Sistema de Monitorização Baseado em Arduíno

Vasco Ventura 1700299, João Aires 1700318

Data: 23/junho/2019

Arquitetura e Sistemas de Computadores

Docente: Gonçalo Marques

Resumo

Este artigo baseia-se num projeto de um sistema de monitorização de duas grandezas meteorológicas, a humidade e a temperatura. Tais leituras que podem ser feitas através de apenas um sensor com um comprimento de aproximadamente 4cm, tornando viável construirmos uma estação meteorológica de pequenas dimensões utilizando poucos componentes.

O que tornou isto possível foi a ligação do sensor a um microcontrolador "Arduíno UNO".

O projeto foi desenvolvido com um sistema de alarme e por isso, este artigo contém exemplos de como conseguimos manipular variáveis (através de um "*LED"* e de um botão), consoante as condições de temperatura/ humidade.

Por fim foi exposto no artigo, a temática da "Internet das Coisas". O principal fator que possibilita este paradigma é a integração de várias tecnologias e soluções de comunicação, tecnologias de identificação e rastreamento, redes de sensores, etc. Como se pode facilmente imaginar, qualquer contribuição séria para o avanço da Internet das Coisas deve necessariamente ser o resultado de atividades conduzidas em diferentes campos do conhecimento, como telecomunicações, informática, eletrónica, ciências sociais…

Introdução

O Arduíno é uma plataforma "*Open-Source*" utilizada para sistemas eletrónicos oferecendo a possibilidade de fazer simples projetos. Foi utilizado neste projeto, pela sua versatilidade através do uso de uma versão da linguagem C++. Tem ainda a vantagem de um utilizador poder programar, apagar e reprogramar sempre que assim o desejar, além de que o Arduíno é um recurso de baixo custo. Através do sensor "DHT11" vão ser recolhidos os dados. Este sensor possui várias vantagens, como pequeno porte, interface simples, resposta rápida e baixo custo. Este artigo apresenta brevemente princípios básicos e métodos de aplicação do DHT11.

As condições climatéricas são um fenómeno simples de programar, tendo acesso a muita informação hoje em dia. Não há dúvida que é um fator que influencia bastante o quotidiano da sociedade quando se fala de condições para a construção civil, estratégias militares, agricultura, etc.

O objetivo do projeto é construir um sistema equivalente a uma estação meteorológica, onde conseguimos ler a temperatura e humidade, sendo essa informação guardada e partilhada posteriormente. É importante referir que, um dos objetivos do projeto é o uso de "*Interrupts*", para utilizar o sistema de leitura do sensor e o desligar e ligar o sistema alarme em simultâneo ("*Multitasking*").

Depois do sensor ter as informações da temperatura e humidade de um determinado espaço os dados podem ser transferidos para serem processados, através de um "*script Python"*, por exemplo com o fim de serem registados "*online"* num canal "*ThingSpeak*". Isso poderá tornar praticável o acesso aos dados mais fácil, por meio de diversos dispositivos. O sistema de monitorização do tempo será atualizado de hora-a-hora.

Palavras-Chave: Temperatura, Humidade, Sistema de Monitorização, Arduino, Python, DHT11, LED, Interrupt.

Trabalhos Relacionados

1. https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

Proposta de trabalho

O sistema proposto utiliza um sensor (DHT11) que permite medir os fatores ambientais de temperatura e humidade, como já referido. Os valores lidos pelos sensores são processados pelo microcontrolador Arduino. Em conjunto, pretende-se construir um mecanismo que informe o seu utilizador de que quando o sensor reconhecer que o ambiente atingiu uma certa temperatura (25ºC, neste caso). O sistema de alarme foi implementado com um "Led". O sistema de alarme pode ser ativado ou desativado através de um botão. Através da plataforma ThingSpeak, vão-se automaticamente gerar gráficos de leitura em que os campos vão corresponder às leituras do clima.

Material

- Microcontrolador Arduino;
- Sensor DHT11;
- Led;
- Botão;
- Resistências (1500 Ω);
- Protoboard;
- Cabos de ligação;
- *Software "Arduino" e "Python".*

Arquitetura do sistema

Figura 1 - Arquitetura do Sistema em Estudo

Projeto do Circuito

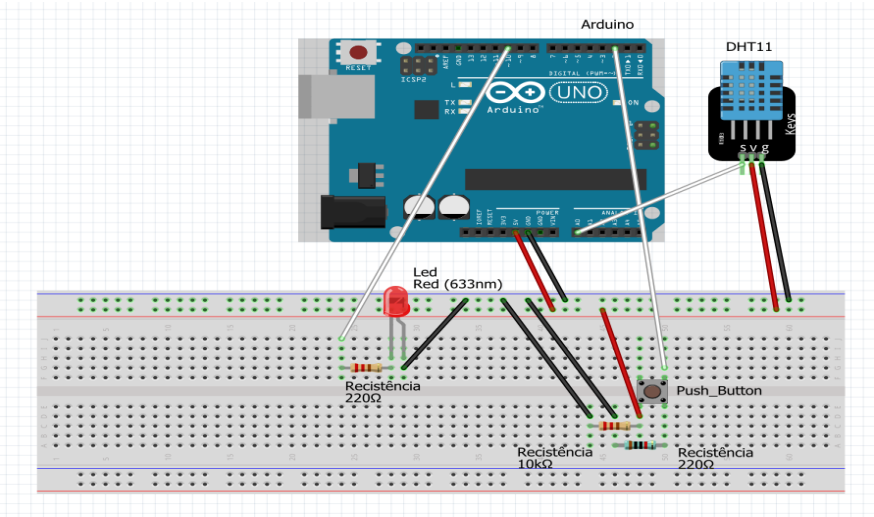


Figura 2 - Projeto do Circuito com Arduíno e restantes componentes

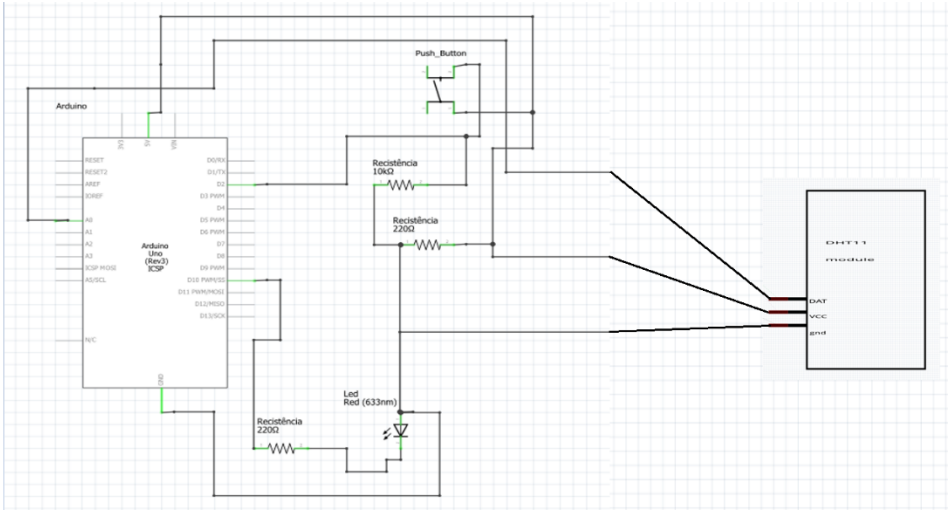


Figura 3 - Esquema Elétrico do Circuito

Procedimentos

Os procedimentos estão divididos por partes. Em ambas as partes estão apresentados de forma explícita os scripts. Na parte I encontra-se o código para compilarmos o nosso programa, ou seja, dividimos o código em algumas partes para desta forma ser mais fácil explicar o que o mesmo executa . Já na parte II temos a execução do código em Python para que seja possível enviar os dados para um servidor online.

Parte I - Arduino

1. Bibliotecas e programa:

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

2. Mapeamento do Hardware

```
#define    dht_pin    A0
int interruptPin = 2;
int ledPin = 10;
```

O sensor está conectado à porta analógica A0, o botão está ligado à porta digital 2, pois esta dispõe a possibilidade de trabalharmos com Interrupt logo, é possível de fazer uma tarefa enquanto uma está a ser executada (conceito de "*Multitasking*"). Neste caso, enquanto estamos a receber leituras do sensor e ao mesmo tempo estamos a controlar o sistema de alarme do "*kit*".

3. Declaração de objetos

```
dht    my_dht;
```

Chamámos ao sensor de "my_dht".

4. Função "setup()"

```
void setup(){

   Serial.begin(9600);

   pinMode(ledPin,OUTPUT);
   pinMode(interruptPin,INPUT);
   attachInterrupt(digitalPinToInterrupt(interruptPin), botao, RISING);

  startMillis = millis();

}
```

Na função "setup" configuramos o tipo de uso que vamos dar aos nossos componentes, se são componentes de entrada ou saída e é também onde configuramos o interrupt juntamente com a função que vamos recorrer quando o botão é pressionado.

Por último a variável "*startMillis*" vai ser um contador de tempo que nos retorna o tempo que já desde o início da aplicação.

5. Função “loop()”

```
void loop() {
  currentMillis = millis();
  if ((currentMillis - startMillis) == period){
    period = 3600000;
    ler();
    startMillis = currentMillis;


  }
```

Anteriormente, na declaração das variáveis globais notamos que a variável “*period*” tem um valor de 5000 ms. Isto significa que a primeira leitura vai ser feita após cinco segundos do programa ter iniciado, dando tempo ao hardware para iniciar. Depois da primeira leitura, com a alteração do período, o programa só irá ter uma nova leitura ao fim de uma hora.

```
   if(button == 0){
     if(temperatura >= 25){
       alerta();
     }else{
       apagaLed();
     }
   }else{
     apagaLed();
   }
}
```

Na função “*loop()*” criámos condições para que ligar e desligar o sistema de alarme (led). O alarme somente vai ser acionado se o estado do botão for “*true*” e se a temperatura do ambiente for superior a 25ºC.

É importante referir que não utilizámos a função “*delay()*” para que uso de interrupt ao pressionarmos o botão e a contínua leitura do sensor sejam capazes de executar em simultâneo.

6. Função de leitura do sensor (função ler())

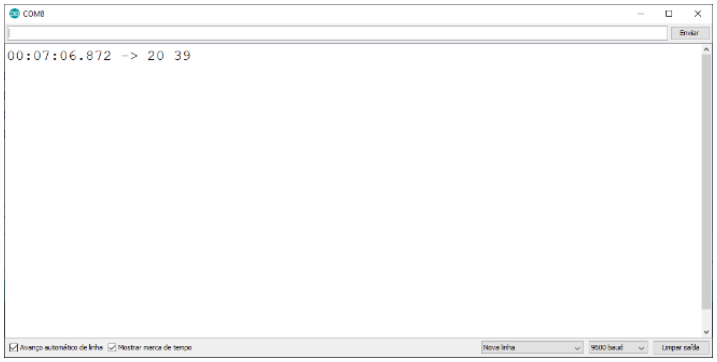
```
void ler(){

   my_dht.read11(dht_pin);

   temperatura = my_dht.temperature;
   humidade    = my_dht.humidity;


  somaTemperatura = 0;
  somaHumidade = 0;
```

Na função ler() as variáveis de declaradas correspondem a cada leitura de temperatura e humidade feita pelo sensor.

```
for(int i = 0; i < 5; i++){
  my_dht.read11(dht_pin);

  temperatura = my_dht.temperature;
  humidade    = my_dht.humidity;


  somaTemperatura = somaTemperatura + temperatura;
  somaHumidade = somaHumidade + humidade;

  temperaturaFinal = somaTemperatura / 5;
  humidadeFinal = somaHumidade / 5;
  }
```

Para ter leituras exatas do sensor, foi introduzido um ciclo, onde uma leitura final é retornada após cinco leituras do sensor através do cálculo da média para essas 5 leituras. Dessa forma não há discrepâncias nos valores referidos. Os dados recolhidos são lidos pelo computador através de uma porta COM, e só posteriormente, serão registados de hora-a-hora num servidor. Mais à frente estarão os exemplos dos gráficos em função do tempo para os dados recolhidos que servirão como consulta dos resultados.

```
   Serial.print(temperaturaFinal);
   Serial.print(" ");
   Serial.print(humidadeFinal);
   Serial.print("\n");

}
```



Por fim da função, eis os dados que aparecerão na consola:

7. Função “botao()”

```
void botao(){
  if(button){
    button = false;
  }else{
    button=true;

  }
}
```

A função botao() tem como fim alterar o estado do alarme, ligando (button = true, estado inicial) ou desligando (button = false) o respetivo.

O formato que aparecerá na consola é relevante para o script Python, como veremos “a posteriori”.

Parte II – Python

```
import http.client
import serial
import array as arr

ser = serial.Serial(
    port='COM8',
    baudrate=9600
)


temperatura = ''
humidade = ''

while True:
    ch = ser.readline()

    sensor = str(ch.decode())

    result = sensor.split(' ')



    temperatura = result[0]
    humidade =  result[1]

    print ("Temperatura: " + temperatura +  " " + "Humidade: " + humidade)

    conn = http.client.HTTPConnection('184.106.153.149')
    conn.request("GET","/update?api_key=GI627UOI6O5TDE6H&field1=" + temperatura + "&field2=" + humidade)
    conn.close()

    temperatura = ''
    humidade = ''

    ch = ''
```

O Script Python vai pegar nos valores examinados pelo sensor que lhe são comunicados pela porta COM e vai dividir (através do "*split*") dividir os dados de temperatura e humidade (daí ser relevante a forma como imprimimos os resultados através do Arduino). Através de um "*request*" os dados vão ser atualizados em tempo-real. O canal do ThingSpeak possui uma chave ("*API key*") na qual a request vai utilizar a chave do canal como destino dos dados.

Resultados experimentais

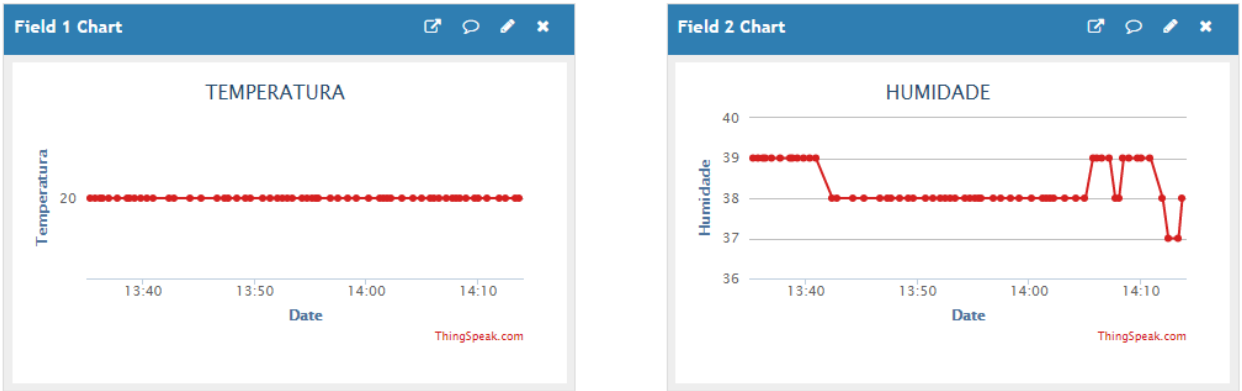


Figura 4 - Gráficos dos resultados no TeamSpeak

Como podemos ver o procedimento funcionou. No entanto o nosso intervalo de tempo é relativamente pequeno e constante porque o mesmo serve apenas para estar dentro de um ambiente fechado, ou seja, nem a temperatura nem a humidade vão ser muito afetadas no dia-a-dia. Concluindo estes resultados são apenas uma amostra do que o nosso programa irá fazer no futuro.

Todos os módulos foram projetados e todos os componentes foram montados. A análise de cada módulo foi realizada com sucesso. As leituras dos sensores foram efetivamente efetuadas num ambiente estável. Há uma necessidade de realização de realizar novas experiências em ambientes mais semelhantes às condições meteorológicas reais.

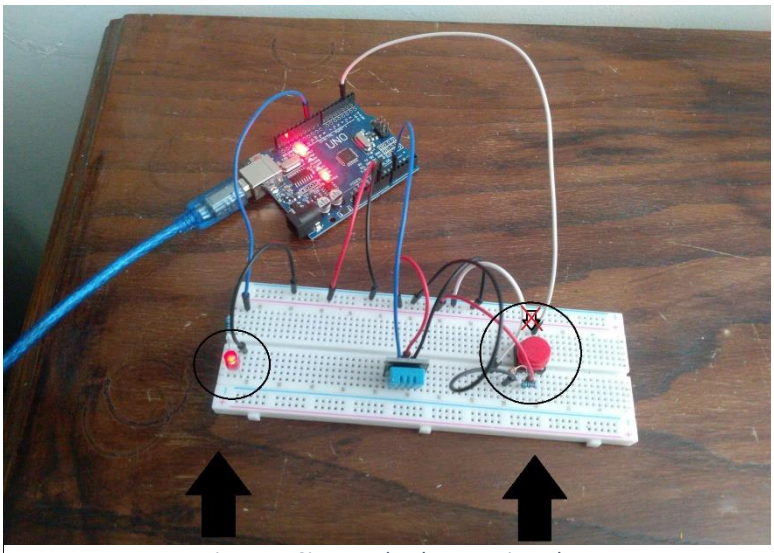


Figura 5- Sistema de Alarme Acionado

Quanto ao sistema de alarme podemos verificar na figura – que o alarme está ligado. Quer isto dizer que a temperatura do ambiente é superior a 25ºC . Por fim, este só será desligado quando o utilizador pressionar o botão.

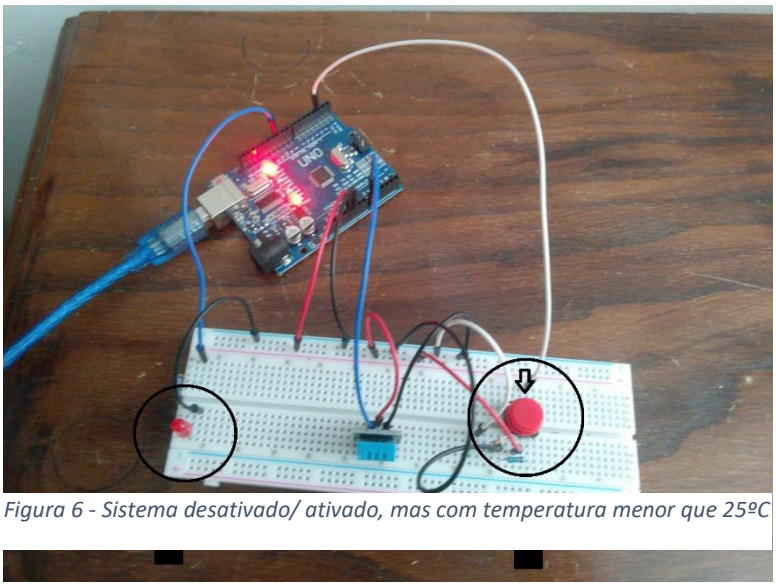


Figura 6 - Sistema desativado/ ativado, mas com temperatura menor que 25ºC

O sistema de alarme apenas não será acionado, quando o utilizador assim o desejar ou quando a temperatura do espaço é menor que 25ºC

Conclusão

Concluímos que o presente trabalho foi um sucesso e fornece um método competente para a gravação em tempo real de leituras meteorológicas e pode futuramente servir para um projeto de controlo de temperatura de um espaço de uma casa ou até mesmo servir como protótipo de um produto comercializado, por exemplo.

É importante referir o uso da Internet, através do conceito de “A Internet das coisas” (IoT), pela compreensão dos aparelhos e objetos que se encontram ligados à Internet, sendo capazes de se identificar numa rede, habilitando um largo acesso a dados em qualquer parte do globo.

Referências

https://www.filipeflop.com/blog/estacao-meteorologica-com-arduino/

https://www.instructables.com/id/Arduino-Software-debouncing-in-interrupt-function/