Internet das Coisas

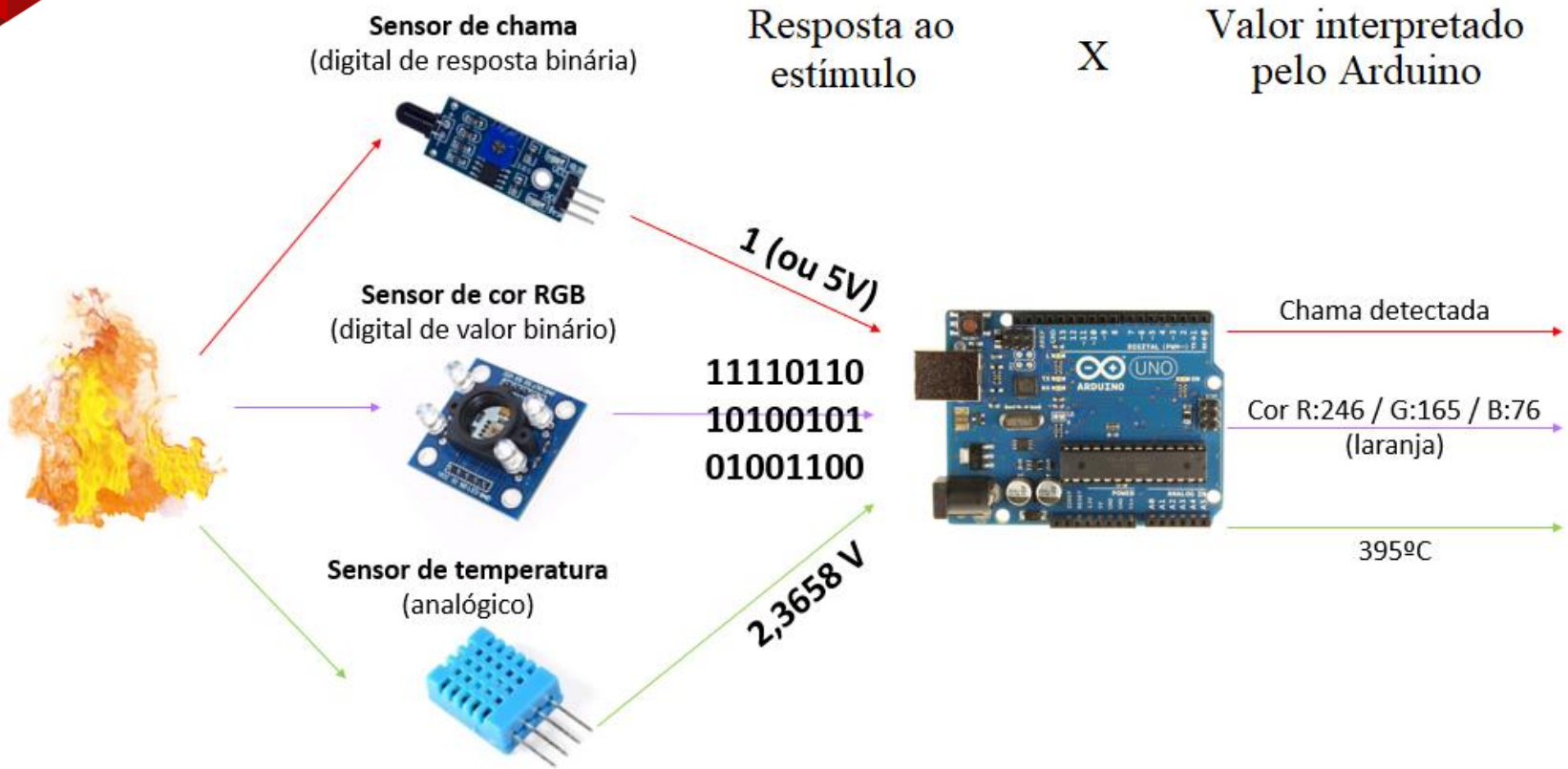
Aula 12 –Lendo sensores de resposta binária

Prof. Rodrigo Barreto rodrigo.barreto@qi.edu.br / Fone/Whats: (51) 98412-5651

Leitura das informações de sensores

Os sensores do Arduino podem ser de 2 tipos distintos: digitais e analógicos. Os sensores analógicos conseguem devolver ao Arduino qualquer tensão entre 0V e 5V como resposta ao estímulo do fenômeno medido (e esse valor, como veremos porteriormente, é mapeado para um número entre 0 e 1023) e os sensores digitais apenas conseguem devolver ao Arduino 0V ou 5V (estado baixo e alto, 0 ou 1 binário). Assim, estes sensores digitais podem ser sensores de resposta binária (aqueles que devolvem apenas duas respostas como sim ou não, detectado ou não etc) ou sensores que utilizam combinações dos estados baixo e alto para o envio de bits 0 ou 1 em uma sequência binária que posteriormente é transformada em valor.

Leitura das informações de sensores



Leitura das informações de sensores

Os três sensores são de propósitos diferentes: o de cima é um sensor que informa a detecção de chama, o do meio é um sensor que informa o valor RGB da cor detectada e o de baixo é um sensor que mede a temperatura. Apesar de sujeitos ao mesmo estímulo (uma mesma chama), cada um enviará ao Arduino um valor diferente, de uma forma diferente e que representa uma informação diferente.

Leitura das informações de sensores

O sensor de chama é um sensor de resposta binária e ao detectar fogo envia ao Arduino 5V (equivalente ao estado alto, ou 1 binário);

5V (HIGH ou 1 binário)

Leitura das informações de sensores

Já o sensor de cor RGB é um sensor de valor binário e alterna seus estados altos e baixos para enviar ao Arduino, de modo binário, os valores de composição das 3 matizes que compõe a cor (no caso 246 de vermelho, 165 de verde e 76 de azul, resultando em um tom de laranja).

11110110 10100101 01001100
(R: 246 / G:165 / B: 76)

Leitura das informações de sensores

O sensor de temperatura varia a sua tensão de acordo com a temperatura medida e envia ao Arduino 2,3658 volts, que serão convertidos em um valor entre 0 e 1023 (adiante entenderemos isso) que finalmente é mapeado para a temperatura medida (no caso, 395ºC).

2,3658 V
(mapeado para 687 que nesse sensor representa 395ºC)

Leitura das informações de sensores

Esse exemplo ilustra que para podermos trabalhar facilmente com sensores no Arduino, necessitamos entender não só como os diferentes tipos de sensores enviam os dados gerados em resposta aos estímulos mas, principalmente, como o Arduino deve lê-los e interpretá-los. Veremos a partir de agora, como trabalhar com os sensores de **resposta binária**, os sensores de **valor binário** e os sensores **analógicos**.

Sensores de resposta binária

Uma parte considerável dos sensores que utilizamos em projetos eletrônicos são digitais de resposta binária como vimos (ou seja, nos retornam apenas estado baixo ou estado alto).

Lendo sensores de resposta binária

Para a leitura de sensores digitais de resposta binária, utilizamos o método digitalRead, informando sua porta e tendo como resposta um estado baixo (0 / LOW) ou um estado alto (1 / HIGH).

```
digitalRead( <porta> );

<porta>
porta à qual o sensor de resposta binário está ligado

resposta:
a resposta do método será o estado LOW (0) ou HIGH (1)
```

Lendo sensores de resposta binária

Para todos os sensores de resposta binária, o código praticamente não tem alteração. Os mais comuns são:

sensor de som

sensor de proximidade (PIR)

sensor de inclinação (TILT)

botão de toque (chave tátil)

push button

chave de 2 estados

sensor magnético de proximidade (fecho magnético)

chave magnética (reed switch)

chave de fim de curso

sensor de chama

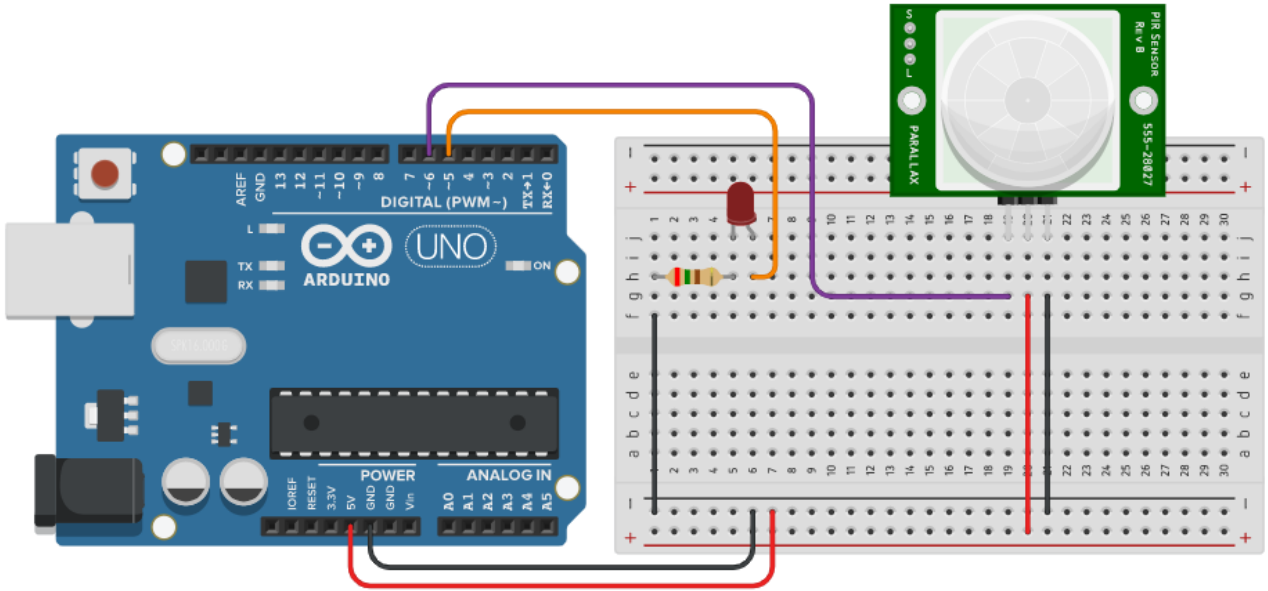
sensor de vibração

Lendo sensores de resposta binária

Exemplo básico: queremos que um sensor de presença do tipo PIR, ao detectar um objeto em movimento próximo a si, ligue um led como indicativo. Os sensores digitais de resposta binária geralmente possuem 3 pinos: GND, Vin (ou ++ ou 5V) e um pino de envio de sinal, que deve ser conectado em uma porta digital qualquer. No caso do PIR, o pino 1 é este pino de sinal (por onde ele nos dará a resposta binária da sua detecção, no caso, estado baixo se não detectar nada ou estado alto caso detectar), enquanto o pino 2 (central) é o 5V e o pino 3 é o GND. Em nosso exemplo, o pino de sinal será ligado na porta 6, enquanto o positivo led (como o negativo em paralelo com um resistor de 250Ω) será ligado à porta 5.

Lendo sensores de resposta binária

Teremos então um circuito eletrônico como demonstrado no esquema abaixo:



Lendo sensores de resposta binária

Com a ligação eletrônica concluída, devemos programar o Arduino para que quando o PIR detectar movimento em seu entorno, o led indicativo ser aceso. Previamente, devemos informar que a porta de sinal do PIR (6) é uma porta de entrada (INPUT), enquanto a porta do led (5) é uma porta de saída (OUTPUT). Já no loop realizamos a leitura digital da porta de sinal para detectar se o mesmo está baixo (neste caso desligamos o led) ou alto (neste caso, ligamos o led). Assim, nosso código será como o apresentado a seguir:

Lendo sensores de resposta binária

```
void setup(){
  pinMode(5, OUTPUT);
  pinMode(6, INPUT);
  digitalWrite(5, LOW);
}

void loop(){
  delay(1000);
  if(digitalRead(6) == HIGH) {
          digitalWrite(5, HIGH);
  } else {
    digitalWrite(5, LOW);
  }
}
```

Lendo sensores de resposta binária

Importante novamente ressaltar, que todos os sensores de resposta binária, utilizam exatamente a mesma lógica. Então se o PIR fosse substituído por um sensor de chama, ou um sensor de som, ou qualquer outro do mesmo tipo, mantida a porta 6 no pino de sinal, o código para a leitura do estado seria exatamente o mesmo, sem qualquer mudança. Ou seja, com um método (digitalRead) já aprendemos a ler os estados de 11 sensores diferentes \o/

Lendo sensores de resposta binária

Observação: quando utilizamos sensores digitais de resposta binária, temos a opção de declarar o mesmo como **INPUT_PULLUP**. Essa é um tipo de entrada especial que ativa um resistor de pullup presente em cada porta digital do Arduino. Não cabe aqui explicar tecnicamente como ele funciona, mas sim o seu efeito: ele elimina estados secundários entre a tensão mínimo (0V) e a tensão máxima (5V), principalmente em sensores que possuem elementos mecânicos que ao serem movimentados podem estar em um estado onde não há total contato, provocando variações breves na tensão que podem ser entendidas erroneamente. Isso é especialmente útil ao usar botões.