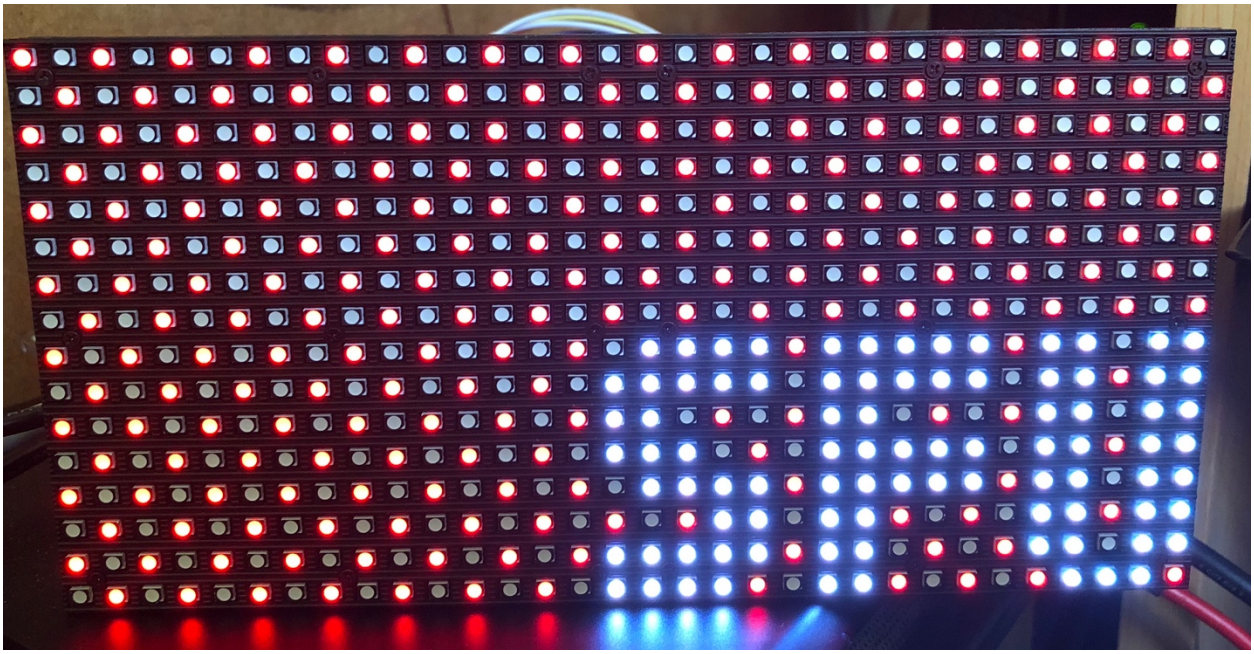
Adafruit 16x32 RGB LED Matrix Guide for BeagleBone Green



Simon Fraser University

Written by: Raymond Chan

Table of Contents

Introduction

The purpose of this guide is to aid students in setting up Adafruit's 16x32 RGB LED Matrix with the BeagleBone Green (BBG) via hardware and software. No prior technical knowledge is required; however, caution and common sense is a necessity. This will also serve as an update to an existing guide created for the BeagleBone Black (BBB) model linked below. Although the BBB variant is similar to the BBG, some of the steps are unnecessary if you're using a BBG due to hardware differences. Likewise, no guides exist online for connecting the display with the BBG as of the writing of this guide. Furthermore, any figures or explanations taken from the BBB guide written by Janet Mardjuki will be explicitly mentioned.

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2015-student-howtos/Adafruit16x32LEDMatrixGuideForBBB.pdf

Prerequisites

The following prerequisites have been broken down into two sections of either being required or totally optional. Canadian based websites that I have personally used with extremely fast shipping and reliable customer service has also been provided to save your wallet of the expensive international fees that await you if you decide to ship from the U.S.

Required

a. *Premium Female/Male 'Extension' Jumper Wires x1*
   (https://www.digikey.ca/product-detail/en/adafruit-industries-llc/1954/1528-1964-ND/6827087)
b. *AC/DC Wall Mount Adaptor 5V 10W*
   (https://www.digikey.ca/product-detail/en/SWI10-5-N-P5/102-4136-ND)
c. *CBL 2.1X5.5MM JACK-GATOR CLIPS*
   (https://www.digikey.ca/product-detail/en/10-01597/839-1219-ND)

d. *ADAFRUIT 16X32 RGB LED MATRIX*
(https://www.digikey.ca/product-detail/en/420/1528-1412-ND)

Optional

a. *Arduino Stackable Headers – 8 Pin, Pkg/4*
(https://www.rpelectronics.com/ls-00008.html)

Setup

Without going into all the technical jargon, I will simply identify what each wire's functionality is, and once you are comfortable we can dive straight into showing you the quickest and easiest way to wire this beauty up.

Label	Signal	Description
R1	Red1	Red colour signal for the top half of the matrix (x-index:0-7)
B1	Blue1	Blue colour signal for the top half of the matrix (x-index:0-7)
G1	Green1	Green colour signal for the top half of the matrix (x-index:0-7)
R2	Red2	Red colour signal for the top half of the matrix (x-index:8-15)
B2	Blue2	Blue colour signal for the top half of the matrix (x-index:8-15)
G2	Green2	Green colour signal for the top half of the matrix (x-index:8-15)
A	Row A	MSB of the row bits for the row selection
B	Row B	Middle bit of the row bits for the row selection
C	Row C	LSB of the row bits (for the 16x32 Matrix) for the row selection
D	Row D	LSB of the row bits (for anything bigger than 16x32 Matrix) for the row selection
CLK	Clock	Ends of each bit of the data (each pixel)
LAT	Latch	Ends of row of data
OE	Output Enable	LED on/off (enable/disable), for row transition
GND	Ground	Signal to ground

Figure 0. Table of signals and their functionality taken from Janet Mardjuki's 16x32 LED Matrix Guide for BeagleBone Black.

Ribbon Cable

R1	G1
B1	GND
R2	G2
B2	GND
A	B
C	D
CLK	LAT
OE	GND

Figure 1. 16x32 pin arrangement head.

G1	R1
GND	B1
G2	R2
GND	B2
B	A
D	C
LAT	CLK
GND	OE

Figure 2. 16x32 pin arrangement end.

Figure 1 is the pin arrangement of the **INPUT** connector on the left side of the back of the display. While Figure 2 is the new pin arrangement after you plug in one end of the ribbon cable to the **INPUT** because the other end of the ribbon cable that is used to connect to the BeagleBone Green is actually flipped. Figure 3 is what the ribbon cable should look like when attached and 'flipped' at the end with jumper wires that is not attached to the display.

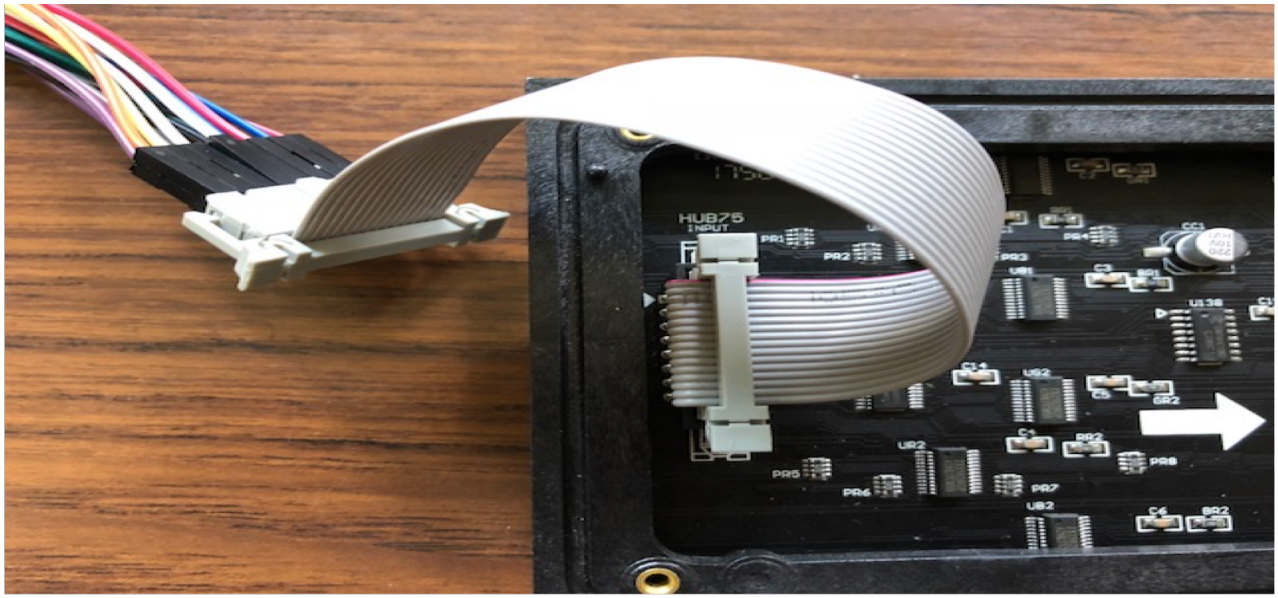


Figure 3. Ribbon cable connection which comes included with the display.

Jumper Wires

The next step is to take the other end of the ribbon cable that is not inserted into the display's pins and while referring to Figure 2 connect the 'male' end of your package of jumper wires into the slots of the ribbon cable. Note that it does not matter what color you use for each slot, but you should at least remember what each is connected to because it will be important in the following subsequent sections when you connect the 'female' ends of the jumper wires to the BeagleBone Green or ZenCape and use the sample code.

Figure 4 shows a closeup of the jumper wires attached to the other end of the ribbon cable not attached to the display. You will also notice that at the end of the jumper wires (female end) they are attached to Arduino stackable headers, this is technically not required but if the pins on the ZenCape are not long enough then you will require these to get around this issue.

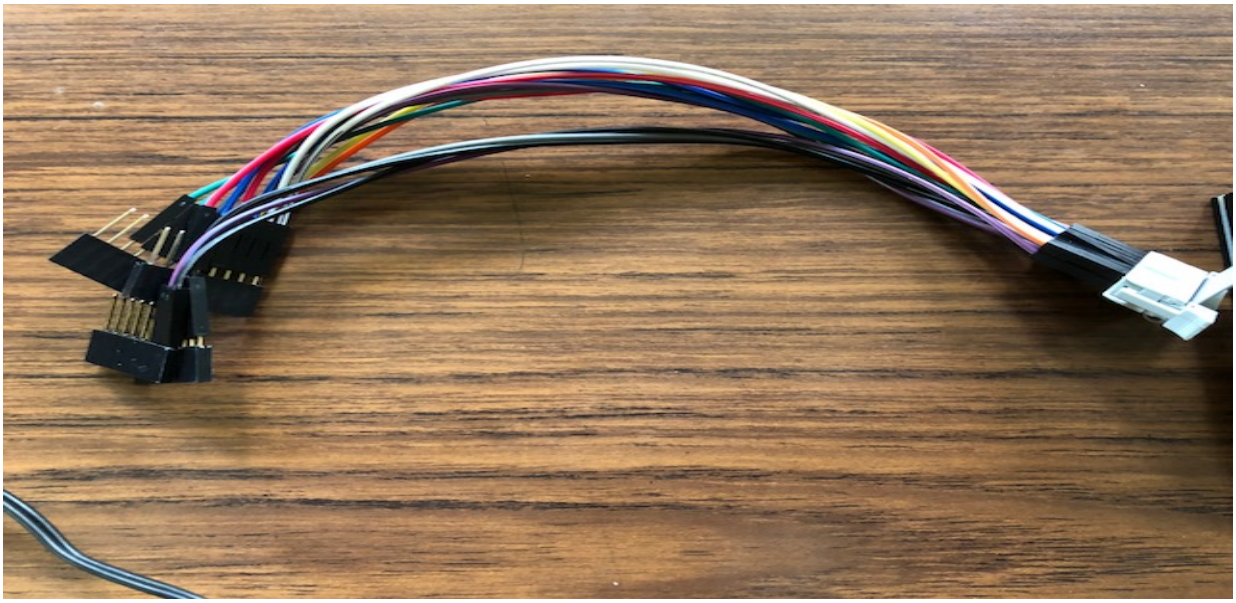


Figure 4. Jumper wires provided in the BBG package bought from Brian attached to ribbon cable along with optional Arduino stackable headers

Power

With the included Molex connector from the display, attach it to the back of the back of the display and with the red and black wires sticking out at the other end of the Molex connector

connect it with the gator clips (red to red, and black to black). Finally, with the ’female’ end of the gator clips, connect it with the end of the AC/DC power supply. The secondary Molex connector can be left dangling as shown in Figure 8 not connected to anything.

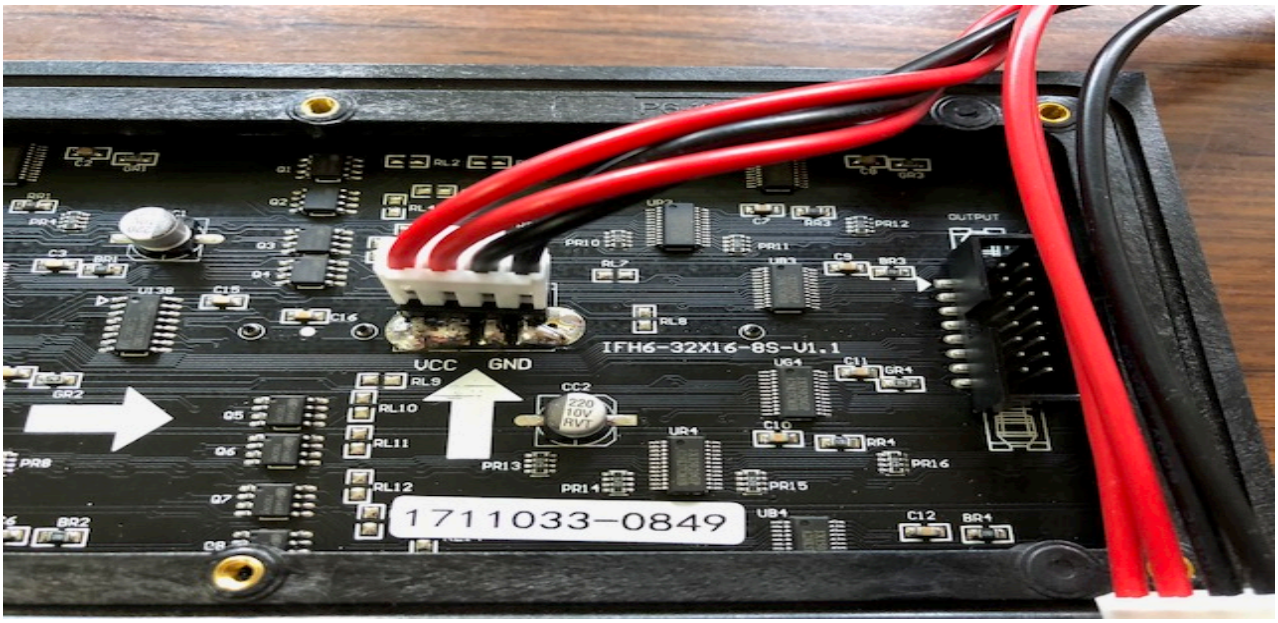


Figure 5. Molex connector included with purchase of display attached to the back.

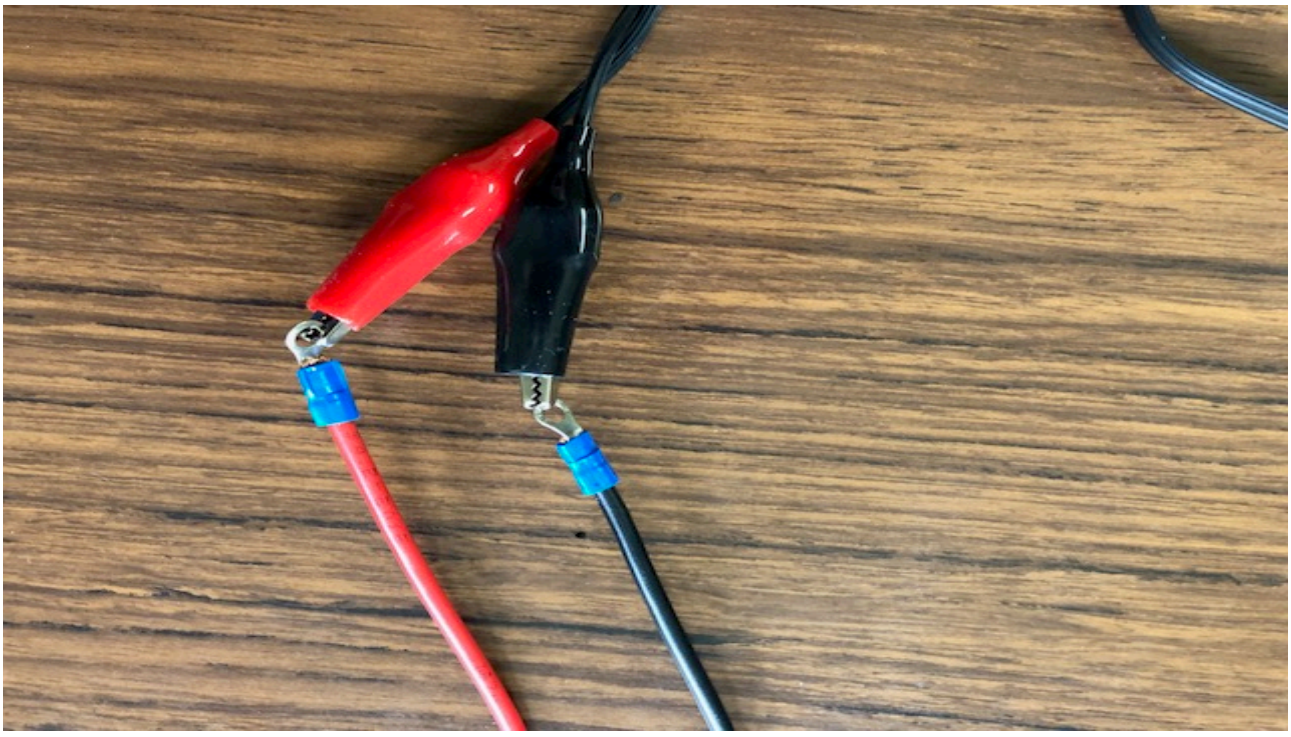


Figure 6. Gator clips attached to the other end of the Molex connector not attached to the display.

Figure 7. Power supply attached to the other end of the Gator clips.

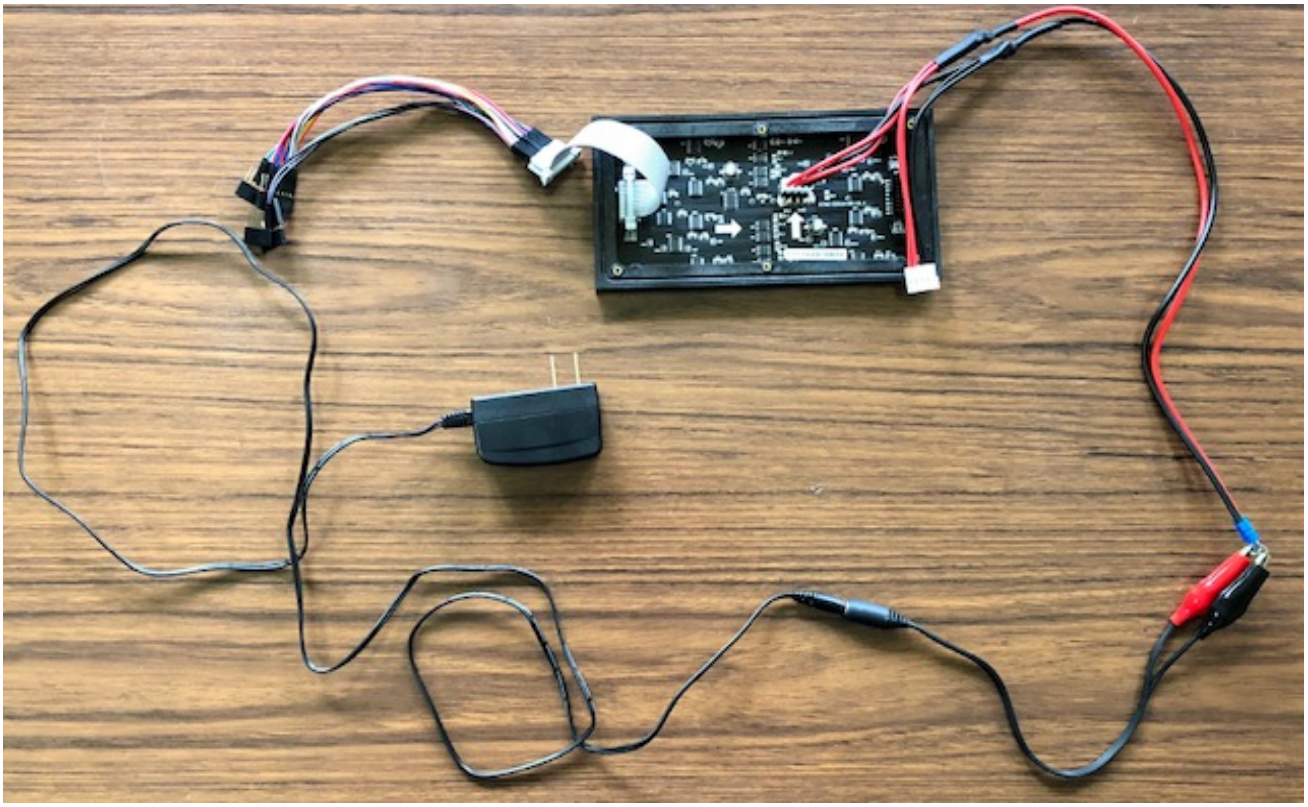


Figure 8. A complete look of what it should the wiring should look like so far into the guide.

BeagleBone Green

Assuming there is a ZenCape attached to the BeagleBone Green, the following Figure 9 below is the mapping of the jumper wires to the **[GPIO]** pins on the ZenCape that directly extend the GPIO pins of the BeagleBone Green of **P8**. The remaining mappings of the jumper wires to the **DGND** is also located on **P9**. Please refer to Brian's guide for a detailed picture of the **P8** and **P9** pins available and where they are located.

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/GPIOGuide.pdf

Note that using the ZenCape is the same thing as directly hooking it up with the BeagleBone Green, but you also get the extra functionality of the ZenCape such as the joystick, 16-segment display, potentiometer, and other functionalities which do not conflict with the GPIO setup this this guide uses

R1 [8]	G1 [80]
B1 [78]	G2 [79]
R2 [76]	B [77]
B2 [74]	LAT [75]
A [72]	CLK [73]
C [70]	OE [71]

GND [P9_43]
GND [P9_44]
GND [P9_45]
D [P9_46]

Figure 9. GPIO mapping to the flipped pin arrangement on Figure 2 and Pins to DGND on P9.

Figure 10. Jumper wires attached to optional Arduino stackable headers attached to ZenCape pin sections P8 and P9. These jumper wires in Figure 10 follow the mapping laid out in Figure 9 above.

Sample Code

Due to already existing sample code provided by Brian Fraser and written by Janet Mardjuki, this guide will reuse it but with minor modifications to ensure it compiles successfully because the original sample code will not work from the start. The already edited version of the sample code and makefile can be obtained from the following public GitHub link:

https://github.com/MontreaI/BeagleBone-Green-Adafruit-16x32-LED-Matrix-Sample-Code

If you have the non-edited version of the sample code that does not compile successfully from the start you may also follow the below steps to get it working as well. After following the steps it will be the same as downloading the modified version.

1. Replace "**#include "general.h"**" with "**#include <time.h>**"
2. There should already be an existing defined timeout value in nanoseconds called "**DELAY_IN_US**", and under that please define a new timeout value "**#define DELAY_IN_SEC 0**"
3. Control-f the following line "**sleep_usec(DELAY_IN_US);**" and replace it with the following lines:
   "**struct timespec reqDelay = {DELAY_IN_SEC, DELAY_IN_US};**"
   "**nanosleep(&reqDelay, (struct timespec *) NULL);**"

The reason why we do this is because the method **sleep_usec** comes from the **general.h** header file which is not available and appears to have been lost to time. Note that not even the original BBB guide provides it and requires you to modify the sample code as well.

Another important note to mention is that because the Adafruit's RGB display has no built-in PWM the display will have to be constantly refreshed in order to keep the LEDs on. If you stop refreshing the display will not show anything. Here are the general steps of what you need to do in order to have the display show something:

1. Fill the 2D screen array with a pixel color between 1-7 using **ledMatrix_setPixel(int x, int y, int colour)**. Currently only 7 colors are supported by the sample code with the 8th color being black which is just "off".
2. Refresh display with **ledMatrix_refresh()** inside a loop (recommended you create a thread that just refreshes the display in background).
3. If you wish to update the screen simply either set the desired LED coordinates to a different color (remember that color '0' is turning off a pixel). You will want to stop refreshing when attempting this operation or else you will get a weird LED effect.

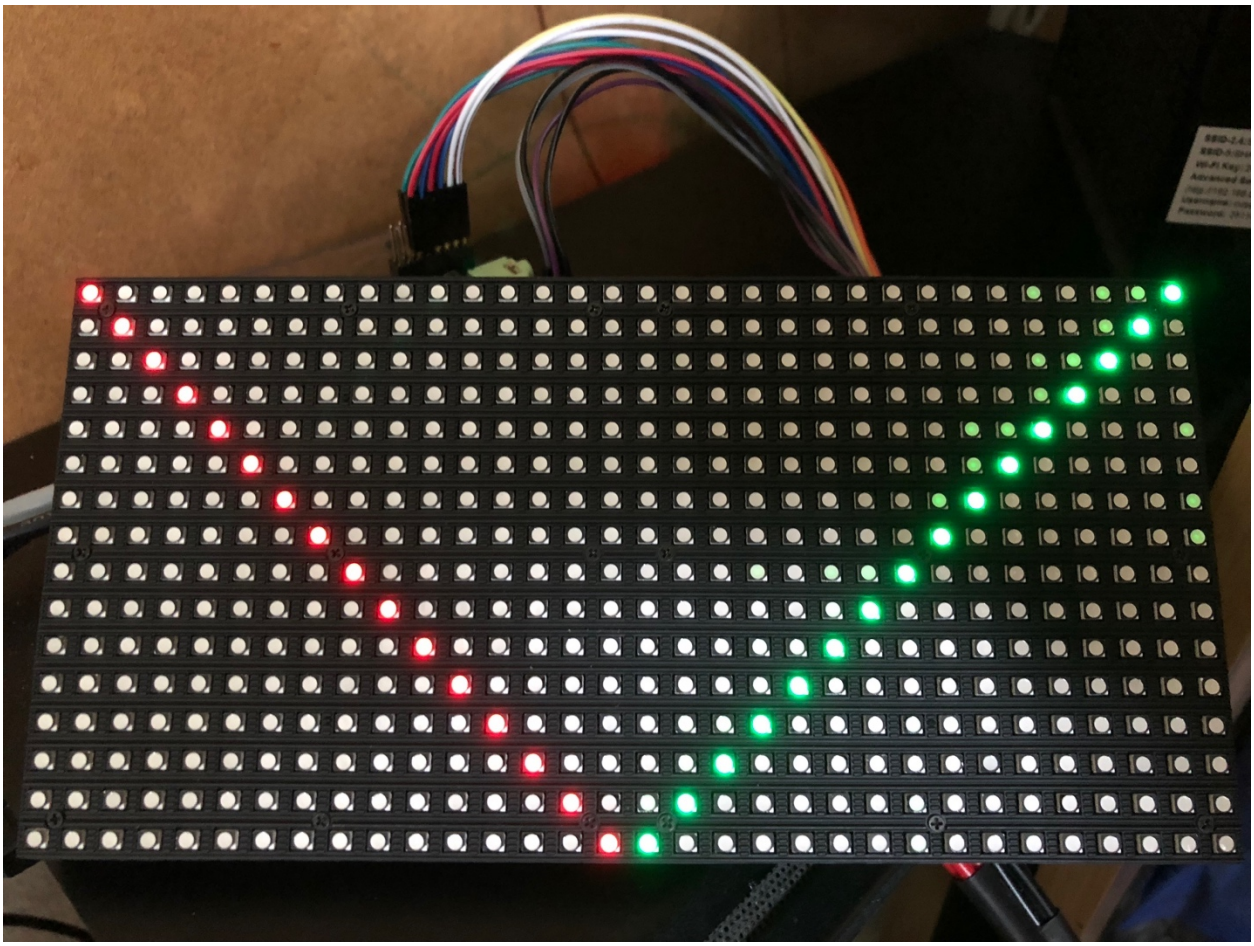


Figure 11. Expected output of LED display after executing the sample program on the BeagleBone Green.

Troubleshooting

1. If the display is flickering a lot, there are a few ways to mitigate the effects such as refreshing the display at a faster rate because the faster the LEDs are flashing it will be harder for our eyes to pick up the flickering.

    1.1. If increasing the refresh rate does not resolve issue, then the next best step would be to look into the PRU. A guide is available here:

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2016-student-howtos/PRU_GPIO_guide.pdf

1.2. Note that as you increase the number of threads in your program, no matter how fast you set the refresh rate programmatically it will not have an effect because at this point the bottleneck is the hardware being overloaded with work.

2. If you see random LEDs light up or discrepancies in color output, it is likely that the wiring is not properly fitted into the slots. Please double check all points of wiring and ensure everything is properly plugged in and nothing is loose.

3. If some parts of the display are not turning on or none at all, ensure the power supply on all ends is fitted properly and that you did not mix up any of the jumper wires with the wrong GPIO. In an unlikely event you might have something else already using the pins that the display uses or worst-case scenario the display is malfunctional. Also make sure the program is executed on the BeagleBone Green, not the host machine.

Useful Miscellaneous Links

1. https://learn.adafruit.com/32x16-32x32-rgb-led-matrix/overview
2. https://learn.sparkfun.com/tutorials/rgb-panel-hookup-guide