4 WIRE RESISTIVE TOUCH SCREEN DRIVER

Table of Contents

OVERVIEW

Two of the four touch screen interface wires are for the X-axis and two are for the Y-axis. The pairs are operated on independently (e.g. the other pair is tristated by the microcontroller). The operation performed by the microcontroller on each pair is to apply Vcc to one wire and apply Gnd to the other. The wire to which Gnd is applied is also connected to an ADC input on the microcontroller, and the voltage read is proportional to the position being touched along that axis.

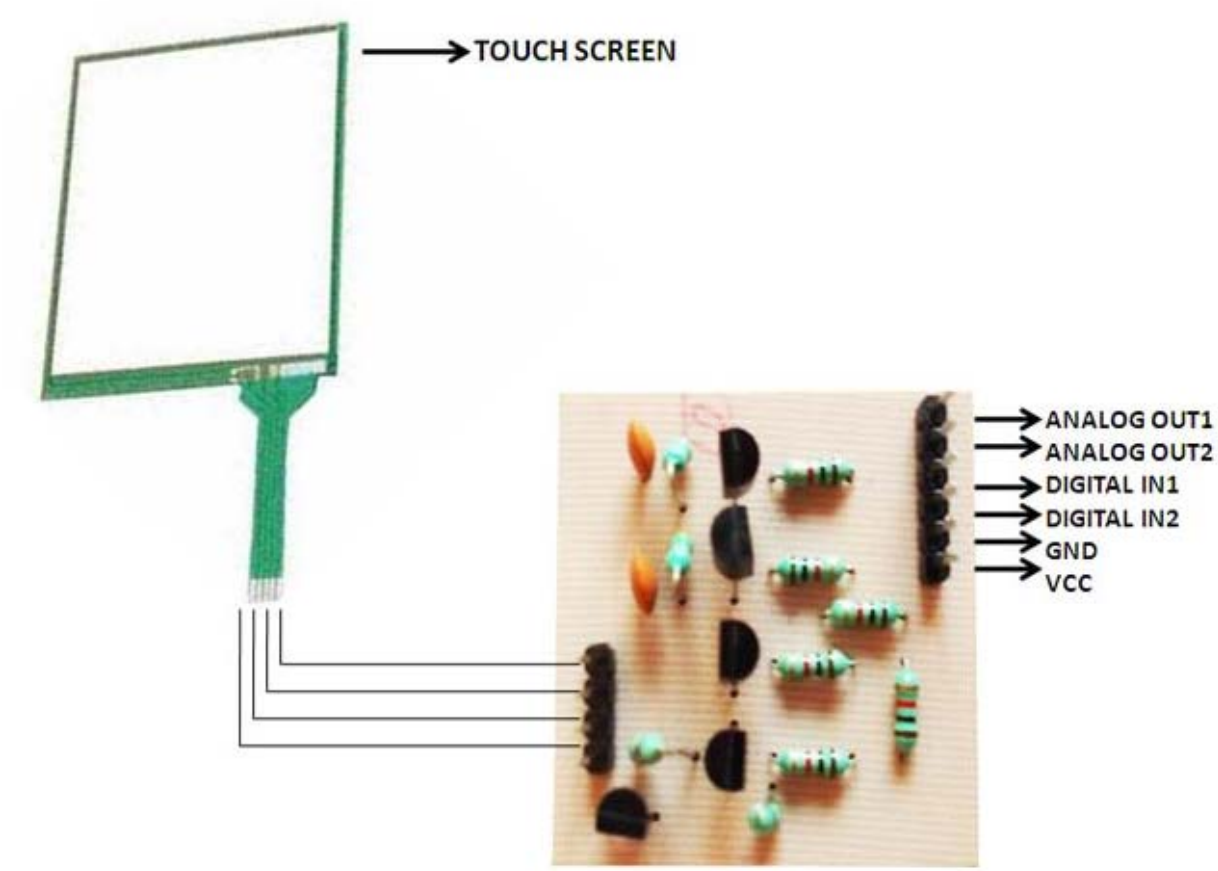
TOUCH SCREEN

This resistive touch screen can be used with a stylus or fingertip and is easy to use with a microcontroller. You can put it over a paper overlay for a touch control panel or attach it to an LCD to DIY a touch-activated display. Readings are taken by putting 5V across two of the pins and doing an analog to digital conversion on the other two pins. Full X and Y position can be achieved with only 4 GPIOs.

FEATURES

- 1.5mm thick.
- 3.2" diagonal active area, 80mm
- 600 ohms across X pins, 300 ohms across Y pins
- 4 wire resistive display, on a 0.5mm FPC connector
- Adhesive backing to attach easily to LCDS
- Easy to use
- 2 analog out pin
- 2 digital in pin

PIN CONFIGURATION



TOUCH SCREEN TEST CODE PIC

```
sbit DriveA at RD1_bit;

sbit DriveB at RD0_bit;

sbit DriveA_Direction at TRISD1_bit;

sbit DriveB_Direction at TRISD0_bit;

 char chVAL[16];

char chVAL1[16];




 long x_coord, y_coord,TEMP,LIGHT;

  unsigned int GetX() {


  DriveA = 1;                    // DRIVEA = 1 (LEFT drive on, RIGHT drive on

                                 // , TOP drive off )

  DriveB = 0;                    // DRIVEB = 0 (BOTTOM drive off )

  Delay_ms(5);

  return ADC_read(0);                    // reading X value from RA0 (BOTTOM)

}



unsigned int GetY() {

  //reading Y
```

```
 DriveA = 0;                    // DRIVEA = 0 (LEFT drive off , RIGHT drive off
                                // , TOP drive on)
 DriveB = 1;                    // DRIVEB = 1 (BOTTOM drive on)
 Delay_ms(5);
 return ADC_read(1);            // reading Y value from RA1 (from LEFT)
}

void main() {

  PORTA = 0x00;
 TRISA = 0x03;
  TRISA  = 0xFF;

    TRISD0_bit=0;
    TRISD1_bit=0;
       UART1_Init(9600);          // Initialize UART module at 9600 bps


 Delay_ms(100);
 UART1_Write_Text("Start");


  DriveA= 0;                    // Set DriveA pin as output
  DriveB= 0;
     Delay_ms(200);
```

```
    Delay_ms(200);
   while (1) {



      Delay_ms(5);
        x_coord= GetX();
y_coord = GetY();


       Delay_ms(200);
        UART1_Write_Text("X = ");
        Delay_ms(20);
 wordToStr(x_coord,chVAL);
         Delay_ms(200);
        UART1_Write_Text(chVAL);


         Delay_ms(200);
          UART1_Write_Text("  y = ");
           Delay_ms(200);
     wordToStr(y_coord,chVAL1);


          Delay_ms(200);
        UART1_Write_Text(chVAL1);
        Delay_ms(1000);
        UART1_Write(13);
```

```
}

}
```