

Trabalho prático 3

**Sensores e Atuadores**

Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e Multimédia

Ano Letivo 2018/2019

**Turma LEIM 11D**

**Docentes:**

Eng. Carlos Carvalho

Eng. Manfred Niehus

**Data:28/11/2018**

**Grupo:1**

**Alunos:**

Paulo Jorge (A45121)

Luís Fonseca (A45125)

João Santos (A45156)

**Índice**

Introdução………………………………………………………3

GrupoA…………………………………………………………4

GrupoB………………………………………………….……...13

Conclusão……………………………………………....………16

Bibliografia……………………………………………..………16

**Introdução**

Neste segundo trabalho prático da disciplina de sensores e atuadores, foi introduzido o estudo do Arduíno e a sua linguagem associada C++, e como se integra nos circuitos associados a matéria do primeiro trabalho prático. Como introdução a este trabalho foi posto, em prática, métodos para achar quando é que um botão é clicado ou não, e de seguida criar outro método no qual o Arduíno recebe um caracter “char”, quando clicado no teclado de um computador. De seguida foram usados os componentes LDR e potenciómetro, e tirar conclusões dos resultados obtidos, e ao mesmo tempo, programar, em Arduíno, métodos para os componentes associados a este trabalho, sendo o principal objetivo de ambos os componentes achar o valor da tensão analógica, e calcular os respetivos valores, nomeadamente a resistência e a tensão máxima associada. Ao longo do relatório vão sendo postos em prática todos os métodos, assim como vários testes para gerar o output e tirar conclusões acerca dos métodos referidos. Cada grupo do enunciado vai ser tratado como “blocos” no qual é onde vai estar a explicação em relação aos métodos implementados.

* Grupo A
* Bloco 1:

Para este primeiro bloco, o grupo tinha como objetivo, calcular o coeficiente “a” e dimensionar os seus componentes. De seguida comprovar a boa funcionalidade dos componentes através do voltímetro, e acabar com a implementação das funções: pressionar um botão, e a consola receber um caracter “char”. O processo usado para achar o coeficiente “a” e dimensionar a resistência foi o seguinte: primeiro achou-se os valores mínimos da resistência R1, e de seguida (e de acordo com o enunciado) para achar a nossa resistência R2 aplicou-se a média geométrica. Com a resistência encontrada, procedeu-se para o cálculo do coeficiente “a” através da expressão:

Figura - fórmula usada para o cálculo do coeficiente "a"

**Usando os valores de R1min e R1max:**

R1min =2.63kΩ

R1max = 9.53kΩ

**Fazendo a média geométrica:**

R2 =

Com este valor calculado, o grupo chegou a conclusão que é necessária uma resistência de 47(normalizada da serie E12, pois é a resistência que se encontra mais perto deste valor).

**Cálculo do coeficiente “a”:**

Considerando que a luminosidade é 100%:

2.63 = a \* 2.63 / = a a = 10.47

Com os cálculos efetuados, passou-se para a realização das funções pretendidas:

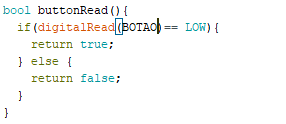
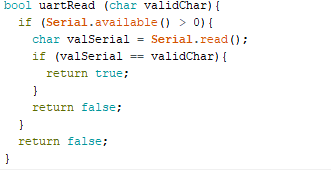


Figura - implementação da função"uartRead(char validChar)"

Figura - implementação da função "buttonRead()"

Para a função “uartRead()” recorreu-se à classe “Serial” do Arduíno, sempre que Serial.available() for maior que 0 (neste caso ser valores positivos) então declara-se outro char no qual irá conter o caracter selecionado. Caso essa condição seja verdadeira, retorna-se true, caso contrário false.

Para a função “buttonRead()” recorreu-se às funções do Arduíno “digitalRead()” no qual lê um pino e verifica se está a HIGH ou LOW. Fazendo uma condição “if” no qual quando o botão não esta pressionado (valor 0) retorna-se true, caso contrário, quando está pressionado, retorna false.

* Bloco 2 - Potenciómetro:

Neste exercício, era pedido que fosse estudado o potenciómetro, nomeadamente achar e calcular os valores da tensão máxima, resistência associada e as suas respetivas posições. Segue-se a implementação das funções pedidas em baixo:



Figura 4 - implementação das funções fornecidas para o estudo do potenciómetro

Para obtermos os valores de 10bits do potenciómetro, recorreu-se à função analogRead() do Arduíno, no qual retorna valores entre 0 e 1023. De seguida foi implementada a função “potReadVolt()” no qual consiste na conversão para a tensão que vai variar entre 0 e 5V, fazendo a divisão entre 5.0V e os 1023. A função “potReadRpot23()” consiste no cálculo da resistência do potenciómetro, para isso recorreu-se às fórmulas fornecidas no enunciado:



Figura 5 - fórmula usada para o cálculo da resistência23 do potenciómetro

De seguida, tivemos que a resistência do potenciómetro, neste caso a medida obtida, através do ohmímetro foi de: 10.75 kΩ, efetuando a divisão entre a resistência do potenciómetro e o nosso valor máximo (1023) conseguimos obter a resistência23 do potenciómetro. Por último fez-se a implementação do método “potReadpos()”, sendo que a posição do potenciómetro varia entre 0 e 100%, efetuou-se a divisão entre o valor da percentagem máxima e o valor de 1023, resultando em 0.098, para obter a posição, multiplicou-se pelo método “potRead10bits()”

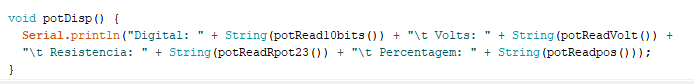
De seguida passou-se para a implementação da função “potDisp()” no qual serve para fazer a representação dos valores obtidos.

Figura 6 - função potDisp() para mostrar os resultados das várias funções implementadas

De seguida foi pedido para incluir um botão e um LED, no qual quando pressionado, fazia a leitura do potenciómetro, e assim que acabava de fazer a última leitura, o LED tinha de piscar 3 vezes.

Para isso foram criados duas funções, a primeira função chamada de “valorBotao()” no qual lê um valor analógico de 1023. De seguida foi feita a implementação da função “estadoBotao()” no qual deteta se o botão atinge o valor analógico máximo( ou quando o botão for premido, neste caso a HIGH), sendo essa igualdade verdadeira, retorna-se true o valor dessa condição.

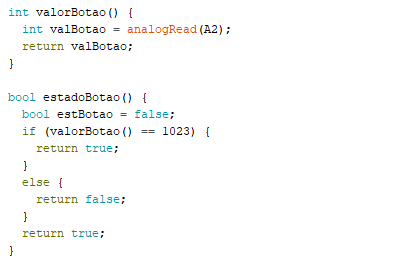


Figura 7 - implementação para a leitura dos valores analógicos do botão

Com o registo do botão feito, passou-se para a implementação da função “funcaoBotao” no qual é esta a função que vai permitir que sejam imprimidos os valores na consola dos 5 registos do potenciómetro. Para isso recorreu-se a um “while” no qual, quando deteta se o estado do botão for false (no momento em que é premido) ele passa a efetuar a leitura das diferentes posições



Figura 8 - implementação da função "funcaoBotao()" para ler os registos do potenciómetro e fazer com que o LED pisque 3 vezes.

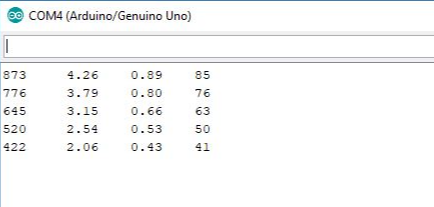


Figura 9 - exemplo de output gerado com as diferentes posições do potenciómetro

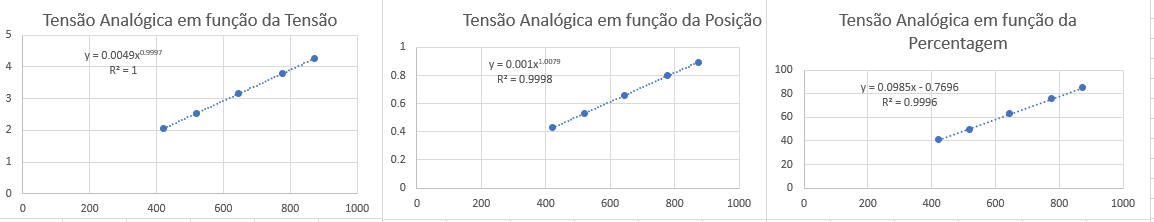
Com os resultados gerados, passou-se para a representação do gráfico no excel:

Figure 10– gráficos em relação aos valores do potenciómetro

Como o potenciómetro é um componente que roda entre um valor máximo e um valor mínimo (entre 0 e 1023), ele tinha de apresentar uma reta de uma função afim, no qual passa na origem do referencial, pelo que concluímos que a representação dos gráficos está correta.

* Bloco 3 - LDR:

Neste bloco, vai se recorrer ao estudo do LDR, sendo que as realizações de algumas funções são idênticas às do potenciómetro.

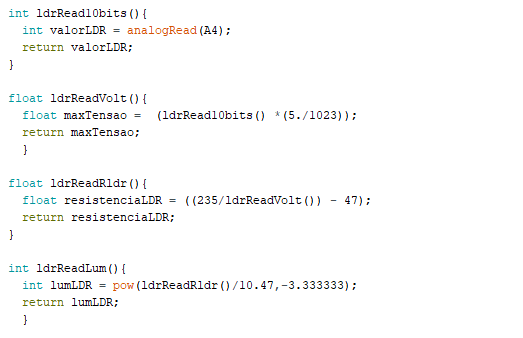


Figura 11 - implementação das funções para a realização do LDR

A maneira de fazer as funções “ldrRead10bits()” e “ldrReadVolt()” são idênticas de como foi feita para o potenciómetro. Para a realização do método “ldrReadRldr()” recorreu-se à formula fornecida no enunciado e usada no primeiro bloco para achar o coeficiente “a”, para descobrir a resistência associada ao ldr, temos que:

Rldr = 47 \* , o valor de VA0 corresponde à tensão máxima do ldr, a implementação desta expressão pode ser verificada na figura9.

Para a realização do método “ldrReadLum()” mais uma vez, recorreu-se à expressão fornecida no enunciado:

Y = 10.47 \* Y/10.47 = = lum = lum

A apresentação deste cálculo pode ser verificada na figura9.

De seguida criou-se a função “ldrDisp()” no qual apresenta os resultados dos diferentes valores do ldr:

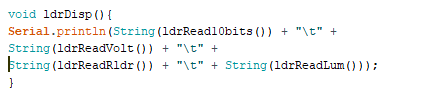


Figura 12 - implementação da função "ldrDisp()"

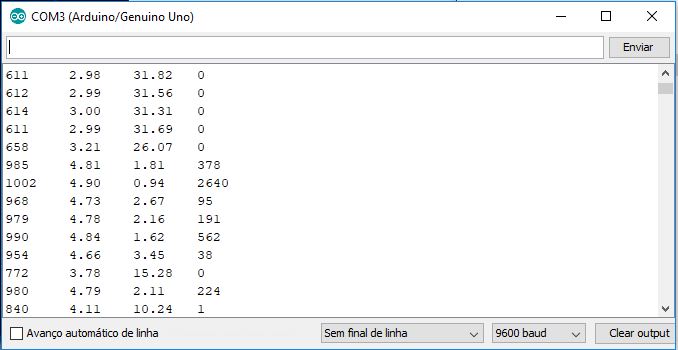


Figura 13 - exemplo de um output com o ldr

Com os resultados, passou-se para a representação do gráfico no excel:

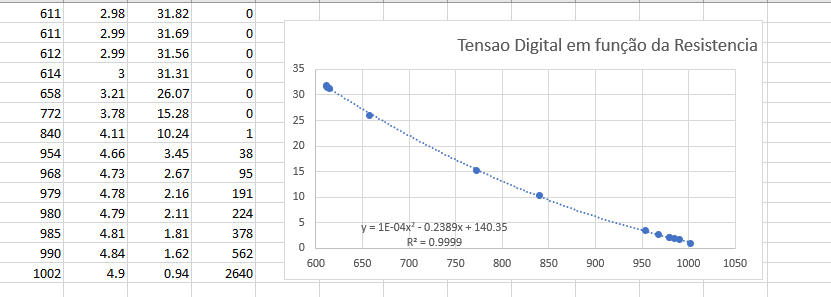


Figura 14 - gráfico dos valores gerados no ldr

Como o ldr é um componente que apresenta uma luminosidade mínima e máxima, a sua resistência também varia entre um valor mínimo e máximo, sendo que quando a luminosidade apresenta um valor baixo, o ldr apresenta uma resistência elevada, mas quando apresenta uma luminosidade elevada, apresenta uma resistência mais baixa, visto que podemos chegar a uma conclusão que o ldr apresenta uma função logarítmica, e através desta representação, podemos concluir que o gráfico gerado está correto,

Depois da implementação da função para mostrar os diferentes resultados, foi adicionado um piezo. Nesta alínea pretendia-se que fosse usada e ouvida duas frequências distintas, para isso recorreu-se à função do Arduíno “tone” no qual consiste em receber uma certa frequência (em Hz) e uma determinada duração (em milissegundos)

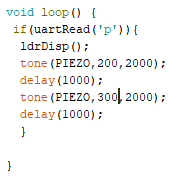


Figure 15 - duas frequências distintas usadas para ouvir a frequência do piezo

* Bloco 3 – analógico-linear e analógico-naolinear:

Através das experiências feitas neste trabalho, podemos concluir que o potenciómetro é um sensor analógico-linear, pois é um componente no qual varia entre um valor máximo e mínimo, sendo que a sua representação gráfica e uma reta de uma equação linear que passa na origem do referencial. Já o LDR é analógico-naolinear pois o seu gráfico apresenta uma função logarítmica, no qual quando a sua luminosidade é muito pequena, apresenta uma resistência maior, mas se apresentar uma luminosidade máxima (neste caso 100%) vai apresentar uma resistência mais pequena.

* Grupo B
* Alínea a)

Para esta alínea foi pedido a um dos alunos do grupo que utilizando os conhecimentos adquiridos na realização dos outros exercícios, que criássemos funções te modo a fazer com que ao carregar no botão o Led acende-se e apaga-se, inversamente, à luz detetada pelo LDR.

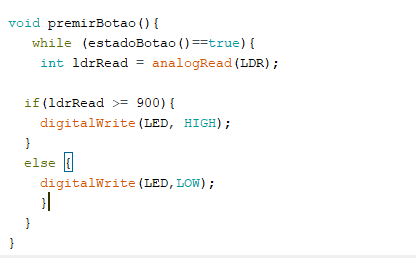
Foi então implementado o seguinte código:

Figura 16 e 17 - código da alínea a) do grupo B

Obtemos assim o resultado desejado: O circuito só funciona se carregarmos no botão. Enquanto o botão está premido o Arduíno deteta os valores do LDR e consequentemente se este estiver destapado o LED está apagado e se o taparmos o LED acende.

* Alínea b)

Nesta alínea era pedido que associa-se os componentes: potenciómetro e piezo(para esta experiência foi usada um buzzer, que tem um comportamento idêntico a um piezo), no qual à medida que se rodasse o potenciómetro, controlava-se o som que saía do buzzer. Para isso foi criada a função “diferentesFrequenciasPiezo()” . Dentro dessa função, foi criada a variável “a” no qual iguala-se à função usada no primeiro exercício de nome “potReadpos()”. Usando um “switch” no qual lê a nossa variável,entre o valor de 0 e 100. De seguida definiu-se 5 classes, nas quais sempre que o potenciómetro é rodado, gera uma frequência de 1000Hz, entre o intervalo escolhido em ambas as classes.

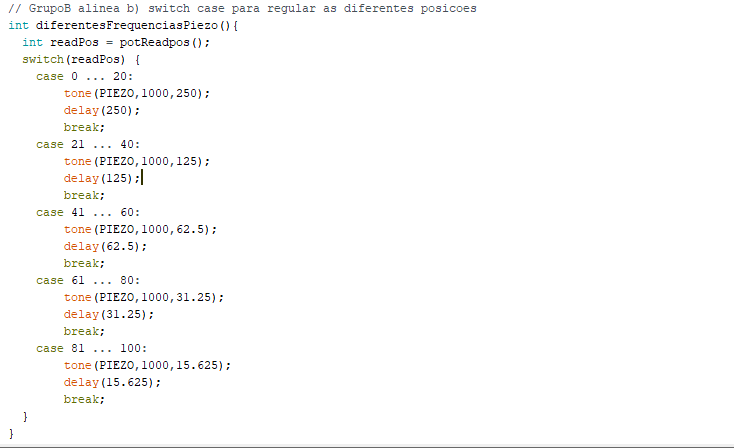


Figura 18 - código para a alínea b) do grupoB

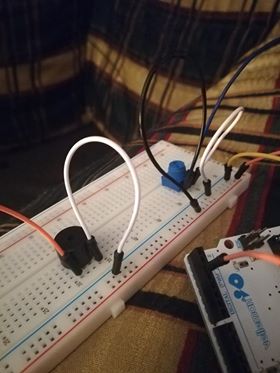


Figure 19 - montagem do circuito, buzzer com potenciómetro

* Alínea c)

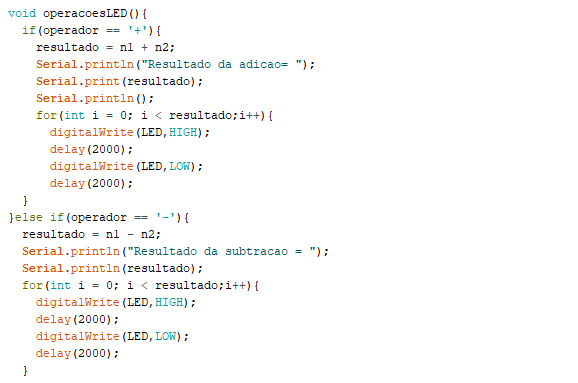
Para esta alínea era pedido que fosse usado um LED, e sempre que uma operação matemática fosse inserida (adição, subtração, divisão e multiplicação) o LED piscava consoante o resultado final de cada uma das operações matemáticas inseridas.

Figure 20 - código da alinea c) grupoB

**Conclusão**

Com este trabalho prático, o grupo consegui-o ter uma forte consolidação de como é que a linguagem C++ é usada em sensores, e os diferentes métodos para obter os resultados. Também aprendeu como funciona o comportamento em estudo do potenciómetro e do LDR, no qual, ambos apresentam valores das resistências mínimas e máximas e as suas propriedades associadas.

**Bibliografia**

[SA2018] sensoresarduino, pp.1-38, Eng Manfred Niegus, Eng.Carlos Carvalho