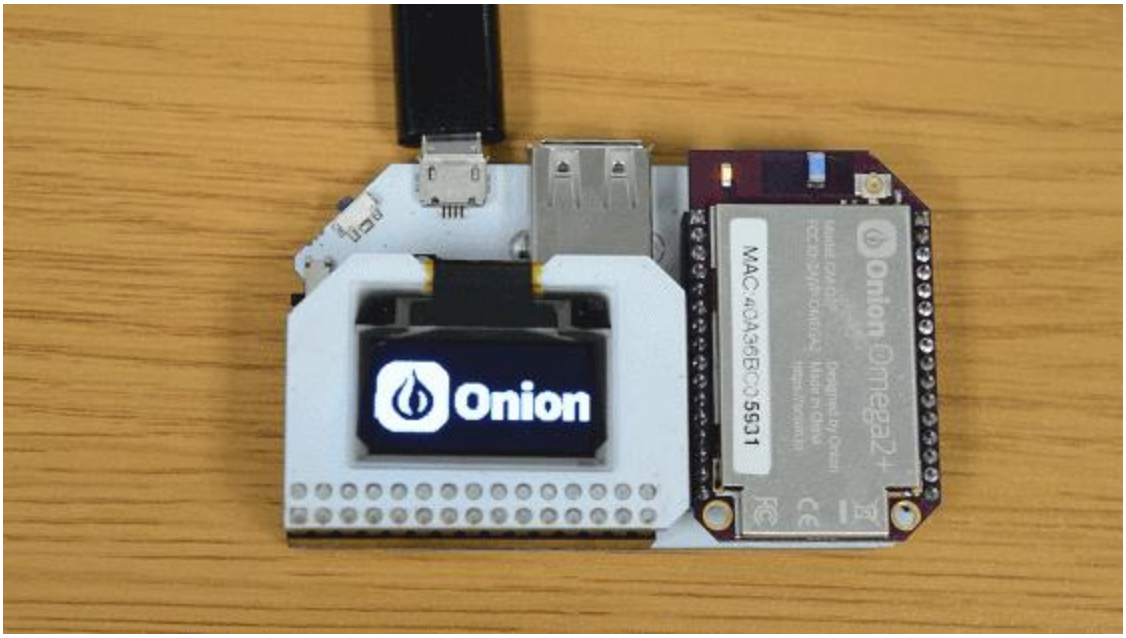
OLED Expansion

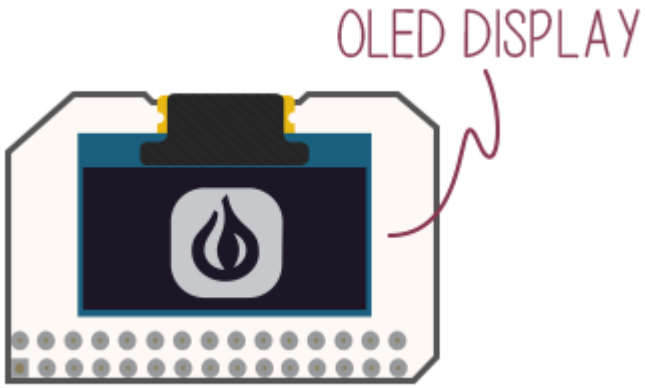
The OLED Expansion is an extremely power efficient 0.96″ monochrome (black and white) OLED display for your Omega. With a resolution of 128×64, it is very handy for displaying text, drawing images, and even animation!



This Expansion communicates with the Omega using the I2C protocol, it's I2C device address is `0x3c`. If you're curious, check out the article on I2C.

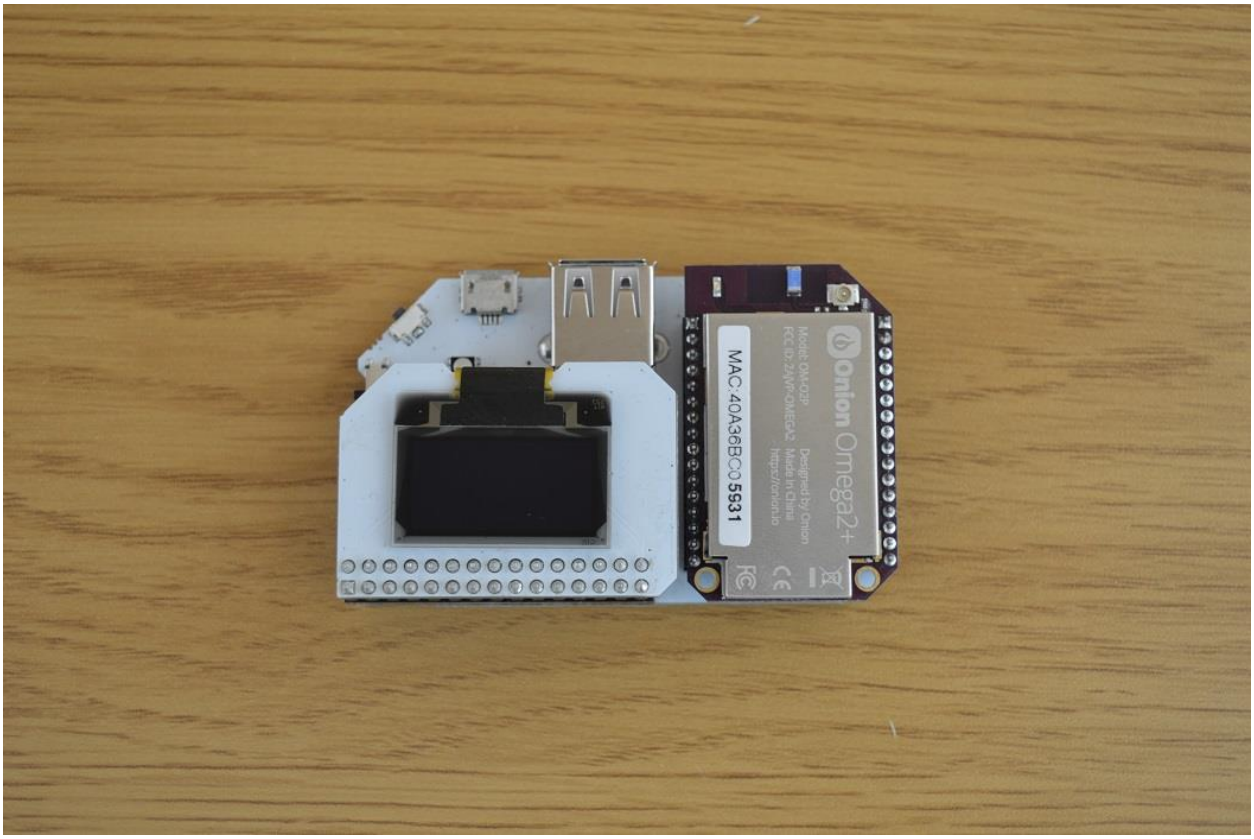
The Hardware

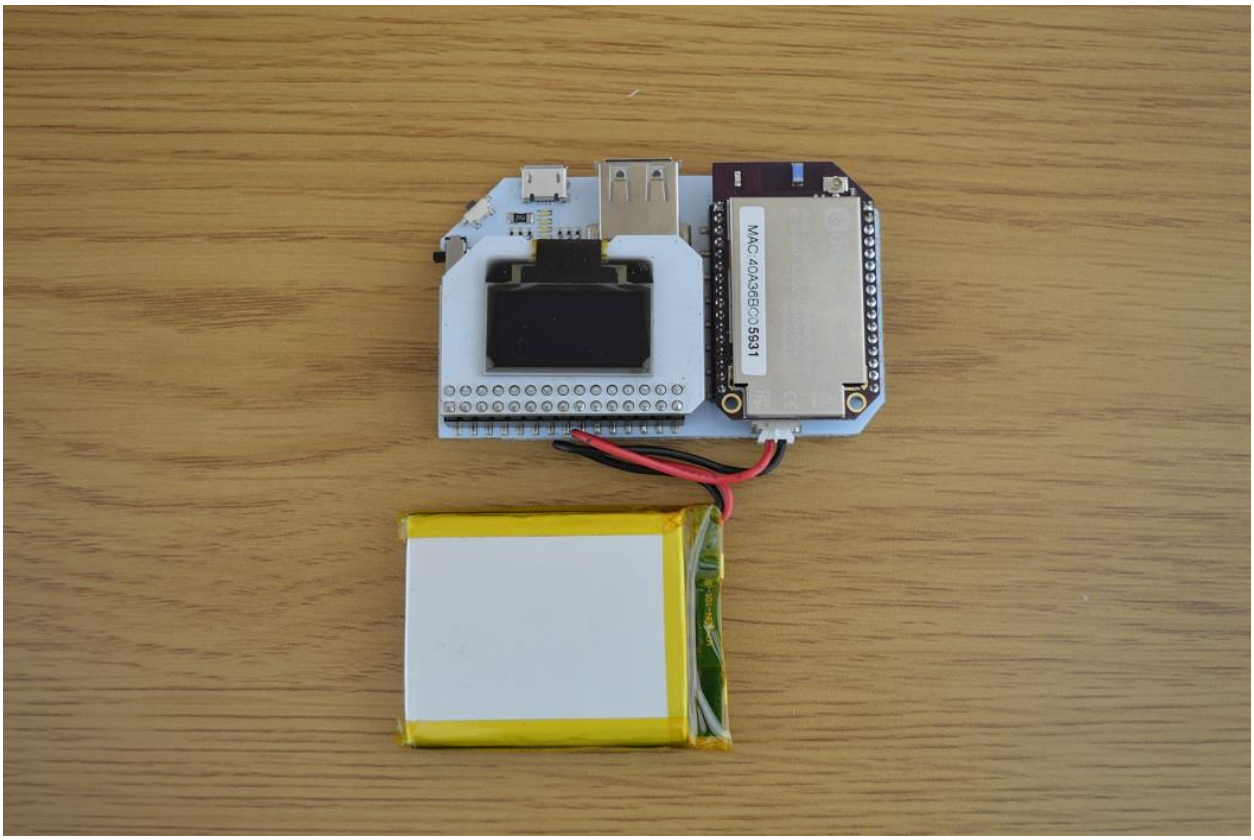
The main purpose of the OLED expansion is to display things on a screen. That's why the only significant hardware on the Expansion is an OLED screen.

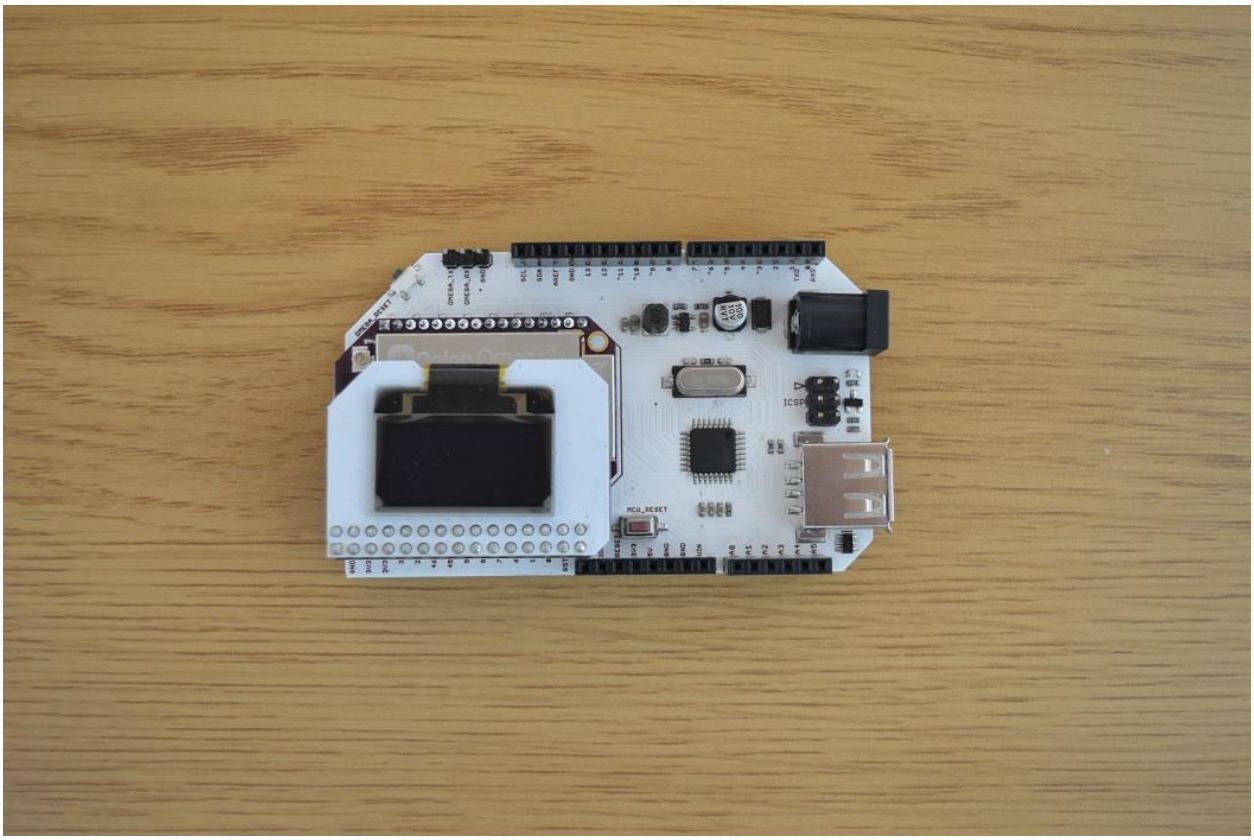


Connecting to a Dock

The OLED Expansion plugs into a Dock with an Expansion Header. You can also stack it on top of other Expansions and use them together.




MAC:40A36B005931


MCU_RESET

The OLED Expansion headers are blind, meaning you can't stack other Expansions on top. Otherwise you wouldn't be able to see the screen!

The Screen

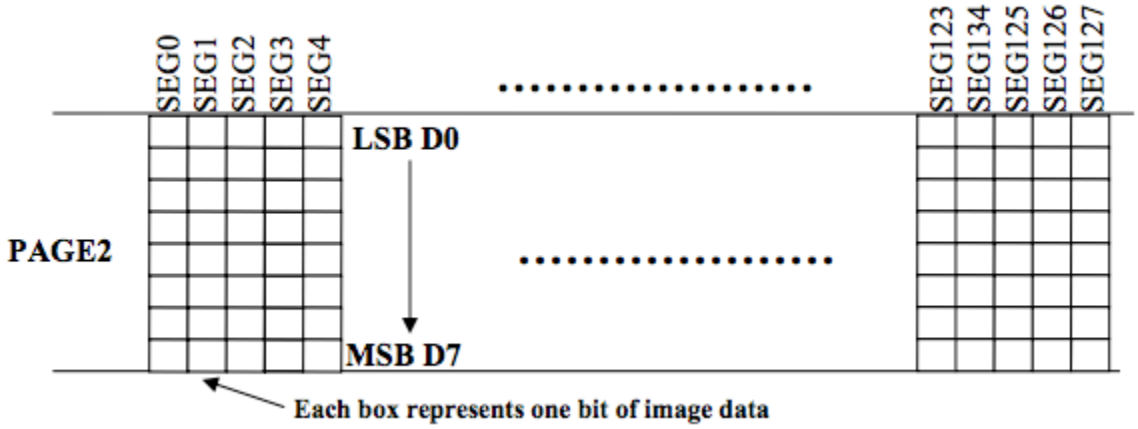
The screen is a 0.96″ OLED (organic light-emitting diode) screen and so the display makes its own light, no backlight is required. This reduces the power required to run the OLED and is why the display has such high contrast. The resolution of the screen is 128x64 (Length x Width), and when displaying text there are 8 rows, with 21 possible characters in each row.

Understanding the Display

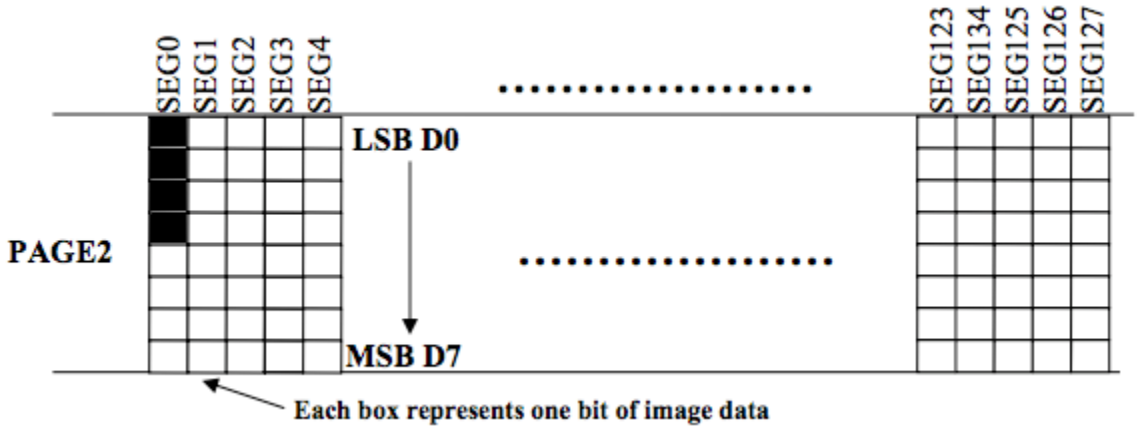
The OLED display is addressable by 128 columns and 8 rows (referred to as pages), as seen in the picture below:

	COL0	COL 1	…..	COL 126	COL 127
PAGE0					
PAGE1					
:	:	:	:	:	:
PAGE6					
PAGE7					

Each page consists of 8 pixel rows. When a byte is written to the display, the Least Significant Byte (LSB) corresponds to the top-most pixel of the page in the current column. The Most Significant Byte (MSB) corresponds to the bottom-most pixel of the page in the current column.



So writing `0x0f` to `SEG0` would produce the top 4 pixels being colored in, and the bottom 4 being left blank in that column:



Mechanical Drawings

We've made available a detailed diagram of the dimensions and geometry of the OLED Expansion.

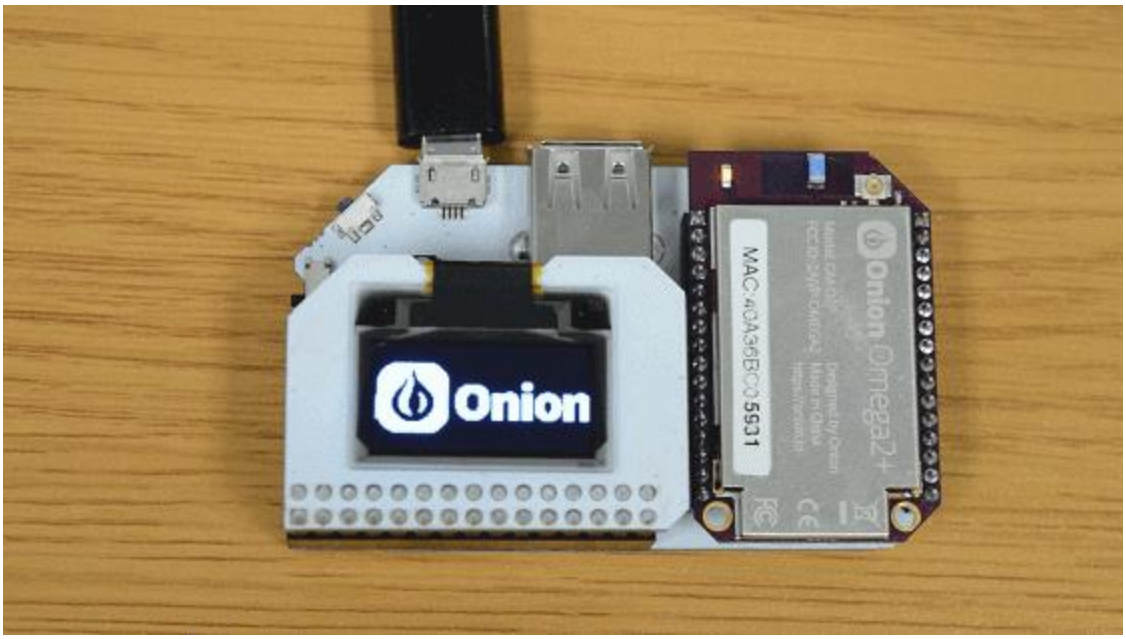
Using the OLED Expansion

The OLED Expansion is incredibly useful because it gives you the ability to display information using your Omega. It can be used in your projects to display things like you twitter feed, stock prices, or even just information about your Omega (disk space, memory usage).

Using the OLED Expansion

The OLED Expansion is an extremely power efficient 0.96″ monochrome (black and white) OLED display for your Omega. With a resolution of 128×64, it is very handy for displaying text, drawing images, and even animation!

We've developed a command line tool called `oled-exp` that will be your helper in all things related to the OLED Expansion.
Also available are a C library and a Python module that allow you to develop your own programs to control the OLED Expansion. See the guides at the bottom of this article for more details.



You can learn more about the technical specifications of the OLED expansion in our OLED Expansion hardware overview

Command Usage

To use your OLED display you'll first need to initialize the display:

```
oled-exp -i
```

If you do not initialize your display then your OLED Expansion will not respond to other commands.

For a print-out of the command's usage, run it with only a `-h` argument:

```
oled-exp -h
```

In general, the commands will be in this form:

```
oled-exp [OPTIONS] <COMMAND> <PARAMETER>
```

Command and Option Table

To reconcile all of the above features to `oled-exp`, refer to the table below:

Action	Option/Command
initialization	`-i`
clear the display	`-c`
toggle display on or off	`power <on/off>`

Action	Option/Command
invert colours	`invert <on/off>`
dim the screen	`dim <on/off>`
set the cursor	`cursor <row>,<column>`
set the cursor by pixel	`cursorPixel <row>,<pixel>`
write text	`write <string>`
write a single byte	`writeByte <byte>`
enable scrolling	`scroll <direction>`
display an image	`draw <lcd file>`

Options

The option arguments can be run by themselves or in conjunction with any other command. They will always be executed first.

Initialization

To initialize the display to accept further commands:

```
// just initialize the display
oled-exp -i

// initialize the display and write a message
oled-exp -i write "Freshly initialized"
```

Clear the Screen

To clear the screen and set the cursor to the top left:

```
// just clear the screen
oled-exp -c

// clear the screen and draw an image
oled-exp -c draw /root/onion.lcd
```

Commands

The commands are more complex and each requires one argument.

Power

Turn the display on or off. Can be used to toggle the display after it has been initialized.

```
oled-exp power <on|off>
```
Note that any text or images on the screen will be preserved.

Invert

Invert black and white on the display. Setting to `on` will enable the invert, setting to `off` will disable the inversion.
```
oled-exp invert <on|off>
```

Dim

Enable dimming the display. Setting to `on` will dim the display, setting to `off` will restore the default brightness.
```
oled-exp dim <on|off>
```

Write

Write a string to the display *at the current position of the cursor*:

```
oled-exp write <string>
```
If the string contains spaces or any special characters (newlines, !?(){}[] etc), it needs to be wrapped with double quotes:

```
oled-exp write "This is so exciting!\nIsn't it?!?!\n\n(I love the Omega)"
```
A list outlining the supported characters can be found in a section below.

Write a Single Byte

Write a single byte of data (eight vertical pixels) to the display *at the current position of the cursor*:

```
oled-exp writeByte <byte>
```
The Least Significant Bit (LSB) in the byte corresponds to the top-most pixel in the column, the Most Significant Bit (MSB) corresponds to the bottom-most pixel in the column.

The cursor will be incremented to the next pixel column after this command.

For example, the following command:

```
oled-exp writeByte 0x0f
```
Will fill in the top four pixels and will leave the bottom four pixels in a column blank.

`0x0f` reads as `00001111` in binary.

Set the Cursor Position

Set the cursor position on the display, any writes after this command will start at the specified row and column.

The `row` parameter represents each character row (8 pixel rows) on the display, so the range is **0 to 7**
The `column` parameter represents each character column, the range is **0 to 20**

```
oled-exp cursor <row>,<column>
```

Examples

To write `Hello` on the first line, and then `How are you?` on the 4th row:

```
oled-exp write Hello cursor 3,0 write "How are you?"
```

Set the cursor to the start of the last character row:

```
oled-exp cursor 7,0
```

Set the cursor to the middle of the 2nd row:

```
oled-exp cursor 2,10
```

Set the Cursor Position by Pixel Column

Set the cursor position on the display, any writes after this command will start at the specified row and **pixel**.

The `row` parameter represents each character row (8 pixel rows) on the display, so the range is **0 to 7**
The `pixel` parameter represents each pixel column, the range is **0 to 127**

```
oled-exp cursorPixel <row>,<pixel>
```

Examples

To write `Hello` a quarter of the way through the first line, and then `How are you?` in the middle of the 4th row:

```
oled-exp cursorPixel 0,31 write Hello cursorPixel 3,63 write "How are you?"
```

Set the cursor to the start of the last character row:

```
oled-exp cursorPixel 7,0
```

Set the cursor to the middle of the 2nd row:

```
oled-exp cursorPixel 2,63
```

Scrolling

The OLED can scroll whatever is on the screen horizontally or diagonally upwards. The text/image will wrap around the edges.

```
oled-exp scroll <direction>
```

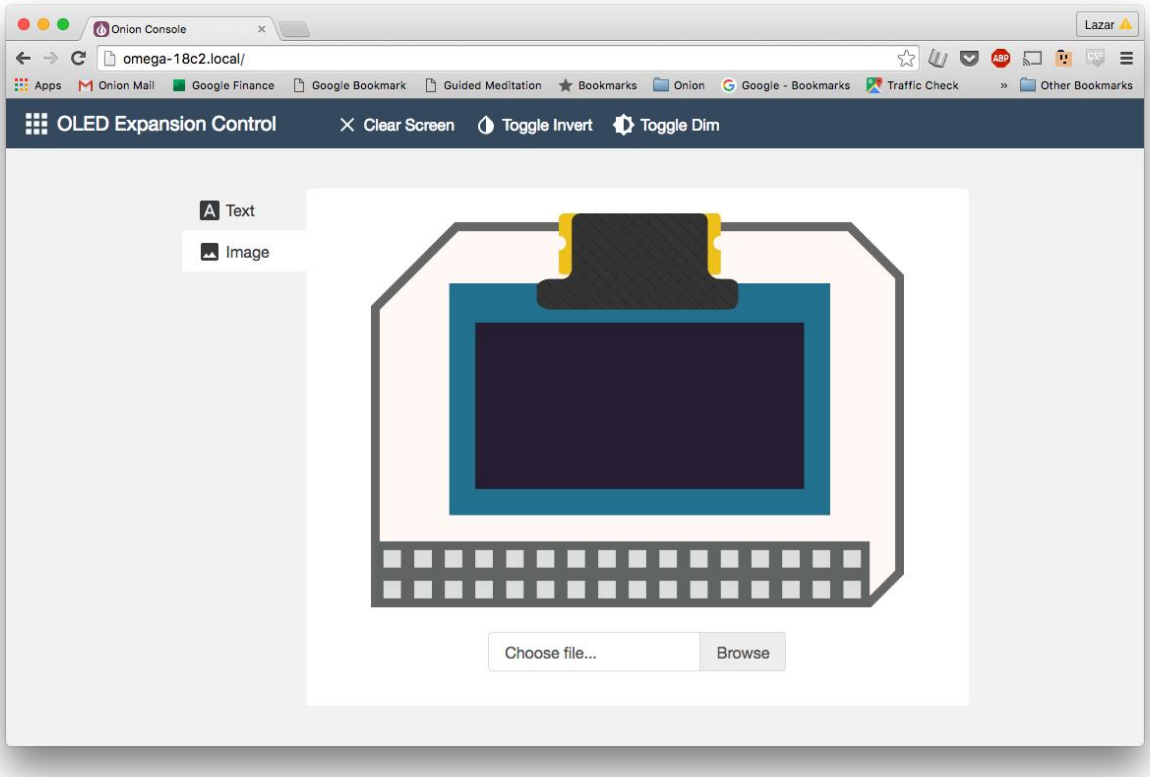
Supported directions: * left * right * diagonal-left * diagonal-right * stop * This is to disable scrolling.

Drawing Images on the Display

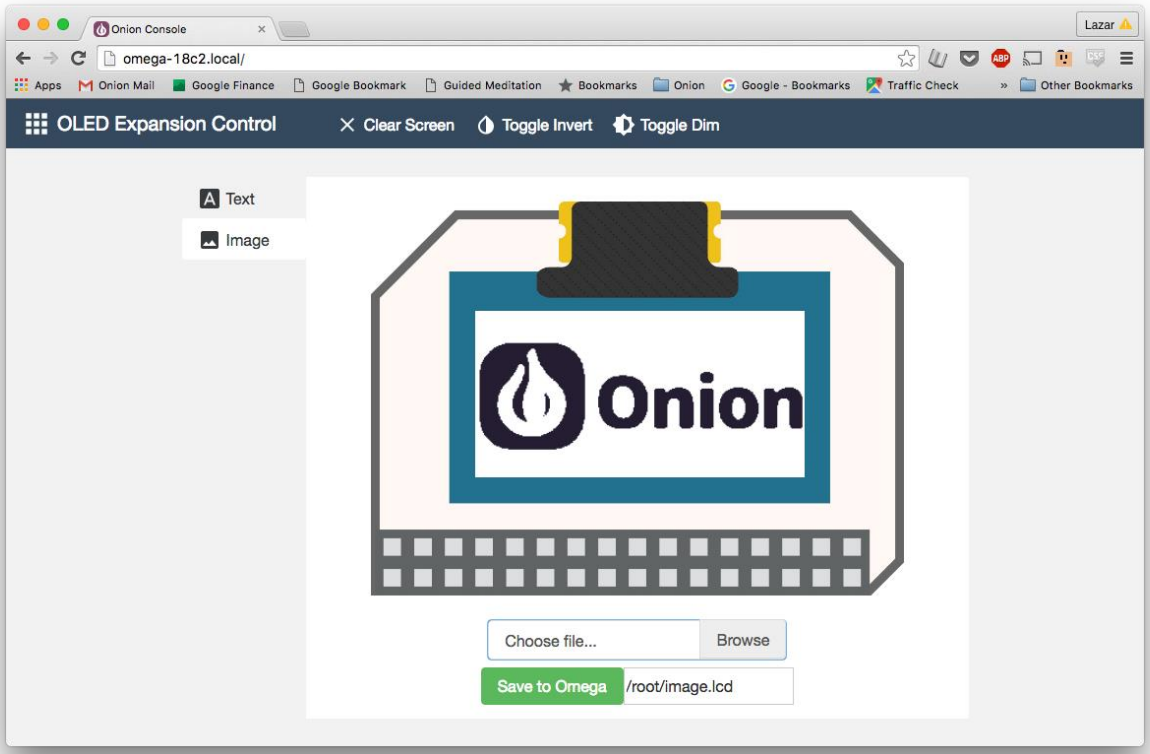
The OLED Screen can also be used to display images. The Console can be used to convert existing images into a format that is compatible with the OLED Expansion and save the output to an Omega. Functions in the C library can read the image data and display it on the OLED Expansion. Alternatively, a buffer can be created programatically and displayed on the OLED.

Creating Image Files

The Console OLED App can be used to create OLED Expansion compatible image files. Navigate to the Image tab of the OLED App:



Once an image has been selected, a button and form will appear that allow you to save the OLED image file to your Omega:



After the image name and location are selected, click the **Save to Omega** button.

OLED Image File Details

The OLED image files store the image data as 1024 bytes represented in hexadecimal. Each byte represents eight vertical pixels, with the first 128 bytes representing the columns in Page 0, the following 128 bytes representing the columns in Page 1, and so on.

If this is unclear, see the Understanding the Display Section in the OLED Expansion Library documentation article for details on how the display is addressed.

Displaying Images from a File

To display the OLED image file on the OLED Expansion:

```
oled-exp draw <path to oled image file>
```

Example

Draw an image saved to `/root/image.lcd` on the display:

```
oled-exp draw /root/image.lcd
```

Initialize the display and draw an image saved to `/tmp/onion.lcd`:

```
oled-exp -i draw /tmp/onion.lcd
```

Supported Characters

The OLED Display supports the following characters:

```
SPACE, !, ", #, $, %, &, ', (, )
, * , + , , , - , . , / , 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , a , b , c , d ,
e , f , g , h , i , j , k , l , m , n , o , p , q , r , s , t , u , v , w , x , y
, z , A , B , C , D , E , F , G , H , I , J , K , L , M , N , O , P , Q , R , S ,
T , U , V , W , X , Y , Z , : , ; , < , = , > , ? , @ , [ , \ , ] , ^ , _ , ` , {
, | , } , ~
```

Using the Libraries

The C library and Python module will give you the flexibility to use the OLED Expansion however you want in your own programs.

For more information, see our guide to using the OLED Expansion C Library, or our guide to using the OLED Expansion Python Module.