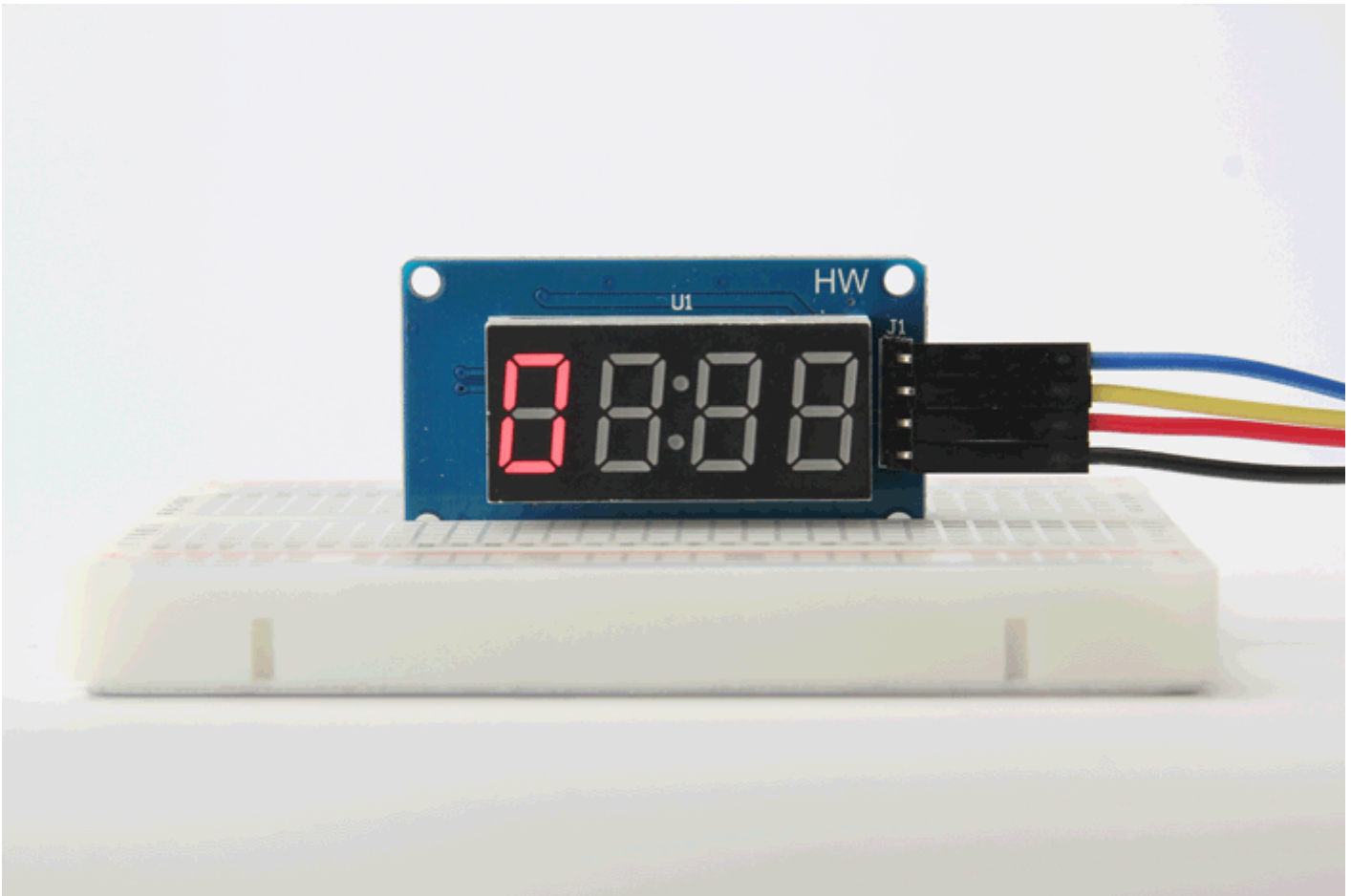


Makerguides.com

TM1637 4-digit 7-segment LED display Arduino tutorial

Written by Benne de Bakker (<https://www.makerguides.com/author/benne-de-bakker/>)



In this tutorial, you will learn how you can control TM1637 4-digit 7-segment displays with Arduino. These displays are fantastic for displaying sensor data, temperature, time, etc.

I have included 3 examples in this tutorial. In the first example, we will look at the basic functions of the TM1637Display library. In the second example, I will show you how you can display the time on a 4-digit display. The last example can be used to create a simple temperature display in combination with the DHT11.

Wiring diagrams are also included. After the example, I break down and explain how the code works, so you should have no problems modifying it to suit your needs.

If you have any questions, please leave a comment below.

For more display tutorials, check out the articles below:

- How to control a character I2C LCD with Arduino
(<https://www.makerguides.com/character-i2c-lcd-arduino-tutorial/>)
- How to use a 16x2 character LCD with Arduino
(<https://www.makerguides.com/character-lcd-arduino-tutorial/>)

Supplies

Hardware components

(https://amzn.to/2PTReoz)	TM1637 4-digit 7-segment display (https://amzn.to/2PTReoz)	× 1	Amazon (http://s.click) AliExpress (http://s.click)
(https://amzn.to/374ajjX)	Arduino Uno Rev3 (https://amzn.to/374ajjX)	× 1	Amazon (http://s.click)
	Breadboard (https://amzn.to/2sZTxNA)	× 1	Amazon (http://s.click)
	Jumper wires	~	

Jumper wires (https://amzn.to/2EG9wDc)	10	Amazon (htt)
DS3231 RTC (https://amzn.to/35Ufn3U)	× 1	Amazon (htt)
Adafruit DS3231 Precision RTC Breakout (https://amzn.to/2ZsmX3m) (alternative)	× 1	Amazon (htt)
DHT11 temperature and humidity sensor (https://amzn.to/2PWoRGl) (3- pin)	× 1	Amazon (htt) AliExpress (http://s.click)
USB cable type A/B (https://amzn.to/34SBuXf)	× 1	Amazon (htt)

Software

Arduino IDE (<https://www.arduino.cc/en/Main/Software>)

Makerguides.com is a participant in the Amazon Services LLC Associates Program, an affiliate advertising program designed to provide a means for sites to earn advertising fees by advertising and linking to products on Amazon.com.

Information about the display

Bare 4-digit 7-segment displays (<https://amzn.to/34TUAyv>) usually require 12 connection pins. That's quite a lot and doesn't leave much room for other sensors or modules.

Thanks to the TM1637 IC mounted on the back of the display module, this number can be reduced to just four. Two pins are required for the power connections and the other two pins are used to control the segments.

7-segment displays contain 7 (or 8) individually addressable LEDs. The segments are labeled A to G and some displays also have a dot (the 8th LED). Use this image as a reference when setting the individual segments in the code later.

You can buy many different display modules that use a TM1637 IC. The color, size, dots, and connection points can all be different. I don't have experience with all the different versions of this display but as long as they use the TM1637, the code examples provided below should work.

Here you can find the basic specifications of the display module that I used in this tutorial.

TM1637 4-Digit 7-Segment Display Specifications

Operating voltage	3.3 – 5 V
Current draw	80 mA
Luminance levels	8
Display dimensions	30 x 14 mm (0.36" digits)
Overall dimensions	42 x 24 x 12 mm
Hole dimensions	38 x 20, \varnothing 2.2 mm
Operating temperature	-10 – 80 °C
Cost	Check price (https://amzn.to/2zt0EOJ)

The TM1637 IC is made by Titan Micro Electronics (<http://www.titanmec.com/index.php/en/>). For more information, you can check out the datasheet below:

TM1637 Datasheet 

Wiring – Connecting TM1637 4-digit 7-segment display to Arduino UNO

Connecting the display to an Arduino or other microcontroller is super easy. You only need to connect 4 wires: 2 for power and 2 to transfer the data.

The wiring diagram below shows you how you can connect the display to the Arduino.

TM1637 4 digit 7 segment display with Arduino UNO wiring diagram.

The connections are also given in the table below:

TM1637 Display Connections

TM1637 4-Digit Display	Arduino
VCC	5 V
GND	GND
CLK	Digital pin 2
DIO	Digital pin 3

Note that the order and location of the pins can be different depending on the manufacturer!

For this tutorial, I connected CLK and DIO to pin 2 and 3 respectively, but you can change this to any of the digital pins you want. You just have to change the pin configuration in the code accordingly.

TM1637 4-digit 7-segment display Arduino example code

Avishay Orpaz has written an excellent library for TM1637 displays, the TM1637Display library

(<https://github.com/avishorp/TM1637/blob/master/examples/TM1637Test/TM1637Test.ino>). This library has several built-in functions that make controlling the display fairly easy.

The main functions include:

- **setSegments()** – Set the raw value of the segments of each digit
- **showNumberDec()** – Display a decimal number
- **showNumberDecEx()** – Display a decimal number with decimal points or colon

- **setBrightness()** – Set the brightness of the display
- **clear()** – Clear the display

The code example below features all of these functions. I will explain how each function can be used in more detail below.

You can upload the example code to your Arduino using the Arduino IDE (<https://www.arduino.cc/en/main/software>).

To install the library, you can download it as a .zip from GitHub here (<https://github.com/avishorp/TM1637>). Next, go to **Sketch > Include Library > Add .ZIP Library...** in the Arduino IDE.

TM1637-master.zip 

Another option is to navigate to **Tools > Manage Libraries...** or type Ctrl + Shift + I on Windows. The Library Manager will open and update the list of installed libraries.

You can search for 'tm1637' and look for the library by Avishay Orpaz. Select the latest version and then click Install.

Example code

You can copy the code by clicking the button in the top right corner of the code field.

```
1.  /* Example code for TM1637 4 digit 7 segment display with Arduino. More info:
    www.makerguides.com */
2.
3.  // Include the library:
4.  #include <TM1637Display.h>
5.
6.  // Define the connections pins:
7.  #define CLK 2
8.  #define DIO 3
9.
10. // Create display object of type TM1637Display:
11. TM1637Display display = TM1637Display(CLK, DIO);
12.
13. // Create array that turns all segments on:
14. const uint8_t data[] = {0xff, 0xff, 0xff, 0xff};
15.
16. // Create array that turns all segments off:
17. const uint8_t blank[] = {0x00, 0x00, 0x00, 0x00};
18.
19. // You can set the individual segments per digit to spell words or create other symbols:
20. const uint8_t done[] = {
21.     SEG_B | SEG_C | SEG_D | SEG_E | SEG_G,           // d
22.     SEG_A | SEG_B | SEG_C | SEG_D | SEG_E | SEG_F,   // O
23.     SEG_C | SEG_E | SEG_G,                           // n
24.     SEG_A | SEG_D | SEG_E | SEG_F | SEG_G            // E
25. };
26.
27. // Create degree Celsius symbol:
28. const uint8_t celsius[] = {
29.     SEG_A | SEG_B | SEG_F | SEG_G, // Circle
30.     SEG_A | SEG_D | SEG_E | SEG_F // C
31. };
32.
33. void setup() {
34.     // Clear the display:
35.     display.clear();
36.     delay(1000);
37. }
38.
39. void loop() {
40.     // Set the brightness:
41.     display.setBrightness(7);
42.     // All segments on:
43.     display.setSegments(data);
44.
45.     delay(1000);
46.     display.clear();
47.     delay(1000);
48.
49.     // Show counter:
50.     int i;
51.     for (i = 0; i < 101; i++) {
52.         display.showNumberDec(i);
```

```
53.     delay(50);
54.   }
55.
56.   delay(1000);
57.   display.clear();
58.   delay(1000);
59.
60.   // Print number in different locations, loops 2 times:
61.   int j;
62.   for (j = 0; j < 2; j++) {
63.     for (i = 0; i < 4; i++) {
64.       display.showNumberDec(i, false, 1, i);
65.       delay(500);
66.       display.clear();
67.     }
68.   }
69.
70.   delay(1000);
71.   display.clear();
72.   delay(1000);
73.
74.   // Set brightness (0-7):
75.   int k;
76.   for (k = 0; k < 8; k++) {
77.     display.setBrightness(k);
78.     display.setSegments(data);
79.     delay(500);
80.   }
81.
82.   delay(1000);
83.   display.clear();
84.   delay(1000);
85.
86.   // Print 1234 with the center colon:
87.   display.showNumberDecEx(1234, 0b11100000, false, 4, 0);
88.
89.   delay(1000);
90.   display.clear();
91.   delay(1000);
92.
93.   int temperature = 24;
94.   display.showNumberDec(temperature, false, 2, 0);
95.   display.setSegments(celsius, 2, 2);
96.
97.   delay(1000);
98.   display.clear();
99.   delay(1000);
100.
101.   display.setSegments(done);
102.   while(1);
103. }
```

How the code works:

The code starts with including the library. Make sure that you have the correct library installed, otherwise you will get an error message while compiling the code.

```
3. // Include the library:
4. #include <TM1637Display.h>
```

The next step is to specify the connection pins. The statement `#define` is used to give a name to a constant value. The compiler will replace any references to this constant with the defined value when the program is compiled. So everywhere you mention `CLK`, the compiler will replace it with the value `2` when the program is compiled.

```
6. // Define the connections pins:
7. #define CLK 2
8. #define DIO 3
```

Next, we create a display object of the type `TM1637Display` with the defined `CLK` and `DIO` pins. Note that I called the display 'display', but you can use other names as well like 'temperature_display'.

The name that you give to the display will be used later to write data to that particular display. You can create and control multiple display objects with different names and connection pins. There currently is no limit in the library.

```
10. // Create display object of type TM1637Display:
11. TM1637Display display = TM1637Display(CLK, DIO);
12.
13. // You can create more than one display object. Give them different names and connection
    pins:
14. TM1637Display display_1 = TM1637Display(2, 3);
15. TM1637Display display_2 = TM1637Display(4, 5);
16. TM1637Display display_3 = TM1637Display(6, 7);
```

There are several ways to control the individual segments of the display. Before the setup section of the code, I specified several arrays to set the individual display segments. We will use the function `setSegments()` later to write them to the display.

The first option is to write hexadecimal numbers to the display for each digit. The hexadecimal 0xff translates to 11111111 in binary, this sets all the segments on (including the dot if your display has one). 0xef for example, translates to 11101111. This would set all the segments on, except for segment E. Note that counting goes from right to left, so 11111111 corresponds to segments (dot)GFEDCBA. You can find a conversion chart for hexadecimal to binary here (<http://cactus.io/resources/toolbox/decimal-binary-octal-hexadecimal-conversion>).

```

13. // Create array that turns all segments on:
14. const uint8_t data[] = {0xff, 0xff, 0xff, 0xff};
15.
16. // Create array that turns all segments off:
17. const uint8_t blank[] = {0x00, 0x00, 0x00, 0x00};

```

The library has a function built-in that makes setting individual segments a bit easier. See the code snippet below. You can create arrays to spell words. Each segment is separated by a | and digits of the display are separated by a comma.

```

19. // You can set the individual segments per digit to spell words or create other symbols:
20. const uint8_t done[] = {
21.     SEG_B | SEG_C | SEG_D | SEG_E | SEG_G, // d
22.     SEG_A | SEG_B | SEG_C | SEG_D | SEG_E | SEG_F, // O
23.     SEG_C | SEG_E | SEG_G, // n
24.     SEG_A | SEG_D | SEG_E | SEG_F | SEG_G // E
25. };

```

You can leave the setup section of the code empty if you want. I just used the function `clear()` to ensure that the display was cleared.

```

33. void setup() {
34.     // Clear the display:
35.     display.clear();
36.     delay(1000);

```

```
37. | }
```

In the loop section of the code, I show several examples of the different library functions:

setSegments(segments[], length, position)

This function can be used to set the individual segments of the display. The first argument is the array that includes the segment information. The second argument specifies the number of digits to be modified (0-4). If you want to spell dOnE, this would be 4, for a °C symbol, this would be 2. The third argument sets the position from which to print (0 – leftmost, 3 – rightmost). So if you want to print a °C symbol on the third and fourth digit, you would use:

```
1. // Create degree Celsius symbol:
2. const uint8_t celsius[] = {
3.     SEG_A | SEG_B | SEG_F | SEG_G, // Circle
4.     SEG_A | SEG_D | SEG_E | SEG_F // C
5. };
6.
7. display.setSegments(celsius, 2, 2);
```

The second and third argument of the function can also be omitted.

showNumberDec(number, leading_zeros, length, position)

This is probably the function that you will use the most. The first argument is a number that you want to display on the screen. The rest of the arguments are optional.

The second argument can be used to turn on or off leading zeros. 10 without leading zeros would print as __10 and with leading zeros as 0010. You can turn them on by setting this argument as true or turn them off by setting it as false. NOTE: leading zero is not supported with negative numbers.

The third and fourth argument are the same as in the previous function.

```
64. // Print the number 12 without leading zeros on the second and third digit:
65. display.showNumberDec(12, false, 2, 1);
```

showNumberDecEx(number, dots, leading_zeros, length, position)

This function allows you to control the dots of the display. Only the second argument is different from the showNumberDec function. It allows you to set the dots between the individual digits.

You can use the following values.

For displays with dots between each digit:

- 0b10000000 – 0.000
- 0b01000000 – 00.00
- 0b00100000 – 000.0
- 0b11100000 – 0.0.0.0

For displays with just a colon:

- 0b01000000 – 00:00

For displays with dots and colons colon:

- 0b11100000 – 0.0:0.0

So if you want to display a clock with center colon on (see clock example below), you would use something like:

```
86. // Print 1234 with the center colon:  
87. display.showNumberDecEx(1234, 0b11100000, false, 4, 0);
```

setBrightness(brightness, true/false)

This function sets the brightness of the display (as the name suggests). You can specify a brightness level from 0 (lowest brightness) to 7 (highest brightness). The second parameter can be used to turn the display on or off, false means off.

```
1. // Set the display brightness (0-7):  
2. display.setBrightness(7);
```

Clock example: TM1637 4-digit 7-segment display with DS3231 RTC

One of the typical uses for a 4-digit 7-segment display is to show the time. By combining the TM1637 with a real time clock module (RTC), you can easily create a 24-hour clock.

In this example I used this commonly used DS3231 RTC module (<https://amzn.to/2L2FIER>).

This module communicates with the Arduino via I2C, so you only need two connections to read the time.

The wiring diagram below shows you how you can connect the DS3231 RTC to the Arduino. Note that the TM1637 display is connected in the same way as before.

TM1637 with DS3231 RTC and Arduino UNO wiring diagram.

The connections are also given in the table below:

DS3231 RTC Connections

DS3231	Arduino
VCC	5 V
GND	GND
SDA	A4
SCL	A5

The following code example can be used to display the time in a 24-hour time format. If your display has a center colon, then this code will make it blink. You can also disable this by removing the last few lines of code.

The first time you upload the code, the RTC will be set to the time that the sketch was compiled.

You can install a coin cell battery on the back of the module, so the time is stored in case it loses power. Apparently the charging circuit of most Chinese modules can possibly overcharge the coin cell battery (<https://forum.arduino.cc/index.php?topic=278270.15>), so you might want to buy a DS3231 module from Adafruit (<https://amzn.to/2MPsYnl>) instead.

The code uses the Adafruit RTC library, which you can download here on GitHub (<https://github.com/adafruit/RTCLib>). You can also install it via the Library Manager in the Arduino IDE by searching for 'RTCLib', or click the download button below:

RTCLib-master.zip 

Example code

```
1.  /* Arduino example code to display a 24 hour time format clock on a TM1637 4 digit 7
    2.  segment display with a DS32321 RTC. More info: www.makerguides.com */
    3.
    4.  // Include the libraries:
    5.  #include "RTCLib.h"
    6.  #include <TM1637Display.h>
    7.
    8.  // Define the connections pins:
    9.  #define CLK 2
   10.  #define DIO 3
   11.
   12.  // Create rtc and display object:
   13.  RTC_DS3231 rtc;
   14.  TM1637Display display = TM1637Display(CLK, DIO);
   15.
   16.  void setup() {
   17.      // Begin serial communication at a baud rate of 9600:
   18.      Serial.begin(9600);
   19.      // Wait for console opening:
```

```
19.     delay(3000);
20.
21.     // Check if RTC is connected correctly:
22.     if (! rtc.begin()) {
23.         Serial.println("Couldn't find RTC");
24.         while (1);
25.     }
26.     // Check if the RTC lost power and if so, set the time:
27.     if (rtc.lostPower()) {
28.         Serial.println("RTC lost power, lets set the time!");
29.         // The following line sets the RTC to the date & time this sketch was compiled:
30.         rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
31.         // This line sets the RTC with an explicit date & time, for example to set
32.         // January 21, 2014 at 3am you would call:
33.         //rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
34.     }
35.
36.     // Set the display brightness (0-7):
37.     display.setBrightness(5);
38.     // Clear the display:
39.     display.clear();
40. }
41.
42. void loop() {
43.     // Get current date and time:
44.     DateTime now = rtc.now();
45.
46.     // Create time format to display:
47.     int displaytime = (now.hour() * 100) + now.minute();
48.
49.     // Print displaytime to the Serial Monitor:
50.     Serial.println(displaytime);
51.
52.     // Display the current time in 24 hour format with leading zeros enabled and a center
53.     colon:
54.     display.showNumberDecEx(displaytime, 0b11100000, true);
55.     // Remove the following lines of code if you want a static instead of a blinking center
56.     colon:
57.     delay(1000);
58.     display.showNumberDec(displaytime, true); // Prints displaytime without center colon.
59.
60.     delay(1000);
61. }
```

Thermometer example: TM1637 4-digit 7-segment display with DHT11 temperature and humidity sensor

4-Digit 7-segment displays are great for displaying sensor readings like temperature, humidity, voltage or speed. In the following example, I will show you how you can display temperature readings on the TM1637 display.

We will be using the popular DHT11 temperature and humidity sensor (<https://amzn.to/2zqPb2b>).

The wiring diagram below shows you how you can connect the DHT11 sensor in combination with the TM1637 display to the Arduino.

Note that the TM1637 display is connected in the same way as before.

TM1637 4 digit 7 segment display with DHT11 temperature and humidity sensor and Arduino UNO wiring diagram.

The connections are also given in the table below:

DHT11 Connections

DHT11	Arduino
+	5 V
-	GND
s	Digital pin 4

Note that the order of the pins can be different, depending on the manufacturer.

If you would like to use a 4 pin sensor, check out my tutorial for the DHT11 and DHT22 temperature and humidity sensors.

- How to use DHT11 and DHT22 Sensors with Arduino (<https://www.makerguides.com/dht11-dht22-arduino-tutorial/>)

The example code below can be used to display the temperature readings on the display. It alternates between the temperature in Celsius and Fahrenheit, both are shown for 2 seconds.

The function `setSegments()` is used to display the Celsius and Fahrenheit symbols.

The code uses the **Adafruit DHT sensor library** which you can download here on GitHub (<https://github.com/adafruit/DHT-sensor-library>). This library only works if you also have the **Adafruit Unified Sensor** library installed, which is also available on GitHub (https://github.com/adafruit/Adafruit_Sensor).

You can also download the two libraries by clicking on the buttons below:

[DHT-sensor-library-master.zip](#) 

Adafruit_Sensor-master.zip 

For more information see my DHT11 with Arduino tutorial (<https://www.makerguides.com/dht11-dht22-arduino-tutorial/>).

Example code

```
1.  /* Arduino example sketch to display DHT11 temperature readings on a TM1637 4-digit 7-
    2.  segment display. More info: www.makerguides.com */
    3.
    4.  // Include the libraries:
    5.  #include <TM1637Display.h>
    6.  #include <Adafruit_Sensor.h>
    7.  #include <DHT.h>
    8.
    9.  // Define the connections pins:
   10.  #define CLK 2
   11.  #define DIO 3
   12.  #define DHTPIN 4
   13.
   14.  // Create variable:
   15.  int temperature_celsius;
   16.  int temperature_fahrenheit;
   17.
   18.  // Create degree Celsius symbol:
   19.  const uint8_t celsius[] = {
   20.      SEG_A | SEG_B | SEG_F | SEG_G, // Circle
   21.      SEG_A | SEG_D | SEG_E | SEG_F // C
   22.  };
   23.
   24.  // Create degree Fahrenheit symbol:
   25.  const uint8_t fahrenheit[] = {
   26.      SEG_A | SEG_B | SEG_F | SEG_G, // Circle
   27.      SEG_A | SEG_E | SEG_F | SEG_G // F
   28.  };
   29.
   30.  // Set DHT type, uncomment whatever type you're using!
   31.  #define DHTTYPE DHT11 // DHT 11
   32.  //#define DHTTYPE DHT22 // DHT 22 (AM2302)
   33.  //#define DHTTYPE DHT21 // DHT 21 (AM2301)
   34.
   35.  // Create display object of type TM1637Display:
   36.  TM1637Display display = TM1637Display(CLK, DIO);
   37.  // Create dht object of type DHT:
   38.  DHT dht = DHT(DHTPIN, DHTTYPE);
   39.
   40.  void setup() {
   41.      // Set the display brightness (0-7):
   42.      display.setBrightness(0);
```

```
42. // Clear the display:
43. display.clear();
44. // Setup sensor:
45. dht.begin();
46. // Begin serial communication at a baud rate of 9600:
47. Serial.begin(9600);
48. // Wait for console opening:
49. delay(2000);
50. }
51.
52. void loop() {
53. // Read the temperature as Celsius and Fahrenheit:
54. temperature_celsius = dht.readTemperature();
55. temperature_fahrenheit = dht.readTemperature(true);
56. // Print the temperature to the Serial Monitor:
57. Serial.println(temperature_celsius);
58. Serial.println(temperature_fahrenheit);
59.
60. // Show the temperature on the TM1637 display:
61. display.showNumberDec(temperature_celsius, false, 2, 0);
62. display.setSegments(celsius, 2, 2);
63.
64. delay(2000);
65.
66. display.showNumberDec(temperature_fahrenheit, false, 2, 0);
67. display.setSegments(fahrenheit, 2, 2);
68.
69. delay(2000);
70. }
```

Conclusion

In this article I have shown you how you can use a TM1637 4-digit 7-segment display with Arduino. We also looked at a clock and thermometer example. I hope you found it useful and informative. If you did, please **share it with a friend** who also likes electronics and making things!

I really like to use these displays to show sensor readings or things like the speed of a motor. With the TM1637Display library, programming the displays becomes very easy, so there is no reason you shouldn't incorporate one in your next project.

I would love to know what projects you plan on building (or have already built) with this display. If you have any questions, suggestions, or if you think that things are missing in this tutorial, **please leave a comment down below.**

Note that comments are held for moderation to prevent spam.

Beginner



What to read next?

LM35 analog temperature sensor with Arduino tutorial
(<https://www.makerguides.com/lm35-arduino-tutorial/>)

TMP36 analog temperature sensor with Arduino tutorial
(<https://www.makerguides.com/tmp36-arduino-tutorial/>)

Arduino Nano Board Guide (Pinout, Specifications, Comparison)
(<https://www.makerguides.com/arduino-nano/>)

The complete guide for DS18B20 digital temperature sensors with Arduino
(<https://www.makerguides.com/ds18b20-arduino-tutorial/>)

How to use an IR receiver and remote with Arduino
(<https://www.makerguides.com/ir-receiver-remote-arduino-tutorial/>)

Comments

Ricky says

February 16, 2021 at 9:37 am (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-6921>)

Hello,

The project is very nice.

I used it with MLX90614 but could not display the third digit with a decimal. Have you tried doing this with three digit thermometer decimal point?

Reply

Forrest says

February 7, 2021 at 11:23 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-6583>)

I have been working on a project for months, having one major setback. This display would not ever light up. I couldn't figure out what I was doing wrong until I found this web page and followed the instructions. Thank you very much. I can now proceed with my machine. Have a blessed day.

Reply

Torkel says

December 1, 2020 at 11:53 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-5090>)

Hey, great tutorial. These TM1637 displays are nice to use with NANO.
Using your tutorial for making a manometer with various functions.
Struggling with getting the digits and dp s in the right places, but I'll get it! Hope you are doing ok in these troublesome times 😊

Kind Regards

Torkel

Reply

Chris says

July 20, 2020 at 10:40 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-3274>)

Hello

Thank you for the clear tutorial, it's one of the better ones which I've come across and explains very well.

I'm now trying to integrate buttons, in order to adjust the time, in the event of drift or the clocks changing. I just seem to be getting a blank screen or flashing.

Don't suppose you could offer any direction?

Kind regards

Chris

Reply

Stellator says

June 3, 2020 at 9:44 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-2853>)

Hi, thanks for your very nice tutorial!

Can you give me a hint how to actually get also the first digit of a temperature sensor displayed? Thanks to your help a get the dot positioned correctly but i dont get the first or second digit of the temperature reading displayed (ADT7410 150°C 16bit sensor 😊)

Reply

Neil Barnett says

April 18, 2020 at 2:17 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-2478>)

It also works fine with the ESP32.

Probably also with the ESP8266, but I used mine...

Reply

Willem says

March 1, 2020 at 1:02 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1957>)

Very clear and good working example!

Thanks a lot.

Reply

Ron Walker says

February 13, 2020 at 3:44 am (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1834>)

Hello Benne,

I would like to run two TM1637 LED displays, one for set (target temperature) and the other for actual temp.

Are the “CLK” and “DIO” hard coded into the library so I can only drive one display, or is there another way round this?

Reply

Benne de Bakker says

May 26, 2020 at 11:12 am (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-2759>)

Hi Ron,

No, the CLK and DIO pins can be set to any of the digital outputs of the Arduino. So for your second display, just create an extra tm1637 object with different CLK and DIO pins:

```
// Define the connections pins:
```

```
#define CLK_1 2
```

```
#define DIO_1 3
```

```
#define CLK_2 4
```

```
#define DIO_2 5
```

```
// Create display object of type TM1637Display:
```

```
TM1637Display display_1 = TM1637Display(CLK_1, DIO_2);
```

```
TM1637Display display_2 = TM1637Display(CLK_2, DIO_2);
```

I hope this helps.

Reply

Nori T. says

January 6, 2020 at 12:33 am (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1504>)

Dear Makerguides.com,

I am interested in your instructions. I want to know how you integrate in the way you are displaying in the beginning, like first 0123, then time and temp plus humidity.

I read carefully if you mention somewhere in your document, but I couldn't find at all.

I want to start first by showing date in 4 digits like JA 06 (meaning Jan. 06), then time and temp plus humidity.

Would please elaborate your tutorial to this extent?

It will be very nice for me to try this way of display to get what I want.

Thank you so much in advance.

Nori T. from Japan.

Reply

brucemangy says

December 5, 2019 at 8:41 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1261>)

```
for (j = 0; j < 2; j++) {
```

```
-> 'lt' was not declared in this scope
```

```
less than ?
```

```
can you explain ?
```

Reply

Benne de Bakker says

December 6, 2019 at 10:47 am (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1264>)

Hi Bruce,

Thanks for pointing that out, it was a typo in my HTML file. The `<` should just be the less-than sign '`<`' in the code.

Benne

Reply

dainius says

December 3, 2019 at 1:43 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1238>)

Thanks for you examples, TM1637 works very well:)

Reply

DS says

November 9, 2019 at 6:37 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1035>)

Hi,

I have bought a cheap version of the display from aliexpress(https://www.banggood.com/nl/4-Digit-7-Segment-0_56-LED-Display-Tube-Decimal-7-Segments-HT16K33-I2C-Clock-Double-Dots-Module-For-Arduino-p-1558434.html?rmmds=myorder&cur_warehouse=CN) (https://www.banggood.com/nl/4-Digit-7-Segment-0_56-LED-Display-Tube-Decimal-7-Segments-HT16K33-I2C-Clock-Double-Dots-Module-For-Arduino-p-1558434.html?rmmds=myorder&cur_warehouse=CN) but it doesn't light up when I connect it the way you suggest. The RTC does work and I get the time stamps on the serial monitor.

Do you have any suggestions on why the display doesn't work?

Thanks

Reply

Benne de Bakker says

November 9, 2019 at 7:18 pm (<https://www.makerguides.com/tm1637-arduino-tutorial/#comment-1037>)

Hi,

It looks like the display that you are using doesn't actually use a TM1637 LED drive control chip but an HT16K33 instead. This means that the library and the rest of this tutorial probably won't work for your display. Adafruit sells several LED displays using the HT16K33 and I think you should be able to use their library to drive your display as well. Here you can find the example code of this library for 7 segment displays:

https://github.com/adafruit/Adafruit_LED_Backpack/blob/master/examples/sevensseg/sevensseg.ino

(https://github.com/adafruit/Adafruit_LED_Backpack/blob/master/examples/sevensseg/sevensseg.ino)

Benne

Reply

(<https://www.makerguides.com/tmp36-arduino-tutorial/>)


TMP36 analog temperature sensor with Arduino tutorial
(<https://www.makerguides.com/tmp36-arduino-tutorial/>)

(<https://www.makerguides.com/hc-sr501-arduino-tutorial/>)

How to use HC-SR501 PIR Motion Sensor with Arduino
(<https://www.makerguides.com/hc-sr501-arduino-tutorial/>)

(<https://www.makerguides.com/arduino-motor-shield-stepper-motor-tutorial/>)

How to control a Stepper Motor with Arduino Motor Shield Rev3
(<https://www.makerguides.com/arduino-motor-shield-stepper-motor-tutorial/>)

 Ezoic (<https://www.ezoic.com/what-is-ezoic/>)

[report this ad](#)

© 2021 Makerguides.com - All Rights Reserved