16x24 LED Matrix
Created by lady ada

Last updated on 2018-08-22 03:36:44 PM UTC
Guide Contents

Guide Contents .................................................. 2
Introduction ...................................................... 3
Wiring .............................................................. 5
Wiring for one panel .............................................. 5
Testing .............................................................. 8
Download Adafruit_HT1632 library ....................... 8
Installing Adafruit GFX ......................................... 8
Libraries ............................................................ 10
  Low and High Level Library ................................ 10
Multiple Displays .................................................. 11
  Two displays! .................................................... 11
  3-10 displays ................................................... 12
  How many cables do I need? ............................... 15
Drawing ............................................................ 16
  Drawing Text and Shapes .................................... 16
Downloads .......................................................... 17
These LED panels take care of all the work of making a big matrix display. Each panel has six 8x8 red matrix modules, for a 16x24 matrix. The panel has a HT1632C chip on the back with does all the multiplexing work for you and has a 3-pin SPI-like serial interface to talk to it and set LEDs on or off. There's a few extras as well, such as being able to change the brightness of the entire display, or blink the entire display at 1 Hz.

One really nice thing about this particular LED matrix module is that it is designed to be 'chainable' - you can connect to 8 panels together to make an extra long display.

© Adafruit Industries
https://learn.adafruit.com/16x24-led-matrix
And of course, we have written a full Arduino library that not only takes care of controlling the display, it also intelligently handles chained displays, so that they appear to be one long matrix. The library has functions for drawing pixels, lines, rectangles, circles and text. You'll be making it display stuff in 15 minutes!
Wiring

Wiring for one panel

Wiring is thankfully fairly simple, much easier than trying to actually wire up 6 x 8x8 matrices. The HT1632C driver chip requires only 3 data pins - **data**, **write**, and **chip select (cs)**. You can't skip using **cs** like many 'true' SPI devices. One of the nice things about these chips is that they are designed to be used in multiples, so you can share the **data** and **write** pins. However, each HT1632C must have its own **cs** pin! So if you have one display, you need 3 pins, two displays need 4 pins, three displays need 5, etc.

Let's assume you'll be using a single panel to start (and then show how to wire up multiples).

Begin by plugging in a 2x5 pin IDC cable into the top left socket:

![Image of a 2x5 pin IDC cable plugged into a panel](image)

You should also check that the **J5** solder jumper is 'filled' with solder - you need to have **J5** shorted for using one or two panels in a cascade. Use a soldering iron to heat up some solder and melt it on to **J5** if it isn't filled yet.

Next, you will need a few wires, we will use the following convention: a Red wire for **VCC (+5V)**, a Black wire for **Ground**, a White wire for **CS0** (chip select #0), an orange wire for **Data** and a yellow wire for **Write**. Make sure that when you look at the other side of the IDC cable to plug in the wires, you have the connector arranged right, the red stripe on the IDC is on the same side as the red wire for **VCC**.

© Adafruit Industries

[https://learn.adafruit.com/16x24-led-matrix](https://learn.adafruit.com/16x24-led-matrix)
Finally, you can wire the panel up to your Arduino. You can use another microcontroller, if you adapt the code but we will use an Arduino.

Connect **VCC** (red) to +5V, **GROUND** (black) to Ground, **DATA** (orange) to digital 2, **WRITE** (yellow) to digital 3, and **CS0** (white) to digital 4.
That's it! Now you are ready to perform the panel test.
Testing

Download Adafruit_HT1632 library

To begin reading sensor data, you will need to download Adafruit_HT1632 from our github repository (https://adafruit.it/ckL). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip:

https://adafruit.it/l4D

Rename the uncompressed folder Adafruit_HT1632 and check that the Adafruit_HT1632 folder contains Adafruit_HT1632.cpp and Adafruit_HT1632.h.

Place the Adafruit_HT1632 library folder in your arduinosketchfolder/libraries/ folder. You may need to create the libraries subfolder if it's your first library. Restart the IDE.

We also have a great tutorial on Arduino library installation at: http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (https://adafruit.it/aYM)

Installing Adafruit GFX

You'll also have to install the Adafruit GFX graphics core library at this github repo (https://adafruit.it/aJa). Rename it to Adafruit_GFX and install it the same way as the Adafruit_HT1632 library.

Now you are ready to test! Open up the IDE and load File→Examples→Adafruit_HT1632→basicdemo and upload it to your Arduino.

This code will do a basic test of the underlying chip, and light up the LEDs on the panel. The LEDs will not light up in order because the memory of the driver doesn't match the layout of the LEDs (this is normal, we fix the problem later!)

Once you have done the low level test, open up File→Examples→Adafruit_HT1632→matrixdemo - this is a more useful demo, it will light up all the LEDs in order on the panel.
Libraries

Low and High Level Library

The library contains two types of objects, one called Adafruit_HT1632 and one called Adafruit_HT1632LEDMatrix. The former is a low level library, that is meant to talk to one controller chip at a time and directly to the locations in memory, and the latter is a full featured library object, that does proper LED location translation, and handles multiple panels.

The last point is an important one. You can "chain" HT1632's together so that they share data/write lines but you still have to figure out how to get text to split nicely across the separate panels. We've taken care of that hard part for you with the library.

For example, if you want two panels in a row, simply add a second CS line at the end of the object creation.

```c
// use this line for single matrix
Adafruit_HT1632LEDMatrix matrix = Adafruit_HT1632LEDMatrix(DATA, WR, CS);
// use this line for two matrices!
Adafruit_HT1632LEDMatrix matrix = Adafruit_HT1632LEDMatrix(DATA, WR, CS, CS2);
```

The Adafruit_HT1632LEDMatrix object will automatically think of itself as a 16x48 LED matrix instead of a 16x24 and when you draw text or shapes, they will be split properly.

Right now the library supports up to 4 panels in a row

Chances are you will never want to use Adafruit_HT1632 objects directly, instead using Adafruit_HT1632LEDMatrix but we wanted to discuss why there are two.
Multiple Displays

Two displays!

Adding a second display is very easy. First, we do suggest testing both separately as above. Once you know they work, connect the upper right header of the first panel to the upper left of the second. You can use a long or shorty IDC cable, we happen to like the short ones but the long ones will work fine (they’re a little more bulky of course).

Then connect another wire (brown this time) to CS1 this is the CS line for the second panel.

Connect CS1 to digital 5 on the Arduino.
Open up the File→Examples→Adafruit_HT1632→matrixdemo example sketch and change the top part so that you have two panels active, like this:

```cpp
// use this line for single matrix
//Adafruit_HT1632LEDMatrix matrix = Adafruit_HT1632LEDMatrix(DATA, WR, CS);
// use this line for two matrices!
Adafruit_HT1632LEDMatrix matrix = Adafruit_HT1632LEDMatrix(DATA, WR, CS, CS2);
```

Now upload the matrixdemo test to see the panels light up in order.

3-10 displays
You can chain up to 8 displays, but the system for wiring is a little bit different. Instead of using one cable, you'll need
two between each panel. That is so each of the CSn pins can be addressed - the second 10-pin cable has 8 CS pins.

To perform the chain, first you will need to remove the J5 jumper using wick and a soldering iron. This will let you use the second IDC CS pins.

Then chain panels so that two cables go between each set, we suggest shorty cables.

Finally, you will need to use both IDC cables. The first (top left) cable has VCC and GND, as well as DATA and WRITE as in the previous tests.
Then on the second IDC (bottom left) you can connect CS0 (white), CS1 (brown), and CS2 (green). We connected CS2 to digital 6.

Then of course change the object creation to match that you have three CS pins.
You can do the same for 4 panels by adding another CS pin.

How many cables do I need?
Depending on how many you want to chain, the number of cables will vary. For 3 or more panels, you will need extras. For more than 2 panels, we suggest using the short IDC cables to keep the wiring neat and avoid having too much power lost to the cable length.

1. One 10-pin IDC cable (included)
2. Two 10-pin IDC cables (included)
3. Six IDC cables - two long ones are included so you need four more short ones
4. Eight IDC cables - two long ones are included so you need six more short ones
5. Ten IDC cables - two long ones are included so you need eight more short ones
6. Twelve IDC cables - two long ones are included so you need ten more short ones
7. Fourteen IDC cables - two long ones are included so you need twelve more short ones
8. Sixteen IDC cables - two long ones are included so you need fourteen more short ones
Drawing Text and Shapes

Now that we have the panels wired up as we want, we can use the more advanced parts of the library for drawing shapes and text. Depending on how many panels you have hooked up, the total 'width' will be different (24 pixels per panel) so call `width()` to get the total pixel width from the matrix object. You can also call `height()` - which will always be 16 in this case.

The Adafruit_GFX library for Arduino provides a common syntax and set of graphics functions for all of our TFT, LCD and OLED displays. This allows Arduino sketches to easily be adapted between display types with minimal fuss...and any new features, performance improvements and bug fixes will immediately apply across our complete offering of color displays.

The GFX library is what lets you draw points, lines, rectangles, round-rects, triangles, text, etc.

Check out our detailed tutorial here [http://learn.adafruit.com/adafruit-gfx-graphics-library](https://adafruit.it/aPx)
Downloads

- HT1632C datasheet (https://adafruit.it/ckM)