DESCRIPTION

The IS31FL3747 is a general purpose 12 × n (n=1~12) LED Matrix programmed via 12MHz SPI or 30MHz VSB compatible interface. Each LED can be dimmed individually with 8-bit PWM or 12-bit PWM data and 8-bit DC scaling data which allowing 256 steps or 4096 steps of linear PWM dimming and 256 steps of DC current adjustable level.

Additionally, each LED open and short state can be detected, IS31FL3747 store the open or short information in Open-Short Registers. The Open-Short Registers allowing MCU to read out via VSB or SPI interface, inform MCU whether there are LEDs open or short and the locations of open or short LEDs.

The IS31FL3747 operates from 2.7V to 5.5V and features a very low shutdown and operational current.

IS31FL3747 is available in QFN-40 (5mm×5mm) package. It operates from 2.7V to 5.5V over the temperature range of -40°C to +125°C.

FEATURES

- Supply voltage range: 2.7V to 5.5V
- Support 12 × n (n=1~12) LED matrix configurations
- Dual IS31FL3747 cascade connection support 24 × n (n=1~24) LED matrix configurations.
- Individual 12-bit, 8-bit and 8+4-bit PWM control steps
- Individual 8-bit DC current steps
- Global 8-bit current setting
- Maximum 32MHz with spread spectrum (SSP)
- 12MHz SPI or 30MHz VSB
- State lookup registers
- Individual open and short detect function
- 180-degree phase delay operation to reduce power noise
- De-Ghost
- ±7.5% (Max.) at 47.8mA channel to channel matching
- ±4% (Max.) at 47.8mA device to device accuracy
- QFN-40 (5mm×5mm×0.75mm)

QUICK START

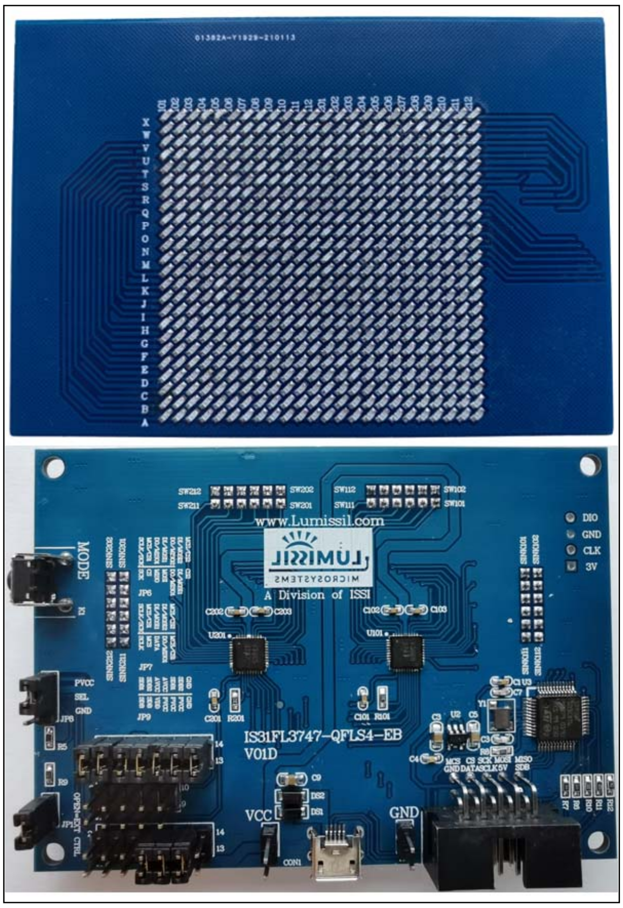

Figure 1: Photo of IS31FL3747 Evaluation Board

RECOMMENDED EQUIPMENT

- 5.0V, 2A power supply

ABSOLUTE MAXIMUM RATINGS

- ≤ 5.5V power supply

Caution: Do not exceed the conditions listed above, otherwise the board will be damaged.

ORDERING INFORMATION

Part No.	Temperature Range	Package
IS31FL3747-QFLS4-EB	-40°C to +125°C (Industrial)	QFN-40, Lead-free

Table 1: Ordering Information

For pricing, delivery, and ordering information, please contacts Lumissil's analog marketing team at analog@Lumissil.com or (408) 969-6600.

PROCEDURE

The IS31FL3747 evaluation board is fully assembled and tested. Follow the steps listed below to verify board operation.

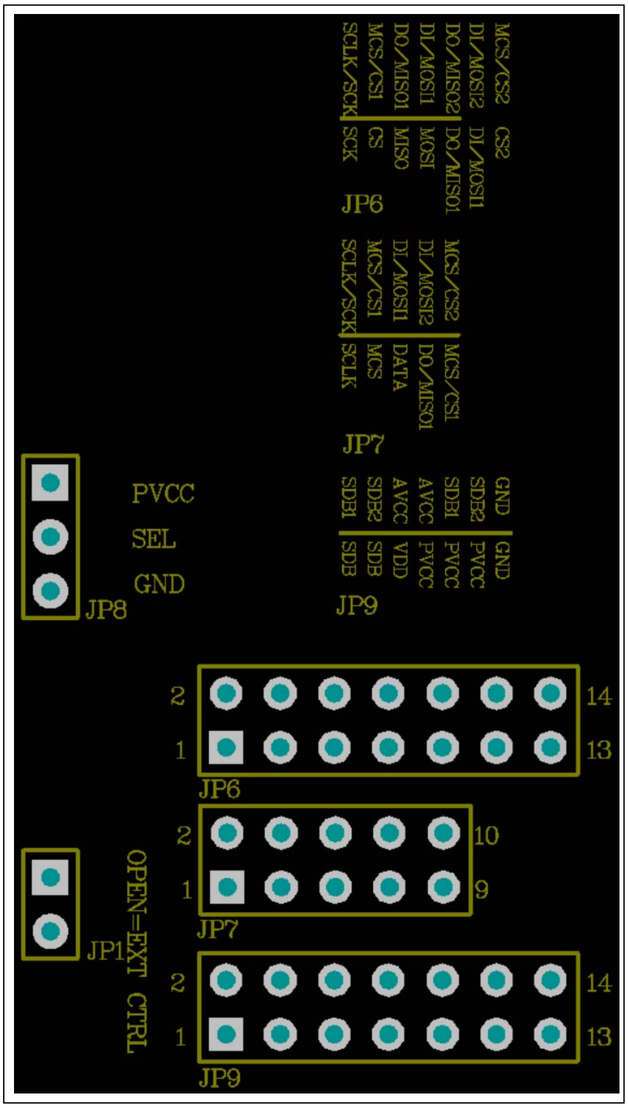

Figure 2: JP1/JP8/JP6/JP7/JP9 Pin Cconfiguration

Caution: Do not turn on the power supply until all connections are completed.

1) Short JP1 (Open=Ext control).
2) Short SLE to PVCC in JP8(select SPI interface).
3) In JP6, short PIN 1 to 2, 3 to 4, 5 to 6, 7 to 8, 9 to 10, 11 to 12. (Connect Chip0&Chip1 SPI to MCU).
4) In JP9, short PIN 7 to 8, 9 to 10, 11 to 12 (Connect Chip0&Chip1 AVCC/SDB to PVCC).
5) Turn on the power supply/Plug in the Micro USB Pay attention to the supply current. If the current exceeds 1.5A, please check for circuit fault.

EVALUATION BOARD OPERATION

The IS31FL3747 evaluation board has three animation displays modes. Press K1 to switch configurations.

1) Blue Mode 24×24 (Half Brightness)
2) Blue Mode 12×24 (Full Brightness)
3) Water lamp Mode
4) Countdown Mode
5) Dolphin animation Mode

Note: IS31FL3747 solely controls the FxLED function on the evaluation board.

SOFTWARE SUPPORT

JP8 default setting is closed (jumper on). If it is open (no jumper), the on-board MCU will configure its own SPI and SDB pins to High Impedance status so an external source can driver the SPI/SDB signals to control the IS31FL3747 LED driver.

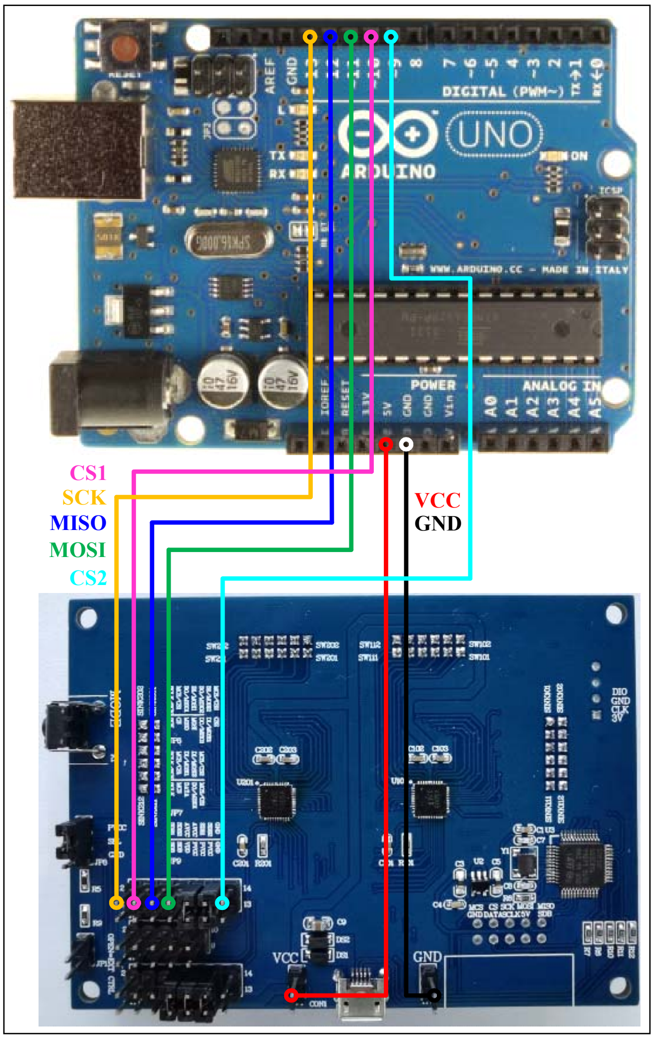

Figure 3: Photo of Arduino UNO connected to Evaluation Board

The steps listed below are an example using the Arduino for external control.

The Arduino hardware consists of an Atmel microcontroller with a bootloader allowing quick firmware updates. First download the latest Arduino Integrated Development Environment IDE (1.6.12 or greater) from www.arduino.cc/en/Main/Software. Also download the Wire.h library from www.arduino.cc/en/reference/wire and verify that pgmspace.h is in the directory …program Files(x86)/Arduino/hardware/tools/avr/avr/include/avr/. Then download the latest IS31FL3747 test firmware (sketch) from the Lumissil website http://www.lumissil.com/products/led-driver/fxled.

1) Perform the following steps on the IS31FL3747 evaluation board
   a) Short JP1 (Open=Ext control).
   b) Short SLE to PVCC in JP8(select SPI interface).
   c) In JP9, short PIN 7 to 8, 9 to 10, 11 to 12 (Connect Chip0&Chip1 AVCC/SDB to PVCC).
   d) In JP6, short PIN 9 to 10, 11 to 12 (Connect Chip0&Chip1 MISO and MOSI).
2) Connect the 7 pins from Arduino board to IS31FL3747 EVB:
   a) Arduino 5V pin to IS31FL3747 EVB VCC.
   b) Arduino GND to IS31FL3747 EVB GND.
   c) Arduino SCK (13) to IS31FL3747 EVB SCLK/SCK.
   d) Arduino MISO (12) to IS31FL3747 EVB DO/MISO1.
   e) Arduino MOSI (11) to IS31FL3747 EVB DI/MOSI1.
   f) Arduino Pin 10 to IS31FL3747 EVB MCS/CS1.
   g) Arduino Pin 9 to IS31FL3747 EVB MCS/CS2.
3) Use the test code in appendix Ⅰ or download the test firmware (sketch) from the Lumissil website, a .txt file and copy the code to Arduino IDE, compile and upload to Arduino.

Please refer to the datasheet to get more information about IS31FL3747.

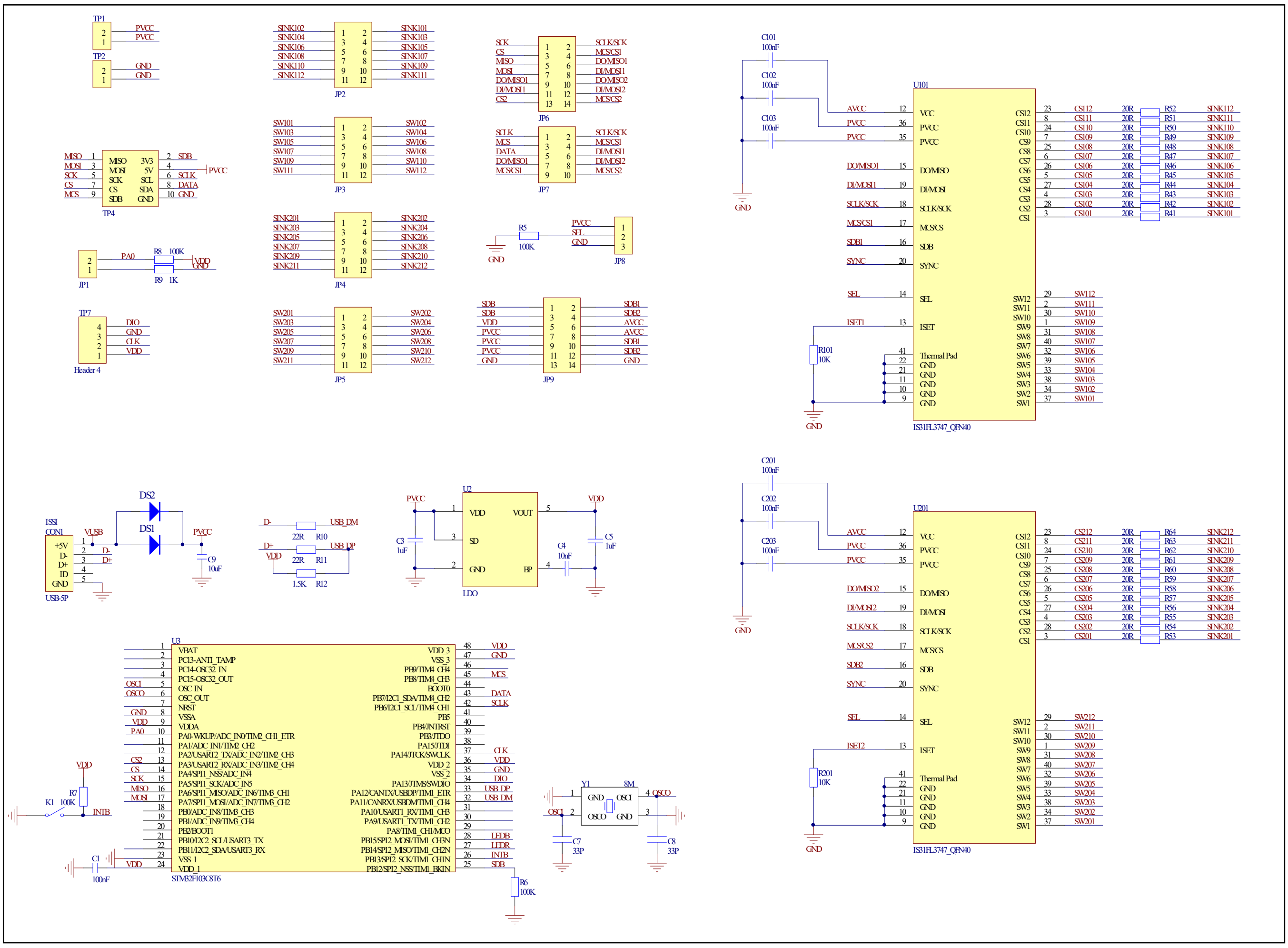

Figure 4: IS31FL3747 Application Schematic

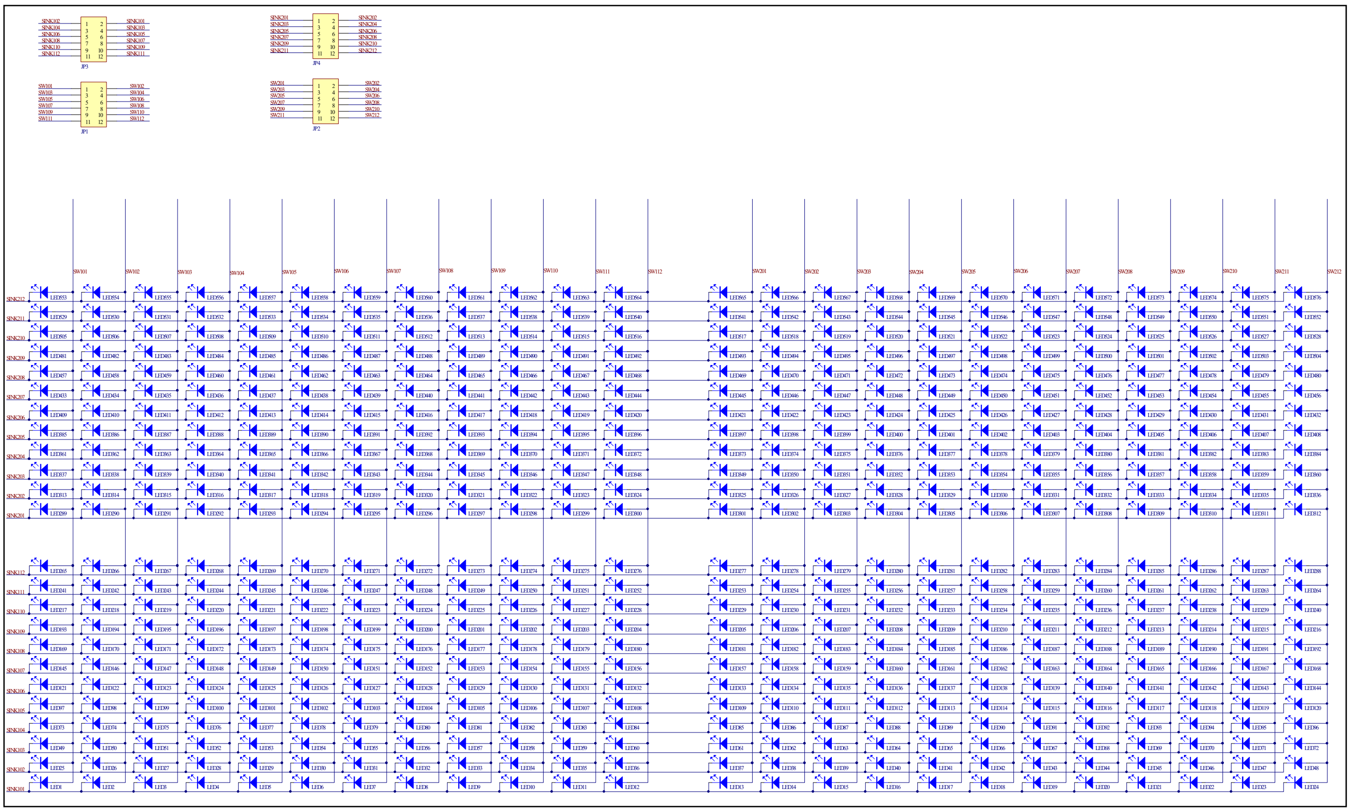

Figure 5: FxLED 24×24 ARRAY Application Schematic

BILL OF MATERIALS

IS31FL3747

Name	Symbol	Description	Qty	Supplier	Part No.
LDO	U2	Linear Regulator	1	SGMICRO	SGM2019-3.3V
MCU	U3	Microcontroller	1	STM	STM32F103C8T6
LED Driver	U101,U201	Matrix LED Driver	2	Lumissil	IS31FL3747
Crystal	Y1	Crystal, 8MHz	1	JB	HC-49S
Diode	DS1,DS2	Diode, SMD	2	DIODES	DFLS240
Resistor	R5,R6,R7,R8	RES,100k,1/10W,±5%,SMD	4	Yageo	RC0603JR-07100KL
Resistor	R9	RES,1k,1/10W,±5%,SMD	1	Yageo	RC0603JR-071KL
Resistor	R10,R11	RES,22R,1/10W,±5%,SMD	2	Yageo	RC0603JR-0722RL
Resistor	R12	RES,1.5k,1/10W,±5%,SMD	1	Yageo	RC0603JR-071K5L
Resistor	R101,R201	RES,10k,1/10W,±5%,SMD	2	Yageo	RC0603JR-0710KL
Resistor	R41,R44,R47,R50, R53,R56,R59,R62	RES,10R,1/10W,±5%,SMD Or RES,20R,1/10W,±5%,SMD (Note 1)	8	Yageo	RC0603JR-0710RL Or RC0603JR-0720RL
Resistor	R42,R43,R45,R46, R48,R49,R51,R52, R54,R55,R57,R58, R60,R61,R63,R64	RES,10R,1/10W,±5%,SMD Or RES,20R,1/10W,±5%,SMD	16	Yageo	RC0603JR-0710RL Or RC0603JR-0720RL
Capacitor	C1,C101,C102, C103,C201,C202, C203	CAP,0.1μF,16V,±20%,SMD	7	Yageo	CC0603MRX7R7BB104
Capacitor	C3,C5	CAP,1μF,16V,±20%,SMD	2	Yageo	CC0603KRX7R7BB105
Capacitor	C7,C8	CAP,33pF,16V,±20%,SMD	2	Yageo	CQ0603JRNPO9BN360
Capacitor	C4	CAP,10nF,16V,±20%,SMD	1	Yageo	CC0603KRX7R7BB103
Capacitor	C9	CAP,10μF,16V, ±20%,SMD	2	Yageo	CC0805MKX5R7BB106
Button	K1	Button SMD	1		

Bill of Materials, refer to Figure 4 above.

FxLED 24×24 ARRAY

Name	Symbol	Description	Qty	Supplier	Part No.
Diode	LED1~ LED576	LED, SMD BLUE	576	EVERLIGHT	19-217/BHC-ZL1M2RY/3T

Bill of Materials, refer to Figure 5 above.

Note 1: The value of these resistors on the evaluation board is 10Ω or 20 Ω. For PVCC=5V and red LED application, prefer 39Ω for these resistors.

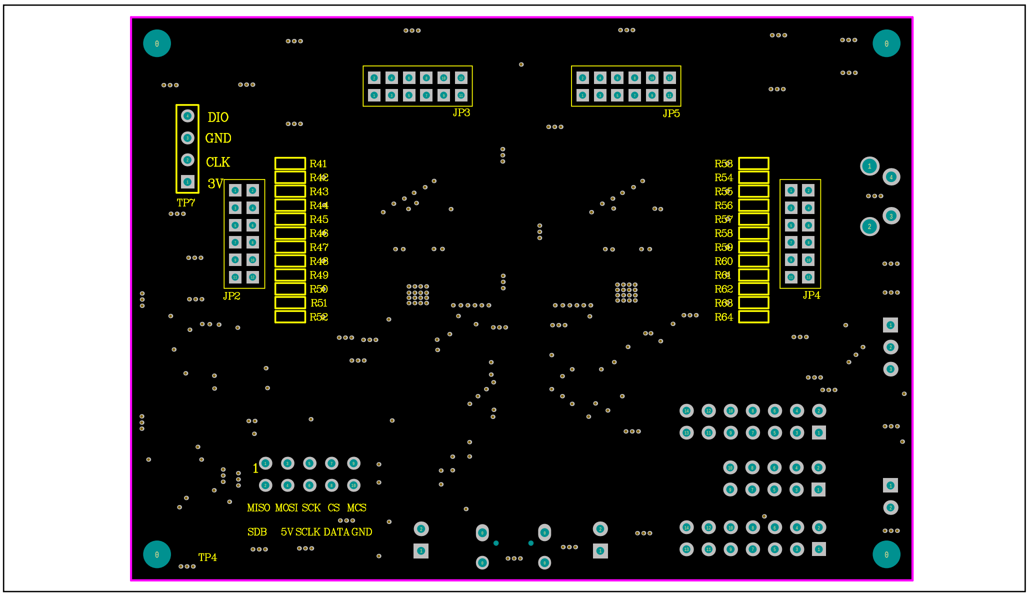

Figure 6: Board Component Placement Guide - Top Layer

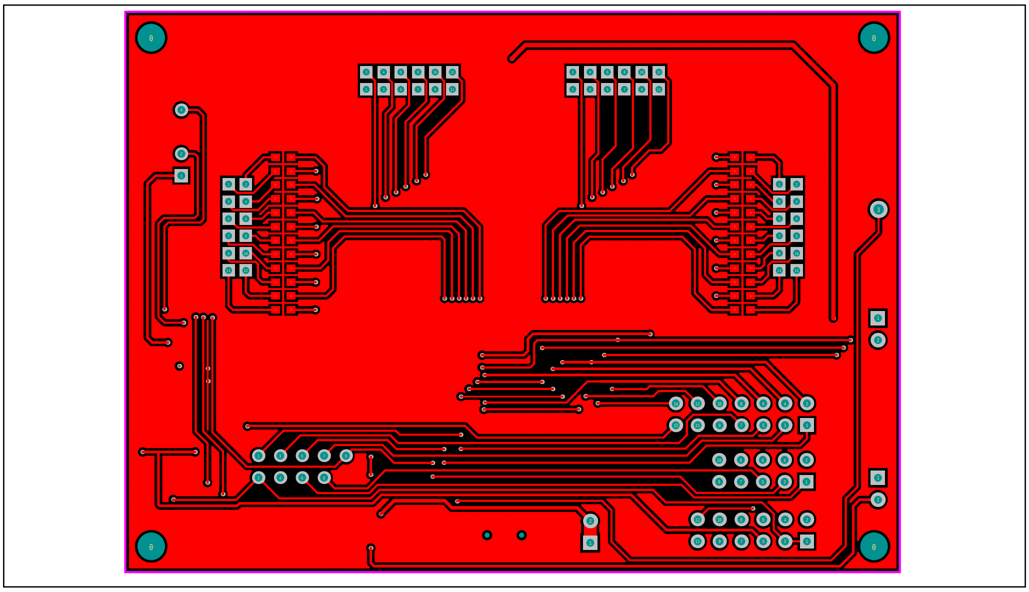

Figure 7: Board PCB Layout - Top Layer

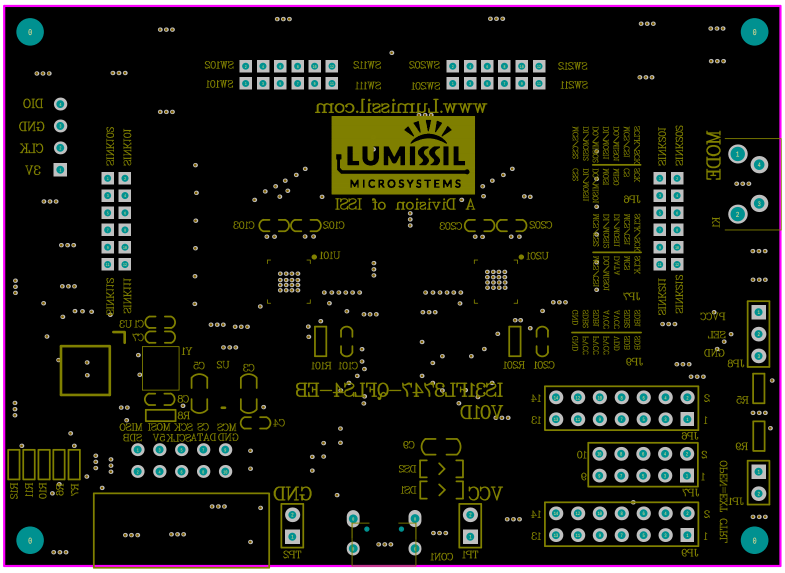

Figure 8: Board Component Placement Guide - Bottom Layer

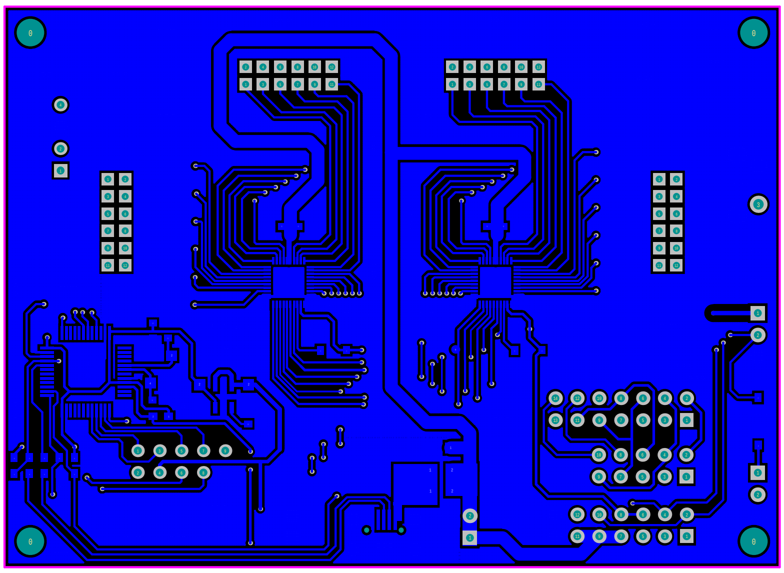

Figure 9: Board PCB Layout - Bottom Layer

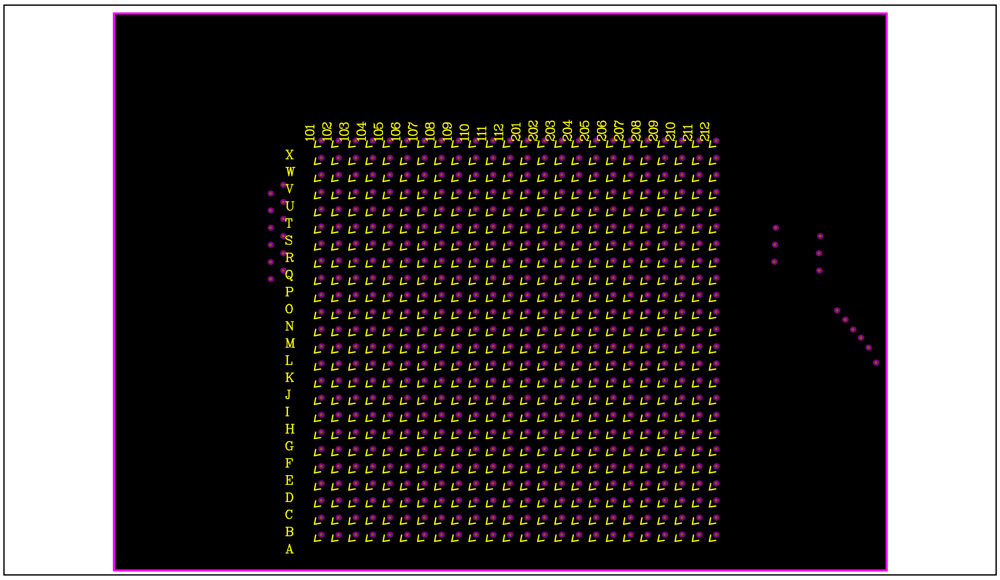

Figure 10: LED Board Component Placement Guide - Top Layer

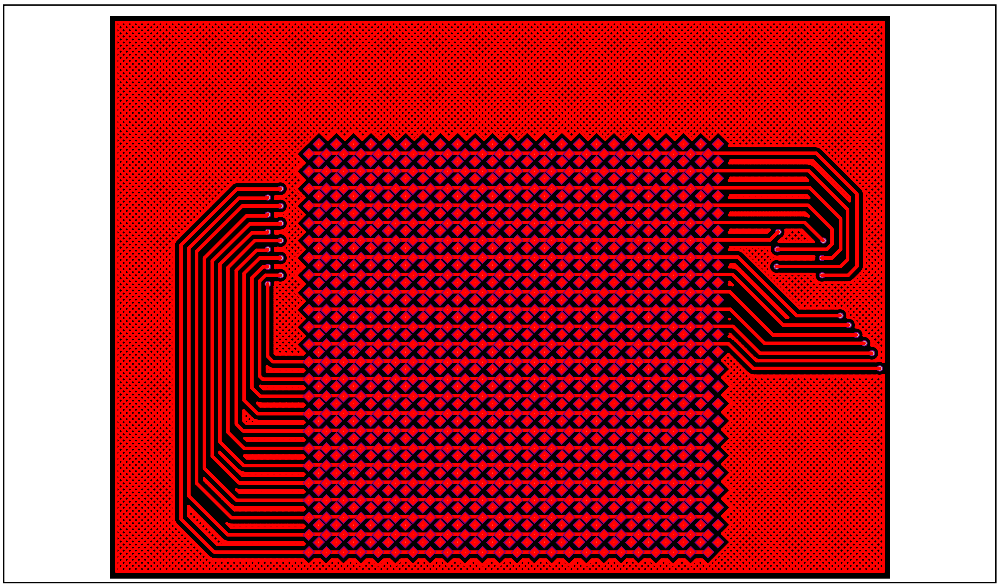

Figure 11: LED Board PCB Layout - Top Layer

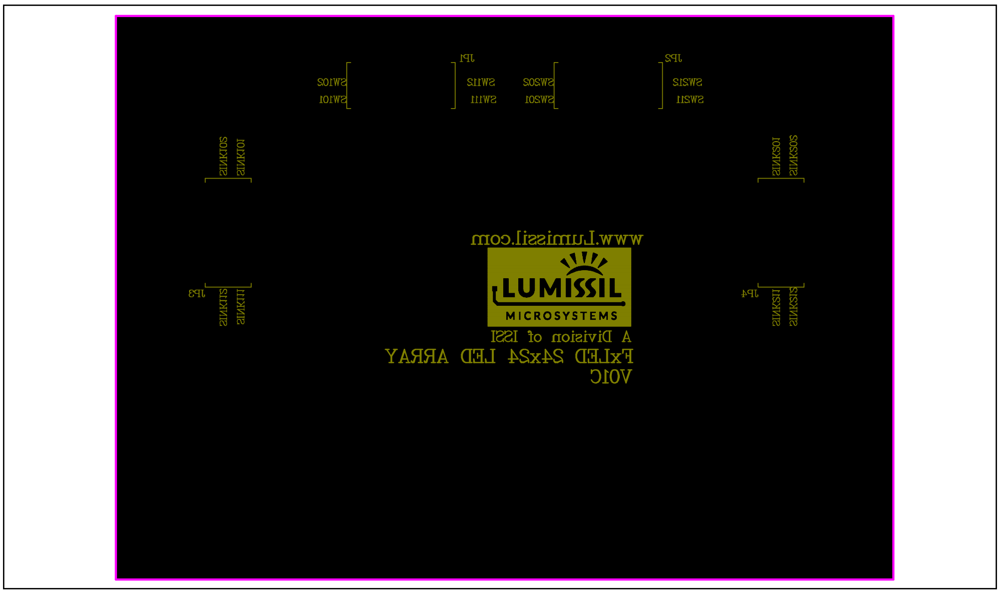

Figure 12: LED Board Component Placement Guide - Bottom Layer

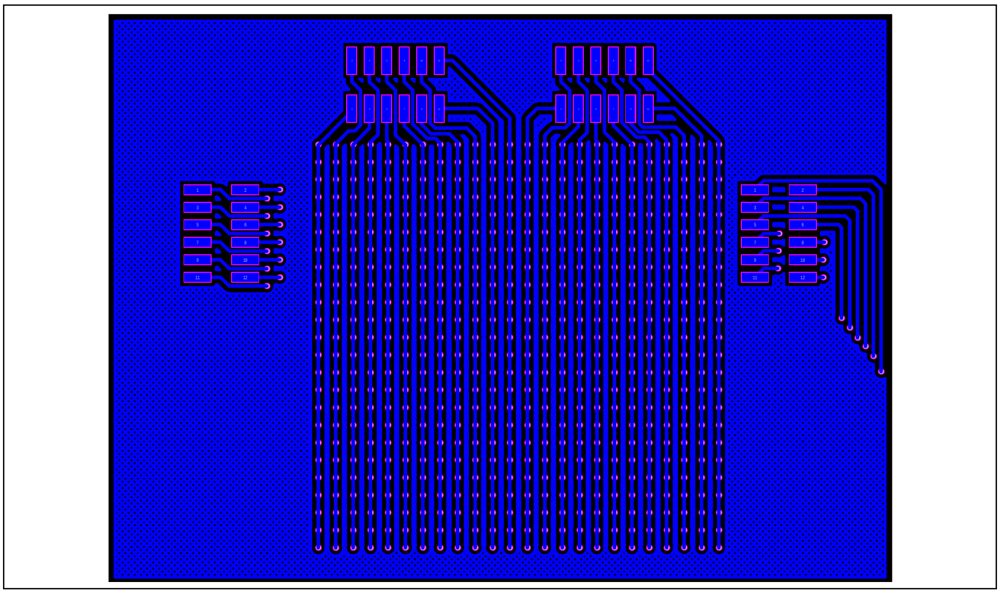

Figure 13: LED Board PCB Layout - Bottom Layer

REVISION HISTORY

Revision	Detail Information	Date
A	Initial release	2021.10.18
B	Update PCB version (PCB modify logo)	2021.12.17

APPENDIX Ⅰ: IS31FL3747 Arduino Test Code V01A

```
#include<SPI.h>
#include<avr/pgmspace.h>

const int slaveSelectPin_0 = 10;
const int slaveSelectPin_1 = 9;

void setup()
{
   int i;
   // put your setup code here, to run once:
   // set the slaveSelectPin as an output:
   pinMode (slaveSelectPin_0, OUTPUT);
   pinMode (slaveSelectPin_1, OUTPUT);
   // initialize SPI:
   SPI.begin();
   SPI.beginTransaction(SPISettings(20000000, MSBFIRST, SPI_MODE0));
   //SPI.setClockDivider(SPI_CLOCK_DIV4);
   SPI.setDataMode(3);
   Init3747();
}
void loop()
{
   // put your main code here, to run repeatedly:
   mainloop();
}

void SPI_WriteByte(uint8_t Reg_Page,uint8_t Reg_Add,uint8_t Reg_Dat)    //write register to chip0&chip1 at the same time
{
   digitalWrite(slaveSelectPin_0, LOW);     // take the CS pin low to select the chip 0:
   digitalWrite(slaveSelectPin_1, LOW);     // take the CS pin low to select the chip 1:
   SPI.transfer(Reg_Page);
   SPI.transfer(Reg_Add);
   SPI.transfer(Reg_Dat);
   digitalWrite(slaveSelectPin_0, HIGH);     // take the CS pin high to de-select the chip 0:
   digitalWrite(slaveSelectPin_1, HIGH);     // take the CS pin high to de-select the chip 1:
}

void SPI_WriteByte_0(uint8_t Reg_Page,uint8_t Reg_Add,uint8_t Reg_Dat)    //write register to chip 0
{
   digitalWrite(slaveSelectPin_0, LOW);     // take the CS pin low to select the chip 0:
   SPI.transfer(Reg_Page);
   SPI.transfer(Reg_Add);
   SPI.transfer(Reg_Dat);
   digitalWrite(slaveSelectPin_0, HIGH);     // take the CS pin high to de-select the chip 0:
}

void SPI_WriteByte_1(uint8_t Reg_Page,uint8_t Reg_Add,uint8_t Reg_Dat)    //write register to chip 1
{
   digitalWrite(slaveSelectPin_1, LOW);     // take the CS pin low to select the chip 0:
   SPI.transfer(Reg_Page);
   SPI.transfer(Reg_Add);
   SPI.transfer(Reg_Dat);
   digitalWrite(slaveSelectPin_1, HIGH);     // take the CS pin high to de-select the chip 0:
}

void Init3747(void)
{
   int i,j;
/******************************** chip 0 Initialization***********************************/
   SPI_WriteByte_0(0x55,0x00,0x41);   // Normal operation, Cascading Chip 0
   SPI_WriteByte_0(0x55,0x01,0xe0);   // Master device,SW1~SW24
   SPI_WriteByte_0(0x55,0x02,0x40);   // GCC
   SPI_WriteByte_0(0x55,0x05,0xaa);   // Deghost:SW1~SW4 Pull Down
   SPI_WriteByte_0(0x55,0x06,0xaa);   // Deghost:SW5~SW8 Pull Down
   SPI_WriteByte_0(0x55,0x07,0xaa);   // Deghost:SW9~SW12 Pull Down
   SPI_WriteByte_0(0x55,0x08,0xaa);   // Deghost:CSy Pull Up

   for(i = 0x01; i <= 0x90; i++)
   {
      SPI_WriteByte_0(0x53,i,0xff); //Page3:scaling
   }
   for(i = 0x01; i <= 0x90; i++)
```

```
    {
        SPI_WriteByte_0(0x54,i,0xff); //Page4:scaling
    }

    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x50,i,0x00);   //Page0:PWM_L
        SPI_WriteByte_0(0x50,i+1,0x00);//Page0:PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x51,i,0x00);   //Page1:PWM_L
        SPI_WriteByte_0(0x51,i+1,0x00);//Page1:PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x52,i,0x00);   //Page2:PWM_L
        SPI_WriteByte_0(0x52,i+1,0x00);//Page2:PWM_H
    }
/******************************** chip 1 Initialization***********************************/
    SPI_WriteByte_1(0x55,0x00,0xc1);   // Normal operation,Cascading Chip 1
    SPI_WriteByte_1(0x55,0x01,0xa0);   // Slave device,SW1~SW24
    SPI_WriteByte_1(0x55,0x02,0x40);   // GCC
    SPI_WriteByte_1(0x55,0x05,0xaa);   // Deghost:SW1~SW4 Pull Down
    SPI_WriteByte_1(0x55,0x06,0xaa);   // Deghost:SW5~SW8 Pull Down
    SPI_WriteByte_1(0x55,0x07,0xaa);   // Deghost:SW9~SW12 Pull Down
    SPI_WriteByte_1(0x55,0x08,0xaa);   // Deghost:CSy Pull Up

    for(i = 0x01; i <= 0x90; i++)
    {
       SPI_WriteByte_1(0x53,i,0xff); //Page3:scaling
    }
    for(i = 0x01; i <= 0x90; i++)
    {
        SPI_WriteByte_1(0x54,i,0xff); //Page4:scaling
    }

    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_1(0x50,i,0x00);   //Page0:PWM_L
        SPI_WriteByte_1(0x50,i+1,0x00);//Page0:PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_1(0x51,i,0x00);   //Page1:PWM_L
        SPI_WriteByte_1(0x51,i+1,0x00);//Page1:PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_1(0x52,i,0x00);   //Page2:PWM_L
        SPI_WriteByte_1(0x52,i+1,0x00);//Page2:PWM_H
    }
    SPI_WriteByte(0x55,0xC8,0x00);     //Update Chip0 & Chip1
}
void mainloop(void)//
{
    int i;
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x50,i,0xff);     //Chip0:Page0 PWM_L
        SPI_WriteByte_0(0x50,i+1,0x0f);//Chip0:Page0 PWM_H
        SPI_WriteByte_1(0x50,i,0xff);     //Chip1:Page0 PWM_L
        SPI_WriteByte_1(0x50,i+1,0x0f);//Chip1:Page0 PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x51,i,0xff);     //Chip0:Page1 PWM_L
        SPI_WriteByte_0(0x51,i+1,0x0f);//Chip0:Page1 PWM_H
        SPI_WriteByte_1(0x51,i,0xff);     //Chip1:Page1 PWM_L
        SPI_WriteByte_1(0x51,i+1,0x0f);//Chip1:Page1 PWM_H
    }
    for(i = 0x01; i <= 0xc0; i=i+2)
    {
        SPI_WriteByte_0(0x52,i,0xff);     //Chip0:Page2 PWM_L
        SPI_WriteByte_0(0x52,i+1,0x0f);//Chip0:Page2 PWM_H
```

```
        SPI_WriteByte_1(0x52,i,0xff);      //Chip1:Page2 PWM_L
        SPI_WriteByte_1(0x52,i+1,0x0f);//Chip1:Page2 PWM_H
    }
    SPI_WriteByte(0x55,0xC8,0x00);    //Update Chip0 & Chip1
    delay(50);
}
```