

PDM: Projeto para Dispositivos Móveis

Aula 04: Utilizando Sensor de Luminosidade (*Light Dependent Resistor*)

Breno Lisi Romano

<http://sites.google.com/site/blromano>

Instituto Federal de São Paulo – IFSP São João da Boa Vista
Especialização em Desenvolvimento para Dispositivos Móveis



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Instituto Federal de São Paulo – IFSP São João da Boa Vista

Sensor de Luminosidade ou LDR (*Light Dependent Resistor*)

- É um componente cuja resistência varia de acordo com a intensidade da luz
 - Quanto mais luz incidir sobre o componente, menor a resistência
- Este sensor de luminosidade pode ser utilizado em projetos com arduino e outros microcontroladores para alarmes, automação residencial, sensores de presença e etc.

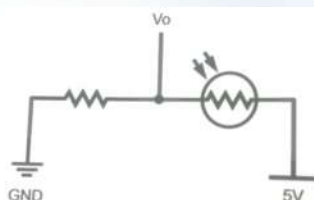


Problem Based Learnd - PBL 01

- **Problema:** Como ler valores de luminosidade de um sensor de Luz ou LDR e Apresentar em uma escala com 4 unidades qual é a intensidade da luminosidade

Informações Necessárias (Background)

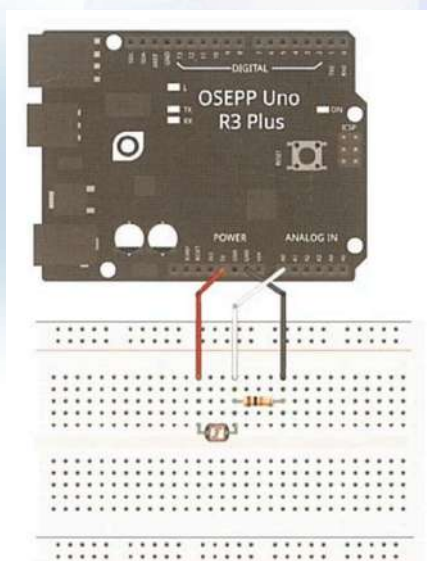
- O Sensor de Luminosidade se comporta como resistor, quanto maior luz aplicada ao sensor menor será a resistência
- Como as Portas Analógicas do Arduino leem Voltagem e não resistência, precisa-se construir um circuito que converte a resistência do sensor em uma voltagem que se consegue ler
- Para fazer isso, é necessário criar um divisor de voltagens adicionando um resistor in série com o sensor



Componentes Utilizados

- Arduino UNO / Arduino MEGA
- Cabo USB
- Protoboard
- 01 LDR
- 01 Resistor de 10k (Marrom, Preto, Laranja e Dourado)
- 03 Cabos de Jumpers

Construindo o Circuito no Protoboard



Funções Necessárias para o Desenvolvimento do Sistema Embarcado (1)

Função	Exemplo	Notas
Serial.begin(taxa) Essa função habilita a porta serial e fixa a taxa de transmissão e recepção em bits por segundo entre o computador e o Arduino.	Serial.begin(9600); Nesse exemplo essa função fixa a taxa de comunicação em 9600 bps. Os pinos digitais 0 e 1 não podem ser utilizados como entrada ou como saída de dados quando a porta serial é habilitada por essa função.	Essa função vai sempre dentro da função setup() .

Funções Necessárias para o Desenvolvimento do Sistema Embarcado (2)

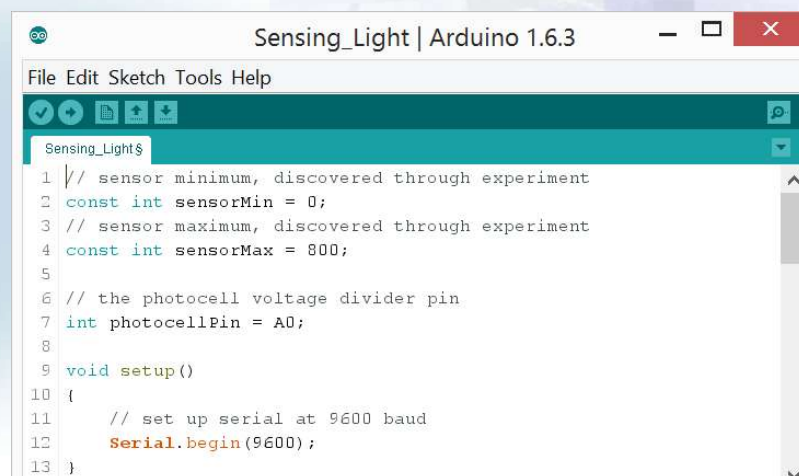
Função	Exemplo	Notas
analogRead(pino) Essa função lê o nível analógico presente no pino indicado pelo parâmetro entre parênteses e, após a conversão para o seu equivalente em bits, o guarda em uma variável determinada pelo programador.	int sensor = analogRead(A0); Aqui a variável inteira 'sensor' vai armazenar a tensão analógica convertida para digital presente no pino A0. Essa informação vai ser um valor inteiro entre 0 (para 0 volt no pino) e 1023 (se 5 volts no pino). Uma tensão de 2,5 volts no pino A0 vai fazer a variável 'sensor' guardar o valor inteiro 512.	Os pinos analógicos são reconhecidos pela linguagem C do Arduino tanto como A0 a A5 como 14 a 19. Assim, a mesma expressão acima pode ser escrita também da seguinte forma: int sensor = analogRead(14);

Uma observação importante em relação a esses pinos analógicos é que eles podem ser configurados também como pinos digitais pela função **pinMode()**, aumentando assim o número desses pinos para 20. Assim, a expressão **pinMode(14, OUTPUT);** transforma o pino analógico A0 em pino de saída digital como qualquer outro presente nas duas barras de pinos digitais.

Funções Necessárias para o Desenvolvimento do Sistema Embarcado (3)

Função	Exemplo	Notas
delay(ms) Essa função pausa o programa por um período em milissegundos indicado pelo parâmetro entre parênteses.	delay(1000); Com esse parâmetro o programa vai pausar durante 1 segundo (1000 ms).	Durante o período em que essa função está ativa qualquer outra função no programa é suspensa; é equivalente ao HALT em Assembly. Somente as interrupções de hardware podem parar essa função.
map(valor,min1,max1,min2,max2) A função map() converte uma faixa de valores para outra faixa. O primeiro parâmetro 'valor' é a variável que será convertida; o segundo e o terceiro parâmetros são os valores mínimo e máximo dessa variável; o quarto e o quinto são os novos valores mínimo e máximo da variável 'valor'.	int valor = map(analogRead(A0),0,1023,0,255); A variável 'valor' vai guardar a leitura do nível analógico no pino A0 convertida da faixa de 0-1023 para a faixa 0-255.	Com essa função é possível reverter uma faixa de valores, exemplo: int valor = map(x,1,100,100,1);

Código Fonte para Solução (1)



```

Sensing_Light | Arduino 1.6.3
File Edit Sketch Tools Help
Sensing_Light$
1 // sensor minimum, discovered through experiment
2 const int sensorMin = 0;
3 // sensor maximum, discovered through experiment
4 const int sensorMax = 800;
5
6 // the photocell voltage divider pin
7 int photocellPin = A0;
8
9 void setup()
10 {
11   // set up serial at 9600 baud
12   Serial.begin(9600);
13 }
  
```

Código Fonte para Solução (2)

```
15 void loop()
16 {
17     int analogValue; int range;
18
19     // read our photocell
20     analogValue = analogRead(photocellPin);
21
22     // map the sensor range to a range of four options
23     range = map(analogValue, sensorMin, sensorMax, 0, 3);
24
25     switch (range)
26     {
27         // your hand is on the sensor
28         case 0:
29             Serial.println("dark");
30             break;
31         // your hand is close to the sensor
32         case 1:
33             Serial.println("dim");
34             break;
35         // your hand is a few inches from the sensor
36         case 2:
37             Serial.println("medium");
38             break;
39         // your hand is nowhere near the sensor
40         case 3:
41             Serial.println("bright");
42             break;
43     }
44     // wait 0.25s before reading the photocell again
45     delay(250);
46 }
```

Verificação e Simulação do Código Desenvolvido

- Embarcar o código-fonte na aplicação
- Abrir o Serial Monitor
- Passar a mão sobre o Sensor LDR para verificar as mudanças de Luminosidade

Problem Based Learnd - PBL 02

- **Problema:** Como ligar automaticamente um LED quando o nível de luminosidade do sensor LDR realizar uma leitura de escuro

Componentes Utilizados

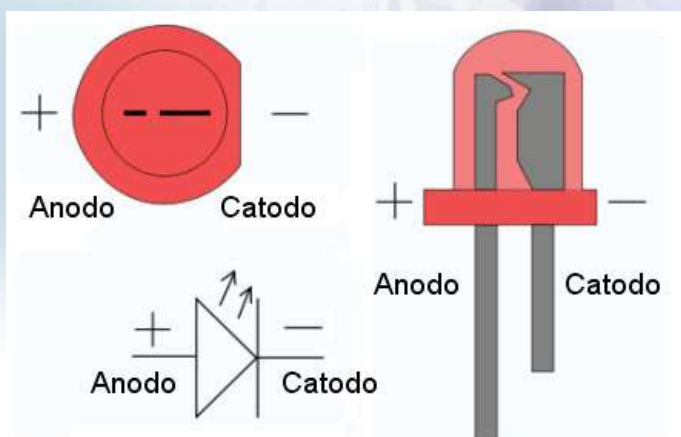
- Arduino UNO / Arduino MEGA
- Cabo USB
- Protoboard
- 01 LDR
- 01 Resistor de 10k ohm
- 01 Resistor de 330 ohm
- 01 LED
- 04 Cabos de Jumpers

Informações Necessárias (*Background*)

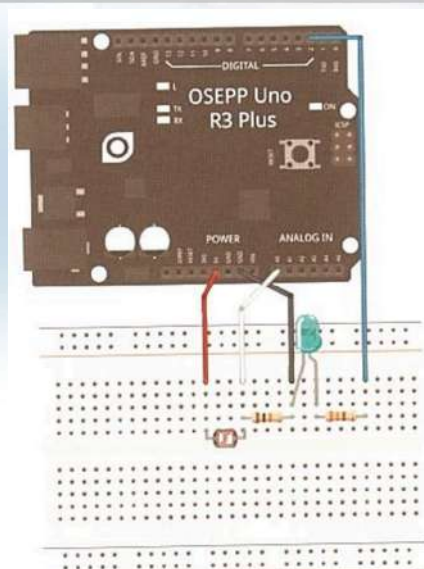
- O Arduino possui 14 Portas Digitais que podem ser configuradas como ENTRADA (**INPUT**) ou SAÍDA (**OUTPUT**)
- Pode-se configurar a entrada que o LED estará conectado ao Arduino como **OUTPUT** e utilizar a função **digitalWrite(LED, LOW)** para setar o pino como 0V e não ligar o LED ou **digitalWrite(LED, HIGH)** para setar o pino como 5V e ligar o LED
- O Resistor de 330 ohms é necessário para ligar o LED para limitar a corrente passada por ele
 - Revisar aula de Conceitos Básicos de Eletrônica

Informações Necessárias (*Background*)

- Como Ligar um LED?



Construindo o Circuito no Protoboard



Funções Necessárias para o Desenvolvimento do Sistema Embarcado

Função	Exemplo	Notas
pinMode(pino, modo) Serve para estabelecer a direção do fluxo de informações em qualquer dos 14 pinos digitais. Dois parâmetros devem ser passados à função: o primeiro indica qual pino vai ser usado; o segundo, se esse pino vai ser entrada ou se vai ser saída dessas informações.	pinMode(2, OUTPUT); Aqui o pino 2 é selecionado para transmitir informações do Arduino para um circuito externo qualquer. Para configurar esse pino como entrada, o segundo parâmetro dessa função deve ser INPUT.	Essa função é sempre escrita dentro da função setup() .
digitalWrite(pino, valor) Para enviar um nível lógico para qualquer pino digital do Arduino utiliza-se essa função. Dois parâmetros são requeridos: o número do pino e o estado lógico (HIGH/LOW) em que esse pino deve permanecer.	digitalWrite(2, HIGH); Aqui uma tensão de 5 volts é colocada no pino 2. Para enviar terra para esse pino o segundo parâmetro deverá ser LOW.	É necessário configurar previamente o pino como saída com a função pinMode() .

Código Fonte para Solução

```

1 // A constant that describes when its dark enough to
2 // light the LED. A value close to 600 will light the led
3 // with less darkness. Play with this number.
4 const int sensorDark = 600;
5
6 // the photocell voltage divider pin
7 int photocellPin = A0;
8 // the LED pin
9 int LEDPin = 2;
10
11 void setup()
12 {
13   pinMode(LEDPin, OUTPUT); // initialize the LED pin as output
14 }
15
16 void loop()
17 {
18   int analogValue;
19
20   // read our photocell
21   analogValue = analogRead(photocellPin);
22
23   // The higher the analogValue reading is the darker it is.
24   // If its atleast as dark as our constant "sensorDark"
25   // light the LED
26   if (analogValue < sensorDark)
27     digitalWrite(LEDPin, HIGH);
28   else
29     digitalWrite(LEDPin, LOW);
30   // wait 1ms for better quality sensor readings
31   delay(1);
32 }

```

Verificação e Simulação do Código Desenvolvido

- Embarcar o código-fonte na aplicação
- Passar a mão sobre o Sensor LDR para verificar se o LED acende

Obs: Mude o valor da constante sensorDark para encontrar um melhor valor para acender o LED

Miniprojeto 01

- Utilizar 3 LEDs (Branco, Vermelho e Amarelo) e acender cada hora um dos LEDs de acordo com a Luminosidade do Local detectada pelo sensor LDR
 - Luminosidade Baixa: LED Amarelo
 - Luminosidade Média: LED Vermelho
 - Luminosidade Alta: LED Branco

PDM: Projeto para Dispositivos Móveis

Aula 04: Utilizando Sensor de Luminosidade (*Light Dependent Resistor*)

Breno Lisi Romano

Obrigado!

Instituto Federal de São Paulo – IFSP São João da Boa Vista
Especialização em Desenvolvimento para Dispositivos Móveis



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista