



SH79F9271 用户指南

-增强型 8051 微控制器



一、SH79F9271 I/O 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 IO 特性为：

- ◆ 26个双向I/O端口
- ◆ I/O端口可与其它功能共享
- ◆ 部分第二功能可通过寄存器配置
- ◆ 2个可选的开漏极I/O口
- ◆ 8个可选的 TTL 电平 I/O 口

SH79F9271 提供 26 个位可编程双向 I/O 端口。端口数据在寄存器 Px 中。每个 I/O 口均有内部上拉电阻。端口控制寄存器 (PxCRy) 控制端口是作为输入或者输出。当端口作为输入时，每个 I/O 端口带有由 PxPCRy 控制的内部上拉电阻 (x = 0-3, y = 0-7)。

2. 控制寄存器

IO 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	PxCRy (x=0-3, y=0-7)	IO 输入输出控制寄存器
	PxPCRy (x=0-3, y=0-7)	IO 上拉电阻控制寄存器
模块输出输入	Px.y (x=0-3, y=0-7)	IO 端口数据寄存器
数据指针选择	INSCON	特殊功能寄存器页选择

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. IO设置

3.1 IO输出设置

IO 作为输出模式，需要将其 PxCRy (x=0-3, y=0-7) 设置为 1，此时若设置 Px.y (x=0-3, y=0-7) 为 0，则 IO 输出低电平，低电平的值为 GND。若设置 Px.y (x=0-3, y=0-7) 为 1，则 IO 输出高电平，高电平的值为 VDD。当端口配置为输出低时，务必将内部上拉电阻配置为关闭状态，否则会有漏电。



3.2 IO输入设置

IO 作为输入模式，需要将其 PxCRy (x=0-3, y=0-7) 设置为 0，此时若设置 PxPCRy (x=0-3, y=0-7) 为 0，则 IO 处于输入 Floating 状态，若设置 PxPCRy (x=0-3, y=0-7) 为 1，则上拉电阻打开，IO 处于输入高的状态。IO 输入的高低之分是在 VDD/2 左右，具体参数请参考“SH79F9271 DATASHEET”。

3.3 开漏极I/O设置

SH79F9271 的 P0.7 和 P1.0 端口可以通过寄存器 POMS 设置为开漏或推挽输出，POMS.x(0-1) 设置为 0 时，为 N 沟道开漏输出；设置为 1 时，作为 COMS 推挽输出。

注意：P0.7, P1.0 端口作为 N-通道的开漏 I/O，此时端口电压不得超过 VDD_IN+0.3V。(POMS.0 为 P0.7, POMS.1 为 P1.0。)

4. 应用实例

4.1 要求

将一个 0~5V 的三角波从 P30 输入，P30 的值通过 P31 输出

4.2 例程

```
bit a;
void main(void)
{
    P3CR = 0xFE;           //P30 输入 P31 输出
    P3PCR |= 0x01;         //P30 上拉电阻打开
    while(1)
    {
        a = P3_0;
        P3_1 = a;          // P31 输出值为 P30 输入值
    }
}
```



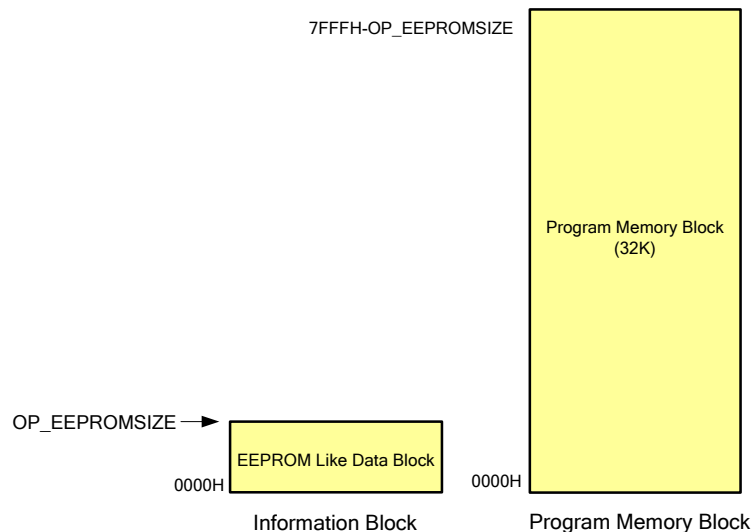
二、SH79F9271 Flash&EEPROM 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 Flash&EEPROM 特性为：

- ◆ Flash 存储器包括 $64 * 512B$ 区块，总共 32KB
- ◆ EEPROM 存储器 0~4KB 代码选项可选
- ◆ 在工作电压范围内都能进行编程和擦除操作
- ◆ 在线编程（ICP）操作支持写入、读取和擦除操作
- ◆ 支持整体/扇区擦除和编程（SSP）
- ◆ 编程/擦除次数：程序区：至少 10,000 次
类 EEPROM 区：至少 100,000 次
- ◆ 数据保存年限：至少 20 年
- ◆ 低功耗



SH79F9271 为存储程序代码内置 32K 可编程 Flash（Program Memory Block），可以通过在线编程（ICP）模式和扇区自编程（SSP）模式对 Flash 存储器操作。每个扇区 512 字节。

SH79F9271 还内置最大 4096 字节的类 EEPROM 存储区用于存放用户数据，每个扇区 512 字节，最多支持 8 个扇区。EEPROM 存储区位于 Flash 存储器，与程序存储区是共用的，举例说明：当 $OP_EEPROMSIZE=0000$ 时，即 EEPROM 大小为 4KB，此时程序存储区的大小为 $32KB-4KB=28KB$ ；当 $OP_EEPROMSIZE=0100$ 时，即 EEPROM 大小为 2KB，此时程序存储区的大小为 $32KB-2KB=30KB$ 。具体 EEPROM 大小选择详见在代码选项 章节。

2. 控制寄存器

Flash&EEPROM 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
----	------	------



模块控制	XPAGE	编程用地址选择寄存器
	IB_OFFSET	编程用地址偏移寄存器
	IB_DATA	编程用数据寄存器
	IB_CON1	SSP 型选择寄存器
	IB_CON2	SSP 流程控制寄存器 1
	IB_CON3	SSP 流程控制寄存器 2
	IB_CON4	SSP 流程控制寄存器 3
	IB_CON5	SSP 流程控制寄存器 4
	FLASHCON	访问控制寄存器

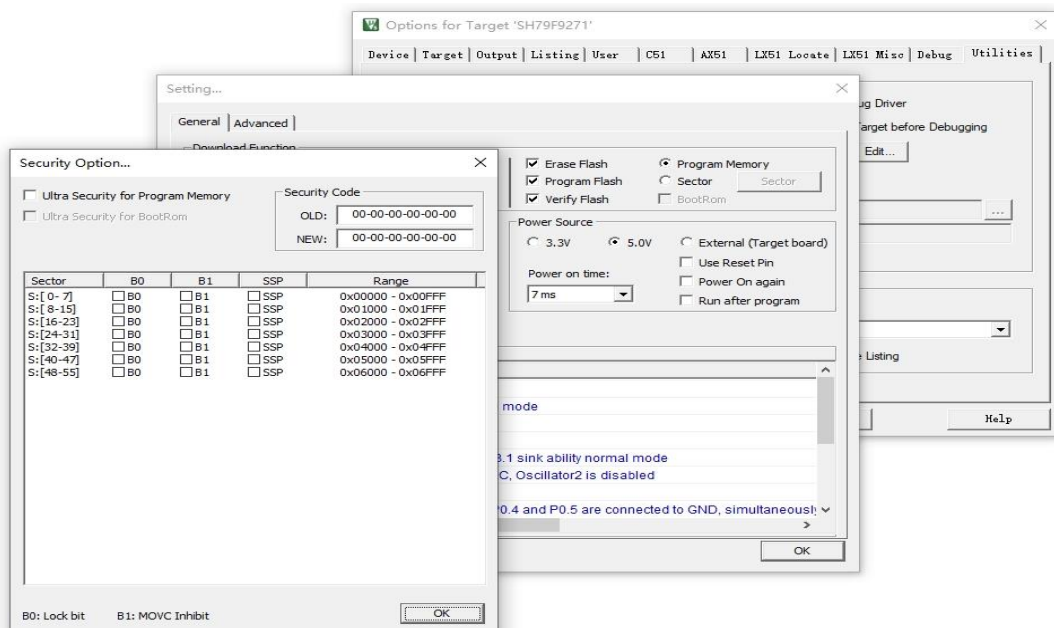
针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. SSP编程设置

3.1 Flash SSP 编程设置

若使用 SSP 功能对 Flash 进行擦除或者编写，首先需要将 FLASHCON 的 FAC 位置 0，然后设置相应的 XPAGE, IB_OFFSET, IB_DATA(需要操作的地址和数据)，再设置 IB_CON1，若 IB_CON1 为 0xE6，则相应的操作为扇区擦除，若 IB_CON1 为 0x6E，则相应的操作为存储单元编程。之后再设置 IB_CON2 为 0x05，IB_CON3 为 0x0A，IB_CON4 为 0x06，IB_CON5 为 0x09 完成 Flash 的整个操作。需要注意，SSP 扇区擦除操作不可选择 SSP 程序所在扇区作为操作对象。

特别地，若需要对特定扇区进行保护，防止 SSP 误操作，可在 Keil 中通过“Options->Utilities->Settings-> Security”路径设置 Sector 加密。其中，B0 加密对应 Flash 代码保护模式 0；B1 加密对应 Flash 代码保护模式 1。Flash 代码保护模式详细介绍，请参照“SH79F9271 DATASHEET”。

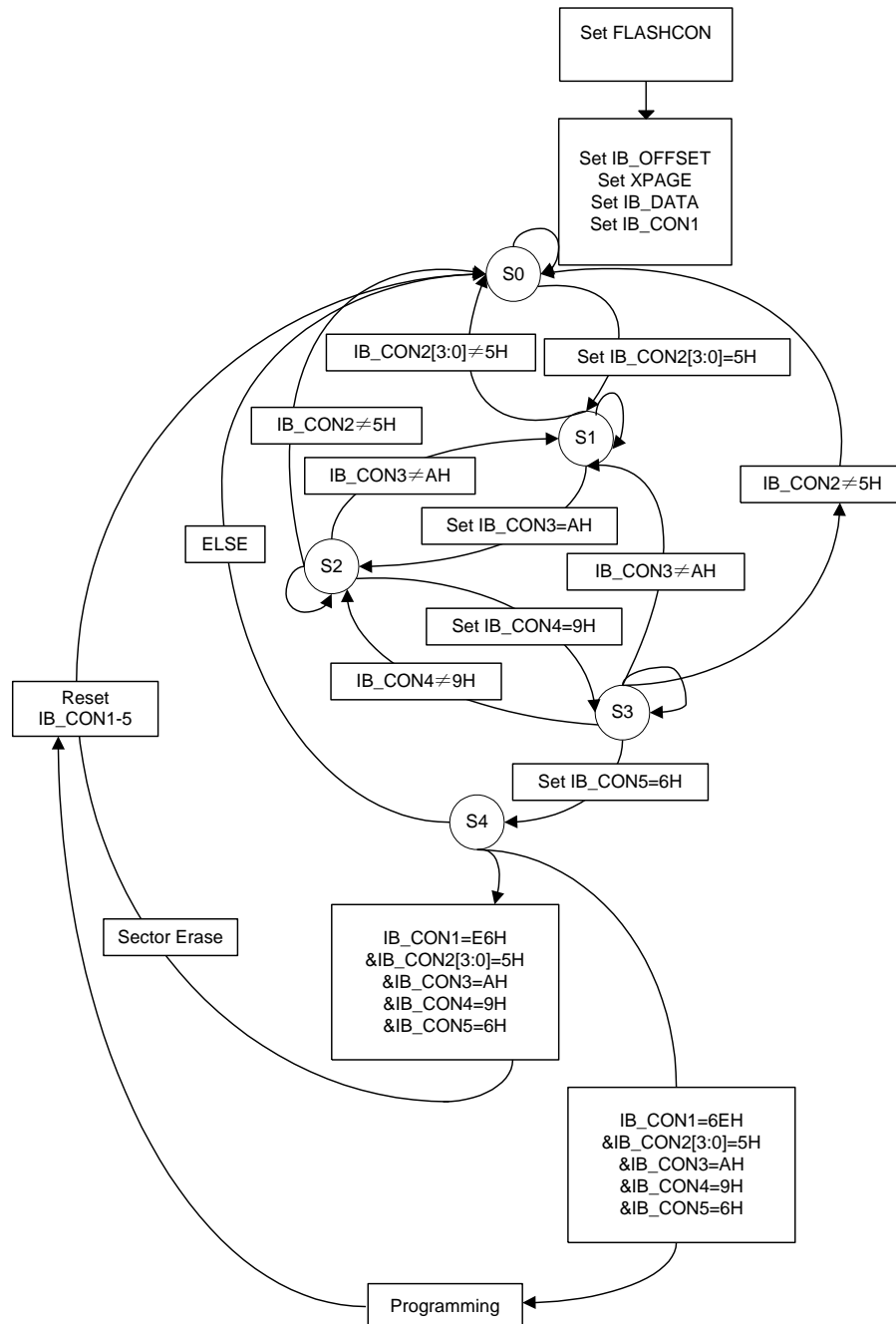




3.2 EEPROM SSP 编程设置

若使用 SSP 功能对 EEPROM 进行擦除或者编写，首先需要将 FLASHCON 的 FAC 位置 1，然后设置相应的 XPAGE，IB_OFFSET，IB_DATA（需要操作的地址和数据），再设置 IB_CON1，若 IB_CON1 为 0xE6，则相应的操作为扇区擦除，若 IB_CON1 为 0x6E，则相应的操作为存储单元编程。之后再设置 IB_CON2 为 0x05，IB_CON3 为 0x0A，IB_CON4 为 0x06，IB_CON5 为 0x09 完成 EEPROM 的整个操作。Flash 控制流程图如 3.3 所示。

3.3 Flash & EEPROM 控制流程图





3.4 应用实例

3.4.1 要求

先将整个 Flash 区域整体擦除，再 sector0 写入 00，sector1 写入 01，依次类推至 sector15 写入 0F。

3.4.2 例程

```
#include <SH79F9271.h>
#include <intrins.h>
#include <absacc.h>
#define uchar unsigned char
#define uint unsigned int
#define NOP _nop_();

uchar i,j,m;
uint n;
uchar count = 0;
uchar Ssp_flag;

void main (void)
{
    uchar temp;
    CLKCON = 0;
    m = 0;
    n = 0;
    Ssp_flag = 0xA5;
    /******sector erase*****/
    EA = 0;
    for(i=0x00;i<0x40;i++)
    {
        FLASHCON = FLASHCON&0xFE;
        NOP
        EA = 0;
        IB_CON1 = 0xE6;
        IB_CON2 = 0x05;
        IB_CON3 = 0x0A;
        IB_CON4 = 0x09;
        if(!(Ssp_flag==0xA5)) goto Error;
```



```
XPAGE = (i<<1)&0xFE;
IB_CON5 = 0x06;
NOP
NOP
NOP
NOP
NOP
}

/*****sector write*****/
temp = 0x00;
for(m=0x00;m<0x10;m++)
{
    for(n=0x00;n<512;n++)
    {
        temp = m;
        FLASHCON = FLASHCON&0xFE;
        NOP
        EA = 0;
        IB_OFFSET = n;
        XPAGE |= n>>8;
        XPAGE |= m<<1;
        IB_DATA = temp;
        IB_CON1 = 0x6E;
        IB_CON2 = 0x05;
        IB_CON3 = 0x0A;
        IB_CON4 = 0x09;
        if(!(Ssp_flag==0xA5)) goto Error;
        IB_CON5 = 0x06;
        NOP
        NOP
        NOP
        NOP
        NOP
        XPAGE = 0;
    }
}
while(1);
```




Error:

```
Ssp_flag = 0;
IB_CON1 = 0x00;
IB_CON2 = 0x00;
IB_CON3 = 0x00;
IB_CON4 = 0x00;
IB_CON5 = 0x00;
XPAGE = 0x00;
FLASHCON = 0x00;
EA = 0;
while(1);
}
```



4. 可读识别码

SH79F9271 每颗芯片出厂后都固化有一个 40 位的可读识别码，它的值为 0 - 0xffffffff 的随机值，它是无法擦除的（存放在地址信息存储区 0x127b - 127f），可以由程序或编程工具读出。

程序读出示例：

```
#include <SH79F9271.h>
#include <intrins.h>
#include <absacc.h>
unsigned char Temp1,Temp2,Temp3,Temp4,Temp5;
void main()
{
    FLASHCON = 0x01;
    Temp1 = CBYTE[0x127b];
    Temp2 = CBYTE[0x127c];
    Temp3 = CBYTE[0x127d];
    Temp4 = CBYTE[0x127e];
    Temp5 = CBYTE[0x127f];
    FLASHCON = 0x00;
    while(1);
}
```

5. 编程注意事项

为确保顺利完成SSP编程，用户软件必须按以下步骤设置：

(1) 用于代码/数据编程：

1. 关闭中断；
2. 根据地址设置 XPAGE，IB_OFFSET；
3. 按编程需要，设置 IB_DATA；
4. 按照顺序设置 IB_CON1 - 5；
5. 添加 4 个 NOP 指令；
6. 开始编程，CPU 将进入 IDLE 模式；烧写完成后自动退出 IDLE 模式；
7. 如需继续写入数据，跳转至第 2 步；
8. XPAGE 寄存器清 0，恢复中断设置。

(2) 用于扇区擦除：

1. 关闭中断；
2. 按相应的扇区设置 XPAGE；
3. 按照顺序设置 IB_CON1 - 5；
4. 添加 4 个 NOP 指令；
5. 开始擦除，CPU 将进入 IDLE 模式；擦除完成后自动退出 IDLE 模式；



6. 如需要继续擦除数据，跳转至第 2 步；

7. XPAGE 寄存器清 0，恢复中断设置。

(3) 读取：

使用“MOVC A,@A+DPTR”或者“MOVC A,@A+PC”指令。

(4) 对于类 EEPROM 区域

对于类 EEPROM 的操作类似于 Flash 的操作，即类似上述(1)/(2)/(3)部分的描述。区别在于：

1. 在对类 EEPROM 进行擦除、写或读之前，应首先将 FLASHCON 寄存器的最低位 FAC 位置 1。
2. 类 EEPROM 的扇区大小可以通过 option 选择。

注意：

1. 系统时钟不得低于 200kHz 以确保 FLASH 的正常编程
2. 当不需对类 EEPROM 操作时，必须将 FAC 位清 0

6. FLASH/类EEPROM 的烧写/擦除的超级抗干扰措施

- 1) 在程序下载时，选择代码选项中的“enable LVR function”。
- 2) 设置扇区时，对 XPAGE 写立即数。
- 3) 在调用烧写或擦除函数前，置一个标志，比如 0A5H；在烧写或擦除函数中判断此标志是否为 0A5H，否则清零此标志，退出函数。

下面类 EEPROM 示例仅供参考：

C 文件：

```
uchar Ssp_flag;
```

```
/******SSP Erase******/
```

```
Ssp_flag = 0xA5;
EA = 0;
FLASHCON = 0x01;
IB_CON1 = 0xE6;
IB_CON2 = 0x05;
IB_CON3 = 0x0A;
IB_CON4 = 0x09;
if(!(Ssp_flag==0xA5)) goto Error;
XPAGE = 0x01;      //XPAGE 写立即数
IB_CON5 = 0x06;
NOP
NOP
NOP
NOP
```

```
/******sector write******/
```



```
Ssp_flag = 0x5A;
EA = 0;
FLASHCON = 0x01;
IB_DATA = data;      //data 为烧写数据
IB_OFFSET = addr;    //addr 为烧写地址低位
IB_CON1 = 0x6E;
IB_CON2 = 0x05;
IB_CON3 = 0x0A;
IB_CON4 = 0x09;
if(!(Ssp_flag==0x5A)) goto Error;
XPAGE = 0x01;        //烧写地址高位，写立即数
IB_CON5 = 0x06;
NOP
NOP
NOP
NOP
while(1);
Error:
Ssp_flag = 0;
IB_CON1 = 0x00;
IB_CON2 = 0x00;
IB_CON3 = 0x00;
IB_CON4 = 0x00;
IB_CON5 = 0x00;
XPAGE = 0x00;
FLASHCON = 0x00;
EA = 0;
while(1);
```



三、SH79F9271 Timer3 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 Timer3 特性为：

- ◆ 定时器3是16位自动重载定时器，且可以工作在掉电模式

定时器 3 是 16 位自动重载定时器，通过两个数据寄存器 TH3 和 TL3 访问，由 T3CON 寄存器控制。IEN0 寄存器的 ET3 位置 1 允许定时器 3 中断（详见中断章节）。

定时器 3 只有一个工作方式：16 位自动重载计数器/定时器，可以设置预分频比，并可以工作在 CPU 掉电模式。

定时器 3 有一个 16 位计数器/定时器寄存器（TH3，TL3）。当 TH3 和 TL3 被写时，用作定时器重载寄存器，当被读时，被用做计数寄存器。TR3 位置 1 使定时器 3 开始递增计数。定时器在 0xFFFF 到 0x0000 溢出并置 TF3 位为 1。溢出同时，定时器重载寄存器的 16 位数据被重新载入计数寄存器中，TH3 写操作也导致重载寄存器的数据重新载入计数寄存器。

TH3 和 TL3 读写操作遵循以下顺序：

写操作：先低位后高位

读操作：先高位后低位

定时器 3 可以工作在掉电模式。

如果 T3CLKS[1:0] 为 00，定时器 3 不能工作在掉电模式下。如果 T3CLKS[1:0] 为 01，T3 端口输入外部时钟，定时器 3 可以工作在普通模式或掉电模式。当 T3CLKS[1:0] 为 10，定时器 3 可以工作在普通模式或掉电模式（在系统为高频 24MRC 时进入掉电模式）。

详见下表：

T3CLKS [1:0]	普通模式	掉电模式
00	工作	不工作
01	工作	工作
10	工作	工作

2. 控制寄存器

Timer3 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	T3CON	控制寄存器
	TL3	低位计数器
	TH3	高位计数器
数据指针选择	INSCON	特殊功能寄存器页选择

注意：



1. 在读或写 TH3 和 TL3 时，如果时钟源不是系统时钟，要确保 TR3=0。
2. 当定时器 3 用 T3 端口作为时钟源时，TR3 由 0 变为 1 之后的 1.5 个系统周期内，T3 的下降沿无效。
3. 定时器 3 相关的寄存器位于 Bank1，对 Timer3 的寄存器进行相关操作时请先将 INSCON 中的 BSK0 位置 1。

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. Timer3 设置

Timer3 只有一种工作方式：16 位自动重载计数器/定时器，且 Timer3 可以在掉电模式下计数，但是在掉电模式下计数对 Timer3 的工作时钟是有要求的，具体参见概述中的表格。

要使用 Timer3 需要设置好 Timer3 的工作时钟，然后设置工作时钟的预分频比，再写入符合自己要求的 TL3，TH3。然后启动 Timer3 即可。

注：Timer3 中断标志位由硬件清零

4. 应用实例

4.1 要求

Timer3 计数时钟选择 T3 口输入的时钟，T3 口输入的时钟频率为 2kHz，设置 Timer3 每一秒中唤醒 Power-Down 一次。

4.2 例程

```
#include <SH79F9271.H>
#include <intrins.H>
void Timer3_init(void)
{
    CLKCON = 0x00;    // 系统时钟设置为 24MHz
    INSCON |= 0x40;    // 切换至 BANK1
    T3CON = 0x01;      // T3 口作为 Timer3 的时钟输入口，该口上拉自动打开
    TL3 = 0x30;        // 定时 1 s 产生中断
    TH3 = 0xF8;
    INSCON &= 0xBF;    // 切换至 BANK0
}

void main(void)
{
    Timer3_init();
    EA = 1;
    IEN0 |= 0x20;      // 允许 Timer3 中断
    INSCON |= 0x40;    // 切换至 BANK1
    TR3 = 1;           // Timer3 开始计数
```



```
INSCON &= 0xBF;    // 切换至 BANK0
while(1)
{
    SUSLO = 0x55;
    PCON |= 0x02;    // 进入掉电模式
    _nop_();
    _nop_();
    _nop_();
    _nop_();
}

}

void timer3_int(void) interrupt 5
{
    _push_(INSCON);
    INSCON |= 0x40;    // 切换至 BANK1
    TF3 = 0;
    INSCON &= 0xBF;    // 切换至 BANK0
    _pop_(INSCON);
}
```

四、SH79F9271 Timer4 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 Timer4 特性为：

定时器 4 是 16 位自动重载定时器。通过两个数据寄存器 TH4 和 TL4 访问，由 T4CON 寄存器控制。IEN1 寄存器的 ET4 位置 1 允许定时器 4 中断（详见中断章节）。

当 TH4 和 TL4 被写时，用作定时器重载寄存器，当被读时，被用做计数寄存器。TR4 位置 1 使定时器 4 开始递增计数。定时器在 0xFFFF 到 0x0000 溢出并置 TF4 位为 1。溢出同时，定时器重载寄存器的 16 位数据重新载入计数寄存器中，对 TH4 的写操作也导致重载寄存器的数据重新载入计数寄存器。

TH4 和 TL4 读写操作遵循以下顺序：

写操作：先低位后高位

读操作：先高位后低位



2. 控制寄存器

Timer4 模块使用的所有控制寄存器如下表所示:

类别	缩写符号	功能说明
模块控制	T4CON	控制寄存器
	TL4	低位计数器
	TH4	高位计数器

针对每个寄存器的详细介绍, 请参照“SH79F9271 DATASHEET”

3. Timer4 设置

定时器 4 有 2 种工作方式: 16 位自动重载定时器, 和有 T4 边沿触发的 16 位自动重载定时器。这些方式通过 T4CON 寄存器的 T4M[1:0]设置。

4. 应用实例

4.1 要求

Timer4 计数时钟选择系统时钟, 设置每次进 Timer4 中断翻转一次 P31 口。

4.2 例程

```
#include <SH79F9271.H>
#include <intrins.H>
void Timer4_init(void)
{
    CLKCON = 0x00;    // 系统时钟设置为 24MHz
    INSCON |= 0x40;    // 切换至 BANK1
    T4CON    = 0x00;
    TL4      = 0xFF;
    TH4      = 0xFF;
    TR4      = 1;
    INSCON &= 0xBF;    // 切换至 BANK0
    IEN0     |= 0x04;
    EA       = 1;
}
void main(void)
{
    P3CR=0xFF;
    Timer4_init();
    while(1)
    {
    }
}
```




```
}  
  
void timer4_int(void) interrupt 2  
{  
    _push_(INSCON);  
    INSCON |= 0x40;    // 切换至 BANK1  
    TF4=0;  
    INSCON &= 0xBF;    // 切换至 BANK0  
    P3_1=~P3_1;  
    _pop_(INSCON);  
}
```

五、SH79F9271 Timer5 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 Timer5 特性为：

定时器 5 是 16 位自动重载定时器。通过两个数据寄存器 TH5 和 TL5 访问，由 T5CON 寄存器控制。IEN0 寄存器的 ET5 位置 1 允许定时器 5 中断（详见中断章节）。

当 TH5 和 TL5 被写时，用作定时器重载寄存器，当被读时，被用做计数寄存器。TR5 位置 1 使定时器 5 开始递增计数。定时器在 0xFFFF 到 0x0000 溢出并置 TF5 位为 1。溢出同时，定时器重载寄存器的 16 位数据重新载入计数寄存器中，对 TH5 的写操作也导致重载寄存器的数据重新载入计数寄存器。

TH5 和 TL5 读写操作遵循以下顺序：

写操作：先低位后高位

读操作：先高位后低位

2. 控制寄存器

Timer5 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	T5CON	控制寄存器
	TL5	低位计数器
	TH5	高位计数器
数据指针选择	INSCON	特殊功能寄存器页选择

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. Timer5 设置

定时器 5 有一种工作方式：16 位自动重载定时器。

定时器 5 在方式 0 为 16 位自动重载定时器。TH5 寄存器存放 16 位计数器/定时器高 8 位，TL5



存放低 8 位。当 16 位定时寄存器从 0xFFFF 到 0x0000 递增，并溢出时，系统置定时器溢出标志 TF5 (T5CON.7) 为 1，16 位寄存器的值被重新载入计数器，如果允许定时器 5 中断则产生中断。

T5CON.1 寄存器的 TR5 位置 1 允许定时器 5，且不清定时器 5 的计数器。在允许定时器 5 之前，将希望的初始值写入定时器重载寄存器中。

4. 应用实例

4.1 要求

Timer5 时钟选择系统时钟，设置每次进 Timer5 中断翻转一次 P32 口。

。

4.2 例程

```
#include <SH79F9271.H>
#include <intrins.H>
void Timer5_init(void)
{
    CLKCON = 0x00;    // 系统时钟设置为 24MHz
    INSCON |= 0x40;    // 切换至 BANK1
    T5CON    = 0x00;
    TL5      = 0xFF;
    TH5      = 0xFF;
    TR5      = 1;
    INSCON &= 0xBF;    // 切换至 BANK0
    IEN0     |= 0x08;
    EA       = 1;
}
void main(void)
{
    P3CR=0xFF;
    Timer5_init();
    while(1)
    {
    }
}
void timer5_int(void) interrupt 3
{
    _push_(INSCON);
    INSCON = 0x40;    // 切换至 BANK1
    TF5 = 0;
```



```
P3_2=~P3_2;
INSCON &= 0xBF; // 切换至 BANK0
_pop_(INSCON);
}
```

六、SH79F9271 EUART 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 EUART 模块特性为：

- ◆ SH79F9271 带有 2 个 EUART，兼容传统 8051
- ◆ EUART0/EUART1 自带波特率发生器，波特率可选择系统时钟分频或自带波特率发生器溢出率的 1/16
- ◆ 增强功能包括帧出错检测及自动地址识别
- ◆ EUART 有四种工作方式

2. 控制寄存器

EUART 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	SCON	EUART 控制及状态寄存器
	SBUF	EUART 数据寄存器
	PCON	电源控制寄存器
	SADDR	EUART 从属地址寄存器
	SADEN	EUART 地址掩码寄存器
	SFINE	波特率微调寄存器
	SBRTH/SBRTL	波特率发生器寄存器

注意：EUART1 相关寄存器在 BANK1 中，对其进行操作前，请先将 INSCON 中的 BSK0 位置 1。

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. EUART 设置

EUART 有 4 种工作方式。在通信之前用户必须先初始化 SCON，选择方式和波特率。

在所有四种方式中，任何将 SBUF 作为目标寄存器的写操作都会启动发送。在方式 0 中由条件 RI = 0 和 REN = 1 初始化接收。这会在 TxD 引脚上产生一个时钟信号，然后在 RxD 引脚上移 8 位数据。在其它方式中由输入的起始位初始化接收（如果 REN = 1）。通过发送起始位，外部发送器开始通信。



EUART 方式列表

SM0	SM1	方式	类型	波特率	帧长度	起始位	停止位	第 9 位
0	0	0	同步	$f_{\text{SYS}} / (4 \text{ 或 } 12)$	8 位	无	无	无
0	1	1	异步	自带波特率发生器的溢出率/16	10 位	1	1	无
1	0	2	异步	$f_{\text{SYS}} / (32 \text{ 或 } 64)$	11 位	1	1	0, 1
1	1	3	异步	自带波特率发生器的溢出率/16	11 位	1	1	0, 1

3.1 方式0设置

使用方式 0，需要先通过置 SM2 位（SCON.5）为 0 或 1，波特率固定为系统时钟的 1/12 或 1/4。当 SM2 位为 0 时，串行端口以系统时钟的 1/12 运行。当置 1 时，串行端口以系统时钟的 1/4 运行。

然后任何将 SBUF 作为目标寄存器的写操作都会启动发送。下一个系统时钟 Tx 控制块开始发送。数据转换发生在移位时钟的下降沿，移位寄存器的内容逐次从左往右移位，空位置 0。当移位寄存器中的所有 8 位都发送后，Tx 控制模块停止发送操作，然后在下一个系统时钟的上升沿将 TI 置 1（SCON.1）。

若接收到数据，则 REN（SCON.4）置 1 和 RI（SCON.0）清 0 初始化接收。下一个系统时钟启动接收，在移位时钟的上升沿锁存数据，接收转换寄存器的内容逐次向左移位。当所有 8 位都接收到接收移位寄存器中后，Rx 控制块停止接收，然后在下一个系统时钟的上升沿上 RI 置 1，直到被软件清 0 才允许接收。

3.2 方式1设置

方式 1 提供 10 位全双工异步通信，10 位由一个起始位（逻辑 0），8 个数据位（低位为第一位）和一个停止位（逻辑 1）组成。在接收时，这 8 个数据位存储在 SBUF 中而停止位储存在 RB8（SCON.2）中。方式 1 中的波特率固定为自带波特率发生器溢出率的 1/16。

任何将 SBUF 作为目标寄存器的写操作都会启动发送，实际上发送是从 16 分频计数器中的下一次跳变之后的系统时钟开始的，因此位时间与 16 分频计数器是同步的，与对 SBUF 的写操作不同步。起始位首先在 TXD 引脚上移出，然后是 8 位数据位。在发送移位寄存器中的所有 8 位数据都发送完后，停止位在 TXD 引脚上移出，在停止位发出的同时 TI 标志置位。

只有 REN 位置 1 时才允许接收。当 RxD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RxD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位，这有助于 16 分频计数器与 RxD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RxD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RxD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。8 个数据位和 1 个停止位移入之后，移位寄存器的内容被分别装入 SBUF 和 RB8 中，RI 置 1，但必须满足下列条件：

1. RI = 0
2. SM2 = 0 或者接收的停止位 = 1

如果这些条件被满足，那么停止位装入 RB8，8 个数据位装入 SBUF，RI 被置 1。否则接收的帧会丢失。这时，接收器将重新去探测 RxD 端是否另一个下降沿。用户必须用软件清除 RI，然后才能



再次接收。

3.3 方式2设置

方式 2 是使用异步全双工通信中的 11 位。一帧由一个起始位（逻辑 0），8 个数据位（低位为第一位），一个可编程的第 9 数据位和一个停止位（逻辑 1）组成。方式 2 支持多机通信和硬件地址识别（详见多机通信章节）。在数据传送时，第 9 数据位（SCON 中的 TB8）可以写 0 或 1，例如，可写入 PSW 中的奇偶位 P，或用作多机通信中的数据/地址标志位。当接收到数据时，第 9 数据位进入 RB8 而停止位不保存。PCON 中的 SMOD 位选择波特率为系统工作频率的 1/32 或 1/64。

任何将 SBUF 作为目标寄存器的写操作都会启动发送，同时也将 TB8 载入到发送移位寄存器的第 9 位中。实际上发送是从 16 分频计数器中的下一次跳变之后的系统时钟开始的，因此位时间与 16 分频计数器是同步的，与对 SBUF 的写操作不同步。起始位首先在 TXD 引脚上移出，然后是 9 位数据。在发送转换寄存器中的所有 9 位数据都发送完后，停止位在 TXD 引脚上移出，在停止位开始发送时 TI 标志置位。

只有 REN 位置 1 时才允许接收。当 RxD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RxD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位。这有助于 16 分频计数器与 RxD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RxD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RxD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。9 个数据位和 1 个停止位移入之后，移位寄存器的内容被分别装入 SBUF 和 RB8 中，RI 置 1，但必须满足下列条件：

1. RI = 0
2. SM2 = 0 或者接收的第 9 位 = 1，且接收的字节符合实际从机地址

如果这些条件被满足，那么第 9 位移入 RB8，8 位数据移入 SBUF，RI 被置 1。否则接收的数据帧会丢失。

在停止位的当中，接收器回到寻找 RxD 引脚上的另一个下降沿。用户必须用软件清除 RI，然后才能再次接收。

3.4 方式3设置

方式 3 使用方式 2 的传输协议以及方式 1 的波特率产生方式。

3.5 EUART 波特率设置

在方式 0 中，波特率可编程为系统时钟的 1/12 或 1/4，由 SM2 位决定。当 SM2 为 0 时，串行端口在系统时钟的 1/12 下运行。当 SM2 为 1 时，串行端口在系统时钟的 1/4 下运行。

在方式 1 和方式 3 中，使用自带波特率发生器，波特率可微调，精度为一个系统时钟，公式如下：

$$\text{BaudRate} = \frac{F_{\text{sys}}}{16 \times (32768 - \text{SBRT}) + \text{SFINE}}$$

例如：F_{sys} = 8MHz，需要得到 115200Hz 的波特率，SBRT 和 SFINE 值计算方法如下：



$$8000000/16/115200 = 4.34$$

$$SBRT = 32768 - 4 = 32764$$

$$115200 = 8000000/(16 \times 4 + SFINE)$$

$$SFINE = 5.4 \approx 5$$

此微调方式计算出的实际波特率为 115942，误差为 0.64%；

在方式 2 中，波特率固定为系统时钟的 1/32 或 1/64，由 SMOD 位（PCON.7）决定。当 SMOD 位为 0 时，EUART 以系统时钟的 1/64 运行。当 SMOD 位为 1 时，EUART 以系统时钟的 1/32 运行。

$$BaudRate = 2^{SMOD} \times \left(\frac{f_{SYS}}{64}\right)$$

4. 应用实例

4.1 要求

使用 EUART 进行串口通讯，通讯的波特率为 9600，1 位起始位，8 位数据位，1 位停止位，通讯内容为：收到一个数据后发送收到的数据出去。

4.2 例程

```
#include <SH79F9271.H>

unsigned char    r_data;
unsigned char    out_flag;

void uart0_init(void)
{
    CLKCON = 0x40;           // 系统时钟选择 6MHz
    SBRTH=0Xff;
    SBRTL=0xd9;
    SFINE=0x01;
    SCON = 0x50;             // 工作方式选择 Mode0
}

void main(void)
{
    uart0_init();
    r_data = 0x00;
    out_flag = 0;
    TI = 0;
    RI = 0;
    IEN0 = 0x90;
```



```
REN = 1;
while(1)
{
    if(out_flag == 1)
    {
        SBUF = r_data;           // 发送接收到的数据
        out_flag = 0;
    }
}

void uart0_int(void) interrupt 4
{
    if(TI == 1)
    {
        TI = 0;
    }
    else if(RI == 1)
    {
        r_data = SBUF;
        RI = 0;
        out_flag = 1;
    }
}
```

5. UART1

UART1 的用法同 UART0，需要注意的是 UART1 的相关寄存器在 BANK1。



七、SH79F9271 ADC 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 ADC 模块特性为：

- 12 位分辨率
- 最高转换速率可达 52Ksps
- 参考电压 VDD，ADC 内建的基准电压 Vref 直接与 Vdd 相连
- 13 路外接模拟输入

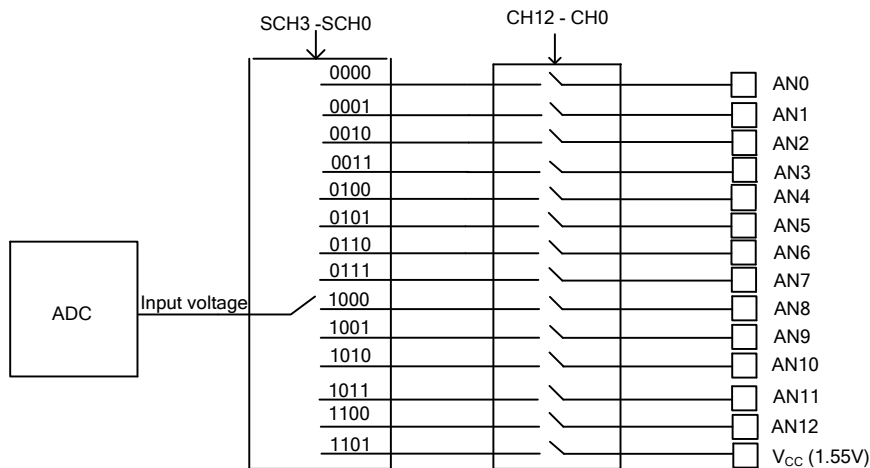
SH79F9271 包括一个单端型、12 位逐次逼近型模/数转换器 (ADC, Analog-to-Digit Converter)，模块如图所示。ADC 负责模拟信号到相应 12 位的数字信号的转换，当输入为 GND 时，输出为 0；当输入大于等于 Vref（所选基准电压）- 1LSB 时，输出最大值。ADC 的基准电压 VREF 与 VDD 直接相连，所以参考电压只能是 VDD。

该模块中有 13 路模拟输入 (CH0 - CH12, VBG)，通过配置通道寄存器，都可以进行转换。结果储存在对应的结果寄存器 ADDH, ADDL 中，每转换一次序列，结果寄存器的值更新一次。

对于单个通道，转换速率最高可达 52KSPS，可由寄存器设置 ADC 时钟速率以及采样时间。

ADC 模块能在 Idle 模式下工作，并且 ADC 中断能够唤醒 Idle 模式。在 Power-Down 模式下，ADC 模块不工作。

ADC 模块图如下：





2. 控制寄存器

ADC 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
	ADCON	ADC 控制寄存器 1
	ADT1	ADC 定时控制寄存器 1
	ADT2	ADC 定时控制寄存器 2
	ADCH1	设置 AD 信道配置寄存器 1
	ADCH2	设置 AD 信道配置寄存器 2
	ADDL	ADC 结果低位寄存器，x=0-3
	ADDH	ADC 结果高位寄存器，x=4-11

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. ADC设置

软件启动 ADC 转换步骤：

1. 选择模拟输入通道
2. 使能 ADC 模块
3. $\overline{\text{GO/DONE}}$ 置 1 开始 ADC 转换
4. 等待 $\overline{\text{GO/DONE}}$ 为 0 或者 ADCIF 为 1，如果 ADC 中断使能，则 ADC 中断将会产生，用户需要软件清 0 ADCIF
5. 从 ADDH/ADDL 获得转换数据
6. 重复步骤 3~5 开始另一次转换

4. 应用实例

4.1 要求

P34 作为 AN0，将 AN0 采集的 AD 值存在 ADC_Data 中。

4.2 例程

```
#include "SH79F9271.h"
#include "intrins.h"
#include "cpu.h"
unsigned int  ADC_Data;
void init_adc()
{
    ADCON = 0x80;    //打开 ADC  选择通道 AN0
    ADT1=0x21;       //ADC 采样转换时间 33*Tsyst
    ADT2=0x21;       //ADC 采样时间
    ADC1H = 0x01; //P34 作为 AN0
    ADC2H = 0;
```



```
While(ADCON & 0X01) ;//等待 ADC 转换完成
ADC_Data=ADDH;
ADC_Data=(ADD_Data<<4)+ADDL;//获取 AD 值
}
```



八、SH79F9271 LPD 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。SH79F9271 其 LPD 模块特性为：

- ◆ 低电压检测并产生中断
- ◆ 可选的LPD检测电压
- ◆ LPD检测含双向去抖功能

低电压检测（LPD）功能用来监测电源电压，如果电压低于指定值时产生内部标志。LPD 功能用来通知 CPU 电源是否被切断或电池是否用尽，因此在电压低于最小工作电压之前，软件可以采取一些保护措施。

LPD 中断可以唤醒 Power-down 模式。

2. 控制寄存器

LPD 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	LPDCON	控制寄存器
档位选择	LPDSEL	电压检测档位控制寄存器

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. LPD设置

低电压检测主要应用于当系统电压低于某个值的时候，系统认为存在断电的风险，需要做一些保护措施，来防止短时间的掉电，如果在一定时间内，系统重新上电，可以继续掉电前的状态往下运行。使用 LPD 时，需要先设置检测模式和一个合适的电压值，SH79F9271 提供 2 种检测模式和 13 档检测电压，可以通过 LPDCON 和 LPDSEL 来选择，设置完毕，使能 LPD 模块，可以通过查询 LPDF 或者使能 LPD 中断来快速检测掉电。

4. 应用实例

4.1. 要求

系统工作电压为 5V，当电压小于 3.6V 时系统认为存在掉电风险，若 Vdd 电压小于 3.6V，则将关键参数存于 C0H~CFH 中，然后系统进入 Power-Down 模式。

4.2. 例程

```
#include <SH79F9271.H>
#include <intrins.h>
unsigned char outflag = 0;
idata unsigned char temp0 _at_ 0xC0;
idata unsigned char temp1 _at_ 0xC1;
idata unsigned char temp2 _at_ 0xC2;
```



```
idata unsigned char temp3 _at_ 0xC3;
idata unsigned char temp4 _at_ 0xC4;
idata unsigned char temp5 _at_ 0xC5;
idata unsigned char temp6 _at_ 0xC6;
idata unsigned char temp7 _at_ 0xC7;
idata unsigned char temp8 _at_ 0xC8;
idata unsigned char temp9 _at_ 0xC9;
idata unsigned char temp10 _at_ 0xCA;
idata unsigned char temp11 _at_ 0xCB;
idata unsigned char temp12 _at_ 0xCC;
idata unsigned char temp13 _at_ 0xCD;
idata unsigned char temp14 _at_ 0xCE;
idata unsigned char temp15 _at_ 0xCF;
unsigned char keydata0;
unsigned char keydata1;
unsigned char keydata2;
unsigned char keydata3;
unsigned char keydata4;
unsigned char keydata5;
unsigned char keydata6;
unsigned char keydata7;
unsigned char keydata8;
unsigned char keydata9;
unsigned char keydata10;
unsigned char keydata11;
unsigned char keydata12;
unsigned char keydata13;
unsigned char keydata14;
unsigned char keydata15;
void main(void)
{
    CLKCON = 0x00;           //系统时钟 = 24MHz
    LPDCON = 0x80;           //使能 LPD 模块
    LPDSEL = 0x08;           //设置 Vlpd =3.6V

    Delay();                 //延时 20us
    LPDCON &= 0xef;          //清标志位
```



```
EA = 1;                //开启总中断
IEN1 |= 0x40;          //允许 LPD 中断
while(1)
{
    if(outflag == 1)    //LPD 发生
    {
        temp0 = keydata0;
        temp1 = keydata1;
        temp2 = keydata2;
        temp3 = keydata3;
        temp4 = keydata4;
        temp5 = keydata5;
        temp6 = keydata6;
        temp7 = keydata7;
        temp8 = keydata8;
        temp9 = keydata9;
        temp10 = keydata10;
        temp11 = keydata11;
        temp12 = keydata12;
        temp13 = keydata13;
        temp14 = keydata14;
        temp15 = keydata15;
        LPDCON &= 0x7F;    //关闭 LPD 模块
        SUSLO = 0x55;
        PCON |= 0x02;
        _nop_();
        _nop_();
        _nop_();
    }
}

void LPD_int(void) interrupt 13
{
    LPDCON &= 0xEF;
    outflag = 1;
}
```



九、SH79F9271 PWM 用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

SH79F9271 其 PWM 驱动器特性为：

- ◆ 两路12位精度PWM模块
- ◆ 提供每个PWM周期溢出中断
- ◆ 输出极性可选择

SH79F9271 集成了两路 12 位 PWM 模块。PWM 模块可以产生周期和占空比分别可调整的脉宽调制波形。PWMxCON (x = 0-1) 寄存器用于控制 PWMx 模块的时钟、波形输出以及周期中断，PWMxPH/L 寄存器用于控制 PWMx 输出波形的周期，PWMxDH/L (x = 0-1) 寄存器用于控制 PWMx 模块输出波形的占空比。

在 PWM 输出允许期间，可以修改这三个寄存器，但在下一个 PWM 周期修改才会起作用。

2. 控制寄存器

PWM0 模块使用的所有控制寄存器如下表所示：

类别	缩写符号	功能说明
模块控制	PWM0CON	控制寄存器
	PWM0PH	周期控制寄存器高 4 位
	PWM0PL	周期控制寄存器低 8 位
	PWM0DL	占空比控制寄存器低 8 位
	PWM0DH	占空比控制寄存器高 4 位

针对每个寄存器的详细介绍，请参照“SH79F9271 DATASHEET”

3. PWM0 设置

PWM 编程分为以下几步：

- (1) 选择 PWM 模块时钟源。
- (2) 通过写适当的值到 PWM 周期控制寄存器 (PWMxPH/L) 或 PWM 占空比寄存器 (PWMxDH/L) 设置 PWM 周期/占空比，先设置低位，再设置高位。注意，即使高位数值不变，也要重写一次，否则，低位的修改无效。
- (3) 通过设置 PWM 控制寄存器 (PWMxCON) 的 PWMxS 位选择 PWMx 输出模式(高电平有效或低电平有效)。
- (4) 如果 PWM 周期或者占空比需要改变，操作流程如同步骤 2 或者步骤 3 说明。修改后的重载计数器的值在下一个周期开始有效。



4. 应用实例

4.1 要求

使用 PWM0，产生一个 2KHz，占空比为 50%的 PWM 波驱动蜂鸣器。

4.2 例程

```
#include <SH79F9271.H>
#include <intrins.h>
void PWM0_init(void)
{
    CLKCON = 0x00;          //系统时钟 = 24MHz
    PWM0CON = 0x29;          //时钟为系统时钟/32,占空比期间输出高，输出允许
    PWM0PL = 0x77;
    PWM0PH = 0x01;          //PWM0 频率为 2KHz
    PWM0DL = 0xBB;
    PWM0DH = 0x00;          //PWM0 占空比为 50%
}
void main(void)
{
    PWM0_init();
    PWM0CON |= 0x80;         //使能 PWM0
    while(1);
}
```

PWM1 用法同 PWM0.



十、SH79F9271 LCM用户指南

1. 概述

SH79F9271 是一种高速高效率 8051 可兼容单片机。在同样振荡频率下，较之传统的 8051 芯片它有着运行更快速的优越特性。

- 5 种数字逻辑功能口可以通过数字逻辑可配置模块重新映像到 I/O,且每种功能可在 8 个 IO 口中选择其一。

2. 控制寄存器

LCM 模块所用的寄存器如下表所示：

类别	缩写符号	功能说明
引脚配置寄存器	UART1CR	TXD1 和 RXD1 配置寄存器
	PWMCR	PWM0 和 PWM1 配置寄存器
	INTCR	INT2 配置寄存器

3. 应用实例

3.1 要求

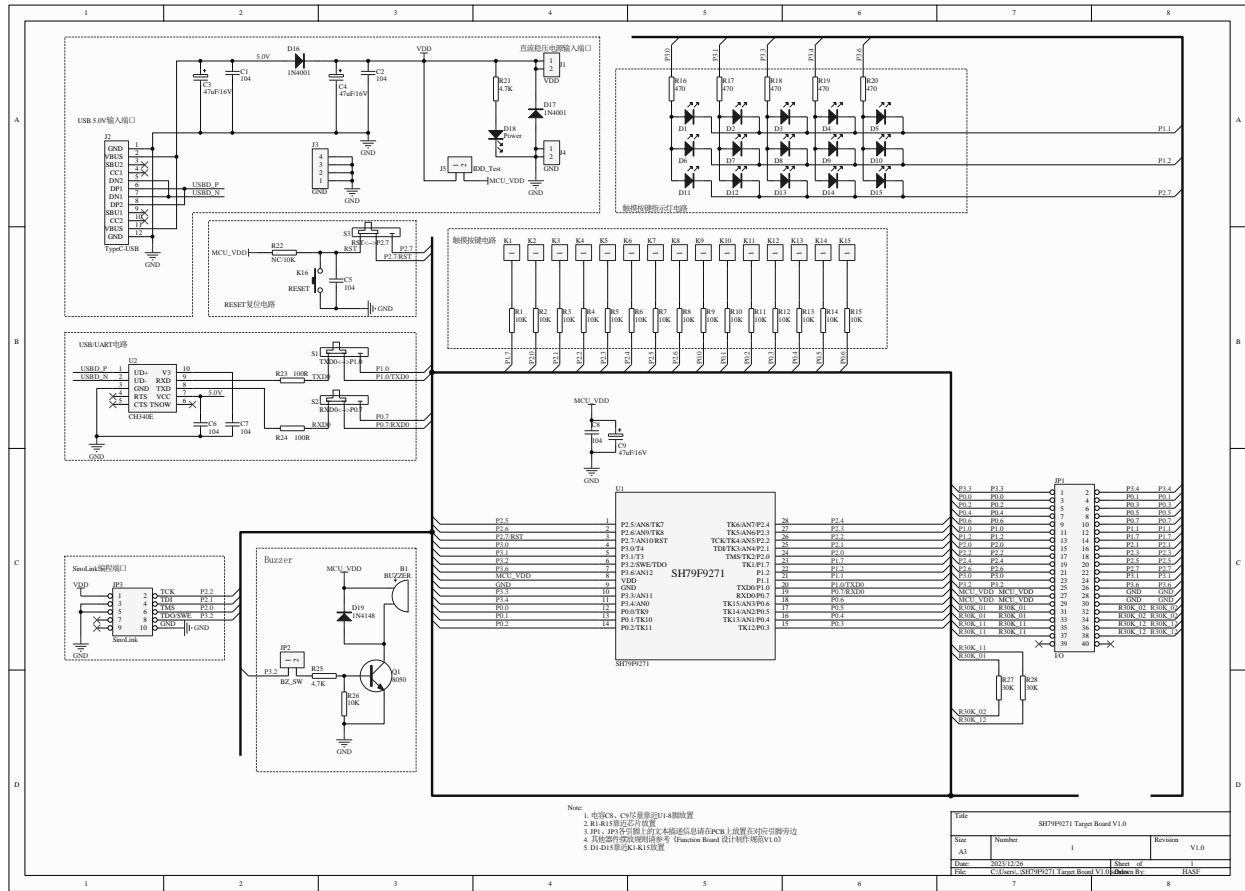
引脚配置为 P0.3 为 PWM1，P1.1 为 PWM0，P0.6 为 TXD1，P0.5 为 RXD1。

3.2 例程

```
#include <SH79F9271.H>
#include "intrins.h"
IO_config()
{
    UART1CR = 0x10;    // P0.6 为 TXD1, P0.5 为 RXD1
    PWMCR = 0x12;      // P0.3 为 PWM1, P1.1 为 PWM0
}
```



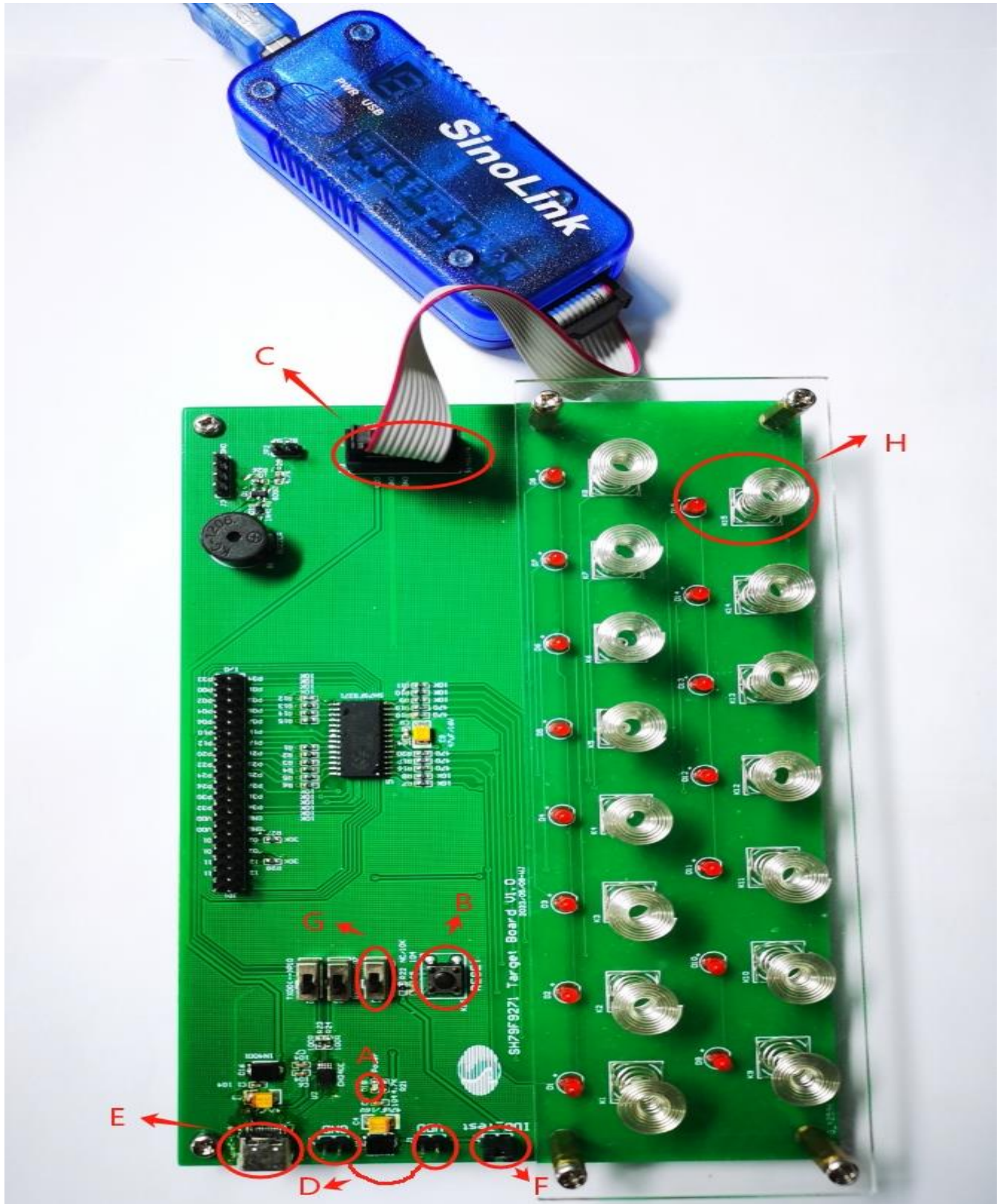

附件一：SH79F9271 Target Board V1.0原理图





附件二：SH79F9271演示板（Target Board）

SH79F9271 提供了一套演示板，供客户熟悉芯片功能。如下图：





板中各部分功能说明：

A. 电源指示灯

B. 复位按键

C. JTAG 仿真接口

JTAG 仿真接口和 SinoLink 仿真器一一对应，可用 10 芯扁平线直接连接。演示板的供电电源可通过仿真接口的 VDD 管脚获得。

D. 外部供电接口 1（VDD/GND）

Target Board 板供电电源接口，分别与芯片的 VDD 和 GND 管脚相连，采用外部供电接口 1 方式供电，芯片供电电压可调。

E. 外部供电接口 2（USB5.0V 输入）

USB 5V 输入端口，同时通过 CH340 串口通信芯片与 SH79F9271 串口通信脚连接，可进行串口数据传输。

注：演示板的供电可以由外部供电接口 1 或外部供电接口 2 或仿真接口提供，不管使用哪一种供电方式，外部供电接口 1 处的电压为演示板的供电电压。

F. 电流测试接口。（不进行电流测试，请用短路跳线连接）

G. 拨码开关（RST PIN）

P2.7 作为 PIN RST 时，拨码开关拨左和板子上的 B 部分组成 RST 电路，B 部分的按钮被按下一次就会发生一次 RESET，P2.7 作为 IO 时，拨码开关拨右，和外部的 RST 电路断开。

H. 触摸按键和触摸指示灯

SH79F9271 Target Board 包含 15 个触摸按键及触摸按键指示灯，可进行简单触摸调试。（触摸效果请结合实际产品进行调试）



规格更改记录

版本	记录	日期
1.0	初始版本	2024 年 3 月



目录

SH79F9271 用户指南	1
一、SH79F9271 I/O 用户指南	2
1. 概述.....	2
2. 控制寄存器.....	2
3. IO 设置	2
3.1 IO 输出设置	2
3.2 IO 输入设置	3
3.3 开漏极 I/O 设置	3
4. 应用实例.....	3
4.1 要求.....	3
4.2 例程.....	3
二、SH79F9271 Flash&EEPROM 用户指南	4
1. 概述.....	4
2. 控制寄存器.....	4
3. SSP 编程设置	5
3.1 Flash SSP 编程设置	5
3.2 EEPROM SSP 编程设置.....	6
3.3 Flash & EEPROM 控制流程图.....	6
3.4 应用实例.....	7



4. 可读识别码.....	10
5. 编程注意事项.....	10
6. FLASH/类 EEPROM 的烧写/擦除的超级抗干扰措施.....	11
三、SH79F9271 Timer3 用户指南.....	13
1. 概述.....	13
2. 控制寄存器.....	13
3. Timer3 设置.....	14
4. 应用实例.....	14
4.1 要求.....	14
4.2 例程.....	14
四、SH79F9271 Timer4 用户指南.....	15
1. 概述.....	15
2. 控制寄存器.....	16
3. Timer4 设置.....	16
4. 应用实例.....	16
4.1 要求.....	16
4.2 例程.....	16
五、SH79F9271 Timer5 用户指南.....	17
1. 概述.....	17
2. 控制寄存器.....	17



3. Timer5 设置	17
4. 应用实例	18
4.1 要求	18
4.2 例程	18
六、SH79F9271 EUART 用户指南	19
1. 概述	19
2. 控制寄存器	19
3. EUART 设置	19
3.1 方式 0 设置	20
3.2 方式 1 设置	20
3.3 方式 2 设置	21
3.4 方式 3 设置	21
3.5 EUART 波特率设置	21
4. 应用实例	22
4.1 要求	22
4.2 例程	22
5. UART1	23
七、SH79F9271 ADC 用户指南	24
1. 概述	24
2. 控制寄存器	25



3. ADC 设置	25
4. 应用实例.....	25
4.1 要求.....	25
4.2 例程.....	25
八、SH79F9271 LPD 用户指南.....	27
1. 概述.....	27
2. 控制寄存器.....	27
3. LPD 设置	27
4. 应用实例.....	27
4.1. 要求.....	27
4.2. 例程.....	27
九、SH79F9271 PWM 用户指南.....	30
1. 概述.....	30
2. 控制寄存器.....	30
3. PWM0 设置	30
4. 应用实例.....	31
4.1 要求.....	31
4.2 例程.....	31
十、SH79F9271 LCM 用户指南.....	32
1. 概述.....	32



2. 控制寄存器.....	32
3. 应用实例.....	32
3.1 要求.....	32
3.2 例程.....	32
附件一：SH79F9271 Target Board V1.0 原理图.....	33
附件二：SH79F9271 演示板（Target Board）	34
规格更改记录.....	36
目录.....	37