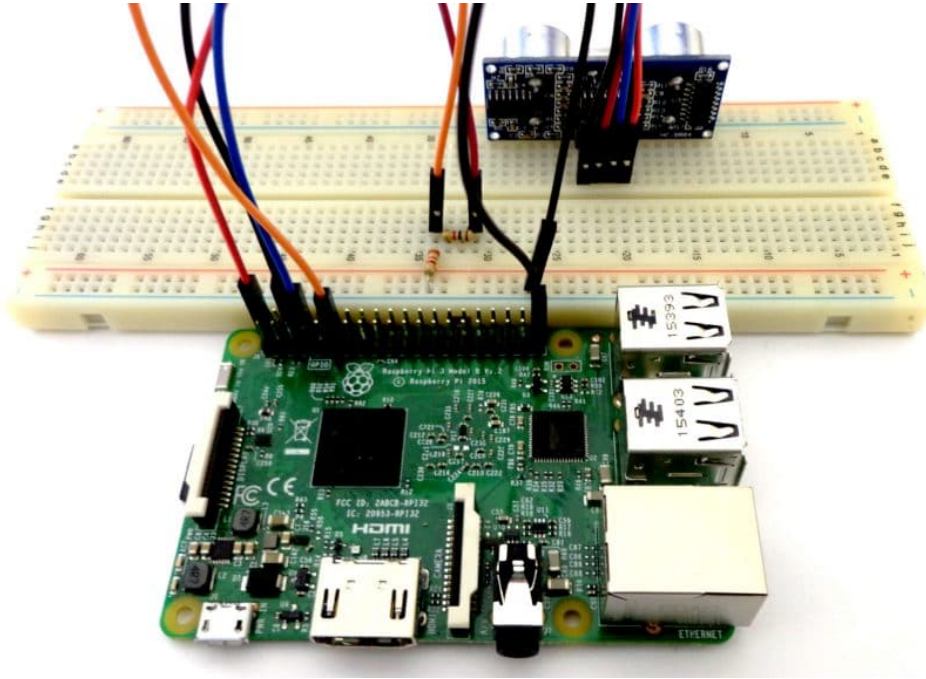


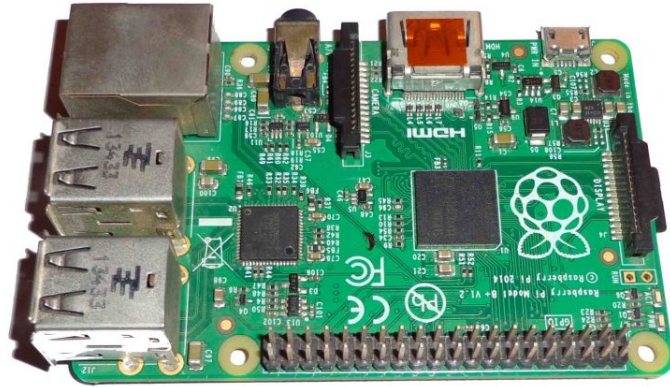
تحديد المسافات باستخدام الـ راسبيري باي

في هذا الدرس سنتعلم كيفية توصيل حساس تحديد المسافات Ultrasonic Module بالراسبيري باي، حيث سيمكّنك هذا الحساس من تصميم الروبوتات التي تتفادى العوائق من حولها حتى وهي تتحرك في الظلام.

فيعتمد هذا الحساس على إرسال موجات فوق سمعية وإستقبالها مرة أخرى ويمكن من خلال حساب فرق الزمن بين الموجة المرسلّة والموجة المستقبلة من تحديد مواقع العوائق وهذه الطريقة هي نفس طريقة الرؤية لدى الخفافيش فهي لا تملك أعين ولكنها تستطيع الطيران بسهولة وتفادي العوائق.



المكونات المطلوبة



راسبيري باي



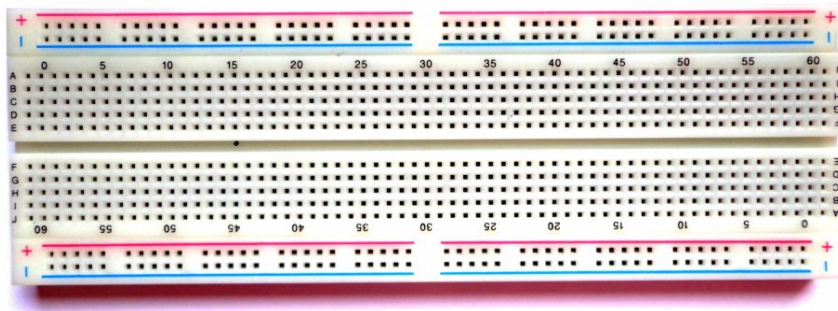
حساس تحديد المسافات Ultrasonic Module



أسلاك توصيل Female / Female jumper



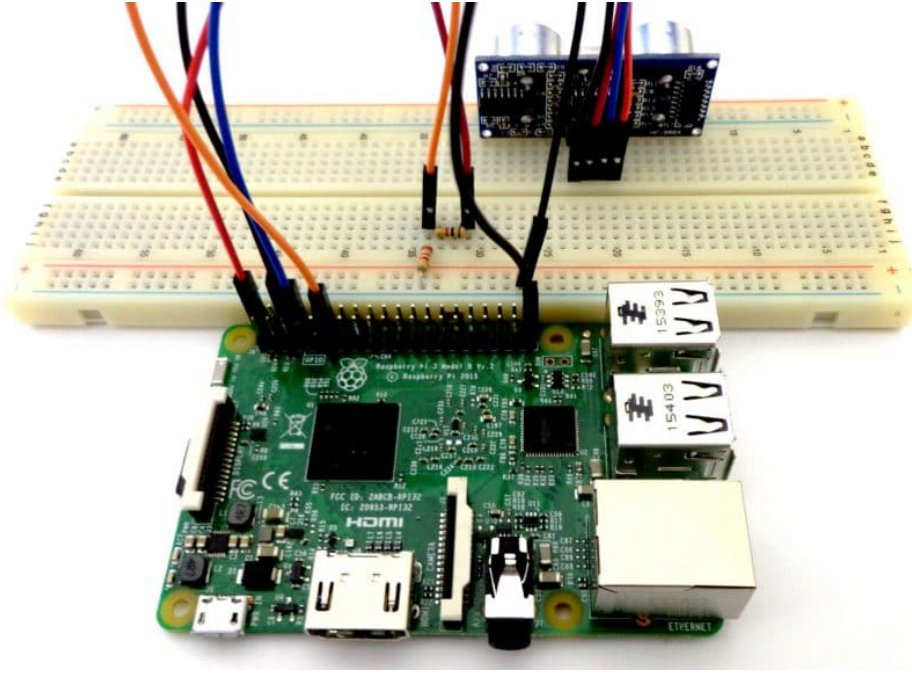
أسلاك توصيل Female / Male jumper



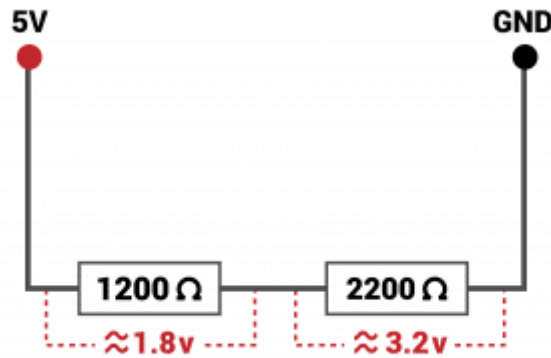
لوحة تجارب Breadboard



مقاومة 1.2 كيلو أوم



توصيل هذه الدائرة سهل وبسيط ولكن الأساس بها مراعاة أن حساس الـ ultrasonic يعمل على 5 فولت وأن الراسبييري باي تعمل على 3 فولت، فيجب علينا تقسيم الفولت الخارج من الحساس لكي يصل للراسبييري باي كفولت منخفض ولا يحرق الدائرة، كل ذلك يتم بإستخدام مقاومتين أحدهما كبيرة 2.2 كيلو أوم والأخرى صغيرة 1.2 كيلو أوم كالتالي.



في البداية نقوم بفتح الـ Terminal الخاص بـ الراسبييري باي وكتابة الأوامر التالية أو نقوم بفتح نافذه الـ SSH الخاصة بها من جهاز آخر مربوط معها علي نفس الشبكة كما تم شرحه في الدرس الخامس.

الأساس في هذا الكود هو المعادلة الحسابية التي يعمل عليها الحساس حيث أن السرعة التي يتحرك بها أي جسم تساوي المسافة التي تحركها مقسومة على الزمن الذي احتاجه لقطع هذه المسافة.

$$\frac{\text{المسافة}}{\text{الزمن}} = \text{السرعة}$$

ومن هنا نحصل على معادلة المسافة والتي تقسم على 2 وذلك حيث أن الصوت المرسل من حساس الـ Ultrasonic يتحرك مرتين مره ذهاباً حتى يصطدم بالجسم الذي أمامه ويعود مرة أخرى للحساس.

$$\text{المسافة} = \text{السرعة} \times \text{الزمن}$$

$$\frac{\text{السرعة} \times \text{الزمن}}{2} = \text{المسافة}$$

فى البداية نقوم بفتح ملف بايثون ونسميه ultrasonic.py

```
sudo nano ultrasonic.py
```

ثم نقوم بكتابة الكود التالي بداخله.

```
import RPi.GPIO as GPIO          #Import GPIO library
import time                      #Import time library
GPIO.setmode(GPIO.BCM)          #Set GPIO pin numbering

TRIG = 4                         #Associate pin 4 to TRIG
ECHO = 17                       #Associate pin 17 to ECHO

print "Distance measurement in progress"

GPIO.setup(TRIG,GPIO.OUT)        #Set pin as GPIO out
GPIO.setup(ECHO,GPIO.IN)         #Set pin as GPIO in

while True:

    GPIO.output(TRIG, False)     #Set TRIG as LOW
    print "Waiting For Sensor To Settle"
    time.sleep(2)                #Delay of 2 seconds

    GPIO.output(TRIG, True)       #Set TRIG as HIGH
    time.sleep(0.00001)           #Delay of 0.00001 seconds
    GPIO.output(TRIG, False)      #Set TRIG as LOW

    while GPIO.input(ECHO)==0:    #Check whether the ECHO is LOW
        pulse_start = time.time() #Saves the last known time of LOW pulse

    while GPIO.input(ECHO)==1:    #Check whether the ECHO is HIGH
        pulse_end = time.time()  #Saves the last known time of HIGH pulse

    pulse_duration = pulse_end - pulse_start #Get pulse duration to a variable

    distance = pulse_duration * 17150 #Multiply pulse duration by 17150 to get
distance                          #Round to two decimal points
    distance = round(distance, 2)

    if distance > 2 and distance < 400: #Check whether the distance is within
range
        print "Distance:",distance,"cm" #Print distance
        print "Out Of Range"           #display out of range
```

ثم نقوم بتشغيل البرنامج عن طريق الأمر التالي.

```
sudo python ultrasonic.py
```

نلاحظ أن البرنامج يعمل ويقوم بعرض المسافة بين حساس الـ ultrasonic والكائن الذي أمام، قم بتغيير المسافة بين الحساس والكائن الذي أمامه ستلاحظ أن المسافة تتغير بالفعل.

```

pi@raspberrypi: ~
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Apr 18 21:17:09 2017
pi@raspberrypi:~ $ sudo pyhton ultrasonic.py
sudo: pyhton: command not found
pi@raspberrypi:~ $ sudo python ultrasonic.py
Distance measurement in progress
Waiting For Sensor To Settle
Distance: 12.72 cm
Waiting For Sensor To Settle
Distance: 12.79 cm
Waiting For Sensor To Settle
Distance: 12.81 cm
Waiting For Sensor To Settle
Distance: 13.22 cm
Waiting For Sensor To Settle
Distance: 12.74 cm
Waiting For Sensor To Settle
Distance: 12.78 cm
Waiting For Sensor To Settle

```

أما عن شرح الكود الذي استخدمناه فهو في غاية السهولة حيث يتم إرسال نبضة للرجل Trig مع الأخذ في الاعتبار مقدار الوقت بين high و الـ low وهو 10 ميكرو ثانية كالتالي.

```

GPIO.output(TRIG, True)           #Set TRIG as HIGH
time.sleep(0.00001)               #Delay of 0.00001 seconds
GPIO.output(TRIG, False)          #Set TRIG as LOW

```

يقوم الحساس بالانتظار حتى تأتي النبضة وتستقبل من خلال الرجل echo فبمجرد إرسال النبضة يقوم بحساب بداية الوقت ثم بمجرد وصول النبضة للحساس يقوم بحساب نهاية الوقت ثم بطرحهم من بعضهم يحصل على الوقت المستغرق لترسل النبضة ثم تستقبل.

```

while GPIO.input(ECHO)==0:         #Check whether the ECHO is LOW
    pulse_start = time.time()      #Saves the last known time of LOW pulse

while GPIO.input(ECHO)==1:         #Check whether the ECHO is HIGH
    pulse_end = time.time()        #Saves the last known time of HIGH pulse

pulse_duration = pulse_end - pulse_start #Get pulse duration to a variable

```

بمعلومية سرعة الصوت في الهواء والتي تساوي 343 متر في الثانية أي تساوي 34300 سنتي متر في الثانية، نقوم بالتعويض في المعادلة لتصبح الصيغة النهائية هي حاصل ضرب الزمن في 17150، ثم نقوم بالتقريب لأقرب رقمين عشريين.

```

distance = pulse_duration * 17150  #Multiply pulse duration by 17150 to get
distance                                  distance
distance = round(distance, 2)       #Round to two decimal points

```

مع الأخذ في الاعتبار بأن هذا الحساس لا يستطيع الأحساس بالمسافات الأقل من 2 سنتي متر ولا أكثر من 4 أمتار، لذلك يتم إضافة الدالة الشرطية التالية.

```

if distance > 2 and distance < 400:    #Check whether the distance is within range

```

```
    print "Distance:",distance,"cm"  #Print distance
else:
    print "Out Of Range"              #display out of range
```