Four Digital Seven Segment

Overview

Now, let’ s try to control more Digit 7-segment display

Experimental Materials:

Raspberry Pi *1

T-type expansion board *1

Breadboard*1

4-Digit 7-segment display *1

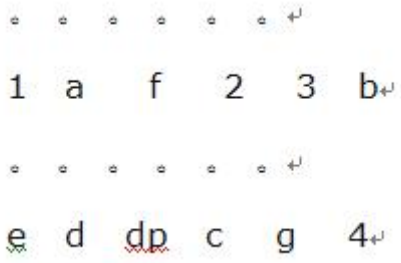
Some DuPont lines

Product description：

● Function: The segments are Light Emitting Diodes and they therefore need a series resistance to prevent burning out.

● Application:Digital display is widely used in instruments, clocks, displays and so on.

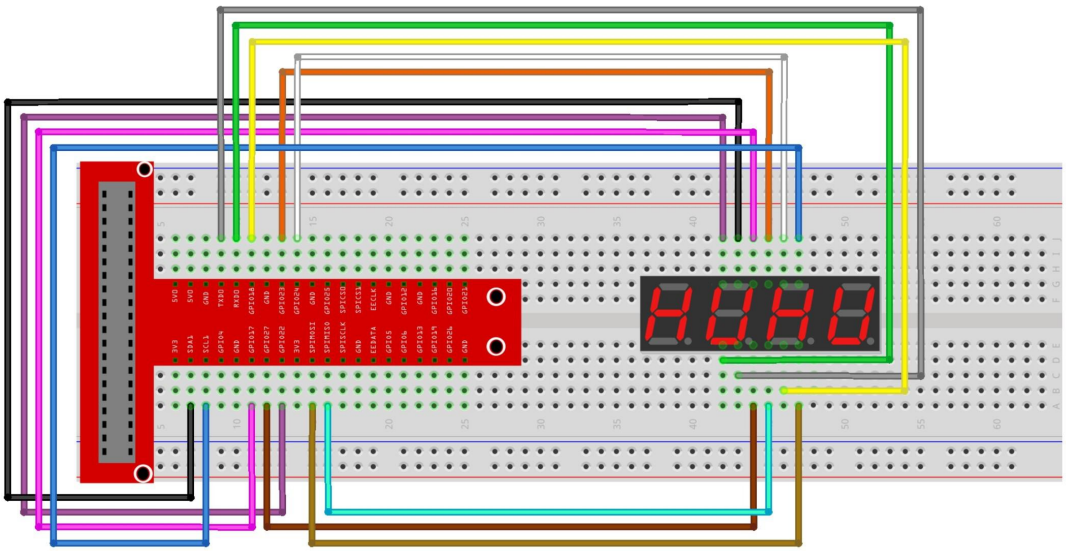
The pin description is as follows:



Technical Parameters:

Type: common anode
Size: 30mm * 14mm * 7.2mm (L*W*T)
Luminous color: Highlight-red

Wiring diagram：



C code：

```
#include <wiringPi.h>
#include <stdio.h>
//display 1234
//Set cathode interface
int a = 8;
int b = 9;
int c = 13;
int d = 15;
int e = 16;
int f = 0;
int g = 1;
int dp = 2;
//Set anode interface
int d4 = 12;
int d3 = 5;
int d2 = 4;
int d1 = 3;
//Set variable
long n = 1230;
int x = 100;
```

```
int del = 55;  //Here to fine tune the clock
void init()
{
    pinMode(d1, OUTPUT);
    pinMode(d2, OUTPUT);
    pinMode(d3, OUTPUT);
    pinMode(d4, OUTPUT);
    pinMode(a, OUTPUT);
    pinMode(b, OUTPUT);
    pinMode(c, OUTPUT);
    pinMode(d, OUTPUT);
    pinMode(e, OUTPUT);
    pinMode(f, OUTPUT);
    pinMode(g, OUTPUT);
    pinMode(dp, OUTPUT);
}
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
void bitSelect(unsigned char n)//
{
    switch(n)
     {
    case 1:
      digitalWrite(d1,HIGH);
      digitalWrite(d2, LOW);
      digitalWrite(d3, LOW);
      digitalWrite(d4, LOW);
     break;
     case 2:
      digitalWrite(d1, LOW);
      digitalWrite(d2, HIGH);
      digitalWrite(d3, LOW);
      digitalWrite(d4, LOW);
        break;
      case 3:
        digitalWrite(d1,LOW);
       digitalWrite(d2, LOW);
       digitalWrite(d3, HIGH);
       digitalWrite(d4, LOW);
        break;
      case 4:
       digitalWrite(d1, LOW);
       digitalWrite(d2, LOW);
```

```
        digitalWrite(d3, LOW);
        digitalWrite(d4, HIGH);
         break;
         default :
            digitalWrite(d1, LOW);
        digitalWrite(d2, LOW);
        digitalWrite(d3, LOW);
        digitalWrite(d4, LOW);
         break;
       }
}
void Num_0()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, LOW);
  digitalWrite(f, LOW);
  digitalWrite(g, HIGH);
  digitalWrite(dp,HIGH);
}
void Num_1()
{
  digitalWrite(a, HIGH);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  digitalWrite(dp,HIGH);
}
void Num_2()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, HIGH);
  digitalWrite(d, LOW);
  digitalWrite(e, LOW);
  digitalWrite(f, HIGH);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
```

```
void Num_3()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
void Num_4()
{
  digitalWrite(a, HIGH);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, LOW);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
void Num_5()
{
  digitalWrite(a, LOW);
  digitalWrite(b, HIGH);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, HIGH);
  digitalWrite(f, LOW);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
void Num_6()
{
  digitalWrite(a, LOW);
  digitalWrite(b, HIGH);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, LOW);
  digitalWrite(f, LOW);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
```

```
void Num_7()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  digitalWrite(dp,HIGH);
}
void Num_8()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, LOW);
  digitalWrite(f, LOW);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
void Num_9()
{
  digitalWrite(a, LOW);
  digitalWrite(b, LOW);
  digitalWrite(c, LOW);
  digitalWrite(d, LOW);
  digitalWrite(e, HIGH);
  digitalWrite(f, LOW);
  digitalWrite(g, LOW);
  digitalWrite(dp,HIGH);
}
void Clear()  // Clear the screen
{
  digitalWrite(a, HIGH);
  digitalWrite(b, HIGH);
  digitalWrite(c, HIGH);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  digitalWrite(dp,HIGH);
}
```

```
void pickNumber(unsigned char n)//Choose the number of
{
  switch(n)
  {
   case 0:Num_0();
   break;
   case 1:Num_1();
   break;
   case 2:Num_2();
   break;
   case 3:Num_3();
   break;
   case 4:Num_4();
   break;
   case 5:Num_5();
   break;
   case 6:Num_6();
   break;
   case 7:Num_7();
   break;
   case 8:Num_8();
   break;
   case 9:Num_9();
   break;
   default:Clear();
   break;
  }
}
void Display(unsigned char x, unsigned char Number)//Show that x is the
coordinate, Number is the number
{
    bitSelect(x);
    pickNumber(Number);
    delay(1);
    //Clear() ; //Vanishing
}

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

```
    init();
    while(1)
    {
        Display(1, 1);
        delay(1000);
        Display(2, 2);
        delay(1000);
        Display(3, 3);
        delay(1000);
        Display(4, 4);
        delay(1000);
    }



}
```

Python code：

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


pins = {'pinA':3, 'pinB':5, 'pinC':21, 'pinD':8, 'pinE':10, 'pinF':11,
'pinG':12, 'pinDP':13, 'pin_1':15, 'pin_2':16, 'pin_3':18, 'pin_4':19}


def init():
    GPIO.setmode(GPIO.BOARD)
    for i in pins:
        GPIO.setup(pins[i], GPIO.OUT)
    print 'gpio init completed!'

def bitSelect(bitNum):
    if(bitNum == 1):
        GPIO.output(pins['pin_1'], GPIO.HIGH)
        GPIO.output(pins['pin_2'], GPIO.LOW)
        GPIO.output(pins['pin_3'], GPIO.LOW)
        GPIO.output(pins['pin_4'], GPIO.LOW)
    elif(bitNum == 2):
```

```
        GPIO.output(pins['pin_1'], GPIO.LOW)
        GPIO.output(pins['pin_2'], GPIO.HIGH)
        GPIO.output(pins['pin_3'], GPIO.LOW)
        GPIO.output(pins['pin_4'], GPIO.LOW)
    elif(bitNum == 3):
        GPIO.output(pins['pin_1'], GPIO.LOW)
        GPIO.output(pins['pin_2'], GPIO.LOW)
        GPIO.output(pins['pin_3'], GPIO.HIGH)
        GPIO.output(pins['pin_4'], GPIO.LOW)
    elif(bitNum == 4):
        GPIO.output(pins['pin_1'], GPIO.LOW)
        GPIO.output(pins['pin_2'], GPIO.LOW)
        GPIO.output(pins['pin_3'], GPIO.LOW)
        GPIO.output(pins['pin_4'], GPIO.HIGH)
    else:
        GPIO.output(pins['pin_1'], GPIO.LOW)
        GPIO.output(pins['pin_2'], GPIO.LOW)
        GPIO.output(pins['pin_3'], GPIO.LOW)
        GPIO.output(pins['pin_4'], GPIO.LOW)
    print 'bitSelect completed!'


def display_0():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.LOW)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.HIGH)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 0'

def display_1():
    GPIO.output(pins['pinA'], GPIO.HIGH)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.HIGH)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.HIGH)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 1'
```

```
def display_2():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.HIGH)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.LOW)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 2'

def display_3():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 3'

def display_4():
    GPIO.output(pins['pinA'], GPIO.HIGH)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.HIGH)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 4'

def display_5():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.HIGH)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 5'
```

```
def display_6():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.HIGH)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.LOW)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 6'


def display_7():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.HIGH)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.HIGH)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 7'

def display_8():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.LOW)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.LOW)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 8'


def display_9():
    GPIO.output(pins['pinA'], GPIO.LOW)
    GPIO.output(pins['pinB'], GPIO.LOW)
    GPIO.output(pins['pinC'], GPIO.LOW)
    GPIO.output(pins['pinD'], GPIO.LOW)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.LOW)
    GPIO.output(pins['pinG'], GPIO.LOW)
```

```
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'display number 9'

def display_dp():
    GPIO.output(pins['pinA'], GPIO.HIGH)
    GPIO.output(pins['pinB'], GPIO.HIGH)
    GPIO.output(pins['pinC'], GPIO.HIGH)
    GPIO.output(pins['pinD'], GPIO.HIGH)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.HIGH)
    GPIO.output(pins['pinDP'], GPIO.LOW)
    print 'display DP'


def clear():   #clear the screen
    GPIO.output(pins['pinA'], GPIO.HIGH)
    GPIO.output(pins['pinB'], GPIO.HIGH)
    GPIO.output(pins['pinC'], GPIO.HIGH)
    GPIO.output(pins['pinD'], GPIO.HIGH)
    GPIO.output(pins['pinE'], GPIO.HIGH)
    GPIO.output(pins['pinF'], GPIO.HIGH)
    GPIO.output(pins['pinG'], GPIO.HIGH)
    GPIO.output(pins['pinDP'], GPIO.HIGH)
    print 'clear the screen!'

def pickNum(number):
    if(number == 0):
        display_0()
    elif(number == 1):
        display_1()
    elif(number == 2):
        display_2()
    elif(number == 3):
        display_3()
    elif(number == 4):
        display_4()
    elif(number == 5):
        display_5()
    elif(number == 6):
        display_6()
    elif(number == 7):
        display_7()
    elif(number == 8):
```

```
        display_8()
    elif(number == 9):
        display_9()
    else:
        clear()


def Display(Bit, Number):
    bitSelect(Bit)
    pickNum(Number)
    time.sleep(0.001)

def loop():
    while True:
        Display(1,1)
        time.sleep(1)
        Display(2,2)
        time.sleep(1)
        Display(3,3)
        time.sleep(1)
        Display(4,4)
        time.sleep(1)

if __name__ == '__main__':
    try:
        init()
                loop()
    except KeyboardInterrupt:
        GPIO.cleanup()
        print 'Key Board Interrupt!'
```

Experimental results:

In the directory where the code file is located, execute the following command

C：

```
gcc -Wall -o 4digitLEDdisplay 4digitLEDdisplay.c -lwiringPi
sudo ./4digitLEDdisplay
```

Python:

```
python 4digitLEDdisplay.py
```

After the instruction is executed, The four digits display the number 1234 cyclically.