代码Code：

sudo modprobe w1-gpio  
sudo modprobe w1-therm  
cd /sys/bus/w1/devices  
ls

**from** sakshat **import** SAKSHAT

**import** time

SAKS **=** SAKSHAT()

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

**while** True:

    temp **=** SAKS.ds18b20.temperature

**if** temp **==** **-**120.0 :

         time.sleep(5)

**continue**

 print (("%5.1f" **%** temp).replace(' ','#'))

    SAKS.digital\_display.show(("%5.1f" **%** temp).replace(' ','#'))

    time.sleep(5)

input("Enter any keys to exit...")

import RPi.GPIO as GPIO

from sakspins import SAKSPins as PINS

import entities

class SAKSHAT(object):

SAKS HAT class, some useful function are declared.

buzzer = None

ledrow = None

ds18b20 = None

digital\_display = None

dip\_switch = None

tactrow = None

def saks\_gpio\_init(self):

GPIO.setwarnings(False)

GPIO.cleanup()

GPIO.setmode(GPIO.BCM)

GPIO.setup(PINS.BUZZER, GPIO.OUT)

GPIO.output(PINS.BUZZER, GPIO.HIGH)

for p in [PINS.IC\_TM1637\_DI, PINS.IC\_TM1637\_CLK, PINS.IC\_74HC595\_DS, PINS.IC\_74HC595\_SHCP, PINS.IC\_74HC595\_STCP]:

GPIO.setup(p, GPIO.OUT)

GPIO.output(p, GPIO.LOW)

for p in [PINS.BUZZER, PINS.TACT\_RIGHT, PINS.TACT\_LEFT, PINS.DIP\_SWITCH\_1, PINS.DIP\_SWITCH\_2]:

GPIO.setup(p, GPIO.OUT)

GPIO.output(p, GPIO.HIGH)

for p in [PINS.TACT\_RIGHT, PINS.TACT\_LEFT, PINS.DIP\_SWITCH\_1, PINS.DIP\_SWITCH\_2]:

GPIO.setup(p, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

def \_\_init\_\_(self):

self.saks\_gpio\_init()

self.buzzer = entities.Buzzer(PINS.BUZZER, GPIO.LOW)

self.ledrow = entities.Led74HC595({'ds': PINS.IC\_74HC595\_DS, 'shcp': PINS.IC\_74HC595\_SHCP, 'stcp': PINS.IC\_74HC595\_STCP}, GPIO.HIGH)

self.ds18b20 = entities.DS18B20(PINS.DS18B20)

self.digital\_display = entities.DigitalDisplayTM1637({'di': PINS.IC\_TM1637\_DI, 'clk': PINS.IC\_TM1637\_CLK}, GPIO.HIGH)

self.dip\_switch = entities.DipSwitch2Bit([PINS.DIP\_SWITCH\_1, PINS.DIP\_SWITCH\_2], GPIO.LOW)

self.dip\_switch.register(self)

self.tactrow = entities.TactRow([PINS.TACT\_LEFT, PINS.TACT\_RIGHT], GPIO.LOW)

for t in self.tactrow.items:

t.register(self)

dip\_switch\_status\_changed\_handler = None

def on\_dip\_switch\_2bit\_status\_changed(self, status):

if self.dip\_switch\_status\_changed\_handler is not None:

self.dip\_switch\_status\_changed\_handler(status)

tact\_event\_handler = None

def on\_tact\_event(self, pin, status):

if self.tact\_event\_handler is not None:

self.tact\_event\_handler(pin, status)

class SAKSPins(object):

SAKS Pins Code With BCM for Raspberry Pi.

IC\_74HC595\_DS = 6

IC\_74HC595\_SHCP = 19

IC\_74HC595\_STCP = 13

IC\_TM1637\_DI = 25

IC\_TM1637\_CLK = 5

BUZZER = 12

TACT\_RIGHT = 20

TACT\_LEFT = 16

DIP\_SWITCH\_1 = 21

DIP\_SWITCH\_2 = 26

IR\_SENDER = 17

IR\_RECEIVER = 9

DS18B20 = 4

UART\_TXD = 14

UART\_RXD = 15

I2C\_SDA = 2

I2C\_SLC = 3

Run：

git clone https://github.com/spoonysonny/SAKS-tutorials.git

cd SAKS-tutorials

cd temperature-display-ds18b20

sudo python main.py