogWrite(greenpin,var1);
  delay(500);
```

```
  analogWrite(redpin,0);
  delay(500);
  analogWrite(greenpin,var1);  //CYAN
  analogWrite(bluepin,var1);
  delay(500);
  analogWrite(greenpin,0);
  delay(500);
  analogWrite(bluepin,var1);    //MAGENTA
  analogWrite(redpin,var1);
  delay(500);
  analogWrite(bluepin,0);
  delay(500);
  analogWrite(bluepin,var1);
  analogWrite(redpin,var1);
  analogWrite(greenpin,var1);
  }

}
```

Na programação, no primeiro loop "for" mostra as cores primárias começando o PWM em 250 e indo até 255. No segundo loop "for" mostra as cores combinando duas cores primárias: Amarelo, Ciano, Magenta e Branco.

Neste tutorial foi utilizado apenas uma variável para a intensidade de todas as cores, mas pode-se utilizar mais variáveis para mudar as intensidades independentes.

E é isso!! Um tutorial simples e prático para mostrar o que o LED RGB pode fazer!!! Qualquer dúvida, poste aqui no blog!! Se tiver alguma sugestão de tutorial poste neste Link!

Referências:

http://labdegarag1.lojatemporaria.com/led-rgb.html

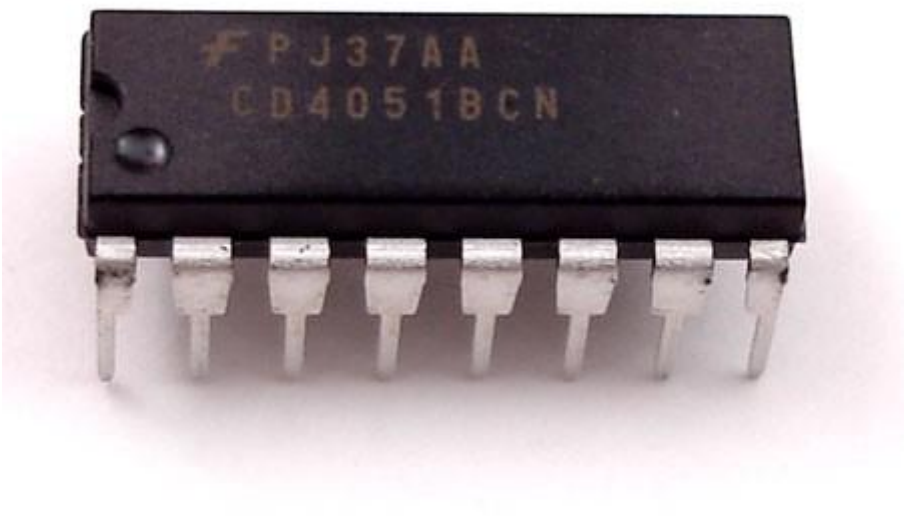
http://www.sparkfun.com/products/105

http://wiring.org.co/learning/basics/rgbled.html

http://www.rapidtables.com/web/color/RGB_Color.htm

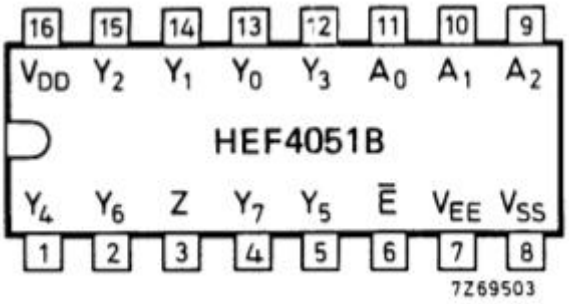
http://arduino.cc/en/Reference/HomePage

4. Tutorial sobre 4051(multiplexador, demultiplexador) e como utilizá-lo com Arduino



Neste tutorial, vamos mostrar como utilizar o multiplexador/demultiplexador 4051 com o arduino. O circuito integrado 4051 é multiplexador ou demultiplexador analógico que por endereçamento pode acionar uma porta desejada.

Olhando o datasheet, podemos ver a pinagem dele, como mostrada abaixo:



Onde:

VDD - Alimentação de 3V a 15V;

VSS - GND ou terra;

VEE - É a tensão negativa para geração de ruído entre as I/O's, neste tutorial, vamos conectá-lo no GND;

A0,A1,A2 - São as portas de endereçamento;

Y0,Y1...Y6,Y7 - São as portas para I/O's;

Z - É porta de entrada/saída a qual você vai utilizar para conectar com Arduino;

E - É o pino de habilitação do circuito integrado 4051.

Olhando novamente no datasheet, podemos ver qual endereço é necessário para ativar uma saída:

FUNCTION TABLE

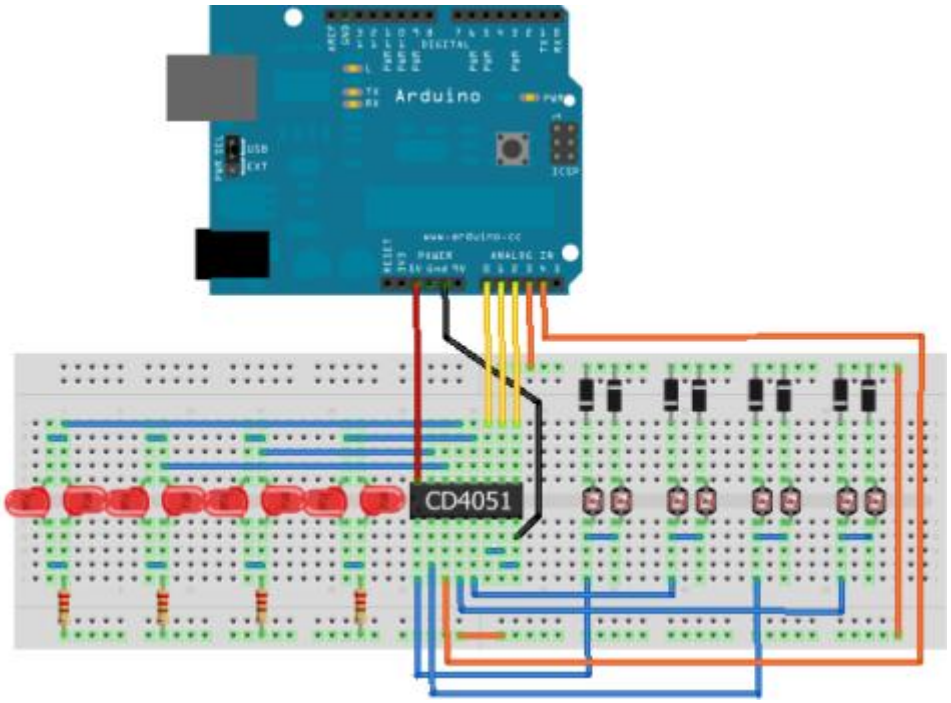
INPUTS				CHANNEL ON
$\overline{E}$	A_2	A_1	A_0	
L	L	L	L	Y_0–Z
L	L	L	H	Y_1–Z
L	L	H	L	Y_2–Z
L	L	H	H	Y_3–Z
L	H	L	L	Y_4–Z
L	H	L	H	Y_5–Z
L	H	H	L	Y_6–Z
L	H	H	H	Y_7–Z
H	X	X	X	none

Notes

1. H = HIGH state (the more positive voltage)
   L = LOW state (the less positive voltage)
   X = state is immaterial

Ao ver a tabela, podemos ver que para que o 4051 funcione, é necessário aterrar o pino "E". E se aterrar os pinos de endereçamento A0,A1 e A2, irá ativar o pino Y0. E se conectar o A0 em VCC e os A1 e A2 no GND, vai ativar o pino Y1. Assim por diante.

Agora, podemos fazer o circuito com Arduino:



Aqui fizemos um teste montando com LDRS na entrada, mas você pode substituir os LDR's e os diodos por potenciometros.

Se tampar a luz no primeiro LDR da direita para esquerda, o primeiro led acenderá respectivamente. E assim por diante.

Agora, abra a IDE do Arduino e passe a seguinte programação:

```
/*
 * Example of getting 16 i/o from 5 pins using a CD4051
 *
 * Based on tutorial and code by david c. and tomek n.* for k3 / malmö högskola
 * http://www.arduino.cc/playground/Learning/4051?action=sourceblock&a...
 */


int selPin[] = { 14, 15, 16 }; // select pins on 4051 (analog A0, A1, A2)
int commonPin[] = { 17, 18};   // common in/out pins (analog A3, A4)
int led[] = {LOW, LOW, LOW, LOW, LOW, LOW, LOW, LOW };  // stores eight LED states
int CdSVal[] = { 0, 0, 0, 0 }; // store last CdS readings
int cnt = 0;  // main loop counter
int persistDelay = 100; // LED ontime in microseconds


void setup(){
  Serial.begin(9600);  // serial comms for troubleshooting (always)
  for(int pin = 0; pin < 3; pin++){ // setup select pins
    pinMode(selPin[pin], OUTPUT);
  }
}


void loop(){
  flashLEDs();
  if (cnt == 0){
    for(int x; x < 8; x++){
      led[x] = random(2);
    }
  }
  cnt++;
  if (cnt > 100) { cnt = 0; }
}


void flashLEDs() {
  for(int pin = 0; pin < 2; pin++) {  // set common pins low
    pinMode(commonPin[pin], OUTPUT);
    digitalWrite(commonPin[pin], LOW);
  }
  for (int bank = 0; bank < 4; bank++) {
    for(int pin = 0; pin < 3; pin++) { // parse out select pin bits
      int signal = (bank >> pin) & 1;  // shift  & bitwise compare
      digitalWrite(selPin[pin], signal);
    }
    if (led[bank * 2]){       // first LED
      digitalWrite(commonPin[0], HIGH);  // turn common on
      delayMicroseconds(persistDelay);   // leave led lit
      digitalWrite(commonPin[0], LOW);   // turn common off
    }
    if (led[bank * 2 + 1]){     // repeat for second LED
      digitalWrite(commonPin[1], HIGH);
      delayMicroseconds(persistDelay);
      digitalWrite(commonPin[1], LOW);
```

```
   }
  }
}
```

Conecte sua placa Arduino, selecione a versão da sua placa Arduino(UNO, Duemilanove, etc), selecione a porta(comX, ttyUSBx, ttyACMx) e lique em UPLOAD.

Pronto!! Você pode ver que ao tampar cada um dos LDR's, os leds acenderão.

E é isso aí!! Boa diversão!!! Se você tiver dúvidas, poste aqui no blog! Se tiver sugestões, poste aqui! Você pode ver nossos outros tutoriais, clicando aqui, e os projetos abertos, clicando aqui!

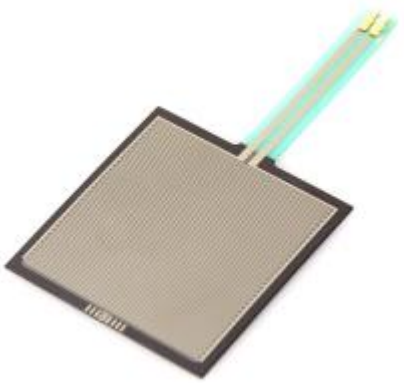
Referências:

http://dawson-station.blogspot.com.br/2010/01/mux-demux-cd4051-parl...

http://www.arduino.cc/playground/Learning/4051

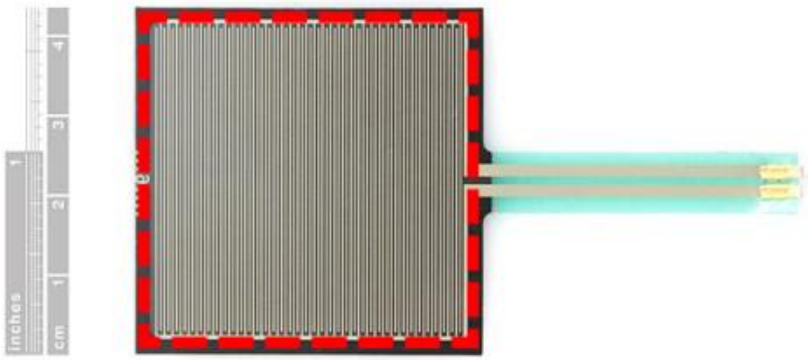
http://www.labdegaragem.org/loja/index.php/29-arduino/arduino-uno.html

5. Tutorial: Como utilizar o sensor de força resistivo com Arduino



Neste tutorial, vamos mostrar como utilizar o
sensor de força resistivo com Arduino. O sensor de força resistivo funciona como um potenciomêtro, isto é, se aplicar força na superfície, a resistência do sensor diminui. Se não tiver nada em cima do sensor, a resistência vai para mais de 20M ohm, tornando-se praticamente um circuito aberto. Este sensor consegue ler objetos com peso de 100g a 10Kg.

É necessário que a força seja aplicada dentro da área com tracejado vermelho, como está demonstrado abaixo:

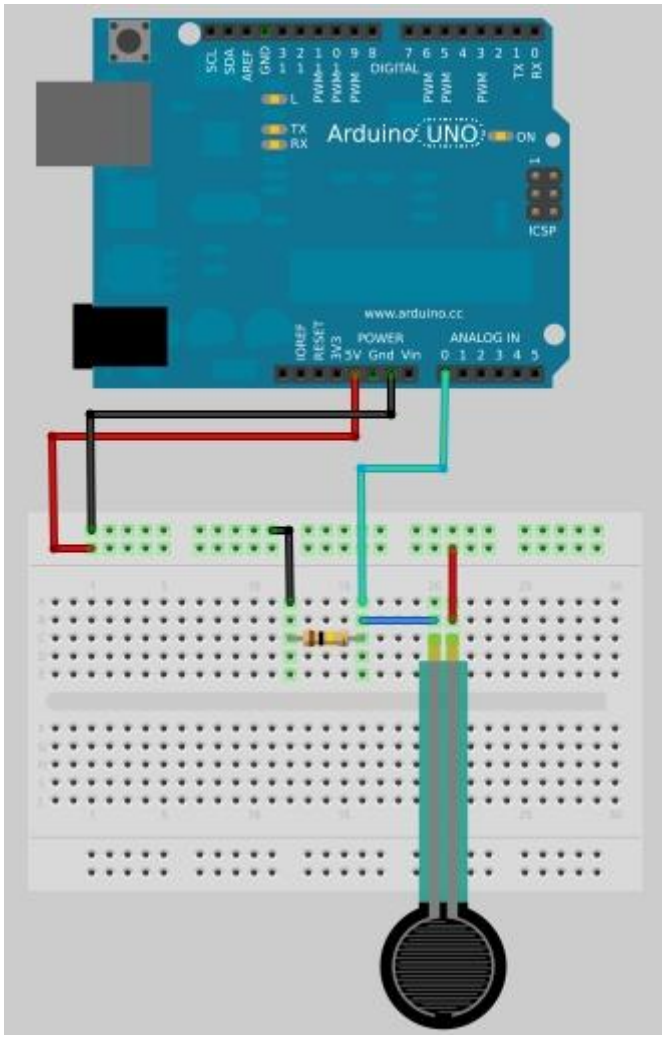


Se parte do objeto estiver fora desta área, o sensor não detetará ou dará uma leitura incorreta. Portanto é necessário que o objeto tenha, no máximo, o mesmo tamanho que o sensor. Você

pode colocar um suporte com o tamanho do sensor e depois uma plataforma para colocar objetos maiores. Cuidado, lembre-se que o sensor suporta de 100g a 10Kg.

Bom, agora que conhecemos algumas características, podemos começar! Para testar o sensor, pegue um multímetro e coloque na escala de maior resistência. Depois ligue os terminais no sensor, você verá que o multímetro não vai conseguir nenhuma leitura (maior que 20M ohm). Mas se você pressionar o sensor com o dedo, verá que uma leitura no multímetro. Mude a escala se necessário. Agora que fizemos o teste, vamos implementá-lo com Arduino.

Com um Arduino, um resistor de 100K ohm e o sensor resistivo, faça a seguinte ligação:



Agora abra a IDE do Arduino e passe a seguinte programação:

```
int Senval=0;
```

```
int Senpin=A0;
void setup()
{
Serial.begin(9600);
}
void loop()
{
Senval=analogRead(Senpin);

Serial.println(Senval);

delay(200);
}
```

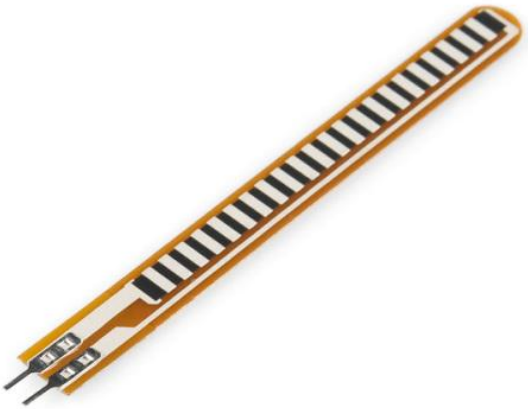
Na IDE do Arduino, selecione a sua placa Arduino (UNO, Duemilanove, etc) e a porta em que a placa Arduino está conectado (COMx, ttyusbx, ttyACMx, etc). Agora clique em "UPLOAD". Assim que terminar, abra a Serial Monitor e selecione "9600 baud" e pronto! Ao aplicar uma força no sensor, verá que o número irá variar de 0 a 1023.

E é isso!!! Esperamos que tenham gostado!! Qualquer dúvida, poste aqui mesmo neste blog! Se tiver sugestões de tutoriais,clique aqui! Você pode ver outros tutoriais, clicando aqui, e projetos abertos desenvolvidos pelo LdG e por outros garagistas,clicando aqui! Até a próxima!!

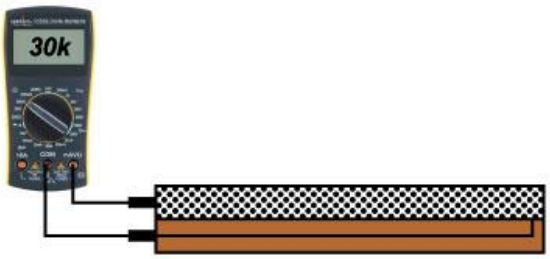
Referências:

http://www.sparkfun.com/products/9376

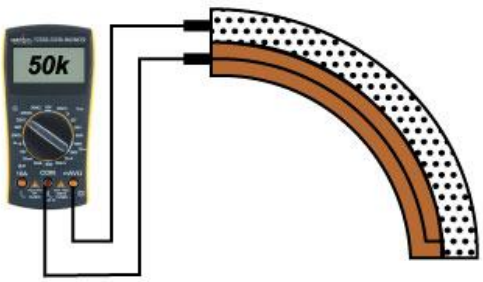
6. Tutorial sobre como utilizar o sensor flex com Arduino



Neste tutorial vamos mostrar como utilizar o sensor flex com Arduino. O sensor flex é um sensor que muda sua resistência ao ser dobrado, como mostra as imagens abaixo:



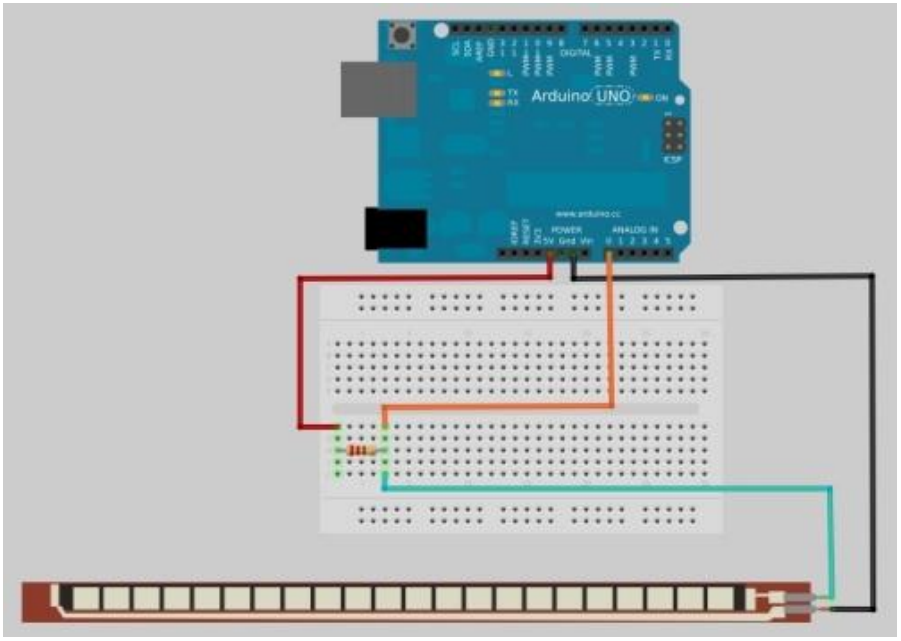
Conductive particles close together - 30K Ohms



Conductive particles further apart - 50K Ohms

Podemos ver que se manter o sensor totalmente reto, ele mostra uma resistência de 30K Ohms e caso dobrado mostra uma resistência de 50K Ohms.

Bom agora que já sabemos como ele funciona, então vamos ao circuito! Utilizando uma placa Arduino, um resistor de 10K Ohms e o sensor flex, o circuito ficará assim:



O resistor e o sensor flex estão ligados em série para fazer um divisor de tensão. A entrada analógica A0 vai entre o resistor e o sensor flex!

Abrindo a IDE do Arduino, passamos a seguinte programação exemplo que mostra quantos graus (0 a 90°) o sensor foi dobrado:

```
// Flex sensor test program
// Mike Grusin, SFE, 2011
// This program is free, use it however you wish!
// HARDWARE:
// Make the following connections between the Arduino and the flex sensor
// Note that the flex sensor pins are interchangeable
// Sensor pin - GND
// Sensor pin - Analog In 0, with 10K resistor to +5V
// INSTRUCTIONS:
// Upload this sketch to your Arduino, then activate the Serial Monitor
// (set the Serial Monitor to 9600 baud)
void setup()
{
// initialize serial communications
Serial.begin(9600);
}
void loop()
{
int sensor, degrees;

// read the voltage from the voltage divider (sensor plus resistor)
sensor = analogRead(0);

// convert the voltage reading to inches
// the first two numbers are the sensor values for straight (768) and bent (853)
// the second two numbers are the degree readings we'll map that to (0 to 90 degrees)
degrees = map(sensor, 768, 853, 0, 90);
// note that the above numbers are ideal, your sensor's values will vary
// to improve the accuracy, run the program, note your sensor's analog values
// when it's straight and bent, and insert those values into the above function.

// print out the result
Serial.print("analog input: ");
Serial.print(sensor,DEC);
```

```
Serial.print(" degrees: ");
Serial.println(degrees,DEC);

// pause before taking the next reading
delay(100);
}
```

Conecte sua placa Arduino no PC, configure a IDE para a versão da sua placa Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx,ttyACMx) e por fim clique UPLOAD.

Assim que o UPLOAD terminar, abra o Serial Monitor e configure-o para 9600 baud. Você verá que ao dobrar o sensor flex, mostrará o ângulo da dobra no Serial Monitor!!

E é isso!!! Esperamos que tenham gostado!! Caso tenham dúvidas, postem o aqui mesmo neste blog! Se tiverem sugestões de tutoriais, fiquem à vontade para sugerir aqui! E caso queiram saber mais sobre outros tutoriais já postados ou projeto desenvolvidos pelos garagistas, clique aqui e aqui respectivamente!! Até a próxima!!!

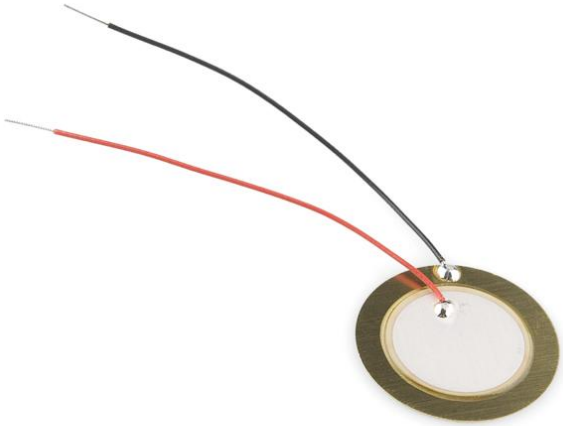
Referências:

http://www.sparkfun.com/products/10264

http://www.sparkfun.com/datasheets/Sensors/Flex/flex22.pdf

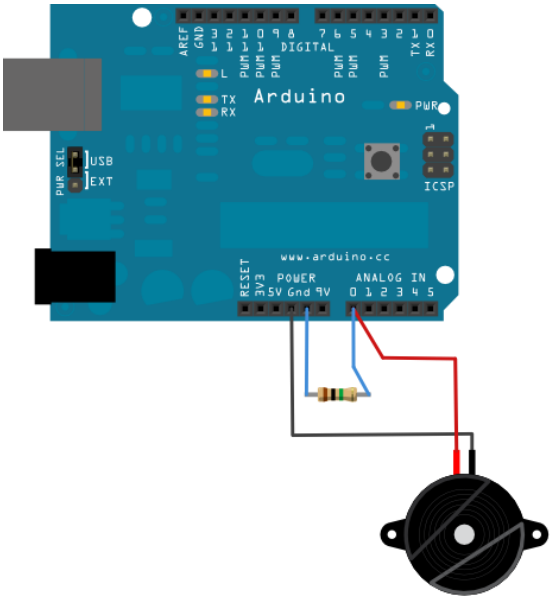
http://www.sparkfun.com/tutorials/270

7. Tutorial: como utilizar o elemento piezo com Arduino



Neste tutorial, vamos falar sobre o elemento piezo e como utilizá-lo com Arduino. Elementos piezo podem ser utilizados quando você precisa detectar a vibração ou a uma batida. Como por exemplo, torneira ou sensores de detonação. Eles também podem ser utilizados para transdutor de áudio, como uma campainha.

Para utilizar com Arduino é necessário um resistor de 1Mega Ohm em paralelo com o elemento piezo e ligá-lo na porta analógica do Arduino. As ligações estão demonstradas abaixo:



Feito as ligações, vamos a programação exemplo tirada da página: http://www.arduino.cc/en/Tutorial/Knock

```
/* Knock Sensor

This sketch reads a piezo element to detect a knocking sound.
It reads an analog pin and compares the result to a set threshold.
If the result is greater than the threshold, it writes
"knock" to the serial port, and toggles the LED on pin 13.

The circuit:
* + connection of the piezo attached to analog in 0
* - connection of the piezo attached to ground
* 1-megohm resistor attached from analog in 0 to ground
http://www.arduino.cc/en/Tutorial/Knock

created 25 Mar 2007
by David Cuartielles a href="http://www.0j0.org%3E">http://www.0j0.org>;
modified 30 Aug 2011
by Tom Igoe

This example code is in the public domain.
*/
// these constants won't change:
const int ledPin = 13; // led connected to digital pin 13
const int knockSensor = A0; // the piezo is connected to analog pin 0
const int threshold = 100; // threshold value to decide when the detected sound is a knock or not

// these variables will change:
int sensorReading = 0; // variable to store the value read from the sensor pin
int ledState = LOW; // variable used to store the last LED status, to toggle the light
void setup() {
pinMode(ledPin, OUTPUT); // declare the ledPin as as OUTPUT
Serial.begin(9600); // use the serial port
}
void loop() {
// read the sensor and store it in the variable sensorReading:
sensorReading = analogRead(knockSensor);

// if the sensor reading is greater than the threshold:
if (sensorReading >= threshold) {
// toggle the status of the ledPin:
ledState = !ledState;
// update the LED pin itself:
digitalWrite(ledPin, ledState);
// send the string "Knock!" back to the computer, followed by newline
Serial.println("Knock!");
}
delay(100); // delay to avoid overloading the serial port buffer
}
```

Agora, selecione a sua versão da sua placa Arduino (UNO, Duemilanove, etc), selecione a porta em que o Arduino está conectado (COMx, ttyUSBx, ttyACMx, etc) e faça UPLOAD.

Na programação acima, ao dar uma batida no elemento piezo, o arduino faz a leitura pela porta analógica A0 e se a leitura for maior que o threshold (>100), ele mostra no Serial Monitor a palavra "Knock!".

E é isso!! Muito simples!! Se você tiver dúvidas sobre o assunto, poste aqui mesmo no blog. Se tiver sugestões para tutoriais, você pode postar aqui! Você pode ver outros tutoriais feitos clicando aqui e se quiser ver projetos abertos disponibilizados, clique aqui! Até a próxima!

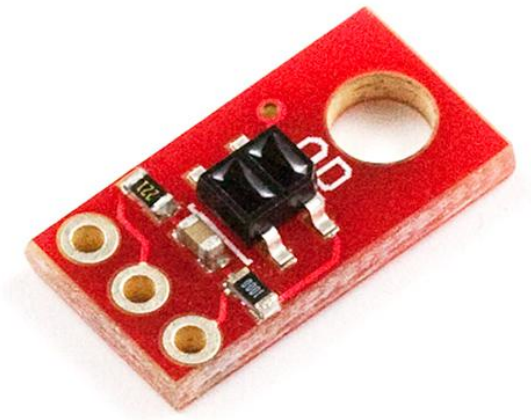
Referências:

http://www.labdegaragem.org/loja/index.php/piezo-element.html

http://www.arduino.cc/en/Tutorial/Knock

http://www.sparkfun.com/products/10293

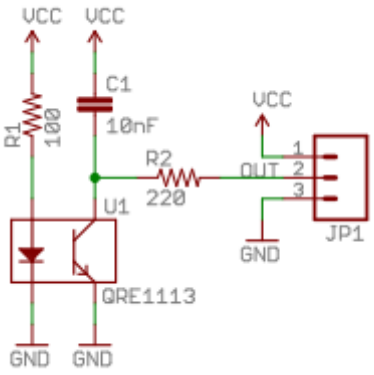
8. Tutorial: Como utilizar o Breakout de Sensor de Linha com Arduino



Neste tutorial vamos mostrar como utilizar o breakout sensor de linha com Arduino. O sensor de linha é utilizado comumente em robôs seguidores de linha ou qualquer aplicação que necessite identificar maior ou menor reflexão de luz.

Este sensor funciona com saída digital, utilizando descarga do capacitor para a reflexão do led para o fototransistor. Quanto mais rápido o capacitor descarrega, maior será a refletividade da superfície. Este breakout tem a versão digital (que será demonstrada aqui) e a versão analógica.

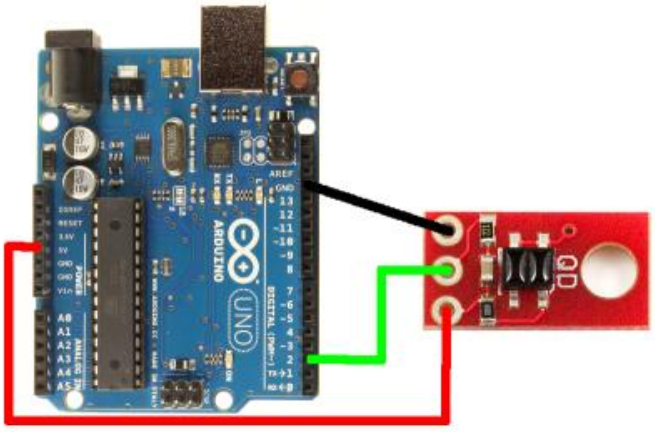
Aqui podemos ver o esquemático do breakout:



O capacitor de 10nF ao ser descarregado, faz com que haja maior ou menor refletividade no fototransistor.

Para ver o datasheet do sensor QRE1113, clique aqui!

Para poder adquirir os dados do sensor faça a seguinte ligação:



Agora, abra a IDE do Arduino e cole a seguinte programação:

```
int SensorPin=2;
int Sensorval=LOW;
void setup()
{

pinMode(13,OUTPUT);
Serial.begin(9600);
}
void loop()
{

pinMode(2,OUTPUT);       // Primeiro é necessário configurar o pino do Arduino (digital 2) que
conecta o Vout do
digitalWrite(2,HIGH);       //breakout como saída para descarregar o capacitor, colocando o
pino digital 2 do Arduino
delayMicroseconds(10);    // em alto (HIGH) e depois uma espera de 10 microssegundos
pinMode(2,INPUT);          // E por fim configura o pino do Arduino (digital 2) como entrada
long time = micros();
// Enquanto o tempo for menor que 3000 microssegundos e o pino do sensor for alto (HIGH), e
ntão o valor do sensor
// será a diferença entre o tempo atual e o tempo anterior.
while (digitalRead(SensorPin) == HIGH && micros() - time < 3000);
int diff = micros() - time;
Sensorval=diff;
if(Serial.available()>0);
{
Serial.println(Sensorval);
}
delay(500);
}
```

Pronto!! Esperamos que tenham gostado do tutorial! Caso tenham dúvidas sobre o sensor de linha, postem aqui mesmo no blog! Se tiverem sugestões, vocês podem postar aqui!Para ver

nossos outros tutoriais e projetos aberto desenvolvidos por garagistas e pela equipe LdG, cliquem aqui e aqui, respectivamente!

Referências:

http://www.labdegaragem.org/loja/index.php/breakout-de-sensor-de-li...

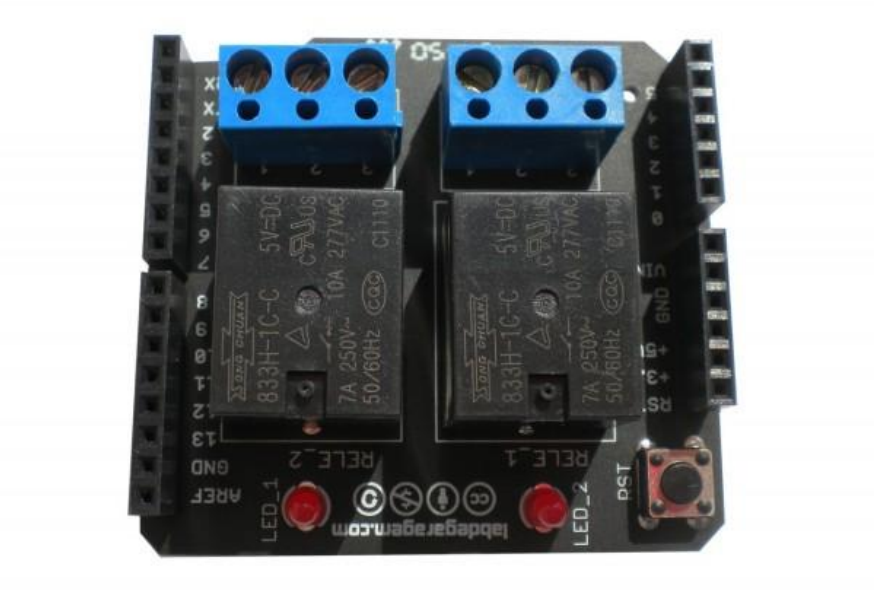
http://www.sparkfun.com/products/9454

http://www.sparkfun.com/datasheets/Robotics/QRE1113-Digital-Breakou...

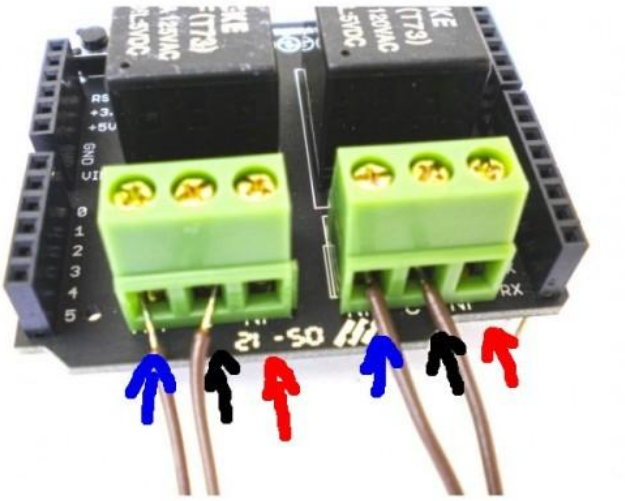
http://bildr.org/2011/06/qre1113-arduino/

http://www.sparkfun.com/datasheets/Robotics/QR_QRE1113.GR.pdf

9. Tutorial: como utilizar o ReleShield com Arduino

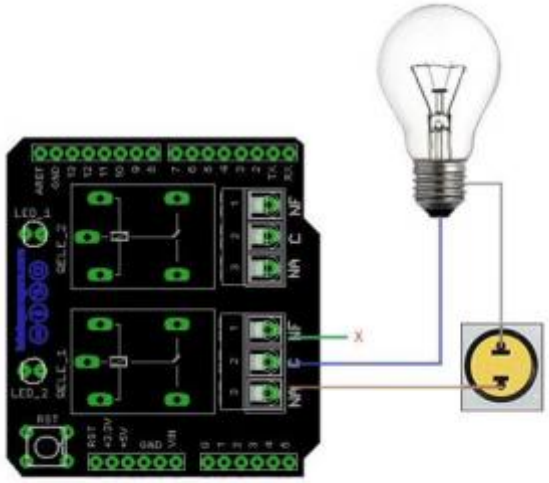


O ReleShield é um shield, disponível na loja LdG desenvolvido pelo garagista Maurício Ortega, com dois Relés para acionamento de cargas com correntes maiores que o Arduino pode fornecer e também isolação entre o Arduino e a carga. Isto é, permite que o Arduino acione, por exemplo, uma lâmpada de 110V ou até mesmo um equipamento doméstico (aparelho de som, ventilador, etc). Cada relé contém três entradas: uma NF (Normalmente fechado) que está marcado em vermelho, uma NA (Normalmente aberto) que está marcado em azul e o Comum que está marcado em preto na figura abaixo.



Neste tutorial, vamos mostrar como utilizá-lo com a programação exemplo que vem na documentação encontrada na loja LdG.

Primeiramente, vendo sua documentação podemos ver que o Releshield utiliza os pinos digitais 7 e 8. Então vamos conectar o Releshield no Arduino e vamos fazer o primeiro exemplo disponível. Faça a seguinte ligação:



OBS: Caso o equipamento está dando interferência no Arduino, conecte os filtros fornecidos nos contatos do Releshield, neste caso, entre o Comum e o NA.

Agora abrindo a IDE do Arduino, vamos colocar a programação exemplo:

```
#define Rele1 7 // Define pino de saida para rele 1
#define Rele2 8 // Define pino de saida para rele 2
void setup(){
pinMode(Rele1,OUTPUT);
pinMode(Rele2,OUTPUT);
}
void loop(){
digitalWrite(Rele1, HIGH);
digitalWrite(Rele2, HIGH);
delay(1000);
digitalWrite(Rele1, LOW);
digitalWrite(Rele2, LOW);
delay(1000);
}
```

Agora conecte seu Arduino, e na IDE do Arduino, selecione a versão do seu Arduino (UNO, Duemilanove, etc) e a porta em que ele está conectado (COMx, ttyUSBx, ttyACMx). E clique em UPLOAD. Você verá os leds e a lâmpada acenderem e apagarem, isto quer dizer que o contato NA está fechado e o contato NF está aberto do Releshield.

E é isso! Esperamos que tenha gostado! Caso tenha dúvidas, poste aqui neste blog! Se tiver sugestões para tutoriais, poste aqui. Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente.

Referências:

http://www.labdegaragem.org/loja/index.php/31-shields/releshield.html

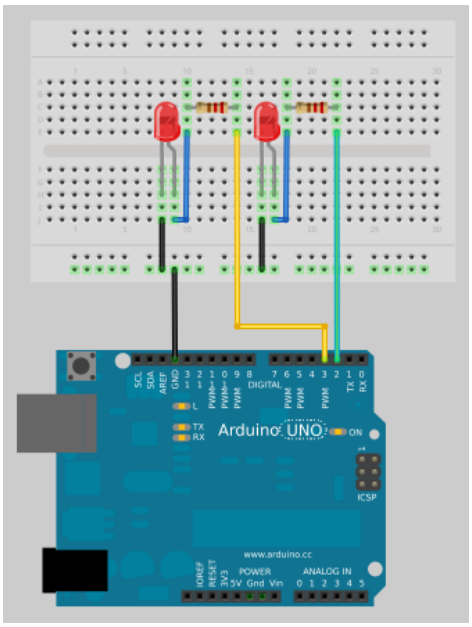
http://www.labdegaragem.com.br/loja/releshield.pdf

http://www.labdegaragem.com.br/loja/emi.pdf

10. Tutorial: como acender leds pelo serial monitor com Arduino

Neste tutorial, vamos mostrar como acender e apagar leds digitando dados pelo Serial Monitor com Arduino. A programação é simples e bem interessante, pois com a mesma tecla, você liga e desliga o led. É uma aplicação simples mas muito útil em qualquer aplicação, seja acender apenas um led ou ligar um equipamento.

Primeiramente, faça as ligações como mostrada abaixo:



Agora, abra a IDE do Arduino e passe a seguinte programação:

```
char c;
void setup(){
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
Serial.begin(9600);
}

void loop()
{
if (Serial.available()>0)
{
c = Serial.read() - '0';
Serial.flush();
digitalWrite(c,!digitalRead(c));
}
}
```

Conecte o Arduino no PC, selecione a versão do seu Arduino (UNO, Duemilanove, etc) e porta (COMx, ttyUSBx, ttyACMx) e clique em UPLOAD. Abra o Serial Monitor e selecione 9600 no baud.

Ao digitar o número 2 no Serial Monitor, irá acender o LED conectado na porta digital 2 do Arduino, e ao digitar novamente o número 2, o LED apagará. Agora se digitar o número 3, irá acender o LED da porta digital 3 e se digitar novamente o número 3, o LED apagará.

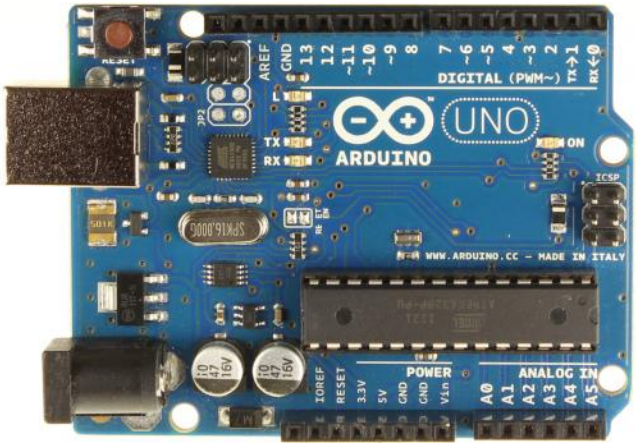
E é isso! Bem simples! Esperamos que tenha gostado! Se tiver dúvidas, poste aqui neste blog! Para sugestões de tutoriais,clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/29-arduino.html

http://arduino.cc/en/

11. Tutorial: o que é e como utilizar o PWM no Arduino



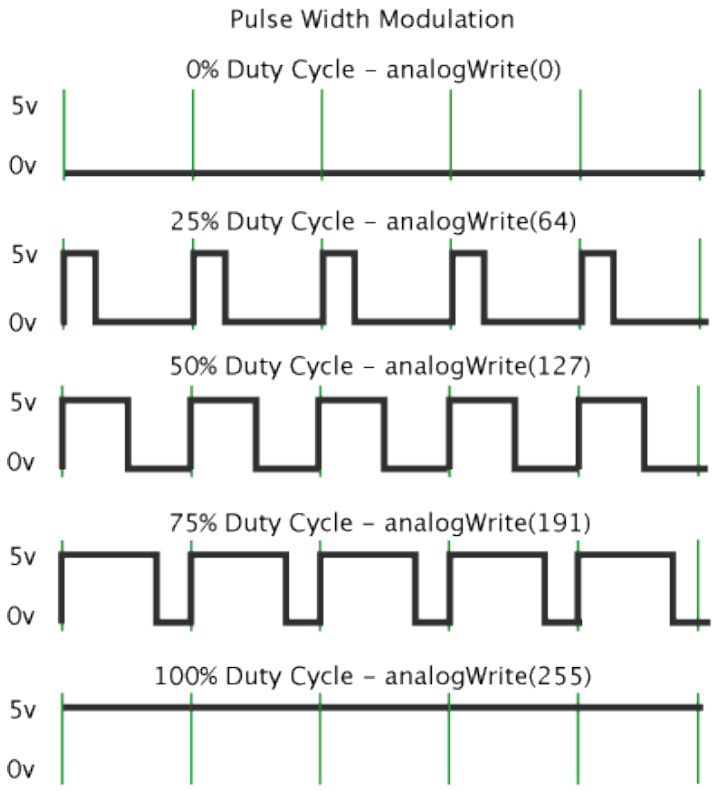
Neste tutorial vamos explicar o que é o PWM (Pulse Width Modulation) e como utilizá-lo no Arduino.

A modulação por Largura de Pulso ou PWM é a modulação ou alteração da largura do pulso de um sinal de onda quadrada que pode ser dados à ser transmitido, um efeito de áudio ou potência fornecida ou regulada.

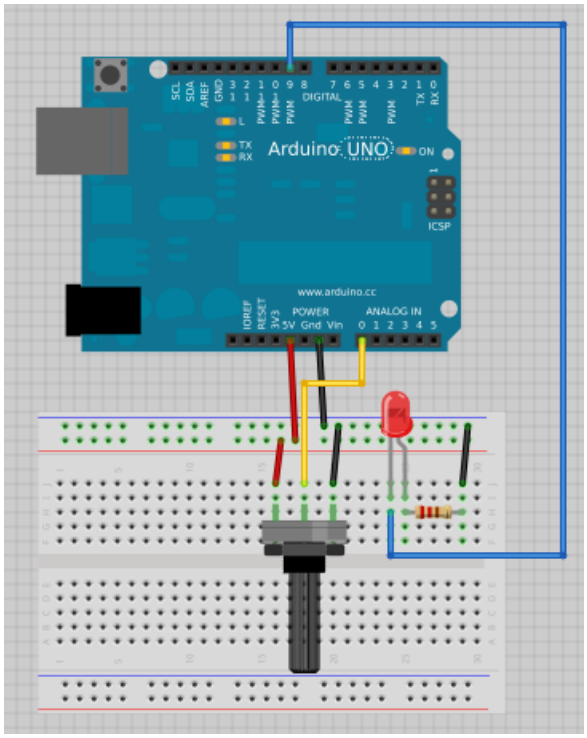
O Arduino contém 6 PWM's que estão destacadas com um "~" na frente do número referente a porta digital. Com estas entradas pode-se obter resultados analógicos por meio de sinal digital.

O controle digital cria ondas quadradas com pulsos de largura variável, podendo assim dar uma tensão média à carga, por exemplo.

A figura abaixo mostra as formas de onda mais usuais:



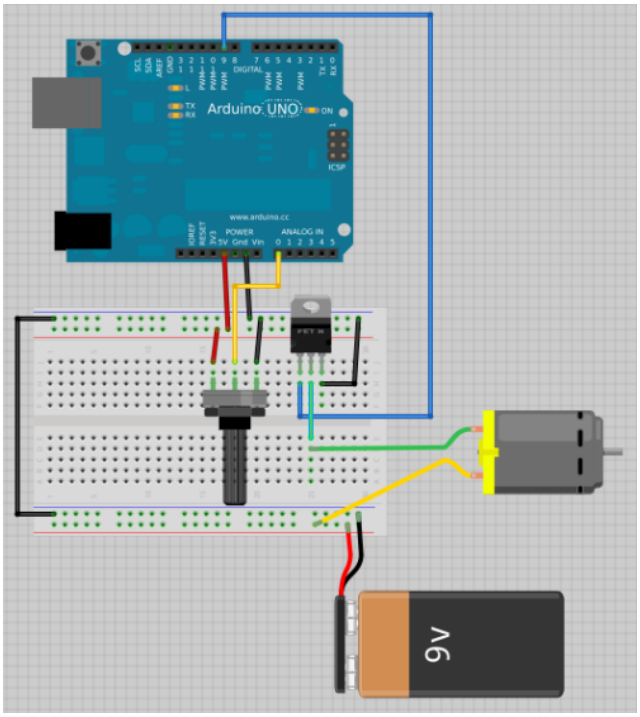
Como podemos ver, o que muda é a largura do pulso como descrito anteriormente. Por exemplo, digitando apenas analogWrite(Pino Digital, 127) tem-se uma largura de pulso de 50% da onda quadrada. Vamos para um exemplo:



Fazendo a ligação como da figura acima. Ao girar o potenciômetro, o Led acenderá e apagará lentamente. Esta é uma aplicação o qual você pode controlar o fornecimento de potência à carga. A programação desta ligação está mostrada abaixo:

```
int Ledpin=9;
int analogpin=A0;
int value=0;
int x;
void setup()
{
Serial.begin(9600);
pinMode(Ledpin, OUTPUT);
}
void loop()
{
value=analogRead(analogpin);
x=map(value,0,1023,0,255);
analogWrite(Ledpin,x);
Serial.println(x);
delay(100);
}
```

Na programação, o Arduino recebe o valor analógico do potenciômetro, converte para o sinal de acordo com o PWM e depois manda para a porta digital "9". Agora, para ligar um motor ou componente que necessite de mais potência ou corrente, faça a seguinte ligação:



A programação é a mesma e não precisa de mudança. A bateria de 9V é apenas para exemplo, verifique a tensão de alimentação do motor ou da carga para evitar queimar seu componente. O funcionamento do circuito é idêntico, porém, ao girar o potenciômetro, o motor aumentará ou diminuirá a rotação.

E é isso! Esperamos que tenha gostado!! Se tiver dúvidas referente ao tutorial, poste aqui neste blog! Se tiver uma sugestão de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG ou por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://arduino.cc/en/Tutorial/PWM

http://arduino.cc/en/Reference/AnalogWrite

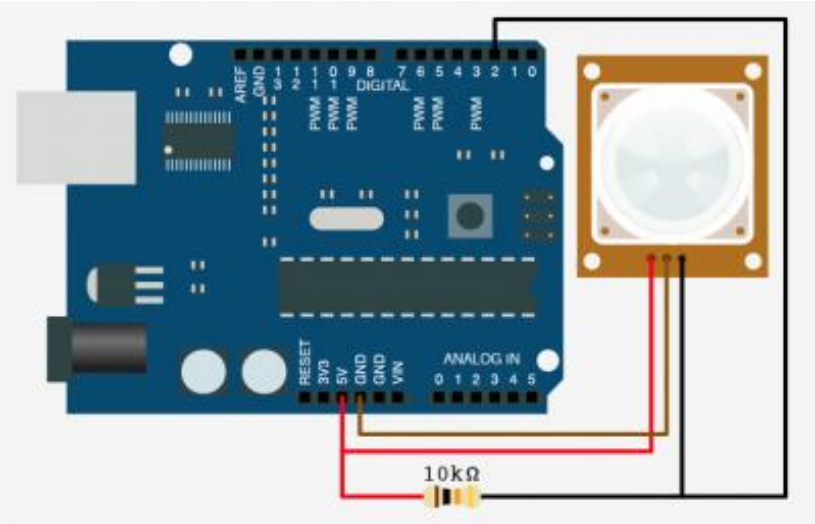
http://pt.wikipedia.org/wiki/Modula%C3%A7%C3%A3o_por_largura_de_pulso

12. Tutorial: como utilizar o sensor PIR com Arduino

Neste tutorial vamos mostrar como utilizar o sensor PIR (Passive Infrared) com Arduino. O sensor PIR é um sensor de movimento por calor (infravermelho). O sensor ativa o alarme assim que detecta uma mudança no infravermelho em um determinado lugar. Portanto, se uma pessoa se movimentar na frente do sensor PIR, este irá detectar a mudança e ativará o alarme.

OBS: A pinagem abaixo está incorreta! Assim como no datasheet! Siga a seguinte pinagem em vermelho: Marrom - GND, Preto - VCC (5V), Vermelho - Saída Digital.

O sensor PIR tem três fios: GND (marrom), VCC (vermelho) e Saída (preto). A figura abaixo mostra a ligação a ser feita no Arduino:



Para que o sensor PIR funcione corretamente é necessário um resistor de 10Kohm como Pull-Up. Para saber mais sobre resistores de Pull-Up, clique aqui.

A programação está mostrada abaixo:

```
int pirPin = 2; //digital 2
int LedPin = 13;
void setup(){
Serial.begin(9600);
pinMode(pirPin, INPUT);
pinMode(LedPin,OUTPUT);
}
void loop(){
int pirVal = digitalRead(pirPin);
if(pirVal == LOW){ //was motion detected
digitalWrite(LedPin,HIGH);
delay(2000);
}
else
{
digitalWrite(LedPin,LOW);
}
}
```

Abra a IDE do Arduino e passe a programação acima. Selecione a versão do seu Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD.

Ao passar a mão na frente do sensor PIR, o Led da placa Arduino acenderá. Caso contrário ficará apagado.

E é isso! Esperamos que tenha gostado! Se tiver dúvidas sobre o tutorial, poste aqui neste blog! Se tiver sugestão para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui eaqui, respectivamente! Até a próxima!

Referências:

http://www.sparkfun.com/products/8630

http://www.sparkfun.com/datasheets/Sensors/Proximity/SE-10.pdf

http://bildr.org/2011/06/pir_arduino/

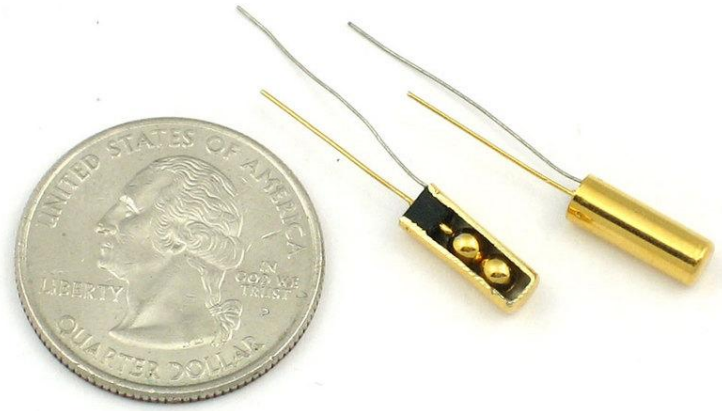
http://www.ladyada.net/learn/sensors/pir.html

13. Tutorial: como utilizar o sensor Tilt com Arduino

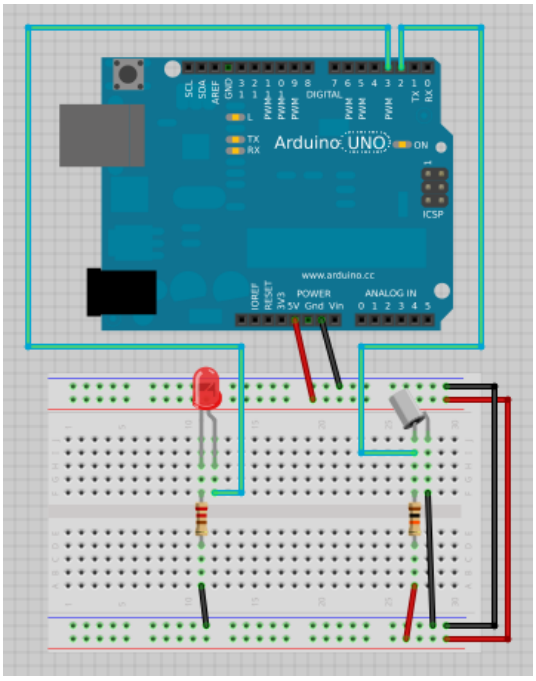
Neste tutorial vamos mostrar como utilizar o Sensor Tilt com Arduino.

O Sensor Tilt é um sensor que detecta orientação ou inclinação. Ele é pequeno, baixo consumo e de fácil utilização.

Dentro do sensor existe duas bolinhas que ao se tocarem, terá continuidade nos contatos (curto-circuito). A imagem abaixo mostra o sensor Tilt internamente:



Sabendo seu funcionamento, vamos para a ligação com o Arduino:



A programação para Arduino está mostrada abaixo:

```
int SensorPin = 2;
int LEDPin = 3;

int LEDstate = HIGH;
int reading;
int previous = LOW;

long time = 0;
long debounce = 50;

void setup()
{
pinMode(SensorPin, INPUT);
digitalWrite(SensorPin, HIGH);
pinMode(LEDPin, OUTPUT);
}

void loop()
{
int switchstate;

reading = digitalRead(SensorPin);


if (reading != previous) {

time = millis();
}

if ((millis() - time) > debounce) {

switchstate = reading;
```

```
if (switchstate == HIGH)
LEDstate = LOW;
else
LEDstate = HIGH;
}
digitalWrite(LEDPin, LEDstate);

previous = reading;
}
```

Abra a IDE do Arduino e passe a programação acima! Selecione a versão da sua PLaca Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD.

Ao inclinar o sensor Tilt para baixo, o led do pino digital 3 irá acender.

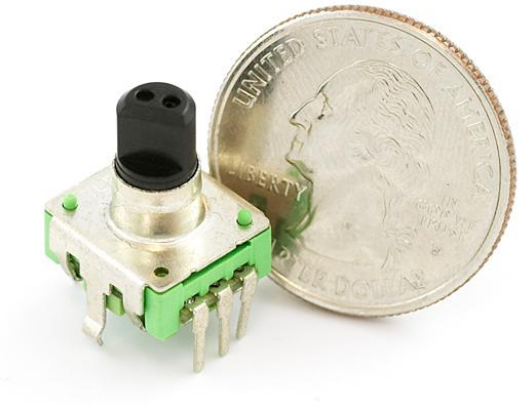
Eé isso! Esperamos que tenha gostado! Se tiver dúvidas sobre o tutorial, poste aqui neste blog! Caso tenha sugestões para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LDG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.sparkfun.com/products/10289

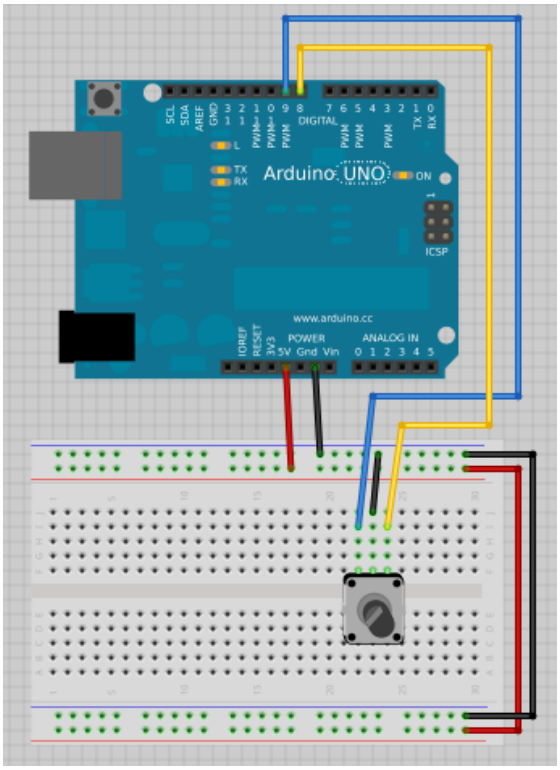
http://learn.adafruit.com/tilt-sensor/overview

14. Tutorial: como utilizar encoder rotativo com Arduino



O encoder rotativo é um dispositivo de medição angular. Com ele é possível medir com precisão a rotação de motores. Alguns deles contém um botão pressionando o eixo. Neste tutorial, vamos mostrar como utilizá-lo com Arduino. Este encoder rotativo contém 7 pinos: dois para fixação, dois para o botão e três para medição de rotação. Vamos apenas utilizar os três pinos para medição de rotação. Para maiores informações sobre o encoder, clique aqui!

Primeiramente, faça a ligação abaixo:

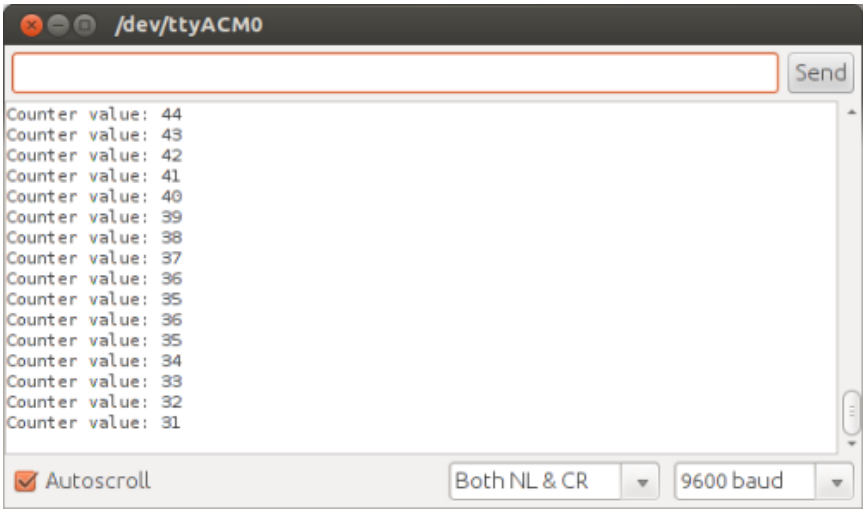


Abra a IDE do Arduino e conecte seu Arduino no PC. Passe a programação abaixo:

```
/* Rotary encoder read example */
#define ENC_A 8
#define ENC_B 9
#define ENC_PORT PINB
void setup()
{
/* Setup encoder pins as inputs */
pinMode(ENC_A, INPUT);
digitalWrite(ENC_A, HIGH);
pinMode(ENC_B, INPUT);
digitalWrite(ENC_B, HIGH);
Serial.begin (9600);
Serial.println("Start");

}

void loop()
{
static uint8_t counter = 0; //this variable will be changed by encoder input
int8_t tmpdata;
/**/
tmpdata = read_encoder();
if( tmpdata ) {
Serial.print("Counter value: ");
Serial.println(counter, DEC);

counter += tmpdata;

}

}

/* returns change in encoder state (-1,0,1) */
int8_t read_encoder()
{
static int8_t enc_states[] = {0,-1,1,0,1,0,0,-1,-1,0,0,1,0,1,-1,0};
static uint8_t old_AB = 0;
/**/
old_AB = old_AB 2; //remember previous state
old_AB |= ( ENC_PORT & 0x03 ); //add current state
return ( enc_states[( old_AB & 0x0f )]);
}
```

Selecione a versão do seu Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD.

Abra o Serial Monitor e configure o baud para 9600. A palavra "Start" aparecerá. Gire o encoder para um lado e para o outro. A imagem abaixo mostra os dados que chegam do encoder:



Ao girar o encoder, o Arduino detecta a variação de pulso (-1,0,1) que chega dos pinos 8 e 9 e os converte em inteiro e mostra no Serial Monitor.

E é isso! Esperamos que tenha gostado! Se tiver dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui! Até a próxima!

Referências:

http://en.wikipedia.org/wiki/Rotary_encoder

http://www.circuitsathome.com/mcu/programming/reading-rotary-encode...

http://www.sparkfun.com/products/9117

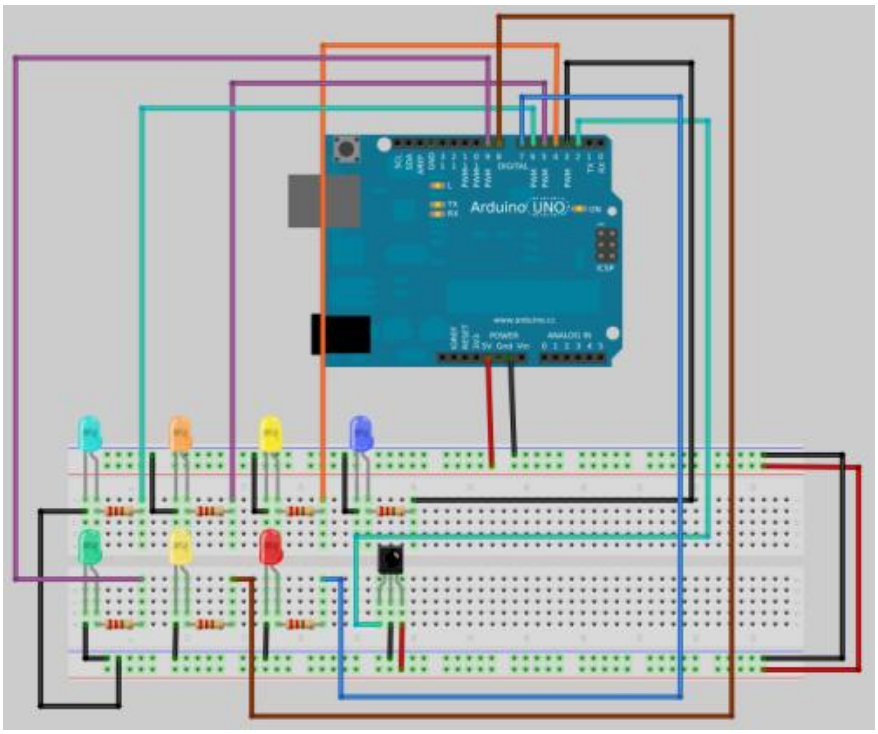
http://arduino.cc/playground/Main/RotaryEncoders

15. Tutorial: como utilizar o controle remoto IR com Arduino

Neste tutorial vamos mostrar como utilizar o IR Control Kit da Sparkfun. O Kit vem com dois emissores Infravermelho, dois receptores Infravermelho e o Controle Remoto. Ao apertar um botão do controle remoto, um LED acenderá. É um LED para cada botão do controle remoto.

Você pode utilizar também a biblioteca IRRemote.

Faça a seguinte ligação no Arduino:



Depois de feito as ligações acima, passe a programação abaixo:

```
int irPin = 2; //Sensor pin 1 wired to Arduino's pin 2
int statLED = 3;
int statLED1 = 4;
int statLED2 = 5;
int statLED3 = 6;
int statLED4 = 7;
int statLED5 = 8;
int statLED6 = 9; //Toggle the status LED every time Power is pressed
int start_bit = 2200; //Start bit threshold (Microseconds)
int bin_1 = 1000; //Binary 1 threshold (Microseconds)
int bin_0 = 400; //Binary 0 threshold (Microseconds)
void setup() {
pinMode(statLED, OUTPUT);
pinMode(statLED1, OUTPUT);
pinMode(statLED2, OUTPUT);
pinMode(statLED3, OUTPUT);
pinMode(statLED4, OUTPUT);
pinMode(statLED5, OUTPUT);
pinMode(statLED6, OUTPUT);
digitalWrite(statLED, LOW);
digitalWrite(statLED1, LOW);
digitalWrite(statLED2, LOW);
digitalWrite(statLED3, LOW);
digitalWrite(statLED4, LOW);
digitalWrite(statLED5, LOW);
digitalWrite(statLED6, LOW);
pinMode(irPin, INPUT);
Serial.begin(9600);
Serial.println("Waiting: ");
}
void loop() {
int key = getIRKey(); //Fetch the key

if(key != 0) //Ignore keys that are zero
{
Serial.print("Key Recieved: ");
switch(key)
{
case 144: Serial.print("CH Up");
if(digitalRead(statLED) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED, HIGH);
else
digitalWrite(statLED, LOW);
break;

case 145: Serial.print("CH Down");
if(digitalRead(statLED1) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED1, HIGH);
else
digitalWrite(statLED1, LOW);
break;

case 146: Serial.print("VOL Right");
if(digitalRead(statLED2) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED2, HIGH);
else
digitalWrite(statLED2, LOW);
break;
case 147: Serial.print("VOL Left");
```

```
if(digitalRead(statLED3) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED3, HIGH);
else
digitalWrite(statLED3, LOW);
break;
case 148: Serial.print("Mute");
if(digitalRead(statLED4) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED4, HIGH);
else
digitalWrite(statLED4, LOW);
break;
case 165: Serial.print("AV/TV");
if(digitalRead(statLED5) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED5, HIGH);
else
digitalWrite(statLED5, LOW);
break;
case 149:
Serial.print("Power");
if(digitalRead(statLED6) == LOW) //This toggles the statLED every time power button is hit
digitalWrite(statLED6, HIGH);
else
digitalWrite(statLED6, LOW);
break;
}

Serial.println();
}
}
int getIRKey() {
int data[12];
int i;
while(pulseIn(irPin, LOW) < start_bit); //Wait for a start bit

for(i = 0 ; i < 11 ; i++)
data[i] = pulseIn(irPin, LOW); //Start measuring bits, I only want low pulses

for(i = 0 ; i < 11 ; i++) //Parse them
{
if(data[i] > bin_1) //is it a 1?
data[i] = 1;
else if(data[i] > bin_0) //is it a 0?
data[i] = 0;
else
return -1; //Flag the data as invalid; I don't know what it is! Return -1 on invalid data
}
int result = 0;
for(i = 0 ; i < 11 ; i++) //Convert data bits to integer
if(data[i] == 1) result |= (1 i);
return result; //Return key number
}
```

A função int getIRKey() conta o tempo do pulso em LOW que o controle remoto transmite. Para cada botão é um número de pulsos diferente. Portanto, ao apertar o botão POWER, por exemplo, o Arduino detectará LOW, fará a contagem em microssegundos e guardará na matriz data[i]. Depois o Arduino fará uma comparação de todos os elementos da matriz com as constantes bin_1 e bin_0 e transfomará os dados em binário (0 ou 1). Por fim converterá os dados em bits para inteiro. Ao terminar, o Arduino volta para o loop() e verifica cada caso, acendendo o LED correspondente ao dado inteiro que foi convertido.

E é isso! Esperamos que tenha gostado! Se tiver dúvidas sobre o tutorial, poste aqui neste blog! Em caso de sugestões, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/32-kits/ir-control-kit-r...

http://arduino.cc/en/

http://labdegaragem.com/profiles/blogs/tutorial-emissor-e-receptor-...

http://www.arcfn.com/2009_08_01_archive.html

http://www.sparkfun.com/tutorials/291

16. Tutorial sobre funções de tempo no Arduino

No Arduino existem várias formas de se utilizar o tempo, como as funções delay's, millis(), micros(), bibliotecas e periféricos as quais podem ser encontradas na internet, seja no site do Arduino, ou em outros sites sobre arduino. Neste tutorial, vamos falar um pouco sobre Timer e estas funções.

As funções delay(ms) e delayMicroseconds(us) são funções de atraso ou espera, ao chamar estas funções elas irão contar o tempo que foi determinado em milissegundos(ms) ou microssegundos(us). Por exemplo: delay(500), o arduino irá contar até 500 milissegundos e depois irá para próxima instrução. O delayMicroseconds(500), o arduino fará o mesmo que o delay, mas em microssegundos. O problema de utilizar estas funções é que elas "pausam" a programação até terminar o tempo que foi determinado.

Com a função millis() o arduino retorna o tempo a partir do momento em que o Arduino roda a programação. Esta função é melhor que a delay, porque o millis() não "pausa" a programação para fazer a contagem. Vamos pegar o exemplo BlinkWithoutDelay:

```
/* Blink without Delay

Turns on and off a light emitting diode(LED) connected to a digital
pin, without using the delay() function. This means that other code
can run at the same time without being interrupted by the LED code.

The circuit:
* LED attached from pin 13 to ground.
* Note: on most Arduinos, there is already an LED on the board
that's attached to pin 13, so no hardware is needed for this example.

created 2005
by David A. Mellis
modified 8 Feb 2010
by Paul Stoffregen

This example code is in the public domain.

http://www.arduino.cc/en/Tutorial/BlinkWithoutDelay
*/

// constants won't change. Used here to
// set pin numbers:
```

```
const int ledPin = 13; // the number of the LED pin
// Variables will change:
int ledState = LOW; // ledState used to set the LED
long previousMillis = 0; // will store last time LED was updated
// the follow variables is a long because the time, measured in miliseconds,
// will quickly become a bigger number than can be stored in an int.
long interval = 1000; // interval at which to blink (milliseconds)
void setup() {
// set the digital pin as output:
pinMode(ledPin, OUTPUT);
}
void loop()
{
// here is where you'd put code that needs to be running all the time.
// check to see if it's time to blink the LED; that is, if the
// difference between the current time and last time you blinked
// the LED is bigger than the interval at which you want to
// blink the LED.
unsigned long currentMillis = millis();

if(currentMillis - previousMillis > interval) {
// save the last time you blinked the LED
previousMillis = currentMillis;
// if the LED is off turn it on and vice-versa:
if (ledState == LOW)
ledState = HIGH;
else
ledState = LOW;
// set the LED with the ledState of the variable:
digitalWrite(ledPin, ledState);
}
}
```

Na linha: "unsigned long currentMillis = millis();", estamos armazenando o número em milissegundos que a função millis() retornou, na variável currentMillis.

Na linha "if(currentMillis - previousMillis > interval)", estamos declarando que se a subtração do tempo atual(currentMillis) com o tempo anterior(previousMillis) for maior que o intervalo de tempo(interval) determinado, então faça as instruções... As instruções declaradas dentro deste "if" fazem o LED do pino 13 acender e apagar dentro do intervalo determinado.

A função micros() tem a mesma funcionalidade que a millis(), mas só que micros() faz a contagem de tempo em microssegundos.

Apesar dessas funções serem muito funcionais, elas não são muito precisas ou zeram depois de um certo tempo por causa do atmega e do cristal de 16Mhz. Agora para uma maior precisão de tempo e obter dados em tempo real(Real Time Clock) por exemplo, existe várias bibliotecas disponíveis como Time que pode ser usado com ou sem hardware externo e esta biblioteca que utiliza o DS1307 para tempo real(Real Time Clock). As duas bibliotecas são utilizadas para contar e mostrar o tempo real como horas, dia, mês e ano.

E é isso aí!! Esperamos que tenham gostado!! Se tiverem dúvidas, postem aqui neste blog! Caso tenham sugestões de tutoriais, postem aqui neste POST!! Vocês podem ver os outros tutoriais clicando AQUI e os projetos clicando AQUI!! Até a próxima!!

Referências:

http://arduino.cc/playground/Main/InterfacingWithHardware#time

http://www.arduino.cc/playground/Code/Time

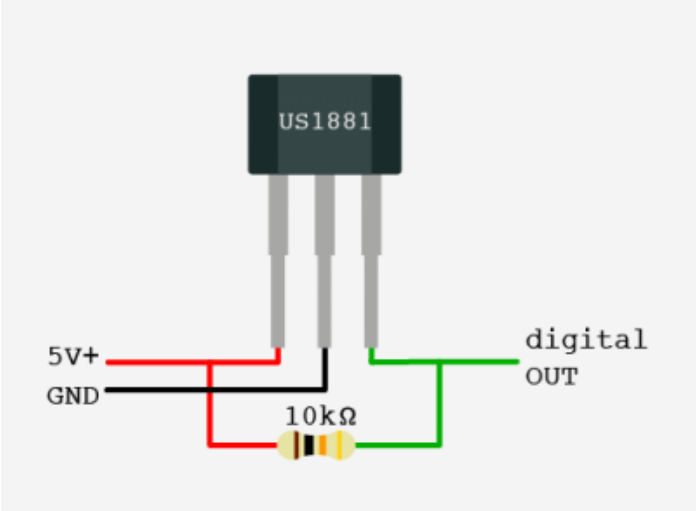
https://github.com/davidhbrown/RealTimeClockDS1307

17. Tutorial: como utilizar o Sensor de efeito Hall com Arduino

Neste tutorial vamos mostrar como utilizar o sensor de efeito Hall com Arduino.

O sensor de efeito Hall é um sensor que detecta o campo magnético de objetos magnéticos (imãs).Este pode servir para detecção de objetos magnéticos (imãs). Este sensor utiliza o modo liga/desliga. Para ligá-lo é apenas aproximar o pólo Norte de um objeto magnético (imã) e para desligá-lo é só aproximar o pólo Sul.

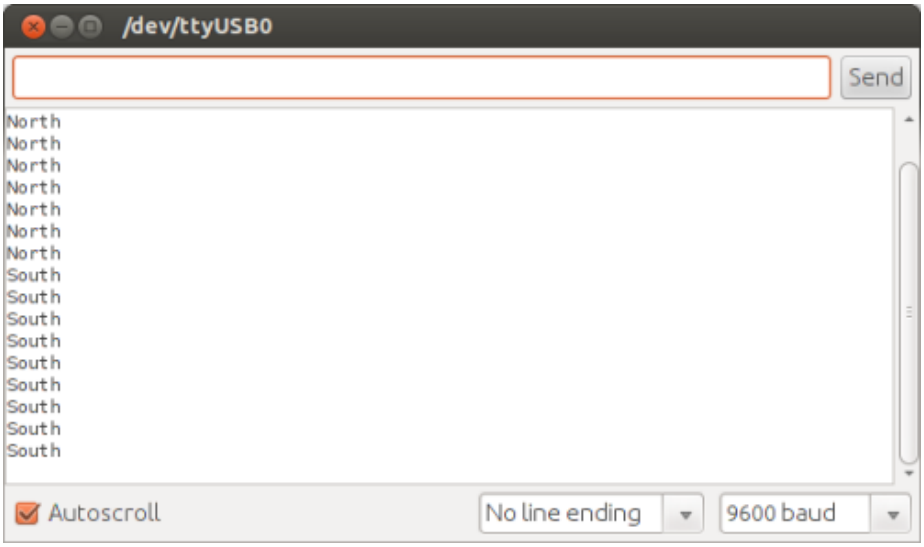
Primeiramente faça a seguinte ligação:



Conecte o pino "digital out" do sensor de efeito Hall no pino digital 13 do Arduino. Agora abra a IDE do Arduino e passe e seguinte programação:

```
int hallPin=13;
int statePin=LOW;

void setup()
{
pinMode(hallPin,INPUT);
Serial.begin(9600);
}
void loop()
{
statePin=digitalRead(hallPin);
if (Serial.available())
{
if( statePin==HIGH)
{
Serial.println("North");
}
else if(statePin==LOW)
{
Serial.println("South");
}
}
delay(500);
}
```

Conecte seu Arduino no PC, selecione a versão do seu Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD. Ao terminar de fazer UPLOAD, abra o Serial Monitor. Digite qualquer tecla para ativar o Serial Monitor. Agora pegue um imã e aproxime-o do sensor. Ao aproximar o imã, o Serial Monitor irá mostrar se é o pólo Norte ou o pólo Sul do imã. A figura abaixo mostra o Serial Monitor com a variação:



E é isso! Esperamos que tenha gostado! Caso tiver dúvidas, poste neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/34-sensores/sensor-de-ef...

https://www.sparkfun.com/products/9312?

http://bildr.org/2011/04/various-hall-effect-sensors/

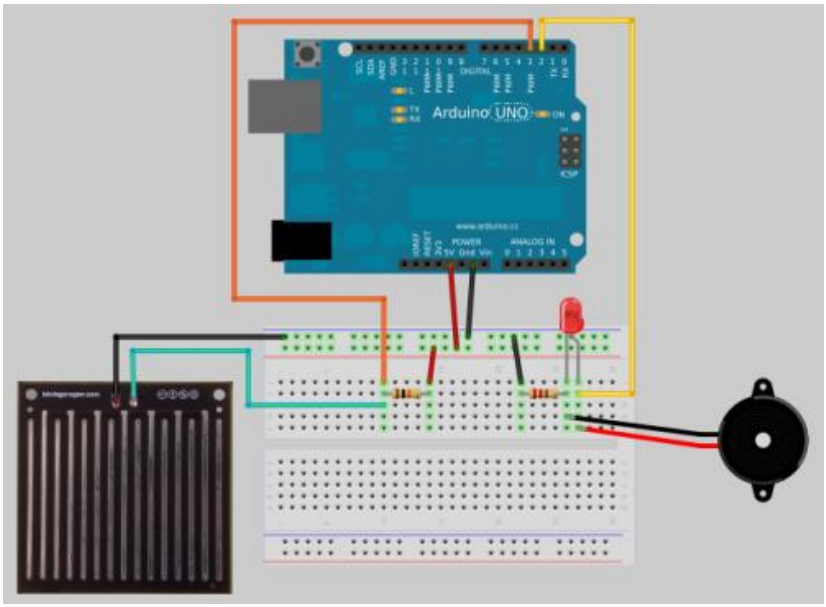
18. Tutorial: Como utilizar o Sensor de Chuva com Arduino

O sensor de chuva é um produto desenvolvido pelo garagista William Lima com intuito de detectar chuva, onde poderá ser utilizado em automação residencial, como fechamento de janelas, fechamento de teto solar, etc. Ao cair água em cima do sensor, este irá entrar em curto-circuito e o Arduino apitará o buzzer e acenderá o led.

Para fazer a ligação mostrada no vídeo, você precisará:

- 1x Arduino
- 1x Sensor de chuva LdG
- 1x Resistor de 220 ohm
- 1x Resistor de 10K ohm
- 1x Led
- 1x Buzzer

Abaixo mostra a ligação utilizada no vídeo:



E a programação utilizada no vídeo está mostrado abaixo:

```
/* Exemplo do Sensor de Chuva
Equipe LdG
*/
int bip = 2;
int sensordechuva = 3;
void setup()
{
pinMode(bip,OUTPUT);
pinMode(sensordechuva,INPUT);
}
void loop()
{
if(digitalRead(sensordechuva) == 0)
{
digitalWrite(bip,HIGH);
delay(500);
digitalWrite(bip,LOW);
}
else
{
digitalWrite(bip,LOW);
}
delay(500);
}
```

Cole a programação na IDE do Arduino e conecte seu Arduino na porta USB do PC. Em "Tools/Board" selecione a versão do seu Arduino (UNO, Duemilanove, etc) e depois em "Tools/Serial Port" selecione a porta em que seu Arduino está conectado (COMx, ttyUSBx, ttyACMx,etc). E por fim clique em "UPLOAD".

Após fazer o UPLOAD da programação para o Arduino, você pode fazer um teste jogando um pouco de água em cima do sensor de chuva. O Arduino irá apitar e acender o Led. Para uma melhor aplicação, deixe-o com uma inclinação de +-45º.

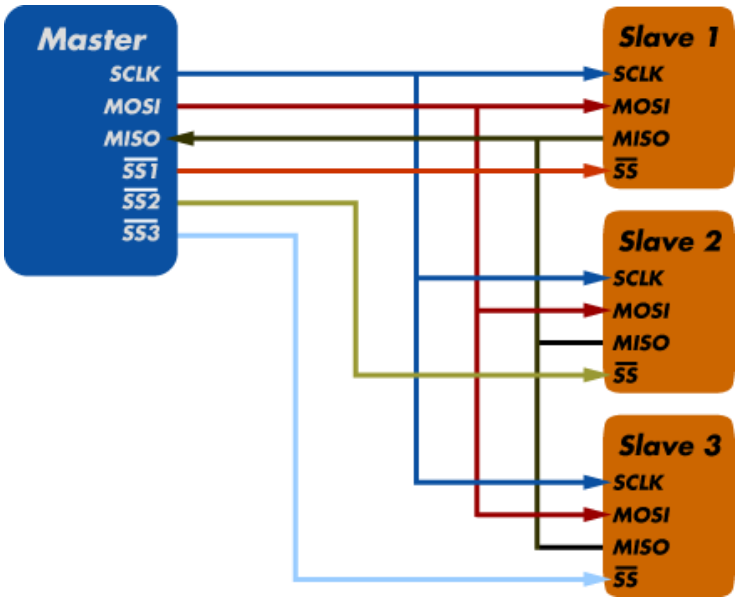
Referências:

http://labdegaragem.com/

http://arduino.cc/en/

http://www.labdegaragem.org/loja/index.php/sensor-de-chuva.html

19. Tutorial: Comunicação SPI com Arduino



A Serial Peripheral Interface é um protocolo de dados seriais síncronos utilizado em microcontroladores para comunicação entre o microcontrolador e um ou mais periféricos. Também pode ser utilizado entre dois microcontroladores.

A comunicação SPI sempre tem um master. Isto é, sempre um será o master e o restante será slave. Por exemplo, o arduino é o master e os outros periféricos são slave. Esta comunicação contém 4 conexões:

- MISO (Master IN Slave OUT) - Dados do Slave para Master;
- MOSI (Master OUT Slave IN) - Dados do Master para Slave;
- SCK (Serial Clock) - Clock de sincronização para transmissão de dados entre o Master e Slave;
- SS (Slave Select) - Seleciona qual Slave receberá os dados.

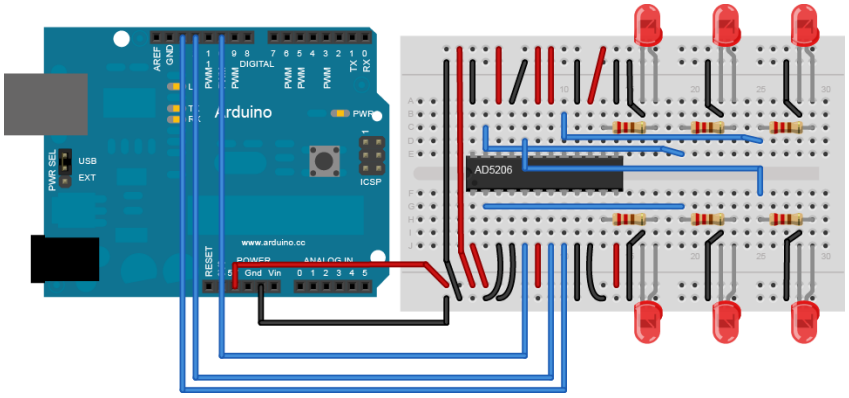
Na figura acima mostra um exemplo de como funciona a comunicação SPI. Alguns periféricos são apenas Slave, por exemplo, cartão SD, memória flash e alguns sensores.

Geralmente estes periféricos contém a mesma pinagem que acima ou a pinagem abaixo:

- SDI - Slave Data IN - Pino de dados de entrada;
- SDO - Slave Data OUT - Pino de dados de saída;
- CS - Seleção de Chip;
- SCK - Clock de sincronização.

O arduino contém uma biblioteca pronta chamada SPI library. O qual você pode mandar dados para periféricos SPI.

Vamos pegar o exemplo "DigitalPotControl" disponível na IDE do Arduino para mostrar seu funcionamento. Para este exemplo vamos utilizar o AD5206. Este Circuito integrado é um potenciômetro digital. Primeiramente faça a seguinte ligação:



Agora abra a IDE do Arduino e abra o exemplo em "File/Examples/SPI/DigitalPotControl". A programação está mostrada abaixo:

```
/*
Digital Pot Control

This example controls an Analog Devices AD5206 digital potentiometer.
The AD5206 has 6 potentiometer channels. Each channel's pins are labeled
A - connect this to voltage
W - this is the pot's wiper, which changes when you set it
B - connect this to ground.

The AD5206 is SPI-compatible,and to command it, you send two bytes,
one with the channel number (0 - 5) and one with the resistance value for the
channel (0 - 255).

The circuit:
* All A pins of AD5206 connected to +5V
* All B pins of AD5206 connected to ground
* An LED and a 220-ohm resisor in series connected from each W pin to ground
* CS - to digital pin 10 (SS pin)
* SDI - to digital pin 11 (MOSI pin)
* CLK - to digital pin 13 (SCK pin)

created 10 Aug 2010
by Tom Igoe

Thanks to Heather Dewey-Hagborg for the original tutorial, 2005

*/

// inslude the SPI library:
#include <SPI.h>

// set pin 10 as the slave select for the digital pot:
const int slaveSelectPin = 10;
```

```
void setup() {
// set the slaveSelectPin as an output:
pinMode (slaveSelectPin, OUTPUT);
// initialize SPI:
SPI.begin();
}
void loop() {
// go through the six channels of the digital pot:
for (int channel = 0; channel < 6; channel++) {
// change the resistance on this channel from min to max:
for (int level = 0; level < 255; level++) {
digitalPotWrite(channel, level);
delay(10);
}
// wait a second at the top:
delay(100);
// change the resistance on this channel from max to min:
for (int level = 0; level < 255; level++) {
digitalPotWrite(channel, 255 - level);
delay(10);
}
}
}
int digitalPotWrite(int address, int value) {
// take the SS pin low to select the chip:
digitalWrite(slaveSelectPin,LOW);
// send in the address and value via SPI:
SPI.transfer(address);
SPI.transfer(value);
// take the SS pin high to de-select the chip:
digitalWrite(slaveSelectPin,HIGH);
}
```

No primeiro "for" é escolhido o Led que acenderá e apagará. No segundo "for" aumenta a intensidade do Led. E no terceiro "for" diminui a intensidade do Led.

Conecte seu Arduino na porta USB do seu PC e selecione a versão do seu Arduino (UNO, Duemilanove, etc) em "Tools/Boards". Selecione a porta (COMx, ttyUSBx, ttyACMx, etc) em que seu Arduino está conectado em "Tools/Serial Port". Por fim, clique em UPLOAD.

Ao terminar de fazer o UPLOAD, cada Led acenderá e apagará devagar.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui neste blog! para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://arduino.cc/en/Tutorial/SPIDigitalPot

http://arduino.cc/en/Reference/SPI

20. Tutorial: Conheça e utilize MOSFETs

Canal P

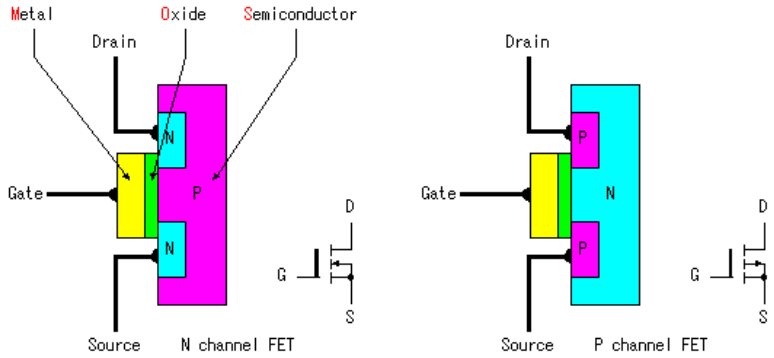
Canal N

Introdução

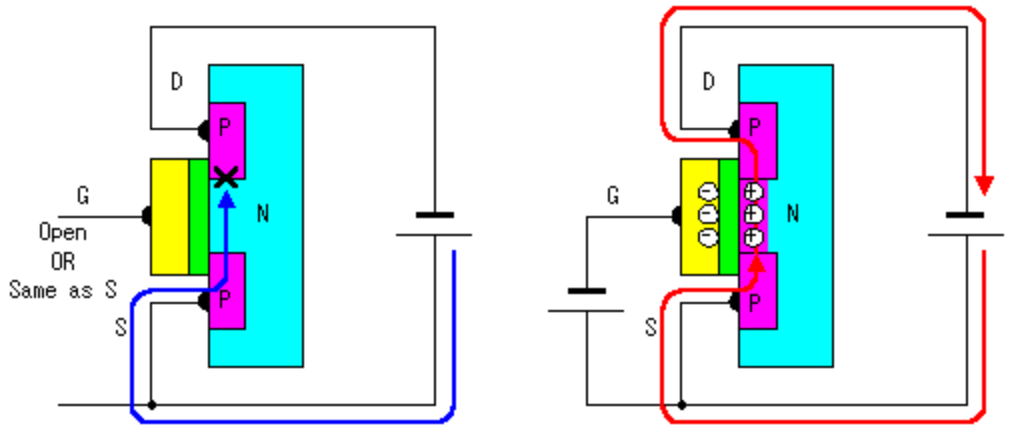
O MOSFET ou simplesmente FET (MOS = *metal-oxide semiconductor* - metal óxido semicondutor e FET = *field effect transistor - transistor de efeito de campo*), é um tipo de transistor, componente usado como chave ou amplificador de sinais elétricos.

TBJs e MOSFETs

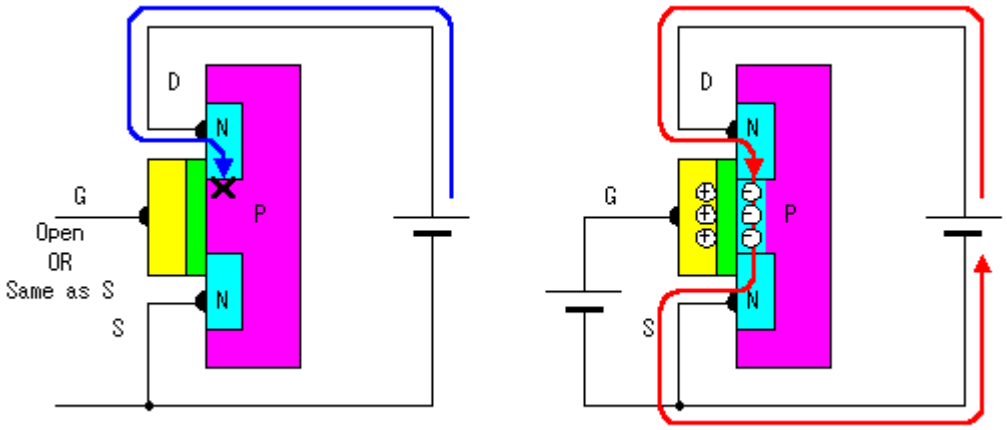
O MOSFET possui normalmente 3 terminais: *Porta, Fonte* e *Dreno* (ou *Gate, Source e Drain* respectivamente). Há dois tipos essenciais: o canal N e o canal P, e se diferenciam basicamente pela polarização. A corrente a ser fornecida para um circuito, que circulará entre o terminal *Fonte* e o *Dreno* do FET, é controlada pela tensão aplicada no terminal *Porta*. Este último possui uma separação dielétrica dos outros dois, gerando portanto uma corrente quase nula no *gate*, e um campo elétrico que influencia no *Dreno* e no *Fonte.* A seguir, está a representação dos dois tipos básicos de FETs e suas usuais simbologias:



Quanto a polarização:



Canal P



Canal N

Um outro tipo de transistor mais conhecido, o TBJ (transistor bipolar de junção), possui também três terminais: *Base*, *Coletore Emissor*. A corrente a ser fornecida para um circuito, que circulará entre o *Coletor* e o *Emissor* do TBJ, é controlada pela*corrente* no terminal *Base* do transistor, e não por uma tensão, como no MOSFET. Essa é uma das principais diferenças entre eles, fazendo com que o TBJ seja aplicado geralmente para circuitos de baixa corrente, e o FET não só para estas, como também para aplicações com maiores valores de potência/corrente.

Exemplos de Aplicações dos MOSFETs

Uma das aplicações mais comuns para os MOSFETs é nos circuitos tipo *CMOS* (ver link de referência no fim do tutorial para mais detalhes). Porém, também há outras, como:

- Resistência controlada por tensão

- Circuitos de comutação de potência

- Misturadores de Frequência

- Etc.

Exemplo: Ligando um motor DC com o MOSFET (aplicação como chave) e o Arduino.

Componentes Necessários:

- 1 placa Arduino

- 1 Transistor MOSFET canal N

- 1 Transistor MOSFET canal P

- 1 motor DC (5V - 1A)

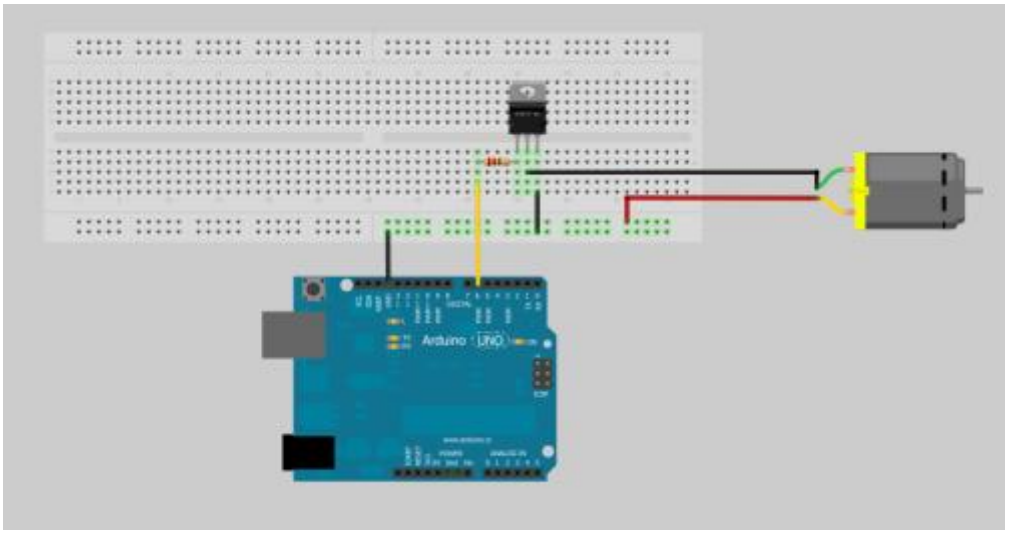
- Cabos para Conexão

- 1 Resistor de 200 Ohms

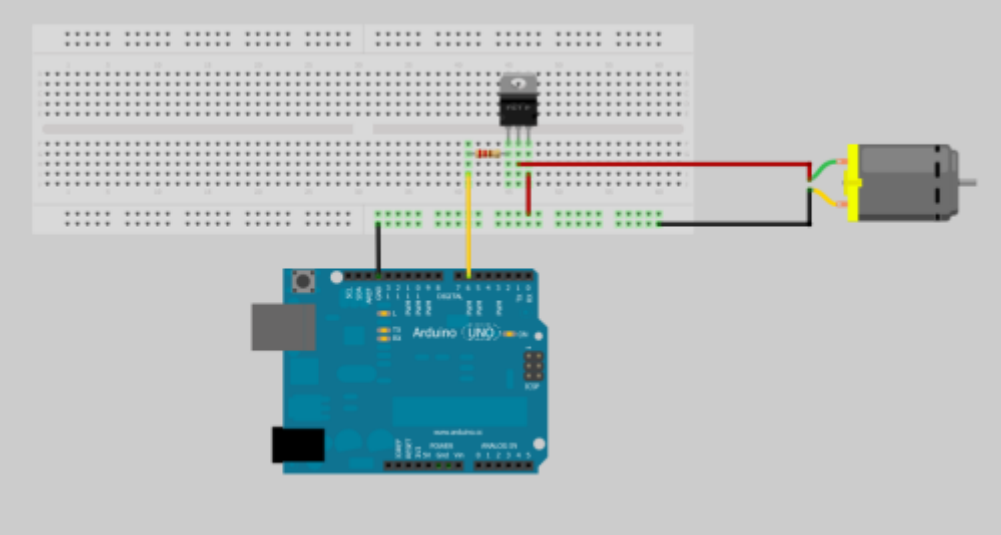
- Fonte de 5V e 1A

Hardware:

Com um MOSFET de Canal N:



Com um MOSFET de Canal P:



Firmware para os dois tipos de MOS (Digitar na IDE do Arduino):

```
// Controle de Motor DC com o transistor MOSFET (aplicação básica) - canais N e P
// O motor utilizado neste Firmware é de 5V - 1A e portanto, 5W.
// O hardware não foi projetado para controlar o sentido de rotação do motor (a não ser que se i
nverta sua polaridade), somente sua velocidade.
// Porém, pode ser controlado por um hardware mais elaborado (uma ponte H de MOSFETs po
r exemplo).
#define GatePin 6 // Define o pino para sinal no Gate do MOSFET.
int sinalGate = 1023; // Valor que será colocado no pino PWM (pode variar de 0 a 1023) para o
controle da velocidade do motor.
void setup () {
pinMode (GatePin, OUTPUT); // Define o pino 5 do Arduino como saída.
}
void loop () {
while ((sinalGate <0) || (sinalGate >1023)) { // Enquanto o valor de "sinalGate" não estiver entre
0 e 1023,
continue; // o Arduino continua sem colocar nada no "GatePin".
}
analogWrite (GatePin, sinalGate); // Se porém estiver nesta faixa, o Arduino envia o sinal para
o "GatePin".
}
```

Observe a parte em azul no código. Na polarização do MOS de canal N, a rotação do motor será máxima quando o valor de "sinalGate" for 1023 (pois no canal N o terminal *Gate* conduz com nível lógico "1"). Já no de canal P, esta rotação será alcançada sendo "sinalGate" igual a 0 (pois no canal P o terminal *Gate* conduz com nível lógico "0"). E o mesmo acontece para a rotação mínima.

E essa é a ideia básica de um MOSFET !!! Esperamos que tenha gostado. Qualquer dúvida, poste no blog, estaremos à disposição.

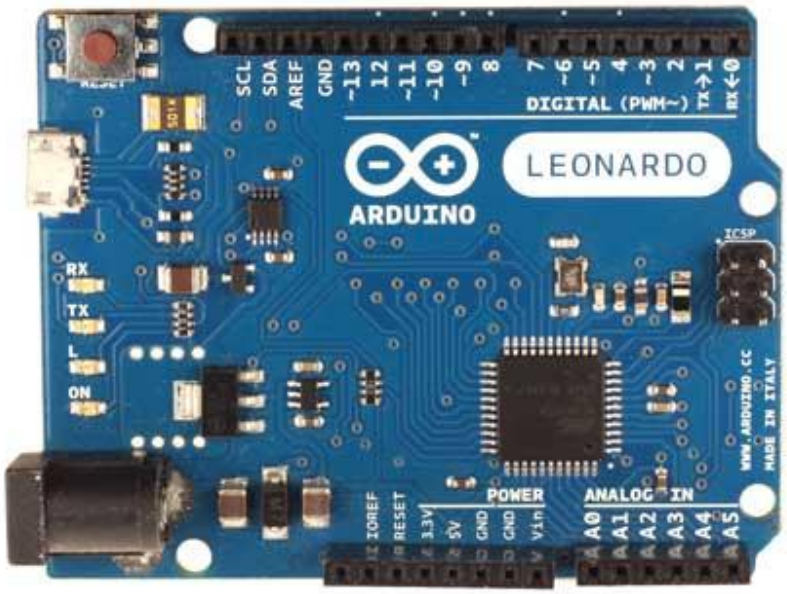
Links de Referência:

- *Datasheet do MOSFET canal P deste tutorial*

- *Datasheet do MOSFET canal N deste tutorial*

- *Tecnologia CMOS*

21. Tutorial: Arduino Leonardo como mouse



Neste tutorial vamos mostrar o novo Arduino Leonardo.

O Arduino Leonardo vem com 14 I/O's digitais e 6 I/O's analógicas. Ele vem com o Atmega32u4, o qual já vem com entrada USB e pode ser utilizado como USB Device. Uma das aplicações mais simples é utilizá-lo como mouse ou teclado. O Atmega32u4 tem 32Kbytes de memória flash, sendo que 4Kbytes são de bootloader. Tem também 2.5Kbytes de memória RAM e 1Kbytes de memória EPPROM.

Uma diferença entre o novo Arduino Leonardo e o Arduino Uno é que o Arduino Leonardo pode ser utilizado como USB Device como dito anteriormente.

Então vamos mostrar como utilizar o novo Arduino Leonardo como mouse. Vamos utilizar o Joystick Shield Kit. Para maior estabilidade nos botões, foi colocado resistores de Pull-UP de 10Kohm nas entradas digitais (2 ao 6) do Arduino Leonardo. Para saber mais sobre resistores de Pull-Up, clique aqui!

Conecte o Joystick Shield Kit no Arduino Leonardo, abra a IDE 1.0.1 do Arduino e conecte o Arduino Leonardo na porta USB do PC. Caso estiver com Windows, instale o Driver do Arduino Leonardo localizado na pasta "drivers" dentro da pasta IDE do Arduino.

Na IDE do Arduino, abra o exemplo JoystickMouseControl. Caso esteja utilizando os resistores de PULL-UP faça as modificações em vermelho abaixo:

```
// set pin numbers for switch, joystick axes, and LED:
const int switchPin = 2; // switch to turn on and off mouse control
const int mouseButton = 3; // input pin for the mouse pushButton
const int xAxis = A0; // joystick X axis
const int yAxis = A1; // joystick Y axis
```

```
const int ledPin = 5; // Mouse control LED
// parameters for reading the joystick:
int range = 12; // output range of X or Y movement
int responseDelay = 5; // response delay of the mouse, in ms
int threshold = range/4; // resting threshold
int center = range/2; // resting position value
boolean mouseIsActive = false; // whether or not to control the mouse
int lastSwitchState = HIGH; // previous switch state
void setup() {
pinMode(switchPin, INPUT); // the switch pin
pinMode(ledPin, OUTPUT); // the LED pin
// take control of the mouse:
Mouse.begin();
}
void loop() {
// read the switch:
int switchState = digitalRead(switchPin);
// if it's changed and it's high, toggle the mouse state:
if (switchState != lastSwitchState) {
if (switchState == LOW) { //Change to HIGH if the board doesn't have Pull Up Resistors
mouseIsActive = !mouseIsActive;
// turn on LED to indicate mouse state:
digitalWrite(ledPin, mouseIsActive);
}
}
// save switch state for next comparison:
lastSwitchState = switchState;
// read and scale the two axes:
int xReading = readAxis(A0);
int yReading = readAxis(A1);
// if the mouse control state is active, move the mouse:
if (mouseIsActive) {
Mouse.move(xReading, yReading, 0);
}
// read the mouse button and click or not click:
// if the mouse button is pressed:
if (digitalRead(mouseButton) == LOW) { //Change to HIGH if the board doesn't have Pull Up
Resistors
// if the mouse is not pressed, press it:
if (!Mouse.isPressed(MOUSE_LEFT)) {
Mouse.press(MOUSE_LEFT);
}
}
// else the mouse button is not pressed:
else {
// if the mouse is pressed, release it:
if (Mouse.isPressed(MOUSE_LEFT)) {
Mouse.release(MOUSE_LEFT);
}
}
delay(responseDelay);
}
/*
reads an axis (0 or 1 for x or y) and scales the
analog input range to a range from 0 to <range>
*/
int readAxis(int thisAxis) {
// read the analog input:
int reading = analogRead(thisAxis);
if(thisAxis == A1)  //invert the Y Axis
```

```
{
reading = map(reading, 1023, 0, 0, range);
}
// map the reading from the analog input range to the output range:
if(thisAxis == A0)
{
reading = map(reading, 0, 1023, 0, range);
}
// if the output reading is outside from the
// rest position threshold, use it:
int distance = reading - center;
if (abs(distance) < threshold) {
distance = 0;
}
// return the distance for this axis:
return distance;
}
```

Na programação foi feito algumas modificações para estabilidade (resistores de Pull-Up) e duas condições para inversão do cursor do mouse.

Depois de instalado o Driver do Arduino Leonardo, vá em Tools/Board e selecione o Arduino Leonardo. Depois em Tools/Serial Port e veja em qual porta o Arduino Leonardo está conectado (COMx, ttyUSBx, ttyACMx, etc). Clique em UPLOAD. Assim que terminar, aperte o botão do analógico do Joystick Shield. Este botão é o botão da porta digital 2.

Agora mova o analógico e você verá que o cursor do mouse se movimentará! O botão da porta digital 3 é o clique do botão esquerdo do mouse.

E é isso! Esperamos que tenha gostado! Caso tenha dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

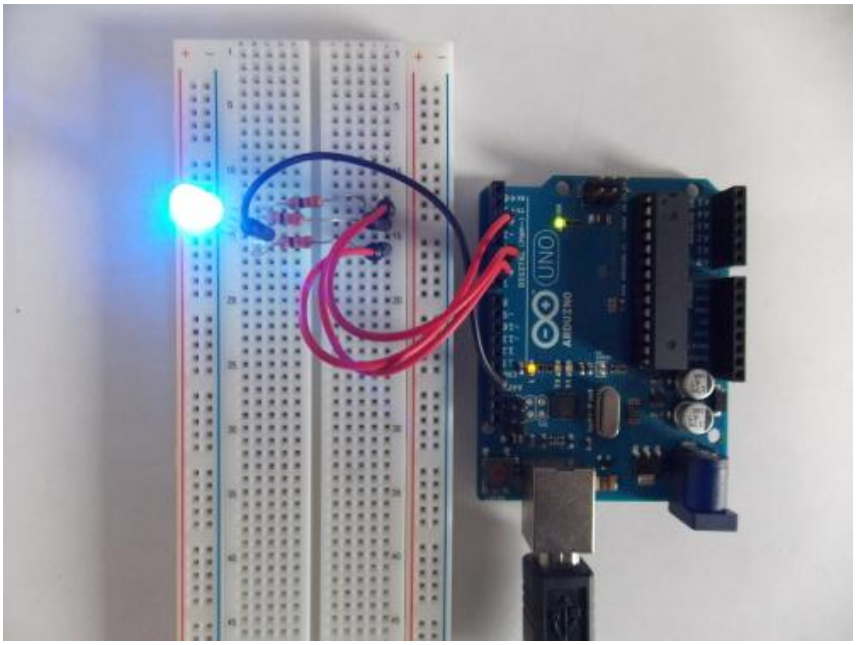
http://www.labdegaragem.org/loja/index.php/31-shields/joystick-shie...

http://arduino.cc/en/Tutorial/JoystickMouseControl

http://arduino.cc/en/Tutorial/HomePage

22. Tutorial: Reprodução de Cores com LED RGB e Arduino

Neste tutorial será mostrado uma maneira simples de controlar um LED RGB com o Arduino. O RGB é um LED que reproduz três cores: vermelho, verde e azul. Ele possui 4 terminais, um para cada cor e o GND. É possível acender cada uma das cores com diferentes intensidades, resultando em cores diferentes das primárias. Esse controle será gerenciado por três portas com saída PWM do Arduino, reproduzindo-as.



Para isto, serão necessários os seguintes componentes:

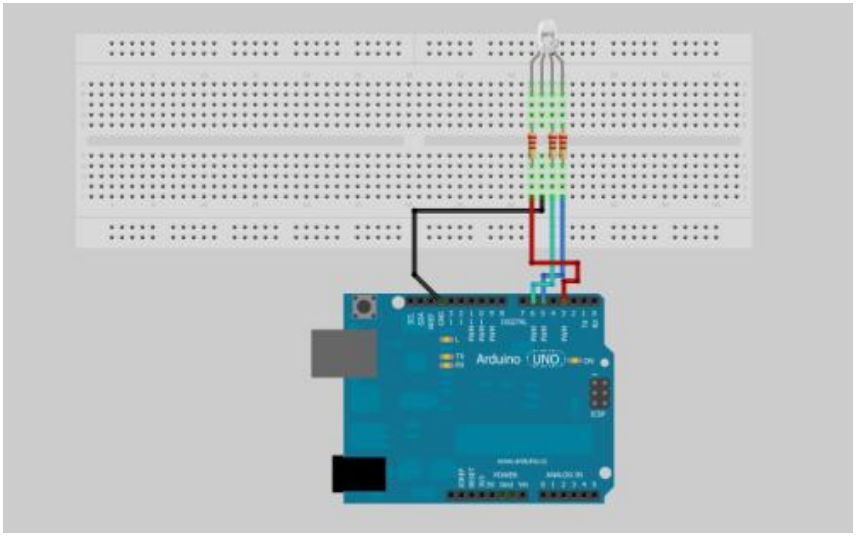
- 1 placa Arduino Uno

- Jumpers para Conexão

- 3 Resistores de 330 Ohms - 0,25W

- 1 LED RGB

A seguir, monte o circuito como mostrado abaixo:



A seguir, abra a IDE do Arduino e digite o seguinte código:

```
// Firmware para Controle de Cores do LED RGB
#define RED 3 // Define o pino 3 com PWM como RED
#define BLUE 5 // Define o pino 5 com PWM como BLUE
#define GREEN 6 // Define o pino 6 com PWM como GREEN
int red = 255, green = 255, blue = 255; // valores de 0 a 255 (variá-los para obter cores diferentes).
void setup() {
pinMode (RED, OUTPUT); // Pino 3 declarado como saída
pinMode (BLUE, OUTPUT); // Pino 5 declarado como saída
pinMode (GREEN, OUTPUT); // Pino 6 declarado como saída
}
void loop() {
analogWrite (GREEN, green); // Envia o sinal de "green" para o terminal de cor verde do LED
analogWrite (BLUE, blue); // Envia o sinal de "blue" para o terminal de cor azul do LED
analogWrite (RED, red); // Envia o sinal de "red" para o terminal de cor vvermelha do LED
}
```

Observe a linha marcada *em azul* no código. Há três variáveis. São elas que controlarão a intensidade da luminosidade de cada LED. Esses valores podem variar de 0 a 255 (devido à faixa do PWM do Arduino) para cada LED. Se você os variar, perceberá a mudança de cores. E o seu reprodutor de cores com o LED RGB está pronto!!! Esperamos que tenha gostado! Qualquer dúvida, poste no blog!

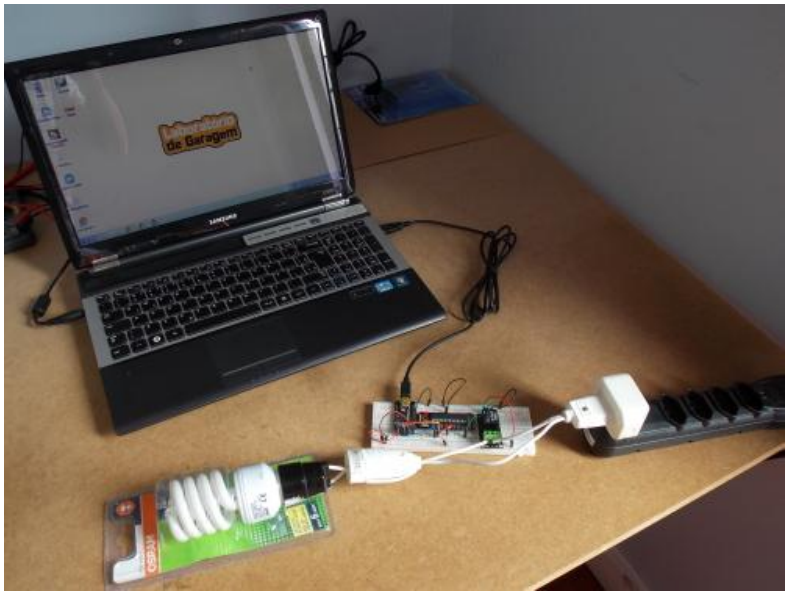
Links de Referência:

- DataSheet do LED RGB usado neste tutorial

- PWM (Arduino)

23. Tutorial: Controlando Módulo Rele com Garagino, enviando comandos via Serial

Olá pessoal! Vamos mostrar um tutorial utilizando o Garagino + Módulo Rele + Conversor USB/Serial para controlar uma lâmpada ligada na rede elétrica.



ATENÇÃO CUIDADO: Evite manusear o Módulo Rele enquanto o mesmo estiver conectado na rede elétrica e remova qualquer tipo de objeto condutivo ou inflamável de perto. Você pode causar um acidente.

Lista de materiais:

- 1 x Garagino
- 1 x Conversor USB/Serial
- 1 x Módulo Rele
- 1 x Lâmpada 127V com bocal e tomada.
- 1 x Protoboard
- Jumpers

Montagem:

1º Corte um dos fios da tomada (figura 1) e uma das pontas conecte aos pinos C (comum) e a outra no pino NA(Normal Aberto) do módulo Rele. Conecte todos os módulos na protoboard.

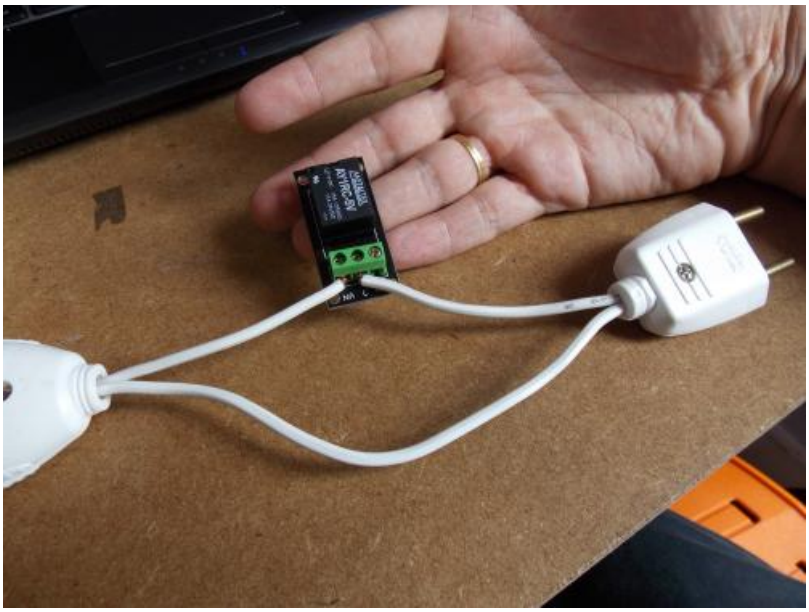


Figura 1

A partir deste passo use como referência as figuras 2 e 3 para fazer as conexões:

2º Conecte o Pino 5V do módulo USB /Serial com o barramento Vcc "+" do protobord (Dê preferência por usar fio vermelho).

3º Conecte o Pino GND do módulo USB /Serial com o barramento GND "-"do protobord (Dê preferência por usar fio preto).

4º Conecte um capacitor de acoplamento (incluso no módulo USB/Serial) entre o pino DTR do conversor USB/Serial e o pino RST do Garagino .

5º Em seguida conecte os Pinos: Vcc do Garagino, CTS do módulo USB /Serial e o Vcc do Módulo Rele ao barramento Vcc "+" do protobord (Dê preferência por usar fios vermelhos).

6º Agora conecte os Pinos: GND do Garagino e GND do Módulo Rele ao barramento GND "–" do protobord (Dê preferência por usar fios pretos).

7º Conecte o Pino TX do USB/Serial ao pino RX do Garagino (fio azul da Figura 2).

8º Conecte o Pino RX do USB/Serial ao pino TX do Garagino (fio laranja da Figura 2).

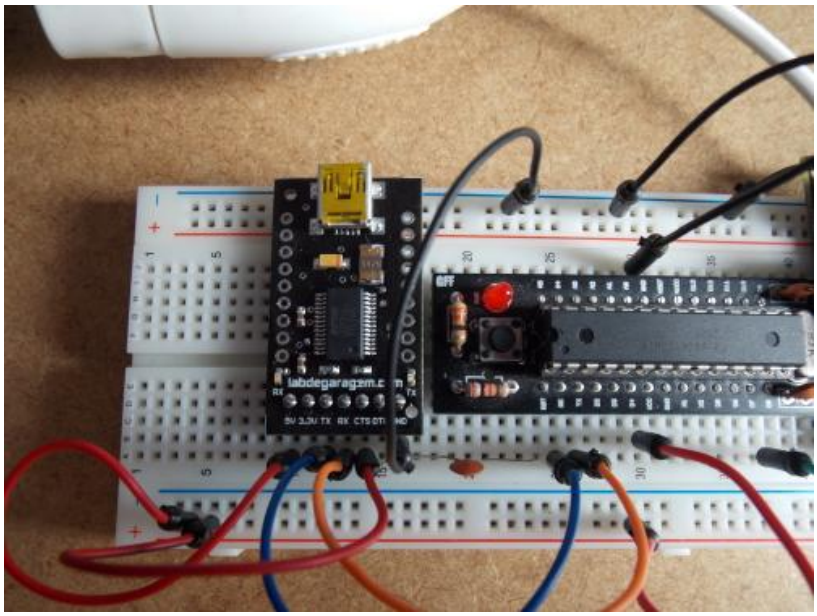


Figura 2

9º Conecte a entrada **IN** do Módulo Rele ao pino D8 do Garagino (fio verde da Figura 3).

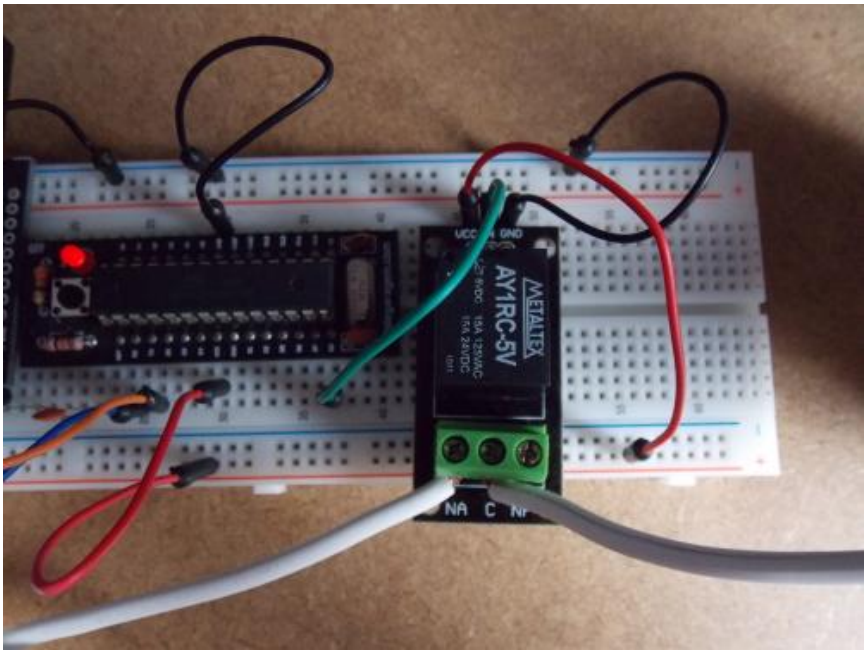


Figura 3

Agora com o protoboard montado, verifique se todas as ligações estão corretas.

Ligue o cabo USB entre o PC e o Conversosr USB/Serial e por último conecte a tomada na rede elétrica.

Copie e cole o Sketch Abaixo e grave no Arduino:

```
char leitura;

#define rele 8

void setup() {
  //Inicializa comunicação Serial
  Serial.begin(9600);
  //Seta o pino indicado por rele como saída
  pinMode(rele, OUTPUT);
  //Mantem rele desligado assim que iniciar o programa
  digitalWrite(rele,LOW);
}
void loop() {
  //Verifica se há conexão com a serial
  while (Serial.available() > 0) {
    //Lê o dado vindo da Serial e armazena na variável leitura
    leitura = Serial.read();
    //Se a variável leitura for igual a 'd' ou 'D' ela Desliga rele
    if (leitura == 'd' || leitura =='D'){// As duas || é a operação booleana OU
      digitalWrite(rele,LOW);
    }
    /*Senão verifica se a variável leitura é
     igual a 'l' ou 'L' ela Liga rele */
    else if (leitura == 'l' || leitura =='L'){
      digitalWrite(rele,HIGH);
    }
    Serial.println(leitura);
  }
}
```

Para enviar os comandos de Liga e Desliga, abra o Serial Monitor ou qualquer programa Monitor, configure para 9600baud. Agora envie a letra “L” ou "l"e a lâmpada ligará e “D” ou "d" ela desligará.

E é isso! Esperamos que tenha gostado! Caso tenha dúvidas, poste aqui neste blog! Se tiver sugestões para tutoriais, poste aqui. Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente

Referências:

http://www.labdegaragem.org/loja/index.php/modulo-rele-10a.html

http://www.labdegaragem.org/loja/index.php/protoboard-840.html

http://www.labdegaragem.org/loja/index.php/jumper-wire-kit.html

http://www.labdegaragem.org/loja/index.php/conversor-usb-serial-ft2...

http://www.labdegaragem.org/loja/index.php/garagino-proto.html

http://www.arduino.cc/

24. Tutorial: Novo firmware com mais recursos para o Android Mini PC

O Mini PC ANDROID 4.0 é um computador portátil com o sistema operacional Google Android 4.0, possui um processador de 1.5GHz (ARM Cortex-A8), com memória RAM de 1GB, e 4GB para armazenamento de dados. Também inclui um cartão Micro SD, uma saída Mini HDMI (vídeo e áudio), portas USB, Mini USB e conexão Wi-Fi.

Neste tutorial mostraremos como atualizar o firmware de seu Mini PC ANDROID 4.0 utilizando o LiveSuit.

Mas porque atualizar o firmware?

O LdG, em parceria com o fabricante do equipamento, obteve com exclusividade um firmware ainda mais completo, permitindo uma gama muito maior de aplicativos disponíveis no Google Play. Assim, recomendamos a todos que atualizem o firmware de seu Mini PC, obtendo assim, ainda mais recursos. Então, vamos lá!

1) Faça o download do Driver USB.

2) Faça o download do FirmwareUpgrade.

- Segue MD5sum do arquivo: **694e4fa53bf69a29ed49c9a73d4ac93c**

3) Faça o download do LiveSuit.

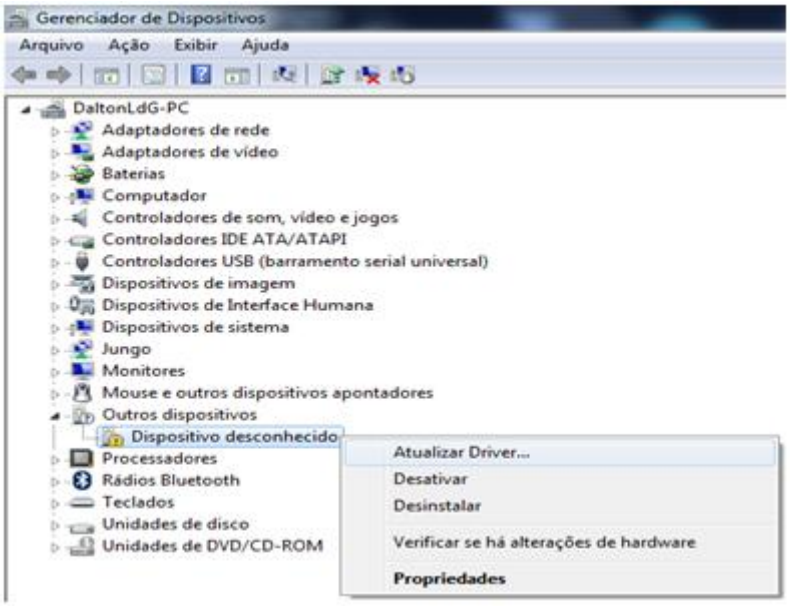
4)Encontre o botão de Reset do seu Mini PC, conforme a figura abaixo.

5) Conecte o cabo USB na porta mini USB e pressione o botão de Reset do seu Mini PC com um palito de dente, sem conectar a outra ponta do cabo no computador.

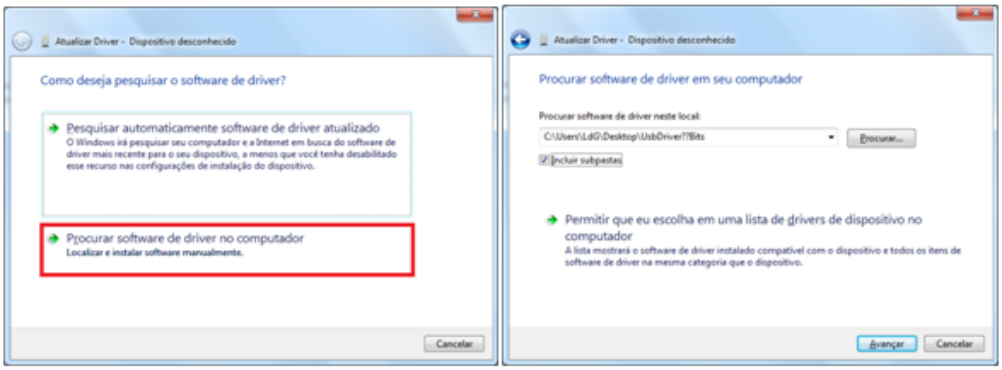
6) Enquanto pressiona o botão de Reset conecte a outra ponta do cabo USB na porta de seu computador.

7) Após conectado, solte o botão de Reset, o computador irá tentar reconhecer e instalar o dispositivo, mas não conseguirá.

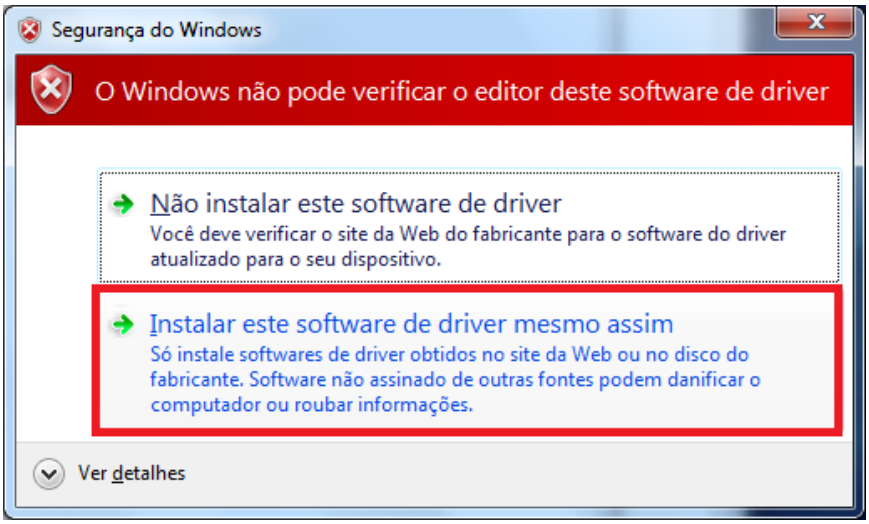
8) Abra o gerenciador de dispositivo pressionando “ Windows + R” e digitando na janela “devmgmt.msc” e clique em Ok, após abrir a janela clique com o botão direito do mouse e em seguida clique em atualizar driver, conforme a figura abaixo.



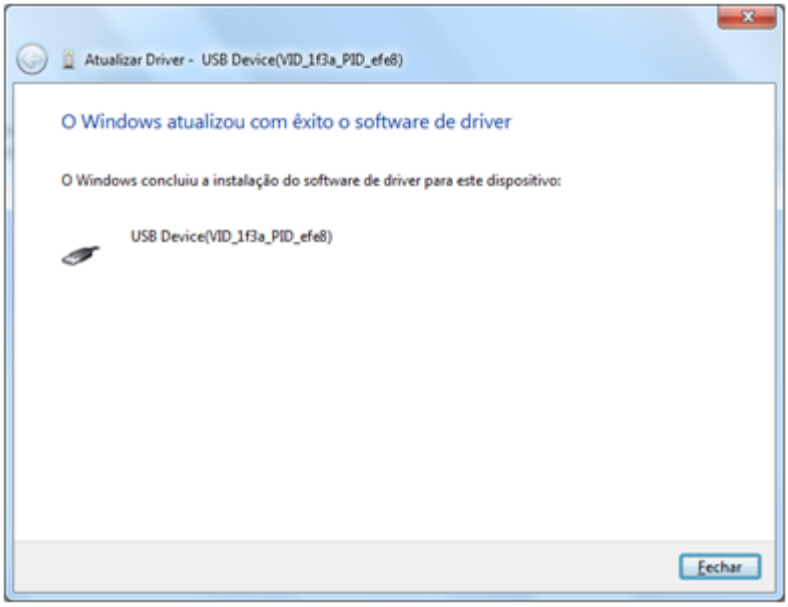
9) Clique na segunda opção, “Procurar software de driver no computador” e selecione a pasta onde você descompactou seu Driver USB, depois clique em Avançar.



10) Após clicar em Avançar, abrirá uma janela dizendo que o windows não pode verificar o editor do software, em seguida clique na segunda opção "Instalar este software de driver mesmo assim", conforme em destaque na figura abaixo.



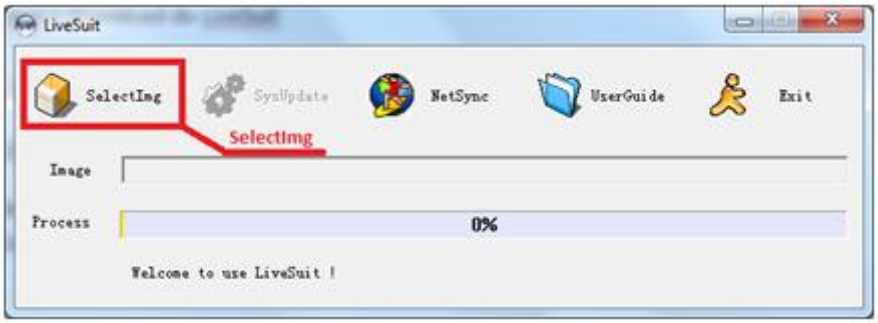
11) Ele irá instalar o dispostivo USB, clique em Fechar.



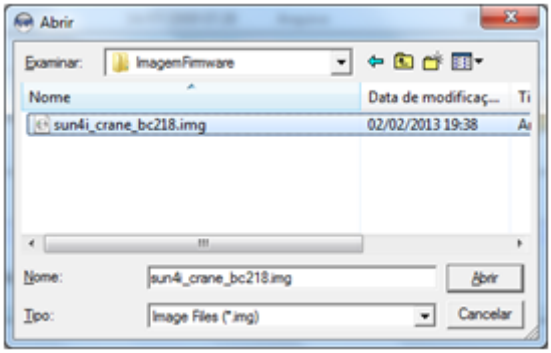
OBS: Seu ícone USB poderá aparecer da seguinte maneira abaixo, porem não interfere em nada.

12) Desplugue o cabo USB de seu Computador.

13) Agora execute o LiveSuite.exe que está dentro da pasta LiveSuit.



14) Clique em SelecImg (Retângulo em Destaque na Imagem Acima) e selecione a Imagem do firmware, conforme a figura abaixo.

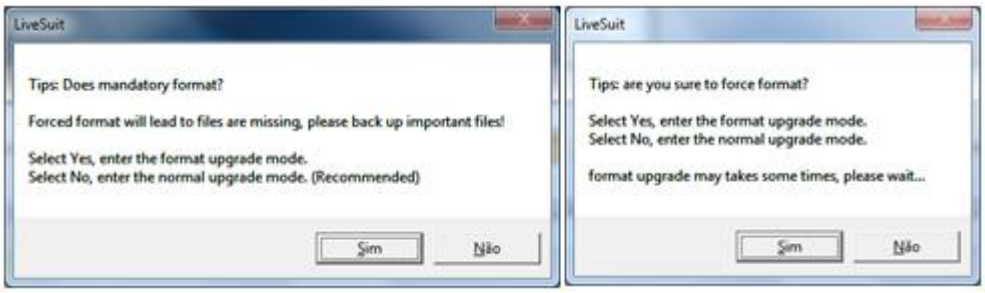


15) Repita o procedimento, pressione o botão de Reset do seu Mini PC com um palito de dente, sem conectar a outra ponta do cabo USB no computador.

16) Enquanto pressiona o botão de Reset conecte a outra ponta do cabo USB na porta de seu computador.

17) Após conectado, retire o palito de dente para despressionar o Reset.

18) Dentro de alguns segundos o programa LiveSuit irá reconher que o dispositivo foi conectado, clique SIM nas duas janelas que aparecerão.



19) Após o processo de atualização irá abrir uma janela com a mensagem “Update sucess”, conforme a imagem abaixo.

Esperamos que tenham gostado do tutorial, futuramente traremos até vocês novas aplicações utilizando o Android Mini PC.

Referências

- http://www.envythisstuff.com/techtalk/mk802faq.html#howtoupdatefirm...

- http://liliputing.com/2012/06/how-to-re-install-mk802-firmware.html

25. Tutorial: Como utilizar o teclado de toque capacitivo

No tutorial de hoje iremos mostrar como utilizar o Teclado de Toque Capacitivo, para pegar uma senha digitada por você para validar.

Materiais Utilizados:

1x Arduino

1x Teclado de Toque Capacitivo - MPR121

1x Barra de Pinos Reto

Alguns Jumpers

1. O Funcionamento

Esse teclado de toque capacitivo utiliza o circuito integrado MPR121, 12 botões sensíveis ao toque e comunicação I²C para se comunicar com o Arduino. A placa também possui 4 furos de montagem que permitem que ele seja acoplado a algum suporte.

Nesse exemplo baseado no código exemplo do fabricante, utilizaremos o Serial Monitor da Arduino IDE para lermos os botões pressionados e comparar com uma senha definida no programa. Se a senha digitada estiver correta, irá aparecer a mensagem "Senha Correta", e se a senha digitar estiver errada, irá aparecer a mensagem "Senha Incorreta". Foi tratada também a situação em que se dois teclados forem pressionados, nenhum dos botões é lido.

O Teclado possui 5 terminais:

GND - (0V): Alimentação

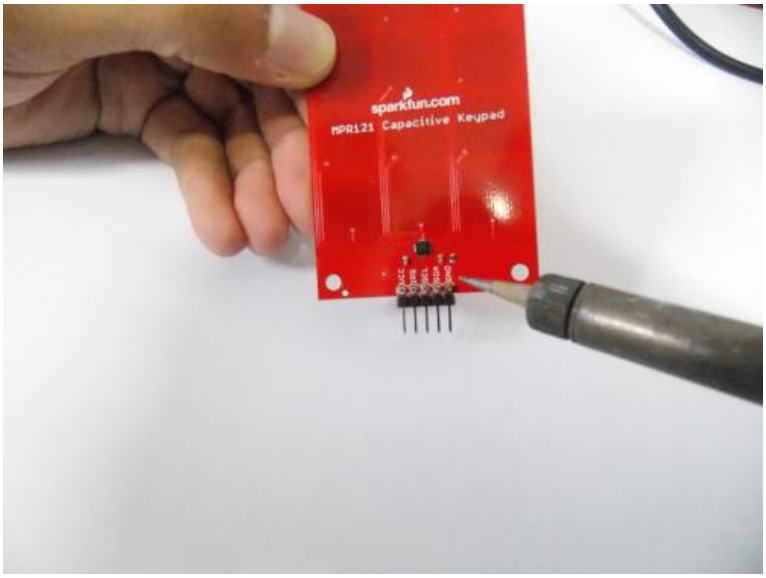
VCC - (3.3V): Alimentação

SDA e SCL: Esses terminais são utilizados para a comunicação I²C, sendo eles Dados Seriais (Serial Data - SDA) e Clock Serial (Serial Clock - SCL)

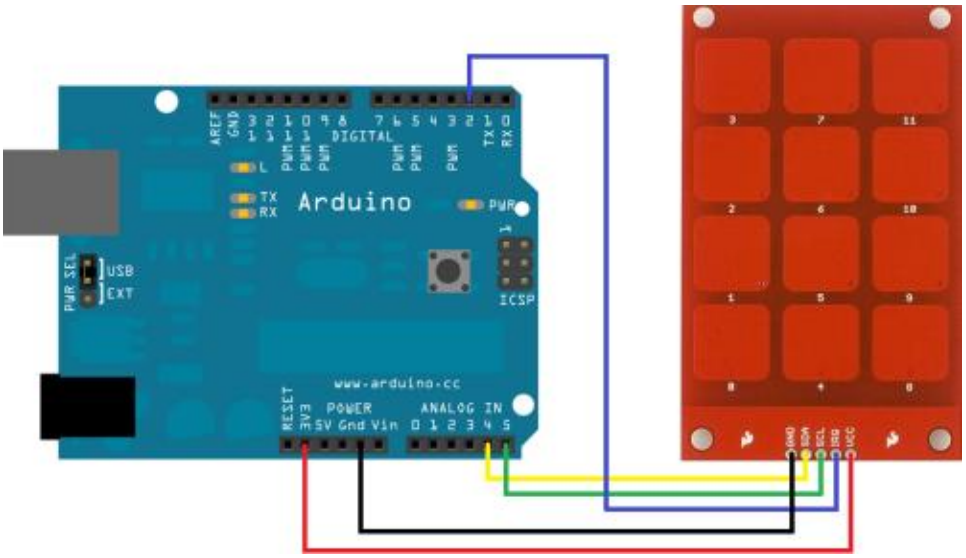
IRQ - (Interrupção de Hardware): Abreviação de "Interrupt Request Line", esses endereços de IRQ são interrupções de hardware, canais que os dispositivos podem utilizar para chamar a atenção do processador, no nosso caso Arduino.

2. A Montagem

2.1) Pegue 5 pinos da sua barra de pinos e solte nos 5 terminais do teclado:



2.2) Conecte os terminais GND, SDA, SCL, IRQ, VCC ao arduino, conforme a imagem abaixo:



3. O Sketch

Você pode baixar o programa completo que utilizamos neste tutorial clicando neste link. No vídeo você pode ver a explicação do código.

Referências:

http://www.labdegaragem.org/loja/index.php/keypad-de-toque-capacitivo-mpr121.html

https://www.sparkfun.com/datasheets/Components/MPR121.pdf

Tutorial: Como utilizar Termistor NTC com Arduino

Olá Garagistas!!!

Neste tutorial vamos utilizar o Termistor NTC de 10K junto com o Arduino e a biblioteca *Thermistor* para obtermos os valores em graus Celsius. Esta biblioteca utiliza a equação de Steinhart-Hart para converter o valor da resistência em valores de temperatura e em uma segunda monstagem, mostramos como fazer o mesmo só que agora utilizando o método do Fator Beta (exemplo baseado em um exemplo da Adafruit).

Lista de Materiais

*1 x Arduino Uno Rev 3 ou Garagino Rev 1**

1 x Termistor NTC de 10KΩ

1 x Resistor de 10KΩ

1 x Protoboard Mini

Alguns jumpers

** Se for utilizar o Garagino você necessitará do conversor USB-Serial, pois vamos imprimir o valor convertido do sensor em ºCelsius no Serial Monitor.*

O Termistor

Para fazer a leitura da temperatura, vamos utilizar um Termistor. Este é um tipo de resistência especial que altera seu valor razão da temperatura onde o componente é colocado.

Existem dois tipos de Termistores os NTC e os PTC.

Termistor PTC (Positive Temperature Coefficient): Este tipo de Termistor tem o coeficiente de temperatura positivo, ou seja, a resistência aumenta com o aumento da temperatura.

Termistor NTC (Negative Temperature Coefficient): Já este é o inverso do anterior e seu coeficiente de temperatura é negativo. Com isto sua resistência diminui com o aumento da temperatura.

O valor nominal do termistor é dado normalmente a 25 ºCelsius e neste caso utilizaremos um termisitor de 10K. Ele atua na faixa de -40 a +125.

Determinação da temperatura

Para determinar a temperatura do existe dois métodos um utilizando a interpolação pela fórmula de Steinhart-Hart ou ainda podemos utilizar a equação do fator Beta.

Abaixo você vê dois métodos para aquisição da temperatura:

Método Steinhart-Hart

$$\frac{1}{T} = A + B\ln(R) + C(\ln(R))^3$$

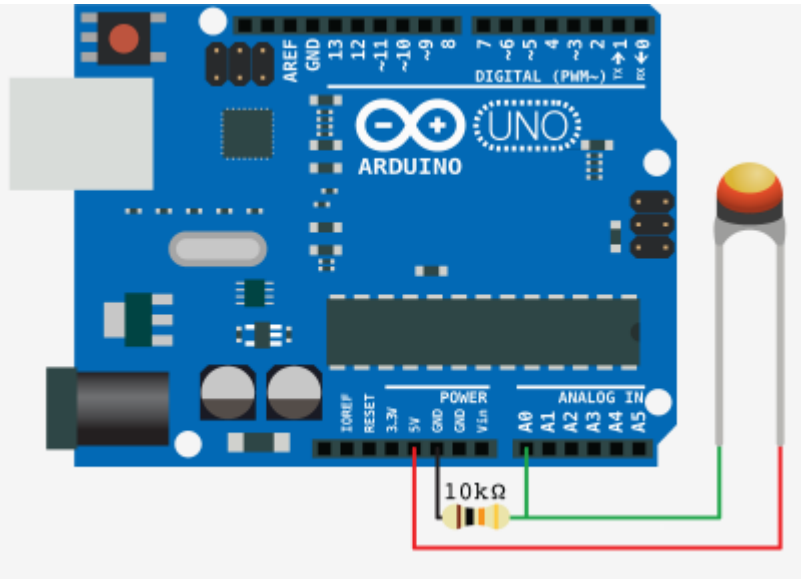
Equação Steinhart-Hart

O método de Steinnhart- Hart é implementado por uma biblioteca que fizemos algumas modificações para mantermos a compatibilidade da mesma com a IDE do Arduino.

A biblioteca pode ser baixada neste link: Thermistor.zip

Realizamos alguns testes aqui com este sensor junto aos valores desta biblioteca e o termistor respondeu a valores muito próximos desejados(mesmo sem ter feito alterações nos valores dos coeficientes de temperatura).

Para este Circuito podemos utilizar e o Sketch exemplo com este circuito:



Circuito para Método Steinhart-Hart

Sketch

```
#include <Thermistor.h>

Thermistor temp(0);

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

void loop() {
  int temperature = temp.getTemp();
  Serial.print("Temperatura no Sensor eh: ");
  Serial.print(temperature);
  Serial.println("*C");
```

```
  delay(1000);
}
```

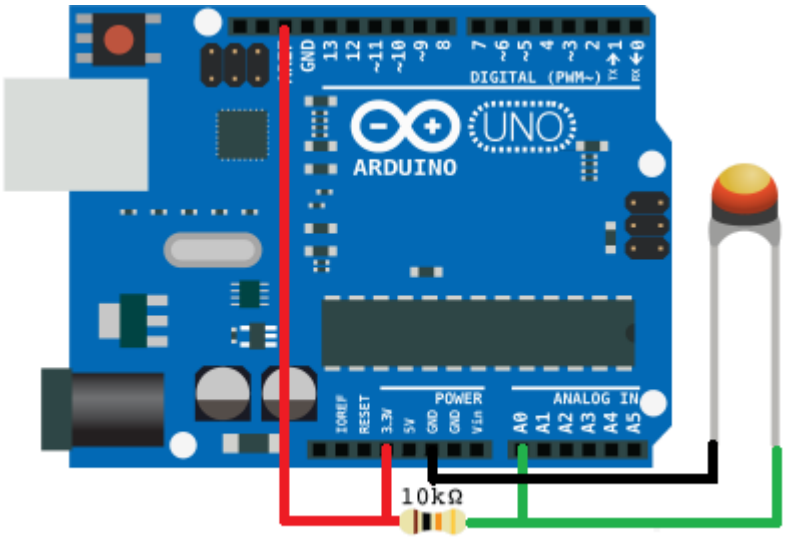
Método do Fator Beta

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{B}\ln\left(\frac{R}{R_0}\right)$$

Equação do fator Beta

O método do fator beta pode ser implementado utilizando um código de exemplo usado pela adafruit (fizemos uma pequena modificação pois o Termistor que utilizamos é um NTC).

O circuito deve ser montado segundo a função abaixo:



Circuito para Método do Fator Beta

Segundo o manual do nosso termistor, o fator beta do mesmo é 3977 e utilizamos ele no Sketch abaixo:

```
// Pino onde o Termistor esta conectado
#define PINOTERMISTOR A0
// Valor do termistor na temperatura nominal
#define TERMISTORNOMINAL 10000
// Temp. nominal descrita no Manual
```

```
#define TEMPERATURENOMINAL 25
// Número de amostragens para
#define NUMAMOSTRAS 5
// Beta do nosso Termistor
#define BCOEFFICIENT 3977
// valor do resistor em série
#define SERIESRESISTOR 10000

int amostra[NUMAMOSTRAS];
int i;
void setup(void) {
  Serial.begin(9600);
  analogReference(EXTERNAL);
}

void loop(void) {
  float media;

  for (i=0; i< NUMAMOSTRAS; i++) {
   amostra[i] = analogRead(PINOTERMISTOR);
   delay(10);
  }

  media = 0;
  for (i=0; i< NUMAMOSTRAS; i++) {
     media += amostra[i];
  }
  media /= NUMAMOSTRAS;
  // Converte o valor da tensão em resistência
  media = 1023 / media - 1;
  media = SERIESRESISTOR / media;

  //Faz o cálculo pela fórmula do Fator Beta
  float temperatura;
  temperatura = media / TERMISTORNOMINAL;     // (R/Ro)
  temperatura = log(temperatura); // ln(R/Ro)
  temperatura /= BCOEFFICIENT;                  // 1/B * ln(R/Ro)
  temperatura += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
```

```
  temperatura = 1.0 / temperatura;                // Inverte o valor
  temperatura -= 273.15;                          // Converte para Celsius

  Serial.print("Temperatura no Sensor eh: ");
  Serial.print(temperatura);
  Serial.println(" *C");

  delay(1000);
}
```

É isto pessoal espero que vocês gostem do Tutorial e que usem estes sensores, eles são uma alternativa barata e simples de conseguir uma amostra da temperatura ambiente.

26. Tutorial: Como utilizar a Chave Óptica

Olá Garagistas! No tutorial de hoje, mostraremos como utilizar a chave óptica, e como demonstração utilizaremos ela para identificar cartões e liberar acesso, utilizando o serial monitor da Arduino IDE.

Materiais Utilizados:

1x Arduino

1x Chave Óptica

1x Resistor 330Ω

1x Protoboard Mini - 170 Pinos

Alguns Jumpers

1. O Funcionamento

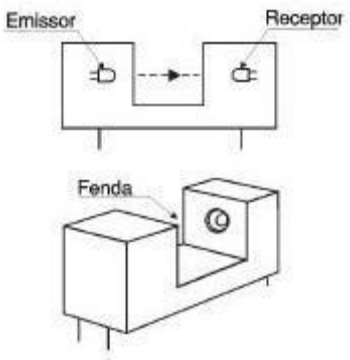


Figura 1 - Chave Óptica

Uma chave óptica geralmente é constituída de um emissor (LED Infravermelho) e um receptor (Fototransistor). O dispositivo é montado de forma que entre o Emissor e o Receptor exista uma fenda onde possa ser introduzido algum objeto. A introdução desse objeto nessa fenda, interrompe o feixe de luz, provocando uma mudança do estado lógico da chave óptica de 1 para 0.

As chaves ópticas são rápidas e têm a vantagem de não utilizarem contatos mecânicos, que com o tempo acabam se desgastando, são muito utilizadas na automação como fim de curso, em robótica, e em muitas outras aplicações importantes relacionadas a mecatrônica e opto-eletrônica.

Na imagem abaixo, você pode ver um tipo de aplicação utilizando a chave óptica, como mecanismo de segurança. Nela é utilizado um cartão com uma certa combinação de recortes, para que quando o cartão for introduzido na “fenda” do mecanismo, ele permita fazer algum

acionamento como, abrir um portão, um armário, ou simplesmente liberar algum tipo de acesso.

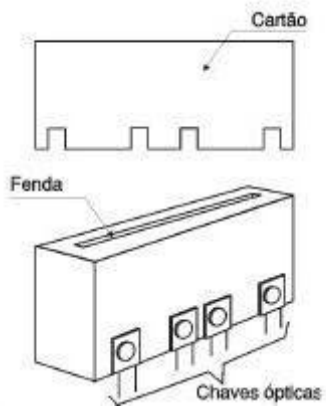
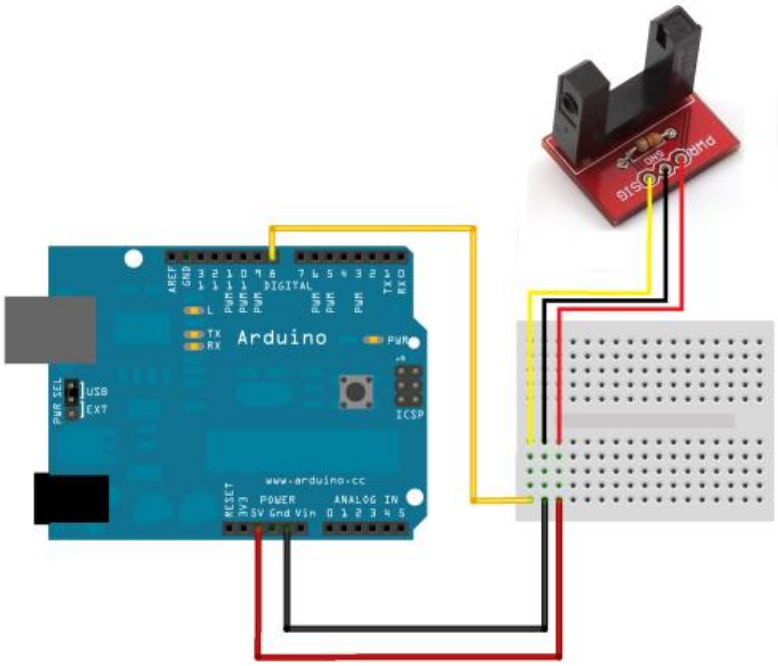


Figura 2 - Exemplo de Aplicação Utilizando-se a Chave Óptica

Em nossa aplicação utilizaremos cartões com recortes retangulares, para que ao passar o cartão pela fenda da chave óptica, a mesma detecte os recortes, deixando a luz infravermelha passar e conte para cada recorte um pulso, e para cada cartão, dependendo da quantidade de recortes atribui-se uma ID.

Imagem 1 - Cartão com 1 recorte (ID 1)

2. A Montagem



3. O Sketch

```
#define sensor 8 //Define "sensor" com valor 8

int estado=0; //Variável que detecta o estado da chave óptica
int ID=0; //Variável para IDentificar qual cartão passou pelo leitor
int i=0; //Variável para contagem

void setup()
{
  pinMode(sensor,INPUT_PULLUP); //Define o pino digital 8(sensor) como entrada...
                                //...habilitando o resitor de pullup
  Serial.begin(9600); //Inicia a Serial
}

void loop()
{
    while(i<200) //Fica lendo a chave óptica por 200 milissegundo
    {

    estado = digitalRead(sensor); //Lê o pino digital 8(sensor) e guarda o estado na variável
estado
    if(estado == 0) //Se o estado da chave ótica for igual a 0(Objeto entre a fenda)
    {
      i=0; //Zera a variável de contagem
      ID = ID+2; //Incrementa +2 na ID
      while(estado == 0) //Enquanto o estado da chave óptica estiver em 0(Objeto entre a fenda)
```

```
  {
   estado = digitalRead(sensor); //Lê o estado da chave óptica
  }
}
delay(1); //Delay de 1 milissegundo
i++;  //Incrementa a variável para contagem do tempo
}

switch(ID) //Pega a variável ID
{
 case 4: //Se o valor na variável ID for igual a 4, libera acesso para a ID 4
        Serial.println("Marco Mello"); //Escreve no serial monitor o Usuário dessa ID
        Serial.println("ACESSO LIBERADO!"); //Escreve a mensagem "ACESSO LIBERADO!"
        Serial.println(""); //Pula uma linha
        Serial.println(""); //Pula uma linha
        break; //Sai da comparação
 case 6: //Se o valor na variável ID for igual a 6, libera acesso para a ID 6
        Serial.println("Francisco Lourenco"); //Escreve no serial monitor o Usuário dessa ID
        Serial.println("ACESSO LIBERADO!"); //Escreve a mensagem "ACESSO LIBERADO!"
        Serial.println(""); //Pula uma linha
        Serial.println(""); //Pula uma linha
        break; //Sai da comparação
 case 8: //Se o valor na variável ID for igual a 8, libera acesso para a ID 8
        Serial.println("Usuario 3"); //Escreve no serial monitor o Usuário dessa ID
        Serial.println("ACESSO LIBERADO!"); //Escreve a mensagem "ACESSO LIBERADO!"
        Serial.println(""); //Pula uma linha
        Serial.println(""); //Pula uma linha
        break; //Sai da comparação
 case 10: //Se o valor na variável ID for igual a 10, libera acesso para a ID 10
        Serial.println("Usuario 4"); //Escreve no serial monitor o Usuário dessa ID
        Serial.println("ACESSO LIBERADO!"); //Escreve a mensagem "ACESSO LIBERADO!"
        Serial.println(""); //Pula uma linha
        Serial.println(""); //Pula uma linha
        break; //Sai da comparação
}

ID=0; //Zera a variável ID
```

```
    i=0; //Zera a variável de contagem(tempo)
}
```

Referências:

https://www.sparkfun.com/products/9299

https://www.sparkfun.com/datasheets/Components/GP1A57HRJ00F.pdf

http://arduinobymyself.blogspot.com.br/2012/08/arduino-chave-optica...

27. Tutorial: Como utilizar a mini fotocélula (LDR)

Olá Garagistas! No tutorial de hoje, mostraremos como utilizar o LDR. Como demonstração, utilizaremos um Arduino para fazer o acionamento de uma lâmpada quando a luminosidade do ambiente estiver muito baixa, sem a necessidade de fazer algum acionamento por interruptores, como geralmente vimos acontecer com os postes de iluminação pública, quando anoitece ou quando o tempo está muito fechado.

Material Utilizado:

1x LDR

1x Arduino

1x Resistência 10KΩ

1x Módulo Relé

1x Lâmpada 110V

1x Cabo AC

1x Protoboard

Alguns Jumpers

1. O LDR

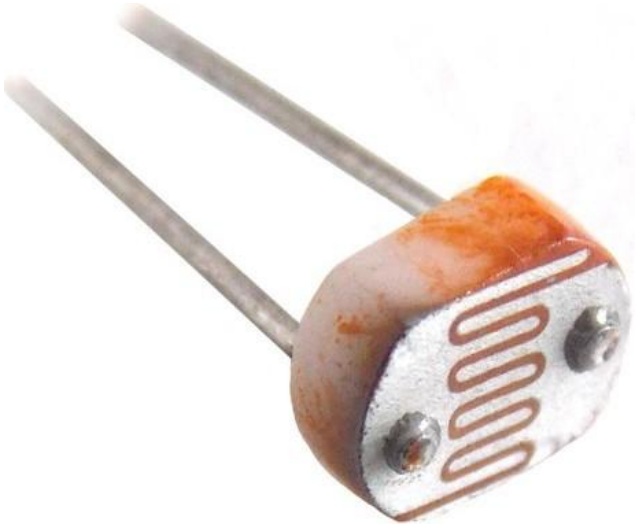


Imagem 1 - LDR

O LDR(Light Dependent Resistor) possui uma característica que faz com que sua resistência varie conforme a luminosidade incendida sobre ele. Isto possibilita a utilização deste componente para desenvolver um sensor que é ativado (ou desativado) quando sobre ele incidir uma certa luminosidade.

A resistência do LDR varia de forma inversamente proporcional à quantidade de luz incidente sobre
ele, isto é, enquanto o feixe de luz estiver incidindo sobre ele, o LDR oferece uma resistência muito baixa, e
quando este feixe é cortado, sua resistência aumenta.

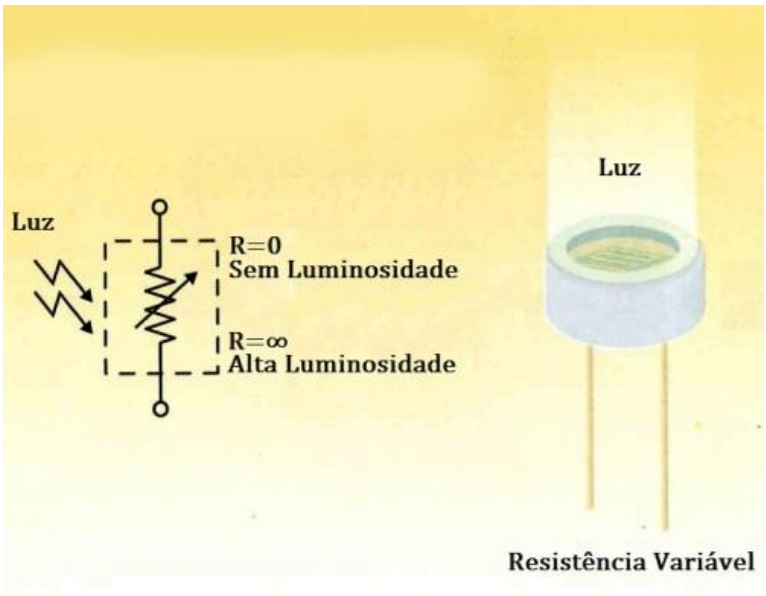


Figura 1 - Representação do Funcionamento do LDR

2. A Montagem

2.1) Módulo Relé

Faça as ligações de sua lâmpada com o módulo relé, conforme a imagem abaixo.

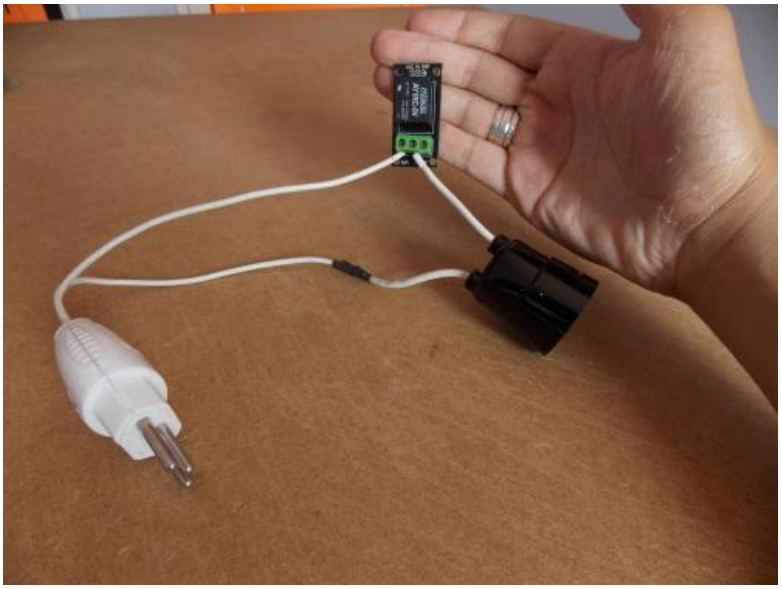


Imagem 2 - Montagem do Módulo Relé + Lâmpada

2.2) Circuito

Faça a montagem do circuito, conforme a figura abaixo.

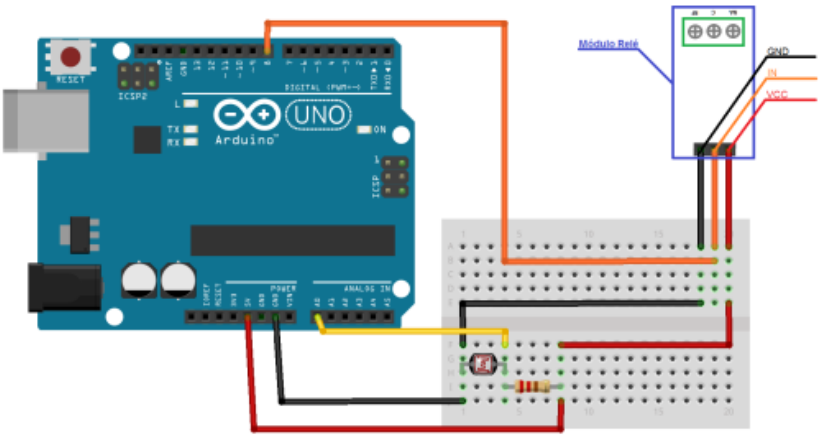


Figura 2 - Circuito da demonstração

3. O Sketch

```
#define LAMP 8 //Define o LAMP como 8

int LDR; //Variável para a leitura do LDR
int cont; //Variável utilizada para armazenar os valores lidos pelo sensor
int i; //Variável para contagem

void setup()
{
  pinMode(LAMP,OUTPUT); //Define o pino D8 como saída
  Serial.begin(9600); //Inicia a serial
}
void loop()
{
  LDR=0; //Zera a variável LDR
  for(i=0;i<=10;i++) //Faz 10 vezes a leitura do sensor
  {
  cont=analogRead(A0); //Lê o valor do sensor (LDR ligado ao pino A0) e guarda na variável
LDR
  LDR = LDR+cont; //Armazenamento na varíavel LDR o valor lido + o valor anterior
  delay(10); //Delay se 10 milissegundos
  }
  LDR=LDR/10; //Divide o valor armazenado por 10
  Serial.println(LDR); //Imprime o valor do LDR

  (LDR >= 400) ? (digitalWrite(LAMP,HIGH)) : (digitalWrite(LAMP,LOW));
  //Se o valor lido (luminosidade) for maior ou igual a 400, liga a lâmpada, senão desliga a
lâmpada
}
```

Referências:

http://www.labdegaragem.org/loja/index.php/mini-fotocelula.html

28. Tutorial - Alarme com sensores de temperatura e umidade

Mostramos neste tutorial um pouco mais sobre os sensor digitais de temperatura e umidade relativa RHT03 (DHT22) e o DHT11. Utilizando um alto-falante, vamos fazer um alarme sonoro que será acionado quando a temperatura ou a umidade passar de um valor determinado via software.

Lista de Materiais

1 x Arduino Uno Rev 3 ou Garagino Rev 1

1 x Sensor RHT03 (também conhecido como DHT22) ou Módulo sensor de temperatura e umidade (com DHT11)

1 x Alto-falante

1 x Resistor de 4K7 Ω

1 x Capacitor de 100µF / 25 V

Alguns jumpers

Bibliotecas necessárias

Iremos utilizar a biblioteca DHT-sensor-library distribuída pela Adafruit, ela já implementa todo o protocolo de comunicação utilizado por estes sensores e você pode baixa-la clicando neste link.

Os sensores DHT22 e DHT11

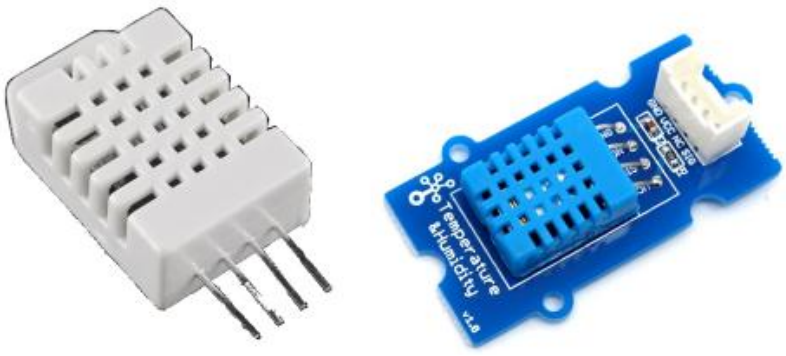


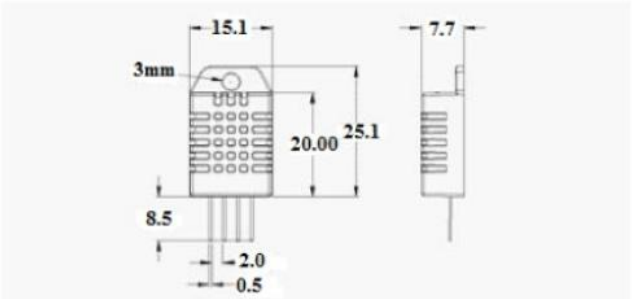
Figura 1 - DHT22/RHT03 (direita) e módulo que utiliza o DHT11 (esquerda)

Estes sensores de temperatura e de umidade relativa de baixo custo, já são calibrados e se comunicam com o Arduino através do protocolo MaxDetect (implementado pela biblioteca) que utiliza apenas 1 fio do microcontrolador para receber as informações.Por este motivo, eles não são muito rápidos, então recomenda-se amostragens com tempos superiores a 2 segundos neste dispositivos (como a latência de sistemas térmicos é alta, isto não deve afetar seus projetos).

O valor de temperatura é aferido através de um termistor NTC e a umidade relativa através de um sensor capacitivo (capacitor de polímero especial). Há também uma curva de compensação de temperatura que fica salva dentro de uma memória OTP (um tipo de memória ROM) e faz ajustes por toda a faixa de atuação deste sensor.

Eles podem ser alimentados de 3,3 a 5,5V e contam com um encapsulamento pequeno de 4 pinos

Abaixo você pode ver mais detalhes sobre o encapsulamento e os pinos.



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA--signal
3	NULL
4	GND

Figura 2 - Detalhes sobre a pinagem e encapsulamento do DHT22/RHt03

Diferença entre o DHT22 e o DH11

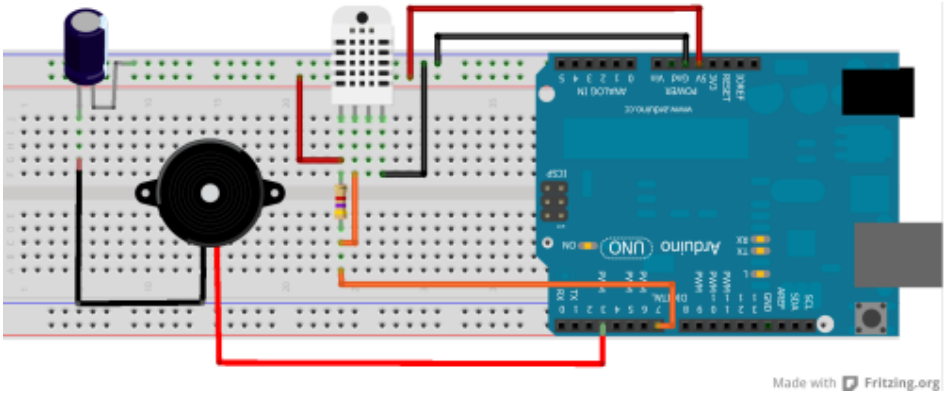
Abaixo segue lista com as principais diferenças entre eles:

	Sensores	
Itens	*DHT11*	*DHT22*
Precisão de Leitura de temperatura	± 2 %	± 0,5%
Precisão de Leitura de umidade	± 5%	± 2 a 5%
Faixa de Leitura de temperatura	0 a 50°C	-40 a 125°C
Faixa de leitura de Umidade	20 a 80 %	0 a 100%

Principais diferenças entre os dois sensores

O Circuito

Imagem com o circuito que utilizamos para fazer o alarme por temperatura e umidade com Arduino:



O Sketch

```
// Example de Alarme por temperatura e umidade
// Lab. de Garagem

#include "DHT.h"

#define DHTPIN 7     // O pino onde o DHT vai esta conectado

// Descomente linha com o sensor que vai utilizar
//#define DHTTYPE DHT11   // DHT 11
#define DHTTYPE DHT22   // DHT 22 - RHT03 - AM2302
//#define DHTTYPE DHT21   // DHT 21 - AM2301

DHT dht(DHTPIN, DHTTYPE);

#define ALARME 3 // pino onde o Alto-falante esta conectado

float tmax=50,tmin=0,temp, umid;

void setup()
{
  Serial.begin(9600);
  Serial.println("Alarme por temperatura e umidade");
  Serial.print("Digite a temperatura maxima: ");
  while(tmax == 50)
  {
```

```
    if (Serial.available() > 0)
    {
      tmax= Serial.parseFloat();
    }
  }
  Serial.println (tmax);
  Serial.print("Digite a temperatura minima: ");
  while(tmin == 0)
  {
    if (Serial.available() > 0)
    {
      tmin= Serial.parseFloat();
    }
  }
  Serial.println (tmin);
  dht.begin();
}

void loop()
{
  delay(2000); // delay para garantir no minimo 2s entre cada uma das leituras
  umid = dht.readHumidity(); // leitura de umidade
  temp = dht.readTemperature(); // leitura da temperatura
  if (isnan(temp) || isnan(umid)) //Verifica se é um valor válido
  {
    Serial.println("\nFalha ao ler o Sensor DHT\n");
    tone (ALARME,440,1000);
  }
  else
  {
    if (temp > tmin && temp < tmax)
    {
      noTone(ALARME);
      leituraSerial();
    }
    else
    {
      tone (ALARME,440,1000);
      Serial.println("PERIGO!!! TEMPERATURA FORA DO ESPERADO");
```

```
      Serial.print("Temperatura deve estar entre ");
      Serial.print(tmin);
      Serial.print(" e ");
      Serial.println(tmax);
      leituraSerial();
    }
  }
}

void leituraSerial()
{
      Serial.print("Umidade relativa: ");
      Serial.print(umid);
      Serial.print(" %\t");
      Serial.print("Temperature: ");
      Serial.print(temp);
      Serial.println(" *C\n");
}
```

Conclusão

Este sensores tem um baixo custo e são de fácil utilização com esta biblioteca. O datasheet solicita cuidado especial com a exposição direta aos raios ultra-violeta e também com vapor de produtos químicos.

29. Tutorial: Como soldar componentes eletrônicos

Este tutorial tem o intuito de mostrar como é **simples** e **fácil** soldar componentes eletrônicos e que não há **nenhum segredo**nesta habilidade que é de suma importância, pois, com um pouco de prática, você poderá soldar qualquer componente.

Lista de Materiais

Você pode utilizar qualquer um dos kits de ferramentas disponíveis em nossa loja:

Kit avançado

Kit intermediário

E vai precisar também de alguns componentes e se quiser pode usar o suporte para segurar placa (3ª mão).

O que é a soldagem

É uma prática simples, que tem como objetivo unir duas peças de maneira mecânica (para mante-las juntas) e ainda eletrônica, com o intuito de poder conduzir os elétrons entre os componentes e o circuito.

É uma habilidade indispensável, tanto para hobbistas como os usuários mais avançados que devem **domina-la** e para isto, devem praticar muito.

No vídeo acima, você verá na prática, como devem ser soldados componentes e abaixo verá mais informações sobre as ferramentas necessárias para soldar.

A solda

Existem muitos tipos de solda no mercado, a maioria delas são feitas com um composto de chumbo, estanho e resina (também conhecido como fluxo). Elas são comercializadas em carretéis, cartelas ou ainda em pequenos tubos. A solda mais usada em eletrônica tem um a liga de resina + 63% estanho e 37% chumbo.

Há também as solda sem chumbo (Lead Free) e com outras ligas que são para aplicações especias. A proibição do chumbo é devido a contaminação que ele pode causar ao meio ambiente quando uma placa de circuito e descartada sem os devidos cuidados. Se ingerido, o chumbo é tóxico ao ser humano, por isto, sempre lave as mãos após manusear a solda.

O ferro de soldar

Eles são a ferramenta básica para as soldas em eletrônica. Em sua grande maioria são elétricos e há desde os simples, que podem ser comprados segundo sua potência (30, 40 e 60W são os mais comuns), como os mais modernos com controle de temperatura, ponteiras intercambiáveis e outras calibrações (estes são chamados de estações de solda).

Abaixo temos vários modelos de ponteiras que podem ser trocadas em alguns modelos de estação de solda. Cada uma tem sua aplicação específica. A mais comum é a ponteira em forma de lápis (2ª da esquerda para direita). Evite de lixar as ponteiras de soldar.

É importante também ter uma esponja vegetal ou lã de bronze específica para limpeza das ponteiras de solda, assim como um suporte para que ele não fique solto no pela bancada.

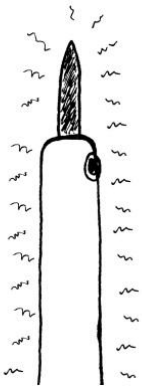
Outros acessórios

Alicate de corte: Para cortas as "pernas" dos componentes que ficam sobrando após a soldagem.

Suporte (3ª mão): Ajuda que você sempre precisa quando esta soldando, ele segura a placa enquanto você segura a solda e o ferro.

Cuidados

- O ferro chega a altas temperaturas (até 600 ºC), então sempre segure ele pela base plástica ou senão você pode se queimar. Proteja a mesa e afaste outros objetos que possam estar perto. **Tome cuidado!!!**

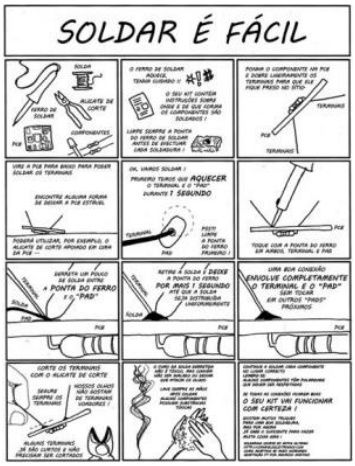


- Após manusear solda, não deixe de **lavar as mãos** pois ela contém chumbo que é prejudicial a saúde.

- Evite **inalar** a fumaça que da resina que esta presente na solda.

Como Soldar

Há também um resumo do que foi falado aqui com um passo a passo em quadrinhos, criado por Mitch Altman, com adaptação aos quadrinhos de Andie Nordgren e tradução de Maurício Martins.



Clique aqui para fazer o download.

Conclusão

É isto pessoal!!! É simples, rápido e com um pouco de treino vocês soldaram da maneira perfeita. Um bom método para treino é com o uso de uma placa velha, onde você pode soltar e recolocar os componentes algumas vezes.

Referências

https://www.sparkfun.com/tutorials/106

http://www.soldabest.com.br/tecnicas_soldagem.htm

https://learn.sparkfun.com/tutorials/how-to-solder---through-hole-soldering/what-is-solder

http://cornfieldelectronics.com/cfe/projects.php?PHPSESSID=v8r3s8e7j14ko8h34o2vj67vu4

30. Tutorial: Instalando Lubuntu Linux no Android Mini PC

Neste tutorial, vamos instalar no Android Mini PC, uma distribuição Linux baseada no Lubuntu (Ubuntu + a interface gráfica LXDE). A instalação será feita em um cartão SD, mantendo com isto o sistema operacional Android original na memória interna do aparelho. Ela foi desenvolvida por um usuário do fórum miniand e tem versões para monitores com os modos de vídeo 720p e 1080p.

Materiais Utilizados

1 x Android Mini PC

1 x Cartão SD (de boa qualidade)

1 x Monitor com entrada HDMI

1 x Teclado

1 x Mouse

Download

Baixe uma das duas imagens do Lubuntu 12.04 v4 correspondente a resolução de seus monitor. Ambas são compatíveis com o nosso Android Mini PC.

- Para monitores com vídeo 720p e Android Mini PC com 1GB de RAM

- Para monitores com vídeo 1080p e Android Mini PC com 1GB de RAM

Instalação

Tenha certeza de fazer o ***backup dos arquivos*** do cartão SD antes de começar o procedimento e siga o passo a passo de acordo com seu sistema operacional.

- ***Windows***

1. Descompacte a imagem que vai utilizar utilize o gratuito 7-zip para isto.

2. Coloque o cartão SD no computador e com o software Win32 Disk Imager selecione a unidade de disco que o cartão foi identificado e o arquivo da imagem descompactada e clique em Write.

3. Após a transferência coloque o cartão, mouse e teclado no Android mini PC e tudo estará pronto para usar.

- ***MAC e Linux***

1. Descompacte a imagem com o "*p7zip"* usando o comando:
   p7zip -d lubuntu-desktop-12.04-4-720p-1GB-miniand.com.img.7z

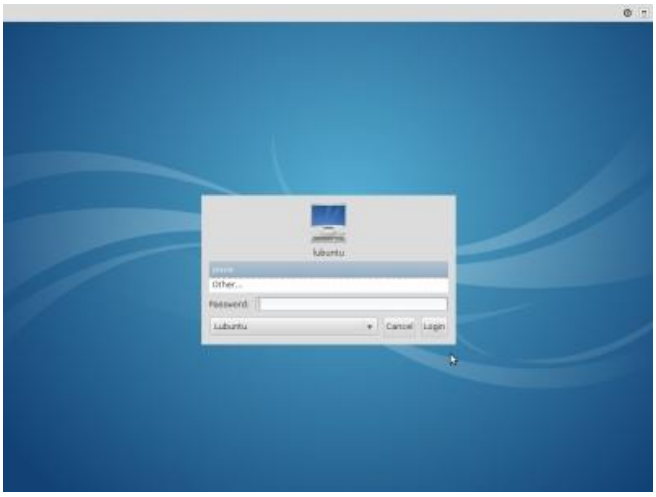
2. Coloque o cartão SD e localize ele com o comando "*fdisk"*:
   sudo fdisk -l.
   Ele vai estar listado como algo assim: */dev/sdd*

3. Agora utilize o "*dd*" para copiar a imagem para o cartão:
   dd if=lubuntu-desktop-12.04-4-720p-1GB-miniand.com.img of=/dev/sdd

4. Verifique se a transferência acabou com: *sudo sync*

5. Pronto, coloque o cartão Android Mini PC e o linux será inicializado!!!

Utilizando o Lubuntu



Ao iniciar o sistema, uma tela semelhante com a imagem acima será apresentada. Uma **senha é solicitada** e tem o mesmo da máquina que é: *miniand*

O Lubuntu conta com vários aplicativos pré-instalados e você pode baixar outros pelo gerenciador de pacotes incluso na distro. A placa de rede sem fio e o dispositivo de áudio já vem instalados e prontos para usar.

Outras Dicas

- Caso o sistema fique reiniciando pode ser que seu cartão SD é de baixa qualidade e não atenda os requisitos de velocidade que o sistema necessita. Neste caso, compre um cartão SD de qualidade.
- Se a tela não ficar cheia em seu monitor, verifique a tecla de zoom de seu monitor e ajuste para que fique com uma boa visualização.

Não quero mais o Lubuntu, e agora?

Desligue o aparelho e remova o cartão SD, com isto, o sistema Android original vai ser inicializado. A instalação é como um live CD e só funciona quando você esta com o cartão conectado.

Conclusão

É simples o processo de instalação dos arquivos no cartão SD e agora você tem mais uma opção de sistema operacional para o Android Mini PC.

Referências

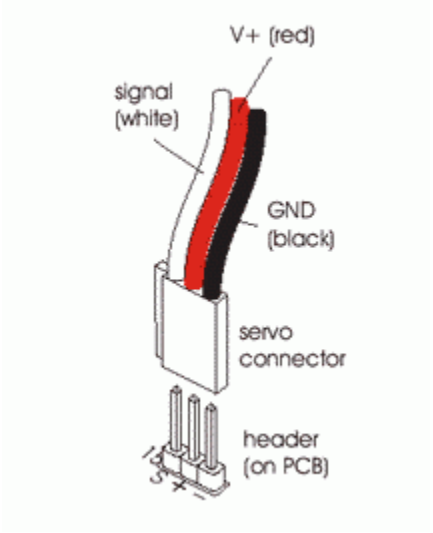
https://www.miniand.com/forums/forums/2/topics/1

http://www.lubuntu.net/

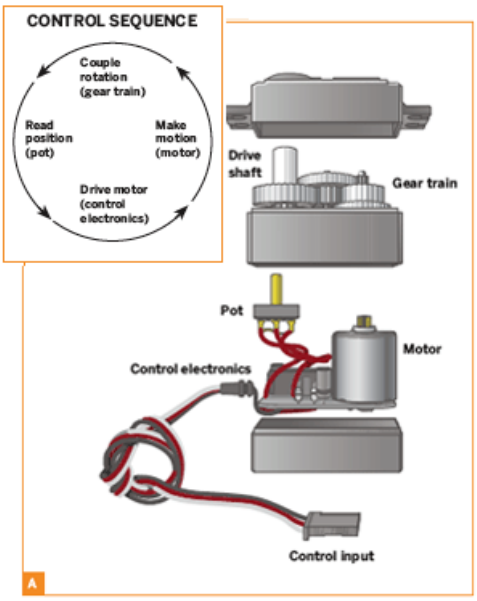
31. Tutorial Arduino com ServoMotor

Servomotores são motores de posição frequentemente usados em aeromodelos, automodelos, e outros veículos radio-controlados em escala reduzida e também são muito utilizados em automação e robótica. Por este motivo, são fáceis de serem encontrados no mercado especializado de radio-controles.

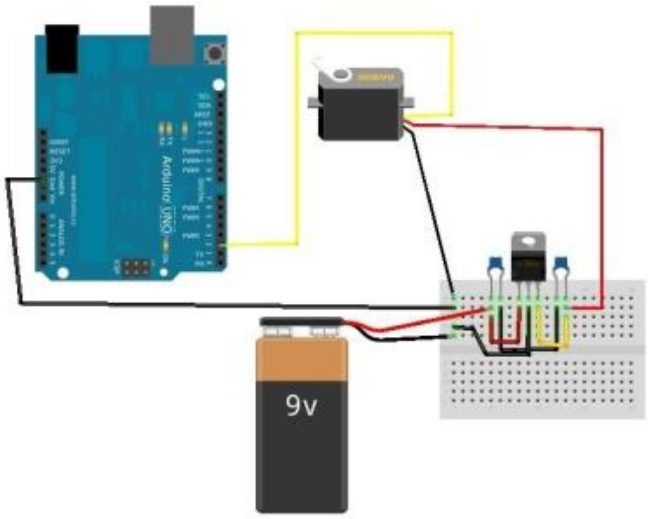
Um servomotor tipo RC consiste essencialmente em um motor de corrente contínua com um circuito de controle de posição acoplado. Os servomotores não dão uma volta completa em seu eixo, eles possuem uma faixa ou de 90 ou 180 graus em seu eixo. Do servomotor sai três cabos: preto, vermelho e branco ou amarelo. Os cabos preto e vermelho são para alimentação e o branco ou amarelo é o cabo de controle.



Internamente, para que o servomotor funcione, consiste de um circuito de controle que recebe um sinal de controle para que o servomotor se posicione em um determinado ângulo.



Faça a seguinte ligação:



Os componentes na protoboard são: um regulador 7805 e dois capacitores ceramicos de 10uF. Este circuito foi feito para alimentar o servo externamente. É aconselhűel fazer isso, pois o regulador da placa Arduino pode queimar.

Agora, abra a IDE do Arduino e abra em File/Examples/Servo/Sweep. Conecte o Arduino e depois clique em UPLOAD.

Quando terminar, o servo girará de 0 a 180 graus e depois de 180 a 0 graus.

Referências:

http://arduino.cc/playground/

http://loja.labdegaragem.com.br/servo-motor-pequeno.html

http://www.sparkfun.com/products/9065

32. Tutorial Arduino com Easydriver + Motor de Passo

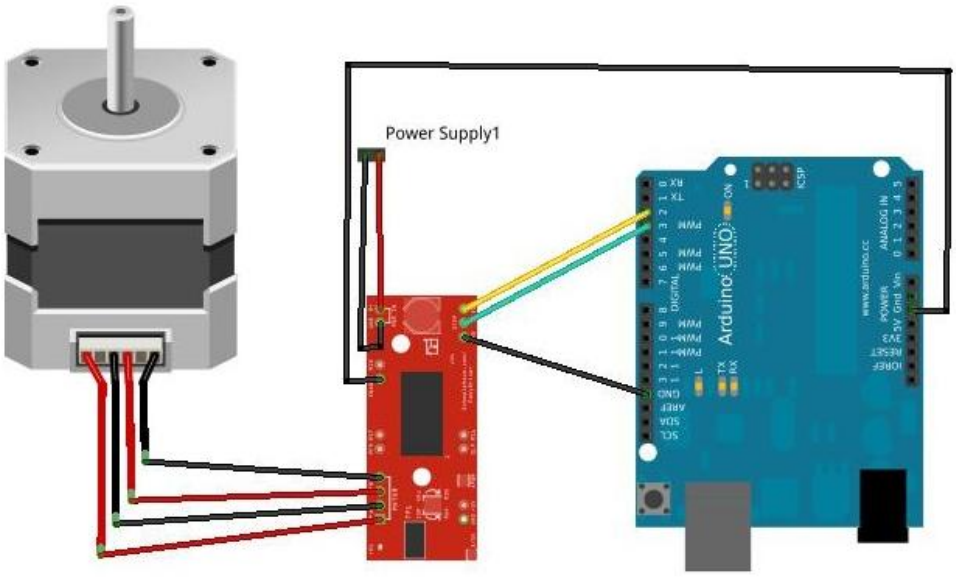
A placa Arduino tem muitas aplicações incluindo controle de motores de corrente continua, Servo-motores e Motores de Passo.

Um motor de passo é um tipo de motor elétrico usado quando algo tem que ser posicionado muito precisamente ou rotacionado em um ângulo exato.

Neste tutorial vamos mostrar como controlar um motor de passo utilizando a placa Arduino.

Para controlar um motor de passo é necessário uma ponte H ou uma placa controladora. Aqui vamos utilizar o Easydriver que é uma placa controladora, pois é de fácil aplicação.

Para utilizá-lo, faça a seguinte ligação:



Conecte o Power Supply1 em uma fonte de 7V a 30V.

Agora abra a IDE do Arduino e cole a seguinte programação:

```
int dir= LOW;
int stepp=LOW;
long previousMillis = 0;
long currentMillis = millis();
long steptime=500;
int x=0;
char c=0;
void setup()
{
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
Serial.begin(9600);
}
void loop()
{
if(Serial.available()>0)
{
c=Serial.read();
if(c=='a')
{
dir=HIGH;
}
if(c=='z')
{
dir=LOW;
}
}
if(currentMillis - previousMillis > steptime)
{
previousMillis = currentMillis;
if(stepp==LOW)
{
stepp=HIGH;
}
else
{
stepp=LOW;
}
digitalWrite(2,dir);
digitalWrite(3,stepp);
}
}
```

Clique em UPLOAD e depois abra o Serial Monitor.

Com o Serial Monitor aberto, digite 'a' e aperte enter. O motor de passo rodará para o outro lado. Se não, digite 'z' e aperte enter.

Referências:

http://arduino.cc/playground/

http://loja.labdegaragem.com.br/easydriver-stepper-motor-driver.html

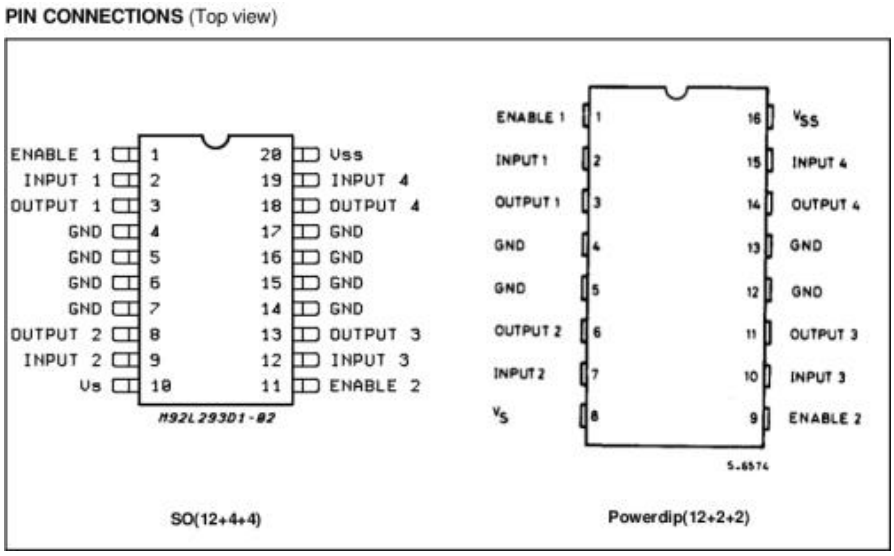
http://loja.labdegaragem.com.br/motor-de-passo-com-cabo.html

33. Tutorial sobre como utilizar motor DC com L293D (ponte-H) e Arduino

Neste tutorial vamos mostrar como utilizar o L293D para controlar motores de corrente contínua (DC) com Arduino.

O L293D é basicamente uma ponte-H em um circuito integrado. Pode-se controlar motores de até 36V de alimentação como mostra no datasheet. Cuidado, que o L293D aguenta corrente constante de 600mA e corrente pico em torno de 1.2A. Portanto não coloque motores DC que exijam mais do que 600mA.

Primeiramente, vamos olhar no datasheet para saber a pinagem do L293D:



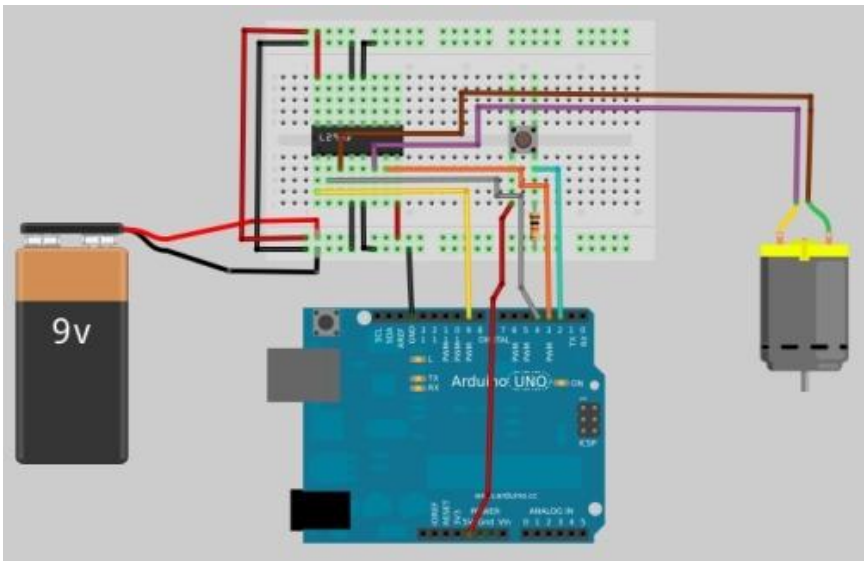
Vendo a pinagem do L293D, é possível ligar até 2 motores em um único L293D, porém é necessário que a fonte de alimentação suporte o consumo de corrente. Neste tutorial iremos demonstrar apenas um motor DC.

Agora, vamos ver como fazer o motor DC girar de um lado e para outro:

Pin 1	Pin 2	Pin 7	Function
High	Low	High	Turn clockwise
High	High	Low	Turn anti-clockwise
High	Low	Low	Stop
High	High	High	Stop
Low	Not applicable	Not applicable	Stop

Se os pinos 1 e 7 estiverem em alto(HIGH) e o pino 2 estiver em baixo(LOW), o motor girará para um lado. Se os pinos 1 e 2 estiverem em alto(HIGH) e o pino 7 estiver em baixo(LOW), o motor girará para o outro lado. E assim por diante.

Utilizando um botão, um resistor de 10K ohm, um motor DC de 9V, uma bateria de 9V, L293D e um Arduino, vamos para a montagem:



Na montagem estou utilizando um motor DC de 9V e alimentando o L293D com uma bateria de 9V. Antes de ligar a alimentação do motor, veja qual é a sua alimentação, caso contrário irá queimá-lo.

Depois de feito a montagem, abra a IDE do Arduino e passe a programação exemplo:

```
int switchPin = 2; // switch input
int motor1Pin1 = 3; // pin 2 on L293D
int motor1Pin2 = 4; // pin 7 on L293D
int enablePin = 9; // pin 1 on L293D
void setup() {
// set the switch as an input:
pinMode(switchPin, INPUT);
// set all the other pins you're using as outputs:
pinMode(motor1Pin1, OUTPUT);
pinMode(motor1Pin2, OUTPUT);
pinMode(enablePin, OUTPUT);
// set enablePin high so that motor can turn on:
digitalWrite(enablePin, HIGH);
}
void loop() {
// if the switch is high, motor will turn on one direction:
if (digitalRead(switchPin) == HIGH) {
digitalWrite(motor1Pin1, LOW); // set pin 2 on L293D low
digitalWrite(motor1Pin2, HIGH); // set pin 7 on L293D high
}
// if the switch is low, motor will turn in the opposite direction:
else {
digitalWrite(motor1Pin1, HIGH); // set pin 2 on L293D high
digitalWrite(motor1Pin2, LOW); // set pin 7 on L293D low
}
}
```

Agora, selecione a versão da placa Arduino (UNO, Duemilanove, etc), depois em selecione a porta a qual a placa Arduino está conectada (COMx, ttyUSBx, ttyACMx, etc) e clique em "UPLOAD".

O motor DC irá girar para um lado. Se apertar o botão, o motor DC girará para o outro lado.

Referências:

http://luckylarry.co.uk/arduino-projects/control-a-dc-motor-with-ar...

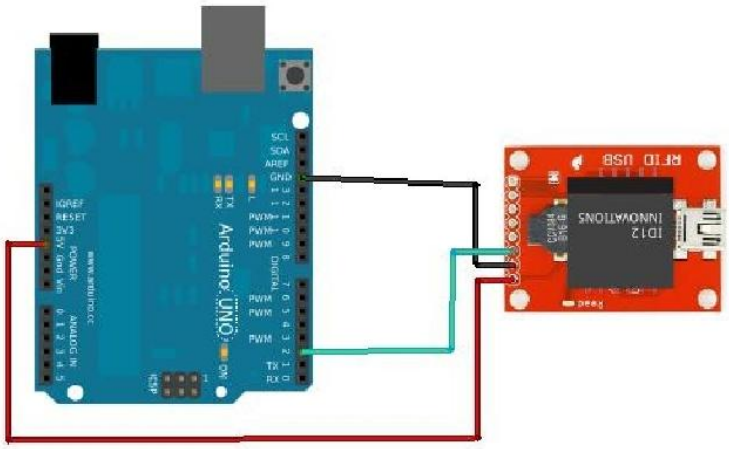
http://arduino.cc/playground/Main/InterfacingWithHardware

http://www.datasheetcatalog.org/datasheet/stmicroelectronics/1330.pdf

34. Tutorial sobre RFID com Arduino

O Leitor RFID utilizado neste tutorial é o Starter Kit da SparkFun disponível na Loja LdG (http://loja.labdegaragem.com.br/rfid-starter-kit.html).

Conecte o Leitor ID12 na placa fornecida e faça a seguinte ligação:



Agora, abra a IDE do Arduino e coloque a seguinte programação para o Arduino:

```
 #include <SoftwareSerial.h>

SoftwareSerial rfidReader(2, 3); //Pin 2 - RX /Pin 3 - TX
char c= 0; // value read for serial port
char temp[20]=0;
int z=0;
int y=0;
char* cards[] = { // put your ID Card
  //Example:
  "3E00274AD9"
};

void setup() {
    Serial.begin(9600); //setup the Serial speed
    rfidReader.begin(9600);  //setup the Rfid reader speed
}

void loop () {

  z=0;
  if(Serial.available() > 0) {

  c = Serial.read(); // read from the serial port
  temp[z]=c; //put the character on temporary variable
  Serial.print(c, BYTE); // Use Serial.write(c) for IDE 1.0 and print it to the monitor
  z=z+1;
  }
  for(int x=0;x<13;x++)
```

```
 {
  if(temp[0]==cards[0])
 {
  y=y+1;
 }

 }
if(y==12)
 {
  Serial.println("ok");
 Serial.print("Valid");
 }
else if (y!=12)
 {
 Serial.println(“Not Valid”);
 }
}
```

Clique em Upload e depois abra o Serial Monitor e passe o cartão com a ID que foi escrita acima. Se estiver certo, aparecerá “ok” e depois “Valid”, senão aparecerá “Not Valid”.

Referências:

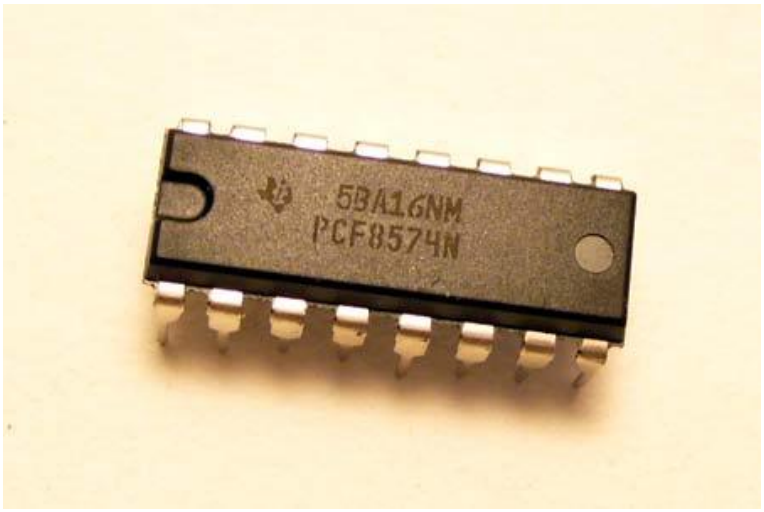
http://loja.labdegaragem.com.br/rfid-starter-kit.html

http://arduino.cc/playground/Code/ID12

http://hcgilje.wordpress.com/resources/rfid_id12_tagreader/

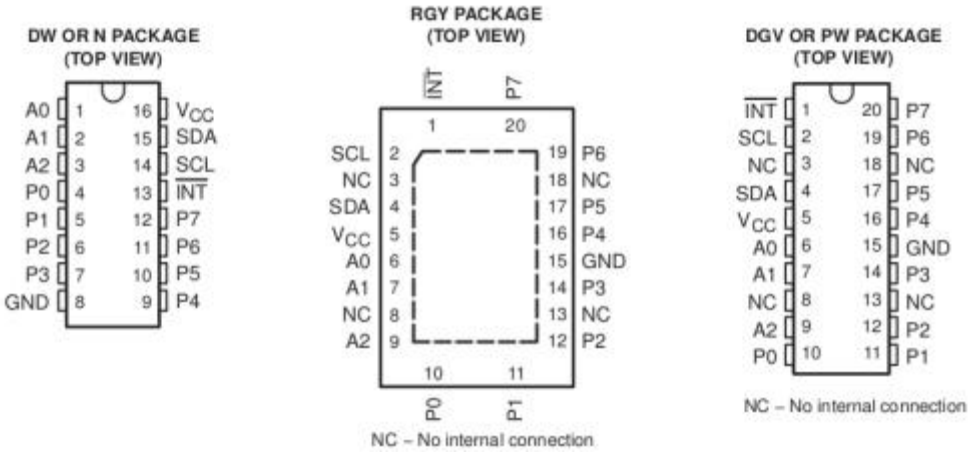
http://www.sparkfun.com/products/9875

35. Tutorial sobre PCF8574 e como utilizá-lo com Arduino



O circuito integrado PCF8574 é um expansor de portas entradas/saídas de 8bits por interface de comunicação I2C(apenas duas portas: serial clock(SCL) e serial data(SDA)). Com ele é possível expandir as portas de um microcontrolador utilizando apenas essas duas portas para 8 portas. Se utiliza endereçamento para se comunicar com os dispositivos conectados a ele. E uma alimentação de 2.5V a 6V.

Olhando o datasheet podemos ver a pinagem do PCF8574:



Onde: A0,A1,A2 são o pinos de endereçamento

P0,P1,P2,P3,P4,P5,P6,P7 são as saídas em bits(cada um é um bit)

SDA,SCL são as entradas de comunicação com o microcontrolador.

Para fazê-lo funcionar é necessário determinar o endereçamento do pcf8574, novamente no datasheet podemos ver quais endereçamentos são disponíveis:

ADDRESS REFERENCE

INPUTS			I²C-BUS SLAVE ADDRESS
A2	A1	A0	
L	L	L	32 (decimal), 20 (hexadecimal)
L	L	H	33 (decimal), 21 (hexadecimal)
L	H	L	34 (decimal), 22 (hexadecimal)
L	H	H	35 (decimal), 23 (hexadecimal)
H	L	L	36 (decimal), 24 (hexadecimal)
H	L	H	37 (decimal), 25 (hexadecimal)
H	H	L	38 (decimal), 26 (hexadecimal)
H	H	H	39 (decimal), 27 (hexadecimal)

onde: L significa LOW ou 0V(GND)

H significa HIGH ou VCC

Cuidado!! O PCF8574 que foi usado neste exemplo foi o PCF8574 (sem letra no final). O PCF8574, o PCF8574N e o PCF8574P tem o mesmo endereçamento (32). Agora, o PCF8574A e o PCF8574AP começam com o endereço 56!

Vendo a definição de interface do PCF8574 no datasheet, podemos ver como este funciona:

INTERFACE DEFINITION

BYTE	BIT							
	7 (MSB)	6	5	4	3	2	1	0 (LSB)
I²C slave address	L	H	L	L	A2	A1	AO	R/$\overline{W}$
I/O data bus	P7	P6	P5	P4	P3	P2	P1	P0

De acordo com o endereçamento que foi determinado, o PCF8574 vai receber ou mandar a informação de acordo com o valor no I/O data bus. Como cada pino é um bit(valor binário) então o PCF tem um total de 255 de entrada ou saída.

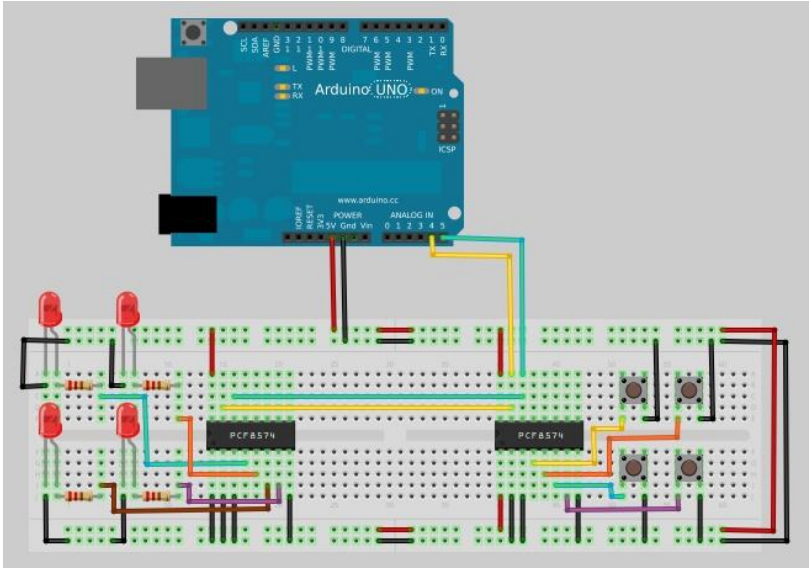
Por exemplo: Se o PCF8574 estiver como receptor de dados, este vai receber o valor 32 pelos SDA e SCL e jogar na saída este valor em binário, ativando apenas o P5 em HIGH.

Se for o valor 48, o PCF8574 vai ativar os pinos P5 e P4 ao mesmo tempo. Isto é, o PCF8574 manda de uma vez o valor que foi recebido.

Agora se o PCF8574 estiver como emissor de dados, é simplesmente o contrário.

Detalhe que o PCF8574 pode ser emissor e receptor, isto depende da montagem e da programação utilizada no microcontrolador. Mas não pode ser utilizado como emissor e receptor ao mesmo tempo!

Agora vamos para a parte prática, utilizando o Arduino, dois PCF8574 conectado a ele e uns leds:



O PCF8574 da esquerda será o receptor, que receberá os comandos e acenderá os leds e o PCF8574 da direita será o emissor e se um botão for apertado, acenderá um led.

O código para o Arduino é este:

```
#include <Wire.h>
byte x=0;
byte y=0;
void setup()
{
Wire.begin();
}
void loop()
{
Wire.requestFrom(33,1);       //Se PCF8574A - mude de 33 para 57
if(Wire.available())     //If the request is available
{
x=Wire.receive();       //Receive the data
}
if(x<255)                  //If the data is less than 255
{
if (x==254) { y = 0; } //P0
if (x==253) { y = 2; } //P1
if (x==247) { y = 8; } //P3
if (x==251) { y = 4; } //P2
}                                  //Se PCF8574A mude 32 para 56
Wire.beginTransmission(32);     //Begin the transmission to PCF8574
Wire.send(y);                        //Send the data to PCF8574
Wire.endTransmission();          //End the Transmission
}
```

Na programação, quando um botão for apertado, o PCF8574, com o endereço 33, irá mandar o dado para o Arduino, o Arduino irá receber, converter e depois mandará o dado para o PCF com endereço 32. E pronto!

Esperamos que tenham gostado! Boa sorte e boa diversão!

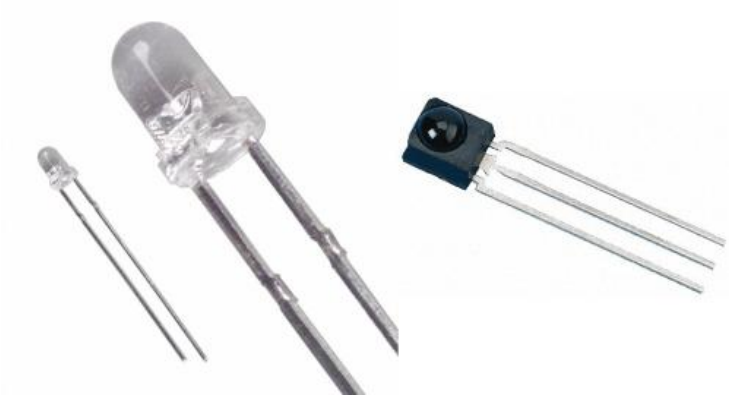
Referências:

http://arduino.cc/en/Reference/Wire

http://www.labdegaragem.org/loja/index.php/33-componentes/expansor-...

http://www.datasheetcatalog.org/datasheet2/b/0fjjhr6h643gldhx3o1rgk...

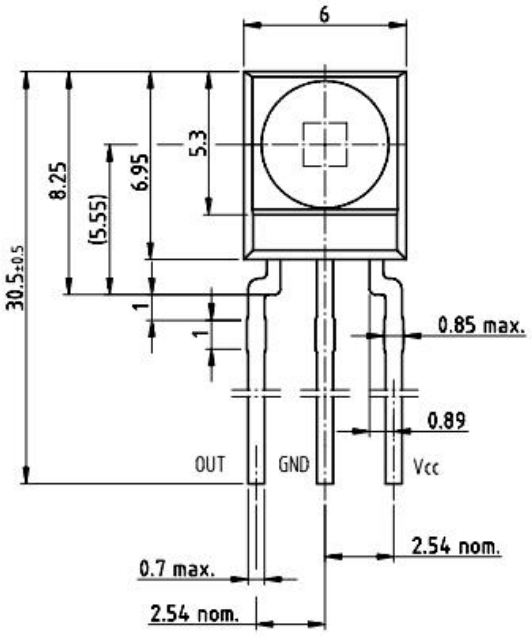
36. Tutorial emissor e receptor Infra-vermelho com Arduino



Neste tutorial, mostraremos como implementar o emissor e receptor de infravermelho com Arduino.

Para este tutorial você irá precisar de dois arduinos, um emissor de infravermelho, um receptor de infravermelho, um resistor de 200 Ohm e pushbuttons. Você pode utilizar o controle remoto de sua tv ao invés do emissor!

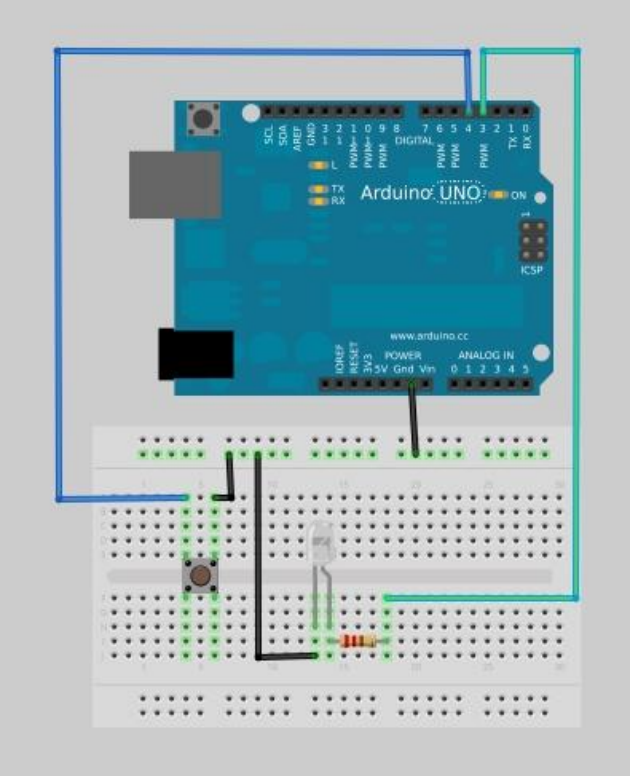
Antes de mais nada, é preciso ver a pinagem do Receptor IR está demonstrada abaixo:



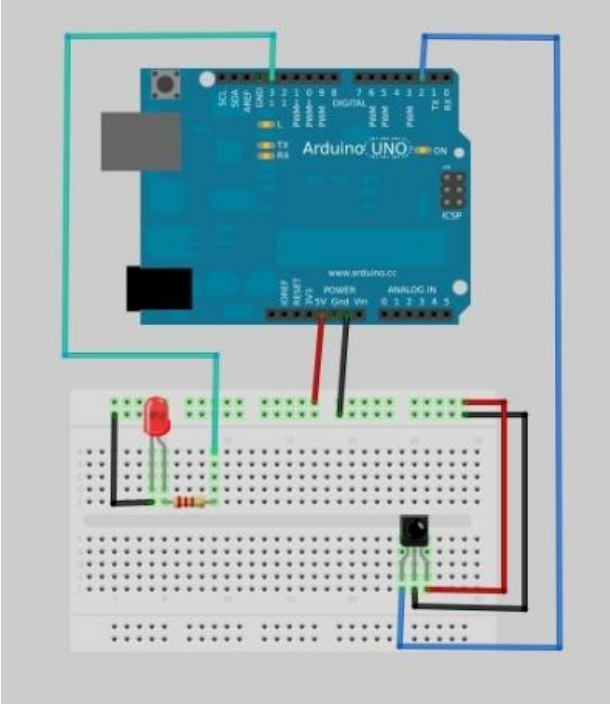
Agora, podemos fazer as ligações para o emissor e receptor IR.

A figura abaixo mostra as ligações a serem feitas:

Esquema de ligações para o Emissor IR:



Esquema de ligações para Receptor IR:



Agora, podemos ir para a programação. Antes de mais nada, baixe a bilbioteca para controlar o IR clicando aqui.

Extraia a bilbioteca na pasta "libraries" da IDE do Arduino. Para a versão 1.0 da IDE do Arduino, abra o arquivo "IRRemoteInt.h" com um editor de texto(bloco de notas, gedit, etc). Dentro do editor de texto, troque a seguinte linha:

"#include <WProgram.h>" para "#include <Arduino.h>".

As programações abaixo foram retiradas e modificadas do livro Arduino Cookbook. Clique aqui para maiores informações.

Agora abra a IDE do arduino e passe a seguinte programação para o arduino emissor IR:

```
/*
irSend sketch
this code needs an IR LED connected to pin 3
and 5 switches connected to pins 4 - 8
*/
#include <IRremote.h>
// IR remote control library
const int numberOfKeys = 1;
const int firstKey = 4;
// the first pin of the 5 sequential pins connected to buttons
boolean buttonState[numberOfKeys];
boolean lastButtonState[numberOfKeys];
long irKeyCodes[numberOfKeys] = {
0x18E758A7, //0 key
};
IRsend irsend;
void setup()
{
for (int i = 0; i < numberOfKeys; i++){
buttonState[i]=true;
lastButtonState[i]=true;
int physicalPin=i + firstKey;
pinMode(physicalPin, INPUT);
digitalWrite(physicalPin, HIGH); // turn on pull-ups
}
Serial.begin(9600);
}
void loop() {
for (int keyNumber=0; keyNumber<numberOfKeys; keyNumber++)
{
int physicalPinToRead=keyNumber+4;
buttonState[keyNumber] = digitalRead(physicalPinToRead);
if (buttonState[keyNumber] != lastButtonState[keyNumber])
{
if (buttonState[keyNumber] == LOW)
{
irsend.sendSony(irKeyCodes[keyNumber], 32);
Serial.println("Sending");
}
lastButtonState[keyNumber] = buttonState[keyNumber];
}
}
}
```

Clique em "UPLOAD" e passe a programação.
Agora para o Arduino Receptor IR, a programação está abaixo:

```
/*
IR_remote_detector sketch
An IR remote receiver is connected to pin 2.
The LED on pin 13 toggles each time a button on the remote is pressed.
*/
#include <IRremote.h> //adds the library code to the sketch
const int irReceiverPin = 2; //pin the receiver is connected to
const int ledPin = 13;
IRrecv irrecv(irReceiverPin); //create an IRrecv object
decode_results decodedSignal; //stores results from IR detector
void setup()
{
pinMode(ledPin, OUTPUT);
irrecv.enableIRIn();
}
boolean lightState = false;
unsigned long last = millis();
// Start the receiver object
//keep track of whether the LED is on
//remember when we last received an IR
void loop()
{
if (irrecv.decode(&decodedSignal) == true) //this is true if a message has been received
{
if (millis() - last > 250) {
//has it been 1/4 sec since last message
lightState = !lightState;
//toggle the LED
digitalWrite(ledPin, lightState);
}
last = millis();
irrecv.resume();
// watch out for another message
}
}
```

Agora que você passou as programações, você pode testar! Ao clicar o botão do Arduino emissor e direcionando o LED infravermelho na direção do receptor, o LED do Arduino Receptor acenderá. Ao clicar o botão novamente, o LED do Arduino receptor apagará! E é isso aí!! Esperamos que tenham gostado! Se tiverem dúvidas, postem aqui neste blog! Caso tiverem sugestões, vocês podem sugerir aqui neste POST e vocês podem ver os outros tutoriais que fizemos clicando aqui e os projetos clicando aqui! Até a próxima!!

Referências:

http://labdegarag1.lojatemporaria.com/kits/starter-kit-com-arduino-...

http://labdegarag1.lojatemporaria.com/livros-tecnicos/arduino-cookb...

http://www.arcfn.com/2009_08_01_archive.html

http://arduino.cc/playground/

http://www.datasheetcatalog.org/datasheets/2300/301522_DS.pdf

37. Tutorial LCD com Arduino

Olá Garagistas! Neste tutorial, vamos mostrar como montar o circuito e como utilizar o LCD com o Arduino. Faremos também um exemplo criado pelo LdG, onde escrevemos na tela do LCD:

"*Ola Garagista!*

LabdeGaragem 1"

Sendo o "Olá Garagista" uma saudação, "LabdeGaragem" nossa assinatura e o "1" escrito na mensagem, está representando uma contagem dos segundos passados após o início do programa. Vamos começar!

Materias Utilizados:

1x Arduino UNO

1x LCD 16x2

1x Potenciômetro 10KΩ

1x Protoboard

Alguns Jumpers

1. O Funcionamento

1.1) LiquidCrystal.h: A "LiquidCrystal.h" é a biblioteca que iremos utilizar para fazer o controle do LCD, utilizando o Arduino. Ela é bastante útil, pois possui funções que nos auxilia nas configurações e tratamento dos dados a serem enviados ao LCD. Na lista abaixo, estão contidas as funções que podem ser utilizadas pela biblioteca. Para mais detalhes sobre uma certa função, basta somente clicar nela que você será redireciona para o site da Arduino.cc, onde você vai encontrar o que essa função faz, e a sintaxe para se utilizar ela.

- LiquidCrystal()
- begin()
- clear()
- home()

- setCursor()
- write()
- print()
- cursor()
- noCursor()
- blink()
- noBlink()
- display()
- noDisplay()
- scrollDisplayLeft()
- scrollDisplayRight()
- autoscroll()
- noAutoscroll()
- leftToRight()
- rightToLeft()
- createChar()

1.2) A Pinagem do LCD: Na parte traseira do LCD você encontrará a numeração dos pinos, encontrando apenas o número 1(à direita) e o número 16(à esquerda), indicando que a contagem dos pinos vai de 1 a 16 da direita, para a esquerda. Na tabela abaixo, você poderá encontrar a descrição de cada pino do LCD:

Pino	Símbolo	Função
1	VSS	GND(Alimentação)
2	VDD	5V(Alimentação)
3	V0	Ajuste de Contraste
4	RS	Habilida/Desabilita Seletor de Registrador
5	R/W	Leitura/Escrita
6	E	Habilita Escrita no LCD
7	DB0	Dado
8	DB1	Dado
9	DB2	Dado
10	DB3	Dado
11	DB4	Dado
12	DB5	Dado
13	DB6	Dado
14	DB7	Dado
15	A	5V(Backlight)
16	K	GND(BackLight)

1.3) Datasheet do LCD: No Datasheet podemos encontrar informações sobre o LCD, e saber quais os pinos vamos utilizar para fazer a montagem do circuito do LCD com o Arduino! Na nossa demonstração, não utilizaremos apenas os pinos 7, 8, 9 e 10. Após consultar o Datasheet, faça a montagem do circuito.

2. A Montagem

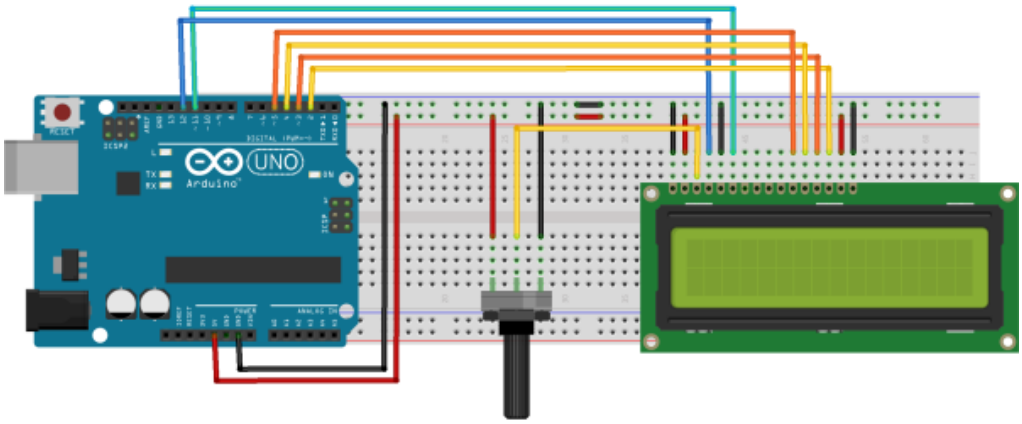


Figura 1 - Montagem do Circuito: LCD + Arduino

3. O Sketch

```
#include <LiquidCrystal.h> //Inclui a biblioteca do LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Configura os pinos do Arduino para se comunicar com o LCD

int temp; //Inicia uma variável inteira(temp), para escrever no LCD a contagem do tempo

void setup()
{

lcd.begin(16, 2); //Inicia o LCD com dimensões 16x2(Colunas x Linhas)
lcd.setCursor(0, 0); //Posiciona o cursor na primeira coluna(0) e na primeira linha(0) do LCD
lcd.print("Ola Garagista!"); //Escreve no LCD "Olá Garagista!"
lcd.setCursor(0, 1); //Posiciona o cursor na primeira coluna(0) e na segunda linha(1) do LCD
lcd.print("LabdeGaragem"); //Escreve no LCD "LabdeGaragem"

}

void loop()
{

lcd.setCursor(13, 1); //Posiciona o cursor na décima quarta coluna(13) e na segunda linha(1) do LCD
lcd.print(temp); //Escreve o valor atual da variável de contagem no LCD
delay(1000); //Aguarda 1 segundo
temp++; //Incrementa variável de contagem

if(temp == 600) //Se a variável temp chegar em 600(10 Minutos),...
 {
 temp = 0; //...zera a variável de contagem
```

```
  }

}
```

Copie e cole o sketch, que está dentro das linhas pontilhadas na Arduino IDE, selecione a versão da sua placa *Arduino*, selecione a *porta serial* e clique em *UPLOAD*. Você verá que após o upload do sketch o LCD escreverá as mensagens e começará a fazer a contagem dos segundos de programa iniciado. Pronto, seu LCD está funcionando!

E é isso aí! Esperamos que tenham gostado deste tutorial! Caso tenham dúvidas sobre o tutorial, vocês podem postar aqui mesmo! Vocês também podem sugerir tutoriais, clicando aqui, podem ver nossos tutoriais anteriores clicando aqui e os projetos, clicando aqui. Até a próxima Garagistas!

Referências:

http://www.labdegaragem.org/loja/index.php/lcd-16x2.html

http://www.labdegaragem.org/loja/index.php/starter-kit-com-arduino-uno-rev-3-original.html

http://www.sparkfun.com/datasheets/LCD/GDM1602K.pdf

http://arduino.cc/en/Tutorial/LiquidCrystal

38. Tutorial LCD 16x2 com PCF8574 e Arduino

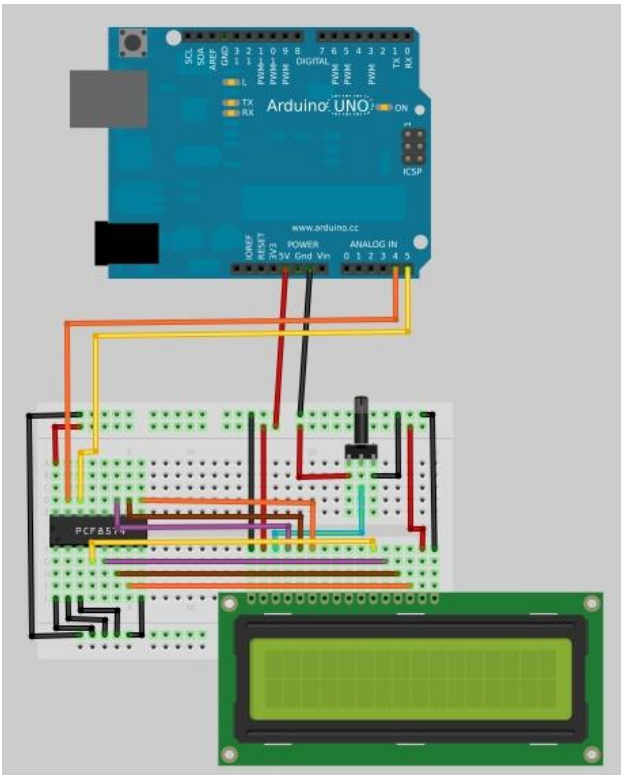
Neste tutorial vamos mostrar como utilizar o LCD 16x2 com PCF8574 e Arduino. Para quem não sabe o que é o PCF8574, temos um tutorial explicando e implementando o PCF8574 com Arduino. Clique aqui para o tutorial sobre PCF8574.

Antes de mais nada, baixe a biblioteca para a versão 1.0 da IDE do Arduino: biblioteca para a versão 1.0 da IDE do Arduino.

Caso você esteja utilizando uma versão mais antiga(002x) da IDE do Arduino, baixe esta biblioteca: biblioteca para IDE de versões anteriores.

Depois de baixado a bilbioteca, extraia para a pasta "libraries" localizada dentro da pasta da IDE do Arduino.

Agora, faça a seguinte ligação:



Você irá ligar o LCD no PCF8574 e um potenciometro para aumentar ou diminuir o contraste. E por fim o PCF8574 no Arduino como está mostrado na imagem.

Depois de feito a ligação, abra a IDE do Arduino e vá em File/Examples/LiquidCrystal_I2C e clique em "Hello World". Irá abrir a seguinte programação:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(32,16,2); // set the LCD address to 0x20 for a 16 chars and 2 line display
void setup()
{
lcd.init(); // initialize the lcd

// Print a message to the LCD.
lcd.backlight();
lcd.print("Hello, world!");
}
void loop()
{
}
```

Agora, selecione a versão da placa Arduino(UNO, Duemilanove, etc) e a porta em que a placa Arduino está conectado. E clique em UPLOAD.

Ao passar a programação, ajuste o contraste pelo potenciomêtro. Pronto! A frase "Hello, world!" irá aparecer no LCD.

E é isso, pessoal!! Até a próxima!! Se tiver dúvidas, poste aqui no blog! Para sugestões de tutoriais, clique aqui! Você pode ver outros tutoriais também, clicando aqui! E projetos abertos desenvolvidos pelos garagistas, clicando aqui!!

Referências:

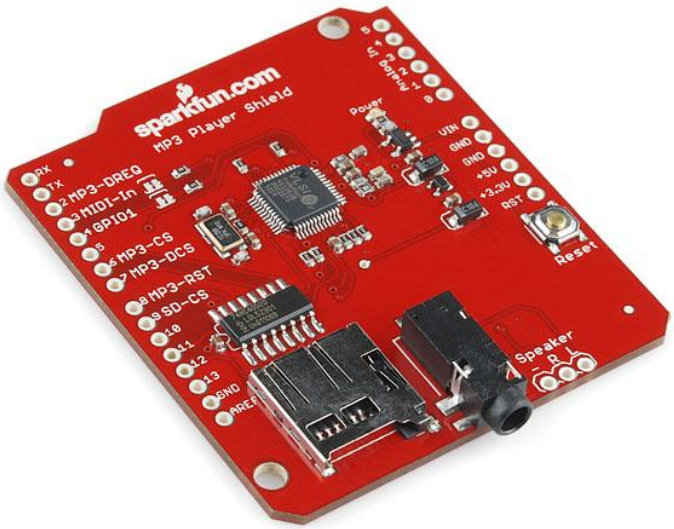
http://arduino.cc/playground/Code/LCDi2c

http://hmario.home.xs4all.nl/arduino/LiquidCrystal_I2C/

http://www.labdegaragem.org/loja/index.php/expansor-de-portas-i-o-i...

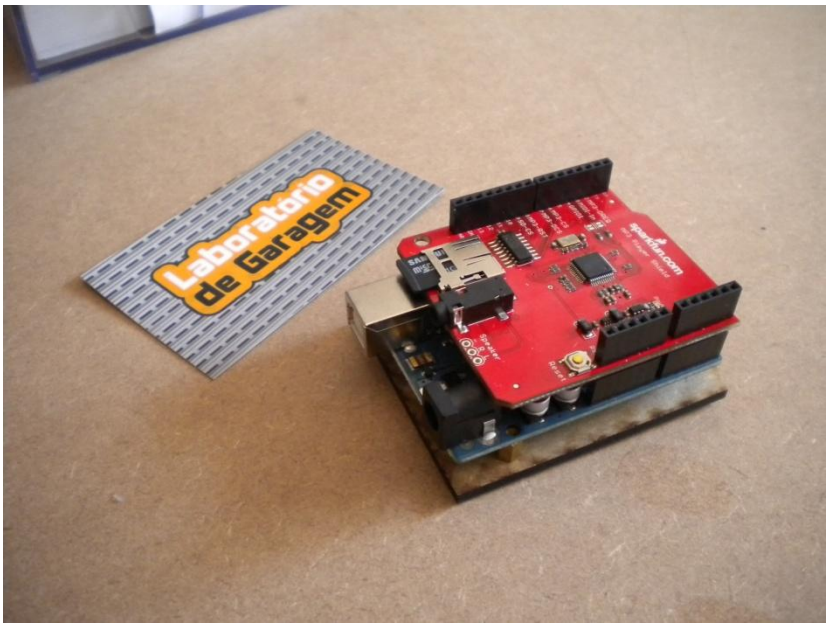
http://www.labdegaragem.org/loja/index.php/lcd-16x2.html

39. Tutorial: como utilizar o MP3 Player Shield com Arduino



Neste tutorial, vamos mostrar como utilizar o MP3 Player Shield com Arduino.

O MP3 Player Shield é um Shield que você acopla no Arduino e este vira um player de MP3. A figura abaixo mostra o Shield acoplado no Arduino.



Antes de mais nada, é necessário baixar a biblioteca SDfat, disponível aqui. Depois de baixar a biblioteca, extraia para a pasta "libraries" dentro da pasta da IDE do Arduino. Dentro da pasta do SDfat que você extraiu, abra o arquivo "Sd2PinMap.h" e localize o seguinte comentário: "// 168 and 328 Arduinos" para Arduino UNO e Duemilanove. Abaixo deste comentário tem uma linha escrito: "uint8_t const SS_PIN = 10;", mude o número 10 para 9, salve e feche o arquivo.

Com um cartão MicroSD, coloque uma música .mp3 e nomeie como track001.mp3. Coloque o cartão MicroSD no MP3 Player Shield.

Agora abra a IDE do Arduino e cole a programação exemplo disponibilizada aqui, vá para a linha destacada em vermelho, como mostrado abaixo, e verifique se é o número 10:

```
/*
4-28-2011
Spark Fun Electronics 2011
Nathan Seidle

This code is public domain but you buy me a beer if you use this and we meet someday
(Beerware license).

This example code plays a MP3 from the SD card called 'track001.mp3'. The theory is that you
can load a
microSD card up with a bunch of MP3s and then play a given 'track' depending on some sort of
input such
as which pin is pulled low.

It relies on the sdfatlib from Bill Greiman:
http://code.google.com/p/sdfatlib/
You will need to download and install his library. To compile, you MUST change Sd2PinMap.h
of the SDfatlib!
The default SS_PIN = 10;. You must change this line under the ATmega328/Arduino area of
code to
uint8_t const SS_PIN = 9;. This will cause the sdfatlib to use pin 9 as the 'chip select' for the
microSD card on pin 9 of the Arduino so that the layout of the shield works.

Attach the shield to an Arduino. Load code (after editing Sd2PinMap.h) then open the terminal
at 57600bps. This
example shows that it takes ~30ms to load up the VS1053 buffer. We can then do whatever we
want for ~100ms
before we need to return to filling the buffer (for another 30ms).

This code is heavily based on the example code I wrote to control the MP3 shield found here:
http://www.sparkfun.com/products/9736
This example code extends the previous example by reading the MP3 from an SD card and file
rather than from internal
memory of the ATmega. Because the current MP3 shield does not have a microSD socket, you
will need to add the microSD
shield to your Arduino stack.

The main gotcha from all of this is that you have to make sure your CS pins for each device on
an SPI bus is carefully
declared. For the SS pin (aka CS) on the SD FAT libaray, you need to correctly set it within
Sd2PinMap.h. The default
pin in Sd2PinMap.h is 10. If you're using the SparkFun microSD shield with the SparkFun MP3
shield, the SD CS pin
is pin 9.
```

Four pins are needed to control the VS1503:
DREQ
CS
DCS
Reset (optional but good to have access to)
Plus the SPI bus

Only the SPI bus pins and another CS pin are needed to control the microSD card.

What surprised me is the fact that with a normal MP3 we can do other things for up to 100ms while the MP3 IC crunches
through it's fairly large buffer of 2048 bytes. As long as you keep your sensor checks or serial reporting to under
100ms and leave ~30ms to then replenish the MP3 buffer, you can do quite a lot while the MP3 is playing glitch free.

*/

```
#include <SPI.h>
//Add the SdFat Libraries
#include <SdFat.h>
#include <SdFatUtil.h>
//Create the variables to be used by SdFat Library
Sd2Card card;
SdVolume volume;
SdFile root;
SdFile track;
//This is the name of the file on the microSD card you would like to play
//Stick with normal 8.3 nomeclature. All lower-case works well.
//Note: you must name the tracks on the SD card with 001, 002, 003, etc.
//For example, the code is expecting to play 'track002.mp3', not track2.mp3.
char trackName[] = "track001.mp3";
int trackNumber = 1;
char errorMsg[100]; //This is a generic array used for sprintf of error messages
#define TRUE 0
#define FALSE 1
//MP3 Player Shield pin mapping. See the schematic
#define MP3_XCS 6 //Control Chip Select Pin (for accessing SPI Control/Status registers)
#define MP3_XDCS 7 //Data Chip Select / BSYNC Pin
#define MP3_DREQ 2 //Data Request Pin: Player asks for more data
#define MP3_RESET 8 //Reset is active low
//Remember you have to edit the Sd2PinMap.h of the sdfatlib library to correct control the SD ca
rd.
//VS10xx SCI Registers
#define SCI_MODE 0x00
#define SCI_STATUS 0x01
#define SCI_BASS 0x02
#define SCI_CLOCKF 0x03
#define SCI_DECODE_TIME 0x04
#define SCI_AUDATA 0x05
#define SCI_WRAM 0x06
#define SCI_WRAMADDR 0x07
#define SCI_HDAT0 0x08
#define SCI_HDAT1 0x09
#define SCI_AIADDR 0x0A
#define SCI_VOL 0x0B
#define SCI_AICTRL0 0x0C
#define SCI_AICTRL1 0x0D
#define SCI_AICTRL2 0x0E
#define SCI_AICTRL3 0x0F
void setup() {
```

```
pinMode(MP3_DREQ, INPUT);
pinMode(MP3_XCS, OUTPUT);
pinMode(MP3_XDCS, OUTPUT);
pinMode(MP3_RESET, OUTPUT);
digitalWrite(MP3_XCS, HIGH); //Deselect Control
digitalWrite(MP3_XDCS, HIGH); //Deselect Data
digitalWrite(MP3_RESET, LOW); //Put VS1053 into hardware reset
Serial.begin(57600); //Use serial for debugging
Serial.println("MP3 Testing");
//Setup SD card interface
pinMode(10, OUTPUT); //Pin 10 must be set as an output for the SD communication to work.
if (!card.init(SPI_FULL_SPEED)) Serial.println("Error: Card init"); //Initialize the SD card and
configure the I/O pins.
if (!volume.init(&card)) Serial.println("Error: Volume ini"); //Initialize a volume on the SD card.
if (!root.openRoot(&volume)) Serial.println("Error: Opening root"); //Open the root directory in
the volume.
//We have no need to setup SPI for VS1053 because this has already been done by the SDfatli
b
//From page 12 of datasheet, max SCI reads are CLKI/7. Input clock is 12.288MHz.
//Internal clock multiplier is 1.0x after power up.
//Therefore, max SPI speed is 1.75MHz. We will use 1MHz to be safe.
SPI.setClockDivider(SPI_CLOCK_DIV16); //Set SPI bus speed to 1MHz (16MHz / 16 = 1MHz)
SPI.transfer(0xFF); //Throw a dummy byte at the bus
//Initialize VS1053 chip
delay(10);
digitalWrite(MP3_RESET, HIGH); //Bring up VS1053
//delay(10); //We don't need this delay because any register changes will check for a high DRE
Q
//Mp3SetVolume(20, 20); //Set initial volume (20 = -10dB) LOUD
Mp3SetVolume(40, 40); //Set initial volume (20 = -10dB) Manageable
//Mp3SetVolume(80, 80); //Set initial volume (20 = -10dB) More quiet
//Let's check the status of the VS1053
int MP3Mode = Mp3ReadRegister(SCI_MODE);
int MP3Status = Mp3ReadRegister(SCI_STATUS);
int MP3Clock = Mp3ReadRegister(SCI_CLOCKF);
Serial.print("SCI_Mode (0x4800) = 0x");
Serial.println(MP3Mode, HEX);
Serial.print("SCI_Status (0x48) = 0x");
Serial.println(MP3Status, HEX);
int vsVersion = (MP3Status >> 4) & 0x000F; //Mask out only the four version bits
Serial.print("VS Version (VS1053 is 4) = ");
Serial.println(vsVersion, DEC); //The 1053B should respond with 4. VS1001 = 0, VS1011 = 1,
VS1002 = 2, VS1003 = 3
Serial.print("SCI_ClockF = 0x");
Serial.println(MP3Clock, HEX);
//Now that we have the VS1053 up and running, increase the internal clock multiplier and up our
 SPI rate
Mp3WriteRegister(SCI_CLOCKF, 0x60, 0x00); //Set multiplier to 3.0x
//From page 12 of datasheet, max SCI reads are CLKI/7. Input clock is 12.288MHz.
//Internal clock multiplier is now 3x.
//Therefore, max SPI speed is 5MHz. 4MHz will be safe.
SPI.setClockDivider(SPI_CLOCK_DIV4); //Set SPI bus speed to 4MHz (16MHz / 4 = 4MHz)
MP3Clock = Mp3ReadRegister(SCI_CLOCKF);
Serial.print("SCI_ClockF = 0x");
Serial.println(MP3Clock, HEX);
//MP3 IC setup complete
}
void loop(){
//Let's play a track of a given number
```

```
sprintf(trackName, "track%03d.mp3", trackNumber); //Splice the new file number into this file name
playMP3(trackName); //Go play trackXXX.mp3
//Once we are done playing or have exited the playback for some reason, decide what track to play next
trackNumber++; //When we loop, advance to next track!
if(trackNumber > 100) {
Serial.println("Whoa there cowboy!"); //Soft limit. We shouldn't be trying to open past track 100.
while(1);
}
}
//PlayMP3 pulls 32 byte chunks from the SD card and throws them at the VS1053
//We monitor the DREQ (data request pin). If it goes low then we determine if
//we need new data or not. If yes, pull new from SD card. Then throw the data
//at the VS1053 until it is full.
void playMP3(char* fileName) {
if (!track.open(&root, fileName, O_READ)) { //Open the file in read mode.
sprintf(errorMsg, "Failed to open %s", fileName);
Serial.println(errorMsg);
return;
}
Serial.println("Track open");
uint8_t mp3DataBuffer[32]; //Buffer of 32 bytes. VS1053 can take 32 bytes at a go.
//track.read(mp3DataBuffer, sizeof(mp3DataBuffer)); //Read the first 32 bytes of the song
int need_data = TRUE;
long replenish_time = millis();
Serial.println("Start MP3 decoding");
while(1) {
while(!digitalRead(MP3_DREQ)) {
//DREQ is low while the receive buffer is full
//You can do something else here, the buffer of the MP3 is full and happy.
//Maybe set the volume or test to see how much we can delay before we hear audible glitches
//If the MP3 IC is happy, but we need to read new data from the SD, now is a great time to do so
if(need_data == TRUE) {
if(!track.read(mp3DataBuffer, sizeof(mp3DataBuffer))) { //Try reading 32 new bytes of the song
//Oh no! There is no data left to read!
//Time to exit
break;
}
need_data = FALSE;
}
//Serial.println("."); //Print a character to show we are doing nothing
//This is here to show how much time is spent transferring new bytes to the VS1053 buffer. Relies on replenish_time below.
Serial.print("Time to replenish buffer: ");
Serial.print(millis() - replenish_time, DEC);
Serial.print("ms");
//Test to see just how much we can do before the audio starts to glitch
long start_time = millis();
//delay(150); //Do NOTHING - audible glitches
//delay(135); //Do NOTHING - audible glitches
//delay(120); //Do NOTHING - barely audible glitches
delay(100); //Do NOTHING - sounds fine
Serial.print(" Idle time: ");
Serial.print(millis() - start_time, DEC);
Serial.println("ms");
//Look at that! We can actually do quite a lot without the audio glitching
//Now that we've completely emptied the VS1053 buffer (2048 bytes) let's see how much
//time the VS1053 keeps the DREQ line high, indicating it needs to be fed
```

```
replenish_time = millis();
}

if(need_data == TRUE){ //This is here in case we haven't had any free time to load new data
if(!track.read(mp3DataBuffer, sizeof(mp3DataBuffer))) { //Go out to SD card and try reading 32 new bytes of the song
//Oh no! There is no data left to read!
//Time to exit
break;
}
need_data = FALSE;
}
//Once DREQ is released (high) we now feed 32 bytes of data to the VS1053 from our SD read buffer
digitalWrite(MP3_XDCS, LOW); //Select Data
for(int y = 0 ; y < sizeof(mp3DataBuffer) ; y++) {
SPI.transfer(mp3DataBuffer[y]); // Send SPI byte
}
digitalWrite(MP3_XDCS, HIGH); //Deselect Data
need_data = TRUE; //We've just dumped 32 bytes into VS1053 so our SD read buffer is empty. Set flag so we go get more data
}
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating transfer is complete
digitalWrite(MP3_XDCS, HIGH); //Deselect Data

track.close(); //Close out this track
sprintf(errorMsg, "Track %s done!", fileName);
Serial.println(errorMsg);
}
//Write to VS10xx register
//SCI: Data transfers are always 16bit. When a new SCI operation comes in
//DREQ goes low. We then have to wait for DREQ to go high again.
//XCS should be low for the full duration of operation.
void Mp3WriteRegister(unsigned char addressbyte, unsigned char highbyte, unsigned char lowbyte){
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating IC is available
digitalWrite(MP3_XCS, LOW); //Select control
//SCI consists of instruction byte, address byte, and 16-bit data word.
SPI.transfer(0x02); //Write instruction
SPI.transfer(addressbyte);
SPI.transfer(highbyte);
SPI.transfer(lowbyte);
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating command is complete
digitalWrite(MP3_XCS, HIGH); //Deselect Control
}
//Read the 16-bit value of a VS10xx register
unsigned int Mp3ReadRegister (unsigned char addressbyte){
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating IC is available
digitalWrite(MP3_XCS, LOW); //Select control
//SCI consists of instruction byte, address byte, and 16-bit data word.
SPI.transfer(0x03); //Read instruction
SPI.transfer(addressbyte);
char response1 = SPI.transfer(0xFF); //Read the first byte
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating command is complete
char response2 = SPI.transfer(0xFF); //Read the second byte
while(!digitalRead(MP3_DREQ)) ; //Wait for DREQ to go high indicating command is complete
digitalWrite(MP3_XCS, HIGH); //Deselect Control
int resultvalue = response1 8;
resultvalue |= response2;
return resultvalue;
```

```
}
//Set VS10xx Volume Register
void Mp3SetVolume(unsigned char leftchannel, unsigned char rightchannel){
Mp3WriteRegister(SCI_VOL, leftchannel, rightchannel);
}
```

Selecione a versão da sua placa Arduino (UNO, Duemilanove, etc) e selecione a Porta a qual o Arduino está conectado (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD. Ao terminar, conecte o MP3 Player Shield no Arduino. O áudio sairá pelo conector PS/2, então conecte as caixinhas de som e pronto! Caso tenha dúvidas, poste aqui mesmo neste blog!

Caso tenha sugestões para tutoriais, poste aqui! Para ver outros tutoriais disponíveis, clique aqui, e para projetos abertos disponibilizados por outros garagistas ou pela equipe LdG, clique aqui! Até a próxima!!

Referências:

http://www.labdegaragem.com.br/wiki/index.php?title=Tocando_Mp3_com...

http://www.labdegaragem.org/loja/index.php/31-shields/mp3-player-sh...

http://www.sparkfun.com/products/10628

http://www.sparkfun.com/tutorials/295

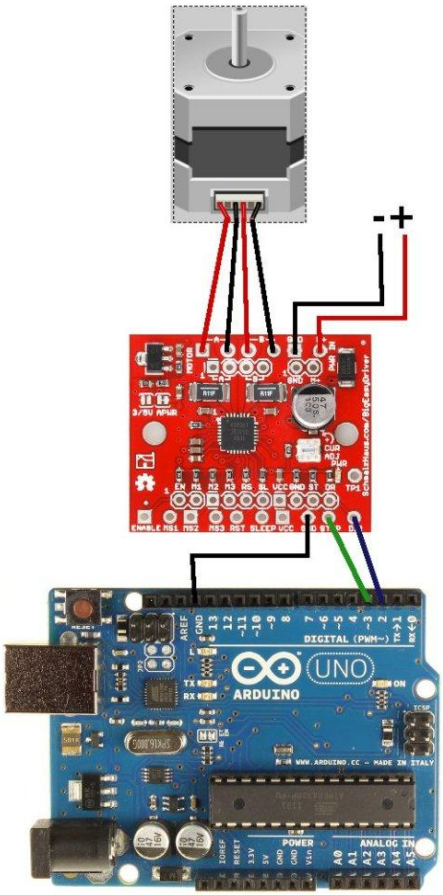
40. Tutorial: como utilizar o Big Easy Driver com um motor de passo e Arduino

Neste tutorial vamos mostrar como controlar um motor de passo com Big Easy Driver e Arduino. Para controlar um motor de passo é necessário um driver ou ponte-H, o Easy Driver e o Big Easy Driver são drivers de controle para motor de passo.

O Big Easy Driver, como o nome já diz, é maior e pode alimentar motores de passo com tensão de até 35V e 2A de corrente, enquanto que o Easy Driver pode alimentar com 30V e 750mA. Resumindo, o Big Easy Driver é um upgrade do Easy Driver. Caso queiram saber mais sobre o Easy Driver, temos um tutorial sobre ele clicando aqui.

Apesar de parecidos, o Big Easy Driver necessita de um tempo menor entre pulsos, isto será melhor demonstrado na programação.

Para controlar o motor de passo, precisamos fazer a seguinte ligação:



Onde o tamanho do passo (step) será o pino digital 3 do Arduino e a direção (dir) no pino digital 2 do Arduino. As ligações "+" e "-" devem ser ligadas a uma fonte externa de até 35V. Olhe na documentação do seu motor de passo para saber qual a corrente máxima e a tensão máxima do motor para não queimá-lo.

Assim como o Easy Driver, o Big Easy Driver tem um ENABLE que pode ser conectado a um pino digital para ligar e desligar o Big Easy Driver e evitar que este fique consumindo corrente o tempo todo. Para desligar o Big Easy Driver é necessário setá-lo para HIGH (alto) e para ligar é simplesmente o contrário.

Como apenas estamos demonstrando seu funcionamento, não foi utilizado o ENABLE. Para maiores informações você pode ver o datasheet do Big Easy Driver e o datasheet do A4983.

Agora vamos para a programação:

```
int stepPin = 3;
int dirPin=2;
// Variables will change:
int dirState = LOW;
int stepState =LOW;
long previousMicros = 0;

long interval = 1000;
void setup() {
pinMode(dirPin, OUTPUT);
pinMode(stepPin, OUTPUT);
}
void loop()
{

unsigned long currentMicros = micros();

if(currentMicros - previousMicros > interval) {

previousMicros = currentMicros;

if (stepState == LOW)
stepState = HIGH;
else
stepState = LOW;

digitalWrite(stepPin, stepState);
}
}
```

Como vocês podem ver na programação, o tamanho do passo (step) é dado por microssegundos e não milissegundos como no Easy Driver. Ao passar o tempo determinado, a programação muda de estado do pino digital 3 do Arduino. Se estiver em baixo (LOW) muda pra alto (HIGH) e vice-versa.

E é isso! Esperamos que tenham gostado! Se tiverem dúvidas, postem aqui neste blog. Para sugerir tutoriais, vocês podem sugerir aqui! Para ver outros tutoriais e projetos abertos, cliquem aqui e aqui respectivamente!

Referências:

http://www.labdegaragem.org/loja/index.php/36-motores-afins/big-eas...

http://www.sparkfun.com/products/10735

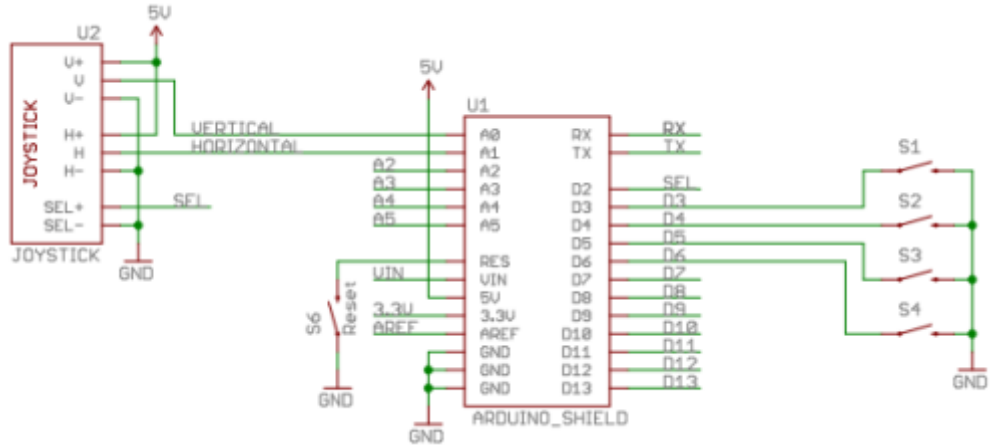
http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Robotics/4983.pdf

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Robotics/BigEasyDriv...

41. Tutorial: como utilizar o Joystick Shield com Arduino

O Joystick Shield é um shield para conectar em cima do Arduino. Este shield contém um joystick polegar de dois eixos, 5 botões de mudança momentânea (4 + 1 botão de seleção) e um botão de mudança momentânea para o reset.

Abaixo podemos ver o esquemático do Shield: (Caso queira ver mais detalhes, clique aqui)



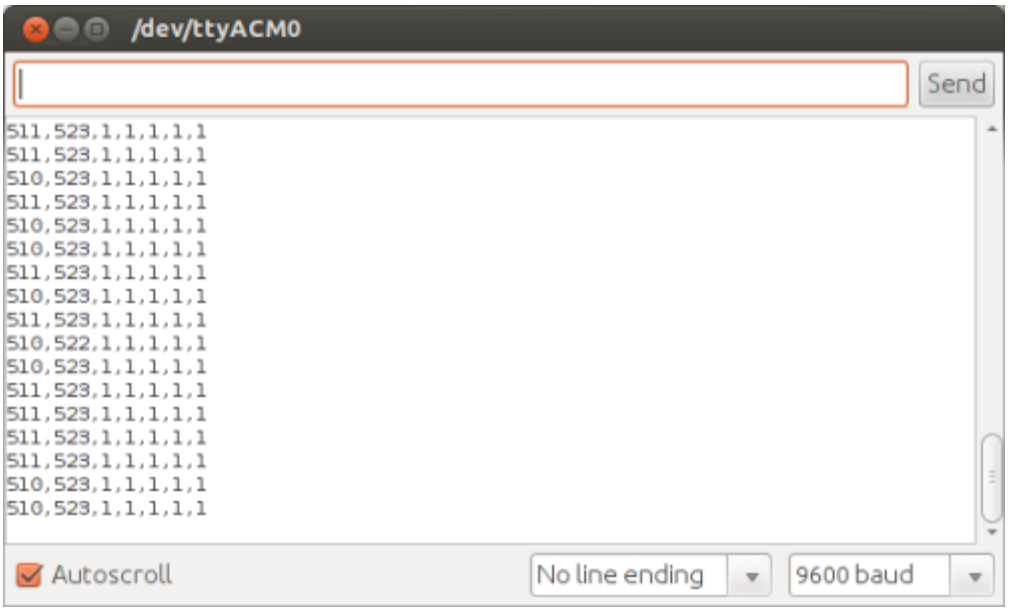
A partir do esquemático, podemos ver sua pinagem referente a conexão com Arduino. Agora que sabemos a pinagem, podemos ir para a programação do Arduino:

Abaixo está a programação exemplo modificada tirada deste link:

```
//Create variables for each button on the Joystick Shield to assign the pin numbers
```

```
char button0=3, button1=4, button2=5, button3=6;
char sel=2;
void setup(void)
{
pinMode(sel, INPUT); //Set the Joystick 'Select'button as an input
digitalWrite(sel, HIGH); //Enable the pull-up resistor on the select button

pinMode(button0, INPUT); //Set the Joystick button 0 as an input
digitalWrite(button0, HIGH); //Enable the pull-up resistor on button 0
pinMode(button1, INPUT); //Set the Joystick button 1 as an input
digitalWrite(button1, HIGH); //Enable the pull-up resistor on button 1

pinMode(button2, INPUT); //Set the Joystick button 2 as an input
digitalWrite(button2, HIGH); //Enable the pull-up resistor on button 2
pinMode(button3, INPUT); //Set the Joystick button 3 as an input
digitalWrite(button3, HIGH); //Enable the pull-up resistor on button 3

Serial.begin(9600); //Turn on the Serial Port at 9600 bps
}
void loop(void)
{
Serial.print(analogRead(0)); //Read the position of the joysticks X axis and print it on the serial
port.
Serial.print(",");
Serial.print(analogRead(1)); //Read the position of the joysticks Y axis and print it on the serial
port.
Serial.print(",");
Serial.print(digitalRead(sel)); //Read the value of the select button and print it on the serial port.
Serial.print(",");
Serial.print(digitalRead(button0)); //Read the value of the button 0 and print it on the serial port.
Serial.print(",");
Serial.print(digitalRead(button1)); //Read the value of the button 1 and print it on the serial port.
Serial.print(",");
Serial.print(digitalRead(button2)); //Read the value of the button 2 and print it on the serial port.
Serial.print(",");
Serial.println(digitalRead(button3)); //Read the value of the button 3 and print it on the serial
port.

//Wait for 100 ms, then go back to the beginning of 'loop' and repeat.
delay(100);
}
```

Abra a IDE do Arduino e cole a programação acima. Conecte o Joystick Shield no Arduino e depois o Arduino no PC. Selecione a versão da sua placa Arduino (UNO, Duemilanove, etc) e depois a porta (COMx, ttyUSBx, ttyACMx, etc)a qual o Arduino está conectado. Faça o UPLOAD e abra o Serial Monitor. Selecione "9600 baud" e "No line ending", você verá vários números como mostra a figura abaixo:



Ao mexer o joystick polegar, você verá os primeiros números variarem (510 e 523). Os números posteriores são o botão de seleção (botão do polegar) e os 4 botões restantes do Joystick Shield. Ao apertar um desses botões, você verá um valor "1" ir para "0". Você pode experimentar apertar todos os botões ao mesmo tempo e todos irão para "0".

Pronto! Agora você pode fazer seu projeto utilizando o Joystick Shield como protótipo! Se tiver dúvidas, poste aqui mesmo neste blog! Caso tenha sugestões para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente!

Referências:

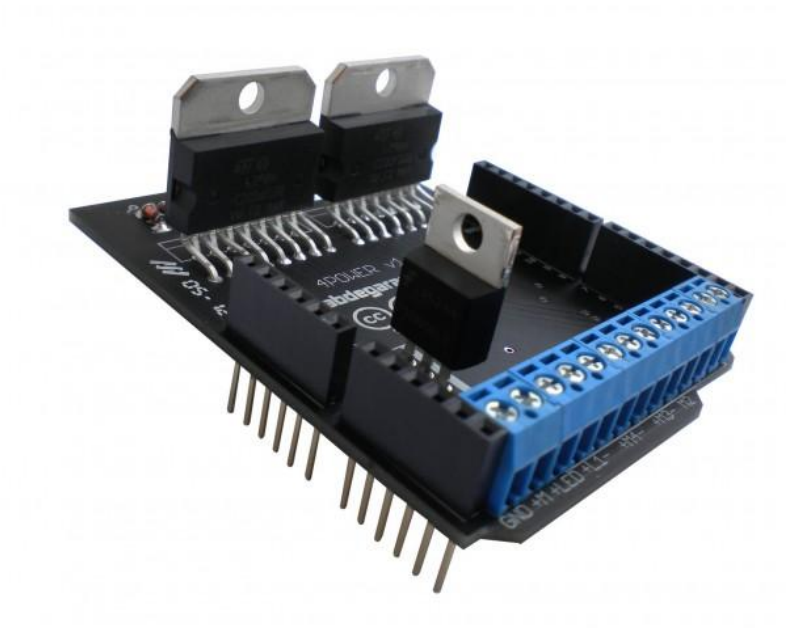
http://www.labdegaragem.org/loja/index.php/31-shields/joystick-shie...

http://www.sparkfun.com/products/9760

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Dev/Arduino/Shields/...

http://www.sparkfun.com/Code/Joystick_Shield_Example.zip

42. Tutorial: Como utilizar o Shield 4 Power com Arduino

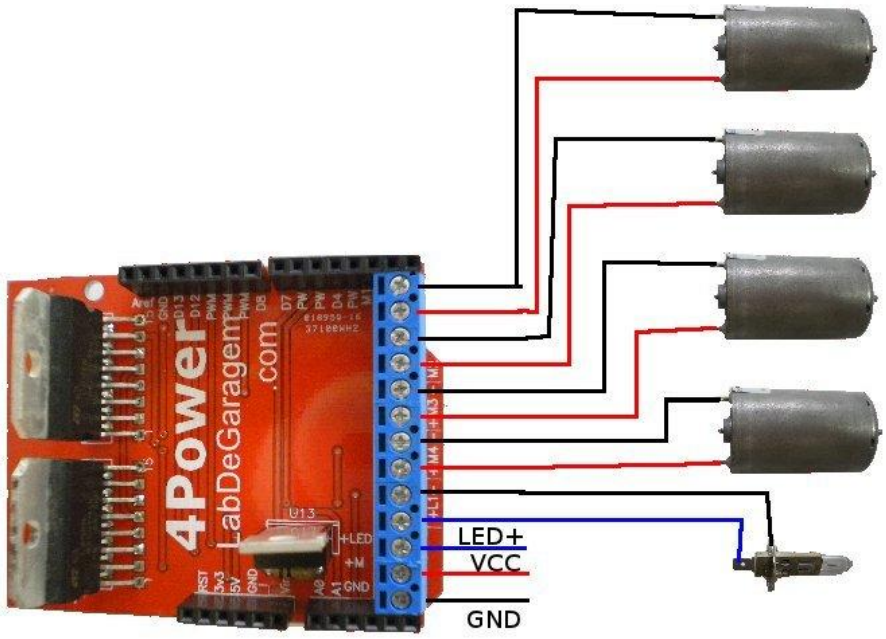


O Shield 4 Power é produto incubado desenvolvido pelo garagista Bruno Dias Rosa. O 4 Power é um shield para Arduino que pode controlar até 4 motores de corrente contínua de no máximo 46V e uma carga resistiva extra a sua escolha. Para usá-lo basta conectar uma fonte de alimentação de até 46V (até 4A), ligando nos terminais M+ e GND. A placa possui indicação de positivo e negativo para cada motor, que podem puxar até 2A, na tensão máxima de alimentação da fonte (M+), mas que pode variar por PWM (controle disponível na biblioteca).

Neste tutorial, vamos mostrar como utilizá-lo com a biblioteca e a programação exemplo.

Baixe os arquivos da biblioteca clicando aqui. Agora descompacte, copie e cole a pasta "S4power" dentro da pasta "libraries" localizada dentro da pasta da IDE do Arduino.

Conecte o shield no Arduino e você pode fazer a seguinte ligação:



o arduino no PC. Abra a IDE do Arduino e passe a seguinte programação exemplo:

```
#include <S4power.h>
S4Power s4power;
void setup()
{
s4power.Config();
}
void loop()
{
for ( int count = -20; count < 20; count++ )
{
s4power.M1.speed = count;
s4power.M2.speed = count;
s4power.M3.speed = count;
s4power.M4.speed = count;

s4power.M1.Update();
s4power.M2.Update();
s4power.M3.Update();
s4power.M4.Update();

s4power.light.intensity = count;
s4power.light.Update();

delay (100);
}
}
```

Agora alimente o Shield com uma fonte externa no M+ e pronto! Você verá que os motores DC irão girar de um lado e depois para o outro.

Esperamos que tenha gostado! Se tiver dúvidas, poste aqui mesmo neste blog! Se tiver sugestões para tutoriais, poste aqui! Caso queira ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui!

Referências:

http://www.labdegaragem.org/loja/index.php/31-shields/4-power.html

http://www.labdegaragem.com.br/wiki/index.php?title=Shield_4Power#P...

43. Tutorial: Array de arrays no Arduino

Neste tutorial, mostraremos como declarar array de arrays. Às vezes você precisa escolher dinamicamente qual array usar, por isso é melhor usar array de arrays.

Você pode ter array de qualquer tipo. Se for um array char, então os valores serão char, array int, valores em int, assim por diante. Se você tiver um char* array, é um array de ponteiros com variáveis do tipo char. Por exemplo, cada valor do "int valor[3]" pode ser acessado por "valor[0]" ou "*valor", e próximo item pode ser acessado por "valor[1]" ou "(*valor)+1".

Portanto, podemos armazenar endereços de um array em outro array.

Antes de mais nada, precisamos saber o tamanho de memória disponível na RAM. O Atmega328P tem 2Kbytes de memória RAM. Portanto só podemos declarar 2Kbytes de variáveis. Como alternativa, você pode armazenar utilizando a memória FLASH usando o PROGMEM.

Agora, vamos a um exemplo:

```
char line0[3] = { 'a', 'b', 'c' };
char line1[3] = { 'd', 'e', 'f' };
char line2[3] = { 'g', 'h', 'i' };
char* lines[4];
char* column;

void setup() {
lines[0] = line0;
lines[1] = line1;
lines[2] = line2;
Serial.begin(9600);
}

void loop() {
int i,j;
for(i=0;i<3;i++)
{
column = line[i]; // get the array address
for(j=0;j<3;j++)
{
Serial.print(*column++); // print the element, then increment the index
}
Serial.println(); // line break
}
while(1); // infinite loop to stop the program
}
```

Na programação, primeiro declaramos 3 arrays com 3 elementos: line0, line1, line2. Na função setup(), pegamos os endereços e armazenamos no array lines. Como o array é um ponteiro, nós não precisamos o símbolo "&" que refere-se ao endereço.

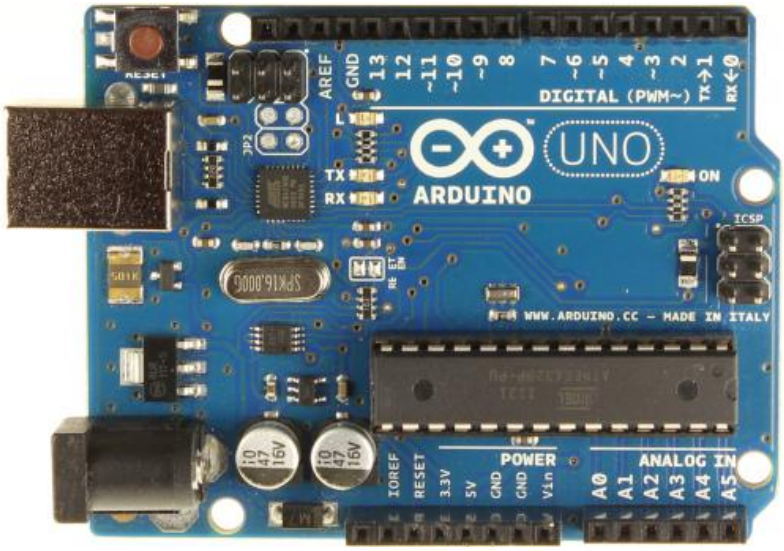
Na função loop(), podemos ver como acessá-lo. A variável column endereça ao elemento atual no array desejado. Copiamos o endereço do array, o qual é o mesmo endereço do primeiro elemento e então incrementamos para o próximo elemento.

O while(1); é para que o microcontrolador pare e não faça tudo de novo.

E é isso! Faça o UPLOAD no seu Arduino e abra a Serial Monitor para ver os endereços!

Esperamos que tenha gostado! Caso tenha dúvidas, poste aqui neste blog! Para sugestões de tutoriais, poste aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente. Até a próxima!

44. Tutorial: utilizando Strings no Arduino



Neste tutorial, vamos mostrar como utilizar e tratar Strings com Arduino.

Às vezes, precisamos tratar dados adquiridos, ou pela internet, ou pela porta serial, ou qualquer outra maneira.

Uma String é nada menos que uma matriz de 1 linha por várias colunas (ou vários caracteres). No Arduino, você pode declarar uma variável como String e não precisar ler caractere a caractere como em muitas linguagens de programação. Ficando assim mais fácil de trabalhar e programar.

Por exemplo, ao utilizar a Serial.read() o dado vem um de cada vez, isto é, caractere a caractere. Para ficar mais fácil de tratar esses dados, guardamos em uma matriz de array e depois colocamos em uma variável String. Agora é só comparar com uma outra string.

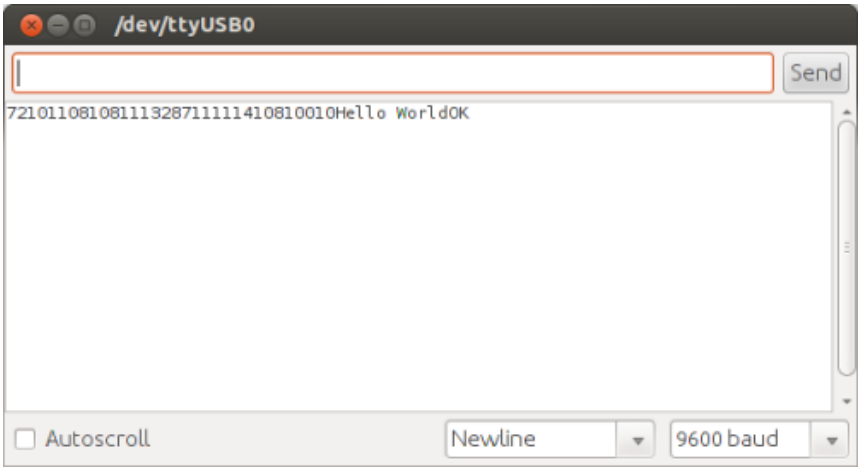
Cuidado que todos os caracteres da tabela ASCII são guardados dentro da string! Por exemplo o "enter" (new line - '\n'). Então é necessário colocar o nulo ('\0') no final de cada String.

Vamos usar a programação abaixo para exemplo:

```
String frase = "Hello World";
String str;
char c;
```

```
char matriz[20];
int x=0;
void setup()
{
Serial.begin(9600);
}
void loop()
{
if(Serial.available())
{
do{
c=Serial.read();
matriz[x]=c;
Serial.print(matriz[x],DEC);
x++;
delay(1);      //Delay para o Arduino não perder o dado da Serial
}while(c!='\n');
matriz[x-1]='\0';
Serial.print(matriz);
str=matriz;
if (str==frase)
{
Serial.println("OK");
}
else
{
Serial.println("Erro");
}
}
}
```

Na programação acima, declaramos uma variável String com "Hello World" e outra String sem nenhum conteúdo. Criamos uma matriz para pegar os caracteres que vem pela porta Serial. Se caso a String da porta Serial for igual a String "Hello World", o Arduino responderá um OK! Caso contrário o Arduino responderá "Erro".



No Serial Monitor, selecione "9600 baud" e "Newline".

Os números representam o código referente ao caractere ASCII. Para ver a tabela de ASCII, clique aqui!

E depois a String "Hello World" que foi digitada pela porta Serial e por fim a resposta de que a String digitada na porta Serial corresponde a String guardada.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas sobre o post, poste aqui mesmo neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

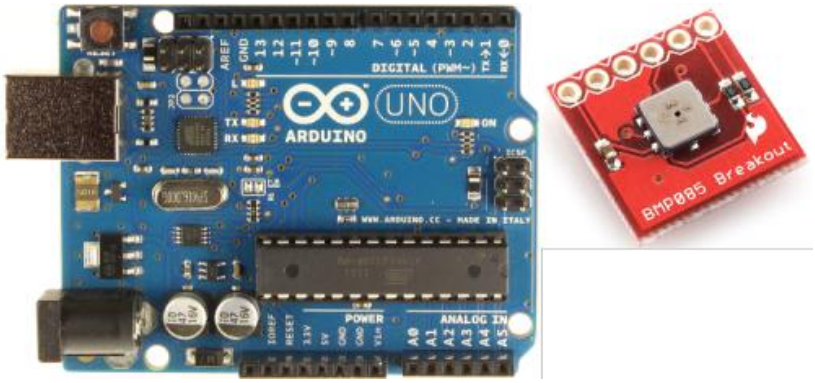
Referências:

http://arduino.cc/

http://arduino.cc/en/Reference/StringObject

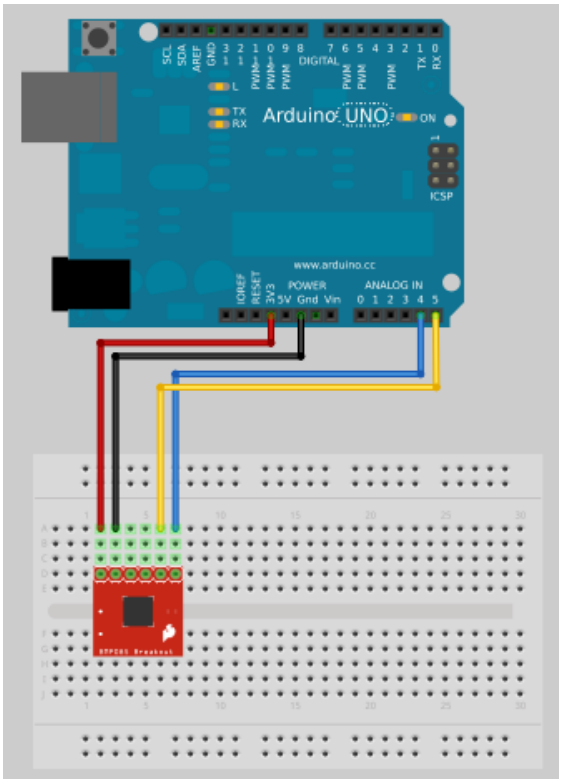
http://www.labdegaragem.com.br/wiki/index.php?title=Tabela_ASCII

45. Tutorial: como utilizar o Breakout de Sensor de pressão barométrica com Arduino



Neste tutorial vamos mostrar como utilizar o breakout sensor de pressão barométrica com Arduino. Este breakout serve para medir pressão barométrica, altitude e temperatura. A sua comunicação é por I2C (SDA, SCL) o que torna fácil sua implementação. Para maiores informações, clique aqui!

Primeiramente, faça a seguinte ligação:



Cuidado! O sensor é alimentado por 3,3V! Depois de ter feito as ligações da figura acima, vamos para a programação exemplo, a qual está disponibilizada aqui, para a IDE do Arduino versão 1.0:

```
#include <Wire.h>
#define BMP085_ADDRESS 0x77 // I2C address of BMP085
const unsigned char OSS = 0; // Oversampling Setting
// Calibration values
int ac1;
int ac2;
int ac3;
unsigned int ac4;
unsigned int ac5;
unsigned int ac6;
int b1;
int b2;
int mb;
int mc;
int md;
// b5 is calculated in bmp085GetTemperature(...), this variable is also used in bmp085GetPress
ure(...)
// so ...Temperature(...) must be called before ...Pressure(...).
long b5;
void setup(){
Serial.begin(9600);
Wire.begin();
bmp085Calibration();
}
void loop()
{
float temperature = bmp085GetTemperature(bmp085ReadUT()); //MUST be called first
float pressure = bmp085GetPressure(bmp085ReadUP());
float atm = pressure / 101325; // "standard atmosphere"
float altitude = calcAltitude(pressure); //Uncompensated caculation - in Meters
Serial.print("Temperature: ");
Serial.print(temperature, 2); //display 2 decimal places
Serial.println("deg C");
Serial.print("Pressure: ");
Serial.print(pressure, 0); //whole number only.
Serial.println(" Pa");
Serial.print("Standard Atmosphere: ");
Serial.println(atm, 4); //display 4 decimal places
Serial.print("Altitude: ");
Serial.print(altitude, 2); //display 2 decimal places
Serial.println(" M");
Serial.println();//line break
delay(1000); //wait a second and get values again.
}
// Stores all of the bmp085's calibration values into global variables
// Calibration values are required to calculate temp and pressure
// This function should be called at the beginning of the program
void bmp085Calibration()
{
ac1 = bmp085ReadInt(0xAA);
ac2 = bmp085ReadInt(0xAC);
ac3 = bmp085ReadInt(0xAE);
ac4 = bmp085ReadInt(0xB0);
ac5 = bmp085ReadInt(0xB2);
ac6 = bmp085ReadInt(0xB4);
```

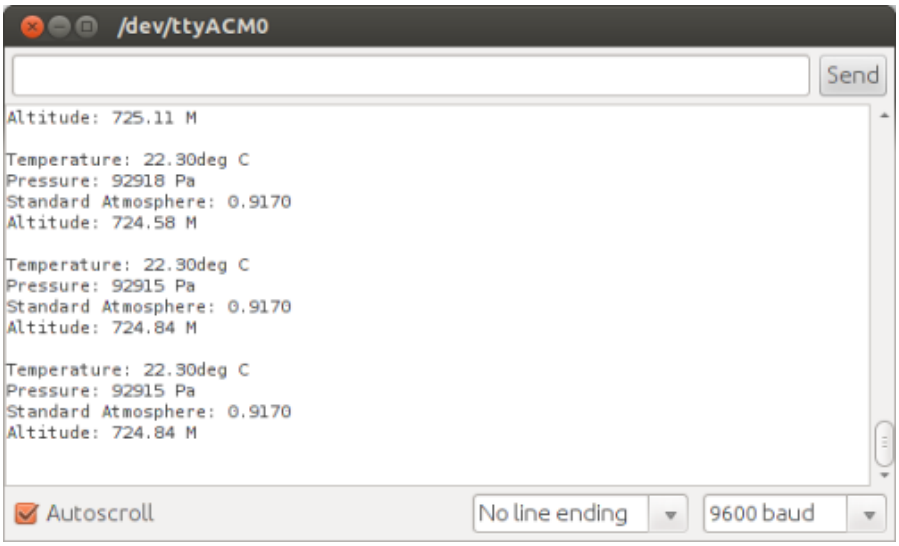
```
b1 = bmp085ReadInt(0xB6);
b2 = bmp085ReadInt(0xB8);
mb = bmp085ReadInt(0xBA);
mc = bmp085ReadInt(0xBC);
md = bmp085ReadInt(0xBE);
}
// Calculate temperature in deg C
float bmp085GetTemperature(unsigned int ut){
long x1, x2;
x1 = (((long)ut - (long)ac6)*(long)ac5) >> 15;
x2 = ((long)mc 11)/(x1 + md);
b5 = x1 + x2;
float temp = ((b5 + 8)>>4);
temp = temp /10;
return temp;
}
// Calculate pressure given up
// calibration values must be known
// b5 is also required so bmp085GetTemperature(...) must be called first.
// Value returned will be pressure in units of Pa.
long bmp085GetPressure(unsigned long up){
long x1, x2, x3, b3, b6, p;
unsigned long b4, b7;
b6 = b5 - 4000;
// Calculate B3
x1 = (b2 * (b6 * b6)>>12)>>11;
x2 = (ac2 * b6)>>11;
x3 = x1 + x2;
b3 = (((((long)ac1)*4 + x3)OSS) + 2)>>2;
// Calculate B4
x1 = (ac3 * b6)>>13;
x2 = (b1 * ((b6 * b6)>>12))>>16;
x3 = ((x1 + x2) + 2)>>2;
b4 = (ac4 * (unsigned long)(x3 + 32768))>>15;
b7 = ((unsigned long)(up - b3) * (50000>>OSS));
if (b7 < 0x80000000)
p = (b71)/b4;
else
p = (b7/b4)1;
x1 = (p>>8) * (p>>8);
x1 = (x1 * 3038)>>16;
x2 = (-7357 * p)>>16;
p += (x1 + x2 + 3791)>>4;
long temp = p;
return temp;
}
// Read 1 byte from the BMP085 at 'address'
char bmp085Read(unsigned char address)
{
unsigned char data;
Wire.beginTransmission(BMP085_ADDRESS);
Wire.write(address);
Wire.endTransmission();
Wire.requestFrom(BMP085_ADDRESS, 1);
while(!Wire.available())
;
return Wire.read();
}
// Read 2 bytes from the BMP085
// First byte will be from 'address'
```

```
// Second byte will be from 'address'+1
int bmp085ReadInt(unsigned char address)
{
unsigned char msb, lsb;
Wire.beginTransmission(BMP085_ADDRESS);
Wire.write(address);
Wire.endTransmission();
Wire.requestFrom(BMP085_ADDRESS, 2);
while(Wire.available()<2)
;
msb = Wire.read();
lsb = Wire.read();
return (int) msb8 | lsb;
}
// Read the uncompensated temperature value
unsigned int bmp085ReadUT(){
unsigned int ut;
// Write 0x2E into Register 0xF4
// This requests a temperature reading
Wire.beginTransmission(BMP085_ADDRESS);
Wire.write(0xF4);
Wire.write(0x2E);
Wire.endTransmission();
// Wait at least 4.5ms
delay(5);
// Read two bytes from registers 0xF6 and 0xF7
ut = bmp085ReadInt(0xF6);
return ut;
}
// Read the uncompensated pressure value
unsigned long bmp085ReadUP(){
unsigned char msb, lsb, xlsb;
unsigned long up = 0;
// Write 0x34+(OSS6) into register 0xF4
// Request a pressure reading w/ oversampling setting
Wire.beginTransmission(BMP085_ADDRESS);
Wire.write(0xF4);
Wire.write(0x34 + (OSS6));
Wire.endTransmission();
// Wait for conversion, delay time dependent on OSS
delay(2 + (3OSS));
// Read register 0xF6 (MSB), 0xF7 (LSB), and 0xF8 (XLSB)
msb = bmp085Read(0xF6);
lsb = bmp085Read(0xF7);
xlsb = bmp085Read(0xF8);
up = (((unsigned long) msb 16) | ((unsigned long) lsb 8) | (unsigned long) xlsb) >> (8-OSS);
return up;
}
void writeRegister(int deviceAddress, byte address, byte val) {
Wire.beginTransmission(deviceAddress); // start transmission to device
Wire.write(address); // send register address
Wire.write(val); // send value to write
Wire.endTransmission(); // end transmission
}
int readRegister(int deviceAddress, byte address){
int v;
Wire.beginTransmission(deviceAddress);
Wire.write(address); // register to read
Wire.endTransmission();
Wire.requestFrom(deviceAddress, 1); // read a byte
```

```
while(!Wire.available()) {
// waiting
}
v = Wire.read();
return v;
}
float calcAltitude(float pressure){
float A = pressure/101325;
float B = 1/5.25588;
float C = pow(A,B);
C = 1 - C;
C = C /0.0000225577;
return C;
}
```

A comunicação com o Arduino é feita por I2C (SDA, SCL). A calibração está dentro da programação acima, dentro da função bmp085Calibration(). Esta função precisa ser colocada no começo do programa, caso contrário o sensor barométrico não funcionará adequadamente, pois sua precisão depende da calibração. A função bmp085GetPressure(unsigned long up) obtém a pressão do tempo atual em Pascal. A função bmp085GetTemperature(unsigned int ut) obtém a temperatura do tempo atual em graus Celsius. E a função calcAltitude(float pressure) calcula a altitude do local a partir da pressão barométrica adquirida.

Agora, conecte seu Arduino e abra a IDE do Arduino versão 1.0, cole a programação acima. Selecione a versão da placa Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc). Clique em "UPLOAD". Agora abra a "Serial Monitor" e selecione o baud para "9600" e "No line ending". Você verá as informações como a figura abaixo:



Sabendo que a altitude de São Paulo é de 760 metros em média e a altitude é calculada a partir da pressão barométrica, podemos concluir que a precisão é muito boa, já que a altitude mostrada é de 724,84 metros.

E é isso! Esperamos que tenha gostado! Caso tiver duvídas, poste aqui mesmo neste blog! Para sugestões de tutoriais,clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui eaqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/34-sensores/breakout-de-...

http://www.sparkfun.com/products/9694

http://bildr.org/2011/06/bmp085-arduino/

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Pressure/BST...

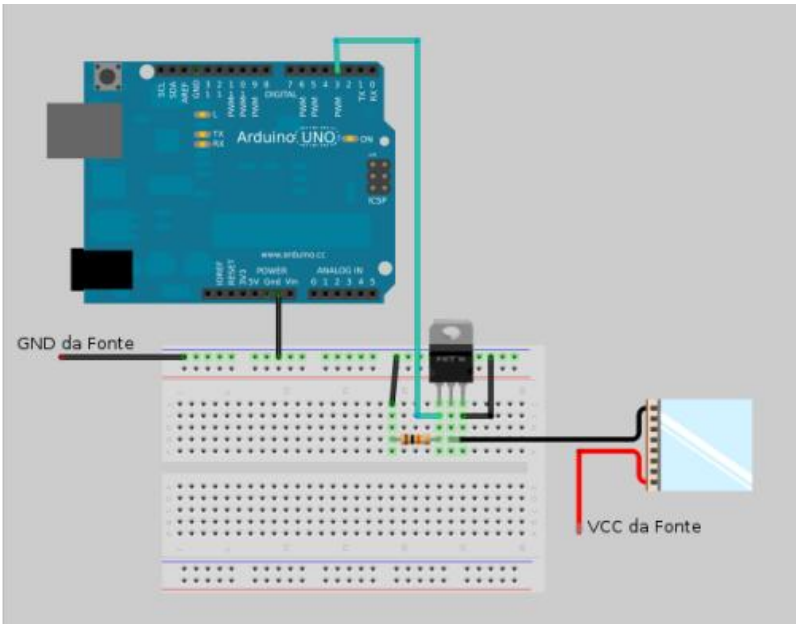
http://www.sparkfun.com/tutorials/253

46. Tutorial: Como utilizar uma pastilha Peltier com Arduino

Neste tutorial vamos mostrar como utilizar uma pastilha Peltier com Arduino. A pastilha Peltier é uma pastilha termo-elétrica, isto é, se aplicar uma dada corrente, a pastilha esfriará de um lado e aquecerá do outro. Com esta pastilha você pode utilizar para aquecer ou esfriar um determinado lugar ou objeto. A pastilha Peltier pode ser alimentada com até 15,4V e 7A.

Para utilizar a pastilha Peltier com Arduino é necessário utilizar um Mosfet canal N e um resistor de 10K Ohm.

A figura abaixo mostra a ligação a ser feita:



Para controlar a corrente é necessário utilizar uma porta digital do Arduino como PWM.

Podemos ver na placa Arduino que as portas digitais 3, 5, 6, 9,10 e 11 são portas para PWM.

Vamos escolher a porta digital 3 do Arduino como PWM.

A programação para utilizar a pastilha Peltier com Arduino está mostrado abaixo:

```
int peltier = 3; //The N-Channel MOSFET is on digital pin 3
int power = 0; //Power level fro 0 to 99%
int peltier_level = map(power, 0, 99, 0, 255); //This is a value from 0 to 255 that actually controls the MOSFET
void setup(){
Serial.begin(9600);
//pinMode(peltier, OUTPUT);
}
void loop(){
char option;
if(Serial.available() > 0)
{
option = Serial.read();
if(option == 'a')
power += 5;
else if(option == 'z')
power -= 5;
if(power > 99) power = 99;
if(power < 0) power = 0;
peltier_level = map(power, 0, 99, 0, 255);
}
Serial.print("Power=");
Serial.print(power);
Serial.print(" PLevel=");
Serial.println(peltier_level);
analogWrite(peltier, peltier_level); //Write this new value out to the port
}
```

Abra a IDE do Arduino e cole a programação acima. Depois conecte seu Arduino na porta USB do PC e selecione a versão do seu Arduino (UNO, Duemilanove, etc) em Tools/Board e depois selecione a porta USB em que seu Arduino está conectado (COMx, ttyUSBx, ttyACMx, etc) em Tools/Serial Port. E clique em UPLOAD. Agora abra o Serial Monitor e irá mostrar o seguinte:

No campo de escrita do Serial Monitor, digite "a" e depois aperte a tecla ENTER. Irá incrementar o POWER de 5 em 5. Para decrementar, digite "z" e aperte a tecla ENTER.

Ao incrementar, irá aumentar a corrente fornecida para a pastilha Peltier. Ao incrementar a corrente na pastilha, você perceberá que ela esquentará de um lado e esfriará do outro.

Cuidado, caso forneça muita corrente, coloque um dissipador no Mosfet e cuidado para não queimar sua protoboard.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/pastilha-peltier.html

https://www.sparkfun.com/products/10080

http://tomswiki.wetpaint.com/page/Peltier+%28TEC%29+Cooling

http://en.wikipedia.org/wiki/Thermoelectric_effect

http://bildr.org/2012/03/rfp30n06le-arduino/

http://sparkfun.com/datasheets/Components/General/Peltier_Testing.pde

47. Tutorial: como utilizar o Sensor de temperatura DS18B20 com Arduino

Neste tutorial vamos mostrar como utilizar o sensor de temperatura DS18B20 à prova de água com Arduino. Este sensor utiliza o protocolo digital de apenas um fio (One Wire), portanto apenas uma porta digital para comunicação.

Para fazer as ligações, o sensor contém três fios: um vermelho (VCC), um preto (GND) e um branco (Dados). Conecte o vermelho no 5V do Arduino e o preto no GND do Arduino. Conecte o fio branco no pino digital 10 e um resistor de 4k7 ohm entre o fio branco e o VCC (resistor de Pull-Up). Para maiores informações sobre o sensor utilizado, clique aqui!

Agora que feito as ligações, baixe a biblioteca necessária aqui para fazer a leitura do sensor. Extraia a biblioteca OneWire para dentro da pasta "libraries" localizada dentro da pasta do Arduino.

Abra a IDE do Arduino e cole a programação abaixo:

```
#include <OneWire.h>
int SensorPin = 10;

OneWire ds(SensorPin);
void setup(void) {
Serial.begin(9600);
}
void loop(void) {
float temp = getTemp();
Serial.println(temp);

delay(100);

}

float getTemp(){
byte data[12];
byte addr[8];
if ( !ds.search(addr)) {
//no more sensors on chain, reset search
ds.reset_search();
return -1000;
}
if ( OneWire::crc8( addr, 7) != addr[7]) {
Serial.println("CRC is not valid!");
return -1000;
}
if ( addr[0] != 0x10 && addr[0] != 0x28) {
Serial.print("Device is not recognized");
return -1000;
}
ds.reset();
ds.select(addr);
ds.write(0x44,1);
byte present = ds.reset();
ds.select(addr);
ds.write(0xBE);

for (int i = 0; i < 9; i++) {
data[i] = ds.read();
}

ds.reset_search();

byte MSB = data[1];
byte LSB = data[0];
float TRead = ((MSB 8) | LSB);
float Temperature = TRead / 16;

return Temperature;

}
```

A programação acima adquire a temperatura em graus Celsius. O Arduino verifica se o sensor está conectado corretamente. Caso estiver, prossegue para a aquisição de dados. Para a aquisição de dados é necessário mandar um endereço para detectar o sensor, um endereço para a leitura dos dados do sensor. E depois o Arduino poder converter e mostrar a leitura exata no Serial Monitor.

Dentro da IDE do Arduino, selecione a versão da sua placa Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e depois clique em "UPLOAD". Ao terminar de fazer o "UPLOAD", abra a Serial Monitor e selecione "9600" de baud. Você verá o valor da temperatura adquirido pelo sensor em graus Celsius como mostra a figura abaixo:

E é isso! Esperamos que tenha gostado! Caso tenha dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/34-sensores/sensor-de-te...

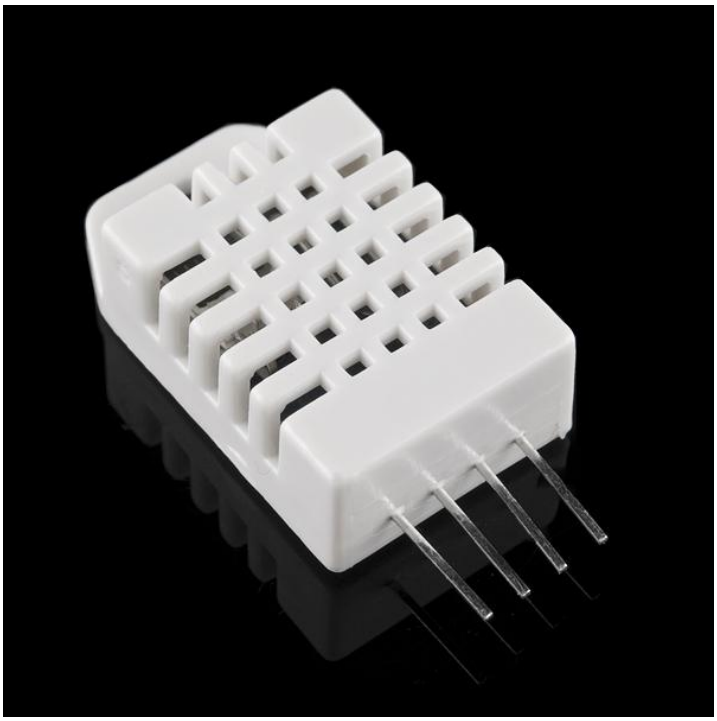
http://www.sparkfun.com/products/11050

http://bildr.org/2011/07/ds18b20-arduino/

http://www.arduino.cc/playground/Learning/OneWire

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Temp/DS18B20...

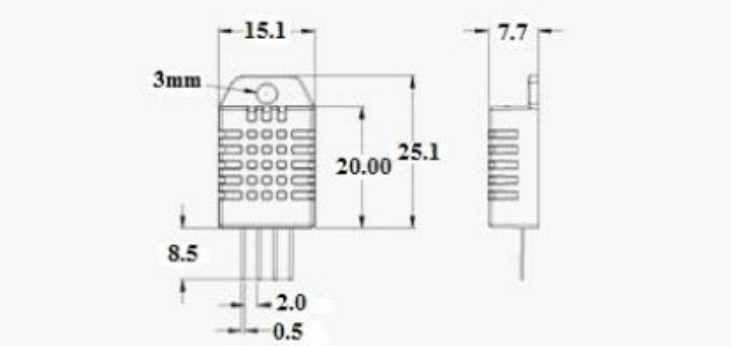
48. Tutorial: como utilizar o Sensor de temperatura e umidade RHT03 com Arduino



O sensor de temperatura e umidade RHT03 é um sensor de baixo custo e fácil utilização. A leitura é feita apenas por uma única saída do sensor. O sensor já está calibrado e não precisa de componentes extras para funcionamento. Exige apenas um Resistor de Pull-UP no pino de leitura.

Para utilizá-lo com Arduino é necessário a biblioteca dht.cpp e dht.h que pode ser adquirida aqui. Na página não tem nenhum link e sim o próprio código postado na página. Ainda na página da biblioteca, está escrito o dht.cpp e logo abaixo está o código da biblioteca. Copie e cole em um editor de texto, salve-o como dht.cpp e coloque em uma pasta própria (você pode chamá-la de DHT). Idem para o dht.h. Agora copie a pasta que você colocou os arquivos dht.cpp e dht.h e cole dentro da pasta "libraries" da IDE do Arduino.

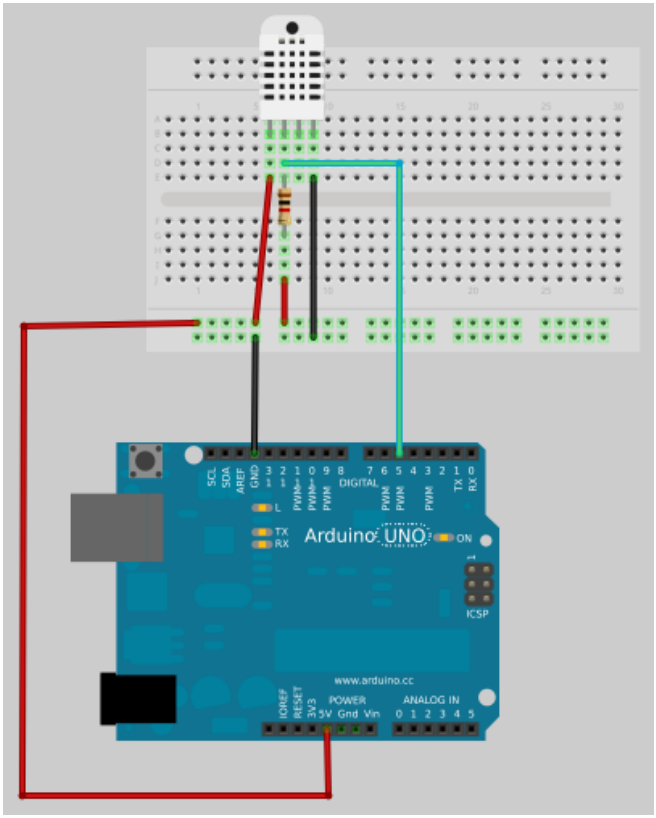
Vendo o datasheet do RHT03, podemos ver a pinagem e assim fazer as ligações com Arduino:



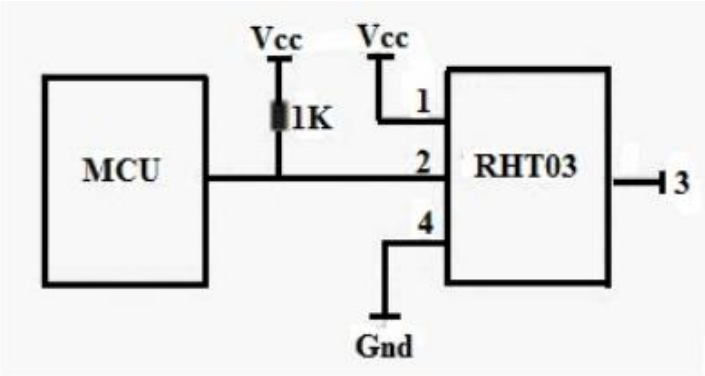
Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA–signal
3	NULL
4	GND

Agora que sabemos sua pinagem, vamos para a ligação do sensor RHT03 com Arduino:



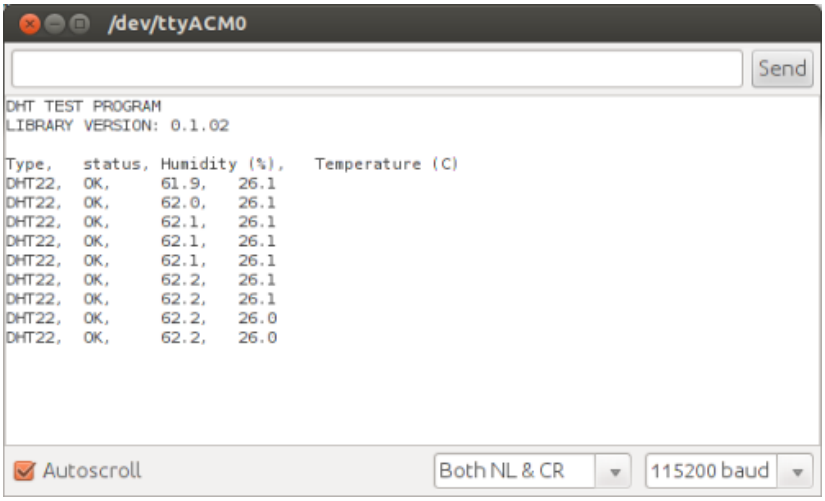
O sensor RHT03, funciona a partir de 3.3V a 6V, neste tutorial fizemos com a saída de 5V do Arduino, mas nada impede que você utilize o 3.3V do Arduino. O pino 2 do sensor vai para o pino digital 5 do Arduino com um resistor de 1Kohm de Pull-Up como é visto no datasheet pela imagem abaixo:



E o pino 3 fica em aberto (Not Connected). Agora que você já colocou a biblioteca na pasta "libraries" localizada na pasta da IDE do Arduino e já fez as ligações demonstradas, vamos para a programação exemplo modificada do site:

```
#include <dht.h>
dht DHT;

#define DHT22_PIN 5
void setup()
{
Serial.begin(115200);
Serial.println("DHT TEST PROGRAM ");
Serial.print("LIBRARY VERSION: ");
Serial.println(DHT_LIB_VERSION);
Serial.println();
Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
}
void loop()
{
// READ DATA
Serial.print("DHT22, \t");
int chk = DHT.read22(DHT22_PIN);
switch (chk)
{
case DHTLIB_OK:
Serial.print("OK,\t");
break;
case DHTLIB_ERROR_CHECKSUM:
Serial.print("Checksum error,\t");
break;
case DHTLIB_ERROR_TIMEOUT:
Serial.print("Time out error,\t");
break;
default:
Serial.print("Unknown error,\t");
break;
}
// DISPLAY DATA
Serial.print(DHT.humidity, 1);
Serial.print(",\t");
Serial.println(DHT.temperature, 1);
delay(1000);
}
```

Na programação exemplo, se as ligações estiverem certas, o sensor vai iniciar e mostrará a umidade e temperatura como mostrado na figura abaixo:



No Serial Monitor a umidade relativa do ar e a temperatura em graus Celsius!

E pronto! Simples e fácil! Esperamos que tenham gostado!! Se tiverem dúvidas, postem aqui mesmo no blog! Temos um post para sugestões de tutoriais clicando aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas,cliquem aqui e aqui, respectivamente!

Referências:

http://arduino.cc/playground/Main/DHTLib

http://www.labdegaragem.org/loja/index.php/34-sensores/sensor-de-te...

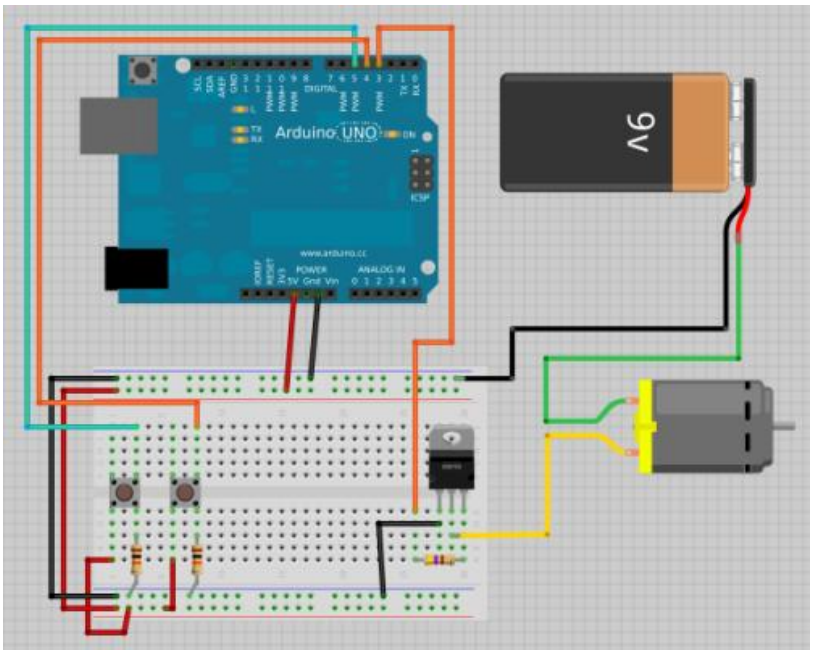
http://www.sparkfun.com/products/10167

49. Tutorial: utilizando Arduino e transistor para controlar motor DC

Neste tutorial vamos mostrar como controlar um motor DC com Arduino e um transistor utilizando a saída PWM (Pulse Width Modulation). Para saber mais sobre PWM, clique aqui.

Com dois botões, vamos controlar o motor DC conectado ao transistor e por fim o Arduino.

A ligação feitas está demonstrada abaixo:



A bateria de 9V é apenas para indicar a tensão de alimentação. Verifique a tensão necessária do seu motor DC para não queimá-lo. O transistor utilizado neste tutorial é o BD139, mas pode utilizar outro contanto que o transistor seja NPN. Para ver o datasheet, clique aqui.

A programação para controlar o motor DC está mostrada abaixo:

```
int motorPin = 3;
int incPin = 4;
int decPin = 5;
int val=0;
int incButton = LOW;
int decButton = LOW;
void setup()
{
pinMode(incPin, INPUT);
pinMode(decPin, INPUT);
pinMode(motorPin, OUTPUT);
analogWrite(motorPin, 0);
Serial.begin(9600);
}
void loop()
{
incButton = digitalRead(incPin);
decButton = digitalRead(decPin);
if (incButton == HIGH)
{
val++;
}
if(decButton == HIGH)
{
val=val-1;
}
Serial.println(val);
delay(100);
analogWrite(motorPin, val);
}
```

Ao apertar o botão da porta digital 4, o motor irá girar mais rápido. E ao apertar o botão da porta digital 5, o motor irá girar mais devagar.

E é isso! Esperamos que tenha gostado! Se tiver dúvidas sobre o tutorial, poste aqui enste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/29-arduino.html

http://www.labdegaragem.org/loja/index.php/32-kits/starter-kit-com-...

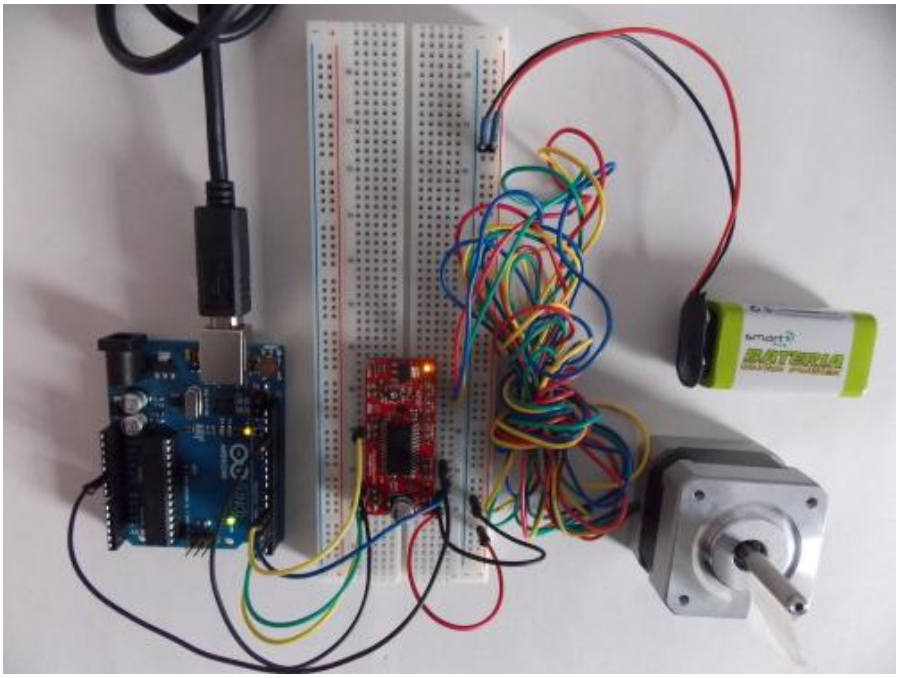
http://www.datasheetcatalog.net/pt/datasheets_pdf/B/D/1/3/BD139.shtml

http://labdegaragem.com/profiles/blogs/tutorial-o-que-e-como-utiliz...

http://arduino.cc/en/

50. Tutorial: Controle do deslocamento do Motor de Passo com Arduino + Easy Driver

Este tutorial mostra como controlar o deslocamento (horário e anti-horário) de um motor de passo bipolar com o Arduino e o *Easy Driver,* rotacionando o motor em vários modos diferentes (passo inteiro, meio passo, 1/4 de passo e 1/8 de passo).



Para isto, serão necessários os seguintes componentes:

- 1 placa Arduino

- 1 motor de passos bipolar (MERCURY SM-42BYG011-25)

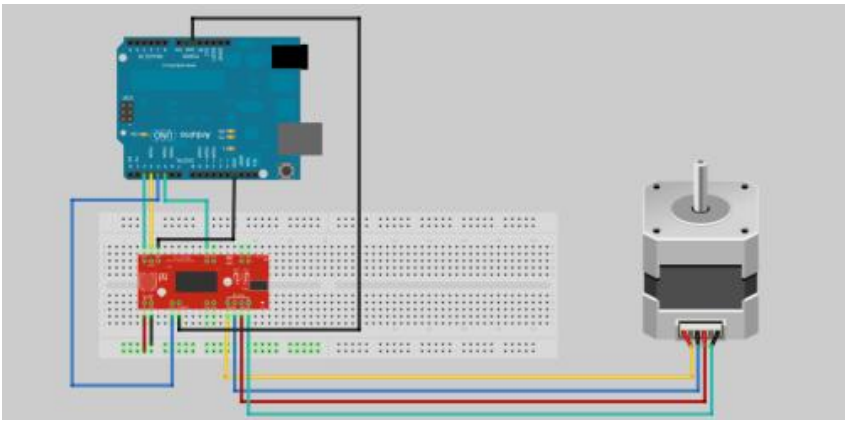
- 1 Easy Driver

- Jumpers para conexão

- 1 Fonte/bateria de 9V

O motor de passo é como motor elétrico comum (recebe a energia elétrica e converte-a em mecânica), porém ele divide sua rotação em vários intervalos (em graus), o que damos o nome de *passos.* Este utilizado no projeto, é de 200 passos por rotação (sendo cada pulso um passo). Pode-se dividir/configurar tanto os passos como os intervalos. E isto será tarefa do *firmware*.

Agora, monte o circuito a seguir com seus componentes:



Obs: Os barramentos de baixo da protoboard serão ligados a sua fonte/bateria de 9V mais adiante.

Então, abra a IDE do Arduino e digite o seguinte código:

```
/* Pinos de Configuração do Deslocamento (no Easy Driver):
////////////////////////////////////////////
// MS1 MS2 //
// //
// LOW LOW = Passo completo //
// HIGH LOW = Meio passo //
// LOW HIGH = Quarto de Passo //
// HIGH HIGH = Oitavo de Passo //
// //
////////////////////////////////////////////
*/
#define step_pin 3    // Define o pino 3 como pino dos passos
#define dir_pin 2    // Define o pino 2 como pino de direção
#define MS1 5     // Define o pino 5 como "MS1"
#define MS2 4      // Define o pino 4 como "MS2"
int direcao;    // Para determinar o sentido do motor
int passos = 200;      // Número de passos que você deseja executar (para passos completos, 200 = 1 volta)
void setup() {
  pinMode(MS1, OUTPUT);     // Configura "MS1" como saída
  pinMode(MS2, OUTPUT);     // Configura "MS2" como saída
  pinMode(dir_pin, OUTPUT);    // Configura "dir_pin" como saída
  pinMode(step_pin, OUTPUT);    // Configura "step_pin" como saída
  digitalWrite(MS1, LOW);      // Configura divisão de passos do motor (ver acima)
  digitalWrite(MS2, LOW);    // Configura divisão de passos do motor (ver acima)
  digitalWrite(dir_pin, LOW);    // Sentido (HIGH = anti-horário / LOW = horário) - Também pode ser alterado
}
void loop() {
while(passos>=0) {    // Enquanto o valor de passos for maior ou igual a zero
       digitalWrite(step_pin, HIGH);    // Envia nível lógico alto para o pino de passos do motor
       delay(5);    // Aguarda 5ms para o próximo passo
       digitalWrite(step_pin, LOW);    // Envia nível lógico baixo para o pino de passos do motor
       delay(5);    // Aguarda 5ms para o próximo passo
      passos--;   // Decrementa a variável "passos"

  }
}
```

Conecte sua fonte/bateria de 9V no seu circuito. Então faça o Upload do programa para o seu Arduino.

O seu controle de deslocamento do motor bipolar de passos está pronto !!! Altere as variáveis indicadas no código e execute o programa quantas vezes quiser. Esperemos que tenha gostado. Qualquer dúvida, poste aqui mesmo.

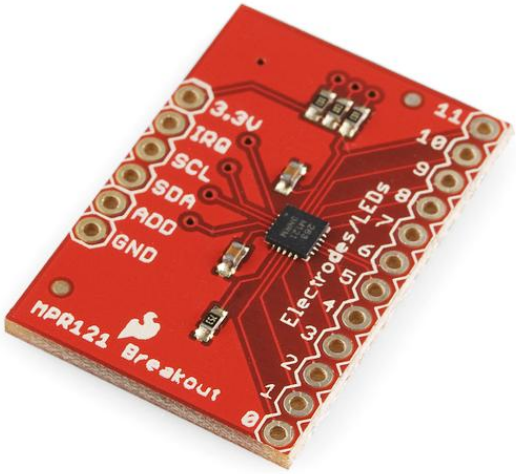
Até a próxima !!!

Links de Referência:

- Motor de Passo

- Datasheet do Motor MERCURY SM-42BYG011-25

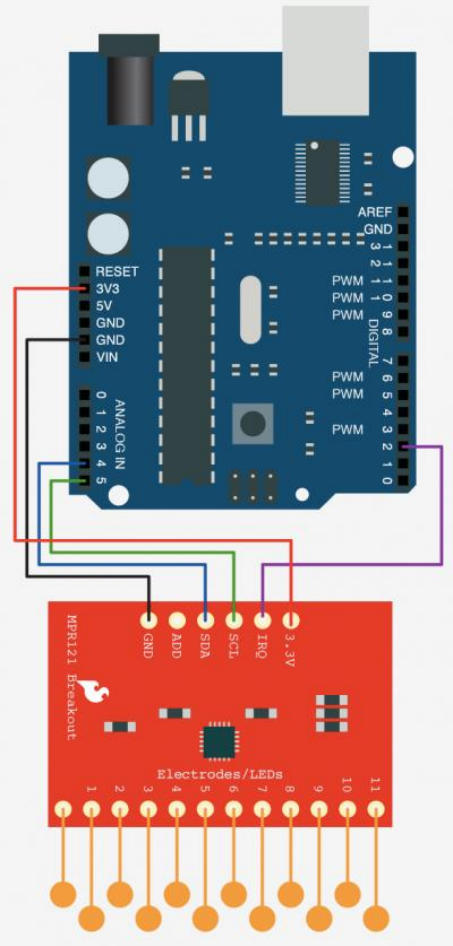
51. Tutorial: como utilizar a Breakout Sensor de toque capacitivo com Arduino



Neste tutorial vamos mostrar como utilizar a breakout sensor de toque capacitivo com Arduino.

Esta breakout funciona por interface I2C e pode controlar até 12 eletrodos ou apenas LEDS.

Primeiramente faça seguinte ligação:



Depois abra a IDE do Arduino e passe a seguinte programação:

```
// MPR121 Register Defines
#define MHD_R 0x2B
#define NHD_R 0x2C
#define NCL_R 0x2D
#define FDL_R 0x2E
#define MHD_F 0x2F
#define NHD_F 0x30
#define NCL_F 0x31
#define FDL_F 0x32
#define ELE0_T 0x41
#define ELE0_R 0x42
#define ELE1_T 0x43
#define ELE1_R 0x44
#define ELE2_T 0x45
#define ELE2_R 0x46
#define ELE3_T 0x47
#define ELE3_R 0x48
#define ELE4_T 0x49
#define ELE4_R 0x4A
#define ELE5_T 0x4B
#define ELE5_R 0x4C
#define ELE6_T 0x4D
#define ELE6_R 0x4E
#define ELE7_T 0x4F
#define ELE7_R 0x50
#define ELE8_T 0x51
#define ELE8_R 0x52
```

```
#define ELE9_T 0x53
#define ELE9_R 0x54
#define ELE10_T 0x55
#define ELE10_R 0x56
#define ELE11_T 0x57
#define ELE11_R 0x58
#define FIL_CFG 0x5D
#define ELE_CFG 0x5E
#define GPIO_CTRL0 0x73
#define GPIO_CTRL1 0x74
#define GPIO_DATA 0x75
#define GPIO_DIR 0x76
#define GPIO_EN 0x77
#define GPIO_SET 0x78
#define GPIO_CLEAR 0x79
#define GPIO_TOGGLE 0x7A
#define ATO_CFG0 0x7B
#define ATO_CFGU 0x7D
#define ATO_CFGL 0x7E
#define ATO_CFGT 0x7F

// Global Constants
#define TOU_THRESH 0x06
#define REL_THRESH 0x0A
#include <Wire.h>
int irqpin = 2; // Digital 2
boolean touchStates[12]; //to keep track of the previous touch states
void setup(){
pinMode(irqpin, INPUT);
digitalWrite(irqpin, HIGH); //enable pullup resistor

Serial.begin(9600);
Wire.begin();
mpr121_setup();
}
void loop(){
readTouchInputs();
}

void readTouchInputs(){
if(!checkInterrupt()){

//read the touch state from the MPR121
Wire.requestFrom(0x5A,2);

byte LSB = Wire.read();
byte MSB = Wire.read();

uint16_t touched = ((MSB 8) | LSB); //16bits that make up the touch states

for (int i=0; i < 12; i++){ // Check what electrodes were pressed
if(touched & (1i)){

if(touchStates[i] == 0){
//pin i was just touched
Serial.print("pin ");
Serial.print(i);
Serial.println(" was just touched");

}else if(touchStates[i] == 1){
```

```
//pin i is still being touched
}

touchStates[i] = 1;
}else{
if(touchStates[i] == 1){
Serial.print("pin ");
Serial.print(i);
Serial.println(" is no longer being touched");

//pin i is no longer being touched
}

touchStates[i] = 0;
}

}

}
}

void mpr121_setup(void){
set_register(0x5A, ELE_CFG, 0x00);

// Section A - Controls filtering when data is > baseline.
set_register(0x5A, MHD_R, 0x01);
set_register(0x5A, NHD_R, 0x01);
set_register(0x5A, NCL_R, 0x00);
set_register(0x5A, FDL_R, 0x00);
// Section B - Controls filtering when data is < baseline.
set_register(0x5A, MHD_F, 0x01);
set_register(0x5A, NHD_F, 0x01);
set_register(0x5A, NCL_F, 0xFF);
set_register(0x5A, FDL_F, 0x02);

// Section C - Sets touch and release thresholds for each electrode
set_register(0x5A, ELE0_T, TOU_THRESH);
set_register(0x5A, ELE0_R, REL_THRESH);

set_register(0x5A, ELE1_T, TOU_THRESH);
set_register(0x5A, ELE1_R, REL_THRESH);

set_register(0x5A, ELE2_T, TOU_THRESH);
set_register(0x5A, ELE2_R, REL_THRESH);

set_register(0x5A, ELE3_T, TOU_THRESH);
set_register(0x5A, ELE3_R, REL_THRESH);

set_register(0x5A, ELE4_T, TOU_THRESH);
set_register(0x5A, ELE4_R, REL_THRESH);

set_register(0x5A, ELE5_T, TOU_THRESH);
set_register(0x5A, ELE5_R, REL_THRESH);

set_register(0x5A, ELE6_T, TOU_THRESH);
set_register(0x5A, ELE6_R, REL_THRESH);

set_register(0x5A, ELE7_T, TOU_THRESH);
set_register(0x5A, ELE7_R, REL_THRESH);
```

```
set_register(0x5A, ELE8_T, TOU_THRESH);
set_register(0x5A, ELE8_R, REL_THRESH);

set_register(0x5A, ELE9_T, TOU_THRESH);
set_register(0x5A, ELE9_R, REL_THRESH);

set_register(0x5A, ELE10_T, TOU_THRESH);
set_register(0x5A, ELE10_R, REL_THRESH);

set_register(0x5A, ELE11_T, TOU_THRESH);
set_register(0x5A, ELE11_R, REL_THRESH);

// Section D
// Set the Filter Configuration
// Set ESI2
set_register(0x5A, FIL_CFG, 0x04);

// Section E
// Electrode Configuration
// Set ELE_CFG to 0x00 to return to standby mode
set_register(0x5A, ELE_CFG, 0x0C); // Enables all 12 Electrodes


// Section F
// Enable Auto Config and auto Reconfig
/*set_register(0x5A, ATO_CFG0, 0x0B);
set_register(0x5A, ATO_CFGU, 0xC9); // USL = (Vdd-
0.7)/vdd*256 = 0xC9 @3.3V set_register(0x5A, ATO_CFGL, 0x82); // LSL = 0.65*USL = 0x82
@3.3V
set_register(0x5A, ATO_CFGT, 0xB5);*/ // Target = 0.9*USL = 0xB5 @3.3V

set_register(0x5A, ELE_CFG, 0x0C);

}

boolean checkInterrupt(void){
return digitalRead(irqpin);
}

void set_register(int address, unsigned char r, unsigned char v){
Wire.beginTransmission(address);
Wire.write(r);
Wire.write(v);
Wire.endTransmission();
}
```

A programação configura todos os registros do controlador e a comunicação I2C com a biblioteca Wire.h. Depois ao tocar em um eletrodo, o Serial Monitor mostrará qual eletrodo foi tocado.

Agora, selecione a versão do seu Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSBx, ttyACMx, etc) e clique em UPLOAD. Ao terminar, abra o Serial Monitor e toque em um dos eletrodos. No Serial Monitor você verá qual pino foi tocado!

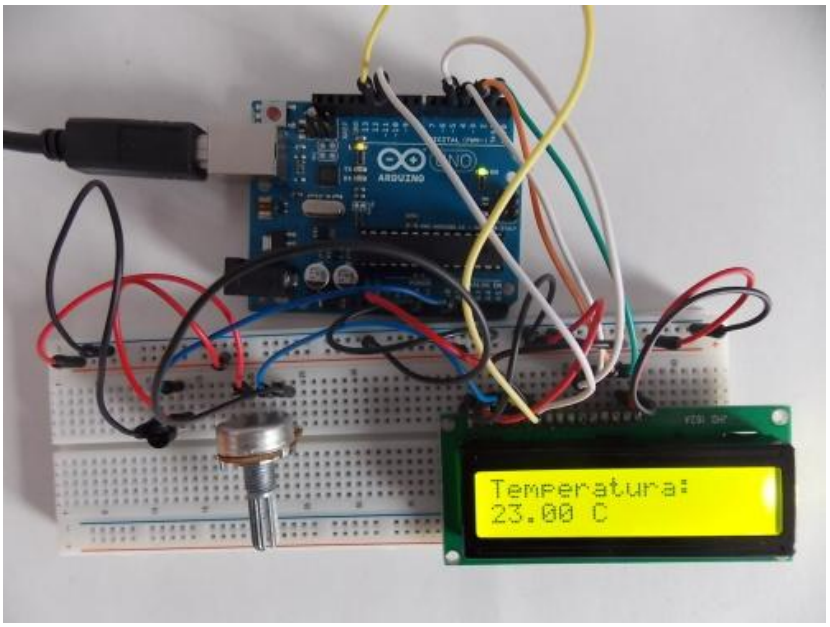
E é isso! Esperamos que tenha gostado! Em caso de dúvidas poste aqui mesmo neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

https://www.sparkfun.com/products/9695?

http://bildr.org/page/10/

52. Tutorial: Mostrar a temperatura ambiente com o Arduino em um display LCD 16x2



Este tutorial mostra uma maneira interessante de fazer a leitura da temperatura ambiente com o Arduino, e exibi-la em um display LCD 16x2 (em graus Celsius), utilizando conceitos básicos de sua programação e o sensor de temperatura LM35DZ.

Para isto, serão necessários os seguintes componentes:

- 1 placa Arduino

- 1 display LCD 16x2

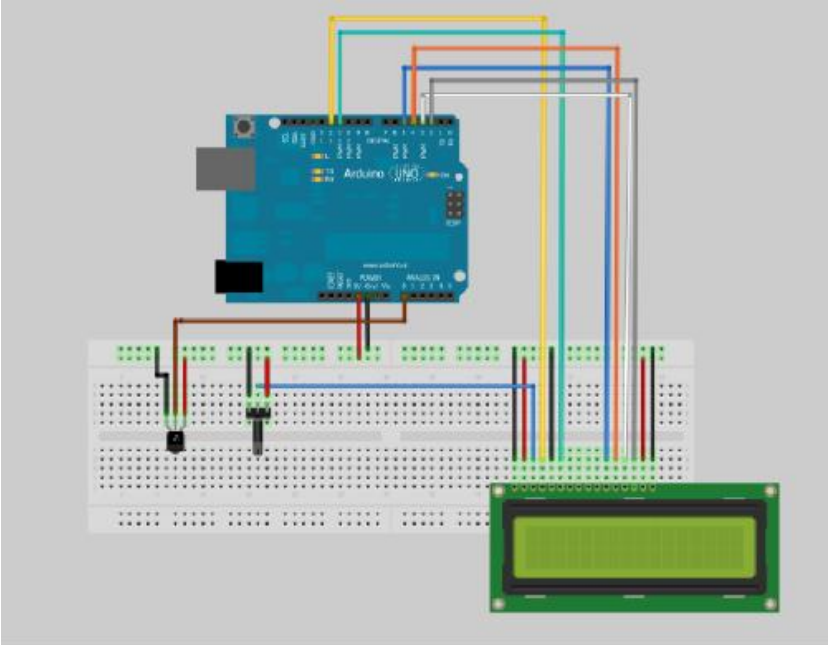
- 1 potenciômetro de 10K

- 1 sensor de temperatura LM35

- Jumpers para conexão

Este sensor captura a temperatura externa e converte-a em um valor de tensão correspondente. No circuito, conectamos a saída Vo do sensor a porta analógica A0 do Arduino, que por sua vez, converte este sinal em um valor (float) de temperatura e assim, exibe no LCD. O potenciômetro de 10K ajusta o contraste do display.

A seguir, monte o circuito abaixo com o Arduino e os outros componentes:



Então, abra a IDE do Arduino e digite o seguinte código:

```
#include  <LiquidCrystal.h>        // Inclui a biblioteca para utilizar o LCD.
#define         sensor          0       // Define o pino A0 como “sensor”

int  Ventrada;          // Variável para ler o sinal do pino do Arduino
float  Temperatura; // Variável que recebe o valor convertido para temperatura.

LiquidCrystal  lcd    (12, 11, 5, 4, 3, 2);
/* Esta função acima declara quais os pinos do Arduino serão utilizados para o controle do LCD
 */

void  setup()
{
  lcd.begin(16, 2);                       // Diz para o Arduino que o display é 16x2.
  lcd.print("Temperatura: ");          // Manda o texto para a tela do display
}

void  loop()
{
  Ventrada = analogRead (sensor);  /*  Manda o Arduino ler o pino e armazena
 o valor em “Ventrada”. */
  Temperatura=(500*Ventrada)/1023;  /* Converte o valor de tensão em
temperatura e armazena na variável “Temperatura” */
  lcd.setCursor(0, 1);          // Move o cursor do display para a segunda linha.
  lcd.print(Temperatura);    // Exibe o valor de temperatura no display.
  lcd.print(" C");        // Escreve “C” para dizer que a escala é Celsius.

  delay(1000); /* Aguarda 1 segundo para efetuar uma nova leitura de
```

```
temperatura. */
}
```

Finalmente, faça o Upload para o seu Arduino.

O seu projeto para indicar temperatura está pronto !!! Esperamos que tenha gostado. Qualquer dúvida, poste aqui mesmo.

Obs: o sensor LM35DZ converte temperaturas na faixa de -55ºC até 150ºC.

Para quem quiser, esta é a explicação para o cálculo de conversão de tensão para temperatura feita no software:

Neste sensor, para cada 1ºC recebido, sua saída é acrescida em 10mV, isto em uma faixa de 0 a 5V, com 10 bits de resolução para conversão Analógica/Digital (1024 valores diferentes para representar a temperatura). Assim, o valor máximo (1023, pois é de 0 a 1023) corresponderá aos 5V. A metade corresponderá a 511 ou 2,5V, e assim por diante. Para efeitos de cálculo, haveria 5V na saída do sensor para uma temperatura de 500ºC (o que é diferente na prática). Porém, esta consideração nos permite generalizar isto para a seguinte regra de três:

Temperatura ----- Ventrada

500ºC ----- 1023 (valores máximos)

Assim, teremos:

Temperatura = (500*Ventrada)/1023, e este valor é exibido no display.

Links de Referência:

- LCD com o Arduino

- "Datasheet" do sensor LM35

53. Tutorial: Controlando motor de passo com L293D e Arduino

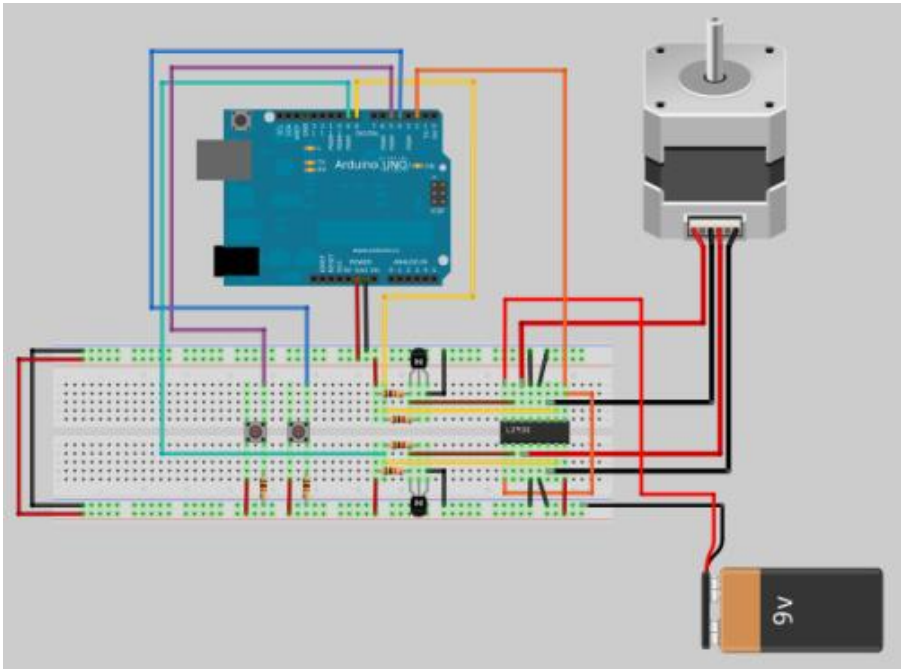
Neste tutorial, vamos mostrar como controlar um motor de passo utilizando o circuito integrado L293D e Arduino.

O L293D é um circuito integrado de ponte-H. Com ele é possível controlar motores DC e motores de passo. Para ver o datasheet, clique aqui! Em relação ao easydriver, você pode configurar a velocidade e o número de passos para dar mais velocidade e/ou mais torque. No L293D você pode utilizar um motor de passo de no máximo 600mA e 36V de alimentação para o motor de passo. No easydriver, porém, é mais fácil sua utilização e mais simples. E não tem necessidade de colocar outros componentes como no L293D. E você pode utilizar um motor de passo de até 750mA e 30V de alimentação do motor de passo.

Para este tutorial, vamos utilizar:

- 1x Circuito Integrado L293D
- 1x Arduino
- 4x resistor de 10K ohm
- 2x resistor de 1K ohm
- 2x transistor BC547 NPN (Pode ser utilizado outro equivalente, contanto que seja NPN)
- 2x botões (pushbutton)
- 1x Motor de passo

Primeiramente, faça a seguinte ligação:



A bateria de 9V é apenas a demonstração da fonte externa para alimentar o motor de passo.

Veja no datasheet do seu motor de passo qual a tensão utilizada.

Agora abra a IDE do Arduino e passe a seguinte programação:

```
#include <Stepper.h>  //Biblioteca já disponível na IDE do Arduino
const int steps=200;  //Número de passos para o motor
int buttonState=0;
int buttonState1=0;
Stepper motor(steps,8,9); //Pinos 8 e 9 do Arduino
void setup()
{
motor.setSpeed(100);    //Velocidade da rotação do motor (RPM)
Serial.begin(9600);
pinMode(4,INPUT);   //Botão 1
pinMode(5,INPUT);   //Botão 2
pinMode(2,OUTPUT);   //Enable do L293D
digitalWrite(2,LOW);
}
void loop()
{
buttonState=digitalRead(4);
buttonState1=digitalRead(5);
if(buttonState==HIGH)   //Gira para um lado
{
digitalWrite(2,HIGH);
Serial.println("botao");
motor.step(steps);
}
else if(buttonState1==HIGH)  //Gira para o outro lado
{
digitalWrite(2,HIGH);
Serial.println("botao 1");
motor.step(-steps);
}
else   //Fica parado
```

```
{
digitalWrite(2,LOW);
Serial.println("Parado");
motor.step(0);

}
}
```

Conecte seu Arduino na porta USB do seu computador, selecione a versão do seu Arduino (UNO, Duemilanove, etc) em "Tools/Boards" e a porta (COMx, ttyUSBx, ttyACMx, etc) em "Tools/Serial Port". E clique em "UPLOAD".

Apertando um botão o motor de passo girará para um lado. Se apertar o outro botão o motor de passo girará para o outro lado. Caso seu motor de passo não gire, experimente inverter os fios das bobinas. Ou experimente diminuir ou aumentar o número de passos do motor e a velocidade.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/36-motores-afins/motor-d...

http://www.labdegaragem.org/loja/index.php/36-motores-afins/easydri...

http://www.labdegaragem.org/loja/index.php/29-arduino.html

http://www.labdegaragem.org/loja/index.php/30-compativeis.html

http://www.tigoe.net/pcomp/code/circuits/motors/stepper-motors/

http://arduino.cc/en/Tutorial/Button

http://arduino.cc/en/Reference/StepperBipolarCircuit

54. Tutorial: Utilizando interrupção e função Random() do Arduino

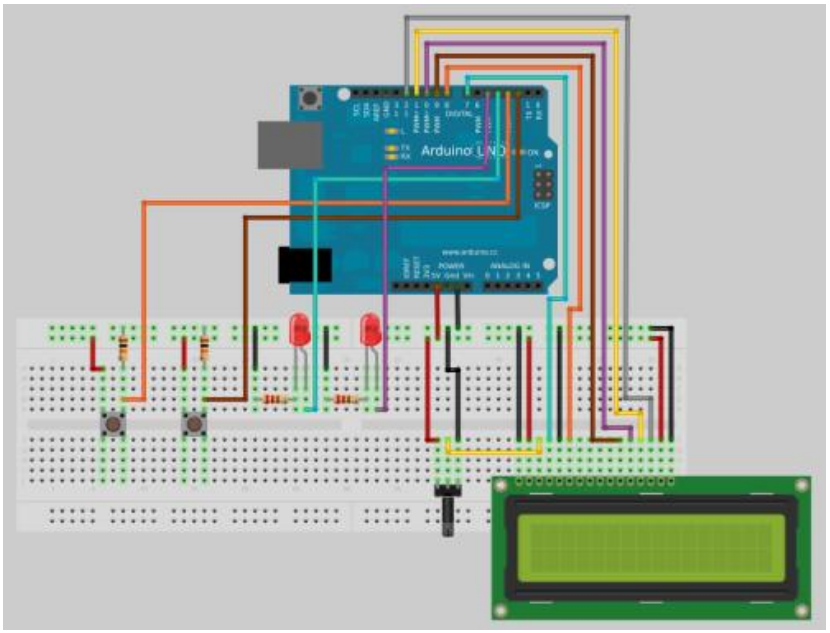
Este tutorial mostra como utilizar a função random() para gerar números aleatórios e mostra como utilizar a interrupção externa por meio de um botão.

A interrupção externa pode ser utilizada nos pinos digitais 2 e 3 do Arduino UNO e nos pinos digitais 18, 19, 20 e 21 do Arduino MEGA.

Ao apertar um botão como visto no vídeo, o Arduino entra na função da interrupção. Ao entrar na função da interrupção, o LED acende e mostra no LCD 16x2.

Para fazer a ligação abaixo, será necessário os seguintes componentes:

- 2x botão (pushbutton)
- 2x LED
- 2x resistor 10K ohm
- 2x resistor 220 ohm
- 1x potenciomêtro
- 1x LCD 16x2
- 1x Arduino UNO ou equivalente



Depois de feito as ligações acima, abra a IDE do Arduino e passe a seguinte programação:

```
#include <LiquidCrystal.h> // we need this library for the LCD commands
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
float noisy = 0;
void setup()
{
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
lcd.begin(16, 2); // need to specify how many columns and rows are in the LCD unit
lcd.setCursor(0,0);
lcd.println(" Lab de Garagem ");
lcd.setCursor(0,1);
delay(4000);
lcd.clear();
randomSeed(analogRead(0)); // reseed the random number generator with some noise
attachInterrupt(0, interruptone, RISING); // so when interrupt zero (digital pin 2) changes state, it
will trigger the interrupt and go to function 'panicone'
attachInterrupt(1, interrupttwo, RISING); // so when interrupt one (digital pin 3) changes state, it
will trigger the interrupt and go to function 'panictwo'
}
void loop()
{
noisy=random(1000);
lcd.setCursor(0,0);
lcd.write("Random Numbers!");
lcd.setCursor(0,1);
lcd.print("Number: ");
lcd.print(noisy,0);
delay(1000);
digitalWrite(4,LOW);
digitalWrite(5,LOW);
}
void interruptone()
{
lcd.clear();
lcd.println("Interrupt one ");
digitalWrite(4,HIGH);
}
void interrupttwo()
{
lcd.clear();
lcd.println("Interrupt two ");
digitalWrite(5,HIGH);
}
```

As interrupções são declaradas como:

- attachInterrupt(0, interruptone, RISING); - Quando o Arduino detectar que a entrada do pino 2 foi pra HIGH, a função interruptone será chamada;
- attachInterrupt(1, interrupttwo, RISING); - Quando o Arduino detectar que a entrada do pino 3 foi pra HIGH, a função interrupttwo será chamada.

Depois de copiar e colar o código acima na IDE do Arduino, conecte seu Arduino na porta USB do seu PC e selecione a versão do seu Arduino (UNO, Duemilanove, etc) em "Tools/Board" e selecione a porta em que seu Arduino está conectado (COMx, ttyUSBx, ttyACMx, etc). Clique em UPLOAD.

Ao terminar de fazer o UPLOAD. O Arduino funcionará como o vídeo.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui neste blog! Para sugestões de tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui! Até a próxima!

Referências:

http://tronixstuff.wordpress.com/2010/04/20/getting-started-with-ar...

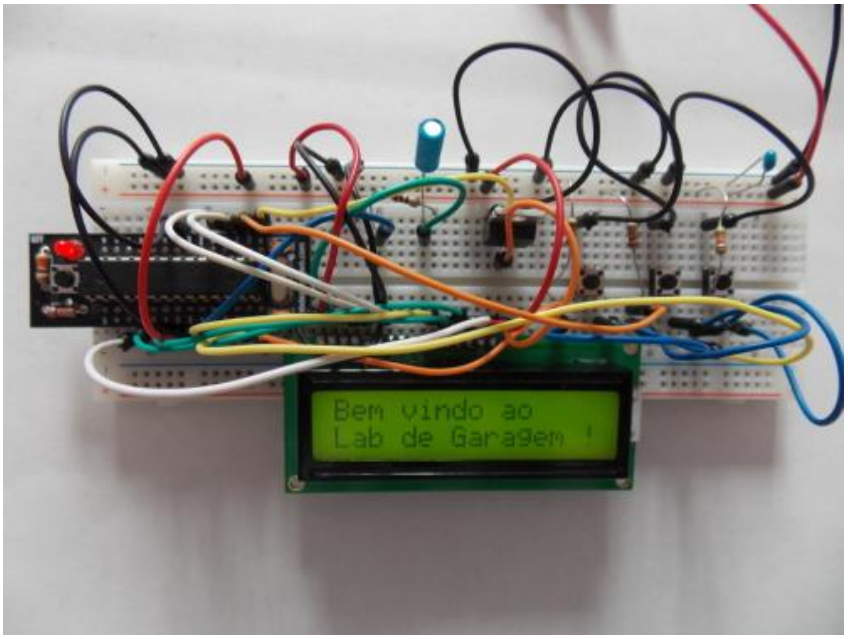
http://arduino.cc/en/Reference/AttachInterrupt

http://labdegaragem.com/profiles/blogs/tutorial-lcd-com-arduino

http://arduino.cc/en/Reference/Random

55. Tutorial: Backlight e contraste por software em LCD 16x2

Geralmente em projetos com display LCD, o contraste é controlado por um potenciômetro e o *backlight* é conectado diretamente à alimentação, portanto sempre ligado. Neste tutorial, será mostrada uma maneira de controlar estes recursos em um LCD 16x2 via software com o Garagino ou Arduino, controlando o *backlight* por um certo tempo ligado (o que economiza energia) e o contraste através de botões. O Garagino é mais barato que o Arduino e pode ser incorporado em seus projetos com maior facilidade por ser bem compacto. Ele pode ser programado via Arduino ou com a ajuda de um conversor USB/Serial. No entanto, você poderá utilizar qualquer um deles para este tutorial, as duas opções serão mostradas.



Para esta montagem, serão necessários os seguintes componentes:

- 1x Arduino Uno ou Garagino

- 3x Resistores de 10K Ohms

- 1x Resistor de 100 Ohms

- 1x Capacitor eletrolítico de 10uF

- 1x Capacitor cerâmico de 100nF

- 3x Push buttons

- 1x Display LCD 16x2

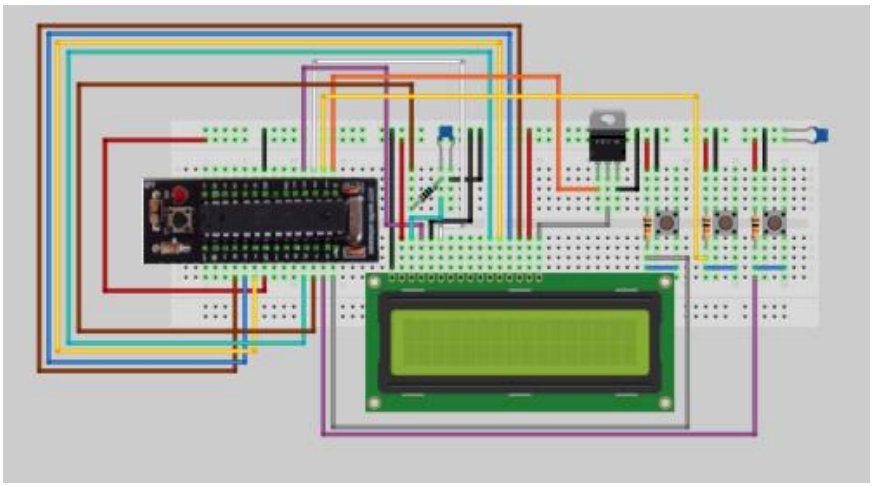
- 1x MOSFET Canal N

- Jumpers para Conexão

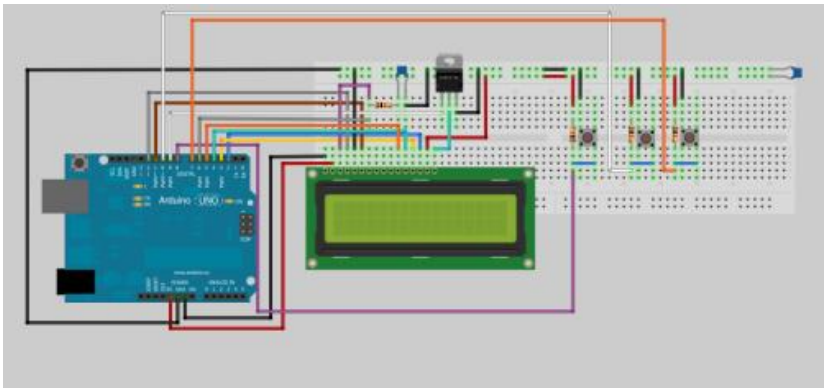
O controle do display é feito basicamente por três *push buttons* ligados aos pinos D7, D8 e D10 do Garagino/Arduino, que por sua vez lê o sinal dos mesmos e controla o display por meio de outros dois pinos (D9 - liga o *backlight* - fazendo com que o MOSFET seja ou não chaveado, e D6 - ajuste de *contraste por PWM*). O capacitor de 100nF conectado à alimentação do circuito serve como filtro de ruídos, o de 10uF conectado ao resistor funciona como um filtro passa baixa mantendo o nível DC estável na entrada do pino Vo (ajuste de contraste) no LCD.

A seguir, monte um dos dois circuitos como mostrado abaixo:

Com o Garagino (clique para ampliar):



Com o Arduino (clique para ampliar):



Então, abra a IDE do Arduino e insira o seguinte código:

```
// Firmware para controle do display LCD 16x2
#include <LiquidCrystal.h>
#define botao_backlight 8 // Pino para verificação de acionamento do backlight
#define backlight 9 // Pino que habilita o acionamento do backlight (pelo MOSFET)
#define contraste 6 // Pino do contraste no LCD
#define ajuste1 7 // Pino para aumentar o contraste (diminui o valor do PWM no pino 6)
#define ajuste2 10 // Pino para diminuir o contraste (aumenta o valor do PWM no pino 6)
long intervalo = 6000; // Intervalo que o backlight permanecerá aceso quando for acionado
unsigned long valoranterior = 0; // Variável para atualização para contar o tempo de intervalo
unsigned char valor = 0; // Variável com o valor do contraste
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // Pinos do Arduino utilizados no LCD
void setup() {
pinMode(botao_backlight, INPUT); // Configurações de I/O
pinMode(ajuste2, INPUT);
pinMode(ajuste1, INPUT);
pinMode(backlight, OUTPUT);
pinMode(contraste, OUTPUT);

lcd.begin(16, 2); // Declaração do número de linhas e colunas do display
lcd.setCursor(0, 0); // Move o cursor para coluna 0 linha 0
lcd.print("Bem vindo ao"); // Exibe esta mensagem na primeira linha
lcd.setCursor(0, 1); // Move o cursor para coluna 0 linha 1
lcd.print("Lab de Garagem !"); // Exibe mensagem na segunda linha
}
void loop() {
/* Essa parte monitora o pino de backlight do display e liga-o
caso o botão "backlight" seja colocado no GND. Isto por 6 segundos (valor de "intervalo") */
unsigned long valoratual = millis(); // Valor atual que armazena o tempo a partir de quando o
ATmega inicializado.
if((digitalRead(botao_backlight)) == LOW) { // Se o pino "backlight" estiver no GND,
digitalWrite(backlight, HIGH); // Após soltar o botão, liga o backlight
}
if(((valoratual - valoranterior) > intervalo)) { // Se passar o tempo de intervalo (no caso 6s),
valoranterior = valoratual; // atualiza o valor anterior para uma nova contagem
digitalWrite(backlight, LOW); // Desliga o backlight
}
/* Esta parte monitora os pinos de ajuste de contraste ("ajuste1" e ajuste2")
a fim de regulá-lo no display */
if(digitalRead(ajuste2) == LOW) { // Se o pino/botão "ajuste2" for colocado no GND,
while(digitalRead(ajuste2) == LOW) { // espera até soltar o botão
continue;
}
if(valor < 255) { // Após soltar o botão, se o valor do PWM não for maior que o máximo (255),
valor = valor + 5; // incrementa a variável "valor" em 5 (diminui o contraste)
}
analogWrite (contraste, valor); // Exibe este valor de contraste no pino Vo do display
}
if(digitalRead(ajuste1) == LOW) { // Se o pino/botão "ajuste1" for colocado no GND,
while(digitalRead(ajuste1) == LOW) { // espera até soltar o botão
continue;
}
if(valor > 0) { // Após soltar o botão, se o valor do PWM não for menor que mínimo (0),
valor = valor - 5; // decrementa a variável "valor" em 5 (aumenta o contraste)
}
analogWrite (contraste, valor); // Exibe este valor de contraste no pino Vo do display
}
}
```

Faça o upload para o seu Arduino/Garagino, os três botões (no pino D8 - *backlight,* nos pinos D7 e D10 - *contraste*) controlarão essas características do display. E o seu controle de display LCD está pronto!!! Esperamos que tenha gostado. Qualquer dúvida, poste aqui. Até a próxima!!!

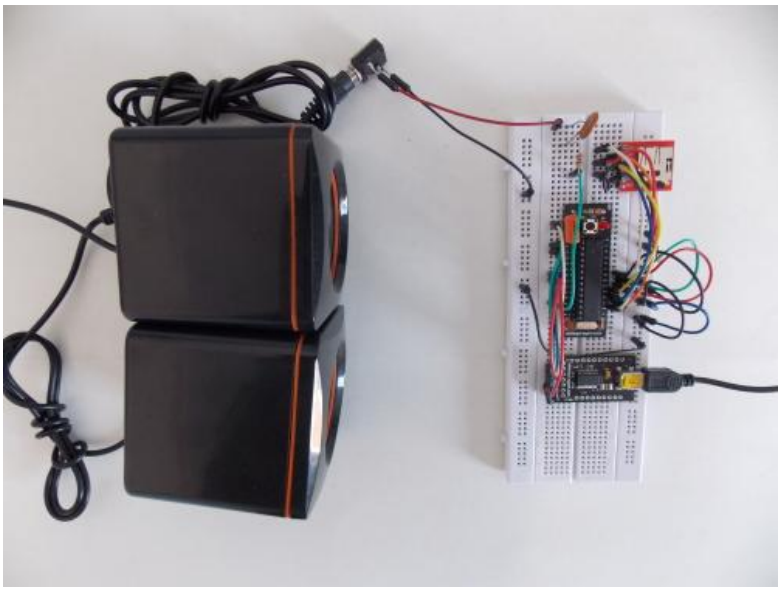
Links de Referência:

- Display de Cristal Líquido

- PWM

56. Tutorial: Música com Garagino ou Arduino (.WAV Player)

Este tutorial mostra como executar um arquivo .wav de um cartão SD com o Garagino e um conversor USB/Serial FTDI, sem a ajuda de nenhum *shield!* O arquivo .wav é uma extensão de arquivo de som das especificações RIFF (*Resource Interchange File Format*) da Microsoft. O arquivo .wav é mais simples de ser executado, uma vez que se trata basicamente de amostras de sons digitalizadas, diferentemente de outros formatos compactados, como o .mp3 por exemplo.



Para isto, serão necessários os seguintes componentes:

- 1 Garagino (ou Arduino)

- 1 Conversor USB/Serial

- 1 resistor de 82 Ohms (pode ser também 100 Ohms)

- 2 capacitores cerâmicos de 100 nF

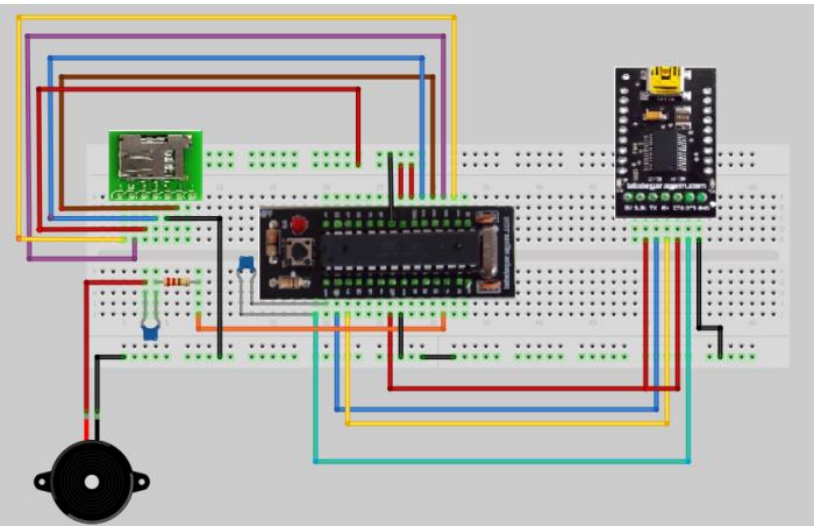
- Cartão SD

- Jumpers para conexão

O cartão SD recebe um valor de tensão máxima igual a 3,3V nos seus terminais, sendo que nas saídas padrão do Arduino, há 5V. Por isso optou-se pelo Garagino Proto, que pode operar

de 1,8V a 5,5V, dispensando a necessidade de divisores resistivos ou conversores de nível lógico para conexão ao cartão SD. O arquivo é lido byte a byte do cartão, e os valores obtidos são colocados em uma das portas PWM do Garagino, que conectados a um alto-falante ou amplificador, produzem o som.

A seguir, monte o circuito como mostrado abaixo (clique na imagem para ampliar):



No circuito acima, um dos capacitores tem a função de "*Reset",* e o outro de desacoplar o nível DC da saída para o alto falante. O resistor de 100 ou 82 Ohms aumenta a impedância da porta do microcontrolador e limita sua corrente para que não seja danificada.

Agora, faça o download de uma de nossas músicas preferidas (".wav") pelo link abaixo:

http://www.4shared.com/music/o6XFq5zY/test.html

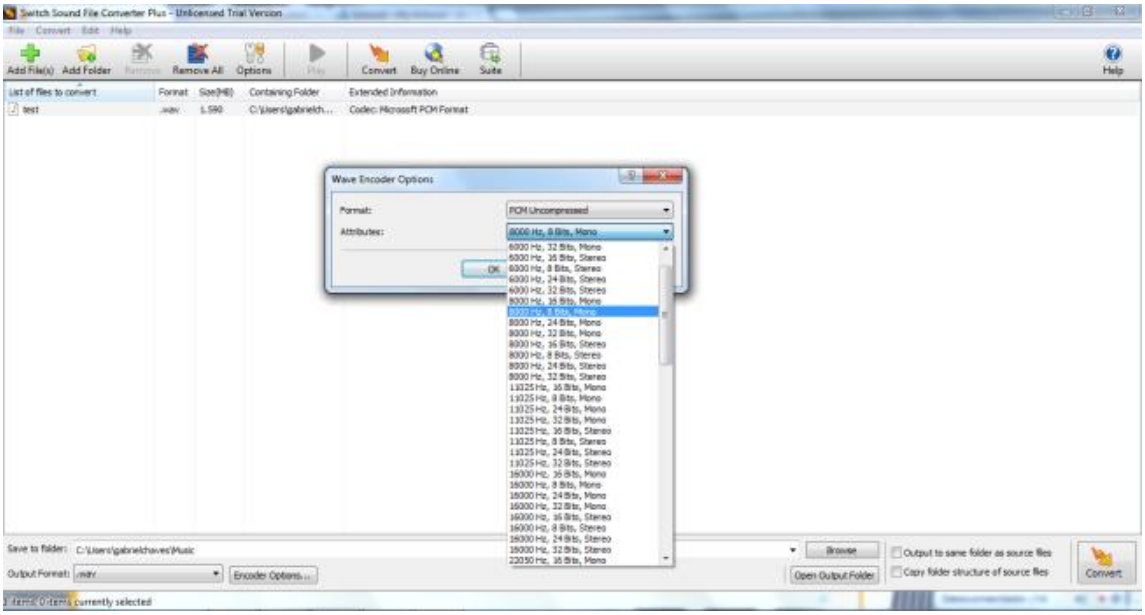
Está com o nome de "test.wav". Não modifique este nome por enquanto !!! Salve o arquivo no seu cartão micro SD e depois coloque-o no seu *breakout*. Abra a IDE do Arduino e digite o

seguinte código:

```
// Firmware para leitura de arquivo .wav do cartão SD e reprodução em alto falante.
#include <SdFat.h>    // Inclui a biblioteca "Sdfat.h" para leitura de cartão SD. Se você não tiver a biblioteca, faça o download e extraia na pasta "libraries" do Arduino
#define sound 6    // Pino 6 com PWM, definido como "sound" (para saída analógica).
const int chipSelect = 10;    // Pino 10 como "chip select" do cartão SD.
SdFat sd;    // Definição para uso da biblioteca.
SdFile myFile;    // Definição para uso da biblioteca.
void setup() {

Serial.begin(9600);    // Inicia a comunicação Serial.
pinMode(sound, OUTPUT);    // Define o pino 6 como saída.
TCCR0B = 0x01;    // Configura a frequência do PWM nos pinos 5 e 6 para 64KHz.
if (!sd.init(SPI_FULL_SPEED, chipSelect)) sd.initErrorHalt();    // Inicia SdFat com máxima velocidade ou imprime mensagem de erro no "Serial Monitor".
if (!myFile.open("test.wav", O_READ)) {    // Abre o arquivo chamado "test.wav" (o nome pode ser mudado), se não, imprime mensagem de erro no "Serial Monitor".
sd.errorHalt("opening test.wav for read failed");
}

unsigned char data;    // Declara uma variável para armazenar os dados do arquivo.
for(int count=0; count<46; count++) {    // Pula o cabeçalho do arquivo test.wav para acessar os dados(depois de 129 leituras). Este número pode variar para cada arquivo.
data = myFile.read();    // Lê byte a byte até o final do cabeçalho.
}

while (data >=0) {    // Se o dado a ser lido for maior ou igual a zero (não nulo),

data = myFile.read();    // Lê um byte do arquivo e armazena-o em "data".
analogWrite(sound, data);    // Envia os dados para o pino 6 (pino ligado ao alto falante).
delayMicroseconds(40);    // Espera um certo intervalo (em microssegundos) para a próxima amostragem (de acordo com a do arquivo .wav).
// Este tempo de espera faz com que as amostragens sejam mais próximas à frequência de 8K Hz do arquivo.
}

myFile.close();    // Fecha o arquivo.
}
void loop() {
// Nada acontece aqui.
}
```

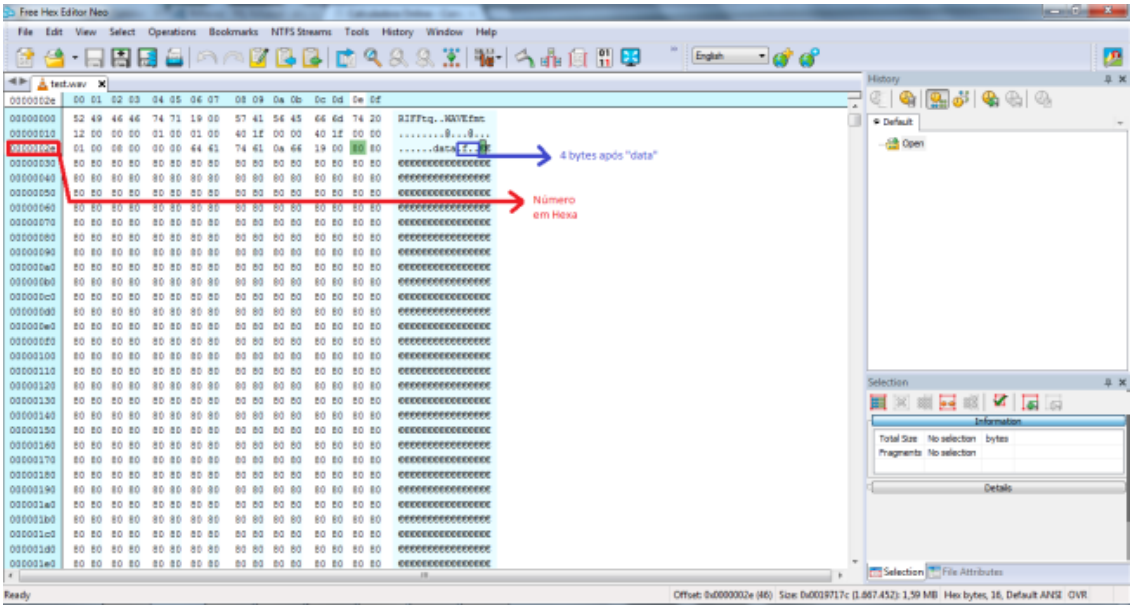
Agora faça o upload para o seu Garagino. Se tudo correr bem, suas caixas de som reproduzirão a música ".wav". E o seu tocador de arquivos ".wav" sem *shield* está pronto !!! Esperamos que tenha gostado. Qualquer dúvida, poste no blog.

Obs: Se quiser executar qualquer outro arquivo ".wav", salve-o no seu cartão SD, vá até a linha marcada em *azul* no código, e altere o nome "test" pelo nome do seu arquivo. Então, abra ele usando um conversor de arquivos de áudio (este ou outro), converta-o para uma taxa de amostragem de 8KHz e 8 bits por amostra, mono:



O valor de 8 bits por amostra foi adotado porque uma porta de saída PWM deste microcontrolador tem até 10 bits de resolução. Porém, como nos conversores geralmente não se encontra este valor, escolheu-se 8 bits. Os 8KHz de frequência permite o microcontrolador processar com mais facilidade as amostras de som (considerando atrasos por ciclos de máquina, varredura, etc).

Agora, execute o *arquivo convertido* com um editor hexadecimal. Vá então à parte escrita do arquivo 4 bytes após "data" e observe o número em hexadecimal correspondente. Veja o exemplo abaixo:



Converta este número (no caso 2E) para decimal. No exemplo, será 46 em decimal. Vá então à linha marcada em *vermelho* no código e modifique o número "46" para o do seu arquivo, esta é a posição onde começam os dados, após o cabeçalho do arquivo. Feche o editor hexa. Faça então o upload para o seu Garagino !!!

Links de Referência:

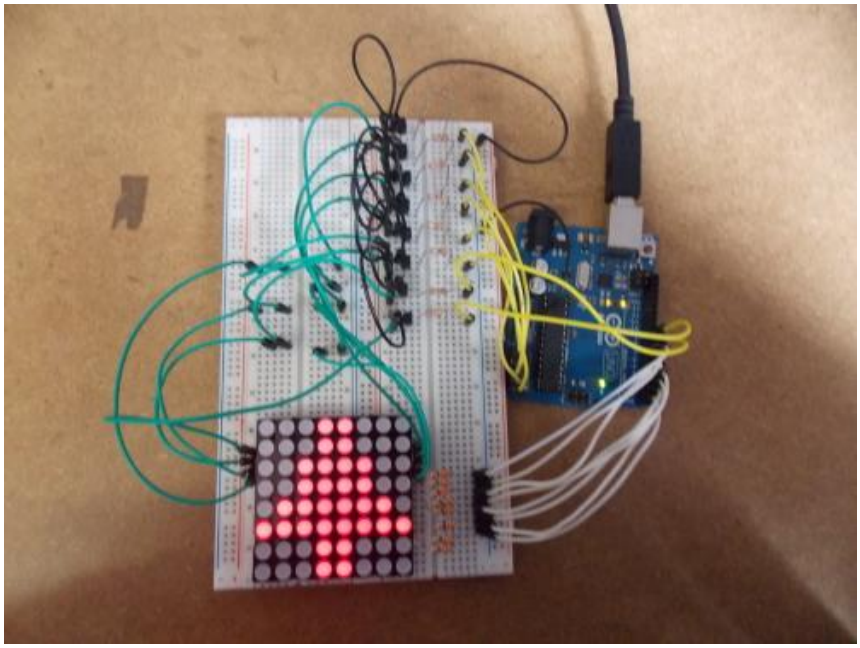
- Garagino Proto

- Arquivos ".wav"

- PWM

57. Letreiro de natal – Matriz 8x8 com mensagem de natal

Aproveitando que o clima Natalino que esta chegando, nós do Laboratório de Garagem resolvemos produzir um tutorial para fazer um letreiro com componentes comuns e facilmente encontrados em qualquer lugar para desejarmos a todos os garagistas um Feliz Natal.



Lista de materiais:

1 x Arduino Uno R3 original

1 x Protoboard Grande.

8 x Resistores de 330Ω para limitar a corrente do Anodo dos LEDs.

8 x Resistores de 1KΩ para a chavear a base dos transistores.

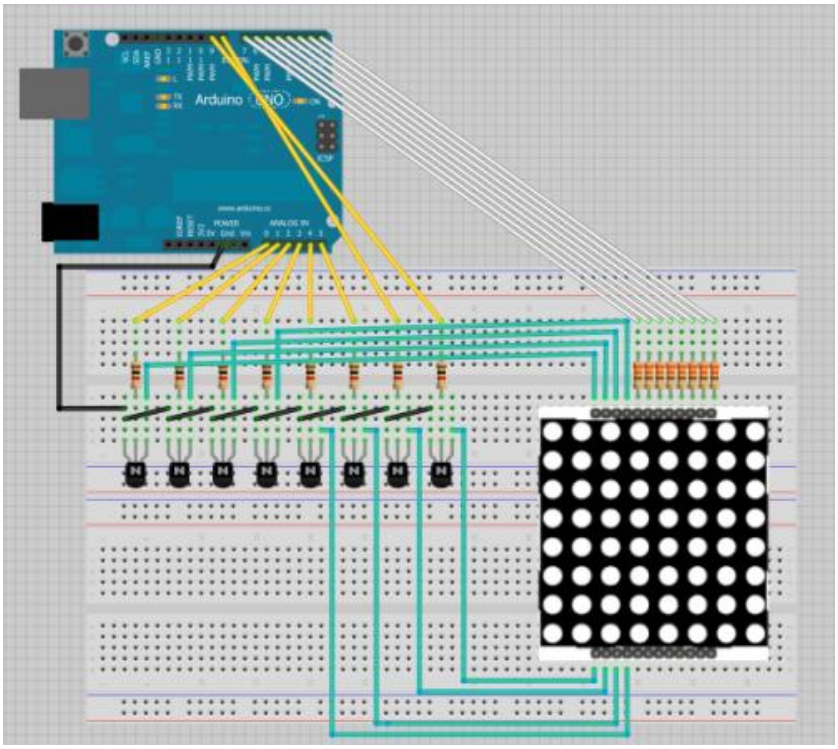
8 x Transistores BC337 (pode ser BC547 ou qualquer NPN compatível).

1 x Matriz de LED 2 cores 8x8 (pode ser unicolor também)

Alguns jumpers.

Os itens podem ser adquiridos na loja do Lab. de Garagem.

Realizamos a montagem do circuito como no esquema abaixo:



O PORTD (pinos digitais de 0 a 7) foram ligados nos resistores de 330Ω que vão aos Anodos dos Diodos Vermelhos(Linhas).

O PORTC (pinos de A0 até A5) e o PORTB (pinos 8 e 9) foram conectados as resistências de 1KΩ das bases dos transistores que tem no coletor os Catodos LEDs e os emissores vão para GND

Funcionamento:

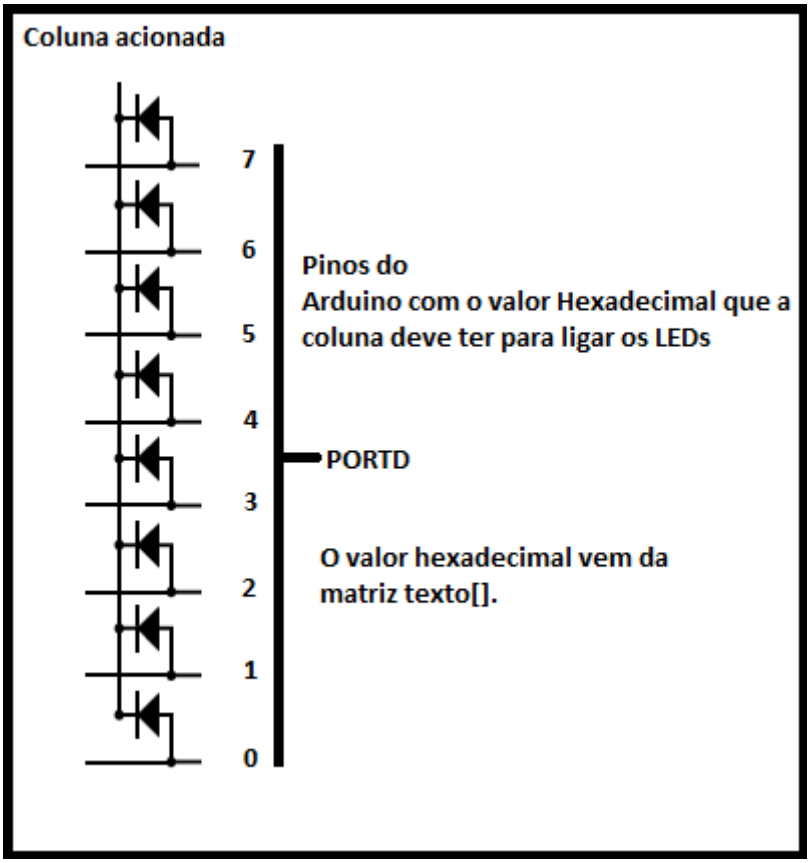
Para o acionamento da matriz, fizemos uma varredura das colunas que são acionadas uma de cada vez e em alta velocidade para que crie um fenômeno conhecido como Persistência Visual que em inglês é conhecido como POV (Persistence Of Vision).

Este efeito gera uma falsa imagem no cérebro humano, fazendo com que pareça com que a figura estática esteja se movendo. Para isto temos que ter uma atualização dos quadros superior a 24 quadros por segundo.

O passo a passo para acionamento das colunas é o seguinte:

A 1ª coluna é acionada através dos transistores que levam os catodos ao GND.

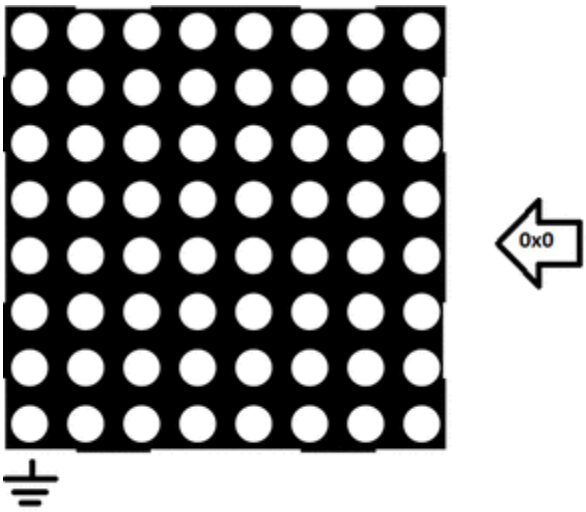
Buscamos o dado na matriz texto[] (que estão em Hexadecimal) com o ponteiro + número de coluna e enviamos para o PORTD ligando os LEDs.



Aguardamos um tempo para os LED's poderem acender.

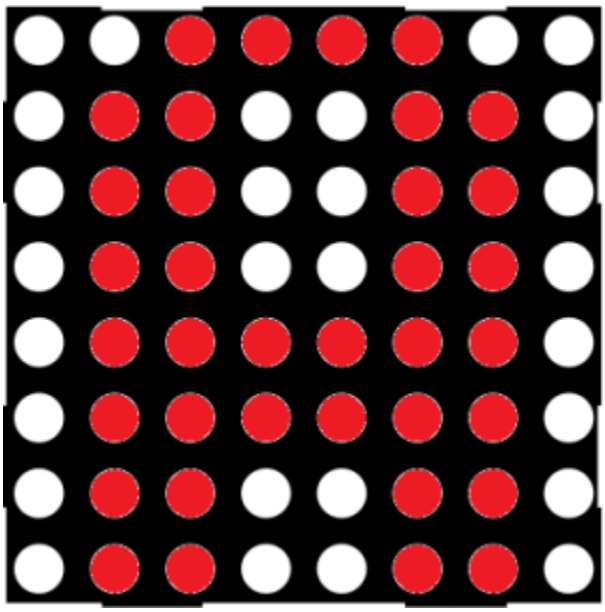
Apagamos os todos e acionamos a próxima coluna repetindo estes passos 8 vezes e na 8ª coluna e com isto acabamos um quadro completo.

No exemplo abaixo montamos a letra A



Repetimos este quadro algumas vezes para controlar a velocidade com que as letras passam no letreiro (no sketch a variável quadros faz este papel).

Você verá a imagem acima deste jeito:



Incrementamos o ponteiro da matriz texto[] e repetimos todos os passos acima para fazer o texto "rolar" pela matriz de LEDs.

A matriz terminará quando alcançar o limite estipulado pelo 1º laço for, recomeçando a passagem das letras.

Upload do Sketch:

Após a montagem e verificação do circuto, copie e cole o sketch abaixo e faça o upload para o Arduino:

```
//
// LETREIRO DE NATAL LAB. DE GARAGEM
//


byte texto[] = {//Matriz onde o texto é colocado para que possa aparecer no display

0x0,  0x0,   0x0,   0x0,   0x0,   0x0,   0x0,   0x0,//Espaço
0x0, 0xFF,  0xFF,  0x3,   0x3,   0x3,   0x3,   0x0,//L
0x0, 0xE,   0x1F,  0x33,  0x33,  0xFF,  0xFF,  0x0,//d
0x0, 0x7E,  0xFF,  0xC3,  0xDB,  0xDF,  0x5E,  0x0,//G
0x0, 0x18,  0x18,  0x18,  0x18,  0x18,  0x18,  0x0,//-
0x0, 0xFF,  0xFF,  0xD8,  0xD8,  0xC0,  0xC0,  0x0,//F
0x0, 0xFF,  0xFF,  0xDB,  0xDB,  0xC3,  0xC3,  0x0,//E
0x0, 0xFF,  0xFF,  0x3,   0x3,   0x3,   0x3,   0x0,//L
0x0, 0x0,   0x0,   0xDF,  0xDF,  0x0,   0x0,   0x0,//I
0x0, 0xC7,  0xCF,  0xDF,  0xFB,  0xF3,  0xE3,  0x0,//Z
0x0,  0x0,   0x0,   0x0,   0x0,   0x0,//Espaço
0x0,   0xFF,  0xE0,  0x70,  0x38,  0x1C,  0xFF,  0x0,//N
0x0, 0x7F,  0xFF,  0x8C,  0x8C,  0xFF,  0x7F,  0x0,//A
0x0, 0xC0,  0xC0,  0xFF,  0xFF,  0xC0,  0xC0,  0x0,//T
0x0, 0x7F,  0xFF,  0x8C,  0x8C,  0xFF,  0x7F,  0x0,//A
0x0, 0xFF,  0xFF,  0x3,   0x3,   0x3,   0x3,   0x0,//L
0x0,  0x0,   0x0,   0x0,   0x0,   0x0,   0x0,   0x0,//Espaço
};


int ponteiro;

int coluna;

int quadros;

void setup()

{

  DDRC = B00011111; //configura os pinos de A0 até A5 como saída digital.

  DDRB = B00000011; //configura os pinos 8 e 9 como saída digital.

  DDRD = B11111111; //configura os pinos de 0 até 7 como saída digital.

}

void loop()

{

  PORTC = B00000000; //Mantem as colunas da matriz desligadas
```

```
  PORTB = B00000000; //Mantem as colunas da matriz desligadas
  for(ponteiro=0; ponteiro < 126;ponteiro++){ //Carrega o ponteiro em zero e limita ele para o tamanho da matriz
    for (quadros =0; quadros < 5;quadros++ ){//Inicia o contador de quadros por passagem
      for(coluna=0; coluna < 8;coluna++) // Inicia Contador das Colunas
      {
        if (coluna<=5)
        {
          PORTB = B00000000;//Deixa pinos 8 e 9 desligados
          PORTC = B00000001 <* coluna;//Seta o bit da coluna que será acessa.
//*********************************************************************************************
// OBS: No lugar do " * " da linha acima, coloque outro sinal de menor que " < "
//*********************************************************************************************
        }
        else
        {
          PORTC = B00000000;//Deixa pinos de A0 a A5 desligados
          PORTB = B00000001 <* coluna-6;//Seta o bit da coluna que será acessa
//*********************************************************************************************
// OBS: No lugar do " * " da linha acima, coloque outro sinal de menor que " < "
//*********************************************************************************************
        }
      PORTD = (texto[ponteiro+coluna]);//Carrega o PORTD(pinos 0 a 7 do Arduino) com o valor indicado pela matriz
      delay(3);// Espera para que o LED Acenda e em seguida apaga os LEDs.
PORTD = 0x00;
      }
    }
  }
}
```

Você também pode modificar o texto escrito alterando os valores em hexadecimal na matriz texto.

Esperamos que tenham gostado deste tutorial!! Boa diversão com este Letreiro para o Natal!!! Caso tenham dúvidas, poste aqui no blog! Para sugestões, clique aqui! Para ver outros tutoriais, clique aqui e projetos abertos, clicando aqui!

Referências:

http://en.wikipedia.org/wiki/Persistence_of_vision

http://www.arduino.cc/en/Reference/PortManipulation

http://www.labdegaragem.org/loja/index.php/32-kits/jumper-wire-kit....

http://www.labdegaragem.org/loja/index.php/matrix-de-led-2-cores-me...

http://www.labdegaragem.org/loja/index.php/protoboard-1890-pinos.html

http://www.labdegaragem.org/loja/index.php/29-arduino/arduino-uno.html

58. Tutorial: Como fazer hinos dos times de futebol de São Paulo com Arduino

Utilizamos o arduino para tocar o Hino dos principais times de futebol de são paulo e demonstraremos como vocês garagistas podem fazer isto utilizando a função tone.

Lista de materiais:

1 x Arduino Uno R3 original

1 x Protoboard

1 x Alto-Falante 8 Ω / 0,25 W

1 x Potenciômetro de 10 KΩ

1 x Capacitor de 100µF / 25 V

Alguns jumpers

Todos os itens podem ser adquiridos na loja do Lab. de Garagem.

Funcionamento:

A Sintaxe da função tone é a seguinte:

tone (pino, frequência, duração);

Onde:

pino - Pino a frequência será gerada para o alto-falante.

frequência - A frequência em Hertz da nota.

duração - Duração em milissegundos da Nota(este é opcional).

Para geramos as notas musicais temos que entender um pouco sobre música.

Cada nota musical nada mais é do que uma frequência que esta dentro da faixa audível do ouvido da maioria de nós que é entre 20Hz a 20.000Hz. Estes valores variam de pessoa para pessoa e com o passar dos anos os tons mais agudos (Frequência próximas ao 20kHz) deixam de ser ouvidas.

A partir disto, temos por padrão que a nota de referência para afinarmos instrumentos musicais é o 1º Lá (ou ainda o Lá da 4ª oitava (ou "A4")), após o Dó central do piano (nota chamada de "C4"), e este tem sua frequência definida para 440Hz.

Na figura 1 abaixo vemos uma partitura com as Claves de Sol e com a Clave de Fá.

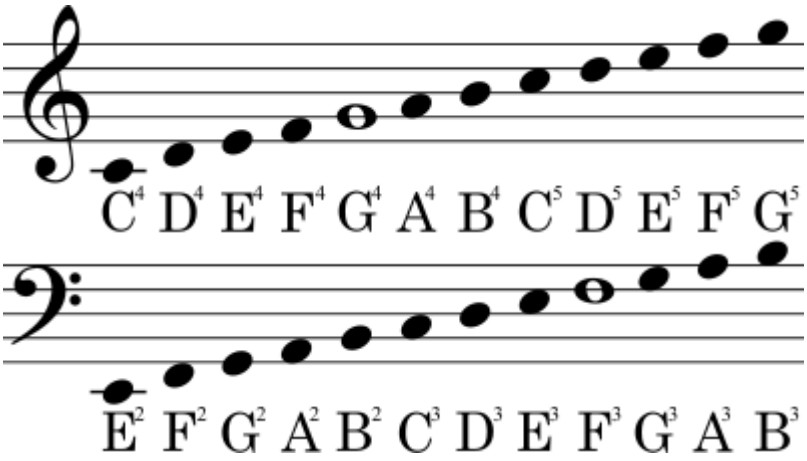


FIgura 1 - Duas Claves de Sol(acima) e de Fá(Abaixo). Nota C4 que é o Dó central do Piano (sua frequência é ≈ 261 Hz).

As demais notas variam de acordo com a tabela abaixo:

Octave → Note ↓	0	1	2	3	4	5	6	7	8	9	10
C	16.352 (−48)	32.703 (−36)	65.406 (−24)	130.81 (−12)	261.63 (±0)	523.25 (+12)	1046.5 (+24)	2093.0 (+36)	4186.0 (+48)	8372.0 (+60)	16744.0 (+72)
C♯/D♭	17.324 (−47)	34.648 (−35)	69.296 (−23)	138.59 (−11)	277.18 (+1)	554.37 (+13)	1108.7 (+25)	2217.5 (+37)	4434.9 (+49)	8869.8 (+61)	17739.7 (+73)
D	18.354 (−46)	36.708 (−34)	73.416 (−22)	146.83 (−10)	293.66 (+2)	587.33 (+14)	1174.7 (+26)	2349.3 (+38)	4698.6 (+50)	9397.3 (+62)	18794.5 (+74)
E♭/D♯	19.445 (−45)	38.891 (−33)	77.782 (−21)	155.56 (−9)	311.13 (+3)	622.25 (+15)	1244.5 (+27)	2489.0 (+39)	4978.0 (+51)	9956.1 (+63)	19912.1 (+75)
E	20.602 (−44)	41.203 (−32)	82.407 (−20)	164.81 (−8)	329.63 (+4)	659.26 (+16)	1318.5 (+28)	2637.0 (+40)	5274.0 (+52)	10548.1 (+64)	21096.2 (+76)
F	21.827 (−43)	43.654 (−31)	87.307 (−19)	174.61 (−7)	349.23 (+5)	698.46 (+17)	1396.9 (+29)	2793.8 (+41)	5587.7 (+53)	11175.3 (+65)	22350.6 (+77)
F♯/G♭	23.125 (−42)	46.249 (−30)	92.499 (−18)	185.00 (−6)	369.99 (+6)	739.99 (+18)	1480.0 (+30)	2960.0 (+42)	5919.9 (+54)	11839.8 (+66)	23679.6 (+78)
G	24.500 (−41)	48.999 (−29)	97.999 (−17)	196.00 (−5)	392.00 (+7)	783.99 (+19)	1568.0 (+31)	3136.0 (+43)	6271.9 (+55)	12543.9 (+67)	25087.7 (+79)
A♭/G♯	25.957 (−40)	51.913 (−28)	103.83 (−16)	207.65 (−4)	415.30 (+8)	830.61 (+20)	1661.2 (+32)	3322.4 (+44)	6644.9 (+56)	13289.8 (+68)	26579.5 (+80)
A	27.500 (−39)	55.000 (−27)	110.00 (−15)	220.00 (−3)	440.00 (+9)	880.00 (+21)	1760.0 (+33)	3520.0 (+45)	7040.0 (+57)	14080.0 (+69)	28160.0 (+81)
B♭/A♯	29.135 (−38)	58.270 (−26)	116.54 (−14)	233.08 (−2)	466.16 (+10)	932.33 (+22)	1864.7 (+34)	3729.3 (+46)	7458.6 (+58)	14917.2 (+70)	29834.5 (+82)
B	30.868 (−37)	61.735 (−25)	123.47 (−13)	246.94 (−1)	493.88 (+11)	987.77 (+23)	1975.5 (+35)	3951.1 (+47)	7902.1 (+59)	15804.3 (+71)	31608.5 (+83)

Fonte: Wikipedia

Baseados nesta tabela, arquivo pitches.h foi criado para o exemplo contido no Arduino chamado Melody, que nada mais é que os valores aproximados das frequências das notas convertido para um nome mais fácil de lembrar.

Por exemplo a frequência 440 é a nota NOTE_A4 é será assim que ela vai ser chamada na matriz melodia[].

A linha abaixo é definida no começo do sketch para ajudar a fazer as pausas durante as músicas.

#define NO_SOUND 0

Como vimos na Figura 1 o que identifica a nota é a altura em que ela se encontra na partitura, já a duração depende do símbolo com o qual a nota é desenhada assim como mostra a figura 2

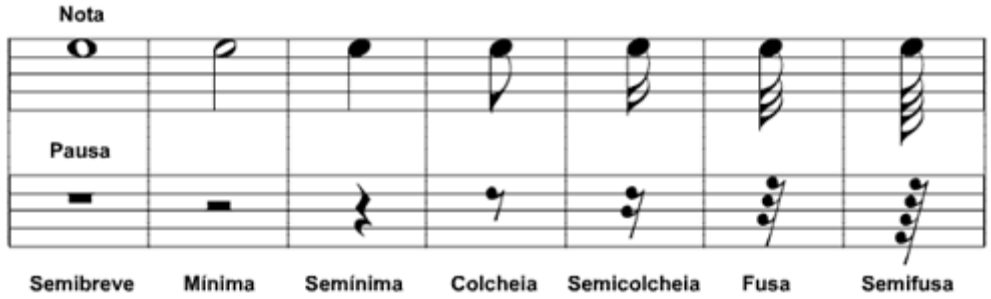


Figura 2 - Duração das Notas (Pentagrama superior) e Duração das Pausas(Pentagrama inferior) e nome dos símbolos(Parte inferior).

A duração de tempo vai ser representada no programa pelos valores colocados dentro da matriz tempoNotas[].

Esta matriz deve receber valores conforme o símbolo contido na partitura conforme mostra a tabela abaixo.

Símbolo na partitura	**Valor de tempo de duração da Nota no Programa**
Semibreve	*1*
Mínima	*2*
Semínima	*4*
Colcheia	*8*
Semicolcheia	*16*
Fusa	*32*
Semifusa	*64*

As duas matrizes trabalham em conjunto e devem receber a mesma quantidade de posição, pois para cada nota ou pausa deve haver seu respectivo tempo de duração e ambos devem ser colocados consecutivamente.

O potenciômetro serve para ajustar o volume e o capacitor servirá para fazer o acoplamento entre o arduino e o alto-falante.

Montagem:

Na figura 3 vocês podem ver a montagem proposta para funcionamento do circuito:

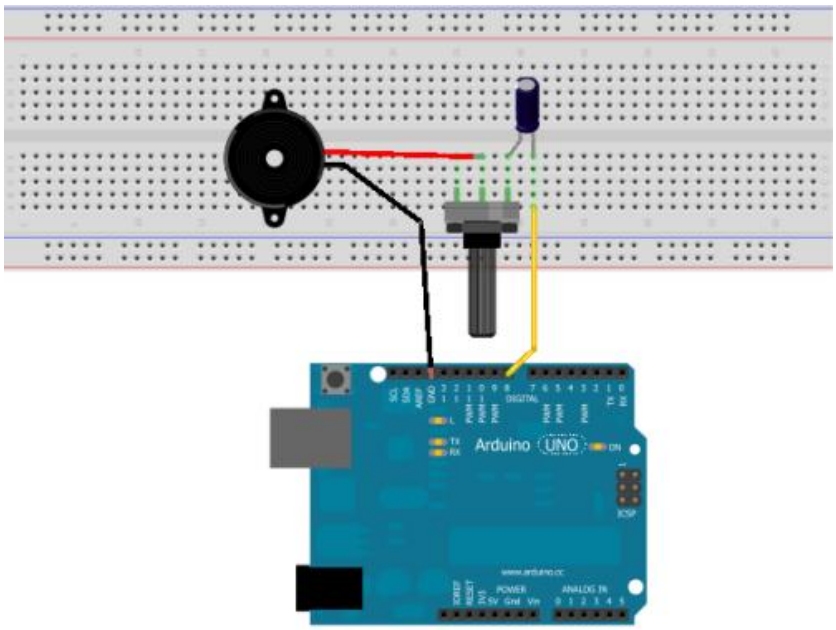
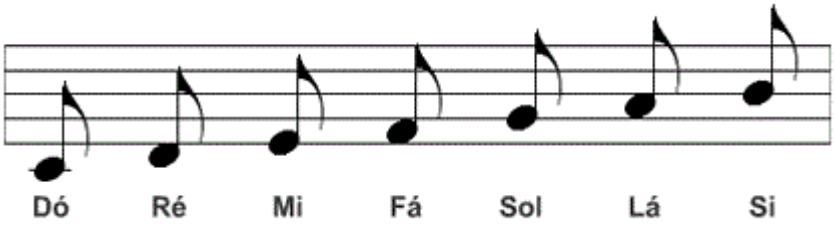


Figura 3 - Montagem proposta.

Os Sketchs

Você necessitará ter o arquivo pitches.h na mesma pasta do sketch que for utilizar.

A estrutura básica do sketch de todas as músicas será igual e ele está todo comentado para ajudar no entendimento.



No código de exemplo abaixo fizemos a tradução da partitura a cima (estando ela na Clave de Sol).

```
#include "pitches.h"
#define NO_SOUND 0

// Notas que devem ser tocadas ordenadamente;

int melodia[] ={
  NOTE_C4,NOTE_D4,NOTE_E4,NOTE_F4,NOTE_G4,NOTE_A4,NOTE_B4
};
// Duração das Notas: Colcheia:8; Semínima: 4; Mínima:2; Semibreve:1
int tempoNotas[] ={
```

```
  8,8,8,8,8,8,8
};

const int compasso = 1450; // Altera o compasso da música
void setup(){
 for (int Nota = 0; Nota <7; Nota++){//o número 7 indica quantas notas tem a nossa matriz.
   int tempo = compasso/tempoNotas[Nota]; //Tempo = compasso dividido pela indicação da
matriz tempoNotas.
   tone(8, melodia[Nota],tempo); //Toca a nota indicada pela matriz melodia durante o tempo.
   // Para distinguir as notas adicionamos um tempo entre elas (tempo da nota + 20%).
   delay(tempo*1.2);
 }
}
void loop(){
 //Não é necessária a repetição pois a mesma será feita pelo botão Reset.
}
//Fim de Programa
```

Agora o que todos esperavam!!! Os hinos dos times estão zipados e com o arquivo pitches.h na mesma pasta:

Hino do Corinthians

Hino do Palmeira

Hino do Santos

Hino do São Paulo

Então é isto garagistas esperamos que gostem e o resultado vocês podem ver no vídeo do inicio deste post. Você pode colocar outras músicas, mas fique atento para os tempos de duração mais complexos e efeitos que não podem ser reproduzidos pelo software. Para mais sobre como ler partituras acesse este site.

Referências

http://en.wikipedia.org/wiki/Pitch_(music)#Pitch_and_frequency

http://en.wikipedia.org/wiki/Scientific_pitch_notation

http://www.cifraclub.com.br

http://www.sotutorial.com/index.php/tutoriais-teorial-musical/

http://pt.wikipedia.org/wiki/Som

59. Automação de lâmpada sem fio com Garabee e Garagino

Neste tutorial mostramos como controlar o acionamento de uma lâmpada através de um controle sem fio montado com um Garagino e dois Garabees que trocam informações para fazer o controle do Módulo Relé, que esta escondido dentro da caixa elétrica junto a um interruptor.

Na figura 1 temos os dois circuitos utilizados:

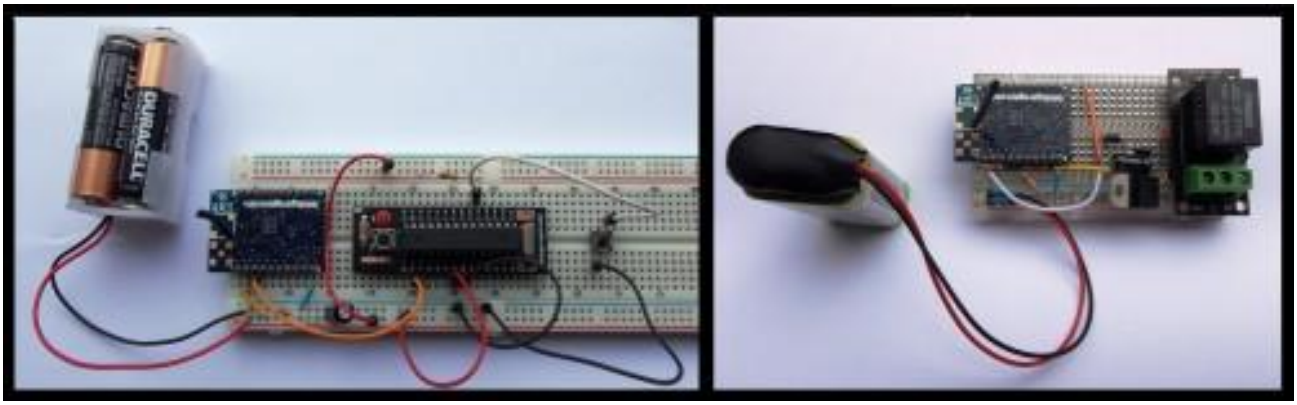


Figura 1 - Transmisor (à esquerda) e o receptor (à direita).

Lista de materiais:

1 x Kit Garagino Básico (Garagino + Conversor USB/Serial)

1 x Kit Garabee (1 Garabee USB + 2 Garabee)

1 x Módulo Relé

1 x Protoboard Pequena (que caiba dentro da caixa elétrica)

1 x Protoboard (para montagem do controle remoto)

1 x Regulador 7805

2 x Diodos 1N4007

2 x Capacitor de 100µF / 25 V

4 x Resistores de 10KΩ

2 x Resistores de 120Ω

2 x LED de 3mm

1 x Jumper Wire Kit

1 x Interruptor Paralelo

1 x Bateria 9 volts

2 x Pilhas

Alguns jumpers

A maior parte dos itens podem ser adquiridos na loja do LdG.

Funcionamento:

Utilizaremos um Garabee para controlar o Módulo Relé e verificar se o interruptor foi acionado. Já o outro Garabee ficará sendo monitorado pelo Garagino para que este possa saber o estado atual da lâmpada e também para comanda-la à distância.

Antes de mais nada, vamos programar os Garabees para fazerem uma comunicação ponto a ponto como o descrito no Tutorial: Como utilizar o Garabee com Arduino - Parte 2.

Instale o Garabee USB com o Driver disponível neste link e utilize um programa Serial de sua preferência (TeraTerm, Serial Monitor, etc...) com as seguintes configurações:

- Baud rate = 19200

- Sem paridade

- Data Bits = 8

- Stop Bits =1

- Carriage return

Programamos o Garabee 1 que ficará junto ao relé na caixa elétrica:

+++ <Enter> - Para entrar em modo de comando.

ATDA 2 <Enter> - Configura o Endereço de Destino (endereço do módulo que enviará comandos).

ATSA 1 <Enter> - Configura endereço deste módulo.

ATCH 20 <Enter> - Configura canal de operação para 20.

ATID 1 <Enter> - Configura o Endereço da Rede.

ATDOI 0 <Enter> - Configura todos os pinos do módulo para iniciarem em nível lógico 0.

ATCT8 1 <Enter> - Configura I/O 8 como Entrada Local (para leitura do estado do relé).

ATCT9 0 <Enter> - Configura I/O 9 como Saída Local para acionar o relé.

ATBD 0 <Enter> - Configura Baud rate para 2400bps.

ATWR <Enter> - Salva as informações passadas para o módulo.

ATCN <Enter> - Comando para Sair do Modo de Comando.

Agora vamos configurar o Garabee 2 que servirá de controle junto com o Garagino. Configure o Software monitor de

com as mesmas especificações descritas acima.

+++ <Enter> - Para entrar em modo de comando.

ATDA 1 <Enter> - Configura o Endereço de Destino (endereço do módulo controla o relé).

ATSA 2 <Enter> - Configura endereço deste módulo.

ATCH 20 <Enter> - Configura canal de operação para 20.

ATID 1 <Enter> - Configura o Endereço da Rede.

ATDOI 0 <Enter> - Configura todos os pinos do módulo para iniciarem em nível lógico 0.

ATBD 0 <Enter> - Configura Baud rate para 2400bps.

ATWR <Enter> - Salva as informações passadas para o módulo.

ATCN <Enter> - Sair do Modo de Comando.

Com os Garabees programados vamos à montagem dos circuitos.

Montagem dos Circuitos

1º O Receptor

Na figura 2 podemos ver o esquema elétrico da ligação entre o módulo Garabee e o Módulo Relé:

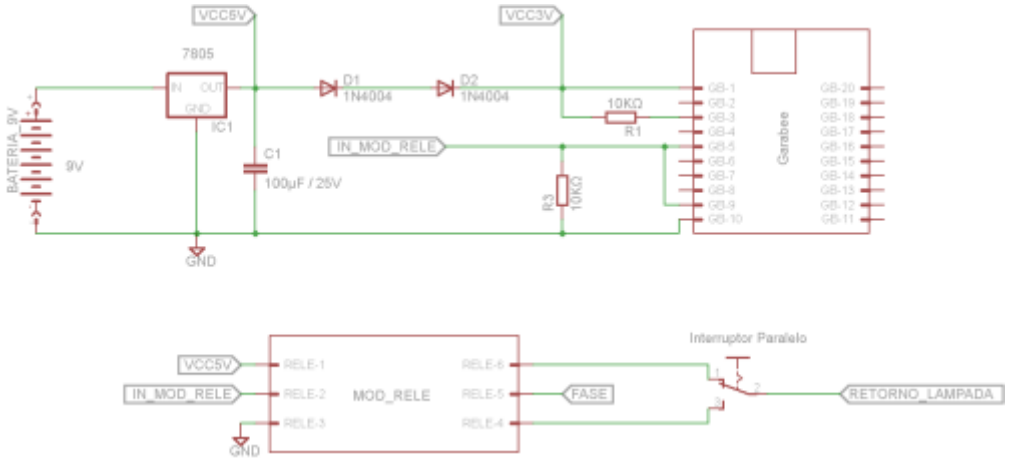


Figura 2 - Esquema elétrico do receptor

A ligação feita entre o relé e o interruptor paralelo, mostrado na figura acima, garante o **duplo acionamento** que pode ser utilizado na lâmpada. O relé esta sendo usado como um outro interruptor paralelo (conforme figura 3) e é assim que o duplo acionamento é garantido.

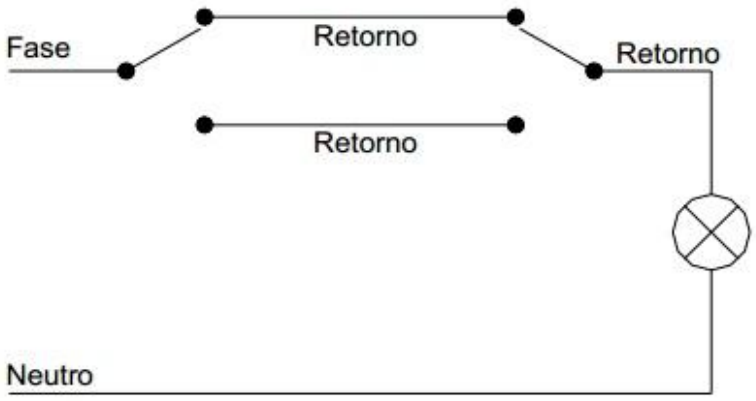


Figura 3 - Esquema elétrico para ligação de interruptor em paralelo. O relé faz o papel de uma das chaves

Na figura 3, temos o detalhe do isolamento feito na parte inferior do Módulo Relé com fita isolante (à esquerda) e a foto do circuito montado (à direita) :

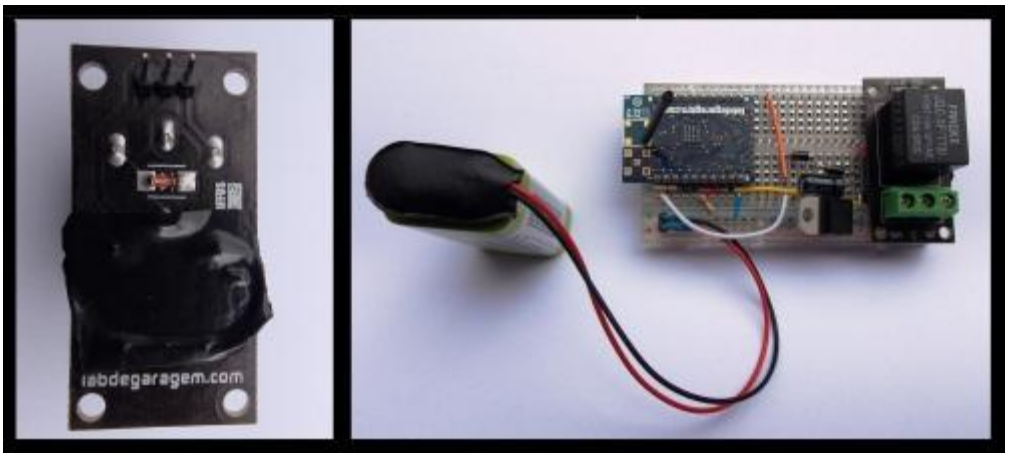


Figura 3 - Detalhe de isolamento do Módulo Relé e foto do circuito montado

Para que o circuito coubesse na caixa onde fica o interruptor, retiramos uma das laterais da protoboard.

2º O Controle Remoto

Primeiro gravamos o sketch no Garagino através do procedimento do tutorial Como utilizar o conversor USB/Serial:

```
#define botao 13 // Define o botão de Leitura na entrada 13
int Status= 0; // Variável Status guarda o estado anterior do relé
void setup()
{
  Serial.begin(2400);//2400 8 bits sem paridade e 1 stop bit.
  delay(4000);//Espera o módulo inicializar.
  Serial.println("+++");//+++<cr> Para entrar em modo de comando do Garabee
  pinMode(botao, INPUT); // Configura o pino 13 como entrada digital.
}
void loop()
{
  while (digitalRead(botao) == 1)
  {
  }//para deboucing
  while (digitalRead(botao) == 0)
```

```
  {
  }
  Status = Status ^1; //inverte o estado do Flag status
  if (Status ==0 )
  {
    Serial.println("ATWRO 0");//envia comando para desacionar rele
  }
  else
  {
    Serial.println("ATWRO 256");//envia comando para acionar o rele
  }
}
```

```
//Fim do Sketch
```

Com o Garagino já gravado montamos o controle remoto que vai enviar comandos para inverter o estado da lâmpada.

Na figura 5 podemos ver o esquema elétrico para a montagem e na figura 6 temos a foto de como ela ficou na protoboard:

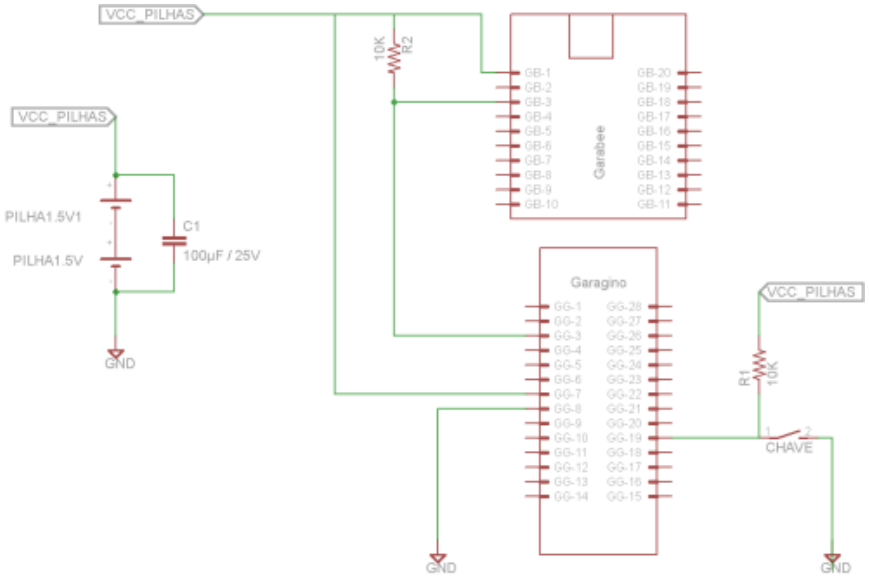


Figura 5 - Esquema elétrico para montagem controle remoto

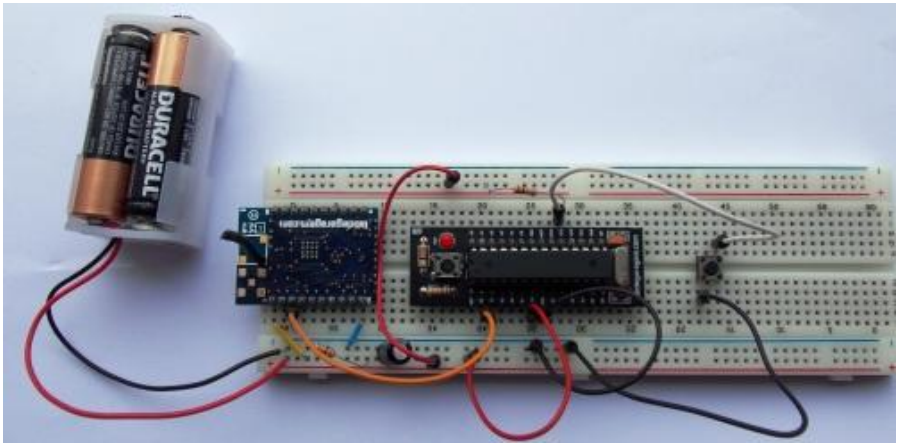


Figura 6 - Montagem na protoboard

Com a montagem concluída efetuamos os testes, e para isto, pode-se ligar um LED em série com um resistor de 120Ω entre o pino 20 e o terra nos dois Garabees.

Alimente ambos os circuitos, e se ambos estiverem na mesma rede, os LEDs irão acender (você pode remover os LEDs após os teste).

Montagem do circuito na caixa elétrica

ATENÇÃO - Certifique-se de desligar o circuito que estiver utilizando no quadro do distribuição do local **antes** de mexer nos cabos ligados a rede elétrica, você pode sofrer um grave acidente, tome cuidado. Só volte a ligar o circuito após tudo estar conectado.

Se não tiver experiência com circuito elétricos, faça este tutorial com muita cautela, ou chame alguém com experiência na área para ajuda-lo.

No lugar da bateria 9 volts poderia ser utilizada uma fonte chaveada ou ainda uma fonte sem transformador (alguns chamam estas fontes de "rabo quente") . Como este não é intuito deste tutorial, nós iremos alimentar os circuitos com 3 volts e 9 volts.

Segue algumas fotos da montagem de que dever ser feita entre o Módulo Relé, o interruptor paralelo, a fase e o retorno da lâmpada.

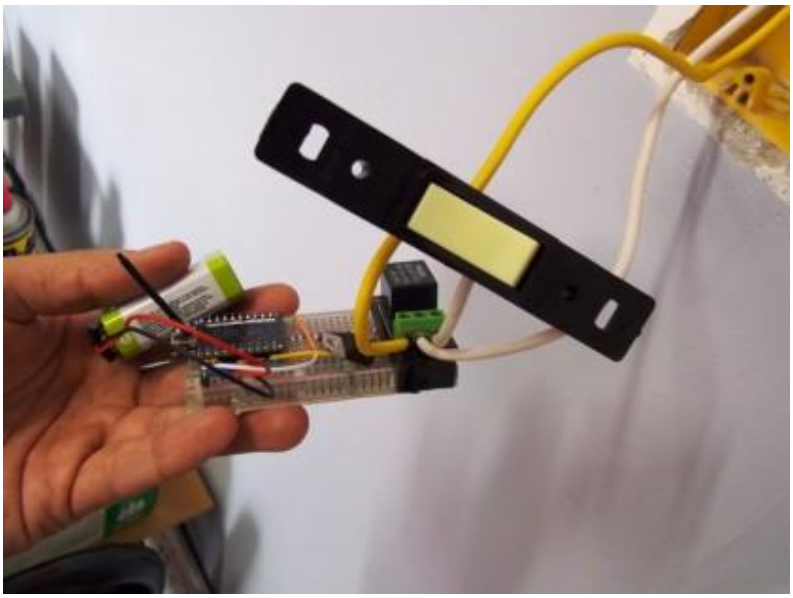


Figura 7 - Circuito montado e conectado ao interruptor paralelo e a fase e ao retorno da lâmpada

Figura 8 - Ufa!!! Esta tudo lá dentro só falta o espelho de acabamento.

Deu um pouco de trabalho para colocar o circuito lá dentro, mas com um pouco de paciência e jeitinho, tudo é possível.

Utilizamos neste tutorial uma protoboard para fazer o duplo acionamento da lâmpada, tendo em vista que ele foi apenas para o aprendizado, ***não recomendamos*** que utilizem a protoboard como forma definitiva de controlar a lâmpada dentro da caixa elétrica.

Esperamos que tenham gostado e em caso de dúvida, sugestões ou problemas enviem comentários...

\o/

Referências:

http://labdegaragem.com/profiles/blogs/tutorial-como-utilizar-o-gar...

http://labdegaragem.com/profiles/blogs/tutorial-como-utilizar-o-gar...

http://aragao.spo.ifsp.edu.br/

60. Tutorial: Robô de Papel - Faça o seu Garabot Controlado por IR

Hoje iremos ensinar como fazer o Garabot, um robô com uma face de papel e alguns componentes eletrônicos. Ele será comandado por um Arduino através de um controle IR.

Lista de materiais:

1 x Arduino Uno R3 original ou Garagino

1 x Protoboard

1 x Resistor de 120Ω

2 x Servomotores Pequenos

1 x IR Control Kit

1 x Cola bastão

1 x Fita dupla-face esponjosa

1 x Tesoura

1 x A4 paper (da cor que desejar)

Alguns jumpers

Funcionamento

Comandos IR

O IR ou Infravermelho é uma frequência que esta fora do espectro visível para o olho humano como mostra a *figura 1* (o comprimentos de onda visível esta compreendido entre 400 a 750nm). E é por isto que não enxergamos o LED do controle piscando.

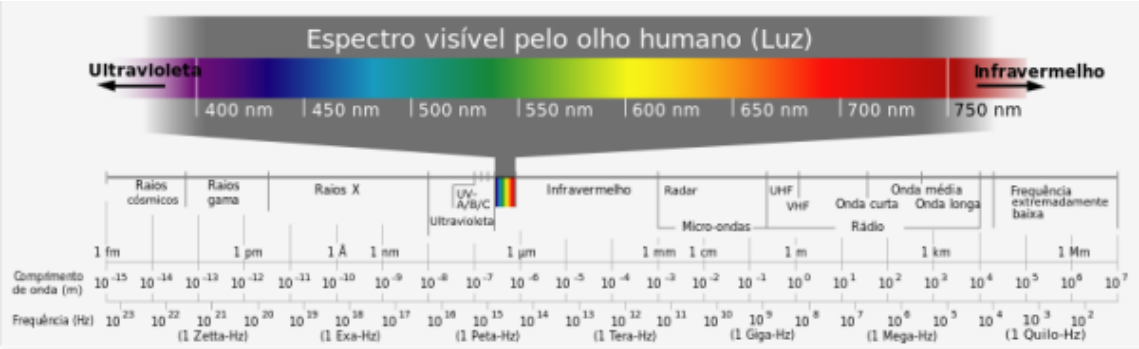


*Figura 1 - Espectro visível (*em destaque na parte superior) e outros espectros com seus respectivos usos e características(abaixo)

Os comandos enviados nada mais são que trens de pulso onde o tempo de duração determina se o valor dele é 0 ou 1. O Arduino através do receptor IR, identifica e transforma estes pulsos em um dado que é depois transformando em um valor inteiro.

Baseado no exemplo que acompanha IR Control Kit, conseguimos identificar com o Arduino as teclas pressionadas no controle IR. Com as teclas identificadas podemos delegar para cada uma delas uma função para que o Garabot execute. A *figura 2 mostra* o que cada tecla fará com o Garabot.

Figura 2 - Controle com os comandos

Servomotores

O servomotor é um componente eletromecânico que consistem de um motor DC controlado por um circuito de controle que recebe um sinal PPM (Pulse Position Modulation) que controla sua angulação dentro de seu range de atuação que é de 180º. Na *figura 3* você vê como são os típicos sinais de controle de um servo motor.

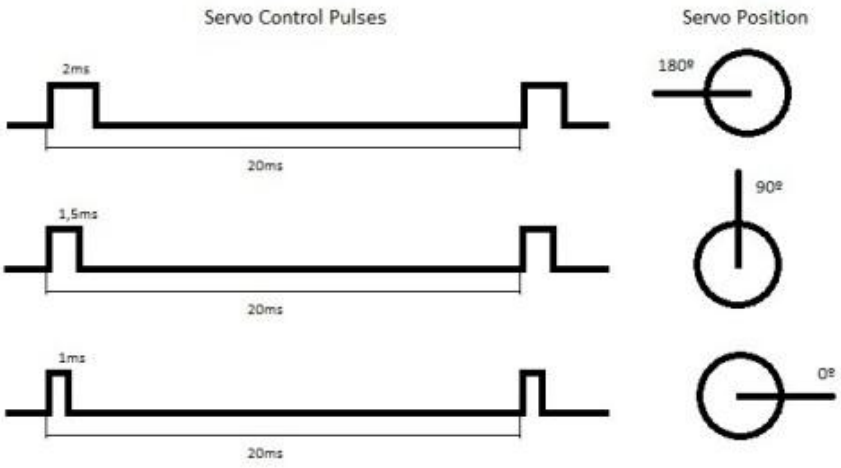


Figura 3 - Características dos sinais de controle do servo motor

Eles normalmente conectados um sistema mecânico de engrenagens conhecido como "*redutor"* que tem o intuito de reduzir a velocidade de saída em relação a de entrada, mas com isto gera um ganho na força de atuação (torque). O sistema mecânico realimenta o circuito de controle por um potenciômetro. Além disto uma das engrenagens tem o papel de delimitar a área de atuação servomotor.

Cada servomotor tem sua especificação, e deve ser utilizado de acordo com a velocidade e carga que o mesmo vai atuar. Fique atento!!!

Depois que a estrutura estiver montada não esqueça de deixar fios longos para não danificar a estrutura de papel.

O Circuito

Na Figura 4 podemos ver o circuito proposta que foi montado*.*

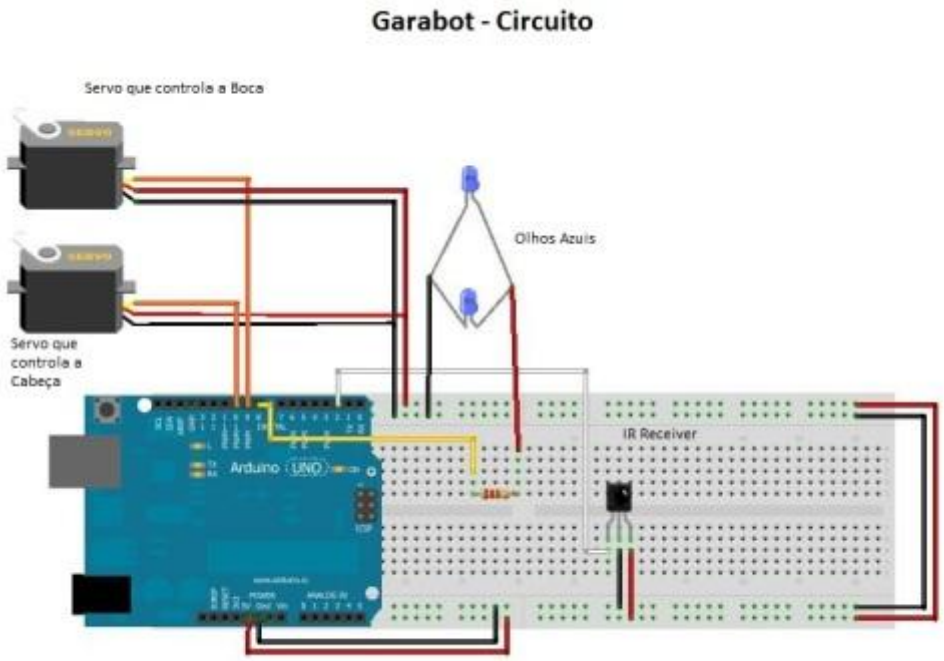


Figura 4 - Montagem proposta para controlar o Garabot

Verifique se o circuito esta corretamente montado e faça o upload do sketch.

O Sketch

```
//Ângulos do Servomotor

#define abre 90

#define fecha 180

#define centro 90

#define esquerda 0

#define direita 180

// Definições dos Pinos

#define olhos 8

#define irPin 2

#include <Servo.h>

Servo servoboca,servocabeca;

int start_bit = 2200; //Start bit threshold (Microseconds)

int bin_1 = 1000; //Binary 1 threshold (Microseconds)

int bin_0 = 400; //Binary 0 threshold (Microseconds)
```

```
boolean powerstatus = false;
boolean statuscentrocabeca = false;
boolean cabecaesq = true;
boolean cabecadir = false;
boolean bocafechada = true;

void setup()
{
  pinMode(olhos, OUTPUT);
  pinMode(irPin, INPUT);

servoboca.attach(9);
  servocabeca.attach(10);

servoboca.write(fecha);
  servocabeca.write (135);
  digitalWrite(olhos, LOW);
}

void loop()
{
  int key = getIRKey(); //Fetch the key
  if(key != 0)
  {
    if (key == 149) //Botão de Power
    {
      powerstatus = powerstatus ^1;
      if(powerstatus == 1)
      {
        digitalWrite(olhos, HIGH);
        servocabeca.write (centro);
        delay(300);
        servoboca.write(abre);
        delay(100);
        servoboca.write (fecha);
        delay(135);
        cabecadir = false;
```

```
        cabecaesq = false;
      }
    }
if (powerstatus == 1)
    {

      switch(key)
      {
case 144: //CH Down - Fecha a Boca
        if (bocafechada == LOW)
        {
          servoboca.write(fecha);
          delay(500);
          bocafechada = HIGH;
          break;
        }
        else //Pisca os olhos se a boca já estiver fechada
        {
          digitalWrite(olhos, LOW);
          delay(150);
          digitalWrite(olhos, HIGH);
          delay(150);
          digitalWrite(olhos, LOW);
          delay(150);
          digitalWrite(olhos, HIGH);
          break;
        }
case 145: //CH Up - Abre a boca
        if (bocafechada == HIGH)
        {
          servoboca.write(abre);
          delay(500);
          bocafechada = LOW;
```

```
      break;
    }
    else //Pisca os olhos se a boca já estiver aberta
    {
      digitalWrite(olhos, LOW);
      delay(150);
      digitalWrite(olhos, HIGH);
      delay(150);
      digitalWrite(olhos, LOW);
      delay(150);
      digitalWrite(olhos, HIGH);
      break;
    }
case 146: //VOL Right - Move a cabeça para direita
    if (cabecadir == false & cabecaesq == false || cabecadir == true & cabecaesq == false)
    {
      servocabeca.write(direita);
      delay(150);
      cabecadir = true;
      break;
    }
    else if (cabecadir == false & cabecaesq == true)
    {
      servocabeca.write(centro);
      delay(150);
      cabecaesq = false;
      break;
    }
case 147: //VOL Left - Move a cabeça para esquerda
    if (cabecadir == false & cabecaesq == false ||cabecadir == false & cabecaesq == true)
    {
      servocabeca.write(esquerda);
      delay(150);
```

```
      cabecaesq = true;
      break;
    }
    else if (cabecadir == true & cabecaesq == false)
    {
      servocabeca.write(centro);
      delay(150);
      cabecadir = false;
      break;
    }
case 148: //Botão Mute - Faz SIM com a cabeça e pisca os olhos
    servoboca.write(abre);
    delay(150);
    digitalWrite(olhos, LOW);
    servoboca.write(fecha);
    delay(150);
    digitalWrite(olhos, HIGH);
    servoboca.write(abre);
    delay(150);
    digitalWrite(olhos, LOW);
    servoboca.write(fecha);
    delay(200);
    digitalWrite(olhos, HIGH);
    break;
case 165: // Botão AV/TV - Faz NÃO com a cabeça e pisca os olhos
    servocabeca.write(45);
    delay(150);
    servocabeca.write(135);
    delay(150);
    digitalWrite(olhos, LOW);
    servocabeca.write(45);
    delay(150);
    digitalWrite(olhos, HIGH);
```

```
        servocabeca.write(135);
        delay(150);
        digitalWrite(olhos, LOW);
        servocabeca.write(centro);
        delay(150);
        digitalWrite(olhos, HIGH);
        cabecadir = false;
        cabecaesq = false;
        break;
      }
    }
    else
    {
      servoboca.write(180);
      delay(150);
      servocabeca.write(135);
      delay(300);
      digitalWrite(olhos, LOW);
    }
  }
}

int getIRKey()
{
  int data[12];
  int i;

while(pulseIn(irPin, LOW) < start_bit); //Wait for a start bit

for(i = 0 ; i < 11 ; i++)
    data[i] = pulseIn(irPin, LOW); //Start measuring bits, I only want low pulses

for(i = 0 ; i < 11 ; i++) //Parse them
  {
    if(data[i] > bin_1) //is it a 1?
      data[i] = 1;
```

```
    else if(data[i] > bin_0) //is it a 0?
      data[i] = 0;
    else
      return -1; //Flag the data as invalid; I don't know what it is! Return -1 on invalid data
  }
int result = 0;
  for(i = 0 ; i < 11 ; i++) //Convert data bits to integer
      if(data[i] == 1) result |= (1<*i); // Atenção - Substitua o * por outro Sinal de menor que
(<)
return result; //Return key number // Limitação do editor usado pelo Site
}
//Fim do sketch
```

Em pouco tempo e com um pouco de imaginação podemos fazer o Garabot para nos divertir. O que esta esperando? Faça o seu também!!!

Referências

http://quarknet.fnal.gov/fnal-uc/quarknet-summer-research/QNET2010/...

http://en.wikipedia.org/wiki/Electromagnetic_spectrum

http://www.sparkfun.com/tutorials/291

http://pt.wikipedia.org/wiki/Espectro_vis%C3%ADvel

61. Tutorial: WebServer para Android Mini PC

Neste tutorial mostraremos como fazer com que seu Android Mini PC funcione como um Web Server e também como hospedar uma página HTML nele, utilizando o aplicativo PAW Server para Android.

O Que é um WebServer?

É um programa de computador, um aplicativo para Android ou até mesmo a própria máquina que fica responsável por aceitar pedidos HTTP de clientes, geralmente os navegadores, e servi-los com respostas HTTP, incluindo opcionalmente dados, que geralmente são páginas web, tais como documentos HTML com objetos embutidos (imagens e etc).

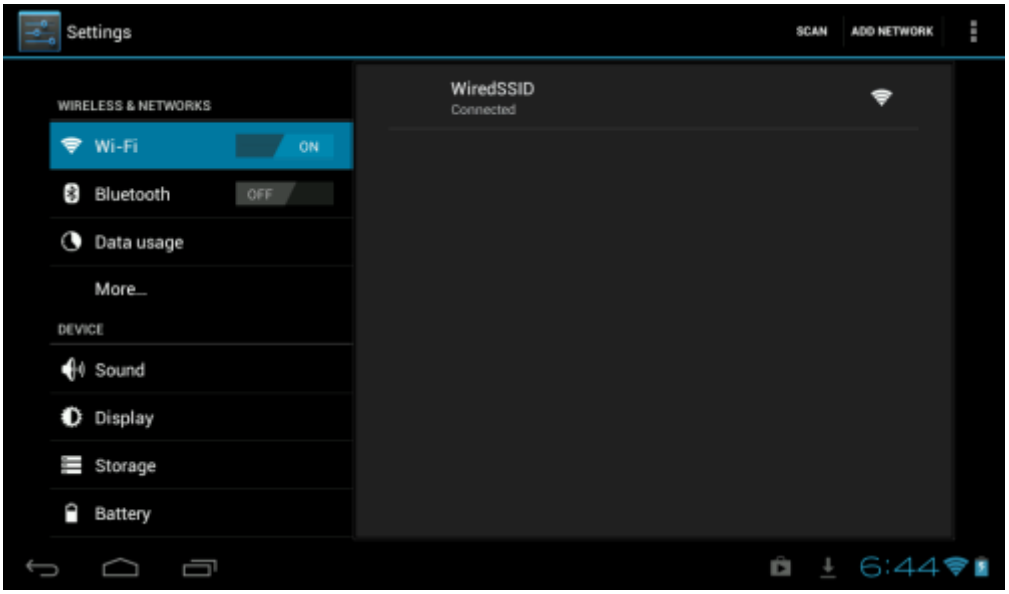
PAW Server

O aplicativo PAW Server permite algo inusitado à primeira vista. Ele faz com que o seu Android se trasforme em um Web Server no qual você pode, a partir de um navegador, executar diversas aplicações, como transferir arquivos e até publicar páginas web em HTML e PHP.

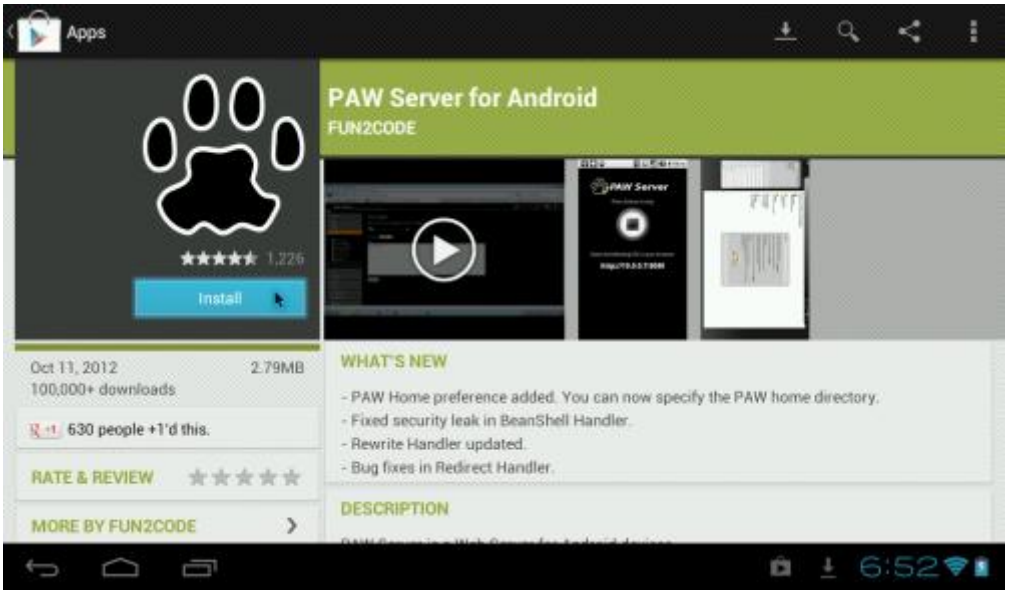
OBS: É necessário uma rede local com acesso a internet, para o download e funcionamento do aplicativo PAW Server.

1. Criando o WebServer

1.1) Conecte seu Android Mini PC à internet via wi-fi



1.2) Execute o aplicativo Play Store e faça o download do aplicativo PAW Server



1.3) Execute o aplicativo PAW Server e clique no botão de Play para que ele crie um Web Server

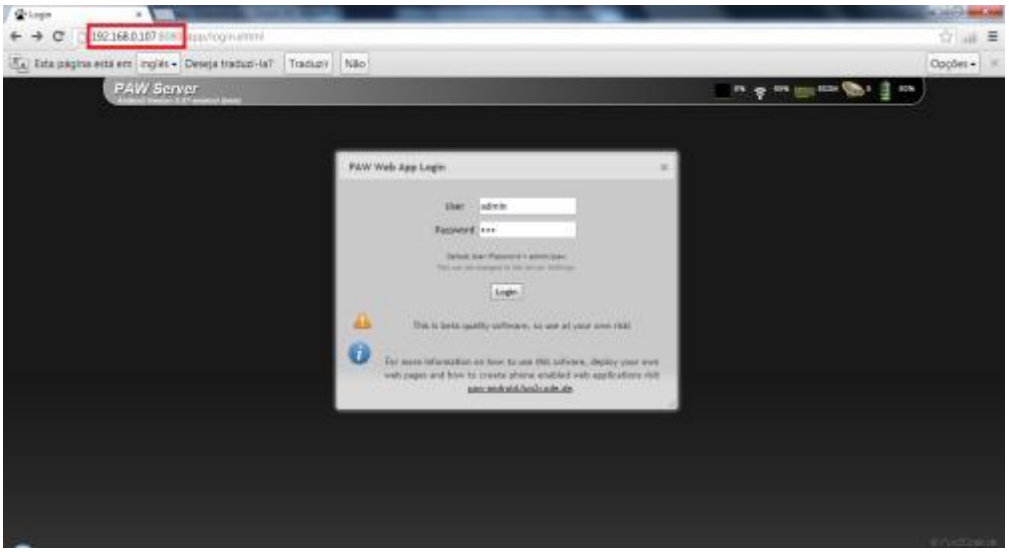
1.4) Ele criará um endereço IP dentro de sua rede local

1.5) Agora você pode acessar o PAW server de um browser de qualquer computador conectado à rede utilizando o IP gerado, no exemplo: "http://192.168.0.107:8080"

1.6) Quando acessá-lo, ele lhe pedirá o login e senha para entrar nas configuração de seu PAW Server, use o login e senha padrão para o primeiro acesso:

Login: admin

Senha: paw



1.7) Essa página permite que você faça diversas configurações de recursos do seu Web Server, tais como:

Gerenciador de arquivos;

MP3 Player;

Enviar emails;

Câmera IP;

Configuração de seu Web Server;

Desenvolvimento em BeanShell;

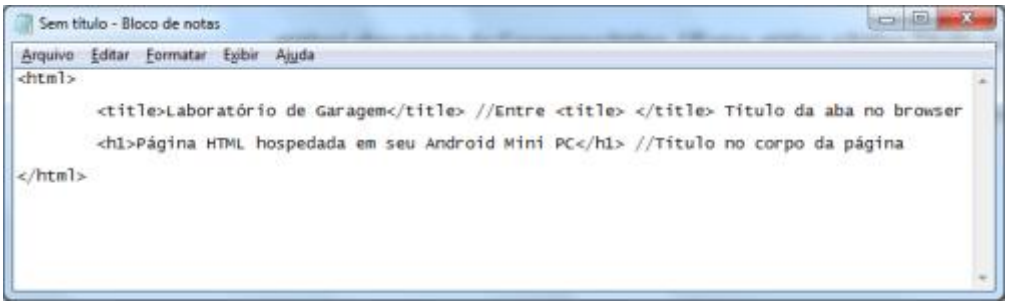
Entre outras.

2. Hospedando Página HTML

Você pode criar um arquivo .html a partir do passo abaixo (2.1), ou fazer o download da página de exemplo doLaboratório de Garagem e continuar do passo 2.3.

2.1) Abra o bloco de notas no seu computador e cole o código conforme descrito abaixo

```
<html>

    <title>Laboratório de Garagem</title> //Título da aba no navegador

    <h1>Página HTML hospedada em seu Android Mini PC</h1> //Título no corpo da página

</html>
```



2.2) Salve o arquivo como .htm ou .html

2.3) Transfira seu(s) arquivo(s) via cartão micro SD para o Android Mini PC

2.4) Copie e cole dentro do diretório "Local Memory\paw\html\"

2.5) Agora execute a página pelo browser de seu Androi Mini PC ou de qualquer microcomputador conectado à rede, digitando o caminho:

Seu.ip.gerado.pelo.paw:server/nome_do_arquivo.html

Exemplo: 192.168.0.107:8080/ldg.html

Esse aplicativo pode ser utilizado em várias aplicações e futuramente traremos tutoriais integrando o Android Mini PC e o Garagino.

Referências:

http://www.beanshell.org/developer.html

http://www.w3schools.com/html/default.asp

http://www.hardware.com.br/noticias/2010-09/pawserver.html

http://www.oficinadanet.com.br/artigo/servidores/o_que_e_um_servido...

62. Tutorial: Controlando Arduino com o Android Mini PC

Neste tutorial mostraremos como utilizar um aplicativo para fazer com que seu Android Mini PC envie comandos para o Arduino via serial.

Faremos como demonstração de aplicação, um acionamento de lâmpada com um Arduino UNO + Módulo Relé feito pela serial do Android Mini PC, a partir do aplicativo.

CUIDADO: Evite manusear o Módulo Rele enquanto o mesmo estiver conectado à rede elétrica e remova qualquer tipo de objeto condutivo ou inflamável de perto. Você pode causar um acidente.

- Lista de Materiais:

1x Android Mini PC

1x Arduino Uno

1x Módulo Relé

1x Lâmpada 127V com bocal e tomada

1x Protoboard

Alguns Jumpers

- Aplicativo Android:

lablampada.apk

- Lista de Softwares:

Projeto do Aplicativo

Eclipse + Ferramentas Para Desenvolvimento Android

Java JDK

OBS:

- Após o Download do Java JDK, siga as Instruções desse Vídeo de como configurar o seu PC para rodar o java. Esse vídeo é um entre 116 Aulas sobre Java, fornecido pelo Ricardo Lopes Costa da Universidade XTI.

- SDK: Dentro da pasta descompactada do SDK, execute o SDK Manager para atualizar as ferramentas de desenvolvimento.

1. O Aplicativo

O aplicativo foi desenvolvido usando o Eclipse ADT, que é um Eclipse com o plugin Android Development Tools (Ferramentas de Desenvolvimento Android).

Abaixo, esta o código do aplicativo onde oferecemos uma breve explicação do funcionamento do App, que foi feito em linguagem java, lembrando que esse não é todo o projeto:

```
/*Aplicativo criado pelo LDG
para controlar seu Arduino via
Android Mini PC*/


package com.example.lablampada;
```

```
///+++Bibliotecas+++
import android.hardware.usb.UsbManager;
import android.os.Bundle;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.view.View;
import android.view.Menu;
import android.widget.Toast;
import android.widget.ToggleButton;
import jp.ksksue.driver.serial.FTDriver;
///=================

public class MainActivity extends Activity
{

///+++PERMISSÃO PARA ACESSAR SERIAL+++
FTDriver mSerial;
private static final String ACTION_USB_PERMISSION =
"jp.ksksue.tutorial.USB_PERMISSION";
///==================================

///+++CRIADO AUTOMATICAMENTE PELO ADT+++
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
///====================================

///+++INICIANDO SERIAL+++
mSerial = new FTDriver((UsbManager) getSystemService(Context.USB_SERVICE));
PendingIntent permissionIntent = PendingIntent.getBroadcast(this, 0, new
Intent(ACTION_USB_PERMISSION), 0);
```

```
mSerial.setPermissionIntent(permissionIntent);
while (!mSerial.begin(FTDriver.BAUD9600)) //Fica em loop de conexão até o Arduino ser
conectado
{
}
Toast.makeText(this, "Conectado", Toast.LENGTH_SHORT).show(); //Exibe mensagem
"Conectado" quando reconhecer o Arduino
///======================

///+++BOTÃO LIGADO/DESLIGADO+++
final ToggleButton bToggle = (ToggleButton) findViewById(R.id.toggleButton1); //Cria uma
varíavel para controle do botão
bToggle.setOnClickListener(new View.OnClickListener() //Monitora o botão Ligado/Desligado
{
public void onClick(View v)
{
boolean botao; //Cria uma variável boleana
botao = bToggle.isChecked(); //Quando o botão for clicado guarda o estado dele na variável
botao
if(botao == true) //Se o botão estiver no estado "Ligado"
{
Toast.makeText(MainActivity.this, "Lâmpada Acesa", Toast.LENGTH_SHORT).show(); //Exibe
mensagem "Lâmpada Acesa"
String wbuf = "l"; //Armazena "l" (Liga) na variável wbuf de escrita
mSerial.write(wbuf.getBytes()); //Envia o valor de wbuf para a serial
}
else //Se o botão estiver no estado "Desligado"
{
Toast.makeText(MainActivity.this, "Lâmpada Apagada",
Toast.LENGTH_SHORT).show(); //Exibe mensagem "Lâmpada Apagada"
String wbuf = "d"; //Armazena "d" (Desliga) na variável wbuf de escrita
mSerial.write(wbuf.getBytes()); //Envia o valor de wbuf para a serial
}
}
```

```
});
}
///============================
///+++CRIADO AUTOMATICAMENTE PELO ADT+++
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
///====================================

///+++ ENCERRA SERIAL QUANDO O APP EH FECHADO+++
@Override
public void onDestroy()
{
String wbuf = "d"; //Armazena "d" (Desliga) na variável wbuf de escrita
mSerial.write(wbuf.getBytes()); //Envia o valor de wbuf para a serial
mSerial.end(); //Encerra serial
super.onDestroy();
}
///============================================


}
```

Se você possuir um breve conhecimento em Java, poderá modificar o programa como quiser, alterando-o no projeto fornecido no início desse tutorial.

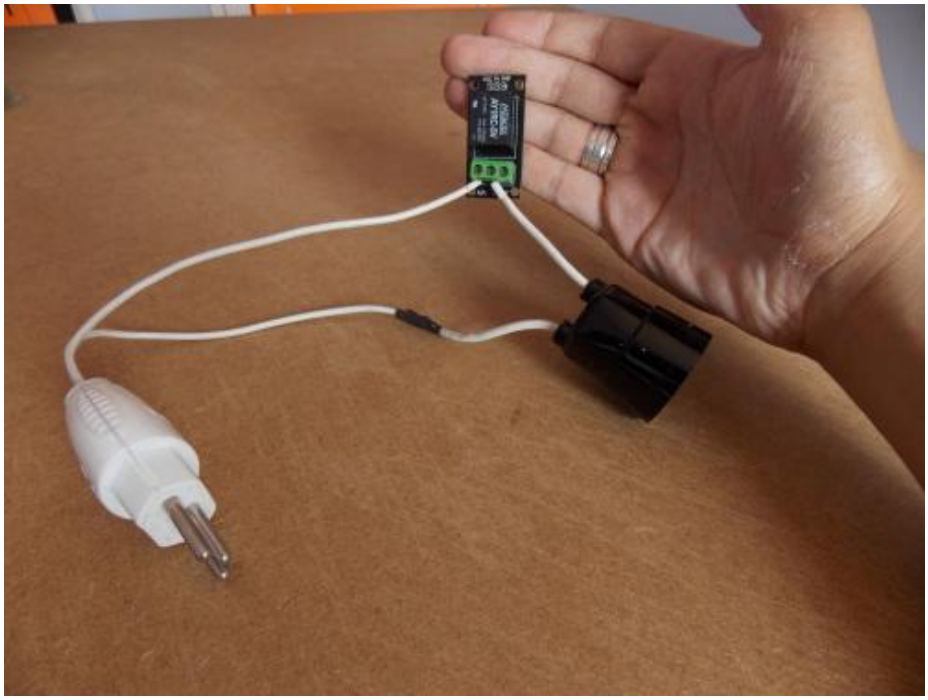
2. O Sketch do Arduino

```
/* Sketch para programar o arduino e
ler a serial, onde os comandos são enviados
pelo aplicativo android lablampada.apk
*/
```

```
char leitura;      // Cria um variável char "leitura"
#define rele 8     // Define o valor 8 à variável rele

void setup()
{
  Serial.begin(9600);     //Inicializa comunicação Serial
  pinMode(rele, OUTPUT);   //Seta o pino pelo valor no "#define rele" como saída
  digitalWrite(rele, LOW);   //Mantém rele desligado assim que iniciar o programa
}
void loop()
{
  while (Serial.available() > 0) //Verifica se há conexão com a serial
  {
    leitura = Serial.read(); //Lê o dado vindo da Serial e armazena na variável leitura
    if (leitura == 'l')  //Se a variável leitura for igual a 'l'(Botão "Ligado" no App)
     {
      digitalWrite(rele, HIGH);
     }
    else if (leitura == 'd') //Senão verifica se é igual a 'd' (Botão "Desligado" no App)
   {
      digitalWrite(rele, LOW);
   }
  }
}
```

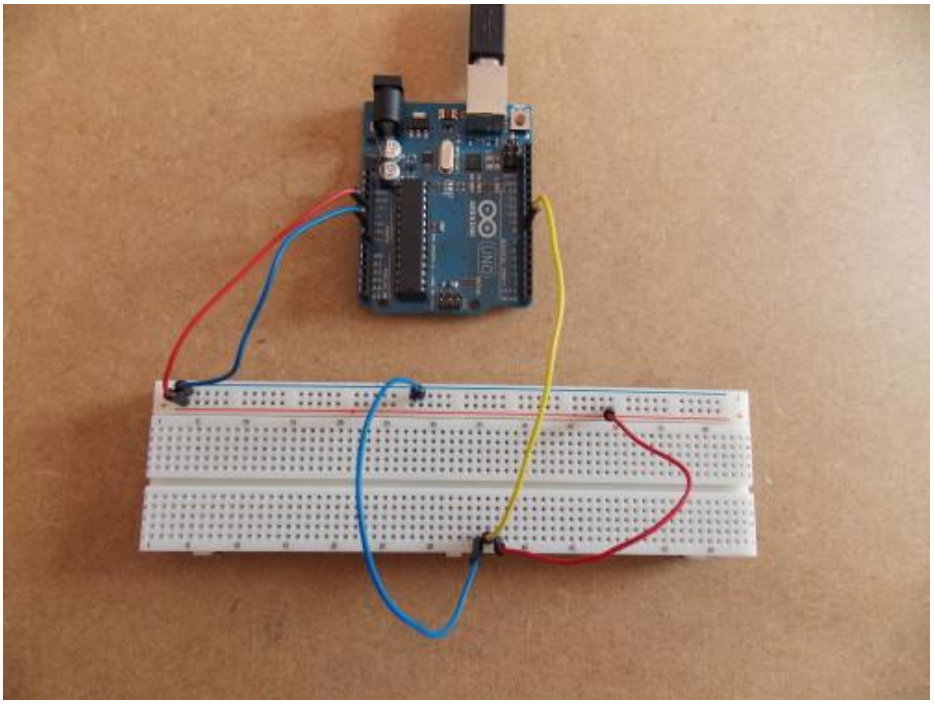
3. A Montagem

3.1) Corte um dos fios da tomada e uma das pontas conecte aos pinos C (comum) e a outra no pino NA (Normal Aberto) do módulo Relé, conforme a imagem abaixo.

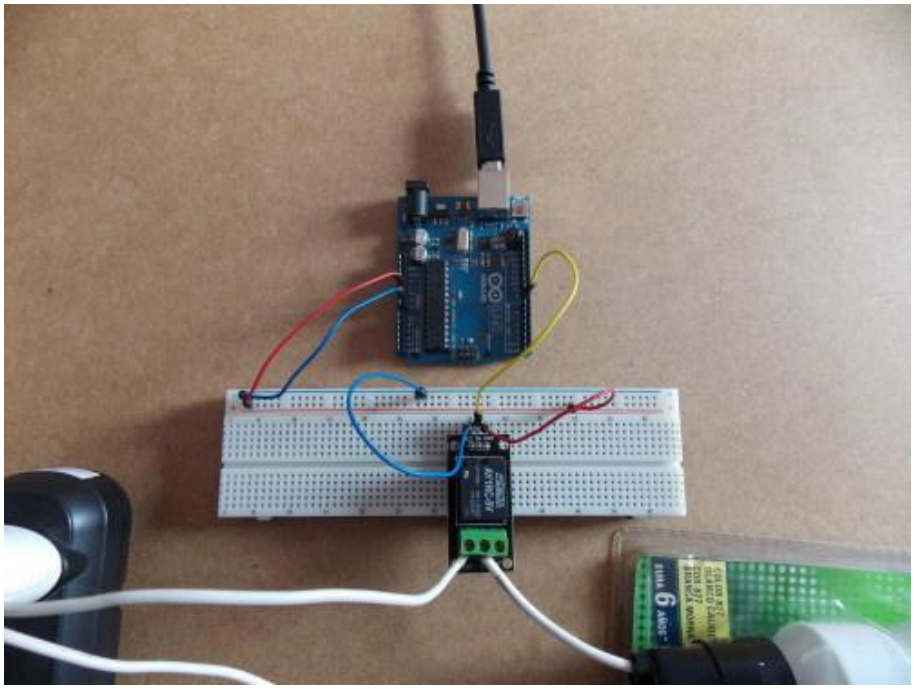


3.2) Conecte com jumpers o Pino Vcc(+) e GND(-) do Arduino aos barramentos da Protoboard.

3.3) Conecte o Pino 8 do Arduino na Protoboard entre Vcc e GND conectados aos barramentos, conforme a figura abaixo.



3.4) Conecte o Módulo Relé à Protoboard e ligue os pinos Vcc e GND, conecte o pino IN do módulo relé ao pino 8 do arduino, conforme a figura abaixo:



3.5) Faça as conexões no seu Android Mini PC, conectando também o cabo USB do arduino no USB Host do Android Mini PC.

3.6) Confira as suas conexões

Agora você pode executar seu aplicativo para acender e apagar sua lâmpada ou acionar qualquer dispositivo pelo Android Mini PC, com esse tutorial você pode começar a desenvolver aplicativo para controlar seu Android Mini PC pela serial.

Referências:

- *http://labdegaragem.com/profiles/blogs/tutorial-controlando-rele-vi...*

- *http://github.com/ksksue/FTDriver*

- *http://devmobilebrasil.com.br/android/trabalhando-com-mais-de-uma-t...*

- *http://ptandroid.com/forum/threads/tutorial-de-desenvolvimento-de-a...*

63. Tutorial: Como Utilizar o Monster Motor Shield

Olá Garagistas! No tutorial de hoje iremos utilizar o Monster Motor Shield para o controle de um Motor DC 12V que está ligado a uma roda onde iremos controlar o sentido e a velocidade de rotação.

- Lista de Materiais:

1x Arduino Uno

1x Monster Motor Shield

1x Resistor 110R

1x LED

1x Motor DC 12V

1x Fonte DC 12V

1x Roda Off - Road

Alguns Cabos

1. Funcionamento

1.1) PWM

A sigla PWM é uma abreviação para ***P**ulse **W**idth **M**odulation* ou **Modulação por Largura de Pulso**. Com essa técnica é possível controlar a velocidade dos motores, mantendo o torque ainda que em baixas velocidades, o que garante partidas suaves mesmo quando há uma carga maior sobre os motores, aspectos que caracterizam o controle **PWM** como ideal para aplicações em robótica.

O controle de velocidade de nosso motor será controlado por um PWM que pode ser definido com um valor entre "0" e "255", onde "0" indica que o motor esta parado e "255" que o motor esta em sua velocidade total.

1.2) Hardware Monster Motor Shield

PinA2 - Pino Analógico 2: Sensor de Corrente para Motor 0 via Hardware >> //cspin[0]

PinA3 - Pino Analógico 3: Sensor de Corrente para Motor 1 via Hardware >> //cspin[1]

OBS: Para definições de valores de corrente limite (**CS_THRESHOLD**) em seu projeto, consultar o Datasheet do fabricante.

PinD7 (D7) - Pino Digital 7: Pino para controle do sentido Horário do Motor 0 (A1) >> // inApin[0]

PinD8 (D8) - Pino Digital 8: Pino para controle do sentido Anti-Horário do Motor 0 (B1) >> // inBpin[0]

PinD4 (D4) - Pino Digital 4: Pino para controle do sentido Horário do Motor 1 (A2) >> // inApin[1]

PinD9 (D9) - Pino Digital 9: Pino para controle do sentido Anti-Horário do Motor 1 (B2) >> // inBpin[1]

PinD5 (D5) - Pino Digital 5: Pino para PWM Motor 0 // pwmpin[0]

PinD6 (D6) - Pino Digital 6: Pino para PWM Motor 1 // pwmpin[1]

- Tabelas da Verdade:

Motor 0		
D7	D8	
A1	B1	Condição
0	0	Parado
0	1	Anti Horário
1	0	Horário
1	1	Parado

Motor 1		
D4	D9	
A2	B2	Condição
0	0	Parado
0	1	Anti Horário
1	0	Horário
1	1	Parado

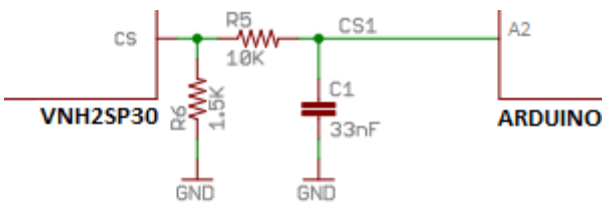
1.3) Monster Shield Exemplo

- O Serial Monitor será utilizado para acompanhar o aumento de velocidade em PWM do motor e o travamento do mesmo se ocorrer.

- O LED e a resistência entre os Pinos 13 e GND são utilizados no exemplo para indicar que o motor travou.

- A definição "**#define CS_THRESHOLD**" na quinta linha do programa serve como proteção para o circuito, onde no mesmo você define um valor máximo de corrente antes de o programa desligar seu motor. Pois, por excesso de peso, ou se por qualquer outro motivo o motor travar, o consumo de corrente aumentará, e com essa definição você protege seu circuito, fazendo com que nenhum de seus componentes queimem se a corrente aumentar muito.

- **OBS**: O valor definido nesse comando ***não é dado em A (Ampere)***, o valor é uma amostragem que o circuito faz via hardware, conforme a figura abaixo:



- **OBS**: Para a definição de corrente usada no comando, consulte o Datasheet.

- A função para controle do nosso motor é "**motorGo(motor, sentido, pwm);**" onde:

motor: É a saída do motor que iremos utilizar: "0" (A1:B1) e/ou "1" (A2:B2)

sentido: "CW" para sentido horário de rotação e "CCW" para sentido anti-horário de rotação

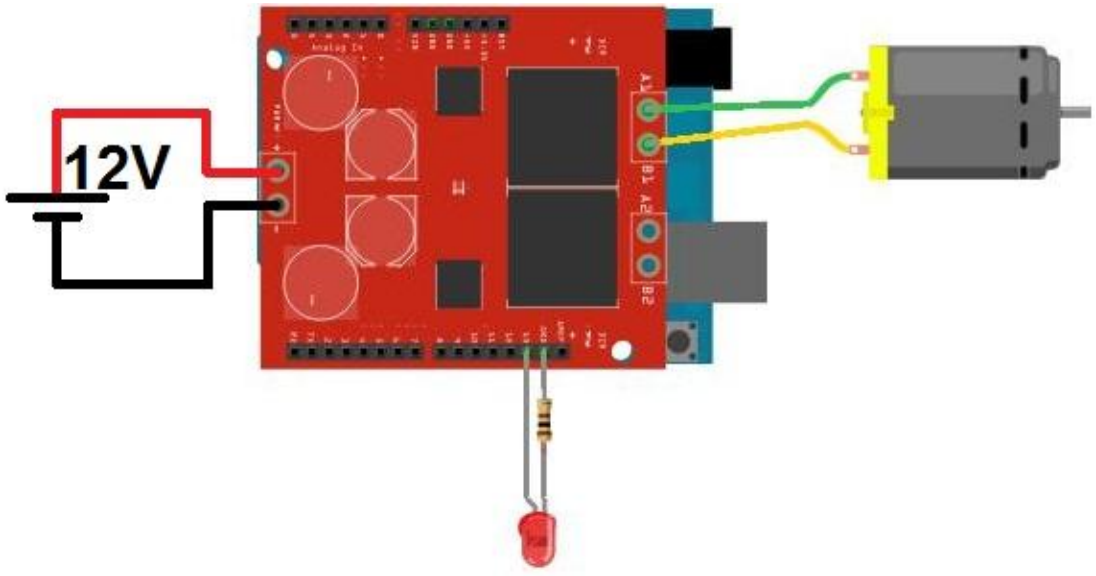
pwm: Valor entre "0" e "255" para controle de velocidade do motor

Exemplo: "motorGo(**0**, **CW**, **255**);"

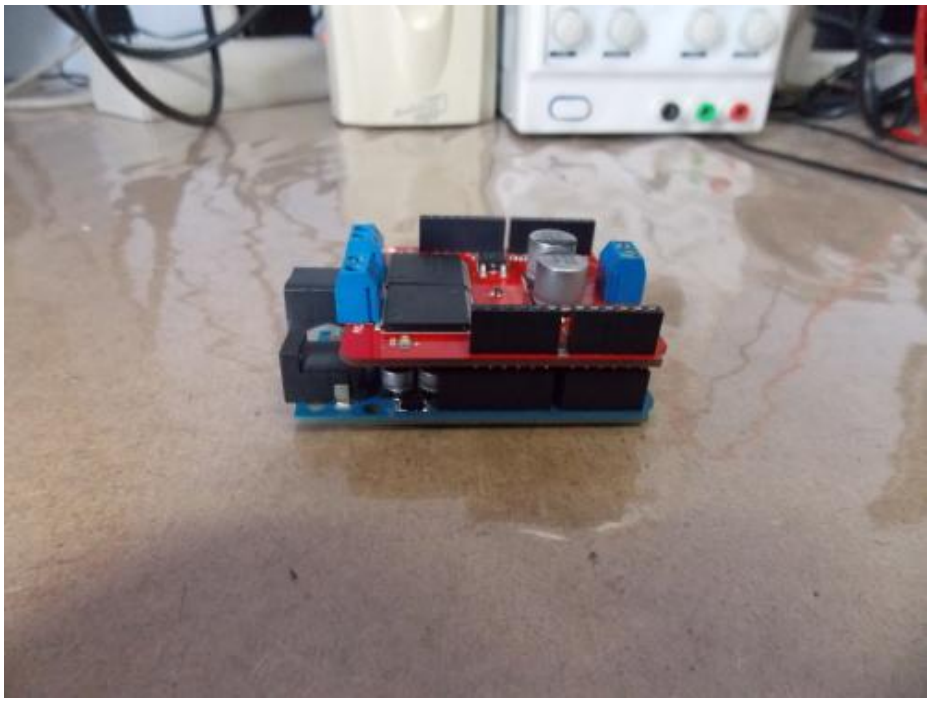
No exemplo, o motor 1 irá girar no sentido horário em sua velocidade máxima.

2. A Montagem

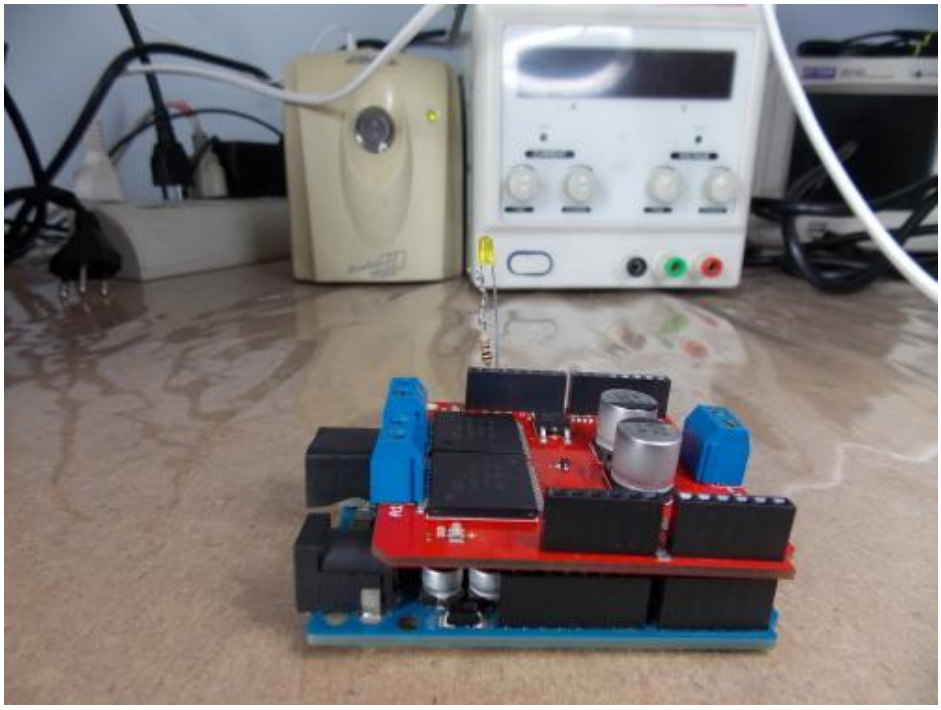
Esquema da Montagem:



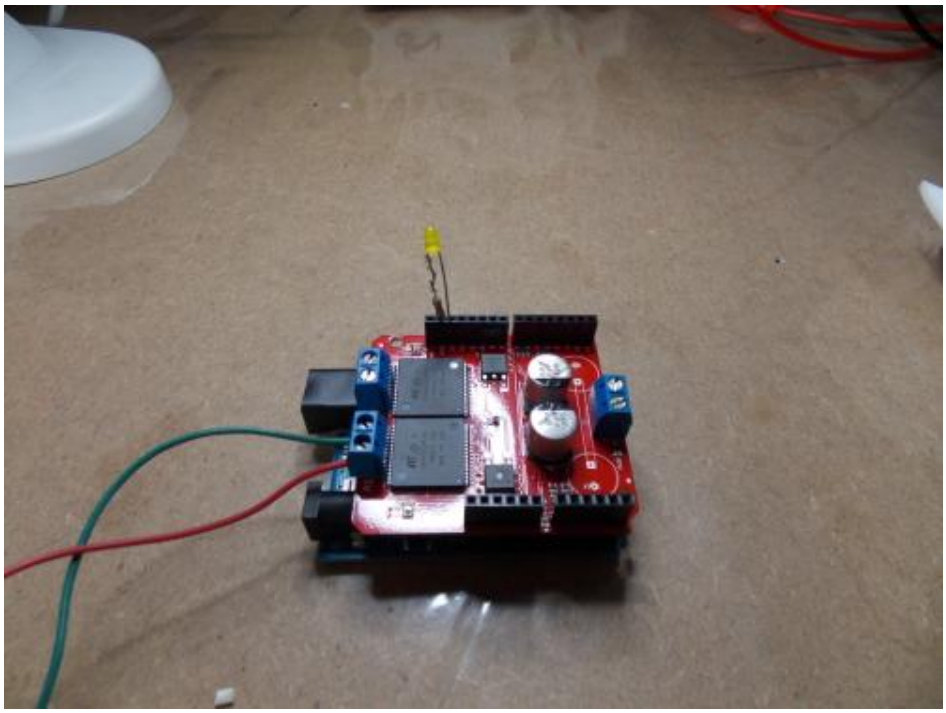
2.1) Conecte seu Monster Motor Shield ao Arduino



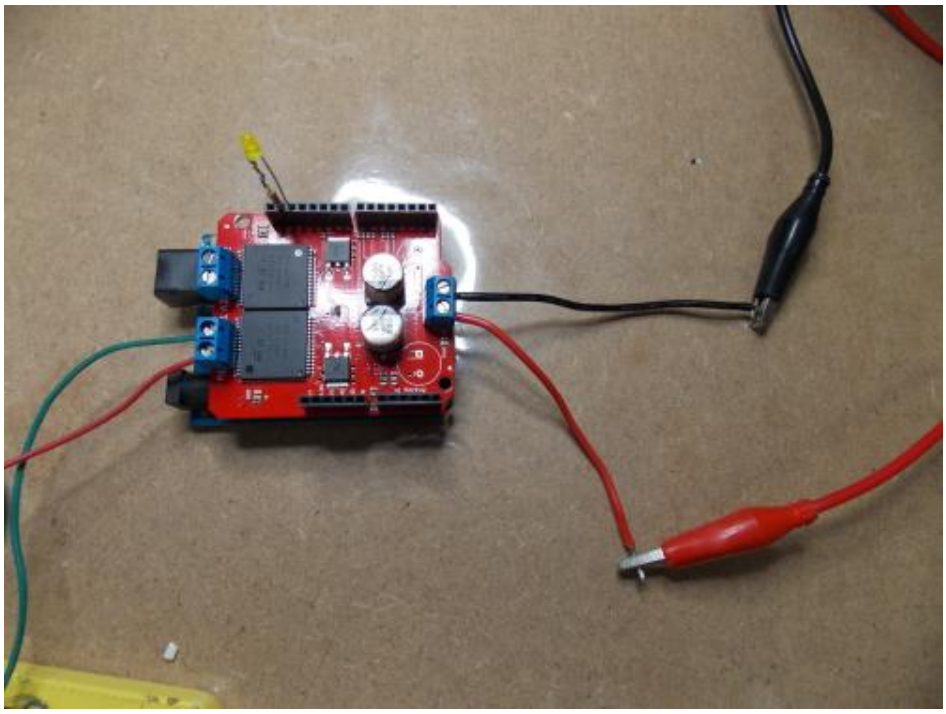
2.2) Conecte o LED com a resitência entre o pino 13 e o GND que será utilizado no exemplo para indicar que o motor foi travado.



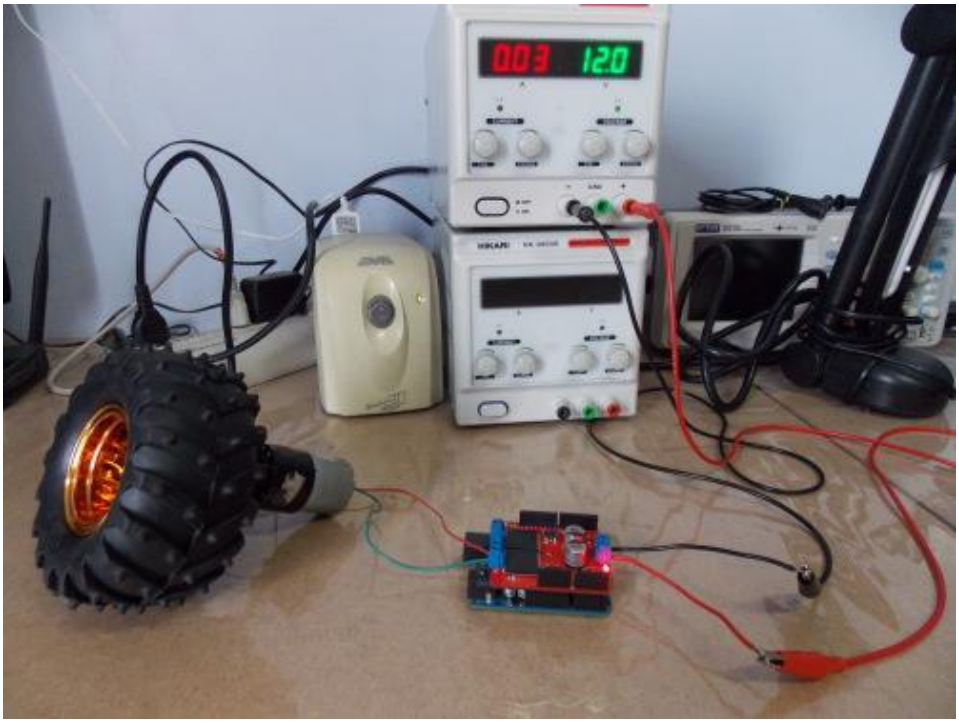
2.3) Conecte o motor nos pinos A1:B1 do Shield



2.4) Conecte a fonte 12V nos pinos + e - do Monster Shield



2.5) Circuito Montado



3. O Sketch

```
#define BRAKEVCC 0
#define CW 1
#define CCW 2
#define BRAKEGND 3
#define CS_THRESHOLD 15   // Definição da corrente de segurança (Consulte: "1.3) Monster Shield Exemplo").


int inApin[2] = {7, 4}; // INA: Sentido Horário Motor0 e Motor1 (Consulte:"1.2) Hardware Monster Motor Shield").
int inBpin[2] = {8, 9}; // INB: Sentido Anti-Horário Motor0 e Motor1 (Consulte: "1.2) Hardware Monster Motor Shield").
int pwmpin[2] = {5, 6};          // Entrada do PWM
int cspin[2] = {2, 3};           // Entrada do Sensor de Corrente
int statpin = 13;
int i=0;;
void setup()                     // Faz as configuração para a utilização das funções no Sketch
{
Serial.begin(9600);              // Iniciar a serial para fazer o monitoramento
pinMode(statpin, OUTPUT);
for (int i=0; i<2; i++)
   {
   pinMode(inApin[i], OUTPUT);
   pinMode(inBpin[i], OUTPUT);
   pinMode(pwmpin[i], OUTPUT);
   }
for (int i=0; i<2; i++)
   {
   digitalWrite(inApin[i], LOW);
   digitalWrite(inBpin[i], LOW);
   }
}
```

```
void loop()                    // Programa roda dentro do loop
{
while(i<255)
   {
   motorGo(0, CW, i);          // Aumento do o PWM do motor até 255
   delay(50);                  // Se o motor travar ele desliga o motor e
   i++;                        // Reinicia o processo de aumento do PWM
   if (analogRead(cspin[0]) > CS_THRESHOLD)
   motorOff(0);
   Serial.println(i);
   digitalWrite(statpin, LOW);
   }
i=1;
while(i!=0)
   {                           // Mantém o PWM em 255 (Velocidade Máxima do Motor)
   motorGo(0, CW, 255);        // Se o motor travar ele desliga o motor e
   if (analogRead(cspin[0]) > CS_THRESHOLD)          // Reinicia o processo de aumento do
PWM
   motorOff(0);
   }
}
void motorOff(int motor)     //Função para desligar o motor se o mesmo travar
{
for (int i=0; i<2; i++)
   {
   digitalWrite(inApin[i], LOW);
   digitalWrite(inBpin[i], LOW);
   }
analogWrite(pwmpin[motor], 0);
i=0;
digitalWrite(13, HIGH);
Serial.println("Motor Travado");
delay(1000);
}
```

```
void motorGo(uint8_t motor, uint8_t direct, uint8_t pwm)         //Função que controla as
variáveis: motor(0 ou 1), sentido (cw ou ccw) e pwm (entra 0 e 255);
{
if (motor <= 1)
   {
   if (direct <=4)
      {
      if (direct <=1)
         digitalWrite(inApin[motor], HIGH);
      else
         digitalWrite(inApin[motor], LOW);

      if ((direct==0)||(direct==2))
         digitalWrite(inBpin[motor], HIGH);
      else
         digitalWrite(inBpin[motor], LOW);

      analogWrite(pwmpin[motor], pwm);
      }
   }
}
```

É isso ai Garagistas! Espero que tenham gostado desde tutorial. Até a Próxima!

Referências:

http://www.pnca.com.br/index.php?option=com_content&view=articl...

http://en.wikiversity.org/wiki/Arduino/MonsterMotorShield

http://en.wikiversity.org/wiki/Arduino/MonsterMotorShield/UnoCode

http://www.labdegaragem.org/loja/index.php/31-shields/monster-moto-...

64. Tutorial - Arduino Due como Mouse

No tutorial de hoje montaremos o exemplo ButtonMouseControl disponível na interface Arduino 1.5.2 para demonstrar como é o funcionamento do Arduino Due como dispositivo USB e vamos apresentar as principais diferenças entre ele e os outros Arduinos.

Lista de Materiais

1 x Arduino Due

1 x Protoboard

5 x Chaves Tactil

5 x Resistores de 1KΩ

Alguns jumpers

O Arduino Due

Arduino Due é a mais nova placa lançada pela Arduino. Ela é baseada no microcontrolador Atmel SAM3X8E ARM Cortex-M3 que e é o primeiro da família Arduino a utilizar a plataforma ARM de 32 bits.

O Arduino Due é essencial em projetos que necessitam de grande poder computacional com alta velocidade de aquisição e processamento de dados, pois com a substituição do ATmega328 (Arduino Uno e Mega), pelo SAM3 (que tem sua CPU baseada no ARM3), faz com que o Arduino Due tenha um desempenho muito superior.

Devido ao DAC (Conversor Digital Analógico), o Due trouxe a possibilidade da criação de uma biblioteca capaz de ler arquivos de áudio para serem reproduzidos nestas saídas que convertem valores digitais em analógicos.

Abaixo você vê as principais características que a diferem das outras placas Arduino:

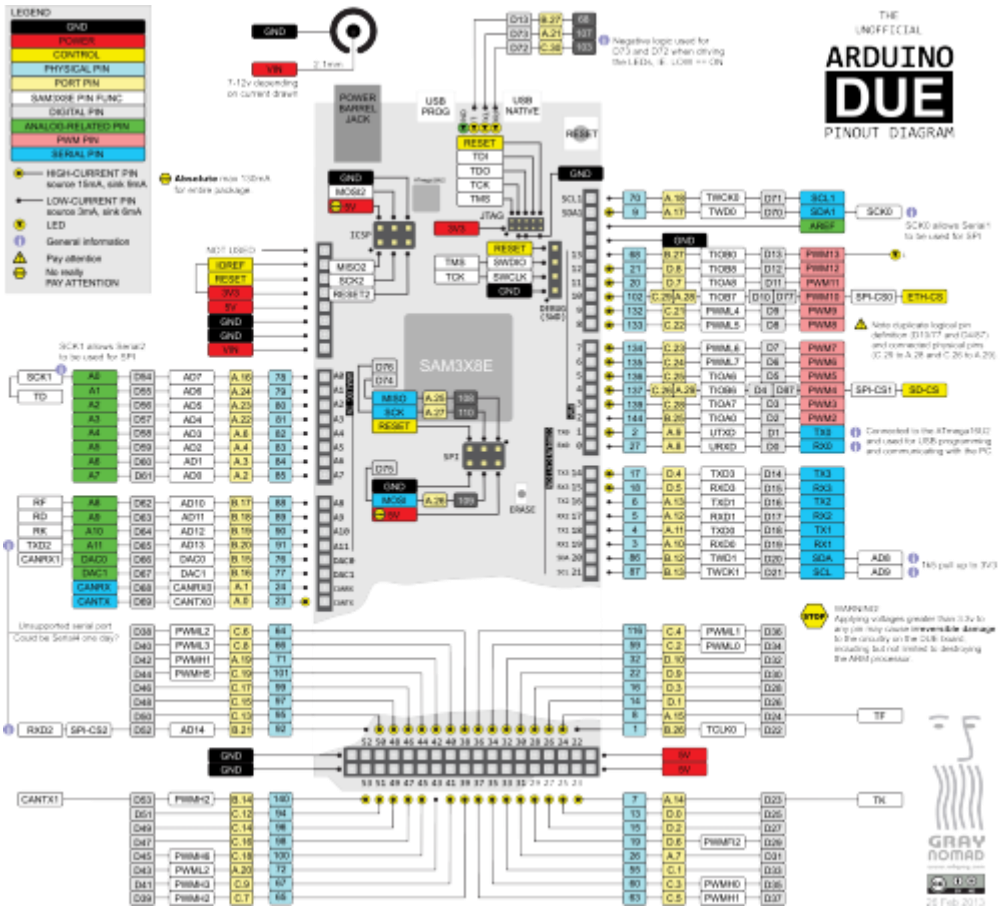
- Frequência de 84MHz (clock 5 vezes maior que os Arduinos UNO e MEGA)
- 54 Pinos de Entrada e Saída (0 a 3,3V)
- 12 Pinos que podem ser usados com PWM
- 12 Entradas Analógicas
- 96 Kbytes de SRAM
- 512 Kbytes de Flash para colocar seu programa
- 2 DACs (Conversor Digital Analógico Integrado)
- 1 Botão Erase para apagar a flash do ARM

Interfaces de Comunicação:

- 4 Interfaces UARTs
- 2 Interfaces I2C
- 1 Interface CAN
- 1 Interface SPI
- 1 Interface JTAG
- 1 Conexão USB OTG (Assim como o Leonardo)

Os Pinos

Na imagem abaixo você pode ver o esquemático com as funções de cada Arduino Due:

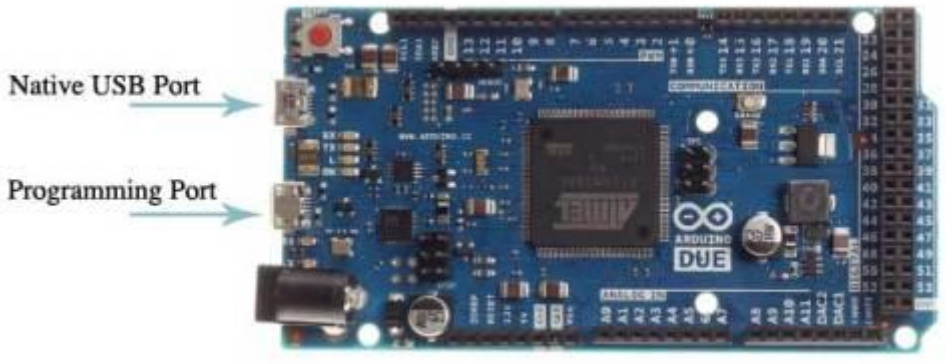


Atenção!!! O Arduino Due opera com tensão de 3,3V em seus pinos de entrada e saída. Qualquer tensão acima de 3,3V nos pinos pode danificar o microcontrolador.

Devido a esta tensão de trabalho do Arduino Due, alguns shields que necessitem de 5V para funcionar, se tornam incompatíveis com o Due contudo, os shields no formato Arduino R3 são 100% compatíveis (como por exemplo o Ethernet Shield).

Programação

Porta Nativa e Porta de Programação



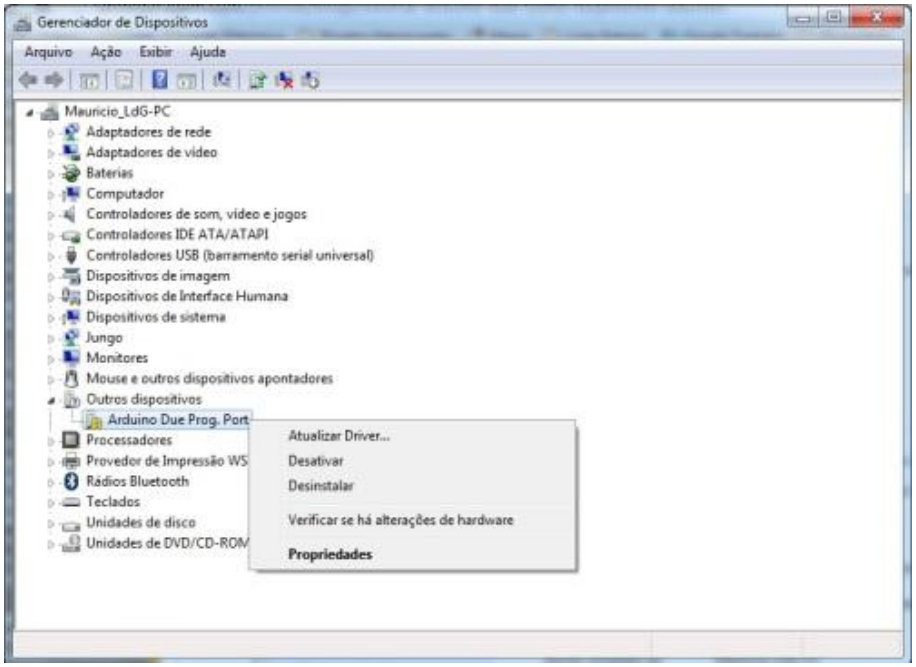
Quando for necessário transferir a programação recomenda-se utilizar porta de programação. A porta nativa deve ser usada quando o Due for programado como um

Dispositivo USB. A programação feita pela porta de programação dá um Reset via Hardware no microcontrolador. A programação pode ser feita também pela porta nativa, mas por esta porta, a programação pode falhar, já que neste modo, ele executa um Soft-Reset que pode causar o travamento do microcontrolador. Caso isto ocorra será necessário pressionar por alguns segundos o botão Erase (que apagará a memória flash do ARM) depois tentar programa-lo novamente (dê preferência pela porta de programação).

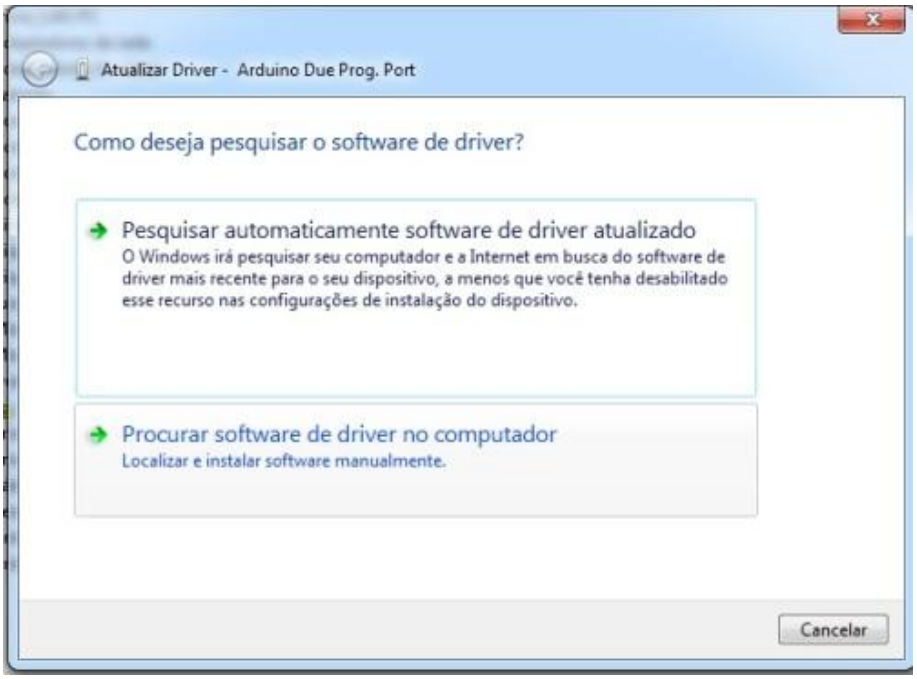
Passo a passo:

Antes de mais nada, faça o download do software Arduino versão 1.5.2 Beta. *O Software Arduino* versão 1.0 **não tem suporte** ao Arduino Due.

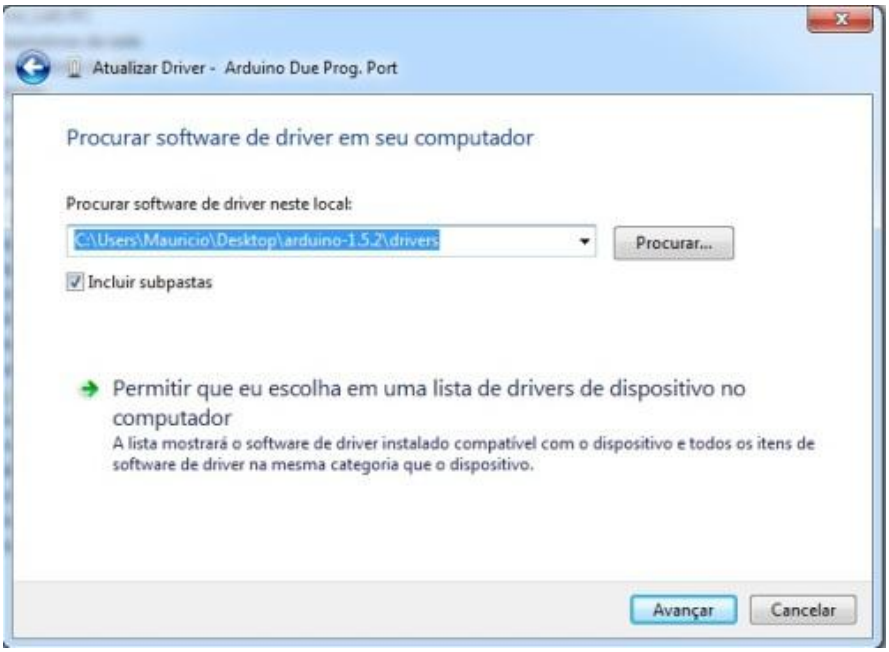
Para fazer a programação do Due, primeiro você deve conecta-lo pela porta de Programação e instalar o Driver contido na pasta Driver conforme segue na sequência de fotos:



Clique em Atualizar Driver



Clique em "Procurar Software de Driver no Computador"



Clique em "Procurar" e selecione a pasta "driver", dentro da pasta arduino-1.5.2

Se a tela acima aparecer, clique em "Instalar este software de driver mesmo assim"

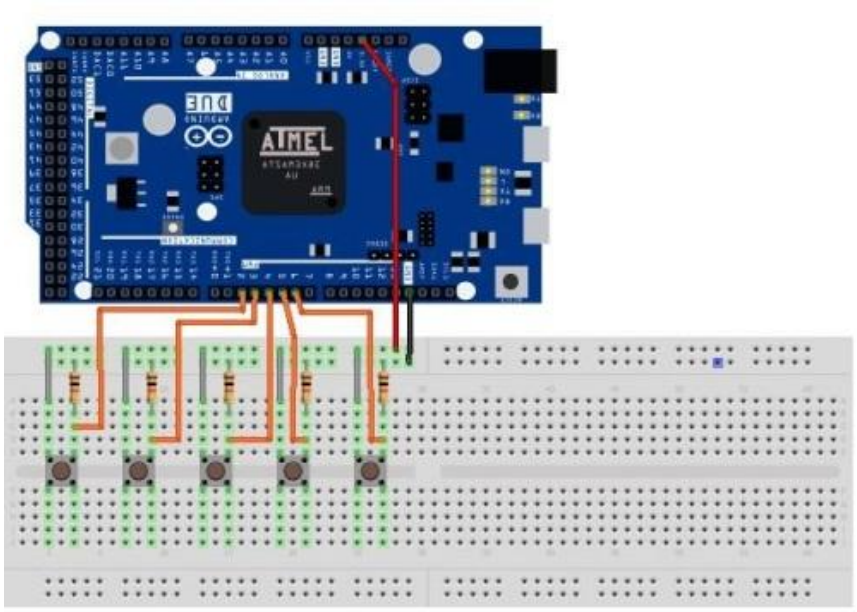
Com isto a porta de programação esta corretamente instalada.

Abra o software Arduino 1.5.2 , Selecione a placa Arduino Due (Programming Port) , Selecione a Porta serial que ela foi identificada e grave o Sketch exemplo localizado na barra de tarefas, pelo caminho:

File -> Examples -> 09.USB -> ButtonMouseControl

Altere a conexão USB para a porta Nativa do Due e faça o mesmo procedimento de instalação do driver descrito para a porta de programação. Ele vai localizar um Hardware chamada Arduino Due.

Agora desconecte o Due da porta USB e monte o circuito conforme a figura abaixo:



Verifique a montagem e reconecte o Arduino Due pela porta Nativa em seu computador. Pressione as teclas de direção e você verá mouse se movimentando na tela. O botão esquerdo de clique também pode ser utilizado.

É isto Garagistas!!! Esperamos que gostem do Tutorial e qualquer dúvida ou sugestão comentem abaixo.

Referências

http://arduino.cc/en/Tutorial/ButtonMouseControl

http://arduino.cc/en/Main/ArduinoBoardDue

65. Tutorial: Seguidor de linha com plataforma Zumo e Garagino

Olá Garagistas!!!

Neste tutorial montamos um seguidor de linha baseado na plataforma robótica Zumo junto com sensores de linha do tipo digital. A pista foi feita com fita isolante sobre um pedaço de cartolina branca, inspirada no circuito Luigi Raceway do jogo Mario Kart 64.

Lista de materiais

1 x Garagino Rev 1 ou Arduino Uno Rev 3

1 x Chassis Zumo

1 x Pololu Motor Driver 1A Dual TB6612FGN

2 x Micro motor com caixa de redução de metal 75:1

2 x Sensores de Linha - QRE1113 -Digital

1 x Protoboard

1 x Fita dupla-face

1 x Fita Isolante

4 x Pilhas recarregáveis (se utilizar pilhas normais você necessitará de um regulador para o Garagino)

Alguns jumpers

Funcionamento

O sensor de linha digital, devolve um valor dependendo da refletância da superfície na qual ele esta apontando. Com isto detectamos se o carrinho esta saindo da linha e acionamos os motores de forma a nunca permitir que saia da linha.

Para mais informações sobre o sensor de linha digital acesse este outro tutorial: http://labdegaragem.com/profiles/blogs/tutorial-como-utilizar-o-breakout-de-sensor-de-linha-com-arduino

Na figura abaixo você pode ver o resumo da lógica utilizada para controlar o Zumo:

Sensor dir.: Sem Faixa
Sensor Esq.: Com Faixa
Ação
Motor dir.: Liga
Motor Esq.: Desliga

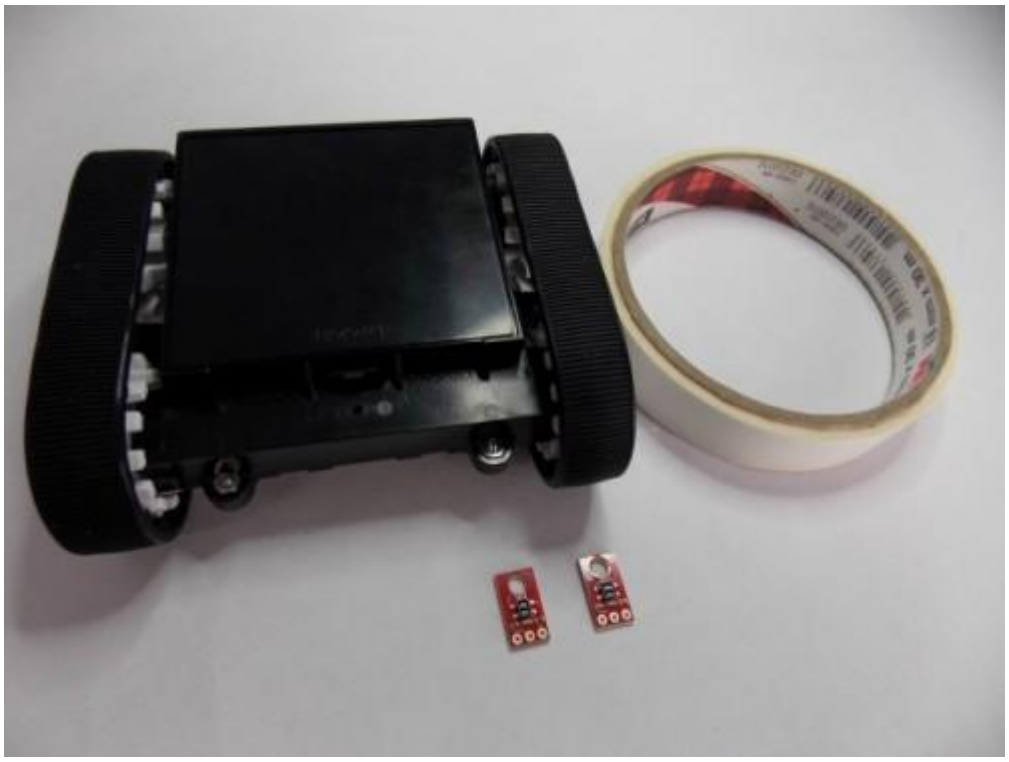
Sensor dir.: Sem Faixa
Sensor Esq.: Sem Faixa
Ação
Motor dir.: Liga
Motor Esq.: Liga

Sensor dir.: Com Faixa
Sensor Esq.: Sem Faixa
Ação
Motor dir.: Desliga
Motor Esq.: Liga

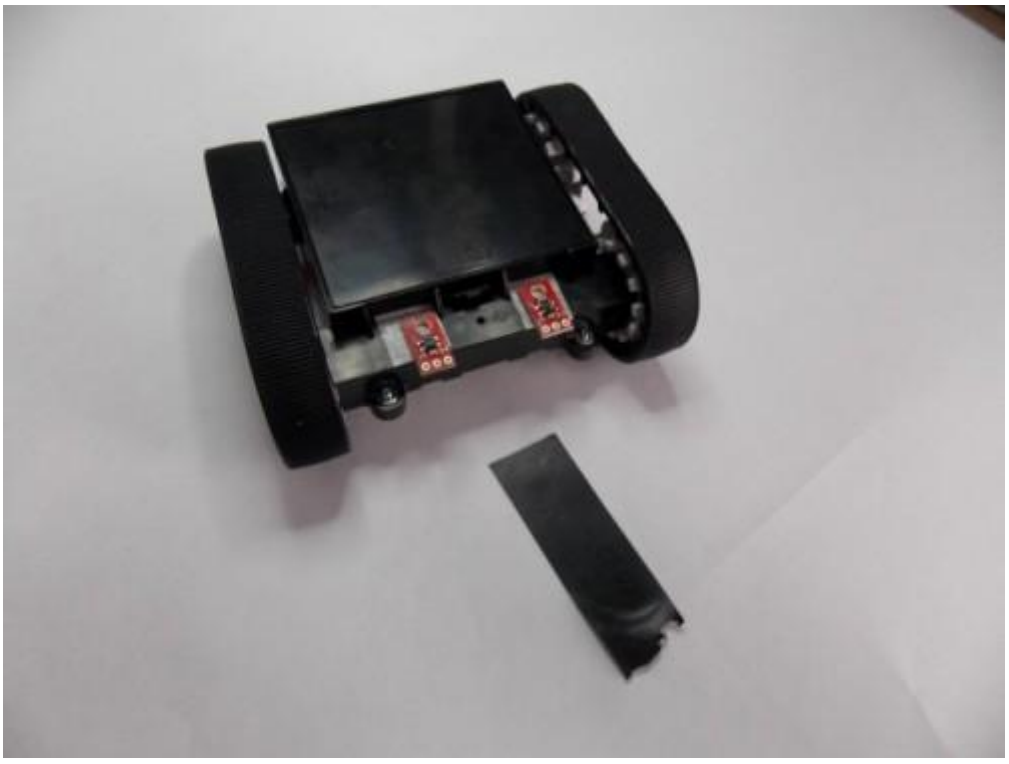
Os micromotores foram controlados pelo driver de motores que utiliza o circuito integrado TB6612FGN, que consiste em duas pontes H e utilizamos cada uma delas para controlar os motores individualmente.

Foi pensado em uma maneira de parar o carrinho caso ele fosse de encontro com uma linha preta de frente, neste caso os dois sensores iriam acionar . Nesta situação o carrinho trava tentando recuar em velocidade reduzida. Isto serve para você proteger a sua pista com um fita isolante mais larga e impedir que o carrinho caia da mesa que ele estiver andando.

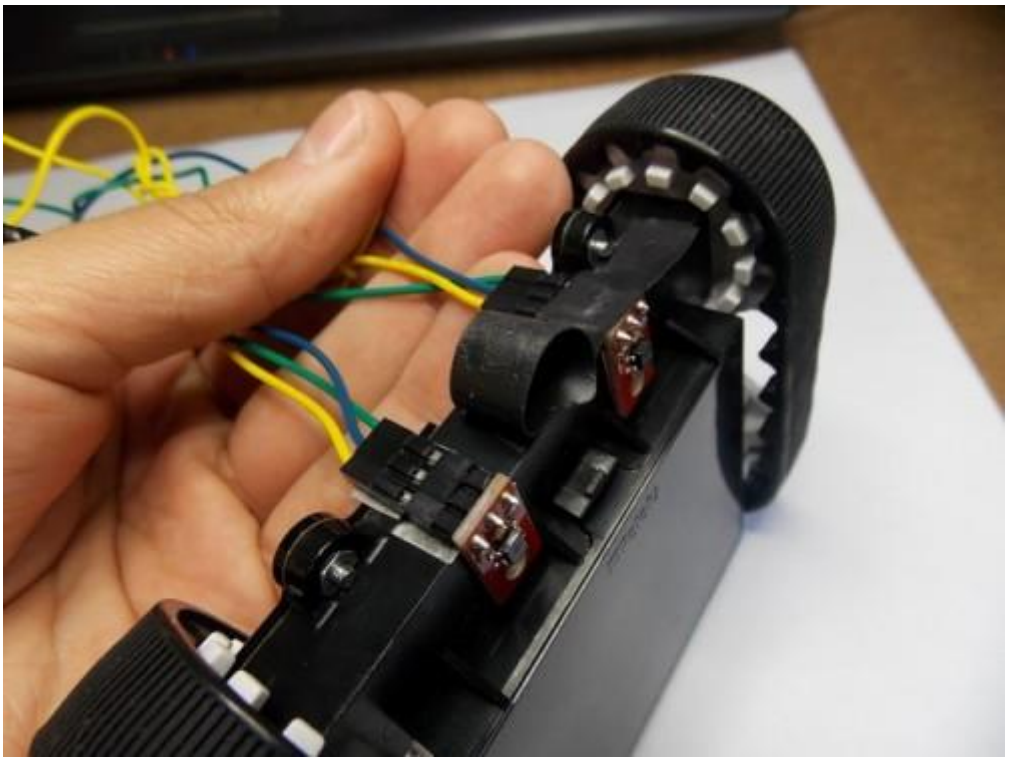
Fotos da montagem



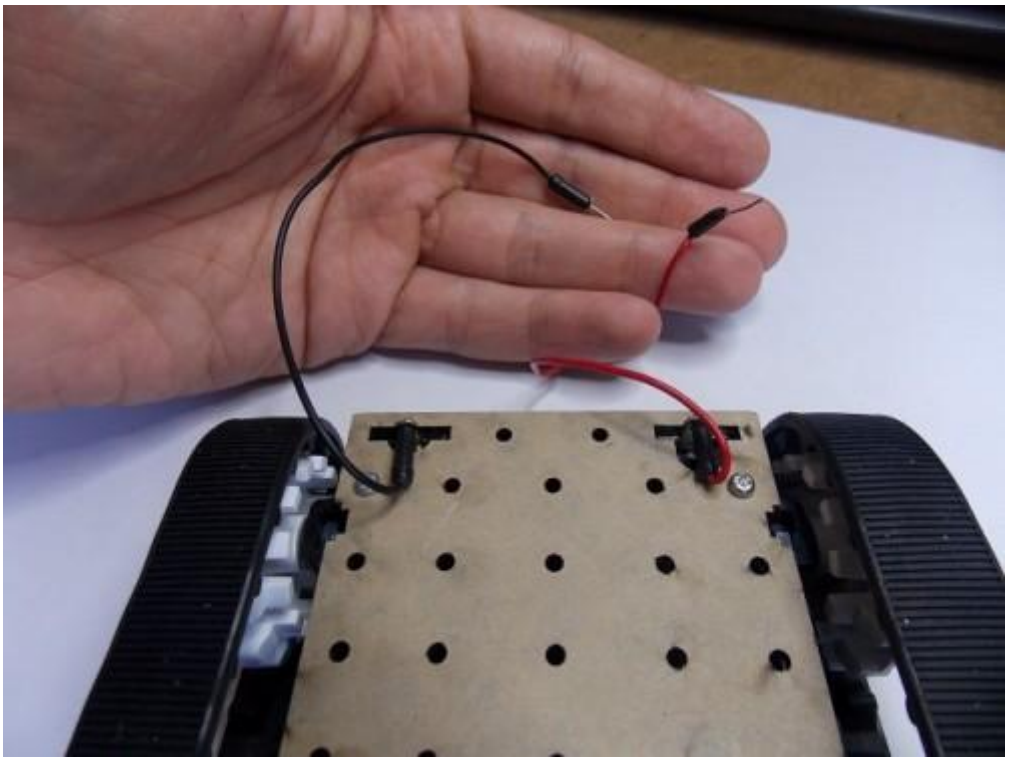
Plataforma Zumo montada com os micromotores, fita dupla face e os sensores de linha.



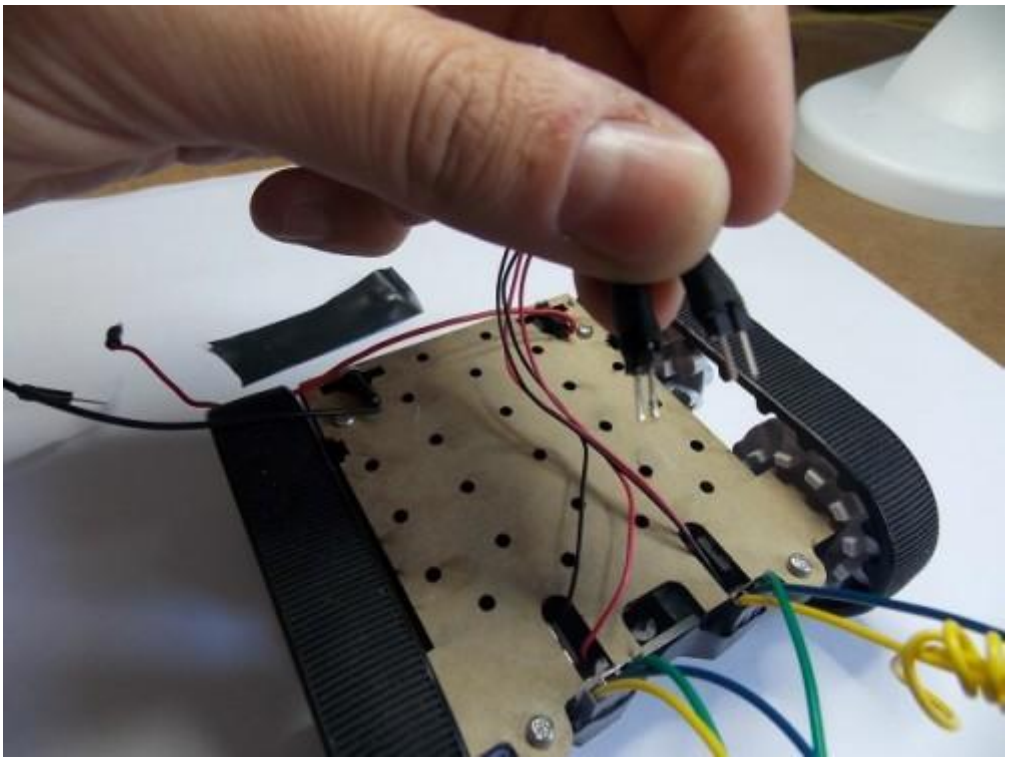
A distância entre os sensores deve ser um pouco maior que a largura da fita isolante.



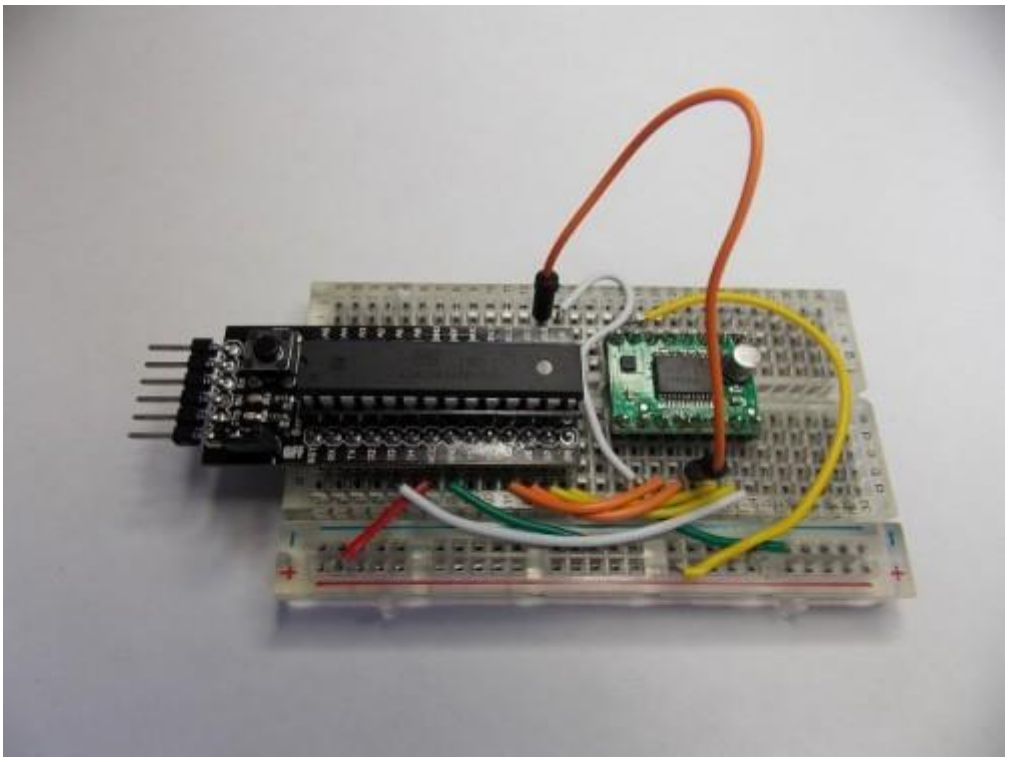
Cabos dos sensores (fios amarelos 5V, fios verdes GND e os fios azuis para os pinos digitais 11 e 12 do Garagino)



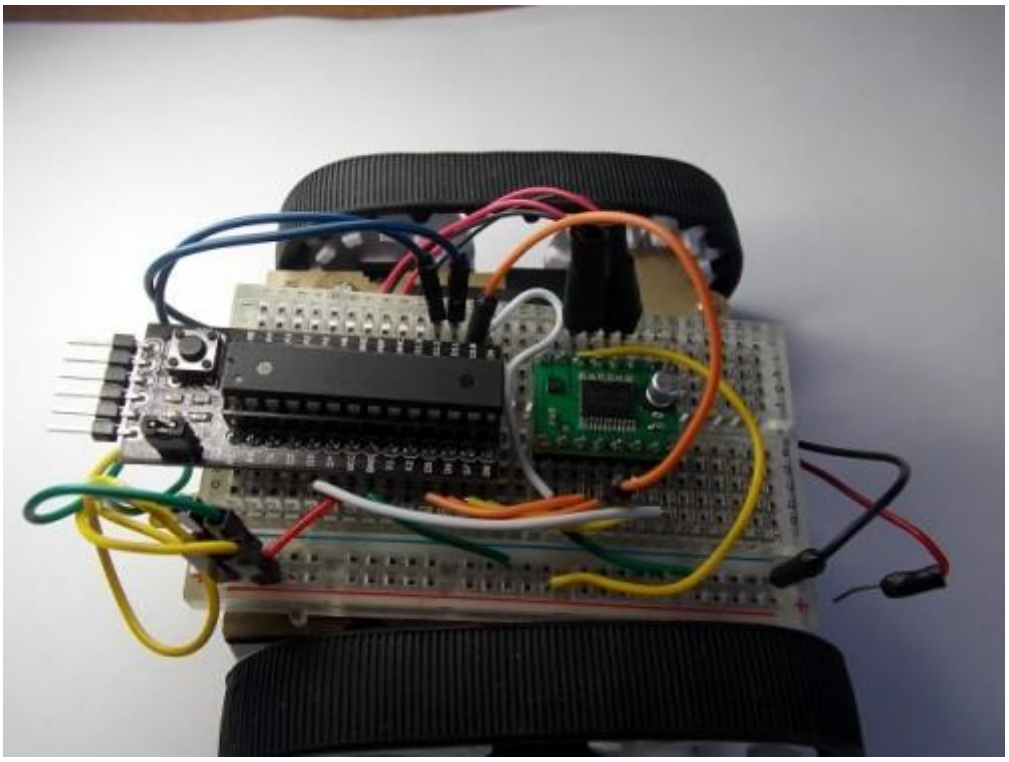
Fios soldados no suporte para as pilhas que vem no Zumo



Cabo dos micromotores com termo retrátil



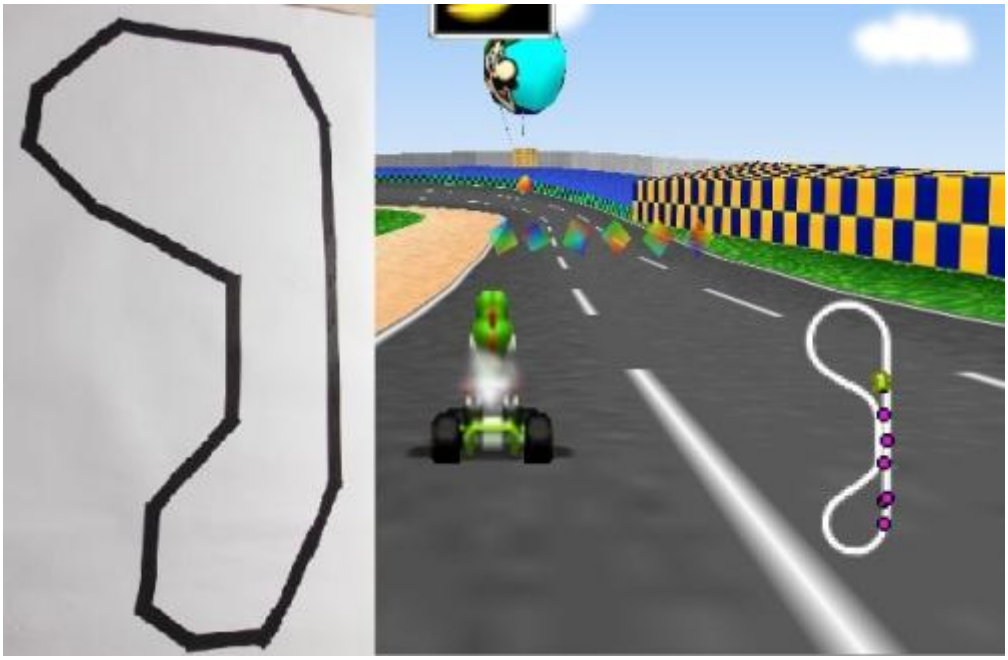
Protoboard com a montagem entre o Driver dos motores (ponte H) e o Garagino



Montagem da Protoboard junto com a ligação dos sensores no Zumo

A Pista

Na imagem abaixo, à esquerda você vê nossa pista de fita isolante e à direita a pista Luigi Raceway do jogo Mario Kart 64 na qual a nossa foi inspirada:



Dêem um desconto...Fiz tudo com fita isolante.

Sketch

Este sketch foi baseado nos tutorias que estão nos links de referência deste tutorial.

```
//Definição do pino que inibe o funcionamento dos motores

#define PARAR 10 //Parar o Carrinho

//Definições para controlar o Motor A (esquerda)
#define PWMA  3 //Controle de Velocidade
#define AIN1  7 //Direção
#define AIN2  8 //Direção
#define MOTOR_E 1 //Define motor da Direita com valor 1

//Definições para controlar o Motor B (direita)
#define PWMB  9  //Pino de Controle da Velocidade
#define BIN1  6  //Pino de Direção
#define BIN2  5  //Pino de Direção
#define MOTOR_D 2 //Define motor da Direita com valor 2

//Definição dos pinos onde estão os sensores
#define linhaD 11
#define linhaE 12

void setup()
{
  //Define pinos como saída
  pinMode(PARAR, OUTPUT);

pinMode(PWMA, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);

pinMode(PWMB, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
}

void loop(){

int val_sensores = lesensores();

switch(val_sensores){
  case 0:
```

```
    move(MOTOR_E, 255, 1); //Motor da Esquerda, a toda velocidade, sentido de rotação
    move(MOTOR_D, 255, 1); //Motor da Direita, a toda velocidade, sentido de rotação
    break;
case 1:
    move(MOTOR_D, 0, 1); //Motor da Direita, para o Motor, sentido de rotação
    move(MOTOR_E, 255, 1); //Motor da Esquerda, a toda velocidade, sentido de rotação
    break;
case 2:
    move(MOTOR_E, 0, 1); //Motor da Esquerda, para o Motor, sentido de rotação
    move(MOTOR_D, 255, 1); //Motor da Direita, a toda velocidade, sentido de rotação
    break;
case 3:
    move(MOTOR_E, 100, 2); //Motor da Esquerda, baixa velocidade, reverso
    move(MOTOR_D, 100, 2); //Motor da Direita, baixa velocidade, reverso
    break;

  }
}
void move(int motor, int speed, int direction){
  digitalWrite(PARAR, HIGH); //Desabilita o standby
boolean inPin1 = LOW;
  boolean inPin2 = HIGH;
if(direction == 1)
  {
    inPin1 = HIGH;
    inPin2 = LOW;
  }
if(motor == 1)
  {
    digitalWrite(AIN1, inPin1);
    digitalWrite(AIN2, inPin2);
    analogWrite(PWMA, speed);
```

```
  }
  else
  {
    digitalWrite(BIN1, inPin1);
    digitalWrite(BIN2, inPin2);
    analogWrite(PWMB, speed);
   }
}

int lesensores(){

//Ler Sensor da Direita
  int sensores = 0;
  pinMode( linhaD, OUTPUT );
  digitalWrite( linhaD, HIGH );
  delayMicroseconds(10);
  pinMode( linhaD, INPUT );
  long time = micros();

  while (digitalRead(linhaD) == HIGH && micros() - time < 3000);
  int diff = micros() - time;

if (diff > 1600)
    sensores += 1;

// Ler Sensor da Esquerda
  pinMode( linhaE, OUTPUT );
  digitalWrite( linhaE, HIGH );
  delayMicroseconds(10);
  pinMode( linhaE, INPUT );
  long time1 = micros();

  while (digitalRead(linhaE) == HIGH && micros() - time1 < 3000);
  int diff1 = micros() - time1;

  if (diff1 > 1600)
    sensores += 2;
```

```
  return sensores;
}
```

Isto é tudo pessoal!!! Se gostaram deste tutorial ou se tiverem alguma dúvida, comentem abaixo e deixe sua opnião.

Referências

http://bildr.org/2012/04/tb6612fng-arduino/

http://bildr.org/2011/06/qre1113-arduino/

http://www.pololu.com/catalog/product/713

http://kile.stravaganza.org/project/lego-robot-line-follower

66. Tutorial: Controlando carrinho pelas teclas WASD via comandos Seriais

Neste tutorial falaremos como controlar um carrinho atráves das teclas WASD ou via teclado numérico com Arduino. Para isto utilizaremos drivers de motores DC baseados no chip TB6612FNG e em nossa loja você encontra duas opções, uma da Sparkfun e a outra da Pololu e abordaremos a diferença entre eles mais a frente.

Lista de Materiais

1 x Arduino Uno Rev 3 ou Garagino Rev 1

1 x Chassis Zumo

1 x Driver Pololu ou Sparkfun - Motor Driver 1A Dual TB6612FGN

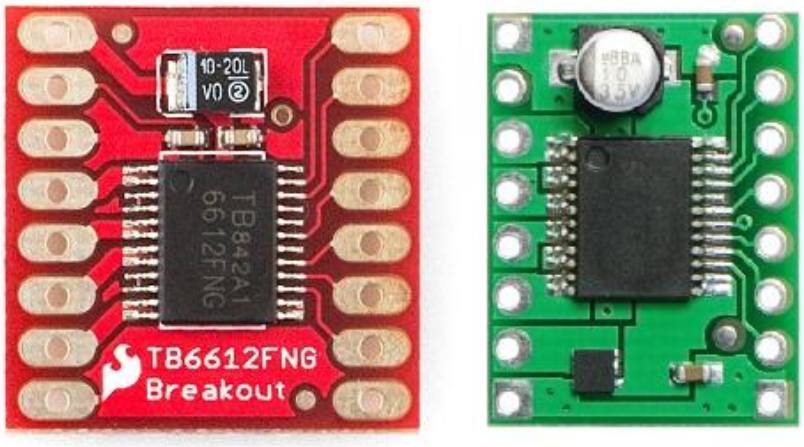
2 x Micro motor com caixa de redução de metal 75:1

1 x Protoboard

Alguns jumpers

Drivers baseados no Chip TB6612FGN

Abaixo você vê a imagem dos dois Drivers:



Vista Superior dos drivers - A esquerda driver da Sparkfun e a direita o da Pololu

No Tutorial: Seguidor de linha com plataforma Zumo e Garagino, utilizamos o driver da Pololu para controlar os motores que moviam o carrinho. Neste tutorial vamos dar mais informações sobre estes Drivers.

Estes drivers são versáteis e podem controlar 2 motores DC com consumo de corrente constante máxima de 1,2 A e corrente de pico máxima de até 3,2A. Cada motor é identificado pelas letras A e B e os pinos de saída são respectivamente AO1, AO2, e BO1, BO2.

Para cada motor, há 3 pinos de controle, sendo eles: PWMA, AIN1 e AIN2 para o motor A e PWMB, BIN1 e BIN2 para o motor B.

Há ainda um pino de Standby (STBY) que se estiver em nível lógico 0, para a operação do Driver e desativa os dois motores.

Modos de Operação

Os modos de operação são 5: Sentido Horário (CW), Sentido Anti-horário(CCW) , Freio (Short Brake), Parar (Stop) e Standby.

Estes modos são selecionados conforme a tabela abaixo:

Input				Output		
IN1	IN2	PWM	STBY	OUT1	OUT2	Mode
H	H	H/L	H	L	L	Short brake
L	H	H	H	L	H	CCW
		L	H	L	L	Short brake
H	L	H	H	H	L	CW
		L	H	L	L	Short brake
L	L	H	H	OFF (High impedance)		Stop
H/L	H/L	H/L	L	OFF (High impedance)		Standby

Legenda:

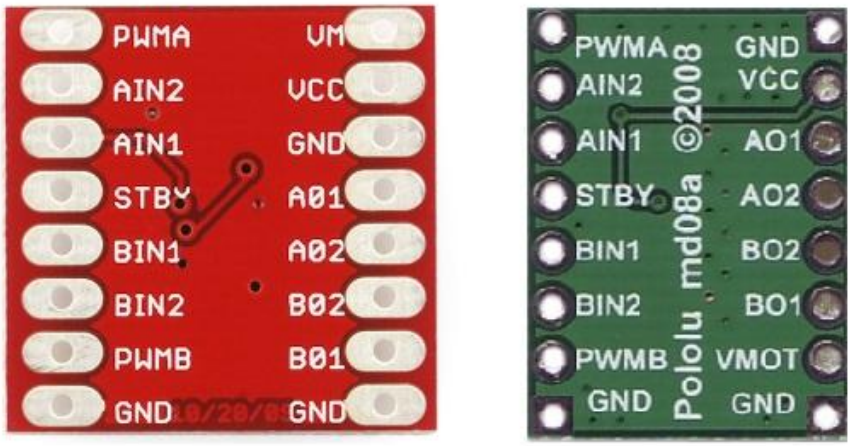
H - Nível lógico 1

L - Nível lógico 0

Diferença entre os Drivers

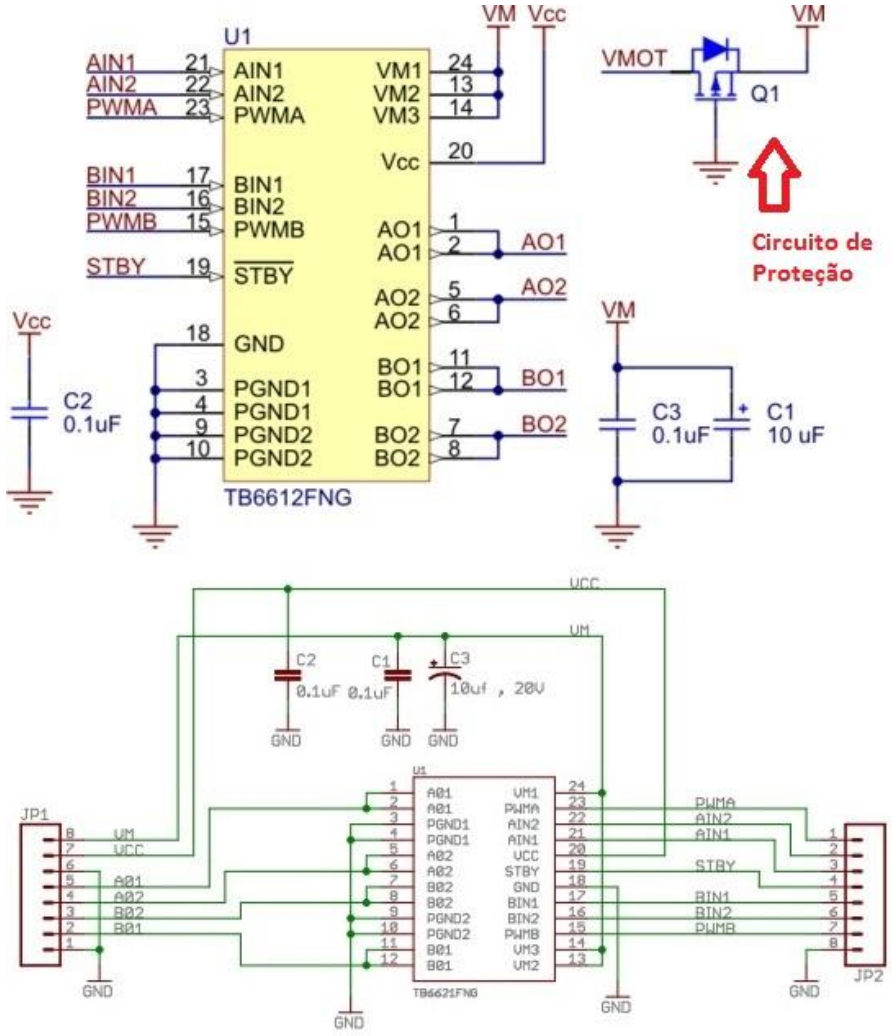
Os dois drivers são praticamente idênticos com relação programação necessária para utiliza-los, mas a pinagem das placas e um circuito de proteção que há no driver da Pololu e é inexistente no da Sparkfun.

A pinagem é diferente entre os dois driver conforme mostra a figura abaixo:



Vista inferior dos drivers - A esquerda driver da Sparkfun e a direita o da Pololu

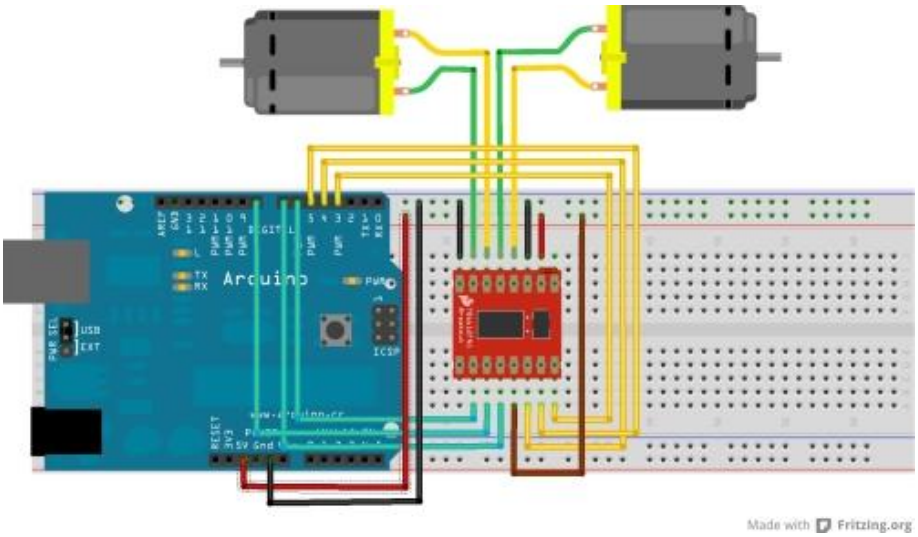
Diferença do circuito de proteção conforme esquemático abaixo:



Esquema elétrico dos Drivers - Na parte inferior driver da Sparkfun e na parte superior o da Pololu

O Circuito

Abaixo você pode ver o esquema de ligação do driver da Sparkfun com o Arduino para controlar o carrinho:



O Controle

Enviamos comandos via comunicação serial para o Arduino através do programa ***UTF-8 Tera Term*** (não utilizamos o Serial Monitor pois ele necessita do Enter para confirmar o envio). Quando as teclas **W, A, S, D** e **P** ou as teclas do teclado numérico **8, 4, 2, 6** e **5** são recebidas pelo Arduino, o mesmo executa as seguintes funções:

Comando Enviado Via Serial	Ação Do Carrinho
W ou 8	Frente
A ou 4	Esquerda
S ou 2	Trás
D ou 6	Direita
P ou 5	Parado

O Sketch

```
//Definições para controlar o Motor A
#define PWMA 3 //Controle de Velocidade
#define AIN1 4 //Modo de Operação
```

```
#define AIN2 5 //Modo de Operação
#define MOTOR_A 1 //Define motor da Direita com valor 1

//Definições para controlar o Motor B
#define PWMB  6 //Controle de Velocidade
#define BIN1  7 //Modo de Operação
#define BIN2  8 //Modo de Operação
#define MOTOR_B 2 //Define motor da Direita com valor 2

byte letra;
void setup()
{
  //Define os pinos de controle como pinos de saída
  pinMode(PWMA, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);

pinMode(PWMB, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
  //pinMode(13, OUTPUT);
  Serial.begin(9600);

}

void loop()
{
  if (Serial.available() > 0)
    {
    letra = Serial.read();
    switch(letra)
      {
      case '8'://FRENTE
      case 'w':
      case 'W':
        move(MOTOR_A, 255, 1); //Motor da Esquerda, para o Motor, sentido de rotação
        move(MOTOR_B, 255, 2); //Motor da Direita, a toda velocidade, sentido de rotação
        break;
```

```
case '4'://ESQUERDA
      case 'a':
      case 'A':
        move(MOTOR_A, 255, 2); //Motor da Esquerda, baixa velocidade, reverso
        move(MOTOR_B, 255, 2); //Motor da Direita, baixa velocidade, reverso
        break;
case '2'://TRÁS
      case 's':
      case 'S':
        move(MOTOR_A, 255, 2); //Motor da Esquerda, para o Motor, sentido de rotação
        move(MOTOR_B, 255, 1); //Motor da Direita, a toda velocidade, sentido de rotação
        break;
case '6'://DIREITA
      case 'd':
      case 'D':
        move(MOTOR_A, 255, 1); //Motor da Esquerda, para o Motor, sentido de rotação
        move(MOTOR_B, 255, 1); //Motor da Direita, a toda velocidade, sentido de rotação
        break;

      case '5'://PARADO
      case 'p':
      case 'P':
        move(MOTOR_A, 0, 2); //Motor da Esquerda, para o Motor, sentido de rotação
        move(MOTOR_B, 0, 1); //Motor da Direita, a toda velocidade, sentido de rotação
        break;
      }
    letra = Serial.read();
    letra = 0;
    }
}
void move(int motor, int speed, int direction)
{
```

```
  boolean inPin1 = LOW;
  boolean inPin2 = HIGH;

if(direction == 1)
  {
    inPin1 = HIGH;
    inPin2 = LOW;
  }

if(motor == 1)
  {
    digitalWrite(AIN1, inPin1);
    digitalWrite(AIN2, inPin2);
    analogWrite(PWMA, speed);
  }
  else
  {
    digitalWrite(BIN1, inPin1);
    digitalWrite(BIN2, inPin2);
    analogWrite(PWMB, speed);
  }
}
```

Então é isto pessoal!!! Espero que tenham gostado do Tutorial pois tem muita gente que procura um modo de controlar carrinhos pelo computador. Se tiverem dúvidas ou opiniões comentem abaixo.

\o/

Referências

https://code.google.com/p/fritzing/issues/detail?id=875

http://bildr.org/2012/04/tb6612fng-arduino/

http://www.pololu.com/catalog/product/713

https://www.sparkfun.com/products/9457

67. Tutorial: Bafômetro utilizando o Sensor de Gás (Álcool) com Arduino

Olá Garagistas! No tutorial de hoje iremos demonstrar como utilizar os sensores de gás e também como você pode fazer um Bafômetro caseiro utilizando-se um Sensor de Álcool, Arduino e LEDs!

Materiais Utilizados:

1x Arduino

1x Sensor de Álcool

1x Protoboard

7x LEDs

6x Resistores 330Ω

1x Resistor 10KΩ

Alguns Jumpers

1. Os Sensores de Gás

Imagem 1 - Sensores de Gás

Todos os sensores de gás fornecidos pela SparkFun têm a mesma pinagem, e se você quiser pode utilizar umBreakout para não ser necessário fazer as ligações diretamente nos pinos do sensor, a pinagem do sensor e como deve ser feitas as ligações no mesmo podem ser vistas na imagem abaixo.

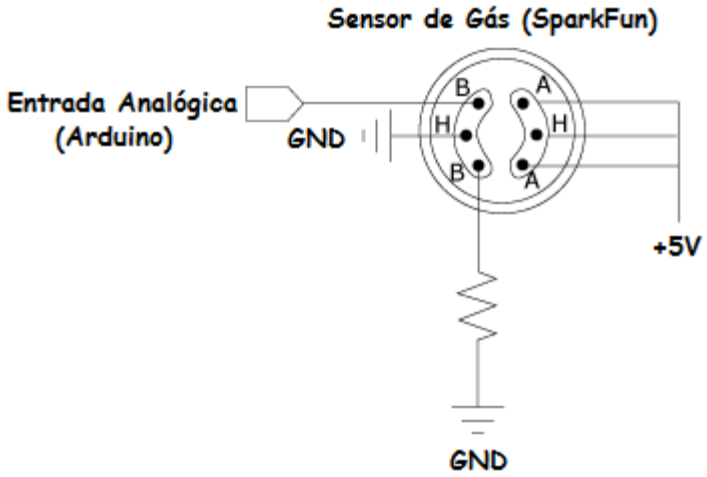


Imagem 2 - Pinagem Sensor de Gás

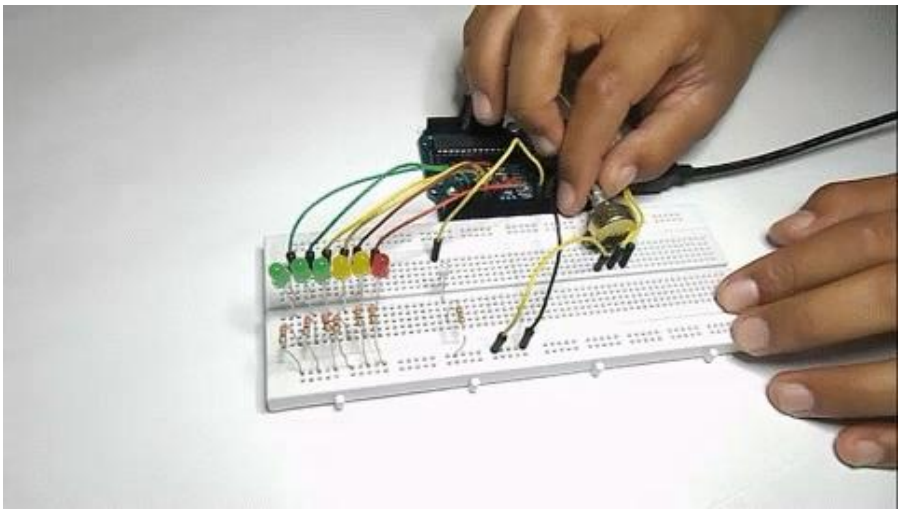
2. O Funcionamento

2.1) Sensor de Álcool:

O Sensor de Álcool irá funcionar como um potênciometro, só que ao invés de pegar a variação da resistência e transforma em valores decimais(0 a 1023) quando lido pelo Arduino, ele irá pegar a variação dependendo da concentração de Álcool que estiver em em seu sensor(0 a 1023).

2.2) LEDs (Bargraph):

Os 6 LEDs na aplicação vão ser utilizados para demonstrar a concentração de Álcool entregue pelo usuário, e eles vão acender indo do verde ao vermelho, passando pelo amarelo, dependendo da concentração de Álcool no sensor, como se fosse um Bargraph.



Animação 1 - Leitura do teor de álcool(simulada pelo potênciometro) demonstrada pelo Bargraph

2.3) Adaptações:

Nesse tutorial utilizaremos um antisséptico bucal(com álcool) para fazermos a demonstração da aplicação, e também reduzimos a variação de 0 a 1023 **para** 0 a 200 conforme a alteração abaixo:

"**map(sensor, 0, 1023, 0, qtdLED);**" **para** "**map(sensor, 0, 200, 0, qtdLED);**".

Para melhor calibração do sensor de álcool, tendo como aplicação um bafômetro mais preciso, siga este tutorial (em inglês).

3. O Sketch

```
#define qtdLED 6  //Declaração da quantidade de LEDs da aplicação
int LEDs[qtdLED]; //Vetor que guardará em cada posição o pino de saída para cada LED
int sensorPin=0; //Variável para guardar o valor lido pelo sensor
int tempo; //Variável para o tempo de leitura do sensor
int lidomaximo; //Variável para armazenar o maior valor lido durante a leitura
int i; //Variável de contagem
int j; //Variável de contagem

void setup()
{
  i=0;
  j=2;

  while(i < qtdLED) //Enquanto i for menor que a quantidade de LEDs que foi definida...
  {               //...guarda no vetor um valor(começando de 2 e incrementando) equivalente a um pino digital
    LEDs[i] = j;
    i++;
    j++;
  }

for(i=0;i<qtdLED;i++) //Define os pinos dos LEDs(nesse exemplo de 2 ao 7) como saída
  {
    pinMode(LEDs[i], OUTPUT);
```

```
  }
  pinMode(13, OUTPUT); //Define o pino 13 como saída para nos indicar quando pode fazer-se
o teste (LED Piscando)...
                    //...e quando o circuito estiver fazendo a leitura do sensor (LED Aceso)
}

void loop()
{
  PORTB = PORTB ^ 100000;  //Inverte o estado do pino digital 13 para dar o efeito de
Blink(Piscagem)
  delay(100); //Delay para efeito de blink do LED, indicado que o teste pode ser feito pelo
usuário
  int sensor = analogRead(sensorPin); //Lê o sensor e guarda na variável
  if(sensor >= 40) //Se a leitura for maior que 40 (valor escolhido para a demonstração
utilizando-se um...
  {               //...antisséptico bucal)
    digitalWrite(13, HIGH); //Acende o LED Azul(Indicando que o sensor detectou um mínimo de
álcool (sensor >= 40)
    lidomaximo = 0;         //Iniciar o valor máximo lido pelo sensor como 0
    for(tempo=0;tempo<=5000;tempo++) //Faz a leitura do sensor por 5 segundos...
    {                               //...a cada 1 milissegundo atualiza o maior valor lido pelo sensor.
      int sensor = analogRead(sensorPin);
      delay(1);
      if(sensor > lidomaximo)
      {
        lidomaximo = sensor;
      }
    }
    digitalWrite(13, LOW); //Após o termino da leitura, apaga o LED Azul
    int nivel = map(lidomaximo, 0, 200, 0, qtdLED); //Converte a escala de 0 a 200 do sensor
```

```
em...
                                        //...0 a 6(LEDs) e armazena na variável nível.
                                        //OBS: Lembrando o que o sensor lê de 0 a 1023, pelo
antisséptico...
                                        //...ter um teor de álcool relativamente baixo, foi utilizado...
                                        //...a escala de 0 a 200
   for(i=0;i<qtdLED;i++) //Compara todos os pinos dos LEDs com o valor convertido da escala...
   {                    //...Ex: Se meu nível convertido foi 5, então o os leds dos pinos 2 ao 6 irão
acender
     if (i < nivel) //Compara o pino atual da contagem, e se for menor que o nível máximo
convertido...
     {
       digitalWrite(LEDs[i], HIGH); //...Acende o LED
     }
     else //Compara o pino atual da contagem, e se for maior que o nível máximo convertido ...
     {
       digitalWrite(LEDs[i], LOW); //...Apaga o LED
     }
   }
   delay(10000); //Aguarda 10 segundos para o usuário conseguir fazer a leitura do bargraph
   for(i=0;i<qtdLED;i++) //Apaga todos os LEDs
   {
     digitalWrite(LEDs[i],LOW);
   }
 }
}
```

4. Montagem

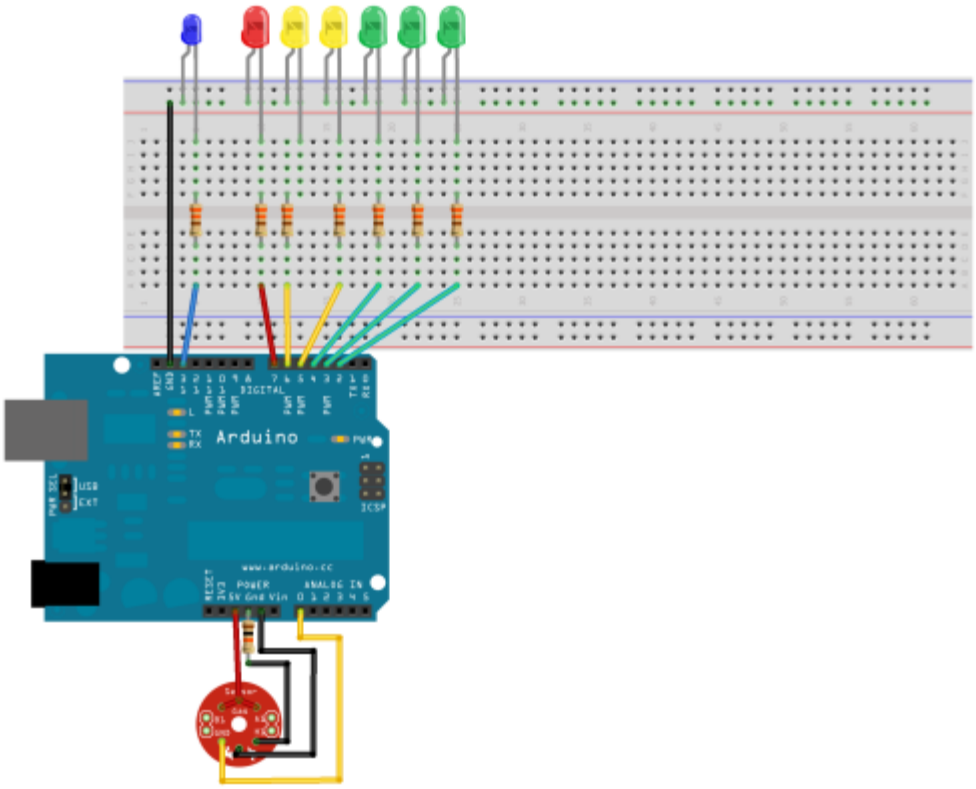


Imagem 3 - Montagem da Aplicação

Referências:

http://electronicpiece.blogspot.com.br/2012/08/arduino-bargraph.html

http://www.labdegaragem.org/loja/index.php/sensor-de-alcool-mq-3.html

68. Tutorial: Como Utilizar o XBee

Neste tutorial, vamos mostrar como utilizar o XBee, utilizando todos os componentes do XBee Wireless Kit Reail e o Arduino. Faremos três aplicações, em uma faremos uma demonstração como "Line Passing" entres os XBees, em outra um aplicação como "Leitura Analógica e Sensoriamento Remoto", e na última demonstração um "Link Serial" entres XBees, utilizando-se um Arduino para ligar ou desligar uma lâmpada.

Materiais Utilizados

1x Arduino

1x XBee Wireless Kit Retail

1x XBee Explorer

3x LEDs 5mm

3x Resistor 220Ω

3x Chave Tátil

1x LM35

1x Potenciômetros 10KΩ

2x Pilhas Lithium AA

1x Suporte Para Duas Pilhas

1x Multímetro

1x Ferro de Solda

1x Módulo Relé

1x Lâmpada 110V

2x Protoboard

Alguns Jumpers

1. ZigBee IEEE 802.15.4

Os módulos XBee, são módulos RF (Rádio Frequência) que fazem comunicações no padrão ZigBee IEEE 802.15.4. O Protocolo ZigBee permite comunicações robustas e opera na freqüência ISM (Industrial, Scientific and Medical), sendo aqui no Brasil 2,4 GHz (16 canais) e em outras partes do mundo, e não requerem licença para funcionamento.

As Redes ZigBee oferecem uma excelente imunidade contra interferências, e a capacidade de hospedar milhares de dispositivos numa Rede (mais que 65.000), com taxas de transferências de dados variando entre 20Kbps a 250Kbps. O Protocolo ZigBee é destinado a aplicações industriais, portanto, o fator velocidade não é crítico numa implementação ZigBee.

Os módulos RF padrão ZigBee foram criados para economizar o máximo de energia possível. Com isso, é possível criar aplicações onde é possível ler sensores em campo remotamente, apenas utilizando pilhas ou baterias comuns, que durarão meses ou mesmo anos sem precisarem ser substituídas. Isso porque, os módulos ZigBee quando não estão transmitindo/recebendo dados, entram num estado de dormência ou em "Sleep", consumindo o mínimo de energia.

Diversas aplicações podem ser feitas utilizando o XBee:

Automação Comercial de Edifícios:

-Segurança

-Ventilação

-Controle de Acesso

-Controle de Iluminação

-Aquecimento

Eletrodomésticos:

-TV

-VCR

-DVD/CD

-Controle Remoto

Hospitalar:

-Monitoramento de Pacientes

-Monitoramento Corporal

Computadores e Periféricos:

-Periféricos Para PC

-Mouse

-Teclado

-Joystick

Controle Industrial:

-Controle de Processo

-Gerenciamento de Energia

-Rastreamento de Equipamentos

Controle Residencial e Comercial:

-Segurança

-Ventilação

-Controle de Iluminação

-Controle de Acesso

-Irrigação de Jadim

-Aquecimento

2. Especificações do XBee

Performance:

- Rendimento da Potência de saída: 1 mW (0 dBm);

- Alcance em ambientes internos/zonas urbanas: 30m;

- Alcance de RF em linha visível para ambientes externos: 100m;

- Sensibilidade do receptor: -92 dBm;

- Freqüência de operação: ISM 2.4 GHz;

- Taxa de dados de RF: 250.000 bps;

- Taxa de dados da Interface (Data Rate): 115.200 bps;

Alimentação:

- Tensão de alimentação: 2.8 à 3.4B;

- Corrente de transmissão (típico): 45 mA @ 3.3V;

- Corrente de Recepção (típico): 50 mA @ 3.3V;

- Corrente de Power-down Sleep: <10 µA;

Propriedades Físicas:

- Dimensões: (2.438cm x 2.761cm);

- Peso: 0.10 oz (3g);

- Temperatura de operação: -40 to 85º C (industrial);

- Opções de antena: Conector U.FL RF, Chip ou Chicote (whip);

Rede:

- Topologia de Rede: Peer-to-peer(Par-a-par), ponto-a-ponto, ponto-a-multiponto e malha;

- Manipulação de erro: Retransmite novamente (Retries) & reconhecimento

(acknowledgements);

- Endereçamento: 65.000 endereços de rede disponíveis para cada canal;

- Opções de filtros: PAN ID, canais e endereços;

- Criptografia: 128-bit AES;

- Número de canais selecionáveis via software: 16 canais de seqüência direta;

Geral:

- Faixa de freqüência: 2.4000 - 2.4835 GHz;

Para informações mais específicas sobre o XBee, consulte o Datasheet ou o Manual.

3. Como Configurar o XBee

Para configurar o XBee, faça as conexões necessária utilizando-se um cabo Cabo USB (Tipo A para Mini B) e utilize o software X-CTU, disponível neste link. O X-CTU é um software criado pelo própio fabricando do XBee para enviar comandos de configuração, fazer atualizações e outra diversas ferramentas para você configurar seu XBee.

3.1) Conexões Para As Configurações

Conecte o XBEE no XBEE Explorer USB para configurá-lo, conforme a figura abaixo:

Imagem 1 - Conexões para as configurações

Conecte o XBEE Explorer USB no seu computador e instale o driver FTDI. Este driver pode ser encontrado dentro da pasta "drivers", localizado dentro da pasta da IDE do Arduino. Utilizando

um programa de comunicação serial como o X-CTU, você pode acessá-lo para configurá-lo via modo de comando.

3.2) Modo de Comando

Para se configurar o XBee, utilizaremos linhas de comando enviados pelo Terminal do própio X-CTU, existem diversos comandos, a relação de todos você encontrar no Datasheet ou no Manual do XBee. Para entrar em modo de comando digite no Terminal do X-CTU "+++", **sem** pressionar <Enter>, seu XBee deverá responder **OK**, para cada comando após entrar em modo de comando você deverá pressionar <Enter>.

Exemplo:

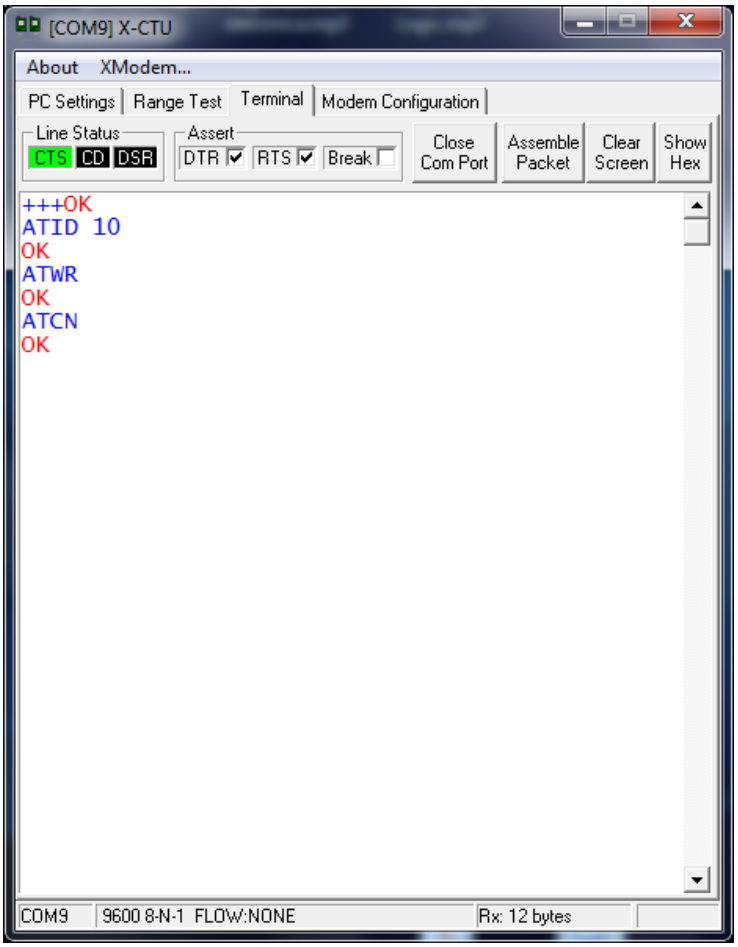


Imagem 2 - Terminal do X-CTU

No exemplo, entro em modo de comando digitando "+++" e **NÃO** clicao no <Enter>, e o XBee responde **OK**. Digito cada comando com o parâmetro desejado e depois clico em <Enter>:

ATID 10 <Enter>: Configuração da Rede (Rede 10).

ATWR <Enter>: Grava os comandos com os parâmetros desejados no XBee.

ATCN <Enter>: Sai do modo de comando.

Repare que em todos os comandos, digitando-os corretamente e colocando uma parâmetro aceitável o XBee responde**OK**. Em todas as configurações do XBee, é ideal você utilizar os comando ATWR e ATCN, para que o XBee guarde as configurações.

3.3) Demonstrações

Nas demonstrações de aplicação(Passos 4, 5 e 6), utilizaremos alguns comandos com certos parâmetros para cada aplicação feita ***neste tutorial***. Para ter acesso a todos os comandos disponíveis, com a descrição de cada um deles, consulte o Datasheet ou o Manual do XBee.

4. Line Passing

O Line Passing simplesmente reflete o estado de uma porta de entrada do XBee local à porta de saída do XBee remoto. Por exemplo, se colocarmos o estado lógico de uma Entrada Digital do transmissor em nível "1", a mesma Entrada Digital do receptor irá copiar o nível lógico "1".

4.1) As Configurações dos XBees

Transmissor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	2	Endereço do Módulo de Destino
ATMY	1	Endereço do Módulo de Local
ATNI	BOTOES	Nome do Módulo
ATD0	3	Configura D0 como Entrada Digital
ATD1	3	Configura D1 como Entrada Digital
ATD2	3	Configura D2 como Entrada Digital
ATIR	0	Define Amostragem
ATIT	1	Amostras Antes de Transmitir
ATIC	7	Monitora Mudança de Estado de D0 a D3
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Receptor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	1	Endereço do Módulo de Destino
ATMY	2	Endereço do Módulo de Local
ATNI	LEDS	Nome do Módulo
ATD0	4	Configura D0 como Saída Digital (Nível Lógico 0)
ATD1	4	Configura D1 como Saída Digital (Nível Lógico 0)
ATD2	4	Configura D2 como Saída Digital (Nível Lógico 0)
ATIU	1	Habilita Saída
ATIA	1	Recebe Alteração de Estado Somente do Endereço
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Tabela 1 - Configurações como Line Passing

4.2) As Ligações

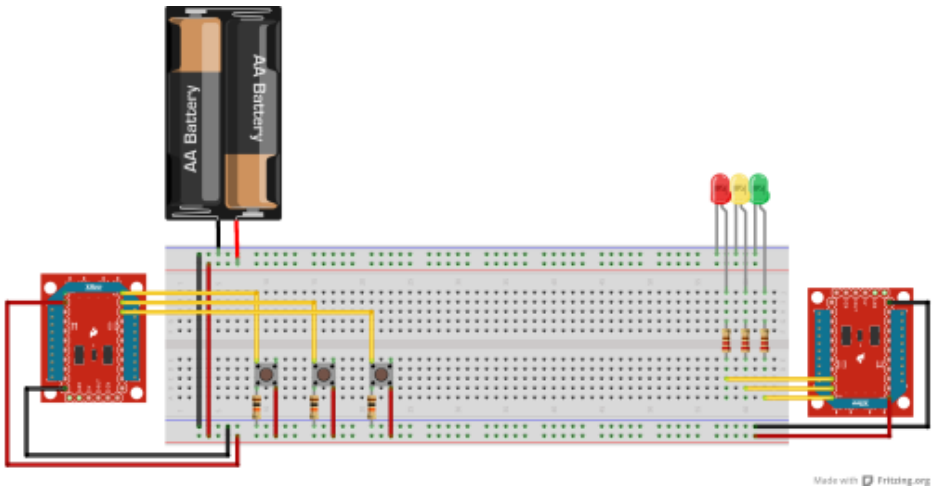


Figura 2 - Ligações como Line Passing

5. Leitura Analógica/Sensoriamento Remoto

Nesta aplicação você poder ler valores analógicos em um XBee transmissor, e enviar a um XBee receptor, por exemplo, você pode fazer sensoriamento remoto ou controlar servomotores, lembrando que cada XBee possui duas saídas PWM. Como demonstração, utilizaremos um LM35 para ler a temperatura e transmitir o valor em mV de um XBee a outro, e também controlaremos o a intensidade de brilho de um LED ligado a um XBee, por meio de um potenciômetro ligado a outro.

5.1) As Configurações dos XBees

Transmissor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	2	Endereço do Módulo de Destino
ATMY	1	Endereço do Módulo de Local
ATNI	ENTRADA	Nome do Módulo
ATD0	2	Configura D0 como Entrada Analógica
ATD1	2	Configura D1 como Entrada Analógica
ATIR	5	Define Amostragem
ATIT	5	Amostras Antes de Transmitir
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Receptor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	1	Endereço do Módulo de Destino
ATMY	2	Endereço do Módulo de Local
ATNI	SAIDA	Nome do Módulo
ATP0	2	Configura PWM0 como Saída PWM
ATP1	2	Configura PWM1 como Saída PWM
ATIU	1	Habilita Saída
ATIA	1	Recebe Alteração de Estado Somente do Endereço
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Tabela 2 - Configurações como Transmissão Analógica

5.2) As Ligações

- 5.2.1) Receptor (XBee + XBee Explorer + Resistência 220Ω + LED 5mm)

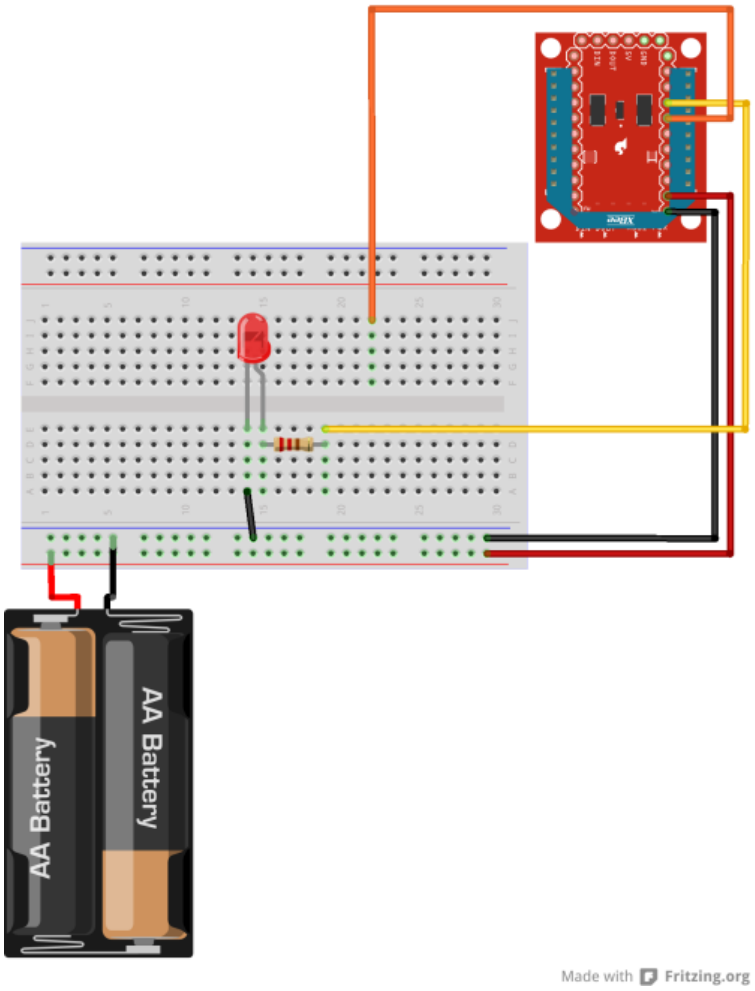


Figura 3 - Ligações do receptor analógico

- 5.2.2) Transmissor (XBee + XBee Explorer + LM35 + Resistência 10kΩ + Potenciômetro 10kΩ)

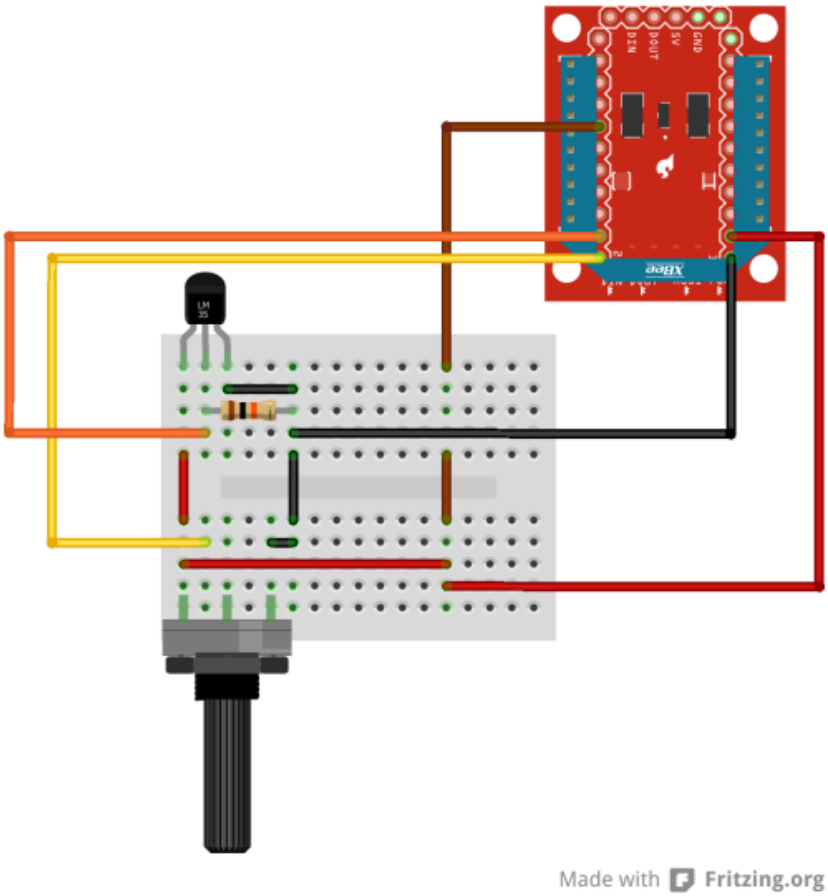


Figura 4 - Ligações do transmissor Analógico

6. Link Serial

Essa aplicação permite que você escreva na serial do transmissor, e os mesmos caracteres sejam escritos na serial do receptor, por exemplo, se na serial do transmissor eu escrever "Laboratório de Garagem", na serial do receptor vai ser escrito também "Laboratório de Garagem". Como uma demonstração, faremos que quando na serial do transmissor for escrito os caracteres "L" ou "D" (Liga/Desliga), o receptor receba esses caracteres, passe eles para o Arduino que fará o acionamento ou não de uma lâmpada com um auxílio de um módulo relé, ou seja, se o receptor receber "L", o Arduino ligará a lâmpada, se o receptor receber "D", o Arduino desligará a lâmpada.

6.1) As Ligações

- 6.1.1) Receptor (XBee + XBee Shield + Arduino)

Faça a montagem conforme a figura abaixo, conecte o cabo USB do Arduino ao seu computador, e identifique qual porta COM seu computador vai reconhecer o Receptor.

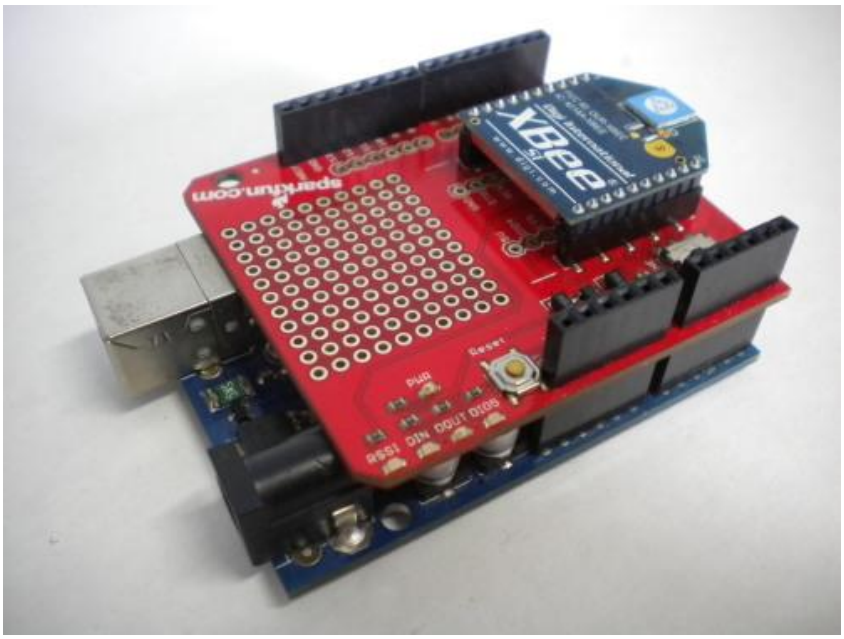


Imagem 3 - Montagem: XBee + XBee Shield + Arduino

Faça as ligações do módulo relé, utilizando os 5V e o GND do Arduino e pino IN, conecte ao pino digital D8.

- 6.1.2) Transmissor (XBee + XBee Explorer)

Conecter o XBee no XBee Explorer, conecte o cabo USB do XBee Explorer no seu computador, e identifique qual porta COM seu computador vai reconhecer o Transmissor.

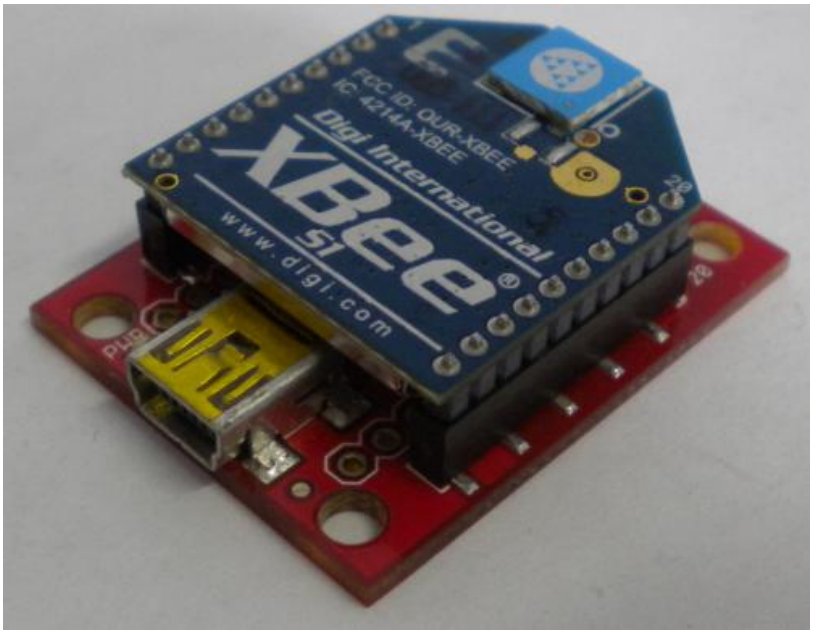


Imagem 4 - Montagem: XBee + XBee Explorer

6.2) As Configurações dos XBees

Transmissor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	2	Endereço do Módulo de Destino
ATMY	1	Endereço do Módulo de Local
ATNI	TRANSMISSOR	Nome do Módulo
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Receptor		
Comando	*Valor*	*Função*
ATID	10	Rede
ATDL	1	Endereço do Módulo de Destino
ATMY	2	Endereço do Módulo de Local
ATNI	RECEPTOR	Nome do Módulo
ATWR	-	Salva as Configurações
ATCN	-	Sai do Modo de Comando

Tabela 3 - Configurações como Link Serial

6.3) O Sketch

```
#define lamp 8

#include <SoftwareSerial.h>

SoftwareSerial mySerial(2, 3);

char lido;

void setup()

{

Serial.begin(9600);

mySerial.begin(9600);

pinMode(lamp,OUTPUT);

Serial.println("Este eh o Receptor");

Serial.println("Digite na Serial do Transmissor");

}

void loop()

{

if (mySerial.available())

{

  lido = mySerial.read();

  if(lido == 'l' || lido == 'L')

  {

   digitalWrite(lamp,HIGH);
```

```
   Serial.print("Lampada Acesa");
  }
  if(lido == 'd' || lido == 'D')
  {
   digitalWrite(lamp,LOW);
   Serial.print("Lampada Apagada");
  }
}
}
```

É isso ai Garagistas! Esperamos que tenham gostado deste Tutotial, até a próxima!

Referências

https://sites.google.com/site/toucatronic/zigbee/comandos-at

https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf

http://www.rogercom.com/ZigBee/ZigBee.htm

69. Tutorial: Utilizando o WiFiShield

Neste tutorial você verá como utilizar o WiFiShield em conjunto com o Arduino e a biblioteca WiflyHQ com a função de rodar uma página HTML, onde você pode fazer acionamentos (no caso um LED ligado ao pino 12), fazer a leitura dos pinos analógicos e também enviar o campo de texto para uma variável (através de um POST).

Lista de Materiais

1 x Arduino Uno Rev 3

1 x WiFiShield

1 x Resistor de 330Ω

1 x LED

Alguns jumpers

Biblioteca

WiflyHQmaster - Atualizada dia 03/07/2013

O Módulo RN171XV

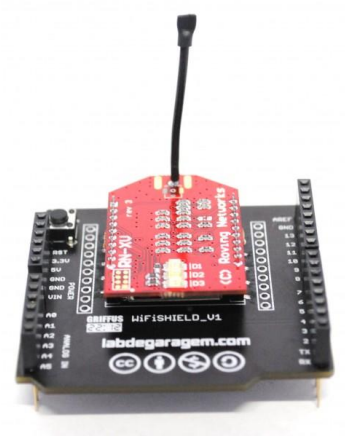
Este é o módulo RN-171XV que é fabricado pela Roving Networks que permite uma rápida conexão com redes 802.11 b/g (WiFi) e tem uma interface de comunicação serial por onde pode ser programado e trocar informações entre o módulo e o Arduino. Ele tem 8 pinos de I/O (3,3V) e 3 pinos para entrada de sensores analógicos (0 ~ 400mV com tolerância para 3,3V).

Ele suporta vários tipos de redes estruturas de rede com suporte aos protocolos DHCP, DNS, ARP, ICMP, FTP client, HTTP client, TCP, UDP. Suporta WPS (**W**i-fi **P**rotected **S**etup) com criptografias do tipo WEP, WAP e WAP2.

Há 3 LEDs que indicam o status do módulo veja na tabela abaixo:

	Apagado	Piscando Devagar	Piscando Rápido	Acesso
Verde	-	IP OK	Sem IP	Conectado via TCP
Amarelo	-	-	Transferindo Dados Pelo Tx	-
Vermelho	Conectado/ Internet OK	Conectado/ Sem Internet	Sem Conexão	-

O WiFiShield



Para facilitar a vida dos garagistas o Lab. desenvolveu o WiFiShield que permite uma rápida conexão deste módulo ao Arduino.

Você pode programar o módulo através do seu Arduino com as instruções contidas no manual do WiFiShield (desenvolvido pelo LdG) ou ainda, ver informações e detalhes sobre o módulo clicando aqui e aqui (neste você encontra o set de instruções).

Arduino como Server

Abaixo você pode ver a imagem da página gerada pelo Arduino:

Baseados no exemplo httpserver da biblioteca WiFlyHQmaster, criamos uma página HTTP dentro do Arduino, por onde, podemos controlar um LED conectado ao pino 12, além de poder visualizar os valores das entradas analógicas e também mandar um "olá"para você Garagista.

Para isto utilizamos as requisições GET e o POST para transferir informações entre o nosso computador e a página dentro do Arduino.

O Sketch

O Sketch por conter trechos com códigos HTML não será mostrado aqui mas você pode fazer o download dele neste link.

Conclusão

Com este tutorial mostramos a você pode utilizar o WiFiShield junto com a biblioteca WiFlyHQ. Esperamos que aprendam muito com este exemplo e se tiverem dúvidas ou sugestões deixem seus comentário logo abaixo.

\o/

Referências

http://www.rovingnetworks.com/products/RN171XV

http://labdegaragem.com.br/loja/wifishield.pdf

https://github.com/harlequin-tech/WiFlyHQ

70. Tutorial: Controlando o MP3 Player Shield por comandos via Serial

Neste tutorial veremos como utilizar o MP3 Player Shield junto com as biblioteca SFEMP3Shield e a SdFat para ler arquivos no formato MP3 armazenados no catão SD e reproduzi-los em uma caixa de som conectada a saída de audio do Shield. Tudo isto sendo controlado por comandos via Serial.

Lista de Materiais

1 x Arduino Uno Rev 3

1 x MP3 Player Shield

1 x Cartão miniSd

1 x Caixa de som ou fones de ouvido com entrada P2

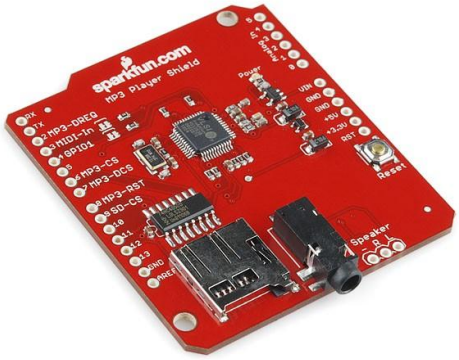
Bibliotecas

Você vai precisar da biblioteca SFEMP3Shield que implementa os comandos para controle do MP3 Player Shield e da SdFat (inclusa no pacote) para fazer a leitura do cartão de memória.

Abaixo você pode fazer o download das duas bibliotecas usadas no tutorial.

Link para download

O MP3 Player Shield



Este Shield é baseado no chip VS1053B, fabricado pela VLSI Solution.

Com este chip você pode decodificar vários formatos de áudio como Ogg Vorbis, MP3, AAC, WMA e MIDI e envia tudo para um amplificador de saída *(P2)* estéreo de alta qualidade que esta conectado a um DAC de 18bits de resolução.

Ele também pode ser usado como encoder para o formato IMA ADPCM e Ogg Vorbis.

***OBS**: O MP3 Player Shield não suporta as funções de encoder*

Veja mais informações sobre o chip no datasheet.

Compatibilidade com arquivos de Audio, tag ID3 e Plugins

No capitulo 8 do datasheet, é abordada a compatibilidade dele com diversos arquivos de audio.

No tutorial, utilizamos arquivos comprimidos no formato MP3 *(MPEG-1/2 Audio Layer 3)* e com bit rate de 192kbps.

Há também a compatibilidade do chip com as tag ID3 da versão 1.0 ou 1.1, com elas você pode visualizar o *Nome da música*, *Artista* e o *Álbum* apenas utilizando apenas comandos já implementados pela biblioteca. Para alterar a tag ID3 de suas músicas vocês podem utilizar o ID3 Tag Editor.

A VLSI Solutions também disponibiliza patch e plugins que são atualizados constantemente no neste link e você pode adicionar funcionalidades ao firmware do chip carregando estes patchs no cartão SD (A biblioteca traz eles para você).

O Sketch

Fizemos um Sketch baseado no exemplo que vem junto com a biblioteca SFEMP3Shield. Com ele através de comandos enviados via serial você pode controlar as músicas que serão tocadas.

```
// PLAYER MP3 LAB DE GARAGEM
// Baseado no exemplo MP3Shield_Library_Demo de Bill Porter e Michael P. Flaga criadores d
a biblioteca SFEMP3Shield
```

```
// Coloque as músicas no SD card para com o nome track00x.mp3 onde x é o numero da música

#include <SPI.h>
#include <SdFat.h>
#include <SFEMP3Shield.h>

SdFat sd;

SFEMP3Shield MP3player;

void setup()
{
  //Variável usada para testar funções durante a programação
  uint8_t resultado;

// Inicializa a Serial
  Serial.begin(115200);

//Inicializando Cartao SD
  if(!sd.begin(SD_SEL, SPI_HALF_SPEED)) sd.initErrorHalt();
  if(!sd.chdir("/")) sd.errorHalt("sd.chdir");

//Inicializando MP3 Shield e checa por erros
  resultado = MP3player.begin();

//Lista de código de Erros de inicialização
  //0 OK
  //1 Falha da SdFat para inicializar o contato físico com o Cartão SD
  //2 Falha SdFat para iniciar o volume do Cartão SD
  //3 Falha SdFat para montar o diretório raiz Cartão SD
  //4 Valores fora do padrão no registrador SCI_MODE.
  //5 Valores divergentes no na leitura do SCI_CLOCKF.
  //6 Plugins não foram carregado. Copie eles para o Cartão SD isto pode causar erros na reprodução

if(resultado != 0)
  {
    Serial.print(F("Codigo de erro: "));
    Serial.print(resultado);
```

```
    Serial.println(F(" ao tentar tocar a musica"));
    if( resultado == 6 )
    {
      Serial.println(F("Atencao: Plugins não encontrados."));
      Serial.println(F("Use a opção \"d\" para verificar se eles estão no cartao SD"));
    }
  }
  comandos();
}

void loop()
{
  if(Serial.available())
  {
    menu(Serial.read()); // Pega o comando da Serial e processa com a função menu();
  }
  delay(100);
}

// Função que processa o comando.
void menu(byte opcao)
{
  uint8_t resultado; // Esta variavel vai armazenar o resultado dos funções que vamos usar
  char titulo[30]; // Buffer para armazenas título da musica
  char artista[30]; // Buffer para armazenas nome do Artista
  char album[30]; // Buffer para armazenas nome do Album

Serial.print(F("Comando Recebido: "));
  Serial.write(opcao);
  Serial.println();

//Parar a música que esta tocando
  if(opcao == 's')
  {
    Serial.println(F("Parar reproducao"));
    MP3player.stopTrack();
  }
```

```
  //Seleciona a faixa de 1 a 9
  else if(opcao >= '1' && opcao <= '9')
  {
    opcao = opcao - 48; //convertendo ASCII para um numero real
    resultado = MP3player.playTrack(opcao);

  //Lista de código de Erros da função playTrack
  //0 OK
  //1 Ocupado tocando música
  //2 Arquivo não encontrado
  //3 Indica que o Shield MP3 esta resetando
//Verifica esta executando
    if(resultado != 0)
    {
      Serial.print(F("Codigo de Erro: "));
      Serial.print(resultado);
      Serial.println(F(" ao tentar tocar a musica"));
    }
    else
    {
      //Retira a Tag ID3 da música
      MP3player.trackTitle((char*)&titulo);
      MP3player.trackArtist((char*)&artista);
      MP3player.trackAlbum((char*)&album);
//Imprime as informações na Serial
      Serial.print(F("Titulo: "));
      Serial.write((byte*)&titulo, 30);
      Serial.println();
      Serial.print(F("Artista: "));
      Serial.write((byte*)&artista, 30);
      Serial.println();
      Serial.print(F("Album: "));
```

```
      Serial.write((byte*)&album, 30);
      Serial.println();
    }
  }
//Altera o Volume
  else if((opcao == '-') || (opcao == '+'))
  {
    union twobyte mp3_vol; // Cria dois bytes um para cada lado
    mp3_vol.word = MP3player.getVolume(); // Lê o valor atual do Volume
if(opcao == '-')
    {
      // O saida é negativa pois é dada em decibéis
      if(mp3_vol.byte[1] >= 254)
      {
        mp3_vol.byte[1] = 254;
      }
      else {
        mp3_vol.byte[1] += 2;
      }
    }
    else
    {
      if(mp3_vol.byte[1] <= 2)
      {
        mp3_vol.byte[1] = 2;
      }
      else {
        mp3_vol.byte[1] -= 2;
      }
    }
    MP3player.setVolume(mp3_vol.byte[1], mp3_vol.byte[1]); // Altera o valor do volume nos dois
lados (L e R)
    Serial.print(F("Volume foi para -"));
```

```
    Serial.print(mp3_vol.byte[1]>>1, 1);
    Serial.println(F("[dB]"));
  }
//Lista arquivos do SD card
  else if(opcao == 'd') {
    if(!MP3player.isPlaying())
    {
      Serial.println(F("Arquivos encontrados:"));
      sd.ls(LS_R | LS_DATE | LS_SIZE);
    }
    else {
      Serial.println(F("Player ocupado. Aguarde..."));
    }
  }
//Informacoes sobre o arquivo de audio
  else if(opcao == 'i') {
    MP3player.getAudioInfo();
  }
//Modo Pause/Play
  else if(opcao == 'p')
  {
    if( MP3player.getState() == playback)
    {
      MP3player.pauseMusic();
      Serial.println(F("Pause"));
    }
    else if( MP3player.getState() == paused_playback)
    {
      MP3player.resumeMusic();
      Serial.println(F("Play"));
    }
    else
    {
```

```
      Serial.println(F("Nao esta tocando!"));
    }
  }
// Reseta o módulo MP3
  else if(opcao == 'r') {
    MP3player.stopTrack();
    MP3player.vs_init();
    Serial.println(F("Resetando o chip do Player"));
}
// Seleciona modo Mono/Stereo
  else if(opcao == 'm') {
    uint16_t monostate = MP3player.getMonoMode();
    if(monostate == 0) {
      MP3player.setMonoMode(1);
      Serial.println(F("Modo Mono"));
    }
    else {
      MP3player.setMonoMode(0);
      Serial.println(F("Modo Stereo"));
    }
  }
//Modo baixo consumo
  else if(opcao == 'a') {
    MP3player.end();
    Serial.println(F("Modo de baixo consumo: Ativado"));
  }
  else if(opcao == 'd') {
    MP3player.begin();
    Serial.println(F("Modo de baixo consumo: Desativado"));
//Estado do Shield MP3
  }
  else if(opcao == 'S') {
    Serial.println(F("Estado do Player:"));
```

```
    switch (MP3player.getState()) {
    case uninitialized:
      Serial.print(F("Nao inicializado"));
      break;
    case initialized:
      Serial.print(F("Inicializado"));
      break;
    case deactivated:
      Serial.print(F("Desativado"));
      break;
    case loading:
      Serial.print(F("Carregando"));
      break;
    case ready:
      Serial.print(F("Pronto"));
      break;
    case playback:
      Serial.print(F("Reproduzindo"));
      break;
    case paused_playback:
      Serial.print(F("Pausado"));
      break;
    }
    Serial.println();
  }
// Lista comandos
  else if(opcao == 'c')
  {
    comandos();
  }
// print prompt after key stroke has been processed.
  Serial.println(F("Digite uma das opcoes 1-9,s,d,+,-,i,p,r,m,a,d,S,c:"));
```

```
  Serial.println();
}

//Imprime a lista de comandos
void comandos()
{
  Serial.println(F("MP3 Player Shield:"));
  Serial.println();
  Serial.println(F("LAB DE GARAGEM "));
  Serial.println();
  Serial.println(F("Lista de comandos:"));
  Serial.println(F(" [1-9] para tocar o arquivo track00x"));
  Serial.println(F(" [s] Stop"));
  Serial.println(F(" [d] Lista arquivos do SD card"));
  Serial.println(F(" [+ ou -] Aumenta e diminui o Volume"));
  Serial.println(F(" [i] Informacoes sobre o arquivo de audio"));
  Serial.println(F(" [p] Pause/Play"));
  Serial.println(F(" [r] Reseta MP3 Shield"));
  Serial.println(F(" [m] Seleciona modo Mono/Stereo"));
  Serial.println(F(" [a} Ativa modo de baixo consumo"));
  Serial.println(F(" [d} Desativa modo de baixo consumo"));
  Serial.println(F(" [S] Estado do Shield MP3"));
  Serial.println(F(" [c] Mostra lista de comandos"));
  Serial.println();
}
```

Conclusão

Com este tutorial você poderá tocar suas músicas através de um Arduino em qualquer projeto e de maneira simples, já que a biblioteca faz boa parte do trabalho para você. Esperamos que tenham gostado e se tiverem sugestões, dúvidas ou elogios comente abaixo.

\o/

Referências:

http://www.vlsi.fi/en/products/vs1053.html

https://www.sparkfun.com/products/10628

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Dev/Arduino/Shields/MP3%20Shield-v13.pdf

http://www.billporter.info/2012/01/28/sparkfun-mp3-shield-arduino-library/

71. Tutorial: Como utilizar o LCD Shield I²C

Neste tutorial, vamos mostrar como utilizar o LCD Shield I²C. Faremos demonstrações de utilização, onde a leitura dos botões do shield é feita para executar uma certa função, como nos exemplos: ajustar o brilho, ajustar o contraste, criar menus e fazer acionamento de um relé.

Material Utilizado:

1x LCD Shield I²C

1x Arduino UNO

1x Módulo Rele

Alguns Jumpers

Lâmpada 110V

Biblioteca utilizada:

Utilizaremos a biblioteca LiquidCrystal_I2C.

1. LCD Shield I²C

Baseado no consagrado DataShield, projeto de Jiri Trnka incubado no LdG, o LCD Shield I²C é uma excelente opção para projetos que necessitem de baixo custo e uma eficiente interface para o usuário. Utilizando bibliotecas prontas e toda a versatilidade do PCF8574, o LCD Shield I²C permite que sejam utilizados poucos pinos do Arduino para oferecer um LCD com backlight e contraste controlados por software, além de 5 botões configuráveis.

O LCD Shield I²C é um shield para Arduino que você poderá utilizar para fazer diversas aplicações, o shield possui:

- *1 x Display LCD 16x2*: Onde você pode escrever em sua tela diversos tipos de mensagens.

- *1 x CI PCF8574*: Responsável por enviar os comando I²C para o Display LCD, quando utilizado a biblioteca "LCD LiquidCrystal_I2C.h".

- *6 x Chaves Táteis*: 5 configuráveis (Pino A0) e 1 para RESET.

- *Headers Para Arduino*: Além de poder simplesmente acoplar em seu Arduino, permite que você utilize os pinos não utilizado do shield para outras finalidades.

O esquemático do Shield pode ser visto na imagem abaixo, você pode adquirir o PDF do esquemático, clicando aqui.

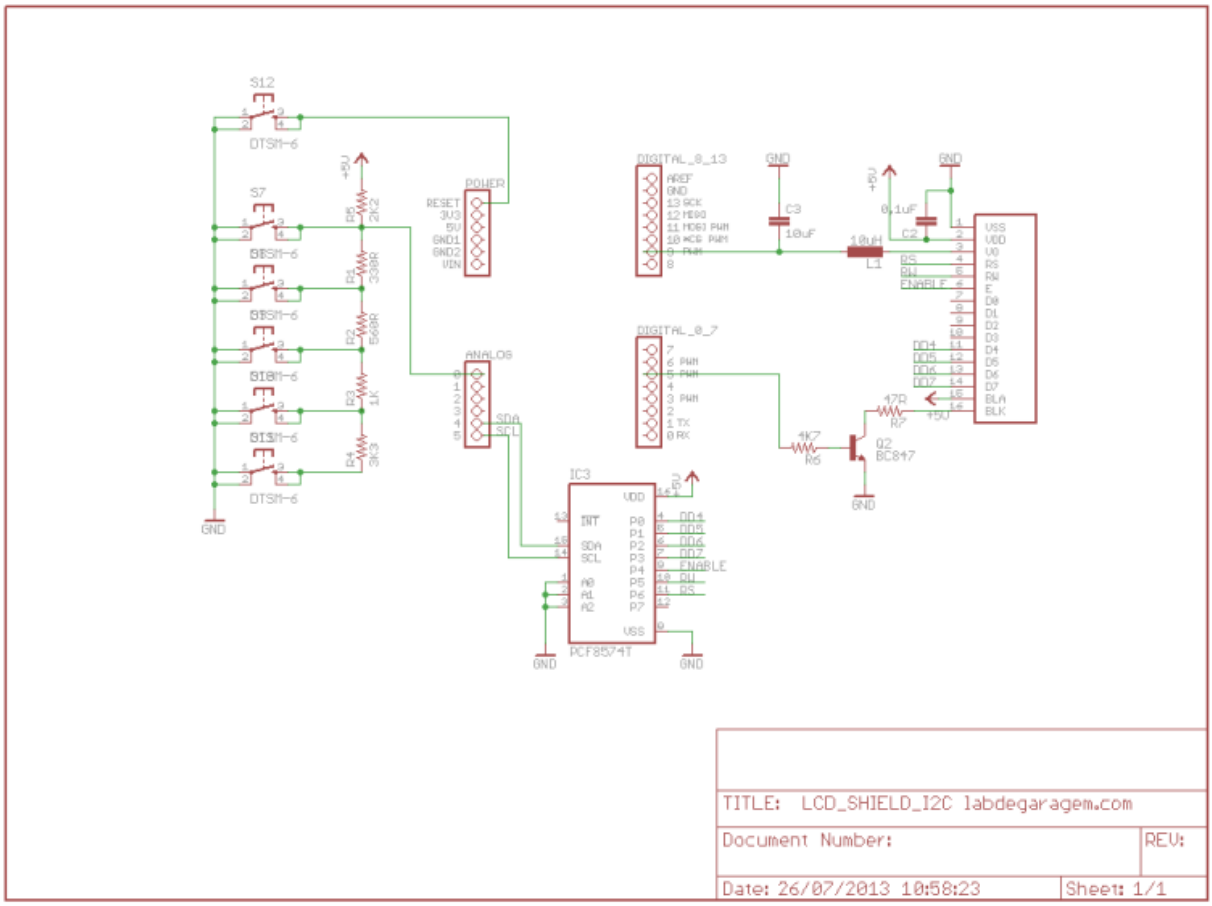


Figura 1 - Esquemático LCD Shield I²C

Como pode ser visto no esquemático acima, o shield utiliza apenas os pinos **A0**, **A4**, **A5** dos pinos analógicos e os pinos **D5** e **D9** dos pinos digitais, com isso todos os outros pinos do Arduino ficam livres, e com eles você pode acoplar outros shields que utilizem esses pinos que ficam livres, fazer acionamentos, utilizar módulos de comunicação como XBee ou Wifly, entre outra diversas aplicações. Também na placa você vai encontrar uma chave tátil para RESET.

2. Sketch das Demonstrações

Nas demonstrações disponível no vídeo, explicamos como fazer a leitura dos botões, como utilizar esse botões para fazer o ajuste do brilho do backligh, ajuste do contraste e também

como criar menus, e com exemplo criamos um menu "Ajuste Brilho", um menu "Ajuste Contraste" e um menu para fazer um simples acionamento de relé.

2.1) Leitura dos Botões

Nessa demonstração, fazemos a leitura de cada botão e escrevemos no Serial Monitor o valor analógico lido pelo Arduino através do pino (A0).

```
#define pinBRILHO 5 //Define como 5 a palavra pinBRILHO
#define pinCONTRASTE 9 //Define como 9 a palavra pinCONTRASTE

#include <Wire.h> //Inclui a biblioteca Wire.h para se utilizar a comunicação I2C
#include <LiquidCrystal_I2C.h> //Inclui a biblioteca LiquidCrystal_I2C.h

                    //para se fazer comunicação I2C com o display LCD
LiquidCrystal_I2C lcd(32,16,2); //Inicia o LCD com os parâmetros (Endereço I2C, Colunas, Linhas)

int botoes; //Variável para a leitura dos botões
int i; //Variável para contagem

float pwm_brilho = 255; //Variável para controle de pwm do brilho
float pwm_contraste = 0; //Variável para controle de pwm do contraste
float brilho_porcent; //Variável para conversão do pwm do brilho em porcentagem (255 = 100%)
float contraste_porcent; //Variável para conversão do pwm do contraste em porcentagem (255 = 100%)

void setup()
{
  Serial.begin(9600); //Inicia a Serial para se utilizar o Serial Monitor

  lcd.init(); //Comando para inicialização do Display LCD
  lcd.backlight(); //Comando para inicialização do Display LCD

  pinMode(pinBRILHO,OUTPUT); //Configura como saída o pino 5 (pinBRILHO)
  pinMode(pinCONTRASTE,OUTPUT); //Configura como saída o pino 9 (pinCONTRASTE)
```

```
  inicializacao(); //Executa a função da tela de início
}

void loop()
{
  botoes=analogRead(A0); //Faz a leitura analógica do pino A0 e guarda na variável botoes
  Serial.println(botoes); //Escreve no Serial Monitor o valor ligo pelo pino A0
  delay(100); //Delay de 100 milissegundos
  ajuste();
}

void inicializacao() //Função de inicialização da tela LCD para a aplicação
{
  analogWrite(pinBRILHO,255); //Inicializa o brilho no máximo (255 = 100%)
  analogWrite(pinCONTRASTE,0); //Inicializa o contraste no mínimo (0 = 0%)

  Serial.println(""); //Pula uma linha
  Serial.println("LCD SHIELD I2C: INICIADO" ); //Escreve no Serial Monitor

  lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
  lcd.print(" LCD Shield I2C"); //Escreve no LCD

  for(i=0;i<=4;i++) //Laço para a o efeito de piscagem da segunda linha
  {
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("               "); //Limpa segunda linha
    delay(250); //Permanece por 250 milissegundos apagada
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("labdegaragem.com"); //Escreve na tela do LCD a mensagem "labdegaragem.com"
    delay(250); //Permanece por 250 milissegundo acesa
  }
}

void ajuste()
{
```

```
  if(botoes > 100 && botoes < 200) //Se botão da esquerda for pressionado
  {
    if(pwm_brilho > 0) //Se o PWM do brilho estiver maior que 0
    {
    pwm_brilho -= 25.5; //Subtrai 25.5(10%) no PWM do brilho
    analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
    }
brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
                                         //o valor do PWM convertido em porcentagem
    Serial.print("BRILHO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
    Serial.print(brilho_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }

  else if(botoes > 700 && botoes < 800) //Senão se o botão da direita for pressionado
  {
    if(pwm_brilho < 255) //Se o PWM do brilho estiver menor que 255
    {
    pwm_brilho += 25.5; //Soma 25.5(10%) no PWM do brilho
    analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
    }

    brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
                                         //o valor do PWM convertido em porcentagem
    Serial.print("BRILHO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
    Serial.print(brilho_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }

  else if(botoes > 200 && botoes < 300) //Senão se o botão de baixo for pressionado
  {
    if(pwm_contraste > 0) //Se o PWM do contraste estiver maior que 0
```

```
    {
    pwm_contraste -= 25.5; //Subtrai 25.5(10%) no PWM do contraste
    analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
    }

    contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
                                        //o valor do PWM convertido em porcentagem
    Serial.print("CONTRASTE = "); //Escreve no Serial Monitor "CONSTRASTE =", sem pular
linha
    Serial.print(contraste_porcent); //Escreve no Serial Monitor o valor em porcentagem do
contraste
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }

  else if(botoes > 400 && botoes < 500) //Senão se o botão de cima for pressionado
  {
    if(pwm_contraste < 255) //Se o PWM do contraste estiver menor que 255
    {
    pwm_contraste += 25.5; //Soma 25.5(10%) no PWM do contraste
    analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
    }

    contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
                                        //o valor do PWM convertido em porcentagem
    Serial.print("CONTRASTE = "); //Escreve no Serial Monitor "CONSTRASTE =", sem pular
linha
    Serial.print(contraste_porcent); //Escreve no Serial Monitor o valor em porcentagem do
contraste
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
}
```

```
void trata_botao()
{
  while(botoes != 1023)
  {
    botoes=analogRead(A0);
  }
}
```

2.2) Ajuste de Brilho e Contraste

Nessa demonstração, mostramos como fazer o ajuste de brilho e contraste do Display LCD via software, utilizando os botões do shield.

```
#define pinBRILHO 5 //Define como 5 a palavra pinBRILHO
#define pinCONTRASTE 9 //Define como 9 a palavra pinCONTRASTE

#include <Wire.h> //Inclui a biblioteca Wire.h para se utilizar a comunicação I2C
#include <LiquidCrystal_I2C.h> //Inclui a biblioteca LiquidCrystal_I2C.h

                        //para se fazer comunicação I2C com o display LCD
LiquidCrystal_I2C lcd(32,16,2); //Inicia o LCD com os parâmetros (Endereço I2C, Colunas, Linhas)

int botoes; //Variável para a leitura dos botões
int i; //Variável para contagem

float pwm_brilho = 255; //Variável para controle de pwm do brilho
float pwm_contraste = 0; //Variável para controle de pwm do contraste
float brilho_porcent; //Variável para conversão do pwm do brilho em porcentagem (255 = 100%)
float contraste_porcent; //Variável para conversão do pwm do contraste em porcentagem (255 = 100%)

void setup()
{
  Serial.begin(9600); //Inicia a Serial para se utilizar o Serial Monitor
```

```
  lcd.init(); //Comando para inicialização do Display LCD
  lcd.backlight(); //Comando para inicialização do Display LCD

  pinMode(pinBRILHO,OUTPUT); //Configura como saída o pino 5 (pinBRILHO)
  pinMode(pinCONTRASTE,OUTPUT); //Configura como saída o pino 9 (pinCONTRASTE)

  inicializacao(); //Executa a função da tela de início
}
void loop()
{
  botoes=analogRead(A0); //Faz a leitura analógica do pino A0 e guarda na variável botoes
  Serial.println(botoes); //Escreve no Serial Monitor o valor ligo pelo pino A0
  delay(100); //Delay de 100 milissegundos
  ajuste();
}
void inicializacao() //Função de inicialização da tela LCD para a aplicação
{
  analogWrite(pinBRILHO,255); //Inicializa o brilho no máximo (255 = 100%)
  analogWrite(pinCONTRASTE,0); //Inicializa o contraste no mínimo (0 = 0%)

  Serial.println(""); //Pula uma linha
  Serial.println("LCD SHIELD I2C: INICIADO" ); //Escreve no Serial Monitor

  lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
  lcd.print(" LCD Shield I2C"); //Escreve no LCD

  for(i=0;i<=4;i++) //Laço para a o efeito de piscagem da segunda linha
  {
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("               "); //Limpa segunda linha
    delay(250); //Permanece por 250 milissegundos apagada
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("labdegaragem.com"); //Escreve na tela do LCD a mensagem "labdegaragem.com"
```

```
    delay(250); //Permanece por 250 milissegundo acesa
  }
}

void ajuste()
{
  if(botoes > 100 && botoes < 200) //Se botão da esquerda for pressionado
  {
    if(pwm_brilho > 0) //Se o PWM do brilho estiver maior que 0
    {
    pwm_brilho -= 25.5; //Subtrai 25.5(10%) no PWM do brilho
    analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
    }

brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
                                         //o valor do PWM convertido em porcentagem
    Serial.print("BRILHO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
    Serial.print(brilho_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }


  else if(botoes > 700 && botoes < 800) //Senão se o botão da direita for pressionado
  {
    if(pwm_brilho < 255) //Se o PWM do brilho estiver menor que 255
    {
    pwm_brilho += 25.5; //Soma 25.5(10%) no PWM do brilho
    analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
    }


    brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
                                             //o valor do PWM convertido em porcentagem
    Serial.print("BRILHO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
    Serial.print(brilho_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
```

```
  trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
 }

 else if(botoes > 200 && botoes < 300) //Senão se o botão de baixo for pressionado
 {
  if(pwm_contraste > 0) //Se o PWM do contraste estiver maior que 0
  {
  pwm_contraste -= 25.5; //Subtrai 25.5(10%) no PWM do contraste
  analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
  }

  contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
                                  //o valor do PWM convertido em porcentagem
  Serial.print("CONTRASTE = "); //Escreve no Serial Monitor "CONSTRASTE =", sem pular
linha
  Serial.print(contraste_porcent); //Escreve no Serial Monitor o valor em porcentagem do
contraste
  Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
  trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
 }

 else if(botoes > 400 && botoes < 500) //Senão se o botão de cima for pressionado
 {
  if(pwm_contraste < 255) //Se o PWM do contraste estiver menor que 255
  {
  pwm_contraste += 25.5; //Soma 25.5(10%) no PWM do contraste
  analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
  }

  contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
                                  //o valor do PWM convertido em porcentagem
  Serial.print("CONTRASTE = "); //Escreve no Serial Monitor "CONSTRASTE =", sem pular
linha
  Serial.print(contraste_porcent); //Escreve no Serial Monitor o valor em porcentagem do
```

```
contraste
    Serial.println("%"); //Escreve no Serial Monitor "%" e pula linha
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
}

void trata_botao()
{
  while(botoes != 1023)
  {
    botoes=analogRead(A0);
  }
}
```

2.3) Criação de Menu / Acionamento de Relé

Nessa demonstração, mostramos como você pode criar um Menu e para cada Menu fazer um tipo de tratamento diferente, como no exemplo, ajustar o brilho do backlight, contraste e fazer um acionamento de relé.

```
#define pinBRILHO 5    //Define como 5 a palavra pinBRILHO
#define pinCONTRASTE 9 //Define como 9 a palavra pinCONTRASTE
#define pinRELE 7      //Define como 7 a palavra pinRELE

#include <Wire.h> //Inclui a biblioteca Wire.h para se utilizar a comunicação I2C
#include <LiquidCrystal_I2C.h> //Inclui a biblioteca LiquidCrystal_I2C.h
                               //para se fazer comunicação I2C com o display LCD

LiquidCrystal_I2C lcd(32,16,2); //Inicia o LCD com os parâmetros (Endereço I2C, Colunas, Linhas)

int botoes; //Variável para a leitura dos botões
int i; //Variável para contagem

int menu_cont; //Variável para contagem dos menus
int sel=0; //Variável para verificar estado do botão selecionar
int estadoRELE=0; //Variável para alteração do estado do Relé (Relé DESLIGADO)
```

```
float pwm_brilho = 255; //Variável para controle de pwm do brilho
float pwm_contraste = 0; //Variável para controle de pwm do contraste
float brilho_porcent; //Variável para conversão do pwm do brilho em porcentagem (255 = 100%)
float contraste_porcent; //Variável para conversão do pwm do contraste em porcentagem (255
= 100%)

void setup()
{
  Serial.begin(9600); //Inicia a Serial para se utilizar o Serial Monitor
lcd.init(); //Comando para inicialização do Display LCD
  lcd.backlight(); //Comando para inicialização do Display LCD
pinMode(pinBRILHO,OUTPUT); //Configura como saída o pino 5 (pinBRILHO)
  pinMode(pinCONTRASTE,OUTPUT); //Configura como saída o pino 9 (pinCONTRASTE)
pinMode(pinRELE,OUTPUT); //Configura como saída o pino 7
  digitalWrite(pinRELE,estadoRELE); //Relé = DESLIGADO (estadoRELE=0);
inicializacao(); //Executa a função da tela de início
}
void loop()
{
  botoes=analogRead(A0); //Faz a leitura analógica do pino A0 e guarda na variável botoes
  Serial.println(botoes); //Escreve no Serial Monitor o valor ligo pelo pino A0
  delay(100); //Delay de 100 milissegundos
  atualiza_menu();       //Executa a função para atualizar o menu
}
//==========================INICIALIZA A TELA DA DEMONSTRAÇÃO==========
================
void inicializacao() //Função de inicialização da tela LCD para a aplicação
{
  analogWrite(pinBRILHO,255); //Inicializa o brilho no máximo (255 = 100%)
  brilho_porcent = (pwm_brilho/255) * 100;
  analogWrite(pinCONTRASTE,0); //Inicializa o contraste no mínimo (0 = 0%)
  contraste_porcent = (pwm_contraste/255) * 100;
```

```
Serial.println(""); //Pula uma linha
  Serial.println("LCD SHIELD I2C: INICIADO" ); //Escreve no Serial Monitor
lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
  lcd.print(" LCD Shield I2C"); //Escreve no LCD
for(i=0;i<=4;i++) //Laço para a o efeito de piscagem da segunda linha
  {
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("               "); //Limpa segunda linha
    delay(250); //Permanece por 250 milissegundos apagada
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("labdegaragem.com"); //Escreve na tela do LCD a mensagem "labdegaragem.com"
    delay(250); //Permanece por 250 milissegundo acesa
  }
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++
//+++++++++++++++++++++++++++++ATUALIZAÇÃO DA LISTA DOS MENUS============
======
void atualiza_menu()
{
  if(botoes > 100 && botoes < 200) //Se botão da esquerda for pressionado
  {
    menu_cont--; //Volta para o MENU anterior
  }
else if(botoes > 700 && botoes < 800) //Senão se o botão da direita for pressionado
  {
    menu_cont++;  //Vai para o próximo MENU
  }
  else if(botoes < 100)
  {
    sel = 1; //Valida que o botão de seleção foi pressionado
  }
  trata_botao(); //Chama a função para tratamento do botão para executar somente quando for
```

```
clicado
  menu_list(); //Vai para a lista de MENU
}
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++
//=============================================LISTA DOS MENUS=========
================
void menu_list()
{

  if(menu_cont == 0) //Menu 1: Tela de Início
  {
    lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
    lcd.print(" LCD Shield I2C "); //Escreve no LCD
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("labdegaragem.com"); //Escreve na tela do LCD a mensagem "labdegaragem.com"
  }
else if(menu_cont == 1) //Menu 2: Tela de Ajuste do Brilho
  {
    lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
    lcd.print("AJUSTE BRILHO   "); //Escreve no LCD
    lcd.setCursor(0,1); //Posiciona cursor
    lcd.print("                "); //Escreve na tela do LCD a mensagem "labdegaragem.com"
    if(sel == 1) //Se botão de seleção for pressionado
    {
      while(sel == 1) //Enquanto o botão de selecionar não for pressionado denovo
      {
        botoes=analogRead(A0); //Lê os botões
        ajuste1(); //Vai para a função de ajuste do Brilho
        lcd.clear(); //Limpa LCD
        lcd.print("BRILHO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
        lcd.print(brilho_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
        lcd.print("%"); //Escreve no Serial Monitor "%" e pula linha
```

```
      trata_botao2(); //Chama a função para tratamento do botão para executar somente
quando for clicado
      if(botoes < 100) //Se o botão de seleção for pressionado
      {
        sel = 0; //Sai do MENU
        trata_botao(); //Chama a função para tratamento do botão para executar somente
quando for clicado
      }
}
   }
 }
else if(menu_cont == 2)//Menu 3: Tela de Ajuste do Contraste
 {
   lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
   lcd.print("AJUSTE CONTRASTE"); //Escreve no LCD
   lcd.setCursor(0,1); //Posiciona cursor
   lcd.print("                ");
   if(sel == 1) //Se botão de seleção for pressionado
   {
     while(sel == 1) //Enquanto o botão de selecionar não for pressionado denovo
     {
       botoes=analogRead(A0); //Lê os botões
       ajuste2();  //Vai para a função de ajuste do Brilho
       lcd.clear(); //Limpa LCD
       lcd.print("CONTRASTE = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
       lcd.print(contraste_porcent); //Escreve no Serial Monitor o valor em porcentagem do brilho
       lcd.print("%"); //Escreve no Serial Monitor "%" e pula linha
       trata_botao2(); //Chama a função para tratamento do botão para executar somente
quando for clicado
       if(botoes < 100) //Se o botão de seleção for pressionado
       {
         sel = 0; //Sai do MENU
         trata_botao(); //Chama a função para tratamento do botão para executar somente
```

```
quando for clicado
      }
    }
  }
 }
else if(menu_cont == 3)//Menu 4: Tela de Estado do Relé
 {
   lcd.setCursor(0,0); //Posiciona o cursor na primeira coluna da primeira linha
   lcd.print("ESTADO DO RELE  "); //Escreve no LCD
   lcd.setCursor(0,1); //Posiciona cursor
   lcd.print("                ");
   if(sel == 1) //Se botão da esquerda for pressionado
   {
     while(sel == 1)
     {
       botoes=analogRead(A0); //Lê os botões
       lcd.clear(); //Limpa LCD
       lcd.print("ESTADO = "); //Escreve no Serial Monitor "BRILHO =", sem pular linha
if(estadoRELE == 0) //Se o relé estiver desacionado
       {
         lcd.setCursor(9,0); //Posiciona CURSOR
         lcd.print("OFF"); //Escreve OFF no LCD
       }
else if(estadoRELE == 1) //Se o relé estiver acionado
       {
         lcd.setCursor(9,0); //Posiciona CURSOR
         lcd.print("ON"); //Escreve ON no LCD
       }
trata_botao2(); //Chama a função para tratamento do botão para executar somente quando for clicado
       ajuste3();
       if(botoes < 100) //Se o botão de seleção for pressionado
       {
```

```
          sel = 0; //Sai do MENU
          trata_botao(); //Chama a função para tratamento do botão para executar somente
quando for clicado
        }
      }
    }
  }
else if(menu_cont == 4) //Se a contagem de menu for para 4
  {
    menu_cont = 0; //Volta para o primeiro menu
  }
  else if(menu_cont == -1) //Se a contagem de menu for para -1
  {
    menu_cont = 3; //Volta para o último menu
  }
}
//---------------------------------------------------------------------------
//============================AJUSTES DOS MENUS=========================
=======
void ajuste1()
{
  if(botoes > 100 && botoes < 200) //Se botão da esquerda for pressionado
  {
    if(pwm_brilho > 0) //Se o PWM do brilho estiver maior que 0
    {
      pwm_brilho -= 25.5; //Subtrai 25.5(10%) no PWM do brilho
      analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
    }
brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
    //o valor do PWM convertido em porcentagem
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
```

```
else if(botoes > 700 && botoes < 800) //Senão se o botão da direita for pressionado
 {
   if(pwm_brilho < 255) //Se o PWM do brilho estiver menor que 255
   {
     pwm_brilho += 25.5; //Soma 25.5(10%) no PWM do brilho
     analogWrite(pinBRILHO,pwm_brilho); //Atualiza o PWM do brilho
   }
brilho_porcent = (pwm_brilho/255) * 100; //Guarda na variável brilho_porcente
   //o valor do PWM convertido em porcentagem
   trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
 }
}
void ajuste2()
{
 if(botoes > 200 && botoes < 300) //Senão se o botão de baixo for pressionado
 {
   if(pwm_contraste > 0) //Se o PWM do contraste estiver maior que 0
   {
     pwm_contraste -= 25.5; //Subtrai 25.5(10%) no PWM do contraste
     analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
   }
contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
   //o valor do PWM convertido em porcentagem
   trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
 }
else if(botoes > 400 && botoes < 500) //Senão se o botão de cima for pressionado
 {
   if(pwm_contraste < 255) //Se o PWM do contraste estiver menor que 255
   {
     pwm_contraste += 25.5; //Soma 25.5(10%) no PWM do contraste
     analogWrite(pinCONTRASTE,pwm_contraste); //Atualiza o PWM do contraste
   }
```

```
contraste_porcent = (pwm_contraste/255) * 100; //Guarda na variável contraste_porcente
    //o valor do PWM convertido em porcentagem
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
}

void ajuste3()
{
  if(botoes > 200 && botoes < 300) //Senão se o botão de baixo for pressionado
  {
    estadoRELE=0;
    digitalWrite(pinRELE,estadoRELE);
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
else if(botoes > 400 && botoes < 500) //Senão se o botão de cima for pressionado
  {
    estadoRELE=1;
    digitalWrite(pinRELE,estadoRELE);
    trata_botao(); //Tratamento do botão para variar somente 10% quando for clicado
  }
}
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++

//================================TRATAMENTO DOS BOTÕES===============
============

void trata_botao()
{
  while(botoes != 1023) //Enquanto o botão não for solto, fica preso no laço
  {
    botoes=analogRead(A0);
  }
}

void trata_botao2()
{
```

```
  while(botoes > 900) //Enquanto nenhum botão for pressionado, fica preso no laço
  {
    botoes=analogRead(A0);
  }
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++
```

3. Montagem da Demonstração 2.3

3.1) Módulo Relé

Faça as ligações de sua lâmpada com o módulo relé, conforme a imagem abaixo.

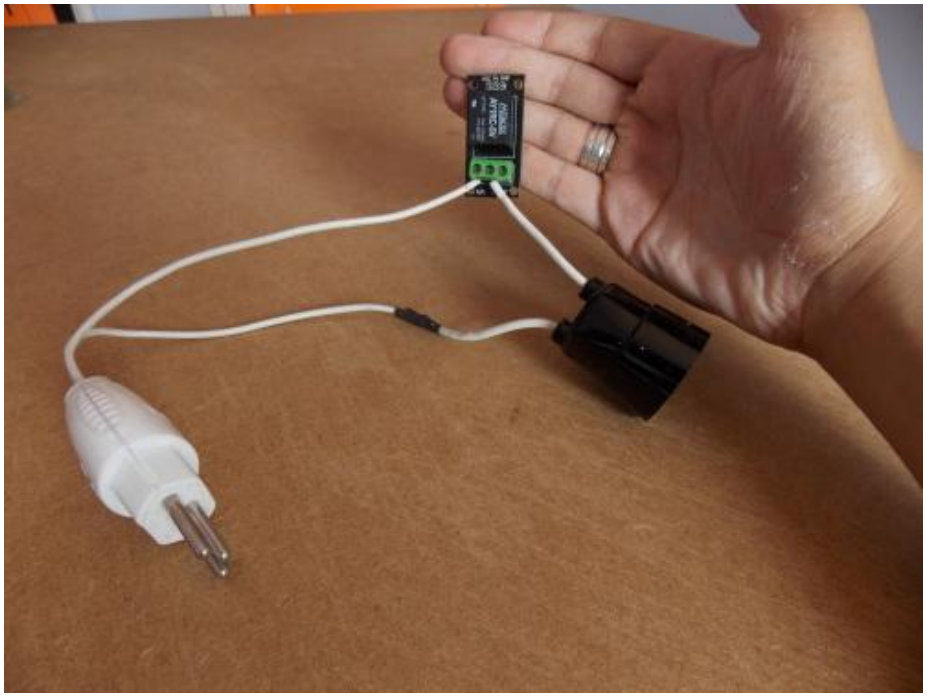


Imagem 1 - Montagem do Módulo Relé + Lâmpada

3.2) Circuito

Conecte o LCD Shield I²C no Arduino e faça as ligações no Módulo Relé, conforme a figura abaixo.

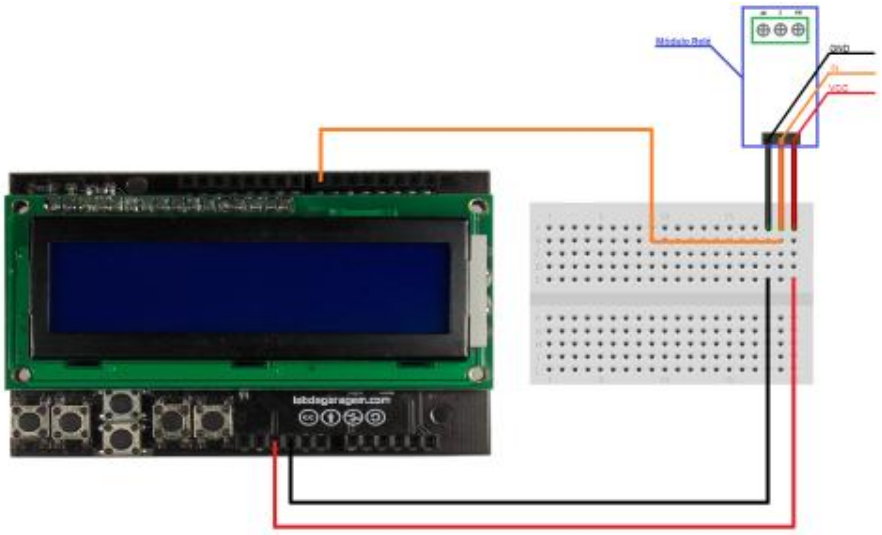


Figura 2 - Circuito da Demonstração 3.3

Então é isso Garagistas! Espero que tenham gostado desse incrível e versátil Shield! Até a Próxima!

Referências:

http://www.labdegaragem.org/loja/31-shields/lcd-shield-i2c.html

72. Tutorial - Utilizando leitor e tags RFID

Neste tutorial vamos abordar de forma simples a utilização do Módulo leitor RFID que temos disponível em nossa loja, junto a tags RFID e falaremos também um pouco sobre como funciona este protocolo de comunicação sem fio que é utilizado em diversas áreas.

Lista de Materiais

1 x Garagino Rev 1 ou Arduino

1 x Protoboard

1 x Módulo RFID 13.56Mhz

Algumas tags RFID (de mesma frequência)

Alguns jumpers

O RFID

A identificação por rádio frequência ou RFID, é uma tecnologia de comunicação que utiliza ondas eletromagnéticas para poder identificar, ler e gravar informações de forma automática com o intuito de agilizar e automatizar processos.

Os leitores RFID (conhecidos como transceptores) ficam transmitindo uma requisição de identificação para as tags, e as que estão dentro da frequência e ao alcance do mesmo, responde com um código de identificação.

As tags

Há uma diversidade de tags e elas podem ser classificadas em:

- **Ativas:** Utilizam fonte de energia proveniente de bateria e tem um longo alcance.
- **Passivas:** É alimentado pela energia advinda da transmissão do leitor RFID.
- **Semi-passiva:** Estas funcionam como as tags passivas, mas com baterias para melhor o alcance da transmissão.

Onde são Usadas

É uma tecnologia utilizada em diversos setores como por exemplo:na agropecuária para rastreamento de gado, na indústria para controle de peças no fluxo de processo e estoque, nas áreas de transporte, controle de fluxo de pessoas em propagandas de marketing ou ainda na identificação de malas de aeroportos.

Para quem mora em São Paulo, o ***Bilhete Único*** é um dos exemplos de aplicação desta tecnologia.

O Leitor RFID

No nosso exemplo, vamos utilizar o módulo leitor RFID RDM880-T-A fornecido pela seeedstudio que funciona na frequência de 13,56Mhz e segue padrão de ISO/IEC 14443 tipo a e você pode obter mais informações sobre ele clicandoneste link.

Este é um módulo que pode ser utilizado tanto para ler como para gravar tags, veja este manual com os comandos

O Sketch

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

char lerID[] = {
  0xAA , 0x00, 0x03, 0x25, 0x26, 0x00, 0x00, 0xBB };

int leitura[12];

int tagValida [5] = {
  0x5E , 0x16, 0x8F, 0x52};//Altere aqui de acordo com a sua TAG
//tagBilhete {0x5E , 0x16, 0x8F, 0x52};
//tagAzul {0x42 , 0xBC, 0x91, 0x5D};
//sem TAG AA0218380BB

void setup()
{
  pinMode(13, OUTPUT);

  Serial.begin(57600);
  Serial.println("Lab de Garagem - RFID");
  Serial.println("Exemplo de Leitura de tags: ");

  mySerial.begin(9600);
}

void loop()
{

  delay (500);
  int IDbyte=0;
```

```
  int j = 0;
  Serial.print("Leitura do ID ");
  while (mySerial.available())
  {
    leitura[j]= mySerial.read(); //buffer para armazenar o
    //Serial.print(leitura[j],HEX);
    j++;
  }
  //Serial.println("");
  for (j=5 ; j<9 ; j++)// Retira apenas os Bytes com o ID dos tag
  {
    Serial.print(leitura[j],HEX);
    if (leitura[j]==tagValida[j-5])
      IDbyte++;
  }
  if (IDbyte ==4)
  {
    Serial.println("");
    Serial.println("ID Valido. Pisca LED!!!");
    digitalWrite(13,!digitalRead(13)); //Inverte o valor do LED do pino 13
  }
  else
  {
    Serial.println("");
    Serial.println("ID Invalido");
  }
  for (j =0 ; j < 8 ; j++) //Envia comando para leitura do tag
    mySerial.write(lerID[j]);
  for (j =0 ; j < 12 ; j++) //Limpa o Buffer de leitura dos tags
    leitura[j] = 0;
  Serial.println("");
}
```

Conclusão

Esta tecnologia tem ganhado muito espaço nos últimos anos e pode ser a revolução no modo como detectamos, identificamos e controlamos estoque, cargas e também no rastreio de produtos. Para ver mais informações sobre RFID vejam os links que estão nas referências.

\o/

Referências

http://www.seeedstudio.com/wiki/index.php?title=13.56Mhz_RFID_module_-_IOS/IEC_14443_type_a

http://www.seeedstudio.com/depot/datasheet/RDM880-Spec..pdf

http://neophob.com/files/rfid/PROTOCOL-821-880%20_2_.pdf

http://www.rfsense.com.br/Tecnologias/RFID-e/Aplicacoes-Tipicas.aspx

73. Tutorial: Como fazer acionamentos com o Celular Shield via SMS

Olá Garagistas! Neste tutorial mostraremos como você pode utilizar o Celular Shield com SM5100B para fazer acionamentos via SMS. Como demonstração, faremos acionamentos de LEDS por SMS e também utilizaremos um sensor de corrente para fazer com que o Celular Shield nos envie uma SMS caso uma lâmpada for acionada.

Material Utilizado:

1x Celular Shield com SM5100B

1x Arduino

1x Sensor de Corrente ACS712

3x LEDS 5mm

3x Resistor 330Ω

1x Lâmpadas 110V

1x Interruptor

1x Protoboard

Alguns Jumpers

1. Celular Shield com SM5100B

Imagem 1 - Celular Shield com SM5100B

Este Shield Contém:

- Módulo SM5100B (sem antena): É um módulo GMS quad-band, que pode ser integrado em um grande número de projetos sem fio. Você pode usar este módulo para realizar quase qualquer coisa que um celular normal pode, como por exemplo: SMS, GSM, TCP/IP e muito mais.

- Soquete para SIM card: Você colocar um chip GSM da operadora de sua preferência.

- Comunicação serial selecionável por jumpers: Hardware (D0 e D1) ou Software (D2 e D3)

- Headers para conexão de microfone ou auto-falante

- Botão de RESET

2. Comandos AT e Biblioteca SerialGSM.h

O Módulo SM5100B é configurado via comandos AT, a lista de todos os comandos você pode encontrar no Datasheet do Módulo.

A SerialGSM.h, é uma biblioteca que tem como finalidade auxiliar o usuário somente com mensagens SMS, ou seja, as funções dela são voltadas apenas para envio, recebimento, e tratamento de SMS.

3. Demonstrações

Todas as demonstrações contidas neste tutorial, podem ser acompanhadas no nosso vídeo, no início desta página.

- 3.1 Acionamento via SMS:

```
#include <SerialGSM.h> //Inclui a biblioteca SerialGSM.h
#include <SoftwareSerial.h> //Inclui a biblioteca SoftwareSerial.h
SerialGSM cel(2,3); //Configura os pinos 2 e 3 para se trabalhar como Rx e Tx via Software

String SMS; //Variável para o armazenamento da SMS recebida
String numeroRemetente; //Variável para o armazenamento do número do remetente da SMS
String numeroPermitido = "011982823030"; //Variável que define o número do celular que o sistema irá responder

boolean sendonce=true; //Variável boleana utilizada pela biblioteca SerialGSM.h

void setup()
{
  int i; //Variável para contagem
  for(i=5;i<=7;i++) //Laço que configura os pinos D5 a D7 como saída e em nível lógico 0(LOW)
  {
    pinMode(i,OUTPUT);
    digitalWrite(i,LOW);
  }

Serial.begin(9600); //Incializa a Serial(Hardware) com um Baud Rate de 9600
  cel.begin(9600); //Incializa a Serial(Software) com um Baud Rate de 9600
  cel.Verbose(true); //Função de configuração da biblioteca SerialGSM.h
  cel.Boot(); //Função de inicialização do Módulo SM5100B
  cel.DeleteAllSMS(); //Função para deletar todas as SMS do chip
  cel.FwdSMS2Serial(); //Função utilizada para enviar as SMS para a Serial
```

```
Serial.println(">>>>> Celular Shield Inicializado <"); //Mensagem de início da aplicação
  Serial.println(); //Pula linha
}

void loop()
{
  if (cel.ReceiveSMS()) //Se o chip no Celular Shield receber uma SMS
  {
    Serial.println(); //Pula linha
    Serial.println(); //Pula linha
    Serial.println(); //Pula linha
    Serial.println(); //Pula linha
    Serial.println("~~NOVA MENSAGEM RECEBIDA~~"); //Imprime na serial "NOVA
MENSAGEM RECEBIDA"
    Serial.println(); //Pula linha

Serial.print("REMETENTE: "); //Imprime na serial a palavra "REMETENTE:"
    Serial.println(cel.Sender()); //Imprime o número do remetente que a SMS foi enviada

numeroRemetente = cel.Sender(); //Armazena na variável numeroRemetente o número do
remetente

if(numeroRemetente != numeroPermitido) //Se o número do remetente não for o mesmo que o
número permitido
    {
      Serial.println("Não Permitido"); //Imprime a mensagem "Não Permitido" e não executa nada
    }

else //Senão, o número é o permitido e executará os comandos
    {
      Serial.print("MENSAGEM: "); //Imprime na serial a palavra "MENSAGEM:"
      Serial.println(cel.Message()); //Imprime a mensagem que foi recebida
      SMS = cel.Message(); //Guarda na mensagem na variável SMS

//++++++++Tratamento do Primeiro Caractere++++++++
      if(SMS[0] == '1') //Se o primeiro caractere da SMS for igual a 1
      {
        digitalWrite(5,HIGH); //Acende o LED verde
```

```
      Serial.println();//Pula uma linha na serial
      Serial.println("LED VERDE ACESO"); //Imprime a mensagem "LED VERDE ACESO"
    }
    else if(SMS[0] == '0') //Senão, se o primeiro caractere for igual a 0
    {
      digitalWrite(5,LOW); //Apaga o LED verde
      Serial.println(); //Pula uma linha na serial
      Serial.println("LED VERDE APAGADO"); //Imprime a mensagem "LED VERDE
APAGADO"
      Serial.println(); //Pula uma linha na serial
    }
    //================================================
  }
//++++++++Tratamento do Segundo Caractere++++++++
  if(SMS[1] == '1') //Se o segundo caractere da SMS for igual a 1
  {
    digitalWrite(6,HIGH); //Acende o LED amarelo
    Serial.println(); //Pula uma linha na serial
    Serial.println("LED AMARELO ACESO"); //Imprime a mensagem "LED AMARELO ACESO"
  }
  else if(SMS[1] == '0') //Senão, se o segundo caractere for igual a 0
  {
    digitalWrite(6,LOW); //Apaga o LED amarelo
    Serial.println(); //Pula uma linha na serial
    Serial.println("LED AMARELO APAGADO"); //Imprime a mensagem "LED AMARELO
APAGADO"
  }
  //================================================
//++++++++Tratamento do Terceiro Caractere++++++++
  if(SMS[2] == '1') //Se o terceiro caractere da SMS for igual a 1
  {
    digitalWrite(7,HIGH); //Acende o LED vermelho
    Serial.println(); //Pula uma linha na serial
```

```
    Serial.println("LED VERMELHO ACESO"); //Imprime a mensagem "LED VERMELHO
ACESO"
   }
   else if(SMS[2] == '0') //Senão, se o terceiro caractere for igual a 0
   {
    digitalWrite(7,LOW); //Apaga o LED vermelho
    Serial.println(); //Pula uma linha na serial
    Serial.println("LED VERMELHO APAGADO"); //Imprime a mensagem "LED VERMELHO
APAGADO"
   }
   //=================================================

cel.DeleteAllSMS(); //Deleta todas as SMS
 }
}
```

- 3.2 Notificação de Acionamento por SMS:

```
#include <SerialGSM.h> //Inclui a biblioteca SerialGSM.h
#include <SoftwareSerial.h> //Incluir a biblioteca SoftwareSerial.h
SerialGSM cell(2,3); //Configura os pinos 2 e 3 para serem utilizados como RX e TX

//Variáveis utilizadas para o cálculo do valor eficaz da corrente em AC
float valorSensor_aux = 0;
float valorSensor = 0;
float valorCorrente = 0;
float voltsporUnidade = 0.0048828125;
//=====================================================================

void setup()
{
  Serial.begin(9600); //Inicializa a serial(Hardware) com Baud Rate = 9600

//Inicializa o Celular Shield
  cell.begin(9600);
  cell.Verbose(true);
  cell.Boot();
```

```
  cell.FwdSMS2Serial();
  //============================
Serial.println(); //Pula linha na Serial
  Serial.println(); //Pula linha na Serial
  delay(2000); //Delay de 2 segundos
}
void loop()
{
  ajusteSensor(); //Chama a função para o cálculo da corrente
  if(valorCorrente >= 0.50) //Se lâmpada estiver acesa
  {
    Serial.println(); //Pula linha na Serial
    Serial.println(); //Pula linha na Serial
    cell.Rcpt("982823030"); //Seleciono o número de envio da SMS
    cell.Message("LAMPADA ACIONADA"); //Escreve a mensagem "Lâmpada Acionada"
    cell.SendSMS(); //Envia a SMS
    Serial.println(); //Pula linha na Serial
    Serial.println(); //Pula linha na Serial
    while(valorCorrente >= 0.50) //Enquanto a lâmpada estiver acesa
    {
      ajusteSensor(); //Chama a função para o cálculo da corrente
    }
  }
}
void ajusteSensor() //Função para o cálculo do valor eficaz da corrente aferida
{
  for(int i=500; i>0; i--)
  {
    valorSensor_aux = (analogRead(A0) - 511); // Faz a leitura do sensor e subtrai 511 (511 =
sem corrente no circuito = 0A)
    valorSensor += pow(valorSensor_aux,2);  // Soma os quadradosos das leituras no laço
  }
  valorSensor = (sqrt(valorSensor/ 500)) * voltsporUnidade; // Tira a raiz quadrada da média da
```

```
somatória dos valores aferidos
  valorCorrente = (valorSensor/66)*1000; // Calcula a corrente considerando a sensibilida do
sensor para 60mV/A
  if(valorCorrente < 0.50) //Se a corrente lida for menor que 0.50
  {
    Serial.println("Lampada Apagada"); //Imprime na serial "Lâmpada Apagada"
  }
  else
  {
    Serial.println("Lampada Acesa"); //Senão, imprime na serial "Lâmpada Acesa"
  }
  delay(1000); //Delay de 1 segundo entre as leituras
}
```

4. Montagem

- 4.1) Circuito da primeira demonstração (Acionamento via SMS):

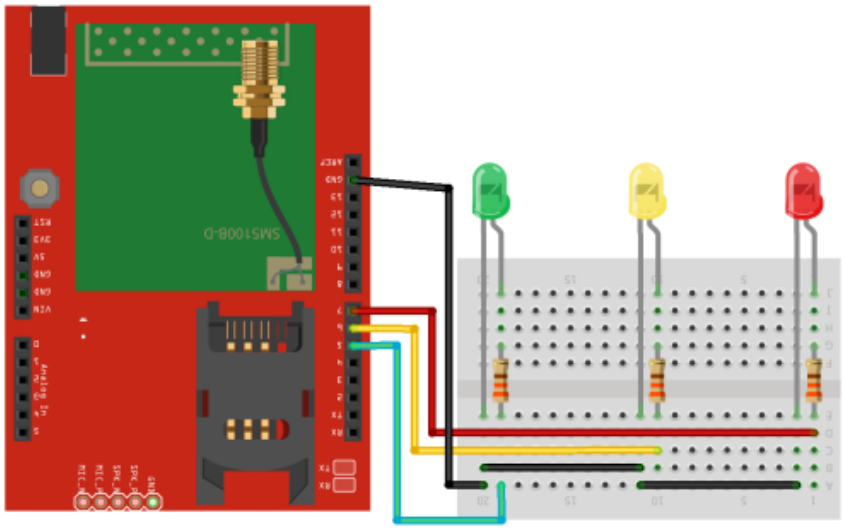


Figura 1 - Circuito da primeira demonstração

- 4.2) Ligações da segunda demonstração (Notificação de Acionamento por SMS):

4.2.1) Conecte o VCC(5V) e o GND do sensor ao Arduino, e para fazer a leitura do sensor neste tutorial, conectamos o VOUT do sensor ao pino A0:

Imagem 2 - Conexões do sensor de corrente no Celular Shield

4.2.2) Faça as ligações em seu interruptor:

Imagem 3 - Ligações no interruptor

4.2.3) Faça as ligações em sua lâmpada:

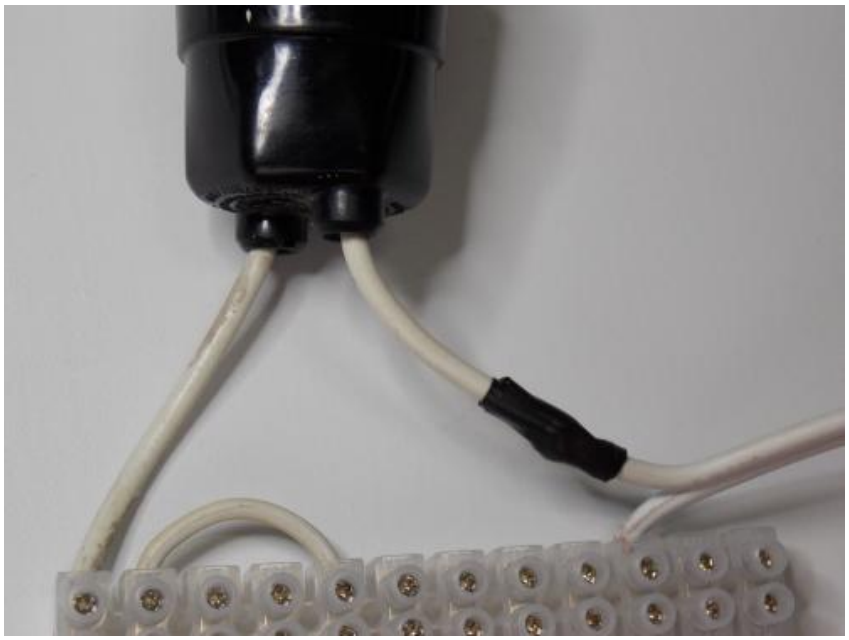


Imagem 4 - Ligação da lâmpada no borne

4.2.4)Faça as ligações em série da lâmpada, sensor de corrente e o interruptor:

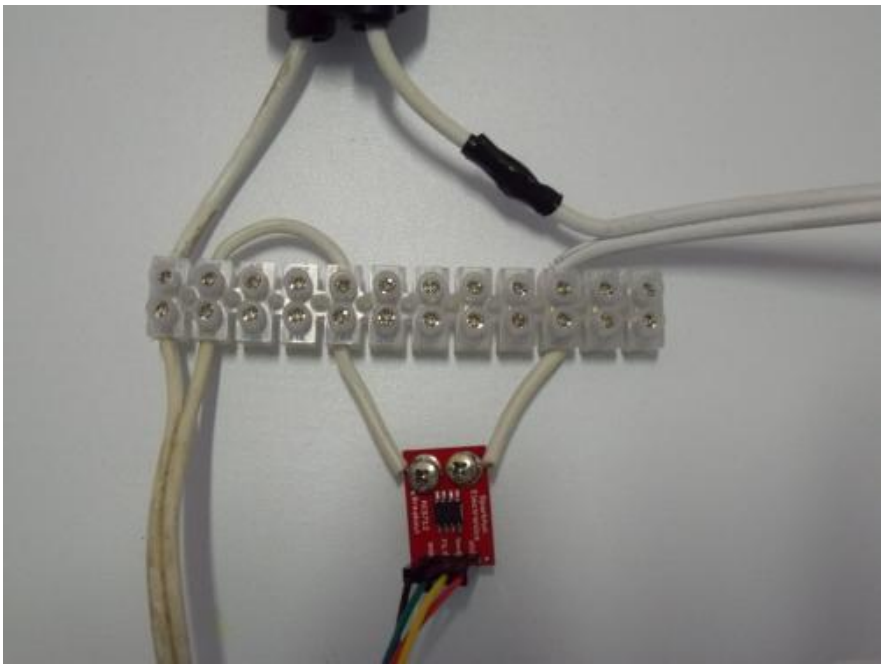


Imagem 5 - Circuito em série nos bornes

Então é isso Garagistas! Esperamos que tenham gostado deste tutorial, até a próxima!

Referências:

https://www.sparkfun.com/products/9607

https://www.sparkfun.com/products/8882

74. Tutorial: Utilizando o GPS Shield como dispositivo de anti-evasão

Olá Garagistas! No tutorial de hoje mostraremos como utilizar o GPS Shield. Faremos uma demonstração de como você pode montar um dispositivo de anti-evasão com ele, determinando uma área, e que se o GPS ultrapassar o limite desta área, o Arduino pode fazer um acionamento ou tocar uma sirene.

Material Utilizados:

1x GPS Shield

1x Arduino

1x Alto-Falante

1x Capacitor 100uF

1x Protoboard

1x Suporte para 4 pilhas

4x Pilhas AA

Alguns Jumpers

1. GPS Shield

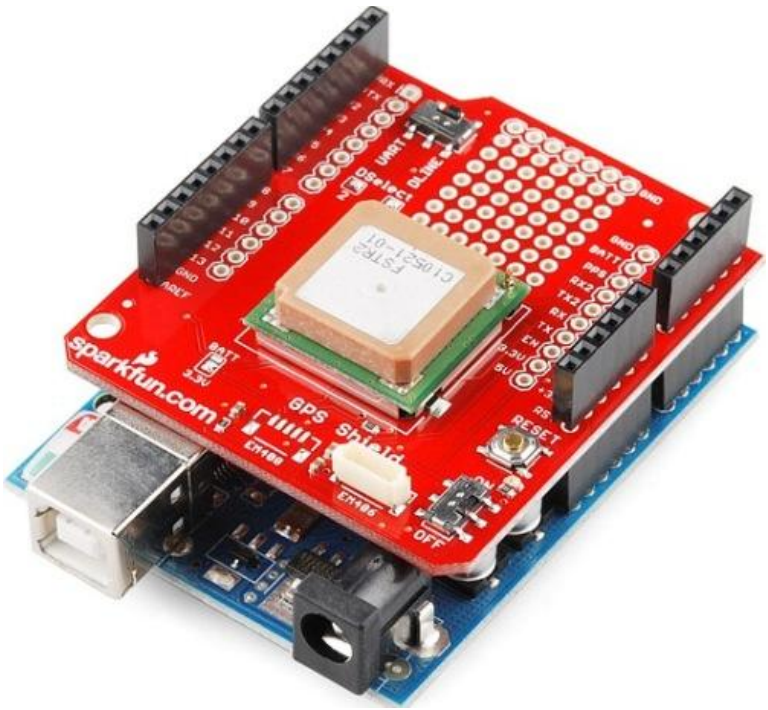


Imagem 1 - Arduino com GPS Shield

Todos os passos para a montagem do seu GPS Shield, podem ser encontrado neste link.

A biblioteca utilizada neste tutorial foi a TinyGPS, a versão utilizada desta biblioteca foi a versão 13, e essa versão pode ser encontrada neste link. Lembre-se que é necessário retirar a versão no nome da pasta do TinyGPS, quando você fizer o download da versão 13 por exemplo o arquivo virá com o nome TinyGPS**-13**, retire o **-13** do nome da pasta e a cole-a dentro da pasta ***libraries***. Essa biblioteca trata os dados recebidos pelo módulo GPS e lhe devolve informações como, longitude, latitude, hora, data, altitude e etc.

Também utilizamos neste tutorial a biblioteca SoftwareSerial (já nativa do Arduino, você não precisará baixar) para se comunicar através dos pinos D2 - RX e D3 - TX com o módulo GPS. Para a visualização dos dados recebidos utilizamos a serial (Hardware, pinos D0 - RX e D1 - TX) para através do Serial Monitor lermos os valores recebidos pelo GPS.

Sketch Exemplo: O sketch exemplo para teste de recebimento de dados do seu GPS Shield pode ser encontrado, neste link.

Para informações mais detalhadas sobre o GPS Shield, clique aqui.

2. Demonstração

Na demonstração deste tutorial, determinamos uma área (Q.G .do Lab), conforme a figura baixo:

Caso o módulo ultrapasse um desses limites, o alto-falante no circuito começará a apitar. Esse tipo de aplicação é semelhante as tornozeleiras de infratores em prisão domiciliar.

Os dados da área para essa demonstração são:

Latitude Máxima = -23.59500;

Latitude Mínima = -23.59530;

Longitude Máxima = -46.63585;

Longitude Mínima = -46.63615;

3. Circuito/Montagem

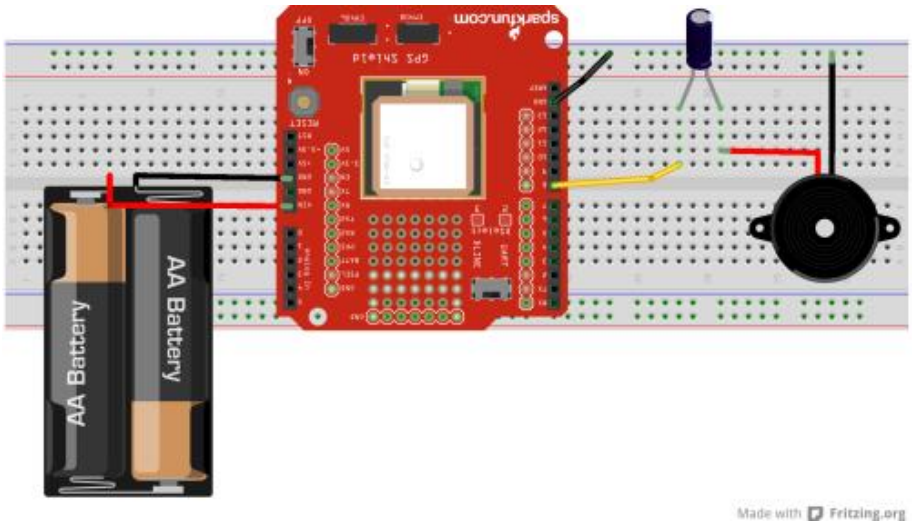


Figura 1 - Circuito da demonstração

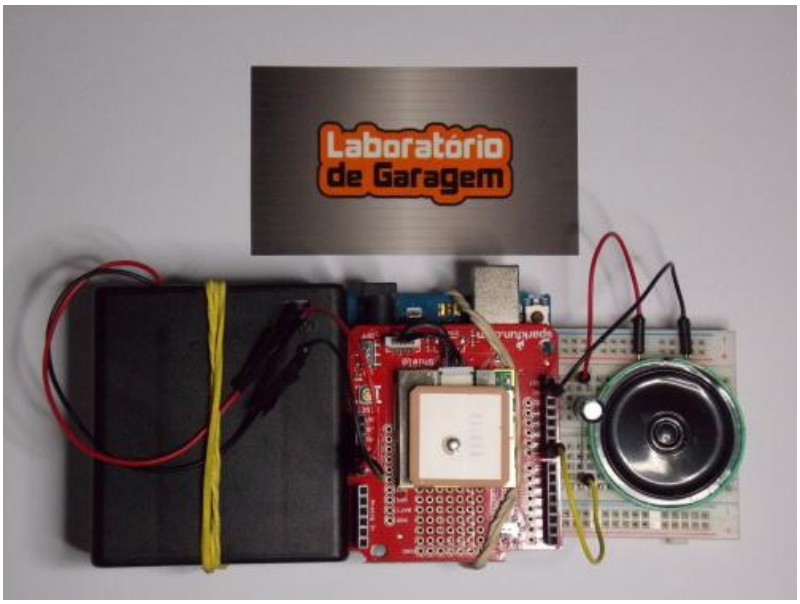


Imagem 2 - Montagem do circuito da demonstração

4. Sketch

```
#include <SoftwareSerial.h> //Biblioteca para utilização dos pinos D2 e D3 como RX e TX
#include <TinyGPS.h> //Biblioteca que faz a leitura e tratamento dos dados recebidos pelo GPS Shield

TinyGPS gps; //Cria a instância gps para ser utilizada pelo programa
SoftwareSerial serialgps(2,3); //Configura os pinos 2 e 3 para serem utilizados pela SoftwareSerial

int sats; //Variáveis para armazenamento de ano e quantidade de satélites
float latitude, longitude; //Variáveis para o armazenamento de latitude e longitude
float latmax = -23.59500; //Variável para a latitude máxima
float latmin = -23.59530; //Variável para a latitude mínima
float longmax = -46.63585; //Variável para a longitude máxima
float longmin = -46.63615; //Variável para a longitude mínima

void setup()
{
```

```
serialgps.begin(4800); //Inicializa a serial(software) com baud rate de 4800 para a comunicação com o GPS
}

void loop()
{
GPS(); //Executa a função para a leitura dos dados do GPS

//Se o GPS ultrapassar os limites determinados
 if(sats > 3)
 {
   if( (latitude < latmin) || (latitude > latmax) || (longitude < longmin) || (longitude > longmax) )
   {
     tone(8, 440, 500); //Toca sirene
   }
 }

else
 {
   tone(8, 0, 500); //Não toca a sirene
 }
}

void GPS() //Função para a leitura dos dados do GPS
{
while(serialgps.available())
 {
   int c = serialgps.read();
   if(gps.encode(c))
   {
```

```
    gps.f_get_position(&latitude, &longitude); //Leitura da latitude e longitude
    sats = gps.satellites(); //Armazena na variável sats a quantidade de satélites
   }
  }
}
```

Referências:

http://www.labdegaragem.org/loja/kit-gps-shield-retail.html

http://labdegaragem.com/profiles/blogs/tutorial-como-utilizar-o-gps-shield-com-arduino

http://www.procriativo.com.br/

75. Tutorial: Como utilizar a RelayBoard

Olá Garagistas! Neste tutorial mostraremos como utilizar a RelayBoard, uma placa muito útil que junta a praticidade com a economia, pois com apenas 3 pinos de controle você pode acionar diversos relés. E como demonstração disponibilizamos ao final deste tutorial um sketch exemplo, utilizando uma biblioteca desenvolvida para Arduino!

Material Utilizado:

RelayBoard

Arduino

Jumper Premium F/F

Jumper Premium F/M

Fonte 12V

Resistor 10KΩ

1. RelayBoard

Imagem 1 - RelayBoard

Criada pelo garagista Gustavo Suim e incubada pelo Lab de Garagem, a RelayBoard é uma placa expansível que com apenas 3 pinos de controle, permite acionar até 8 relés por placa. Com mais de uma placa ligadas juntas é possível controlar 8, 16, 24, até 80 relés, com apenas um microcontrolador. Foi desenvolvida uma biblioteca para Arduino para facilitar o controle de

todos os relés, podendo determinar qual a Placa a ser acionada, o Relé e seu Estado (ON/OFF) independentemente.

Pinos de Controle da RelayBoard:

Data: Pino responsável pelo envio dos dados responsáveis pelo controle das saídas.

Strobe: Pino responsável por disponibilizar nas saídas as entradas carregadas.

Clock: Pino responsável pelo clock das placas.

2. Biblioteca - RelayBoard.h

A biblioteca, e toda a documentação de como utilizá-la pode ser encontrada clicando aqui. A descrição das funções da biblioteca podem ser encontradas abaixo:

- #include <RelayBoard.h>

Declaração necessária para utilização da placa.

- RelayBoard::RelayBoard(int date, int str, int clk, int numberboard)

- **Date:** Pino responsável pelo envio dos dados responsáveis pelo controle das saídas.
- **Str:** Pino responsável por disponibilizar nas saídas as entradas carregadas.
- **Clk:** Pino responsável pelo clock das placas.
- **Numberboard:** variável que informa para todo o sistema a quantidade de placas acopladas.

Ex.: RelayBoard relay(2,4,6,2); // cria o objeto "relay" com as devidas propriedades

Observação:

- TODOS os pinos desta função são configurados como OUT automaticamente.

- void RelayBoard::set(int shield, int out, boolean status)

- **Shield:** Informa qual a placa que deverá ser utilizada
- **Out:** informa qual a saída da placa informada acima deverá ser acionada (ON/OFF).
- **Status:** Devemos informar "1" para LIGAR a saída ou "0" para DESLIGAR a saída.

Ex.: relay.set(0,3,1);

Observações:

- A contagem de placas e saídas iniciam-se com o valor 0 (zero).
- Esta função realiza o acionamento da saída sempre que for utilizada.

- void RelayBoard::load(int board, int out, boolean status)

- **Board:** Informa qual a placa que deverá ser acionada
- **Out:** informa qual a saída da placa informada acima deverá ser acionada.
- **Status:** Devemos informar “1” para LIGAR a saída ou “0” para DESLIGAR a saída.

Ex.: relay.load(0,2,1);

Observação:

- Esta função carrega o STATUS desejado das saídas, porém NÃO faz o acionamento das mesmas.

- void RelayBoard::go()

Após carregar as saídas solicitadas na função LOAD, à função GO faz o acionamento simultâneo das saídas solicitadas.

Ex.: relay.go();

3. Circuito da Demonstração

Utilizando uma fonte 12V para alimentar a(s) RelayBoard e o Arduino, faça as ligações do circuito abaixo.

Imagem 2 - Circuito da demonstração

4. Sketch da Demonstração

```
//=================================================================
=============

#include <RelayBoard.h> //Inclui a biblioteca RelayBoard.h

#define data    6 //Define a palavra data como 6 para o pino D6 ser utilizado como o pino do
DATA
#define strobe  4 //Define a palavra strobe como 4 para o pino D4 ser utilizado como o pino do
STROBE
#define clock   2 //Define a palavra clock como 2 para o pino D2 ser utilizado como o pino do
CLOCK
#define numberboards 2 //Define a palavra numberboards como 2, onde é definido quantas
RelayBoard há no circuito

RelayBoard relay(data, strobe, clock, numberboards); //Cria a instância, informando os pinos de
controle e o número                                  //de placas no circuito

int i; //Variável para contagem
int j; //Variável para contagem

void setup()
{
  delay(1000); //Aguarda 1 segundo antes de iniciar o programa
}

void loop()
{

//relay.set(int board, int out, boolean status)
        //Board: Informa qual a placa que deverá ser utilizada, começando de 0.
```

```
        //Out: informa qual a saída da placa informada acima deverá ser acionada (ON/OFF), de 0 a 7
        //Status: Devemos informar "1" para LIGAR a saída ou "0" para DESLIGAR a saída.
//relay.load(int board, int out, boolean status)
        //Board: Informa qual a placa que deverá ser utilizada, começando de 0.
        //Out: informa qual a saída da placa informada acima deverá ser acionada (ON/OFF), de 0 a 7
        //Status: Devemos informar "1" para LIGAR a saída ou "0" para DESLIGAR a saída.
//relay.go()
        //Após carregar as saídas solicitadas na função LOAD, à função GO faz o acionamento simultâneo das          //saídas solicitadas.
for(i=0; i<=7; i++)
 {
   relay.set(0,i,1);
   delay(50);
   relay.set(0,i,0);
   delay(50);
 }
for(i=7; i>=0; i--)
 {
   relay.set(0,i,1);
   delay(50);
   relay.set(0,i,0);
   delay(50);
 }
for(i=0,j=7;i<=3;i++,j--)
 {
   relay.load(0,i,1);
   relay.load(0,j,1);
   relay.go();
   delay(50);
 }
```

```
for(i=3,j=4;i>=0;i--,j++)
  {
    relay.load(0,i,0);
    relay.load(0,j,0);
    relay.go();
    delay(50);
  }
for(i=0;i<=1;i++)
  {
    for(j=0;j<=1;j++)
    {
      relay.load(0,0,j);
      relay.load(0,1,j);
      relay.load(0,6,j);
      relay.load(0,7,j);
      relay.go();
      delay(500);
    }
  }
for(i=0;i<=1;i++)
  {
    for(j=0;j<=1;j++)
    {
      relay.load(0,2,j);
      relay.load(0,3,j);
      relay.load(0,4,j);
      relay.load(0,5,j);
      relay.go();
      delay(500);
    }
  }
for(i=0; i<=7; i++)
  {
```

```
    relay.load(0,i,0);
  }
relay.go();
  delay(500);
for(i=0; i<=7; i++)
  {
    relay.load(0,i,1);
  }
relay.go();
  delay(500);
for(i=0; i<=7; i++)
  {
    relay.set(1,i,1);
    delay(50);
    relay.set(1,i,0);
    delay(50);
  }
for(i=7; i>=0; i--)
  {
    relay.set(1,i,1);
    delay(50);
    relay.set(1,i,0);
    delay(50);
  }
for(i=0,j=7;i<=3;i++,j--)
  {
    relay.load(1,i,1);
    relay.load(1,j,1);
    relay.go();
    delay(50);
  }
```

```
for(i=3,j=4;i>=0;i--,j++)
 {
   relay.load(1,i,0);
   relay.load(1,j,0);
   relay.go();
   delay(50);
 }
for(i=0;i<=1;i++)
 {
   for(j=0;j<=1;j++)
   {
     relay.load(1,0,j);
     relay.load(1,1,j);
     relay.load(1,6,j);
     relay.load(1,7,j);
     relay.go();
     delay(500);
   }
 }
for(i=0;i<=1;i++)
 {
   for(j=0;j<=1;j++)
   {
     relay.load(1,2,j);
     relay.load(1,3,j);
     relay.load(1,4,j);
     relay.load(1,5,j);
     relay.go();
     delay(500);
   }
 }
for(i=0; i<=7; i++)
 {
```

```
    relay.load(1,i,0);
  }
relay.go();
  delay(500);
for(i=0; i<=7; i++)
  {
    relay.load(1,i,1);
  }
relay.go();
  delay(1000);
for(i=0; i<=7; i++)
  {
    relay.load(1,i,0);
    relay.load(0,i,0);
  }
relay.go();
}
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++
```

Então é isso Garagistas! Agora vocês podem começar a utilizar essa placa para fazer uma automação na sua residência, esperamos que tenham gostado deste tutorial, até a próxima!

=D

Referências:

http://www.labdegaragem.org/loja/relayboard.html

76. Tutorial: Como utilizar o Dual Motor Shield

Olá Garagistas! No tutorial de hoje mostraremos como utilizar o Dual Motor Shield. E como demonstração controlaremos o Zumo pelas teclas 8 (Frente), 2 (Trás), 4 (Esquerda), e 6 (Direita) do teclado, através do Serial Monitor.

Material Utilizados:

1x Arduino

1x Cabo USB A para B - 1,80 metros

1x Dual Motor Driver

1x KIT Chassi Zumo Pololu

4x Pilha AA

Alguns Jumpers

1. Dual Motor Shield

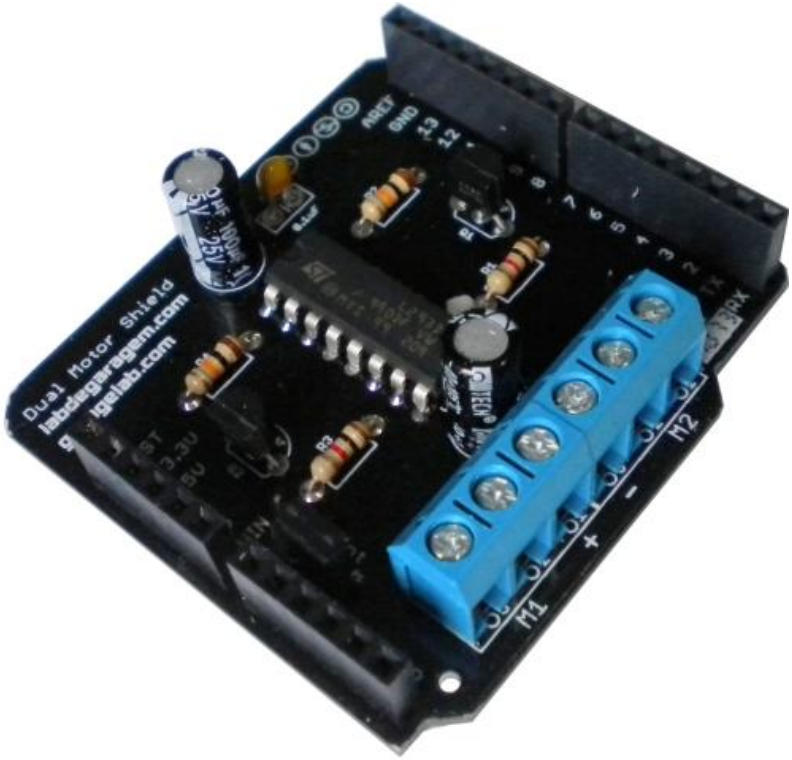


Imagem 1 - Dual Motor Shield

O Dual Motor Shield é um Shield para Arduino muito prático, e com a biblioteca desenvolvida pelo Laboratório de Garagem sua utilização fica bem mais simples, pois com apenas um comando você define o sentido e velocidade de até 2 motores DC. Muito prático, é uma excelente solução para quem quer controlar plataformas robóticas como o Zumo, o Magician, e a Esteira Tamiya.

2. Biblioteca DualMotor.h

A biblioteca desenvolvida para o Shield é composta por poucas funções:

DualMotor dualmotor; //Instância a DualMotor

dualmotor.M1move(velocidade, sentido); //Aciona o motor 1, velocidade (0 a 255) e sentido (0 - Horário ou 1 - Anti-horário)

dualmotor.M2move(velocidade, sentido); //Aciona o motor 2, velocidade (0 a 255) e sentido (0 - Horário ou 1 - Anti-horário)

dualmotor.M1parar(); //Para o motor 1

dualmotor.M2parar(); //Para o motor 2

A biblioteca pode ser baixada clicando aqui. Basta você extrair a pasta e copia-la para dentro da pasta *libraries* do Arduino.

3. Ligações no Shield

As conexões dos motores nos bornes do Dual Motor Shield, influenciam na hora em que você for desenvolver o seu Sketch, na demonstração foram feitas as ligações conforme a imagem abaixo:

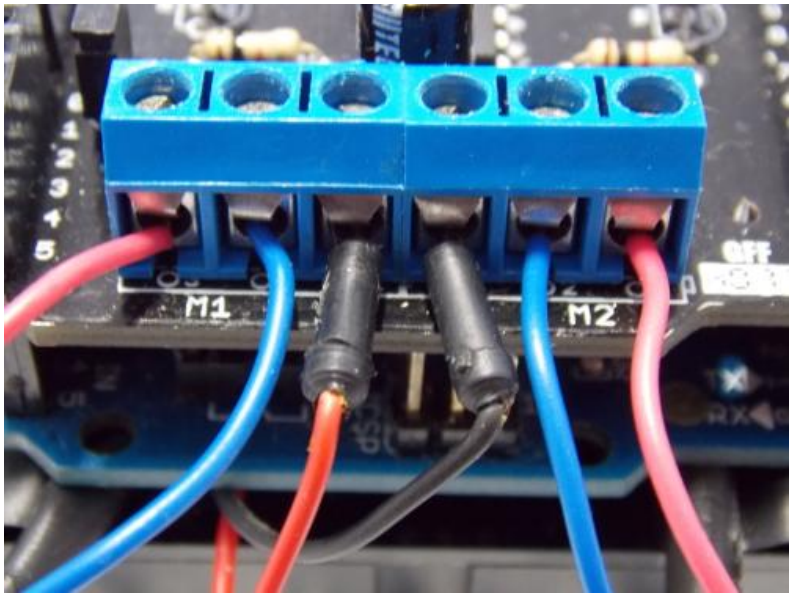


Imagem 2 - Conexões dos motor no Dual Motor Shield

Nessa montagem, quando se girar o motor no sentido **1**, o respectivo motor irá mover o Zumo para frente, e quando se girar no sentido **0**, o mesmo irá mover o Zumo para trás. Lembrando que, para a plataforma robótica girar, virar, ou fazer curva, é necessário que se pare um motor e rotacione o outro, ou que se gire um motor em um sentido e o outro em sentido oposto, na tabela abaixo você irá encontrar a configuração de cada um dos motores e a resposta para a movimentação da plataforma Zumo, para a montagem da imagem acima:

Tabela 1 - Resposta em movimentação dos motores

M1	M2	Movimentação
0	0	Trás
0	1	Esquerda
1	0	Direita
1	1	Frente

OBS: Para que a plataforma robótica fique parada, é necessário executar a função **dualmotor.Mnparar();.**

4. O Sketch

Este sketch pode ser encontrado dentro da biblioteca DualMotor.h, ou se preferir copie o cole o código dentro da Arduino IDE.

OBS: É necessário que você tenha copiado a pasta ***DualMotor*** para dentro da pasta libraries.

```
//=======================================================

//=======================================================


#include <DualMotor.h> //Inclui a biblioteca DualMotor.h


DualMotor dualmotor; //Instância a DualMotor


char letra; //Cria uma variável char para armazenamento dos caraceteres lidos
```

```
void setup()
{
  Serial.begin(9600); //Inicia a Serial com um Baud Rate de 9600bps
    dualmotor.M1parar(); //Para o motor1
    dualmotor.M2parar(); //Para o motor2
}


void loop()
{

  if (Serial.available() > 0) //Se algo for recebido na serial
  {
    letra = Serial.read(); //Armazena o que foi recebido na variável letra
    switch(letra) //Entra no switch para a comparação do caractere recebido
    {

    case '8'://FRENTE //Caso o caracter for igual a 8, vai pra FRENTE
      dualmotor.M1move(255,1);
      dualmotor.M2move(255,1);
      break; //Sai do switch

    case '4'://ESQUERDA //Caso o caracter for igual a 4, vira pra ESQUERDA
      dualmotor.M1move(255,0);
      dualmotor.M2move(255,1);
      break; //Sai do switch

    case '2'://TRÁS //Caso o caracter for igual a 2, vai pra TRAS
      dualmotor.M1move(255,0);
      dualmotor.M2move(255,0);
      break; //Sai do switch

    case '6'://senEITA //Caso o caracter for igual a 6, vira pra senEITA
      dualmotor.M1move(255,1);
```

```
    dualmotor.M2move(255,0);
    break; //Sai do switch

  }

  delay(250); //Delay de 250 milissegundos
  dualmotor.M1parar(); //Para o motor 1
  dualmotor.M2parar(); //Para o motor 2

letra = 0;//Limpa a variável letra
 }
}


//=====================================================

//=====================================================
```

Bem, então é isso Garagistas! Esperamos que tenham gostado deste Tutorial, até a próxima!

=D

Referências:

http://labdegaragem.com/profiles/blogs/tutorial-driver-para-motores...

77. Tutorial: Como utilizar os módulos RF Link (315MHz/434MHz)

Olá Garagistas! No tutorial de hoje mostraremos como você pode facilmente utilizar os módulos de transmissão RF Link (315MHz/434MHz) utilizando o Garagino. Como demonstração faremos acionamentos no módulo receptor, a partir de botões pressionador no módulo transmissor.

Material Utilizados:

2x Kit Garagino Rev 1

1x Módulo Transmissor(315MHz)**/**Módulo Receptor(315MHz) ou Módulo Transmissor(434MHz)**/**Módulo Receptor(434MHz)

4x Chave Tactil

3x Resistor 330Ω (Acionamento do LEDs)

4x Resistor 10KΩ (Pull-down das chaves táteis)

3x LEDs (Verde, Amarelo e Vermelho)

2x Protoboard

Alguns Jumpers (Circuito e Antena)

1. RF Link (315MHz/434MHz)

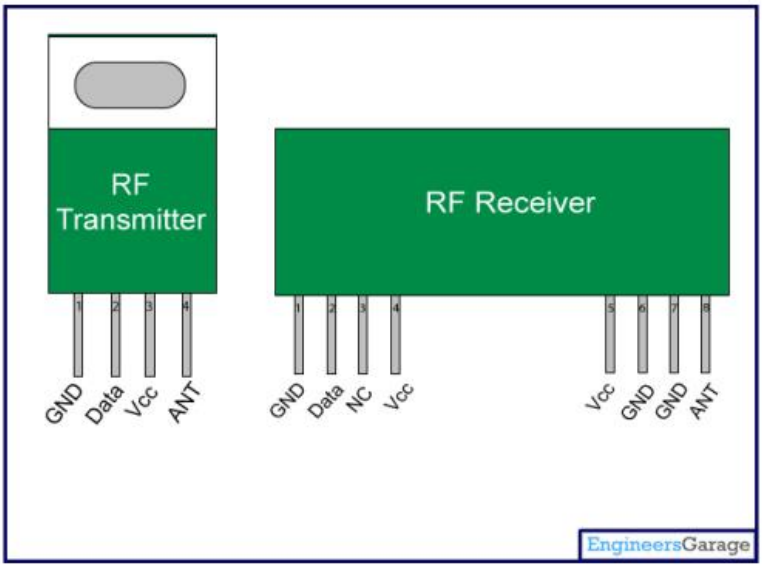


Figura 1 - Pinout Módulos RF

Os Módulos RF Link são módulos para transmissão por Rádio Frequência. Normalmente quando se pensamos em comunicação sem fio para ser utilizado com Garagino ou Arduino, a primeira coisa que vem à mente são os famosos XBee, contudo nem todos tem a possibilidade de comprar um XBee, pois são relativamente caros se comparado com os módulos RF.

Esses módulos são bem simples, o transmissor enviar dados em série para o receptor fazendo um método de transmissão simplex e ponto, o transmissor envia e o receptor recebe, simples assim. Contudo, seria necessário você utilizar um encoder (codificador) e um decoder (decodificador) para facilitar o processo de transmissão entre seus módulos, daí a vantagem de você pode utilizar um Garagino ou Arduino, porque além de você utilizá-lo como encoder ou decoder, você irá consumir somente 1 pino de cada Garagino para a comunicação e terá todo os outros disponíveis para a leitura de sensores, acionamentos de carga e pode também tratar os dados que serão enviados e recebidos.

E qual a diferença entre as frequências de 315MHz e 434MHz? Como a diferença entre as frequências desses módulos é pequena, basicamente em aplicações práticas os resultados vão ser bem próximos, já na teoria a frequência de 433MHz deve se comportar melhor em ambientes fechados enquanto a de 315MHz terá o maior comprimento de onda, logo você terá maior alcance ao ar livre. Lembrando que no Brasil essas frequência são permitidas para uso livre pela Anatel.

Em nossa aplicação teremos então um Garagino (encoder) para o módulo transmissor e um Garagino (decoder) para o módulo receptor. Faremos a leitura dos botões conectados nos

pinos digitais D6~D9 do módulo transmissor e faremos o acionamento desses respectivos pinos digitais no módulo receptor, onde estão conectados nesses pinos:

D6 - LED Verde

D7 - LED Amarelo

D8 - LED Vermelho

D9 - Módulo Relé (Lâmpada)

2. Biblioteca VirtualWire.h

Nessa demonstração utilizamos a biblioteca VirtualWire.h, ela facilita a aplicação e utilização desses módulos. É uma biblioteca bastante útil e simples de se utilizar e ela pode ser utilizada tanto para o módulo RF Link de 315MHz quanto para o módulo RF Link de 434MHz, para informações mais detalhadas sobre as biblioteca, consulte o material neste link. Neste tutorial foi utilizada a versão 1.20 desta biblioteca, e ela pode ser baixada, clicando aqui.

3. Sketch Módulo Receptor

- Transfira o sketch abaixo para o Garagino do módulo receptor:

```
#include <VirtualWire.h> //Inclui a biblioteca VirtualWire.h

void setup()
{

  //+++++++++++++++Inicializa o módulo receptor++++++++++++++++++++
  vw_set_ptt_inverted(true);
  vw_setup(2000);
  vw_set_rx_pin(2); //Configura o pino D2 para a leitura dos dados
  vw_rx_start(); //Inicia a leitura de dados do módulo receptor
  //===========================================================


  //Configura os pinos de 6 a 9 como saída
  for(int i=6;i<=9;i++)
  {
    pinMode(i, OUTPUT);
  }
  //====================================
```

```
}
void loop()
{

  uint8_t buf[VW_MAX_MESSAGE_LEN]; //Variável para o armazenamento do buffer dos dados
  uint8_t buflen = VW_MAX_MESSAGE_LEN; //Variável para o armazenamento do tamanho do buffer

  if(vw_get_message(buf, &buflen)) //Se no buffer tiver algum dado (O ou 1)
  {

    //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    //Incrementa a posição para a leitura do buffer por i (0 a 4)
    //Incrementa a posição dos pinos digitais por j (6 a 9)
    int j=6;
    for (int i = 0; i < buflen; i++,j++)
    {
      buf[i] = buf[i] - 48; //Pega o dado do buffer que é recebido em hexadecimal e subtrai 48
      if(buf[i] == 1) //Se o dado na determinada posição do buffer for igual 1
      {
        digitalWrite(j,!digitalRead(j)); //Inverte o estado do respectivo pino digital
      }
    }
    //==============================================================================

  }

}
```

4. Sketch Módulo Transmissor

- Transfira o sketch abaixo para o Garagino do módulo transmissor:

```
#include <VirtualWire.h> //Inclui a biblioteca VirtualWire.h


char *nibble = "0000"; //Cria a variável nibble com o valor 0000


int le_pino; //Cria a variável para a leitura dos pinos digitais


void setup()
{

  //+++++++++++++++Inicializa o módulo transmissor++++++++++++++++++++
  vw_set_ptt_inverted(true);
  vw_setup(2000);
  vw_set_tx_pin(3); //Configura o pino D3 para a leitura dos dados
  //=====================================================================

  //Configura os pinos de 6 a 9 como entrada
  for(int i=6; i<=9; i++)
  {
    pinMode(i, INPUT_PULLUP);
  }
  //=====================================


}

void loop()
{

  //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++
```

```
  //Incrementa a posição para o armazenamento no nibble por i (0 a 4)
  //Incrementa a posição dos pinos digitais por j (6 a 9)
  int j=0;
  for(int i=6;i<=9;i++,j++)
  {
    le_pino = digitalRead(i); //Lê o estado do botão
    if(le_pino == 1) //Se o botão estiver pressionado
    {
      while(le_pino == 1) le_pino = digitalRead(i); //Aguarda o botão ser despressionado
      nibble[j] = '1'; //Armazena na respectiva posição do no nible o caracter 1
    }
    else nibble[j] = '0'; //Senão armazena na respectiva posição do no nible o caracter 1
  }
vw_send((uint8_t *)nibble, strlen(nibble)); //Envia a variável nibble
  vw_wait_tx(); //Aguarda o fim de transmissão
  //=========================================================================
==================


}
```

5. Circuito do Módulo Receptor

Faça as ligações do circuito do módulo receptor:

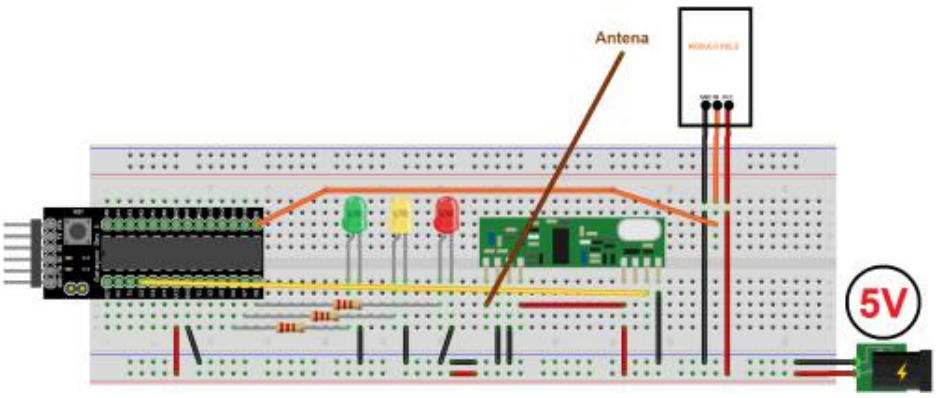


Figura 2 - Circuito do Módulo Receptor

6. Circuito do Módulo Transmissor

Faça as ligações do circuito do módulo transmissor:

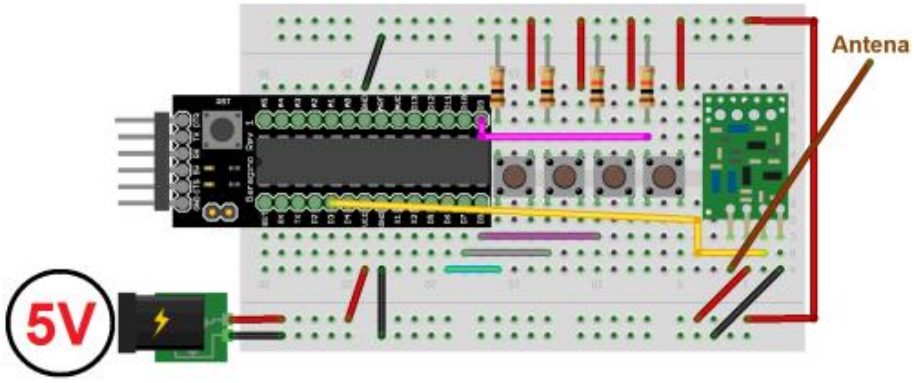


Figura 3 - Circuito do Módulo Transmissor

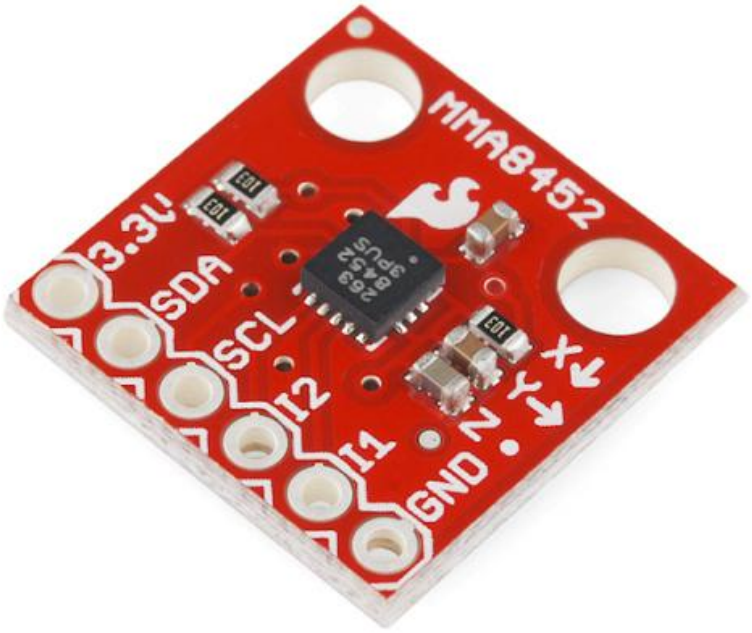
Referências

http://www.airspayce.com/mikem/arduino/

http://lusorobotica.com/index.php?topic=772.0

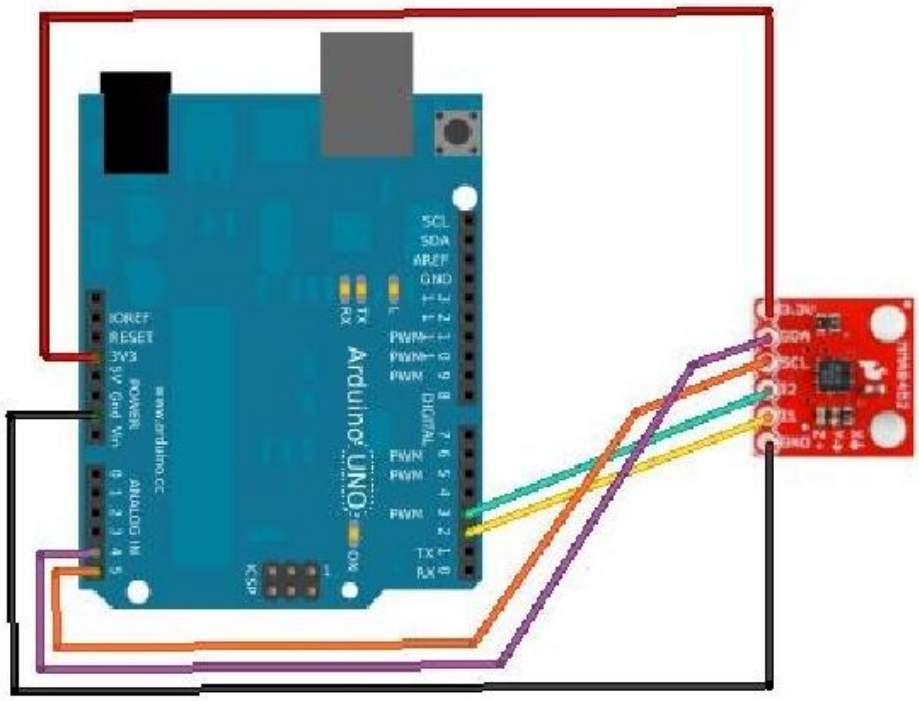
http://www.univasf.edu.br/~gari/futvasf/paginas/download/Apresenta%C3%A7%C3%A3oRF_Manoel%2009-04-2010.pdf

78. Tutorial: Como utilizar o acelerômetro Breakout MMA8452Q com Arduino



O Breakout MMA8452Q é um acelerômetro de baixo custo e fácil implementação. Aqui vamos mostrar como fazê-lo funcionar com o exemplo junto com a biblioteca disponível.

Faça a seguinte ligação:



Baixe a biblioteca com exemplo no link: http://www.labdegaragem.com.br/loja/MMA8452.zip

Extraia a biblioteca na pasta “libraries” dentro da pasta do Software do Arduino.

Agora conecte o Arduino no PC e abra o software do Arduino. Vá em

“File/Examples/MMA8452/MMA8452Q_Example” e mostrará o código abaixo:

```
/* MMA8452Q Example Code
   by: Jim Lindblom
   SparkFun Electronics
   date: November 17, 2011
   license: Beerware - Use this code however you'd like. If you
   find it useful you can buy me a beer some time.

   This code should provide example usage for most features of
   the MMA8452Q 3-axis, I2C accelerometer. In the loop function
   the accelerometer interrupt outputs will be polled, and either
   the x/y/z accel data will be output, or single/double-taps,
   portrait/landscape changes will be announced to the serial port.
   Feel free to comment/uncomment out some of the Serial.print
   lines so you can see the information you're most intereseted in.

   The skeleton is here, feel free to cut/paste what code you need.
   Play around with the settings in initMMA8452Q. Try running the
   code without printing the accel values, to really appreciate
   the single/double-tap and portrait landscape functions. The
   P/L stuff is really neat, something not many accelerometers have.

   Hardware setup:
   MMA8452 Breakout ------------ Arduino
                    3.3V --------------------- 3.3V
```

```
                SDA ---------------------- A4
                SCL ---------------------- A5
                INT2 --------------------- D3
                INT1 --------------------- D2
                GND ---------------------- GND

  SDA and SCL should have external pull-up resistors (to 3.3V).
  10k resistors worked for me. They should be on the breakout
  board.

  Note: The MMA8452 is an I2C sensor, however this code does
  not make use of the Arduino Wire library. Because the sensor
  is not 5V tolerant, we can't use the internal pull-ups used
  by the Wire library. Instead use the included i2c.h, defs.h and types.h files.
*/
#include "MMA8452Q.h"  // not the wire library, can't use pull-ups

// the SparkFun breakout board defaults to 1, set to 0 if SA0 jumper on the bottom of the board i
s set
#define SA0 1
#if SA0
  #define MMA8452_ADDRESS 0x1D  // SA0 is high, 0x1C if low
#else
  #define MMA8452_ADDRESS 0x1C
#endif

/* Set the scale below either 2, 4 or 8*/
const byte SCALE = 2;  // Sets full-scale range to +/-2, 4, or 8g. Used to calc real g values.
/* Set the output data rate below. Value should be between 0 and 7*/
const byte dataRate = 0;  // 0=800Hz, 1=400, 2=200, 3=100, 4=50, 5=12.5, 6=6.25, 7=1.56

/* Pin definitions */
int int1Pin = 2;  // These can be changed, 2 and 3 are the Arduinos ext int pins
int int2Pin = 3;

int accelCount[3];  // Stores the 12-bit signed value
float accelG[3];  // Stores the real accel value in g's

void setup()
{
  byte c;

  Serial.begin(115200);

  /* Set up the interrupt pins, they're set as active high, push-pull */
  pinMode(int1Pin, INPUT);
  digitalWrite(int1Pin, LOW);
  pinMode(int2Pin, INPUT);
  digitalWrite(int2Pin, LOW);

  /* Read the WHO_AM_I register, this is a good test of communication */
  c = readRegister(0x0D);  // Read WHO_AM_I register
  if (c == 0x2A) // WHO_AM_I should always be 0x2A
  {
    initMMA8452(SCALE, dataRate);  // init the accelerometer if communication is OK
    Serial.println("MMA8452Q is online...");
  }
  else
  {
    Serial.print("Could not connect to MMA8452Q: 0x");
```

```
    Serial.println(c, HEX);
    while (1)  // Loop forever if communication doesn't happen
      ;
  }
}

void loop()
{
  static byte source;

  /* If int1 goes high, all data registers have new data */
  if (digitalRead(int1Pin)==1)  // Interrupt pin, should probably attach to interrupt function
  {
    readAccelData(accelCount);  // Read the x/y/z adc values

    /* Below we'll print out the ADC values for acceleration
    for (int i=0; i<3; i++)
    {
       Serial.print(accelCount[i]);
       Serial.print("\t\t");
    }
    Serial.println();*/

    /* Now we'll calculate the accleration value into actual g's */
    for (int i=0; i<3; i++)
      accelG[i] = (float) accelCount[i]/((112)/(2*SCALE));  // get actual g value, this depends on
scale being set
    /* print out values */
    for (int i=0; i<3; i++)
    {
      Serial.print(accelG[i], 4);  // Print g values
      Serial.print("\t\t");  // tabs in between axes
    }
    Serial.println();
  }

  /* If int2 goes high, either p/l has changed or there's been a single/double tap */
  if (digitalRead(int2Pin)==1)
  {
    source = readRegister(0x0C);  // Read the interrupt source reg.
    if ((source & 0x10)==0x10)  // If the p/l bit is set, go check those registers
      portraitLandscapeHandler();
    else if ((source & 0x08)==0x08)  // Otherwise, if tap register is set go check that
      tapHandler();
  }
  delay(100);  // Delay here for visibility
}

void readAccelData(int * destination)
{
  byte rawData[6];  // x/y/z accel register data stored here

  readRegisters(0x01, 6, &rawData[0]);  // Read the six raw data registers into data array

  /* loop to calculate 12-bit ADC and g value for each axis */
  for (int i=0; i<6; i+=2)
  {
    destination[i/2] = ((rawData[i] 8) | rawData[i+1]) >> 4;  // Turn the MSB and LSB into a 12-bit
value
    if (rawData[i] > 0x7F)
```

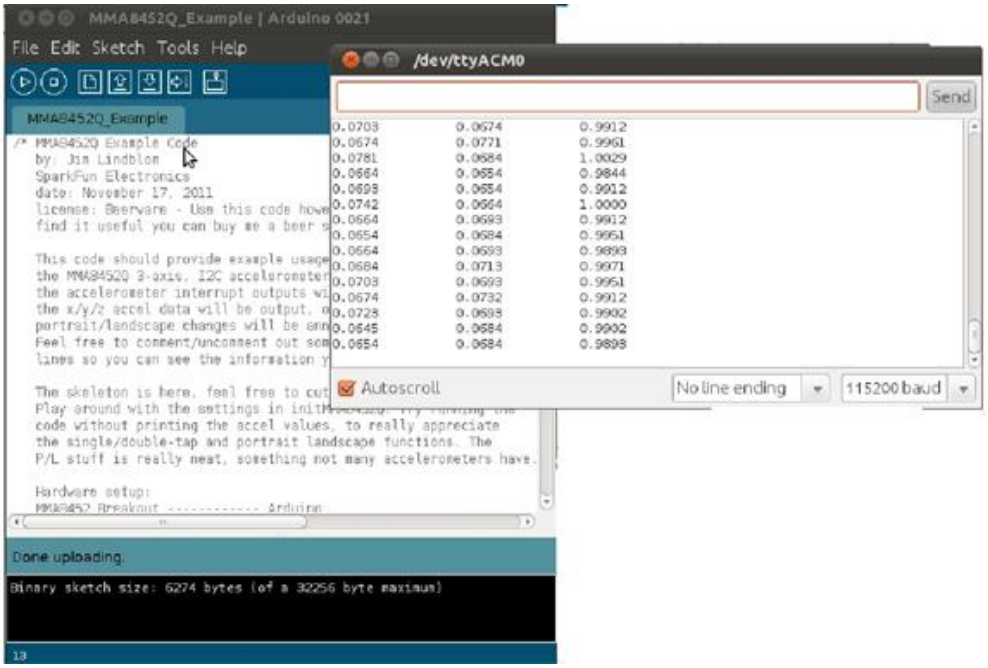
```
    {  // If the number is negative, we have to make it so manually (no 12-bit data type)
      destination[i/2] = ~destination[i/2] + 1;
      destination[i/2] *= -1;  // Transform into negative 2's complement #
    }
  }
}

/* This function will read the status of the tap source register.
   And print if there's been a single or double tap, and on what
   axis. */
void tapHandler()
{
  byte source = readRegister(0x22);  // Reads the PULSE_SRC register

  if ((source & 0x10)==0x10)  // If AxX bit is set
  {
    if ((source & 0x08)==0x08)  // If DPE (double puls) bit is set
      Serial.print("    Double Tap (2) on X");  // tabbing here for visibility
    else
      Serial.print("Single (1) tap on X");

    if ((source & 0x01)==0x01)  // If PoIX is set
      Serial.println(" +");
    else
      Serial.println(" -");
  }
  if ((source & 0x20)==0x20)  // If AxY bit is set
  {
    if ((source & 0x08)==0x08)  // If DPE (double puls) bit is set
      Serial.print("    Double Tap (2) on Y");
    else
      Serial.print("Single (1) tap on Y");

    if ((source & 0x02)==0x02)  // If PoIY is set
      Serial.println(" +");
    else
      Serial.println(" -");
  }
  if ((source & 0x40)==0x40)  // If AxZ bit is set
  {
    if ((source & 0x08)==0x08)  // If DPE (double puls) bit is set
      Serial.print("    Double Tap (2) on Z");
    else
      Serial.print("Single (1) tap on Z");
    if ((source & 0x04)==0x04)  // If PoIZ is set
      Serial.println(" +");
    else
      Serial.println(" -");
  }
}

/* This function will read the p/l source register and
   print what direction the sensor is now facing */
void portraitLandscapeHandler()
{
  byte pl = readRegister(0x10);  // Reads the PL_STATUS register
  switch((pl&0x06)>>1)  // Check on the LAPO[1:0] bits
  {
    case 0:
      Serial.print("Portrait up, ");
```

```
      break;
    case 1:
      Serial.print("Portrait Down, ");
      break;
    case 2:
      Serial.print("Landscape Right, ");
      break;
    case 3:
      Serial.print("Landscape Left, ");
      break;
  }
  if (pl&0x01)  // Check the BAFRO bit
    Serial.print("Back");
  else
    Serial.print("Front");
  if (pl&0x40)  // Check the LO bit
    Serial.print(", Z-tilt!");
  Serial.println();
}

/* Initialize the MMA8452 registers
   See the many application notes for more info on setting
   all of these registers:
   http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MMA8...

   Feel free to modify any values, these are settings that work well for me.
*/
void initMMA8452(byte fsr, byte dataRate)
{
  MMA8452Standby();  // Must be in standby to change registers

  /* Set up the full scale range to 2, 4, or 8g. */
  if ((fsr==2)||(fsr==4)||(fsr==8))
    writeRegister(0x0E, fsr >> 2);
  else
    writeRegister(0x0E, 0);

  /* Setup the 3 data rate bits, from 0 to 7 */
  writeRegister(0x2A, readRegister(0x2A) & ~(0x38));
  if (dataRate <= 7)
    writeRegister(0x2A, readRegister(0x2A) | (dataRate 3));

  /* Set up portrait/landscap registers - 4 steps:
     1. Enable P/L
     2. Set the back/front angle trigger points (z-lock)
     3. Set the threshold/hysteresis angle
     4. Set the debouce rate
  // For more info check out this app note: http://cache.freescale.com/files/sensors/doc/app_note
/AN4068.pdf */
  writeRegister(0x11, 0x40);  // 1. Enable P/L
  writeRegister(0x13, 0x44);  // 2. 29deg z-lock (don't think this register is actually writable)
  writeRegister(0x14, 0x84);  // 3. 45deg thresh, 14deg hyst (don't think this register is writable
either)
  writeRegister(0x12, 0x50);  // 4. debounce counter at 100ms (at 800 hz)

  /* Set up single and double tap - 5 steps:
     1. Set up single and/or double tap detection on each axis individually.
     2. Set the threshold - minimum required acceleration to cause a tap.
     3. Set the time limit - the maximum time that a tap can be above the threshold
     4. Set the pulse latency - the minimum required time between one pulse and the next
```

```
    5. Set the second pulse window -
 maximum allowed time between end of latency and start of second pulse
    for more info check out this app note: http://cache.freescale.com/files/sensors/doc/app_note/
AN4072.pdf */
  writeRegister(0x21, 0x7F);  // 1. enable single/double taps on all axes
  // writeRegister(0x21, 0x55);  // 1. single taps only on all axes
  // writeRegister(0x21, 0x6A);  // 1. double taps only on all axes
  writeRegister(0x23, 0x20);  // 2. x thresh at 2g, multiply the value by 0.0625g/LSB to get the
threshold
  writeRegister(0x24, 0x20);  // 2. y thresh at 2g, multiply the value by 0.0625g/LSB to get the
threshold
  writeRegister(0x25, 0x08);  // 2. z thresh at .5g, multiply the value by 0.0625g/LSB to get the
threshold
  writeRegister(0x26, 0x30);  // 3. 30ms time limit at 800Hz odr, this is very dependent on data
rate, see the app note
  writeRegister(0x27, 0xA0);  // 4. 200ms (at 800Hz odr) between taps min, this also depends on
the data rate
  writeRegister(0x28, 0xFF);  // 5. 318ms (max value) between taps max

  /* Set up interrupt 1 and 2 */
  writeRegister(0x2C, 0x02);  // Active high, push-pull interrupts
  writeRegister(0x2D, 0x19);  // DRDY, P/L and tap ints enabled
  writeRegister(0x2E, 0x01);  // DRDY on INT1, P/L and taps on INT2

  MMA8452Active();  // Set to active to start reading
}

/* Sets the MMA8452 to standby mode.
   It must be in standby to change most register settings */
void MMA8452Standby()
{
  byte c = readRegister(0x2A);
  writeRegister(0x2A, c & ~(0x01));
}

/* Sets the MMA8452 to active mode.
   Needs to be in this mode to output data */
void MMA8452Active()
{
  byte c = readRegister(0x2A);
  writeRegister(0x2A, c | 0x01);
}

/* Read i registers sequentially, starting at address
   into the dest byte arra */
void readRegisters(byte address, int i, byte * dest)
{
  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte((MMA8452_ADDRESS1));       // write 0xB4
  i2cWaitForComplete();

  i2cSendByte(address);        // write register address
  i2cWaitForComplete();

  i2cSendStart();
  i2cSendByte((MMA8452_ADDRESS1)|0x01);          // write 0xB5
  i2cWaitForComplete();
  for (int j=0; j<i; j++)
```

```
  {
    i2cReceiveByte(TRUE);
    i2cWaitForComplete();
    dest[j] = i2cGetReceivedByte();   // Get MSB result
  }
  i2cWaitForComplete();
  i2cSendStop();

  cbi(TWCR, TWEN);          // Disable TWI
  sbi(TWCR, TWEN);          // Enable TWI
}

/* read a single byte from address and return it as a byte */
byte readRegister(uint8_t address)
{
  byte data;

  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte((MMA8452_ADDRESS1));       // write 0xB4
  i2cWaitForComplete();

  i2cSendByte(address);        // write register address
  i2cWaitForComplete();

  i2cSendStart();

  i2cSendByte((MMA8452_ADDRESS1)|0x01);          // write 0xB5
  i2cWaitForComplete();
  i2cReceiveByte(TRUE);
  i2cWaitForComplete();

  data = i2cGetReceivedByte();        // Get MSB result
  i2cWaitForComplete();
  i2cSendStop();

  cbi(TWCR, TWEN);          // Disable TWI
  sbi(TWCR, TWEN);          // Enable TWI

  return data;
}

/* Writes a single byte (data) into address */
void writeRegister(unsigned char address, unsigned char data)
{
  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte((MMA8452_ADDRESS1));// write 0xB4
  i2cWaitForComplete();

  i2cSendByte(address);        // write register address
  i2cWaitForComplete();

  i2cSendByte(data);
  i2cWaitForComplete();

  i2cSendStop();
}
```

Irá abrir a programação. Agora clique em “UPLOAD” e depois abra o Serial Monitor como está mostrado na figura abaixo. No Serial Monitor irá mostrar os valores de X, Y e Z respectivamente.



Referência:

http://arduino.cc/en/

http://www.sparkfun.com/products/10955

79. Tutorial sobre Bluetooth e Arduino

O Bluetooth Mate Silver é um módulo bluetooth o qual pode-se acessá-lo por qualquer dispositivo bluetooth.

Assim que tirado da embalagem, ele vem com 6 conexões: RTS-0, RX-I, TX-0,VCC,CTS-I, GND. E vem com a seguinte configuração de fábrica:

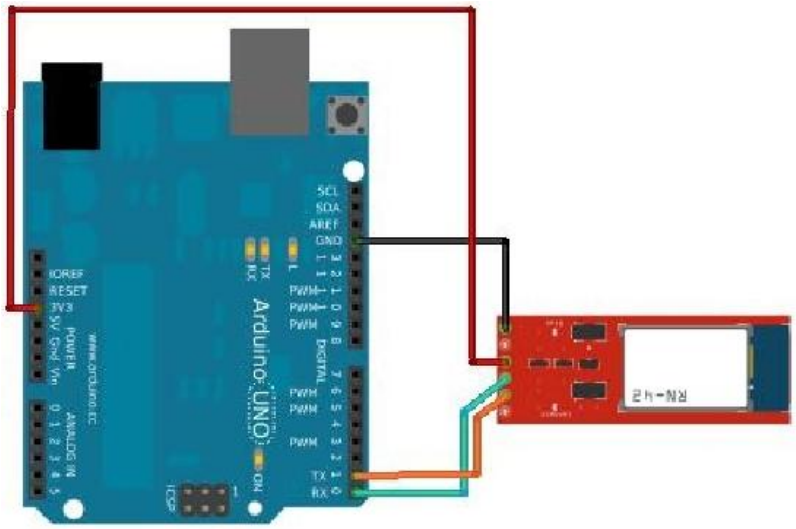
Baud Rate:115200, 8Bits, no parity(sem paridade), Stopbits:1, FlowControl:Xon/Xoff. Comando de configuração: $$$. Senha Padrão(PIN):1234.

Antes de mais nada, leia a documentação fornecida: Comandos de configuração.

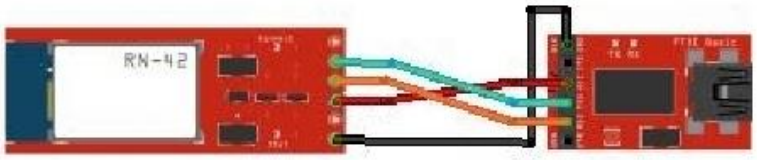
1º Passo:

Na documentação fala que é possível conectá-lo remotamente durante 60 segundos iniciais, mas aqui no LdG não conseguimos acessá-lo remotamente para configurá-lo. Para configurá-lo foi preciso utilizar o próprio Arduino ou um breakout FTDI como mostra as figuras abaixo:

OBS: Para utilizar o Arduino para configurar o Bluetooth, é necessário tirar o ATMEGA328P-PU da placa Arduino.

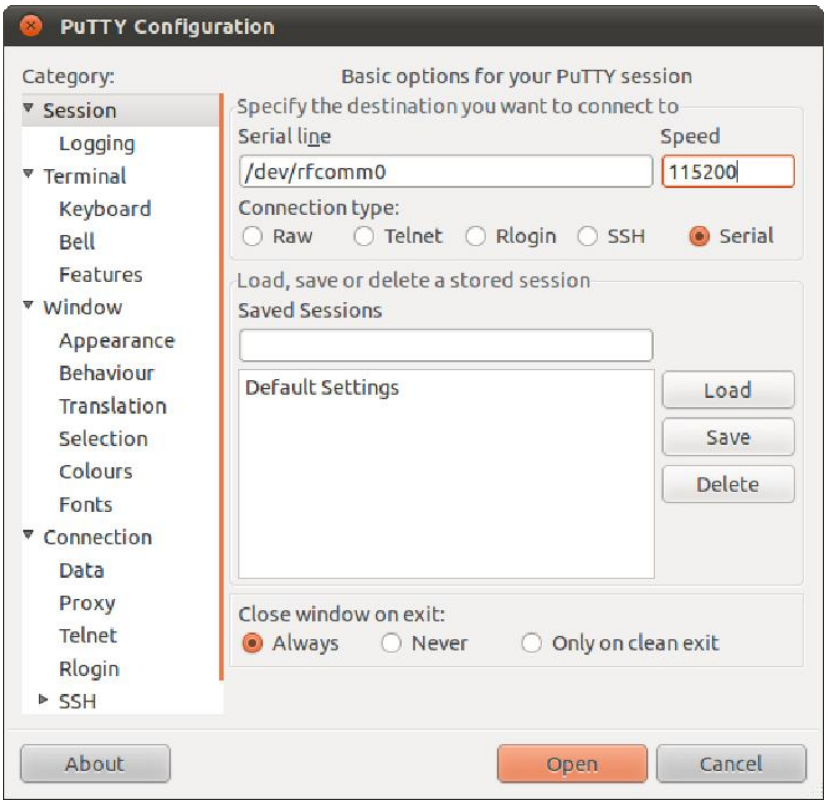


OU

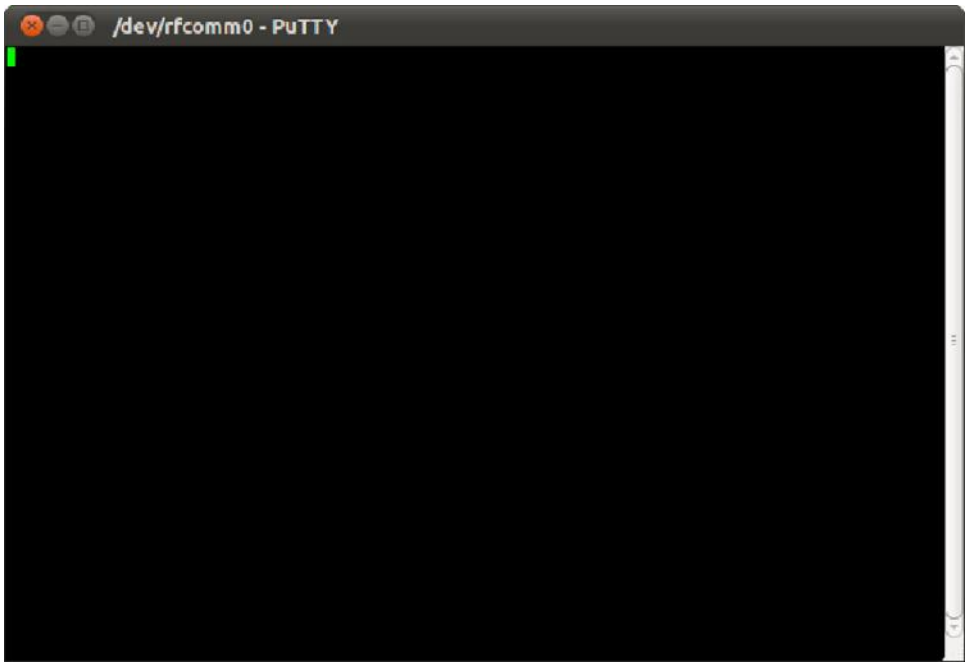


Agora, para acessar o bluetooth pelo Arduino ou FTDI, pode-se utilizar o Serial Monitor do próprio programa Arduino ou um programa de terminal como o PUTTY(http://www.chiark.greenend.org.uk/~sgtatham/putty/ Embutir em PUTTY) para configurar o Bluetooth Mate Silver.

A figura abaixo mostra o PUTTY. Com o PUTTY aberto, selecione SERIAL, depois coloque a porta em que o Bluetooth está conectado. No Windows verifique em qual porta está em Painel de Controle/Sistemas/Gerenciador de Dispositivos. No Linux e Mac deve ser as portas /dev/ttyUSBX ou ttyACMX, sendo “X” o número da porta.



Depois disso, clique em “OPEN” e mostrará uma janela como a figura abaixo:



Nessa janela digite “$$$” e ele responderá “CMD”, agora pode configurá-lo. Para sair da configuração digite “---”.

Depois de digitado “$$$”, caso queira saber o que está sendo digitado digite “+”. Para ficar mais fácil, mude o nome do bluetooth com o comando “SN, nome” e assim saberá qual será seu bluetooth. Agora digite “ST,255” para poder configurar tanto por serial a cabo ou pela conexão bluetooth. Digite “R,1” para reiniciar o bluetooth.

Agora, pode-se conectar diretamente pelo bluetooth e configurá-lo sem precisar de cabo.

Para configurar o bluetooth pela conexão por bluetooth. Conecte pelo bluetooth digitando o PIN: 1234(Padrão). Agora, verifique em qual porta o bluetooth está conectado. No Ubuntu, a porta se chama rfcommX, sendo 'X' o número referente a porta do bluetooth. No Windows, vá em painel de controle/sistema/dispositivos de hardware/ e a porta será 'COMX'

Arduino Pro Mini com Bluetooth

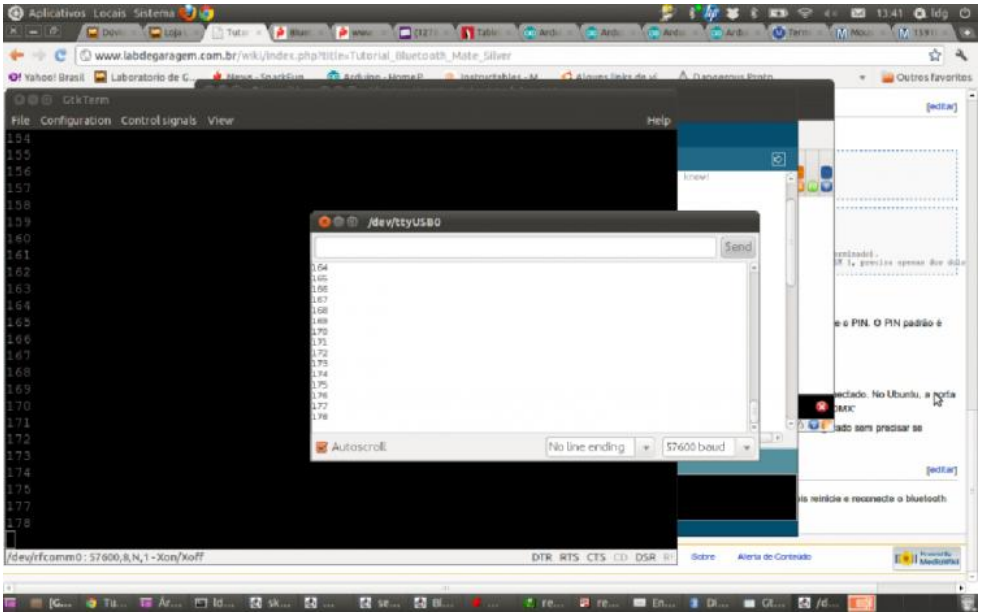
O Arduino Pro Mini está configurado com a velocidade de 57600, para mudar a velocidade do bluetooth digite “SU,57”, reinicie o bluetooth e conecte novamente. Abra a IDE do Arduino e cole a seguinte programação:

Fonte: http://www.sparkfun.com/tutorial/BluetoothMate-Quickstart/Bluetooth...

```
/* Bluetooth Mate Echo
 by: Jim Lindblom - jim at sparkfun.com
 date: 3/15/11
 license: CC-SA v3.0 - Use this code however you'd like, for any
 purpose. If you happen to find it useful, or make it better, let us know!
 Conexao necessaria:
 Bluetooth Mate----------------Arduino
    CTS-I    (Nao Conectado)
    VCC----------------------5V ou 3.3V
    GND----------------------GND
    TX-O---------------------D2
    RX-I---------------------D3
    RTS-O    (Nao Conectado)
*/
#include <NewSoftSerial.h>  // Conexao Serial para bluetooth mate
int bluetoothTx = 2;  // Pino TX-O of bluetooth mate para Arduino D2
int bluetoothRx = 3;  // Pino RX-I of bluetooth mate para Arduino D3
NewSoftSerial bluetooth(bluetoothTx, bluetoothRx); Cria as duas portas D2 e D3 em Serial
para conectar o bluetooth
int counter = 0;
int incomingByte;
void setup()
{
 Serial.begin(57600);  //Inicializa a Serial do Arduino com 57600
 bluetooth.begin(57600);  // Inicializa a comunicacao com o Bluetooth Mate com 57600
 delay(100);  // atraso de 100 milisegundos
```

```
}
void loop()
{
 if (Serial.available() > 0) { //Se a Serial estiver disponivel
   incomingByte = Serial.read(); //Le o comando vindo da serial
   if (incomingByte == '0') { //Se o comando for '0' (zero)
     Serial.println("RESET");
     bluetooth.println("RESET");
     counter=0; // Reseta contador
   }
 }
 Serial.println(counter);   // Mostra o contador na serial do Arduino
 bluetooth.println(counter);  //Mostra o contador pela conexao bluetooth
 counter++;
 delay(250);
}
```

Agora conecte o bluetooth pelo conexao bluetooth e utilize o Putty e configure-o para a porta da conexao bluetooth para ver suas saídas. A figura abaixo mostra o resultado:



Na figura mostra o terminal da porta rfcomm0 referente da conexao do bluetooth e o Serial Monitor do arduino conectado pelo USB0. As duas portas mostram o mesmo contador. Se digitar '0' pelo Serial monitor do arduino, o contador reseta e mostrará tanto na Serial Monitor quanto na conexao bluetooth.

Passando uma programação para o Arduino Pro Mini via bluetooth

Conecte o RX-I do Arduino no TX-O do Bluetooth Mate Silver e o TX-O do Arduino no RX-I do Bluetooth Mate Silver. Alimente o Bluetooth Mate Silver de 3,3V a 6V.

OBS1: Caso seu sistema operacional seja linux, a IDE do Arduino mostra apenas as portas tty* disponíveis, portanto digite o seguinte comando como superusuario ou root no terminal: ln -sf /dev/rfcommX /dev/ttyUSBY. Este comando reflete a porta rfcommX para ttyUSBY. Sendo 'X' o número da porta referente a conexão do bluetooth e 'Y' um número qualquer à escolha do usuário.

OBS2: Esse comando: ln -sf /dev/rfcommX /dev/ttyUSBY, tem que ser digitado no terminal toda vez que o computador for reiniciado.

Feito isso, abra a IDE do Arduino, em Tools/Boards/ selecione a placa Arduino Pro or Pro Mini. Depois em Tools/Ports/ selecione a porta da conexão bluetooth. Se quiser testar a conexão do bluetooth com o arduino passe a seguinte programação:

```
void setup()
{
Serial.begin(57600); //Inicializa a porta serial com 57600
}
void loop()
{
if (Serial.available()>0)
 {
   Serial.read();
   if(Serial.read()!='NULL')  //Se a entrada serial for diferente de nulo
   {
   Serial.println("OK");  // Retorne 'OK'
   }
   delay(500);    //atraso de 500 milisegundos
 }
}
```

No Arduino Pro Mini é necessário ficar apertado o 'RESET' até que apareça: 'Binary Sketch Size....' na IDE do Arduino. Quando aparecer esta frase, solte o botão 'RESET' do Arduino. No final irá aparecer 'Done Uploading'. Agora abra o Serial Monitor da IDE do Arduino e selecione: "No ending line" e "57600". Agora digite qualquer caractere no Serial Monitor e aperte o botão 'Enter' do teclado, o Serial Monitor deverá retornar 'OK'.

Referências:

http://arduino.cc/playground/

http://loja.labdegaragem.com.br/bluetooth-mate-prata.html

http://www.labdegaragem.com.br/wiki/index.php?title=Tutorial_Blueto...

http://www.sparkfun.com/products/10393

80. Tutorial de como utilizar Ultrasom com Arduino

O Ultrasom é um sensor que pode ser utilizado como medidor de distância, detecção de objetos e entre outros.

Neste tutorial, mostraremos como utilizar o Ultrasom da Maxbotix com Arduino.

Este Ultrasom pode ser usado por comunicação Serial (RS-232, portanto não conecte diretamente no Arduino senão irá queimá-lo), por tensão analógica e por comprimento de pulso (PWM). Aqui mostraremos apenas o uso por tensão analógica e por comprimento de pulso(PWM). A menor distância que o ultrasom consegue detectar é de 20cm e a maior distância é de 6,5m.

O Ultrasom da Maxbotix tem seis furos, um GND, um 5V, um TX, um RX, um AN, um PW e um BW. Aqui só usaremos o GND, 5V, AN e PW.

Para utilizá-lo por tensão analógica, conecte o GND no GND do Arduino, o 5V no 5V do Arduino e o AN no pino A0 do Arduino.

A programação está demonstrada abaixo:

```
//Feel free to use this code.
//Please be respectful by acknowledging the author in the code if you use or modify it.
//Author: Bruce Allen
//Date: 23/07/09
//Analog pin 0 for reading in the analog voltage from the MaxSonar device.
//This variable is a constant because the pin will not change throughout execution of this code.
const int anPin = 0;
//variables needed to store values
long anVolt, inches, cm;
int sum=0;//Create sum variable so it can be averaged
int avgrange=60;//Quantity of values to average (sample size)
void setup() {
//This opens up a serial connection to shoot the results back to the PC console
Serial.begin(9600);
}
void loop() {
```

```
pinMode(anPin, INPUT);
//MaxSonar Analog reads are known to be very sensitive. See the Arduino forum for more infor
mation.
//A simple fix is to average out a sample of n readings to get a more consistant reading.\\
//Even with averaging I still find it to be less accurate than the pw method.\\
//This loop gets 60 reads and averages them
for(int i = 0; i < avgrange ; i++) {
//Used to read in the analog voltage output that is being sent by the MaxSonar device. //Scale fa
ctor is (Vcc/512) per inch. A 5V supply yields ~9.8mV/in
anVolt = analogRead(anPin);
sum += anVolt;
delay(10);
}

inches = sum/avgrange; cm = inches * 2.54;
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
//reset sample total
sum = 0;
delay(500);
}
```

Agora, abra o **Serial** Monitor e mostrará os valores em polegadas (in) e em centimêtros(cm).
Agora, demonstraremos o funcionamento por PWM. A vantagem de utilizar por PWM é que este é mais preciso do que por tensão analógica.
conecte o GND no GND do Arduino, o 5V no 5V do Arduino e o PW no pino 7 do Arduino.
A programação está demonstrada abaixo:

```
//Feel free to use this code.
//Please be respectful by acknowledging the author in the code if you use or modify it.
//Author: Bruce Allen
//Date: 23/07/09
//Digital pin 7 for reading in the pulse width from the MaxSonar device.
//This variable is a constant because the pin will not change throughout execution of this code.
const int pwPin = 7;
//variables needed to store values
long pulse, inches, cm;
void setup() {

//This opens up a serial connection to shoot the results back to the PC console
Serial.begin(9600);
}
void loop() {
pinMode(pwPin, INPUT);
//Used to read in the pulse that is being sent by the MaxSonar device.
//Pulse Width representation with a scale factor of 147 uS per Inch.
pulse = pulseIn(pwPin, HIGH); //147uS per inch
inches = pulse/147; //change inches to centimetres
cm = inches * 2.54;
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(500);
}
```

Agora, abra a Serial Monitor e mostrará os valores em polegadas(in) e em centimêtros (cm).

E pronto! Agora conseguimos medir distância com um arduino e um ultrasom! Até a próxima!

Referências:

http://www.arduino.cc/playground/Main/MaxSonar

http://labdegarag1.lojatemporaria.com/arduino-original/arduino-uno....

http://labdegarag1.lojatemporaria.com/sensores/sensor-de-distancia-...

81. Tutorial testando o transceptor RFM12B-S2 Wireless Transceiver com Arduino

Para a utilização do transceptor, é ncessário dois ou mais transceptores para se comunicarem!

E é necessário um Arduino para cada transceptor!

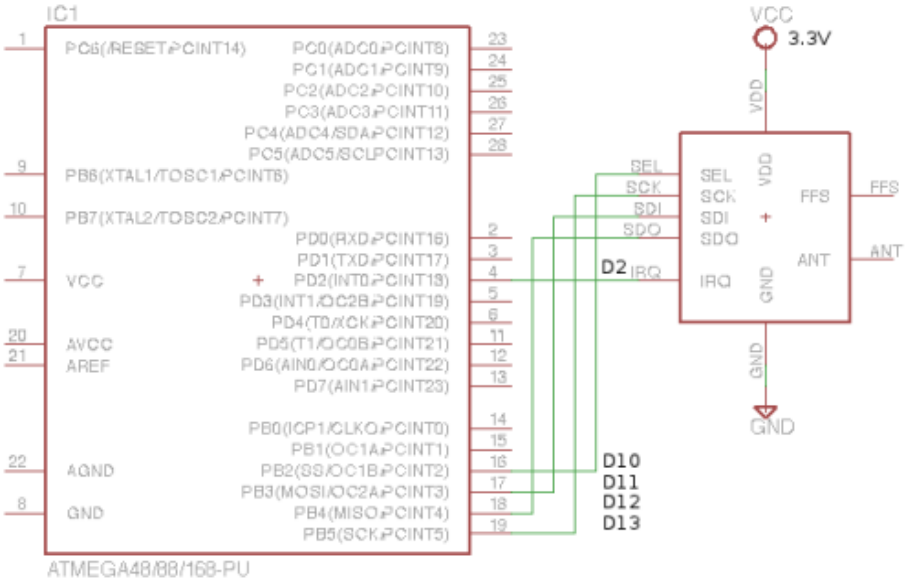
Antes de começar, é necessário baixar uma biblioteca para Arduino. A biblioteca utilizada aqui foi a do JeeLabs, clique aqui para baixar. OBS: Esta biblioteca funciona apenas com a versão 1.0 da IDE do Arduino.

Baixado a biblioteca, extraia para a pasta "libraries" dentro da pasta da IDE do Arduino-1.0. Agora localize a pasta extraída e mude o nome para RFM12 para ficar mais fácil localizar.

Agora tem que conectar o transceptor no arduino. Aqui no LdG fizemos uma placa para poder conectar na protoboard. Os arquivos da placa para eagle são esses: BreakoutRFM12.brd, BreakoutRFM12.sch.

Depois de soldado na placa, temos que ver a pinagem do transceptor no datasheet.

Abaixo mostra o esquemático para conectar o transceptor ao Arduino:



Onde D2 é a porta digital 2 do Arduino, o D10 é a porta digital 10 do Arduino e assim por diante. Agora faça uma antena de EXATAMENTE 173mm, descasque apenas um pedaço bem pequeno(não desencape totalmente o cabo) e solde direto no transceptor. (É preferivel que a antena seja de cabo de rede, por exemplo)

Lembre-se que a alimentação do transceptor é de 3.3V, portanto não erre!

Lembrando também que é necessário utiizar um arduino para cada transceptor. Portanto é necessário de dois ou mais para se comunicarem!

Depois de feito as conexões, abra a IDE do Arduino-1.0 e abra o exemplo localizado em File/Examples/RFM12/RF12/RF12Demo. Irá abrir a programação abaixo:

```
// Configure some values in EEPROM for easy config of the RF12 later on.
// 2009-05-06 <jc@wippler.nl> http://opensource.org/licenses/mit-license.php
// this version adds flash memory support, 2009-11-19
#include <JeeLib.h>
#include <util/crc16.h>
#include <util/parity.h>
#include <avr/eeprom.h>
#include <avr/pgmspace.h>
#define DATAFLASH 1 // check for presence of DataFlash memory on JeeLink
#define FLASH_MBIT 16 // support for various dataflash sizes: 4/8/16 Mbit
#define LED_PIN 9 // activity LED, comment out to disable
#define COLLECT 0x20 // collect mode, i.e. pass incoming without sending acks
static unsigned long now () {
// FIXME 49-day overflow
return millis() / 1000;
}
static void activityLed (byte on) {
#ifdef LED_PIN
pinMode(LED_PIN, OUTPUT);
digitalWrite(LED_PIN, !on);
#endif
```

```
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// RF12 configuration setup code
typedef struct {
byte nodeId;
byte group;
char msg[RF12_EEPROM_SIZE-4];
word crc;
} RF12Config;
static RF12Config config;
static char cmd;
static byte value, stack[RF12_MAXDATA], top, sendLen, dest, quiet;
static byte testbuf[RF12_MAXDATA], testCounter;
static void addCh (char* msg, char c) {
byte n = strlen(msg);
msg[n] = c;
}
static void addInt (char* msg, word v) {
if (v >= 10)
addInt(msg, v / 10);
addCh(msg, '0' + v % 10);
}
static void saveConfig () {
// set up a nice config string to be shown on startup
memset(config.msg, 0, sizeof config.msg);
strcpy(config.msg, " ");

byte id = config.nodeId & 0x1F;
addCh(config.msg, '@' + id);
strcat(config.msg, " i");
addInt(config.msg, id);
if (config.nodeId & COLLECT)
addCh(config.msg, '*');

strcat(config.msg, " g");
addInt(config.msg, config.group);

strcat(config.msg, " @ ");
static word bands[4] = { 315, 433, 868, 915 };
word band = config.nodeId >> 6;
addInt(config.msg, bands[band]);
strcat(config.msg, " MHz ");

config.crc = ~0;
for (byte i = 0; i < sizeof config - 2; ++i)
config.crc = _crc16_update(config.crc, ((byte*) &config)[i]);
// save to EEPROM
for (byte i = 0; i < sizeof config; ++i) {
byte b = ((byte*) &config)[i];
eeprom_write_byte(RF12_EEPROM_ADDR + i, b);
}

if (!rf12_config())
Serial.println("config save failed");
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// OOK transmit code
// Turn transmitter on or off, but also apply asymmetric correction and account
// for 25 us SPI overhead to end up with the proper on-the-air pulse widths.
// With thanks to JGJ Veken for his help in getting these values right.
```

```
static void ookPulse(int on, int off) {
rf12_onOff(1);
delayMicroseconds(on + 150);
rf12_onOff(0);
delayMicroseconds(off - 200);
}
static void fs20sendBits(word data, byte bits) {
if (bits == 8) {
++bits;
data = (data 1) | parity_even_bit(data);
}
for (word mask = bit(bits-1); mask != 0; mask >>= 1) {
int width = data & mask ? 600 : 400;
ookPulse(width, width);
}
}
static void fs20cmd(word house, byte addr, byte cmd) {
byte sum = 6 + (house >> 8) + house + addr + cmd;
for (byte i = 0; i < 3; ++i) {
fs20sendBits(1, 13);
fs20sendBits(house >> 8, 8);
fs20sendBits(house, 8);
fs20sendBits(addr, 8);
fs20sendBits(cmd, 8);
fs20sendBits(sum, 8);
fs20sendBits(0, 1);
delay(10);
}
}
static void kakuSend(char addr, byte device, byte on) {
int cmd = 0x600 | ((device - 1) 4) | ((addr - 1) & 0xF);
if (on)
cmd |= 0x800;
for (byte i = 0; i < 4; ++i) {
for (byte bit = 0; bit < 12; ++bit) {
ookPulse(375, 1125);
int on = bitRead(cmd, bit) ? 1125 : 375;
ookPulse(on, 1500 - on);
}
ookPulse(375, 375);
delay(11); // approximate
}
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// DataFlash code
#if DATAFLASH
#define DF_ENABLE_PIN 8 // PB0
#if FLASH_MBIT == 4
// settings for 0.5 Mbyte flash in JLv2
#define DF_BLOCK_SIZE 16 // number of pages erased at same time
#define DF_LOG_BEGIN 32 // first 2 blocks reserved for future use
#define DF_LOG_LIMIT 0x0700 // last 64k is not used for logging
#define DF_MEM_TOTAL 0x0800 // 2048 pages, i.e. 0.5 Mbyte
#define DF_DEVICE_ID 0x1F44 // see AT25DF041A datasheet
#define DF_PAGE_ERASE 0x20 // erase one block of flash memory
#endif
#if FLASH_MBIT == 8
// settings for 1 Mbyte flash in JLv2
#define DF_BLOCK_SIZE 16 // number of pages erased at same time
#define DF_LOG_BEGIN 32 // first 2 blocks reserved for future use
```

```
#define DF_LOG_LIMIT 0x0F00 // last 64k is not used for logging
#define DF_MEM_TOTAL 0x1000 // 4096 pages, i.e. 1 Mbyte
#define DF_DEVICE_ID 0x1F45 // see AT26DF081A datasheet
#define DF_PAGE_ERASE 0x20 // erase one block of flash memory
#endif
#if FLASH_MBIT == 16
// settings for 2 Mbyte flash in JLv3
#define DF_BLOCK_SIZE 256 // number of pages erased at same time
#define DF_LOG_BEGIN 512 // first 2 blocks reserved for future use
#define DF_LOG_LIMIT 0x1F00 // last 64k is not used for logging
#define DF_MEM_TOTAL 0x2000 // 8192 pages, i.e. 2 Mbyte
#define DF_DEVICE_ID 0x2020 // see M25P16 datasheet
#define DF_PAGE_ERASE 0xD8 // erase one block of flash memory
#endif
// structure of each page in the log buffer, size must be exactly 256 bytes
typedef struct {
byte data [248];
word seqnum;
long timestamp;
word crc;
} FlashPage;
// structure of consecutive entries in the data area of each FlashPage
typedef struct {
byte length;
byte offset;
byte header;
byte data[RF12_MAXDATA];
} FlashEntry;
static FlashPage dfBuf; // for data not yet written to flash
static word dfLastPage; // page number last written
static byte dfFill; // next byte available in buffer to store entries
static byte df_present () {
return dfLastPage != 0;
}
static void df_enable () {
// digitalWrite(ENABLE_PIN, 0);
bitClear(PORTB, 0);
}
static void df_disable () {
// digitalWrite(ENABLE_PIN, 1);
bitSet(PORTB, 0);
}
static byte df_xfer (byte cmd) {
SPDR = cmd;
while (!bitRead(SPSR, SPIF))
;
return SPDR;
}
void df_command (byte cmd) {
for (;;) {
cli();
df_enable();
df_xfer(0x05); // Read Status Register
byte status = df_xfer(0);
df_disable();
sei();
// don't wait for ready bit if there is clearly no dataflash connected
if (status == 0xFF || (status & 1) == 0)
break;
}
```

```
cli();
df_enable();
df_xfer(cmd);
}
static void df_deselect () {
df_disable();
sei();
}
static void df_writeCmd (byte cmd) {
df_command(0x06); // Write Enable
df_deselect();
df_command(cmd);
}
void df_read (word block, word off, void* buf, word len) {
df_command(0x03); // Read Array (Low Frequency)
df_xfer(block >> 8);
df_xfer(block);
df_xfer(off);
for (word i = 0; i < len; ++i)
((byte*) buf)[(byte) i] = df_xfer(0);
df_deselect();
}
void df_write (word block, const void* buf) {
df_writeCmd(0x02); // Byte/Page Program
df_xfer(block >> 8);
df_xfer(block);
df_xfer(0);
for (word i = 0; i < 256; ++i)
df_xfer(((const byte*) buf)[(byte) i]);
df_deselect();
}
// wait for current command to complete
void df_flush () {
df_read(0, 0, 0, 0);
}
static void df_wipe () {
Serial.println("DF W");

df_writeCmd(0xC7); // Chip Erase
df_deselect();
df_flush();
}
static void df_erase (word block) {
Serial.print("DF E ");
Serial.println(block);

df_writeCmd(DF_PAGE_ERASE); // Block Erase
df_xfer(block >> 8);
df_xfer(block);
df_xfer(0);
df_deselect();
df_flush();
}
static word df_wrap (word page) {
return page < DF_LOG_LIMIT ? page : DF_LOG_BEGIN;
}
static void df_saveBuf () {
if (dfFill == 0)
return;
dfLastPage = df_wrap(dfLastPage + 1);
```

```
if (dfLastPage == DF_LOG_BEGIN)
++dfBuf.seqnum; // bump to next seqnum when wrapping

// set remainder of buffer data to 0xFF and calculate crc over entire buffer
dfBuf.crc = ~0;
for (byte i = 0; i < sizeof dfBuf - 2; ++i) {
if (dfFill <= i && i < sizeof dfBuf.data)
dfBuf.data[i] = 0xFF;
dfBuf.crc = _crc16_update(dfBuf.crc, dfBuf.data[i]);
}

df_write(dfLastPage, &dfBuf);
dfFill = 0;

// wait for write to finish before reporting page, seqnum, and time stamp
df_flush();
Serial.print("DF S ");
Serial.print(dfLastPage);
Serial.print(' ');
Serial.print(dfBuf.seqnum);
Serial.print(' ');
Serial.println(dfBuf.timestamp);

// erase next block if we just saved data into a fresh block
// at this point in time dfBuf is empty, so a lengthy erase cycle is ok
if (dfLastPage % DF_BLOCK_SIZE == 0)
df_erase(df_wrap(dfLastPage + DF_BLOCK_SIZE));
}
static void df_append (const void* buf, byte len) {
//FIXME the current logic can't append incoming packets during a save!
// fill in page time stamp when appending to a fresh page
if (dfFill == 0)
dfBuf.timestamp = now();

long offset = now() - dfBuf.timestamp;
if (offset >= 255 || dfFill + 1 + len > sizeof dfBuf.data) {
df_saveBuf();
dfBuf.timestamp = now();
offset = 0;
}
// append new entry to flash buffer
dfBuf.data[dfFill++] = offset;
memcpy(dfBuf.data + dfFill, buf, len);
dfFill += len;
}
// go through entire log buffer to figure out which page was last saved
static void scanForLastSave () {
dfBuf.seqnum = 0;
dfLastPage = DF_LOG_LIMIT - 1;
// look for last page before an empty page
for (word page = DF_LOG_BEGIN; page < DF_LOG_LIMIT; ++page) {
word currseq;
df_read(page, sizeof dfBuf.data, &currseq, sizeof currseq);
if (currseq != 0xFFFF) {
dfLastPage = page;
dfBuf.seqnum = currseq + 1;
} else if (dfLastPage == page - 1)
break; // careful with empty-filled-empty case, i.e. after wrap
}
}
```

```
static void df_initialize () {
// assumes SPI has already been initialized for the RFM12B
df_disable();
pinMode(DF_ENABLE_PIN, OUTPUT);
df_command(0x9F); // Read Manufacturer and Device ID
word info = df_xfer(0) 8;
info |= df_xfer(0);
df_deselect();
if (info == DF_DEVICE_ID) {
df_writeCmd(0x01); // Write Status Register ...
df_xfer(0); // ... Global Unprotect
df_deselect();
scanForLastSave();

Serial.print("DF I ");
Serial.print(dfLastPage);
Serial.print(' ');
Serial.println(dfBuf.seqnum);

// df_wipe();
df_saveBuf(); //XXX
}
}
static void discardInput () {
while (Serial.read() >= 0)
;
}
static void df_dump () {
struct { word seqnum; long timestamp; word crc; } curr;
discardInput();
for (word page = DF_LOG_BEGIN; page < DF_LOG_LIMIT; ++page) {
if (Serial.read() >= 0)
break;
// read marker from page in flash
df_read(page, sizeof dfBuf.data, &curr, sizeof curr);
if (curr.seqnum == 0xFFFF)
continue; // page never written to
Serial.print(" df# ");
Serial.print(page);
Serial.print(" : ");
Serial.print(curr.seqnum);
Serial.print(' ');
Serial.print(curr.timestamp);
Serial.print(' ');
Serial.println(curr.crc);
}
}
static word scanForMarker (word seqnum, long asof) {
word lastPage = 0;
struct { word seqnum; long timestamp; } last, curr;
last.seqnum = 0xFFFF;
// go through all the pages in log area of flash
for (word page = DF_LOG_BEGIN; page < DF_LOG_LIMIT; ++page) {
// read seqnum and timestamp from page in flash
df_read(page, sizeof dfBuf.data, &curr, sizeof curr);
if (curr.seqnum == 0xFFFF)
continue; // page never written to
if (curr.seqnum >= seqnum && curr.seqnum < last.seqnum) {
last = curr;
lastPage = page;
```

```
}
if (curr.seqnum == last.seqnum && curr.timestamp <= asof)
lastPage = page;
}
return lastPage;
}
static void df_replay (word seqnum, long asof) {
word page = scanForMarker(seqnum, asof);
Serial.print("r: page ");
Serial.print(page);
Serial.print(' ');
Serial.println(dfLastPage);
discardInput();
word savedSeqnum = dfBuf.seqnum;
while (page != dfLastPage) {
if (Serial.read() >= 0)
break;
page = df_wrap(page + 1);
df_read(page, 0, &dfBuf, sizeof dfBuf); // overwrites ram buffer!
if (dfBuf.seqnum == 0xFFFF)
continue; // page never written to
// skip and report bad pages
word crc = ~0;
for (word i = 0; i < sizeof dfBuf; ++i)
crc = _crc16_update(crc, dfBuf.data[i]);
if (crc != 0) {
Serial.print("DF C? ");
Serial.print(page);
Serial.print(' ');
Serial.println(crc);
continue;
}
// report each entry as "R seqnum time <data...>"
byte i = 0;
while (i < sizeof dfBuf.data && dfBuf.data[i] < 255) {
if (Serial.available())
break;
Serial.print("R ");
Serial.print(dfBuf.seqnum);
Serial.print(' ');
Serial.print(dfBuf.timestamp + dfBuf.data[i++]);
Serial.print(' ');
Serial.print((int) dfBuf.data[i++]);
byte n = dfBuf.data[i++];
while (n-- > 0) {
Serial.print(' ');
Serial.print((int) dfBuf.data[i++]);
}
Serial.println();
}
// at end of each page, report a "DF R" marker, to allow re-starting
Serial.print("DF R ");
Serial.print(page);
Serial.print(' ');
Serial.print(dfBuf.seqnum);
Serial.print(' ');
Serial.println(dfBuf.timestamp);
}
dfFill = 0; // ram buffer is no longer valid
dfBuf.seqnum = savedSeqnum + 1; // so next replay will start at a new value
```

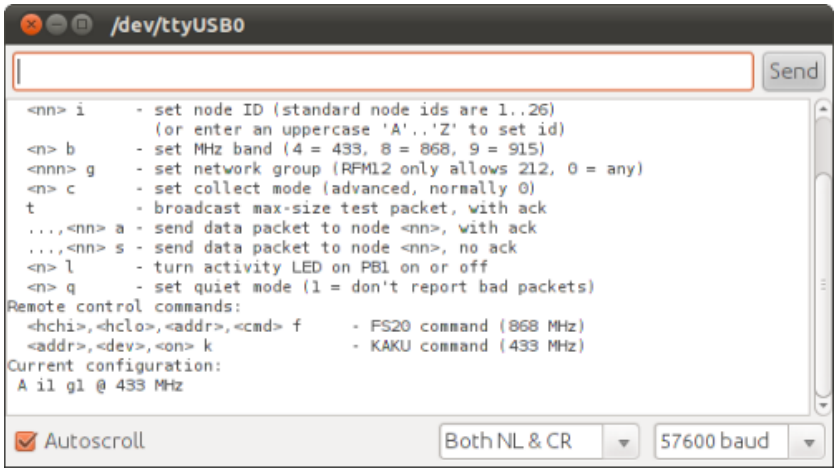
```
Serial.print("DF E ");
Serial.print(dfLastPage);
Serial.print(' ');
Serial.print(dfBuf.seqnum);
Serial.print(' ');
Serial.println(millis());
}
#else // DATAFLASH
#define df_present() 0
#define df_initialize()
#define df_dump()
#define df_replay(x,y)
#define df_erase(x)
#endif
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
char helpText1[] PROGMEM =
"\n"
"Available commands:" "\n"
" <nn> i - set node ID (standard node ids are 1..26)" "\n"
" (or enter an uppercase 'A'..'Z' to set id)" "\n"
" <n> b - set MHz band (4 = 433, 8 = 868, 9 = 915)" "\n"
" <nnn> g - set network group (RFM12 only allows 212, 0 = any)" "\n"
" <n> c - set collect mode (advanced, normally 0)" "\n"
" t - broadcast max-size test packet, with ack" "\n"
" ...,<nn> a - send data packet to node <nn>, with ack" "\n"
" ...,<nn> s - send data packet to node <nn>, no ack" "\n"
" <n> l - turn activity LED on PB1 on or off" "\n"
" <n> q - set quiet mode (1 = don't report bad packets)" "\n"
"Remote control commands:" "\n"
" <hchi>,<hclo>,<addr>,<cmd> f - FS20 command (868 MHz)" "\n"
" <addr>,<dev>,<on> k - KAKU command (433 MHz)" "\n"
;
char helpText2[] PROGMEM =
"Flash storage (JeeLink only):" "\n"
" d - dump all log markers" "\n"
" <sh>,<sl>,<t3>,<t2>,<t1>,<t0> r - replay from specified marker" "\n"
" 123,<bhi>,<blo> e - erase 4K block" "\n"
" 12,34 w - wipe entire flash memory" "\n"
;
static void showString (PGM_P s) {
for (;;) {
char c = pgm_read_byte(s++);
if (c == 0)
break;
if (c == '\n')
Serial.print('\r');
Serial.print(c);
}
}
static void showHelp () {
showString(helpText1);
if (df_present())
showString(helpText2);
Serial.println("Current configuration:");
rf12_config();
}
static void handleInput (char c) {
if ('0' <= c && c <= '9')
value = 10 * value + c - '0';
else if (c == ',') {
```

```
if (top < sizeof stack)
stack[top++] = value;
value = 0;
} else if ('a' <= c && c <='z') {
Serial.print("> ");
Serial.print((int) value);
Serial.println(c);
switch (c) {
default:
showHelp();
break;
case 'i': // set node id
config.nodeId = (config.nodeId & 0xE0) + (value & 0x1F);
saveConfig();
break;
case 'b': // set band: 4 = 433, 8 = 868, 9 = 915
value = value == 8 ? RF12_868MHZ :
value == 9 ? RF12_915MHZ : RF12_433MHZ;
config.nodeId = (value 6) + (config.nodeId & 0x3F);
saveConfig();
break;
case 'g': // set network group
config.group = value;
saveConfig();
break;
case 'c': // set collect mode (off = 0, on = 1)
if (value)
config.nodeId |= COLLECT;
else
config.nodeId &= ~COLLECT;
saveConfig();
break;
case 't': // broadcast a maximum size test packet, request an ack
cmd = 'a';
sendLen = RF12_MAXDATA;
dest = 0;
for (byte i = 0; i < RF12_MAXDATA; ++i)
testbuf[i] = i + testCounter;
Serial.print("test ");
Serial.println((int) testCounter); // first byte in test buffer
++testCounter;
break;
case 'a': // send packet to node ID N, request an ack
case 's': // send packet to node ID N, no ack
cmd = c;
sendLen = top;
dest = value;
memcpy(testbuf, stack, top);
break;
case 'l': // turn activity LED on or off
activityLed(value);
break;
case 'f': // send FS20 command: <hchi>,<hclo>,<addr>,<cmd>f
rf12_initialize(0, RF12_868MHZ);
activityLed(1);
fs20cmd(256 * stack[0] + stack[1], stack[2], value);
activityLed(0);
rf12_config(); // restore normal packet listening mode
break;
case 'k': // send KAKU command: <addr>,<dev>,<on>k
```

```
rf12_initialize(0, RF12_433MHZ);
activityLed(1);
kakuSend(stack[0], stack[1], value);
activityLed(0);
rf12_config(); // restore normal packet listening mode
break;
case 'd': // dump all log markers
if (df_present())
df_dump();
break;
case 'r': // replay from specified seqnum/time marker
if (df_present()) {
word seqnum = (stack[0] 8) || stack[1];
long asof = (stack[2] 8) || stack[3];
asof = (asof 16) | ((stack[4] 8) || value);
df_replay(seqnum, asof);
}
break;
case 'e': // erase specified 4Kb block
if (df_present() && stack[0] == 123) {
word block = (stack[1] 8) | value;
df_erase(block);
}
break;
case 'w': // wipe entire flash memory
if (df_present() && stack[0] == 12 && value == 34) {
df_wipe();
Serial.println("erased");
}
break;
case 'q': // turn quiet mode on or off (don't report bad packets)
quiet = value;
break;
}
value = top = 0;
memset(stack, 0, sizeof stack);
} else if ('A' <= c && c <= 'Z') {
config.nodeId = (config.nodeId & 0xE0) + (c & 0x1F);
saveConfig();
} else if (c > ' ')
showHelp();
}
void setup() {
Serial.begin(57600);
Serial.print("\n[RF12demo.8]");
if (rf12_config()) {
config.nodeId = eeprom_read_byte(RF12_EEPROM_ADDR);
config.group = eeprom_read_byte(RF12_EEPROM_ADDR + 1);
} else {
config.nodeId = 0x41; // node A1 @ 433 MHz
config.group = 0xD4;
saveConfig();
}
df_initialize();

showHelp();
}
void loop() {
if (Serial.available())
handleInput(Serial.read());
```

```
if (rf12_recvDone()) {
byte n = rf12_len;
if (rf12_crc == 0) {
Serial.print("OK");
} else {
if (quiet)
return;
Serial.print(" ?");
if (n > 20) // print at most 20 bytes if crc is wrong
n = 20;
}
if (config.group == 0) {
Serial.print("G ");
Serial.print((int) rf12_grp);
}
Serial.print(' ');
Serial.print((int) rf12_hdr);
for (byte i = 0; i < n; ++i) {
Serial.print(' ');
Serial.print((int) rf12_data[i]);
}
Serial.println();

if (rf12_crc == 0) {
activityLed(1);

if (df_present())
df_append((const char*) rf12_data - 2, rf12_len + 2);
if (RF12_WANTS_ACK && (config.nodeId & COLLECT) == 0) {
Serial.println(" -> ack");
rf12_sendStart(RF12_ACK_REPLY, 0, 0);
}

activityLed(0);
}
}
if (cmd && rf12_canSend()) {
activityLed(1);
Serial.print(" -> ");
Serial.print((int) sendLen);
Serial.println(" b");
byte header = cmd == 'a' ? RF12_HDR_ACK : 0;
if (dest)
header |= RF12_HDR_DST | dest;
rf12_sendStart(header, testbuf, sendLen);
cmd = 0;
activityLed(0);
}
}
```

Faça o UPLOAD para o Arduino. Assim que terminar, abra o Serial monitor e configure para 57600. E irá aparecer o seguinte:



No Serial Monitor, irá mostrar a configuração atual do transceptor. Cada transceptor precisa ter uma ID diferente, para mudar a ID digite <N>i, sendo <N> o número desejado para este transceptor. Outra configuração necessária é o grupo e a frequência de operação. O grupo e a frequência de operação devem ser os mesmo para todos os transceptores. No caso da imagem do Serial Monitor, ele está configurado com a ID 1, grupo 1 e frequência 433Mhz.

Para cada transceptor, defina a ID, o grupo e a frequencia!

Esta programação é apenas um teste para mostrar que os transceptores estão se comunicando.

Como teste digite: "123,<N>a" ou "123,<N>s" sendo <N> a ID do transceptor que deseja mandar. Por exemplo, se você está conectado com o transceptor ID 1 e quer mandar para o ID 2, digite: "123,2a" e vice-versa.

E é isso aí! Fique ligado que iremos soltar mais tutoriais sobre o RFM12B-S2! Boa sorte e boa diversão!

Referências:

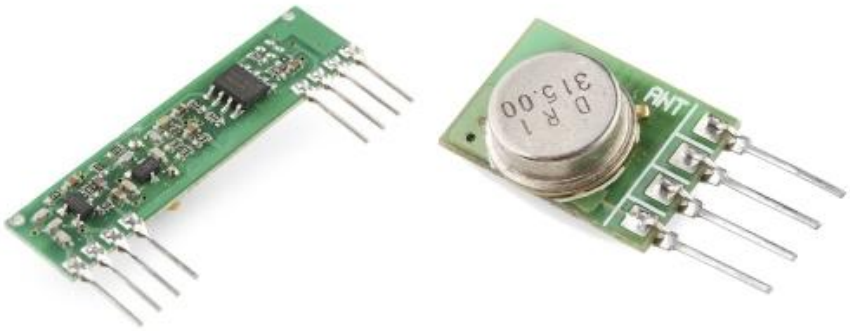
http://labdegarag1.lojatemporaria.com/comunicacao/transceptor-sem-f...

http://www.sparkfun.com/products/9582

http://jeelabs.org/2009/02/10/rfm12b-library-for-arduino/

https://github.com/jcw/jeelib

82. Tutorial transmissor e receptor RF Link com Arduino

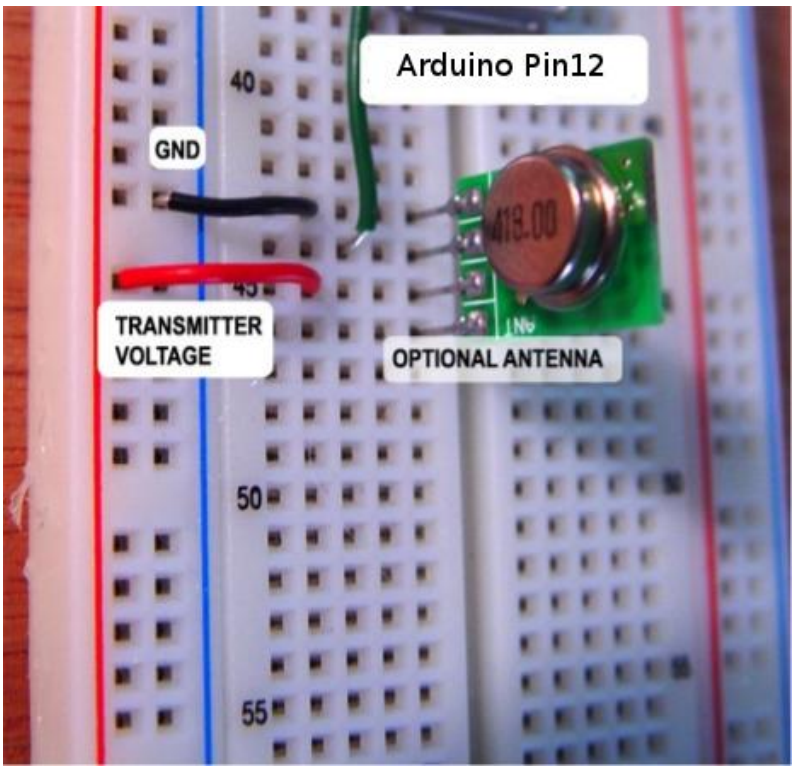


No mercado existe vários tipos e meios de comunicação para usar com Arduino. Neste tutorial usaremos os RF Link Receiver - 4800bps(315Mhz) e RF Link Transmitter - 315Mhz e testaremos a comunicção entre eles utilizando a biblioteca Virtual Wire e a documentação disponível aqui.

Neste tutorial vamos mostrar como mandar uma mensagem utilizando o RF Link Transmitter - 315Mhz com um Arduino e recebendo a mensagem utilizando o RF Link Receiver - 4800bps(315Mhz) com outro Arduino.

Antes de mais nada, baixe a biblioteca Virtual Wire e extraia para a pasta "libraries" localizada dentro da pasta da IDE do Arduino. Nesta biblioteca é bem fácil de usar, pois já existe um tratamento de erros para filtrar os erros que chegam no receptor.

Agora para montar o emissor RF Link Transmitter - 315Mhz com Arduino, faça a seguinte ligação:



Onde:

- O Fio vermelho vai para o VCC ou 5V
- O Fio preto vai para o GND

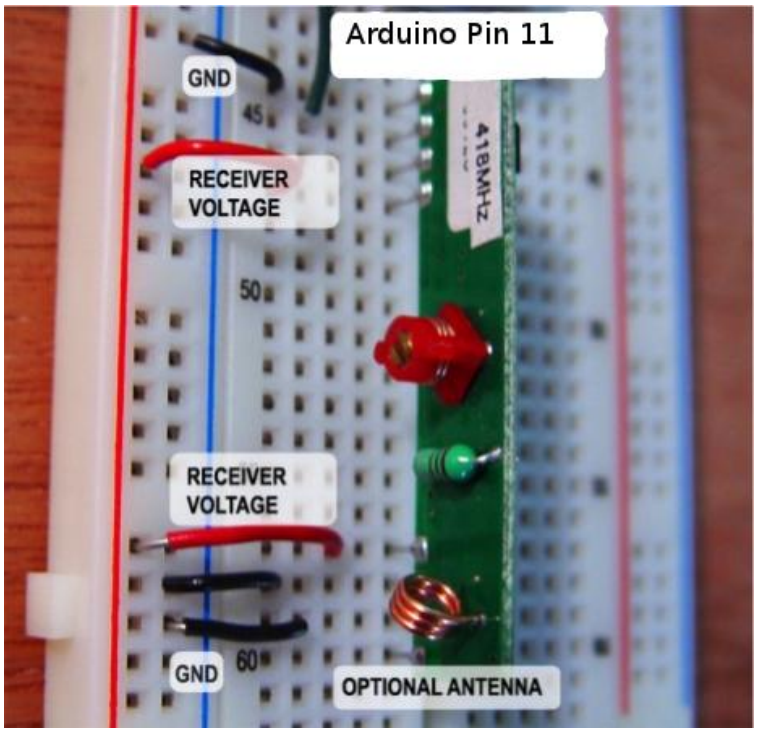
- O FIo verde vai para o pino digital 12 do Arduino
- Por opção, pode-se colocar uma antena para melhor captação de dados. O comprimento da antena é de cerca de 25cm

Agora, abra a IDE do Arduino e vá em File/Examples/VirtualWire e selecione "transmitter",conecte o Arduino, selecione a versão do Arduino(UNO, Duemilanove,etc) e clique em UPLOAD. Abaixo mostra a programação:

```
// transmitter.pde
//
// Simple example of how to use VirtualWire to transmit messages
// Implements a simplex (one-way) transmitter with an TX-C1 module
//
// See VirtualWire.h for detailed API docs
// Author: Mike McCauley (mikem@open.com.au)
// Copyright (C) 2008 Mike McCauley
// $Id: transmitter.pde,v 1.3 2009/03/30 00:07:24 mikem Exp $
#include <VirtualWire.h>
void setup()
{
Serial.begin(9600);                    // Debugging only
Serial.println("setup");
// Initialise the IO and ISR
vw_set_ptt_inverted(true);             // Required for DR3100
vw_setup(2000);                        // Bits per sec
}
void loop()
{
const char *msg = "hello";
digitalWrite(13, true);                // Flash a light to show transmitting
vw_send((uint8_t *)msg, strlen(msg));  //Send the message
vw_wait_tx();                          // Wait until the whole message is gone
digitalWrite(13, false);
delay(200);
}
```

A programação "transmitter" é um exemplo básico onde manda a palavra "hello" para o receptor.

Pronto, agora vamos montar o RF Link Receiver - 4800bps(315Mhz):



Onde:

- O fio vermelho vai para o VCC ou 5V
- O fio preto vai para GND
- O fio verde vai para o pino digital 11 do Arduino
- Por opção, pode-se colocar uma antena para melhor captação de dados. O comprimento da antena é de cerca de 25cm

Agora, com outro Arduino e com a IDE do Arduino aberto, vá em File/Examples/ e clique em "receiver" e faça o mesmo procedimento que o do emissor. Abaixo mostra a programação do "receiver":

```
// receiver.pde
//
// Simple example of how to use VirtualWire to receive messages
// Implements a simplex (one-way) receiver with an Rx-B1 module
//
// See VirtualWire.h for detailed API docs
// Author: Mike McCauley (mikem@open.com.au)
// Copyright (C) 2008 Mike McCauley
// $Id: receiver.pde,v 1.3 2009/03/30 00:07:24 mikem Exp $
#include <VirtualWire.h>
void setup()
{
Serial.begin(9600); // Debugging only
Serial.println("setup");
// Initialise the IO and ISR
vw_set_ptt_inverted(true); // Required for DR3100
vw_setup(2000); // Bits per sec
vw_rx_start(); // Start the receiver PLL running
}
void loop()
{
uint8_t buf[VW_MAX_MESSAGE_LEN];
uint8_t buflen = VW_MAX_MESSAGE_LEN;
if (vw_get_message(buf, &buflen)) // Non-blocking
```

```
{
int i;
digitalWrite(13, true); // Flash a light to show received good message
// Message with a good checksum received, dump it.
Serial.print("Got: ");

for (i = 0; i < buflen; i++)
{
Serial.print(buf[i], HEX);
Serial.print(" ");
}
Serial.println("");
digitalWrite(13, false);}
}
```

Este exemplo mostra o receiver recebendo a informação do emissor, que nesse caso é o "hello". Ao abrir o Serial monitor do Arduino-1.0, você verá que os dados que chegam são números, estes números são números da tabela ASCII. Você pode ver que o número hexadecimal "68" equivale a letra "h".

E é isso!!! Esperamos que tenham gostado!!! Caso tenham dúvida sobre este tutorial, postem aqui mesmo neste blog. Vocês podem ver outros tutoriais, clicando aqui. Temos a seção de projetos também que vocês podem ver e postar seus projetos! Até a próxima!!

Referências:

http://www.labdegaragem.org/loja/index.php/38-comunicacao/rf-link-r...

http://www.labdegaragem.org/loja/index.php/38-comunicacao/rf-link-t...

http://arduino.cc/playground/Main/InterfacingWithHardware#Communica...

http://www.sparkfun.com/products/10533

http://www.open.com.au/mikem/arduino/VirtualWire.pdf

http://www.open.com.au/mikem/arduino/VirtualWire-1.9.zip

http://www.sparkfun.com/datasheets/RF/KLP_Walkthrough.pdf

http://winavr.scienceprog.com/example-avr-projects/running-tx433-an...

83. Tutorial: como utilizar o Color LCD Shield com Arduino

O Color LCD Shield é um shield com Nokia 6100 LCD a qual você conecta no arduino e assim você pode gerar imagens com arduino. Neste tutorial vamos mostrar como o shield funciona e como utilizá-lo com Arduino.

O Color LCD Shield pode suportar imagens de 132x132 pixels. Caso sua imagem seja maior, reduza-a até esse tamanho. Como as imagens precisam estar em hexadecimal, então fizemos um programa em PYTHON (é necessário instalá-lo no sistema operacional) para converter imagens .bmp em hexadecimal. O código está demonstrado abaixo:

```
import Image
import ImageColor
image='/home/ldg/Documentos/Python/logopbmp.bmp' #Coloque aqui o nome do arquivo que deseja converter
texto='/home/ldg/Documentos/Python/Imagem.txt' #Crie um arquivo texto em branco em um diretório a sua escolha e coloque aqui o nome do arquivo texto para guardar a matriz da imagem em hexadecimal
f=open(texto,'w') #Irá abrir o arquivo texto
imopen=Image.open(image) #Abrirá a imagem
x=y=0
h=0
while y<52: #Coloque aqui a altura da imagem. A imagem do Ldg tem 125x51 pixels
#print "y"
x=0
while x<125: # Coloque aqui o comprimento da imagem.
#print "x"
z=imopen.getpixel((x,y))
h=0
h = h+((z[0]/16))
h = h+((z[1]/16))*16
```

```
h = h+((z[2]/16))*16*16
h=hex(h)
#print h
f.write(str(h))
f.write(',')
x=x+1
y=y+1
f.write('\n')
f.close()
```

Copie e cole em um editor de texto (gedit, bloco de notas, notepad, etc) e salve o arquivo com a extensão .py. Agora, abra o prompt de comando ou terminal e digite: python nomedocodigo.py e irá converter a imagem .bmp em .txt.

Gerado o arquivo .txt abra-o e copie o conteúdo. Aqui fizemos com um logo do LdG.

Neste tutorial foi utilizado a biblioteca Color LCD Shield. Extraia a biblioteca na pasta "libraries" localizada dentro da pasta da IDE do Arduino. Depois disso,abra a IDE do arduino, cole o conteúdo que você copiou do arquivo txt no código abaixo:

```
#include <ColorLCDShield.h> //Inclui a biblioteca do Color LCD Shield
#include <avr/pgmspace.h> //Inclui a biblioteca do avr para gravar a matriz da imagem na memória flash
LCDShield lcd; //Define o lcd
byte cont = 40; // Contraste
void setup()
{
//Cuidado, se não funcionar, troque PHILLIPS por EPSON (tudo maiusculo)
lcd.init(PHILLIPS); // Inicializa o LCD
lcd.contrast(cont); // Contraste
lcd.clear(WHITE); // Fundo branco
printldg(); // Coloca a imagem do logo LdG
testPattern(); // Coloca varios retangulos de varias cores
}
void loop()
{

}
void testPattern()
{
lcd.setRect(80, 2, 131, 19, 1, WHITE); //Coloca retangulo branco
lcd.setRect(80, 19, 131, 35, 1, YELLOW); //Coloca retangulo amarelo
lcd.setRect(80, 35, 131, 51, 1, CYAN); //Coloca retangulo ciano
lcd.setRect(80, 51, 131, 67, 1, GREEN); //Coloca retangulo verde
lcd.setRect(80, 67, 131, 83, 1, MAGENTA); //Coloca retangulo magenta
lcd.setRect(80, 83, 131, 99, 1, RED); //Coloca retangulo vermelho
lcd.setRect(80, 99, 131, 115, 1, BLUE); //Coloca retangulo azul
lcd.setRect(80, 115, 131, 131, 1, BLACK); //Coloca retangulo preto
}
PROGMEM static unsigned int logoldg[6550] = { //A matriz da imagem contendo elementos hexadecimais no formato correto para o LCD
0xfff,0xfff,0xfff,0xfff,0xfff,0xeee,0x666,0x333,0x0,0x0,0x0,0x0,0x0,0x0,0x333,0x777,0xefe,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfee,0x666,0x322,0x0,0x0,0x0,0x0,
0x0,0x0,0x222,0x566,0xcdd,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xccc,0x555,0x333,0x0,0x0,0x0,0x0,0x0,0x222,0x444,0xa9a,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0x999,0x0,0x0,0x0,0x13,0x35,0x34,0x35,0x35,0x35,0x12,0x0,0x0,0x0,0xaaa,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xa99,0x0,0x0,0x0,0x13,0x36,0x36,0x36,
0x36,0x35,0x23,0x0,0x0,0x0,0x788,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x555,0x0,0x0,0x0,0x23,0x24,0x24,0x24,0x24,0x24,0x1,0x0,
```

0x222,0xeee,0xfff,0xfff,0xfff,0x888,0x767,0x999,0x9a9,0xbbb,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xccc,0x999,0x899,0x889,0x999,0xbbb,0xeee,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xaaa,0x0,0x12,0x8d,0x1af,0x19f,0x19f,0x9f,0x9f,0x9f,0xaf,0x8d,0x1,0x0,
0xbbb,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xbbb,0x0,0x1,0x18d,0xaf,0x19f,0x19f,0x9f,
0x9f,0x9f,0x9f,0x9e,0x23,0x0,0x999,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0x555,0x0,0x36,0x19f,0x9f,0x19f,0xaf,0x19f,0x9f,0x9f,0x19f,
0x59,0x0,0x111,0xede,0x767,0x0,0x0,0x0,0x1,0x0,0x0,0x112,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,
0x999,0x10,0x0,0x0,0x0,0x0,0x0,0x110,0x0,0x0,0x655,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0x0,0x11,0xaf,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x1,0x100,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x0,0x1,0x19f,0x9f,0x19f,0x9f,0x19f,0x9f,0x19f,0x9f,
0x9f,0x9f,0x19f,0x12,0x0,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xeee,0xccc,0xbbb,0xbbb,0xbbb,0xddd,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,
0xbbb,0x0,0x46,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x17b,0x0,0x0,0x0,0x12,0x7b,
0x19f,0x9f,0x9f,0x69,0x0,0x0,0xddd,0xfff,0xeee,0xbbb,0x555,0x0,0x12,0x16a,0x18d,0x8e,0x9e,0x18d,
0x17c,0x136,0x0,0x0,0xbbb,0xaaa,0xaaa,0xaaa,0xaaa,0xccc,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xaaa,0x0,0x17b,0x9f,0x9f,0x9f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x6a,0x0,0x988,
0xeee,0xeee,0xeee,0xeee,0xeee,0x99a,0x0,0x69,0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x19f,0x9f,0x19f,0x9f,
0x9f,0x7d,0x0,0x0,0x100,0x455,0xdcd,0xfff,0xeee,0xeee,0xdee,0xdee,0xeee,0xeee,0xeee,0xfff,0xccc,0x555,
0x111,0x100,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x666,0xddd,0xede,0xeee,0x433,0x0,0xaf,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x19f,0x24,0x12,0x7b,0xaf,0x9f,0x19f,0x9f,0x9f,0x9f,0x9e,0x0,0x0,0x100,0x0,
0x0,0x0,0x36,0x9f,0x8e,0x18e,0x8e,0x9f,0x8d,0x18e,0x9f,0x7b,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x333,
0xaaa,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0x666,0x0,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x18f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x9f,0x9f,0x9f,0x19f,0x8f,0x9f,0x8f,0x9f,0x9f,0x19f,0x9f,0x19f,0x9f,0x6a,0x48,0x124,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x111,0x145,0x59,0x16a,0x6a,0x6a,0x6a,0x6a,0x58,0x23,0x0,
0x0,0x0,0x0,0x0,0x0,0x35,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x8e,0x19f,0x9f,0x19f,
0x9f,0x9f,0x19f,0x19f,0x9f,0x9f,0x6a,0x12,0x46,0x16a,0x6a,0x6b,0x9f,0x9f,0x9f,0x8e,0x19e,0x19f,0x17d,0x8d,
0x18e,0x9f,0x8d,0x17b,0x7b,0x7b,0x7b,0x7b,0x16b,0x6a,0x146,0x11,0x0,0x111,0xcdd,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0x344,0x11,0x19f,0x9f,0x19f,0x9f,0x7cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x19f,0x0,0x24,
0x48,0x48,0x48,0x48,0x48,0x48,0x58,0x9f,0x18f,0x9f,0x8f,0x5af,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x9f,0x9f,
0x18f,0x8f,0x9f,0x9f,0x36,0x36,0x47,0x58,0x48,0x48,0x58,0x48,0x47,0x24,0x47,0x8f,0x18e,0x18e,0x8f,0x9f,0x9f,
0x19f,0x8f,0x9f,0x19f,0x17b,0x48,0x48,0x48,0x48,0x7b,0x9f,0x9f,0x9f,0x8f,0xcef,0xfff,0xfff,0xeff,0x9f,0x9f,
0x9f,0x19f,0x9f,0x19f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x18f,0x8e,0x18d,0x8e,0x9f,0x9f,0x9f,
0x19f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x19f,0x9f,0x8f,0x8e,0x9f,0x19f,0x6a,0x0,0x0,0xddd,0xfff,0xfff,0xfff,
0xfff,0xfff,0x344,0x2,0x9f,0x19f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,

0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x19f,0x5af,0xfff,0xfff,0xfff,0x4bf,0x9f,0x19f,0x9f,0x9f,0
x9f,0x8e,
0x8e,0x8f,0x9f,0x9f,0x19f,0x8e,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x7d,0x8d,0x8e,0
x9f,0x9f,0x9f,
0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x8f,0xdef,0xfff,0xfff,0xeff,0x9f,0x
19f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x3af,0x5bf,0x9f,0x19f,0x9f,0x9f,0x8f,0x7d,0x7d,0x8f,0x19f,0x9f,0
x19f,0x9f,
0x29f,0x8cf,0x7cf,0x5bf,0x9f,0x19f,0x9f,0x9f,0x19f,0x19f,0x19f,0x9f,0x9f,0x8f,0x18e,0x18f,0x9
e,0x8f,0x18e,0x1,0x0,0xfff,0xfff,0xfff,
0xfff,0xfff,0x434,0x11,0x1af,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x19f,
0x9f,
0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x5bf,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x9
f,0x9f,
0x8e,0x8f,0x9f,0x19f,0x9f,0x9f,0x8e,0x19f,0x9f,0x9f,0x9f,0x8f,0x9f,0x19f,0x9f,0x9f,0x9f,0x7d,0
x18d,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x8f,0xdef,0xfff,0xfff,0xeff
,0x8f,
0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x2af,0xdef,0xfff,0xcef,0x8f,0x9f,0x19f,0x9f,0x9f,0x8f,0x7d,0x8f
,0x9f,0x9f,
0x9f,0x9f,0x7bf,0xfff,0xfff,0xeff,0x8f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x8f,0x8e,0x8e,
0x8f,0x8f,0x18f,0x7d,
0x0,0x666,0xfff,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x5af,0xfff,0xfff,0xfff,0x6bf,0x2af,0x2af,0x29f,
0x2af,
0x29f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x19f,0x19f,0x8f,0x19f,0
x29f,0x2af,
0x2af,0x2af,0x29f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8f,0xdef,0xfff,0xfff,0
xeff,0x2af,
0x2af,0x2af,0x9f,0x9f,0x9f,0x9f,0x8cf,0xeff,0x5bf,0x8f,0x19f,0x9f,0x9f,0x9f,0x19f,0x9f,0x29f,0x
2af,0x2af,0x2af,
0x2af,0x19f,0x5bf,0xeff,0xeff,0xcef,0x8f,0x9f,0x9f,0x29f,0x2af,0x2af,0x2af,0x29f,0x2af,0x2af,0x
18e,0x8e,0x8e,
0x18f,0x8f,0x9f,0x35,0x0,0xfff,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x9f,0x
19f,0x9f,
0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x5af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0
xfff,0xfff,
0xfff,0xfff,0x3af,0x9f,0x9f,0x8f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x8f,0x8f,0x6bf,0
xfff,0xfff,
0xfff,0xfff,0xfff,0xadf,0x9f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0xdef,
0xfff,0xfff,
0xfff,0xfff,0xfff,0xdff,0x8f,0x19f,0x9f,0x19f,0x19f,0x8f,0x19f,0x19f,0x19f,0x9f,0x9f,0x19f,0x9f,0x
cef,0xfff,
0xfff,0xfff,0xfff,0xfff,0x4bf,0x8f,0x9f,0x8f,0x8f,0x9f,0x39f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0x8cf,0x7e,0x8e,0x18e,0x8f,0x17b,0x0,0xbbb,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x8f,0x
9f,0x29f,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xbef,0x9f,0x4af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xcef,0x8f,0x19f,0xeff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8df,0x9f,0x29f,0xfff,0
xfff,
0xfff,0xfff,0xfff,0xfff,0xadf,0x8f,0xcef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x3af,0x8f,0xdef,0x
fff,
0xfff,0xfff,0xfff,0xfff,0xcef,0x8f,0x6bf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x4af,0x9f,0x8cf,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0x3af,0x7bf,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xff
f,
0xfff,0xfff,0xfff,0x29e,0x8d,0x8e,0x18e,0x18f,0x0,0x888,0xfff,

0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x19f,0
x9f,0x29f,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x7cf,0x5af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xeff,0x8f,0xcdf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x4af,0x6bf,0xfff,0xfff,0xfff,0xff
f,
0xfff,0xfff,0xadf,0x8f,0xcef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff
,
0xfff,0xfff,0xdff,0x18f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xeff,0x8f,0xdef,0xfff,0xfff,0xff
f,
0xfff,0xfff,0xfff,0x3af,0x6bf,0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff
,
0xfff,0x4bf,0x8e,0x8e,0x18e,0x19f,0x1,0x888,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x19f,0
x9f,0x39f,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9df,0x5af,0xfff,0xfff,0xfff,0xadf,0x8cf,0x8cf,0x9cf,0xfff,
0xfff,
0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x5bf,0x6bf,0xfff,0xfff,0xfff,0xb
ef,0x8cf,
0x8cf,0x5bf,0x8f,0xdef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x
7cf,0x8cf,
0x6cf,0x29f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x9df,0x
8cf,0x9cf,
0x19f,0x7bf,0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xfff,0x8cf,0x8cf,0x8cf,0xdef,0xfff,0xfff,0xfff,0x4b
f,0x19f,
0x9f,0x9f,0x19f,0x1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x19f,0x9f,0x9f,
0x8f,0x8f,0x9f,0x8f,0xdef,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0x5bf,0x9f,0x9f,0x8f,0xcdf,0xfff,0
xfff,0xfff,0x8f,
0xeef,0xfff,0xfff,0xbef,0x29f,0x3af,0x3af,0xfff,0xfff,0xfff,0x6bf,0x6bf,0xfff,0xfff,0xfff,0x4bf,0x9f,0x
9f,0x9f,0x9f,
0x9f,0x8f,0x8f,0x9f,0x8f,0x8f,0x5bf,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x9f,0x9f,0x9f,0x3a
f,0xfff,0xfff,
0xfff,0x6bf,0x29f,0x29f,0x9cf,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xeff,0x8f,0x19f,0x9f,0x9f,0x7
bf,0xfff,0xfff,
0xeff,0x8f,0xfff,0xfff,0xfff,0xbef,0x8f,0x9f,0x9f,0x5bf,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x9f,0x9f,0x
1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x19f,0x9f,0
x9f,0x9f,0x7cf,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0x4bf,0x19f,0x19f,0x9f,0xbdf,
0xfff,
0xfff,0xfff,0x8f,0xeef,0xfff,0xfff,0x7cf,0x9f,0x9f,0x8f,0xfff,0xfff,0xfff,0x6cf,0x6bf,0xfff,0xfff,0xfff,0x
4bf,0x19f,
0x9f,0x9f,0x9f,0x29f,0xeef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xeff,0x
8f,0x19f,
0x9f,0x3af,0xfff,0xfff,0xfff,0x19f,0x19f,0x19f,0x4af,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x9f,
0x9f,0x19f,
0x9f,0x6bf,0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xbef,0x8f,0x19f,0x19f,0x5af,0xfff,0xfff,0xfff,0x4bf
,0x19f,0x9f,
0x9f,0x19f,0x1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x2af,0x9f,0x9f,0x9f,0x9f,0x19f,0
x9f,0x6af,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0x5bf,0x19f,0x9f,0x9f,0xcdf,0
xfff,0xfff,
0xfff,0x8f,0xeef,0xfff,0xfff,0x7cf,0x9f,0x9f,0x8f,0xeff,0xfff,0xfff,0x6cf,0x6bf,0xfff,0xfff,0xfff,0x4af,
0x9f,0x9f,0x9f,
0x8f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x8f,0x9f,0x9f,0
x39f,
0xfff,0xfff,0xfff,0x19f,0x9f,0x19f,0x4af,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x8f,0x9f,0x9f,0x
9f,0x7bf,

0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xbef,0x8f,0x9f,0x9f,0x5af,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x
9f,0xaf,0x1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0x29f,0x9f,0x9f,0x9f,0x9f,0x8f,0x
9f,0x9df,0xfff,
0xfff,0xfff,0xdff,0xeff,0xeff,0xfff,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x8f,0xbdf,
0xfff,0xfff,
0xfff,0x8f,0xeef,0xfff,0xfff,0x8cf,0x9f,0x19f,0x9f,0xeff,0xfff,0xfff,0x6cf,0x6bf,0xfff,0xfff,0xfff,0x4bf
,0x9f,0x19f,
0x19f,0x29f,0xfff,0xfff,0xfff,0xeff,0xeff,0xeff,0xeff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xeff,0x8f,
0x9f,0x9f,
0x3af,0xfff,0xfff,0xfff,0x19f,0x9f,0x9f,0x4af,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x8f,0x9f,0x
9f,0x9f,0x7bf,
0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xbef,0x8f,0x9f,0x9f,0x5af,0xfff,0xfff,0xfff,0x4bf,0x9f,0x9f,0x
9f,0x9f,0x1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0xfff,0xeff,0xfff,0xeff,0xeff,0xeff,0
x9f,0xbdf,0xfff,
0xfff,0xbef,0x8f,0x8f,0x8f,0xddf,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0x8cf,0x5bf,0x5bf,0x5bf,0x
eff,0xfff,0xfff,
0xfff,0x8f,0xeef,0xfff,0xfff,0x9df,0x8f,0x9f,0x19f,0xfff,0xfff,0xfff,0x6cf,0x6bf,0xfff,0xfff,0xfff,0x4bf,
0x9f,0x9f,
0x9f,0x2af,0xfff,0xfff,0xfff,0x19f,0x8f,0x8f,0x5af,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x5bf,0
x5bf,0x4bf,
0x39f,0xfff,0xfff,0xfff,0x3af,0x9f,0x8f,0x6bf,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff,0x8f,0x9f,0x
9f,0x9f,
0x7bf,0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xeff,0x5bf,0x5bf,0x4bf,0xbef,0xfff,0xfff,0xfff,0x4bf,0x9
f,0x9f,0x9f,
0x9f,0x1,0x444,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x8cf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9
f,0xbdf,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9df,0x4af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0x8f,0xeef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x6bf,0x6bf,0xfff,0xfff,0xff
f,
0x4af,0x9f,0x9f,0x19f,0x29f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0x
fff,0xfff,
0xfff,0xfff,0xcef,0x29f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xfff
,
0x8f,0x9f,0x9f,0x9f,0x7bf,0xfff,0xfff,0xeff,0x8f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,
0x4bf,0x19f,0x9f,0x19f,0x19f,0x1,0x444,0xfff,
0xfff,0xfff,0x444,0x11,0x1af,0x19f,0x9f,0x9f,0x7bf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0
x9f,0x9cf,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8cf,0x4af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xdef,0x8f,0xdef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x4bf,0x6bf,0xfff,0xfff,0x
fff,
0x4bf,0x9f,0x9f,0x9f,0x19f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xbdf,0xfff,0xff
f,0xfff,
0xfff,0xfff,0xcef,0x19f,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x8f,0xdef,0xfff,0xfff,0xef
f,
0x8f,0x19f,0x9f,0x9f,0x6bf,0xfff,0xfff,0xfff,0x8f,0xeef,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xff
f,0xfff,
0x2af,0x8e,0x8e,0x8e,0x9f,0x0,0x778,0xfff,
0xfff,0xfff,0x434,0x11,0xaf,0x9f,0x9f,0x9f,0x7cf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x9
f,0x19f,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xeff,0x19f,0x5bf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,
0xfff,0xfff,0x3bf,0x8f,0x4af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xdff,0x8f,0x6bf,0xfff,0xfff,0xf
ff,
0x4bf,0x9f,0x19f,0x19f,0x8f,0xadf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x7cf,0x9f,0x29f,0xfff,
0xfff,

0xfff,0xfff,0xfff,0xdef,0x8f,0xadf,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x7cf,0x8f,0xdef,0xfff,0x
fff,
0xfff,0x8f,0x9f,0x9f,0x9f,0x6bf,0xfff,0xfff,0xeff,0x8f,0x4af,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff
,
0xfff,0xaef,0x8e,0x8e,0x8e,0x18f,0x18e,0x0,0x888,0xfff,
0xfff,0xfff,0x555,0x12,0x9f,0x8f,0x8f,0x8f,0x29f,0x3af,0x2af,0x3af,0x3af,0x3af,0x3af,0x3af,0x3a
f,0x3af,0x9f,0x9f,
0x8f,0x2af,0x2af,0x3af,0x3af,0x3af,0x2af,0x29f,0x8f,0x9f,0x29f,0x3af,0x3af,0x3af,0x3af,0x3af,0
x3af,0x3af,0x3af,
0x3af,0x9f,0x18f,0x9f,0x19f,0x8f,0x3af,0x3af,0x3af,0x3af,0x3af,0x2af,0x29f,0x8f,0x9f,0x19f,0x3
af,0x3af,0x3af,
0x19f,0x19f,0x9f,0x19f,0x9f,0x8f,0x19f,0x2af,0x3af,0x2af,0x3af,0x2af,0x3af,0x8f,0x19f,0x9f,0x1
9f,0x8f,0x29f,
0x3af,0x3af,0x3af,0x2af,0x9f,0x8f,0x19f,0x3af,0x3af,0x3af,0x3af,0x3af,0x3af,0x19f,0x9f,0x9f,0x
3af,0x2af,0x3af,
0x2af,0x9f,0x19f,0x9f,0x19f,0x19f,0x3af,0x3af,0x3af,0x19f,0x9f,0x8f,0x2af,0x2af,0x2af,0x2af,0x
3af,0x3af,0x3af,
0x19f,0x8e,0x8e,0x18e,0x18e,0x8f,0x7b,0x0,0xbbc,0xfff,
0xfff,0xfff,0x455,0x0,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x9f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9
f,0x9f,0x19f,
0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19
f,0x19f,0x19f,
0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x19f,
0x9f,0x9f,0x19f,0x8e,0x18f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x8f,0x19f,
0x19f,0x19f,
0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x19f,0x19f,0x18f,
0x9f,0x9f,0x9f,
0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x9f,0x19f,0x18f,0x18
e,0x8d,0x18e,
0x8e,0x18f,0x9f,0x124,0x222,0xfff,0xfff,
0xfff,0xfff,0x999,0x0,0x17d,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0xaf,0
x19f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,
0x19f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x18e,0x8f,0x9f,0x9f,0x19
f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x
19f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x18f,0x8e,0x8f,0x8f,0x17b,0x0,0x999,0x
fff,0xfff,
0xfff,0xfff,0xfff,0x0,0x22,0x1af,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x9f,0x5a,0x0,0x0,0x
0,0x19d,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x9f,0x9f,0x19e,0
x2,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x6a,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19
f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0
x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x7b,0x0,0x111,0xfff,0xf
ff,0xfff,
0xfff,0xfff,0xfff,0x444,0x0,0x7c,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x
9e,0x9f,
0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x24,0x0,0x0
,0x0,0x0,0x0,

0x0,0x0,0x0,0x0,0x0,0x6a,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x19f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x8e,0x18e,0x12,0x211,0xfff,0xfff,0xfff,
0xfff,0xfff,0xbbb,0x111,0x47,0x9f,0x8e,0x8f,0x18f,0x8f,0x9f,0x9f,0x9f,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x9e,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x8e,0x8e,0x8e,0x9f,0x9f,0x9f,0x8f,0x8f,0x9f,0x9f,0x1,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x6a,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x7d,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x8f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x8f,0x8e,0x7d,0x7d,0x17d,0x8d,0x8e,0x111,0x455,0xfff,0xfff,
0xfff,0xfff,0x334,0x0,0x9f,0x18f,0x8e,0x8e,0x8e,0x8e,0x59,0x59,0x59,0x59,0x59,0x36,0x0,0x0,0x0,0x19e,0x9f,0x9f,
0x19f,0x9f,0x9f,0x9f,0x19f,0x8d,0x8e,0x8e,0x18e,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x6a,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x18f,0x9f,0x9f,0x59,0x59,0x59,0x59,0x169,
0x9f,0x9f,0x19f,0x9f,0x9f,0x19f,0x19f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x17c,0x59,0x58,0x59,0x59,0x59,0x59,0x59,
0x59,0x7b,0x9f,0x9f,0x19f,0x19f,0x19f,0x9f,0x9f,0x19f,0x9f,0x19f,0x9f,0x19f,0x19f,0x6b,0x59,0x59,0x69,0x59,0x159,
0x59,0x58,0x59,0x59,0x59,0x59,0x7b,0x8d,0x17c,0x8d,0x18e,0x18e,0x158,0x0,0xeee,0xfff,
0xfff,0xfff,0x0,0x135,0x19f,0x8e,0x18e,0x8e,0x16b,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x9e,0x9f,0x9f,0x1af,0xaf,
0xaf,0xaf,0xaf,0x1af,0x9f,0x8e,0x8e,0x18e,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x35,0xaf,0x9f,0x9f,0x9f,0x9f,0x9f,
0x19f,0x9f,0x9f,0xaf,0x1af,0x1af,0xaf,0xaf,0x1af,0x9f,0x19f,0x18f,0x9f,0x6b,0x0,0x0,0x0,0x0,0x0,0x0,0x12,0x9f,0x9f,0xaf,
0x1af,0x1af,0xaf,0x1af,0x1af,0x1af,0x9f,0x19f,0xaf,0x1,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x47,0x9f,0x9f,0x9f,0x9f,
0x1af,0x19f,0xaf,0x9f,0x19f,0x9f,0x9f,0x19f,0x1,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x12,0x8e,0x7d,
0x8d,0x18e,0x18d,0x100,0x999,0xfff,
0xfff,0xdcd,0x0,0x59,0x18f,0x8e,0x8e,0x9f,0x24,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x9e,0x19e,0x13,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x13,0x19f,0x8e,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x35,0x19f,0x9f,0x1af,0xaf,0xaf,0xaf,0xaf,0x9f,
0x7c,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x12,0x9e,0x9f,0x1,0x0,0x0,0x0,0x0,0x0,0x0,0x12,0x1af,0x35,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x47,0x19f,0x6b,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x47,0x9f,0x159,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x7b,0x9f,0x35,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x69,0x17c,0x7d,0x18e,0x8f,0x122,0x555,0xfff,
0xfff,0xddd,0x0,0x6a,0x8f,0x8e,0x8e,0x9f,0x2,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x9e,0x36,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x46,0x9f,0x8d,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x35,0x1af,0x35,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x7a,0x7c,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x34,0x19f,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x11,0x1af,0x35,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x7b,0x59,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x47,0x8d,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0xaf,0x24,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x57,0x8d,0x7d,0x18e,0x8e,0x234,0x555,0xfff,

0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0x0,0x0,0x11,0x9f,0x9f,0x19f,0x59,0x0,0x0,0x0,0x0,0
x19f,0x13,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x23,0x9f,0x18e,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x35,0x19f,0x35,0x
0,0x0,0x0,0x0,0x0,0x0,0x6a,
0x7c,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x11,0xaf,0x0,0x0,0x0,0x0,0x24,0x19f,0x9f,0x19f,0x9f,0x25,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x58,0x5a,0x0,0x0,0x0,0x0,0x7b,0x19f,0x19f,0x19f,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x0,0x
0,0x0,0x0,0x0,0x0,0x0,0x0,0x9e,
0x24,0x0,0x0,0x0,0x1,0x19f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x18d,0x19f,0x7d,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0
x9f,0xaf,0x12,0x0,0xfff,
0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0x0,0x0,0x23,0x9f,0x9f,0x19f,0x59,0x0,0x0,0x0,0x0,0
x19f,0x13,0x0,0x0,0x0,0x79,
0x19f,0x19f,0x6a,0x0,0x0,0x0,0x23,0xaf,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x35,0xaf,0x35,
0x0,0x0,0x0,0x0,0x0,0x0,0x7a,
0x9f,0x9e,0x1af,0xaf,0xaf,0xaf,0x7b,0x0,0x0,0x0,0x11,0xaf,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x9f,0
x9f,0xaf,0x1af,0x1af,0xaf,
0x1af,0x34,0x0,0x0,0x0,0x58,0x5a,0x0,0x0,0x0,0x0,0x18d,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0
x0,0x23,0x1af,0x9f,0x1af,0x1,
0x0,0x0,0x9e,0x24,0x0,0x0,0x0,0x11,0x1af,0x9f,0x6a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x19f,0x0,0x0,0x0,0x0,0x
57,0x9f,0x9f,0x9f,0xaf,0x12,0x0,0xfff,
0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0x0,0x0,0x23,0x19f,0x9f,0x19f,0x59,0x0,0x0,0x0,0x0,
0x19f,0x13,0x0,0x0,0x0,0x1,
0x1,0x1,0x0,0x0,0x0,0x0,0x24,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x45,0x19f,0x5a,0
x46,0x36,0x0,0x0,0x0,0x0,0x17a,
0x9f,0x23,0x0,0x1,0x0,0x1,0x0,0x0,0x0,0x0,0x11,0xaf,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x9f,0x7d,0
x1,0x1,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x58,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x0,0x1,0x1,0x
1,0x0,0x0,0x0,0x0,0x9f,0x24,0x0,
0x0,0x11,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0x9f,0xaf,0x1
2,0x0,0xfff,
0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0x0,0x0,0x23,0x19f,0x9f,0x19f,0x59,0x0,0x0,0x0,0x0,
0x19f,0x13,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x69,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x36,0x9f,0x19f,0x19
f,0x8e,0x0,0x0,0x0,0x7a,
0x159,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x11,0xaf,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x19f,0x12
,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x58,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x0,0x0,0x
0,0x0,0x0,0x0,0x0,0x11,0x1af,
0x24,0x0,0x0,0x0,0x11,0x19f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x19f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0
x9f,0xaf,0x12,0x0,0xfff,
0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x19f,0x9f,0x12,0x0,0x0,0x0,0x22,0x1af,0xaf,0xaf,0x6a,0x0,0x0,0x0,0x0,
0x9f,0x13,0x0,0x0,0x0,0x24,
0x36,0x36,0x36,0x36,0x7b,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x12,0x8c,0x7
d,0x7c,0x7a,0x0,0x0,0x0,0x0,0x7b,
0x35,0x0,0x0,0x0,0x22,0x36,0x36,0x24,0x0,0x0,0x0,0x11,0xaf,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x19f,0
x0,0x0,0x0,0x0,0x35,0x36,0x36,0x1,
0x0,0x0,0x0,0x59,0x59,0x0,0x0,0x0,0x0,0x17c,0xaf,0x19f,0x1af,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x1,0x
36,0x36,0x36,0x36,0x48,0x9e,0x9f,
0x24,0x0,0x0,0x0,0x11,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0x9f
,0xaf,0x12,0x0,0xfff,
0xfff,0xccc,0x0,0x6b,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0x0,0x0,0x0,0x1,0x1,0x1,0x0,0x0,0x0,0x0,0x0,0x9f,0
x13,0x0,0x0,0x0,0x45,0x6a,0x6a,
0x6a,0x6a,0x6a,0x8d,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0
x0,0x0,0x0,0x0,0x7b,0x35,0x0,0x0,
0x35,0x6a,0x6a,0x47,0x0,0x0,0x0,0x12,0x1af,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x
16a,0x6a,0x16a,0x13,0x0,0x0,
0x158,0x5a,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x11,0x6a,0x6a,0x6
a,0x6a,0x69,0x16b,0xaf,0x24,0x0,
0x0,0x11,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0x9f,0xaf,0x1
2,0x0,0xfff,

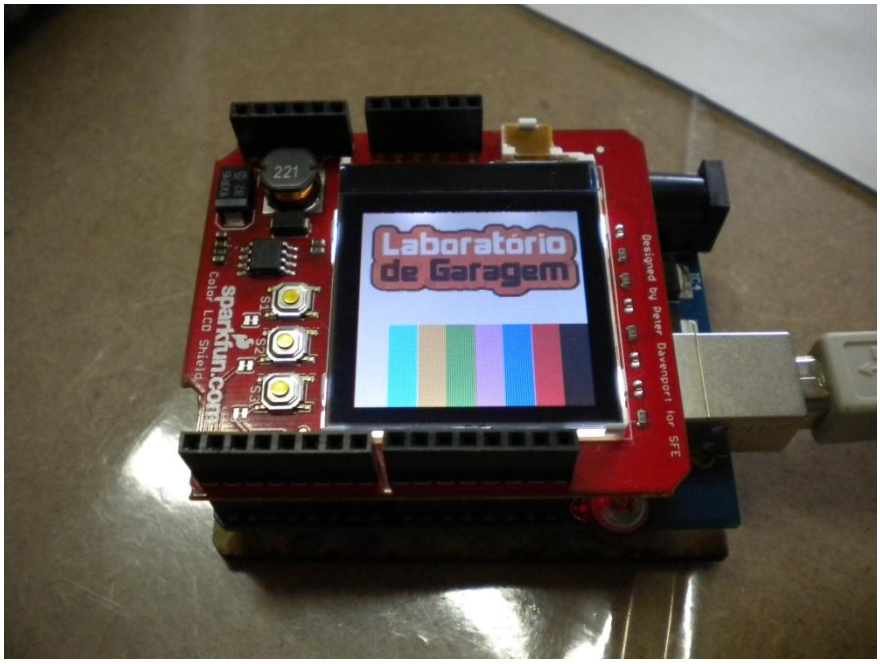
0xfff,0xccd,0x0,0x6a,0x8f,0x8f,0x8e,0x19f,0x13,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x9f,
0x14,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x7a,0x19f,0x9f,0x9f,0x9f,0x8e,0x8e,0x9f,0x1,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x
0,0x0,0x7b,0x35,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x12,0xaf,0x0,0x0,0x0,0x0,0x46,0x9f,0x9f,0x9f,0x9f,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x59,0x6b,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x47,0x6b,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x23,0x1af,
0x24,0x0,0x0,0x0,0x11,0x9f,0x9f,
0x5a,0x0,0x0,0x0,0x0,0x8d,0x9f,0x9f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0x9f,0xaf,0x12,0x0,0xfff,
0xfff,0xcdd,0x0,0x169,0x8f,0x18e,0x18e,0x8e,0x59,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x9e,0x48,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x7a,0x9f,0x19f,0x9f,0x9f,0x8e,0x8e,0x8e,0x37,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x7a,0x5a,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x36,0x9f,0x0,0x0,0x0,0x0,0x46,0x19f,0x9f,0x19f,0x9f,0x13,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x8d,0x9f,0x1,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x46,0x19f,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x23,0x1af,
0x24,0x0,0x0,0x0,0x11,0x1af,0x9f,
0x5a,0x0,0x0,0x0,0x0,0x18d,0x9f,0x9f,0x0,0x0,0x0,0x0,0x57,0x9f,0x9f,0x9f,0xaf,0x12,0x0,0xfff,
0xfff,0xfff,0x0,0x36,0x19f,0x8e,0x18e,0x8e,0x9f,0x6b,0x25,0x24,0x24,0x24,0x24,0x24,0x24,0x24,0x
24,0x24,0x19e,0x9f,0x59,0x24,
0x24,0x24,0x24,0x24,0x24,0x23,0x17b,0x9f,0x19f,0x9f,0x9f,0x8e,0x18e,0x8d,0x9f,0x16a,0x24,
0x24,0x24,0x24,0x24,0x24,0x24,0x24,
0x24,0x23,0x7b,0x19f,0x6a,0x24,0x24,0x24,0x24,0x24,0x24,0x58,0x19f,0x9f,0x24,0x24,0x24,0
x58,0x9f,0x9f,0x9f,0x9f,0x18e,
0x37,0x24,0x24,0x24,0x24,0x24,0x34,0x7c,0x9f,0x9f,0x19f,0x59,0x47,0x47,0x47,0x47,0x47,0x
0,0x0,0x0,0x0,0x47,0x9f,0x18d,0x35,
0x24,0x24,0x24,0x24,0x23,0x24,0x146,0x1af,0x47,0x24,0x24,0x35,0x19f,0x9f,0x6b,0x23,0x24,
0x24,0x9e,0x9f,0x19f,0x14,0x24,
0x24,0x69,0x9f,0x9f,0x9f,0x9f,0x12,0x0,0xfff,
0xfff,0xfff,0x333,0x101,0x19f,0x19e,0x8f,0x8e,0x8e,0x18f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,
0x9f,0x19f,0x19f,0x19f,0x19f,
0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x9f,0x9f,0x19f,0x8e,0x8e,0x18e,0x8d,0x8e,0x
9f,0x9f,0x9f,0x9f,0x9f,0x19f,
0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x18
f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x19f,0x9f,0x9f,0x19f,0x17d,0x58,0x59,0x59,0x59,0x59,0x158,
0x58,0x0,0x0,0x0,0x47,0x9f,
0x9f,0x9f,0x19f,0x19f,0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x8f,0x19f,0x9f,0x19
f,0x9f,0x19f,0x19f,0x19f,
0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0xaf,0x12,0x222,0xfff,
0xfff,0xfff,0xaab,0x0,0x158,0x18f,0x8e,0x18f,0x8e,0x8e,0x8e,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x9f,0x8f,0x19f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8e,0x8f,0x8e,0x8e,0x8e,0x8d,0x9f,0x9f
,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x8f,0x9f,0x
9f,0x9f,0x19f,0x8e,0x18e,
0x19f,0x9f,0x19f,0x19f,0x9f,0x9f,0x19f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x
57,0x19f,0x19f,0x9f,0x9f,0x19f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x
9f,0x9f,0x9f,0x19f,0x9f,
0x9f,0x19f,0x9f,0x9f,0x0,0x444,0xfff,
0xfff,0xfff,0xfff,0x223,0x0,0x7c,0x19f,0x9e,0x18e,0x8e,0x18e,0x9f,0x19f,0x19f,0x19f,0x19f,0x9f
,0x9f,0x9f,0x9f,0x19f,0x9f,
0x19f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x35,0x8e,0x18f,0x18f,0x18e,0x18e,0
x8e,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x18
f,0x9f,0x9f,0x19f,0x9f,0x18e,
0x8e,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x5a,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0
x7c,0x9f,0x9f,0x9f,0x9f,0x9f,

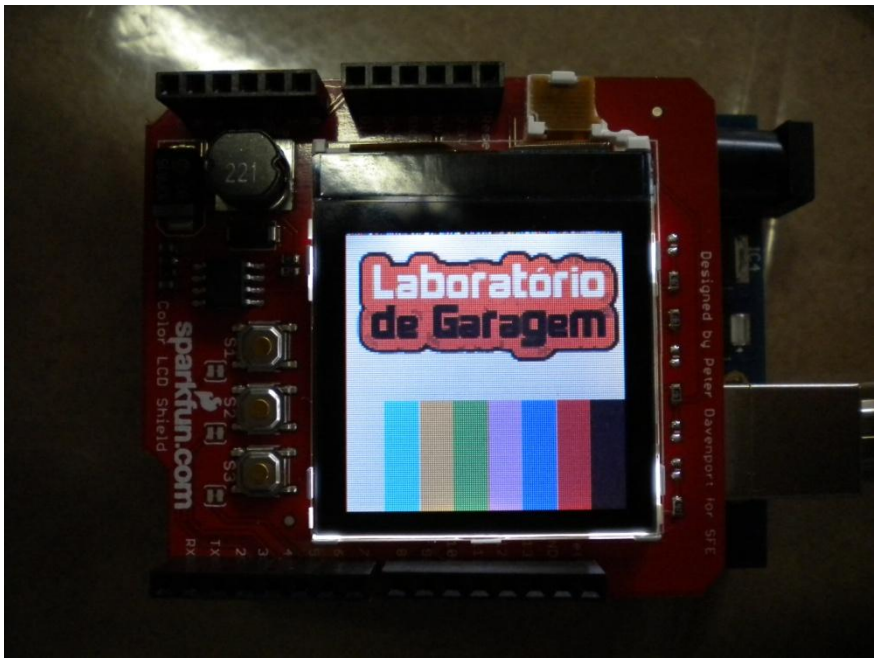
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9f,0x9f,0x19f,0x9f,0x19f,0x
19f,0x19f,0x19f,0x9f,0x9f,
0x9f,0x9f,0x19f,0x58,0x0,0xaaa,0xfff,
0xfff,0xfff,0xfff,0xeee,0x0,0x0,0x6a,0x9f,0x18f,0x18f,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x9f,0x9f,0x9f,0x8f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0xaf,0x59,0x0,0x0,0x17b,0x9f,0x18f,0x18f,0x8e,0x9f,0
x9f,0x9f,0x9f,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x8f,0x8e,0x9f,0x9f,0x9f,0x9f,0x8e,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x8f,0x9f,0x9f,0
x19f,0x9f,0x8e,0x8e,0x19f,
0x9f,0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x59,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x57,0x9f,0x1
9f,0x9f,0x9f,0x9f,0x19f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x9f,0x
9f,0x19f,0x19f,0x19f,
0x19f,0x7c,0x0,0x222,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0x666,0x0,0x13,0x58,0x18d,0x8f,0x19f,0x19f,0x19f,0x19f,0x19f,0x19f,
0x19f,0x9f,0x8f,0x6b,
0x18d,0x8e,0x9f,0x19f,0x19f,0x19f,0x19f,0x9f,0x19f,0x7b,0x24,0x0,0x333,0x666,0x0,0x23,0x1
69,0x8d,0x8e,0x19f,0x19f,0x19f,
0x19f,0x19f,0x19f,0x19f,0x19f,0x9e,0x7c,0x7d,0x8e,0x19f,0x19f,0x19f,0x19f,0x8e,0x18d,0x6b,
0x19e,0x19f,0x9f,0x9f,0x18f,
0x8e,0x47,0x0,0x35,0x6b,0x18e,0x9f,0x19f,0x9f,0x19f,0x9f,0x9f,0x9f,0x8f,0x8e,0x18d,0x8d,0x
8d,0x8d,0x8d,0x8d,0x8d,0x8d,
0x19f,0x9f,0x8e,0x8f,0x19f,0x9f,0x9f,0x19f,0x9f,0x19f,0x19f,0x19f,0x9f,0x8f,0x19f,0x19f,0x19f,
0x9f,0x18d,0x7d,0x9f,0x9f,
0x19f,0x9f,0x8f,0x7c,0x19e,0x19f,0x9f,0x9f,0x9e,0x8c,0x35,0x0,0x211,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xddd,0x555,0x111,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x
0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x999,0xfff,0xfff,0xccc,0x343,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0
x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x444,0x111,0x0,0x0,0x0,0x0,0x0,0
x1,0x1af,0x19f,0x19f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x19f,0x9f,0x9e,0x8e,0x18e,0x8e,0x18e,0x8e,0x19f,0x24,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,
0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x666,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xddd,0xeee,0xaba,0xaaa,0xaaa,0xaaa,0xaaa,0xa
aa,0xaaa,0xbbb,0xbbb,
0xddd,0xccc,0xded,0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xcbb,0xbbb,0xfff,0xfff,0xfff,0xfff,0
xfff,0xfff,0xfef,0xccd,0xded,
0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xaaa,0xbcb,0xccc,0xccc,0xeee,0xaaa,0xaaa,0
xaaa,0xaaa,0xeed,0xbcc,0xccc,0xbbb,
0xccc,0xaaa,0xaaa,0xccc,0xaaa,0xfff,0xfff,0xfff,0xeef,0xddd,0xccc,0xaaa,0x766,0x0,0x18d,0x9
f,0x9f,0x9f,0x19f,0x9f,0x19f,
0x9f,0x9f,0x9f,0x9f,0x9f,0x18e,0x8e,0x8e,0x8e,0x8f,0x8f,0x6a,0x0,0x767,0xaaa,0xaaa,0xaaa,0
xaaa,0xaab,0xaaa,0xaaa,0xaaa,
0xaaa,0xaaa,0xabb,0xaab,0xaaa,0xaaa,0xaaa,0xbab,0xbbb,0xabb,0xaaa,0xaa9,0xaaa,0xaa9,
0xaaa,0xbbb,0xfee,0xfff,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x0,0x13,0xaf,0x9f,0x9f,0x9f,0x
19f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x19f,0x18e,0x8e,0x8f,0x19e,0x9f,0x17b,0x0,0x544,0xfff,0xfff,0xfff,0xfff,0xff
f,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,

```
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xaaa,0x0,0x24,0x19f,0x9f,0x9f
,0x9f,0x9f,0x9f,
0x9f,0x9f,0x9f,0x9f,0x8f,0x19f,0x19f,0x8e,0x35,0x0,0x888,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x
fff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xaaa,0x0,0x0,0x24,0x46,
0x36,0x36,0x36,
0x36,0x36,0x36,0x36,0x146,0x35,0x12,0x0,0x322,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xff
f,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0x888,0x444,0x2
22,0x111,
0x111,0x111,0x111,0x111,0x111,0x111,0x332,0x555,0x888,0xddd,0xfff,0xfff,0xfff,0xfff,0xfff,0xff
f,0xfff,0xfff,0xfff,0xfff,
0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xfff,0xf
ff,
0xfff,0xfff,0xfff,0xfff,0xfff,
};
void printldg(void) //Função para ordenar os pixels da matriz logoldg e montar a imagem
{
int x=0; int y=0; unsigned int z=0;
for(y=10;y<62;y=y+1) //Localização de onde deve começar a imagem e onde termina.
Começando no ponto y=10 do LCD
{ // e terminando no ponto y=62(52+10), onde 52 é a altura da imagem e 10 o começo do
primeiro ponto do LCD
for(x=5;x<130;x=x+1) //Localização do ponto x=5 do LCD terminando no ponto x=130(125+5),
onde 125 é o comprimento da imagem
{ //e 5 é o começo do LCD
lcd.setPixel(pgm_read_word_near(logoldg+z),y,x); //Coloca o pixel da matriz logoldg no ponto
[x,y] do LCD
z=z+1; //Vai para o próximo elemento da matriz logoldg
}
}
}
```

Por fim, conecte o Shield no Arduino e depois o Arduino no PC. Selecione a versão da sua placa Arduino (UNO, Duemilanove, etc) e selecione a porta a qual seu Arduino está conectado (COMx, ttyUSBx, ttyACMx, etc). Clique em UPLOAD e assim que terminar irá aparecer a

imagem. Aqui fizemos com o logo e mais barras coloridas para demonstrar. Abaixo tem umas imagens mostrando o LCD em funcionamento.





Tivemos que colar a imagem em hexadecimal dentro da memória Flash, pois senão não funciona por causa da memória RAM limitada.

E é isso! Esperamos que tenham gostado! Caso tenham dúvidas, postem aqui mesmo no blog! Caso tenham sugestões de tutoriais, postem aqui! E se quiserem ver outros tutoriais e projetos abertos, cliquem aqui eaqui respectivamente! Até a próxima galera!

Referências:

http://www.labdegaragem.org/loja/index.php/31-shields/lcd-shield-co...

http://www.labdegaragem.com.br/wiki/index.php?title=Color_Shield_LCD

http://www.sparkfun.com/products/9363

http://www.sparkfun.com/tutorials/286

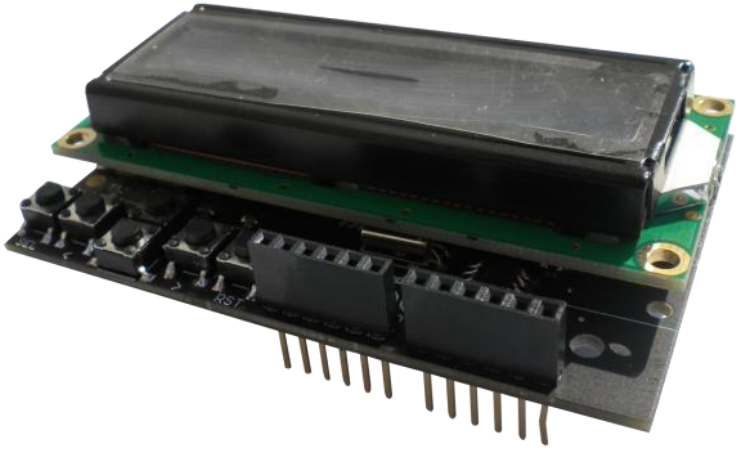
https://github.com/downloads/jimblom/ColorLCDShield/ColorLCDShield-...

http://tronixstuff.wordpress.com/2011/02/07/tutorial-arduino-and-co...

http://peterdavenport.posterous.com/pages/prodjects

http://www.sparkfun.com/tutorial/Nokia%206100%20LCD%20Display%20Dri...

84. Tutorial: como utilizar o Data Shield com Arduino



Neste tutorial, vamos mostrar como utilizar o Data Shield com Arduino. O Data Shield consiste em um display de LCD, um suporte para cartão MicroSD, um suporte para bateria do "Real Time Clock" e botões.

O Data Shield pode ser usado como central de monitoramento de horários, de pessoas e de entrada e saída (data logger). Pode ser usado também para armazenamento, organização de dados e mostrar os dados no display LCD.

Incubadora Lab de Garagem

Primeiramente coloque a bateria de 3V que vem junto com o Data Shield, depois coloque um cartão micro-SD (sem o cartão o Data Shield não funciona). Agora conecte o Data Shield no Arduino.

Agora, baixe as bibliotecas disponíveis na Loja LdG: Se caso estiver utilizando a versão 002x da IDE do Arduino, baixe deste link. Se estiver utilizando a versão 1.0 da IDE do Arduino, baixe deste link.

Baixe o código exemplo disponível na Loja LdG, clicando aqui.

OBS: Para a versão 1.0 da IDE comente a linha em vermelho.

```
/*
Exemplo de programa escrito para o DATA SHILED do Laboratório de garagem http://labdegar
agem.com
Versão 1.0 - 01 Mar 2012
Versão 1.1 - 08 Mar 2012 -
 correção do bug do fadeout do Display, assim como no tempo do fadeout.
Escrito por Jiri Trnka
Compilar com Arduino IDE 0019 a 0023
Código sob licença de código aberto "open source" GPL.
Cópia autorizada desde que mantida a referência aos autores
OBS: Para utilizar o LCD adicione DS1307RTC.h e LiquidCrystal_I2c.h
*/
// ***************** INCLUDES of the necessary libraries *************
#include <LiquidCrystal_I2C.h>
#include <EEPROM.h>
//#include <WProgram.h>
#include <avr/pgmspace.h>
#include <Wire.h>
#include <Time.h>
#include <TimeAlarms.h>
#include <DS1307RTC.h> //Necessary to LCD
#include <SD.h>
// ***************** CONSTANTS *****************
const int ERROR_WINDOW = 50; // tolerance +/- this value for the button's analog input
const int BUTTONDELAY = 50; // button's debounce delay
const int BUTTONREPEAT = 200; // repeat delay of the pressed buttons
const int TIMEOUT = 30000; // timout value to exit to default display in miliseconds (remember
that INT type max allowed value is 32767)
const int LEDPIN = 5; // output pin for the PWM of the LCD backlight
const int CONTRASTPIN = 9; // output pin for the PWM of the LCD contrast
const int SD_CS = 8; // Pin 8 is the CS chip select of the SD memory card on this Shield
const int BUTTONPIN = A0; // input pin for the buttons
// ************ BUTTONS position and analog values ************************
// [UP (2)] 460
// [SEL (5)] 0 [LFT (1)] 133 [RGT (4)] 715 [RST]
// [DWN (3)] 273
const int SELBTN = 0; // Analog value for the SELECT button
const int LFTBTN = 133; // Analog value for the LEFT button
const int DWNBTN = 273; // Analog value for the DOWN button
const int UPBTN = 460; // Analog value for the UP button
const int RGTBTN = 715; // Analog value for the RIGHT button
// Uncomment the following line to send DEBUG messages to the serial port at 9600 bauds
// #define DEBUG_ON
// Flash memory stored string table. Necessary because otherwise the strings will take up too m
uch of our very limited RAM.
// Use either PrintLCDAt_P or PrintLCD_P to print these strings to the LCD.
prog_char string_0[] PROGMEM = "1. Contraste "; //0
prog_char string_1[] PROGMEM = "2. Backlight "; //1
prog_char string_2[] PROGMEM = "3. Data "; //2
prog_char string_3[] PROGMEM = "4. Hora "; //3
```

```
prog_char string_4[] PROGMEM = " Menu Principal "; //4
prog_char string_5[] PROGMEM = " Contraste "; //5
prog_char string_6[] PROGMEM = " Backlight "; //6
prog_char string_7[] PROGMEM = " Data "; //7
prog_char string_8[] PROGMEM = " Hora "; //8
//array of pointers to the flash based string table
PROGMEM const char *string_table[] = {
string_0, string_1, string_2, string_3, string_4, string_5, string_6, string_7, string_8
};
// ********** GLOBAL VARIABLES **********
// LCD
LiquidCrystal_I2C lcd(0x20,16,2); // set the i2c LCD address to 0x20 and define a 16 rows and 2 lines display
byte backlight; // values from 0 to 255 LCD backlight
byte contrast; // values from 0 to 255 LCD contraste
long previousMillis; // will store last time of timout
unsigned long currentMillis; // will store the current time counter
boolean BacklightON = true; // used for the control of the backlight timout
boolean SDCARD = true; // used for disabling the logging in case of SD CARD failure
File myFile;
char DATAFILE[] = "data.log"; // Name of the Data file that will be created on the SD memomry card, this name must be in the 8.3 DOS format
// Buttons
unsigned long buttonLastChecked = 0; // variable to limit the button getting checked every cycle
short previousval = 0; // initialization of the variable to 0
short pushedbutton = 0; // initialization of the variable to 0
short previousbutton = 0; // initialization of the variable to 0
unsigned long lastbtnpress = 0; // initialization of the variable to 0
// ********** ARDUINO SETUP **********
void setup()
{
pinMode(LEDPIN, OUTPUT); // sets the LCD LED backlight pin as output
pinMode(CONTRASTPIN, OUTPUT); // sets the LCD Contrast pin as output
pinMode(A1, INPUT); // sets the Analog pin 1 as input
pinMode(A2, INPUT); // sets the Analog pin 2 as input
pinMode(A3, INPUT); // sets the Analog pin 3 as input

// Up to 6 timers can be defined, see documentation of the TimeAlarms Library for details
// Alarm.timerRepeat(15, RepeatTask); // timer that calls the "RepeatTask function every 15 seconds
Alarm.timerRepeat(1,0,0, RepeatTask); // timer that calls the "RepeatTask function every hour (up to 6 different timers can be set)

lcd.begin(16, 2); // set up the LCD's number of columns and rows:
lcd.print("DATA SHIELD v1.1"); // Print a message to the LCD.
lcd.setCursor(0, 1); // set the cursor to column 0, line 1 (note: line 1 is the second line, since counting starts with 0):
lcd.print("labdegaragem.com"); // Print a message to the LCD.
backlight=EEPROM.read(1); // Read from the EEPROM the backlight value set by the interface. It is stored in the EEPROM address 1
analogWrite(LEDPIN, backlight); // Turn On the LCD Backlight to the value stored in the EEPROM
contrast=EEPROM.read(0); // Read from the EEPROM the contrast value set by the interface. It is stored in the EEPROM address 0
if (contrast > 81) contrast=0; // A formatted or new chip has all it's EEPROM bytes set to 255, and contrast at 255 is the minimum contrast, so in order to have something on the display we test new chips here.
analogWrite(CONTRASTPIN, contrast); // Set the LCD contrast to the value stored in the EEPROM
#ifdef DEBUG_ON
```

```
Serial.begin(9600); // Serial port initialize
Serial.println("Data logger BOOT");
#endif
// On the labdegaragem DATA SHILED, CS is pin 8. It's set as an output by default.
// Note that even if it's not used as the CS pin, the hardware SS pin
// (10 on most Arduino boards, 53 on the Mega) must be left as an output
// or the SD library functions will not work.
pinMode(10, OUTPUT);
if (!SD.begin(SD_CS)) {
lcd.clear();
lcd.print("Cartao SD falhou");
lcd.setCursor(0, 1); // set the cursor to column 0, line 1 (note: line 1 is the second line, since counting starts with 0):
lcd.print(" Log desativado "); // Print a message to the LCD.
SDCARD = false;
#ifdef DEBUG_ON
Serial.println("INIT SD Falhou!");
#endif
Alarm.delay(2000); // For the library TimeAlarms, it is necessary to use the Alarm.delay function instead of the standard Delay function.
}
setSyncProvider(RTC.get); // Sync the Arduino clock from the time of the RTC, resync every 5 minutes (default value)
Alarm.delay(5000); // Wait for splash screen
previousMillis = millis(); // Reset timout counter
if (SDCARD) {
myFile = SD.open(DATAFILE, FILE_WRITE); // if SD card present, open the data.log file, note that only one file can be open at a time, so you have to close this one before opening another.
digitalClockWrite();
myFile.println(";Boot");
myFile.close(); // close the file
}
lcd.clear(); // pretty self explaining, isn´t it?
}
// ********** ARDUINO MAIN LOOP **********
void loop()
{
digitalClockDisplay(); //call function
Alarm.delay(50); // refresh of the alarm trigger, with i2c displays do not choose values under 50 ms in order to avoid ugly flickering.
if( buttonLastChecked == 0 ) // see if this is the first time checking the buttons
buttonLastChecked = millis()+BUTTONDELAY; // force a check this cycle because we do not need to check every cycle
if( millis() - buttonLastChecked > BUTTONDELAY ) { // make sure a reasonable delay passed
if( int buttNum = buttonPushed(BUTTONPIN) ) {
previousbutton=buttNum;
mainMenu();
}
buttonLastChecked = millis(); // reset the lastChecked value
}
// check to see if it's time to timeout the backlight; that is, if the
// difference between the current time and last time of some button event
// is bigger than the interval.
currentMillis = millis();
if(currentMillis - previousMillis > TIMEOUT) {
if (BacklightON != false) {
fadeBacklight();
}
}
}
```

```
// ******************* FUNCTIONS ****************************
// ************* Read Buttons *************
int buttonPushed(int BUTTONPIN) {
int val = ERROR_WINDOW; // variable to store the read value
val = analogRead(BUTTONPIN); // read the input pin
#ifdef DEBUG_ON
Serial.println(val); // Output to serial port the analog value of the pressed button
#endif
if (val < (previousval-ERROR_WINDOW) or val > (previousval+ERROR_WINDOW)) { // Avoid
duplicate button presses
// we don't use the upper 1023 position because that is the same as the
// all-open switch value when the internal 20K ohm pullup is enabled.
if( val >= (RGTBTN-ERROR_WINDOW) and val <= (RGTBTN+ERROR_WINDOW) ) { // 716
previousval = val;
pushedbutton = 4; // RIGHT
previousMillis = currentMillis; // reset of timout
BacklightON = true;
lastbtnpress = millis(); // reset the lastChecked value
return pushedbutton;
}
else if ( val >= (UPBTN-ERROR_WINDOW) and val <= (UPBTN+ERROR_WINDOW) ) { // 469
previousval = val;
pushedbutton = 2; // UP
previousMillis = currentMillis; // reset of timout
BacklightON = true;
lastbtnpress = millis(); // reset the lastChecked value
return pushedbutton;
}
else if ( val >= (DWNBTN-ERROR_WINDOW) and val <= (DWNBTN+ERROR_WINDOW) ) { //
293
previousMillis = currentMillis; // reset of timout
previousval = val;
pushedbutton = 3; // DOWN
BacklightON = true;
lastbtnpress = millis(); // reset the lastChecked value
return pushedbutton;
}
else if ( val >= (LFTBTN-ERROR_WINDOW) and val <= (LFTBTN+ERROR_WINDOW) ) { //
132
previousMillis = currentMillis; // reset of timout
previousval = val;
pushedbutton = 1; // LEFT
BacklightON = true;
lastbtnpress = millis(); // reset the lastChecked value
return pushedbutton;
}
else if( val >= SELBTN and val <= (SELBTN+ERROR_WINDOW) ) {
previousMillis = currentMillis; // reset of timout
previousval = val;
pushedbutton = 5; // SELECT
BacklightON = true;
lastbtnpress = millis() ; // reset the lastChecked value
return pushedbutton;
}
else {
previousval = (-ERROR_WINDOW-1);
return 0; // no button found to have been pushed
}
}
```

```
else if (pushedbutton < 5) {
if( millis() - lastbtnpress > BUTTONREPEAT ) { // repeat key presses at BUTTONREPEAT
interval when buttons are keep pressed
lastbtnpress = millis(); // reset the time of the last key press
previousMillis = currentMillis; // reset of timout
return pushedbutton;
}
}
}
// ************* Main Menu *************
void mainMenu() {
byte offset=0;
int buttNum;
analogWrite(LEDPIN, backlight);
lcd.clear();
lcd.noBlink();
PrintLCD_P(4); //" Main Menu "
do {
PrintLCDAt_P(offset, 0, 1);
buttNum = buttonPushed(BUTTONPIN);
previousbutton = buttNum;
Alarm.delay(BUTTONDELAY);
currentMillis = millis();
switch(buttNum) {
case 1:
loop();
break;
case 2:
if (offset > 0) offset--;
break;
case 3:
if (offset < 3) offset++;
break;
case 5:
switch (offset) {
case 0: //Contrast
ContrastMenu();
break;
case 1: //Backlight
BacklightMenu();
break;
case 2: //Date
DateMenu();
break;
case 3: //Hour
HourMenu();
break;
}
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
}
// ************* Contrast Menu *************
void ContrastMenu() {
int bar=(contrast/16);
byte prevcontr = contrast;
int prevBar = bar;
lcd.clear();
PrintLCD_P(5); //" Contrast "
```

```
do {
bar=(15-(contrast/5));
lcd.setCursor(prevBar,1);
lcd.print(" ");
lcd.setCursor(bar, 1);
lcd.print(char(255));
analogWrite(CONTRASTPIN, contrast);
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
prevBar = bar;
currentMillis = millis();
switch(buttNum) {
case 1: // esc
contrast = prevcontr;
analogWrite(CONTRASTPIN, contrast);
mainMenu();
break;
case 2:
if (contrast >=5) contrast=contrast-5; // 0 = contrast MAX
break;
case 3:
if (contrast < 81) contrast=contrast+5; // does not make sense to go upper than aprox. 90, the
contrast is already at MIN arround this value
break;
case 5: // save
EEPROM.write(0, contrast); // write the new contrast value to EEPROM at address 0
mainMenu();
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
}
// ************* Backlight Menu *************
void BacklightMenu() {
int bar = (backlight/16);
byte prevback = backlight;
int prevBar = bar;
lcd.clear();
PrintLCD_P(6); //" Backlight "
do {
bar=(backlight/16);
lcd.setCursor(prevBar,1);
lcd.print(" ");
lcd.setCursor(bar, 1);
lcd.print(char(255));
analogWrite(LEDPIN, backlight);
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
prevBar = bar;
currentMillis = millis();
switch(buttNum) {
case 1: // ESC
backlight = prevback;
analogWrite(LEDPIN, backlight);
mainMenu();
break;
case 2:
if (backlight < 240) backlight=backlight+16; // 255 = backlight MAX
break;
case 3:
```

```
if (backlight >=16) backlight=backlight-16; // 0 = backlight OFF
break;
case 5: // SAVE
EEPROM.write(1, backlight); // write the new backlight value to EEPROM at address 1
mainMenu();
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
}
// ************* Date Menu *************
void DateMenu() {
lcd.clear();
PrintLCD_P(7); //" Date "
lcd.setCursor(1, 1); //line=1 (0 is the first line), x=3
printDayOfWeek(weekday());
lcd.print(" ");
lcd.print(zerolead(day()));
lcd.print("/");
lcd.print(zerolead(month()));
lcd.print("/");
lcd.print(year());
do {
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
currentMillis = millis();
switch(buttNum) {
case 1: // ESC
mainMenu();
break;
case 5: // Call setup menu
DateSetMenu(); // Back to menu
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
}
// ************* Date Set Menu *************
void DateSetMenu() {
byte offset=0;
int Day,Month,Year;
Day = day();
Month = month();
Year = year();
lcd.clear();
PrintLCD_P(7); //" Date "
lcd.setCursor(3, 1); //line=1 (0 is the first line), x=3
lcd.print(zerolead(day()));
lcd.print("/");
lcd.print(zerolead(month()));
lcd.print("/");
lcd.print(year());
lcd.blink();
do {
lcd.setCursor(4+(offset*3), 1);
if (offset == 2) lcd.setCursor(12, 1);
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
currentMillis = millis();
switch(buttNum) {
```

```
case 1:
if (offset > 0) offset--;
break;
case 2:
switch(offset) {
case 0:
if (Day < 31) Day++;
lcd.setCursor(3, 1); //line=2 (1 is the second line), x=5
lcd.print(zerolead(Day));
break;
case 1:
if (Month < 12) Month++;
lcd.setCursor(6, 1); //line=2 (1 is the second line), x=8
lcd.print(zerolead(Month));
break;
case 2:
if (Year < 2099) Year++;
lcd.setCursor(9, 1); //line=2 (1 is the second line), x=11
lcd.print(Year);
break;
}
break;
case 3:
switch(offset) {
case 0:
if (Day > 1) Day--;
lcd.setCursor(3, 1); //line=2 (1 is the second line), x=5
lcd.print(zerolead(Day));
break;
case 1:
if (Month > 1) Month--;
lcd.setCursor(6, 1); //line=2 (1 is the second line), x=8
lcd.print(zerolead(Month));
break;
case 2:
if (Year > 2010) Year--;
lcd.setCursor(9, 1); //line=2 (1 is the second line), x=11
lcd.print(Year);
break;
}
break;
case 4:
if (offset < 2) offset++;
break;
case 5: // SAVE
setTime(hour(),minute(),second(),Day,Month,Year); // set Arduino system time
time_t t = now(); // convert to UNIX type time_t
RTC.set(t); // set the time of the RTC chip
lcd.noBlink();
DateMenu(); // Back to menu
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
lcd.noBlink();
}
// ************* Hour Menu *************
void HourMenu() {
lcd.clear();
PrintLCD_P(8); //" Hour "
```

```
do {
lcd.setCursor(4, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(hour()));
lcd.print(":");
lcd.print(zerolead(minute()));
lcd.print(":");
lcd.print(zerolead(second()));
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
currentMillis = millis();
switch(buttNum) {
case 1: // ESC
mainMenu();
break;
case 5: // call setup menu
HourSetMenu(); // Back to menu
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
}
// ************* Hour Set Menu *************
void HourSetMenu() {
byte offset=0;
int Hour,Minute,Second;
Hour = hour();
Minute = minute();
Second = 0;
lcd.clear();
PrintLCD_P(8); //" Hour "
lcd.setCursor(4, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(Hour));
lcd.print(":");
lcd.print(zerolead(Minute));
lcd.print(":");
lcd.print(zerolead(Second));
lcd.blink();
do {
lcd.setCursor(5+(offset*3), 1);
int buttNum = buttonPushed(BUTTONPIN);
Alarm.delay(BUTTONDELAY);
currentMillis = millis();
switch(buttNum) {
case 1:
if (offset > 0) offset--;
break;
case 2:
if (offset==0) {
if (Hour < 23) Hour++;
lcd.setCursor(4, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(Hour));
}
else
{
if (Minute < 59) Minute++;
lcd.setCursor(7, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(Minute));
}
break;
case 3:
```

```
if (offset==0)
{
if (Hour > 0) Hour--;
lcd.setCursor(4, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(Hour));
}
else
{
if (Minute > 0) Minute--;
lcd.setCursor(7, 1); //line=2 (1 is the second line), x=4
lcd.print(zerolead(Minute));
}
break;
case 4:
if (offset < 1) offset++;
break;
case 5: // SAVE
setTime(Hour,Minute,Second,day(),month(),year()); // Set arduino system time
time_t t = now(); // convert to UNIX type time_t
RTC.set(t); // set the time of the RTC chip
lcd.noBlink();
HourMenu();
break;
}
}
while (currentMillis - previousMillis < TIMEOUT);
lcd.noBlink();
}
// ************* Print PROGMEM string *************
// This is the gatekeeper function. The only one to really touch the strings in program memory. T
his keeps the need
// for a buffer string to only one. So, basically we're only using 17 bytes of RAM to store strings.
void PrintLCD_P(int which) {
char buffer[17];
strcpy_P(buffer, (char*)pgm_read_word(&(string_table[which])));
lcd.print(buffer);
}
// ************* Print PROGMEM string at LCD location *************
// Print a string found in program memory to the LCD at a certain location
// Basically this just sets the location and then calls PrintLCD_P to do the work
void PrintLCDAt_P(int which, char x, char y) {
lcd.setCursor(x, y);
PrintLCD_P(which);
}
// ************* Display date / time on LCD display *************
void digitalClockDisplay()
{
// digital clock display of the date and time
lcd.setCursor(1, 0); //line=1 is the LCD´s second line (0 is the first line)
printDayOfWeek(weekday());
lcd.print(" ");
lcd.print(zerolead(day()));
lcd.print("/");
lcd.print(zerolead(month()));
lcd.print("/");
lcd.print(year());
lcd.setCursor(4, 1); //line=2 (1 is the second line), x=3
lcd.print(zerolead(hour()));
lcd.print(":");
lcd.print(zerolead(minute()));
```

```
lcd.print(":");
lcd.print(zerolead(second()));
}
// ************* Write date / time into a file on the SD card *************
void digitalClockWrite()
{
// digital clock display of the date and time
myFile.print(zerolead(day()));
myFile.print("/");
myFile.print(zerolead(month()));
myFile.print("/");
myFile.print(year());
myFile.print(";");
myFile.print(zerolead(hour()));
myFile.print(":");
myFile.print(zerolead(minute()));
myFile.print(":");
myFile.print(zerolead(second()));
}
// ************* The day name corresponding to the day number given by the RTC *************
void printDayOfWeek(int dayOfWeek)
{
switch(dayOfWeek){
case 1:
lcd.print("Dom");
break;
case 2:
lcd.print("Seg");
break;
case 3:
lcd.print("Ter");
break;
case 4:
lcd.print("Qua");
break;
case 5:
lcd.print("Qui");
break;
case 6:
lcd.print("Sex");
break;
case 7:
lcd.print("Sab");
break;
}
}
// ************* Utility function for digital clock/date display: prints leading 0 for single digit number
s *************
String zerolead(int digits)
{
String lead;
if(digits < 10) {
lead ='0'+String(digits);
}
else {
lead = String(digits);
}
return lead;
}
// ************* Tasks to be executed at the interval defined in the Timer ****************************
```

```
void RepeatTask (){
if (SDCARD) {
myFile = SD.open(DATAFILE, FILE_WRITE); // if SD card present, open the data.log file, note that only one file can be open at a time, so you have to close this one before opening another.
digitalClockWrite();
myFile.print(";");
myFile.print(analogRead(A1));
myFile.print(";");
myFile.print(analogRead(A2));
myFile.print(";");
myFile.println(analogRead(A3));
myFile.close(); // close the file
}
}
// ************* Fade backlight after inactivity timout and reset to date/time default display *********
****
void fadeBacklight()
{
BacklightON = false;
// fade out from max to min in increments of 1 point:
lcd.clear();
for(int fadeValue = backlight; fadeValue >= 0; fadeValue -=1) {
// Dims the backlight from the current value to 0 (OFF)
analogWrite(LEDPIN, fadeValue);
Alarm.delay(10); // wait for 10 milliseconds to see the dimming effect.

if (millis()-currentMillis > 50) { // To avoid nasty flickering of the display, do not refresh it on every cycle, keep at least 50 mS between each display refresh
digitalClockDisplay();
currentMillis = millis();
}
// Check if some button was pressed during the dimming progress
if( int buttNum = buttonPushed(BUTTONPIN) ) {
analogWrite(LEDPIN, backlight);
currentMillis = millis(); // reset of timout
BacklightON = true;
mainMenu();
break;
}
}
}
```

A programação exemplo aplica todas as funções de uma vez só. O Data Shield mostrará a data e hora e armazenará estes dados no cartão Micro-SD periodicamente. Portanto se não tiver cartão Micro-SD no suporte do Shield, este não funcionará. Você pode também mudar a data, hora, o contraste e a iluminação do LCD pelos botões do Shield.

Por fim, conecte o Arduino no PC, selecione a versão da sua placa Arduino (UNO, Duemilanove, etc), depois selecione a porta (COMx, ttyUSBx, ttyACMx) e clique em UPLOAD.

Você verá o LCD acender e escrever "Data Shield v1.1 labdegaragem.com" e depois a data e a hora. Para configurar a data, clique qualquer botão para entrar no Menu Principal (exceto os

botões "sel" e "rst"). No Menu Principal, irá aparecer 1.Contraste, ao apertar o botão para baixo, você verá 2.Backlight, 3.Data e 4.Hora. Selecione 4.Data e clique em "sel" para configurá-lo. Ao terminar, clique no botão em que está escrito "<" abaixo dele e voltará para o menu principal. Agora é só repetir o processo para cada item do Menu Principal para mudar o que desejar.

E é isso! Esperamos que tenham gostado! Caso tenham dúvidas, postem aqui mesmo neste blog! Se tiverem sugestões para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respecitvamente!

Referências:

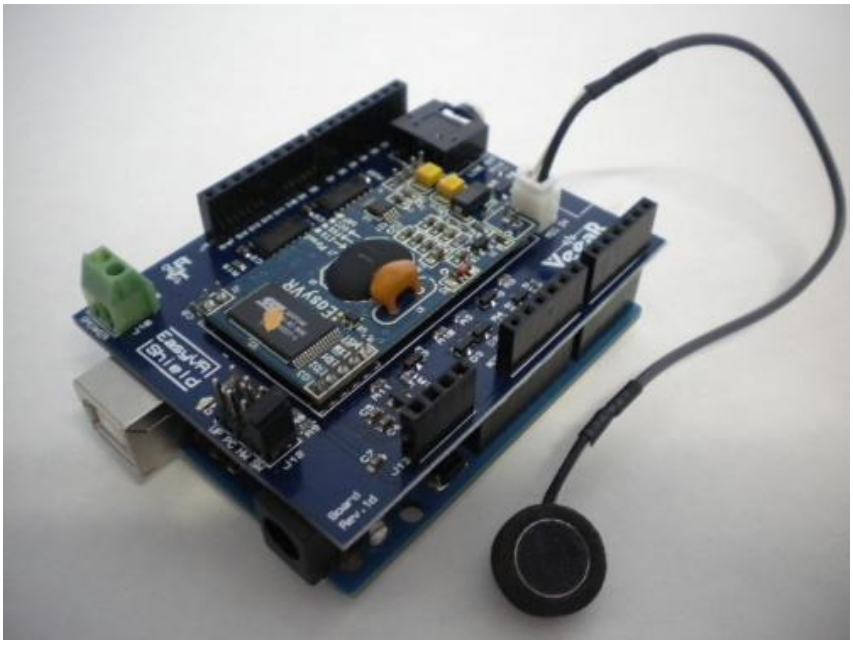
http://www.labdegaragem.org/loja/index.php/31-shields/datashield.html

Biblioteca para Arduino versão 002x: Libraries_002x.zip

Biblioteca para Arduino 1.x:Libraries_1_0.zip

http://labdegaragem.com.br/loja/Example_DATASHIELD.pde

85. Tutorial: Reconhecimento de voz com Arduino



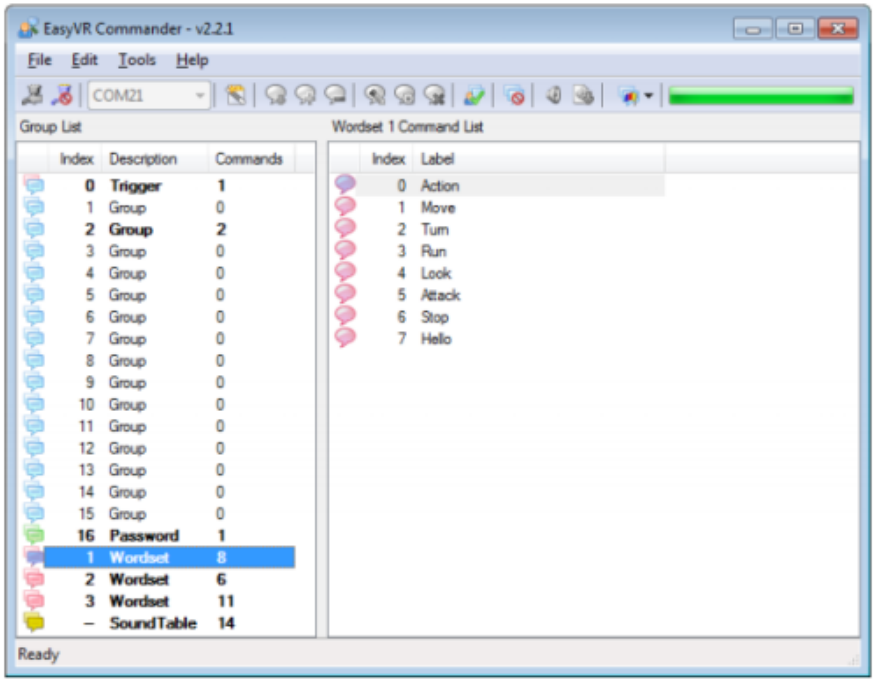
Neste tutorial, vamos mostrar como utilizar o EasyVR Shield com Arduino com um exemplo já disponível na biblioteca do EasyVR Shield. Com o EasyVR Shield você poderá conversar com seu Arduino, dando comandos por voz e programando o que ele fará ao receber os comandos. Este Shield é de fácil aplicação e bastante robusto.

Primeiramente, baixe e abra o arquivo EasyVR_User_Manual_3.3.pdf. Este arquivo é a documentação. Na documentação tem toda a explicação e utilização do Shield com Arduino, mas neste tutorial vamos mostrar de forma mais resumida.

Agora baixe o arquivo EasyVR Commander. Abra o arquivo e extraia em uma pasta de sua escolha. Agora vá para pasta e abra o arquivo .exe (este software é somente para windows) e instale o programa. Assim que acabar de instalar, abra-o. Com o programa aberto, pegue o Shield, conecte o microfone e depois conecte o Shield no Arduino. Certifique que o JUMPER está na posição PC como na figura abaixo:

Abra o programa EasyVR Commander, vá em File/Port e selecione a porta em que o Arduino está conectado. Agora clique no botão "Connect" localizado no canto superior esquerdo do programa.

O programa está demonstrado abaixo:



Na tabela esquerda, podemos ver vários grupos os quais você pode criar gravando a própria voz utilizando QuickSyntetizer e assim ter um reconhecimento de voz próprio. O Wordset são palavras prontas como exemplo e qualquer pessoa que disser a palavra, o Shield reconhecerá a palavra.

Agora vamos testar o EasyVR Shield. Selecione um Wordset, clique em "test the group", irá aparecer uma nova janela escrito "Speaking now". Diga uma das palavras dentro do Wordset que você escolheu no microfone. Se ele reconhecer ele vai piscar a palavra que você falou no programa. Se não irá aparecer uma nova janela dizendo que não foi reconhecido ou não foi claro o suficiente para o EasyVR entender.

Agora feche o EasyVR Commander e mude o JUMPER para a posição SW. Baixe a biblioteca aqui. Abra-o e se estiver utilizando a a versão 1.0 da IDE do Arduino, escolha a pasta "arduino1.0" e extraia em uma pasta de sua escolha. Copie a pasta "EasyVR" e cole na pasta "libraries" localizada dentro da pasta da IDE do Arduino. Por fim abra a IDE do Arduino.

Vá em File/Examples/EasyVR e selecione TestEasyVR. Ao abrir, mude a língua para ENGLISH, como está destacado em vermelho:

```
/**
EasyVR Tester

Dump contents of attached EasyVR module
and exercise it with playback and recognition.

Serial monitor can be used to send a few basic commands:
'c' - cycles through available command groups
'b' - cycles through built-in word sets
's123.' - play back sound 123 if available (or beep)

With EasyVR Shield, the green LED is ON while the module
is listening (using pin IO1 of EasyVR).
Successful recognition is acknowledged with a beep.
Details are displayed on the serial monitor window.
**
Example code for the EasyVR library v1.0
Written in 2011 by RoboTech srl for VeeaR <http:://www.veear.eu>
To the extent possible under law, the author(s) have dedicated all
copyright and related and neighboring rights to this software to the
public domain worldwide. This software is distributed without any warranty.
You should have received a copy of the CC0 Public Domain Dedication
along with this software.
If not, see a href="http://creativecommons.org/publicdomain/zero/1.0/%3E">http://creativecomm
ons.org/publicdomain/zero/1.0/>;.
*/
#if defined(ARDUINO) && ARDUINO >= 100
#include "Arduino.h"
#include "SoftwareSerial.h"
SoftwareSerial port(12,13);
#else // Arduino 0022 - use modified NewSoftSerial
#include "WProgram.h"
#include "NewSoftSerial.h"
NewSoftSerial port(12,13);
#endif
#include "EasyVR.h"
EasyVR easyvr(port);
```

```
int8_t set = 0;
int8_t group = 0;
uint32_t mask = 0;
uint8_t train = 0;
char name[32];
bool useCommands = true;
EasyVRBridge bridge;
void setup()
{
// bridge mode?
if (bridge.check())
{
cli();
bridge.loop(0, 1, 12, 13);
}
// run normally
Serial.begin(9600);
port.begin(9600);
if (!easyvr.detect())
{
Serial.println("EasyVR not detected!");
for (;;);
}
easyvr.setPinOutput(EasyVR::IO1, LOW);
Serial.println("EasyVR detected!");
easyvr.setTimeout(5);
easyvr.setLanguage(EasyVR::ENGLISH);

int16_t count = 0;

Serial.print("Sound table: ");
if (easyvr.dumpSoundTable(name, count))
{
Serial.println(name);
Serial.print("Sound entries: ");
Serial.println(count);
}
else
Serial.println("n/a");

if (easyvr.getGroupMask(mask))
{
uint32_t msk = mask;
for (group = 0; group <= EasyVR::PASSWORD; ++group, msk >>= 1)
{
if (!(msk & 1)) continue;
if (group == EasyVR::TRIGGER)
Serial.print("Trigger: ");
else if (group == EasyVR::PASSWORD)
Serial.print("Password: ");
else
{
Serial.print("Group ");
Serial.print(group);
Serial.print(": ");
}
count = easyvr.getCommandCount(group);
Serial.println(count);
for (int8_t idx = 0; idx < count; ++idx)
{
```

```
if (easyvr.dumpCommand(group, idx, name, train))
{
Serial.print(idx);
Serial.print(" = ");
Serial.print(name);
Serial.print(", Trained ");
Serial.print(train, DEC);
if (!easyvr.isConflict())
Serial.println(" times, OK");
else
{
int8_t confl = easyvr.getWord();
if (confl >= 0)
Serial.print(" times, Similar to Word ");
else
{
confl = easyvr.getCommand();
Serial.print(" times, Similar to Command ");
}
Serial.println(confl);
}
}
}
}
}
group = 0;
mask |= 1; // force to use trigger
useCommands = (mask != 1);
}
const char* ws0[] =
{
"ROBOT",
};
const char* ws1[] =
{
"ACTION",
"MOVE",
"TURN",
"RUN",
"LOOK",
"ATTACK",
"STOP",
"HELLO",
};
const char* ws2[] =
{
"LEFT",
"RIGHT",
"UP",
"DOWN",
"FORWARD",
"BACKWARD",
};
const char* ws3[] =
{
"ZERO",
"ONE",
"TWO",
"THREE",
"FOUR",
```

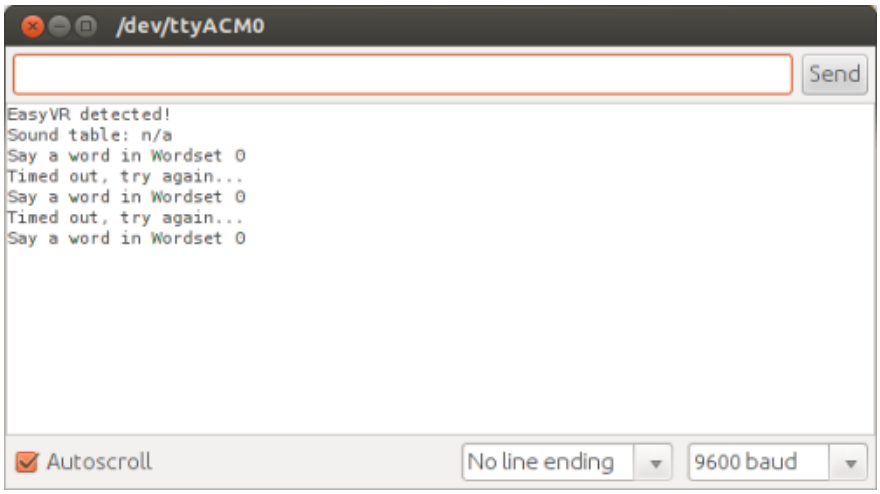
```
"FIVE",
"SIX",
"SEVEN",
"EIGHT",
"NINE",
"TEN",
};
const char** ws[] = { ws0, ws1, ws2, ws3 };
bool checkMonitorInput()
{
if (Serial.available() <= 0)
return false;

// check console commands
int16_t rx = Serial.read();
if (rx == 'b')
{
useCommands = false;
set++;
if (set > 3)
set = 0;
}
if (rx == 'c')
{
useCommands = true;
do
{
group++;
if (group > EasyVR::PASSWORD)
group = 0;
} while (!((mask >> group) & 1));
}
if (rx == 's')
{
int16_t num = 0;
delay(5);
while ((rx = Serial.read()) >= 0)
{
if (isdigit(rx))
num = num * 10 + (rx - '0');
else
break;
delay(5);
}
if (rx == '.')
{
easyvr.stop();
easyvr.playSound(num, EasyVR::VOL_DOUBLE);
}
}
if (rx >= 0)
{
easyvr.stop();
Serial.flush();
return true;
}
return false;
}
void loop()
{
```

```
checkMonitorInput();

easyvr.setPinOutput(EasyVR::IO1, HIGH); // LED on (listening)
if (useCommands)
{
Serial.print("Say a command in Group ");
Serial.println(group);
easyvr.recognizeCommand(group);
}
else
{
Serial.print("Say a word in Wordset ");
Serial.println(set);
easyvr.recognizeWord(set);
}
do
{
if (checkMonitorInput())
return;
}
while (!easyvr.hasFinished());

easyvr.setPinOutput(EasyVR::IO1, LOW); // LED off
int16_t idx = easyvr.getWord();
if (idx >= 0)
{
Serial.print("Word: ");
Serial.print(easyvr.getWord());
Serial.print(" = ");
if (useCommands)
Serial.println(ws[group][idx]);
else
Serial.println(ws[set][idx]);
// ok, let's try another set
set++;
if (set > 3)
set = 0;
easyvr.playSound(0, EasyVR::VOL_FULL);
}
else
{
idx = easyvr.getCommand();
if (idx >= 0)
{
Serial.print("Command: ");
Serial.print(easyvr.getCommand());
if (easyvr.dumpCommand(group, idx, name, train))
{
Serial.print(" = ");
Serial.println(name);
}
else
Serial.println();
// ok, let's try another group
do
{
group++;
if (group > EasyVR::PASSWORD)
group = 0;
} while (!((mask >> group) & 1));
```

```
easyvr.playSound(0, EasyVR::VOL_FULL);
}
else // errors or timeout
{
if (easyvr.isTimeout())
Serial.println("Timed out, try again...");
int16_t err = easyvr.getError();
if (err >= 0)
{
Serial.print("Error ");
Serial.println(err, HEX);
}
}
}
}
```

Na IDE do Arduino, selecione a versão da sua placa Arduino (UNO, Duemilanove, etc) e a porta (COMx, ttyUSB0, ttyACMx) e clique UPLOAD. Abra o Serial Monitor e selecione "9600" de baud. digite a letra "b" e aperte ENTER.

A figura abaixo mostra o Serial Monitor rodando a programação exemplo:



O Wordset 0 é o Trigger, isto é, diga a palavra "ROBOT", ao acertar a palavra, ele mostrará a palavra e irá para o Wordset1 e assim por diante.

E é isso! Esperamos que gostado! Caso tenha dúvidas, poste aqui mesmo no blog! Se tiver sugestões para tutoriais, poste aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG ou por outros garagistas, clique aqui e aqui! Até a próxima!

Referências:

http://www.labdegaragem.org/loja/index.php/31-shields/easyvr-shield...

http://www.sparkfun.com/products/10963

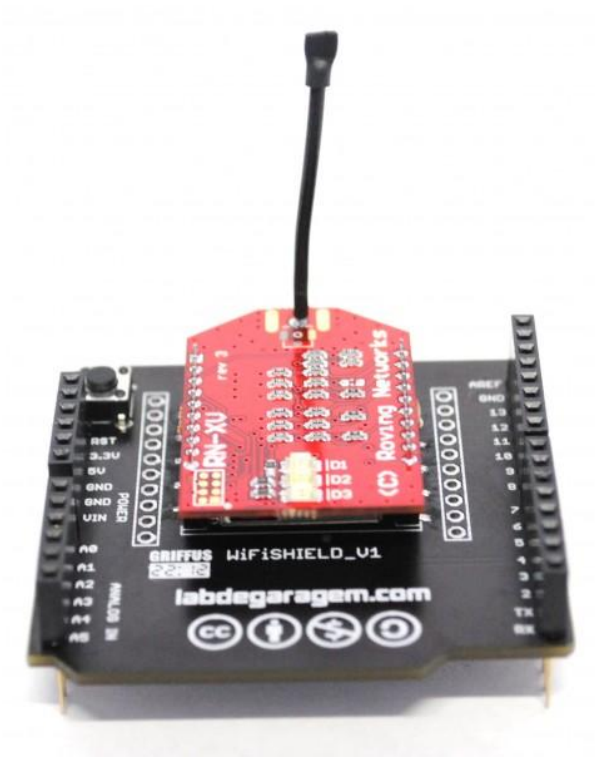
http://www.tigal.com/product/2333

http://download.tigal.com/veear/EasyVR-Commander-3.3.7-QS-5.2.1.zip

http://download.tigal.com/veear/EasyVR-Arduino-library-1.1.zip

http://download.tigal.com/veear/EasyVR_User_Manual_3.3.pdf

86. Utilizando o WiFiShield do LdG e Arduino



O WifiShield é um Shield desenvolvido pelo LdG para acoplar o módulo Wifly da empresa Roving Netwoks. Este Shield funciona como o Ethernet Shield, mas a vantagem é que não tem cabo e é muito fácil implementá-lo e utilizálo! É apenas acoplar o módulo Wifly no WifiShield, depois no Arduino, configurá-lo e pronto!

O modulo Wifly é baseado no robusto RN-171 que incorpora IEEE 802.11b/g, processador de 32 bits, pilha TCP/IP, real time, unidade de gerenciamento de alimentação e entradas para sensores analógicos, carrega um firmware desenvolvido pelo fabricante que facilita a integração e minimiza o tempo de desenvolvimento. Neste modelo utilizado, o módulo necessita de apenas quatro conexões (PWR, TX, RX e GND) para criar uma conexão de dados sem fio.

Para este tutorial será preciso instalar as bibliotecas clicando nos links abaixo,:

WiFlyHQmaster

Streaming3

Pstring5

Time

Primeiros passos.

Neste tutorial mostraremos como fazer uma conexão como Client em uma pagina HTML.

Primeiramente baixar as bibliotecas citadas acima. Depois abra o exemplo “ httpclient.ino” fornecido na biblioteca WiflyHQ e mude alguns parâmetros no Sketch, conforme abaixo:

1ª Trocar os pinos da serial SoftwareSerial wifiSerial(8,9); para SoftwareSerial wifiSerial(4,5);

São os pinos serial que o arduino se comunica com WiFly.

2ª Preencher os dados de sua rede como Nome e Senha.

const char mySSID[] = "nome da rede WiFi";

const char myPassword[] = "senha da rede WiFi";

3º Clique em Upload para passar a programação para o Arduino;

4º Abra o SerialMonitor e configure a serial para 115200 baud;

No terminal você deverá ter este resultado como mostrado abaixo.

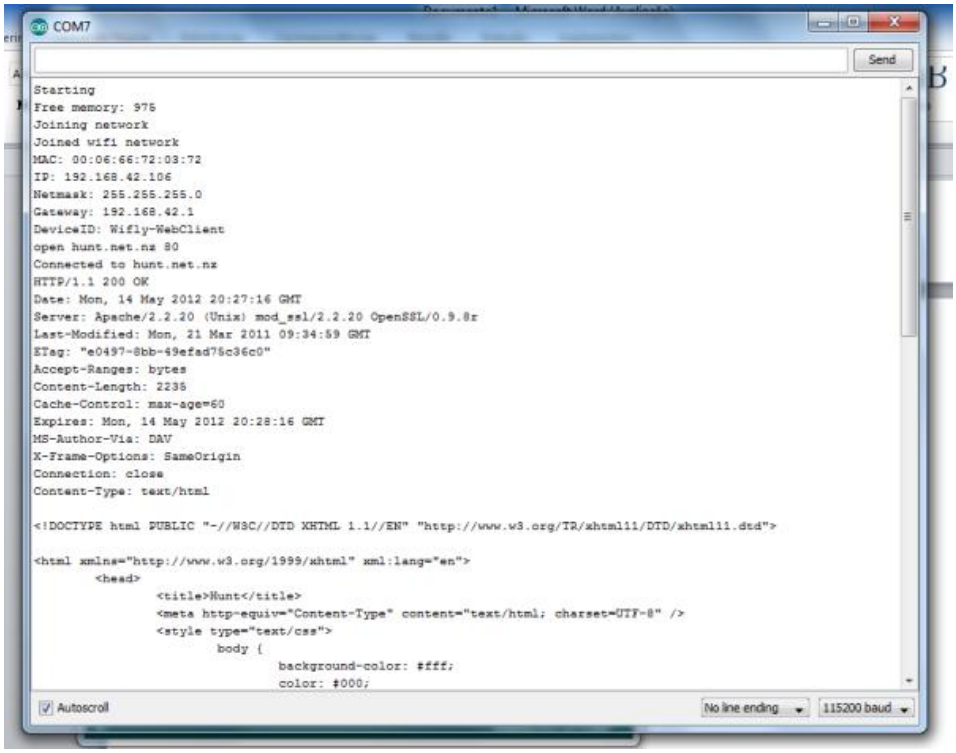


Fig 3 Serial Terminal

Funcionamento deste programa:

Este software configura o modulo WiFly para se conectar a rede Wi-Fi indicada pelo programa, após esta conexão estabelecida ele enviará um comando “GET” para as páginas determinadas pelo código abaixo.

Código:

```
//  carrega as bibliotecas
#include <WiFlyHQ.h>
#include <SoftwareSerial.h>

//define os pinos utilizados pela serial
SoftwareSerial wifiSerial(4,5);

//cria uma instancia chamada wifly
WiFly wifly;

/* trocar os dados com os de sua Rede WiFi*/
const char mySSID[] = "teste"; //Troque o teste pelo nome da sua rede wifi
const char myPassword[] = "teste"; //Troque o teste pela senha da sua rede wifi

/*Pagina a ser carregada*/
const char site[] = "www.labdegaragem.com";
//const char site[] = "arduino.cc";
//const char site[] = "www.google.co.nz";
//const char site[] = "hunt.net.nz";

void setup()
{
char buf[32];//define o tipo e tamanho da variável buf
//define a velocidade da porta serial do Monitor
Serial.begin(115200);
Serial.println("Starting");
Serial.print("Free memory: ");
```

```
Serial.println(wifly.getFreeMemory(),DEC);
//define a velocidade da porta serial do WiFly
wifiSerial.begin(9600);
//Inicializa o modulo WiFly
if (!wifly.begin(&wifiSerial, &Serial)) {
Serial.println("Failed to start wifly");
terminal();
}
/*associa-se a uma rede WiFi se não associado*/
if (!wifly.isAssociated()) {
/*carrega os parâmetros da rede WiFi, informados anteriormente*/
Serial.println("Joining network");
wifly.setSSID(mySSID);
wifly.setPassphrase(myPassword);
wifly.enableDHCP();
//se conecta na rede WiFi
if (wifly.join()) {
Serial.println("Joined wifi network");
}
else {
Serial.println("Failed to join wifi network");
terminal();
}}
else {
Serial.println("Already joined network");
}
//terminal();
Serial.print("MAC: "); //mostra o endereço MAC do Modulo
Serial.println(wifly.getMAC(buf, sizeof(buf)));
Serial.print("IP: ");//mostra o endereço IP
Serial.println(wifly.getIP(buf, sizeof(buf)));
Serial.print("Netmask: ");//mostra a Mascara de Rede
Serial.println(wifly.getNetmask(buf, sizeof(buf)));
```

```
Serial.print("Gateway: ");//mostra o endereço do Gateway
Serial.println(wifly.getGateway(buf, sizeof(buf)));

wifly.setDeviceID("Wifly-WebClient");
Serial.print("DeviceID: ");
Serial.println(wifly.getDeviceID(buf, sizeof(buf)));

if (wifly.isConnected()) { //fechar conexão anterior caso esteja aberta
Serial.println("Old connection active. Closing");
wifly.close();
}
/*manda uma requisição para pagina caso esteja conectado*/
if (wifly.open(site, 80)) {
Serial.print("Connected to ");
Serial.println(site);

wifly.println("GET / HTTP/1.0");
wifly.println();
}
else {
Serial.println("Failed to connect");
}
}

void loop()
{   //lê a porta serial do WiFi e imprime o resultado no terminal
if (wifly.available()) {
char ch = wifly.read();
Serial.write(ch);
if (ch == '\n') {
/*adiciona uma nova linha*/
Serial.write('\r');
} }
if (Serial.available() > 0) {
wifly.write(Serial.read());
}}
/* Conecta o modulo WiFly,  com o serial monitor. */
```

```
void terminal()
{
while (1) {
if (wifly.available() > 0) {
Serial.write(wifly.read());
}
if (Serial.available() > 0) {
wifly.write(Serial.read());
}}}
```

E é isso aí!! Boa diversão!!! Caso tenham dúvidas, poste aqui no blog! Para sugestões, clique aqui! Para ver outros tutoriais, clique aqui e projetos abertos, clicando aqui!

Referencias:

http://www.sparkfun.com/products/10822

https://github.com/harlequin-tech

http://rovingnetworks.com/products/RN_XV

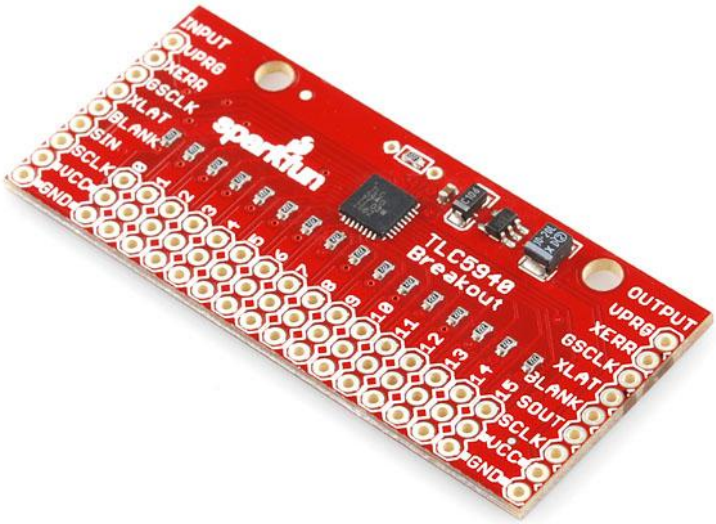
WiflyHQ - **https://github.com/harlequin-tech/WiFlyHQ**

Streaming3 - **http://arduiniana.org/libraries/streaming/**

Pstring5 - **http://arduiniana.org/libraries/pstring/**

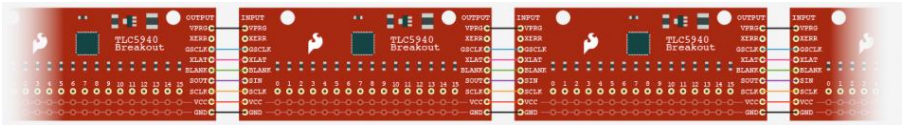
Time - **http://arduino.cc/playground/Code/Time**

87. Tutorial: como utilizar TLC5940 com Arduino para controlar vários servos

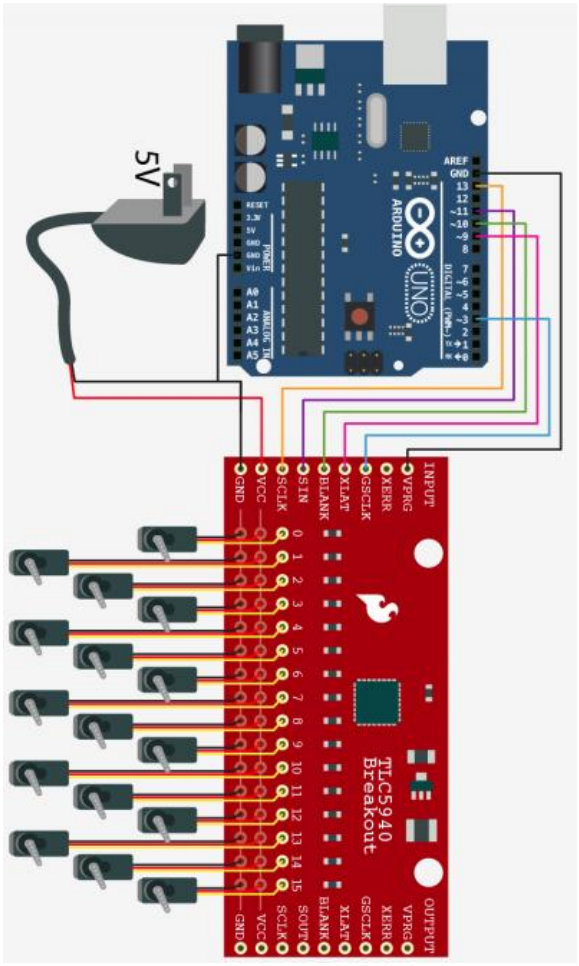


Neste tutorial vamos mostrar como controlar vários servos utilizando a breakout TLC5940 com Arduino.

Esta breakout TLC5940 pode controlar até 16 servos, mas pode juntar várias breakout TLC5940 em série e controlar mais servos. Para conectar em série é só fazer a ligação da figura abaixo:



Para conectar a breakout TLC5940 no Arduino faça a seguinte ligação:



É aconselhável utilizar uma fonte externa de 5V para alimentar os servos.

Agora, baixe a biblioteca disponível aqui e extraia na pasta "libraries" localizada dentro da pasta da IDE do Arduino. Depois abra a IDE do Arduino e passe a seguinte programação:

```
//From the bildr article: http://bildr.org/2012/03/servos-tlc5940-arduino
//Requires the Tlc5940 library. http://code.google.com/p/tlc5940arduino/downloads/list
#include "Tlc5940.h"
#include "tlc_servos.h"
int numberOfServos = 9; //how many servos on the chain?
void setup(){
tlc_initServos(); // Note: this will drop the PWM freqency down to 50Hz.
}
void loop(){
//loop through all the servos and move one at a time to 180º
for(int i = 0; i<numberOfServos; i++){
for (int angle = 0; angle < 180; angle+= 10) {
tlc_setServo(i, angle);
Tlc.update();
delay(20);
}
delay(200);
}
//loop through all the servos and move one at a time to 0º
for(int i = 0; i<numberOfServos; i++){
```

```
for (int angle = 180; angle >= 0; angle-= 10) {
tlc_setServo(i, angle);
Tlc.update();
delay(20);
}
delay(200);
}
}
```

Na IDE do Arduino, vá em "Tools/board" e selecione a versão da placa Arduino (UNO, Duemilanove, etc) e a porta em "Tools/Serial port" (COMx, ttyUSBx, ttyACMx, etc) e clique em "UPLOAD".

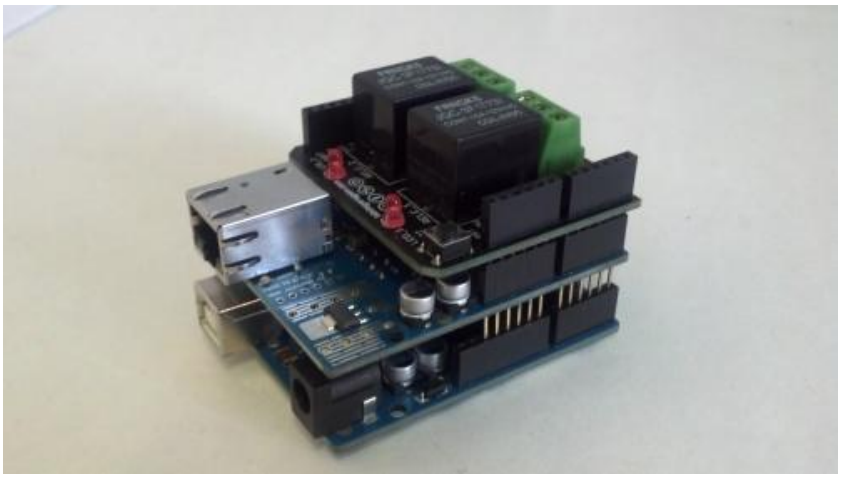
Ao terminar o UPLOAD, você verá cada servo indo para 180º e depois cada servo voltando para 0º.

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui mesmo neste blog! Se tiver sugestões para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://bildr.org/2012/03/servos-tlc5940-arduino/

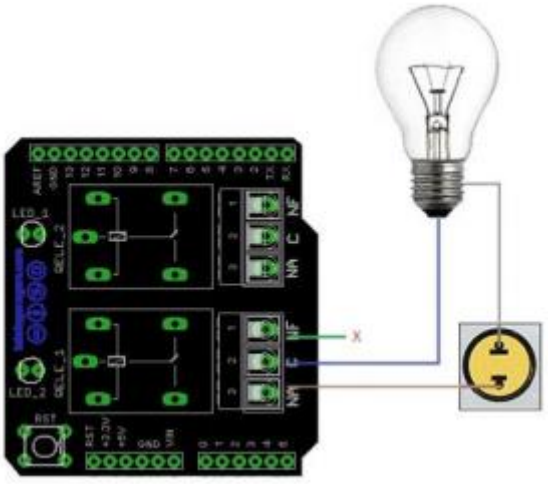
88. Tutorial: Utilizando o Releshield com Ethernet Shield e Arduino



Neste tutorial vamos mostrar como utilizar o Arduino com Releshield e Ethernet Shield para acender uma lâmpada através da internet.

Primeiramente, conecte o Releshield em cima do Ethernet Shield e depois conecte no Arduino.

Agora conecte a lâmpada no Releshield como mostrado na figura abaixo:



Conecte seu Arduino no PC, abra a IDE do Arduino e cole a seguinte programação:

/*Mude o mac[] para o mac da sua placa Ethernet Shield (números localizados embaixo da placa) e o IP de acordo com sua rede local. Por exemplo, se o seu roteador tiver o IP 192.168.1.1. Digite um IP 192.168.1.xxx. Sendo xxx de sua escolha. Cuidado para não colocar um IP igual de um PC de sua rede local. Aqui foi escolhido o final 177.

Cuidado também com a alimentação do Arduino. Utilize conectado ao USB do PC ou a uma fonte externa de 0 a 12V com 1A de corrente.*/

```
#include <SPI.h>
#include <Ethernet.h>
boolean incoming = 0;
int rele1 = 0;
int rele2 = 0;
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,1, 177);
EthernetServer server(80);
void setup()
{
Ethernet.begin(mac, ip);
server.begin();
Serial.begin(9600);
pinMode(8, OUTPUT);
digitalWrite(8, LOW);
pinMode(7, OUTPUT);
digitalWrite(7, LOW);
}
void loop()
{
// listen for incoming clients
EthernetClient client = server.available();
if (client) {
// an http request ends with a blank line
boolean currentLineIsBlank = true;
String str;
while (client.connected()) {
if (client.available()) {
char c = client.read();
str.concat(c);
if(str.endsWith("/1on")) rele1 =1;
else if(str.endsWith("/1off")) rele1 =0;
if(str.endsWith("/2on")) rele2 =1;
else if(str.endsWith("/2off")) rele2 =0;
if (c == '\n' && currentLineIsBlank) {
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println();
if(rele1 == 1)
{
client.println("ligado1" );
digitalWrite(7, HIGH);
}
else if (rele1 ==0)
{
client.println("desligado1" );
digitalWrite(7, LOW);
}
if(rele2 == 1)
{
client.println("ligado2" );
digitalWrite(8, HIGH);
}
else if (rele2 ==0)
{
client.println("desligado2" );
digitalWrite(8, LOW);
}
```

```
break;
}
if (c == '\n') {
currentLineIsBlank = true;
}
else if (c != '\r') {
currentLineIsBlank = false;
}
}
}
// give the web browser time to receive the data
delay(1);
// close the connection:
client.stop();
}
}
```

Depois disso, selecione a sua versão do Arduino (UNO, Duemilanove, etc) em Tools/board e a porta (COMx, ttyUSBx, ttyACMx, etc) em Tools/Serial Port. E clique em UPLOAD.

Agora abra seu navegador de internet (Mozilla Firefox, Google Chrome, Internet Explorer, Safari, etc) e digite o ip que você determinou adicionando o "/1on" no final. O Led do relé 1 irá ligar. Ao digitar novamente o ip com "/1off" no final, o LED do relé 1 irá desligar. Com o relé 2 faça a mesma coisa. Pronto agora você consegue ligar e desligar o lâmpadas como Releshield e Ethernet!

E é isso! Esperamos que tenha gostado! Em caso de dúvidas, poste aqui neste blog! Caso tenha sugestões para tutoriais, clique aqui! Para ver outros tutoriais e projetos desenvolvidos pela equipe LdG e por outros garagistas, clique aqui e aqui, respectivamente! Até a próxima!

Referências:

http://arduino.cc/en/

http://arduino.cc/en/Tutorial/WebServer

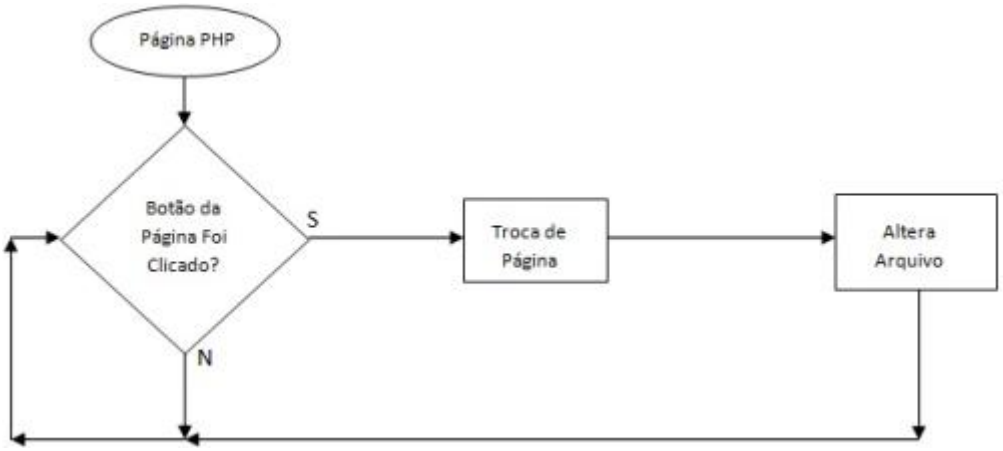
http://labdegaragem.com/forum/topics/rele-shield-ethernet-shield

89. Tutorial: Acionamento de Lâmpada via Internet com Android Mini PC e Arduino

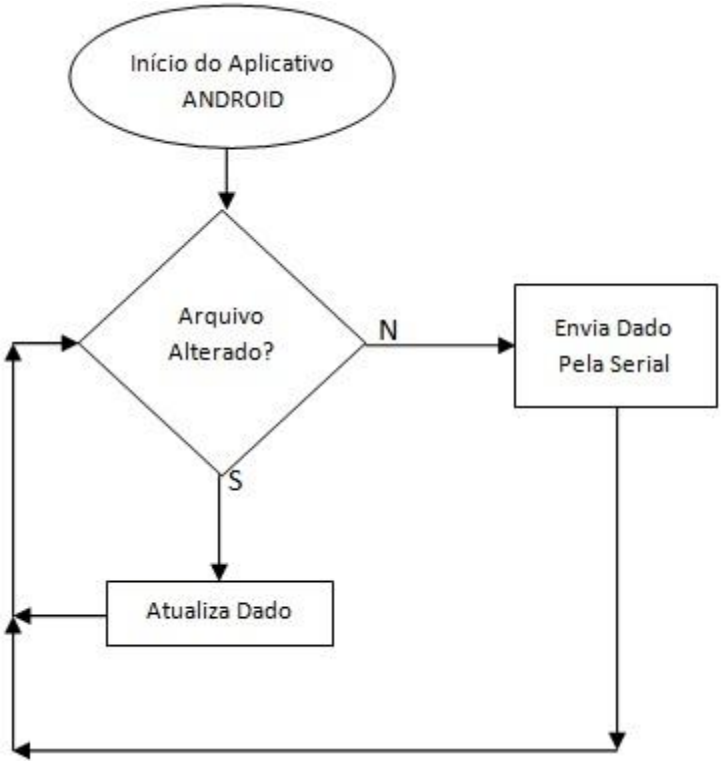
Olá Garagistas! Dando sequência aos tutoriais com o Android Mini PC, hoje faremos um projeto de automação via web com o Android Mini PC e o Arduino. Iremos controlar o acionamento de uma lâmpada utilizando um aplicativo Android que irá ler um arquivo .txt que foi gerado por um servidor PHP, onde o mesmo recebe ações de botões de uma página e altera o arquivo .txt, com isso a lâmpada é acionada, via comando serial enviado pelo Android Mini PC ao Arduino.

FLUXOGRAMA DA COMUNICAÇÃO

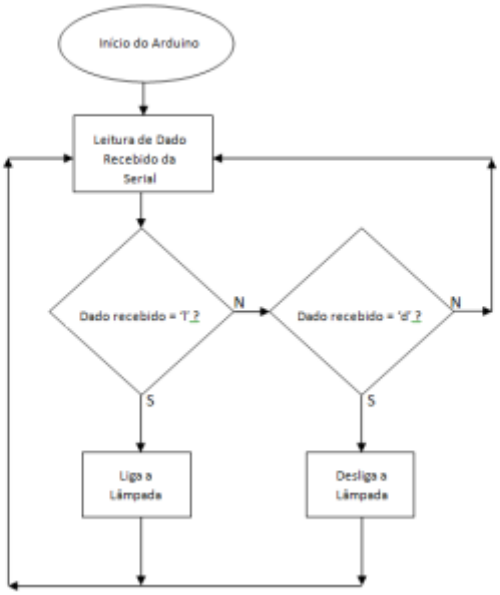
- Página PHP:



- Aplicativo ANDROID:



- Arduino UNO:



- Lista de Materiais:

1x Android Mini PC

1x Arduino Uno

1x Módulo Relé

1x Lâmpada 127V com bocal e tomada

1x Protoboard

Alguns Jumpers

CUIDADO: Evite manusear o Módulo Rele enquanto o mesmo estiver conectado à rede elétrica e remova qualquer tipo de objeto condutivo ou inflamável de perto. Você pode causar um acidente.

- Aplicativos Android:

phpandroid.apk

PAW Server

- Páginas HTML:

ldg.zip

- Lista de Softwares:

Projeto do aplicativo Android

Eclipse + Ferramentas Para Desenvolvimento Android

Java JDK

OBS: -Para a instalação e configuração do Servidor PHP (PAW Server) acesse este Tutorial.

-Para a montagem do circuito, siga os passos de montagem deste Tutorial.

1. O PAW Server

Após a configuração do PAW Server e a instalação do Plugin como foi feito neste Tutorial.

Copie e cole a pasta ldg para seu Android Mini PC e coloque-a dentro do diretório paw/html/:

Exemplo: paw/html/ldg/"arquivos da página"

2. O Aplicativo

O aplicativo foi desenvolvido usando o Eclipse ADT, que é um Eclipse com o plugin Android Development Tools (Ferramentas de Desenvolvimento Android).

Abaixo, esta o código do aplicativo onde você pode ver uma breve explicação de seu funcionamento. Ele foi feito em linguagem java, lembrando que esse não é todo o projeto (o projeto pode ser baixado no inicio do tutorial):

```
package com.example.phpandroid;



/*Aplicativo criado pelo LDG
para controlar seu Arduino via
Android Mini PC*/

///+++Bibliotecas+++
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;

import android.hardware.usb.UsbManager;

import android.os.Bundle;
import android.os.Environment;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.Context;
```

```
import android.content.Intent;
import android.view.View;
import android.view.Menu;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;
import jp.ksksue.driver.serial.FTDriver;
///=========================


public class MainActivity extends Activity
{
//+++DECLARAÇÃO DE VARIÁVEIS UTILIZADAS NO PROGRAMA+++
public TextView txtDir;
public TextView txtLer;
public TextView txtArq;
public String lerlinha;
public String wbuf;
public String nomeArq = "ldg.txt"; //Informa o arquivo que será lido
public Boolean serial;
//=====================================================


///+++PERMISSÃO PARA ACESSAR SERIAL+++
FTDriver mSerial;
public static final String ACTION_USB_PERMISSION = "jp.ksksue.tutorial.USB_PERMISSION";
///=================================


///+++CRIADO AUTOMATICAMENTE PELO ADT+++
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
///=================================
```

```
///+++CHAMA FUNÇÃO PARA CICLO DE LEITURA DO ARQUIVO+++
cicloleitura();
//=================================================

///+++DIRETÓRIO, ARQUIVO QUE ESTÁ SENDO LIDO E O QUE LEU NO ARQUIVO+++
txtDir = (TextView) findViewById(R.id.dir);
txtLer = (TextView) findViewById(R.id.edtLer);
txtArq = (TextView) findViewById(R.id.textView1);
///=================================================================

///++++RECEBE AS STRING PARA ESCREVER NA TELA++++
txtArq.append(nomeArq);
txtDir.append(GetRoot());
///=============================================

///+++INICIANDO SERIAL+++
mSerial = new FTDriver((UsbManager) getSystemService(Context.USB_SERVICE));
PendingIntent permissionIntent = PendingIntent.getBroadcast(this, 0, new
Intent(ACTION_USB_PERMISSION), 0);
mSerial.setPermissionIntent(permissionIntent);
///======================

///+++BOTÃO LIGADO/DESLIGADO+++
final ToggleButton bToggle = (ToggleButton) findViewById(R.id.toggleButton1); // Cria uma
varíavel para controle do botão
bToggle.setOnClickListener(new View.OnClickListener() // Monitora o botão
Ligado/Desligado
{
public void onClick(View v)
{
boolean botao; // Cria uma variável boleana
botao = bToggle.isChecked(); // Quando o botão for clicado guarda o estado dele na
variável botao
```

```
if(botao == true) // Se o botão estiver no estado "Ligado"
{
serial = mSerial.begin(FTDriver.BAUD9600); //Tenta iniciar e guarda resultado na variável
serial
if(serial != false) //Se o arduino não tiver conectado, retornará falso
{

Toast.makeText(MainActivity.this, "Arduino Não Conectado",
Toast.LENGTH_SHORT).show(); //Exibe mensagem "Arduino Não Conectado"
bToggle.setChecked(false);
}
else //Caso o arduino esteja conectado, retornará verdadeiro
{
Toast.makeText(MainActivity.this, "Serial Ativada", Toast.LENGTH_SHORT).show(); //Exibe
mensagem "Serial Inicializada"
}
}

else // Se o botão estiver no estado "Desligado"
{
mSerial.end(); //Encerra serial
Toast.makeText(MainActivity.this, "Serial Desativada",
Toast.LENGTH_SHORT).show(); //Exibe mensagem "Serial Finalizada"
}
}
});
}
///==============================================
///+++CICLO PARA LEITURA DO ARQUIVO+++
public void cicloleitura()
{
runOnUiThread(new Runnable()
{
```

```
public void run()
{
LerArquivo(); //Quando a Thread é Executada chama a função
Thread timer = new Thread()
{
public void run()
{
try
{
sleep(500);
}
catch (InterruptedException e)
{
e.printStackTrace();
}
finally
{
cicloleitura();
}
}
};
timer.start();
}
});
}
///================================


///+++FUNÇÃO PARA LEITURA ARQUIVO+++
public void LerArquivo()
{
File arq;
try
{
txtLer.setText("");
```

```
arq = new File(Environment.getExternalStorageDirectory(), nomeArq);
BufferedReader br = new BufferedReader(new FileReader(arq));
lerlinha = br.readLine(); //Lê a primeira linha do arquivo
wbuf = lerlinha; //Armazena o que foi lido ("l" ou "d")
txtLer.append(lerlinha); //Mostra na tela do aplicativo o que foi lido no arquivo
mSerial.write(wbuf.getBytes()); //Envia o valor de wbuf para a serial
}
catch (Exception e)
{
}
}
///==================================

///+++EXIBE DIRETÓRIO DO ARQUIVO DA APLICAÇÃO+++
public String GetRoot()
{
File root = android.os.Environment.getExternalStorageDirectory();
return root.toString();
}
///==========================================


///+++CRIADO AUTOMATICAMENTE PELO ADT+++
public boolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
///====================================

///+++INICIA SERIAL QUANDO O APP EH EXECUTADO+++
public void onStart(){
super.onStart();
serial = mSerial.begin(FTDriver.BAUD9600);
```

```
}
///=========================================

///+++PERMITE QUE O APLICATIVO CONTINUE EXECUTANDO MESMO QUANDO
MINIMIZADO++++
public void onPause()
{
super.onPause();
LerArquivo();
}
///=========================================================

///+++ ENCERRA SERIAL QUANDO O APP EH FECHADO+++
public void onDestroy()
{
String wbuf = "d"; // Armazena d(Desliga) na variável wbuf de escrita
mSerial.write(wbuf.getBytes()); // Envia o valor de wbuf para a serial
mSerial.end(); //Encerra serial
super.onDestroy();
}
///=========================================

}
```

3. O Sketch do Arduino

```
/* Sketch para programar o arduino e
ler a serial, onde os comandos são enviados
pelo aplicativo android lablampada.apk
*/
```

```
char leitura;      // Cria um variável char "leitura"

#define rele 8     // Define o valor 8 à variável rele


void setup()

{

Serial.begin(9600);     //Inicializa comunicação Serial

pinMode(rele, OUTPUT);   //Seta o pino pelo valor no "#define rele" como saída

digitalWrite(rele, LOW);   //Mantém rele desligado assim que iniciar o programa

}

void loop()

{

while (Serial.available() > 0) //Verifica se há conexão com a serial

{

leitura = Serial.read(); //Lê o dado vindo da Serial e armazena na variável leitura

if (leitura == 'l')  //Se a variável leitura for igual a 'l'(Botão "Ligado" no App)

{

digitalWrite(rele, HIGH);

}

else if (leitura == 'd') //Senão verifica se é igual a 'd' (Botão "Desligado" no App)

{

digitalWrite(rele, LOW);

}

}

}
```

4. A Montagem

A montagem é a mesma que foi utilizada neste Tutorial.

5. A Execução do Aplicativo

5.1) Execute e Inicie o Servidor com o PAW Server, conforme a figura abaixo.

5.2)Execute o aplicativo no seu Android Mini PC e clique no botão "Ativar Serial" para iniciar a comunicação serial do seu Android Mini PC com o Arduino, conforme a figura abaixo:

5.3) Abra seu browser e acesse a página php que foi colocada na sua pasta dentro do servidor php:

Seu.ip.gerado.pelo.paw:server/nome_do_arquivo.php

No nosso exemplo: 192.168.0.107:8080/ldg/lampada.php

5.4) A página conforme a figura abaixo será aberta, e ao acessar a página lampada.php, ela irá criar um arquivo .txt na memória interna do seu Android Mini PC, no nosso exemplo o arquivo é o ldg.txt que será atualizado a cada ação dos botões no nosso WebSever:

5.5) Ao clicar no botão "Ligar", a página l.php(figura abaixo) irá abrir e escrever no arquivo ldg.txt a letra "l"

5.6) Como o nosso aplicativo Android ficará lendo esse arquivo constantemente, ele enviará "l" pela serial.

5.7) O Arduino irá receber a letra "l", e com isso ligará a lâmpada

5.8) Quando o botão "Desligar" for clicado, a página d.php(figura abaixo) irá abrir e escrever no arquivo ldg.txt a letra "d"

5.6) Como o nosso aplicativo Android ficará lendo esse arquivo constantemente, ele enviará "d" pela serial.

5.7) O Arduino irá receber a letra "d", e com isso desligará a lâmpada

Para Alterar o arquivo a ser lido:

1) Será necessário o nome do arquivo na linha "public String nomeArq = "ldg.txt";" que está contida dentro do projeto do nosso aplicativo.

2) Será necessário trocar os arquivos .php nos arquivo do seu servidor PHP alterando o nome do arquivo "ldg.txt" para outro de sua preferência.

Então é isso pessoal, esperamos que tenham gostado desse tutorial, até a próxima!

Referências:

http://pt.wikibooks.org/wiki/Aplicativos_em_PHP/Trabalhando_em_PHP_...

http://escoladeandroid.blogspot.com.br/2012/02/android-trabalhando-...

http://imasters.com.br/artigo/20205/java/how-to-implementando-threa...

http://www.devmedia.com.br/blocos-try-catch/7339

http://desenhosmaiquel.blogspot.com.br/2009/11/desenhos-antigos.html

90. Tutorial - UnoJoy utilize seu Arduino Uno como um Joystick

Olá Garagistas!!!

Neste tutorial iremos fazer um passo a passo de como transformar seu Arduino UNO R3 em um Joystick com o UnoJoy. Ele é totalmente compatível com as plataformas Windows, OSX e PS3, além de ser é capaz de simular todas as teclas do controle de PS3.

Iremos fazer todo o procedimento no Windows 7 x64

Lista de Materiais

1 x Arduino Uno Rev 3

1 x Joystick Shield

1 x Jumper

Softwares

No Windows você deve instalar o software Flip da Atmel (neste pacote já vem com JRE)

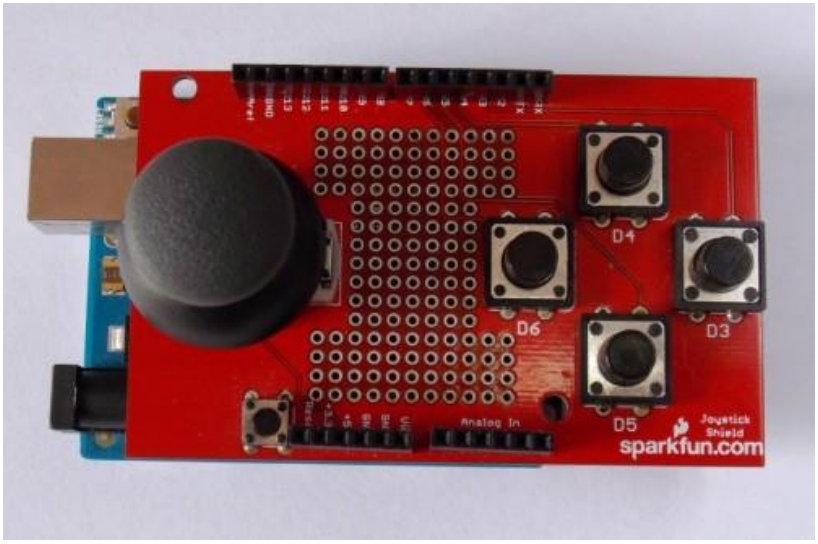
Baixe a pasta do projeto UnoJoy e descompacte onde preferir. Utilizamos os arquivos deste link:

UnoJoyWin-21Feb2013.zip

Para outros sistemas operacionais veja a página do projeto.

Hardware

Neste tutorial a única montagem com Hardware necessária é colocar o Joystick Shield no Arduino Uno R3



Testando as teclas

Depois de instalar os softwares descritos acima e descompactar a pasta do UnoJoy, baixe este Sketch feito para o Joystick Shield e grave no seu Arduino.

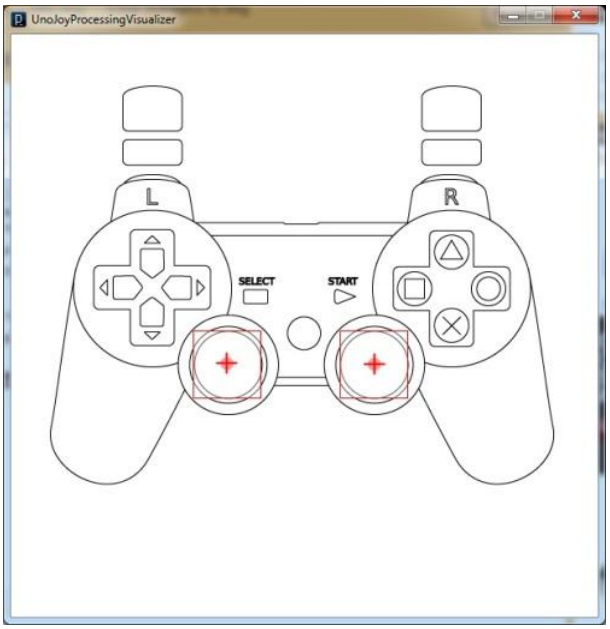
Neste Sketch, fizemos algumas modificações no exemplo do UnoJoy para que funciona-se com o Joystick Shield.

Mantenha o Arduino Conectado na USB.

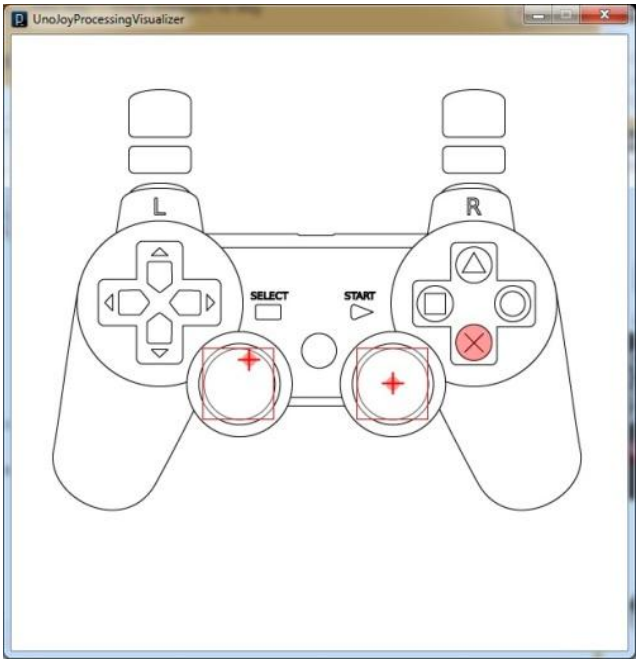
Agora execute o programa que esta na pasta que você descompactou o UnoJoy no caminho abaixo:

UnoJoyWin > UnoJoyProcessingVisualizer > UnoJoyProcessingVisualizer.exe

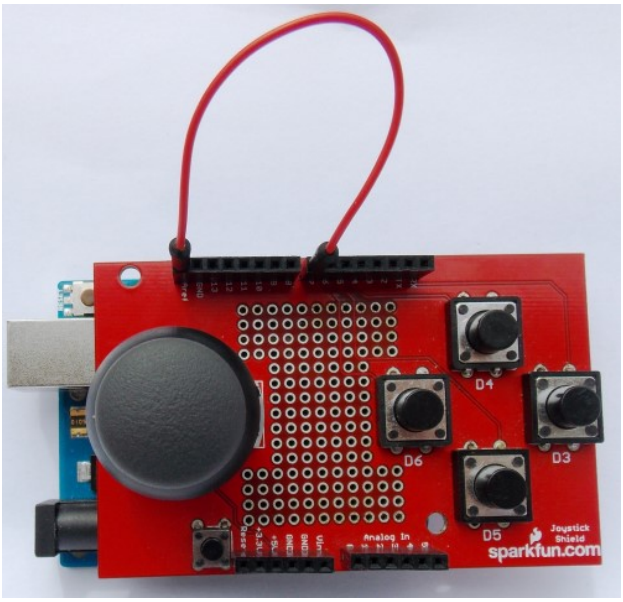
Ao fazer isto aparecerá esta tela.



Mexa o Joystick e você verá na tela as teclas sendo acionadas.



O botão Start pode ser acionado com um Jumper entre o pino D7 e o GND (deixamos ele no pino Aref pois não queremos o start sempre acionado).



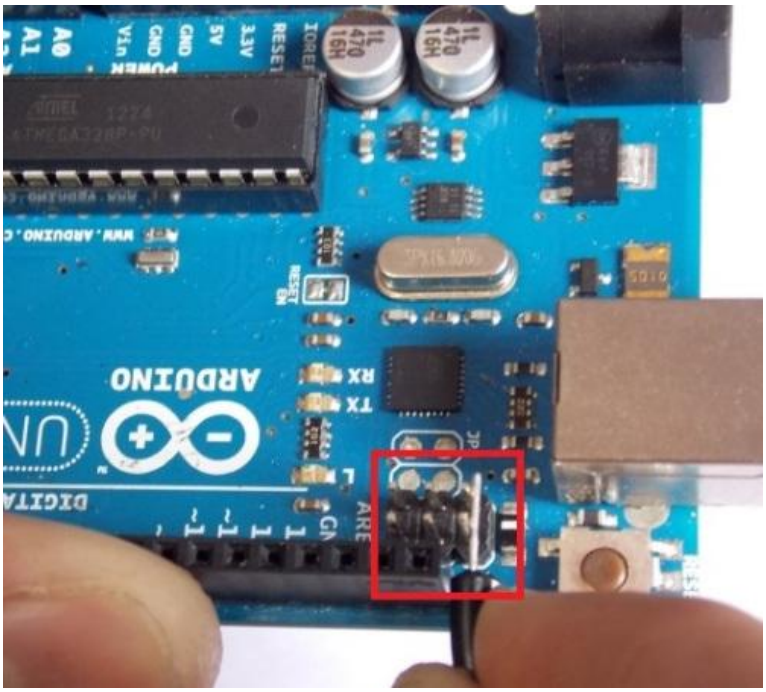
Com a teclas testadas, retiramos o Arduino da USB e tiramos o Joystick Shield.

Transformando Uno em Joystick

Cuidados:

- *Ao fazer o procedimento abaixo você vai modificar o firmware do conversor USB/Serial que esta na placa do Arduino. Isto vai mudar a forma como ele é reconhecido pelo computador. Faça por sua conta em risco.*
- ***Todo o processo pode ser desfeito (leia como no fim to tutorial).***
- *Evite manusear o Arduino com anel e qualquer outra coisa metálica.*

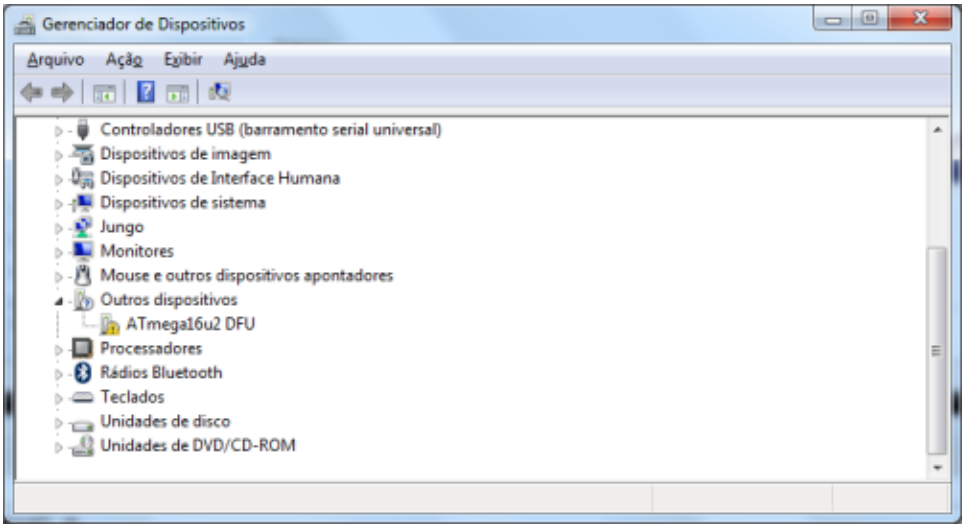
Reconecte o Arduino no PC e dê um curto nos pinos que são mostrados na imagem abaixo:



Tire o jumper de perto do Arduino para evitar curto-circuito (mas mantenha ele conectado ao PC).

Isto serve para poder ser reprogramado o chip de comunicação USB/Serial (ATmega8u2 ou ATmega16u2) e deixa-lo em modo DFU (Device Firmware Update).

No gerenciador de dispositivos o Arduino será reconhecido como mostra a figura abaixo:



Clique em:

UnoJoyWin > UnoJoyDriverInstaller.bat

E se o driver ainda não for instalado, mande procurar nesta pasta:

UnoJoyWin > Drivers

Instalado o dispositivo, certifique-se se você instalou o FLIP e execute o arquivo:

UnoJoyWin > TurnIntoAJoystick.bat

Uma tela como esta irá aparecer:

Desconecte o Arduino da USB e reconecte.

Clique no ícone do *Windows* e em *Dispositivos e Impressoras.*

Você verá que o Joystick esta instalado.

***ATENÇÃO:* Para reverter o processo, você deve repetir os passo do item "Transformando Uno em Joystick" e ao chegar no ponto de executar o arquivo "TurnIntoAJoystick.bat" execute "TurnIntoAnArduino.bat" que esta dentro da pasta do UnoJoyWin.**

É isto pessoal!!! Agora configure seu UnoJoy em seu emulador preferido ou ainda em seu PS3 e ponha os dedos para trabalhar. Se gostaram do tutorial, ficaram com dúvidas ou tem algum comentário a fazer, deixe ele abaixo.

\o/

Referências

https://code.google.com/p/unojoy/

http://arduino.cc/en/Hacking/DFUProgramming8U2

91. Tutorial: Controlando Pan/Tilt com servos, módulo de Ethernet e Garagino

Olá Garagistas!!! Neste tutorial baseado no exemplo do WebServer disponível na IDE do Arduino, criamos um server com Garagino e módulo Ethernet Wiznet, e com isto uma página HTML que mostra os botões que controlam os servos conectados no Pan/Tilt. Você também pode fazer a montagem com o Arduino Uno e o Ethernet Shield.

Lista de Materiais

1 x Garagino Rev 1 ou Arduino Uno Rev 3

1 x Módulo Ethernet Wiznet ou Ethernet Shield

2 x Servomotores Pequenos

1 x Suporte Pan/Tilt

1 x Protoboard

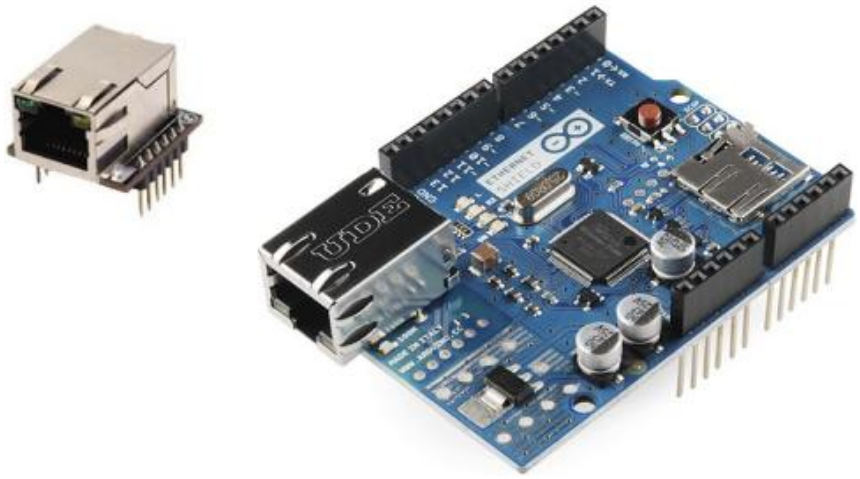
Alguns jumpers

Softwares

WebcamXP 5

Biblioteca para módulo Ethernet Wiznet

Módulos Ethernet



Módulo de rede Wiznet(esq.) e Ethernet Shield(dir.)

Você pode utilizar os dois módulo de rede, tanto o o módulo da Wiznet quanto o Ethernet Shield. Ambos funcionam via interface SPI (Serial Peripheral Interface) e são baseados em respectivamente nos chips W5200 e W5100.

O módulo Wiznet tem a vantagem de ter um tamanho reduzido e também pode ser conectado na protoboard. Já o Ethernet Shield pode ser conectado diretamente no Arduino sem a necessidade de criar um ligações com jumper.

Para utilizar o módulo de rede Wiznet na IDE do Arduino é necessário baixar este arquivo e substituir os dois que estão na pasta *Libraries>Ethernet>Utility.* Serão substituidos dois arquivos nesta pasta o w5100.ccp e o w5100.h. Não esqueça de fazer o backup dos arquivos que existiam antes nesta pasta.Para o Ethernet Shield este procedimento não é necessário.

Vídeo da câmera para HTML

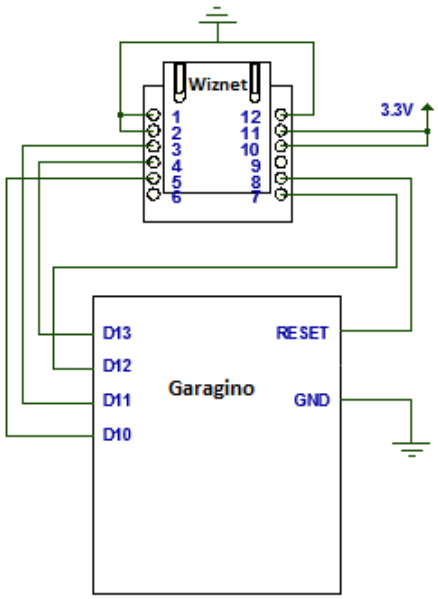
Para gerar o código HTML que mostra a imagem da câmera utilizamos o software WebcamXP 5. Ele tem uma versão Free que dá suporte a monitorar uma câmera. Para gerar o código vamos no menu *Tools> Generate HTML code* selecionamos o código como Flash JPEG e colamos o código no Arduino.

Imagem do Software WebcamXP 5

Com este software e um serviço de DDNS (como o No-Ip) podemos criar uma espécie de "câmera IP" com acesso remoto, mas claro que para isto você devemos fazer as devidas configurações em nossa rede.

O Circuito

Na imagem a abaixo podemos ver as ligações que devem ser feitas entre o Garagino e o módulo Wiznet.



Ligação entre o Garagino e Ethernet Wiznet

Os pinos de controle dos servo-motores devem ser ligados nos pinos D5 e D6.

A alimentação dos servos não deve ser feita pela porta USB, pois o consumo deles é elevado podendo danificar sua porta USB. Alimente eles com uma fonte separada.

A tensão de alimentação do módulo Ethernet Wiznet é de 3,3V. Os pinos de controle podem ser ligados diretamente sem o uso de acoplamentos.

O Sketch

Você pode baixar o programa completo que utilizamos neste tutorial clicando neste link. No vídeo você pode ver a explicação do código.

É pessoal espero que vocês automatizem suas câmeras e quem sabe não surge um sistema de segurança mais complexo a partir deste tutorial. Em caso de dúvidas ou sugestões comentem abaixo.

\o/

Referências

http://arduino.cc/en/Tutorial/WebServer

http://www.webcamxp.com/download.aspx

http://blog.wiznet.co.kr/intro-to-the-wiz820io-module-by-ben-roberts/#.UaTLeUCsiSo