DS18B20 Temperature Sensor

Overview

This course uses the Raspberry Pi to control the DS18B20 temperature sensor and obtains temperature value.

Experimental Materials

RaspberryPi \*1

Breadboard \*1

DS18B20 module \*1

Dupont Line

Ready to work

1. Install python interpreter in your Raspberry Pi system

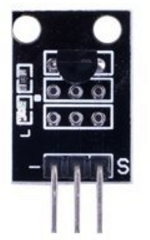
2. Install the RPi.GPIO library in your Raspberry Pi system

3. Install the wiringPi library in your Raspberry Pi system

Refer to the attached "Installing a Python Interpreter and Corresponding Libraries in a Raspberry Pi System" for details.

product description

DS18B20 is a commonly used digital temperature sensor. It has the characteristics of small size, low hardware cost, strong anti-interference ability and high precision. DS18B20 digital temperature sensor wiring is convenient, and it can be encapsulated with pipe, threaded, magnet adsorption, stainless steel. There are a variety of models,such as LTM8877, LTM8874 and so on. We can change its appearance depending on the application. The encapsulated DS18B20 can be used for cable trench temperature measurement, blast furnace water circulation temperature measurement, boiler temperature measurement, machine room temperature measurement, agricultural greenhouse temperature measurement, clean room temperature measurement, ammunition store temperature measurement and other non-limit temperature applications. It is wear-resistant, impact-resistant, small in size, easy to use and encapsulated in various forms. It is suitable for digital temperature measurement and control in various narrow spaces.



the characteristic parameters

The DS18B20 single-wire digital temperature sensor, the "one-wire device," has unique advantages:

(1) It use a single bus interface. Only one port line is required to connect the microprocessor to achieve two-way communication between the microprocessor and the DS18B20.

(2) Wide measurement temperature range and high measurement accuracy. The DS18B20 has a measurement range of -55 °C to + 125 °C. In the -10 to + 85 °C range, the accuracy is ± 0.5 °C.

(3) No external components are required in use.

(4) Multi-point networking function. Multiple DS18B20s can be connected in parallel on a single wire to achieve multi-point temperature measurement.

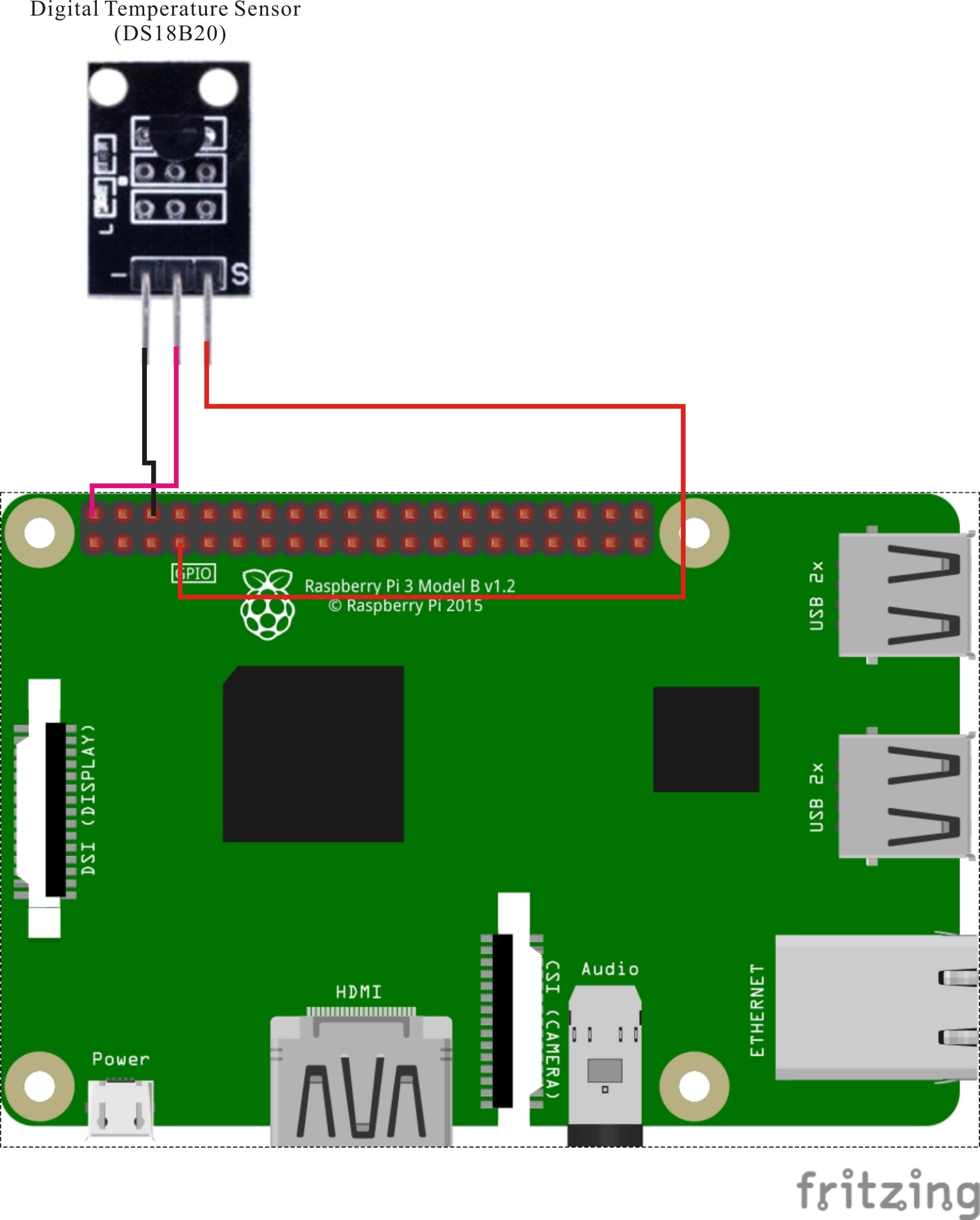
(5) Flexible power supply. The DS18B20 can be powered by the data line through an internal parasitic circuit. Therefore, when the timing on the data line meets the requirements,you need not connect the power supply, thereby the system structure will be simpler and more reliable.

(6) Measurement parameters can be configured. The DS18B20's measurement resolution can be programmed from 9 to 12 bits.

(7) When the power supply polarity of negative pressure characteristics is reversed, the thermometer will not be burned, but it will not work normally.

(8) Power-off protection function. The DS18B20 contains an internal EEPROM. After the system is powered down, it can still save the resolution and alarm temperature settings.

Wiring diagram



Sample code

1. python code

#!/usr/bin/env python

# Note:

# ds18b20's data pin must be connected to pin7.

# Reads temperature from sensor and prints to stdout

# id is the id of the sensor

import os

import time

def readSensor(id):

tfile = open("/sys/bus/w1/devices/"+id+"/w1\_slave")

text = tfile.read()

tfile.close()

secondline = text.split("\n")[1]

temperaturedata = secondline.split(" ")[9]

temperature = float(temperaturedata[2:])

temperature = temperature / 1000

print "Sensor: " + id + " - Current temperature : %0.3f C" % temperature

# Reads temperature from all sensors found in /sys/bus/w1/devices/

# starting with "28-...

def readSensors():

count = 0

sensor = ""

for file in os.listdir("/sys/bus/w1/devices/"):

if (file.startswith("28-")):

readSensor(file)

count+=1

if (count == 0):

print "No sensor found! Check connection"

# read temperature every second for all connected sensors

def loop():

while True:

readSensors()

time.sleep(1)

# Nothing to cleanup

def destroy():

pass

# Main starts here

if \_\_name\_\_ == "\_\_main\_\_":

try:

loop()

except KeyboardInterrupt:

destroy()

1. C code

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

#include <errno.h>

#include <stdlib.h>

#include <string.h>

#include <dirent.h>

#define BUFSIZE 128

char Buff[256] = {0};

char\* get\_ds18b20\_id(char \*DirName)

{

DIR \*dirp = NULL;

struct dirent \*Filep = NULL;

dirp = opendir(DirName);

while(1)

{

Filep = readdir(dirp);

strcpy(Buff, Filep->d\_name);

if(!strncmp(Buff, "28-", 3))

{

return Buff;

}

}

}

int main(void)

{

float temp;

int i, j, fd, ret;

char buf[BUFSIZE];

char tempBuf[5];

char fileName[256] = {0};

char Dir[256] = "/sys/bus/w1/devices/";

char \*ds18b20\_id = NULL;

ds18b20\_id = get\_ds18b20\_id(Dir);

strcpy(fileName, Dir);

strcat(fileName, ds18b20\_id);

strcat(fileName, "/w1\_slave");

printf("filename is %s\n", fileName);

while(1)

{

if((fd = open(fileName, O\_RDONLY)) == -1)

{

perror("open device file error");

return -1;

}

while(1)

{

ret = read(fd, buf, BUFSIZE);

if(0 == ret)

{

break;

}

if(-1 == ret)

{

if(errno == EINTR)

{

continue;

}

perror("read()");

close(fd);

return -1;

}

}

for(i=0;i<sizeof(buf);i++)

{

if(buf[i] == 't')

{

for(j=0;j<sizeof(tempBuf);j++)

{

tempBuf[j] = buf[i+2+j];

}

}

}

temp = (float)atoi(tempBuf) / 1000;

printf("%.3f C\n",temp);

sleep(1);

close(fd);

}

return 0;

}

Experimental phenomena

After running the program, you can read the current temperature value in the Raspberry Pi command line interface.