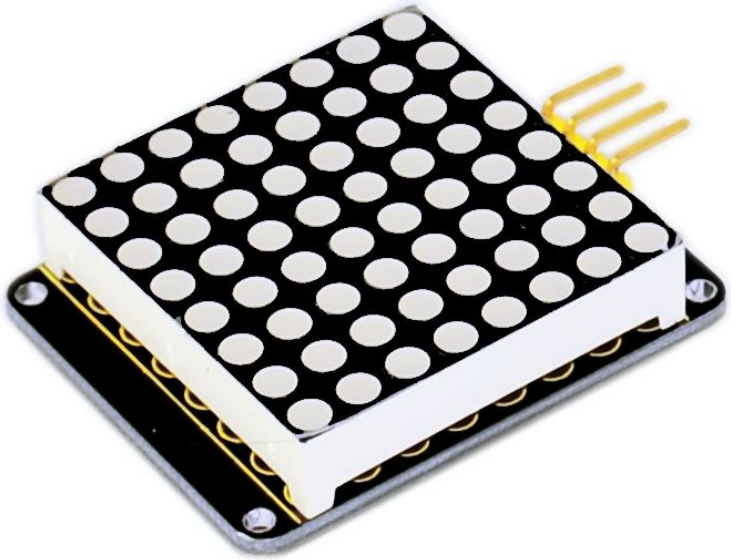
Keyestudio I2C 8x8 LED Matrix HT16K33



Introduction

What's better than a single LED? Lots of LEDs! A fun way to make a small display is to use an 8x8 matrix or a 4-digit 7-segment display. Matrices like these are 'multiplexed' - so to control 64 LEDs you need 16 pins. That's a lot of pins, and there are driver chips like the MAX7219 that can control a matrix for you but there's a lot of wiring to set up and they take up a ton of space. Here we feel your pain! After all, wouldn't it be awesome if you could control a matrix without tons of wiring? That's where these lovely LED matrix backpacks come in.

The matrices use a driver chip that does all the heavy lifting for you: They have a built in clock so they multiplex the display. They use constant-current drivers for ultra-bright, consistent color, 1/16 step display dimming, all via a simple I2C interface. These 1.2" matrix backpacks come with three address-selection jumpers so you can connect up to eight 1.2" 8x8's together (or a combination, such as four 1.2" 8x8's and four 7-segments, etc) on a single I2C bus.

https://github.com/adafruit/Adafruit-LED-Backpack-Library

Arduino code:

```
#include <Wire.h>
#include "Adafruit_LEDBackpack.h"
#include "Adafruit_GFX.h"
#ifndef _BV
#define _BV(bit) (1<<(bit))
#endif
Adafruit_LEDBackpack matrix = Adafruit_LEDBackpack();
uint8_t counter = 0;
void setup() {
  Serial.begin(9600);
  Serial.println("HT16K33 test");
  matrix.begin(0x70);  // pass in the address
}
void loop() {
  // paint one LED per row. The HT16K33 internal memory looks like
  // a 8x16 bit matrix (8 rows, 16 columns)
  for (uint8_t i=0; i<8; i++) {
    // draw a diagonal row of pixels
    matrix.displaybuffer[i] = _BV((counter+i) % 16) | _BV((counter+i+8) % 16)  ;
  }
  // write the changes we just made to the display
  matrix.writeDisplay();
  delay(100);
 counter++;
  if (counter >= 16) counter = 0;
}
```