Grove OLED Display (I2C) Guide

By Team WiFi (Chris Kuan, Alan Lee, Matheson Mawhinney, & Ivan Ngan), Fall 2016

This document guides the user through:

1. Connecting and setting up the OLED display
2. Downloading libraries to help with OLED development
3. C programming example for OLED display

Formatting and Terms

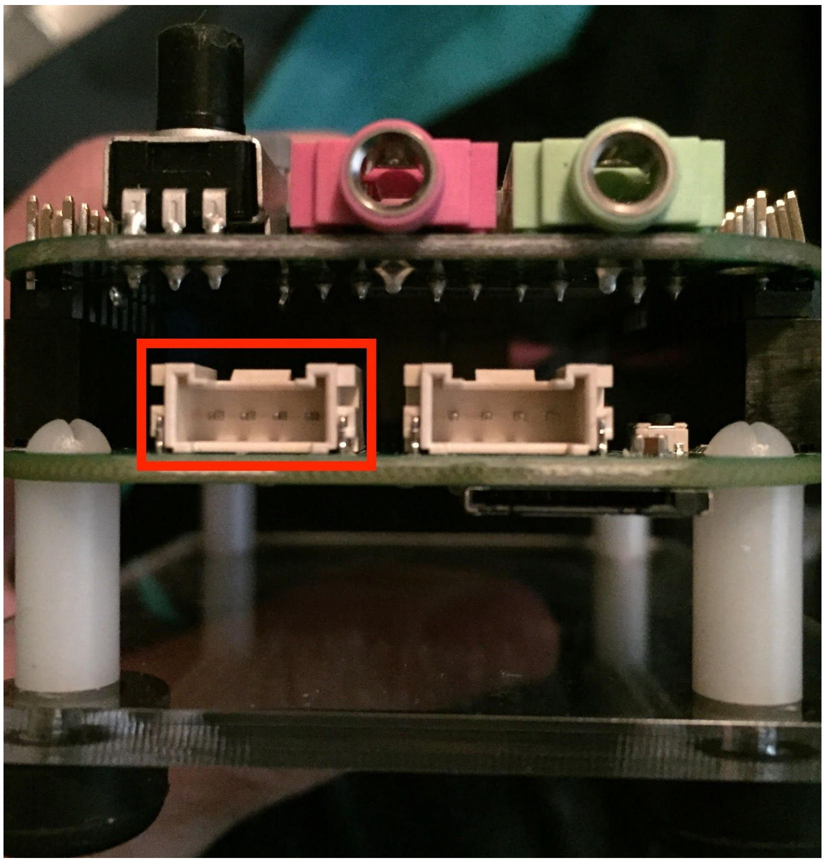
- Host (desktop) commands starting with $ are Linux console commands:
  ```
  $ echo “Hello world”
  ```
- Target (board) commands start with #:
  ```
  # echo “On embedded board”
  ```
- Almost all commands are case sensitive in Linux and U-Boot

Introduction

The BeagleBone Green we're using has two onboard Grove connections. Be careful to confirm with the hardware schematics first because one of them utilizes UART and the other utilizes I2C for interfacing. We will want to connect to the I2C one, which should be on the P9 header at pins #19 and #20.

Setup

Connecting the OLED display to the BeagleBone is a relatively simple process. Take the included Grove cable and plug one end into the display and the other end into the I2C Grove port on the BeagleBone. The cable and connector are keyed so that they only plug in one way, meaning there is no need to make sure the cable connects to the correct pins in the connectors.



On the BeageBone Green, use the highlighted Grove connector to connect the display

Warning: Ensure that the BeagleBone is not powered on when you plug the OLED display in. Doing so risks damaging the the BeagleBone's OS, the board itself, or the display.

Setting up the Bus

1. The I2C Grove connector runs off the I2C2 bus on the board and is likely enabled on board startup. You can confirm this by running this command:

```
# i2cdetect -l
I2C-0 i2c        OMAP I2C adapter                  I2C adapter
I2C-2 i2c        OMAP I2C adapter                  I2C adapter
```

2. If the I2C2 bus is not enabled, run the following command to enable it:

```
# echo BB-I2C2 > /sys/devices/platform/bone_capemgr/slots
```

   Re-run the first command to ensure it’s enabled:

```
# i2cdetect -l
I2C-0 i2c        OMAP I2C adapter            I2C adapter
I2C-2 i2c        OMAP I2C adapter            I2C adapter
```

3. With the display connected to the board, run the command below to ensure it is being detected:

```
# i2cdetect -y -r 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- 18 -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- UU UU UU UU -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

   You should get an output that looks similar to that above. Specifically you should see **3c** under column **c**, row **30**, as that is the address for the OLED display.

Libraries and Cross-compiling

Controlling the OLED display requires no additional libraries aside from those in the C standard library and the I2C library included with Linux. All interactions with the display are controlled by sending commands and data to the device over the I2C bus.

C Programming

1. Working with the Provided Library

It is recommended to use the library provided by Seeed Studio as a starting point for working with the display. Doing so saves from having to decipher the display's datasheet and schematics to figure out which registers correspond to which commands. In addition, the library provides a basic font in hex as well as the code needed to print the characters to the screen, all necessary initialization commands, and various commands to change the positioning and greyscale of text.

The provided library can not be used on the BeagleBone immediately though; it was written for an Arduino board and makes use of Arduino specific libraries that are not on the BeagleBone and will thus have to be rewritten for our purposes.

2. Initialization

Just like with any other I2C device, it must first be initialized so a file descriptor can be obtained to send our commands/data to.

```
// Function to initalize the I2C device for use
// char *bus => the I2C device to use, “/dev/i2c-2”
// int address => the address for the device, 0x3C
int initI2cBus(char *bus, int address) {
    int fd = open(bus, O_RDWR);
    if(fd < 0) {
        printf("i2c: Unable to open bus for read/write (%s)\n",
bus);
        exit(-1);
    }

    int result = ioctl(fd, I2C_SLAVE, address);
    if(result < 0) {
        printf("i2c: Unable to set i2c device to slave
address\n");
        exit(-1);
    }
    return fd;
}
```

3. Sending Commands

In order to send commands, we must rewrite the library’s sendCommand function

```
#define COMMAND_MODE 0x80
// cmd will be a hex code specifying a command defined in the
// display’s datasheet
void sendCommand(unsigned char cmd) {
    char buf[2];
    buf[0] = COMMAND_MODE;
    buf[1] = cmd;
    int result = write(i2c_fd, buf, 2);
    if(result != 2) {
        printf("Unable to write to i2c register\n");
        exit(-1);
    }
}
```

4. Sending Data

Like with sending commands, we must rewrite the sendData function

```
#define DATA_MODE 0x40
// c will contain the data you wish to display
void sendData(unsigned char c) {
    char buf[2];
    buf[0] = DATA_MODE;
    buf[1] = c;
    int result = write(i2c_fd, buf, 2);
    if(result != 2) {
        printf("Error sending data...\n");
        exit(-1);
    }
}
```

5. Initializing the Display

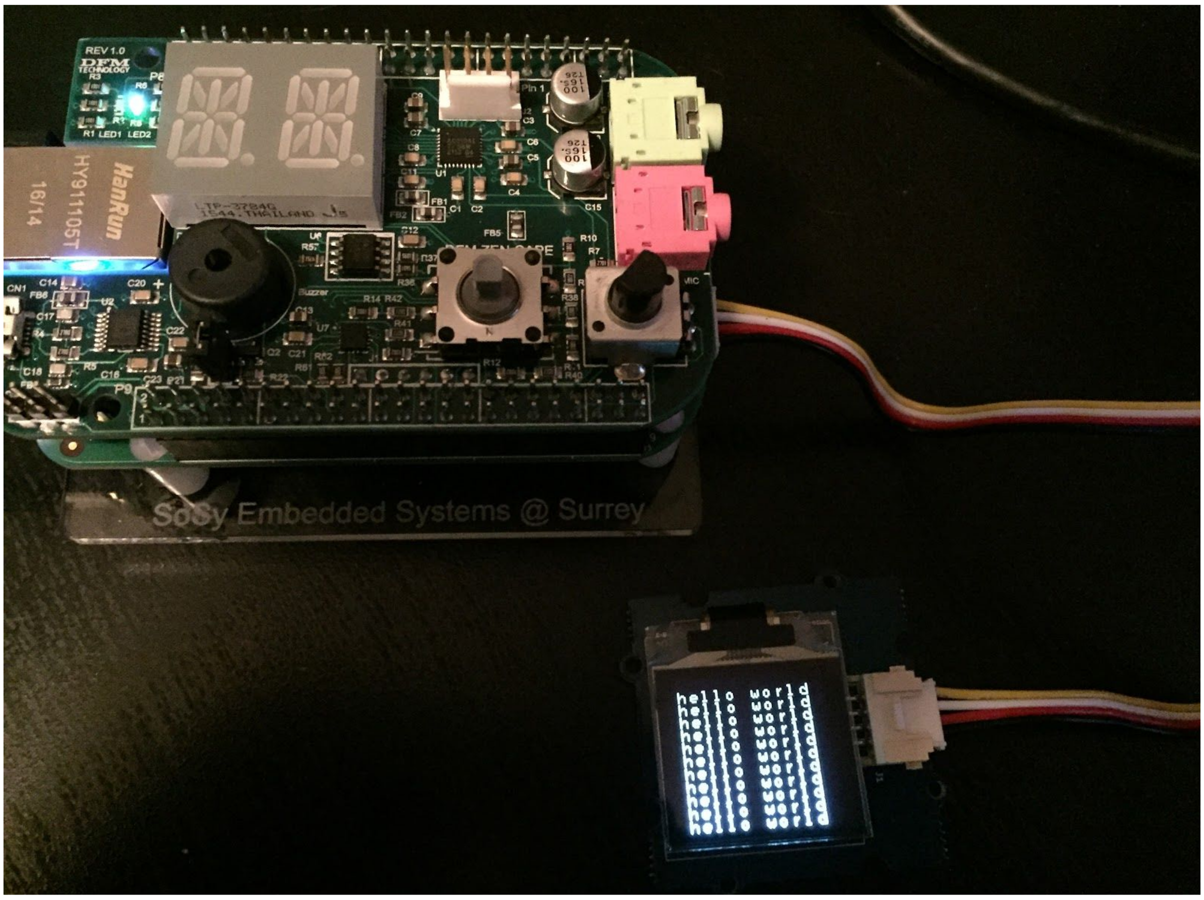
Before attempting to print anything to the display, you must first call the provided `init()` function which sends various commands to the display to prep it for displaying data. The effects of each of these commands can be found in the display’s data sheet and can be adjusted to suit your needs.

6. Example Program

```
// The following will display the string “hello world” 12 times,
// once per line
int main() {
    i2c_fd = initI2cBus(I2C_BUS_2, I2C_ADDRESS);\

    init();
    clearDisplay();
    setVerticalMode();

    for (int i = 0; i < 12; i++) {
        setTextXY(i,0);
        setGrayLevel(i);
        putString("hello world");
    }
}
```



Display after running the sample program

7. clearDisplay()

If you experience issues with the `clearDisplay()` functions not reliably clearing the entire screen, add the following code to the beginning of the function:

```
// Row Address
sendCommand(0x75);     // Set Row Address
sendCommand(0x00);     // Start 0
sendCommand(0x5f);     // End 95

// Column Address
sendCommand(0x15);     // Set Column Address
sendCommand(0x08);     // Start from 8th Column of driver IC.
                       // This is 0th Column for OLED

sendCommand(0x37);     // End at  (8 + 47)th column.
                       // Each Column has 2 pixels(segments)
```

Useful References

1. http://wiki.seeed.cc/Grove-OLED_Display_1.12inch/
   Contains the sample library and relevant data sheets