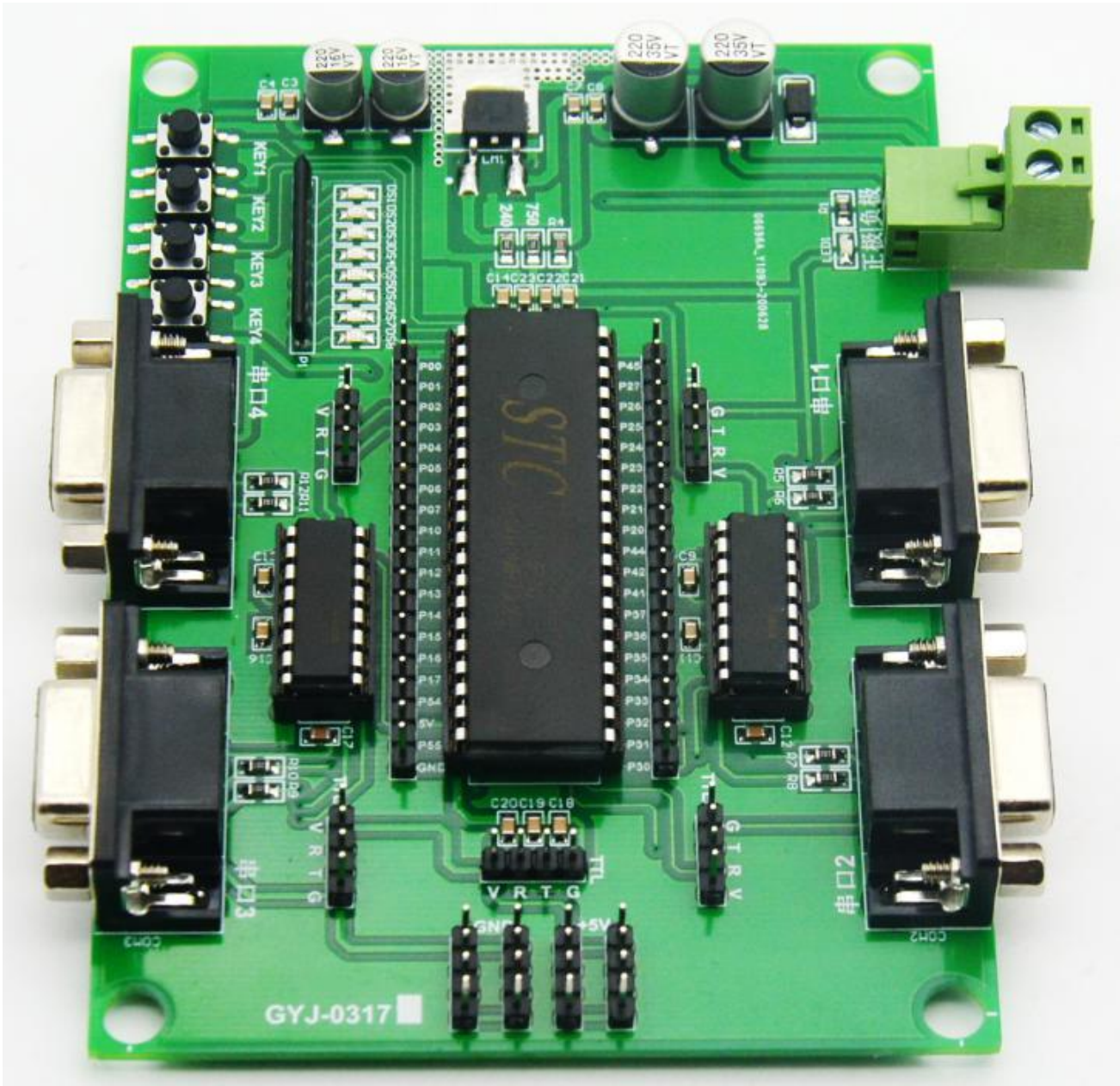


GYJ-0317_STC15W4K32S4 四串口开发板使用手册

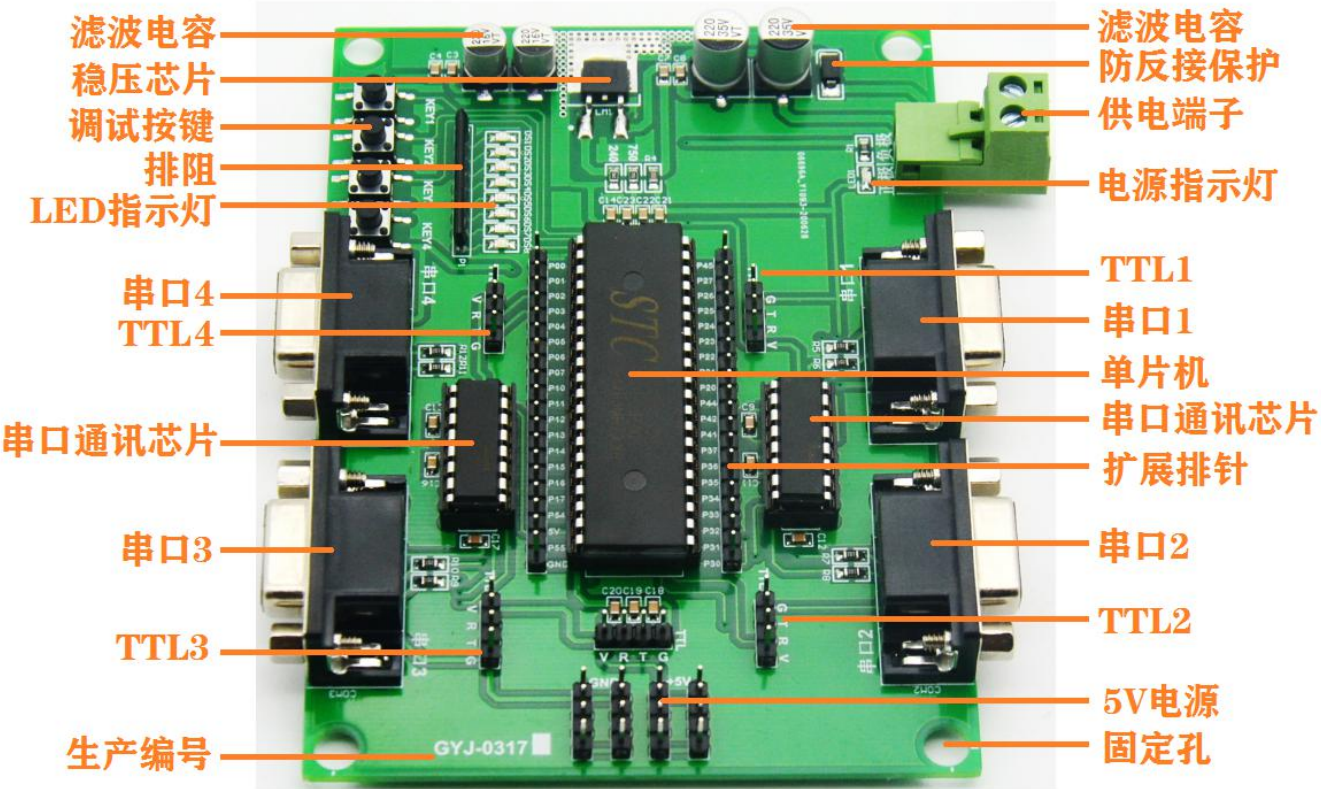


【简要说明】

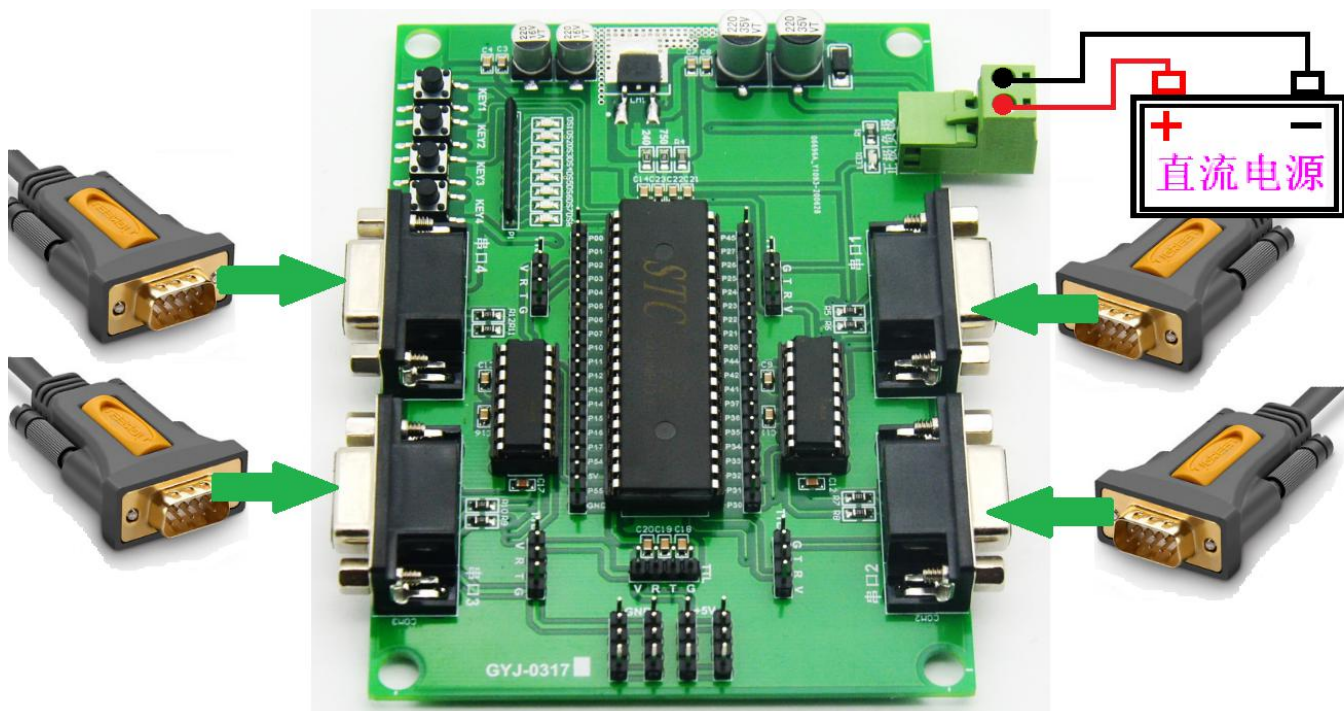
序号	Type ♦ 产品型号	GYJ-0317
1.	Appearance♦产品外形图	参考图在下面
2.	Outline ♦外形尺寸 长 x 宽 x 高	122mmX 80mmX 18mm
3.	Important chips♦重要芯片	STC15W4K32S4
4.	power voltage ♦供电电压	DC5V to DC24V
5.	Features♦主要特征	具有电源指示灯 插拔螺旋压接端子

		支持四路 RS232 独立通讯 支持不同波特率
		通讯协议支持：单字符 十六进制 提供测试程序源码
		波特率支持 1200 to 38400
		防反接保护、过流保护、高温保护
		四位独立按键输入
		四路独立串口通讯 支持二次开发
		所有 IO 口都引出
		8 路 IO 口 LED 灯显示
		支持 TTL 下载
		51 单片机内核，C 语言编程
		提供原理图、尺寸图、例程、编程软件，下载软件
6.	Ambient Temperature ♦环境温度	-30℃ to +70℃
7.	Ambient humidity♦环境湿度	20% to 80%RH

【标注说明】



【接线说明】 4 路串口接线示意图



【单片机程序下载接口】



连接图

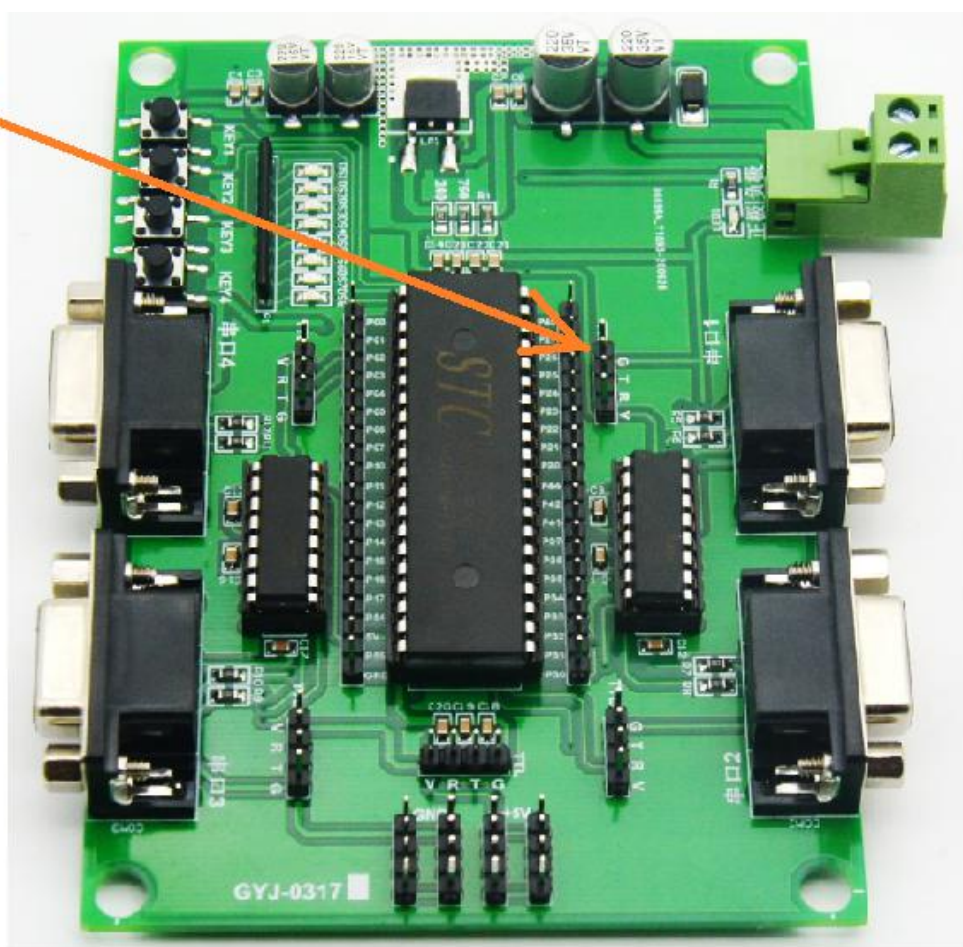
STC下载线 工控板

GND-----G (GND)

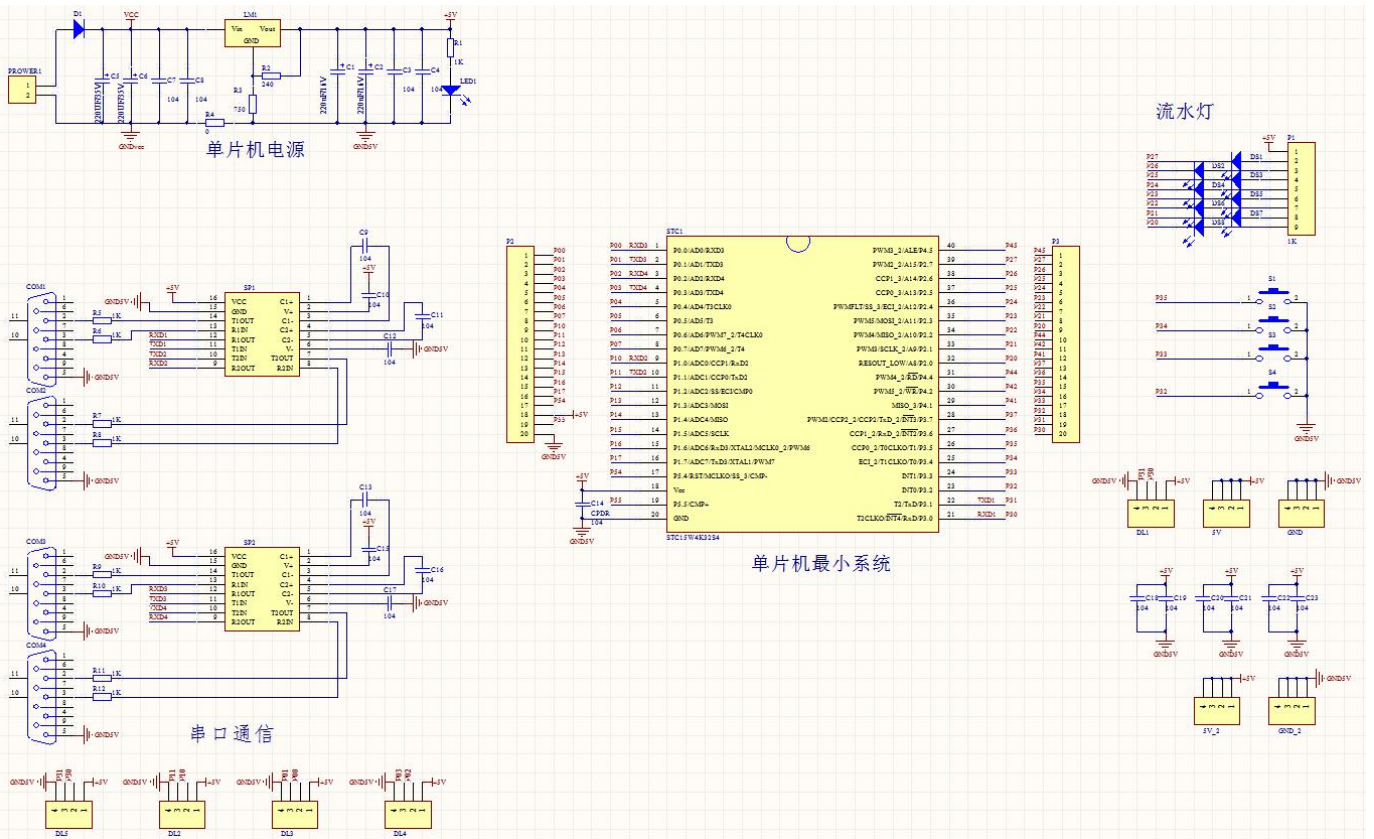
RXD-----T (P3.1)

TXD-----R (P3.0)

5V0-----V (+5V)



【原理图】提供 PDF 格式



串口 1 串口 2 串口 3 串口 4 通信实验:

- 1、该实验是串口 1(P3.6 P3.7)和串口 2(P1.0 P1.1)以及串口 3(P0.0 P0.1)和串口 4(P0.2 P0.3)可同时通信实验;
- 2、下载程序, 下载频率选择 11.0592MHZ;
- 3、打开串口调试助手, 选择串口 1 连接的 USB 转串口模块对应的串口号, 波特率设置为 9600, 在发送区发送 UART1, 可观察到数据接收区收到 UART1 CHECK OK!的字符串;
- 4、打开串口调试助手, 选择串口 2 连接的 USB 转串口模块对应的串口号, 波特率设置为 9600, 在发送区发送 UART2, 可观察到数据接收区收到 UART2 CHECK OK!的字符串;
- 5、打开串口调试助手, 选择串口 3 连接的 USB 转串口模块对应的串口号, 波特率设置为 9600, 在发送区发送 UART3, 可观察到数据接收区收到 UART3 CHECK OK!的字符串;
- 6、打开串口调试助手, 选择串口 4 连接的 USB 转串口模块对应的串口号, 波特率设置为 9600, 在发送区发送 UART4, 可观察到数据接收区收到 UART4 CHECK OK!的字符串;

备注: 使用 USB 转串口模块时务必将模块 GND 与开发板 GND 相连, 即 USB 转串口模块与开发板之间是 3 根线相连。

```
#include "15W4KxxS4.H"
```

```
#include <intrins.h>
```

```
// 加入此头文件后,可使用_nop_库函数
```

```
#include <string.h>
```

```
// 加入此头文件后,可使用 strstr 库函数
```

```
#define uint8 unsigned char
```

```
#define uint16 unsigned int
```

```

#define BAUD 9600                // 波特率
#define TM (65536 - (MAIN_Fosc/4/BAUD))
#define Buf_Max 50
#define S2_S 0x01
#define S2RI 0x01
#define S2TI 0x02
#define S3RI 0x01
#define S3TI 0x02
#define S4RI 0x01
#define S4TI 0x02
// #define S1_S0 0x40             // 设置 S1_S0 = 1;
// #define S1_S1 0x00             // 设置 S1_S1 = 0; 把串口 1 转到 3.6 3.7 上

```

```

uint8 xdata Rec_Buf1[Buf_Max]; // 接收串口 1 缓存数组
uint8 xdata Rec_Buf2[Buf_Max]; // 接收串口 2 缓存数组
uint8 xdata Rec_Buf3[Buf_Max]; // 接收串口 3 缓存数组
uint8 xdata Rec_Buf4[Buf_Max]; // 接收串口 4 缓存数组
uint8 i = 0;
uint8 j = 0;
uint8 m = 0;
uint8 n = 0;
uint16 k=10;

```

```

sbit LED1=P2^7;
sbit LED2=P2^6;
sbit LED3=P2^5;
sbit LED4=P2^4;
sbit LED5=P2^3;
sbit LED6=P2^2;
sbit LED7=P2^1;
sbit LED8=P2^0;
sbit S1=P3^5;
sbit S2=P3^4;
sbit S3=P3^3;
sbit S4=P3^2;

```

```

/*****

```

延时函数

```

*****/

```

```

void delay(uint8 t)

```

```

{
    uint16 i,j;
    for(i=0;i<t;i++)

```

```

{
for(j=30000;j>0;j--);
{;
}
}
}

/*****
* 描 述：串口 1/2/3/4 初始化函数
* 入 参：无
* 返回值：无
备注：波特率 9600bps    晶振 11.0592MHz
*****/

void Uart1234_Init(void)
{
// P_SW1|=0X80;          //选择 P16 P17 为串口 1
//      P_SW1&=0XBF;      //选择 P16 P17 为串口 1
//      P_SW2|=S2_S;      //选择 P46 P47 为串口 2
// ACC = P_SW1;
// ACC &= ~(S1_S0 | S1_S1); //S1_S0=1 S1_S1=0
// ACC |= S1_S0;          //(P3.6/RxD_2, P3.7/TxD_2)
P_SW1 = ACC;
//串口 1 配置
PCON &= 0x3f;           //串口 1 波特率不倍速，串行口工作方式由 SM0、SM1 决定
SCON = 0x50;            //串口 1 的 8 位数据,可变波特率，启动串行接收器
AUXR |= 0x01;           //串口 1 选择定时器 2 为波特率发生器
//串口 2 配置
S2CON = 0x50;           //串口 2 的 8 位数据,可变波特率
//串口 3 配置
S3CON |= 0x10;          //串口 3 启动串行接收器
S3CON &= 0x30;          //串口 3 选择定时器 2 为波特率发生器，8 位数据,可变波特率
//串口 4 配置
S4CON |= 0x10;          //启动串行接收器
S4CON &= 0x30;          //8 位数据,可变波特率，串口 4 选择定时器 2 为波特率发生器

AUXR |= 0x04;           //定时器 2 时钟为 Fosc,即 1T
T2L = 0xE0;             //设定定时初值
T2H = 0xFE;             //设定定时初值
AUXR |= 0x10;           //启动定时器 2
}

//PWMC
/*****
* 描 述：串口 1 发送数据函数
* 入 参：uint8 数据
* 返回值：无

```

```

*****/
void SendDataByUart1(uint8 dat)
{
    SBUF = dat;           //写数据到 UART 数据寄存器
    while(TI == 0);       //在停止位没有发送时，TI 为 0 即一直等待
    TI = 0;               //清除 TI 位（该位必须软件清零）
}

/*****
* 描 述：串口 1 发送字符串函数
* 入 参：字符串
* 返回值：无
*****/
void SendStringByUart1(uint8 *s)
{
    while(*s)
    {
        SendDataByUart1(*s++);    //将字符串中的字符一个一个发送
    }
}

/*****
功能描述：握手成功与否函数
入口参数：uint8 *a
返回值：位
*****/
bit Hand1(uint8 *a)
{
    if(strstr(Rec_Buf1,a)!=NULL)    //判断字符串 a 是否是字符串 Rec_Buf1 的子串
    return 1;                       //如果字符串 a 是字符串 Rec_Buf1 的子串
    else
    return 0;                       //如果字符串 a 不是字符串 Rec_Buf1 的子串
}

/*****
功能描述：清除串口 1 缓存内容函数
入口参数：无
返回值：无
*****/
void CLR_Buf1(void)
{
    uint8 k;
    for(k=0;k<Buf_Max;k++)    //将串口 1 缓存数组的值都清为零
    {

```

```

Rec_Buf1[k] = 0;
}
i = 0;
}

/*****
* 描 述：串口 1 中断服务函数
* 入 参：无
* 返回值：无
*****/

void Uart1() interrupt UART1_VECTOR using 1
{
    ES = 0;                // 串口 1 中断关闭
    if (RI)                // 串行接收到停止位的中间时刻时，该位置 1
    {
        RI = 0;           // 清除 RI 位（该位必须软件清零）
        Rec_Buf1[i] = SBUF; // 把串口 1 缓存 SBUF 寄存器数据依次存放到数组 Rec_Buf1 中
        i++;
        if(i>Buf_Max)      // 接收数大于定义接收数组最大个数时，覆盖接收数组之前值
        {
            i = 0;
        }
    }
    if (TI)                // 在停止位开始发送时，该位置 1
    {
        TI = 0;           // 清除 TI 位（该位必须软件清零）
    }
    ES = 1;               // 串口 1 中断打开
}

/*****
* 描 述：串口 2 发送数据函数
* 入 参：uint8 数据
* 返回值：无
*****/

void SendDataByUart2(uint8 dat)
{
    S2BUF = dat;           // 写数据到 UART 数据寄存器
    while(!(S2CON&S2TI));  // 在停止位没有发送时，S2TI 为 0 即一直等待
    S2CON&=~S2TI;         // 清除 S2CON 寄存器对应 S2TI 位（该位必须软件清零）
}

/*****
* 描 述：串口 2 发送字符串函数

```


* 入 参：字符串

* 返回值：无

*****/

void SendStringByUart2(uint8 *s)

```
{
    IE2 &= 0xFE;          // 串口 2 中断关闭
    while (*s)             //检测字符串结束标志
    {
        SendDataByUart2(*s++);    //发送当前字符
    }
    IE2 |= 0x01;          // 串口 2 中断打开
}
```

*****/

功能描述：握手成功与否函数

入口参数：uint8 *a

返回值：位

*****/

bit Hand2(uint8 *a)

```
{
    if(strstr(Rec_Buf2,a)!=NULL)    //判断字符串 a 是否是字符串 Rec_Buf2 的子串
        return 1;                  //如果字符串 a 是字符串 Rec_Buf2 的子串
    else
        return 0;                  //如果字符串 a 不是字符串 Rec_Buf2 的子串
}
```

*****/

功能描述：清除串口 2 缓存内容函数

入口参数：无

返回值：无

*****/

void CLR_Buf2(void)

```
{
    unsigned char k;
    for(k=0;k<Buf_Max;k++)    //将串口 2 缓存数组的值都清为零
    {
        Rec_Buf2[k] = 0;
    }
    j = 0;
}
```

*****/

* 描 述：串口 2 中断服务函数

* 入 参：无

```

* 返回值：无
*****/

void Uart2() interrupt UART2_VECTOR using 1
{
    IE2 &= 0xFE;                // 串口 2 中断关闭
    if (S2CON & S2RI)            // 串行接收到停止位的中间时刻时，该位置 1
    {
        S2CON &= ~S2RI;         // 清除 S2CON 寄存器对应 S2RI 位（该位必须软件清零）
        Rec_Buf2[j] = S2BUF;     // 把串口 2 缓存 SBUF 寄存器数据依次存放到数组 Rec_Buf2 中
        j++;
        if (j > Buf_Max)         // 接收数大于定义接收数组最大个数时，覆盖接收数组之前值
        {
            j = 0;
        }
    }
    if (S2CON & S2TI)            // 在停止位开始发送时，该位置 1
    {
        S2CON &= ~S2TI;         // 清除 S2CON 寄存器对应 S2TI 位（该位必须软件清零）
    }
    IE2 |= 0x01;                // 串口 2 中断打开
}

/*****
* 描 述：串口 3 发送数据函数
* 入 参：uint8 数据
* 返回值：无
*****/

void SendDataByUart3(uint8 dat)
{
    S3BUF = dat;                // 写数据到 UART 数据寄存器
    while (!(S3CON & S3TI));     // 在停止位没有发送时，S3TI 为 0 即一直等待
    S3CON &= ~S3TI;             // 清除 S3CON 寄存器对应 S3TI 位（该位必须软件清零）
}

/*****
* 描 述：串口 3 发送字符串函数
* 入 参：字符串
* 返回值：无
*****/

void SendStringByUart3(char *s)
{
    IE2 &= 0xF7;                // 串口 3 中断关闭
    while (*s)                  // 检测字符串结束标志
    {

```

```

SendDataByUart3(*s++);          //发送当前字符
}
IE2 |= 0x08;                    // 串口 3 中断打开
}

/*****
功能描述：握手成功与否函数
入口参数：uint8 *a
返回值：位
*****/
bit Hand3(uint8 *a)
{
if(strstr(Rec_Buf3,a)!=NULL)    //判断字符串 a 是否是字符串 Rec_Buf3 的子串
return 1;                      //如果字符串 a 是字符串 Rec_Buf3 的子串
else
return 0;                      //如果字符串 a 不是字符串 Rec_Buf3 的子串
}

/*****
功能描述：清除缓存内容函数
入口参数：无
返回值：无
*****/
void CLR_Buf3(void)
{
uint8 k;
for(k=0;k<Buf_Max;k++)        //将串口 3 缓存数组的值都清为零
{
Rec_Buf3[k] = 0;
}
m = 0;
}

/*****
* 描 述：串口 3 中断服务函数
* 入 参：无
* 返回值：无
*****/
void Uart3() interrupt UART3_VECTOR using 1
{
IE2 &= 0xF7;                  // 串口 3 中断关闭
if (S3CON & S3RI)              //串行接收到停止位的中间时刻时，该位置 1
{
S3CON &= ~S3RI;               //清除 S3CON 寄存器对应 S3RI 位（该位必须软件清零）
}
}

```

```

Rec_Buf3[m] = S3BUF;          //把串口 3 缓存 SBUF 寄存器数据依次存放到数组 Rec_Buf3 中
m++;
if(m>Buf_Max)                 //接收数大于定义接收数组最大个数时，覆盖接收数组之前值
{
m = 0;
}
}
if (S3CON & S3TI)             //在停止位开始发送时，该位置 1
{
S3CON &= ~S3TI;               //清除 S3CON 寄存器对应 S3TI 位（该位必须软件清零）
}
IE2 |= 0x08;                  // 串口 3 中断打开
}

```

/******

* 描 述：串口 4 发送数据函数

* 入 参：uint8 数据

* 返回值：无

*****/

```
void SendDataByUart4(uint8 dat)
```

```

{
S4BUF = dat;                  //写数据到 UART 数据寄存器
while(!(S4CON&S4TI));        //在停止位没有发送时，S4TI 为 0 即一直等待
S4CON&=~S4TI;                //清除 S4CON 寄存器对应 S4TI 位（该位必须软件清零）
}

```

/******

* 描 述：串口 4 发送字符串函数

* 入 参：字符串

* 返回值：无

*****/

```
void SendStringByUart4(char *s)
```

```

{
IE2 &= 0xEF;                  //串口 4 中断关闭
while (*s)                    //检测字符串结束标志
{
SendDataByUart4(*s++);       //发送当前字符
}
IE2 |= 0x10;                  //串口 4 中断打开
}

```

/******

功能描述：握手成功与否函数

入口参数：unsigned char *a

返回值：位

```
*****/
```

```
bit Hand4(unsigned char *a)
```

```
{
```

```
if(strstr(Rec_Buf4,a)!=NULL)    //判断字符串 a 是否是字符串 Rec_Buf4 的子串
```

```
return 1;                        //如果字符串 a 是字符串 Rec_Buf4 的子串
```

```
else
```

```
return 0;                        //如果字符串 a 不是字符串 Rec_Buf4 的子串
```

```
}
```

```
/******
```

功能描述：清除缓存内容函数

入口参数：无

返回值：无

```
*****/
```

```
void CLR_Buf4(void)
```

```
{
```

```
unsigned char k;
```

```
for(k=0;k<Buf_Max;k++)    //将串口 4 缓存数组的值都清为零
```

```
{
```

```
Rec_Buf4[k] = 0;
```

```
}
```

```
n = 0;
```

```
}
```

```
/******
```

* 描 述：串口 4 中断服务函数

* 入 参：无

* 返回值：无

```
*****/
```

```
void Uart4() interrupt UART4_VECTOR
```

```
{
```

```
IE2 &= 0xEF;                // 串口 4 中断关闭
```

```
if(S4CON & S4RI)            //串行接收到停止位的中间时刻时，该位置 1
```

```
{
```

```
S4CON &= ~S4RI;             //清除 S4CON 寄存器对应 S4RI 位（该位必须软件清零）
```

```
Rec_Buf4[n] = S4BUF;        //把串口 4 缓存 SBUF 寄存器数据依次存放到数组 Rec_Buf4 中
```

```
n++;
```

```
if(n>Buf_Max)               //接收数大于定义接收数组最大个数时，覆盖接收数组之前值
```

```
{
```

```
n = 0;
```

```
}
```

```
}
```

```
if(S4CON & S4TI)            //在停止位开始发送时，该位置 1
```

```

{
S4CON &= ~S4TI;          //清除 S4CON 寄存器对应 S4TI 位（该位必须软件清零）
}
IE2 |= 0x10;              // 串口 4 中断打开
}

/*****
* 描 述：主函数
* 入 参：无
* 返回值：无
*****/

int main()
{
////////////////////////////////////
//注意: STC15W4K32S4 系列的芯片,上电后所有与 PWM 相关的 IO 口均为
//      高阻态,需将这些口设置为准双向口或强推挽模式方可正常使用
//相关 IO: P0.6/P0.7/P1.6/P1.7/P2.1/P2.2
//          P2.3/P2.7/P3.7/P4.2/P4.4/P4.5
////////////////////////////////////

P0M0=0X00;P0M1=0X00;
P1M0=0X00;P1M1=0X00;
P2M0=0X00;P2M1=0X00;
P3M0=0X00;P3M1=0X00;
P4M0=0X00;P4M1=0X00;
Uart1234_Init();          // 串口 1/2/3/4 初始化
ES = 1;                   // 串口 1 中断打开
IE2 |= 0x01;              // 串口 2 中断打开
IE2 |= 0x08;              // 串口 3 中断打开
IE2 |= 0x10;              // 串口 4 中断打开
EA = 1;                   // 总中断打开
LED1=0;delay(k);LED1=1;
LED2=0;delay(k);LED2=1;
LED3=0;delay(k);LED3=1;
LED4=0;delay(k);LED4=1;
LED5=0;delay(k);LED5=1;
LED6=0;delay(k);LED6=1;
LED7=0;delay(k);LED7=1;
LED8=0;delay(k);LED8=1;
while(1)
{
if(S1==0){LED1=0;LED2=0;LED3=0;LED4=0;LED5=1;LED6=1;LED7=1;LED8=1;}

```

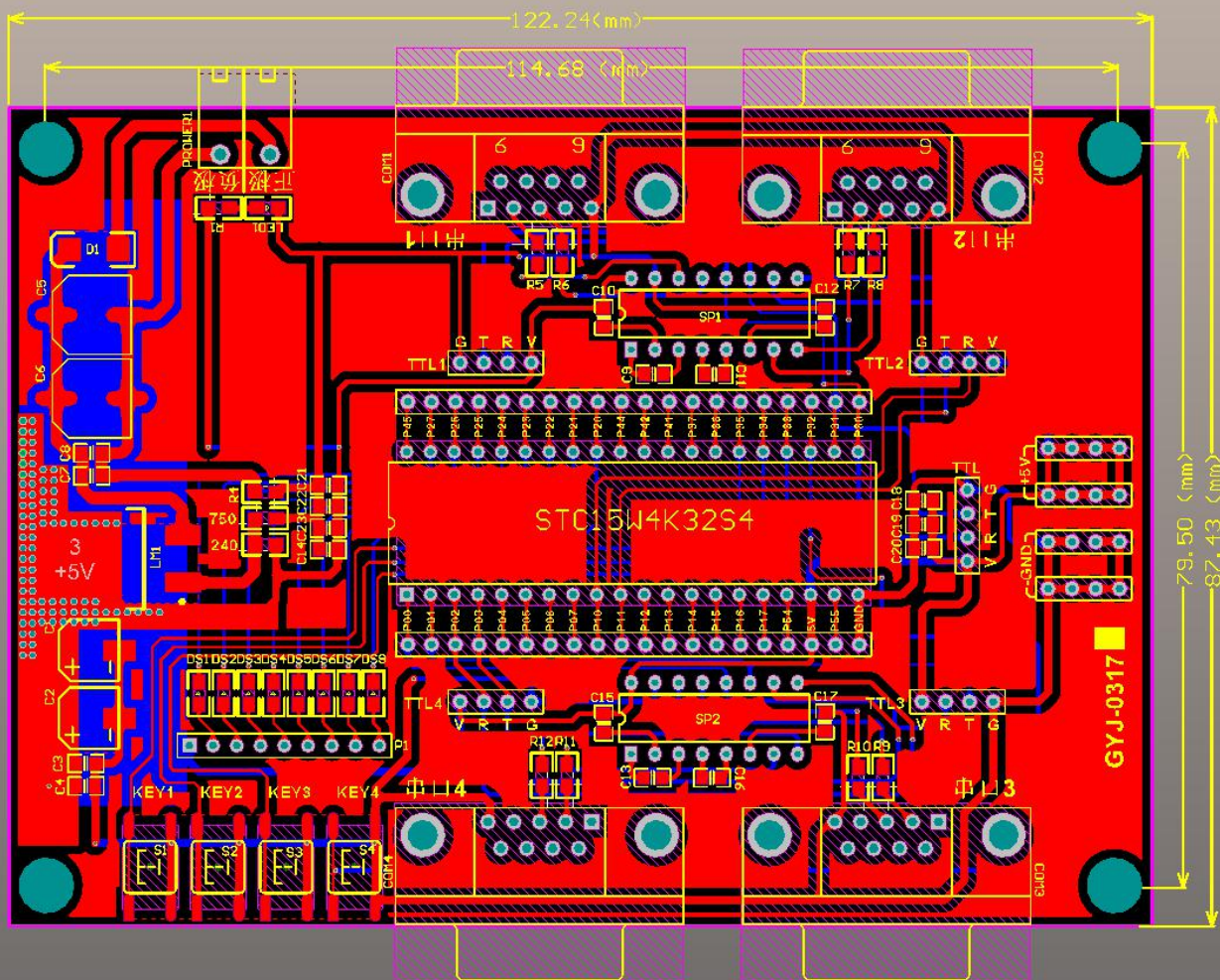
```

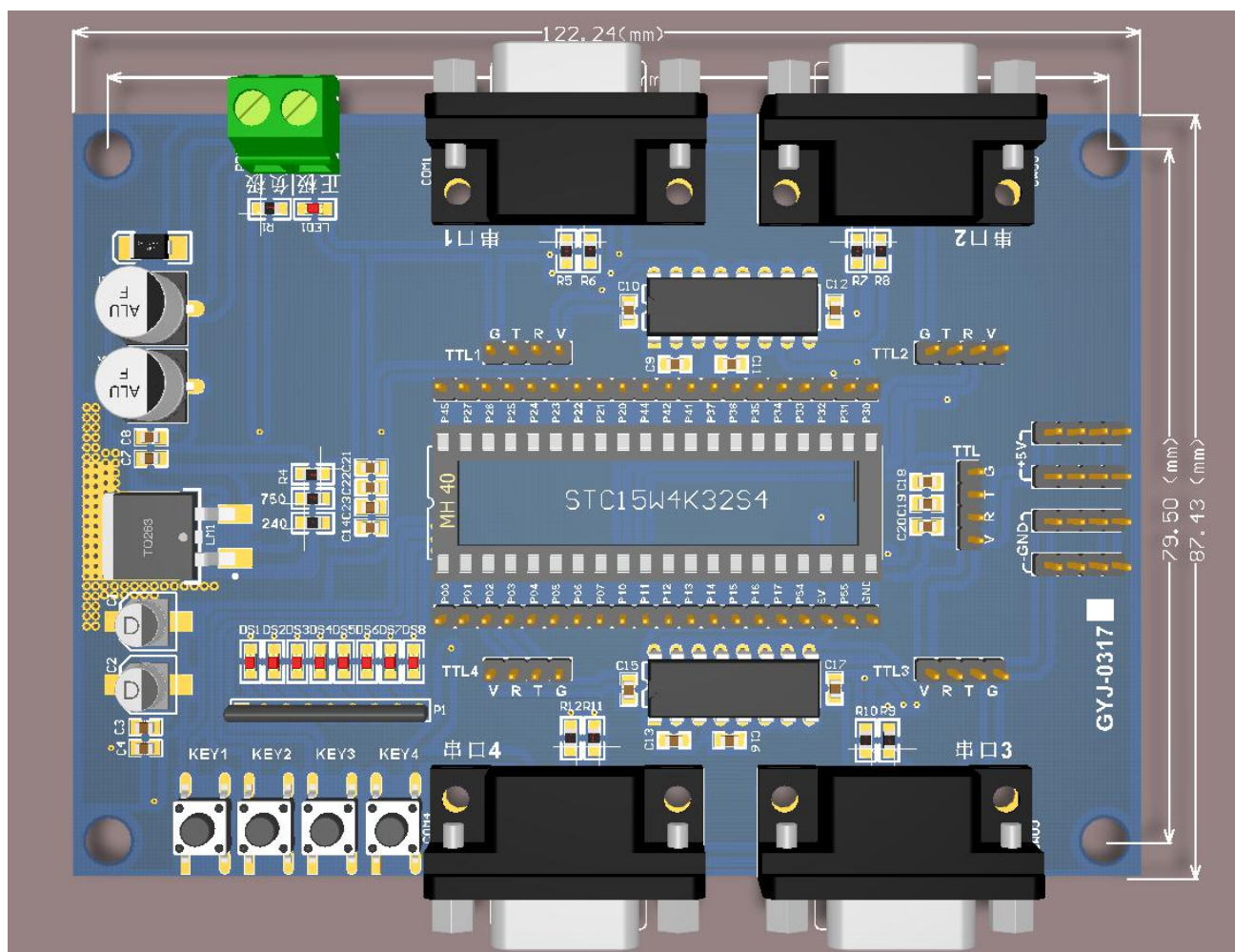
if(S2==0){LED1=1;LED2=1;LED3=1;LED4=1;LED5=0;LED6=0;LED7=0;LED8=0;}
if(S3==0){LED1=0;LED2=1;LED3=0;LED4=1;LED5=0;LED6=1;LED7=0;LED8=1;}
if(S4==0){LED1=1;LED2=0;LED3=1;LED4=0;LED5=1;LED6=0;LED7=1;LED8=0;}

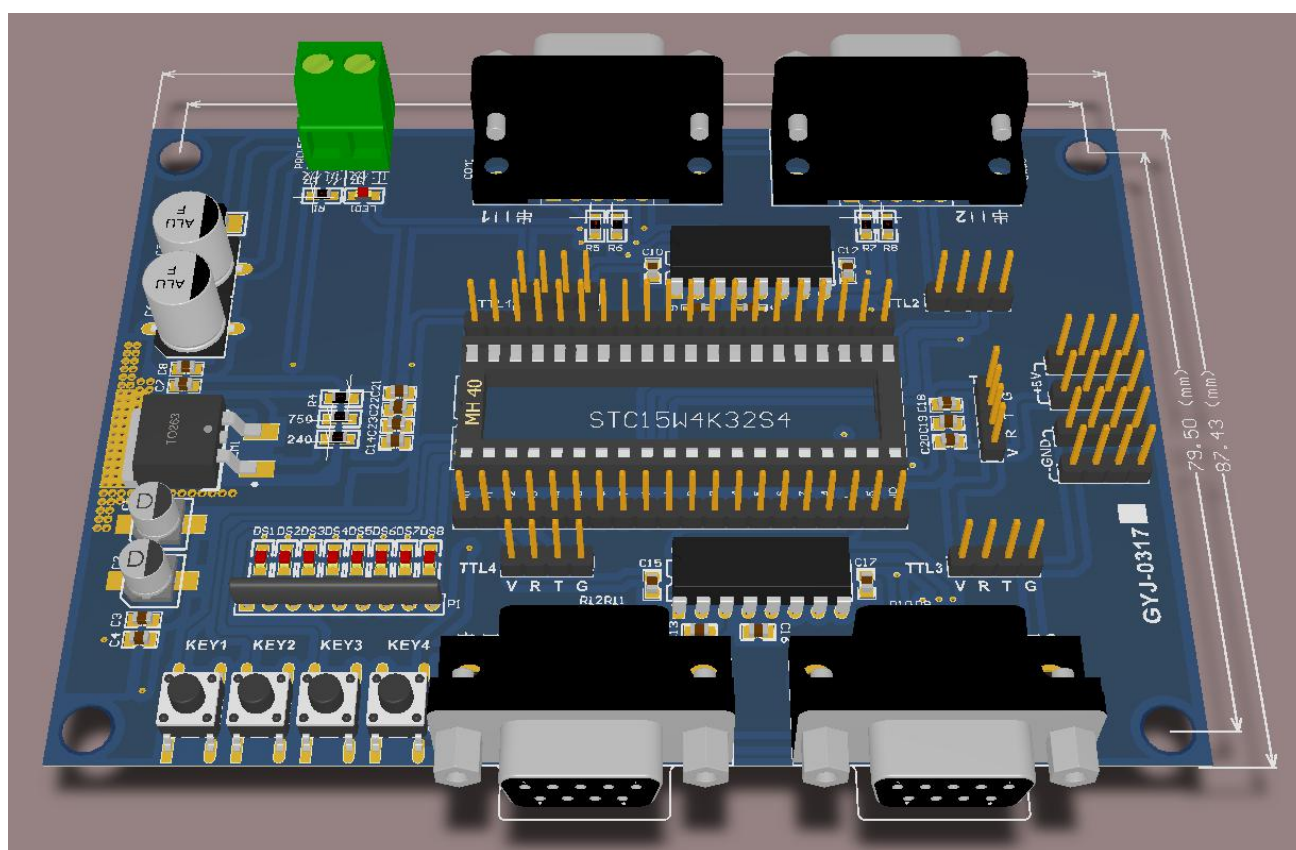
if(Hand1("UART1"))                                //串口 1 收到字符串 UART1
{
    ES = 0;                                          // 串口 1 中断关闭
    CLR_Buf1();                                     //将串口 1 缓存数组的值都清为零
    SendStringByUart1("UART1 CHECK OK!\r\n");       //串口 1 发送字符串 UART1 CHECK OK!
    ES = 1;                                          // 串口 1 中断打开
}
if(Hand2("UART2"))                                //串口 2 收到字符串 UART2
{
    IE2 &= 0xFE;                                     // 串口 2 中断关闭
    CLR_Buf2();                                     //将串口 2 缓存数组的值都清为零
    SendStringByUart2("UART2 CHECK OK!\r\n");       //串口 2 发送字符串 UART2 CHECK OK!
    IE2 |= 0x01;                                    // 串口 2 中断打开
}
if(Hand3("UART3"))                                // 收到打开 LED1 的指令
{
    IE2 &= 0xF7;                                     // 串口 3 中断关闭
    CLR_Buf3();                                     //将串口 3 缓存数组的值都清为零
    SendStringByUart3("UART3 CHECK OK!\r\n");       //串口 3 发送字符串 UART3 CHECK OK!
    IE2 |= 0x08;                                    // 串口 3 中断打开
}
if(Hand4("UART4"))                                // 收到打开 LED1 的指令
{
    IE2 &= 0xEF;                                     // 串口 4 中断关闭
    CLR_Buf4();                                     //将串口 4 缓存数组的值都清为零
    SendStringByUart4("UART4 CHECK OK!\r\n");       //串口 4 发送字符串 UART4 CHECK OK!
    IE2 |= 0x10;                                    // 串口 4 中断打开
}
}
}
}

```

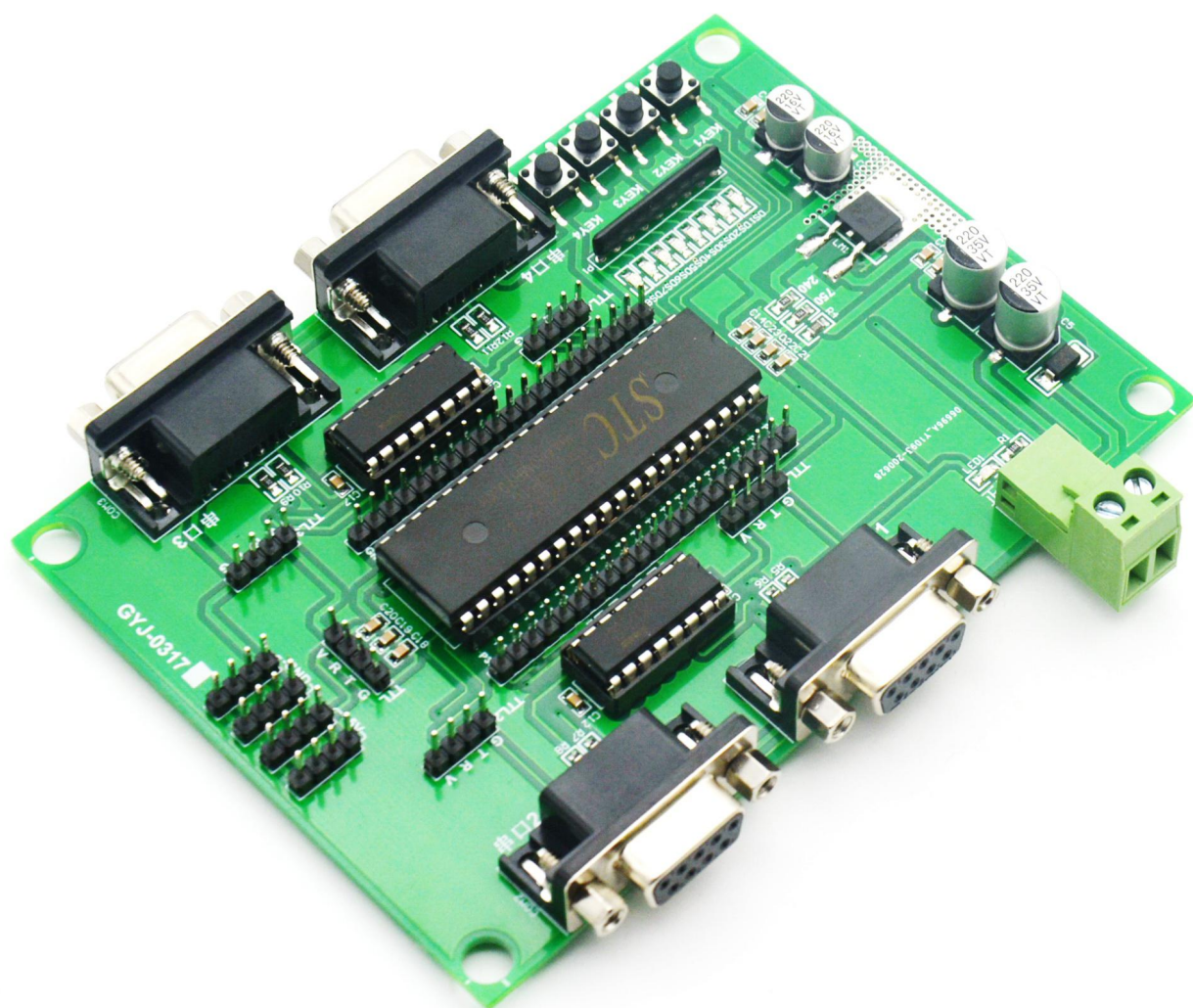
【尺寸图】提供高清 PDF 格式

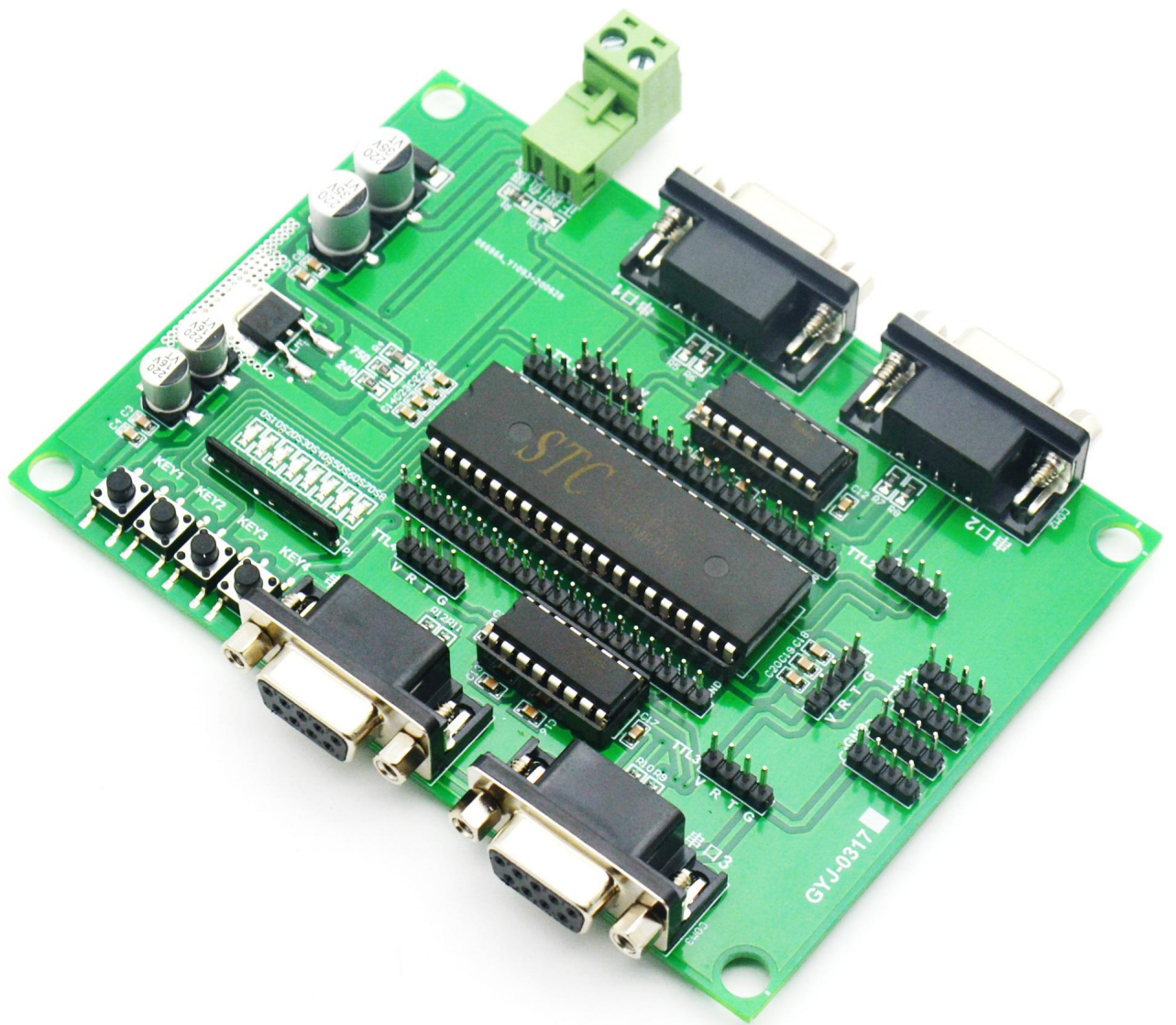


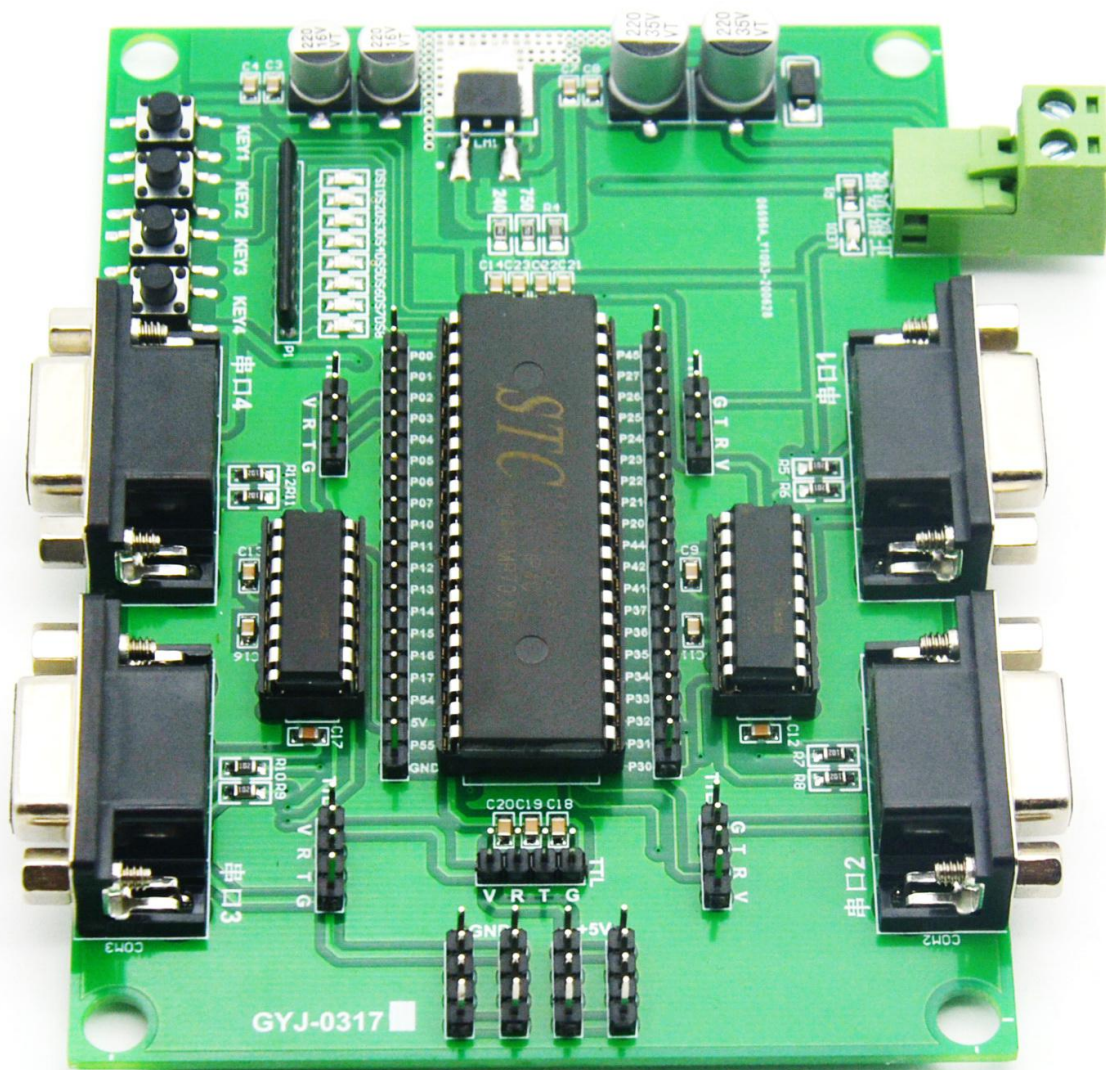


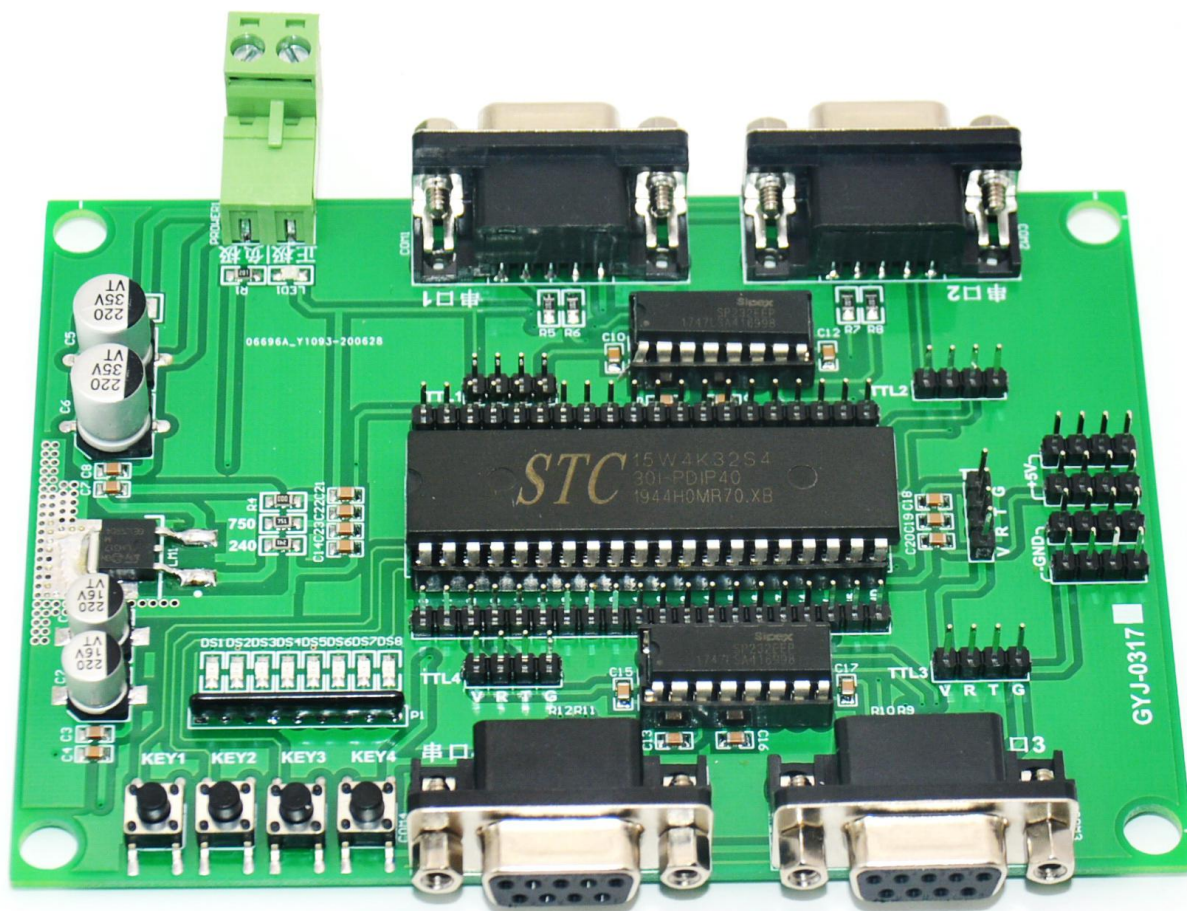


【图片展示】

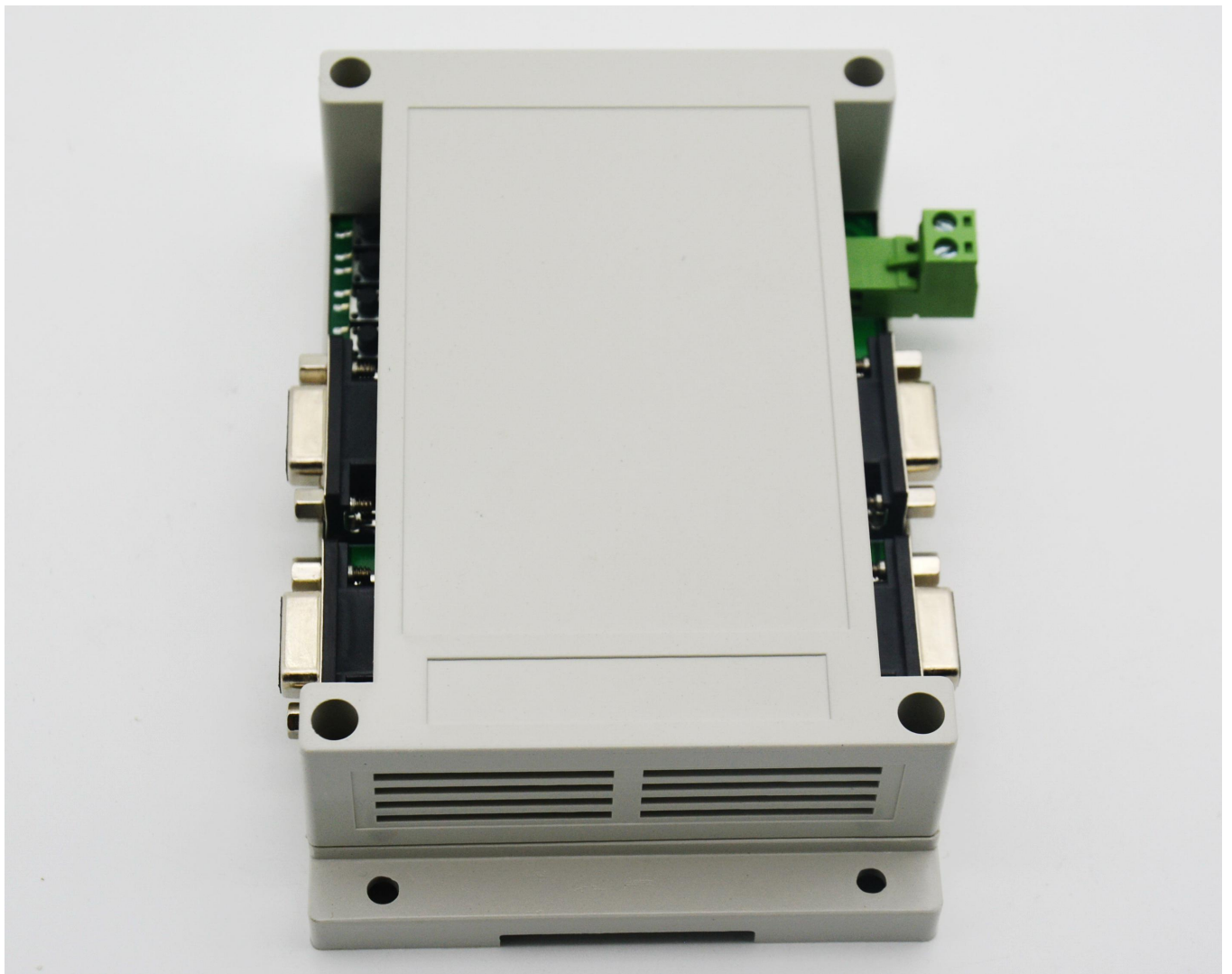


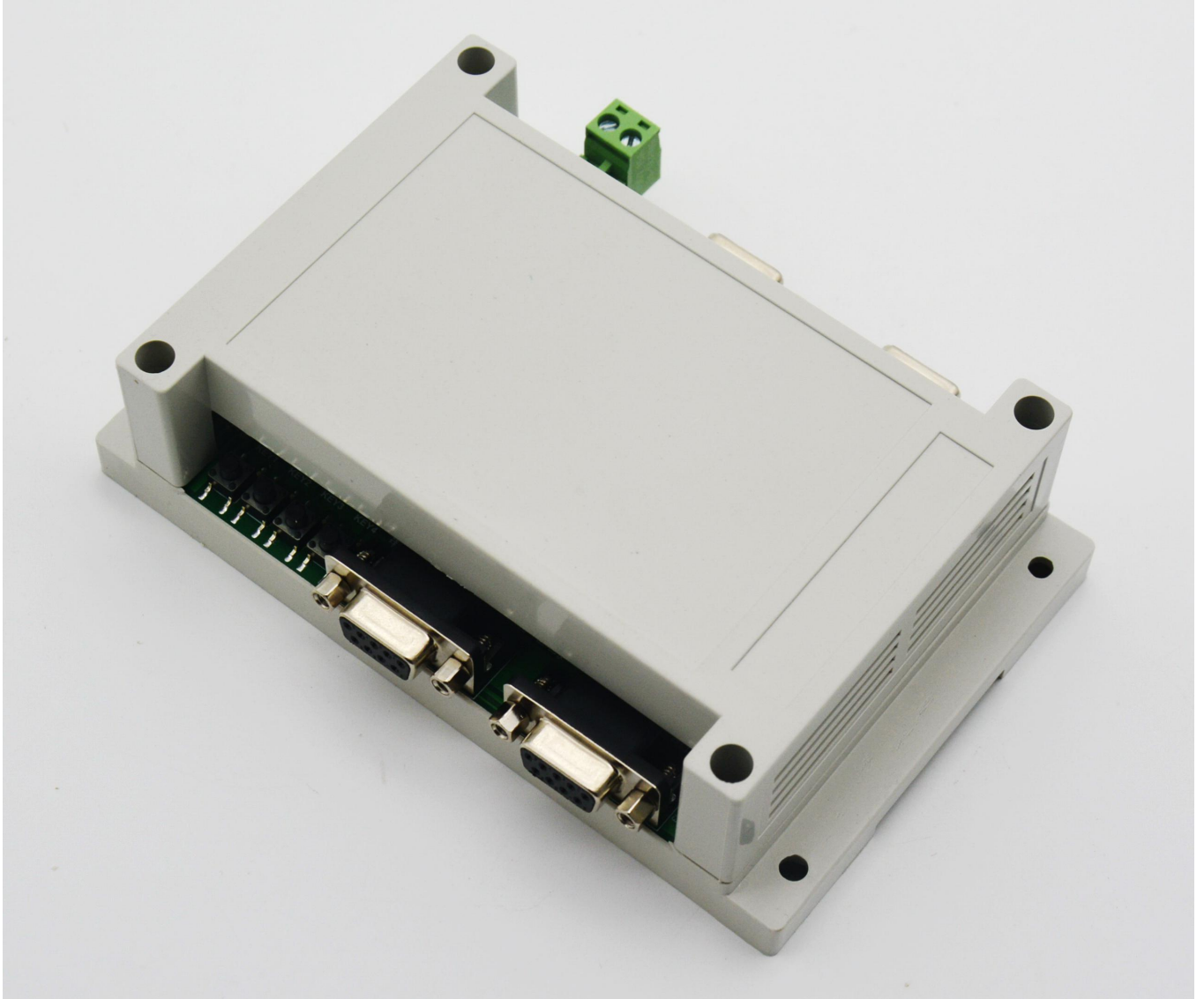






【帶壳图片展示】







在线销售产品技术支持联系信息：13603455408（微信同号）QQ：115451619

电路设计 项目定制 产品开发：15981910271（微信同号）

电子设计 自动化改造 产品升级：15003949398（微信同号）

产品有售淘宝 1店：<https://ourhc.taobao.com>

产品有售淘宝 2店：<https://g88888.taobao.com>

产品有售淘宝企业店：<https://shop404420384.taobao.com>

SMT 贴片加工 电子元件焊接 联系生产部：卢经理 电话 13503710441(微信同号)零元起步 不限数量，不限价格！