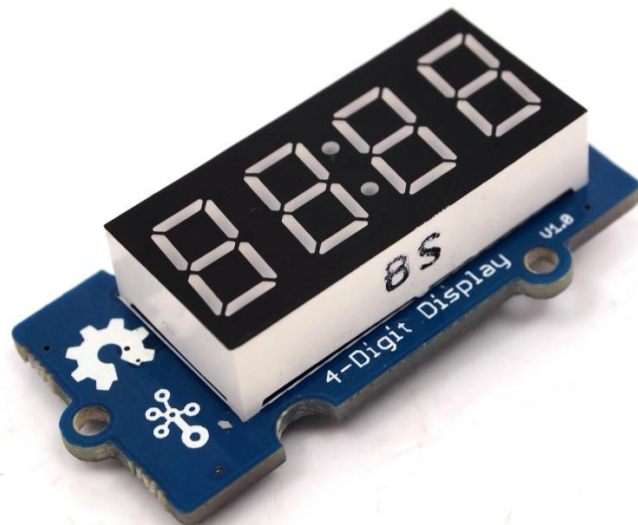


Grove - 4-Digit Display



Grove - 4-Digit Display module is a 12-pin module. In this module, we utilise a TM1637 to scale down the number of controlling pins to 2. That is to say, it controls both the content and the luminance via only 2 digital pins of Arduino or Seeeduino. For projects that require alpha-numeric display, this can be a nice choice.

Version

Product Version	Changes	Released Date
Grove - 4-Digit Display V1.0	Initial	May 2012

Features

- 4 digit red alpha-numeric display
- Grove compatible interface (3.3V/5V)
- 8 adjustable luminance levels

Tip

More details about Grove modules please refer to [Grove System](#)

Specifications

Item	Min	Typical	Max	Unit
Voltage	3.3	5.0	5.5	VDC
Current	0.2	27	80	mA
Dimensions	42x24x14			mm
Net Weight	7±1			g

Application Ideas

- Time display
- Stopwatch
- Sensors' input display

Platforms Supported

Arduino	Raspberry Pi	BeagleBone	Wio	LinkIt ONE
				

Caution

The platforms mentioned above as supported is/are an indication of the module's software or theoretical compatibility. We only provide software library or code examples for Arduino platform in most cases. It is not possible to provide software library / demo code for all possible MCU platforms. Hence, users have to write their own software library.

Getting Started




Note

If this is the first time you work with Arduino, we firmly recommend you to see [Getting Started with Arduino](#) before the start.

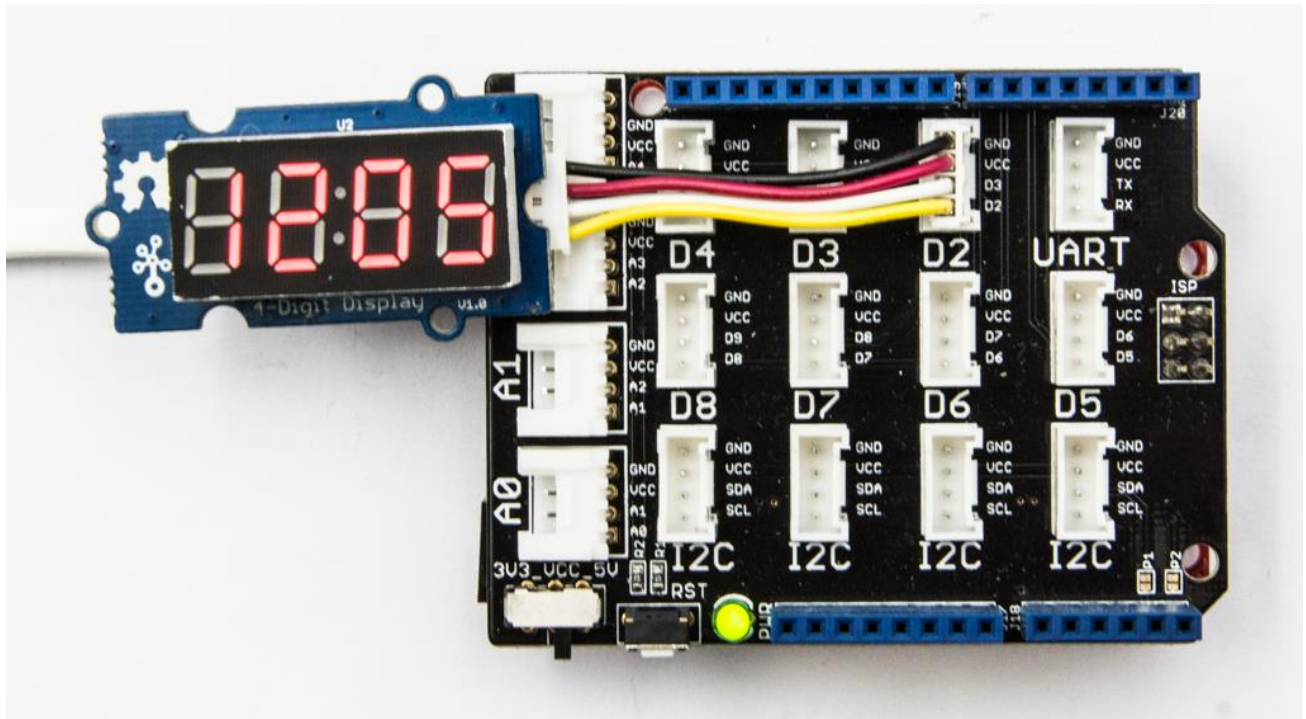
Play With Arduino

Hardware

- **Step 1.** Prepare the below stuffs:

Seeeduino V4.2	Base Shield	Grove-4-Digit Display
		

- **Step 2.** Connect Grove-4-Digit Display to **D2** port of Grove-Base Shield.
- **Step 3.** Plug Grove - Base Shield into Seeeduino.
- **Step 4.** Connect Seeeduino to PC via a USB cable.



Note

If we don't have Grove Base Shield, We also can directly connect Grove-4-Digit Display to Seeeduino as below. We also can plug Grove-4-Digit Display to other Grove digital port.

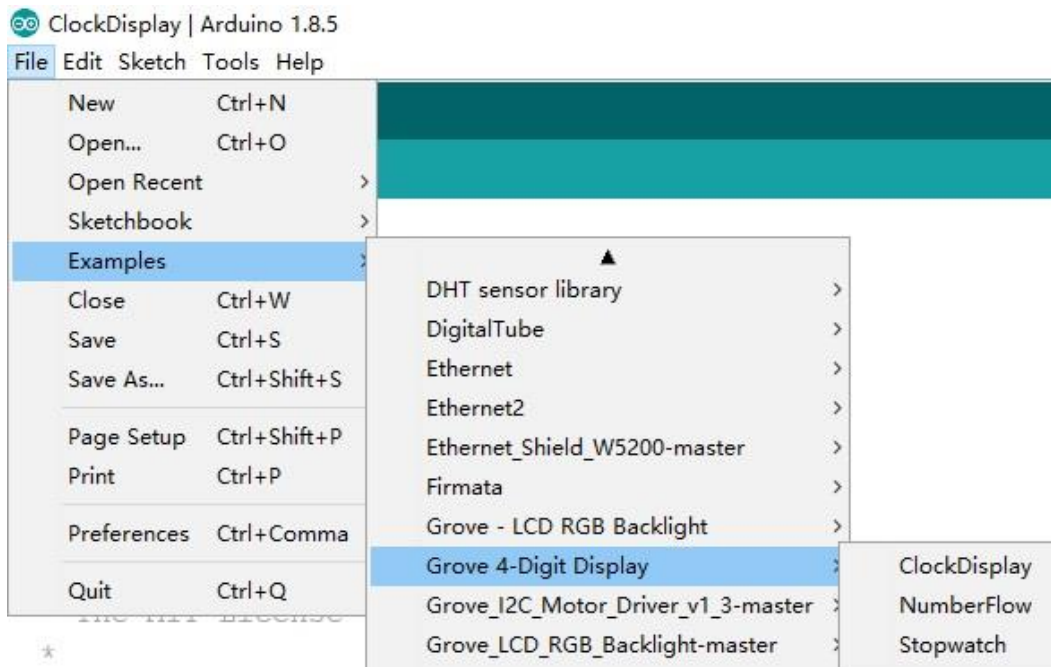
Seeeduino	Grove-4-Digit Display
5V	Red
GND	Black
D3	White (DIO)
D2	Yellow(CLK)

Warning

The Grove-4-Digit Display includes 4 pins, GND, VCC, DIO, CLK. We can connect DIO and CLK to any digital pin. It is not I2C protocol.

Software

- **Step 1.** Download the [Grove-4-Digit Display Library](#) and [TimerOne Library](#).
- **Step 2.** Refer [How to install library](#) to install library for Arduino.
- **Step 3.** Follow below instructions to select code into Arduino IDE and upload. If you do not know how to upload the code, please check [how to upload code](#). There are 3 examples as below.
 - Clock Display
 - Number Flow
 - Stop Watch



- **Step 4.** We will see the Grove-4-Digit Display being turned on.

Play with Codecraft

Hardware

Step 1. Connect Grove - 4-Digit Display to port D2 in a Base Shield

Step 2. Plug the Base Shield to your Seeeduino/Arduino.

Step 3. Link Seeeduino/Arduino to your PC via an USB cable.

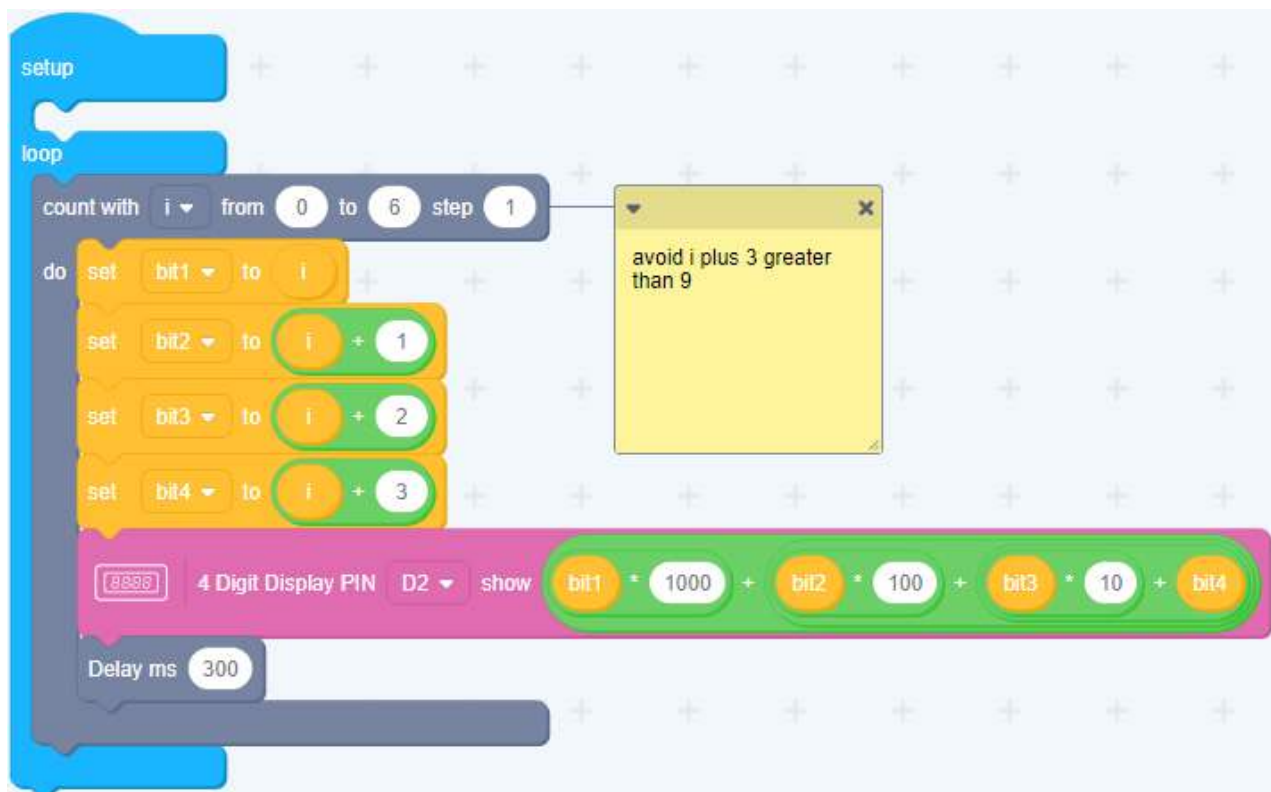
Software

Step 1. Open [Codecraft](#), add Arduino support, and drag a main procedure to working area.

Note

If this is your first time using Codecraft, see also [Guide for Codecraft using Arduino](#).

Step 2. Drag blocks as picture below or open the cdc file which can be downloaded at the end of this page.



Upload the program to your Arduino/Seeeduino.

Success

When the code finishes uploaded, you will see number flowing from 0 to 9.

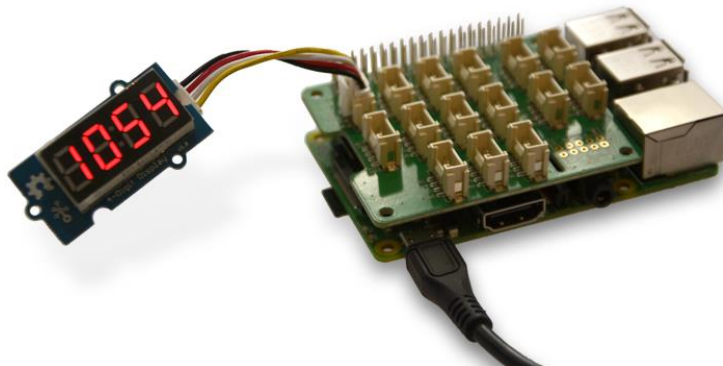
Play With Raspberry Pi (With Grove Base Hat for Raspberry Pi)

Hardware

- **Step 1.** Things used in this project:

Raspberry pi	Grove Base Hat for RasPi	Grove - 4 Digit Display
		

- **Step 2.** Plug the Grove Base Hat into Raspberry Pi.
- **Step 3.** Connect the 4-digit display to port 12 of the Base Hat.
- **Step 4.** Connect the Raspberry Pi to PC through USB cable.



Note

For step 3 you are able to connect the digit to **any GPIO Port** but make sure you change the command with the corresponding port number.

Software

- **Step 1.** Follow [Setting Software](#) to configure the development environment.
- **Step 2.** Download the source file by cloning the grove.py library.

```
1cd ~  
2git clone https://github.com/Seeed-Studio/grove.py
```


- **Step 3.** Excute below commands to run the code.

```
1cd grove.py/grove
2python grove_4_digit_display.py 12 13
```

Following is the grove_4_digit_display.py code.

```
1import sys
2import time
3from grove.gpio import GPIO
4
5
6charmap = {
7    '0': 0x3f,
8    '1': 0x06,
9    '2': 0x5b,
10   '3': 0x4f,
11   '4': 0x66,
12   '5': 0x6d,
13   '6': 0x7d,
14   '7': 0x07,
15   '8': 0x7f,
16   '9': 0x6f,
17   'A': 0x77,
18   'B': 0x7f,
19   'b': 0x7C,
20   'C': 0x39,
21   'c': 0x58,
22   'D': 0x3f,
23   'd': 0x5E,
24   'E': 0x79,
25   'F': 0x71,
26   'G': 0x7d,
27   'H': 0x76,
28   'h': 0x74,
29   'I': 0x06,
30   'J': 0x1f,
31   'K': 0x76,
32   'L': 0x38,
33   'l': 0x06,
34   'n': 0x54,
35   'O': 0x3f,
36   'o': 0x5c,
37   'P': 0x73,
38   'r': 0x50,
39   'S': 0x6d,
40   'U': 0x3e,
41   'V': 0x3e,
42   'Y': 0x66,
43   'Z': 0x5b,
44   '-': 0x40,
45   '_': 0x08,
46   ' ': 0x00
47}
48
```

```

49ADDR_AUTO = 0x40
50ADDR_FIXED = 0x44
51STARTADDR = 0xC0
52BRIGHT_DARKEST = 0
53BRIGHT_DEFAULT = 2
54BRIGHT_HIGHEST = 7
55
56
57class Grove4DigitDisplay(object):
58    colon_index = 1
59
60    def __init__(self, clk, dio, brightness=BRIGHT_DEFAULT):
61        self.brightness = brightness
62
63        self.clk = GPIO(clk, direction=GPIO.OUT)
64        self.dio = GPIO(dio, direction=GPIO.OUT)
65        self.data = [0] * 4
66        self.show_colon = False
67
68    def clear(self):
69        self.show_colon = False
70        self.data = [0] * 4
71        self._show()
72
73    def show(self, data):
74        if type(data) is str:
75            for i, c in enumerate(data):
76                if c in charmap:
77                    self.data[i] = charmap[c]
78                else:
79                    self.data[i] = 0
80            if i == self.colon_index and self.show_colon:
81                self.data[i] |= 0x80
82            if i == 3:
83                break
84        elif type(data) is int:
85            self.data = [0, 0, 0, charmap['0']]
86            if data < 0:
87                negative = True
88                data = -data
89            else:
90                negative = False
91            index = 3
92            while data != 0:
93                self.data[index] = charmap[str(data % 10)]
94                index -= 1
95                if index < 0:
96                    break
97                data = int(data / 10)
98
99            if negative:
100                if index >= 0:
101                    self.data[index] = charmap['-']
102                else:
103                    self.data = charmap['_'] + [charmap['9']] * 3
104        else:
105            raise ValueError('Not support {}'.format(type(data)))

```



```

106     self._show()
107
108     def _show(self):
109         with self:
110             self._transfer(ADDR_AUTO)
111
112         with self:
113             self._transfer(STARTADDR)
114             for i in range(4):
115                 self._transfer(self.data[i])
116
117         with self:
118             self._transfer(0x88 + self.brightness)
119
120     def update(self, index, value):
121         if index < 0 or index > 4:
122             return
123
124         if value in charmap:
125             self.data[index] = charmap[value]
126         else:
127             self.data[index] = 0
128
129         if index == self.colon_index and self.show_colon:
130             self.data[index] |= 0x80
131
132         with self:
133             self._transfer(ADDR_FIXED)
134
135         with self:
136             self._transfer(STARTADDR | index)
137             self._transfer(self.data[index])
138
139         with self:
140             self._transfer(0x88 + self.brightness)
141
142
143     def set_brightness(self, brightness):
144         if brightness > 7:
145             brightness = 7
146
147         self.brightness = brightness
148         self._show()
149
150     def set_colon(self, enable):
151         self.show_colon = enable
152         if self.show_colon:
153             self.data[self.colon_index] |= 0x80
154         else:
155             self.data[self.colon_index] &= 0x7F
156         self._show()
157
158     def _transfer(self, data):
159         for _ in range(8):
160             self.clk.write(0)
161             if data & 0x01:
162                 self.dio.write(1)

```

```

163         else:
164             self.dio.write(0)
165             data >>= 1
166             time.sleep(0.000001)
167             self.clk.write(1)
168             time.sleep(0.000001)
169
170     self.clk.write(0)
171     self.dio.write(1)
172     self.clk.write(1)
173     self.dio.dir(GPIO.IN)
174
175     while self.dio.read():
176         time.sleep(0.001)
177         if self.dio.read():
178             self.dio.dir(GPIO.OUT)
179             self.dio.write(0)
180             self.dio.dir(GPIO.IN)
181     self.dio.dir(GPIO.OUT)
182
183     def _start(self):
184         self.clk.write(1)
185         self.dio.write(1)
186         self.dio.write(0)
187         self.clk.write(0)
188
189     def _stop(self):
190         self.clk.write(0)
191         self.dio.write(0)
192         self.clk.write(1)
193         self.dio.write(1)
194
195     def __enter__(self):
196         self._start()
197
198     def __exit__(self, exc_type, exc_val, exc_tb):
199         self._stop()
200
201
202 Grove = Grove4DigitDisplay
203
204
205 def main():
206     if len(sys.argv) < 3:
207         print('Usage: {} clk dio'.format(sys.argv[0]))
208         sys.exit(1)
209
210     display = Grove4DigitDisplay(int(sys.argv[1]), int(sys.argv[2]))
211
212     count = 0
213     while True:
214         t = time.strftime("%H%M", time.localtime(time.time()))
215         display.show(t)
216         display.set_colon(count & 1)
217         count += 1
218         time.sleep(1)
219

```

```
220
221if __name__ == '__main__':
222    main()
```

Success

If everything goes well, the 4-digit display will show the current time.

You can quit this program by simply press `Ctrl + C`.

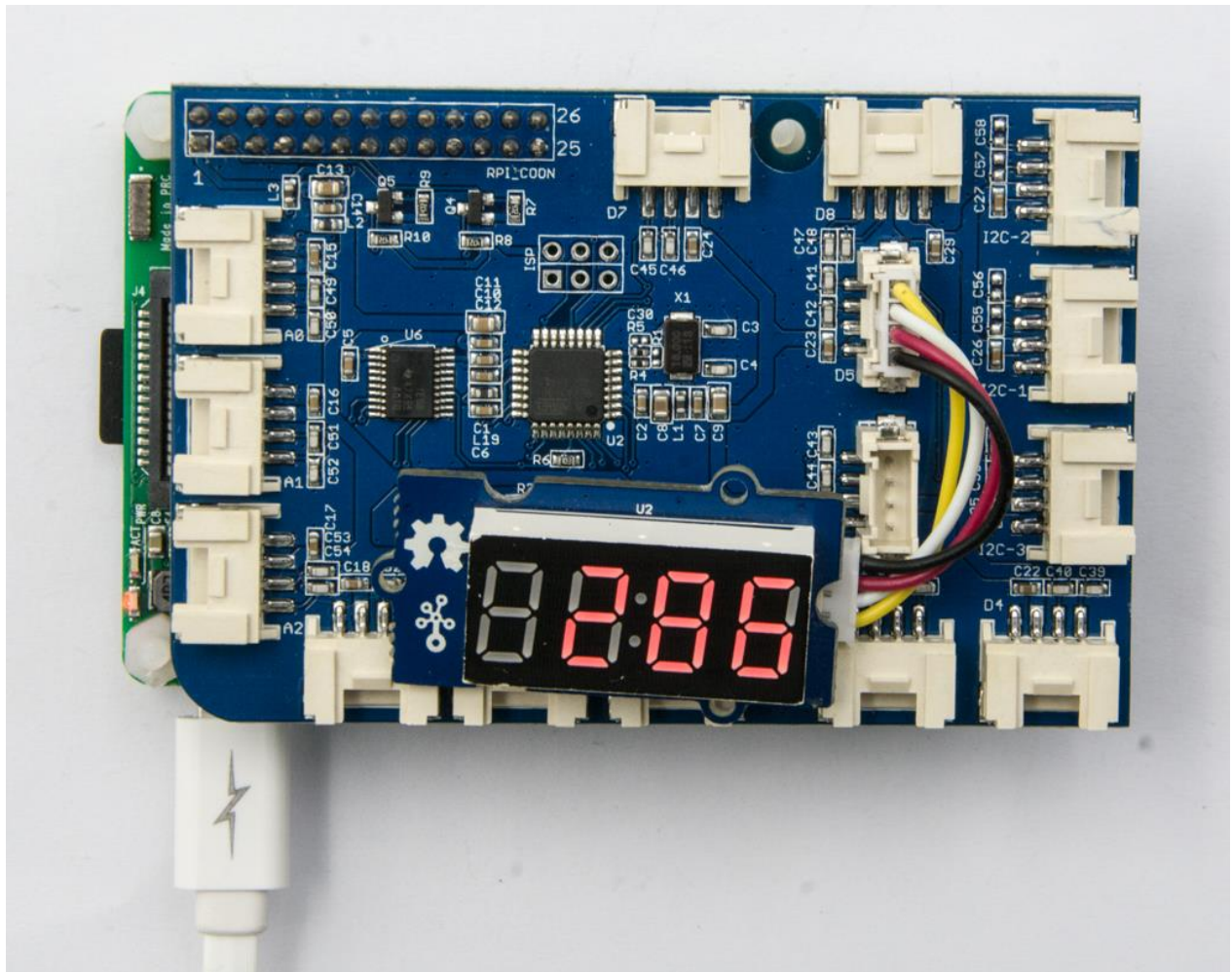
Play With Raspberry Pi (with GrovePi_Plus)

Hardware

- **Step 1.** Prepare the below stuffs:

Raspberry pi	GrovePi_Plus	Grove-4-Digit Display
		

- **Step 2.** Plug the GrovePi_Plus into Raspberry.
- **Step 3.** Connect Grove-4-Digit Display to **D5** port of GrovePi_Plus.
- **Step 4.** Connect the Raspberry to PC through USB cable.



Software

- **Step 1.** Follow [Setting Software](#) to configure the development environment.
- **Step 2.** Git clone the Github repository.

```
1cd ~  
2git clone https://github.com/DexterInd/GrovePi.git
```

- **Step 3.** Excute below commands to monitor the loudness.

```
1cd ~/GrovePi/Software/Python  
2python grove_4_digit_display.py
```

Here is the grove_4_digit_display.py code.

```
1# NOTE: 4x red 7 segment display with colon and 8 luminance levels, but no  
2decimal points  
3
```

```

4import time
5import grovepi
6
7# Connect the Grove 4 Digit Display to digital port D5
8# CLK,DIO,VCC,GND
9display = 5
10grovepi.pinMode(display,"OUTPUT")
11
12# If you have an analog sensor connect it to A0 so you can monitor it
13below
14sensor = 0
15grovepi.pinMode(sensor,"INPUT")
16
17time.sleep(.5)
18
19# 4 Digit Display methods
20# grovepi.fourDigit_init(pin)
21# grovepi.fourDigit_number(pin,value,leading_zero)
22# grovepi.fourDigit_brightness(pin,brightness)
23# grovepi.fourDigit_digit(pin,segment,value)
24# grovepi.fourDigit_segment(pin,segment,leds)
25# grovepi.fourDigit_score(pin,left,right)
26# grovepi.fourDigit_monitor(pin,analog,duration)
27# grovepi.fourDigit_on(pin)
28# grovepi.fourDigit_off(pin)
29
30while True:
31    try:
32        print ("Test 1) Initialise")
33        grovepi.fourDigit_init(display)
34        time.sleep(.5)
35
36        print ("Test 2) Set brightness")
37        for i in range(0,8):
38            grovepi.fourDigit_brightness(display,i)
39            time.sleep(.2)
40        time.sleep(.3)
41
42        # set to lowest brightness level
43        grovepi.fourDigit_brightness(display,0)
44        time.sleep(.5)
45
46        print ("Test 3) Set number without leading zeros")
47        leading_zero = 0
48        grovepi.fourDigit_number(display,1,leading_zero)
49        time.sleep(.5)
50        grovepi.fourDigit_number(display,12,leading_zero)
51        time.sleep(.5)
52        grovepi.fourDigit_number(display,123,leading_zero)
53        time.sleep(.5)
54        grovepi.fourDigit_number(display,1234,leading_zero)
55        time.sleep(.5)
56
57        print ("Test 4) Set number with leading zeros")
58        leading_zero = 1
59        grovepi.fourDigit_number(display,5,leading_zero)
60        time.sleep(.5)

```

```

61     grovepi.fourDigit_number(display,56,leading_zero)
62     time.sleep(.5)
63     grovepi.fourDigit_number(display,567,leading_zero)
64     time.sleep(.5)
65     grovepi.fourDigit_number(display,5678,leading_zero)
66     time.sleep(.5)
67
68     print ("Test 5) Set individual digit")
69     grovepi.fourDigit_digit(display,0,2)
70     grovepi.fourDigit_digit(display,1,6)
71     grovepi.fourDigit_digit(display,2,9)
72     grovepi.fourDigit_digit(display,3,15) # 15 = F
73     time.sleep(.5)
74
75     print ("Test 6) Set individual segment")
76     grovepi.fourDigit_segment(display,0,118) # 118 = H
77     grovepi.fourDigit_segment(display,1,121) # 121 = E
78     grovepi.fourDigit_segment(display,2,118) # 118 = H
79     grovepi.fourDigit_segment(display,3,121) # 121 = E
80     time.sleep(.5)
81
82     grovepi.fourDigit_segment(display,0,57) # 57 = C
83     grovepi.fourDigit_segment(display,1,63) # 63 = O
84     grovepi.fourDigit_segment(display,2,63) # 63 = O
85     grovepi.fourDigit_segment(display,3,56) # 56 = L
86     time.sleep(.5)
87
88     print ("Test 7) Set score")
89     grovepi.fourDigit_score(display,0,0)
90     time.sleep(.2)
91     grovepi.fourDigit_score(display,1,0)
92     time.sleep(.2)
93     grovepi.fourDigit_score(display,1,1)
94     time.sleep(.2)
95     grovepi.fourDigit_score(display,1,2)
96     time.sleep(.2)
97     grovepi.fourDigit_score(display,1,3)
98     time.sleep(.2)
99     grovepi.fourDigit_score(display,1,4)
100    time.sleep(.2)
101    grovepi.fourDigit_score(display,1,5)
102    time.sleep(.5)
103
104    print ("Test 8) Set time")
105    grovepi.fourDigit_score(display,12,59)
106    time.sleep(.5)
107
108    print ("Test 9) Monitor analog pin")
109    seconds = 10
110    grovepi.fourDigit_monitor(display,sensor,seconds)
111    time.sleep(.5)
112
113    print ("Test 10) Switch all on")
114    grovepi.fourDigit_on(display)
115    time.sleep(.5)
116
117    print ("Test 11) Switch all off")

```

```

118     grovepi.fourDigit_off(display)
119     time.sleep(.5)
120
121     except KeyboardInterrupt:
122         grovepi.fourDigit_off(display)
123         break
124     except IOError:
125         print ("Error")

```

- **Step 4.** We will see the Grove-4-Digit Display as below.

```

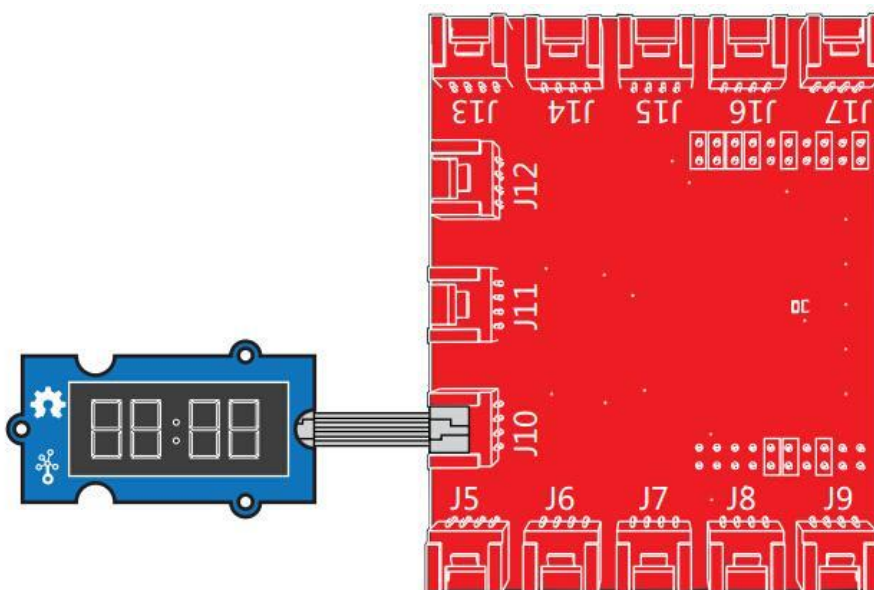
1pi@raspberrypi:~/GrovePi/Software/Python $ python grove_4_digit_display.py
2Test 1) Initialise
3Test 2) Set brightness
4Test 3) Set number without leading zeros
5Test 4) Set number with leading zeros
6Test 5) Set individual digit
7Test 6) Set individual segment
8Test 7) Set score
9Test 8) Set time
10Test 9) Monitor analog pin
11Test 10) Switch all on
12Test 11) Switch all off

```

Play with TI LaunchPad

Displaying the Numbers (4-Digital-Display)

This example demonstrates how to display some digital numbers using a Grove-4-Digital Display.




```

1/*
2 * TM1637.cpp
3 * A library for the 4 digit display
4 */
5#include "TM1637.h"
6#define CLK 39 //pins definitions for TM1637 and can be changed to other
7ports
8#define DIO 38
9TM1637 tm1637(CLK,DIO);
10void setup()
11{
12     tm1637.init();
13     tm1637.set(BRIGHT_TYPICAL); //BRIGHT_TYPICAL = 2,BRIGHT_DARKEST =
140,BRIGHTTEST = 7;
15}
16void loop()
17{
18     int8_t NumTab[] =
19{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}; //0~9,A,b,C,d,E,F
20     int8_t ListDisp[4];
21     unsigned char i = 0;
22     unsigned char count = 0;
23     delay(150);
24     while(1)
25     {
26         i = count;
27         count ++;
28         if(count == sizeof(NumTab)) count = 0;
29         for(unsigned char BitSelect = 0;BitSelect < 4;BitSelect ++)
30         {
31             ListDisp[BitSelect] = NumTab[i];
32             i ++;
33             if(i == sizeof(NumTab)) i = 0;
34         }
35         tm1637.display(0,ListDisp[0]);
36         tm1637.display(1,ListDisp[1]);
37         tm1637.display(2,ListDisp[2]);
38         tm1637.display(3,ListDisp[3]);
39         delay(300);
40     }
41 }

```

Resources

- [\[Eagle&PDF\] Grove-4-Digit Display V1.0 Schematic](#)
- [\[Library\] 4-Digit Display library](#)
- [\[Library\] TimerOne library](#)
- [\[Library\] Four-Digit Display Suli Library](#)
- [\[Library\] CodeCraft Code](#)
- [\[Datasheet\] TM1637 datasheet](#)
- [\[More Reading\] The Wooden Laser Gun](#)



Inspired by OVERWATCH, we have made a very cool Wooden Laser Gun toy for fun these day!

The Wooden Laser Gun and the Gun Target are all based on an Arduino board called Seeeduno Lotus. The laser emitter on the Laser Gun is controlled to fire laser pulse to "activate" the Gun Target. And there are 3 light sensors on the Gun Target to detect the laser pulse. It seems very simple right? If you are interested in our project, please make one for yourself or your child! It's worth to spend one day DIY it as a Xmas present.

Projects

MSP430 Alarm Clock with Grove Modules: Create your own alarm clock using the MSP430F5529 LaunchPad and the SeeedStudio Grove Modules.

Clock - Grove 4-digit Display Using Photon: Your first clock with 4 components, based on Grove and TM1637

Tech Support

Please submit any technical issue into our forum.

