48-CHANNEL, 16-BIT PWM LED DRIVER

DESCRIPTION

The IS32FL3248 is a 48-channel constant current LED driver. Each channel has 16-bit PWM brightness control, and 64 steps of constant-current scaling (SL). SL can adjust brightness deviation between channels. GCC can adjusts brightness deviation between the R, G, and B color groups. The 8 steps maximum current band control (CB) selects the maximum output current range for all channels.

Proprietary programmable algorithms are used in IS32FL3248 to minimize audible noise that can result from MLCC decoupling capacitor. SL, GCC, CB and all other registers can be programmed via a high-speed VSB (video series bus, up to 33MHz) or SPI (up to 33MHz) serial interface port.

IS32FL3248 software shutdown mode can put the device to sleep (for minimum power consumption) while retaining all register values.

IS32FL3248 is available in WFQFN-64 (9mm×9mm) package. It operates from 3.0V to 5.5V over the temperature range of -40°C to +125°C.

QUICK START

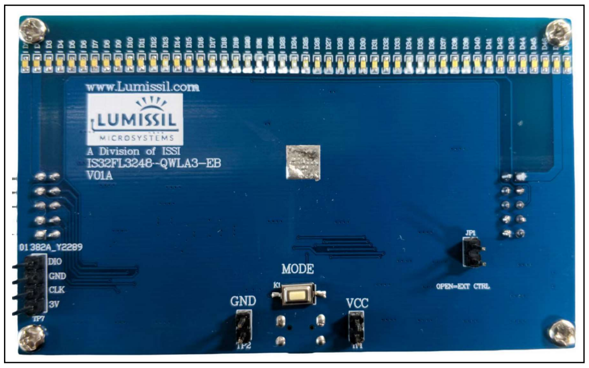

Figure 1: Photo of IS32FL3248 Evaluation Board

FEATURES

- V_{CC}= 3.0V to 5.5V
- Tolerate up to 18V, nominal operation voltage between 4.5V to 16V, multiple LED's can be connected in series
- Support 48 current sink output channels
- 33mA@V_{CC}= 5V/channel maximum
- 23mA@V_{CC}= 3V/channel maximum
- Maximum Current Band (CB)
  - 3-bit (8 steps) with a 9.7% to 100% range
- DC Current Scaling (SL)
- 6-bit (64 steps) with a 25.9% to 100% range
- Individual 16-bit, 8+8-bit dithering, 8+4-bit dithering, 8-bit PWM mode
- Global Current Control (GCC)
- 8-bit (256 steps) with a 9.6% to 100% range
- 3 GCC sets for each color group
- Constant Current Accuracy
- Channel to Channel = ±2%(Typ.), ±5%(Max.)
- Device to Device = ±2%(Typ.), ±5%(Max.)
- Chain topology via VSB interface, PWM data I/O is daisy chained with bi-directional data transmission (write and read)
- Support bi-directional data output via DI
- Grayscale Control Clock: 33MHz
- Display timing reset
- Real-time LED open detection (LOD)
- Real-time LED short detection (LSD)
- Over temperature protection
- Spread Spectrums
- Software shutdown mode
- 180-degree phase delay operation to reduce power noise
- AEC-Q100 qualification in progress
- Operation temperature range: -40°C to +125°C
- WFQFN-64 (9mm×9mm) package
- RoHS & Halogen-Free Compliance
- TSCA Compliance

RECOMMENDED EQUIPMENT

- 5V, ≥1.5A power supply

ABSOLUTE MAXIMUM RATINGS

- VIN+, ≤7V power supply

Caution: EVB is designed for 5V application, higher than 7V will cause extra-heat on the IC and if VIN (TP1) is higher than 7V, the IS32FL3248 will be too hot and enter thermal shutdown mode, if VIN (TP1) exceeds the conditions listed above, the board may be damaged.

PROCEDURE

The IS32FL3248 evaluation board is fully assembled and tested. Follow the steps listed below to verify board operation.

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

1) Connect the ground terminal of the power supply to the GND and the positive terminal to the TP1 (VCC).
2) Connect JP1 (Internal Control) to enable the control of board MCU (default status).
3) VSB MODE:
   Connect 'DO" to '3' pin in Left pin header.
   Connect 'DI' to 'SIN' pin in Right pin header.
   Connect 'CSK/SCLK' to 'SCK' pin in Right pin header.
   Connect 'CS/LAT' to 'LAT' pin in Right pin header.
   Connect 'GND' to 'GND' pin in Right pin header.
4) SPI MODE:
   Connect 'MISO" to '3' pin in Left pin header.
   Connect 'MOSI' to 'SIN' pin in Right pin header.
   Connect 'CSK/SCLK' to 'SCK' pin in Right pin header.
   Connect 'CS/LAT' to 'LAT' pin in Right pin header.
   Connect 'GND' to 'GND' pin in Right pin header.
5) Turn on the power supply and pay attention to the supply current. If the current exceeds 1.5A, please check for circuit fault.
6) Enter the desired mode of display by toggling the MODE button (K1).

CASCADE

The IS32FL3248 can be cascaded, it through headers implement. If you want to use cascadinplease according to procedure to follow the steps, and the cascade PCB-board need to set to 'ExtCTRL' (External Control),is to Pull the cascade PCB-board(Number 2) 'JP1' Jumper caps.

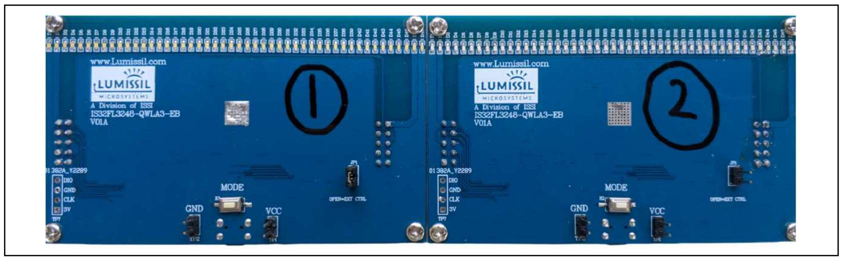

Figure 2: Photo of IS32FL3248 Evaluation Board cascade

ORDERING INFORMATION

Part No.	Temperature Range	Package
IS32FL3248-QWLA3-EB	-40°C to +125°C (Automotive)	WFQFN-64, 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.

EVALUATION BOARD OPERATION

The IS32FL3248 evaluation board has 3 display modes. Press MODE button (K1) to switch configurations.

1) (Default mode) Running Water Mode
2) Marquee Mode
3) Breath Mode

Note: IS32FL3248 solely controls the FxLED function on the evaluation board.

SOFTWARE SUPPORT

JP1 default setting is 'IntCTRL' (Internal Control). If it is set to 'ExtCTRL' (External Control), the on-board MCU will configure all the IO pins to high impedance mode and enter sleep mode. The VSB pins and SDB pin are also set to High Impedance. External VSB and SDB signals can be connected to Right pin header to control the IS32FL3248 LED driver.

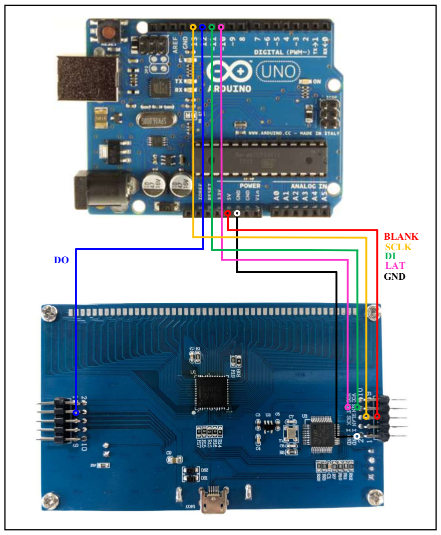

Figure 2: Arduino UNO Connected to Evaluation Board

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 IS32FL3248 test firmware (sketch) from the ISSI website http://www.lumissil.com/products/led-driver/fxled.

1) Open JP1 to set to 'ExtCTRL' (External Control).
2) Connect the ground terminal of the power supply to the GND and the positive terminal to the TP1 (VCC).
3) Connect 6 pins from Arduino board to IS32FL3248 EVB:
   a) Arduino GND to IS32FL3248 EVB GND.
   b) Arduino SDB (5V) to IS32FL3248 EVB SDB in 'BLAN' in Right pin header.
   c) Arduino SCK (13) to IS32FL3248 EVB 'SCK' in Right pin header.
   d) Arduino DO (12) to IS32FL3248 EVB 'SOUT' in Left pin header.
   e) Arduino DI (11) to IS32FL3248 EVB 'SIN' in Right pin header.
   f) Arduino LAT (10) to IS32FL3248 EVB 'LAT' in Right pin header.
   g) If Arduino use 3.3V MCU VCC, connect 3.3V to IS32FL3248 EVB SDB, if Arduino use 5.0V MCU VCC, connect 5.0V to EVB SDB. (Arduino UNO's VCC (VOH) is 5.0V, so SDB=5.0V)
4) Run the Arduino code for desired mode setting by Arduino code.
5) To set the SPI communication mode, follow the steps d) MISO replace DO, e) MOSI replace DI.

Please refer to the datasheet to get more information about IS32FL3248.

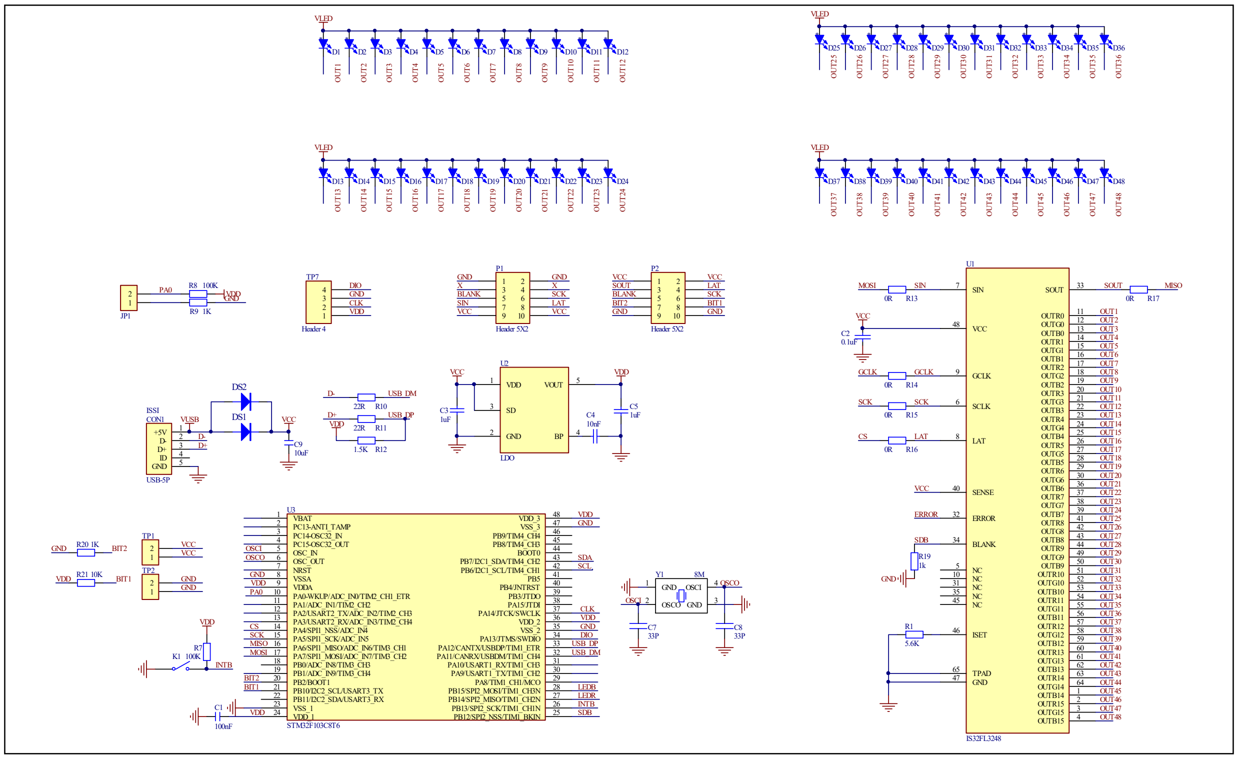

Figure 3: IS32FL3248 EVB Schematic

BILL OF MATERIALS

IS32FL3248

Name	Symbol	Description	Qty	Supplier	Part No.
LED Driver	U1	Matrix LED Driver	1	LUMISSIL	IS32FL3248-QWLA3
MCU	U2	Microcontroller	1	STM	STM32F103C8T6
LDO	U3	Low-dropout Regulator	1	ONSEMI	RS3236-3.3
Crystal	Y1	Crystal, 8MHz	1	JB	HC-49S
Diode	DS1,DS2	Diode, SMD	2	DIODES	DFLS240
Resistor	R1	RES,5.6k,1/10W,±5%,SMD	1	Yageo	RC0603JR-075K6L
Resistor	R7,R8	RES,100k,1/10W,±5%,SMD	2	Yageo	RC0603JR-07100KL
Resistor	R9,R19,R20	RES,1k,1/10W,±5%,SMD	3	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	R13,R14,R15, R16,R17,R18	RES,0R,1/10W,±5%,SMD	6	Yageo	RC0603JR-130RL
Resistor	R21	RES,10k,1/10W,±5%,SMD	1	Yageo	RC0603JR-0710KL
Capacitor	C1,C2	CAP,100nF,16V,±20%,SMD	2	Yageo	CC0603MPX7R7BB104
Capacitor	C4	CAP,10nF,25V,±10%,SMD	1	Yageo	CC0603KPX7R8BB103
Capacitor	C3,C5	CAP,1µF,16V,±10%,SMD	2	Yageo	CC0603KRX5R7BB105
Capacitor	C7,C8	CAP,33pF,50V,±5%,SMD	2	Yageo	CQ0603JRNPO9BN330
Capacitor	C9	CAP,10µF,50V,±10%,SMD	1	Yageo	CC0805KKX5R9BB106
Header	JP1,TP1,TP2	Header2.54,1X2p	3	HCTL	PZ254-1-02-Z-8.5
Header	P1	Header2.54,2X5p	1	HCTL	PZ254-2-05-W-8.5
Header	P2	Header2.54,2X5p,90°	1	HCTL	PM254-2-05-W-8.5
Header	TP7	Header2.54,1X4p	1	HCTL	PZ254-1-04-Z-8.5
LED	D1-D48	LED,SMD,white	48	Orient	ORH-W46G
Mirco USB	CON1	MircoUSB-5P,SMD	1	MOLEX	473461015
Button	K1	Button,SMD	1		

Bill of Materials, refer to Figure 3 above.

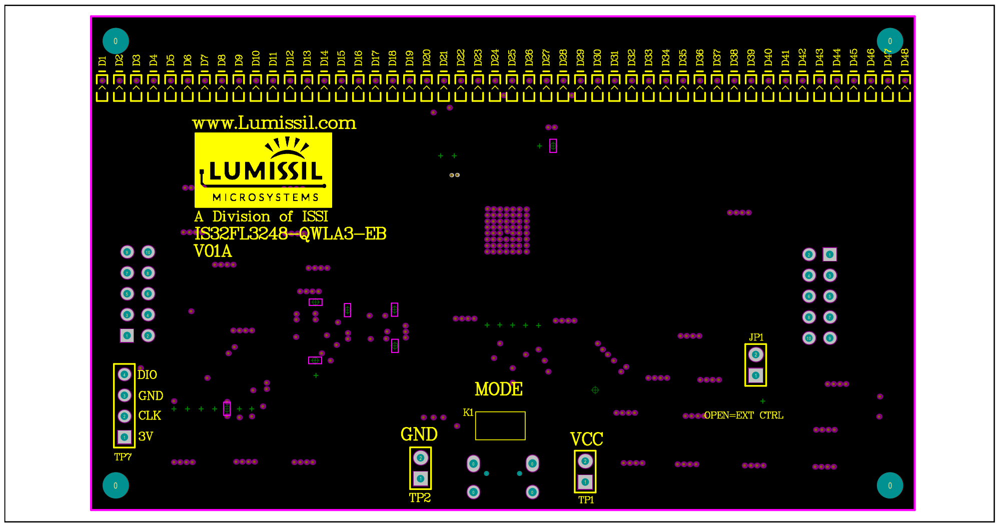

Figure 4: Board Component Placement Guide - Top Layer

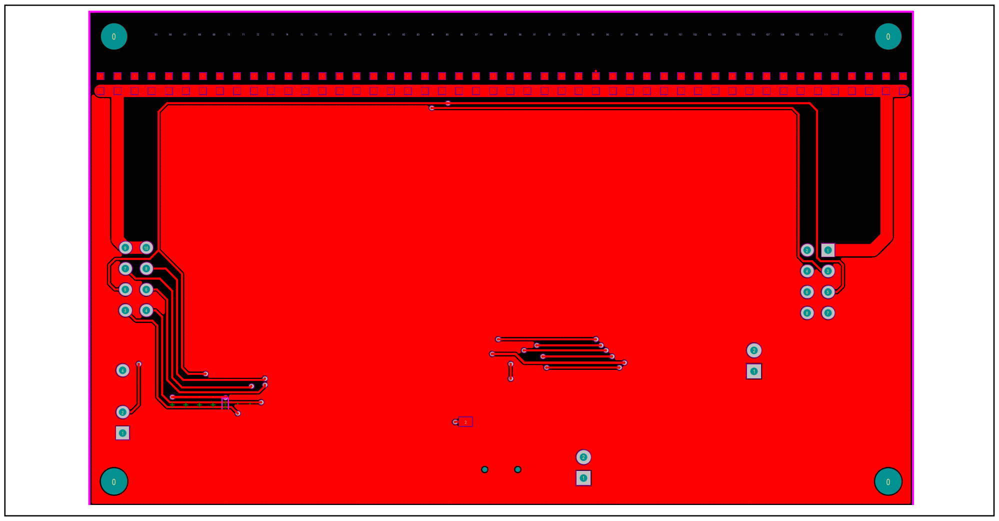

Figure 5: Board PCB Layout - Top Layer

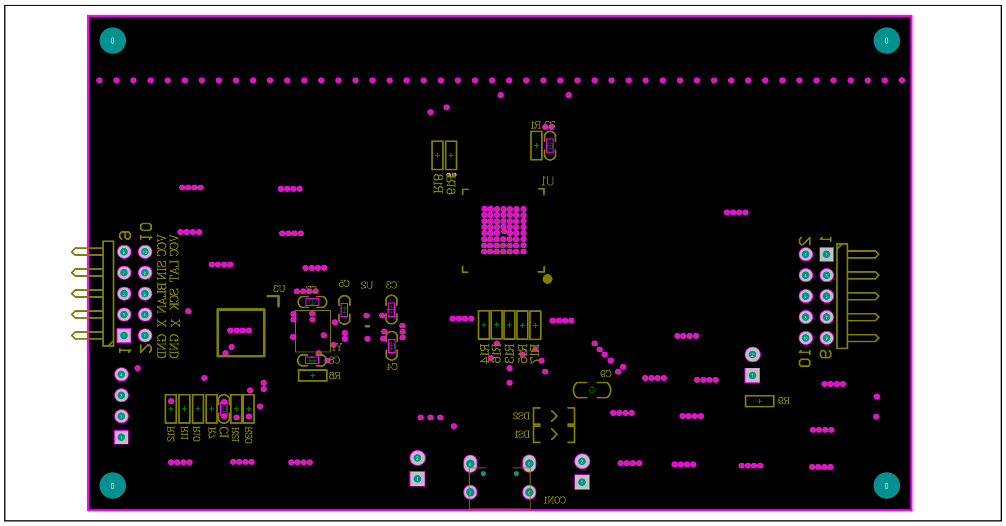

Figure 6: Board Component Placement Guide - Bottom Layer

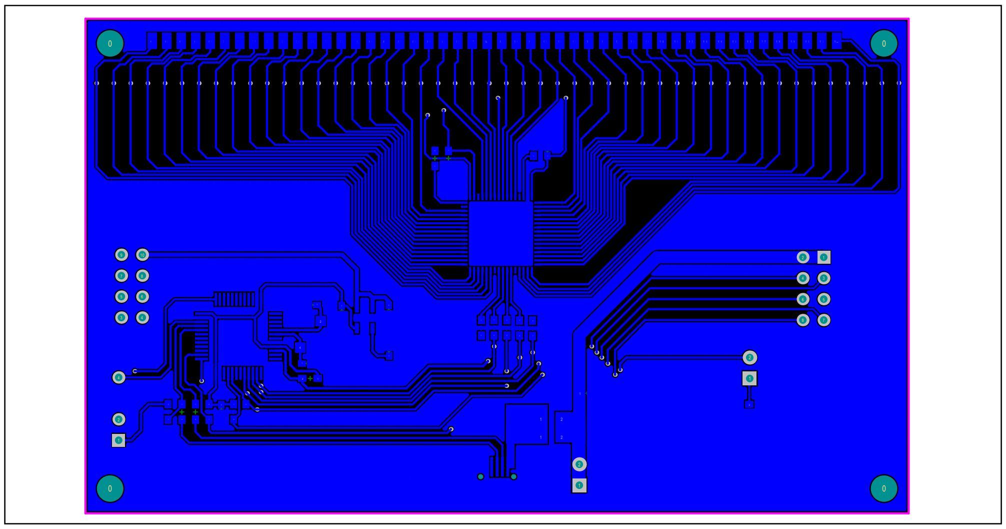

Figure 7: Board PCB Layout - Bottom Layer

REVISION HISTORY

Revision	Detail Information	Data
A	Initial Release	2022.12.19

APPENDIX Ⅰ: IS32FL3248 Arduino Test Code VSB V01A

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

uint8_t PWM_data[96];
uint8_t SL_data[48];
uint8_t FC1_data[6]={0x00,0x00,0x03,0xB1,0x00,0xBA};
uint8_t FC0_data[6]={0x00,0x00,0x0B,0xFF,0xFF,0xFF};
const int slaveSelectPin = 10;

byte PWM_Gamma64[64]=
{
  0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,
  0x08,0x09,0x0b,0x0d,0x0f,0x11,0x13,0x16,
  0x1a,0x1c,0x1d,0x1f,0x22,0x25,0x28,0x2e,
  0x34,0x38,0x3c,0x40,0x44,0x48,0x4b,0x4f,
  0x55,0x5a,0x5f,0x64,0x69,0x6d,0x72,0x77,
  0x7d,0x80,0x88,0x8d,0x94,0x9a,0xa0,0xa7,
  0xac,0xb0,0xb9,0xbf,0xc6,0xcb,0xcf,0xd6,
  0xe1,0xe9,0xed,0xf1,0xf6,0xfa,0xfe,0xff
};

void setup()
{
  // put your setup code here, to run once:
  // set the slaveSelectPin as an output:
  pinMode (slaveSelectPin, OUTPUT);
  // initialize SPI:
  SPI.begin();
  SPI.beginTransaction(SPISettings(20000000, MSBFIRST, SPI_MODE0));
  //SPI.setClockDivider(SPI_CLOCK_DIV4);
  SPI.setDataMode(3);
  pinMode(4,OUTPUT);//SDB
  digitalWrite(4,HIGH);//SDB_HIGH
  Init3248();
}

void loop()
{
  // put your main code here, to run repeatedly:
  IS32FL3248_mode1();
}

void VSB_WriteBuffer(uint8_t* pBuffer, int length_dat, uint8_t Dev_Add)
{
  digitalWrite(slaveSelectPin, LOW);    // take the SS pin low to select the chip:
  SPI.transfer(Dev_Add);    //   send in the address and value via SPI:
  while(length_dat--)
  {
    SPI.transfer(*pBuffer);
    pBuffer++;
  }
  digitalWrite(slaveSelectPin, HIGH);    // take the SS pin high to de-select the chip:
}

// SET the Scaling Register//
void SL_set(uint8_t byte1)
{
  uint8_t i;
  for(i=0;i<48;i+=2)
  {
    SL_data[i] = byte1;
  }
  VSB_WriteBuffer(SL_data,48,0x40);// DC SL
}

void FL3248_FC0_set(uint8_t byte6,uint8_t byte5,uint8_t byte4,uint8_t byte3,uint8_t byte2,uint8_t byte1)
{
  FC0_data[0] = byte6;//BIT41-48
  FC0_data[1] = byte5;
  FC0_data[2] = byte4;
  FC0_data[3] = byte3;
  FC0_data[4] = byte2;
```

```
    FC0_data[5] = byte1;//BIT0-7
    VSB_WriteBuffer(FC0_data,6,0x20);//FC0
}
void FL3248_FC1_set(uint8_t byte6,uint8_t byte5,uint8_t byte4,uint8_t byte3,uint8_t byte2,uint8_t byte1)
{
    FC1_data[0] = byte6;//BIT41-48
    FC1_data[1] = byte5;
    FC1_data[2] = byte4;
    FC1_data[3] = byte3;
    FC1_data[4] = byte2;
    FC1_data[5] = byte1;//BIT0-7
    VSB_WriteBuffer(FC1_data,6,0x22);//FC1
}

void Init3248(void)
{

    SL_set(0xFF); // SL init
    FL3248_FC0_set(0x00,0x00,0x0B,0xFF,0xFF,0xFF);// FC0 init
    FL3248_FC1_set(0x00,0x00,0x03,0xB1,0x00,0xBA);// FC1 init
}

void IS32FL3248_mode1(void)//breath mode
{
    int i,j;
    while(1)
    {
    for(j=0;j<64;j++)
    {
    for(i=0;i<96;i+=2)
    {
        PWM_data[i]=PWM_Gamma64[j];
        PWM_data[i+1]=PWM_Gamma64[j];
    }
        VSB_WriteBuffer(PWM_data,96,0x60);
        delay(40);
    }
        delay(60);

    for(j=64;j>0;j--)
    {
    for(i=0;i<96;i+=2)
    {
        PWM_data[i]=PWM_Gamma64[j-1];
        PWM_data[i+1]=PWM_Gamma64[j-1];
    }
        VSB_WriteBuffer(PWM_data,96,0x60);
        delay(40);
    }
        delay(60);
    }
    }
```

APPENDIX II: IS32FL3248 Arduino Test Code SPI V01A

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

uint8_t PWM_data[96];
uint8_t SL_data[48];
uint8_t FC1_data[6]={0x00,0x00,0x03,0xB1,0x00,0xBA};
uint8_t FC0_data[6]={0x00,0x00,0x0B,0xFF,0xFF,0xFF};
const int slaveSelectPin = 10;

byte PWM_Gamma64[64]=
{
  0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,
  0x08,0x09,0x0b,0x0d,0x0f,0x11,0x13,0x16,
  0x1a,0x1c,0x1d,0x1f,0x22,0x25,0x28,0x2e,
  0x34,0x38,0x3c,0x40,0x44,0x48,0x4b,0x4f,
  0x55,0x5a,0x5f,0x64,0x69,0x6d,0x72,0x77,
  0x7d,0x80,0x88,0x8d,0x94,0x9a,0xa0,0xa7,
  0xac,0xb0,0xb9,0xbf,0xc6,0xcb,0xcf,0xd6,
  0xe1,0xe9,0xed,0xf1,0xf6,0xfa,0xfe,0xff
};

void setup()
{
  // put your setup code here, to run once:
  // set the slaveSelectPin as an output:
  pinMode (slaveSelectPin, OUTPUT);
  // initialize SPI:
  SPI.begin();
  SPI.beginTransaction(SPISettings(20000000, MSBFIRST, SPI_MODE0));
  //SPI.setClockDivider(SPI_CLOCK_DIV4);
  SPI.setDataMode(3);
  pinMode(4,OUTPUT);//SDB
  digitalWrite(4,HIGH);//SDB_HIGH
  Init3248();
}

void loop()
{
  // put your main code here, to run repeatedly:
  IS32FL3248_mode1();
}

void SPI_WriteBuffer(uint8_t* pBuffer, int length_dat, uint8_t Dev_Add)
{
  digitalWrite(slaveSelectPin, LOW);    // take the SS pin low to select the chip:
  SPI.transfer(Dev_Add);    //   send in the address and value via SPI:
  while(length_dat--)
  {
    SPI.transfer(*pBuffer);
    pBuffer++;
  }
  digitalWrite(slaveSelectPin, HIGH);    // take the SS pin high to de-select the chip:
}

// SET the Scaling Register//
void SL_set(uint8_t byte1)
{
  uint8_t i;
  for(i=0;i<48;i+=2)
  {
    SL_data[i] = byte1;
  }
  SPI_WriteBuffer(SL_data,48,0x40);// DC SL
}

void FL3248_FC0_set(uint8_t byte6,uint8_t byte5,uint8_t byte4,uint8_t byte3,uint8_t byte2,uint8_t byte1)
{
  FC0_data[0] = byte6;//BIT41-48
  FC0_data[1] = byte5;
  FC0_data[2] = byte4;
  FC0_data[3] = byte3;
  FC0_data[4] = byte2;
```

```
   FC0_data[5] = byte1;//BIT0-7
   SPI_WriteBuffer(FC0_data,6,0x20);//FC0
}
void FL3248_FC1_set(uint8_t byte6,uint8_t byte5,uint8_t byte4,uint8_t byte3,uint8_t byte2,uint8_t byte1)
{
   FC1_data[0] = byte6;//BIT41-48
   FC1_data[1] = byte5;
   FC1_data[2] = byte4;
   FC1_data[3] = byte3;
   FC1_data[4] = byte2;
   FC1_data[5] = byte1;//BIT0-7
   SPI_WriteBuffer(FC1_data,6,0x22);//FC1
}

void Init3248(void)
{

   SL_set(0xFF); // SL init
   FL3248_FC0_set(0x00,0x00,0x0B,0xFF,0xFF,0xFF);// FC0 init
   FL3248_FC1_set(0x00,0x00,0x03,0xB1,0x00,0xBA);// FC1 init
}

void IS32FL3248_mode1(void)//breath mode
{
   int i,j;
   while(1)
   {
   for(j=0;j<64;j++)
   {
   for(i=0;i<96;i+=2)
   {
      PWM_data[i]=PWM_Gamma64[j];
      PWM_data[i+1]=PWM_Gamma64[j];
   }
      SPI_WriteBuffer(PWM_data,96,0x60);
      delay(40);
   }
      delay(60);

   for(j=64;j>0;j--)
   {
   for(i=0;i<96;i+=2)
   {
      PWM_data[i]=PWM_Gamma64[j-1];
      PWM_data[i+1]=PWM_Gamma64[j-1];
   }
      SPI_WriteBuffer(PWM_data,96,0x60);
      delay(40);
   }
      delay(60);
   }
   }
```