4 Bit Digital Tube

Overview

We will learn to use the Raspberry Pi to control the Four digit tube to show numbers 1234 in circle.

Materials Needed

RaspberryPi \*1

Breadboard \*1

4bit\_digital\_tube \*1

Dupont Line

Preparatory 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

See the attached <<Installing a Python Interpreter and Corresponding Libraries in a Raspberry Pi System>> for details.

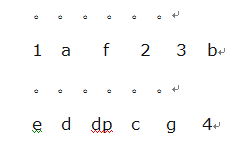
Product Description

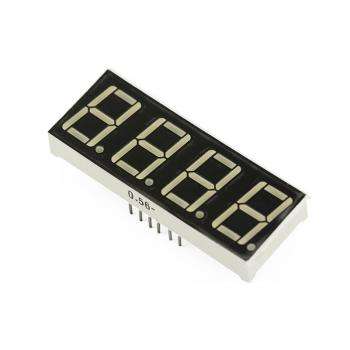
Brief Introduction

Function:The digital tube is a semiconductor light emitting device whose basic unit is a light emitting diode. Therefore, series resistance is also needed to prevent excessive current.

Application: Digital tube is widely used in instruments, clocks, stations, etc.

Pin description:





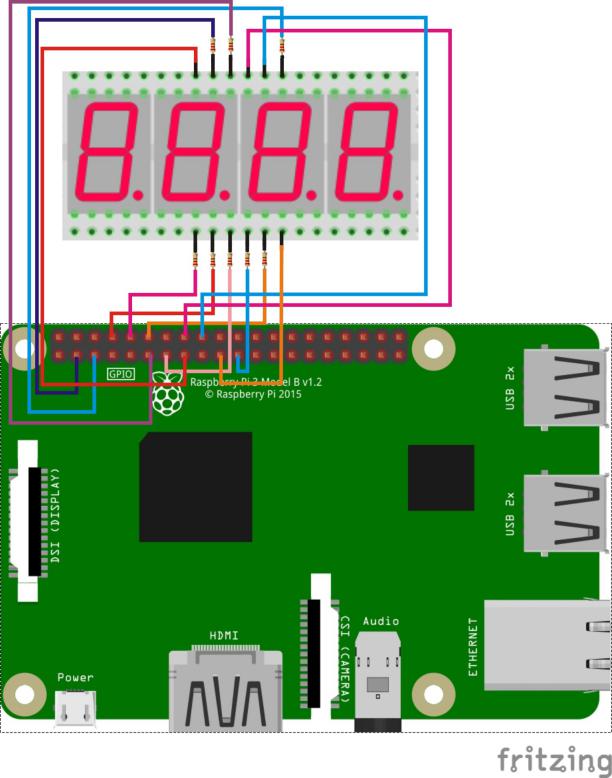
Parameters

◆ Model: common anode

◆ Size: length 30mm\* width 14mm\* thickness 7.2mm

◆ lighting color: bright red

Wiring diagram



Sample Code

1. 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!'

2. 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);

}

}

Experimental Phenomena

The four-bit digital tube shows the number 1234 in crycle.