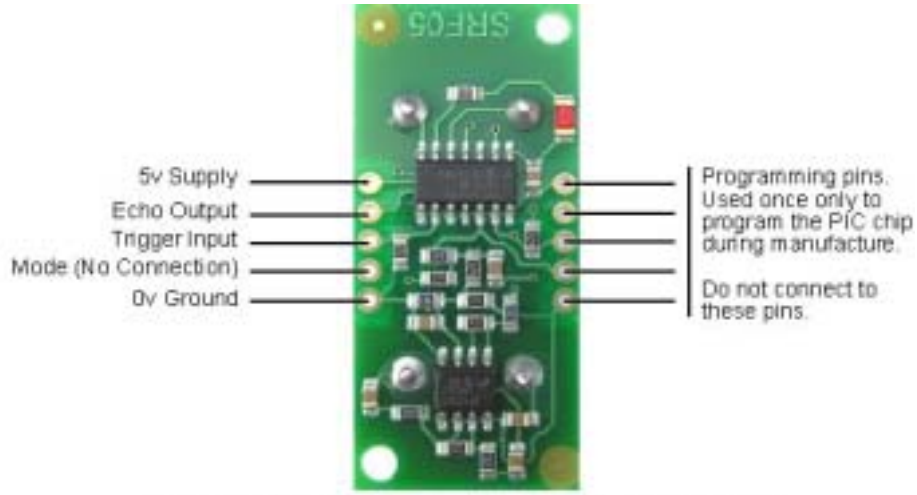
ULTRASONIC RANGE SENSOR

Specification:

The ultrasonic range sensor detects objects in it's path and can be used to calculate the range to the object. It is sensitive enough to detect a 3cm diameter broom handle at a distance of over 3m.

Voltage	- 5v
Current	- 30mA Typ. 50mA Max.
Frequency	- 40KHz
Max Range	- 3 m
Min Range	- 3 cm
Sensitivity	- Detect 3cm diameter broom handle at > 3 m
Input Trigger	- 10uS Min. TTL level pulse
Echo Pulse	- Positive TTL level signal, width proportional to range.
Small Size	- 43mm x 20mm x 17mm height

Electrical connection (Mode 1 - separate trigger / echo pins):



Connections for 2-pin Trigger/Echo Mode (SRF04 compatible)

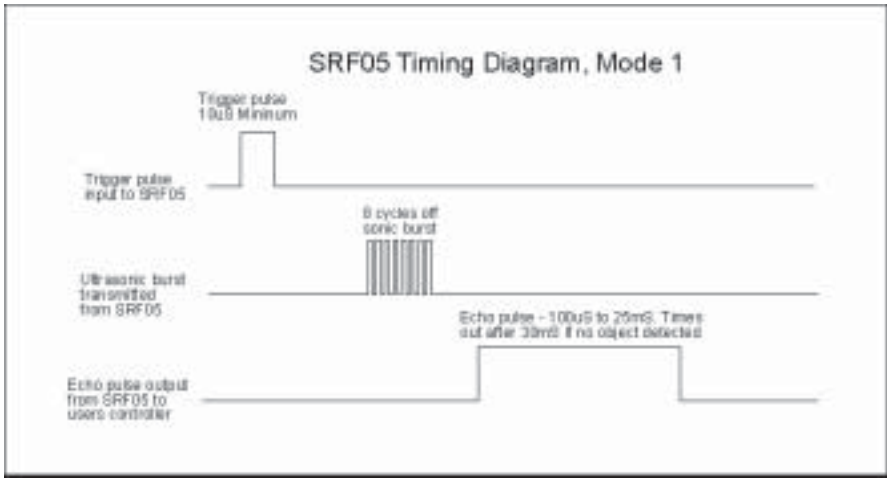
The SRF005 ultrasonic range finder has 5 connections pins. The power supply is connected to the 5V and 0V ground connections on the SRF005.

Important - Note that the 'Mode' (hole 4) connection **MUST NOT** be connected for correct operation in this mode. This is a different layout to the discontinued SRF004 module, where the mode pin was connected to 0V when used with the PICAXE system. If using the SRF005 module on a PCB designed originally for the SRF004 (e.g. the Micro-robot) cut off the mode pin from the 5 pin connector, so that no connection is made on this pin when inserted onto the main PCB.

Take care not to overheat, and therefore damage, the solder connection pads whilst making connections.

The SRF005 **Trigger Input** is connected to a PICAXE **output** pin.
The SRF005 **Echo Output** is connected to a PICAXE **input** pin.

Operation with the PICAXE microcontroller:



The following program gives an example of how to use the SRF005 module with a PICAXE-18 microcontroller. Output 3 is used to trigger the SRF005 module via a 'pulsout' command. The SRF005 module then sends out the sonic burst, and sets the Echo Output connection high for the time it takes the sonic burst to be returned. Therefore the PICAXE input (input 6) is used to receive and time this echo pulse via a 'pulsin' command.

The length of the echo pulse is then divided by 5.8 to give a value in cm, and displayed on the computer screen via the 'debug' command. Note that a word variable, w1, is used for the echo timing, as the echo pulse will be a value greater than 255 (maximum value of a byte variable). Word variables are made up of two byte variables and so have a maximum value of 65535 (in this case w1 is made up of b2 and b3, so these two byte variables must not be used anywhere else in the program).

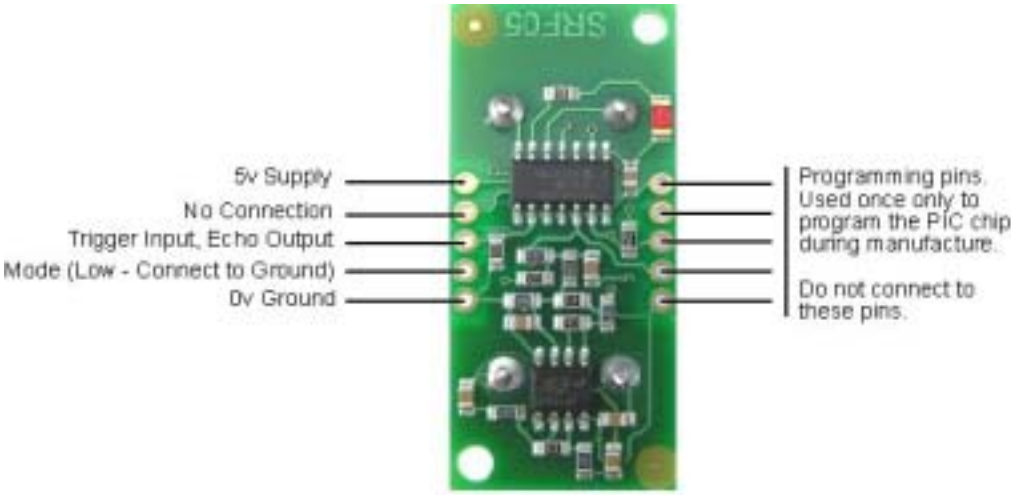
Sample PICAXE Program:

```
symbol trig = 3          ' Define output pin for Trigger pulse
symbol echo  = 6         ' Define input pin for Echo pulse
symbol range = w1        ' 16 bit word variable for range

main:
   pulsout trig,2      ' produce 20uS trigger pulse (must be minimum of 10uS)
   pulsin echo,1,range   ' measures the range in 10uS steps
   pause 10            ' recharge period after ranging completes
' now convert range to cm (divide by 5.8) or inches (divide by 14.8)
' as picaxe cannot use 5.8, multiply by 10 then divide by 58 instead
   let range = range * 10 / 58 ' multiply by 10 then divide by 58
   debug range         ' display range via debug command
   goto main           ' and around forever
```

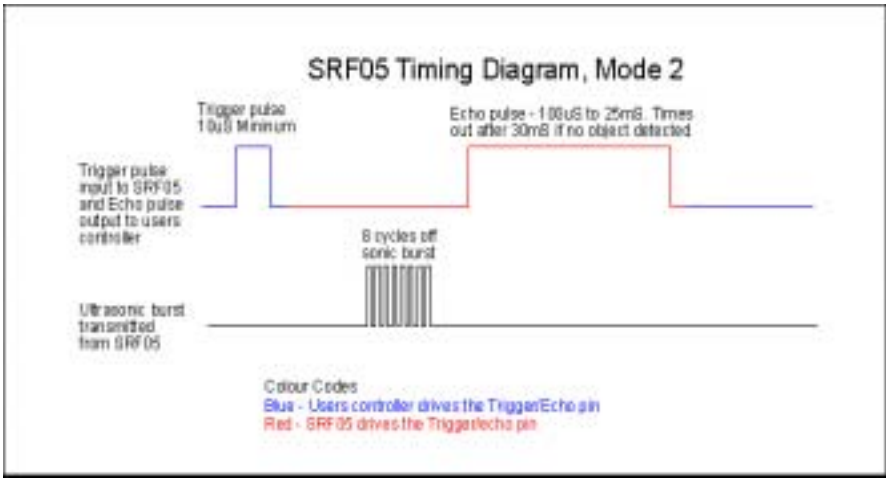
Special Mode 2 - For use with PICAXE-08/08M microcontroller:

As the PICAXE-08 and 08M have limited input/output pins, the SRF005 has a special mode for use with these chips. In this mode a single pin is used for both input and output (not possible with 18 and 28 pin PICAXE chips).



Connections for single pin Trigger/Echo Mode

This PICAXE pin used can be pin1, pin2 or pin4.



The following program gives an example of how to use the SRF005 module with a PICAXE-08 microcontroller.

Sample PICAXE Program:

```
symbol trig = 1          ' Define output pin for Trigger pulse

main:
   pulsout trig,2       ' produce 20uS trigger pulse (must be minimum of 10uS)
   pulsin trig,1,range    ' measures the range in 10uS steps
' now convert range to cm (divide by 5.8) or inches (divide by 14.8)
' as picaxe cannot use 5.8, multiply by 10 then divide by 62 instead
   let range = range * 10 / 58 ' multiply by 10 then divide by 62
   debug range          ' display range via debug command
    pasue 50            ' short delay
   goto main            ' and around forever
```