PICAXE COLOUR SENSOR

Overview:

The PICAXE Colour Sensor is a complete RGB (red green blue) colour sensor module for colour detection and sorting operations. The sensor can be interfaced to all PICAXE chips that support the 'count' command.

The sensor module can be purchased individually (part AXE045) or as part of a kit (AXE112S). The kit additionally contains a PICAXE-08M chip and AXE024 project board, as well as mechanical mounting hardware. The project board is designed with the same dimensions as the sensor to build a complete, compact self contained 'tower' system.

AXE045 Colour Sensor

Contents (AXE045 Colour Sensor):

- Colour sensor PCB
- 2 while LEDs. 2 LED support posts. 2 LED angle support posts.
- Lens assembly

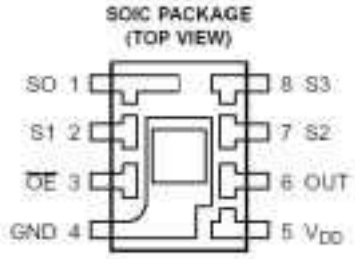
AXE112S Sensor Kit

Contents (AXE112S Starter Pack):

- AXE045 Colour Sensor Kit (as above)
- AXE024 PICAXE-08M Servo Driver Kit
- 5 pin double row PCB connection socket
- 4 30mm support posts, 4 12mm support posts, 4 M3 6mm screws

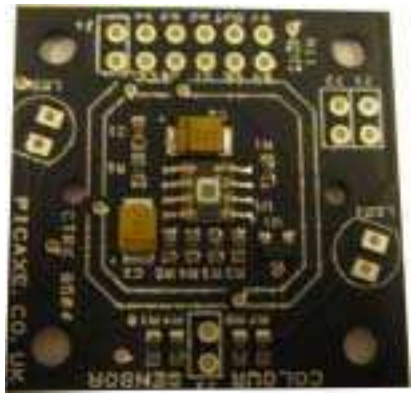
General Operation:

The AXE045 Colour Sensor module is built around a TAOS TCS230 colour sensor. The TAOS TCS230 sensor chip is a 'programmable colour light-to-frequency converter', supplied as a small 8 pin SOIC format chip. In simple terms this means that it is a sensor that can measure the reflected light intensity from an illuminated sample, by use of an 8 by 8 array of photodiodes. Of these 64 photodiodes, 16 are covered by blue filters, 16 have red filters, 16 have green filters and 16 have no filter.



White light is an equal mixture of red, green and blue light. By mixing red, green and blue light in different combinations you can get any other colour – as demonstrated by your television screen (which displays colours as combinations of red, green and blue dots).

The red, green and blue (RGB) filters embedded in the TAOS sensor ensure only these colours are exposed to the underlying photodiodes. Therefore, by using a PICAXE microcontroller, the user can programmatically select which coloured filtered photodiodes to use at a particular time, and, by cycling through the 4 different options, come to a very good approximation of the RGB content of the sample – hence identifying its colour! This may sound complicated, but is actually quite straight forward to do!

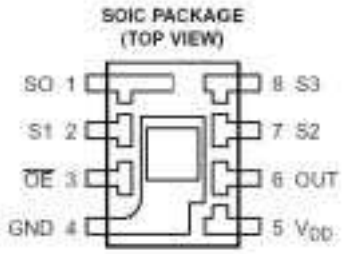


As the TAOS TCS230 colour sensor is a tiny surface mount device, it is supplied pre-soldered on the AXE045 module PCB. This module also contains two white LEDs – these shine down at 45 degrees onto the sample to flood it with white light. The reflected light is then focussed by a small lens (as typically used in CCD cameras) onto the TCS230 chip. The lens also filters out any unwanted background infra-red light.

Selection of the different filtered photodiodes is achieved by the S2 and S3 input pins, as defined in the table shown below. By placing these two pins high or low the different photodiodes are selected.

TAOS Colour Filter Selection table

S2	S3	Filter
0	0	Red
0	1	Blue
1	0	None
1	1	Green



The output of the sensor is a square wave with the frequency directly proportional to the light intensity. Therefore by counting the number of pulses in a short sample time (e.g. 50ms), you can gain a reliable indication of the light intensity. This counting is easily achieved, for example, by a PICAXE-08M chip using the COUNT command.

Therefore the process for measuring the RGB light intensity from a sample is:

1. Connect S0 and S1 pins to 5V and connect OE pin to 0V.
2. Select red filters (S2=0, S3=0)
3. Count pulses on OUT for a short time (red value)
4. Select blue filters (S2=0, S3=1)
5. Count pulses on OUT for a short time (blue value)
6. Select green filters (S2=1, S3=1)
7. Count pulses on OUT for a short time (green value)

Note that background lighting conditions and distance from the sample will also make a small difference to the readings, so you may need to re-calibrate the sensor when moving it's position. Focusing of the lens and angle/position of the white LEDs is also important.

How accurate is the sensor?

Most colour sensors vary in accuracy across the RGB spectrum, and the TAOS TCS230 is no exception to this rule. With, for instance, a video camera it is normal to perform a 'white balance' test before use. In theory, the three RGB sensors should record an equal value for pure white light, but in practice they don't. The purpose of a 'white balance' test is to calculate a 'scaling factor' to apply to each colour to balance them, so that pure white light gives an equal (recorded) amount of red green and blue. In the case of the video camera this makes the playback of the colours more accurate on a television screen.

However with a simple microcontroller colour detection system it is actually not necessary to scale the readings – if you are not ever going to try and 'reproduce' the colours elsewhere the inaccuracy is not so relevant. Here we are simply interested in a 'threshold point' for each colour. Therefore if we account for the imbalance within the thresholds set for each colour by experimentation, no 'white balance' test is required. However for more advanced work a white balance test and scaling can be fairly simply added to the software program if required (although this will probably require the extra memory space of an X part).

White LED assembly

Before use the two white LEDs must be soldered to the sensor module. Start by laying the the LED down and threading the right angle support post, and then the straight support post, onto the LED legs as shown in the picture. Note in the picture that the anode (long leg) of the LED is furthest away, and that the 'solid' end of the straight support post is closest the end of the LED leg.
The two LEDs should then be inserted into the holes of the PCB, so that the LEDs are mounted beside the lens as shown. Double check that the long leg of the LED is being inserted into the hole marked +.
When the LED legs are gently pulled through the holes on the PCB the LEDs will automatically bend to the correct angle of 45 degrees.
Finally carefully solder the LEDs in place and trim the legs.

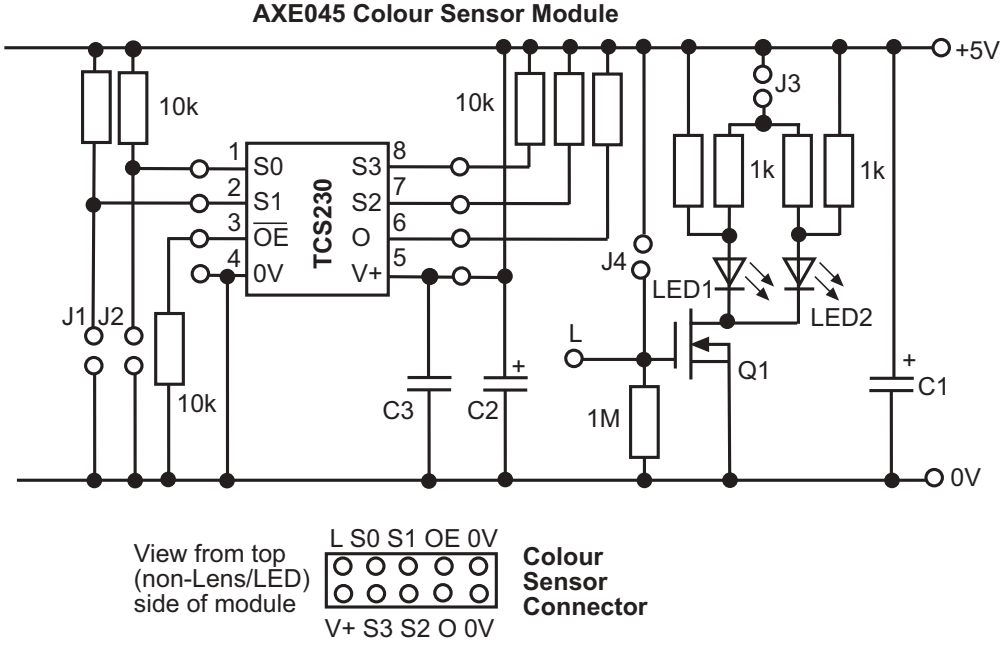
Lens insertion

The lens must be screwed into the holder until the top of the lens is level with the top of the holder. Do not touch the lens window.

Kit Assembly (AXE112S only)

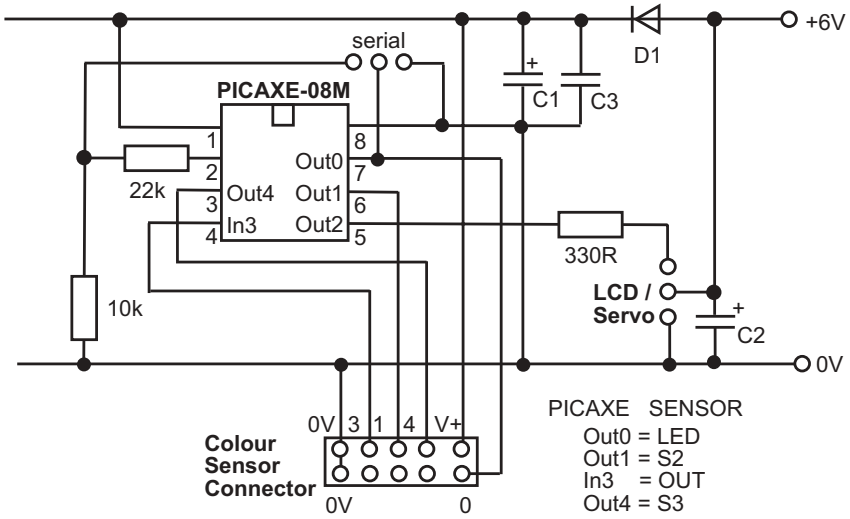
Assemble the AXE024 PICAXE-08M servo driver kit according to the separate instructions supplied in the AXE024 kit.
Then solder the 2x5 way connector socket on the **underside** of the servo driver PCB (ie solder joints are on the top side, next to the other components). This is so the connector socket mates correctly with the header on the colour sensor PCB.
Finally assemble the two PCBS together by use of the 12mm spacers between the two PCBs. Assemble the 30mm spacers under the colour sensor as 'legs' for the complete unit.

Colour Sensor Circuit Diagram.



In normal use all connections are taken from the 2x5 connector header. J1, J2 and J4, are not normally used. J3 halves the resistance of the LED series resistor, and so should only be used when a 3V supply is used (leave jumper open for a 4.5V or 5V supply).

Typical PICAXE Circuit Diagram.



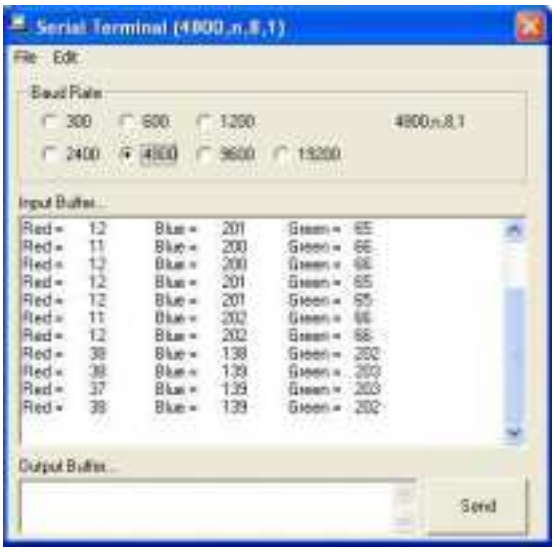
The circuit diagram shows the PICAXE interfacing circuit from the AXE024 project board. If building your own circuit (e.g. with the larger PICAXE-18X chip) a similar interfacing arrangement will be required.

Note that the project board is designed to run from a 6V (4xAA) battery pack. A diode provides a 0.7V drop before the PICAXE circuit, ensuring an acceptable maximum voltage of 5.3V at the PICAXE circuit.

With this PICAXE-08M circuit, output 0 controls the white LEDs, and outputs 1 and 4 are linked to control pins S2 and S3 to select the appropriate colour filter. Input 3 is connected to the TCS230 pulse output. On the 08M chip this leaves one output (output 2) available for other use e.g. to drive a serial LCD or servo. Naturally if you require more input/output pins for a project the same circuit and program could be ported to a PICAXE-18X or 28X chip. When building your own circuit you could also leave the white LEDs permanently on (via jumper J4), freeing up another PICAXE pin for use elsewhere. However jumper J4 **must not** be used with the AXE024 project board, as this would short circuit output 0. J4 is also not recommended for battery applications, as the LEDs are permanently on.

Testing:

Once the circuit is complete it is necessary to run some tests to 'learn' the reflected light properties of some samples. A convenient item to first test are small sweets (candy). You will need to adjust the height of the tower to compensate for the height of the sweet - the most suitable height is about 30mm from the top surface of the subject. It is also important that the two LEDs shine together onto the subject - the light from each LED should merge to form a single light dot. If you have a 'figure of 8' light pattern try adjusting the angle of the LEDs or height from subject.



The test program is shown in Appendix 1. Download this program into the PICAXE-08M chip and then use the Programming Editor Terminal function (PICAXE>Terminal menu at baud rate 4800) to display the RGB data being output via the 'sertxd' command.

If you wish to perform the testing away from the computer, you can use a serial LCD module (part AXE033) connected to output 2. Appendix 2 shows the 'main' section of program 1 altered for use with the serial LCD module.

Colour Identifcation

A table of colour values, found by experimentation, is shown below. Note these figures are from a test setup and will need to be repeated and adjusted for your own setup. These figures use very broad 'thresholds' as the reading will be slightly different each time the sweet moves, particularly as it is a rounded shape. However the important principle is that each colour sweet can be uniquely identified from these values – no colour has the same threshold values for each of the three colour filtered photodiodes.

Sample Threshold Values

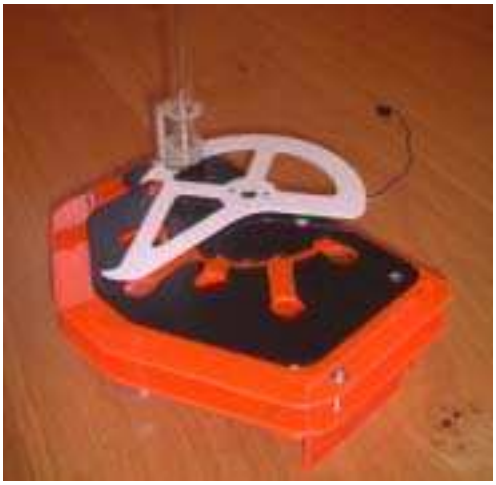
Sweet	red _value	blue_value	green_value
blue	0<w4<50	200<w5<350	50<w6<150
green	0<w4<50	100<w5<200	200<w6<300
red	50<w4<100	20<w5<100	20<w6<80
yellow	150<w4<250	80<w5<120	230<w6<350

Sample Project

Appendix 3 demonstrates how to use these figures for colour identification. This program also controls a normal radio control type servo. The servo arm is used as a 'pointer' to indicate which colour sweet is under the sensor. A simple dial can be easily made from a blank CDROM, which is then placed under the servo arm. The servo itself is simply connected to output 2 of the PICAXE chip. This program is a bit more involved as it has to determine which sweet is which colour by some mathematical comparisons. This is achieved by testing the threshold values for each of the RGB values from the table above.

Classroom Demonstration Kit

For use in schools a complete laser cut 'sweet sorter' mechanism (part MOD020) from KH Resources is also available. This model uses the colour sensor to sort sweets by colour into separate bins.



Appendix 1

This program displays the RGB value on screen via the 'sertxd' command and the Serial Terminal function.

```
' ****************************************
' PICAXE-08M input/output pins

symbol LED = 0      ' Colour sensor white LEDs      (output 0)
symbol S2  = 1      ' Colour sensor select S2       (output 1)
symbol ser = 2      ' Servo or serial LCD           (output 2)
symbol CSI = 3      ' Colour sensor pulse           (input 3)
symbol S3  = 4      ' Colour sensor select S3       (output 4)

' ****************************************
' Variables

symbol red_value     = w4  ' Colour sensor red content
symbol blue_value    = w5  ' Colour sensor blue content
symbol green_value = w6    ' Colour sensor green content
                           ' Remember w4-w6 uses b8-b13!
' ****************************************
' scan and display every second
main:
       gosub colour ' scan the colour

       sertxd ("Red =", 9, #red_value, 9)
       sertxd ("Blue =", 9, #blue_value, 9)
       sertxd ("Green =", 9, #green_value, CR, LF)
       pause 1000
       goto main

' ****************************************
' sub to scan colours

colour:
       high LED                  ' LED on
       low S2                    ' read red into w4
       low S3
       count 3, 50, red_value
       high S3                   ' read blue into w5
       count 3, 50, blue_value
       high S2                   ' read green into w6
       count 3, 50, green_value
       low LED                   ' LED off
       return
```

Program 2

This section of code provides an alternate 'main' section of code for use with the AXE033 serial LCD module. Use this section of code within the complete program above.

```
main:
       gosub colour ' scan the colour

       serout 2,N2400,(254,128"R=",#red_value, "   ")
       serout 2,N2400,(254,136"B=",#blue_value, "   ")
       serout 2,N2400,(254,192"G=",#green_value, "   ")
       pause 1000
       goto main
```

Appendix 3

This program displays the current colour via use of a servo 'pointer'. Note that the 'new-pos' positions for each of the the pointer positions will need to be adjusted by experimentation.

```
'  ****************************************
'  PICAXE-08M  input/output  pins

symbol LED  = 0       '  Colour  sensor  white  LEDs        (output  0)
symbol S2   = 1       '  Colour  sensor  select  S2         (output  1)
symbol ser  = 2       '  Servo  or  serial  LCD             (output  2)
symbol CSI  = 3       '  Colour  sensor  pulse              (input  3)
symbol S3   = 4       '  Colour  sensor  select  S3         (output  4)

'  ****************************************
'  Variables

symbol  red_value      = w4         '  Colour  sensor  red  content
symbol  blue_value     = w5         '  Colour  sensor  blue  content
symbol  green_value  =  w6          '  Colour  sensor  green  content
                                    '  Remember  w4-w6  uses  b8-b13!
'  ****************************************

main:
       gosub  colour                '  scan  the  colour
       gosub  evaluate              '  set  the  servo  position
       servo  2,  new_pos           '  move  the  servo
       pause  1000
       goto  main


'  ****************************************
'  sub  to  scan  colours

colour:
       high  LED                    '  LED  on
       low  S2                      '  read  red  into  w4
       low  S3
       count  3,  50,  red_value
       high  S3                     '  read  blue  into  w5
       count  3,  50,  blue_value
       high  S2                     '  read  green  into  w6
       count  3,  50,  green_value
       low  LED                     '  LED  off
       return

'...(continued  overleaf)
```

```
'... (continued from previous page)

  '  ****************************************
  '  sub to evaluate colour and then set the servo position

  evaluate:
        new_pos = 190        ' preload reject position

  '  now identify correct colour using the threshold values
        if red_value > 150 and red_value < 250 then test_yellow
        if red_value > 50 and red_value < 100 then test_red
        if red_value < 50 then test_blue_or_green
        return

  '  test_blue_or_green:
        if blue_value > 200 and blue_value < 350 then test_blue
        if blue_value > 100 and blue_value < 200 then test_green
        return

  test_blue:
        if green_value > 50 and green_value < 150 then is_blue
        return
  is_blue:
        new_pos = 170        ' load blue colour position
        return

  test_green:
        if green_value > 200 and green_value < 300 then is_green
        return
  is_green:
        new_pos = 90         ' load green colour position
        return

  test_red:
        if blue_value > 20 and blue_value < 100 then test_r2
        return
  test_r2:
        if green_value > 20 and green_value < 80 then is_red
        return
  is_red:
        new_pos = 145        ' load red colour position
        return

  test_yellow:
        if blue_value > 80 and blue_value < 120 then test_y2
        return
  test_y2:
        if green_value > 230 and green_value < 350 then is_yellow
        return
  is_yellow:
        new_pos = 120        ' load yellow colour position
        return
```