SC9351_AN01

Silan 杭州士兰微电子股份有限公司

SC9351用户手册

士兰保留说明书的更改权，恕不另行通知！产品提升永无止境，我公司将竭诚为客户提供更优秀的产品！

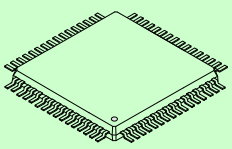
地址：杭州市黄姑山路4号
电话：+86-0571-88210880
传真：+86-0571-88211612

邮编：310012
Http：//www.silan.com.cn
Email：silan@silan.com.cn

目　　录

1. 概述

SC9351 是一款采用 8051 架构的 MCU 产品。内置 64K Byte FLASH、8KByte RAM。同时提供丰富的片内外设模块，如 I²C、UART、SPI、ADC、RTC 等。



LQFP-64-10x10-0.5

2. 主要特点

- 在系统编程（ISP）
- 2.4-3.6V 供电，集成 LDO 模块，给芯片内核供电；也可外接 LDO；
- 双时钟　75KHz /12MHz OSC
- 采用 8051 架构，速度提高 4 倍；兼容 8051 指令集；2～4 个时钟指令周期
- 内嵌 64Kx8 的 FLASH，可作为程序存储器或数据存储器使用；可通过片上程序直接对 FLASH 进行编程；也可由烧录器直接烧录；
- 数据存储器 IDATA： 256Byte(兼容 8051) + 64Byte(内核掉电保存数据)
- XDATA： 8KByte 外部数据存储器，其中低 4K 可以作为程序存储器使用，以支持对 FLASH 的编程。
- 集成 RTC 模块，具有日历、时钟、闰年自动识别切换功能，提供定时闹钟功能；具有时钟调整功能；
- 灵活的中断系统,4 路外部中断,共支持 18 个中断源
- 集成 2 路 UART 接口；
- 集成 1 路 SPI 接口；
- 集成 3 通道 8Bits 的 A/D；
- 集成 1 路 I²C 接口；
- 支持多种低功耗工作模式：低频工作模式、Sleep 模式、掉电保持模式。

3. 应用

- 台式音响
- 汽车音响

4. 产品规格分类

产品型号	封装形式	打印名称
SC9351	LQFP-64-10 x 10-0.5	SC9351

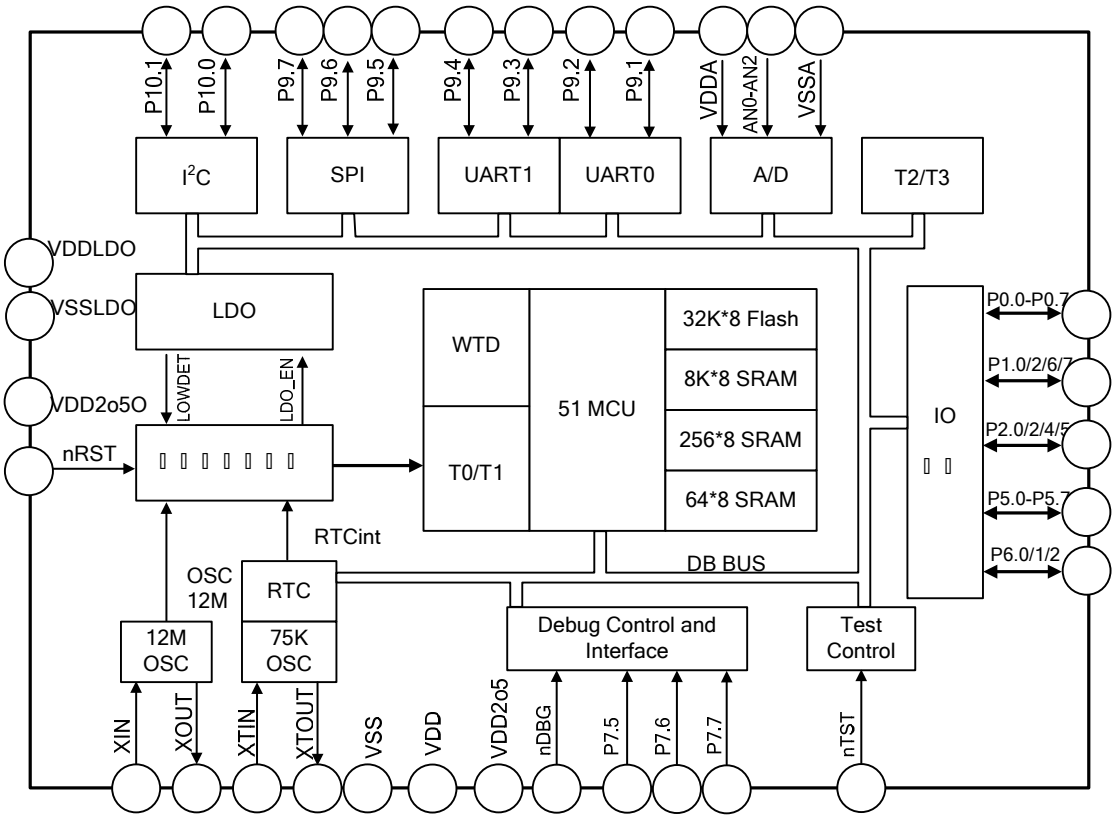
资源列表

产品	Timer	ADC 通道	SPI	UART	I²C	IO 数目[注1]	外部中断
SC9351	4	3	1	2	1	40	4

注：

1. P7 口 三个管脚与调试口复用。

5. 内部框图



6. 参数说明

6.1. 极限参数

参　　数	符 号	参 数 范 围	单位

电源电压	VDD	-0.3~+5.0	V
输入电压	VIN	-0.3~VDD+0.3	V
存储温度	TSTG	-65~+150	°C
工作温度	TOPR	-40~+85	°C
ESD	Vesd	2	KV

6.2. 电气参数（除非特殊说明，VDD=3.3V，Tamb=25°C）

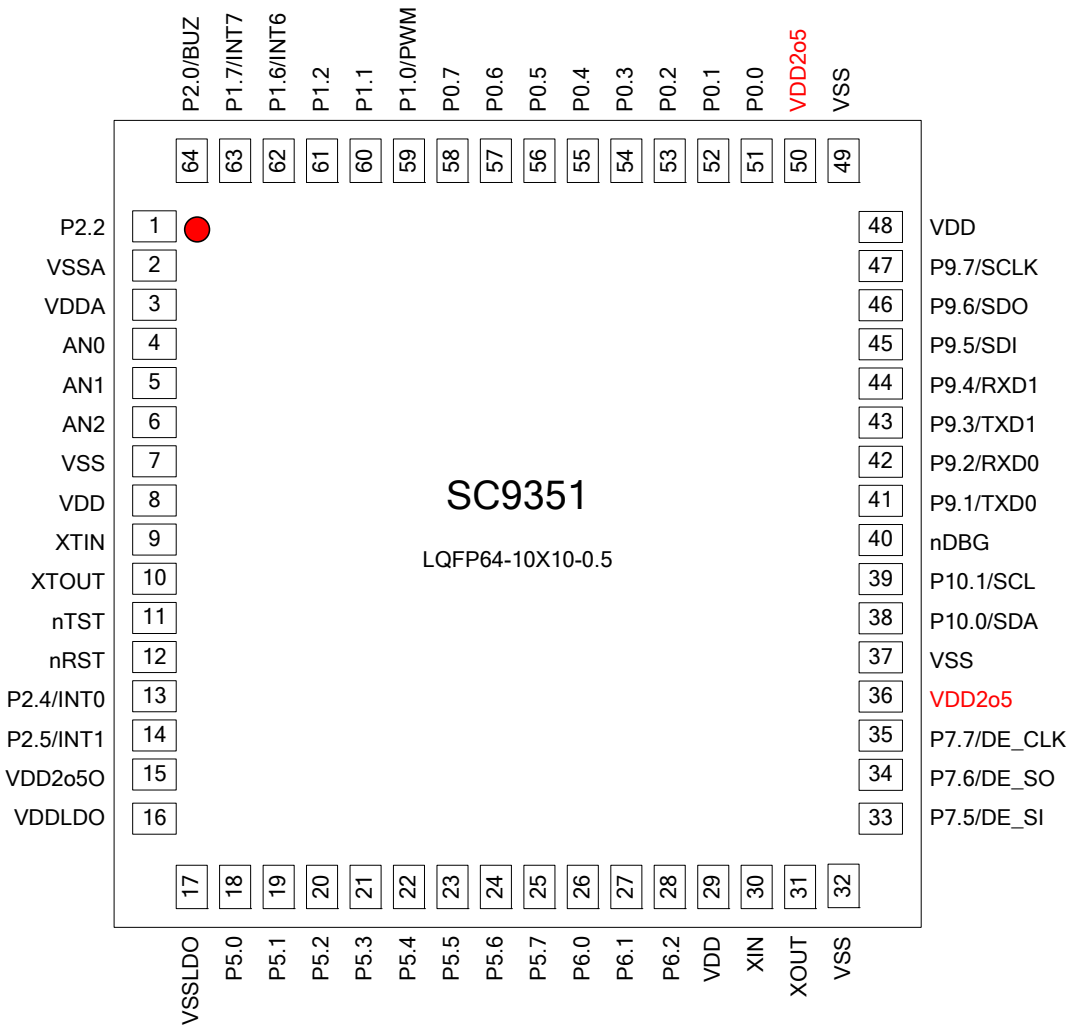
参　　数	符 号	测 试 条 件	最小值	典型值	最大值	单位
电源电压	VDD	-	2.7	3.3	3.6	V
I/O 上拉电阻	Rpu	-	--	50		KΩ
工作频率	FCPU	-		12		MHz
RTC 输入频率	FRTC	-	--	75K		Hz
高频工作电流 1	IOPH1	FCPU = 12MHz（MCU 工作，程序存储器选择 SRAM，其他模块关闭。）	--	7.5	--	mA
高频工作电流 2	IOPH2	FCPU = 12MHz（MCU 工作，程序存储器选择 FLASH，其他模块关闭。）	--	8	--	mA
低频工作电流 1	IOPL1	FCPU = 75KHz（MCU 工作，程序存储器选择 SRAM，RTC 工作，其他模块关闭，采用外部 LDO 供电（LDO 功耗不计算入内）	--	70	--	μA
低频工作电流 2	IOPL2	FCPU = 75KHz（MCU 工作，程序存储器选择 SRAM，RTC 工作，其他模块关闭，采用内部 LDO 供电）	--	400	--	μA
低频工作电流 3	IOPL3	FCPU = 75KHz（MCU 工作，程序存储器选择 flash，RTC 工作，其他模块关闭，采用内部 LDO 供电）	--	1.5	--	mA
SLEEP 电流 1	Is1	FCPU = 75KHz（MCU 在 sleep 模式，程序存储器选择 SRAM，RTC 工作，其他模块关闭，采用外部 LDO 供电（LDO 功耗不计算入内））	--	40	--	μA
SLEEP 电流 2	Is2	FCPU = 75KHz（MCU 在 sleep 模式，程序存储器选择 SRAM，RTC 工作，其他模块关闭，采用内部 LDO 供电）	--	370	--	μA

参　　数	符 号	测 试 条 件	最小值	典型值	最大值	单位
SLEEP 电流 3	Is3	FCPU = 12MHz（MCU 在 sleep 模式，程序存储器选择 SRAM 或 FLASH 都可，RTC 工作，其他模块关闭，采用内部 LDO 供电）	--	3.5	--	mA
静态电流	IQ	关闭主振荡，RTC 采用 75K 时钟工作，LDO 关闭，其他模块关闭。	-	14	--	μA
拉电流 (I^2C 外的 I/O)	IOH	VOH = 3V	-	-3.0	-	mA
灌电流 (I^2C 外的 I/O)	IOL	VOL = 0.3V	-	4.0	-	mA
灌电流 (I^2C 的 I/O)	IOL	VOL = 0.3V	-	9.0	-	mA
输入高电平	VIH	P0/P1/P2/P9	2.0	-	-	V
输入高电平	VIH	SEG31~0/COM3~1	1.5	-		V
输入高电平	VIH	COM0	2.0	-		V
输入高电平	VIH	P10	1.5	-		V
输入低电平	VIL	P0/P1/P2/P9		-	0.7	V
输入低电平	VIL	SEG31~0/COM3~1		-	0.8	V
输入低电平	VIL	COM0		-	0.7	V
输入低电平	VIL	P10		-	0.8	V

7. 管脚排列及说明

7.1 管脚排列图

7.1.1 SC9351



7.2 管脚描述

7.2.1 SC9351 管脚说明

管脚号	管脚名称	I/O	管脚描述
1	P2.2	I/O	在外扩总线模式，输出 notDMRD;复用外部中断 InT2
2	VSSA	--	ADC 模块地
3	VDDA	--	ADC 模块电源
4~6	AN0~AN2		ADC 输入通道 0~2
7	VSS	--	数字地
8	VDD	--	IO、RTC、64Byte RAM 的电源
9	XTIN	I	75KHz 晶振输入脚
10	XTOUT	O	75KHz 晶振输出脚
11	nTST	I	测试使能脚，内接上拉电阻；正常使用接高
12	nRST	I	复位脚，内接上拉电阻；低电平复位
13	P2.4	I/O	通用输入输出脚；复用为外部中断输入 INT0
14	P2.5	I/O	通用输入输出脚；复用为外部中断输入 INT1
15	VDD2o5O	--	LDO 的 2.5V 输出脚，外接 1~10uF 电容.给内核供电
16	VDDLDO	--	LDO 供电电源，输入电压 2.7~3.3V
17	VSSLDO	--	LDO 地
18~25	P5.0~5.7	I/O	通用输入输出 端口 P5, 共 8 个管脚
26~28	P6.0~6.2	I/O	通用输入输出 端口 P6, 共 3 个管脚
29	VDD	--	3.3V 电源
30	Xin	I	12MHz 晶振输入脚。
31	Xout	O	12MHz 晶振输出脚。
32	VSS	--	地
33	P7.5	I/O	通用输入输出脚；**debug 模式，作为串行通讯的数据输入**
34	P7.6	I/O	通用输入输出脚；**debug 模式，作为串行通讯的数据输出**
35	P7.7	I/O	通用输入输出脚；**debug 模式，输入同步通信时钟**
36	VDD2o5	--	2.5V 电源输入
37	VSS	--	地
38	P10.0	I/O	通用输入输出脚；可复用为 I^2C 的数据口
39	P10.1	I/O	通用输入输出脚；可复用为 I^2C 的时钟口
40	nDBG	I	Debug 模式选择，内接上拉电阻；接地，进入 Debug 模式
41	P9.1	I/O	可复用为 UART0 的 TXD
42	P9.2	I/O	可复用为 UART0 的 RXD
43	P9.3	I/O	可复用为 UART1 的 TXD
44	P9.4	I/O	可复用为 UART1 的 RXD

管脚号	管脚名称	I/O	管脚描述
45	P9.5	I/O	通用输入输出脚；复用为 SPI 的数据输入脚
46	P9.6	I/O	通用输入输出脚；复用为 SPI 的数据输出脚
47	P9.7	I/O	通用输入输出脚；复用为 SPI 的时钟脚
48	VDD	--	3.3V 电源
49	VSS	--	地
50	VDD2o5	--	2.5V 电源输入
51~58	P0.0~0.7	I/O	通用输入输出 端口 P0. 共 8 个管脚
59	P1.0	I/O	通用输入输出脚；复用为 PWM 波形信号输出
60	P1.1	I/O	通用输入输出脚
61	P1.2	I/O	通用输入输出脚
62	P1.6	I/O	通用输入输出脚；复用为外部中断输入 INT6
63	P1.7	I/O	通用输入输出脚；复用为外部中断输入 INT7
64	P2.0	I/O	通用输入输出脚；复用为 BUZ 输出

7.2.2 管脚使用参考

在使用 SC9351 时要注意以下事项。

- 如果采用内部 LDO，那么 VDD2o5O 脚输出 2.5V 电压。其它 VDD2o5 脚可以从这里取电。如果内部 LDO 禁止,那么 VDD2o5O 脚需要用一个大电阻（>200K 欧姆）接地，此时需要外部提供一个 2.5V 电源供给所有 VDD2o5 脚。
- 不用的 IO 脚建议通过程序配置成固定状态，使这些管脚输出固定的低电平或高电平。如果设置成输入，建议外部通过 10K 电阻上拉或下拉。既可以保护内部 CMOS 器件，也能仿减低电流消耗。
- ADC 电源地
- …
-

8. S51 MCU核

SC9351采用兼容MCS-51指令集的S51 MCU核，用户可以使用标准的805x汇编器和编译器进行软件开发。

S51 MCU核与标准8051核的区别在于：

- S51 MCU 核支持双 DPTR。
- 在相同的外接晶振频率情况下，S51 MCU 的指令周期比标准 8051 快 3~6 倍。
- S51 MCU 核支持两个扩展 RAM 区。
- S51 MCU 核的 MOVX A,@Ri 和 MOVX @Ri,A 指令只能访问片内 XRAM 中的 00H~FFH 的地址空间。
-

S51 MCU 核仅支持 2 种工作模式：正常工作模式和待机模式。工作模式控制方式也与标准 8051 核的不同。

8.1 CPU架构

（系统 SFR 描述：PC/SP/ACC/PSW/DPTR 等）S51 核是 Silan 公司 8bit MCU 产品的中央处理单元。在保持指令集兼容的基础上对传统 51 基础上进行了改进。本节介绍 CPU 相关的工作寄存器，累计器单元，程序状态字等。

8.1.1 工作寄存器

直接寻址空间的低32字节（地址范围：00H~1FH）可以作为4个工作寄存器组访问。每个组有8个8位寄存器，称为R0~R7。设置程序状态字（PSW）中的RS0（PSW.3）和RS1（PSW.4）位可选择某一个工作寄存器组作为当前的工作寄存器组。在调用子程序或进入中断服务程序时可通过设置工作寄存器组进行快速地现场切换。未使用到的工作寄存器组可作为普通的用户RAM使用。

间接寻址方式使用R0和R1作为间址寄存器。

8.1.2 CPU 特殊寄存器

直接寻址空间的80H~FFH的范围为特殊功能寄存器区（SFR区），因此只允许使用直接寻址方式访问特殊功能寄存器（SFR）。S51 MCU核包含了标准8051中的一些SFR，有：

- 累加器 ACC （字节地址：E0H）
- B 寄存器 B （字节地址：F0H）
- 程序状态字 PSW （字节地址：D0H）
- 堆栈指针 SP （字节地址：81H）
- 数据指针 DPTR （字节地址：83H，82H）
- 定时器控制寄存器 TCON （字节地址：88H）
- 中断允许控制寄存器 IE （字节地址：A8H）
- 中断优先级控制寄存器 IP （字节地址：B8H）

还包含一些新增的SFR，有：

- DPTR 选择寄存器 AUXR1 （字节地址：A2H）
- 内部扩展 RAM 区选择控制寄存器 CS_IntDM_CTRL （字节地址：9EH）
- 工作模式控制寄存器 ModeCtrl （字节地址：94H）

以上仅列出S51 MCU核的SFR，外设模块的SFR不在此列出。SFR区中未使用的地址保留为将来使用，访问这些地址可能会产生不确定的结果，应予避免。

本节只介绍部分SFR，其余将在后续章节中讲述。

1) 累加器 ACC （字节地址：E0H）

SFR/地址：	ACC / E0H							
位 序 号：	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位 定 义：	ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0
位 地 址：	E7H	E6H	E5H	E4H	E3H	E2H	E1H	E0H
访问权限：	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复 位 值：	0	0	0	0	0	0	0	0

说明：

- 累加器 ACC 是可按位寻址的 SFR。
- 在指令集的汇编表达式中用 A 作为累加器 ACC 的助记符。

2) B 寄存器 B （字节地址：F0H）

SFR/地址：	B / F0H							
位 序 号：	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位 定 义：	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
位 地 址：	F7H	F6H	F5H	F4H	F3H	F2H	F1H	F0H
访问权限：	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复 位 值：	0	0	0	0	0	0	0	0

说明：

- B 寄存器是可按位寻址的 SFR。
- 乘法指令的两个操作数分别取自 A 和 B，乘积的高 8 位存于 B 中，低 8 位存于 A 中。
- 除法指令中，A 存放被除数，B 存放除数，商存放于 A 中，余数存放于 B 中。
- 其它指令中，B 可作为一般通用寄存器或 RAM 单元使用。

3) 程序状态字 PSW （字节地址：D0H）

SFR/地址：	PSW / D0H							
位 序 号：	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位 定 义：	CY	AC	F0	RS1	RS0	OV	F1	P
位 地 址：	D7H	D6H	D5H	D4H	D3H	D2H	D1H	D0H
访问权限：	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复 位 值：	0	0	0	0	0	0	0	0

说明：

- 程序状态字 PSW 是可按位寻址的 SFR，它包含了程序执行后的状态信息，供程序查询或判别之用。
- CY：进位标志位。在执行加法指令之后，若运算结果最高位有向前进位则 CY 置‘1’，若运算结果最高位没有向前进位则 CY 清‘0’；在执行减法指令之后，若运算结果最高位有向前借位则 CY 置‘1’，若运算结果最高位没有向前借位则 CY 清‘0’；乘、除法运算后，CY 总被清‘0’；CY 也是执行位操作时的位累加器，在指令集的汇编表达式中用 C 作为位累加器 CY 的助记符。
- AC：半进位标志位。在执行加法指令之后，若运算结果低半字节有向高半字节进位则 AC 置‘1’，若运算结果低半字节没有向高半字节进位则 AC 清‘0’；在执行减法指令之后，若运算结果低半字节有向高半字节借位则 AC 置‘1’，若运算结果低半字节没有向高半字节借位则 AC 清‘0’。
- F0：用户标志位。其含义由用户自定义。
- RS1、RS0：工作寄存器组选择控制位。设置 RS1、RS0 的值的组合，可切换当前的工作寄存器组，对应关系如下：

RS1	RS0	寄存器组	内部 RAM 地址
0	0	第 0 组	00H~07H
0	1	第 1 组	08H~0FH
1	0	第 2 组	10H~17H
1	1	第 3 组	18H~1FH

- ✧ OV：溢出标志位。加、减法运算后，若补码结果超出(-128,127)范围则 OV 置‘1’，无溢出则 OV 清‘0’；乘法运算后，若乘积大于 FFH，则 OV 置‘1’，否则 OV 清‘0’；除法运算后，正常情况下 OV 被清‘0’，但若除数为 0 导致结果无法确定，则 OV 置‘1’。
- ✧ F1：用户标志位。其含义由用户自定义。
- ✧ P：奇偶校验标志位。任意一条指令执行完之后，若累加器 ACC 中 8 个位的和为奇数时 P 置‘1’，为偶数时 P 清‘0’。

4) 堆栈指针 SP （字节地址：81H）

S51 MCU核的堆栈结构属于向上生成型。在使用堆栈之前，须先给SP赋值。数据进栈时，SP先自动增1，再把数据放到SP指向的RAM单元；数据出栈时，先把SP指向的RAM单元的值读出，SP再自动减1。

SP始终指向内部数据空间的RAM单元，当SP的值为00H~7FH时，指向直接寻址的RAM区，当SP的值为80H~FFH时，指向间接寻址的扩展RAM区。由于S51有两个扩展RAM区（复用C0H~FFH的地址空间），因此最好不要将堆栈区设置到C0H~FFH区域。

SP的复位值为07H。

5) 数据指针 DPTR （字节地址：83H，82H）

DPTR是一个16位的SFR，不可按位寻址，其高位字节寄存器用DPH表示（地址：83H），低位字节用DPL表示（地址：82H）。DPTR主要用作16位间址寄存器，也可以作为两个独立的8位寄存器（DPH、DPL）使用。

DPTR的复位值为0000H。

6) DPTR 选择寄存器 AUXR1 （字节地址：A2H）

S51有两个DPTR寄存器，使用相同的地址，可通过设置DPTR选择寄存器AUXR1的值来选择其中一个DPTR作为当前的16位数据指针（或DPH、DPL寄存器）。双DPTR在查表操作时可以节省很多因切换数据指针造成的代码和时间。

AUXR1是只写寄存器，只能按字节寻址，读该寄存器将获得不确定的值。AUXR1=00H时，选择系统默认的第一个DPTR作为当前的16位数据指针（或DPH、DPL寄存器）；AUXR1=01H时，选择第二个DPTR作为当前的16位数据指针（或DPH、DPL寄存器）。

AUXR1 的复位值为00H。

8.2 存储器组织

S51 8bit CPU 的存储器组织与标准 **8051** 的存储器组织相似. 哈佛结构的存储器结构由分离的数据存储器与程序存储器组成,各自有 **64K** 字节空间. 地址范围从 **0000H** 到 **FFFFH.**

❖ 8.2.1 程序空间

程序空间内包括 4K 字节的 ROM（地址范围：0000H~0FFFH）和最大 60K 字节的 FLASH（地址范围：1000H~FFFFH）。通过调用 ROM 里的系统函数，可对 FLASH 进行擦/写操作。

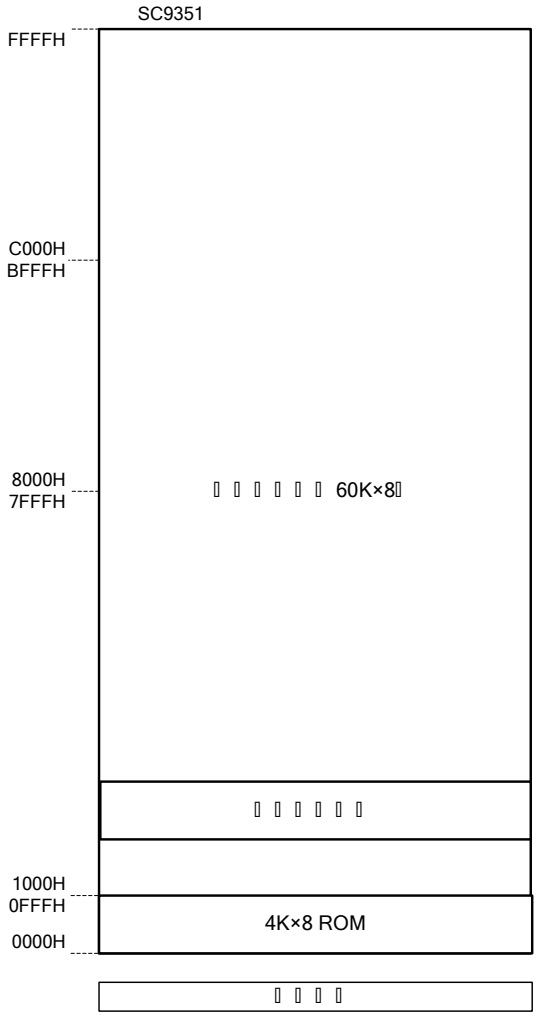


图 8.2.1-1：程序地址空间

指令存储器的某些地址空间被用于特定的程序段：

- ✧ 0000H~0002H：程序的起始地址，电路复位后，总是从 0000H 开始执行程序；
- ✧ 0003H~000AH：外部中断 0 的向量地址；
- ✧ 000BH~0012H：定时器 0 中断的向量地址；

- ✧ 0013H~001AH：外部中断 1 的向量地址；
- ✧ 001BH~0022H：定时器 1 中断的向量地址；
- ✧ 0023H~002AH：串行口中断的向量地址；
- ✧ 002BH~0032H：定时器 2 中断的向量地址；

SC9351 没有用到定时器 2 中断。

❖ 8.2.2 数据空间

SC9351 的数据存储器 有两部分组成.分别是内部数据存储器以及外部数据存储器.MOV 指令访问内部数据存储器, MOVX 指令访问外部数据存储器.

内部数据存储器地址空间为 0000H~00FFH，物理上分为几个不同存储区。

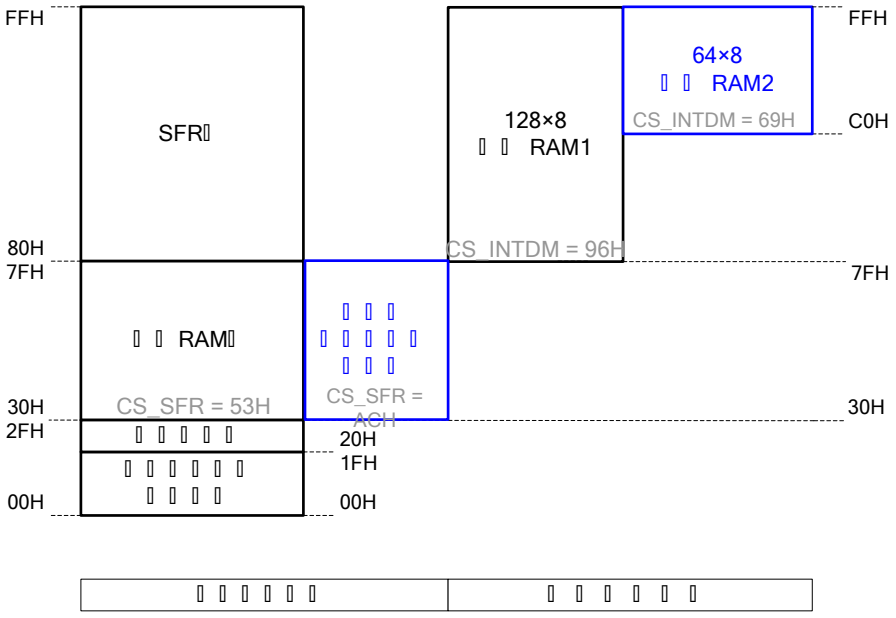


图 8.2.2-1： 内部数据空间分配

00H~7FH 的 128 字节地址空间是 RAM 区。该存储区内数据可以通过间接或直接寻址访问。同时 30H~7FH 的 80 字节地址空间可以扩展为特殊功能寄存器，寻址方式和 RAM 相同。

从 80H~FFH 的 128 字节地址空间是 RAM 区和特殊功能寄存器的重叠区。直接寻址指令访问特殊功能寄存器，间接寻址指令访问 RAM 区。和 8051 不同，C0H~FFH 的 64 字节地址空间可以扩展为额外的 RAM 区 RAM2。3.3V 电源 VDD 给 RAM2 供电，当内核掉电时，可以利用这片 RAM 保存数据。同样 RAM2 也只能 通过间接寻址访问。

扩展 SFR 区以及扩展的数据区 RAM2 访问控制参见 8.2.3 节存储器管理。

注意：地址空间的扩展会影响堆栈的使用，软件开发人员要注意堆栈地址的设置，最好设置在 0x80~0xBF 地址空间，否则要特别注意堆栈地址和扩展地址空间的冲突。

位	位地址

字节地址		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
用户 RAM 区	7FH … 30H	（只能是字节寻址，可作堆栈和数据缓冲）							
可位寻址区	2FH	7FH	7EH	7DH	7CH	7BH	7AH	79H	78H
	2EH	77H	76H	75H	74H	73H	72H	71H	70H
	2DH	6FH	6EH	6DH	6CH	6BH	6AH	69H	68H
	2CH	67H	66H	65H	64H	63H	62H	61H	60H
	2BH	5FH	5EH	5DH	5CH	5BH	5AH	59H	58H
	2AH	57H	56H	55H	54H	53H	52H	51H	50H
	29H	4FH	4EH	4DH	4CH	4BH	4AH	49H	48H
	28H	47H	46H	45H	44H	43H	42H	41H	40H
	27H	3FH	3EH	3DH	3CH	3BH	3AH	39H	38H
	26H	37H	36H	35H	34H	33H	32H	31H	30H
	25H	2FH	2EH	2DH	2CH	2BH	2AH	29H	28H
	24H	27H	26H	25H	24H	23H	22H	21H	20H
	23H	1FH	1EH	1DH	1CH	1BH	1AH	19H	18H
	22H	17H	16H	15H	14H	13H	12H	11H	10H
	21H	0FH	0EH	0DH	0CH	0BH	0AH	09H	08H
	20H	07H	06H	05H	04H	03H	02H	01H	00H
通用寄存器区	1FH … 18H	第 3 组工作寄存器的 R7 … 第 3 组工作寄存器的 R0							
	17H … 10H	第 2 组工作寄存器的 R7 … 第 2 组工作寄存器的 R0							
	0FH … 08H	第 1 组工作寄存器的 R7 … 第 1 组工作寄存器的 R0							
	07H … 00H	第 0 组工作寄存器的 R7 … 第 0 组工作寄存器的 R0							

图 8.2.2-2：直接寻址的 128 字节 RAM

外部数据寄存器

外部数据存储器只能通过 MOVX 指令访问，地址空间为 0000H~FFFFH。SC9351 内部

集成了 8K 字节的 RAM 作为外部数据存储器，地址空间为 0000H~1FFFH。由于管脚限制，SC9351 不支持外部数据扩展。

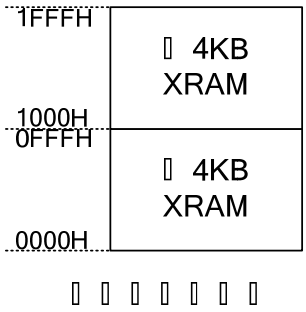


图 8.2.2-3：外部数据空间

外部 8K 字节 XRAM 分为高低两块，各有 4K 字节。用户程序运行时这两片 RAM 没有任何差别。进入 FLASH 编程模式时，低 4K XRAM 用于装载编程程序并执行。能够对整个 64K FLASH 空间进行编程。有些应用中低 4K Flash 已经固化相关系统程序，所以用户不能对该区域进行擦写操作。低 4K XRAM 运行程序时，高 4KXRAM 作为外部数据存储器使用。此时 MOVX 指令能读取低 4K Flash ，高 4K XRAM， 8K 以上空间的 Flash 区。而 1000~1FFFH 区的 Flash 无法通过 MOVX 读出。

❖ 8.2.3 存储器管理

内部 64K ByteFlash 既可以定义为程序存储器也可以定义为数据存储器。

当需要对 FLASH 中的程序升级时，首先将 FLASH 编程程序由 FLASH 导入内嵌的外部数据 RAM 区（0000H~0FFFH），然后使程序跳转到该 RAM 区执行程序，此时该 4096 字节的 RAM 作为程序存储器，而把 FLASH 作为外部数据存储器进行读操作，注意只能对 2000H~FFFFH 地址空间进行读操作。

通过执行 RAM 区的 FLASH 编程程序，控制 FLASH 编程接口，完成 FLASH 内的程序和数据的升级；

程序升级完成后，再跳转到 FLASH，进行正常的程序运行，此时 FLASH 程序存储器，而 RAM 作为外部数据存储器。

程序在低 4K XRAM 中运行时，低 4kFlash，高 4K XRAM 以及高于 2000H（8K 以上）空间映射作外部数据区可通过 MOVX 访问。

1) 芯片内集成的 8K 字节 RAM：

SC9351 在芯片内部集成了 8K 字节的 RAM 作为外部数据存储器，地址空间为 0000H~1FFFH，参考“地址空间说明”。

◆ **寄存器说明**

1) 程序在 FLASH 和 4K 字节 RAM 间的切换控制：

切换控制寄存器：

FlashCtrl（AEH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	ReadIFREN	ExtBoot	--	RunInExt	Jmp2Int

- ReadIFREN：当 FLASH 作为 ExtDM 进行读操作时：
  0：从 FLASH 的主存储区读取数据。
  1：从 FLASH 的 Information 存储区读取数据。
- ExtBoot：引导程序控制位。注意：ExtBoot=1，软件不可写。
  0：程序从 ExtDM 启动。
  1：程序从 flash 启动。
- RunInExt：RunInExt=1，程序在 flash 运行，RunInExt 缺省为 1。
- Jmp2Int：Jmp2Int=1，程序在 ExtDM 运行，Jmp2Int 缺省为 0。
- 当 RunInExt=1、Jmp2Int=0 时，程序在 flash 运行；当 RunInExt=0、Jmp2Int=0 时，程序在 RAM 运行；当 RunInExt=0、Jmp2Int=1 时，程序在 RAM 运行；当 RunInExt=1、Jmp2Int=1 时，程序在 RAM 运行。
- **在设置 RunInExt/Jmp2Int 后，程序并不会立即跳入 FLASH/RAM，在执行跳转指令后才会跳入 FLASH/RAM 运行该跳转指令指向指令。**

2) 内部数据存储器扩展控制：

参考“地址空间说明”。

- 64x8 RAM2（VDD 供电）和 128x8 RAM1 的地址切换：

CS_INTDM（94H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	CS_INTDM[7：0]							

由于内部数据存储器在掉电保持模式处于掉电状态，而一些关键数据又需要保存下来，所以 SC9351 在芯片内部集成了 64x8 的由外部直接供电的 RAM 模块 RAM2。RAM2 的地址空间为 0XC0~0xFF，和内部数据存储器 RAM1 的地址空间部分重合，需要通过 CS_INTDM 切换访问重叠的地址。

复位后默认选择访问 RAM1 区。当写 69H 到 CS_INTDM 后，选择 RAM2 为操作区；如果需要访问 RAM1 区，写 96H 到 CS_INTDM 寄存器既可。

- 扩展 SFR 寄存器区和用户 RAM 地址切换：

CS_SFR（93H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	CS_SFR[7：0]							

复位后默认选择访问用户 RAM 区。

当写 53H 到 CS_SFR 后，选择 RAM 区为操作区；如果需要访问扩展的 SFR 区，写 ACH 到 CS_SFR 寄存器既可。。

注意：堆栈指针最好设置在 0X80~0XBF 的地址空间，否则在编写中断程序时，要特别注意数据存储器的切换控制

❖ 8.2.4 特殊功能寄存器

地址	名　称	R/W	说　　明
8051 专用寄存器			
81H	SP	R/W	堆栈指针
82H	DPL	R/W	数据指针低字节
83H	DPH	R/W	数据指针高字节
87H	PCON	R/W	电源控制寄存器
88H	TCON	R/W	定时器/计数器控制寄存器
89H	TMOD	R/W	定时器/计数器方式控制寄存器
98H	SCON	R/W	串行控制寄存器
99H	SBUF	R/W	串行数据缓冲器
8AH	TL0	R/W	定时器/计数器 0（低字节）
8BH	TL1	R/W	定时器/计数器 1（低字节）
8CH	TH0	R/W	定时器/计数器 0（高字节）
8DH	TH1	R/W	定时器/计数器 1（高字节）
8EH	TIMPS	R/W	TIMER 预分频控制寄存器
A2H	AUXR1	R/W	DPTR 数据指针选择寄存器
A8H	IE	R/W	中断允许控制寄存器
B8H	IP	R/W	中断优先级控制寄存器
D0H	PSW	R/W	程序状态寄存器
E0H	ACC	R/W	CPU 累加器
F0H	B	R/W	CPU 的 B 寄存器
工作模式寄存器（扩展寄存器）			
31H	PSM_OSCREF	W	75K 晶振增益设置访问控制地址
32H	PDN_OSCREF	W	75K 晶振使能访问控制地址
33H	MCLKSEL	W	CPU 时钟选择访问控制地址
34H	PDN_OSCIN	W	12M 晶振使能访问控制地址
35H	PDN_LDO	W	LDO 使能访问控制地址
36H	OSCRSTCTRL	R	系统时钟与电源状态寄存器
37H	HSCSEL	W	高速晶振时钟选择访问控制地址
38H	LBDCTRL	R/W	LBD 控制寄存器
外部中断寄存器（扩展寄存器）			
39H	EINTF	R/W	外部中断标志位
3AH	EXTINTENABLE	W	外部中断源识别使能寄存器
3BH	EINT_EDGE	W	外部中断控制寄存器
3CH	IPLSR3_E	R/W	中断优先级选择寄存器 4
3DH	IPLSR2_E	R/W	中断优先级选择寄存器 3

地址	名　称	R/W	说　　明
3EH	IPLSR1_E	R/W	中断优先级选择寄存器 2
3FH	IPLSR0_E	R/W	中断优先级选择寄存器 1
40H	IER_E	R/W	外部中断(INT0 扩展)使能寄存器
41H	IPR_E	R/W	优先级中断状态寄存器
42H	ISR_E	R/W	中断状态寄存器
43H	ICR_E	R/W	中断屏蔽控制寄存器
IO 寄存器（扩展寄存器,位于 30~7FH 地址空间内）			
46H	P10OD	R/W	Not used
47H	P10PU	R/W	P10 口上拉控制寄存器
48H	P10_IOConfig	R/W	Not used
49H	P10	R/W	P10 端口寄存器
4BH	P9OD	R/W	P9 口开漏输出控制
4CH	P9PU	R/W	P9 口上拉控制寄存器
4DH	P9_IOConfig	R/W	P9 口 I/O 控制寄存器
C0H	P9	R/W	P9 端口寄存器
4F~50H 寄存器未使用，用户程序禁止读写。			
51H	P7OD	R/W	P7 口开漏输出控制
52H	P7PU	R/W	P7 口上拉控制寄存器
53H	P6OD	R/W	P6 口开漏输出控制
54H	P6PU	R/W	P6 口上拉控制寄存器
55H	P5OD	R/W	P5 口开漏输出控制
56H	P5PU	R/W	P5 口上拉控制寄存器
5AH~5DH 寄存器未使用，用户程序禁止读写。			
5FH	P2OD	R/W	P2 口开漏输出控制
60H	P2PU	R/W	P2 口上拉控制寄存器
D4H	P2_IOConfig	R/W	P2 口 I/O 控制寄存器
A0H	P2	R/W	P2 端口寄存器
64H	P1OD	R/W	P1 口开漏输出控制
65H	P1PU	R/W	P1 口上拉控制寄存器
66H	P1_IOConfig	R/W	P1 口 I/O 控制寄存器
90H	P1	R/W	P1 端口寄存器
69H	P0D	R/W	P0 口开漏输出控制
6AH	P0PU	R/W	P0 口上拉控制寄存器
6BH	P0_IOConfig	R/W	P0 口 I/O 控制寄存器
80H	P0	R/W	P0 端口寄存器
RTC 寄存器（扩展寄存器,位于 30~7FH 地址空间内）			

地址	名　称	R/W	说　　明
6DH	SECADJL	R/W	秒计时周期调整寄存器
6EH	SECADJH	R/W	秒计时周期调整寄存器
6FH	SECADJCON	R/W	秒校准控制寄存器
70H	RTC_CS	R/W	RTC 控制与状态寄存器
71H	YEARH	R/W	年高八位寄存器
72H	SEC	R/W	秒寄存器
73H	MIN	R/W	分钟寄存器
74H	HOUR	R/W	小时寄存器
75H	DAY	R/W	天寄存器
76H	WEEK	R/W	星期寄存器
77H	MON	R/W	月寄存器
78H	YEARL	R/W	年低八位寄存器
79H	**MIN_ALARM**	R/W	MIN 报警控制寄存器
7AH	HOUR_ALARM	R/W	HOUR 报警控制寄存器
7BH	DAY_ALARM	R/W	DAY 报警控制寄存器
7CH	**WK_ALARM**	R/W	WEEK 报警控制寄存器
7DH	CLKOUT_CTRL	R/W	CLKOUT 控制寄存器
7EH	TMCON	R/W	RTC 模块内置 8bit TIMER 控制寄存器
7FH	TIMER	R/W	RTC 模块内置 8bit TIMER 重载寄存器
WDT 寄存器			
84H	WDT_CTRL	R/W	WDT 控制寄存器
85H	WDT_CLR0	W	WDT 清零寄存器 0
86H	WDT_CLR1	W	WDT 清零寄存器 1
91H	SLEEP_CTRL	R/W	Sleep 模式控制寄存器
92H	SYS_STATUS	R/W	Sleep/WDT 状态寄存器
寄存器扩展区选择寄存器			
93H	CS_SFR	W	数据区 30~7F 访问切换控制寄存器
RAM 扩展区选择寄存器			
94H	CS_INTDM	W	数据区 C0~FFH 访问切换控制寄存器
端口复用控制寄存器			
96H	IOMUX	R/W	SPI/I2C 等外设复用通道使能寄存器
E6H	IOMUX5L	R/W	P5[3:0] 输入输出设置寄存器
E7H	IOMUX5H	R/W	P5[7:4] 输入输出设置寄存器
E3H	IOMUX6L	R/W	P6[2:0] 输入输出设置寄存器
EBH	IOMUX7H	R/W	P7[7:5] 输入输出设置寄存器
内部中断寄存器			

地址	名　称	R/W	说　　明
97H	ICR_I	R/W	中断屏蔽控制寄存器
9AH	ISR_I	R	中断状态寄存器
9BH	IPR_I	R	优先级中断状态寄存器
9CH	IER_I	R/W	INT1 扩展中断(内部各模块产生)使能控制
9DH	IPLSR0_I	R/W	中断优先级选择寄存器 4
9EH	IPLSR1_I	R/W	中断优先级选择寄存器 3
9FH	IPLSR2_I	R/W	中断优先级选择寄存器 2
A1H	IPLSR3_I	R/W	中断优先级选择寄存器 1
Flash 编程寄存器			
A5H	FSHWRADRH	R/W	FLASH 编程高 8 位地址寄存器
A6H	FSHWRADRL	R/W	FLASH 编程低 8 位地址寄存器
A7H	FSHWRDATA	R/W	FLASH 编程数据寄存器
A9H	FSHWRCON1	R/W	FLASH 编程控制寄存器 1
AAH	FSHWRCON2	R/W	FLASH 编程控制寄存器 2
ABH	FSHERSCON1	R/W	FLASH 擦除控制寄存器 1
ACH	FSHERSCON2	R/W	FLASH 擦除控制寄存器 2
ADH	FSHTIMER	R/W	FLASH 编程/擦除预分频控制寄存器
AEH	FlashCtrl	R/W	FLASH 切换控制寄存器
SPI 寄存器			
B1H	SPICR	R/W	SPI 控制寄存器
B2H	SPISR	R	SPI 状态寄存器
B3H	SPIBUF	W/R	SPI 发送/接收缓冲器
B4H	SPIBR	R/W	SPI 波特率设置寄存器
B5~BCH 寄存器未使用，用户程序禁止读写。			
BDH	BUZCR	W/R	**蜂鸣器**输出控制寄存器

地址	名　称	R/W	说　　明
I^2C 寄存器			
BEH	I2CRXB	R	数据接收二级缓冲寄存器
BFH	I2CSR	R	状态寄存器
DFH	I2CCR	W/R	控制寄存器
C1H	I2CSLA	W/R	从机地址/主机波特率设置寄存器
C2H	I^2CBUF	W/R	发送/接收缓冲寄存器
UART0 寄存器			
C3H	UART_BUF0	W/R	UART0 接收/发送缓冲寄存器
C4H	SCON0	W/R	UART0 控制寄存器
C5H	BRCON0	W/R	UART0 波特率控制寄存器
C6H	BRTIMER0	W/R	UART0 波特率设置寄存器
UART1 寄存器			
C7H	UART_BUF1	W/R	UART1 接收/发送缓冲寄存器
C9H	SCON1	W/R	UART1 控制寄存器
CEH	BRCON1	W/R	UART1 波特率控制寄存器
CFH	BRTIMER1	W/R	UART1 波特率设置寄存器
ADC 寄存器			
D1H	ADATA	R	AD 转换数据寄存器
D2H	ADCON	W	AD 控制寄存器
D3H	ADCIS	W	AD 通道输入选择寄存器
T2/T3 寄存器			
D5H	T2CON	R/W	T2 控制寄存器
D6H	T2REF	R/W	T2 预设寄存器
DAH	T3CON	R/W	TIMER3 控制寄存器
DBH	T3REF	R/W	TIMER3 预设寄存器
EE~FFH 寄存器未使用，用户程序禁止读写。			

使用提示： 各寄存器具体说明参见后面各章节. 另外请注意以下几点:

- 上表中标记为可读写的寄存器 实际上可能某些位是只读的. 而其它位可读写.
- 寄存器的某些位物理上没有实现, 但是读操作对应位是 0. 对应位标记为"R0"
- 寄存器的某些位物理上已实现, 但是没有使用. 对应位标记为"NU"
- 寄存器的某些位物理上没有实现, 读操作结果不定, 对应位标记为"X"
- 在描述寄存器复位值时, 后面统一采用 16 进制的方式. 为了方便书写, 所以有些物理上未实现的位也用 0 表示, 请参考有使用价值的位。

8.3 指令集

本节介绍 S51 指令集所包含的每条指令的基本功能，执行所需时钟周期，以及相应标志位的影响。

8.3.1 符号说明

- Rn — 当前选中的工作寄存器区的 8 个工作寄存器 R0~R7（n=0~7）。
- Ri — 当前选中的工作寄存器区中可作间址寄存器的 2 个寄存器 R0、R1（i=0，1）。
- dir — 8 位内部数据存储器单元的地址。可以是内部 RAM 单元的地址（00H~FFH）或 SFR 的地址（如 I/O 端口、控制寄存器、状态寄存器等）。
- #data — 包含在指令中的 8 位立即数。
- #data16 — 包含在指令中的 16 位立即数。
- addr16 — 16 位目的地址，用于 LCALL 和 LJMP 指令中。
- addr11 — 11 位目的地址，用于 ACALL 和 AJMP 指令中，它的地址必须与下一条指令的第一个字节的地址的高 5 位相同。
- rel — 8 位带符号的地址偏移量，用于 SJMP 和所有的条件转移指令中；偏移值相对于下一条指令的第一个字节的地址计算，在-128~+127 范围内取值。
- bit — 内部 RAM 或 SFR 中的直接寻址位。
- A — 累加器。
- B — B 寄存器，用于 MUL 和 DIV 指令中。
- C — 进位或借位标志，或布尔处理器中的累加器。
- @ — 间址寄存器或基址寄存器的前缀，如@Ri,@A,@DPTR。
- \ — 表示余数。
- ~ — 按位取反。
- (X) — X 中的内容。
- ← — 箭头左边的内容被箭头右边的内容所代替。
- → — 箭头右边的内容被箭头左边的内容所代替。
- ←→ —箭头两边的内容互换。
- ∧，∨，⊕ — 分别表示与，或，异或。
- [X1:X0] — 表示第 X1 位至第 X0 位。
- |←| — 表示循环左移一位。
- |→| — 表示循环右移一位。
- , — 用来分隔指令功能一栏顺序执行的步骤。

8.3.2 寻址方式

S51 核支持 6 种操作数寻址方式。分别是：

- 直接选址　指令中包含访问数据的 8 位地址，00~7FH 内的用户数据区以及扩展的 SFR 区以及 80~FFH 区内的 SFR 寄存器可以通过直接寻址访问。
- 间接寻址　指令中通过指定的寄存器获取访问数据的地址。S51 核利用寄存器 R0/R1 保存操作数的地址，通过改变 R0/R1 的值改变操作数的地址。所有的 RAM 区都可以间接寻

址。

- 寄存器寻址　指令操作的对象为寄存器（R0~R7）。直接访问寄存器完成数据处理。另外有些操作默认某些寄存器（累加器 A，寄存器 B，堆栈指针 SP）作为操作对象。比如：MUL　AB；　POP；　PUSH 等。
- 立即数寻址　指令需处理的 8 位或 16 位数据直接包含在指令中。如：MOV　A，#5AH。
- 变址寻址　用 PC 或 DPTR 作为基地址，累加器 A 的值作为偏移地址得到操作数目的地址。该寻址方式常见于查表操作。
- 位寻址　该寻址方式支持对 20~2FH 区中 16 字节数据共 128bit 进行位操作。

8.3.3　指令列表

表 1 – 指令列表

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
数据传送指令									
1	MOV A,Rn	A←Rn	E8H~EFH	1	2	×	×	×	√
2	MOV A,dir	A←(dir)	E5H	2	3	×	×	×	√
3	MOV A,@Ri	A←(Ri)	E6H,E7H	1	3	×	×	×	√
4	MOV A,#data	A←data	74H	2	2	×	×	×	√
5	MOV Rn,A	Rn←A	F8H~FFH	1	3	×	×	×	×
6	MOV Rn,dir	Rn←(dir)	A8H~AFH	2	3	×	×	×	×
7	MOV Rn,#data	Rn←data	78H~7FH	2	3	×	×	×	×
8	MOV dir,A	(dir) ←A	F5H	2	4	×	×	×	×
9	MOV dir,Rn	(dir) ←Rn	88H~8FH	2	2	×	×	×	×
10	MOV dir1,dir2	(dir1) ←(dir2)	85H	3	3	×	×	×	×
11	MOV dir,@Ri	(dir) ←(Ri)	86H,87H	2	3	×	×	×	×
12	MOV dir,#data	(dir) ←data	75H	3	4	×	×	×	×
13	MOV @Ri,A	(Ri)←A	F6H,F7H	1	4	×	×	×	×
14	MOV @Ri,dir	(Ri) ←(dir)	A6H,A7H	2	3	×	×	×	×
15	MOV @Ri,#data	(Ri) ←data	76H,77H	2	4	×	×	×	×
16	MOV DPTR,#data16	DPTR←data16	90H	3	3	×	×	×	×
17	MOVC A,@A+DPTR	A←(A+DPTR)	93H	1	4	×	×	×	√
18	MOVC A,@A+PC	A←(A+PC)	83H	1	4	×	×	×	√
19	MOVX A,@Ri	A←(Ri)	E2H,E3H	1	4	×	×	×	√
20	MOVX A,@DPTR	A←(DPTR)	E0H	1	4	×	×	×	√
21	MOVX @Ri,A	(Ri) ←A	F2H,F3H	1	4	×	×	×	×
22	MOVX @DPTR,A	(DPTR) ←A	F0H	1	4	×	×	×	×

(见下页)

(接上页)

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
23	PUSH dir	SP←SP+1,SP←(dir)	C0H	2	3	×	×	×	×
24	POP dir	(dir)←(SP),SP←SP-1	D0H	2	4	×	×	×	×
25	XCH A,Rn	A←→Rn	C8H~CFH	1	5	×	×	×	√
26	XCH A,dir	A←→(dir)	C5H	2	5	×	×	×	√
27	XCH A,@Ri	A←→(Ri)	C6H,C7H	1	5	×	×	×	√
28	XCHD A,@Ri	A[3:0]←→(Ri)[3:0]	D6H,D7H	1	4	×	×	×	√
算术运算指令									
1	ADD A,Rn	A←A+Rn	28H~2FH	1	2	√	√	√	√
2	ADD A,dir	A←A+(dir)	25H	2	3	√	√	√	√
3	ADD A,@Ri	A←A+(Ri)	26H,27H	1	3	√	√	√	√
4	ADD A,#data	A←A+data	24H	2	2	√	√	√	√
5	ADDC A,Rn	A←A+Rn+CY	38H~3FH	1	2	√	√	√	√
6	ADDC A,dir	A←A+(dir)+CY	35H	2	3	√	√	√	√
7	ADDC A,@Ri	A←A+(Ri)+CY	36H,37H	1	3	√	√	√	√
8	ADDC A,#data	A←A+data+CY	34H	2	2	√	√	√	√
9	SUBB A,Rn	A←A-Rn-CY	98H~9FH	1	2	√	√	√	√
10	SUBB A,dir	A←A-(dir)-CY	95H	2	3	√	√	√	√
11	SUBB A,@Ri	A←A-(Ri)-CY	96H,97H	1	3	√	√	√	√
12	SUBB A,#data	A←A-data-CY	94H	2	2	√	√	√	√
13	INC A	A←A+1	04H	1	2	×	×	×	√
14	INC Rn	Rn←Rn+1	08H~0FH	1	3	×	×	×	×
15	INC dir	(dir)←(dir)+1	05H	2	4	×	×	×	×
16	INC @Ri	(Ri) ←(Ri)+1	06H,07H	1	4	×	×	×	×
17	INC DPTR	DPTR←DPTR+1	A3H	1	2	×	×	×	×
18	DEC A	A←A-1	14H	1	2	×	×	×	√
19	DEC Rn	Rn←Rn-1	18H~1FH	1	3	×	×	×	×
20	DEC dir	(dir)←(dir)-1	15H	2	4	×	×	×	×
21	DEC @Ri	(Ri) ←(Ri)-1	16H,17H	1	4	×	×	×	×
22	MUL AB	BA←A×B	A4H	1	8	0	×	√	√
23	DIV AB	A\B←A÷B	84H	1	8	0	×	√	√
24	DA A	对 A 进行十进制调整	D4H	1	2	√	√	×	√

(见下页)

(接上页)

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
逻辑运算和移位指令									
1	ANL A,Rn	A←A∧Rn	58H~5FH	1	2	×	×	×	√
2	ANL A,dir	A←A∧(dir)	55H	2	3	×	×	×	√
3	ANL A,@Ri	A←A∧(Ri)	56H,57H	1	3	×	×	×	√
4	ANL A,#data	A←A∧data	54H	2	2	×	×	×	√
5	ANL dir,A	(dir)←(dir)∧A	52H	2	4	×	×	×	×
6	ANL dir,#data	(dir)←(dir)∧data	53H	3	4	×	×	×	×
7	ORL A,Rn	A←A∨Rn	48H~4FH	1	2	×	×	×	√
8	ORL A,dir	A←A∨(dir)	45H	2	3	×	×	×	√
9	ORL A,@Ri	A←A∨(Ri)	46H,47H	1	3	×	×	×	√
10	ORL A,#data	A←A∨data	44H	2	2	×	×	×	√
11	ORL dir,A	(dir)←(dir)∨A	42H	2	4	×	×	×	×
12	ORL dir,#data	(dir)←(dir)∨data	43H	3	4	×	×	×	×
13	XRL A,Rn	A←A⊕Rn	68H~6FH	1	2	×	×	×	√
14	XRL A,dir	A←A⊕(dir)	65H	2	3	×	×	×	√
15	XRL A,@Ri	A←A⊕(Ri)	66H,67H	1	3	×	×	×	√
16	XRL A,#data	A←A⊕data	64H	2	2	×	×	×	√
17	XRL dir,A	(dir)←(dir)⊕A	62H	2	4	×	×	×	×
18	XRL dir,#data	(dir)←(dir)⊕data	63H	3	4	×	×	×	×
19	CLR A	A←0	E4H	1	2	×	×	×	√
20	CPL A	A←~A	F4H	1	2	×	×	×	×
21	RL A	⌊←A[7]←…←A[0]←⌊	23H	1	2	×	×	×	×
22	RR A	⌊→A[7]→…→A[0]→⌊	03H	1	2	×	×	×	×
23	RLC A	⌊←CY←A[7]←…←A[0]←⌊	33H	1	2	√	×	×	√
24	RRC A	⌊→CY→A[7]→…→A[0]→⌊	13H	1	2	√	×	×	√
25	SWAP A	A[7:4]←→A[3:0]	C4H	1	2	×	×	×	×
控制转移指令									
1	AJMP addr11	PC←PC+2, PC[10:0]←addr11	addr[10:8] 00001B	2	3	×	×	×	×
2	LJMP addr16	PC←addr16	02H	3	3	×	×	×	×
3	SJMP rel	PC←PC+2+rel	80H	2	3	×	×	×	×

(见下页)

(接上页)

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
4	JMP @A+DPTR	PC←(A+DPTR)	73H	1	3	×	×	×	×
5	JZ rel	PC←PC+2, 若 A=0,则 PC←PC+rel	60H	2	2 +2	×	×	×	×
6	JNZ rel	PC←PC+2, 若 A≠0,则 PC←PC+rel	70H	2	2 +2	×	×	×	×
7	CJNE A,dir,rel	PC←PC+3, 若 A≠(dir),则 PC←PC+rel； 若 A≥(dir),则 CY←0, 否则 CY←1	B5H	3	3 +2	√	×	×	×
8	CJNE A,#data,rel	PC←PC+3, 若 A≠data,则 PC←PC+rel； 若 A≥data,则 CY←0, 否则 CY←1	B4H	3	2 +2	√	×	×	×
9	CJNE Rn,#data,rel	PC←PC+3, 若 Rn≠data,则 PC←PC+rel； 若 Rn≥data,则 CY←0,否则 CY←1	B8H~BFH	3	2 +2	√	×	×	×
10	CJNE @Ri,#data,rel	PC←PC+3, 若(Ri)≠data,则 PC←PC+rel； 若(Ri)≥data,则 CY←0,否则 CY←1	B6H,B7H	3	3 +2	√	×	×	×
11	DJNZ Rn,rel	Rn←Rn- 1,PC←PC+2, 若 Rn≠0,则 PC←PC+rel	D8H~DFH	2	3 +2	×	×	×	×

(见下页)

(接上页)

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
12	DJNZ dir,rel	(dir)←(dir)-1,PC←PC+3,若(dir)≠0,则PC←PC+rel	D5H	3	4+2	×	×	×	×
13	ACALL addr11	PC←PC+2,SP←SP+1,(SP)←PCL,SP←SP+1,(SP)←PCH,PC[10:0]←addr11	addr[10:8]10001B	2	6	×	×	×	×
14	LCALL addr16	PC←PC+3,SP←SP+1,(SP)←PCL,SP←SP+1,(SP)←PCH,PC←addr16	12H	3	8	×	×	×	×
15	RET	PCH←(SP),SP←SP-1,PCL←(SP),SP←SP-1	22H	1	5	×	×	×	×
16	RETI	PCH←(SP),SP←SP-1,PCL←(SP),SP←SP-1,从中断返回	32H	1	5	×	×	×	×
17	NOP	PC←PC+1	00H	1	2	×	×	×	×
位操作指令									
1	CLR C	CY←0	C3H	1	2	√	×	×	×
2	CLR bit	bit←0	C2H	2	4	×	×	×	×
3	SETB C	CY←1	D3H	1	2	√	×	×	×
4	SETB bit	bit←1	D2H	2	4	×	×	×	×
5	CPL C	CY←~CY	B3H	1	2	√	×	×	×
6	CPL bit	bit←~(bit)	B2H	2	4	×	×	×	×
7	ANL C,bit	CY←CY∧(bit)	82H	2	3	√	×	×	×

(见下页)

(接上页)

序号	助记符	指令功能	操作码	字节数	振荡周期	对标志位影响			
						CY	AC	OV	P
8	ANL C,/bit	CY←CY∧~(bit)	B0H	2	3	√	×	×	×
9	ORL C,bit	CY←CY∨(bit)	72H	2	3	√	×	×	×
10	ORL C,/bit	CY←CY∨~(bit)	A0H	2	3	√	×	×	×
11	MOV C,bit	CY←bit	A2H	2	3	√	×	×	×
12	MOV bit,C	bit←CY	92H	2	4	×	×	×	×
13	JC rel	PC←PC+2, 若 CY=1,则 PC←PC+rel	40H	2	2 +2	×	×	×	×
14	JNC rel	PC←PC+2, 若 CY=0,则 PC←PC+rel	50H	2	2 +2	×	×	×	×
15	JB bit,rel	PC←PC+3, 若(bit)=1,则 PC←PC+rel	20H	3	3 +2	×	×	×	×
16	JNB bit,rel	PC←PC+3, 若(bit)=0,则 PC←PC+rel	30H	3	3 +2	×	×	×	×
17	JBC bit,rel	PC←PC+3, 若(bit)=1,则 PC←PC+rel, (bit)←0	10H	3	5 +2	×	×	×	×

注：振荡周期是指外接晶振的振荡周期；有些指令的振荡周期表达成 M+N，是指当条件满足时再增加 N 个振荡周期。

9. 复位、时钟与工作模式

9.1 复位控制

SC9351 的复位源包括：上电复位，外部管脚复位，低压检测复位功能。此外在掉电保持模式，RTC 中断、外部中断也会产生复位信号，使 CPU 复位，系统重新进入工作状态。

复位发生后，所有 IO 口设置成输入，上拉功能禁止。程序从起始地址 0000H 才是执行。通常开始地址安排一条跳转指令，转入到主程序入口开始执行。

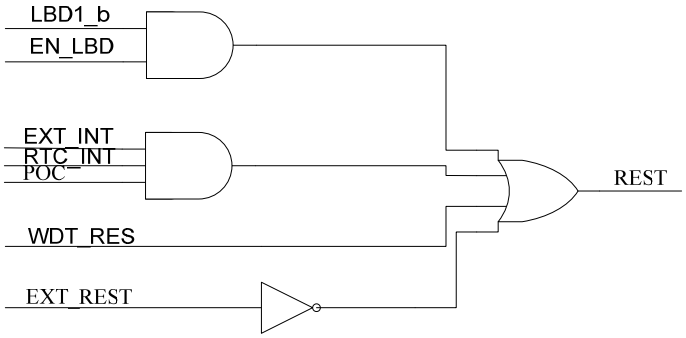


图 9.1-1：复位控制逻辑电路

使用提示：

- 在掉电保持模式，内部 LDO 关闭。POC＝1，外部中断、RTC 中断会产生复位信号，打开 LDO，使 CPU 复位。但不会影响 RTC、晶振控制、时钟控制、中断扩展控制。由 LDO 启动到 2.5V 电源稳定需要一段时间，所以需要保持约 15ms 的 CPU 复位时间。
- 在其他工作模式（高频模式、低频模式、Sleep 模式），LDO 正常工作，外部中断、RTC 中断则不会产生复位信号，而是产生中断请求，使 MCU 进行中断处理。
- notExtRst 管脚可以通过外接电阻、电容实现上电复位，使系统直接复位。也可以通过外接复位按键实现外部按键复位，使系统直接复位。
- LDO 低压检测信号可以直接使 MCU 复位，但不影响 RTC 模块的状态。
- WDT 溢出复位直接使 CPU 复位，但不影响电源管理、RTC、晶振控制、时钟控制、工作模式控制、外部中断扩展控制，内部中断扩展控制。

9.2 时钟控制

SC9351 有两路时钟，分别是 75KHz 晶振、12MHz 晶振。在高频工作模式，由 12MHz 晶振向 CPU 提供时钟（12MHz/6MHz）；在低频工作模式，由 75KHz 晶振向 CPU 提供时钟。

75KHz 晶振、12MHz 晶振的开关可编程控制。

在掉电保持模式，可关闭 12MHz 晶振，由 75KHz 晶振为 RTC、外部中断扩展模块提供时钟。

a) 75KHz 晶振向 RTC、TIMER、蜂鸣器提供时钟。

12MHz 晶振向 TIMER、I²C、UART、SPI、ADC 提供时钟。

9.3 工作模式

SC9351 具有多种工作模式：高频工作模式、低频工作模式、Sleep 模式、掉电保持模式，下面分别对各种工作模式进行详细说明：

9.3.1 高频工作模式

在高频工作模式，可通过软件选择由 12M 晶振产生的 12MHz 时钟，或者 2 分频时钟（6MHz）作为 CPU、I²C、SPI、UART、ADC、TIMER、WDT 等功能模块的时钟。75KHz 晶振为 RTC 提供时钟。所有功能模块都可以正常工作。通过程序设置，可以由高频工作模式进入低频工作模式、Sleep 模式、掉电保持模式。

9.3.1 低频工作模式

在低频工作模式，关闭 12MHz 晶振，由 75KHz 晶振产生的 75k 时钟为 CPU、I²C、SPI、UART、ADC、TIMER、WDT、RTC 提供时钟。通过程序设置，可以由低频工作模式进入高频工作模式、Sleep 模式、掉电保持模式。

注意：MCU 切换到低频工作模式后，12MHz 晶振并不会自动关闭，需要软件关闭。

低频模式进入高频模式前需要先软件打开高频晶振，且最好延时等待 3ms，高频晶振稳定后切换回高频模式。

9.3.2 Sleep模式

在 Sleep 模式，CPU、WDT 时钟被关闭，振荡电路在工作，而中断系统则继续在时钟驱动下工作，使 CPU 能够由中断事件从 Sleep 模式唤醒。

在 Sleep 模式，I²C、SPI、UART、ADC 的时钟也被关闭，而 TIMER、RTC、IO 端口，外部中断扩展模块和内部中断扩展模块继续在时钟驱动下工作，所以 CPU 可以由外部中断 Int0~Int7、RTC、TIMER 等中断事件唤醒，返回原来的工作模式执行中断服务程序。

只有被允许的中断事件才能将 CPU 唤醒，CPU 唤醒后将会首先执行相应的中断服务程序。

9.3.3 掉电保持模式

该模式只针对芯片采用内部 LDO 工作而言（采用外部 LDO，无该工作模式），关闭 LDO 后，LDO 不再为 MCU、64Kx8FLASH、8Kx8RAM、256x8RAM、I²C、SPI、UART、ADC、TIMER、WDT，内部中断扩展模块提供 2.5V 电源。而 12MHz 晶振、75KHz 晶振、RTC、64 字节扩展 RAM、IO 端口和外部中断扩展模块由外部提供电源，可以继续工作。

在掉电保持模式，75KHz 晶振为 RTC 提供时钟，IO、外部中断扩展模块则由 12MHz 晶振或 75KHz 晶振提供时钟。RTC 中断、外部中断 Int0~Int7 可以再打开 LDO，并使 CPU 复位，系统退出掉电保持模式，返回原来的工作模式重新开始工作。

在掉电保持模式，64 字节扩展 RAM 可以保存需要保存的数据。

9.3.4 寄存器说明

SLEEP_CTRL（91H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	--	--	--	--	Sleep

- Sleep：Sleep 模式控制位，Sleep 写 1，CPU 进入 Sleep 模式。

OSCRSTCTRL(36H)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	PDN_LDO	pdn_OscIn	pdn_OscRef	psm_OscRef	MClkSel	HSCSEL

- PDN_LDO：LDO 控制位，PDN_LDO=1，关闭 LDO 进入掉电保持模式，向 **35H** 依次写入 **0X39**、**0X6C**，缺省为 0。外部复位、RTC 中断、外部中断 Int0~Int7 可以使 PDN_LDO 复位为 0。
- pdn_OscIn：12MHz 晶振控制位，pdn_OscIn=1，关闭 12MHz 晶振。向 **34H** 依次写入 **0X65**、**0X9A**；pdn_OscIn=0，向 **34H** 依次写入 **0X65**、**0Xxx**（不等于 0X9A 的任意数）。缺省为 0。
- pdn_OscRef：75KHz 晶振控制位，pdn_OscRef=1，关闭 75KHz 晶振。向 **32H** 依次写入 **0X67**、**0X98**；pdn_OscRef=0，向 **32H** 依次写入 **0X67**、**0Xxx**（不等于 0X98 的任意数）。缺省为 0。
- psm_OscRef：75KHz 晶振控制位，psm_OscRef =1，75KHz 晶振工作在小电流模式。向 **31H** 依次写入 **0XA3**、**0X5C**；psm_OscRef=0，向 **31H** 依次写入 **0XA3**、**0Xxx**（不等于 **0X5C** 的任意数）。缺省为 0。
- MclkSel：MCU 时钟切换控制位。MclkSel =1，MCU 时钟由 75KHz 晶振提供，向 **33H** 依次写入 **0X68**、**0X97**；MclkSel =0，MCU 时钟由 12MHz 晶振提供，向 **33H** 依次写入 **0X68**、**0Xxx**（不等于 **0X97** 的任意数）。缺省为 0。
- HSCSEL：高频时钟分频控制位。HSCSEL =1，高频时钟为 12MHz 晶振的 2 分频时钟，向 **37H** 依次写入 **0X37**、**0X53**；HSCSEL =0，高频时钟为 12MHz 晶振时钟，向 **37H** 依次写入 **0X37**、**0Xxx**（不等于 **0X53** 的任意数）。缺省为 1。

SYS_STATUS（92H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	--	--	--	WDTOVF	SLPFG

- WDTOVF：看门狗溢出标志，看门够溢出时该为置 1；清看门狗该位置 0。
- SLPFG： 睡眠标志，SLEEP_CTRL 最低位置 1 电路进入睡眠状态该标志位置 1；清看门狗该位置 0。

注： 任何其它复位信号都不能清除这两个标志，只有清看门狗操作才能清除。

◆ **工作模式切换控制说明：**

1) 高频工作模式与低频工作模式的切换由 MclkSel 控制，当设置 MclkSel 为 1，MCU 工作在 75KHz 的低频模式；当设置 MclkSel 为 0，MCU 工作在高频模式；
2) MCU 从高频工作模式、低频工作模式都可以进入 Sleep 模式，程序设置 Sleep =1，MCU 即进入 Sleep 模式。当被中断事件唤醒，MCU 返回到原来的高频工作模式或低频工作模式。

3) MCU 从高频工作模式、低频工作模式都可以进入掉电保持模式，但为降低掉电保持模式的静态功耗，要求关闭 12MHz 晶振，由低频工作模式进入掉电保持模式。设置 PDN_LDO 为 1，则关闭 LDO 进入掉电保持模式。

高频工作模式的工作时钟可以选择 12MHz、6MHz，由 HSCSEL 控制。

10. 中断系统

除复位相关信号外，**SC9351** 一共有 18 个中断源， 这 18 个中断源通过 5 个与 8051 核相同的通道进入到中断处理模块进行处理。

S51 内核的 5 路中断分别是：INT0，INT1，TF0，TF1，TI，RI。 INT0 扩展成 4 路外部中断。而内部集成的各模块（比如 I2C，SPI 等）则共用 INT1 这路中断入口（后文简称扩展的内部中断）。两个 UART 口的发送与接收中断分别对应 TI 与 RI 中断通道。

扩展的外部中断与扩张的内部中断都可以独立设置优先级或启动屏蔽功能。由于它们各自共用一个中断入口，因此中断程序需要软件方式进行查询。比如 INT0 中断响应后，必须检查中断源寄存器才能确定是哪个管脚触发了中断。

外部 4 路中断分别来自 P1.6/P1.7/P2.4/P2.5 这几个管脚。可程序配置成上升沿或下降沿触发，每个中断源可以选择一个对应的优先级别(0~7)。不同的中断源应该赋予不同的优先级别。通过设置优先级控制寄存器(ICR)可以使 CPU 只响应比 ICR 优先级高的中断请求(数字越大优先级越高)。 由于该组中断对 CPU 具有同一级别，所以处理时高优先级的中断不会打断低优先级中断服务程序的执行，只有在低优先级中断服务程序完成后，才能响应高优先级的中断。只要中断标志位保持有效，CPU 就能够响应该中断。因此可以防止外部中断丢失。所有外部中断共用 8051 核 INT0 对应的入口地址 0003H。

内部模块中断主要来自 SC9351 内集成的数字与模拟模块。如 I^2C，SPI， ADC，T2，T3，RTC 等。 所有内部模块中断共用 8051 核 INT1 对应的入口地址 0013H。

由于 **SC9351** 集成了两个 UART 模块，在处理串行中断时，UART0 的 RI 与 TI 请求共用 8051 的 TI 请求。而 UART1 的 RI 与 TI 请求共用 8051 的 RI 请求。中断发生时需要查询对应标志才能确定中断来源。**SC9351** 内部串行口中断标志位 RI、TI 在响应中断后由硬件自动清零。

TF0、 TF1 分别为 8051 定时器/计数器T0、定时器/计数器T1 中断，高电平触发中断。详细说明见“定时/计数器”。

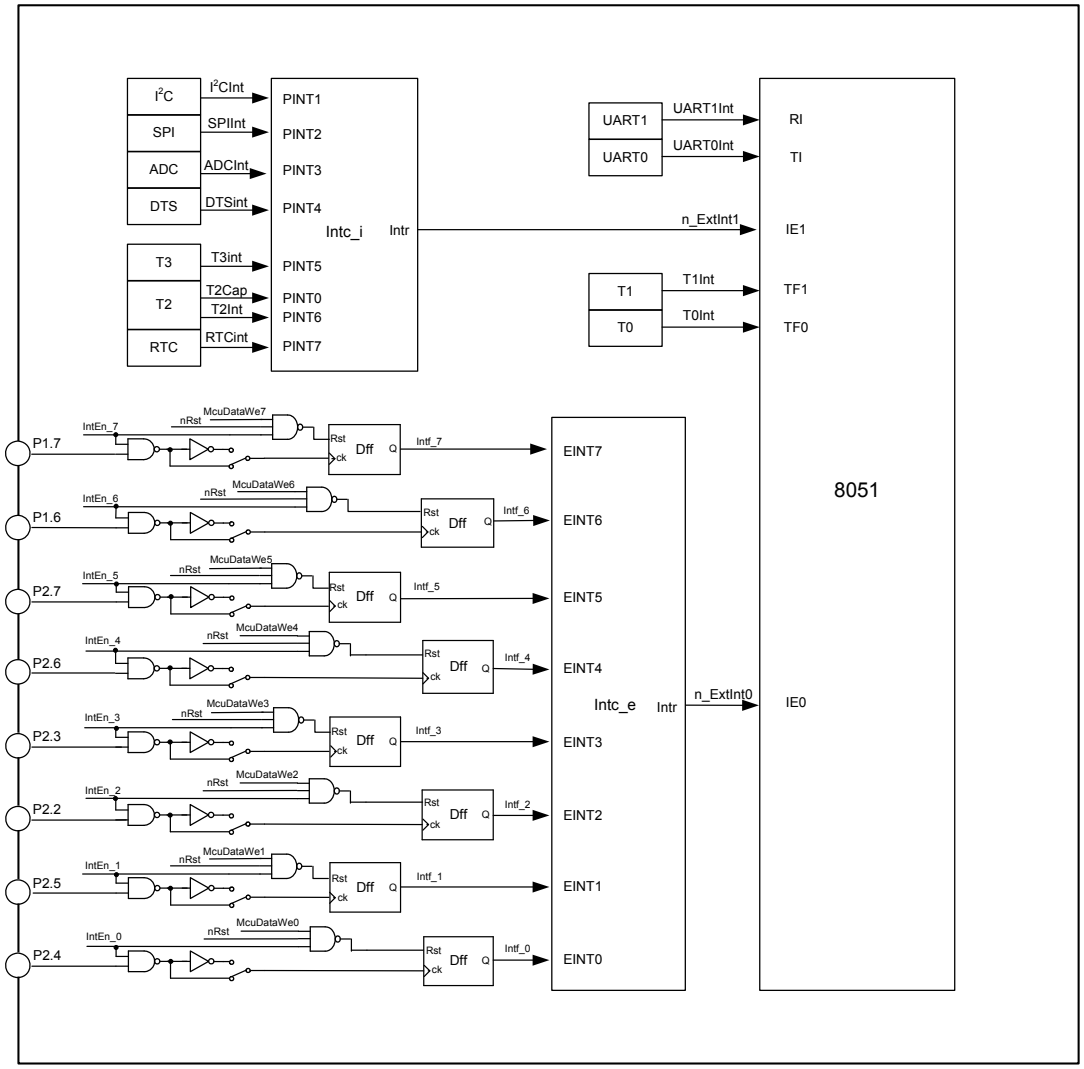


图 10-1：SC9351 中断结构

进入到 S51 核后， 中断处理模式和 8051 的中断控制模式一样。主要由中断允许控制寄存器 IE 和中断优先级寄存器 IP 控制。

要使用 S51 核中断，必须执行以下 3 个步骤：

1. 设置 IE 寄存器中 EA 位为 1
2. 设置对应的中断使能位为 1
3. 中断触发后进入到程序指针跳转到对应的矢量地址开始执行中断服务子程序。

中断模块	中断来源		8051 核入口	对应向量地址
外部中断 (4 路)	EINT0	P2.4	INT0	0003H
	EINT1	P2.5		
	EINT6	P1.6		
	EINT7	P1.7		
定时器 0	T0 溢出中断		TF0	000BH

内部模块中断	PINT0	T2 捕获中断	INT1	0013H
	PINT1	I^2C 中断		
	PINT2	SPI 中断		
	PINT3	ADC 中断		
	PINT4	保留		
	PINT5	T2 溢出中断		
	PINT6	T3 溢出中断		
	PINT7	RTC 中断		
定时器 1	T1 溢出中断		TF1	001BH
串口中断	UART0(RI0,TI0)		TI	0023H
	UART1(RI1,TI1)		RI	

图 10-2：中断矢量地址

寄存器说明

1) 内核中断允许控制寄存器 IE （字节地址：A8H）

SFR/地址：	IE / A8H							
位 序 号：	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位 定 义：	EA	（保留）	（保留）	ES	ET1	EX1	ET0	EX0
位 地 址：	AFH	AEH	ADH	ACH	ABH	AAH	A9H	A8H
访问权限：	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复 位 值：	0	0	0	0	0	0	0	0

说明：

- 中断允许控制寄存器 IE 是可位寻址的 SFR，它对中断的开放和屏蔽实现两级控制，如**错误！未找到引用源。**所示。
- EA：中断允许总控制位。EA=0，屏蔽所有的中断请求；EA=1，CPU 开放中断，但对各中断源的中断请求是否允许，还要取决于各中断源的中断允许控制位的状态。
- ES： 串口中断使能位。 ES=1，允许串口中断；ES=1，屏蔽串口中断。两个串口 UART0/1 的收发中断共用一个通道。所以源识别与使能在串口模块中各自的寄存器实现。
- ET1：定时器 1 溢出中断允许控制位。ET1=1，允许定时器 1 中断请求；ET1=0，屏蔽定时器 1 中断请求。
- EX1：外部中断 INT1 中断允许控制位。EX1=1，允许内部扩展中断模块产生的中断请求；EX1=0，屏蔽内部扩展中断模块产生的中断请求。
- ET0：定时器 0 溢出中断允许控制位。其含义与 ET1 类似。
- EX0：外部中断 INT0 中断允许控制位。EX0=1，允许外部扩展中断模块产生的中断请求；EX0=0，屏蔽外部扩展中断模块产生的中断请求

- IE 的复位值为 00H。

注: 保留位必须赋值为 0。

2) 中断优先级控制寄存器 IP （字节地址：B8H）

SFR/地址：	IP / B8H							
位 序 号：	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位 定 义：	（保留）	（保留）	（保留）	（保留）	PT1	（保留）	PT0	（保留）
位 地 址：	BFH	BEH	BDH	BCH	BBH	BAH	B9H	B8H
访问权限：	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复 位 值：	0	0	0	0	0	0	0	0

说明：

- 中断优先级控制寄存器 IP 是可位寻址的 SFR。
- PT1：定时器 1 中断优先级控制位。PT1=1，定时器 1 中断为高优先级；PT1=0，定时器 1 中断为低优先级。
- PT0：定时器 0 中断优先级控制位。其含义与 PT1 类似。
- 相同优先级的中断请求遵守自然优先级，定时器 0 中断的自然优先级比定时器 1 中断的自然优先级高。
- IP 的复位值为 00H。

保留位必须赋值为 0。

11.1.1 外部中断识别

SC9351 的 4 路外部中断从 P2.4/2.5 以及 P1.6/1.7 输入。可程序选择上升沿/下降沿有效记录中断标志。如果外部中断使能寄存器中对应的使能位没有置 1，那么外部中断请求不会记录在源寄存器 EINTF 中。

寄存器说明

外部中断边沿选择寄存器（复位值：00H）

EINT_EDGE（3BH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	EINT_EDGE7	EINT_EDGE6	没有使用				EINT_EDGE1	EINT_EDGE0

- EINT_EDGEn：外部中断 Intx 的上升沿/下降沿触发有效控制位；
  1：对应中断源上升沿有效；
  0：对应中断源下降沿有效；

注意：这里的上升沿/下降沿触发有效是针对外部中断标志位 EINTF 的，而不是 S51 核的。我们通过 EINT_EDGE 控制寄存器可以选择 IO 端口输入上升沿或下降沿来使中断源寄存器 EINTF 对应位置 1，由 EINTF 向外部中断处理模块块发出中断请求，外部中断处理模块块是高电平触发的。然后由外部中断处理模块块向 8051 发出低电平/下降沿的中断请求。

外部中断源使能寄存器（复位值：00H）

EINTE (3AH)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	EINTE 7	EINTE 6	没有使用				EINTE 1	EINTE 0

- EINTEn：外部中断的使能控制位
  1：允许外部中断请求，IO 端口作为外部中断输入；
  0：不允许外部中断请求，该管脚为通用 IO 端口；

外部中断请求标志寄存器（复位值：00H）

EINTF （39H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	EINTF 7	EINTF 6	没有使用				EINTF 1	EINTF 0

- EINTFn：外部中断请求标志位，由外部中断的边沿（上升沿/下降沿）触发，必须通过软件清零；
  1：表明发生了外部中断请求；
  0：没有发生外部中断请求；

注意：EINTF 向外部中断处理模块块发出高电平中断请求，所以在中断服务程序里必须立即对 EINTF 寄存器相应标志位清零，使外部中断处理模块块能够向 8051 请求低优先级的中断，8051 外部中断 Int0 应设置为低电平有效，这样 MCU 在退出中断服务程序后，即可响应其他中断。以保证低优先级的中断不会丢掉。

11.1.1 外部中断处理

外部中断处理模块负责接收处理过的外部中断请求。前级处理阶段使能的中断源产生中断请求后输出高电平到这里。由该模块负责处理，产生一个低电平请求通过 INT0 通道进入到 S51 核的中断模块。

最大支持 8 个中断源输入，所有的中断源输入均为沿有效。SC9351 使用了 4 个外部中断。

每个中断源可分配一个相应的从 0-7 的中断优先级，每个优先级相应的中断向量值不能被重新编程，中断向量值即为对应的优先级数。

每个优先级的中断都可分别控制其使能与否，可屏蔽指定优先级及其以下的中断请求。

当 MCU 工作于 Sleep 模式、掉电保持模式，外部中断处理模块块仍然能够采集外部中断 EInt0~EInt7，产生中断请求信号，使 MCU 从 Sleep 模式唤醒，或使 MCU 复位而退出掉电保持模式。

注意： 外部中断处理模块块的 8 个中断请求对应的是 8051 的同一个中断 INT0，所以高优先级的中断不会打断低优先级中断服务程序的执行，**只有在低优先级中断服务程序完成后，才能响应高优先级的中断。**

寄存器说明

1) ICR_E：中断控制寄存器

ICR_E	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

（43H）	--	--	--	--	ME	MASK2	MASK1	MASK0

- ME：中断屏蔽使能位，缺省为 0；
  - 0：禁止中断屏蔽
  - 1：使能中断屏蔽
- MASK2-0：指定中断屏蔽的优先级范围，缺省为 0h；
  - 000：屏蔽 0 优先级的中断请求
  - 001：屏蔽 1-0 优先级的中断请求
  - 010：屏蔽 2-0 优先级的中断请求
  - 011：屏蔽 3-0 优先级的中断请求
  - 100：屏蔽 4-0 优先级的中断请求
  - 101：屏蔽 5-0 优先级的中断请求
  - 110：屏蔽 6-0 优先级的中断请求
  - 111：屏蔽 7-0 优先级的中断请求

2) ISR_E：中断状态寄存器，反映了输出给 CPU 的中断状态；

ISR_E（42H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	INTR	--	VEC2	VEC1	VEC0

- INTR：中断请求标志位，缺省为 0；在中断请求信号无效后，该标志位也立即会复位为 0。
  - 0：没有产生中断给 CPU
  - 1：有中断请求输出给 CPU
- VEC2-0：输出给 CPU 的中断向量值，其数值即为输出到 CPU 的中断的优先级，这个数值会一直保持到新的中断请求发生，缺省为 0h；

3) IPR_E：中断源识别寄存器

IPR_E（41H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0

- IPx：未决中断标志位，缺省为 0；在中断请求信号无效后，该标志位也立即会复位为 0。
  - 1：表示至少有一个优先级为 x 的中断请求产生
  - 0：表示没有优先级为 x 的中断请求产生

4) IER_E：中断请求使能寄存器

IER_E（40H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0

- IEx：中断使能位，缺省为 0；
  - 1：表示与中断源 x 优先级相同的中断请求被允许；（建议：实际应用时对不同的中断源设置不同的优先级）
  - 0：表示对中断源 x 的中断请求是否被允许不产生影响。

5) 中断优先级控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IPLSR0_E（3FH）	--	PL2_1	PL1_1	PL0_1	--	PL2_0	PL1_0	PL0_0
IPLSR1_E（3EH）	--	PL2_3	PL1_3	PL0_3	--	PL2_2	PL1_2	PL0_2
IPLSR2_E（3DH）	--	PL2_5	PL1_5	PL0_5	--	PL2_4	PL1_4	PL0_4
IPLSR3_E（3CH）	--	PL2_7	PL1_7	PL0_7	--	PL2_6	PL1_6	PL0_6

- PL2_x-PL0_x：指明了中断源 x 的中断优先级，缺省为 0h，如下表所示：

PL2_x-PL0_x	优先级	中断向量 VEC2-0
000	0（最低优先级）	000
001-110	1-6	001-110
111	7（最高优先级）	111

与中断源的对应关系如下表：

PL2_0-PL0_0	外部中断 EInt0
PL2_1-PL0_1	外部中断 EInt1
PL2_2-PL0_2	外部中断 EInt2
PL2_3-PL0_3	外部中断 EInt3
PL2_4-PL0_4	外部中断 EInt4
PL2_5-PL0_5	外部中断 EInt5
PL2_6-PL0_6	外部中断 EInt6
PL2_7-PL0_7	外部中断 EInt7

11.1.1 内部中断处理

负责收集 I^2C、UART、SPI、TIMER、ADC、RTC 等功能模块产生的中断请求信号，并转换为对 CPU 的中断请求 INT1。

注意：TIMER2、TIMER3、ADC 功能模块没有提供相应的中断标志位，I^2C、UART、SPI、RTC 则在其模块内部有相应的中断标志位。

支持 8 个中断源输入，所有的中断源输入均为高电平有效。

每个中断源可分配一个相应的从 0-7 的中断优先级，每个优先级相应的中断向量值不能被重新编程，中断向量值即为对应的优先级数。

每个优先级的中断都可分别控制其使能与否，可屏蔽指定优先级及其以下的中断请求。

intc_i 输出高电平有效的中断请求，经过反相后，输出到 8051 的 INT1 中断请求端。

当 MCU 工作于 Sleep 模式，intc_i 仍然能够采集中断请求信号，产生中断请求信号，使 MCU

从 Sleep 模式唤醒。在掉电保持模式，由于 LDO 关闭 2.5V 电源输出，intc_i 和 I²C、UART、SPI、TIMER 等功能模块也停止工作。

1) ICR_I：中断控制寄存器

ICR_I（97H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	--	ME	MASK2	MASK1	MASK0

- ME：中断屏蔽使能位，缺省为 0；
  - 0：禁止中断屏蔽
  - 1：使能中断屏蔽
- MASK2-0：指定中断屏蔽的优先级范围，缺省为 0h；
  - 000：屏蔽 0 优先级的中断请求
  - 001：屏蔽 1-0 优先级的中断请求
  - 010：屏蔽 2-0 优先级的中断请求
  - 011：屏蔽 3-0 优先级的中断请求
  - 100：屏蔽 4-0 优先级的中断请求
  - 101：屏蔽 5-0 优先级的中断请求
  - 110：屏蔽 6-0 优先级的中断请求
  - 111：屏蔽 7-0 优先级的中断请求

2) ISR_I：中断状态寄存器，反映了输出给 CPU 的中断状态；

ISR_I（9AH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	--	--	--	INTR	--	VEC2	VEC1	VEC0

- INTR：中断请求标志位，缺省为 0；在中断请求信号无效后，该标志位也立即会复位为 0。
  - 0：没有产生中断给 CPU
  - 1：有中断请求输出给 CPU
- VEC2-0：输出给 CPU 的中断向量值，其数值即为输出到 CPU 的中断的优先级，这个数值会一直保持到新的中断请求发生，缺省为 0h；

3) IPR_I：未决中断寄存器

IPR_I（9BH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0

- IPx：未决中断标志位，缺省为 0；在中断请求信号无效后，该标志位也立即会复位为 0。
  - 1：表示至少有一个优先级为 x 的中断请求宣称
  - 0：表示没有或忽略所有的优先级为 x 的中断请求

4) IER_I：中断请求使能寄存器

IER_I	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

（9CH）	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0

- IEx：中断使能位，缺省为 0；

  1：表示与中断源 x 优先级相同的中断请求被允许；（建议：实际应用时对不同的中断源设置不同的优先级）

  0：表示对中断源 x 的中断请求是否被允许不产生影响。

5) 中断优先级控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IPLSR0_I (9DH)	--	PL2_1	PL1_1	PL0_1	--	PL2_0	PL1_0	PL0_0
IPLSR1_I (9EH)	--	PL2_3	PL1_3	PL0_3	--	PL2_2	PL1_2	PL0_2
IPLSR2_I (9FH)	--	PL2_5	PL1_5	PL0_5	--	PL2_4	PL1_4	PL0_4
IPLSR3_I (A1H)	--	PL2_7	PL1_7	PL0_7	--	PL2_6	PL1_6	PL0_6

- PL2_x-PL0_x：指明了中断源 x 的中断优先级，缺省为 0h，如下表所示：

PL2_x-PL0_x	优先级	中断向量 VEC2-0
000	0（最低优先级）	000
001-110	1-6	001-110
111	7（最高优先级）	111

与中断源的对应关系如下表：

PL2_0-PL0_0	未使用
PL2_1-PL0_1	I^2C 中断
PL2_2-PL0_2	SPI 中断
PL2_3-PL0_3	ADC 中断
PL2_4-PL0_4	未使用
PL2_5-PL0_5	T3 溢出中断
PL2_6-PL0_6	T2 溢出中断
PL2_7-PL0_7	RTC 中断

11.1.1 中断使用

SC9351 的中断系统设计相对复杂。以外部中断为例，前级（源识别阶段）有中断源识别记录控制，后级还有各自的优先级与屏蔽功能控制。很多应用中可以简化使用。

一种最直观的办法,只利用前级两个寄存器来控制,分别实现使能与记录标志的功能.此时后级相关寄存器可这样设置:ICR_E.3 置 0,所有级别中断都可以响应.另外 IER_E 设置成 8'hFF,使能所有前级进来的中断(高电平有效).而优先级寄存器可根据需要设置,如果都设置成最高级(7). 那么任何外部中断生效 IPR_E 寄存器最高位置 1(也只有这一位才可能出现 1).同时记录中断位,ISR_E 的 VEC2~0 置成 111.所以需要查询前级的 EINTF 寄存器来判断是那个源产生的中断.

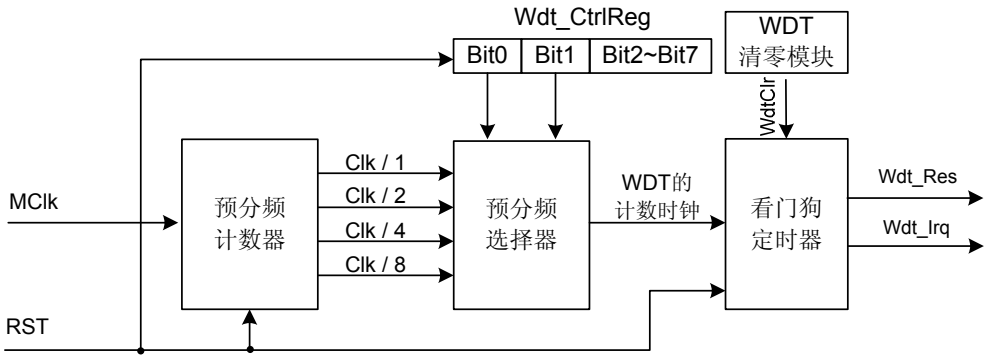
如果需要设置成不同的优先级,那么两个不同优先级的中断同时发生时,总是先响应高优先级中断(7--->0).如果再配合屏蔽寄存器 ICR_E,那么用起来就比较复杂了.比如 8 个外部中断源 EInt0~7 中断设置的优先级分别对应 7~0(默认顺序翻转).那么 EInt0 有中断来时(假设前后级已使 能),IPR_E.7 将置 1.其它类推.假设 ICR_E.2~0(mask2~0)设置成 011,那么 EInt4~7 进来的中断在后级就被屏蔽了.虽然不妨碍它们在前级置中断标志.

11. 外设功能描述

本节重点介绍 SC9351 内部集成的各模块，每个独立小节介绍各模块的功能，寄存器说明以及使用参考等。

11.1 WDT模块

11.1.1 功能说明



IP_WDT设计结构示意图

- WDT 时钟源与当前 CPU 工作时钟相同,
- 看门狗采用独立的计数器实现。看门狗（WDT）主要用于程序监控，在计数溢出后产生 WDT 复位信号，使 MCU 复位，避免进入死机等错误执行状态。
- WDT 计数器设计成 20 位，加上 3 位预分频位，共有 23 位。**WDT 在计数的 3/4 周期时产生预警中断**，溢出后产生复位信号。看门狗潜伏时间 ＝ 2^{20}/(wdtclk)，其中 wdtclk 由 WDT_CTRL 低两位选择。复位后默认潜伏时间约 175ms。
- SYS_STATUS 寄存器的 bit1 位记录看门狗复位标志，当产生看门狗复位后，该标志位为 1，对看门狗进行清零操作时，该标志位清零（同时清零的还有 bit0）。
- 在 Sleep 模式，WDT 时钟被关闭，WDT 不工作。

11.1.1 寄存器说明

WDT 时钟源选择寄存器（复位值：xxxxxx00B,可读写）

Wdt_CtrlReg

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
不使用						看门狗预分频设置	

WDT_CTRL（84H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	X	X	X	X	X	X	B1	B0

- Bit7-Bit2：不使用
- Bit1-Bit0：看门狗时钟源选择

  00：WdtClk=MClk

  01：WdtClk=MClk/2

  10：WdtClk=MClk/4

  11：WdtClk=MClk/8

Wdt_Clr0（地址：85H），Wdt_Clr1（地址：86H）：看门狗清零寄存器。

看门狗清零操作必须遵循以下步骤：

1 写 00H->Wdt_Clr0，写 00H->Wdt_Clr1；

2 写 53H->Wdt_Clr0，写 ACH->Wdt_Clr1。

注意：

- **看门狗一直处于开启状态，不能关闭。（程序中必须在看门狗定时周期内对清零寄存器进行特写操作，否则将产生看门狗溢出复位。不使用看门狗中断）。**
- 在调试模式（nDBG 接地），MCU 单步运行时 WDT 不工作，全速运行时，WDT 正常工作。

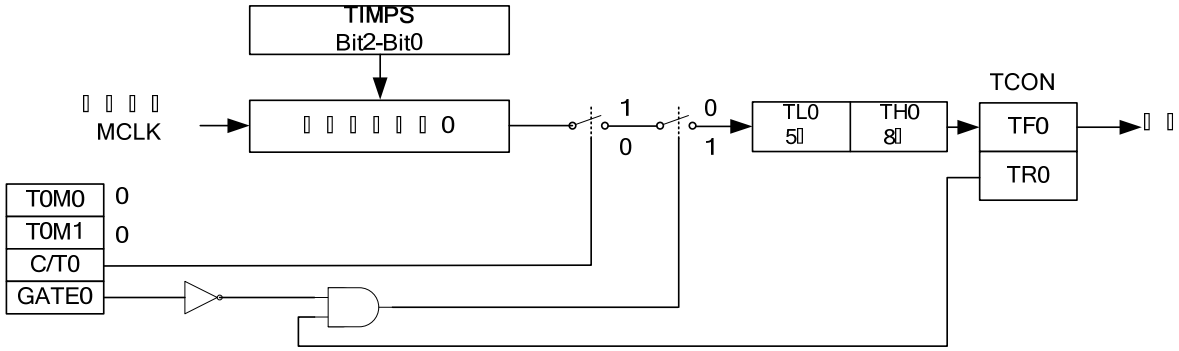
11.2 定时/计数器（T0/T1）

11.2.1 功能说明

定时器 0 和定时器 1 都是 16 位的寄存器，在被访问时以两个字节的形式出现。定时器 0 的低字节为 TL0，高字节为 TH0；定时器 1 的低字节为 TL1，高字节为 TH1。定时器 0 和定时器 1 各自有独立的时钟源，通过设置定时器时钟分频控制寄存器 **TIMPS** 对系统时钟分频而得。定时器控制寄存器 TCON 用于允许定时器 0 和定时器 1 以及指示它们的状态。通过将 IE 寄存器中 ET0 位置“1”来允许定时器 0 中断，通过将 ET1 位置“1”来允许定时器 1 中断。这两个定时器通过设置 TMOD 寄存器中的模式选择位 T0M1，T0M0（或 T1M1，T1M0）来选择工作方式，每个定时器都可以被独立配置。下面以定时器 0 为例介绍定时器的 4 种工作模式。

1） 工作模式 0

TMOD 寄存器中定时器 0 的方式选择位 T0M1，T0M0 设成 00B，定时器 0 将工作在模式 0。模式 0 将定时器 0 作为 13 位的定时器使用。下面给出了定时器 0 工作在模式 0 时的原理框图。



工作模式 0 的原理框图

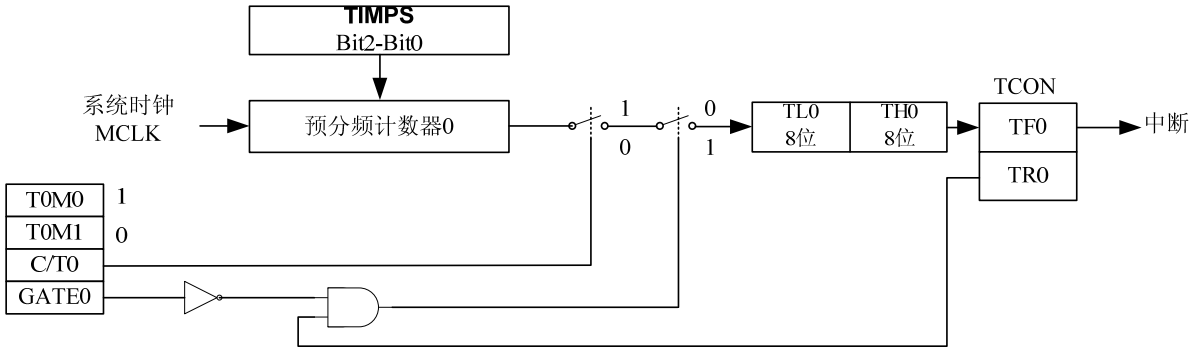
下面介绍对定时器 0 的配置和操作。

13 位定时器由 16 位定时器 0 的高 8 位和低 5 位组成，即 TH0 的 8 位和 TL0 的低 5 位。作为 13 位定时器寄存器，计到 1FFFH（全 1）后再计一次将发生溢出，使计数值回到 0000H，此时定时器溢出标志 TF0 被置位并产生一个中断（如果该中断被允许）。

该模式下要让定时器 0 工作，C/T0 和 GATE0 必须都置为‘0’，并将 TR0 置成‘1’。

2） 工作模式 1

TMOD 寄存器中定时器 0 的方式选择位 T0M1，T0M0 设成 01B，定时器 0 将工作在模式 1。与模式 0 不同的是定时器使用全部 16 位，控制方式与模式 0 相同。



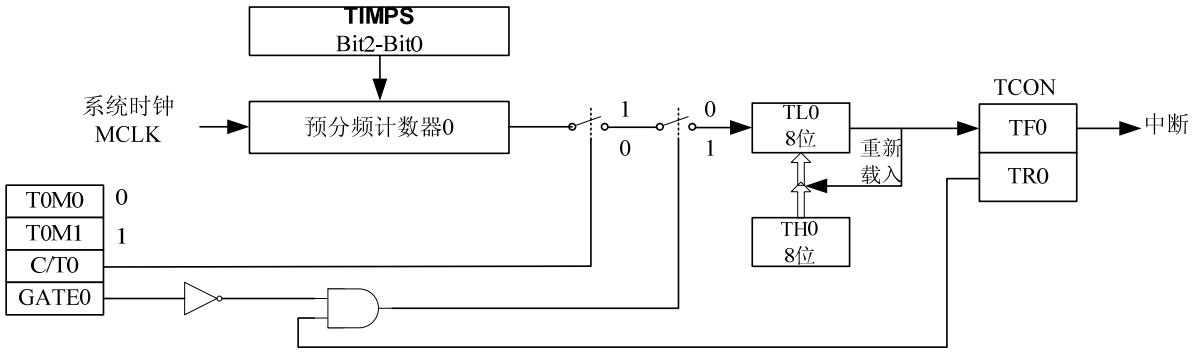
工作模式 1 的原理框图

3） 工作模式 2

TMOD 寄存器中定时器 0 的方式选择位 T0M1，T0M0 设成 10B，定时器 0 将工作在模式 2。

模式 2 将定时器 0 配置为具有自动重新装入计数初值能力的 8 位定时器（如图 13 所示）。TL0 做 8 位定时器，而 TH0 保持重载值。当 TL0 中的计数值发生溢出（从 FFH 到 00H）时，定时器溢出标志 TF0 被置位，TH0 中的重载值被重新装入到 TL0。如果中断被允许，在 TF0 被置位时将产生一个中断。TH0 中的重载值保持不变。为了保证第一次计数正确，必须在允许定时器之前将 TL0 初始化为所希望的计数值。

该模式下要让定时器 0 工作，C/T0 和 GATE0 必须都置为‘0’，并将 TR0 置成‘1’。



工作模式 2 的原理框图

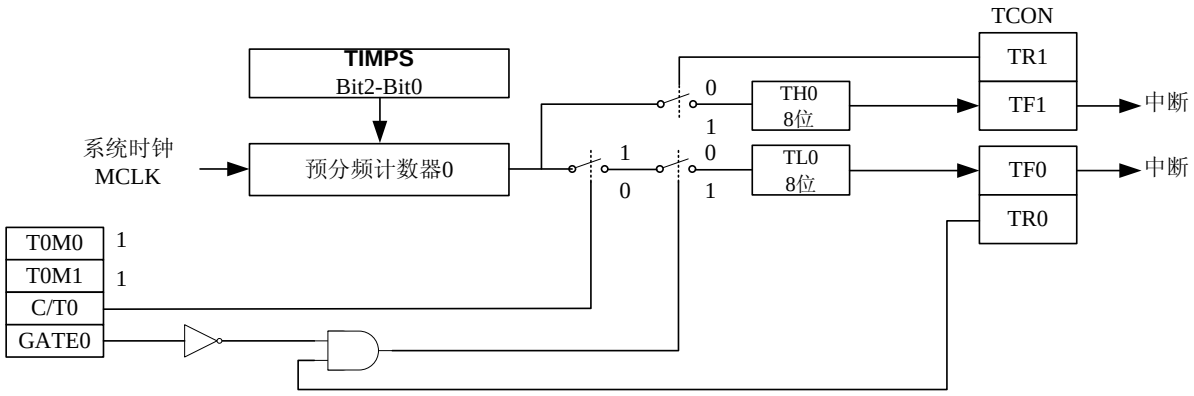
4） 工作模式 3

TMOD 寄存器中定时器 0 的方式选择位 T0M1，T0M0 设成 11B，定时器 0 将工作在模式 3。模式 3 将定时器 0 的 TL0 和 TH0 被分为两个独立的 8 位定时器。TL0 定时器使用 TCON 和 TMOD 中定时器 0 的控制/状态位；TR0，C/T0，GATE0 和 TF0。TH0 定时器使用定时器 1 的运行控制位 TR1，并在发生溢出时定时器 1 的溢出标志位 TF1 置‘1’，此时它控制定时器 1 的中断。

定时器 1 无工作模式 3 状态，若将 T1 设成模式 3，就会使 T1 立即停止计数。在定时器 0 工作在模式 3 时，定时器 1 仍可设置成模式 0，模式 1，模式 2，但不能置 TF1 和产生中断。

该模式下要让定时器 0 的 TL0 工作，C/T0 和 GATE0 必须都置为‘0’，并将 TR0 置成

‘1’，而要让 TH0 工作只需要将 TR1 置成‘1’即可。



工作模式 3 的原理框图

11.2.1 寄存器器说明

1) T0/T1 预分频控制寄存器 （复位值：x111x111B,可读写）

TIMPS
（8EH）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
--	prescaler_ctrl1<2:0>			--	prescaler_ctrl0<2:0>		

- B7、B3：不使用。
- B6~B4：T1 预分频选择（MClk 为 CPU 的工作时钟，有 12MHz、6MHz、 75KHz 三种选择）

  000：MClk/2
  001：MClk/4
  010：MClk/8
  011：MClk/16
  100：MClk/32
  101：MClk/64
  110：MClk/128
  111：MClk/256
- B2~B0：T0 的预分频选择位

  000：MClk/2
  001：MClk/4
  010：MClk/8
  011：MClk/16
  100：MClk/32
  101：MClk/64
  110：MClk/128
  111：MClk/256

2) T0/T1 工作模式控制寄存器（ 复位值：00000000B，可读写）

TMOD（89H）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
GATE	C/T	M1	M0	GATE	C/T	M1	M0
定时/计数器 1				定时/计数器 0			

- GATE：定时/计数器 x 选通控制位。
  0：只要 TRx 置 1，定时/计数器 x 就被选通。
  1：仅当 INT1/INT0 信号为高电平且 TRx 置 1 时，定时/计数器 x 才被选通工作。

注意：在 SC9351，INT0 信号接 IO 外部中断，INT1 信号接内部功能模块中断。

- C/T：定时/计数器 x 的定时、计数方式选择位。
  0：设置为定时器方式，由预分频时钟触发定时器。
  1：设置为计数器方式，由脉冲信号 Tx 触发计数器。

注意：在 SC9351 电路中，T0 固定接 0，T1 固定接 1，所以没有计数器方式。

- M1~M0：定时/计数器 x 工作模式控制位：
  00：模式 0，13 位计数模式。重载定时器初值时必须先关掉定时器 x(TRx=0)，装完后再打开定时器 x（TRx = 1）。
  01：模式 1，16 位计数模式。重载定时器初值时必须先关掉定时器 x(TRx=0)，装完后再打开定时器 x（TRx = 1）。
  10：模式 2，自动再装入的 8 位计数模式。
  11：模式 3，仅适用于定时器/计数器 0（定时器/计数器 1 无该工作方式）。

3) T0/T1 控制寄存器（复位值：00000000B，可读写）

TCON（88H）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TF1	TR1	TF0	TR0	IE1	IE0	IT1	IT0

- TF1：定时器/计数器 1 溢出中断请求标志位。
  0：当 MCU 响应中断，转向该中断服务程序执行时，硬件自动清 0（TF1=0）。
  1：当定时器/计数器 1 计数回 0 并产生溢出信号，硬件置位 TF1（TF1=1），向 MCU 请求中断。
- TR1：定时器/计数器 1 启动/停止控制位。
  0：停止（TR1=0）。
  1：可编程控制定时器/计数器 1 启动（TR1=1）。
- TF0：定时器/计数器 0 溢出中断请求标志位。
  0：当 MCU 响应中断，转向该中断服务程序执行时，硬件自动清 0（TF0=0）。
  1：当定时器/计数器 0 计数回 0 并产生溢出信号，硬件置位 TF0（TF0=1），向 MCU 请求中断。
- TR0：定时器/计数器 0 启动/停止控制位。
  0：停止。
  1：可编程控制定时器/计数器 0 启动

- IE1：外部中断 INT1 中断请求标志位。
  0：当 MCU 响应该中断执行中断服务程序时，硬件自动清 0。
  1：当 INT1 触发中断时，硬件自动置 1。
- IE0：外部中断 INT0 中断请求标志位。
  0：当 MCU 响应该中断执行中断服务程序时，硬件自动清 0。
  1：当 INT0 触发中断时，硬件自动置 1。
- IT1：
  0：外部中断 INT1 低电平触发中断请求。
  1：外部中断 INT1 下降沿触发中断请求。
- IT0：
  0：外部中断 INT0 低电平触发中断请求。
  1：外部中断 INT0 下降沿触发中断请求。

11.3 定时/计数器TIMER2

除实现正常的定时和**计数**功能外，还有 PWM 输出功能；

11.2.1 功能说明

两种工作模式：内部计数模式和 PWM 输出模式。

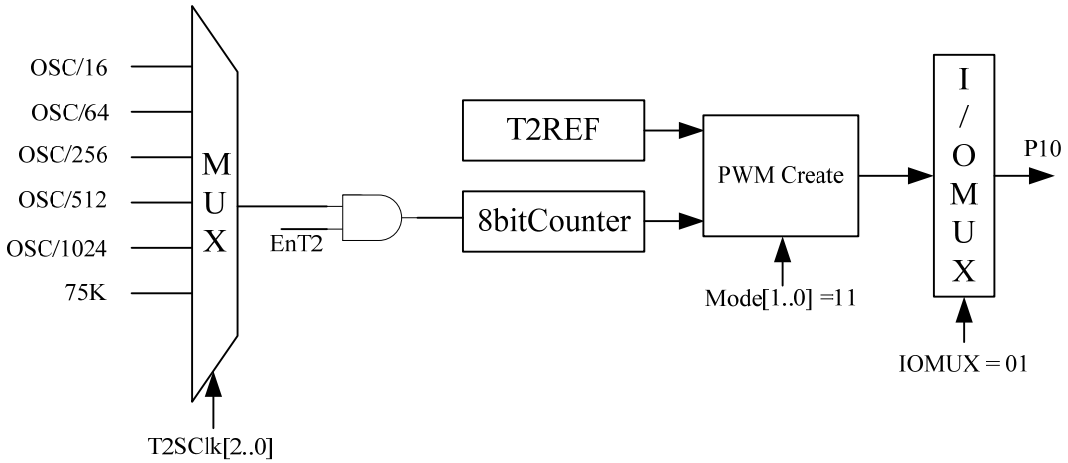
1）内部计数模式

内部计数模式必须通过设置 T2SClk[2..0]选择 OSC/16， OSC/64，OSC/256，OSC/512，OSC/1024，75k 时钟，并且将 Mode[1..0]设置成内部计数模式。T2Ref 为预设定时数值，当计数器计到与 T2Ref 值相等时产生溢出中断信号。

2）PWM 输出模式

若需 PWM 输出，要先在 **T2REF** 中设高电平持续宽度（即主时钟周期个数），则低电平的持续宽度就是 256-**T2REF**，然后将模式选择 Mode[1:0]设为 2'b11 即可；

通过设置 T2SClk[2..0]可选择 OSCIN/16，OSCIN/64，75K，OSCIN/256，OSCIN/512，OSCIN/1024，共六路时钟源。



PWM 输出模式原理图

T2 中断信号通过中断扩展模块intc_i向MCU发出中断请求，中断扩展的详细说明见“中断扩展intc_i”。

11.2.1 寄存器说明

1) T2 控制寄存器 （复位值：0x000000B，可读写）

T2CON（D5H）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EnT2	--	T2SClk[2]	T2SClk[1]	T2SClk[0]	—	Mode1	Mode0

- EnT2：T2 工作使能，缺省为 0
  - 1：T2 使能
  - 0：T2 禁止
- T2SClk[2:0]：8 路时钟源选择，缺省为 3’b000
  - 000：选择 OscIn(直接来自振荡，跟 CPU 工作时钟频率一致，有 12MHz/6MHz/75KHz 三种选择)的 16 分频
  - 001：选择 OscIn 的 64 分频
  - 010：选择 OscIn 的 256 分频
  - 011：选择 OscIn 的 512 分频
  - 100：选择 OscIn 的 1024 分频
  - 101：选择 75K 时钟
  - 11x：禁止使用
- Mode[1:0]: 工作模式选择，缺省为 0h
  - 00：不工作
  - 01：内部计数模式
  - 10： 保留模式，禁止使用
  - 11： PWM 输出模式

2) T2 预设寄存器 （复位值：11111111B，可读写）

T2REF（D6H）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

- 预设定时数值，**计数**器自加 1 到这个预设值后溢出，产生中断，然后从 1 开始重新**计数**

11.4 定时/计数器TIMER3

实现正常的定时和**计数**功能；

11.2.1 功能说明

只有一种工作模式，可以实现内部**计数**和外部**计数**功能；

通过设置 T3SClk[2..0]可选择 OSCIN/16，OSCIN/64，F75K，OSCIN/256， OSCIN/512，OSCIN/1024，共六路时钟源。

T3 中断信号通过中断扩展模块intc_i向MCU发出中断请求，中断扩展的详细说明见“中断扩

展intc_i”。

11.2.1 寄存器说明

1) T3 控制寄存器 （复位值：0x000xxxB，可读写）

T3CON(DAH)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EnT3	--	T3SClk[2]	T3SClk[1]	T3SClk[0]			

- EnT3：T3 工作使能，缺省为 0
  1：表示 T3 使能
  0：表示 T3 禁止
- T3SClk[2:0]: 8 路时钟源选择，缺省为 000
  000：选择 OscIn(直接来自振荡，跟 CPU 工作时钟频率一致，有 12MHz/6MHz/75KHz 三种选择)的 16 分频
  001：选择 OscIn 的 64 分频
  010：选择 OscIn 的 256 分频
  011：选择 OscIn 的 512 分频
  100：选择 OscIn 的 1024 分频
  101：选择 75K 时钟
  11x：禁止使用

2) T3 预设寄存器 （复位值：11111111B，可读写）

T3REF(DBH)

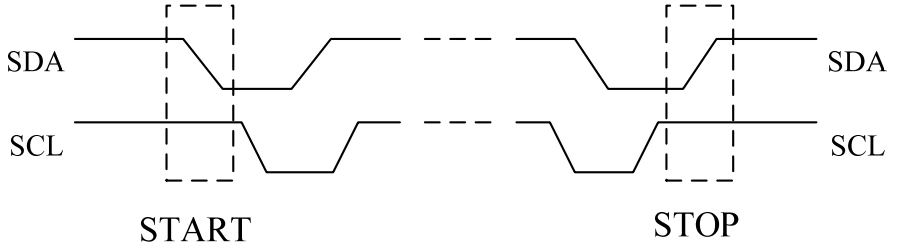
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

- 预设定时数值，**计数**器自加 1 到这个预设值后溢出，产生中断，然后从 1 开始重新**计数**。

11.5 I²C

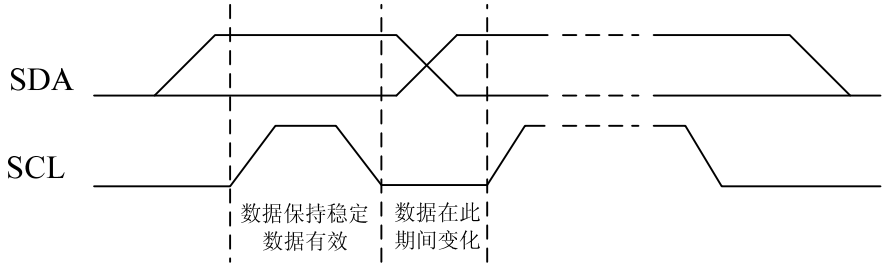
I2C 是由 Philips 开发的两线制双向串行通信协议。数据传输的启动和结束由主机控制，主机产生时钟来同步数据的传送。主机和从机都可以作为发送器和接收器。当主机对从机写数据时，主机为发送器（主发送），而从机则为接收器（从接收器）；当主机读从机数据时，主机为接收器（主接收），从机则为发送器（从发送器）。但是哪一种传送方式被激活是由主机确定的。

在总线空闲时，数据线（SDA）和时钟线（SCL）都保持高电平。当 SCL 为高电平而 SDA 产生下降沿时，总线产生启动信号并开始进行从机寻址及数据传输；当时 SCL 为高电平而 SDA 产生上升沿时，总线产生结束信号，总线将结束传输而保持空闲状态。见下图：



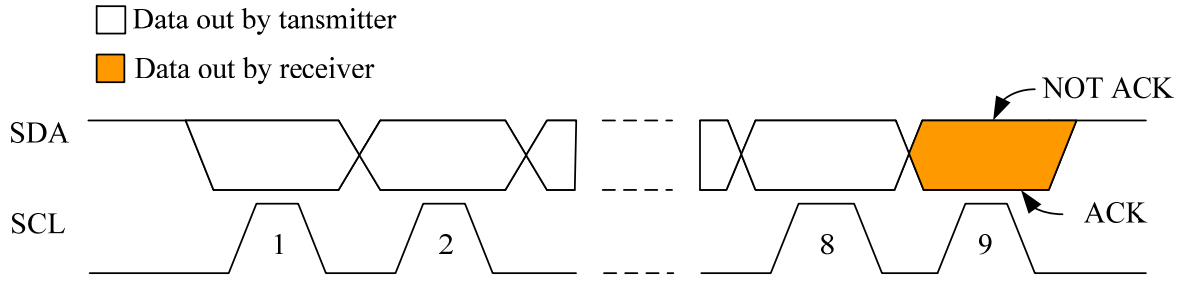
I2C 总线的启动(START)和停止(停止)信号

每个时钟脉冲传送一个数据，SDA 线上的数据在 SCL 为高电平时应保持稳定，否则 SDA 线上的数据将成为总线启动或停止的控制信号，使正在进行的数据传输错误结束或启动。见下图：



I2C 总线上的位传送

I2C 总线上所有数据的传输都必须以字节为单位，在启动状态和停止状态之间传输的字节数没有限制。当主机向从机发送数据时，一个字节数据传送完后，主机必须在 SCL 线产生一个与应答位对应的额外的时钟脉冲，并且放开 SDA 线，以便从机产生应答位（ACK）。如果从机寻址或数据接收成功，从机发送应答位，然后释放 SDA 线，以便主机继续进行数据传输；如果从机未对接收到的从机地址或数据作出应答，则主机将不再继续传输，此时从机放开 SDA 线，以便主机产生停止状态条件。当主机接收从机数据时，主机在接收完一个字节数据后，也需要发出一个应答信号。此时从机发送完一个字节数据后要放开 SDA 线，以便主机产生应答位。当主机接收完最后一个字节数据后，为了通知从机数据传输结束，主机不应答，从机收不到应答位就会放开 SDA 线，以便主机产生停止状态条件。见下图。



I2C 总线上一个字节的传输过程

SC9351 的 I²C 接口具有主从模式可配置，可 7 位寻址器件功能，但不支持多主机及其涉及的仲裁机制处理等。主要有四种工作方式：主发送方式，主接收方式，从接收方式。工作模式的详细说明请参考 I²C 总线协议。

I²C中断信号通过中断扩展模块intc_i向MCU发出中断请求，中断扩展的详细说明见“中断扩展intc_i”。

I²C的SDA、SCL与IO端口复用，控制说明见“端口说明”。

11.2.1 寄存器说明

1) I2C 控制寄存器 （复位值：x0000000B，可读写）

I2CCR（DFH）

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
--	MASTER_ ACKDT	MASTER_ RSTART	MASTER_ STOP	MASTER_ START	INT_FLAG	WAIT_ FLAG	MASTER_ SLAVE

- MASTER_ACKDT（bit6）：主机应答控制位，高有效

  当作为主接收器时，每接收完一个字节的数据(第一次进中断)，主机将通过操作该

位，来表示是否作出应答，该位为高有效，既为“1”时作出应答，否则不作应答。只能用在主机模式且处于接收方时才有意义。

- MASTER_RSTART（bit5）：I^2C 总线重启控制位，高有效

在主机模式时使用，在没产生停止信号之前使用 MASTER_RSTART 来重启 I2C，重新对从机寻址。

- MASTER_STOP（bit4）：I^2C 总线停止控制位，高有效

作为主机，在 I^2C 总线应答位无效后，写入该控制位可以使 I^2C 产生停止状态。只能用于主机模式。

- MASTER_START（bit3）：I^2C 总线启动控制位，高有效

作为主机，写入该控制位可以使 I^2C 从空闲状态产生启动信号。注意：只能用于主机模式。

- INT_FLAG（bit2）：I^2C 中断标志位，高有效

中断标志位，告知 MCU 发生中断事件。对该位清 0 方法如下：对 CONTROL_REG 寄存器进行写操作时，该中断标志位清零，只要是对控制寄存器写操作即可清 0。

- WAIT_FLAG（bit1）：I^2C 等待标志位，高有效

等待标志位与中断标志位同时产生，使 I^2C 总线进入等待状态，等待中断处理完成后再开始数据传输，具有暂停 SCL 时钟的功能，当进入等待后，要产生下一节拍的 SCL 时钟，必须先对该标志位清 0。清 0 方法如同 I^2_INT_FLAG 的清 0 方法，只需对控制寄存器写操作即可。

- MASTER_SLAVE（bit0）：主机/从机控制位

控制当前 I^2C 器件工作于主机模式还是从机模式，“1”为主机模式，“0”为从机模式。

2) I2C 状态寄存器 （复位值：00000000B，只读）

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSR （BFH）	MASTER_ RECEIVE_ ACKED	SLAVE_ ACKED	TRANS/ RECEIVE	RECEIVE_ ACK	RECEIVE_ OVER	BUF_FULL	STOP_FL AG	START_ FLAG

- MASTER_RECEIVE_ACKED（bit7）：主接收器应答有效标志位，高有效

  当作为主接收器，每接收完一个字节数据后，记录主接收器是否发出了有效的应答位（前一次中断发出的 MASTER_ACKDT）。若该标志位有效，则继续接受，否则则停止数据接收，产生停止状态。

- SLAVE_ACKED（bit6）：从接收器应答有效标志位，高有效

  当作为主发送器，每发送完一个字节数据后，记录从接收器是否发出了有效的应答位（从机发出有效的低电平 SDA）。若该标志位有效，则主发送方认为从机可以继续接收；否则，表示从机不能有效 ACK，如此主发送方可以停止数据传送，产生停止状态。

- TRANS/RECEIVE（bit5）：发送/接收标志位

  判断本机是发送器还是接收器，“1”表示本机为发送器，“0”表示本机为接收器。

- RECEIVE_ACK（bit4）：接收器应答标志位，高有效

  当作为接收器时，该标志位表示刚接收完的一个字节的数据，用以告知 MCU 中断程序作出相应应答。(用于接收器才有意义，表示已经接收到了新数据，用这位告知 MCU 请求作出相应反应)。

- RECEIVE_OVER（bit3）：接收溢出标志位，高有效

  当作为接收器，接收完一个字节的数据后，如果 BUF_FULL 标志位已被置高，即上一次接收放于缓存器 BUF 中的数据还未被读取，则该标志位将被置高，表示接收溢出，刚接收到的这个数据暂存在移位寄存器 PSR 中。当移位寄存器 PSR 中的数据被读取后，溢出标志位自动清零复位。

- BUF_FULL（bit2）：接收缓存器满标志位，高有效

  本机作为接收器，接收完一个字节数据后，将会置位该标志位，表示当前缓存器 BUF 中的数据有效。读取缓存器 BUF 的数据后，该标志位自动清零复位。

- I^2_STOP_FLAG（bit1）：I^2C 停止标志位，高有效

  I^2C 总线停止，处于空闲状态。

- I^2_START_FLAG（bit0）：I^2C 启动标志位，高有效

  I^2C 总线启动标志位。

3) 从机地址/波特率设置寄存器 （复位值：00000000B，可读写）

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSLA （C1H）								

该寄存器在主机模式或从机模式，有不同的定义。当作为从机时，为从机地址寄存器，用户可以设定从机地址，高七位有效。作为主机时，为波特率寄存器，它控制串行时钟线 SCL 的频率，从而确定 I^2C 总线数据传输的速率。注意：本设计的 SCL 时钟高低电平占空比是 1：2。

波特率计算公式如下：Fosc 为 MCU 工作时钟（12MHz/6MHz/75KHz）

$$波特率=\frac{1}{BAUD+BAUD/2}XFosc$$

4) 发送/接收缓冲器（复位后状态不定，可读写）

I2CBUF（C2H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

该寄存器是作为移位寄存器的缓存器，移位寄存器不可见，通过对缓存器读写来实现对移位寄存器数据的读取。当发送时，MCU 将数据写到 I2CBUF，转存到移位寄存器移位输出；当接收时，收到的数据经移位寄存器后转存到 I2CBUF。

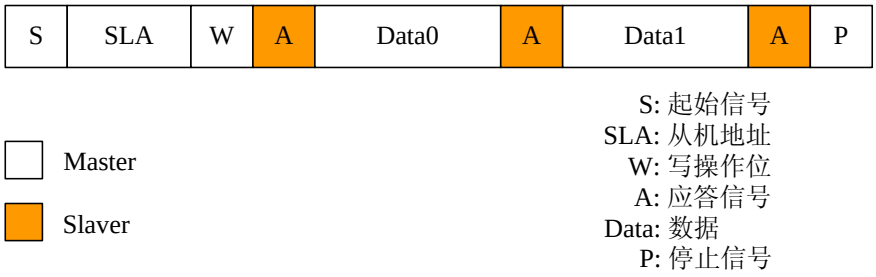
5) 数据接收二级缓冲器（复位后状态不定，只读）

I2CRXB（BEH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

与发送/接收缓冲器（I2CBUF）形成二级接收缓冲，当前一次收到的数据没被读走，则这个数据会被移到 I2CRXB 暂存，如此形成二级缓冲。

主发送方式

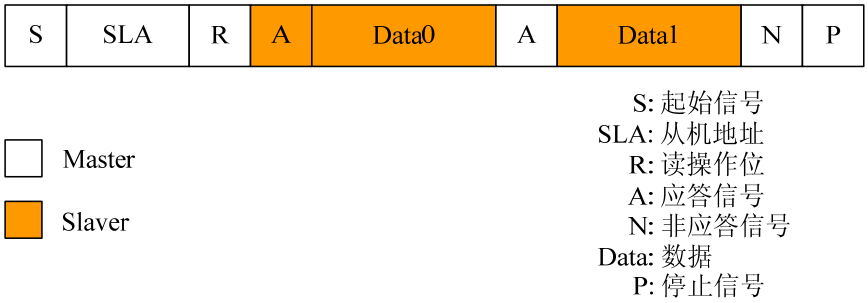
在 SDA 上发送串行数据，在 SCL 上输出串行时钟。I2C 接口首先产生一个起始信号，然后发送从机地址和写控制位组成一个字节数据。主发送器接着发送一个或多个字节的串行数据。在每发送一个字节后，从机发出确认位。当主机发出一个停止信号后，I2C 传输结束。



主机启动 I2C 总线后，从机地址和读写控制位依次在 SCL 的下降沿发送到 SDA 线上，在 SCL 第八个脉冲的下降沿，主机放开 SDA 线以便从机发出应答位，主机在 SCL 的第九个脉冲的下降沿采样 SDA 以确定从机是否作出应答（即是否产生 SLAVE_ACKED），同时主机亦置位中断标志位（INT_FLAG）和等待标志位（WAIT_FLAG），主机拉低 SCL，放开 SDA，使 I2C 处于等待状态。然后读 SLAVE_ACKED 判断接收的应答信号是否有效，如果从机地址应答有效则向 **I2CBUF** 写入数据，清除 WAIT_FLAG 标志位同时释放了 SCL，主机依此过程继续发送下一字节数据，直到数据传输完成，最后置位 I2C 结束标志位（MASTER_STOP），主机将产生停止信号，I2C 传输结束。如果从机地址应答无效，则主机同样发送停止信号结束 I2C 传输。

主接收方式

在 SDA 上接收串行数据，在 SCL 上输出串行时钟。I2C 接口首先产生一个起始信号，然后发送从机地址和读控制位组成一个字节数据。接着从 SDA 接收来自从机的串行数据并在 SCL 上输出串行时钟。从机发送一个或多个字节的串行数据。每收到一个字节后，主机发送应答信号。在接收完最后一个字节后发送非应答信号。然后主机再产生一个停止信号，结束 I2C 传输。



主机在对从机寻址并发出高电平的读写控制位，通知从机作为发送器发送数据。从机在 SCL 时钟脉冲的下降沿将数据发送到 SDA 线上，主机则在 SCL 时钟脉冲的上升沿锁存 SDA 线上的数据（此时主机是接收）。在 SCL 的第八个时钟脉冲的下降沿，主机置位 INT_FLAG 和 WAIT_FLAG,使 I2C 总线等待主机决定对接收到的数据是否作出应答。

如主机要作出应答，则置位主机应答控制位（RECEIVE_ACK），否则就不置位 RECEIVE_ACK。然后清除 INT_FLAG 和 WAIT_FLAG，结束 I2C 的等待状态，然后主机发出应答位和第九个 SCL 时钟脉冲。

在 SCL 的第九个时钟脉冲的下降沿主机监测应答位，并再次置位 INT_FLAG 和 WAIT_FLAG，使 I2C 总线处于等待状态。然后判断主接收器应答有效标志位（MASTER_RECEIVE_ACKED）是否有效。若应答有效，清除 INT_FLAG 和 WAIT_FLAG，结束 I2C 的等待状态，继续接收数据。若应答无效，主机则置位 MASTER_STOP，然后清除 INT_FLAG 和 WAIT_FLAG，结束 I2C 的等待状态。主机产生停止状态，结束 I2C 数据传输。

从发送方式

从接收方式

11.6 UART0

Reg	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UART_BUF0(C3H)								
SCON0(C4H)	SM1	SM0	SM2	REN	TB8	RB8	TI0	RI0
BRCON0(C5H)	--	--	RXD_EN	TXD_EN	UartClkEn	SerialMask	TimerEn	SMOD
BRTIMER0(C6H)								

1) 接收发送缓冲寄存器 UART_BUF0（复位后状态不定，可读写）

8 位接收和发送的缓冲寄存器，发送模式时将要发送的数据写入该寄存器；接收模式时从该寄存器读取接收到的数据。

2) UART0 控制寄存器 SCON0（复位值：00100000B，可读写）

各控制位定义如下：

- SM0~SM1：UART 工作模式选择信号，可以选择：

  1：8 位模式异步通讯模式，波特率可调；

2：9 位模式异步通讯模式，波特率固定（fosc/16 或 fosc/32）；

3：9 位模式异步通讯模式，波特率可调。

- SM2：工作模式 2 和 3 的多机通讯控制位。在方式 2 或方式 3 时，如 SM2=l，REN=1，则从机只有接收到第九位 RB8=l 才会激发中断请求标志位 RI=1，向主机请求中断处理；如 SM2=0，REN=1，则接收到任意第九位，都会激发中断请求标志位 RI=1。在方式 l 时，如 SM2=1，REN=1，只有在接收到停止位为 1 时才置位中断请求标志位 RI=l，SM2=0，REN=1，接收到任意停止位，都会　　产生中断请求。
- REN：允许 / 禁止串行接收控制位。由软件置位 REN＝1 为允许串行接收状态，可启动串行接收器 RXD，开始接收信息。软件复位 REN＝0，则禁止接收。
- TB8：在方式 2 或方式 3，它为要发送的第 9 位数据，可以由软件置位或清 0；例如，可用作数据的校验位或多机通讯中表示地址帧 / 数据帧的标志位。
- RB8：在方式 2 或方式 3，是接收到的第 9 位数据。在方式 1，若 SM2=0，则 RB8 是接收到的停止位。
- TI：发送中断请求标志位。在方式 0，当串行发送数据第 8 位结束时，由内部硬件自动置位 TI＝l，向主机请求中断，响应中断后必须用软件复位 TI。在其他方式中，则在停止位开始发送时由内部硬件置位，必须用软件复位（写零复位）。
- RI：接收中断请求标志位，接收到停止位时由内部硬件置位 RI＝1(例外情况见 SM2 说明)，必须由软件复位 RI＝0（写零复位）。

3)　波特率控制寄存器 BRCON0 （复位值：xx000101B，可读写）

- RXD_EN：接收数据使能，为 1 时允许接收数据，为 0 时则为通用 IO，缺省为 0；
- TXD_EN：发送数据使能，为 1 时允许发送数据，为 0 时则为通用 IO，缺省为 0；
- UartClkEn：UART 工作时钟使能位。UART 不工作时，设置 UartClkEn 为 0，关闭 UART 时钟；UART 工作时设置 UartClkEn 为 1，打开 UART 时钟。UartClkEn 缺省为 0。
- SerialMask：UART 中断使能信号，低有效，缺省为 0。
- TimerEn： BRTIMER 使能信号，当 UART 在工作方式 1 或方式 3，设置 TimerEn 为 1，产生 UART 发送接收的时钟。
- SMOD：波特率选择位，SMOD 为 1 时的波特率是 SMOD 为 0 时的 2 倍。

4)　波特率设置寄存器 BRTIMER0 （复位后状态不定，可读写）

BR_TIMER 是一个 8 位定时器，自加 1 溢出。给 BR_TIMER 设定初始值后，BR_TIMER 加 1 到 FFH 时，则定时器溢出，产生一个溢出脉冲。每次溢出后定时器会自动载入初始值，重新加 1 计数。工作方式 1 或方式 3，UART 模块使用该溢出率时钟作为时钟源，使 UART 的波特率可以有较大的调整范围。

方式 1 与方式 3 的波特率：（BR_TIMER 的计数时钟为 Fosc/2，Fosc 为主晶振工作时钟）

$$\text{波特率}=\frac{2^{SMOD}}{32}\times\frac{Fosc/2}{0x100-BRreg}$$

方式 2 的波特率：

$$\text{波特率}=\frac{2^{SMOD}}{32}X(Fosc/2)$$

11.7 UART1

串口 1 的 功能与串口 0 功能完全相同。 IO 脚复用如下：

- 发送端 TX 与 P9.3 脚（PIN43）复用
- 接收端 RX 与 P9.4 脚（PIN44）复用

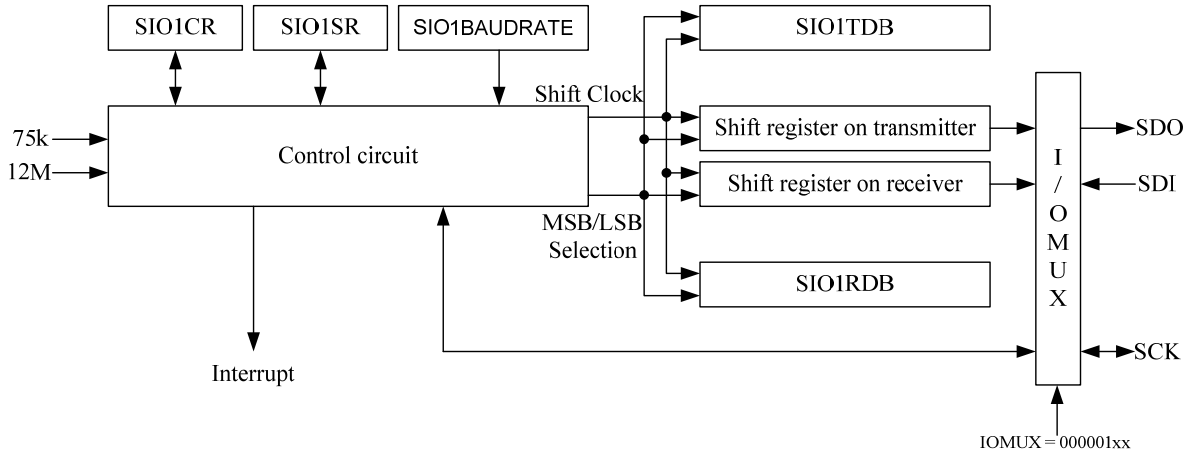
串口 1 占用的寄存器资源如下：

Reg	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UART_BUF1 (C7H)								
SCON1(C9H)	SM1	SM0	SM2	REN	TB8	RB8	TI1	RI1
BRCON1 (CEH)	--	--	RXD_EN	TXD_EN	UartClkEn	SerialMask	TimerEn	SMOD
BRTIMER1 (CFH)								

上表各寄存器的定义和工作模式说明和UART0 相同，见“UART0”。

11.8 SPI

SPI 采用三线制传输方式，分别为 SCK（时钟线双向） SDI（数据输出口） 和 SDO（数据输入口），支持单工，半双工，全双工传输模式；时钟 SCK 可选择内部产生或外部输入。



SPI 内部结构

11.2.1 寄存器说明：

1) SPI 控制寄存器（复位值：00000100 B，可读写）

SPICR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
（B1H）	SIOS	SIONH	SIOM		SIODIR	Fast_Slow_Clock	Internal_External	

- SIOS：SIO 使能控制位，缺省为 0；
  1：传输允许，
  0：传输结束；
- SIONH：强行复位控制位，
  0：缺省；
  1：强行复位 SIOS、**SPISR**、**SPITXB**、**SPIRXB**

  SIONH 具有强行复位功能，能复位 SIOS、**SPISR**、**SPITXB**、**SPIRXB** 到初始化设置，当用户认为运行处于死机时，可以编程设置 SIONH=1，使 SPI 复位到一个初始状态；
- SIOM：工作模式控制位，缺省为 0h；
  00：发送模式
  01：接收模式
  10：发送/接收

  SIOM 传输模式应当使传输双方匹配，否则不能正常通信；
- SIODIR：传输控制位，缺省为 0；
  1：LSB 先传输
  0：MSB 先传输

  SIODIR 决定数据传输时的方向，传输双方应当匹配；
- Fast_Slow_Clock：内时钟选择位，缺省为 1；
  1：在内时钟模式，波特率分频时钟选择 12MHz 晶振时钟
  0：在内时钟模式，波特率分频时钟选择 75KHz 晶振时钟
- Internal_External：传输时钟控制位，缺省为 0h；
  11：时钟由传输的另一方提供，由 SCK 脚引入，即工作在外时钟模式；
  00、01、10：为内时钟模式，由内部时钟分频产生传输时钟，并由 SCK 脚输出给另一方。

  Fast_Slow_Clock 要与 Internal_External 配合使用，当选择外时钟时，Fast_Slow_Clock 不起作用；当选择内时钟时，Fast_Slow_Clock 决定 12MHz 时钟/75KHz 晶振时钟作为波特率分频时钟源。

2) SPISR 状态寄存器（复位值：0x1000xxB）

SPISR（B2H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SIOF	--	TXF	RXF	TXERR	RXERR	--	--

- SIOF：传输进行标志位，只读

  SIOF=1 表征传输正在进行，当 SIOS 启动后，由 SCK 的第 0 个下降沿触发跳高，当由 SIOS 下跳正常结束 SPI 传输时，SIOF 在 SCK 的第 7 个上升沿跳低。
- TXF：发送缓冲器空标志位，只读

  TXF=1 表示发送缓冲器 SPITXB 中无待发数据，当 MCU 向其写入数据，则 TXF 跳低，TXF 为 0 时，向 SPITXB 写数据无效；移位时钟的第 0 个上升沿使 TXF 重新跳高，表征 SPITXB 中的数据已经**装载**进移位寄存器中。

- RXF：接收缓冲器满标志位，只读

  RXF=1 表示接收缓冲器 **SPIRXB** 收到新数据，若不及时读走该缓冲区数据，将会发生接收错误 RXERR，覆盖以前接收的数据；MCU 读取 **SPIRXB**，将使其回 0；在内时钟接收或内时钟发送/接收模式时，MCU 读取 **SPIRXB** 用来启动下一轮的数据接收，否则一直处于 Automatic wait 等待。

- TXERR：发送错误标志位，可读写

  TXERR=1 表示发生发送错误，当处于外时钟发送模式和外时钟接收/发送模式时，当

  SPITXB 缓冲中无数据，此时继续发送数据被认为是错误的。当 TXERR 出现错误，MCU 可以对该标志位清 0。

- RXERR：接收错误标志位，可读写

  RXERR=1 表示接收发生错误，当 **SPIRXB** 有新接收数据没有被读走，又接收到新的数据，则会发生覆盖前面数据的现象，协议认为发生接收错误。当 RXERR 出现错误，MCU 可以对该标志位清 0。

3) 发送缓冲器 （复位后状态不定，只写）

SPITXB （B3H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

发送缓冲器只写，不可读。

该缓冲器由 MCU 对其写入待发数据。SCK 的第 0 个下降沿，将 SPITXB 中数据装载进移位寄存器，同时发出第一个数据（MSB 或 LSB）。

注：该寄存器逻辑地址与 SPIRXB 相同，对应 SFR 列表中的 SPIBUF 寄存器。为了使用方便用不同名字加以区分。

4) 接收缓冲器（复位后状态不定，只读）

SPIRXB （B3H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

接收缓冲器只读，不可写。

由 MCU 读取该缓冲器数据。在 SCK 的第 7 个上升沿（接收完最后一个数据 MSB 或 LSB），将移位寄存器数据装载进 SPIRXB 中。

发送缓冲器与接收缓冲器逻辑上共用同一地址，但是物理上是两个独立的 8 位缓冲器。

5) 波特率设置寄存器（复位值：11111111B，可读写）

SPIBR （B4H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	不能设置为 00h							

为了适应传输速率需要，内部时钟电路模块负责分频产生波特率时钟，该 8 位计数器可设置范围从 1 到 255(不能设置为 0)，假设该波特率寄存器预置值为 N，其波特率计算方法如下：

Fclk_div = 1/[2（N+1）] Fclk_fc_fs_mux

Fclk_div：波特率时钟；

Fclk_fc_fs_mux：由 Fast_Slow_Clock 选择的 12MHz 时钟/75KHz 时钟；

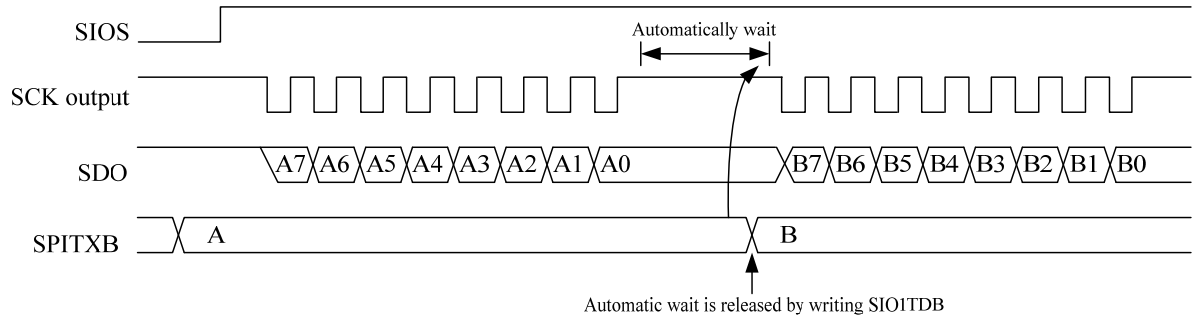
串行时钟 SCK

1）时钟源

通过设置 **SPICR** <Internal_External>，SCK 可以选择内部产生或外部输入。只有 SPI 处在停止状态时，时钟选择设置才有效。

（1）内部时钟

通过设置 **SPICR**< Fast_Slow_Clock >选择分频器的时钟源 75k 或 12M，再设置 SIO1BAUDRATE 产生一定频率的 SCK 信号。当一个字节传输完毕后，SCK 保持高电平，直到写新发送数据或读取完接收数据后，SCK 才继续产生时钟。

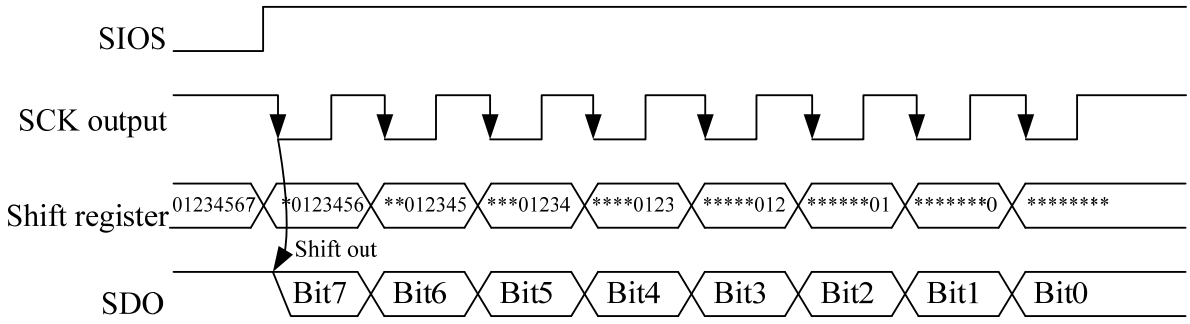


自动等待功能

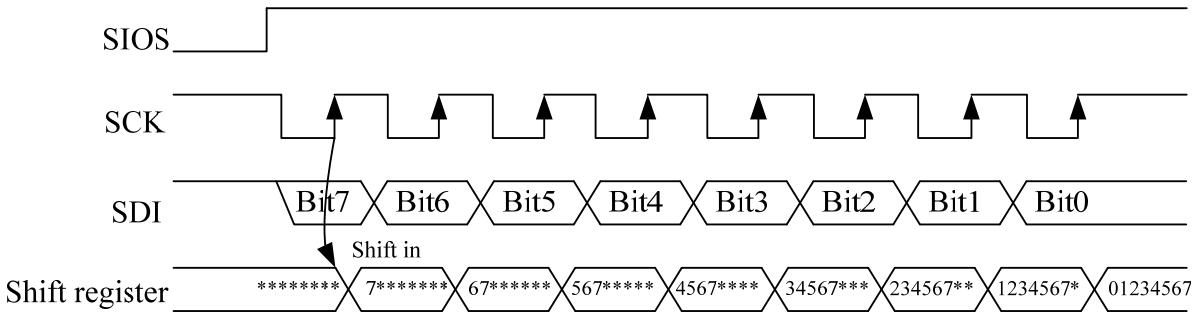
（2）外部时钟

2）数据移位

当 SCK 下降沿时，SDO 数据输出，当 SCK 上升沿时，SDI 数据输入。



输出时序（举例 MSB 传输）



输入时序（举例 MSB 传输）

传输模式

通过设置 **SPICR**< SIOM >，可将 SPI 设置成为发送模式，接收模式和发送接收模式。

1）发送模式

（1）启动发送

写‘00’到 **SPICR**< SIOM >，SPI 将设置成为发送模式。

当一个发送数据被写到发送缓冲寄存器 **SPITXB** 时，**SPISR**<TXF>清零。

然后 **SPISR**<SIOS>置 1，**SPISR**<SIOF>在 SCK 下降沿时置 1。在 SCK 下降沿时，发送数据位被发送到 SDO。SCK 的第 0 个上升沿使 **SPISR**<TXF>重新置高，表示 **SPITXB** 中的数据已经导入到发送移位寄存器。

（2）发送过程

当发送数据写入 SPITXB 时，**SPISR**<TXF>被清零。

在内部时钟模式时，如果新的发送数据没有写入到 **SPITXB**，SCK 保持高电平。

在外部时钟模式时，**SPISR**<TXF>置 1 后，新的发送数据必须在发送新的数据之前写入发送缓冲区中。如果在发送新的数据之前发送缓冲区没有更新，**SPISR**<TXERR>会被置 1。

（3）结束发送

有两种方式可以结束发送操作。

第一种方式：**SPICR**<SIOS>写 0，发送操作会在数据发送完后结束。当发送操作结束后，**SPISR**<SIOF>被清零，SDO 保持高电平。

第二种方式：**SPICR**<SIOINH>写 1，发送操作将立即结束。SIOS、**SPISR**、**SPITXB**、**SPIRXB** 恢复到初始状态。

2）接收模式

（1）启动接收

写‘01’到 **SPICR**< SIOM >，SPI 将设置成为接收模式。

然后 **SPISR**<SIOS>置 1，**SPISR**<SIOF>在 SCK 下降沿时置 1。在 SCK 上升沿时，将 SDI 数据移位到移位寄存器中。

当 8bit 数据被接收完后，数据从移位寄存器传到 **SPIRXB** 中，同时 **SPISR**<RXF>置 1。

（2）接收过程

读取 **SPIRXB** 数据后，**SPISR**<RXF>清零。

在内部时钟模式时，如果数据没被读走，SCK 保持高电平。

在外部时钟模式时，在 **SPISR**<RXF>置 1 后，必须在接收完新的数据之前读取 **SPIRXB** 中的数据。如果在接收完新的数据后没有读取 **SPIRXB**，**SPISR**<RXERR>会被置 1。

（3）停止接收

有两种方式可以结束接收操作。

第一种方式：**SPICR**<SIOS>写 0，接收操作会在数据接收完后结束。当接收操作结束后，**SPISR**<SIOF>被清零。

第二种方式：**SPICR**<SIOINH>写 1，接收操作将立即结束。SIOS、**SPISR**、**SPITXB**、**SPIRXB** 恢复到初始状态。

3）发送/接收模式

（1）启动发送/接收

写‘10’到 **SPICR**< SIOM >，SPI 将设置成为发送/接收模式。

当一个发送数据被写到发送缓冲寄存器 **SPITXB** 时，**SPISR**<TXF>清零。

然后 **SPISR**<SIOS>置 1，**SPISR**<SIOF>在 SCK 下降沿时置 1。在 SCK 下降沿时，发送数据位被发送到 SDO。同时在 SCK 上升沿时，将 SDI 数据移位到移位寄存器中。**SPISR**<TXF>置 1 时表示 **SPITXB** 的数据已经传到发送移位寄存器中，**SPISR**<RXF>置 1 时表示 8 位数据已经接收完成，并从接收移位寄存器传到 **SPIRXB** 中。

（2）发送/接收过程

当发送数据写入 **SPITXB** 时，**SPISR**<TXF>被清零。读取 **SPIRXB** 数据后，**SPISR**<RXF>清零。

在内部时钟模式时，以下几种情况会使 SCK 在一字节数据传输完成后保持高电平。

1. 在读取 **SPIRXB** 后，下一个发送数据没有被写到 **SPITXB** 中
2. 在下一个发送数据发送写到 **SPITXB** 后，没有读取 **SPIRXB**
3. 在传输完成后，下一个发送数据没写到 **SPITXB** 中，且也没读取 **SPIRXB**。

在外部时钟模式时，必须在传输下一个数据之前，读取 **SPIRXB** 并将下一个发送数据写到 **SPITXB** 中。在传输完一个数据后，如果没有读取 **SPIRXB**，**SPISR**<RXERR>会被置 1；如果没有将下一个发送数据写到 **SPITXB**，**SPISR**<TXERR>会被置 1。

（3）结束发送/接收

有两种方式可以结束发送/接收操作。

第一种方式：**SPICR**<SIOS>写 0，发送/接收操作会在数据传输完后结束。当发送/接收操作结束后，**SPISR**<SIOF>被清零。

第二种方式：**SPICR**<SIOINH>写 1，发送/接收操作将立即结束。SIOS、**SPISR**、**SPITXB**、**SPIRXB** 恢复到初始状态。

11.9 ADC

逐次逼近型 8bits 8 通道 AD 模块，3 通道，多个转速速率可选。

11.2.1 寄存器说明

1) ADC 控制寄存器（复位值：xx000000B）

ADCON (D2H)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
--	--	SEL_VREF	ADSTR	ADEN	SEL_CLK[1：0]		EOC

- SEL_VREF：参考电压选择位，缺省为 0；
  0：选择内部参考电平
  1：选择外部电源(VDD)
- ADSTR：AD 启动控制位，缺省为 0；置 1，ADC 开始工作。
- ADEN：AD 输入时钟使能位，缺省为 0；ENABLE =1，允许时钟输入。
- SEL_CLK [1:0]：AD 转换时钟选择位，缺省为 0h
  00：75KHz
  01：选择主时钟（12MHz/6MHz/75KHz）的 8 分频
  10：选择主时钟（12MHz/6MHz/75KHz）的 16 分频
  11：选择主时钟（12MHz/6MHz/75KHz）的 32 分频
- EOC：AD 是否转化完成标志位（只读，仅有一个转化时钟的长度，AD 转换周期为 11 个时钟），因为仅有一个转化时钟的长度，所以通过查询方式不一定读的到，最好使用 AD 中断进行操作。

2) ADC 结果数据寄存器（复位后状态不定，只读）

ADATA (D1H)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DATA[7：0]							

注意：ADC 模块产生中断时 AD 转换结果还没有进入结果寄存器，所以进入 AD 中断服务程序后，不能立即读取 AD 转换数据，应在检测到 AD 中断标志位变低后（或者等待一段时间），再读取转换数据。

3) AD 输入通道选择寄存器（复位值：xxxxx000B，可写）

ADCIS (D3H)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
--	--	--	--	--	SEL_VIN[2：0]		

- SEL_VIN：000：选择通道 AN0
  001：选择通道 AN1
  010：选择通道 AN2
  其它：禁止使用

11.10 FLASH编程接口

11.2.1 功能说明

集成 64kx8 的FLASH模块，提供批量烧录接口、内部编程接口（支持Debug模式）。批量烧录接口在“FLASH烧录模式”已有描述，这里主要描述内部编程接口。

根据程序对编程接口的设置，编程接口可以产生 flash 的烧录、擦除控制信号，完成对 flash 的单字节烧录、多字节连续烧录、块擦除、整片擦除等功能。

11.2.1 编程接口的控制寄存器说明如下：

1) 编程单元地址寄存器、数据寄存器：

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FSHWRADRH（A5H）	FSHWRADRH[7：0]							
FSHWRADRL（A6H）	FSHWRADRL[7：0]							
FSHWRDATA（A7H）	FSHWRDATA[7：0]							

- FSHWRADRH：编程单元的高 8 位地址；
- FSHWRADRL：编程单元的低 8 位地址；
- FSHWRDATA：编程数据。

2) FLASH 编程控制寄存器 1：

FSHWRCON1（A9H）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	WREN	WR	WRIF	WRERR	WRSQU	WRSEQBIF	WRIFO	WRSQUSTOP

- WREN：FLASH 编程使能位。
  - 0：禁止对 FLASH 编程，当出现复位或者编程操作完成时，该位由硬件清零，对此位写零无效。
  - 1：允许对 FLASH 编程，该位只能写 1，当对 FLASH 编程操作完成时，自动硬件清零。
- WR：FLASH 编程启动位。
  - 1：开始 FLASH 编程操作，在编程过程中此位为 1，编程操作完成硬件自动清零。
- WRIF：编程操作完成标志，只能软件清零，当连续编程或单字节编程操作完成时，该位由硬件自动置 1，软件写 1 无效。一般在启动一次编程操作前应该首先对该位清零，然后启动编程操作，循环检测是否为 1，如为 1，表示编程操作已经完成，可以进行下一次编程操作。
- WRERR：编程出错标志位，可以软件置 1、清 0。在编程过程中，如果出现了复位，则编程出错，复位信号对该标志位置位为 1；如果连续烧写时间超过了 8ms，则硬件会自动置 1，表示编程出错。当系统复位时，该标志位置位为 1。
- WRIFO：主存储区、信息存储区的编程选择位。

0：对信息存储区编程。

1：对主存储区编程。

- WRSQU：多字节连续编程、单字节编程选择位。

0：单字节编程。

1：多字节连续编程，可以连续编程 64 个字节数据，而不需要重新启动编程操作；

- WRSEQBIF：在多字节连续编程方式，如果一个字节数据编程结束，该位由硬件自动置 1，表示 1 个字节数据编程完成，可以进行下一个字节的编程。软件写 1 无效，软件清 0。
- WRSQUSTOP：编程操作停止控制位。

1：编程操作将停止，当编程总线停止时序完成后，硬件自动对该位清 0，软件写 0 无效。

3) FLASH 编程控制寄存器 2：

FSHWRCON2（AAH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	FSHWRCON2[7：0]							

FLASH 编程操作的安全控制寄存器，物理上不存在，读此寄存器值为 0。只有当 WREN=1 时，向该寄存器对应的地址，依次写 55h、AAh，才能进行后面的编程操作。

4) FLASH 擦除控制寄存器 1：

FSHERSCON1（ABH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	ERSEN	ERS	ERSIF	ERSERR	ERSPG	--	--	--

- ERSEN：擦除使能位，ERSEN=1，允许擦除 FLASH，该位只能写 1；当出现复位或擦除操作完成时，硬件自动清零，软件对此位写 0 无效。
- ERS：FLASH 擦除启动位，ERS=1，开始擦除操作，在擦除过程中，此位保持为 1；擦除操作完成后硬件自动清零，软件对此位写 0 无效。
- ERSIF：擦除完成标志位，ERSIF=1，表示擦除完成。该位只能软件清零，当擦除完成时，该位由硬件自动置 1，软件写 1 无效。
- ERSERR：擦除出错标志位，当执行擦除操作时，如果出现了复位，则该位由硬件自动置 1，表示擦除出错。当系统复位时，该标志位置位为 1。
- ERSPG：擦除方式控制位。

0：PAGE 擦除方式，对某一页（512 字节）进行擦除，所擦除的页由 FSHWRADRH 的高位确定。

1：MASS 擦除方式，对整个 64K 的 FLASH 进行擦除。

5) FLASH 擦除控制寄存器 2：

FSHERSCON2（ACH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	FSHERSCON2[7：0]							

用于擦除的安全控制寄存器，物理上不存在，读此寄存器值为 0。只有当 ERSEN=1 时，向该寄存器对应的地址，依次写 55h 和 AAh，才能进行后面的擦除操作。

6) FLASH 编程、擦除时钟分频控制寄存器：

FSHTIMER（ADH）	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	FSHTIMER[7：0]							

针对不同的主晶振时钟频率，设置不同的分频值，对主晶振时钟进行分频，产生频率等于1MHz 的时钟（用于 FLASH 编程、擦除的时序控制）。

- FLASH 擦除流程说明：
  - 根据主晶振时钟频率，向寄存器 FSHTIMER 写入分频值；
  - 设置擦除地址，如果为页擦除，写寄存器 FSHWRADRH；
  - 写寄存器 FSHERSCON1，设置 ERSEN=1，ERSIF=0；
  - 写寄存器 FSHERSCON2=55h；
  - 写寄存器 FSHERSCON2=AAh；
  - 写寄存器 FSHERSCON1，设置 ERS=1，开始擦除操作；
  - 检测 ERSIF 是否为 1，如果为 1，表示擦除完成；否则擦除未完成，继续检测。
- 单字节编程流程说明：
  - 根据主晶振时钟频率，向寄存器 FSHTIMER 写入分频值；
  - 向 FSHWRADRH、FSHWRADRL 写编程地址，向 FSHWRDATA 写编程数据；
  - 写寄存器 FSHWRCON1，设置编程方式；单字节编程，设置 WREN=1，WRIF=0；
  - 写寄存器 FSHWRCON2=55h；
  - 写寄存器 FSHWRCON2=AAh；
  - 写寄存器 FSHWRCON1，设置 WR=1，开始编程操作；
  - 检测 WRIF 是否为 1，如果为 1，表示编程完成；如果要编程下一字节数据，重复 2-7 的流程。如果 WRIF 为 0，则编程未完成，继续检测。
- 多字节连续编程流程说明：
  - 根据主晶振时钟频率，向寄存器 FSHTIMER 写入分频值；
  - 向 FSHWRADRH、FSHWRADRL 写编程地址，向 FSHWRDATA 写编程数据；
  - 写寄存器 FSHWRCON1，设置编程方式；多字节连续编程，设置 WREN=1，WRSQU=1，WRSEQBIF=0，WRIF=0；
  - 写寄存器 FSHWRCON2=55h；
  - 写寄存器 FSHWRCON2=AAh；
  - 写寄存器 FSHWRCON1，设置 WR=1，开始编程操作；
  - 检测 WRSEQBIF 是否为 1，如果为 1，表示这一个字节编程已经完成，WR 由硬件自动清 0，可以编程下一地址；软件清零 WRSEQBIF 标志，写入新的地址和数据，并使 WR=1，开始下一地址的编程；如果 WRSEQBIF 为 0，则继续检测；
  - 如果完成了多个字节数据的编程，需要写 WRSQUSTOP=1 来停止连续编程方式；编程总线的停止时序完成后，硬件自动使 WRSQUSTOP=0，WRIF=1，WREN=0，多字节连续编程流程结束；
  - 在多字节连续编程方式，编程地址的高 10 位必须相同，因此每次最多只能连续编程 64 字节数据。每次连续编程的时间不能超过 8ms，超过 8ms 硬件将自动停止编程。

请软件人员注意！！

11.11 端口说明

11.2.1 功能说明

SC9351 最大支持 40 个 IO 端口，支持多种功能模块的输出/输入复用。

- 在读写片外数据存储器时，P0 扩展为低 8 位地址输出和 8 位数据输入/输出，P1 扩展为高 8 位地址输出，P1.6、P1.7 复用外部中断 Int6、Int7；复用外部中断 Int2；
- P2.4~P2.5 可扩展为外部中断输入 Int0~Int1；
- P10.0、P10.1 可扩展为 I^2C 的 SDA、SCL；
- P9.1、P9.2 可扩展为 UART0 的 TXD、RXD；
- P9.3、P9.4 可扩展为 UART1 的 TXD、RXD；
- P9.5、P9.6、P9.7 可扩展为 SPI 的 SDI、SDO、SCK；
- P0、P1、P2、P9 为 SMIT 输入 IO
- P10 为开漏端口
- 所有端口上电默认状态是高阻态（端口是输入状态，开漏无输出，上拉电阻关闭）

11.2.1 端口扩展控制

1. I²C、SPI、PWM 的扩展控制：

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOMUX（96H）	--	--	--	--	--	SPIEn	I²CEn	PWMEn

- SPIEn：SPI 端口扩展使能位，缺省为 0。SPIEn=1，P9.5、P9.6、P9.7 可扩展为 SPI 的 SPIIN、SPIOUT、SPICLK；SPIEn=0，P9.5、P9.6、P9.7 为通用 IO 管脚。
- I²CEn：I²C 端口扩展使能位，缺省为 0。I²CEn =1，P10.0、P10.1 可扩展为 I²C 的 SDA、SCL；I²CEn =0，P10.0、P10.1 为通用 IO 管脚。
- PWMEn：PWM 端口扩展使能位，缺省为 0。PWMEn =1，P1.0 可扩展为 PWM 输出；PWMEn =0，P1.0 为通用 IO 管脚。

2. 外部中断的扩展控制：

P2.4、P2.5、P1.6、P1.7 作为外部中断输入端口时，程序要设置相应的端口为输入，并且要设置外部中断控制寄存器 EXTINTENABLE（3AH）相应的使能位为 1。如果不作为外部中断输入端口，程序必须设置外部中断控制寄存器 EXTINTENABLE（3AH）相应的使能位为 0，避免产生错误中断信号。

3. 蜂鸣器输出控制：

程序打开**蜂鸣器**模块，P2.0 即作为**蜂鸣器**输出管脚。

4. UART 扩展控制：

UART模块提供TXD、RXD的端口控制信号，当相应控制信号有效时，P9.1、P9.2 即作为UART0 的TXD、RXD，P9.3、P9.4 即作为UART1 的TXD、RXD。模块UART0、UART1 的详细说明可参考“UART0”、“UART1”。

11.2.1 端口寄存器说明

- P0：在读写片外数据存储器模式，作为低 8 位地址输出和数据 IO。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P0（80H）	P0Reg [7：0]（端口寄存器）							
P0_IOConfig（6BH）	P0OE[7：0]（输入输出控制寄存器，1 为输出，0 为输入，缺省为 0）							
P0PU（6AH）	P0PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，，缺省为 0）							
P0OD（69H）	P0OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							

- P1：在读写片外数据存储器模式，作为高 8 位地址输出。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P1（90H）	P1Reg [7：0]（端口寄存器）							
P1_IOConfig（66H）	P1OE[7：0]（输入输出控制寄存器，1 为输出，0 为输入，缺省为 0）							
P1PU（65H）	P1PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P1OD（64H）	P1OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							

- P2：在读写片外数据存储器模式，作为 ALE 和读写控制信号。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P2（A0H）	P2Reg [7：0]（端口寄存器）							
P2_IOConfig（D4H）	P2OE[7：0]（输入输出控制寄存器，1 为输出，0 为输入，缺省为 0）							
P2PU（60H）	P2PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P2OD（5FH）	P2OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							

- P5、P6、P7

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P5PU（56H）	P5PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P5OD（55H）	P5OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							
P6PU（54H）	P6PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P6OD（53H）	P6OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							
P7PU（52H）	P7PU[7：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P7OD（51H）	P7OD[7：0]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							
IOMUX5L(E6H)	高四位控制 P5[3:0]输入输入，置 0：输出，置 1：输入。低 4 位必须置 0							
IOMUX5H(E7H)	高四位控制 P5[7:4]输入输入，置 0：输出，置 1：输入。低 4 位必须置 0							
IOMUX6L(E3H)	B[6:4]控制 P6[2:0]输入输入，置 0：输出，置 1：输入。其它位必须置 0							
IOMUX7H(EBH)	高三位控制 P7[7:5]输入输入，置 0：输出，置 1：输入。其它必须置 0							

- P9：可扩展为模块 SPI、UART0、UART1 的管脚。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P9(C0H)	P9Reg [7：1]（端口寄存器）							
P9_IOConfig(4DH)	P9OE[7：1]（输入输出控制寄存器，1 为输出，0 为输入，缺省为 0）							
P9PU(4CH)	P9PU[7：1]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P9OD(4BH)	P9OD[7：1]（输出开漏控制寄存器，置 0：开漏输出，缺省为 1）							

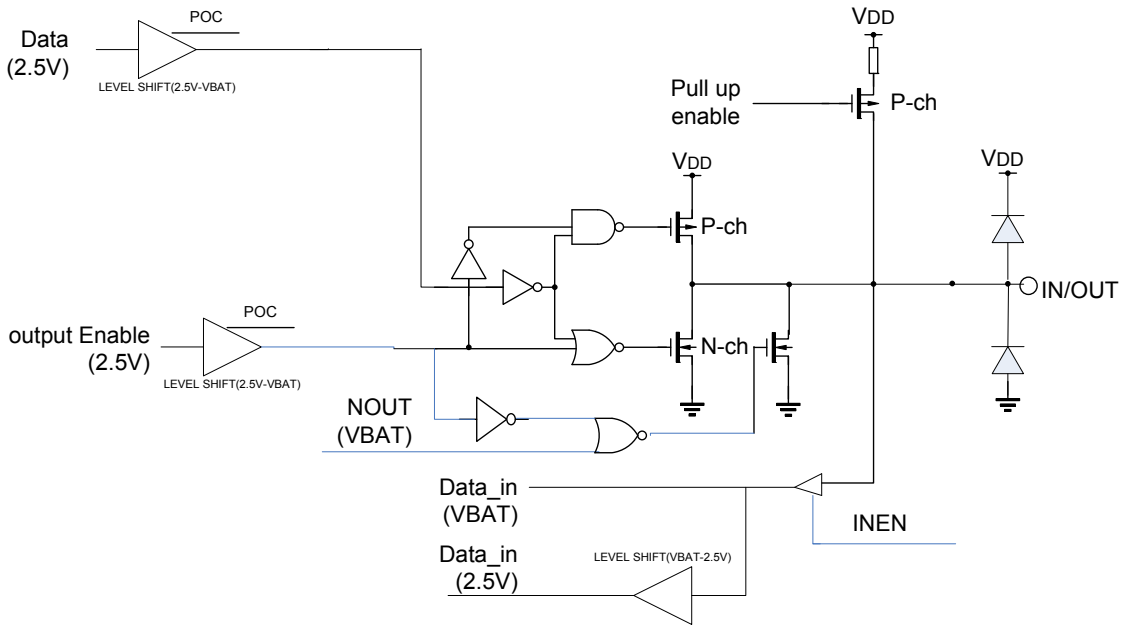
- P10：可扩展为模块 I²C 的管脚。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P10(49H)	P10Reg [1：0]（端口寄存器）							
P10_IOConfig(48H)								
P10PU(47H)	P10PU[1：0]（上拉控制寄存器，1 上拉有效，0 上拉无效，缺省为 0）							
P10OD(46H)								

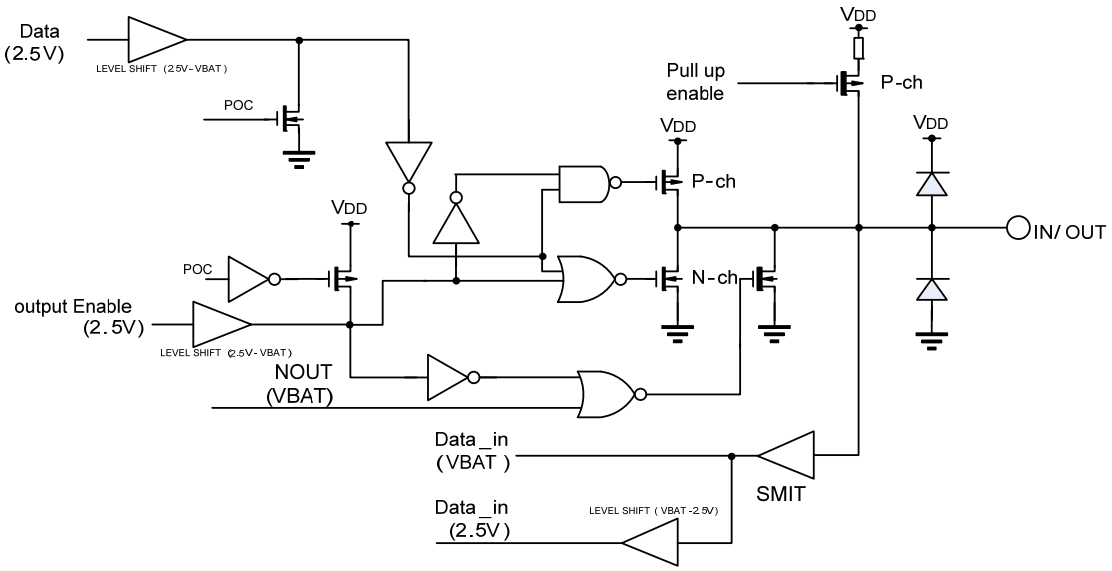
11.2.1 IO 端口结构：

在待机状态（2.5v 电源关闭）下 I/O 为输入状态，可编程将 PAD 设置成上拉或者下拉状态，实现按键扫描输出功能或其他输出功能。

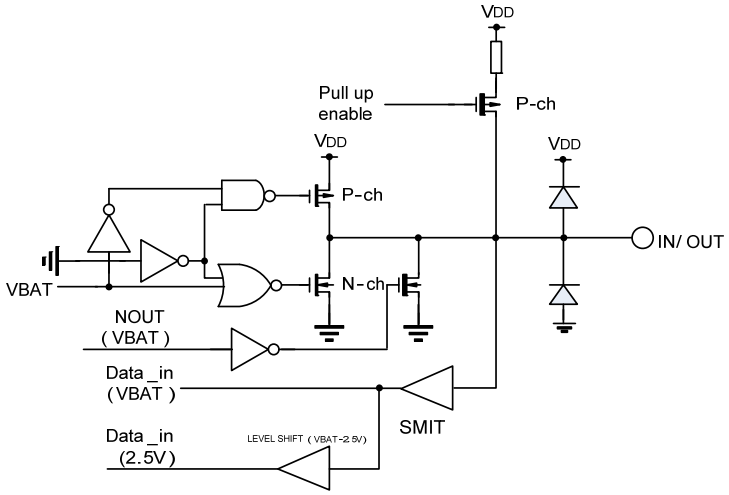
1. IO 端口（P5，P6，P7）



2. IO 端口（P0，P1，P2，P9）


Data
(2.5V)
LEVEL SHIFT (2.5V-VBAT)
POC
VDD
Pull up
enable
P-ch
VDD
VDD
P-ch
IN/ OUT
POC
VDD
N-ch
output Enable
(2.5V)
LEVEL SHIFT (2.5V-VBAT)
NOUT
(VBAT)
Data_in
(VBAT)
SMIT
Data_in
(2.5V)
LEVEL SHIFT (VBAT-2.5V)

3. 开漏（P10）



11.12 RTC

实时时钟电路由 75k 晶振的 2 分频时钟驱动，提供年、月、日、星期、小时、分钟、秒的时钟和日历功能，具有闰年自动识别切换功能。提供定时闹钟功能，可以在设定的星期、日、小时、分钟时发出闹钟中断，可以通过对应的闹钟控制位来关闭或启动某一闹钟功能。

具有硬件的秒时钟调整功能，结合温度~晶振频率曲线，可以有效调整计时偏差，可以使计时精度达到 0.5ppm。

在待机状态，RTC 需要保持工作状态，所以 RTC 模块由电池供电。

	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
TIMER（7FH）	定时器预设值寄存器							
TMCON（7Eh）	--	--	--	--	--	TE	TD1	TD0
WK_ALARM(7CH)	--	--	--	--	WAE	0~6		
DAY_ALARM（7BH）	--	DAE	01~31BCD 码格式数					
HR_ALARM（7AH）	--	HAE	00~23BCD 码格式数					
MIN_ALARM（79H）	MAE	00~59BCD 码格式数						
YEARL（78H）	00~99 BCD 码格式数							
MON（77H）	leap	--	--	01~12BCD 码格式数				
WEEK（76H）	--	--	--	--	--	0~6		
DAY（75H）	--	--	01~31BCD 码格式数					
HOUR（74H）	--	--	00~23BCD 码格式数					
MIN（73H）	--	00~59BCD 码格式数						
SEC（72H）	--	00~59BCD 码格式数						
YEARH（71H）	00~99BCD 码格式数							
RTC_CS（70H）	--	--	--	TI/TP	AF	TF	AIE	TIE
RTC_Ctrl（6FH）	--	--	--	--	--	--	SCtrl<1：0>	

SCounterLoadH(6EH)	秒计时周期调整寄存器
SCounterLoadL(6DH)	秒计时周期调整寄存器

- **日历功能**：提供时钟/日历功能，编码为 BCD 格式，闰年自动识别，提供闰年标志位 Leap；程序可以直接读写寄存器，能够很方便的完成时钟/日历的设置和读取。
- **定时器功能**：提供一个 8 位定时器，定时器加 1 到预设值时产生溢出脉冲，置位 TF 标志位，如果中断使能位 TIE 为 1 则产生中断信号；TF 只能软件清零。
  - TE：定时器使能信号，为 1 时，定时器工作。
  - TD1~TD0：定时器定时时钟选择信号：
    00：4687Hz
    01：73Hz
    10：1Hz
    11：75K/2
  - TF：定时器定时溢出标志位。
    1：定时器定时溢出。
  - TIE：定时器中断使能位。
    1：定时器溢出可以触发中断。
  - TI_TP：定时器中断触发信号选择位。
    0：定时器溢出脉冲触发中断，定时器每次溢出都会触发中断。
    1：TF 信号触发中断，则在软件清零 TF 后才能触发下一次中断。
- **闹钟功能**：提供星期、日、小时、分钟的闹钟功能，当设置某个闹钟使能位为 0 时，则相应的闹钟条件有效。当日历/时钟数据符合闹钟条件时，则会置位闹钟标志位 AF，如果中断使能位 AIE 为 1 则产生中断信号。闹钟标志位 AF 软件清零。
  - AF：闹钟标志位。
  - AIE：闹钟中断使能位。
  - **WK_ALARM**[3：0]：星期闹钟寄存器，WAE 为 1 屏蔽星期闹钟功能。
  - **DAY_ALARM**[6：0]：日闹钟寄存器，DAE 为 1 屏蔽日闹钟功能。
  - **HR_ALARM**[6：0]：小时闹钟寄存器，HAE 为 1 屏蔽小时闹钟功能。
  - **MIN_ALARM**[7：0]：分钟闹钟寄存器，MAE 为 1 屏蔽分钟闹钟功能。

  注意：AIE、TIE 不能同时设置为 1。
- **硬件秒时钟调整功能**：集成了数字式时间调整电路，通过内部的高精度时间调整电路，可以有效调整由于晶振频率随环境温度变化而导致的时钟走时偏差；也可以调整由于晶振固有偏差而导致的时钟走时偏差。
  - SCtrl<1：0>：时钟调整周期选择位：
    00：10（秒），调整精度约±2.6（ppm）
    01：20（秒），调整精度约±1.3（ppm）
    10：40（秒），调整精度约±0.66（ppm）
    11：60（秒），调整精度约±0.44（ppm）

  可以根据不同的时钟走时精度要求，选择适当的时钟调整周期。时钟调整电路原理如

下：时钟调整是在每个时钟调整周期内，根据预先设置的数据改变被调整的秒周期的计数脉冲个数，从而弥补在这个时钟调整周期内时钟走时的偏差，未被调整的秒周期的计数脉冲为37500 个。例如使用 75000Hz 晶振，由于晶振温漂而使振荡频率为 75004Hz：

如果时钟调整周期选择为 20 秒，RTC 在 20 秒内准确的计数脉冲应该为 750040 个，则被调整的秒周期的计数脉冲个数应设置为：37500+（37502-37500）x20=37540（92A4h）。则在每 20 秒内的某个秒时刻（例如 00、20、40），计数脉冲个数为 37540，这样每 20 秒内的计数脉冲就为：（37500x19）+37540=750040，RTC 的时钟走时就很准确。

如果每 20 秒补偿 1 个时钟脉冲，则时钟走时精度为 1/750000=±1.3ppm。不过这种方法只是对时钟走时进行调整，并不对晶振的振荡频率进行调整。

11.13 蜂鸣器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BUZCR（BDH）	--	--	--	--	--	BuzOutEn	BUZ_Sel1	BUZ_Sel0

- BuzOutEn：**蜂鸣器**使能控制位，缺省为 0。
  - 1：打开**蜂鸣器**，P2.0 扩展为**蜂鸣器**输出；
  - 0：关闭**蜂鸣器**。
- BUZ_Sel[1:0]：**蜂鸣器**输出频率控制位，缺省为 0h。
  - 00：输出频率为 1KHz；
  - 01：输出频率为 3KHz；
  - 10：输出频率为 5KHz；
  - 11：输出频率为 12.5KHz；

注意：使用 BUZ 功能，首先要讲 P2.0 口设置为输出，并将输出寄存器设置为 1

12. 开发调试

复位时，检测到 nDBG 脚为低，那么进入调试模式。电路通过串行接口与上位机通讯。调试时需要给电路提供一个独立的时钟（因开发工具而异）。

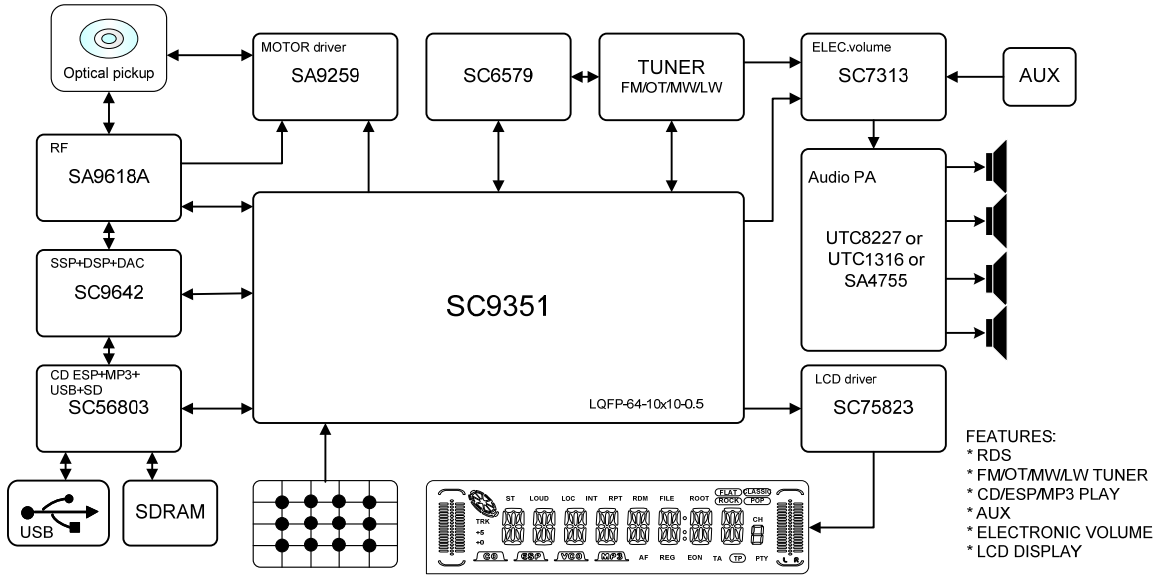
程序空间的低 4k 空间已经被调试程序占用，这部分程序主要功能包括快速下载用户程序代码或实现在应用编程功能。所以中断矢量需要重新映射，具体参见 51 系列产品调试用户手册。

用户程序下载到 FLASH 后即可进行调试，**SC9351** 共支持 4 个硬件断点。在单步运行的 debug 模式 WDT 关闭，其他模块均能正常工作。

启用在系统调试功能时至少需要占用 P7.5~7.7 这几个 IO，各管脚对应功能如下

管脚名称	Debug 信号	说明
P7.7	SetClk	调试时钟输入
P7.5	SerilIn	调试数据输入
P7.6	SerilOut	调试数据输出

13. 典型应用电路图


Optical pickup
MOTOR driver
SA9259
SC6579
TUNER
FM/OT/MW/LW
ELEC.volume
SC7313
AUX
RF
SA9618A
Audio PA
UTC8227 or
UTC1316 or
SA4755
SSP+DSP+DAC
SC9642
SC9351
LQFP-64-10x10-0.5
CD ESP+MP3+
USB+SD
SC56803
LCD driver
SC75823
USB
SDRAM
FEATURES:
* RDS
* FM/OT/MW/LW TUNER
* CD/ESP/MP3 PLAY
* AUX
* ELECTRONIC VOLUME
* LCD DISPLAY

14. 封装外形图

LQFP-64-10x10-0.5 Unit: mm