

STEMKIT
4SCHOOLS

Utilização de Sensores Infravermelhos em Sistemas de Alarme

Plano de Aula 1



Cofinanciado pelo
Programa Erasmus+
da União Europeia

Este projeto é cofinanciado pelo Programa Erasmus + da União Europeia.

Este projeto foi financiado com o apoio da Comissão Europeia. Esta publicação reflete apenas as opiniões dos autores, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação aqui contida.

Índice

1. Utilização de sensores infravermelhos em sistemas de alarme.....	2
1.1 Informação geral.....	2
1.1.1 Breve descrição	2
1.1.2 Objetivos de aprendizagem.....	2
1.1.3 Ligação com o currículo	3
1.1.4 Materiais necessários	3
1.1.5 Duração	3
1.2 Plano de aula.....	4
1.2.1 Introdução.....	4
1.2.2 Preparação	4
1.2.3 Investigação.....	9
1.2.4 Conclusão.....	10
1.2.5 Exercício de Follow-up (opcional)	10
1.3 Referências ou Recursos.....	11



1. Utilização de sensores infravermelhos em sistemas de alarme

1.1 Informação geral

1.1.1 Breve descrição

Esta lição introduz a utilização do sensor ótico refletor TCRT5000 com saída de transístor para conceber um circuito simples que funcionará como alarme para janelas que possam ter sido abertas por uma pessoa não autorizada. O posicionamento de tal sensor pode ser introduzido em configuração real junto à moldura da janela pois o alcance de funcionamento do sensor é de 0,2 mm a 15 mm.

1.1.2 Objetivos de aprendizagem

Os principais objetivos de aprendizagem deste plano de aula/atividade educativa são:

- Compreensão do conceito e conteúdo do hardware para estabelecer um código em Scratch.
- Familiarização com as atividades de construção de consolas para melhorar a experimentação em temas relacionados com as STEM (ciência, tecnologia, engenharia e matemática).
- Familiarização com leituras de pinos GPIO.
- Compreender as etapas de codificação no Scratch.
- Conceber a codificação no Scratch.
- Utilização de sensores, emissor de infravermelhos, foto transístor e outros elementos para experimentar com a consola STEMKIT.
- Executar a codificação na consola STEMKIT com o caso prático de sensores infravermelhos em sistemas de alarme.
- Realizar a montagem de sensores numa breadboard.
- Experimentar a ligação de sensores à consola Raspberry Pi e STEMKIT.
- Autonomia na criação de um circuito simples que pode servir como demonstração de uma instalação de segurança num edifício.
- Autonomia na introdução dos conceitos de codificação no ambiente da sala de aula.



1.1.3 Ligação com o currículo

Os domínios, subdomínios, assuntos/tópicos a que este plano de aula pode ser ligado são:

Ciência (Física/Química/Biologia/Geologia): tensão, potência, circuitos, disparadores de alarme, método científico, investigação, experimentação, análise e interpretação dos resultados

Ciência de Computação/Informática: unidade de processamento e periféricos, interfaces, linguagem de programação e estruturas principais, codificação

Tecnologia: eletrónica, hardware e software de código aberto, sensores, sinal digital, computadores de placa única, consola.

1.1.4 Materiais necessários

Para realizar este plano de aula, é necessária a consola STEMKIT com Raspberry Pi, juntamente com os seguintes elementos:

3 x sensores TCRT5000

1 x campainha com gerador

2 x Fios jumper fêmea a fêmea

5 x fios jumper macho para fêmea

3 x resistências 10k Ω

3 x resistências 330 Ω

1 x breadboard

1.1.5 Duração

A duração deste plano de aula é estimada em cerca de 45-60 minutos, ou seja, uma hora de aula.

1.2 Plano de aula

O plano de aula está dividido em quatro fases, que são introdução, preparação, investigação e conclusão. Como seguimento, há também um exercício opcional no final.

1.2.1 Introdução

O TCRT5000 é um sensor composto por dois elementos num único invólucro. O emissor emite o comprimento de onda de 950 nm que deve então ser recebido pelo fototransístor que trabalha como detetor. Quanto mais próximo estiver o objeto do sensor, maior será a leitura da voltagem do fototransístor. Como o Raspberry Pi oferece pinos GPIO, esta leitura pode então ser seguida para verificar se o obstáculo (a moldura da janela) está perto do sensor e, como resultado, se a janela está fechada.

Dentro desta aula, o Scratch será utilizado para demonstrar o código de exemplo que pode ser utilizado para controlar este circuito simples.

1.2.2 Preparação

A fase de preparação requer a montagem básica dos sensores numa breadboard e a configuração do código em Scratch. Começemos primeiro com a breadboard.

Colocar três sensores TCRT5000 sobre uma breadboard vazia, ligando o emissor do fototransístor e o cátodo do emissor de infravermelhos ao carril de terra. Depois disso, ligar o coletor do fototransístor usando uma resistência 10k Ω à calha positiva da breadboard. Da mesma forma, ligar o ânodo do emissor de infravermelhos ao trilho positivo usando uma resistência 330 Ω . Repita para os dois restantes sensores TCRT5000.

Agora é altura de ligar os sensores ao Raspberry Pi. Ligar um fio jumper de comprimento apropriado a cada sensor TCRT5000. Deve ligar-se ao carril onde o coletor do fototransístor é ligado e alimentado por um carril de voltagem positiva utilizando uma resistência 10k Ω . Nesta fase deve haver três fios jumper - um por cada sensor TCRT5000. Estes fios jumper devem ser ligados aos pinos GPIO do Raspberry Pi marcados como 35, 33 e 31 (ou em GPIO: 19, 13 e 6). Agora é também o momento de ligar a campainha diretamente ao Raspberry Pi. Pode fazê-lo ligando o pino GND da campainha ao pino 39 e o seu fio positivo ao pino 38 (ou em GPIO: 26). Finalmente, a breadboard precisa de receber a energia do Raspberry Pi. Para este efeito, pode usar o carril +5V da Raspberry Pi (pino 4) e GND (pino 6). Utilizar os fios jumper para alimentar a breadboard. A configuração do hardware está feita, por isso, agora podemos passar ao Scratch.



Dentro do Scratch, pode seleccionar qualquer pano de fundo que tenha pelo menos três janelas visíveis. Para esta lição vamos utilizar o pano de fundo urbano1. Configure-o para toda a cena. Ao mesmo tempo, é necessário acrescentar três objetos que mudarão de roupa assim que o alarme for acionado. Aqui a sugestão é utilizar um sprite que combine botão4-a e botão5-b. O código mudará os trajes com base na leitura dos sensores. Como é necessário ter três sprites (um por cada janela), por favor duplique-os para ter a seguinte configuração (o sprite do meio tem o outro fato ativo para mostrar a diferença):



Imagem 1. Ambiente Scratch com todos os elementos necessários posicionados no ecrã
Fonte: projeto STEMKIT4Schools

Antes de irmos para o código principal, vamos trabalhar nos três sprites que são colocados nas janelas.

Para simular o alarme, vamos utilizar as mensagens de emissão do Scratch para reagir em conformidade com as leituras dos sensores. Ao passar para o separador Scripts, será necessário adicionar duas reações para as mensagens recebidas. Vamos supor que para o primeiro sensor emitiremos mensagens com *janela1-aberta* e *janela1-fechada*. Quando a mensagem recebida for *janela1-aberta*, então precisamos de acionar o alarme e mudar o fato de uma marca de verificação verde para uma cruz vermelha. Da mesma forma, quando a janela está fechada, precisamos de ter novamente o fato de marca de verificação verde. O código exemplo é apresentado abaixo:



Imagem 2. Roteiro de sprites reagindo às leituras dos pinos GPIO

Fonte: projeto STEMKIT4Schools

Replicar esta configuração para os restantes dois sprites colocados sobre as janelas e lembrar de alterar os títulos das mensagens (*janela2-aberta*, *janela2-fechada*, *janela3-aberta*, *janela3-fechada*).

O código principal é onde tudo se torna interessante. Uma vez selecionado o gato, mude para o separador Scripts e comece a adicionar o código. A primeira coisa que gostaríamos de configurar é informar Scratch que os pinos GPIO 19, 13 e 6 devem ser lidos como pinos de entrada.



Imagem 3. Início do código - definição de pinos GPIO relevantes como pinos de entrada

Fonte: projeto STEMKIT4Schools

A seguir, pode adicionar algumas mensagens antes de lançarmos o código principal.

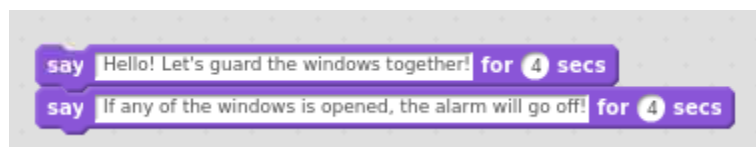


Imagem 4. Mensagens de introdução executadas pelo código

Fonte: projeto STEMKIT4Schools

No passo seguinte, vamos acrescentar um loop de repetição que fará 30 ciclos, entendidos como "verificações" dos sensores.



Imagem 5. Loop de repetição principal
Fonte: projeto STEMKIT4Schools

Dentro de cada loop, estamos interessados em alcançar os três sensores para ver se as janelas estão fechadas ou não. A estrutura é muito simples, pois podemos verificar com uma declaração if/se/condicional, se a leitura dos respectivos pinos GPIO é alta e age em conformidade. Abaixo encontrará um exemplo de pinos GPIO número 19.

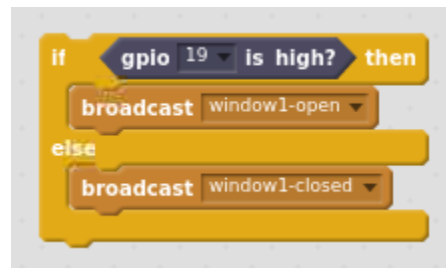


Imagem 6. Uma declaração if/se, lendo o valor GPIO e mensagens de difusão
Fonte: projeto STEMKIT4Schools

Note-se que a única ação que estamos a tomar aqui é emitir ou mensagem de janela1-aberta ou de janela1-fechada com base no valor do pino número 19 da GPIO. Da mesma forma, blocos se/blocos condicionais devem ser adicionados para os restantes GPIO: 13 e 6. Mais uma vez, não se esqueça de alterar o número do pino GPIO e também a mensagem a ser transmitida!

No passo seguinte, também queremos controlar a nossa campainha. A condição aqui é que, se todas as janelas estiverem fechadas, o alarme é silencioso. Quando pelo menos uma janela é aberta, emitimos o som do alarme e também deixamos o gato dizer Alarme! para ter também um aviso visual. Para tal, utilizaremos uma declaração com três condições combinadas por uma lógica ou instrução.



Imagem 7. Uma declaração de if/se/condicional para acionar o alarme, se necessário
 Fonte: projeto STEMKIT4Schools

Pouco antes de completar o ciclo atual, faremos uma pausa de 1 segundo na execução. Ao fazê-lo, definimos a frequência das nossas verificações a serem executadas uma vez por cada segundo.

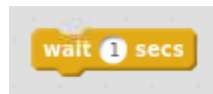


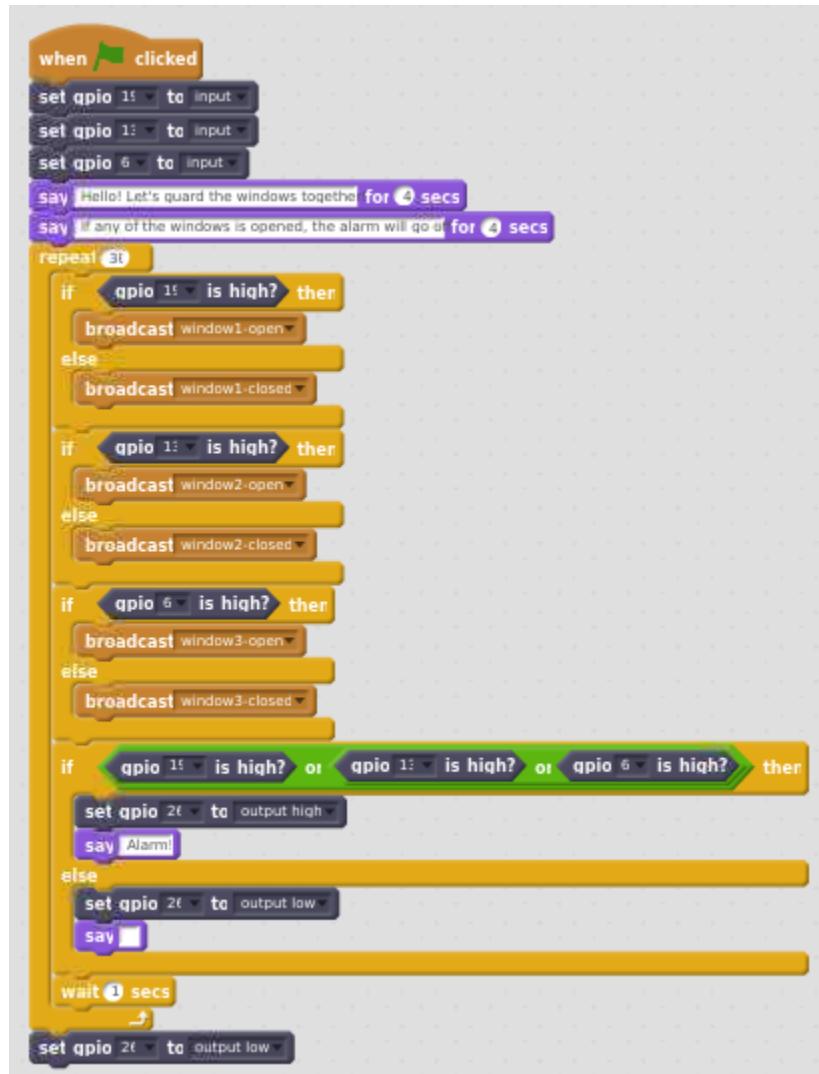
Imagem 8. Pausa de execução do código durante 1 segundo imediatamente antes de deixar o loop
 Fonte: projeto STEMKIT4Schools

Finalmente, como não queremos ter a campainha a emitir som, depois de terminarmos todo o nosso código e logo após a repetição do loop, precisamos de adicionar a instrução para definir o pino GPIO número 26 (aquele com a campainha) para permanecer no estado baixo (o que significa sem som).



Imagem 9. Colocar a campainha GPIO em estado baixo após o laço principal
 Fonte: projeto STEMKIT4Schools

O resultado final que define o nosso código é apresentado abaixo.



```

when green flag clicked
  set qpio 15 to input
  set qpio 13 to input
  set qpio 6 to input
  say Hello! Let's guard the windows together! for 4 secs
  say If any of the windows is opened, the alarm will go off for 4 secs
  repeat (3)
    if qpio 15 is high? then
      broadcast window1-open
    else
      broadcast window1-closed
    if qpio 13 is high? then
      broadcast window2-open
    else
      broadcast window2-closed
    if qpio 6 is high? then
      broadcast window3-open
    else
      broadcast window3-closed
    if qpio 15 is high? or qpio 13 is high? or qpio 6 is high? then
      set qpio 24 to output high
      say Alarm!
    else
      set qpio 24 to output low
      say 
    wait 1 secs
  set qpio 24 to output low
  
```

Imagem 10. Código principal para o circuito de amostras

Fonte: Projeto STEMKIT4Schools

1.2.3 Investigação

Podemos finalmente executar o nosso código! Por favor, siga as próximas subtarefas para saber mais sobre este exemplo. Para evitar leituras falsas, certifique-se de que todos os elementos utilizados para construir o circuito (fios jumper ou resistências) não cobrem nenhum dos sensores TCRT5000.

Recolha de dados

Para a primeira tentativa, certifique-se de que os sensores não estão cobertos, ou, por outras palavras, que não há elementos que reflitam o feixe infravermelho emitido e, portanto, transforme os pinos GPIO em estado elevado. Na tentativa seguinte, tente cobrir um sensor com o dedo ou qualquer outro elemento. Observe o comportamento quando os três sensores estiverem cobertos, simulando a situação em que todas as janelas estão fechadas.

Análise de dados

Com base nos dados observados, é capaz de dizer se a distância do obstáculo ao sensor está de acordo com o intervalo de trabalho esperado? Tente notar quão perto o obstáculo precisa de estar acima do sensor para o fazer transformar o pino GPIO em estado elevado. O comportamento do código é consistente com as expectativas? Os sprites no ecrã mudam em resposta a sensores específicos que estão a ser cobertos/descobertos? Será que o alarme dispara quando pelo menos uma janela é aberta e o gato diz Alarme! nesta situação?

Apresentação dos resultados

Nesta fase, somos convidados a partilhar os resultados do nosso trabalho com outros grupos. Será que tudo funcionou bem? Houve dificuldades na montagem de todo o circuito? Foram introduzidas algumas alterações ao código? Em caso afirmativo, que tipo de alterações? As leituras sobre quão perto o obstáculo tem de estar para transformar a porta GPIO do sensor em estado elevado foram consistentes para todos os sensores? Foi também o mesmo caso em outros grupos?

1.2.4 Conclusão

Conseguimos criar um circuito muito simples que pode servir como demonstração de uma instalação de segurança no edifício. Nesta fase, podemos trocar ideias com outros grupos, o que foi feito, de que forma, por que ordem e esclarecer quaisquer questões que possam surgir.

1.2.5 Exercício de Follow-up (opcional)

O exercício de seguimento pode incluir um multímetro, medindo a voltagem dos sensores, em modo direto, à medida que o obstáculo se aproxima. Desta forma, poderemos saber a que voltagem o Raspberry Pi considera a entrada no estado *alto/elevado*. Podemos também tentar mudar o loop principal utilizado no código de *repetição* para *sempre*.



Erasmus+

2019-1-FR01-KA201-062281



STEMKIT
4SCHOOLS

1.3 Referências ou Recursos

Recurso relacionado com este plano de aula:

<https://www.vishay.com/docs/83760/tcrt5000.pdf>