3216 Bicolor LED 3mm (5mm) Dot Matrix Display Information Board User's Guide

3216 BICOLOR LED 3MM (5MM) DOT MATRIX DISPLAY INFOMATION BOARD USER'S GUIDE

Table of Contents

NOTES:

Product Version : Ver 1.0

Document Version : Ver 1.1

3216 BICOLOR LED 3MM (5MM) DOT MATRIX DISPLAY INFORMATION BOARD USER'S GUIDE

Chapter 1. Overview

1.1 Overview

Thanks for using 3216 bicolor LED dot matrix info board series by Sure Electronics. Integrating four HT1632Cs as driver chips on each info board, these info boards support 16-level PWM brightness control and all LED dot matrixes displayed are mapped to the RAM of HT1632Cs. Peripheral circuits are required to light up LEDs via the ports on the boards, providing red, green and even yellow display when red and green LEDs are illuminated simultaneously. These info boards can be used to display digits, letters and even graphs. It is recommended to connect up to 2 boards of the same kind in series for wider applications such as info display in banks, stores, households and so on. You may refer to the following table for members of this series.

TABLE 1-1 3216 BICOLOR LED DOT MATRIX DISPLAY INFO BOARD SERIES

Product Number	Product Name
DE-DP14112	3216 Bicolor LED 3mm Dot Matrix Display Information Board
DE-DP14211	3216 Bicolor LED 5mm Dot Matrix Display Information Board

Note: 4 info boards connected in series are available when the boards are powered by 5V, 2A supply. If the power supply you used can output more power and the info boards are only used to display common ASCII, more boards can be connected in series. However, we assume no liability for any damage or risk caused by this.

FIGURE 1-1 FRONT VIEW OF 3216 LED 3MM DOT MATRIX DISPLAY INFO BOARD

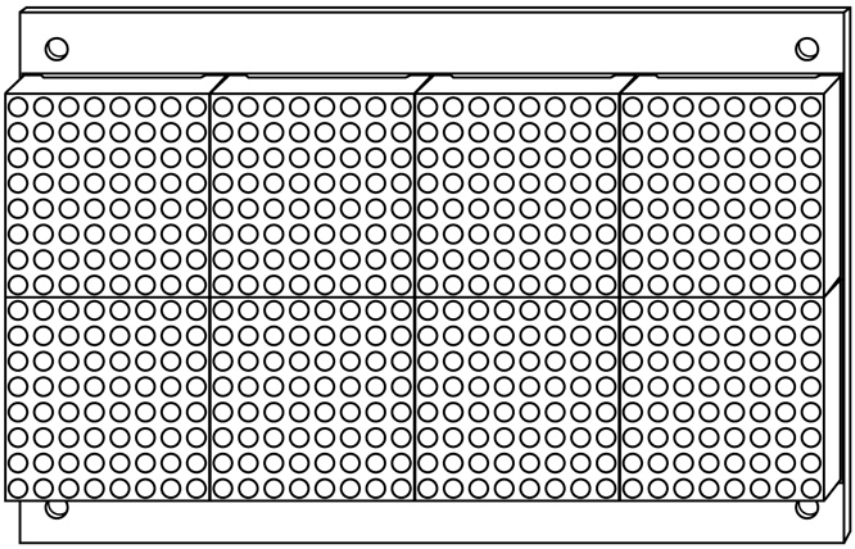


FIGURE 1-2 FRONT VIEW OF 3216 LED 5MM DOT MATRIX DISPLAY INFO BOARD

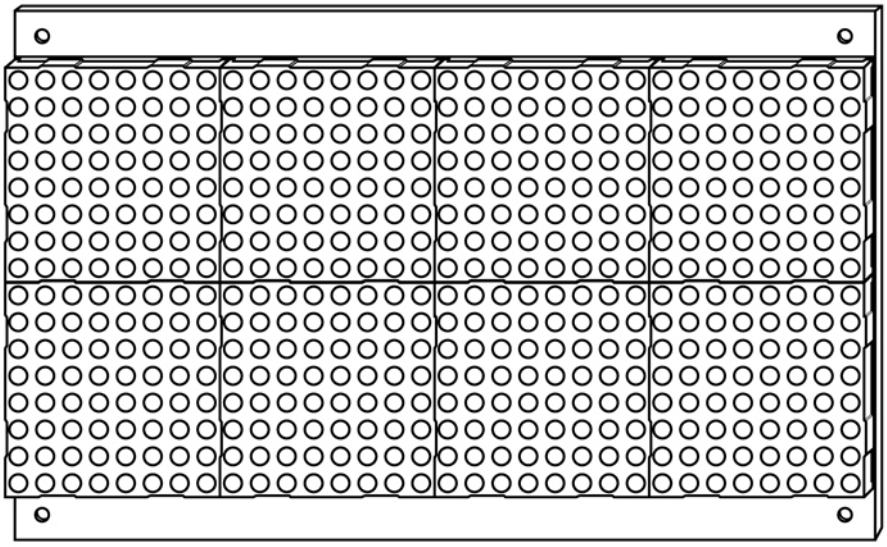


FIGURE 1-3 BACK VIEW OF 3216 LED 3MM DOT MATRIX DISPLAY INFO BOARD

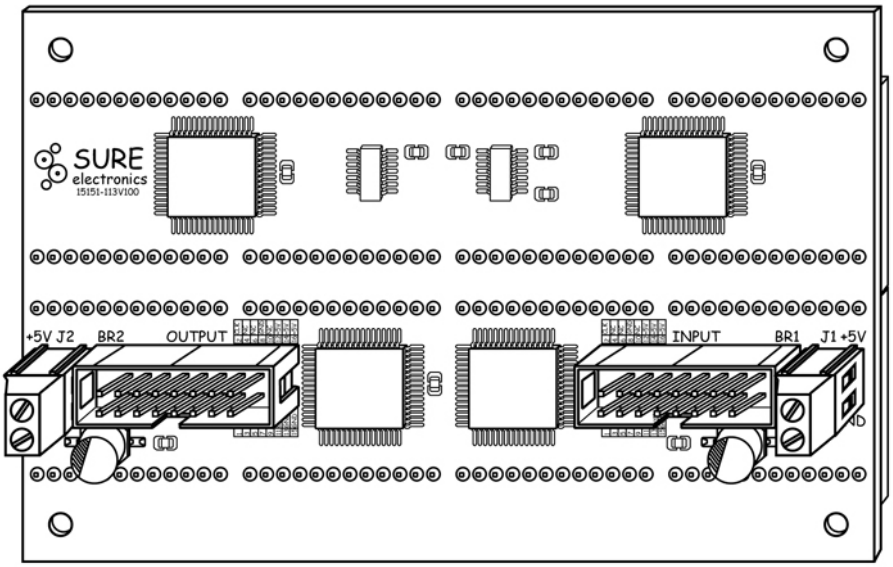
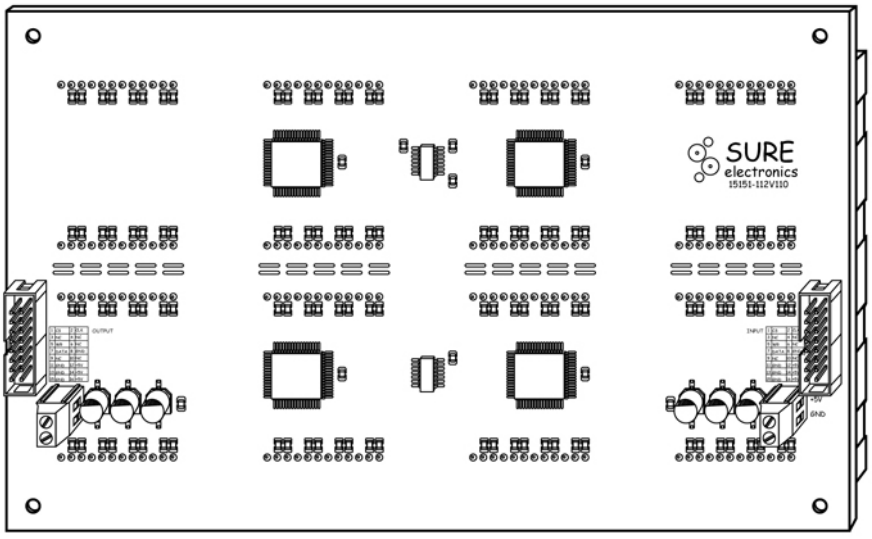


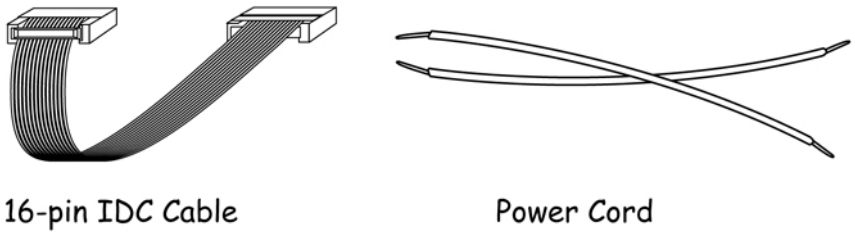
FIGURE 1-4 BACK VIEW OF 3216 LED 5MM DOT MATRIX DISPLAY INFO BOARD



1.2 Quick Start

A 16-pin IDC cable and two power cords are provided for free. DE-DD210 by Sure Electronics is used in this manual as a driver board. Program this driver board to control the display on the info board.

FIGURE 1-5 ACCESSARIES

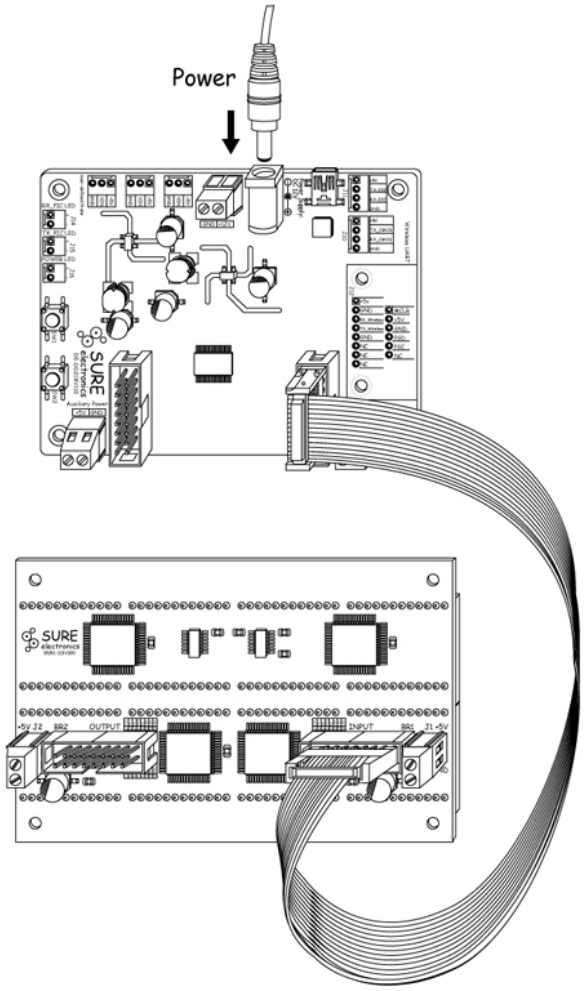


Note:
1. Other driver board can be used. You may refer to **2.2 Port Definition** to do relative adjustments.
2. Sample codes are provided in this manual for reference.

1.2.1 Connection of One Info Board and The Driver Board

Connect BR1 of the info board and BR1 of the driver board with a 16-pin IDC cable.

FIGURE 1-6 CONNECTION OF THE DRIVER BOARD AND ONE INFO BOARD



Program codes to the chip of the driver board and repower the board.

Note: If you're not familiar with programming, try using the sample codes first.

1.2.2 Connection of Many Info Boards

First, auxiliary power cords are recommended to be used when more info boards connected in series. Connect +5V, GND of terminal J2 on one info board and the corresponding +5V, GND of terminal J1 on the next info board with auxiliary power cords. The auxiliary supply should be able to output DC5V 4A. Up to two info boards are recommended to be connected in series.

Connect BR1 of the driver board and BR1 of the info board with a 16-pin IDC cable. Then, as shown in figure 1-7, connect two 3216 info boards and the driver board with two16-pin IDC cables and power cords.

FIGURE 1-7 CONNECTION OF TWO 3216 BICOLOR 3MM LED DOT MATRIX DISPLAY INFO BOARDS IN SERIES

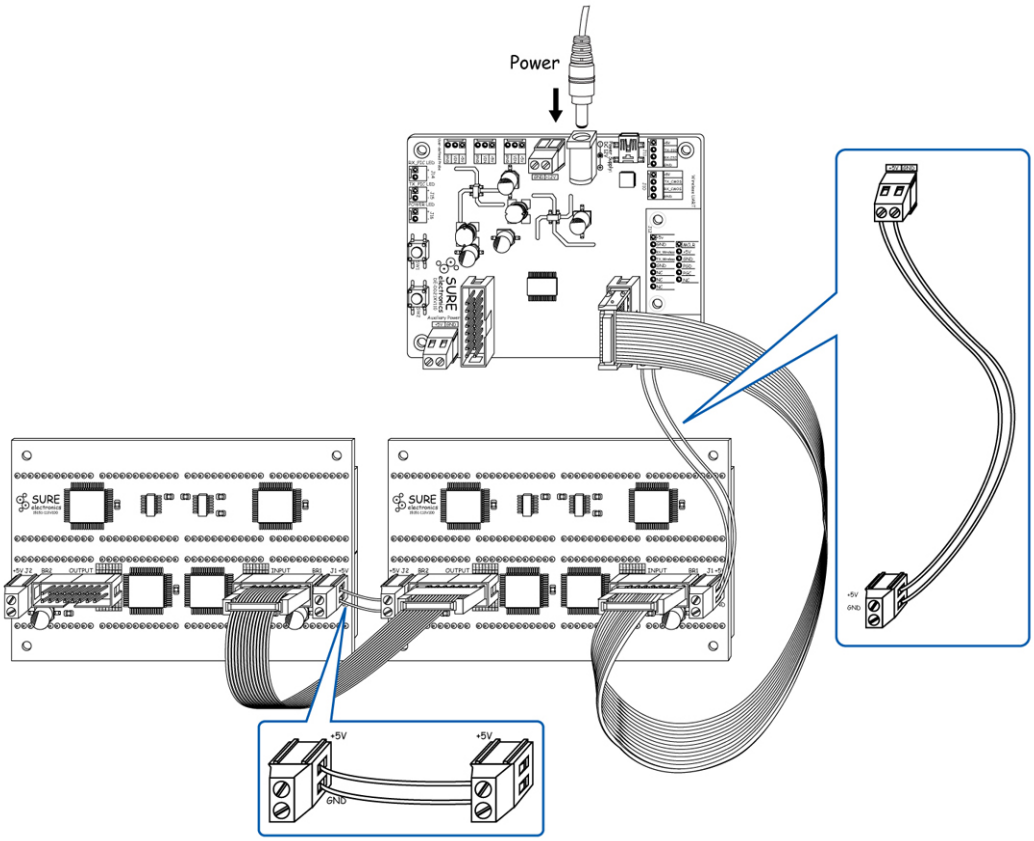
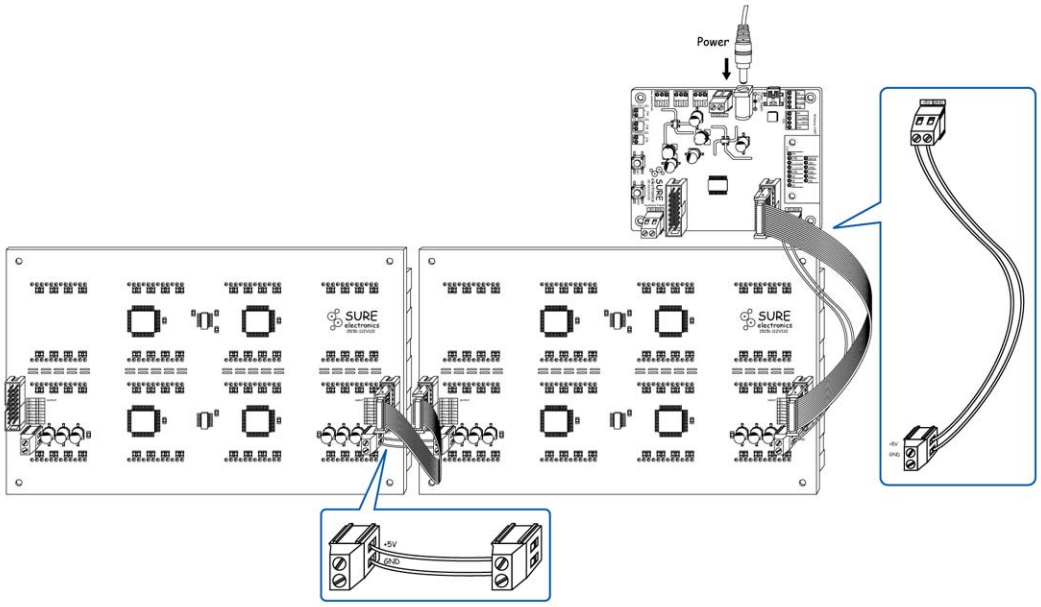


FIGURE 1-8 CONNECTION OF TWO 3216 BICOLOR 5MM LED DOT MATRIX DISPLAY INFO BOARDS IN SERIES



Program the chip on the driver board to control the LED display.

Note: If you use the sample codes, all the boards will display the same content.

Chapter 2. Hardware Detail

2.1 Hardware

1. 8 pieces of 8*8 bicolor LED dot matrix
   Light-emitting diameter of DE-DP14112 is 3mm. Light-emitting diameter of DE-DP14211 is 5mm.
2. LED drive chip (U2, U3, U5, and U6): four HT1632C chips, QFP packaging.
3. 16-pin male sockets (BR1 and BR2): used for data, clock, control signal and +5V supply input.
4. Auxiliary power supply terminal (+5V) (J1and J2): for external power input when more info boards are connected in series.

2.2 Port Definition

TABLE 2-1 PIN DEFINITION OF BR1

Pin Number	Pin Name	Function Description
8, 11, 13, 15	GND	GND
12, 14, 16	VCC	Power Supply
3, 4, 6, 9, 10	NC	No Connection
1	CS	Chip Select signal input
2	CLK	Chip Select clock signal input
5	WR	WRITE data clock input
7	DATA	Data Input

TABLE 2-2 PIN DEFINITION OF BR2

Pin Number	Pin Name	Function Description
8, 11, 13, 15	GND	GND
12, 14, 16	VCC	Power Supply
3, 4, 6, 9, 10	NC	No Connection
1	CS	Chip Select signal output
2	CLK	Chip Select clock signal input
5	WR	WRITE data clock output
7	DATA	Data Output

2.3 Display Memory

Display is controlled by modifying the data stored in RAM of HT1632Cs. Every two 8*8 dot matrix displays on the board are controlled by one HT1632C, as shown in the following figure 2-2. The first half of HT1632C's addresses control the circuits of green LEDs and the rest addresses control circuits of red LEDs.

The distribution of HT1632Cs' corresponding address is shown as follows:

TABLE 2-3 THE CORRESPONDING ADDRESS OF ONE HT1632C

	Com7	Com6	Com5	Com4	Addr	Com3	Com2	Com1	Com0	Addr
Out0	The rest green LEDs of the first matrix				01H	The upper half green LEDs of the first matrix				00H
Out1					03H					02H
Out2					05H					04H
Out3					07H					06H
Out4					09H					08H
Out5					0BH					0AH
Out6					0DH					0CH
Out7					0FH					0EH
Out8	The rest green LEDs of the second matrix				11H	The upper half green LEDs of the second matrix				10H
Out9					13H					12H
Out10					15H					14H
Out11					17H					16H
Out12					19H					18H
Out13					1BH					1AH
Out14					1DH					1CH
Out15					1FH					1EH
Out16	The rest red LEDs of the first matrix				21H	The upper half red LEDs of the first matrix				20H
Out17					23H					22H
Out18					25H					24H
Out19					27H					26H
Out20					29H					28H
Out21					2BH					2AH
Out22					2DH					2CH
Out23					2FH					2EH
Out24	The rest red LEDs of the second matrix				31H	The upper half red LEDs of the second matrix				30H
Out25					33H					32H
Out26					35H					34H
Out27					37H					36H
Out28					39H					38H
Out29					3BH					3AH
Out30					3DH					3CH
Out31					3FH					3EH

FIGURE 2-1 THE CORRESPONDING ADDRESS OF ONE HT1632C

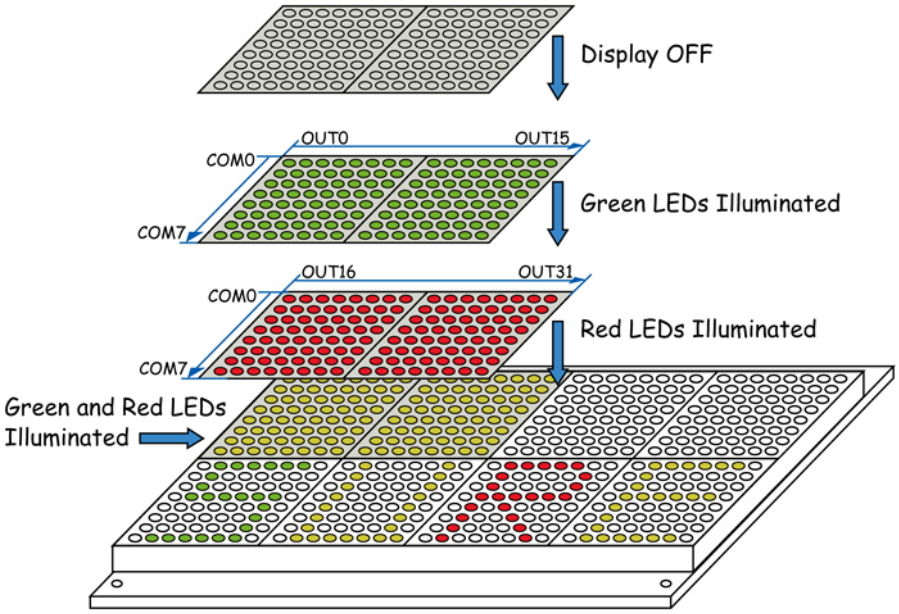
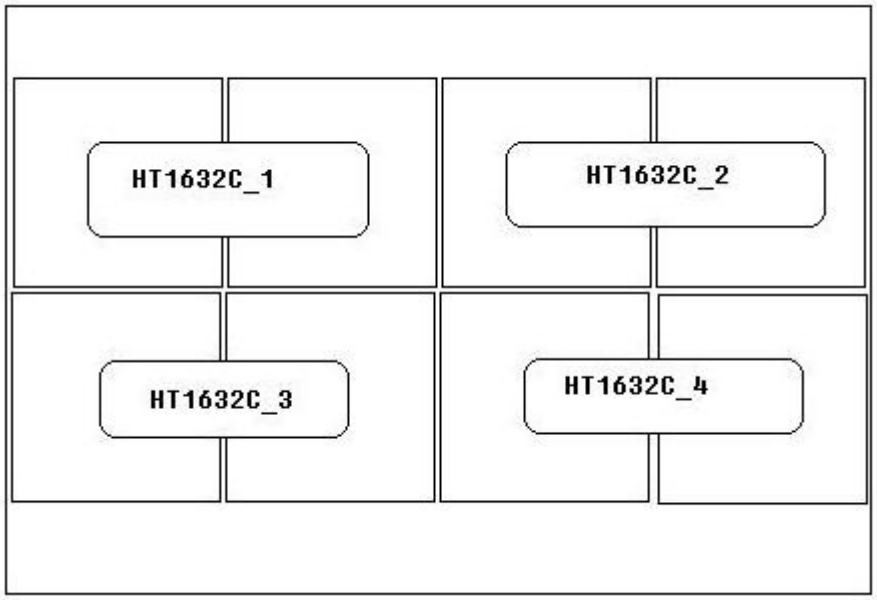


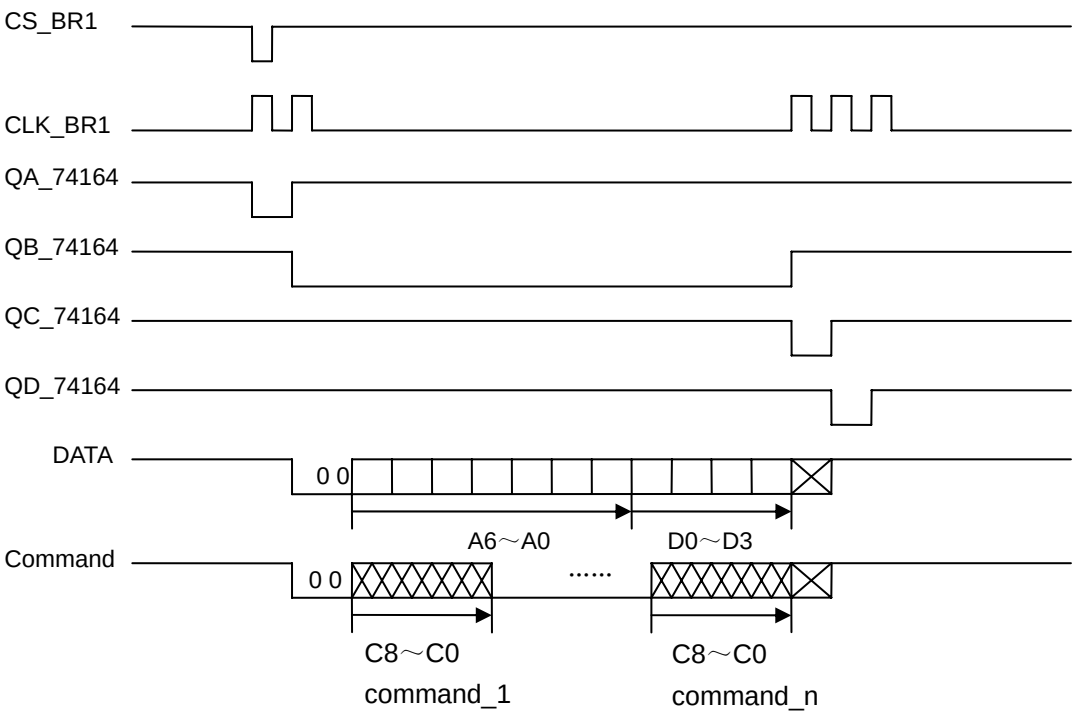
FIGURE 2-2 DISTRIBUTION OF FOUR HT1632CS



2.4 Command Format

CS (CS1、CS2、CS3、CS4) of HT1632C must be set to low before data or command is sent to this HT1632C. When the transmission is complete, CS must be reset to high. Enabling U2 (HT1632C) to individually send data to some address space is taken as an example. The timing diagram is shown as follows:

FIGURE 2-3 TIMING DIAGRAM



Note: You may refer to HT1632C data sheet for details.

2.5 Command Summary

Command summary is shown in the following table. For this info board, these commands - WRITE, SYS EN, LED On, LED Off, RC Master Mode, COM Option and PWM Duty are recommended to be used. Using the commands which are not recommended may cause abnormal display.

FIGURE 2-4 COMMAND SUMMARY

Command Summary

Name	ID	Command Code	D/C	Function	Def.
READ	1 1 0	A6A5A4A3A2A1A0D0D1D2D3	D	Read data from the RAM	
WRITE	1 0 1	A6A5A4A3A2A1A0D0D1D2D3	D	Write data to the RAM	
READ-MODIFY-WRITE	1 0 1	A6A5A4A3A2A1A0D0D1D2D3	D	Read and Write data to the RAM	
SYS DIS	1 0 0	0000-0000-X	C	Turn off both system oscillator and LED duty cycle generator	Yes
SYS EN	1 0 0	0000-0001-X	C	Turn on system oscillator	
LED Off	1 0 0	0000-0010-X	C	Turn off LED duty cycle generator	Yes
LED On	1 0 0	0000-0011-X	C	Turn on LED duty cycle generator	
BLINK Off	1 0 0	0000-1000-X	C	Turn off blinking function	Yes
BLINK On	1 0 0	0000-1001-X	C	Turn on blinking function	
SLAVE Mode	1 0 0	0001-0XXX-X	C	Set slave mode and clock source from external clock, the system clock input from OSC pin and synchronous signal input from SYN pin	
RC Master Mode	1 0 0	0001-10XX-X	C	Set master mode and clock source from on-chip RC oscillator, the system clock output to OSC pin and synchronous signal output to SYN pin	Yes
EXT CLK Master Mode	1 0 0	0001-11XX-X	C	Set master mode and clock source from external clock, the system clock input from OSC pin and synchronous signal output to SYN pin	
COM Option	1 0 0	0010-abXX-X	C	ab=00: N-MOS open drain output and 8 COM option ab=01: N-MOS open drain output and 16 COM option ab=10: P-MOS open drain output and 8 COM option ab=11: P-MOS open drain output and 16 COM option	ab =00
PWM Duty	1 0 0	101X-0000-X	C	PWM 1/16 duty	
	1 0 0	101X-0001-X	C	PWM 2/16 duty	
	1 0 0	101X-0010-X	C	PWM 3/16 duty	
	1 0 0	101X-0011-X	C	PWM 4/16 duty	
	1 0 0	101X-0100-X	C	PWM 5/16 duty	
	1 0 0	101X-0101-X	C	PWM 6/16 duty	
	1 0 0	101X-0110-X	C	PWM 7/16 duty	
	1 0 0	101X-0111-X	C	PWM 8/16 duty	
	1 0 0	101X-1000-X	C	PWM 9/16 duty	
	1 0 0	101X-1001-X	C	PWM 10/16 duty	
	1 0 0	101X-1010-X	C	PWM 11/16 duty	
	1 0 0	101X-1011-X	C	PWM 12/16 duty	
	1 0 0	101X-1100-X	C	PWM 13/16 duty	
	1 0 0	101X-1101-X	C	PWM 14/16 duty	
	1 0 0	101X-1110-X	C	PWM 15/16 duty	
	1 0 0	101X-1111-X	C	PWM 16/16 duty	

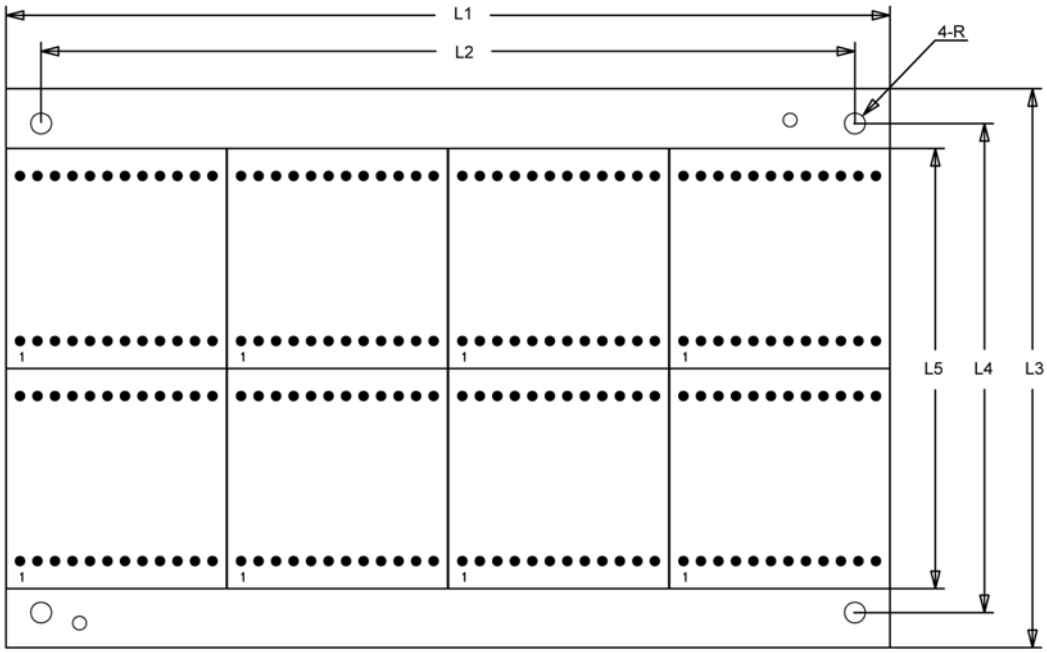
Chapter 3. Electrical Characteristics

TABLE 3-1 ELECTRICAL CHARACTERISTICS

Parameter	Symbol	Value	Unit
Operating Voltage	V_{in}	5	V
Storage Temperature	T_{stg}	-20 to 80	℃
Average Operating Current	I_{avrg}	0.5	A
Maximum Operating Current (All LEDs on, 100% PWM duty cycle)	I_{max}	1.37	A

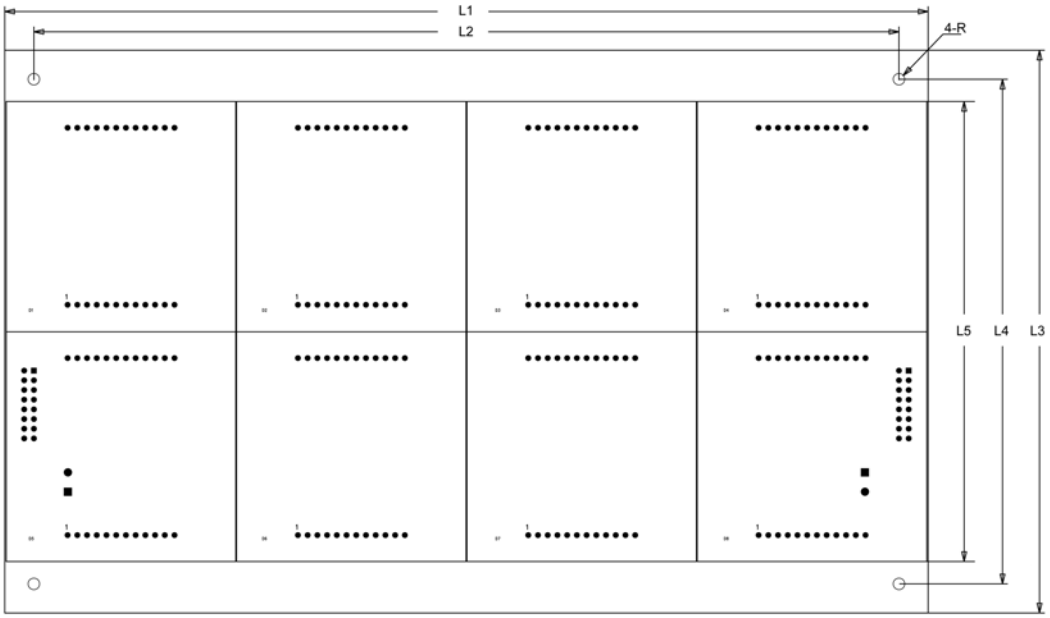
Chapter 4. Mechanical Drawing

FIGURE 4-1 MECHANICAL DRAWING OF ONE 3216 BICOLOR LED 3MM DOT MATRIX DISPLAY INFO BOARD



Symbol	L1	L2	L3	L4	L5	R
inch	5.05	4.65	3.20	2.80	2.54	0.06
mm	128.27	118.11	81.3	71. 1	64.4	1.50

FIGURE 4-2 MECHANICAL DRAWING OF ONE 3216 BICOLOR LED 5MM DOT MATRIX DISPLAY INFO BOARD

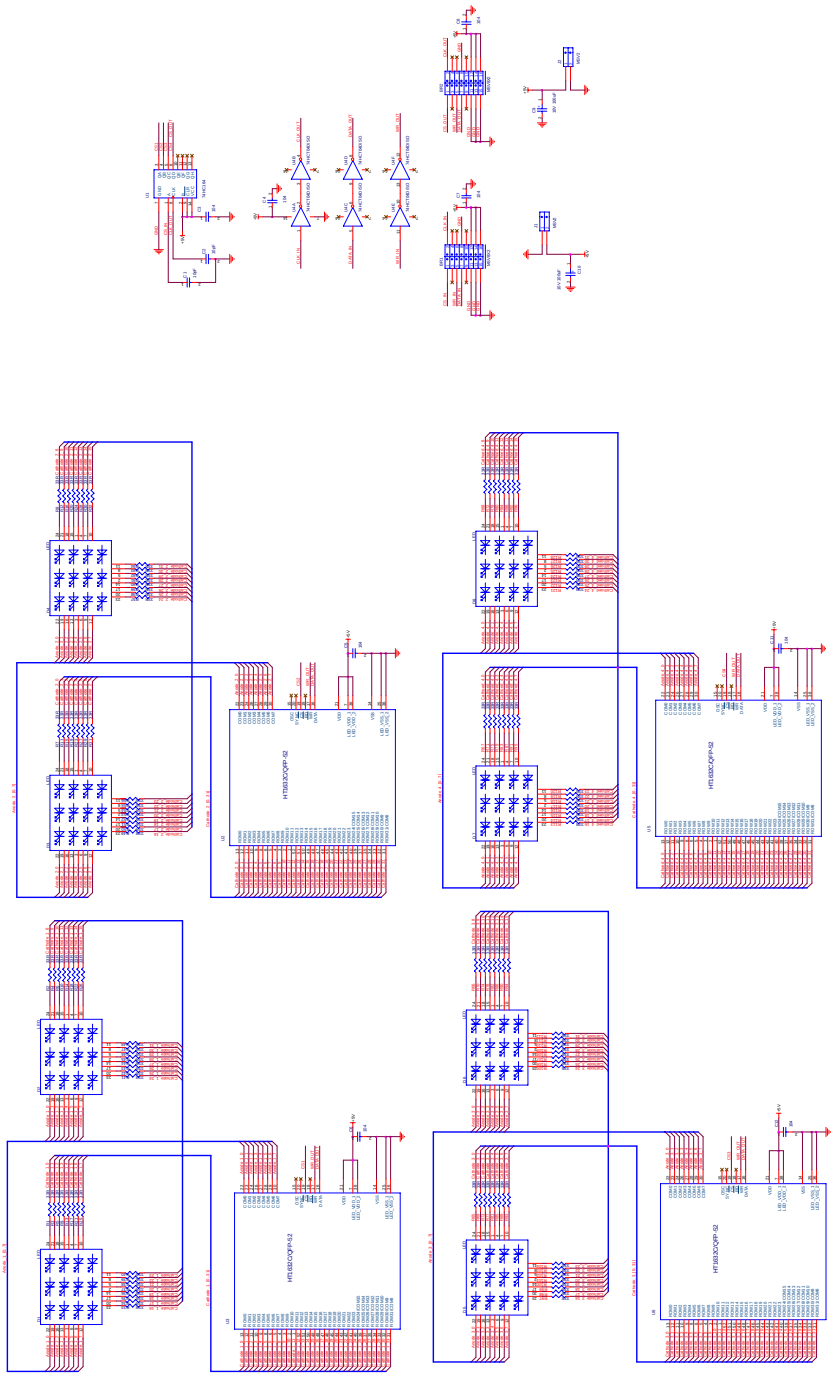


Symbol	L1	L2	L3	L4	L5	R
inch	9.50	8.90	5.80	5.20	4.74	0.06
mm	241.30	226.06	147.32	132.08	120.40	1.50

Chapter 5. Appendix

5.1 Schematic

FIGURE 5-1 SCHEMATIC



5.2 Sample Code

The driver board DE-DD210, integrating PIC16F723 as its master chip, is used as an example. This sample code is used to illuminate the odd rows of LEDs in green.
Compilation environment: MPLAB IDE v8.40
Compiler: HI-TECH ANSI C Compiler PRO 9.65

File "Declare.h"

```
#ifndef   _DECLARE_
#define   _DECLARE_

//Macro definition of ports used
#define   A_74164      RB1
#define   CLK_74164    RB2

#define   CLK          RC3        //Clock line simulating SPI communication (this
                                  //port is also the clock line of SPI communication
                                  //integrated by MCU)
#define   DAT          RC5        //Data line simulating SPI communication (this
                                  //port is also the data line of SPI communication
                                  //integrated by MCU)

//Following is the functions defined in a way of macro definition.
#define CLK_DELAY      NOP()


//Following definition facilitates compilation of HT1632C control commands.
#define SLAVE_MODE           0b100000100000
//[HT1632C]Set slave mode and clock source from external clock

#define RC_MASTER_MODE       0b100000110000
//[HT1632C]Set master mode and clock source from on-chip RC oscillator

#define EXT_CLK_MASTER_MODE  0b100000111000
//[HT1632C]Set master mode and clock source from external clock

#define SYS_EN         0b100000000010    //Turn on system oscillator
#define LED_ON         0b100000000110    //Turn on LED duty cycle generator

#define N_MOS_COM8           0b100001000000
//N-MOS open drain output and 8 common option

#define PWM_10         0b100101010010    //PWM 10/16 duty
#endif
```

File “SampleCode.c”

```
#include <pic.h>
#include "Declare.h"

//*****************************************************************************************************
//Function Name: device file configuration
//Function Feature: configure MCU’s working modes and status
//Input Argument: INTIO: INTOSCIO- internal oscillator, OSC1 and OSC2 used as I/O
//ports
//WDTDIS: disable watchdog timer
//PWRTDIS: disable power-delay timer
//MCLREN: enable MCLR
//UNPROTECT: do NOT protect the code
//BORDIS: brown out reset disable
//BORV25: brown-out reset voltage set to 2.5V nominal
//PLLEN:
//DEBUGEN: in-circuit debugger enabled
//
//VCAPDIS: voltage regulator capacitor disabled
// Output Argument: void
//*****************************************************************************************************
__CONFIG(INTIO & WDTDIS & PWRTEN & MCLREN & UNPROTECT & BORDIS &
BORV25 & PLLEN & DEBUGEN);
__CONFIG(VCAPDIS);

//Constant Definition
#define CHIP_MAX      4                    //Four HT1632Cs on one board

//Function Prototype Declaration
void SystemInit(void);                     //System Initialization
void SetHT1632C_As3208(void);              //Set HT1632Cs work on mode 32*8
void OutputCLK_Pulse(void);                //CLK pin outputs a clock pulse
void OutputA_74164(unsigned char x);       //CS pin output a level
void ChipSelect(int select);               //Chip selection
void CommandWriteHT1632C(unsigned int command);   //Write command to ALL
                                                  //HT1632Cs
void AddressWriteHT1632C(unsigned char address);//Write Address to HT1632Cs
void SPI_ModelConfigure(void);
void SPI_DataSend(const unsigned char data);

void main()
{
      unsigned char i, j;
      SystemInit();                  //System Initialization
      SetHT1632C_As3208();       //Set all HT1632Cs to work in 32*8 master mode
```

```
    for(i=1; i<=CHIP_MAX; i++)
    {
        ChipSelect(i);                  //Chip select the corresponding HT1632C
        AddressWriteHT1632C(0x00);      //Get the selected start address of the chip
        SPI_ModelConfigure();           //Open SPI mode, continuously send data to
HT1632C
        for(j=0; j<32; j++)             //Take "0x00" as the start address, continuously
write data
        {
            SPI_DataSend(0xaa);         //Write data as 0xaa
        }
        SSPCON = 0x11;                  //Close SPI mode
    }
    while(1);
}

//*************************************************************************************************
//Function Name: system initialization
//Function Feature: set corresponding data reading and writing of PORTB and PORTC
//Input Argument: void
//Output Argument: void
//*************************************************************************************************
void SystemInit(void)
{
    IRCF1 = 1;
    IRCF0 = 0;              //Set the frequency of the internal oscillator as 8MHz
    OSCTUNE = 0x1f;         //Oscillator at the maximum frequency
    ANSELB = 0x00;          //PORTB as a digital I/O port
    TRISB = 0x00;           //PORTB as an output port
    PORTB = 0x00;           //Clear PORTB output (PORTB's status is uncertain after
power-on reset)
    TRISC0 = 1;             //PORTC0 (SW1 port) as an input port
    TRISC1 = 1;             //PORTC1 (SW2 port) as an input port
    TRISC3 = 0;             //PORTC3 (CLK signal) as an output port
    TRISC5 = 0;             //PORTC5 (DATA signal) as an output port
    T0IE = 0;               //Turn off interruption of timer0
}


//*************************************************************************************************
//Function Name: SetHT1632C_As3208
//Function Feature: write basic configuration to HT1632C in command words
//Input Argument: void
//Output Argument: void
//*************************************************************************************************
void SetHT1632C_As3208(void)
```

```
{
    CommandWriteHT1632C(SYS_EN);            //Enable system oscillator
    CommandWriteHT1632C(LED_ON);            //Turn on LED
    CommandWriteHT1632C(RC_MASTER_MODE);    //Select on-chip RC as the
clock's master mode. Select this sentence when HT1632C is changed.
    CommandWriteHT1632C(N_MOS_COM8);            //N-MOS open-drain output
                                                //and 32 ROW * 8 COM
    CommandWriteHT1632C(PWM_10);            //PWM 10/16 duty
}


//*************************************************************************************************
//Function Name: OutputCLK_Pulse
//Function Feature: enable CLK_74164 pin to output a clock pulse
//Input Argument: void
//Output Argument: void
//*************************************************************************************************
void OutputCLK_Pulse(void)                      //Output a clock pulse
{
    CLK_74164 = 1;
    CLK_DELAY;
    CLK_74164 = 0;
    CLK_DELAY;
}


//*************************************************************************************************
//Function Name: OutputA_74164
//Function Feature: enable pin A of 74164 to output 0 or 1
//Input Argument: x: if x=1, 74164 outputs high. If x≠1, 74164 outputs low.
//Output Argument: void
//*************************************************************************************************
void OutputA_74164(unsigned char x)                 //Input a digital level to 74164
{
    if(x==1)
    {
        A_74164 = 1;
        CLK_DELAY;
    }
    else
    {
        A_74164 = 0;
        CLK_DELAY;
    }
}
```

```
//*****************************************************************************************************
//Function Name: CommandWriteHT1632C
//Function Feature: Write control commands to all HT1632Cs
//Input Argument: command words written to “command”, specifically stated in “declare”
//function
//Output Argument: void
//Argument Description: compile control commands to all external HT1632Cs for the
//requirement of the project
//*****************************************************************************************************
void CommandWriteHT1632C(unsigned int command)
{
    unsigned char i;
    unsigned int j;
    command = command & 0x0fff;   //12-bit command word, mask upper four bits
    ChipSelect(0);                //Disable all HT1632Cs
    CLK_DELAY;
    ChipSelect(-1);               //Enable all HT1632Cs
    CLK_DELAY;
    for(i=0; i<12; i++)           //Write command words in HI1632C register
    {
        CLK = 0;
        CLK_DELAY;
        j = command & 0x0800;     //Return the MSB
        command = command << 1;//Move the control character to the left one
        j = j >> 11;              //Position the value at the LSB
        DAT = j;                  //Send the value to the data port
        CLK_DELAY;
        CLK = 1;                  //Data transmission (data valid on rising edge)
        CLK_DELAY;
    }
    ChipSelect(0);                //Disable all HT1632Cs
}


//*****************************************************************************************************
//Function Name: AddressWriteHT1632C
//Function Feature: write start address of data to HT1632Cs
//Input Argument: address: address to be written
//Output Argument: void
//*****************************************************************************************************
void AddressWriteHT1632C(unsigned char address)
{
    unsigned char i,temp;
    SSPCON = 0x11;
    address = address & 0x7f;   //7-bit address, mask the MSB
```

```
        CLK = 0;                    //Clock line is 0
        CLK_DELAY;
        DAT = 1;                    //Send “1” to data port
        CLK_DELAY;
        CLK = 1;                    //Data transmission
        CLK_DELAY;
        CLK = 0;
        CLK_DELAY;
        DAT = 0;                    //Send “0” to data port
        CLK_DELAY;
        CLK = 1;                    //Data transmission
        CLK_DELAY;
        CLK = 0;
        CLK_DELAY;
        DAT = 1;                    //Send “1” to data port
        CLK_DELAY;
        CLK = 1;                    //Data transmission
        CLK_DELAY;
        for(i=0; i<7; i++)          //Write “address” to HT1632C register
        {
            CLK = 0;                //Clock line is 0
            CLK_DELAY;
            temp = address & 0x40;          //Return the MSB
            address = address << 1;         //Move the control character to the left one
            temp = temp >> 6;               //Position the value at the LSB
            DAT = temp;                     //Send the value to the data port
            CLK_DELAY;
            CLK = 1;                        //Data transmission
            CLK_DELAY;
        }
}


//*************************************************************************************************
//Function Name: ChipSelect
//Function Feature: enable HT1632C
//Input Argument: select: HT1632C to be selected
//                        If select=0, select none.
//                        If s<0, select all.
//Output Argument: void
//*************************************************************************************************
void ChipSelect(int select)
{
    unsigned char tmp = 0;
    if(select<0)                //Enable all HT1632Cs
    {
```

```
            OutputA_74164(0);
            CLK_DELAY;
            for(tmp=0; tmp<CHIP_MAX; tmp++)
            {
                OutputCLK_Pulse();
            }
        }
        else if(select==0)          //Disable all HT1632Cs
        {
            OutputA_74164(1);
            CLK_DELAY;
            for(tmp=0; tmp<CHIP_MAX; tmp++)
            {
                OutputCLK_Pulse();
            }
        }
        else
        {
            OutputA_74164(1);
            CLK_DELAY;
            for(tmp=0; tmp<CHIP_MAX; tmp++)
            {
                OutputCLK_Pulse();
            }
            OutputA_74164(0);
            CLK_DELAY;
            OutputCLK_Pulse();
            CLK_DELAY;
            OutputA_74164(1);
            CLK_DELAY;
            tmp = 1;
            for( ; tmp<select; tmp++)
            {
                OutputCLK_Pulse();
            }
        }
    }


    //*************************************************************************************************
    //Function Name: SPI_ModelConfigure
    //Function Feature: configure the corresponding data transfer port of PIC microcontroller
    //for SPI communication
    //Input Argument: void
    //Output Argument: void
    //*************************************************************************************************
```

```
void SPI_ModelConfigure(void)
{
    SSPIF = 0;          //Initial state: waiting to send data
    SSPCON = 0x31;  //Write in this register: SSPEN=1 (enable serial port); CKP=1
(CLK high in an idle state); CLK is FOSC/16
    SSPSTAT = 0x80;       //Write in this register: SMP=1(Input data sampled at end of
                          //data output time); CKE=0(data stable on rising edge of SCK)

}


//*****************************************************************************************************
//Function Name: SPI_DataSend
//Function Feature: transmit data in SPI mode of PIC microcontroller
//Input Argument: data: bytes of data to be transmitted
//Output Argument: void
//*****************************************************************************************************
void SPI_DataSend(const unsigned char data)
{
    SSPBUF = data;              //Start sending
    while(!SSPIF);              //Wait for data being sent
    SSPIF = 0;                  //Clear flag
}
```

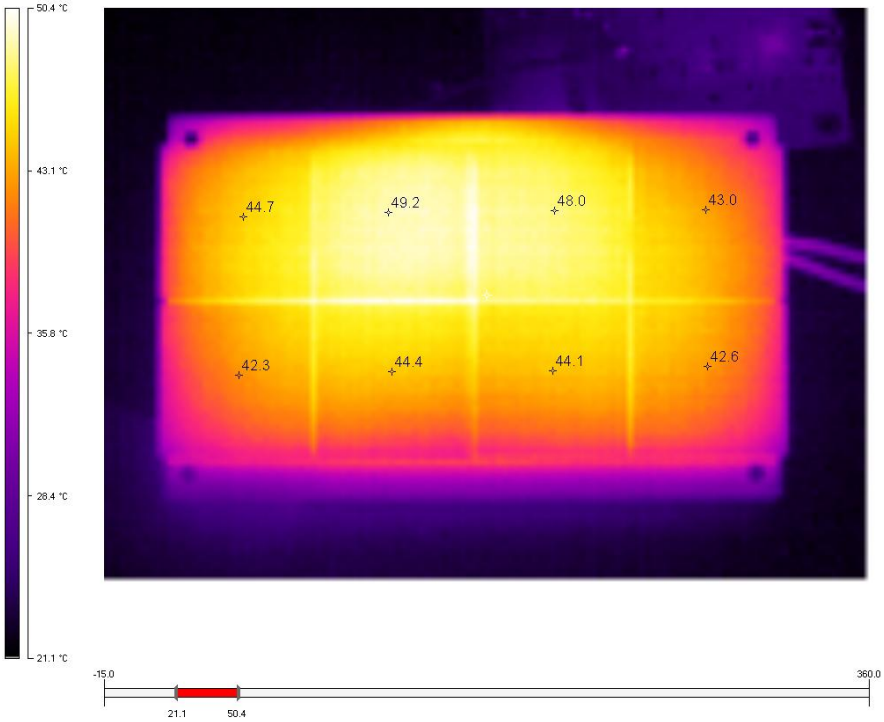
5.3 Heat Dissipation

Following are pictures of heat dissipation gained by Fluke Ti20 Thermal Imager in the condition of info board working at full load, all LEDs on and 100% PWM duty cycle.

5.3.1 3216 Bicolor LED 3mm Dot Matrix Display Info Board

FIGURE 5-2 HEAT DISTRIBUTION OF THE BACK PANEL

FIGURE 5-3 HEAT DISTRIBUTION OF THE FRONT PANEL



Note: The temperature around U2, U3, U5 and U6 is much higher when 3216 bicolor LED 5mm dot matrix display info board product works at full load. Take care and ensure good heat dissipation when using this product.

5.3.2 3216 Bicolor LED 5mm Dot Matrix Display Info Board

FIGURE 5-4 HEAT DISTRIBUTION OF HT1632C'S CENTER ON THE BACK PANEL

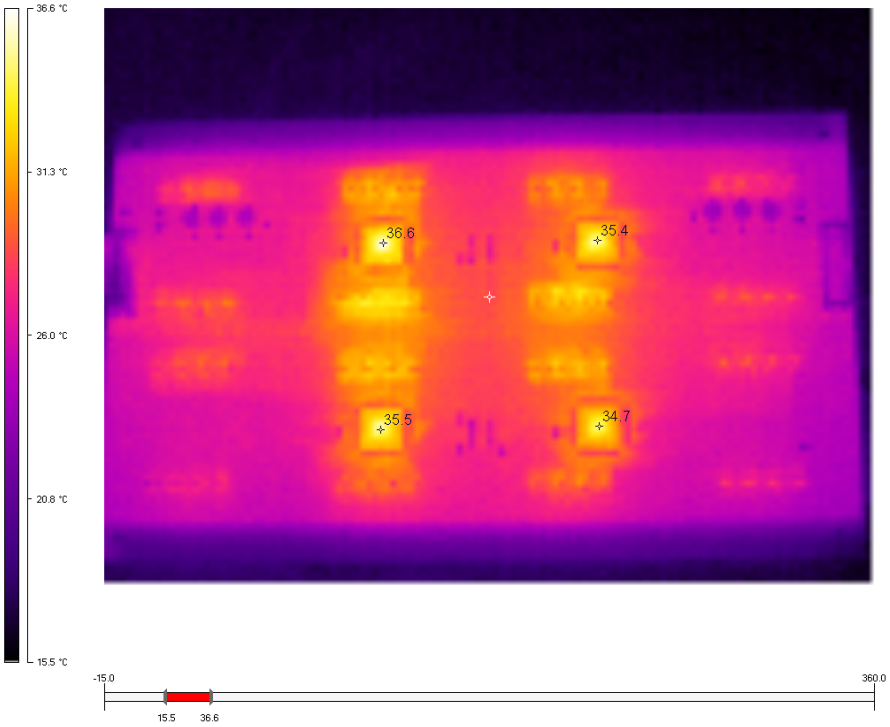
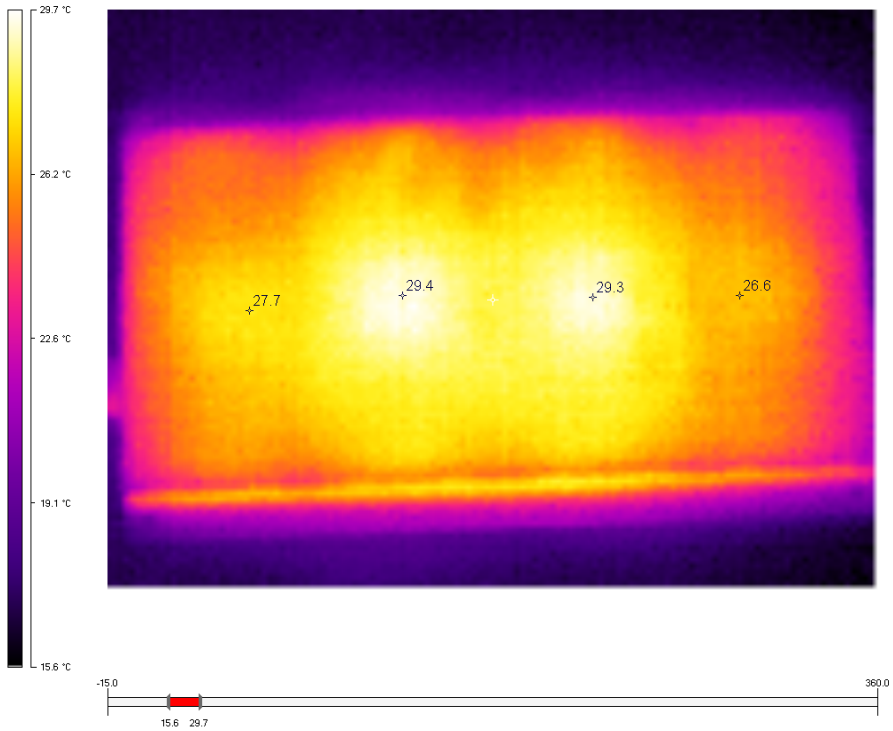


FIGURE 5-5 HEAT DISTRIBUTION OF THE FRONT PANEL



Note: It's recommended to use these products in a good thermal environment.

Chapter 6. Contact Us

Sure Electronics Co., Ltd.
5F, Zone A,
Qinhuai Technology Innovation Center
105-2 DaMing Rd (ZIP:210022)
Nanjing
P.R.China
Tel: +86-13601408832 (For technical questions only)
+86-25-66606340 (English service, from GMT1-10AM)
Fax: +86-25- 66606341-866
Website: www.sure-electronics.com
www.sure-electronics.net