www.kumantech.com

Hybrid Temperature Sensor

Overview

Sometimes conventionally called a “digital temperature sensor,” the hybrid temperature sensor is actually an *analog* temperature sensor, capable of reporting a voltage that varies with the outside temperature, which is then packaged a convenience circuit. This convenience circuit includes a digital switch that allows you to determine easily whether the current analog temperature exceeds some user-adjustable threshold temperature. You might prefer the digital output of a hybrid sensor to a purely analog temperature sensor in settings where you want to be able to adjust the threshold in the field. (The sensor includes a threshold dial—a variable resistor or trimpot—that allows you to alter the threshold without changing any source code on your microcontroller.) You might also use it in places where don’t want to configure an analog-to-digital converter to monitor the analog signal yourself.

In this experiment, you will use the Raspberry Pi to report on both the analog reading of the temperature sensor, as converted by the ADC0832; and the current value of the hybrid sensor’s digital threshold-detecting switch. (For more general information on analog sensors, see the Analog Temperature Sensor description.)

Experimental Materials

Raspberry Pi	x1
Breadboard	x1
Hybrid temperature sensor	x1
ADC0832	x1
Dupont jumper wires	

Experimental Procedure

1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GPIO library, and wiringPi library as described in READ_ME_FIRST.TXT.
2. Install the ADC0832 analog/digital converter IC and hybrid temperature sensor on your breadboard, and use Dupont jumper wires to connect them to each other and your Raspberry Pi as illustrated in the Wiring Diagram below.

3. Execute the sample code stored in this experiment's subfolder.
   If using C, compile and execute the C code:

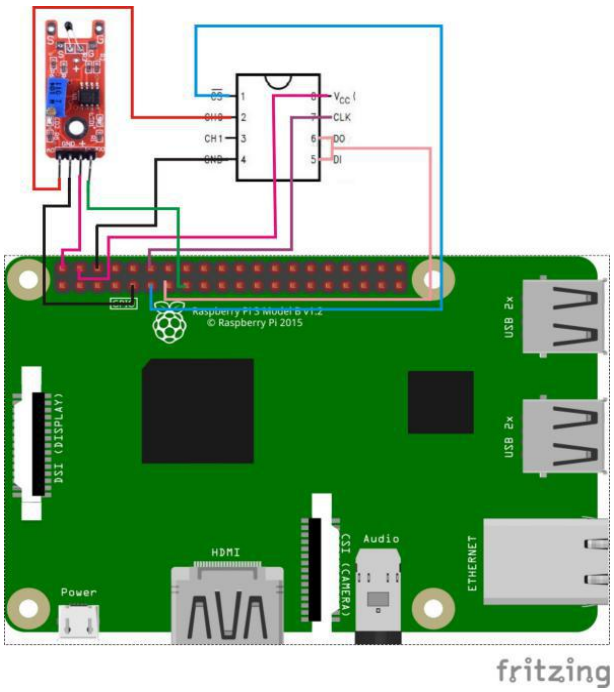
```
cd Code/C
gcc hybridTemp.c -o hybridTemp.out -lwiringPi
./hybridTemp.out
```

   If using Python, launch the Python script:

```
cd Code/Python
python hybridTemp.py
```

4. Make experimental observations. The Raspberry Pi command line output repeatedly outputs the numeric value of the analog temperature sensor. Slowly approach the sensor with an open flame (a match or candle) to increase the local temperature. Once the sensor's digital switch reports the increased heat has exceeded a certain level, the Raspberry Pi also reports "Temperature alarm! Threshold exceeded!" Remove the heat to quiet the alarm. If you wish to calibrate your temperature alarm to other heat sources—such as body-temperature, when you gently squeeze the thermistor with your fingertips—you can experiment with different settings of the threshold dial on the sensor.

Wiring Diagram



AD0382 Pin position:

CS	↔	Raspberry Pi pin 11
CLK	↔	Raspberry Pi pin 12
DO	↔	Raspberry Pi pin 13
DI	↔	Raspberry Pi pin 13
CH0	↔	Hybrid Temperature Sensor pin "S"
VCC	↔	Raspberry Pi +5V
GND	↔	Raspberry Pi GND

Hybrid Temperature Sensor pin position:

D0	↔	Raspberry Pi pin 15
A0	↔	ADC0382 pin CH0
"+"	↔	Raspberry Pi +5V
GND	↔	Raspberry Pi GND

Sample Code

Python Code

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import ADC0832
import time

TempSensor_DO_PIN = 15

def init():
   GPIO.setmode(GPIO.BOARD)
   GPIO.setup(TempSensor_DO_PIN, GPIO.IN)
   ADC0832.setup()
```

```
def loop():
    while True:
        global digitalVal
        print 'Current analog value is %d'%
        ADC0832.getResult(0)
        digitalVal = GPIO.input(TempSensor_DO_PIN)
        if(digitalVal == 1):
          print "Temperature alarm ... threshold exceeded!"
        else:
          pass
        time.sleep(0.2)

if __name__ == '__main__':
    init()
    try:
        loop()
    except KeyboardInterrupt:
        ADC0832.destroy()
        print 'The end !'
```

C Code

```
#include <wiringPi.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <stdlib.h>

#define     ADC_CS    0
#define     ADC_CLK   1
#define     ADC_DIO   2

#define  TempSensor_DO_Pin   3

typedef unsigned char uchar;
typedef unsigned int uint;
```

```
uchar get_ADC_Result(void)
{
   uchar i;
   uchar dat1=0, dat2=0;

   digitalWrite(ADC_CS, 0);
   digitalWrite(ADC_CLK,0);
   digitalWrite(ADC_DIO,1);  delayMicroseconds(2);
   digitalWrite(ADC_CLK,1);  delayMicroseconds(2);

   digitalWrite(ADC_CLK,0);
   digitalWrite(ADC_DIO,1);   delayMicroseconds(2);
   digitalWrite(ADC_CLK,1);  delayMicroseconds(2);

   digitalWrite(ADC_CLK,0);
   digitalWrite(ADC_DIO,0);  delayMicroseconds(2);
   digitalWrite(ADC_CLK,1);
   digitalWrite(ADC_DIO,1);   delayMicroseconds(2);
   digitalWrite(ADC_CLK,0);
   digitalWrite(ADC_DIO,1);   delayMicroseconds(2);

   for(i=0;i<8;i++)
   {
      digitalWrite(ADC_CLK,1);  delayMicroseconds(2);
      digitalWrite(ADC_CLK,0);   delayMicroseconds(2);

      pinMode(ADC_DIO, INPUT);
      dat1=dat1<<1 | digitalRead(ADC_DIO);
   }

   for(i=0;i<8;i++)
   {
      dat2 = dat2 | ((uchar)(digitalRead(ADC_DIO))<<i);
      digitalWrite(ADC_CLK,1);  delayMicroseconds(2);
      digitalWrite(ADC_CLK,0);   delayMicroseconds(2);
   }

   digitalWrite(ADC_CS,1);
   pinMode(ADC_DIO, OUTPUT);
   return(dat1==dat2) ? dat1 : 0;
}
```

```
int main(void)
{
   if(wiringPiSetup() == -1)
   {
      printf("setup wiringPi failed !");
      return -1;
   }

   pinMode(ADC_CS,  OUTPUT);
   pinMode(ADC_CLK, OUTPUT);
   pinMode(TempSensor_DO_Pin, INPUT);

   while(1)
   {
      printf("Current    analog    value    is    %d.\n",
get_ADC_Result());

      if(HIGH == digitalRead(TempSensor_DO_Pin)
      {
         printf("Temperature       alarm!       Threshold
exceeded!\n");
      }
      // else LOW; temperature threshold not exceeded

      delay(200);
   }

   return 0;
}
```