KUMAN
www.kumantech.com

4 Digit LED Display

(aka 4-Bit Digital Tube)

Overview

“4-Bit Digital Tube” is the conventional name for a display capable of showing four decimal digits, where any possible digit, in turn, is composed of up to eight separate LED segments. (A decimal point potentially trailing each digit represents an eighth possible LED in that digit display.) Such low-cost, low-power numeric displays are common in microwave ovens, alarm clocks, induction cookers, automatic washing machines, and similar devices. In this experiment, you’ll program the Raspberry Pi to show the digits 0 through 9, sequentially, on the LED display.

Experimental Materials

Raspberry Pi	x1
Breadboard	x1
4 Digit LED Display	x1
Resistors (330Ω)	x8
Dupont jumper wires	

Experimental Procedure

1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GPIO library, and wiringPi library as described in READ_ME_FIRST.TXT.

2. Install the 4 Digit LED Display on your breadboard, and use Dupont jumper wires and resistors to connect it to your Raspberry Pi as illustrated in the Wiring Diagram below. The resistors connect to the eight LED pins (**A-G**, **DP**) and protect them from the current of the Raspberry Pi GPIO pins. The other four pins (**1-4**) do not connect to fragile LEDs and therefore do not need resistors.

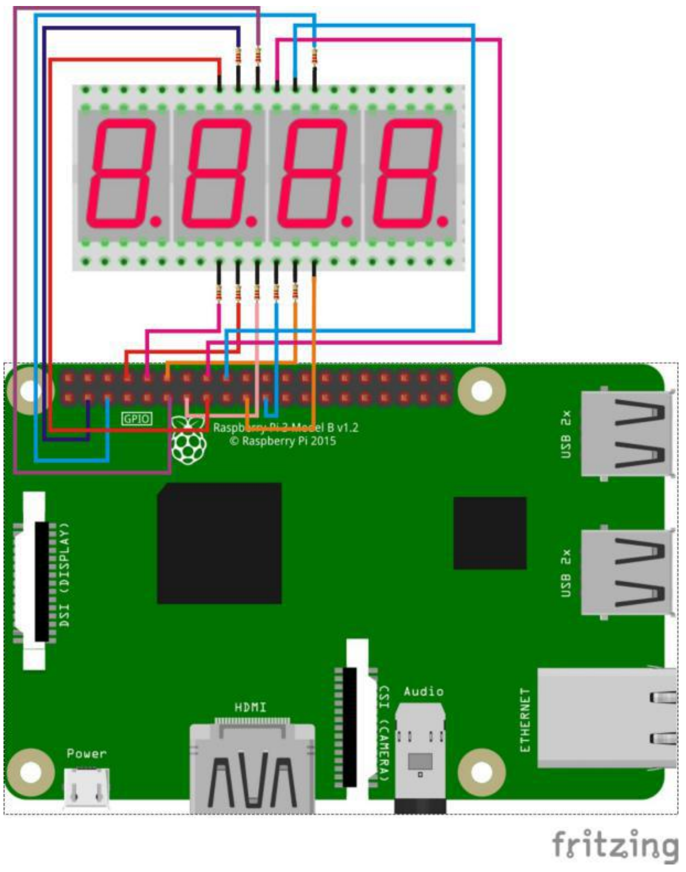
3. Execute the sample stored in this experiment's subfolder.
   If using C, compile and execute the C code:
   ```
   cd Code/C
   gcc 4digitLEDdisplay.c -o 4digitLEDdisplay.out -lwiringPi
   ./4digitLEDdisplay.out
   ```

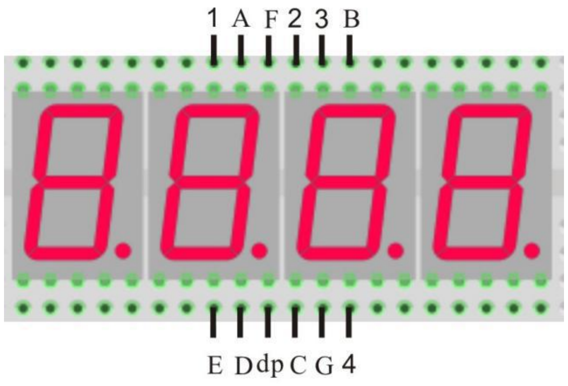
   If using Python, launch the Python script:

   ```
   cd Code/Python
   python 4digitLEDdisplay.py
   ```

4. Make experimental observations. The multidigit LED display shows the digits "1234" in a cycle.

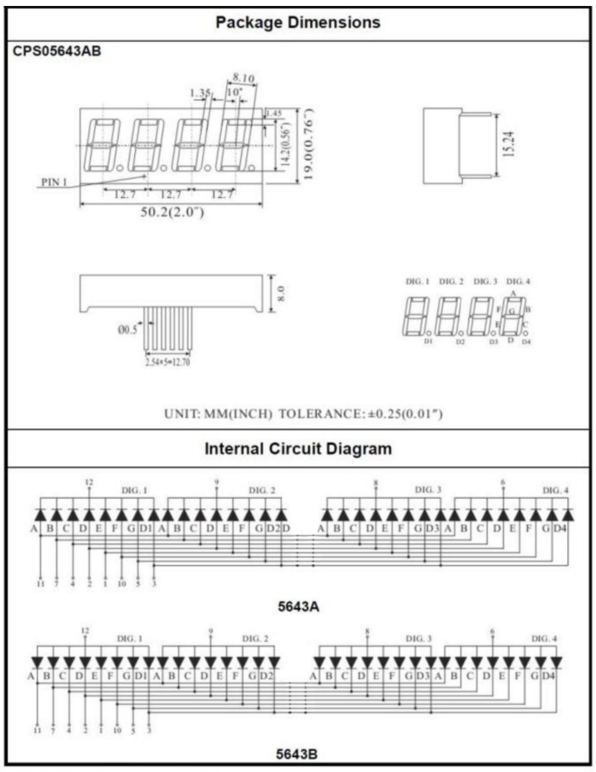
Wiring Diagram





4 Digit LED Display pin position:

'A'	↔	Raspberry Pi pin 3 (through resistor)
'B'	↔	Raspberry Pi pin 5 (through resistor)
'C'	↔	Raspberry Pi pin 21 (through resistor)
'D'	↔	Raspberry Pi pin 8 (through resistor)
'E'	↔	Raspberry Pi pin 10(through resistor)
'F'	↔	Raspberry Pi pin 11 (through resistor)
'G'	↔	Raspberry Pi pin 12 (through resistor)
'DP'	↔	Raspberry Pi pin 13 (through resistor)
'1'	↔	Raspberry Pi pin 15
'2'	↔	Raspberry Pi pin 16
'3'	↔	Raspberry Pi pin 18
'4'	↔	Raspberry Pi pin 19



Four Digits Displays Series

Technical Background

◆ Model: common anode

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

◆ lighting color: bright red

Logical control of the display. To set the display, the controlling circuit or microcontroller logic targets a single digit at a time. First raise one of the four digit pins (**1-4**) HIGH will setting the other three to LOW to *selects* the digit whose display will be controlled by the eight segment pins. Pin **1** selects the leftmost digit and pin **4** the rightmost digit. Then choose which of these eight-segment pins (**A-G**, **dp**) to display in the selected digit by holding a segment pin LOW to illuminate it, or HIGH to turn it off.

To set display all four digits with different values simultaneously, the four-digit display relies on *persistence of vision*. Control logic should set the first digit and delay a millisecond or two; then set the second digit and delay again; then the third digit and delay; and subsequently the fourth digit and delay; and then immediately repeat. In this process, each digit is actually “set” (illuminated) only ¼ of the total display time between refreshes, but this is sufficient illumination to guarantee all of them persist in the human eyes. Thus though the control logic can target only a single digit at a time, the LED display effectively represents multiple digits simultaneously.

Sample Code

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!'
```

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;
  }
}
```

```
//Show that x is the coordinate, Number is the number
void Display(unsigned char x, unsigned char 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);
    }
}
```