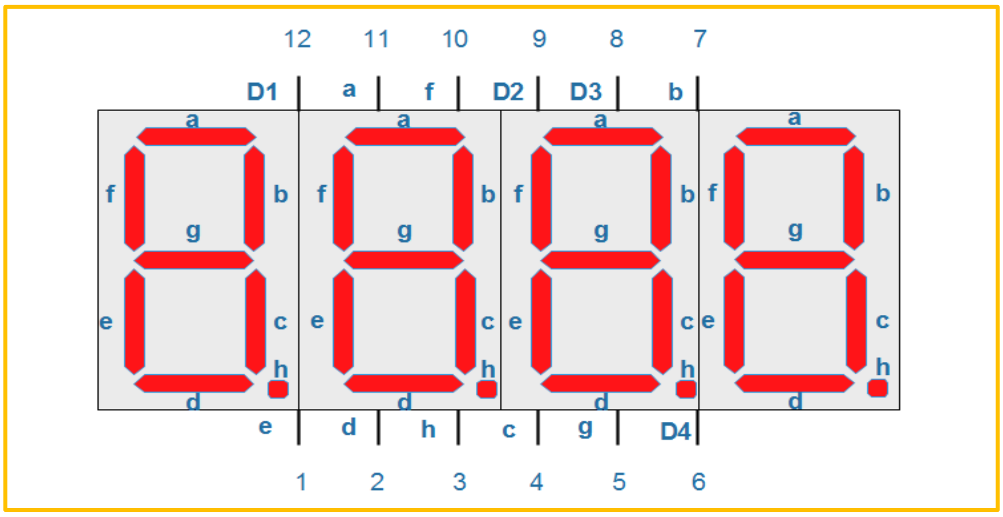
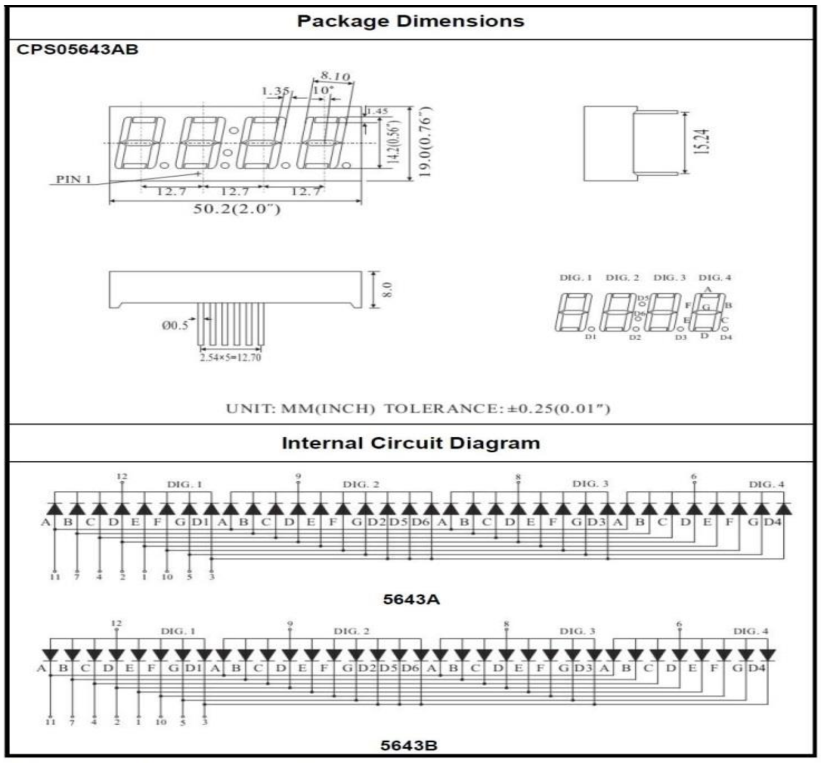
4-digit 7-segment digital tube display experiment

Introduction

We have used 7-segment digital tubes before, so we know that when we want to display multiple numbers, we need a multi-digit digital tube. Next, we will introduction 4-digit 7-segment digital tubes. In fact, each 7-segment digital tube is almost the same as the tube used before. In this experiment, we will use Raspberry Pi as the main board to drive 4-digit 7-segment common anode digital tubes.





Four Digits Displays Series

This digital tube has a total of 12 pins and the pin number in the upper left corner is 12. Eight of its pins are used to light up the “abcdefgh”segment tube, and the remaining four pins D1, D2, D3 and D4 are used to represent “bit” pins. When the corresponding “bit” pin of these four digital tubes is high level voltage, it will be lit. The display principle of these four digital tubes is to continuously scan the D1, D2, D3 and D4 pins, and then light the corresponding eight-segment tubes in sequence. Since the speed of lighting up the digital tubes is so fast that humans eyes can not see it, so it looks like four digital tubes are displayed at the same time.

Based on the principles introduced above, we will now make a simulated countdown time bomb similar to the movie. This bomb will explode in a minute.

Experimental principle

The most import purpose of this program is how to dynamically scan these four digital tubes. In fact, from the single digital tube display experiment we have done before, it is quite easy to realize the simultaneous display of these four digital tubes. As it is a common anode digital tube, first of, we set D1，D2，D3 and D4 to low level voltage and all of the LEDs will be off, and then we output the truth table of “abcdefgh” to the corresponding GPIO port, and select the corresponding bit pin are scanned continuously. How to achieve the 1 minute countdown effect? We delay for about 1 second, subtract 1 from the countdown time, and then constantly refresh the time displayed on this digital tube.

Experimental purpose

The aim of this experiment is to cyclically display the numbers 0-60 by dynamically scanning a 4-digit 7-segment digital tube.

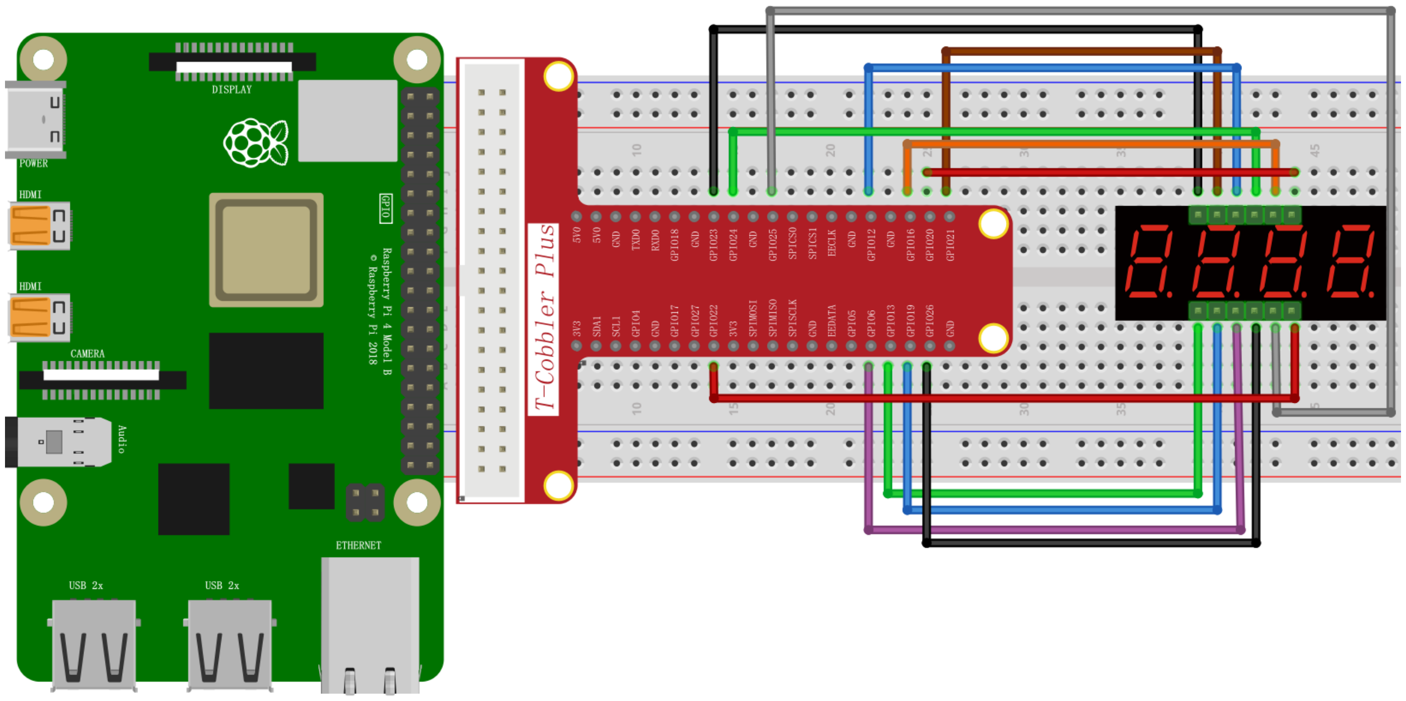
Component list

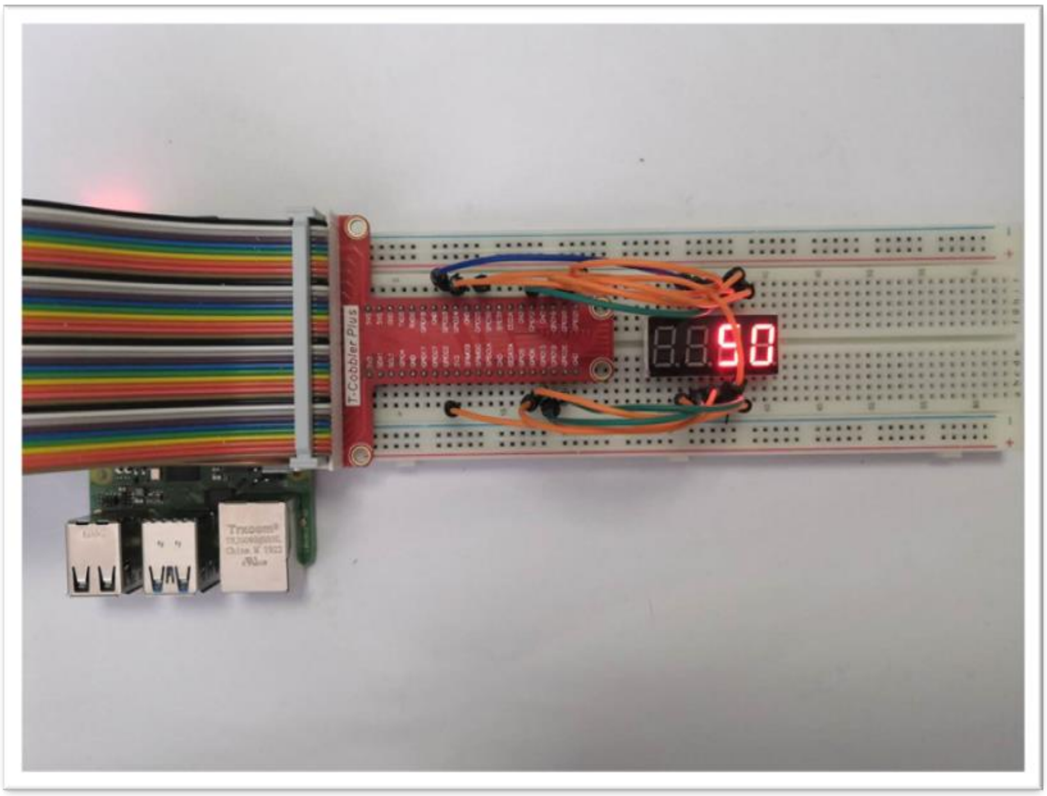
- Raspberry Pi main board
- T-Cobbler Plus expansion board
- Cable
- 4-digit 7-segment digital tube * 1
- 1k Resistor * 12
- Several jumper wires

Wiring

Raspberry Pi Board	Digital Tube
IO29(wiringPi)/21(BCM)	(a)
IO28(wiringPi)/20(BCM)	(b)

IO25(wiringPi)/26(BCM)	(c)
IO24(wiringPi)/19(BCM)	(d)
IO23(wiringPi)/13(BCM)	(e)
IO26(wiringPi)/12(BCM)	(f)
IO6(wiringPi)/25(BCM)	(g)
IO22(wiringPi)/6(BCM)	(h)
IO4(wiringPi)/23(BCM)	(d1)
IO5(wiringPi)/24(BCM)	(d2)
IO27(wiringPi)/16(BCM)	(d3)
IO3(wiringPi)/22(BCM)	(d4)





C++ partial program

```
#include <iostream>
#include <wiringPi.h>
#include "SegmentDisplay.h"
using namespace std;

#define  LED_A   29
#define  LED_B   28
#define  LED_C   25
#define  LED_D   24
#define  LED_E   23
#define  LED_F   26
#define  LED_G   6
#define  LED_H   22
#define  LED_D1  4
#define  LED_D2  5
#define  LED_D3  27
#define  LED_D4  3

int main ()
{
    wiringPiSetup();
    SegmentDisplay _4Bit_7SegmentDisplay(LED_A, LED_B, LED_C, LED_D, LED_E, LED_F, LED_G,
LED_H, LED_D1, LED_D2, LED_D3, LED_D4);

    int ShowTime = 0, count = 0;
    while(1)
    {
       if (ShowTime > 60 )
       {
       ShowTime = 0 ;
       }
       _4Bit_7SegmentDisplay.DisplayChar((int)ShowTime);
       ShowTime++ ;
    }
}
```

Python program

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time
BIT0 = 22
BIT1 = 16
BIT2 = 24
BIT3 = 23

segCode = [0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90]  #0~9
pins = [21, 20, 26, 19, 13, 12, 25, 6, 22, 16, 24, 23]
bits = [BIT0, BIT1, BIT2, BIT3]

def print_msg():
  print ('Program is running...')
  print ('Please press Ctrl+C end the program...')

def digitalWriteByte(val):
  GPIO.output(21, val & (0x01 << 0))
  GPIO.output(20, val & (0x01 << 1))
  GPIO.output(26, val & (0x01 << 2))
  GPIO.output(19, val & (0x01 << 3))
  GPIO.output(13, val & (0x01 << 4))
  GPIO.output(12, val & (0x01 << 5))
  GPIO.output(25, val & (0x01 << 6))
  GPIO.output(6, val & (0x01 << 7))
```

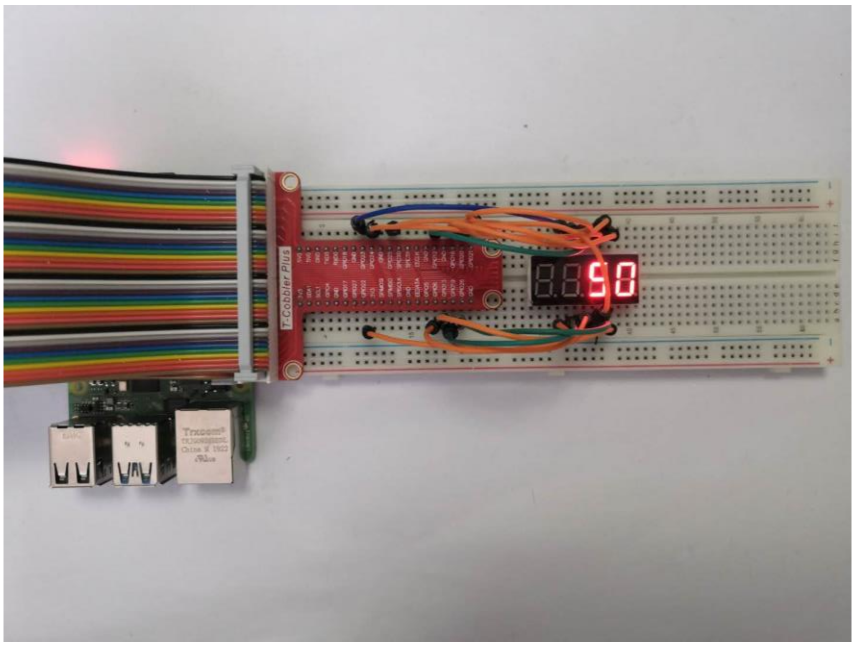
```
def hide():
  GPIO.output(BIT0, GPIO.LOW)
  GPIO.output(BIT1, GPIO.LOW)
  GPIO.output(BIT2, GPIO.LOW)
  GPIO.output(BIT3, GPIO.LOW)

def show():
  GPIO.output(BIT0, GPIO.HIGH)
  GPIO.output(BIT1, GPIO.HIGH)
  GPIO.output(BIT2, GPIO.HIGH)
  GPIO.output(BIT3, GPIO.HIGH)

def showNum(bit,num):
  hide()
  GPIO.output(bits[bit], GPIO.HIGH)
  digitalWriteByte(segCode[num])
  time.sleep(0.025)

def display_3(num):
  b0 = num % 10
  b1 = int(num % 100 / 10 )
  b2 = int(num % 1000 / 100)
  b3 = int(num / 1000)

  if num < 10:
    showNum(0,b0)
  elif num >= 10 and num < 100:
    showNum(0, b0)
    showNum(1, b1)
  elif num >= 100 and num < 1000:
    showNum(0, b0)
    showNum(1, b1)
    showNum(2, b2)
  elif num >= 1000 and num < 10000:
    showNum(0, b0)
    showNum(1, b1)
    showNum(2, b2)
    showNum(3, b3)
  else:
    print ('Out of range, num should be 0~9999 !')

def setup():
  #GPIO.setmode(GPIO.BOARD)    #Number GPIOs by its physical location
  GPIO.setmode(GPIO.BCM)
  for pin in pins:
    GPIO.setup(pin, GPIO.OUT)    #set all pins' mode is output
    GPIO.output(pin, GPIO.HIGH)  #set all pins are high level(3.3V)

def loop():
  while True:
    print_msg()
    for i in range(60):
        display_3(i)

def destroy():   #When program ending, the function is executed.
  for pin in pins:
    GPIO.output(pin, GPIO.LOW) #set all pins are low level(0V)
    GPIO.setup(pin, GPIO.IN)   #set all pins' mode is input

if __name__ == '__main__': #Program starting from here
  setup()
  try:
    loop()
  except KeyboardInterrupt:
    destroy()
```

Java program

```
import com.pi4j.wiringpi.Gpio;
import java.util.List;
import java.util.ArrayList;
import java.util.Scanner;
enum MODE
{
    SEGMENT_DISPLY_1BIT,
    SEGMENT_DISPLY_4BIT,
    SEGMENT_DISPLY_8BIT;
}

    public class Digital_Tube {
        char dight_pin[] = {29, 28, 25, 24, 23, 26, 6, 22};
        char dight_display[] = {0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90,
0x00};
```

```
        char dight_select_pin[] = {4, 5, 27, 3};
        List list = new ArrayList();
        MODE mode;

    static {
        Gpio.wiringPiSetup();
    }

    void SegmentDisplay() {
        mode = MODE.SEGMENT_DISPLY_1BIT;
        for (int i = 0; i < 8; i++) {
            Gpio.pinMode(dight_pin[i], Gpio.OUTPUT);  //set all led diplay array pin output
mode
            Gpio.digitalWrite(dight_pin[i], Gpio.HIGH);
        }
    }

    void SegmentDisplay(int segment) {
        mode = MODE.SEGMENT_DISPLY_4BIT;
        for (int i = 0; i < 8; i++) {
            Gpio.pinMode(dight_pin[i], Gpio.OUTPUT);  //set all led diplay array pin output
mode
            Gpio.digitalWrite(dight_pin[i], Gpio.LOW);
        }
        for (int i = 0; i < 4; i++) {
            Gpio.pinMode(dight_select_pin[i], Gpio.OUTPUT);  //set all led diplay array pin
output mode
            Gpio.digitalWrite(dight_select_pin[i], Gpio.LOW);
        }
    }

    void TurnOffAllLed() {
        for (int i = 0; i < 8; i++)
            Gpio.digitalWrite(dight_pin[i], Gpio.LOW);
        if (mode == MODE.SEGMENT_DISPLY_4BIT) {
            for (int i = 0; i < 4; i++)
                Gpio.digitalWrite(dight_select_pin[i], Gpio.LOW);
        }
    }

    void numble2dis(int numble) {
        int numble_bit = 0;
        int bit_base = 1000;
        for (numble_bit = 0; numble_bit < 4; numble_bit++ ) {
            if (numble/bit_base != 0) {
                list.add(numble_bit, numble/bit_base);
                numble = numble%bit_base;
            } else {
                list.add(numble_bit, 0);
            }
            bit_base = bit_base / 10;
        }
    }

    void Display_One_Char(int n) {
        char ch = dight_display[n];
        if (n < 10) {
            for (int i = 0; i < 8; i++) {
                if ((ch & (1 << i)) == 0) {
                    Gpio.digitalWrite(dight_pin[i], Gpio.LOW);
                } else {
                    Gpio.digitalWrite(dight_pin[i], Gpio.HIGH);
                }
            }
        }
    }

    void Display_Four_Char(int n) {
        char ch = dight_display[n];
        if (n < 10) {
            for (int i = 0; i < 8; i++) {
                if ((ch & (1 << i)) == 0) {
                    Gpio.digitalWrite(dight_pin[i], Gpio.HIGH);
                } else {
                    Gpio.digitalWrite(dight_pin[i], Gpio.LOW);
                }
            }
        }
    }

    void DisplayChar(int n) {
        if (mode == MODE.SEGMENT_DISPLY_4BIT) {
            numble2dis(n);
            for(int i = 0; i < 4; i++) {
                Display_Four_Char((int)list.get(i));
                Gpio.digitalWrite(dight_select_pin[i], Gpio.LOW);
                Gpio.delay(5);
                Gpio.digitalWrite(dight_select_pin[i], Gpio.HIGH);
            }
        }
    }

    void DisplayChar(char sel, char n) {
        char ch = dight_display[n];
        for (int i = 0; i < 8; i++) {
            if ((ch & (1 << i)) == 0) {
                Gpio.digitalWrite(dight_pin[i], Gpio.HIGH);
            } else {
                Gpio.digitalWrite(dight_pin[i], Gpio.LOW);
            }
        }
        Gpio.digitalWrite(dight_select_pin[sel] , Gpio.HIGH);
    }

    public static void main(String[] args) {
        Digital_Tube digital_tube = new Digital_Tube();
        digital_tube.SegmentDisplay(4);
        int ShowTime = 0, count = 0;
        while (true) {
            for (int i = 60; i >= 0; i--) {
                if (ShowTime > 60 ) {
                ShowTime = 0 ;
                }
                digital_tube.DisplayChar(ShowTime);
                ShowTime++ ;
            }
        }
    }
}
```

Experiment effect



You will see the numbers 0-60 cycled displayed on the digital tube.