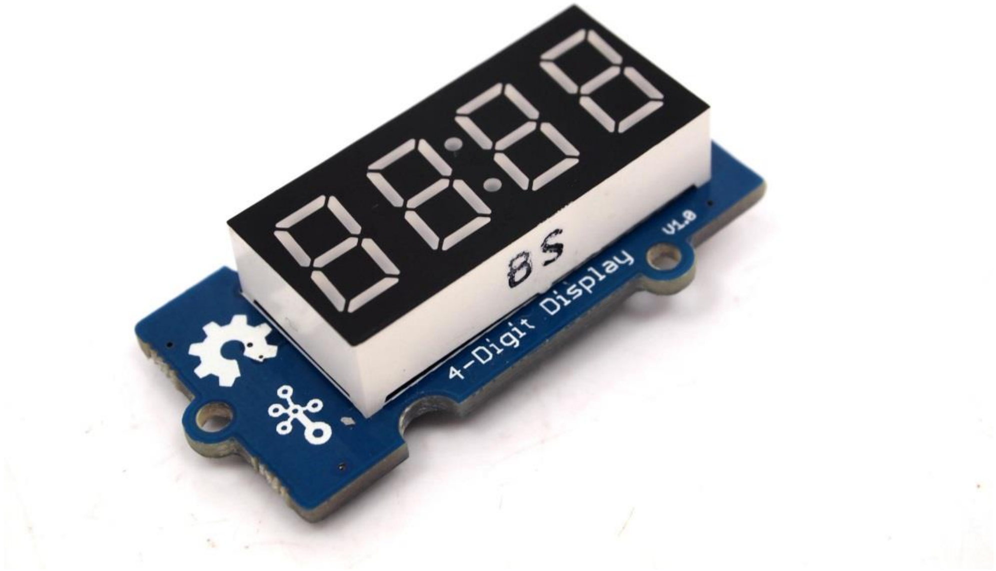
Grove - 4-Digit Display



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
		TO DO		TO DO

Caution
The platforms mentioned above as supported is/are an indication of the module's software or theoritical 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 Arduinobefore the start.

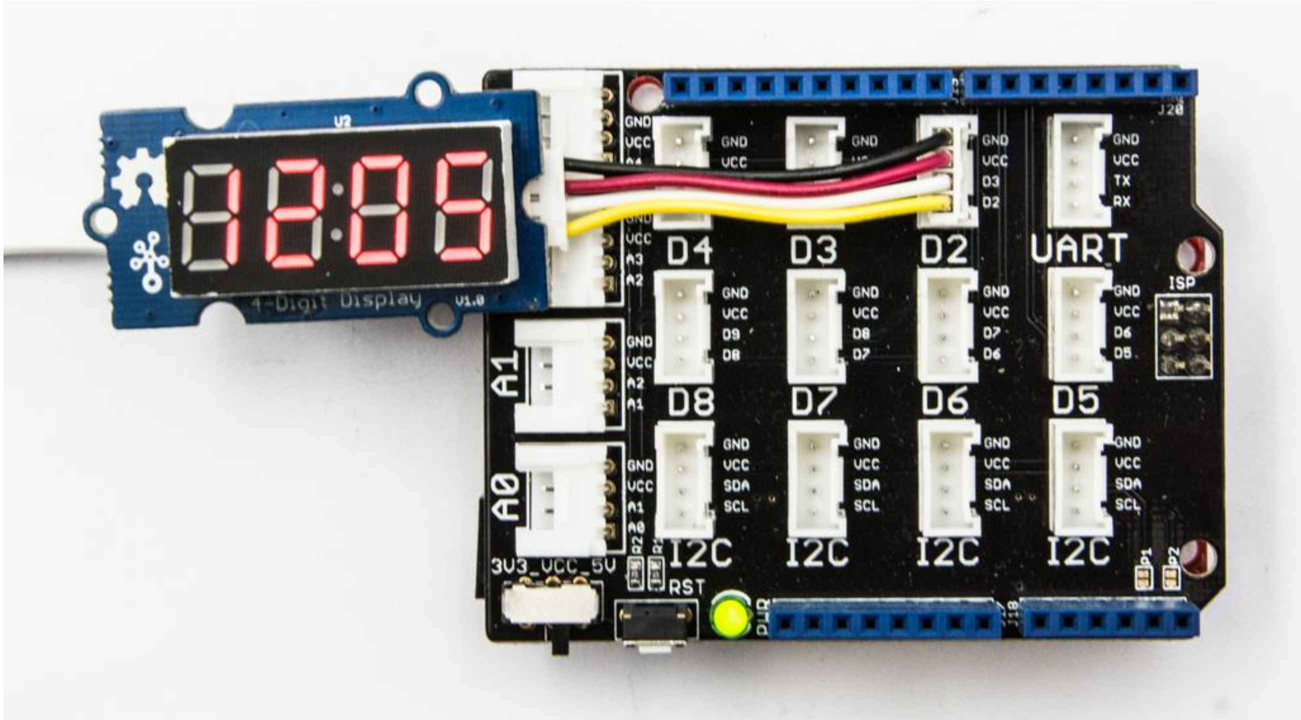
Play With Arduino

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.



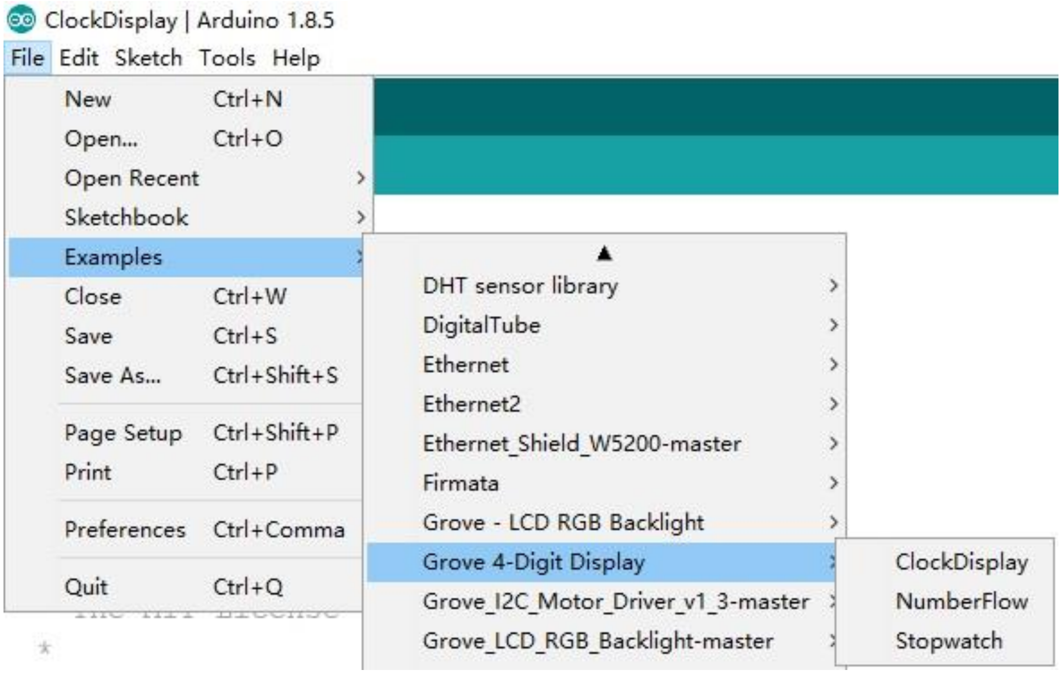
Note
If we don't have Grove Base Shield, We also can directly connect Grove-4-Digit Display to Seeeduino as below. We also can plug Grove-4-Digit Display to other Grove digital port.

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 Diaplsy to port D2 in a Base Shield

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

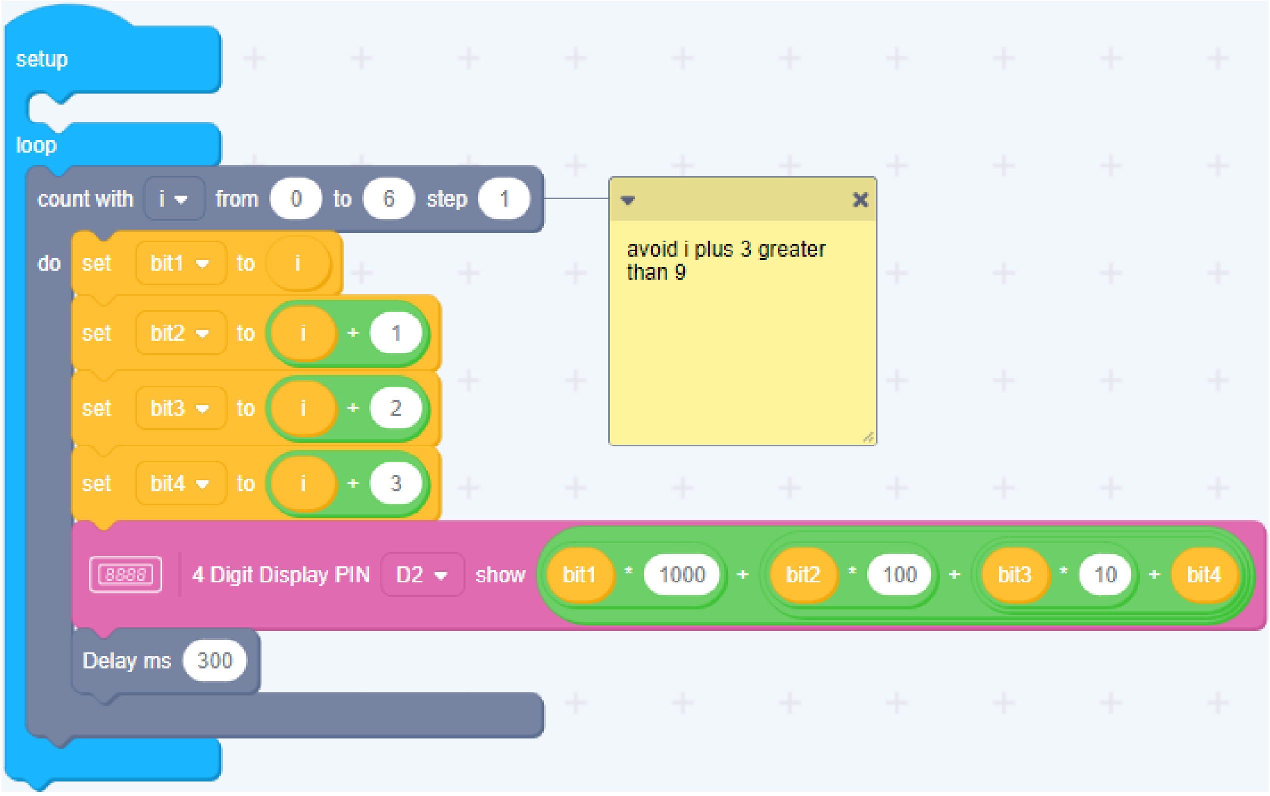
Step 3. Link Seeeduino/Arduino to your PC via an USB cable.

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.



```
setup
loop
  count with i from 0 to 6 step 1
  do  set bit1 to i
      set bit2 to i + 1
      set bit3 to i + 2
      set bit4 to i + 3
      4 Digit Display PIN D2 show bit1 * 1000 + bit2 * 100 + bit3 * 10 + bit4
      Delay ms 300
```

avoid i plus 3 greater than 9

Upload the program to your Arduino/Seeeduino.

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

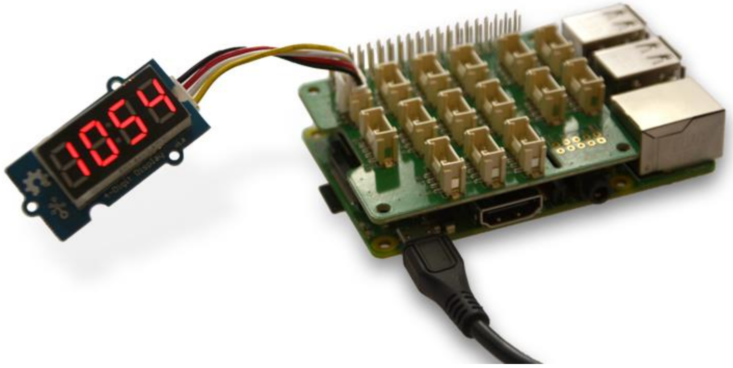
Play With Raspberry Pi (With Grove Base Hat for Raspberry Pi)

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.

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

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

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

Following is the grove_4_digit_display.py code.

```python
import sys
import time
from grove.gpio import GPIO


charmap = {
    '0': 0x3f,
    '1': 0x06,
    '2': 0x5b,
    '3': 0x4f,
    '4': 0x66,
    '5': 0x6d,
    '6': 0x7d,
    '7': 0x07,
    '8': 0x7f,
    '9': 0x6f,
    'A': 0x77,
    'B': 0x7f,
    'b': 0x7C,
    'C': 0x39,
    'c': 0x58,
    'D': 0x3f,
    'd': 0x5E,
    'E': 0x79,
    'F': 0x71,
    'G': 0x7d,
    'H': 0x76,
    'h': 0x74,
    'I': 0x06,
    'J': 0x1f,
    'K': 0x76,
    'L': 0x38,
    'l': 0x06,
    'n': 0x54,
    'O': 0x3f,
    'o': 0x5c,
    'P': 0x73,
    'r': 0x50,
    'S': 0x6d,
    'U': 0x3e,
    'V': 0x3e,
    'Y': 0x66,
    'Z': 0x5b,
    '-': 0x40,
    '_': 0x08,
    ' ': 0x00
}
```

```
ADDR_AUTO = 0x40
ADDR_FIXED = 0x44
STARTADDR = 0xC0
BRIGHT_DARKEST = 0
BRIGHT_DEFAULT = 2
BRIGHT_HIGHEST = 7


class Grove4DigitDisplay(object):
    colon_index = 1

    def __init__(self, clk, dio, brightness=BRIGHT_DEFAULT):
        self.brightness = brightness

        self.clk = GPIO(clk, direction=GPIO.OUT)
        self.dio = GPIO(dio, direction=GPIO.OUT)
        self.data = [0] * 4
        self.show_colon = False

    def clear(self):
        self.show_colon = False
        self.data = [0] * 4
        self._show()

    def show(self, data):
        if type(data) is str:
            for i, c in enumerate(data):
                if c in charmap:
                    self.data[i] = charmap[c]
                else:
                    self.data[i] = 0
                if i == self.colon_index and self.show_colon:
                    self.data[i] |= 0x80
                if i == 3:
                    break
        elif type(data) is int:
            self.data = [0, 0, 0, charmap['0']]
            if data < 0:
                negative = True
                data = -data
            else:
                negative = False
            index = 3
            while data != 0:
                self.data[index] = charmap[str(data % 10)]
                index -= 1
                if index < 0:
                    break
                data = int(data / 10)

            if negative:
                if index >= 0:
                    self.data[index] = charmap['-']
                else:
                    self.data = charmap['_'] + [charmap['9']] * 3
        else:
            raise ValueError('Not support {}'.format(type(data)))
```

```python
        self._show()

    def _show(self):
        with self:
            self._transfer(ADDR_AUTO)

        with self:
            self._transfer(STARTADDR)
            for i in range(4):
                self._transfer(self.data[i])

        with self:
            self._transfer(0x88 + self.brightness)

    def update(self, index, value):
        if index < 0 or index > 4:
            return

        if value in charmap:
            self.data[index] = charmap[value]
        else:
            self.data[index] = 0

        if index == self.colon_index and self.show_colon:
            self.data[index] |= 0x80

        with self:
            self._transfer(ADDR_FIXED)

        with self:
            self._transfer(STARTADDR | index)
            self._transfer(self.data[index])

        with self:
            self._transfer(0x88 + self.brightness)


    def set_brightness(self, brightness):
        if brightness > 7:
            brightness = 7

        self.brightness = brightness
        self._show()

    def set_colon(self, enable):
        self.show_colon = enable
        if self.show_colon:
            self.data[self.colon_index] |= 0x80
        else:
            self.data[self.colon_index] &= 0x7F
        self._show()

    def _transfer(self, data):
        for _ in range(8):
            self.clk.write(0)
            if data & 0x01:
                self.dio.write(1)
            else:
                self.dio.write(0)
            data >>= 1
            time.sleep(0.000001)
            self.clk.write(1)
            time.sleep(0.000001)

        self.clk.write(0)
        self.dio.write(1)
        self.clk.write(1)
        self.dio.dir(GPIO.IN)

        while self.dio.read():
            time.sleep(0.001)
            if self.dio.read():
                self.dio.dir(GPIO.OUT)
                self.dio.write(0)
                self.dio.dir(GPIO.IN)
        self.dio.dir(GPIO.OUT)

    def _start(self):
        self.clk.write(1)
        self.dio.write(1)
        self.dio.write(0)
        self.clk.write(0)

    def _stop(self):
        self.clk.write(0)
        self.dio.write(0)
        self.clk.write(1)
        self.dio.write(1)

    def __enter__(self):
        self._start()

    def __exit__(self, exc_type, exc_val, exc_tb):
        self._stop()


Grove = Grove4DigitDisplay


def main():
    if len(sys.argv) < 3:
        print('Usage: {} clk dio'.format(sys.argv[0]))
        sys.exit(1)

    display = Grove4DigitDisplay(int(sys.argv[1]), int(sys.argv[2]))

    count = 0
    while True:
        t = time.strftime("%H%M", time.localtime(time.time()))
        display.show(t)
        display.set_colon(count & 1)
        count += 1
        time.sleep(1)
```

```

if __name__ == '__main__':
    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`.

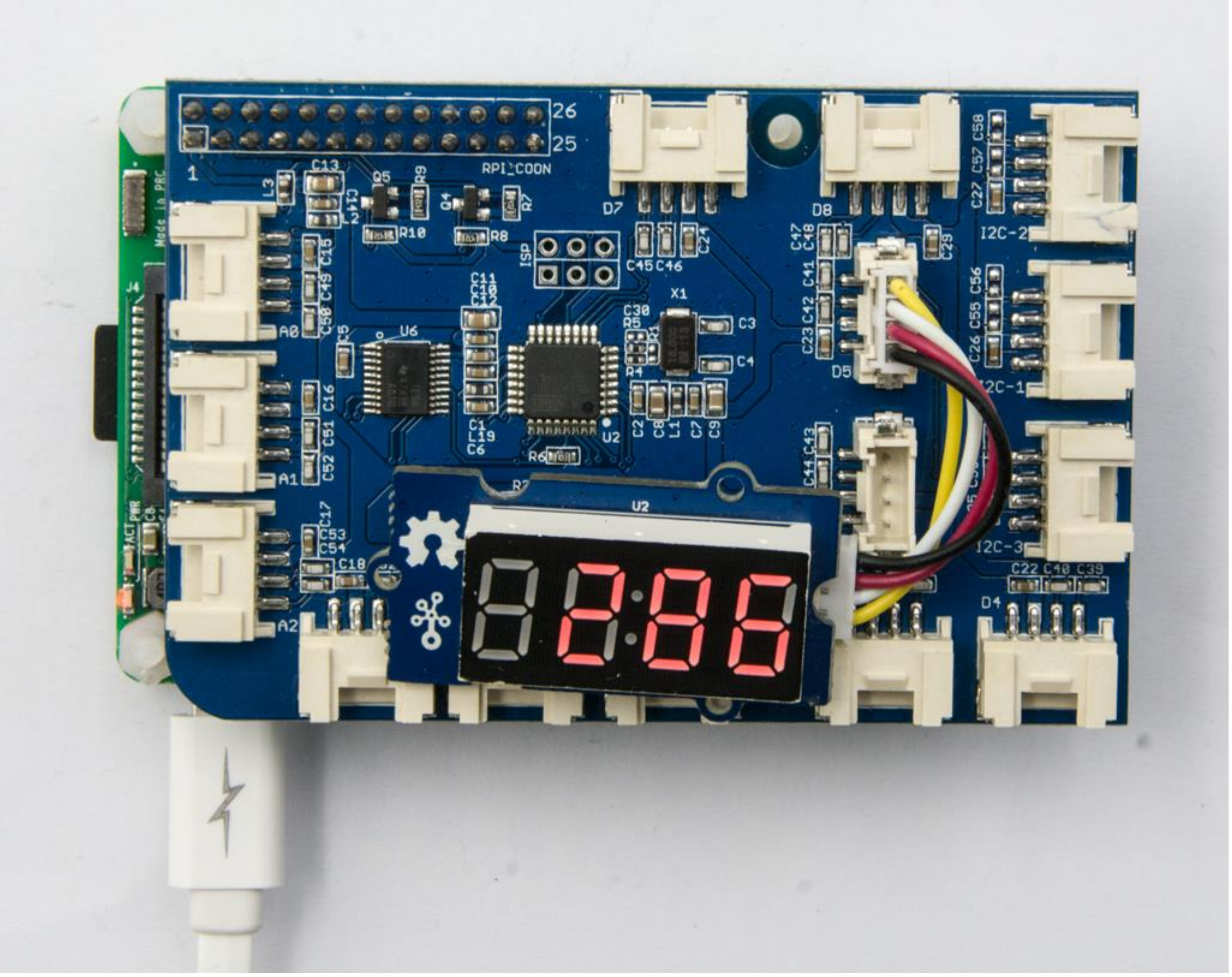
Play With Raspberry Pi (with GrovePi_Plus)

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.

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

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

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

Here is the grove_4_digit_display.py code.

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

```

```
import time
import grovepi

# Connect the Grove 4 Digit Display to digital port D5
# CLK,DIO,VCC,GND
display = 5
grovepi.pinMode(display,"OUTPUT")

# If you have an analog sensor connect it to A0 so you can monitor it
below
sensor = 0
grovepi.pinMode(sensor,"INPUT")

time.sleep(.5)

# 4 Digit Display methods
# grovepi.fourDigit_init(pin)
# grovepi.fourDigit_number(pin,value,leading_zero)
# grovepi.fourDigit_brightness(pin,brightness)
# grovepi.fourDigit_digit(pin,segment,value)
# grovepi.fourDigit_segment(pin,segment,leds)
# grovepi.fourDigit_score(pin,left,right)
# grovepi.fourDigit_monitor(pin,analog,duration)
# grovepi.fourDigit_on(pin)
# grovepi.fourDigit_off(pin)

while True:
    try:
        print ("Test 1) Initialise")
        grovepi.fourDigit_init(display)
        time.sleep(.5)

        print ("Test 2) Set brightness")
        for i in range(0,8):
            grovepi.fourDigit_brightness(display,i)
            time.sleep(.2)
        time.sleep(.3)

        # set to lowest brightness level
        grovepi.fourDigit_brightness(display,0)
        time.sleep(.5)

        print ("Test 3) Set number without leading zeros")
        leading_zero = 0
        grovepi.fourDigit_number(display,1,leading_zero)
        time.sleep(.5)
        grovepi.fourDigit_number(display,12,leading_zero)
        time.sleep(.5)
        grovepi.fourDigit_number(display,123,leading_zero)
        time.sleep(.5)
        grovepi.fourDigit_number(display,1234,leading_zero)
        time.sleep(.5)

        print ("Test 4) Set number with leading zeros")
        leading_zero = 1
        grovepi.fourDigit_number(display,5,leading_zero)
        time.sleep(.5)
```

```python
        grovepi.fourDigit_number(display,56,leading_zero)
        time.sleep(.5)
        grovepi.fourDigit_number(display,567,leading_zero)
        time.sleep(.5)
        grovepi.fourDigit_number(display,5678,leading_zero)
        time.sleep(.5)

        print ("Test 5) Set individual digit")
        grovepi.fourDigit_digit(display,0,2)
        grovepi.fourDigit_digit(display,1,6)
        grovepi.fourDigit_digit(display,2,9)
        grovepi.fourDigit_digit(display,3,15) # 15 = F
        time.sleep(.5)

        print ("Test 6) Set individual segment")
        grovepi.fourDigit_segment(display,0,118) # 118 = H
        grovepi.fourDigit_segment(display,1,121) # 121 = E
        grovepi.fourDigit_segment(display,2,118) # 118 = H
        grovepi.fourDigit_segment(display,3,121) # 121 = E
        time.sleep(.5)

        grovepi.fourDigit_segment(display,0,57) # 57 = C
        grovepi.fourDigit_segment(display,1,63) # 63 = O
        grovepi.fourDigit_segment(display,2,63) # 63 = O
        grovepi.fourDigit_segment(display,3,56) # 56 = L
        time.sleep(.5)

        print ("Test 7) Set score")
        grovepi.fourDigit_score(display,0,0)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,0)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,1)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,2)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,3)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,4)
        time.sleep(.2)
        grovepi.fourDigit_score(display,1,5)
        time.sleep(.5)

        print ("Test 8) Set time")
        grovepi.fourDigit_score(display,12,59)
        time.sleep(.5)

        print ("Test 9) Monitor analog pin")
        seconds = 10
        grovepi.fourDigit_monitor(display,sensor,seconds)
        time.sleep(.5)

        print ("Test 10) Switch all on")
        grovepi.fourDigit_on(display)
        time.sleep(.5)

        print ("Test 11) Switch all off")
        grovepi.fourDigit_off(display)
        time.sleep(.5)

    except KeyboardInterrupt:
        grovepi.fourDigit_off(display)
        break
    except IOError:
        print ("Error")
```

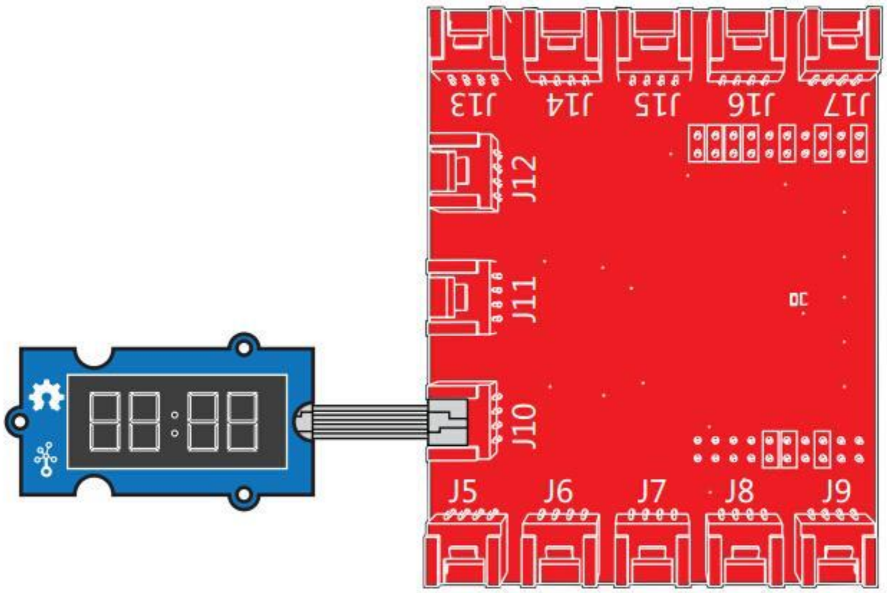
- **Step 4.** We will see the Grove-4-Digit Display as below.

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



```
/*
 * TM1637.cpp
 * A library for the 4 digit display
 */
#include "TM1637.h"
#define CLK 39 //pins definitions for TM1637 and can be changed to other
ports
#define DIO 38
TM1637 tm1637(CLK,DIO);
void setup()
{
    tm1637.init();
    tm1637.set(BRIGHT_TYPICAL);//BRIGHT_TYPICAL = 2,BRIGHT_DARKEST =
0,BRIGHTEST = 7;
}
void loop()
{
    int8_t NumTab[] =
{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};//0~9,A,b,C,d,E,F
    int8_t ListDisp[4];
    unsigned char i = 0;
    unsigned char count = 0;
    delay(150);
    while(1)
    {
        i = count;
        count ++;
        if(count == sizeof(NumTab)) count = 0;
        for(unsigned char BitSelect = 0;BitSelect < 4;BitSelect ++)
        {
            ListDisp[BitSelect] = NumTab[i];
            i ++;
            if(i == sizeof(NumTab)) i = 0;
        }
        tm1637.display(0,ListDisp[0]);
        tm1637.display(1,ListDisp[1]);
        tm1637.display(2,ListDisp[2]);
        tm1637.display(3,ListDisp[3]);
        delay(300);
    }
}
```

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 Seeeduino 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.

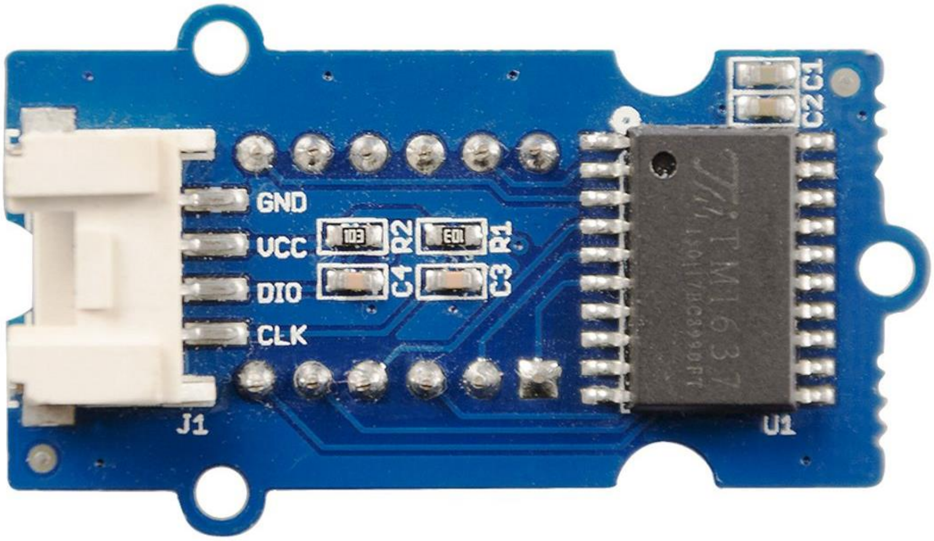
Projects

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

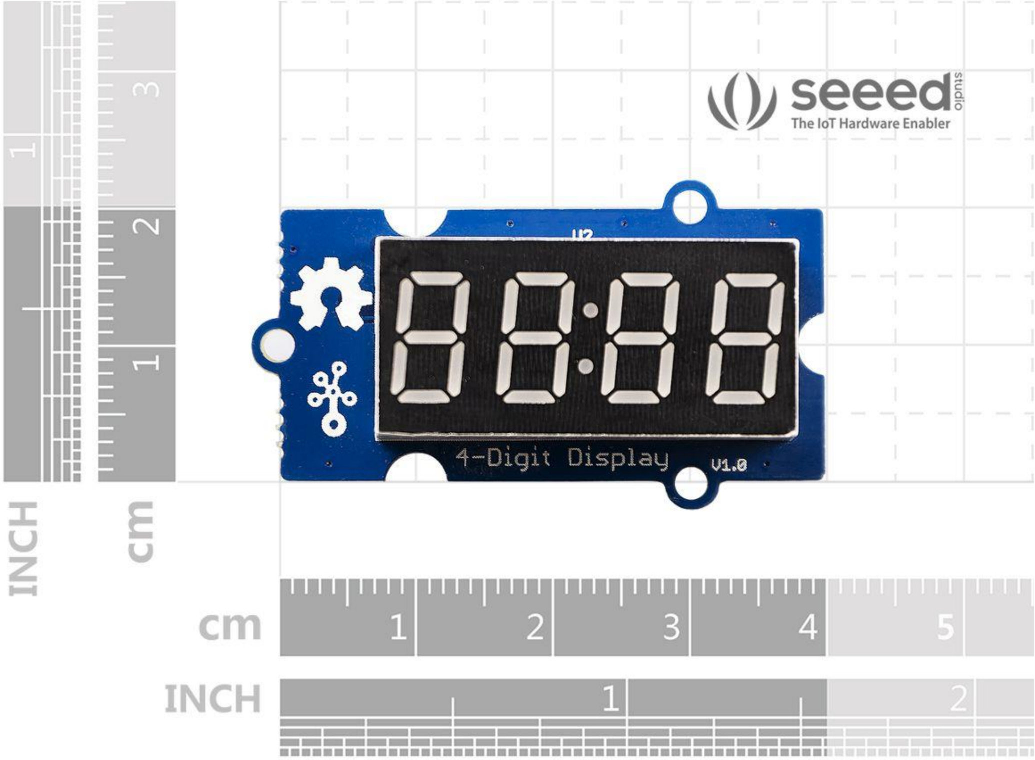
Clock - Grove 4-digit Display Using Photon: Your first clock with 4 components, based on Grove and TM1637

Tech Support

Please submit any technical issue into our forum.


GND
VCC
DIO
CLK
103
R2
R1
C4
C3
C2
C1
TM1637
J1
U1

U2
4-Digit Display
V1.0



https://www.seeedstudio.com/Grove-4-Digit-Display-p-1198.html//3-29-19