

LEDmatrix7219

Arduino and chipKit library for LED matrices driven by MAX7219

Manual

The logo for Rinky-Dink Electronics features the company name in a stylized, glowing cyan font with a 3D effect. The text is set against a dark background that includes a close-up image of a green printed circuit board (PCB) with various electronic components and traces visible.

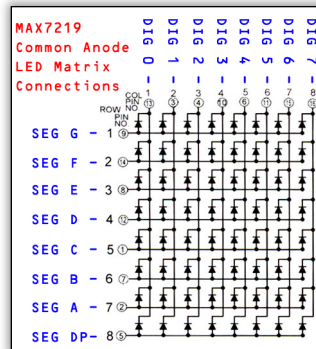
Rinky-Dink Electronics

Introduction:

I needed a library to drive some 8x8 LED matrices using MAX7219 drivers. This library is the result.

It should be noted that the MAX7219 is designed to drive Common Cathode LED displays but my 8x8 matrices were all Common Anode. The library has not been tested with Common Cathode matrices.

For Common Anode matrices the columns should be connected to the DIGx outputs from the MAX7219, and the matrix rows should be connected to the SEGx outputs. SEG DP is the bottom row.



IMPORTANT: This library uses Timer2 (for AVR-based boards like Arduino Uno and Mega2560)/Timer1 (for PIC32-based boards like the chipKits)/Timer Counter 1, Channel 0 (for Arduino Due) for the marquee interrupt. This may cause conflicts with other libraries. The timer interrupt is only used while the marquee is running. If you are not using the marquee you do not have to worry about this.

You can always find the latest version of the library at <http://www.RinkyDinkElectronics.com/>

For version information, please refer to `version.txt`.

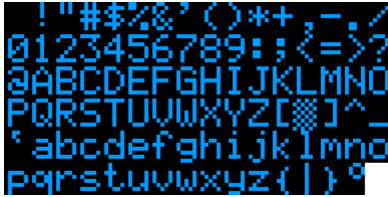
This library is licensed under a [CC BY-NC-SA 3.0](https://creativecommons.org/licenses/by-nc-sa/3.0/) (Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported) License.


For more information see: <http://creativecommons.org/licenses/by-nc-sa/3.0/>

Defined Literals:

Alignment
For use with scroll()
SCROLL_LEFT: 1
SCROLL_RIGHT: 2
SCROLL_UP: 3
SCROLL_DOWN: 4

Included Fonts:

TextFont

Character size: 6x8 pixels
Number of characters: 95

NumFont

Character size: 6x8 pixels
Number of characters: 18
NOTE: These characters are the same characters used in the TextFont with the exception of the semicolon (;) which has been replaced by a space. This font is intended for use when you only need numbers and some other relevant characters and want to save a little space in flash memory.

Functions:

LEDmatrix7219(data, clock, load);	
The main class constructor.	
Parameters:	data: Pin for MAX7219 DIN signal (MAX7219 Pin 1) clock: Pin for MAX7219 CLK signal (MAX7219 Pin 13) load: Pin for MAX7219 LOAD signal (MAX7219 Pin 12)
Usage:	LEDmatrix7219 myMatrix(5, 6, 7); // Start an instance of the LEDmatrix7219 class
begin([num]);	
Initialize the MAX7219 array.	
Parameters:	num: <optional> The number of MAX7219/LED matrices attached (1-8) Default is 4
Usage:	myMatrix.begin(); // Initialize the array
Notes:	The MAX7219 array will be in sleep mode after it has been initialized.
enableSleep();	
Put the MAX7219 array in Sleep Mode.	
Parameters:	None
Usage:	myMatrix.enableSleep(); // Put the MAX7219 array into Sleep Mode
disableSleep();	
Re-enable the MAX7219 array after it has been put in Sleep Mode.	
Parameters:	None
Usage:	myMatrix.disableSleep(); // Wake the MAX7219 array from Sleep Mode
enableTest();	
Put the MAX7219 array in Test Mode.	
Parameters:	None
Usage:	myMatrix.enableTest(); // Put the MAX7219 array into Test Mode
disableTest();	
Take the MAX7219 array out of Test Mode.	
Parameters:	None
Usage:	myMatrix.disableTest(); // Return the MAX7219 array to normal Mode
setIntensity(level);	
Set the intensity/brightness of the LED matrices.	
Parameters:	level: The required intensity level (0-15)
Usage:	myMatrix.setIntensity(15); // Set the intensity to maximum
Notes:	The library defaults to intensity level 8.

`clear();`

Clear the LED matrices.

Parameters: None

Usage: `myMatrix.clear(); // Clear the LED matrices`

`setPixel(x, y);`

Turn on the specified LED.

Parameters: x: X-coordinate of the pixel
 y: Y-coordinate of the pixel

Usage: `myMatrix.setPixel(0, 0); // Turn on the upper left LED`

`clrPixel(x, y);`

Turn off the specified LED.

Parameters: x: X-coordinate of the pixel
 y: Y-coordinate of the pixel

Usage: `myMatrix.clrPixel(0, 0); // Turn off the upper left LED`

`invPixel(x, y);`

Invert the state of the specified LED.

Parameters: x: X-coordinate of the pixel
 y: Y-coordinate of the pixel

Usage: `myMatrix.invPixel(0, 0); // Invert the upper left LED`

`scroll(direction[, wrap]);`

Scroll the contents of the LED matrix 1 pixel in the chosen direction.

Parameters: direction: Direction to scroll. Use `SCROLL_LEFT`, `SCROLL_RIGHT`, `SCROLL_UP` or `SCROLL_DOWN`
 wrap: `<optional>`
 Wrap pixels disappearing off one edge around to the opposite edge
 Default is `false`

Usage: `myMatrix.scroll(SCROLL_DOWN); // Scroll the matrix 1 pixel down leaving an empty row at the top`

`print(st, x[, y]);`

Print a string at the specified coordinates.

Parameters: `st`: The string to print
 `x`: X-coordinate of the upper, left corner of the first character
 `y`: **<optional>**
 Y-coordinate of the upper, left corner of the first character
 Default is 0 (top line)

Usage: `myMatrix.print("Hello", 0); // Print "Hello" starting at the left edge of the LED matrix`

Notes: The string can be either a char array or a String object

`print(num, x[, y[, length[, filler]]]);`

Print an integer number at the specified coordinates.

Parameters: `num`: The value to print (-2,147,483,648 to 2,147,483,647) **INTEGERS ONLY**
 `x`: X-coordinate of the upper, left corner of the first digit/sign
 `y`: **<optional>**
 Y-coordinate of the upper, left corner of the first digit/sign
 Default is 0 (top line)
 `length`: **<optional>**
 Minimum number of digits/characters (including sign) to display
 `filler`: **<optional>**
 Filler character to use to get the minimum length. The character will be inserted in front
 of the number, but after the sign. Default is ' ' (space)

Usage: `myMatrix.print(num, 0); // Print the value of "num" starting at the left edge of the LED matrix`

Notes: This library **does not** support printing floating point numbers

`setFont(fontname);`

Select font to use with `print()`.

Parameters: `fontname`: Name of the array containing the font you wish to use

Usage: `myMatrix.setFont(TextFont); // Select the font called TextFont`

Notes: You must declare the font-array as an external or include it in your sketch.
 This library supports fonts made with the online Font Maker (Monochrome)

marquee(st, speed);

Start an interrupt-driven marquee that will scroll the chosen string across the entire matrix from right to left.

Parameters: st: The string to scroll
 speed: The number of milliseconds between each step of the scroll

Returns: (boolean) **true** if OK, **false** if it failed to start

Usage: myMatrix.marquee("Hello", 100); // Start scrolling "Hello" across the entire matrix

Notes: The marquee will use the font that has been set when starting the marquee
 The string can be either a char array or a String object

stopMarquee();

Stop a running marquee.

Parameters: None

Returns: (boolean) **true** if OK, **false** if it failed to stop

Usage: myMatrix.stopMarquee(); // Stop a running marquee

setMarqueeSpeed(speed);

Change the speed of a running marquee.

Parameters: speed: The number of milliseconds between each step of the scroll

Usage: myMatrix.setMarqueeSpeed(200); // Change the running speed of the marquee to one step every 200ms